22/2/25

Data Representation & Arithmetic Algorithms

Floating Point Representation

- 23.678 this is a decimal no. with a fractional and then integer part.
- Similarly a very large or a very small number can be represented using mantissa and the exponent part.

Mantissa.

$12345567.12 = 1.234566712 \times 10^{8}$ ← Exponent

$-0000000123 = -123 \times 10^{10}$

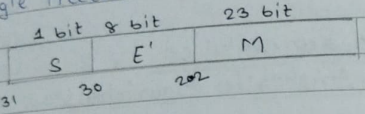
- It can be seen that in position of the radix point in the number varies according to the significant digit of the number but still the computer must be able to adjust automatically hence they are known as floating point numbers.
- In case of numbers where position of the radix remain same they are turn as fixed point number.
- The decimal number shown above are adjusted to have only one significant digit before the radix point these number are set to be normalised.

- Exponent indicate the current decimal location. number after radix point are called mantissa.

IMP: ANSI/IEEE

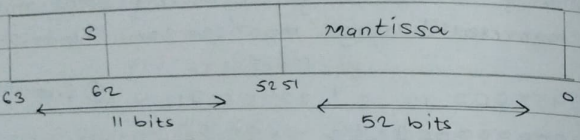
- It is standard for floating point binary numbe

1) Single Precesion Format of IEEE (32 bits)



$E' = E + 127$

2) Double Precesion Floating Point format (64 bit)



$E' = E + 1023$

Q. Represent $(1259.125)_{10}$ in single and double precesion format.

→ Step 1:

2	1259
2	629 - 1
	314 - 1
	157 - 0
	78 - 1
	39 - 0
	19 - 0
	9 - 1

$(10011101011)_2$

$0.125 \times 2 = 0.250$

$0.250 \times 2 = 0.50$

$0.50 \times 2 = 1.00$

(001)

$\therefore (10011101011.001)_2$

Step 2: Normalizing the above number

$1.0011101011001 \times 2^{10}$

Step 3: Representing single precesion

S = 0 E = 10 M = 00111010 11001

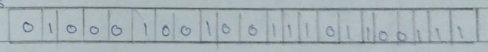
$E' = E + 127$

$= 10 + 127$

$= 137$

$\therefore E' = 10000010 \; 10001101$

2	137
2	68 - 1
2	34 - 0
2	17 - 1
2	8 - 1
2	4 - 0
2	2 - 0
	1 - 0



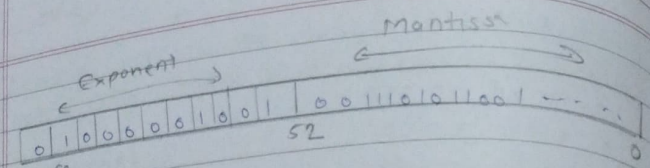
Step 4: Double Precesion

S = 0 E = 10 M = 00111010 11001

$E' = E + 1023$

$= 10 + 1023$

$= 1033$



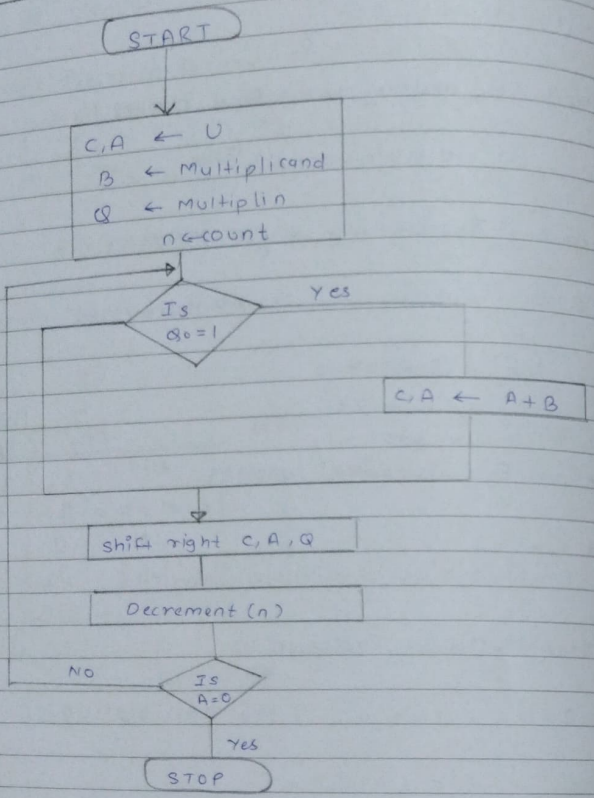


Fig: Multiplication Algorithm

B = 0100

C	A	Q	Count	Remarks
0	0000	$q_3q_2q_1q_0$ 0100	4	Initialization
	0100			Add
0	01000	0101		Add ~~shift right~~
0	00100	010(0)	3	shift right
0	0001	000(1)	2	Shift right
	0100			
0	01100	01		
0	00110	0	1	Shift right
0	00011	00	0	Shift right
	Result			

-:- Drawback

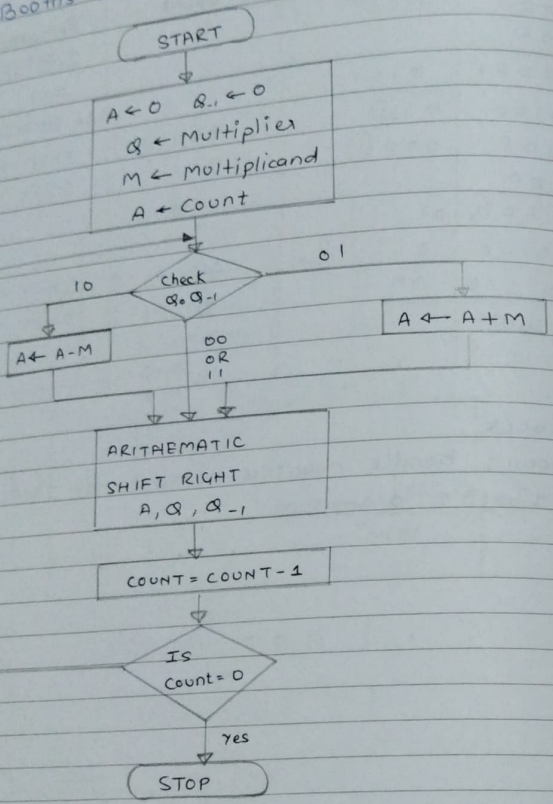
- It can't handle negative number for that we use booth's algorithm

10m

26/2/25

V.V.V Imp

* Booth's Algorithm



Eg: Multiply 5^{π} and 2 using booth's algorithm

1) → 5 × 2

M = 0 1 0 1 Q = 0 0 1 0

1 0 1 0

-M = 1 0 1 1

A	Q	Count	Remarks
0 0 0 0	0 0 1 0 0	4	Initial
0 0 0 0	0 0 0 1 0	3	Right Shift
1 0 1 1			
1 0 1 1	0 0 0 1 0		A ← A-M
1 1 0 1	1 0 0 0 1	2	Shift Right
0 1 0 1			
0 0 1 0	1 0 0 0 1		A ← A+M
0 0 0 1	0 1 0 0 0	1	Shift Right
0 0 0 0	1 0 1 0 0	0	Right Shift

Result 8421

2) Multiply the numbers given below.

→ 5 × -2

M = 0 1 0 1 Q = 1 1 1 0

A	Q	Count	Remark
	$Q_3\ Q_2\ Q_1\ Q_0\ Q_{-1}$		
0000	1 1 1 0 0	4	Initial
0000	0 1 1 1 0	3	SR
1011			A ← A-M
1011	0 1 1 1 0		
1101	1 0 1 1 1	2	SR
1110	1 1 0 1 1	1	SR
1111	1 0 1 1 0	0	SR

→ 2's complement form

Result

0 0 0 0 1 0 0 1

1

0 0 0 0 1 0 1 0

3) Multiply -5 x 2 using booth's algorithm.

→

-5 x 2

M = 0 1 0 1

1 0 1 0

+1

1 0 1 1

Q = 0 0 1 0

A	Q	COUNT	REMARK
	$Q_3\ Q_2\ Q_1\ Q_0\ Q_{-1}$		
0 0 0 0	0 0 1 0 0	4	Initial
0 0 0 0	0 0 0 1 0	3	SR
0 1 0 1			A ← A-M
0 1 0 1	0 0 0 1 0		
0 0 1 0	1 0 0 0 1	2	SR
1 0 1 1			
1 1 0 1	1 0 0 0 1	1	A ← A+M
1 1 1 0	1 1 0 0 0	1	SR
1 1 1 1	0 1 1 0 0	0	SR

→ 2's complement form

Result

0 0 0 0 1 0 1 0

4) -5 x -2

→

-5

M = 0 1 0 1

1 0 1 0

+1

1 0 1 1

-2

Q = 1 1 1 0

A	Q: Q_3	Q_2	Q_1	Q_0	Q_{-1}	COUNT	REMARK
0 0 0 0	1	1	1	0	0	4	Initial
0 0 0 0	0	1	1	1	0	3	SR
0 0 0 0 0 1 0 1 0 1 0 1	0	1	1	1	0		A ← A-M
0 0 1 0	1	0	1	1	1	2	SR
0 0 0 1	0	1	0	1	1	1	SR
0 0 0 0	1	0	1	0	1	0	SR

2' complement form → 0 0 0 0 1 0 1 0 1

Result

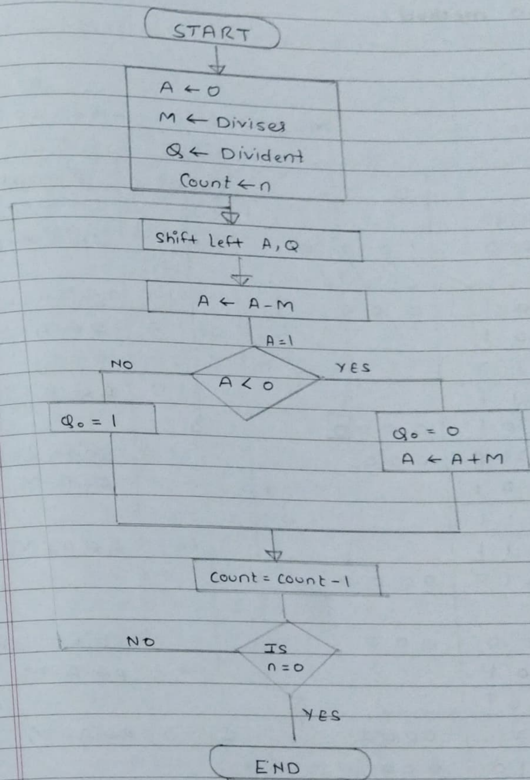
-:- Division of Integers

Eg: 219 → 11011011
6 → 110

```
        100~~0~~100
110 ) 11011011
      110↓↓↓
      000110
         110
      00011
```

4/3/25

-:- Restoring division algorithm



Note: M should always be 1 bit greater than Q

Q. Dividing the following unsigned numbes using restoring division methed.

→ 8 ÷ 3

Q = 1000 M = 00011 -M = 11101

A	Q	count	Remarks
00000	1000	4	Initialization
00001	000		Shift Left
11101			A ← A-M
11110			
00011			A ← A+M
00010	0000	3	Shift Left
11101	000		A ← A-M
11111			
00011			A ← A+M
00010	0000	2	
00100	000		Shift Left
11101			A ← A-M
00001			
00010	0001	1	A ← A+M
00101	0010		
11101			
11111			
00011			
00010	0010	0	

2> 17 ÷ 3

Q = 10001 M = 00011 -M = 11101

A	Q	count	Remarks
00000	10001	5	Initialization
00001	0001		Shift Left
11101			A ← A-M
11110			
00011			A ← A+M
00001	00010	4	
00010	0010		Shift Left
11101			A ← A-M
11111			
00011			A ← A+M
00010	00100	3	
00100	0100		Shift Left
11101			A ← A-M
00001			A ← A+M
00001	01001	2	
00010	1001		Shift Left
11101			A ← A-M
11111			A ← A+M
00011			
00001	01001 0	1	
00010	1010		Shift Left
11101	1		A ← A+M
00001	0	0	
00010	00010		

Imp 5 mks

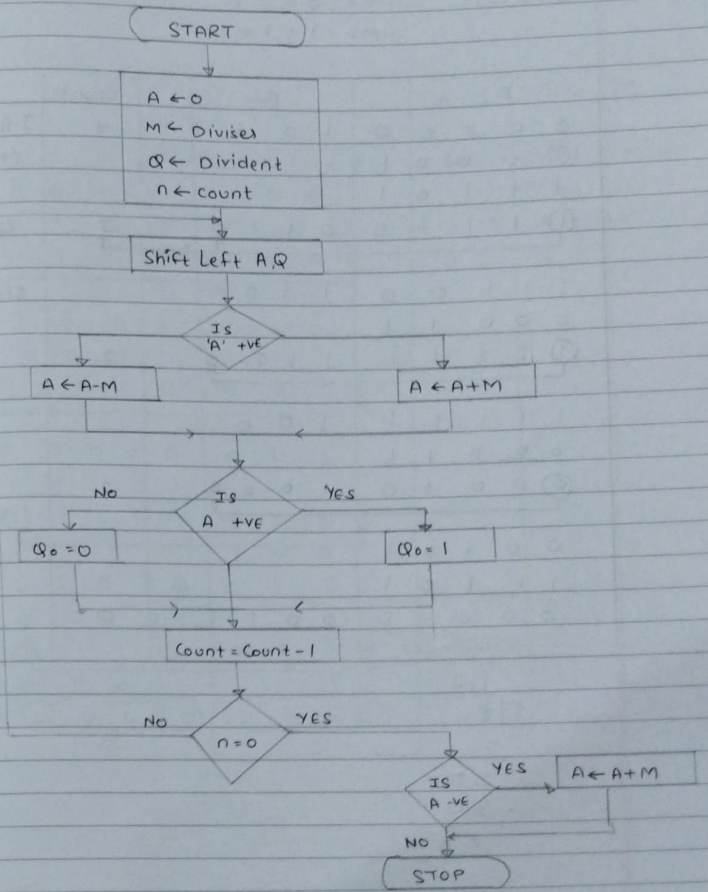
Q. Differentiate / Compare restoring and non-restoring D.M.

→

Restoring D.M	Non-Restoring D.M
1) Needs restoring of register A if the result of substraction is negative.	Does not need restoring.
2) In each cycle content of reg A is shifted Left then divisor is substracted from it.	In each cycle content of A is first shift left then divisor is added or subtracted with content of A depending upon sign of A.
3) Does not require restoring of remainder.	Needs restoring of remainder if remainder is negative.
4) Restoring is slower algorithm.	Non-Restoring is faster algorithm.

6-3-25

NON-RESTORING DIVISION



Eg:- Divide 11 by 3 using non-restoring division method.

→ Q = 1011 M = 00011

-M = 11100

A	Q	count	Remarks
00000	1011	4	Initialization
00001			shift Left A,Q
11101			
11110	0110	3	$Q_0 = 0$
11100	110		shift left
00011			
11111	1100	2	
11111	100		
00011			
00010	1001	1	
00101	001		
11101			
00010	0011	0	
Remainder	Q		

1.6

5) 8

-5

30

-30

00

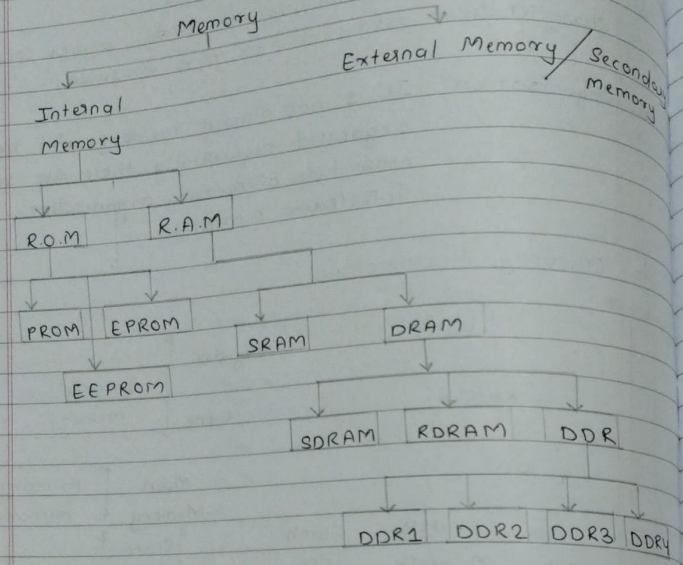
Eg:- Divide 12 by 3 using non-restoring division method.

→ Q = 1100 M = 00011

-M = 11100

A	Q	count	Remark
00000	1100	4	Initialization
00001	100		Shift Left
11101			A ← A-M
11110	1000	3	$Q_0 = 0$
11101	000		
00011			
00000	0001	2	
00001			
11101			
11110	0010	1	
11101			
11011	0100	0	
00011			
11101			
00000	0100		
R	Q		

-/- Memory Classification.



19/3/25

-/- Difference between SRAM and DRAM

SRAM	DRAM
Static RAM	Dynamic RAM
FlipFlop	CAPACITOR
Not Requires Refreshing clk	Requires Refreshing clk.
Bit density is less.	Bit density is more.
FAST	SLOW
costly	Cheapest
Used in Cache	Used in main memory.

-/- Direct Mapping

$k \bmod n$

↑ Block no ↑ Line no

In cache memory word is accessed by

3	2	2
TAG	LINE NO	BLOCK OFFSET

< 7 >

0 mod 4 = 0
1 mod 4 = 1
⋮
4 mod 4 = 0
5 mod 4 = 1

TAG tells us which block stored in the cache line

Associative Mapping:

W_0, W_1, W_2 B_0
W_3, W_4, W_5 B_1
B_2

128 words / 4

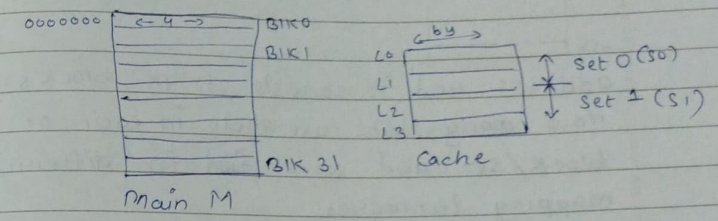
32 block

W_2, W_1, W_1 B_{31}

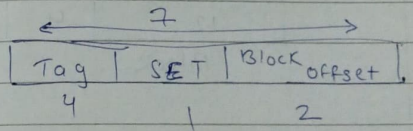
$\frac{7}{2} = 128$

- Advantage:-
  1) No Conflict Miss
  2) Tag followed by a block offset
- Drawback:-
  Time Consuming

Set Associative Mapping:



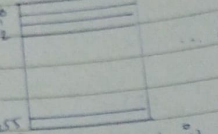
K set associative mapping
2 set
4 set

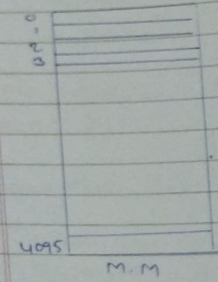


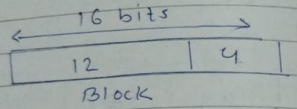
- Advantage:-
  1) Better Flexibility
  2) Search is Limited
- Disadvantage:-
  1) Costly

ample:-

nsider a cache consisting of 256 blocks of 16 words each for a total of 4096 words and assume that main-m is addressable by 16-bit address and it consists of 4K blocks. How many bits are there in each of tag, block/set and words field for different mapping techniques.





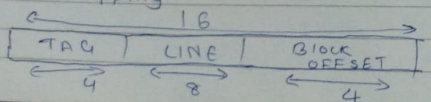


Direct Mapping

Since there are 16 words

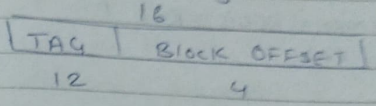
$\therefore$ word = 2^4 = 4 bits

Direct Mapping



Cache $= \frac{256}{2} = 2^8 = 8$

Associative Mapping



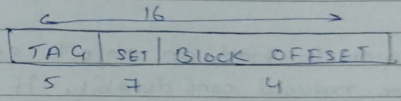
SET ASSOCIATIVE MAPPING

Assuming 2 way set associative

$\therefore$ No. of set $= \frac{256}{2} = 128$

$\therefore$ $2^7 = 128$

7 bits

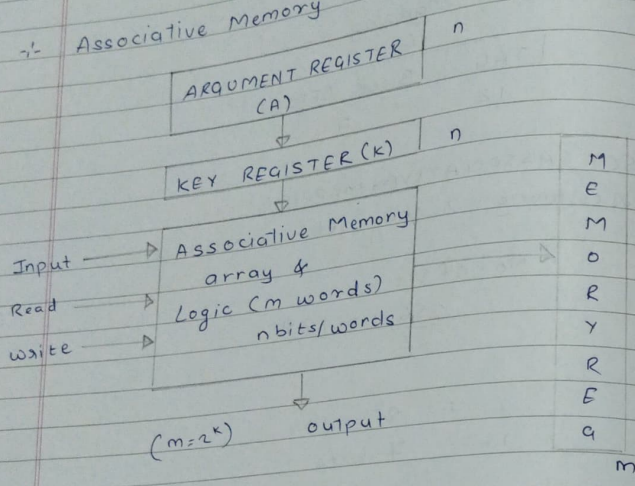


-:- Cache Coherancy :-

Whenever a new block is brought into the cache the old block is flashed into memory. This is done by checking a bit known as 'Dirty Bit'/'Altered Bit' or changed Bit. If this bit is ON it indicate cache block is altered so the block is to be coppied

back into the memory.
There are times when two copies [one in cache / one in m.m] are not the same this inconsistency is known as cache coherance problem.

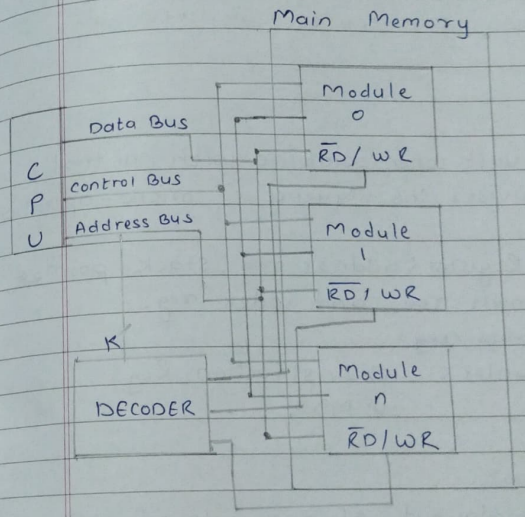
Associative Memory



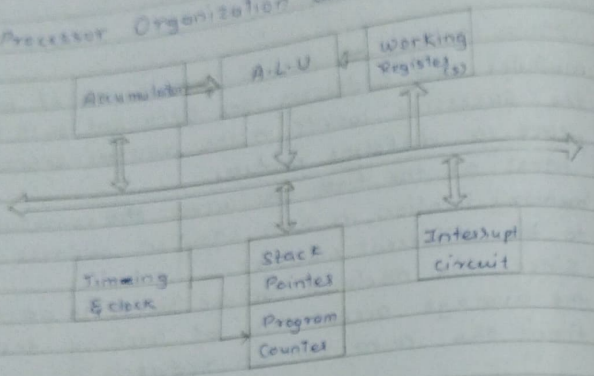
- The time require to find an object stored in memory can be reduced considerably if objects are selected based on their contents not on their locations.
- A memory unit accessed by the content is called as associative memory.
- This type of memory is accessed simultaneously and in parallel.

Interleaved Memory

- Main Memory is divided into a no. of memory modules and addresses are arranged such that the sucessive words in the address space are place in different modules.
- Most of the time CPU access the consecutive in memory location in such situation address will be to the different modules.
- since this modules can be access in parallel the average access time of fetching the word from the m.m can be reduced.



-:- Processor Organization and architecture



1) Fetch 2) Decode 3) Execute ... Instruction Cycle

MAR - Address

MBR - Buffer

- The processor unit when combine with control unit that supervises the sequence of micro-operation is called CPU.
- General Purpose Register; address reg, stack pointer, data reg, segment reg, and index reg.
- Control and Status reg:
  1) Program Counter 2) Instruction Reg
  3) MAR 4) MDR

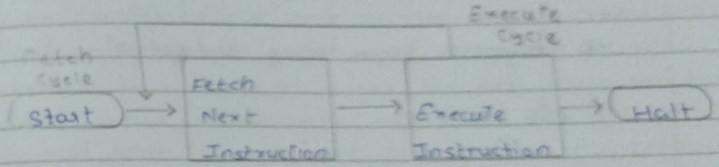
-:- Instruction Format.

- It defines the layout of the bits of an instruction.
- An instruction format must include an opcode and zero or more operands in one of the addressing modes

Instruction Class	Example
Three Address	ADD R_1, R_2, R_3
Two Address	ADD R_1, R_2
One Address	ADD R_1
Zero Address	ADD

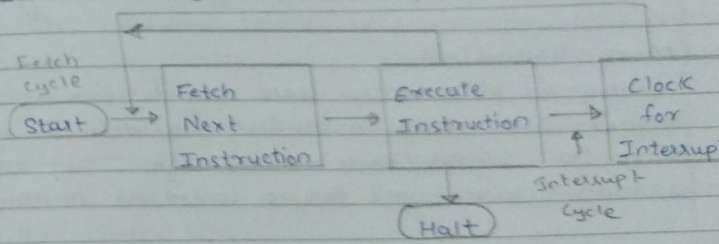
2nd

• Basic Instruction Cycle:-



V. Imp

• Instruction Cycle with Interrupt:-



27/3/25

-:- Instruction & Instruction Sequence

It must be capable of performing following operations.

1) Data Transfer between the memory and processor registers:
- Processor registers are $R_1, R_2, R_3, \ldots$
- Address of memory location: Location Place Memory
- IO reg are represented by 'data in' and 'data out'
- Content of register or memory location are denoted by '[ ]'

Eg:- $R_2 \leftarrow [loc]$

2) Perform arithmetic / logic operation on data:

3) Program sequencing and control.

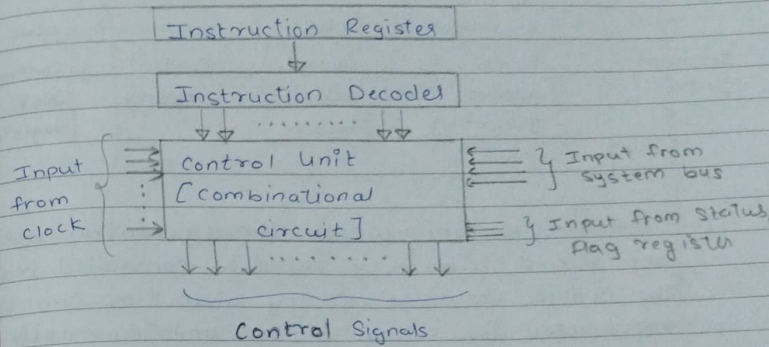
4) I/O control.

> Micro-operation:-
Operations executed on data stored in registers.

> Micro-Instruction:-
Every bit of a control word stands for a control signal. Each micro-instruction is defined by a set of control signals these control signals are encoded in a control world and this is called as micro-instruction

> Micro-Program:-
A sequence of micro-instruction is known as micro-program also called firmware.

-:- Hardware Control Organization



The control unit is implemented using hardware [gates, flipflops etc], we need a special digital circuit that uses as inputs the bits from the opcod field in over instructions, bits from the flag reg, signals from the bus, and signal from the clock. It should produce as output the control signal to drive various components in the computer.

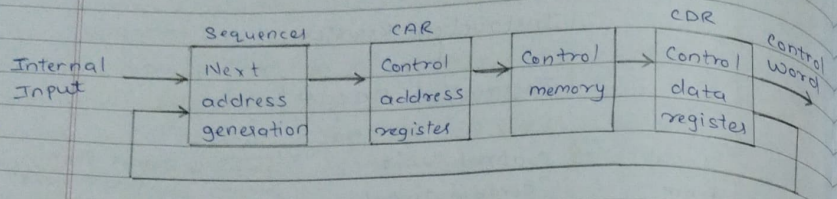
In hardware control unit we physically connect all of the control lights to the actual machine instruction.

The instructions are divided up into fields a different bits in the instructions are combine through various digital logic components to driv the control line.

Advantage of hardwired control unit is:
1) Very fast

Disadvantage:
1) Difficult to design and modify.

:- Micro-Program Control Organization



A control unit whose binary control variables are stored in memory called as controlled memory. The control variables at any given time can be represented by strings of One and Zeros called as control word. Each word in control memory contains micro-instruction within which spacifies one or more micro-operations for the Systems.

A sequence of micro-instruction consists a micro-program.

The control memory is assumed to be wrong within which all control info are stored permenantly.

The CAR specifies address of the micro-instruction and CDR holds micro-instruction read from the memory, the micro-instruction contains a control word that specifies one or more micro-operations. Once the operations are executed control must determine the next address, the next address may also be a function of external input condition.

Advantages:-
No hardware change

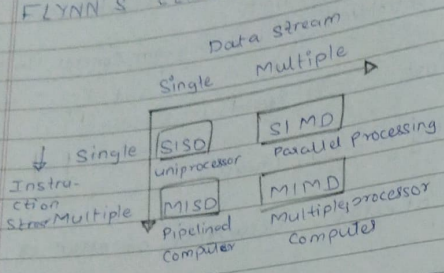
Disadvantage:-
Slow

:- Difference Between hardwired control unit and micro-program control unit.

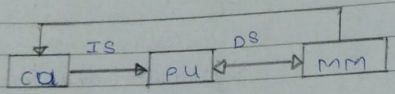
	Hardwired Control Unit	Micro-program Control Unit
1)	Speed is fast	Speed is slow
2)	More Costly	Cheap
3)	Occurrence of error is more	Occurrance of error is less.
4)	Control functions implemented in hardware	Control functions implemented in Software.
5)	Not flexible	More flexible.
6)	Difficult to handle complex instruction set.	Easy to handle complex
7)	Complicated Design	Systematic and Simple design
8)	Complex decoding & sequencing logic.	Easies decoding &
9)	More chip area	Less chip area.
10)	Used in RISC processors.	Used in main frames.

08/4/25

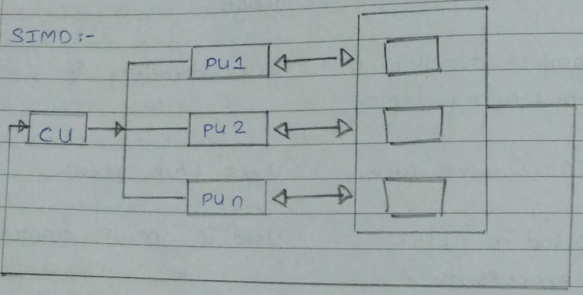
Imp :- FLYNN'S CLASSIFICATION



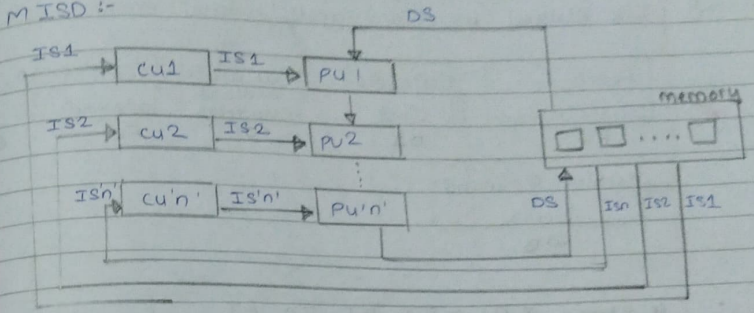
- Instruction Stream :-
  Collection of instruction read from the memory
- Data Stream :-
  Result of the action done on the data
- SISD :-



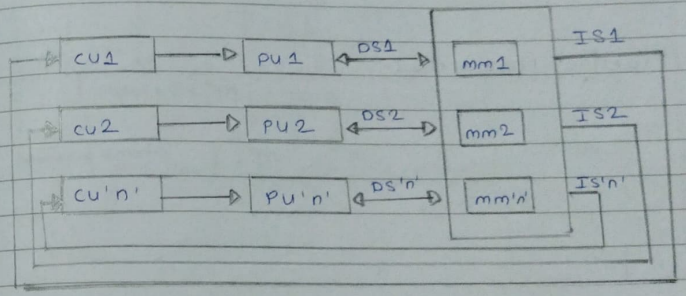
- SIMD :-



- MISD :-



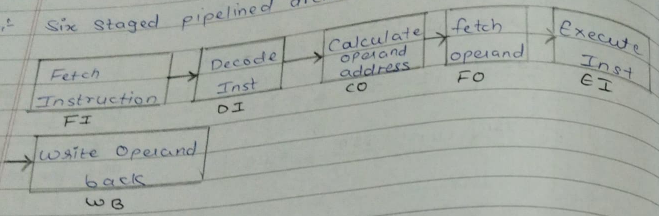
- MIMD :-



* Instruction Pipelining :-
  It is a technique used to design computer to increa their instruction throughput [no. of instruction executed p unit time].
  The instruction pipelining a stream of instruction can executed in a overlapping manner.
  Instruction pipelining is used to achieve instruct level paralysim within a single processor.

9/4/25

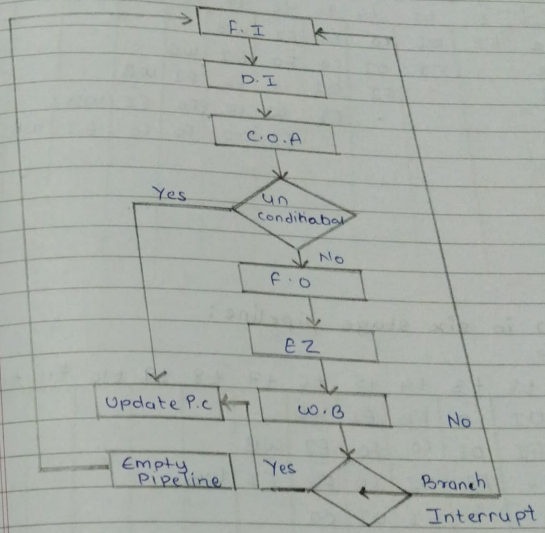
-:- Six Staged pipelined architecture



- 4 Stage P.A → FI, DI, FO, EI
- 5 stage P.A → FI, DI, FO, EI, WB

-:- Pipeline Hazards :-

-:- Flow Chart :-



-:- Six Stage Pipelined processor timining diagram:

Instruction ↓ / Time →	t_1	t_2	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}
1	FI	DI	CO	FO	EI	WB						
2		FI	DI	CO	FO	EI	WB					
3		-	FI	DI	CO	FO	EI	WB				
4			-	FI	DI	CO	FO	EI	WB			
5				-	FI	DI	CO	FO	EI	WB		
6												
7												
8												
9												

-:- Branch in six stage pipeline:

Instruction ↓ / Time →	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}
1	FI	DI	CO	FO	EI	WB							
2		FI	DI	CO	FO	EI	WB						
3		-	FI	DI	CO	FO	EI	WB					
4			-	FI	DI	CO							
5				-	FI	DI	CO	FO					
6					-	FI	DI						
7						-	FI						
8								FI	DI	CO	FO	EI	WB
9								-	FI	DI	CO	FO	EI