C. V. Raman Polytechnic, Bhubaneswar

STUDY MATERIAL

ON

COMPUTER SYSTEM ARCHITECTURE

3RD SEMESTER, COMPUTERSCIENCE ENGINEERING

Prepared By

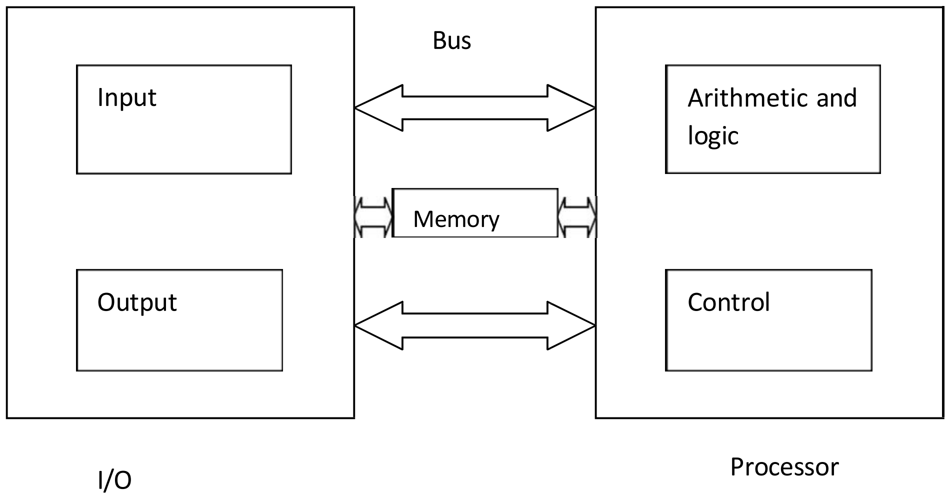
SOUMYA SAHOO, Asst.prof., CSE,CVRP

CONTENTS

<u>**CHAPTER-1**</u>

BASIC STRUCTURE OF COMPUTER HARDWARE

Computer is a machine which accepts input information in the digitized form, process the input according to a set of stored instruction and gives an output in form understandable by humans.



Function unit : computer consist of

1. Memory unit
2. General add special purpose registers.
3. Control circuits consisting of flip flops, decoders
4. Common data and address pulse.

Input unit:- Computer accept coded information through input unit and delivers the output through output unit.

Memory unit:-memory unit used to store programs as well as data. It may be classified into primary storage and secondary storage.

Primary storage:- It is a memory mode of up semi conductor storage cells. The these cells are grouped together in a fixed size called word. The word may be 16-bit to 64-bits.

Secondary storage:- It is used to store large amount of data add program (hard disk, floppy disk, magnetic disk, optical disk).

CPU:-

The ALU and CU together form the control processing unit. It is known as processor (The computer performs the following functions.

⇒ Accepts program and data through input unit and store them in memory.

⇒ The store data are processed by the ALU under program control.

⇒ The process information is delivered through the logic and output unit.

⇒ The program that is to be executed is stored in memory the CPU then fetches the instructions from the memory one after another and program the required operation given by the instructions.

The data are number characters used as operands by the instructions.

Pulse:- Pulse is a single which carry some information CMOS- Complementary metal oxide semiconductor.

ASCII-American standard code for information and interchange.

Bit		X	Y	X+Y
Byte	A.D.	T	1	1
KB	Log	0	1	1
MB	Rel	1	0	1
GB	Boolen	0	0	0
TB				

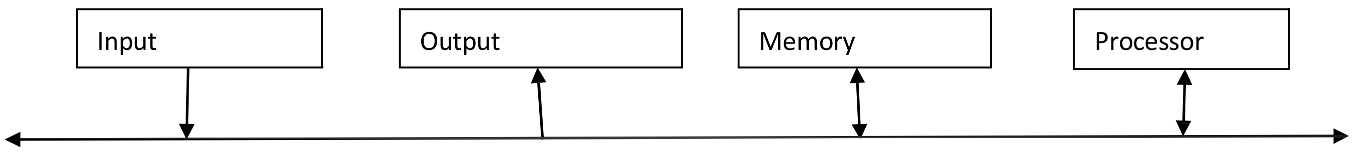
CSA

Computer components:-

Computer consists of central processing unit, main memory and the input output components. Those computer are inter connected in order axid the basic function of the computer.

The main memory stores both data and instruction in binary format. Every memory location can be individuals address instructions are educated one after another in a pre defined sequence.

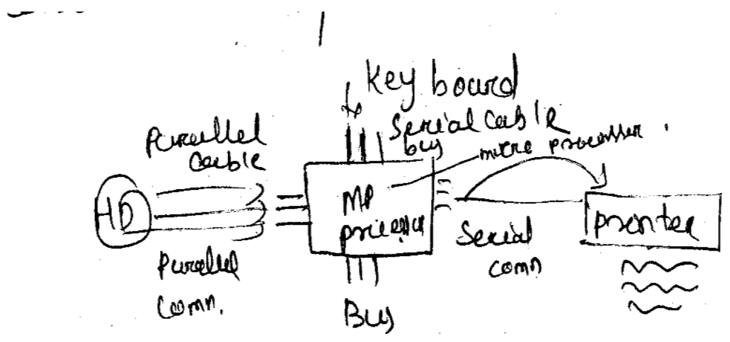
Bus structure:



All the components of the computer are interlive are inter connected so that word of data is transfer between the unit using parallel set up lines called Bus.

The lines that carry data most have the lines per address and controls purpose devices connected to the bus bary widely is their speed operations.

While some device are slow other like hard disk drive are faster memory and processor unit operate at electronic speed.



Performance:- performance is the ability of the computer and to quickly exact the program. the speed at which the computer execute the program OS design by its hardware and machine language instruction.

Basic performance mesure:-

The speed of operator of an com is generally decided by

i) Response time
ii) Through put.

1) **Response time:-**Response time is the time spent to complete and even or ad operation (Execution time).
2) **Through put:**- Through put is the amount of work done per unit of time that is called through put.
   ⇒ The amount of processing a accomplished during giving interval time (Band width).
   ⇒ Time spent from the start of execution of a program to its completion as called elapsed.

$$\left.\begin{array}{l} A = \overrightarrow{100} \\ G = 200 \\ C = a+b \end{array}\right\} \downarrow \quad \text{time el time.}$$

Performance parameter:-

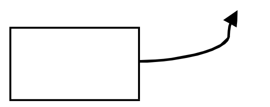
The basic performance parameter equation

T = (NxS)/R

Where T performance parameter of an application program N number of machine language instruction required to complete the execution required to complete the execution if an program S. Average num of basic steps required are clock rate of the processor in cycle per second.

Pipe lining and parallel processing:-

C:\\ $\overrightarrow{\text{DIR:}\rightarrow \text{a}}$bc



In case of pipe lining of the instruction to a computer have executed one after another. But in case of parallel processing multiple functional unit are used to create a parallel path. True which different on completed instructions can be execution. Thus due to state of execution several instruction in every where clock cycle.

Measuring performance:-

Response time and true pull are independent entities. When measuring the performance of a system.

Performance = 1/execution time

CPU performance equation:-

The CPU are construct using the clock running at a constant time.

CPU time = CPU clock cycle for program x clock cycle time

Cycles per instructions:-

Program consists of a number of CPU instruction be represented by instruction count.

CPI = instruction Count X Clock Cycle time / clock rate

Memory location and address:-

15	A	100
	100	101

110 111 1000

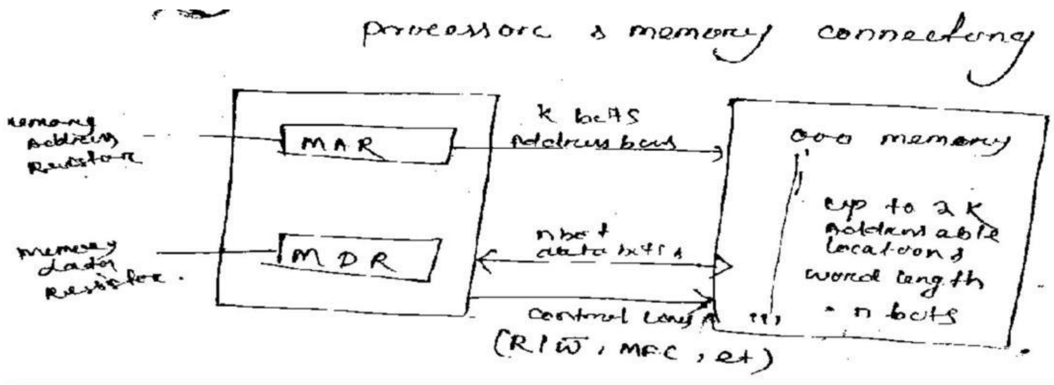
r index
r (10)

r (1)	r (2)	r (3)	---	r (10)
↓	↓	↓		
10	21	50		100

The maximum size of the memory that can be used in nay computer are determine by the addressing skim. A 16 bit computer can generate a 16 bit address capable of addressing upto 2^{16} K (kilobit) memory locations

2^{32} = 4 GB

Memory operations:-



⇒ Memory is usually dissguing to store and retrieve data in word length quantities.

⇒ Data transfer between the memory and the processor takes place through the use of 2 processor register called.

1) MAR – Memory Address register
2) MDR – Memory data register.

If Mar is K bits long, it can address up to 2 K address location. It MDR is n bit longs during a memory cycle, n both data are transfer between processing and memory 2 controls line read, write bar and memory function complete.

Co-ordinate the data transfer

Resister

To Hold an information

Address MAR   DATA MDR.

Instruction and instruction sequencing:-

The instruction said define many functions perform by the CPU. Information need to be provided on the various types of the data and operations to be perform on them.

This include the length instruction in bits, no of address to be used and the size of the each field.

The number of CP register that can be accessed by instruction for storage of data and operand.

Performance measures:-

Performance is the ability of the computer to quickly execute a program.

⇒ The speed at which the computer executes a program is decided by the design of its hardware and machine language instruction.

⇒ Computer performance measures is of very big term when used in context of the computer system.

⇒ System that execute program in less time are called to have higher performance.

Basic performance measures:-

The speed of operation of a system is generally decided by two fractions.

i) Response time
ii) Throughput.

Response Time:-

⇒ Response time is the time spend to complete an event or an operation.

⇒ It is also called as execution time or latency.

Throughput:-

Throughput is the amount of work done per unit of time. i.e. the amount of processing that can be accomplished during a given interval of time.

$\Rightarrow$ It is also called as bandwidth of the system.
$\Rightarrow$ In general, faster response time leads to better throughput.

Elapsed time:-

$\Rightarrow$ Elapsed time is a time spent from the start of execution of the program to its completion is called elapsed time.
$\Rightarrow$ This performance measure is affected by the clock speed of the processor and the concerned input output device.

MIPS

A nearly measure of computer performance has the rate at which a given machine executed instruction.
$\Rightarrow$ This is calculated by dividing the no. of instruction and the time required to run the program

CPI/IPC

CPI – Clock cycle per Instruction

IPC – Instruction per cycle.

It is another measuring that which is calculated as the number of clock cycle required to execute one instruction (cycle per instruction) by the instruction executed per cycle.

Speed up:-

Computer architecture use the speed up to describe the performance of architectural charges as different improvement are made to the system.

It is defined as ratio of execution time before to the execution time after the charge.

$$\text{Speed up} = \frac{\text{execution time before}}{\text{Execution time after}}$$
$$= \frac{\text{performance of system after}}{\text{Execution time before}}$$

Amdahl's law:-

This law states that "performance improvement to be gained by using a faster mode of execution is limited by the fraction of time the faster made can be used".

Amdahl's law defines the term speed up.

$$\text{Speed up} = \frac{\text{performance of entire task using enhancement}}{\text{Performance of in time task without using enhancement}}$$

$$\text{Performance} = \frac{1}{\text{Execution time}}$$

$$\text{Speed up} = \frac{\text{execution time without using enhancement}}{\text{Execution time with using enhancement}}$$

Factors affecting speedup are as follows:

i) The fraction of computation time in the original machine can be modified to use the advantage of the enhancement.

ii) This is called fraction enhanced which is always less than or equal to one.
Fraction enhanced ≤ 1
Ex: if a program that usually it will take 30 seconds for execution using the enhancement
Fraction enhanced = 30/100

2) Improvement granted by the enhanced execution made is the speed with which the taks could run faster using the enhancement.

Speed up > 1

$$\text{Speed up enhanced} = \frac{\text{time in original mode}}{\text{Time in enhance mode}}$$

Ex. Let us a program takes 5 second in enhanced mode while it takes 10 second earlier.

So, speedup enhanced = 10/8 = 2

The new execution time can be calculated as follow:

Execution time new =

Execution time original X ((1-fraction enhanced) + fraction enhanced/speedup enhanced)

The speedup overall = execution time original/execution time new

Speedup overall = 1/(1-fraction enhancement)+fraction enhances/Speedup enhance

Performance parameter

The basic performance equation is given by T = NxS/R

Where Q.t. – Performance parameter of an application program.

NS – No. of instruction required to complete the exe^2 of a program.

R-Clock rate of the processor in cycles per second.

S-A vg. no. of basic step required to execute one machine instruction.

Clock rate:-

Clock rate is one of the important performance measures by improving in the clock rate. There are two ways in which clock rate may be increased.

1) Improving IC technology which makes logic circuits faster thus reading time taken to complete a basic step.
2) By reducing the processing amount in one basic step which by reduces the clock period as R = 1/T.

CPU performance Equation:-

Normally the CPUs are constructed by using a clock running at a constant rate. This discrete time events is known as a clock cycle.

CPU time = the time of a program may be represented as

= CPU clock cycle for a program x clock cycle time

= CPU clock cycle/ clock rate

= instruction count X CPU

IC X CPI X C

Questions:-

1. **Define computer architecture.**
2. **Explain the basic functional units of a computer.**
3. **What is non-neuman architecture?**
4. **Explain virtual memory.**

CHAPTER-2

INSTRUCTION AND INSTRUCTION SEQUENCING

The types of addressing modes are available with instruction set.

1) functionality complex minimal instruction set.
2) Instruction set based on speed of execution 'RISC' reduce Instruction set computer.
3) A more elaborate instruction set that into frequently used sequence including a single processor operation.

Types of operands:-

Machine operations depends on the types of data being process . operand can be one of the following.

1. Address
2. Numbers
3. Characters
4. Logical data

Address:- Address are the for of number that represent specific location in memory.

Number data type:- Number data type are used by all machine language data type fix point floating point and decimal.

Character:- character are entered using ASCIIC. Encoding also another including character i.e. EBCDIC (Extended binary coded decimal inter change code) used for character.

Logical date:-

The Boolean data can be stored using 1(True) 0(false).

OPcode types:-

1. Arithmetic – add supply multiply divide
2. Logical – AND, OR, NOT
3. Data transfer – move, store, load, push, pop
4. Conversion – translate, convert
5. system control – reserve for operating system
6. Input output control – Input-readout – white.
7. Control transfer – Jump, returned, skip, halt.

Instruction format:- Opcode 4 bit Op1 Op2
6 bit 6 bit

An instruction is read into the instruction register (IR).

OP Code

Operation code	Address of Operand

→ Format of instruction

Operation Code (OP Code)

This specifies the operation to be performed.

e.g. :- Add, sub & Load

Address of Operand:-

The operand specified by the OP code may involve one or more sources for operands. These operands are the inputs for operations.

Classification of Instruction:-

Instruction are classified depending upon the number of operand address they contain such as classification is follows:-

1. **0-Address instruction**:-
   The 0-address type instruction do not contain any operand address. The operand address are implied.
   e.g. :- ADD TO S ← (A+B)
2. **1-Address Instruction:-**
   In 1-adress instruction only one operand address is specified in the instruction. The other operand is in accumulator.
   e.g.:- LOAD A AC ← M [A]
   ADD B AC ← AC+M[B]
3. **2-Address instruction:-**
   In 2-address instruction both operand address are specified. The result is placed in one of the specified address.
   e.g.:-MOV R_1 , A R_1 ← M[A]
   ADD R_1, B R_2 ← R_1 + M[B]
   MOV R_2, C R_2 ← M[C]
4. **3-Address instruction:-**
   In 3-address instruction two address are specified for two operands & one address for the result
   e.g.:-ADD R_1 , A, B R_1 ← M[A] + M[B]
   ADD R_2, C, D R_2 ← M[C] + M[D]
   MUL *, R_1, R_2 M[X] ← R_1*R_2

Addressing technique:-

To specify a memory address in an instruction word, the most obvious technique is simply to give the address in binary form. This called direct addressing, although direct addressing provides the most straight forward way to give a memory address, several other techniques are also used. The use of the these techniques is generally motivated by one of the following considerations.

(1) Desire to shorten address section
(2) Programmer convenience.
(3) System operation facilities.

Addressing mode:-

Each instruction needs data on which it has to perform the specified operation. The data (operand) may be in accumulator, general purpose, register, therefore there are various ways to specify data. The techniques of specifying the address of the data are known as addressing modes. We will discuss here only six type of addressing modes.

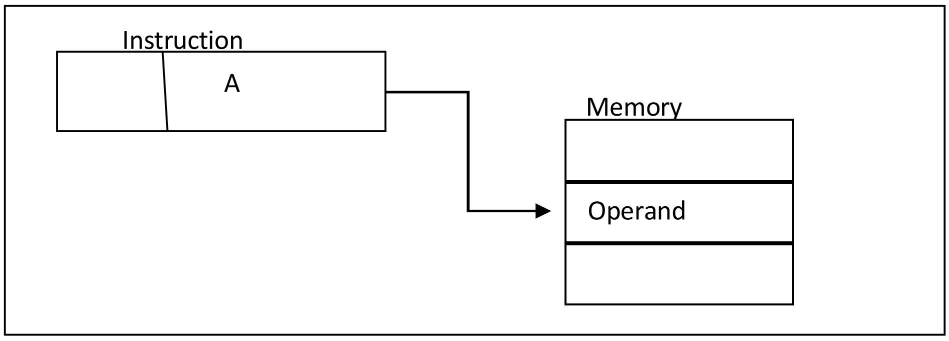
(1) Direct addressing
(2) Register addressing
(3) Register indirect addressing
(4) Immediate addressing
(5) Base register addressing
(6) Indirect addressing

(1) Direct addressing

In direct addressing the address of the data (Operand) is specified within the instruction itself.

e.g. :-

# STA 2500H -	Store the contents of accumulator in the memory location 2500 H. Here 2500H is the memory address. Where a data are to be stored.
# LDA 2500H -	Load the accumulator with the contents of the memory location 2500 H.
# In 01 -	Read the data from input device whose address is 01, here 01 is the address of an input device where data are to be read.



(2) Register addressing

In register addressing the operands are located in general purpose registers.
In other words the contents of a register is the operand.
Therefore only the names of the registers are to be specified in the instruction.

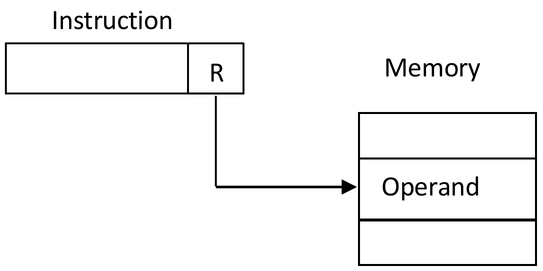
e.g. :-

# MOV A, B = Transfer the contents of register B to register A. the Opcode of this instruction is 78H in binary form is 01111000.

The1st 2 bit 01 denote move operation, the next three bit 111 are binary code for register A & last three bit 000 are binary code of register B of Intel 8085.

# ADDB = Add the contents of the register B to the content of the accumulator. The OP code of this instruction 804 in binary from 10000000. The 1st five bits 10000 specify the odd operation to be performed the last three bits 000 are for the

binary code of register B.

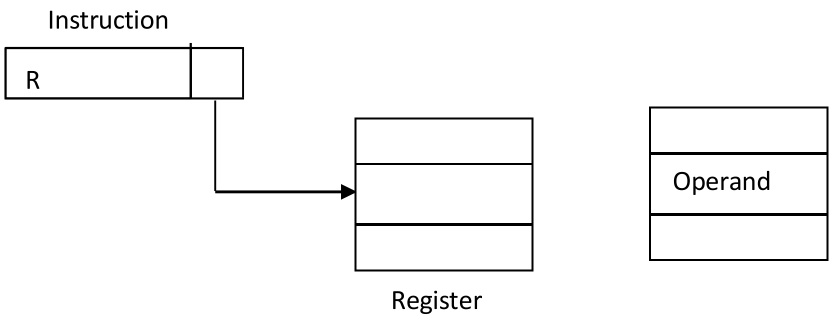


(3) In register indirect addressing address of the operand is given indirectly. The contents of the register or register pair are the address of operand.

# LXIH, 2400 MOV A, M =	Load H-L pair with 2400 H move the content of the memory location (M) whose address is in H-L pair)1, 0, 2400 H), to the contents of the accumulation.

In this example MOV A, M is an example of register indirect addressing. The address of the operand is not directly given but the address of the memory location is stored in H-L pair, which has been specified by the earlier instruction LXIH, 2400 H.

# LXIH, 2200 H ADDM -	Load H-L pair with 2200H add the content the memory location (M) whose address is in H-L pair to the contents of accumulator. Here ADDM is an example of indirect addressing.



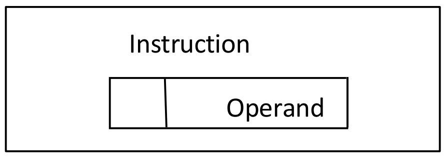
(4) Immediate addressing:-

In immediate addressing the operand is given in the instruction it self.

e.g. - # MV1, 06 move 06 to accumulator

# ADI 05 ADD immediate 05 to the contents of a accumulator

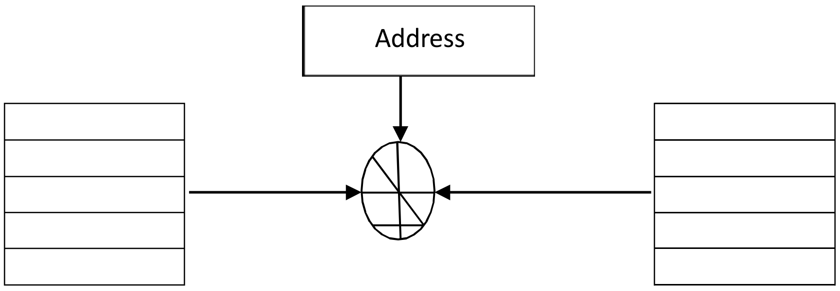
# LXIH, 2500 H

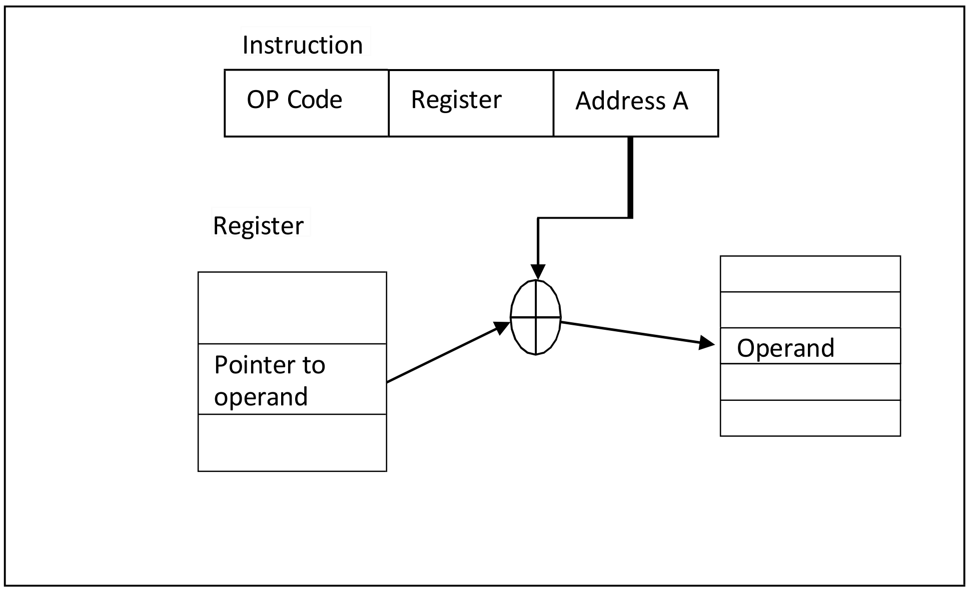


(5) Base register addressing:-

In many computer system it becomes necessary to move programme from one place to another, in the memory. To solve this problem many computer use base register. The addressing mode employing a base register is known as Base register. In this mode of addressing an offset is added to the contents of the base register to obtain the effective address.

The instruction contain an offset. When a program is moved into memory. The starting address is placed in the base register.

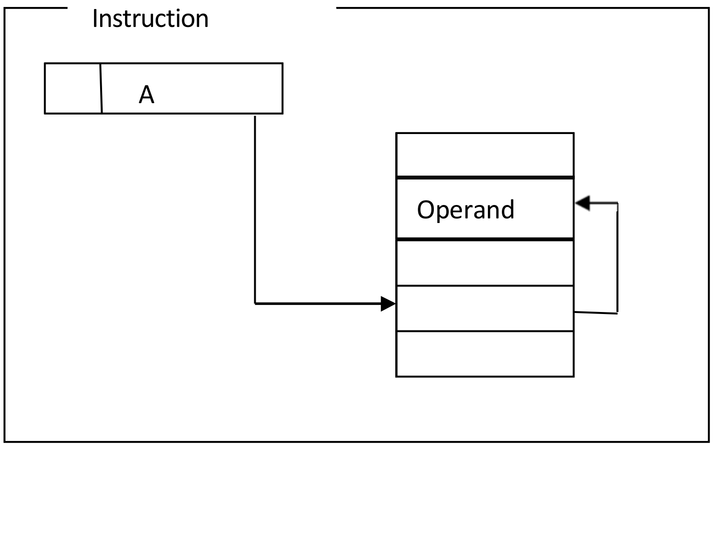




(6) Indirect addressing:-

In indirect addressing the address field of the instruction gives the address of the instruction gives the address of the operand. Control fetches the instruction from memory and uses its address port to access memory register to read the effective address.

e.g. - # IAD 302 – Indirect add the number whose address is stored at the address 302.



Instruction set:- Instruction set based on number of address.

1) Pre address instruction

    Ex:- x=(A+B)X(C+D)

    3Addness instruction:-

    ADD = R1 A, B, R1← M[A]+M[B] – add B with a

    Add = R2, C,D,R2 ← M[C]+M[D]

    NUL = X1R1,R2, M[X] ← R1*R2

Computer using 3 address instruction have an address field to specify either process register or a memory address from where operands can be paste.

Addressing technique:-

To specify a memory address in a instruction words the most obvious technique is simply to give the address in binary form.

This called direct addressing

Addressing mode:-

The operand field of instruction specified to address from where the data has to be fetched. The addressing mode specifies a rule for interpreting or translating of the instructor into an effective address from where the operand is actually reference. The control unit of computer should go through an instruction cycle to execute an instruction.

A computer has a resister called the program counter which keeps the stack program instruction that was store memory. It holds the address of next sequential instruction to be fetched. The decoding determines the operation to be perform.

The operands are fetch either form memory or the register.

The instruction is executed and the result will be put pack into the operand address. The various addressing model are.

1) **Immediate addressing:-** In the mode of addressing the operand in the part of instruction and it specified in address field. This called immediate addressing.
   Add-5 Operand
2) **Direct Addressing:-** The effective address is equal to the address part of the instruction. The operand register in the memory and its address is gen directly by the address field of instruction.
3) **In directing Addressing:**- the addressing field of the instructed uses the address filed of the operand stored in the memory.
4) **Resister addressing:**- here the addressing field register to register.

Register in directing addressing:- The address field refers to a resister and the effective address is the address given by the register given by of the address field.

Stack addressing:- Stack addressing in form of addressing.

Questions:

1. **Explain addressing mode.**
2. **Explain the various types of instructions.**

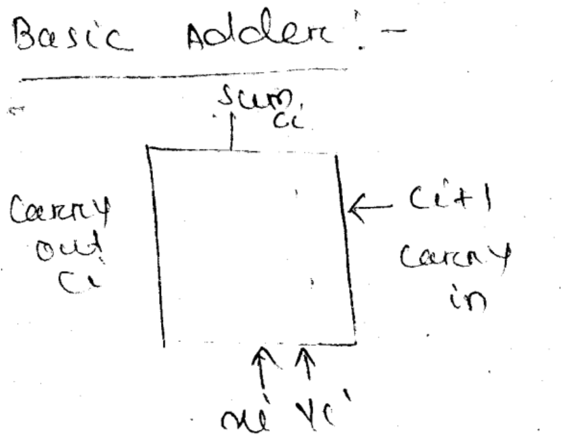
CHAPTER-3

ARITHMETIC OPERATIONS

Basic Arithmetic operations:-

The 4 basic arithmetic operations are addition subtraction, multiplication, division.

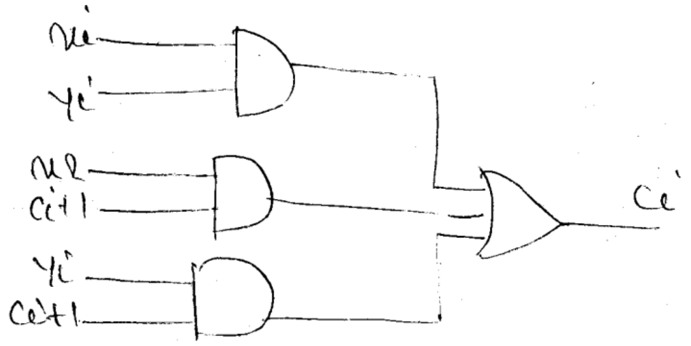
Addition:-



A full adder ckt can be uses with the help of logic gate for adder ckt. The simplest form of adder is the serial binary adder. Which adds 2 numbers bit by bit and hence required a cycle. Compute come of 2 n bit number. It consists of combinations full adder and flip flop to store ci. Output is generated to each click cycle. A carry i.e. generated is stored for using during the next clock cycle. There are ckt called parallel adder that can add all the bits of 2 number simultaneously in parallel in one clock cycle.

Full adder ckt:-

Flip flop is a register which storing the element



The full order ckt simple binary operation

0+0 = 00

0+1 = 01

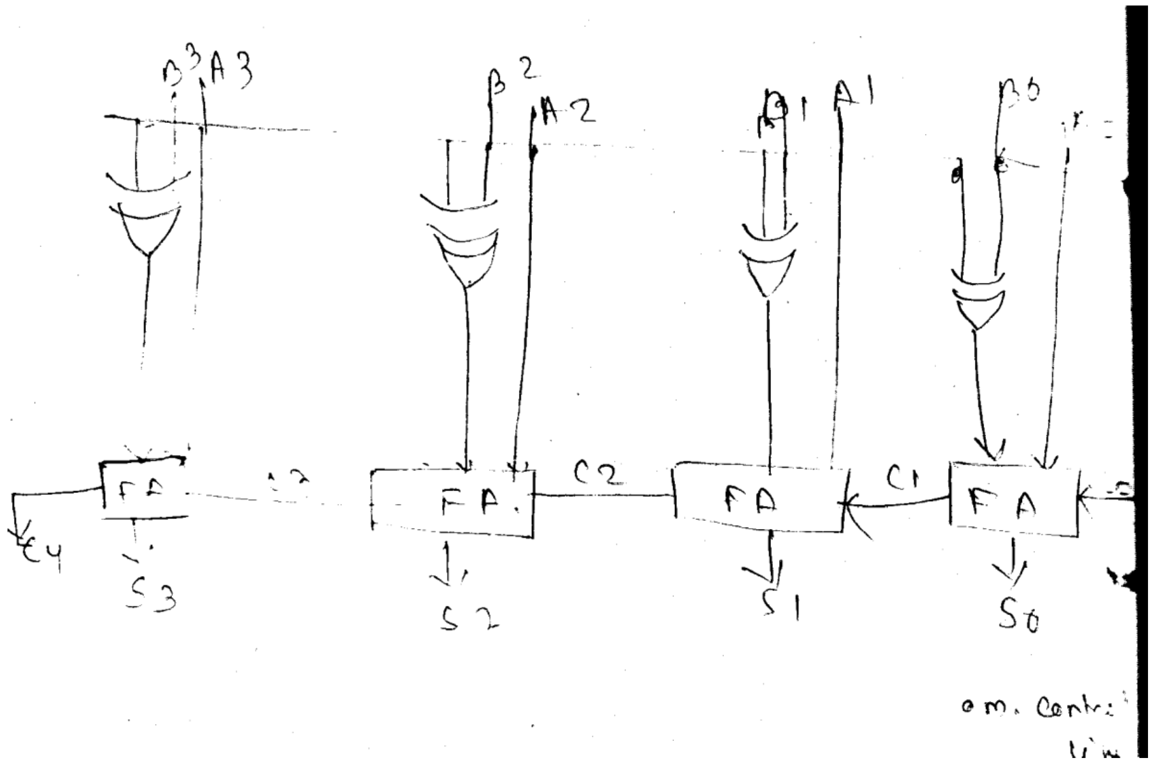
1+0=01

1+1 = 10

Input			Output	
Xi	Yi	Ci+1	Ci	Zi
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Subtraction:-

6-12 = -6	1100
0110	0011
1100	+1
-1010	0100
	1010
	0101
	+ 1
	0110



The subtraction of binary number can be done by means of complement. Subtraction of A-B can be done using 2’s complement of B adding to A.

The 2’s complement can be obtained by taking the 1’s complement and adding one to the cost significant bit.

Ex. 011011

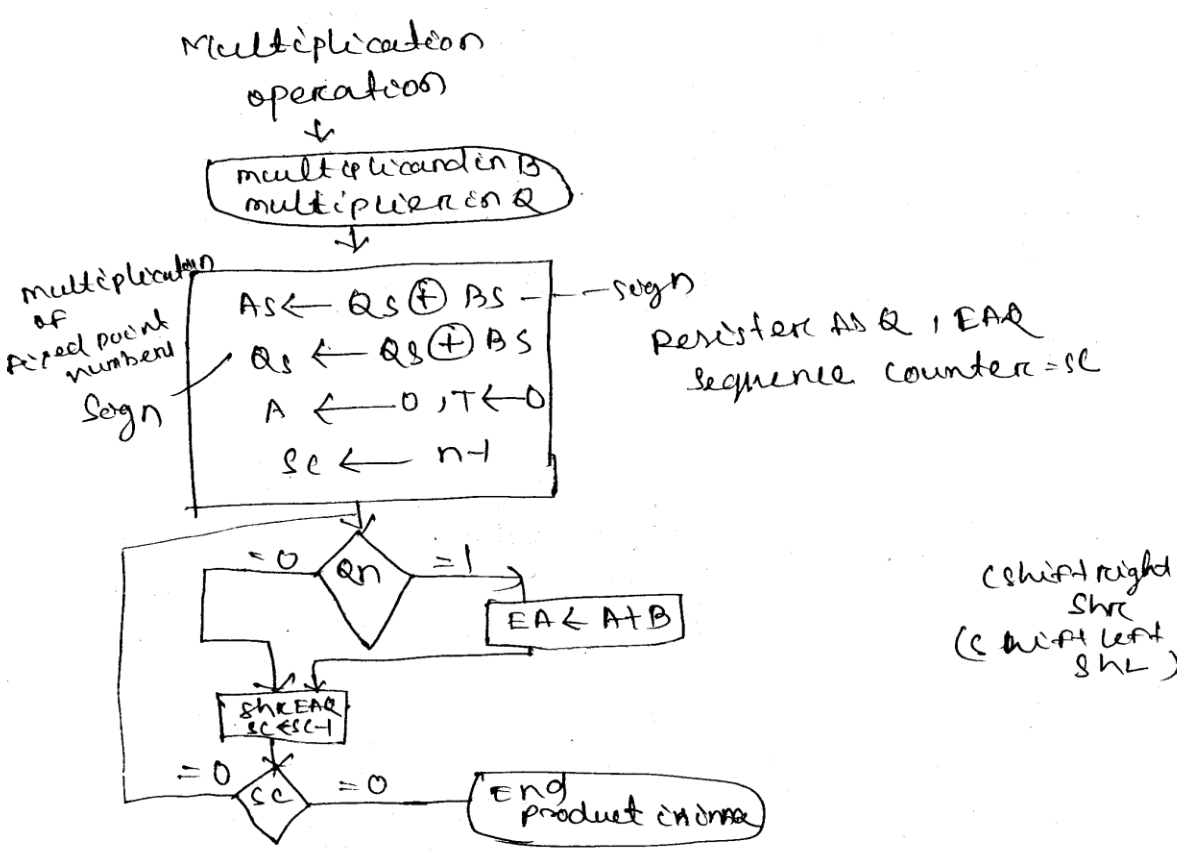
Msb csb

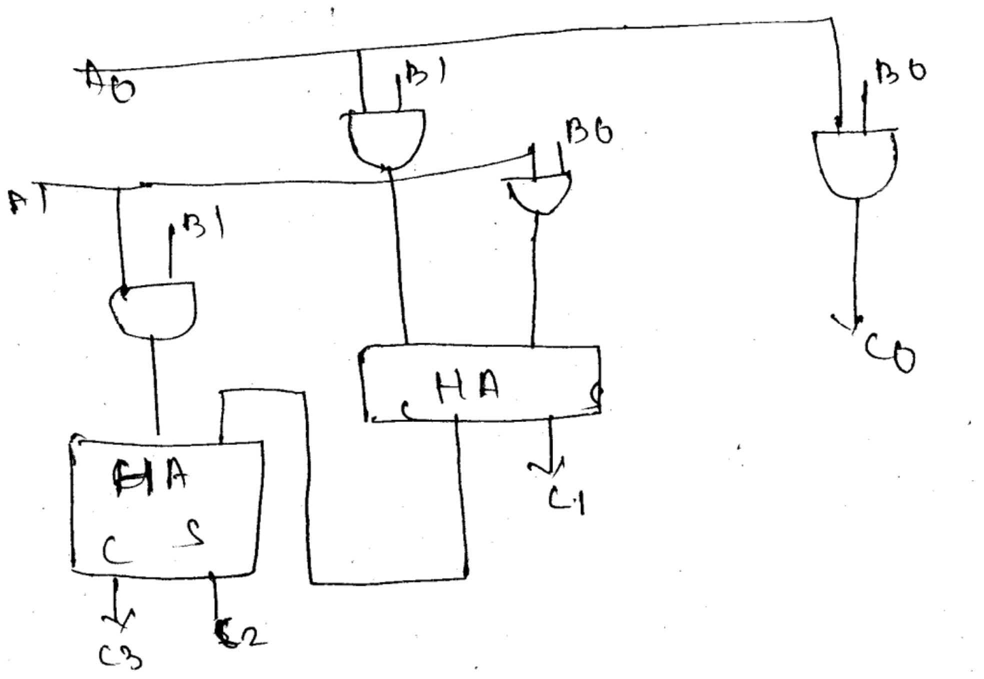
The addition and subtraction operation can be perform using one common CK1 by using an executive or gate with each full order. The control line mark m will control the operation.

Multiplication algorithm:-

Multiplication of two fixed point binary numbers in signed magnitude representation is done by a shift and add operation. If the multiplier bit is a1, the multiplicand is copied down otherwise here are copies down.

The numbers copied down in successive line are sifted one position to the left from the previous number. Finally the numbers are added and their sum forms the product. Multiplication of fixed point numbers:-





2 bit by 2 bit array multiples

Multipli land bits B11B0

Multiplier bits A11A0

Products C3 C2 C1 C0

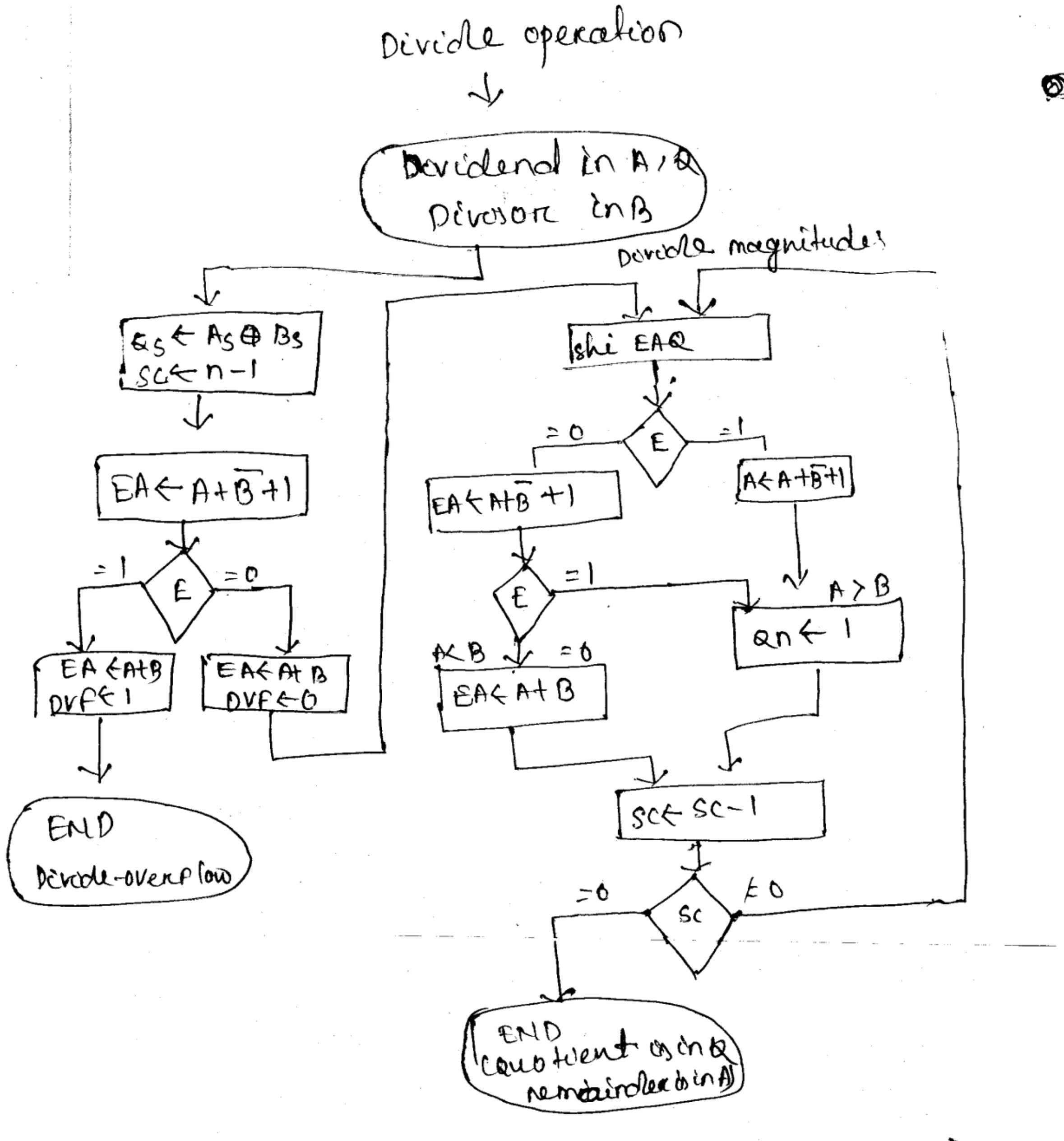
Partical product A0 with B1, B6

Multiplication of two bits A0, B6 = 1

Otherwise = 0

Because and operation

<u>Division algorithm:-</u> the division of two fix point binary number in sign magnitude representation is done by process of successive compare, shift, and subtract operation.



(Flow chart for the Divide operation)

+1234.567

-5372400x10^2 56780x10^5

-0001580x10^2 -56430x10^5

Floating point arithmetic

$$\begin{tikzcd}[column sep=small] & \text{Number system} \arrow[dl] \arrow[dr] & \\ \text{Fixed point x without del pt.} & & \text{floating point with del point} \end{tikzcd}$$

56	56.0
56	5.6
	0.56

$\underline{+0.1234567x10^{+7}}$ = +0.1234567E+4

+1234.567 mxr^2

$123.45678x10^{-2}$

$\frac{123.45678}{100}$ = 12345.678

$.5372400x10^{2}$

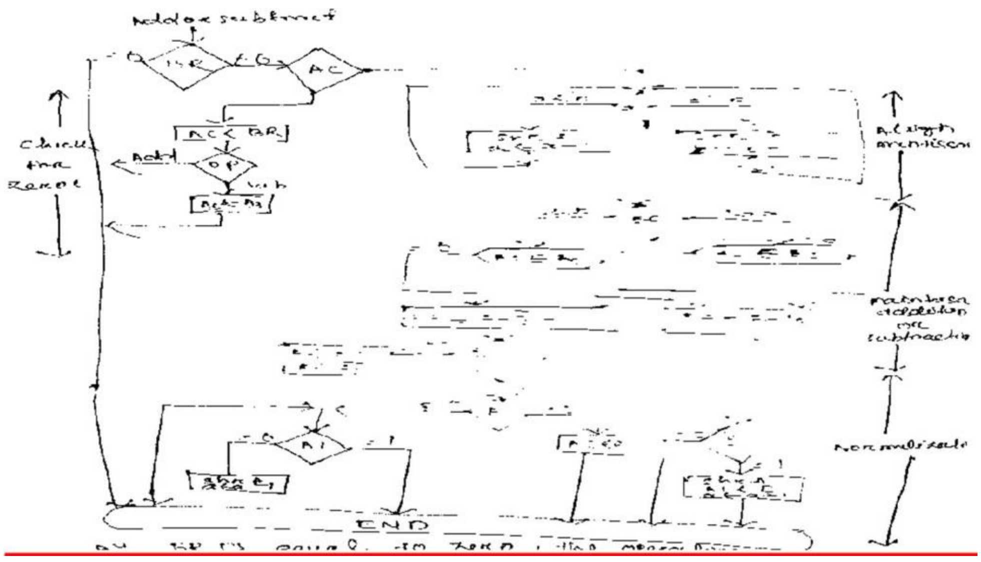
$\underline{.0001580x10^{2}}$

$+0.5373980x10^{+2}$

Floating point addition and subtraction:-

During addition or substraction, the two floating point operands are in Ac and BR. The sum of difference is formed in the AC. The algorithm can be divided into four consecutive parts.

1. Check for zeros
2. Align the mantissas
3. Add or substract the mantissas
4. Normalize the result.



If BR is equal to

zero, the operation is terminated with the value in AC being the result. If AC is equal to zero, we transfer the content of BR into AC and also complement its sign if the numbers are to be subtracted. If neither number is equal to zero. We processed to align the mantissas. The magnitude comparator attached to exponents a and b provides three outputs that indicate their relative magnitude. If the two exponent are not equal, the mantissa having the smaller exponent is shifted to the right and its exponent incremented. This process is repeated until the two exponents are equal.

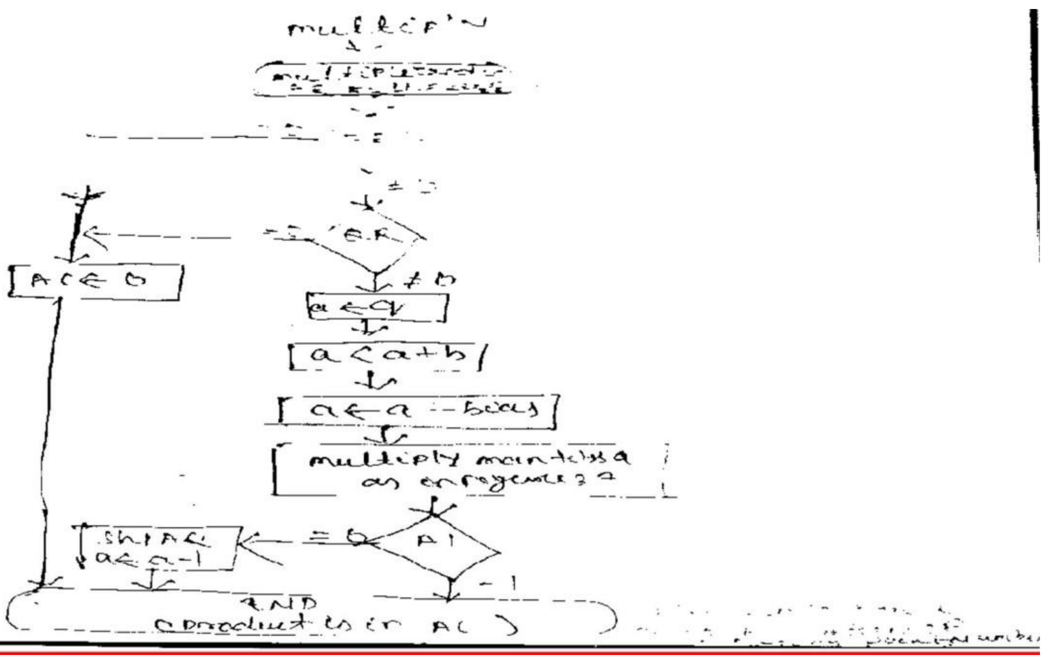
The addition and subtraction of the two mantissa as identical to the fixed point addition and the subtraction algorithm presented earlier. The magnitude part is added or subtracted dependency on the operation and the signs of the two mantissas. If an overflow occurs when the magnitudes are added, it is transferred into flip flop. E if e is equal to 1, the bit is transferred into A1 and all other bits of area shifted right. The exponent must be incremented to maintain the correct number. No underflow may occur on this case because the original mantissa that was not shifted during the alignment was already on a normalized position. If the magnitudes were subtracted, the result may be zero or may have and underflow. If the mantissa is zero, the entire floating point number in AC is made zero. Otherwise, the mantissa must have at least one but that is equal to 1. The mantissa has an underflow if the most significant bit in position A1 as 0. In that case, 1 the mantissa is shifted left and the exponent decremented. The bit in A1 is checked again and process is repeated until it is equal to 1. When A1 = . 1 the mantissa is normalized and the operation in completed.

Floating point multiplication:-

The multiplication two floating point numbers requires that we multiply the mantissas and add the exponents.

The multiplication algorithm can be subdivided into four parts.

1. Check for zero
2. Add the exponents
3. Multiply the mantissa
4. Normalize the product



The two operands are checked to determine if they contain a zero. If either operand is equal to zero the product in the AC is set to zero and the operation is terminated. If neither of the operands is equal to zero, the process continues with the exponent addition. The exponent of the multiplier is in q and the adder is between exponents a and b. it is necessary to transfer the exponent from q to a, add the two exponents and transfer the sum into a. since both exponents are biased by the addition of a constant, the exponent sum will have double the bias. The correct biased exponent for the product is obtained by subtracting the bias number from the sum. The multiplication of the mantissas is done as on fixed point case with the product residing in A and Q. over flow cannot occur during multiplication, so there is no need to check for it.

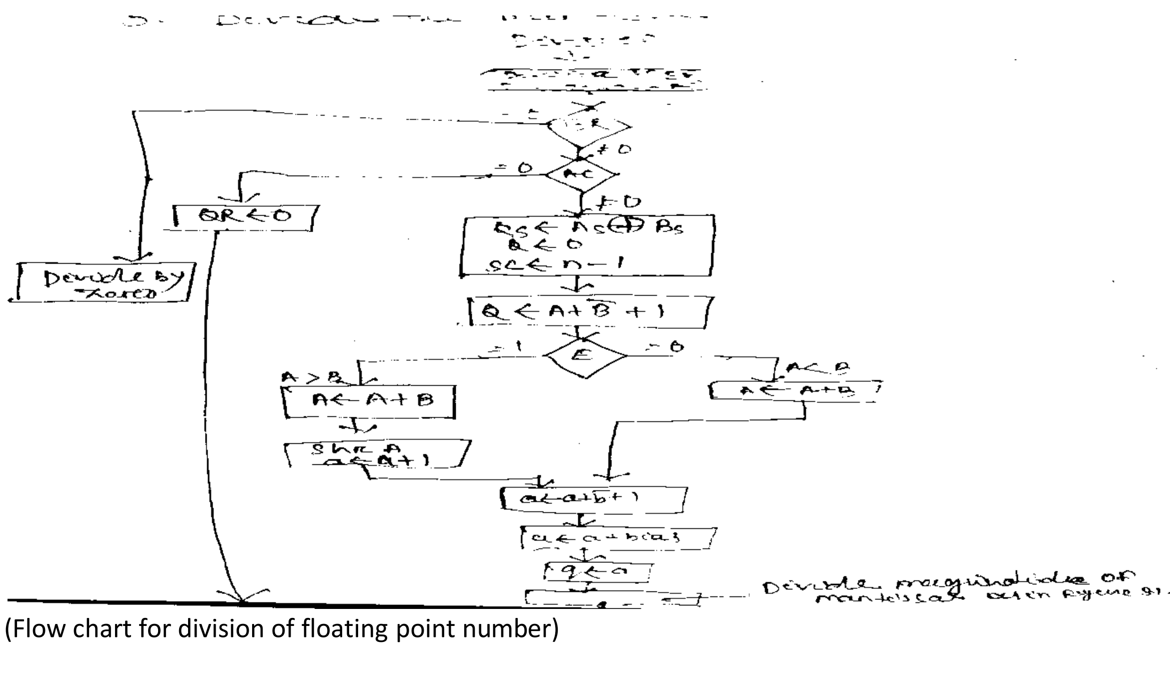
The product may have an underflow, so the most significant bit in A is checked. If its A1, the product is already normalized and if it is a 0, the mantissas in AQ are shifted left and the exponent decremented. Note that only one normalization shift is necessary. The multiplier and the multiplicand were originally normalized and contained fractions. The smallest normalized operand is 0.1 and the smallest possible product is 0.01. Therefore only one loading zero may occur.

Although the low-order half of the mantissa is in Q, we do not use it for the floating point product.
However, only the value in AC is taken as product.

Floating point Division:-

Floating point division requires that the exponent be subtracted and the mantissas divided.
The division algorithm can be subdivided into five parts:-

1. Check for zeros
2. Initialize resister and evaluate the sign
3. Align the divided
4. Subtract the exponent
5. Divide the mantissas.



(Flow chart for division of floating point number)

If the operands are not zero, we proceed to determine the sign of the quotient and store it on QS. The sign of the dividend in As is left unchanged to be the sign of the remainder. The Q register is cleared and the sequence counter SC is set to a number equal to the number of bits in the quotient. The dividend alignment is similar to the divide overflow check in the fixed point operations. Proper alignment requires that the fraction divided be smaller than the divisor. The two fractions are compared by a subtraction test. The carry on E determines their negative magnitude. The dividend fraction is restored to its original value by adding the divisor. If A=B, of on necessary to shift A once to the right and increment the dividend exponent since both operands are normalized, the alignment ensures that A<B. Next, the divisor exponent is subtract from the divided exponent. Since both exponents were originally biases. The subtraction operations give the deference without the bias. The bias on then added and the result transferred into q because the quotient is formed in QR.

The magnitude of the mantissas are divided as in the fixed point case. After the operation, the mantissa quotient resides in Q and the remainder in A. the floating point quotient is already normalized should be the same as the exponent of the dividend. The binary point for the reminder mantissa lies at (n-1) positions to the left of P1. The reminder can be converted to a normalized fraction by subtracting n-1 from the divided exponent and by shift and by decrement until the bit in A1 is equal to 1.

Arithmetic Micro operation

The micro operation must after encountered in digital computer are classified into four categories.

1) Register transfer micro operation transfers binary information from one register to another.
2) Arithmetic micro operations perform arithmetic operation on the data stored in register.
3) Logic micro operations perform bit manipulation operation on non numeric data stored in register.
4) Shift micro operation perform shift operation on data stored in register.
   Register transfer – does not change the content of the register.
   Arithmetic logic shift – change the content of the register

Symbolic description	Arithmetic	Micro operation description
R3← R1+R2	Contents of R1 plus R2 transferred to R3	
R3← R1-R2	Contents of R1 minus R2 transferred to R3	
R2← R2	Complement the contents of R2 is complement	
R2← R2+1	2 is complement the contents of R2 negat.	
R3← R1+R2+1	R1 plus 2 is complement of R2	
R1← R1+1	Increment the contents of R1 by one	
R1← R1-1	Decrement the content of R2 by an	

Logic micro operation

Logic micro operation specify binary operations for strings of bits stored in register. These operations consider each bit of the register separately and threat they as binary variable.

Truth table for 16 functions of two variables

X	Y	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15
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

Logic micro operation

Bollcan function	**Micro operation**	**Name**
$F0 = 0$	$F \leftarrow 0$	Clear
$F1 = xy$	$F \leftarrow A^\wedge B$	And
$F2 = xy^1$	$F \leftarrow A^\wedge B$	Transfer A
$F3 = x$	$F \leftarrow A$	Transfer on B
$F4 = x^1y$	$F \leftarrow A^\wedge B$	Exclusive 0 R
$F5 = y$	$F \leftarrow B$	Or
$F6 = x + y$	$F \leftarrow A+B$	Nor
$F7= x + y$	$F \leftarrow AvB$	Exclusive NOR
$F8= (x+y)^1$	$F \leftarrow \underline{AVB}$	Complement B
$F9 = (x+y)^1$	$F \leftarrow A+B$	
$F10 = y^1$	$F \leftarrow B$	
$F11= x+y^1$	$F \leftarrow AVB$	
$F12=x^1$	$F \leftarrow A^1$	Complement A
$F13=x^1+y$	$F \leftarrow AVB$	
$F14= (xy)^1$	$F \leftarrow A^\wedge B$	Non
$F15= 1$	$F \leftarrow$ ALL IS	Set to all 1's

Shift micro operation

In shift operation the contents of the register can be shifted to the left on the right . at the same time the bits are shifted, the first flip flop receives its binary information from the serial input. During a shift left operation the serial input transfers a bit into the rightmost portion. During a shift right operation the serial input transfer a bit into the leftmost operation.

The information transferred through the serial input determines the type of shift. There are three types of shifts logical, circular and arithmetic.

Logical shift:-

A logical shift is one that transfers 0 through the serial input.

Eg. $R1 \leftarrow$ sh/ R1

$R2 \leftarrow$ sh/ R2

Circular shift:-

The circular shift clear known as a rotate operation circulates the bits of the register around the two area without loss of information.

$R \leftarrow$ ci/R

$R \leftarrow$ shr R

Arithmetic shift:-

It is a micro operation that shifts arranged binary number to the left on right. An arithmetic shift left multiplies a singed number by 2. An arithmetic shift right levels a signed number by 2.

Eg:

R ← ash/R

R ← AshrR

Instruction cycle:-

The program residing in the memory unit of the computer consists of a sequence of instructions. The program is executed in the computer by going through a cycle for each instruction. Each instruction cycle is turn is multi drives into a sequence of sub cycles or phases.

In the basic computer each instruction cycle consists of the following phases.

1. Fetch an instruction from memory
2. Decode the instruction.
3. Read the effective address from memory if the instruction has an indirect address.
4. Execute the instruction.

Fetch and decode:-

Initially the program counter CPC is loaded with the address of the first instruction in the program. The sequence counter SC is cleared to 0, providing a decoded timing signal/70. After each clock please, SC is incremented by one, so that the timing signal go through a sequence to,70, 7, 72 and so on.

The micro operation for the fetch and decode phases can be specified by the following register transfer statement.

70: AR ← PC

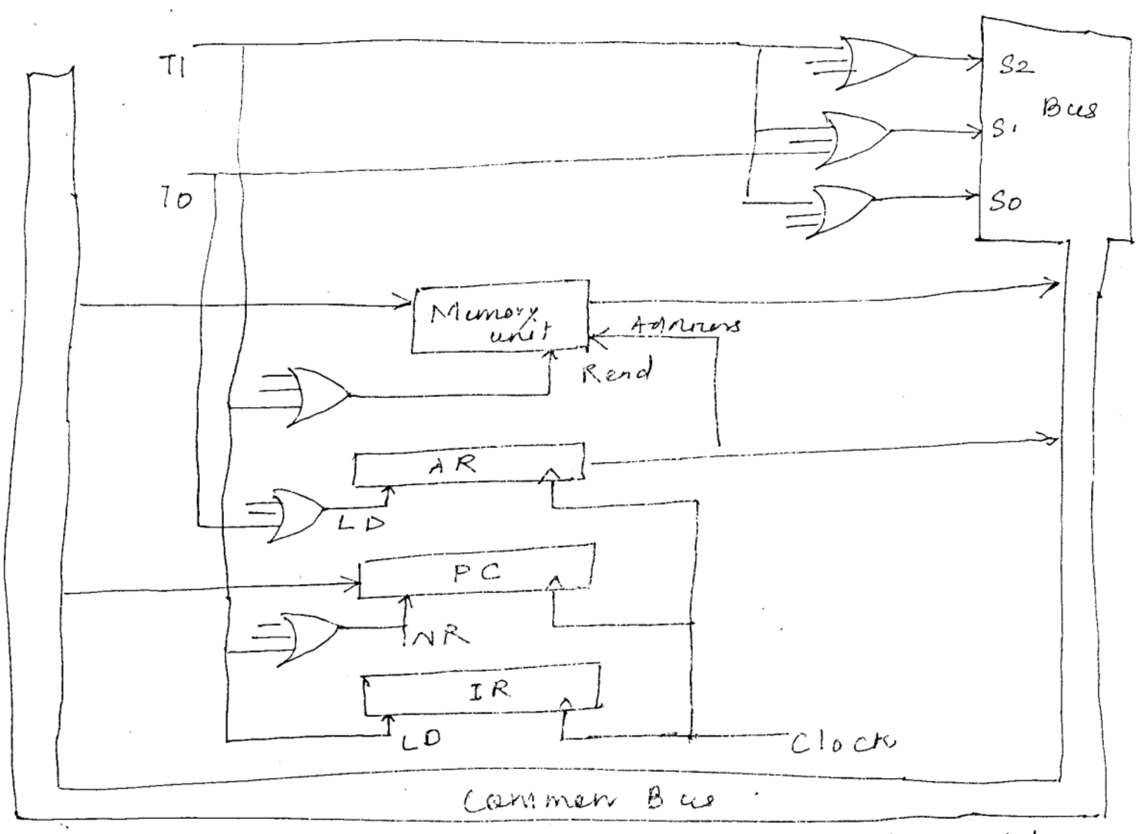
71 : IR ← M[AR], PC ← PC+1

72 : Do, ---, Dy ← Decodes [R(12-14), AR ← IR(0-4), I ← IR(15).

Since only AR is cancelled to the address input of memory, it is necessary to transfer the address from PC to AR during the clock transition associated with timing signal 70. The instruction read from memory is then placed in the instruction register IR with the clock transition associated with timing signal 71. At the same time PC is incremented by one to prepare it for the address of the next instruction in the program. At time 72, the operation code in IR is decoded, the indirect bit is transferred to flip flop I, and the address part of the instruction is transferred to Ar. SC is incremented after each dock poles to produce the sequence, 70, 7 and 72.

To provide the data path for the transfer of PC to AR timing signal 70 is applied to achieve the following connection.

1. Place the content of PC into the bus by making the bus selection inputs S2S1S0 equal to 0/0.
2. Transfer the content of the bus to AR by enabling the LD input of AR. The next clock transition initializes the transfer from PC to AR since 70 = 1.



Register transfer for the fetch phase

In order to implement the second statement.

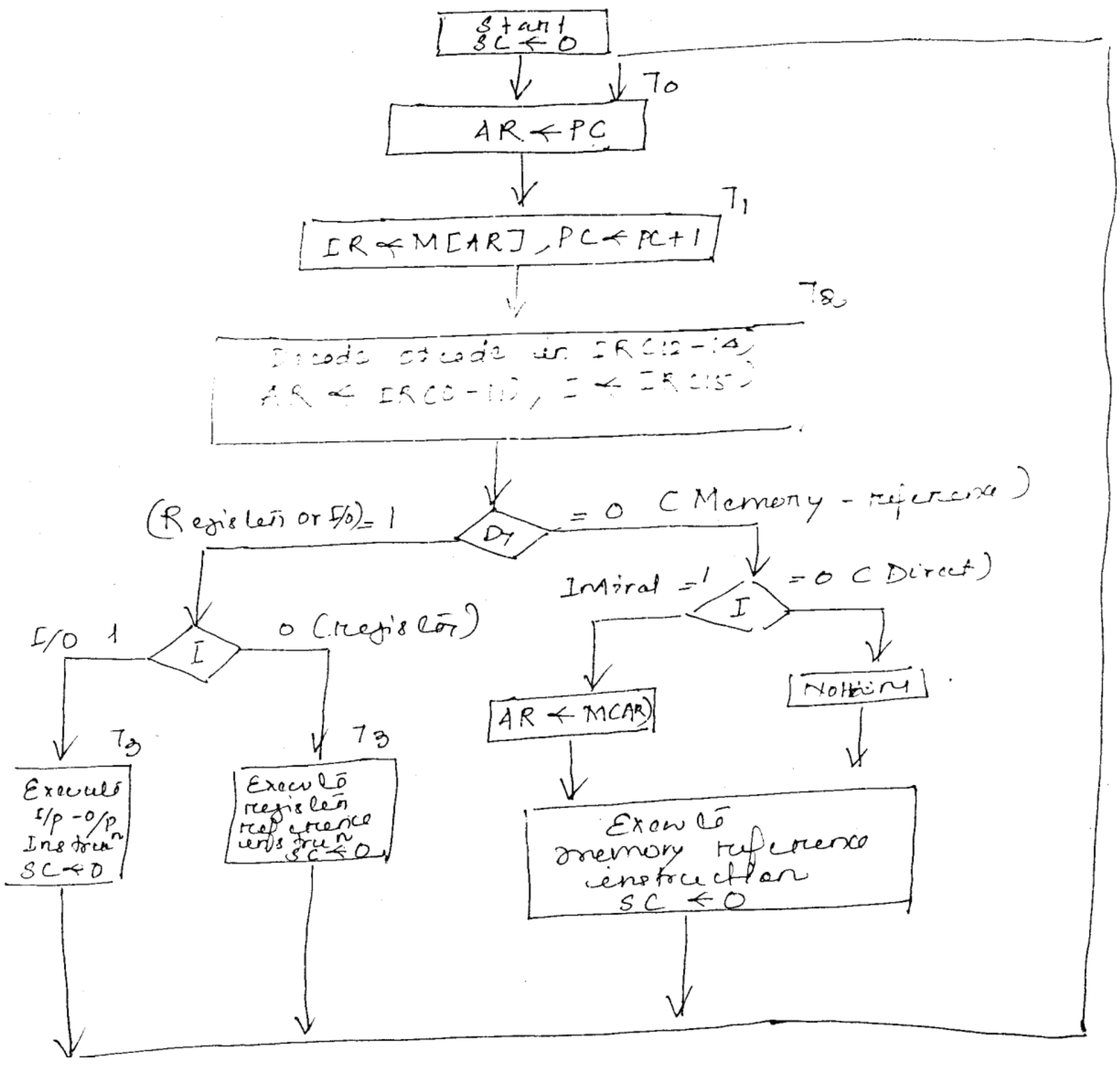
71: IR ← M [AR], PC ← PC+1

It is necessary to use the timing single 71 to provide the following connections in the bus resister.

1. Enable the read i/p of memory.
2. Place the content of memory into the bus by making S2S1S0=111.
3. Transfer the content of the bus to IR by enabling the LO input of IR.
4. Increment PC by enabling the INR i/p of PC.

The next clock transistor initializes the rend and increment operations since 70=1.

Flow chart for Instruction cycle



Start SC ← 0
T0
AR ← PC
T1
IR ← M[AR], PC ← PC+1
T2
Decode opcode in IR(12-14), AR ← IR(0-11), I ← IR(15)
(Register or I/O) = 1
D7
= 0 (Memory - reference)
Indirect = 1
= 0 (Direct)
I
AR ← M[AR]
Nothing
I/O 1
I
0 (register)
T3
T3
Execute I/p -O/p Instrn SC ← 0
Execute register reference instrn SC ← 0
Execute memory reference instruction SC ← 0

Types of instruction:-

1) Memory reference instruction
2) Register reference instruction
3) Input output reference instruction

Instruction format:-

15 14	12 11	0	
I	OP code	Address	(Op code = 000 through 110

Memory – reference instruction I = O/I O= direct , I =Indirect

15	12 11	0
0 1 1 1	Register operation	(Op code = 111, I = 0)

Register reference instruction

15	12 11	0
0 1 1 1	I/O Operation	(Op code = 111, I = 0)

Input – Output Operation

Basic computer instruction

Hexadecimal code

Symbol	I = O	I = 1	Description
AND	0xxx	8xxx	And memory work to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load memory word to AC
STA	3xxx	Bxxx	Store content of AC in memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save retun address
JSZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear Ac
CKE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Complement E
CIR	7080		Circulate right AC and E
CIL	7040		Circulate right Ac and E
INC	7020		Increment Ac
SPA	7010		Snip next instruction if Ac positive

Hexadecimal code

Symbol		Description
SNA	7008	Snip next instruction if AC
SZA	7004	Snip next instruction if AC = 0
SZE	7002	Snip next instruction if E = 0
HLT	7001	Halt computer
INP	F800	Input character to AC
OUT	F400	Output character from Ac
SKI	F200	Skip on I/P flag
SKO	F100	Skip on O/P flag
ION	F080	Interrupt on
IOF	F040	Interrupt off

Instruction set completeness

The set of instruction are said to be complete if the computer includes sufficient number of instructions in each of the following categories.

1. Arithmetic, logical and shipt instruction.
2. Instructions for moving information to a from memory and processor resisters.
3. Program control instructions together with instructions that check steps connection.
4. Input output instruction.

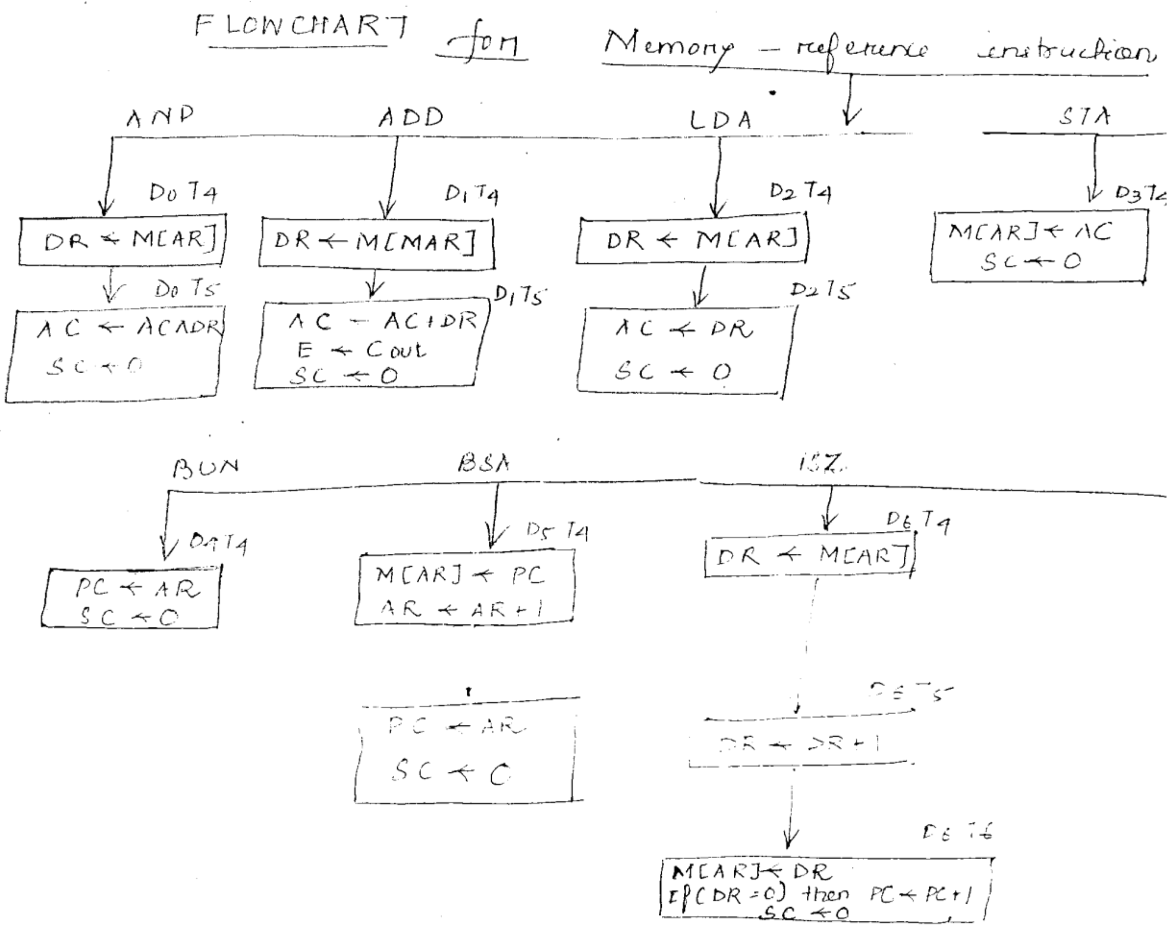
Execution programme register reference instruction:-

D7 I'T3=r (common to all register – reference instruction)
IR(i) = Bi (bit in IR(0-11) that specifies the operation)
R : fc ← 0 clear SC
CLA n B11: AC ← 0 clear AC CLE
n B10: E ← 0 clear E
CMA n B9: AC ← $\overline{AC}$ complement AC
CME n B8: AC ← E complement E (register)
CIR n B7: AC ← shr AC, A(15) ← E, E←A(10)
Circulate right
CIL n B6: AC ← sh/AC, AC(0) ← E, E←AC(15)
Circulate left
INC n B5: AC ← AC+1, Increment AC
SPA n B4: If (AC(15)=0) THEN (PC←1)
Skip if positive
SNA n B3: If (AC(15)=1) THEN (PC←PC+1)
Skip if negative
SZA n B2: If (AC=0) the (PC←PC+1)
Skip if Ac zero
SPA n B1: If (E=0) the (PC*PC+1)
Skip if E zero
HLT Nb0: S←0 (S is a start stop flip flop

Memory reference instructions

Symbol	**Operation decoder**	**Symbolic description**
AND	Do	AC← AC^M[AR]
ADD	D1	AC← AC+M[AR], E←cout
LAD	D2	AC←M[AR]
STA	D3	M[AR] ←AC
BUN	D4	PC← AR
BSA	D5	M[AR]←PC, PC←AR+1
ISZ	D6	M[AR]←m[AR]+1, if M[AR]+1=0 the PC←PC+1

Flowchart for memory-refenrnce instruction:-



Input output instruction:-

D7IT3 = P (common to all input – output instructions

IR(i) = Bi[bit in IR(6-11) that specifies the instruction).

	P: S← 0	Clear SC
INP	PB11 : AC(0-7)←INPR, FGI←0	Input character
OUT	PB10: OUTR←AC(0-7), FGO←0	OUTPUT CHARCTER
SKI	PB:9: if (FGI=1) then (PC←PC+1)	Skip on i/p flag
SKO	PB8:If (FGO=9) then (PC←PC+1)	Skip on O/P flag
ION	PB9: IFN←1	Interrupt enable on
IOF	PB6: IFN←0	Interrupt enable off

Fixed point representation:-

The sign of an integer number can be represented with a bit placed in the leftmost, pavilion of the number. The convention is to make the sign bit equal to 0 for positive and 1 for negative.

In addition to a sign, a number may have a binary or decimal point. Whose position is needed to represent fractions, integers, or mixed integer fraction numbers. There are two ways of specifying the position of the binary point in register by giving it a fixed position or by employing floating point representation.

The fixed point method assumes that the binary point is always fixed in one position. The two positions mast-widely used are.

i) A binary point in the extreme left of the register to make the stored number a fraction.
ii) A binary point in the extreme right of the register to make the stored number an integer.

The floating point representation used a second register to store a number that designates the position of the decimal point in the first register.

Integer representation

Signed numbers:- when an integer binary number is positive, the sign is represented by 0 and the magnitude by a positive binary number is represented by 1 but the rest of the number may be represented in one of the three possible ways.

1. Signed – magnitude representation
2. Signed – 1's complement representation
3. Signed – 2's complement representation.
   eg. +14 : 0000 1110
   -14
   Signed – magnitude representation 10001110
   Signed – 1' complement representation 11110001
   Signed 2's complement representation 11110010
   In compiler arithmetic signed
   2's complement representation is used.

Arithmetic addition

The adding numbers in the signed 2's complement system require only addition and complementation. Add the two numbers, including hut sign bits , and discard any carry out any the sign (leftmost) bit position.

Eg.	+6	00000110	-6	11111010
	+13	00001101	+13	00001101
	+19	00010011	+7	00000111
	+6	00000110	-6	11111010
	-13	11110011	-13	11110011
	-7	11111001	-19	11101101

In each of the four cases, the operation performed is always addition, including. The sign bit. Any carry out of the sign bit position is discarded and negative results are automatically in 2's complement form.

To determine the value of a negative number when in signed – 2 complement it is necessary to convert it to a positive number to place it in a more familiar form.

Eg. No is 11111001 is –ve

2’s complement 00000111 which is binary equivalent of +ve

Therefore the original –ve number is –y

Arithmetic subtraction:-

Subtraction of two signed binary numbers when negative number are in 2’s complement form is the complement of the subtracted (including the sign bit) and added to the minuend (including the sign bit). A carry out of the sign bit position is discarded.

```
Eg.    -6        11111011 => 11111011

       --13 -    11110011 +   00001101

       +7               100000111

                End carry ↓

                Discarded +→
```

Overflow:-

When two numbers of n digits each are added and the sum occupies n+1 digit an overflow occurs. An overflow is a problem in digital computers because the width of register is finite. Many computers detect the occurrence of an overflow and when it occure, a corresponding flip flop is set which can then be checked by the user.

When two unsigned numbers are added, an overflow is deleted from the carry out of the most significant position.

In case of signed numbers leftmost negative numbers are in 2’s complement form when two signed numbers are added, the sign bit is treated as part of the number and the end carry does not indicates overflow.

Eg.

Carries 01

```
+70    0      1000110

+80    0      1010000

+150   1      0010110
```

Carries 10

-70	1	0111010
-80	1	0110000
-150		1101010

Since the sum of the two numbers is +150 and -150, it exclude the value range +127 to -128 so it cannot be accommodated in a 8 bit register.

If the carry out of the sign bit position is taken as the sign bit affect result, the 9 bit answer so obtained will be correct. Since the answer cannot be accommodated within 8 bits an overflow occurs.

Overflow detection:-

An overflow condition can be detected by observing the carry into the sign bit position and carry out of the sign bit position. If these two carries are hot equal an overflow condition is produced.

If the two carries are applied to an exclusive or gate, an overflow will be detected when the output of the gates is equal to 1.

Decimal fixed point representation:-

The representation of decimal numbers in register is a function of the binary code used to represent a decimal digit. A pour bit decimal code requires four flips flops per each decimal digit.

The representation of 4385 in BCD requires 16 flip flops, four flip flop for each digit. The number will be represented in a register with 16 flip flops as follows.

0100 0011 1000 0101

By representing number in decimal a considerable amount of storage space is wasted since the number. The number of bits needed to store a decimal number in a binary code is greater than the number of bits needed for its equivalent binary representation.

However there are some advantages of use of decimal representation.

- ⇒ Computer input and output data generated by people who use the decimal system.
- ⇒ Some application require for input and output of decimal data.

The representation of signed decimal number in BCD uses either singed magnitude system on signed complement system. The sign of decimal number is represented with four bits a plus with four zeros (0000) and which is 1001.

Eg. +375

+ - 240

+135

10's complement of 240

0 3 7 5 (0000	0011	0111	0101) BC
9 7 6 0 (1001	0111	0110	0000)BC
0 1 3 5 (0000	0001	0001	0101)BC

Addition and subtraction with signed magnitude data:-

Operation	Add magnitude	Subtracts magnitude when A>B A<B A+B
(+A) + (+B)	+(A+B)	
(+A) + (-B)		+(A-B) -(B-A) +(A-B)
(-A) + (+B)		-(A-B) +(B-A) +(A-B)
(-A) + (-B)	-(A+B)	
(+A) - (+B)		+(A-B) -(B-A) +(A-B)
(+A) - (-B)	+(A+B)	
(-A) - (+B)	-(A+B)	
(-A) - (-B)		-(A-B)+(B-A) +(A-B)

Let the magnitude of two number be A and B. when the signed number are added or subtracted, there are eight different conditions to consider, depending on the sign of the numbers and the operation performed. That is shown in the first column of the table. In the above table the second column shows the actual operation to be performed with the imagination of the numbers. The last column is needed to prevent a negative zero.

Floating point representation:-

The floating point representation of a number has two parts. The first part represents a singed, fixed point number called the mantissa. The second part designed the position of the decimal can binary point and is called the exponent. The floating point is always interpreted to represent a number in the following form.

$m \times n^e$

only the mantissa m and exponent e are physically represented in the register (including their signs).

Eg. Decimal Number = +6 13 2.789 is represented in floating point with a fraction and an exponent as follows.

Fraction	exponent
+0.6132789	+04
Represented in first register	represented in second register.

A floating point binary number is represented in a similar manner except that is uses base n for the exponent.

Eg. Binary number +1001.11 is represented with a 8 bit fraction and 6 bit exponent as follows.

Fraction	exponent
0100110	000100
Signal binary point	sign bit net shown.

The floating point number is equivalent to

m x 2^e = $(.100110)_2$ x 2^{+4}

Normalization:- A floating point number is numerical example for Binary Multiplier

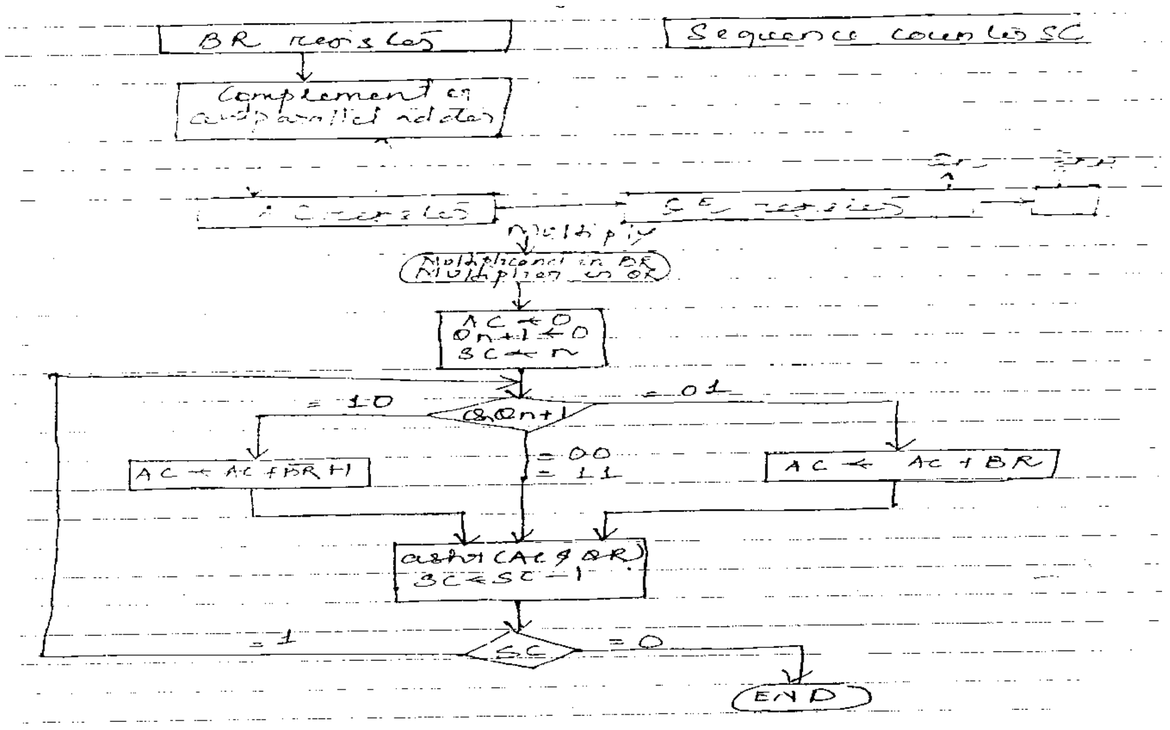
Multiplicand	B=10111	E	A	B	SC
Multiplier is	Q	0	00000	10011	101
Qn =1, add B		0	10111		
First partial product		0	10111		
Shift right	EAQ		01011	11001	100
Qn = 1, add B		1	10111		
second partial product		0	00010		
Shift right	FAQ	0	10001	01100	011
Qn = 0, shift right	EAQ	0	01000	10110	010
Qn = 1, Shift right	EAQ		00100	01011	001
Qn = 1, add B		0	10111		
Fifth partial product		0	11011		
Shift right	EAQ		01101	10101	000
Final product in	AQ		=	011011 0101	

Booth multiplication Algorithm

Booth algorithm gives a procedure for multiplying binary inlayers in signed – 2'S complement representation.

Booth algorithm requires examination of the multiplier bits and shifting of the partial product. Poison to the shifting, the multiplicand may be added to the partial product subtracted from the partial product or left unchanged according to the following twles.

1. The multiplicand is subtracted from the partial present upon encountering the first least significant of in a string of 1's in the multiplier.
2. The multiplicand is added to the partial product upon encountering the first OC provided that there was a pervious of in a staring of O's in the multiplier O. the partial product does not change when the multiplier is identical to the previous multiplier bit.



Prorate algorithm for multiplication of ringed – 2 complement number

Example of multiplication with broth Algo.

= (-9) x (-13) = +117

BR = 10111

BR +1 = 01001

Qn. Qn+1	Br	AC	QR	Qn+1	SC
	Initial	00000	10011	0	101
1 0	Subtact BR	01001			
		01001			
	Ashr	00100	11001	1	100
1 1	Ashr	00010	01100	1	011
0 1	Add BR	10111			
		11001			
	Ashr	111000	10110	0	010
0 0	Ashr	11110	01011	0	001
1 0	Substact BR	01001			
		00111			
	Ashr	00011	10101	1	000

Livonian algorithm:-

Divirian of two fixed point binary numbers in signed magnitude representation in done un paper and pencil by process of successive compare, shift and subtract operation.

Eg. Dividend = 0111000000

Divisor = 10001

Division	-	110010 quotient = Q
10001 01110000000	Dividend = A	
01110	5 bits of A< B	
011100	6 bits of A>/B	
-10001	shift right B and subtract entities Q	
-010110	7 bits of reminder >/ B	
--10001	shift right B and subtract; enter 1 in Q Shift	
--001010	reminder <B, enter 0 in Q, right B	
---010100	reminder >/ B	
---10001	shift right B and subtract in Q.	
----000110	Reminder < B	
-----00110		

<u>Hardware implementation for signed magnitude data:</u>

When the division is implemented in a digital computer it is convenient to change the process rightly. Instead of shifting the division to the right, the dividend, or thus leaving the two numbers in the required relative position. Subtraction may be achieved by adding a to the 2's complement of B. The information about the relative magnitudes is then available from the end carry.

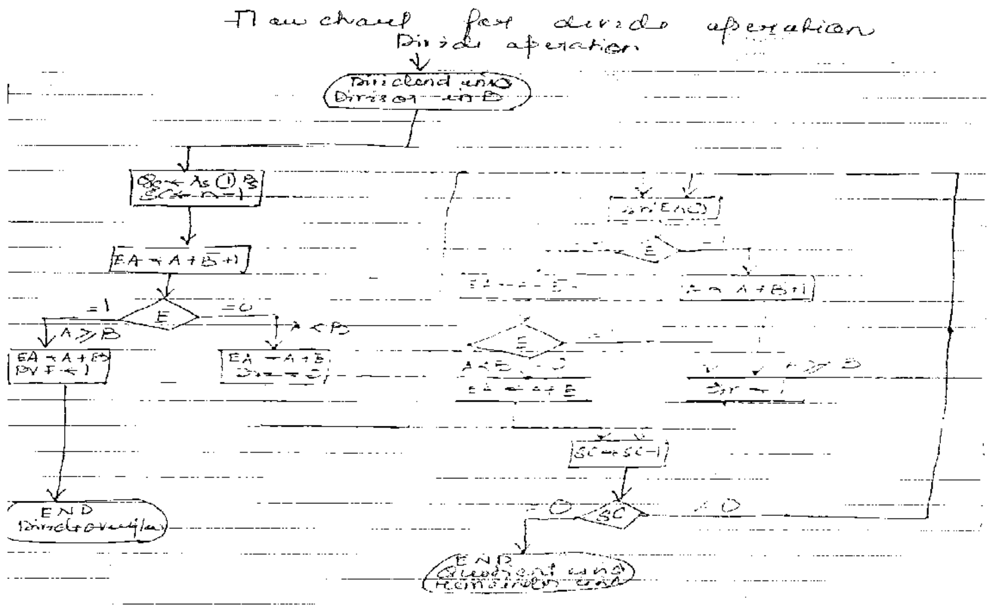
Eg. Dividend = 01110000000

Division B = 10001

B + 1 = 01111

	F	A	Q	SC
Dividend	0	01110	00000	101
Sht EAQ	0	11100	00000	
Add 5+1		01111		
E = 1	1	01011	00001	
She EAQ	0	10110	00010	100
Add 5+1		01111		
E =1	1	00101		

Set Qn = 1	1	00101	00011	011
She EAQ	0	01010	00110	
Add 5+1		01111		
E=0, Qn = 0		11001	00110	
Add B		10001		
Restore remainder	1	01010		010
She E^Q	0	10100	01100	
Add B+1		01111		
E=1	1	00011		
Set Qn = 1	1	00011	01101	001
She EAQ	0	00110	11010	
Add B+1		01111		
E=0, Qn = 0		10101		
Add B	0	10001	11010	
Restore remainder		00110		
Neglect E	1	00110		
Reminder in	A	11010		
Quotient in Q	Q			



Divide overflow:-

When the dividend is twice as long us the divisor, the condition for overflow can be stated as follows.
A divide:- Overflow condition occure of the high – order that bit of the dividend constitute a number greater than o equal to the division. Another problem associated with division is the fact that a division by zero must be avoided. The divide overflow condition takes care of this condition as well this occure because and dividend will be greater than or equal to a divisor which is equal o zero overflow condition. When a special flip flop divides overflow grip flop (pvt) is act.

Floating point arithmetic operations:-

A floating point number in computer registers consists of two parts.

A mantissa m an exponent e. the two parts represent a number obtain from multiplying on times a racier or raised o the value of e, thus

m x n^2

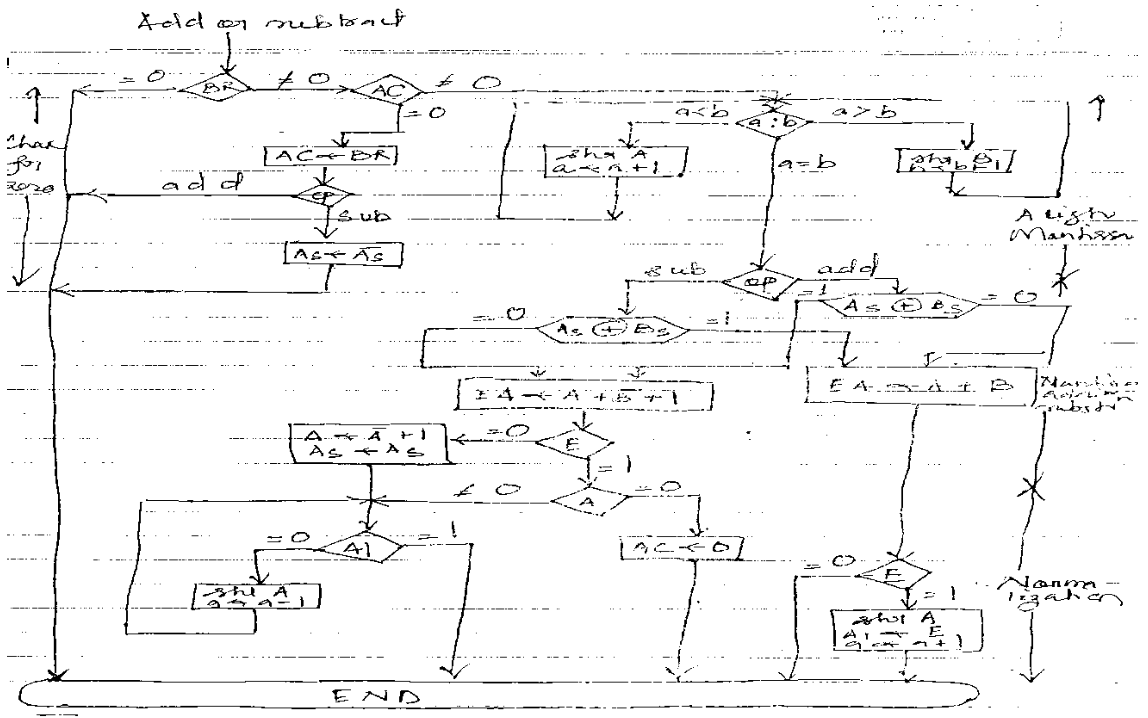
the mantissa may be fraction ad integer. The location of the raider point and the value of the radix or are assume and are not included in the registers.

A floating point number is norm ling if the most significant digit of the mantissa is han zero. Floating point representation increases the range of numbers that can be accommodated in a given register.

Addition and subtraction

During addition or mutation the two floating point operands are in AC and BR. The sum or obi glance, i.e. formed in the AC. The algorithm can be divided into four consecutive parts.

1. Check per zeros.
2. Design the mantissa
3. Add or subtract the mantissa
4. Normalize the zero it.

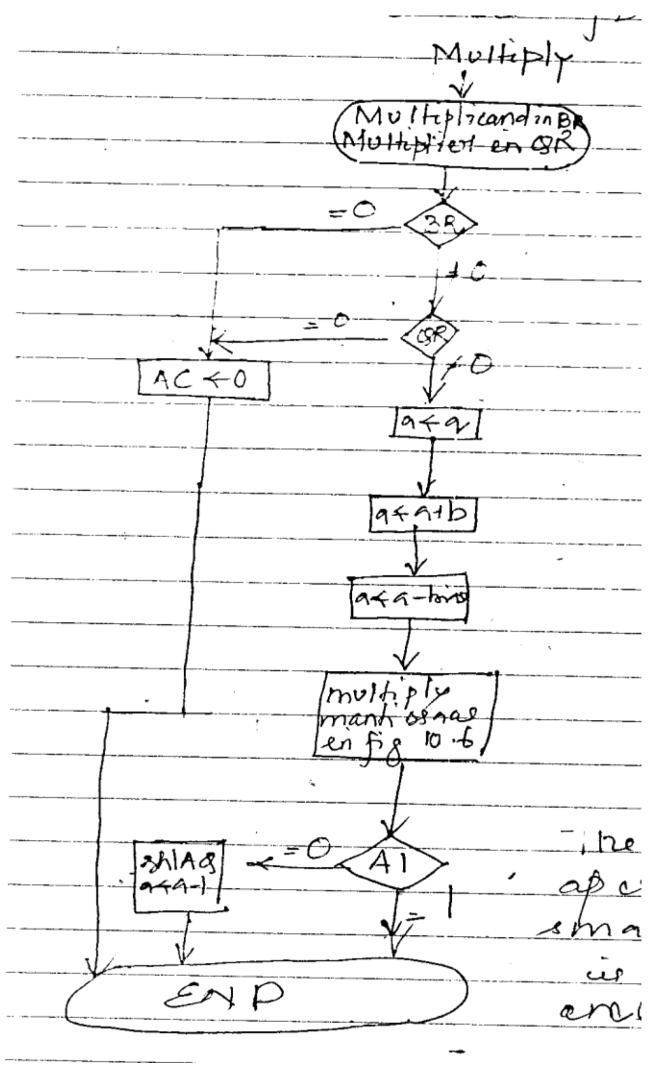


Addition Substraction of Floating Point Numbers

Multiplication:-

The multiplication algorithm is subdivided into flow parts.

1. Check firi zeros
2. Add the exponent
3. Multiply the mantissa
4. Normalize the product.



The smallest normalized operand is 0.1 so their smallest possible product is 0.01. Therefore ante an leading zero may occur.

Follow representation employed in many computers is known as a biased exponent. In this representation, the sign bit is removed from being a separate entities. The bias is a positive number that is added to each exponent as the floating point number is formed. So that internally all exponents are positive.

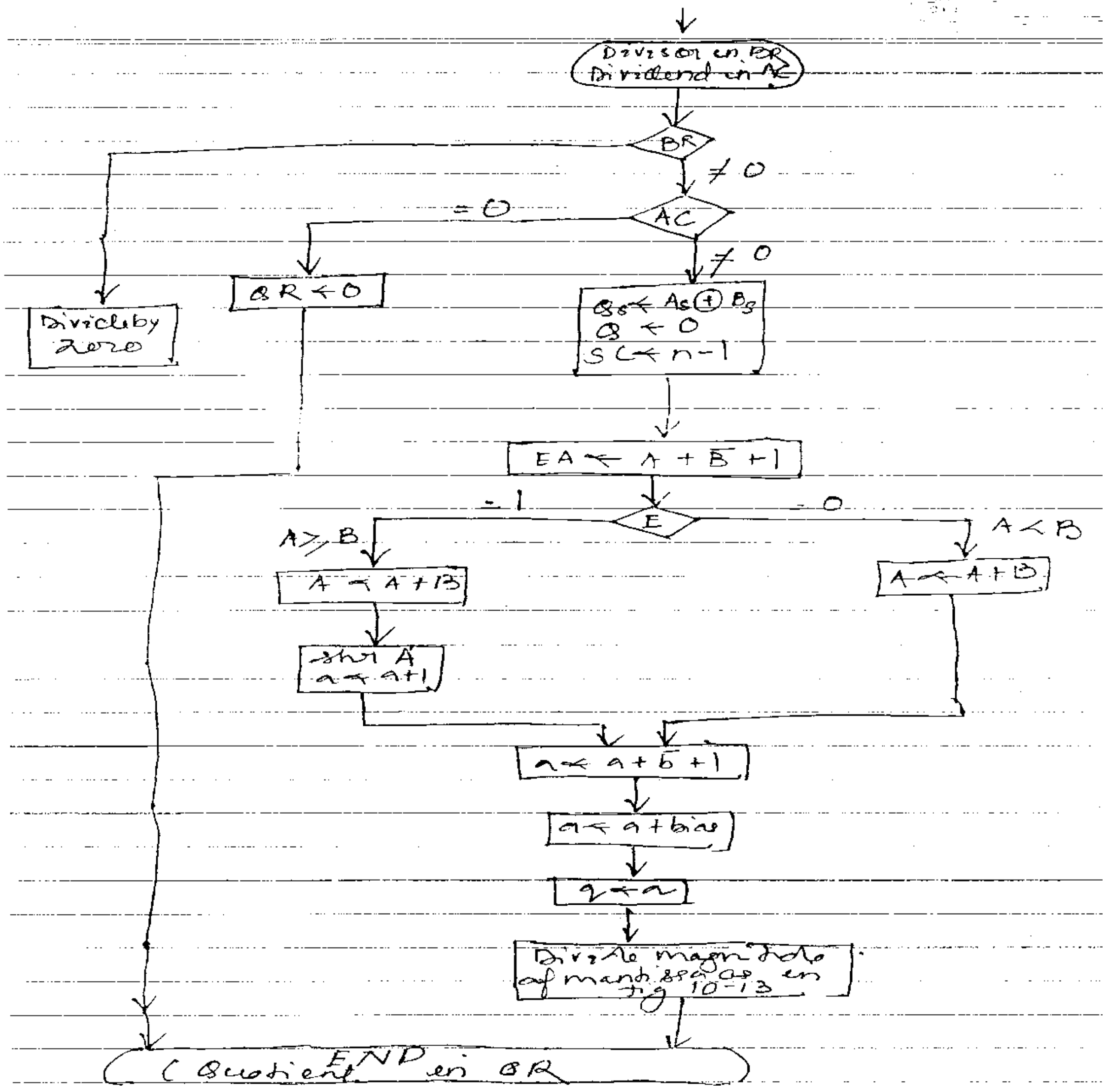
Eg. Consider an exponent that ranges from – 50 to 49. Internally, it is represented by two digits (without a sign) by adding to it a bias of 50. The exponent register contains the number e + 50 where e is the actual exponent.

Division

The division algorithm can be sub divided into five parts.

1. Check for zeros.
2. Initially registers and evaluate the sign.

3. Align dividend
4. Subtract the exponents
5. Divide the mantissa.

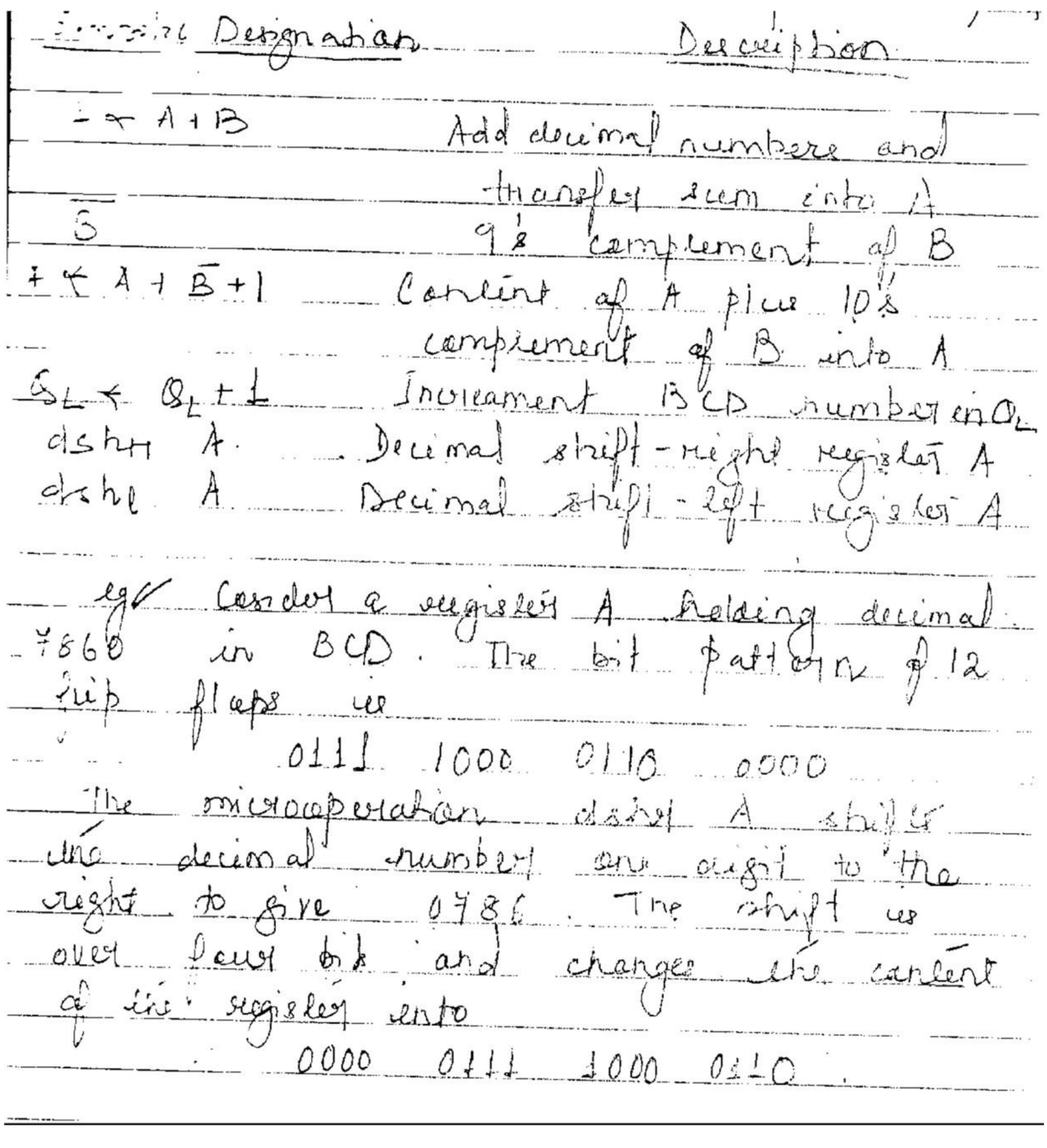


Decimal arithmetic operations

Decimal number in BCD are stored in computer register in groups of your bits. Each 4 bit represent a decimal digit and must be takes as a unit when performing decimal micro operation.

Decimal arithmetic micro operation symbol

Symbolic designation



Symbolic Designation	Description
$A \leftarrow A + B$	Add decimal numbers and transfer sum into A
$\overline{B}$	9's complement of B
$A \leftarrow A + \overline{B} + 1$	Content of A plus 10's complement of B into A
$Q_L \leftarrow Q_L + 1$	Increament BCD number in Q_L
dshr A	Decimal shift-right register A
dshl A	Decimal shift-left register A

eg/ Consider a register A holding decimal 7860 in BCD. The bit pattern of 12 flip flops is

0111 1000 0110 0000

The microoperation dshr A shifts the decimal number one digit to the right to give 0786. The shift is over four bits and changes the content of the register into

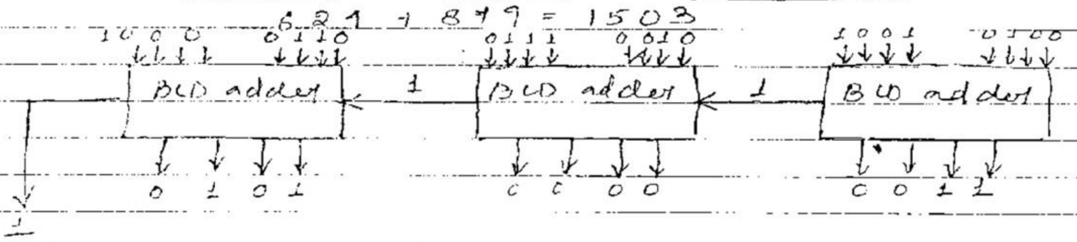
0000 0111 1000 0110.

Addition and Subtraction

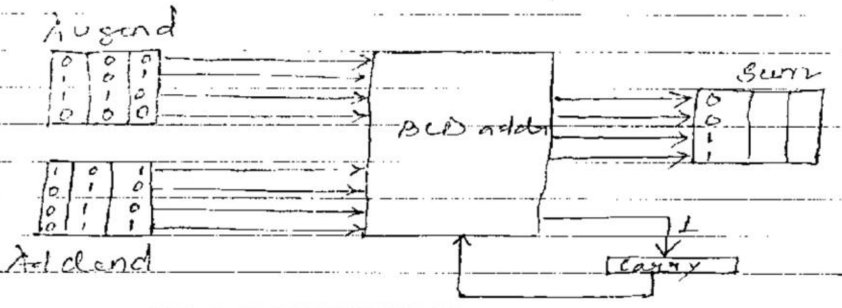
Decimal data can be added in three different ways:

① Parallel decimal addition

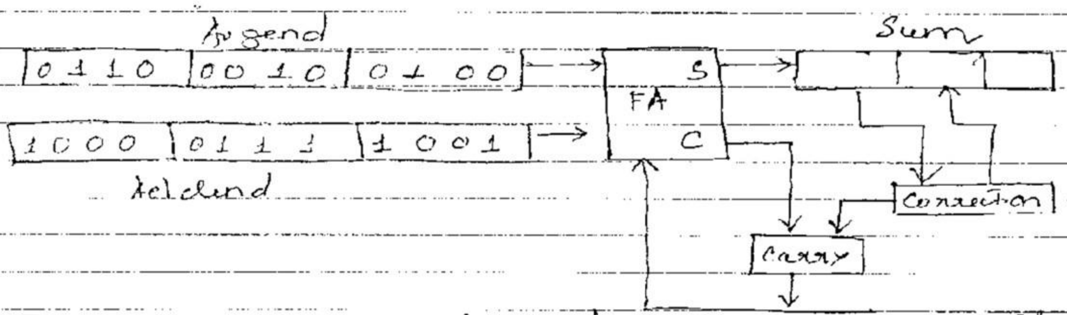
624 + 879 = 1503



② ~~[illegible]~~ serial, bit parallel decimal add



③ All serial decimal addition



The binary sum formed after four shifts must be corrected into a valid BCD digit by adding 0110 if it is > than 1010

Questions:

1. **Writes short notes on look ahead carry generator.**
2. **Give an example of binary subtraction.**
3. **Write short notes on gray code.**

CHAPTER-4

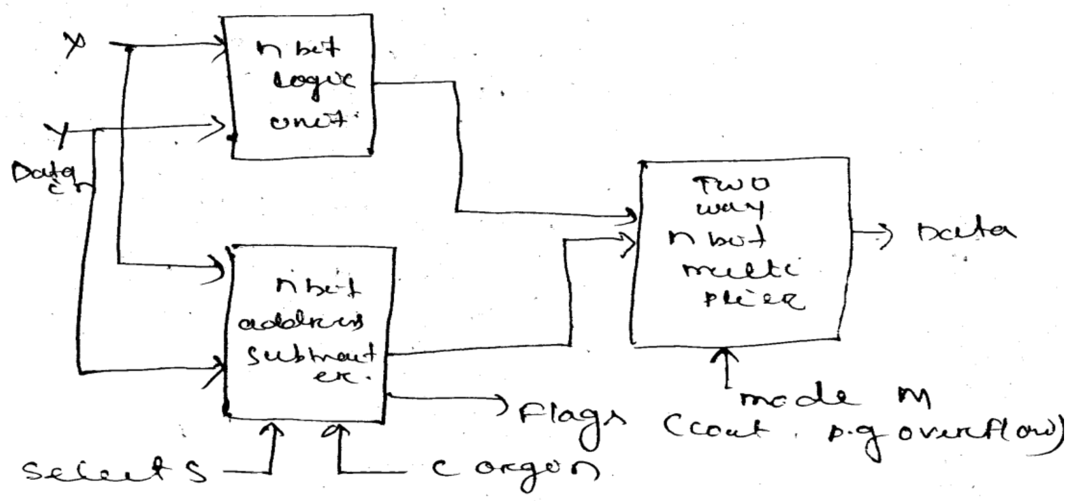
PROCESSOR SYSTEM

Processor unit:-

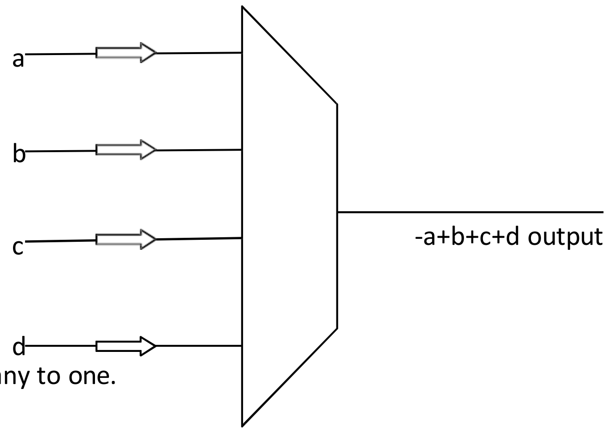
Design of ALU:- The ckt that carryout the data processor instruction. The arithmetic logic unit. The ALU using combinational ckt that performs, subtraction ward base logical operation multiplication and division. This can be complemented some time using auxiliary arithmetic unit called coprocessor.
80286 – 80287
80386 → 80387

Combinational ALU:-

Basic ALU n a bit:-



The fig shows the simple ALU combine with adder, subtractor with word base logic functions such as AND, Ex-OR, NOT for fixed point instruction.
Then the carry in select way.
The mode control line m attach to the two way n bit multiplexer determined the type of operation that is arithmetic or logical to be perform by the ckt.
The result in channel out thought the output bus Z. the select control line determines the specific operation to be perform by the sub unit. The multiplexer.



Many to one.

The ALU performs the bit close operations for input data lines. The maximum number of distance. Logical operation is $16-2^4 = 16$.
So the select bus needs to be size of 4, $2^2 = 4$.
M1 = xy
M1 = xy
M1 = xy
M1 = xy
Sum of the product.
P(x,y) = xys3+
Dy so
Sy sit
Sy sn+

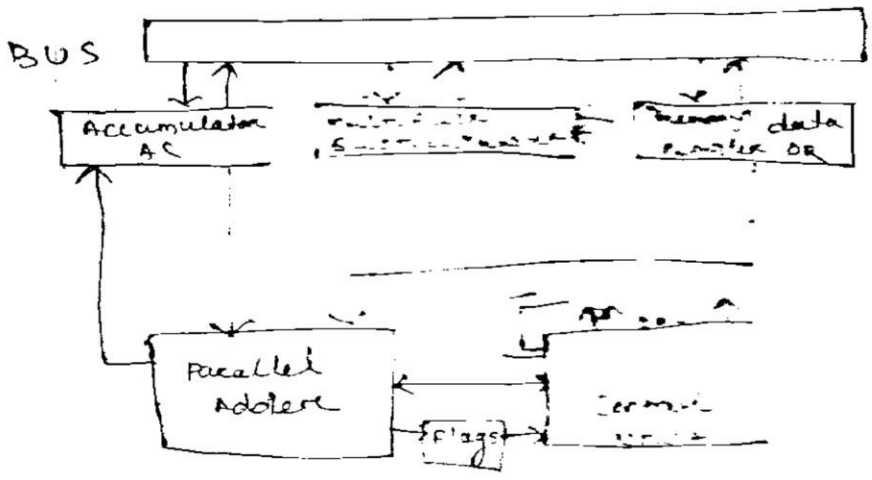
Arithmetic

Operations	Resister usage
Addition	AC = AC+DR
Subtract	AC = AC-DR
Multiply	AC-MQ = DRxMQ
Division	Ac-MR = MQ/DR

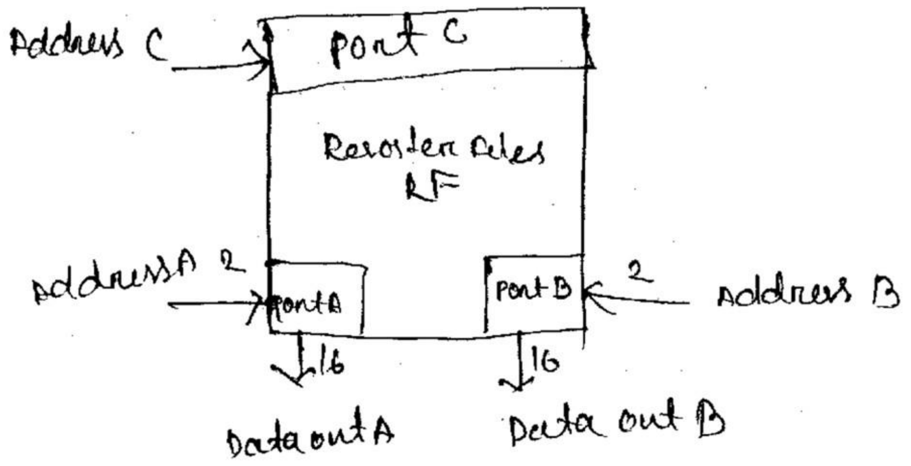
The multiplication and division is done using one sequential digit by digit sift and add or subtract algorithm. The one word resisters namely accumulator, multiplier/quotient (MQ) and the data resister (DR) are use for operand storage.

AC and MQ are organize as a single register and at capable of sifting right or lect. The adder substractor unit and the derived input spore AC and DR placed its result in AC. The MQ resister stores multiplier during multiplication and quotient division.

Basic sequential ALU:



Register files:-

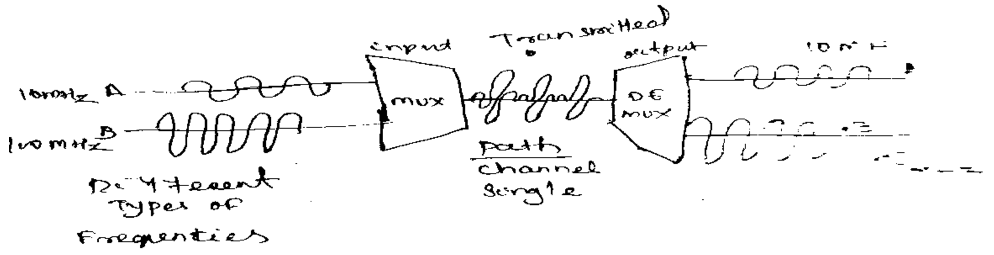


In modern CPU have a set up general purpose register R02, M-1 called resister file replacing AC-(Accumulator), DR-(Data Resister), AR – (Address Resister).

Each resister are subscript Rm. Rm register file is individually quick address subscript m0.
Sta R1, R2
Add R3, R1 R2
R3 = F(R1, R2)
R2 = F(R1, R2).

The register file performs its function it random access memory (RAM).
A multiport resister file is build a set up resister and multiplexor and A multiplexor combination. Allows the data to be operation transform any thus register to output port read and various input port to register.

The figure show 3.4 resister file read port ~~and~~ A and B , and Right write takes place using port C .



many input and output – [illegible] .

(DSP)

Data path design:-

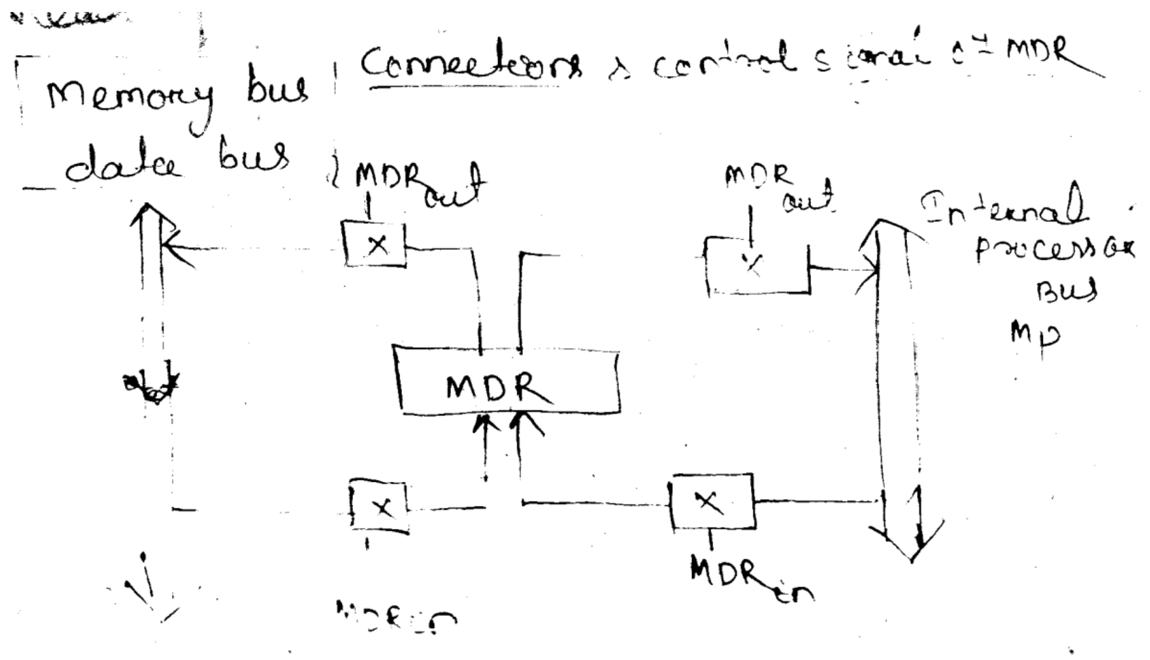
⇒ It is a unit for implementing logical and fixed point operation it consists of a register file and a combinational ALU capable of addition or subtraction. The entire sequential ALU for fixed point no. 5 can be built on a single IC. The ALU can be designed for expansion to handle operand of bigger size using (i) special expansion bit slice processor, (ii) Temporal expansion bit slice processor.

1) **Special expansion bit slice processor:-**
   This connects K-copies and M-bit ALU to form a single ALU of processing K-M-bit directly for addition the resulting array like circuit is said to be bit slice because each component ALU concern the processor a separate slice form each km bit operand.
2) **TEBSP:-**Temporal expansion bit slice processor:- this uses n copy of n bit ALU in the manner of serial adder to perform and operation on KM-bit word in k conjugative pluck cycles. Here the ALU process a separate m bit slice or each operand. This is called multi cycle processing. Here the data buses and register rules of individual slices are effective input to increase the size from 0 bit to 16 bit.

Basic memory operation:-

All data and instructions are stored in memory before and after their used. These data should be transfer back and fourth using read, write operations in memory.

Read operation:-



The information to be fetched from memory may represent on instruction in a program. The processor has to specify the address of the memory location where this information is stored on request read operation the processor transfer the address to MAR. The requested data are receive from memory and store in MDR (Memory data register).

During the memory read and write operation the timing of internal processor operations most be coordinated with the response of the address device on memory bus.

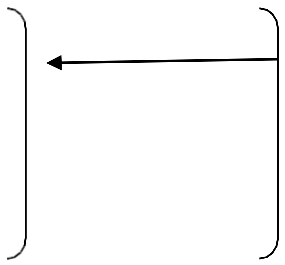
The processor complete one internal data transfer in one clock cycle. The control signal called memory function completed is used for thus purpose.

Ex: MOV	R3	R4
	11	11

MAR←[R3]
R4←[MDR]

Start a read operation on the memory bus. Wait for MRC response from the memory. Load the MDR from the memory bus.

1. R3 out, MAR in, READ
2. MDR in, MFC
3. MD Rout, R4 in



Memory read operation using 3 steps:-
Memory write operation using 3 steps:-

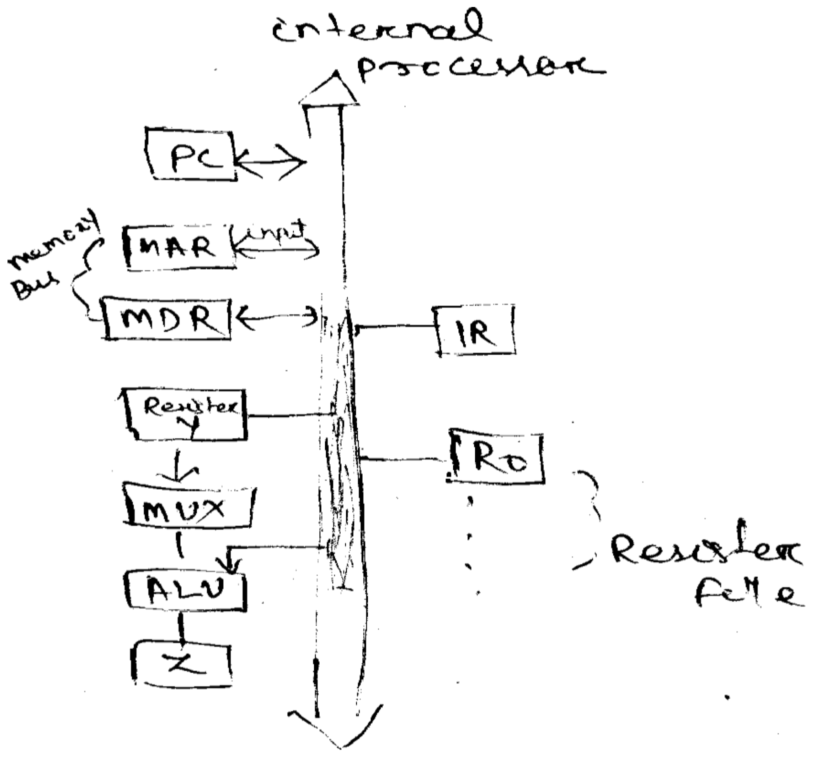
1. R3 out, MAR in
2. R4 out, MDR in, WRITE
3. MDR out, MFC

The memory address where the data is to be written is loaded into MAR. the data should be written are loaded into MDR and a write command issued until the memory operation is completed [MFC].

Complete instruction execution:-
ADD (R4), R2

Executing this instruction requires the following:
Steps:-

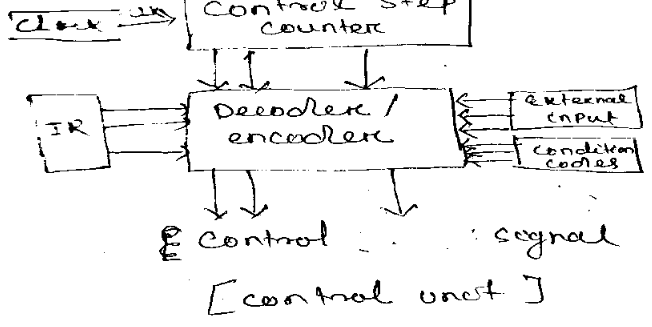
1. Fetch the instruction
2. Fetch the first operand
3. Perform the addition
4. Load the result into R2

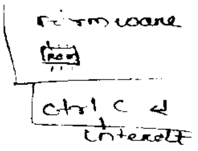


1. Pcout, MAR in, Read, Select-4, ADD, Zin
2. Zout, Pcon, Yin, Mfc
3. MDR out, IRin
4. R4, MAR in, READ
5. R2out, yin, MFC
6. Rout, Select Y, ADD, Zin
7. Zout, R2in, End

PC- Program counter points to the address of the next instruction to be executed.

Hardware Control:-





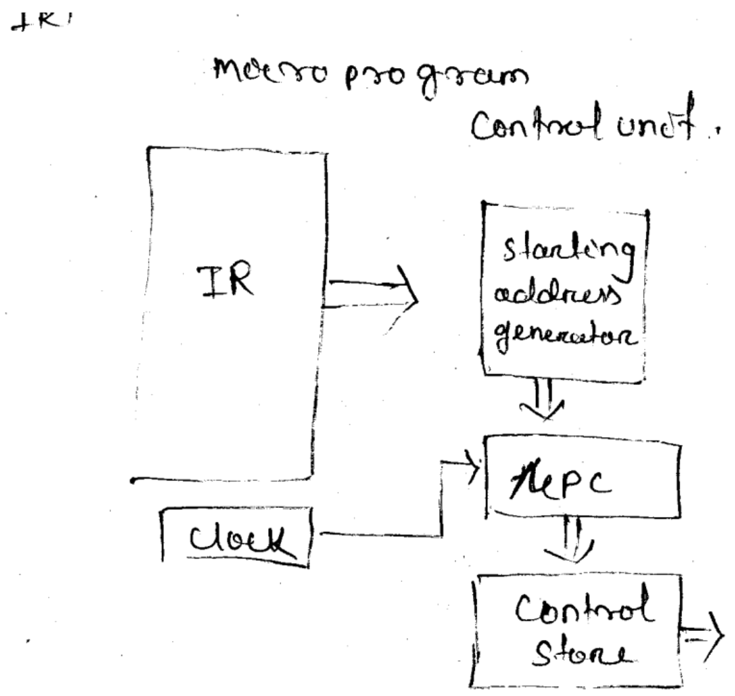
For an instruction to be executed the processor most generate the control signal in proper sequence. i.e. either in Hardware control on microware control.
In the control unit each step of the sequence is completed in one clock period. A counter is used to keep track of the control steps.

The control signals are determined by the instruction contend in the following

1) Step counter
2) Condition code flag
3) External input singles. Such as MFC inter request.

<u>Microprogrammed Control:-</u>
In microprogrammed control a program similar to the machine language program generates the control signal. A control word (CW) represents various control signals. Each of the control steps in the control sequence of an instruction detain a unique combination of 1s, 0s in the control word. The individual control word in the micro routine is refer as micro instructions. The micro routin for all instructions of a computer are stored in a special memory called the control store. The control unit can generate the control signal for any instructions.

A microprogram counter is used to read control words from the control stored. The starting address generator is loaded to micro program counter every time a new instruction is loaded into instruction register (IR).



The control signal are delivered to various pants of the processor in the current sequent.
MOV R1, R2 ....
ADD R1, R2
SUB
HLT

The MPC (micro Program counter) is coded with.

1. Starting address of the micro instructions when a new instruction is loaded.
2. Branch address when a branch micro instruction is encountered and the condition is satisfied.
3. The address of the first control load of the next instructions when end is encountered.

```
001    MOV    R1 R2
       ADD    R1 R2
 00    SUB
       JMP
010    END
```

CONTROL SIGNALS

To execute instructions, the CPU must have some means of generating control signals in the proper sequence computer designers have used a wide variety of techniques to solve this problem. Most of these techniques, however, fall into one of two categories.

1. Hardwired control
2. Micro programmed control

Hardwired control:-

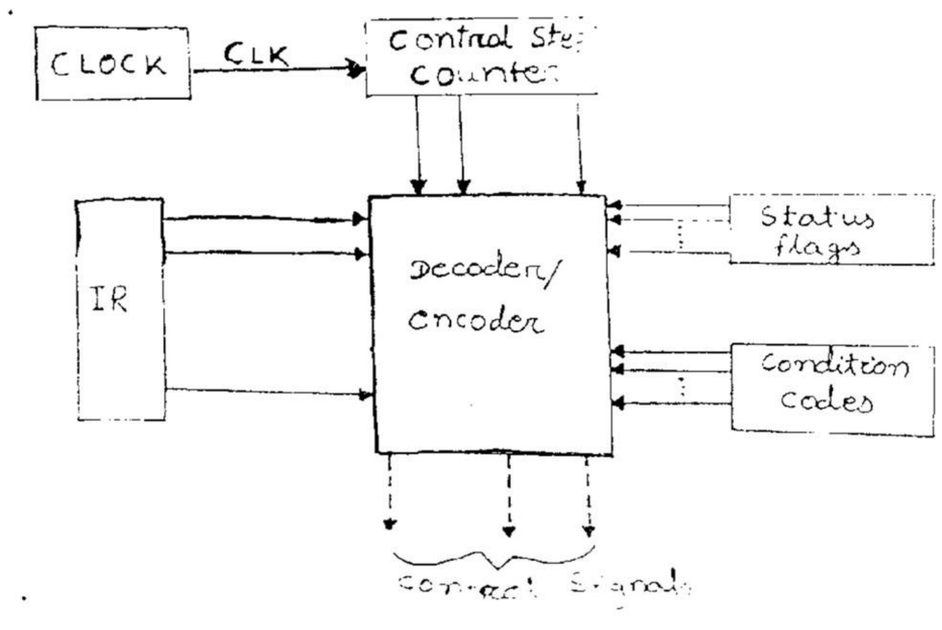


Fig Control unit design

The control unit based on the use of a counter device by a clock signal, CLK, as shown in the above figure. Each state or count of this counter corresponds to one of the control signal. The control signals are uniquely determined by the following information:

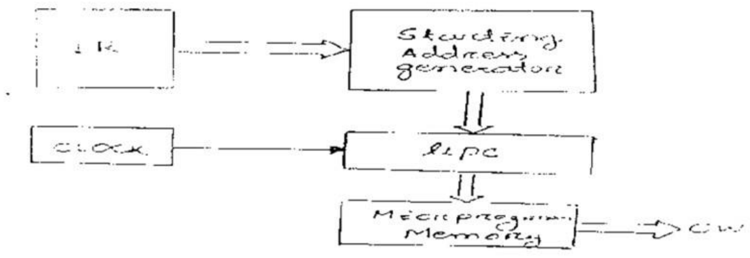
# contents of the control counter

# contents of the instruction Register

# contents of the condition code or other status flags.

In order to gain some insight into the structure of the control unit we will start by giving a simplified view of the hardwired involved. The decoder-encoder block is a combinational circuit that generates the required control outputs depending on the state of all its inputs. The instruction register contains the opcode of the current instruction which is used to determine which micro operation to perform during the execution cycle. The status flags are needed by the control unit to determine the status of the CPU and the outcome of the previous ALU operations.

Micro programmed control:-



let us start by defining a control word (CW) as a word whose individual bits represents various control signals. The micro program corresponding to the instruction set of computer are stored in a special memory called micro program memory. The control unit can generate the control signals for any instruction by sequentially reading the CWs of the corresponding microroutine form the micro program memory. To read the control words sequentially from the micro program memory a microprogram counter (MPU) is used. Every time a new instruction is loaded into the IR, the output of the block labled "starting odder generator" is loaded into the MPU. The MPU is then automatically incremented by the clock, causing successive micro instructions to be read from the memory. Hence the control signals will be delivered to various parts of the CPU in the correct sequence.

Questions:

1. **What is sequential ALU?**
2. **What is Micro Programming?**
3. **Write notes on Micro Programmed control unit.**
4. **List the advantages and disadvantages of Micro Programmed control.**

CHAPTER-5

MEMORY SYSTEM

Characteristics of memory:-

The characteristics of the memory is its capacity for internal memory, it is expressed in terms of byte or words. Word lengths are 8,16,32,64 bits. External memory capacity is topically expressed in terms of bytes.

Word:- it is the natural unit organization of memory. The length of word is topically equal to the number of bits used to represent a number and to the instruction length (normal comput 32 bit).

Addressable Unit:- The addressable unit is the word. 2A=N many system allow necessary in byte legal.

Unit of transfer:- This is the number of bit read out or written onto memory at a time.

Sequential:- Access most be made in a specific linear sequence.

Direct Access:- The individual blocks or records have a unique address based on physical location. Access is accomplished by direct access to reach to the final location.

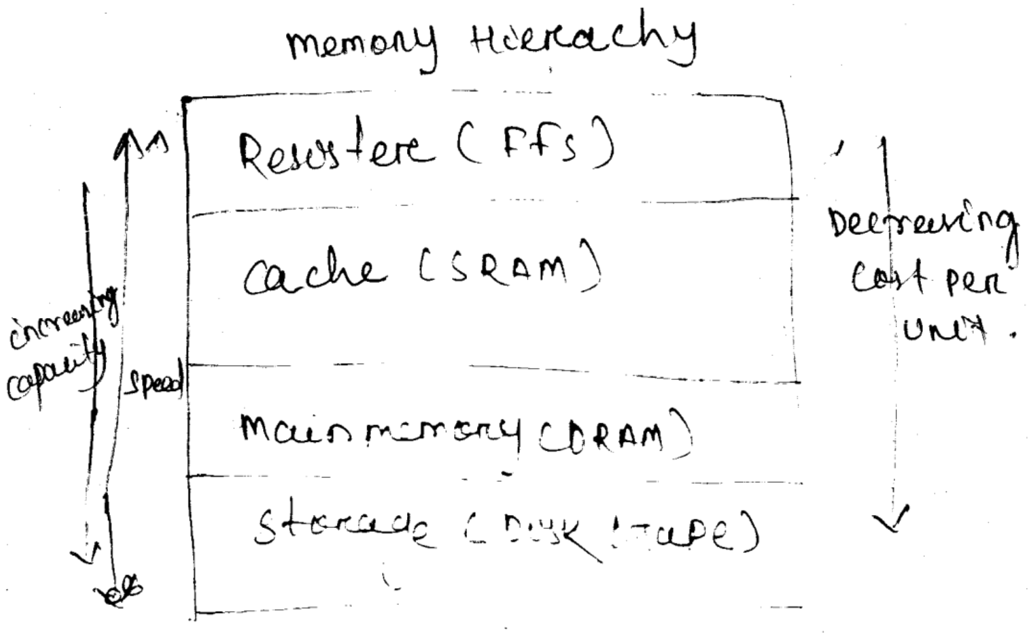
1	Name	ADD	Sal	Deign
2				
3				
4				
5000, 5				
100,000				

Random Access:- Each addressable location in memory has a unique, physically addressable mechanism any location can be selected at random and directly accessed. The main memory and cache memory used random access.

Three performance parameters are used:-

1. **Access time (latency):-** For random Access memory this is the time of takes to perform read or wrote operations. For Non RAM, access time is the time taken to position the read/write mechanism at the designed location.
2. **Memory cycle time:-** It is applied to random access memory and consist of access time and any addition time required before a second access can start. It is concerned with system bus.
3. **Transferred:-** This is the rate at which date can be transfer in and out of a memory unit. For RAM it is equal to 1 for non RAM it is
   TN = TA+N/R
   TN – Average time to read or write n bits.
   TA – Average Access time
   N – Number of bits.
   R – Transferees in Bits for second.

Memory Hierarchy:-
The principle of locality (temporal locality and special locality) states that programs do not access code and data uniformly.



Basic concept of memory:-

→ byte address

Word address

0	1	2	3
4	5	6	7
2^k-4			2k-1

2^k-4

Big-endian assignment

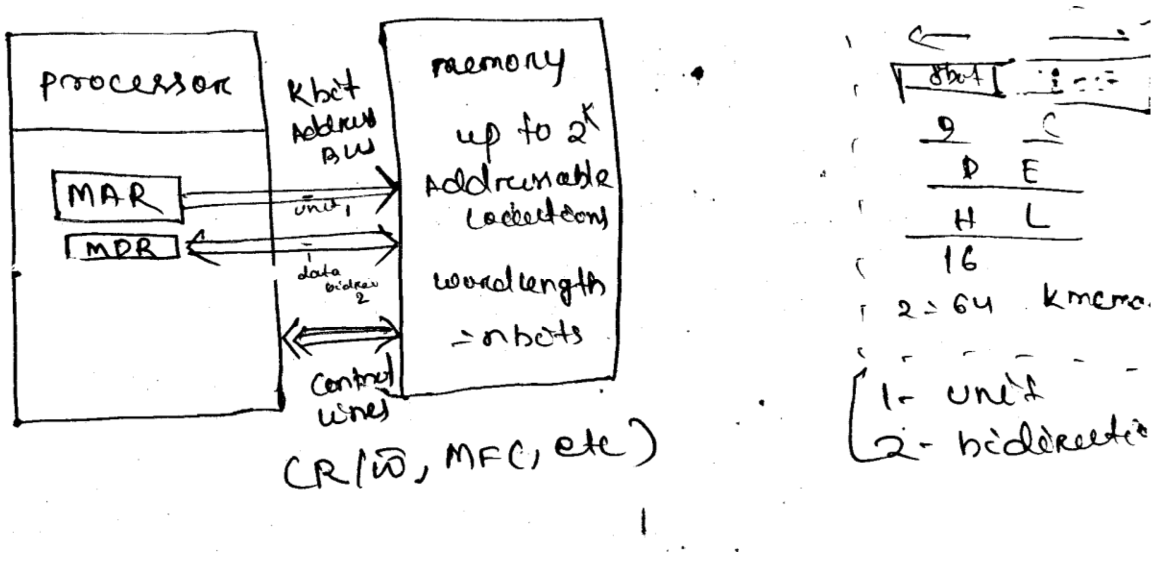
→ byte address

0	3	2	1	0
4	7	6	5	4
	2^k-1			2k-4

2^k-4

Little-endian assignment

The addressing screen the maximum size of the memory that can be used in any computer, a 16 bit address is capable of addressing upto 2^{16} = 64 K (kilo bite) memory location. When lower byte address is use for more significant bytes(left) of the work it is known as big-endian assignment. Otherwise the right most assignment is called little-endian. The memory is normally design to store and retrieve data in word length quantities. The data transfer between the memory and the processor takes place through the use of two processor registers. Namely 1) memory address register (MAR), 2) Memory data register (MDR).

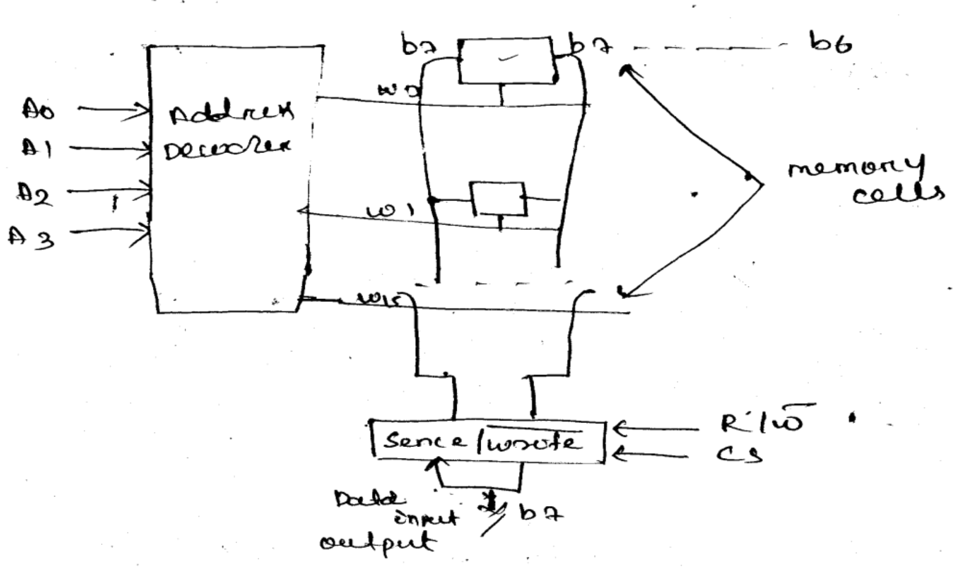


If MAR is k bit long, MDR is 8 bits long, than the memory context upto 2^k address locations and n bit of data transfer between the memory and the processor in a memory cycle. The read write bar (R/W) and memory function completed co-ordinates the data transfer. To read the data from memory by loading the address of the required memory location into MAR and setting R/W line to high.

The memory responds by putting the data from the address location on the data line with MFC signal. Similarly for write operation set R/W line to low. The processor writes the data into memory location by loading the address into MAR. the access are synchronies using a clock.

SEMICONDUCTOR RAM

Organization of memory chip:-

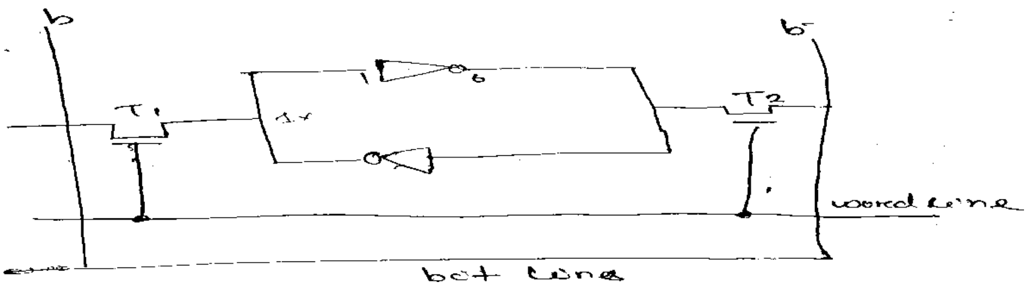


Memory cells are organized in the form of an arrays. Each cell is capable of storing 1 bit information each row of cells constitute a memory word of 8 bit. And the cells of a rows are connected to a common signal line called word line. Which is driven by address decoder. The sence/ Write ckt are connected data I/O lines. During read operations the ckt reads information the store in the cell and transmit to output lines. During write operations the sence write ckt receive the input information and store of in. the selected cell. The R/W signal satisfies the required operation. The CS signal line is selct a given chip in multi chip memory system.

Static RAM:-

The memory i.e. capable of retaining of current states is long as the power is applied is known as static RAM.

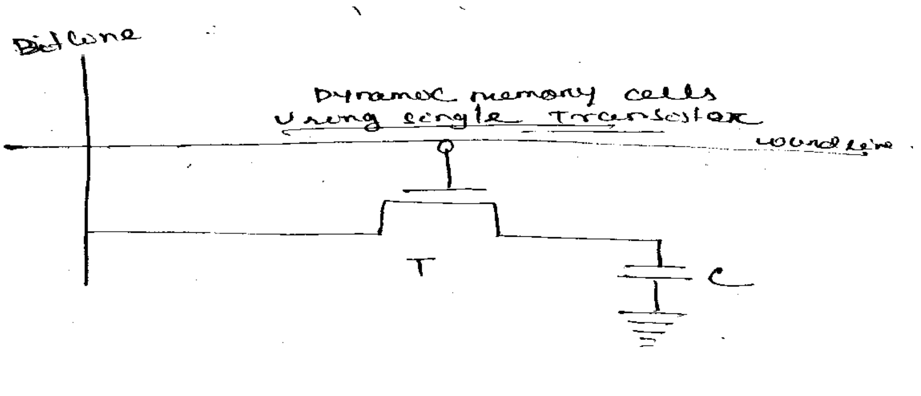
Static RAM



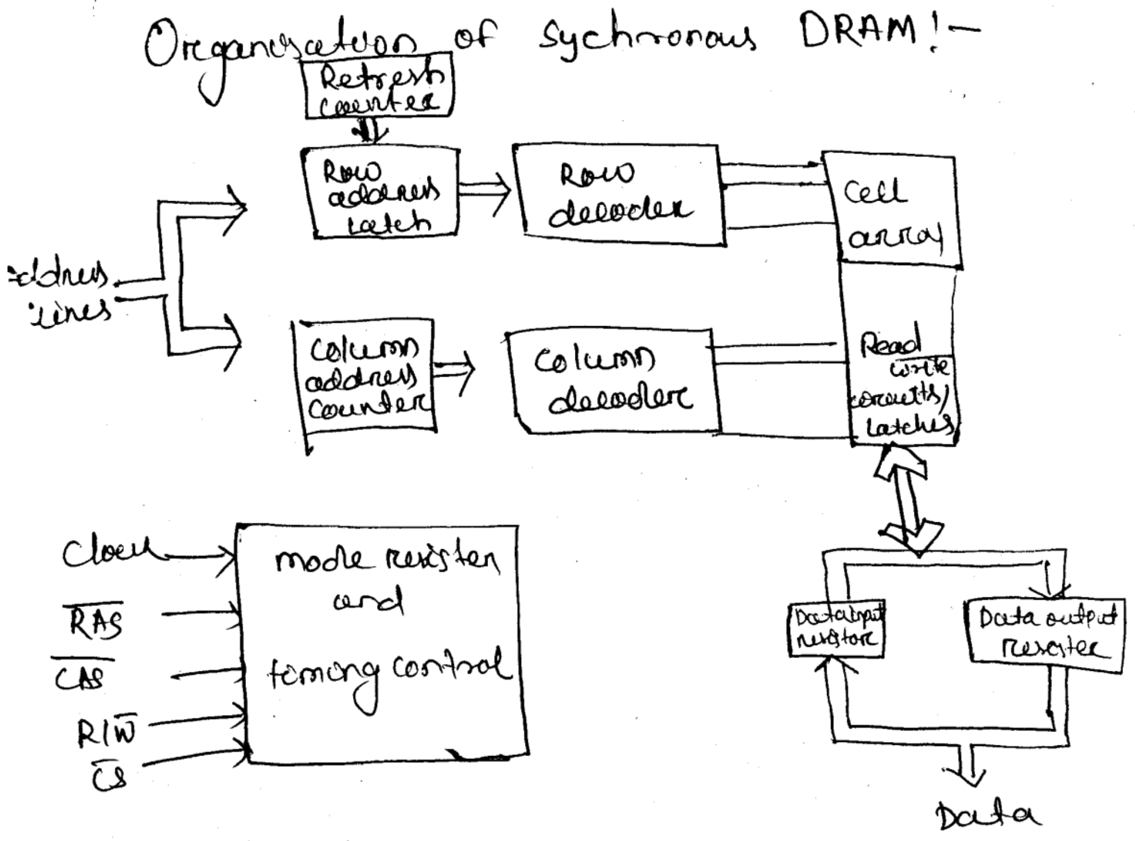
This latch is connected to 2 bit lines by transistors T1 and T2. These transistor can be open or closed. Under the control of word line. When the word line is at ground label, the transistor are turn off and latches returutten their original state. When the cell is in state 1 if the logic value at point x is one and at point y is zero. T0 read SRAM (Static RAM cell) the word is activated to close switches T1 and T2. If the cell is state 1 the signal on bit line is high i.e. b=1, b=0. If the cell is to be state zero, the switches T1 and T2 should be open.

Synchronous dynamic RAM (SDRAM)

SRAM are faster and their cells requires several transistor. Such cells don't written their state indefinitely and hence are called (DRAM) Dynamic RAM. Information is store in a dynamic memory cell that consist of a capacitor and a transistor. As the cell is required to store the information for longer period its contains are periodically refreshed.



Synchronous DRAM



Hence, the memory operation is directly synchronies by a clock. The address and data connection are buffer by means of register and the output is connected a latch.

In road operation the containt of all the cells in a selected row are loaded into these latches. Data held in the latch that corresponds to the selected column are transfer to the data output register.

1) Basic memory operation
   Read write operation
   Static RAM
   Dynamic RAM
2) Describe fix point addition, subtraction and floating point multiplication.
3) Write short notes (any two).
   i) Inter leaved memory.
   ii) Floating point arithmetic operations
   iii) Design of ALU
4) Indentify the advantages of parallel processing, what is a pipeline? Draw a space time diagram to show how an instruction is executed.

5) Identify the techniques/modes which makes data transfer to and from peripherals. Explain DMA method of data transfer by a suitable diagram.
6) Define the function of control unit. What do you means by micro program control? How it is different from hardware control?
7) What is the function of cache memory? What is memory mapping process 1 explain the structure the mapping process on cache memory.
8) Answer any five questions:
   a) Define virtual memory. How the memory table is used for mapping a virtual address?
   b) What are the basic memory operations? Explain how an instruction gets executed in a computer.
   c) Describe the architecture, function and working of I/W channels.
   d) What is the function of bus? Explain bus architecture.
   e) What do you mean by addressing mode? Explain different addressing mode.
   f) Describe Flynn's classification.
   g) Explain interrupt handling tech.

9) Answer all question:
   i. Name two replacement techniques in cache memory.
   ii. Define hit ratio.
   iii. What is the function of an interface?
   iv. What do you mean by interrupt?
   v. Define speed up.
   vi. What do you mean by performance measure?
   vii. Define instruction format.
   viii. What is strobe?
   ix. Define handshaking.
   x. What is register memory?

Virtual memory:-

Virtual memory is the most fundamental memory management concept to be implemented for memory management functions such as space allocation, program relocation, code sharing and protection.

The key idea is to allow or user program more memory locations than those available in a physical memory. This concept work and as follow:-

A virtual address is generated by a user program and the set of virtual address constitutes the virtual address space.

Hence it was necessary to divide the program into small portion called overlay, to fit them into the primary memory. A programmer has to design overlays to make them independent to each other with this kind of provision, one can successively bring each overlay into the main memory and execute them in a sequence. In the a system using virtual memory there of virtual address. Space is much larger than the physical address space.

A virtual memory system performs a series of mapping operations that mechanizes the process of overlay generation.

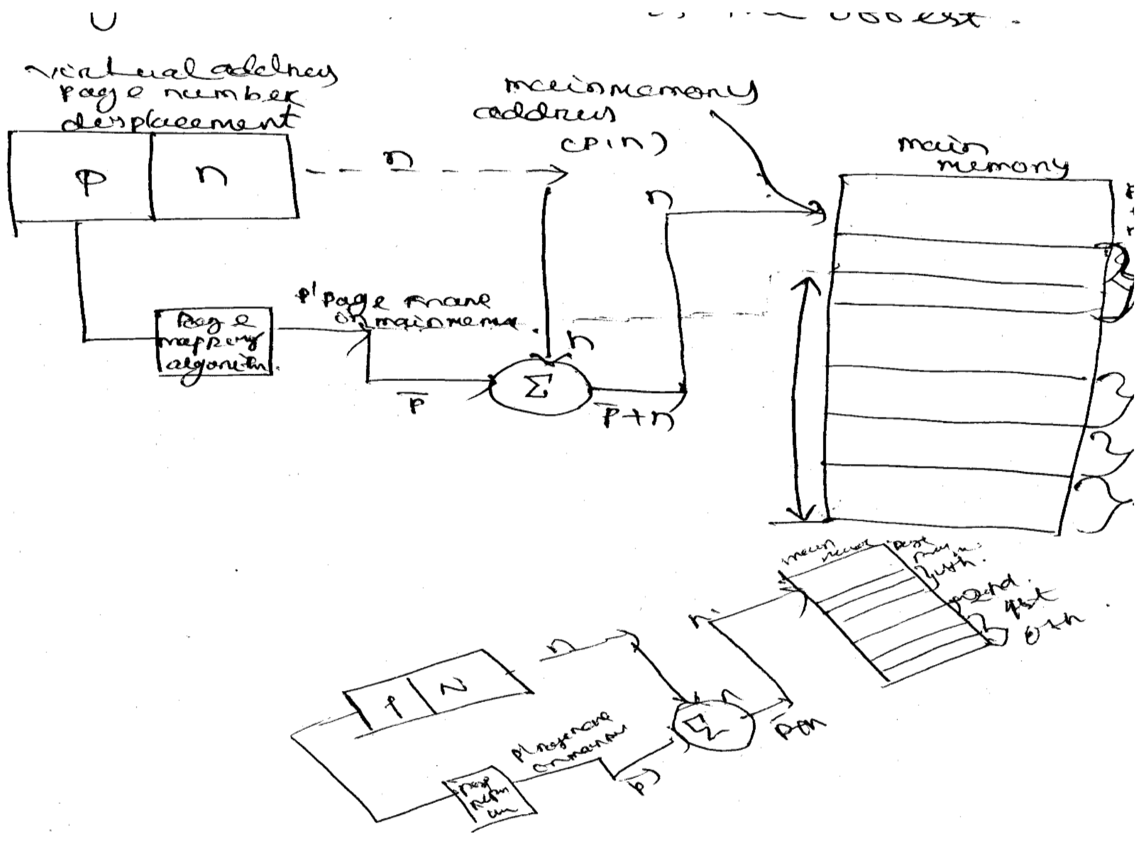
A virtual memory system can be configured as follows:-

1. Memory paging
2. Memory segmentation.

<u>Memory paging:-</u>

In memory paging the virtual memory is divided into blocks of equal size. There are called pages. The physical memory is also divided into frames in the same way. A page has the same size as a frame and may be blocks of 512, 1027 or 2048 words.

In paging system each virtual address can be considered to be an ordered power of <p,n>, where p is the page number and n is the offset.



Mapping scheme for virtual address:-

Virtual address (K)	Page number
0-2	0
2-4	1
4-6	2
6-8	3
8-10	4
10-12	5
12-14	6
14-16	7
16-18	8
18-20	9
20-22	10
22-24	11
24-25	12
26-28	13
28-30	14
30-32	15

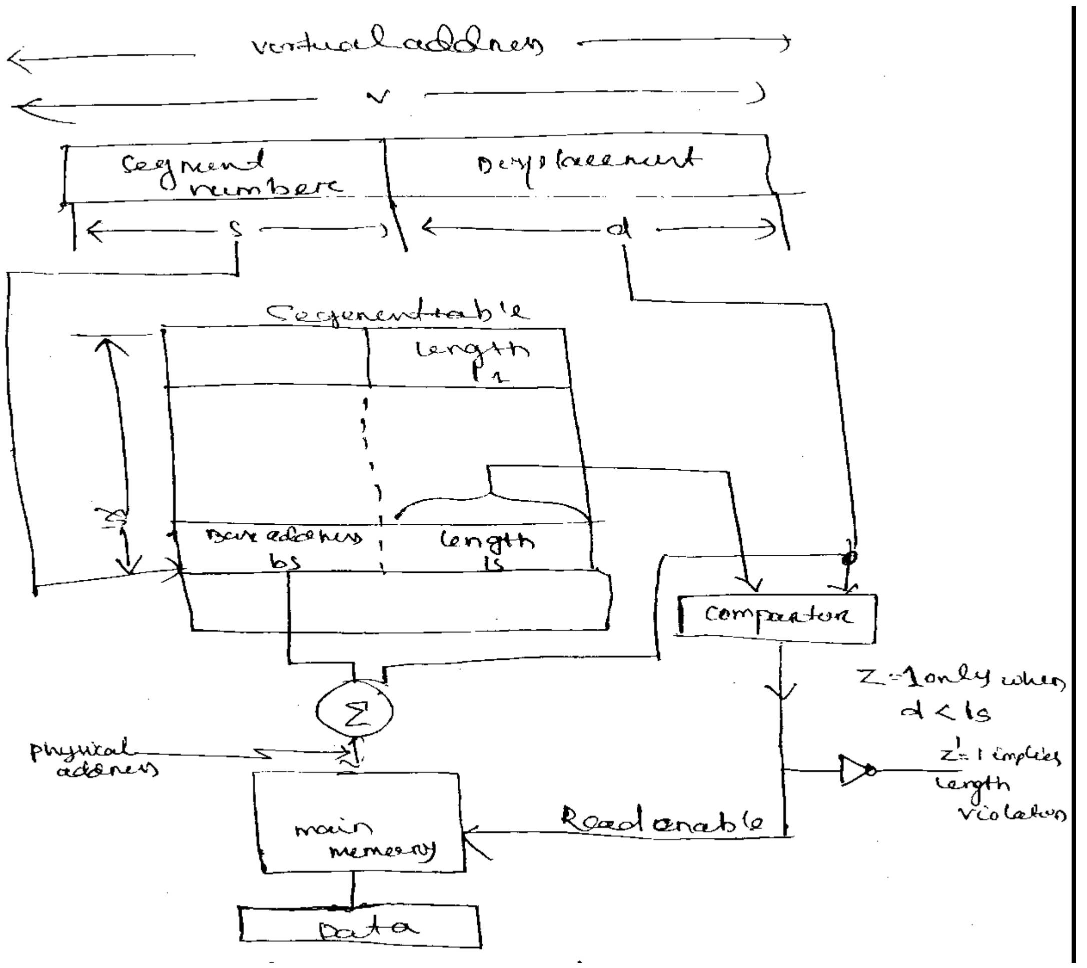
Frame		Physical Address (K)
Binary	Decimal	
06	6	0-2
01	1	2-4
10	2	4-6
11	3	6-8

Memory segmentation:-

The paging concept is viewed as one dimensional technique as virtual addresses generated by a program increase linearly from 0 to same maximum value. In a segmentation system, each logical entity like stack, array subrountine etc. have a separate virtual address space. The virtual address is called a segment and each segment can grow from zero to a maximum value. As each segment refers to a separate virtual address space, it can grow or shrink independently without affecting other segment.

Typically, a segment descriptor consists of the following information.

1. Segment base address b
2. Segment length 1
3. Segment presence bit
4. Protection bits.



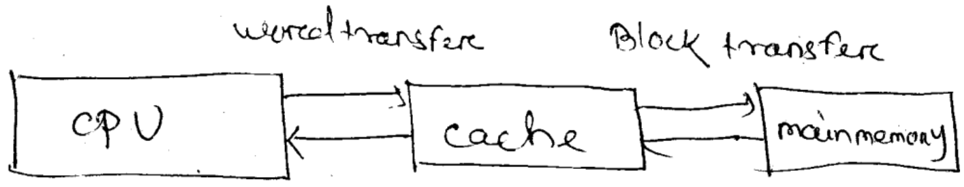
The following common protection protocols are used on a segmentation system:-

1. Read only
2. Execute only
3. Read and execute only
4. Unlimited access
5. No access.

Cache memory:-

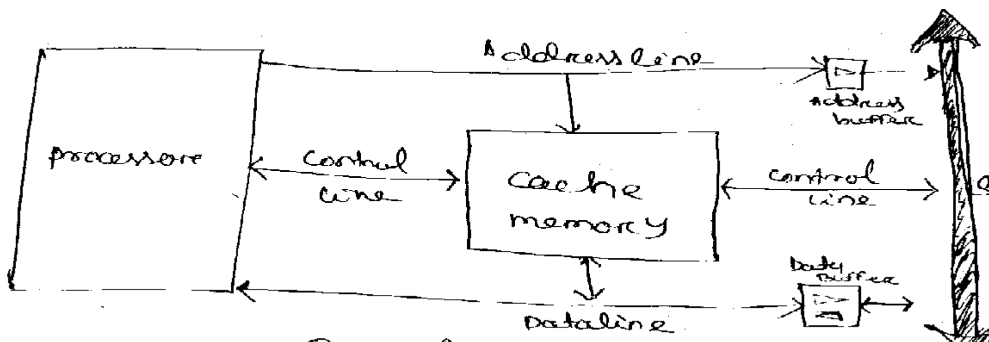
Cache memory provides a memory with the speed approaching that of the fastest available memories and a large memory size at the price of less expensive semiconductor memories.

A smaller and faster cache memory is connected to the large and slow main memory. A copy of portions of the main memory is contained in the cache. When the processor attempts to read a word word of memory, a check is made to find if the word is available in the cache. If available the word is delivered to the processor, other wise a block of main memory, consisting of some fixed number of words, is read in to the cache and then the required word is delivered to the processor.

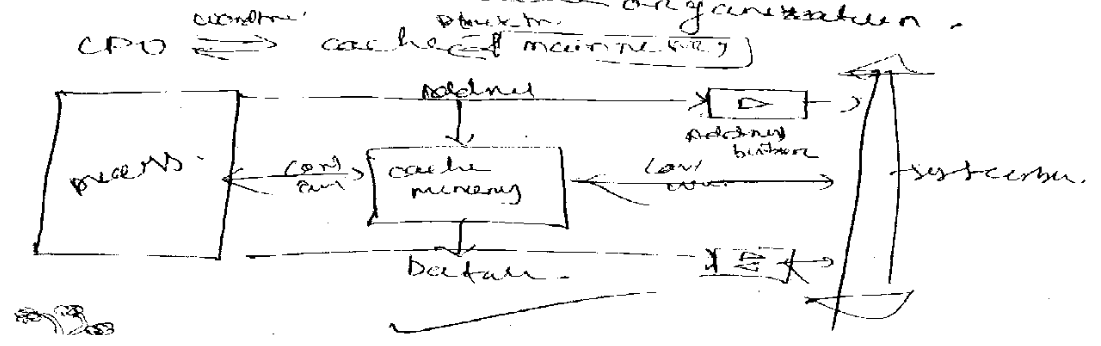


principle of cache memory ,

If the word is found in the cache, it is delivered to the processor. Otherwise, the block containing that word is loaded into the cache from main memory and then the word is delivered to the processor. On one side, the cache connects to the processor via data, control and address buffers are disabled and communication is only between processor and cache, with no system bus traffic. When a cache does not contain the required data, the desired address is loaded on to the system bus and the data are returned through the data buffer to both the cache and the processor. The desired word is first read into the cache and then transferred from cache to processor.



Typical cache organization .



Questions:

1. **What is deadlock?**
2. **What are deadlock prevention?**
3. **What is deadlock avoidance?**
4. **What are the condition of deadlock?**

CHAPTER-6

INPUT–OUTPUT SYSTEM

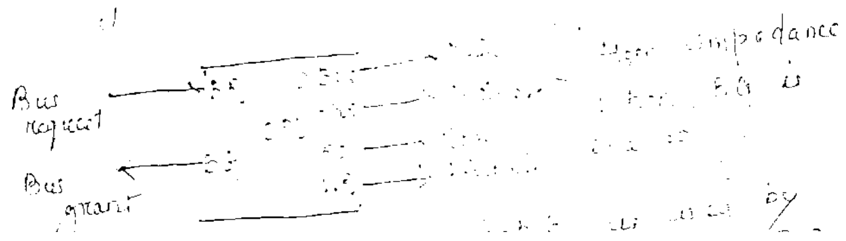
I/O channel Architecture:-

An increasing complexity and sophistication of individual components were brought in as computer system and evolved. This is more evident in the development of I/O function and the evolutionary steps can be summarized as below.

1. As seen in a simple microprocessor controlled device, the CPU directly control is a peripheral device.
2. A controller or an I/O module is added between the CPU and I/O. the CPU uses programmed I/O without interrupts and the CPU becomes some what removed from the specific details of the external device interface.
3. The same configuration in step 2 is improved to use interrupts. The CPU spends no time waiting for an I/O operation, thus improving efficiency.
4. The I/O module is given direct access to memory using DMA. Now, a block of data to or from the memory can be moved without involving the CPU, except at the beginning and end of the transfer.
5. The I/O module becomes of processor of its own right, with a specialized instruction set centre on I/O. the CPU allows the I/O processor to execute an I/O program on memory, as it fetches and executes these instructions without CPU interrupted only when the entire sequence has been completed. Thus type of I/O modules as referred to as an I/O channel.
6. A local memory is added to the I/O module., making it a computer in its own night with this architecture, a large number of I/O devices can be added and controlled with minimal CPU involvement. A common use for such architect are called I/O processor is to control communication with interactive terminates and the I/O processor takes care of most of the taks5. As one travels along the evolutionary path, an increasing number of the I/O functions are performed without CPU involvement. The CPU is increasingly relived of I/O related tasks, thus improving performance and efficiency.

Direct memory access (DMA)

$\Rightarrow$ In DMA transfer, the CPU is edle and has no control of the memory busses. The transfer of data takes place directly between the fast storage device and memory controlled by DMA controller which mange the bus.



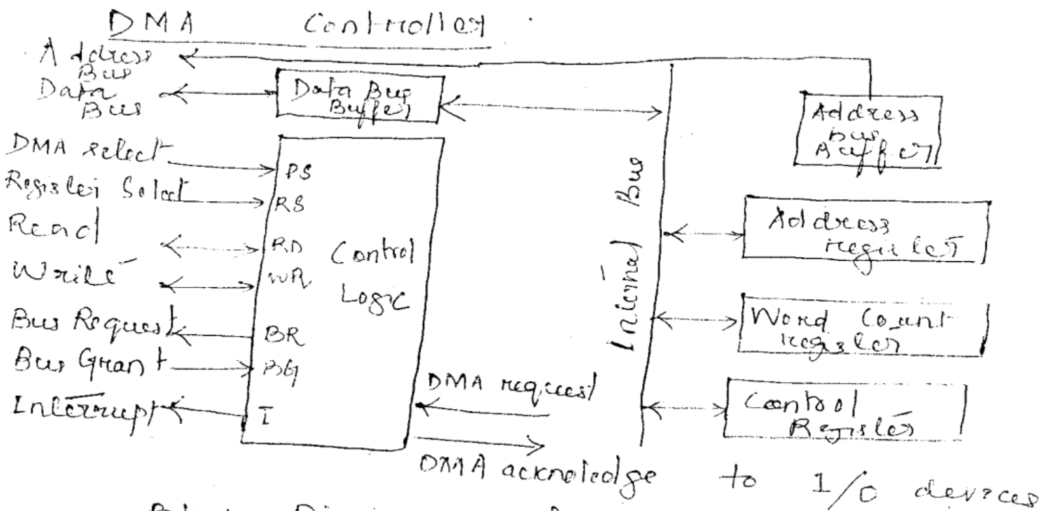
⇒ The Bus request input is used by the DMA controller to request the CPU to relinquish control of the buses.

⇒ The CPU activates the Bus Grant output to inform the external DMA that the buses are now rol under the control of CPU and can be used for memory transfer without the processer intervention.

⇒ Transfer of data can be done in many ways.

⇒ In burst transfer, a block sequence consisting of a number of memory words is transferred in the continuous hugest while the DMA controller is master of the memory.

⇒ In cycle stealing the DMA controlled transfer are data word at a time, after which it returns the control of busses to the CPU. The CPU uses the bus for one memory cycle.



Block Diagram of DMA C...

⇒ The unit communicates with the CPU via data bus and control lines.

⇒ The register in the DMA are selected by the (in through the address bus by enabling the DS (DMA select) and RS (register select) inputs.

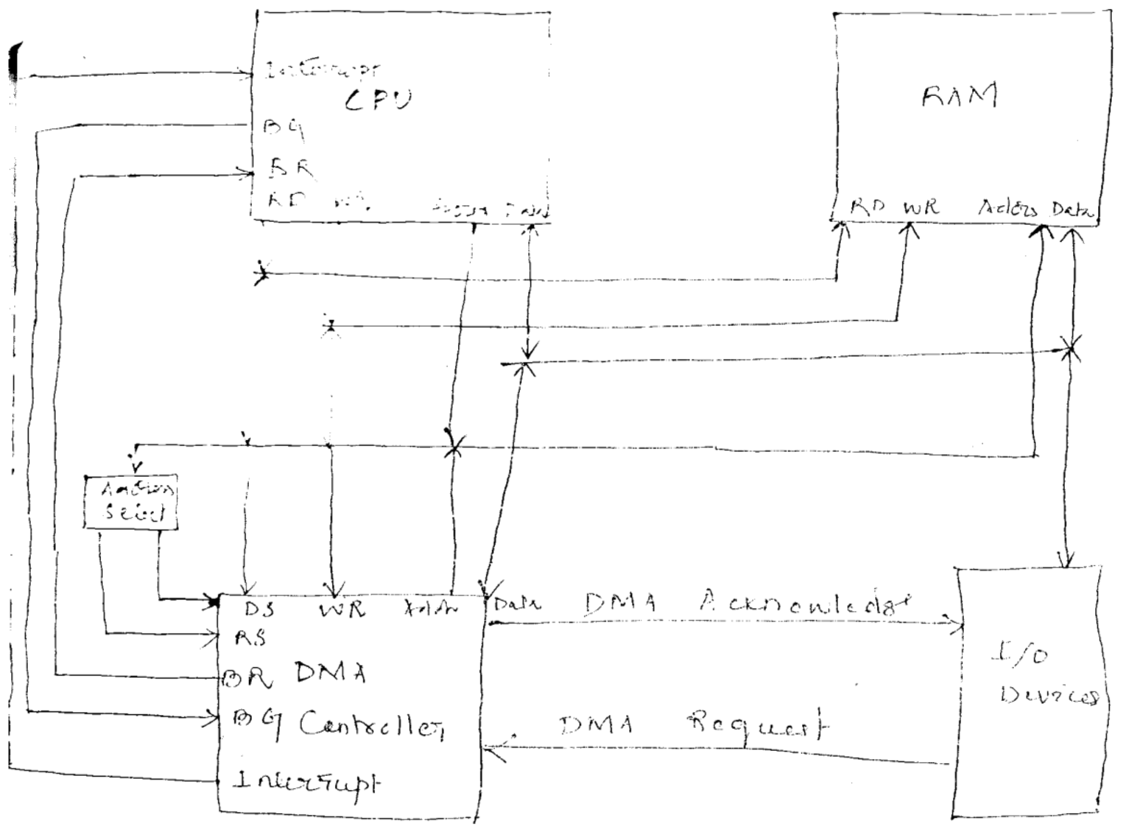
⇒ The RD and WR inputs are bialirection.

⇒ When the BG=0, the CPU can communicate with the DMA resisters to read framer write to the DMA register.

⇒ When the BG=1, the CPU has relinquished the buses and the DMA can communicate directly with the memory specifying the address in the address bus and activating the RD at WR control.

⇒ The address register control an address to specify the desired location in memory.
⇒ The work count register holds the number of words to be transferred.
⇒ The control register specifies the mode of transfer.

DMA transfer



DMA transfer in computer system

⇒ In DMA has its own address which activates the DS and RS lines.
⇒ When the peripheral device send a DMA request, the DMA controller activates the BR ins, in framing the CM to relinquish the buses.
⇒ The CM responds with its B4 low, informing the DMA that its buses are disabled.
⇒ The DMA plots the current value of its address register into its address bus. Initiates the FR or WR signal and sends a DMA acknowledge to the peripheral device.
⇒ When BG=0 the CPU communicates with the internal DMA register.
⇒ When BG=1, the RD and WR ace O/P line from the DMA controller to KAM.
⇒ When the peripheral device receives a DMA acknowledge, it puts a word in it data bus (for write) or receives a word from the data bus (for read).

⇒ For each word that is transferred the DMA increments the address register and decrements its word count register.

⇒ Till the word count is not zero the DMA checks the DMA request line.

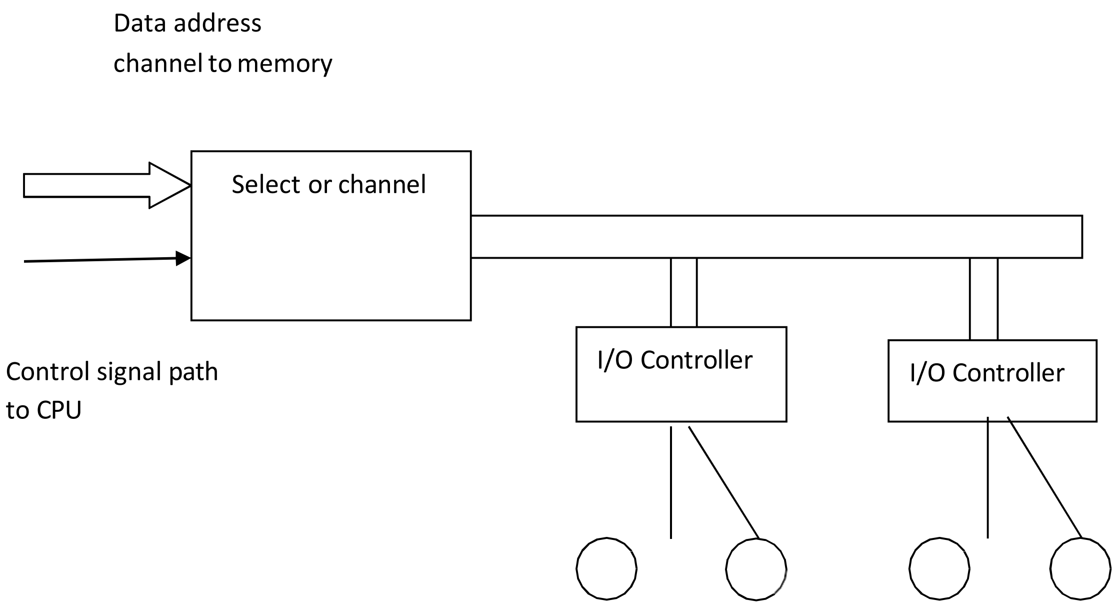
I/O channel architecture:-

The I/O channels can be considered as an extension of DMA module. Here another component used called I/O processor. The I/O channels has the ability to execute I/O instruction because it gets complete control over I/O operation. In such a computer system with I/o channels, the I/o instruction are not executed by the CPU but are stored in main memory to be executed by a special purpose I/o processor. Thus the CPU initiate an I/o transfer by instructing the I/o channels to execute a program in memory. The I/o processor execute an I/o program in memory without CPU intervention. The CPU is interrupted only when the entire sequence has been completed.

Two major type of I/o channels are commonly used namely.

a) Selector
b) Multiplexor

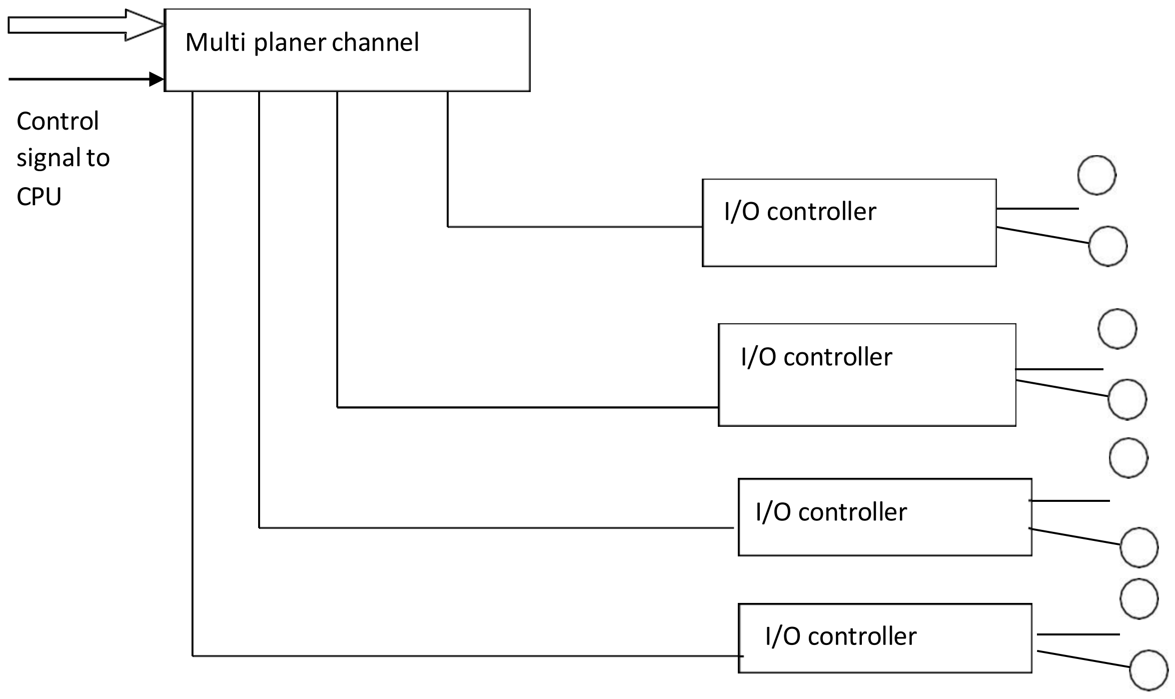
a) Selector:-



A selector channel controls multiple high speed I/O device and is dedicated to the transfer of data with one of those of any point of time.

b) Multiplexor:-

Data address Channel
to main Memory



A multiplexer channel can handle multiple I/O device at the same time.

<u>Questions:</u>

1. **Difference between compiler, interpreter.**
2. **Different phases of compiler.**
3. **Describe disk scheduling algorithm.**

CHAPTER-7

I/O INTERFACE & BUS ARCHITECTURE

BUS INTERCONNECTION:-

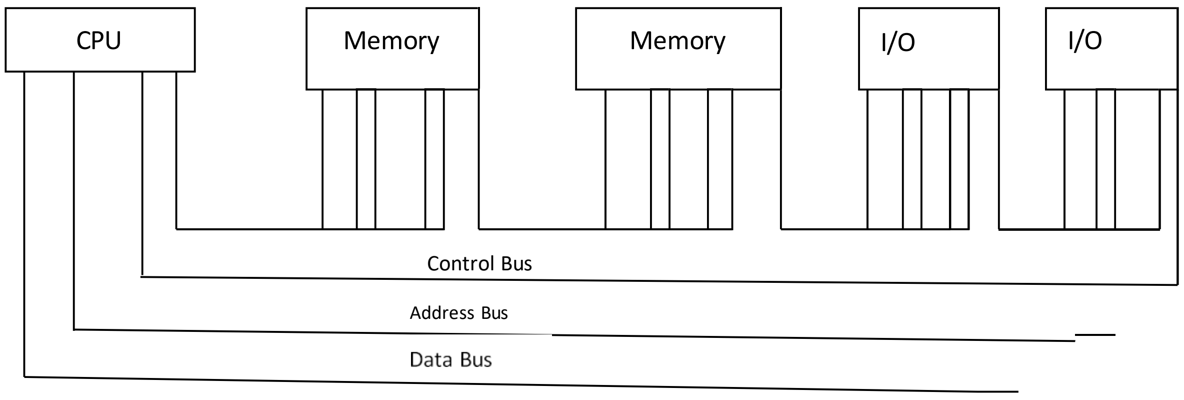
Bus consists of a number of communication lines and each lines can carry a signal representing a binary '0' or '1'. Bus is an inter connecting pathway between two or more device. It is a shared communication medium connected to multiple device.

Bus structure:-

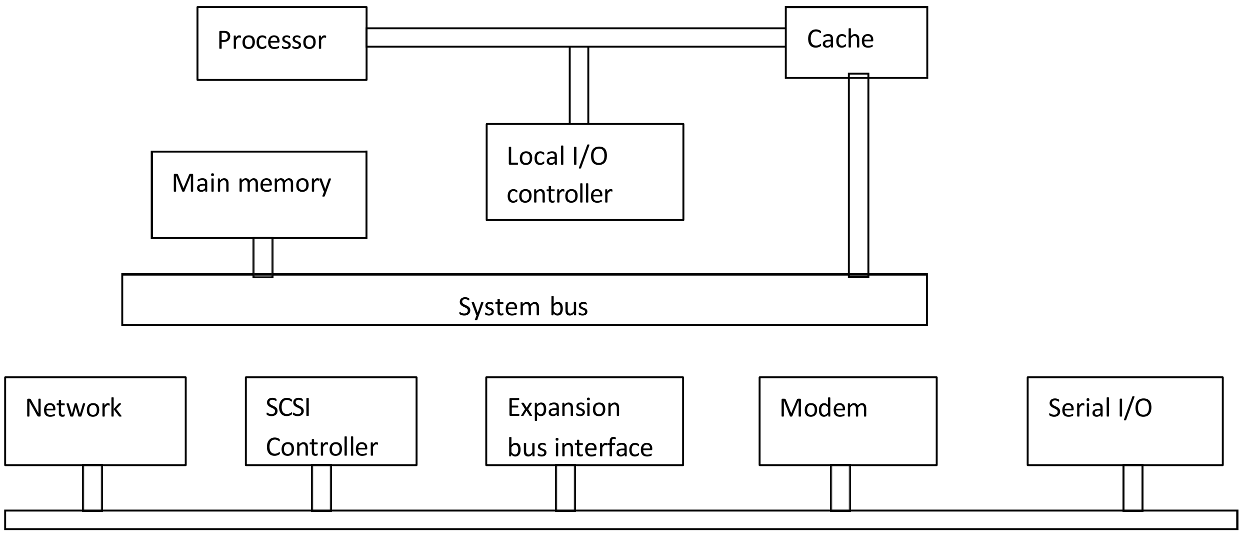
System bus consists of 50 to 100 communication line and each line assigned a function. The system bus can be classified into three basic groups i.e. data bus, address bus, control bus.

1. **Data Bus:-**
   The data bus provide a path for data transfer between system module. The data bus will consist of 16,32 or 64 lines referred as width of data bus. The width of data bus is equal to the memory word length.

2. **Address bus:-**
   The address lines are used to locate the destination of data on the data bus. The width of the address bus determines. The maximum memory capacity of the system. If 16 bit address bus means memory capacity is 2^{16} location.



3. **Control bus:-**
   Control Bus is used to transmit command and timing information. The command indicates the operation to be performed and timing signal indicates the duration of validity of data and address information.



Basic parameter of Bus design:-

The following are the Basic parameters to be considered while designing a bus.

i) Type of Bus:-
   - o Dedicated
   - o Multiplexed

ii) Method of timing:-
   - ⊗ Asynchronies
   - ⊗ Synchronous

iii) Method of Arbitration:-
   - ⊗ Centralized
   - ⊗ Distributed

iv) Width of bus:-
   - ⊗ Data
   - ⊗ Address
   - ⊗ Control

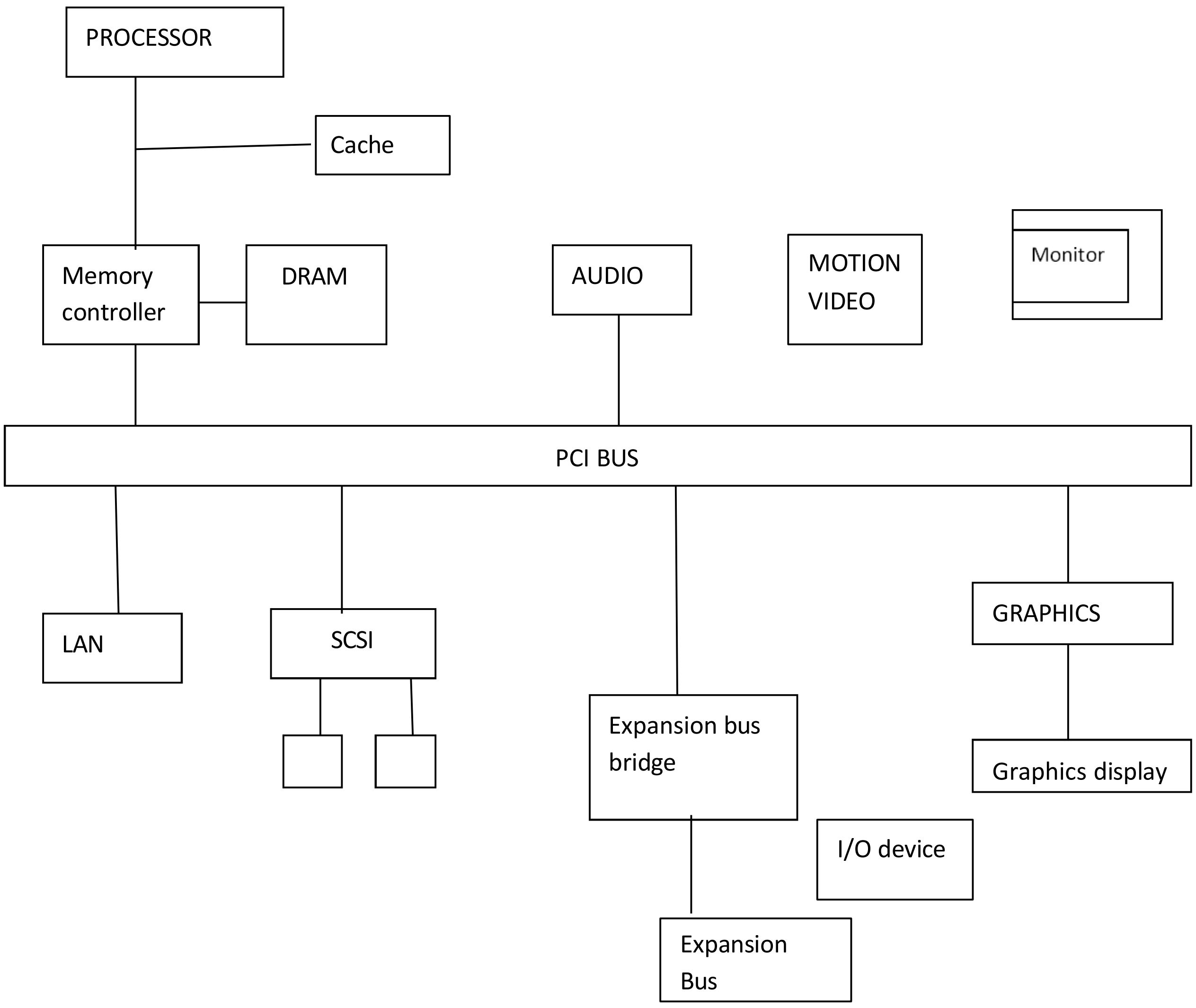
v) Type of data transfer:-
   - ⊗ Read
   - ⊗ Read-modify-write
   - ⊗ Read-after-write
   - ⊗ Block transfer.

Peripheral component inter connect BUS (PCI)

PCI bus in a well known, high band width and processor independent bus. It gives better performance for high speed I/O like graphics adapter, network interface and DMA controllers for disk.

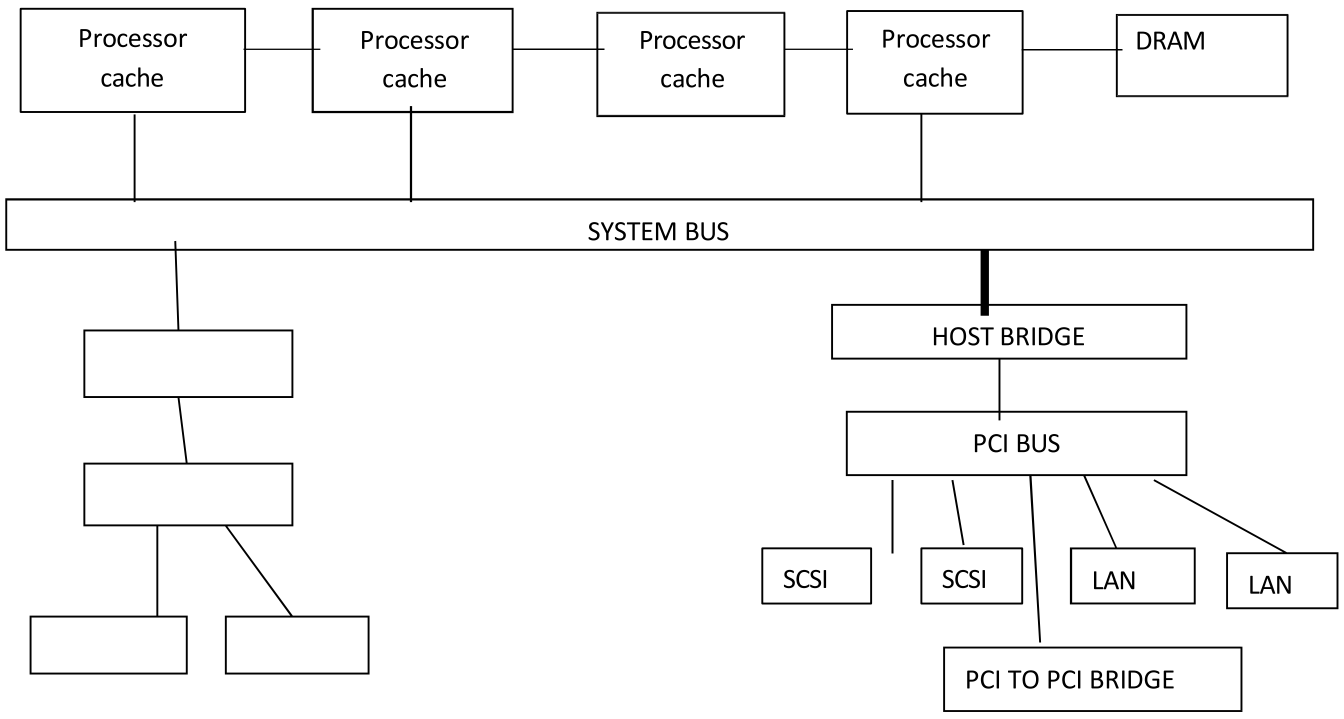
PCI bus use upto 64 data lines at 66 MHZ speed. The data transfer rate is 528 MBPS to 4.224 GBPS also economical for most of the modern I/O system requirements. It requires only a few chips to implement and support other buses.

⇒ It can support both single and multiprocessor system and provide a general purpose set of function.



PCI bus makes use of synchronous timing and a centralized arbitration scheme.

In a multiprocessor system as shown below more than one processor with multiple PCI configuration is connected by bridges to the processors system bus. The system bus supports only processor/ can be main memory and PLI bridge.



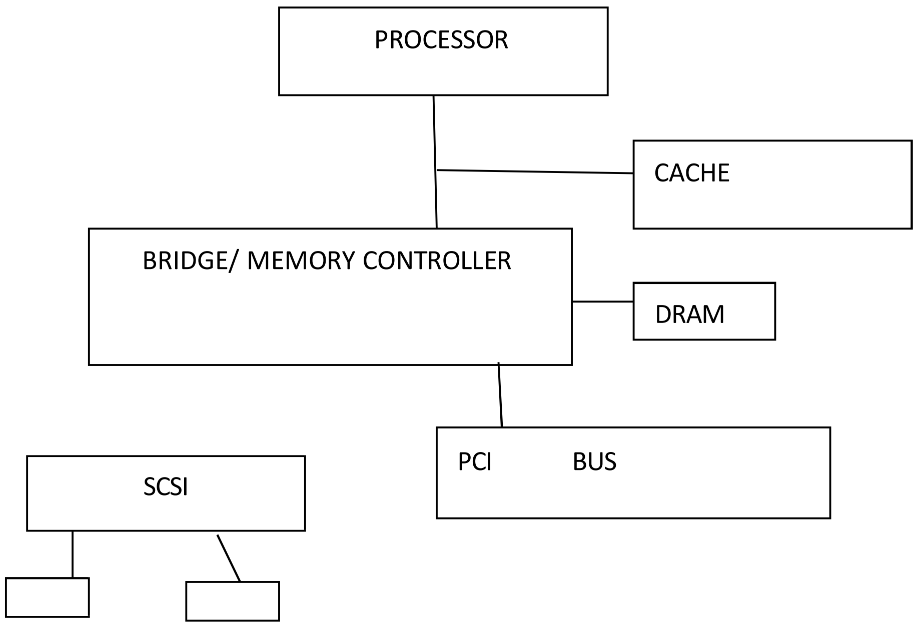
Small computer system interface (SCSI):-

Small computer system interface (SCSI) refers to a standard bus defined by American National Standard Institute (ANSI). A SCSI bus may have 8 data line which is called narrow bus and transfers data one byte at a time. A wide SCSI bus 16 data lines and transfers data 2 bytes at a time.

The SCSI bus is connected to the processor bus through an SCSI controller which uses DMA to transfer data from main memory to the device or vice-versa.

The operation of SCSI bus is explained using a disk drive. Communication with a disk drive differs from main memory. Data are stored on disk sectors and not in contiguous sector. Hence for a read or write operation, need to access several disk sector which may not be continuous. Due to the constraints of the mechanical motion of the disk. There is a delay of several millisecond before the 1st sector from which data are to be transferred is reached. A block of data is transferred at high speed and after some delay another set of data my follow.

$\Rightarrow$ The SCSI connector may have 50,68 or 80 pins and maximum transfer rate 5 to 16 Mbps.



<u>Universal serial Bus (USB):-</u>

A modern computer system uses variety of devices such as key board micro phone, camera, speaker and display device.

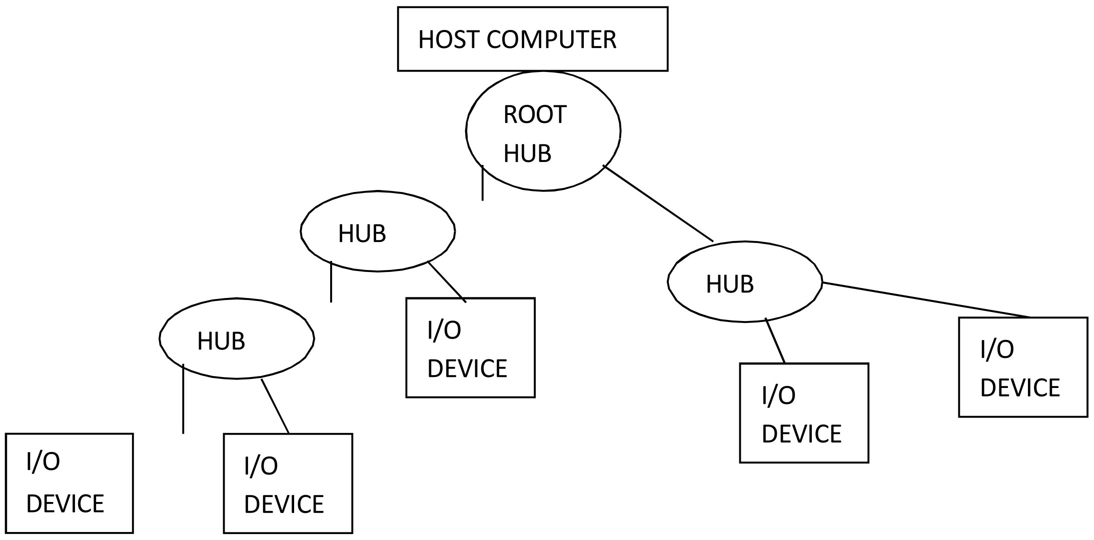
⇒ The simple, low-cost mechanism to connect these device to computer is the universal Serial Bus(USB)

⇒ USB has been designed to meet several objectives mainly are

- ➢ Plug & play mode of operation
- ➢ To connect variety of device including telephone and internet.
- ➢ Simply and low cost and easy to use
- ➢ To add many device to computer at any time.

⇒ The USB supports two speed operations one called low speed at 1.5 Mbps and the other full speed at 12 mbps. A recent version introduced a third speed operation, called high speed at 480 Mbps.

USB architecture:-



A large no of devices can be added or removed at any time, the USB uses the tree structure as shown in the fig.

⇒ Each hub has a no. of ports where device may be connected of which one device can be HUB. A message sent by the host computer is board cost to all the I/O devices and only the addressed device respond to that message.

⇒ The message sent from an I/O devices sent only Up stream towards the root of the tree and is not seen by other devices. Hence the USB enable the host to communicate with the I/O device but it doesn’t enable these device to communicate with each other.

Questions:

1. **Describe a Bus structure.**
2. **What is USB and explain?**

CHAPTER-8

PARALLEL PROCESSING

PARALLEL PROCESSING

Parallel processing is a technique used to provide simultaneous data processing task for the purpose of increasing the processing speed of a computer system. Instead of processing each instruction sequentially as in conventional computer, a parallel processing system is able to perform concurrent data processing to achieve faster execution time. The system may have two or more ALU's and be able to execute two or more instructions at the same time. Further more, the system may have two or more processors operating concurrently. The purpose of parallel processing is to speed up the computer processing capability and increase its throughput.

Parallel processing mechanism:-

A number of parallel processing mechanism have been developed in uniprocessor computers. We identify them in the following six categories.

- Multiplicity of functional units.
- Parallelism and pipeline within the CPU
- Overlapped CPU and I/O operations
- Use of hierarchical memory system
- Balancing of subsystem bandwidths.
- Multiprogramming and time sharing.

Register transfer language:-

A micro operation is an elementary operation performed on the information store in one or more register. The result of the operation may replace the previous binary information of a register or may be transferred to another register.

e.g. shift, count, clear, load.

The internal h/w/ organization of a digital computer is best defined by specifying.

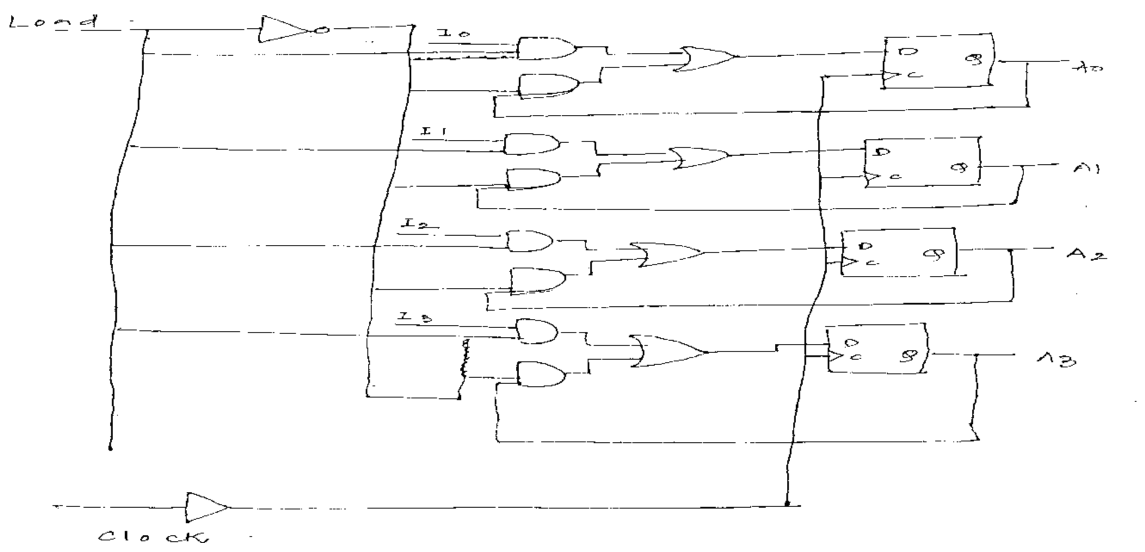
1) The set of register it contains and their function.
2) The sequence of micro operation performed in the binary information stored in the register.
3) The control that initiated the sequence of micro operation.

The symbolic notation used to describe the micro operation transfer among registers is called a register transfer language. The term register transfer implies the availability of h/w logic ckts that can perform a stated micro operation and transfer the result of the operation to the same or another register. The term language is borrowed from programming language.

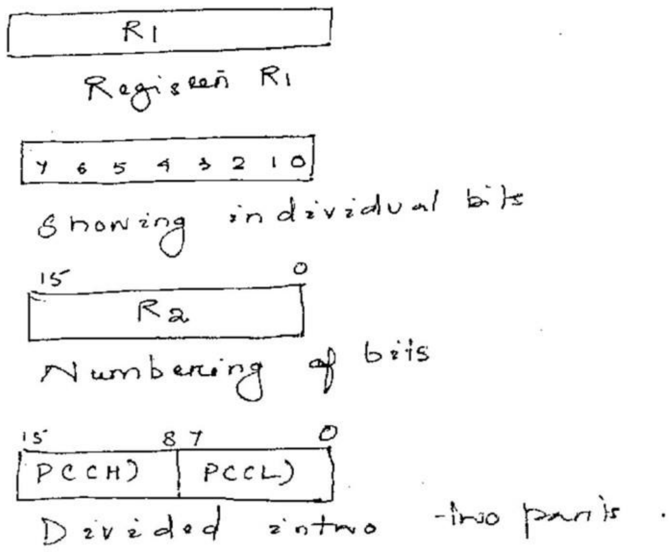
Register transfer:-

A resister consists of a group of flip flops that hold binary information and the gates that control when and how new information is transferred to the flip flop.

A 4 bit register with parallel load



Representation of resisters an Block Diagram



Resister transfer

A register transfer is the transfer of data from one register to another or to the same.

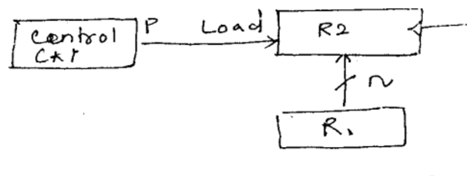
A register transfer statement .

$P : \overset{\smile}{R_2} \leftarrow R_1$.

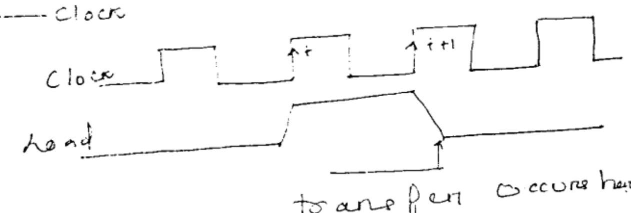
control function (A Boolean variable that is equal to 1 or 0) The transfer takes place if P = 1)

Data transfer takes place from R_1 (source register to R_2 (destination register)

Transfer from R_1 to R_2 when P = 1



Block Diagram



Timing Diagram

Flynn's Classification:-

Flynn's classic taxonomy is based on the number of control units and the number of processes available in a typical computer. Accordingly, the basic computers can be classified as follows:-

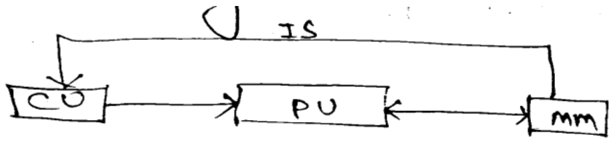
Single instruction:- Single data Stream (SISD) – This architecture has a single control unit producing a single stream of instructions. These instructions are executed sequentially but may be overlapped in their execution pipeline stages. An SISD computer may have more than one functional unit and all the functional units are under the supervision of one control unit.

Single instruction – Multiple Data Stream (SIMD):- in this implementation, a single machine instruction controls the simultaneously execution of a number of processing element and each processing element has its won associated data memory. This facilities each instruction to be executed on different sets of data by different processors.

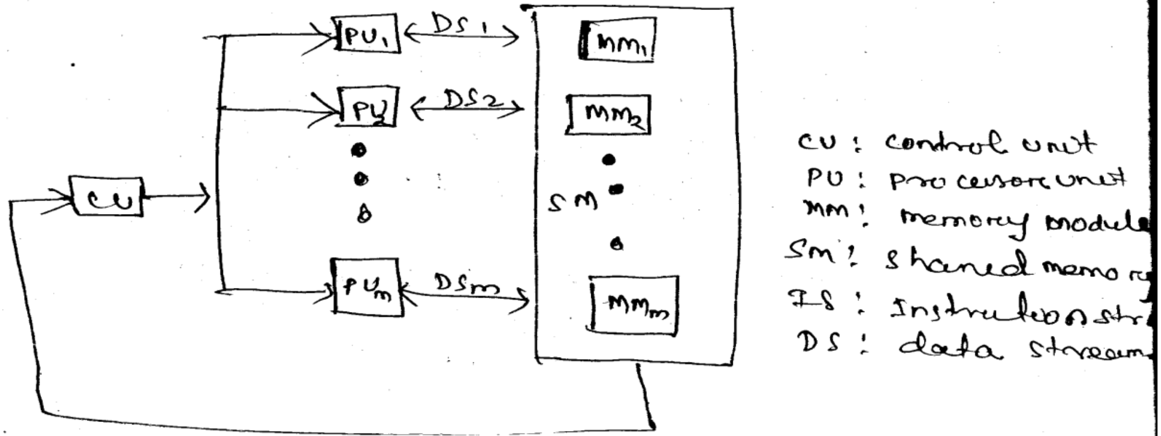
Multiple instruction:- Single data stream (MISD):- A single processor is available which executes a single data stream of data using multiple instruction. This structure has never been implemented.

Multiple instruction:- Multiple data stream (MIMD):- A set of multiple processor is available which executing a different stream of data using different set of instruction sequences. Symmetric multiprocessor (SMP) and Non-Uniform Memory Access System (NUMA) are such kind of implementations.

A comprehensive implementation of all the above types of organizations are shown in figure:-



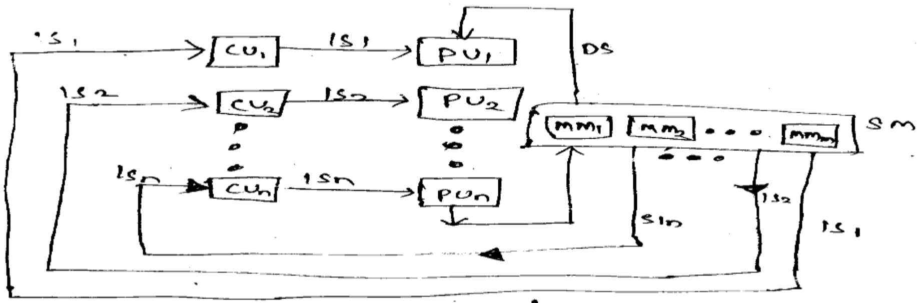
(a) SISD computer



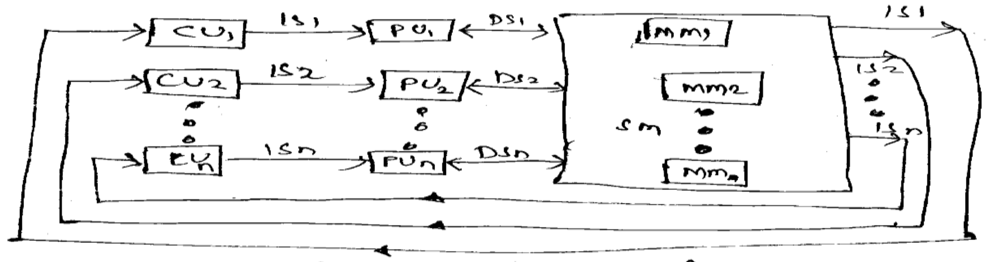
(b) SIMD Computer.

Flynn's classification of various computer

A complete taxonomy of parallel processor arrangement are given in figure



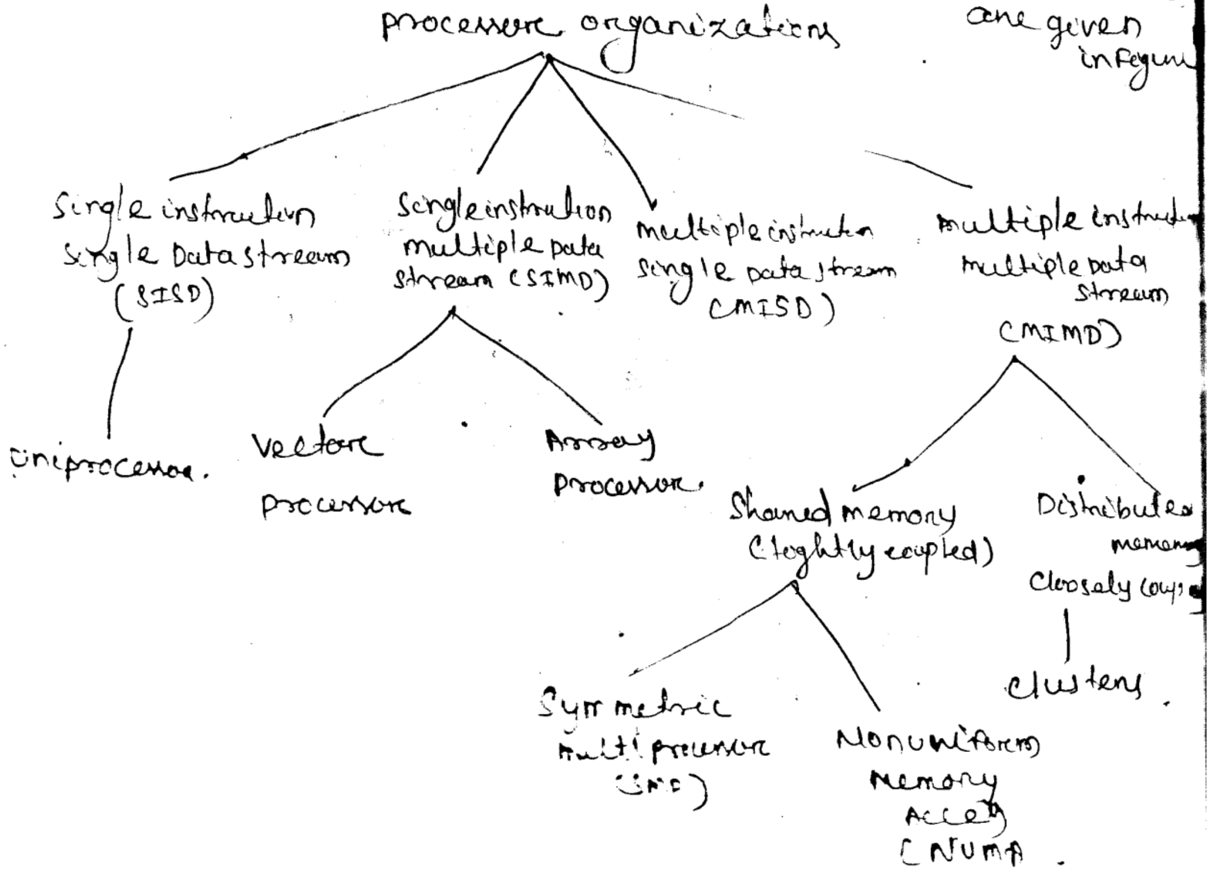
(c) MISD computer.



(d) MIMD computer.

Flynn's Classification of various computer

A complete taxonomy of parallel processor organization architecture are given in figure


processor organization
Single instruction Single Data stream (SISD)
Single instruction multiple Data stream (SIMD)
multiple instruction single Data stream (MISD)
multiple instruction multiple Data stream (MIMD)
Uniprocessor.
Vector processor
Array processor.
Shared memory (tightly coupled)
Distributed memory Closely coupled
clusters.
Symmetric multiprocessor (SMP)
Nonuniform memory Access (NUMA

In the parallel architecture taxonomy, the MIMD model has clearly emerged as the chore architecture on recent years. It offers the flexibility of operating as a single user machine providing high performance for one application and as a multi programmed machine running many task simultaneously. It can be built on the off the shelf microprocessor available in the market with a better cost/performance advantage.

The existing MIMD machines fall into two classes, viz centralized (Sym-metric) shared memory arithmetic tunes. (tightly coupled) and distributed shared memory architecture (loosely coupled).

<u>Questions:</u>

1. **What is parallel processing?**
2. **Explain Flynn's classification.**