Computer Architecture & Assembly Language

(CAAL)

BCA - 2nd Year

3rd semester

Computer Architecture and Assembly language

Unit ⇒ 1

PAGE NO. / /
DATE / /

★ **Introduction :-** Computer Architecture is a set of rules and method that describe the functionally, organisation and implementation of computer System. Some definitions of Architecture define it and describing the capabilities of programming model of a computer.

✓ Computer architecture involves instructions at architecture design, micro architecture design, logical design and implementation.

M.Imp There are three categories of computer arch. :-

1- **System design :-** This includs all hardware components in the system, including data processor aside from the CPU such as graphics designing units and direct access memory.

2- **Instruction set Architecture (ISA) :-** This is the embedded programming language of the central processing unit. ✓ It define the CPU's functions and capabilities

basid on what programming it can perform or process. This include the word size, processor register type, memory addressing mode, data formate and the instruction set that programers use.

3- Micro Architecture :-
It known as Computer organisation. This type of architecture define the data path, data processing and storage elements, as well as how they should be implemented in the ISA.

* Basic Concepts of Component of Computer Architecture :-
The model of a computer can be described by four basic units in High level abstraction. These basic units are :-
1) Central Processing unit (CPU)
2) Input device
3) Output device.
4) Memory unit

1- Central Processing unit :-
Central Processing unit consist of two basic block.

i) The program control unit has a set of registers and control circuits to generate control signals.

ii) The execution unit or data processing unit contain a set of registers for storing data and an arithemtic and logic unit. ALU. For execution of arithematic and logical operations.

- In addition, CPU may have some additional registers for using temporary storage the data.

2- Input device :-
With the help of Input unit data from outside can be supplied to the computer. program or data is read by the Input unit.
ex:- keyboard, mouse,

3- Output device :-
With the help of output unit computer results can be provided to the user or it can be stored in storage device permanently for the future use.
ex:- Printer, Monitor, plotter, Scanner.

4- Memory unit :- Memory unit is use to store the data and program. This memory is termed as primary memory or main memory.

i) Primary memory :-
Primary memory is a temporary memory which store the data only when the work on it. This memory data automatically storage destroyed. When your system will be shut-down.
ex:- RAM, ROM, EROM, PROM, EPROM, EEROM.

(ii) Secondary memory:-
Secondary memory is use to permanent Storage of data and program.
ex:- Harddisk Hardisk, floppy disk, CDROM.

★ Program execution and Instruction execution

★ Instruction:- In computer architecture, an instruction is a single operation of a processor define by the processor instruction set. The size or length of a instruction is 4 bits.

★ Instruction code:- An Instruction code is a group of bits that instruct the computer to perform a specific operation. An Instructions must also include one or more oprands. Which indigate the registers and/or memory addresses from which data is taken or to which data is deposite.

PAGE NO. // //
DATE // //

★ Instruction formate :-
An Instruction formate defines the layout of a bit of an Instruction. An Instruction formate must include an opcode and zero or more operands Implicitly and explicitly. Is explicit oprands is referenced using one of the addressing mode. That is available in the machine.

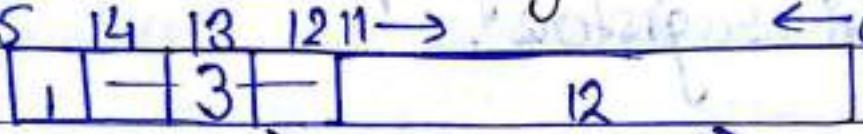
★ OPcode :- An opcode is a Single Instruction that can be executed by the CPU. In machine language it is a binary no. or hexadecimal no. which is loaded into Instructions registers.

★ Operand :- Operand are manuplated by opcode.

★ Execution :- It is a process to of the program into memory to decode. Instructions are perform functionality according the demand of program.

★ Timed and control In CU (central unit).

IR (Instruction Register) (16 bits)



0-Indirect → decode 3×8 → address provide.
1-direct. → Signals generate
→ Control on gates

- It provides timing and control signals to the micro processor to perform the various operation. It has three control signals. It controls all external and internal circuits. It operates with reference of clock signals. It synchronizes on the data transfer.

1- ALE (arithmetic latch enable) :-

It provide control signals to syncronizes the component of micro processor.

2- WR (write) :-

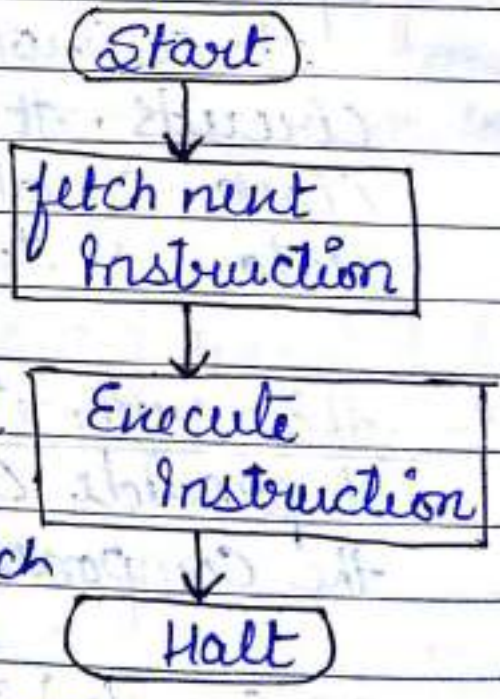
This is used for writing operation in instruction registers.

3- RD (Read) :-

It is used for reading operation in Instruction registers.

- There are status signals use in micro processor. S0, S1 and I/o/m (memory). It changes its status. According the providing in put to the Instruction registers.

4. Instruction Cycle :-



An Instruction cycle (also known as the fetch-decode-Execute cycle or fetch execute cycle) is the basic operational process of a computer. It is the process by which a computer retrieves a program instruction from its memory, determines what actions the instruction dedicate and carry out these actions.

PAGE NO. // //
DATE // //

• There are four step of Instruction cycle.

1- Fetch - Retrieve and Instruction from the memory.
2- Decode - Translate the retrieved Instruction into a series of computer command.
3- Execute - Execute the computer command.
4- Store - Send and write the results back In memory.

★ Program counter (PC) :-
An Intrementry counter that keeps track of the memory address of the Instruction that is to be executed next or hold the address of the Instruction to be executed next.

★. Memory address register (MAR) -
It hold the address of a block of memory for reading form or writing too.

★. Memory data register (MDR) -
A two way register that hold data

fetched from memory (and ready for the CPU to process) or data writing to be stored in memory (this is also known as memory buffer register (MBR).

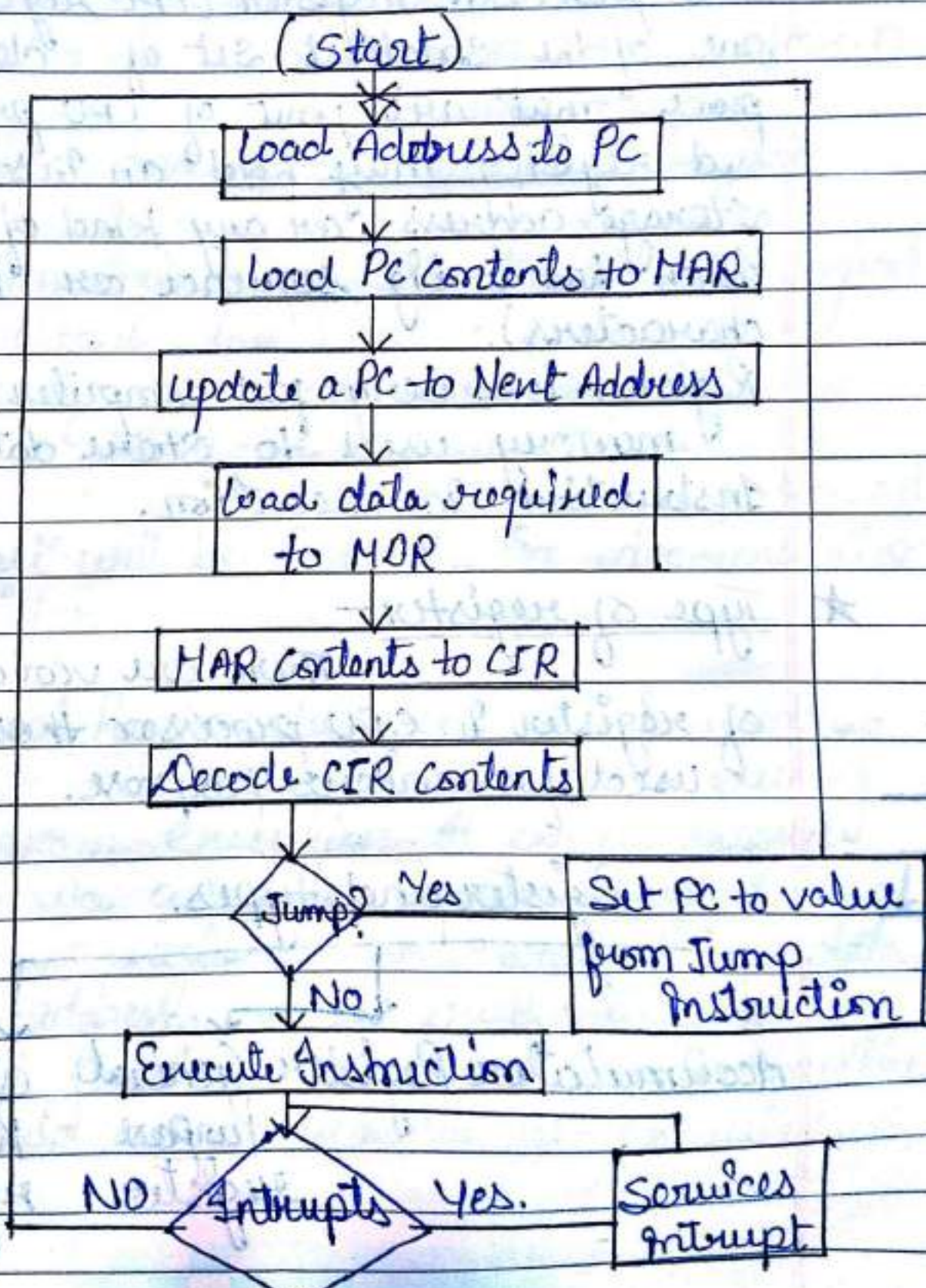
★ Instruction register (IR):-
A temporary holding ground for the instruction that has just we fetch from memory.

NOTE:- CIR (Current Instruction Register) -
An Instruction register which process or called currently being executed or decoded that register is called CIR.

★ Control unit (CU) -
Decode the program instruction in IR selecting machine resources such as a data source register and a particular arithmetic operation and coordinate activation of those resources.

PAGE NO. // //
DATE // //

★. Block diagram In Instruction Cycle-



PAGE NO. // //
DATE // //

★ Registers and Its types –

A processor register (CPU register) is one of the smallest set of data holding place that are part of CPU processor. A register may hold an instruction, storage address or any kind of data (such has a big sequence or individual characters).

Register is a very fast computer primary memory used to store data/ instruction in execution.

★. Type of register –

There are various type of register in CPU processor those are used for various purpose.

Register and types.

- Accumulator Register
- General Purpose register
- Special Purpose register

PAGE NO. // //
DATE // //

1- Accumulator Register :- In computer central processing unit (CPU) an accumulator register is a register which perform connection b/w general purpose register and special purpose register. Accumulator register perform intermediate b/w arithmetic and logical operations. This register is used to storing the results those produce by CPU. When the CPU with generates some results after the processing then all result will be stored in Accumulator register.

2- General Purpose register :- This is used to store data intermediate results during program execution. It can be excessed via assembly programming. General purpose register can store both data and address. It so truth value often used to determined the some instructions should or should not be executed.

PAGE NO. / /
DATE / /

★ Special purpose Register –
Users do not access these registers. These registers reserved for computer System. There are many types of special purpose register according to their purpose.

1- MAR- Memory Address register
2- PC- Program counter.
3- MBR- Memory buffer register.
4- MDR- Memory data register
5- IR – Instruction register.
6- CIR – Current Instruction register.
7- Index register.

• MBR:-
Memory buffer register store Instruction and data received from the memory and sent from the memory. This register hold the content of data or Instructions read from or write in memory. It means that register is used to stored data / Instructions. Coming from the memory or going to the memory.

- **Index Register:-** An index register in a CPU is a processor register used for modifying operand addresses during the run of program an execute the program.
It hardware element which hold a no. that can be added to (In case of subtract form). the address portion of a computer instruction to form an effective address. Index register also known as base register.

Micro-operation:- Micro-operation In computer CPU micro-operation (also known as micro-ops). are the functional or atomic operation of a processor. Micro-ops are low level instructions. Micro-ops perform basic ops. on data (which is stored in registers) and also transfer the data between registers and buses.

PAGE NO. // //
DATE // //

Types of micro-operation-

1. Arithmetic micro-operation
2. Logic micro-operation
3. Shift micro-operation
4. Register transfer.

1. Arithmetic micro-operation-

This micro operation performs the basic arithmetic operations like - addition, subtraction, Increment, decrement.
Example-

$R_3 \leftarrow R_1 + R_2$ Content of R_1 plus R_2 transfered to R_3
$R_3 \leftarrow R_1 - R_2$ Contents of R_1 minus R_2 transfered to R_3
$R_2 \leftarrow R_2'$ Complement the contents of R_2
$R_2 \leftarrow R_2' + 1$ 2's complement the contents of R_2 (Negate)
$R_1 \leftarrow R_1 + 1$ Increment
$R_1 \leftarrow R_1 - 1$ Decrement.

2. Logic micro-operation-

Logic micro-operation are bitwise operation, they work on the individual bits of data.

PAGE NO. / /
DATE / /

Logical micro operation perform these logic operations -

1. AND (∧)
2. OR (∨)
3. XOR (⊕)
4. Complement / Not

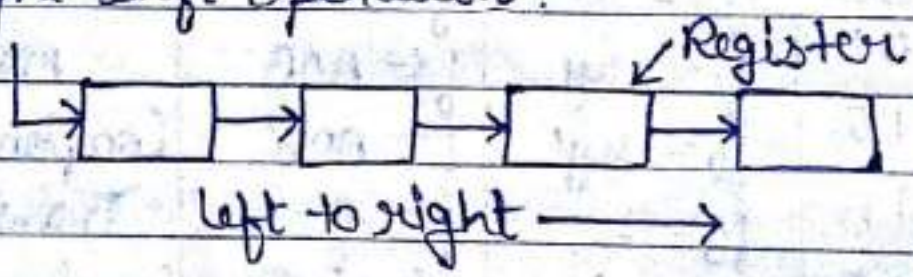
Example -

x 0011 y 0101	Boolean function	Micro-operation	Name
0000	$f_0 = 0$	$f \leftarrow 0$	Clear
0001	$f_1 = xy$	$f \leftarrow A \wedge B$	AND
0010	$f_2 = xy'$	$f \leftarrow A \wedge B'$	Complement AND (B)
0011	$f_3 = x$	$f \leftarrow A$	Transfer A
0100	$f_4 = x'y$	$f \leftarrow A' \wedge B$	Complement AND (A')
0101	$f_5 = y$	$f \leftarrow B$	Transfer B
0110	$f_6 = x \oplus y$	$f \leftarrow A \oplus B$	Exclusive OR
0111	$f_7 = x + y$	$f \leftarrow A \vee B$	OR
1000	$f_8 = (x \oplus y)'$	$f \leftarrow (A \oplus B)'$	Exclusive NOR
1001	$f_9 = (x + y)'$	$f \leftarrow (A \vee B)'$	NOR
1010	$f_{10} = y'$	$f \leftarrow B'$	Complement B
1011	$f_{11} = x + y'$	$f \leftarrow A + B'$	Complement OR (B')
1100	$f_{12} = x'$	$f \leftarrow A'$	Complement A
1101	$f_{13} = x' + y$	$f \leftarrow A' \vee B$	Complement OR (A')
1110	$f_{14} = (xy)'$	$f \leftarrow (A \wedge B)'$	NAND
1111	$f_{15} = 1$	$f \leftarrow$ All 1's	Start on 1's

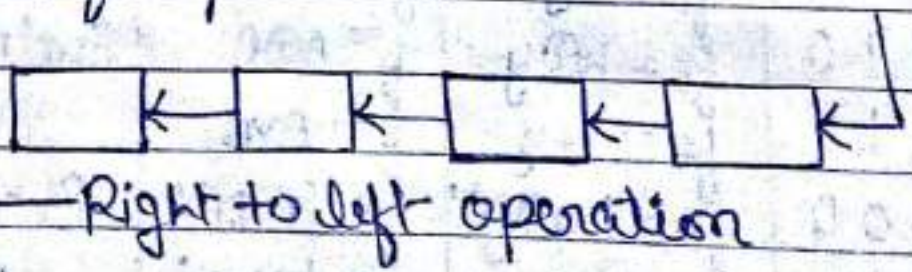
PAGE NO. / /
DATE / /

3. Shift micro-operation -
Shift micro operation are the operation in which the content of the register can be shifted left to right or right to left. Shift micro-operation are used for serial transfer of data. They can also used in arithmetic, logic and other data processing operations.

(i) A right shift operation.



(ii) A left shift operation



4. Register transfer micro-operation -
Information transferred from one register to another register (or one resource to another resource)

is designed in symbolic form by means of replacement operator.

★ Interrupts :-

What is it?

It is a mechanism by which modules like I/O or memory may interrupt the normal processing of CPU.

Why required?

To improve the processing efficiency of CPU.

How?

Most external devices are slower than C.P.U. If no interrupts → the CPU would waste a lot of time waiting for these external devices to match up with C.P.U's speed.

1- Wastage of C.P.U time.
2- Wastage of instruction cycles.
3- Continuous checking to find the task completion.

Without Interrupts –

1)- CPU instructs printer to print.
2)- While printer does its task, CPU waits for task completion.
3)- User program stopped.
4)- Repeated checking by CPU.
5)- When task done, CPU proceeds.

With Interrupt –

1- CPU instruction printer to print.
2- While printer does its task, CPU engaged in executing other instructions.
3- User program proceeds con currently with printing.
4- When task done, printer tells CPU.

Types of Interrupts

1- Program Interrupt – It occur when some instruction within the program creats a condition that leads to an interrupt. Ex: Divide by 0, arithmetic overflow, attempt to access an illegal memory location.

2- Timer Interrupt-
- Generated by the timer present within the processor.
- OS set the timer to perform certain operation on regular basis.

3- I/O Interrupt-
- Generated by I/O devices.
- Signal successful task completion or error conditions.

4- Hardware Interrupt -
Cause: failure related to hardware.
eg:- Memory Parity Error.

★. Interrupt Request and Interrupt Handler:-
CPU executing some task
⇓
User uses the keyboard to issue a high-priority command.
⇓
This issues an interrupt request to the CPU
⇓

PAGE NO. // //
DATE // //

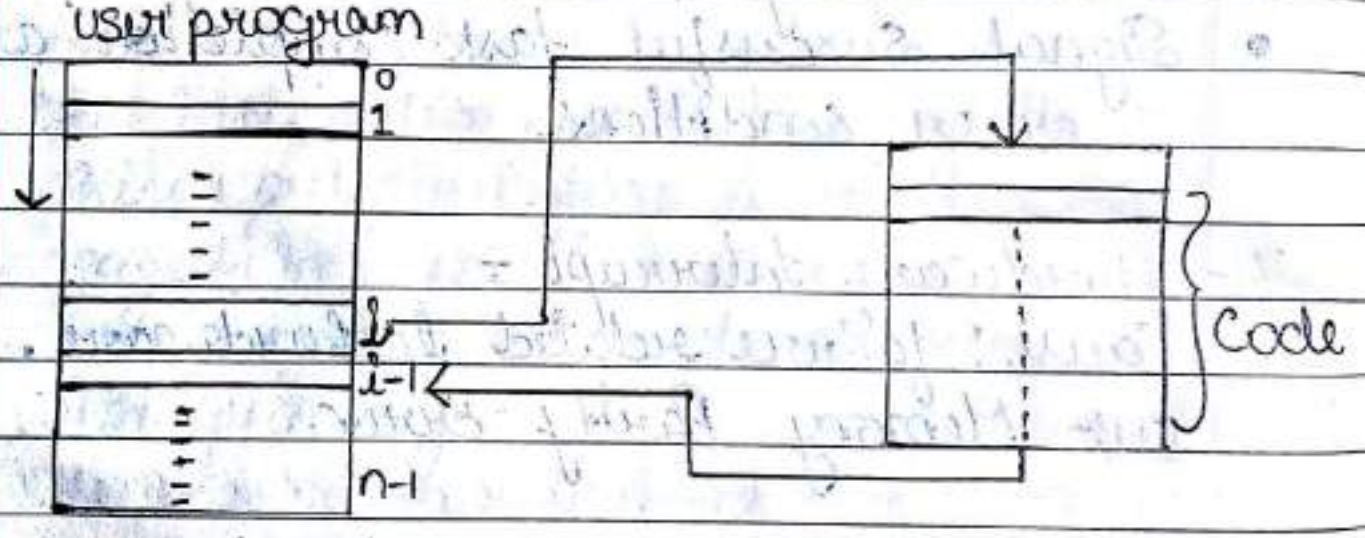
CPU suspends the current execution of the task.

⇓

CPU executes the (code) written to handle Implement this command. ← Interrupt handler

⇓

CPU resumes its previous execution.



Control Transfers in case of Interrupt

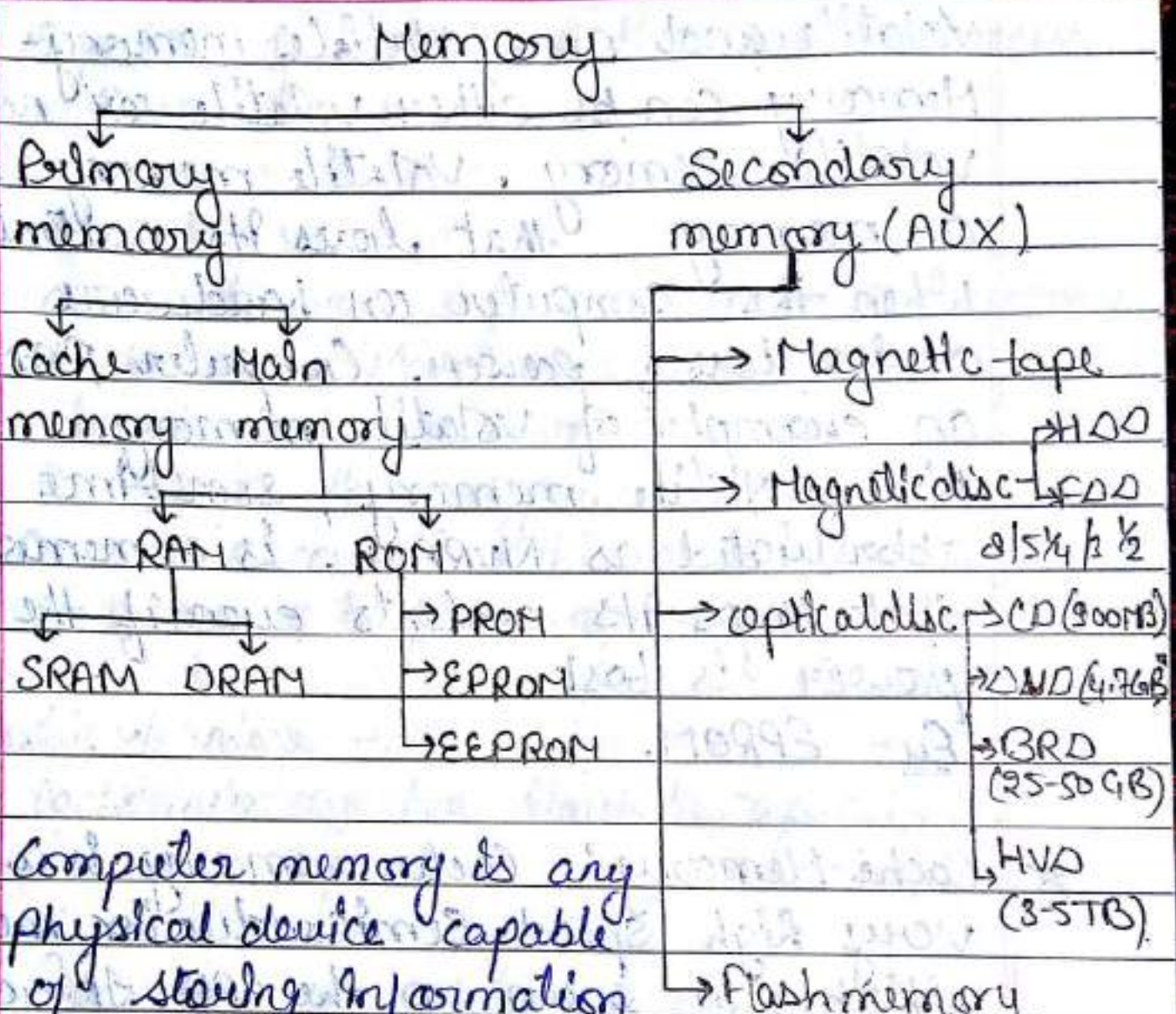
Advantage - Efficiency of CPU improves.

Disadvantage - Overhead required to service the Interrupt request.

- Saving Current Content
- Decoding of the code to handle Interrupt
- Switch back

PAGE NO. // //
DATE // //

★ Memory



Computer memory is any physical device capable of storing information temporarily or permanently. For eg RAM is a volatile memory that stores information on an integrated circuit used by the operating system, software and hardware.

PAGE NO. // //
DATE // //

Volatile and non-volatile memory- Memory can be either volatile or non-volatile memory. Volatile memory is a memory that loses its contents when the computer or hardware device loses power. Computer RAM is an example of volatile memory. Non-volatile memory, sometime abbreviated as NVRAM, is a memory that keeps its contents even if the power is lost.
Eg- EPROM.

*. Cache Memory :- Cache memory is a very high speed semiconductor memory which can speep up the CPU. It acts as a buffer between the CPU and the main memory. It is used to hold those parts of data and program which are most frequently used by the CPU. The parts of data and programs are transferred from the disk to Cache memory by the operating.

System, from where the CPU can access them.

Advantages -

1- Cache memory is faster than main memory.
2- It consumes less access time as compared to main memory.
3- It stores the program that ~~period~~ can be executed within a short time period.
4- It stores data for temporary use.

Disadvantages -

1- Cache memory has limited capacity. It is very expensive.

☆ Primary memory (Main memory):-
Primary memory holds only those data and instructions on which the computer currently working. It has a limited capacity and data is lost when power is switched off.
These are semiconductor memories. It is known as the main memory.

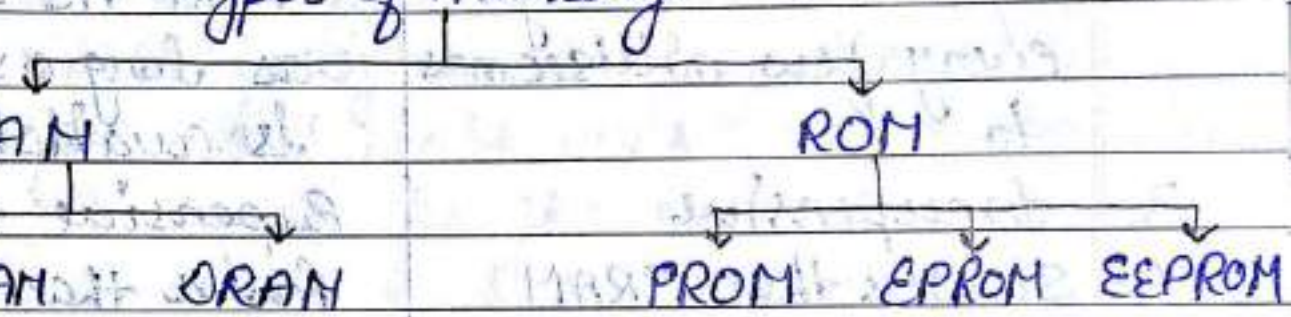
usually volatile memory.
Data is lost in case power is switched off.
It is the working memory of the computer.
Faster than secondary memories.
A computer cannot run without the primary memory.

Secondary memory :- This type of memory is also known as external memory or non-volatile. It is slower than the main memory. Eg- disk, CD-ROM, DVD etc.

1- These are magnetic and optical memories.
2- It is known as the back-up memory.
3- It is a non-volatile memory.
4- Data is permanently stored even if power is switched off.
5- It is used for storage of data in a computer.
6- Computer may run without the secondary memory.

7- Slower than primary memories.

* Types of Computer memory :-



1- Random Access Memory (RAM) -

(i) It is also called as read write memory or the main or primary memory.

(ii) The programs and data that the CPU requires during execution of a program are stored in this memory.

(iii) It is a volatile memory as the data loses when the power is turned off.

(iv) RAM is further classified into two types - SRAM and DRAM.

PAGE NO. // //
DATE // //

	DRAM	SRAM
1-	Constructed of tiny capacitors that leak electricity.	Constructed of circuits similar to D flip-flops.
2-	Requires a recharge every few milliseconds to	Hold its contents as long as power is available.
3-	Inexpensive	Expensive
4-	Slower than SRAM	Faster than DRAM.
5-	Can store many bits per chip	Cannot store many bits per chip.
6-	Uses less power	Uses more power.
7-	Generates less heat	Generates more heat.
8-	Used for main memory	Used for cache.

2- Read only memory (ROM) -

(i) Stores crucial information essential to operate the system, like the program essential to boot the computer.

(ii) It is non-volatile.

(iii) Always retains its data.

(iv) Used in embedded systems or where the programming needs no change.

(v). Used in calculators and peripheral devices.

Types of ROM -

1- PROM (Programmable read-only memory) - It can be programmed by user. Once programmed, the data and instructions in it cannot be changed.

2- EPROM (Erasable programmable read-only memory) - It can be reprogrammed. To erase data from it, expose it to ultra violet light. To reprogram it, erase all the previous data.

3- EEPROM (Electrically erasable programmable read only memory) - The data can be erased by applying electric field, no need of ultra violet light. We can erase only portions of the chip.

PAGE NO. // //
DATE // //

	RAM	ROM.
1-	Temporary storage Store data In MBs volatile	Permanent storage Store data In GBs Non-volatile.
2-	Used In normal operations writing data is faster	used for startup process of computer. Writing data is slower.

*. Buses:- components communicate with each other using buses. A bus is a set of parallel wires connecting different components. The processor is connected to the main memory by three seperate buses. The three types of buses -

1. Address bus - Address bus is apathway or a set of parallel wires that carries the location of the data to be read from or written to. An address bus is one directional only - from the processor to the memory or an I/o controller.

PAGE NO. // //
DATE // //

The number of lines for the address bus determines the maximum number of bits it can carry, and in turn determines the maximum possible memory capacity of the computer. Ex- Think if the postal office only allows 3 digits for house numbers, the the maximum addressable houses will be 999. So, if the computer has a 32-bit bus, the maximum addressable memory locations will be 2^{32}, which is 4GB, assuming each memory location stores one byte of data.

2.- Data Bus:- Data bus is a bi-directional pathway or wires that carries data or instructions between computer components. The width of data bus is a key factor in determining the overall computer performance. Typical data bus is 8, 16, 32 or 64 bits wide. If a computer has a word size of 32 but with a 16-bit data bus, then the data bus has to fetch the

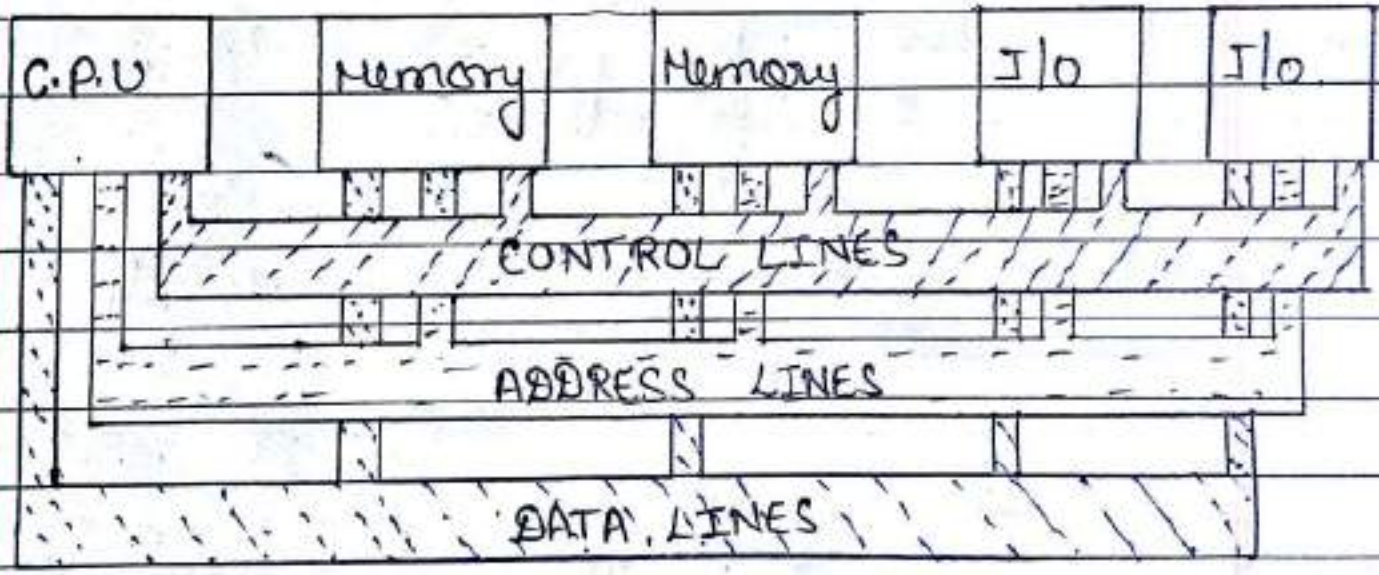
word twice from the main memory.

3:- Control Bus- Control bus is a bi-directional pathway that carries command, timing and specific status information among components. The following are the some of the control information a control bus may carry-
Write to memory
read to from memory
Write to I/O port
read from I/O port
request for data bus
grant for data bus
Sync clock
reset all components.

I/O controllers :- An I/O controller is a device (an electronic circuit board) that manages the communication between the processor and the I/O device.

PAGE NO. // //
DATE // //

Each Input/O device has a seperate controller which connects to the control bus.
I/o controllers receive input and output request from the processor, then send device specific control signals to the device.
I/o controllers manage the data flow from and to the device.



PAGE NO. // //
DATE // //

UNIT-2

General Purpose register

A. 8086 General Data Register (Register organisation) :-
8086 has a powerful registers known as general purpose register. All the 16 bit register, these registers also use at 8 bit registers, registers are use for holding data, variable and temporary storage of data.
Four types of registers-
1.- General data registers
2.- Segment registers
3.- Pointer and Index registers.
4.- Flag registers.

1.- General data registers-

AX	AH	AL
BX	BH	BL
CX	CH	CL
DX	DH	DL

- AX is use as 16 bit accumulator.
AL is low 8 bit
HH is high 8 bit

PAGE NO. // //
DATE // //

- BX use as offset storage to form physical address.
- CX use as default counter in string and loop instruction.
- DX use as operands and destination instructions.

2- 8086 Segment Register - 8086 has a powerful registers known as general purpose register, segment register is second type registers. In which 1 megabyte memory which 8086 addresses. Each segment contains 64 Kbyte

CS	code segment register use for addressing memory location.
SS	Stack segment register use addressing stack segment of memory.
DS	Data segment register use data segment of memory.
ES	Extra segment register also use data of memory.

3- Pointer and Index registers contains offsets with a particular segment -

SP	IP, SP, BP, offset.
BP	
SI	SI use for store offset of source data in data segment.
DI	DI is use for store offset of destination data in data segment.
IP	

4- Flag Register -

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	O	D	I	T	S	Z	X	Ac	X	P	X	Cy

S = Sign flag = If result is -ve this flag is set it is denoted by MSB.

Z = Zero flag = If previous Instructions is Zero this flag is set.

P = Parity flag = If lower byte of result contain even no. of 1's this flag is set.

Cy = Carry flag = If carry produce in MSB this flag is set.

T = Trap flag = If process in single step execution mode this flag is set.

PAGE NO. // //
DATE // //

I= Interrupt flag = If Maskable Interrupt detect by CPU this flag is set.
D= Direction flag = this flag is 0 string is process in auto increment mode, and 1 in auto decrement mode.
AC= auxiliary carry flag = this flag is set if carry from lower nibble (bit
O= Overflow flag = this flag is if set if result is large then a destination register.

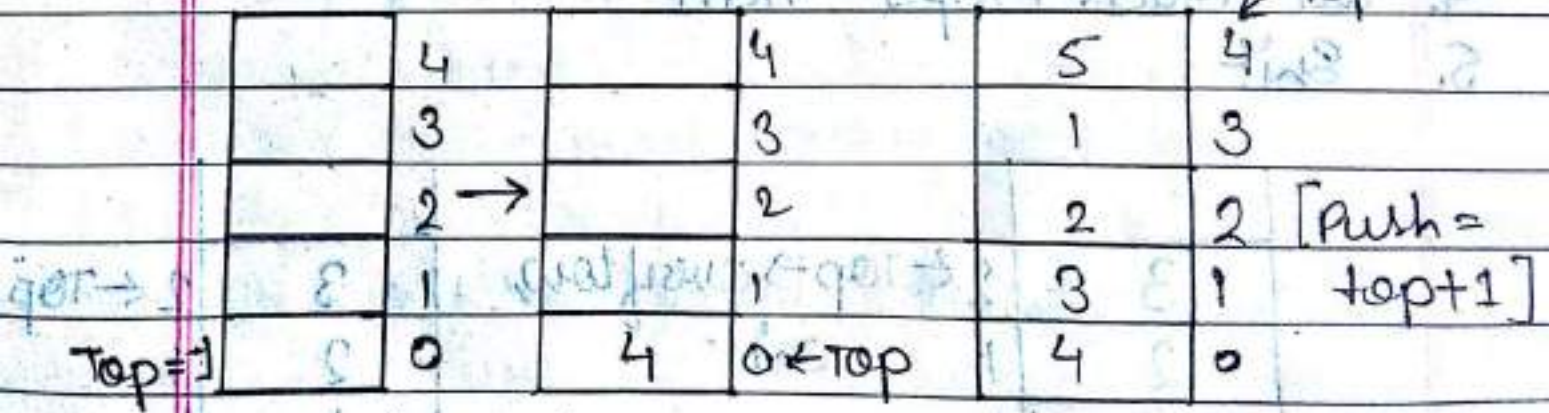
A. Stack Structure of 8086 or 8088:-
Stack is sequence set of data which PUSH and POP instruction work.
Stack is use to save useful data.
At the starting of subroutine all the register contain main program push into the stack one by one.

How we can calculate stack top address
Stack pointer register 16 bit register contain offset address in stage segment.

It is 64 byte memory location.
SS stack segment contain base address of stack segment in memory.
SS stack segment and SP make a address together stack top.

Stacks :- A data structure in which last item inserted is taken out first LIFO (Last In First out).

- Only one item is inserted / Push at a time on top of stack.
- only one item is deleted / Pop at a time from or top of stack.



Stack is empty

Pop = {top = top - 1}.

	4		4	
1	3 ← Top		3	
2	2		2	
3	1		1	top = -1
4	0		0	
		Empty		

★ Push operation –

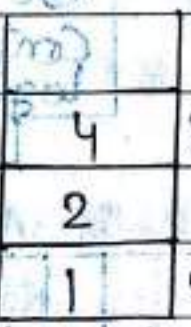
1.- Stack overflow ?
If Top = max – Stack, write overflow and exit.
2. Read Item.
3. Set top = top + 1
4. Set stack [top] = Item
5. Exit

3	2 ← Top → overflow, exit		3	2 ← Top
2	1		2	1
1	0		1	0

★ Pop operation –

PAGE NO. // //
DATE // //

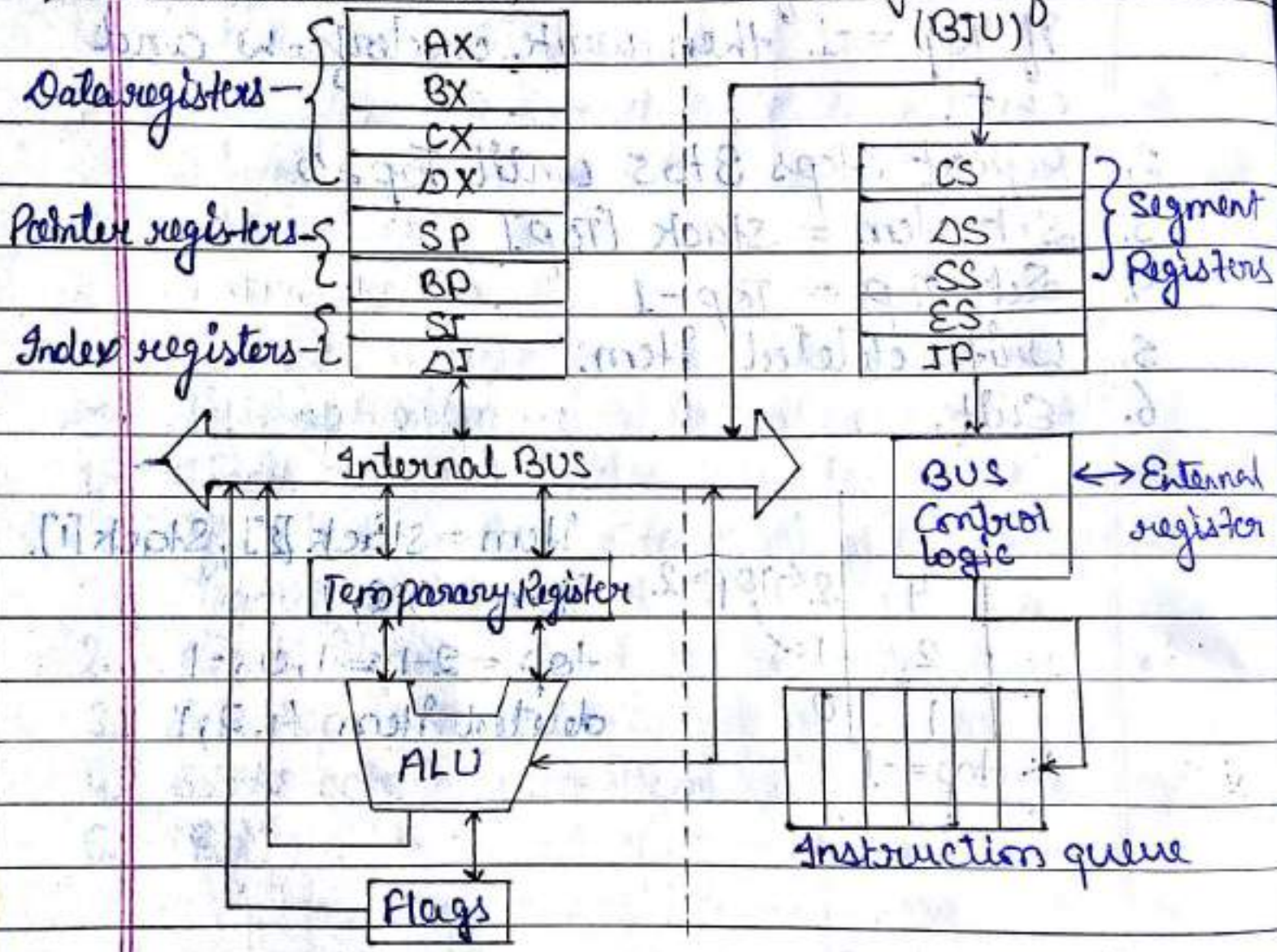
1. Stack overflow ?
   If Top = -1 then write underflow and exit.
2. Repeat steps 3 to 5 until Top ≥ 0
3. Set Item = stack [Top]
4. Set Top = Top - 1
5. Write deleted Item.
6. Exit.



Item = Stack [2], Stack [1].
Item = 4, 2, 1
top = 2-1 = 1, 0, -1
deleted Item 4, 2, 1

PAGE NO. // //
DATE // //

A. Execution unit (EU) | Bus Interface unit (BIU)



Internal Architecture of 8086 Microprocessor.

★. Data lines :-

They provide a path for moving data among system modules.

PAGE NO. // //
DATE // //

- These lines are collectively refered as data Bus.
- The number of lines in Data Bus is called width of the data bus.
  like, 16, 32, 64, 128 or more.
- Each line carries 1 bit at a time, so, the no. of lines in data bus determine how many bits can be transferred parrallely.
- Data bus width determines overall system performance.

* Address lines :-
  Specify where to store the data from where to retrieve the data (present on the data bus).
- Collectively, address lines are known as Address Bus.
- Address Bus width determines the memory capacity → Total no. of unique addresses/ locations.
- The same address lines can refer to I/O ports.

PAGE NO. // //
DATE // //

1- Designate High order bits to specify the Module that we want to access.
2- Low order bits specify location within memory or a particular part.

CONTROL LINES:-
They are used to control the access to the Address Bus and Data Bus and monitor their use.

Why?
They are required because A.B. and D.B. are shared between components and a mechanism to control their use is required.

How?
Generates control signals.

Timing Information
- Timing signals tell validity of data and address on DB and AB.

Command Info.
- Specify the operations to be performed.

PAGE NO. // //
DATE // //

- Memory Write : Data → Memory
- Memory Read : Memory → Data Bus
- I/O Write : Data Bus → I/O Port
- I/O Read : I/O Port data → Data Bus
- Transfer ACK : Indicates success (Data)
- Bus Request : Req. to again control of bus.
- Bus Grant : Control granted.
- Interrupt Request : Pending Interrupt
- Interrupt ACK : Pending INTR recognised.
- Clock : Synchronization of operation.
- Reset : Initialization of modules.

A. Addressing Modes for Control transfer Instructions :-

Addressing modes for control transfer instructions. They are :

1- Intra Segment Direct Mode :-

If the location to which control is to transferred is in the same segment, it is called Intra segment mode. If address to which the control is to be transferred appears directly in the instruction as a displacement value, it is what

called as Intra Segment direct mode.

2.- Intra Segment Indirect Mode :-
If the location to which control is to transferred is in the same segment, it is called Intra segment mode. If address to which the control is to be transfered appears indirectly in the instruction, it is what called as Intra segment direct mode.

3.- Inter Segment Direct Mode :-
If the location to which control is to transferred is not in the same segment, it is called inter segment mode. If address of segment to which the control is to be transferred and and location in the segment appears directly in the instruction, it is what called as inter segment direct mode.

4.- Inter Segment Indirect mode :-
If the location to which control is to transferred is not in the same segment, it is called inter segment mode.

If address of segment to which the control is to be transferred and location in the segment appears indirectly in the instruction, it is what called as inter segment indirect mode.

A Addressing modes:-

1.- Addressing modes are nothing but the different ways in which the location of an operand can be specified in an instruction. The number of addressing modes that a processor supports changes according to the instruction set it is based on, however there are a few generic ones that are present in almost all processors and are thus of utmost importance.

2.- They are as follows:

1) Immediate mode.
2) Register mode.
3) Indirect mode.
4) Index mode
5) Base with Index
6) Base with Index and offset
7) Relative

PAGE NO. // "
DATE // //

1:- Immediate mode :- In this mode, the operand is specified in the instruction itself.

2:- Register mode :- The operand is the contents of a register. We specify the operand in this case by specifying the name of the register in the instruction. Processor register often used for intermediate storage during arithmetic operations. This addressing mode is used at that time to access the registers.

3:- Indirect mode :- The effective address (E.A.) of the operands is the contents of a register or the memory location whose address appears in the instruction. The name of the register or the memory address is placed in parentheses to denote indirection or in other words that the contents are addresses of the operands.

4:- Index mode :- The effective address of the operand is calculated by adding a constant

value to the contents of a register, which is clearly shown in the address can be in a register used specially for this purpose or any of the general purpose registers. In other case it is called as an Index register.

5.- Base with Index mode :- The effective address is the sum of contents of two registers. The first register as before is called the Index and the second register is called the base register. This mode provides more flexibility since both the components are registers and can thus be changed.

6.- Base with Index and offset mode :- The effective address is the sum of contents of two registers and a constant. The constant value in this case is often called the offset or the displacement.

7.- Relative mode :- For relative addressing, also called PC-relative addressing, the

PAGE NO. // //
DATE // //

Implicitly referenced register is the program counter (PC). That is, the next instruction address is added to the address field to produce the EA, typically, the address field is treated as a twos complement number for this operation.

Example:-

- Auto Increment mode :- The effective address of the operand is the contents of a register specified in the instruction. After accessing the operand, the contents of this register are automatically incremented to the next value. This increment is 1 for byte sized operands, 2 for 16 bit operands and so on.

- Auto decrement mode :- The effective address of the operand is the contents of a register specified in the instruction. Before accessing the operand, the contents of this register are automatically decremented and then the value is accessed.

PAGE NO. // //
DATE // //

★ Data Transfer and Manipulation :-
Data Transfer Instructions cause transfer of data from one location to another without changing the binary information. The most common transfer are between the.

- Memory and Processor registers.
- Processor registers and Input output devices.
- Processor register themselves.

Typical Data Transfer Instructions.

Name	Mnemonic
load	LD
Store	ST
Move	MOV
Exchange	XCH
Input	IN
Output	OUT
Push	PUSH
Pop	POP

PAGE NO. // //
DATE // //

A. Data manipulation Instructions.

Data manipulation Instructions perform operations on data and provide the computational capabilities for the computer. These Instructions perform arithmetic, logic and shift operations.

1- Arithmetic Instructions:-

Name	Mnemonic
Increment	INC
Decrement	DEC
Add	ADD
Subtract	SUB
Multiply	MUL
Divide	DIV
Add with carry	ADDC
Subtract with borrow	SUBB
Negate (2's complement)	NEG

2- Logical and Bit Manipulation Instructions:-

Name	Mnemonic
Clear	CLR
Complement	COM

AND	AND
OR	OR
Exclusive-OR	XOR
Clear carry	CLRC
Enable Interrupt	SETC CI
Disable Interrupt	DI
Set carry	SETC
Complement carry	COMC.

3- Shift Instructions:-

Name	Mnemonic.
Logical shift right	SHR
Logical shift left	SHL
Arithmetic shift right	SHRA
Arithmetic shift left	SHLA
Rotate right	& ROR
Rotate left.	ROL
Rotate right through carry	RORC
Rotate left through carry.	ROLC

PAGE NO. // //
DATE // //

A. What is Pipelining?

Pipelining is the process of accumulating instruction from the processor through a pipeline. It allows storing and executing instructions in an orderly process. It is also known as pipeline processing. Pipeline is a technique where multiple instructions are overlapped during execution. Pipeline is divided into stages and connected with one another to form a pipe like structure. Instruction enters from one end and exit from another end.

Pipelining increases the overall instruction throughput. In pipeline system, each segment consists of an input register followed by a combinational circuit. The register is used to hold data and combinational circuit performs operations on it. The output of combinational circuit is applied to the input register of the next segment.

PAGE NO. // //
DATE // //

Types of pipeline - It is divided into 2 categories -

1.- Arithmetic pipeline.
2.- Instruction pipeline

1.- Arithmetic pipeline - Arithmetic pipelines are usually found in most of the computers. They are used for floating point operations, multiplication of fixed point numbers etc.

2.- Instruction pipeline - In this a stream can be executed by overlapping fetch, decode and execute phases of an instruction cycle. This type of technique is used to increase the throughput of the computer system.

Advantages of pipelining -

1.- The cycle time of the processor is reduced.
2.- It increases the throughput of the system.
3.- It makes the system reliable.

PAGE NO. // //
DATE // //

Disadvantages of pipelining :-

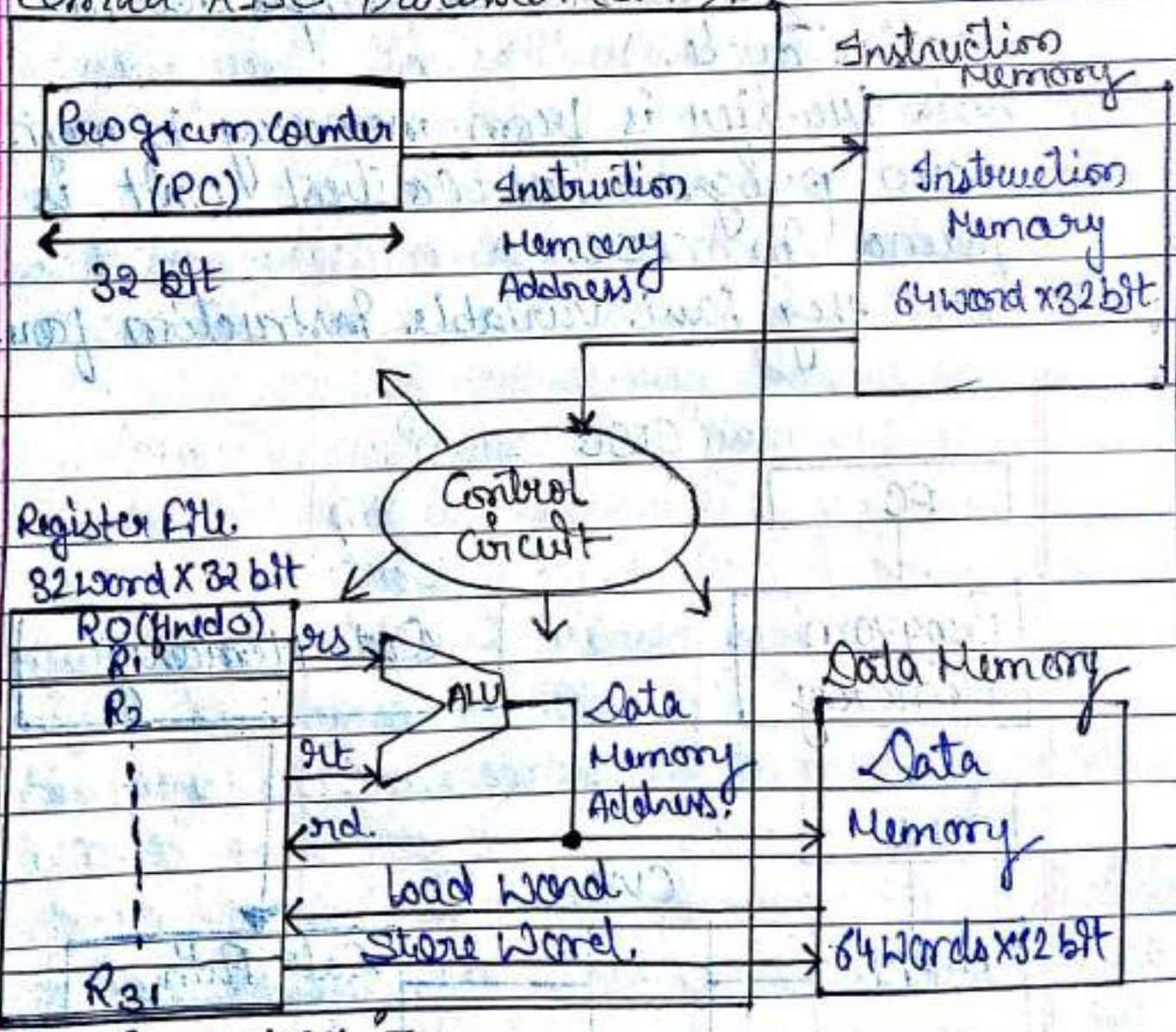
1.- The design of pipelined processor is complen and costly to manufacture.
2.- The Instruction latency is more.

★ RISC Processor :-

It is known as Reduced Instruction set Computer. It is a type of microprocessor that has a limited number of instructions. They can execute their instructions very fast because instructions are very small and simple.
RISC chip requires fewer transistor which make them cheapers to design and produce.
In RISC, the instruction set contains simple and basic instructions from which more complen instructions can be produced. Most instructions complete in one cycle, which allows the processor to handle many instructions at same time.

In this, Instructions are register based and transfer takes place from register to register.

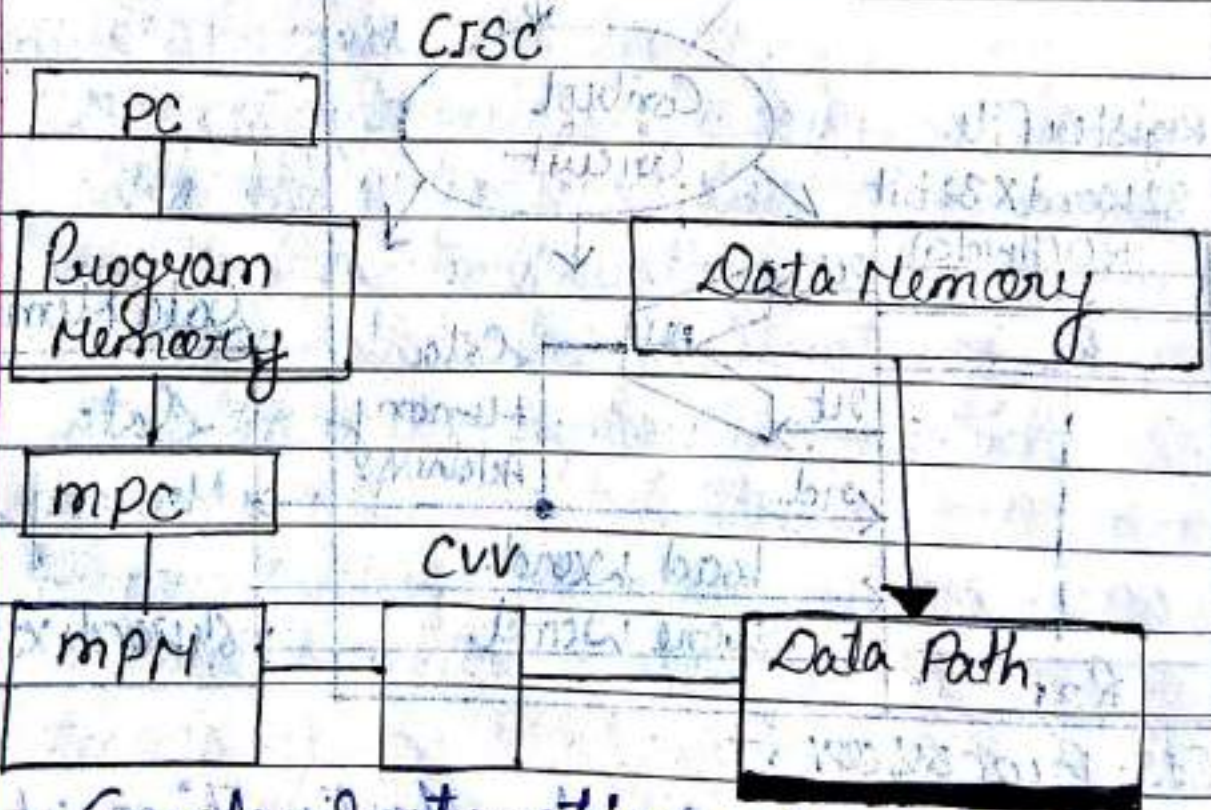
Small RISC Processor (SRP).



A. CIS Processor :-

1- It is known as Complex Instruction set Computer.

2.- It was first developed by Intel.
3.- It contains large number of complex Instructions.
4.- In this Instructions are not register based.
5.- Instructions cannot be completed in one machine cycle.
6.- Data transfer is from memory to memory.
7.- Micro programmed control unit is found in CISC.
8.- Also they have variable Instruction formats.



Complex Instructions
One Instruction in several CVV PM standard.

PAGE NO. // //
DATE // //

A. Difference between CISC and RISC

Architectural characteristics	Complex Instruction Set Computer (CISC)	Reduced Instruction Set Computer (RISC)
Instruction size and format	Large set of instructions with variable formats (16-64 bits per instruction).	Small set of instructions with fixed format (32 bit).
Data transfer CPU control	Memory to memory. Most micro coded using control memory (ROM) but modern CISC use hardwired control.	Register to register. Mostly hardwired without control memory.
Instruction type	Not register based instructions	Register based instructions.
Memory access	More memory access	less memory access.
Clocks	Includes multi-clocks	Includes single clock
Instructions & nature	Instructions are complex.	Instructions are reduced and simple

PAGE NO. // //
DATE // //

Vector (Array) Processor and Its Types.

Array processors are also known as multiprocessors or vector processors. They perform computations on large arrays of data. Thus, they are used to improve the performance of the computer.

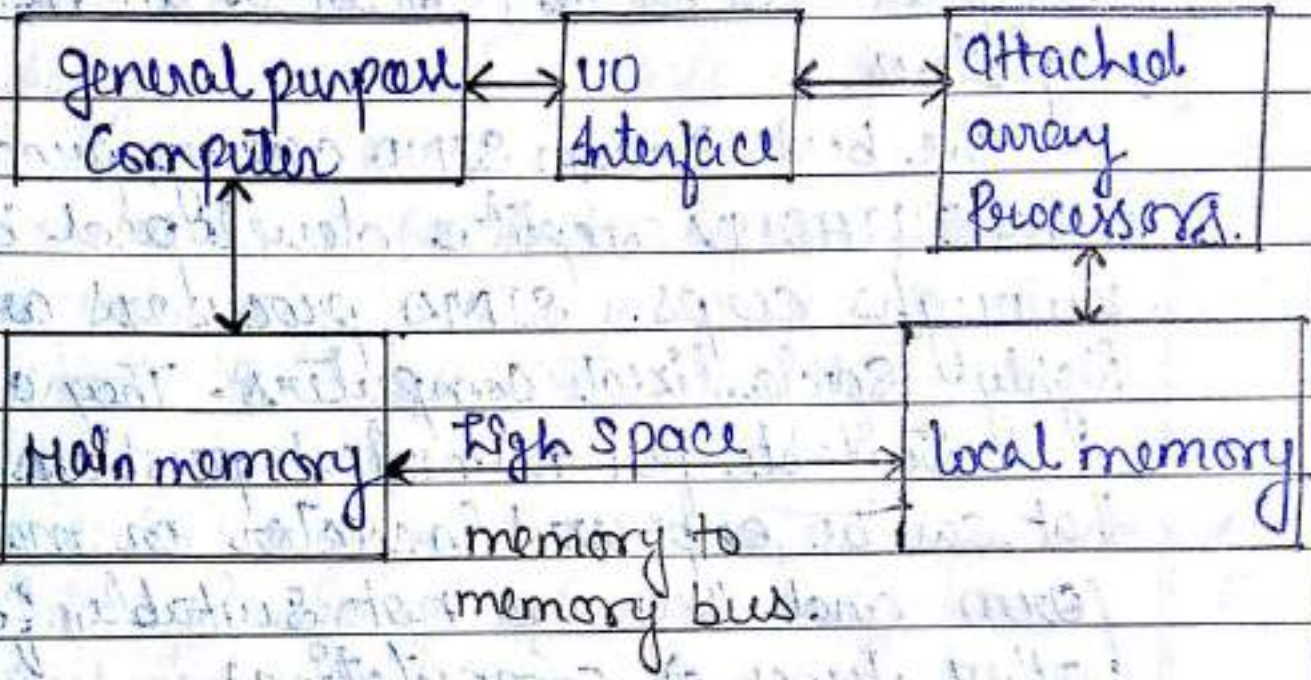
Types of array processors -

There are basically two types of any processors:

1. Attached array processors.
2. SIMD Array processors.

1. Attached Array Processors -

An attached array processor is a processor which is attached to a general purpose computer and its purpose is to enhance and improve the performance of that computer in numerical computational tasks. It achieves high performance by means of parallel processing with multiple functional units.



2:- SIMD (Single Instructions, multiple data) Array Processors :-

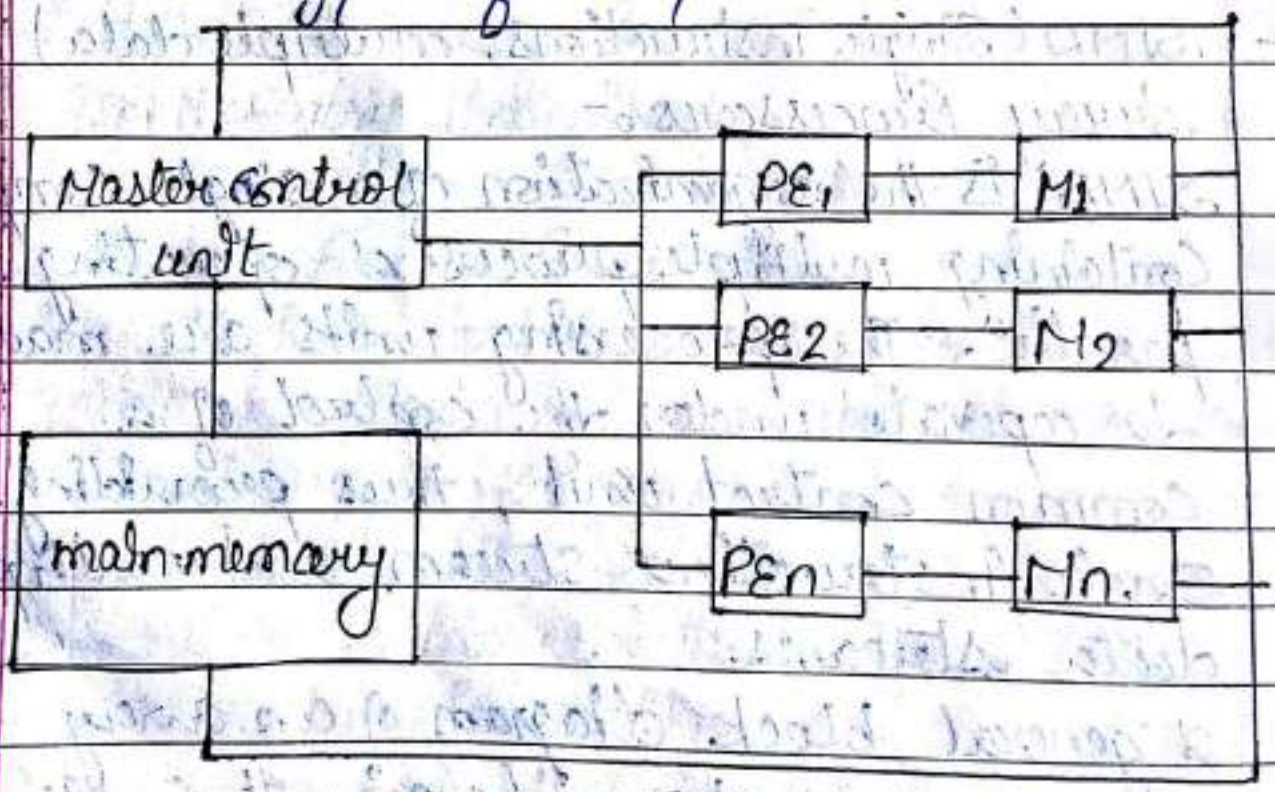
SIMD is the organization of a single computer containing multiple processors operating in parallel. The processing units are made to operate under the control of a common control unit, thus providing a single instruction stream and multiple data streams.

A general block diagram of an array processor is shown below. It contains a set of identical processing elements (PE's), each of which is having a local memory M.

PAGE NO. // //
DATE // //

Each processor element includes an ALU and registers.

The best known SIMD array processor is the ILLIAC IV computer developed by the Burroughs corps. SIMD processors are highly specialized computers. They are only suitable for numerical problems that can be expressed in vector or matrix form and they are not suitable for other types of computations.



PAGE NO. // //
DATE // //

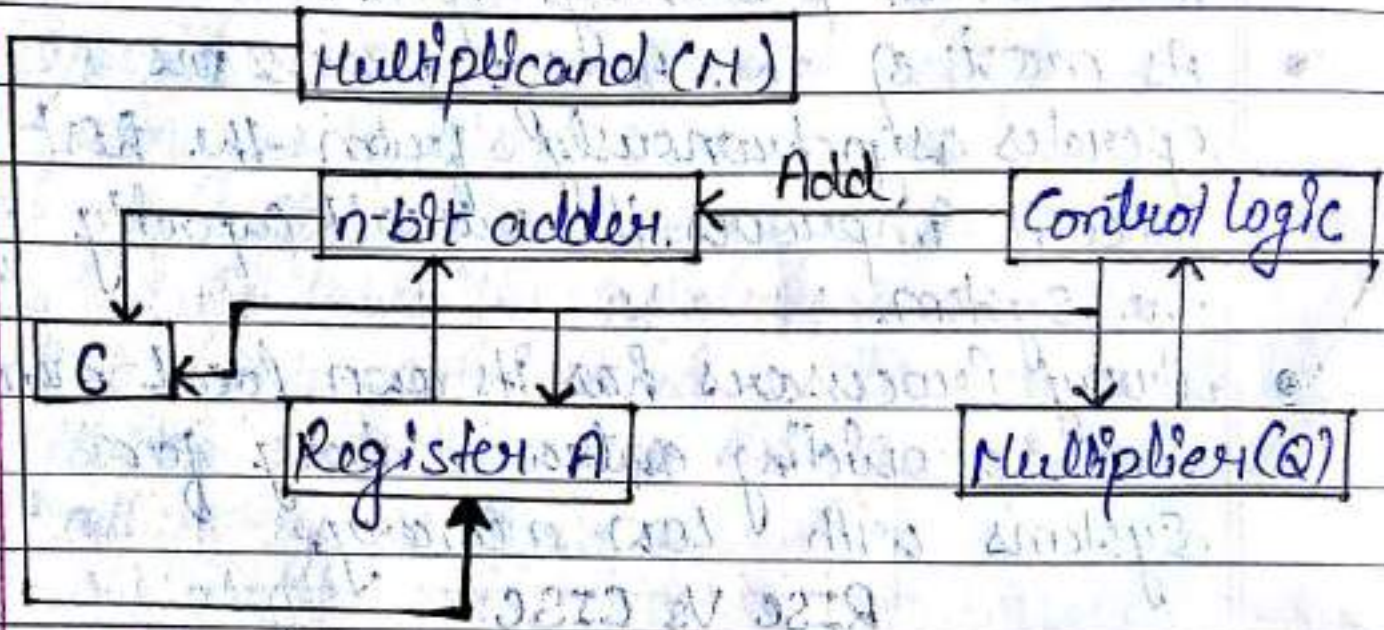
* Why use the Array Processor :-
- Array processor increases the overall instruction processing speed.
- As most of the Array processors operates asynchronously from the host CPU, hence it improves the overall capacity of the system.
- Array Processors has its own local memory, hence providing extra memory for systems with low memory.

RISC Vs CISC

Parameter	RISC	CISC
Instruction types	Simple	Complex.
Number of Instructions	Reduced (30-40)	Extended (100-200)
Duration of an Instruction	One cycle	More cycles (4-120)
Instruction formate	Fixed	Variable.
Instruction execution	In parallel (pipeline)	Sequential
Addressing modes	Simple	Complex
Instructions accessing the memory	Two: load and Store	almost all
Register set	multiple	from the set
Complexity	In compiler	unique. In CPU (micro-program).

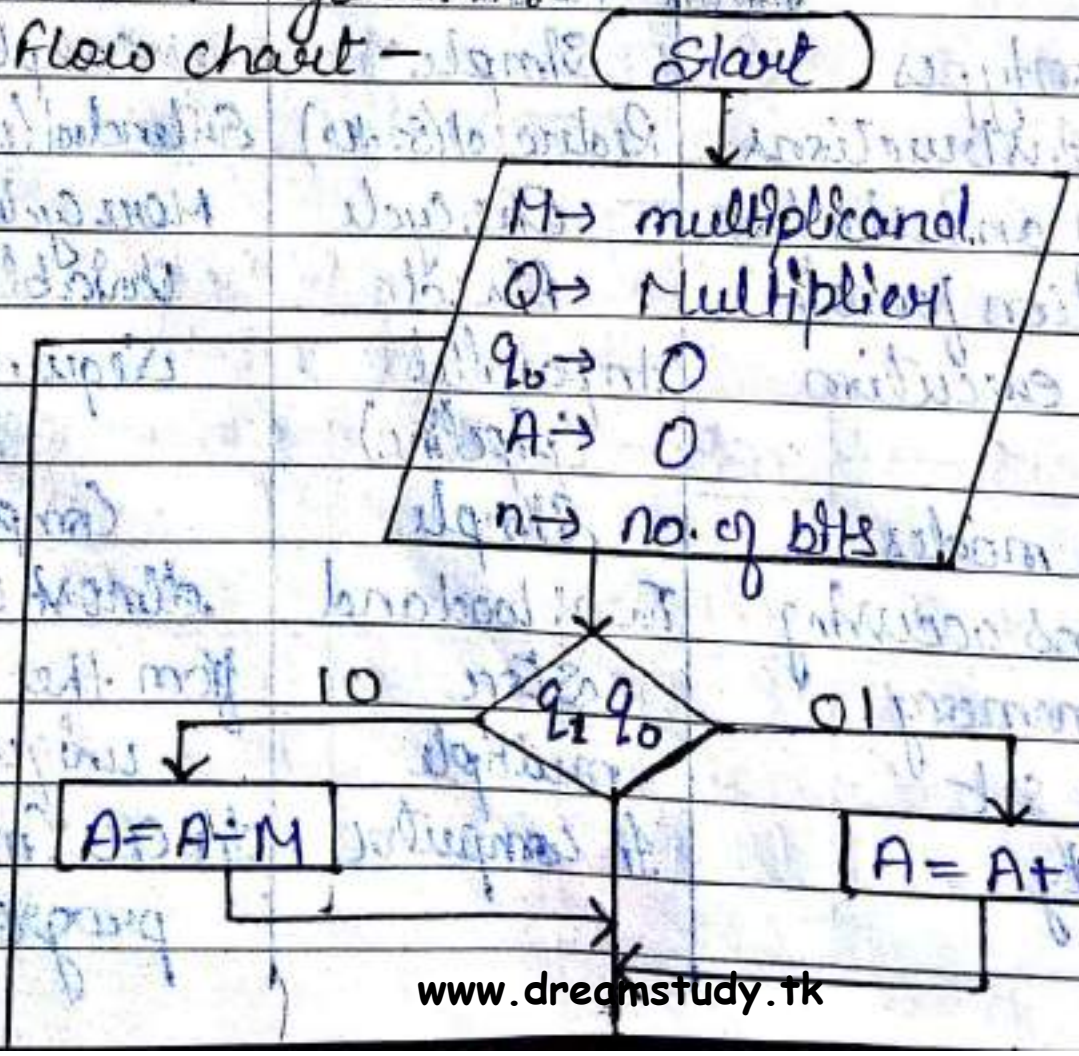
PAGE NO. // //
DATE // //

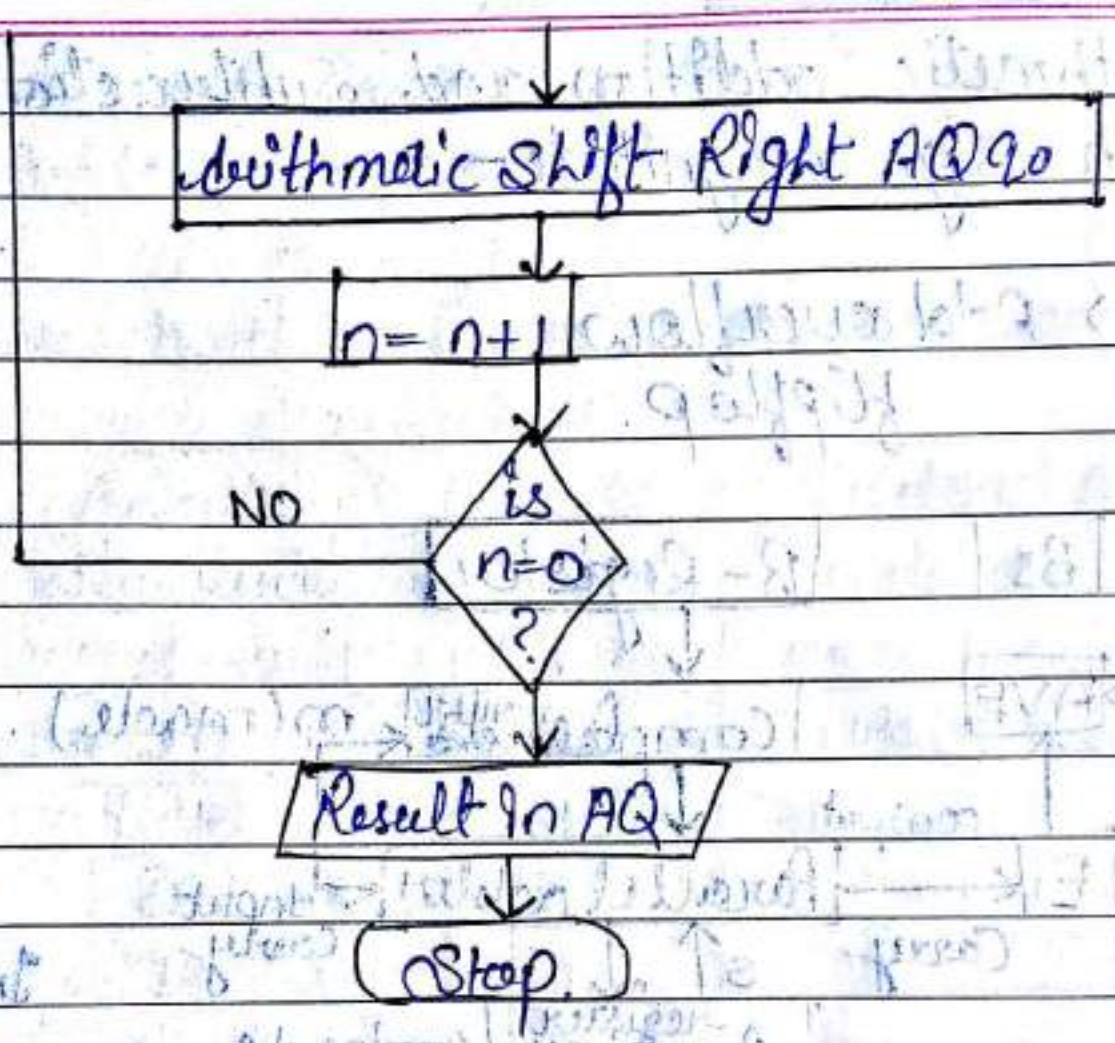
Hardware Structure of Multiplication, using Shift and Add.



Booth's algorithm :-

Flow chart -



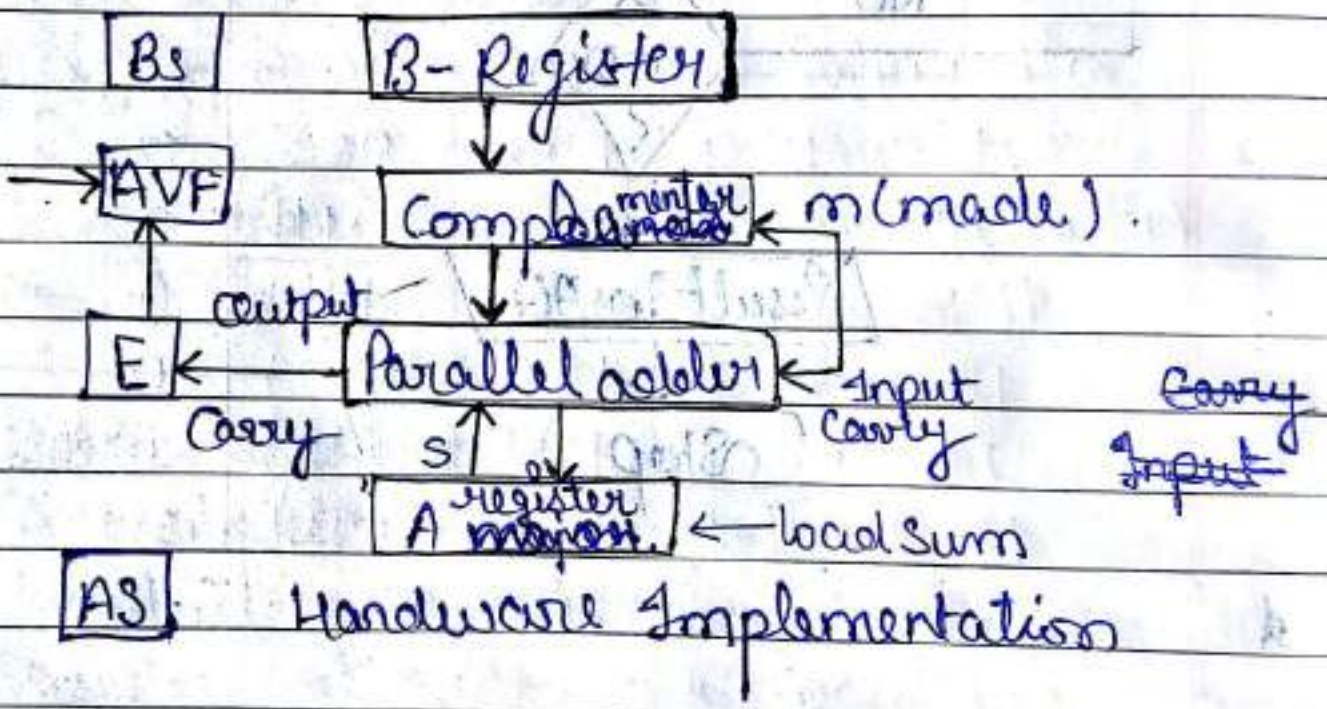


PAGE NO. // //
DATE // //

UNIT-3

★ Arithmetic addition and Subtraction with sign magnitude:-

AVF → Add overflow flipflop.

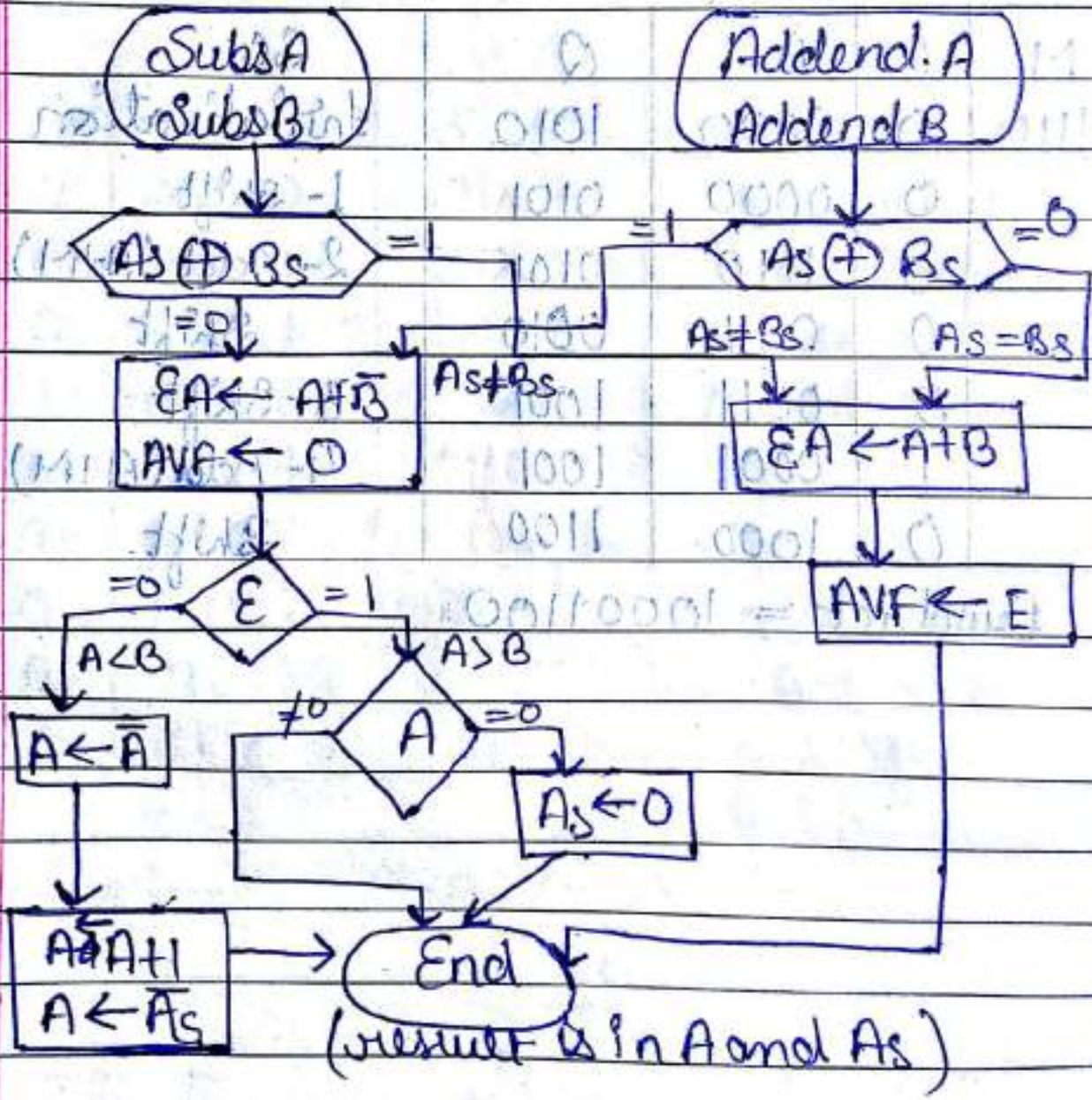


Binary addition and multiplication with sign magnitude example -

operation	Add.	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)	+(A-B)	-(B-A)	+(A-B)
+A - (-B)	+(A+B)			

-A-(+B)	-(A+B)			
-A-(-B)		-(A-B)	+(B-A)	+(A-B)

★ Flowchart :- (Hardware algorithm).



PAGE NO. // //
DATE // //

Q. Example of Booth multiplication algorithm-

1- 14 × 10.

14 → 1110 — M

10 → 1010 — Q

M	C	A	Q	SC
1110	0	0000	1010	Initialization
	0	0000	0101	1- Shift
	0	1110	0101	2- Add (A+M)
	0	0111	0010	Shift
	0	0011	1001	3- Shift
	1	0001	1001	4- Add (A+M)
	0	1000	1100	Shift

Product = 10001100

PAGE NO. // //
DATE // //

Multiplication with sign-magnitude (Booth's algorithm)-

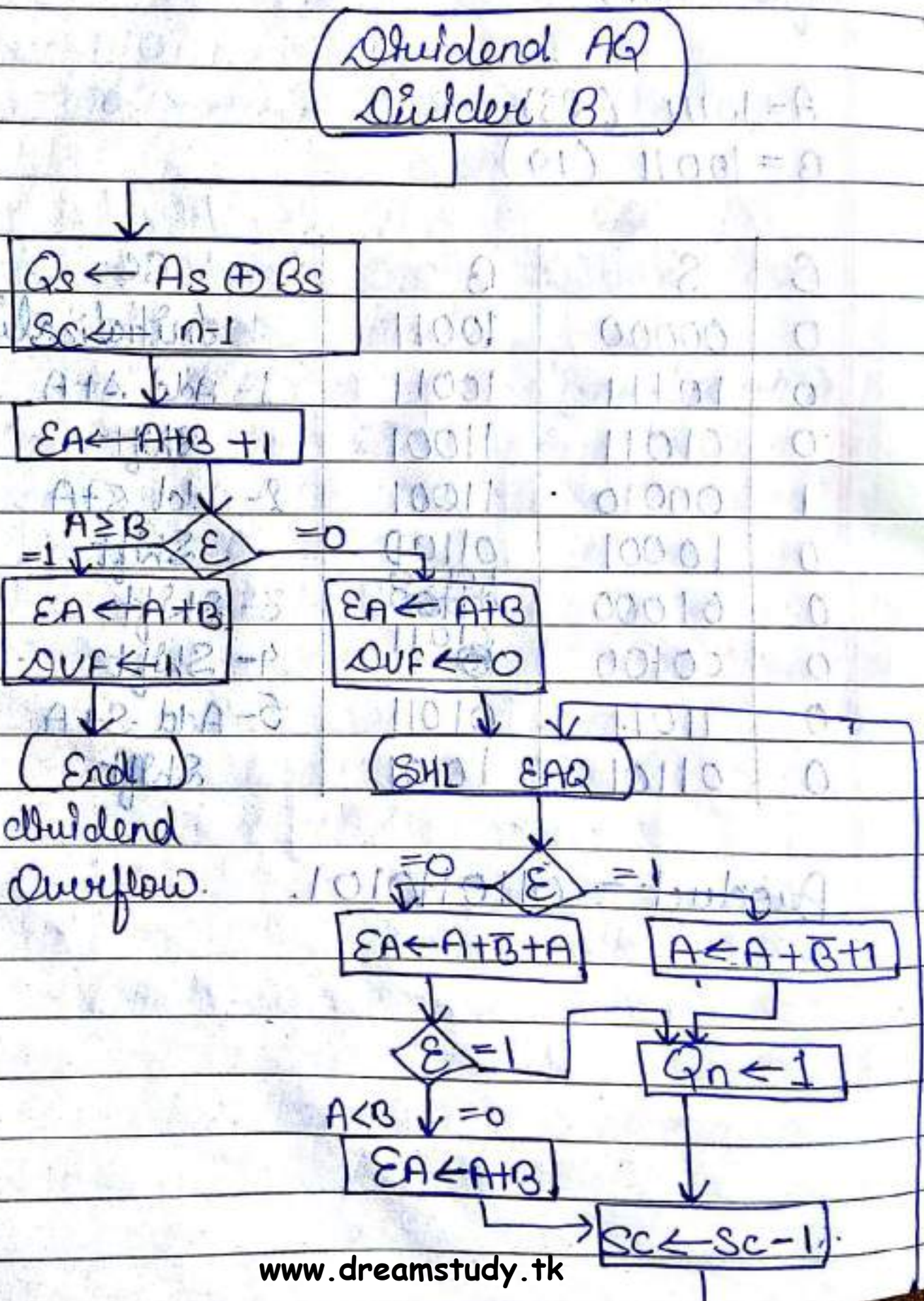
A = 10111 (23)
B = 10011 (19)

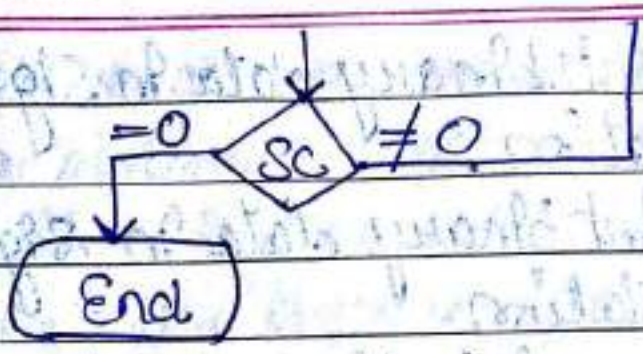
C	S	B	SC
0	00000	10011	Initialization
0	10111	10011	1- Add S+A
0	01011	11001	Shift
1	00010	11001	2- Add S+A
0	10001	01100	Shift
0	01000	00110	3- Shift
0	00100	00011	4- Shift
0	11011	00011	5- Add S+A
0	01101	10001	Shift

Product = 0110110101.

PAGE NO. // //
DATE // //

Binary Division flow-chart :-





★ Introduction:-
Arithmetic Instructions In digital computers manipulate data to produce results necessary for the solutions of computational problems. These Instructions perform arithmetic calculations and are responsible for the bulk of activity Involved In processing data in a computer.

★ What is algorithm?
✓ The Solution to any problem that is stated by a finite number of well-defined procedural steps is called an algorithm

✓ we develop the various arithmetic algorithms and show the procedure for Implementation them with digital hardware. we consider addition, Subtraction, multi, and division for the following types of data.

PAGE NO. // //
DATE // //

1 ✓ Fixed point binary data in signed-magnitude representation.
2 ✓ Fixed point binary data in signed-2's complement representation.
3 ✓ Floating point binary data.
4 ✓ Binary-coded decimal (BCD) data.

- Addition and Subtraction :-

The addition and subtraction algorithm for data represented in signed magnitude and again data represented in signed-2's complement.

It is important to realize that the adopted representation for negative numbers refers to the representation of numbers in the registers before and after the execution of the arithmetic operations.

- Addition and Subtraction with signed-magnitude data :-

The representation of nos. in signed-magnitude is familiar because it is used in everyday arithmetic calculations.

The pencil procedure for adding or subtracting two signed binary nos. with papers and pencils simple and straight-forward. A review of this procedure will be helpful for deriving the hardware algorithm.

- Hardware Implementation:-

To implement the two arithmetic operations with hardware, it is the first necessary that the two nos. be stored in registers. Let A and B be two registers that hold the magnitude of the nos, and As and Bs be two flipflops that hold the corresponding signs. The results of the operation may be transferred to a third register however, a saving achieved if the result is transferred into A and As. Thus A and As together form an accumulator register.

Consider now the hardware implementation of the algorithms above. First, a parallel adder is needed to perform the micro operations A+B.

PAGE NO. // //
DATE // //

Second, comparator circuit is needed to ~~perform the micro~~ establish if A>B, or A<B. third, two parallel subtractor circuits are needed to perform the micro operations: A-B and B-A. The sign relationship can be determined from an exclusive-OR gate with A_s and B_s as input.

★ Flowchart (Hardware algorithm):-

- The two signs A_s and B_s are compared by an exclusive-OR gate. For an add operation, identical signs dictate that the magnitudes be added, for subtract operation different signs dictate that the magnitudes be added. The magnitudes are added with a micro operation $EA \leftarrow A+B$. Where EA is a register that combines E and A.
- For A0 indicates that A<B, for this case it is necessary to take the 2's complement of the value in A. this operation can be done with one micro operation $A \leftarrow \bar{A}+1$.

PAGE NO. // //
DATE // //

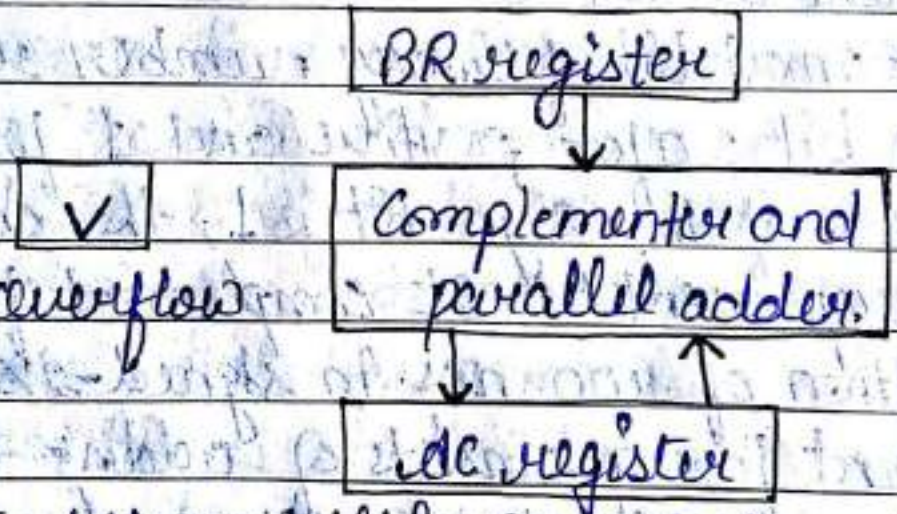
- However, we assume that A register as circuits for micro operation: complement and increment, so the 2's compliment is obtain from these two micro operations.
- The value in AVF provides an overflow indication.
- The final value of E is immaterial.

- Addition and Subtraction with Signed 2's complement data :-

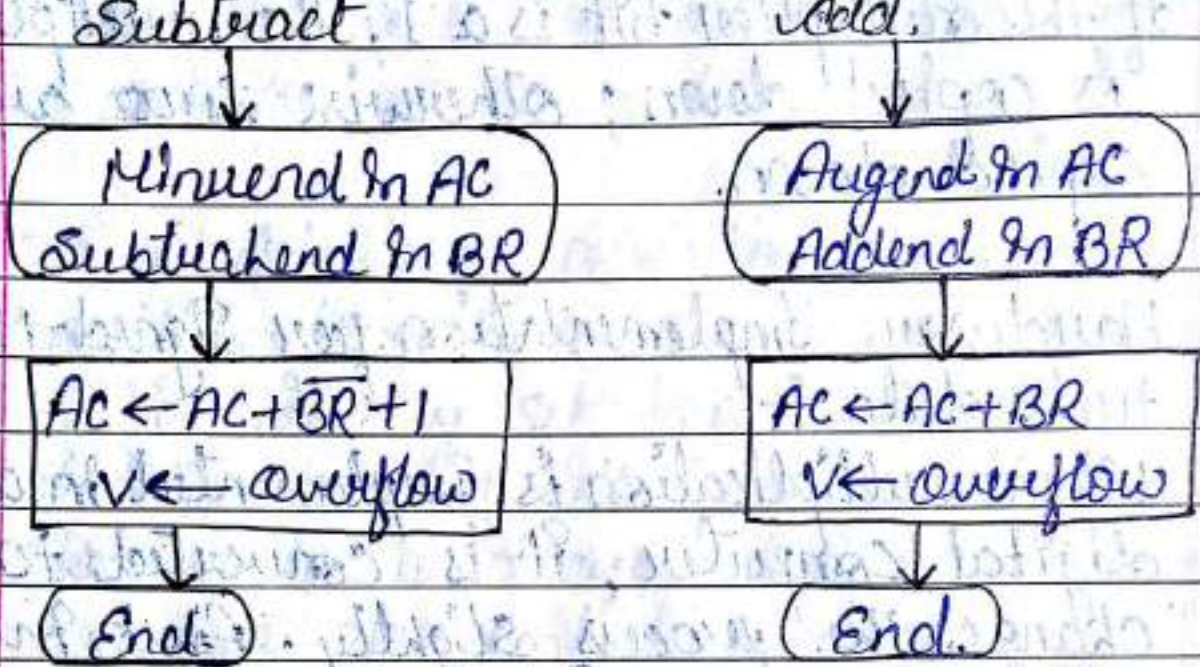
The left most bit of binary number represents the sign bit; 0 for positive and 1 for negative. If the sign bit is 1, the entire no. is represent in 2's complement form. The addition of two nos. in signed-2's complement form consists of adding the nos. with the sign bits treated the same as the other bits of the nos. A carry out of the sign bit position is discarded. The subtraction consists of first taking the 2's complement of the subtrahend and then adding it to the minuend.

When two nos. of n digits each are added and the sum occupies n+1 digits, we say that an overflow occured.
When the two carries are applied to an exclusive-OR gate, the overflow is detected when the output of the gate is equal to 1.

Hardware for signed-2's complement addition and subtraction



→ The left most bit in AC and BR represents the sign bits of the numbers.
→ The over flow flip-flops V is set to 1 if there is an overflow. The output carry in this case is discarded.



Algorithm for adding and subtraction nos. In signed 2's complement representation.

★ Multiplication algorithms:-

Multiplication of two fixed point binary numbers in signed magnitude representation is done with paper and pencil of successive shift and add operation.

```
 23          10111  → Multiplicand.
x 19        X10011  → Multiplier.
---         -----
437         10111
           10111X
          00000XX
         00000XXX
      + 10111XXXX
      ----------
        110110101   Product
```

If the multiplier bit is a 1, the multiplicand is copied down; otherwise zeros are copied down.

- Hardware Implementation for Signed-Magnitude data:-

→ When multiplication is implemented in a digital computer, it is convenient to change the process slightly. First instead of providing register to store and add simultaneously as many binary nos. as there are bits in the multiplier, as it is convenient to provide an adder for the summation of only two binary nos. and successively accumulate the partial products in a register. Second instead of shifting the multiplicand to the left, the partial product is shifted to the right.

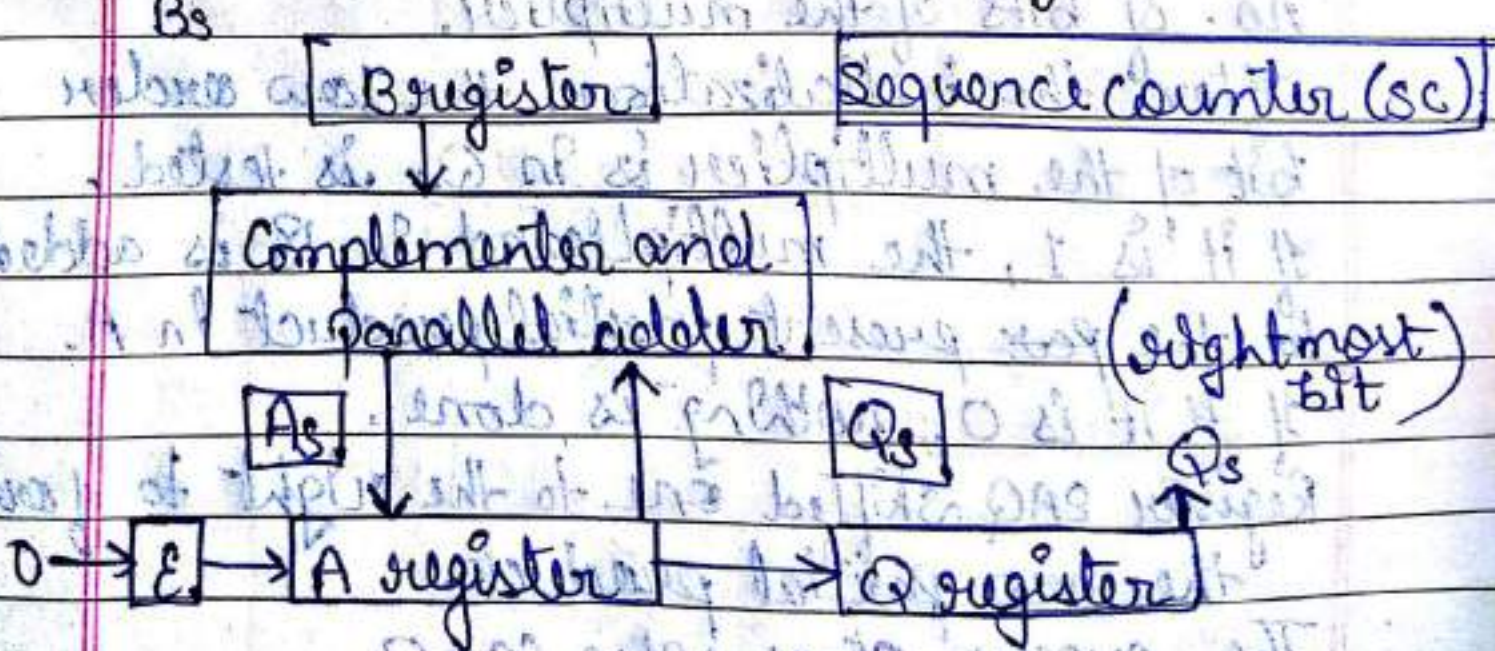
→ The hardware for multiplications consist of the equipment shown in fig. plus two are more registers.

→ These registers are together with registers A and B.

→ The multiplier stored in the Q register and its sign in Q. The Sequence counter SC is initially set to a number equal to the no. of bits in the multiplier. The counter is decremented by 1 after forming each partial product.

→ The sum of A and B forms a partial product which is transferred to the EA register.

Hardware for multiply operation



PAGE NO. // //
DATE // //

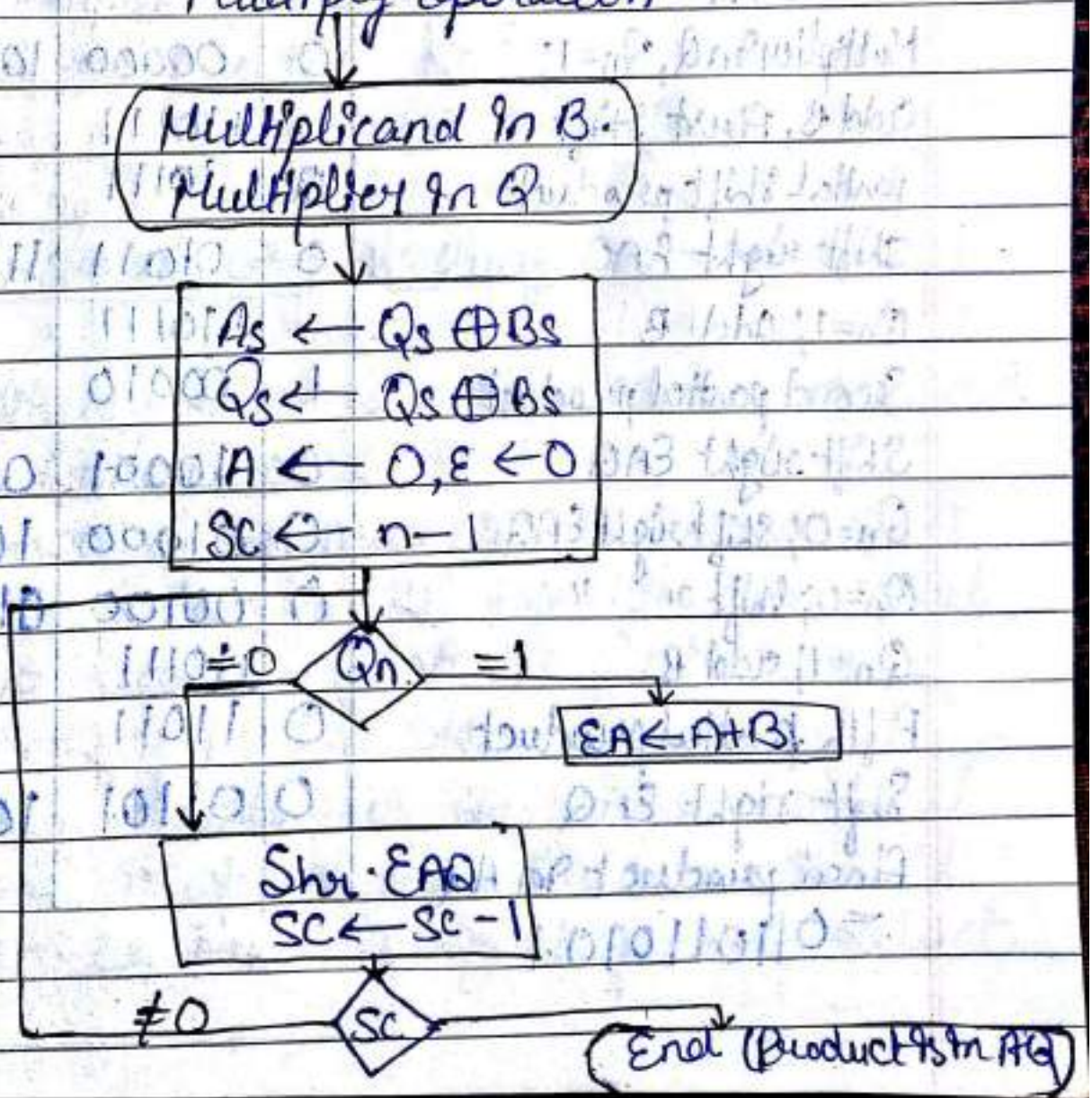
The shift will be denoted by the statement shr EAQ to designate the right shift depicted.
The least significant bit of A is shifted into the most significant position of Q.

- Hardware algorithm :-

Below fig is a flowchart of the hardware multiply algorithm. Initially the multiplicand is ~~in~~ in B and the multiplier in Q there corresponding signs are in Bs and Qs. respectively.
Register A and E cleared and the sequence counter SC is set to a no. equal to the no. of bits of the multiplier.
After the initialization, the low order bit of the multiplier is in Qn is tested.
If it is 1, the multiplicand in B is added to the ~~per~~ present partial product in A.
If it is 0, nothing is done.
Register EAQ shifted one to the right to form the new partial product.
The process stops when SC = 0.

Note that the partial product formed in A is shifted into Q one bit at a time and eventually replaces multiplier. The final product is available in both A and Q, with A holding the most significant bits and Q holding the least significant bits.

Flowchart for multiply operation.



PAGE NO. // //
DATE // //

Booth multiplication Algorithm :-

→ Booth algorithm gives a procedure for multiplying binary integers in signed - 2's complement representation.

Table → Numerical Example for Binary Multiplier

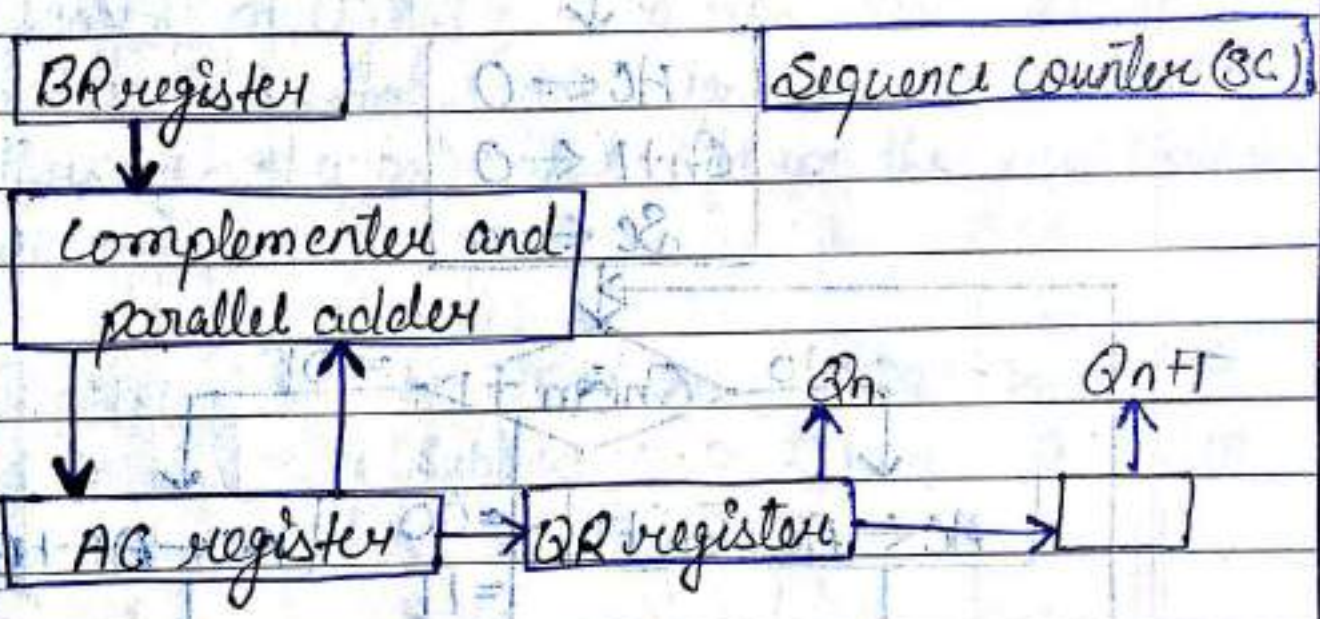
Multiplicand B = 10111	E	A	Q	SC
Multiplier in Q, Q_n = 1;	0	00000	10011	101
Add B, first ~~Shift~~		+10111		
partial Shift product	0	10111		
Shift right EAQ	0	01011	11001	100
Q_n = 1; add B		+10111		
Second partial product	1	00010		
Shift right EAQ	0	10001	01100	011
Q_n = 0; Shift right EAQ	0	01000	10110	010
Q_n = 0; Shift " "	0	00100	01011	001
Q_n = 1; add B.		+10111		
Fifth partial product	0	11011		
Shift right EAQ	0	01101	10101	000
Final product in AQ = 0110110101				

PAGE NO. // //
DATE // //

- Hardware for booth algorithm :-

→ The algorithm requires the register configuration as shown in fig.

(Hardware for booth algorithm)

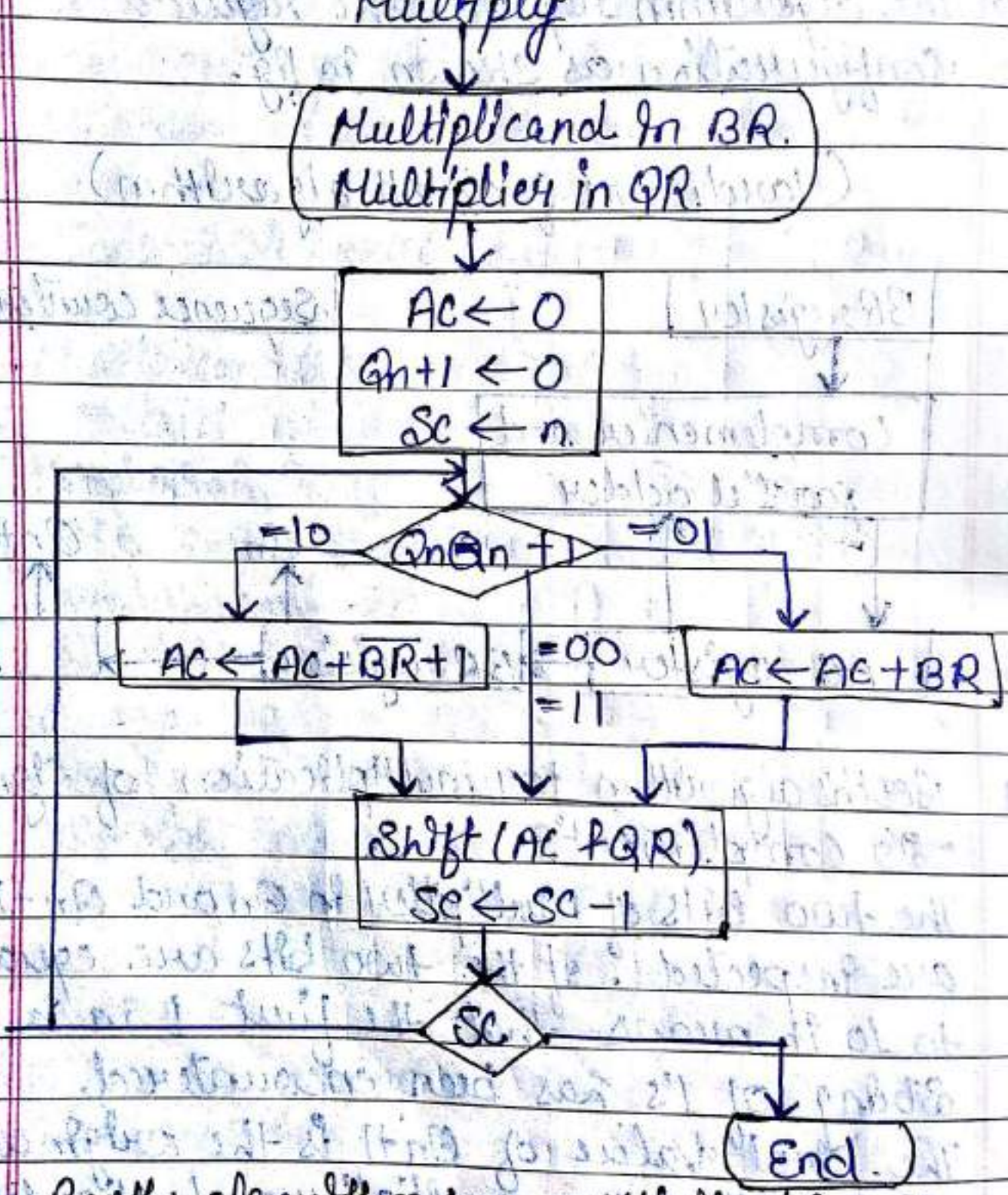


- Booth's algorithm for multiplication of Signed -2's complement :-

The two bits of multiplier in Qn and Qn+1 are inspected. If the two bits are equal to 10 it means that the first 1 in a string of 1's has been encountered. The final value of Qn+1 is the original sign bit of the multiplier and should not be taken as part of the product.

PAGE NO. // //
DATE // //

Multiplication Algorithms.



Booths algorithm for multiplication of Signed - 2's complement nos.

PAGE NO. // //
DATE // //

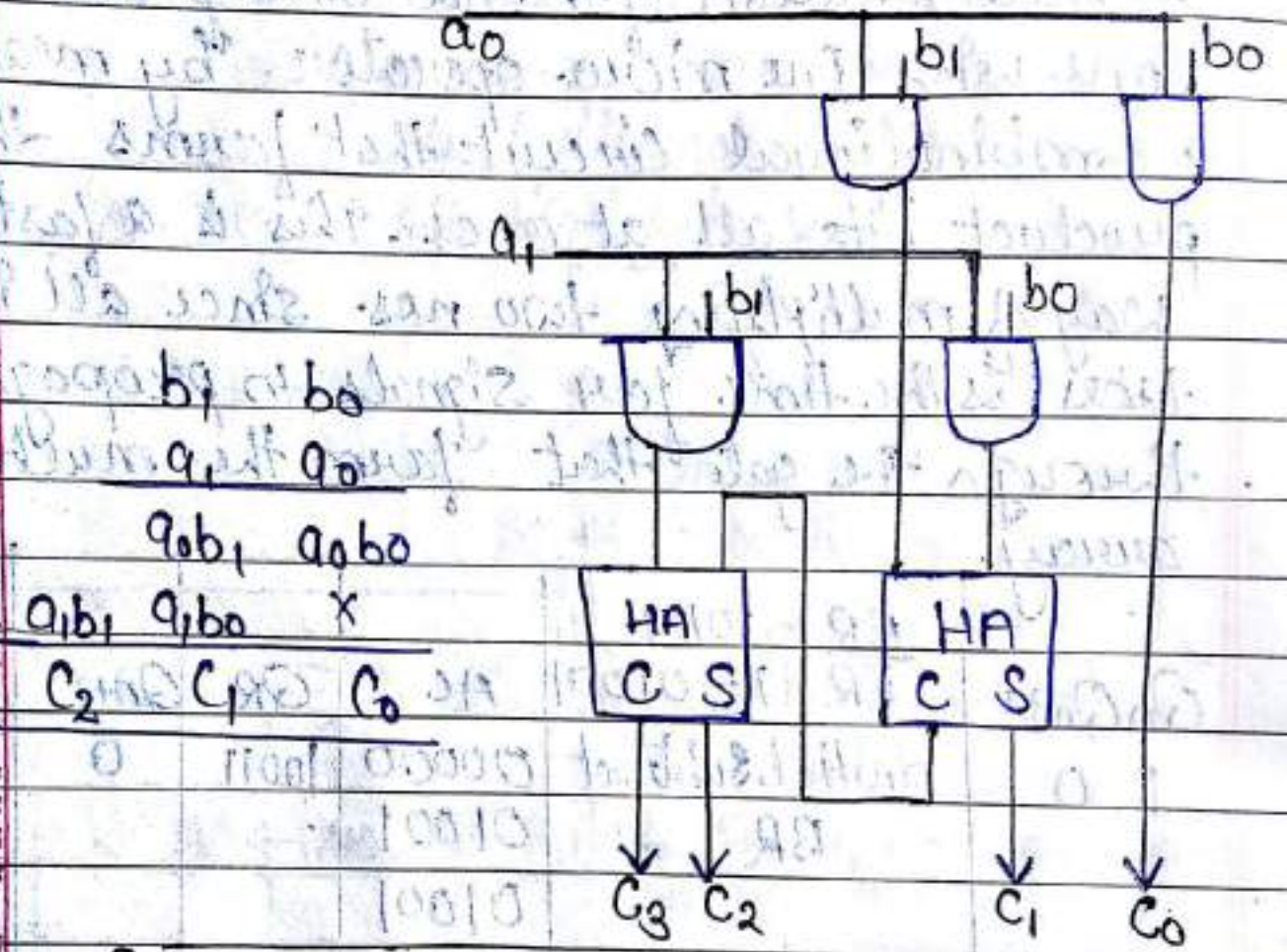
- **Array multiplier :-**

The multiplication of the two binary nos. can be done with one micro-operation by means of a combinational circuit that forms the product bits all at once. This is a fast way of multiplying two nos. since all it takes is the time for signals to propagate through the gate that form the multiplication array.

	BR = 10111				
	BR + 1 = 01001	AC	QR	Qn+1	SC.
Qn Qn+1					
1 0	Initial Subtract	00000	10011	0	101
	BR.	01001			
		01001			
	ashr.	00100	11001	1	100
1 1	ashr	00010	01100	1	011
0 1	Add BR	10111			
		11001			
	ashr	11100	10110	0	010
0 0	ashr	11110	01011	0	001
1 0	Subtract BR.	01001			
		00111			
	ashr	00011	10101	1	000

PAGE NO. // //
DATE // //

2-bit by 2-bit array multiplier.



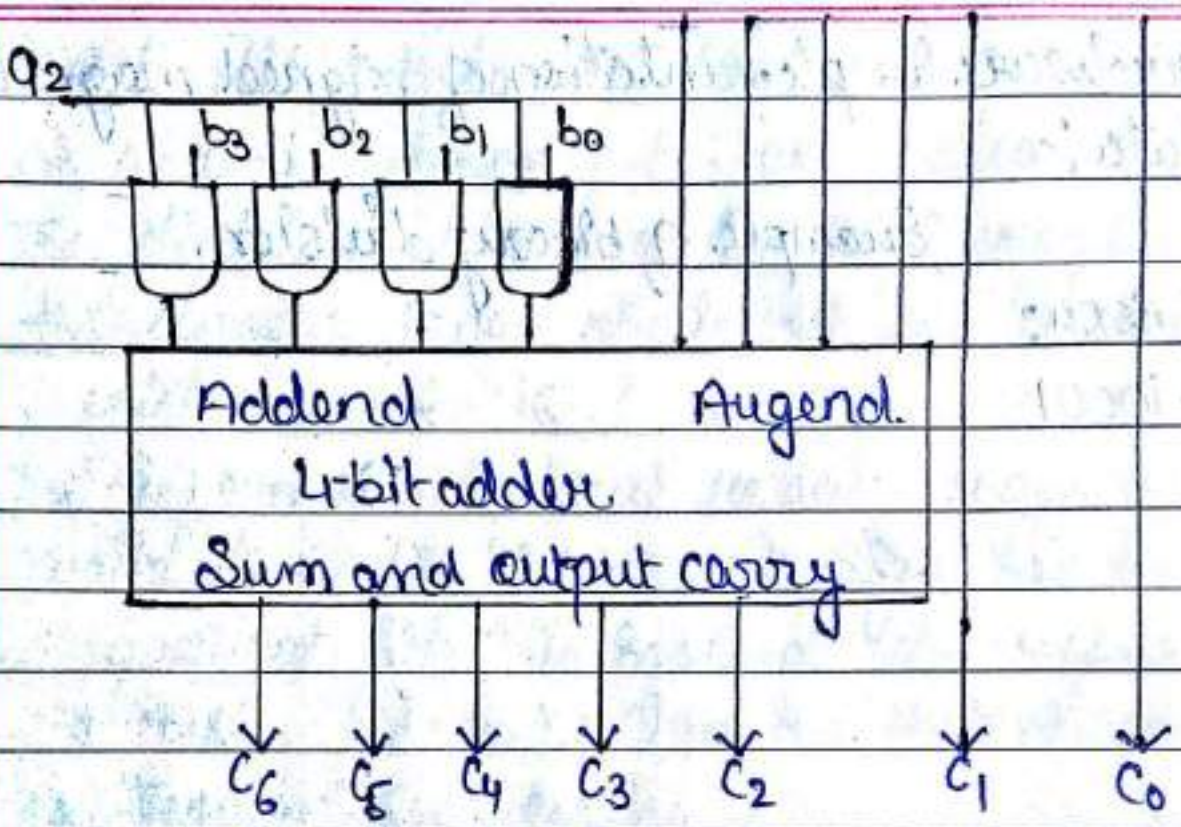
$b_1\ b_0$

$a_1\ a_0$

$a_0b_1\ \ a_0b_0$

$a_1b_1\ \ a_1b_0$ x

$C_3\ \ C_2\ \ C_1\ \ C_0$



★(4 bit by 3-bit array multiplier.)

- Division Algorithm :-

→ Division of two fixed-point binary nos. in signed magnitude representation is done with paper and pencil by a process of successive compare, shift, and subtract operations.

A: Example of binary division with digital hardware. Instead of shifting the divisor to the right, the dividend or partial remainder, is shifted to the left, thus leaving the two nos. in the required relative position. Subtraction may be achieved by adding A to the 2's compliment of B.

EAQ is shifted to the left with 0 instead of Qn and the previous value of E lost.
The divisor is stored in the B register and the double length dividend is stored in register A and Q.
The information about relative magnitude is available in E. If E=1, it signifies that A≥B. A quotient bit 1 is inserted to repeat the process. If E=0, it signifies that A<B so the quotient in Qn remains a 0.
The sign of the remainder is the same as the sign of the dividend.

Divisor B = 10001, B+1 = 01111

	E	A	Qn	SC
Dividend:	0	01110	00000	5
shl EAQ		11100	00000	
add B+1		+01111		
E=1.	1	01011		
Set Qn=1.	1	01011	00001	4
shl EAQ	0	10110	00010	
Add B+1		+01111		
E=1.	1	00101		
Set Qn=1.	1	00101	00011	3

PAGE NO. " "
DATE " "

Shl EAQ	0	01010	00110	
Add $\bar{B}+1$		+01111		
E=0; leave Q_n=0	0	11001		
Add B		+10001		
Restore remainder	1	01010		2
Shl EAQ		10100	01100	
Add $\bar{B}+1$		+01111		
E=1	1	00011		
Set Q_n=1	1	00011	01101	1
Shl EAQ		00110	11010	
Add $\bar{B}+1$		+01111		
E=0; leave Q_n=0	0	10101	11010	
Add B		+10001		
Restore remainder	1	00110	11010	0
Neglect E				
Remainder in A!		00110		
Quotient in Q!			11010	

Example of binary division with digital hardware.

★ Divide Overflow:-

- This occurs because any dividend will be greater than or equal to zero.

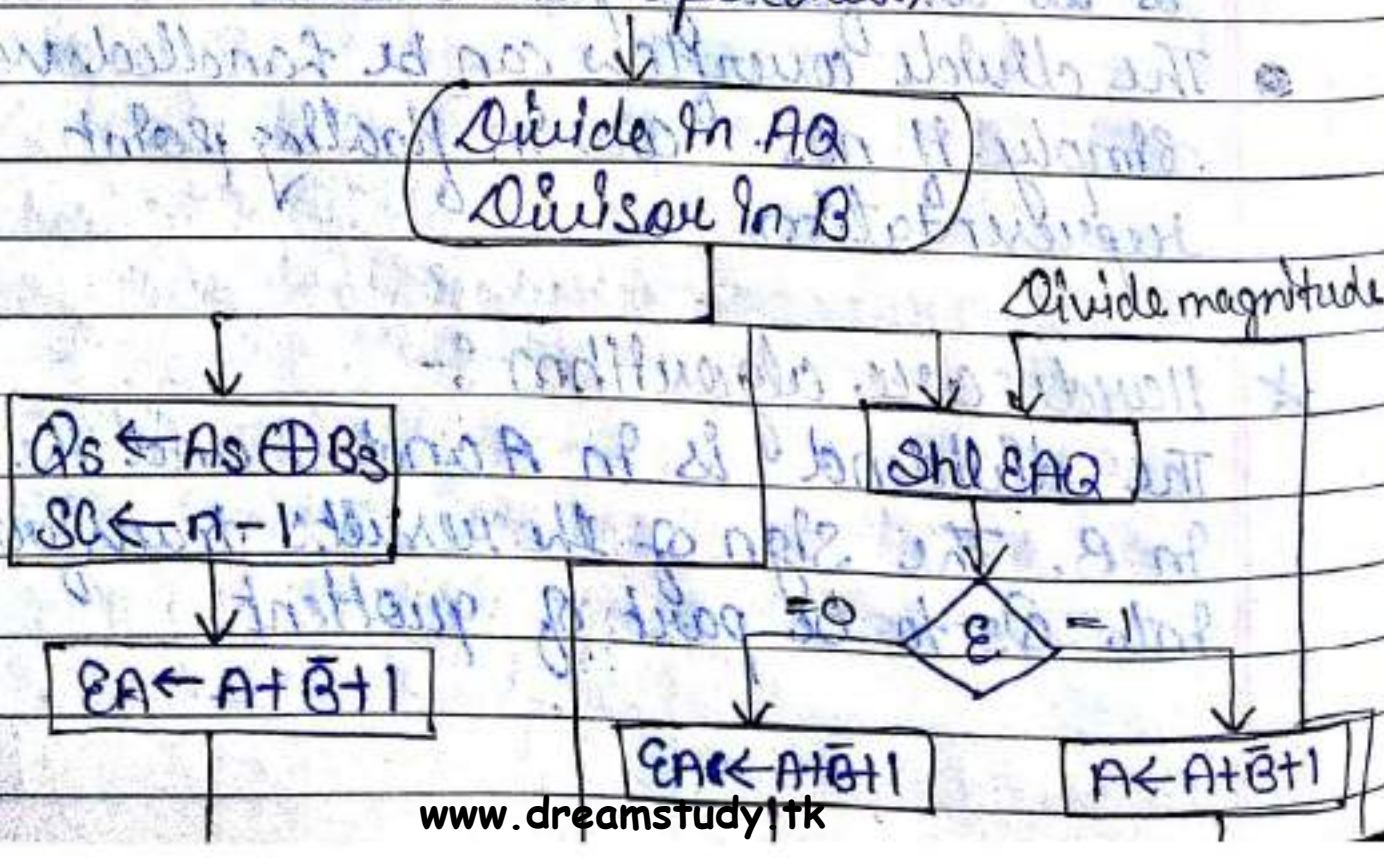
- Overflow condition is usually detected when a special flip-flop is set. Which will call it a divide overflow flip-flop and label it DVF.
- The occurience of a divide overflow can be handled in variety of ways.
- In some computers it is the responsibility of the programmers to check if DVF is set after each divide instruction.
- The occurance of a divide overflow stopped the computer and this condition was reffered to as a DIVIDE Stop.
- The best way to avoid a divide overflow is to use floating point data.
- The divide overflow can be handled very simply if nos. are in floating point representation.

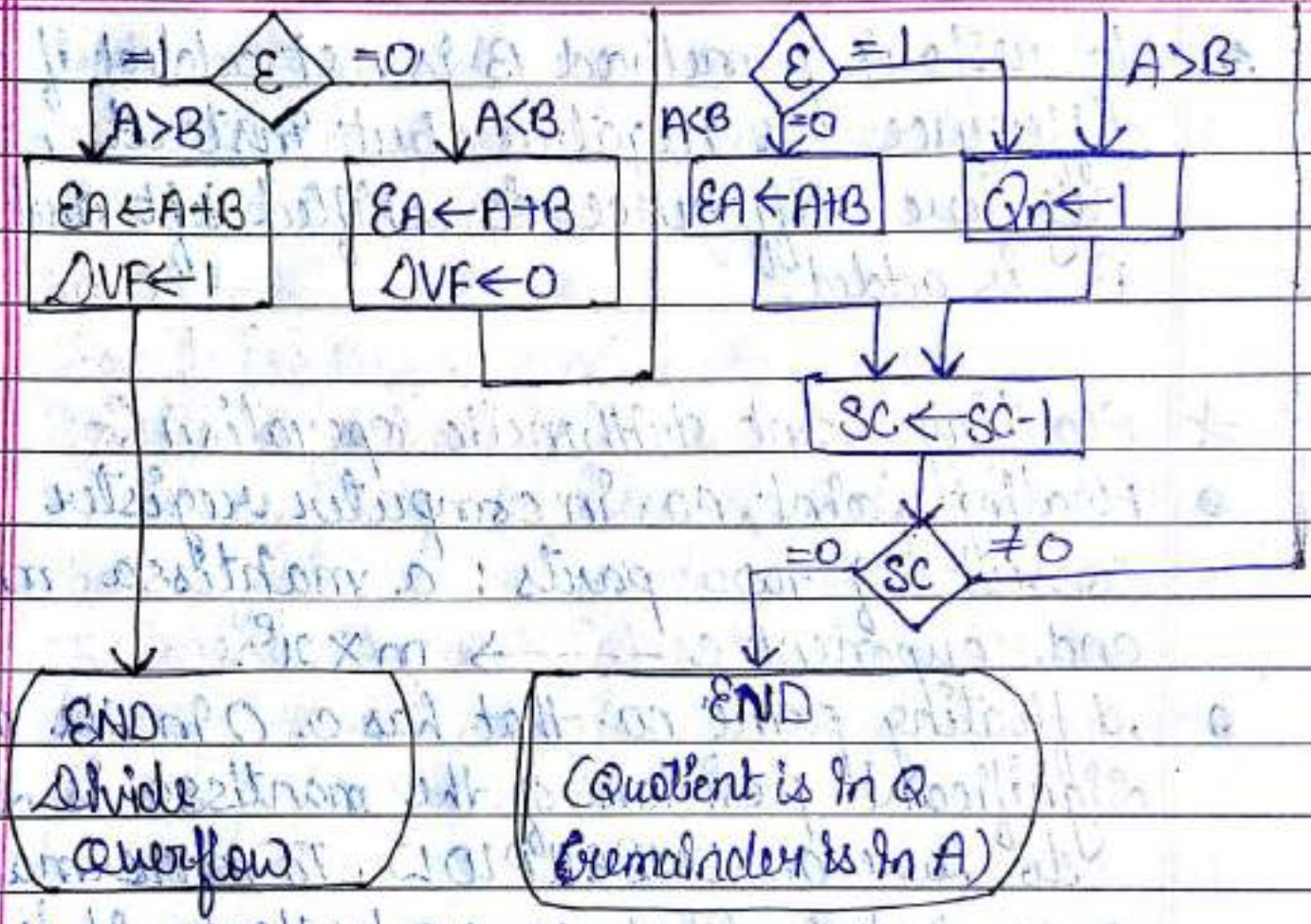
★ Hardware algorithm :-

The dividend is in A and Q and the divisor in B. The sign of the results transferred into Qs to be part of quotient.

A divide overflow condition is tested by subtracting divisor in B from half of the bits of the dividend stored in A. If $A \geq B$, the divide overflow flip-flop DVF set and the operation is terminated prematurely. By doing the process as shown in the flowchart the quotient magnitude is formed in register Q and the remainder is found in the register A. The quotient sign is in Qs and the sign of the remainder in As is the same as the original sign of the dividend.

Flowchart for divide operation





★ What is restoring method?

● The hardware method just described is called the RESTORING METHOD. The reason for the name is that the partial remainder is restored by adding the divisor to the negative difference. Two other methods are available for dividing nos., the COMPARISION method and the NONRESTORING method. In the comparision method A and B are compared prior to the subtraction operation.

PAGE NO. // //
DATE // //

- No restoring method B is not added if the difference is negative but instead, the negative difference is shifted left and then B is added.

★ Floating point arithmetic operation :-

- Floating point no. in computer register consist of two parts : a mantissa m and exponent e ——→ $m \times r^e$.
- A floating point no. that has a 0 in the most significant position of the mantissa is said to have an UNDERFLOW. To normalize a no. that contains an underflow, it is necessary to shift the mantissa to the left and decrement the exponent until a nonzero digit appears in the first position.

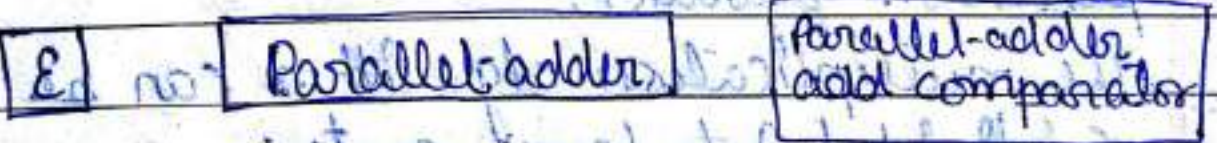
★ Register Configuration :-

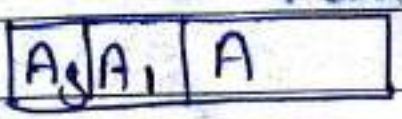
The register configuration for floating point operation is quite similar to the layout for fixed point operation. As a general rule, the same register

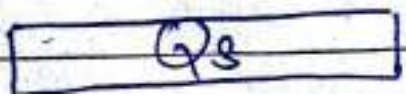
and adder used for fixed point arithmetic are used for processing the mantissa, The difference lies in the way the exponents are handled.

★ Register organization :-
There are three registers, BR, AC, and QR. Each register is subdivided into two parts. The mantissa part has same upper case letters, the exponents part uses corresponding lower case letters.
A parallel adder adds the two mantissas and transfers the sum into A and the carry into E. A separate parallel adder is used for the exponents. Since the exponents are biased.

Register for floating point arithmetic operations :-







PAGE NO. // //
DATE // //

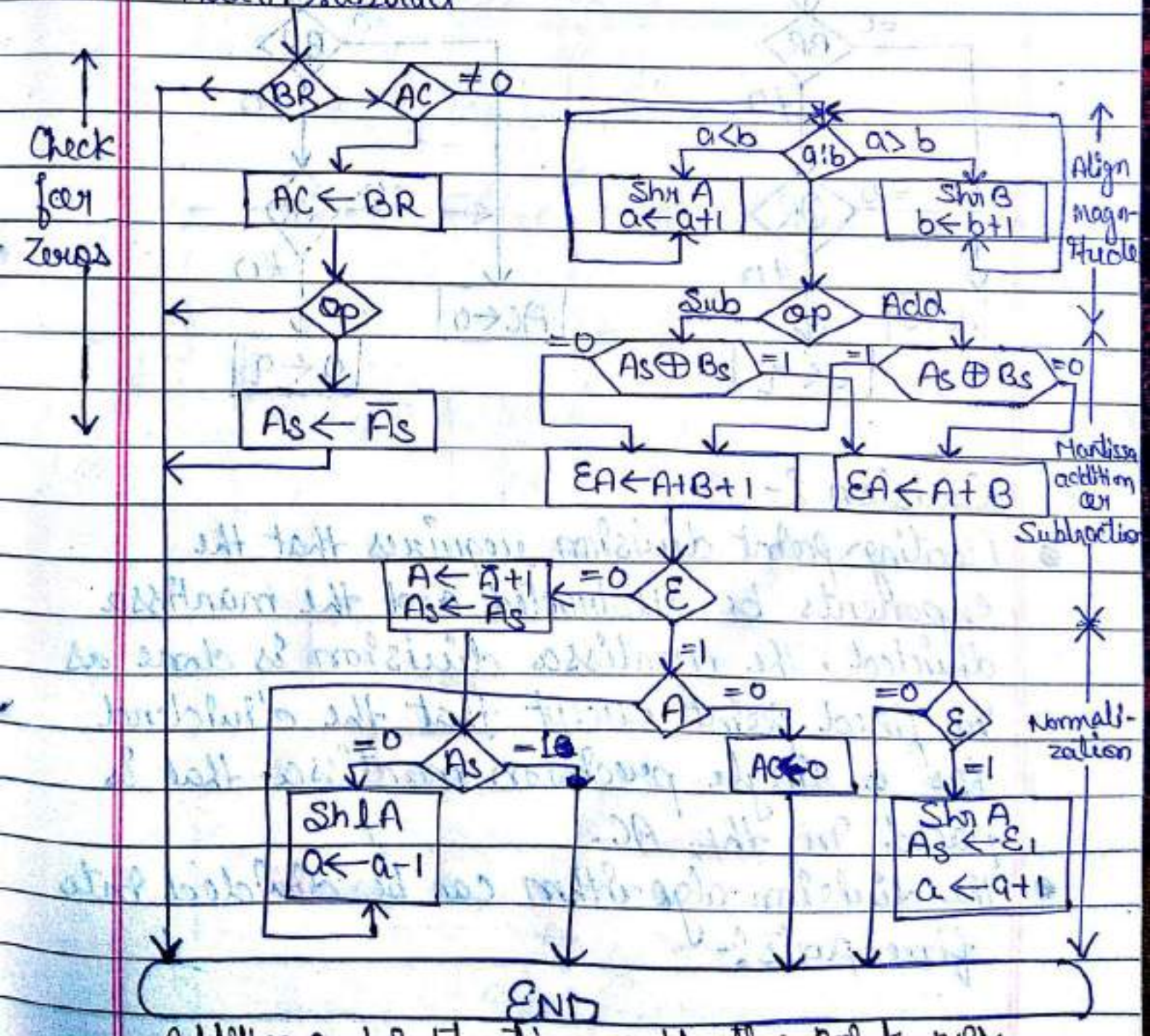
★ Addition and Subtraction :-
During addition and subtraction, the two floating point operands are in AC and BR. The sum of difference is formed in the AC. The algorithm can be subdivided into four consecutive parts :-

1- Check for zeros.
2- Align the mantissa
3- Add or subtract the mantissa
4- Normalize the result.

Multiplication :-
The multiplication of two floating point nos. requires that we multiply the mantissas and add the exponents. No comparison of exponents or alignment of mantissa is necessary.
The multiplication of the mantissa is performed same as fixed point to provide a double precision product.
The multiplication algorithm can be subdivided into four parts :-

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

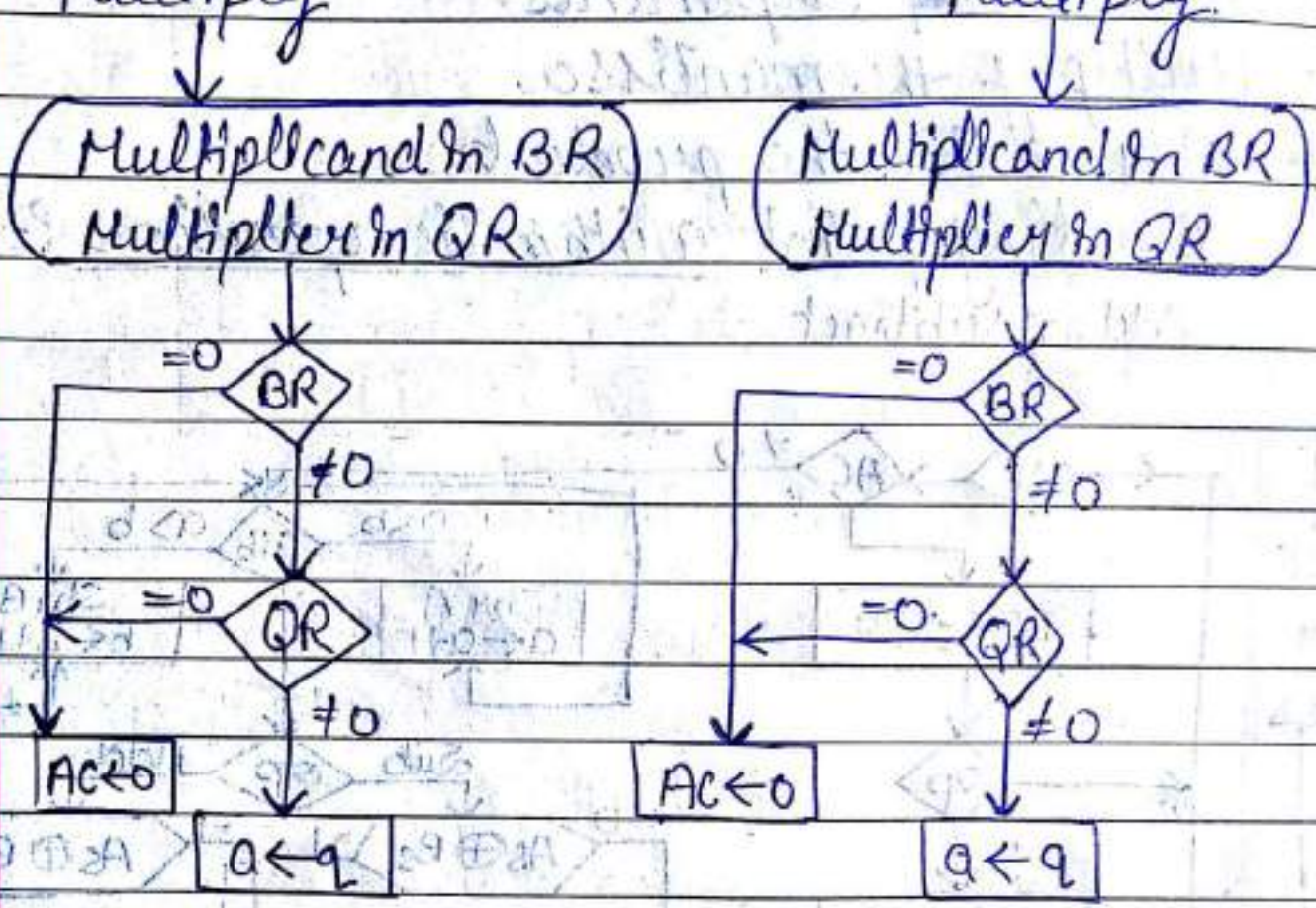
Floating-point arithmetic operations :-



Addition and subtraction of floating point numbers.

PAGE NO. // //
DATE // //

Multiplication of floating point nos.

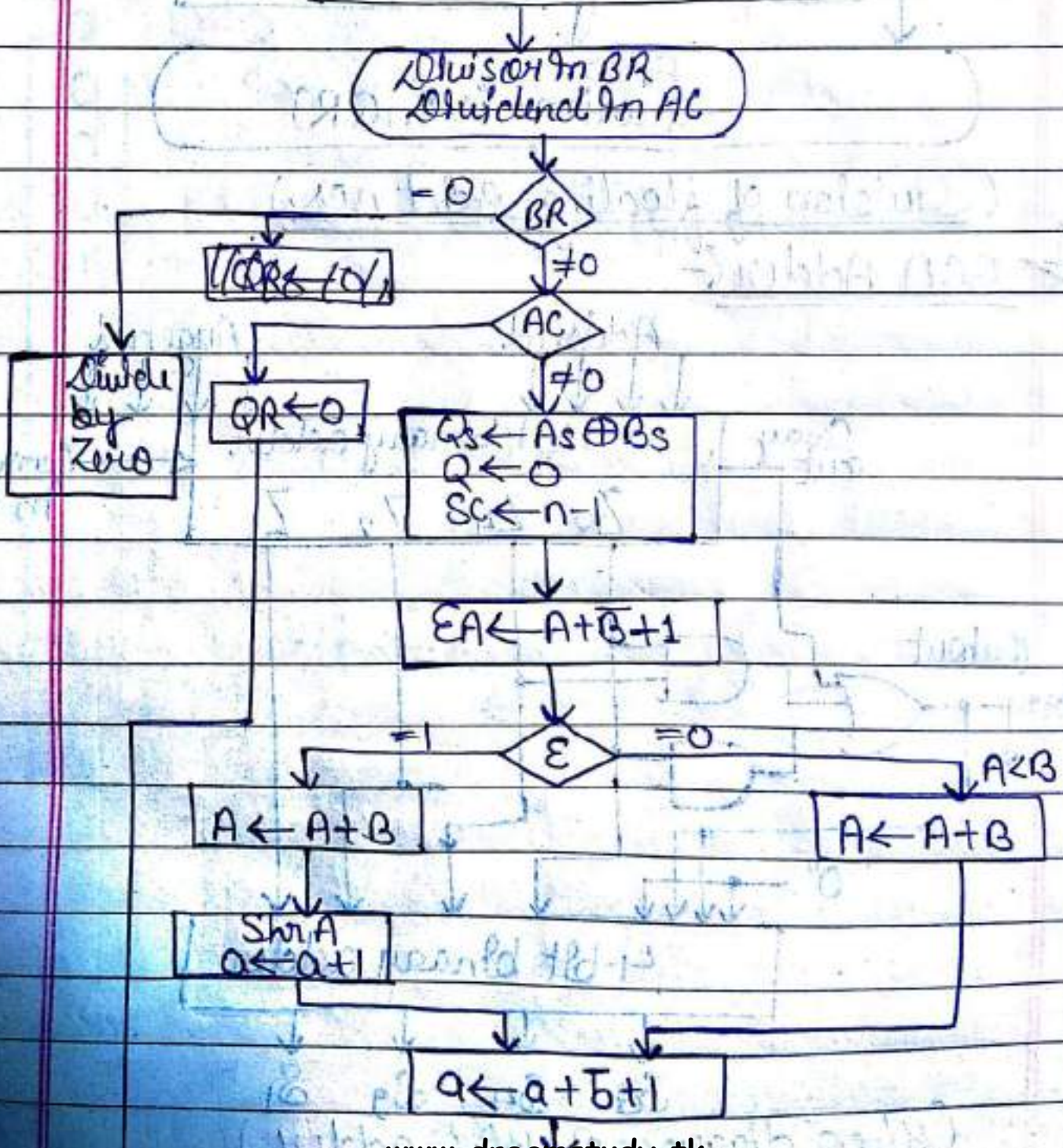


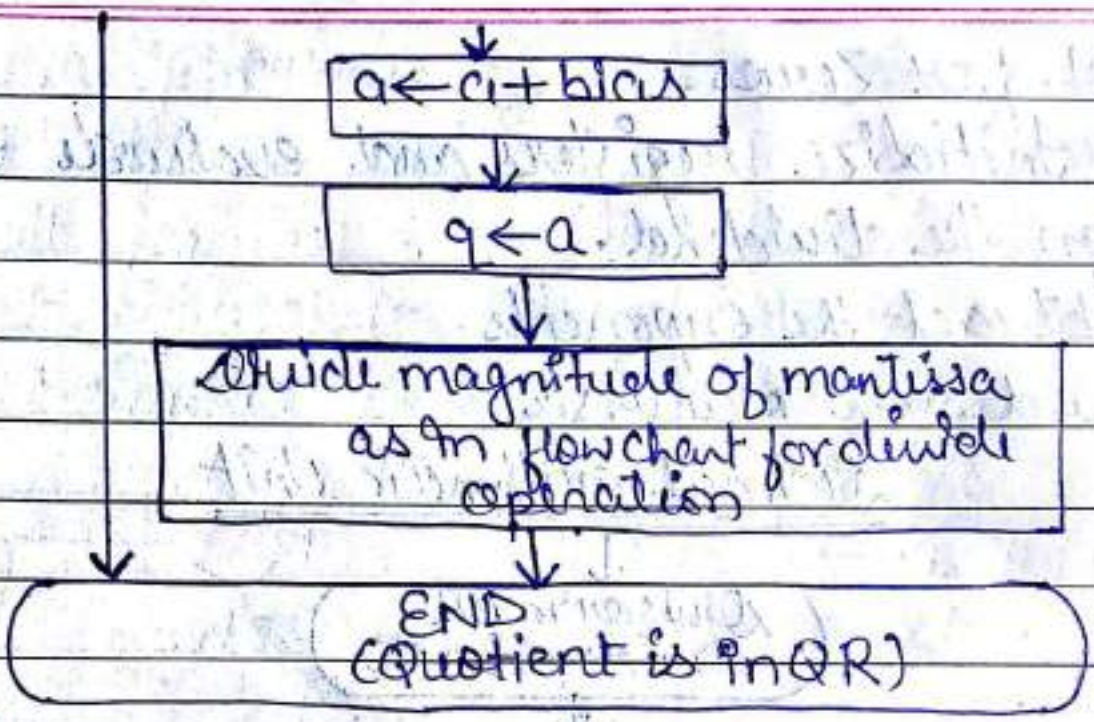
★ Division :-

- Floating point division requires that the exponents be subtracted and the mantissa divided. The mantissa division is done as in fixed point except that the dividend has a single precision mantissa that is placed in the AC.
- The division algorithm can be divided into five parts :-

1- Check for Zeros.
2- Initialize registers and evaluate the sign.
3- Align the dividend.
4- Subtract the exponents.
5- Divide the mantissa.

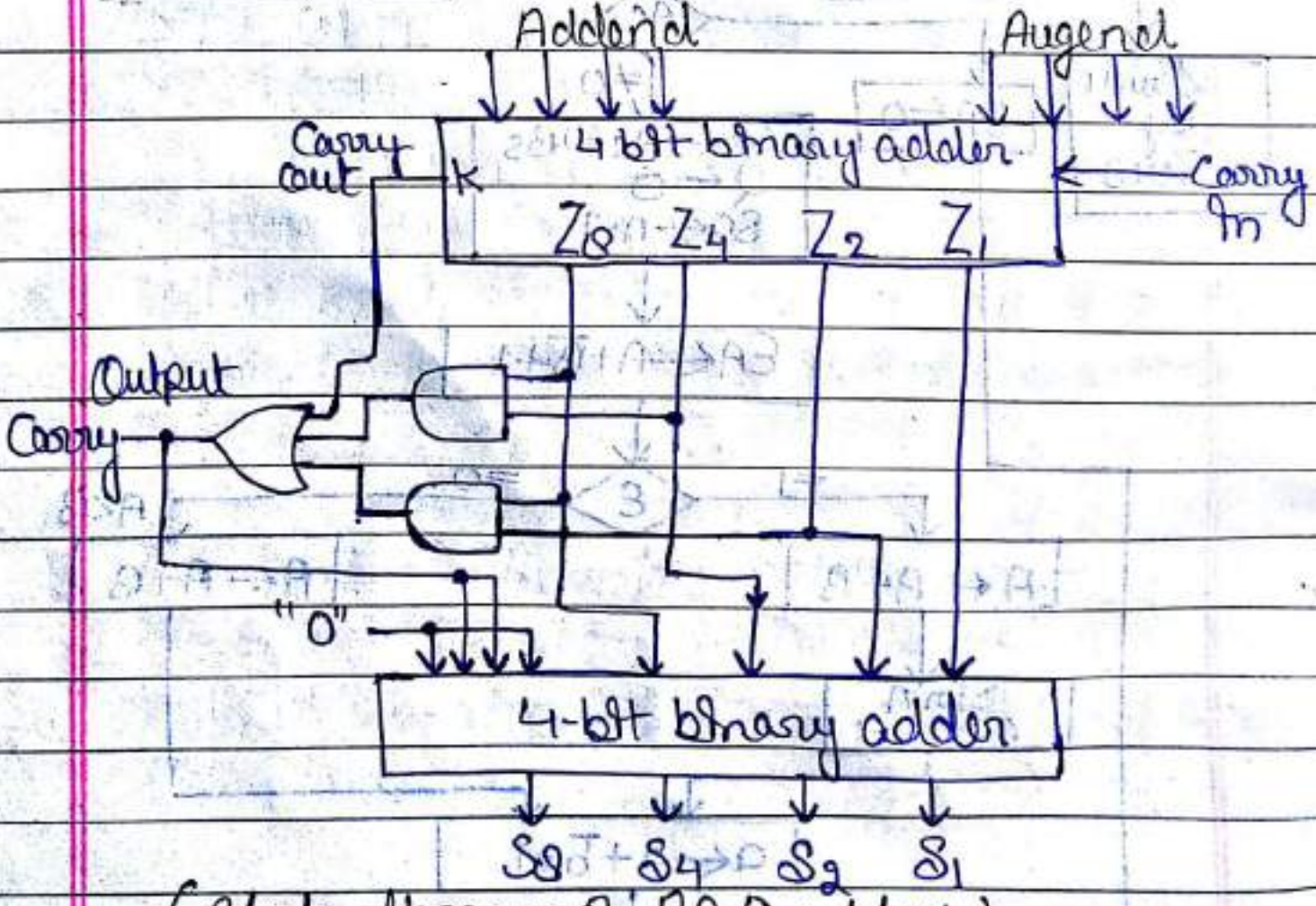
Decimal arithmetic Unit





(Division of floating-point nos.)

★ BCD Adder :-



(Block diagram of BCD adder)

PAGE NO. // //
DATE // //

Table :- Derivation of BCD Adder.

K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	Decimal.
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19
	Binary Sum				BCD Sum					

PAGE NO. // //
DATE // //

★ BCD Subtraction :-

1. ✓ A straight subtraction of two decimal nos. will require a subtraction circuit that will be somewhat different from a BCD adder.
2. ✓ It is more economical to perform the subtraction by taking 9's or 10's complement of the subtrahend and adding it to the minuend. Since the BCD is not a self-complementing code.
3. ✓ The Boolean functions for the 9's complement circuit are-

$X1 = B1N' + B1'N$

$X2 = B2$

$X4 = B4N' + (B4'B2 + B4B2')N$

$X8 = B8N' + B8'B4'B2'N$

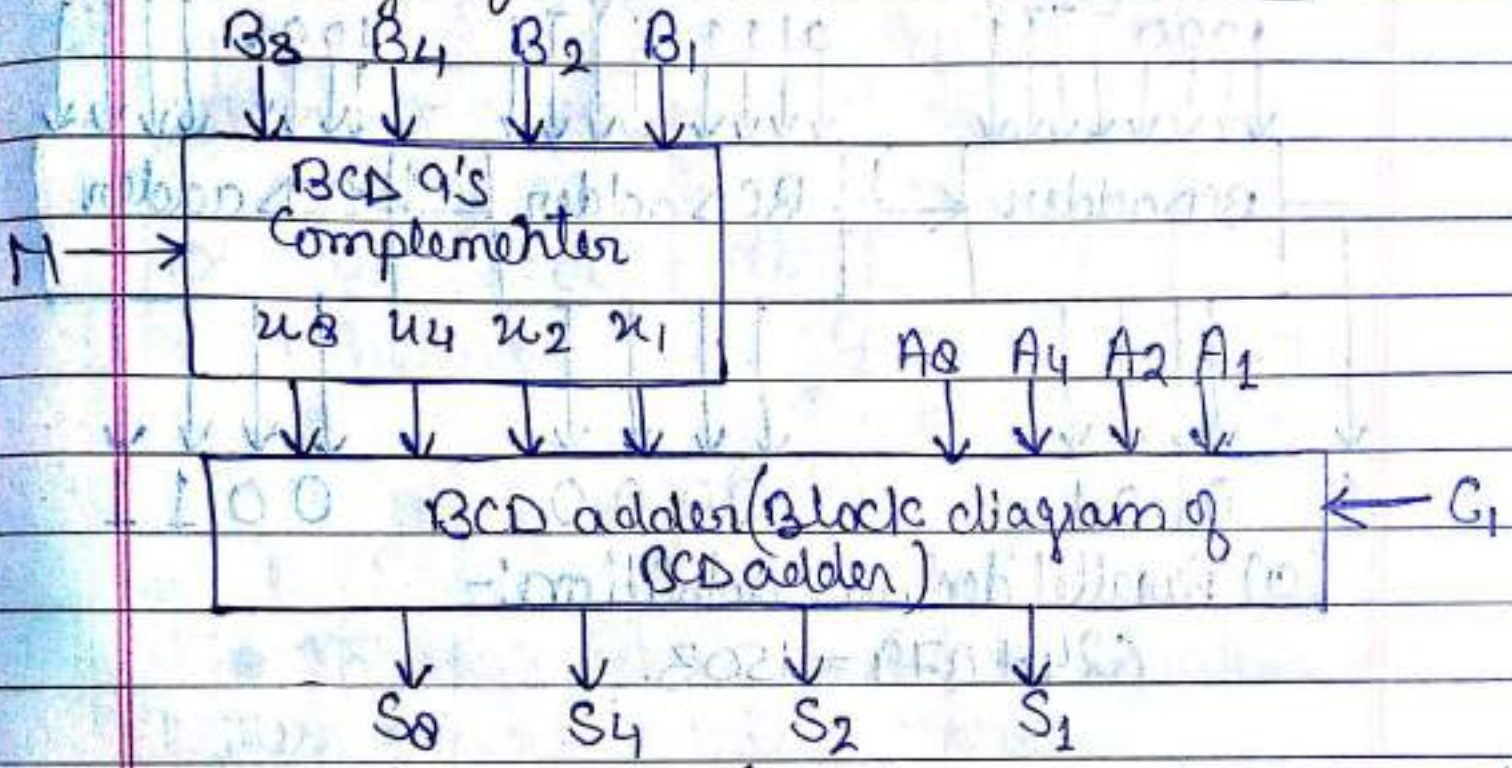
From these eq^n's we see that X=B when N=0, when N=1, the X output produce the 9's complement of B.

★ Decimal Arithmetic operation :-

- The algorithm for arithmetic operations with decimal data are similar to the algorithms for the corresponding operations with binary data.

PAGE NO. // //
DATE // //

• One stage of a decimal arithmetic unit.



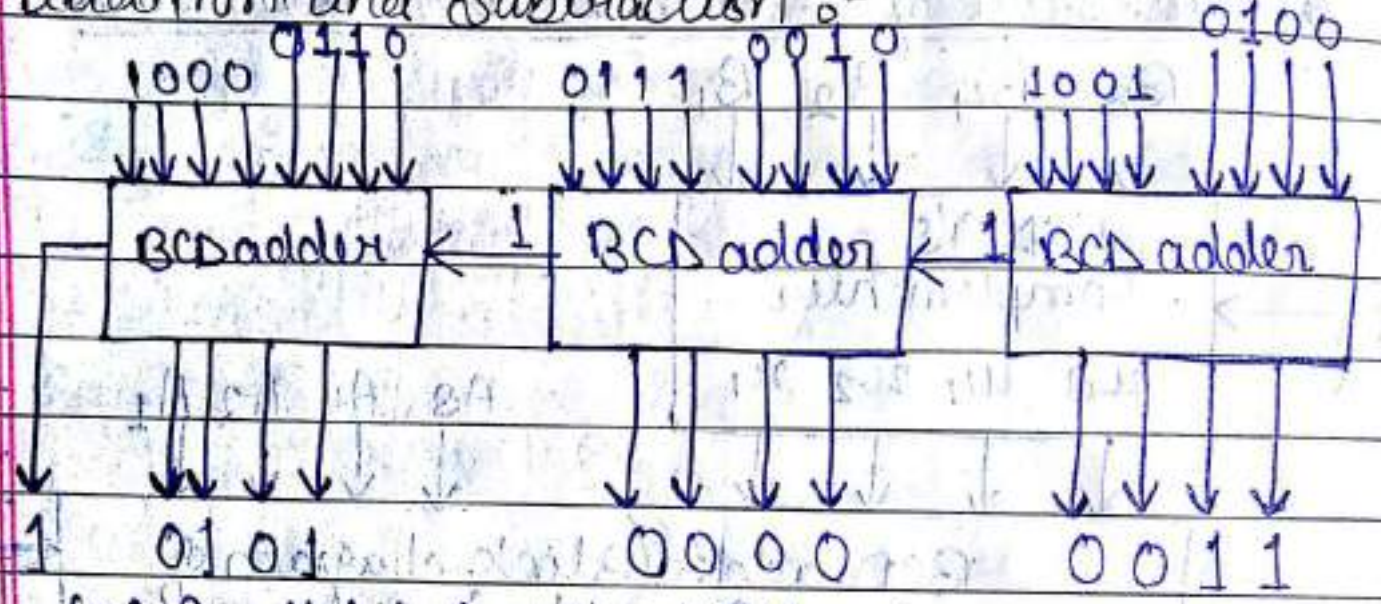
★ Decimal arithmetic micro operation symbols

Table:- Decimal arithmetic Microoperation Symbols

Symbolic designation	Description.
$A \leftarrow A+B$	Add decimal nos. and transfer sum into A
$\bar{B}$	9's complement of B.
$A \leftarrow A+\bar{B}+1$	Content of A plus 10's complement of B into A.
$Q_1 \leftarrow Q+1$	Increment BCD no. in Q_1
dshr A	Decimal shift-right register A
dshl A	Decimal shift-Left register A.

PAGE NO. // //
DATE // //

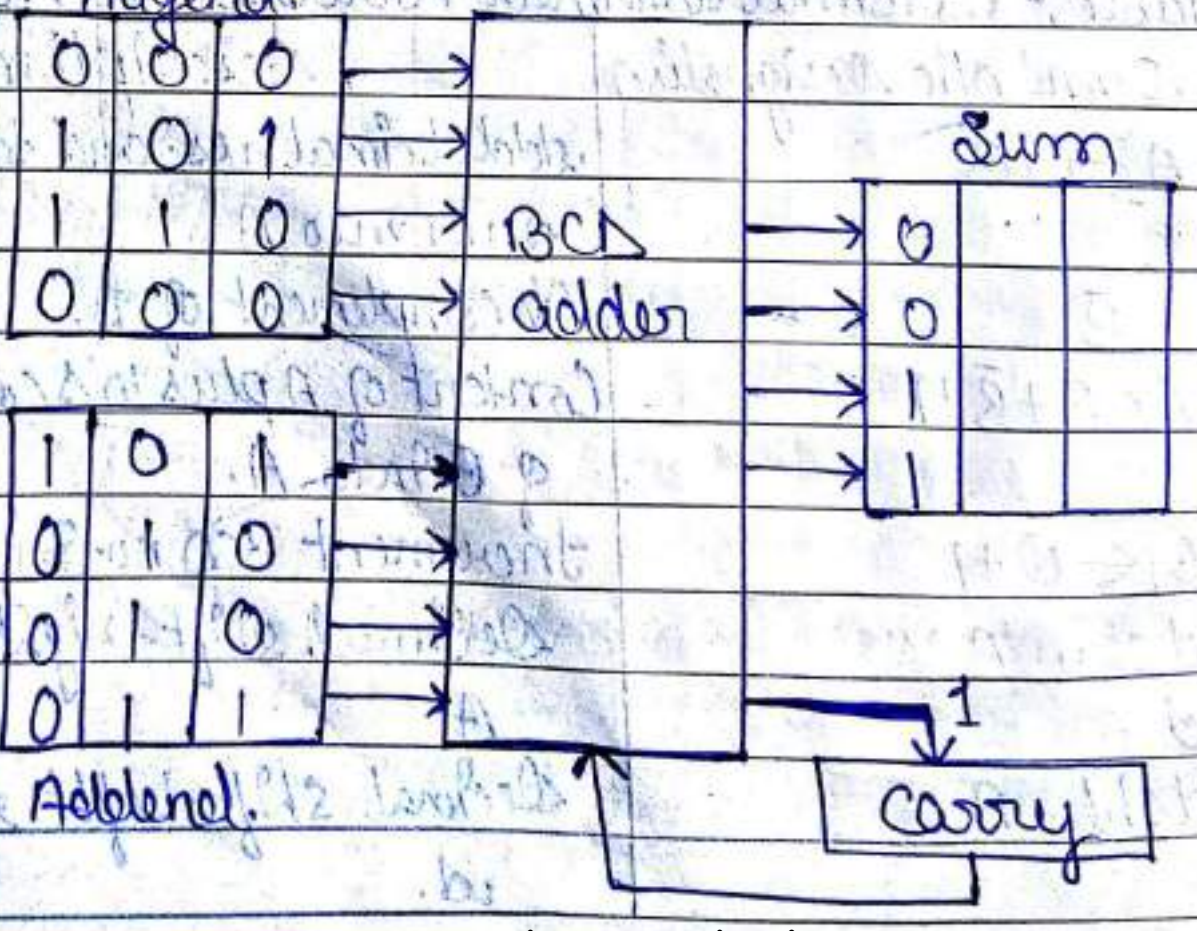
Addition and subtraction :-



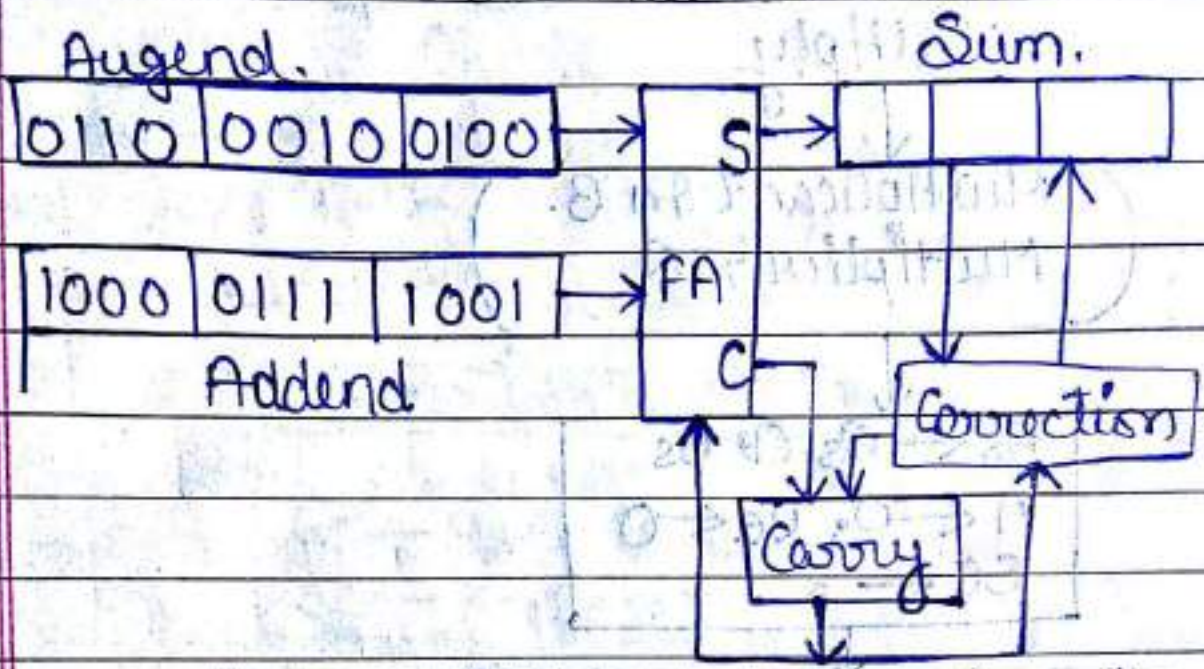
(a) Parallel decimal addition :-

624 + 879 = 1503.

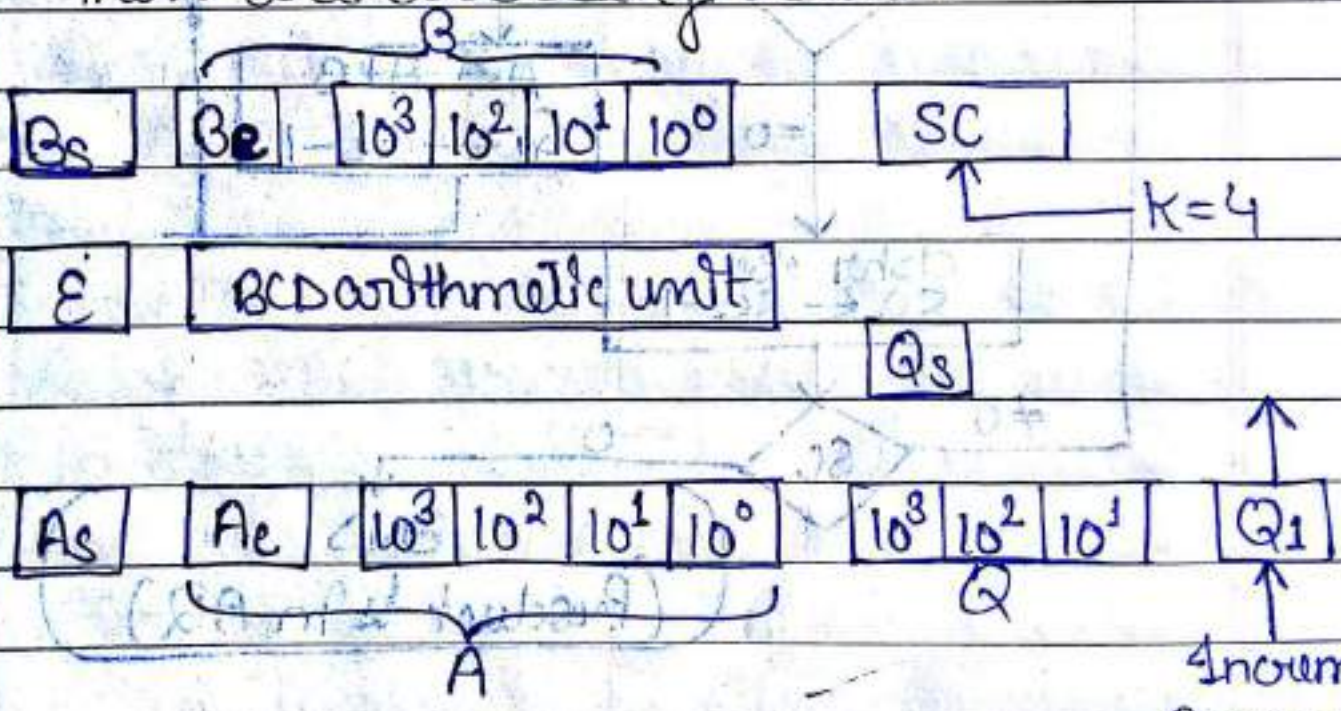
(b) Digit-serial, bit parallel decimal addition



(c) All serial decimal addition.

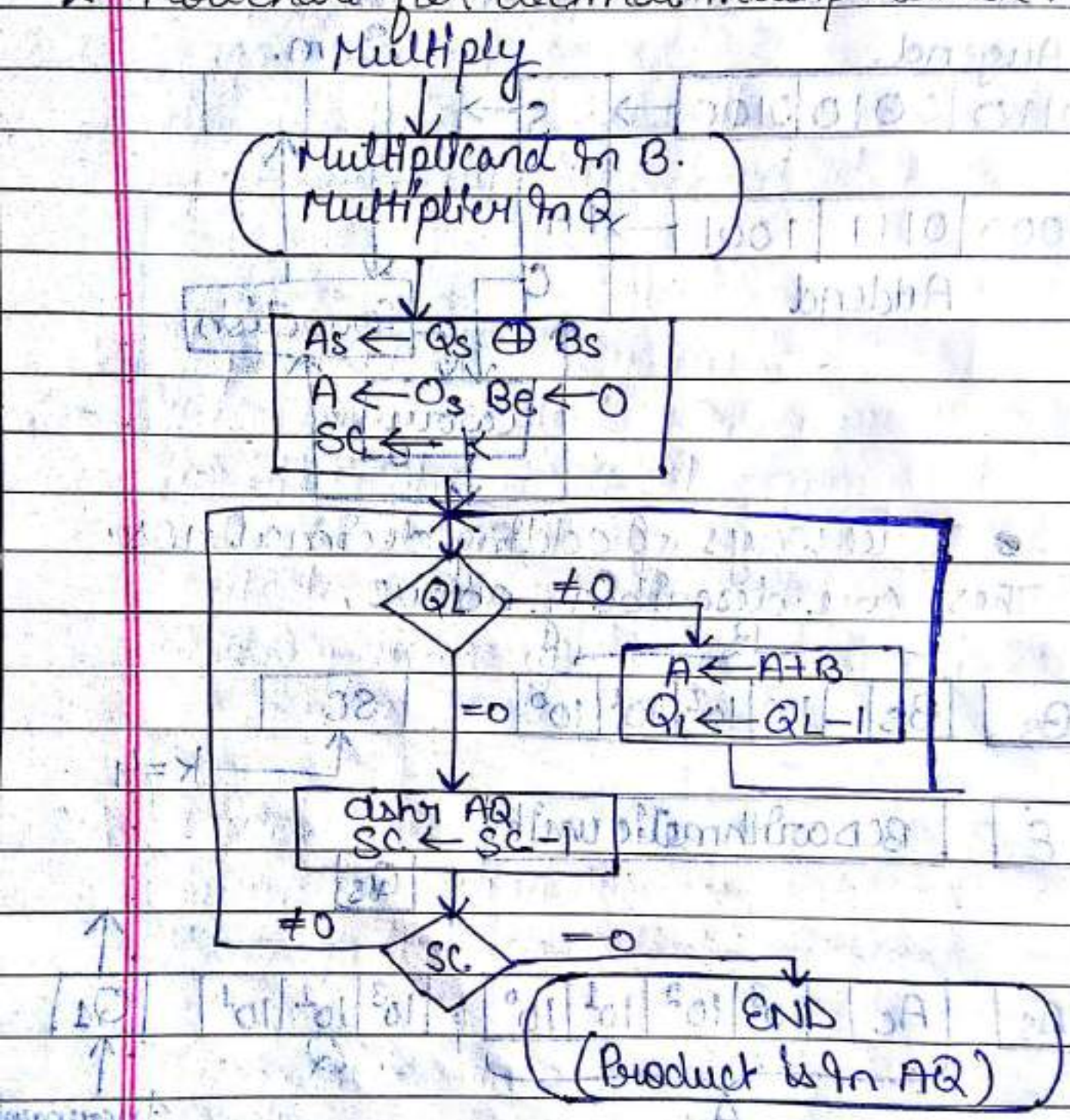


- Three ways of adding decimal nos. There are describing above.



☆ Registers for decimal arithmetic multiplication and division.

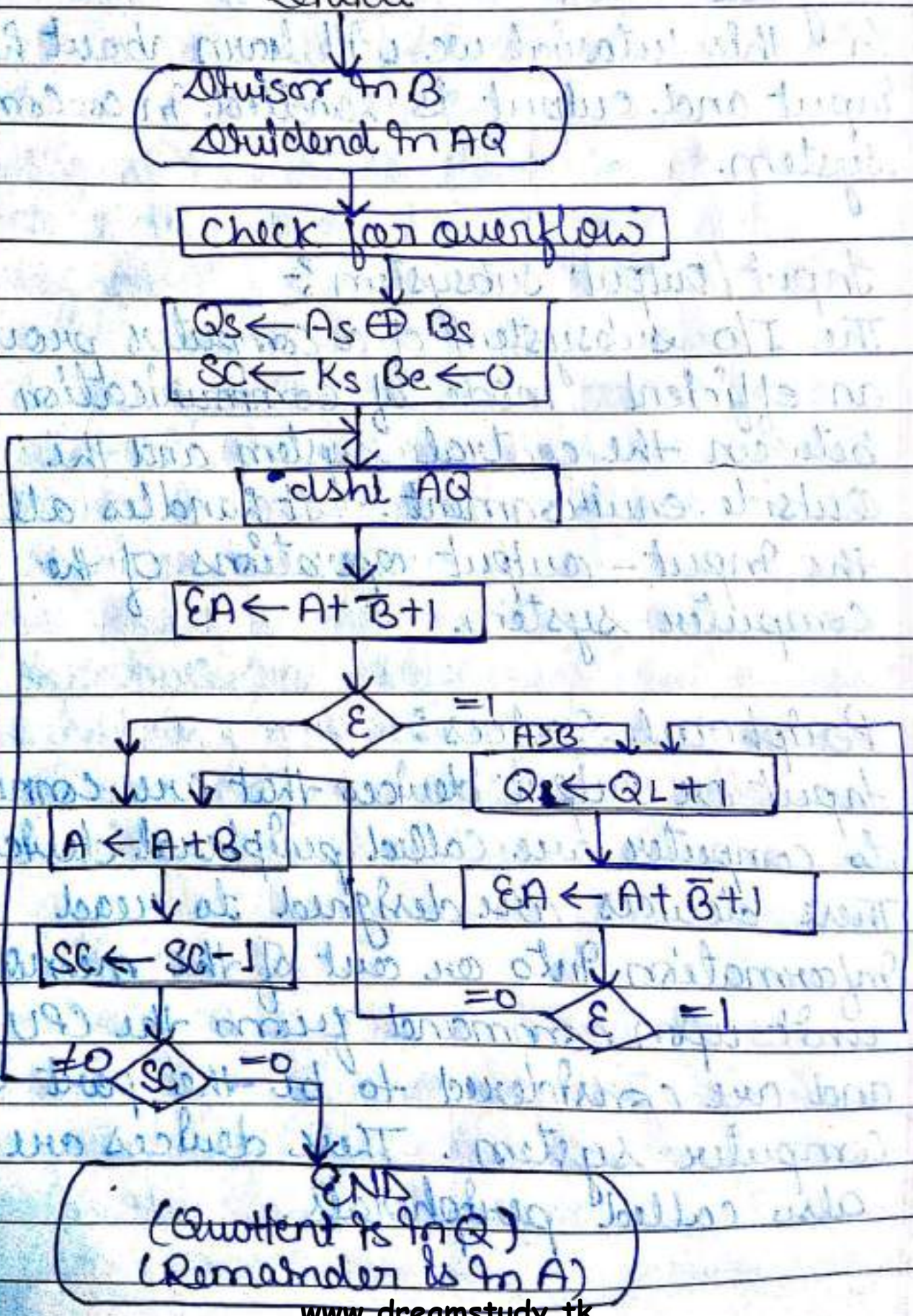
★ Flowchart for decimal multiplication.



★ Flowchart for decimal division :-

PAGE NO. // //
DATE // //

Division Algorithm



UNIT-4

PAGE NO. // //
DATE // //

★ Computer Architecture: Input/output Organisation:-
In this tutorial we will learn about how Input and output is handled in a computer system.

★ Input/output subsystem:-
The I/O subsystem of a computer provides an efficient mode of communication between the central system and the outside environment. It handles all the input-output operations of the computer system.

★ Peripheral Devices:-
Input or output devices that are connected to computer are called peripheral devices. These devices are designed to read information into or out of the memory unit upon command from the CPU and are considered to be the part of computer system. These devices are also called peripherals.

PAGE NO. // //
DATE // //

for example: Keyboards, display units and printers are common peripheral devices.
There are three types of peripherals:

1- Input peripherals :-
Allows ~~information~~ user input, from the outside world to the computer. Example: Keyboard, Mouse etc.

2- Output peripherals :- Allows information output, from the computer to the outside world. Example: Printer, Monitor etc.

3- Input-Output peripherals :- Allows both input (from ~~outside~~ outside world to Computer) as well as, output (from computer to the outside world). Example: Touch Screen etc.

★ Interfaces :-
Interfaces is a shared boundary between two separate components of the computer system which can be used to attach two or more components to the system for communication purpose.

There are two types of Interface :-

1- CPU Interface.
2- I/O Interface.

Input - Output Interface :-
Peripherals connected to a computer need special communication links for interfacing with CPU. In computer system, there are special hardware components b/w the CPU and peripherals to control or manage the Input-output transfers. These components are called Input-output Interface units because they provide communication links b/w processor bus and peripherals. They provide a method for transferring information b/w internal system and Input-output devices.

★. Modes of I/O Data Transfer :-
Data transfer between the central unit and I/O devices can be handled in generally three types of modes which are given below :-

PAGE NO. // //
DATE // //

1- Programmed I/O
2- Interrupt Initialized I/O
3- Direct Memory Access.

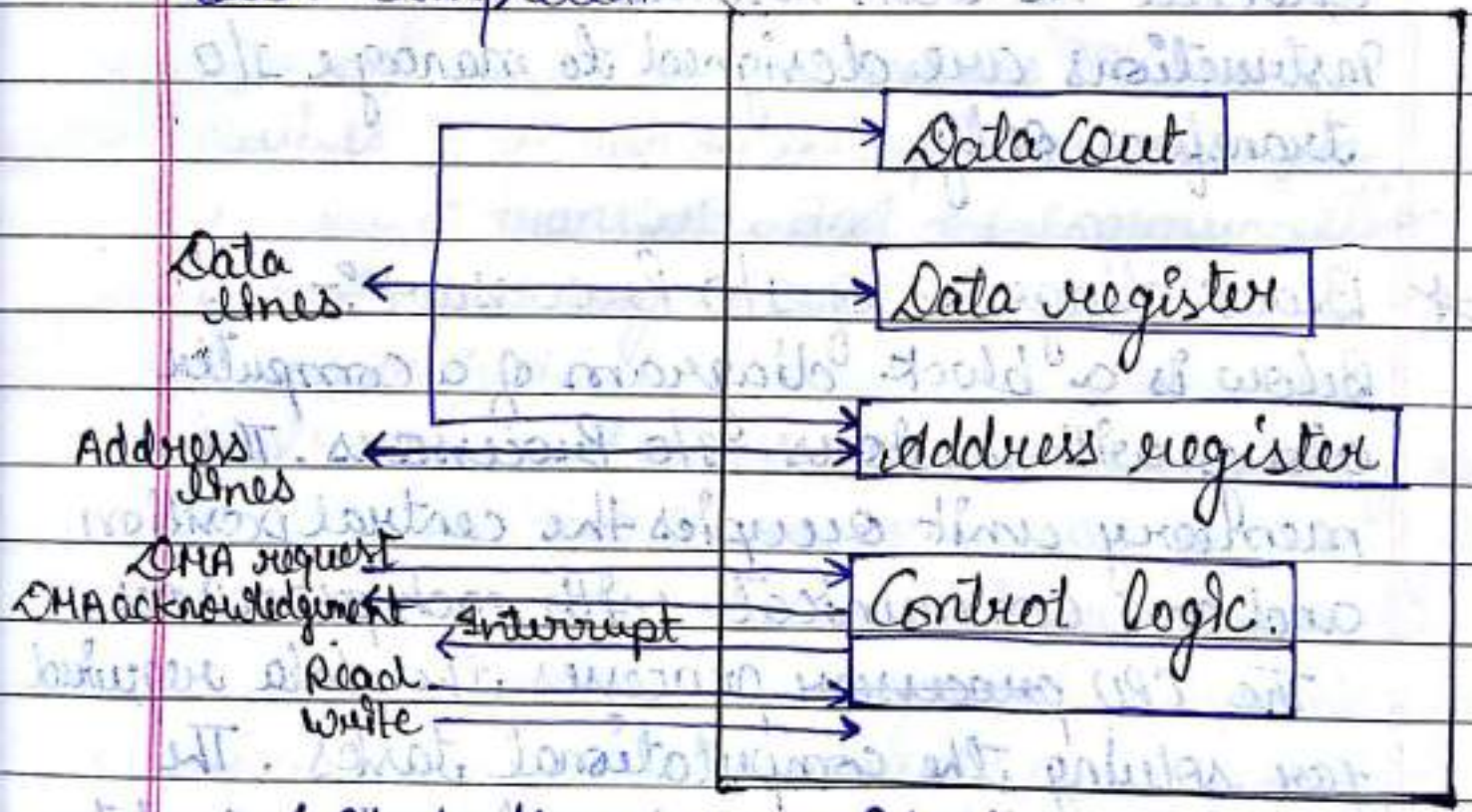
1★ Programmed I/O :-
Programmed I/O instructions are the result of I/O instructions written in computer program. Each data item transfer is initiated by the instruction in the program.
Usually the program controls data transfer to and from CPU and peripheral. Transferring data under programmed I/O requires constant monitoring of the peripherals by the CPU.

2- Interrupt Initiated I/O :-
In the programmed I/O method the CPU stays in the program loop until the I/O unit indicates that it is ready for data transfer. This is time consuming processor because it keeps the processor busy needlessly.
This problem can be overcome by using interrupt initiated I/O.

In this when the interface determines that the peripheral is ready for data transfer, it generates an interrupt. After receiving the interrupt signal, the CPU stops the task which it is processing and service the I/O transfer and then returns back to its previous processing task.

3- Direct Memory Access :-
Removing the CPU from the path and letting the peripheral device manage the memory buses directly would improve the speed of transfer. This technique is known as DMA.
In this, the interface transfer data to and from the memory through memory bus. A DMA controller manages to transfer data b/w peripherals and memory unit.
Many hardware system use DMA, such as disk drive controllers, graphic cards, network cards and sound

Cards etc. It is also used for Intra Chip. data transfer in multicore processors. In DMA, CPU would initiate the transfer, do other operations while the transfer is in progress and receive an interrupt from the DMA controller when the transfer has been completed.



(Block diagram of DMA)

★ Computer Architecture :-

Input/output Processor :-

An Input-output processor (IOP) is a processor with direct memory access capability.

PAGE NO. // //
DATE // //

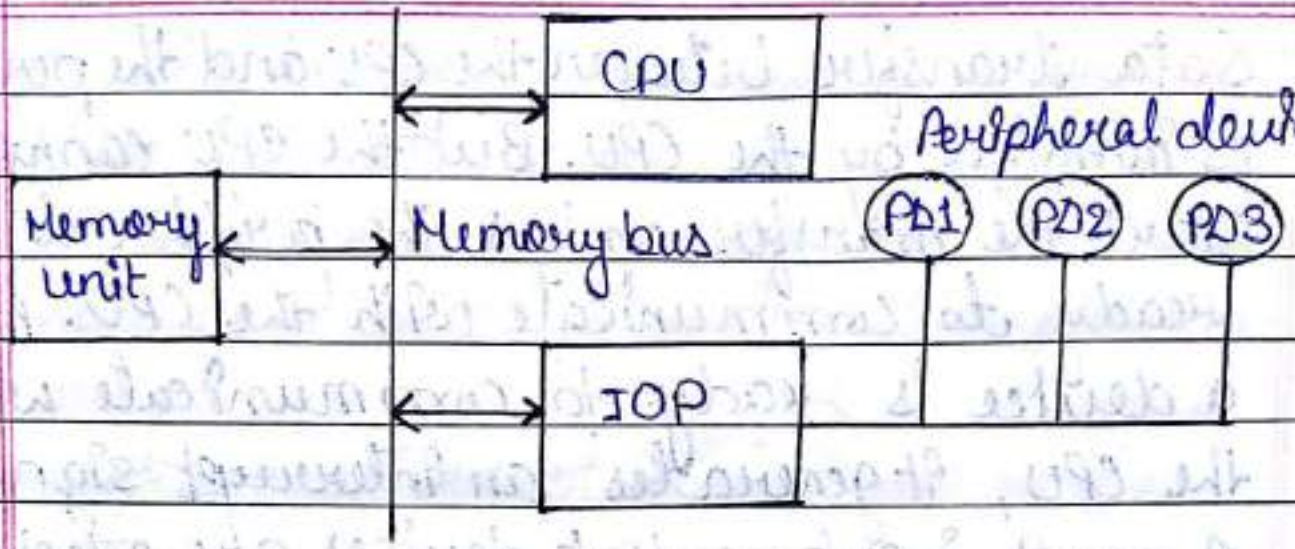
In this, the computer system is devided into a memory unit and no. of processor. Each IOP controls and manage the Input-output tasks. The IOP is similar to CPU except the it handles only the details of I/O processing. The IOP can fetch and execute its own instructions. These IOP instructions are designed to manage I/O transfers only.

★ Block diagram of I/O Processor:-
Below is a block diagram of a computer along with various I/O processors. The memory unit occupies the central position and can communicate with each processor.

The CPU ~~processor~~ processes the data required for solving the computational tasks. The IOP provides a path for transfer of data between peripherals and memory. The CPU assigns the tasks of initial the I/O program.

~~The Input~~

The IOP operates independent from CPU and transfer data between peripherals and memory.



The communication between the IOP and the devices is similar to that program control method of transfer. And the communication with the memory is similar to the direct memory access method.

Instructions that are read from memory by an IOP are also called commands to distinguish them from instructions that are read by CPU. Commands are prepared by programmers and are stored in memory. Command words make the program for IOP. CPU informs the IOP where to find the commands in memory.

A. Compiler & Architecture ? Interrupts ?-

PAGE NO. // //
DATE // //

Data transfer between the CPU and the peripherals is initiated by the CPU. But the CPU cannot start the transfer unless the peripheral is ready to communicate with the CPU. When a device is ready to communicate with the CPU, it generates an interrupt signal. A no. of input-output devices are attached to the computer and each device is able to generate an interrupt request. The main job of the interrupt system is to identify the source of the interrupt.

- Priority Interrupt :-
A priority interrupt is a system which decides the priority at which various devices, which generates the interrupt signal at the same time, will be serviced by the CPU. The system has authority to decide which conditions are allowed to interrupt the CPU, while some other interrupt is being serviced. Generally, devices with high speed transfer such as magnetic disks are given high priority and slow devices

PAGE NO. // //
DATE // //

Such as keyboards are given low priority. When two or more devices interrupt the computer simultaneously, the computer services devices with the high priority first.

* Types of Interrupt :-

* Serial Data Communication :-

• Data communication processor :-

A data communication processor is an I/O processor that distributes and collects data from numerous remote terminals connected through telephone and other communication lines to the computer. It is a specialized I/O processor designed to communicate with data communication networks.

Such a communication network consists of variety of devices such as printers, display devices, digital sensors etc. serving many users at once. The data communication with CPU and memory in the same manner as any I/O processor does.

PAGE NO. // //
DATE // //

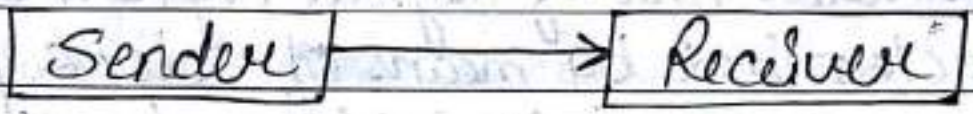
- What is Modem?

In a data Communication Network, the remote terminals are connected to the data comm^{n} processor through telephone lines or other wires. Such telephone lines are specially designed for voice communicat^{n} and computers use them to communicate in digital signals, therefore some conversion is required. These conversions are called modem (modulator-demodulator).
A modem converts digital signal into audio tones to be transmitted over telephone lines and also converts audio tones into digital signal for machine use.

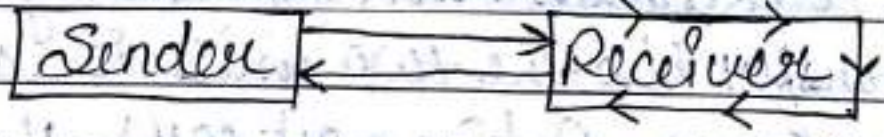
★ Modes of Transmission:-
Data can be transmitted b/w 2 points by three different modes:

1- Simplex:-
A simplex line carries information in one direction only. In this mode receiver cannot communicate with the sender

to Indicate the occurance of errors that means only sender can send data but receivere cannot. For enample: Radio and Television Broadcasting.

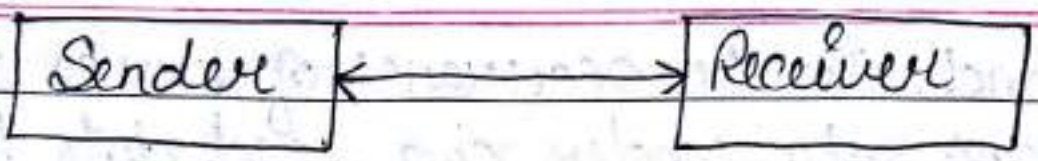


2- Half Duplex:-
In half duplex mode, system is capable of transmitting data in both directions but data can be transmitted in one direction only at a time. A pair of wires is needed for this mode. For Enample: Walkie-Talkie.



3- Full Duplex:-
In this mode data can be send and received in both directions simultaneously. In this four wire link is used. For enample: Video calling, Audio calling etc.

PAGE NO. // //
DATE // //



★ Asynchronous Data Transfer

We know, the Internal operations in Individual unit of digital system are synchronized by means of clock pulse, means clock pulse is given to all registers within a unit, and all data transfer among internal registers occurs simultaneously during occurrence of clock pulse. Now, Suppose any two units of digital system are designed ~~to~~ independently such as CPU I/O Interface.

And if the registers in the Interface (I/O Interface) share a common clock with CPU registers, then transfer b/w the two units is said to be Synchronous. But in most cases, the Internal timing in each unit is independent from each other in such a way that each uses its own private clock for its Internal registers. In that case, the two units are said to be asynchronous to each other, and if data transfer occurs b/w them this data transfer is said to be

PAGE NO. // //
DATE // //

Asynchronous data Transfer.

This asynchronous way of data transfer can be achieved by two methods:-

1- One way is by means of strobe pulse which is supplied by one of the units of other unit. When transfer has to occure. This method is known as "Strobe Control."

2- Another method commonly used is to accompany each data item being transferred with a control signal that indicates the presence of data in the bus. The unit receiving the data item responds with another signals to acknowledge receipt of the data. This method of data transfer btw two independent units is said to be "Handshaking".

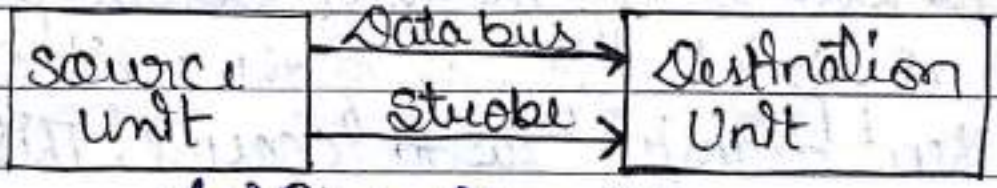
1- Strobe Control :-

The strobe control method of asynchronous data transfer employs a single control lines to time each transfer. The control line is also known as strobe and it may be achieved either by source or destination, depending on which initiate transfer.

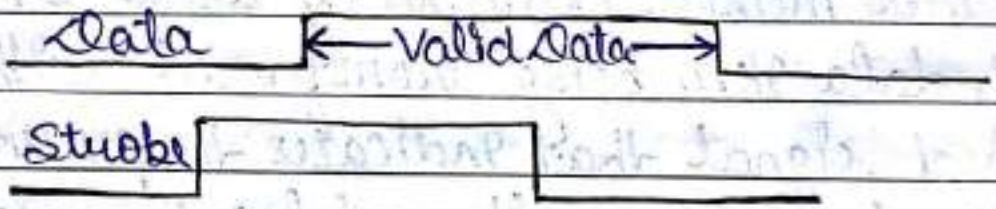
PAGE NO. // //
DATE // //

- Source Initiated strobe for data transfer.

The block diagram and timing diagram of strobe initiated by source unit is shown in figure below:



(a) Block Diagram.

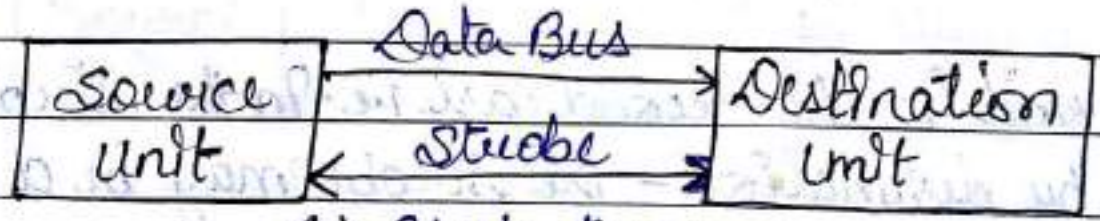


(b) Timing Diagram.

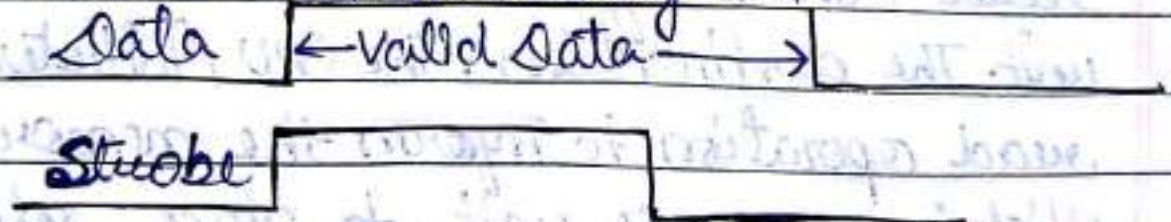
In block diagram we see that strobe is initiated by source, and as shown in timing diagram, the source unit first places the data on the data bus. After a brief delay to ensure that the data settle to a steady value, the source activates a strobe pulse. The information on data bus and strobe control signal remain in the active state for a sufficient period of time to allow.

destination unit to receive the data. Actually, the destination unit, uses a falling edge of strobe control to transfer the contents of data bus to one of its ~~initial~~ internal registers. The source removes the data from the data bus after it disable its strobe ~~unit~~ ~~pluse~~ pulse. New valid data will be available only after the strobe is enabled again.

- Destination-Initiated strobe for data transfer. The block diagram and timing diagram of strobe initiated by destination is shown in figure below:



(a) Block diagram.



(b) Timing diagram.

In block diagram, we see that, the strobe initiated by destination, and as shown in timing diagram the destination unit first activates the strobe pulse, informing the source to provide the data.

✓ Now, actually in computer, in the first case means in strobe initiated by source — the strobe may be a memory-write control signal from the CPU to a memory unit. The source, CPU places the word on the data bus and informs the memory unit, which is the destination, that is a write operation.

✓ And in the second case i.e. in the strobe initiated by destination — the strobe may be a memory read control from the CPU to a memory unit. The destination, the CPU, initiates the read operation to inform the memory, which is a source unit, to place selected word into the data bus.

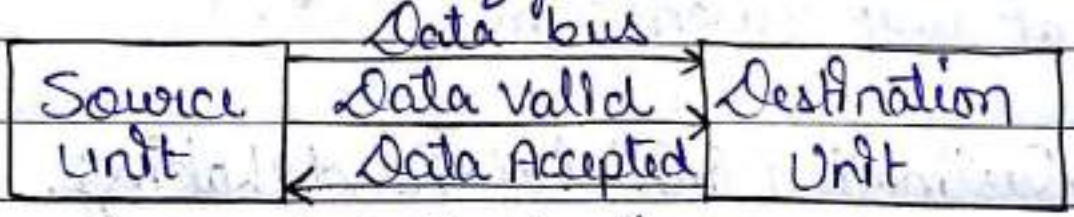
PAGE NO. // //
DATE // //

2- Handshaking :-

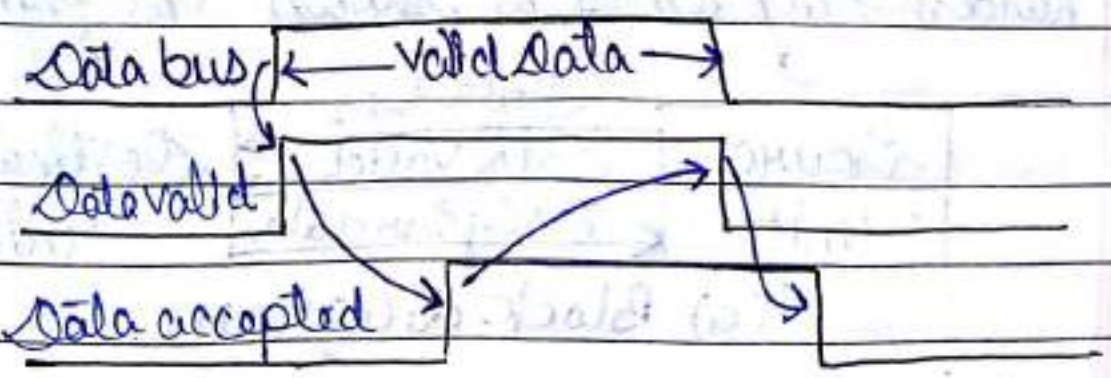
The disadvantage of Strobe method is that source unit that initiates the transfer has no way of knowing whether the destination has actually received the data that was placed in the bus. Similarly, a destination unit that initiates the transfer has no way of knowing whether the source unit has actually placed data on the bus.

- **Source Initiated Handshaking**

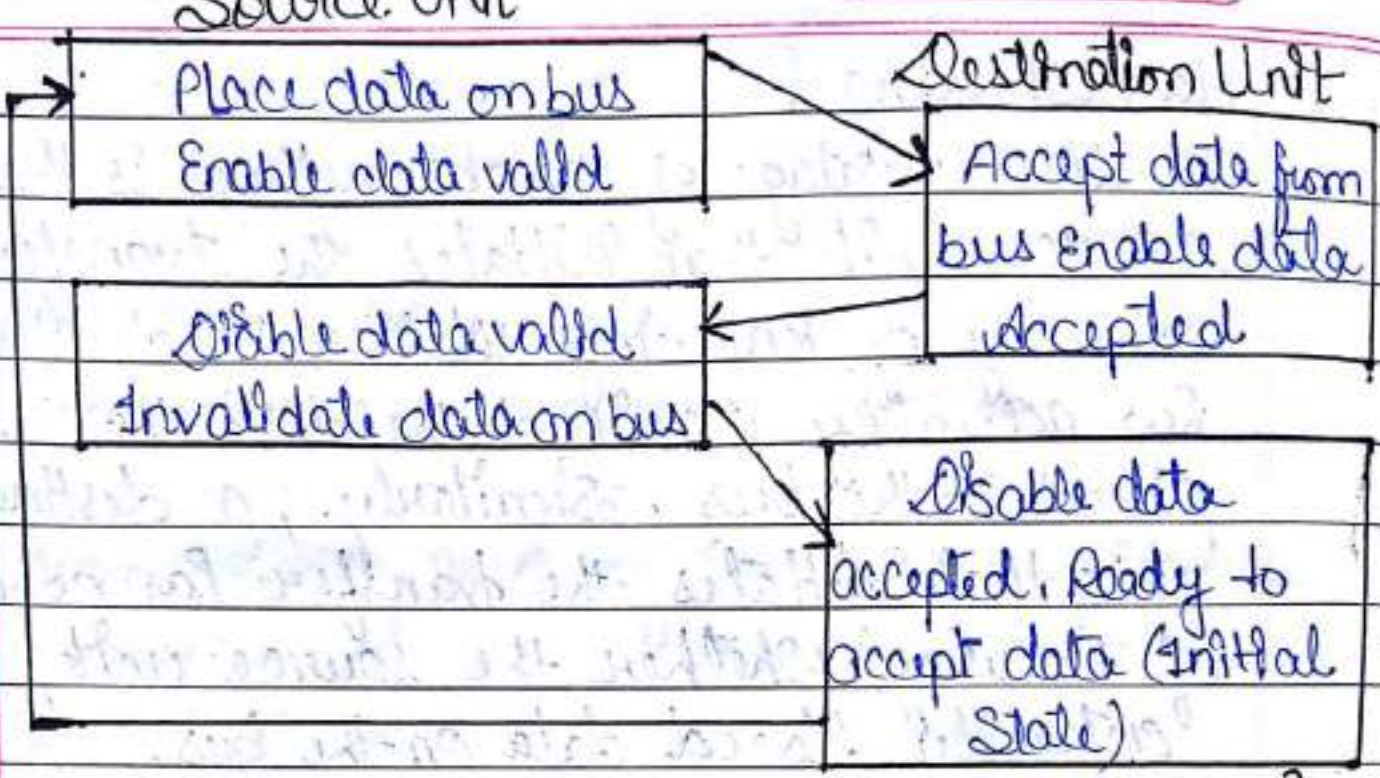
The source initiated transfer using handshaking lines is shown in figure below:



(a) Block diagram



(b) Timing diagram.

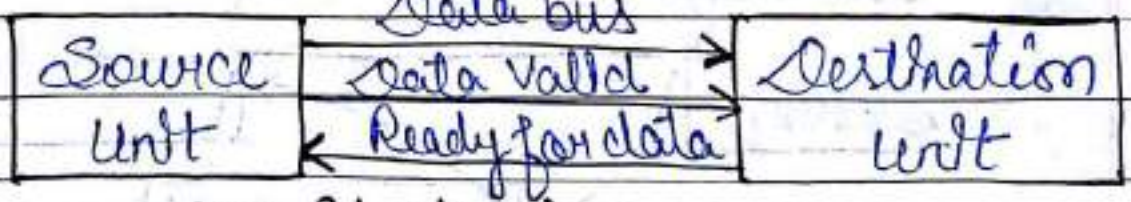


(C) Sequence Diagram (sequence of events)

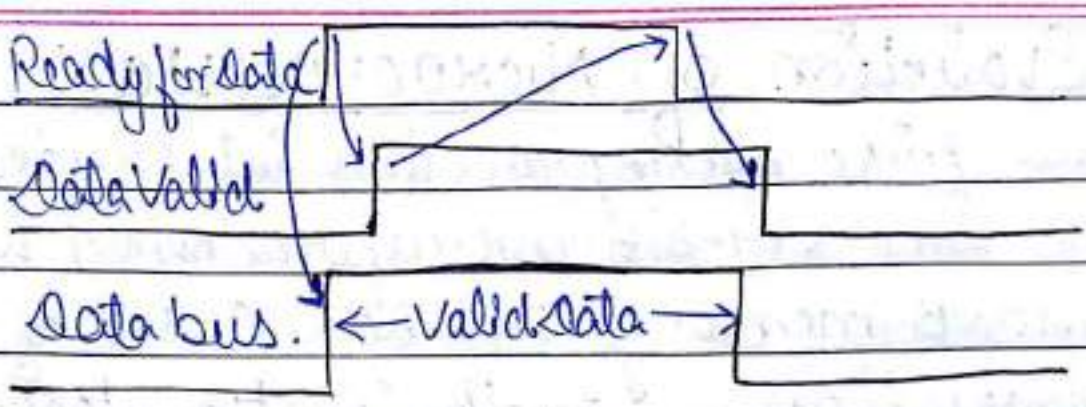
This sequence of events described in its sequence diagram, which shows the above sequence in which the system is present, at any given time.

- Destination Initiated handshaking.

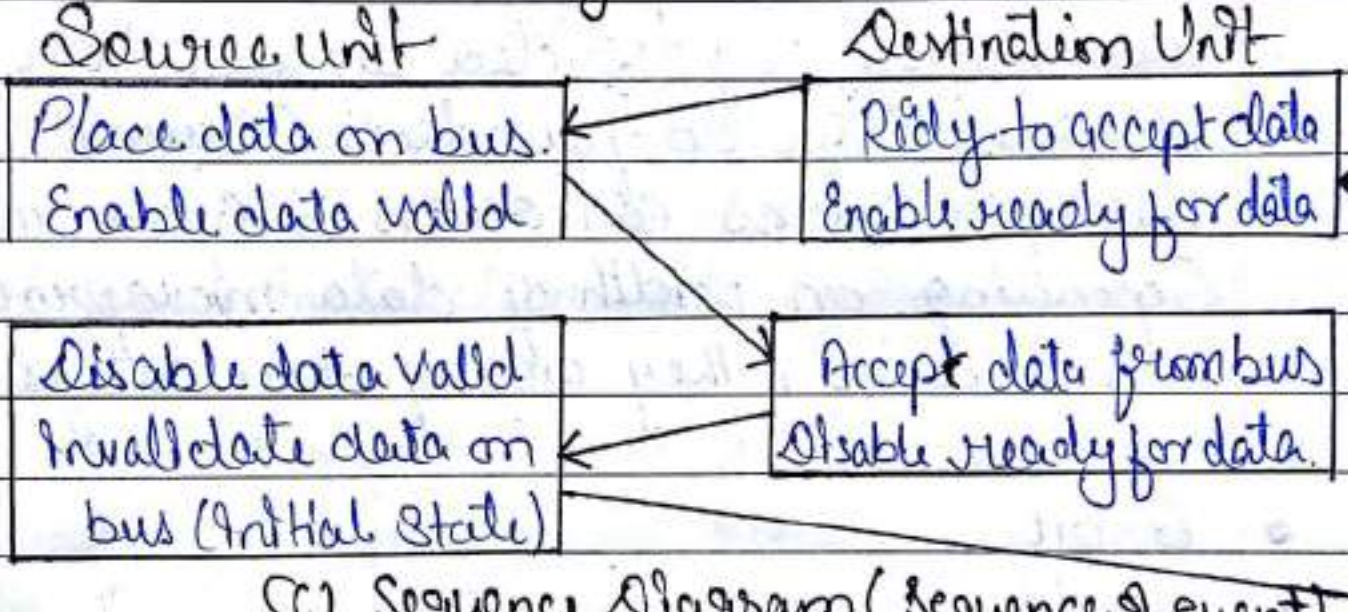
The destination initiated transfer using handshaking ones is shown in figure below:



(a) Block diagram.



(b) Timing Data



(c) Sequence Diagram (Sequence of event).

The sequence of event in it are shown in its sequence diagram and timing relationship b/w signals is shown in its timing diagram.

UNIT-5

PAGE NO. // //
DATE // //

Evaluation of Microprocessor

The first microprocessor introduced in 1971 was a 4-bit microprocessor with 4m5KB memory and had a set of 45 instructions. In the past 5 decades microprocessor speed has doubled every two years, as predicted by Gordon Moore, Intel co-founder. Current microprocessors can access 64GB memory. Depending on width of data microprocessors can process, they are of these categories-

- 8-bit
- 16-bit
- 32-bit
- 64-bit

Size of instructions set is another important consideration while categorizing microprocessors. Initially, microprocessors had very small instructions sets because complex hardware was expensive as well as difficult to build.

PAGE NO. // //
DATE // //

As technology developed to overcome these issues, more and more complex instruction were added to increase functionality of the microprocessor. However, soon it was realized that having large instructions sets was counterproductive as many instructions that were rarely used sat idle on pro precious memory space. So the old school of thought that supported smaller instruction sets gained popularity.

Name	Year	Transistors	Data width	Clock Speed
8080	1974	6,000	8 bits	2MHz
8085	1976	6,500	8 bits	5MHz
8086	1978	29,000	16 bits	5MHz
8088	1979	29,000	8 bit	5MHz
80286	1982	134,000	16 bits	6MHz
80386	1985	275000	32 bits	16MHz
80486	1989	1200,000	32 bits	25MHz
PENTIUM	1993	3,100,000	82/64 bits	60MHz
PENTIUM II	1997	7500,000	64 bits	233MHz
PENTIUM III	1999	9,500,000	64 bits	450MHz
PENTIUM IV	2000	42,000,000	64 bits	1.5GHz

PAGE NO. // //
DATE // //

Microprocessor - 8085 :-
8085 is pronounced as "eighty-eighty-five" microprocessor. It is an 8-bit microprocessor ~~by~~ designed by Intel in 1977 using NMOS technology.

- 8-bit data Bus
- 16-bit address bus, which can address upto 64Kb.
- A 16-bit program counter
- A 16-bit stack pointer
- Six 8-bit registers arranged in pairs:
- BC, DE, HL.
- Requires +5V Supply to operate at 3.2 MHz Single phase clock. It is used in Washing machines, microwaves ovens, mobile phone, etc.

8085 Microprocessor - Functional Units :-

1- Accumulator :- It is an 8-bit register used to perform arithmetic, logical, I/O and LOAD/STORE operations. It is connected to internal data bus and ALU.

PAGE NO. // //
DATE // //

2- Arithmetic and Logic unit :-
As the name suggests, it performs arithmetic and logical operations like Addition, Subtraction, AND, OR, etc. on 8-bit data.

3- General purpose register :- There are 6 general purpose registers in 8085 processor i.e. B, C, D, E, H and L. Each register can hold 8-bit data. These registers can work in pair to hold 16-bit data and their pairing combination is like B-C, D-E and H-L.

4- Program Counter :- It is a 16-bit register used to store the memory address location of the next instruction to be executed. Microprocessor increments the program whenever an instruction is being executed, so that the program counter points to the memory address of the next instruction that is going to be executed.

5- Stack pointer :- It is also a 16-bit register works like stack, which is always incremented / decremented by 2 during push

and pop operations.

6- Temporary register :- It is 8-bit register which hold the temporary data of arithmetic and logical operations.

7- Flag register - It is an 8-bit register having five 1-bit flip-flop which hold either 0 or 1 depending upon the result stored in the accumulator.

i) Sign (S).
ii) Zero (Z).
iii) Auxiliary carry (AC)
iv) Parity (P)
v) Carry (C).

Its bit position is shown in the following table-

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z		AC		P		CY

8- Instruction register and decoder -
It is provides timing and control signal to the microprocessor to perform

operations. Following are the timing and control signals, which control external and internal circuits - control signals - READY, RD', WR', ALE
Status signals - S0, S1, IO/M'
DMA signals - HOLD, HLDA
RESET signals - RESET IN, RESET OUT.

10- Interrupt control - As the name suggests it controls the interrupts during a process. When a microprocessor is executing a main program an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming requests. After the request is completed, the control goes back to the main program. There are 5 interrupt signals in 8085 microprocessor - INTR, RST 7.5, RST 6.5, RST 5.5, TRAP.

11- Serial Input/Output control - It controls the serial data communication by using these two instructions -

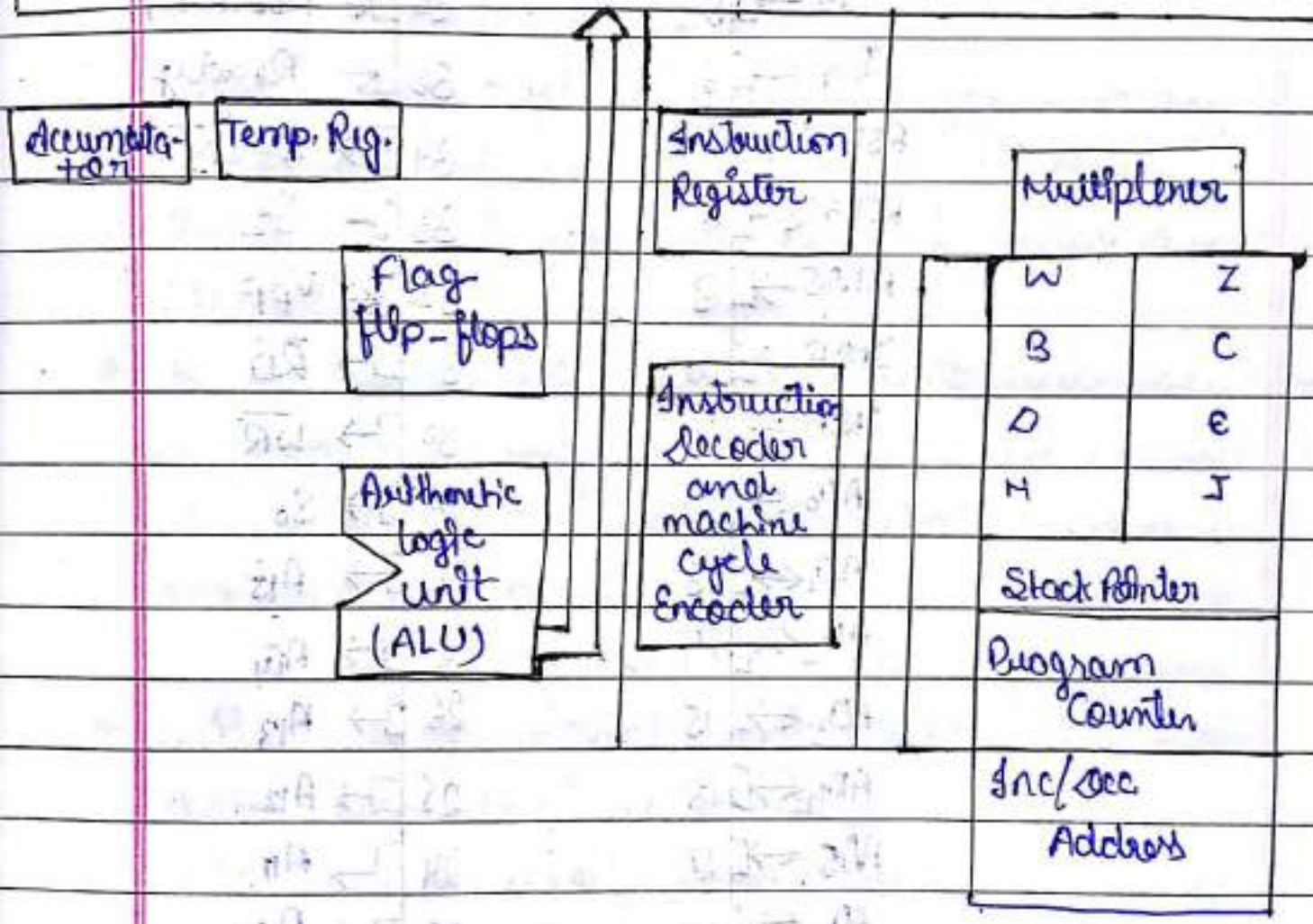
(ii) SID (Serial input data)
(iii) SOD (Serial output data).

12- Address buffer and address-data buffer - The content stored in the stack pointer and program counter is loaded into the address buffer and address-data buffer to communicate with the ~~connected~~ CPU. The memory and I/O chips are connected to these buses; the CPU can exchange the desired data with the memory and I/O chips.

13- Address bus and data bus - Data bus carries the data to be stored. It is bidirectional, whereas address bus carries the location to where it should be stored and it is unidirectional. It is used to transfer the data and address I/O devices.

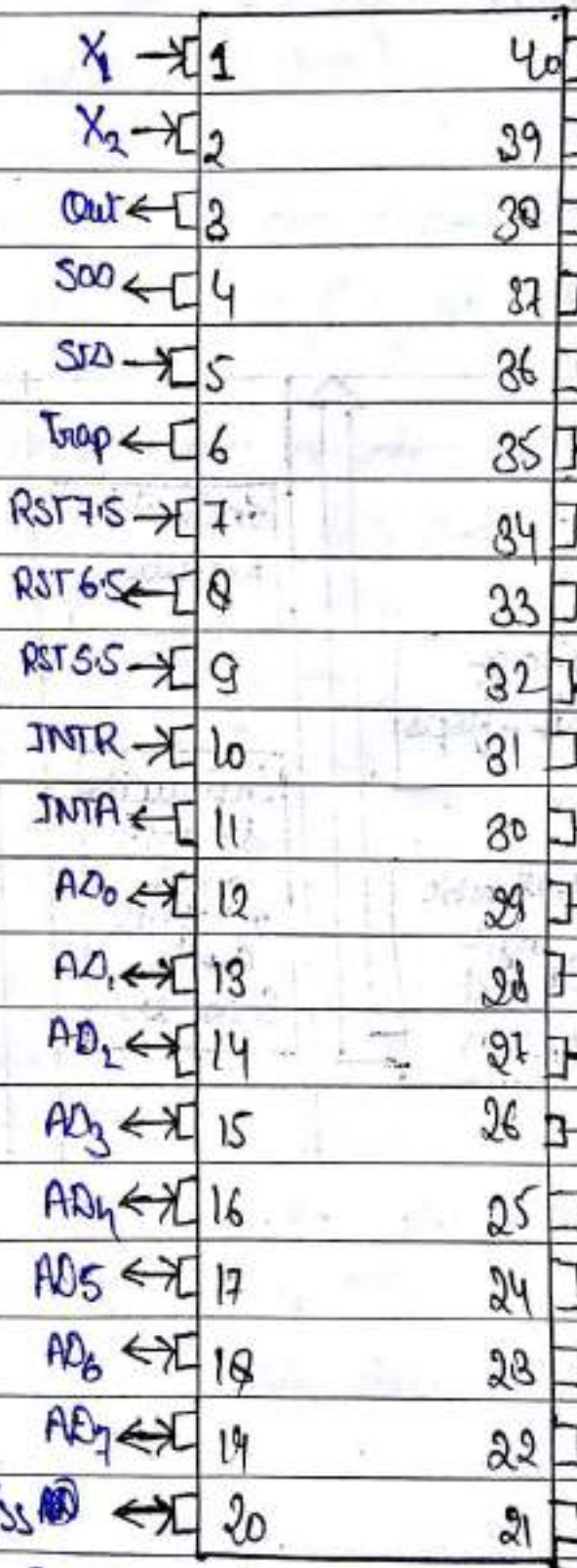
PAGE NO. // //
DATE // //

★ 8085 Architecture-



The following figure depicts the pin diagram of 8085 microprocessor –

PAGE NO. // //
DATE // //



The pin of a 8085 microprocessor can be classified into seven groups –

PAGE NO. // //
DATE // //

1- Address bus- A15 - A8, It carries the most significant 8-bits of memory / IO address.

2- Data bus - ~~AD8~~ AD7 - AD0, It carries the least significant 8-bit address and data bus.

3- Control and status signals- These signals are used to identify the nature of operation. There are 3 control signal and 3 status signals.

- RD - This signal indicates that the selected IO or memory device is to be read and is ready for accepting data available on the data bus.

- WR - This signal indicates that the data on the data bus is to be written into a selected memory or IO location.

- ALE - It is a positive going pulse generated when a new operation is starting in the microprocessor. When the pulse goes high, it indicates address. When the pulse goes down it indicates data.

PAGE NO. // //
DATE // //

Three status signals are IO/M, S0 & S1.

1- IO/M - This signal is used to differentiate between IO and memory operations, i.e. when it is high it indicates IO operation and when it is low then it indicates memory operation.

2- S1 & S0 - These signals are used to identify the type of current operation.

3- Power Supply - There are two power supply signals - VCC & VSS indicates +5V power supply and it indicates ground signal.

Clock signals - There are 3 clock signals - i.e. X_1, X_2, CLK OUT.

- X_1, X_2 - A crystal (RC, LC N/W) is connected at these two pins and is used to set frequency of the internal clock generator. This frequency is internally divided by 2.

PAGE NO. // //
DATE // //

- CLK Out :- This signal is used as the system clock for devices connected with the microprocessor.

* Interrupts & externally Initiated Signals :- Interrupt are the signals generated by external devices to request the microprocessor do perform a task. There are 5 Interrupt signals, i.e TRAP, RST 7.5, RST 6.5, RST 5.5 and INTR.

- INTA - It is an Interrupt acknowledgement signal.
- RESET IN - This signal is used to reset the microprocessor by setting the program counter to Zero.
- RESET OUT - This signal is used to reset all the connected devices when the microprocessor is reset.
- READY - This signal indicates that the device is ready to send or receive data. If READY is low, then the CPU has to wait for READY to go High.
- HOLD - This signal indicates that another master is requesting the use of address and data buses.

PAGE NO. // //
DATE // //

- HLDA (Hold Acknowledgement) - It indicates that the CPU has received the HOLD request and it will relinquish the bus in the next clock cycle. HLDA is set to low after the HOLD signal is removed.

7. Serial I/O signals :- There are 2 serial signals, i.e. SID and SOD and these signals are used for serial communication.

- SOD (serial output data line) :- The output SOD is set / reset as specified by the SIM instruction.

- SID (Serial Input data line) :- The data on this line is loaded into accumulator whenever a RIM instruction is executed.

★ Addressing Modes in 8085 :-
These are the instructions used to transfer the data from one register to another register, from the memory to the register, and from the register to

PAGE NO. // //
DATE // //

the memory without any alteration in the content. Addressing modes in 8085 is classified into 5 groups -

1- Immediate addressing mode - In this mode, the 8/16 bit data is specified in the instruction itself as one of its operand. For example:- MVI K, 20F: means 20F is copied into register K.

2- Register addressing mode - In this mode, the data is copied from one register to another. For en:- MOV K, B: means data in register B is copied to register K.

3- Direct addressing mode - In this mode, the data is directly copied from the given address to the register.
For en - LDA 5000K: means the data at address 5000K is copied to register B.

4- Indirect addressing mode - In this mode, the data is transferred from one register to another by using the address pointed

by the register.
For en- MOV K, B: means data is transferred from the memory address pointed by the register to the register K.

5- Implied addressing mode- This mode does not req require any operand, the data is specified by the opcode itself.
For en- CMP.

★. Interrupts in 8085:-
Interrupts are the signals generated by the external devices to request the microprocessor to perform a task. There are 5 interrupt signals, i.e. TRAP, RST 6.5, RST 7.5, RST5.5 and INTR.
Interrupt are classified into following groups based on their parameter-

1- Vector Interrupt- In this type of interrupt the interrupt address is known to the processor.
For en:- RST 5.5, RST 6.5, RST 7.5, TRAP.

PAGE NO. // //
DATE // //

2- Non-Vector Interrupt - In this type of Interrupt, the Interrupt is not known do the processor so, the Interrupt address needs do be sent externally by the device to perform Interrupts. for ex - INTR.

3 Markable Interrupt :- In this type of Interrupt, we can disable the Interrupt by writing some Instructions into the program. For example: RST7.5, RST6.5, RST 5.5.

4- Non-Markable Interrupt :- In this type of Interrupt we cannot disable the Interrupt by writing some Instructions into the program. for example: TRAP.

5- Software Interrupt :- In this type of Interrupt, the programmer has to add the Instructions into the program to execute the Interrupt. There are 8 Software Interrupt in 8085, i.e. RST0, RST1, RST2, RST3, RST4, RST5, RST6, and RST7.

PAGE NO. // //
DATE // //

6- • Hardware Interrupt :- There are 5 interrupts pins in 8085 used as hardware interrupt, i.e., TRAP, RST 7.5, RST 6.5, RST 5.5, INTA.

NOTE :- INTA is not an interrupt, It is used by the microprocessor for sending acknowledgement. TRAP has the highest priority, then RST 7.5 and so on.

• Interrupt Service Routine (ISR) :-
A small program or a routine that when executed, services the corresponding interrupting source is called an ISR.

★. TRAP :-
It is a non-markable interrupt; having the highest priority among all interrupts. By default, It is enabled until it gets acknowledged.
In case of failure, It executes as ISR and sends the data to backup memory. This interrupt transfer the control to the location 0024H.

PAGE NO. // //
DATE // //

RST 7.5 :- It is a Maskable Interrupt, having the second highest priority among all Interrupts. When this Interrupt is executed, the processor saves the content of the PC register into the stack and branches to 003CH address.

RST 6.5 :- It is a maskable Interrupt, having the third highest priority among all Interrupts. When this Interrupt is executed the processor saves the content of the PC register into the stack branches to 0034H address.

RST 5.5 :- It is a maskable Interrupt. When this Interrupt is executed, the processor saves the content of the PC register into the stack and branches to 002CH address.

INTR :- It is a maskable Interrupt, having the lowest priority among all Interrupts. It can be disabled by reseting the microprocessor.

- When INTR signal goes high, the following event can occur -

PAGE NO. // //
DATE // //

- The microprocessor checks the status of INTR signal during the execution of each instruction.
- When the INTR signal is High, then the microprocessor completes its current instructions and sends active low interrupt acknowledge signal.
- When instructions are received, then the microprocessor saves the address of the next instruction on stack and executes the received instruction.

END

Hope the study material was helpful , to stay connected with us : Visit Us

www.dreamstudy.tk

https://www.facebook.com/dreamstudy

https://www.instagram.com/dream.study/

https://www.youtube.com/channel/UC8l0Dfy2YekfiigaXEtbAow or
search on youtube with - dreamstudy website

https://twitter.com/DreamStudy_