Computer Organisation.

Unit 1

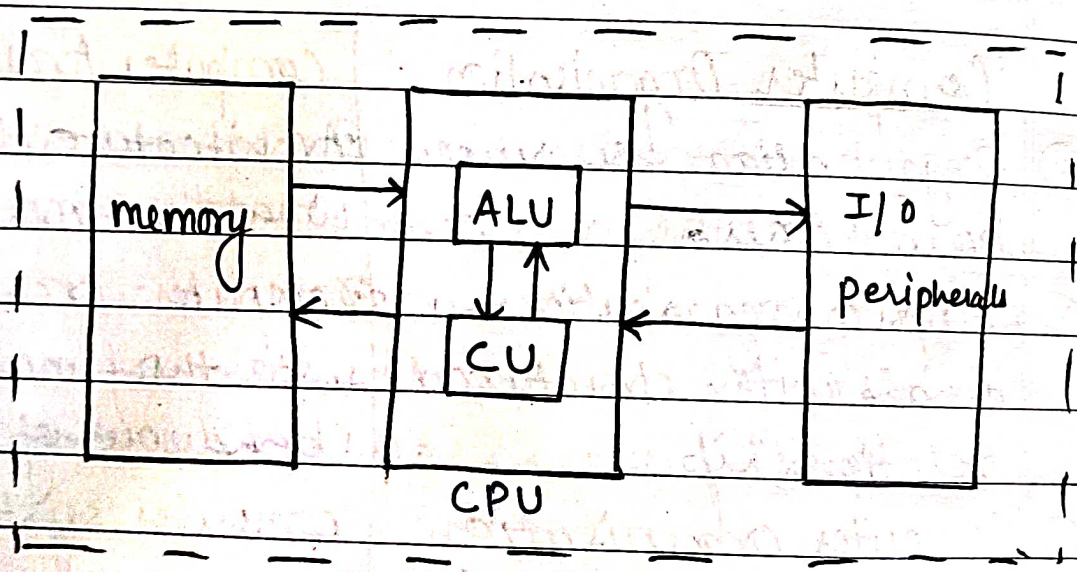
General Purpose system
For general purpose.

Embbed system
For particular purpose.

Difference b/w computer Architecture & Organisation

Computer Organisation	Computer Architecture
Organisation describes how it does.	Architecture describes what the computer does.
Computer organisation deals with structural relationship.	Computer Arch. deals with the functional behaviour of the computer system.
Computer organisation deals with low level design issues.	Comp Arch. deals with high level design issues.
Computer organisation indicates its performance.	Arch. indicates its hardware.
For designing a comp. organisation is decided after its architecture.	For designing a computer its Architecture is fixed first.
Also called micro architecture frequently.	Also called instruction set architecture.

consist of physical unit circuit designs pheripherals or adders	Comp. Arch comprises logical functions such as instructions set, register, data types, addressing modes.



Computing System.

1. Computer is programmable machine
2. Computer is a machine that manipulates data according to list of instructions.
3. computer is any device which aids humans in performing various kinds of computation or calculations.

Three Principle characteristics of Computer.

1. Its ~~rep~~ responses to a specific set of instructions. in a well defined manner.
2. It can execute pre recorded list of instructions.
3. It can quickly store or retrieve the large amount of data.

Structure of the computer.

1. Input Unit

- Computer accepts coded information through input units. the most ~~comm~~ common input device is Key whenever key is pressed the corresponding letter or digit is automatically translated into corresponding binary code and transmitted to processor.
- Many other kinds of input devices for the human computer interaction are available Including the touchpad, mouse, joysticks and trackball these are often used as graphic input devices in conjunction with a display.
- Microphones can be used to capture audio inputs which is then sampled and converted into digital codes for storage and processing
- Similarly cameras can be used to capture video.

Digital Communication facilities such as internet can also provide inputs to the computer from other data base server.

② Output Unit

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

For example:- printer

however printers are mechanical devices and such are quit slow compares to electronic speed of a processor.

Some units such as graphic displays provide both an output function, showing texts and graphics and then input function through touch screen capability

- The ~~is dual~~ dual goal for such unit is the reason for ~~separate~~ input output in many places.

③ Memory Unit

- The function of memory unit is to store programs and data.

There are few classes of storage, primary and secondary.

Primary Memory → also called main memory is a fast memory that operates at electronic speed.

- Programs must be stored in this memory while there are being executed.

~~Them~~

- The memory consist of large no. of memory cells each capable of storing 1 bit of information.
- Instruction and data can be written into or read from the memory under control of the processor.

Secondary Storage

- Although primary memory is essential, it tends to be expensive & doesn't retain information when power is turn off.
- An additional, less expensive permanent secondary storage is used when large amount of data and many programs have to be stored particularly for information that is accessed infrequently.
- Access time for secondary storage are longer than for primary memory.
- A wide selection of secondary storage is available including magnetic disk, optical disk & memory devices.

CPU

1. ALU (Arithmetic logic Unit)

- Computer operations and executed is the arithmetic and logic unit of the processor.
- Any arithmetic or logic operation is subtraction addition, multiply etc is initiated by bringing required operands into the processor where operations is performed by ~~accumulator~~ ALU.
- When operands are bought into the processor they are stored in high speed storage element called registers.
- Each register can store 1 word of data.
- Access time to resistors ~~are~~ are even shorter than access time to cache unit of processor chip.

2. Control Unit (CU)

- The memory, arithematic and logic unit and input/output units store and process information and perform input/output operation.
- The operations of these units must be coordinated in some way.
- This is responsibility of control unit.

- The control is sends control signals to other units and senses their states
- CU are responsible for sending timing signals that govern the transfers and determine when a given action will take place.
- Data transfer between processor and the memory are also managed by the control unit through timing signals.
- A large set of control lines, wires carries the signals used for timing and synchronisation of events in all units.

Computer Generation

The zero generation (1642 - 1945)

- In this duration mechanical calculating machines were invented that could add, sub, multiply or divide
- None of these devices could be programmed or had memory.
- They required manual intermention through out each of their calculations.

For example:- Prov to 1500 a typical European bussinessman used an abacus for calculation.

First Generation Computer (1946 - 1958)

- The first Computer used vaccum tubes for circuitary and magnetic drums for memory and were often enormous taking up entire room.
- They were very expensive to operate and in addition to using a great deal of an electricity, generated a lot of heat which was often a cause of malfunction.
- First Generation computers relied on machine language the lowest level programing language understood by the computers to perform operations and they could only solve one problem at a time.
- Input was based on punched cards and paper tape and output was displayed on printouts

Second Generation Comp. (1959 - 1964)

- Transistors replaced vaccum tubes and firstly used in second generation computer.
- One transistor replaced equivalent of 40 vaccum tubes. allowing computers to become smaller, faster, cheaper and more energy efficient and more reliable still generate a great deal of heat that can create

Date.
Page No.

damage to computer.

- Second generation computer moved from cryptic binary machine to symbolic / assembly languages which allowed programmers to specify instruction in words.

- Second Generation computers still relied on punched cards on input and printout for outputs.
- The memory moved from magnetic drum to magnetic core technology.

Third Generation Computer (1965-1970)

- The development of IC was the hallmark of the third generation
- Transistors were minaturized and placed on silicon chip called semiconductors, which drastically increased the speed and efficiency of computer.
- Much smaller and cheaper compared to second generation computer it could carry out instructions in billions of seconds.

User interacted through keyboards and monitors and interfaced with operating system which allowed many different applications at same time with a central program that monitored the ~~program~~ memory.

Date.
Page No.

Computer for the first time become accessible to mass audiena because they were smaller & cheaper.

Fourth Generation Computers (1971 - today)

- The microprocessor brought 4th gen of computer as 1000s of IC's were built on to a single chip.

- These more powerful computers could be linked together to form networks.

4th generation computer also saw the development of GUI's, mouse and held computers.

Fifth Generation Computer (today - future)

Based on AI

- use of parallel processing and supercomputers is helping to make artificial intelligency
- The goal is to develope devices the respons to natural language input and are capable of learning and self organisation.
- There are some applications such as voice recognization and that are being used today.

Date
Page No.

Moore's law

In 1965, intel founder Gordon Moore stated "The density of a transistor in an IC will double every year. The current version of this predicition is usually conveyed as the density of a silicon chips doubles every 18th month. This assertion has become known as Moore's laws.

- Moore intented this postulate to hold for only ten years however the advances in chip manufacturing process have allowed this law to hold for almost 40 years. However this law cannot hold for ever there are physical and financial limitations that must ultimately come into play.

Hardwired Architecture

- In hardwired control units controls signal emanate from blocks of digital logic components these signals transmitted through all appropriate parts of the systems.
- These are typically very fast because they are actually physical components however ones implemented they are very difficult to modifie for the same reason.

Date.
Page No.

- Microprogram

1. It is a program written in a low level language that is implemented directly by the hardware, ~~machine ins~~
2. Machine instructions are produced in level 2. (instruction set architecture) are fed into microprograms which then interprets the instructions by activating hardware suited to executed the original instructions.
3. One machine level instruction is often translated into several micro code instructions.
4. Microprograms are popular because they can be modified relatably easily.
5. The disadvantage of microprograming is that the additional layer of translation typically results in slower instruction execution.

- ~~Von Neumann Model~~

- John W Mauchly & J Presper Eckart concise of an easier way to change the behaviour of their calculating machine the reckoned that memory devices in the form of ~~memory~~ mercury delay lines could provide a way

Date.
Page No.

* Not Important

to store a program instructions. This would forever end the Talium of rewiring the system.

- Each time it had a new problem to solve or an old one to debug.
- Mauchly & Eckart could not immediately publish their insight due to world war 2.
- After reading their proposal Von-Neumann Published the idea.
-

Von - Neumann Model

- All stored program computers have come to be known as Von-Neumann systems. using Von-Neumann architecture.

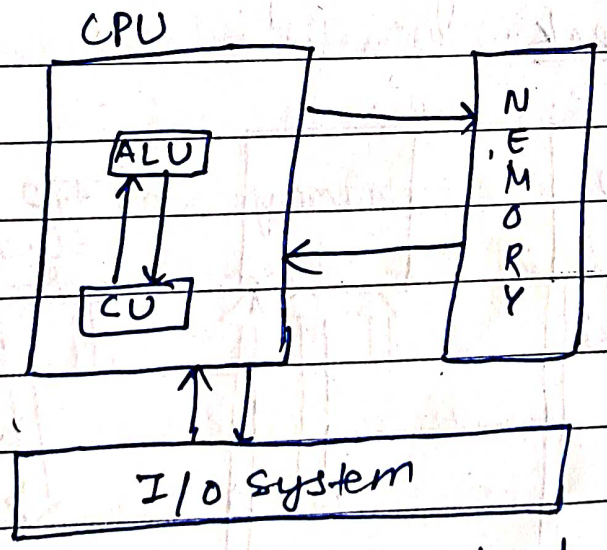


Fig 1-1

Todays version of the stored program machine architecture satisfies atleast the following characteristics.

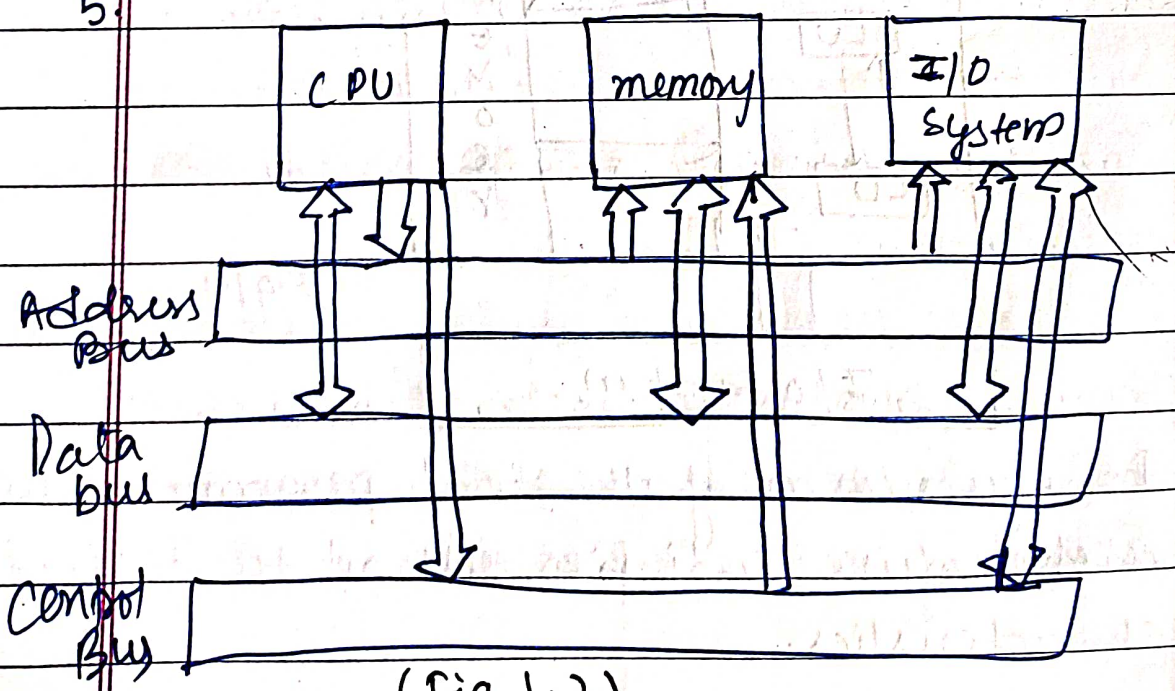
consist of three hardware system:
→ A CPU with a control unit and ALU, registers, program counter, instruction register, a main memory which holds program that controls the computer operations and an input output systems.

2. Capacity to carry out sequential instruction processing.

3. Contain a single path either physically or logically between the main memory system and the control unit of CPU.
Forcing alternation of instruction and execution cycles.

4. This single path is often referred to as <u>Von-Neumann bottle neck</u>.

5.



(Fig 1.2)

Date.
Page No.

~~Modified~~ Von-Neumann Model.

- ~~As~~ Figure (1.1) shows how these features work together in mordern computer system the system shown in the figure passes all of its input output through the arithmetic logic unit (ALU).
- The architecture runs programs in what known as the Von-Neumann execution cycle also called fetch-decode-execute cycle. which describes how the machine works one iteration of this cycle is as follows.

(1) The control unit fetches the next program instruction from the memory using the program counter. to determine ~~whether~~ where the instruction is located.

(2) The instruction is decoded into a language the ALU can understand and data operand requires to execute the instructions are fetch from the memory and placed into registers within the CPU.

(3) ALU executes the instructions and placed the results in registers and memory.

Non-Von Neumann Architecture

- Until recently almost all general purpose computers followed the Von-Neumann model However the Von-Neumann Bottleneck. continuous to baffle engineers looking for ways to build fast systems that are inexpensive and compatible with the ~~great~~ vast body of commerically available software Engineers who are not constraint by the need to maintain compatability with Von-Neumann systems are free to use many different models of computing.

- No. of different subfields fall into the Non-Von-Neumann Category including neural networks, genetic algorithm quantum computation and parallel computers. currently the most popular. Even parallel computing has its limits.

As the no. of processor increases so ~~the~~ does the overhead of managing how tasks are distributed to those processors

- Some parallel processing systems requires extra processors just to manage the ~~rest~~ list of processors and the ~~some~~ resources assigned to them.

Date.
Page No.

Modified Von Neumann Architecture

- The modified Von-Neumann Architecture (also known as the 'registers on bus' architecture). is a parallel processor architecture that is slightly different from normal von-Neumann architecture to allow for easier parallel processing.
- A modified Von-Neumann architecture processor is that some registers are read from a bus and the update registers values are then written to that bus.
- A modified von-Neumann architecture where I/O is directly coupled to multiple compute units and memory units respectively.
- In this memory mapped I/O lets input and output devices be treated
- It is based on the concept where there are separate storage and separate buses (signal path) for instruction and data. It was basically developed to overcome the bottle neck of Von Neumann

* Evolution of x86 Architecture.

- The x86 architecture continues to dominate the processor market outside of embedded systems (deticated task system). Although the
- organisation and technology of the x86 machine has changed dramatically over the decades the instruction set architecture has evolved to remain backward compatible with the earlier versions.

- The first 8086 was introduced with a clock speed of 5MHz and had 29 thousand transistors in 2008 a quad core intel core 2 introduced which operates at 3GHz.
- A speed up of a factor of 600 and has 820 million transistors. About 28000 times as many as the 8086.

It is worth while to list some of the highlight of the evolution of the intel x86 architecture.

① 8080 :- The world's first general purpose microprocessor this was an ~~[illegible]~~ 8-bit machine with 8-bit data path to memory.
The 8080 was used in the first personal computer.

Date.
Page No.

(2) 8086 → It is a far more powerful 16-bit machine in an addition to a wider data path and larger registers.
The 8086 has an instruction cache or queue that pre fetches a few instructions before their executing.
The 8086 is the first appearance of x86 architecture.

(3) 80286 →
- This extention of 8086 enabled addressing a 16 Mbyte memory instead of just 1MB.
-

(4) 80386 →
Intel's first 32-bit machine and major overhaul of the product.
Within 32-bit architecture the 80386 rivaled the complexity and power of many computers and main frames introduced just a few years earlier. This was first intel processor to support multi-tasking, meaning it could run multiple programs at the same time

(5) 80486 →

The 80486 introduced the use of much more sophisticated and powerful cache technology and sophisticated instruction pipelines the 80486 also offered a built in math core ~~operations~~ processor offloading complex math operations from the main CPU.

(6) Pentium :- With pentium intel introduced the use of superscaler techinques which allow multiple instructions to execute in parallel

(7) Pentium Pro :- The pentium pro continued the move into super scaler organisation began with the pentium with aggressive use of register renaming, branch prediction data flow analysis and speculative execution.

(8)

The pentium 2 incorporated intel mmx technology which is design specifically to process, video audio, graphic data efficiently.

Pentium 3 incorporate addition floating point instructions to support 3-D graphic software.

Date.
Page No.

Pentium4 it includes additional floating point and enhancement in multimedia.

(9) Intel Core Processor

It is a intel X86 up with a dual core referring to the implementation of two processor ~~of~~ on a singer chip.

Core 2 :- The core 2 extends the architecture to 64 bit. The core 2 quad provides 4 processor on a singer chip.

Data Representation.

- Magnitude
  - Unsigned Numbers (+ve values) only.
  - Signed Magnitude
- Complement
  - 1's complement
  - 2's complement

* All positive numbers are same in each type of Data Representation technique.

* Range $-(2^{n+1}-1)$ to $(2^{n-1}-1)$

where n - no. of bits

Date.
Page No.

* Range → $-(2^{n-1})$ to $+(2^{n-1}-1)$

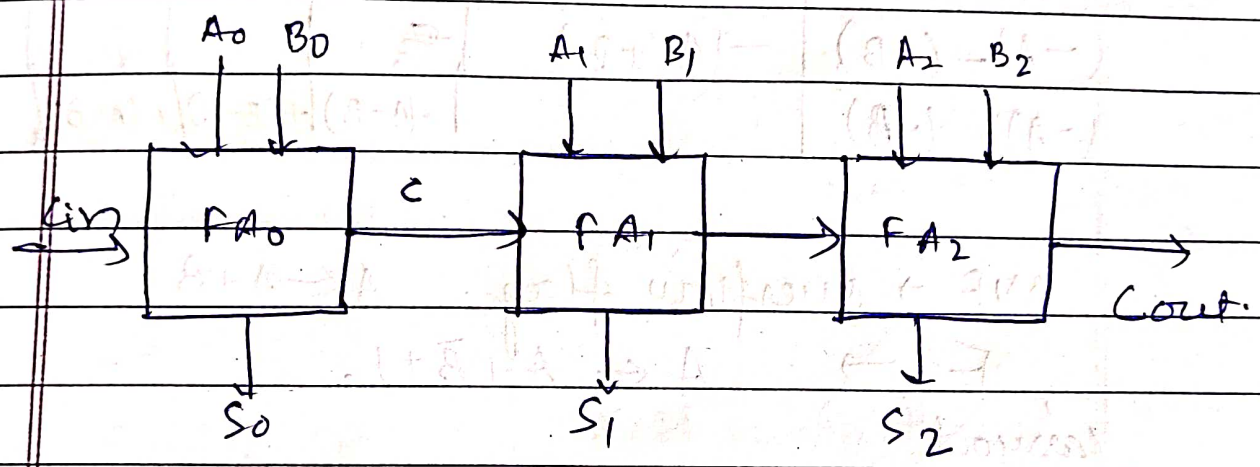
Binary	signed magnitude	1's compl	2's comp.
0111	+7	+7	+7
0110	+6	+6	+6
0101	+5	+5	+5
0100	+4	+4	+4
0011	+3	+3	+3
0010	+2	+2	+2
0001	+1	+1	+1
0000	+0	+0	0
1000	-0	-7	-8
1001	-1	-6	-7
1010	-2	-5	-6
1011	-3	-4	-5
1100	-4	-3	
1101	-5	-2	
1110	-6	-1	
1111	-7	-0	

Date.
Page No.

Adder → Parallel Adder

- Ripple Carry Adder
- Carry propogation Adder or Carry look Ahead Adder

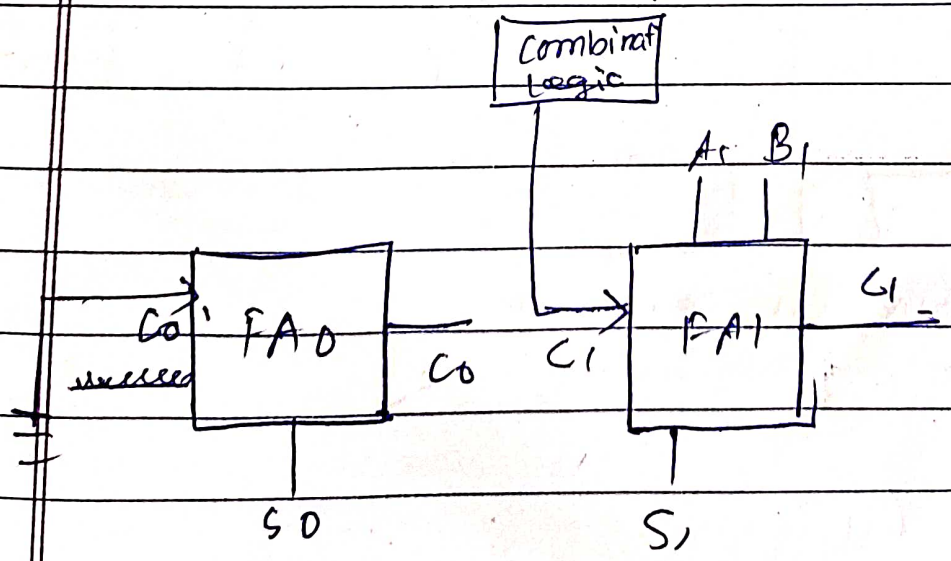
Ripple Carry adder



$C_{n+1} = C_n + P_i C_i$

$C_n = A \cdot B$

$P_i = A_i \oplus B_i$



Date.
Page No.

Addition / substraction of Signed Magnitude.

	Operation	Addition	Subtraction		
			$A > B$	$A < B$	$A = B$
Addition	$(+A) + (+B)$	$+(A+B)$			
	$(+A) + (-B)$		$+(A-B)$	$-(B-A)$	$+(A-B)$
	$(-A) + (+B)$		$-(A-B)$	$+(B-A)$	$+(A-B)$
	$(-A) + (-B)$	$-(A+B)$			
	$(+A) - (+B)$		$+(A-B)$	$-(B-A)$	$+(A-B)$
	$(+A) - (-B)$	$+(A+B)$			
	$(-A) - (+B)$	$-(A+B)$			
	$(-A) - (-B)$		$-(A-B)$	$+(B-A)$	$+(A-B)$

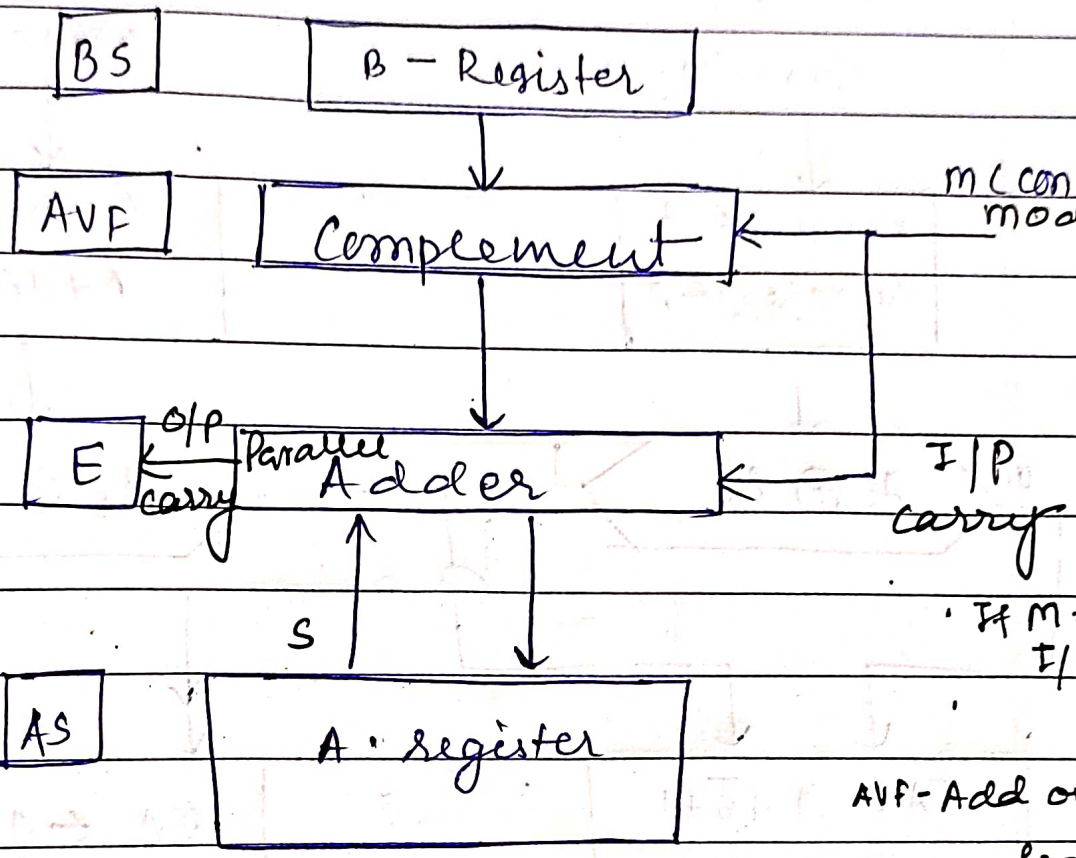
OVF → overflow flag. $A \leftarrow A+B$

E → $A \leftarrow A+\bar{B}+1$.

carryout storing

Date.
Page No.

Hardware for Add/Sub



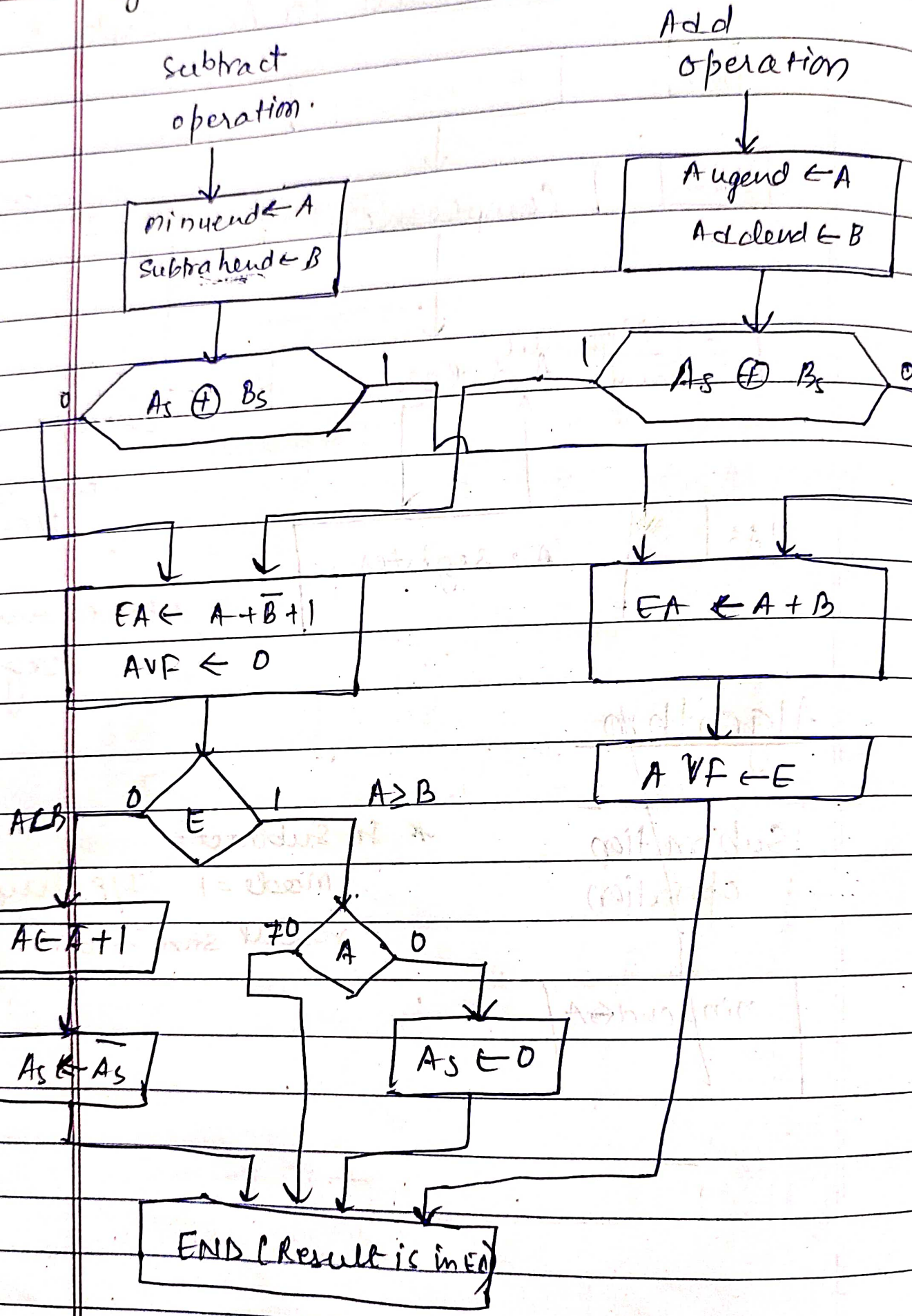
Algorithm

Subtraction operation

Minuend ↔ A

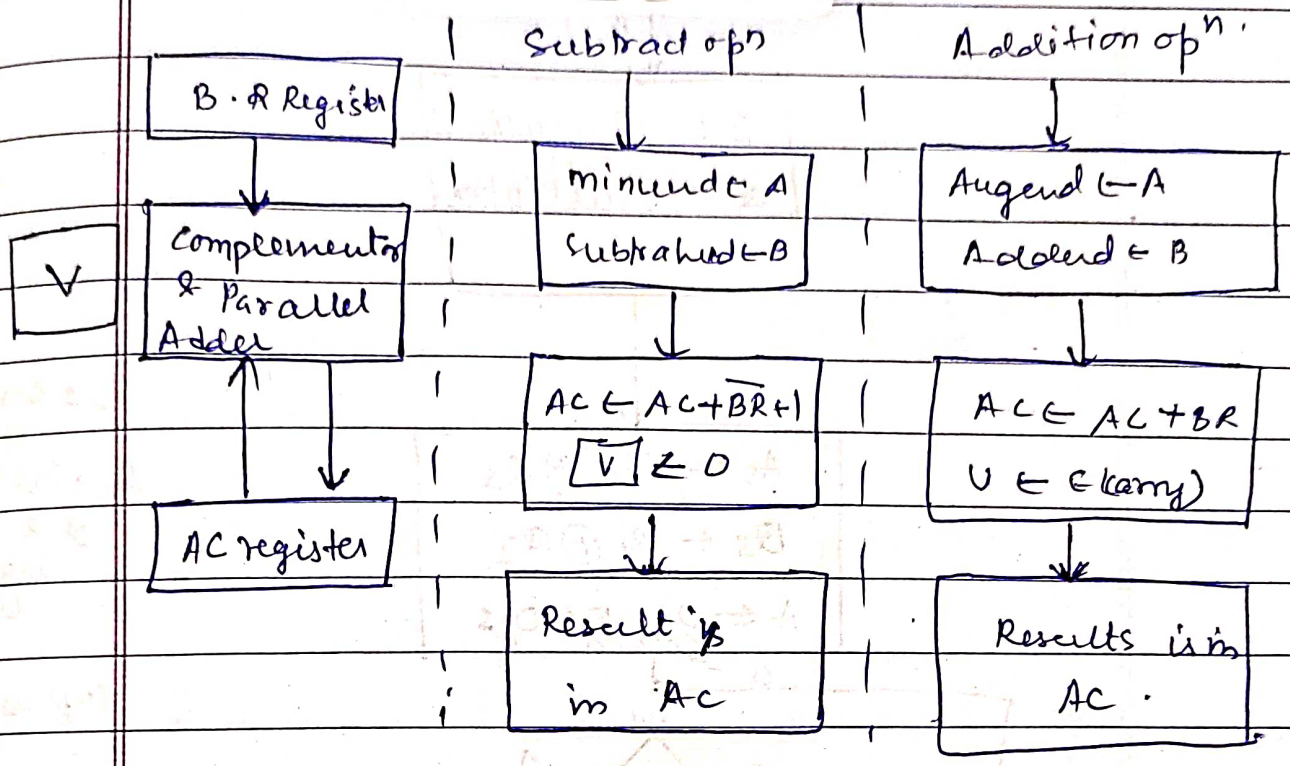
* In subtract
mode = 1 I/P carry = 1
result store in EA

Algorithm

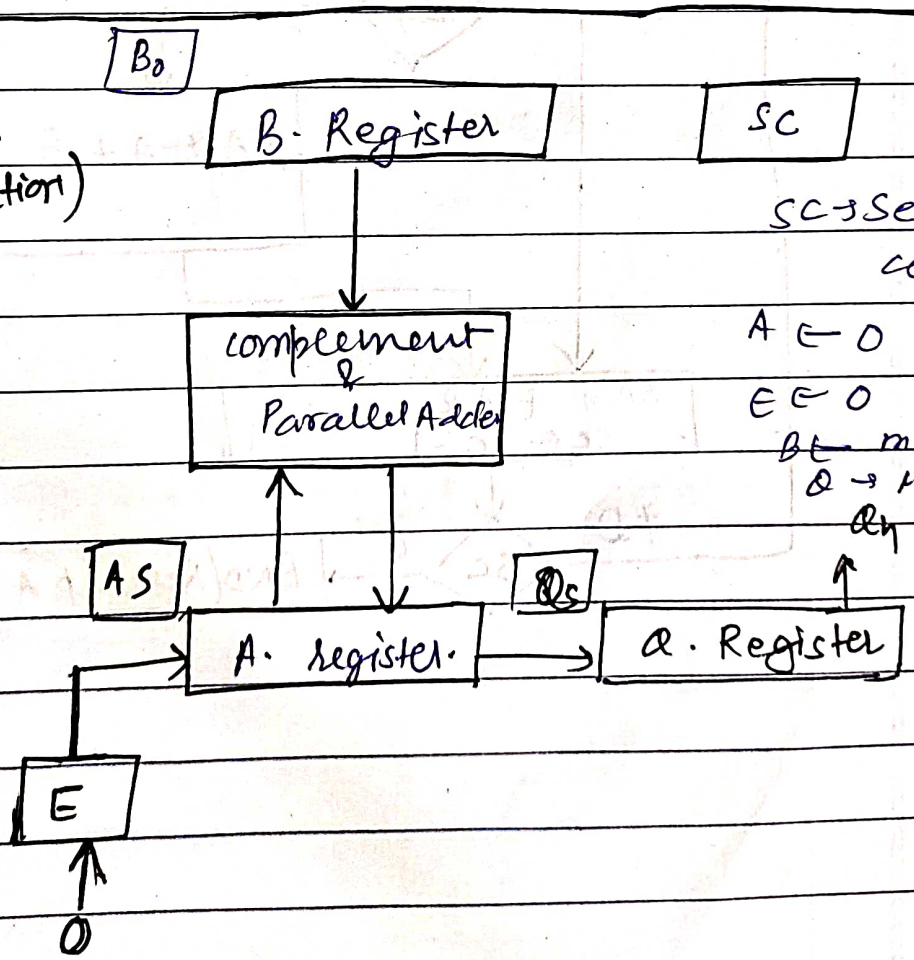

Subtract operation.
Add operation
minuend ← A
Subtrahend ← B
Augend ← A
Addend ← B
$A_s \oplus B_s$
$A_s \oplus B_s$
0
1
1
0
EA ← A + $\overline{B}$ + 1
AVF ← 0
EA ← A + B
A VF ← E
E
0
1
A<B
A≥B
A ← $\overline{A}$ + 1
A
≠0
0
A_s ← $\overline{A_s}$
A_s ← 0
END (Result is in EA)

Date.
Page No.

Addition / subtraction using 2's complement

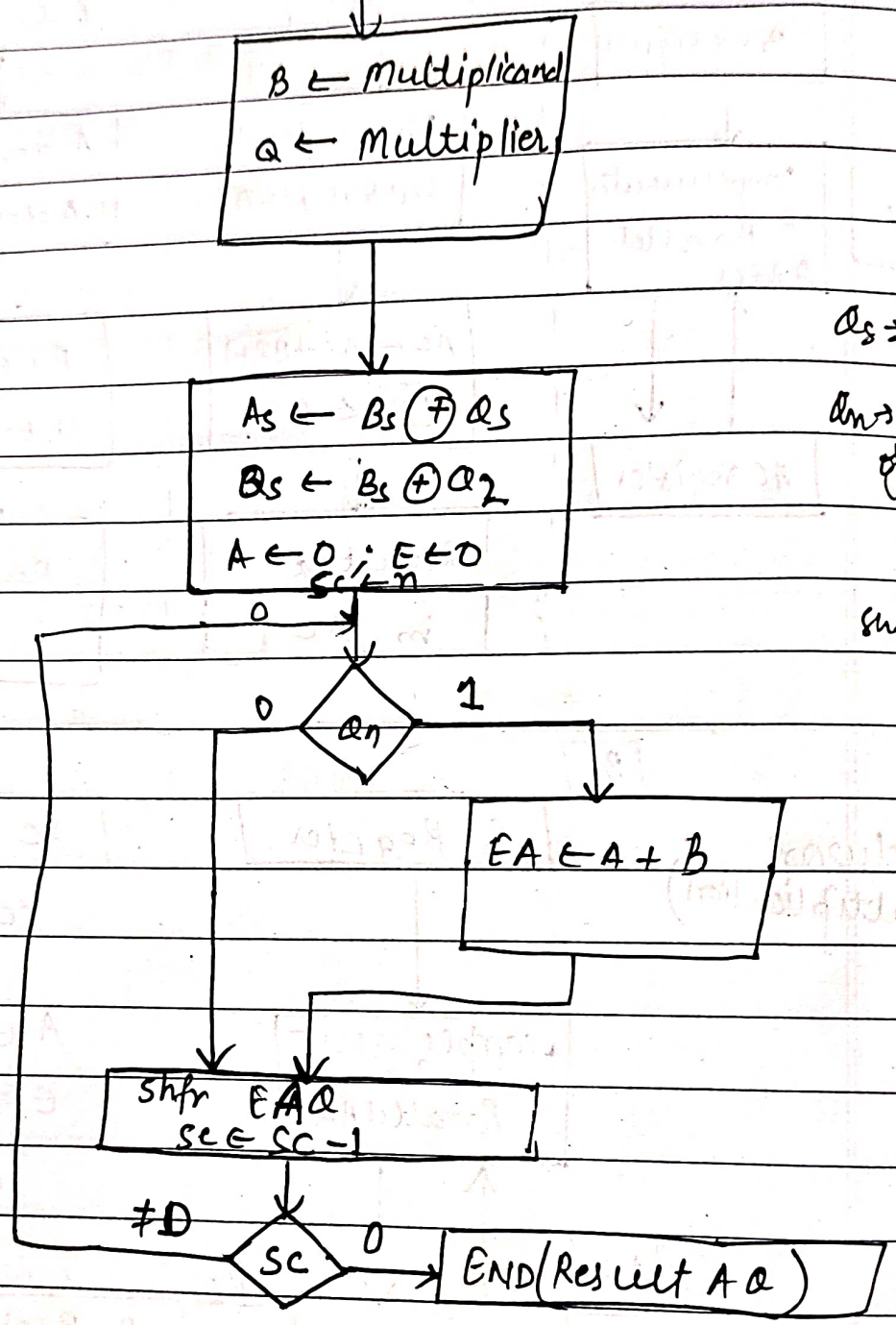


Hardware (multiplication)



Date.
Page No.

- Multiplication Algorithm for Signed Number.



Qs ⇒ signed bit
Qn ⇒ last bit of Q register

shr ⇒ shift right

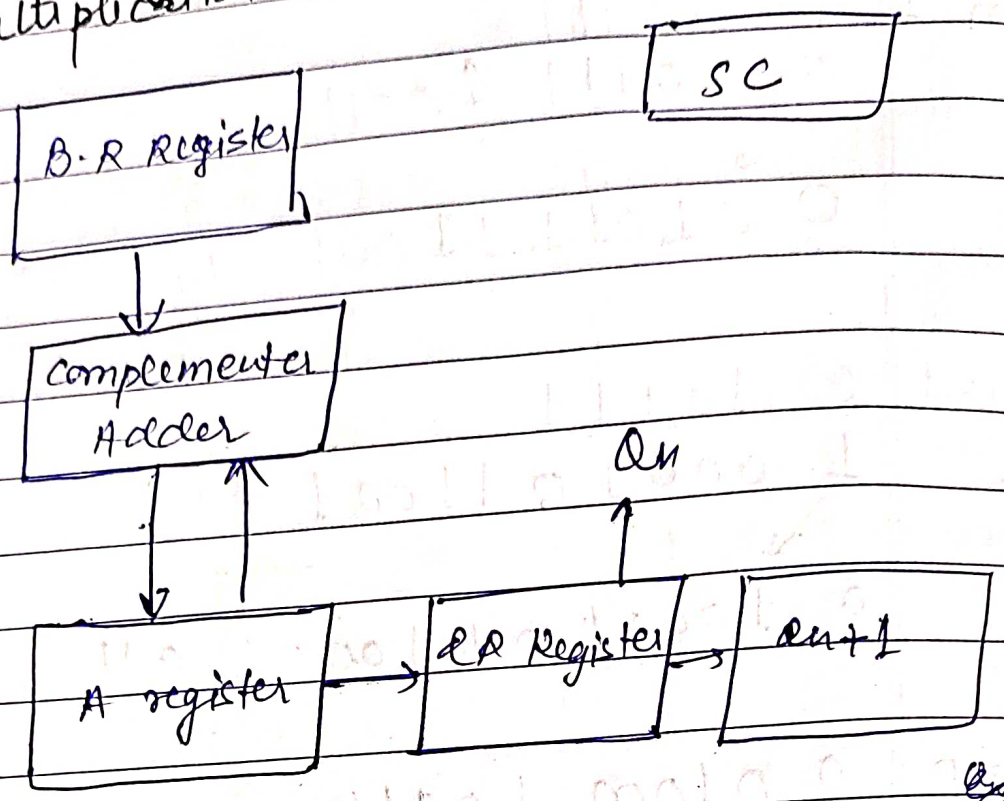
Date.
Page No.

(B = 10111)

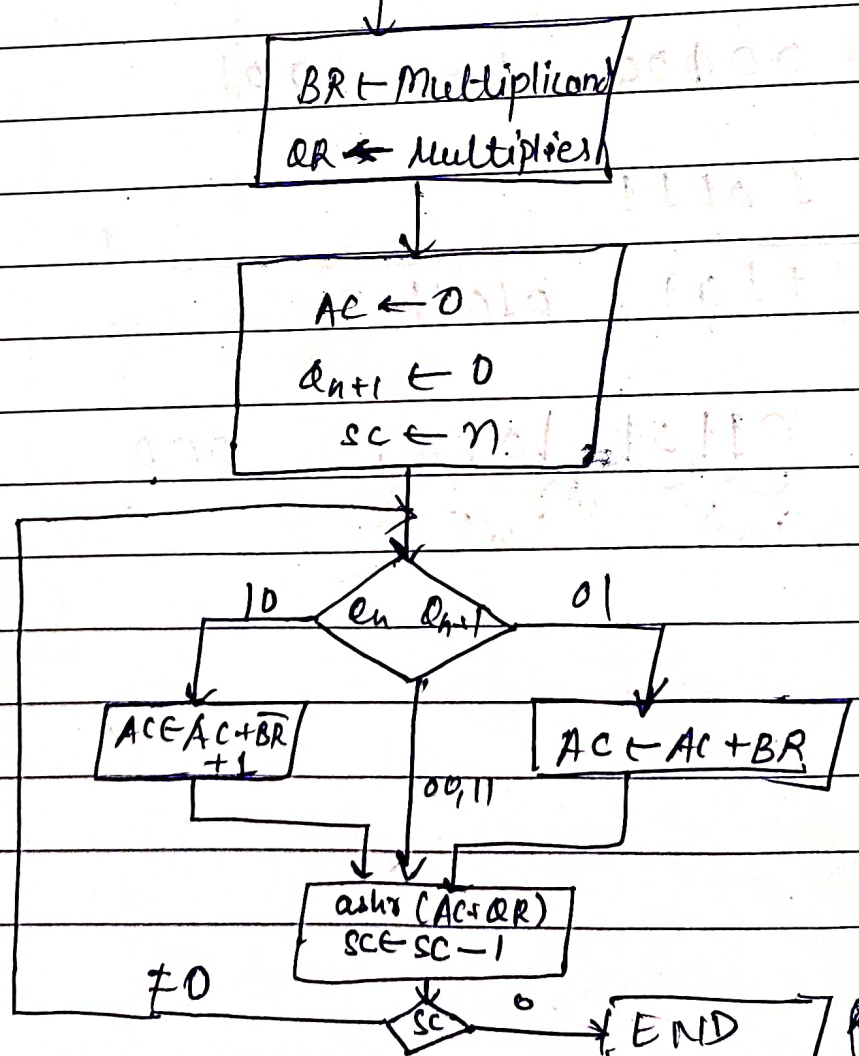
	E	A	Q	SC
Initialize	0	00000	10011	101
$Q_n = 1$		10111		
	0	10111	10011	
	0	01011	11001	100
$Q_n = 1$		10111		
	1	00010	11001	
	0	10001	01100	011
$Q_n = 0$ shift right	0	01000	10010	010
$Q_n = 0$	0	00100	01011	001
$Q_n = 1$		10111		
	0	11011	01011	
shift right	0	01101 (in A)	10101 (in Q)	000

Important

Booth's Algorithm for 2's complement multiplication.



Multiply op^n



Signed Integer Representation.

- A signed Integer is an integer with a +ve & -ve sign associated with it.
- Since the computer only understands binary, it is necessary to represent these signed integers in binary form.

Signed Integer

Signed bit	1's Complement	2's Complement
→ MSB represent sign of the Integer.	• for +ve same as signed	Range is $-(2^{n-1})$ to $(2^{n-1}-1)$
→ Range $-(2^{n-1}-1)$ to $(2^{n-1}-1)$	• for -ve write -ve no. in binary then take 1's cmp (-5) 1's cmp is +5	
Ex:- Signed extension for +5	Range :- $-(2^{n-1}-1)$ to $(2^{n-1}-1)$	* No ambiguity in the representation of 0.
+5 — 0101 (4 bit), 00101 (5 bit), 000101 (6 bit)		* Number are in cyclic order ie +7, -8
-5 — 1101 (4 bit), 10101 (5 bit)		* Signed Extension works.

Date.
Page No.

Table

Sign Magnitude	Decimal	1's comp	2's comp
	Positive Numbers		
0 0 0 0	+0	+0	0
0 0 0 1	+1	+1	+1
0 0 1 0	+2	+2	+2
0 0 1 1	+3	+3	+3
0 1 0 0	+4	+4	+4
0 1 0 1	+5	+5	+5
0 1 1 0	+6	+6	+6
0 1 1 1	+7	+7	+7
Negative No.			
1 0 0 0	-0	-7	-8
1 0 0 1	-1	-6	-7
1 0 1 0	-2	-5	-6
1 0 1 1	-3	-4	-5
1 1 0 0	-4	-3	-4
1 1 0 1	-5	-2	-3
1 1 1 0	-6	-1	-2
1 1 1 1	-7	-0	-1

Adder

A combination circuit to add/subtract the binary bits.

Ripple carry Adder

* Addition and subtraction of signed Magnitude Numbers.

Operations	Add of Magnitudes	Subtract		
		$P>Q$	$P<Q$	$P=Q$
$(+P)+(+Q)$	$+(P+Q)$			
$(+P)+(-Q)$		$+(P-Q)$	$-(Q-P)$	$+(P-Q)$
$(-P)+(+Q)$		$-(P-Q)$	$+(Q-P)$	$+(P-Q)$
$(-P)+(-Q)$	$-(P+Q)$			
$(+P)-(+Q)$		$+(P-Q)$	$-(Q-P)$	$+(P-Q)$
$(+P)-(-Q)$	$+(P+Q)$			
$(-P)-(+Q)$	$-(P+Q)$			
$(-P)-(-Q)$		$-(P-Q)$	$+(Q-P)$	$+(P-Q)$

* Carry lookahead adder (CLA)

It solves the carry delay problem by claculating the carry signals in advance, based on input signals.

Based on the fact that a carry signal will be generated in two cases.

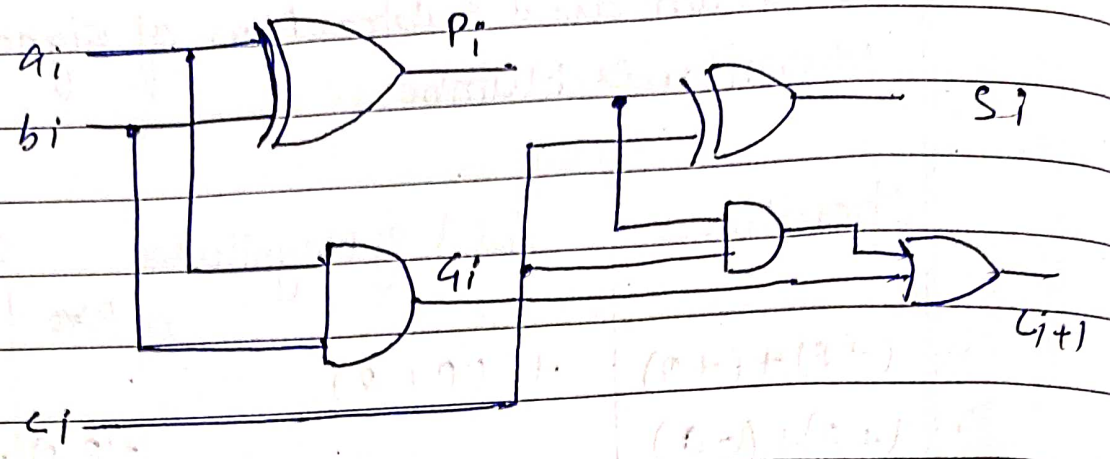
(1) when both bits a_i and b_i are 1 or

(2) when one of the two bits is 1 and carry in is 1

so

$$c_{i+1} = a_i b_i + (a_i \oplus b_i)\, c_i$$

$$s_i = (a_i \oplus b_i) \oplus c_i$$



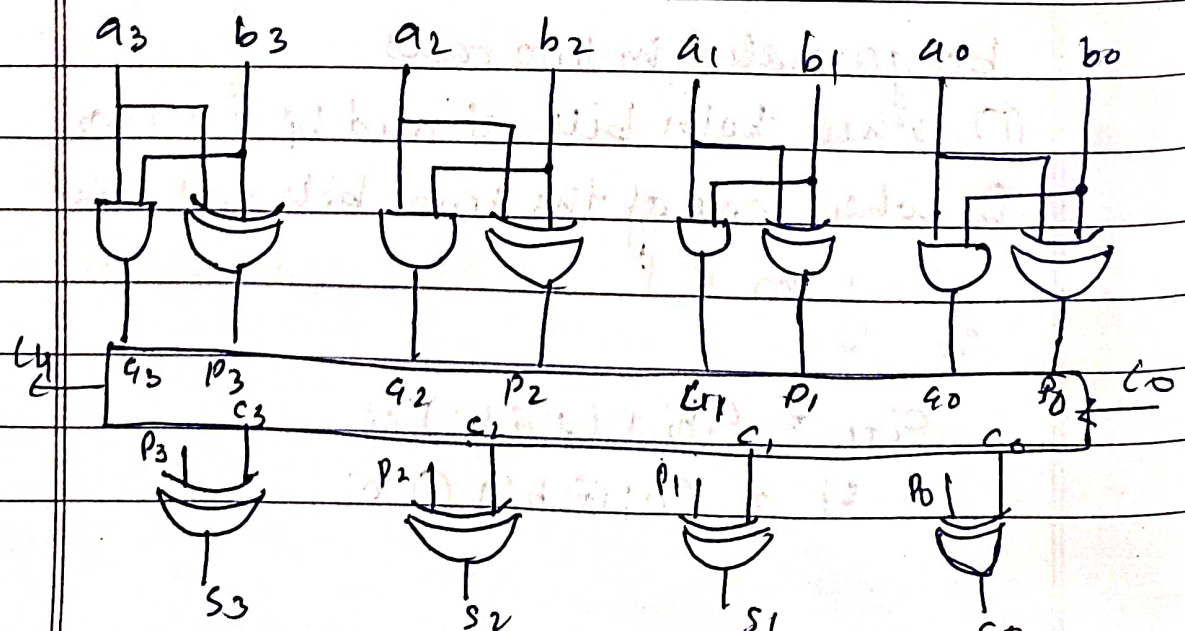
two new signal P_i and G_i called carry generate and carry propogated respectively

$$C_{i+1} = G_i \oplus P_i C_i$$
$$S_i = P_i \oplus C_i$$

$$G_i = a_i b_i$$
$$P_i = a_i \oplus b_i$$

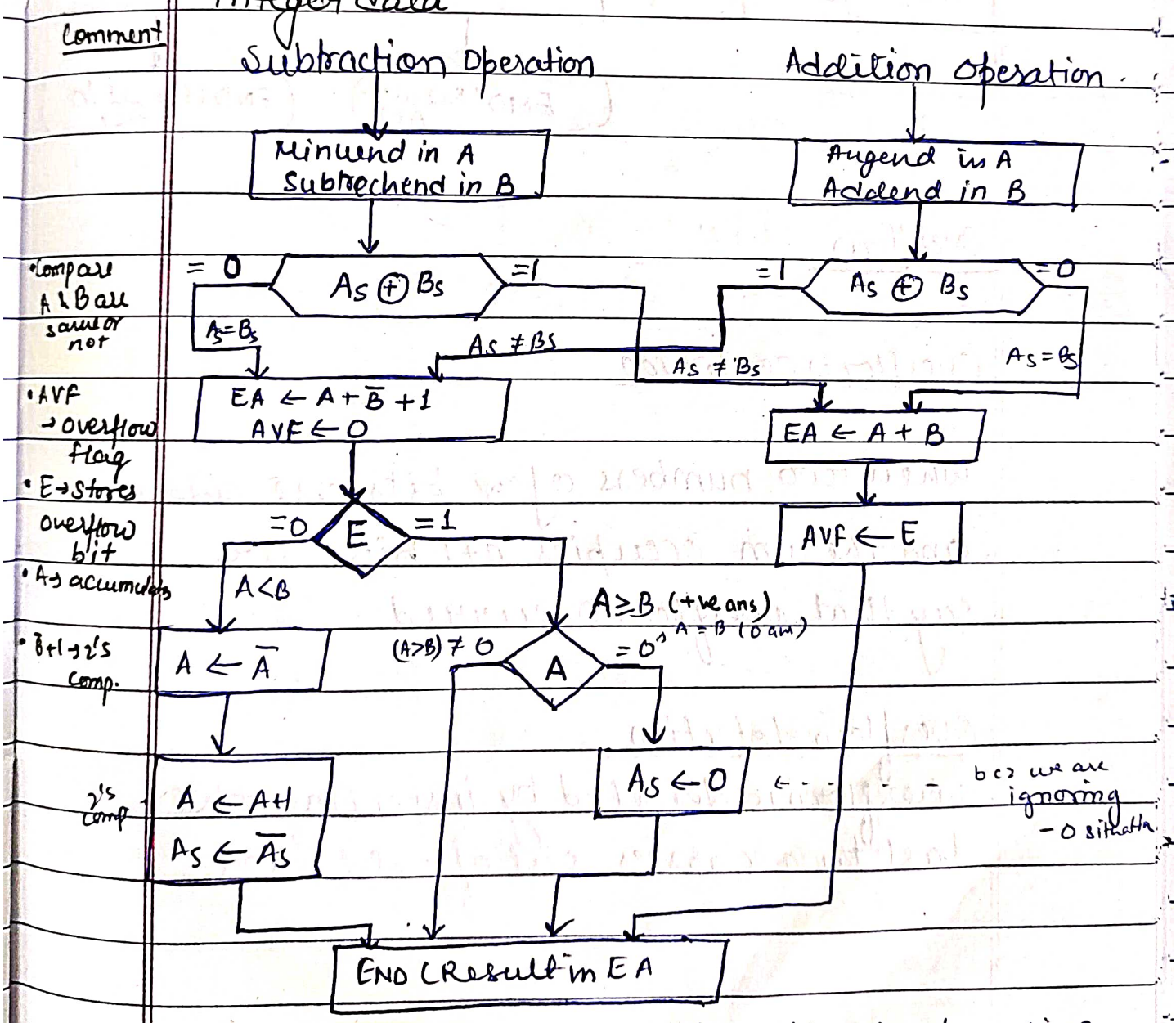
Put $i = 0, 1, 2, 3$

$$C_1 = G_0 + P_0 C_0 \qquad C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0$$



The disadvantage of CLA is that the carry logic block gets very complicated for more than bits.
For that reason, CLA's are usually implemented as 4-bit modules and are used in a hierarchical structure to realize adders that have multiples of 4-bits.

Algorithm for Addition and subtraction of signed integer data.

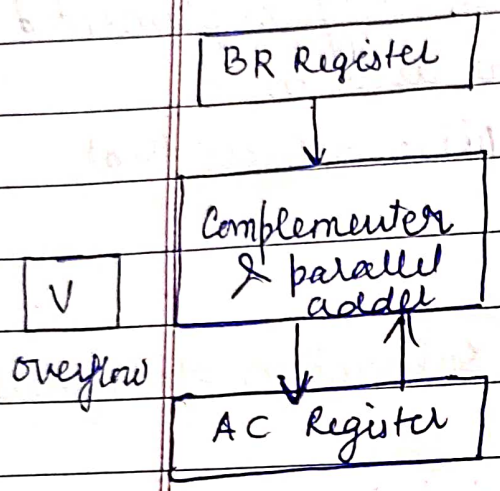


Flowchart of Add & subtracte operation

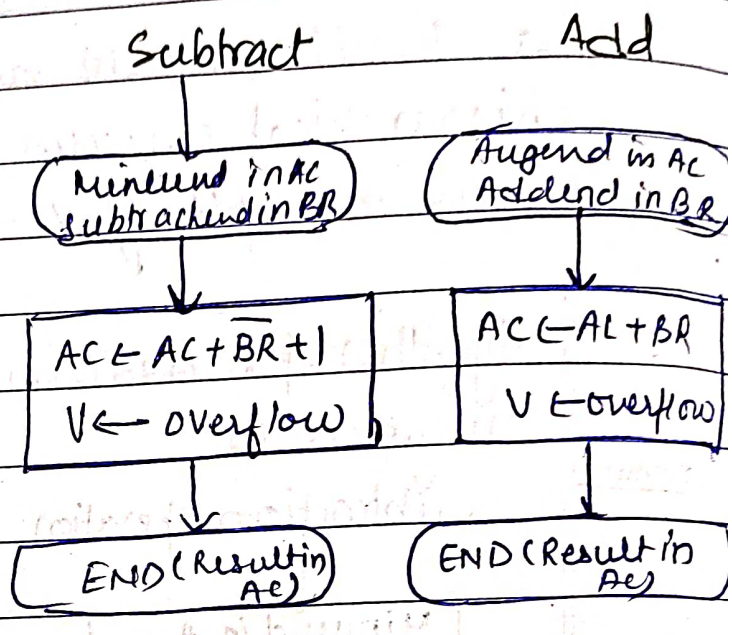
Date.
Page No.

* 2's Complement Addition and subtraction

Hardware



Flowchart



Overflow

Overflow condition

When two numbers of 'n' bits are added and the sum occupies n+1 bits, we say that overflow occurred.

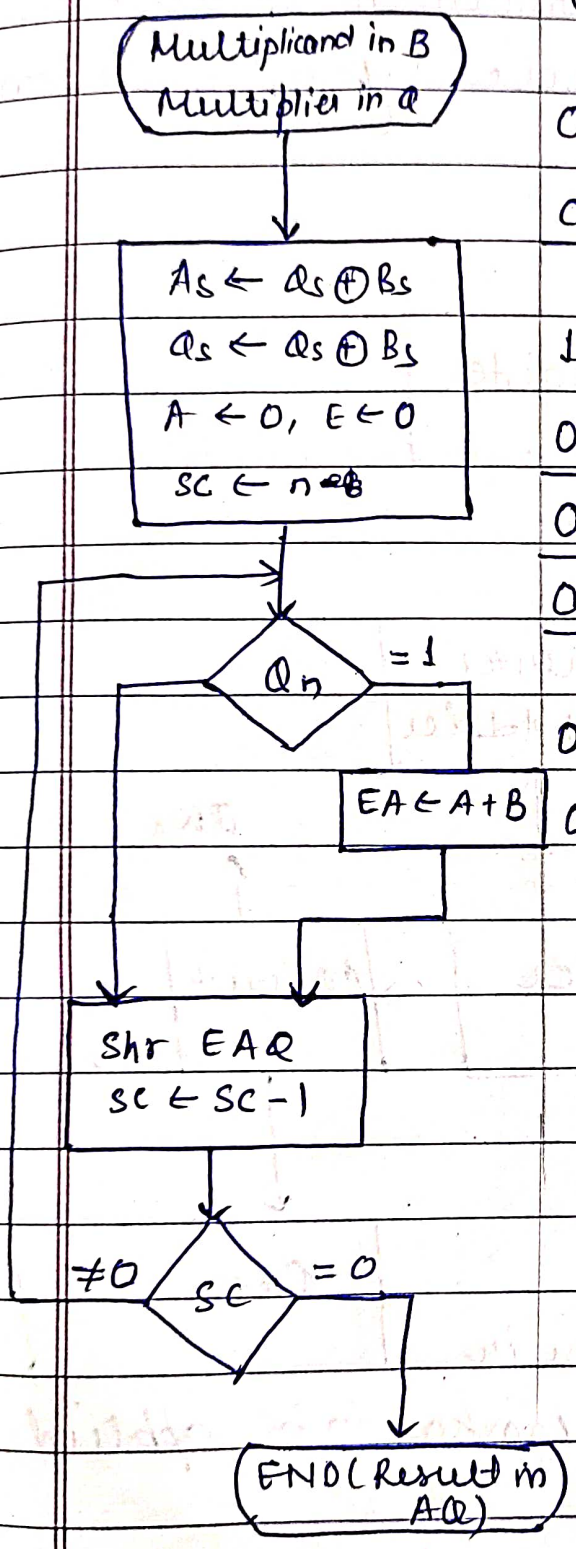
Overflow detection

Overflow is detected by inspecting the last two carries out of addition.

Multiplication of Signed data

B = 10111

Flow chart



E	A	Q	SC
0	00000	10011	101
	+10111		
0	10111	10011	101
0	01011	11001	
	10111		100
1	00010	11001	
0	10001	01100	
0	01000	10110	011
0	00100	0101(1)	010
	10111		001
0	11011	01011	
0	01101	10101	000

Final answer.

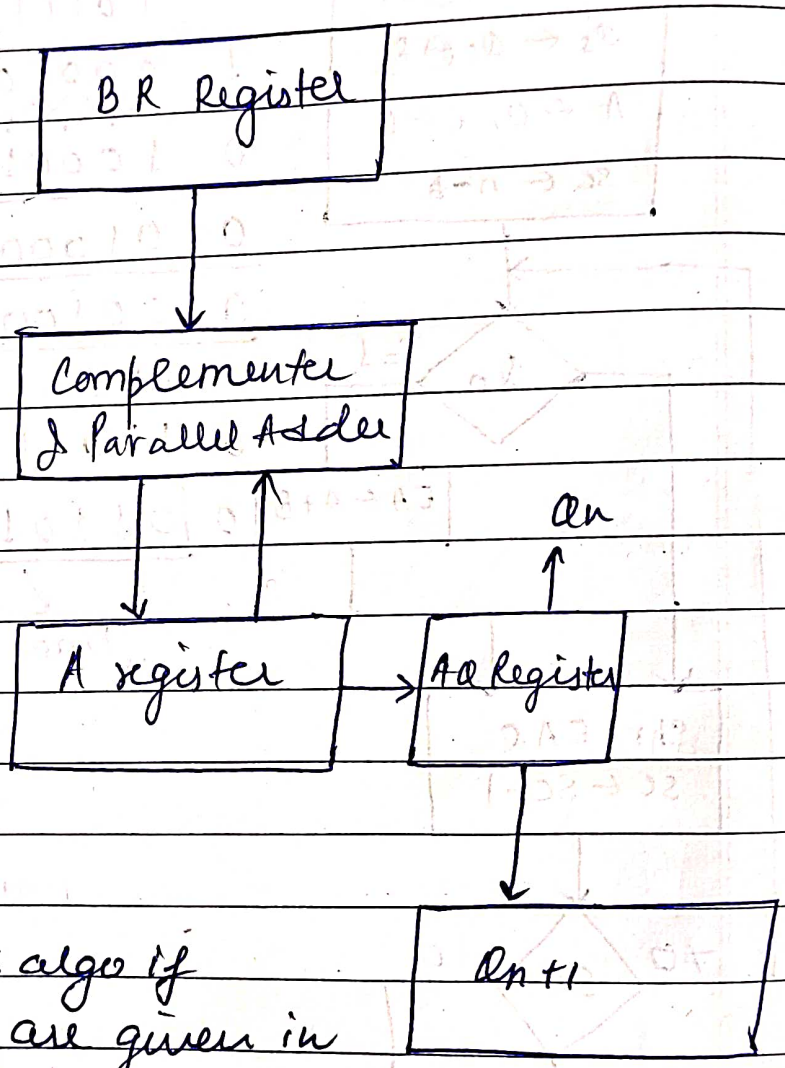
Important

Q_n →	last bit of Q
Shr →	shift right
SC →	Sequence Counter. When SC = 0 final result get

Booth's Algorithm for Multiplication

Booth's multiplication algorithm is a multiplicati algorithm that multiplies two signed binary numbers in two's complement notations. Algorithm was invented by Andrew Donald Booth in 1950.

Hardware



Important

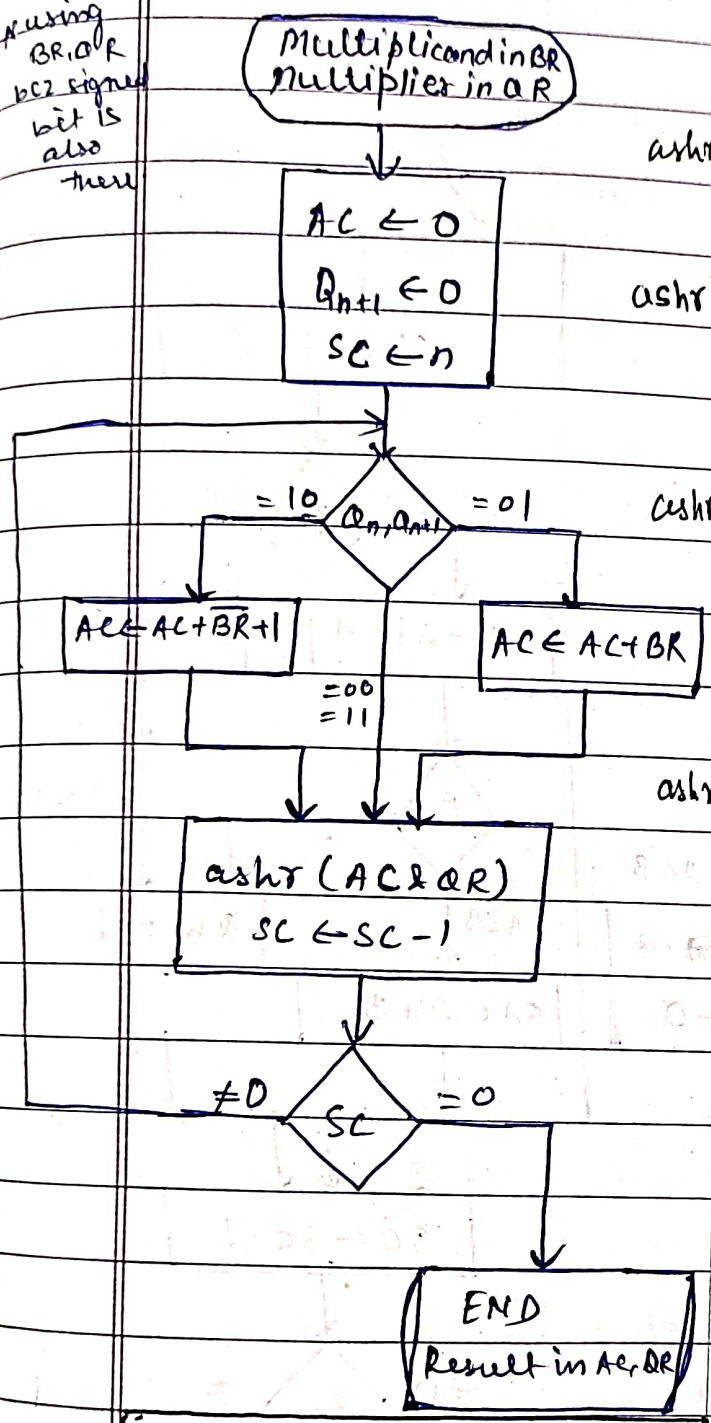
* In Booth's algo if numbers are given in binary simply algo has to be applied (Booth)
* But if no. in decimal then need to find 2's complement of -ve no. then carry on with booth's algorithm.

Date.
Page No.

Booth's Algo for Multiplication.

BR = 10111 , $\overline{BR}$ = 01000 , $\overline{BR}$+1 = 01001

Flowchart



* using BR, QR bcz signed bit is also there

	AC	QR	Q_{n+1}	SC
	00000	10011	0	101
	01001			
ashr	01001	10011	0	
	00100	11001	1	
ashr	00010	01100	1	100
	101111			011
	11001	01100	1	
ashr	11100	10110	0	
	11110	01011	0	010
	01001			
	100111	01011	0	
ashr	00011	10101	1	000

Final answer.

Important

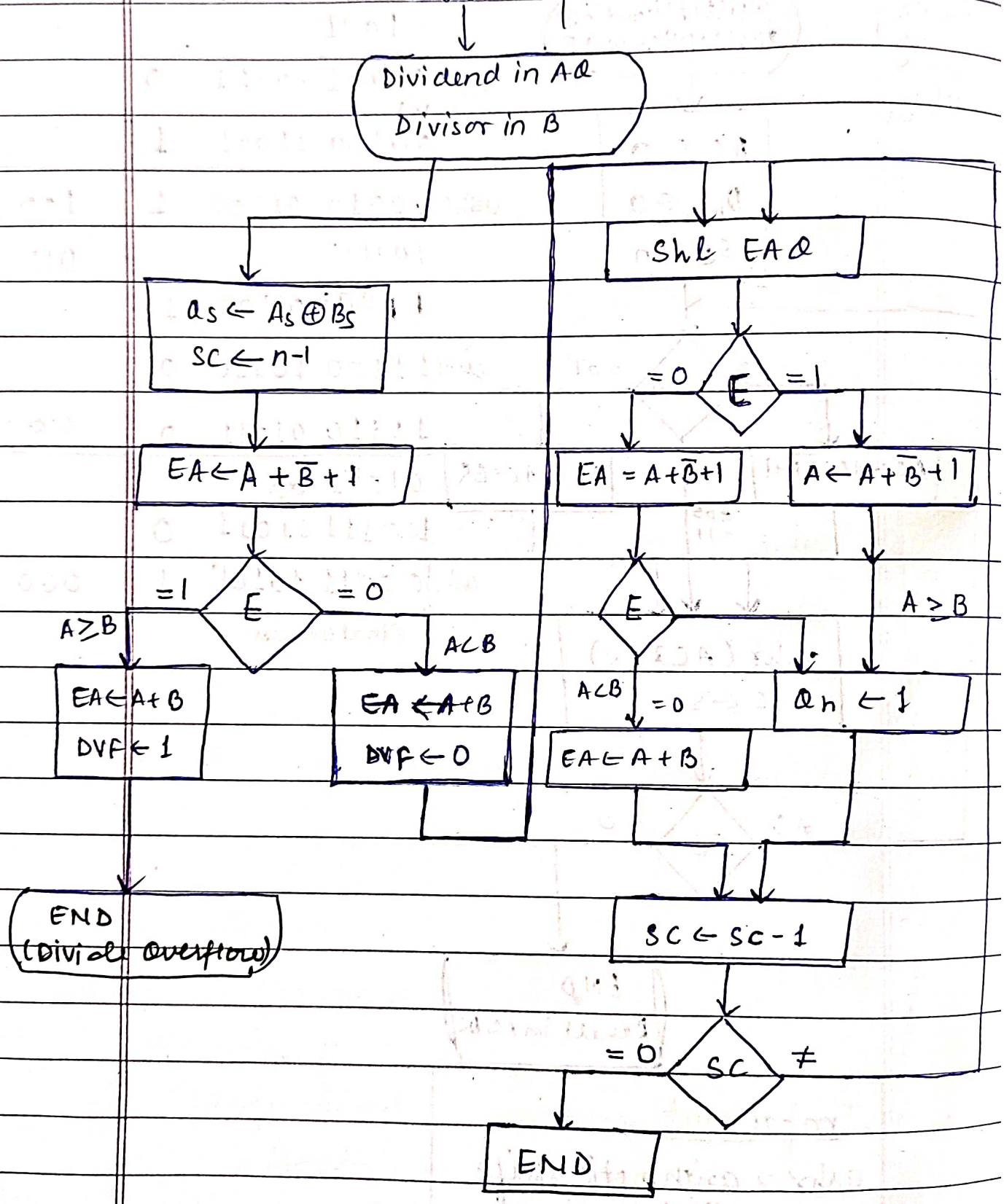
ashr → arithmetic shift right

SC → sequence counter.

no. of steps = no. of bits

Date.
Page No.

Algorithm/Flowchart for division.



Date.
Page No.

IEEE -754 For Floating point Representation

- The IEEE - 754 is a technical standard for floating-point computation.

Ex: 123456×10^0
1.23456×10^5
Normalized number

The IEEE 754 has 3 basic components.

IEEE-754 representation for floating point

S	E	M

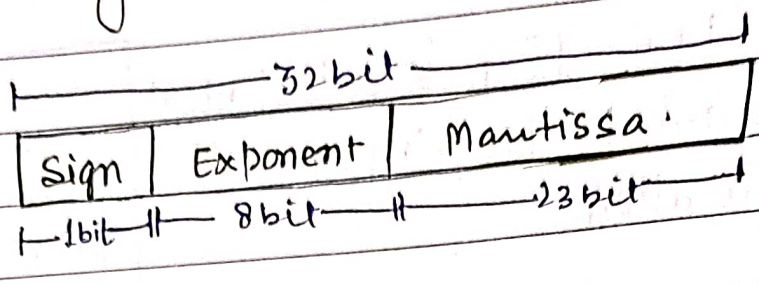
S = sign.
E = exponent
M = mantisa.

1. The sign of Mantissa
This is as simple as the name. 0 represents a positive number. 1 represent a negative number.

2. The Biased exponent
The exponent field needs to represent both +ve and -ve exponents. A bias is added to the actual exponent in order to get the stored exponent.

3. The Normalized Mantissa.
The mantissa is part of a number in scientific notation or a floating-point number, consisting of its significant digits. Here we have only 2 digits, ie 0 and 1, so a normalized mantissa is one with only one 1 to the left of the decimal.

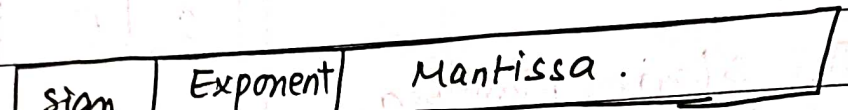
Single Precision.



127 0111111

+ 110

$(-1)^S \times (1.m) \times 2^{E-127}$ → Single $\boxed{10000101}$ Precision.



$(-1)^S \times (1.m) \times 2^{E-1023}$ → Double Precision.

1023 → 0111 1111111

Ex:- $+(105.2)_{10}$ represent the number in single precision floating point IEEE-754.

→ 1101001.0011

Normalize.

→ 1.1010010011×2^6.

Compare with $(-1)^S \times (1.m) \times 2^{E-125}$

S	E	M.
0	10000101	10100100110000 ..

IEEE representation.

$6 = E - 127$

$E = 127 + 6 = 133 = 10000101$

128 16 8 4 2 0

Date.
Page No.

Double

$6 = E - 1023$

$E = 1023 + 6$

$E = 1029$

$=$

S	E	M
0	1000000101	1010110011000 . . .

$(2)^{11}$

$(2)^6 \times (2)^5$

$(2)^3 \times (2)^2 \times (2)^5$

$8 \times 4 \times 32$

32
32
64
64
128
3

ans 4.040000H convert into decimal

0 00000000 1000000 00000000 0000 0000

S e m

$= 0 \cdot 1000000010000000000000000000000$

$S = 0$

Exponent $= 10000000$

Mantissa $= 1000 \ldots$

S	E	M
0	10000000	1000 . . . . .

$(2)^7$

$(2)^5 (2)^2$

64
$\times 2$
128

$10000000 - 0111111$

$= (-1)^0 \times (1.1) \times 2^{1}$

$1 \times 2^0 + 1$
2

$= (-1)^0 \times (1.1) \times 2^1$

$1 + 0.5$
1.5×2

$= +1.5 \times 2$

$= +(3.0)_{10}$

Date.
Page No.

Assignment-1 (Due-Date - 29/3/22)

1. Division Algorithm.
2. Floating point based examples.)
   → Represent the numbers in single and double precession format.
   (1) $+(102.85)_{10}$
   (2) $-(92.32)_{10}$
   (3) 80900000H
   (4) $(10111000100100 0 \ldots)_2$

(3) Solve using Booth's alg multiplication

(a) $(+2) \times (+6)$
(b) $(+15) \times (-12)$
(c) $(-12) \times (-2)$

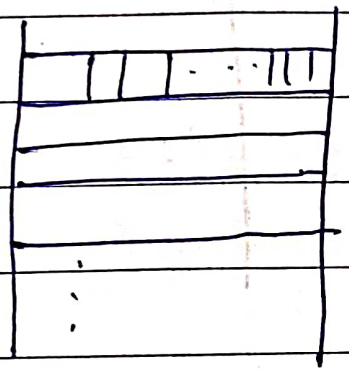
* Class test :- Unit 1

04/3/2024

UNIT-2

→ Machine Instructions and programming
→ Performance of computer

Memory
collection of cells.



→ 1-cell can store ~~[illegible]~~ single bit information.
→ Memory is a collection of cell.
→ Memory may be byte Addressable or word addressable.

Note:-

0/1 → bit
4bit → Nibble
8bit → Byte
16bit → words.

Ex:-

$2^4 \times 2^4$

Date.
Page No.

2^n

	Bits req. to Add. Each words	Bits req. to Add each byte
4096 x 16	12 bit	13 bits
1024 x 32	10 bit	12 bits
258 x 16	8 bit	9 bits
512 x 64	9 bits	12 bits

2x32
2x16x2
2x2x2x8

Little Endian / Big Endian Concepts

k = no. of bits of address lines.

	0	1	2	3
	4	5	6	7
	8	9	10	11
	12	13	14	15
	⋮			
2^k-1	2^k-1	2^k-2	2^k-3	2^k-4

Big Endian

0	3	2	1	0
1	7	6	5	4
2	11	10	9	8
	⋮			
2^k-1	2^k-4	2^k-3	2^k-2	2^k-1

Little Endian.

→ Way of representing data in computer

→ Some computers works on Little Endian & some on Big Endian.

→ Exchange of data Bytes while communicating such two computers is necessary according to little Endian & Big Endian.

(24 32 64 22)
lower higher.

24	32	64	22

Big Endian

22	64	32	24

Little Endian

lower order byte lower Address & Higher order byte at higher address	Lower order byte at higher address & Higher order byte at lower address.

* Instruction

Control word which generate control signals.

Notation

Assembly Language Notation	Register transfer Notation.
	$A \leftarrow A - B$
	$A \leftarrow \bar{A}$
Ex:- ADD A, B	$R1 \leftarrow R1 + R_3$
SUB A, B	
ADD R	

Types of Instruction

1) Data Transfer Instruction.
2) Arithmetic & logical Instructions
3) Branching Instruction.

$R1 \leftarrow A$, $R1 \leftarrow M[A]$ MOV R1, A

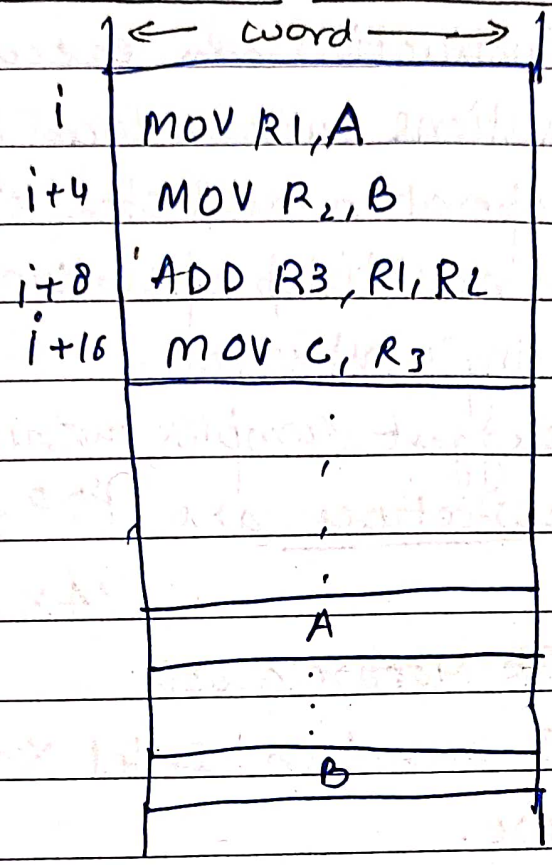
$R2 \leftarrow M[B]$ MOV R2, B

$R_3 \leftarrow R_1 + R_2$ ADD R_3, R_1, R_2

$C \leftarrow R_3$ MOV C, R_3

Date.
Page No.

INSTRUCTION SEQUENCING



it is a byte addressable memory.

PC → [ ]

· stores Address of next instruction.

→ Fetch instruction
→ Decode Instruction
→ Execute Instruction.

Instruction set Architecture (ISA)

- Instruction set Architecture is a set of processor design techniques used to implement the instruction work flow on hardward.

- In more practical words ISA tells you that how your processor going to process your program instructions.

Date.
Page No.

Complex Instruction set (CIS)

- Complex Instruction set computer is a computer where single instruction can execute several low level operations such as load from memory an arithmetic operation and store to the memory are capable of multistep operations or addressing modes within single instructions.

As its name suggest complex instruction sets.

Ex of CIS architectures are PDP-11
VAX

- ~~Mortolola 68k~~ Motorola 68K
- Desktop PC based on Intel X-86 architecture.

Reduce Instruction Set Computer

It is a computer which only use simple instructions that can ~~di~~ be divide int multiple instructions which perform low level operations within single clock cycle. as its name suggest reduced instruction set.

Example of RIS family include

- DEC -α
- AMD - 29K
- ATMLAVR
- ARM processor
- MOTOROLA 88000 processor

Date.
Page No.

CISC	RISC
• CISC processor is a complex instruction set.	• Reduced instruction set computer.
• CISC processor has complex instructions that multiple clocks for execution. They average clock cycle per instruction is in the range of 2 and 15.	• RISC processor have simple instruction taking about 1 clock cycle per instruction CPI is 1.5
• CISC's performance is optimised with more focus on hardware.	• RISC performance is optimised with more focused on ~~soft to~~ software.
• It has a memory unit to implement complex instructions.	It is no memory unit & uses a separate hardware to implement instruction.
• It has a micro programming unit.	• It has a hardware unit of programming.
• The instruction set has a variety of instructions that can be used for complex operations.	• The instruction set is reduced i.e. it has only a few instructions in a instruction set. Many of these instruction are very primitive.

• CISC already supports complex addressing modes	~~The instruction set a~~ • Complex addressing modes are synthesised using the software
• Only has a single register set	• Multiple register sets are present.
• There are normally not pipelining or less pipelined.	• RISC processors are high pipelining.
• Execution time is very high	• RISC execution time is very less.
• code expansion is not a problem.	• Code expansion can be a ~~prot~~ problem.
• CISC architecture is used in low end application such as security systems, ~~homation~~ homodonation.	• RISC type architectures are used in highend application such as video processing, telecommunication and image processing.

Date.
Page No.

ques:- RISC vs CISC

Ans: we can not differentiate RISC & CISC technology both are suitable at its specific application. what counts is how fast a chip can executes the instruction it is given and how well it runs existing software. Today both RISC & CISC manufacturers are Doing to get an edge of competition.

Addressing modes.

Addressing Modes Specifies a rule for interpreting or modifying the address field of the instruction. (before the operand is actually referenced).

Types of Addressing Modes.

Variety of addressing modes are used to give programming flexibility to the user and to use the bits in the ~~user~~ & address field of the instruction efficiency.

Opcode	Address field	Address field

- Address of the operands are specified implicity in the defination of the instruction.

* No need to specify the address in the instruction.

Ex:- CMA, RRA

Date.
Page No.

Immediate Addressing Mode

- ~~The~~ Instead of specifying the address of the ~~The address of the~~ operand itself is specified
- No need to specify address in the instruction However operands itself needs to be specified Sometimes requires more bits than the address.
- Fast to acquire the operand.
- Useful for initializing registers.

Register Addressing Mode

- The address is specified in the instruction is the register address.
- Designated operands are need to be a register shorter address then the memory
- ~~shorter~~ address.
- Faster to acquire an operand than the memory addressing.

Register Indirect

- Instruction specifies the register which contains the memory of the operand.
- Since register instruction

Ex:- ADD M
MOV A, M

Date.
Page No.

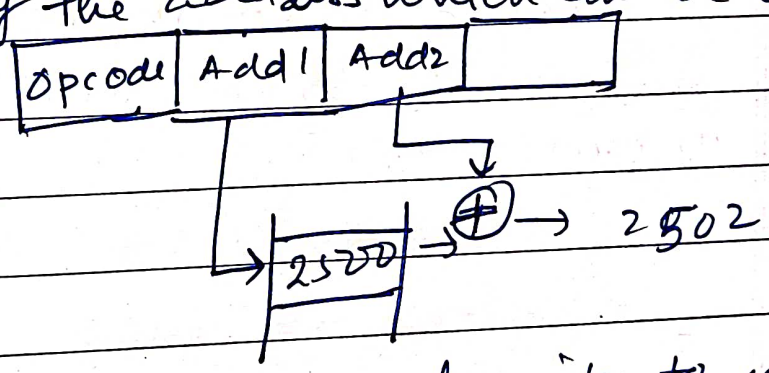
Direct Addressing

- Inst. specifies the memory address which can be used directly to access the memory
- operand resides in memory and its address given in inst used to access table of data in memory.

Ex:- LDA 2000H.

Relative Addressing

- the address field of an instruction specifies the part of the address which can be used



along with the designated register to calculate the address of operand.

- Address field of instruction is short.
- large physical memory.

3 different relative addressing mode.

1. PC relative addressing mode.
2. Index addressing mode.
3. Base register addressing mode.

→ Types of Instructions

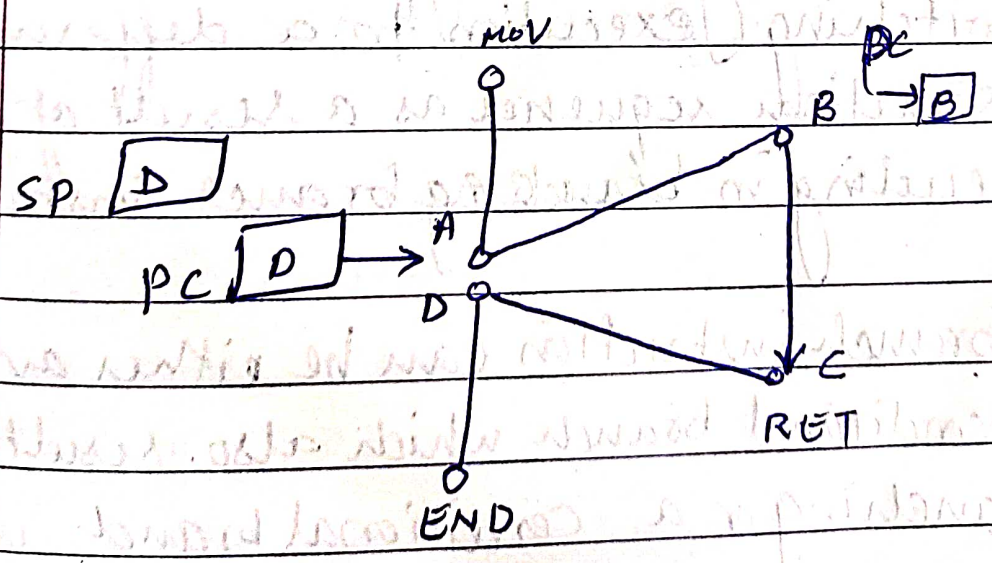
1. Arithmetic & logic Instruction.
2. Data Transfer Instruction.
3. I/O Transfer
4. Branching & subroutine Instruction.

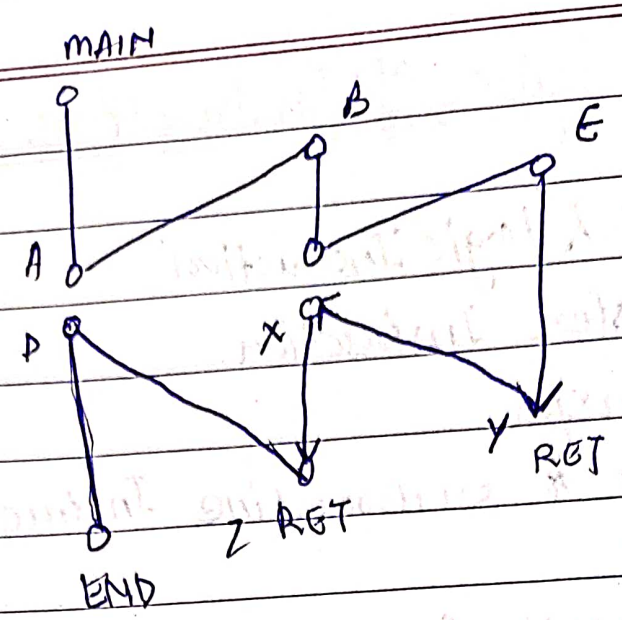
Code:

Ex:

```
HERE : MOV  A, #55H
       MOV  P0, A
       CALL DELAY
       MOV  A, #AA
       MOV  P0, A
       CALL DELAY
       JMP  HERE

DELAY : MOV R0, #255
HERE1 : DJNZ R0, HERE1
        RET
```





Branch & subroutine

- A Branch is an instruction in a computer program that can cause a computer to begin executing ~~an~~ different instructi sequence and does deviate from its default behavior of executing instruction in order.

- Branch may also reffered to the act of switching execution to a different instruction sequence as a result of executing in ~~branching~~ branch instruction

- A branch instruction can be either an unconditional branch which also results in branching or a conditional branch which may or may not cause branching depending on some conditions.

- Branch instruction are use to implement control flow in program. loops and conditionals

Ex:- JUMP
DJNZ
JNC
JC

Subroutines

- In computer programing a subroutine is a sequence of program instructions that performs a specific task packaged as a unit.
- This unit can then we used in program where ever that particular task should be performed.
- Subprograms may be defined with in a program or separately libraries that can be used by multiple programs.
- In different programming languages a subroutine may be called a procedure, a function, a routine, a method or a subprogram. The generic called callable unit is sometime used. The name subprogram suggests a subroutine behaves in must the same way as a computer program le used as one

Date.
Page No.

step in a larger program or another sub-program.

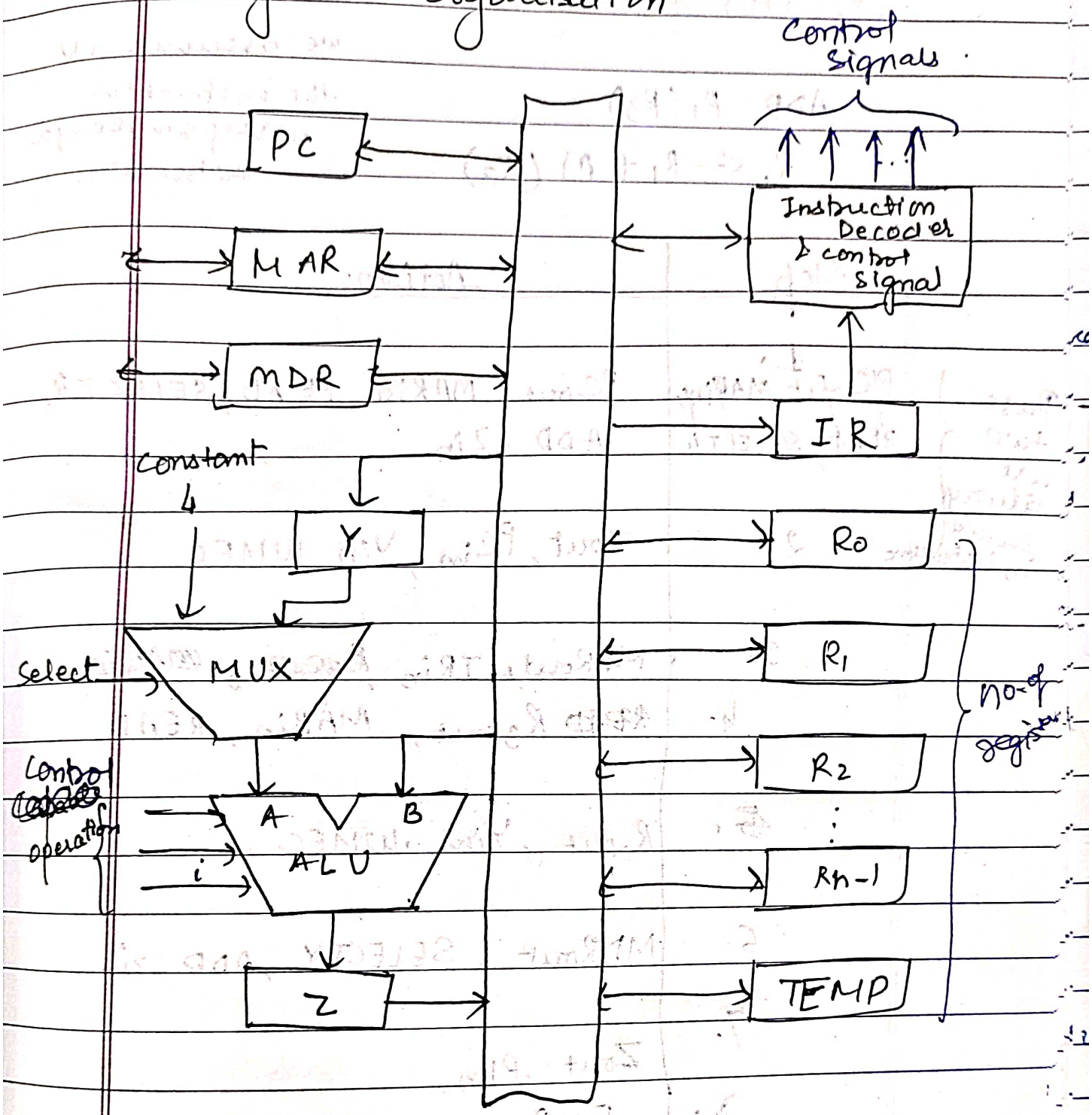
Ex:- CALL, RET

Disadvantage

- Invoking a subroutine imposes some computational overhead in the call mechanism.

- A subroutine typically requires standard housekeeping codes both at ~~entering~~ entering to, and exit from, the function (usually saving general purpose registers and return address as much as minimum

Single bus Organisation.



MAR - Memory address register.

MDR - Memory Data registers.

IR - Instruction Register.

Signals :

→ READ · PCin · WMFC → wait for memory function to complete

→ WRITE · PCout

→ END

Date.
Page No.

Execution of complete Instruction.

we assume all the instruction carry single byte instruction.

ADD $R_1(R_3)$

$R_1 \leftarrow R_1 + M(R_3)$

Step	Action.
1.	PCout, MARin, READ, SELECT4, ADD, Zin
2.	Zout, PCin, Yin, WMFC
3.	MDRout, IRin
4.	R_3out, MARin, READ
5.	R_1out, Yin, WMFC.
6.	MDRout, SELECTY, ADD, Zin
7.	Zout, R_1in,
8.	END

These two are always same in instructions

Date.
Page No.

Q:- Three Address Instruction

$X = (A+B) * (C+D)$

ADD R1, A, B R1 ← A+B.

ADD R2, C, D

MUL X, R1, R2 X ← R1 * R2

Two Address

MOV R1, A

ADD R1, B R1 ← A+B.

MOV R2, C

ADD R2, D R2 ← C+D

MUL R1, R2

MOV X, R1

One Address

→ Use of Accumulator

LOAD A ; ACC ← A

ADD B ; ACC ← ACC + B.

STORE T ; T ← ACC ← A+B.

LOAD C ACC ← C

ADD D ACC ← C+D

MUL T

STORE X

Date.
Page No.

Zero Addressing (using stack)

$= (A+B) * (C+D)$

$= AB + CD + *$

B	
A	A+B

PUSH A

PUSH B

ADD

~~POP~~ PUSH C

PUSH D

ADD

	D
C	C
A+B	A+B

MUL

POP X

C+D	
A+B	(A+B)*(C+D)

Exp (1) $(A * B) + (C * D) + (E * F)$

(2) $(A * B) + (A * (B * D)) + (C * E)$

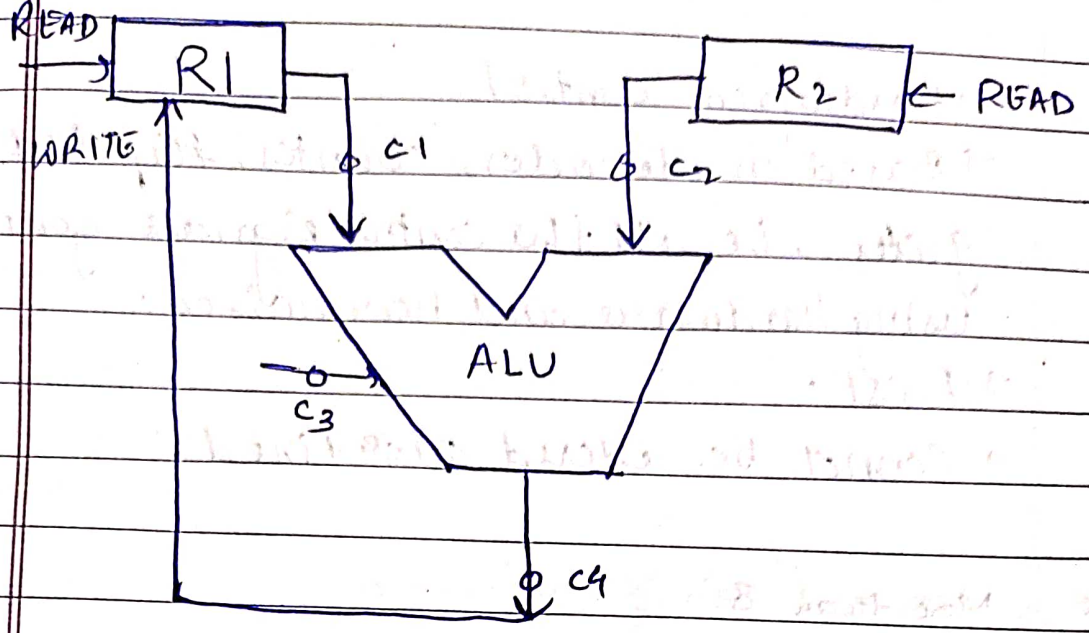
(3) $[A * [B + C * (D + E)]] / [F * (G + H)]$

(4) $(A + B * C) / (D - E * F + G * H)$

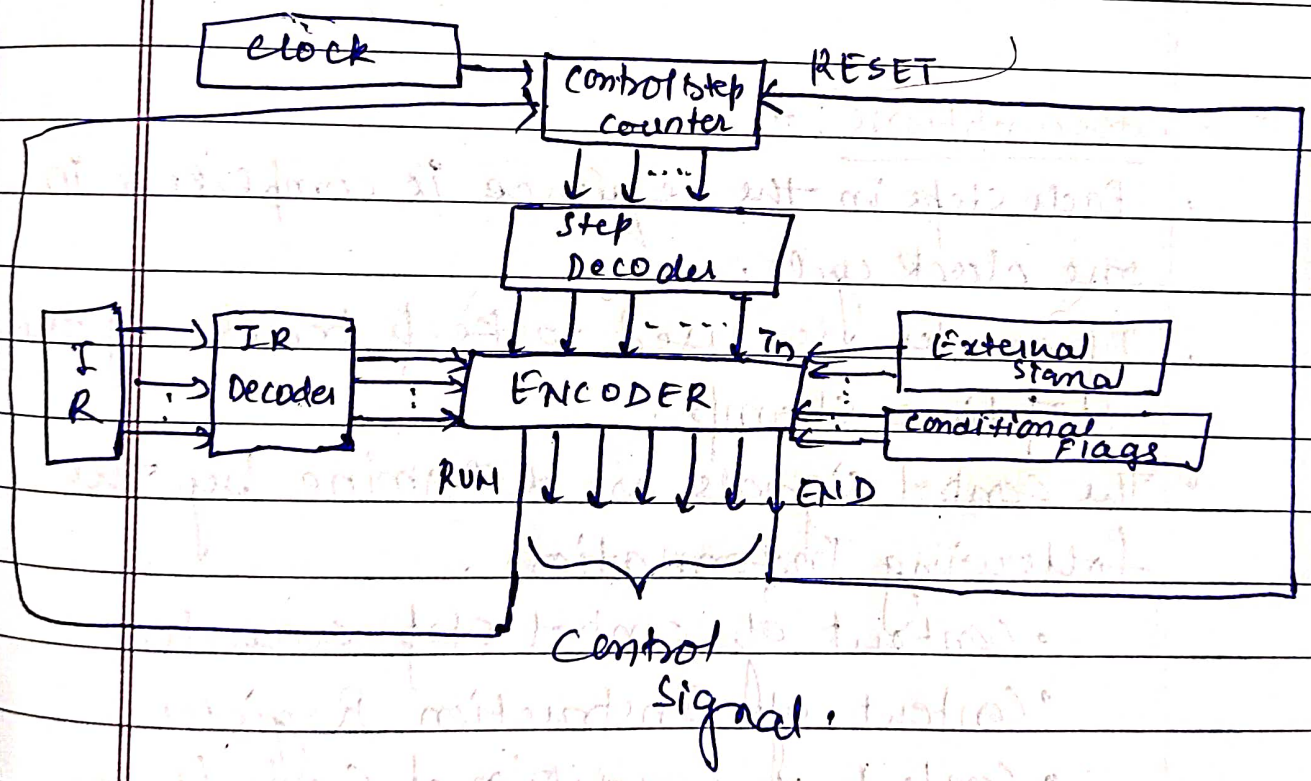
(5) $A + B * [C * D + E * (F + G)]$

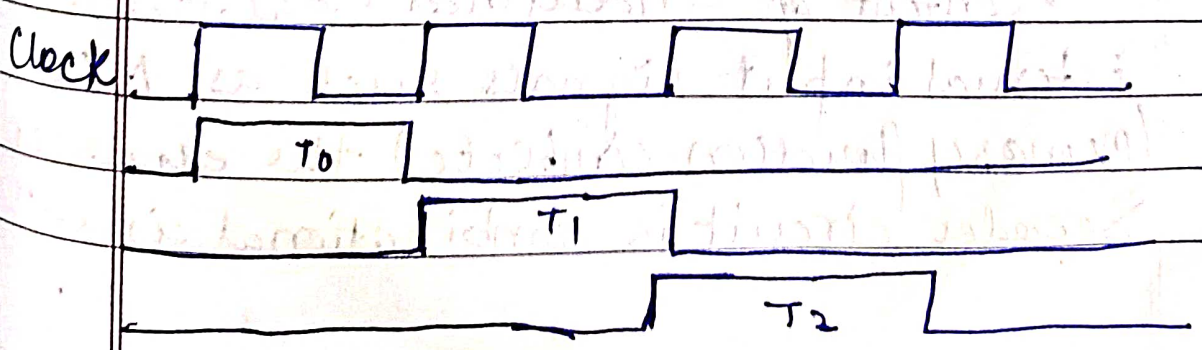
(6) $(A + B - C) / (D * (5 * F - G))$

ADD R1, R2.



STEPS of sequence is followed according to instructions.





Control Unit Design

→ Hardwired Control
→ Based on decoder, counter, flip flops, gates, i.e all the control signals generated using hardware and hardwired.
→ Fast.
→ Cannot be altered once fired.

* More than 8.

~~Steps~~
Hardwire Control Unit design

1. Assumptions :-
- Each steps in the sequence is completed in one clock cycle.
- If counter is used to keep track of the step time stamp.
- The control signals are determine by the following Information.
  - Content of control step counter
  - Content of Instruction Register
  - Content of conditional Code flags.
- External input signals such as MFC (memory function complete) the encoder Decoder circuit is combinational circuit

which generates control signals depending on the inputs provided.

- The step decoder generates separate signal lines for each steps in the control sequence ($T_1, T_2, T_3 \ldots T_n$)

  - Depending on maximum steps required for an instructions, the step decoder is designed.

  - If a maximum of ten steps are required then a 4 x 16 step decoder is used.

- Among the total sets of instructions the instruction decoder is used to select one of them.

  - If a maximum of 100 instructions are present in the ISA (Instruction Set Architecture) then a 7 x 128 instruction decoder is used.

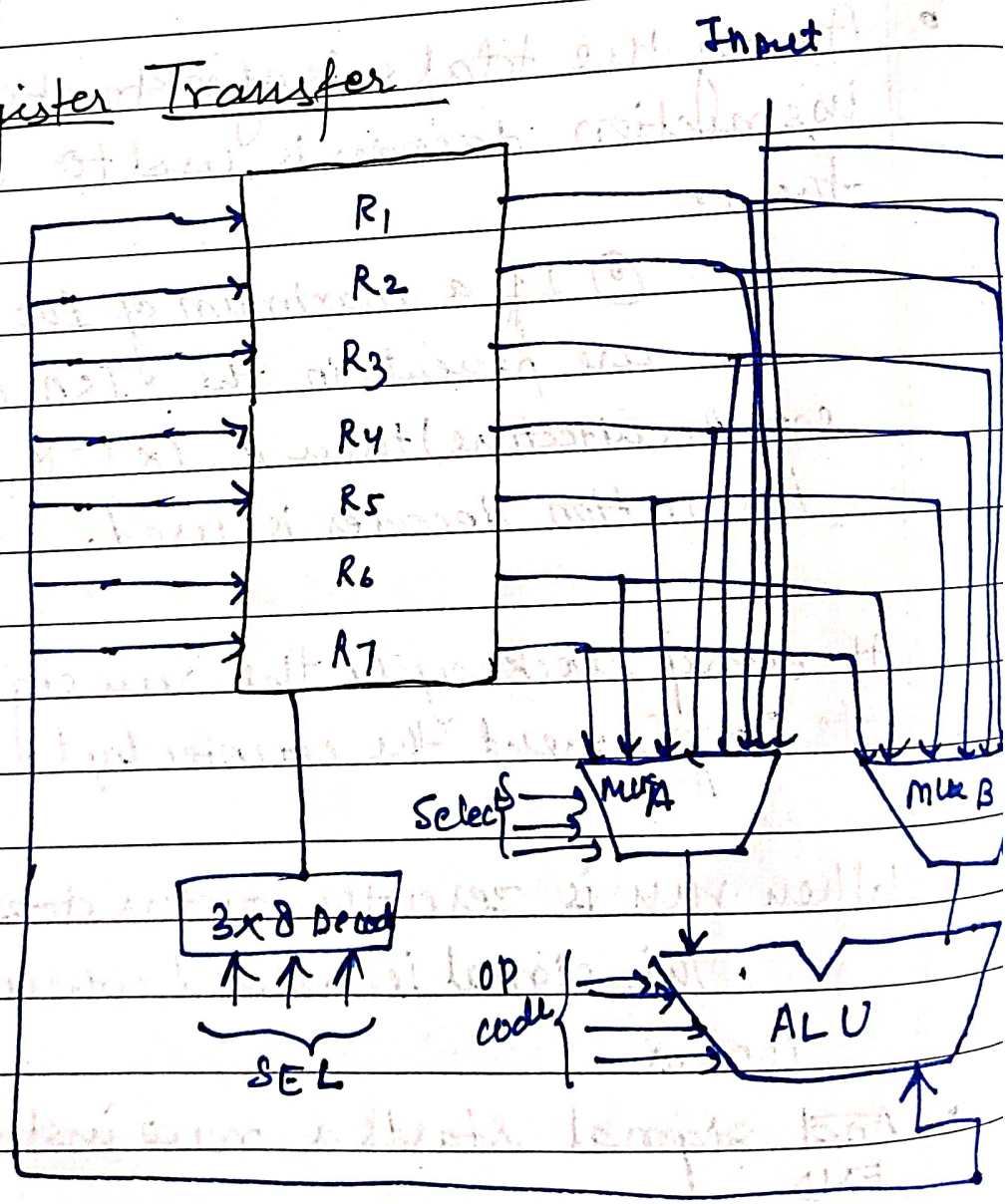
- At every clock cycle the run signal is used to implement the counter by 1.

- When run is zero the counters stops counting.
  - This signal is needed when WMFC is issued.
- End signal starts a new instruction.
- END

It resets the control step counter to its starting value.

- The sequence of operations carried out by the control unit by sequence of operation is determined by the wiring of the logic elements. and ends it is named hardwired. This approach of control unit design is fast but limited to the complexity of instruction set that is implemented.

Register Transfer



BIN	SELA INPUT	SELB INPUT	SELD INPUT
000	R1	R1	R1
001	R2	R2	R2
010	R3	R3	R3
011	R4	R4	R4
100	R5	R5	R5
101	R6	R6	R6
110	R7	R7	R7
111			

Microinstruction:

SELA	SELB	SELD	OPCODE
3bit	3bit	3bit	5bit

OPCODE	OPERATION
00000	TRANSFER A
00001	Increment A
00010	ADD A+B
00101	Sub A-B
00100	Decrement A
01000	Ending of A and B
01010	OR A B
01100	EXOR A and B
01110	Compliment A
10000	Shift right A
11000	Shift left A

select