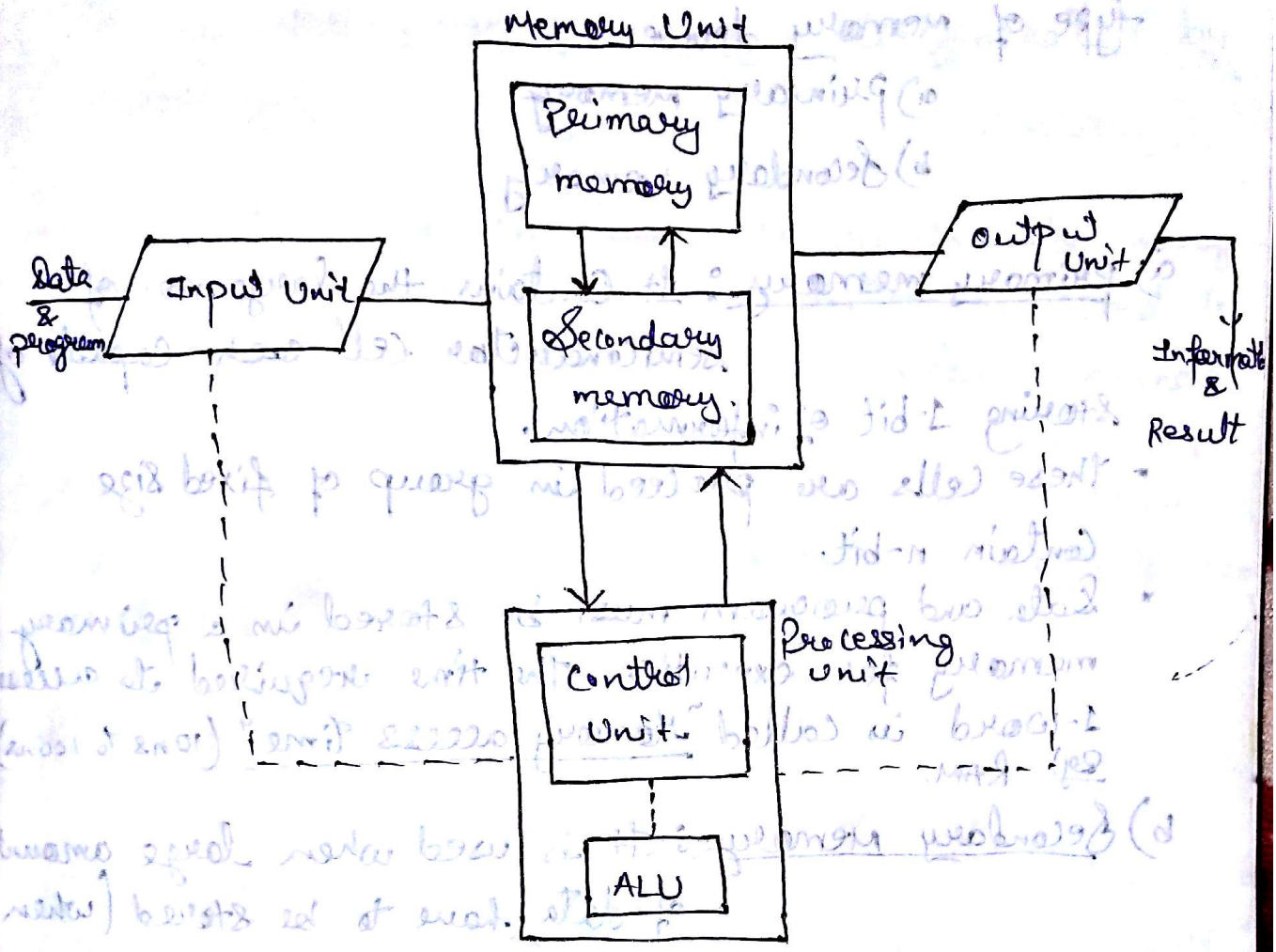
Functional Unit of Computer



The internal architecture of a Computer differ from one system model to another. A block diagram of the basic Computer organization specifying different functional unit shown above in fig:

Here the solid line indicate flow of instruction and data where as doted line represent the control exercise done by control unit.

Following functional unit :-

1) Input unit :

The input unit accepts coded information from human operater through electromechanical devices such as keyboard, mouse, joystick, trackball, Scanner, etc

ii) Memory Unit:

It stores the program and data. There are two type of memory devices -

a) primary memory
b) Secondary memory

a) primary memory: It contains the large no. of semiconductor cell each capable of storing 1-bit of information.

- These cells are proceed in group of fixed size contain n-bit.
- Data and program must be stored in a primary memory for execution. The time required to access 1-word is called "Memory access time" (10 ns to 100 ns)

eg) RAM

b) Secondary memory: It is used when large amount of data have to be stored (when frequently access is not required)

eg) Hard disk, floppy disk, Magnetic disk, CD, etc.

iii) Processor Unit:

It is a heart of computer system, it consists of ALU and control unit.

a) ALU: Mostly computer operations related to the arithmatics (gates) are executed by the help of ALU.

b) Control unit: The operations of all the units are co-ordinated by the control unit (CU). And the timing signal that

govern the input and output transfer are generated by control unit.

The synchronization of signal is also generated by control unit.

NOTE: The control unit and ALU are usually many times faster than other devices connected to the computer system. This enables a signal processor to control a no. of external devices such as video terminal, magetic tape, disk memory, sensor and mechanical control which are much slower then processor.

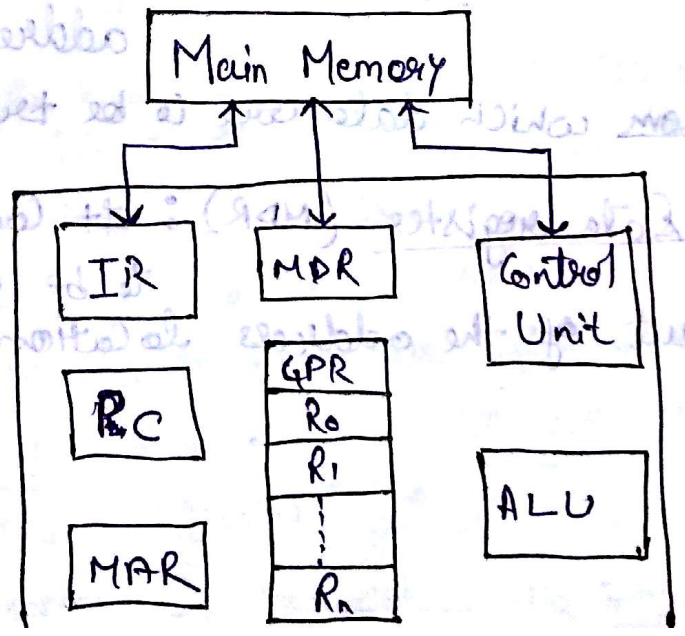
(iv) Output unit :-

It is a counter part of input unit.

Output devices accept binary data from the computer, decode it into original form and supply this result to the outside world.

eg printer, monitor, video terminal, graphical display

Internal Structure of Processor



Processor contains a no. of register used for temporary storage of data other than ALU and Control unit. A processor contain different type of registers, these are followings —

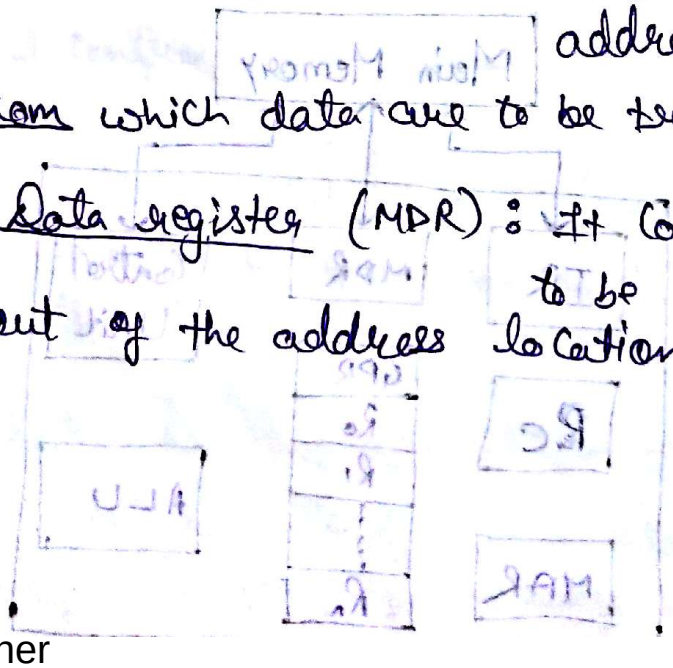
1) Instruction register : It hold the instruction that is currently being executed, its output is available to the Control Circuit which generate the timing signal that control the various processing element involving in executing the instruction.

2) Program Counter register (PC) : It contain the address of the instruction currently being execute during the execution of the instruction the content of PC are updated to hold the address of the next instruction to be executed.

3) General Purpose register (GPR) : The GPR facilitate communication with the main memory. Access data in these register is much faster than data stored in main memory. because these registers are inside the processor.

4) Memory Address register (MAR) : It hold the address of the location to or from which data are to be transferred.

5) Memory Data register (MDR) : It contain the data to be written or read out of the address location.



Bus:

A group of wires that connects several devices to carry data or information is called a bus.

A bus is a common path that connects a no. of devices.

A computer has many register and path must be provided to transfer the information from one register to another. This transfer is done by the bus. If separate lines are used between each register and all other devices, the no. of wires will increase excessively and network will be costly and complex.

To shortout above problem, a scheme for transferring information between register is ~~common~~ solved by using common bus system.

Type of Buses

1) Data bus:

* It is a bidirectional bus in nature.
* Its function is to carry data from one system unit to another.
* The data bus consists of 8, 16, 32 or more parallel lines. This means that CPU can read data on these lines from memory or from a port as well as data transfer on these lines to a memory location or a port.

2) Address Bus:

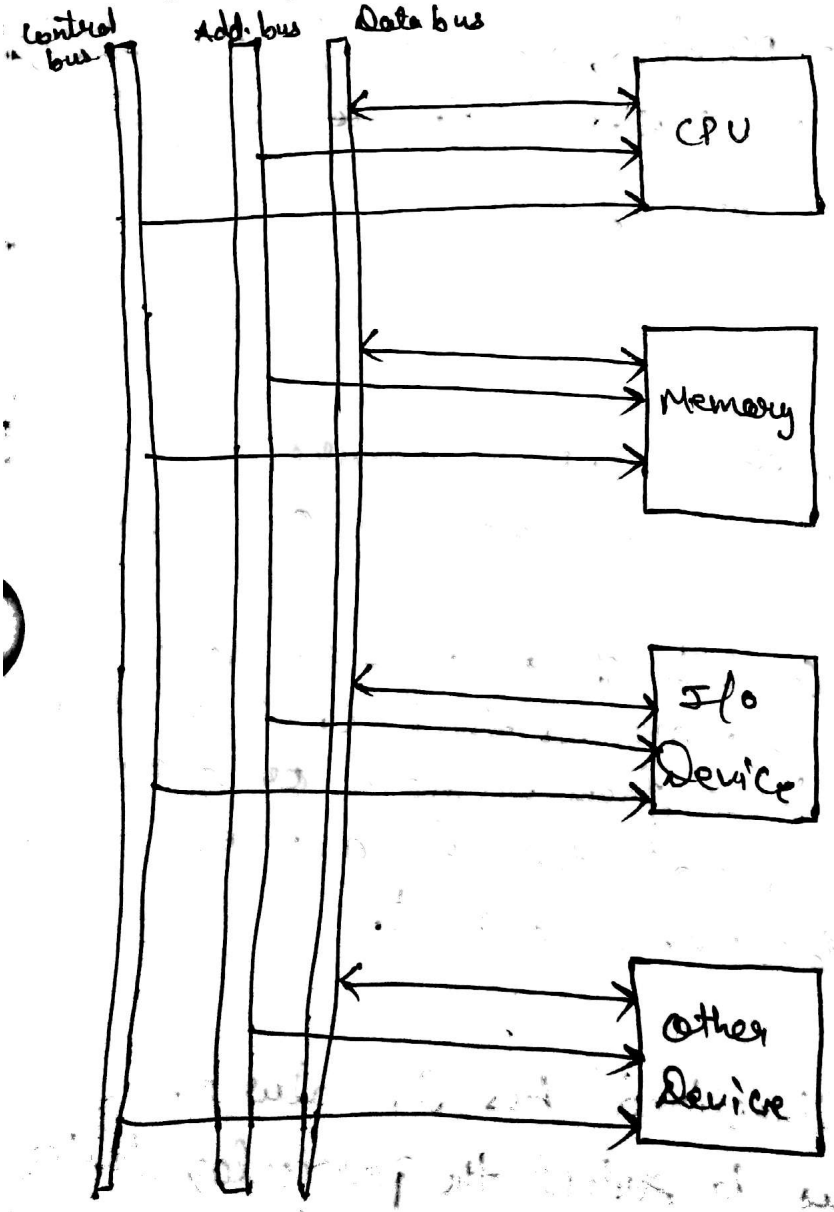
* It is the unidirectional bus in nature.
* Its function is to select the particular device

or memory location for transfer or receiving data.
- The address bus consists of 16, 20, 24, or more parallel lines

* The CPU send out the address of the memory location or I/o port that is to be written or read from by using address bus.

3) Control Bus :

* It is also unidirectional in nature.
* Its function is to control the units based on their request.
* The CPU sends singal on the control bus to enable the output of address memory device or port device.



Bus Designing Parameter

1) Mode of bus - dedicated or multiplexed.
2) Method of arbitration - centralized or distributed.
3) Timing - Synchronous or asynchronous.
4) Bus width - 8, 16, 32 bit.
5) Data transfer type - Read, write, read after write, Block, etc.

1) Mode of bus

a) Dedicated bus : In this mode, a bus is permanently assigned to one function or to a physical subset of a Computer Component. Dedicated mode is simple but it increase size and cost when there is a more functions.

b) Multiplexed bus : In this mode, a bus is assigned for more than one function at a different time zone.

2) Method of arbitration

a) Centralized arbitration : In this approach, a single hardware device referred as a bus controller or arbitrater and it is also responsible for allocating time on the bus for the bus masters.

b) Distributed arbitration : In this approach, each bus master contains access control logic and they act together to share the common bus.

3) Timing:

a) Synchronous: In Synchronous timing, the occurance of event on the bus is determined by the CPU clock. A ~~Synchronous~~

b) Asynchronous: In asynchronous timing the occurance of event on the bus is determined by the previous event on the bus.

4) Bus width:

The data bus width decides the no. of bits transfer at one time (16, 32, bit) whereas address bus decides the range of location that can be accessed.

5) Data transfer type:

Read, write, Read after write, block, etc.

Bus Arbitration:

A mechanism which decide the selection of current master to access bus is known as bus arbitration.

When more than one device want to communicate with each other simultaneously or we can say that one device want to be a bus master at the same time. But as we know only single device may became a bus master - then how will be decision made who will be the bus master? this problem

(5) is solve by bus arbitration method. These methods are following

i) Daisy chain Method
ii) Parallel arbitration Method
iii) Independent arbitration Method.

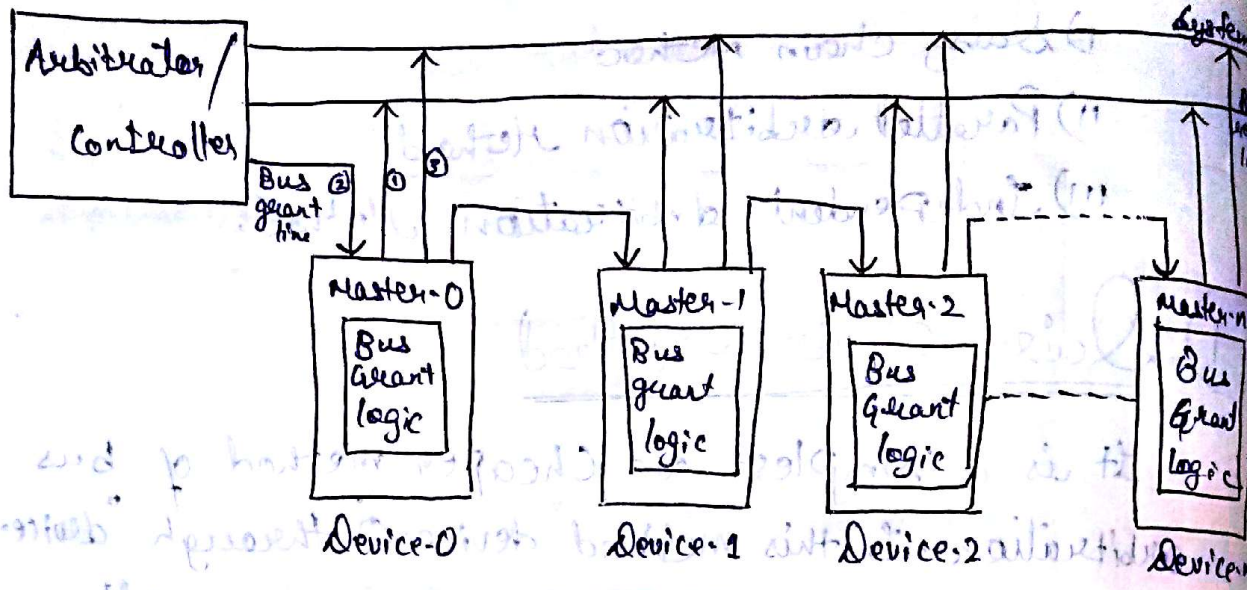
i) Daisy Chain Method

It is a simplest and cheaper method of bus arbitration. In this method "device-0" through "device-n" are all attached to the same bus and they all shared a bus request line that goes into a arbitrator.

when a device wants to be a bus-master - it assert (hold) the bus request line. when the arbitrator see the bus request, it issued bus grant to the concerned device and arbitrator assert the bus grant line so that, another device cannot access bus in this period of time.

The arbitrator sends bus grant signal in series to the connected devices. If a device does not want a bus then it simply pass the bus grant to the next device and so on until it find out the bus request device.

Bus Request ⇒ Bus grant ⇒ Data send
① ② ③



2) Parallel Arbitration

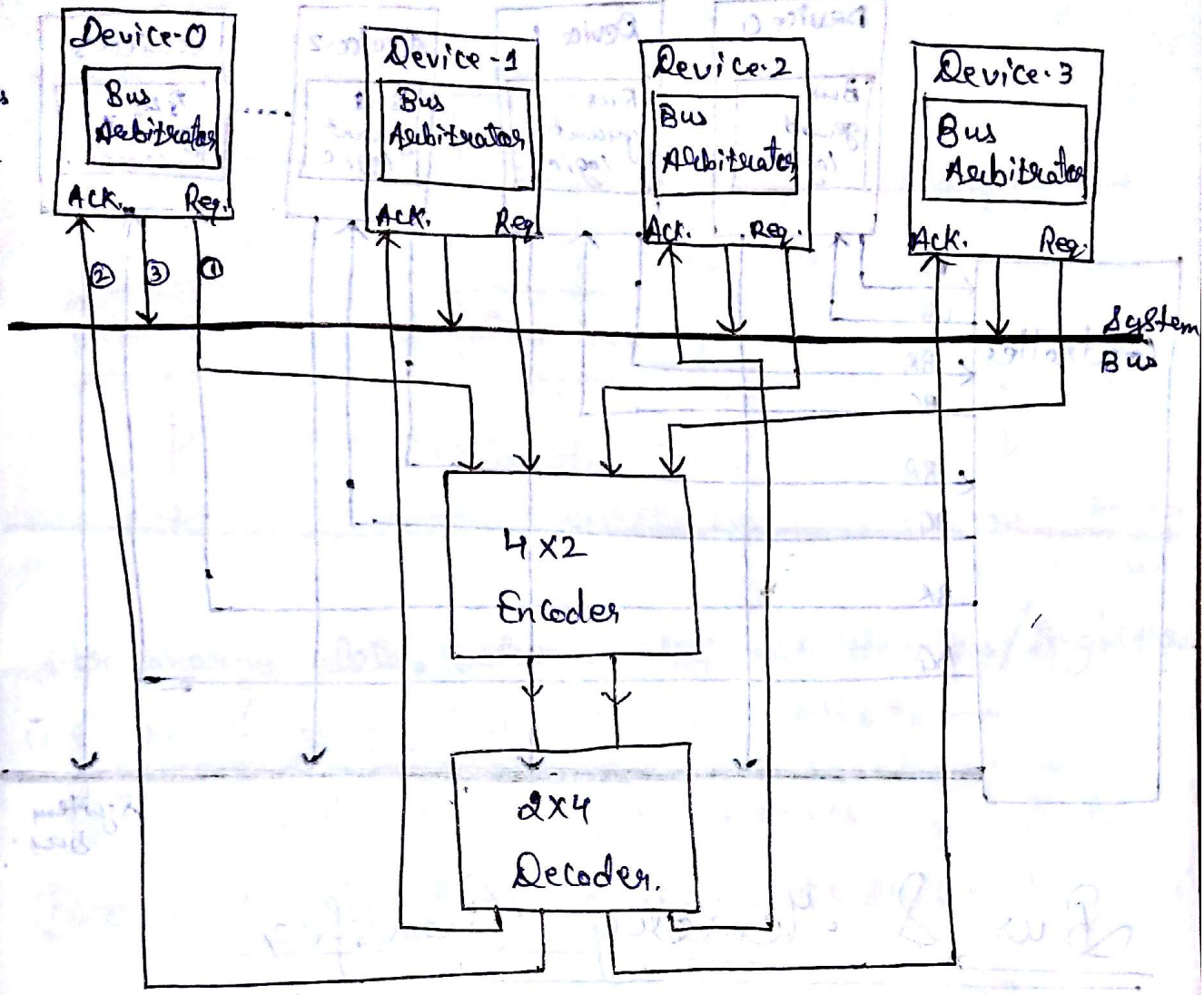
The parallel arbitration consists of priority encoder or decoder. In this mechanism each bus arbitrator has a different bus request output line and bus acknowledgement input line. Each arbitrator enable a bus request line when its device is requesting to access system bus.

Each arbitrator bus request output line is connected to the input of priority encoder, and the output of the encoder generate a 2-bit code which represent the highest priority unit among these request of the buses.

The 2-bit encoder output is given to input of 2x4 decoder which enable the proper acknowledge line to grant bus access to the highest priority unit.

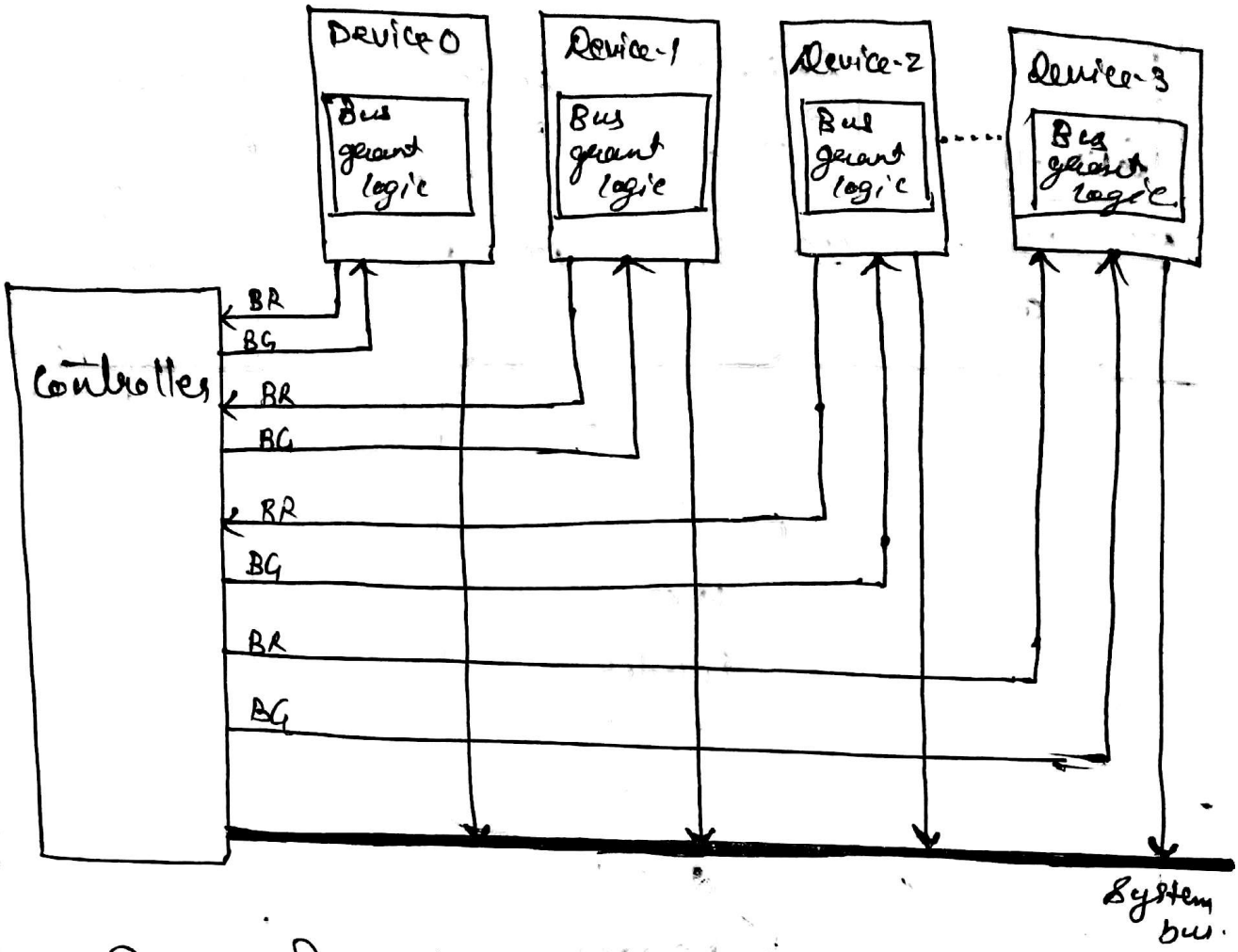
NOTE: A device is utilized system bus only when it receive acknowledgement signal.

Bus Request ⇒ Bus grant ⇒ Data send.
① ② ③



3) Independent Arbitration

In this method of arbitration each master as a ~~route~~ seperate pair of bus request and bus grant line and each pair has priority. The encoder within the Controller select the heighest priority request and assert the corresponding bus grant signal.



Bus & Memory Transfer

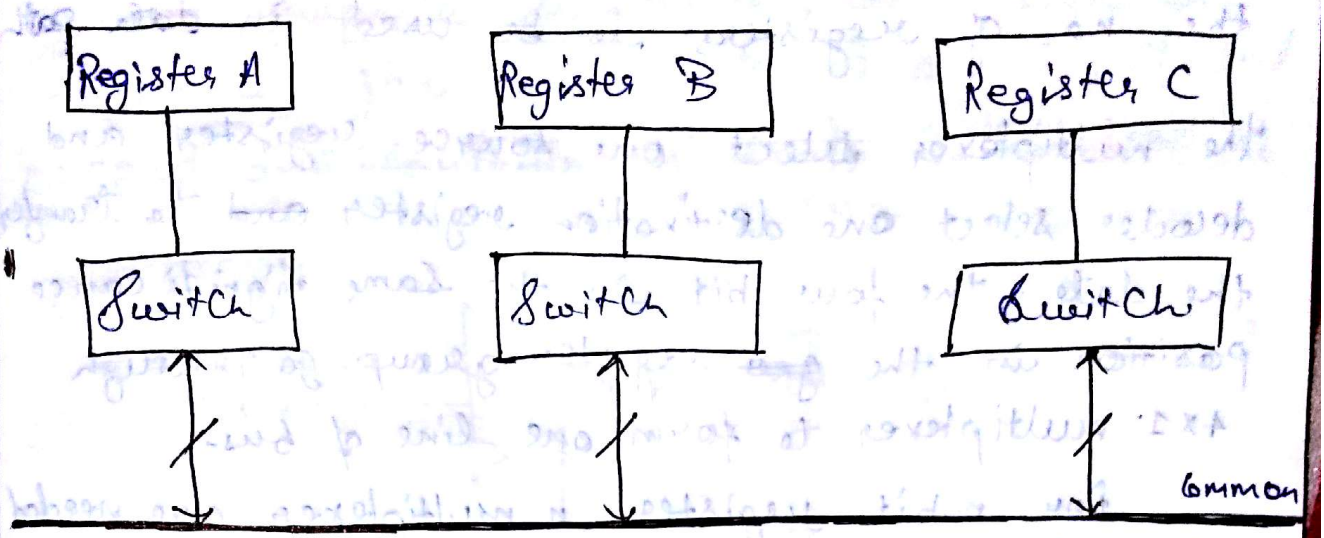
(1) Bus transfer :

It is a more efficient scheme for transferring information b/w registers in a multiple register configuration by using common bus system.

In this method, bus is shared by all the units. So, switch are required to enable path b/w different registers.

These switches are implemented by the help of multiplexer. One common control signal is used to control all the switches, the switch are closed (connection enable) and switch are

open (connection disable)

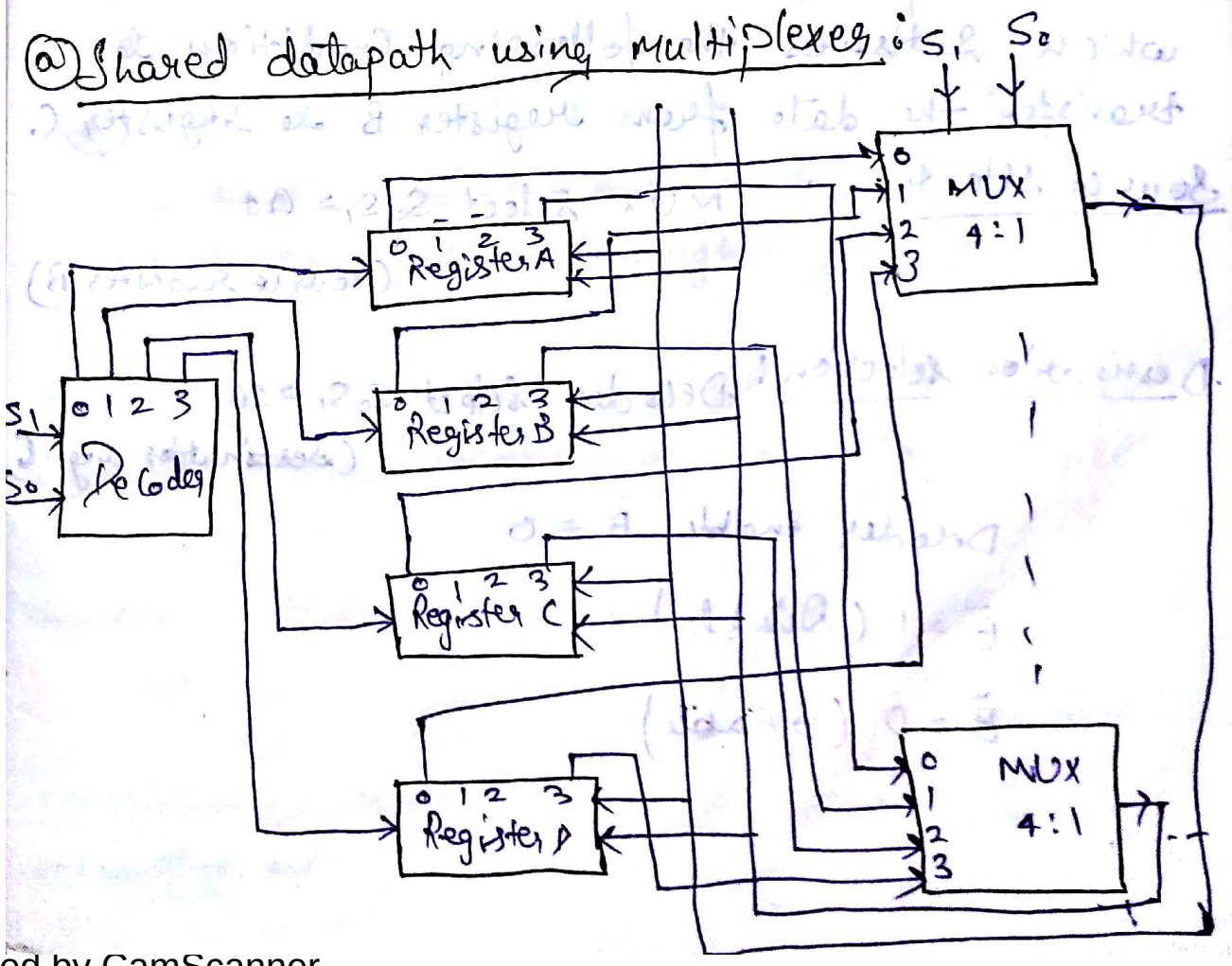


For sharing data between different devices/registers we use following bus transfer methods —

ⓐ Shared datapath using multiplexer

ⓑ Shared datapath using transistor buffer

ⓐ Shared datapath using multiplexer :



The required no. of multiplexer is based on the no. of registers to be used in data path. The multiplexer select one source register and decoder select one destination register to transfer the data. the four bit in the same significance position in the register group go through 4x1 multiplexer to form one line of bus.

For n-bit register, n multiplexer are needed to produce 'n' common line bus or shared path

eg) Consider the following statement

B → C, from the statement, we can identify the source register and destination register.

Source → B destination → C

The multiplexer and decoder select the line which satisfies the following condition to transfer the data from register B to register C.

Source selection: MUX select $S_0S_1 = 01$

(Source register B)

Destination selection: Decoder select $S_0S_1 = 10$

(Destination reg: C)

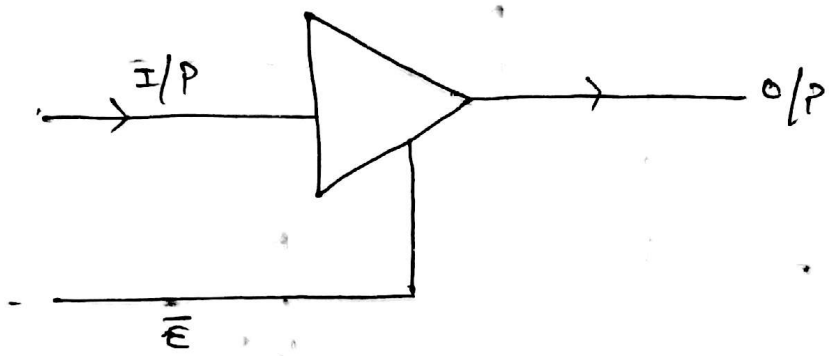
Decoder enable $\bar{E} = 0$

$\bar{E} = 1$ (Disable)

$\bar{E} = 0$ (Enable)

(5) Shared data path using transistor buffer :

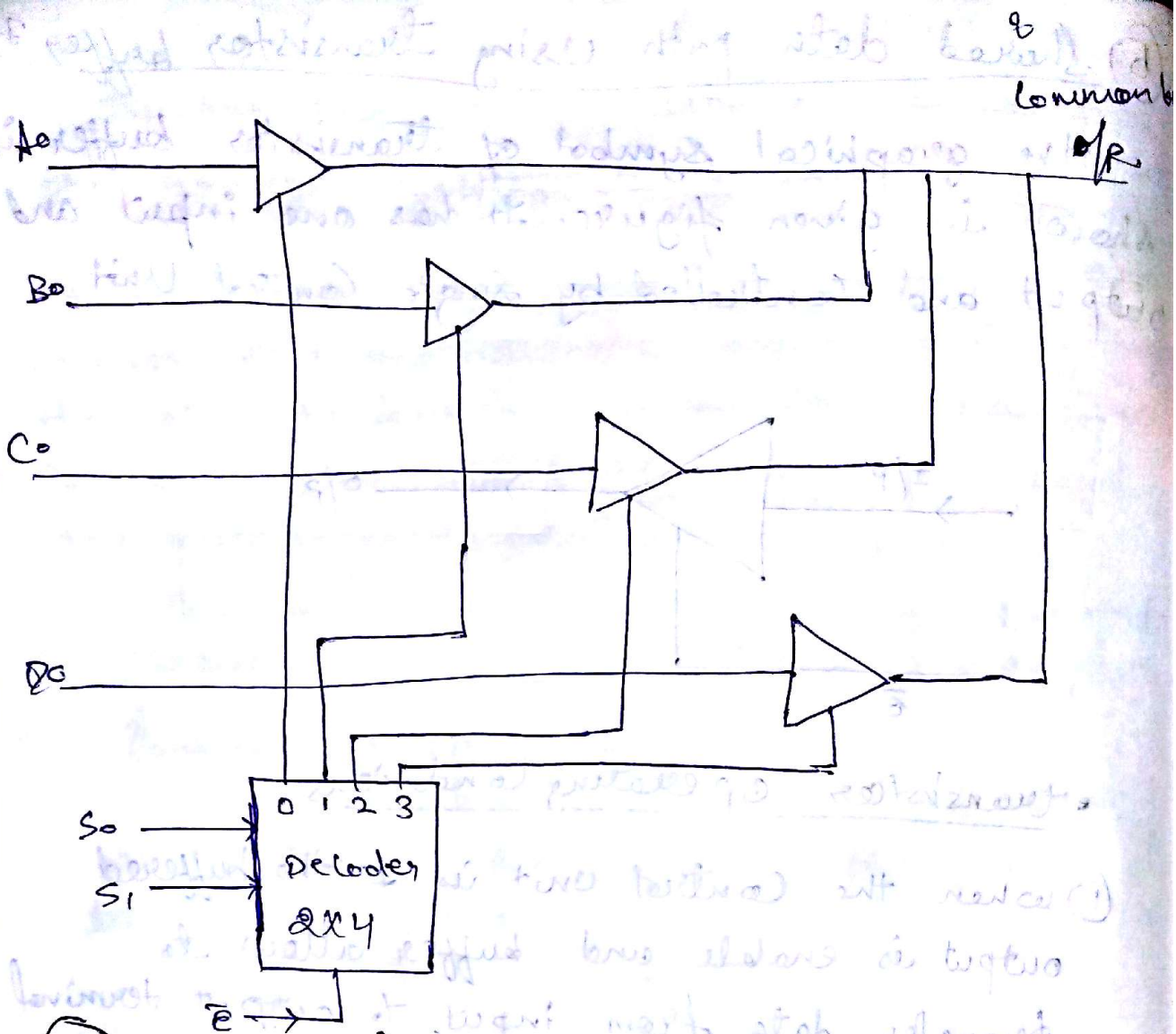
The graphical symbol of transistor buffer is shown in given figure. It has one input and output and controlled by single control unit.



- **transistor operating conditions**

(1) when the control unit is 1 - the buffered output is enable and buffer allows to transfer data from input to output terminal through them.

(2) when the control unit is 0 - the buffered output is disable and it doesn't allows the transfer of data through it.



(2) Memory transfer :

The basic operation of memory transfer is to fetch data (read) and store data (write)

(a) **Fetch operation (Read)** : The read operation, transfer a copy of the content from memory location to CPU location (processor unit). In this operation, word remain unchanged in memory.

The memory word is symbolize by 'M', address register by 'AR' and data register by 'DR'

In read operation, the ~~operate~~ information is transferred into data register 'DR' from the

memory word 'M' selected by the address in address register 'AR'.
A simple read operation stats as follows

Read : DR ← M[AR]

(b) Store operation (write) : In this operation, processor transfer the data of content ~~to~~ register ~~that~~ is selected by the address in address register. The content of register 'B' is transfer to the selected memory location. A simple write operation state as follows

write : M[AR] ← B

Multiple-Bus Hierarchy :

why we use multiple bus hierarchy.

the performance of computer system degrade when large no. of devices are connected to the system bus. this is because of major two reasons —

i) when more devices are connected to the common bus system. we need to share two buses among these devices.
the sharing mechanism coordinate the use of buses to different devices.

ii) when the aggregate data transfer demand approach the maximum capacity of bus, the bus may become ~~bottleneck~~ bottleneck. In such situation, we have to increase data rate or we have to use much wider buses to transfer data.

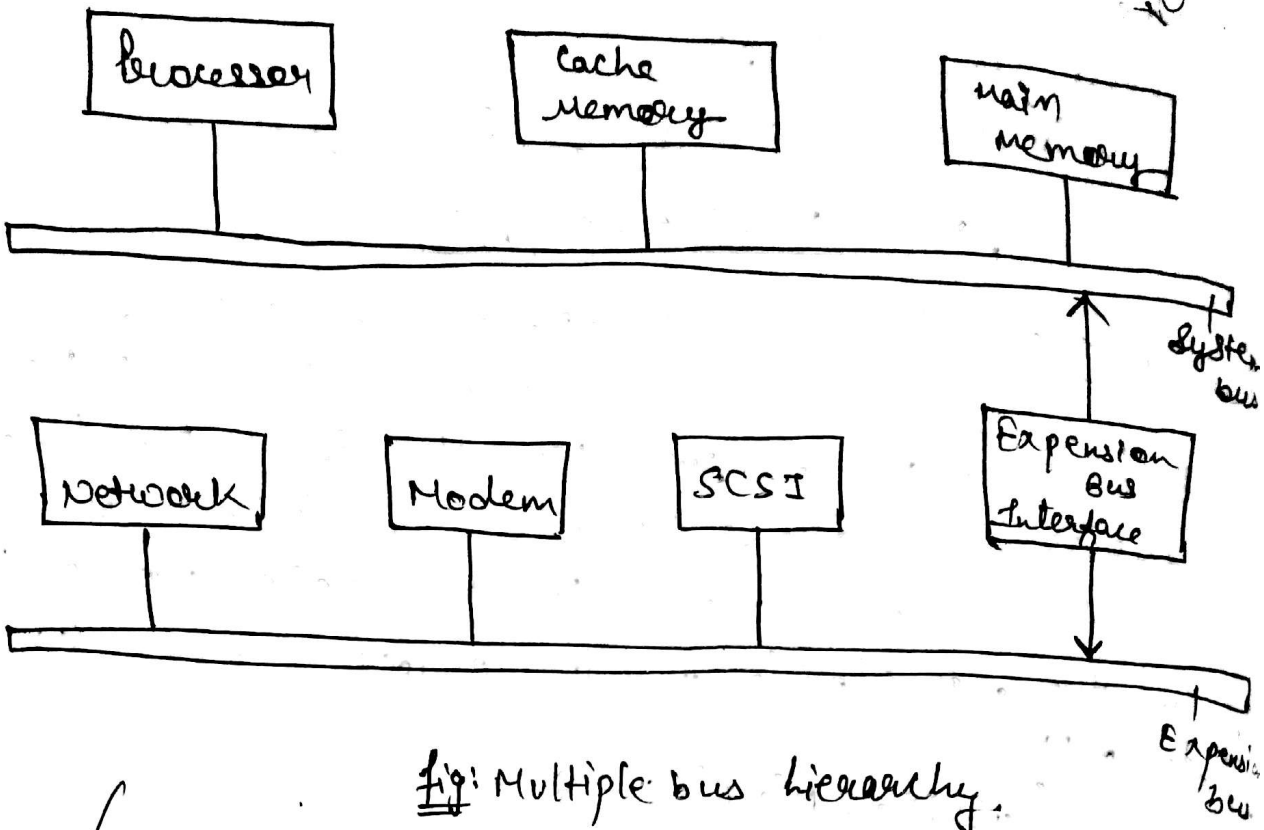


Fig: Multiple bus hierarchy.

Number System :

Decimal to Binary Conversion:

① $(23)_{10} \rightarrow (\quad)_2$

$\Rightarrow 10111$

② $(-23)_{10} \rightarrow (\quad)_2$

Assignment

① $(21)_{10} = (\ )_2$

② $(.4375)_{10} = (\ )_2$

③ $(153)_{10} = (\ )_8$

④ $(19.11)_{10} = (\ )_8$

⑤ $(2479)_{16} = (\ )_{16}$

⑥ $(1001)_2 = (\ )_{10}$

⑦ $(125)_8 = (\ )_{10}$

⑧ $(5271)_8 = (\ )_2$

⑨ $(123)_8 = (\ )_{16}$

⑩ $(125)_{16} = (\ )_{10}$

⑪ 1's Complement

⑫ 2's Complement

⑬ ADD of two binary no.

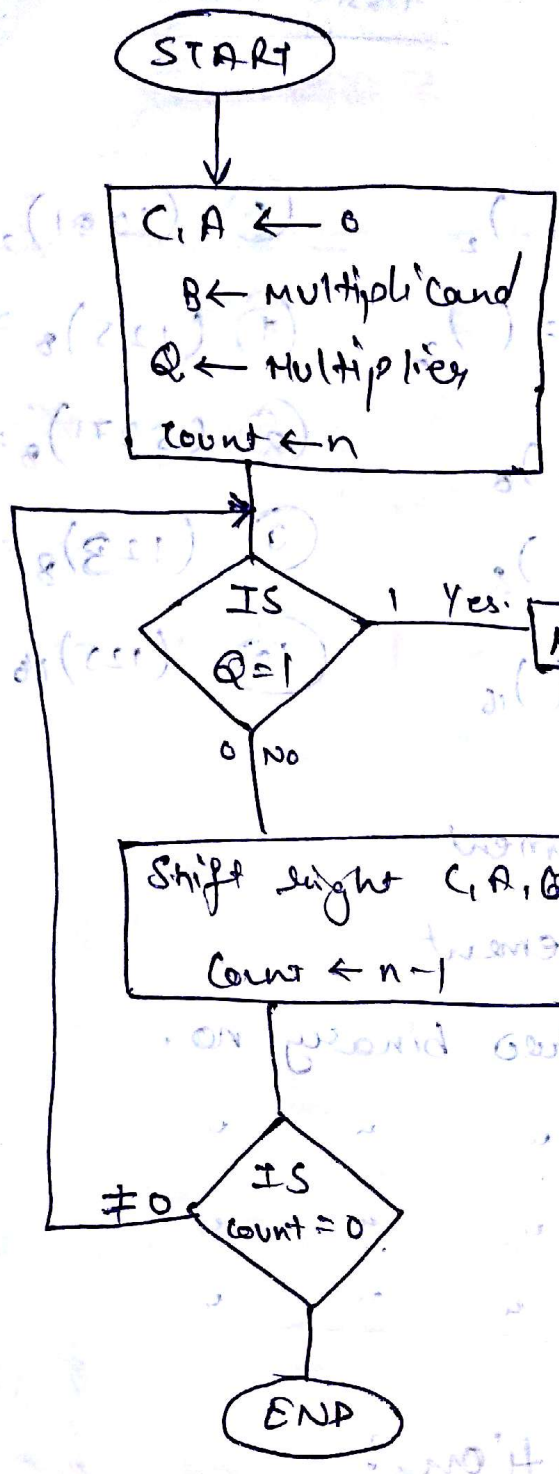
⑭ SUB " " " "

⑮ MUL " " " "

⑯ DIV " " " "

Multiplication:

PTO.



Multiplier and multiplicand are stored in the seperate register Q and B respectively. If bit Q_0 of register Q is 1 then multiplicand and partial product are added $A \leftarrow A + B$. All bits of register C, A, Q are shifted one right bit.

If Q_0 of Q is 0 - then no addition

it is performed only shift operation is carried out.

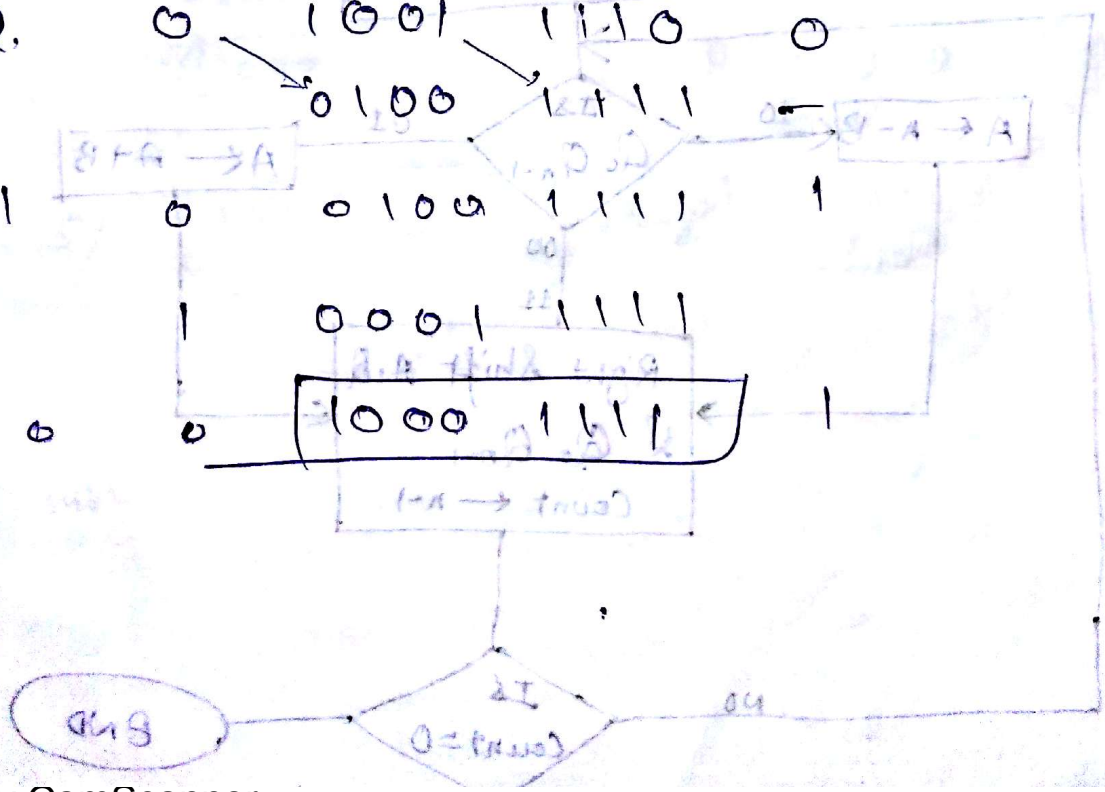
eg) 13 x 11 → Multiplier
↓
Multiplicand
↓
(BR)
(Q.R)

= 1101 X 1011

C → 0
A → 0000

B = 1101

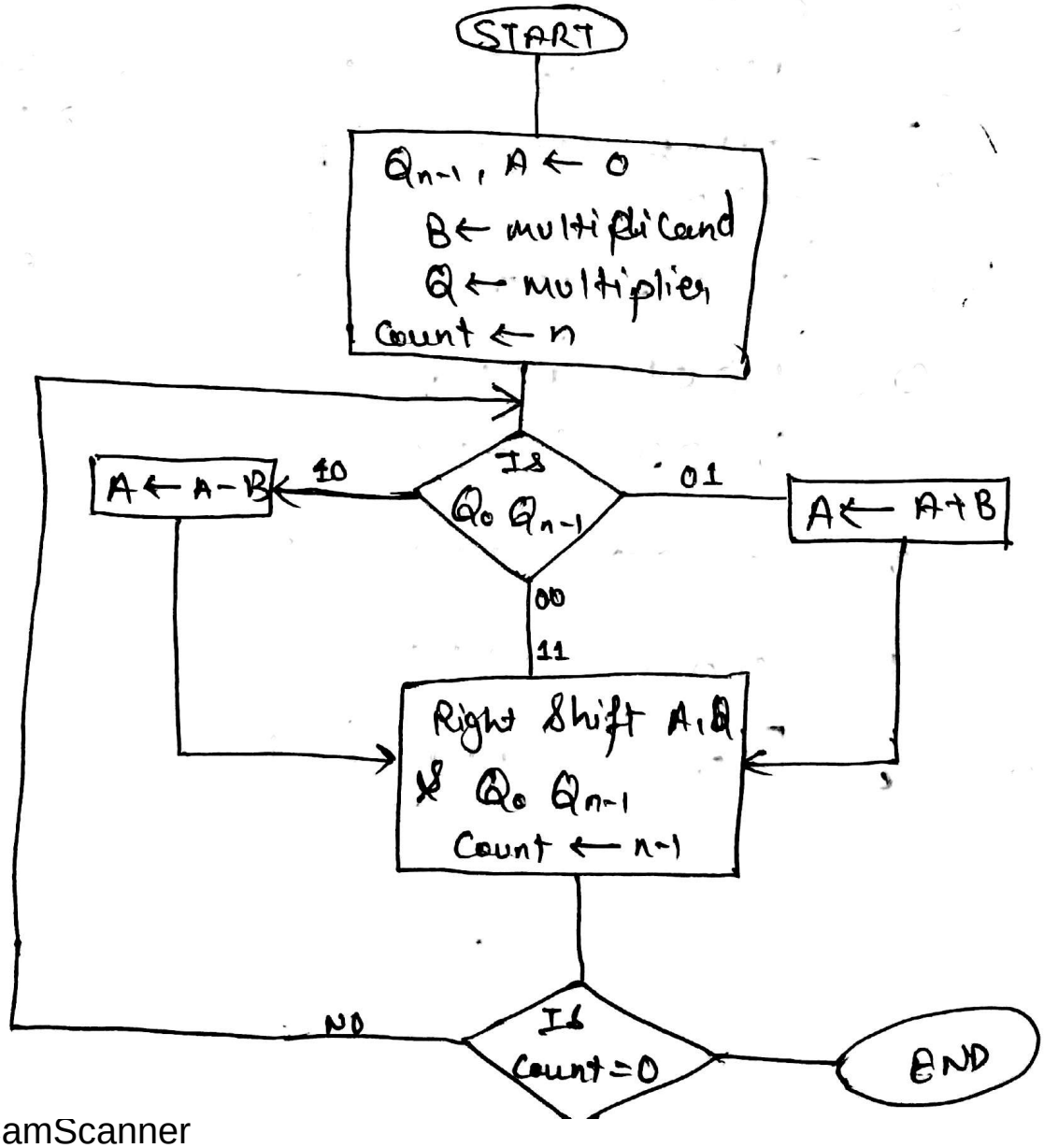
Count	Carry	A	Q	Q_0	Comment
—	0	0000	1011	1	Initial values
4	0	0000	1011	1	A ← A+B
	0	1101	1011	1	RS (C, A, Q) & Count ← n-1
3	0	0110	1101	1	
	1	0011	1101	1	A ← A+B
2	0	1001	1110	0	
	0	0100	1111	0	
1	0	0100	1111	1	
	0	0010	1111	1	
0	0	1000	1111	1	



Booth Multiplication Algorithm

Booth algorithm gives a procedure for multiply binary integer in Sign 2's complement representati This expression was invented by Andrew Donald booth in 1950's.

The booth algorithm use desk calculator that were faster at shifting as compare to addition., that's why ~~the increase~~ speed of prolessor will increase The booth algorithm perform few addition and Substraction then the normal multiplication algorithm.



eg) $\frac{13}{B} \times \frac{11}{Q}$

$B = 1101$

$Q = 1011$

Count	A	$Q_3 Q_2 Q_1 Q_0$	Q_0 Q_{n-1}	Comment
—	0000	1011	— 0	Initial values
4	0000	1011	10	$A \leftarrow A-B$ or $A \leftarrow A+\bar{B}+1$ (2's)
	−1101			
	0011	1011	10	Right shift
	0001	1101	1	
		Q_0	Q_0	
3	0001	1101	11	⇒ Right shift
	0000	1110	11	
		Q_0	Q_0	
2.	0000	1110	01	⇒ $A \leftarrow A+B$.
	1101			
	1101	1110	01	Right shift
	1110	1111		
1:	1110	$Q_0 = \bar{Q}_0$	0	
	+0001	1111	10	⇒ $A \leftarrow A-B$
	Carry 1 0001			Right shift
		1111	10	
0)	1000	1111	10	Right shift
		Q_0	$\bar{Q}_0$	

O/P - 1110

Subn.

Q) 13 X -11

→ B = 13 = 1101

Q = -11 = 0101 (2's Comp. of 11

count	A	Q	Q_0 Q_{n-1}	Comment
4	0000	0101	1 0	
	0011			A ← A+B+1
	0011	0101	1 0	
				Right Shi
				and count -
	0001	1010	— 1	
3.	0001	1010	01	
	1101			A ← A+B,
	1110	1010	01	
				Right Shi
				and count
	1111	0101	— 0	
2.	11110	0101	10	
	0010			A ← A+B+1
	×① 0010			
		0101	10	
				Right Shi
1	0001	0010	— 1	

1.)

1001	0010	01	A ← A+B
0011			
0100	0010	01	
			Right Shi
1011	0001	— 0	

1)

1001	0010	01	A ← A+B
0010			
1011			

16

1. 0001 0010 01 A ← A+B

1101

1110 0010 01 R.S

1111 0001 0

(9) −13 × 11

B = −13 = 0011 (2's comp of 13)

Q = 11 = 1011

Count	A	Q	Q_0 Q_{n-1}	Comment
4)	0000	1011	10	$A \leftarrow A+\bar{B}+1$
	1101			
	1101	1011	10	right shift & Count-1
	1110	1101	−1	
3)	1110	1101	11	right shift
	1111	0110	−1	
2)	1111	0110	01	$A \leftarrow A+B$
	0011			
	(1) 0010			right shift
	1001	0011	0	
1)	1001	0011	10	$A \leftarrow A+\bar{B}+1$
	1101			
	(1) 0110	0011	10	right shi
	1011	0001		

eg) -13×-11

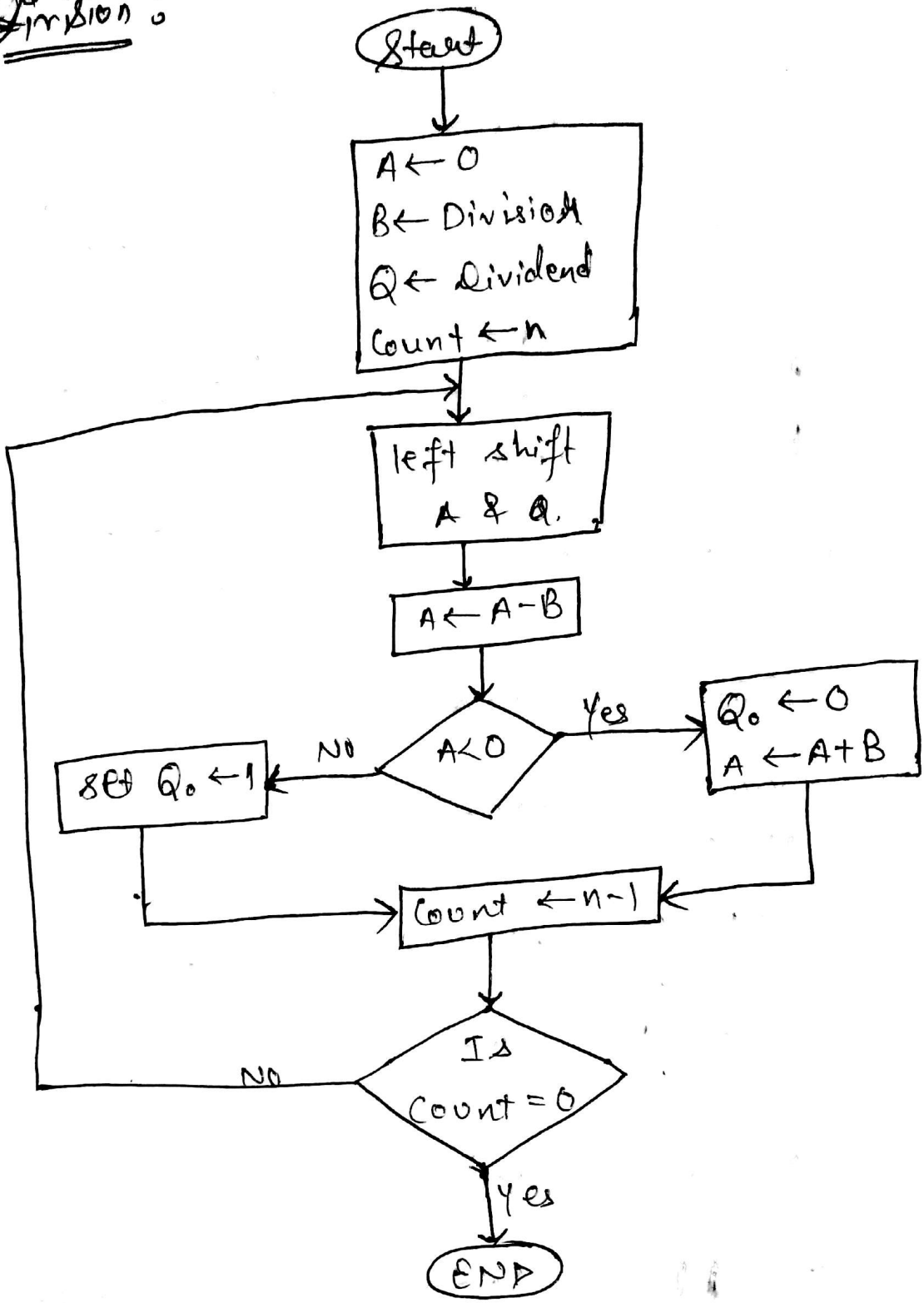
$B = -13 = 0011$

$Q = -11 = 0101$

Count	A	Q	$Q_0 Q_{n-1}$	Comment
4)	0000	0101	10	$A \leftarrow A + \bar{B} + 1$
	1101			
	1101	0101	10	right shift
	1110	1010	-1	
3.)	1110	1010	01	$A \leftarrow A + B$
	0011			
	① 0001	1010	01	right shift
	1000	1101	-0	
2.)	1000	1101	10	$A \leftarrow A + \bar{B} + 1$
	1101			
	① 0101	1101	10	right shift
	0010	1110	-1	
1)	0010	1110	01	$A \leftarrow A + B$
	0011			
	0101	1110	01	right shift

1110	1111	-0

Division :



Q Find 7/3

Sol

7 → dividend → Q. → 0111

3 → divisor → B → 0011

B = 00011

Count	A	Q	Comment
	00000	0111	initial value
4	00000	0111	L. Shift of A & Q
	00000	1110	$A \leftarrow A-B$ $\leftarrow A+\bar{B}+1$
	11101	1110	$Q_0 \leftarrow 0$ & $A \leftarrow A_0$ ~~Count ← n-1~~
	00011		
	⑦① 00000	1110	Count ← n-1
3	00000	1110	left shift AQ
	00001	1100	$A \leftarrow A+\bar{B}+1$
	11100		
	11110	1100	$Q_0 \leftarrow 0$, $A \leftarrow A_0$
	00011		
	×① 00001	1100	Count ← n-1
2.	00001	1100	L.S. of A & Q.
	00011	1000	$A \leftarrow A+\bar{B}+1$
	11101		
	×① 00000	1000	$Q_0 \leftarrow 1$ Count ← n-1
1.)	00000	1001	left. shift
	00001	0010	~~$A \leftarrow A+\bar{B}+1$~~ ~~(Q_0)~~
	11101		
	11110	0010	$Q_0 \leftarrow 0$, $A \leftarrow A_0$

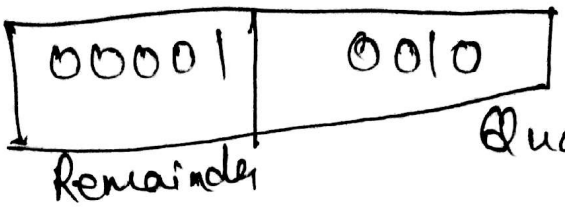
00010

00011

(1) 00001

count ← n - 1

0.)



Q Find 13/4.

Sol → Q = 13 = 01101

B = 4 = 00100

Count	A	Q	Comment.
4	00000	1101	left shift
	00001	1010	A ← A+B̄+1
	11011		
	11101	1010	Q0 ← 0 & A ← A
	00100		
	(1) 00001	1010	Count ← Count-
3.)	00001	1010	L.S of A &
	00011	0100	A ← A+B̄+1
	11011		
	11111	0100	Q0 ← 0 & A ← A
	00100		
	(1) 00011	0100	count - 1
2)	00011	0100	L.S of A & Q
	00110	1000	A ← A+B̄+1
	11011		
	00001	0	

00010 1000 $Q_0 \leftarrow 1$

Count -1

1) 00010 .1001 L.S of A

00101 0010 $A \leftarrow A + \bar{B} + 1$

11011

---

x(1) 00001 0010 $Q_0 \leftarrow 1$,

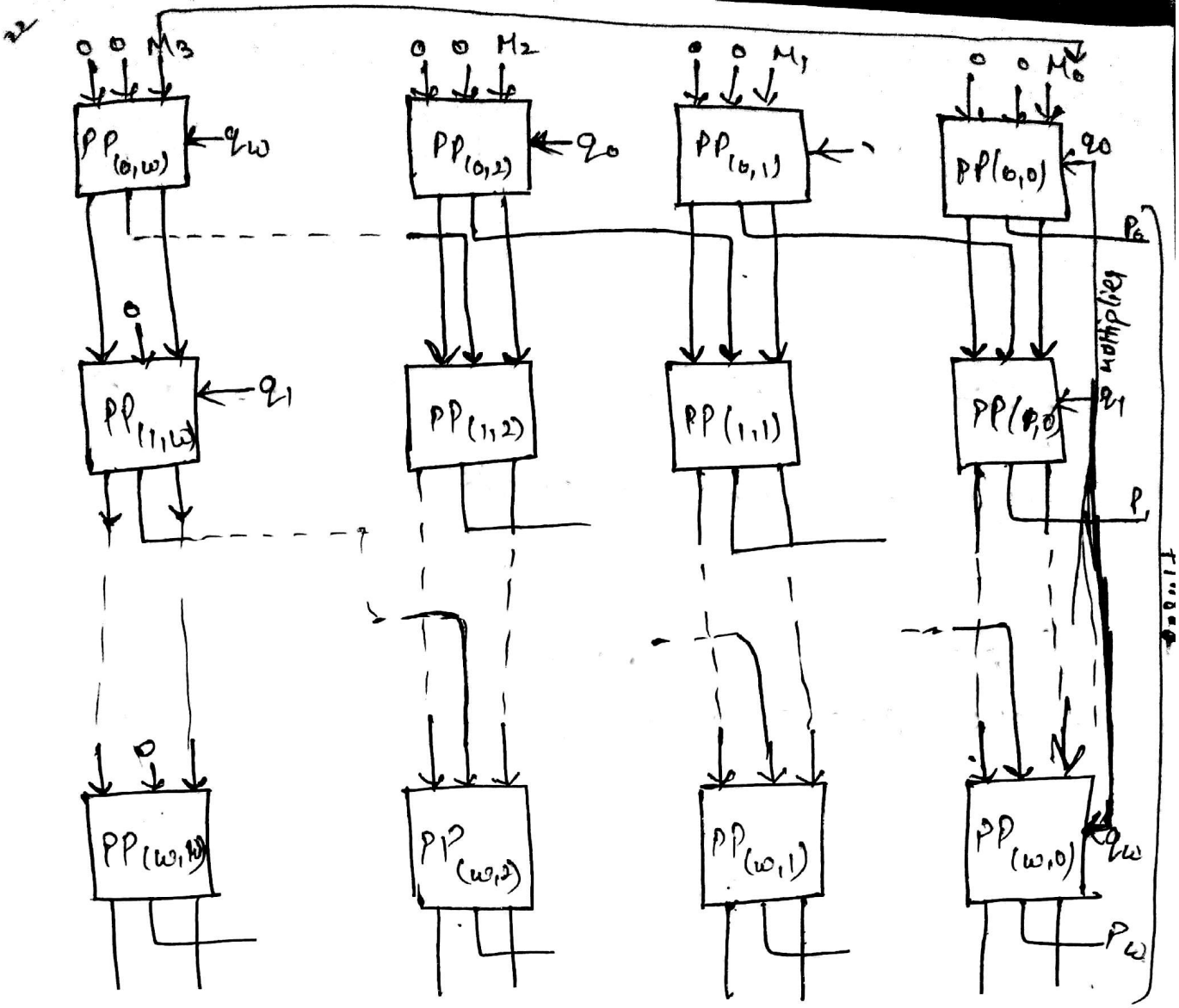
Count -1

0) 00001 0011

Remainder Quotient.

Array Multiplier :

The serial method we use for multiplying two unsigned numbers required only small amount of hardware, but the time required to multiply two numbers of length 'n' needs n^2 times. So the time complexity for multiplication of 2 n will increase. To overcome the above problem, we use parallel pipeline multiple array to multiply two numbers. The symbolic representation of array multiplier is given below—



Floating Point Arithematic

All the arithematic operations in floating point number can be carried out using the fixed Point arithematic operation. In this section, we explore floating point arithematic in base 2 & base 10 by keeping the requirement of floating point representation in mind. Floating point number consists of two parts —

1. Mantisa (m)
2. Exponent (e)
3. Radix (r)

and the floating point no. is represented by

$$m \times r^e$$

The floating point arithematic consists of 4 operations —

(1) Addition (2) Subtraction (3) Multiplication (4) Division

Eg. Perform the following operation —

(1) A+B (3) A * B

(2) A-B (4) A/B

where A = 225 & B = 30

Sol:

Base 10

$A = 225 = 2.25 \times 10^2$

$B = 30 = 0.30 \times 10^2$

Problem in floating point Arithmetic

1) Mantisa Overflow $(.99 \times 10^{99})$

2) Mantisa Underflow $(.100 \times 10^{-99})$

3) Exponent Overflow

4) Exponent Underflow

Adder : It is a combinational, logical circuit which perform the addition of two nos. to produce the output.

Adder are of different types —

i) Half Adder

ii) Full Adder

iii) Parallel Adder

iv) Look Ahead Carry adder

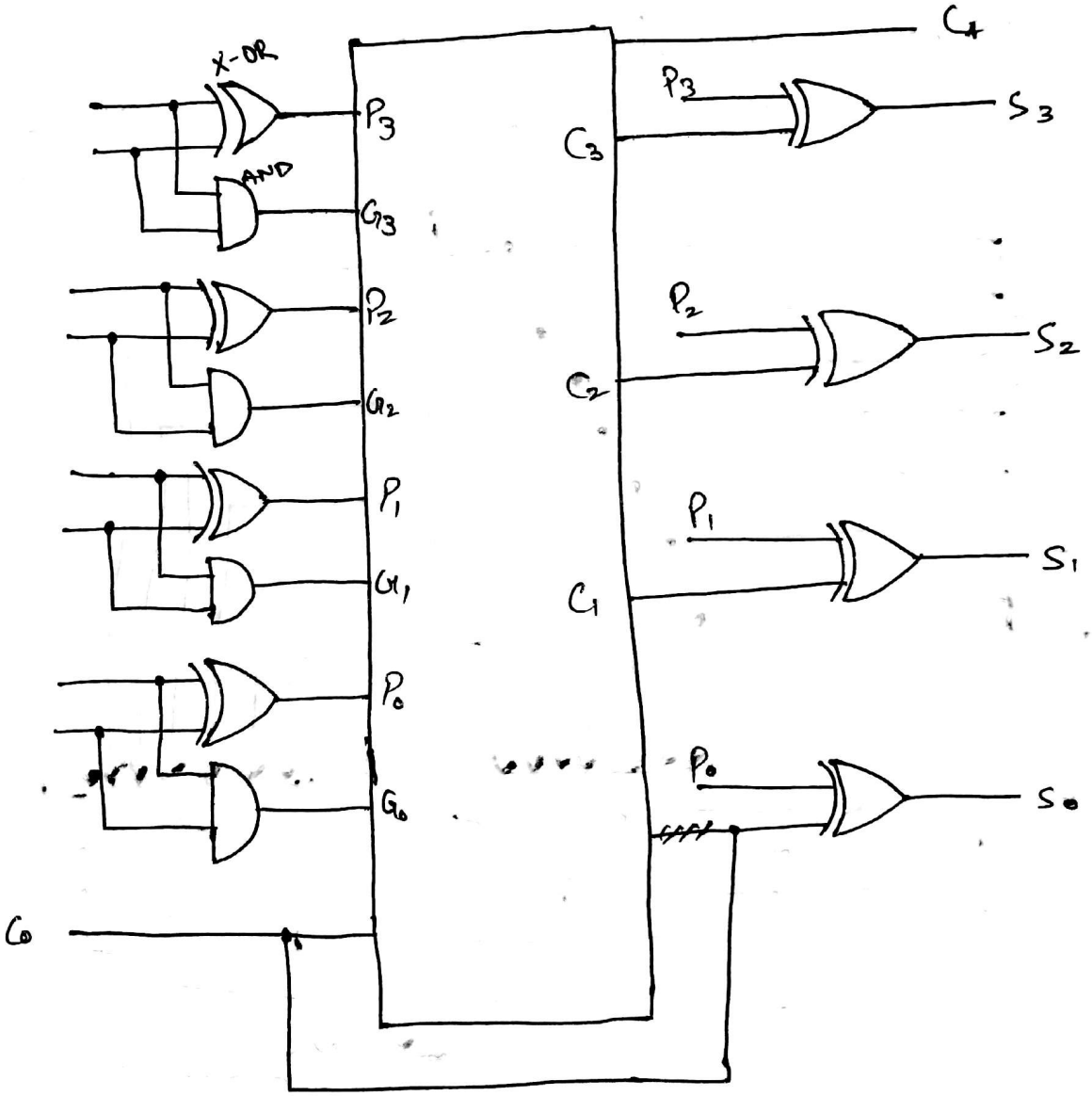


Fig: 4-bit look ahead carry Adder

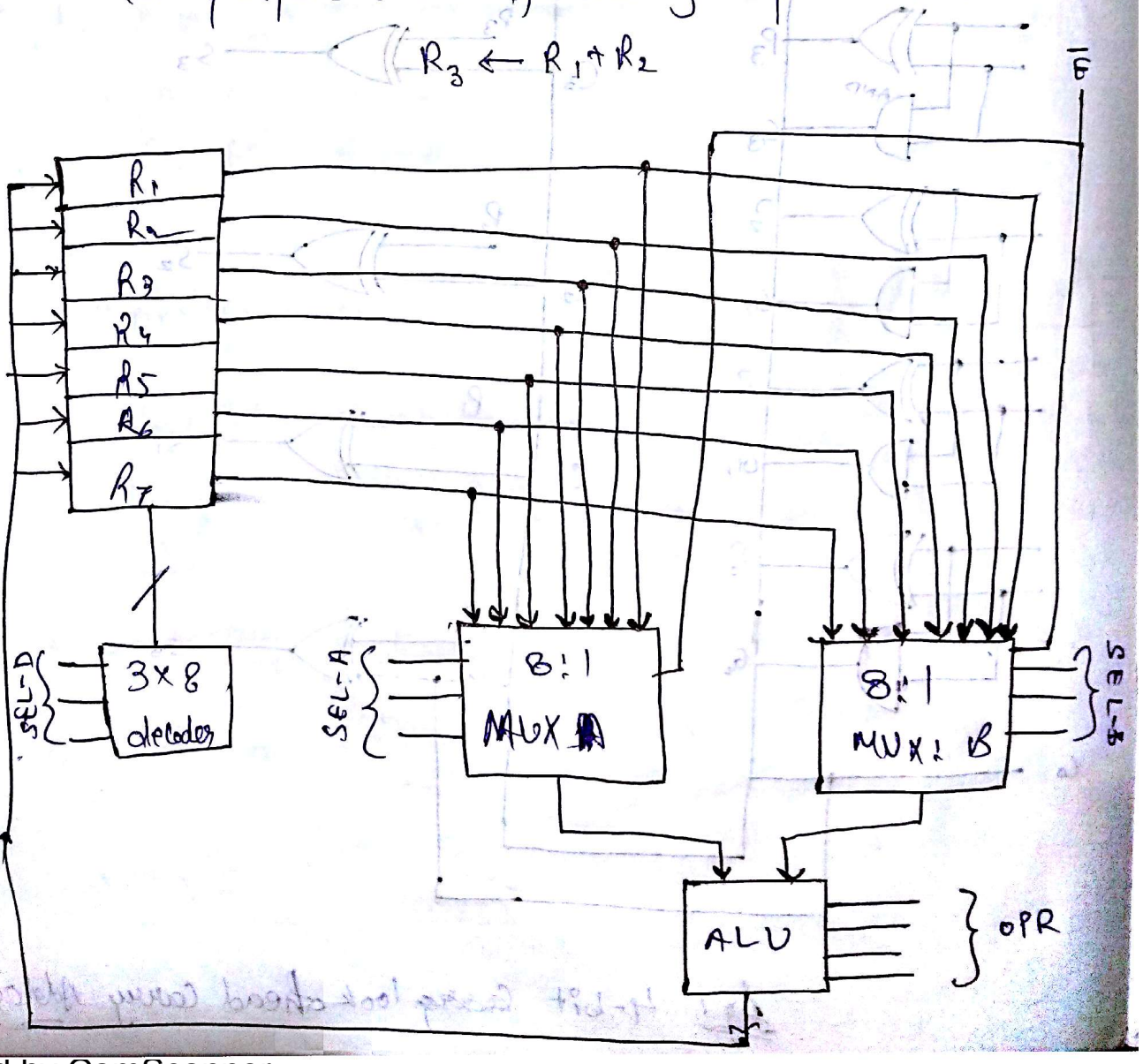
General Register Organization

It is a storage unit capable of storing holding a couple of bits. The register set store immediate data used during the execution of instruction. Therefore the ALU perform the required micro-operations for executing an instruction.

eg Consider 7 general purpose register (GPR) are used for following operations-, where we use multiplexer & decoder.

→ To perform the following operation -

$R_3 \leftarrow R_1 + R_2$



Stack Organization:

Stack: It is a group of memory location in read-write memory i.e., use for temporary storage of binary information during the execution of program.

Stack pointer: The register that hold the address for the stack is called stack pointer

Operation on Stack:

① Push ② Pop.

1) **Push**: It is used to write operation.

2) **Pop**: It is used to read operation.

Memory Stack: the implementation of stack in CPU is done by assigning a portion of memory to a stack operation.

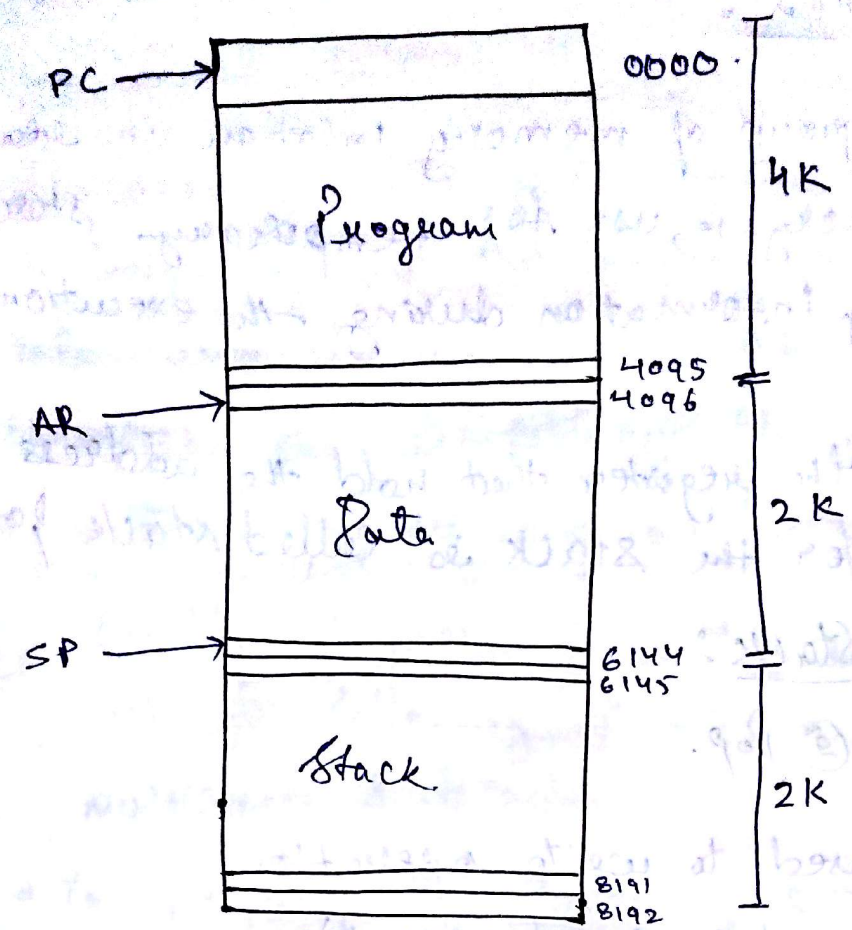
These operation are segmented into three categories:

a) Program

b) Data

c) Stack.

Eg) Let us consider, the total capacity of main memory is 8 Kb. The first 4 Kb is assign for a program 2 Kb is allocated for data & 2 Kb is allocated for stack.



Addressing Mode :

It is a method by which addressing source or destination is given in the instruction. They are of different types -

1) Direct addressing mode - In this mode, the address of the operand is given in the instruction given in the instruction itself.

eg) LDA 250H

where, - LDA is operation. (Load to Accumulator)
- 250H is address source.
- Accumulator is Destination.

ii) Indirect addressing Mode - In this mode, the address field of the instruction gives the address at which the

effective address is stored in the memory.

iii) Register Addressing Mode -

MOV A,B.

iv) Register Addressing Mode.

v) Immediate Addressing Mode

vi) Increment Addressing Mode

vii) Decrement Addressing Mode

viii) Index Addressing mode

Register Transfer Logic/Language

RTL is used to describe the micro-operations. This oper language is used some predefine symbolic notation for each micro-operation. The term 'transfer' implies the availability of hardware logic circuit that can perform a stated micro-operation and transfer the result of operation to the same or other register.

The word "Language" is borrowed from program who apply this term to programming language.

The basic features of RTL are —

(a) RTL is a Symbolic language.

(b) RTL is convinient tool for describing the internal organization of digital computer.

(c) RTL can also be used to facilitate the designed proces