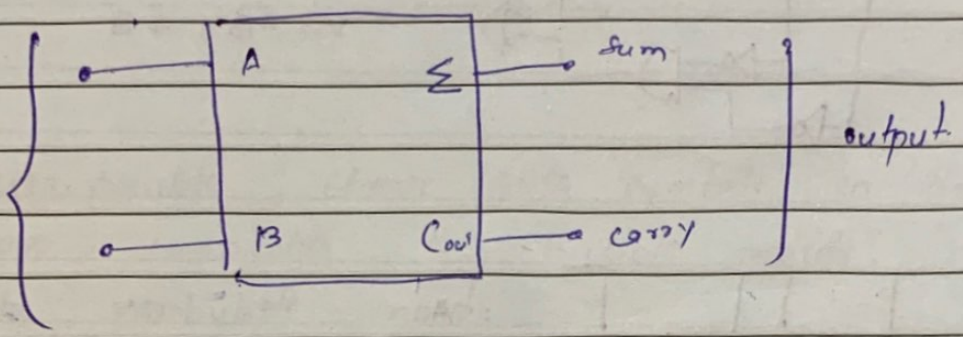
CH #06

Date

Half Adders : used to single bit no:

↳ it doesn't take carry from previous sum

logic symbol.



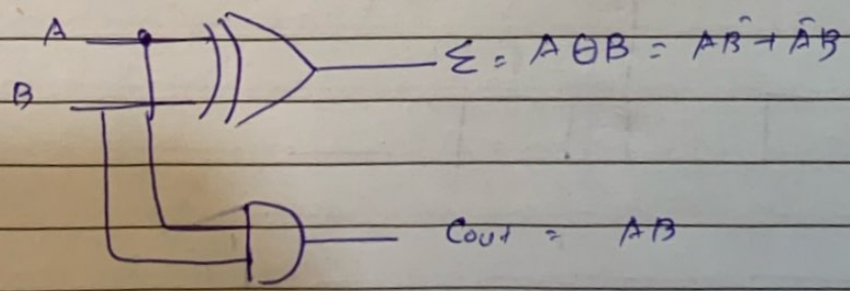
Truth Table.

A	B	Cout	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

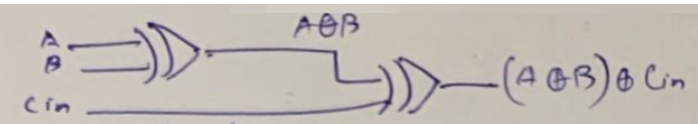
$\Sigma = A \oplus B$

$C_{out} = AB$

logic circuit



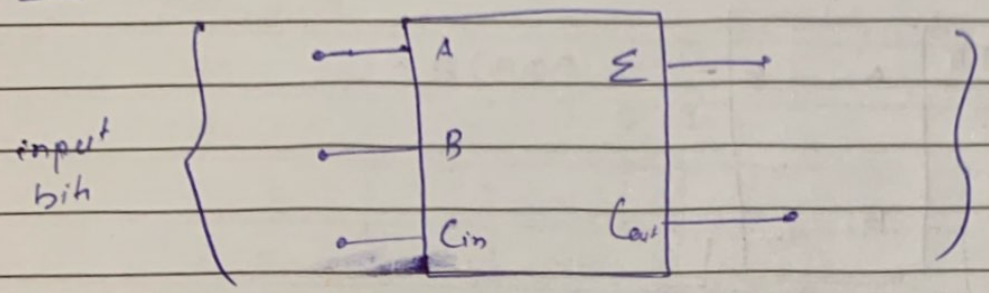
Al Habib



a) logic to form the sum of three bits

Date _____

Full Adders. → accepts two bits and an input carry and generates a sum output & a carry output



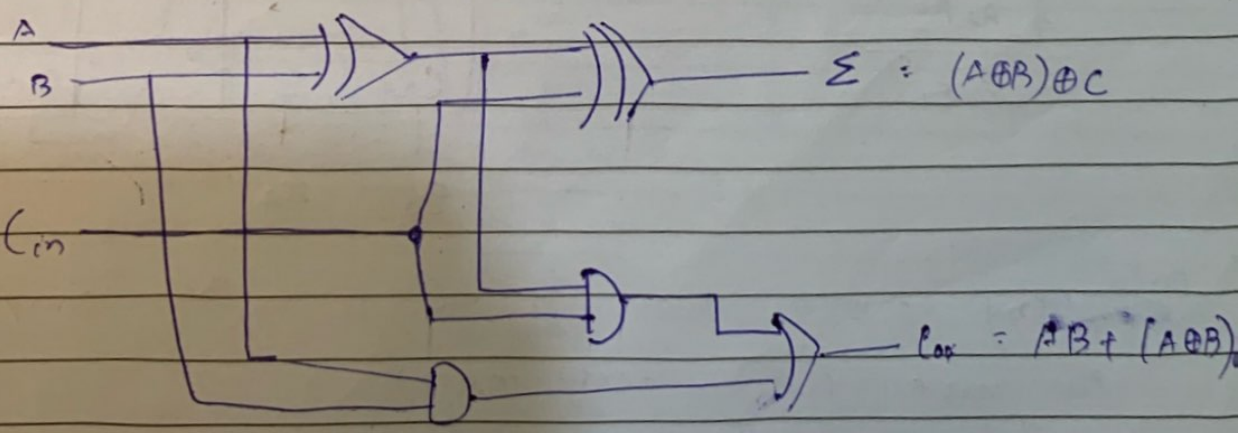
Truth Table.

A	B	Cin	Σ	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$(A \oplus B) \oplus C$:

$A'B + AB'$

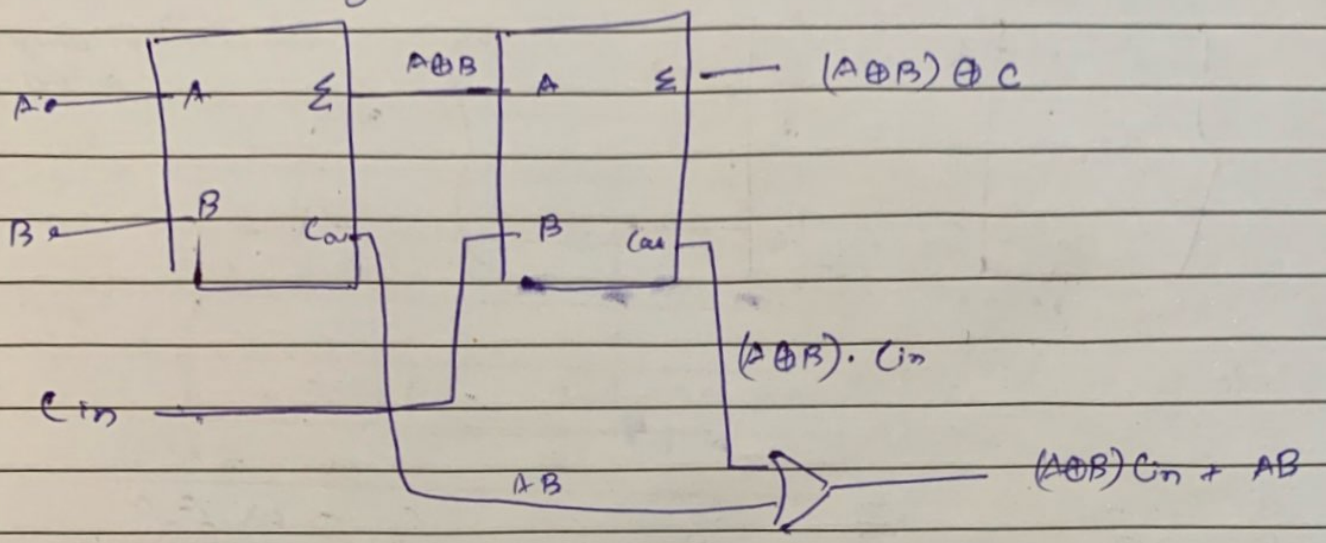
Logic circuit



Al Habib

Date _____

full adder using half adders



Parallel Adders

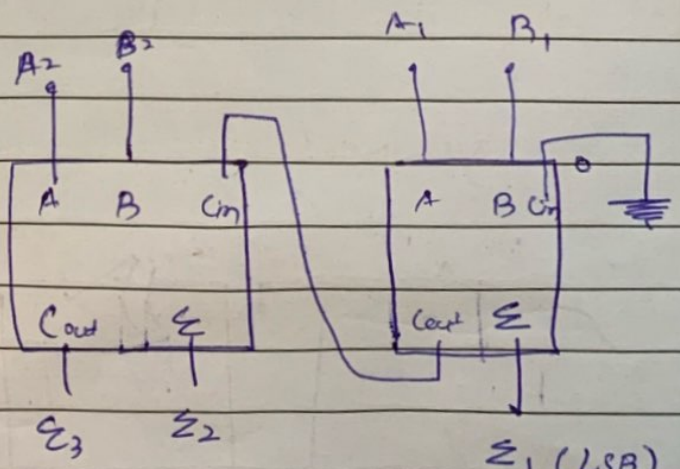
- for adding n-bit binary numbers we need n-full adder

- Adding two bit binary

↓

General formal

$A_2 A_1$
+ $B_2 B_1$
$\Sigma_3 \Sigma_2 \Sigma_1$



Note:
adding parallel adder → we pass the carry out to the next full adders carry

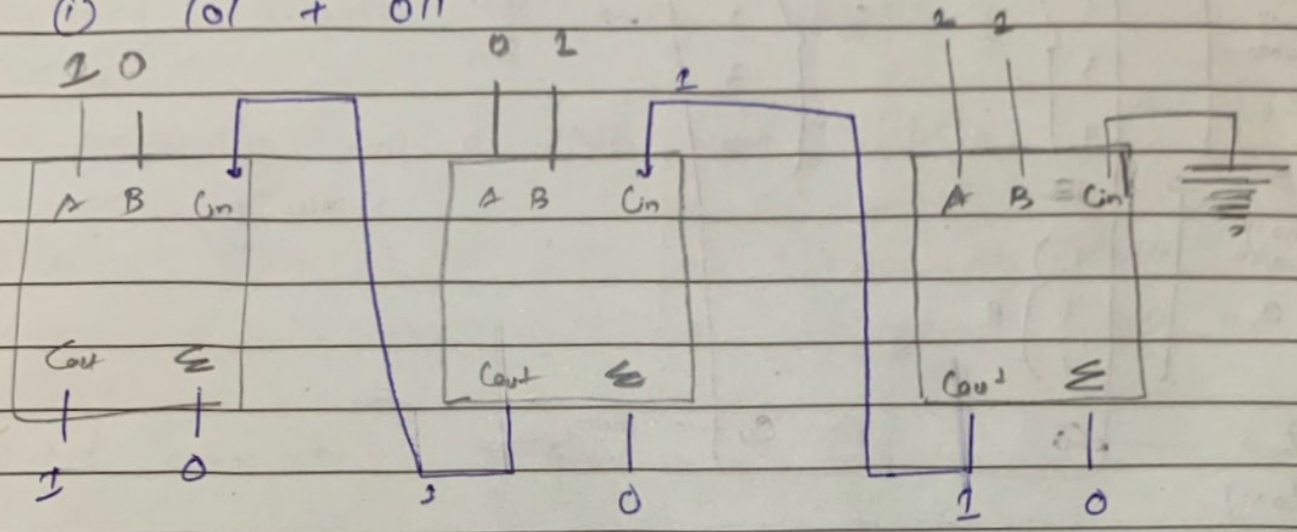
Al Habib

101
+ 011
1+0, 0+1, 1+1

Date _____

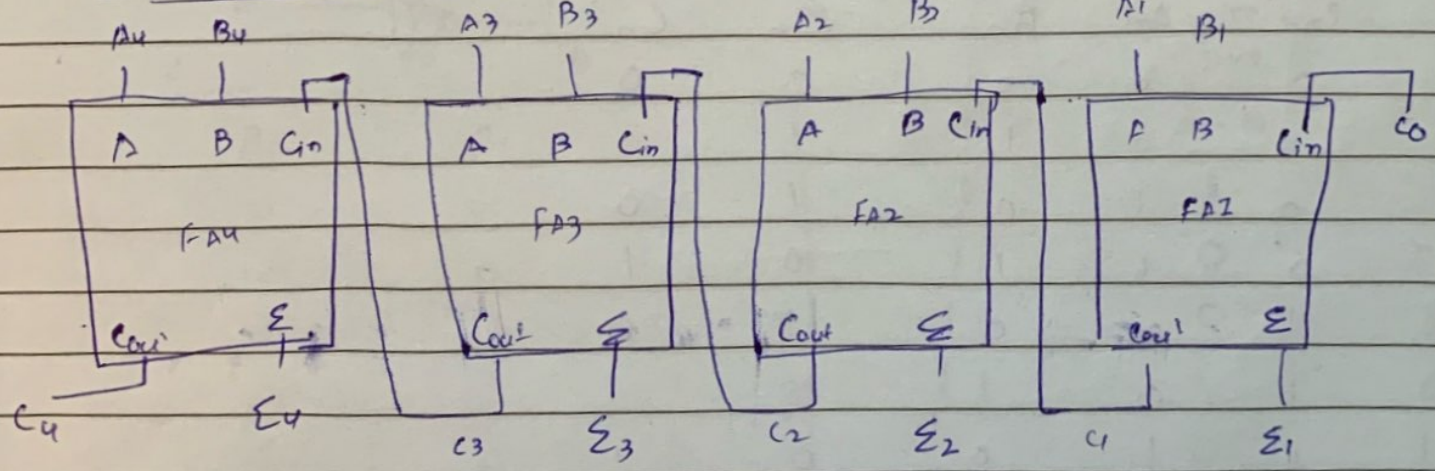
Eg: Determine the sum generated by 3 bit parallel adder

(i) 101 + 011



Sum = 1000

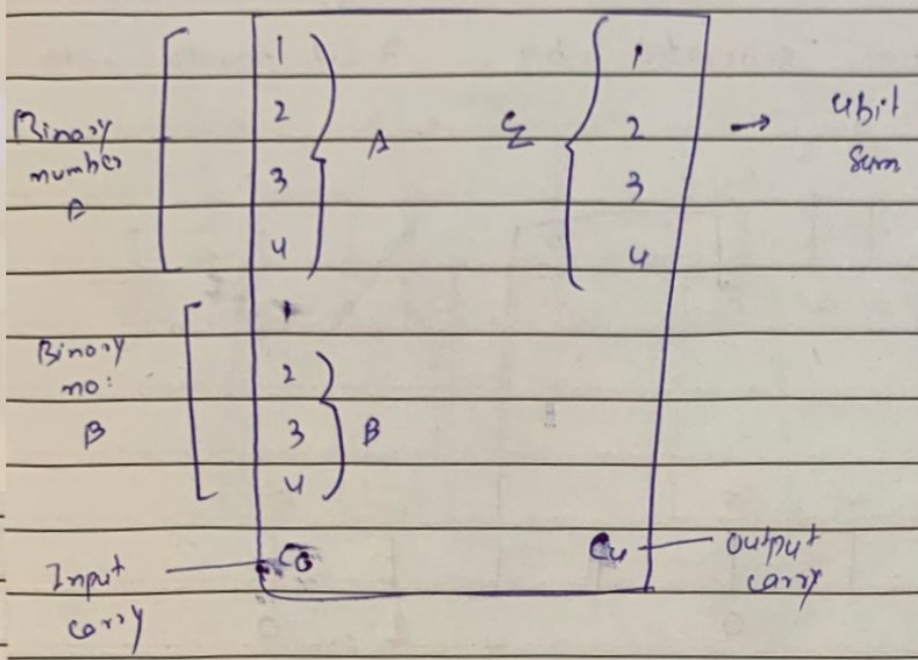
(ii) 4-Bit Parallel Adders.



a) block diagram

Al Habib

Date _____



→ logic diagram

4) Truth Table for each stg

C_{n-1}	A_n	B_n	Σ_n	C_n
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

so you can find sum of any two 4bit no: just by this truth table

$$\text{Sum} = C_4 \Sigma_4 \Sigma_3 \Sigma_2 \Sigma_1$$

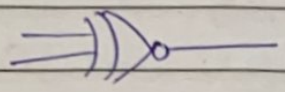
Al Habib

Date _____

Comparators → Has three outputs

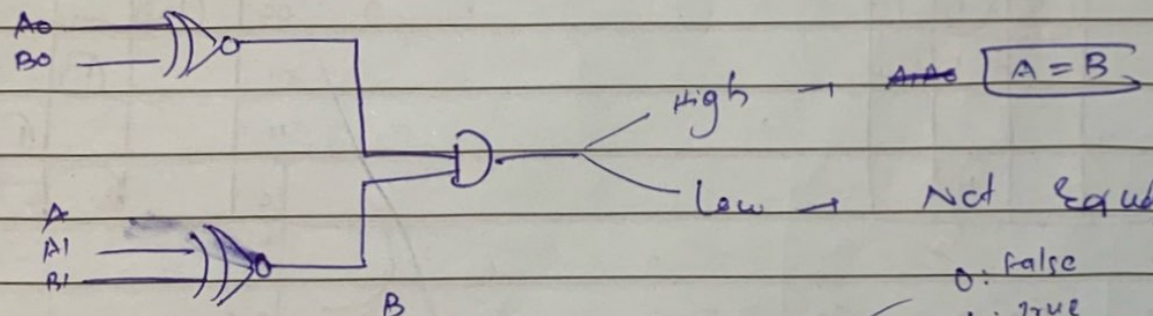
↳ are used to compare two binary quantities

1) Basic comparator was a XNOR Gate (1-bit)



→ b/c it will give "one" when both inputs are equal and zero when both are

2) 2-bit comparator



0: false
1: true

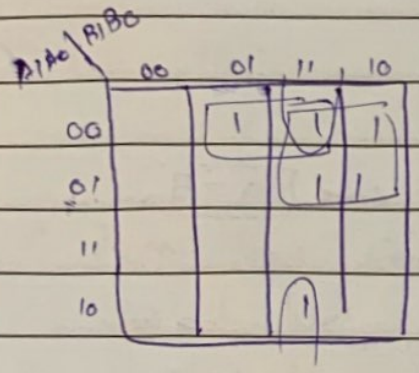
A		B		A<B	A=B	A>B
A_1	A_0	B_1	B_0			
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	0	0
0	1	1	1	0	0	0

Al Habib

Date _____

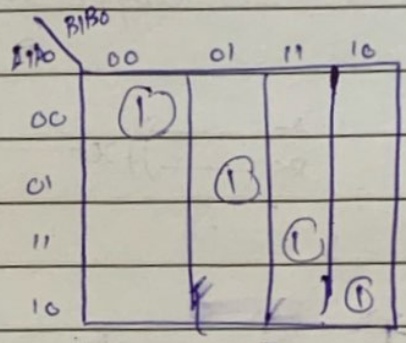
				A<B	A=B	A>B
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

For A<B



$A<B = \bar{A_1}\bar{A_0}B_0 + \bar{A_1}B_1 + \bar{A_0}B_1B_0$

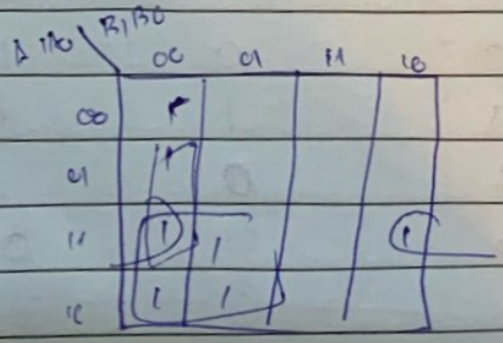
For A=B



Minimal Expn.

$= A=B \Rightarrow (A_1 \oplus B_1) \cdot (A_0 \oplus B_0)$

For A>B



$A>B = A_0\bar{B_1}B_0 + A_1A_0\bar{B_0} + A_1\cdot\bar{B_1}$

Date _____

Inequality → 4-bit comparator

↳ provide additional info $A>B$ or $A<B$

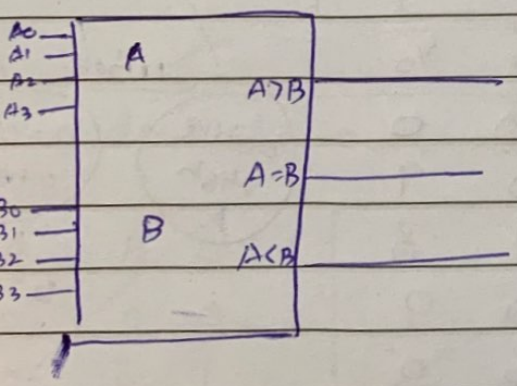
- compare MSB → A_3 & B_3

If different ✓ → $A>B$ or $A<B$

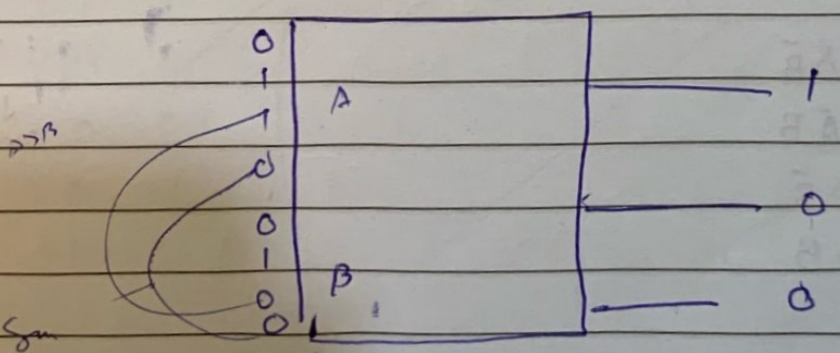
$A_3=1$ $B=0$ $A_3=0$ $B_3=1$

If same, then we check the lower bit (A_2 & B_2) ↓ continue.

Ex. $A_3=B_3$, $A_2=B_2$, $A_1=B_1$, $A_0=B_0$ → means a number is equal.



Q.

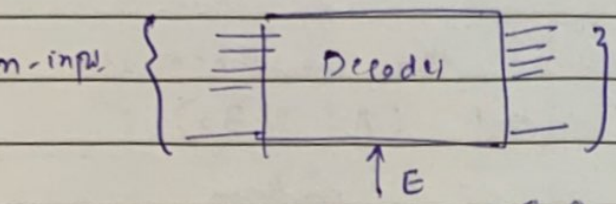


Al Habib

Date _____

Decoders

A decoder is a multi-input, multi-output logic circuit which decodes n-inputs to 2^n-outputs.



where E = Enable
- E=0 Decoder is disable → it will provide zero at output
- E=1 Decoder is Enable

2 to 4 - Decoder

Enable	input		output			
			2^3	2^2	2^1	2^0
E	A	B	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

output: 1 — except all zero

Active High

when A = 0 B = 0

n = 0

0+0 = 0

1

2^0

A = 1 B = 0

n = 2

2^1

1

Here

$Y_0 = \bar{A}\bar{B}$

$Y_1 = \bar{A}B$

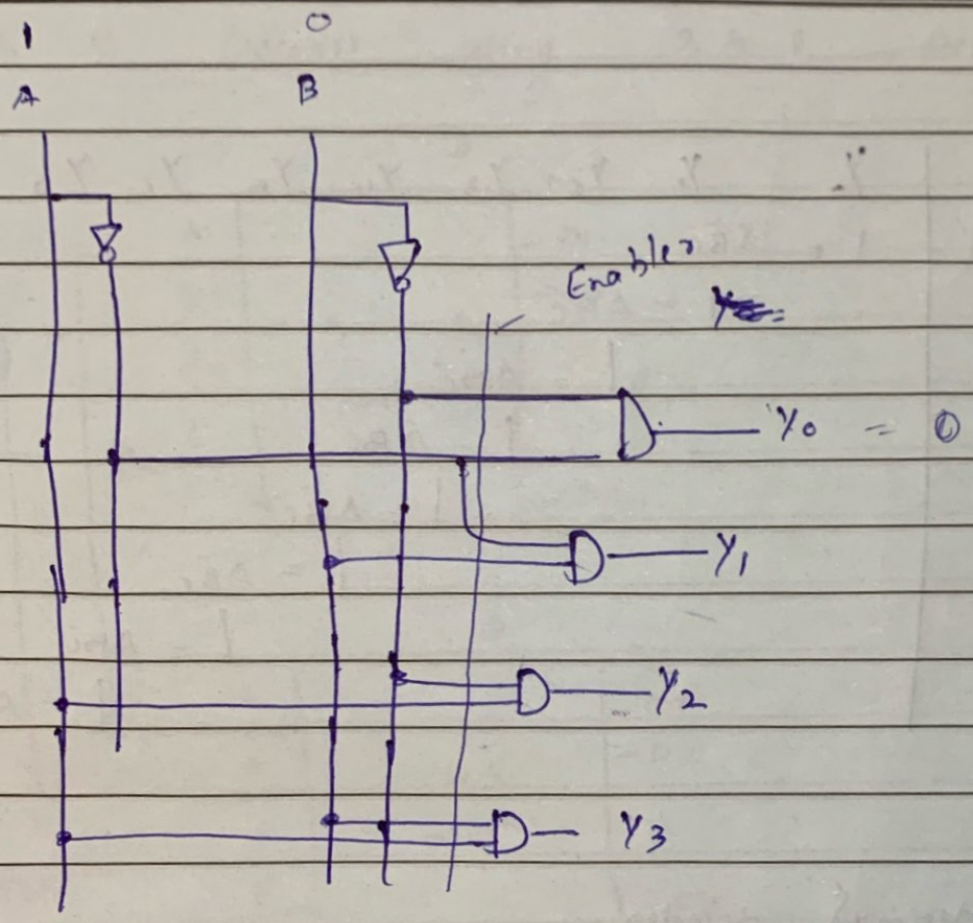
$Y_2 = A\bar{B}$

$Y_3 = AB$

In decoder, as per the specific input combination, only one of the output is high.

Date _____

Circuit



3-8 decoder

E	A	B	C	2^0 Y_0	2^1 Y_1	2^2 Y_2	2^3 Y_3	2^4 Y_4	2^5 Y_5	2^6 Y_6
1	0	0	0	1						
1	0	0	1		1					
1	0	1	0			1				
1	0	1	1				1			
1	1	0	0						1	
1	1	0	1							1

Date _____

3 to 8 decoder

E	A	B	C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
1	0	0	0	1 = $\bar{A}\bar{B}\bar{C}$							
1	0	0	1		1 = $\bar{A}\bar{B}C$						
1	0	1	0			1 = $\bar{A}B\bar{C}$					
1	0	1	1				1 = $\bar{A}BC$				
1	1	0	0					1 = $A\bar{B}\bar{C}$			
1	1	0	1						1 = $A\bar{B}C$		
1	1	1	0							1 = $AB\bar{C}$	
1	1	1	1								1 = ABC

(iv) BCD to decimal decoder

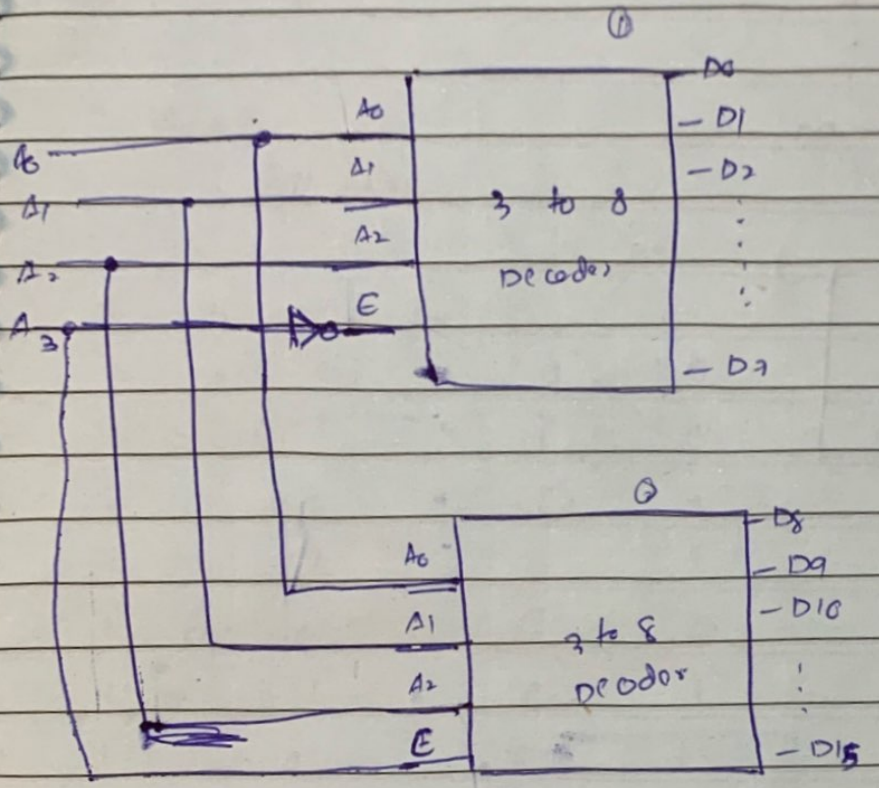
↳ use 4-16 decoder → but (only 10 outputs are considered (0-9)

~~A~~	A	B	C	D	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9	~~Y_{10} Y_{11}~~ Y_{12} ~~Y_{15}~~
	0	0	0	0	1 = $\bar{A}\bar{B}\bar{C}\bar{D}$										
	0	0	0	1		1 = $\bar{A}\bar{B}\bar{C}D$									
	0	0	1	0			1 = $\bar{A}\bar{B}C\bar{D}$								
	0	0	1	1				1 = $\bar{A}\bar{B}CD$							
	0	1	0	0					1 = $\bar{A}B\bar{C}\bar{D}$						
	0	1	0	1						1 = $\bar{A}B\bar{C}D$					
	0	1	1	0							1 = $\bar{A}BC\bar{D}$				
	0	1	1	1								1 = $\bar{A}BCD$			
	1	0	0	0									1 = $A\bar{B}\bar{C}\bar{D}$		
	1	0	0	1										1 = $A\bar{B}\bar{C}D$	

Al Habib

Date _____

4 to 16 Decoder using 3 to 8 Decoders.

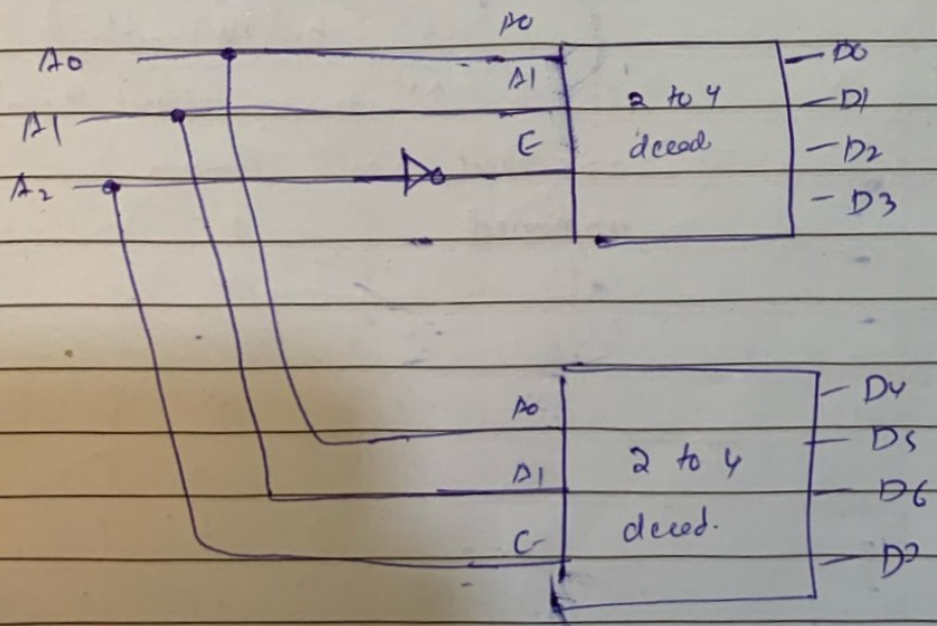


A_3 handle the Enable

If $A_3 = 0$ then first decoder will be Enable and 2nd will be disable

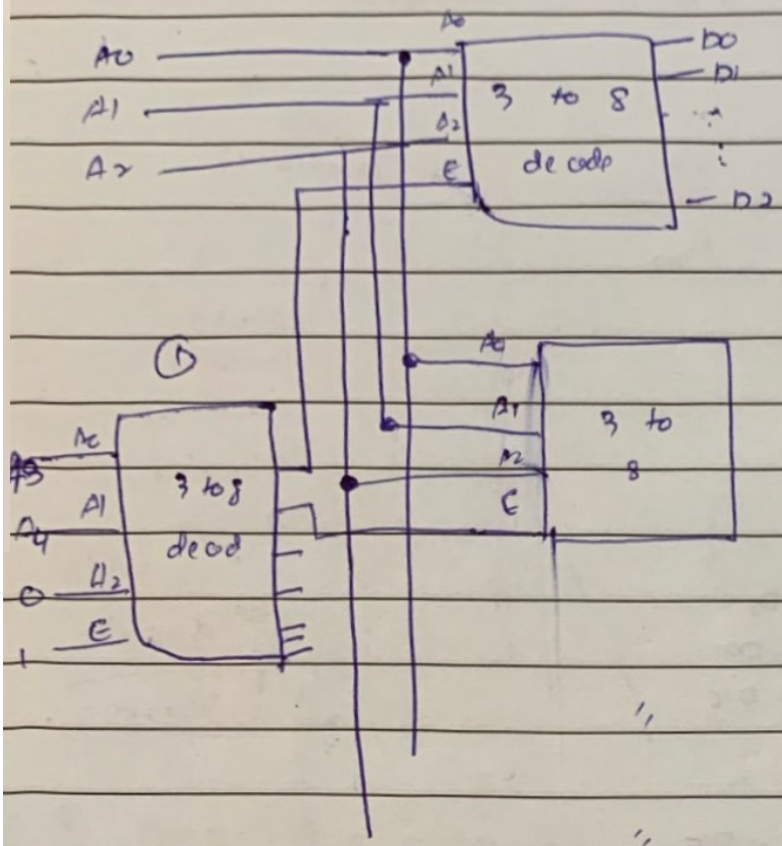
Vice versa

3 to 8 decoder using 2 to 4 decoder



Al Habib

Date 5 to 32 decoder using 3 to 8 - Decoder



Similarly all the enablers will be pinned by the output of the 1st decoder (first 4 pins) and rest 3 will remain unpinned

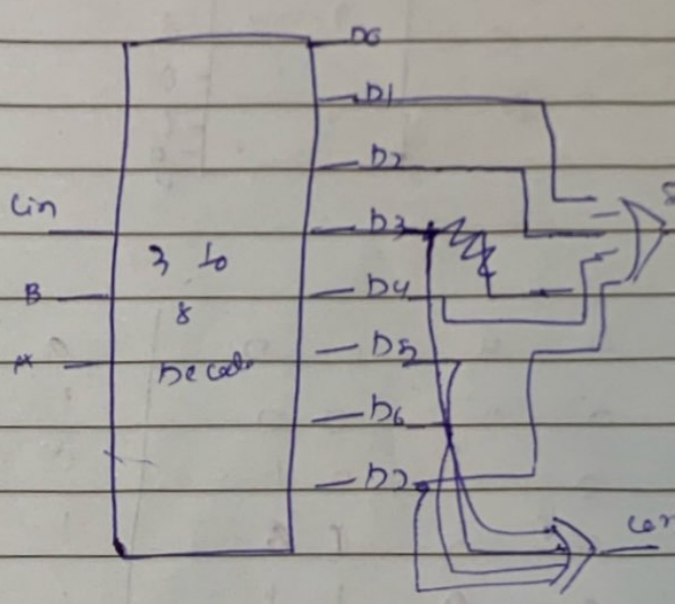
Date _____

2) implementing a logic circuit from decoder we use ~~or~~ 'OR' gate.

Ex Q

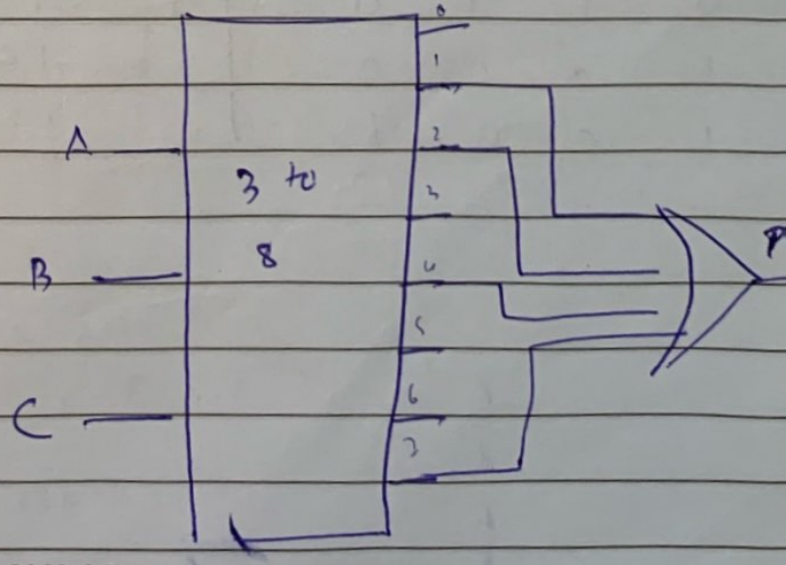
a full adder

A	B	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Even Parity checker

A	B	C	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

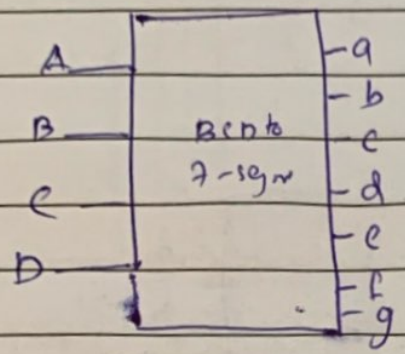


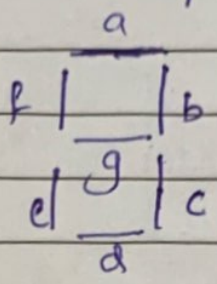
Al Habib

Date _____

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

BCD to 7 segment Decoder





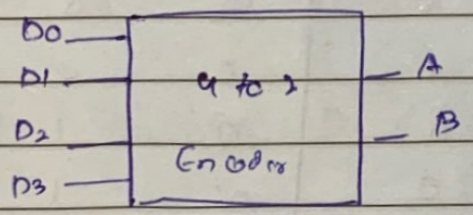
A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	0	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

Al Habib

Date _____

Encoders :- it has been assume that at any given time only one of the input is High and depending on the which input is High, we get the specific code at output

① 4 to 2 line Encoder



N = 2

input = 2^N =

D0	D1	D2	D3	A	B
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

Logic Circuit :

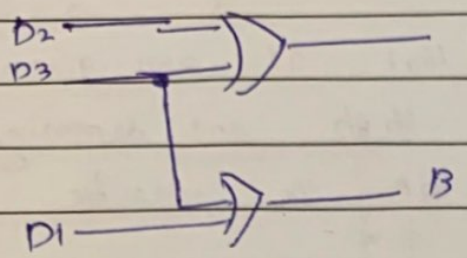
A : High when D2 & D3 ↑

B : " when D1 & D3 ↑

A = D2 + D3 , B = D1 + D3

Al Habib

Date _____



form 0 from A B C inputs

$n = 0$

$D_0 = 2^0$

$D_2 = 2^2 = n = 2$

A = 0 B = 1 C = 0

2 forms

8 line to 3 line encoder.

D0	D1	D2	D3	D4	D5	D6	D7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

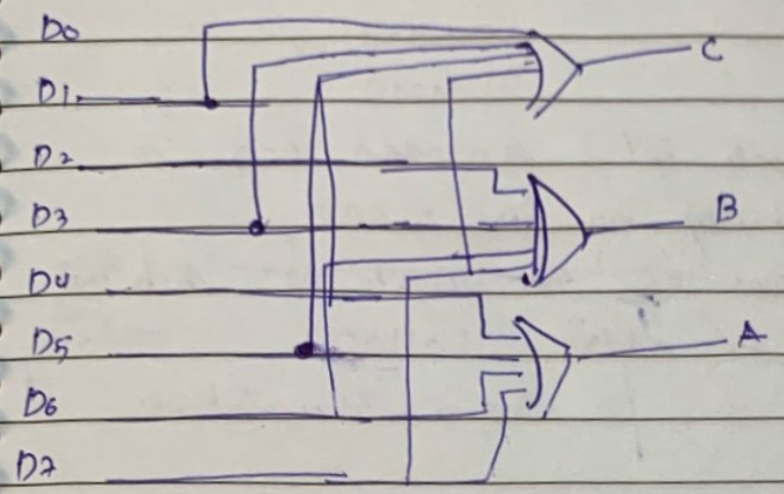
$A = D_4 + D_5 + D_6 + D_7$

$B = D_2 + D_3 + D_6 + D_7$

$C = D_1 + D_3 + D_5 + D_7$

Al Habib

Date



Decimal to BCD Encoder

D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	A	B	C	D
1	0	0	0	0	0	0	0	0	0	0	0	0	0
	1									0	0	0	1
		1								0	0	1	0
			1							0	0	1	1
				1						0	1	0	0
					1					0	1	0	1
						1				0	1	1	0
							1			0	1	1	1
								1		1	0	0	0
									1	1	0	0	1

$A = D8 + D9$

$B = D4 + D5 + D6 + D7$

$C = D2 + D3 + D6 + D7$

$D = D1 + D3 + D5 + D7 + D9$

Al Habib

Parallel bit adder & Comparator ICs.

Date

BCD to Binary conversion.

1) The value, or weight, of each bit in the BCD no: is represented by a binary number.
2) All of the binary representations of the weights of bits that are 1s in the BCD no: are added
3) The result of this addition is the binary equivalent of the BCD no:

4) The binary numbers representing the weights of the BCD bits are summed to produce the total binary no:

	Tens Digit				Unit Digit			
Weight:	80	40	20	10	8	4	2	1
Bit designation:-	B_3	B_2	B_1	B_0	A_3	A_2	A_1	A_0

Q

Convert the BCD numbr (01001111) (Decimal 27) to binary

Sol:

80	40	20	10	8	4	2	1
0	1	0	0	1	1	1	1

Al Habib

Date _____

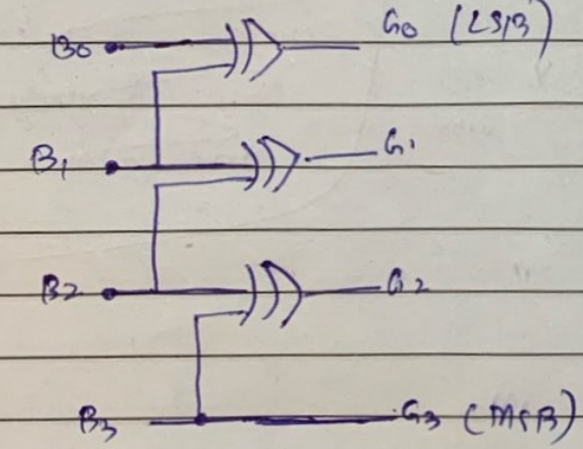
0000001 1
0000010 2
0000100 4
0010100 20

make 20 in binary → 0010100 (Binary)

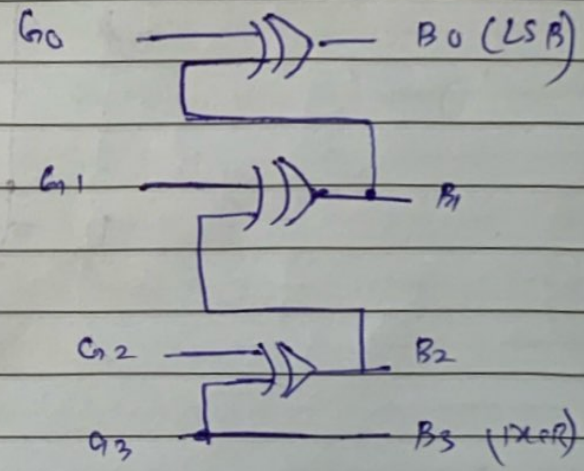
Binary to Gray

4-bit binary

Binary ⟶ Gray



Gray to Binary

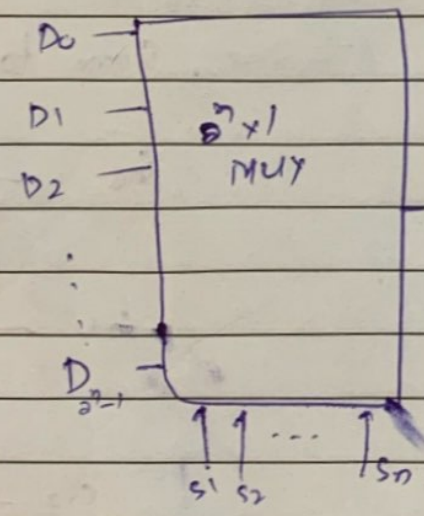


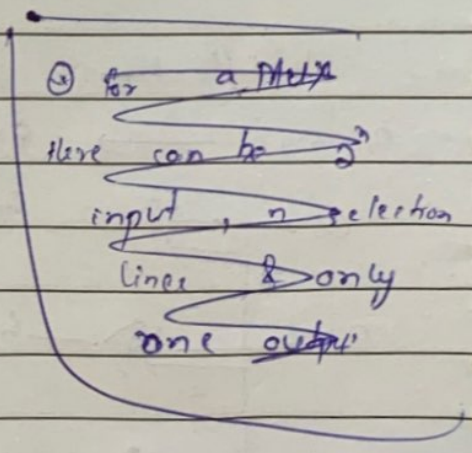
Date _____

Multiplexer (MUX)

* multiplexer is a combinational logic circuit used to select only one input among ~~several~~ several input based on selection line.

*) This can act as digital switch
*) This is also called as data selector
*) For a "MUX" there can be 2^n inputs, n selection line & only one output

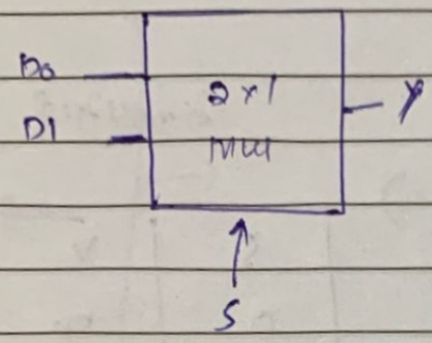


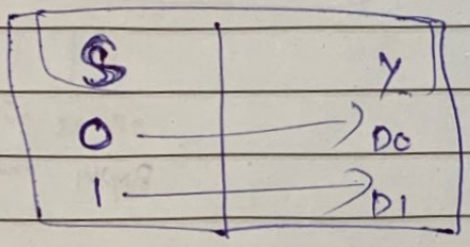


Date _____

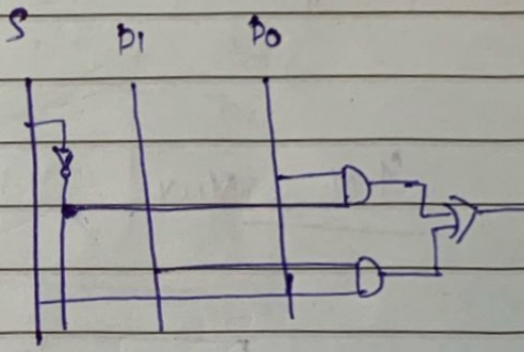
Q. 2×1 MUX:

⊙ when we are designing a mux, we consider selection lines as input, not the actual input

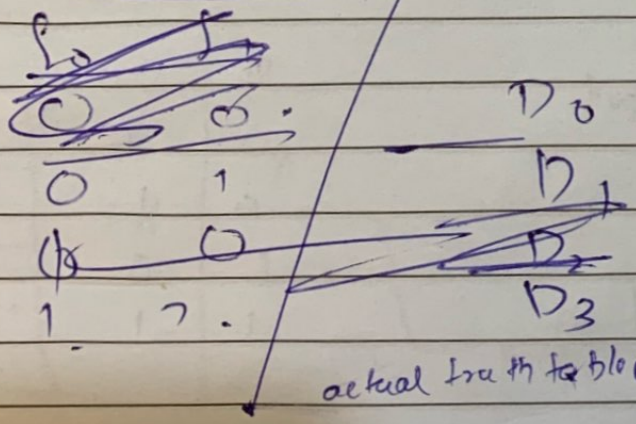




Simplified truth table



$Y = D_0\bar{S} + D_1S$



actual truth table

Date

$D_1 = 1 \rightarrow Y = 1$

$D_1 = 0 \rightarrow Y = 0$

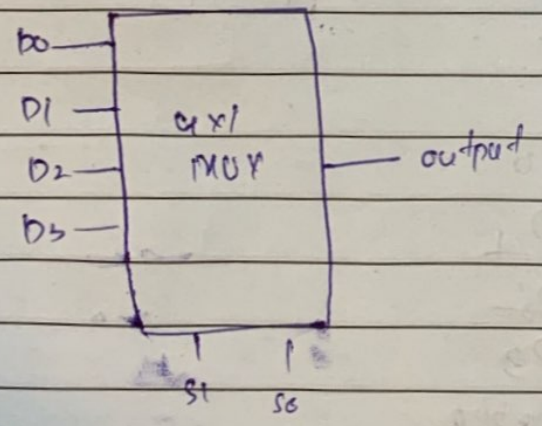
D_1	D_0	S	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

simplified

⇒

S	Y
0	D_0
1	D_1

4x1 MUX



simply make the Binary of $S_1 S_0$ ⇒ value will be $D_{(S_1 S_0)}$

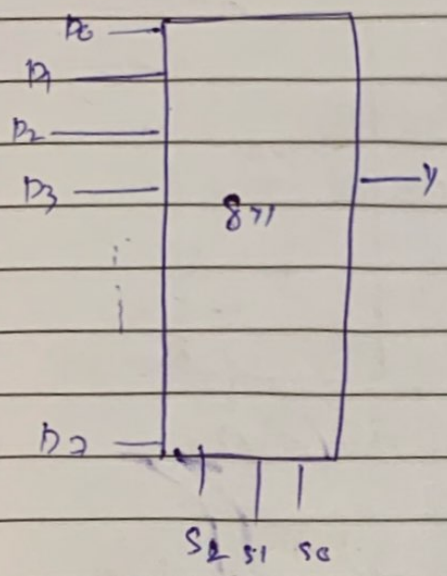
Truth Table

S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$Y = D_0 S_1 \bar{S_0} + D_1 S_1 \bar{S_0} + D_2 S_1 \bar{S_0} + D_3 S_1 S_0$

Date _____

8x1 MUX



7.7

S_2	S_1	S_0	Y
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

8x1 using 4x1

S_2 S_1 S_0

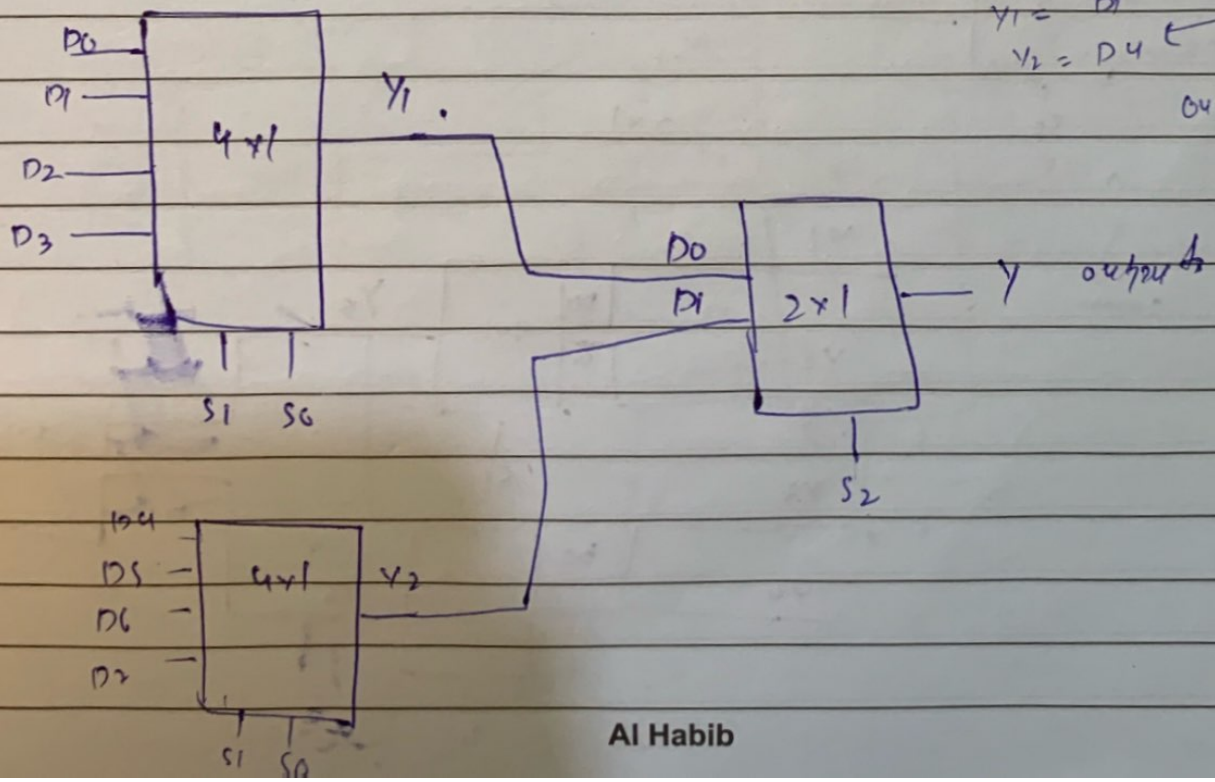
Eg 1 0 0

for

$Y_1 = D_1$ ← D_1

$Y_2 = D_4$ ← $S_2 = 1$

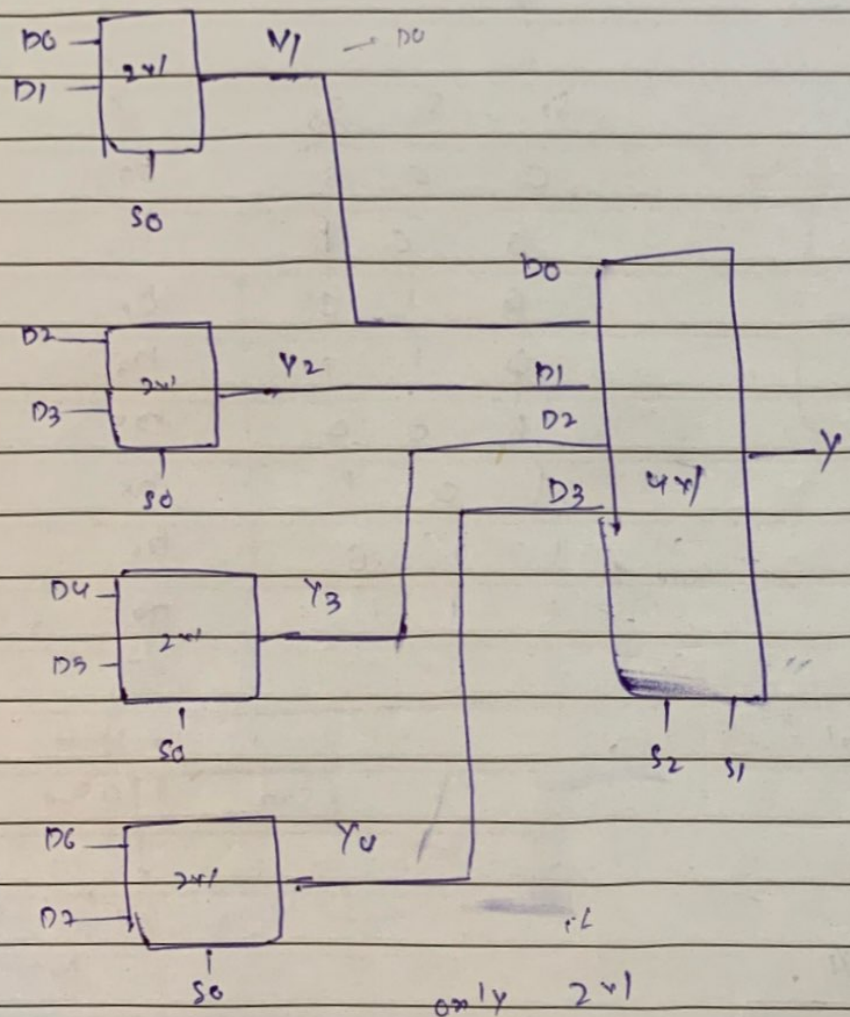
Output = D_4



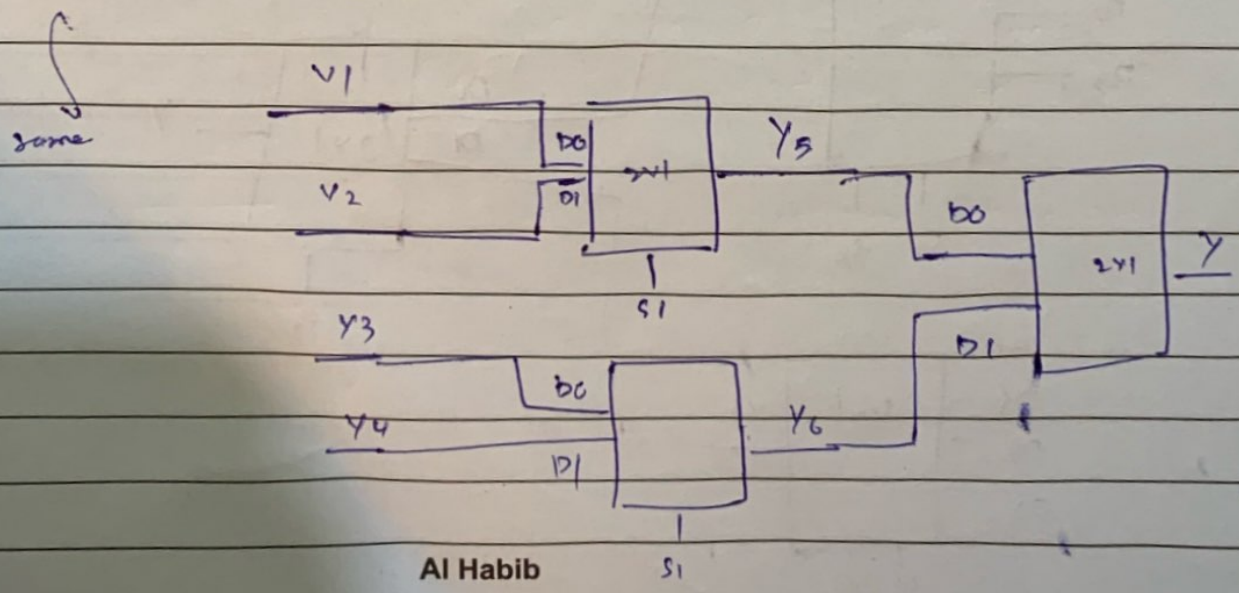
Al Habib

Date

8x1 using 2x1 & 4x1



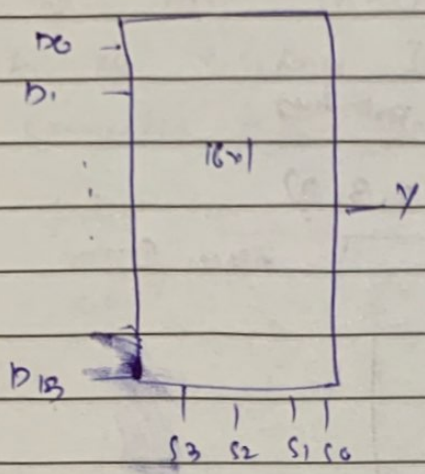
only 2x1



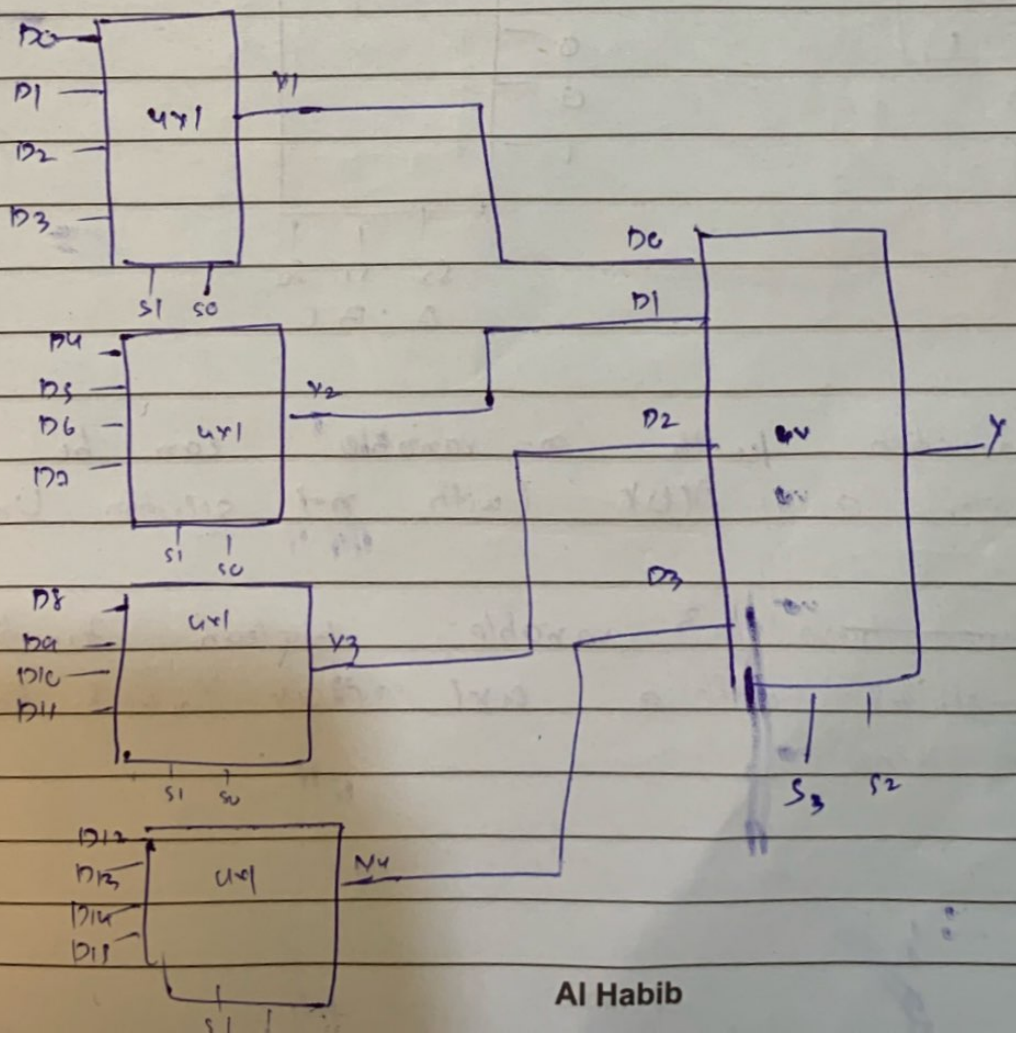
Al Habib

Date _____

16x1 MUX



using 4x1.



Al Habib

Date _____

implementation of Boolean function using mux.

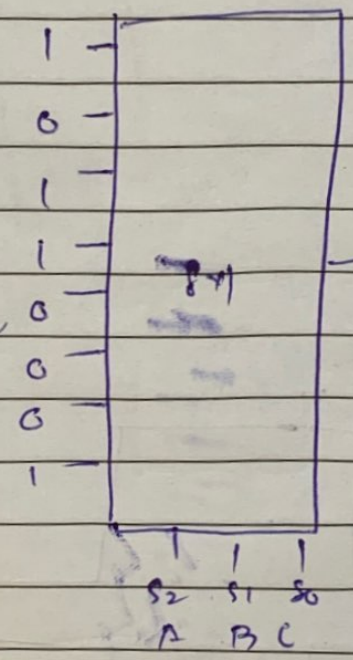
7.7

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

minterms

$F(A,B,C) = \sum m(0,2,3,7)$

place 1



A Boolean function with n- variable can be implemented with a MUX (with n-1 selection lines)

Ex: if we have 3 variable boolean function can be implemented with a 4x1 MUX (which has 2 selection lines)

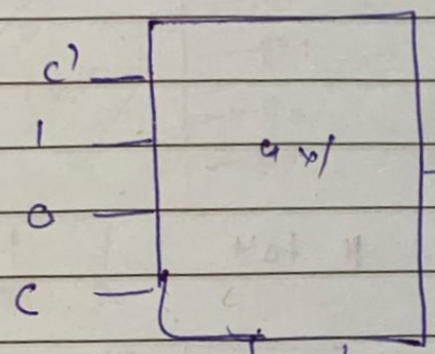
Date

or its complement

⇒ During the implementation, starting from the MSB the first (n-1) variables are connected to the selection lines and the last variable is connected to the data lines.

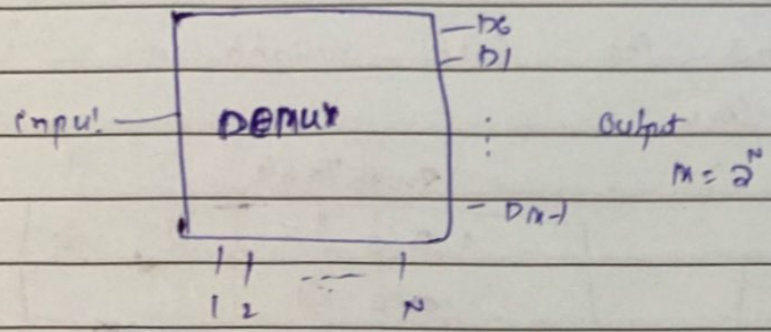
Truth Table

A	B	C	F	
0	0	0	1	
0	0	1	0	F = C'
0	1	0	1	
0	1	1	1	F = 1
1	0	0	0	F = 0
1	0	1	0	
1	1	0	0	F = C
1	1	1	1	

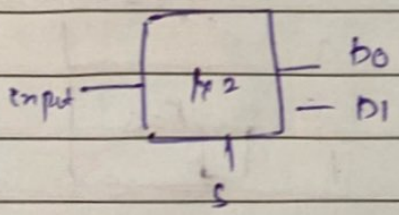


Date _____

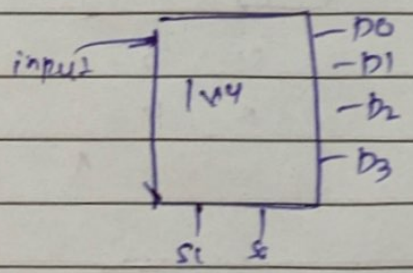
Demultiplexer 1 to Many.



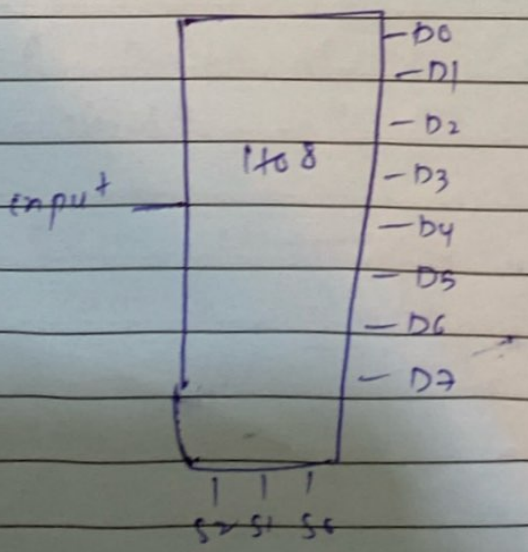
1 to 2



4 to 4



1 to 8

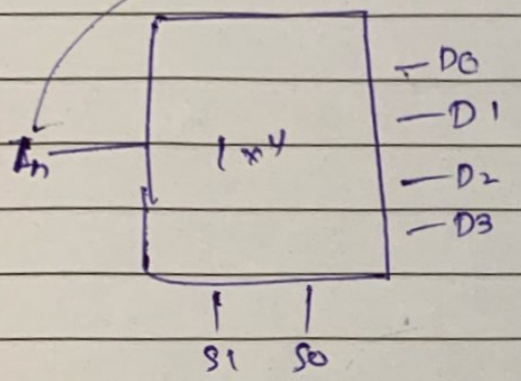


Al Habib

Date _____

(4) Decoder can also be used as demultiplexer

1×4 DEMUX.



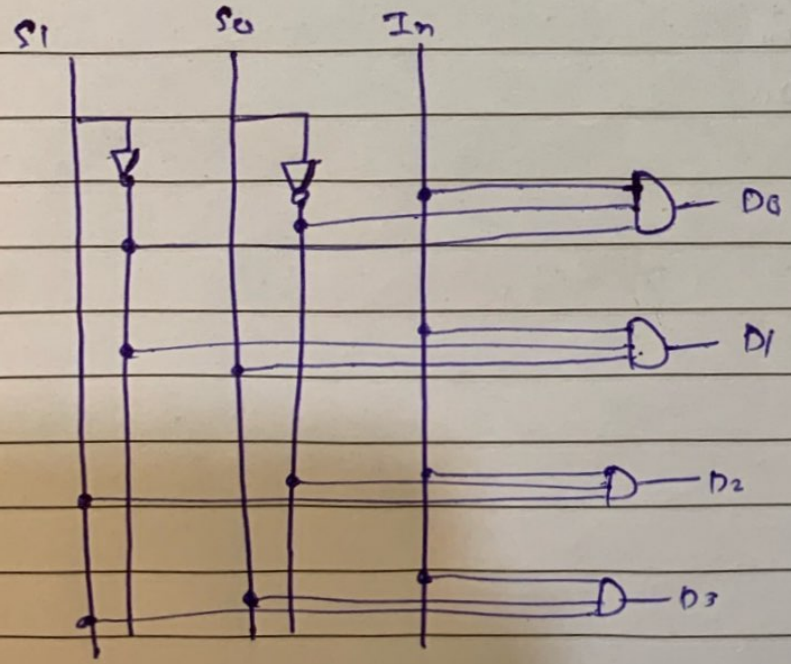
S1	S0	D0	D1	D2	D3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

$D_0 = In\ S_1\bar{S}_0$

$D_1 = In\ \bar{S}_1 S_0$

$D_2 = In\ S_1 \bar{S}_0$

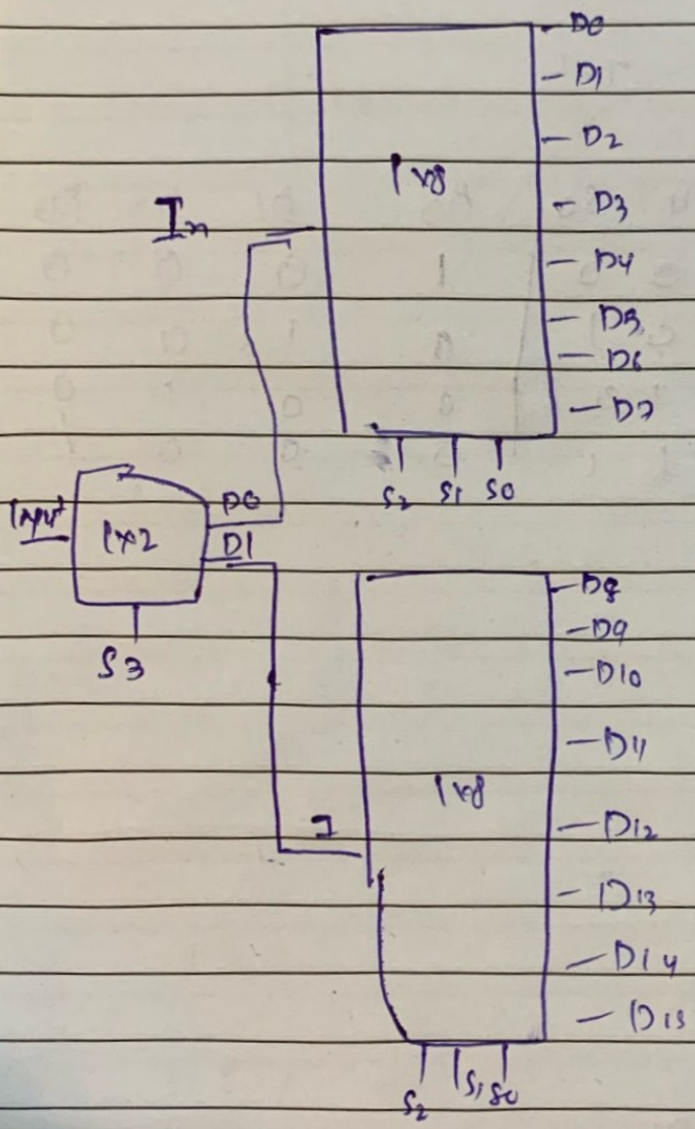
$D_3 = In\ S_1 S_0$



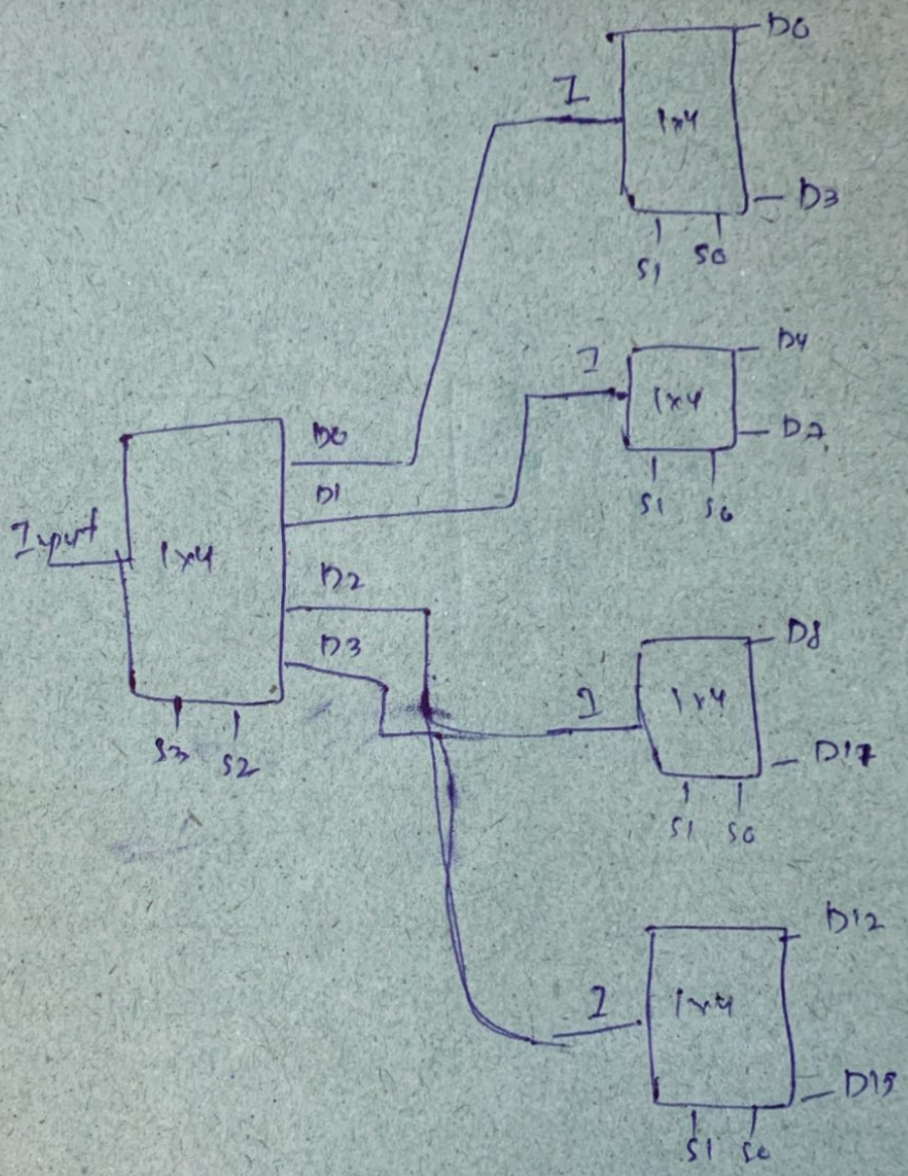
Al Habib

Date _____

1x16 using 1x8



1×16 way 1×4 DeMux



Basic circuit

Implementation — alternative