NAME: Soumyajit Bhowmik
Student ID: 23100100301O
Batch:- BEE2

---

1. Explain the basic logic gate expression (AND, OR, NOT, NAND, NOR, XOR) with truth tables and logic gate symbols, Design a logic gates ~~to implement~~ and Symbols. Design a logic ~~gate~~ circuit using only NAND gates to implement an XOR function.

Ans:- AND Gate :-

Symbol: 

Expression: $A.B = Y$

Truth Table:

A	B	output
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate :-

Symbol: 

Expression: $A+B = Y$

Truth Table :-

A	B	output
0	0	0
1	0	1
0	1	1
1	1	1

NOT Gate :-

Symbol: 

Expression: $\overline{A} = Y$

Truth Table :-

A	Output
0	1
1	0

NAND Gate:-

Symbol: 

Expression: $\overline{A.B} = Y$

Truth Table:

A	B	output
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate:-

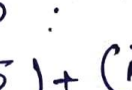
Symbol:

Expression: $\overline{A+B} = Y$

Truth Table:-

A	B	output
0	0	1
0	1	0
1	0	0
1	1	0

XOR Gate:-

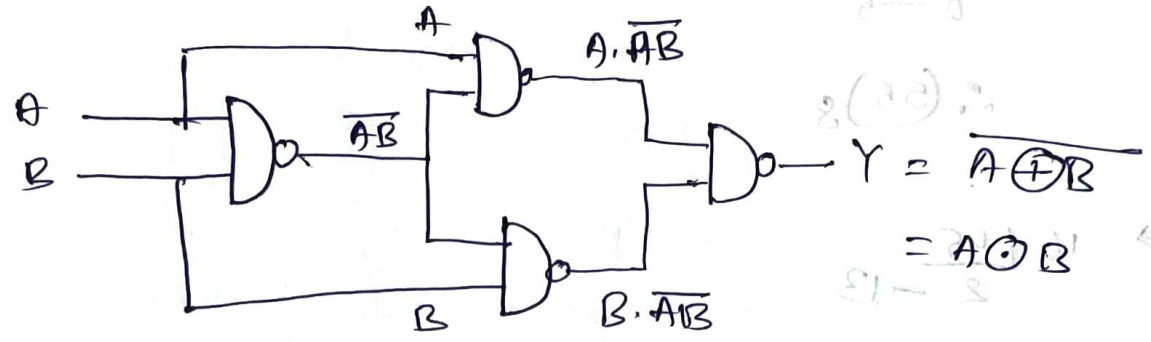
Symbol: 

Expression: $(A.\bar{B}) + (\bar{A}.B)$

Truth Table:-

A	B	output
0	0	0
1	0	1
0	1	1
1	1	0

XOR using NAND:-



Truth Table:-

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

2. Convert the following decimals numbers to binary, octal and hexadecimal

i) 45

ii) 255

Also perform 8-bit 2's compliment subtraction of 45-100 in binary.

Ans:- i) $(45)_{10} \rightarrow (?)_2$

Deviding by 2

$2 \mid 45$

$2 \mid 22 - 1$

$2 \mid 11 - 0$

$2 \mid 5 - 1$

$2 \mid 2 - 1$

$1 - 0$

$\therefore (101101)_2$

→ 8 | 45
8 | 5 — 5
0 — 5

∴ $(55)_8$

→ 16 | 45
2 — 13

∴ $(2D)_{16}$

ii) $(255)_{10} \rightarrow (?)_2$

2 | 255
2 | 127 — 1
2 | 63 — 1
2 | 31 — 1
2 | 15 — 1
2 | 7 — 1
2 | 3 — 1
1 — 1

∴ $(11111111)_2$

→ 8 | 255
8 | 31 — 7
3 — 7

∴ $(377)_8$

→ ~~18 | 255~~ 16 | 255
15 — 15

∴ $(15F)_{16}$

Now, 45 in decimal 00101101
100 in decimal 01100100

1's complimend 100 → 10011011

For 2's Compliment → 10011011
+ 1
10011100

Now, adding 2's complimend 100 & 45

00101101
+ 10011100
11001001 → result

The result 11001001 is in 2's compliment form, since the MSB is 1, the number is negative.

Now, 1's compliment of result 00110110 & adding 1 to it we get,

00110110
+1
00110111

which is desimal of 55

Thus, the final value is $\boxed{-55}$. Ans

3. Differentiate between TTL, Schottky TTL, and CMOS logic families in terms of speed, power, consumption and fan-out. Also explain how CMOS and TTL families can be interfaced.

Ans:

Parameters	TTL	Schottky TTL	CMOS
→ Speed	Moderate (1050ns delay)	High (310ns) delay	Very High (0.010)ns
→ Power consumption	High (mW range)	Lower than TTL	Very Low (W range)
→ Fan-out	-10	~1020	very high (> 50)

To interface CMOS and TTL logic families, consider the following method:-

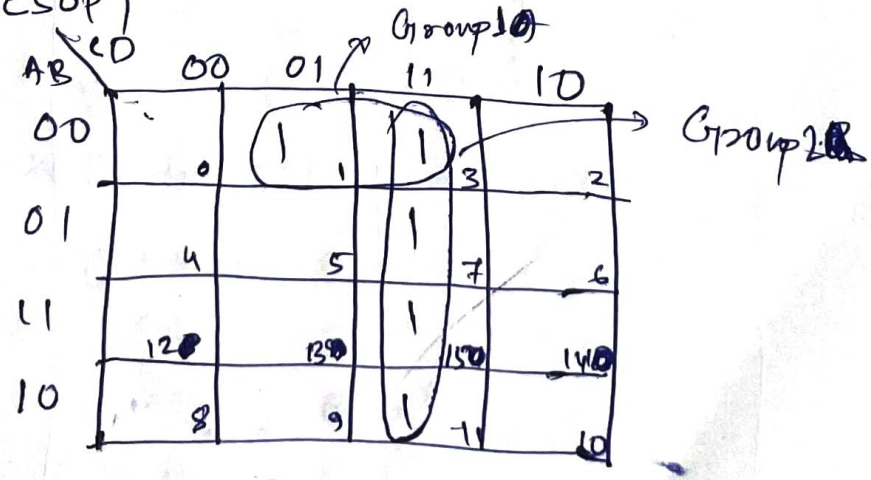
1) CMOS to TTL interface:- When both derice operates at same 5V supply, a CMOS output can drive a TTL input directly. However, if the CMOS operates at higher voltage, a level sifter or CMOS buffer is reccommend to ensure voltage compatibility.

2) TTL to CMOS:- TTL outputs may not meet the voltage levels required for CMOS inputs. A common solution is to use a pull-up resistor to raise the TTL output to a sufficient voltage of CMOS inputs. Alternatively, an open-collector TTL buffer can be used to interface with CMOS.

These techniques ensure proper voltage and current levels facilitating reliable communication between CMOS & TTL devices.

4. Simply the boolean function $F(A,B,C,D) = \Sigma(1,3,7,11,15)$ using a 4 variable Karnaugh map. Implement the simplified expressing using basic gates.

Ans:- Since, the highest value of $F(A,B,C,D) = \Sigma(1,3,7,11,15)$ is 15, we are going to choose a 4×4 map {following csop}

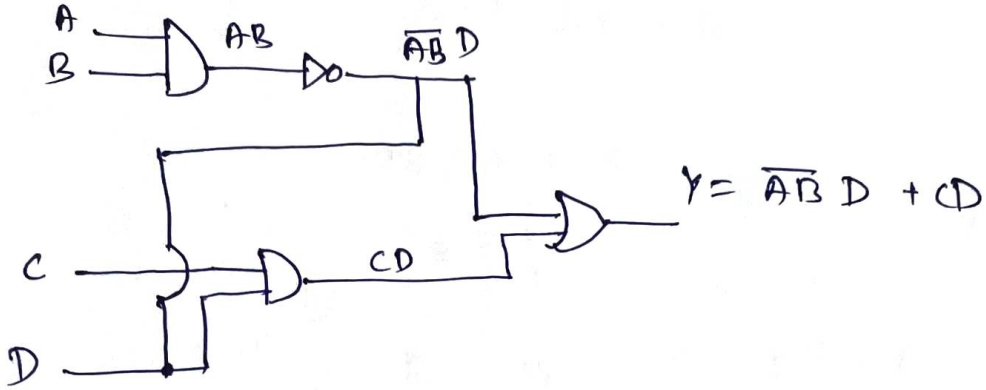


Nomenclature of Group 1 → $\bar{A}\bar{B}D$

Nomenclature of Group 2 → CD

$\therefore Y = \bar{A}\bar{B}D + CD$

$Y = (\bar{A}\bar{B}D + C)D$



5) Design a 4:1 multiplexer using logic gates. Use the multiplexer to implement the function $F(A, B) = A$ XOR B

Ans:-

$S_0S_0 = 00$, $Y = I_0$

$S_0S_1 = 01$, $Y = I_1$

$S_1S_0 = 10$, $Y = I_2$

$S_1S_1 = 11$, $Y = I_3$

$\therefore A \oplus B$

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Mapping XOR to 4:1 Mux inputs.

$I_0 = 0$ $(A=0, B=0)$

$I_1 = 1$ $(A=0, B=1)$

$I_2 = 1$ $(A=1, B=0)$

$I_3 = 0$ $(A=1, B=1)$

Now, selecting inputs using select lines,

$S_1 = A$

$S_0 = B$

So, i) A = 0 and B = 0, $S_0S_0 = 00$

ii) A = 0 and B = 1, $S_0S_1 = 01$

iii) A = 1 and B = 0, $S_1S_0 = 10$

iv) A = 1 and B = 1, $S_1S_1 = 11$

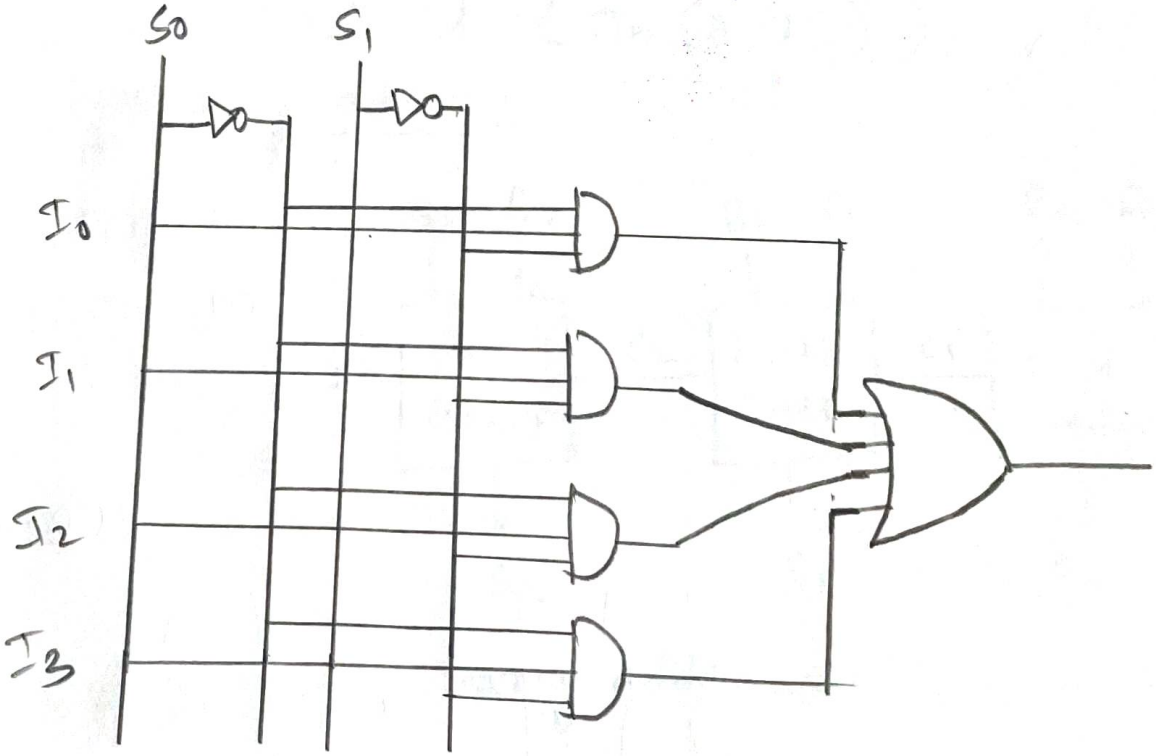
Hence, i) $Y_1 = I_0 = 0$

ii) $Y_2 = I_1 = 1$

iii) $Y_3 = I_2 = 1$

iv) $Y_4 = I_3 = 0$

which produces desired XOR input.



6) Explain the working of a full adder. Design a 4-bit ripple carry adder using full adder circuits. What are its limitations? suggest how a carry look-ahead adder can improve performance.

Ans:- A full adder is a digital circuit that completes the sum of three binary bits: two significant and a carry input It produces two outputs.

Sum(S): The least significant bit of the addition.

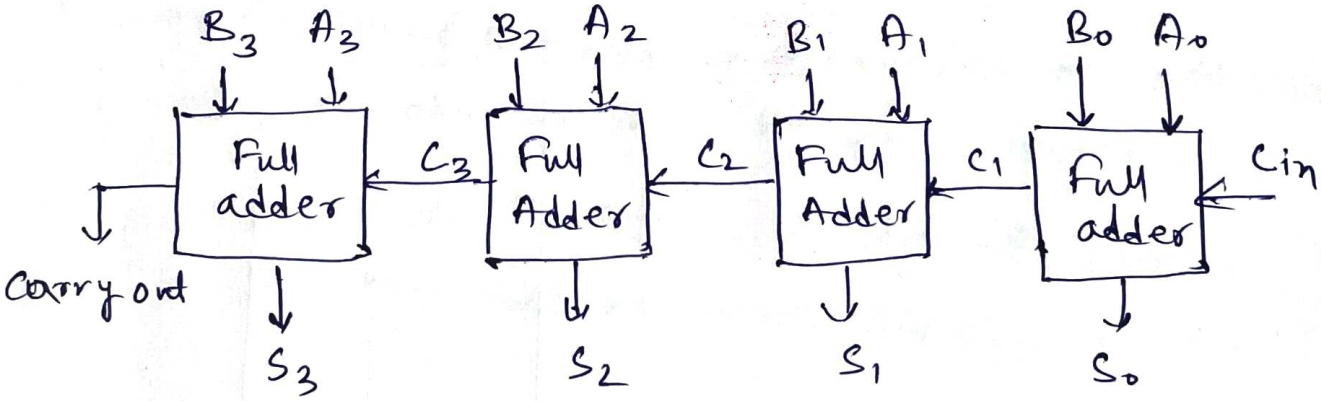
Carry out:- The most significant bit, which is carried to the next addition stage.

Boolean equations:-

$$Sum = (A \oplus B) \oplus C_{in}$$

$$C_{out} = (A \cdot B) + (C_{in}(A \oplus B))$$

4-Bit Ripple Carry adder:-



→ Limitation of ripple carry adder:-

i) Propagation delay:- For an n-bit adder, the delay is proportional to n. Each full adder must wait for the carry from the previous stage leading to a cumulative delay.

ii) Speed:- As the bit-width increases, the delay becomes significant, reducing the overall speed of the addition operation.

iii) Scalability :- Not ideal for a large bit-width additions due to ~~inder~~ increase delay.

→ Carry look Ahead Adder :- (CLA)

A carry look-ahead adder improves upon the Ripple carry Adder by reducing the carry propagation delay.

Working :-

→ Generate :- indicates if a carry will be generated at bit position.

→ Propagate : indicates if a carry input to bit position will propagate to the next bit.

Advantages :-

→ Reduce delay :- By calculating carries in parallel, the overall delay is significantly reduced.

→ Improved speed :- Enhances the speed of addition operates, specially for large bit width numbers.

7) Design and explain a BCD to 7-segment decoder circuit. What role does a driver IC play in this context?

Ans :- A. BCD to 7-segment decoder is a combination circuit that transforms a 4 bit BCD input representing desimal digit into the signal nesseasary to creative the segments of a 7-segment display displaying the corresponding digit.

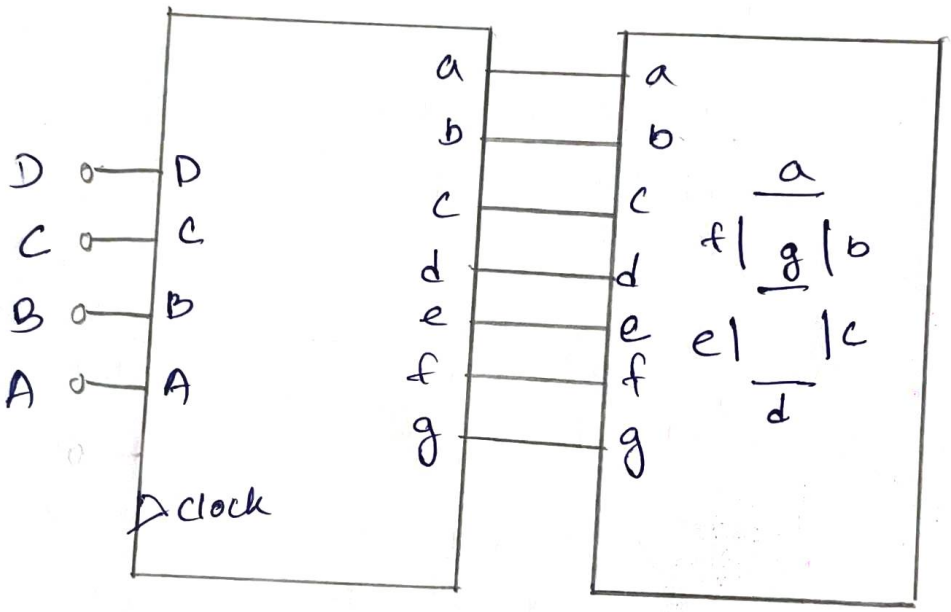
working of BCD to 7-segment decoder :-

BCD input :- the circuit recivers a 4 bit BCD input (A, B, C, D), where each bit represents a power of 2 ($2^3, 2^2, 2^1, 2^0$)

Truth Table :- A truth table defines the relationship between the BCD input and the 7-segment display segment output - (a, b, c, d, e, f, g)

Logic Gates :-

The output signals are implemented. using logic gates (AND, or NOT) to translate the BCD input into the ~~approximal~~ appropiciate segment activation.



BCD to 7-segment decoder

7-Segment LED Display

Role of Driver IC in BCD to 7-segment Decoding :-

1) Signal conversation :- It converts the 4-bit BCD input to the appropriate signal for the 7-segment display.

2) Current ~~Sour~~ Sinking :- The IC sink current to drive the LED's in the display ensuring illumination display.

3) Control features :- Many drivers ICs offers additional features like ramp test (LT), blanking and Latch enable (LE) for enhanced control over the display.