COMPUTER SCIENCE & IT

DIGITAL LOGIC

Lecture No. 03

Combinational Circuit

By- Chandan Gupta Sir

Parallel adder

Look Ahead carry adder

Look Ahead Carry adder:

$$A = a_3\, a_2\, a_1\, a_0$$
$$B = b_3\, b_2\, b_1\, b_0$$
$$\quad\; C_3\, C_2\, C_1\, C_0$$
$$\overline{\quad C_4\, S_3\, S_2\, S_1\, S_0 \quad}$$

$$\boxed{\begin{aligned} C_{i+1} &= P_i C_i + Q_i \\ P_i &= (a_i \oplus b_i) \\ Q_i &= a_i \cdot b_i \end{aligned}}$$

$$C_1 = (a_0 \oplus b_0) C_0 + a_0 b_0$$

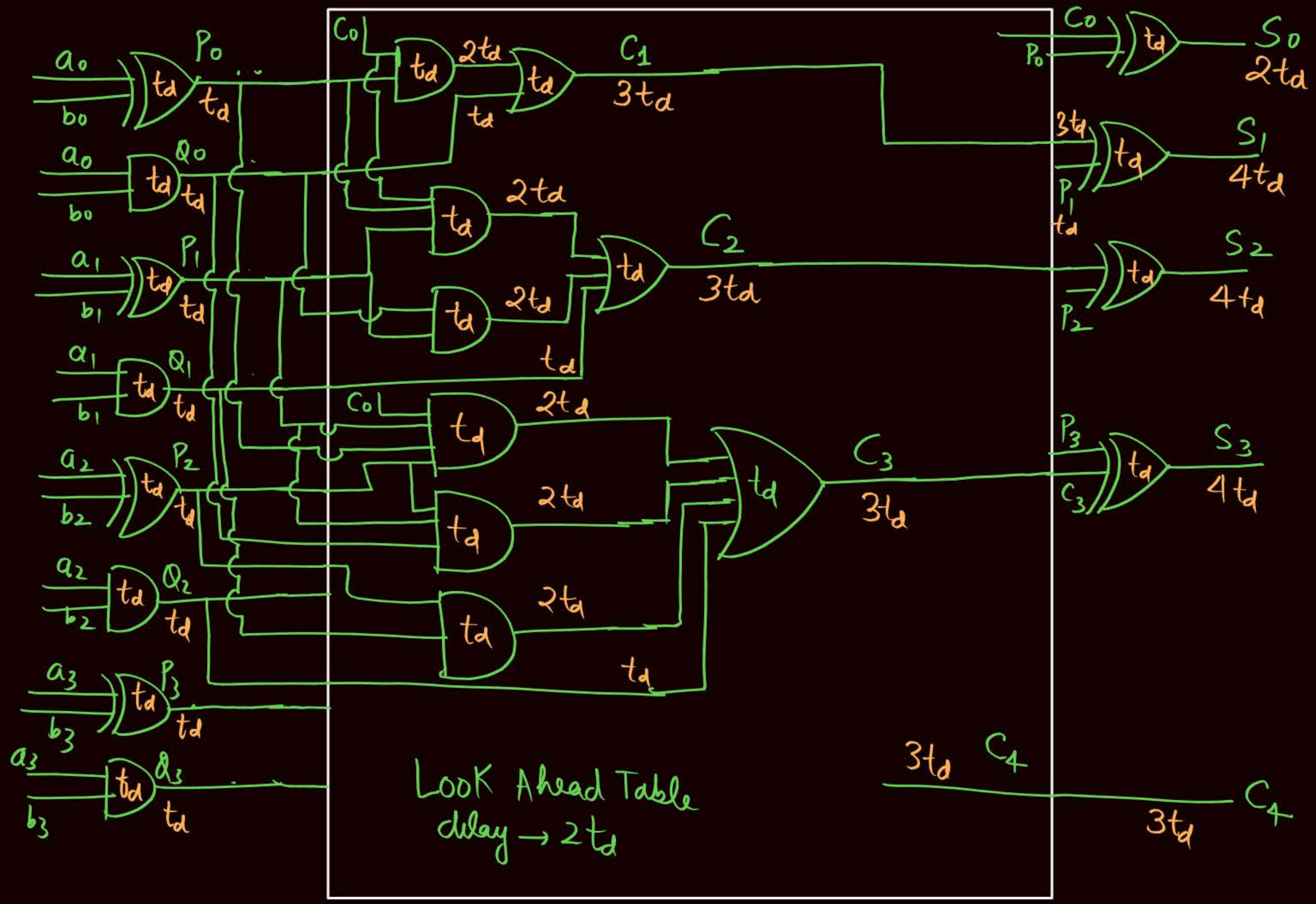
$$C_2 = (a_1 \oplus b_1) C_1 + a_1 b_1$$

$$C_1 = P_0 \cdot C_0 + Q_0$$

$$C_2 = P_1 \cdot C_1 + Q_1 = P_1 [P_0 C_0 + Q_0] + Q_1 = P_1 P_0 C_0 + P_1 Q_0 + Q_1$$

$$C_3 = P_2 C_2 + Q_2 = P_2 [P_1 P_0 C_0 + P_1 Q_0 + Q_1] + Q_2 = P_2 P_1 P_0 C_0 + P_2 P_1 Q_0 + P_2 Q_1 + Q_2$$

$$C_4 = P_3 C_3 + Q_3 = P_3 [P_2 P_1 P_0 C_0 + P_2 P_1 Q_0 + P_2 Q_1 + Q_2] + Q_3 = P_3 P_2 P_1 P_0 C_0 + P_3 P_2 P_1 Q_0 + P_3 P_2 Q_1 + P_3 Q_2 + Q_3$$



Overall delay $= 4t_d$

*** independent of no. of bits

Look ahead table utilizes two-stage AND-OR operation.

No. of AND gates used in Look ahead table $\Longrightarrow$ $N_A = 1+2+3+ \cdots +n$

$$N_A = \frac{n(n+1)}{2}$$

No. of OR gates used in Look ahead table

$$N_{OR} = n - \text{OR gates}$$

highest i/p OR gate $\longrightarrow$ $(n+1)$ i/p OR gate.

Here both no.s are of n bits $\longrightarrow$

$A = a_{n-1} \cdots a_0$

$B = b_{n-1} \quad b_0$

$$A = a_3\ a_2\ a_1\ a_0$$

$$B = b_3\ b_2\ b_1\ b_0$$

$$\begin{array}{r} B_3\ B_2\ B_1\ \\ \hline B_4\ D_3\ D_2\ D_1\ D_0 \end{array}$$

$$A = 1011 = 11$$

$$B = 0111 = 7$$

$$\begin{array}{r} 1000 \\ \hline 00100 \\ \hline \end{array} = +4$$

+ve

$$\Rightarrow \quad A = 0110 = \underline{\underline{6}}$$

$$B = 1011 = \underline{\underline{11}}$$

$$\begin{array}{r} 011\ \\ \hline 11011 \\ \hline \end{array} \longrightarrow -[2's\ compliment] = -[00101] = (-5)$$

-ve

Result

[Half Subtractor]

- What is Half subtractor and what are input lines and what are output lines ?

⇒ 2-i/p lines ⟶ Corresponding bits of the two no.s x & y

$x = a_3$
$y = b_3$
$x = a_2$
$y = b_2$
$x = a_1$
$y = b_1$

⇒ 2-o/p lines ⟶ Borrow o/p – B ⟶ if for subtraction borrow is needed from next bit then borrow o/p will be '1' otherwise 0.

Difference o/p ⟶ after taking borrow ⟶ the result of subtraction is difference o/p D.

$(x-y)$

Input lines		Output lines	
x	y	D-o/p	B-o/p
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$D(x,y) = \Sigma(1,2) = \pi(0,3) = x \oplus y$

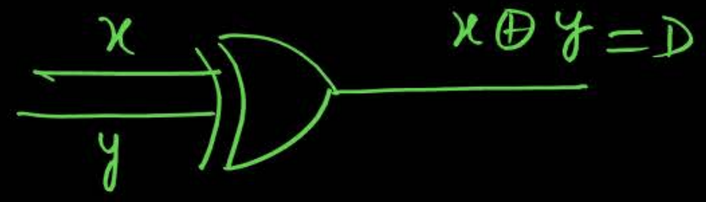
$\hookrightarrow$ non self dual

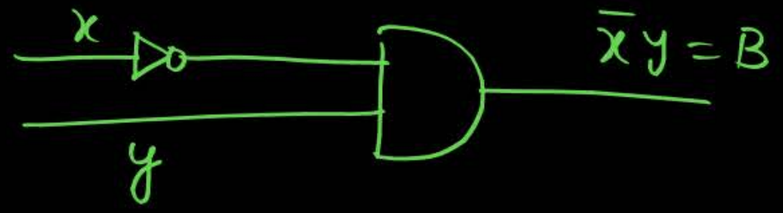
$B(x,y) = \Sigma(1) = \pi(0,2,3)$

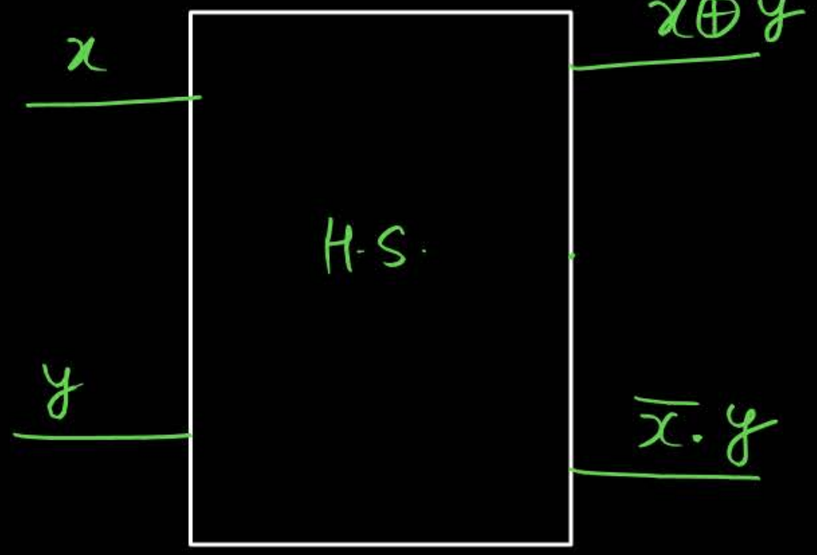
$= \bar{x}y$

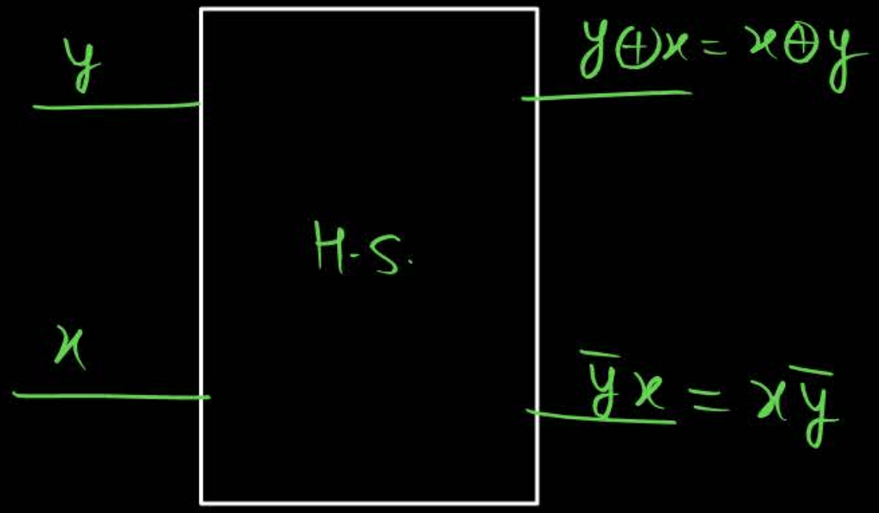
non self dual

Implementation using gates :



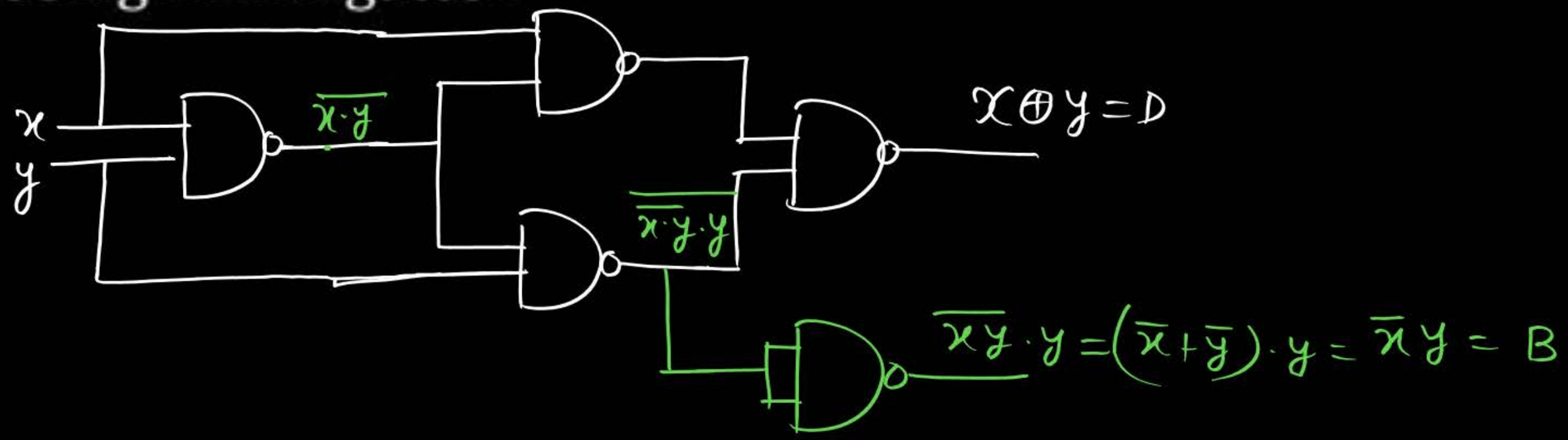






- Using NAND gates :

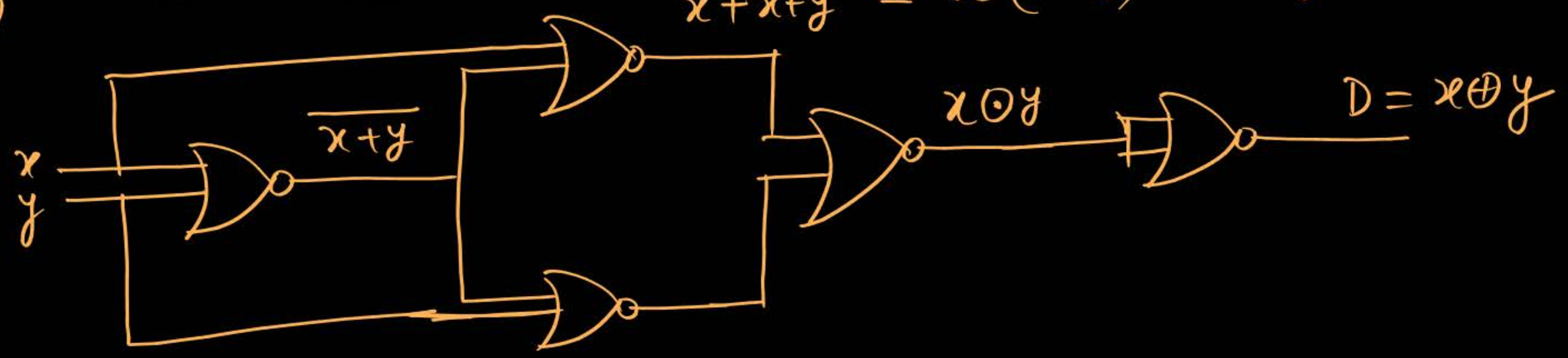
$(x-y)$



- Using NOR gates :

$(x-y)$

$$\overline{x+\overline{x+y}} = \bar{x}\cdot(x+y) = \bar{x}y = B$$



Note :→ To implement H.S. we require 5 (2-i/p NAND gate) or 5 (2-i/p NOR) gates.

Full Subtractor

- What is full subtractor and what are input lines and what are output lines ?

⇒ 3-i/p lines ⟶ Corresponding bits of the two no.s x & y and borrow o/p from previous subtraction z $(x-y-z)$

2-o/p lines ⟶ Borrow o/p B ⟶ will be '1' if borrow is required for $(x-y-z)$ operation from the next bit otherwise '0'.

⟶ Difference o/p D ⟶ final result of $(x-y-z)$ after incorporating the borrow.

$x-y-z$

Input lines			Output lines	
x	y	z	D-o/p	B-o/p
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$D(x, y, z) = \Sigma(1,2,4,7) = x \oplus y \oplus z$

↳ self dual boolean function

$B(x, y, z) = \Sigma(1,2,3,7)$

↳ self dual boolean function

$= \bar{x}\bar{y}z + \bar{x}y\bar{z} + \bar{x}yz + xyz$

$= \bar{x}z + \bar{x}y + yz$

$= \bar{x}y + yz + \bar{x}z$

$= \bar{x}y + (\bar{x} \oplus y)z$

$= \bar{x}y + (\overline{x \oplus y})z$

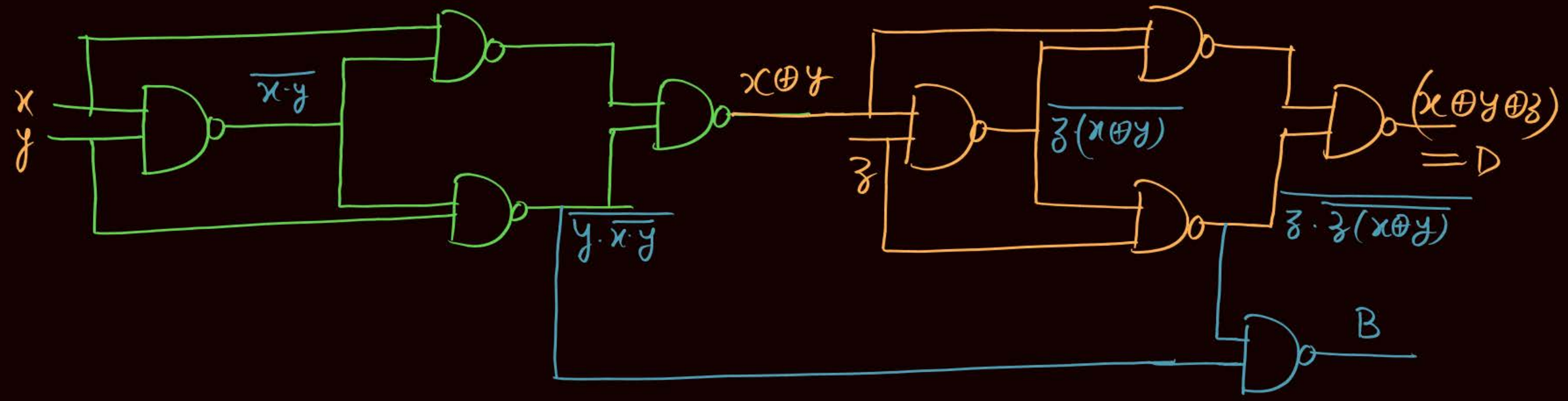
$= \bar{x}y + (x \odot y)z$

$= \bar{x}\bar{y}z + \bar{x}y\bar{z} + \bar{x}yz + xyz = \bar{x}y + z(x \odot y)$

$xy + (x \oplus y)z$

Implementation using gates :

$= \bar{x}y + (\bar{x} \oplus y) \cdot z = \bar{x}y + (x \odot y)z$



$$= y \cdot \overline{xy} + z \cdot \left[\overline{z \cdot (x \oplus y)}\right]$$
$$= y(\bar{x} + \bar{y}) + z\left[\bar{z} + \overline{x \oplus y}\right]$$
$$= \bar{x}y + (x \odot y) \cdot z$$
$$= \bar{x}y + (\bar{x} \oplus y) \cdot z$$

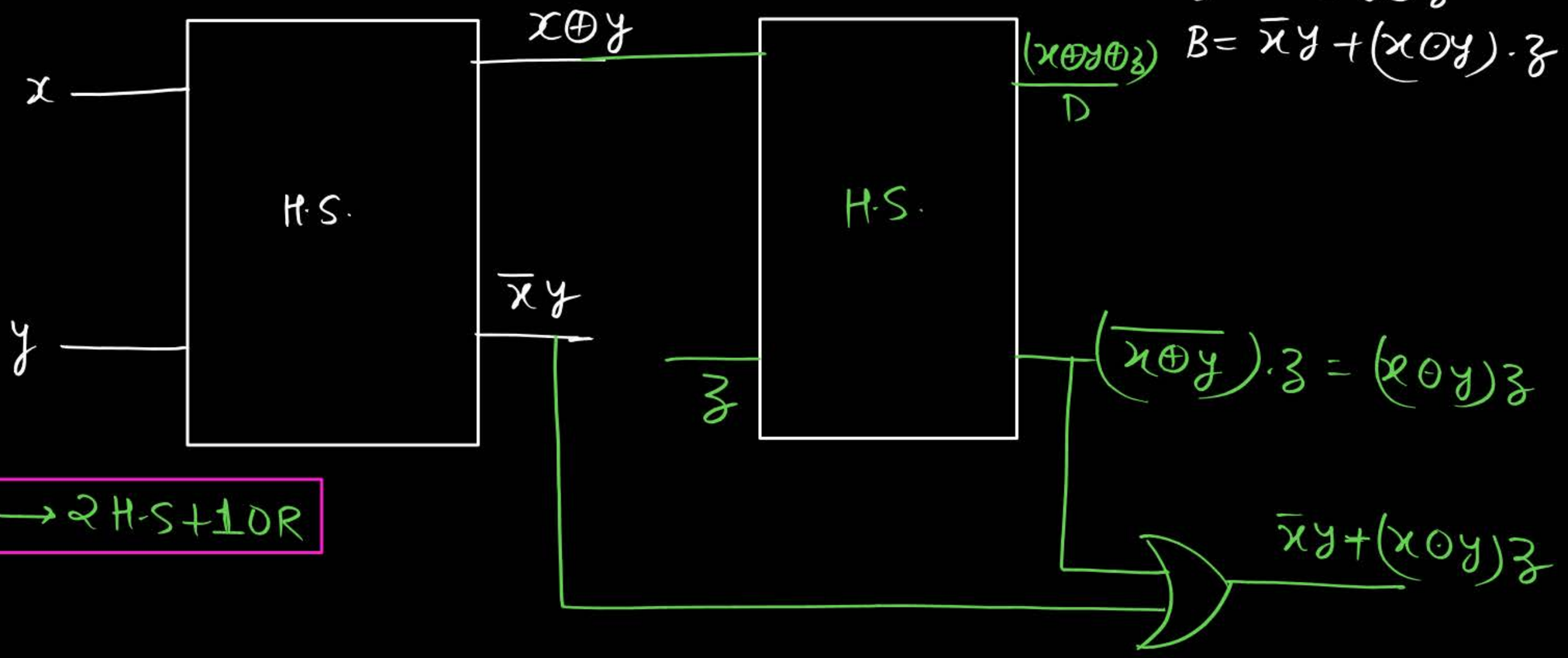
$\Rightarrow$ To implement F.S. we require 9 [2-i/p NAND] or 9 - [2-i/p NOR gates]

- Implementation using H.S and OR gate :

F.S. $\longrightarrow (x-y-z)$

$D = x \oplus y \oplus z$

$B = \bar{x}y + (x \odot y) \cdot z$



$\boxed{1\,F.S. \longrightarrow 2\,H.S + 1\,OR}$

- To subtract two n-bit numbers,, number of H.S. and number of OR gate required is :

$$
\begin{array}{r}
a_3\,a_2\,a_1\,a_0 \\
b_3\,b_2\,b_1\,b_0 \\
B_3\;\;B_2\;\;B_1\;\;0 \\
\hline
D_4\,D_3\;\;D_2\;\;D_1\;\;D_0
\end{array}
$$

$$= (n-1)\,F.S. + 1\,H.S.$$

$$= (n-1)[2\,H.S + 1\,OR] + 1\,H.S.$$

$$= 2(n-1)\,H.S + (n-1)\,OR + 1\,H.S$$

$$= (2n-1)\,H.S + (n-1)\,OR$$

Question

Two subtract two 4-bit numbers, minimum number of H.S. required is m and minimum number of OR gates require is n then value of $(m + n)$ is __10__.

$\Rightarrow$ 3 F.S. + 1 H.S.

$\Rightarrow$ 6 H.S + 3 OR + 1 H.S

= 7 H.S + 3 OR

$\Rightarrow m = 7$

$n = 3$

$(m+n) = 10$

2 Minute Summary

- Look ahead carry adder
- H.S. & F.S.

Thank you GW Soldiers !