Adder

1. Half Adder

Combinational Logic Design
Binary Adder-Subtractor

Objectives:

1. **Half Adder.**
2. **Full Adder.**
3. **Binary Adder.**
4. **Binary Subtractor.**
5. **Binary Adder-Subtractor.**

If you recall, what we have discussed so far; We have looked at different ways of realizing/writing a given function in the form of an expression; sum of products, product of sums, canonical SOP, canonical POS etc.

And we talked about several methods using which you can minimize the size of the expression so that the resulting gates level realization becomes cheaper in some respect : Number of gates can be less and the size(number of inputs to the gate) of each gate can also be smaller.

Now, let us look at some real circuits and how you can design those circuits and apply them in various applications. So, we shall be looking at the design of such basic building blocks which can be used to build larger systems.

We have already seen some fundamental circuits, like,

- Multiplexer
- Decoder
- Demultiplexer
- Encoder, etc.

Now we look at the most fundamental thing that we need in many many applications.

Think of an Adder, we need to add numbers in almost any applications you can think of. So, we shall be talking about the design of adders.

We will study Addition of Unsigned Numbers. After that we will see how to handle addition of signed numbers, using the same circuits we design for Addition of Unsigned Numbers.

Unsigned Addition ⟹ most of the part of this lecture.

Signed Addition
Subtraction
} at the end

Digital computers perform a variety of information-processing tasks. Among the functions encountered are the various arithmetic operations. The most basic arithmetic operation is the addition of two binary digits. Adder is a digital logic circuit that implements addition of binary numbers. Adder's circuit forms a basic component of ALU (Arithmetic Logic Unit). This simple addition consists of four possible elementary operations: $0 + 0 = 0$, $0 + 1 = 1$, $1 + 0 = 1$, and $1 + 1 = 10$. The first three operations produce a sum of one digit, but when both augend and addend bits are equal to 1, the binary sum consists of two digits. The higher significant bit of this result is called a carry . When the augend and addend numbers contain more significant digits, the carry obtained from the addition of two bits is added to the next higher order pair of significant bits. A combinational circuit that performs the addition of two bits is called a half adder . One that performs the addition of three bits (two significant bits and a previous carry) is a full adder .

In electronics an adder is digital circuit that perform addition of numbers. In modern computer adder reside in the arithmetic logic unit (ALU).
Adders are important not only in the computer but also in many types of digital systems in which the numeric data are processed.

We first study the half adder circuit, from which we develop the full adder. Connecting n full adders in cascade produces a binary adder for two n -bit numbers.

Before designing a binary adder, let us know some basic rules of binary addition. The most basic **binary addition** is addition of two single bit **binary numbers** that is addition of two binary digits.

The binary digits are 0 and 1. Hence, there must be four possible combinations of binary addition of two binary bits :

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 10$$

In the above list, first three binary operations result in one bit but fourth one result in two bits. In one bit binary addition, if augend and addend are 1, the sum will have two digits. The higher significant bit (HSB) or Left side bit is called carry and the list significant bit (LSB) or right side bit of the result is called sum bit. The logical circuit performs this one bit binary addition is called half adder.

Binary Adder

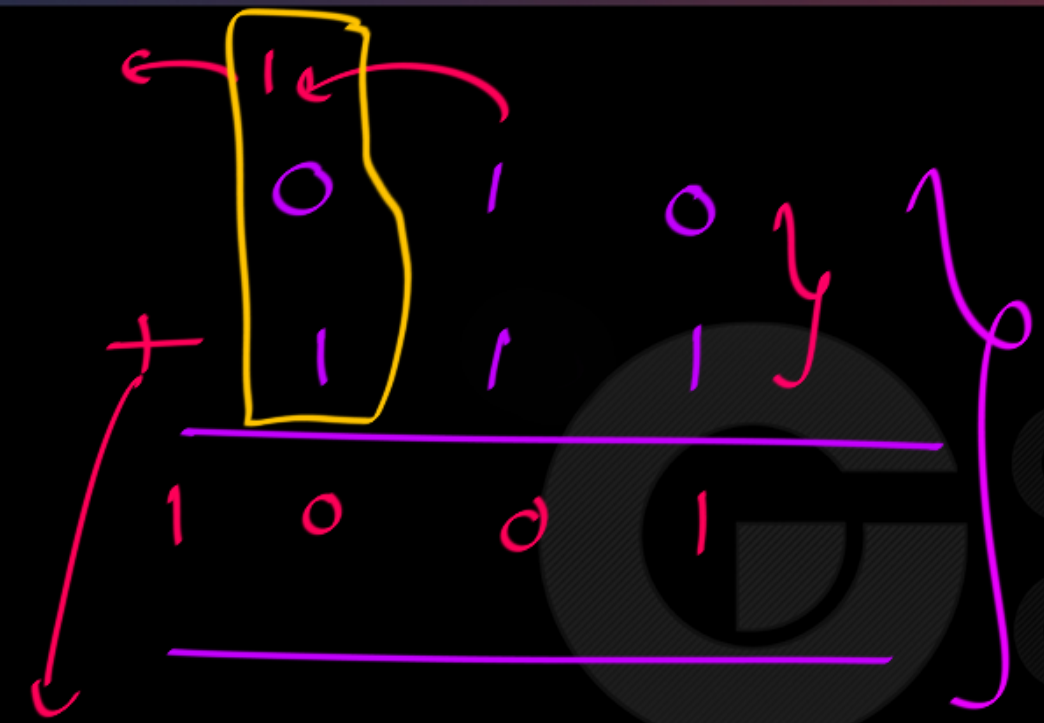
- Binary Addition
  - single bit addition

x	*y*	*x + y (binary sum)*
0	0	0
0	1	1
1	0	1
1	1	10 (binary, i.e. 2 in base-10)

Addition of two 1-bit numbers
(there are four possible cases)

x	0	0	1	1
$+\ y$	$+\ 0$	$+\ 1$	$+\ 0$	$+\ 1$
$c\ \ s$	0 0	0 1	0 1	1 0

Carry ← c ← s → Sum



Binary Addition

1+1=2=10

1+0+1=2=10

Binary Addition

||

Most fundamental op^n.

Addition of two bits.

Add two bits : Half Adder

opn : Circuit

Add three bits : full Adder

opn

Add two multi-bit binary numbers : Ripple carry Adder, CLA

1. Half Adder

Half Adder: is a combinational circuit that performs the addition of two bits, this circuit needs two binary inputs and two binary outputs.

Inputs		Outputs	
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Truth table

The simplified Boolean function from the truth table:

$$\left.\begin{cases} S = \bar{X}Y + X\bar{Y} \\ C = XY \end{cases}\right\} \quad 1$$ (Using sum of product form)

Where S is the sum and C is the carry.

$$\left.\begin{cases} S = X \oplus Y \\ C = XY \end{cases}\right\} \quad 2$$ (Using XOR and AND Gates)

Half Adder : ⇒ Add two bits.

Binary Add	Decimal sum	Binary sum
0 + 0 =	0	= 0 0
0 + 1 =	1	= 0 1
1 + 0 =	1	= 0 1
1 + 1 =	2	= 1 0

Let x, y be two binary bits : $x \to 0/1$

$y \to 0/1$

$$\begin{array}{r} x \\ +\ y \\ \hline C\ \ S \end{array}$$

C → Carry bit, S → Sum bit

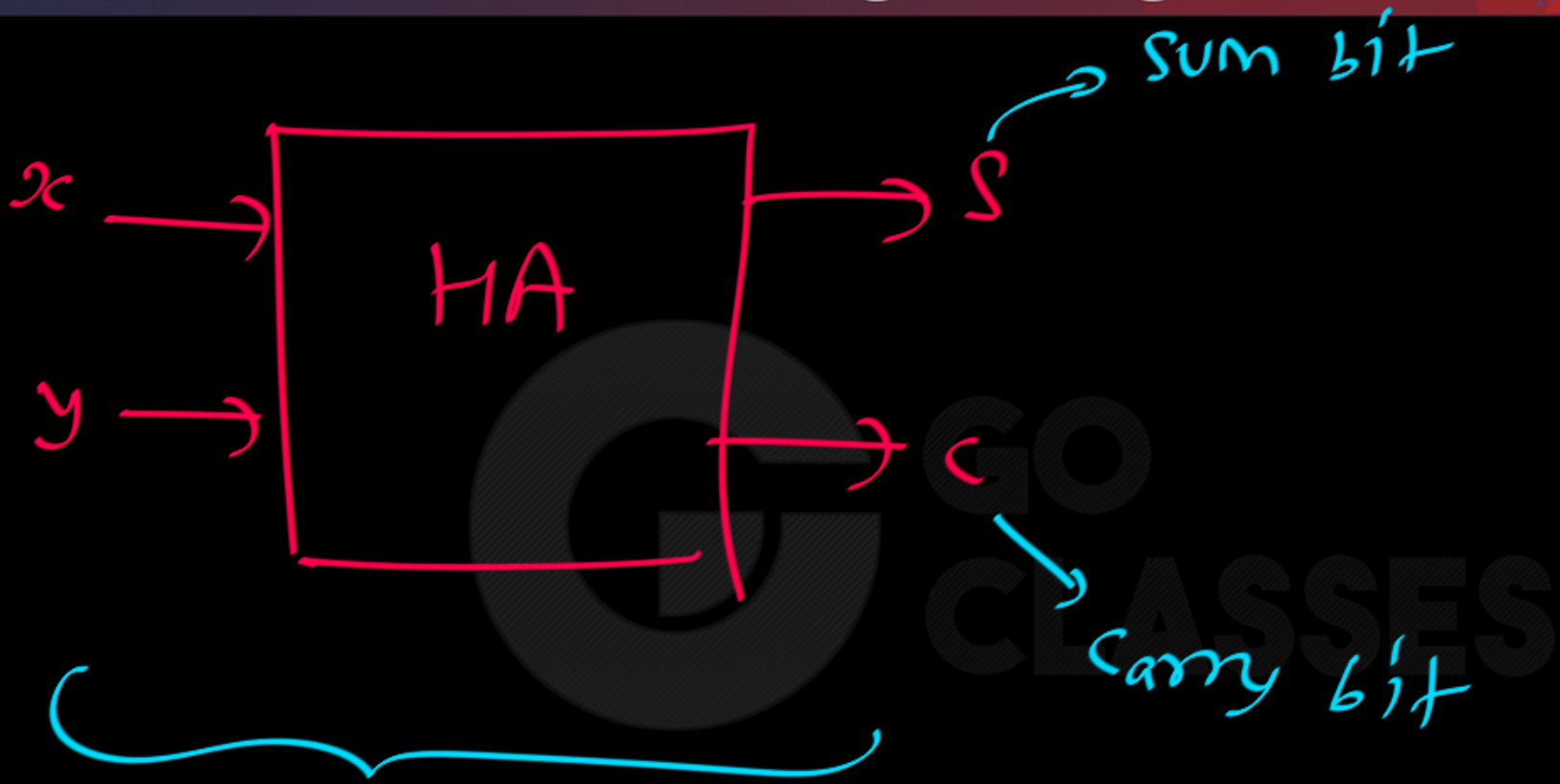
$$\begin{array}{r} 0 \\ +\ 0 \\ \hline 0\ 0 \end{array}$$

$$\begin{array}{r} 0 \\ +\ 1 \\ \hline 0\ 1 \end{array}$$

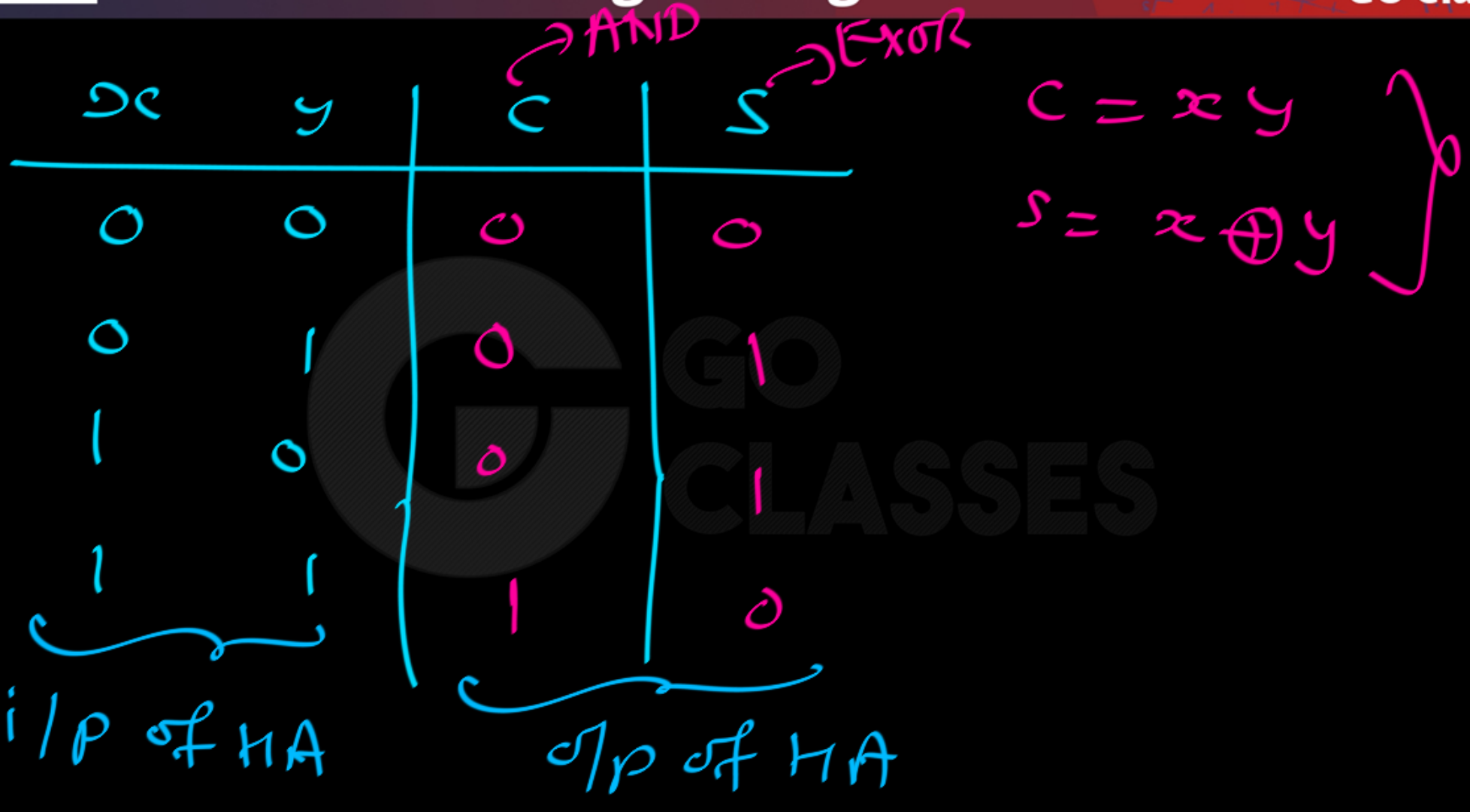
$$\begin{array}{r} 1 \\ +\ 0 \\ \hline 0\ 1 \end{array}$$

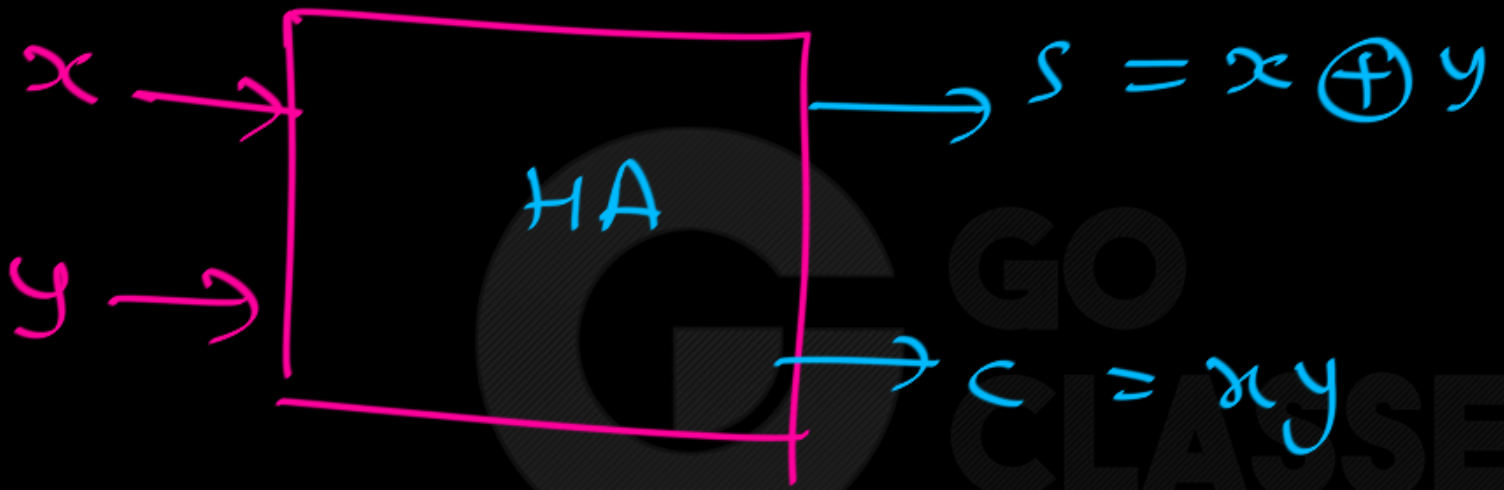
$$\begin{array}{r} 1 \\ +\ 1 \\ \hline 1\ 0 \end{array}$$

Carry bit, Sum bit



Block Diagram of HA.




x
y
HA
S = x ⊕ y
C = xy

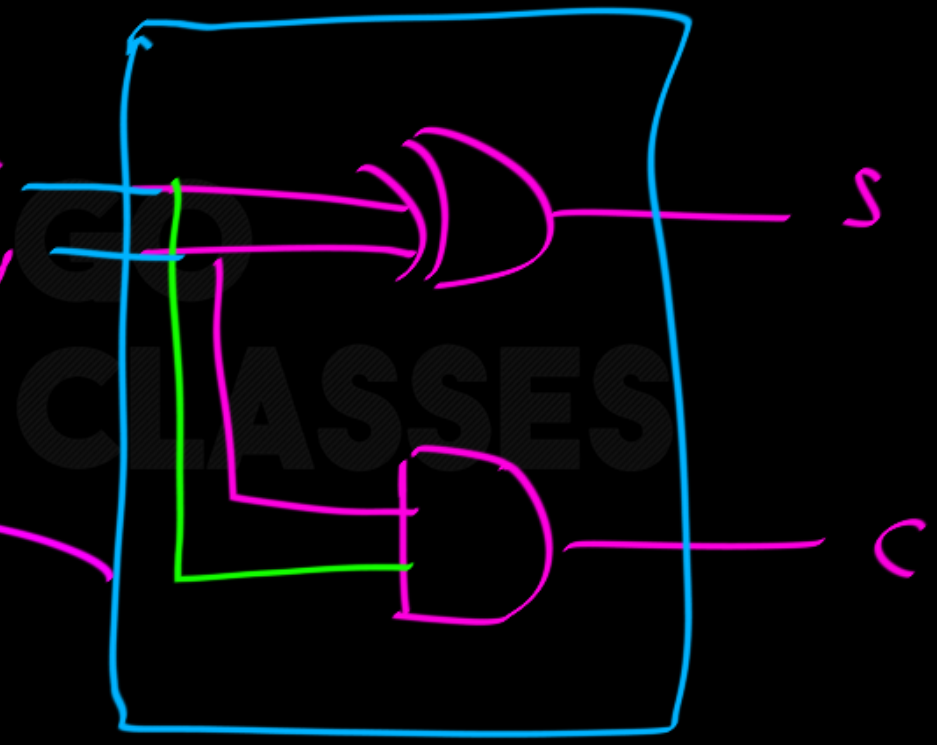
Implementation of HA using logic GATES:

$S = x \oplus y$

$C = xy$

- AND,
- EX OR

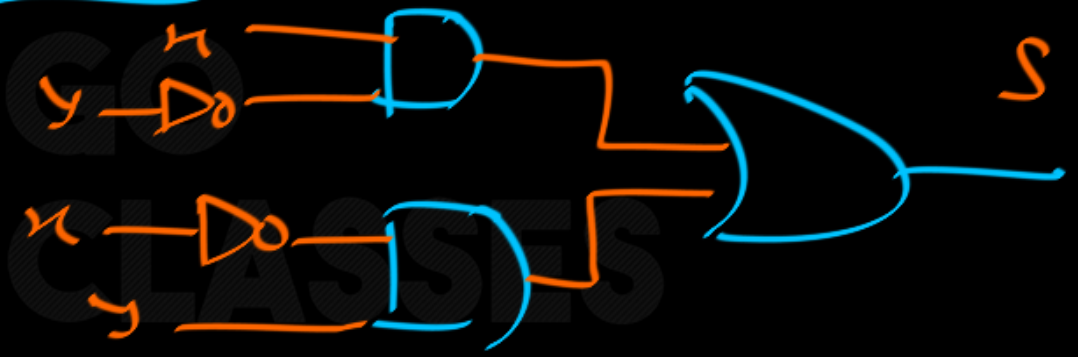
implementation of HA

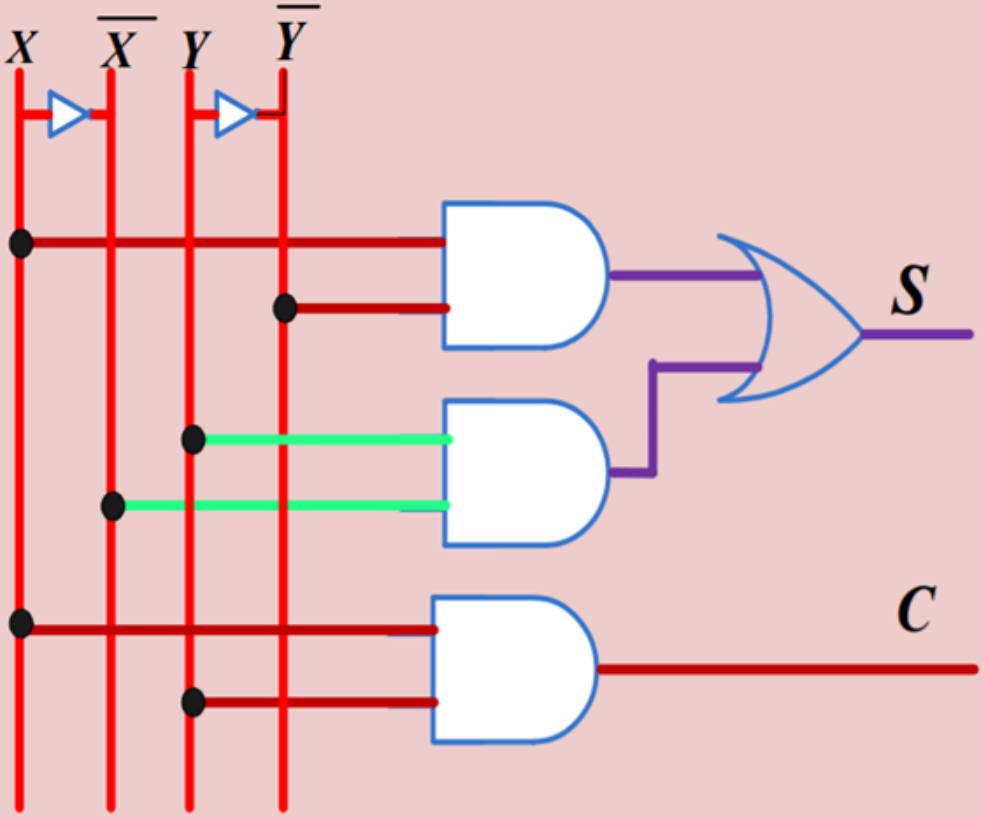


two level AND – OR implementation of HA;

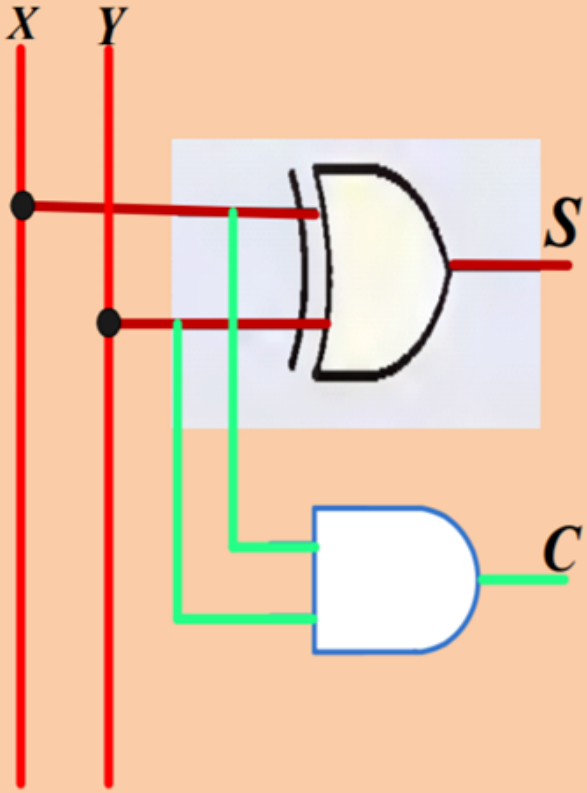
$S = x \oplus y = x\bar{y} + \bar{x}y$

$C = xy$



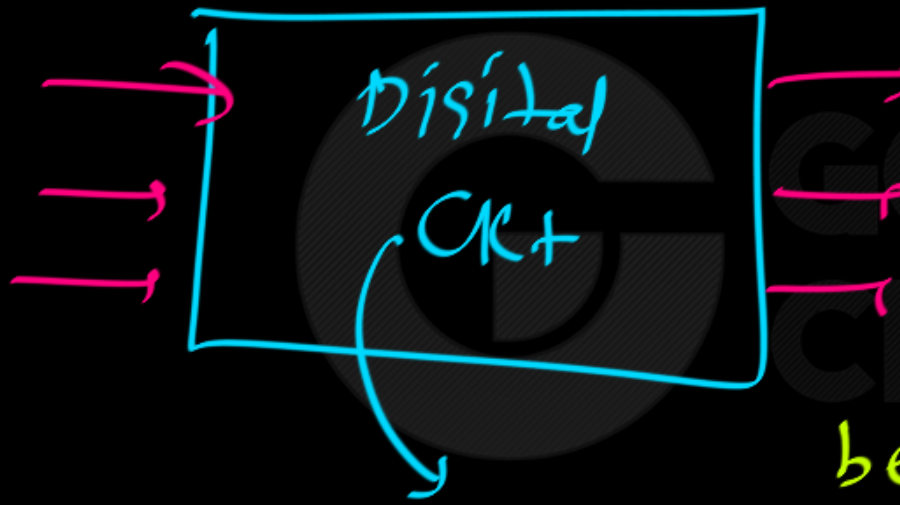


Implementation of Half Adder using equation (1)

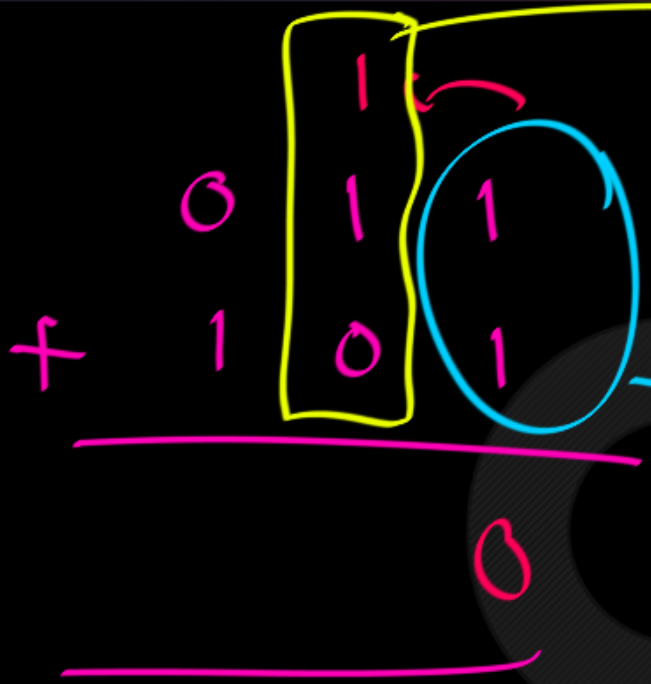


Implementation of Half Adder using equation (2)

Propagation delay : Time taken by a Dig. ckt to respond / react to the input changes / input Applied before producing correct o/p.



for eg: logic gate : NOT gate, OR gate etc



Can not Apply HA.

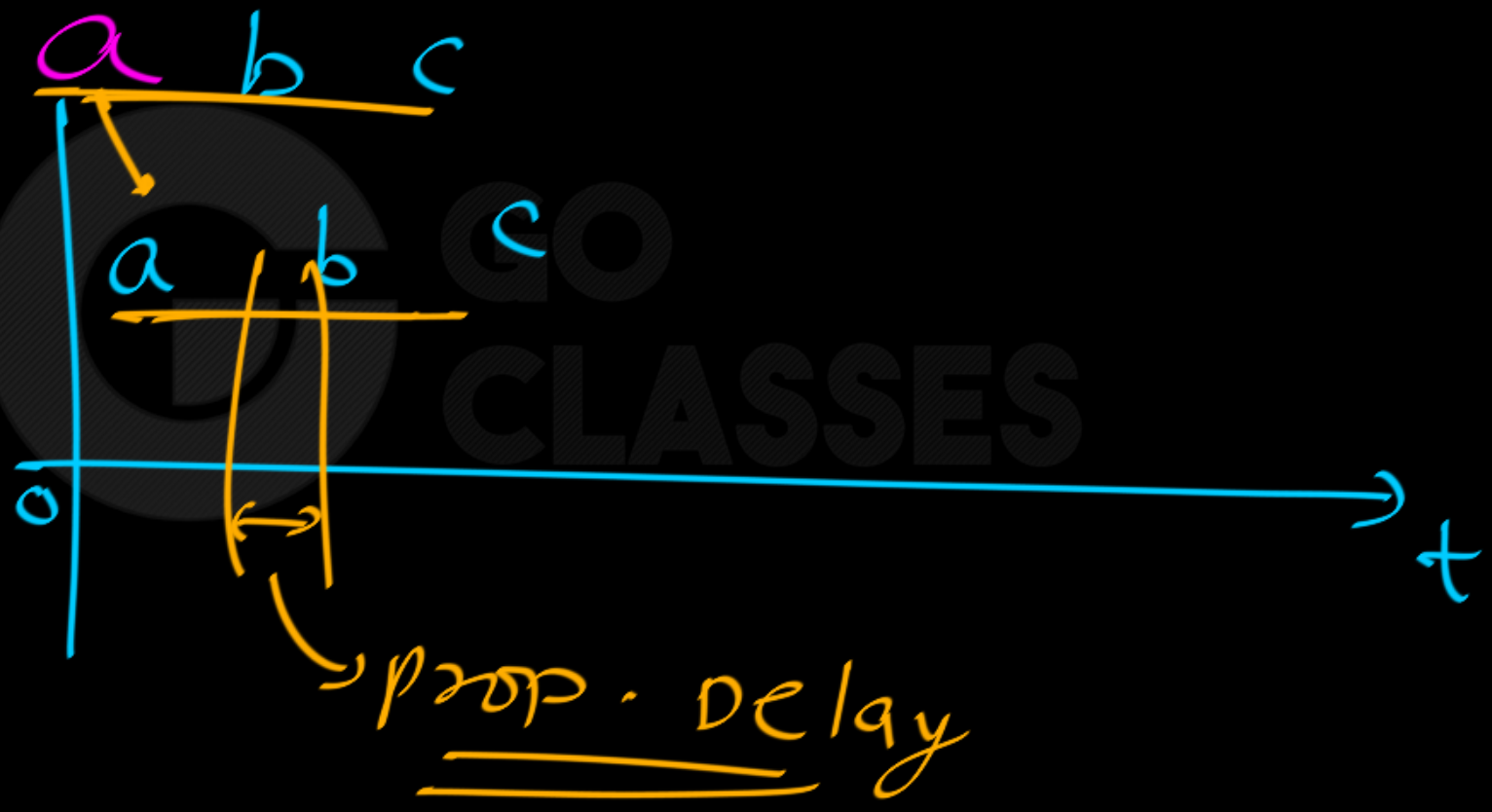
LSB position ⇒ needs to Add two bits.

Can Apply HA.

Ex: typing on keyboard:

You typing: a b c

Screen:



Q 1 :

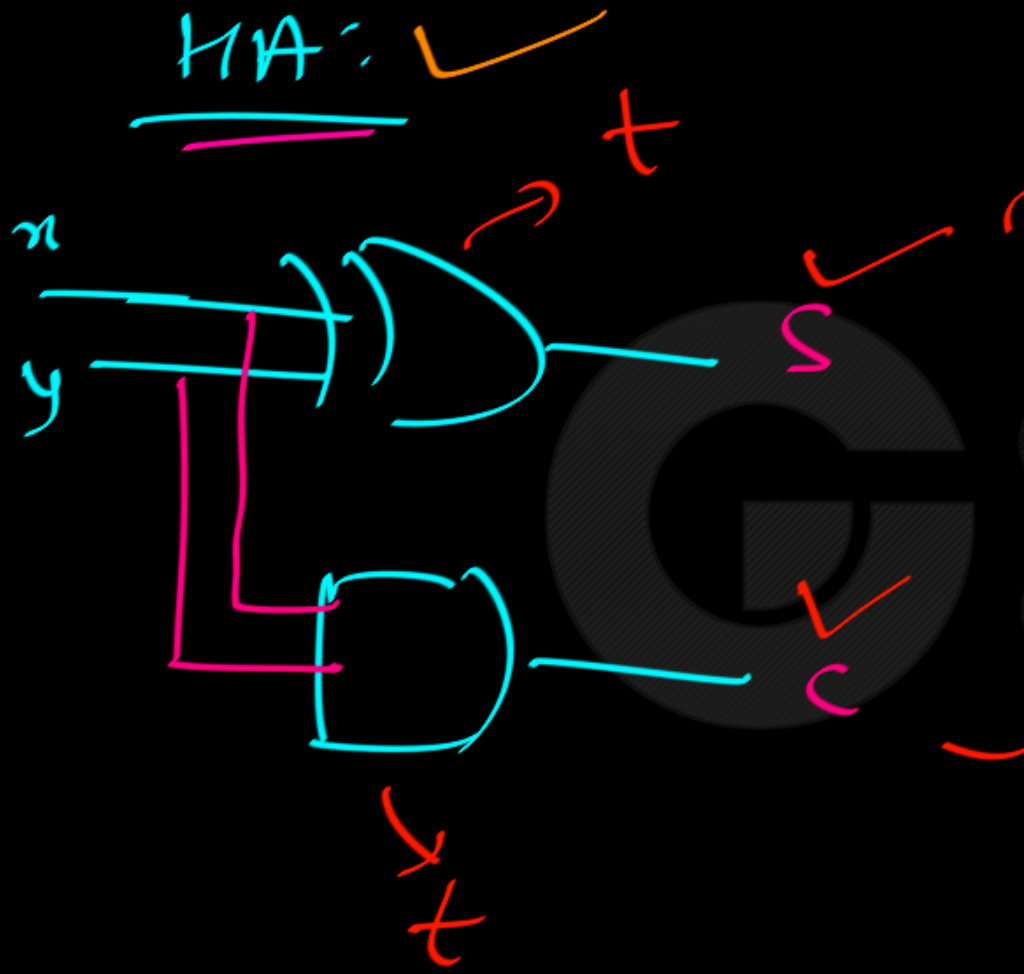
If we say that "t" is the delay of a AND, OR, Exor, NOT gate etc, then what is the delay of Half Adder ?

Q 1 :

If we say that "t" is the delay of a AND, OR, Exor, NOT gate etc, then what is the delay of Half Adder ?

Ans : t

HA :



Delay for S ⇒ t

Delay for C ⇒ t

Delay for HA : ⇒ (critical path)

max (Delay of S, Delay of C)

= t

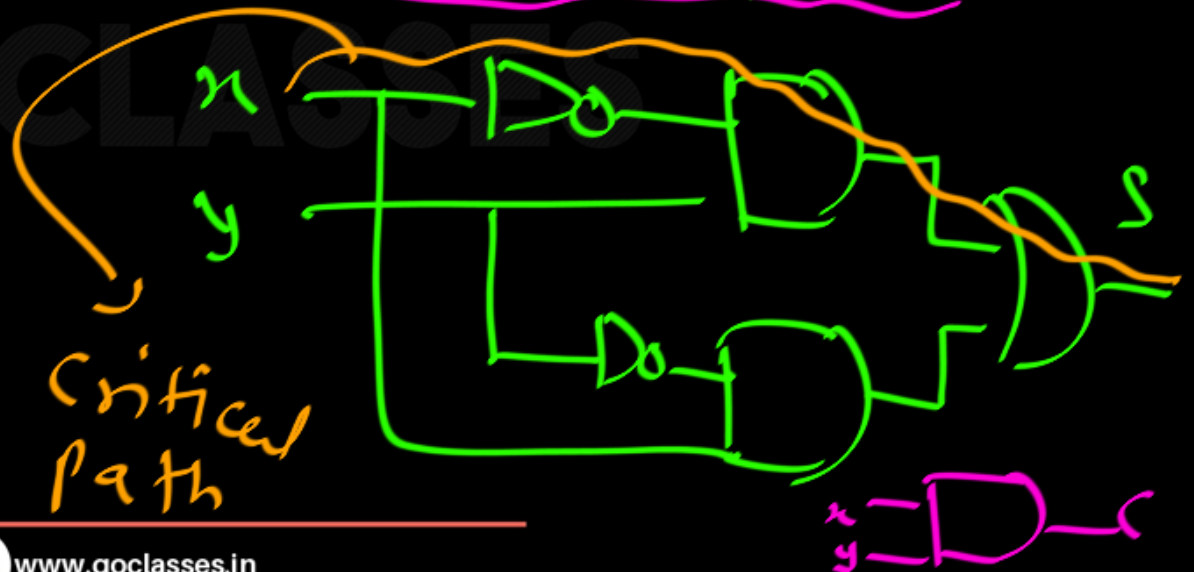
Mistake: for prev. Ques., if we take AND–OR implementation of HA:

then critical path is generation of S.

Delay of C: t

Delay of S: 3t

Delay of HA: 3t



Delay of any Circuit:

Longest possible Delay in a Circuit.

↓

Critical Path

Q 2 :

Exclusive-OR is sometimes not regarded as a basic gate.

Assume that AND, OR, NOT gate are basic gates.

If we say that "t" is the delay of a basic gate and we can only use basic gates in any circuit implementation, then what is the delay of Half Adder ?

Q 2 :

Exclusive-OR is sometimes not regarded as a basic gate.

Assume that AND, OR, NOT gate are basic gates.

If we say that "t" is the delay of a basic gate and we can only use basic gates in any circuit implementation, then what is the delay of Half Adder ?

Ans: 3t

→ Cannot use EXOR-AND implementation of HA.

Delay of S = 3t

Delay of C = t

Delay of HA = 3t

Q 3 :

Exclusive-OR is sometimes not regarded as a basic gate.

Assume that AND, OR, NOT gate are basic gates. Assume inputs are available in original and complementary form.

If we say that "t" is the delay of a basic gate and we can only use basic gates in any circuit implementation, then what is the delay of Half Adder ?

Q 3 :

Exclusive-OR is sometimes not regarded as a basic gate.

Assume that AND, OR, NOT gate are basic gates. Assume inputs are available in original and complementary form.

If we say that "t" is the delay of a basic gate and we can only use basic gates in any circuit implementation, then what is the delay of Half Adder ?

Ans : 2t

Delay of S : 2t

" " C = t

Delay of HA = 2t