Next Topic:

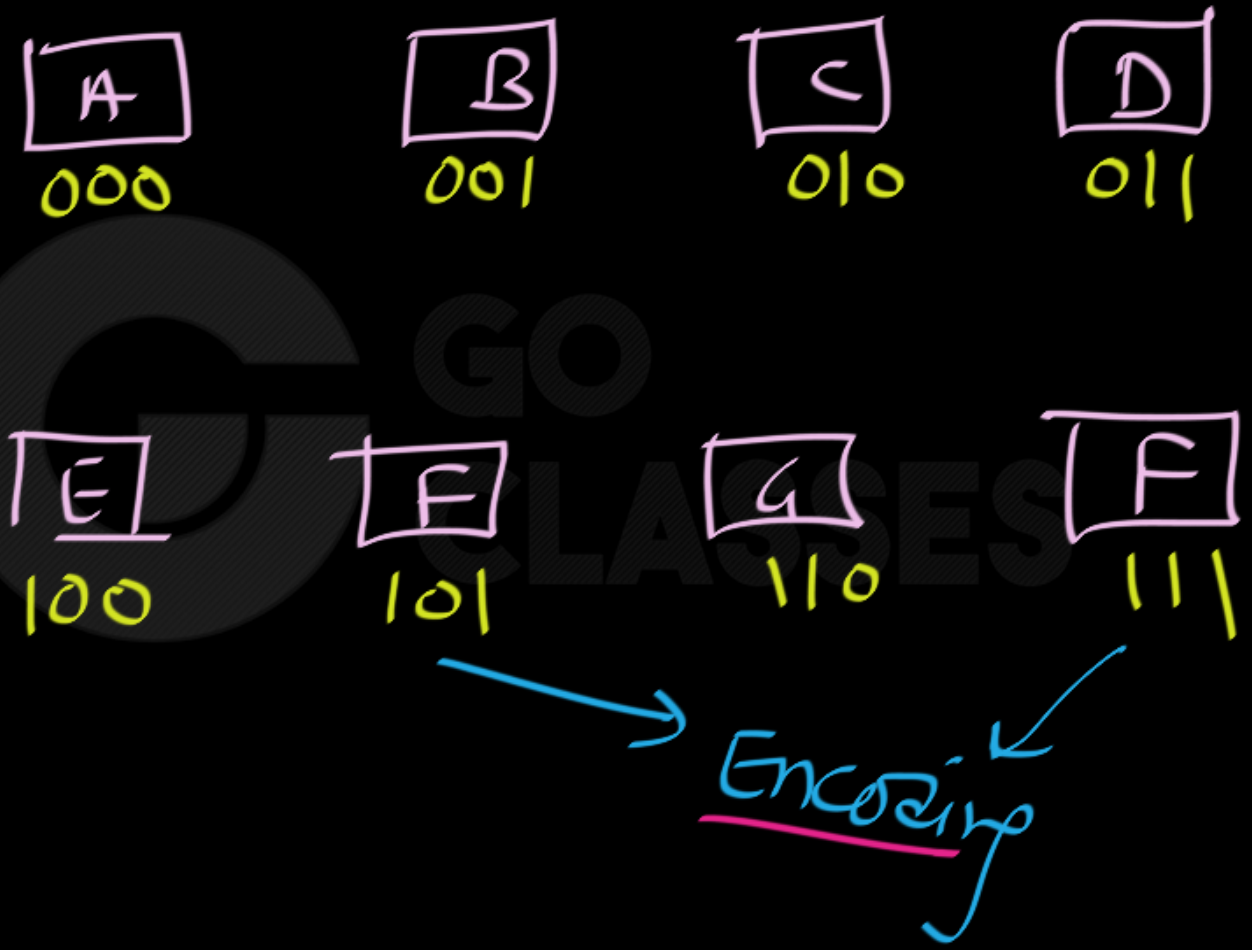
Decoder

Houses:

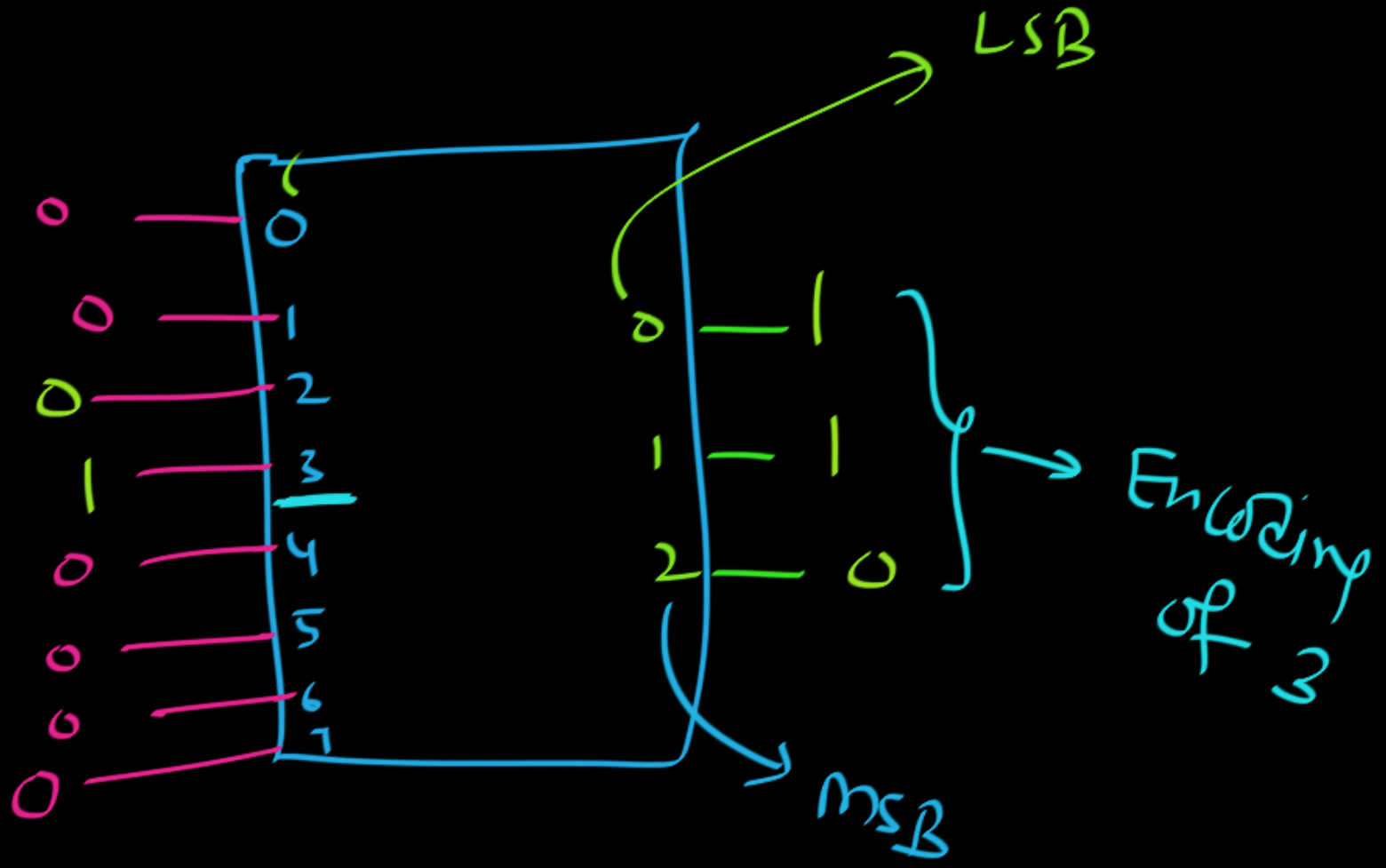
memory locations

8 bytes

Each byte has Address



Encoding



Why to Encode ?

To Decode when need to access

Ex:

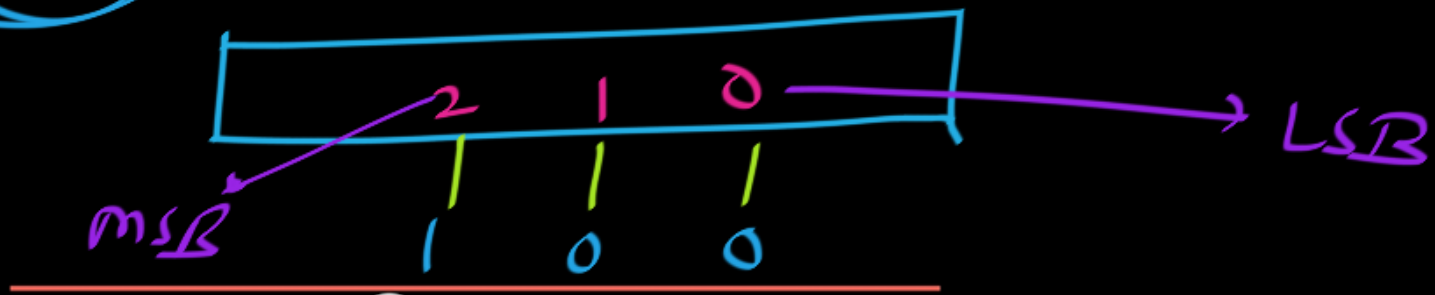
E

Houses:

memory locations

8 bytes

Each byte has Address

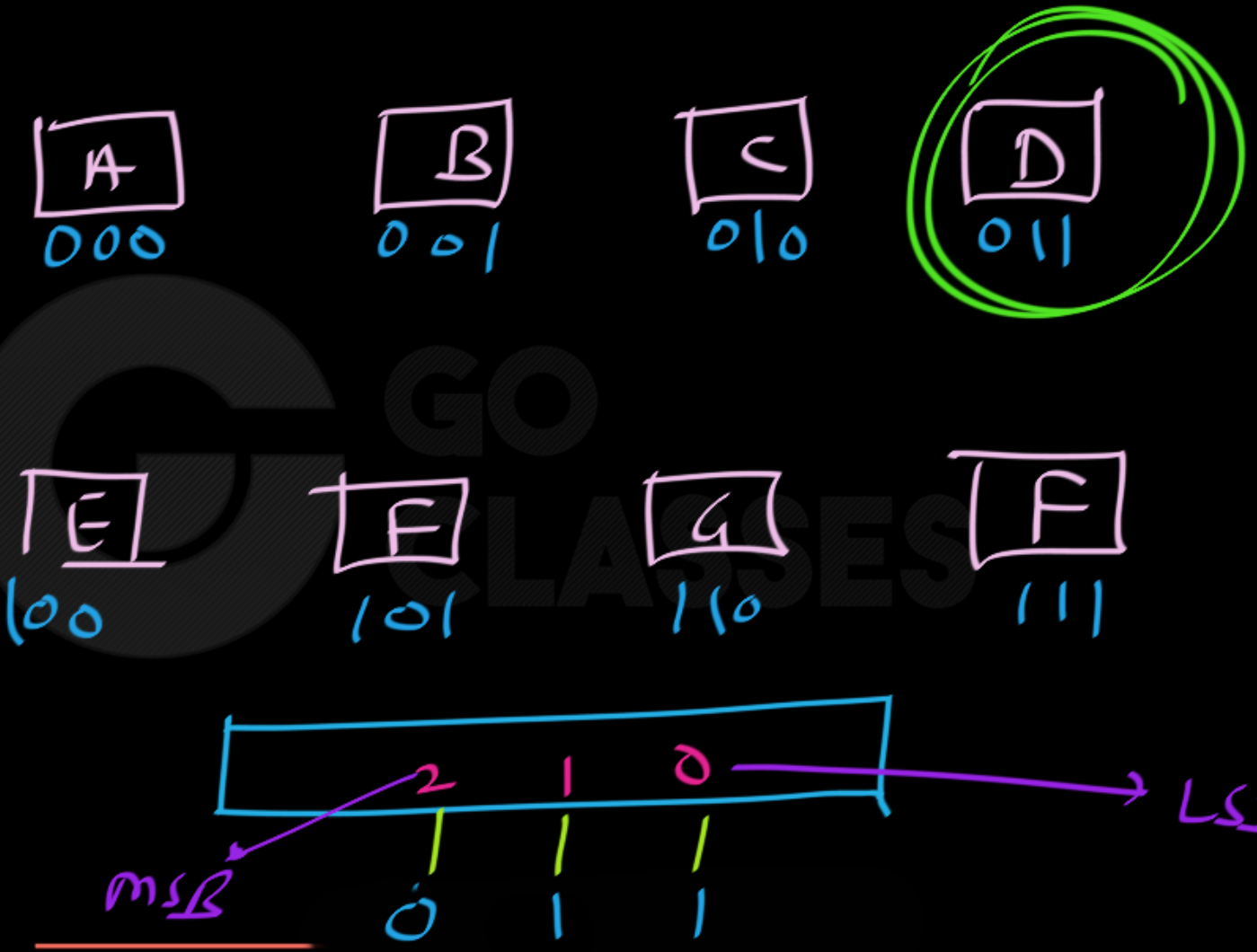


Houses:

memory locations

8 bytes

Each byte has Address



Decoder : opposite of Encoder

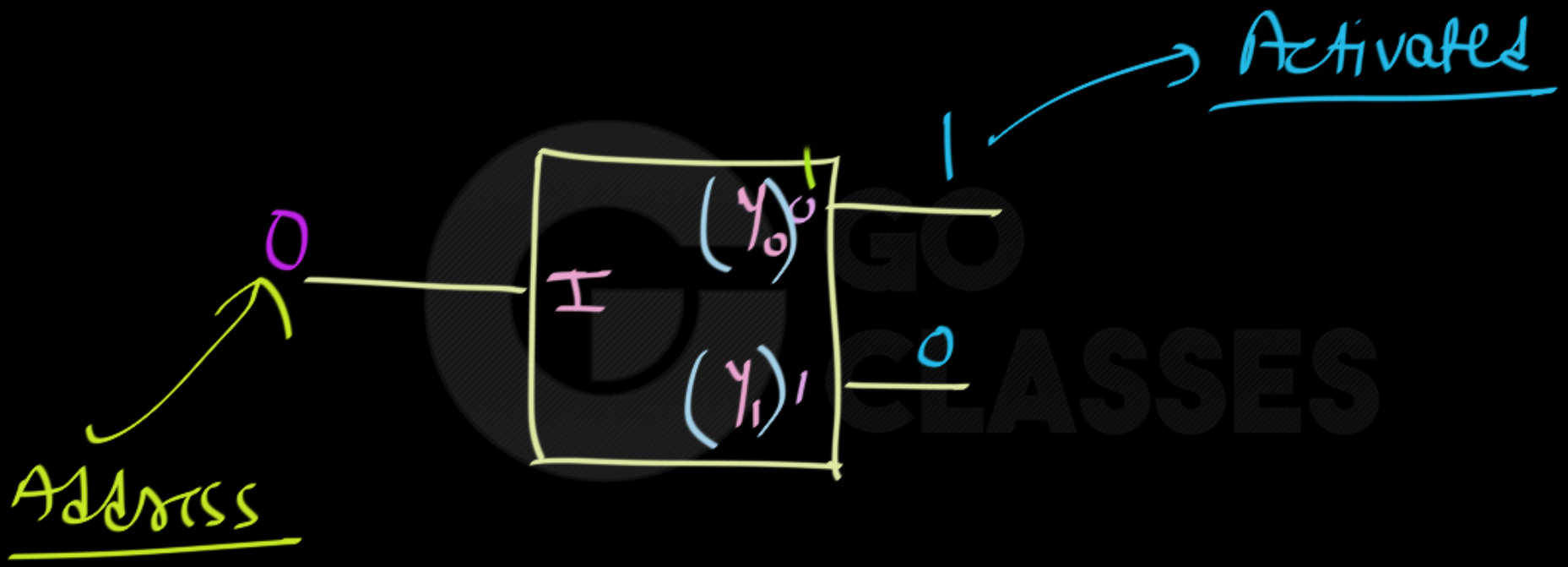
Decoder → n to 2^n Decoder

Using code, Access the location (Activate corresponding location/output)

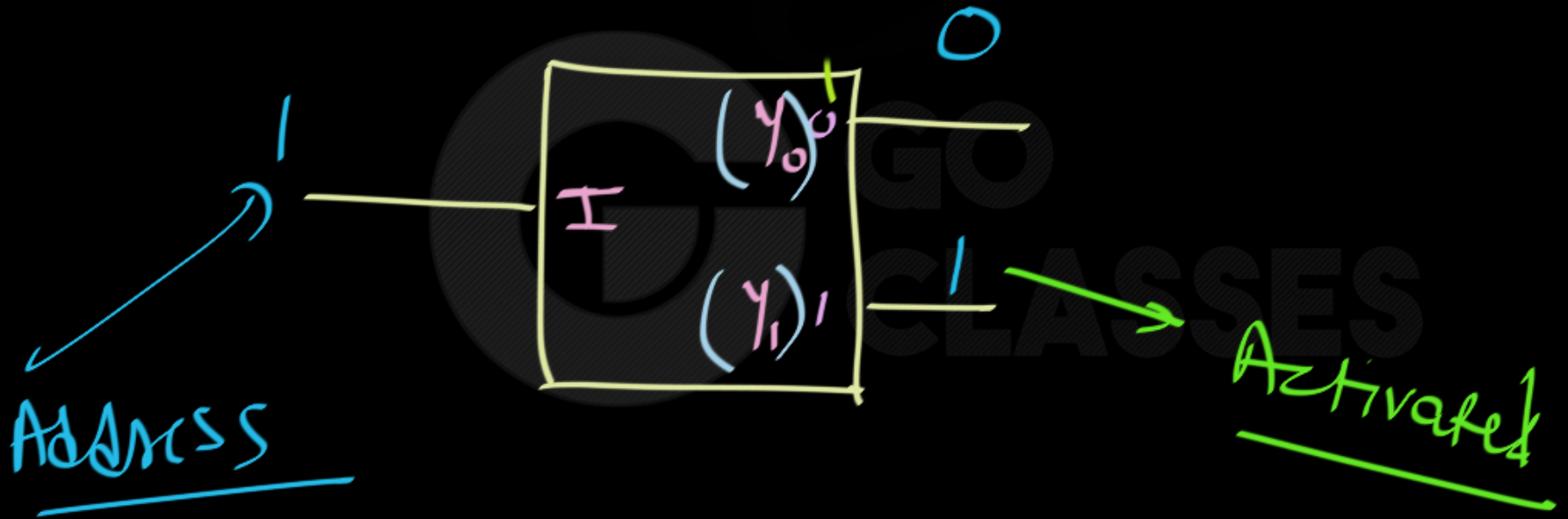
Encoder → Give code

2^n to n Encoder

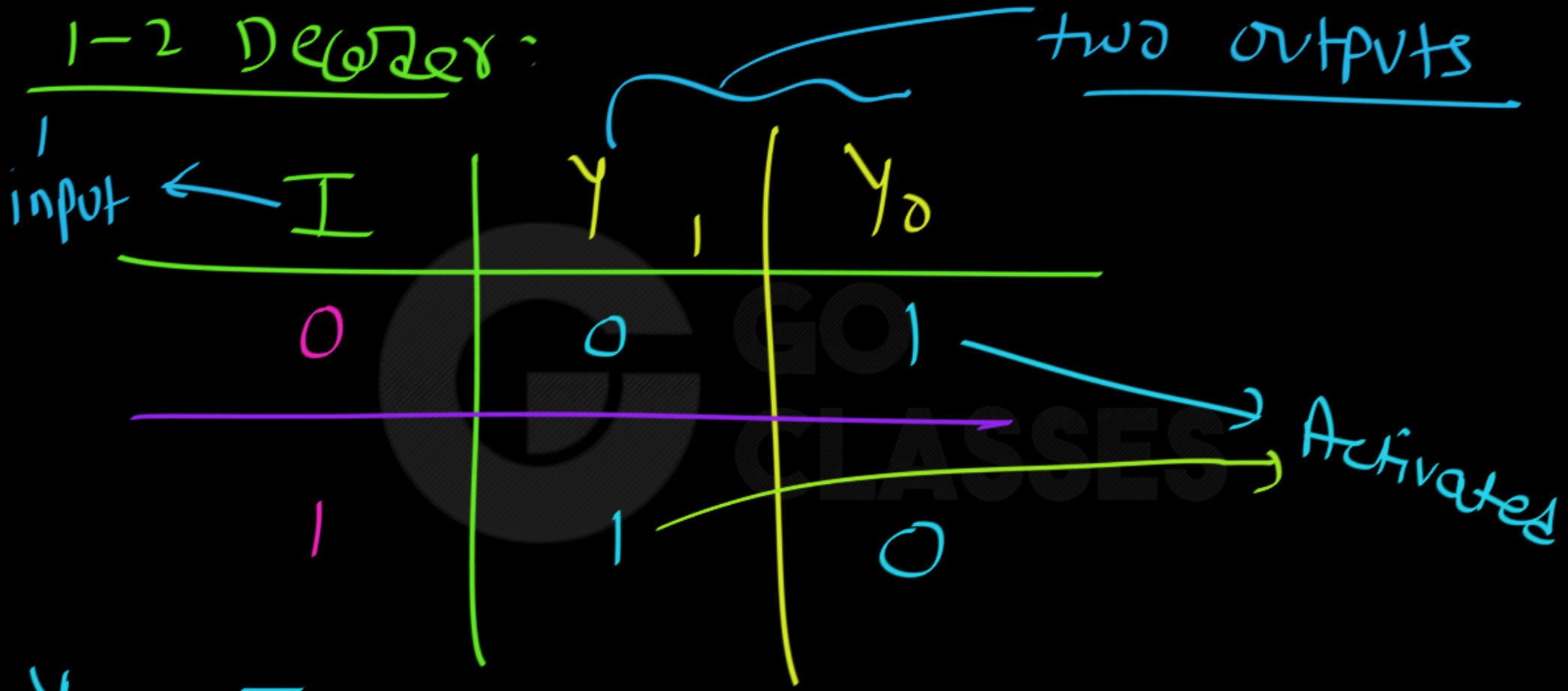
1-2 Decoder:



1-2 Decoder:



1-2 Decoder:



$Y_1 = I$; $Y_0 = \overline{I}$

Implementation:

I	Y_0	Y_1
0	1	0
1	0	1

Decoder (1*2)

2 x 4 Decoder:

Inputs

Outputs

(I_1, I_0)

MSB

LSB

Behavior:

I_1	I_0	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

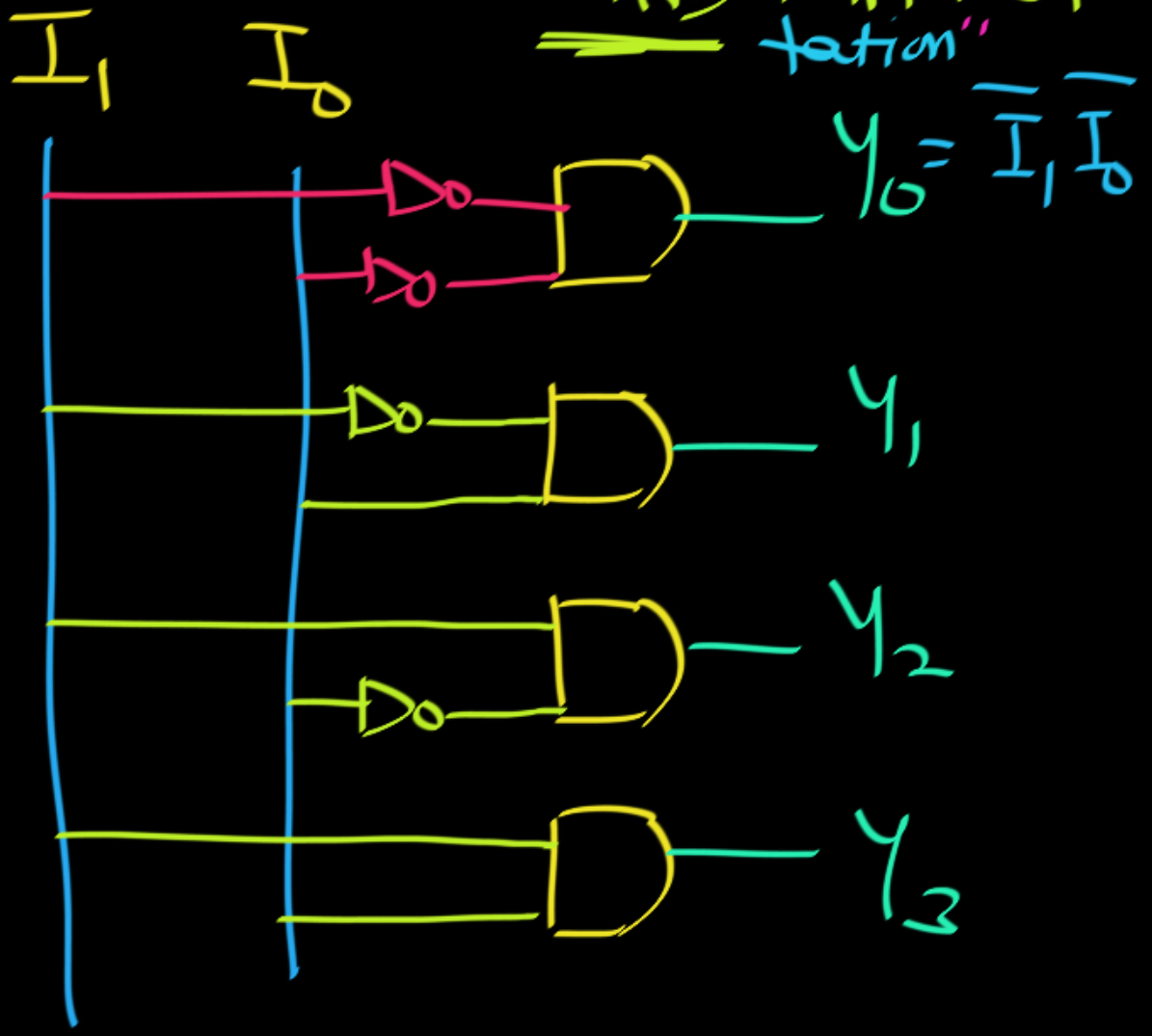
$Y_0 = \overline{I_1}\,\overline{I_0}$

$Y_1 = \overline{I_1}\,I_0$

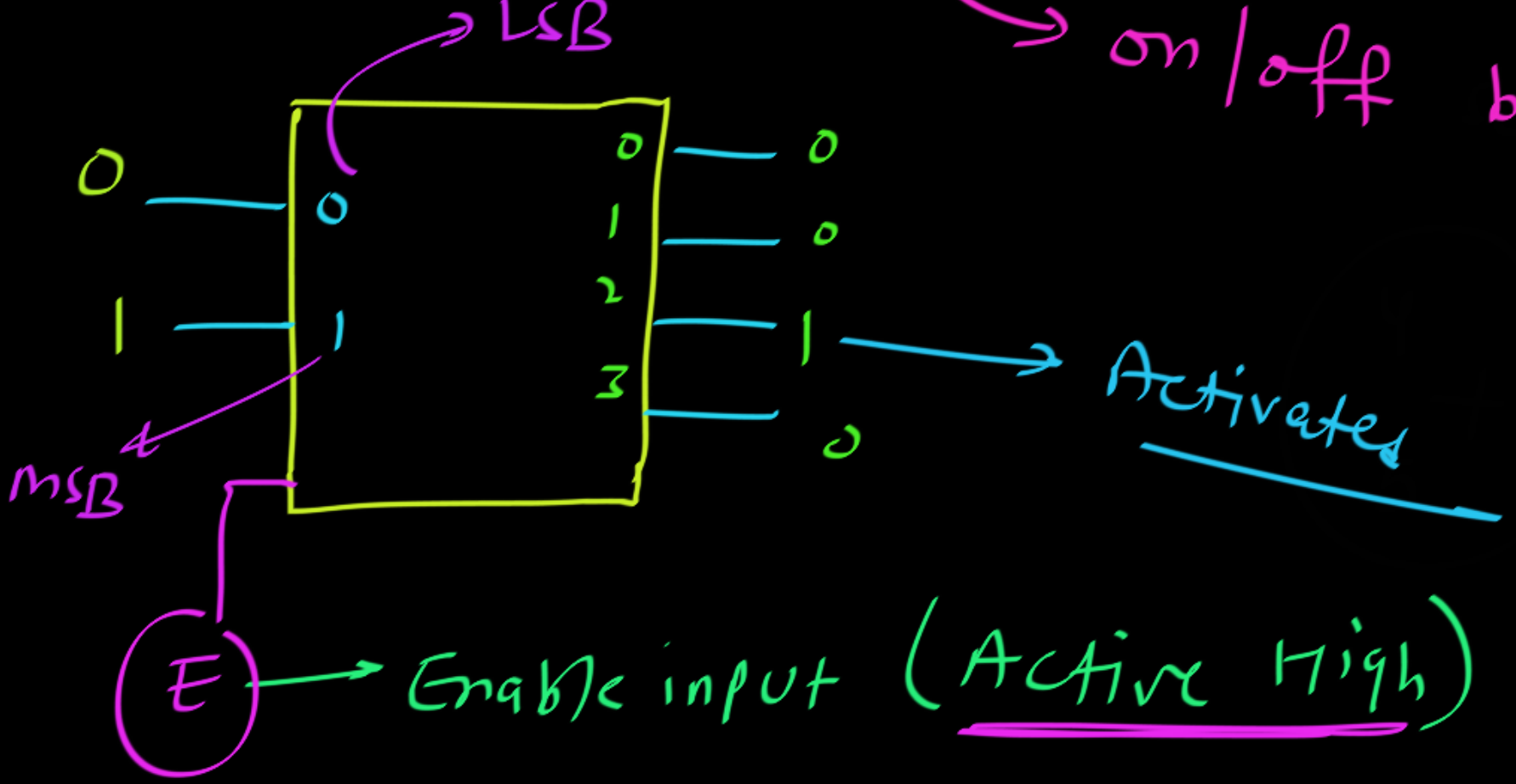
$Y_2 = I_1\,\overline{I_0}$

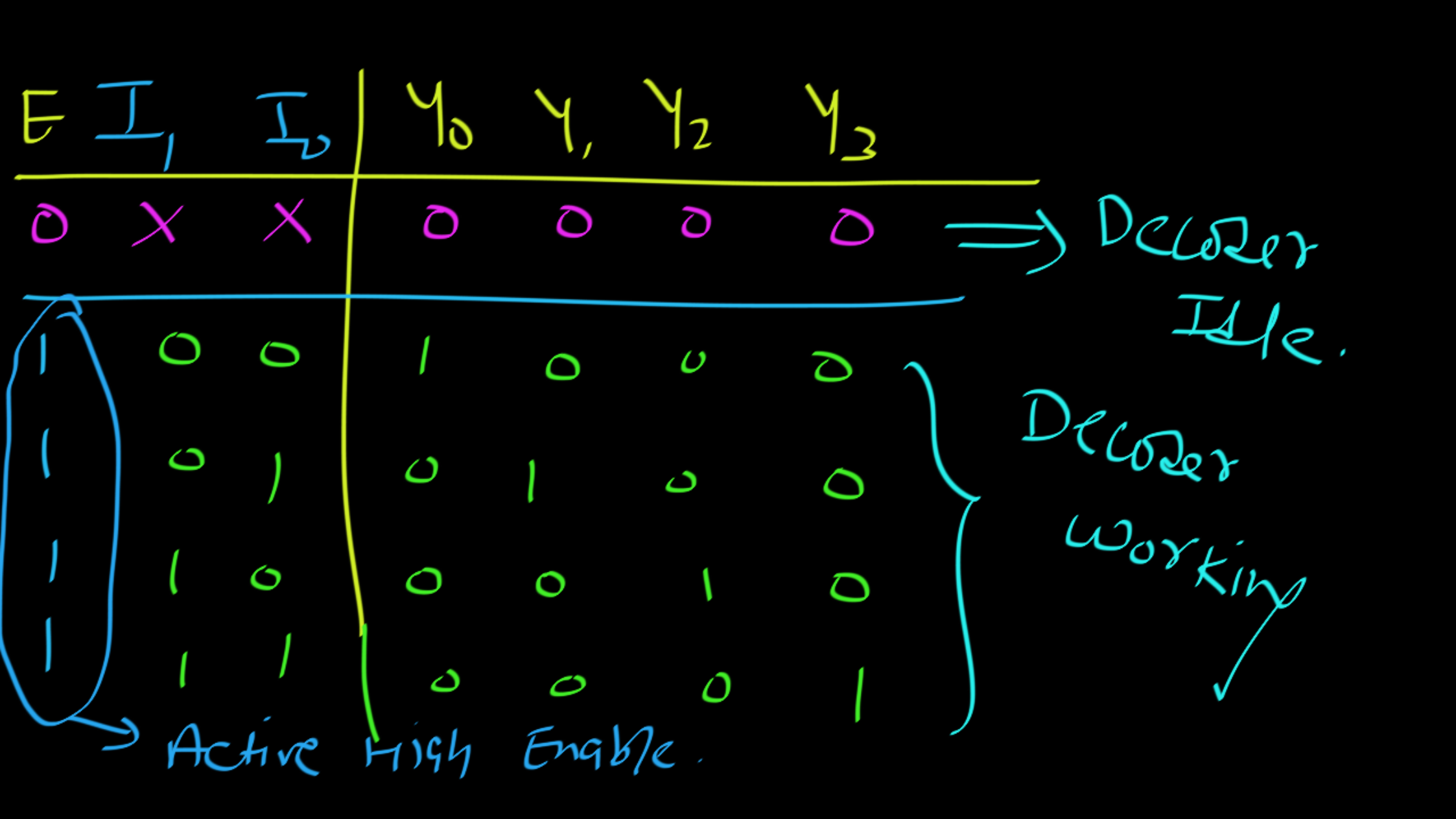
$Y_3 = I_1\,I_0$

"AND implementation"

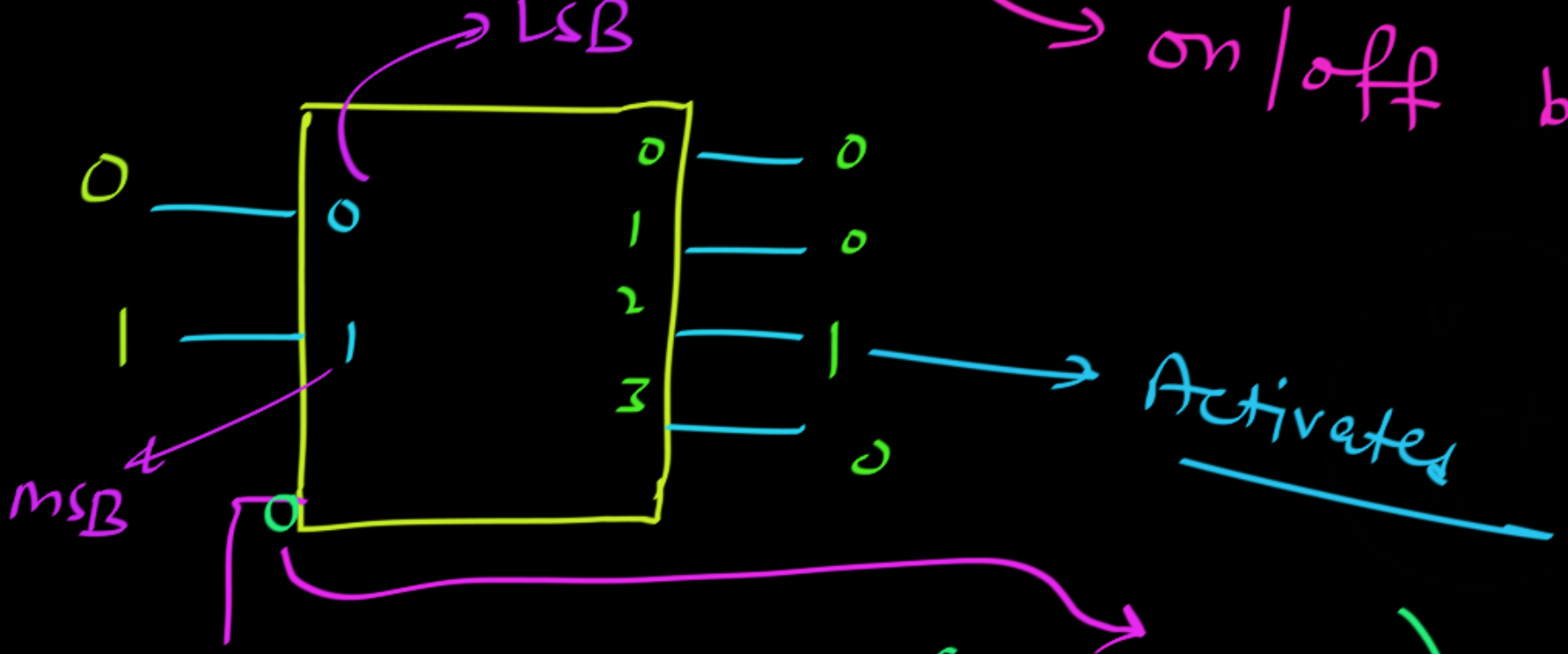


Decoder with Enable input :



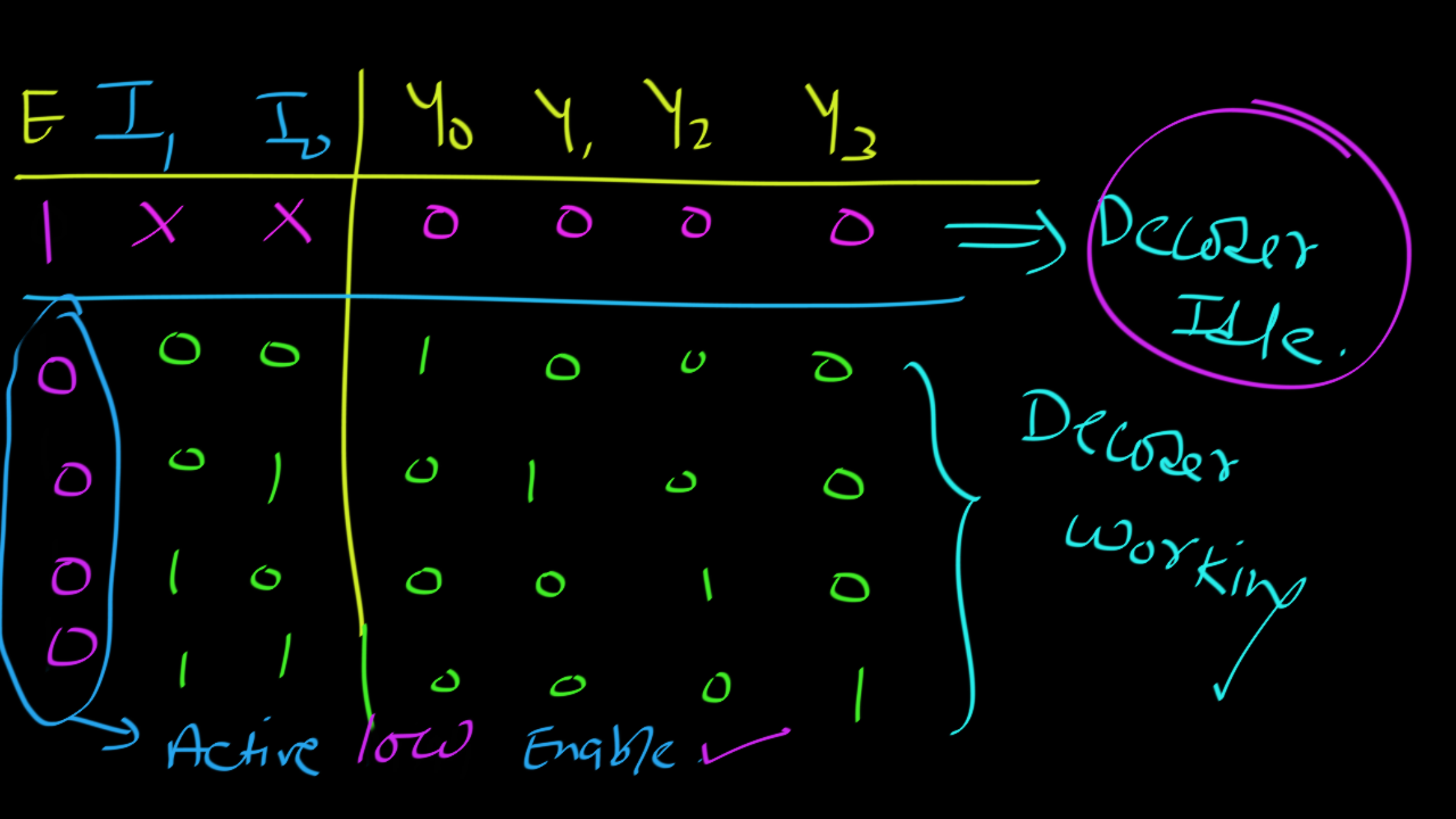

E I1 I0 Y0 Y1 Y2 Y3
0 X X 0 0 0 0
Decoder Idle.
1 0 0 1 0 0 0
1 0 1 0 1 0 0
1 1 0 0 0 1 0
1 1 1 0 0 0 1
Decoder working
Active High Enable.

Decoder with Enable input :

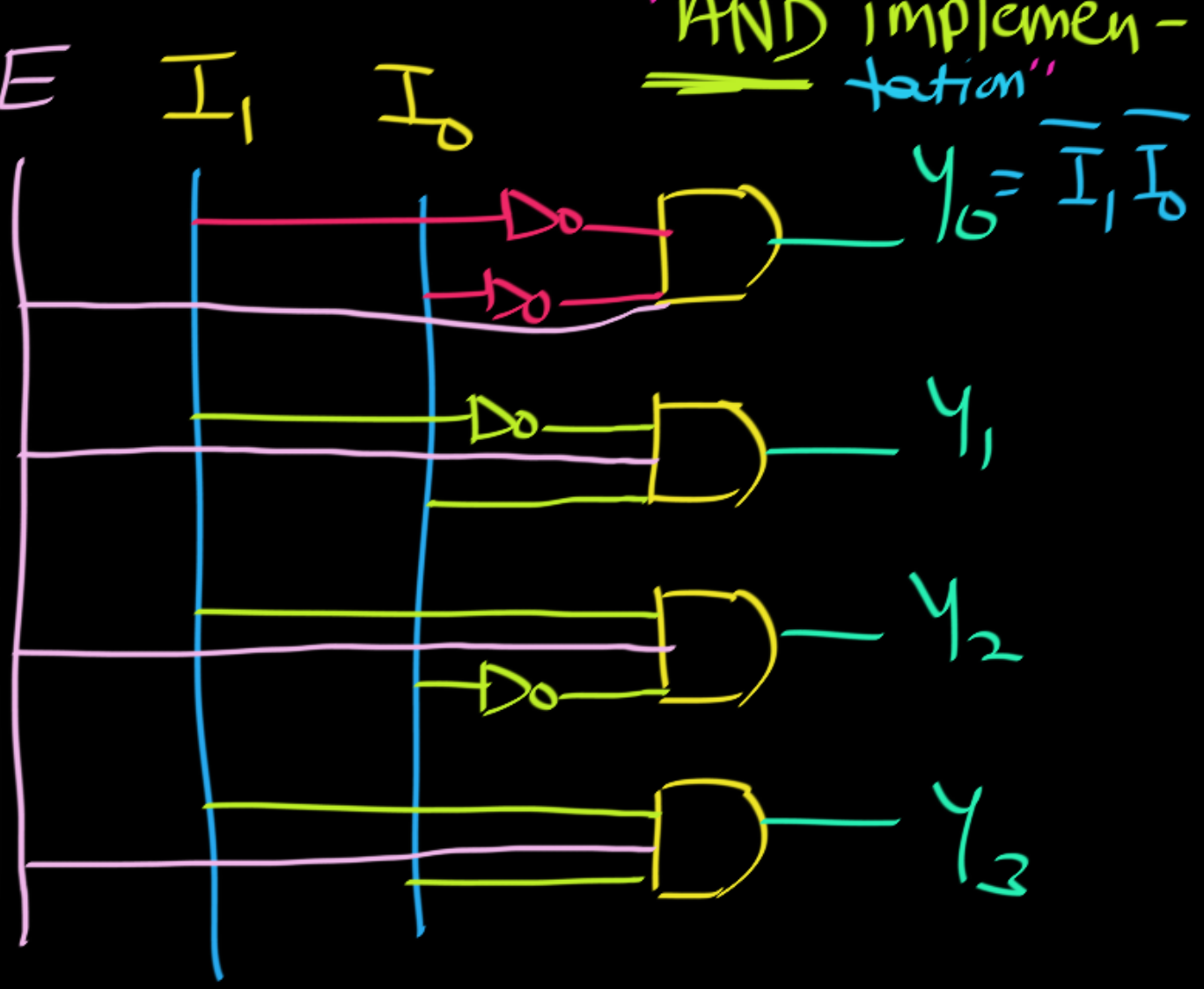


on/off button

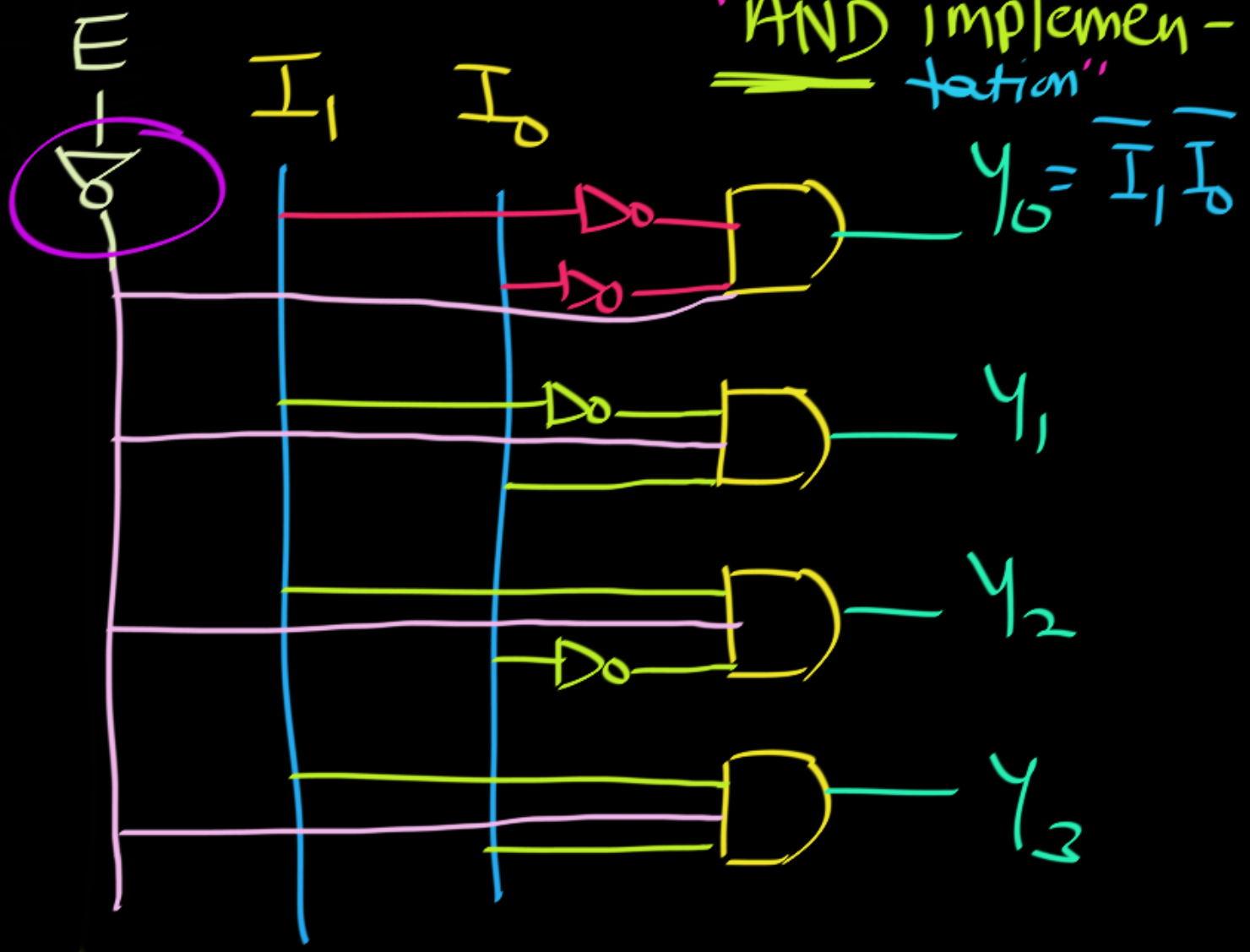
E → Enable input (Active low)


E I1 I0 Y0 Y1 Y2 Y3
1 X X 0 0 0 0
Decoder Idle.
0 0 0 1 0 0 0
0 0 1 0 1 0 0
0 1 0 0 0 1 0
0 1 1 0 0 0 1
Decoder working
Active low Enable

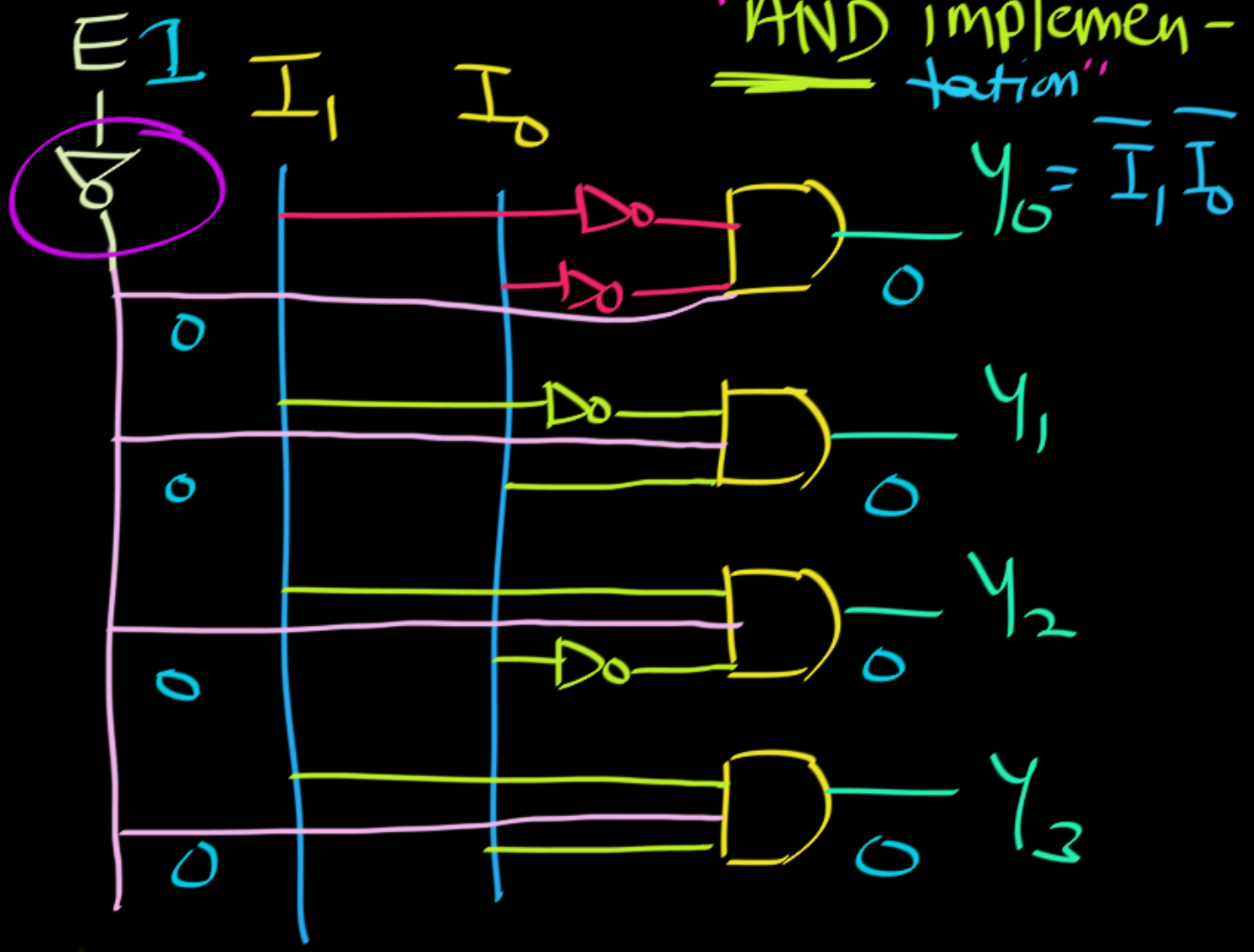
Decoder with Active High Enable E :

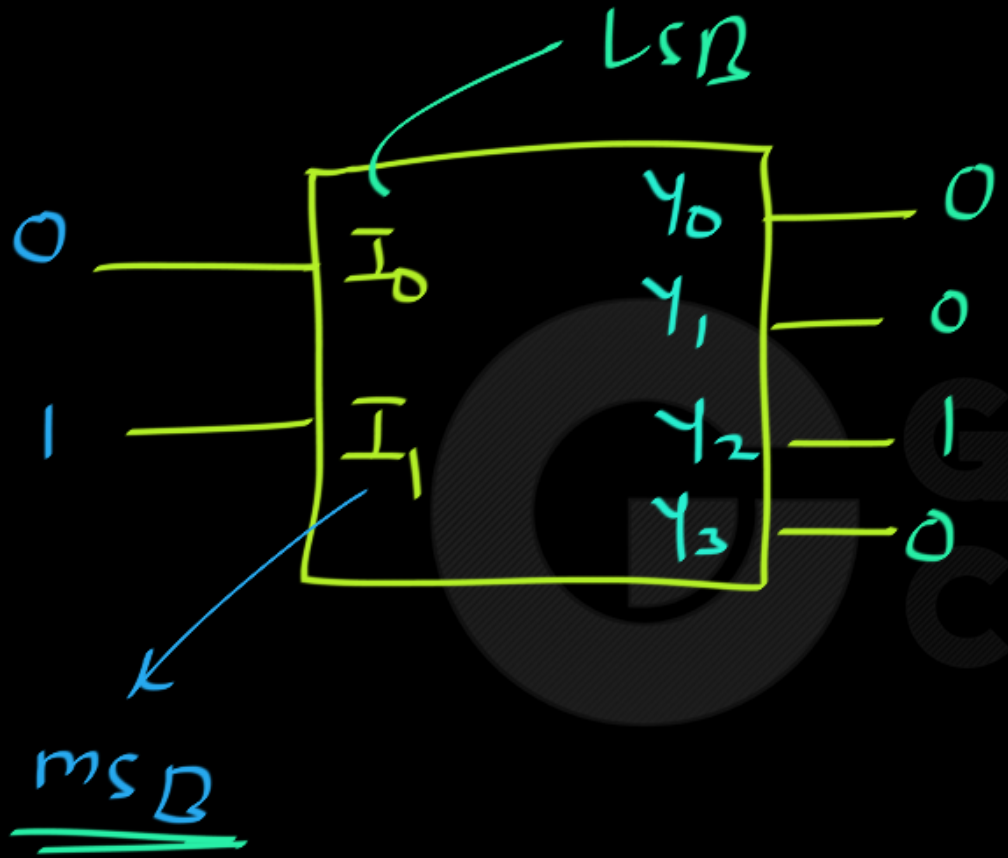


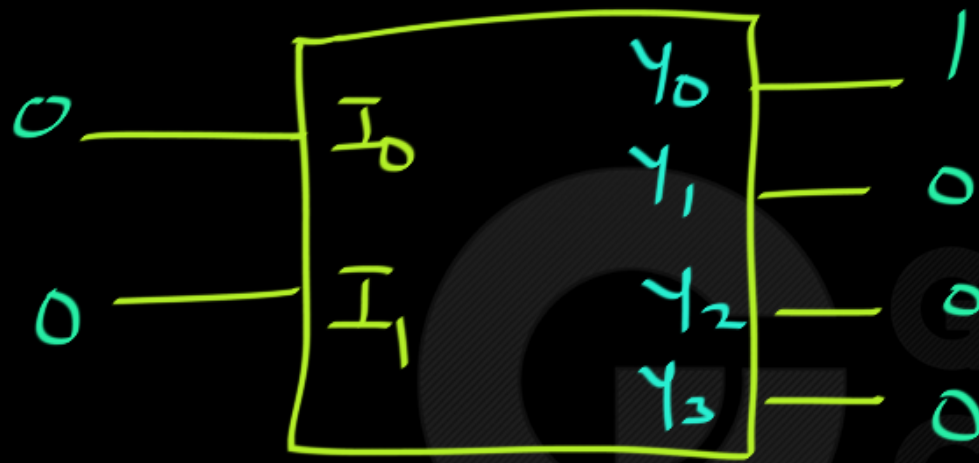
Decoder with Active Low Enable E :

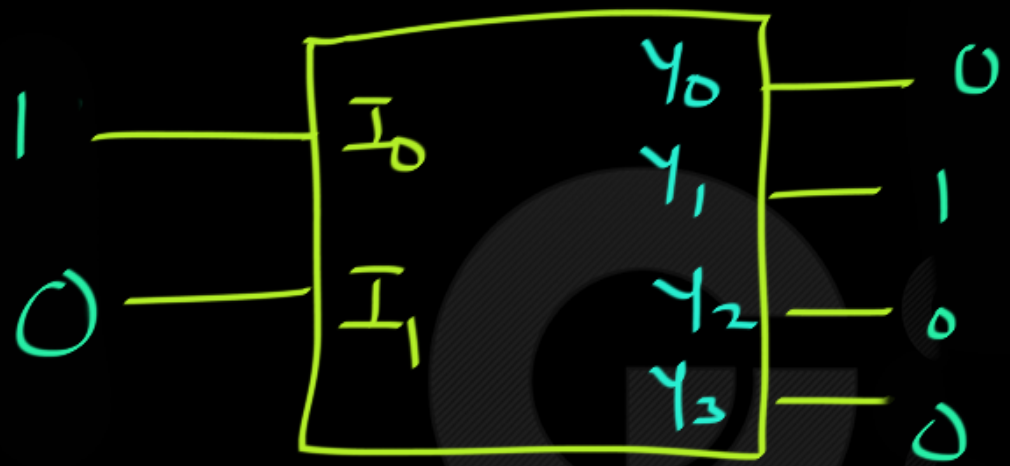


Decoder with Active Low Enable E :

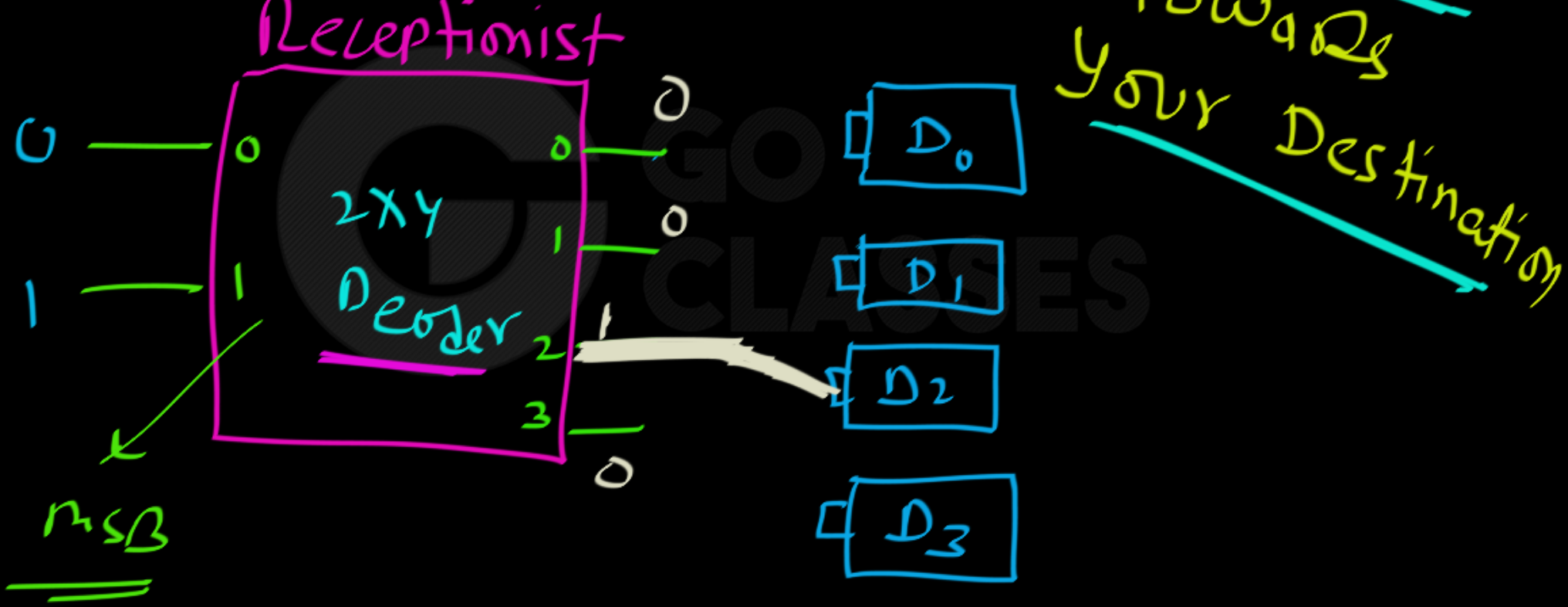



LSB
0
1
I_0
I_1
Y_0
Y_1
Y_2
Y_3
0
0
1
0
MSB

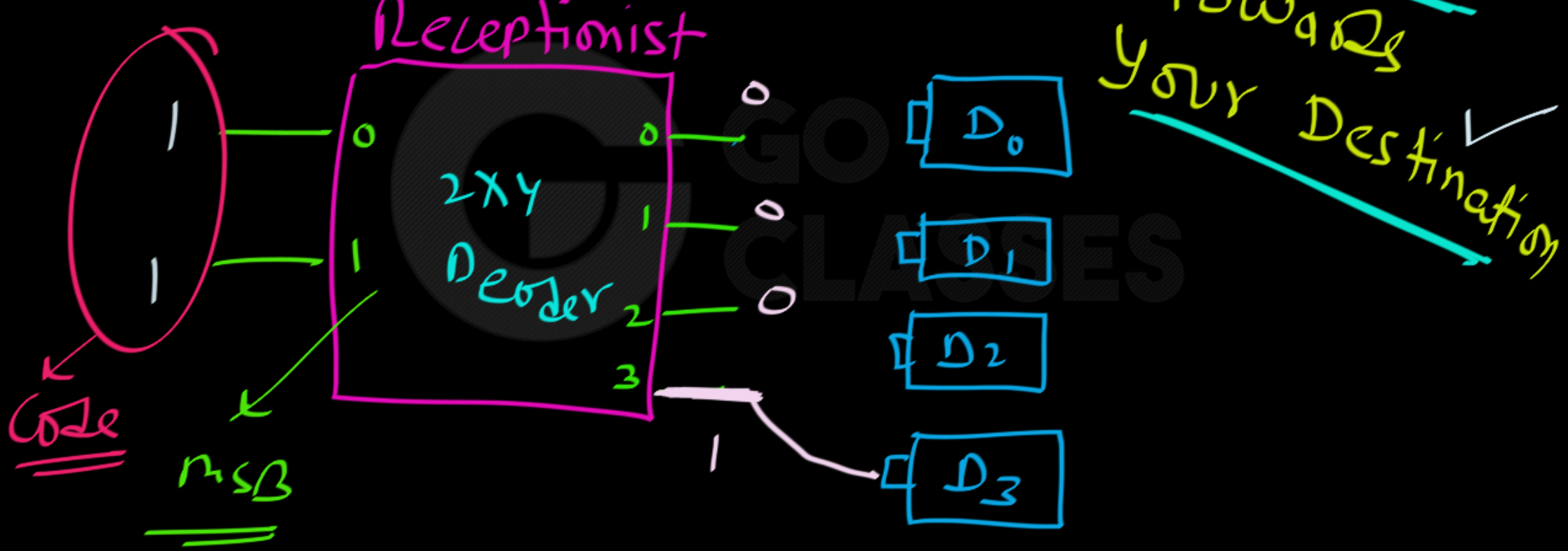

0
0
I_0
I_1
Y_0
Y_1
Y_2
Y_3
1
0
0
0


1
0
I_0
I_1
Y_0
Y_1
Y_2
Y_3
0
1
0
0

Decoder : Receptionist ⟹ Guide you towards your Destination



Decoder : Receptionist ⟹ Guide you towards your Destination

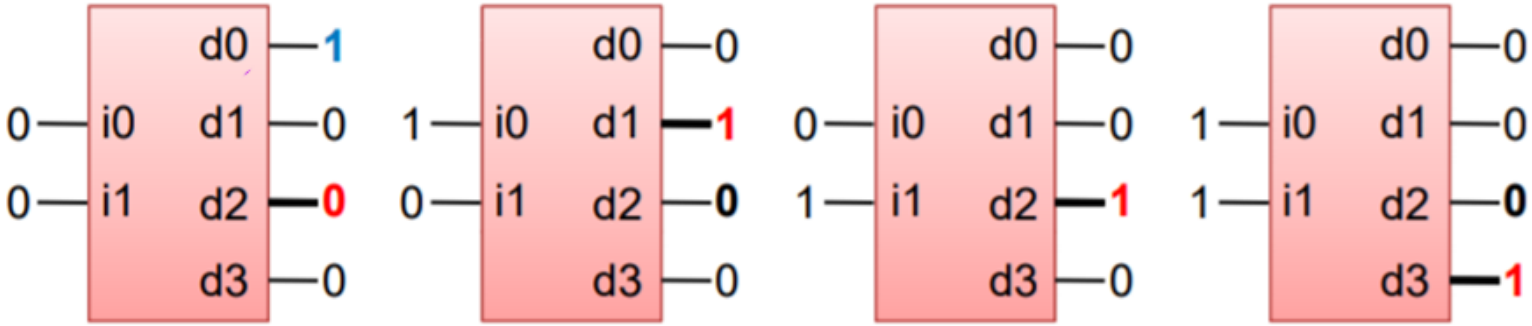


Decoder

- Reception counter : When you reach a Academic Institute
  - Receptionist Ask: Which Dept to Go ?
  - Customer : CSE — the Encoding
  - Receptionist Redirect you to some building according to your Answer. == > Go to Core II
- Decoder : knows what to do with this: Decode
- Digital Case: == > N input: 2^N output
- Memory Addressing
  - Address to a particular location

Decoders

- **Decoder**: Popular combinational logic building block, in addition to logic gates
  - Converts input binary number to one high output
- 2-input decoder: four possible input binary numbers
  - So has four outputs, one for each possible input binary number



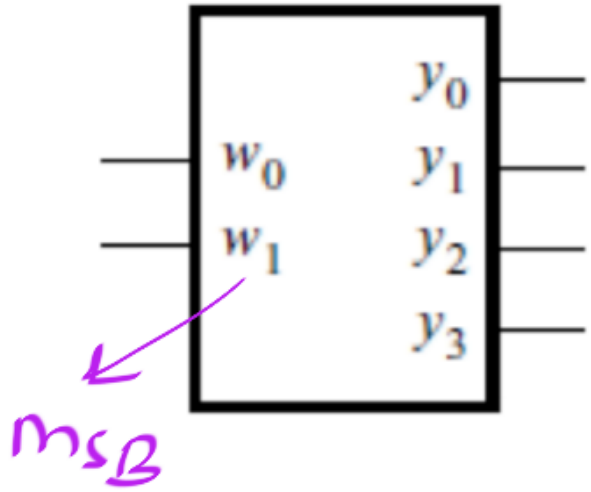
2-to-4 Decoder (Definition)

- Has two inputs: w_1 and w_0
- Has four outputs: y_0 , y_1 , y_2 , and y_3
- If w_1=0 and w_0=0, then the output y_0 is set to 1
- If w_1=0 and w_0=1, then the output y_1 is set to 1
- If w_1=1 and w_0=0, then the output y_2 is set to 1
- If w_1=1 and w_0=1, then the output y_3 is set to 1
- Only one output is set to 1. All others are set to 0.

Truth Table and Graphical Symbol for a 2-to-4 Decoder

w_1	w_0	y_0	y_1	y_2	y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

(a) Truth table



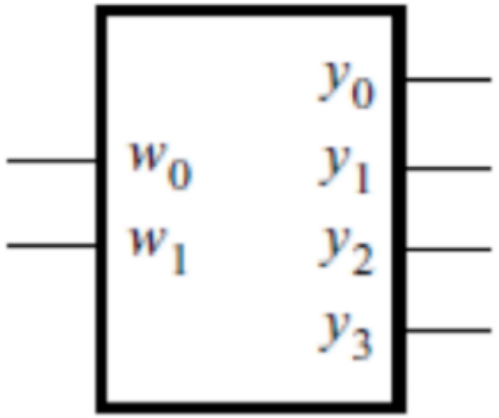
(b) Graphical symbol

Truth Table and Graphical Symbol for a 2-to-4 Decoder

w_1	w_0	y_0	y_1	y_2	y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

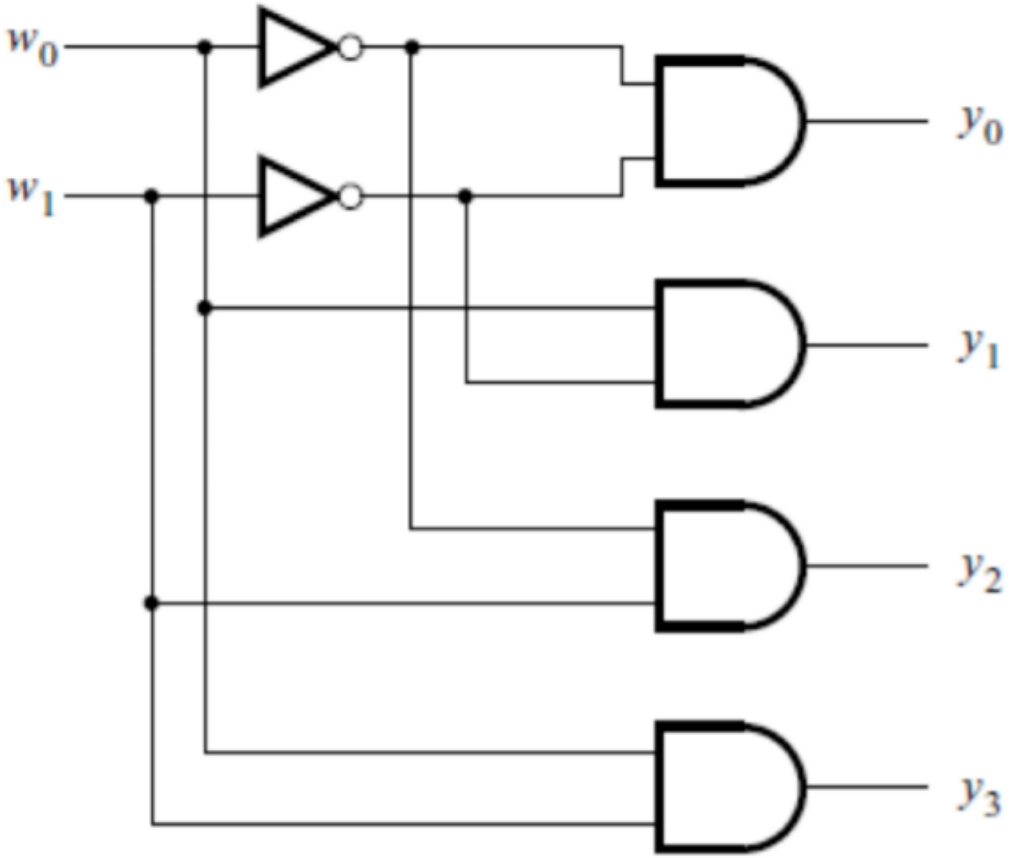
The outputs are "one-hot" encoded

(a) Truth table

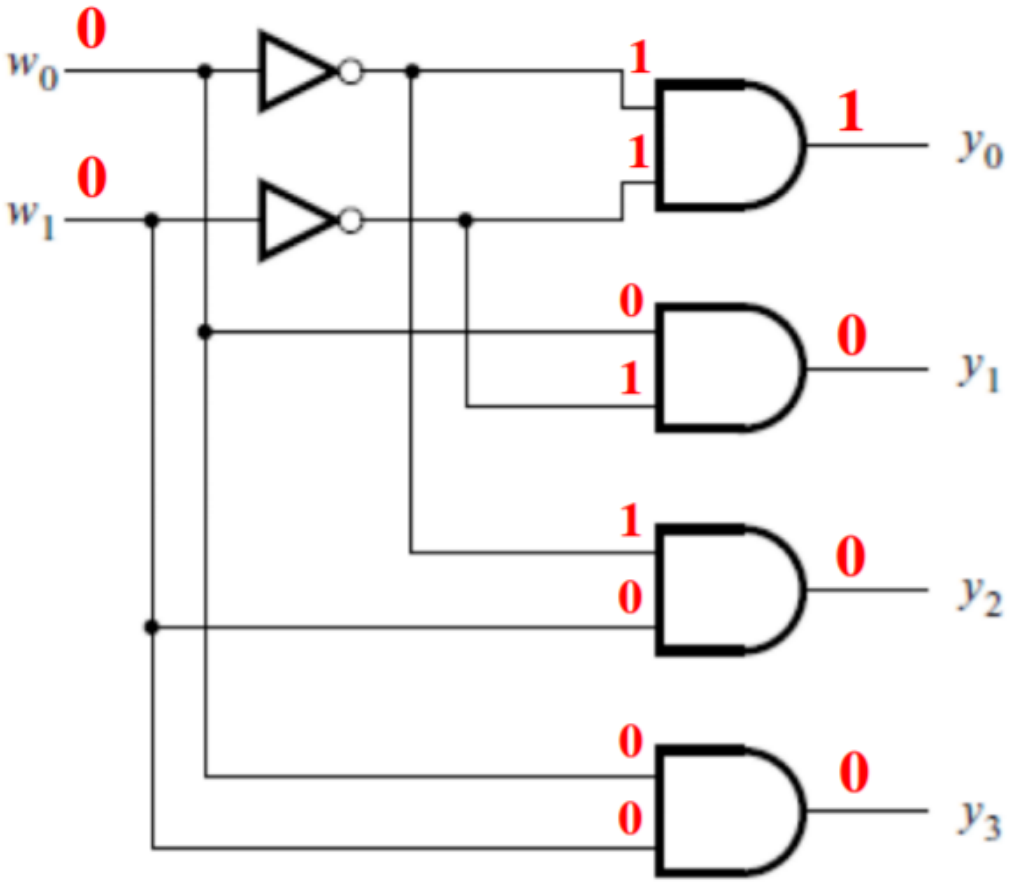


(b) Graphical symbol

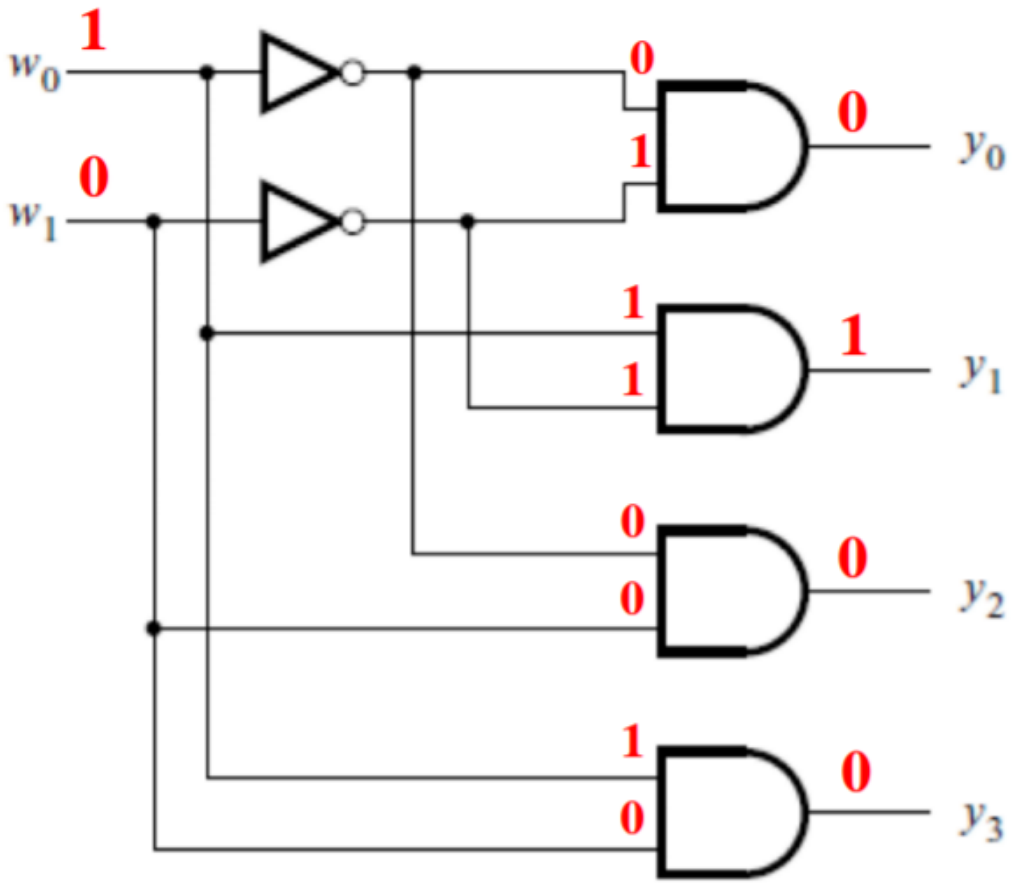
The Logic Circuit for a 2-to-4 Decoder



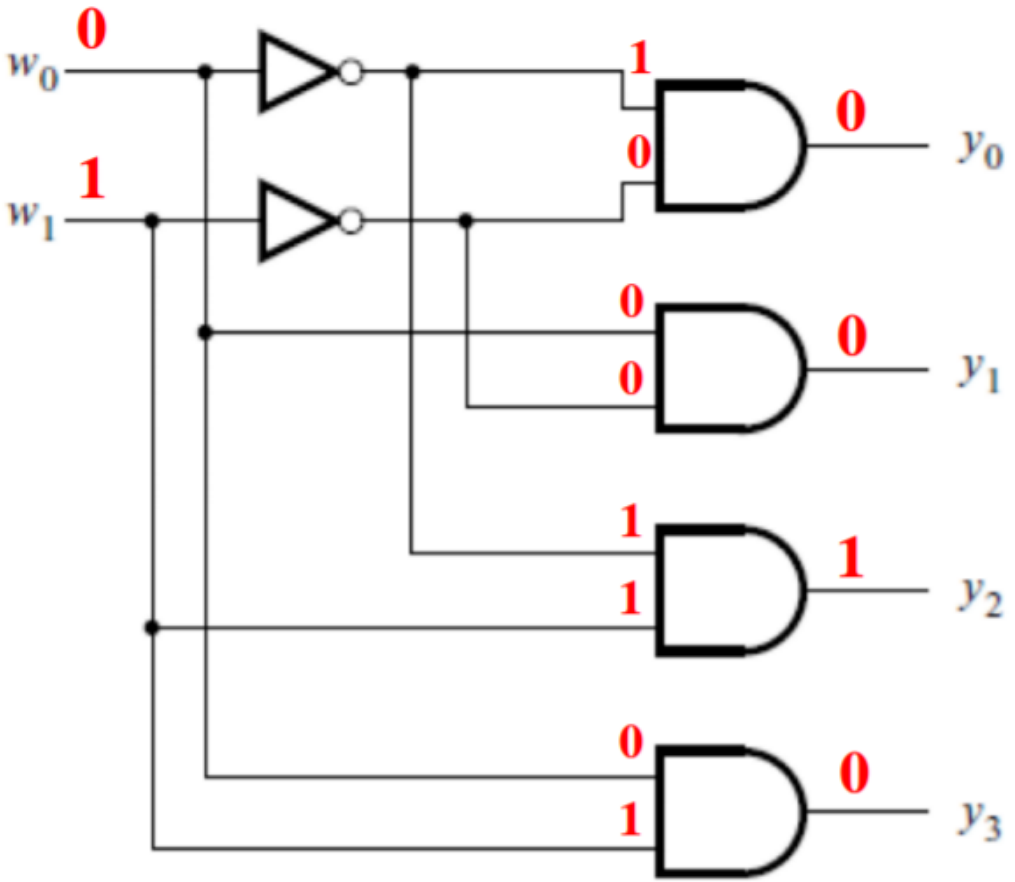
The Logic Circuit for a 2-to-4 Decoder



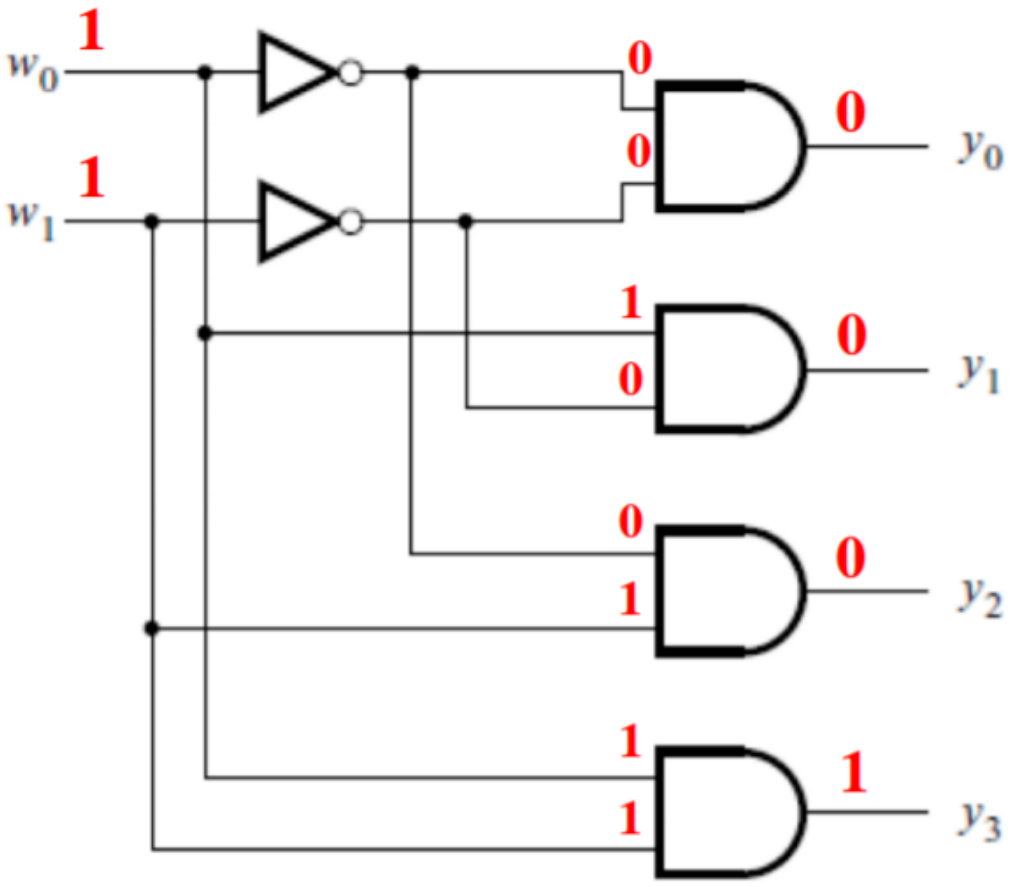
The Logic Circuit for a 2-to-4 Decoder



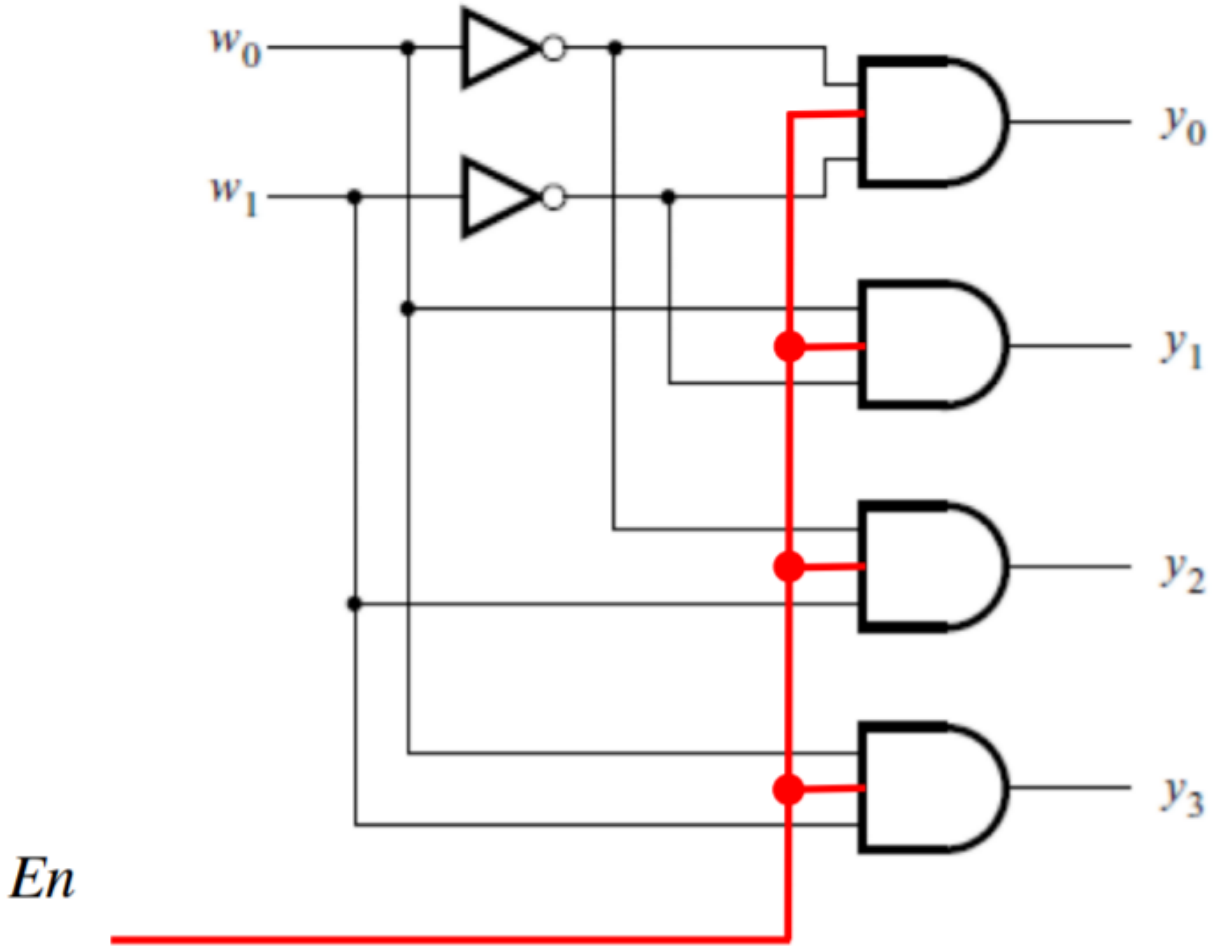
The Logic Circuit for a 2-to-4 Decoder



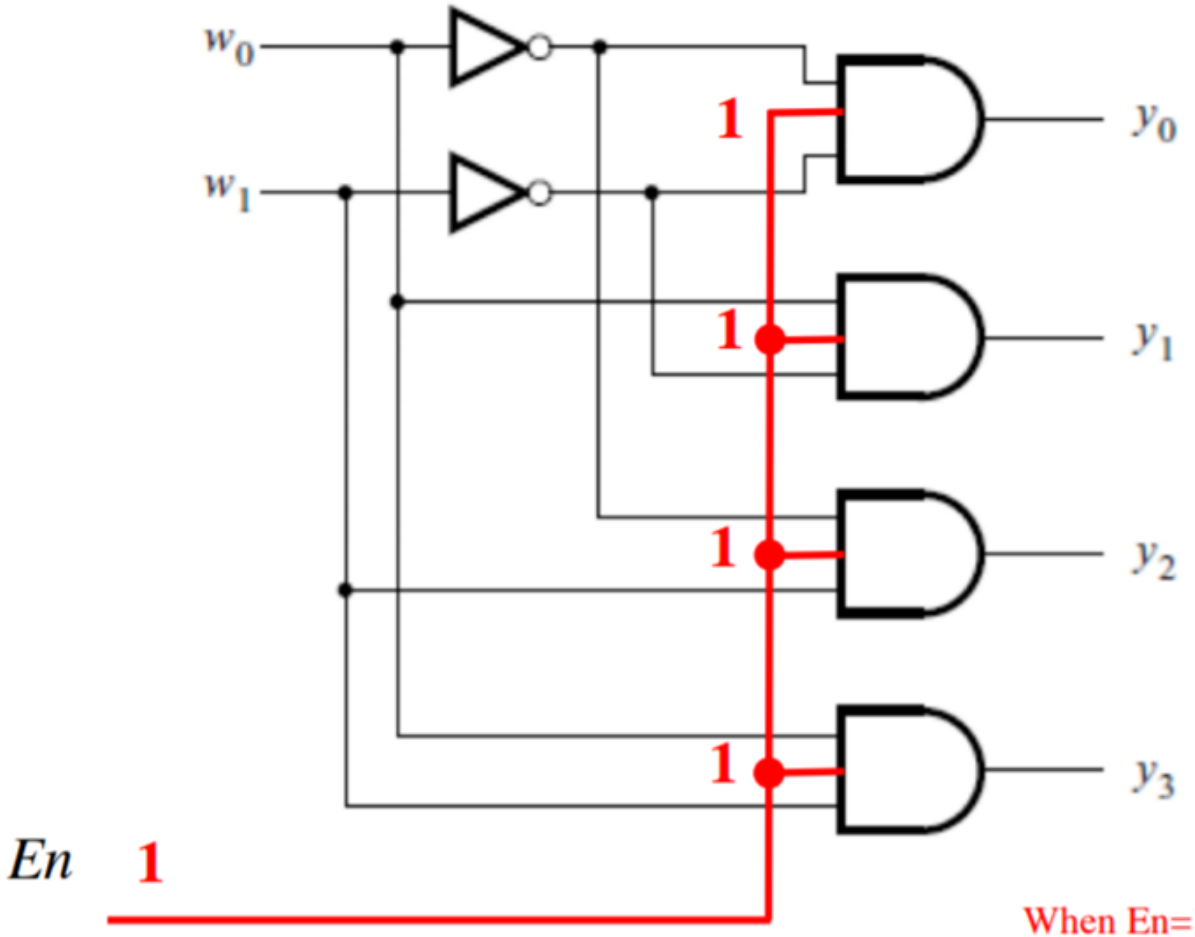
The Logic Circuit for a 2-to-4 Decoder



Adding an Enable Input

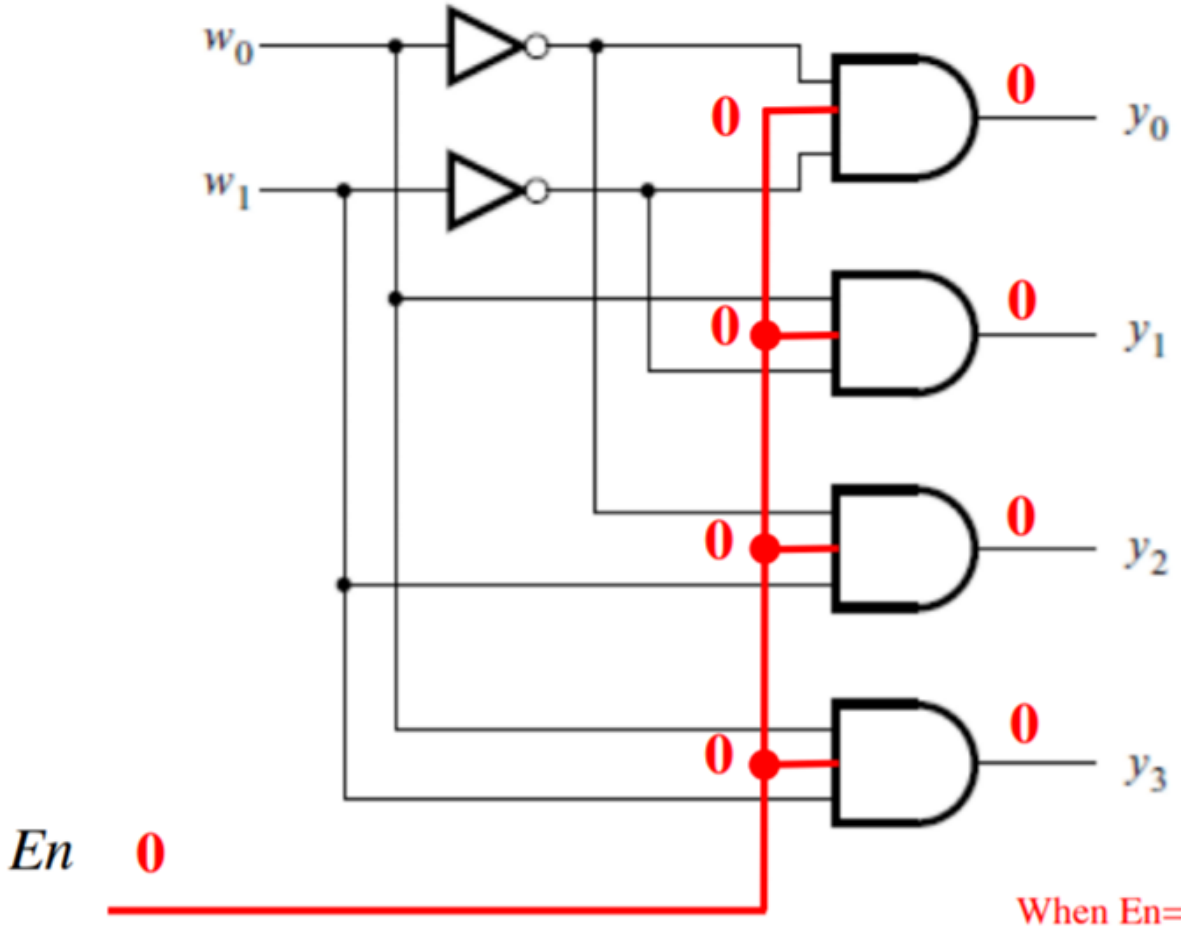


Adding an Enable Input



When En=1 this circuit behaves like the one without enable.

Adding an Enable Input

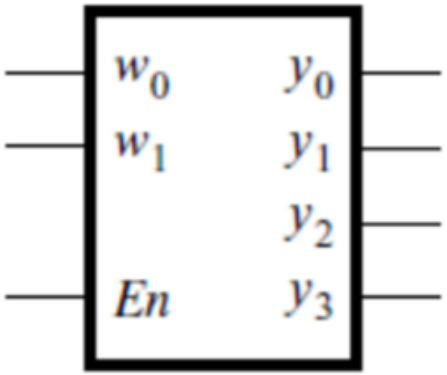


When En=0 all outputs are set to 0 and the decoder is disabled.

Truth Table and Graphical Symbol for a 2-to-4 Decoder with an Enable Input

En	w_1	w_0	y_0	y_1	y_2	y_3
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1
0	x	x	0	0	0	0

(a) Truth table



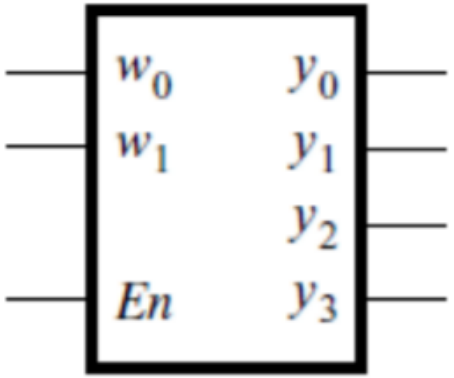
(b) Graphical symbol

x indicates that it does not matter what the value of this variable is for this row of the truth table

Truth Table and Graphical Symbol for a 2-to-4 Decoder with an Enable Input

En	w_1	w_0	y_0	y_1	y_2	y_3
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0

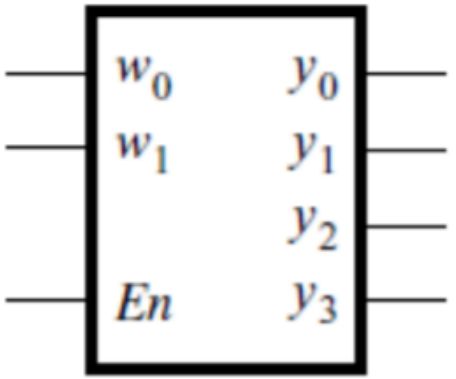
$En=0$



(b) Graphical symbol

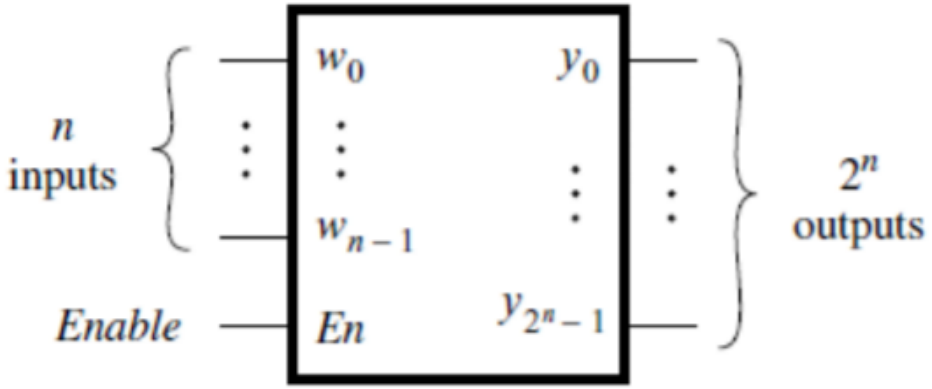
Truth Table and Graphical Symbol for a 2-to-4 Decoder with an Enable Input

En	w_1	w_0	y_0	y_1	y_2	y_3
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1



(b) Graphical symbol

Graphical Symbol for a Binary n-to-2^n Decoder with an Enable Input

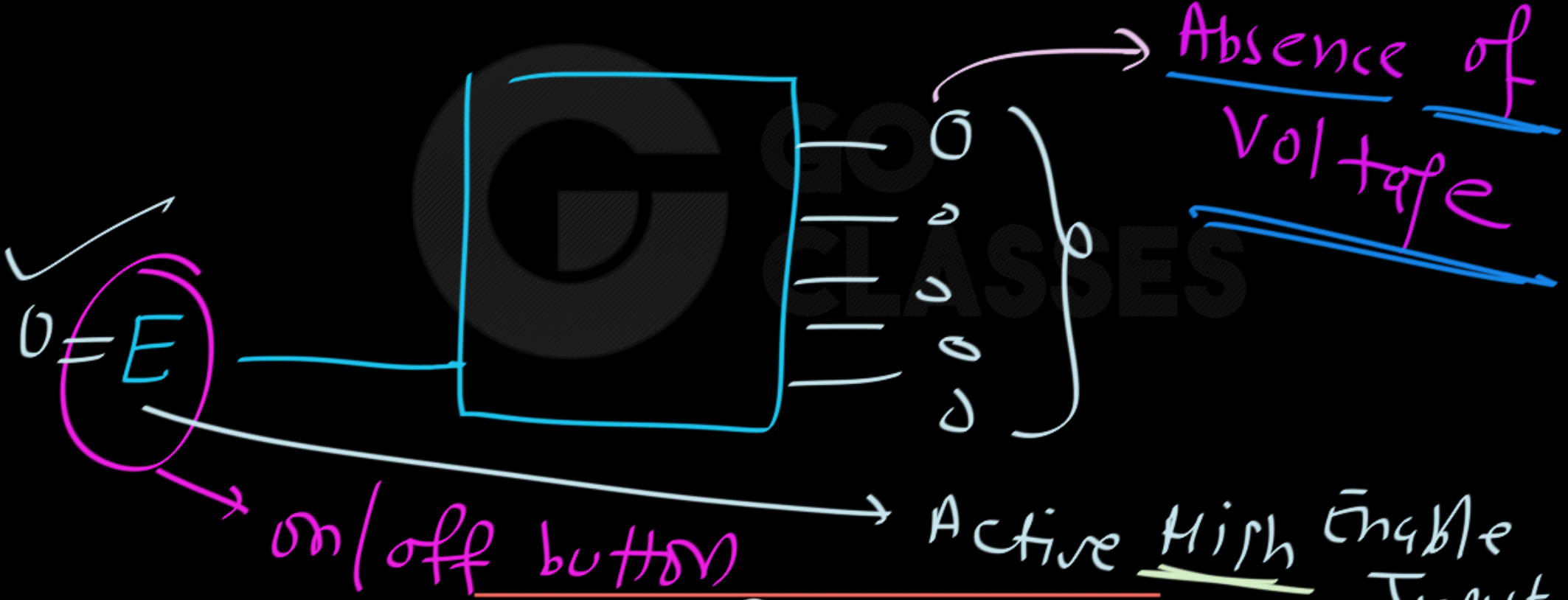


(d) An n-to-2^n decoder

A binary decoder with n inputs has 2^n outputs

The outputs of an enabled binary decoder are "one-hot" encoded, meaning that only a single bit is set to 1, i.e., it is *hot*.

Any Combinational Digital Circuit :

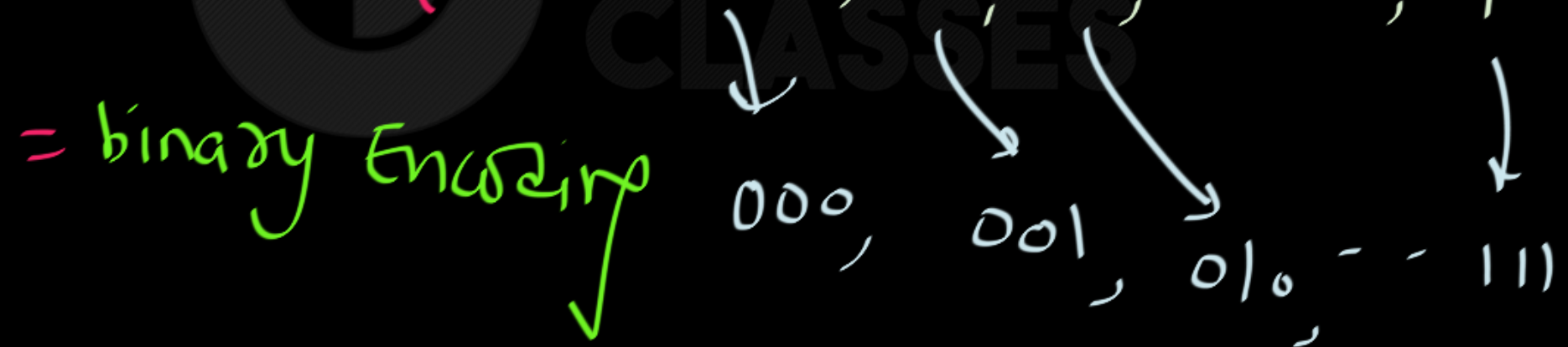


Example :

Encoder : octal to binary Encoder

input = octal Digit

output = binary Encoding



Example:

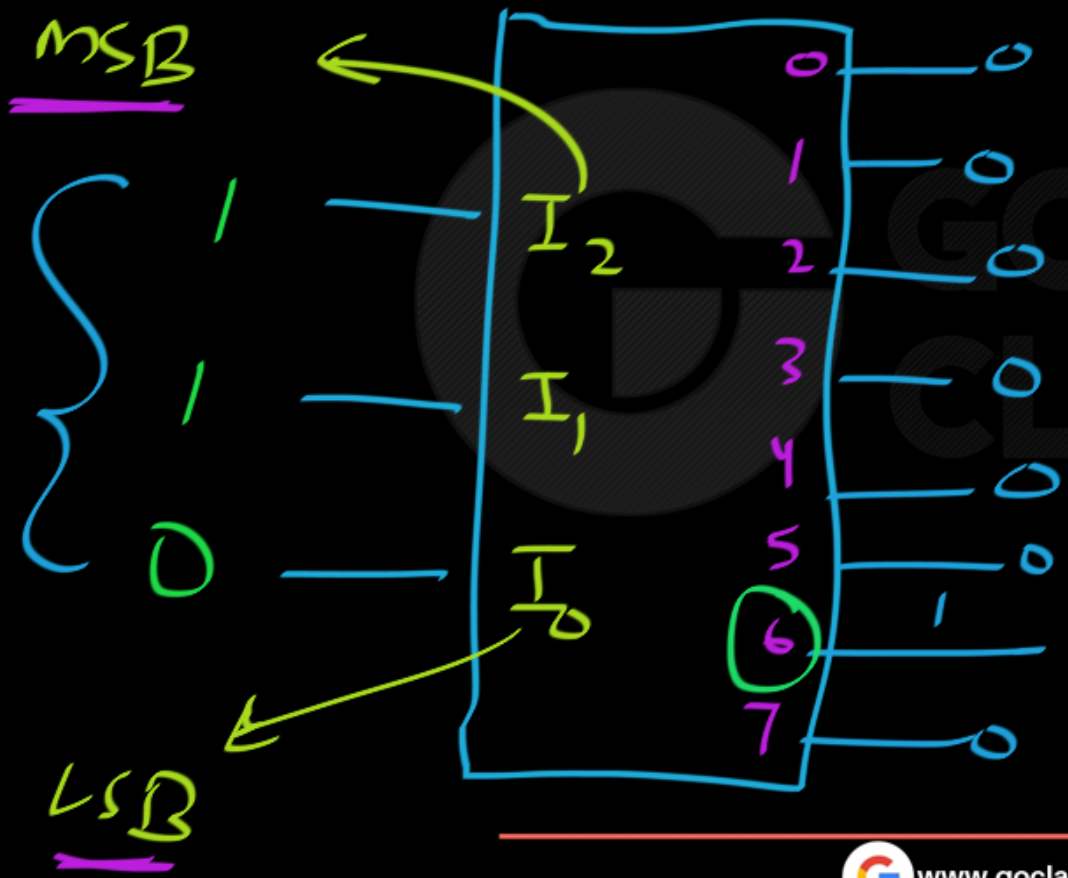
Encoder: octal to binary Encoder

Decoder: binary Encoding to octal Digit

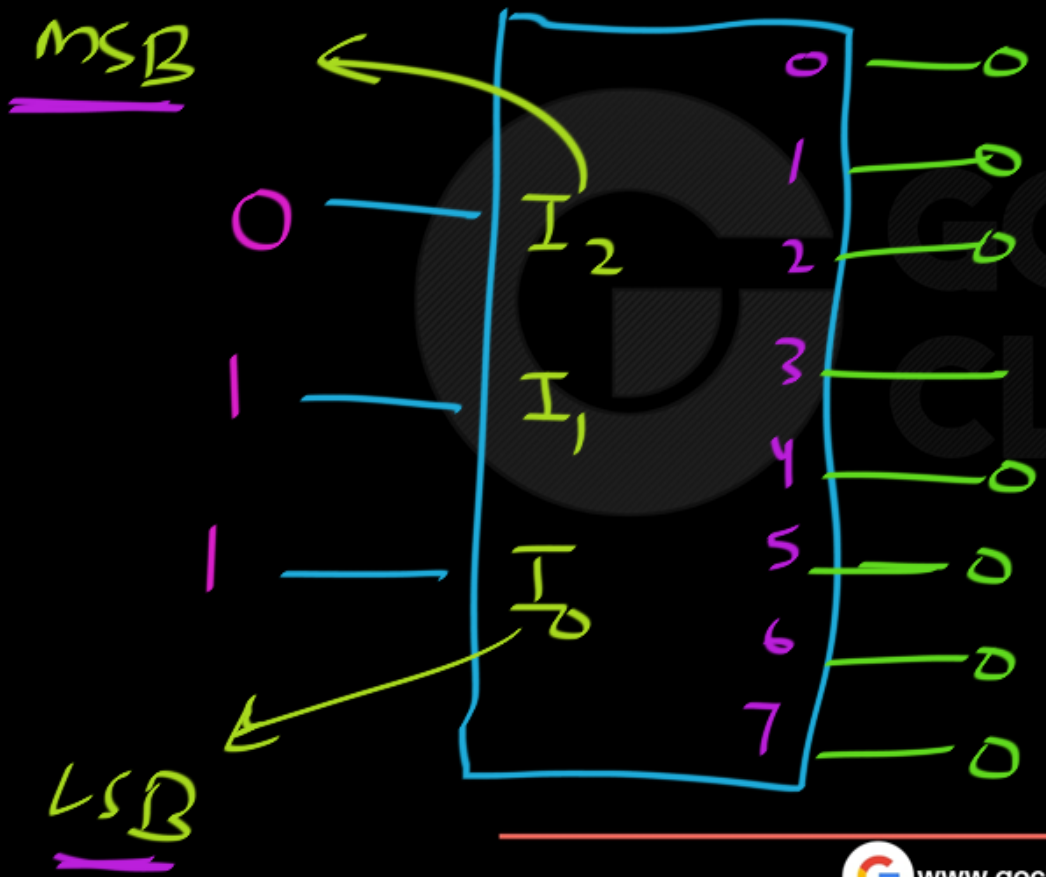
input: 3 bit binary Encoding

o/p : Octal Digit

binary to octal Decoder : 3X8 Decoder



binary to octal Decoder : 3X8 Decoder

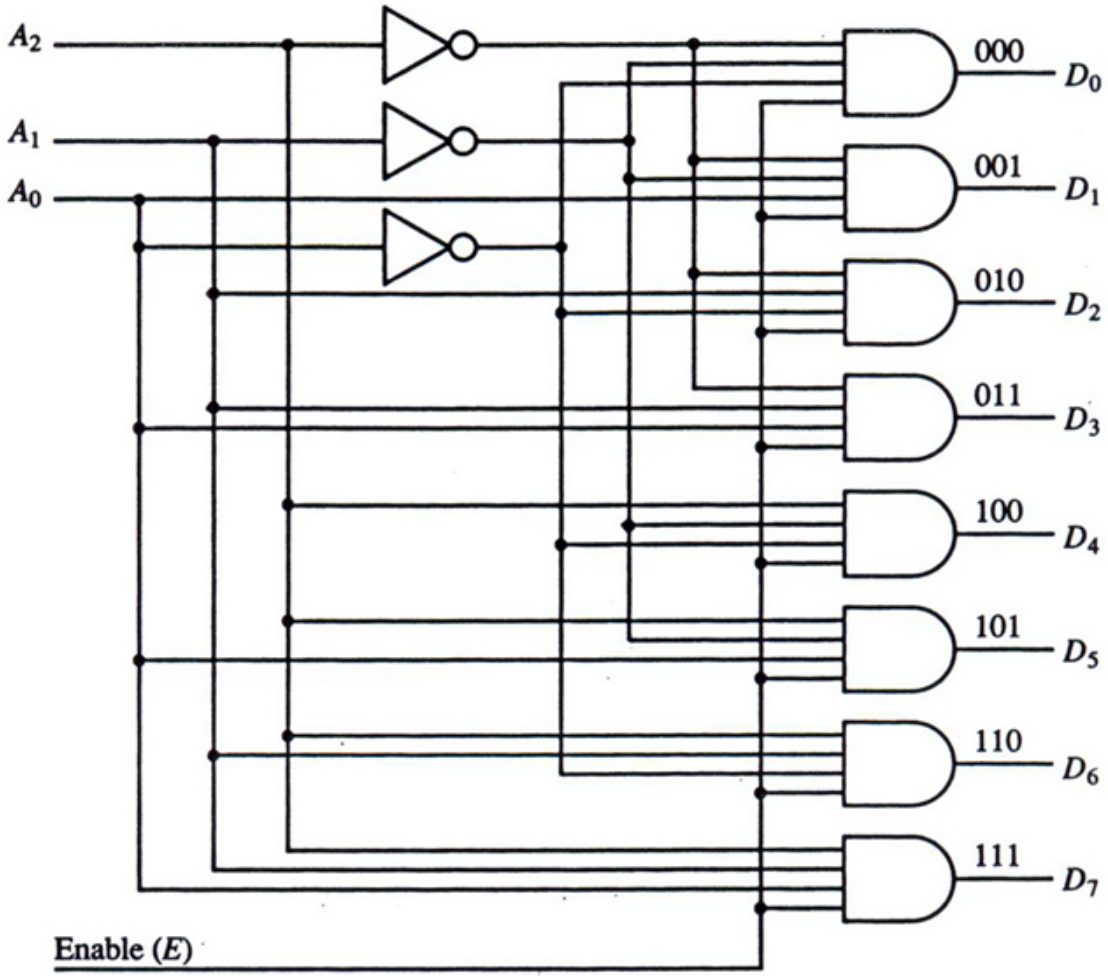


I_2	I_1	I_0	Y_0	Y_1	Y_2	-	-	Y_7
0	0	0	1	0	0	-	-	0
0	0	1	0	1	0	-	-	0
0	1	0	0	0	1	-	-	0
⋮	⋮	⋮						
1	1	1	0	0	0	0	0	1

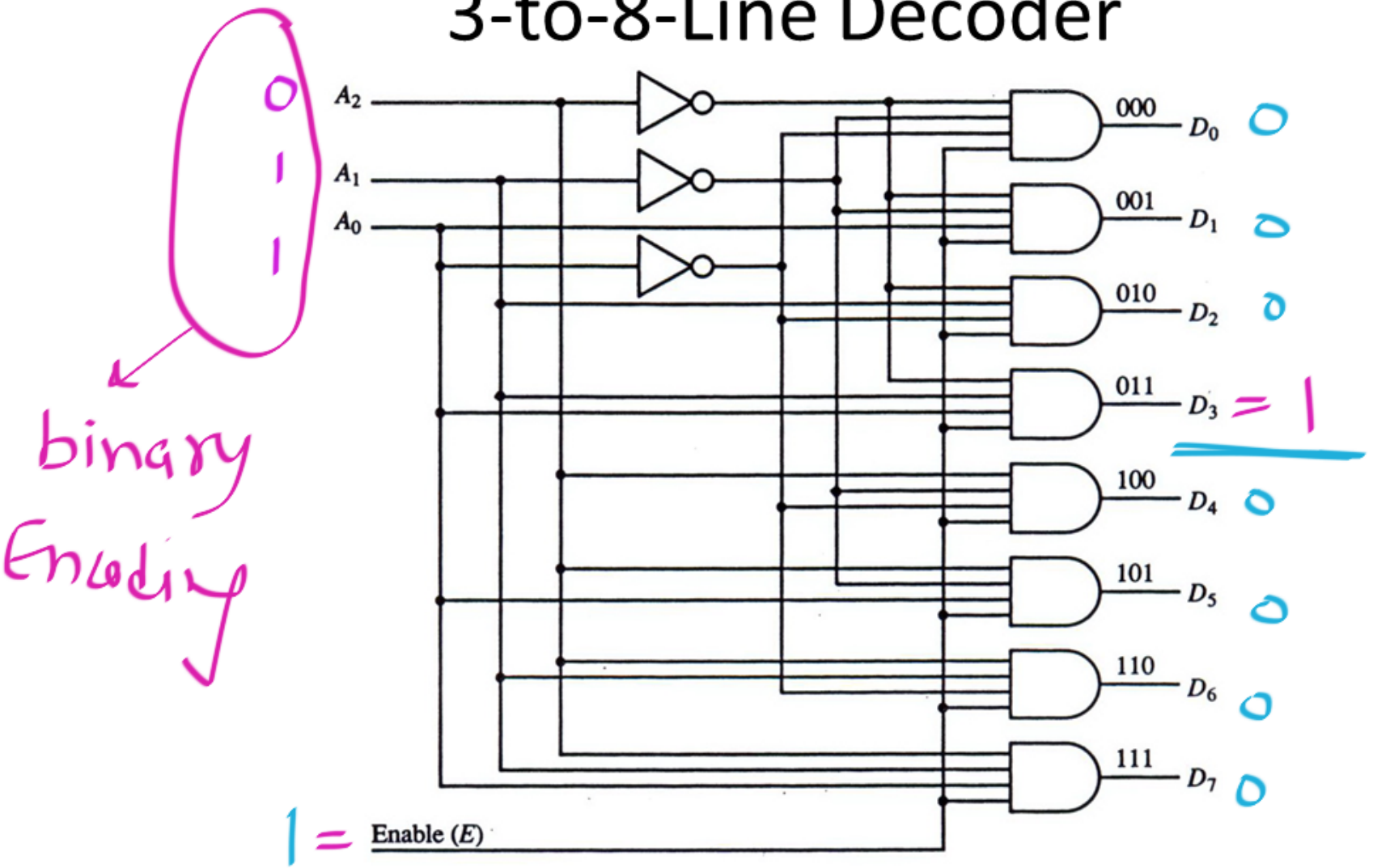
Decoders

- A decoder is a combinational circuit that converts binary information from the n coded inputs to a maximum of 2^n unique outputs.
- The decoders presented in this sections are called n-to-m-line decoders, where $m = 2^n$.
- Their purpose is to generate the 2^n binary combinations of the n input variables.
- A decoder has n inputs and m outputs and is also referred to as an $n \times m$ decoder.

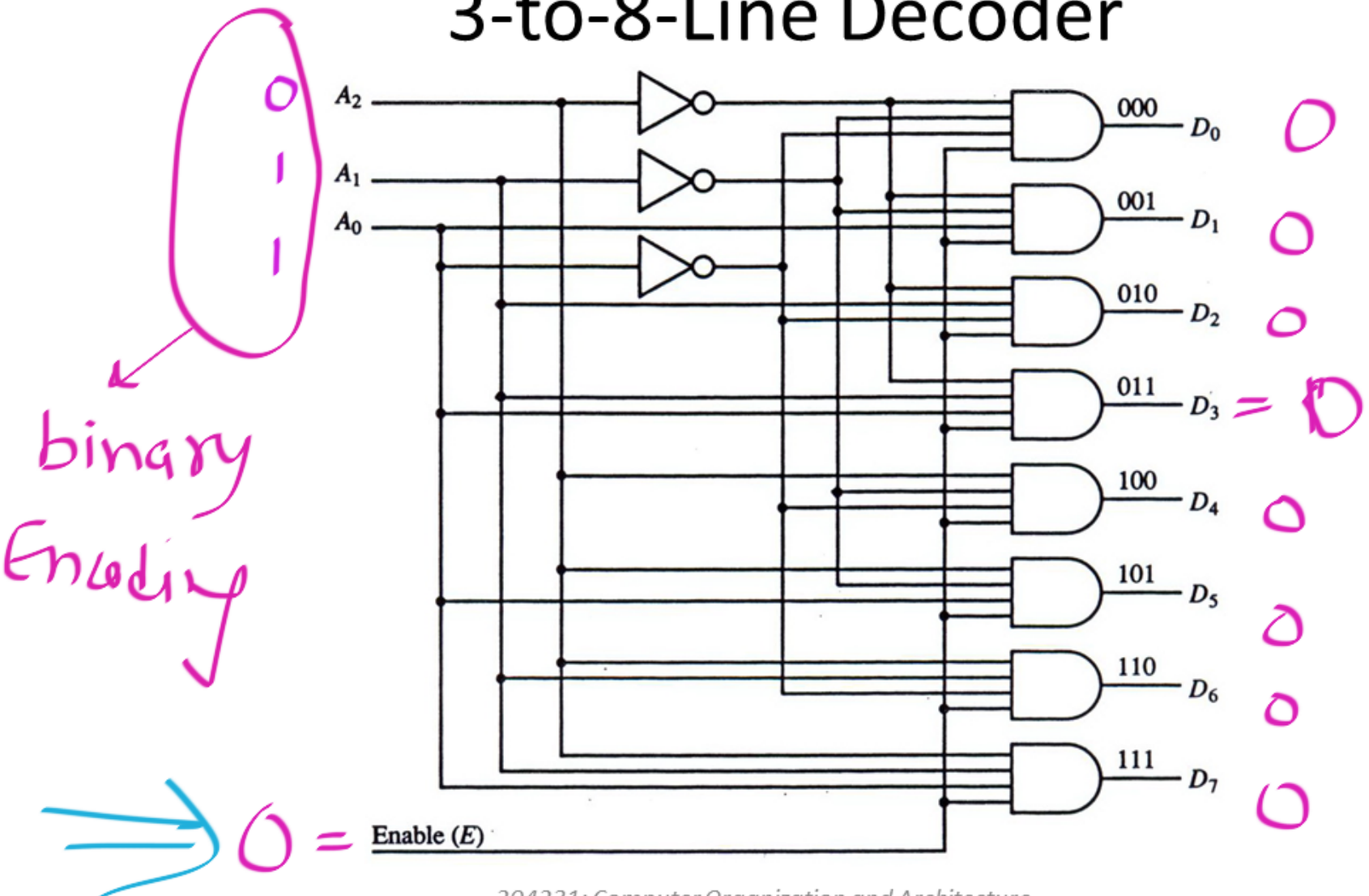
3-to-8-Line Decoder



3-to-8-Line Decoder



3-to-8-Line Decoder



3-to-8-Line Decoder

- The three data inputs, A_0, A_1, and A_2, are decoded into eight outputs, each output represent one of the combinations of the three binary input variables.
- A particular application of this decoder is a binary-to-octal conversion.
- Commercial decoders include one or more enable inputs to control the operation of the circuit.
- The decoder is enabled when E is equal to 1 and disabled when E is equal to 0.

Truth Table for 3-to-8-Line Decoder

- When the enable input E is equal to 0, all the outputs are equal to 0 regardless of the values of the other there data inputs.
- The there x's in the table designate don't-care conditions.
- When the enable input is equal to 1, the decoder operates in a normal fashion.

Truth Table for 3-to-8-Line Decoder

- For each possible input combination, there are seven outputs that are equal to 0 and only one that is equal to 1.
- The output variable whose value is equal to 1 represents the octal number equivalent of the binary number that is available in the input data lines.

Truth Table for 3-to-8-Line Decoder

Enable	Inputs			Outputs							
E	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	×	×	×	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Important Combinational Circuits:

Mux, Decoder, Adder

Very less important:

Demux; Encoder { binary, Priority ⇒ more important

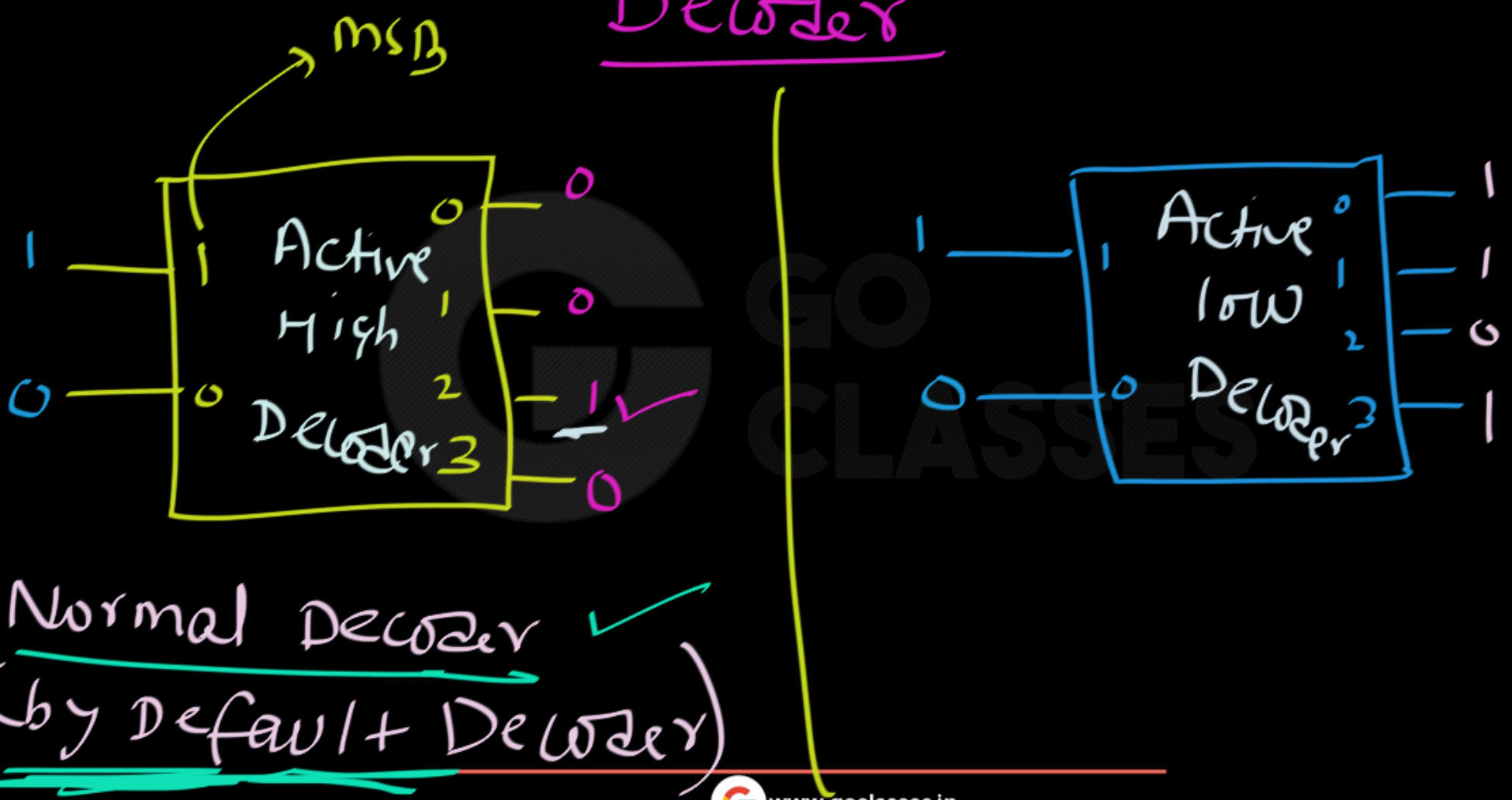
very very less imp.

Next Topic:

Active Low Decoder

(NAND GATE Decoder)

Decoder



Normal Decoder ✓

(by Default Decoder)

Decoder

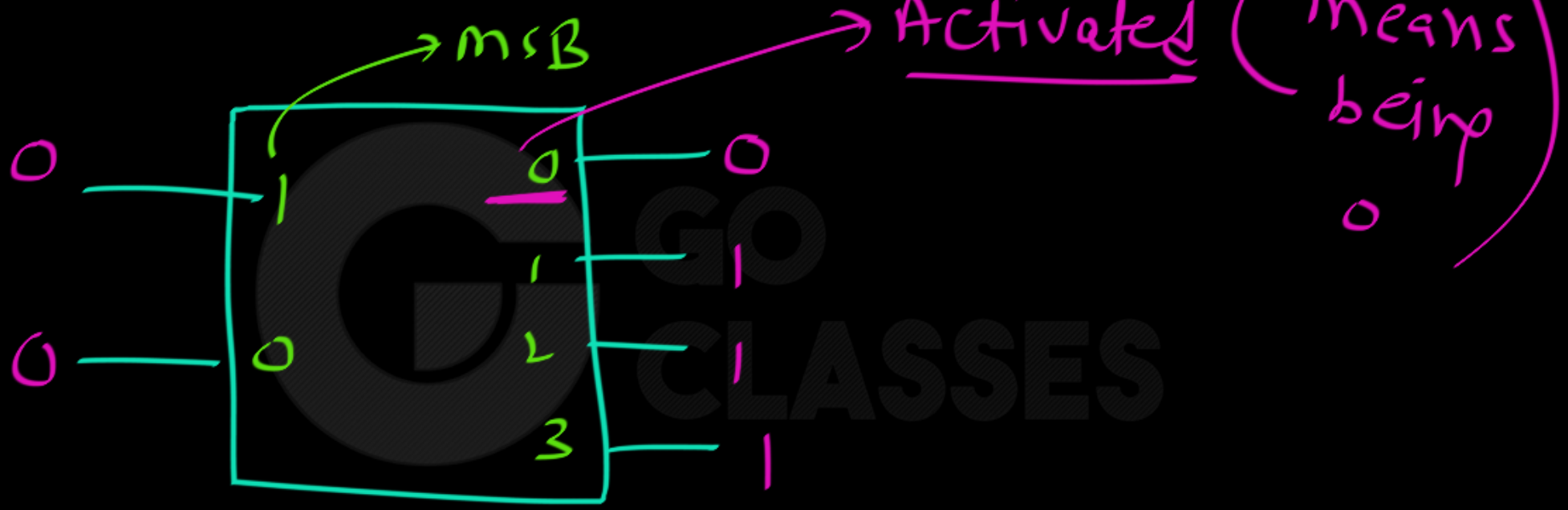
Active High Decoder (Activates = being 1)

for any input Combination
Exactly one o/p is 1/high.

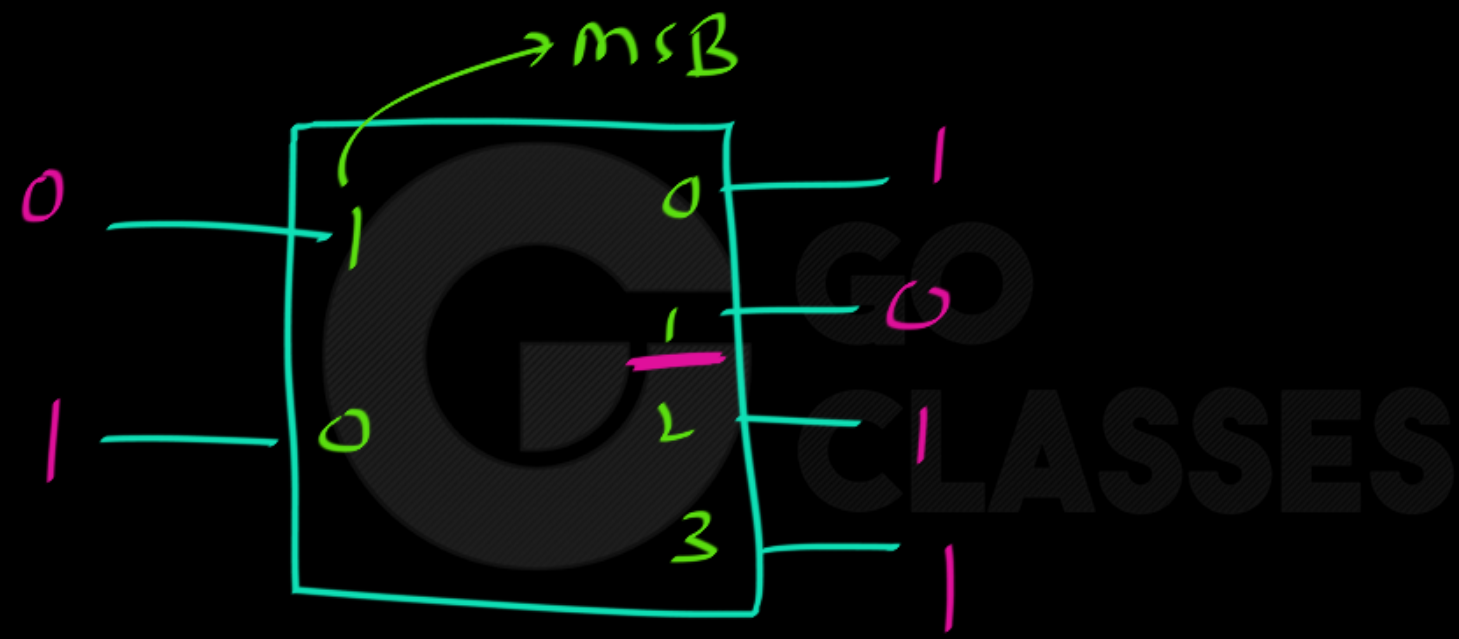
Active low Decoder (Activates = being 0)

for any input Comb,
Exactly one o/p is 0/Low.

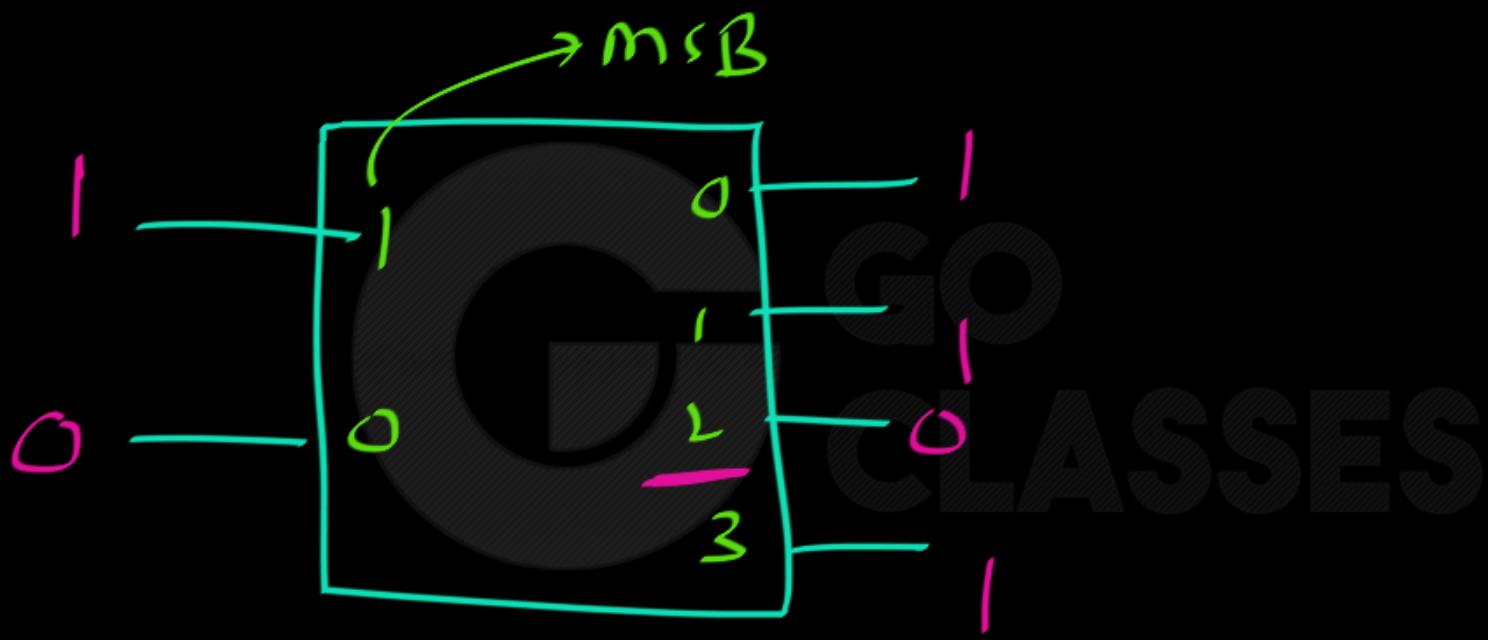
Active low Decoder :



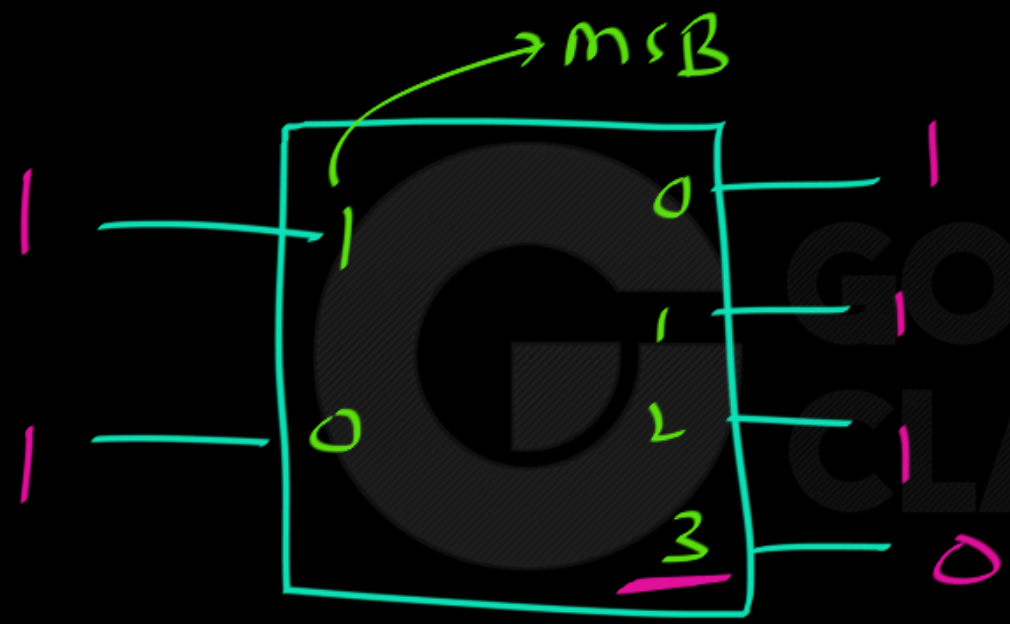
Active low Decoder :



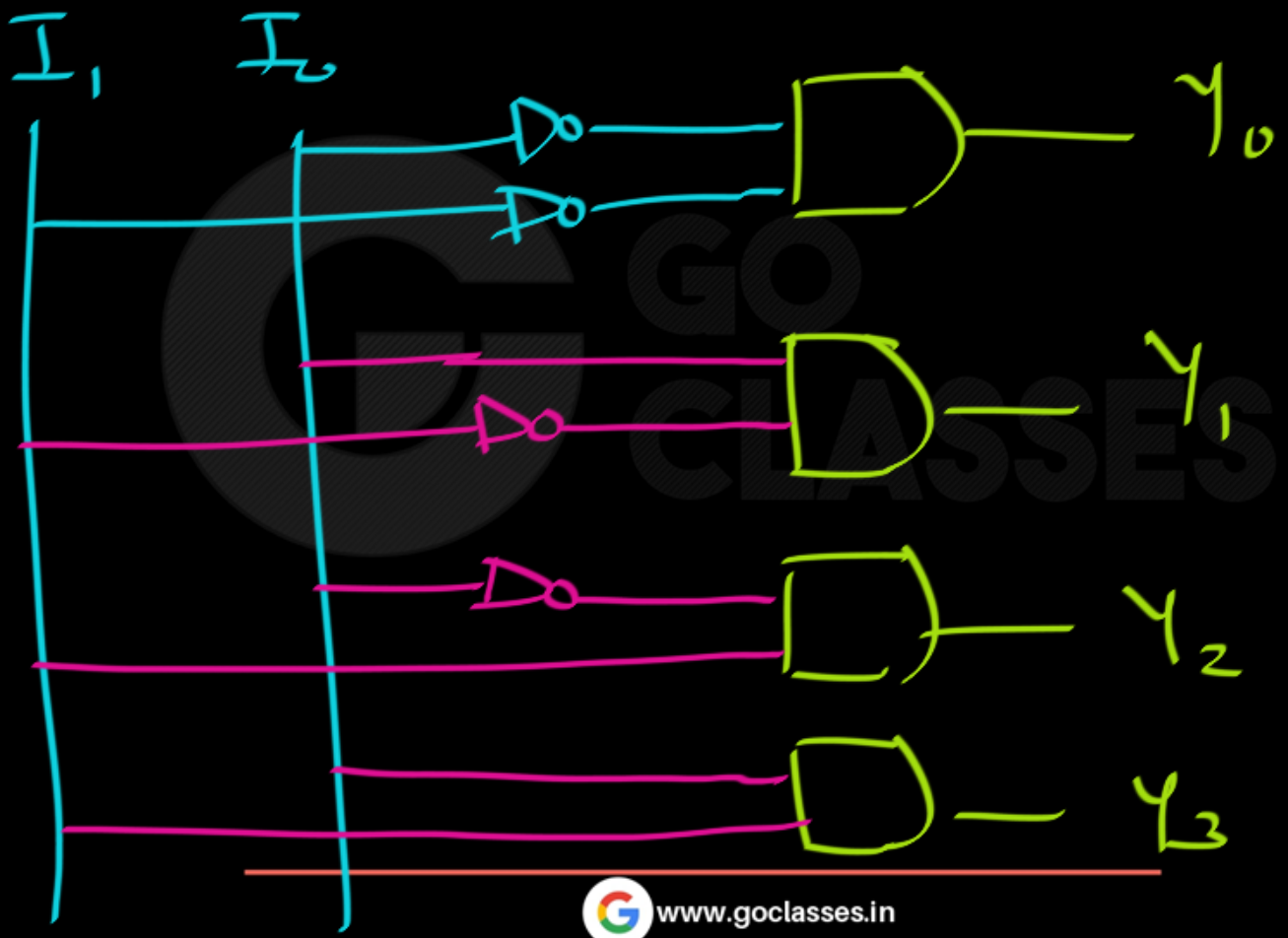
Active low Decoder :



Active low Decoder :



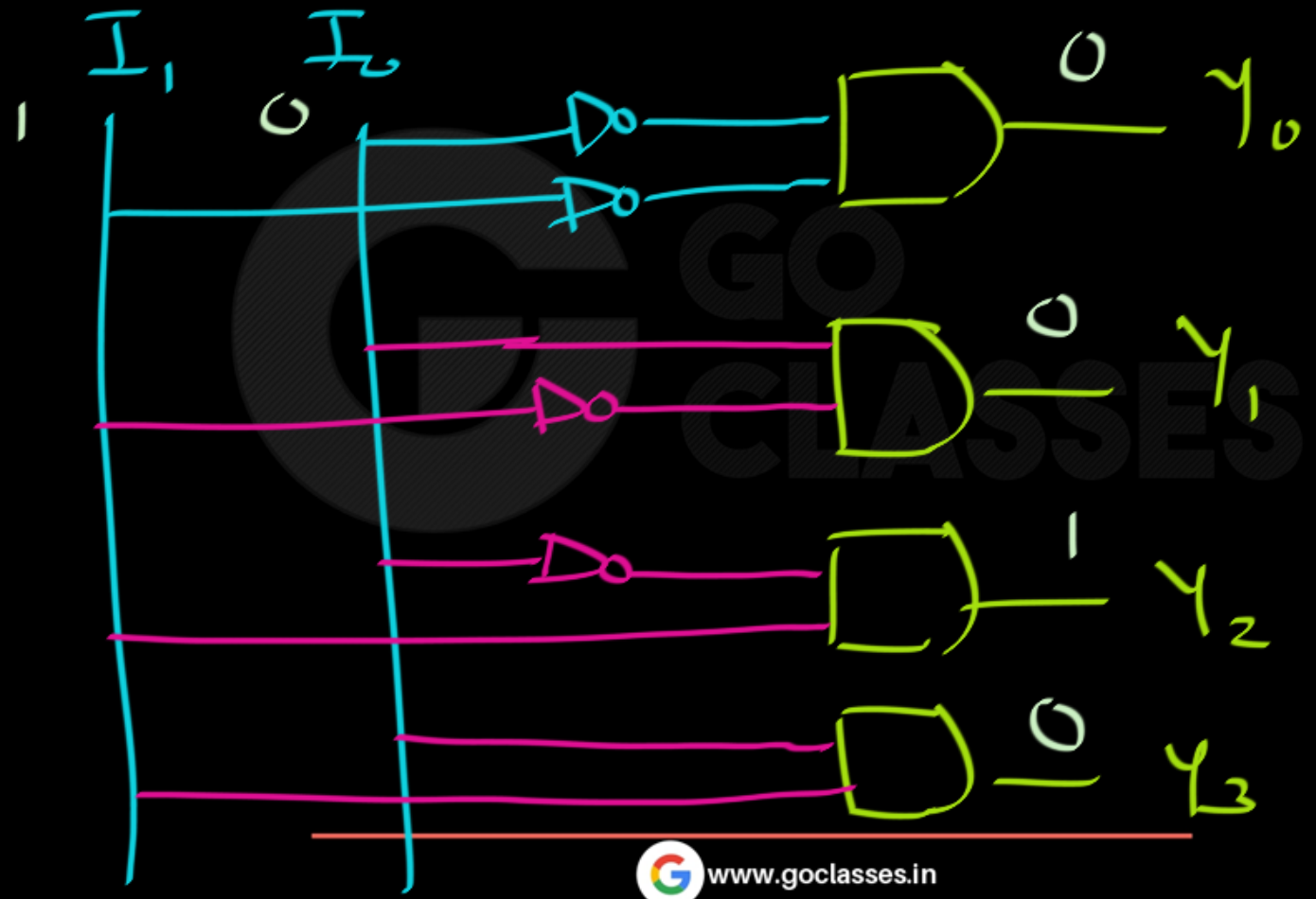
Implementation: (Active High Decoder)



Normal (by Default) Decoder

"AND" implementation of Decoder

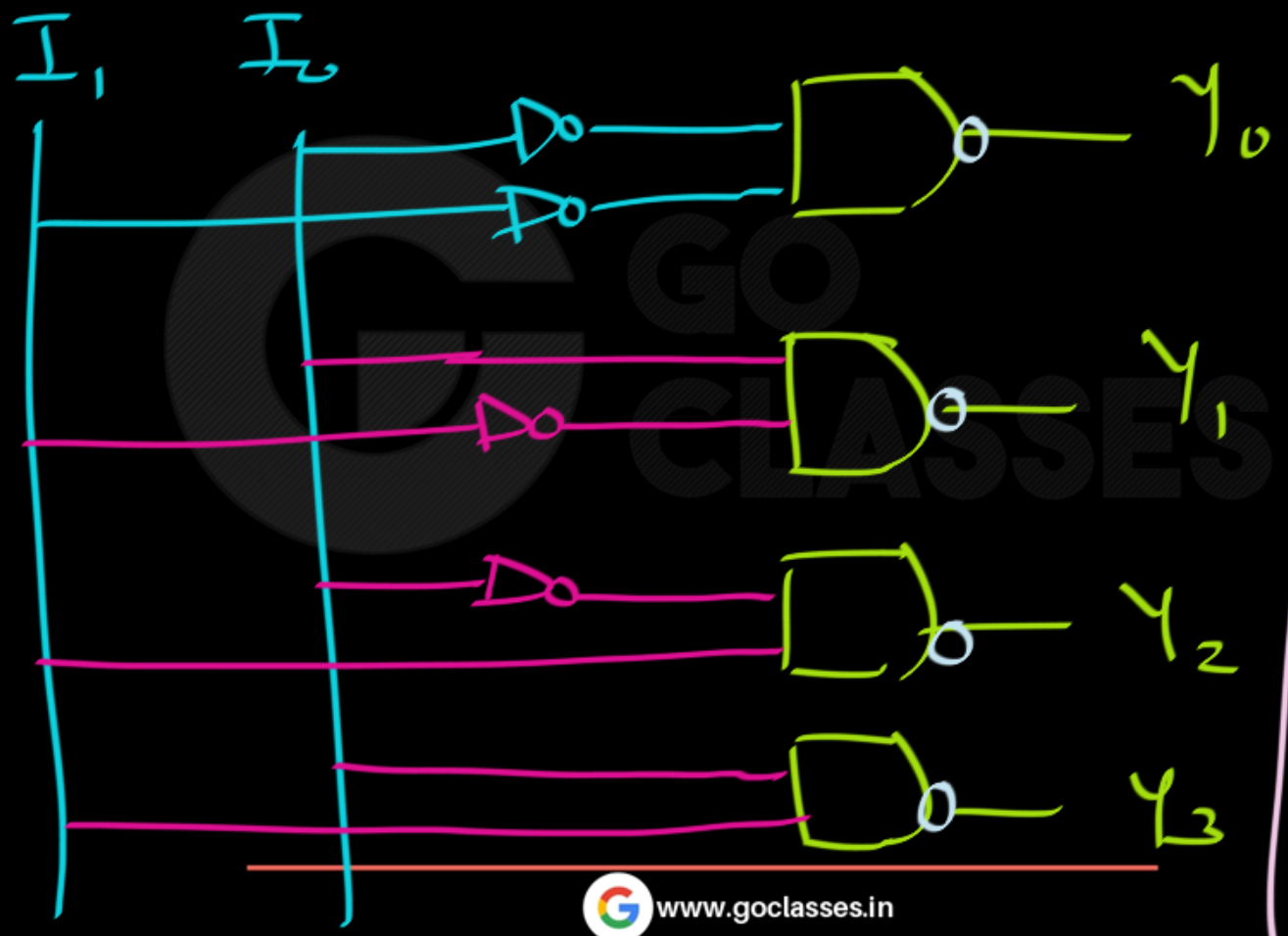
Implementation : (Active High Decoder)



Normal (by Default) Decoder

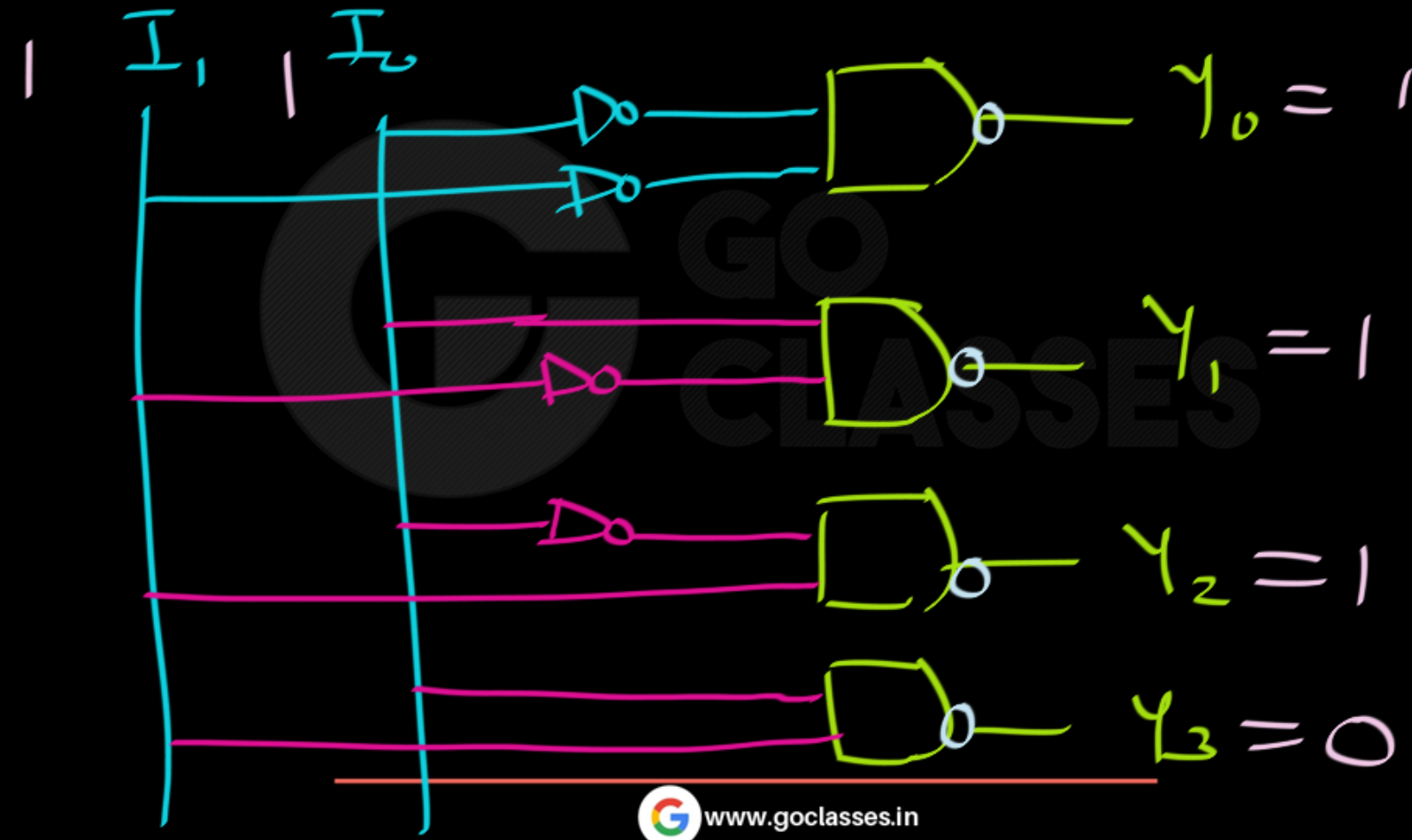
"AND" implementation of Decoder

Implementation: (Active Low Decoder)

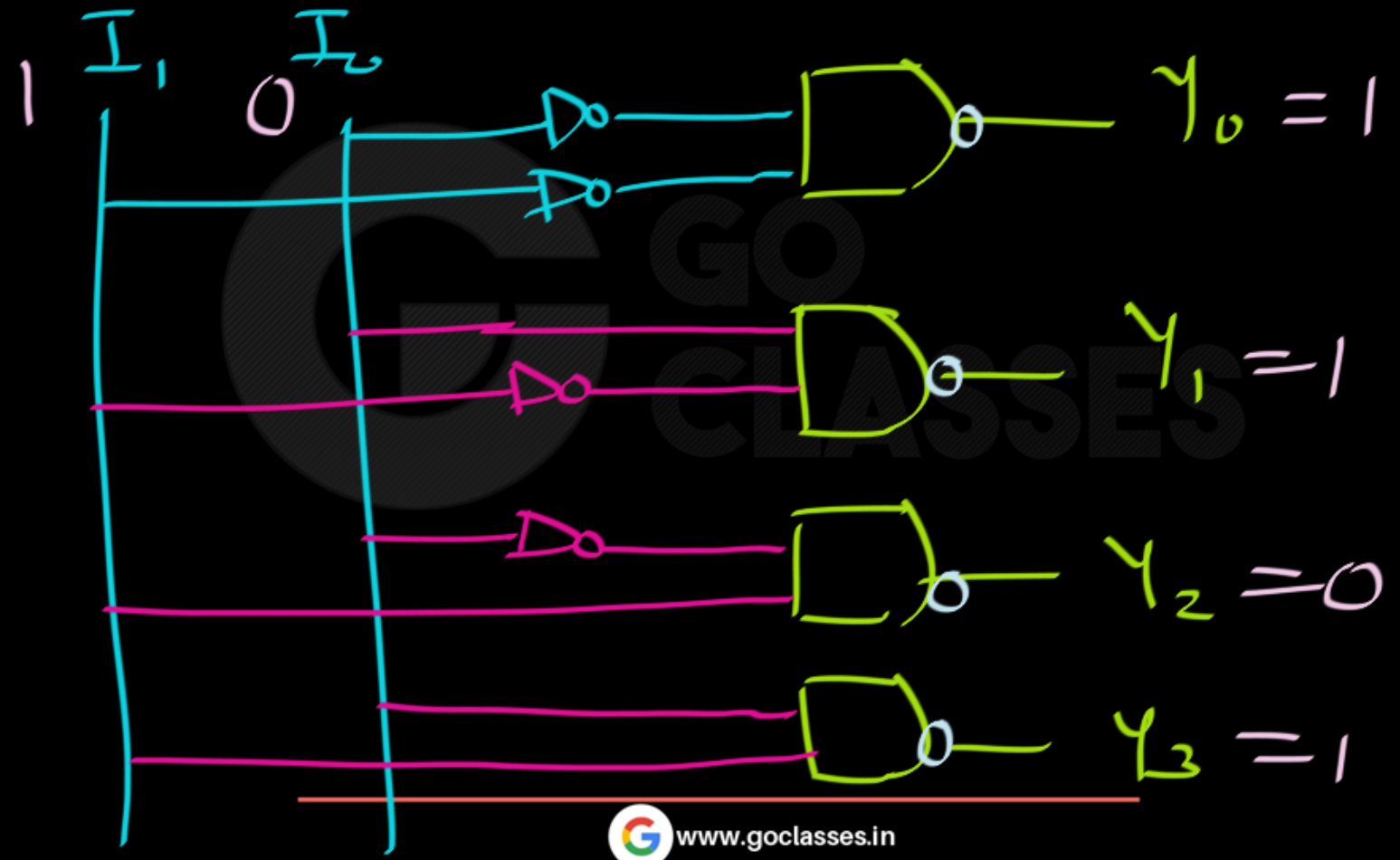


NAND implementation of decoder

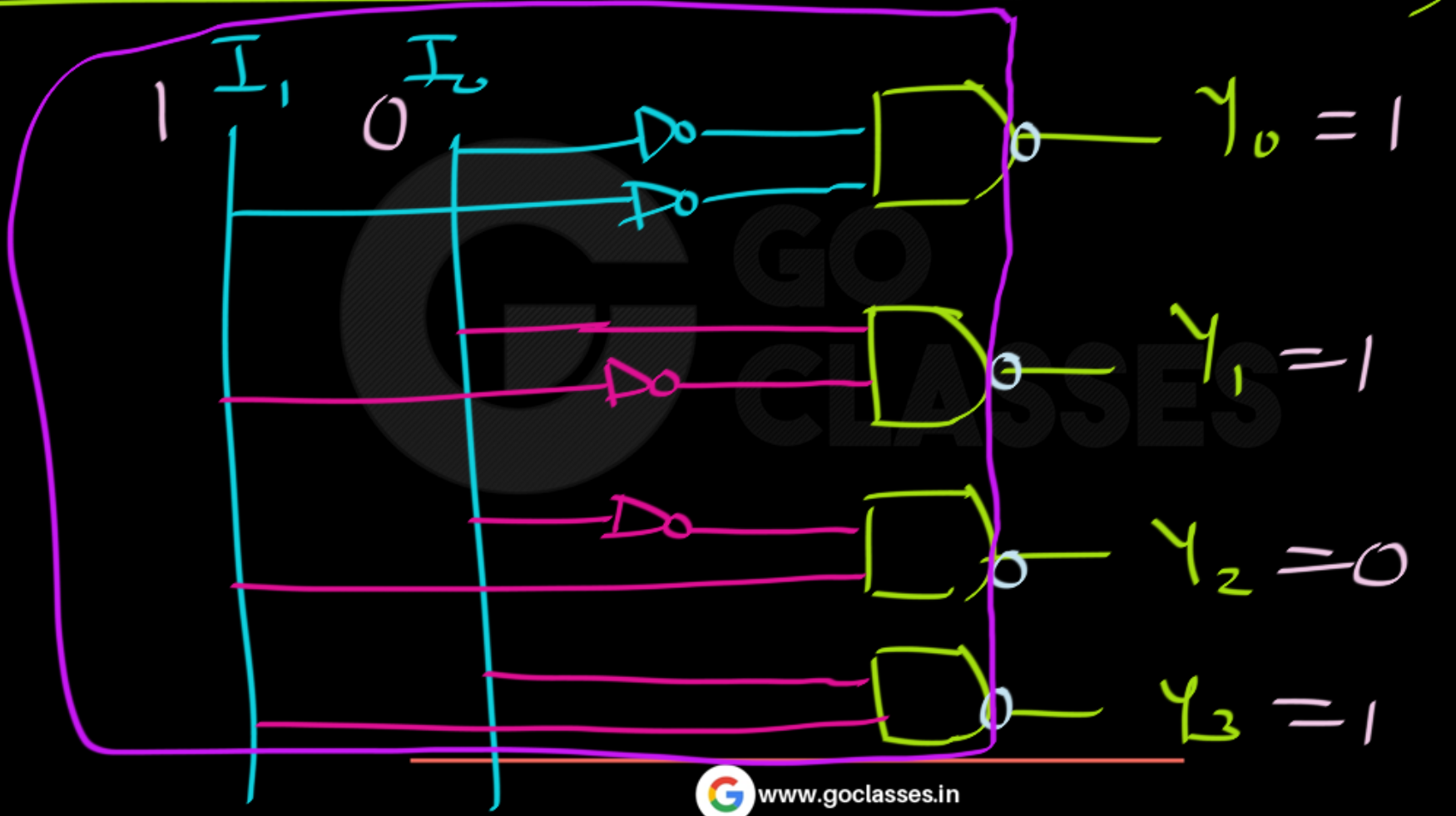
Implementation : (Active Low Decoder)



Implementation : (Active Low Decoder)



Implementation : (Active Low Decoder)



Active low Decoder ≡ NAND Decoder

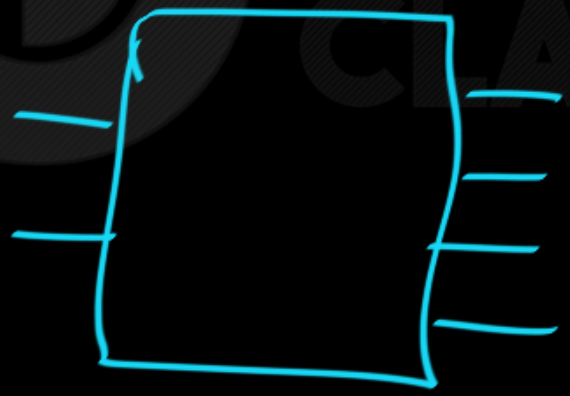
Notation:

Active low decoder

Active High Decoder ≡ by Default Decoder

≡ "AND" implementation of Decoder

Notation:



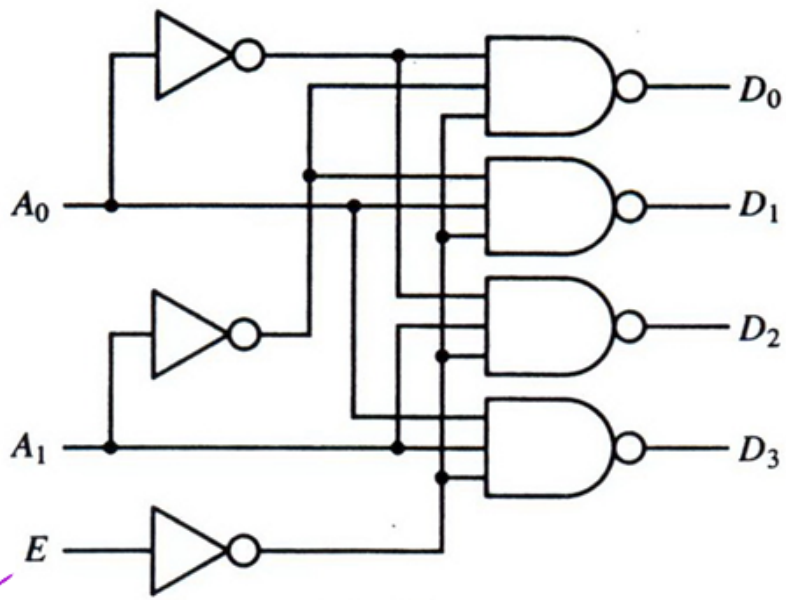
NAND Gate Decoder

- Some decoders are constructed with NAND instead of AND gates.
- The decoder is enabled when E is equal to 0.
- As indicated by the truth table, only one output is equal to 0 at any give time; the other three outputs are equal to 1.

NAND Gate Decoder

- The output whose value is equal to 0 represents the equivalent binary number in inputs A_1 and A_0.
- The circuit is disabled when E is equal to 1, regardless of the values of the other two inputs.

2-to-4-Line Decoder with NAND Gates



(a) Logic diagram

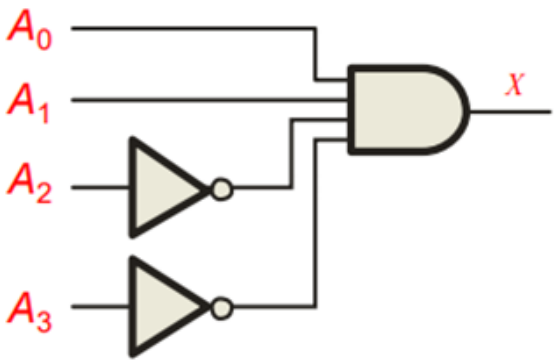
E	A_1	A_0	D_0	D_1	D_2	D_3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	×	×	1	1	1	1

(b) Truth table

Active low Enable

Q:

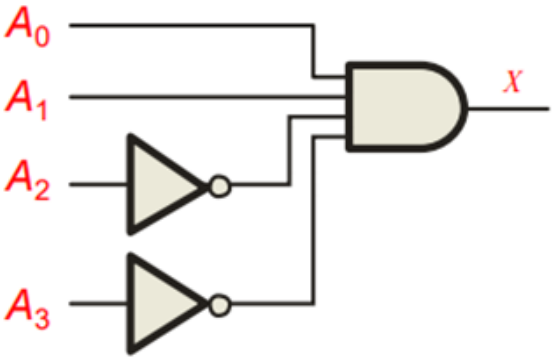
Which binary number is Detected by this Circuit?



(A_0 is the LSB and A_3 is the MSB)

Q:

Which binary number is Detected by this Circuit?



"any number other than"

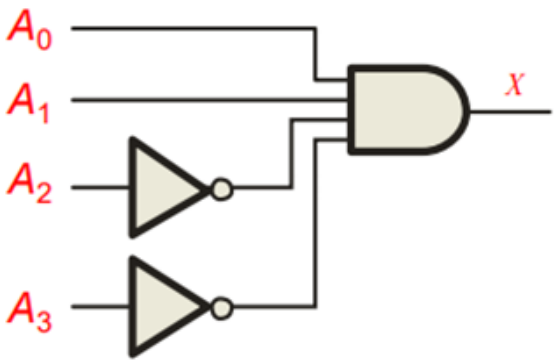
A_3	A_2	A_1	A_0
0	0	1	1

$x = 0$

(A_0 is the LSB and A_3 is the MSB)

Q:

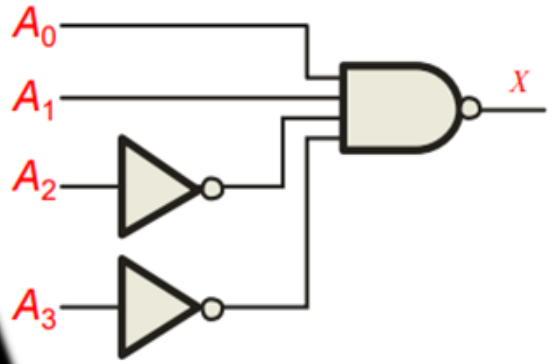
Which binary number is Detected by this Circuit? ⇒ 0011



for $A_3A_2A_1A_0$ = 0 0 1 1 ⇒ $X = 1$

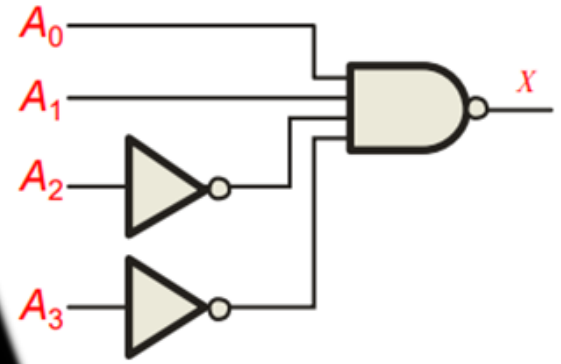
(A_0 is the LSB and A_3 is the MSB)

Which binary number is Detected by this Circuit?



(A_0 is the LSB and A_3 is the MSB)

Which binary number is Detected by this Circuit?

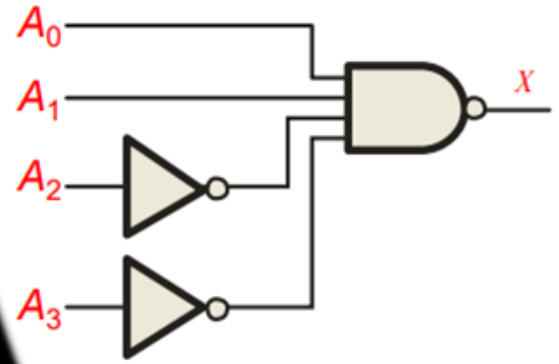


Input: $q_3\ q_2\ q_1\ q_0$

0 0 1 1 $\Rightarrow X = 0$

(A_0 is the LSB and A_3 is the MSB)

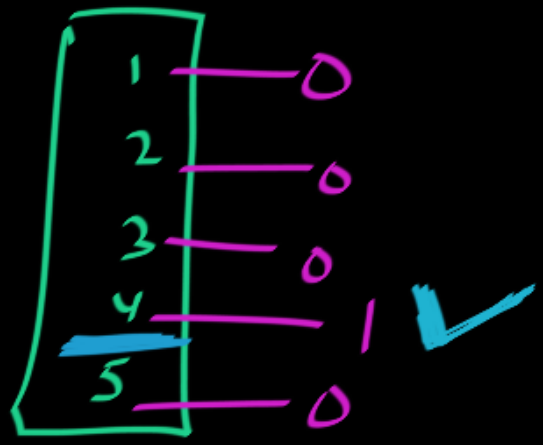
Which binary number is Detected by this Circuit?



for all other inputs $\Rightarrow$ $X = 1$

(A_0 is the LSB and A_3 is the MSB)

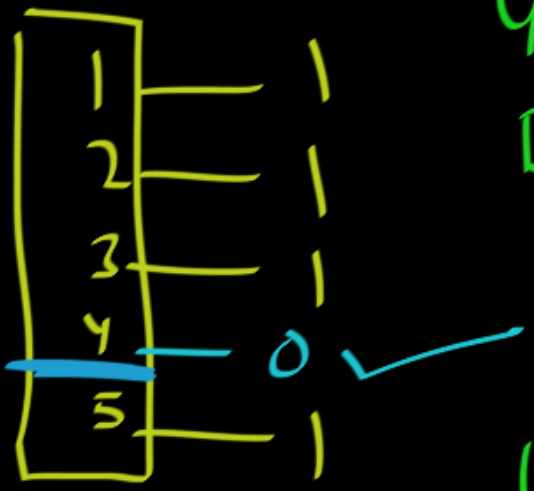
Home 1



button

Y is Detected in Active High manner

Home 2

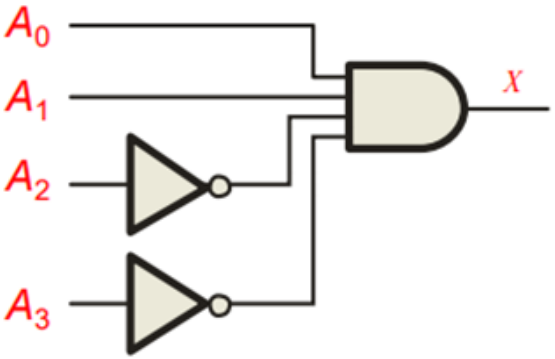


Y is Detected in Active low manner

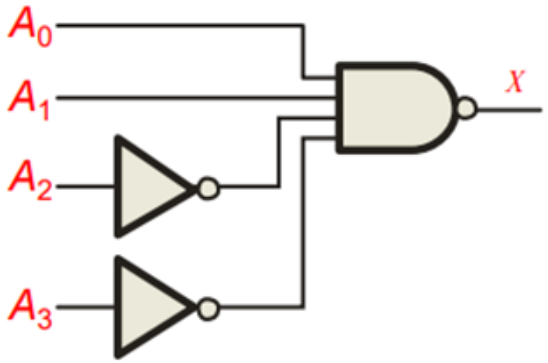
Simple Decoder

A **decoder** is a logic circuit that detects the presence of a specific combination of bits at its input.

Two simple decoders that **detect the presence** of the binary code 0011 are shown below. The first has an active HIGH output; the second has an active LOW output.



Active HIGH decoder for 0011

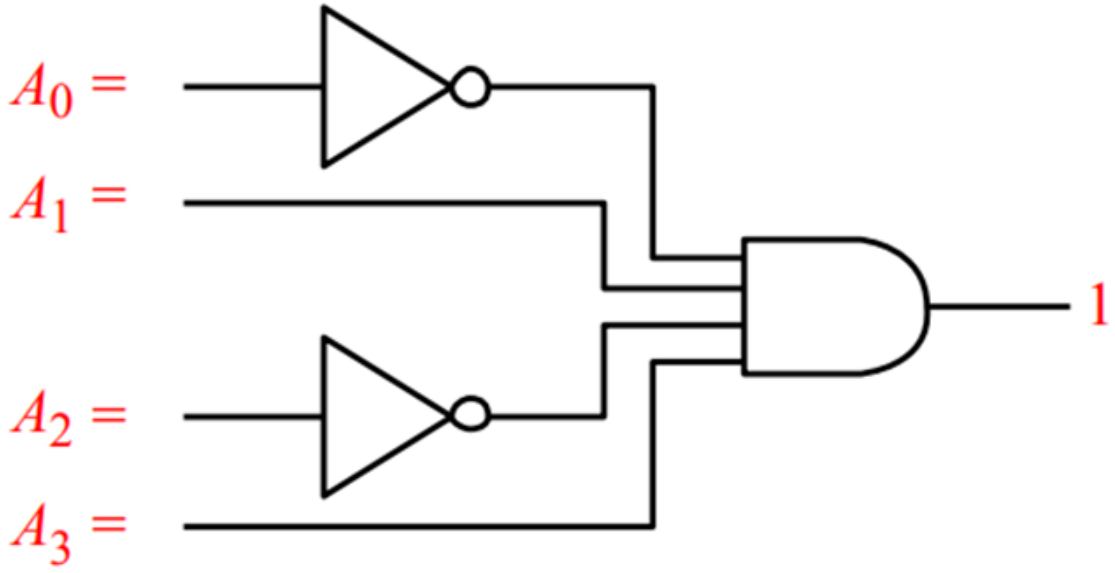


Active LOW decoder for 0011

(A_0 is the LSB and A_3 is the MSB)

Exercise

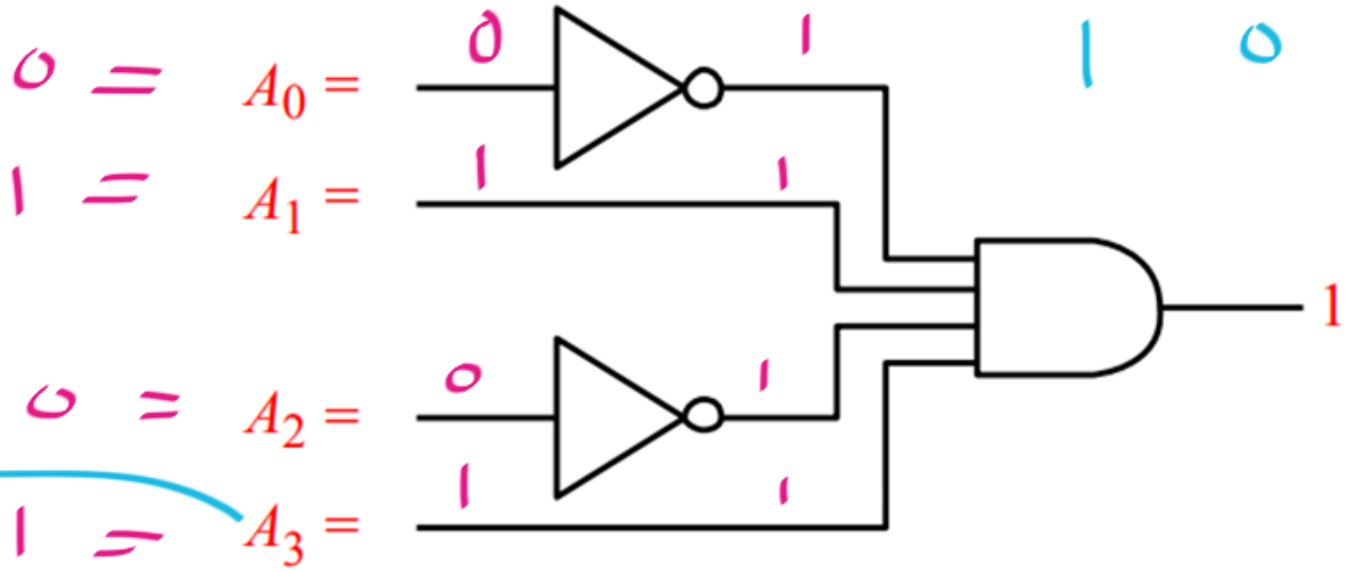
Assume the output of the decoder shown below is a logic 1. What are the inputs to the decoder?



Exercise

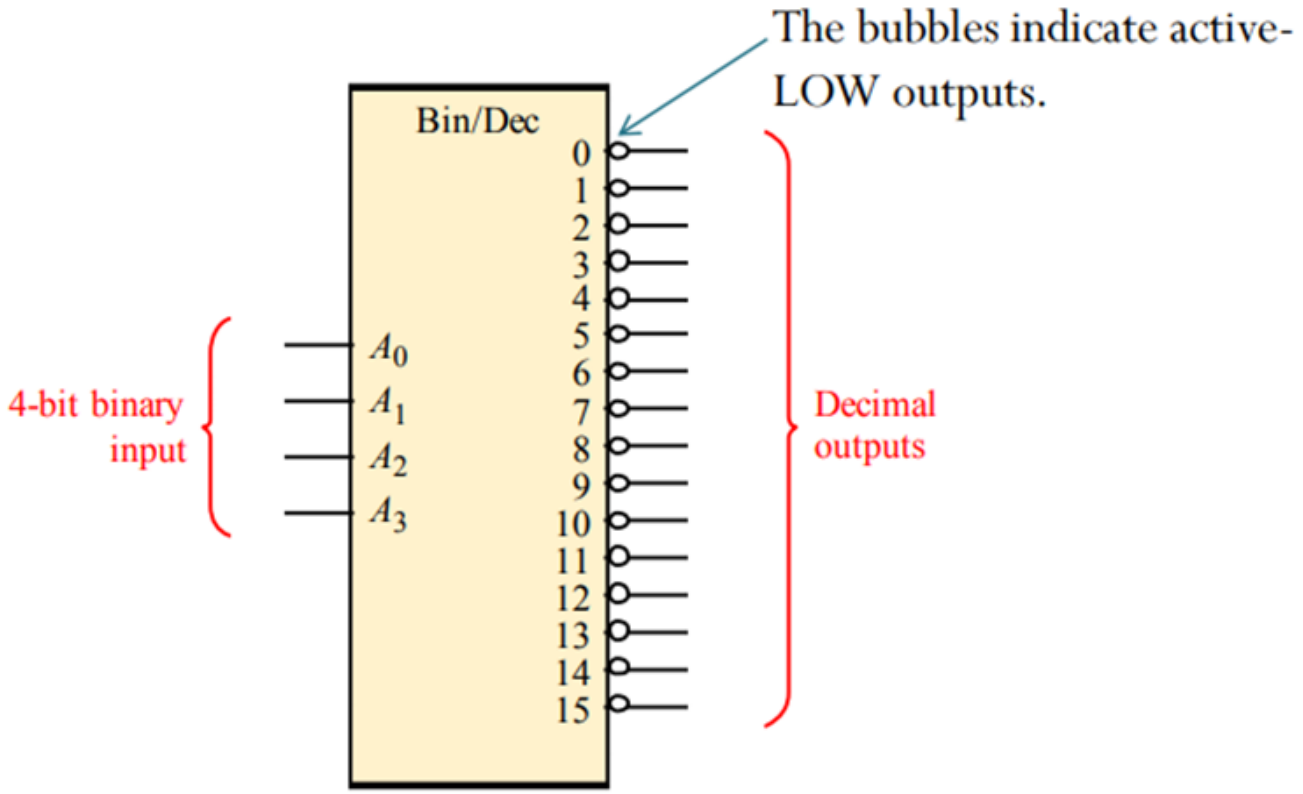
Assume the output of the decoder shown below is a logic 1. What are the inputs to the decoder?

A_3 A_2 A_1 A_0
1 0 1 0



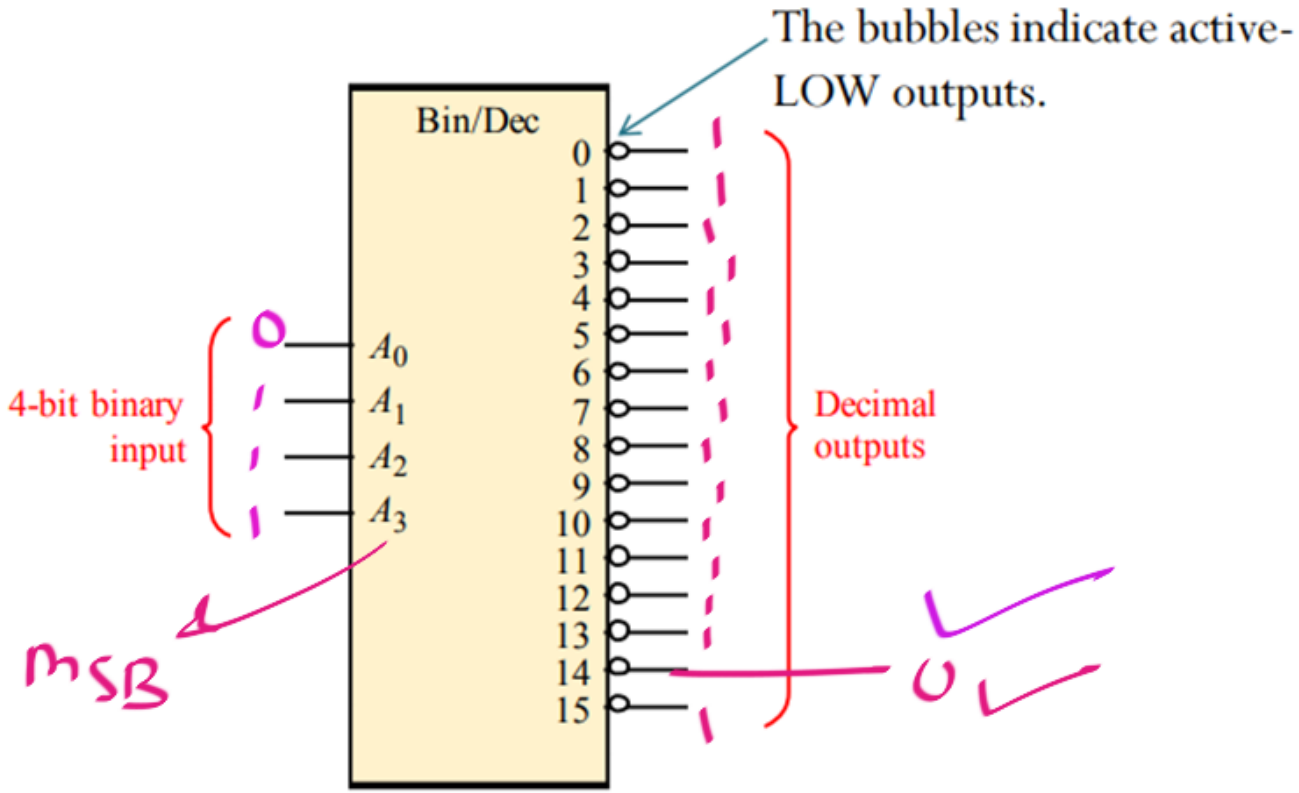
Binary-to-Decimal Decoder

The binary-to-decimal decoder shown here has 16 outputs – one for each combination of binary inputs.



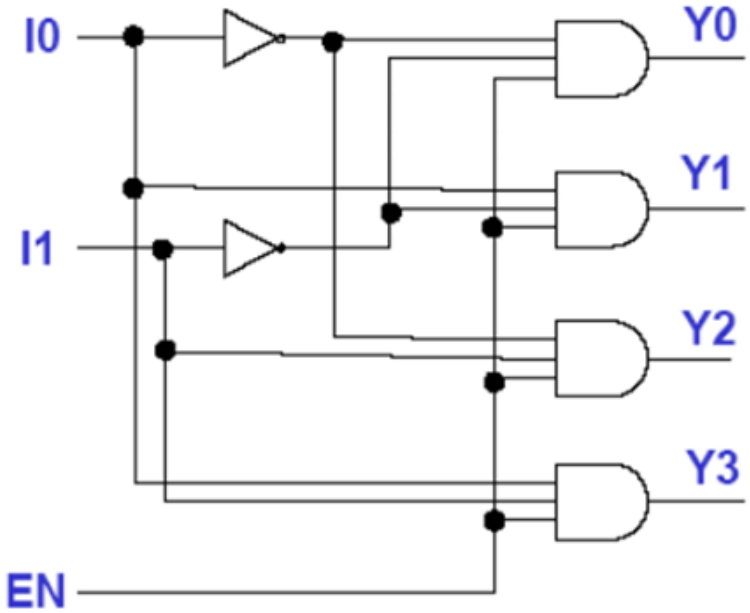
Binary-to-Decimal Decoder

The binary-to-decimal decoder shown here has 16 outputs – one for each combination of binary inputs.



2:4 decoder

2-to-4 line decoder with enable input

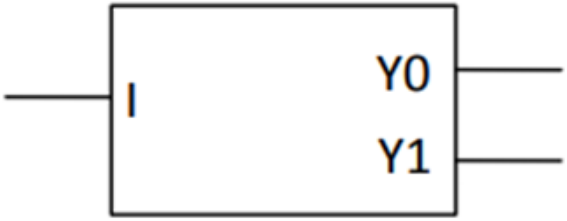


Inputs			Outputs			
EN	I1	I0	Y3	Y2	Y1	Y0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Exercise

Find the truth table of the 1-to-2 line decoder below.

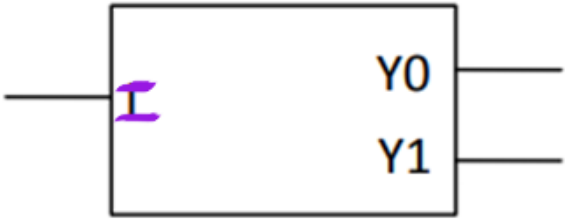
Then, implement the 1-to-2 line decoder.



Exercise

Find the truth table of the 1-to-2 line decoder below.

Then, implement the 1-to-2 line decoder.



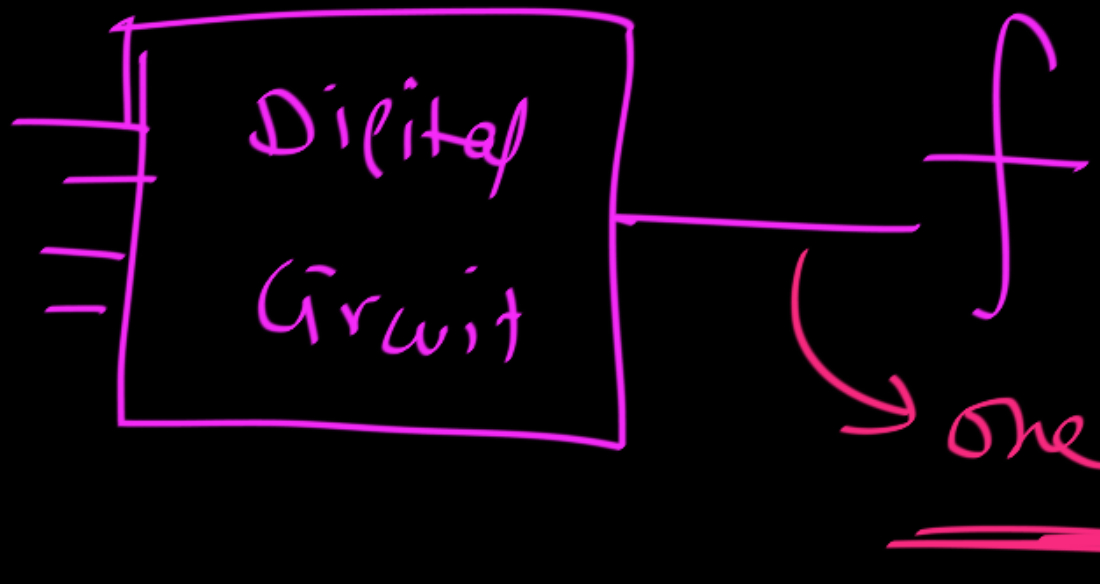
I	Y_0	Y_1
0	1	0
1	0	1

$Y_0 = \bar{I}$; $Y_1 = I$

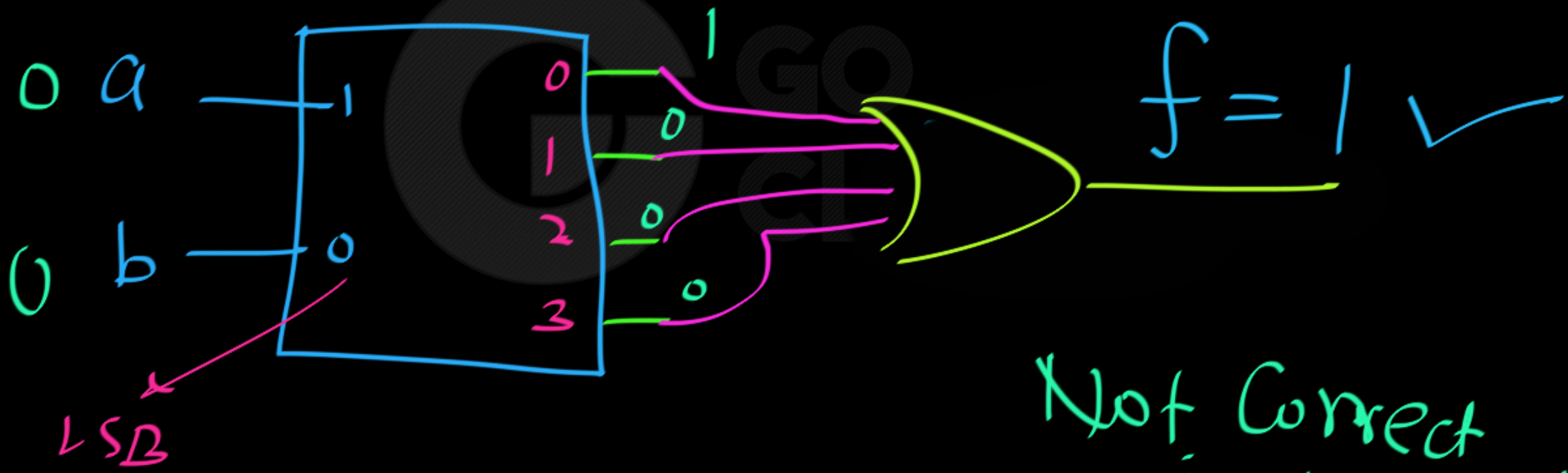
Next Topic

Implementation (Realization) of functions using Decoder

Implementation of f using any Digital Circuit ;

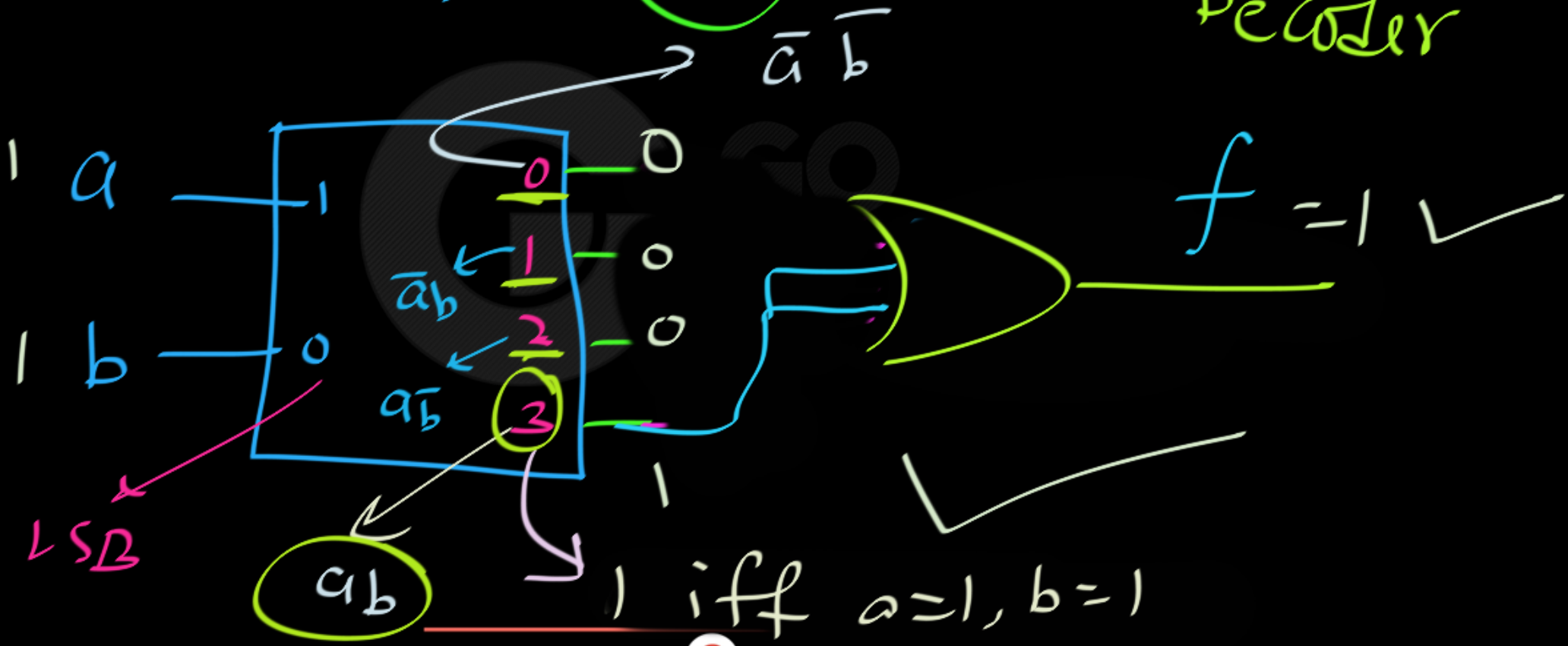


Ex: $f(a,b) = ab$ Using Active High Decoder

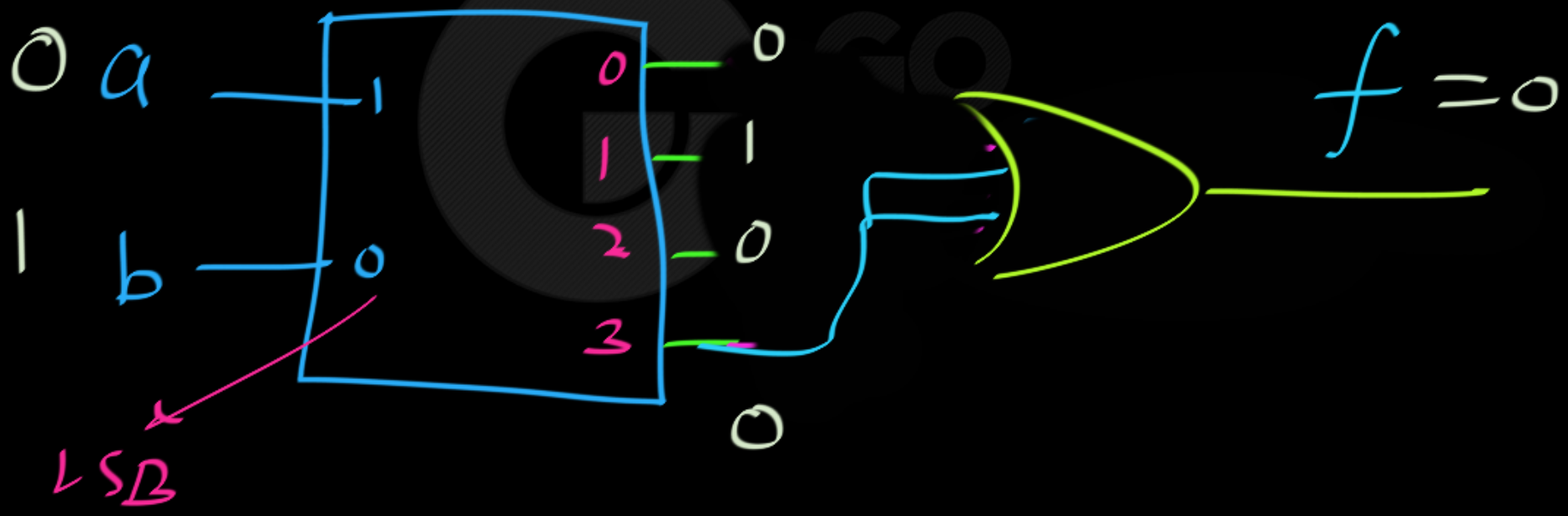


Not Correct implementation

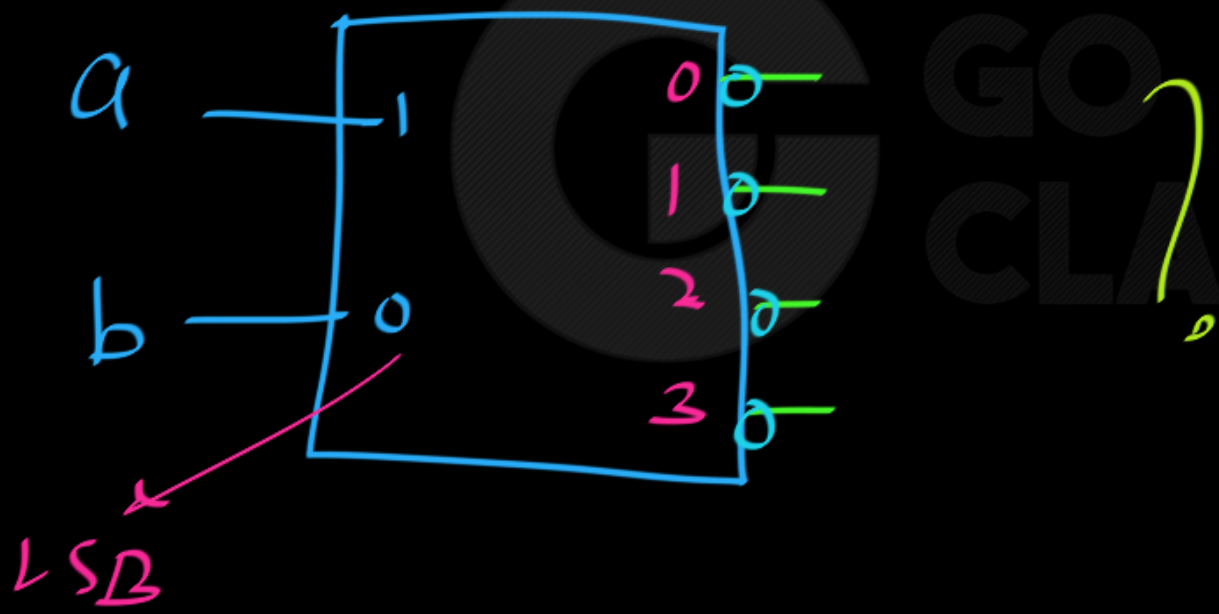
Eg: $f(a,b) = ab$ Using Active High Decoder



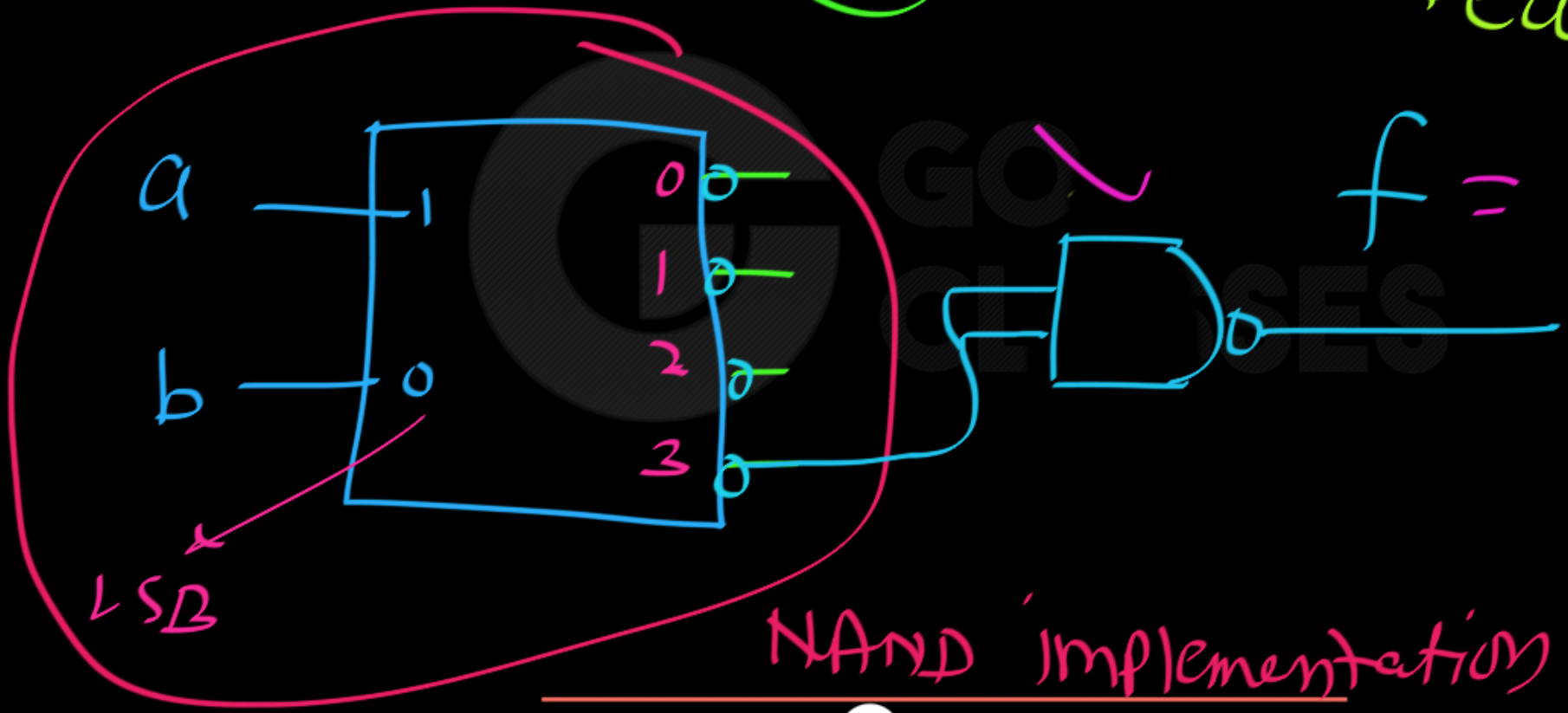
Ex: $f(a,b) = ab$ using Active High Decoder



Ex: $f(a,b) = ab$ using Active Low Decoder

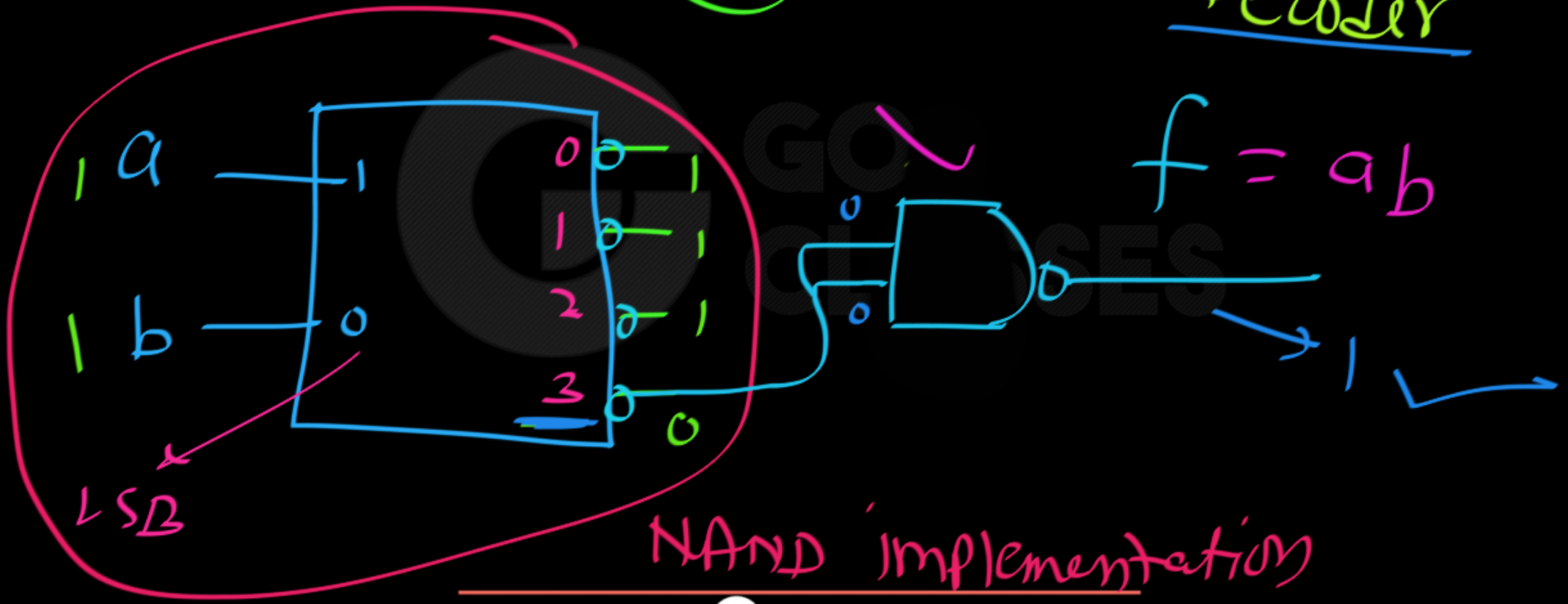


Ex: $f(a,b) = ab$ using Active Low Decoder



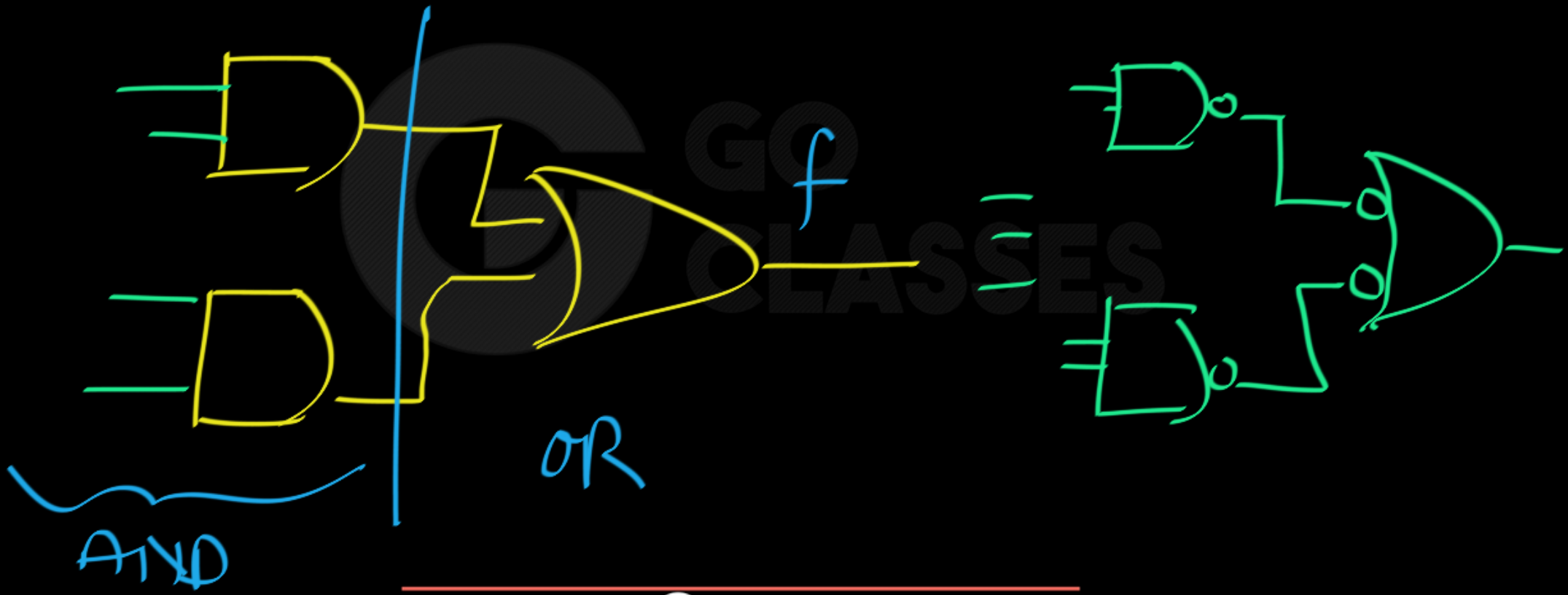
NAND implementation

Ex: $f(a,b) = \overline{ab}$ using Active Low Decoder

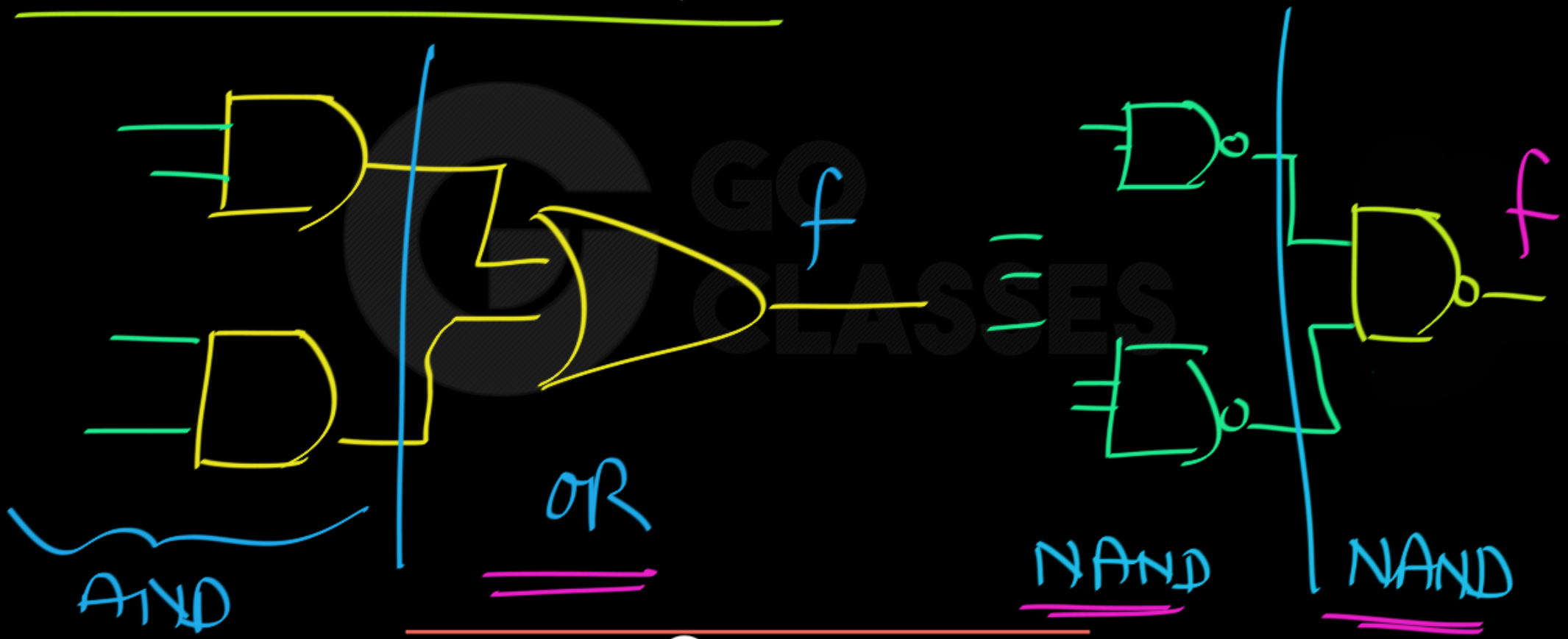


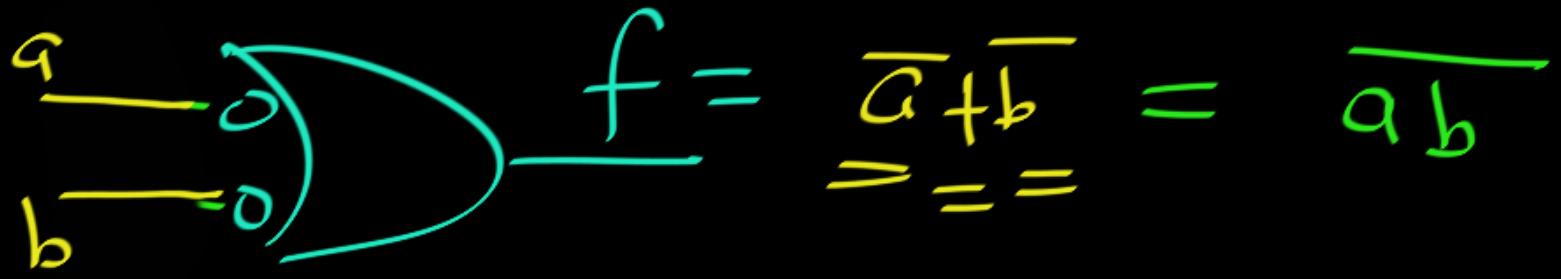
NAND implementation

Important Concepts:

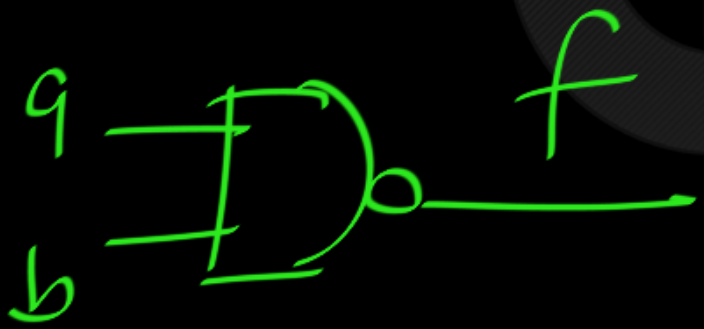


Important Concepts:

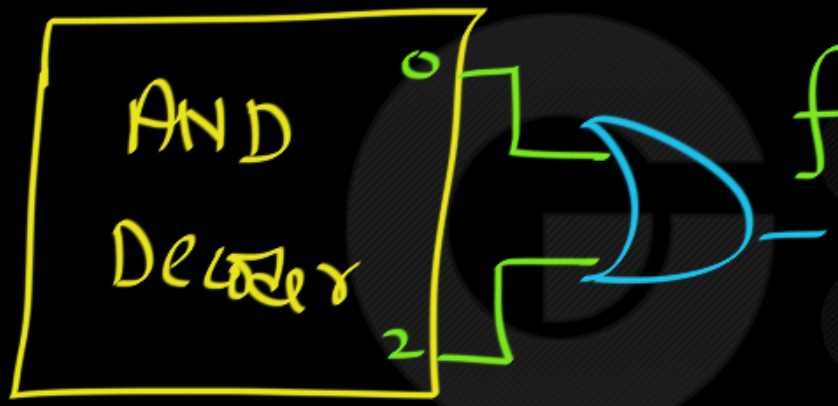


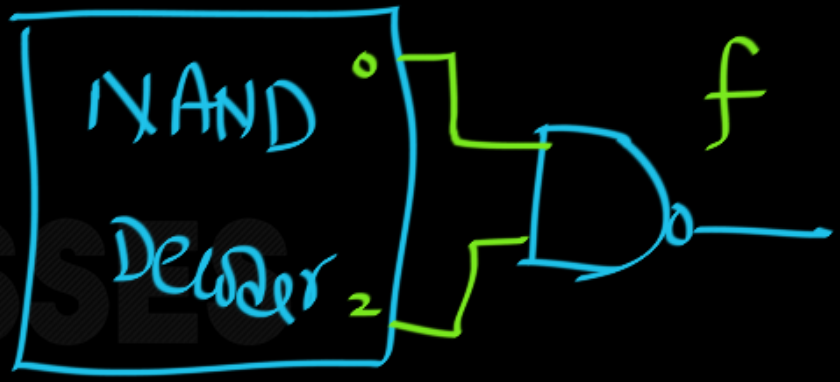


1 1 1

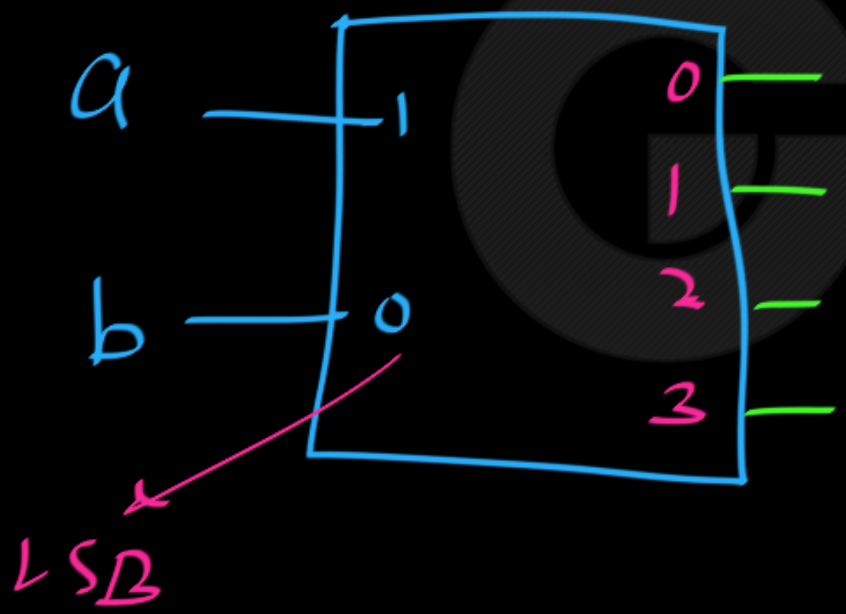


Remember:



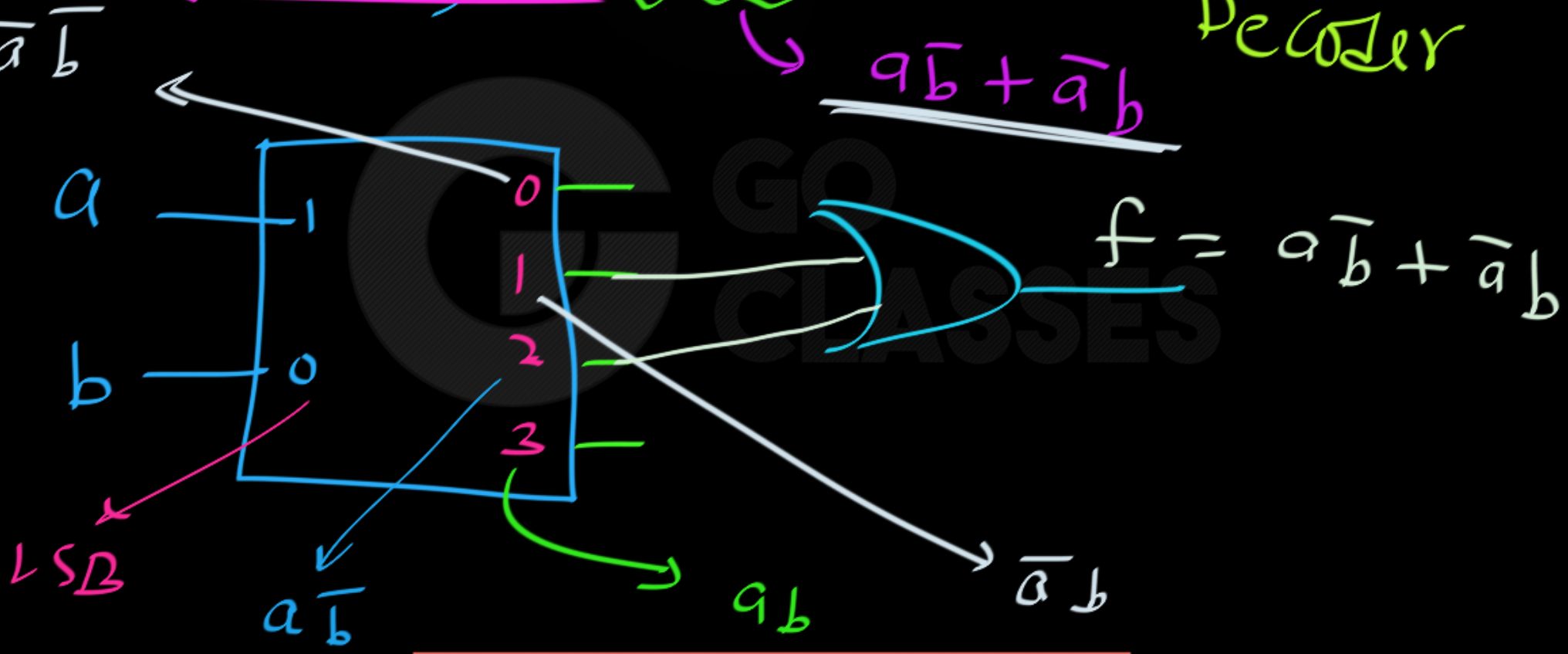


Ex: $f(a,b) = a \oplus b$ Using Active High Decoder



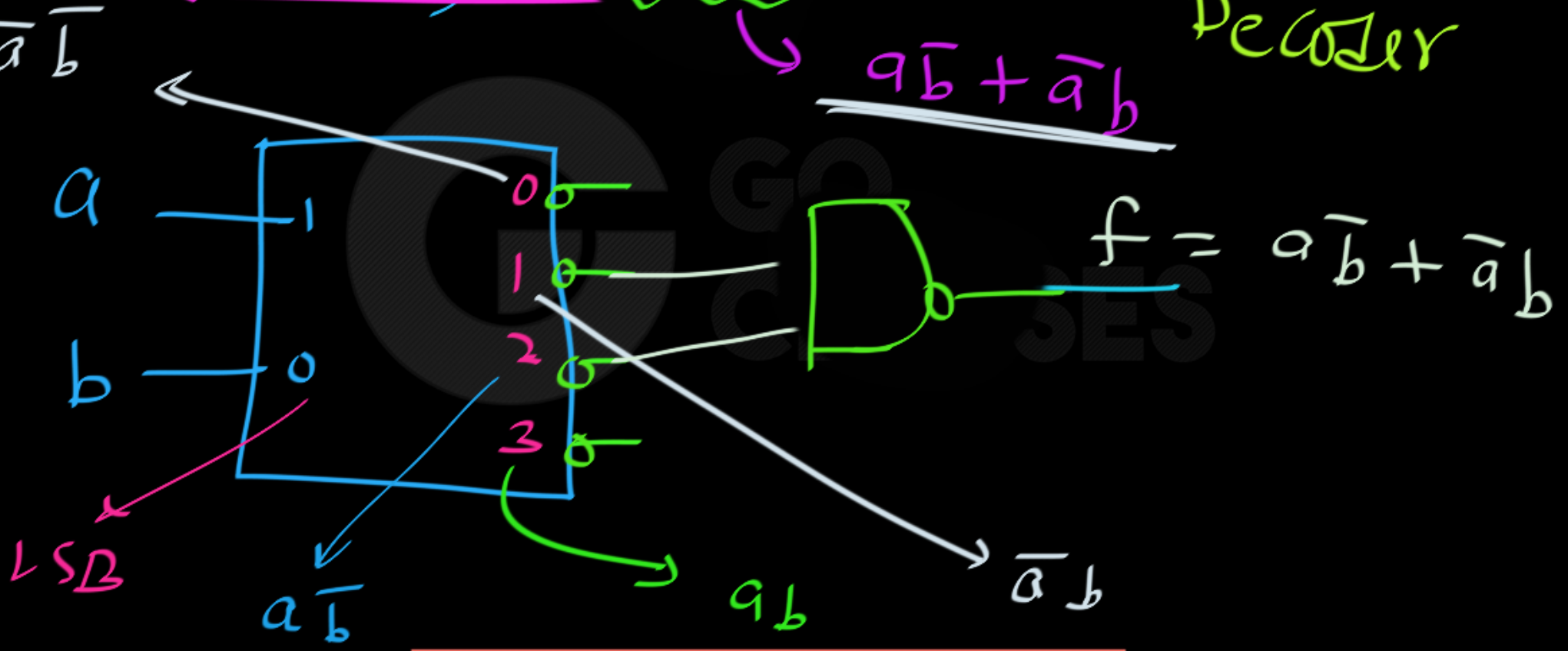
Ex: $f(a,b) = a \oplus b$ using Active High Decoder

$a \oplus b = a\bar{b} + \bar{a}b$



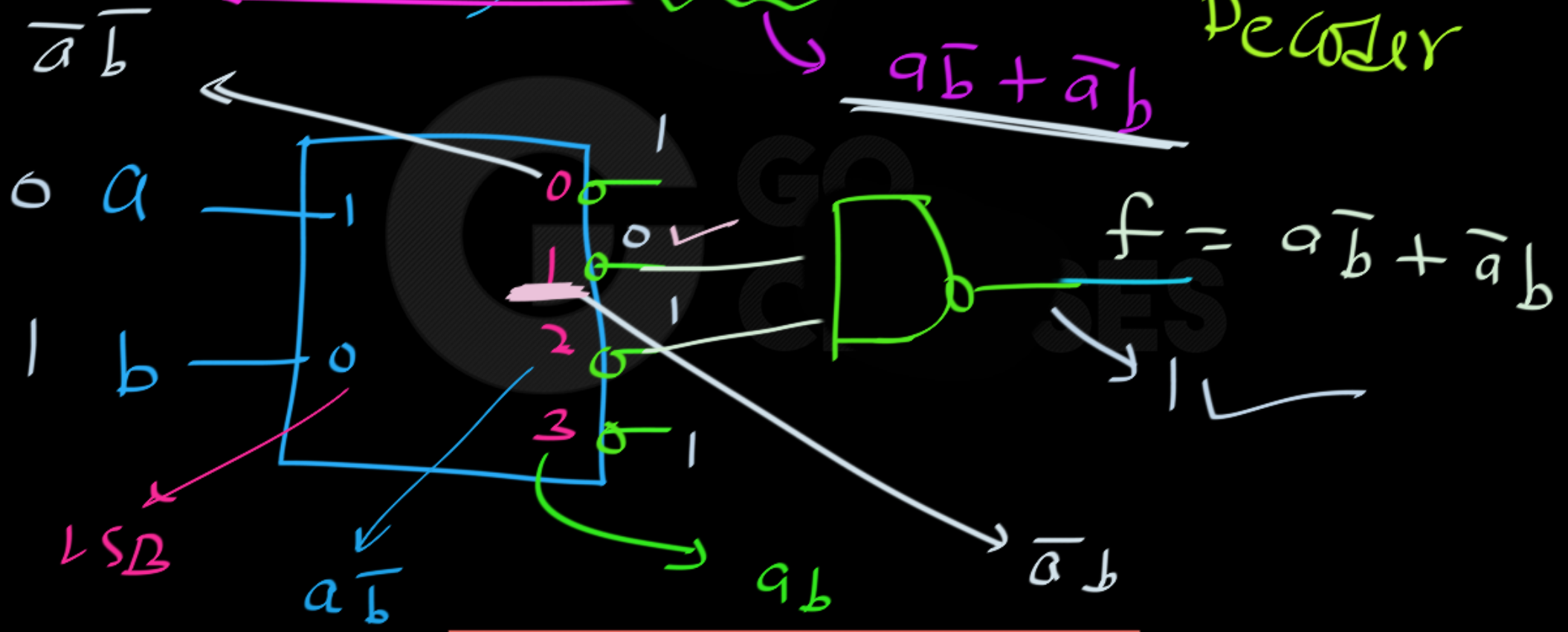
Ex: $f(a,b) = a \oplus b$ using Active Low Decoder

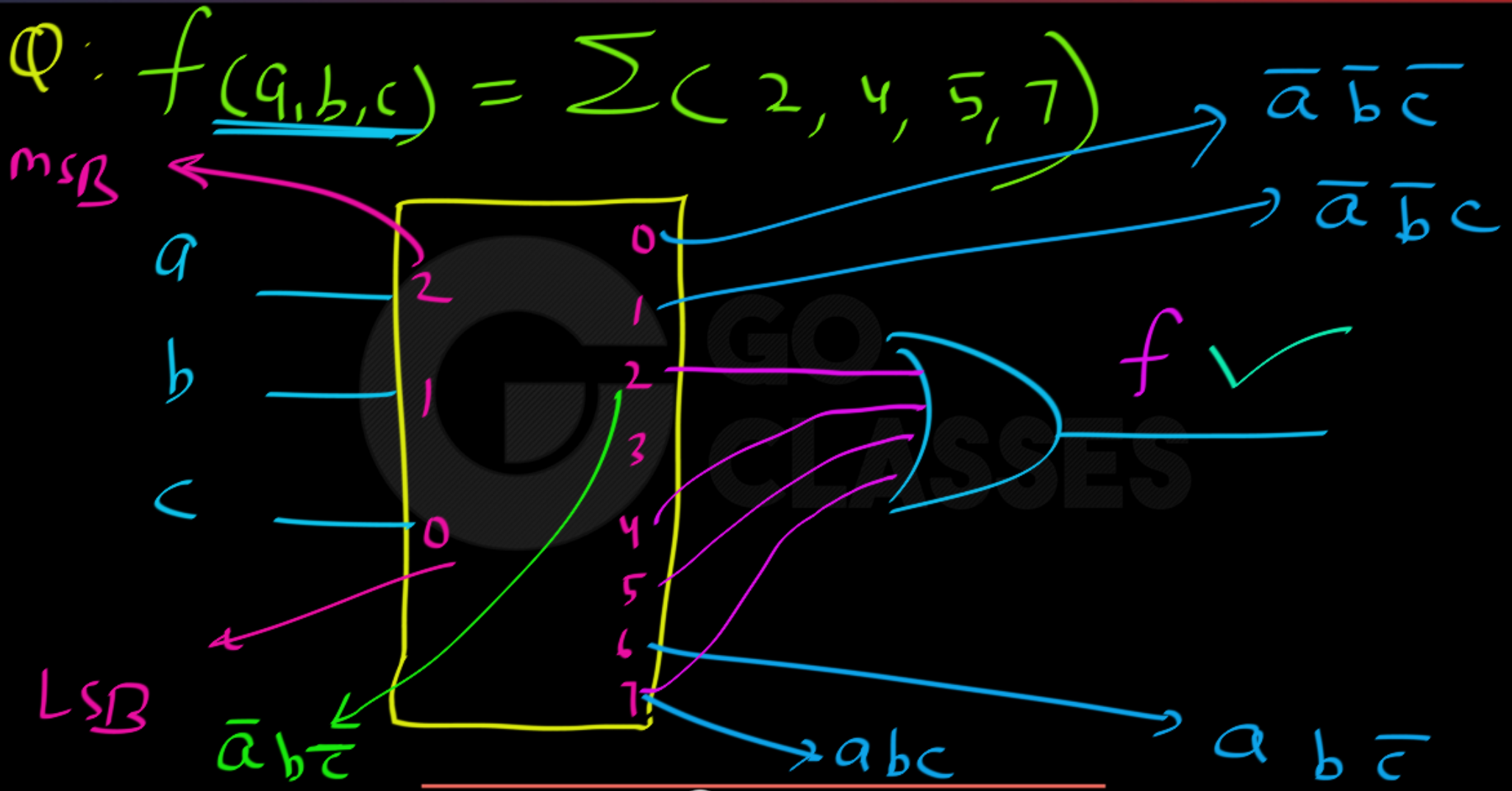
$a \oplus b = a\bar{b} + \bar{a}b$



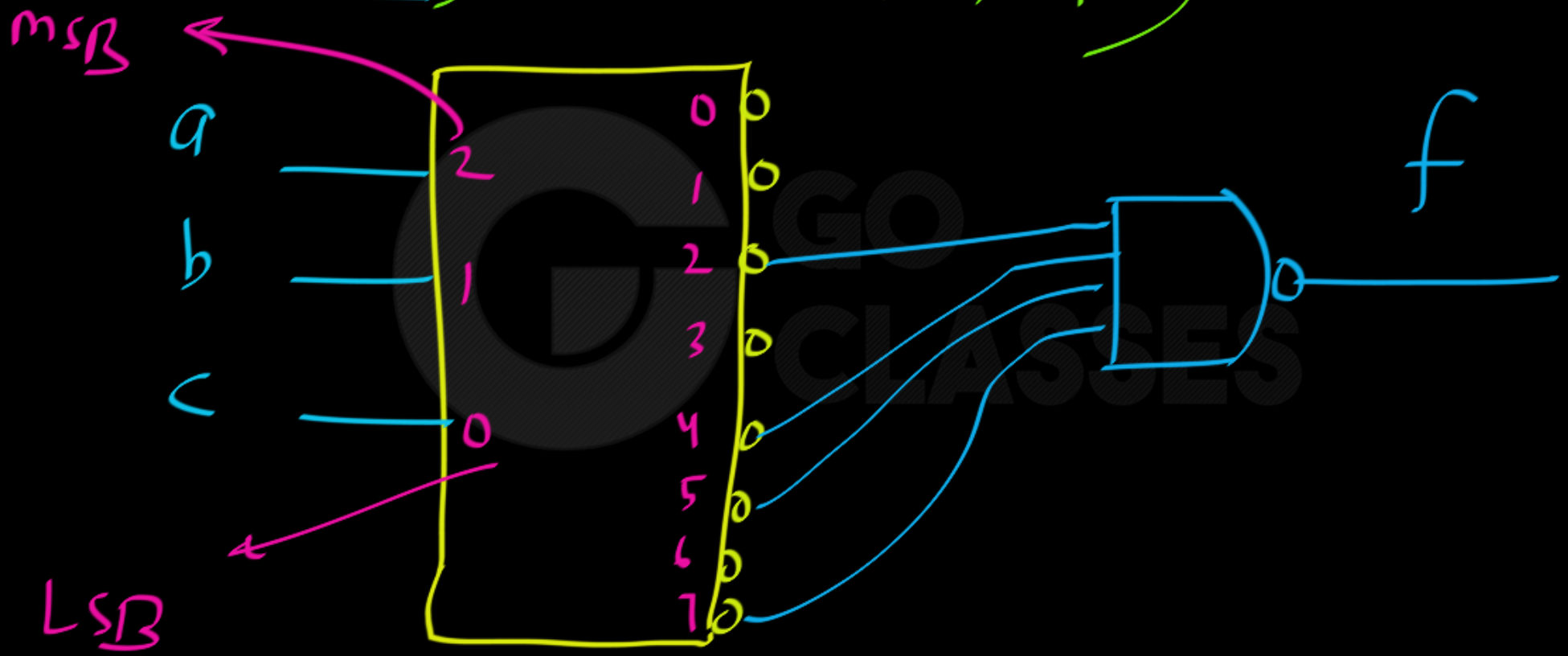
Ex: $f(a,b) = a \oplus b$ using Active Low Decoder

$a \oplus b = a\bar{b} + \bar{a}b$




Q: f(a,b,c) = Σ(2, 4, 5, 7)
MSB
a
b
c
LSB
2
1
0
0
1
2
3
4
5
6
7
f
$\bar{a}\bar{b}\bar{c}$
$\bar{a}\bar{b}c$
$\bar{a}b\bar{c}$
abc
$ab\bar{c}$

Q: $f(a,b,c) = \Sigma(2, 4, 5, 7)$



Summary:

Implementation of function on 3 Variables:

(3X8 Decoder + 1 OR gate)

Summary:

Implementation of function on 3 Variables:

(3X8 Decoder + 1 NAND gate)

Active low

Summary:

Implementation of function on n Variables:

$(n \times 2^n$ Decoder$) + (1$ OR gate$)$

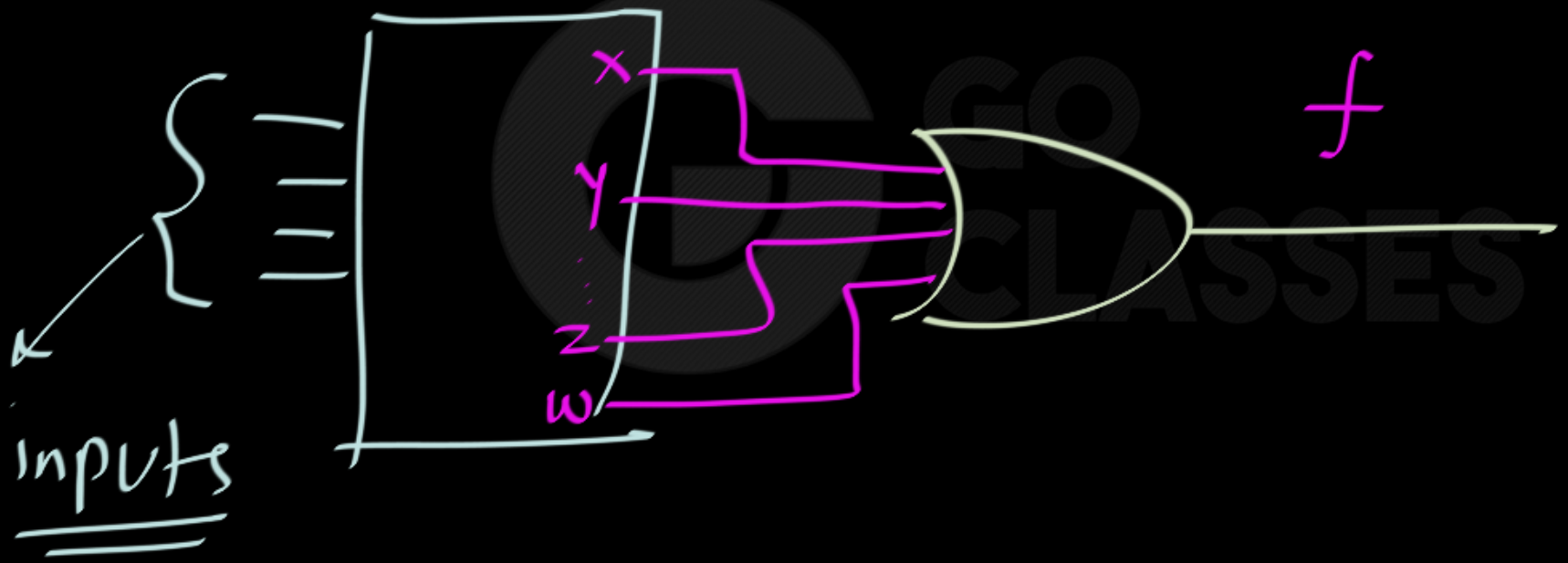
Summary:

Implementation of function on n Variables:

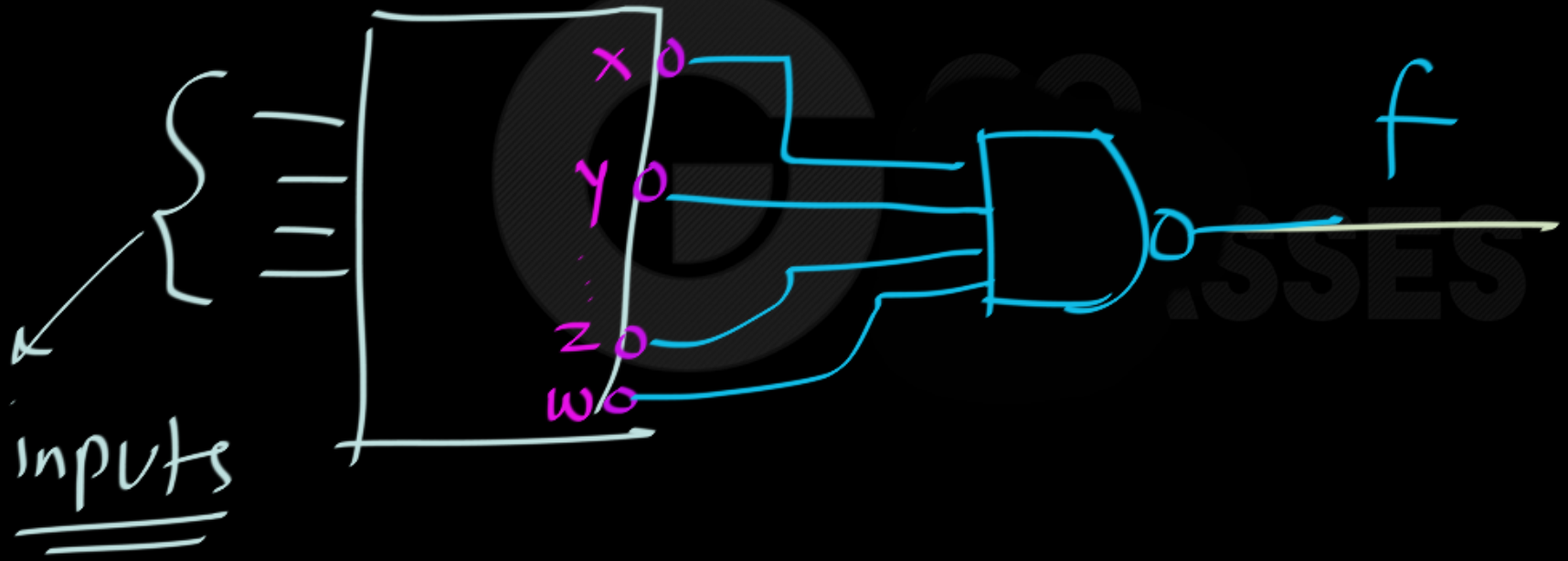
$(n \times 2^n$ Decoder$) + ($1 NAND gate$)$

Active low

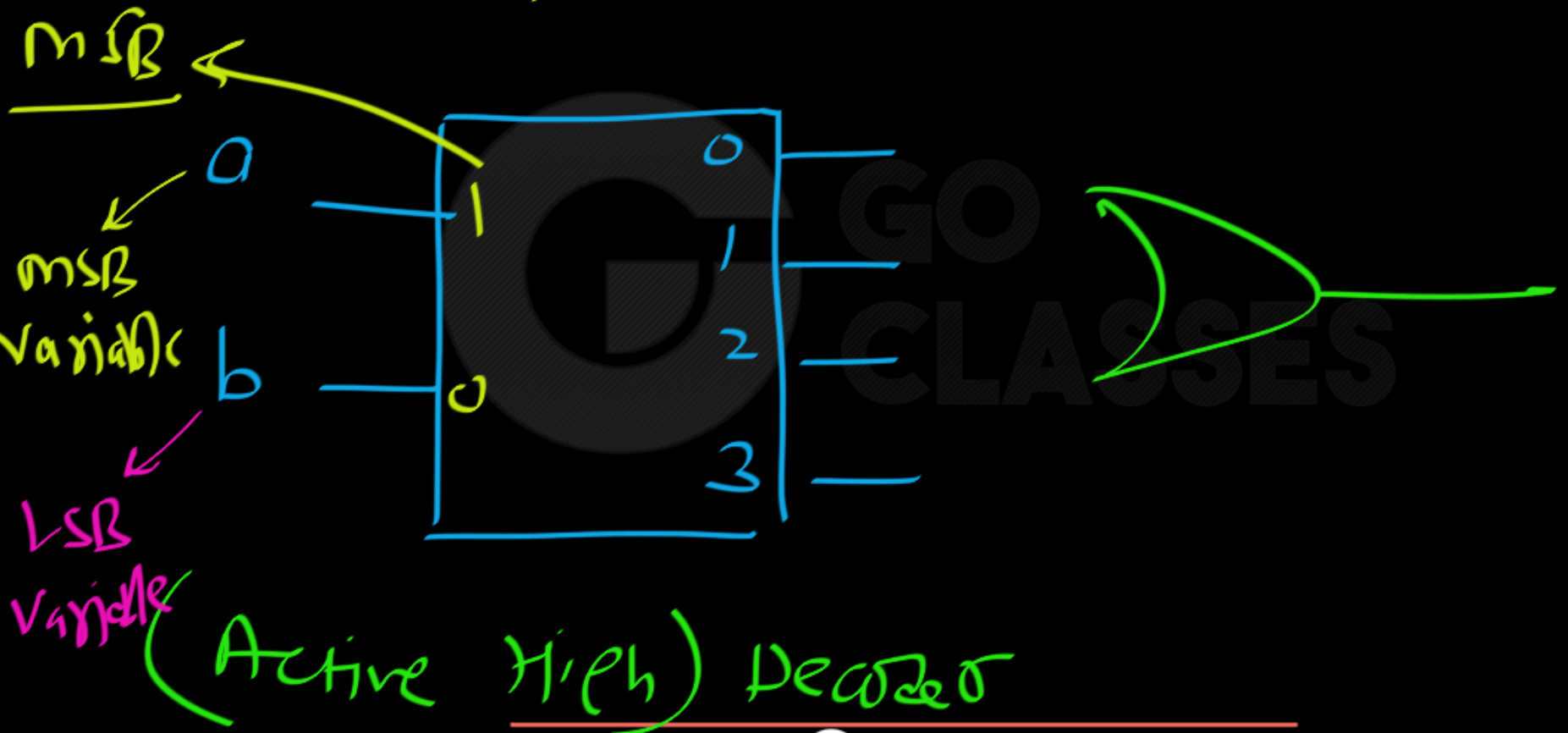
(Active High) Decoder : $f = \Sigma(X, Y, Z, W)$



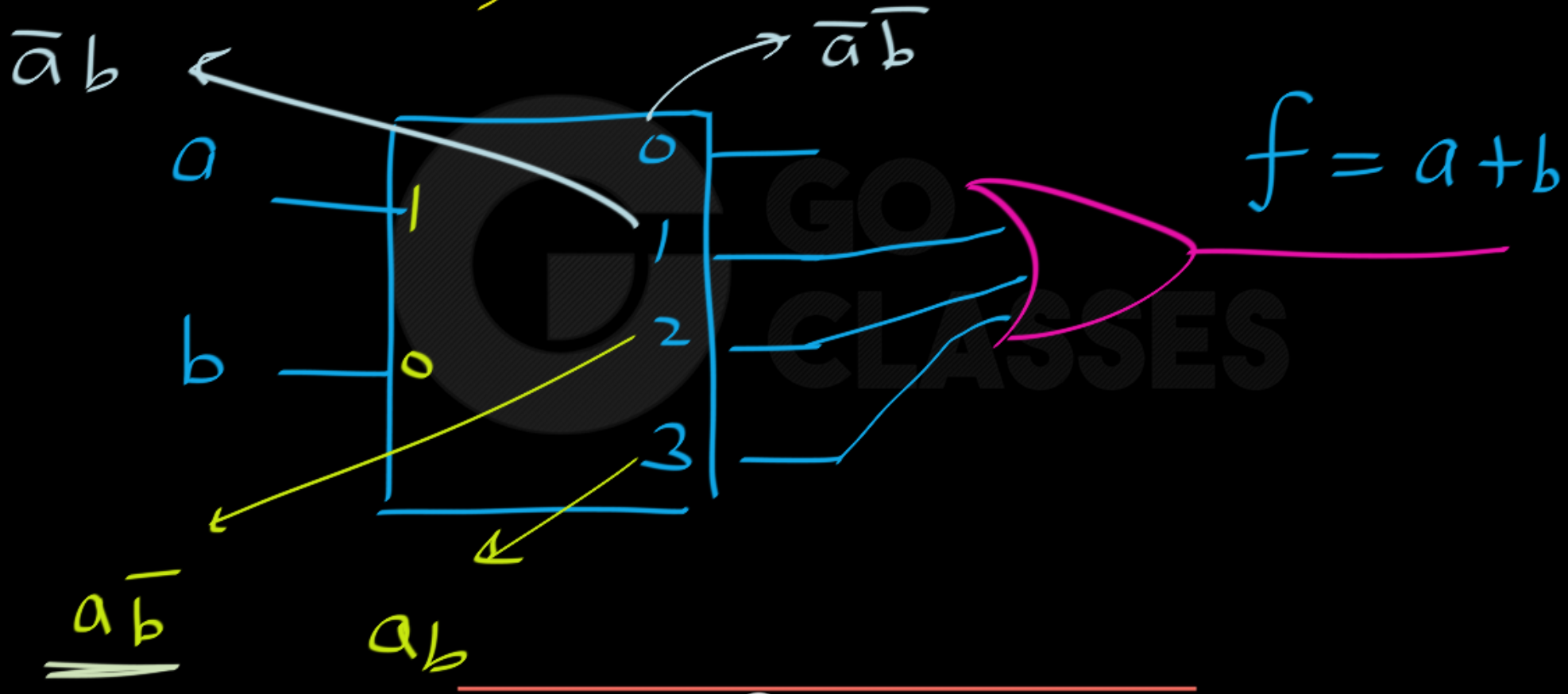
(Active Low) Decoder : $f = \Sigma(X, Y, Z, W)$



Ex: $f(a,b) = a + b \Rightarrow$ 2 Variables



Ex: $f(a,b) = a + b \Rightarrow$ 2 Variables



Ex: $f(a,b) = a + b \Rightarrow$ 2 Variables