Multiplexer with Enable Input

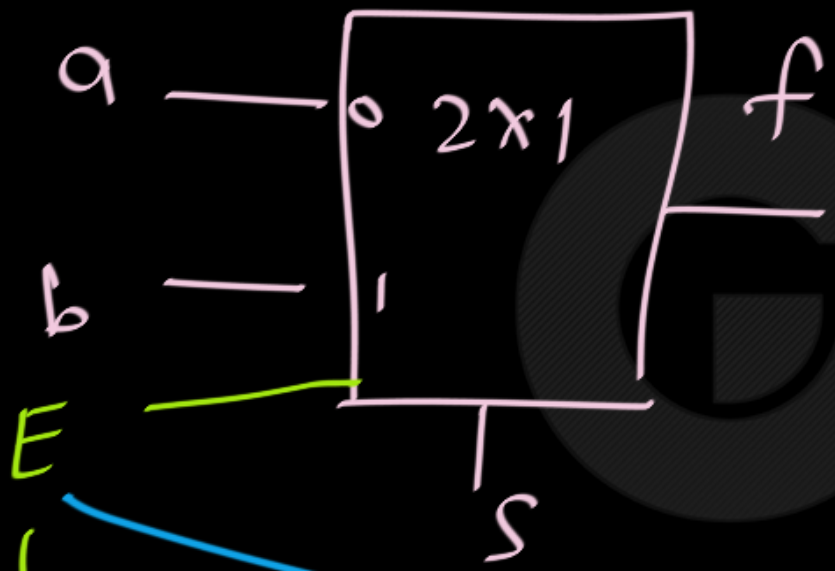

CPU :
mux
Adder
Decoder
Control unit
mux
brain of cpu
Demux
X
ALU
"Enable input"

Enable 'input' : like on/off switch

- ON : Circuit works nomally
- Off : Circuit 'idle' ($o/p = 0$)

Next Topic:

Mux with Enable Input



E	S	f(E,a,b,S)	
0	X	0	} mux idle.
1	0	a	}
1	1	b	} → mux working normally

Enable input → ON/off button

Analogy:

Safety

{ ON
  Off

Gun
Idle

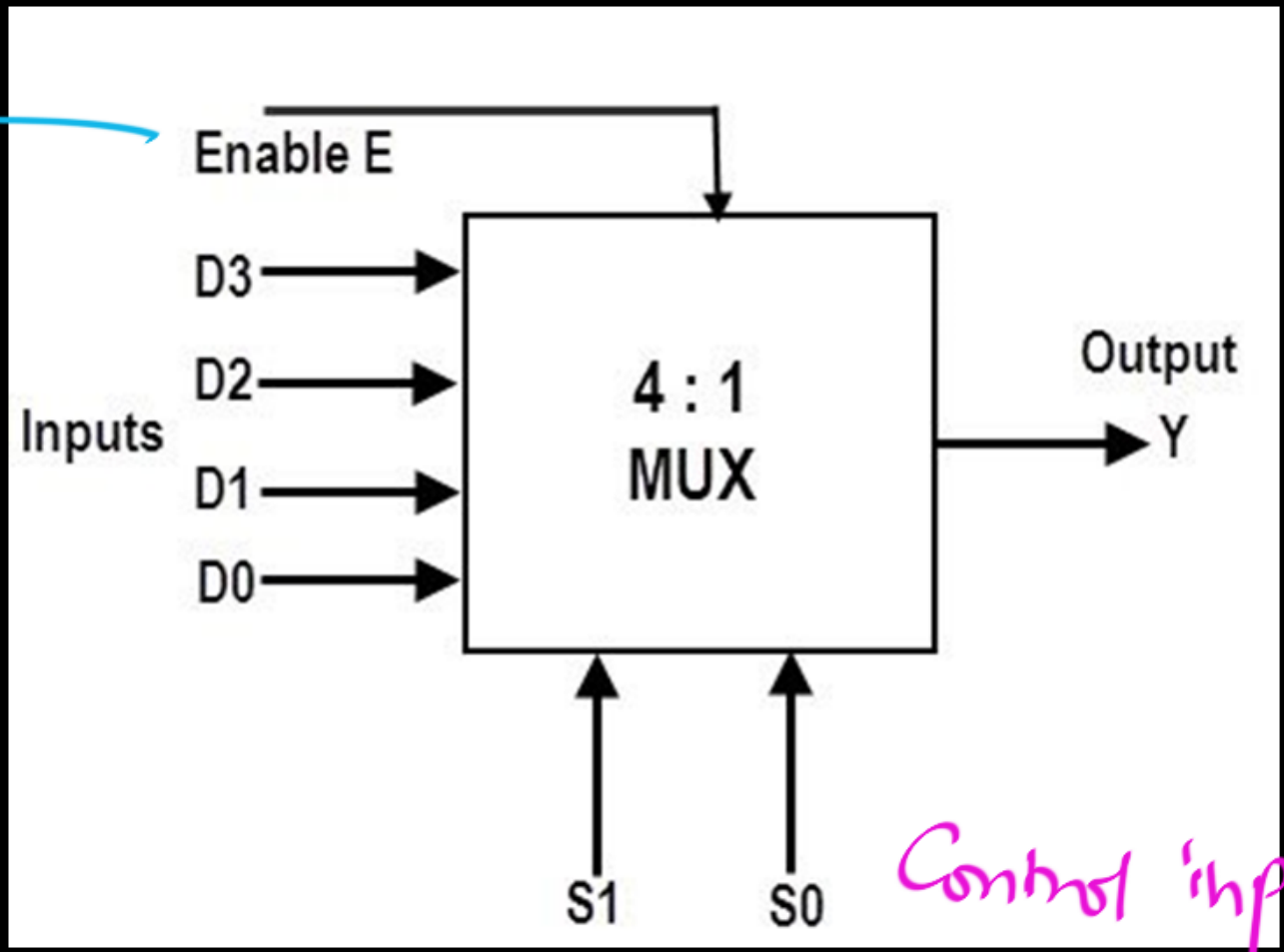
Ex: Power button on Laptop

{ Off : Laptop Idle
  ON : " Works

Gun
Idle
work (Thai Thai)

Enable input

Data



E	S_1	S_0	$Y(E, S_1, S_0, D_3, D_2, D_1, D_0)$
0	X	X	0 } mux idle
1	0	0	D_0
1	0	1	D_1
1	1	0	D_2
1	1	1	D_3

mux working

Enable E

Active low Enable

E = 0 Circuit works

E = 1 " Idle

Active High Enable

E = 1 Circuit works

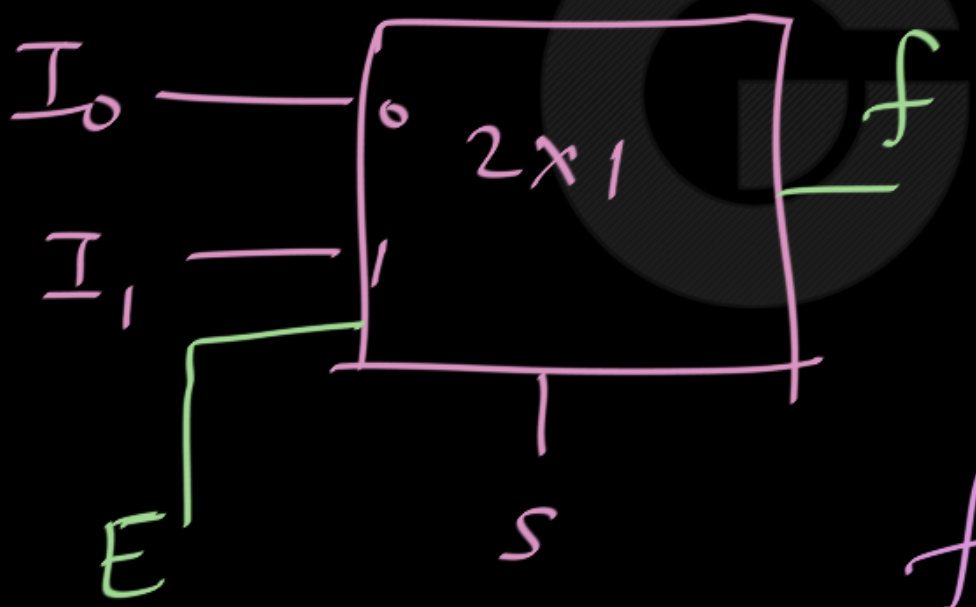
E = 0 " Idle.

Active low Enable :

Select Input S0	Enable Input E	Output Y
x	1	0
0	0	Y=D0
1	0	Y=D1

Active High Enable :

mux with



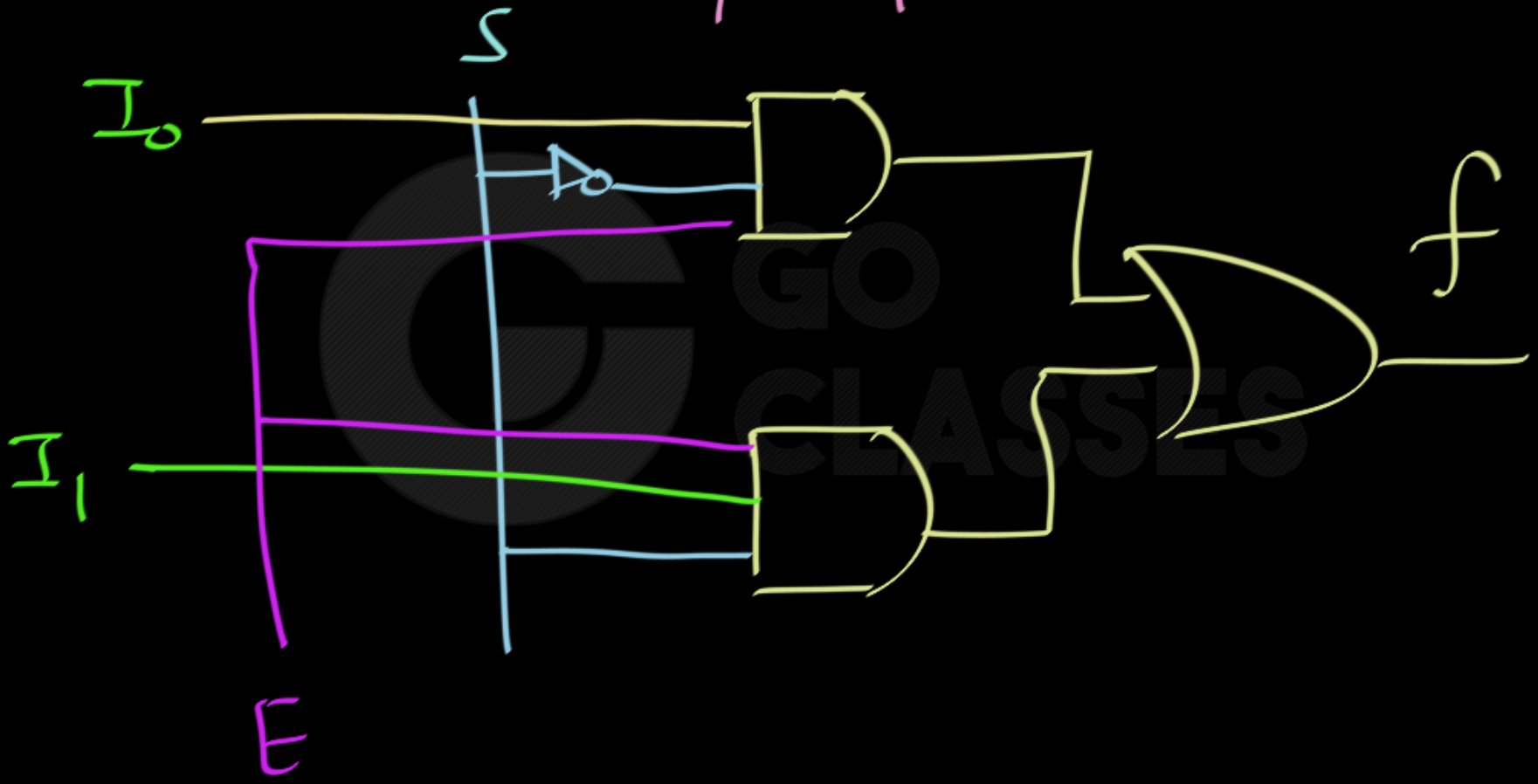
E	S	f(E, S, I_0, I_1)
0	X	0
1	0	I_0
1	1	I_1

$$f = E\bar{S}I_0 + ESI_1 = E(\bar{S}I_0 + SI_1)$$

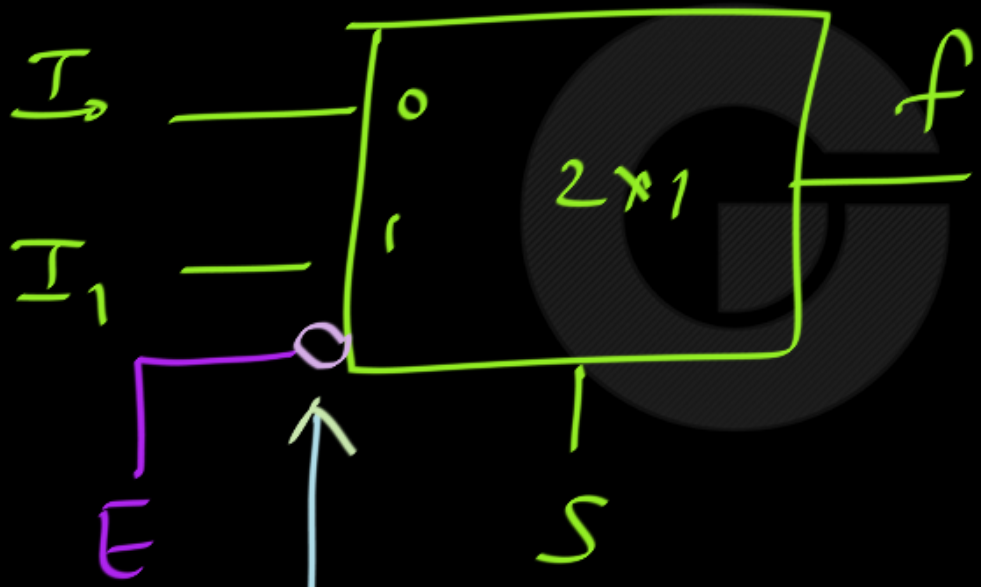
$$f_{(E, S, I_1, I_0)} = E.(\bar{S} I_0 + S I_1)$$

$$\begin{cases} E = 0 \Rightarrow f = 0 \\ E = 1 \Rightarrow f = \bar{S} I_0 + S I_1 \end{cases}$$

Implementation using logic gates:



mux with Active low Enable:



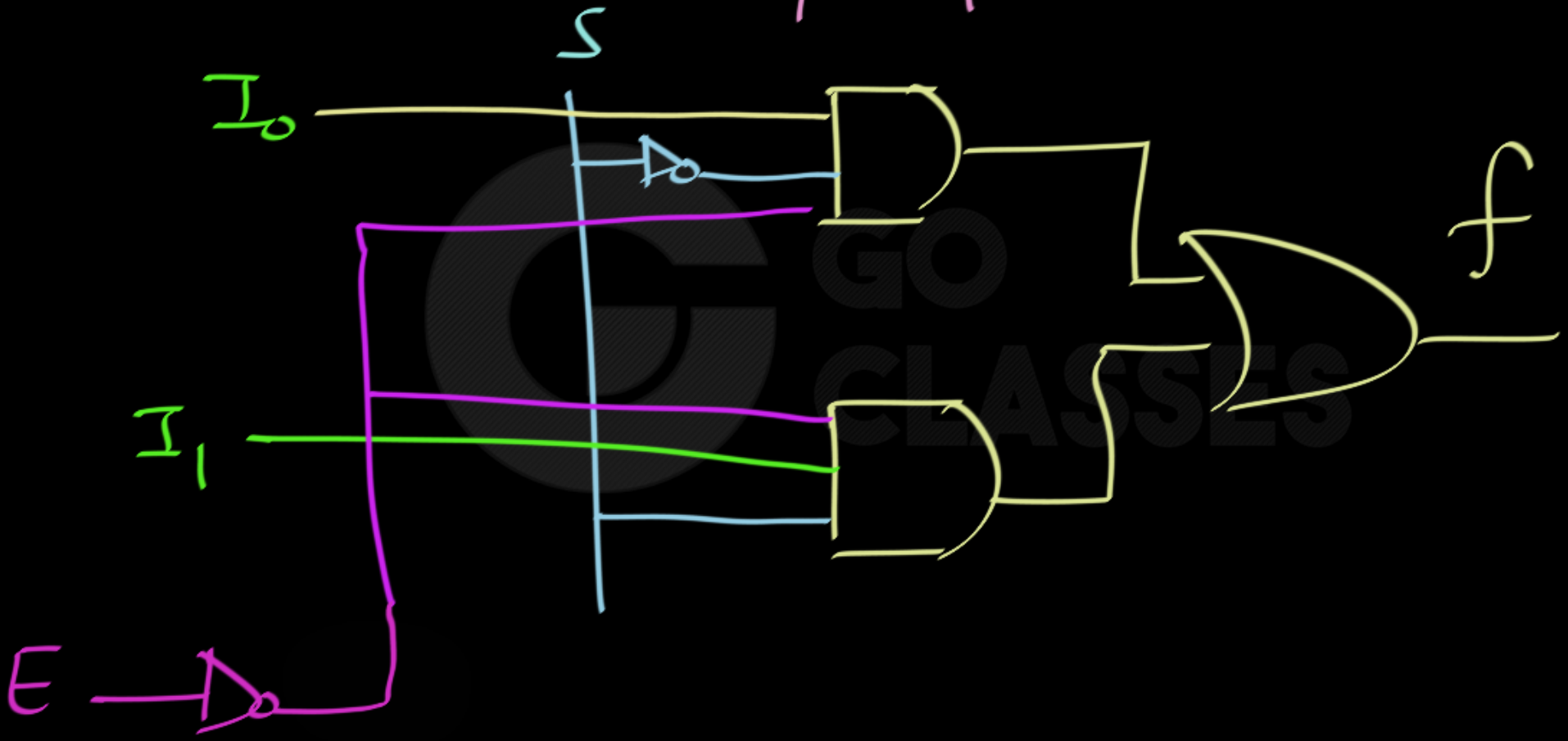
E	S	f
1	X	0
0	0	I_0
0	1	I_1

→ indicates that E is Active low Enable.

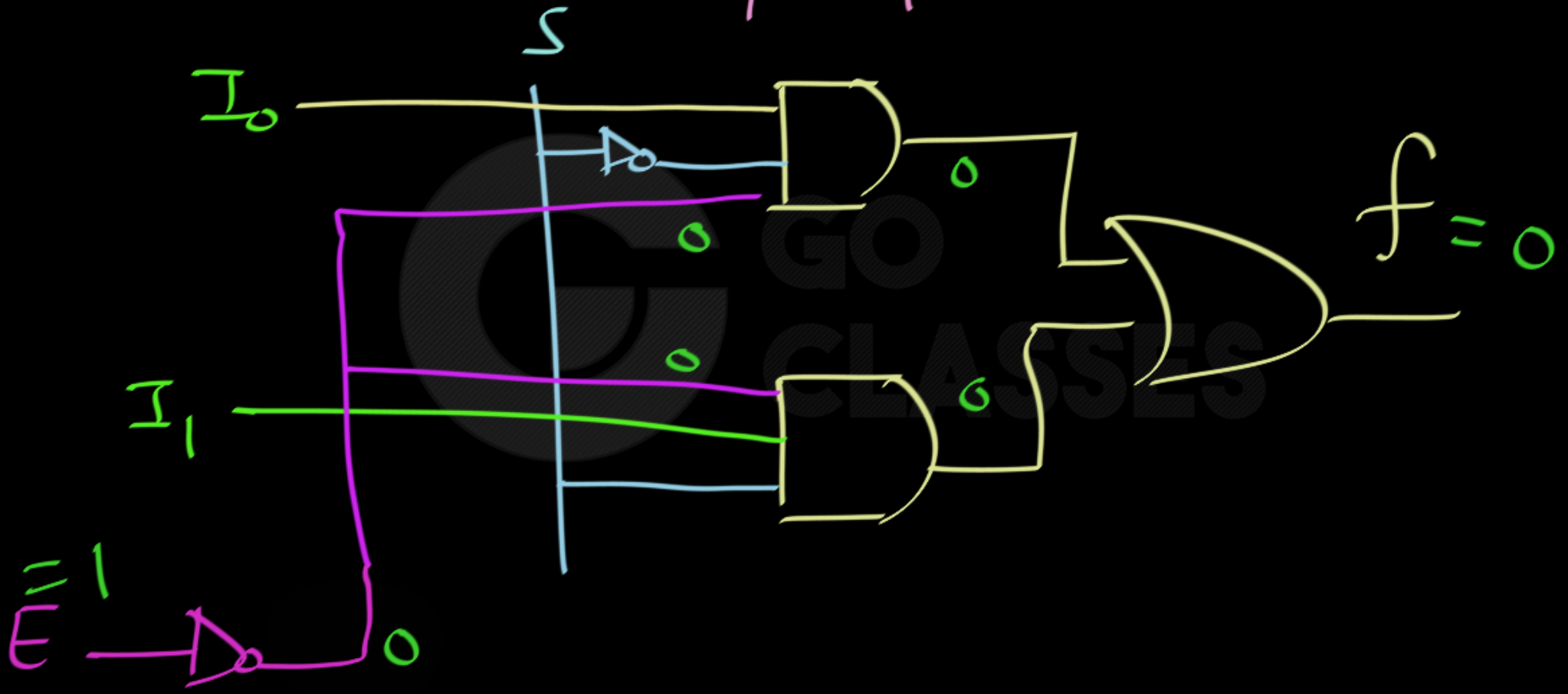
$$f_{(E, S, I_1, I_0)} = \overline{E}\left(\overline{S} I_0 + S I_1\right)$$

$$\begin{cases} E = 0 \Rightarrow f = \overline{S} I_0 + S I_1 \\ E = 1 \Rightarrow f = 0 \end{cases}$$

Implementation using logic gates:

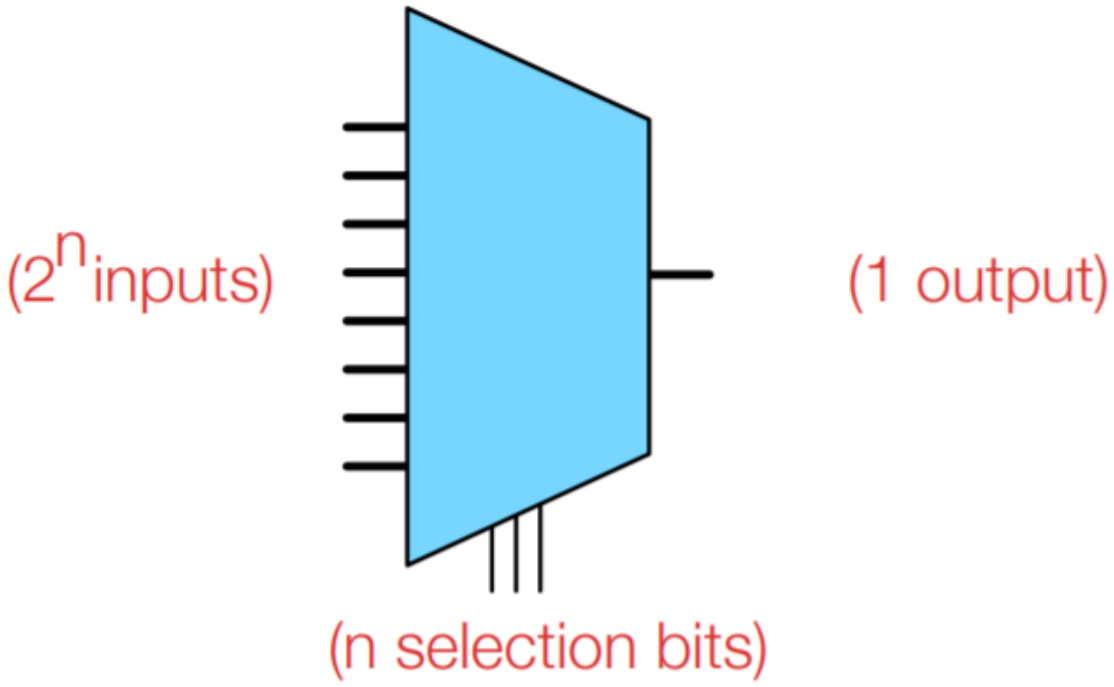


Implementation using logic gates:



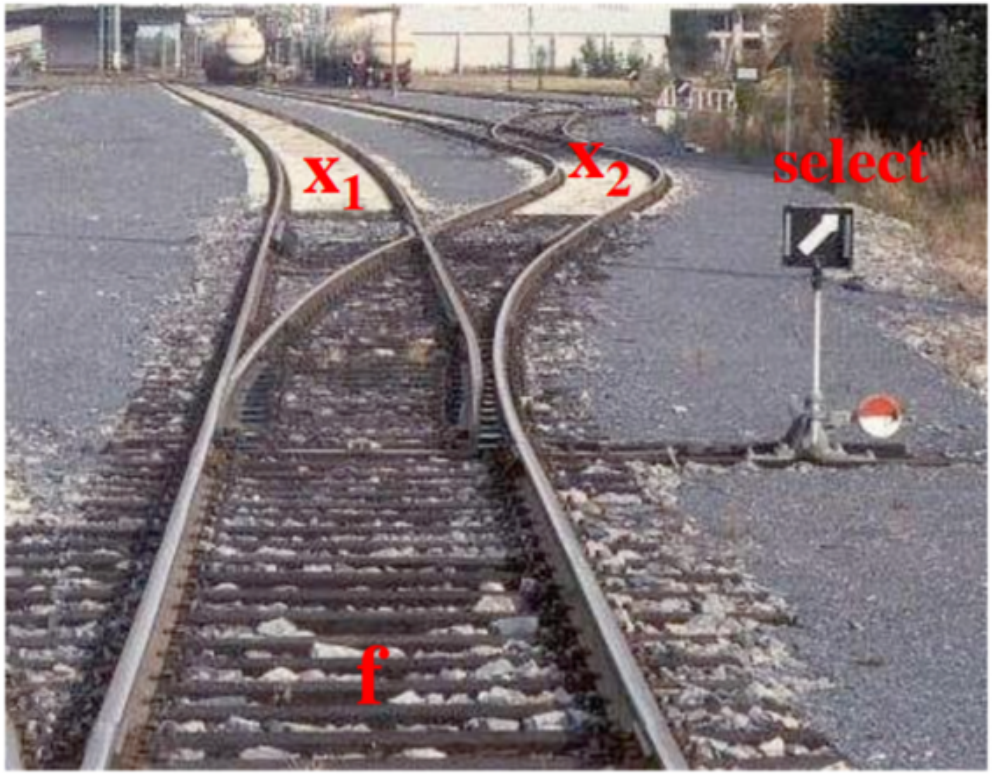
Multiplexers

- Combinational circuit that selects binary information from one of many input lines and directs it to one output line



Analogy: Railroad Switch

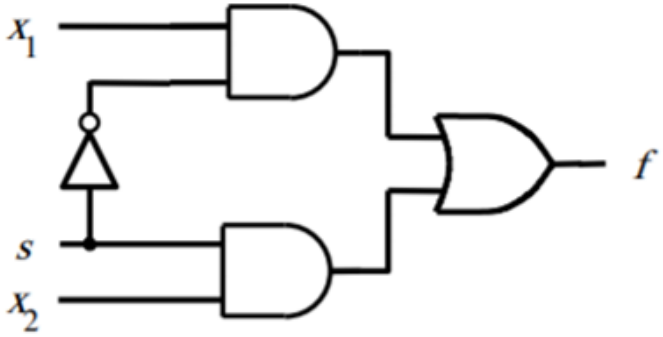
Analogy: Railroad Switch



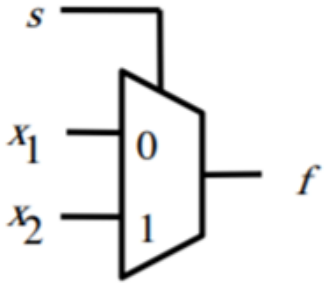
This is not a perfect analogy because the trains can go in either direction, while the multiplexer would only allow them to go from top to bottom.

http://en.wikipedia.org/wiki/Railroad_switch]

Circuit for 2-1 Multiplexer

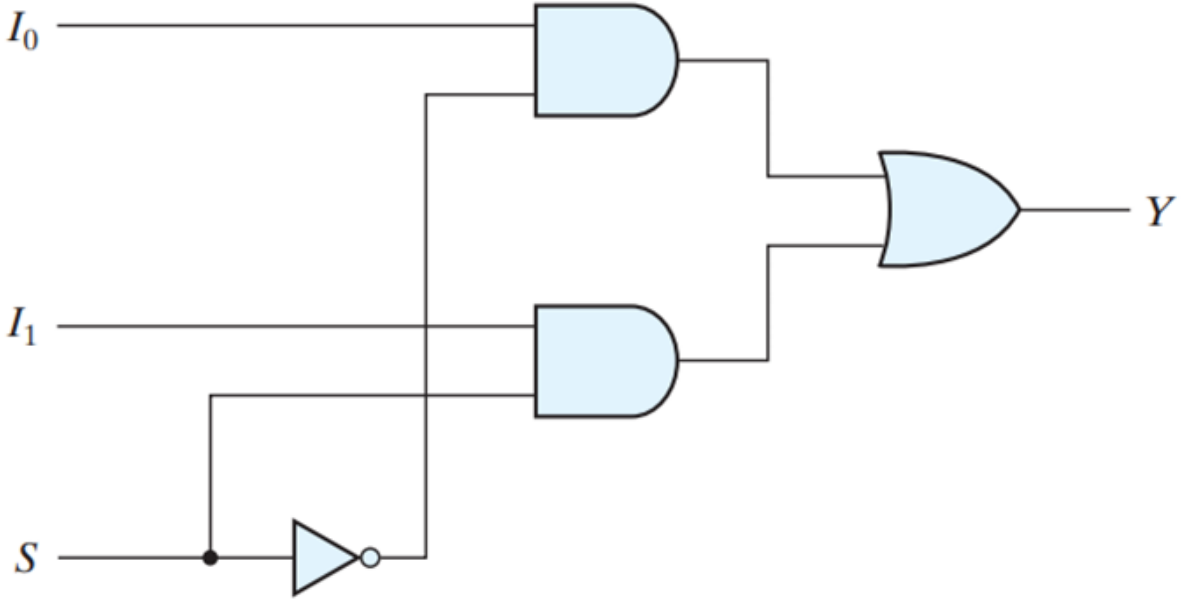


(b) Circuit

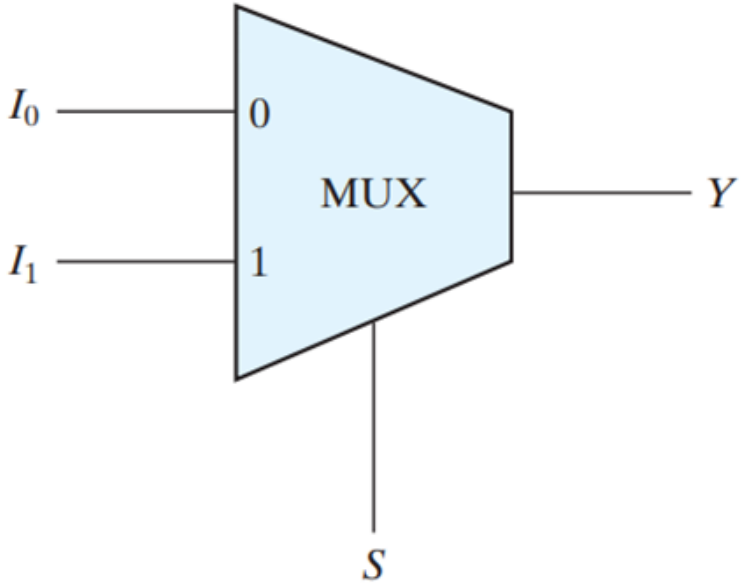


(c) Graphical symbol

$$f(s, x_1, x_2) = \overline{s}\, x_1 + s\, x_2$$



(a) Logic diagram



(b) Block diagram

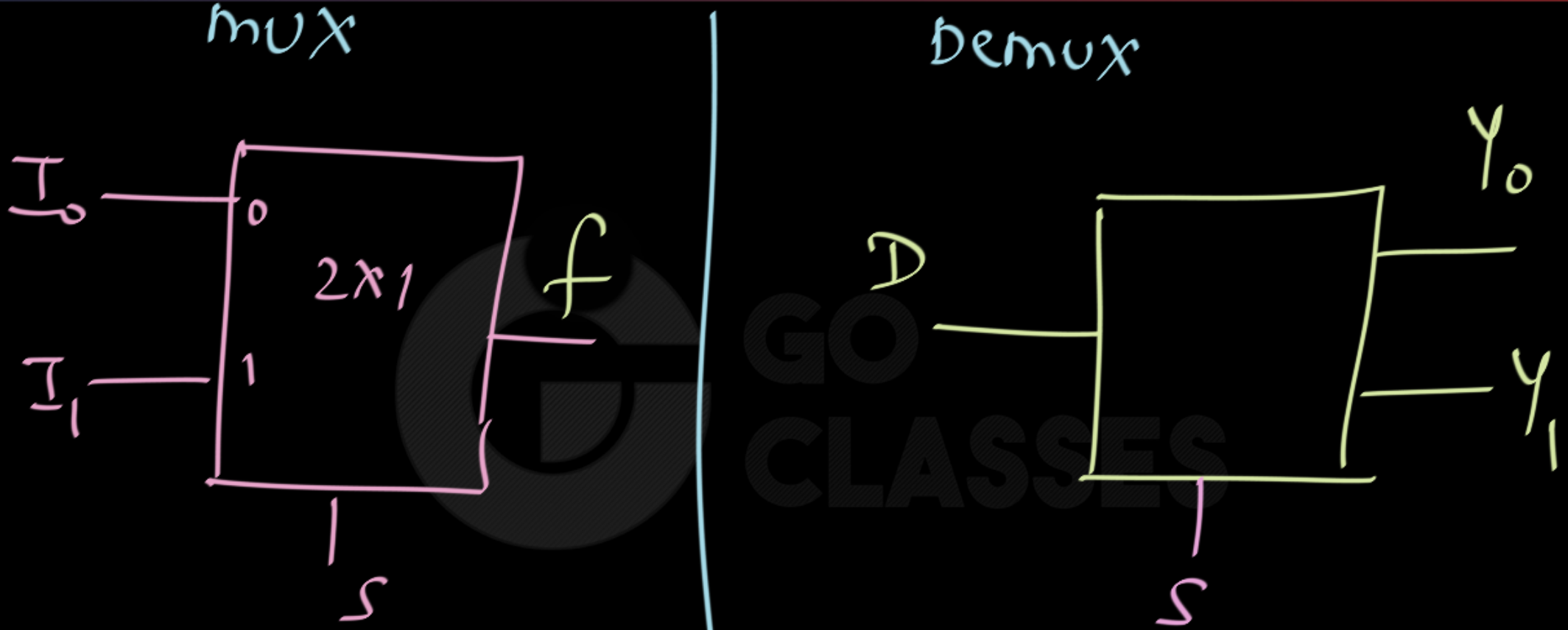
FIGURE 4.24
Two-to-one-line multiplexer

Next Topic:

Demultiplexer

DeMux

MUX	Demux
2^n input lines	1 input line
1 output lines	2^n o/p line
n select lines	n select lines
Select one input to send on o/p	Select one o/p line to send incoming Data.

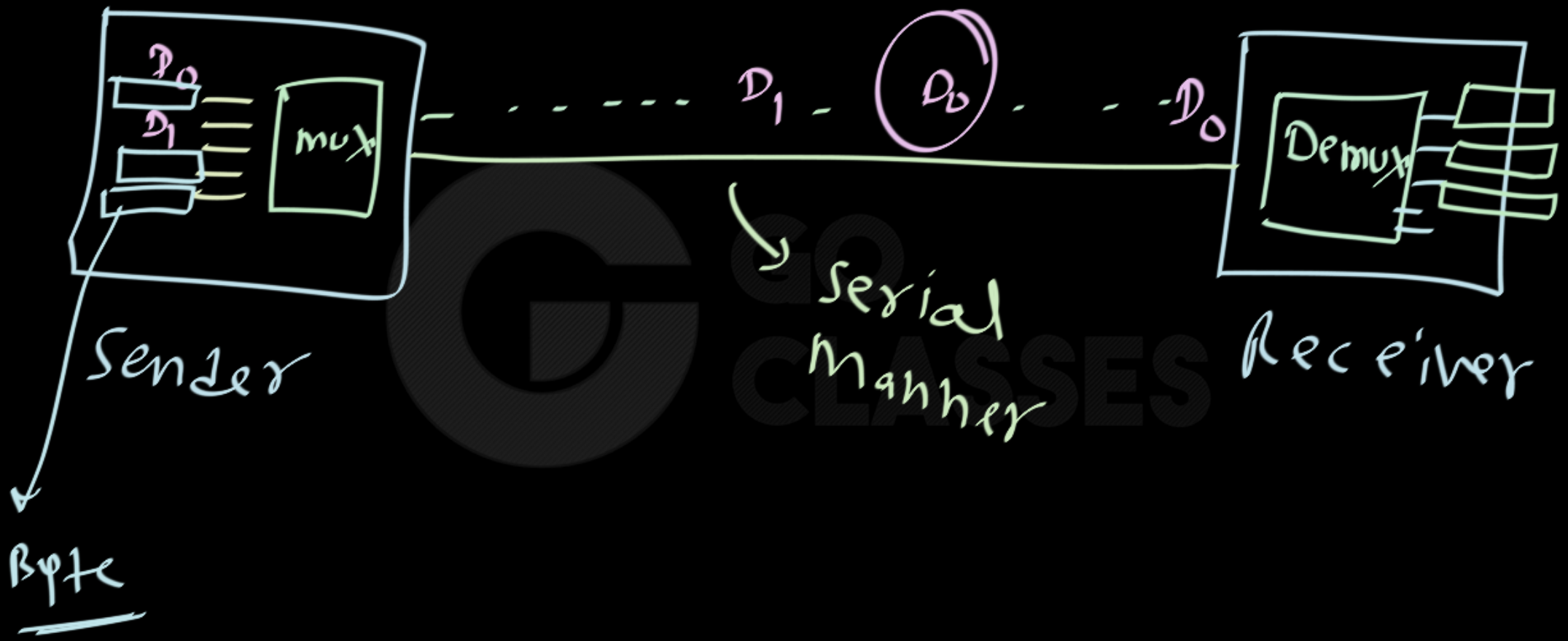

MUX
I_0
I_1
0
1
2x1
f
S
Demux
D
Y_0
Y_1
S

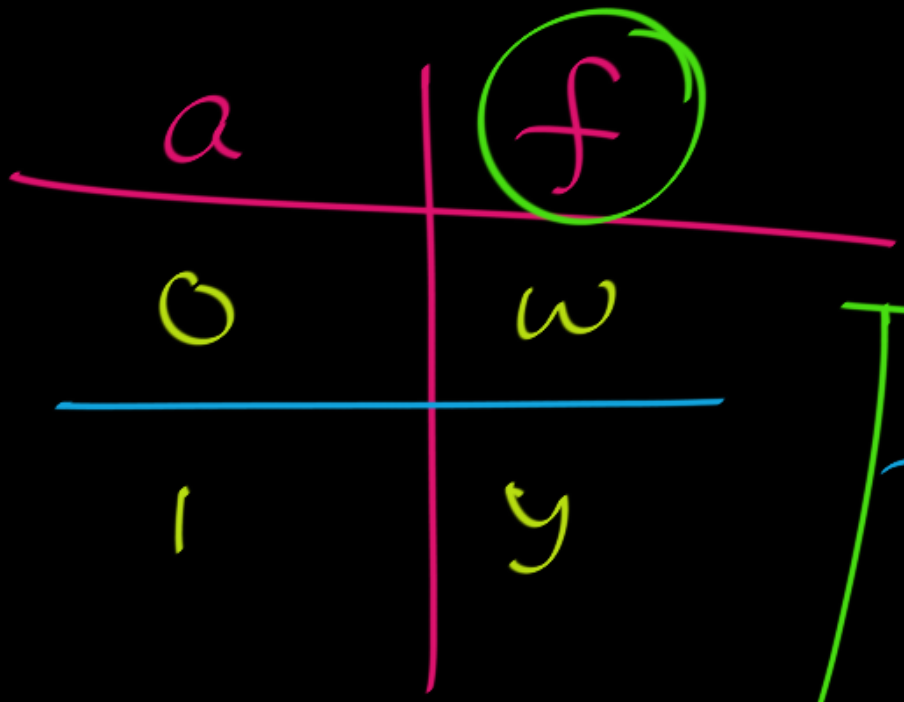
Demux:

1×2^n Demux ; n select lines

input line (Data input)

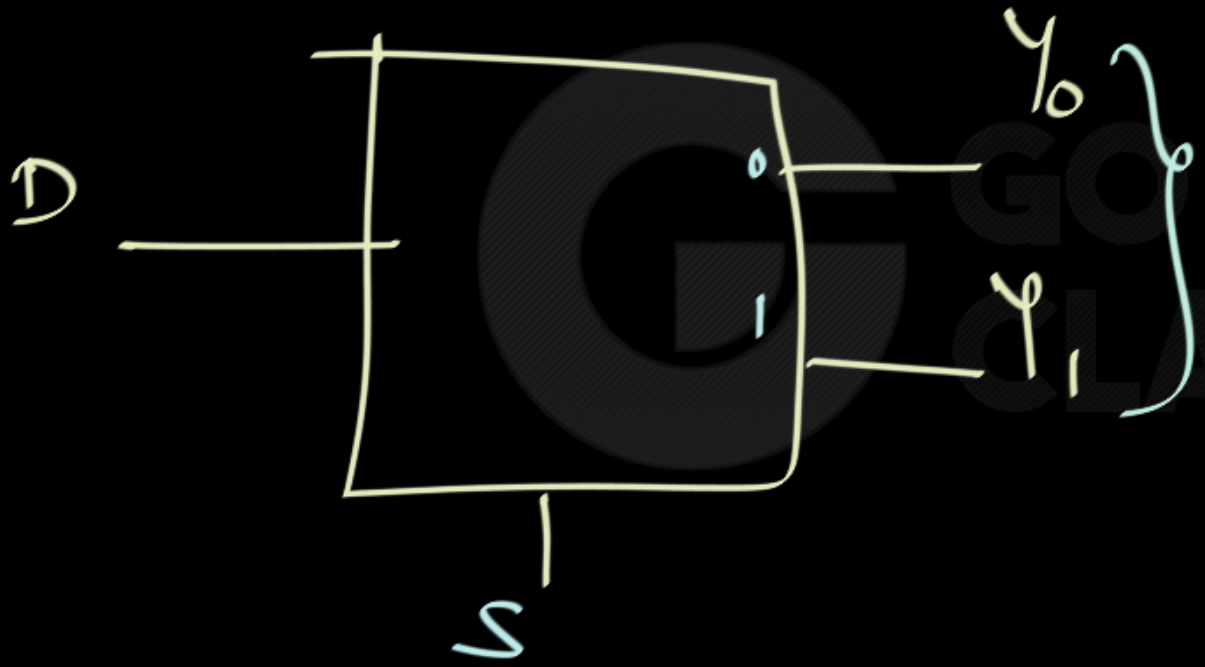
output line


D_0
D_1
mux
D_1
D_0
D_0
Demux
Serial Manner
Sender
Receiver
Byte



$$f = \bar{a}w + ay$$

1x2 Demux:



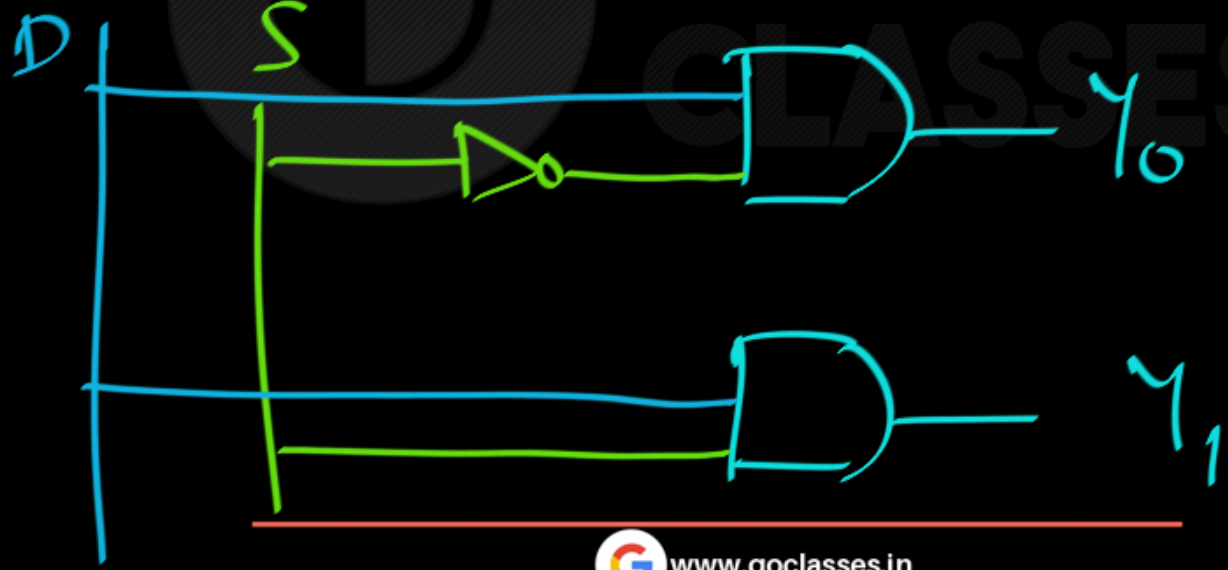
S	Y_0	Y_1
0	D	0
1	0	D

$Y_0 (D, S)$

$Y_1 (D, S)$

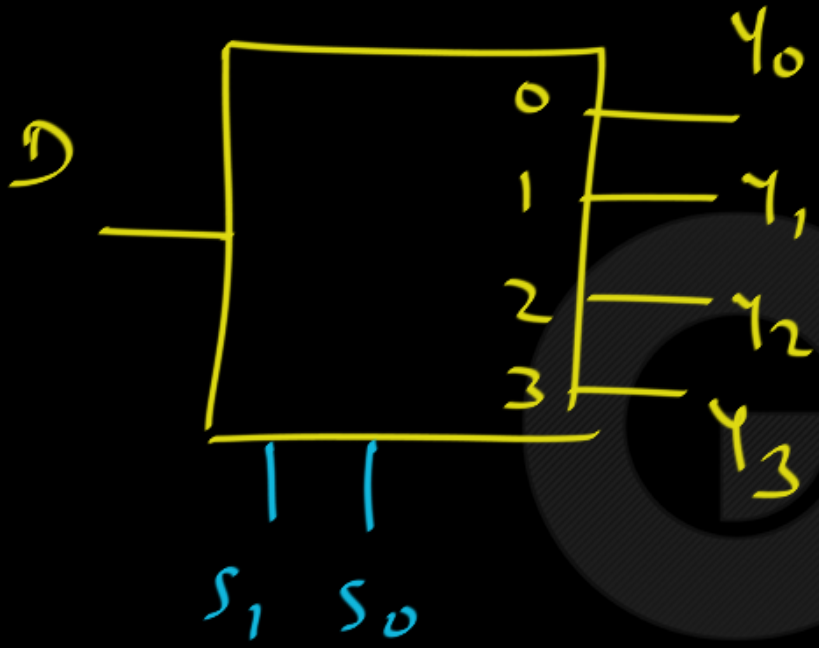
$Y_0(D,S) = \bar{S}D + 0S = \bar{S}D$

$Y_1(D,S) = 0\bar{S} + DS = SD$

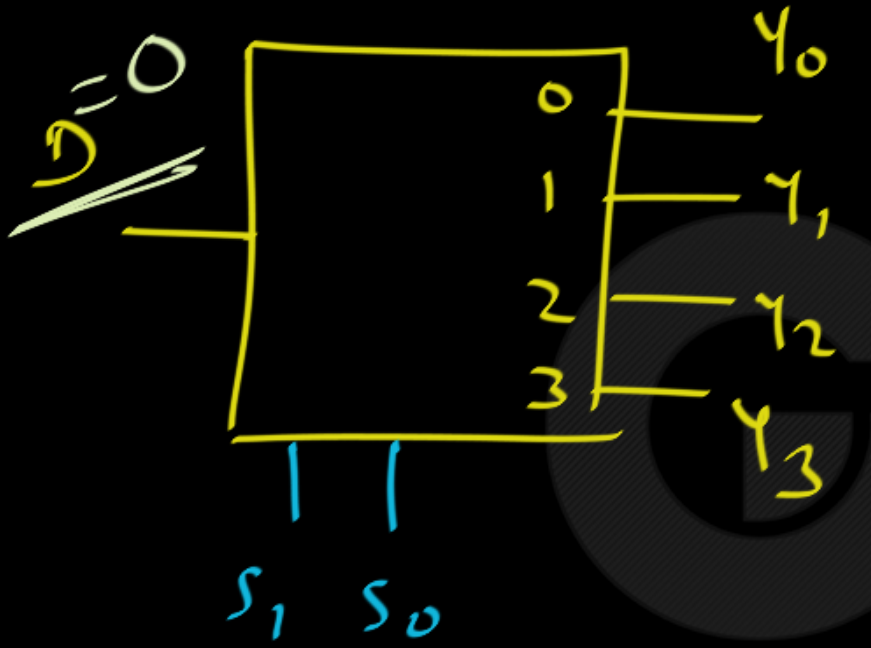


1-to-4 Demultiplexer (Definition)

- Has one data input line: D
- Has two output select lines: 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 D
- If w_1=0 and w_0=1, then the output y_1 is set to D
- If w_1=1 and w_0=0, then the output y_2 is set to D
- If w_1=1 and w_0=1, then the output y_3 is set to D
- Only one output is set to D. All others are set to 0.



S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	D	0	0	0
0	1	0	D	0	0
1	0	0	0	D	0
1	1	0	0	0	D



S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	0	0	0	0
0	1	0	0	0	0
1	0	0	0	0	0
1	1	0	0	0	0

Next Topic:

Encoder

Encoding : Giving Code

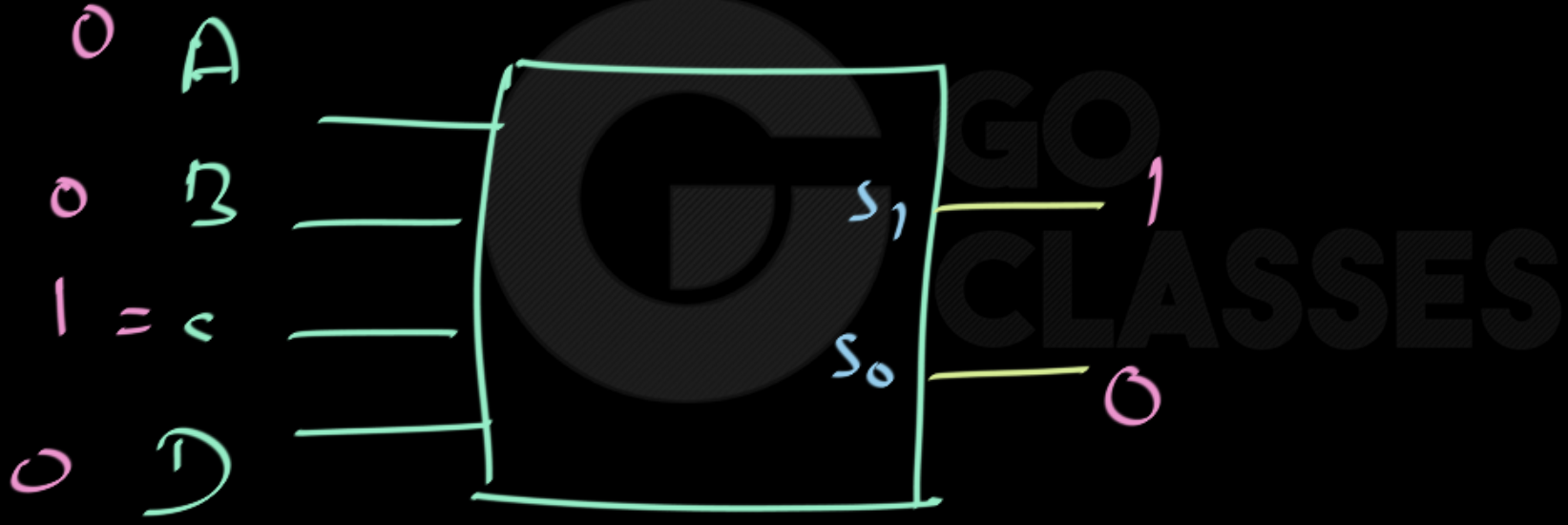
People	Encoding
A	0
B	1

People	Binary Encoding
A	0 0
B	0 1
C	1 0
D	1 1

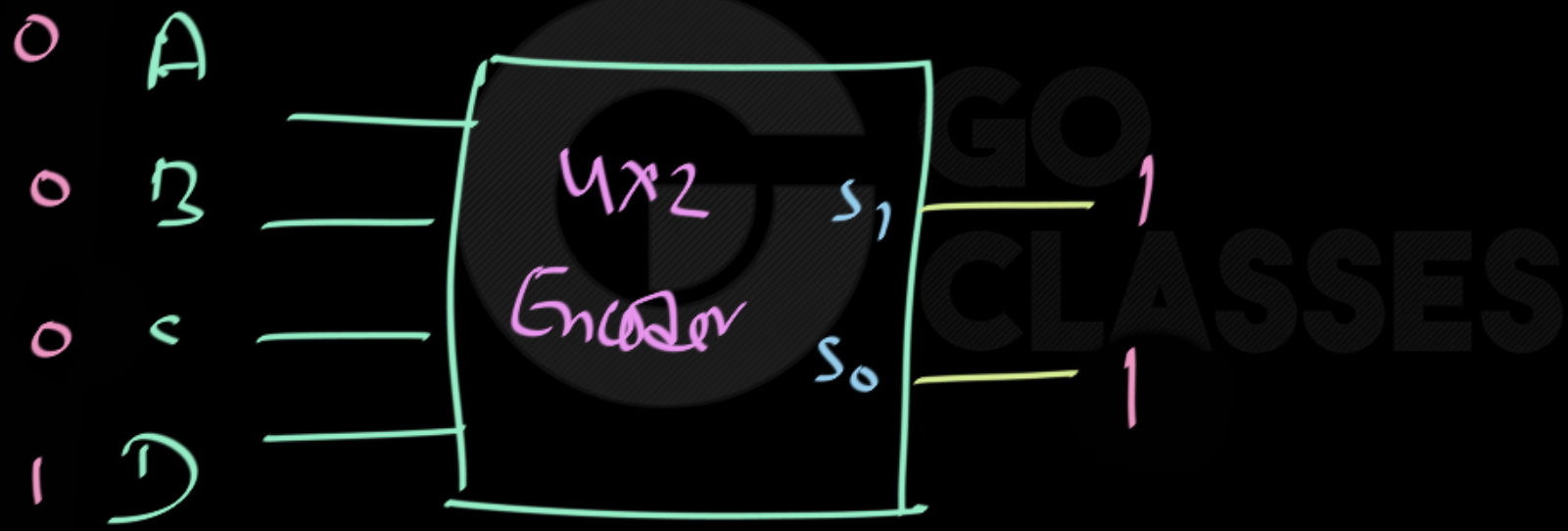
Encoding : Giving Code

People	Binary Encoding
a_0	00...0 (n bits)
a_1	00...1
⋮	⋮
a_{2^n-1}	11...1

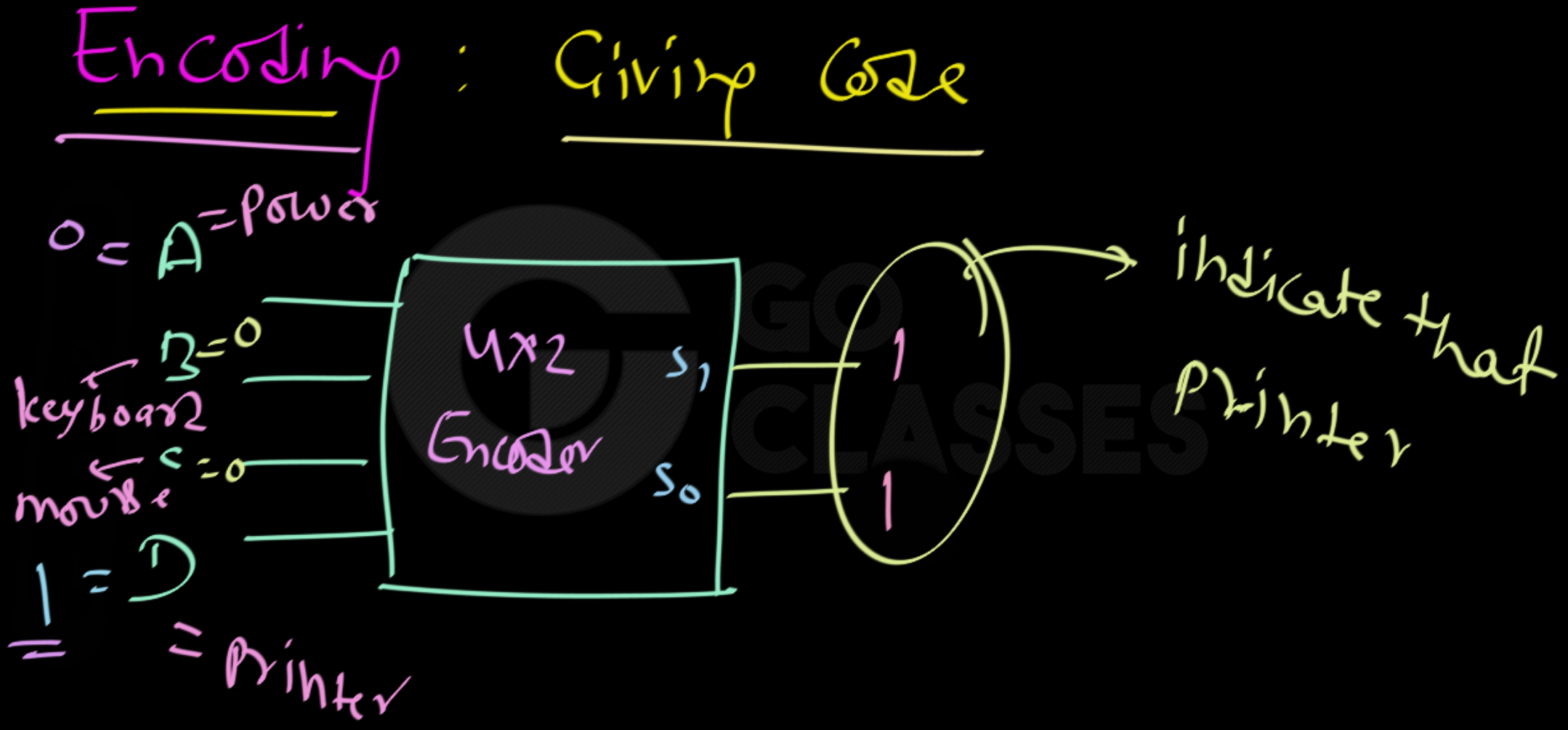
Encoding : Giving Code



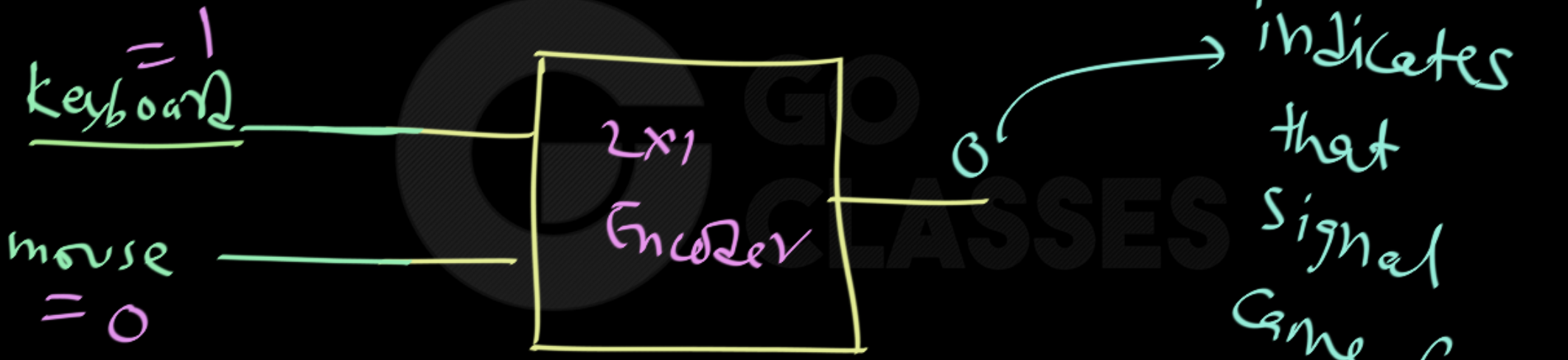
Encoding : Giving Code



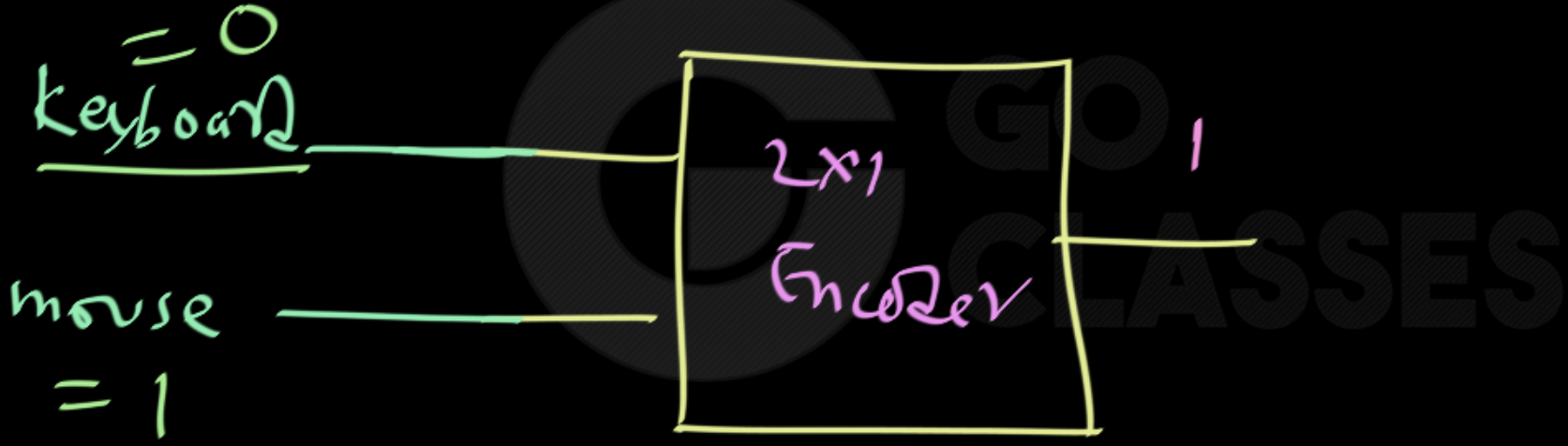
Encoding : Giving Code



Encoding : Giving Code



Encoding : Giving Code



Encoders
(there are several types)

Next Topic:

Binary Encoders

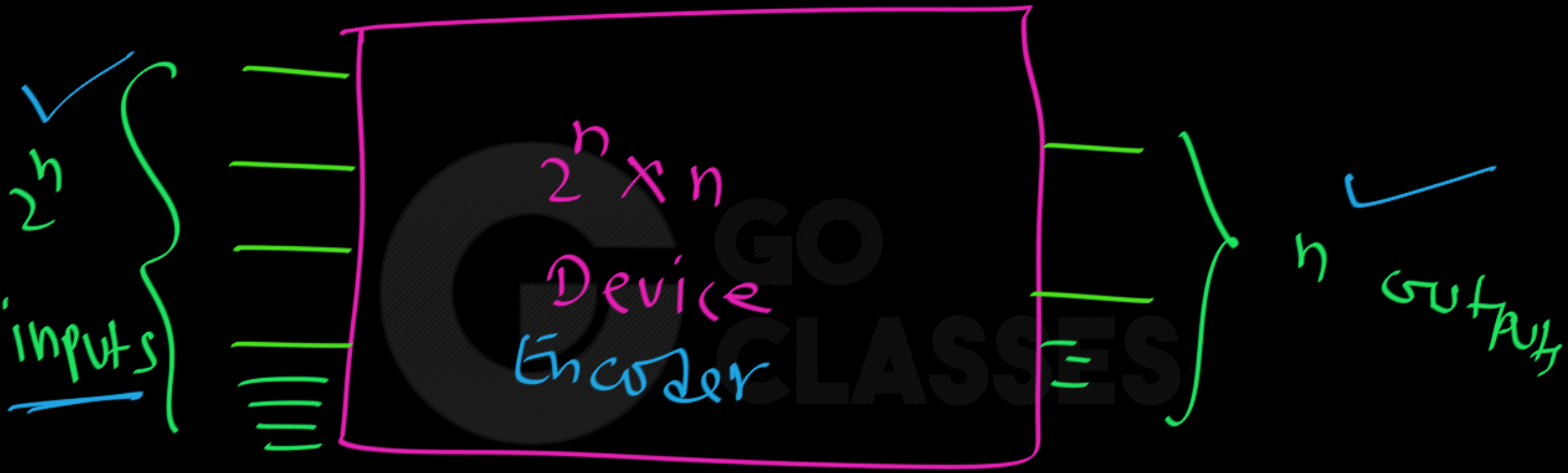
= Allows "only one input to be 1" at a time.

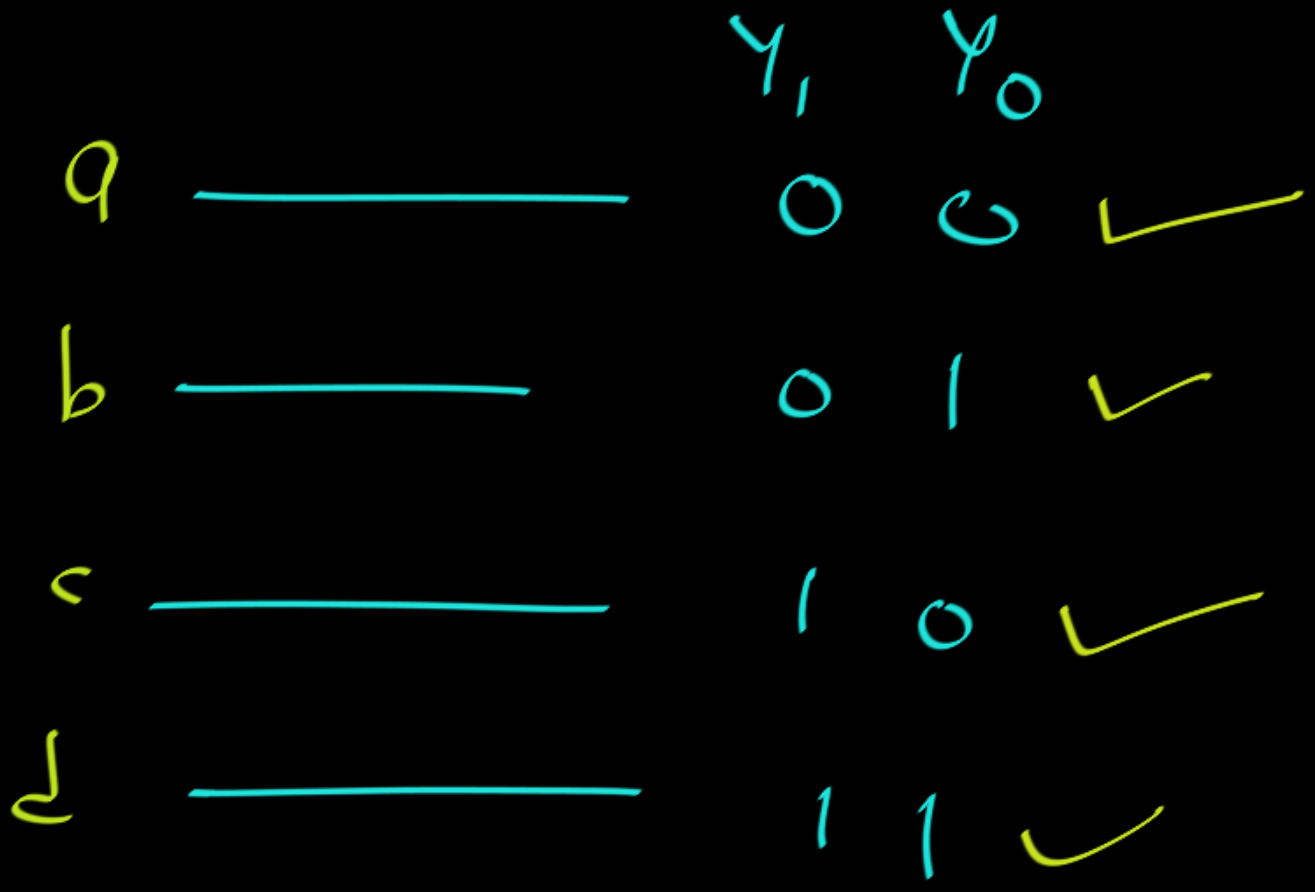
Binary Encoder

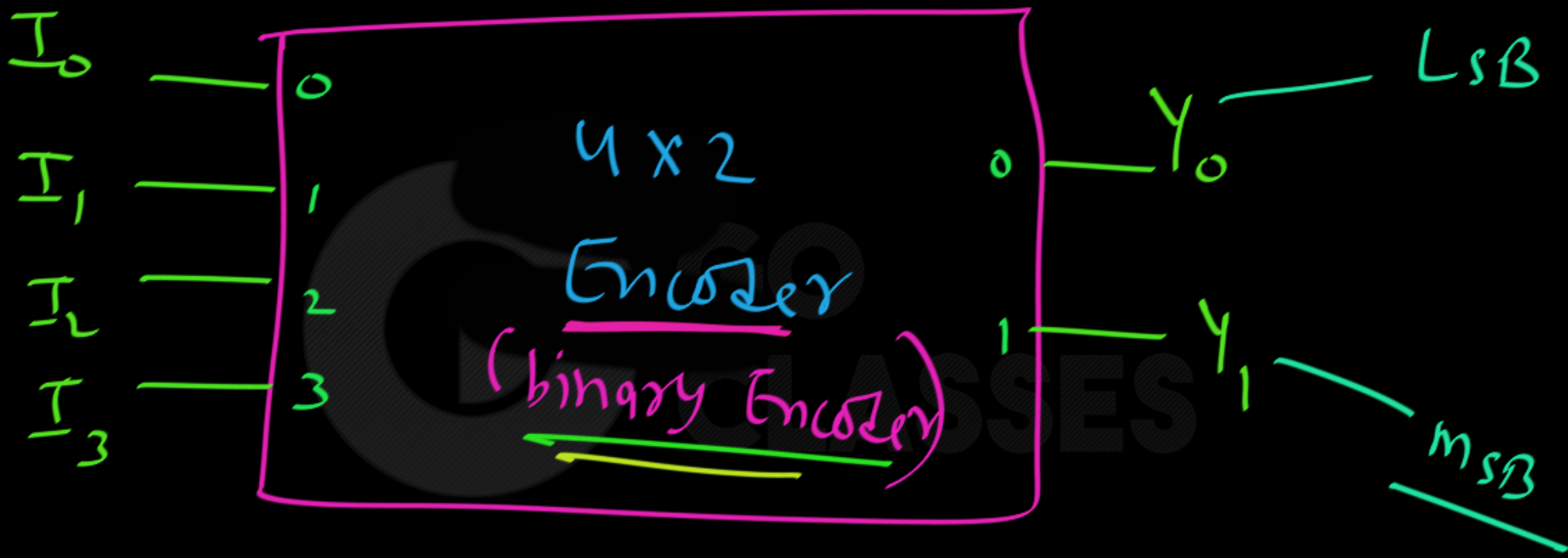
Only one input signal can be 1 at a time.

Analogy

Giving Code to People.


2^n
inputs
$2^n \times n$
Device
Encoder
n
outputs


y_1 y_0
a
0 0
b
0 1
c
1 0
d
1 1


I_0
I_1
I_2
I_3
0
1
2
3
4 x 2
Encoder
(binary Encoder)
0
1
Y_0
Y_1
LSB
MSB

MSB

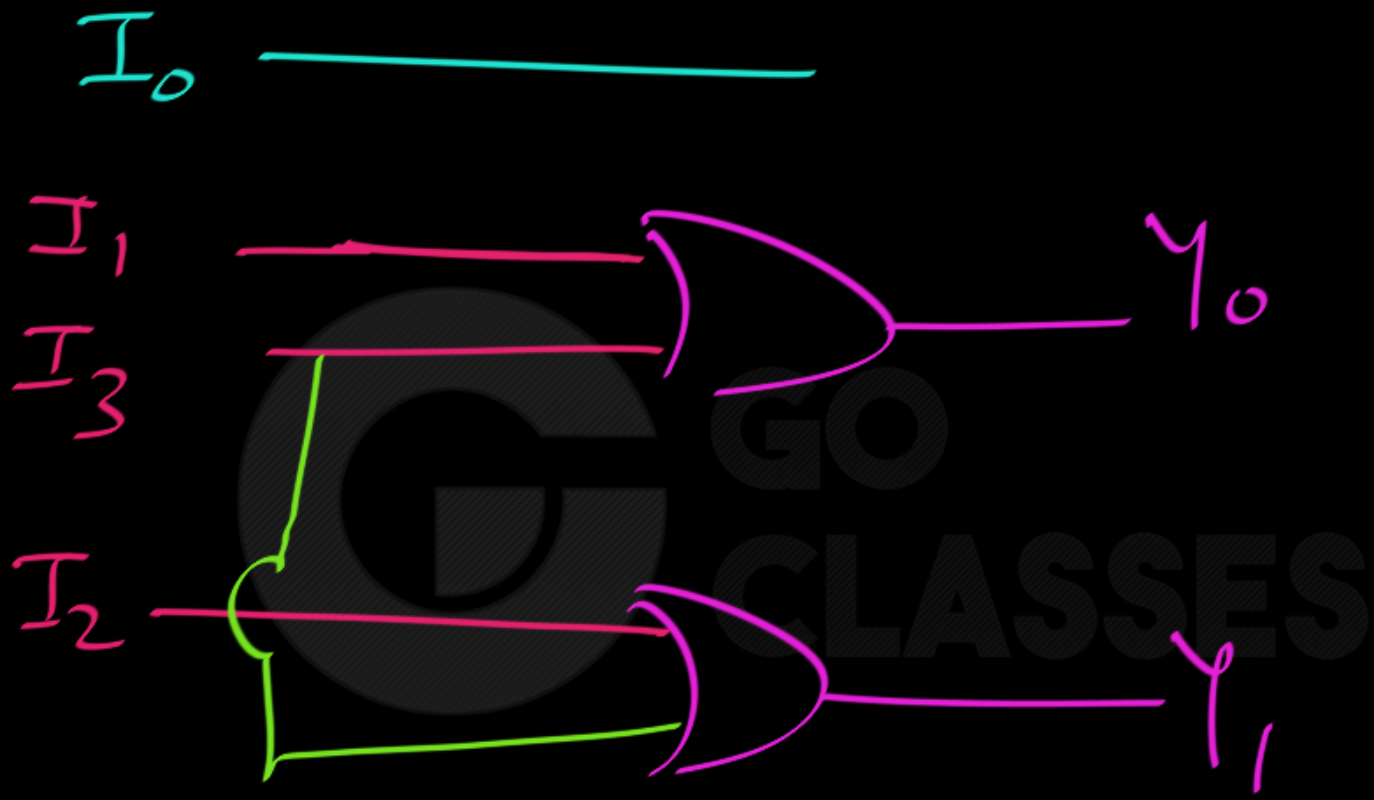
I_0	I_1	I_2	I_3	Y_1	Y_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

Remaining Combinations Can NEVER occur

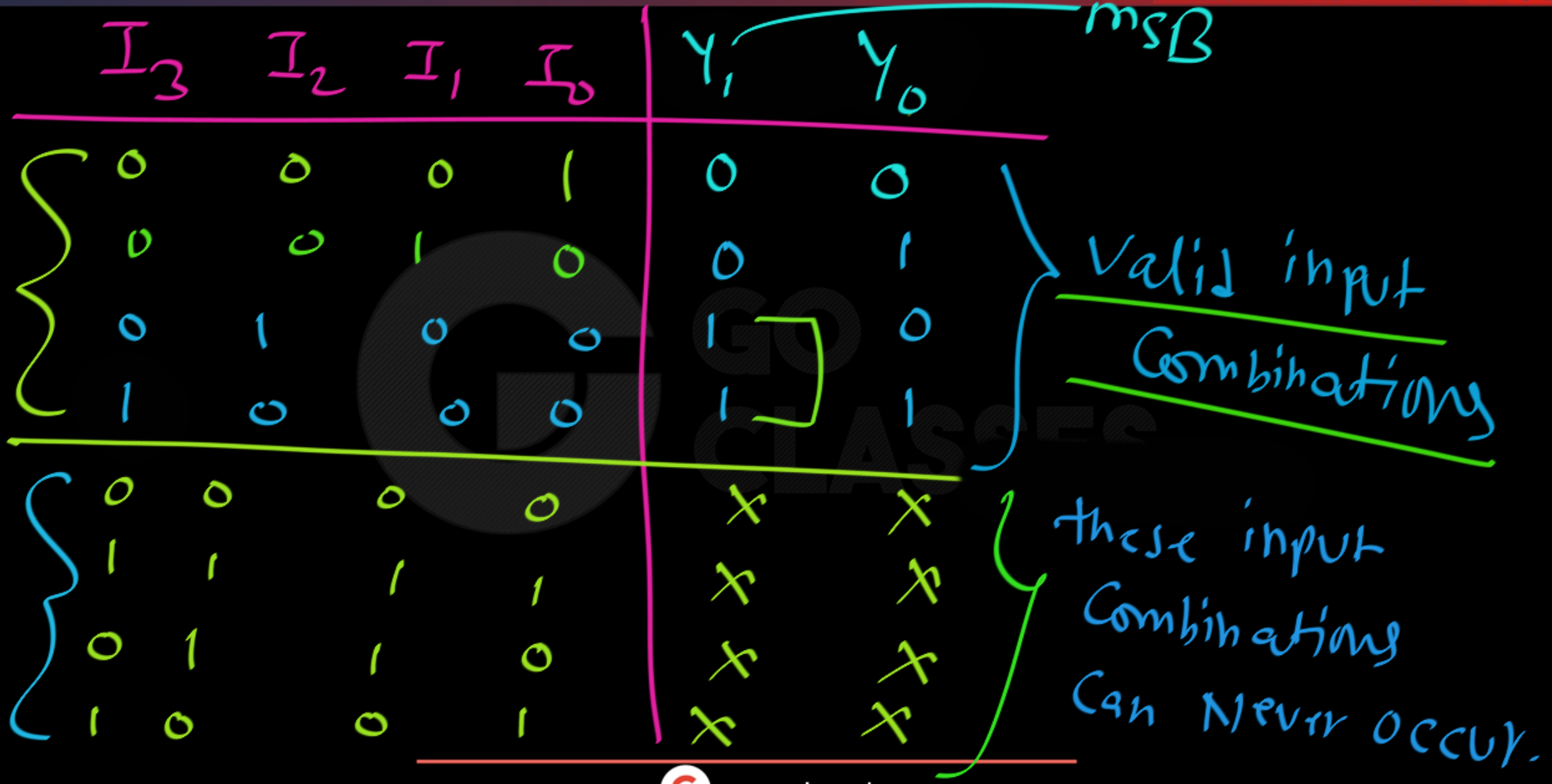
All Don't Cares

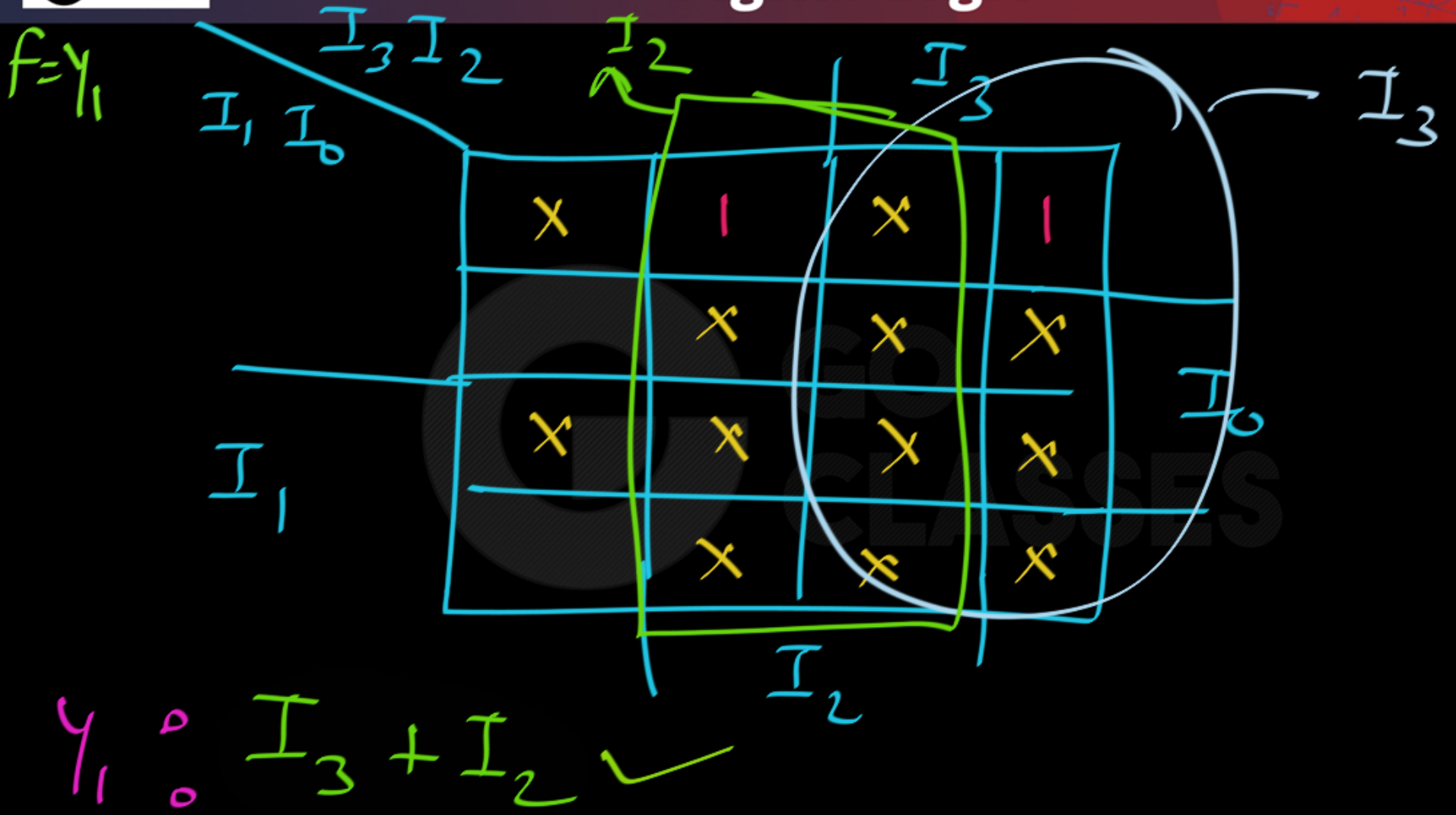
$$Y_1(I_3, I_2, I_1, I_0) = I_2 + I_3$$

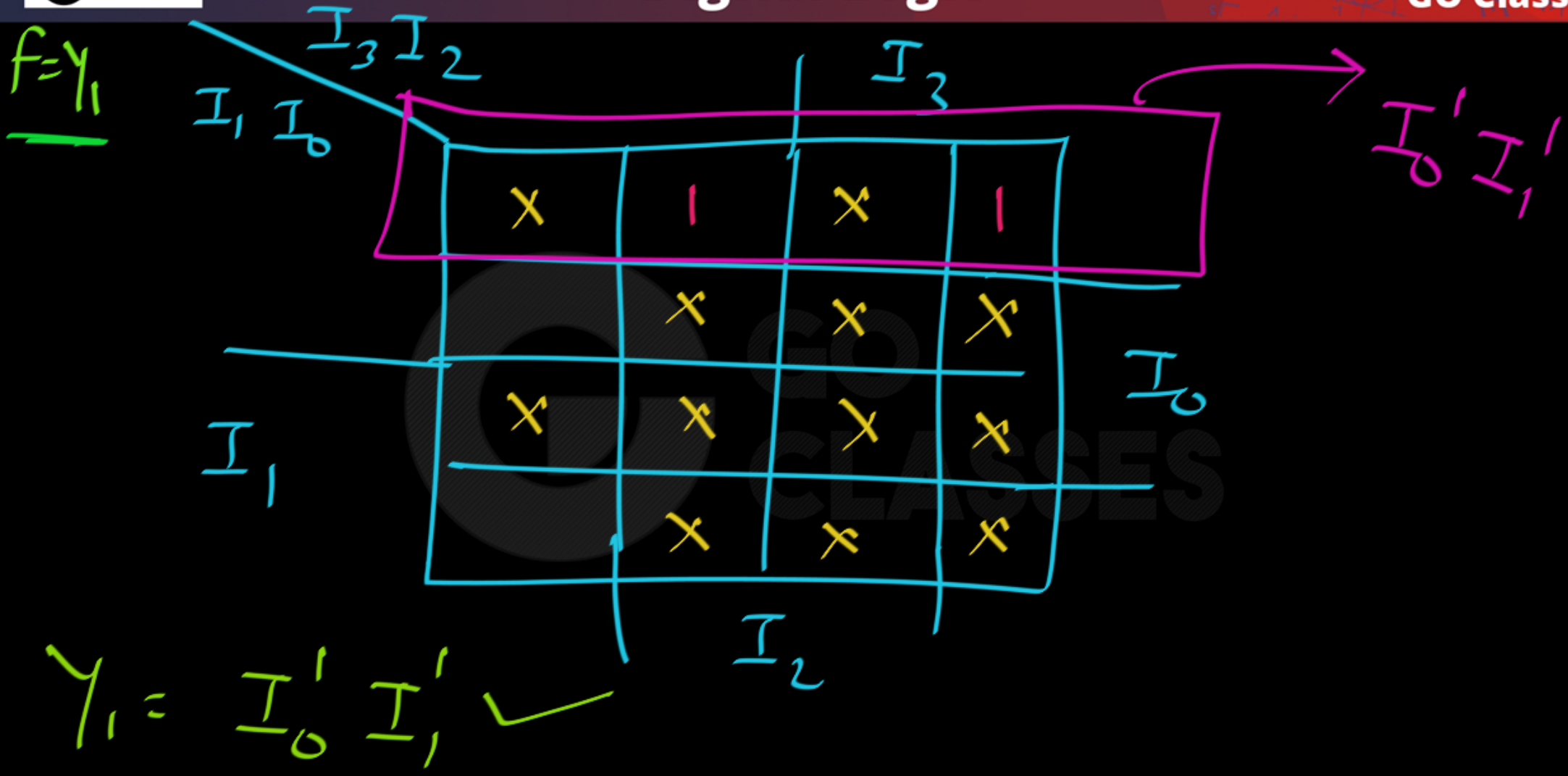
$$Y_0(I_3, I_2, I_1, I_0) = I_1 + I_3$$



4 X 2 Binary Encoder implementation using logic gates


I_3 I_2 I_1 I_0
Y_1 Y_0
MSB
0 0 0 1 — 0 0
0 0 1 0 — 0 1
0 1 0 0 — 1 0
1 0 0 0 — 1 1
Valid input Combinations
0 0 0 0 — x x
1 1 1 1 — x x
0 1 1 0 — x x
1 0 0 1 — x x
these input Combinations Can Never occur.


f = Y1
I3 I2
I1 I0
I2
I3
I3
I1
I2
Y1 = I3 + I2


f = Y1
I3 I2
I1 I0
I3
I0' I1'
I0
I1
I2
Y1 = I0' I1'

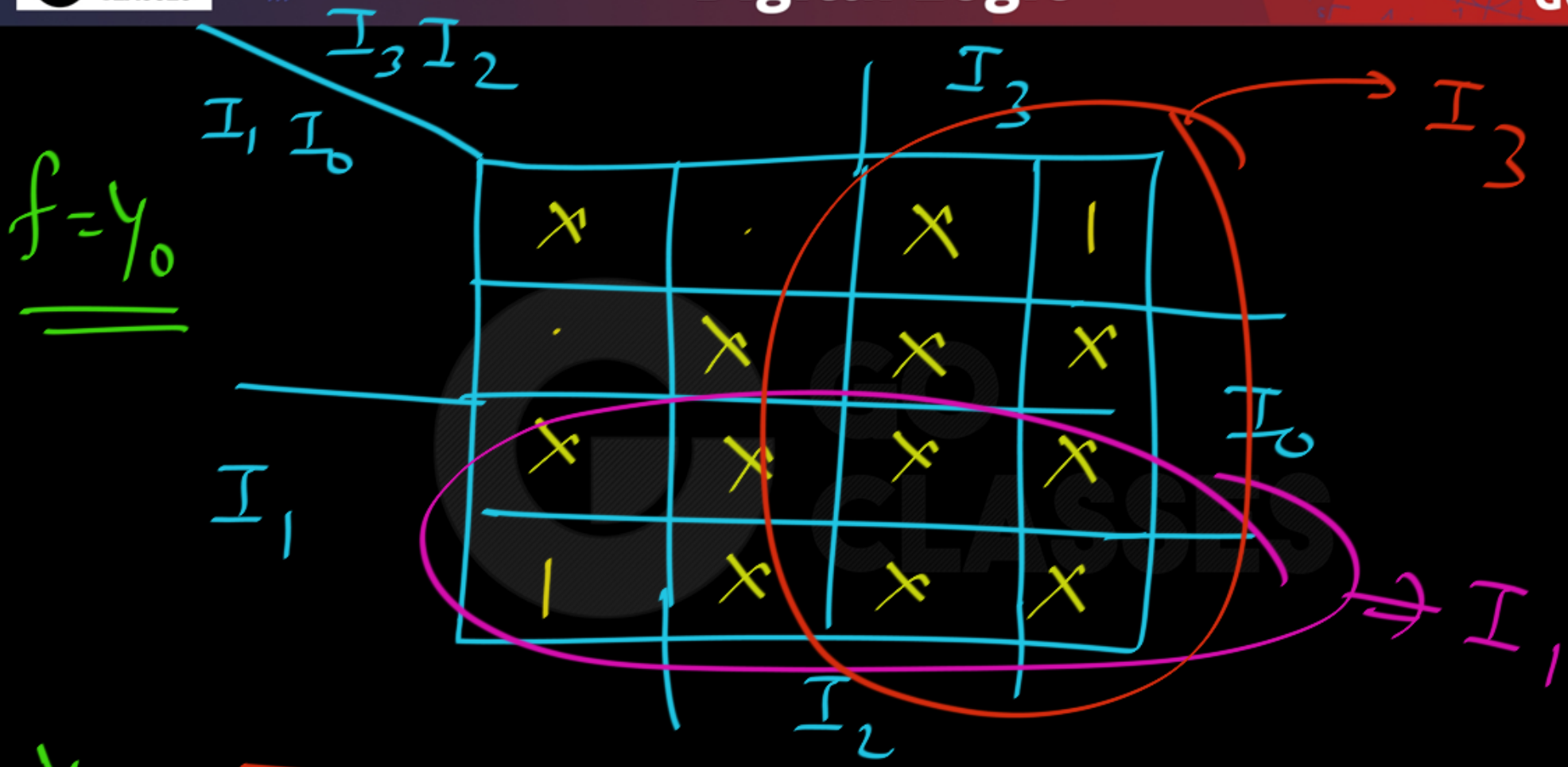
$$Y_1 = \left\{ \begin{array}{l} I_3 + I_2 \\ I_0' \, I_1' \end{array} \right\}$$

both minimum

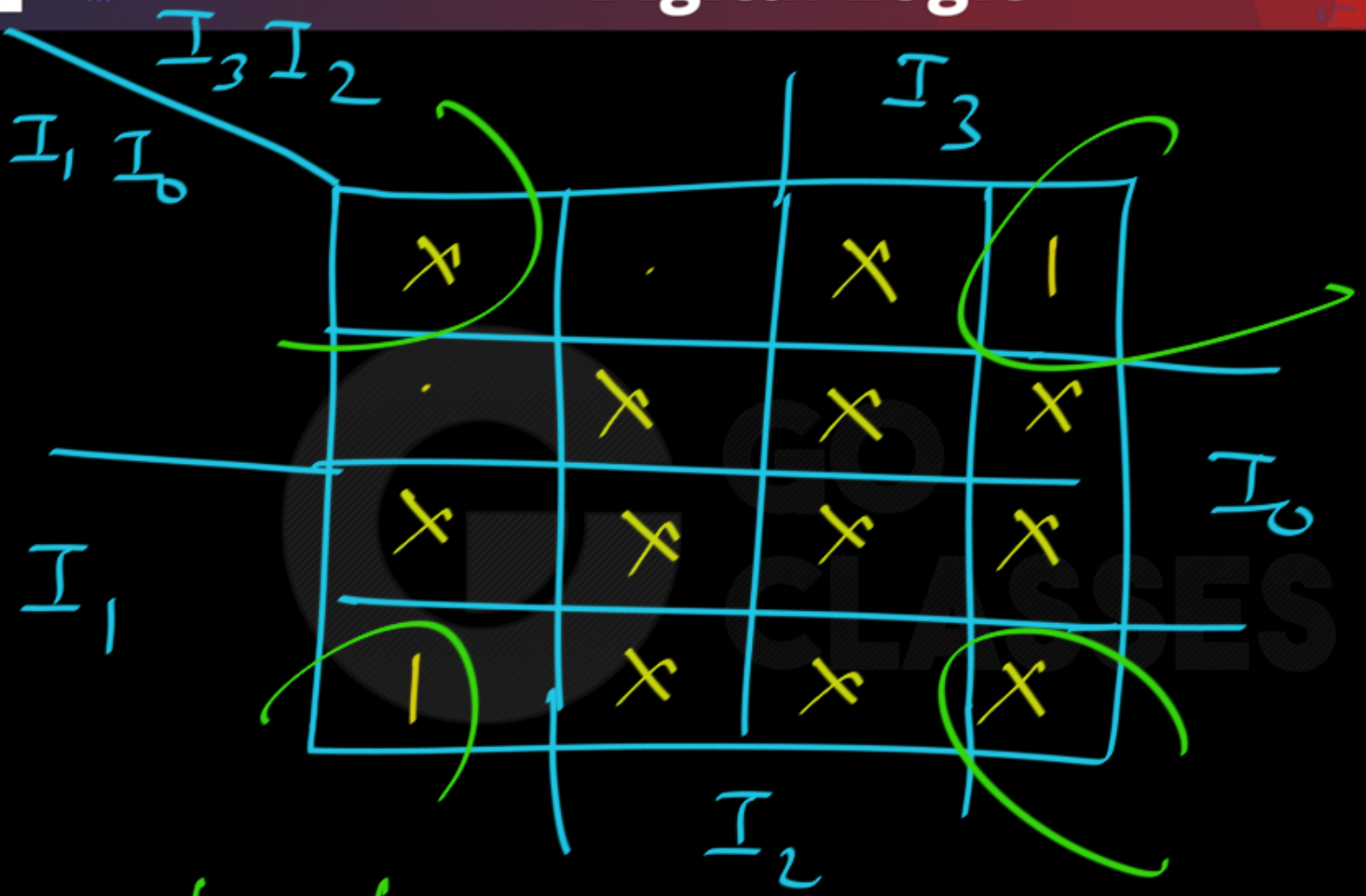
" Correct

" SOP

" POS



$Y_0 = I_3 + I_1$



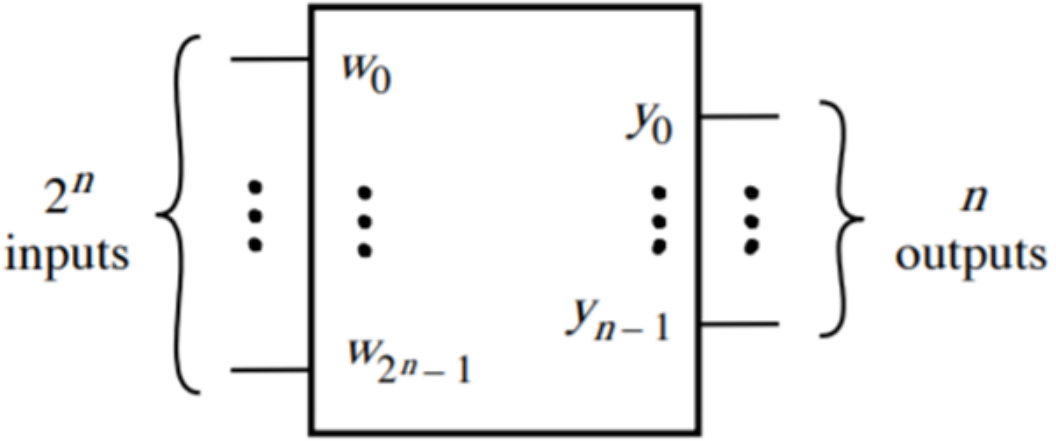
$f = Y_0$

$Y_0 = I_2' I_0'$

$$Y_0 = I_3 + I_1$$

$$= I_2' I_0'$$

The Most General Case: 2^n-to-n binary encoder



4-to-2 Binary Encoder (Definition)

- Has four inputs: w_3 , w_2 , w_1 , and w_0
- Has two outputs: y_1 and y_0
- Only one input is set to 1 ("one-hot" encoded). All others are set to 0.
- If w_0=1 then y_1=0 and y_0=0
- If w_1=1 then y_1=0 and y_0=1
- If w_2=1 then y_1=1 and y_0=0
- If w_3=1 then y_1=1 and y_0=1

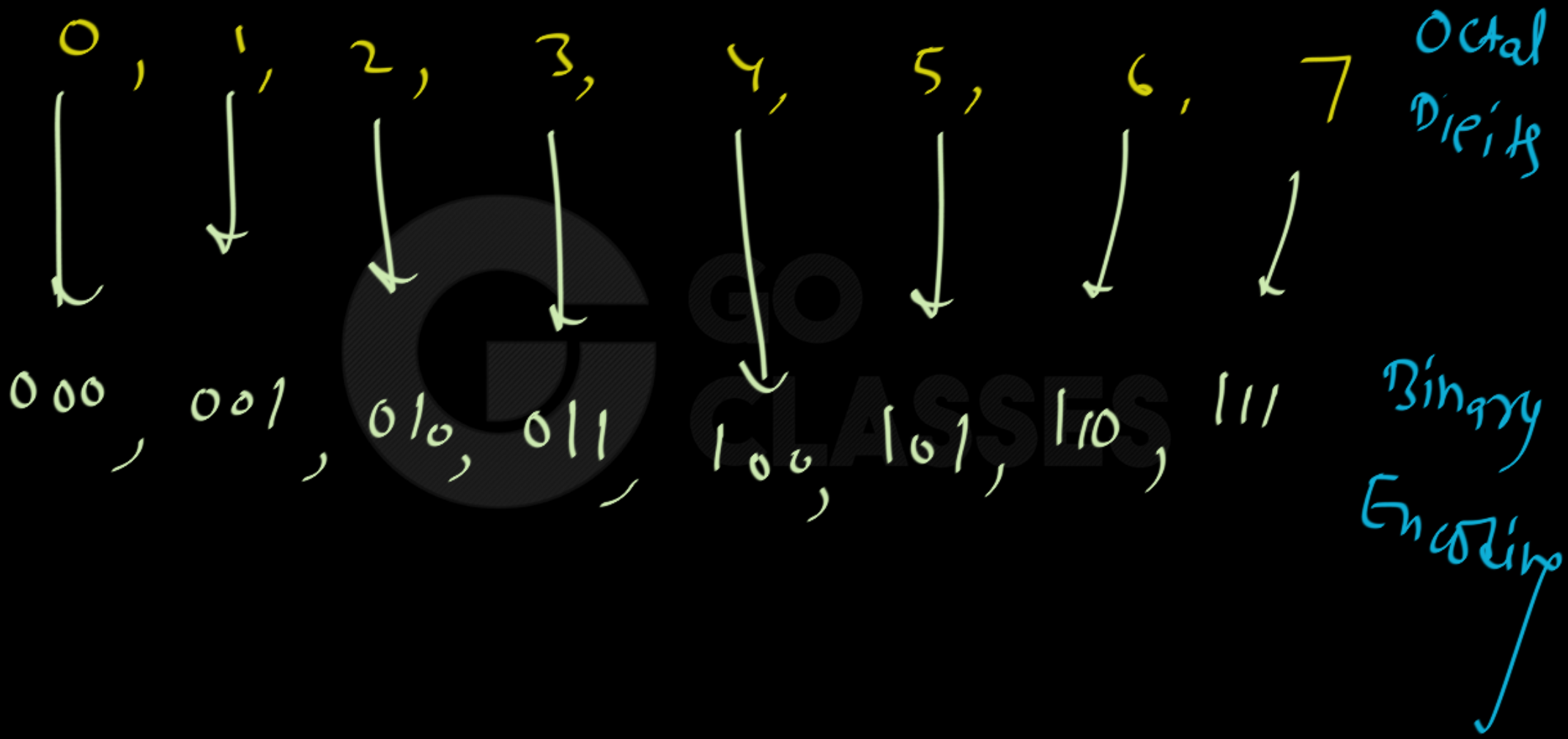
Truth table for a 4-to-2 binary encoder

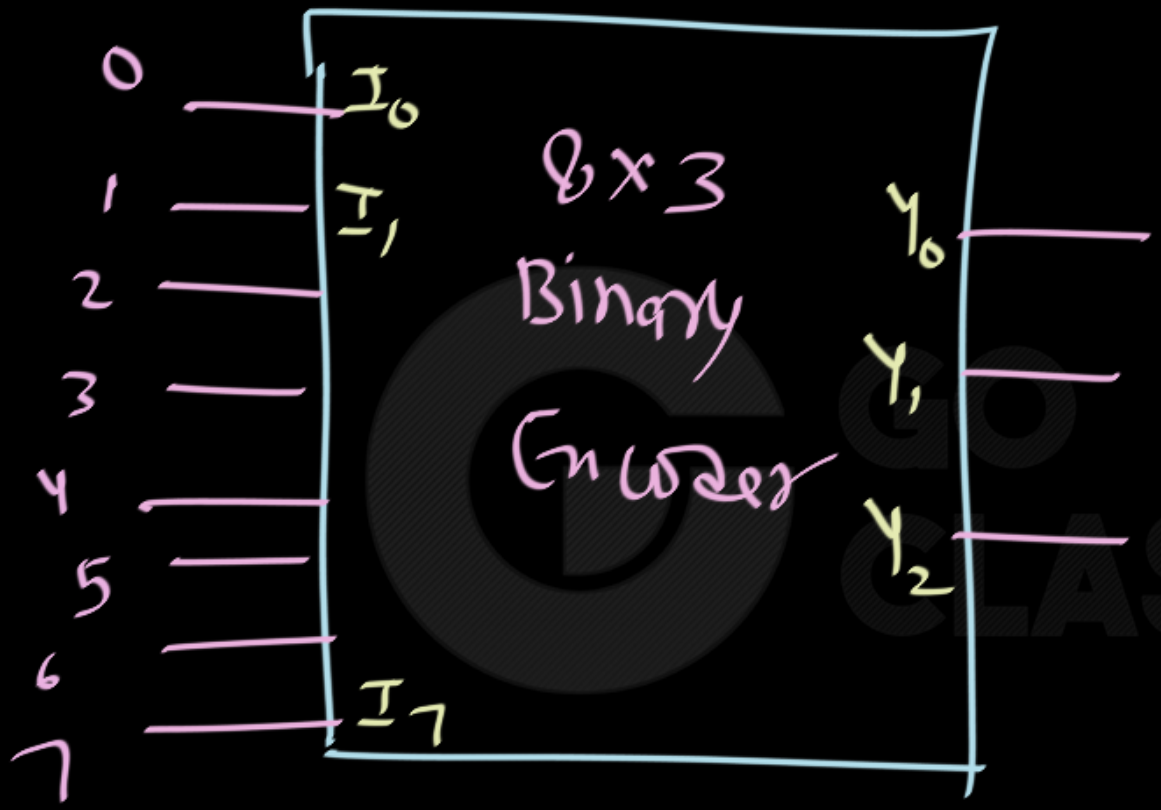
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

The inputs are "one-hot" encoded

Example :

Octal to binary Encoder


0, 1, 2, 3, 4, 5, 6, 7
Octal Digits
000, 001, 010, 011, 100, 101, 110, 111
Binary Encoding


0
1
2
3
4
5
6
7
I_0
I_1
I_7
8 x 3
Binary
Encoder
Y_0
Y_1
Y_2

0 =
0 =
0 =
1 =
0 =
0 =
0 =
0 =

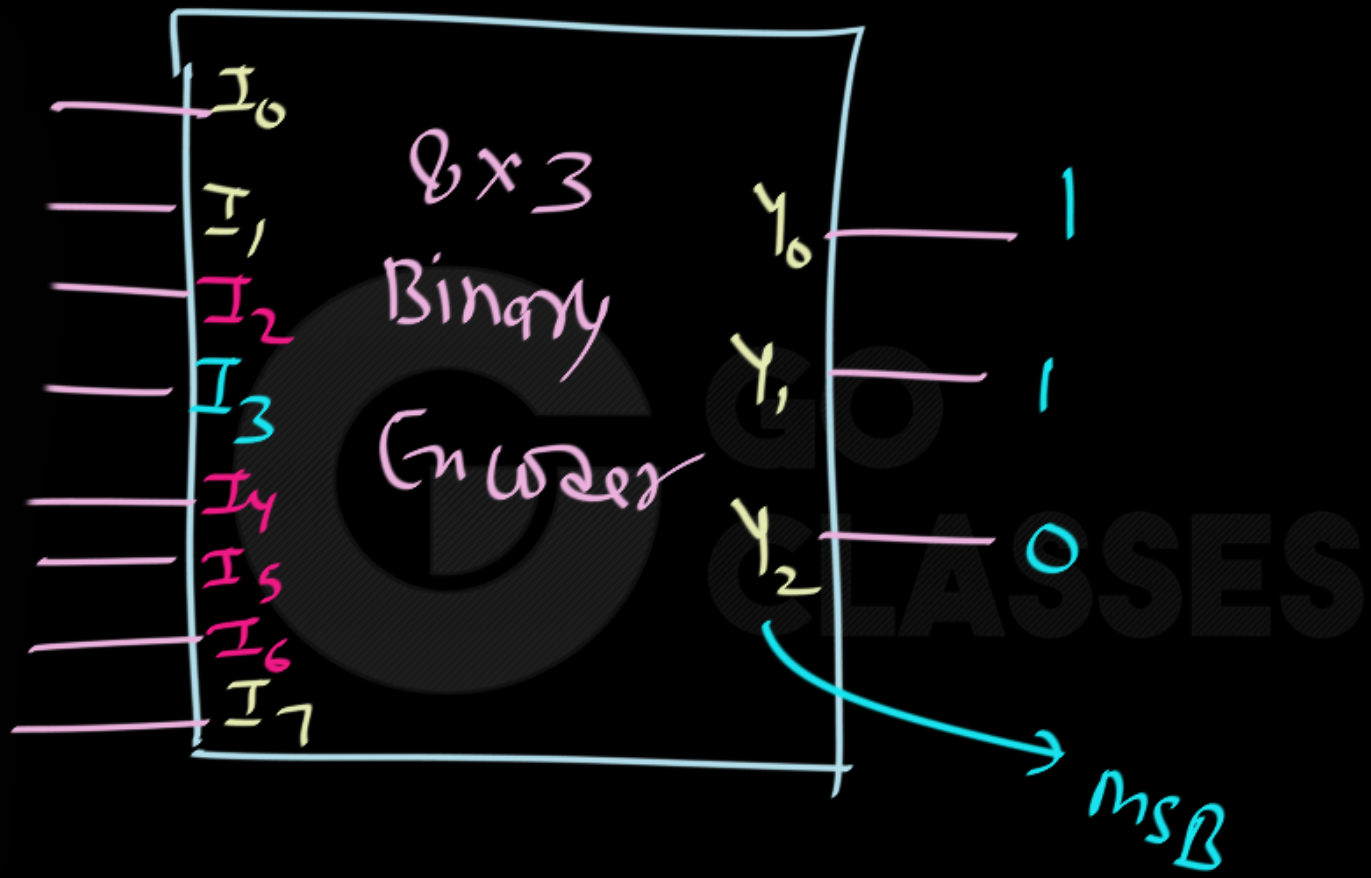


TABLE Truth Table for Octal-to-Binary Encoder

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

Circuit for a 4-to-2 binary encoder

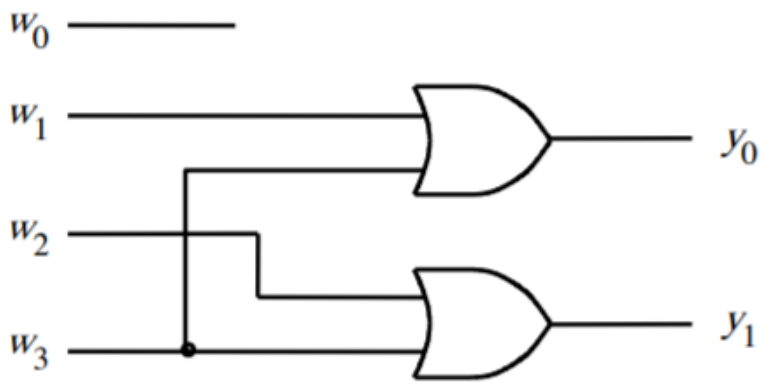
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$Y_1 = w_3 + w_2$

$= \overline{w_0}\ \overline{w_1}$

$Y_0 = w_3 + w_1$

$= \overline{w_0}\ \overline{w_2}$



Circuit for a 4-to-2 binary encoder

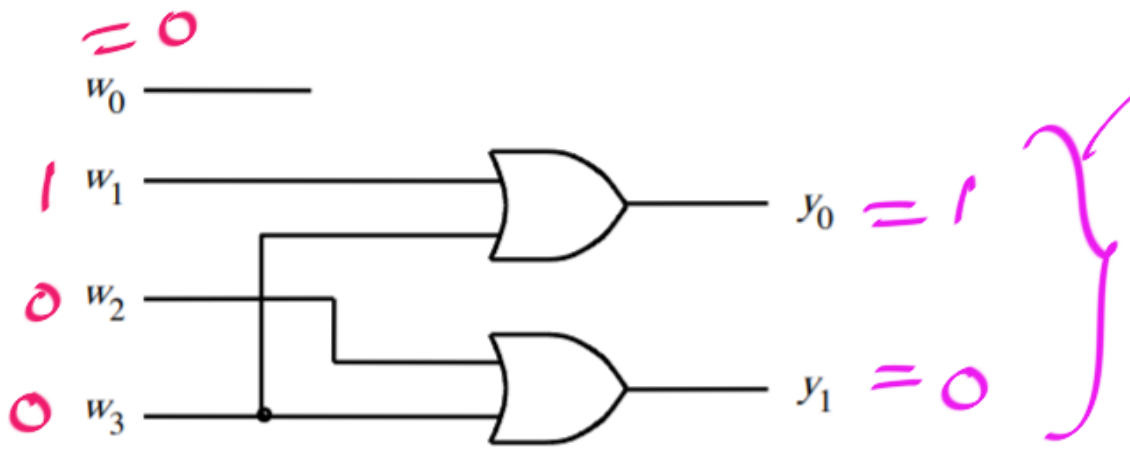
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$Y_1 = w_3 + w_2$

$= \overline{w_0}\ \overline{w_1}$

$Y_0 = w_3 + w_1$

$= \overline{w_0}\ \overline{w_2}$



Encoding of ω_1

Circuit for a 4-to-2 binary encoder

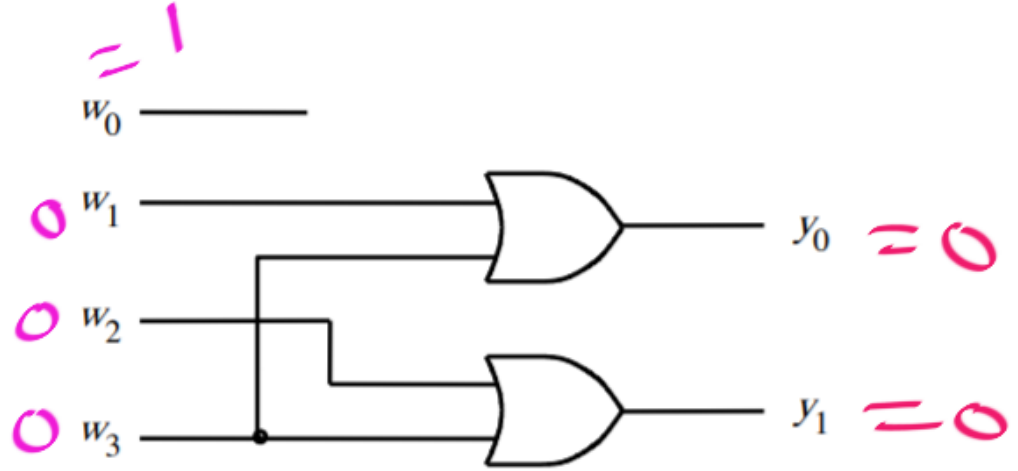
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$Y_1 = w_3 + w_2$

$= \overline{w_0}\ \overline{w_1}$

$Y_0 = w_3 + w_1$

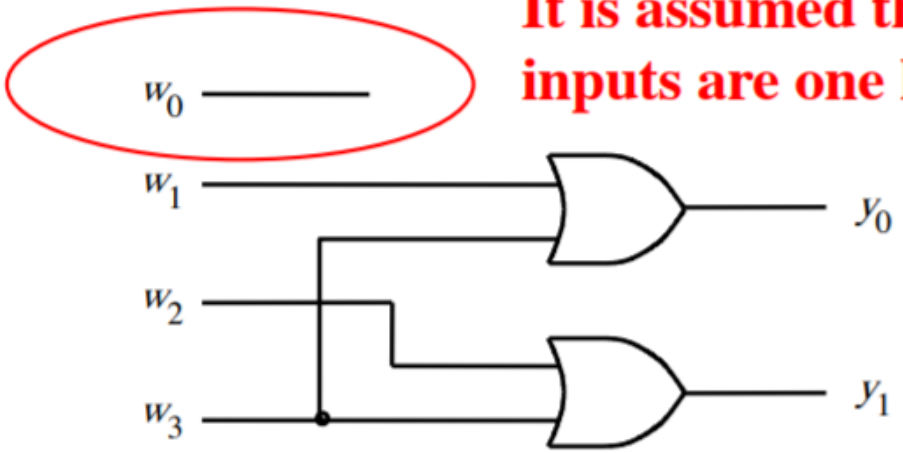
$= \overline{w_0}\ \overline{w_2}$



Encoding of w_3

Circuit for a 4-to-2 binary encoder

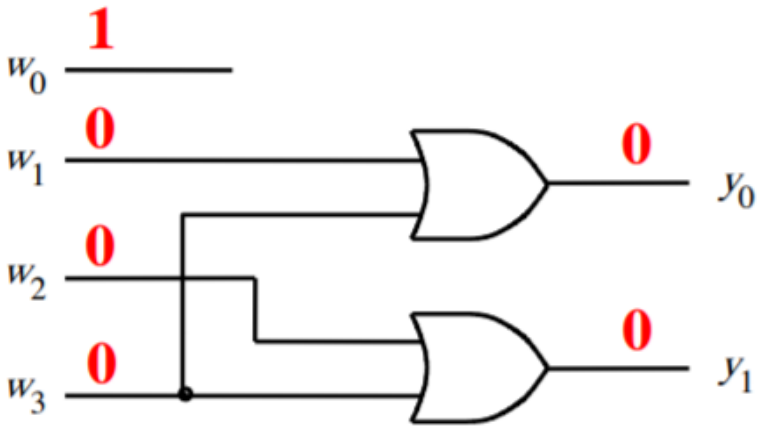
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



It is assumed that the inputs are one hot encoded

Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



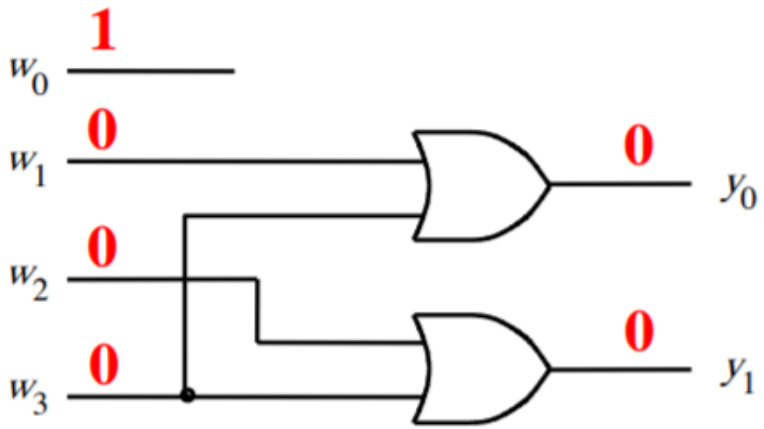
Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

As this level of abstraction we need that w_0 input for this to be a proper 4-to-2 binary encoder.

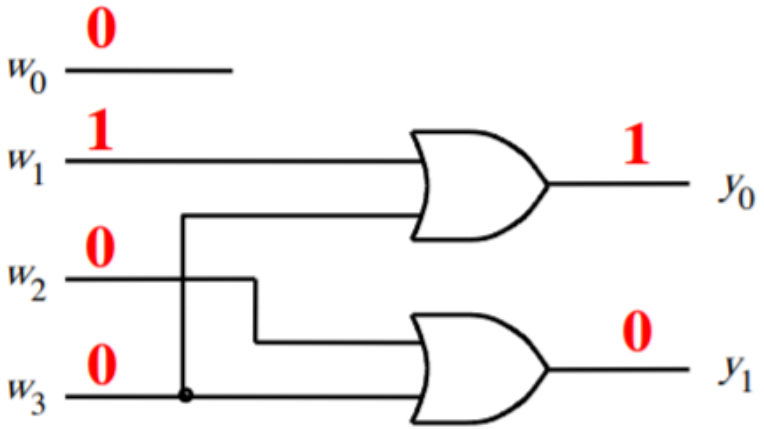
Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



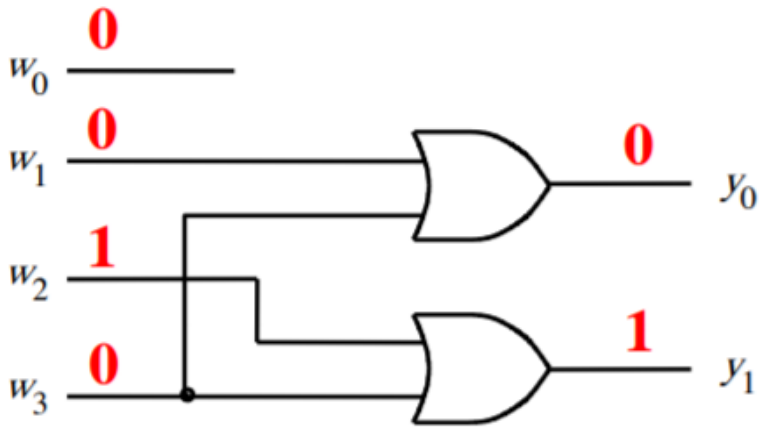
Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



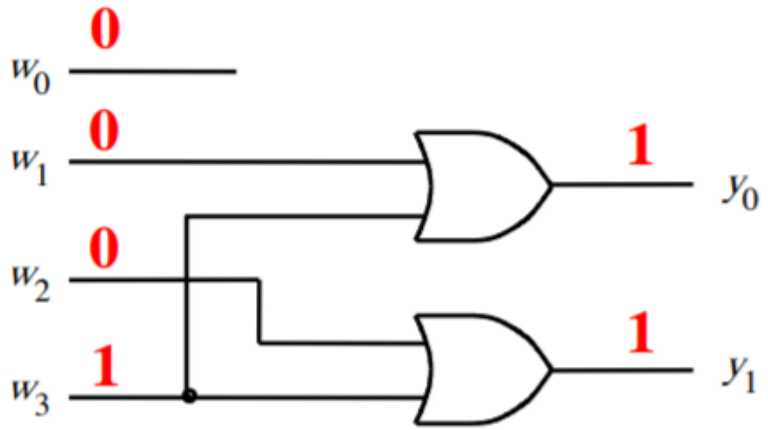
Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



Expressions for 4-to-2 binary encoder

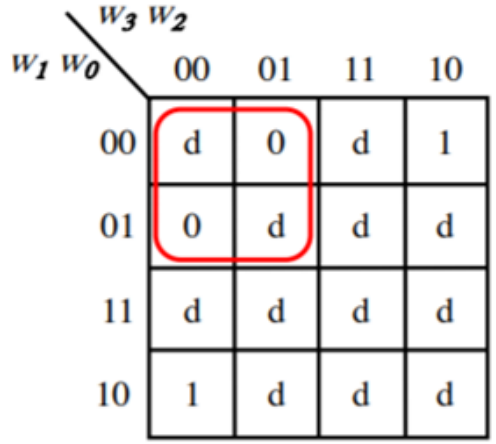
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0		
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1		
0	1	0	0	1	0
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0	1	1
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

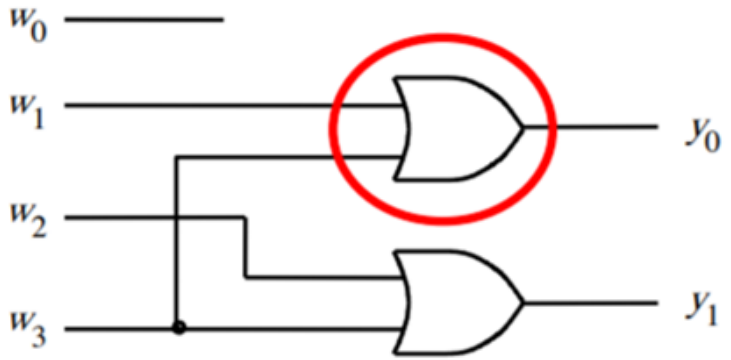
Valid input Combinations

Expressions for 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

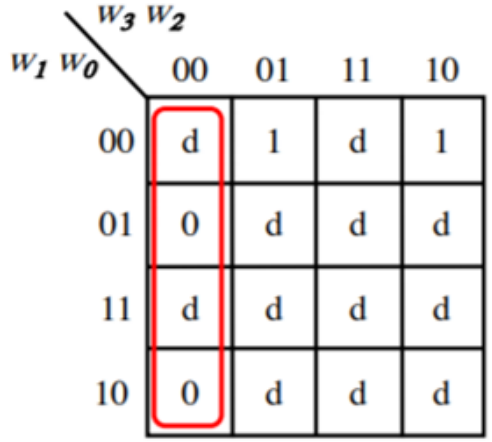


$$y_0 = (w_1 + w_3)$$

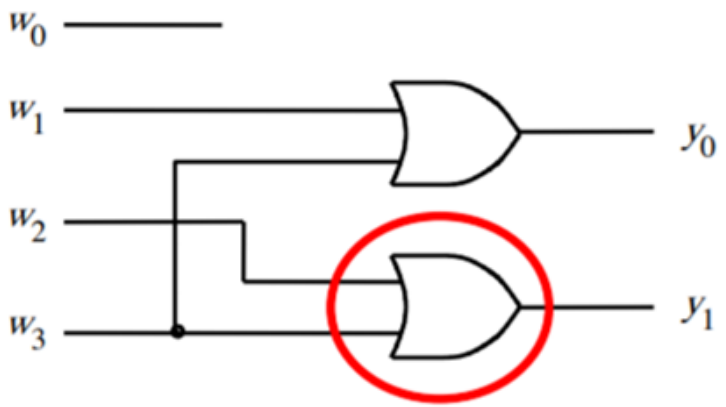


Expressions for 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d



$$y_1 = (w_3 + w_2)$$



Encoder :

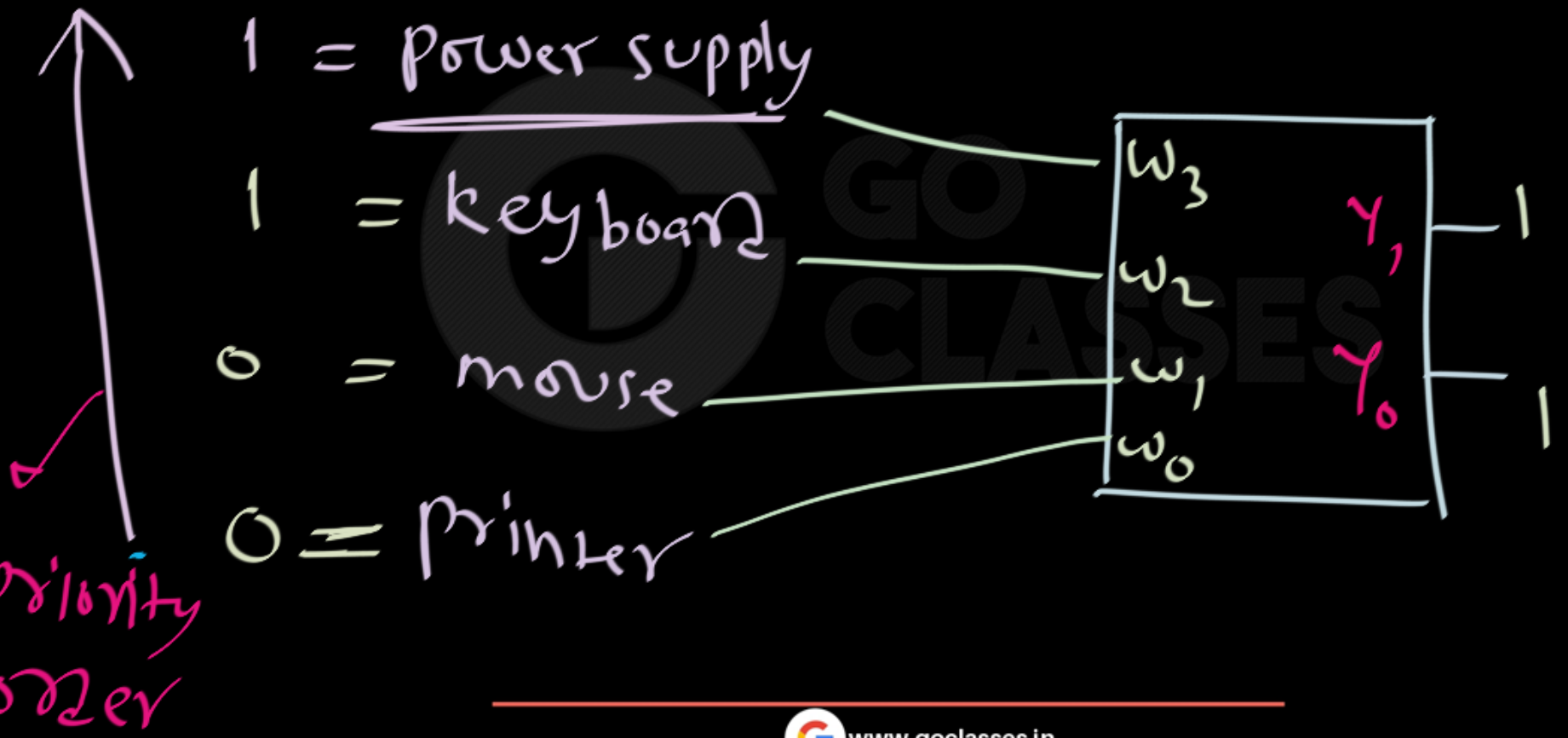
Encoding

Next Topic:

Priority Encoders

I/O Devices

CPU



I/O Devices

highest priority

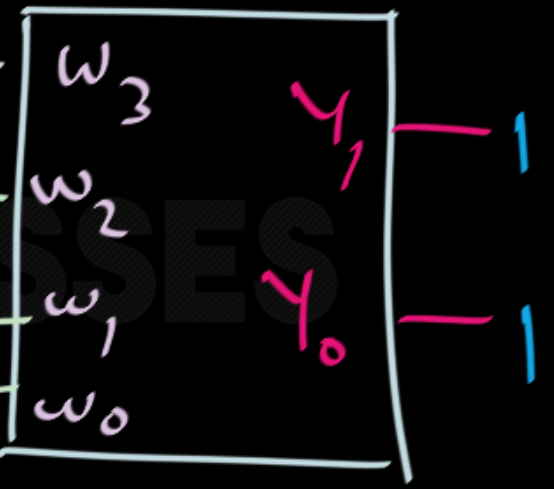
CPU

1 = Power supply

1 = keyboard

1 = mouse

1 = Printer



lowest priority

I/O Devices

highest priority

CPU

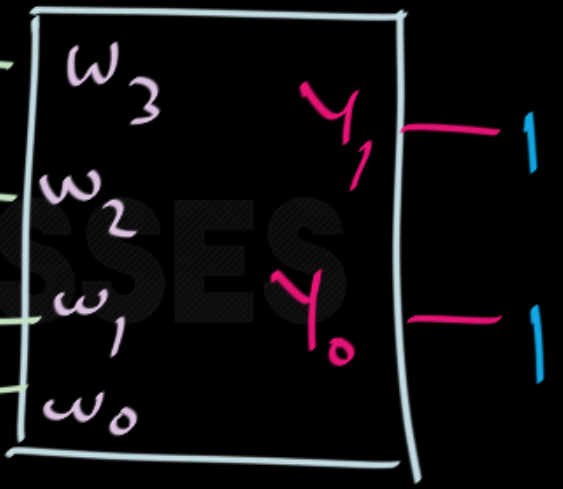
1 = Power supply

x = keyboard

x = mouse

x = Printer

Don't cares



lowest priority

I/O Devices

highest priority

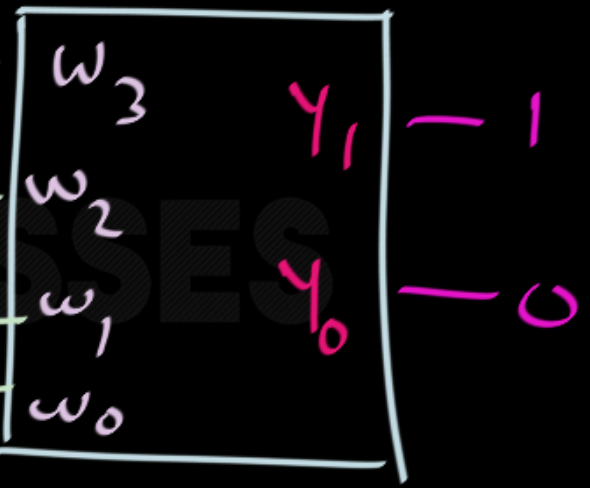
CPU

0 = Power supply

1 = keyboard

1 = mouse

0 = Printer



lowest priority

I/O Devices

highest priority

CPU

0 = Power supply

1 = keyboard

{ x = mouse
{ x = printer

Don't Care

lowest priority

By Default;

w_0 = lowest priority

w_3 = Highest "

} for Priority Encoder.

I/O Devices

highest priority

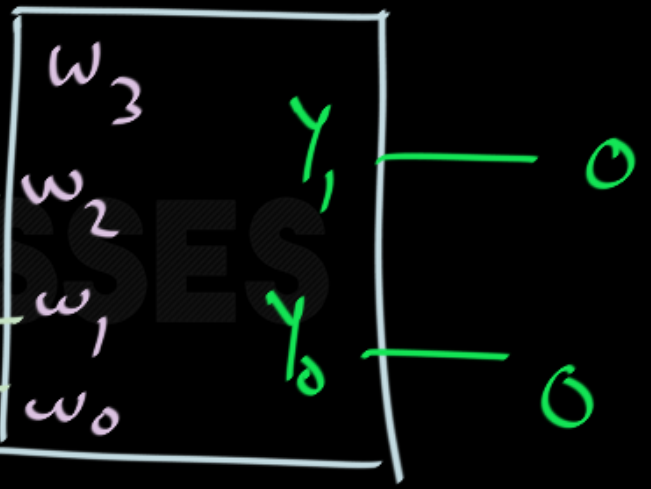
0 = power supply

0 = keyboard

0 = mouse

1 = printer

lowest priority



I/O Devices

highest priority

CPU

0 = power supply

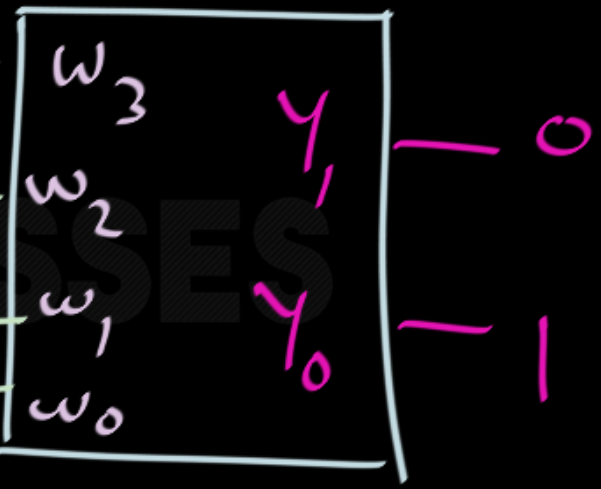
0 = keyboard

1 = mouse

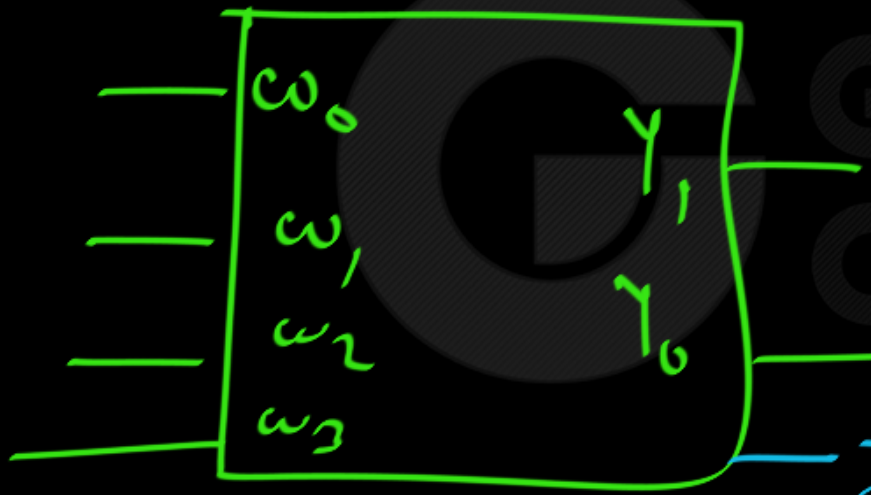
X = Printer

Don't care

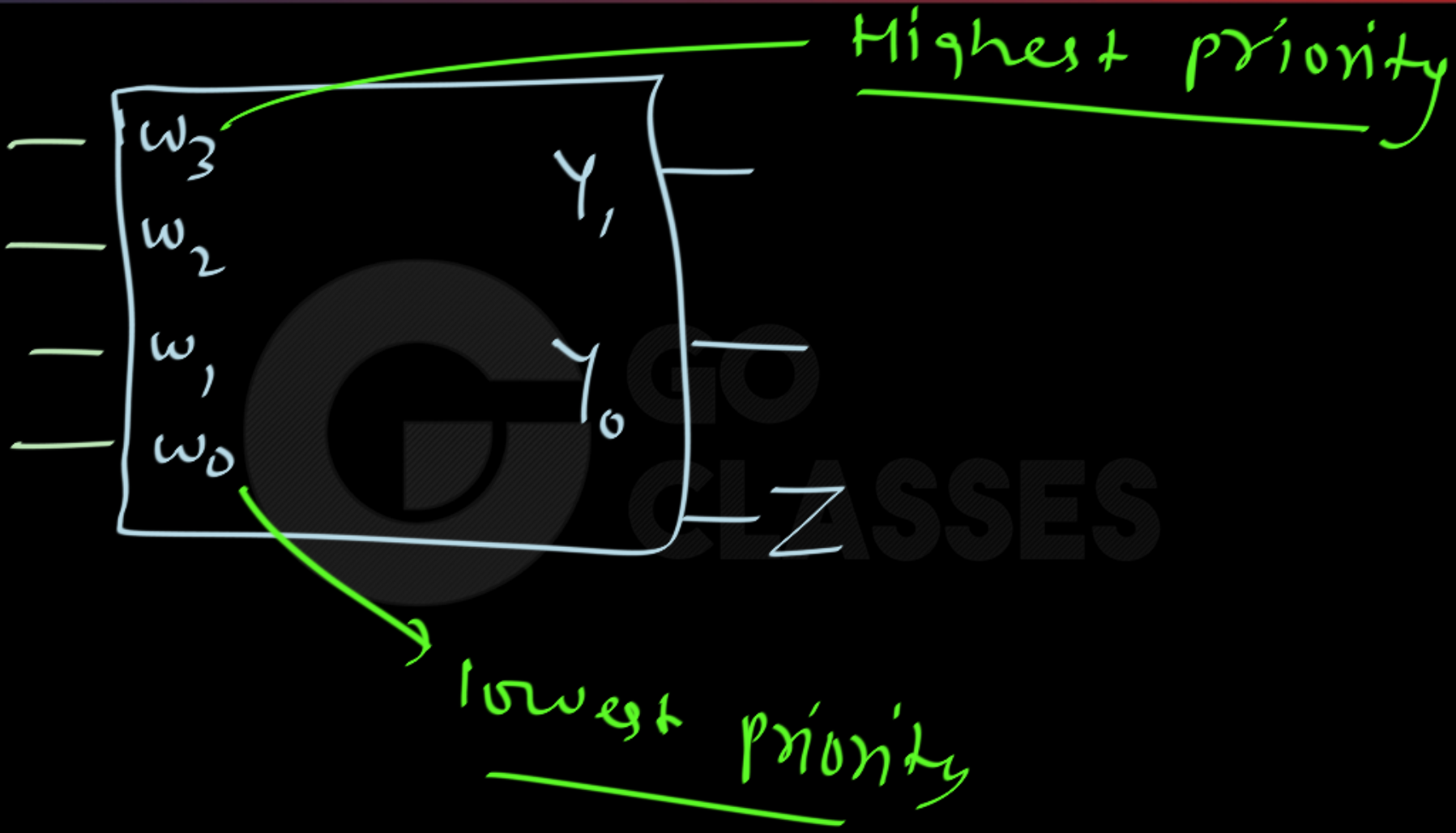
lowest priority



Priority Encoder :→ One Additional output



→ Someone needs attention


Highest priority
w_3
w_2
w_1
w_0
Y_1
Y_0
Z
lowest priority

Power Supply

w_3	w_2	w_1	w_0	Y_1	Y_0	Z
1	X	X	X	1	1	1
0	1	X	X	1	0	1
0	0	1	X	0	1	1
0	0	0	1	0	0	1
0	0	0	0	0	0	0

$$Y_1 = \omega_3 + \overline{\omega_3}\,\omega_2 = \omega_3 + \omega_2$$

$$Y_0 = \omega_3 + \overline{\omega_3}\,\overline{\omega_2}\,\omega_1$$

$$Y_0 = \omega_3 + \overline{\omega_2}\,\omega_1$$

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

Z ≠ Enable input

Special o/p

input

in Priority Encoder

$Z = 1$ iff at least one input is 1.

$$\overbrace{Z} = \boxed{\omega_3 + \omega_2 + \omega_1 + \omega_0}$$

4-to-2 Priority Encoder (Definition)

- Has four inputs: w_3 , w_2 , w_1 , and w_0
- Has two primary outputs: y_1 and y_0
- Has one other output: z

- The inputs are NOT "one-hot" encoded.
- More than one input can be set to 1 but they have priorities associated with them: w_3 – highest priority and w_0 – lowest priority.

- y_1=0 and y_0=0 (if w_0=1 and w_3=w_2=w_1=0)
- y_1=0 and y_0=1 (if w_1=1 and w_3=w_2=0)
- y_1=1 and y_0=0 (if w_2=1 and w_3=0)
- y_1=1 and y_0=1 (if w_3=1)

4-to-2 Priority Encoder (Definition)

- Has four inputs: w_3 , w_2 , w_1 , and w_0
- Has two primary outputs: y_1 and y_0
- Has one other output: z

- The inputs are NOT "one-hot" encoded.
- More than one input can be set to 1 but they have priorities associated with them: w_3 – highest priority and w_0 – lowest priority.

- y_1=0 and y_0=0 (if w_0=1 and w_3=w_2=w_1=0)
- y_1=0 and y_0=1 (if w_1=1 and w_3=w_2=0) w_0
- y_1=1 and y_0=0 (if w_2=1 and w_3=0) w_0 , w_1
- y_1=1 and y_0=1 (if w_3=1) w_0 , w_1 , w_2

these have lower priorities and can be either 0 or 1.

4-to-2 Priority Encoder (Definition)

- Has four inputs: w_3 , w_2 , w_1 , and w_0
- Has two primary outputs: y_1 and y_0
- Has one other output: z

- The inputs are NOT "one-hot" encoded.
- More than one input can be set to 1 but they have priorities associated with them: w_3 – highest priority and w_0 – lowest priority.

- y_1=0 and y_0=0 (if w_0=1 and w_3=w_2=w_1=0)
- y_1=0 and y_0=1 (if w_1=1 and w_3=w_2=0)
- y_1=1 and y_0=0 (if w_2=1 and w_3=0)
- y_1=1 and y_0=1 (if w_3=1)

- z =0 if w_3=w_2=w_1=w_0=0; otherwise z=1.

Truth table for a 4-to-2 priority encoder (abbreviated version)

'implies all inputs are 0.

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

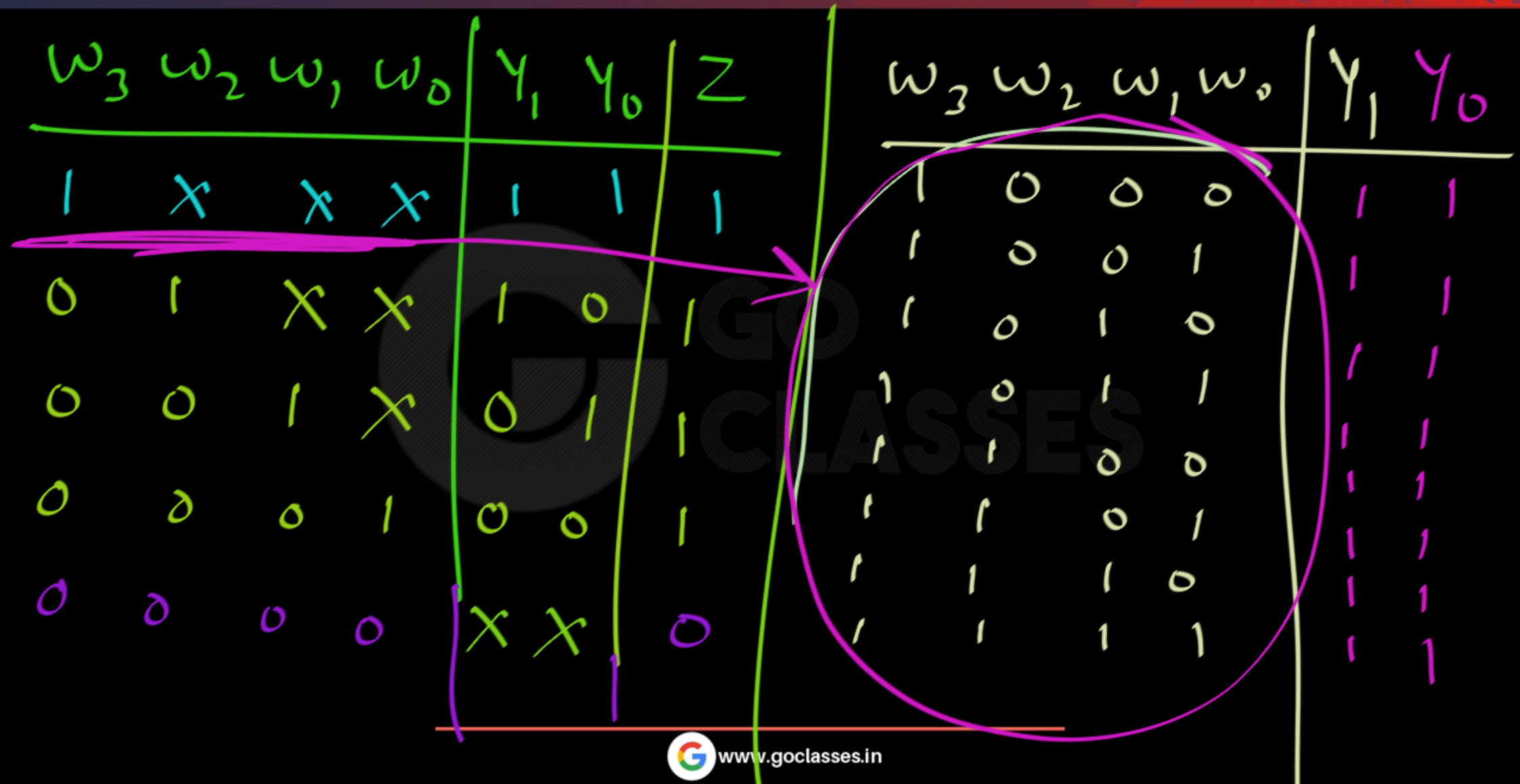
Truth table for a 4-to-2 priority encoder (abbreviated version)

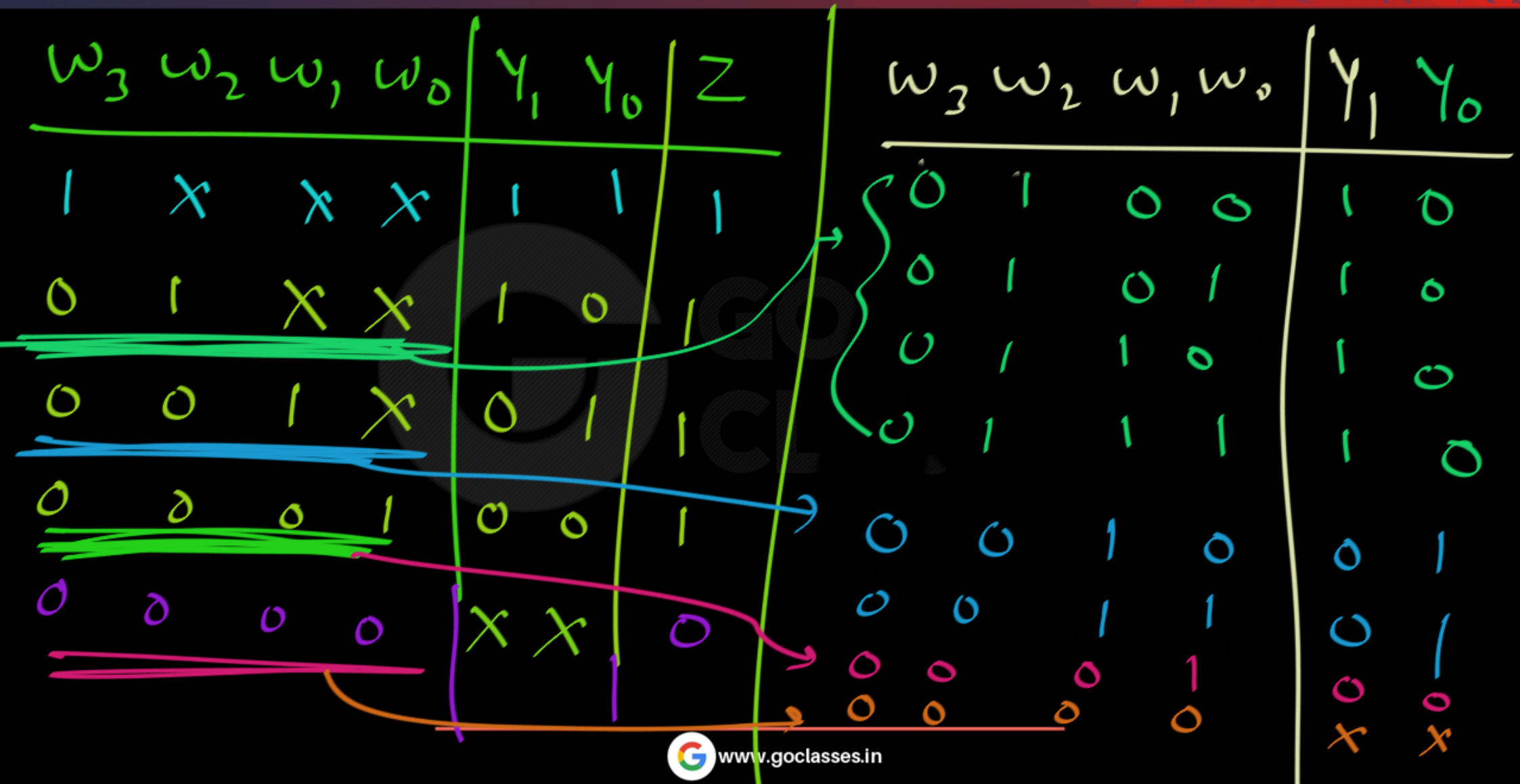
$y_1 = w_3 + w_2$

$y_0 = w_3 + \overline{w}_2 w_1$

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

'implies all inputs are 0.


w_3 w_2 w_1 w_0 Y_1 Y_0 Z
1 X X X 1 1 1
0 1 X X 1 0 1
0 0 1 X 0 1 1
0 0 0 1 0 0 1
0 0 0 0 X X 0
w_3 w_2 w_1 w_0 Y_1 Y_0
1 0 0 0 1 1
1 0 0 1 1 1
1 0 1 0 1 1
1 0 1 1 1 1
1 1 0 0 1 1
1 1 0 1 1 1
1 1 1 0 1 1
1 1 1 1 1 1

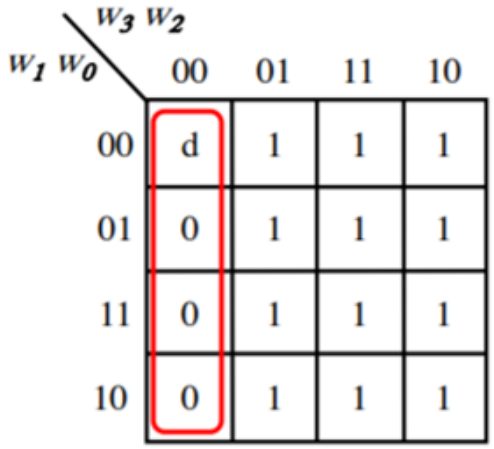

w_3 w_2 w_1 w_0 Y_1 Y_0 Z
1 X X X 1 1 1
0 1 X X 1 0 1
0 0 1 X 0 1 1
0 0 0 1 0 0 1
0 0 0 0 X X 0
w_3 w_2 w_1 w_0 Y_1 Y_0
0 1 0 0 1 0
0 1 0 1 1 0
0 1 1 0 1 0
0 1 1 1 1 0
0 0 1 0 0 1
0 0 1 1 0 1
0 0 0 1 0 1
0 0 0 0 X X

Truth table for a 4-to-2 priority encoder

	w_3	w_2	w_1	w_0	y_1	y_0	z
0 0 0 0	0	0	0	0	d	d	0
0 0 0 1	0	0	0	1	0	0	1
0 0 1 x	0	0	1	0	0	1	1
	0	0	1	1	0	1	1
0 1 x x	0	1	0	0	1	0	1
	0	1	0	1	1	0	1
	0	1	1	0	1	0	1
	0	1	1	1	1	0	1
1 x x x	1	0	0	0	1	1	1
	1	0	0	1	1	1	1
	1	0	1	0	1	1	1
	1	0	1	1	1	1	1
	1	1	0	0	1	1	1
	1	1	0	1	1	1	1
	1	1	1	0	1	1	1
	1	1	1	1	1	1	1

Expressions for 4-to-2 priority encoder

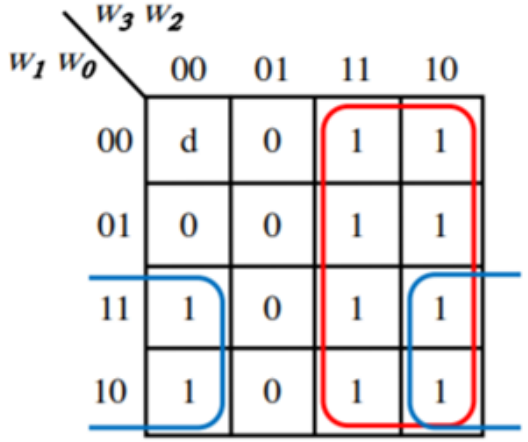
w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1



$$y_1 = w_3 + w_2$$

Expressions for 4-to-2 priority encoder

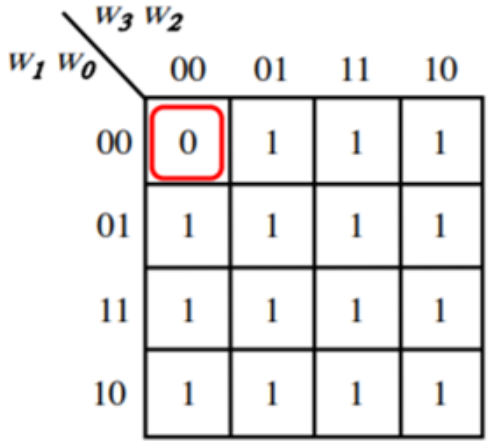
w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1



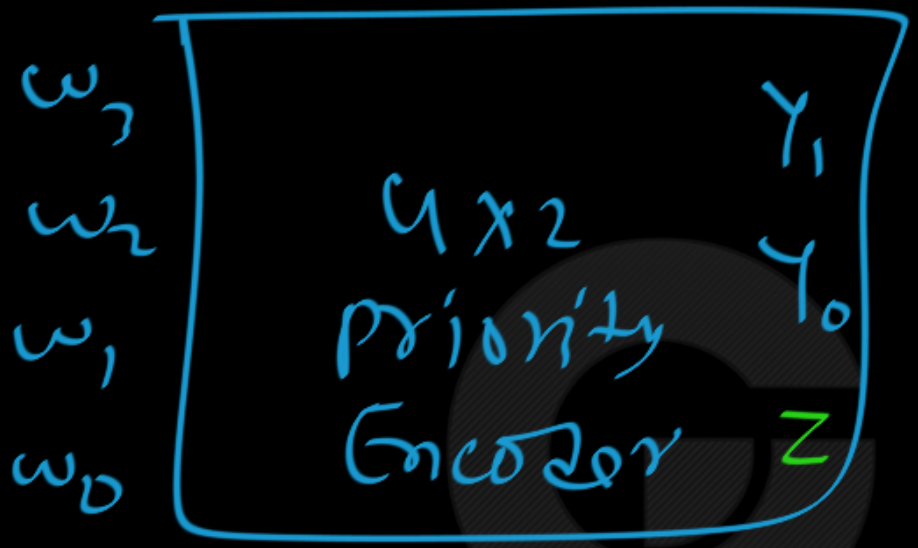
$$y_0 = w_3 + w_1 \overline{w_2}$$

Expressions for 4-to-2 priority encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1



$$Z = W_3 + W_2 + W_1 + W_0$$



$$Y_1 = w_3 + w_2$$

$$Y_0 = w_3 + \overline{w}_2 w_1$$

$$Z = w_0 + w_1 + w_2 + w_3$$

Circuit for the 4-to-2 priority encoder