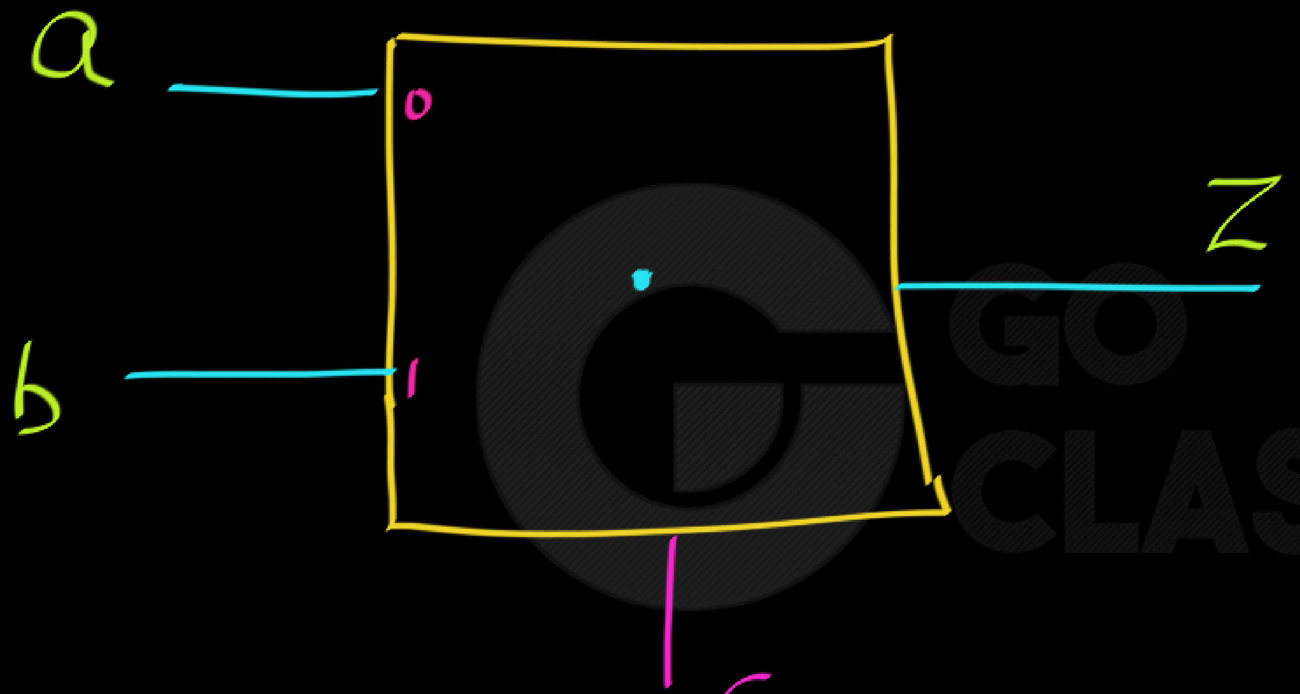
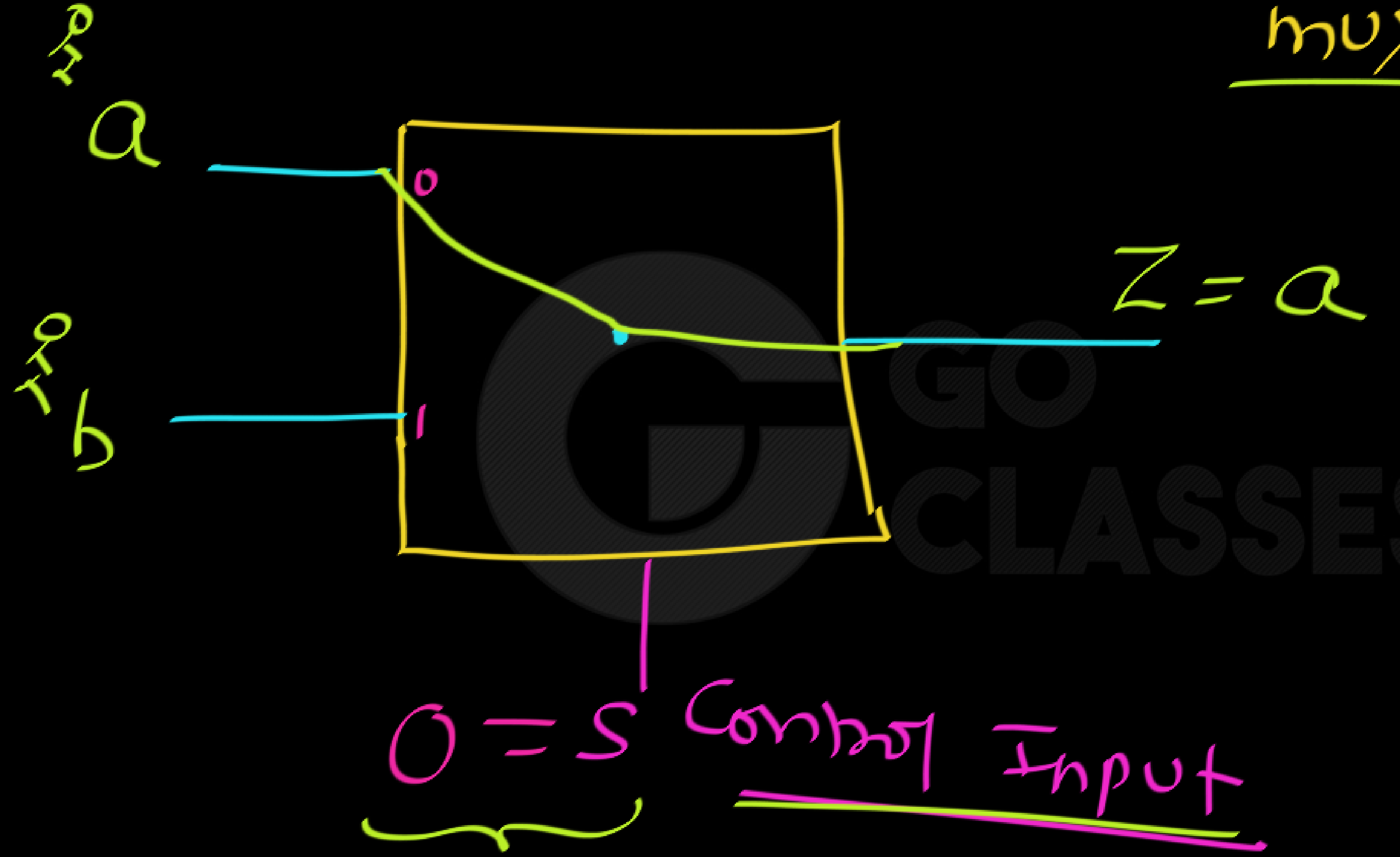
Next Topic:

Multiplexer (mux)



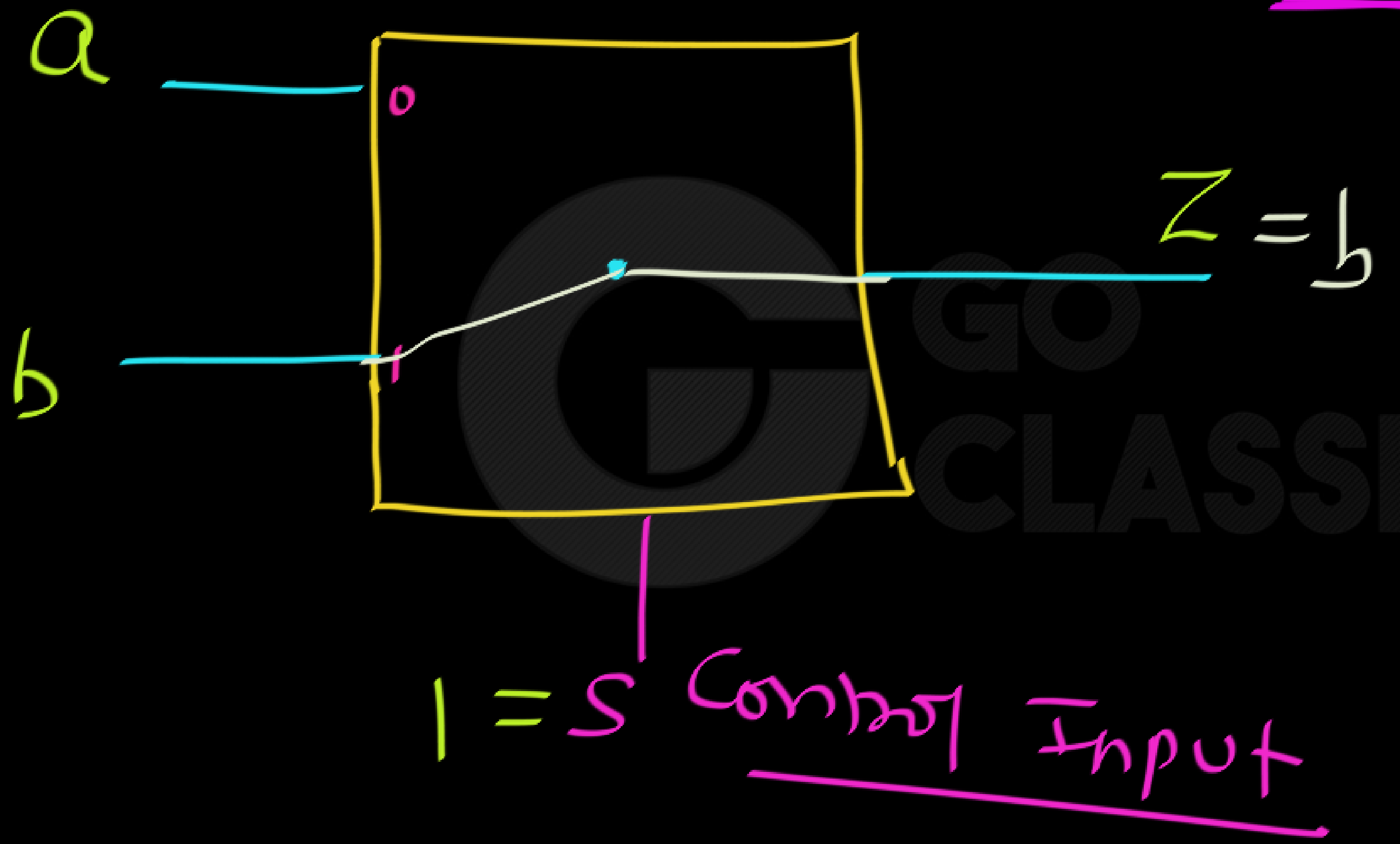
mux is like a Bridge.

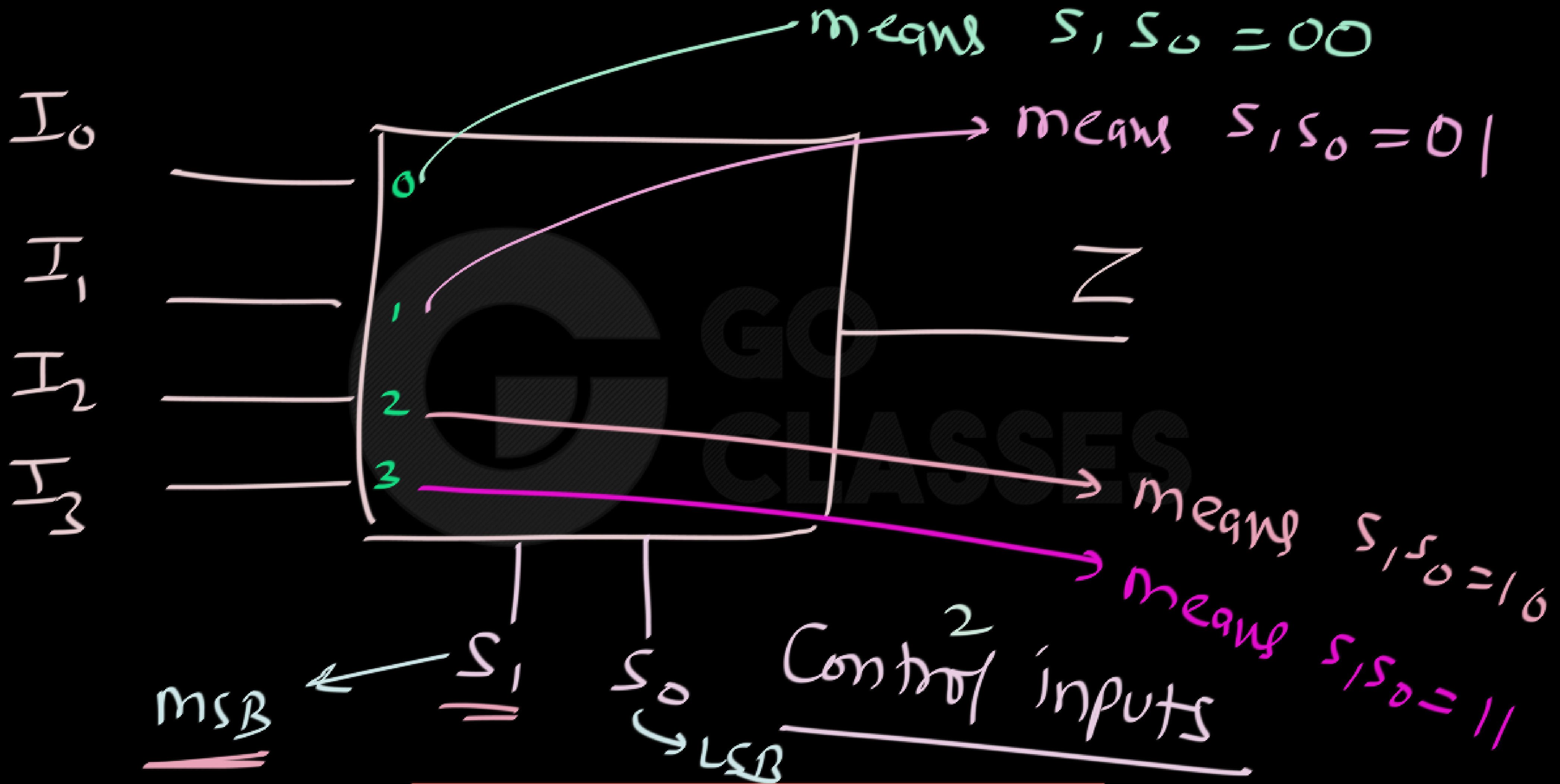
Control Input / Select lines

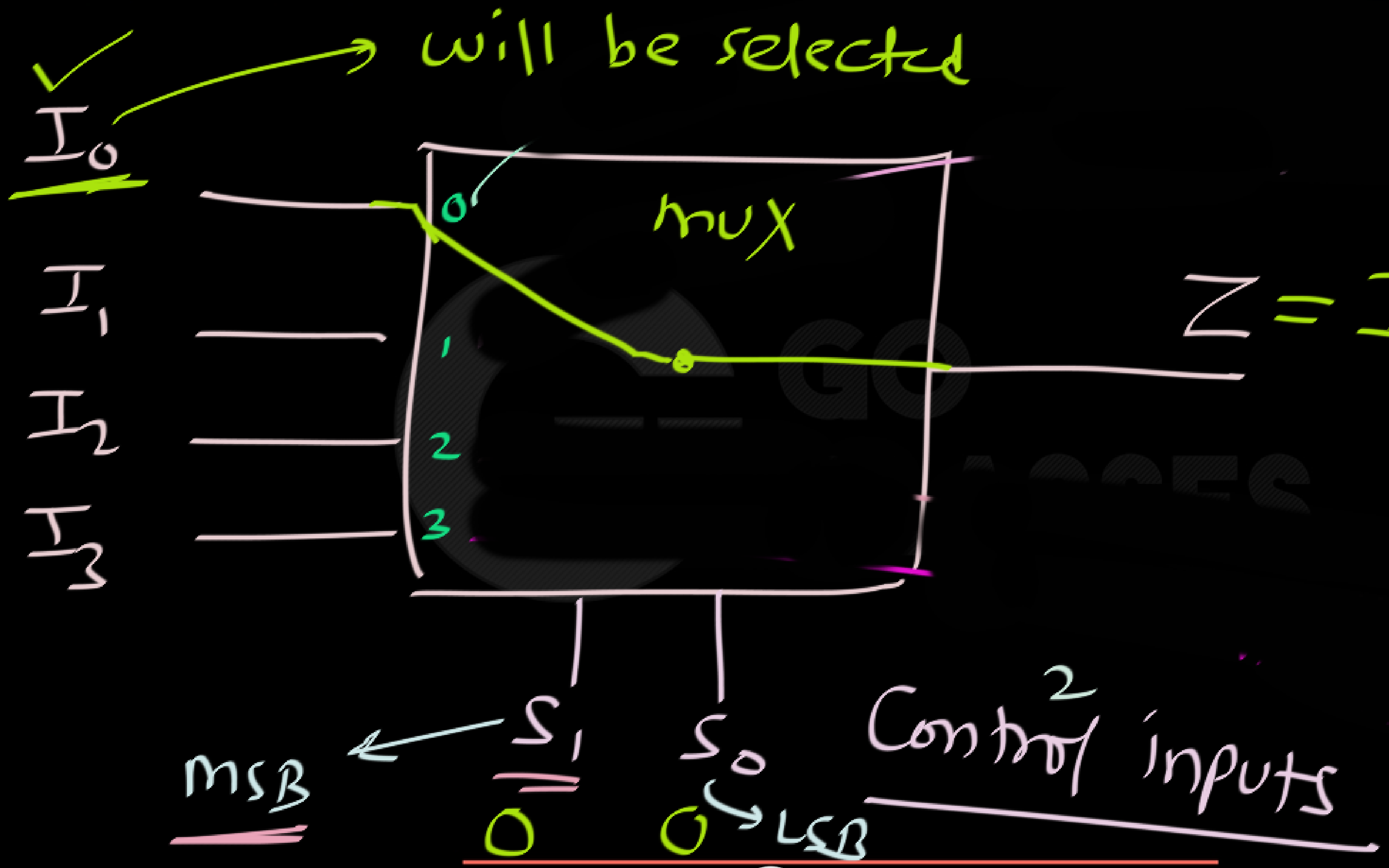


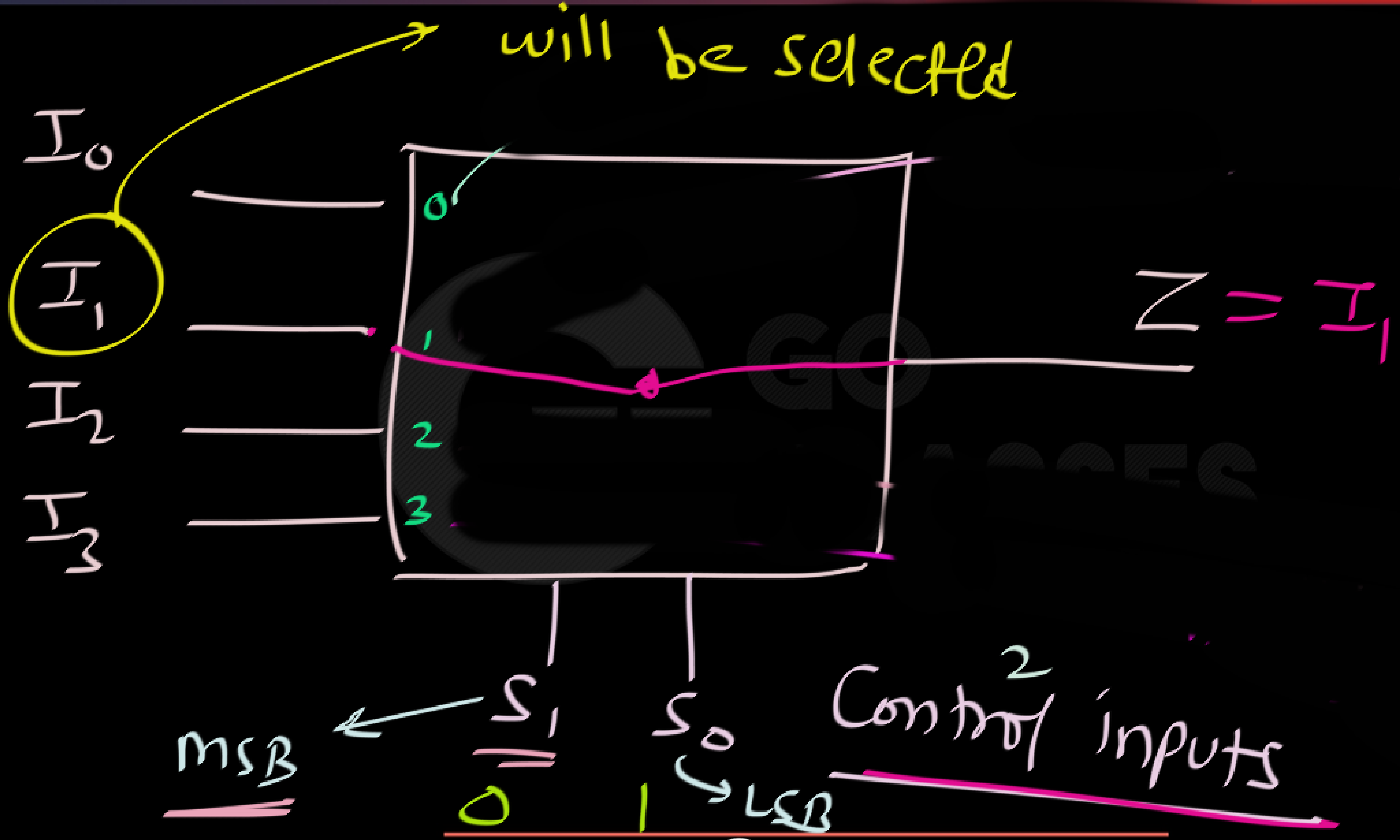
mux is like a Bridge.

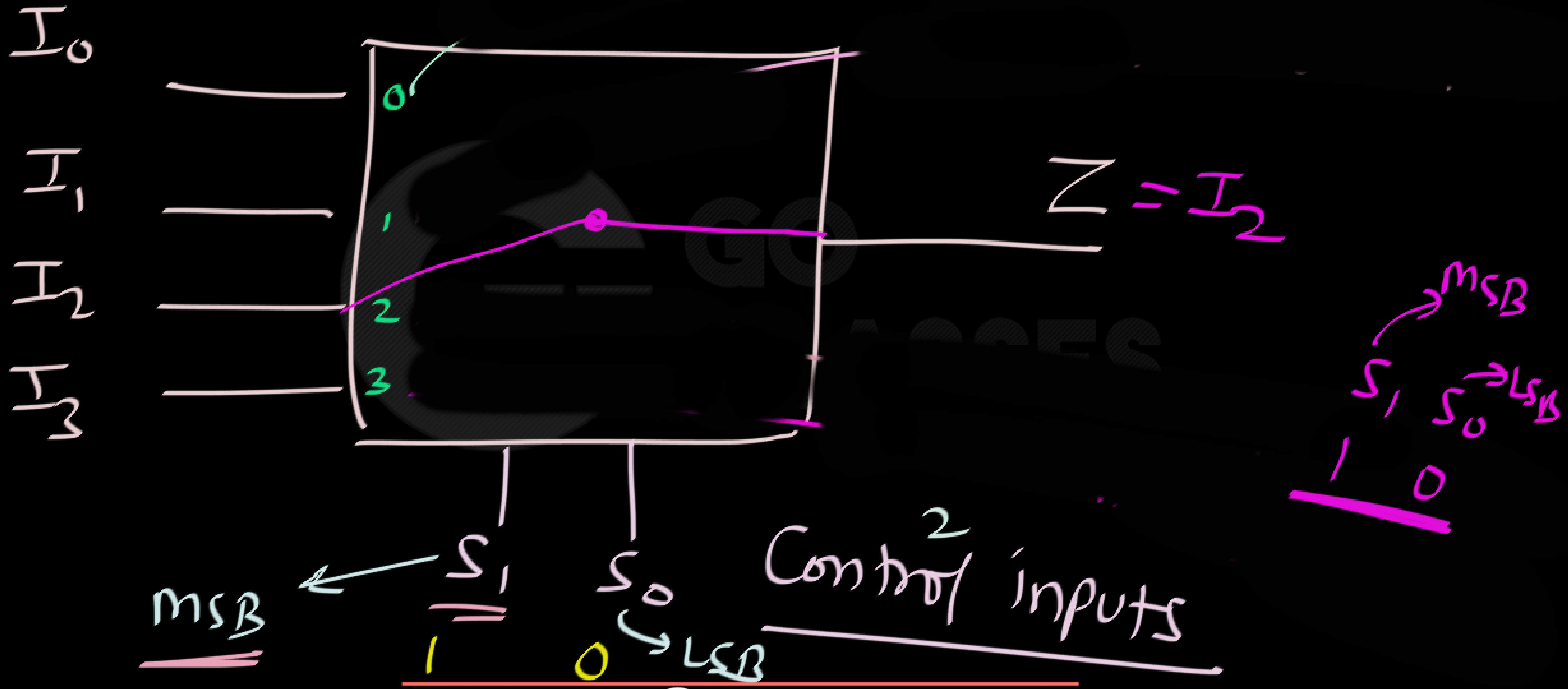
mux is like
a Bridge.

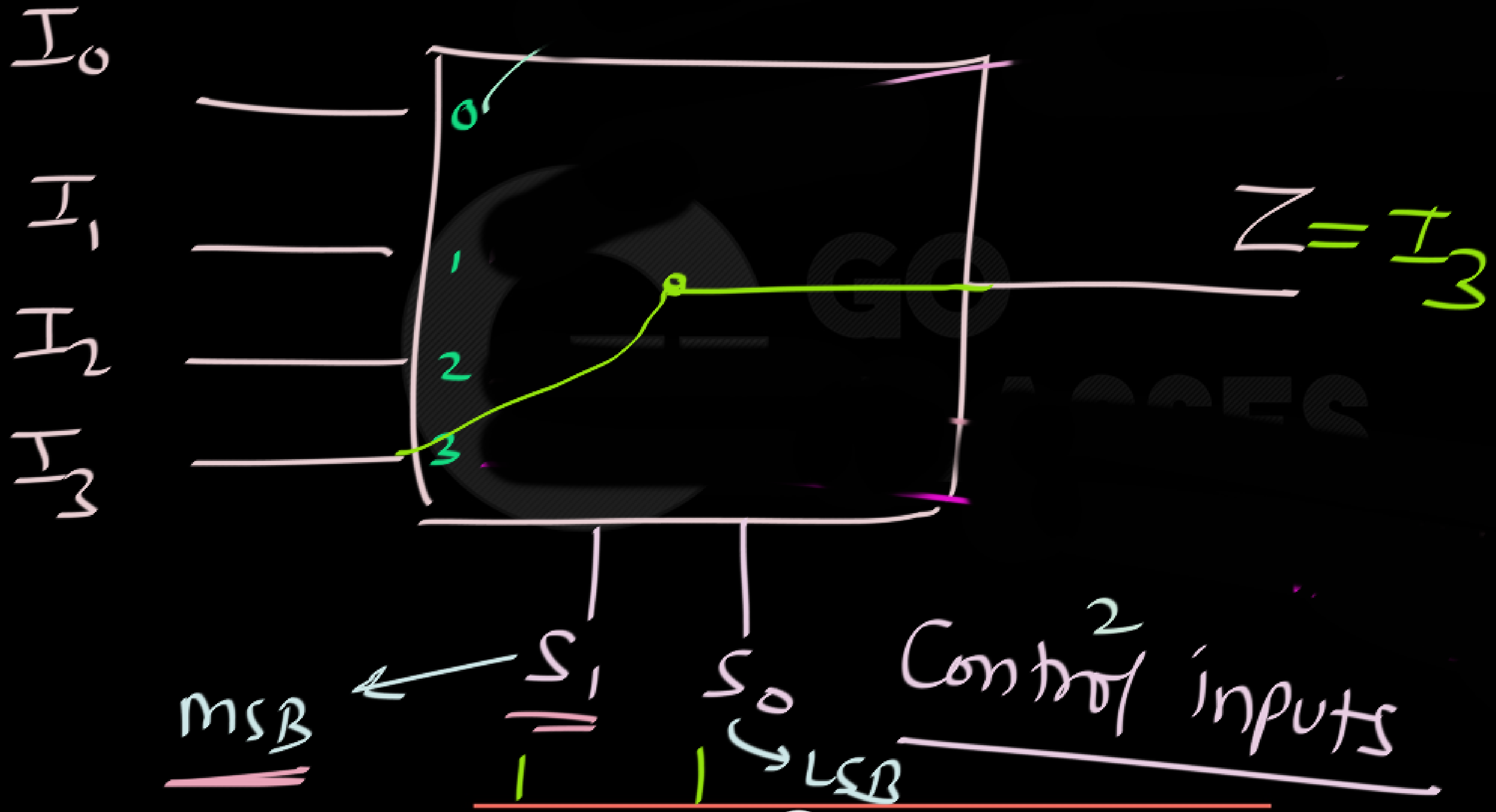



means $S_1 S_0 = 00$
I_0
means $S_1 S_0 = 01$
I_1
Z
I_2
I_3
0
1
2
3
means $S_1 S_0 = 10$
means $S_1 S_0 = 11$
S_1
S_0
MSB
LSB
2
Control inputs


will be selected
I_0
I_1
I_2
I_3
mux
0
1
2
3
$Z = I_0$
S_1
S_0
MSB
LSB
0
0
2 Control inputs

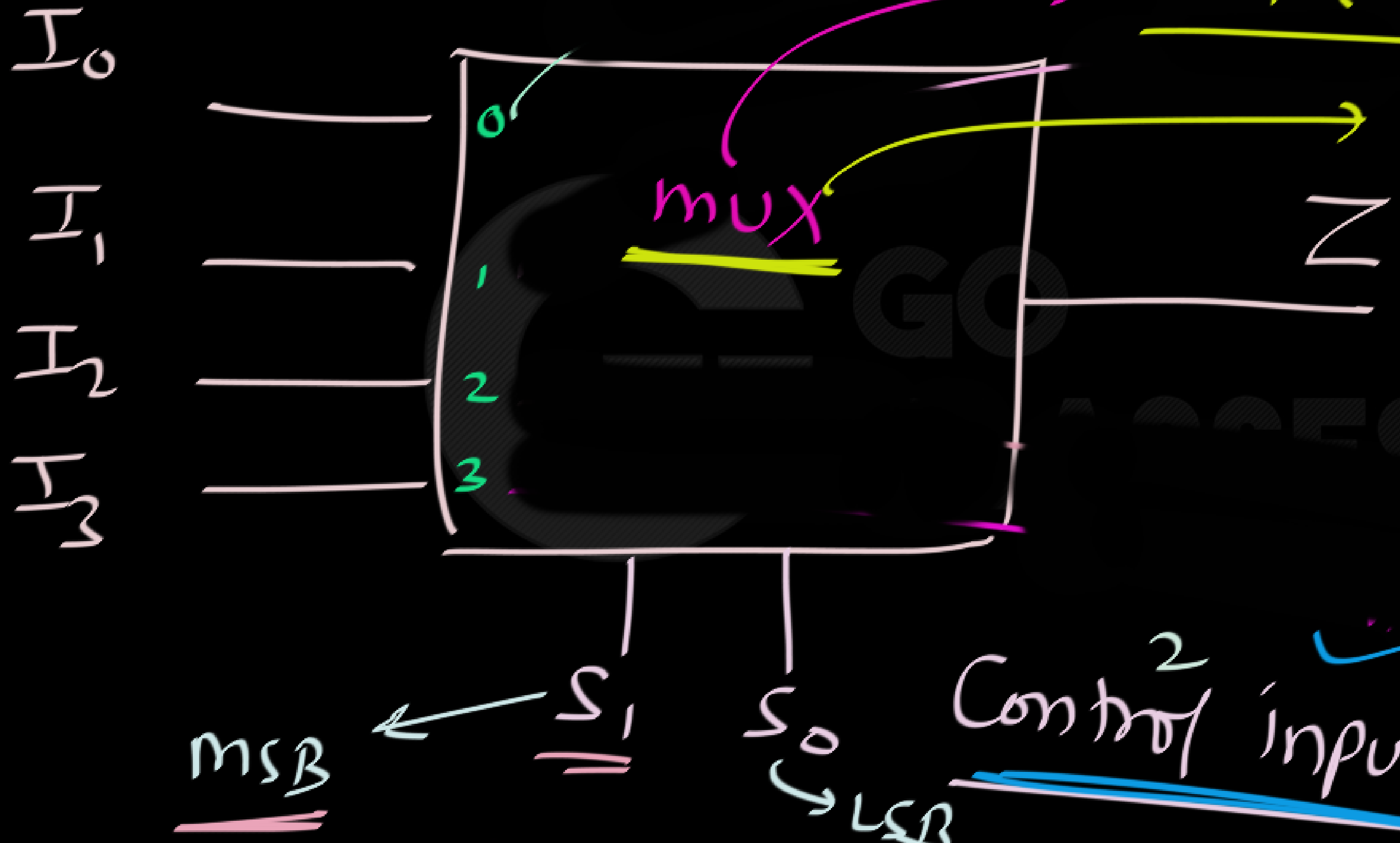

will be selected
I_0
I_1
I_2
I_3
0
1
2
3
$Z = I_1$
S_1
S_0
MSB
LSB
0
1
Control inputs
2


I_0
I_1
I_2
I_3
0
1
2
3
$Z = I_2$
S_1
S_0
MSB
LSB
1
0
S_1 S_0
MSB
LSB
1 0
2 Control inputs


I0
I1
I2
I3
0
1
2
3
Z = I3
S1
S0
MSB
LSB
1
1
Control inputs
2

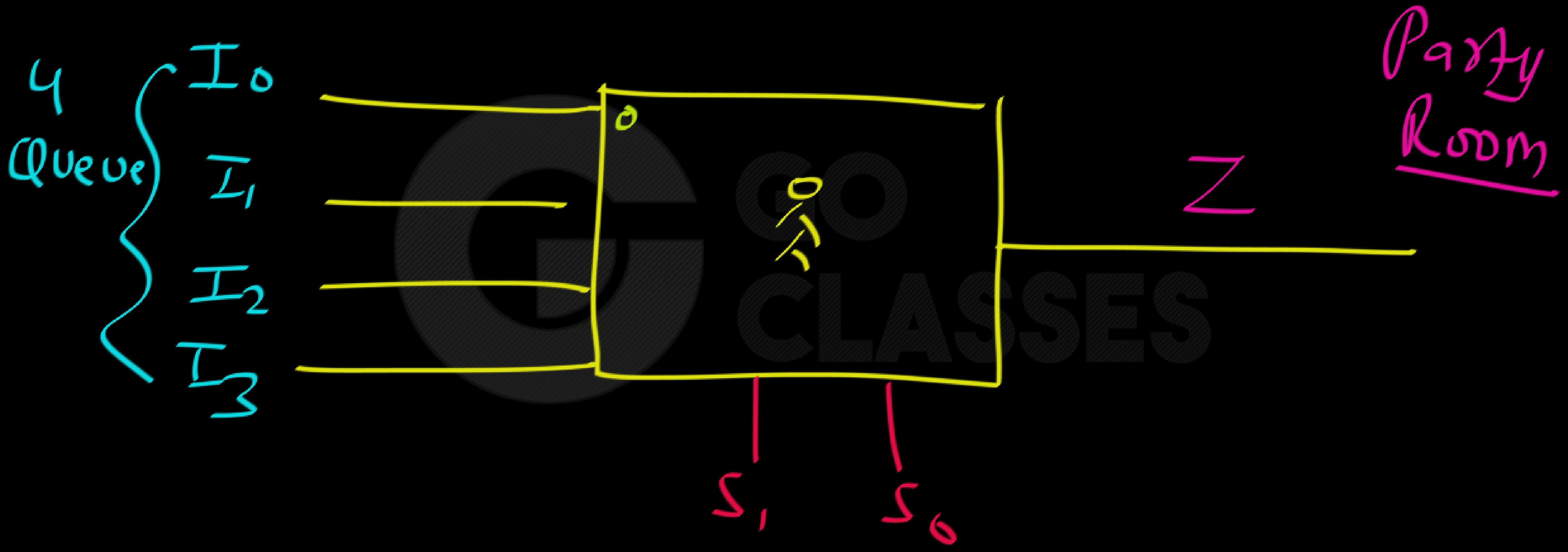
Data selector

works like a bridge

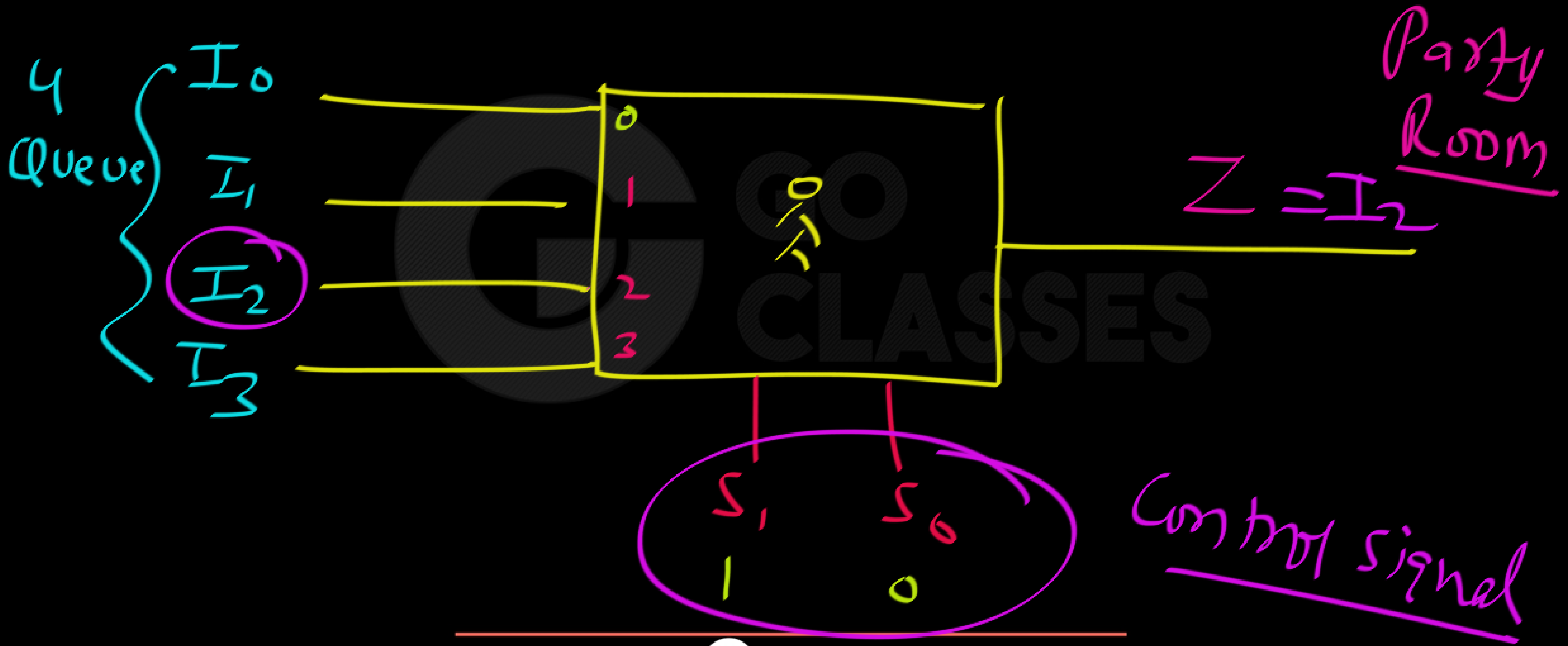


2 Control inputs

Analogy:



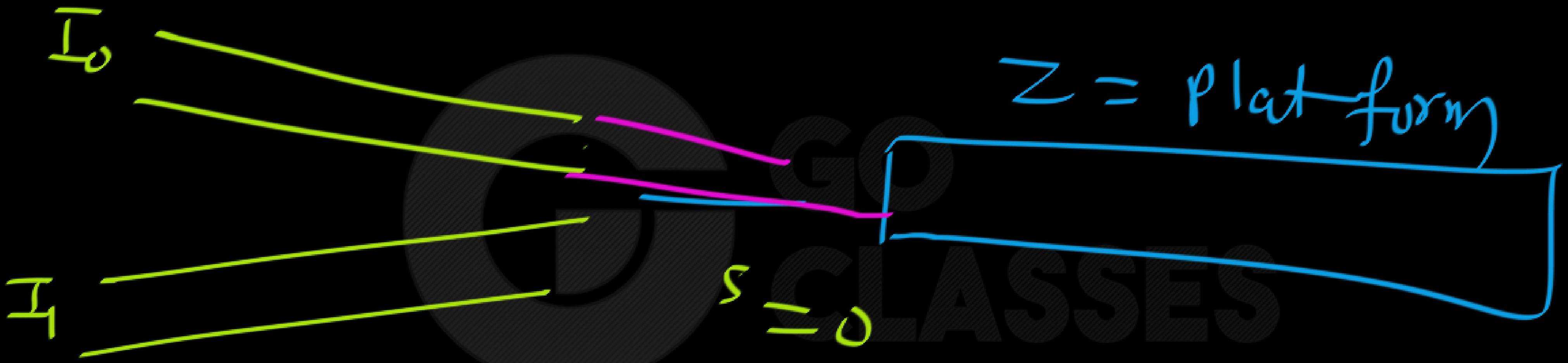
Analogy:

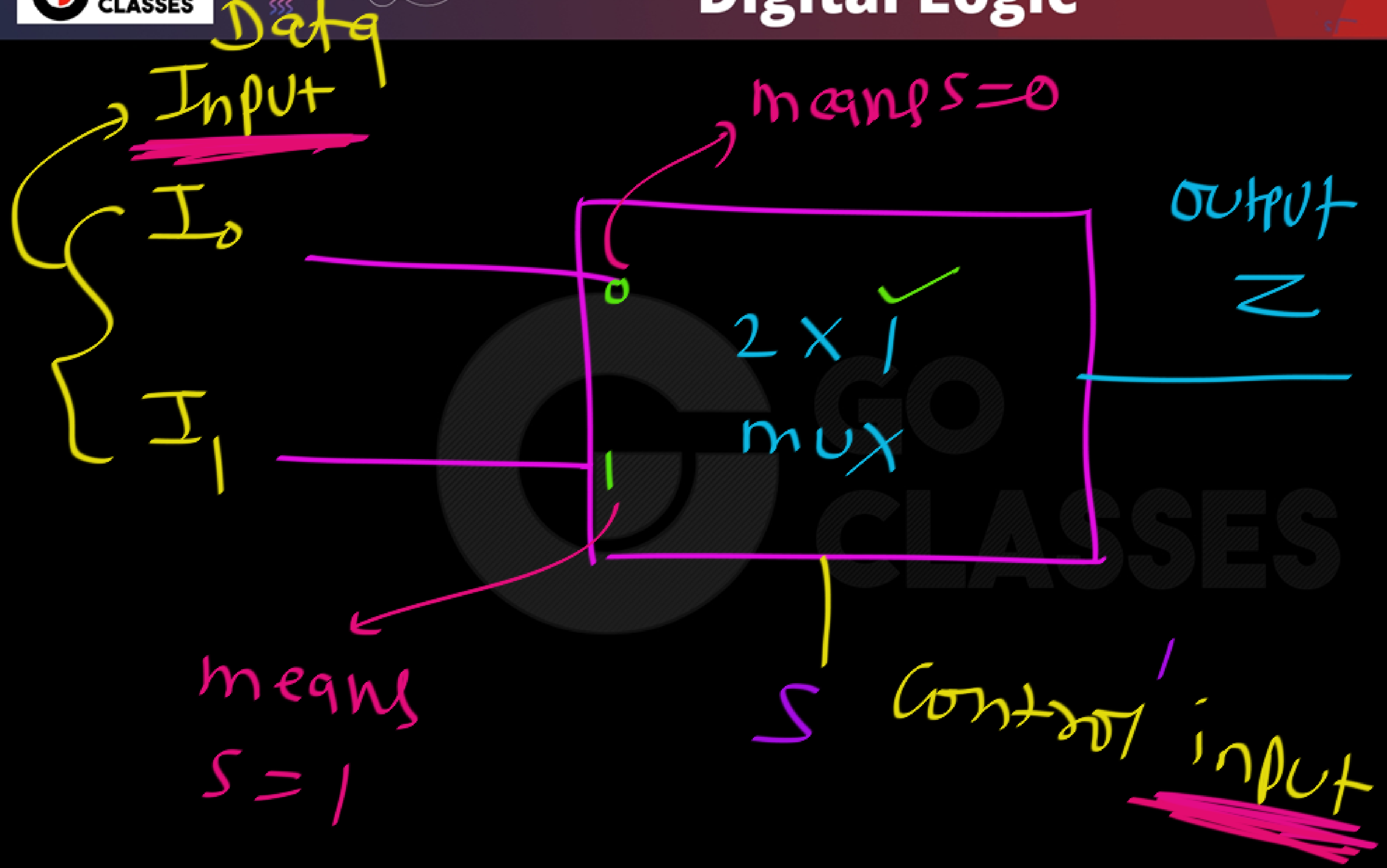


Train;

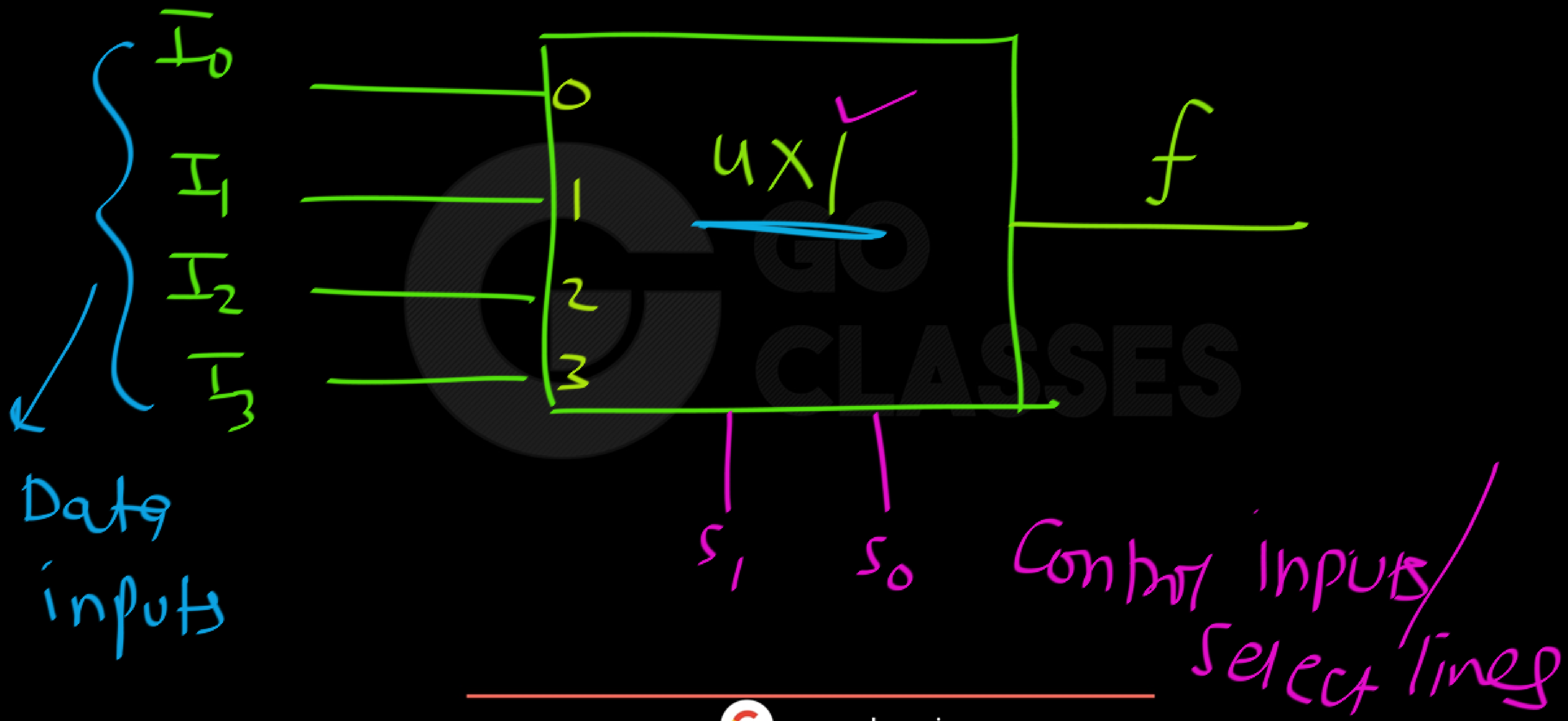
Train ;

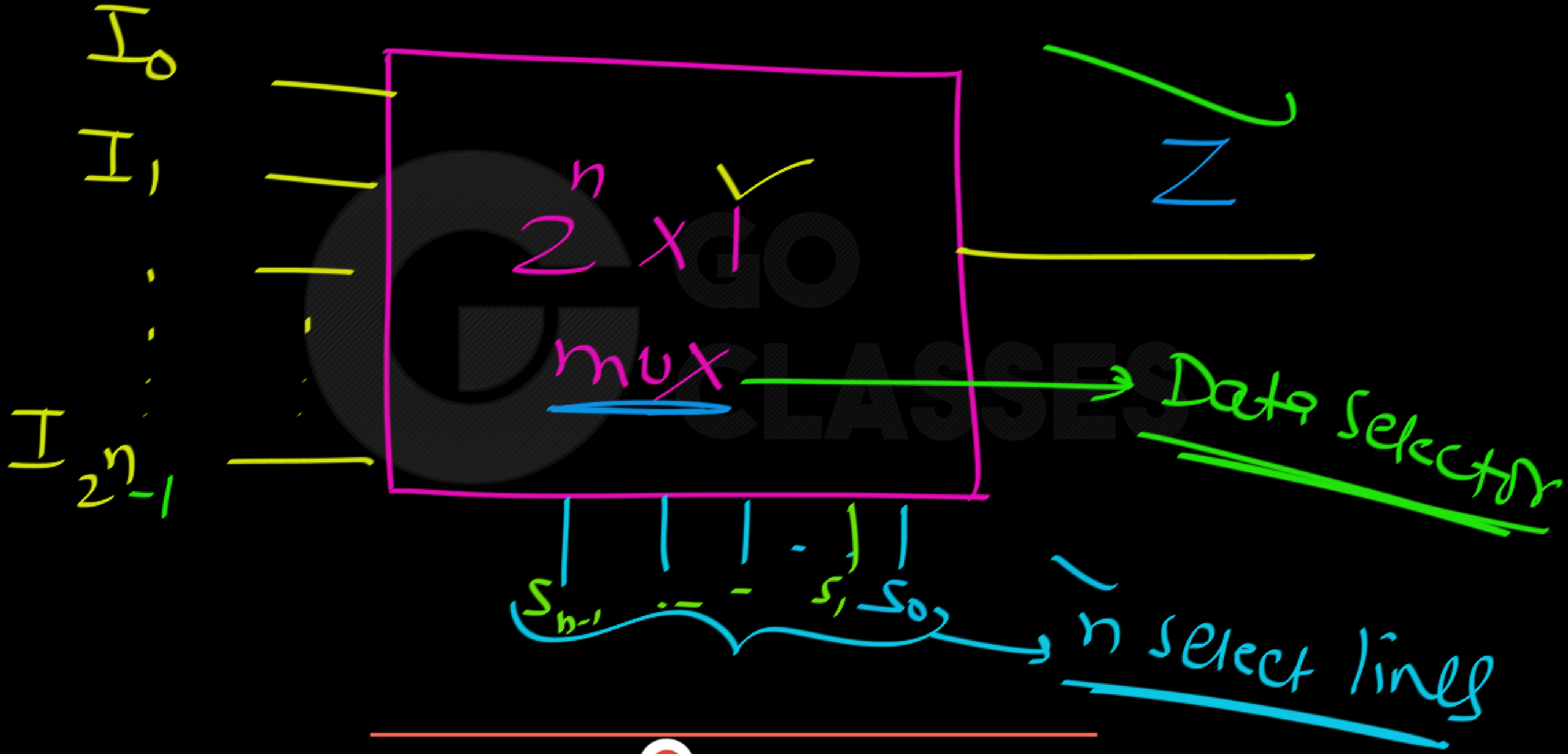
Train;




Data Input
I_0
I_1
means S=0
0
1
2 x 1
mux
output
Z
means
S=1
S
Control input

$$Z = f(S, I_0, I_1)$$


I0
I1
I2
I3
0
1
2
3
4x1
f
Data inputs
S1
S0
Control inputs/
Select lines


I_0
I_1
I_{2^n-1}
$2^n \times 1$
mux
Data selector
Z
S_{n-1} - - S_1 S_0
n select lines

Q: Why No 4x2 mux ?

Why 1 output line only ?

mux

Digital Circuit is Defined like this.

Summer Season: Cooler
Heater
AC
or
Circuits

Heater Heats.

Why Heater Does not cool?
Why AC Does not Heat?

many Digital Circuits:

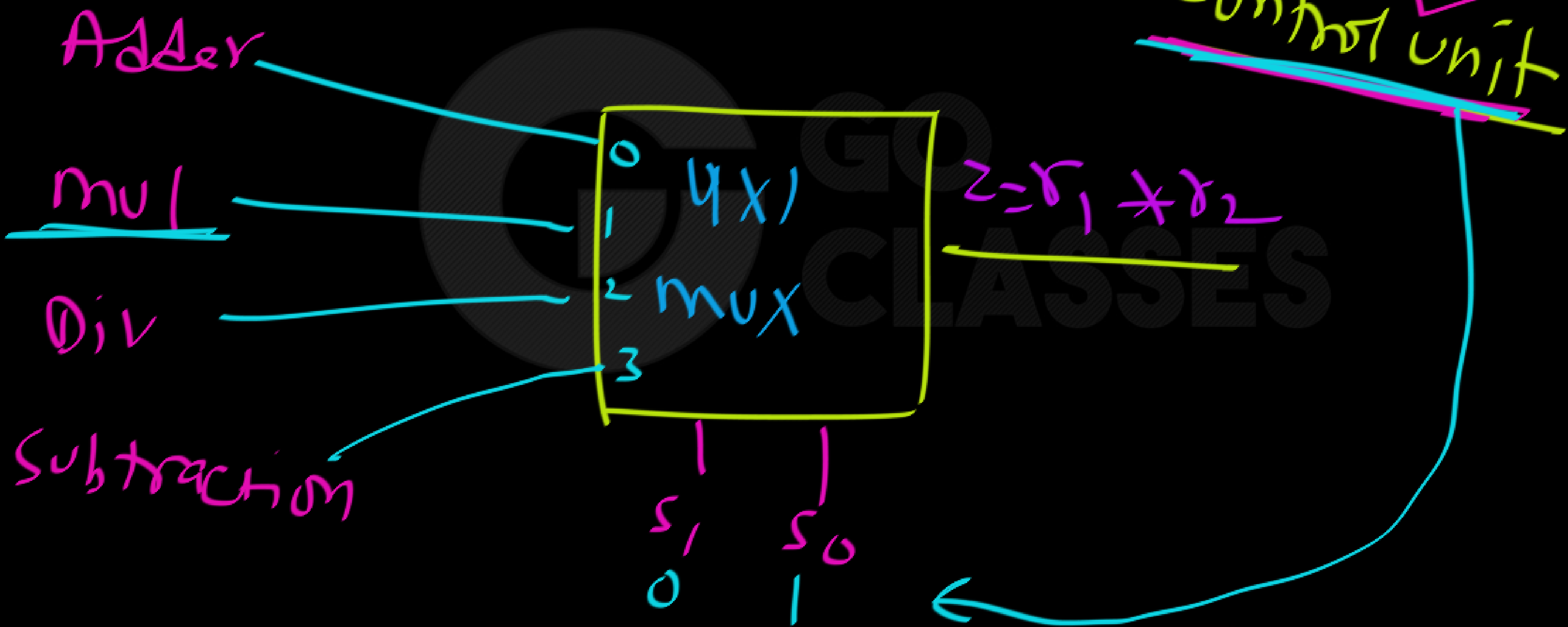
1. mux
2. Decoder
3. Encoder
4. Adder
5. Demux

— - - - . . . — — —

Different uses

Subject COA:

CPU
&
Control unit



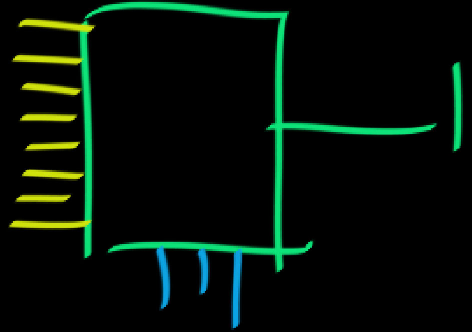
Multiplexers:

It is a combinational circuit which selects one of the 2^n input lines and transmits the information from that line to the output line. The selection of the input line depends upon the 'n' input selection lines. It is also called **data selector** and is also referred to by only **MUX.**

functionality of mux

Smallest MUX we have is 2 to 1 mux which has 2 input lines, 1 output line and 1 selection line. We also have 4 to 1, 8 to 1 mux and so on

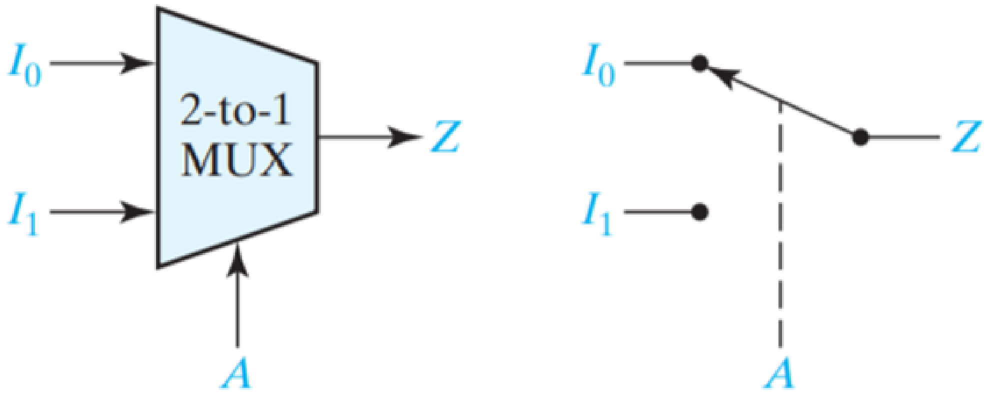
(Data) Input lines = 4
Select lines = 2
output " = 1

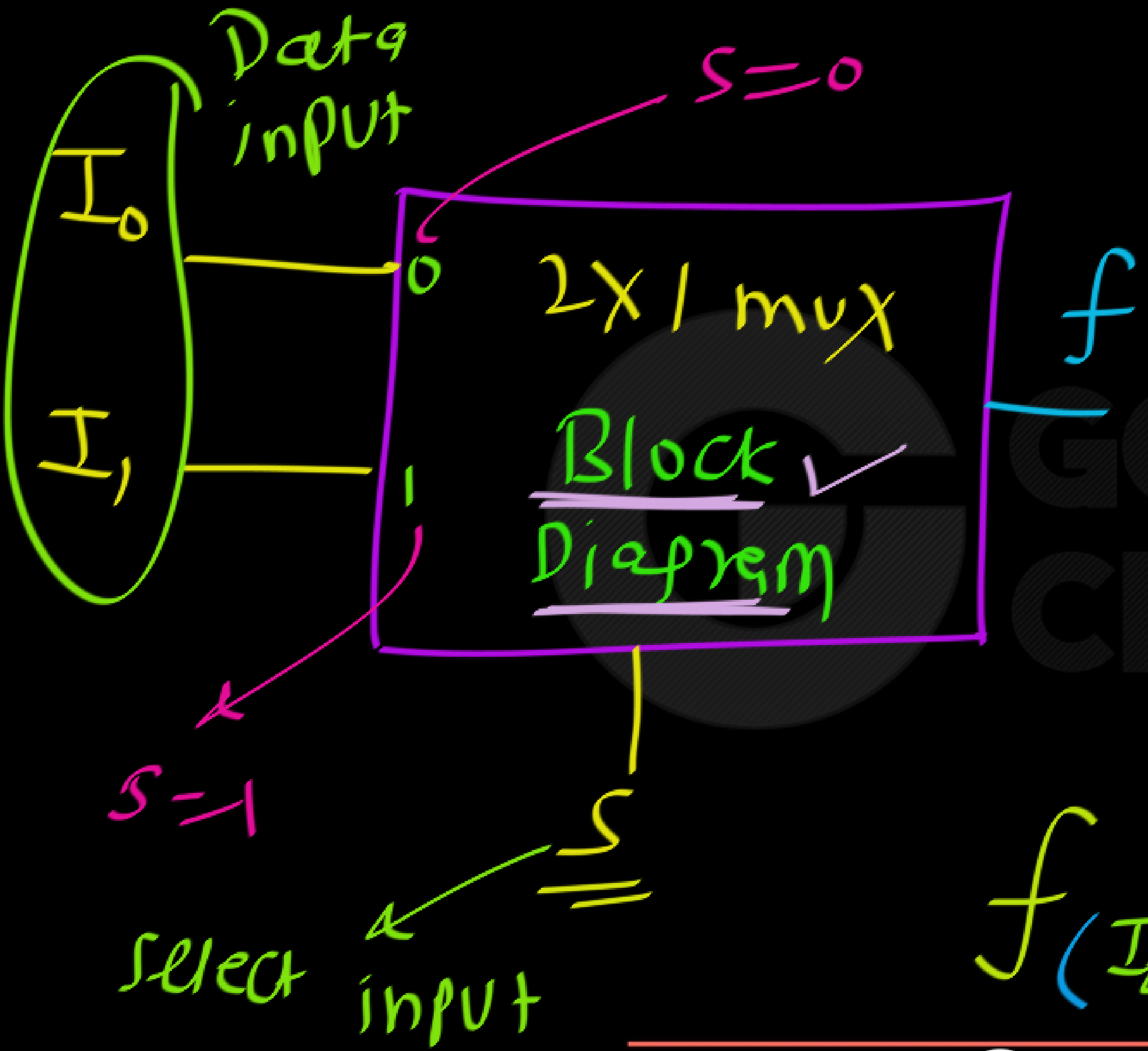


Multiplexers

A multiplexer (or data selector, abbreviated as MUX) has a group of data inputs and a group of control inputs. The control inputs are used to select one of the data inputs and connect it to the output terminal. Figure 9-1 shows a 2-to-1 multiplexer and its switch analog. When the control input A is 0, the switch is in the upper position and the MUX output is $Z = I_0$; when A is 1, the switch is in the lower position and the MUX output is $Z = I_1$. In other words, a MUX acts like a switch that selects one of the data inputs (I_0 or I_1) and transmits it to the output. The logic equation for the 2-to-1 MUX is therefore:

$$Z = A'I_0 + AI_1$$



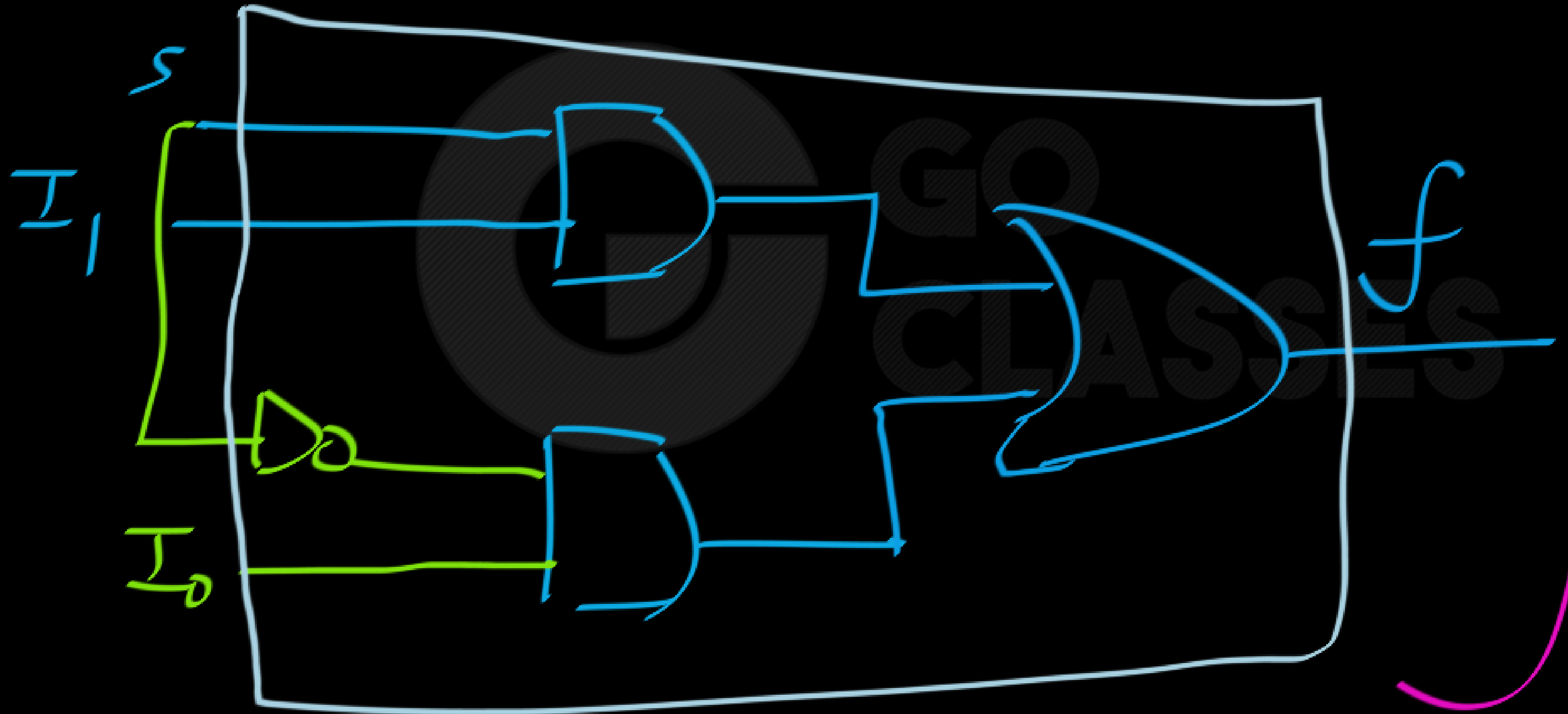


S	f
0	I_0
1	I_1

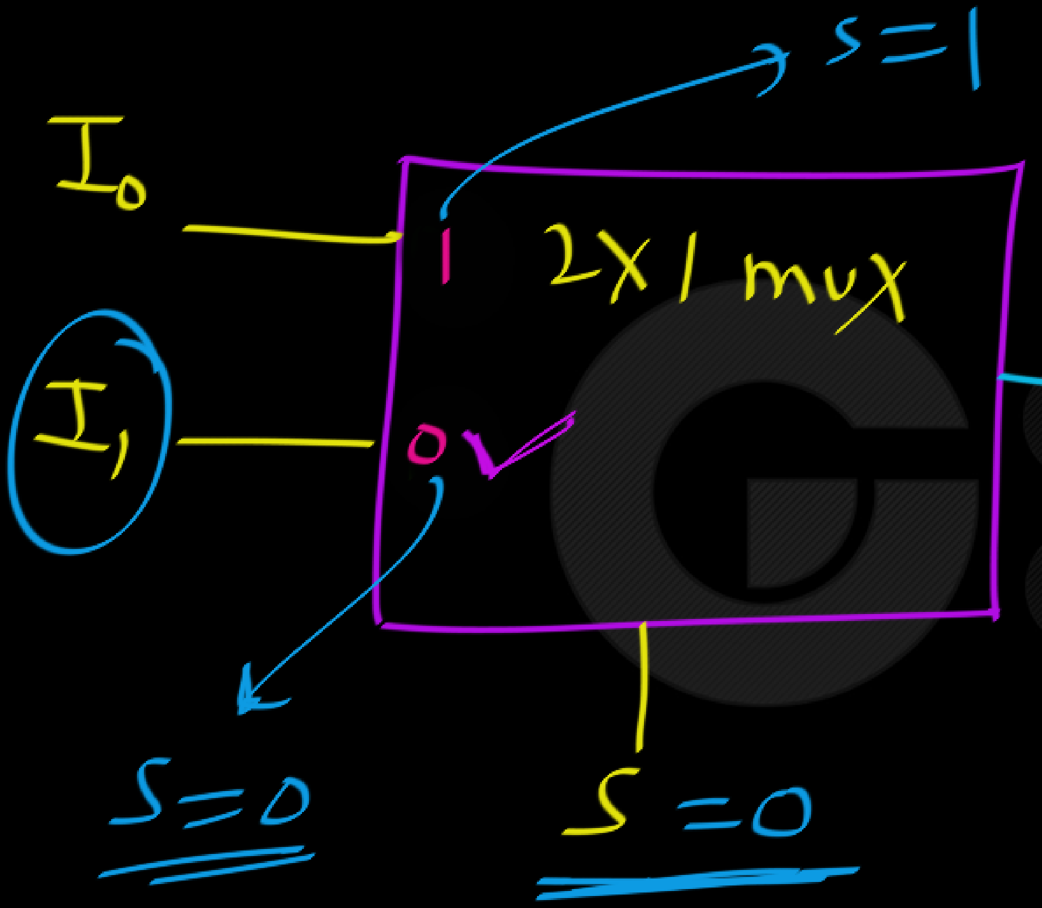
Compact Truth Table

$$f(I_0, I_1, S) = \bar{S} I_0 + S I_1$$

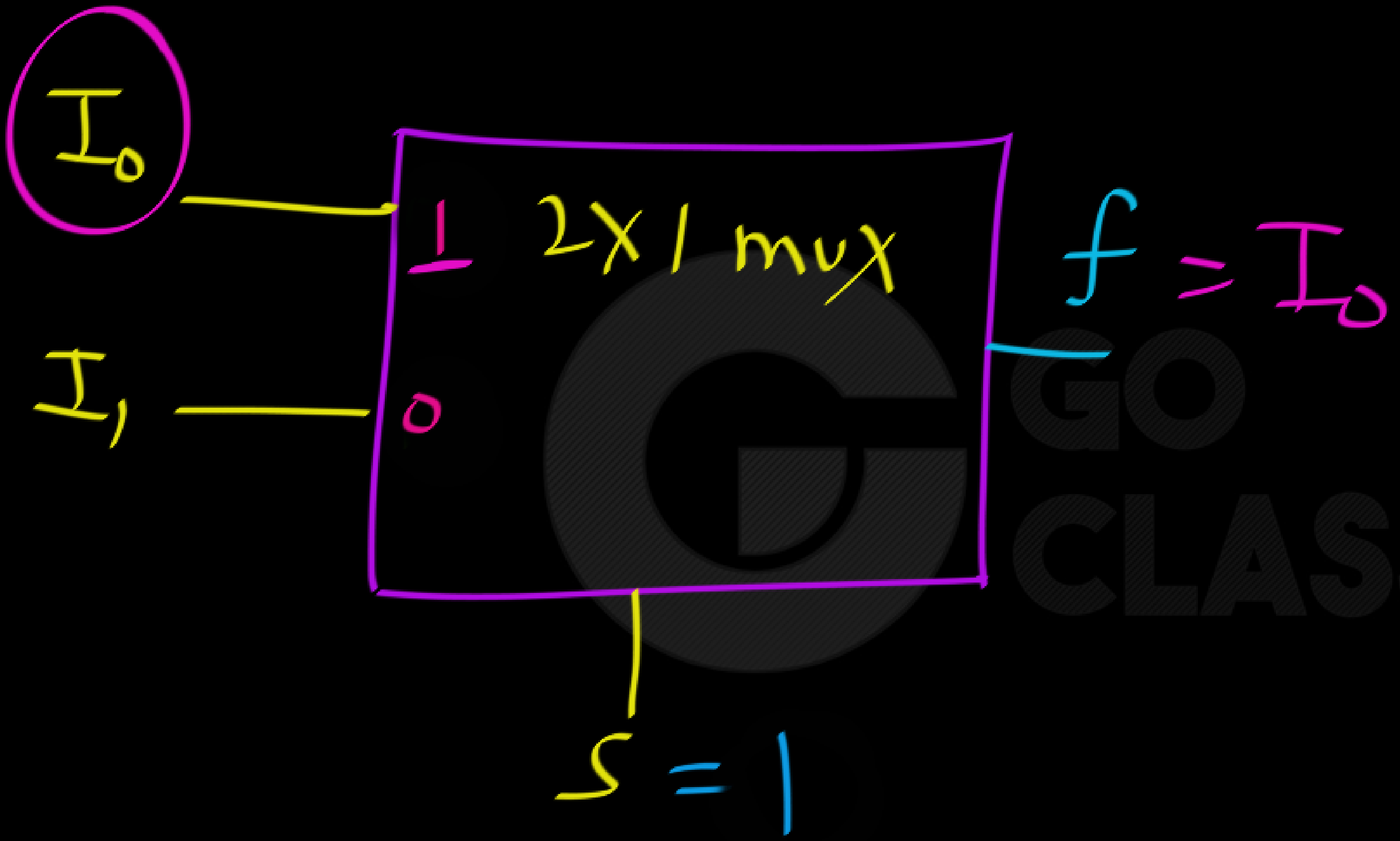
$$f(I_0, I_1, s) = \underbrace{\bar{S} I_0 + s I_1}$$



Actual mux circuit



$f = I_1$


I_0
1
2X1 mux
I_1
0
$f = I_0$
$S = 1$

GO CLASSES

GO Classes

Q- Implement the 2 to 1 MUX

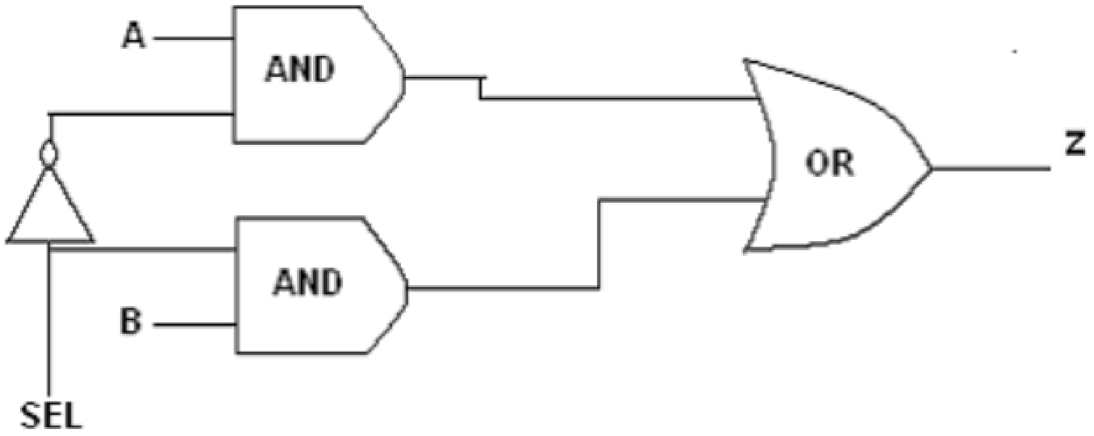
Ans: Here we have 2 input, 1 selection pin and 1 output pin and truth table is

SEL(s)	Y
0	D0
1	D1

So the equation for MUX can be written as

$Y = s'D0 + sD1$

which can be implemented using gates as follow:

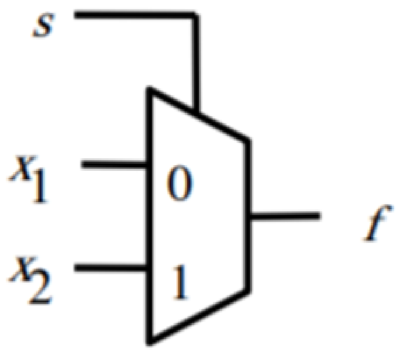


Graphical Symbol for a 2-1 Multiplexer

$$f_{(s, x_1, x_2)} = \bar{s} x_1 + s x_2$$

s	f
0	x_1
1	x_2

Compact Truth Table.



GO CLASSES

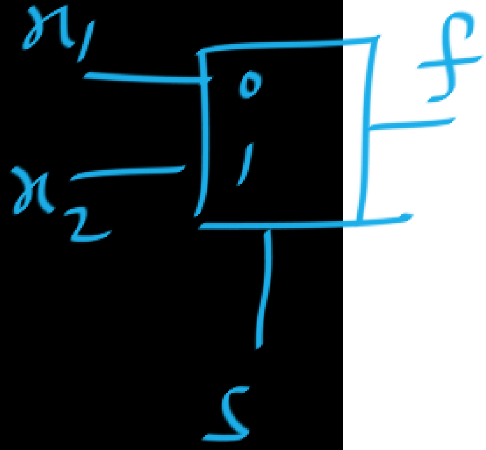
Truth Table for a 2-1 Multiplexer

GO Classes

$s\ x_1\ x_2$	$f(s, x_1, x_2)$
0 0 0	[redacted]
0 0 1	
0 1 0	
0 1 1	?
1 0 0	
1 0 1	
1 1 0	
1 1 1	

www.goclasses.in

Truth Table for a 2-1 Multiplexer



$s\ x_1\ x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

s=0 (rows 000–011)

s=1 (rows 100–111)

GO CLASSES

Truth Table for a 2-1 Multiplexer

GO Classes

$s\ x_1\ x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

www.goclasses.in

More Compact Truth-Table Representation

$s\ x_1\ x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

(a)Truth table

s	$f(s, x_1, x_2)$
0	x_1
1	x_2

GO CLASSES

Let's Derive the SOP form

GO Classes

$s\ x_1\ x_2$	$f(s, x_1, x_2)$	
0 0 0	0	
0 0 1	0	
0 1 0	$\boxed{1}$ ✓	$\overline{s}\, x_1\, \overline{x}_2$
0 1 1	$\boxed{1}$ ✓	$\overline{s}\, x_1\, x_2$
1 0 0	0	
1 0 1	$\boxed{1}$ ✓	$s\, \overline{x}_1\, x_2$
1 1 0	0	
1 1 1	$\boxed{1}$ ✓	$s\, x_1\, x_2$

$$f(s, x_1, x_2) = \overline{s}\, x_1\, \overline{x}_2 + \overline{s}\, x_1\, x_2 + s\, \overline{x}_1\, x_2 + s\, x_1\, x_2$$

Let's simplify this expression

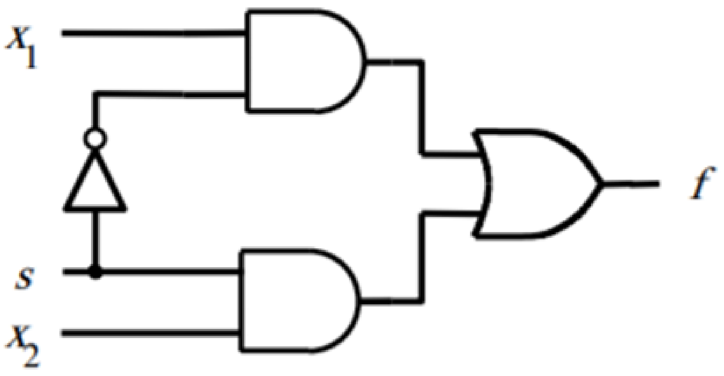
$$f(s, x_1, x_2) = \overline{s}\, x_1\, \overline{x_2} + \overline{s}\, x_1\, x_2 + s\, \overline{x_1}\, x_2 + s\, x_1\, x_2$$

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

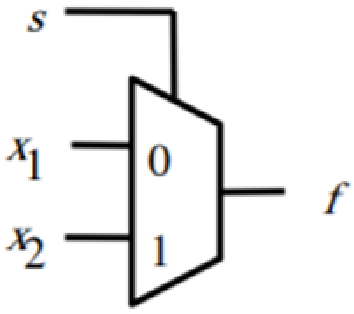
$$f(s, x_1, x_2) = \overline{s}\, x_1 + s\, x_2$$

Circuit for 2-1 Multiplexer

GO Classes



(b) Circuit

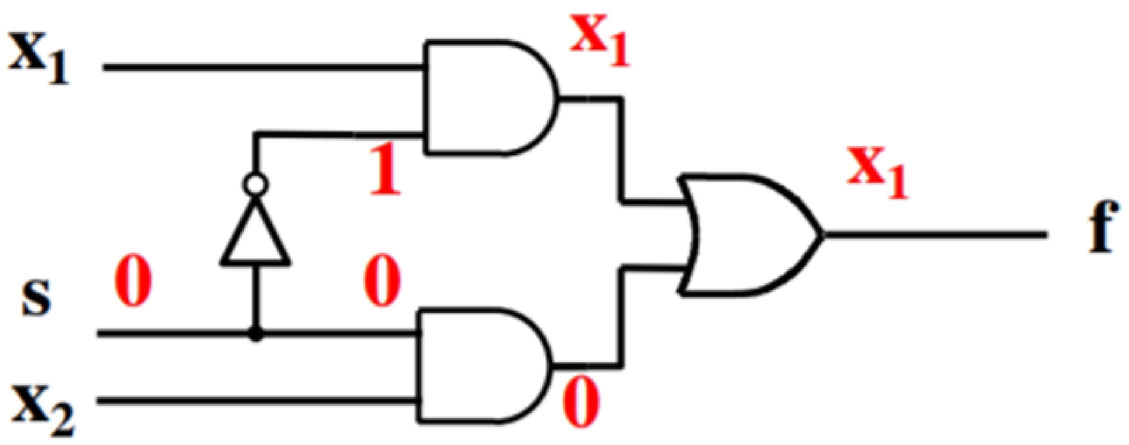


(c) Graphical symbol

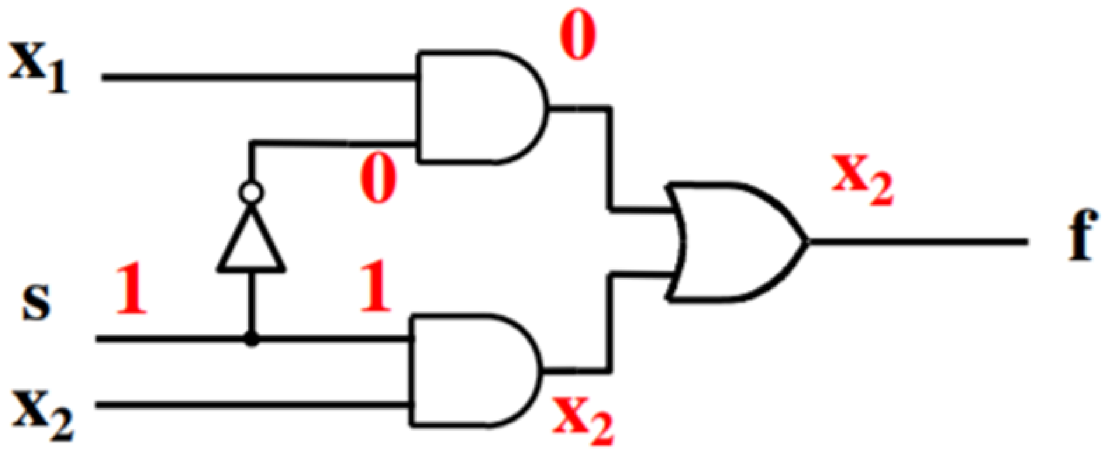
(Block Diagram)

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

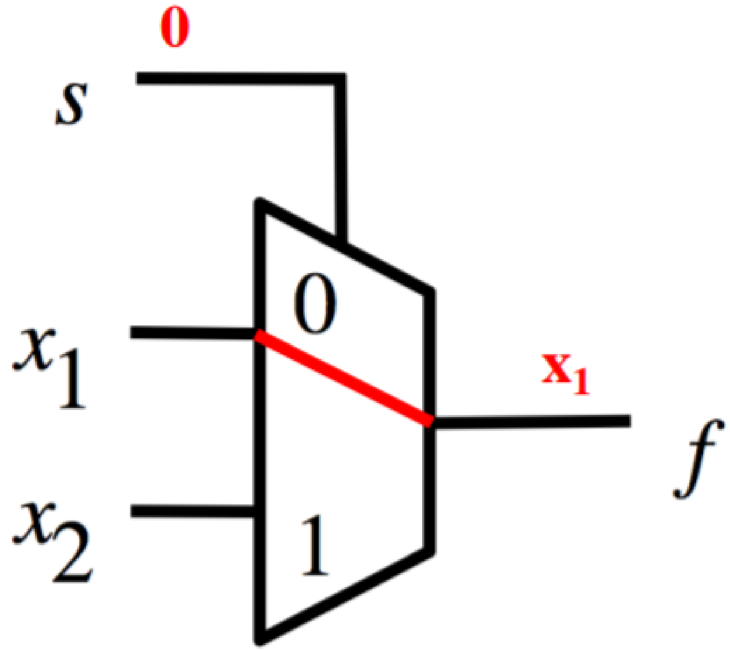
Analysis of the 2-1 Multiplexer (when the input s=0)



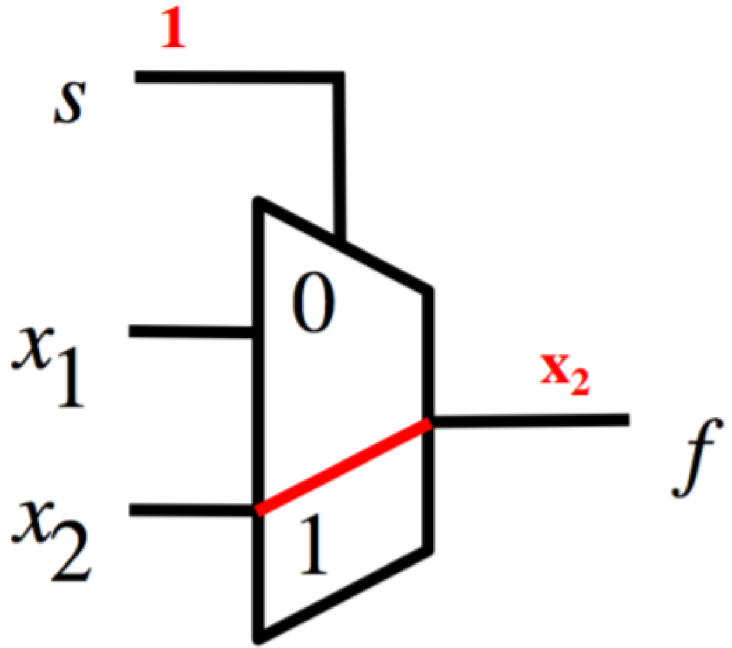
Analysis of the 2-1 Multiplexer (when the input s=1)



Analysis of the 2-1 Multiplexer (when the input s=0)



Analysis of the 2-1 Multiplexer (when the input s=1)



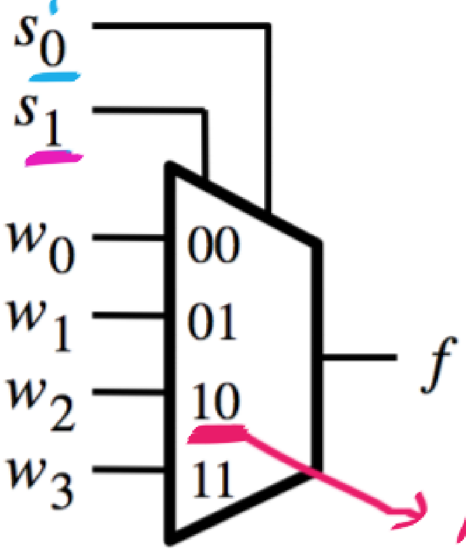
4-1 Multiplexer (Definition)

- Has four inputs: w_0 , w_1, w_2, w_3
- Also has two select lines: s_1 and s_0
- If s_1=0 and s_0=0, then the output f is equal to w_0
- If s_1=0 and s_0=1, then the output f is equal to w_1
- If s_1=1 and s_0=0, then the output f is equal to w_2
- If s_1=1 and s_0=1, then the output f is equal to w_3

Graphical Symbol and Truth Table

by Default MSB

by Default LSB



means $s_1=1, s_0=0$

(a) Graphic symbol

s_1	s_0	f
0	0	w_0
0	1	w_1
1	0	w_2
1	1	w_3

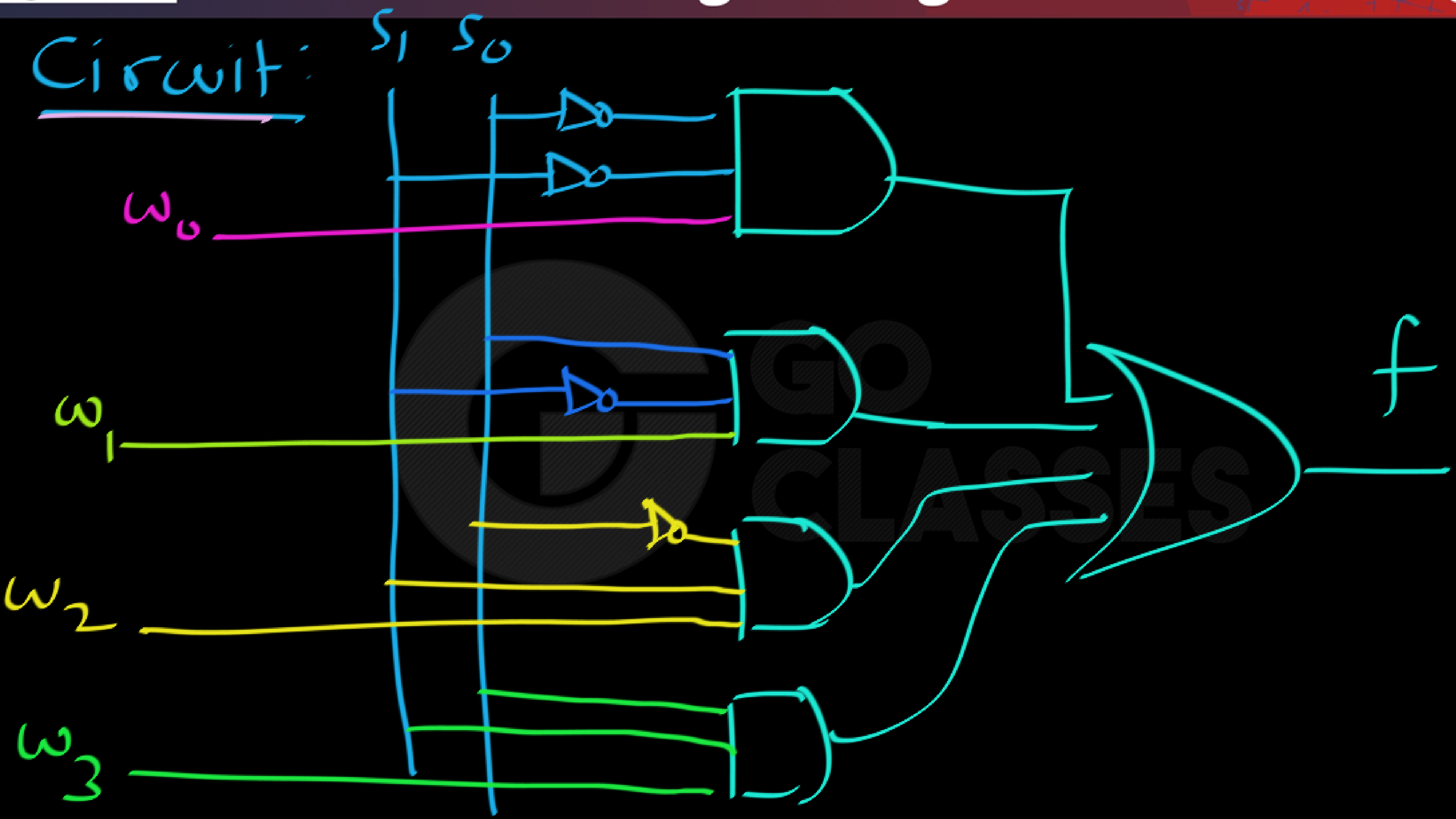
$s_1 \bar{s}_0 w_2$

(b) Truth table

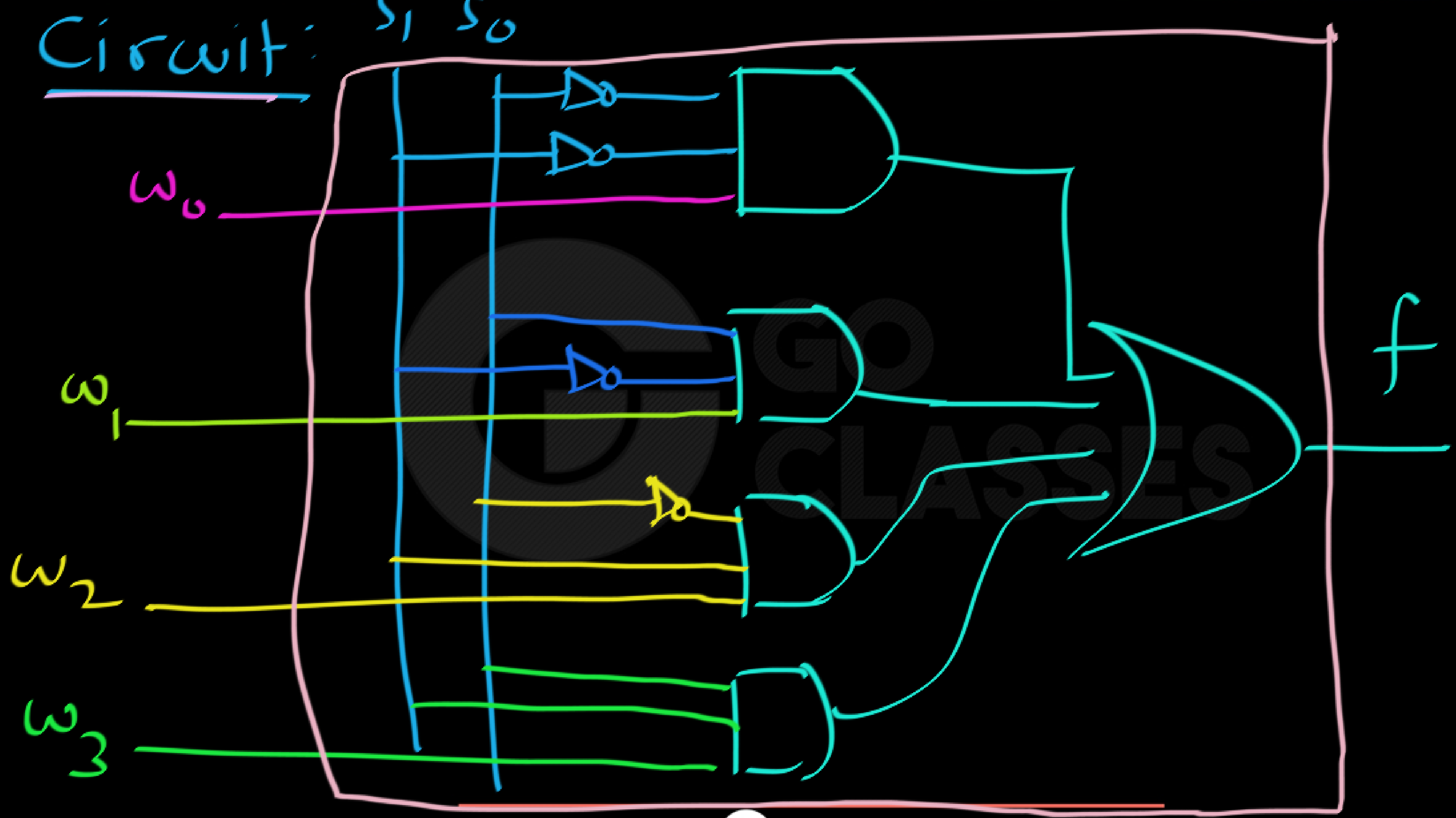
$$f(s_1, s_0, w_0, w_1, w_2, w_3) = \bar{s}_1 \bar{s}_0 w_0 + \bar{s}_1 s_0 w_1 + s_1 \bar{s}_0 w_2 + s_1 s_0 w_3$$

$$(\text{—}) + (\text{—}) + (\text{—}) + (\text{—})$$

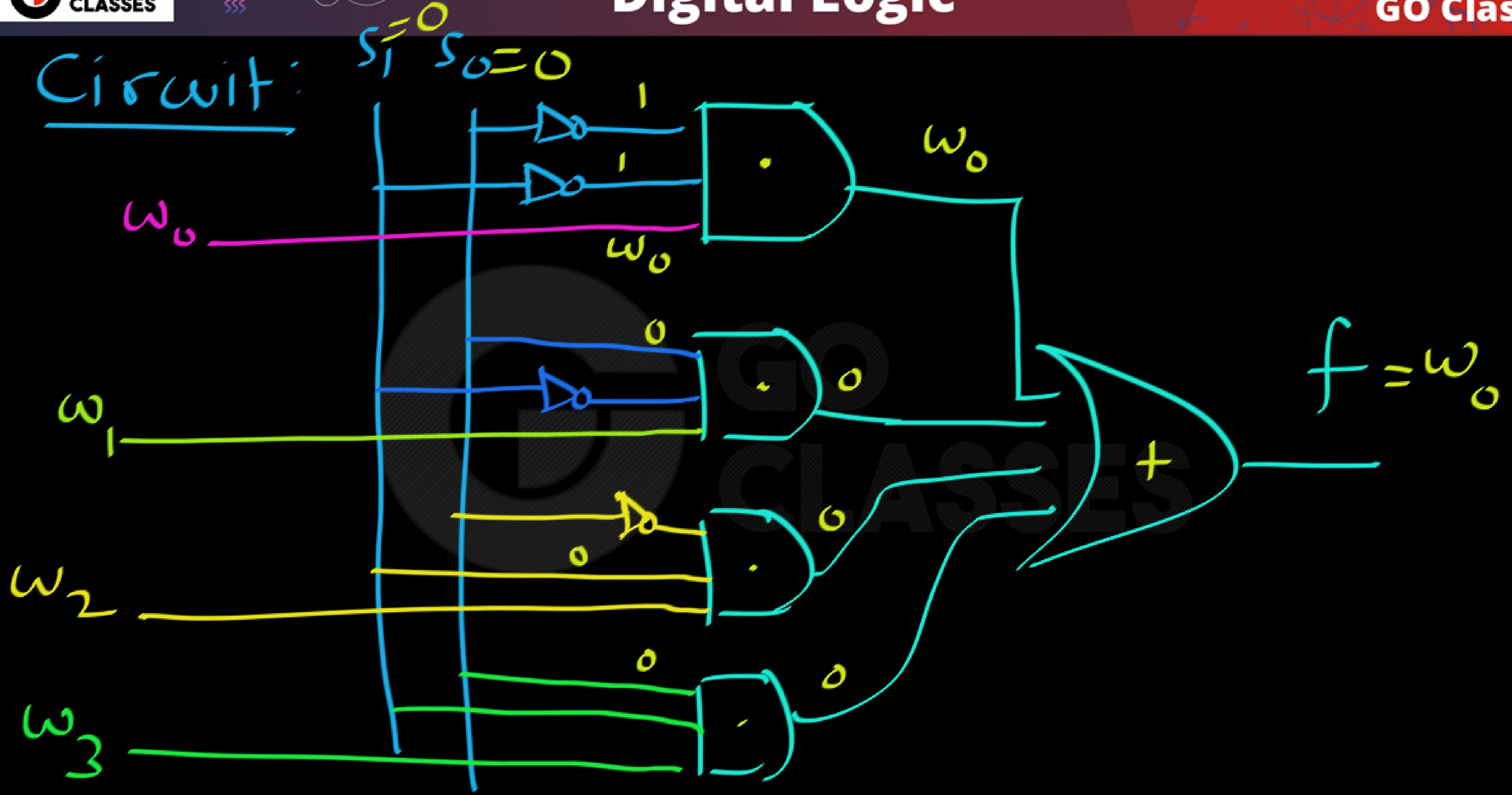
Circuit:



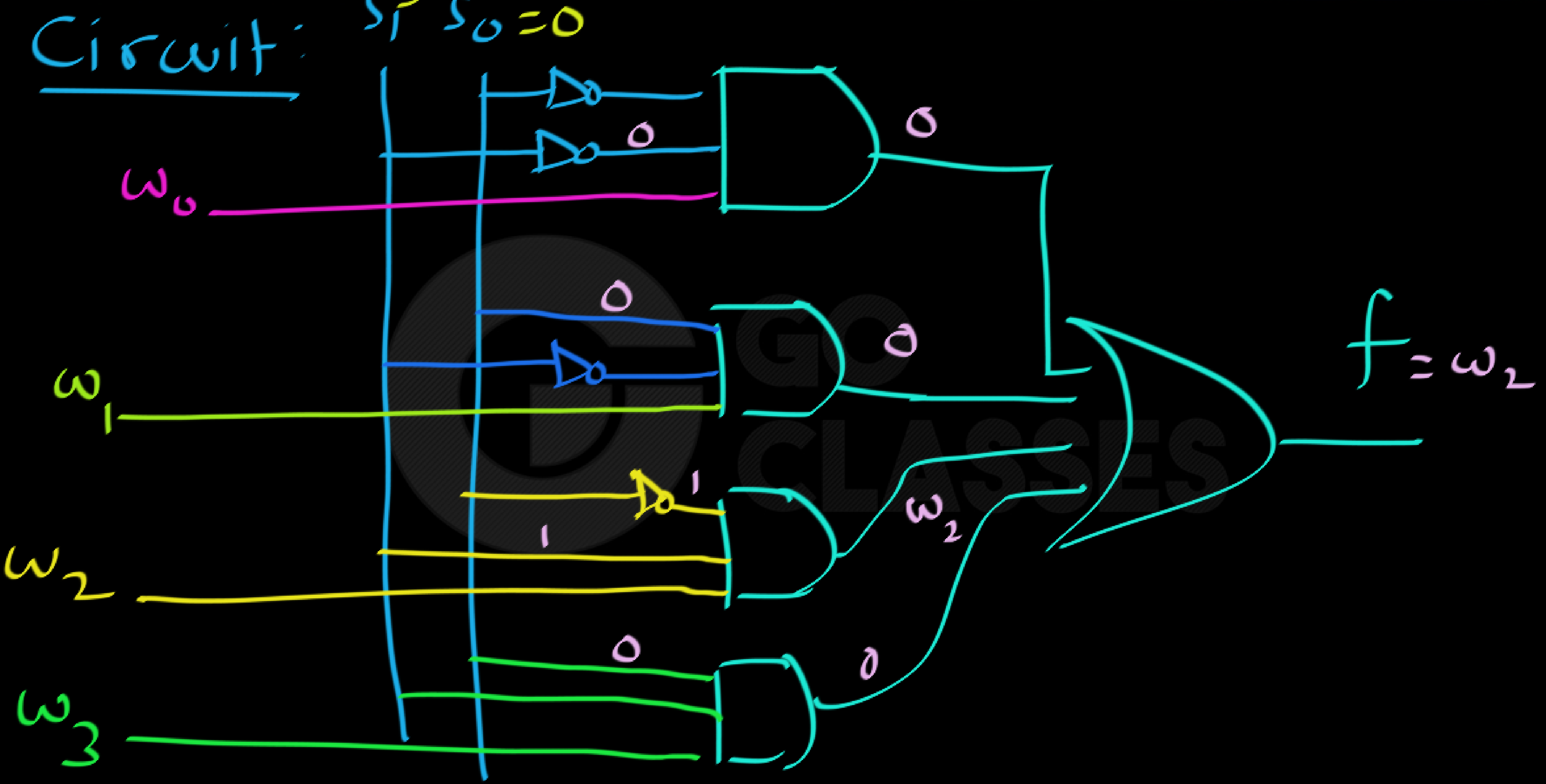
Cirwit:



Cirwit:



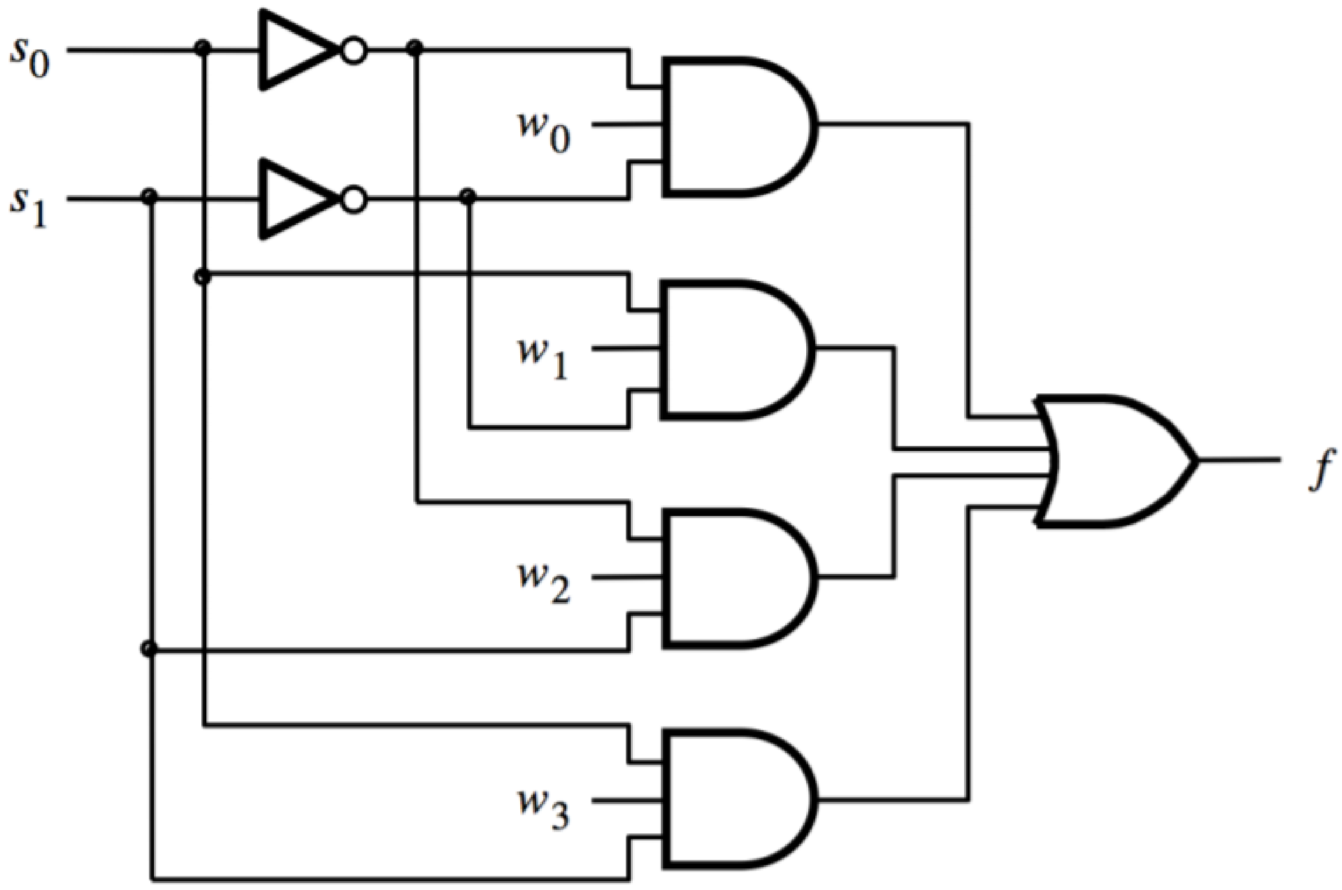
Cirwit:



GO CLASSES

4-1 Multiplexer (SOP circuit)

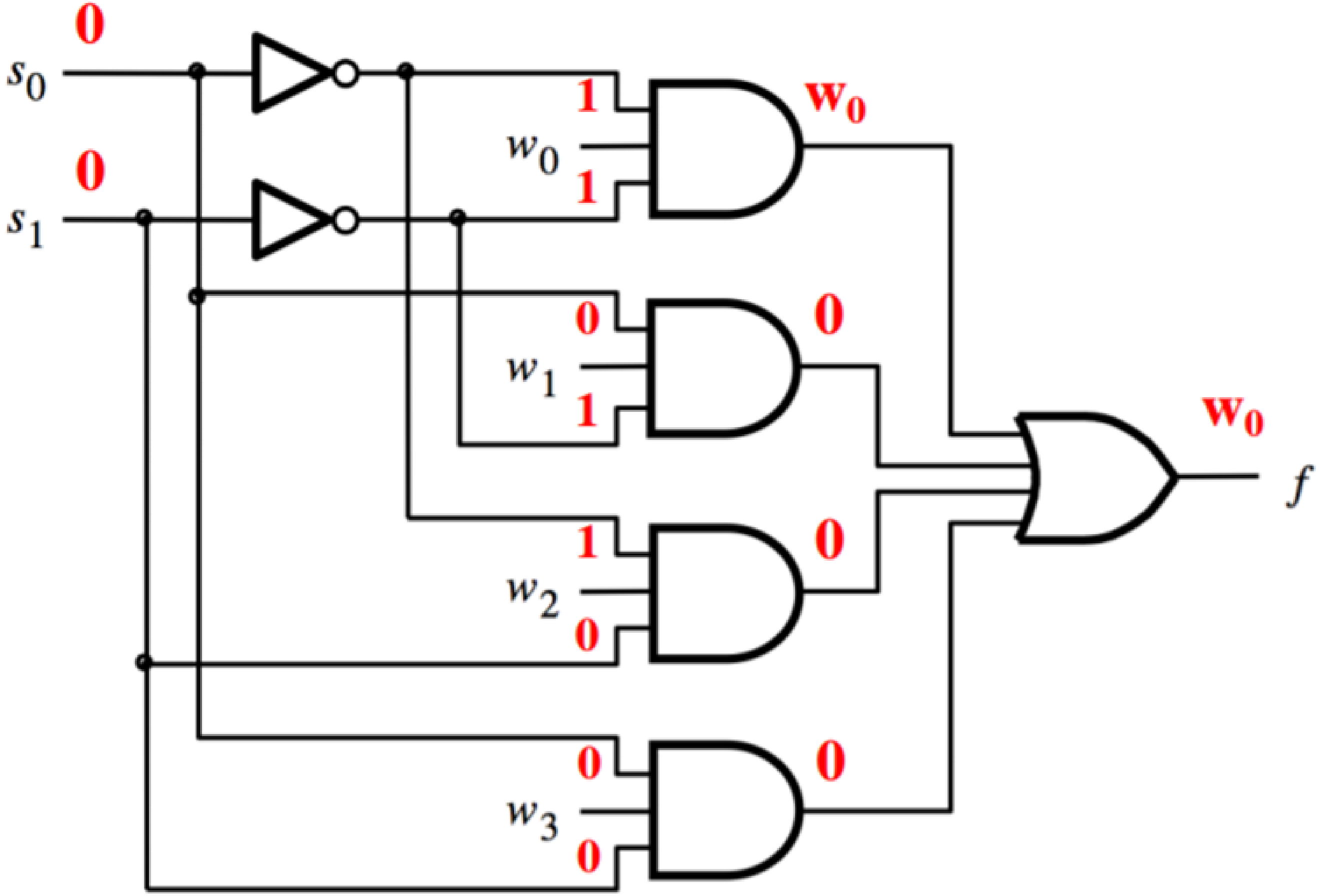
O Classes



$$f = \overline{s_1}\,\overline{s_0}\,w_0 + \overline{s_1}\,s_0\,w_1 + s_1\,\overline{s_0}\,w_2 + s_1\,s_0\,w_3$$

GO CLASSES

Analysis of the 4-1 Multiplexer
(s_1=0 and s_0=0)

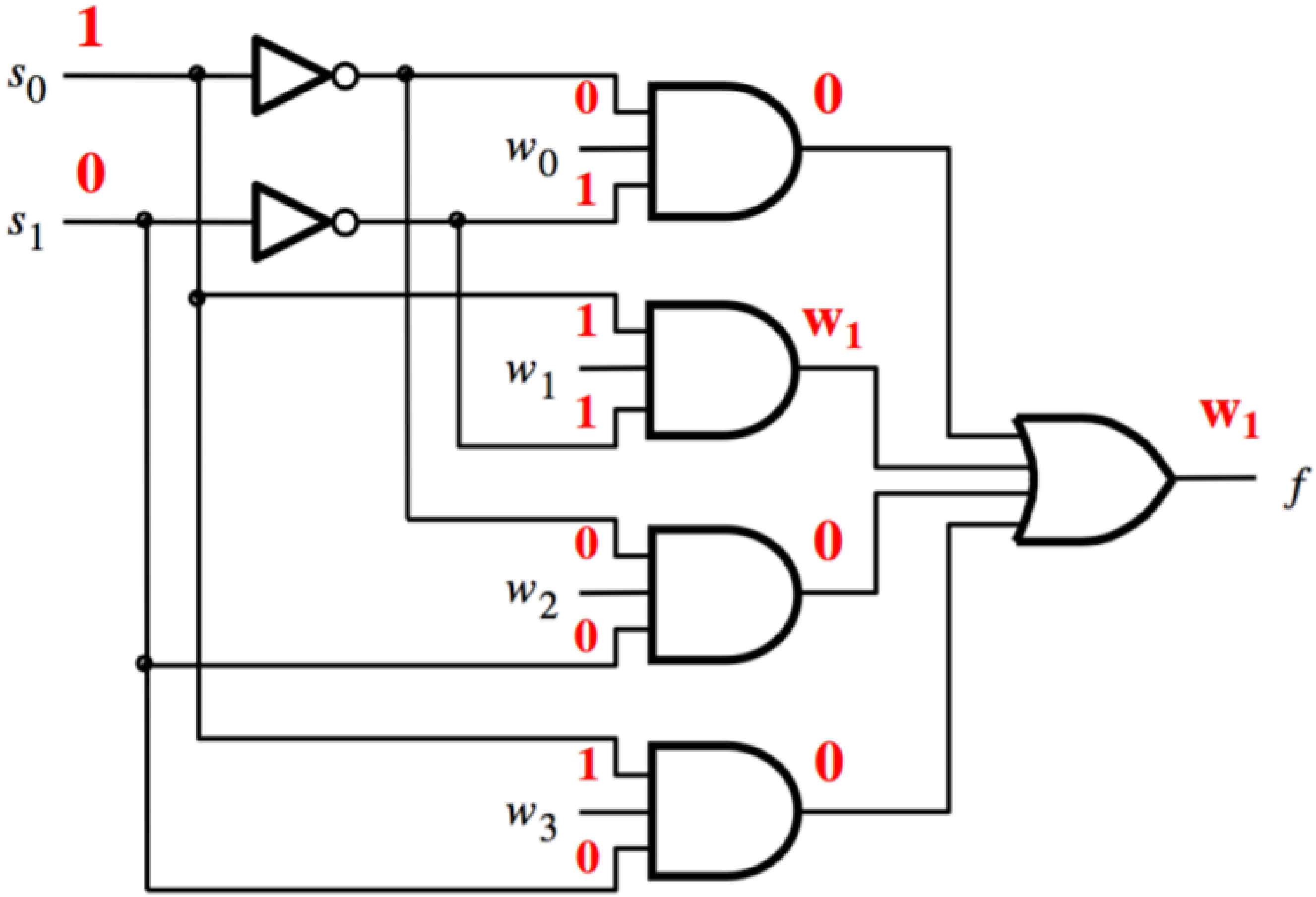


GO Classes

GO CLASSES

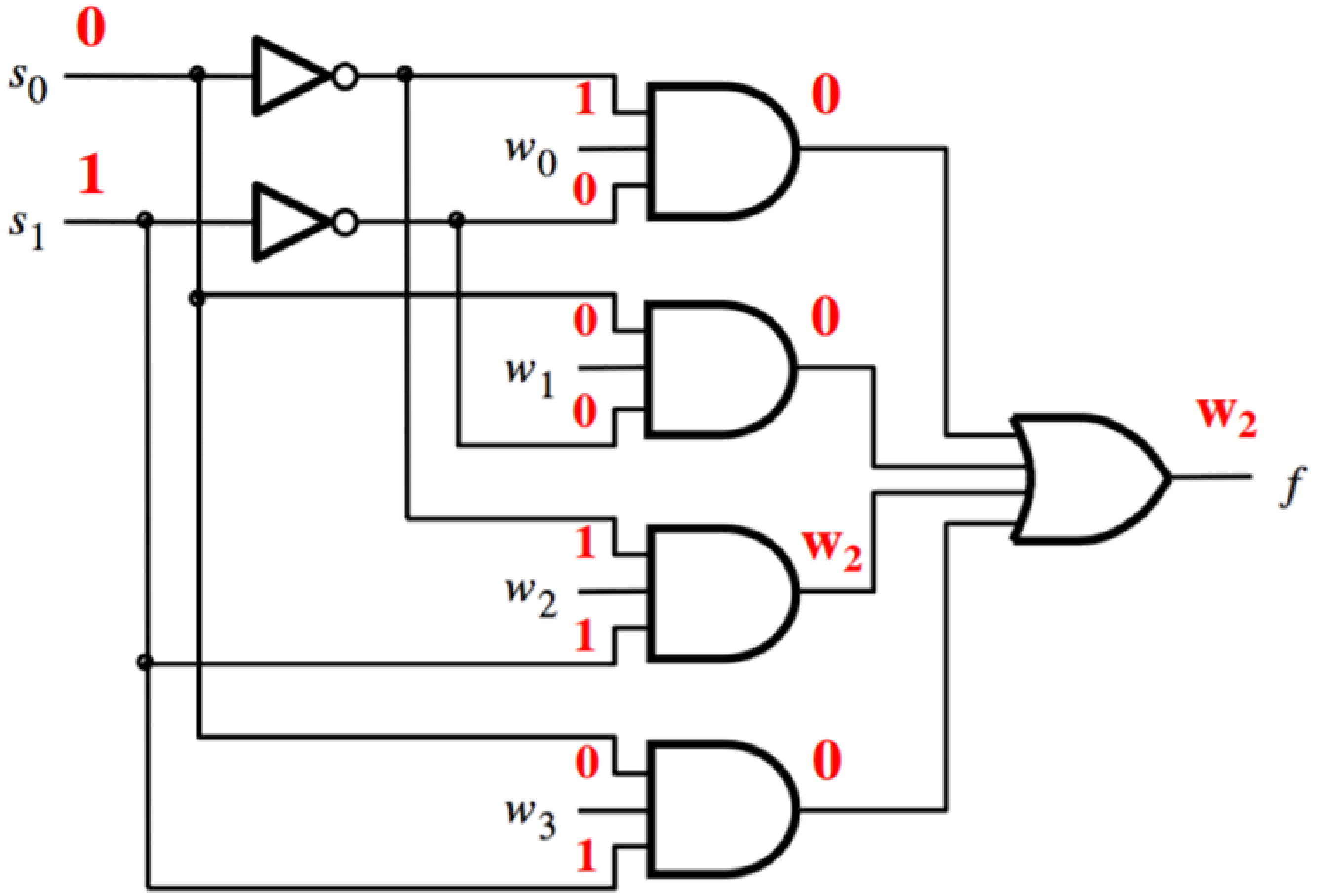
Analysis of the 4-1 Multiplexer (s_1=0 and s_0=1)

GO Classes



GO CLASSES

Analysis of the 4-1 Multiplexer
(s_1=1 and s_0=0)

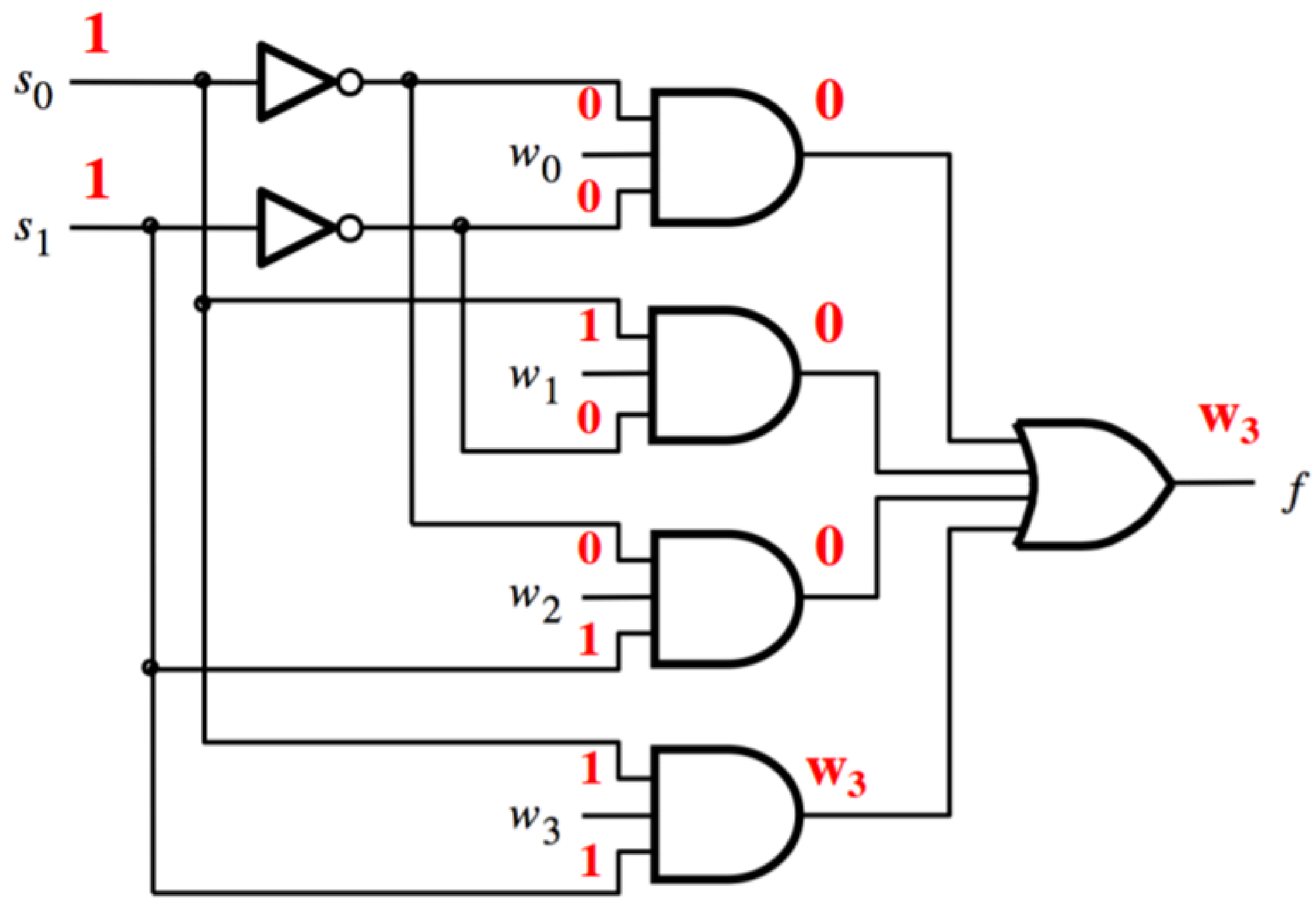


GO Classes

GO CLASSES

Analysis of the 4-1 Multiplexer
(s_1=1 and s_0=1)

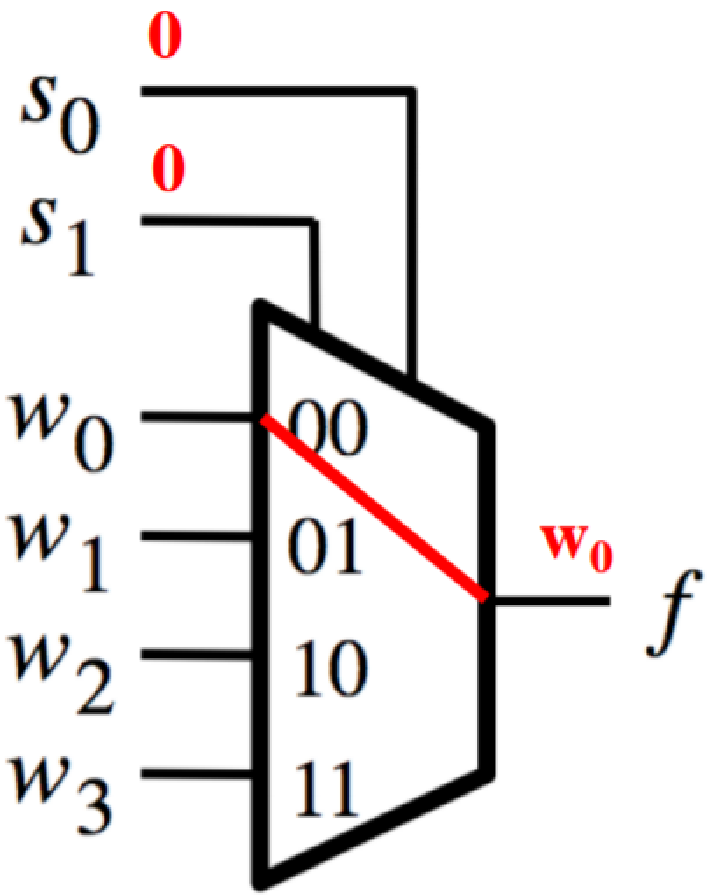
GO Classes



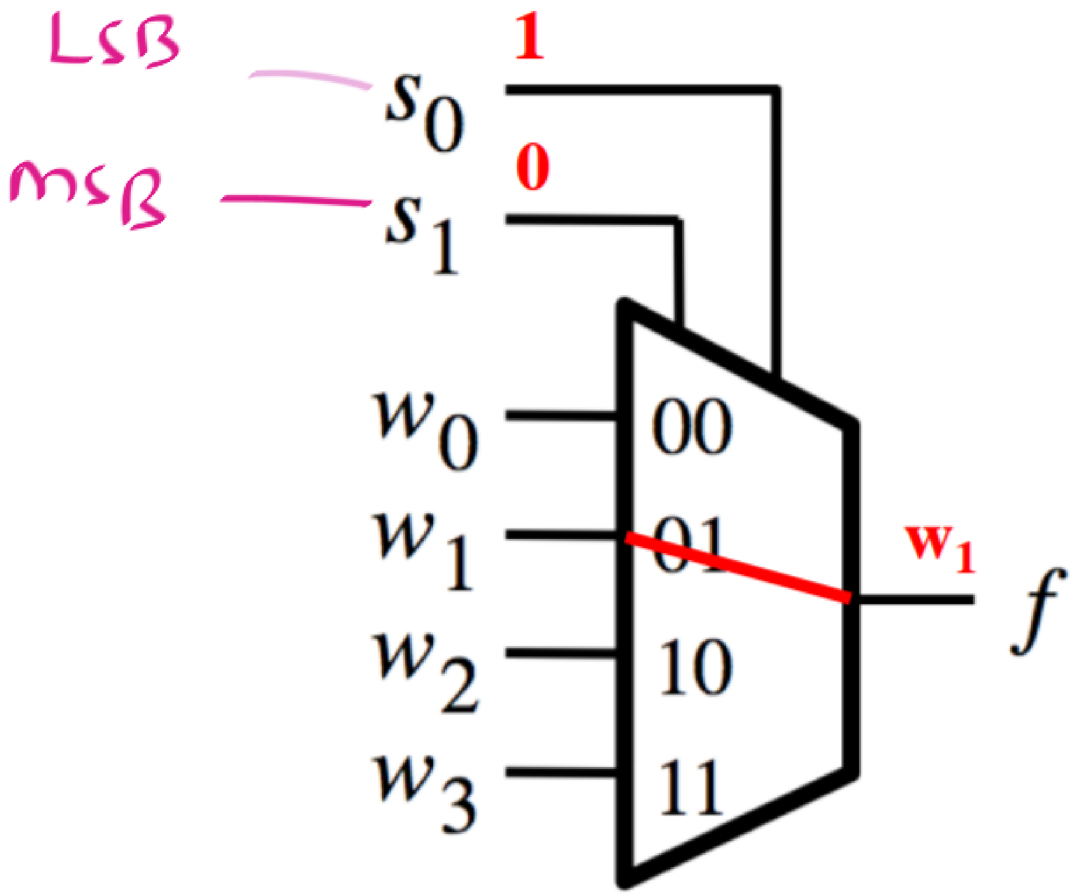
GO CLASSES

Analysis of the 4-1 Multiplexer (s_1=0 and s_0=0)

GO Classes

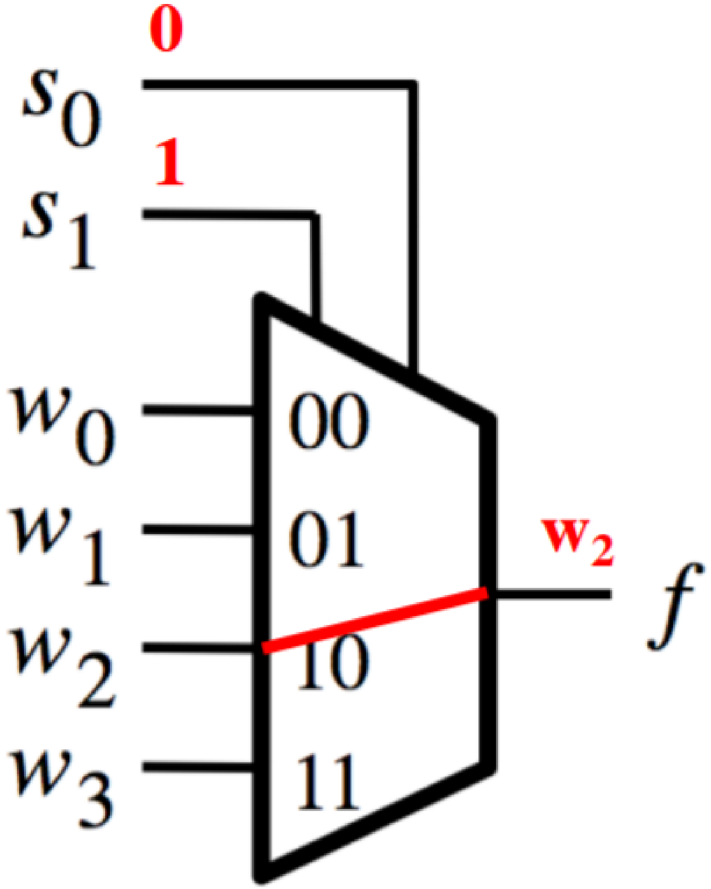


Analysis of the 4-1 Multiplexer (s_1=0 and s_0=1)



GO CLASSES

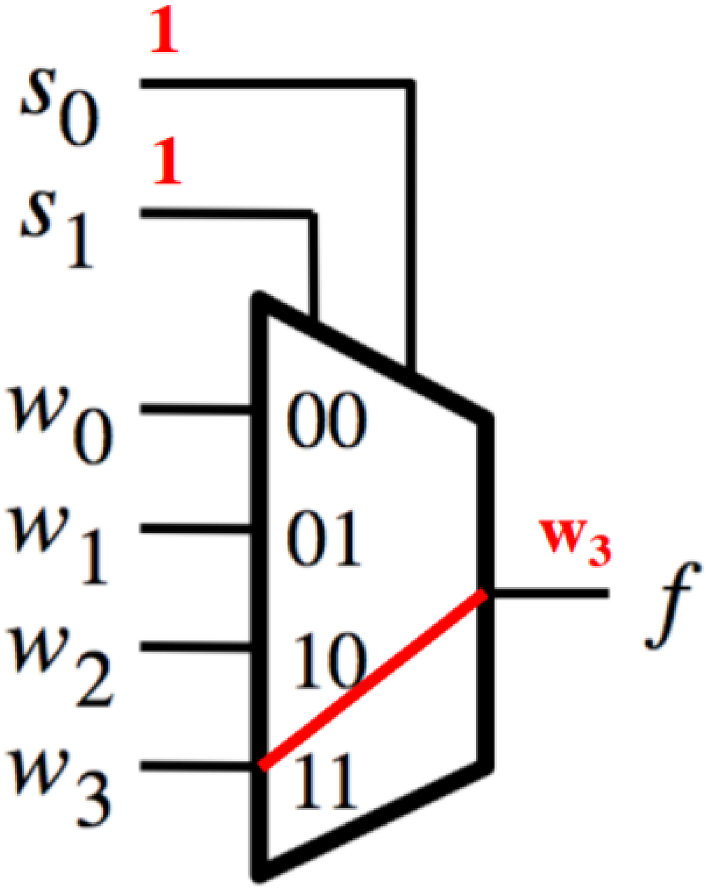
Analysis of the 4-1 Multiplexer (s_1=1 and s_0=0)



GO Classes

GO CLASSES

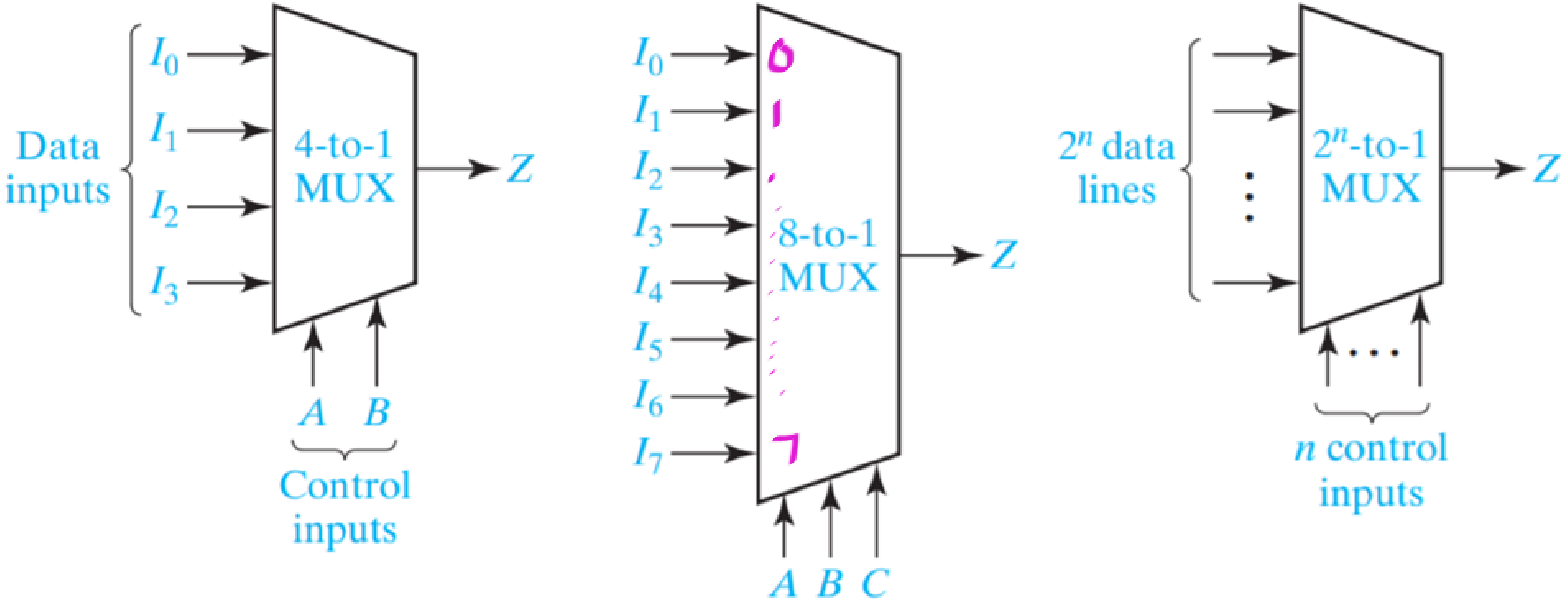
Analysis of the 4-1 Multiplexer
(s_1=1 and s_0=1)



GO Classes

Figure 9-2 shows diagrams for a 4-to-1 multiplexer, 8-to-1 multiplexer, and 2^n-to-1 multiplexer. The 4-to-1 MUX acts like a four-position switch that transmits one of the four inputs to the output. Two control inputs (A and B) are needed to select one of the four inputs. If the control inputs are $AB = 00$, the output is I_0; similarly, the control inputs 01, 10, and 11 give outputs of I_1, I_2, and I_3, respectively. The 4-to-1 multiplexer is described by the equation

$$Z = A'B'I_0 + A'BI_1 + AB'I_2 + ABI_3 \tag{9-1}$$



Similarly, the 8-to-1 MUX selects one of eight data inputs using three control inputs. It is described by the equation

$$Z = A'B'C'I_0 + A'B'CI_1 + A'BC'I_2 + A'BCI_3 + AB'C'I_4 + AB'CI_5 + ABC'I_6 + ABCI_7 \tag{9-2}$$

Circuit : 8 AND Gates → 4 input AND gate

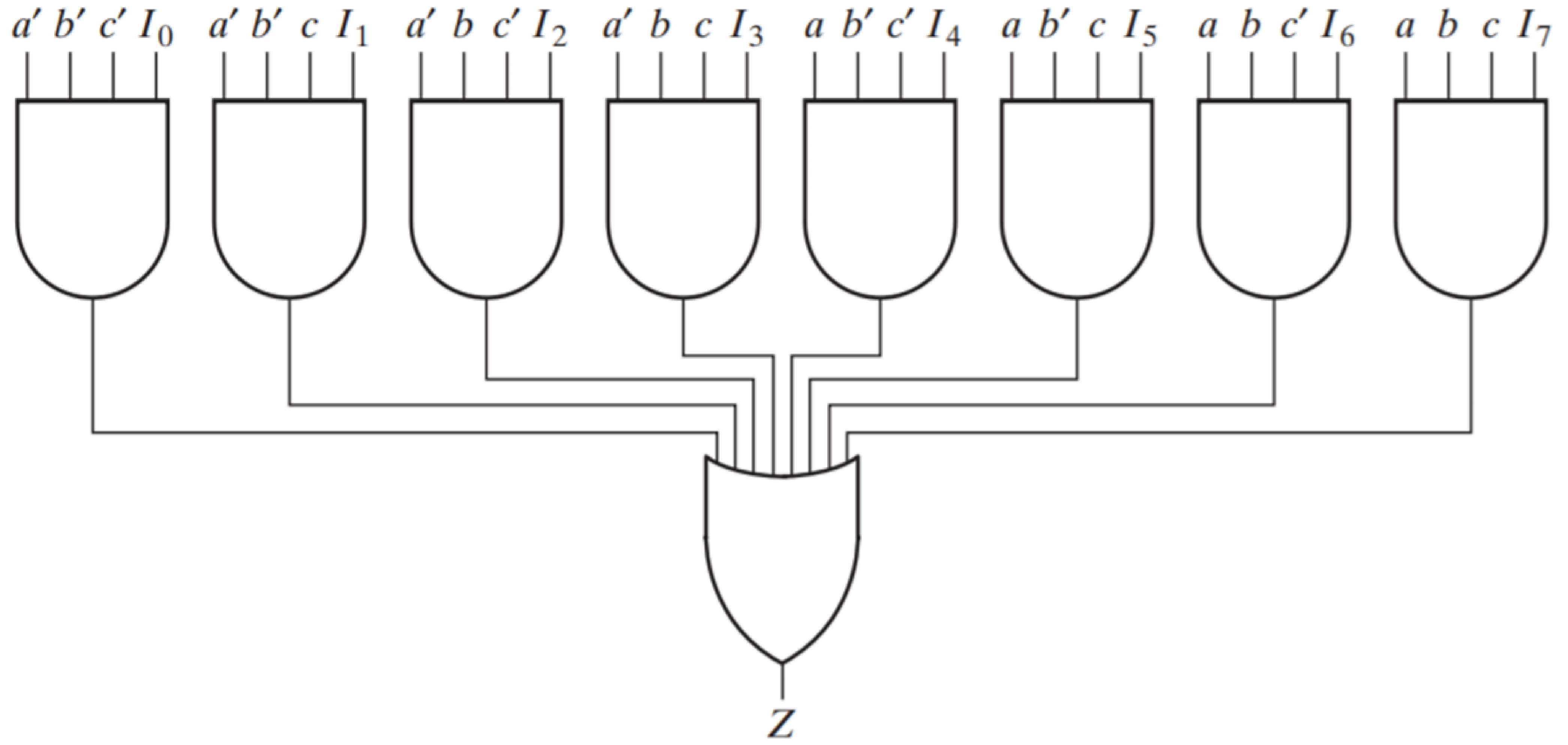
1 OR gate → 8 inputs OR gate

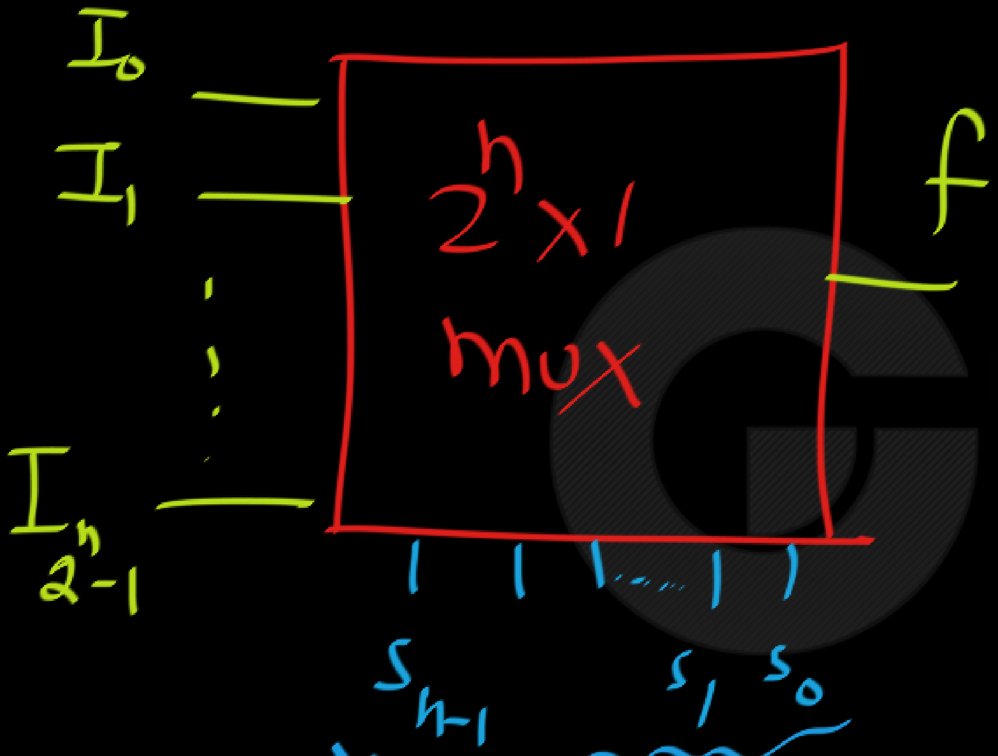
When the control inputs are $ABC = 011$, the output is I_3, and the other outputs are selected in a similar manner. Figure 9-3 shows an internal logic diagram for the 8-to-1 MUX. In general, a multiplexer with n control inputs can be used to select any one of 2^n data inputs. The general equation for the output of a MUX with n control inputs and 2^n data inputs is

$$Z = \sum_{k=0}^{2^n - 1} m_k I_k$$

where m_k is a minterm of the n control variables and I_k is the corresponding data input.

Logic Diagram for 8-to-1 MUX





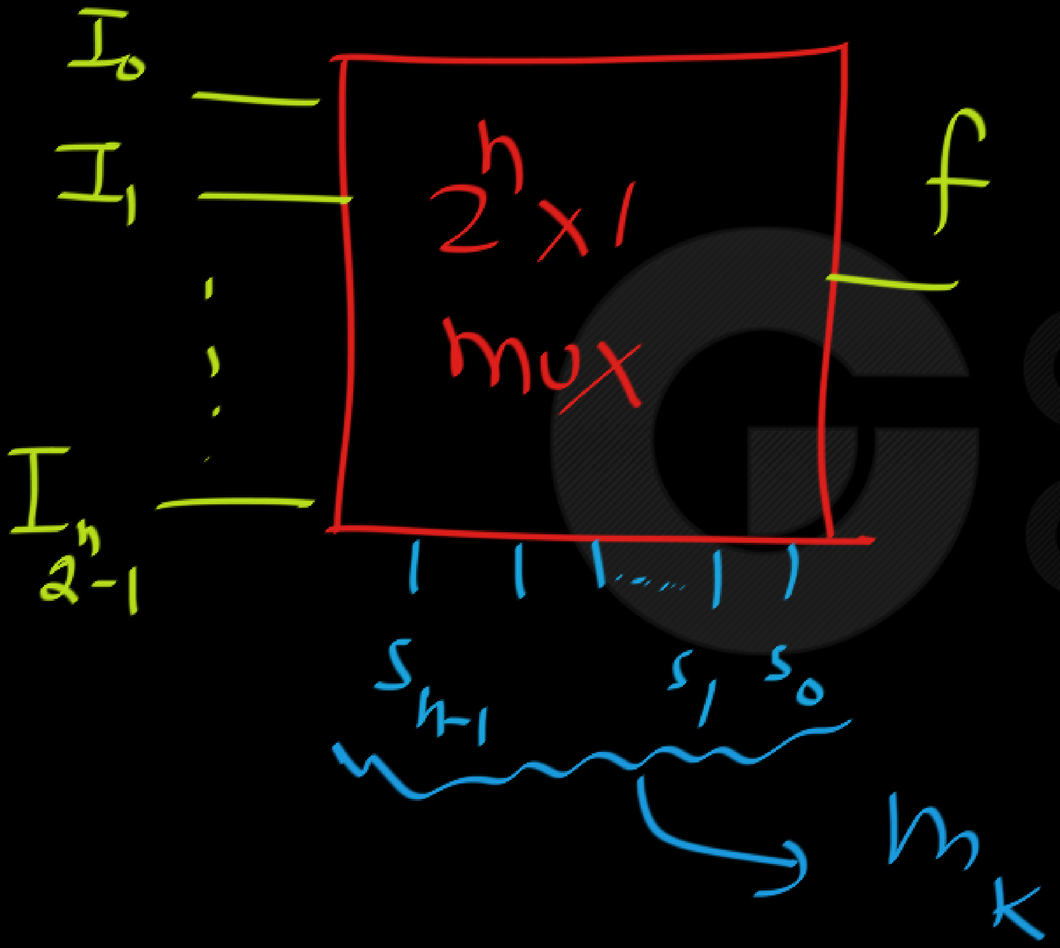
$$f = m_0 I_0 + m_1 I_1 + \cdots + m_k I_k + \cdots$$

$$\underbrace{s_{n-1} \ldots s_1 s_0} \rightarrow m_k$$

$$\text{minterm}_k \leftarrow m_k$$

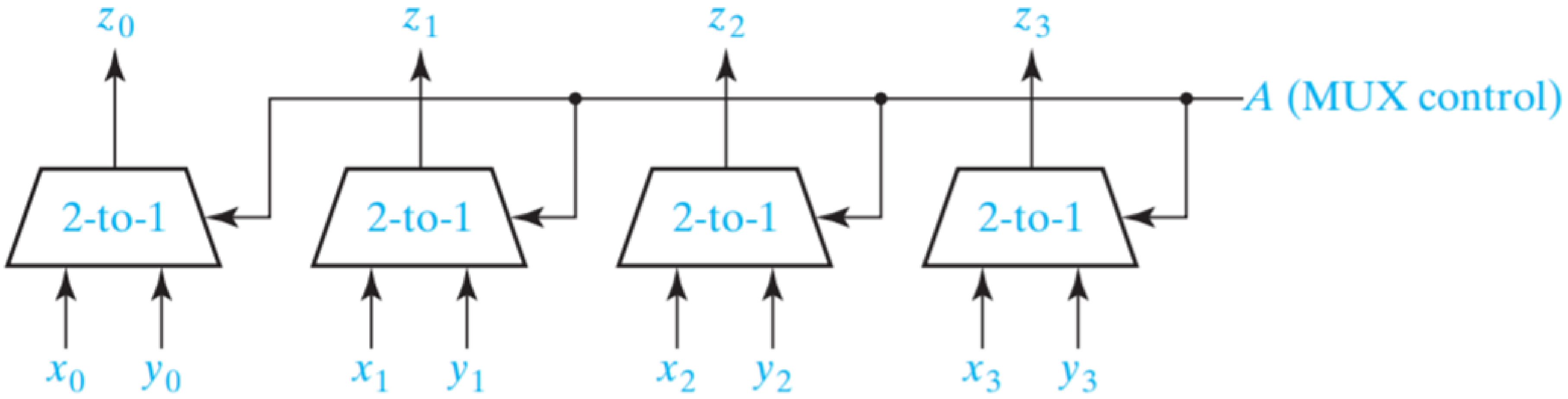
$$m_0 = \bar{s}_{n-1} \cdots \bar{s}_1 \bar{s}_0$$

$$m_1 = \bar{s}_{n-1} \cdots \bar{s}_1 s_0$$



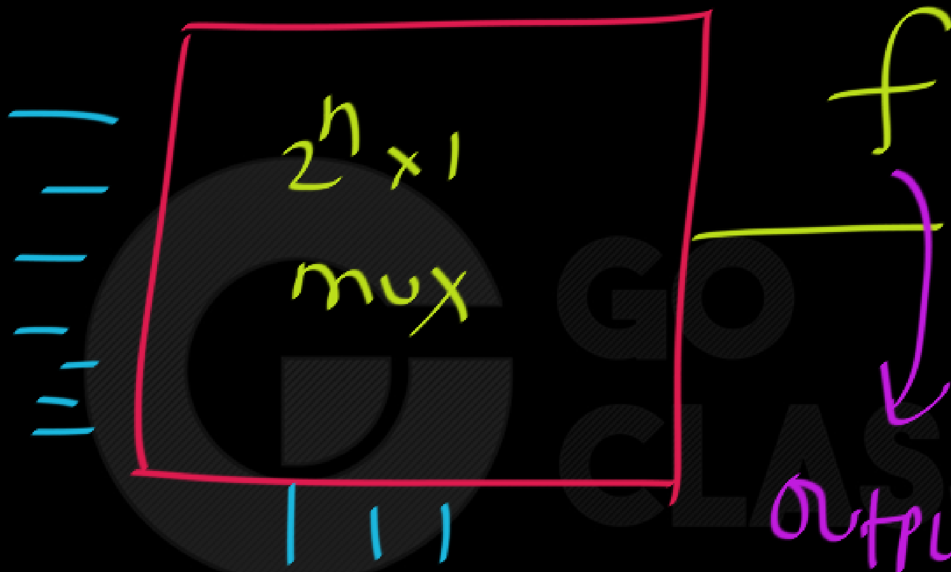
$$f = \sum_{k=0}^{(2^n)-1} m_k I_k$$

Multiplexers are frequently used in digital system design to select the data which is to be processed or stored. Figure 9-6 shows how a quadruple 2-to-1 MUX is used to select one of two 4-bit data words. If the control is $A = 0$, the values of x_0, x_1, x_2, and x_3 will appear at the z_0, z_1, z_2, and z_3 outputs; if $A = 1$, the values of y_0, y_1, y_2, and y_3 will appear at the outputs.



Next Topic:

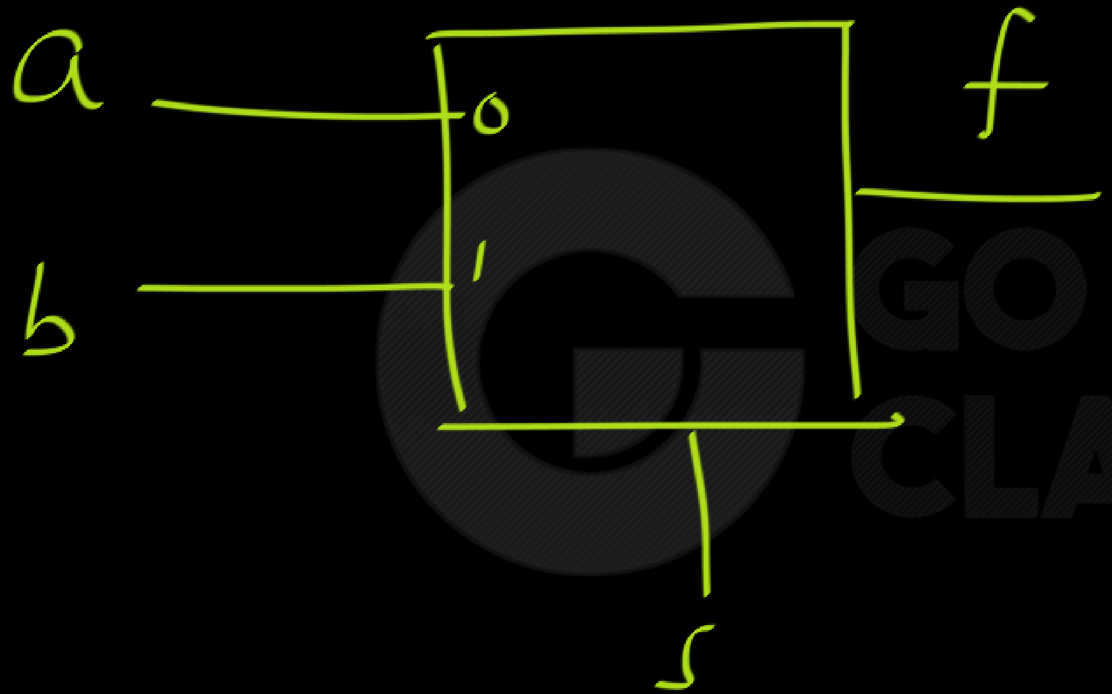
Realization(Implementation) Of Functions using Mux



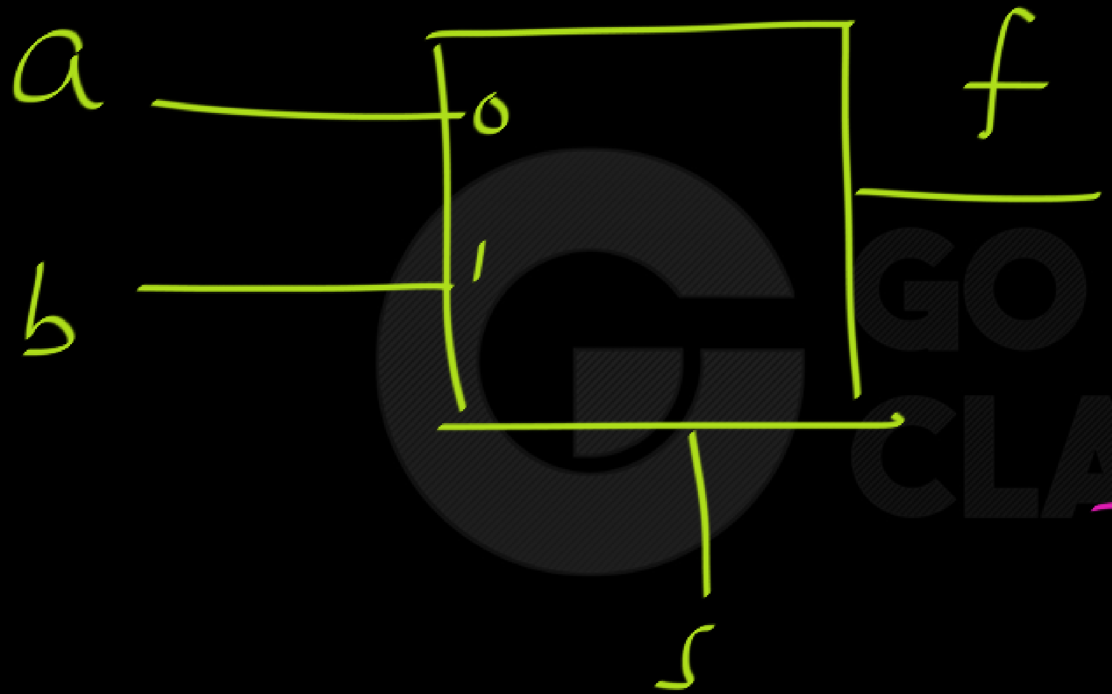
$f = ?$

output
$\equiv$
Realized function

output of mux
$\equiv$
the fun that is implemented.

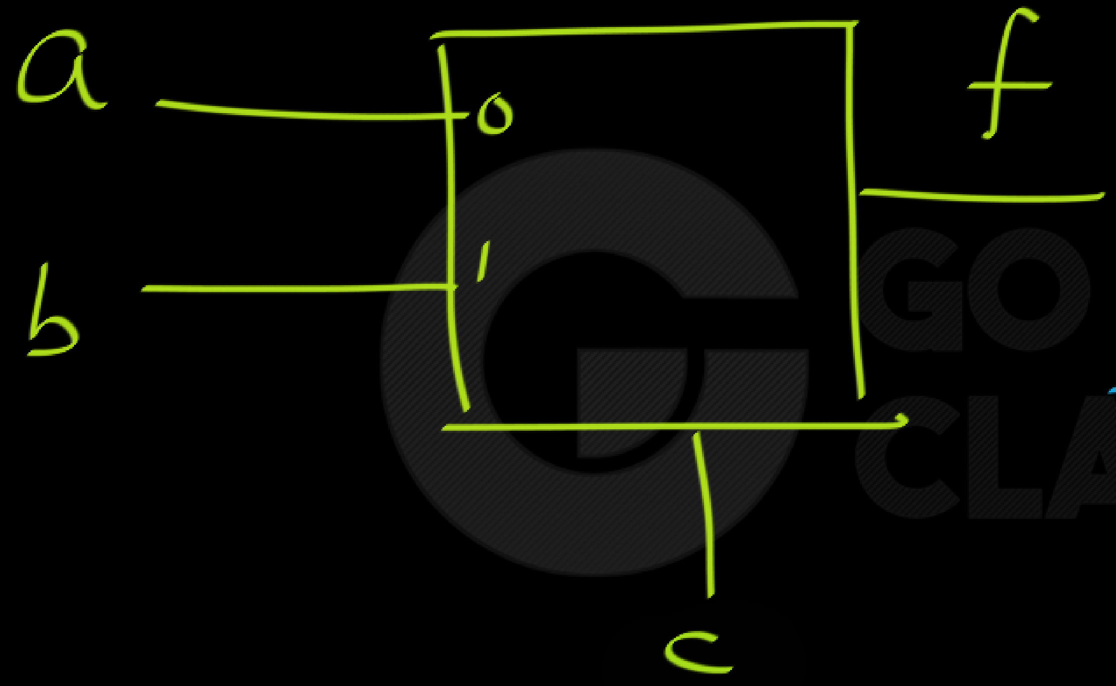


fun implemented by this mux?



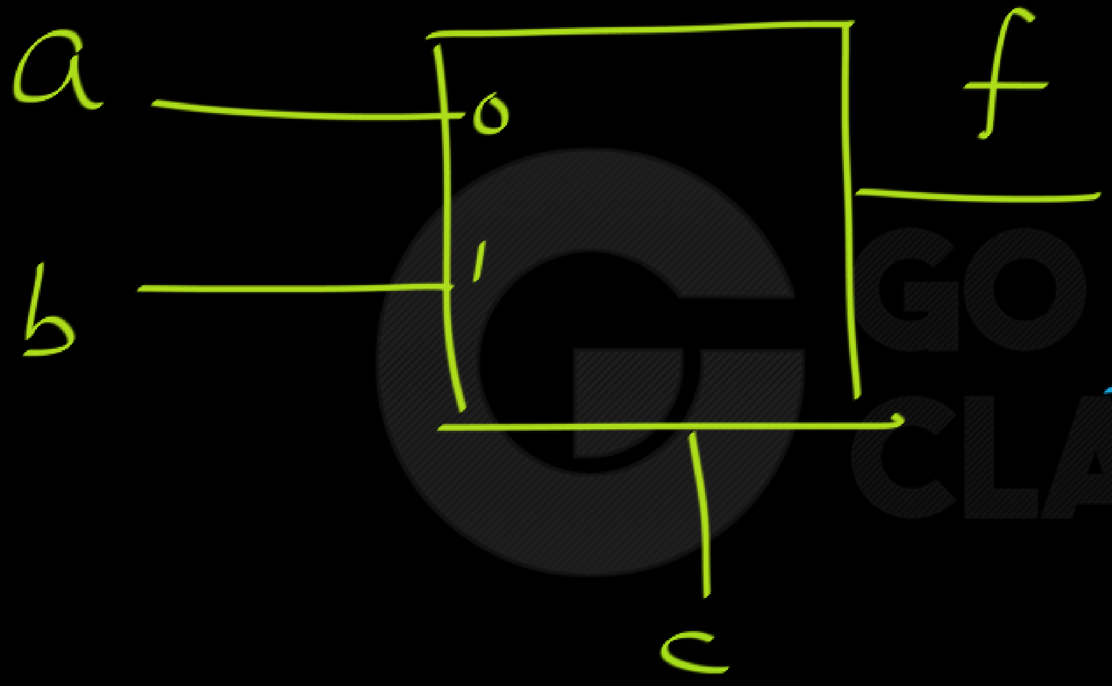
fun implemented by this mux?

$$f = \overline{s}a + sb$$



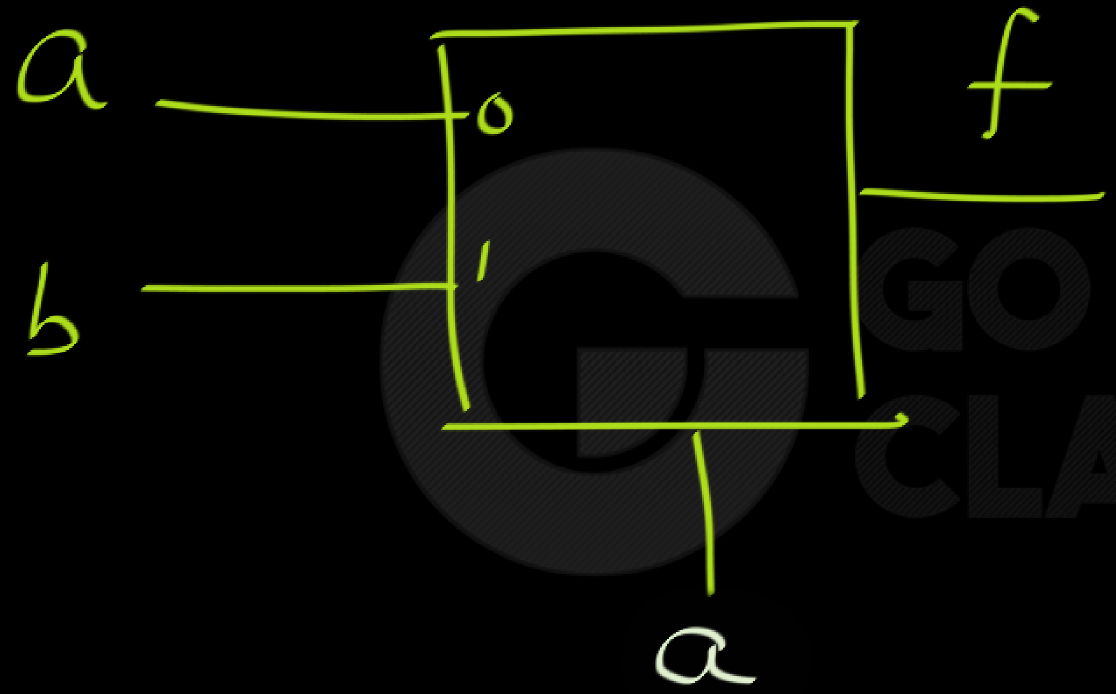
fun implementes by this mux?

$f(a,b,c) = ?$

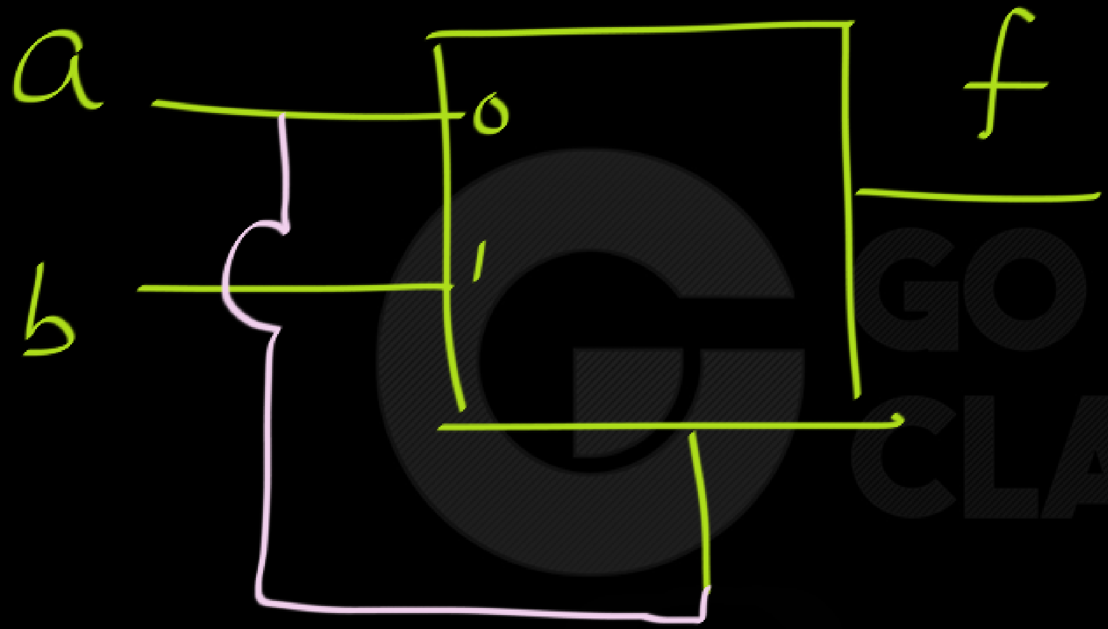


fun implementes by this mux?

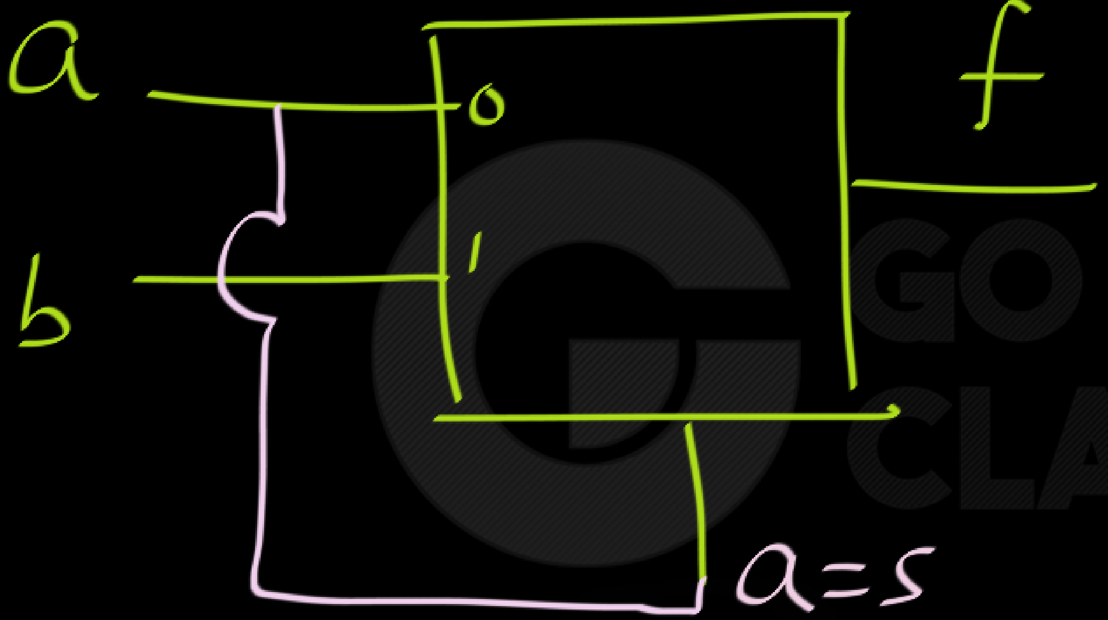
$f(a,b,c) = \bar{c}a + cb$



fun implementes by this mux?



fun implementes by this mux?



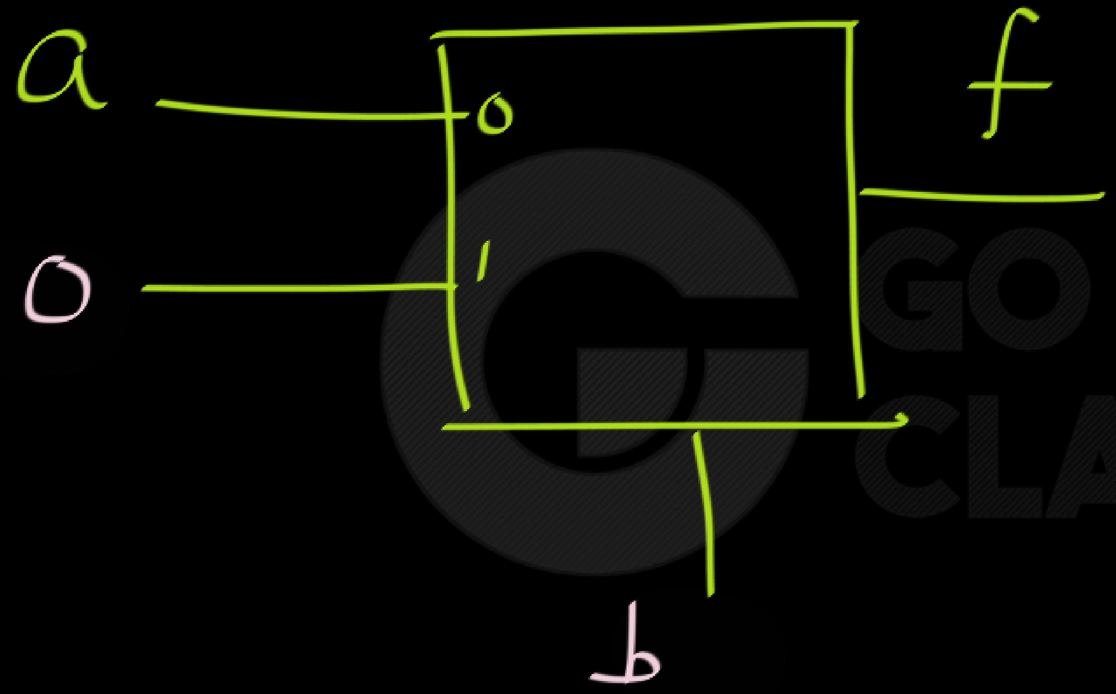
fun implementes by this mux?

$f(a,b) = \bar{s}a + sb$

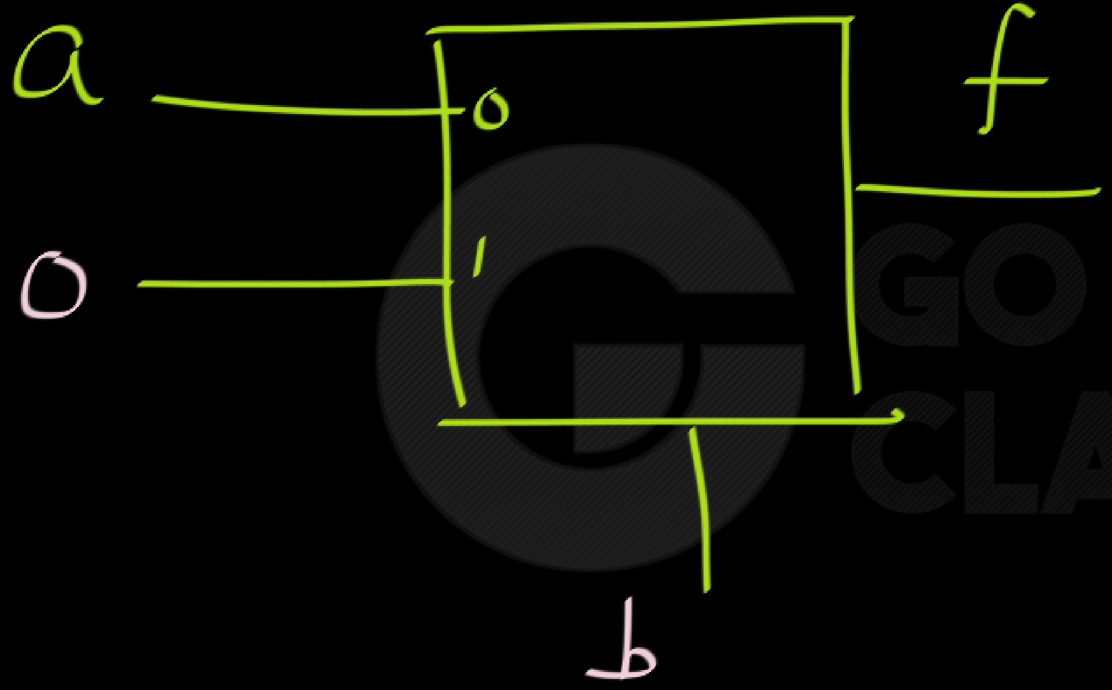
$= \bar{a}a + ab$

$= ab$ ✓

fun implementes by this mux?

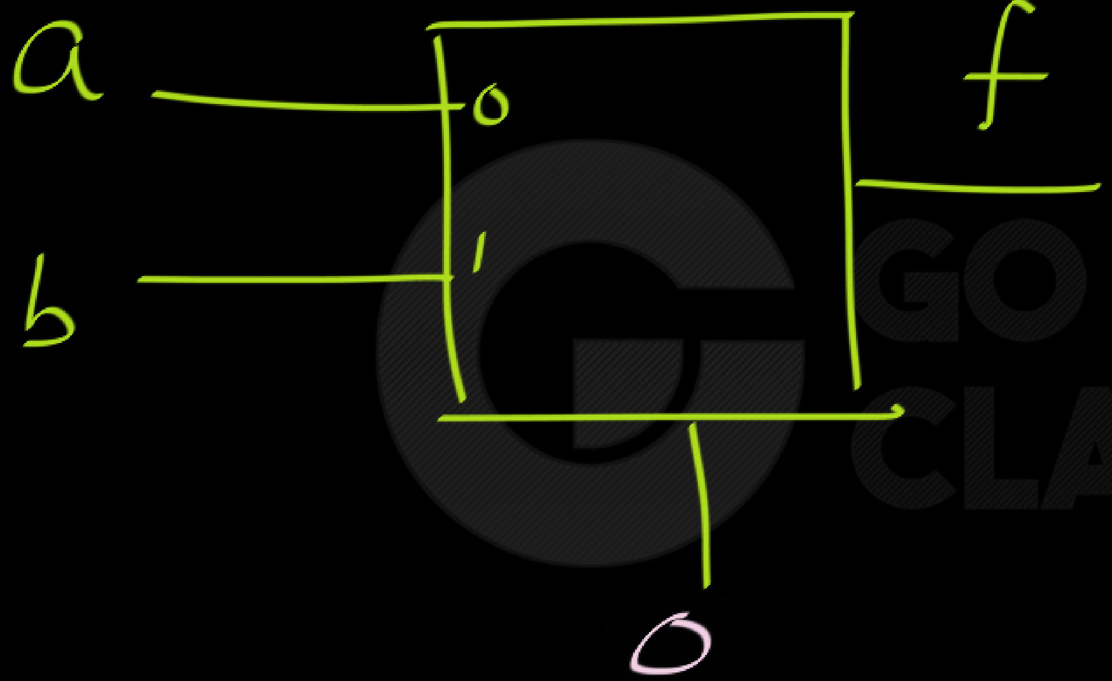


$f(a,b) =$



fun implementes by this mux?

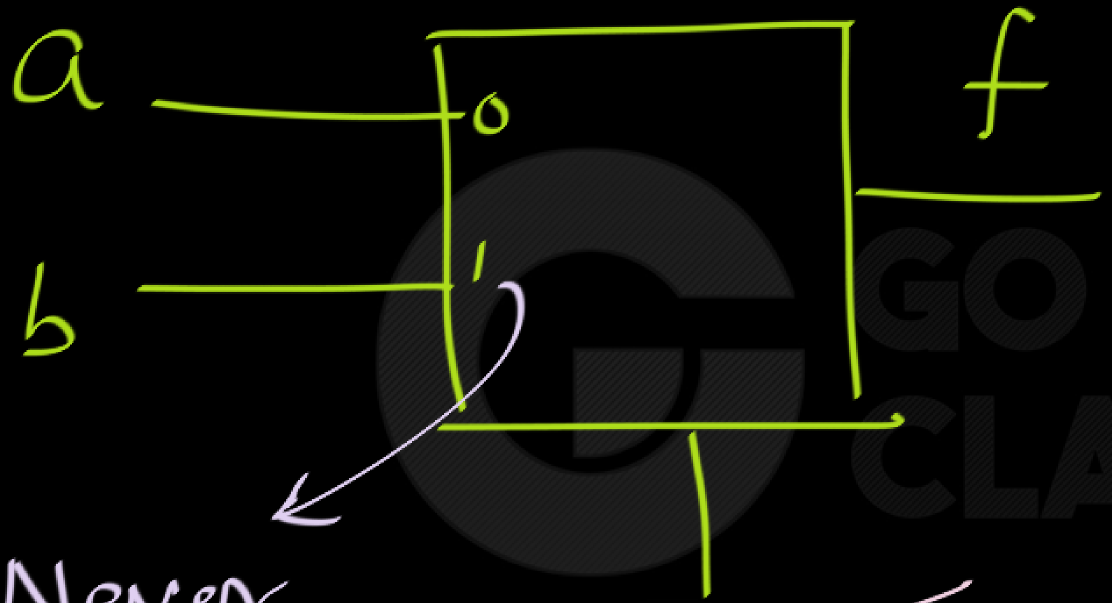
$$f(a,b) = \bar{b}a + b0$$
$$= \bar{b}a$$



fun implementes by this mux?

$f(a,b) = ?$

fun implementes by this mux?



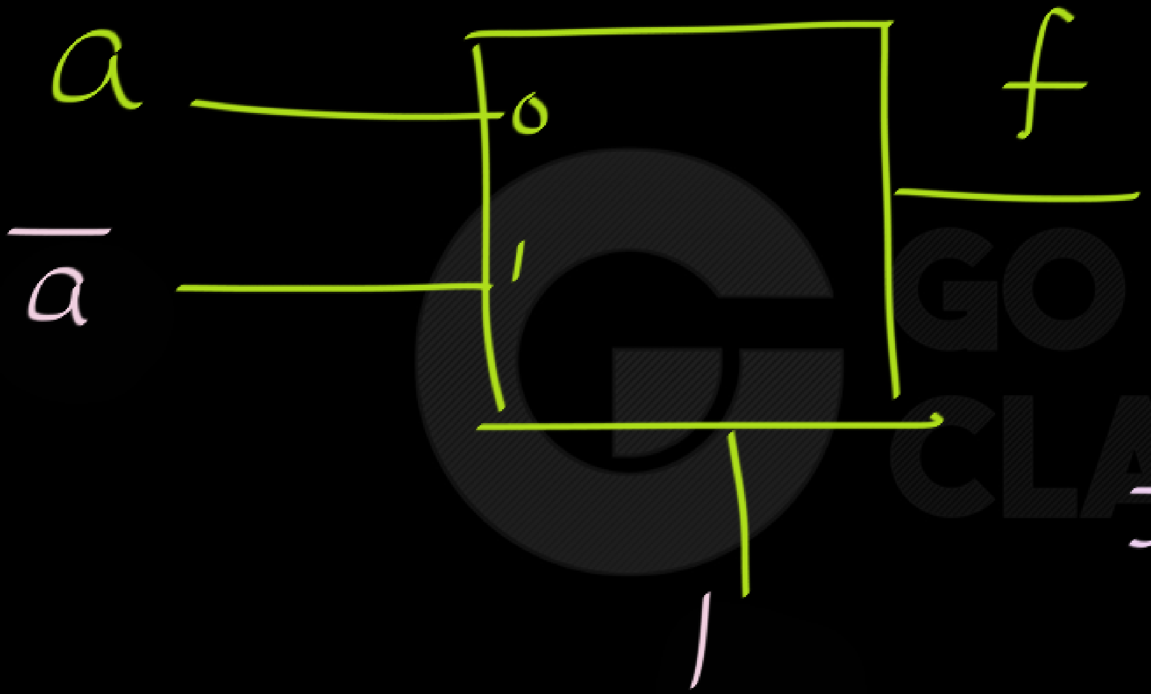
$f(a,b) = a$

Never be Activates

0 ✓

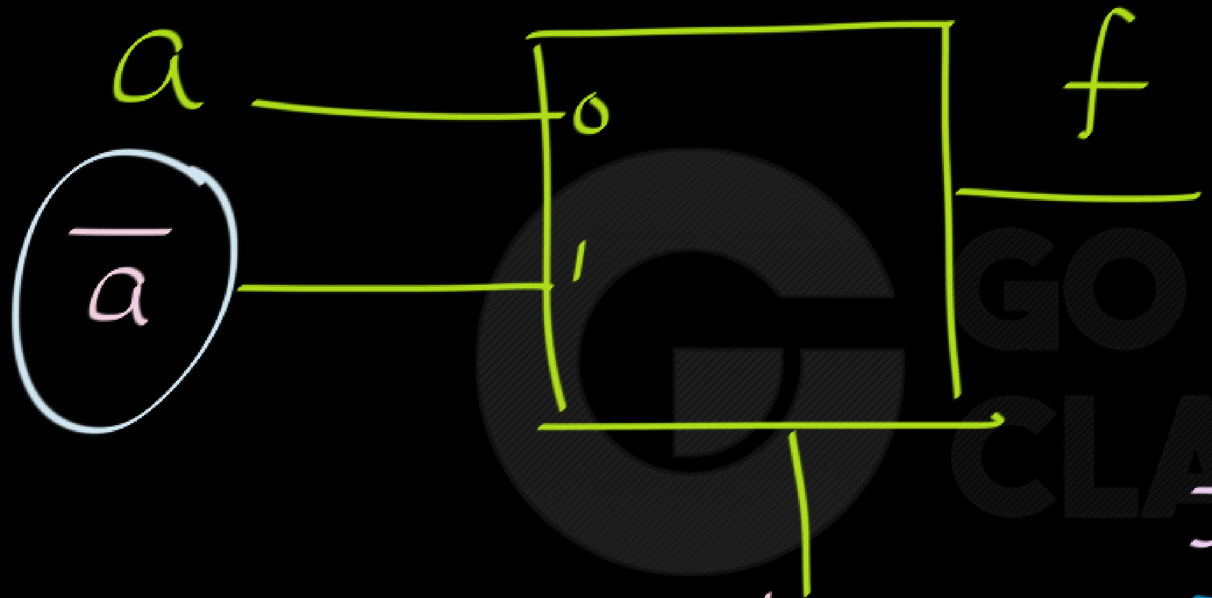
Grounded (low)

fun implementes by this mux?



$f(a) = ?$

fun implementes by this mux?

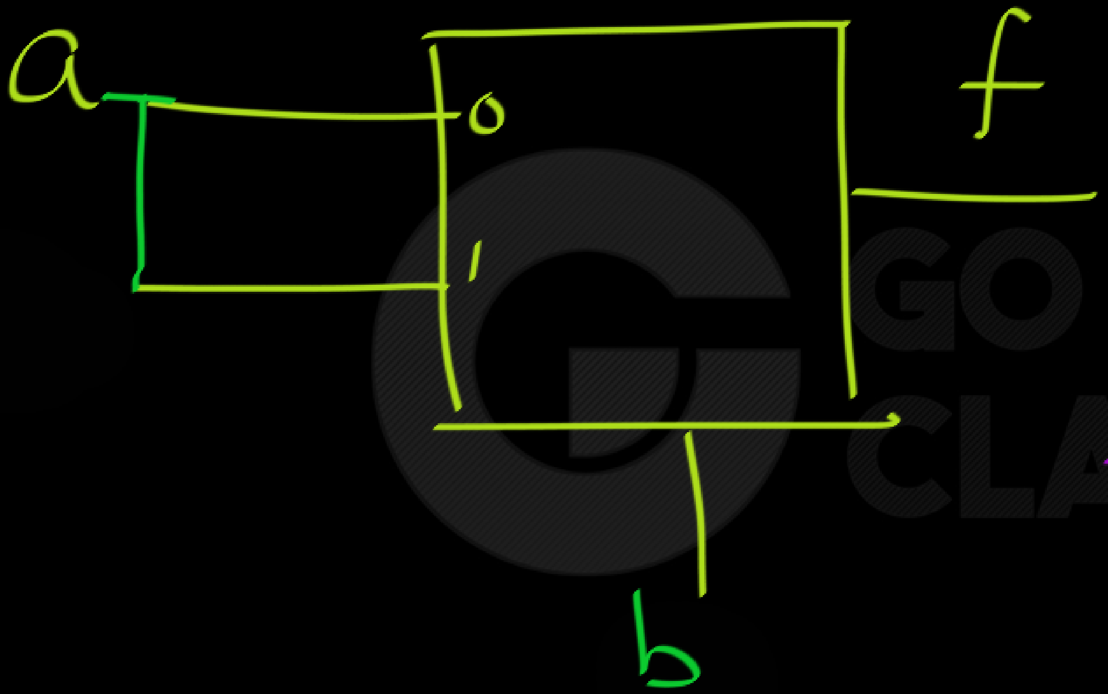


1

fix

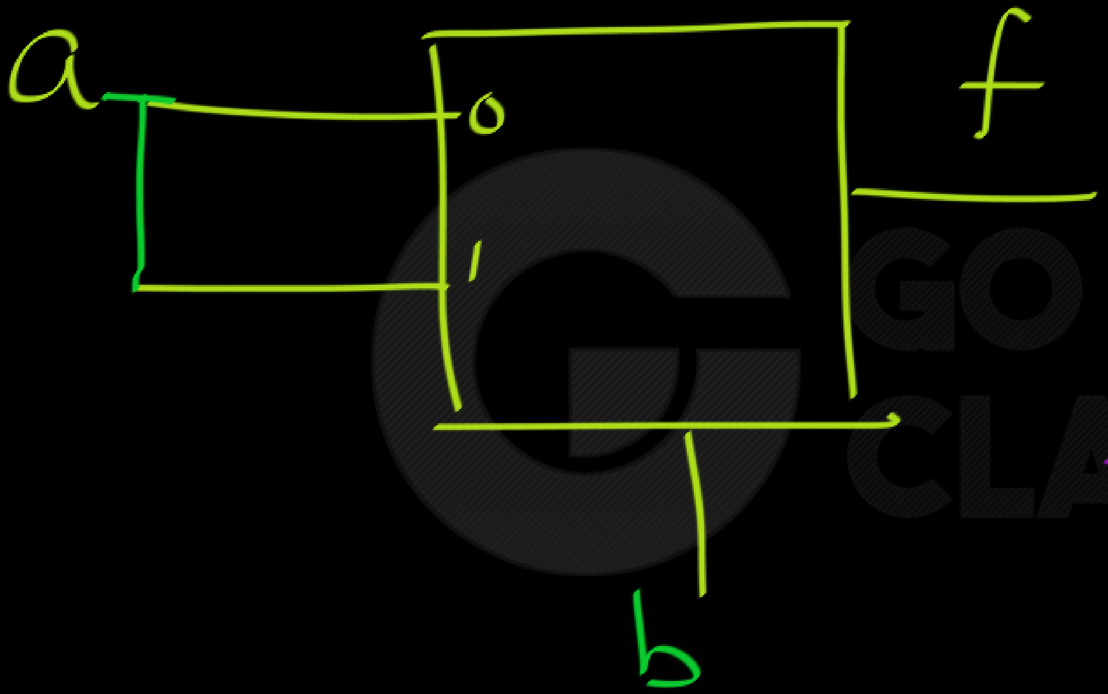
High

$f(a) = \bar{a}$



fun implementes by this mux?

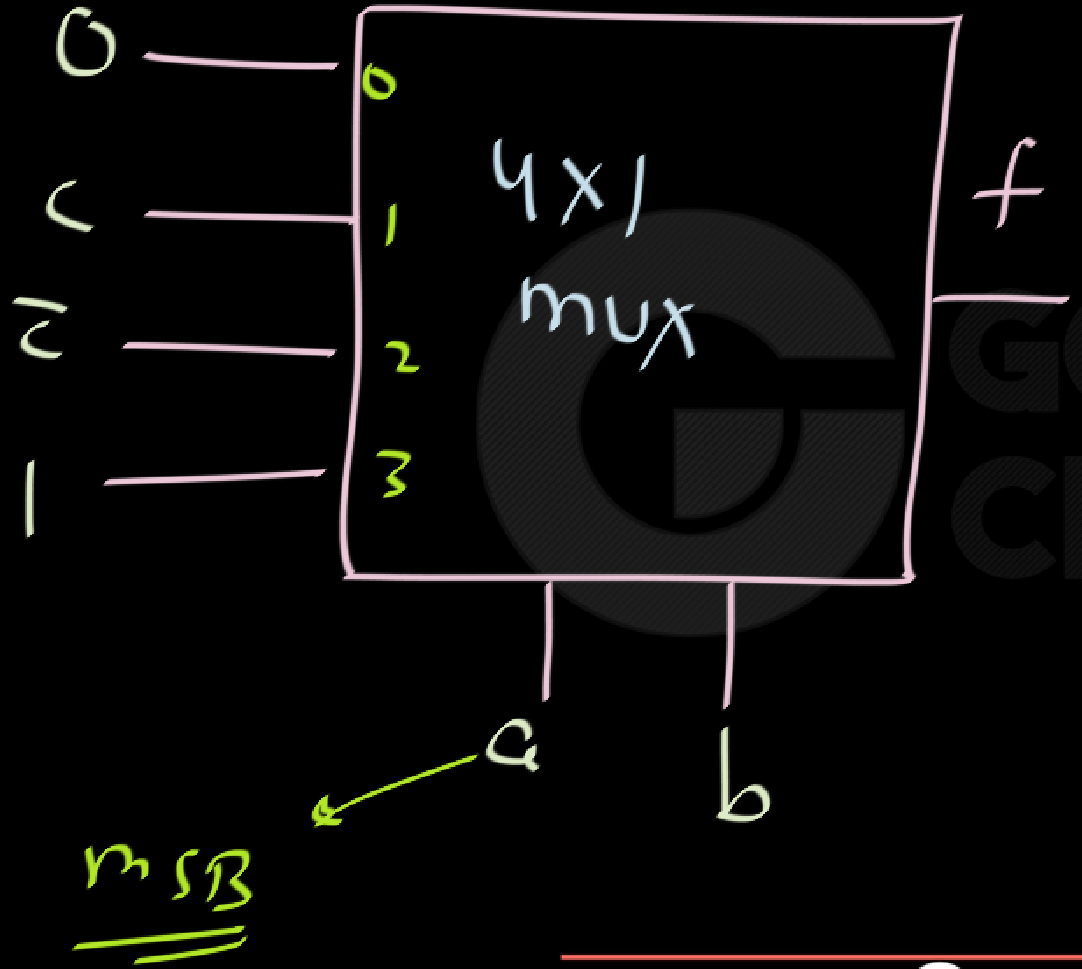
$f(a,b) = ?$



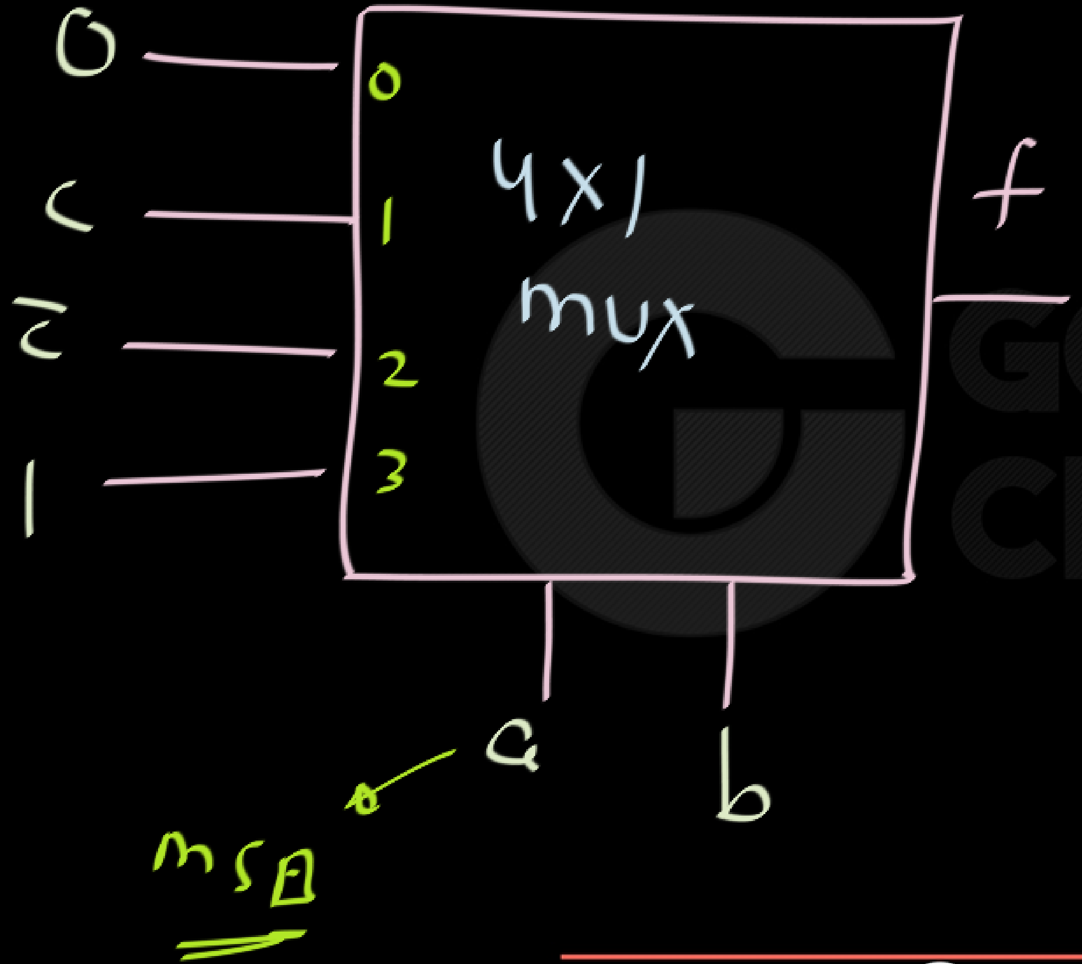
fun implemented by this mux?

$f(a,b) = \bar{b}a + ba$

$= a$



$f(a,b,c) = ?$



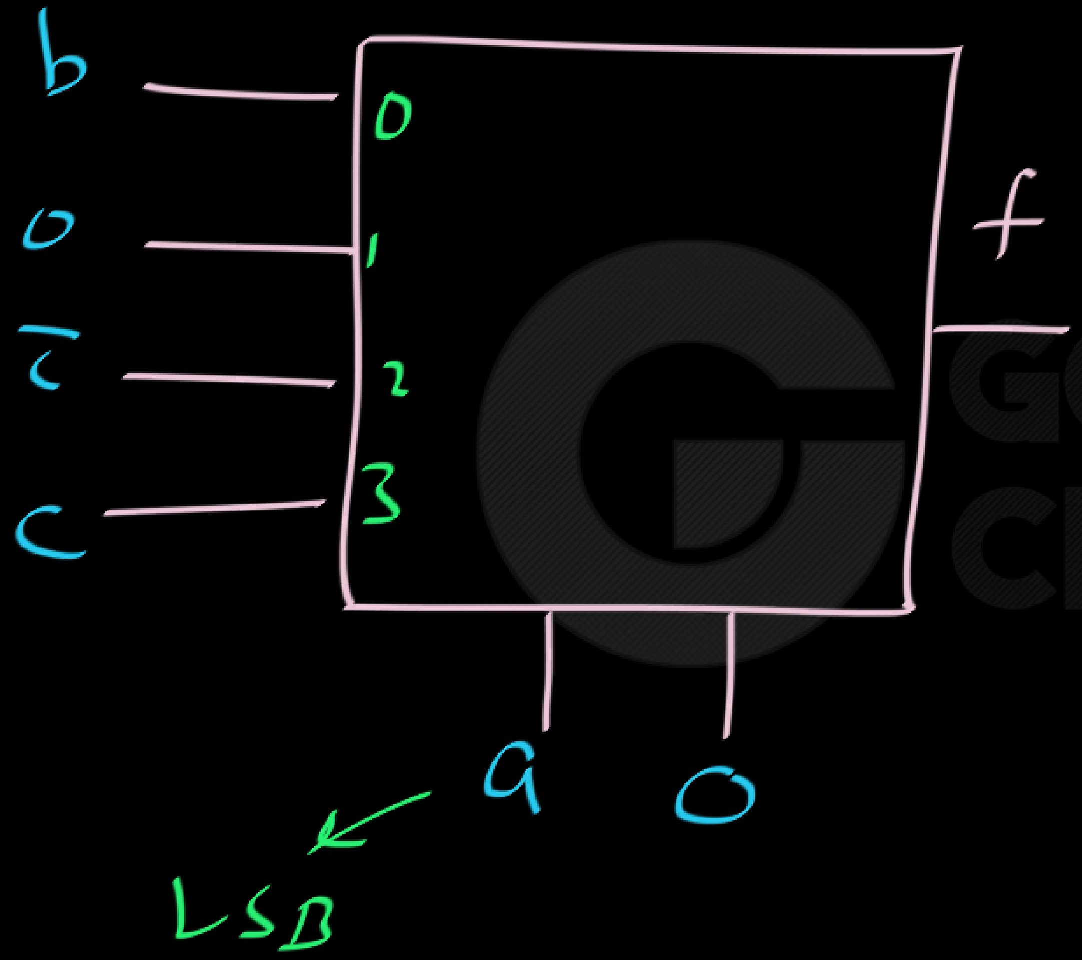
$$f(a,b,c) = \bar{a}\,\bar{b}\,0 + \bar{a}\,b\,c + a\,\bar{b}\,\bar{c} + a\,b\,1$$

$$f = \bar{a}bc + a\bar{b}\bar{c} + ab$$

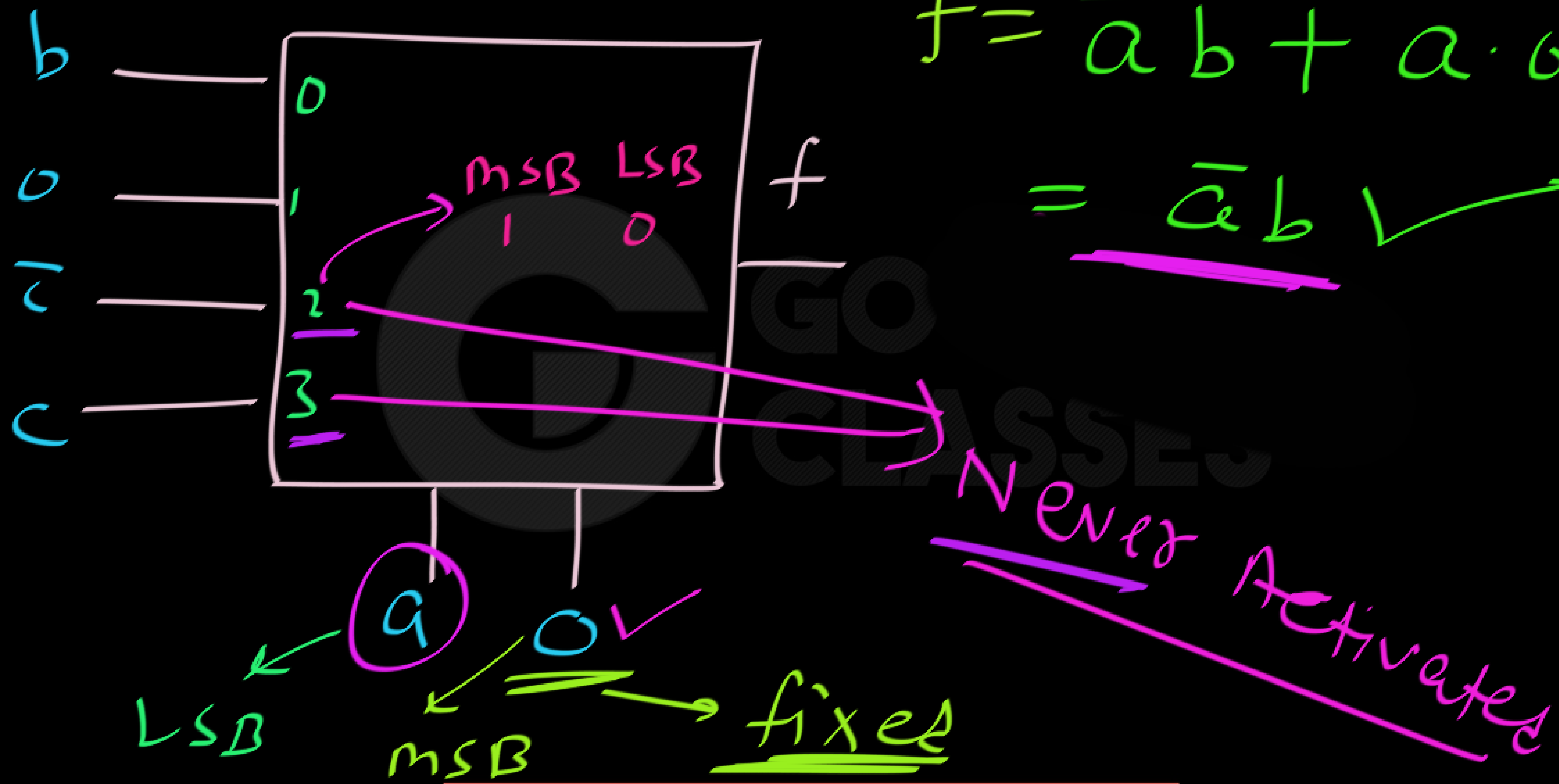
$$= \bar{a}bc + a(b + \bar{c})$$

$$= \bar{a}bc + ab + a\bar{c}$$

$$= ab + bc + a\bar{c} = \underbrace{bc + a\bar{c}}$$

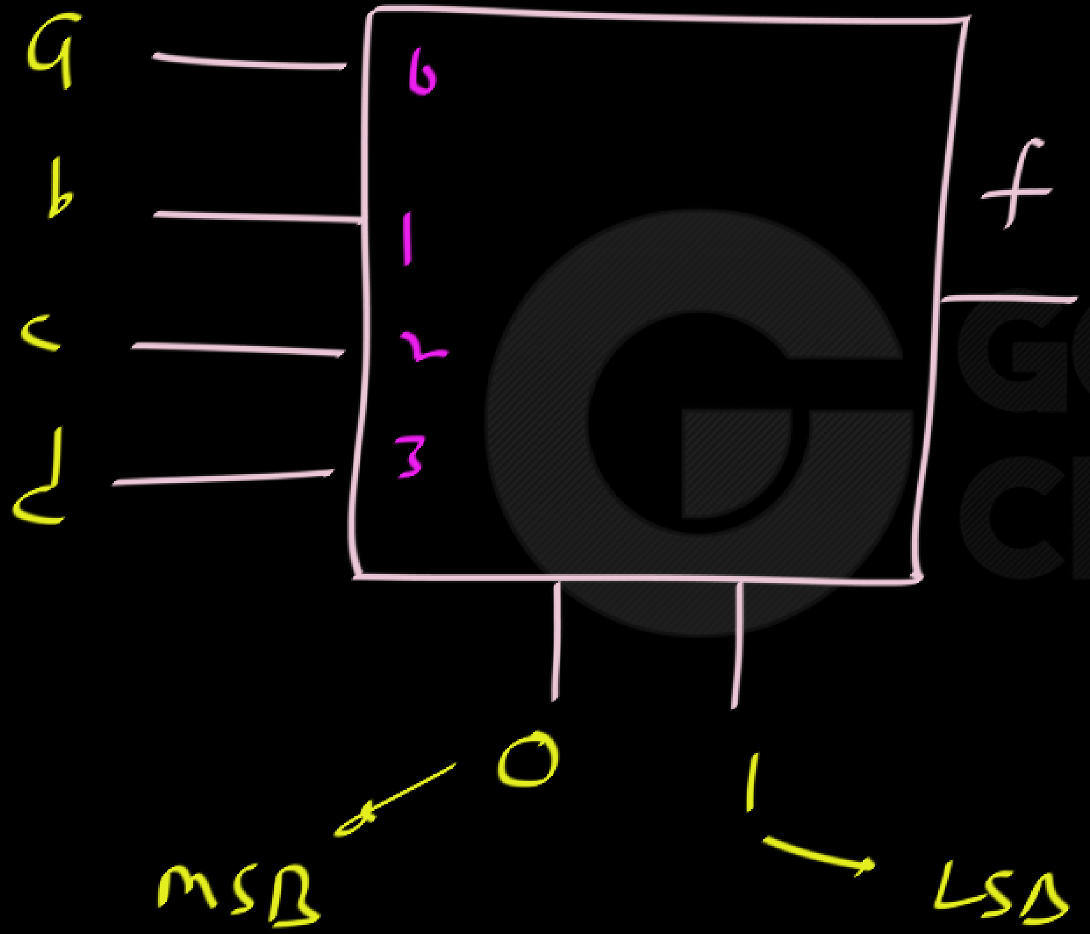


$f(a,b,c) = ?$

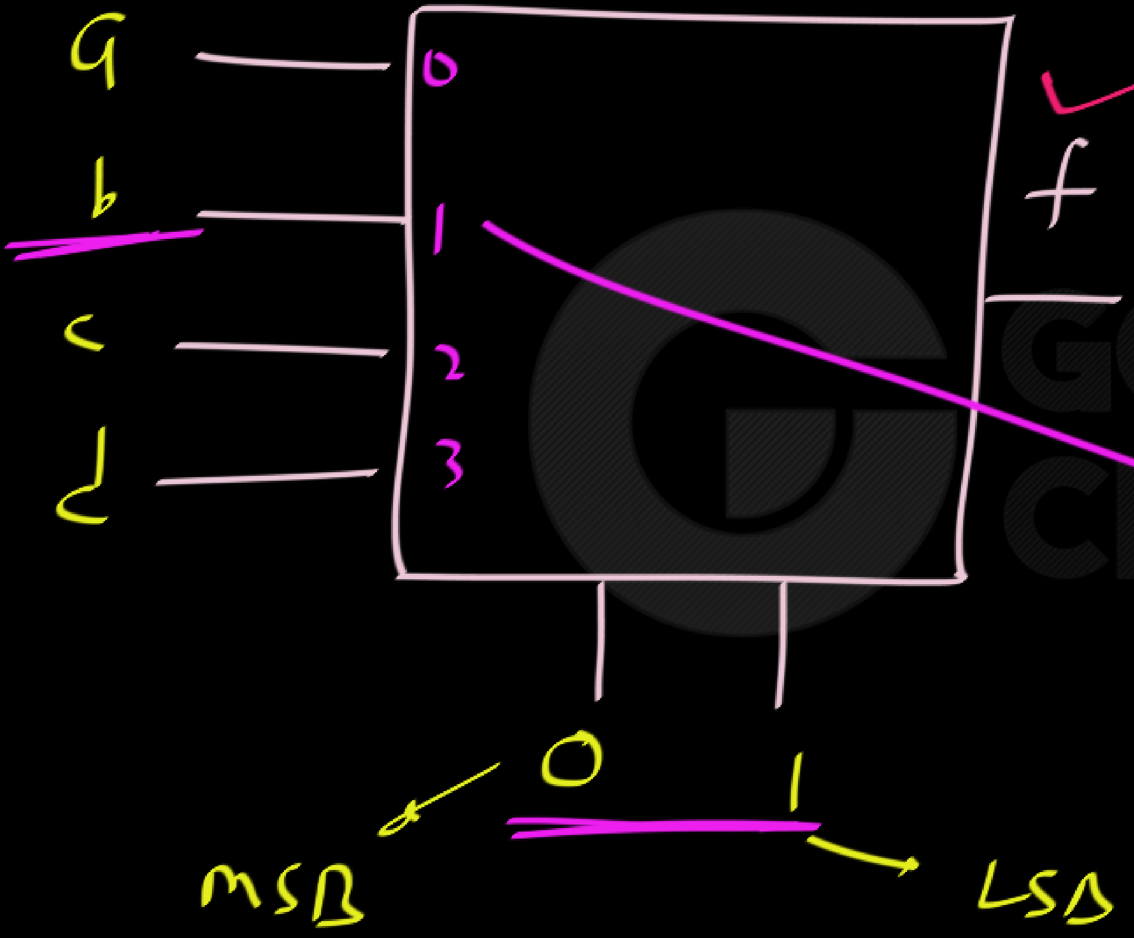


$f = \bar{a}b + a \cdot 0$

$= \bar{a}b$



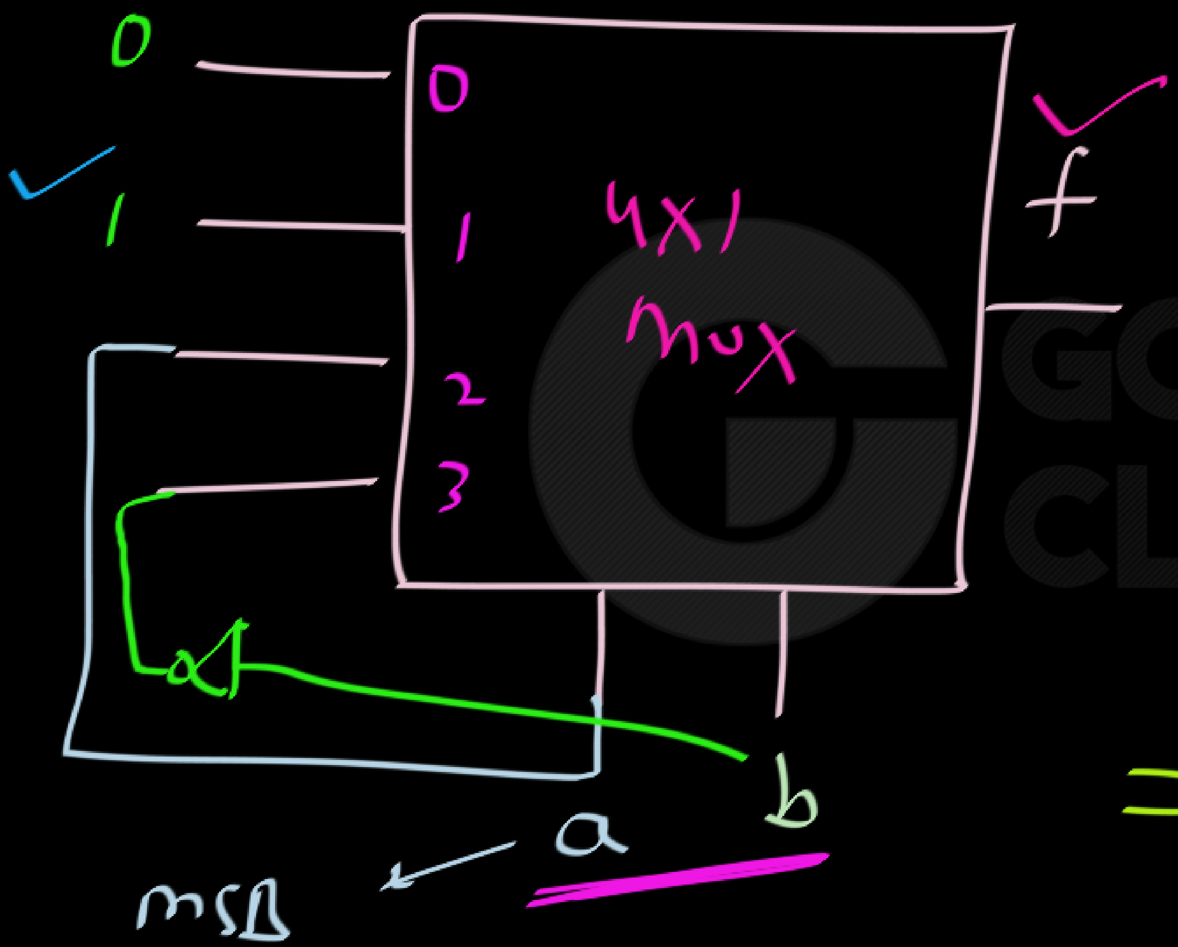
$f(a,b,c,d) = ?$



$f(a,b,c,d) = b$

Activate (Always)

$f(a,b) = ?$



$$f(a,b) =$$

$$\bar{a}\bar{b}0 + \bar{a}b1 + a\bar{b}a + ab\bar{b}$$

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

$$= a \oplus b$$

So far:

mux → function

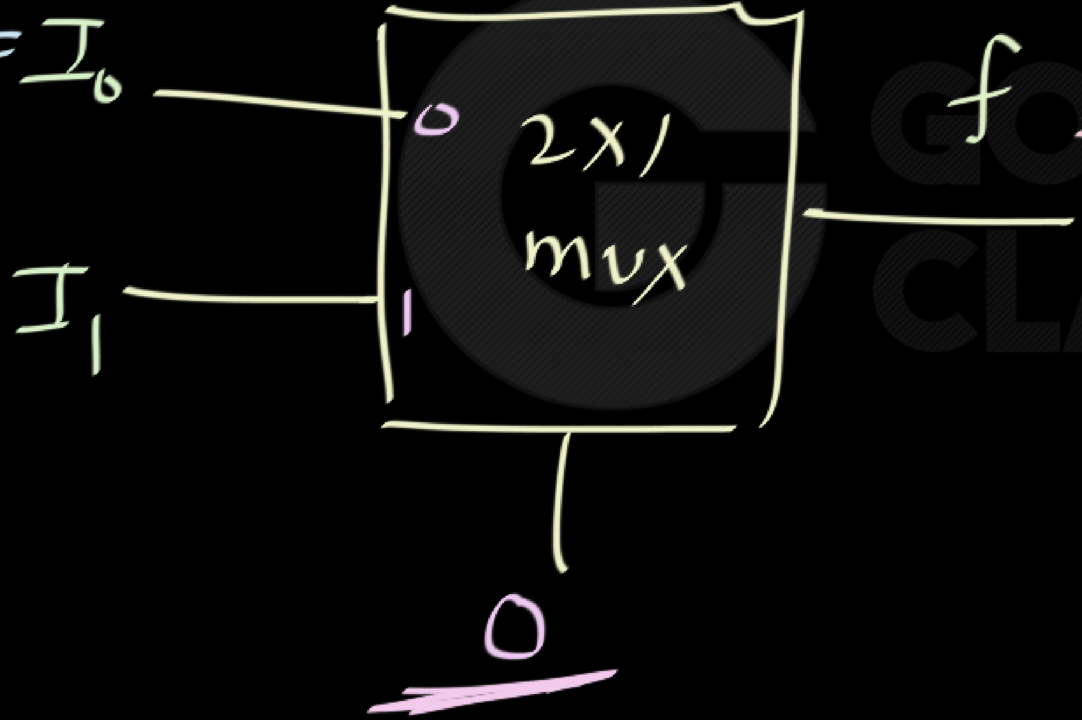
implemented ?

Q- Using 2 to 1 MUX implement the following 2-input gates: (a) OR (b) AND (c) NOR (d) NAND (e) XOR (f) XNOR (g) NOT.

Using 2x1 mux ; $f_{(a,b)} = a+b$ ✓

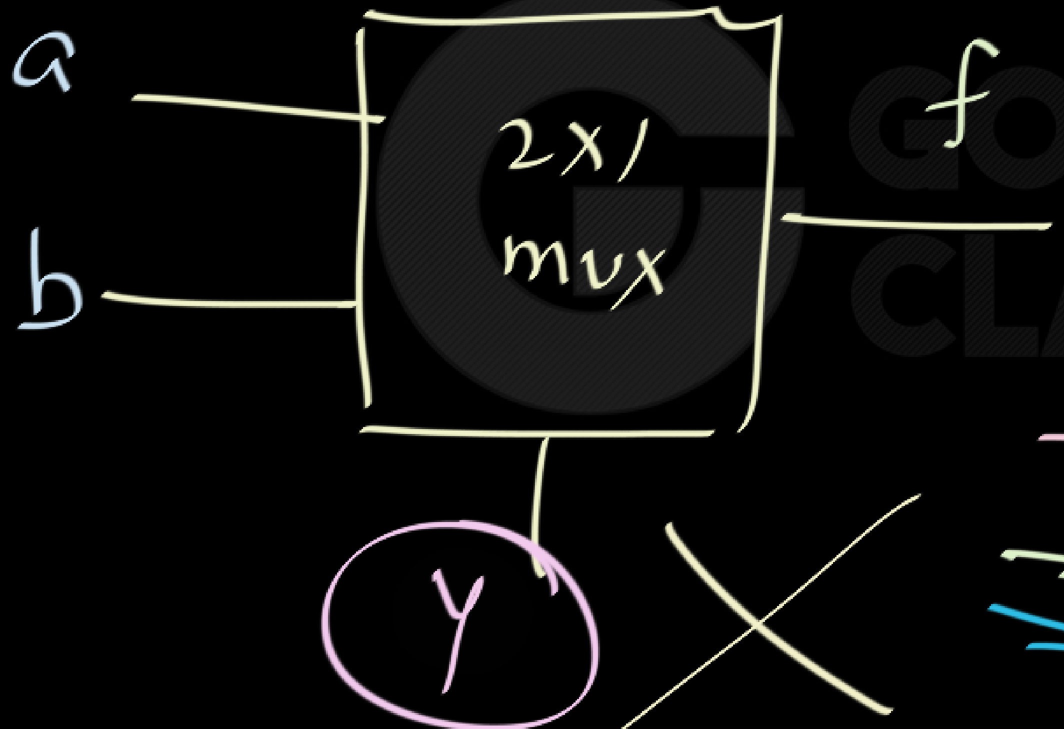
$a+b$ → OR ⇒ Extra circuit

$a+b = I_0$



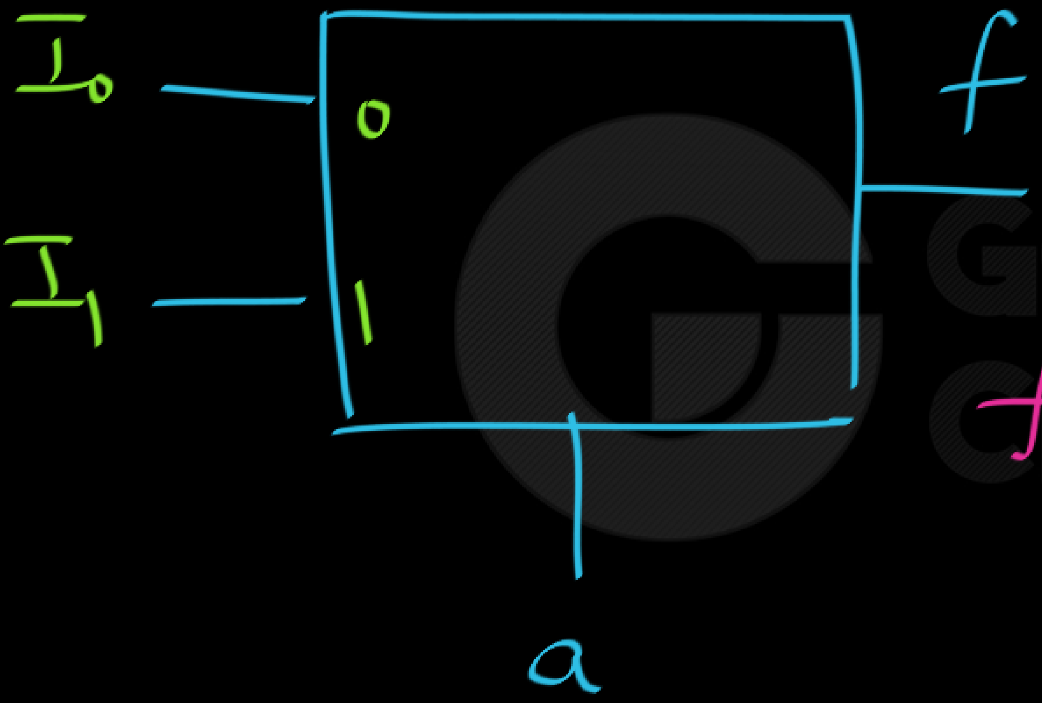
$f = I_0 = a+b$

using 2x1 mux ; $f(a,b) = a+b$



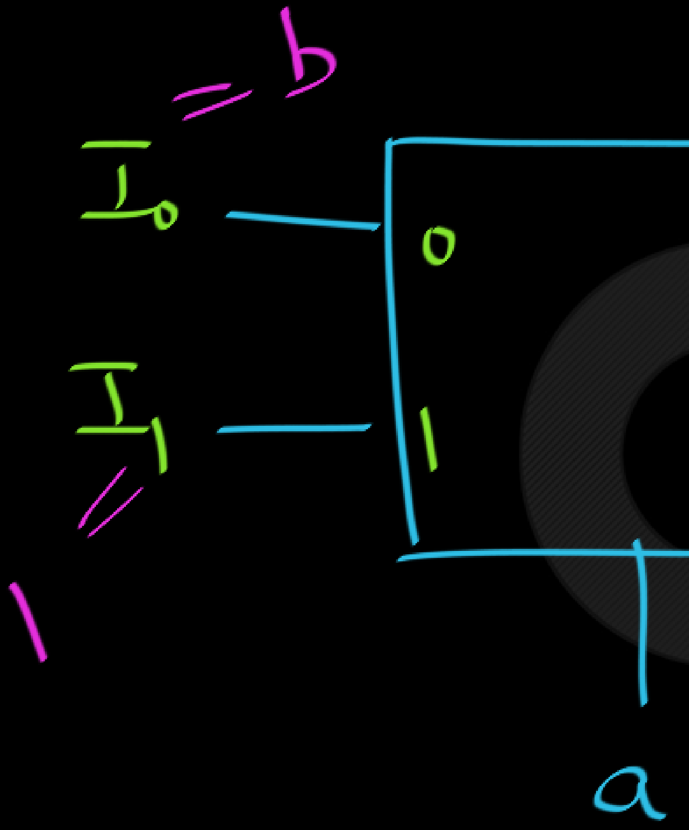
Don't use New Variables for select lines.

$\cancel{f(a,b,y)}$



$f = a + b$

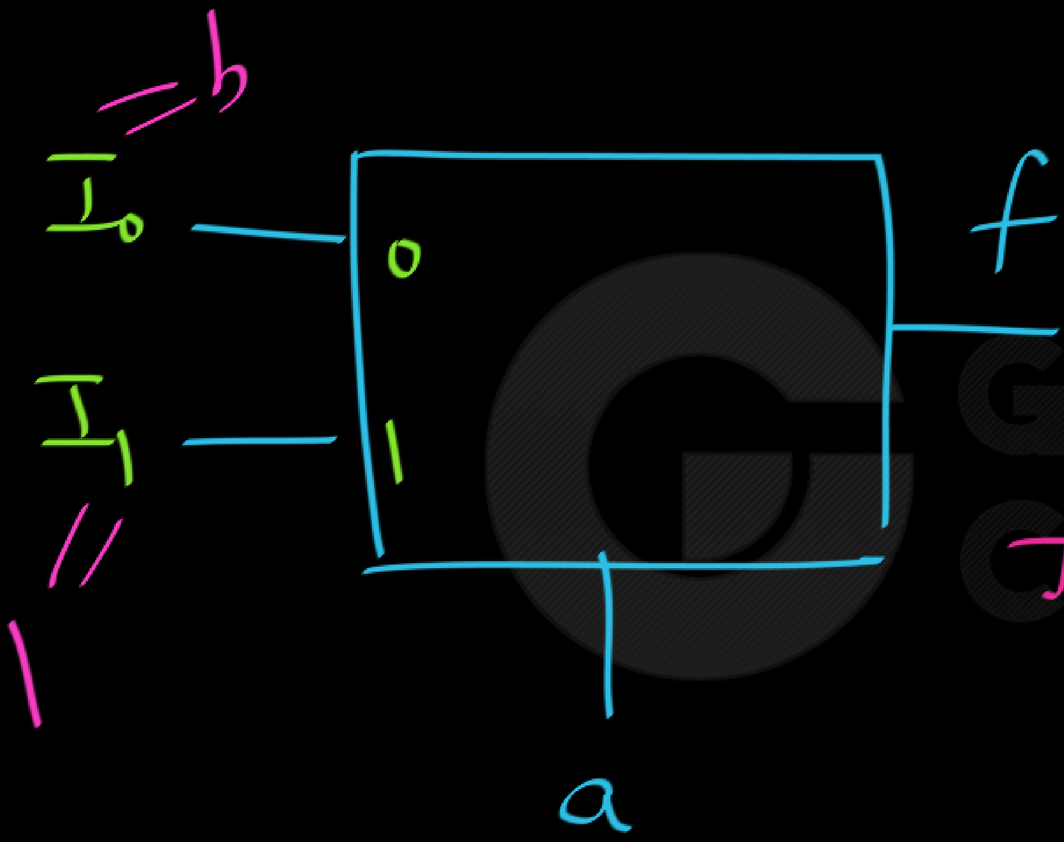
$f = \bar{a} I_0 + a I_1$



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

$$f = \bar{a} I_0 + a I_1$$

$I_0 \rightarrow b$, $I_1 \rightarrow 1$



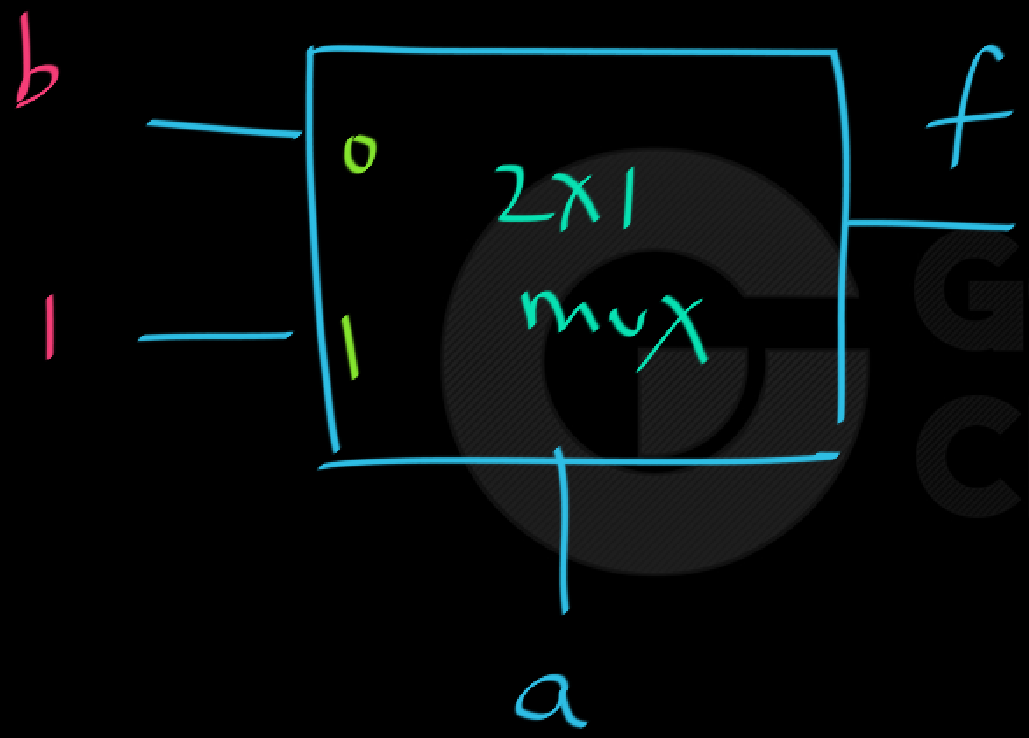
$f = a + b = \bar{a}b + a\bar{b} + ab$

Csop of $a+b$

$f = \bar{a}I_0 + aI_1$

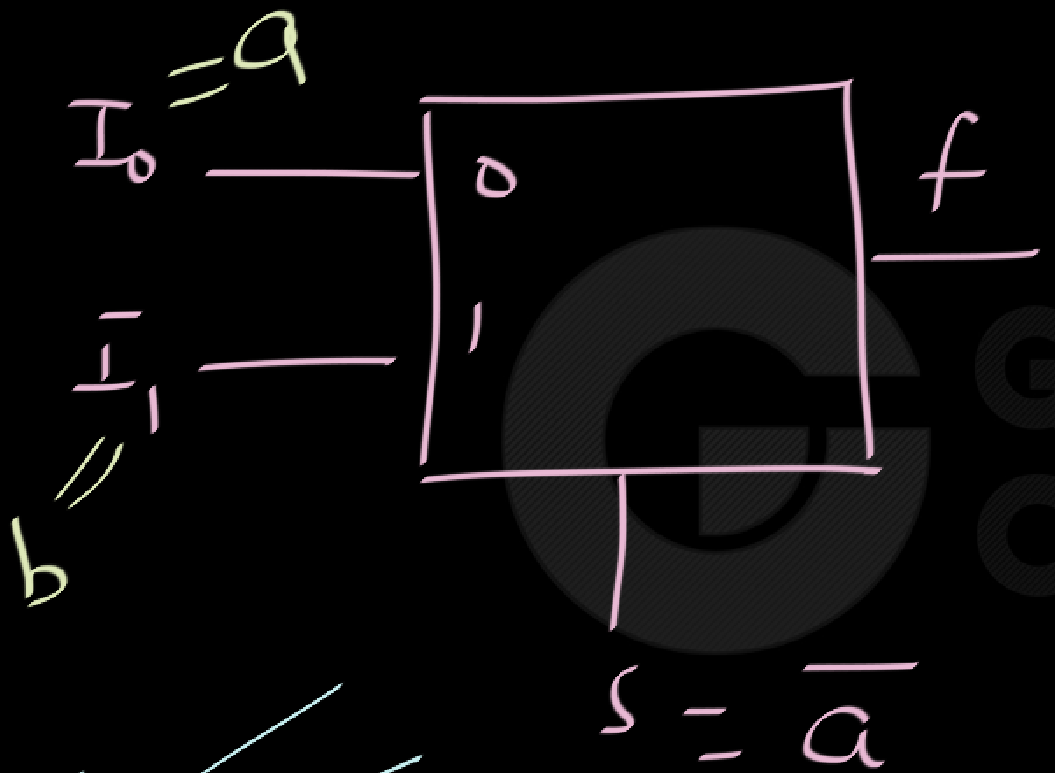
$I_0 \to b$

$I_1 \to 1$

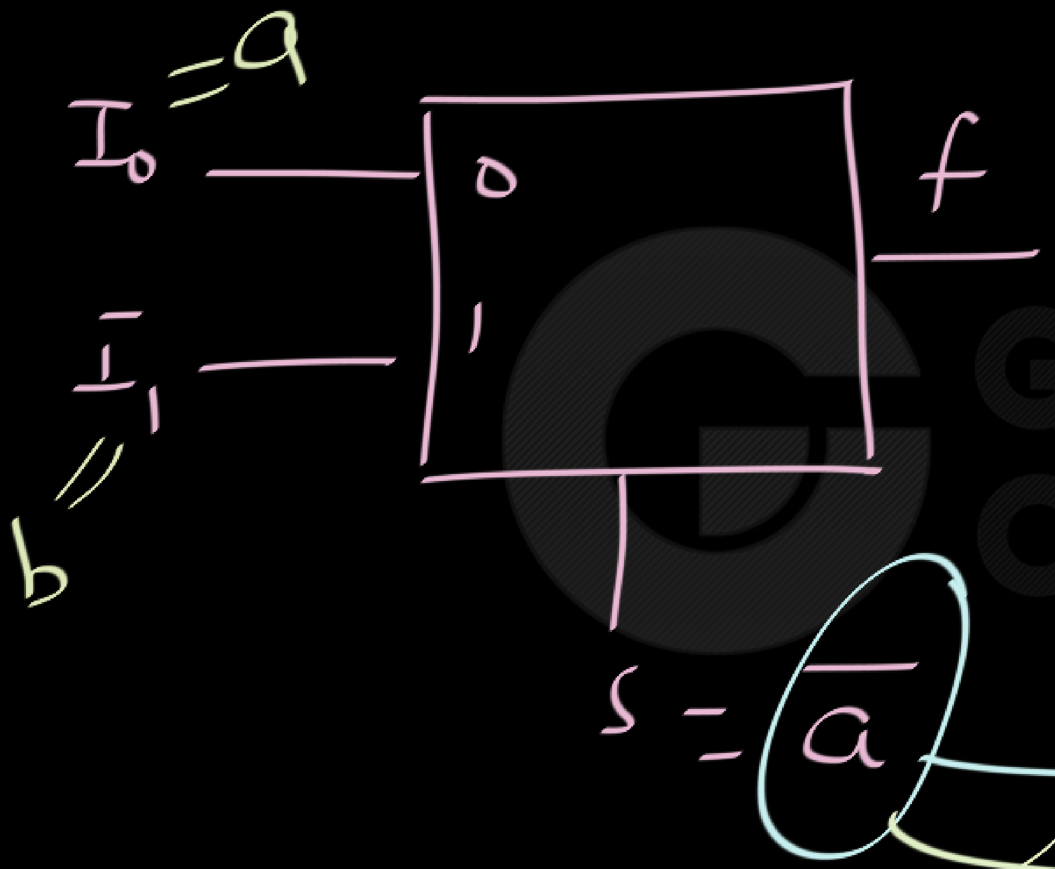


$f = a + b$ ✓

No Extra Hardware



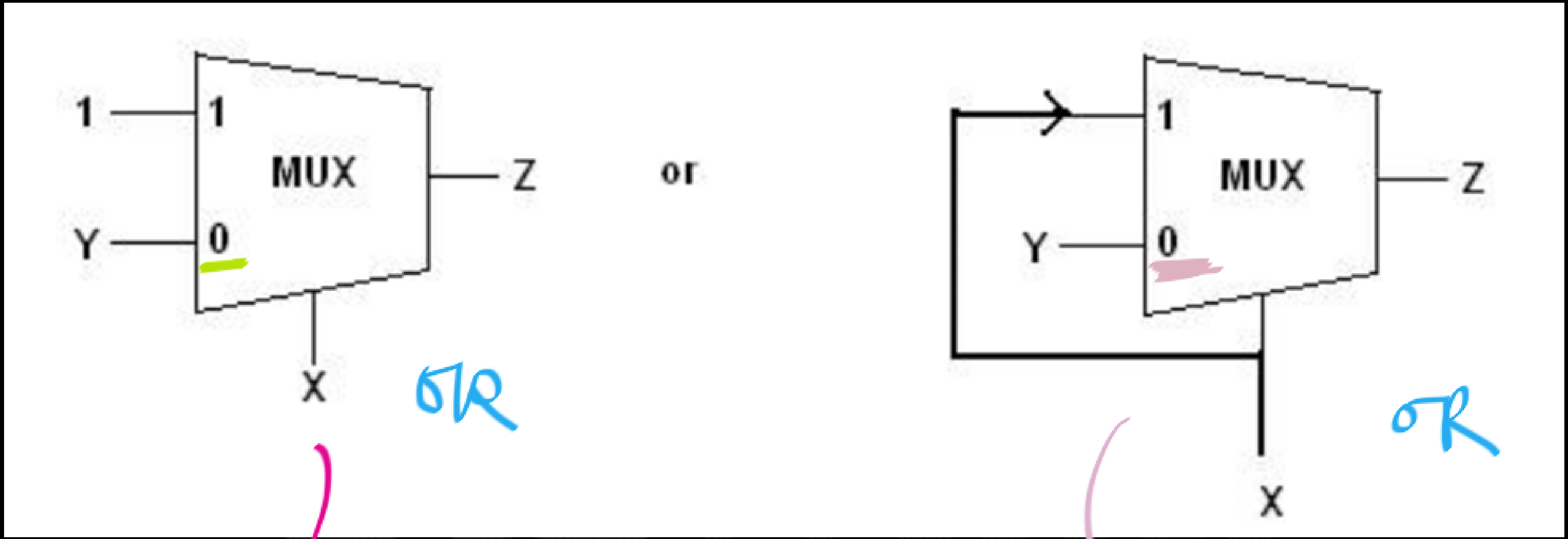
$$f = \bar{s} I_0 + s I_1$$
$$= a a + \bar{a} b$$
$$= a + \bar{a} b$$
$$= a + b$$



Not a ideal Circuit

Extra Hardware

Inverter

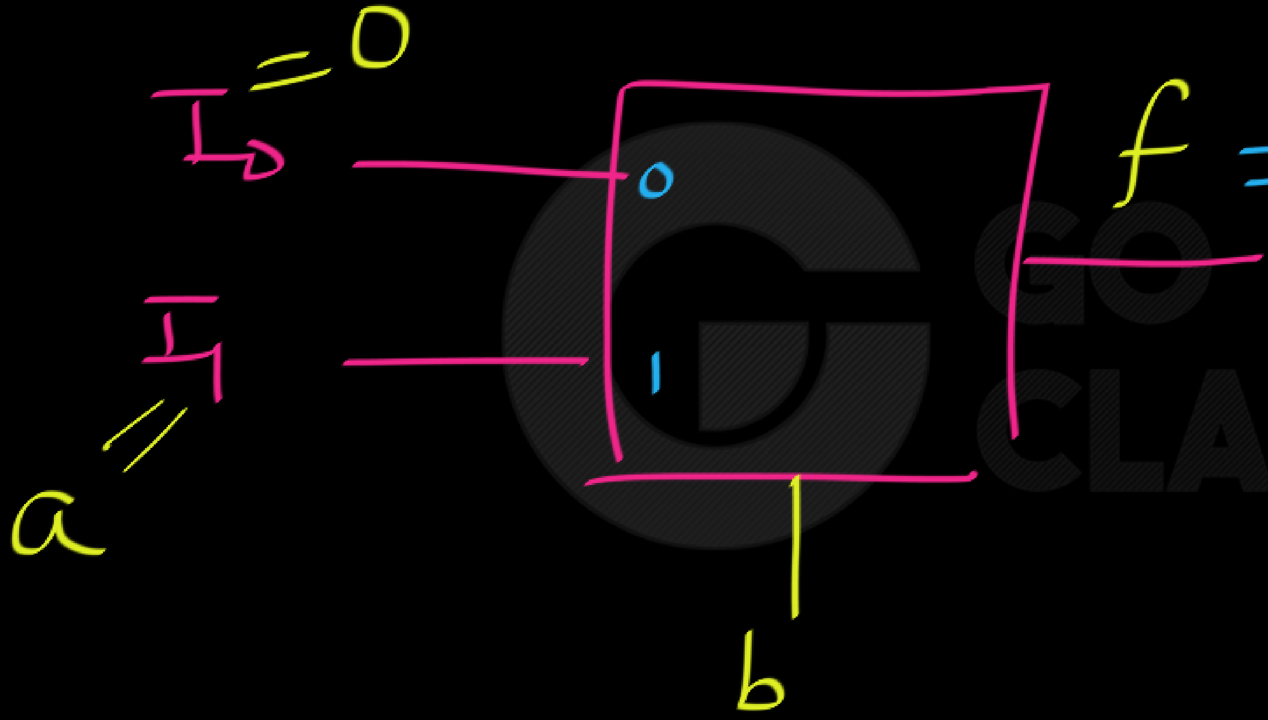


$$Z = \bar{x}y + x \cdot 1$$
$$= x + \cancel{\bar{x}}y = x + y$$

$$Z = \bar{x}y + xx$$
$$= x + \bar{x}y = x + y$$

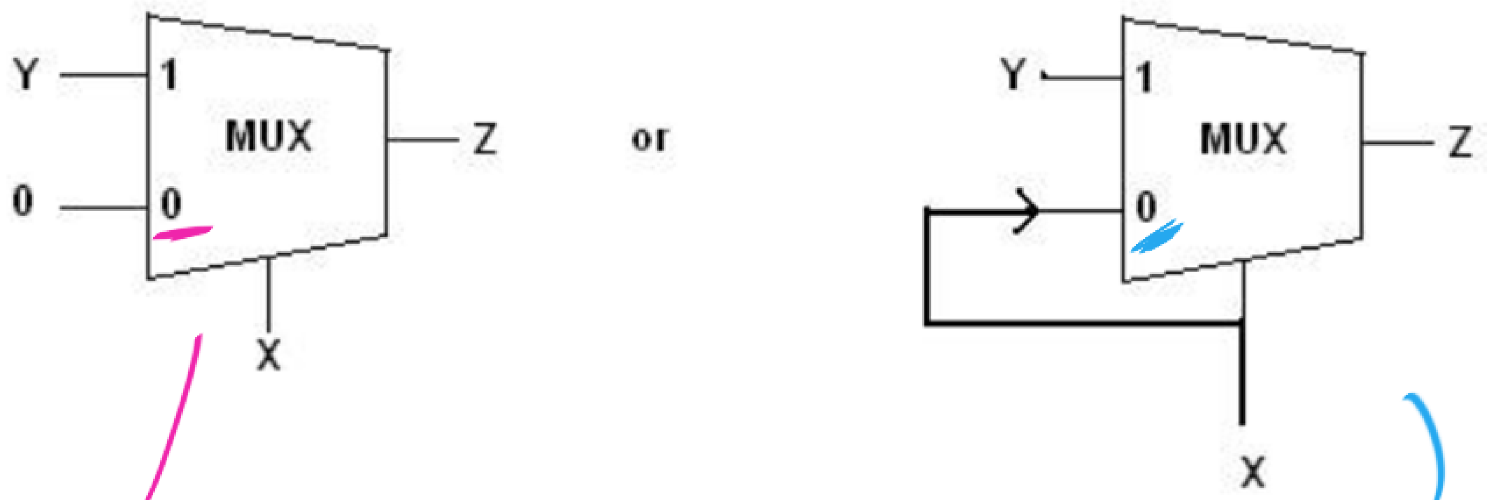
AND:

$f = ab$



$f = \bar{b}\,\underbrace{I_0}_{0} + b\,\underbrace{I_1}_{a}$

(b) AND: Similar to the case of OR gate we can derive the circuit for AND gate as below:



$Z = \bar{x}0 + xy = xy$

$Z = \bar{x}x + xy$
$= xy$ ✓

NOR : HW :

NAND : HW

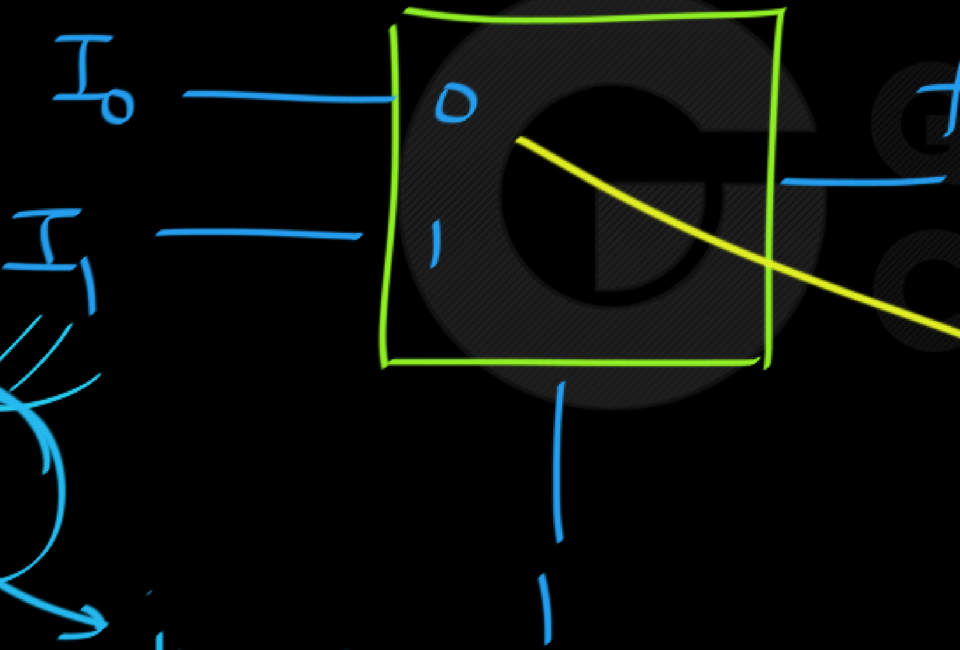
ExOR : HW

ExNOR : HW

NOT : HW

$f = \overline{a}$; $f(a) = \overline{a}$

Not a good Circuit

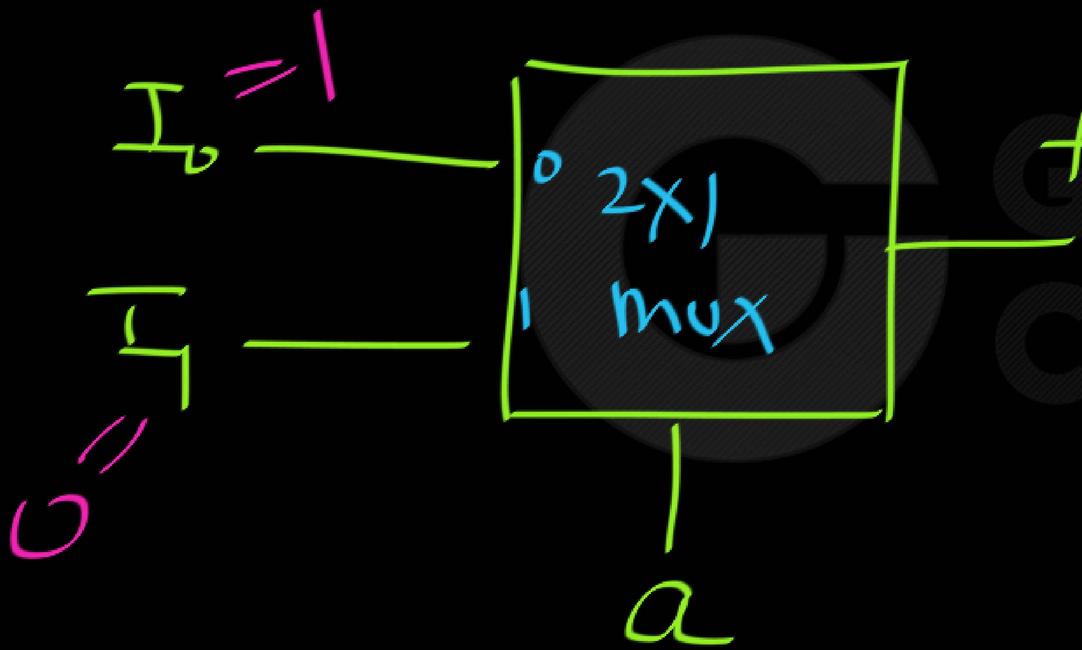


$f = I_1 = \overline{a}$

$\overline{a}$ → Inverter

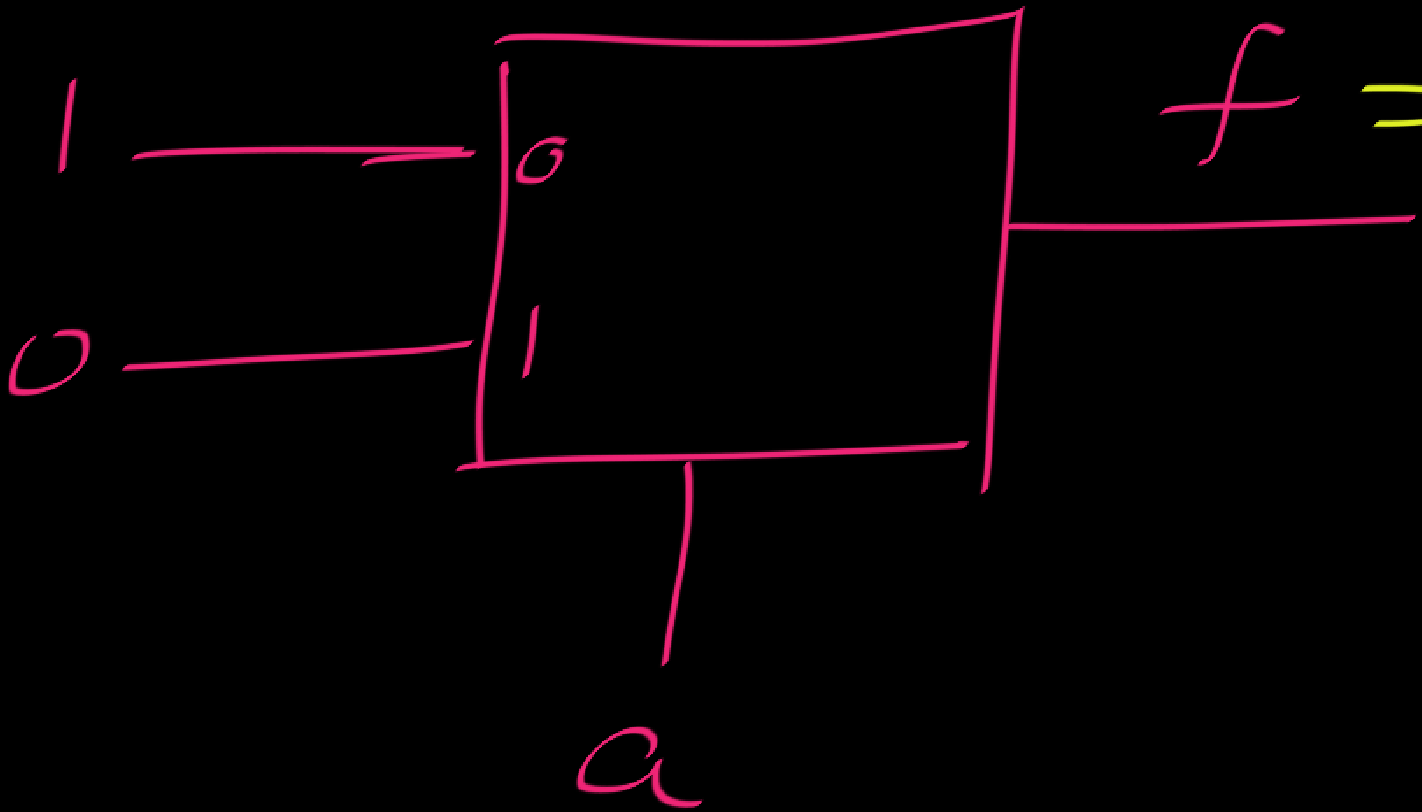
Never Activates

$f = \overline{a}$; $f(a) = \overline{a}$



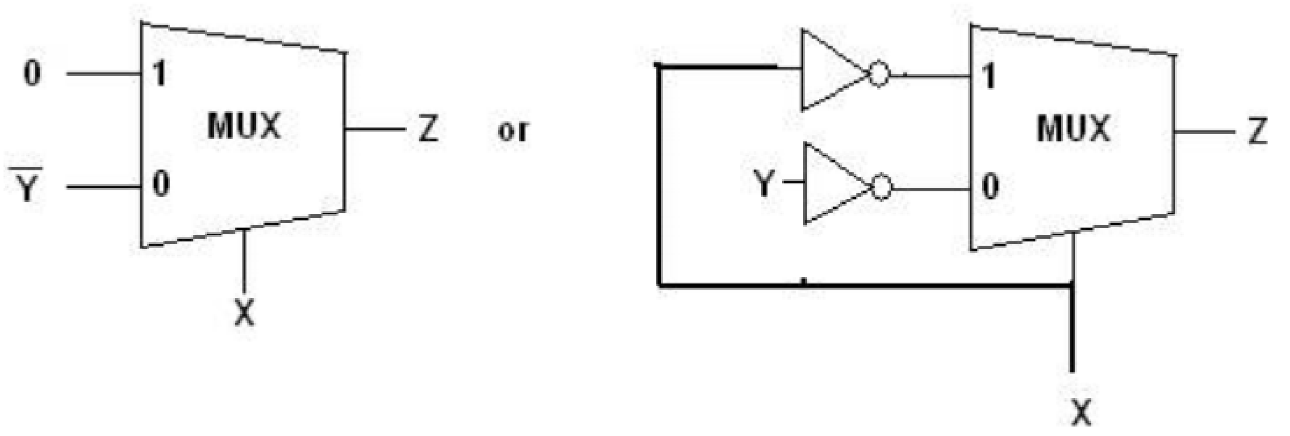
$f = \overline{a} I_0 + a I_1$

$I_0 = 1$, $I_1 = 0$

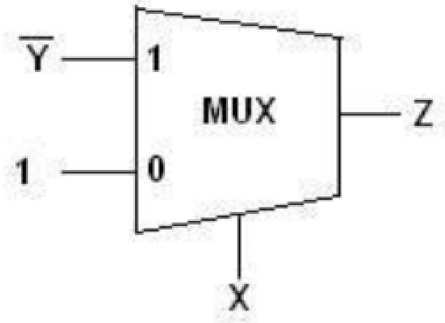


$f = \overline{a}$

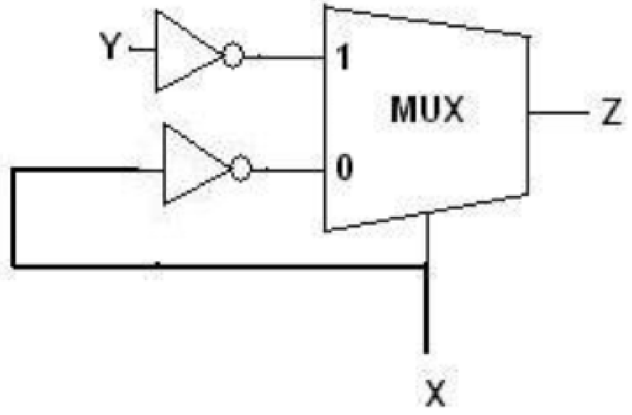
(c) NOR



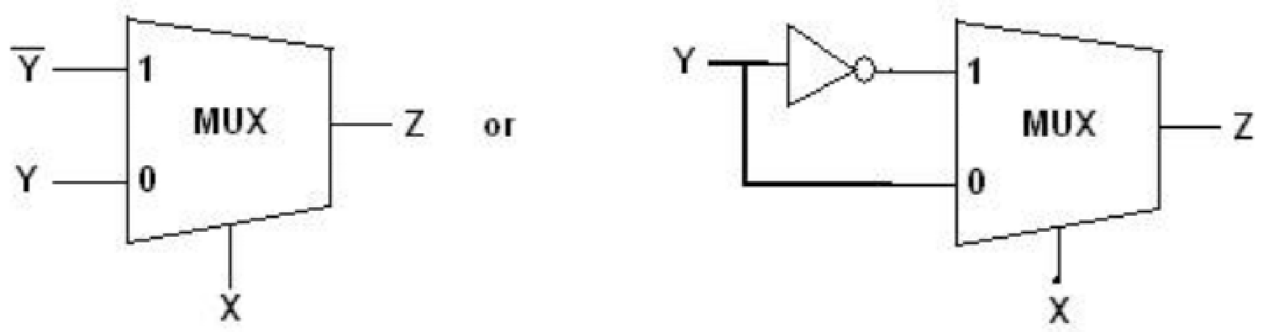
(d) NAND



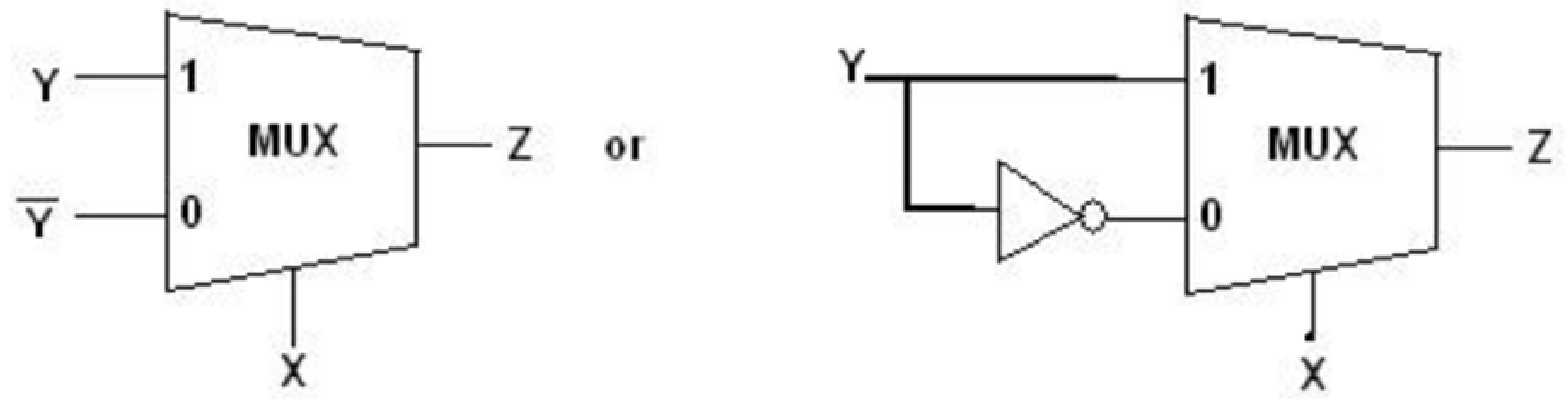
or



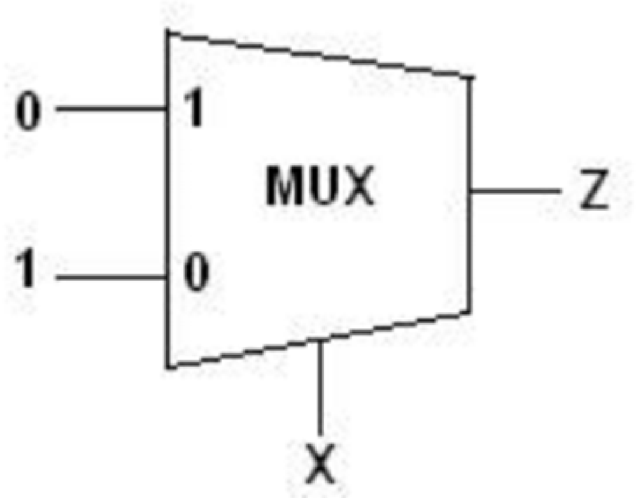
(e) XOR



(f) XNOR



(g) NOT.

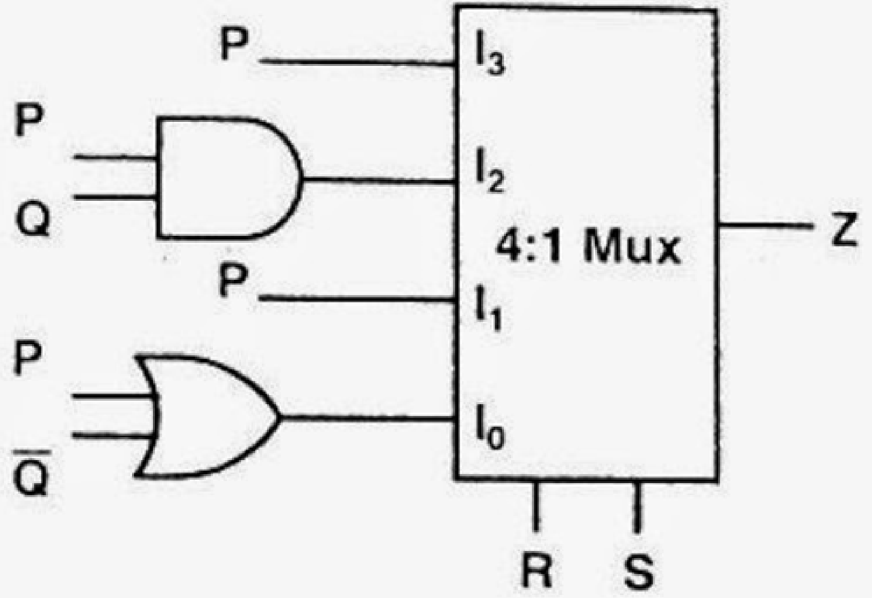


Consensus Property (Emotional Property Happy - SaD "")

$$a b + \bar{a} c + bc = ab + \bar{a}c$$

Pawri going on

Conspiracy against 'a'

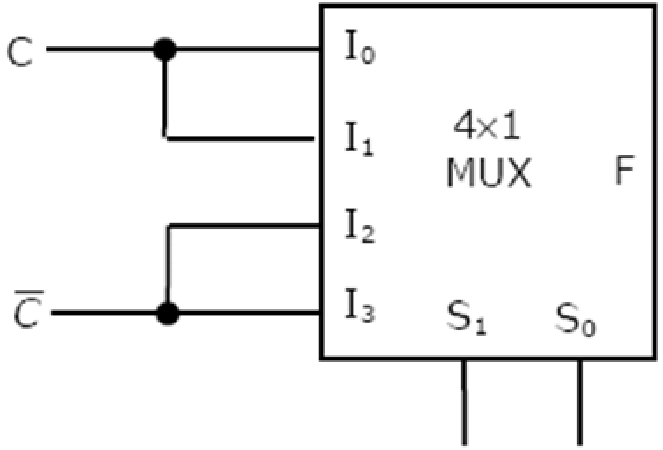


(a) $PQ + P\overline{Q}S + \overline{Q}\,\overline{R}\,\overline{S}$

(b) $P\overline{Q} + PQ\overline{R} + \overline{P}\,\overline{Q}S$

(c) $P\overline{Q}\,\overline{R} + \overline{P}QR + PQRS + \overline{Q}\,\overline{R}\,\overline{S}$

(d) $PQ\overline{R} + PQR\overline{S} - P\overline{Q}\,\overline{R}S + \overline{Q}\,\overline{R}\,\overline{S}$

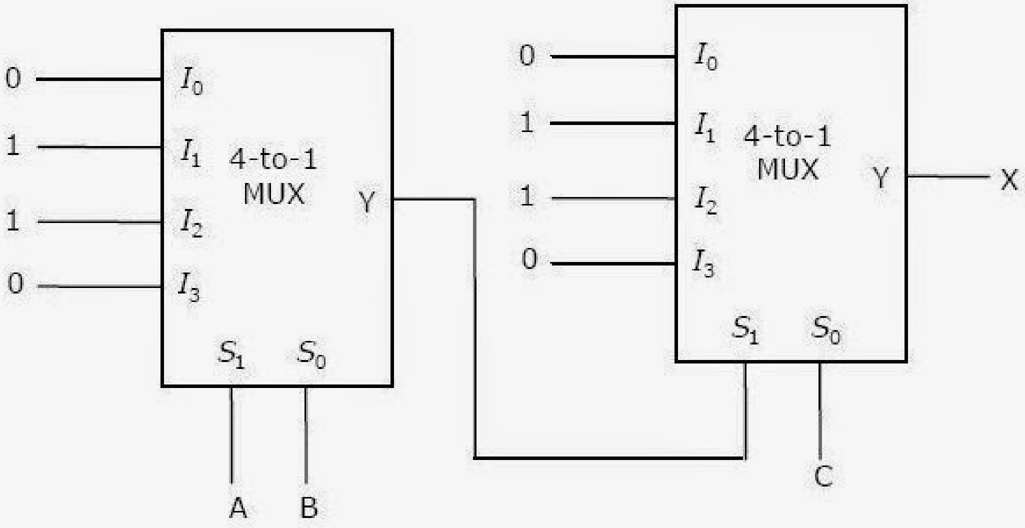


(a) $F = A.C$

(b) $F = A + C$

(c) $F = B.C$

(d) $F = B + C$



(A) $X = A\,\overline{B}\,\overline{C} + \overline{A}\,B\,\overline{C} + \overline{A}\,\overline{B}\,C + A\,B\,C$

(B) $X = \overline{A}\,B\,C + A\,\overline{B}\,C + A\,B\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C}$

(C) $X = AB + BC + AC$

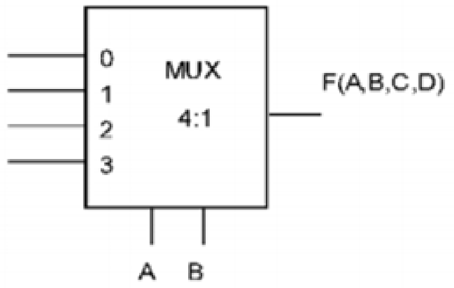
(D) $X = \overline{A}\,\overline{B} + \overline{B}\,\overline{C} + \overline{A}\,\overline{C}$

GO CLASSES

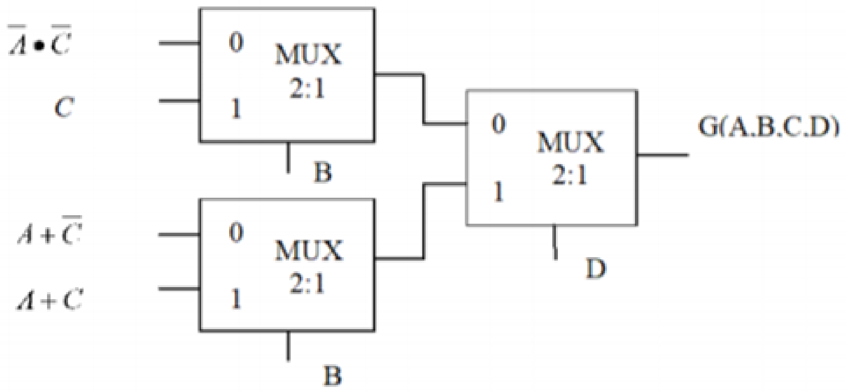
GO Classes

(a) A logic function is given as $F(A,B,C,D) = AD + BC' + \overline{A+B+C}$

Show how to implement this function by a 4:1 MUX with (A, B) as control inputs (clearly label every input)



(b) Determine the output function G(A,B,C,D) when ABCD = 0001 for the schematic given below:

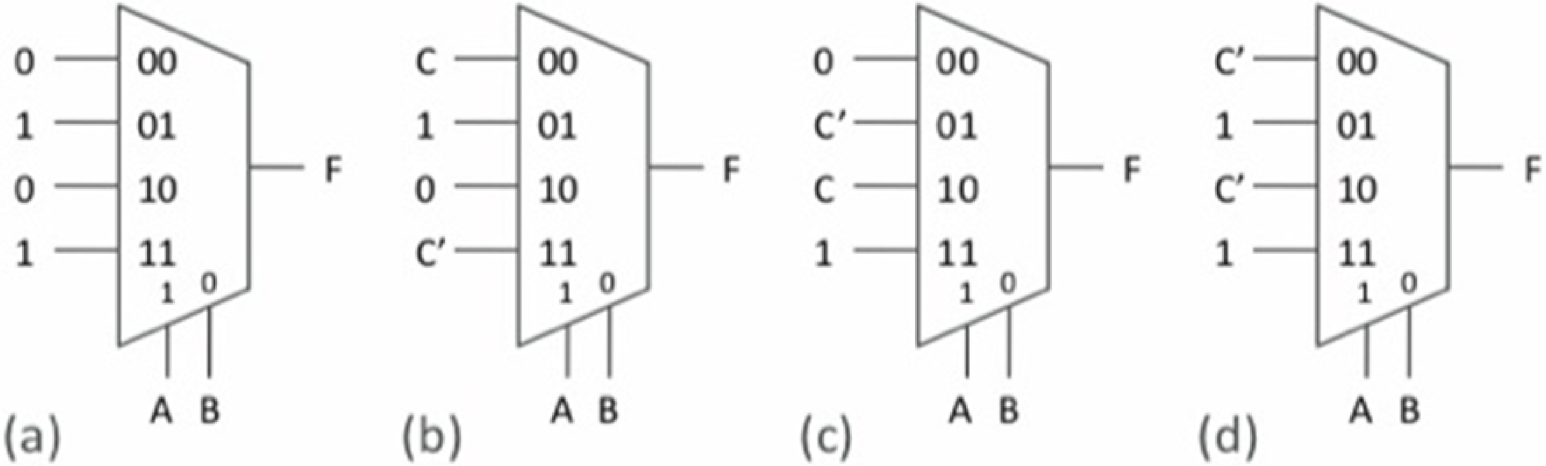


www.goclasses.in

Question #7

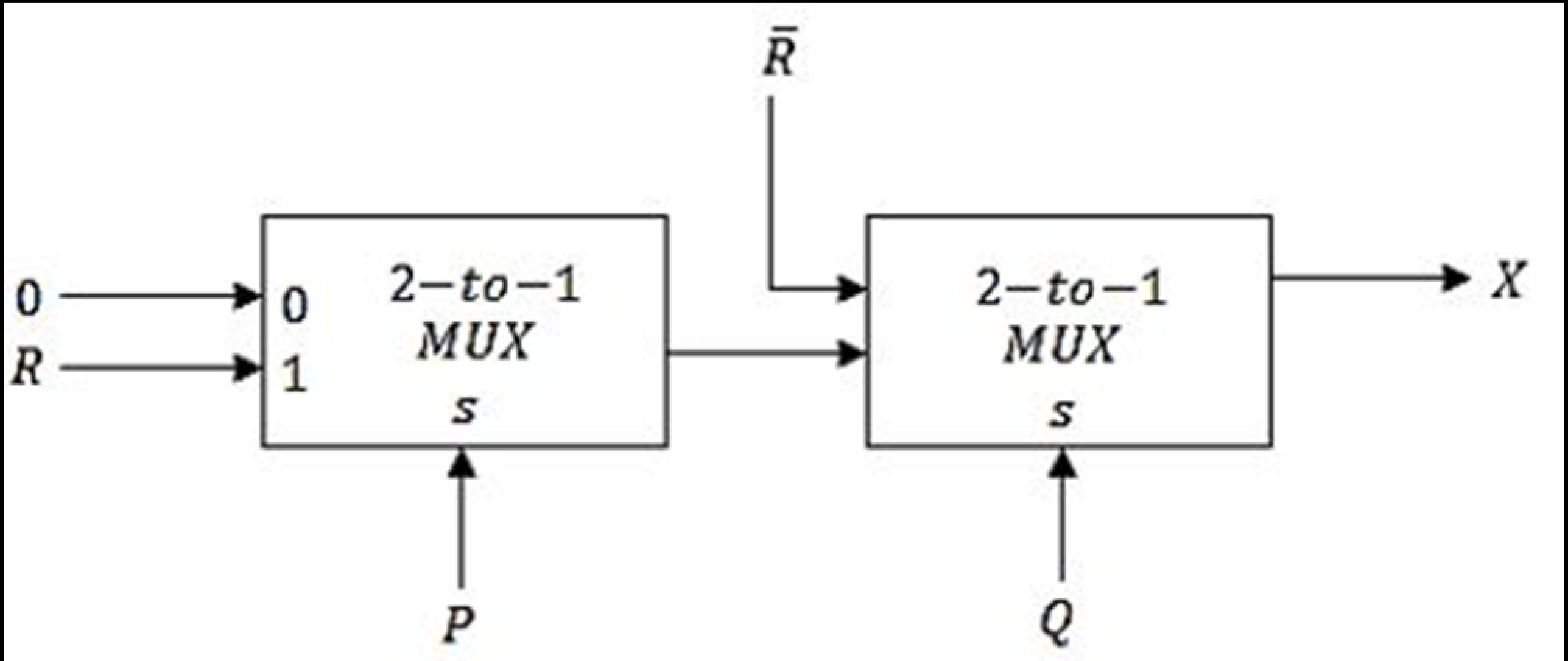
- Implement the given logic function using an **4-to-1** multiplexer.

$$F(A, B, C) = B.C' + A'.C$$



(e) None of the above.

Q :



Expression of X ?