DECODERS AND ENCODERS
(PART 01)

WHAT IS DECODER?

2^n

"Decoder is a combinational circuit that has 'n' input lines and maximum of 2^n output line. Where the outputs of decoder are nothing but the minterms of 'n' input variable lines, when it is Enabled"

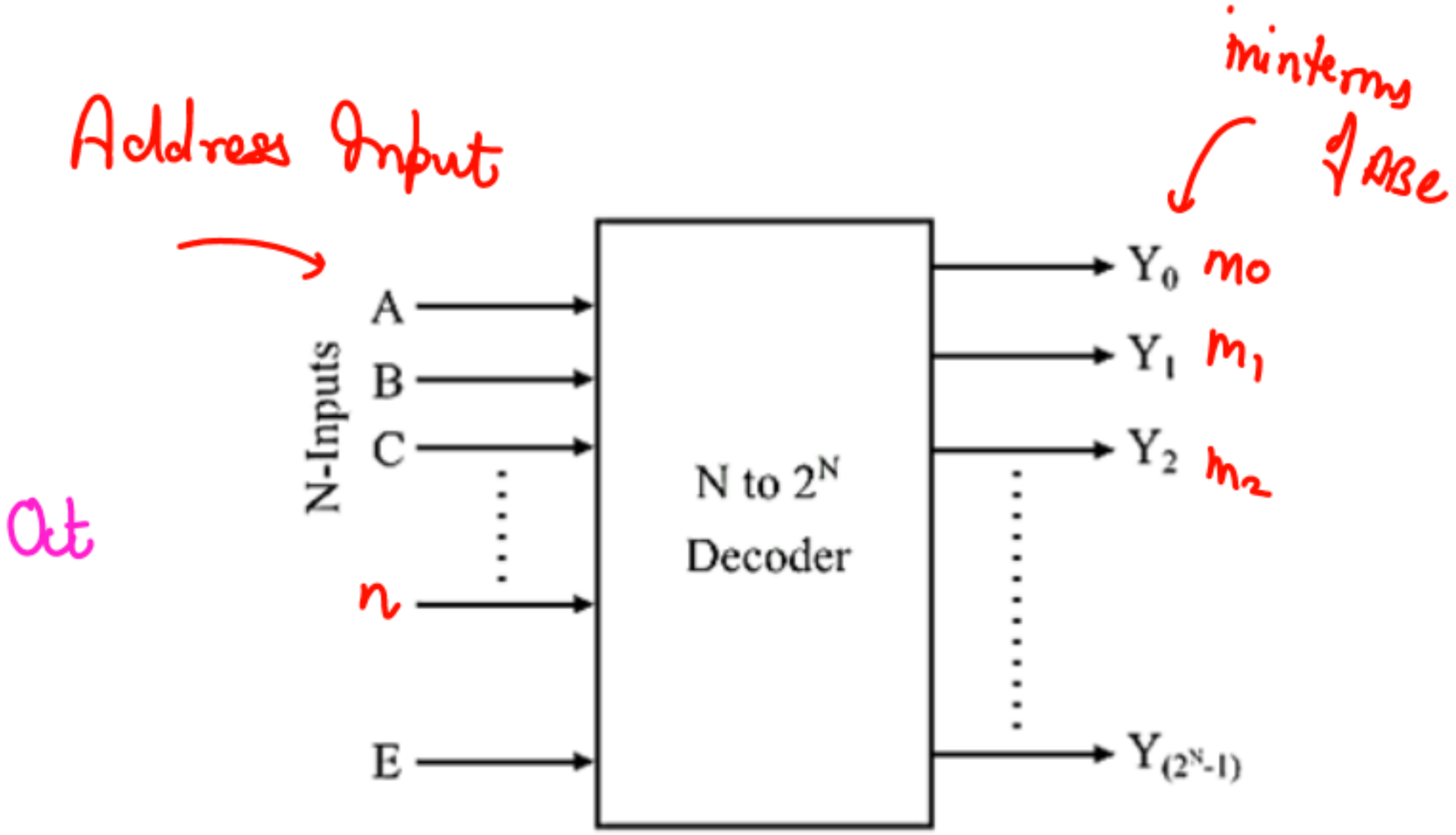
The input combinations of decoder are also termed as Address input, at it act an interface between CPU and Memory (RAM).

The combination of each input will select a cell of memory.

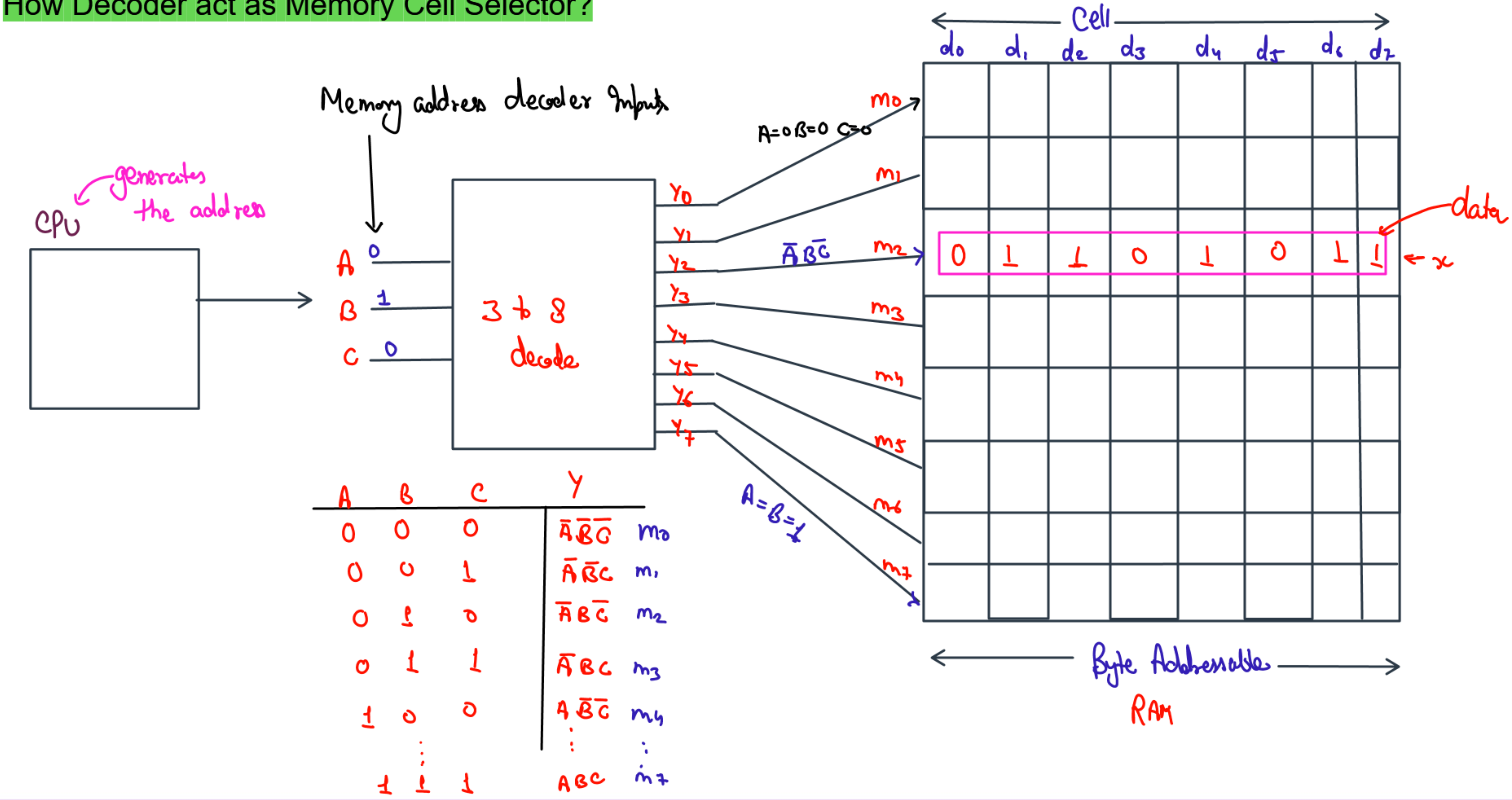
Mostly used in Code Converters

Eg If we have 3 inputs → we can represent 8 numbers (0 to 7) ↳ Octal } Bin → Oct

If we have 4 input → 16 numbers ↳ 0-15 OR 0 to F ↓ hexadecimal } Bin → Hex



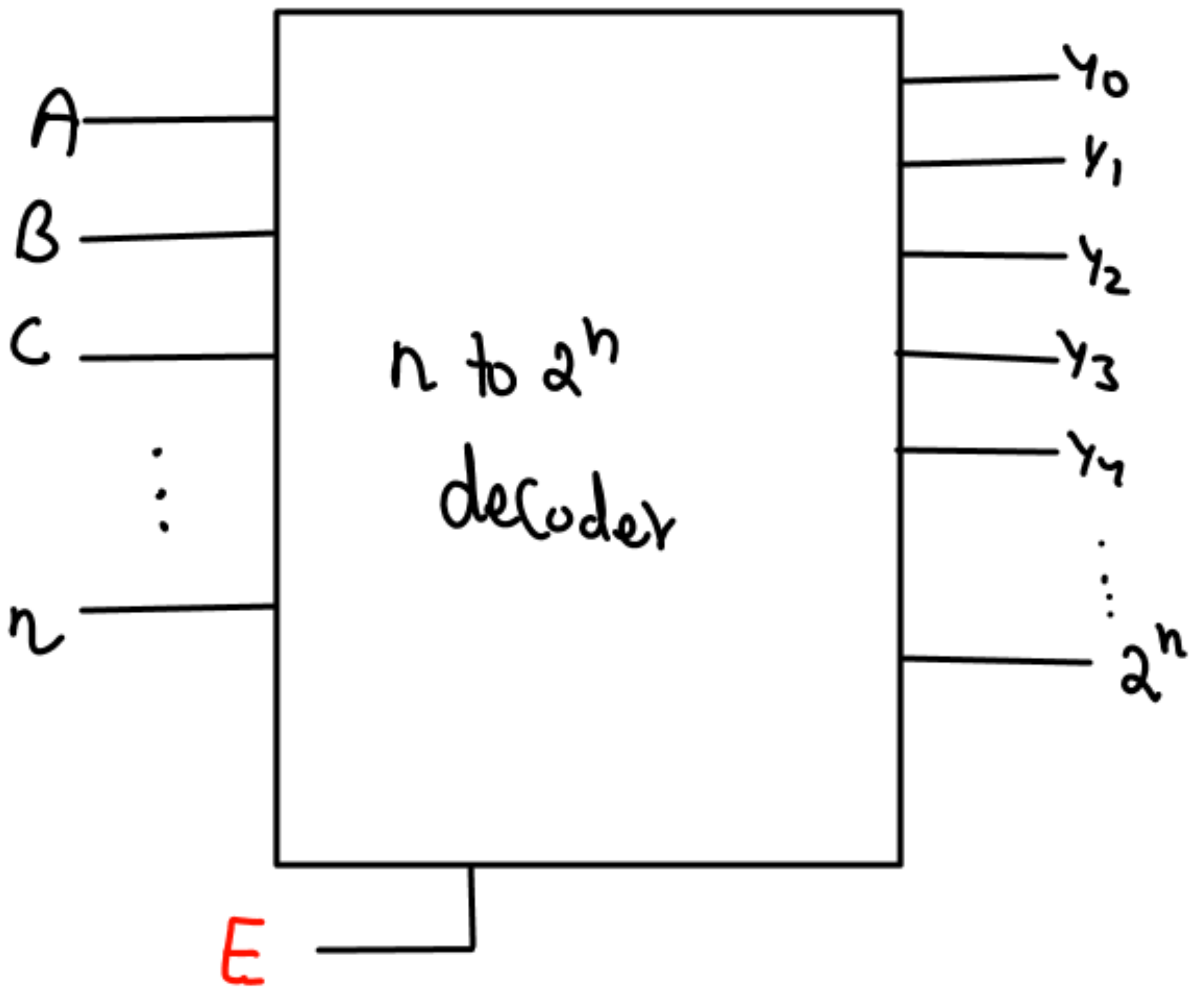
How Decoder act as Memory Cell Selector?



A	B	C	Y	
0	0	0	$\bar{A}\bar{B}\bar{C}$	m_0
0	0	1	$\bar{A}\bar{B}C$	m_1
0	1	0	$\bar{A}B\bar{C}$	m_2
0	1	1	$\bar{A}BC$	m_3
1	0	0	$A\bar{B}\bar{C}$	m_4
	⋮		⋮	⋮
1	1	1	ABC	m_7

Active high and Active low Decoder

Active high



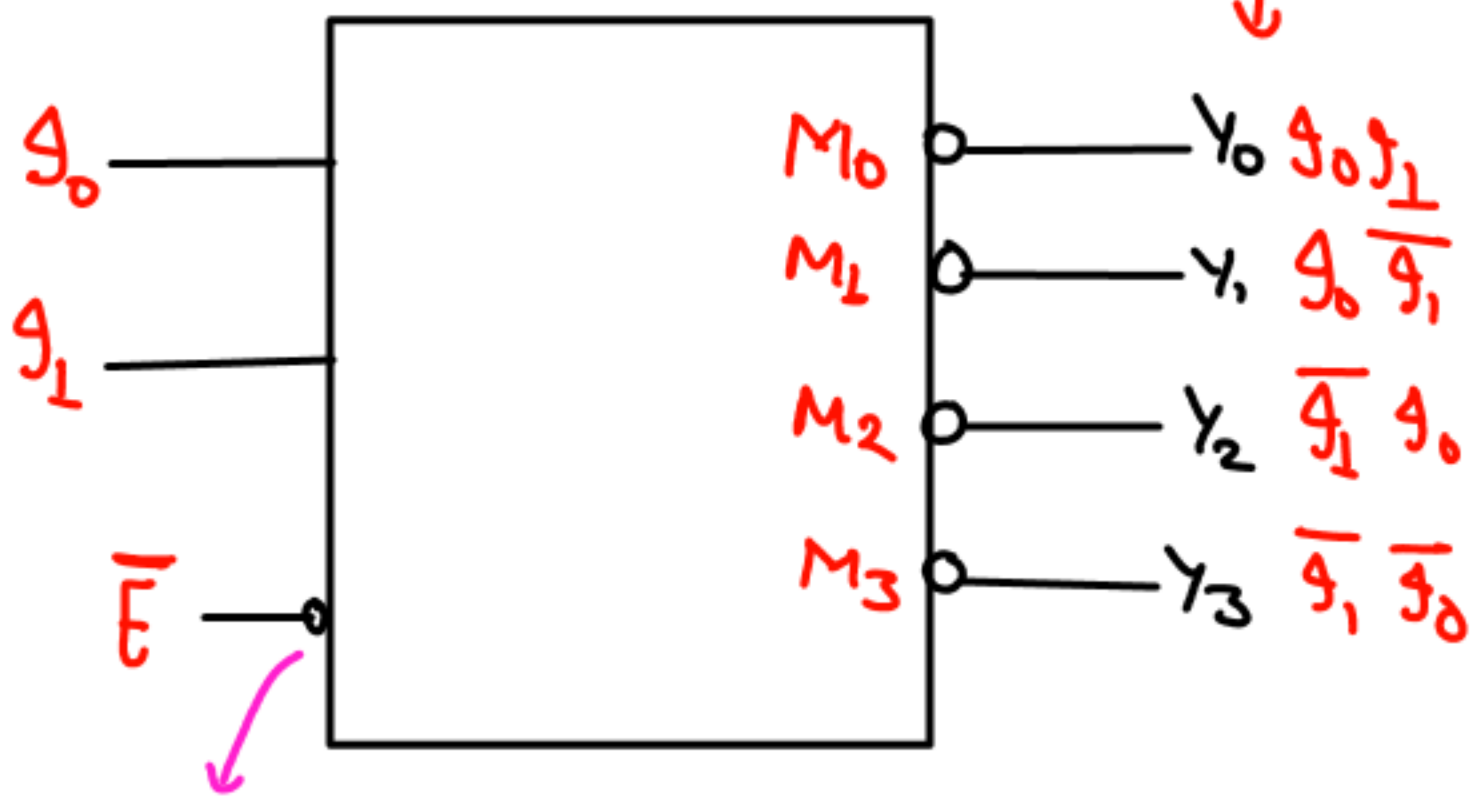
In Active high decoder if Enable (E) is high then decoder is Active otherwise Inactive

EN = 1 → ON

EN = 0 → OFF

Active low decoder

here outputs are just maxterms.



Initially all inputs are 1 Except the output Corresponding to the Input

Bubble implies Active low

E	g_0	g_1	Y_0	Y_1	Y_2	Y_3
1	X	X	X	X	X	X
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

here 1 means low

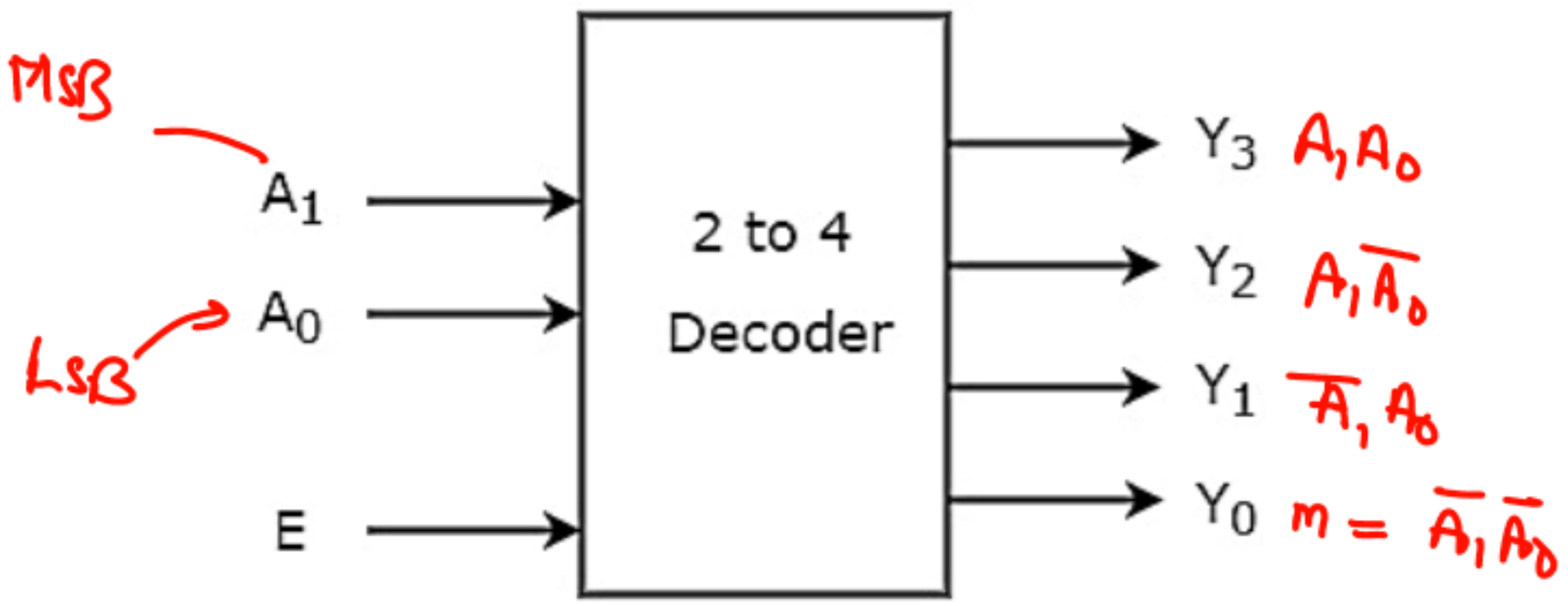
2 to 4 Decoder

2 to 4 Decoder has 2 input lines and 4 output lines.

Enable	Inputs		Outputs			
	A1	A0	Y3	Y2	Y1	Y0
0	x	x	x	x	x	x
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

m_0

Active high



$$Y_3 = E \cdot A_1 A_0$$

$$Y_2 = E \cdot A_1 \bar{A_0}$$

$$Y_1 = E \cdot \bar{A_1} A_0$$

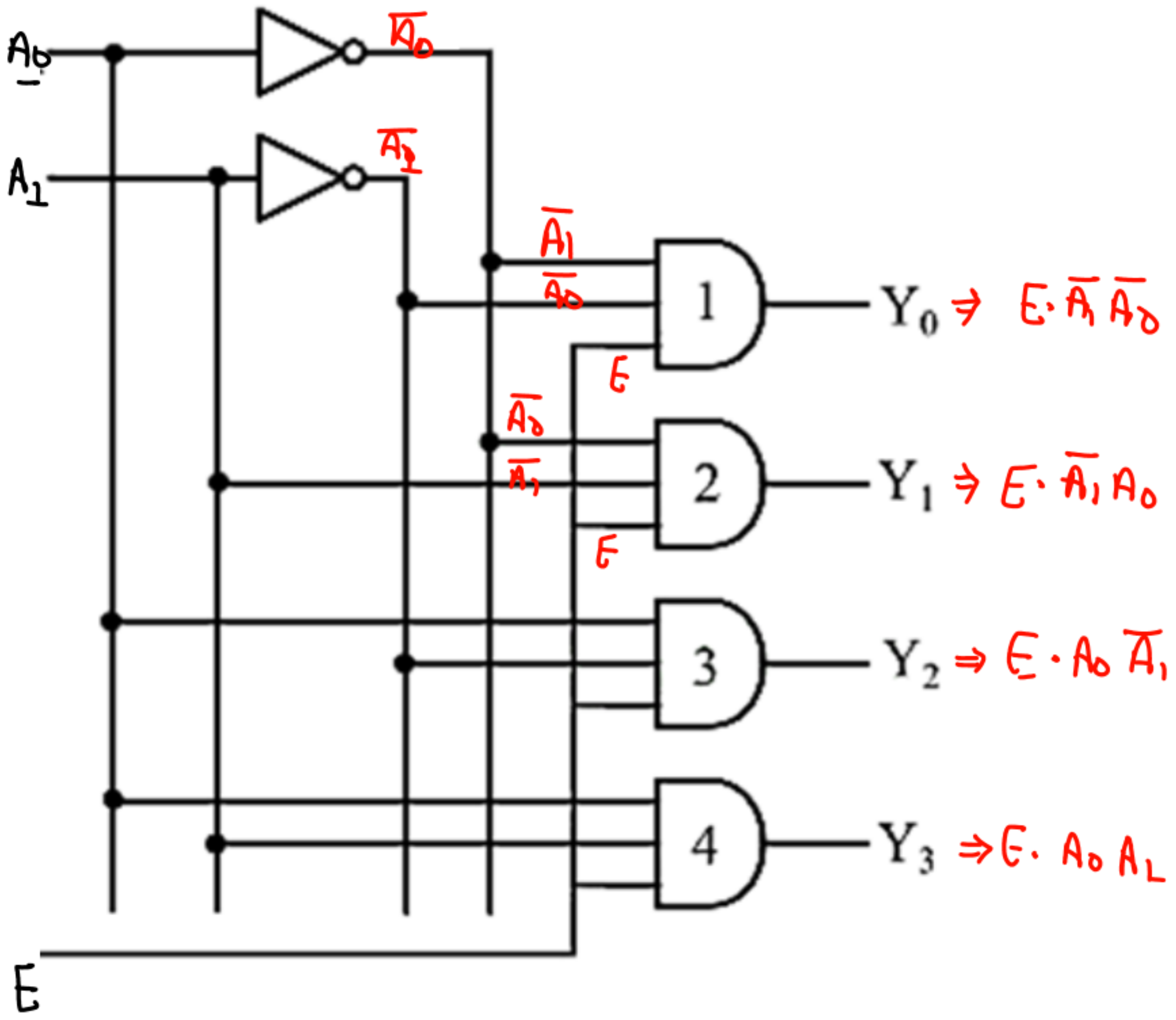
$$Y_0 = E \cdot \bar{A_1} \bar{A_0}$$

* Each output is one product term
↳ AND

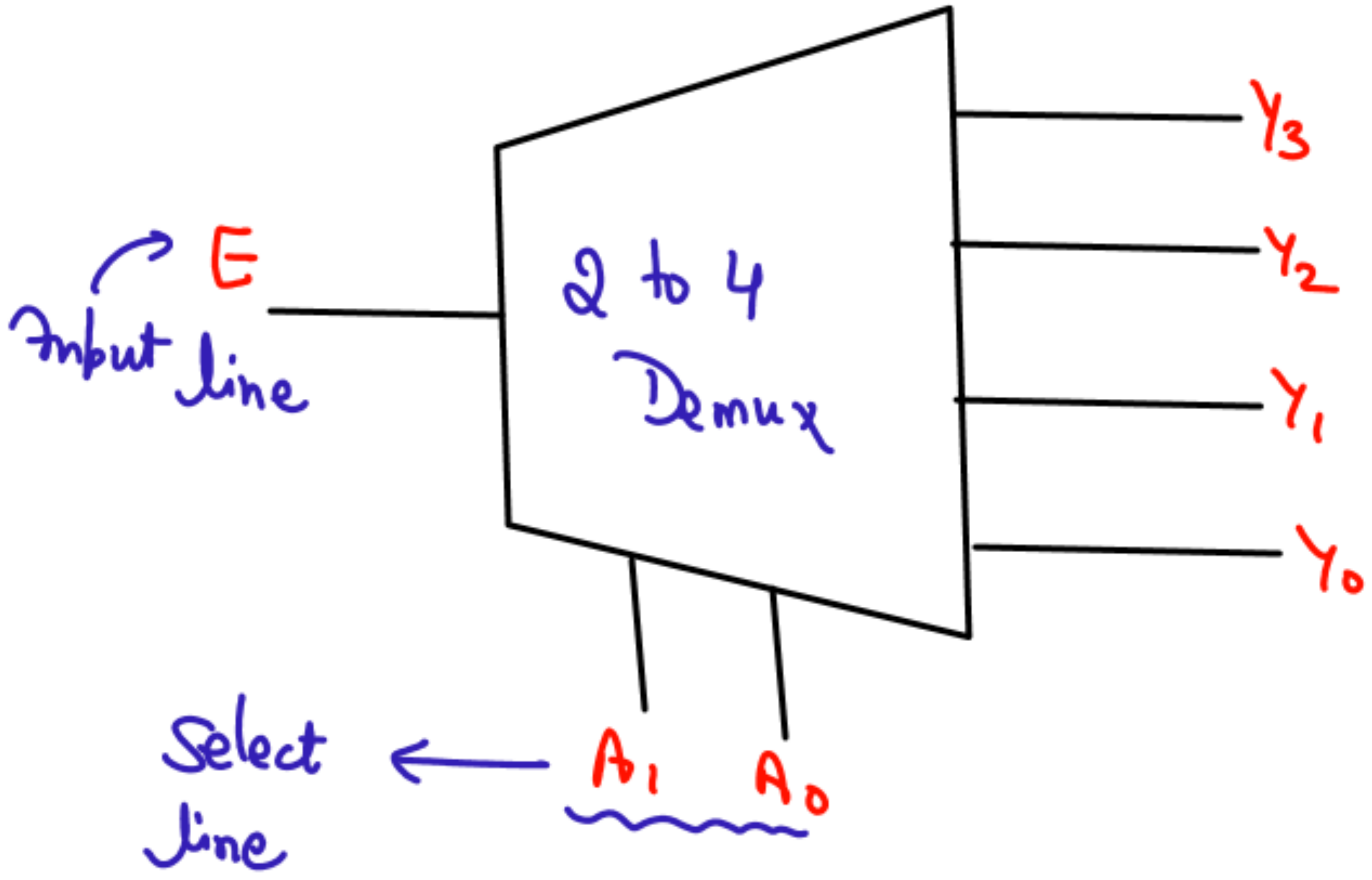
total 4 Product terms
↳ total 4 AND gates

* If there are 'n' output lines then total 'n' product terms will be there hence It required 'n' AND gates

2 to 4 Decoder



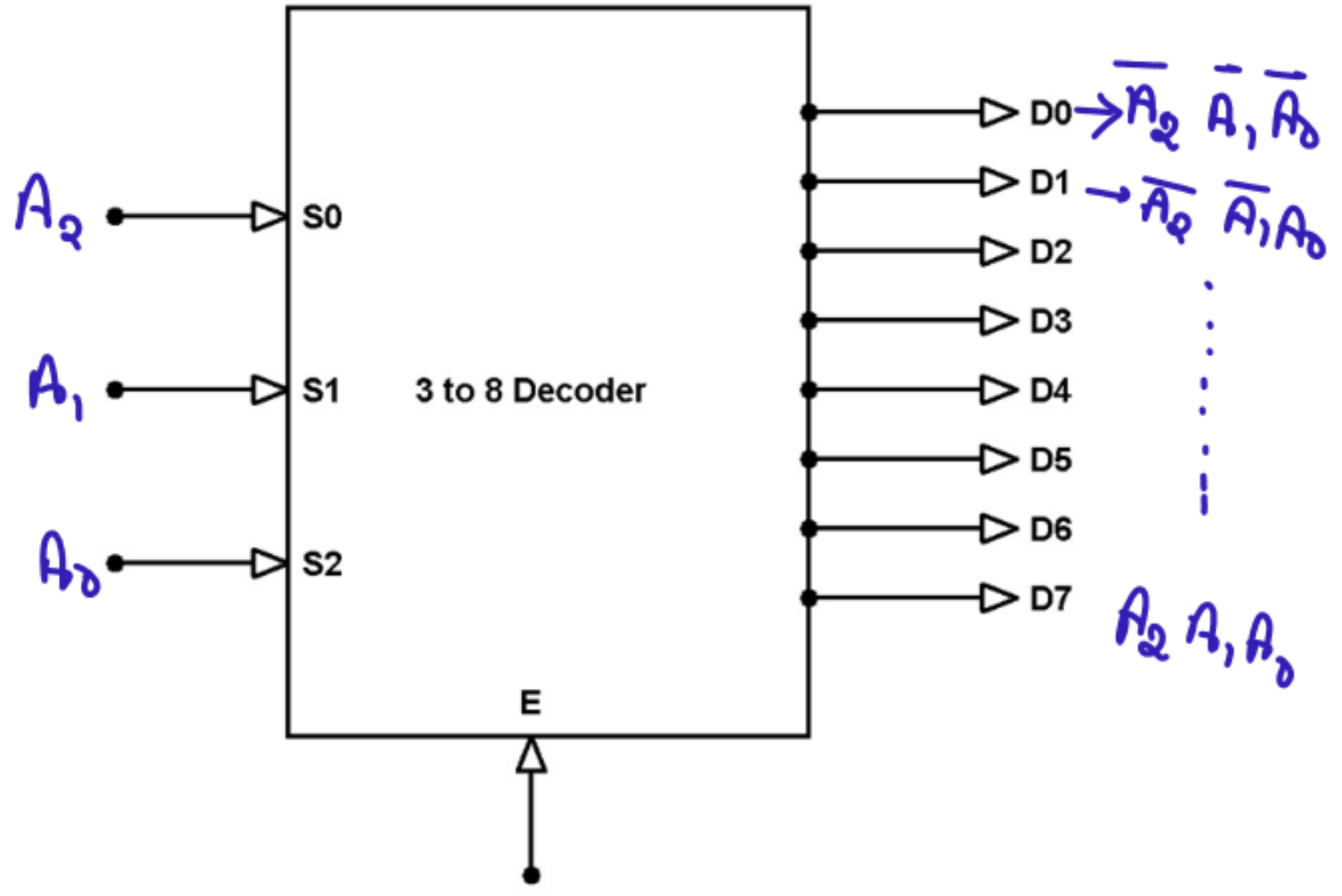
* A decoder can works as Demultiplexer where the input of Demux = E of decoder and the select lines of demux will work as the input lines of decoder



3 to 8 Decoder

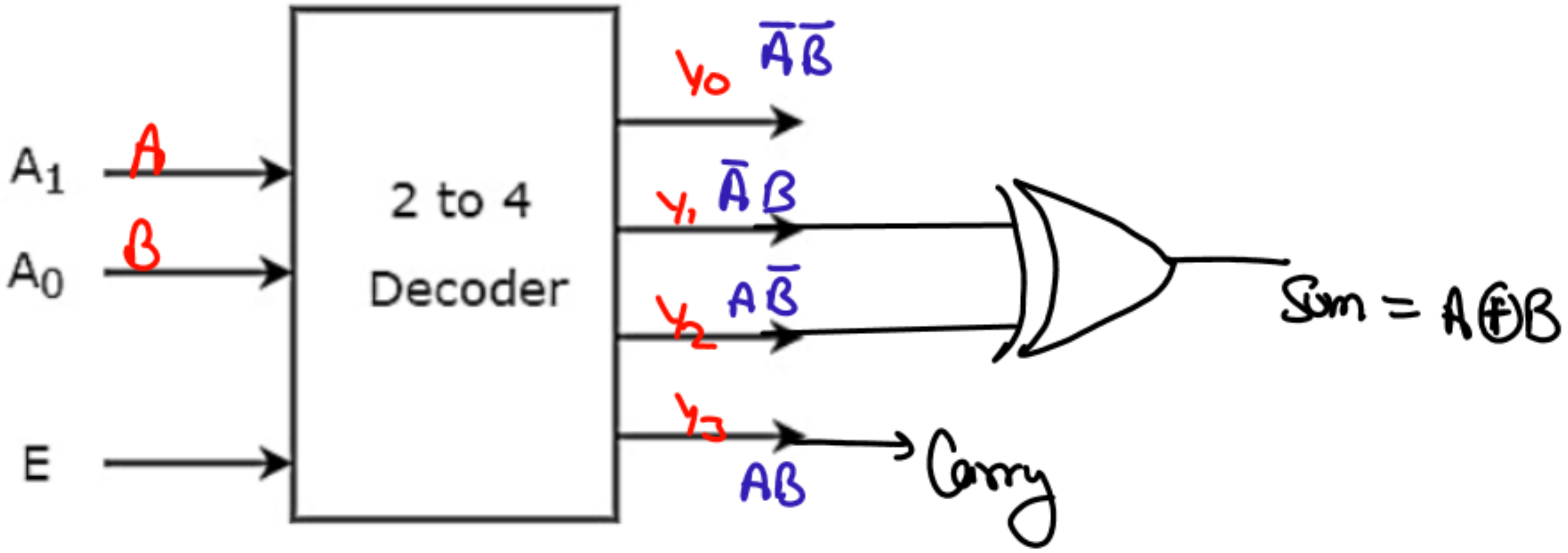
3 to 8 Decoder has 3 input lines and 8 output lines.

Enable	**Inputs**			**Outputs**							
	A2	**A1**	**A0**	**Y7**	**Y6**	**Y5**	**Y4**	**Y3**	**Y2**	**Y1**	**Y0**
0	x	x	x	x	x	x	x	x	x	x	x
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



Active high decoder

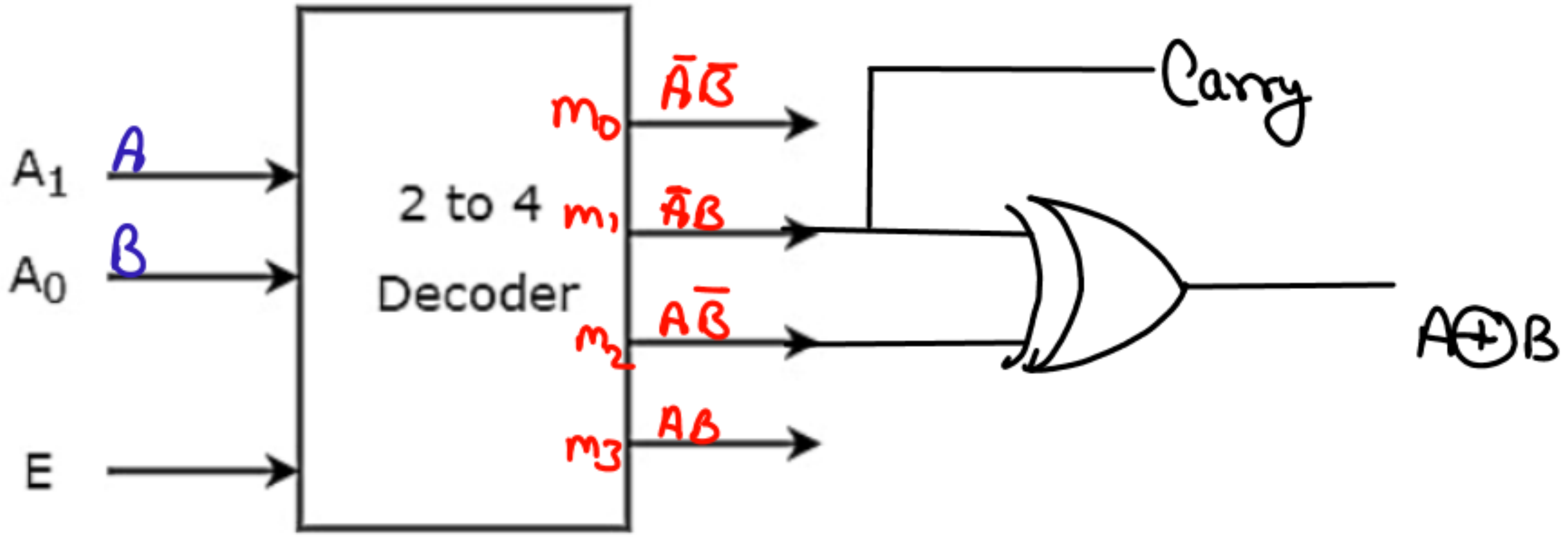
Half Adder using Decoder



$$Sum = A \oplus B = A\bar{B} + \bar{A}B$$

$$Carry = AB$$

Half Subtractor using Decoder



difference = $A \oplus B$ → $A\bar{B} + \bar{A}B \Rightarrow \Sigma m(1, 2)$

Borrow = $\bar{A}B$ → $\Sigma m(1)$

full adder or Subtractor → minterm form

↓

SOP

Implementing the function using Decoder

To implement a function using decoder following steps we have to follow:

1. Connect function variables to the select lines (input variables)→ Address Input
2. Convert the output into Canonical SOP form
3. Connect all the output to the corresponding minterms using OR gates.

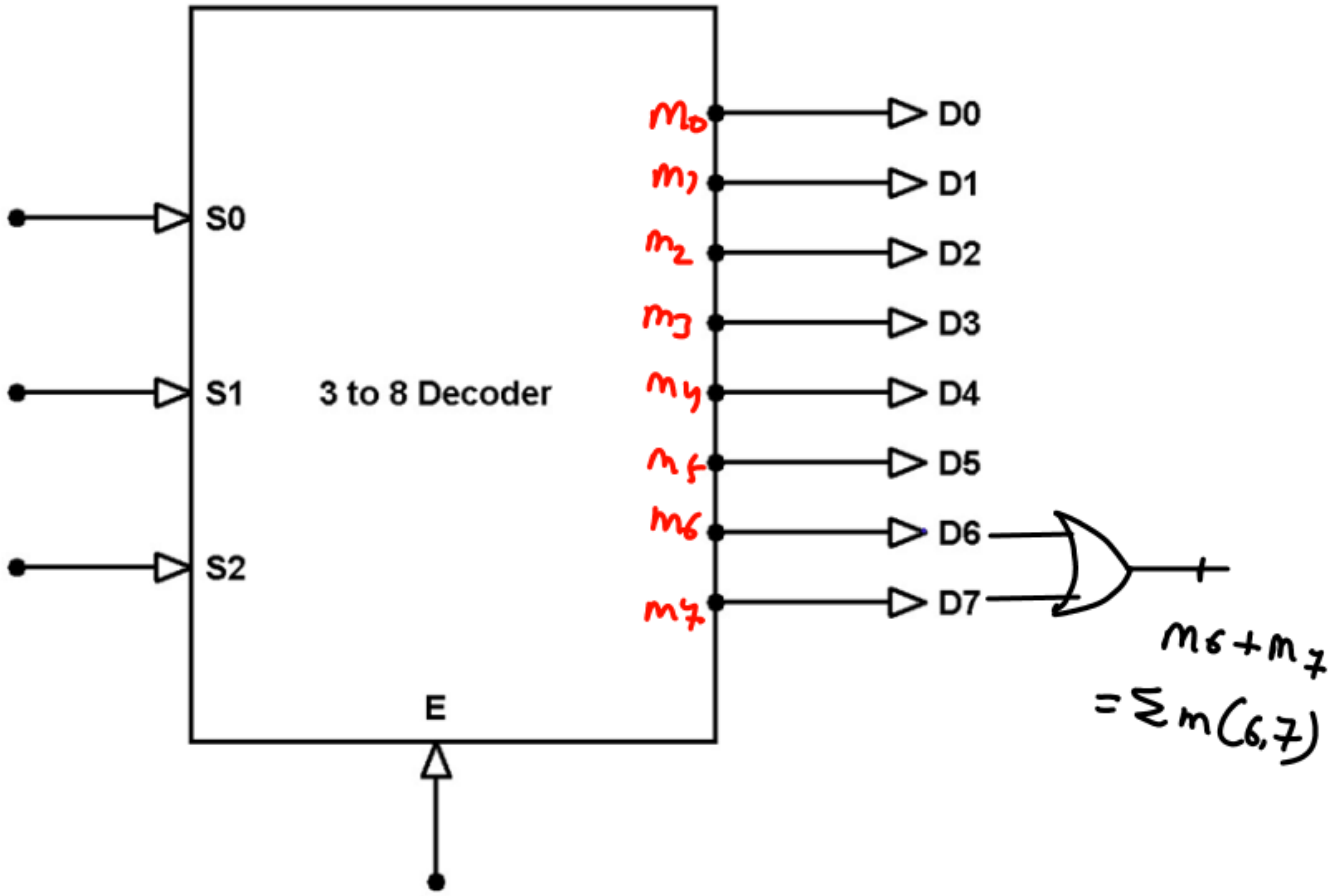
Example: f(A,B,C) = AB (Standard SOP form)

= ABC + ABC' (Canonical SOP form)

$$f(A,B,C) = AB(C+\bar{C})$$

$$= \underset{\underset{111}{\downarrow}}{ABC} + \underset{\underset{110}{\downarrow}}{AB\bar{C}}$$

$$f(ABC) = \Sigma m(6,7)$$



$m_6 + m_7$

$= \Sigma m(6,7)$

Example: f(A,B,C) = A+BC → Simplified

A B C

x x

↓

$A\bar{B}\bar{C}$

$A\bar{B}C$

$AB\bar{C}$

ABC

BC A

x

↓

$BC(A+\bar{A})$

$= ABC + \bar{A}BC$

BC	
00 →	$\bar{B}\bar{C}$
01	$\bar{B}C$
10	$B\bar{C}$
11	BC

⇒ Canonical form

$$\Rightarrow A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C} + ABC + \cancel{ABC} + \bar{A}BC$$

$A\bar{B}\bar{C}$	$A\bar{B}C$	$AB\bar{C}$	ABC	$\bar{A}BC$
100	101	110	111	011
4	5	6	7	3

$$\Rightarrow \sum m(3,4,5,6,7)$$