ACOL215

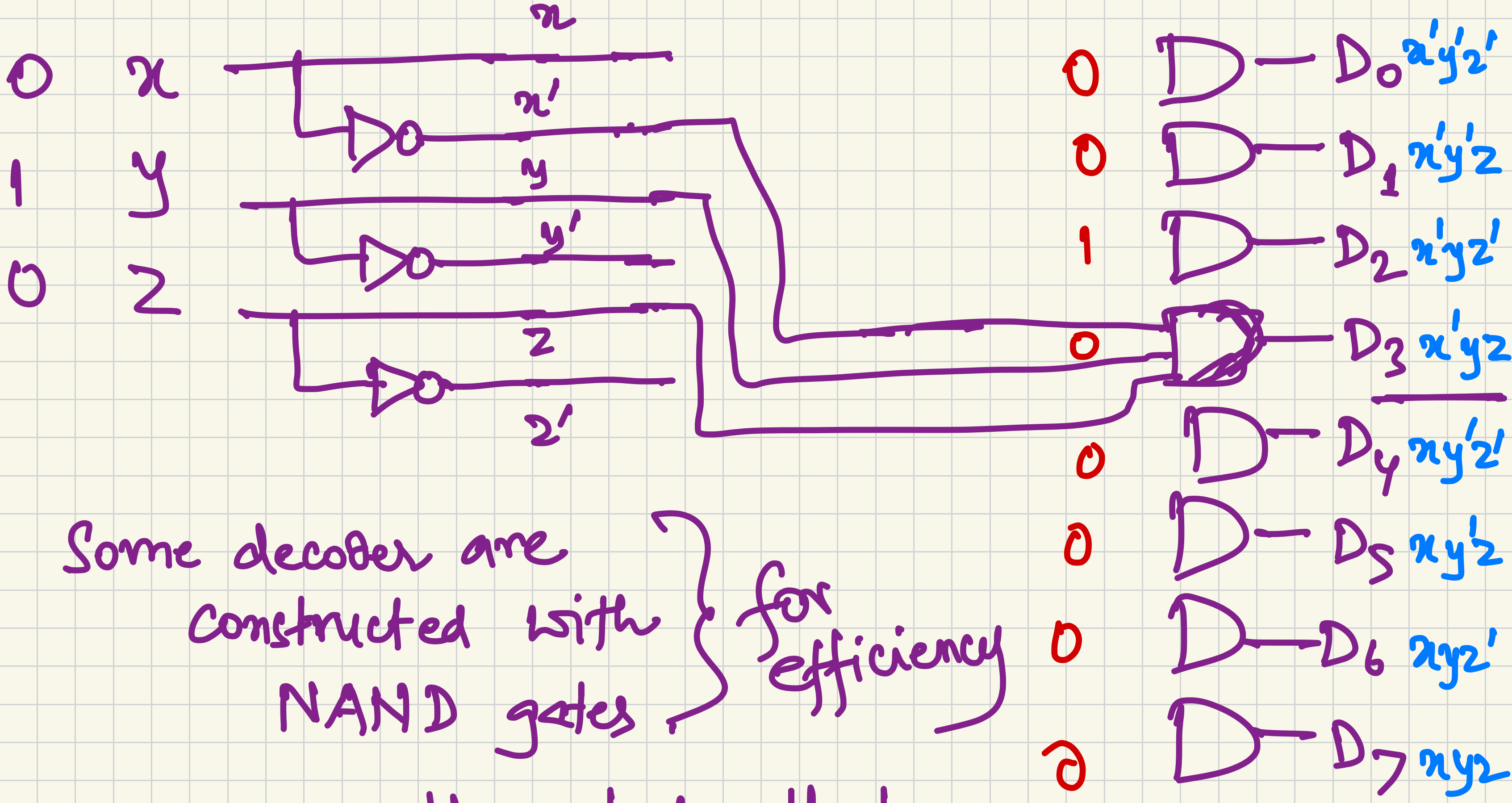
(Oct. 8th)

Decoders

A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n output lines.

In particular, we look at n-to-m-line decoders, where $m \leq 2^n$.

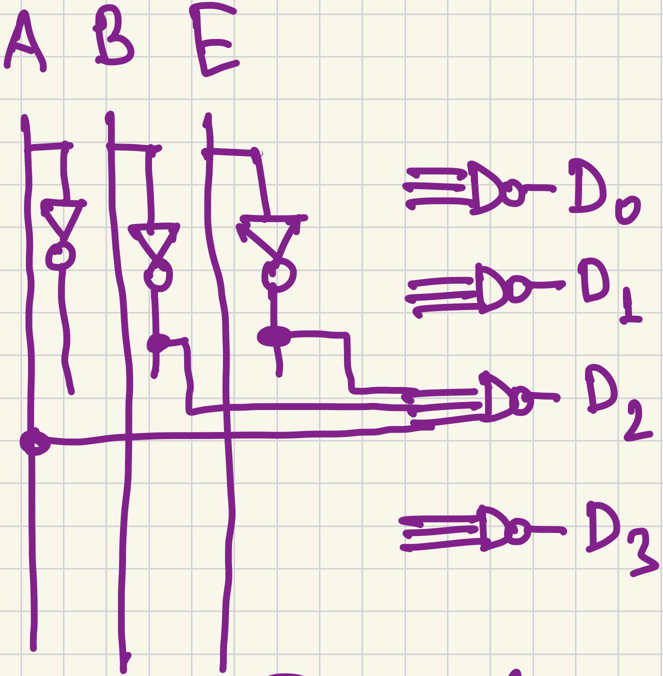
A binary code of n bits can represent 2^n distinct elements of coded information.



Some decoder are constructed with NAND gates } for efficiency

→ complemented output

Decoders can have one or more enable inputs to control the circuit operation.



E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
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

Two-to-four line decoder with an enable input constructed with NAND gates

$\underbrace{\text{complemented input}}_{\text{NAND gates}}$ & $\underbrace{\text{complement enable input}}_{\text{outputs are enabled when } E = 0 \text{ (active-low enabled)}}$

A decoder with enable input can function as a <u>demultiplexer</u>

→ a circuit that receives information from a single line and directs it to one of 2^n possible output lines.

E can be thought of as data input line and A, B as selection input lines.

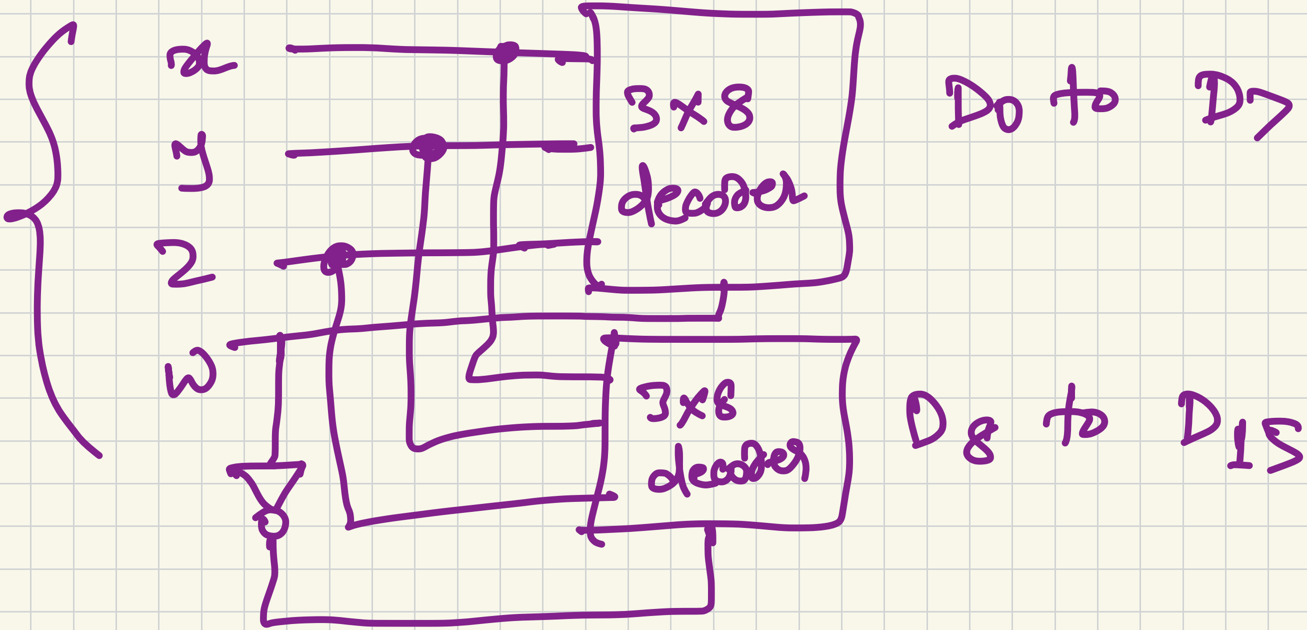
→ For different selection inputs, E is directed to different outputs.

when AB is 10, E is directed to D_2.

AB is 11, E is directed to D_3.

A decoder with an enable input is also called a decoder demultiplexer.

Decoders with enable inputs can be connected to form a larger decoder circuit.

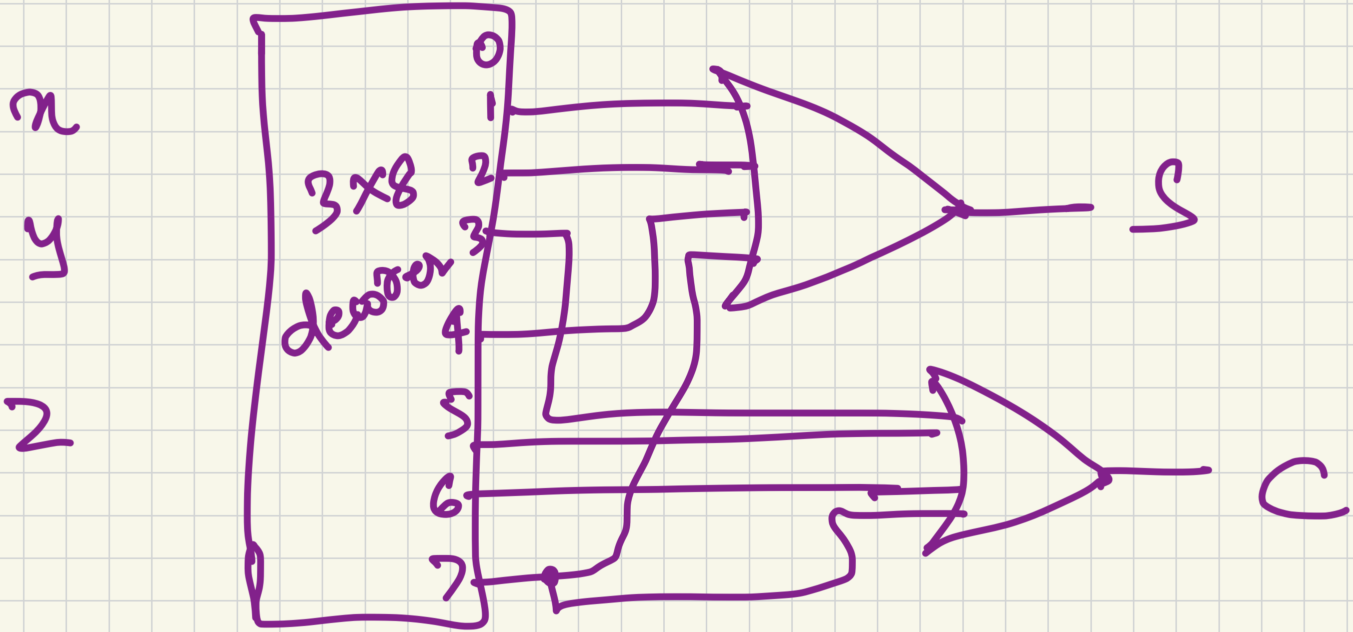


Implement a full-adder with a decoder.

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S(x,y,z) = \underline{\underline{\sum(1,2,4,7)}}$$

$$C(x,y,z) = \underline{\underline{\sum(3,5,6,7)}}$$


x
y
z
3x8
decoder
0
1
2
3
4
5
6
7
S
C

Encoder

inverse operation of a decoder

2^n (or fewer) input lines

→ n output lines.

Example: octal to binary encoder

D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

$x = D_4 + D_5 + D_6 + D_7$

$y = D_2 + D_3 + D_6 + D_7$

$z = D_1 + D_3 + D_5 + D_7$

If D_5 and D_6 are 1 at the same time the output is 111 (which is absurd).

↳ Fixed by assigning higher priority to higher index input.

If all inputs are 0, then output is 000 which is the code when $D_0 = 1$.

↳ Fixed with a valid bit indicator.

D_0	D_1	D_2	D_3
0	0	0	0
1	0	0	0
X	1	0	0
X	X	1	0
X	X	X	1
0	0	0	1
0	0	1	1
0	1	0	1
X	X	X	1
1	0	0	1

x	y	V
X	X	0
0	0	1
0	1	1
1	0	1
1	1	1

	00	01	11	10
00	x	1	1	0
10	0	1	1	0
01	1	1	1	0
11	1	1	1	0

D_2

D_0D_1 \ D_2D_3	00	01	11	10
00	X	1	1	1
01		1	1	1
11		1	1	1
10		1	1	1

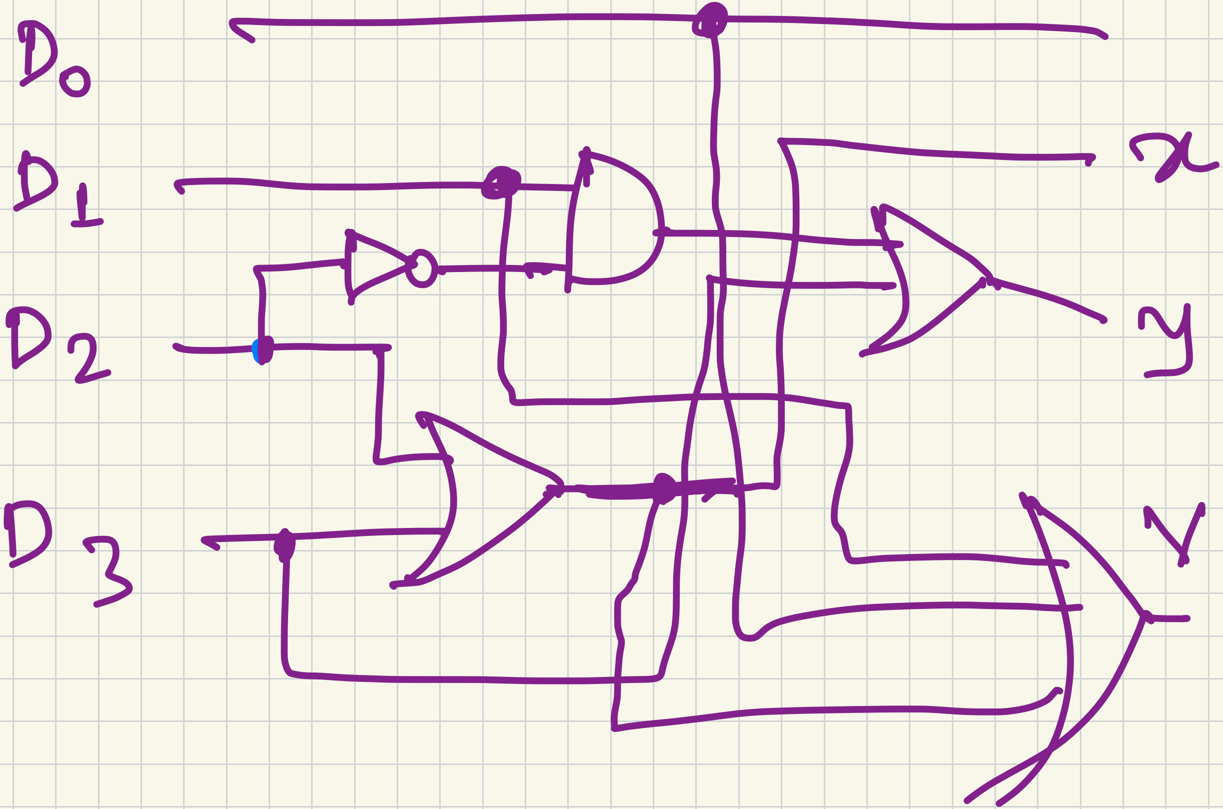
D_3

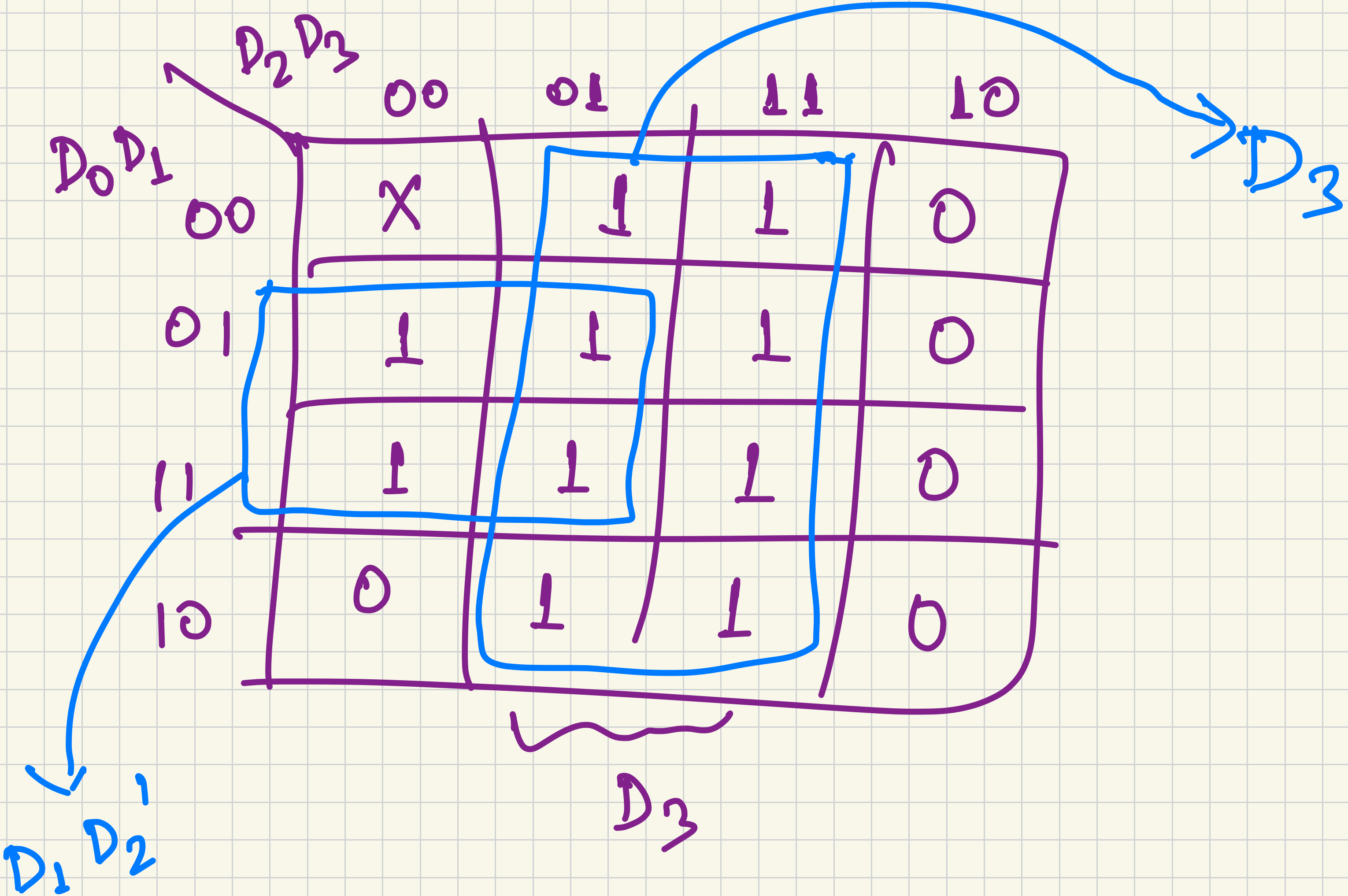
$x = D_2 + D_3$

y ?

$D_1 \cdot D_2' + D_3$

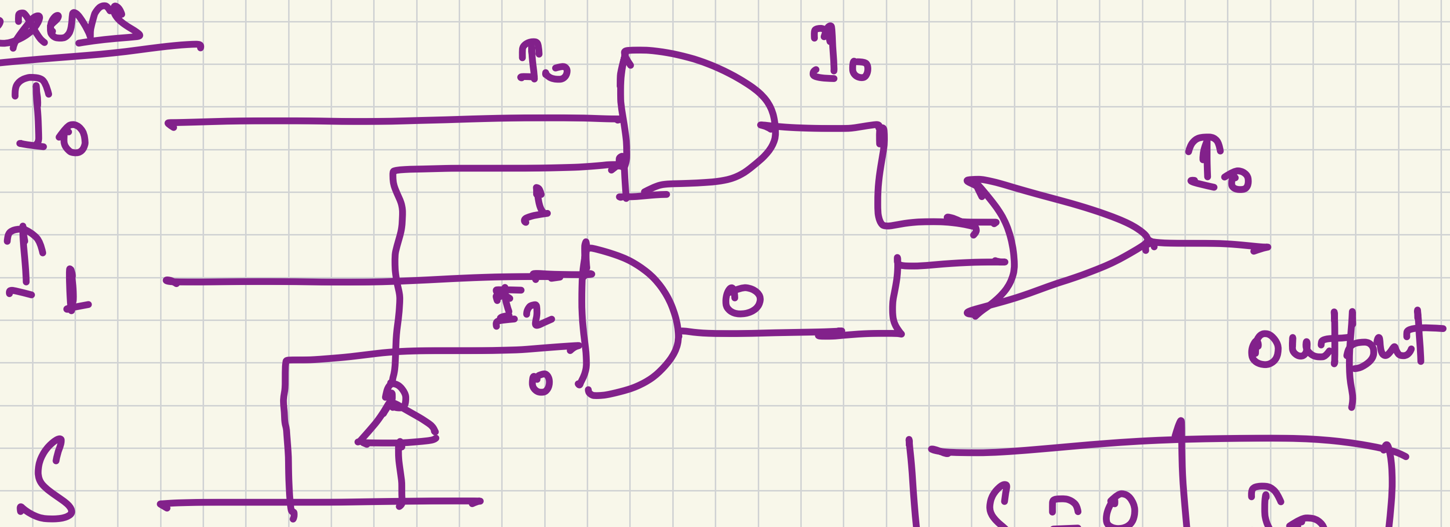
$V = D_0 + D_1 + D_2 + D_3$


D0
D1
D2
D3
x
y
Y



D_0D_1 D_2D_3	00	01	11	10
00	X	1	1	0
01	1	1	1	0
11	1	1	1	0
10	0	1	1	0

Multiplexers



S = 0	I_0
S = 1	I_1