Any number system can be represented by this series:

The most significant digit of decimal num

$$A_{n-1}\, A_{n-2} \cdots A_1\, A_0 \,.\, A_{-1}\, A_{-2} \cdots A_{-m+1}\, A_{-m}$$

This is the least significant Digit for integers

* Conversion Between Bases

① Conversion from any base r to Decimal.

* starting with the Decimal system we find we can represent numbers in this system by this way:

as an example:

$$(783)_{10} = 3\times10^0 + 8\times10^1 + 7\times10^2$$

Converting By Power Series

use this series for all systems to convert them into decimal system using the base of each one

SUBJECT:

ex1 $(732)_8$

$$2\times 8^0 + 3\times 8^1 + 7\times 8^2 = (474)_{10}$$

LSD

$(1100)_2$

$$0\times 2^0 + 0\times 2^1 + 1\times 2^2 + 1\times 2^3 = (12)_{10}$$

LSD

$(A12)_H$

$$2\times 16^0 + 1\times 16^1 + 10\times 16^2 = (\dots)_{10}$$

A

LSD

if you noticed, we've started converting from the *least significant digit* → LSD

The first number from right.

And now converting the numbers after the decimal comma, we will start with the most significant digit (MSD), which is the first number after the decimal comma.

SUBJECT: .... / /

ex: $(0,735)_{10}$

The power series of it is:

$7\times10^{-1}+3\times10^{-2}+5\times10^{-3}=0,735$

MSD

now use it to convert the others

$(11,101)_2$

converting integers:

$1\times2^0+1\times2^1=(3)_{10}$

LSD

converting Decimal numbers:

$1\times2^{-1}+0\times2^{-2}+1\times2^{-3}=(0.625)_{10}$

MSD

The total converted number: $(3.625)_{10}$

SUBJECT: / /

② conversion from decimal to any base r

Use the method Devide and Multiply

→ for the Integral part you devide by the new radix

↳ The digits for new radix are the remainders in reverse order of their computation

→ for the Fractional part you multiply by the new radix

↳ ~~The~~ * you save the Integral part of the result of multiplication

* Then the digits for the new radix are the integer digits in order of their computation.

Scanned with CamScanner

(3) Conversion from Base r_1 to Base r_2

* Use base 10 (decimal) as proxy.

* Special case:

Hexadecimal → Binary (vice versa)

Octal → Binary (vice versa)

Hexadecimal	Octal
Restate numbers as four binary digits	Restate numbers as three binary digits.

* And you can use Binary as proxy between hexadecimal and octal.

Exercises / slide 22:-

* $(3A.5)_{16} = (\ ?\ )_8$ * using binary as proxy

$(3A.5)_{16} = (111010.0101)_2 = (72.24)_8$

* $(635.177)_8 = (\ ?\ )_{16} = (19D.3F8)_{16}$

$(635.177)_8 = (110011101.001111111)_2$

Scanned with CamScanner

SUBJECT:

Exercises / slide 26 :-

① $(E3D2.3)_{16} = (171722.14)$ * using binary as proxy

② $(341)_8 = (E1)_{16}$ * using binary as proxy

③ $(153)_{10} = (10011001)_2$ * division

④ $(1001.101)_2 = (9.625)_{10}$ * power series

⑤ $(245.5)_{10} = (D5.8)_{16}$

⑥ $(99)_{10} = (1100011)_2$

* properties of ~~[illegible]~~ numeric systems

$r \rightarrow$ base
$n \rightarrow$ number of allowed digits
$m \rightarrow$ number of allowed fractional ~~digits~~ digits.

$\rightarrow$ for integral part:
number of combinations $= r^n$
maximum number $= r^n - 1$
minimum number $= 0$

$\rightarrow$ for fractional part:
maximum number $= (r^m - 1)/r$
minimum " $= 0$
number of fractional side combinations $= r^m$

Scanned with CamScanner

Exercise slide 30

* binary 3 bits before radix point
4 bits after " "

→ number of combinations before radix point $= 2^3 = 8$

→ " " " after " " $= 2^4 = 16$

maximum number $= (2^3 - 1) + (2^4 - 1)/2^4$

$= 7 + \frac{7}{8}$

$= (7.9375)_{10}$

Scanned with CamScanner

SUBJECT: .... / /

* Binary Codes

→ when you're dealing with non-binary values, you should encode them to ~~[illegible]~~ be representable

عكسيًّا

لو بدي أعرف كم bit بحتاج لأمثل عدد عناصر معين

ceiling function ← $\lceil \log_r M = n \rceil$ ← بستخدم اللوغاريتم

* Exercise slide 33

$\log_2 1040 = 10, \ldots$ → by ceiling it, the result is 11

→ you can code by ASCII code table

→ you can use the (BCD)

from * Decimal → binary

each digit in here → will be converted into its 4 bits in binary

and this is not a conversion ~~to~~ between systems

* 4 bits * are used in coding ←

↳ because you have 10 elements in decimal system to encode each one of them, so by the ceiling function $\lceil \log_2 10 = 4 \rceil$, you will need 4 bits to be able to represent them all

Exercise slide 34

$(856)_{10} \rightarrow (1000\,0101\,0110)_{BCD}$

Scanned with CamScanner

زي ما بقدر أعمل بال encoding بال binary ، بقدر اعمل بال octal و ال hexadecimal

you have 1040 comination to encode ~~[illegible]~~, how many digit you need to encode it in binary, octal, hexa decimal?

<u>binary</u> $\rightarrow \log_2 1040 = 10,\dots \rightarrow$ by ceiling bits $= 11$

when bits in binary $= 11$

octal

بعتبره صفر ونضيفه من عندي

_ _ _ | _ _ _ | _ _ _ | _ _ _

4 3 2 1

you need 4 digits in octal to encode 1040 element

بعتبره صفر ونضيفه من عندي

_ _ _ _ | _ _ _ _ | _ _ _ _

2 1

you need 3 digits in hexadecimal

باخذ مقابل كل 4 او 3 bit يلي بال binary ، حسب قانون التحويل الخاص بين binary و octal و hexadecimal

Scanned with CamScanner

N is a 3 digits in octal, what - - - - - - - in (BCD)

$8^3 = 512$ combination. → for max $(777)_8$

$\log_{10} 512$ = in ceiling (3) → every digit has 4 correspondi binary bits

so you need 12 bits in (BCD)

10^3

$\underline{1000}$ →

element

$\log_2 1000 \rightarrow 9, \ldots$

10 bit, in binary

let N be a 3-digit number in decimal, what is the minimum number of bits needed to represent N in hexadecimal

max $N = (999)_{10}$ → how many digits in 16?

$10^3 = 1000$ combinations

→ $\log_2(1000) = 9, \ldots$ → bits in binary = 10

so digits in hexadecimal are 3 digits.

Scanned with CamScanner

SUBJECT: ......

Bit parity code

* even parity

اللازم يكون عدد زوجي من ال (0) و ال (1) في الكود بضيف رقم ع هالأساس

* odd parity

اللازم يكون عدد فردي من ال (0) و ال (1) في الكود وبضيف 1 ع هالأساس

* Gray Code
  * 1 bit difference order

~~16x2~~ 32

16x2 = 32
16x3 = 48
x4 = 64
x5 = 80
x6 = 96
x7 = 112
x8 = 128
x9 = 144
x10 = 160
x11 = 176
x12 = 192

poly deg#2

$$\frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

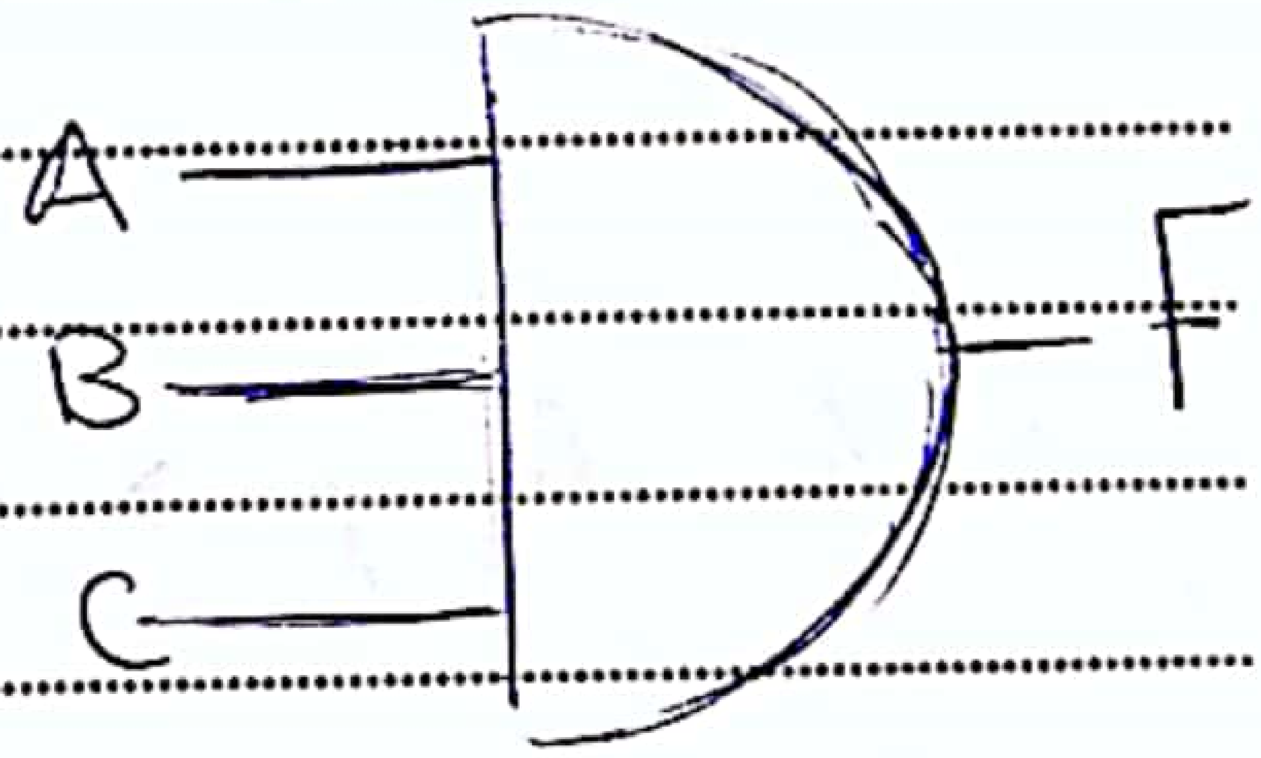
chapter 2 : Combinational logic circuits

Binary logic gates , Bolean Algebra

→ boolean equations and variables , have limited number of possibilities

→ you can represent a logical circuits by boolean equation , truth table + figure and timing diagram.

* $A \cdot B \cdot C = F$ →



$2^3 = 8$ combinations in truth table

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

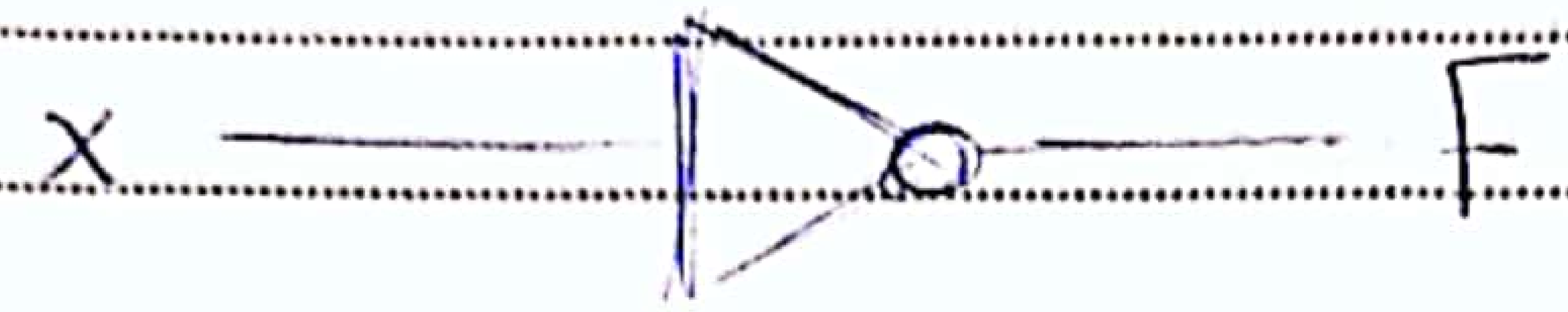
→ True ما بطلع
الا اذا كله true

S T A R S N O T E B O O K

SUBJECT:

في OR ، لو مدخلة وحدة عبارة true ع الأقل، فالناتج حيكون true

not gate (inverter)



$F = \overline{X}$ →

X	F
0	1
1	0

Gate delay → عملياً ~~[illegible]~~ موجودة لكن مهملة [illegible] حالياً

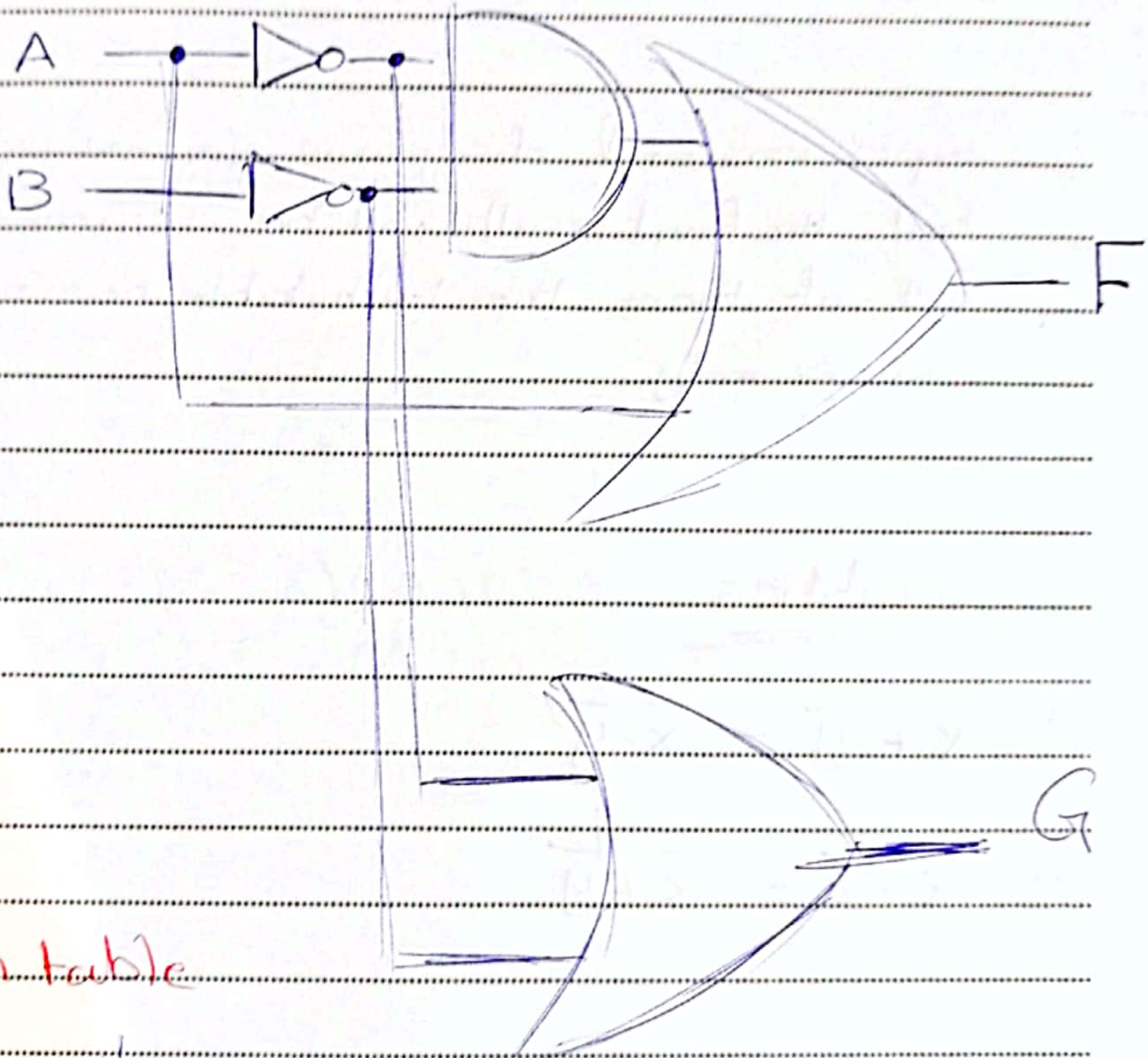
SUBJECT: / /

Example slide 16 [redacted]

$F(A,B) = \bar{A}\bar{B} + A$

$G(A,B) = \bar{A} + \bar{B}$

→ show the logic diagram



truth table

A	B	F	G
0	0	1	1
0	1	0	1
1	0	1	1
1	1	1	0

* الرسمات وال expressions ممكن يختلفوا لنفس ال truth table ، الي إله شكل واحد بس

* حتى لو عندي رسمتين لنفس ال expression ولهم نفس النتيجة رياضيًا ، بس هما كدوائر كهربائية يكونوا مختلفات تمامًا ، بس هما نفسهم لِ truth table واحد

expressions and diagrams are not unique, but the final result will be equivalent for all of them the truth table has one shape only.

Identities

① $\overline{X+Y} = \overline{X} \cdot \overline{Y}$

② $\overline{X \cdot Y} = \overline{X} + \overline{Y}$

اهم طريقة في الإثبات استخدام الـ truth table عشان نبين إنه النتيجة صح

proof for Minimization

$xy + \bar{x}y = y$

$y(x + \bar{x}) =$

$y \cdot 1$

y #

proof for simplification

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

$A + \bar{A} \cdot B = (A + \bar{A})(A + B)$

$1 \cdot (A + B) = (A + B)$ #

proof of consensus theorem

$$AB + \bar{A}C + BC = AB + \bar{A}C$$

$$AB + \bar{A}C + 1 \cdot BC =$$

$$AB + \bar{A}C + (A + \bar{A}) \cdot BC$$

$$AB + \bar{A}C + ABC + \bar{A}BC =$$

$$AB + ABC + \bar{A}C + \bar{A}BC =$$

$$AB(1 + C) + \bar{A}C(1 + B) =$$

$$AB \cdot 1 + \bar{A}C \cdot 1 =$$

$$AB + \bar{A}C \ \#$$

Scanned with CamScanner

SUBJECT: / /

Exercise slide 22

Simplify /boolean algebra

$y + \bar{x}z + x\bar{y}$

$y + x\bar{y} + \bar{x}z$

$(y+x)\cdot(\bar{y}+y) + \bar{x}z$

$(y+x)\cdot 1 + \bar{x}z$

$y + x + \bar{x}z$

$x + \bar{x}z + y$

$(x+\bar{x})\cdot(x+z) + y$

$1\cdot(x+z)+y$

$x+z+y$ # simplified answer

SUBJECT: ....

Exercise slide 23

$(x+y)(\bar{x}+z) = xz + \bar{x}y$

$x\bar{x} + xz + \bar{x}y + yz =$

$0 + \bar{x}y + xz + yz =$

$\bar{x}y + xz + (x+\bar{x}) \cdot yz =$

consensus

$\bar{x}y + xz + xyz + \bar{x}yz$

$\bar{x}y + \bar{x}yz + xz + xyz =$

$\bar{x}y(\overset{1}{\cancel{1+z}}) + xz(\overset{1}{\cancel{1+y}}) =$

$\bar{x}y + xz$

Scanned with CamScanner

SUBJECT: / /

Excercise slide 24

$\bar{x}+xy+x\bar{z}+x\bar{y}\bar{z}=\bar{x}+y+\bar{z}$

$\bar{x}+xy+x\bar{z}(\overset{1}{\cancel{1+\bar{y}}})=$

$\underbrace{\bar{x}+xy}+x\bar{z}=$

$(\bar{x}+x)(\bar{x}+y)+x\bar{z}=$

$\bar{x}+y+x\bar{z}=$

$\bar{x}+x\bar{z}+y=$

$(\bar{x}+x)(\bar{x}+\bar{z})+y=$

$\bar{x}+\bar{z}+y$ #

Dualting حتى لما أحول لازم أحافظ على الأولويات

بنتبه إذا تحويل الإشارات بغير الأولويات

أولاً، إذا كانت بغيرها، بحط أقواس عشان

أرجع الأولويات زي ما كانت في المعادلة الأصلية

العمليات بتنقلب، والـ 0 بيصير 1 و بالعكس

* it helps for finding the complementing functions

Exercise slide 28

(1) complement $G = (a + \bar{b}c)d + \bar{e}$

$\rightarrow$ duality

$$G = ((a \cdot (\bar{b} + c)) + d) \cdot \bar{e}$$

$$\bar{G} = ((\bar{a} \cdot (b + \bar{c})) + \bar{d}) \cdot e$$

(2) $G = ab(c + xy)$

$$\text{dual } G = (a+b) + (c \cdot (x+y))$$

SUBJECT: ....

③ $G = \bar{x}yz + \bar{x}y\bar{z} + xz$

$= (\bar{x}+y+z)\cdot(\bar{x}+y+\bar{z}) + (x+z)$

$\bar{G} = (x+\bar{y}+\bar{z})\cdot(x+\bar{y}+z) + (\bar{x}+\bar{z})$

end of part 1

Scanned with CamScanner

when you say you have 8 minterms, it means you have 8 combinations

← طريقتين عن ال 0 وال 1 للتعبير عن ال combinations

A	B	C	minterms
0	0	0	m_0
0	0	1	m_1
0	1	0	m_2
0	1	1	m_3
1	0	0	m_4
1	0	1	m_5
1	1	0	m_6
1	1	1	m_7

8 minterms

$m_0 \rightarrow m_7$

← اذا بدك تعرف الرقم المقابل للترميز خذ ال رقم إلي أسفل حرف ال m وحوله لـ binary

← $m_3 = 011$

* طريقة ثانية للكتابة باستخدام رموز او كـ expressions

A	B	C	minterms
0	0	0	$\bar{A}\bar{B}\bar{C}$
0	0	1	$\bar{A}\bar{B}C$
0	1	0	$\bar{A}B\bar{C}$
0	1	1	$\bar{A}BC$
1	0	0	$A\bar{B}\bar{C}$
1	0	1	$A\bar{B}C$
1	1	0	$AB\bar{C}$
1	1	1	ABC

using expressions to write minterms

← هاي طريقتين للكتابة الـ output هي مستطاعة تعرفوا ال combinations بالثلاث طرق محلولين

SUBJECT:

minterms help in solving the following:

اذا أعطاني ال Truth table بيقدر اكتب ال f
كمجموع لل minterms

find F(A,B,C) ع جمع طلاعي:

sum-of-minterms (SOM)
you find ones, and write the answer as minterms.

So in the latest example

$F(A,B,C) = \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}BC + AB\bar{C} + ABC$

$F(A,B,C) = m_1 + m_2 + m_3 + m_6 + m_7$

equivalent to $\rightarrow \sum_m (1,2,3,6,7)$

SUBJECT: ....

example → output عشوائي

	X	Y	H
m_0	0	0	0
m_1	0	1	0
m_2	1	0	1
m_3	1	1	1

$$H(X,Y) = m_2 + m_3$$

$$= \sum_m (2,3)$$

$$= x\bar{y} + xy$$

$\underline{\underline{3}}$ ← minterm لما يكونوا 3 احرف بكون عندي ال

$\underline{\underline{2}}$ ← 〃 〃 〃 〃 〃 حرفين 2 〃 〃

$\underline{\underline{4}}$ ← 〃 〃 〃 〃 〃 4 احرف 〃 〃

A	B	C	D	
0	0	0	0	$m_0 \rightarrow \bar{A}\bar{B}\bar{C}\bar{D}$
1	1	1	1	$m_{15} \rightarrow ABCD$

SUBJECT: ...... / /

* وبسمة ال SOM داتاً عبارة عن And's
ويتجمع or ومرة

you can find F(A,B,C) as
→ (S-O-M) : sum of minterms
→ (P-O-M) : product of max terms

* متساوين رياضياً
مختلفين كهربائياً

example:

A	B	C	maxterms		(F) output عشوائي
0	0	0	M_0	$A+B+C$	→ 1
0	0	1	M_1	$A+B+\bar{C}$	→ 0
0	1	0	M_2	$A+\bar{B}+C$	→ 0
0	1	1	M_3	$A+\bar{B}+\bar{C}$	→ 1
1	0	0	M_4	$\bar{A}+B+C$	→ 1
1	0	1	M_5	$\bar{A}+B+\bar{C}$	→ 1
1	1	0	M_6	$\bar{A}+\bar{B}+C$	→ 0
1	1	1	M_7	$\bar{A}+\bar{B}+\bar{C}$	→ 0

find zeros and multiply them to each other

$$F(A,B,C) = M_1 \cdot M_2 \cdot M_6 \cdot M_7$$

$$= (A+B+\bar{C}) \cdot (A+\bar{B}+C) \cdot (\bar{A}+\bar{B}+C) \cdot (\bar{A}+\bar{B}+\bar{C})$$

$$= \prod_M (1, 2, 6, 7)$$

SUBJECT: .... / /

* ورجعة ال (POM) بكل OR وبينهم and وحدة.

example:

x	y	F
0	0	0
0	1	0
1	0	1
1	1	1

$F = M_0 \cdot M_1$

$F = (x+y) \cdot (x+\bar{y})$

A	B	C	minterms		maxterms
0	0	0	$\bar{A}\bar{B}\bar{C}$	$\longrightarrow$	$A+B+C$
↓					
1	1	1	ABC	$\longrightarrow$	$\bar{A}+\bar{B}+C$

$\overline{m_i} = M_i$

$\overline{M_i} = m_i$

each is the complementation of the other

SUBJECT:

ملاحظة:

ال maxterms وال minterms متعاكسات

لكن ال SOM وال POM متساويات رياضياً
ومختلفات كهربائياً

Exercise slide 33

$F = m_0 + m_1 + m_4 + m_5 + m_6 + m_7$
$= \bar{x}\bar{y}\bar{z} + \bar{x}\bar{y}z + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z} + xyz$

Exercise slide 34

① $F(A,B,C,D,E) = m_2 + m_9 + m_{17} + m_{23}$

A	B	C	D	E	F
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	1 (m_2)
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	0
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	1 (m_9)

* only m_2, m_9, m_{17}, m_{23} are ones

and so on

SUBJECT: .... / /

② $G(x,y,z) = \Sigma m(2,5,6)$

$= G(x,y,z) = m_2 + m_5 + m_6$

x	y	z	G
0	0	0	0
0	0	1	0
0	1	0	1 (m_2)
0	1	1	0
1	0	0	0
1	0	1	1 (m_5)
1	1	0	1 (m_6)
1	1	1	0

~~[illegible]~~

Exercise slide 38

② $G(x,y,z) = \Pi_M(0,6,7)$

$= M_0 + M_6 + M_7$

x	y	z	G
0	0	0	0 (M_0)
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0 (M_6)
1	1	1	0 (M_7)

SUBJECT: ....

Q: $F(A,B,C) = AB\bar{C} + AC$

is it (SoM)?

No, the second (and) does not hold all inputs.

it's now called in this form, the sum of product

(SoP) is special case of (SoM)

Q: $F(A,B) = \bar{A}\bar{B} + AB$ → is it (SoM)?

yes, it is, each and hold all inputs.

لما يسأل سؤال زي هيك لا تسقط جوا وحس المعطيات كحالة.

Q: $F(A,B,C) = A + \bar{C}(\bar{A} + B)$

is it (SoM)?

No, it's not (SoM) and not even a (SoP)

Q: $F(A,B,C) = (A+\bar{B}+\bar{C})(A+C)$

is it (PoM)?

No.

Special cases ?! weird cases.

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

↳ it can be considered a (PoM)

$F(A,B,C,D) = \bar{A}\bar{B}CD$

↳ it can be considered a (SoM).

معناها (and) جاية

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

it's not SoM

but SoP

(ج or جاية)

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

it's not PoM

but PoS (product of sum)

SUBJECT: ....

SoM (sum of minterms)

General case: SoP (sum of product)

PoM (product of maxterms)

General case: PoS (product of sum)

Excersize slide 43

what is the SoM form of

$F(A,B,C) = A + B'C$ → عملت عليها
بحيث ما تأخذ حقيقتها رياضية

بال truth table ← اسهل طريقة للحل :-

بالحت وين الـ ones ، ووين الـ zeros ، وحسب المطلوب بجمع او بضرب

A	B	C	
0	0	0	→
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

SUBJECT: / /

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

$A \cdot (B + \bar{B}) \cdot (C + \bar{C}) + \bar{B}C \cdot (A + \bar{A})$

$ABC + AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C} + \cancel{A\bar{B}C} + \bar{A}\bar{B}C$ → بشيل المتشابهات

$ABC + AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C} + \bar{A}\bar{B}C$

$= m_7 + m_6 + m_5 + m_4 + m_1$

Exercise

$F(w,x,y) = wx + \bar{x}y$

$= (wx + \bar{x}) \cdot (wx + y)$

$= (w + \bar{x}) \cdot \underbrace{(x + \bar{x})}_{1} \cdot (w + y) \cdot (x + y)$

$= (w + \bar{x} + y \cdot \bar{y}) \cdot (w + y + x \cdot \bar{x}) \cdot$
$(x + y + w \cdot \bar{w})$

$(w + \bar{x} + y) \cdot (w + \bar{x} + \bar{y}) \cdot (w + y + x) \cdot \cancel{(w + y + \bar{x})}$
$\cdot \cancel{(x + y + w)} \cdot (x + y + \bar{w})$

$= (w + x + y) \cdot (w + \bar{x} + y) \cdot (w + \bar{x} + \bar{y}) \cdot (\bar{w} + x + y)$

$= M_0 \cdot M_2 \cdot M_3 \cdot M_4$

SUBJECT: ....

اختلاف الـ Function كهربائياً، يختلف معه عدد الـ transistors، وبختلف التكلفة

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

optimal F

كيف أعرف أحسن دالة كهربائياً؟

Way (1) Literal cost → بحسب عدد الأحرف *

* مش دايماً بتزبط، لأنه أحياناً بكون في دالتين بنفس عدد الأحرف، وما بقدر أميز. (مش دقيقة)

Way (2) Gate Input cost → بحسب عدد الـ inputs *

لكن ما بحسب الـ inverters

← برسم الدالة وبعد عدد الـ inputs ومجموعها G

والـ GN هي G + عدد الـ inverters

G = L + #terms الحكم

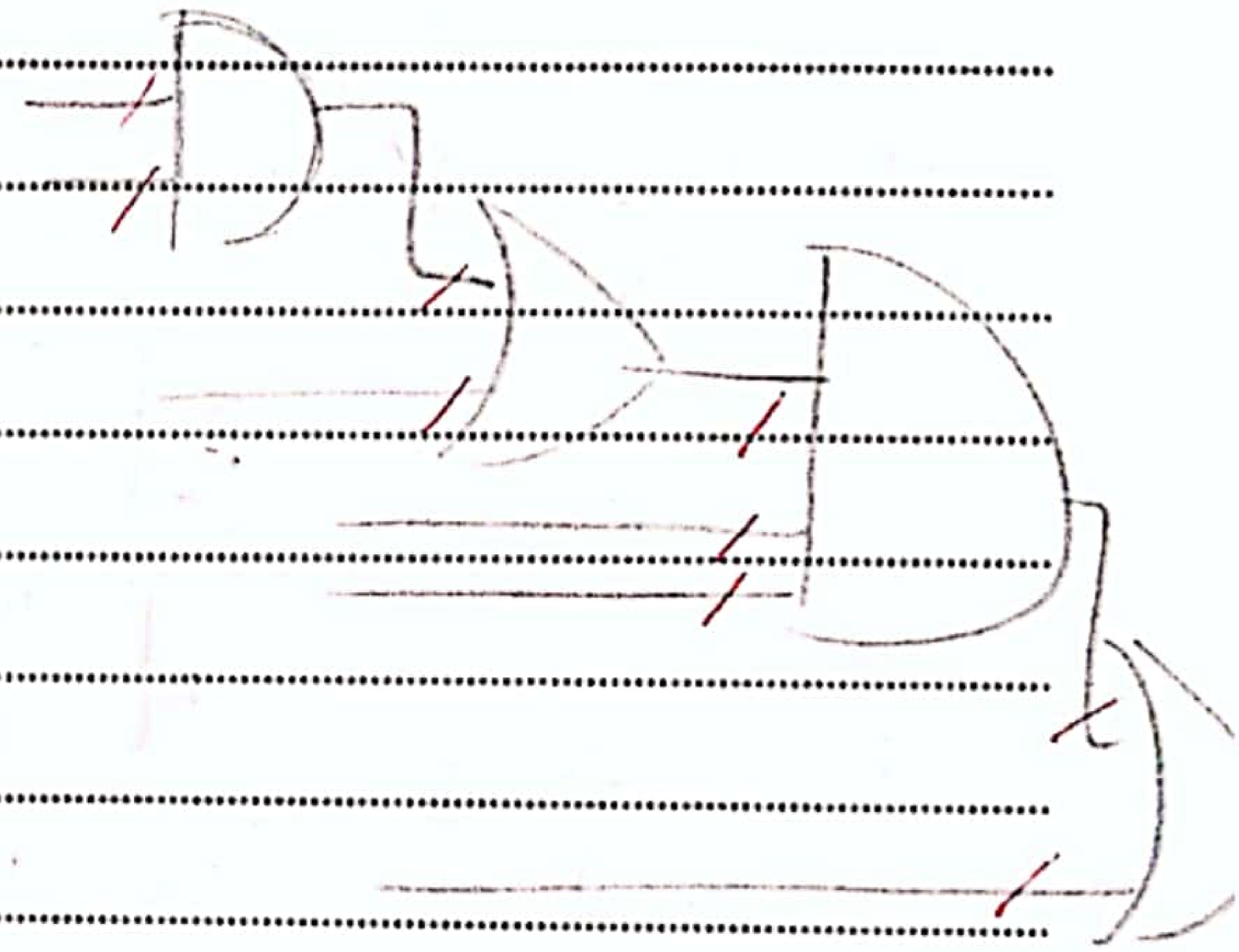
SUBJECT: .... / /

example

$F = A\bar{B}(A + C\bar{D}) + B$

G1 = 9

GU = 11



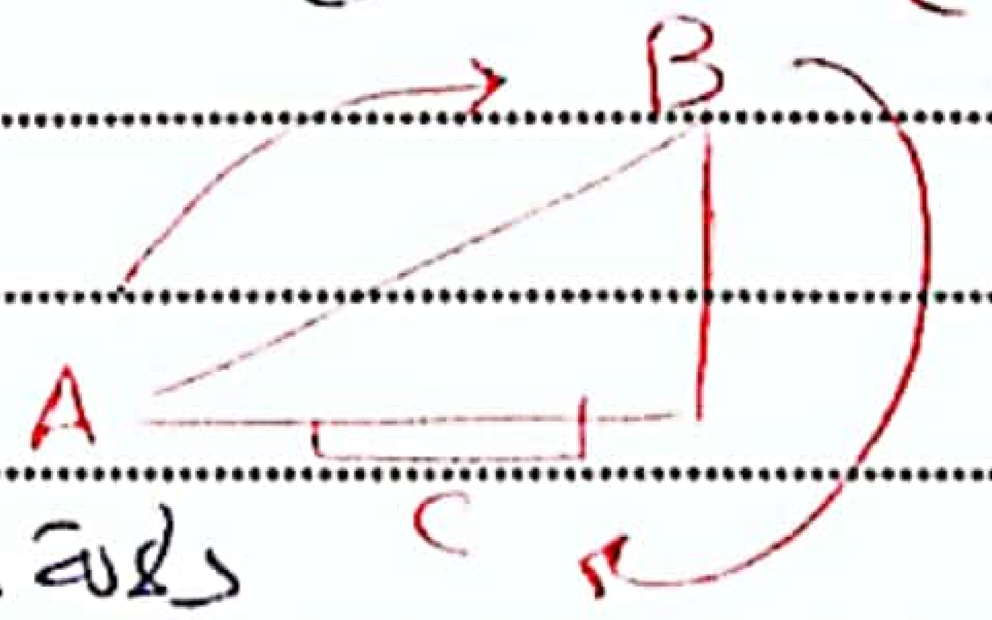
S T A R S N O T E

[Karnaugh Map]

بيرتب الأرقام و ال combinations جوا ال array باستخدام ال Grey code

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

	$\bar{B}$		B	
	$\bar{C}$	C	C	$\bar{C}$
$\bar{A}$ (قيمة 0 دايم)	000 #0 1	001 #1 0	011 #3 0	010 #2 1
A (قيمة 1 دايم)	100 #4 1	101 #5 0	111 #7 1	110 #6 0



دلالة ال رمز الي بيثبت لهاد الأرقام .

* فكرة ثانية ، رقم المربع الداخلي = المساحة

يعني مثلاً B و Ā و C بيتقاطعوا عند مربع 3 وترجمتهم ĀBC
وترجمة لتميزهم 011 ، الي هي 3 برضو

SUBJECT: .... / /

w	x	y	
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

→ tabular form

x

000	001	011	010
1	0	0	0
100	101	111	110
1	1	0	1

w

y

Array form
multi dimentional form

* The only difference between truth table and k-map is the visualizing (different forms)

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

B

00	01
1 #0	0 #1
10	11
0 #2	1 #3

A

SUBJECT: / /

A	B	C	D
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
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

		$\bar{C}$		C		
$\bar{A}$		0	1	3	2	$\bar{B}$
$\bar{A}$		4	5	7	6	B
A		12	13	15	14	B
A		8	9	11	10	$\bar{B}$
		$\bar{D}$	D	D	$\bar{D}$	

SUBJECT:

Exercise slide 60

$\boxed{F}$

0	1	3	2
1	1	0	0
4	5	7	6
1	1	1	1

$\boxed{G}$

0	1	3	2
1	0	0	1
4	5	7	6
1	1	0	1

$\boxed{H}$

0	1	3	2
0	1	1	0
4	5	7	6
1	1	0	1

طب هلا، الفرق بين ال k-map وال truth table

① وحدة بالطول ووحدة على شكل array

② ال k-map بتعتمد ال grey code، ال truth table لا.

SUBJECT: .... / /

[Exercises for algebric simplification]

① $\bar{w}x(\bar{z}+\bar{y}z)+x(w+\bar{w}yz)$

simplify to one letter

$x(\bar{w}\bar{z}+\bar{w}\bar{y}z+w+\bar{w}yz)$

$x(w+\bar{w}\bar{z}+\bar{w}\bar{y}z+\bar{w}yz)$

$x(\cancel{(w+\bar{w})}(w+\bar{z})+\bar{w}\bar{y}z+\bar{w}yz)$ باستخدام قانون ال simplification

$x(w+\bar{z}+\bar{w}\bar{y}z+\bar{w}yz)$

$x(\cancel{(w+\bar{w})}(w+\bar{y}z)+\bar{z}+\bar{w}yz)$

$x(w+\bar{y}z+\bar{z}+\bar{w}yz)$

$x((w+\bar{w})(w+yz)+\bar{y}z+\bar{z})$

$x(w+yz+\bar{y}z+\bar{z})$

$x(w+(\bar{z}+\bar{y})(\bar{z}+z)+yz$

$x(w+\bar{z}+\bar{y}+yz)$

$x(w+(\bar{z}+z)\cdot(\bar{z}+y)+\bar{y})$

$x(w+\bar{z}+y+\bar{y})$

$x(w+y+1) \rightarrow$ one + anything = 1

$x\cdot 1$

X

SUBJECT: / /

② $A\bar{D}+\bar{A}B+\bar{C}D+\bar{B}C = (\bar{A}+\bar{B}+\bar{C}+\bar{D})(A+B+C+D)$

prove

use right

$= \overset{0}{\cancel{\bar{A}A}} + \bar{A}B + \bar{A}C + \bar{A}D + \overset{0}{\cancel{\bar{B}B}} + \bar{B}A + \bar{B}C + \bar{B}D +$

$\overset{0}{\cancel{\bar{C}C}} + \bar{C}A + \bar{C}B + \bar{C}D + \underset{0}{\cancel{\bar{D}D}} + \bar{D}A + \bar{D}B + \bar{D}C$

العلاقة الموجودة

الي بكل دسته هاي

هي ال consensus

SUBJECT: ....

* باستخدام الطرف اليسار
كأنه عندي SOP بدي احولها ل POS موجودة
ببساطة ال SOP وال POS كم

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

دمجها كأنها variable واحد
وبوزعها على BC

*لا يمكن بدي توسيع خطوتها
اساس لانه حيبين مزعم و صعب
مفهوم بس صعب

$$(A\bar{D} + \bar{A}B + \bar{C}D + \bar{B})(A\bar{D} + \bar{A}B + \bar{C}D + C)$$ using simplification

$$(A\bar{D} + \bar{A} + \bar{B} + \bar{C}D)(A\bar{D} + \bar{A}B + \bar{C}D + C)$$ ״

$$(\bar{D} + \bar{A} + \bar{B} + \bar{C}D)(A + D + C + \bar{A}B)$$ ״

$$(\bar{D} + \bar{C} + \bar{A} + \bar{B})(A + B + D + C)$$ #

SUBJECT: / /

* الsimplification دال cons-- بيساعدني بال reducing عدادالحدود

* ال طريقة بتجيب بالليات SOP و POS والتحويل بينهم بيساعدني اني اكبر الحدود واصغرها

if A = B
then dual A = dual B
← وممكن تكون حد اخر سؤال

لو بكتب اني اثبت تاني بسطر تاني بجنها باستحق ابسط ، وهيك بكون اثبت الي فوق

How to simplify with k-maps

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0

	$\bar{B}$	B
$\bar{A}$	0: $\boxed{1}$	1: $\boxed{1}$
A	2: 0	3: 0

جم بستعمل رلي بعطي اكبر عدد من ال ones ممكن يكون
(1, 2, 4, 16, 32 ... 2^x (عدد ال ones الي بعطيها)

في اثبات رياضي بحكي انه ال simplification بصير لما آخذ
الرموز الي بتثبت لاكبر عدد ones .
الرموز الي ثبتت لاكبر عدد من ال ones هو ال $\bar{A}$
لانه ال B مرة ب 1 ، ومرة ال $\bar{B}$ ال $\bar{A}$ ثابتة

So $F(A,B) = \bar{A}$

* ولو بدي اثبتها algebric ا
ال Terms الي عندي هي ال ones هم 0 و 1
يعني $\bar{A}\bar{B}$ و $\bar{A}B$ ← فانا بصير اعمل الهم
سوا $\bar{A}B + \bar{A}\bar{B}$ ال simplification بجمع
طلع $\bar{A}$

Scanned with CamScanner

example

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

		B
	0	1
	0	1
A	2	3
	1	1

برسم مستطيلين ، عادي يكونوا متقاطعين ، بس ممنوع يكون واحد كلياً داخل شي.

F(A,B) = B + A

example

x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

	$\bar{y}$		y	
	0	1	3	2
	1	1	0	0
x	4	5	7	6
	1	1	0	0
		z	z	

$F(x,y,z) = \bar{y}$

Scanned with CamScanner

SUBJECT: .... / /

example

			x	x
		y	y	
$\bar{w}$	1	1	1	1
w	0	0	1	0

$F(w,x,y) =$
$\bar{w} + xy$

لما نرسم مستطيلات لازم نلتزم بقاعدتين :

① Minimize number of rect.

② Maximize size of rect.

اقل عدد من المستطيلات بغطوا كل ال ones ، بأكبر حجم ممكن .

* كل ما صغر حجم المستطيل ، بزيد عدد الاحرف الي بكتبها

S T A R S N O T E B O O K

Example

	$\bar{B}$	$\bar{B}$	B	B
	0	1	3	2
$\bar{A}$	1	1	0	1
	4	5	7	6
A	1	0	0	1
	$\bar{C}$	C	C	$\bar{C}$

ما بصير عشان بندخل 3 مع المستطيل

Remeber that basically the k-map is three dimentional

→ which means that edges 2 and 6 are connected with 0 and 4

→ and so one with the horizontal edges 0,1,3,2 with 4,5,7,6.

$F = \bar{C} + \bar{A}\bar{B}$ ✓

SUBJECT:

Example

	$\bar{y}$		y	
$\bar{x}$	0	0	1	1
x	1	1	1	0
		z		

$$F(x,y,z) = \bar{x}y + zx + x\bar{y}$$

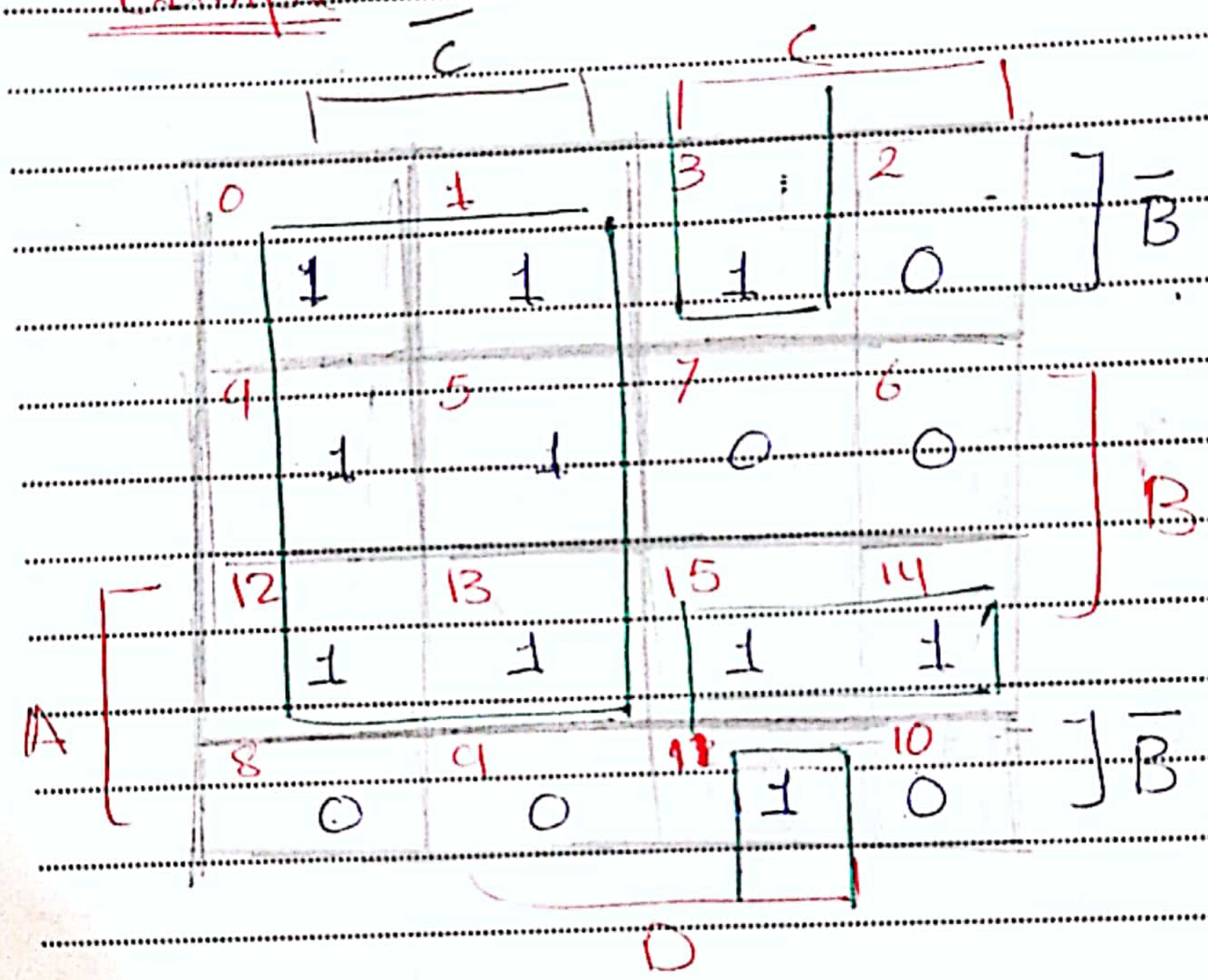
$$\bar{x}y + yz + x\bar{y}$$

example

			y	y
$\bar{x}$	1	0	0	1
x	0	0	1	0
	$\bar{z}$	z	z	$\bar{z}$

$$F = \bar{x}\bar{z} + xyz$$

example



$F = \bar{C} + CD\bar{B} + BAC$

او

$F = \bar{C}\bar{A} + CD\bar{B} + BA$

SUBJECT:

example

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1	0	1	0
$\bar{A}B$	0	1	1	1
AB	0	0	1	0
$A\bar{B}$	0	0	1	0

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + CD + B\bar{A}C + D\bar{A}B$$

example

C

B

A

D

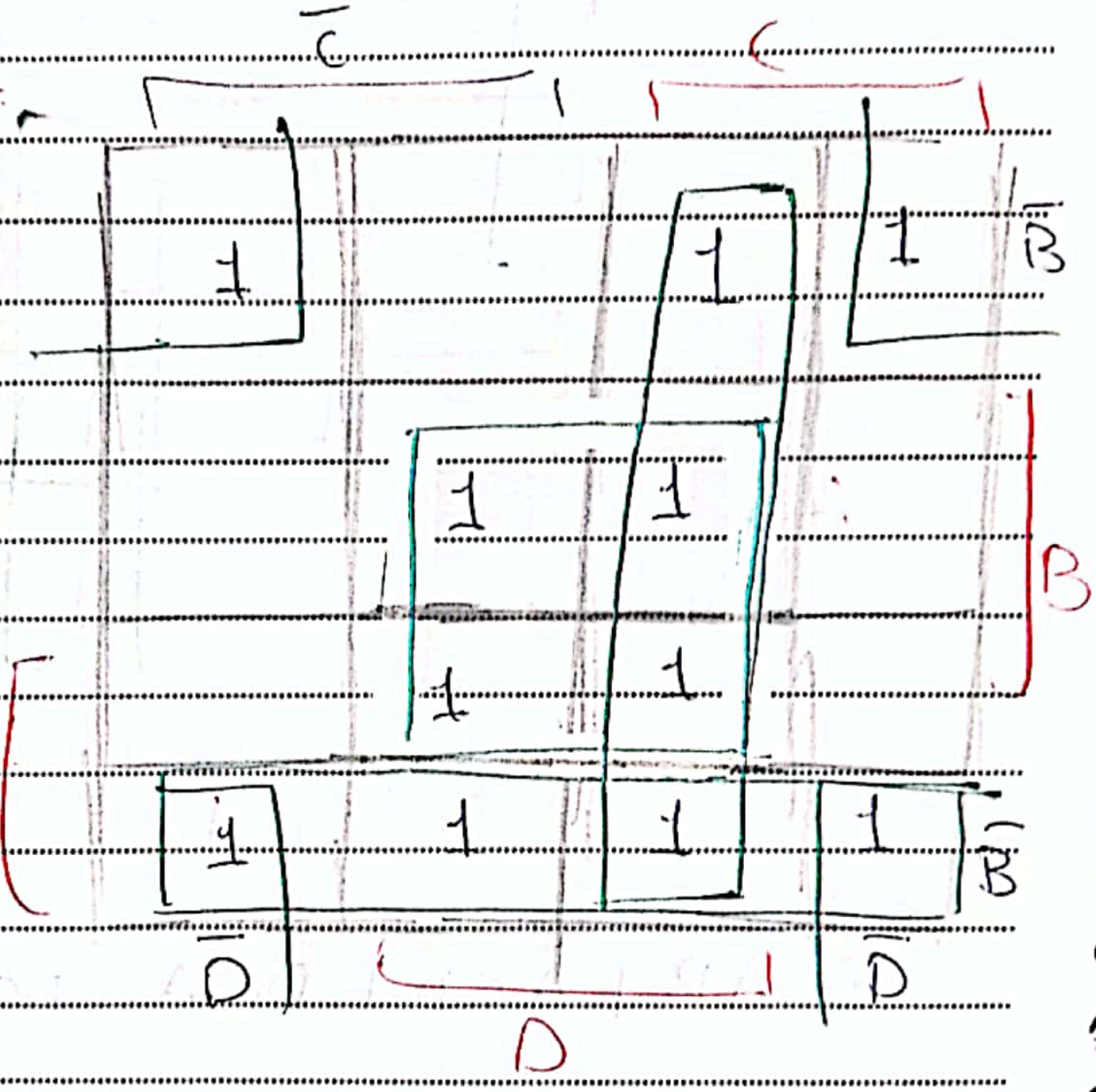
1	0	0	0
1	0	1	0
1	1	1	1
0	0	1	0

$$F = AB + \bar{A}\bar{C}\bar{D} + DCA + DCB$$

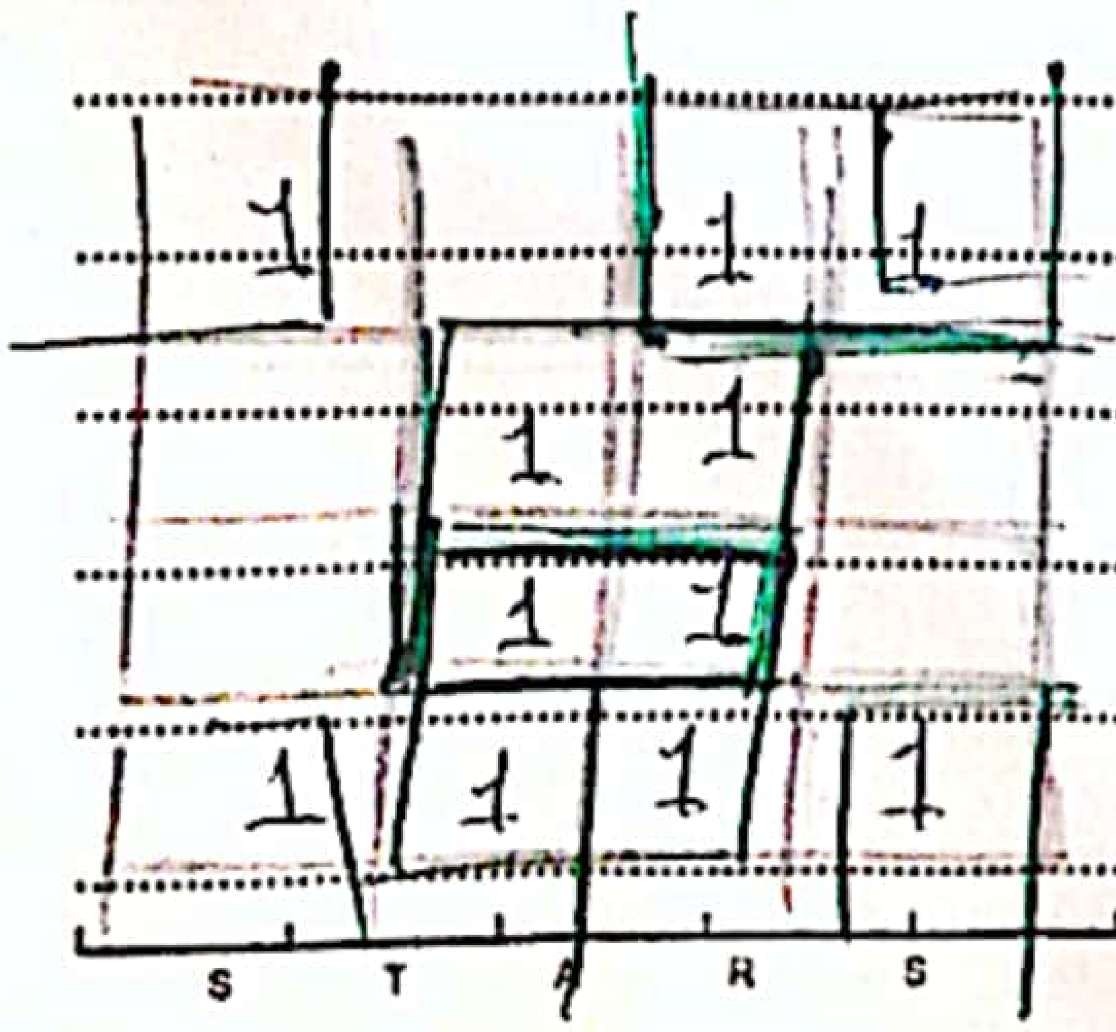
Scanned with CamScanner

مش شرط يكون الحل unique. عادي يكون في أسئلة
الها أكثر من احتمال

example



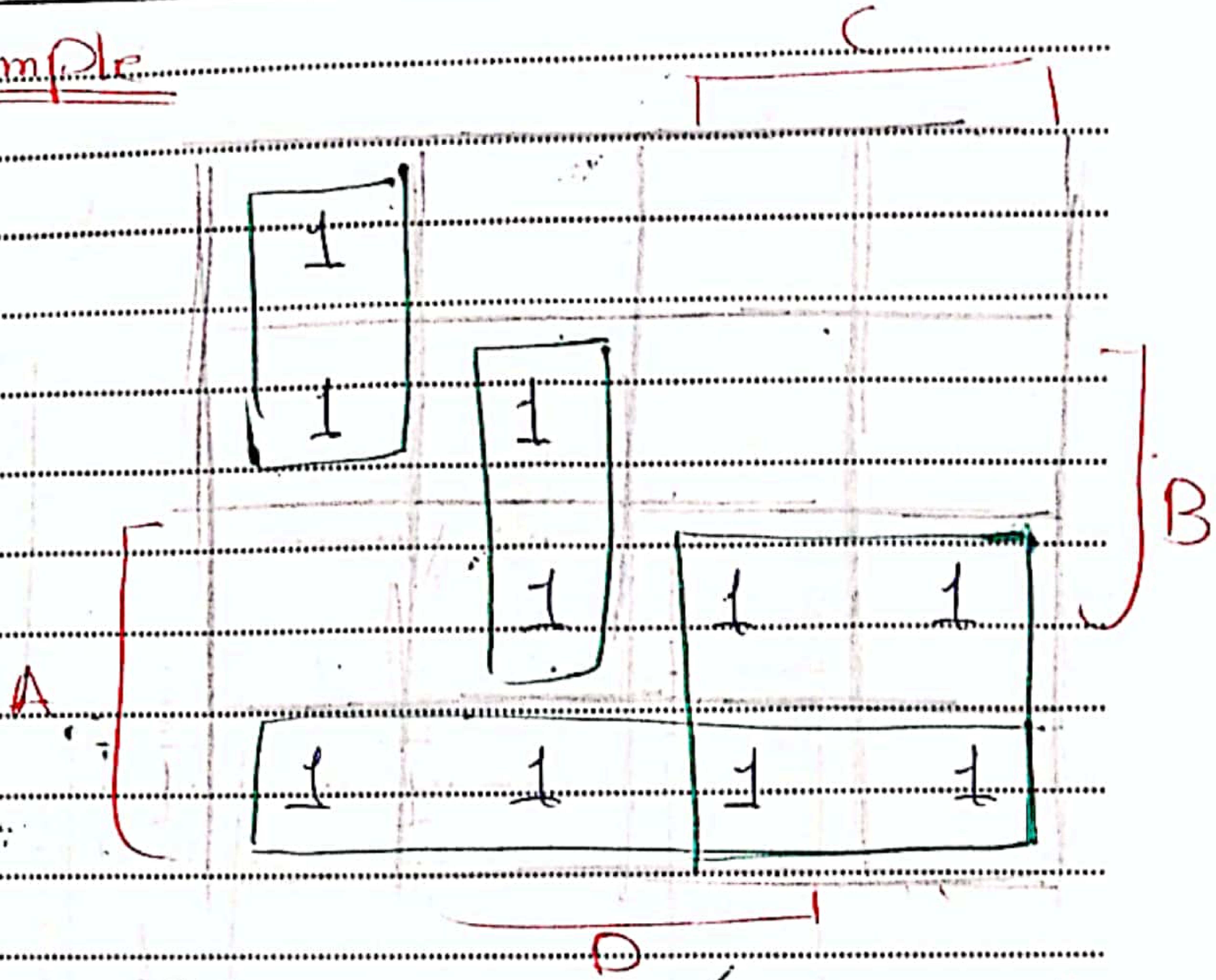
$$F = A\bar{B} + BD + DC + \bar{B}\bar{D}$$



ممكن حل ثاني

SUBJECT:

example

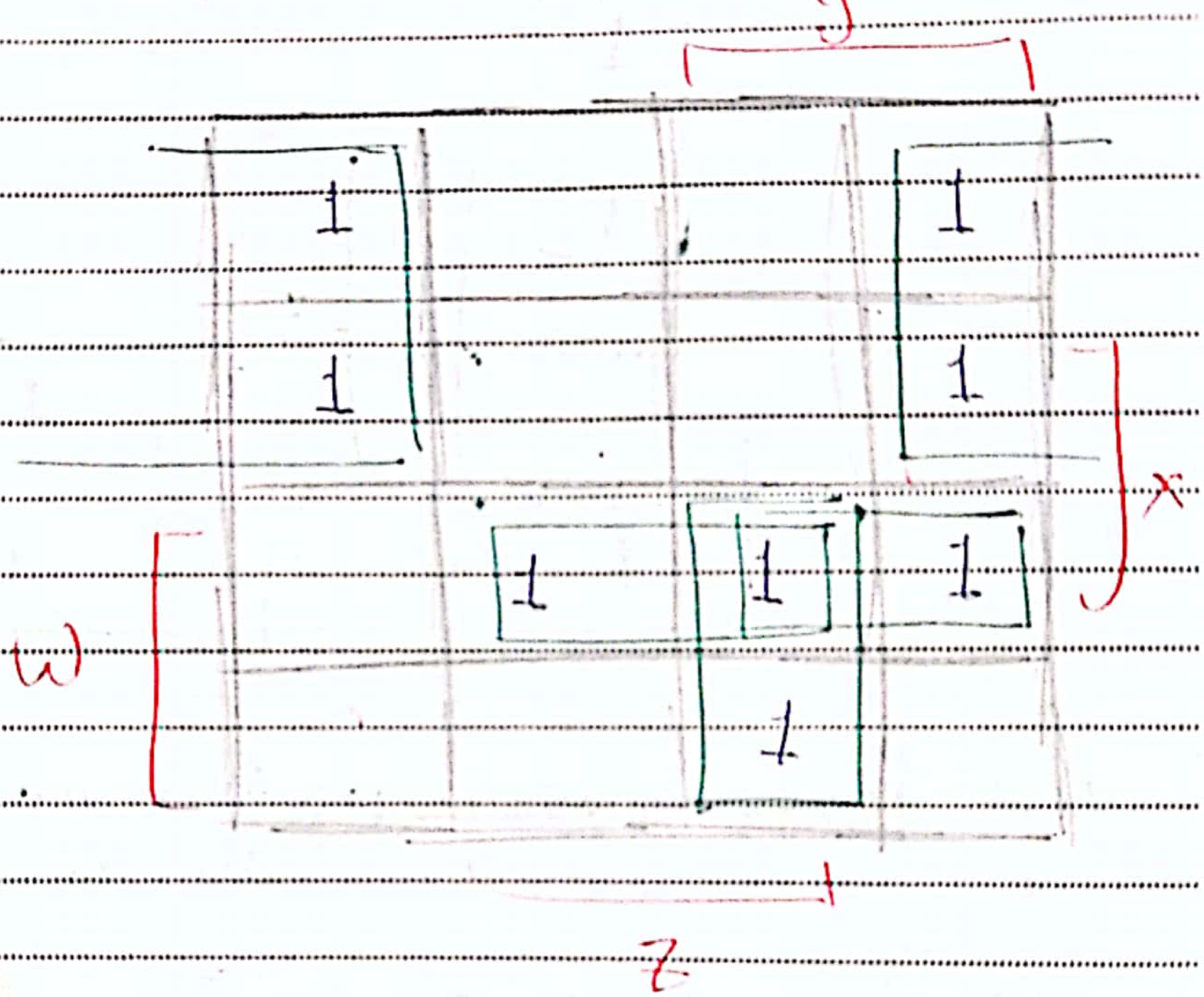


طول ما انا قادر اكبر المربع بحيث يضل حجمه 2^x وحوله مشترك
مع شي بكفي ع بكبره

$$F = A\bar{B} + CA + D\bar{C}B + \bar{C}\bar{D}\bar{A}$$

SUBJECT:

example



$$F = \overline{w}\,\overline{z} + zyw + wxy + wxz$$

SUBJECT: .... / /

New Concept:

example

→ 6 prime implicants

(1) prime implicants → all legal rectangles

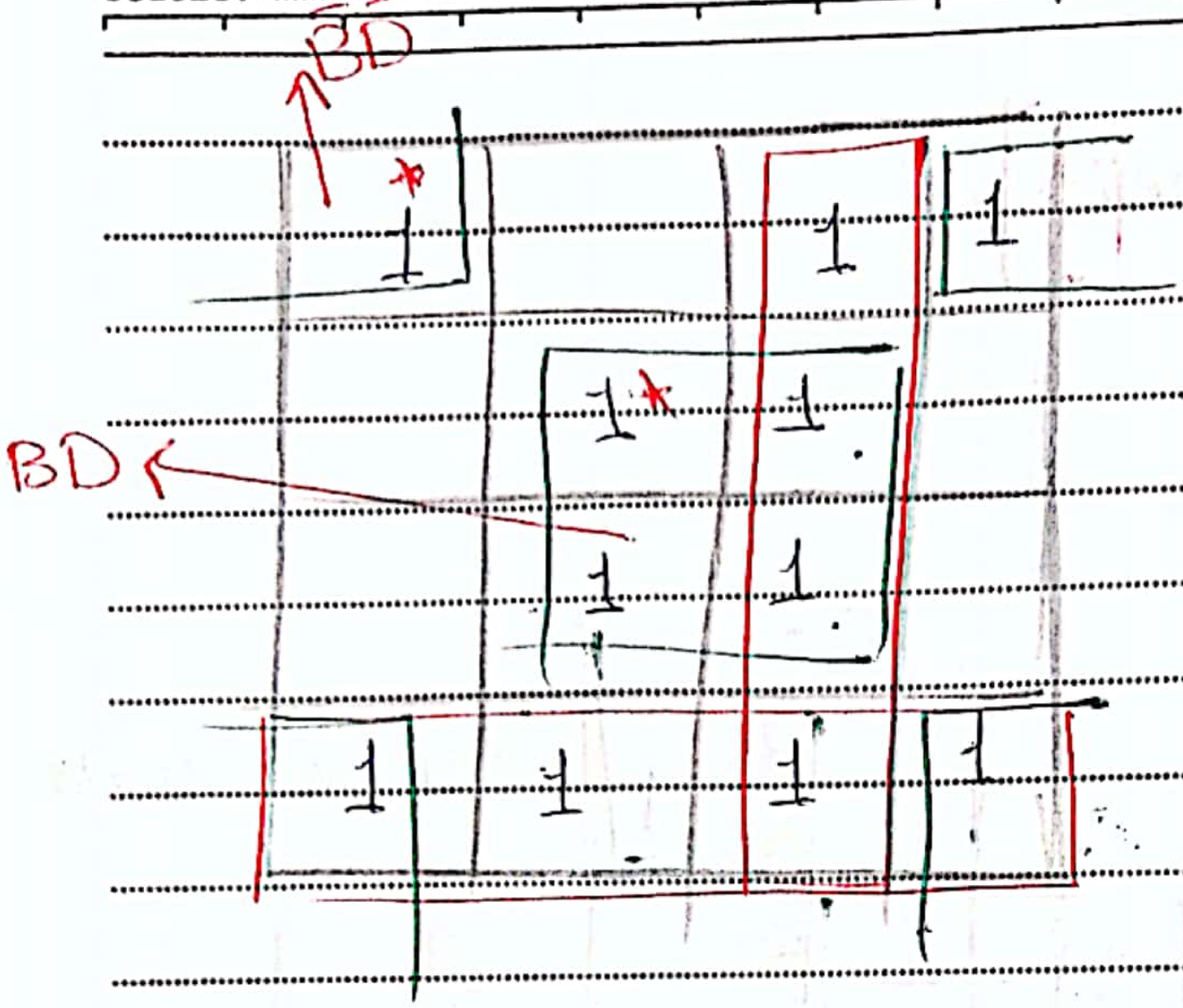
كل المستطيلات الممكنة

~~legal~~

* كل رقم 1 لحاله مستطيله يعتبر legal

(2) essential prime implicants ones المستطيلات اللي لسه مش موجودة مستغيرهم.

SUBJECT:

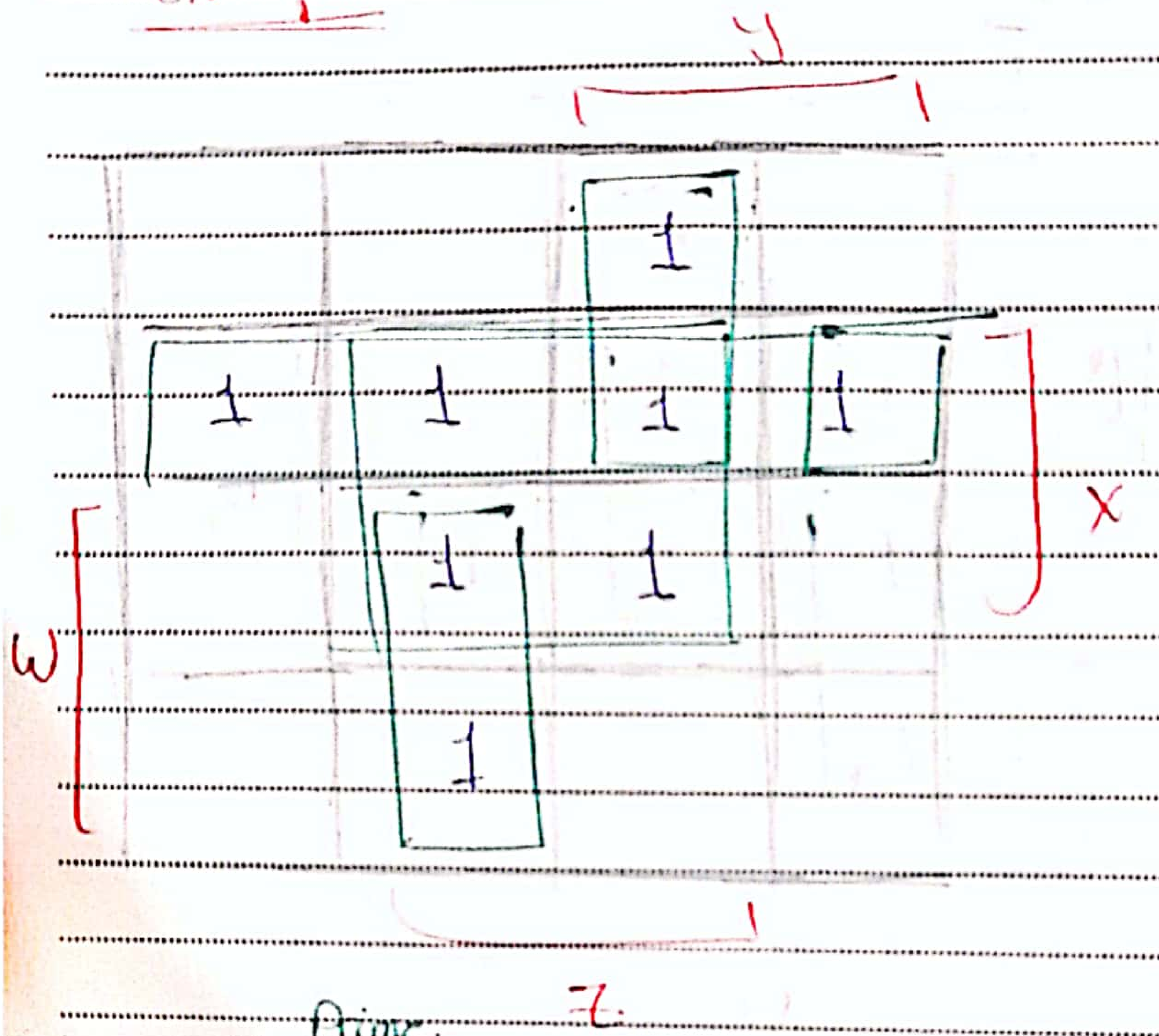


essentials prime implicants

الفكرة إنه شو ما اختلف ال F لما احسبها
ضروري يكون عندي ال essential موجودين شو ما اختلفت الاجابات.

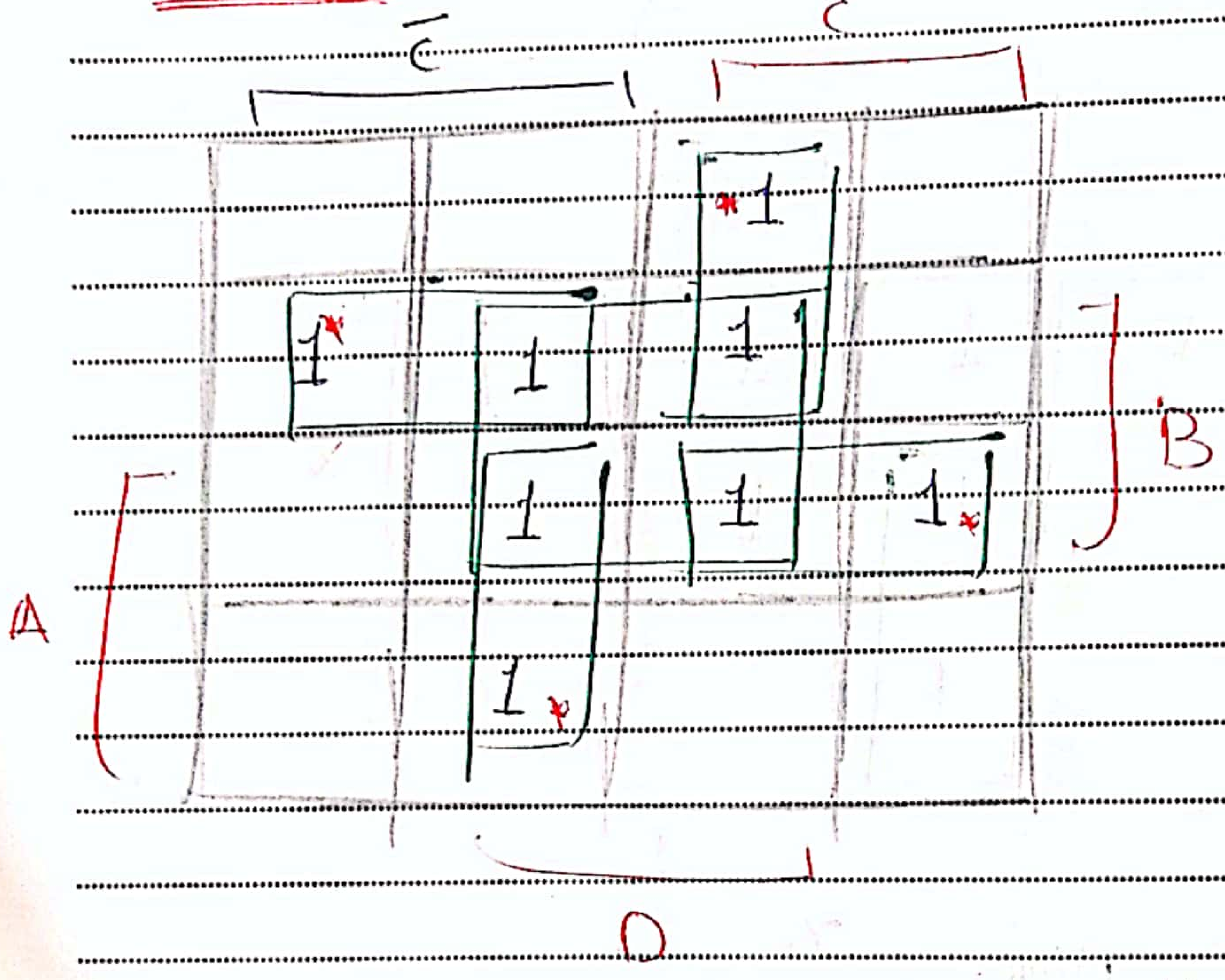
احد ما اطلع ال essential ، ولما يتبقى ones ، باحط اقل عدد مستطيلات بتغطي باقي ال ones
زي الي بالاحمر فوق مثلا.

example



Prime
4 implicants → all are essential
→ unique simplification

example



5 prime implicants
4 essential

$$F = ABC + A\bar{C}D + \bar{A}CD + \bar{A}B\bar{C}$$

SUBJECT: / /

[Don't cares in k-map]

25 دقيقة

Example / slide 70 finding the ideal scenario

احسن طريقة افهم المسألة الي بيها هي ال truth table

w	x	y	z	
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

ال hint الي بحكي انه
لما العدد اصغر 5 = 0
وبعدين لما ترتيب
ك term اكبر من 5

بوقف عند ال 9 لانه
هو عم يمثل ال terms داله
decimal وهو BCD
بوقف عند 9
لانه بحتاج 4 bits
ال input اكثر من 10
combinations

جدول مستقبلي
ال combinations
Unused combinations
their outputs (dont care outputs)
wether you put the output 0,1
either way they won't happen.

⟹ don't care

SUBJECT: .... / /

معناها بحكي إني راح اعبي ال K-Map بالهيك

	$y'z'$	$y'z$	yz	yz'
$w'x'$	0 (0)	0 (1)	0 (3)	0 (2)
$w'x$	0 (4)	1 (5)	1 (7)	1 (6)
wx	X (12)	X (13)	X (15)	X (14)
wx'	1 (8)	1 (9)	X (11)	X (10)

x مسموح استخدام ال dont cares حوالين ال ones
اذا بيساعدوا تكبر المستطيل (زي الخط البني)

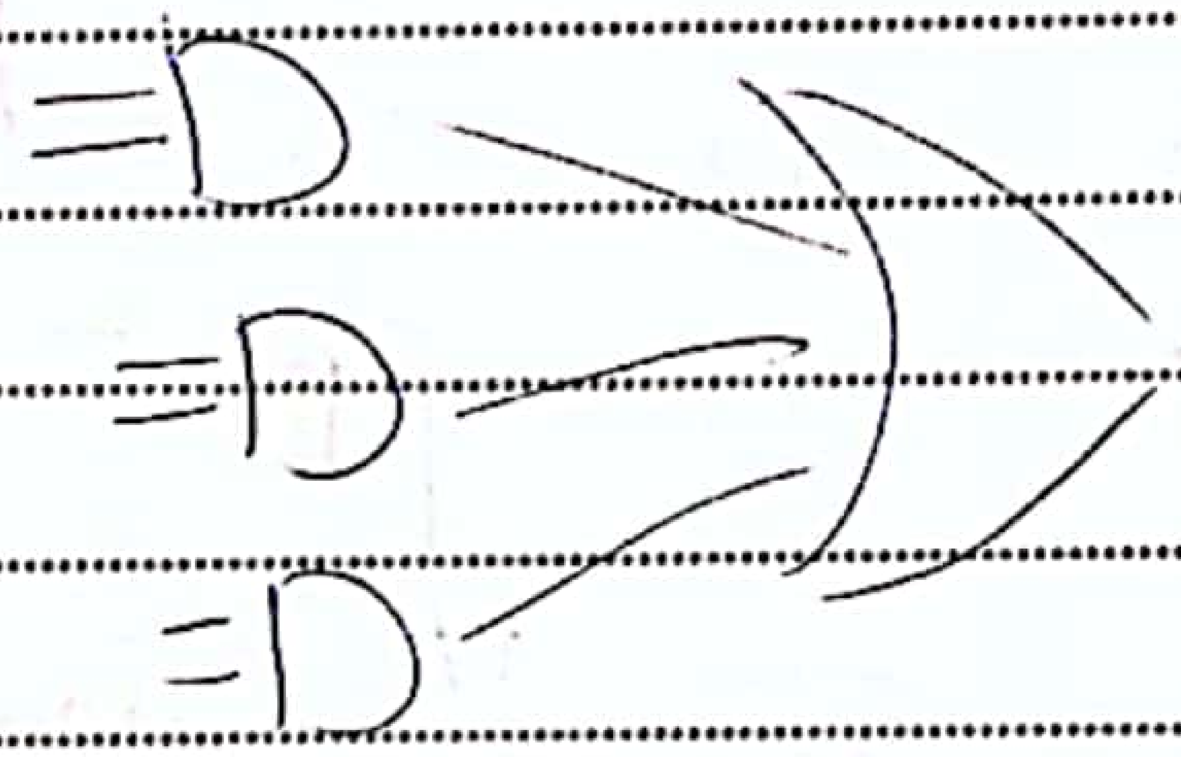
$F = w + xy + xz$

SUBJECT:

[NOTE]

		B	B	
	0	0	1	0
A	1	1	1	1
		C	C	

$F = A + BC \rightarrow$



This is SOP
k-map
simplifies SOP

Find simplified F as SOP ? ← ممكن سؤال
عادي من ال K-map

// // // // POS ← وممكن

ممكن لما يكون حاطط (Ed) زي مثال عل سلايد 72
وقصده كل حاجة don't cares

SUBJECT:

Example slide 72

SOP $\longleftrightarrow$ POS

k-map of $\rightarrow$ F

1	x	0	1
x	1	1	x
0	0	0	0
1	0	0	1

C

B

A

D

طريقة حل:
احسب $\bar{F}$ وحولها
لـ F بال demorgan
وال duality

$$\bar{F} = \bar{A}B + \bar{B}\bar{D}$$

$$F = (B+D)\cdot(A+\bar{B})$$

example → find as POS.

y

0	0	0	0
1	0	0	0
0	1	1	1
0	1	1	1

x w z

$\bar{w}x$ $\bar{w}y$ $\bar{w}z$ $w\bar{y}\bar{z}$

$$\overline{F} = w\bar{y}\bar{z} + \bar{w}y + \bar{w}z + \bar{w}\bar{x}$$

duality

$$\to F = (\bar{w}\cdot y\cdot z) + (w\cdot \bar{y}) + (w\cdot \bar{z}) + (w\cdot X)$$

SUBJECT: ....

Exercise slide 73

[1] $F(A,B,C,D) = AC\bar{D} + BC + ABC\bar{D} + A\bar{B}C$

كيف بدي اطلع جدول ال expression

A	B	C	D	F
0	0	0	0	
1	1	1	1	

احسب كل جواب.

بجيب الرسم ال k-map

وطلع ال simplified

طريقة رقم (2)

طلع لبعض طلات

C

0	0	0	0
0	0	1	1
0	0	1	1
0	0	1	1

BC

B

$ABC\bar{D}$

$AC\bar{D}$

A

D

$A\bar{B}C$

$F = BC + AC$

SUBJECT: / /

[2] $G(A,B,C) = \Sigma m(0,1,3,4,5,6,7)$ → شكل ال 8
فنحط لذفي سطر

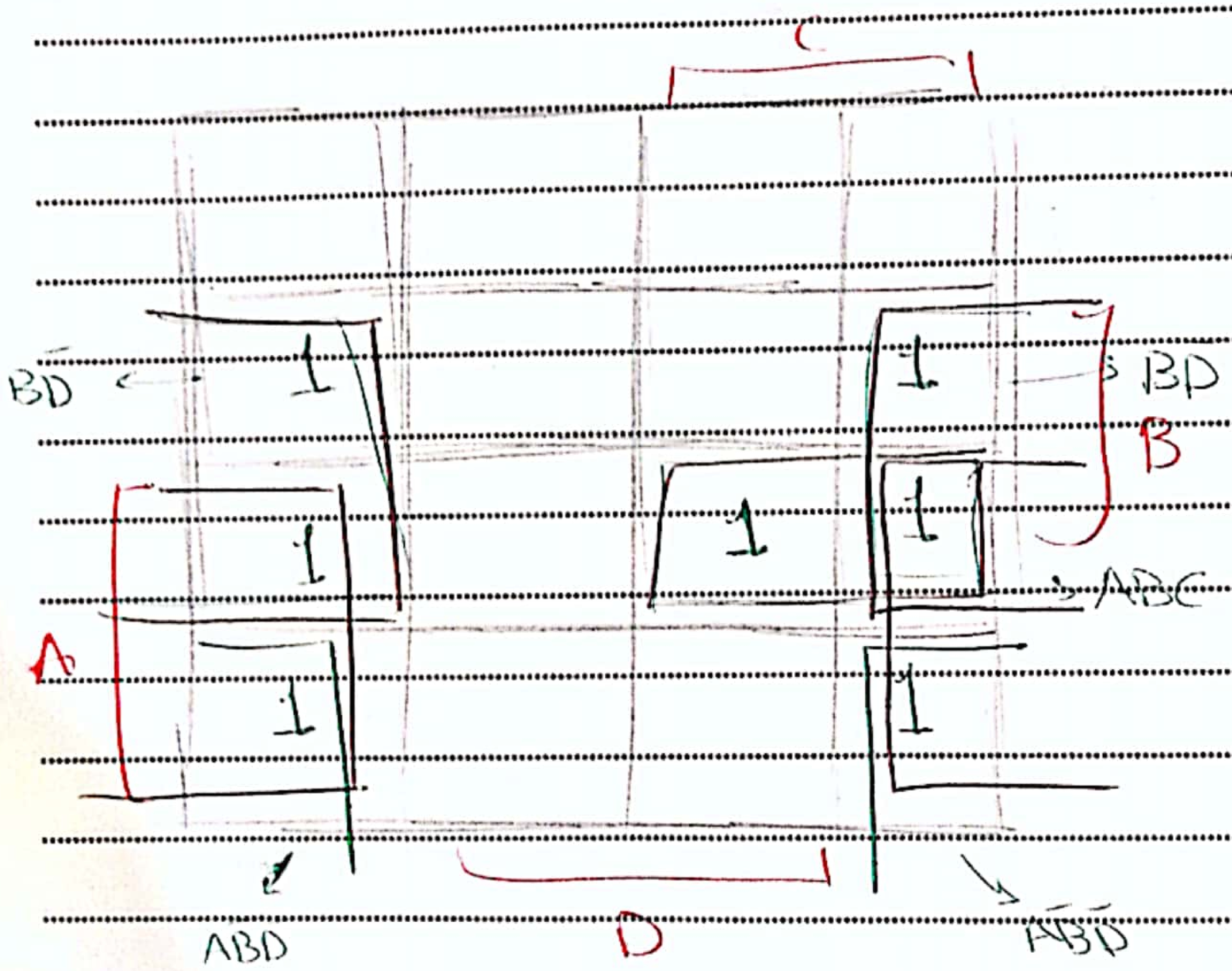
		B	
1	1	1	0
1	1	1	1

A marks the second row and C the two middle columns.

[3] $H(w,x,y,z) = \Pi_M(3,11,13,15)$ → maxterms
نحطش وين ما zeros

Example for an expression :-

$F(A,B,C,D) = A\bar{B}\bar{D} + B\bar{D} + ABC$



$F = B\bar{D} + A\bar{D} + ABC$

Example

$$F(A,B,C) = \Pi_M(1,5) + \Sigma_d(0,6,7)$$

	$\bar{C}$	C	C	$\bar{C}$
	X	0	1	1
A	1	0	X	X

B

$$F = B + \bar{C}$$

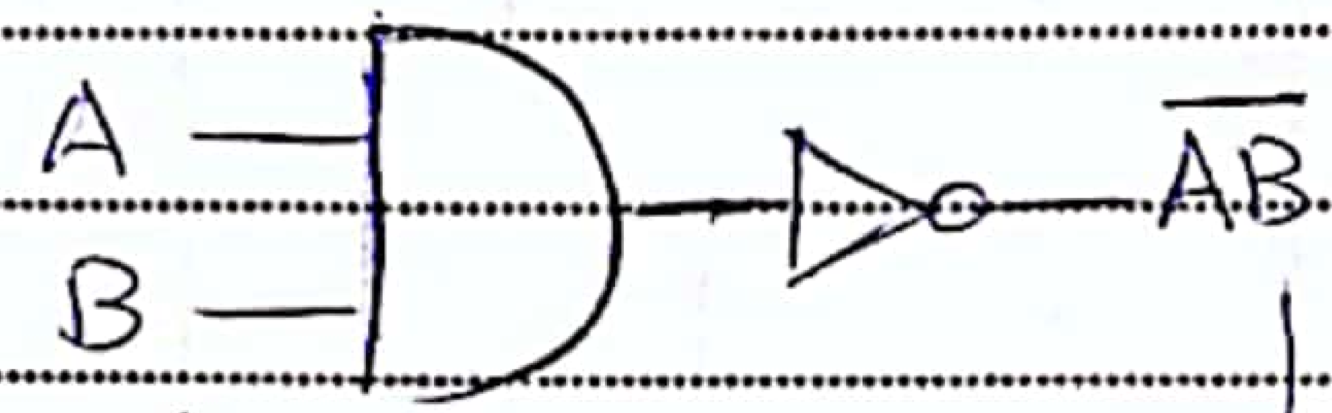
End of part 2

SUBJECT: .... / /

[Another types of Gates]

① Buffer ما بتعمل فرق رياضياً

② NAND



هي أصلاً عكس جو ال And

مختلفة كهربائياً عن الرسمة الثانية، وهيك هي الطريقة المناسبة للتصميم والأرخص والمباشرة.

هي الأساس لل CMOS Technology

ولها Mathematical equivalents برسمات كهربائية مختلفة

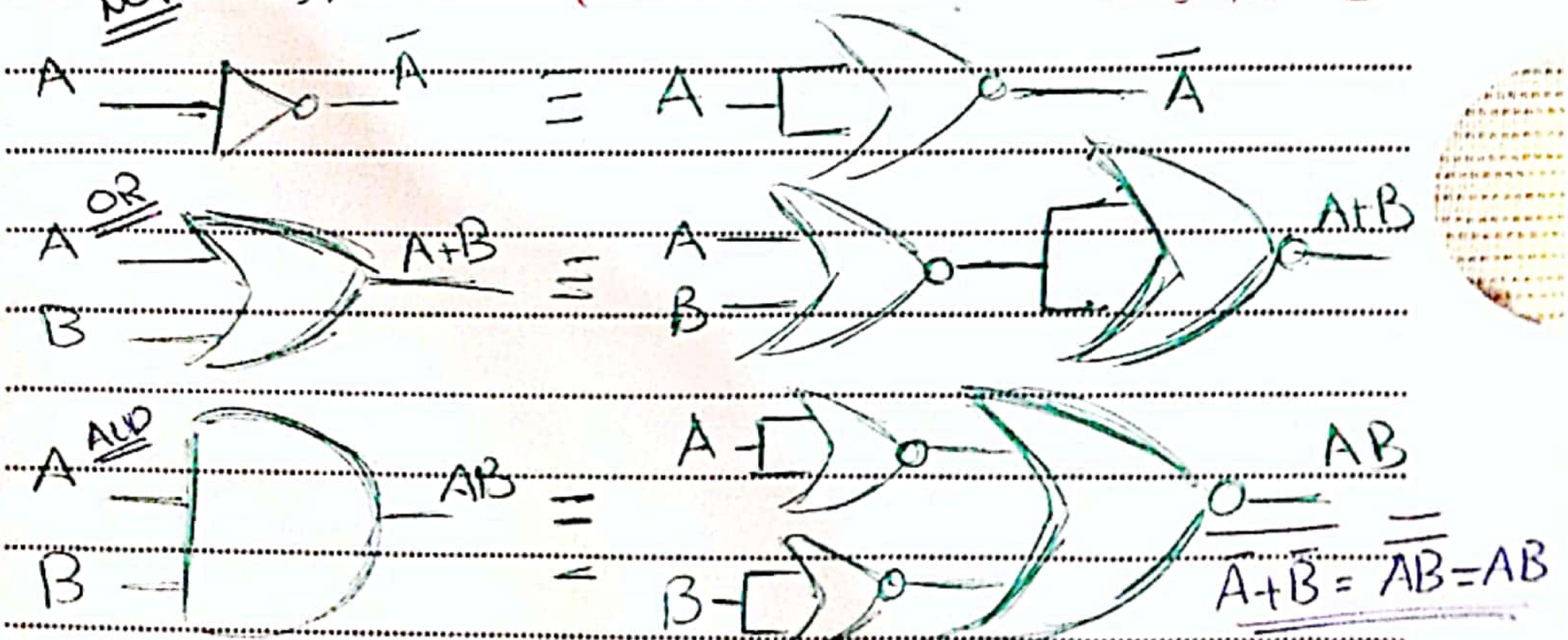
SUBJECT:

③ NOR

Universal Gates :-

الوصلات المقابلة ال OR وال And وال NOT
باستخدام ال NAUD موجودين في سلايد 78

X —[← هاي لأنو وحدناها
NAUD المدخلين ال input دخلو مرتين
ونفس الحكي ال NOR (← universal) بس شي مهين زي ال



SUBJECT: ....

4) XOR

لو بدنا نحكي عن ال XOR رياضيًا.

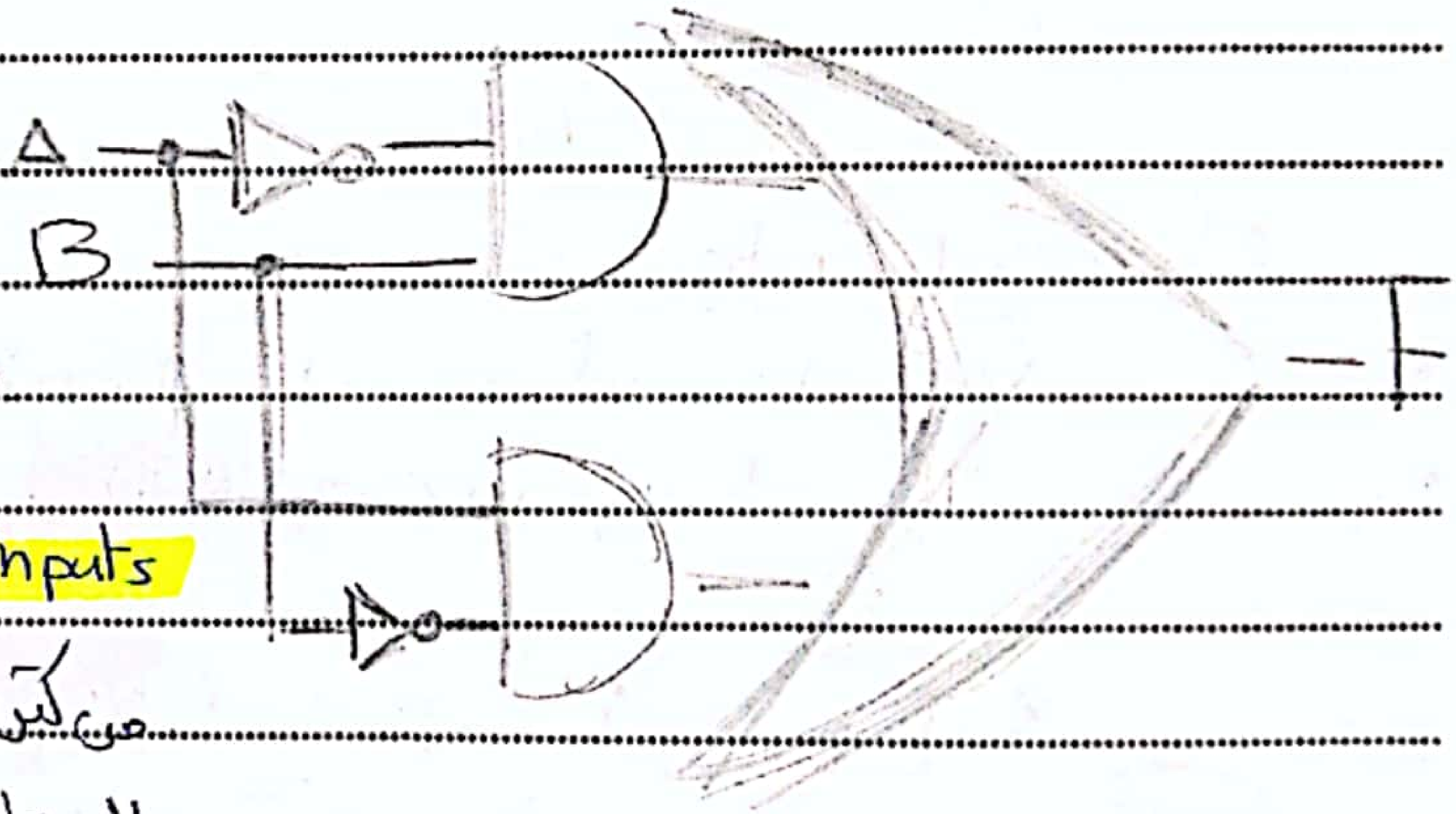
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

$F = \bar{A}B + A\bar{B}$ (SOM)

if either is 1 but not both

Diagram

for 2 inputs



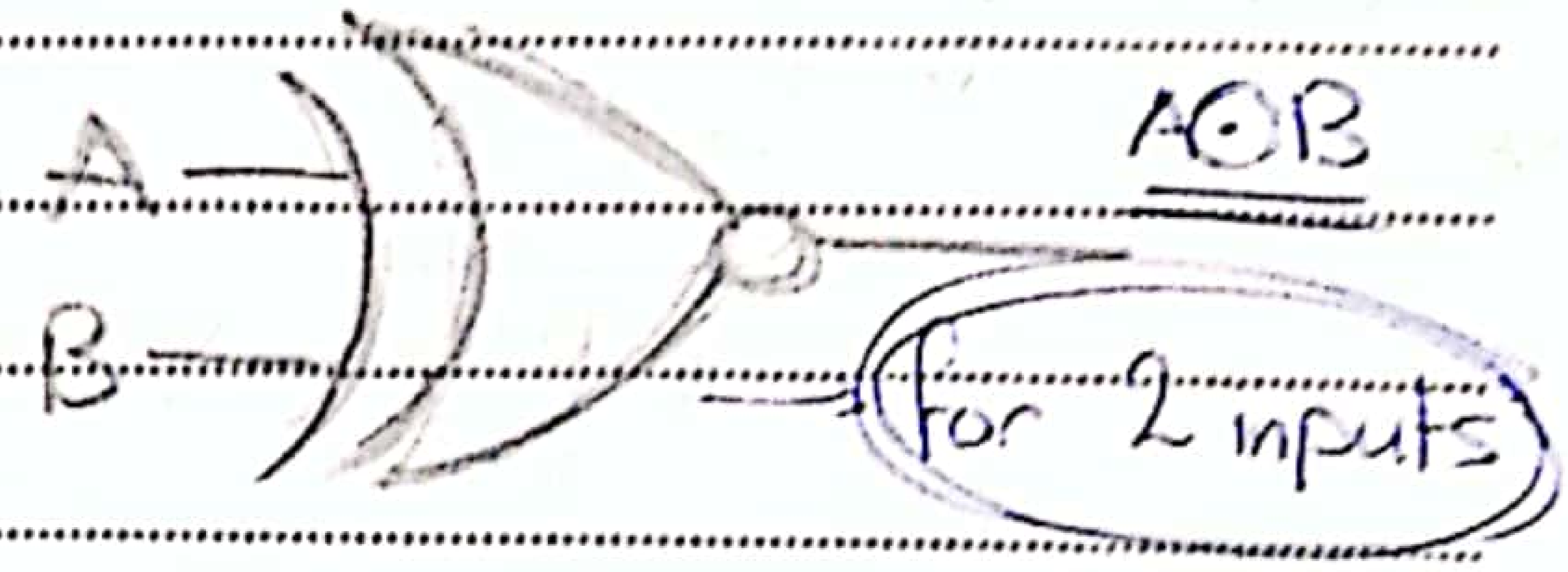
من كتر ما هي مهمة اخذوا ال block تبعها وعملوها وحدة لوحدة

SUBJECT: / /

[5] XNOR it gives true, if the both of inputs are equal

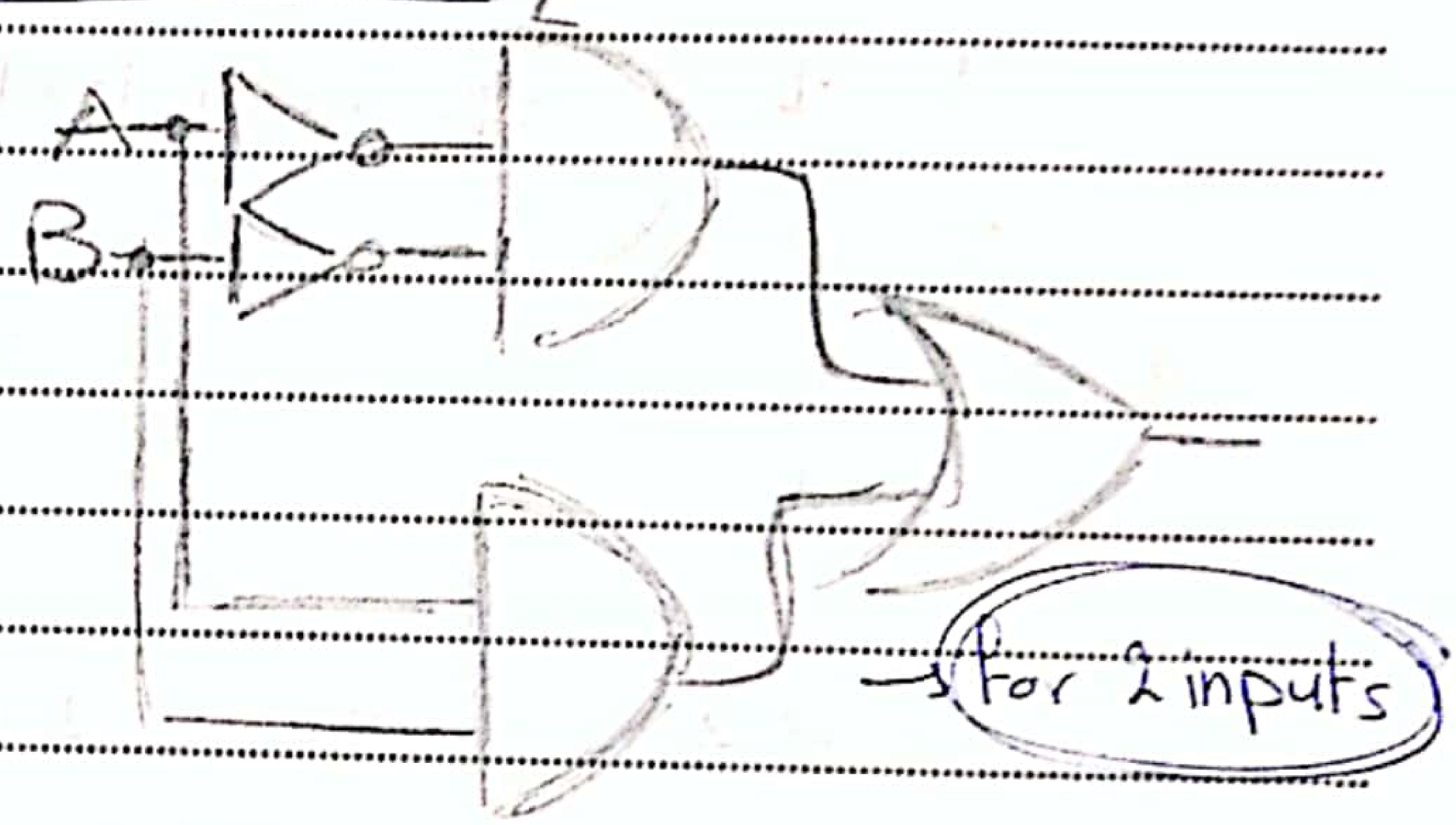
it is called [Equivalence fuction]

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1



(SOM) → $F = \bar{A}\bar{B} + A B$

رسمة تانية حسب المعادلة



نعتمد بالذات الـ SOP ← implementation بالذات الـ truth table
وعلى الطريقة من عملية تعتمد (SOM) على الـ minterms
للـ XOR و الـ XNOR

[redacted]

سلايد 85 ديجتل لشابتر 3

S T A R S NOTEBOOK

ال Identities ال XOR و ال XNOR دولا انا بارديم
دليل truth table

اذا جربت فيهم على ال Barbdo، يقلب بال truth table لل XNOR

مثلاً ←
$$\boxed{x \oplus \bar{y}} = \overline{x \oplus y} = x \odot y = \bar{x} \oplus y$$

x	y	F
0	$\cancel{0}$ 1	1
0	$\cancel{1}$ 0	0
1	$\cancel{0}$ 1	0
1	$\cancel{1}$ 0	1

⟹ Truth table of XNOR

مهم

$(x \oplus y) \oplus z$ ← بحسب كل 2 مع بعض في بال truth table

x	y	z	
0	0	0 → 0	0
0	0	0 → 1	1
1	0	1 → 0	1
1	0	1 → 1	0
1	1	0 → 0	1
1	1	0 → 1	0
1	1	1 → 0	0
1	1	1 → 1	1

Using pairs

وكمان لو كانوا 4 زي ←
$$F = (((A \oplus B) \oplus C) \oplus D)$$
← بحسب A و B سوا
← ناتج A و B بحسبه مع C
← ناتج ال 3 بحسبه مع D

و نفس الكلام لل XNOR

قاعدة مهمة !! ال XOR مالها علاقة بال XNOR

اذا بعدّي ال 2 variable

$x \oplus y \oplus z \neq$ عكس ال $x \odot y \odot z$

العلاقة الوحيدة الصحيحة هي :-

$x \odot y = \overline{x \oplus y}$

ملاحظة :

$F = A \oplus B \oplus C$

XOR is called "odd function"

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

The output is 1 when the number of ones in the input is odd.

وهاي صحيحة حتى لـ 4 او variables اكثر

وهي اقصر لمعرفة ال output

لما يكون عدد ال inputs كثير، بدل طريقة ال pairs

هي ال minterms ، بساعدني فيها (بس يكون واحد عليه Bar ع ال term)

$F = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$

SUBJECT:

وبناءً على ذلك، بقدر أكتب معادلة ال (SOM) من الدارة المنطقية بعمل ال truth table لل 4 متغيرات، لا يكون لأن بتحدد وجود ال Bar على كل term

مثلاً:

$F = w \oplus x \oplus y \oplus z$

$F = w\bar{x}\bar{y}\bar{z} + \bar{w}x\bar{y}\bar{z} + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}z + wxy\bar{z} + \bar{w}xyz + w\bar{x}yz + wx\bar{y}z$

4 حالات بـ 4 حالات

دائماً في ال XOR بطلع عدد ال 0 في ال output = عدد ال 1 عددهم متساوي

يعني لو كان عندي 16 input ← رح يكون عدد ال 1 ، 8 ، و 8 حالات بال odd terms

ومن هون جاءت فكرة حساب ال parity

كيف نجدها؟

بكون جوابها ← بطلع 1 ← XOR
اذا عندي odd parity مثلاً F ← output name

بعمل $\bar{F}$ ← عشان تطلع 1
اذا عندي zero parity

يعني لما أكتب ال (SOM) أكتبه لل zero terms (outputs)

Scanned with CamScanner

SUBJECT: / /

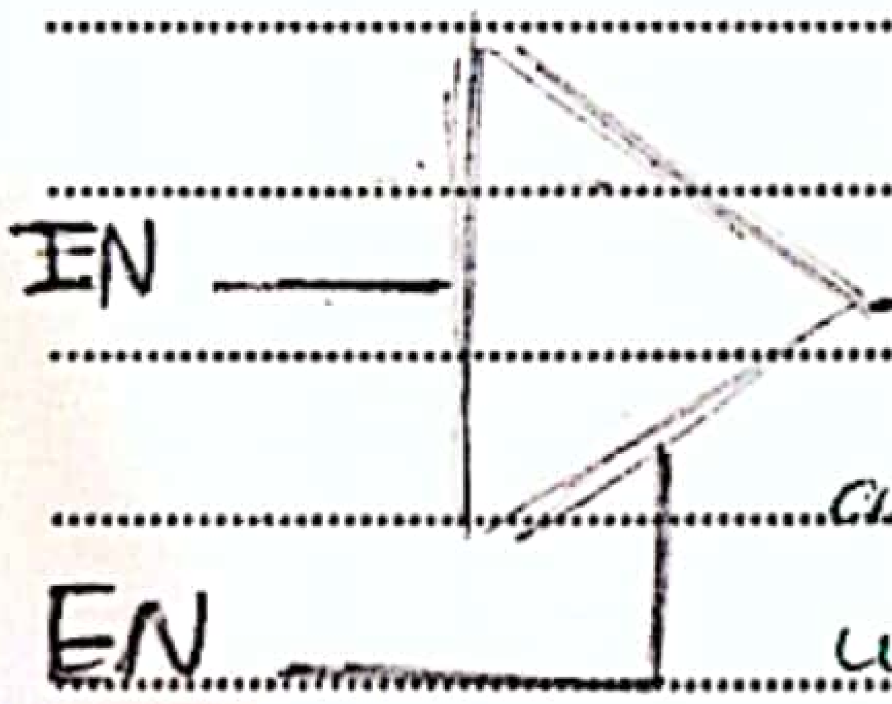
Tri-state Buffer

EN	IN	output
0	0	Hi-Z
0	1	Hi-Z
1	0	0
1	1	1

هاي ال circuit مهمة ورح نحكي عنه. اذا ال (enable = 0)، بتصير المقاومة بتقرب بدرجة عالية جداً (كأنها تقترب الى ما لا نهاية)

← بصير التيار = 0 ← بتصير open circuit
بفصل ال input وال output منفصلين/ عزل بينهم

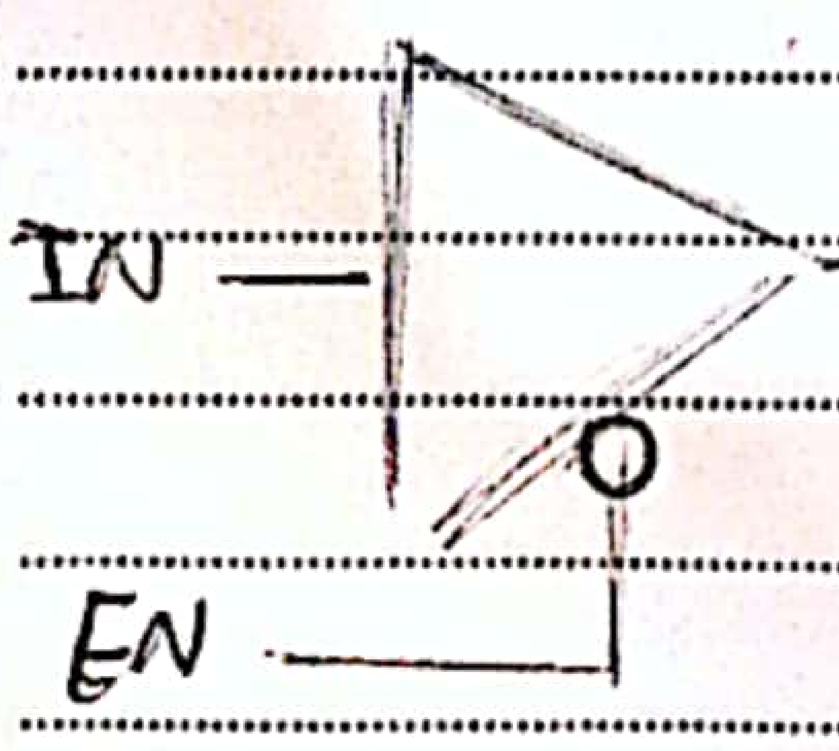
* لما يكون (EN=1) بصير يرجع Buffer عادي.



active high ← بشتغل ال EN ك
بكون (1) وبطفي لما (EN=0)

and the one with the (bubble) is on (EN) ←

سلايد 93
الرسمة الي ع اليمين رسم ← The (IN) is inverted.



active low

SUBJECT: / /

الأولوية هي لسيركت الـ EN ، حسب اذا خارجة او داخلة
بتشتغل ع ال (Input)

ما بجمع إلي أربط ، بنقلين كل واحد منهم بياخذ input ع يكف
زي ما أحنا قايلين ، باخد سالب 94 (بتصمم حالي)

لذلك لازم نضمن إنه التصميم يخلي وحدة من ال circuits
on ووحدة off ، ما بزبط يكونوا on مع بعض ،

Tri-state logic Circuit :

ال inverter اللي على ال S
بتأكد دايماً إنه واحد فيهم يكون off وواحد on

circuit by selection

	EN_1	EN_0	IN_1	IN_0	OL
S=0	0	1	X	0	0
	0	1	X	1	1
S=1	1	0	0	X	0
	1	0	1	X	1

بنقدر نستعمل ال Tri-state buffer عشان نطلع اي سيركت احنا بدنا

مثال:

$F = x \oplus y = \bar{x}y + x\bar{y}$

Tri-state buffers, inverters

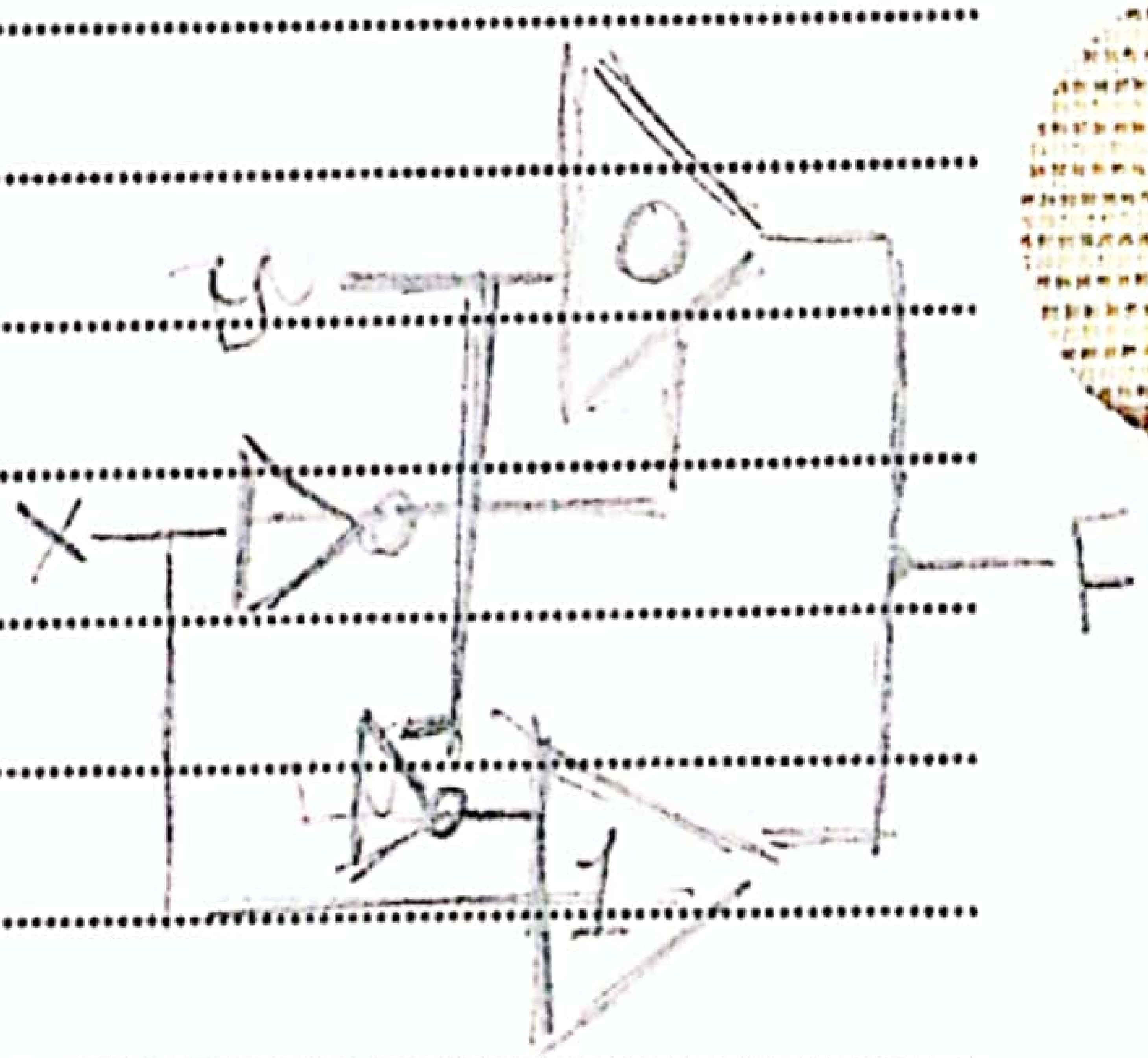
X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

$x = 0 \rightarrow y$

كل ما صار نفس الحروف

$x = 1 \rightarrow \bar{y}$

النفي اللي تحت



SUBJECT:

circuit by selecti

	EN_1	EN_0	IN_1		
s=0	0	1	X		
	0	1	X		
s=1	1	0	0	X	0
	1	0	1	X	1

ال buffer بيحط concept Tri-state
لل inputs، بتقرر وحدة فيهم
input والتانية بتلعب دور ال EN

بنقدر نستعمل ال Tri-state buffer عشان نطلع اي سيركت يلي بدنا

مثال :

$F = x \oplus y = \bar{x}y + x\bar{y}$

Tri-state buffers, inverters

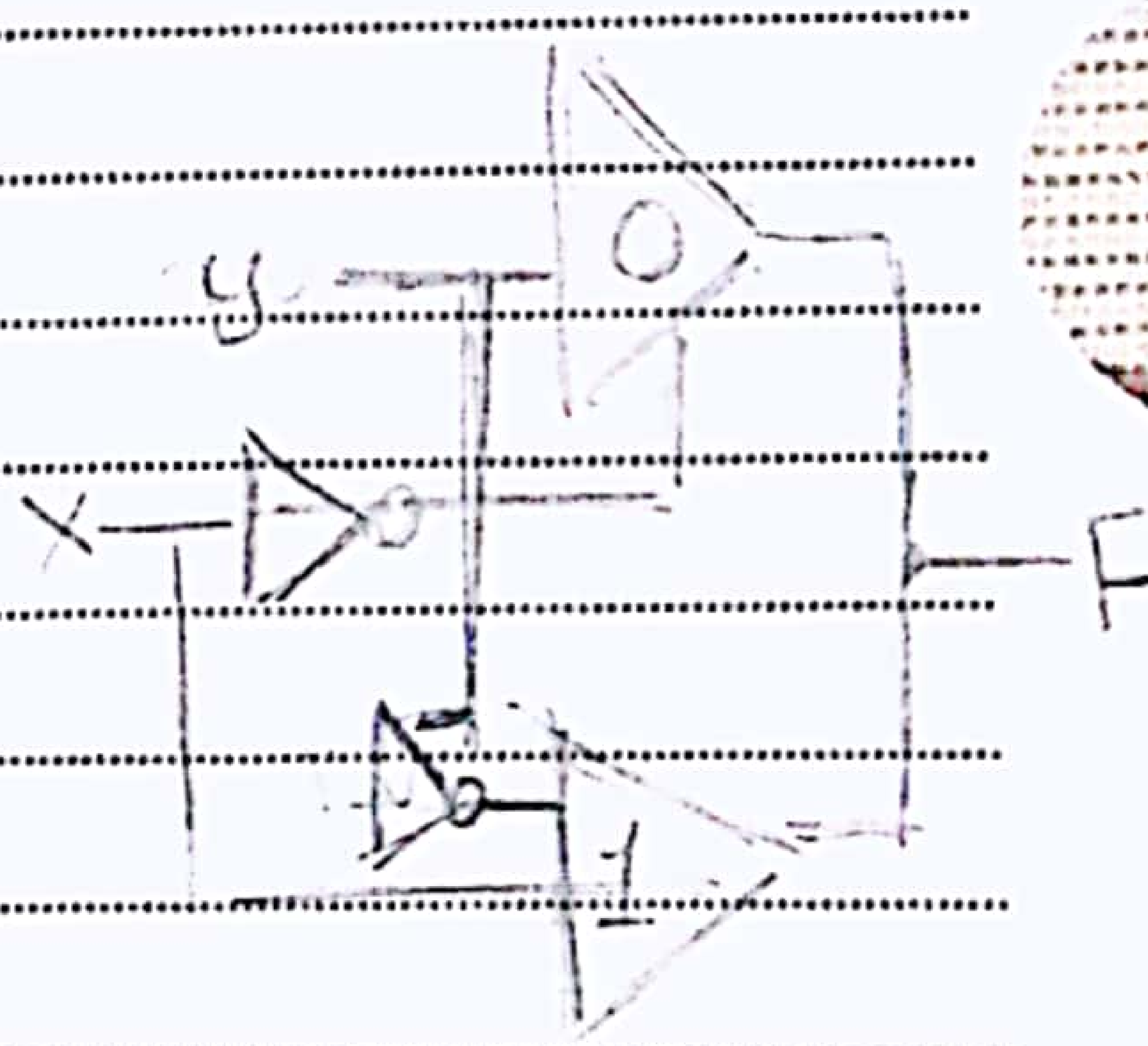
x	y	F
0	0	0
0	1	1
1	0	1
1	1	0

$x = 0 \rightarrow y$

[illegible]

$x = 1 \rightarrow \bar{y}$

العكس الي تحت



Exercise slide 98

X	y	Z	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

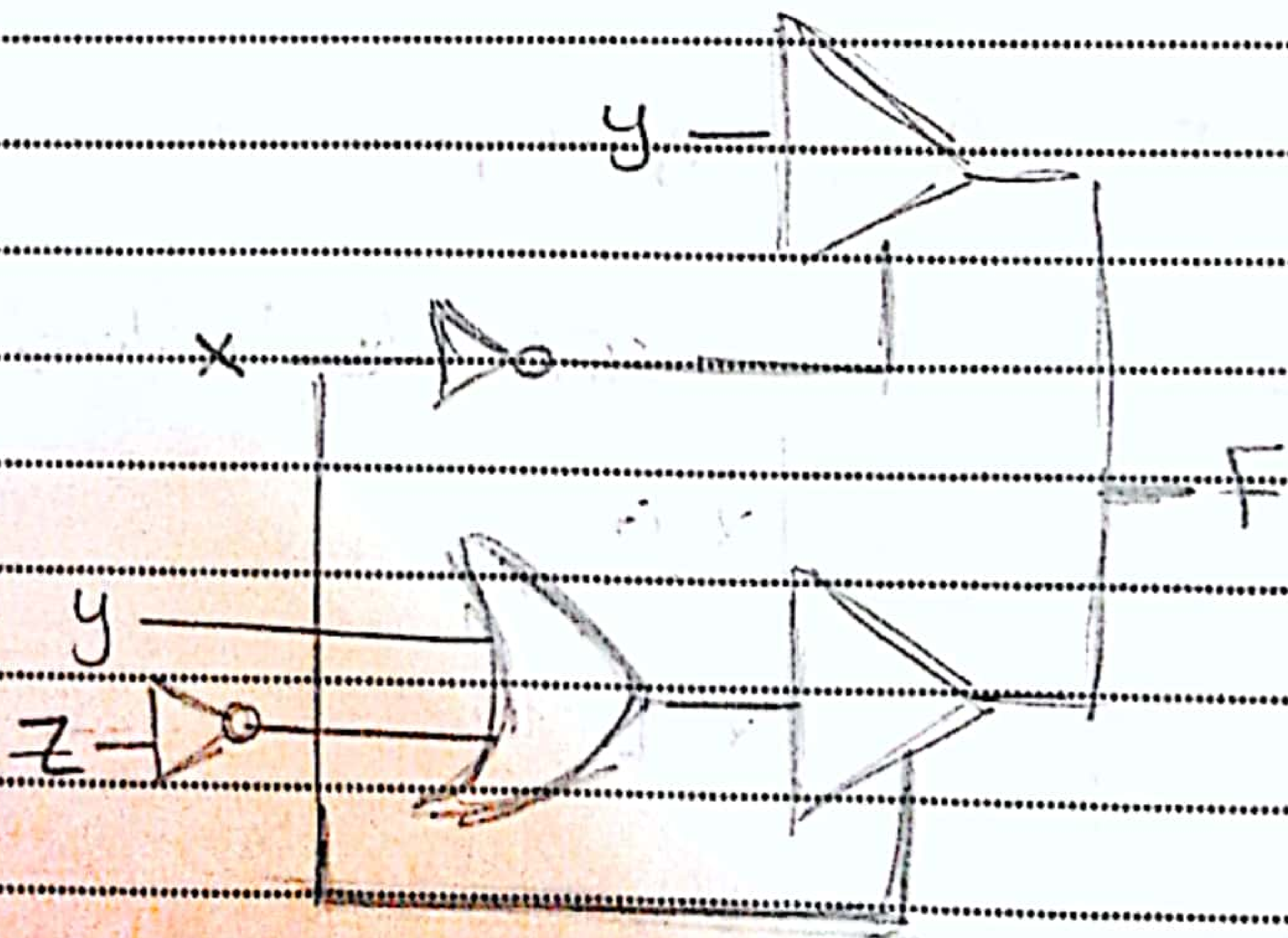
$x=0$ → $F = y$

$x=1$ → $F = \bar{y}\bar{z} + y\bar{z} + yz$

↳ by simplification

$F = y + \bar{z}$

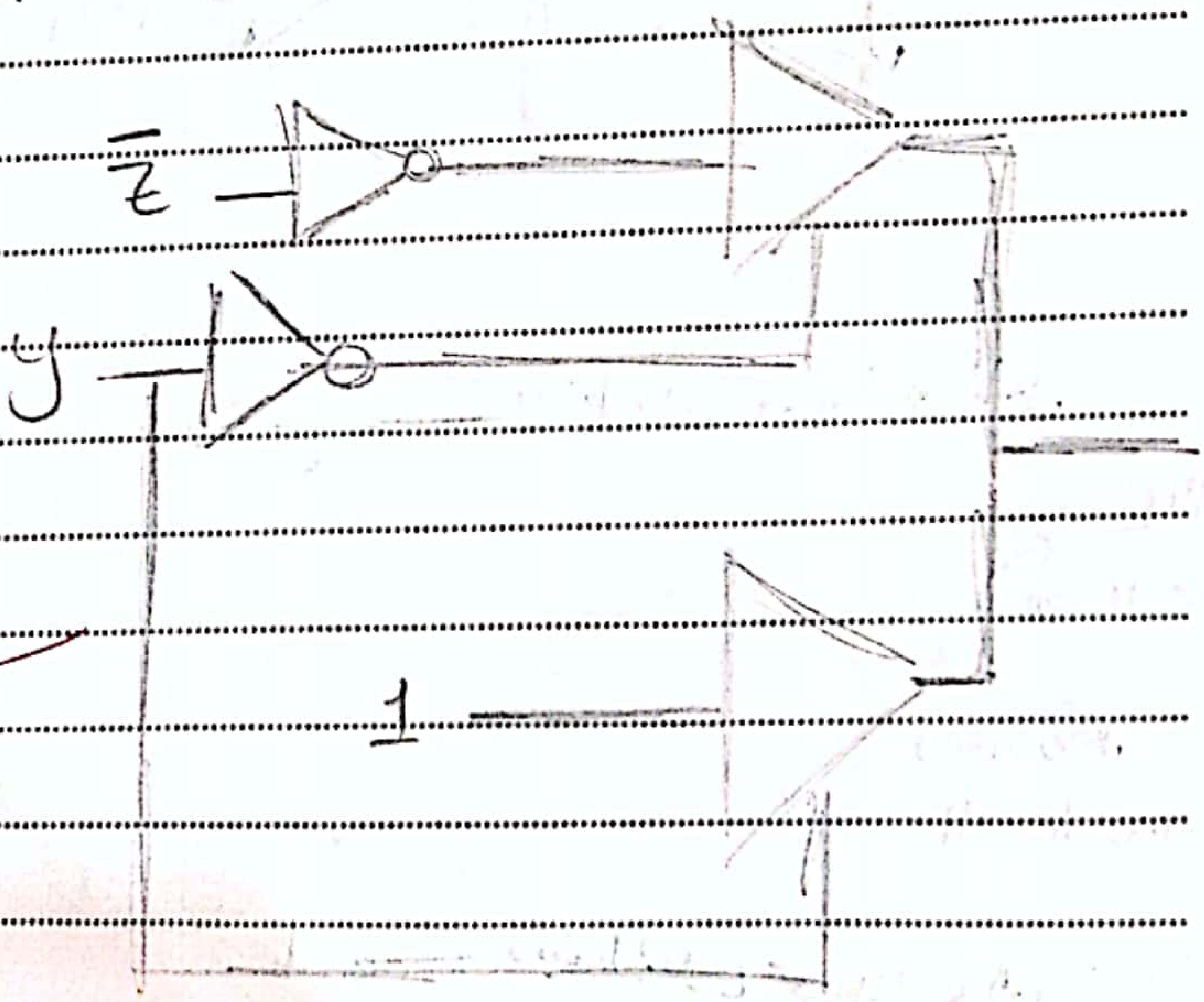
طالما مستخدم ال Tri-state buffers و inverters و [illegible]
↳ or



SUBJECT:

لو مش محتاج بال OR ، بعتبر ال F اللي بالجزء اللي تحت
Truth table منفصلة وبحطهم ا لسيركت نتيجة

	y	Z	F	
	0	0	1	$\bar{Z}$
$y=0$	0	1	0	
	1	0	1	1
$y=1$	1	1	1	



بدخلهم هاي محل ال OR هناك

example

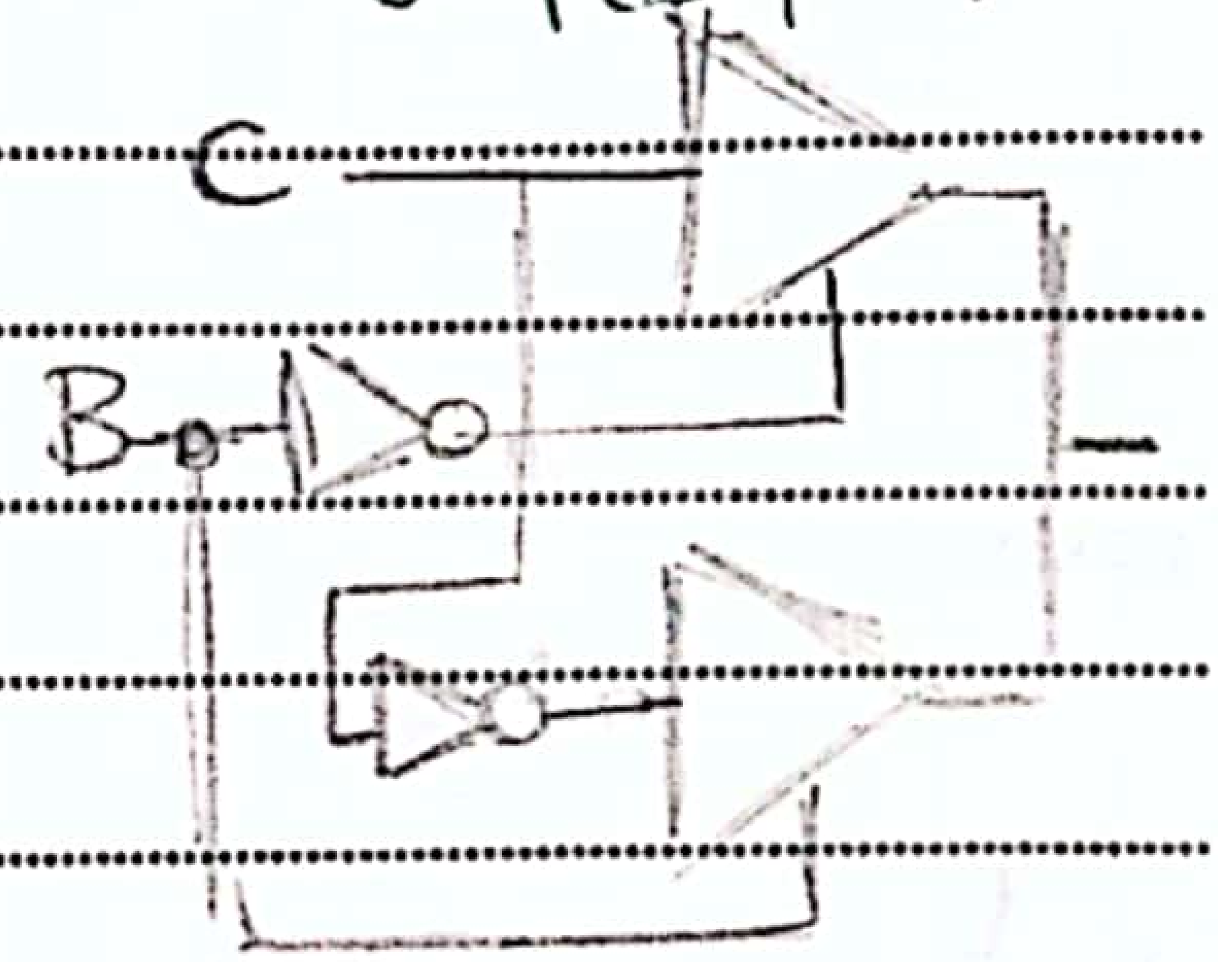
أعقد سؤال ممكن يجي على هالموضوع:

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

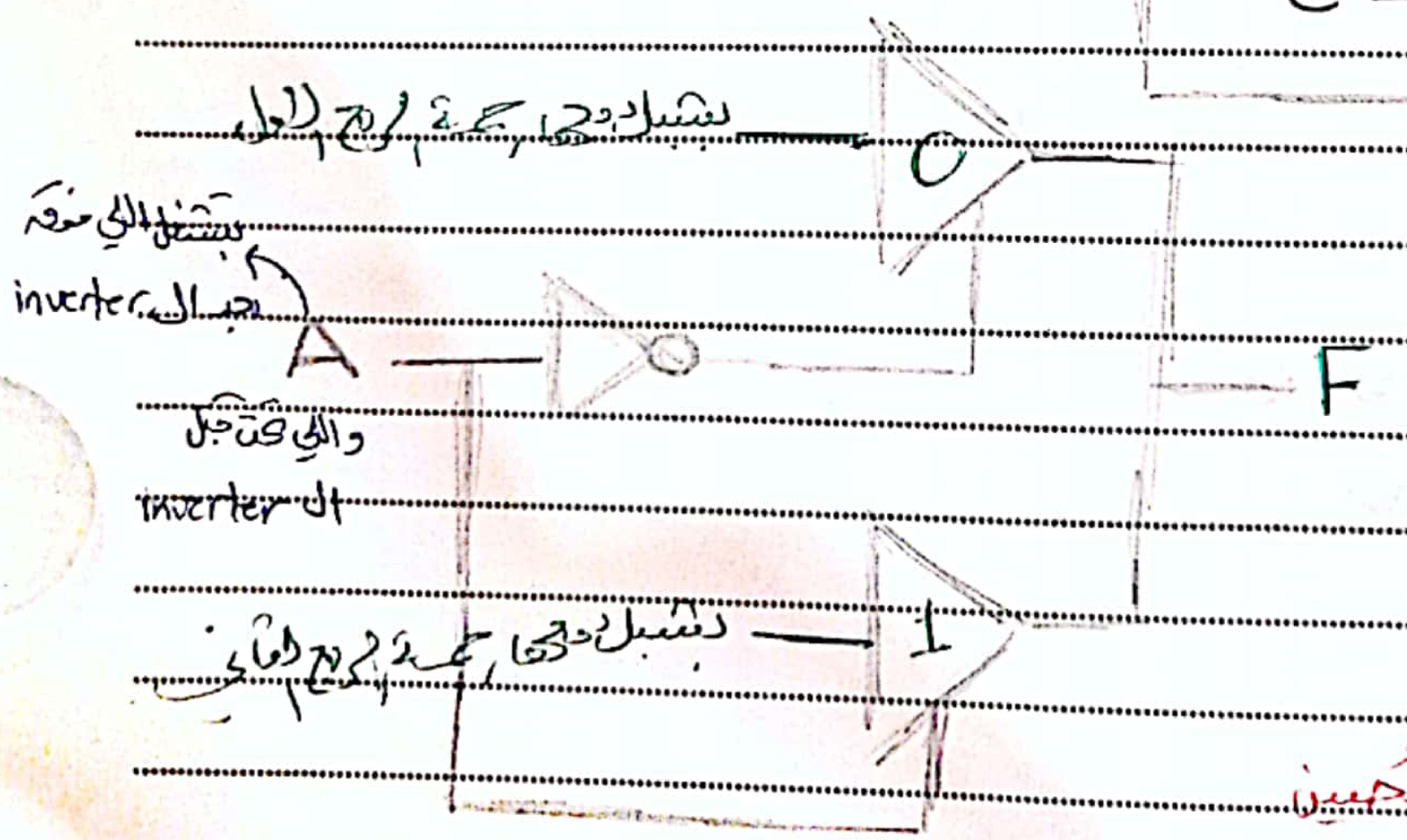
A=0 (B=0, B=1)

A=1 (B=0, B=1)

مخرج الربع الأول



مخرج الربع الثاني



قسمنا ال A لنصين
وال B لربعين
وصار ال C هو ال input لحاله

SUBJECT: / /

Exercise slide 99 ← هذا Truth table

هو احسبه ال:

A	B	C	F	
0	0	0	0	
0	0	1	0	ما دام الربع بشتغل ← السيركت الي تحت
0	1	0	0	
0	1	1	1	← ما دام يشتغل الي فوق
1	0	0	0	
1	0	1	1	← الربع الي تحت
1	1	0	0	
1	1	1	0	← الربع الي فوق

B=0 B=1 B=0 B=1

النص السفلي بالربع

النص العلوي

ال Bubble هاي ما تشتغل بكون شغلها بس مثل ال inverter

هي بتشتغل ال circuit عندما ال 0=EN

$$F(A,B,C) = \bar{A}BC + A\bar{B}C$$

End of chapter 2
part 3

SUBJECT: / /

chapter 3 : combinational logic design

آخر خطوة دائماً في أي تصميم هي ال verification
مهمة جداً لأنه إذا لم يشتغل تصميمك صح ← هاي الخطوة بتعمل
في ال Lab

(Design Example 1)

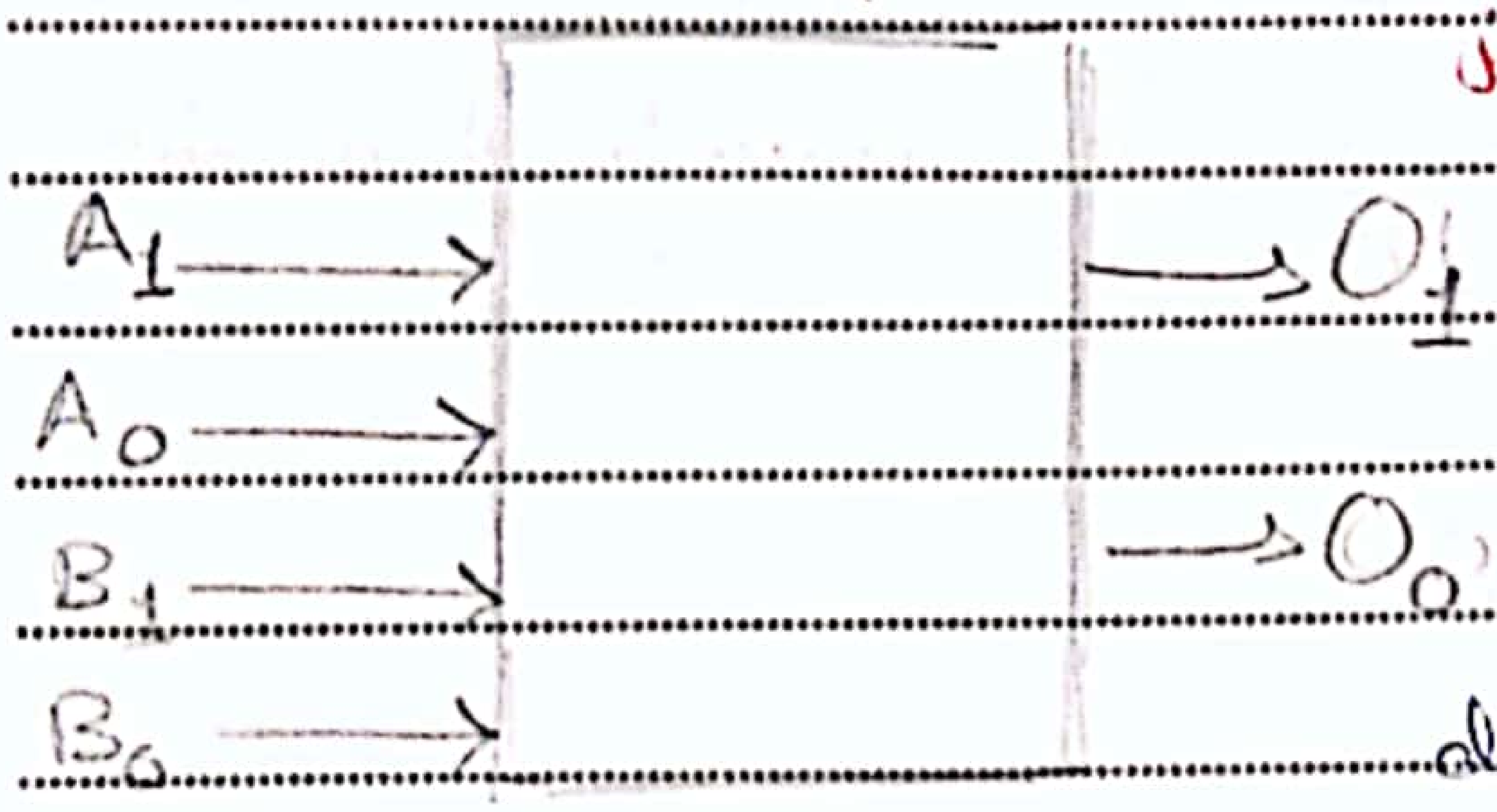
x	y	z	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

0	1	3	2
0	0	1	0
4	5	7	6
0	1	1	1

SUBJECT: .... / /

Design Example 2

دول conceptually هما عدد
رقمين Binary ، بس بالادر داخل
على ال circuit اربع لال



بجمع ال even وال odd
حيث انه تحقق رقم ال decimal
هو ال even او ال odd

4 inputs ✓
2 outputs ✓

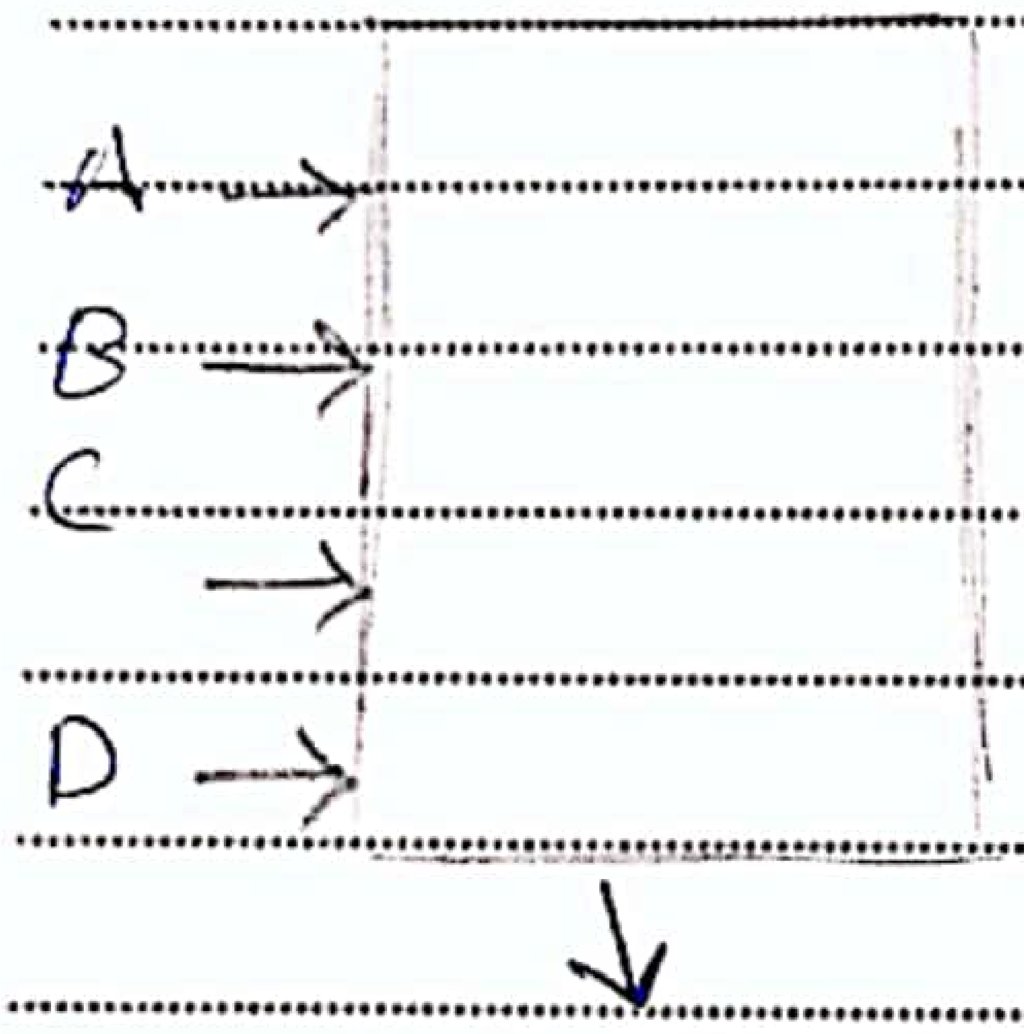
A_1	A_0	B_1	B_0	
0	0	0	0	0 0
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	1 1

نازل على ال 7

SUBJECT: .... / /

2 outputs → 2 k-maps

Design example 3



⇒ نعمل 4 k-map

واحدة ل x

y " "

z " "

w " "

ال BCD → Excess 3

عشري كلها ↳ is BCD+3

من 0 → 9

يعني 0000

↓

1001

SUBJECT: .... / /

لما يطلب مني رسمة NAND و NOR بشيل ال AND و
ال OR وبس بحط ال inverter عادي

ال inverters بس يجوا ورا بعض بلغوا بعض

بشرة اسهل رسمات ال NAND :
← كل And بحط عليها NAND وراها inverter
← " or " " " " وبعكس كل ال ~~xnxp~~ inputs
← وزع ال inverters اذا بموضع ما فيه غايبة
← الغي ال inverters اللي ورا بعض

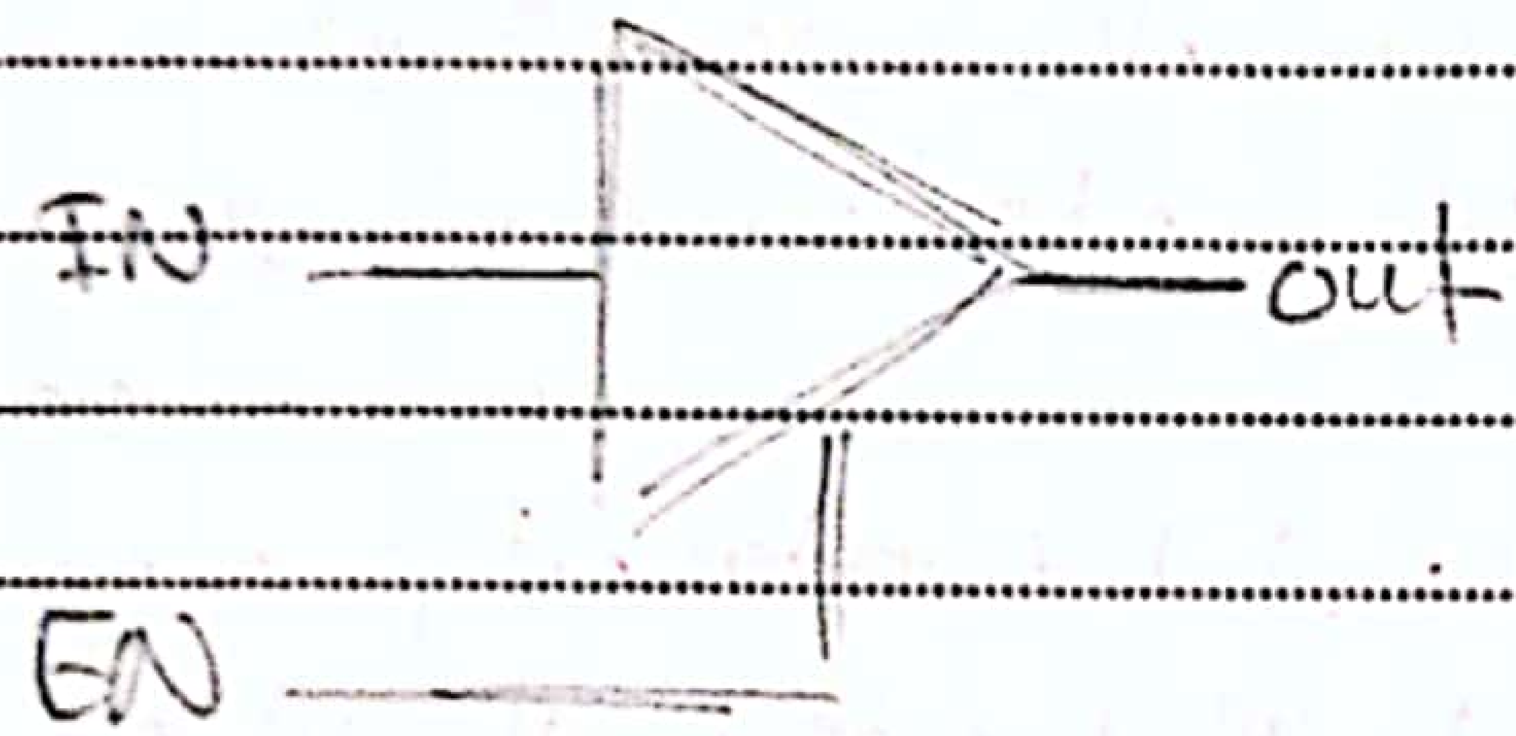
S T A R S N O T E B O O K

Scanned with CamScanner

SUBJECT: / /

Enabling Function

To show the difference



EN	IN	out
0	x	Hi-Z
1	0	0
1	1	1

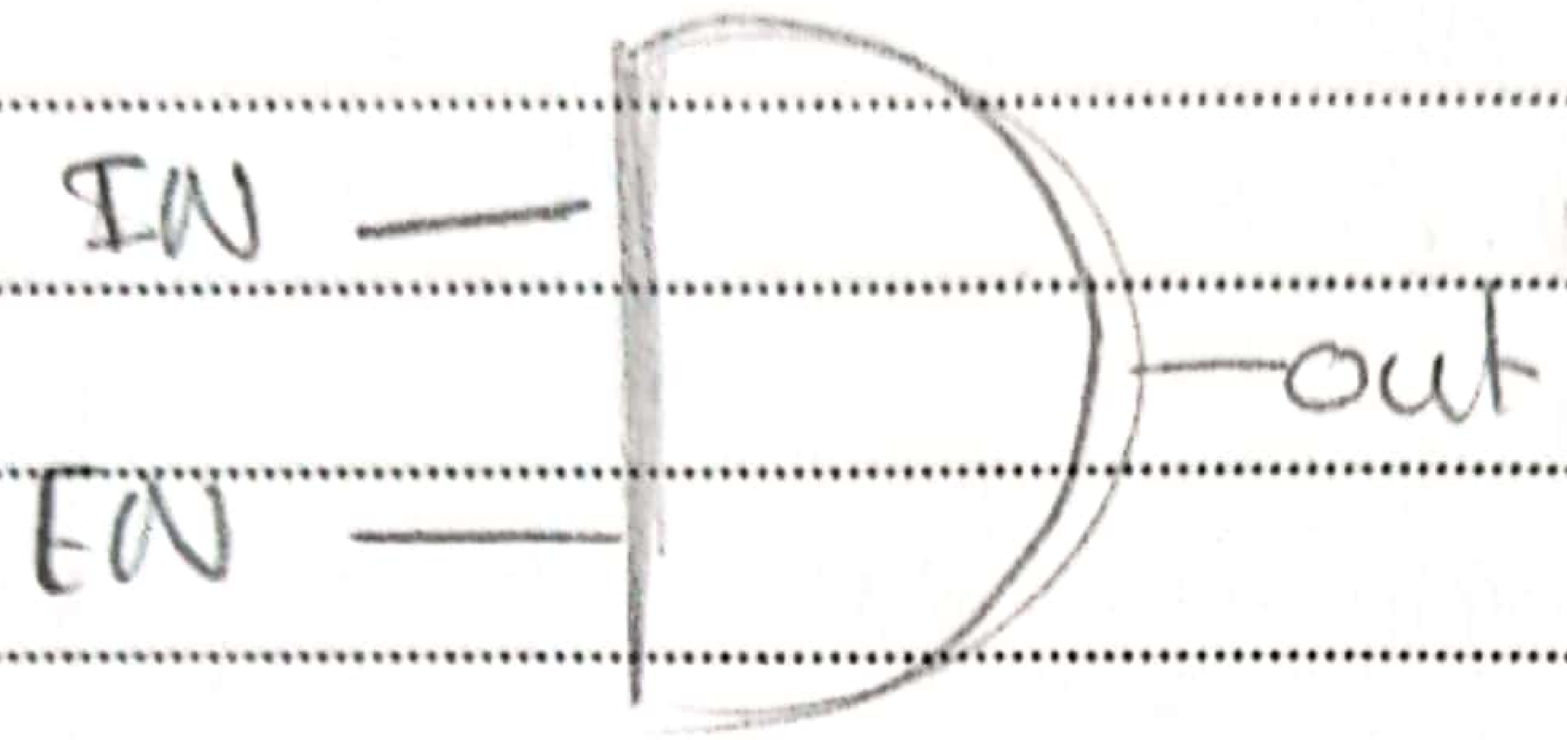
off ← بس لو EN = 0 لا

output = input when EN = 1

Scanned with CamScanner

But

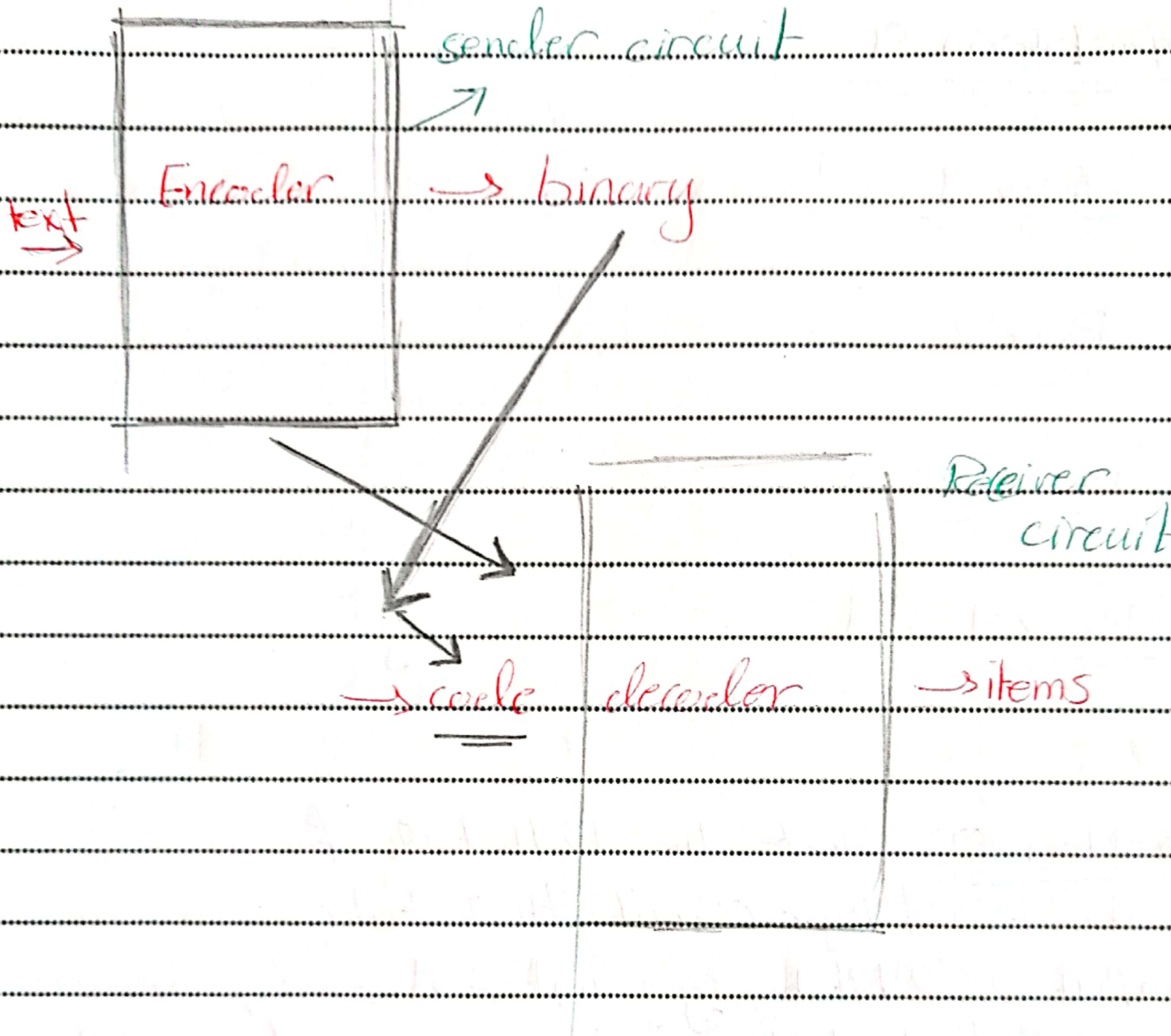
مشكلتي بس اثبت ال output ع zero دايم
ع كل الحالات حتى ، احسن من ال buffer



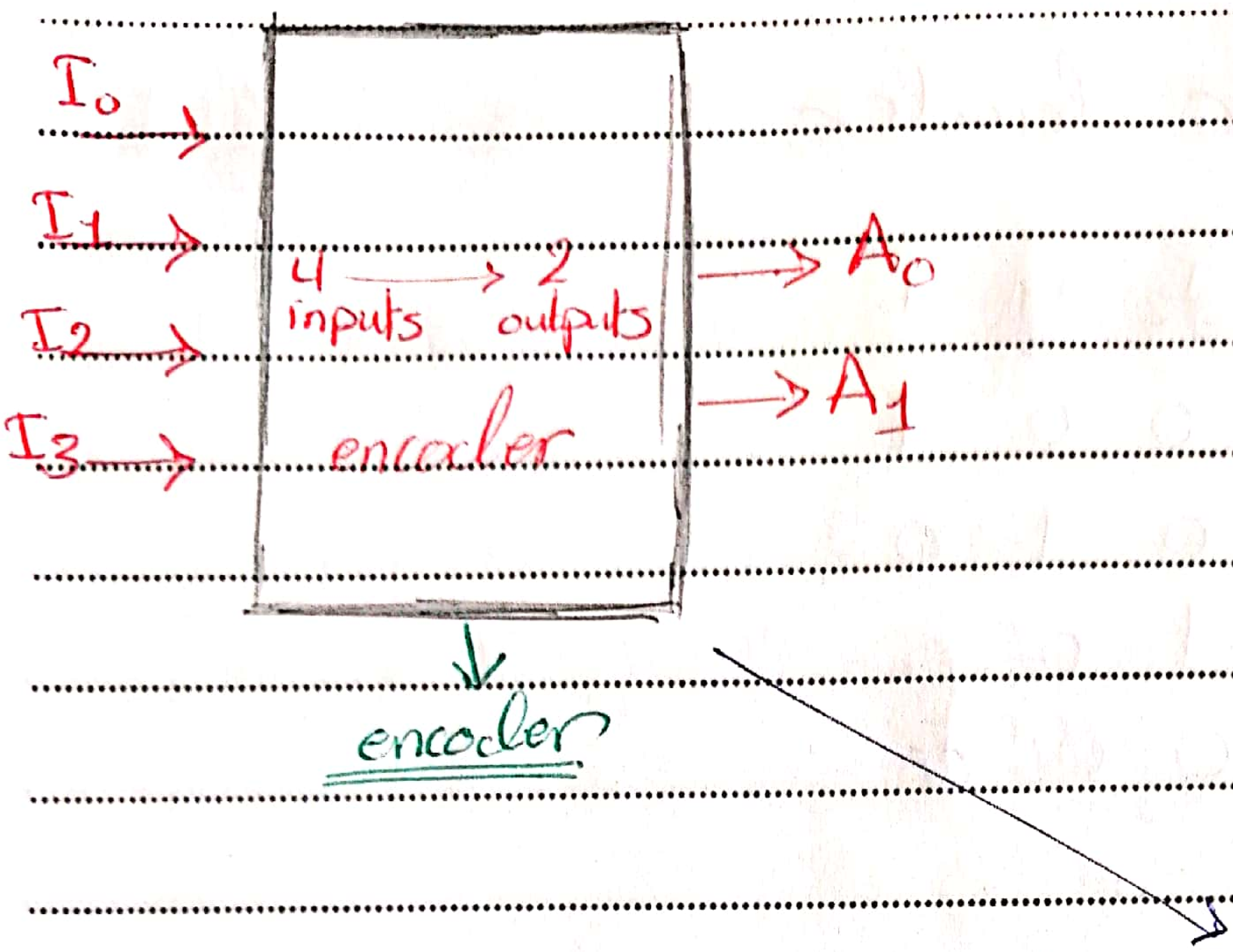
EN	IN	out
0	X	0
1	0	0
1	1	1

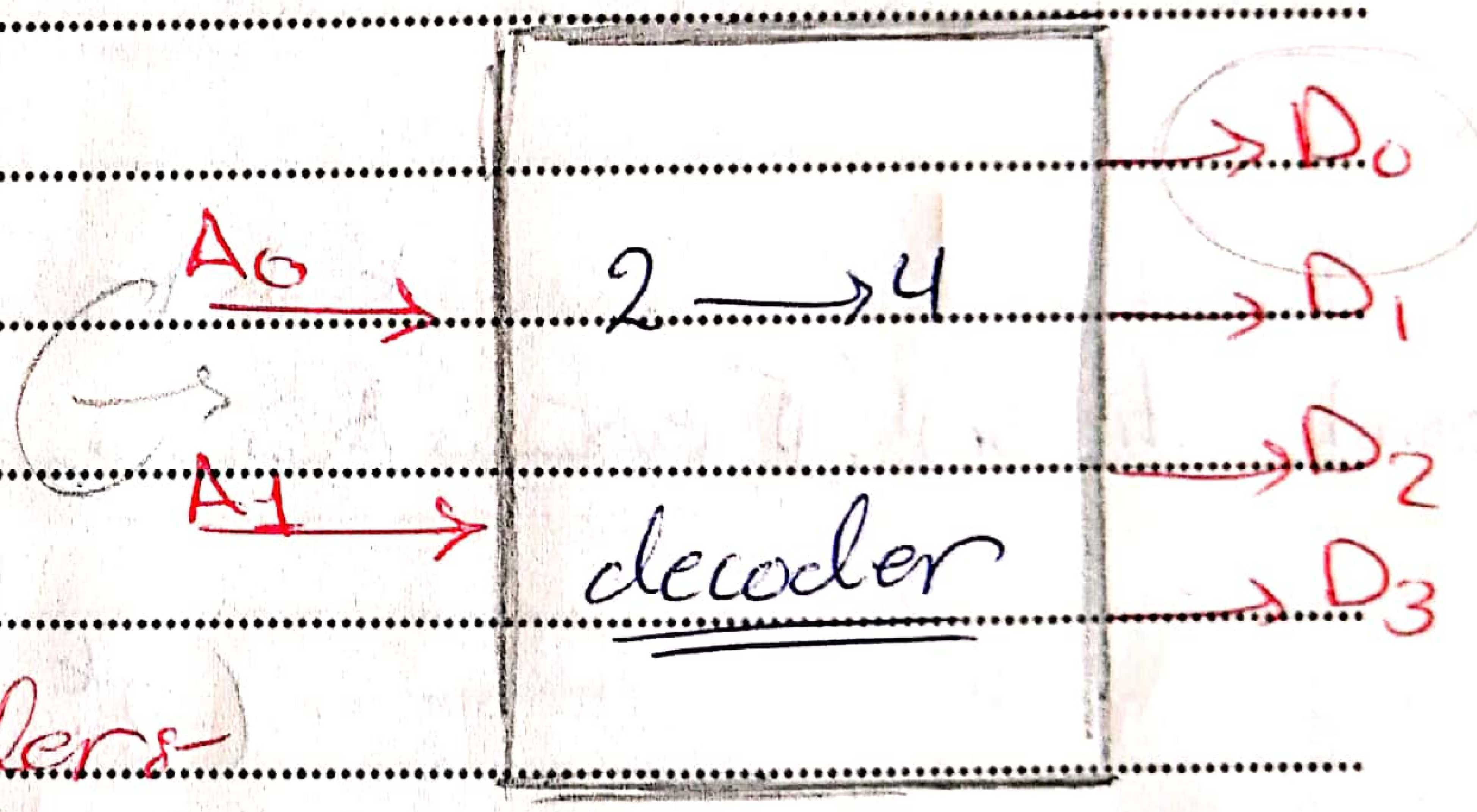
هاد هو لغز بين ال
Buffer العادي
دال And
مع ال EN

لما EN=0 ، المشكلة ما بتصير بس ال
output فيها ثبتت على ال 0 دائماً



Suppose you have 4 colors (items)
4 combinations → you need 2 Bits.
Black → 00
white → 01
silver → 10
Gold → 11





truth table for encoders

I_3	I_2	I_1	I_0	A_1	A_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

عادةً بيجي واحد ←
من ال items
بس

ال user ما بيختار
ثنين مع بعض ، عشان نقدر
نعمل ال encoding لـ عنوانهم

truth table for decoder

A_1	A_0	I_3	I_2	I_1	I_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

اذا حجم ال code ← k-bits

العدد ال input = 2^k ومش شرط بيستخدم كل ال combinations

فال encoder ($2^k \rightarrow k$)

بيدخله input بعدد 2^k وبطلع منه outputs بعدد k

والـ decoder ($k \rightarrow 2^k$)

بيدخل عليه input بعدد k، وبطلع منه outputs بعدد 2^k

Decoder

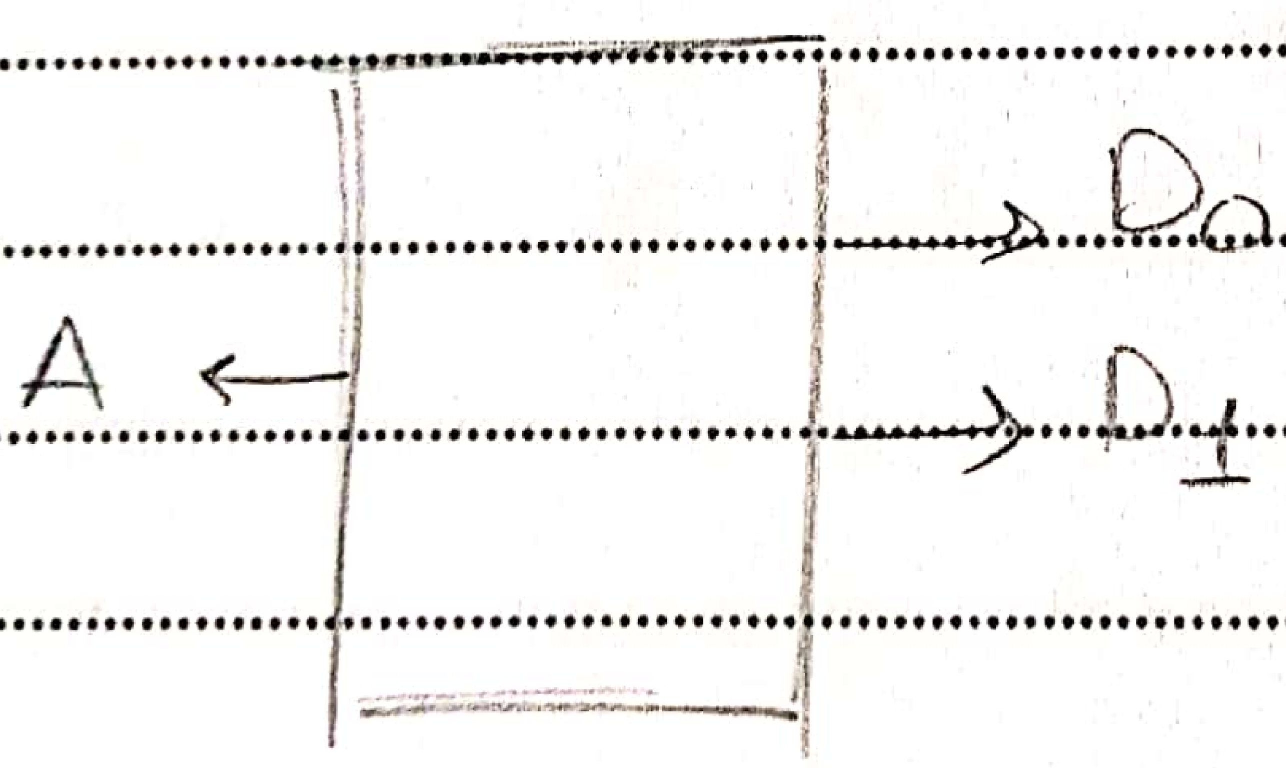
بنوت عليه code بنطلع منه items

Specification of the encoder

واحد من ال bits في ال output لازم يكون (1)، والباقي
بكونوا zeros، ممنوع يطلع أكثر من 1 item بنفس الوقت

الهدف
تصميم [illegible] decoder

1-2-Line decoder (1 input bit 2 bits output)



A	D_1	D_0
0	0	1
1	1	0

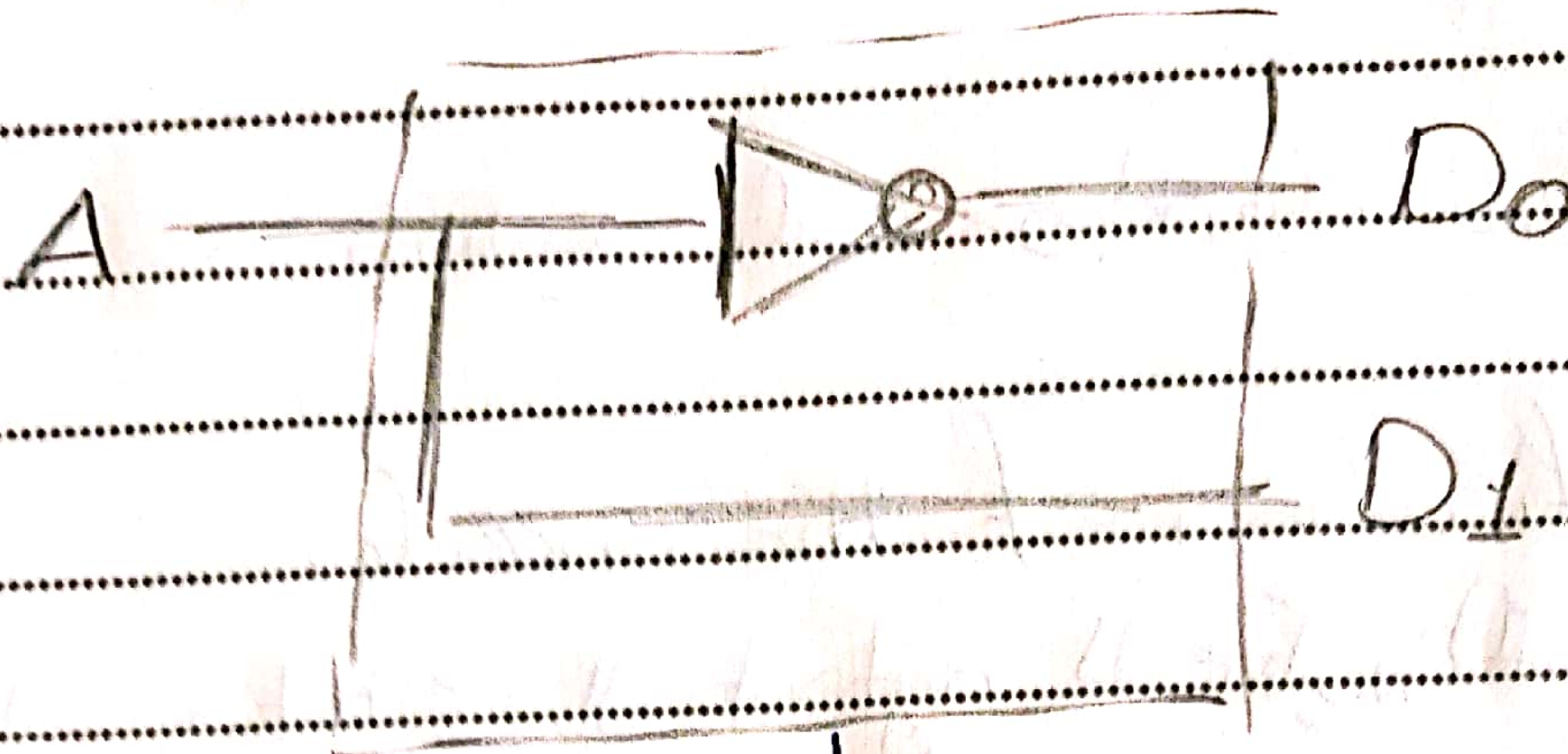
هو [illegible] ال 1 بالأول

functions of outputs seperately:

$D_0 = \overline{A}$

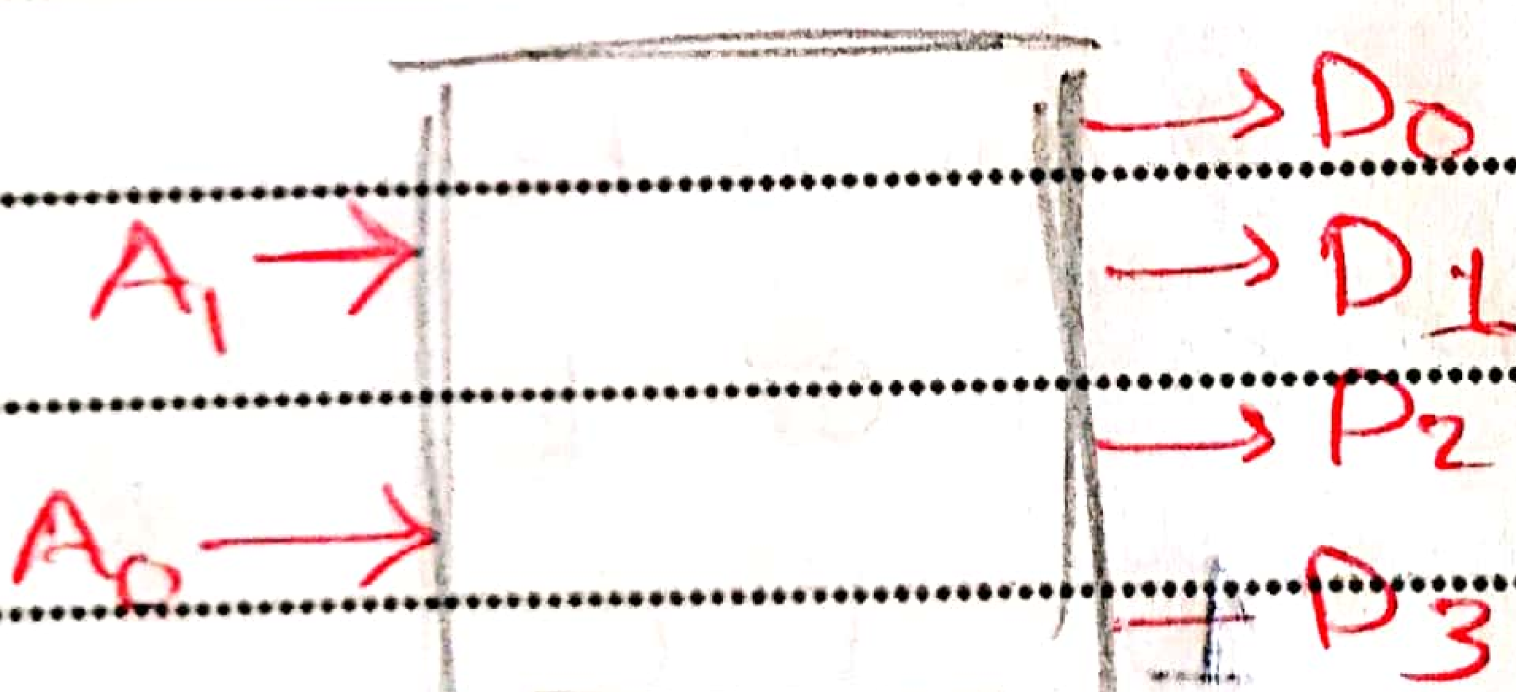
$D_1 = A$

الرسمة :



هاي هي الرسمة الداخلية

2-4- decoder



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

SUBJECT: .... / /

Functions

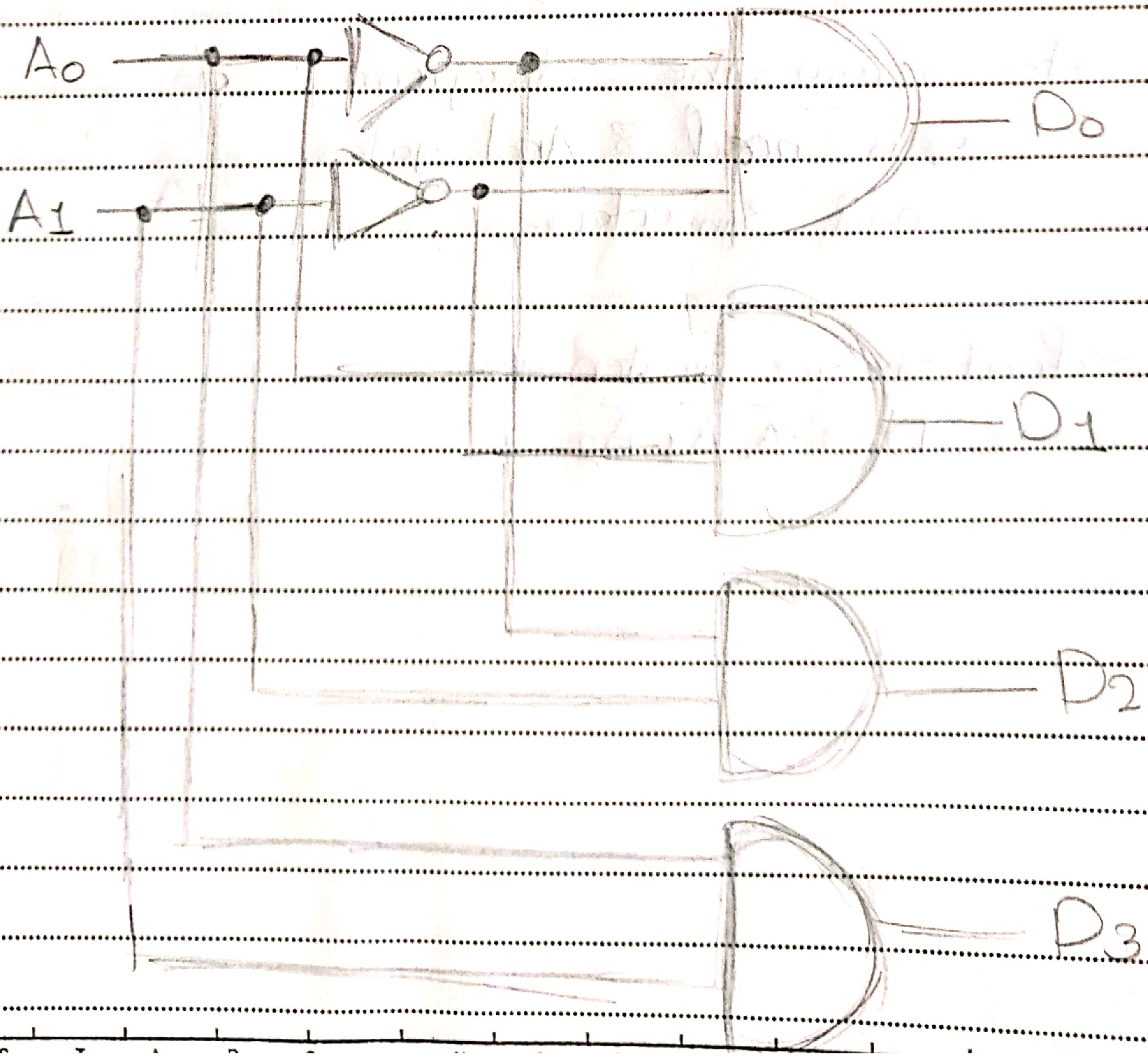
$D_0 = \bar{A}_1 \bar{A}_0$

$D_1 = A_0 \bar{A}_1$

$D_2 = \bar{A}_0 A_1$

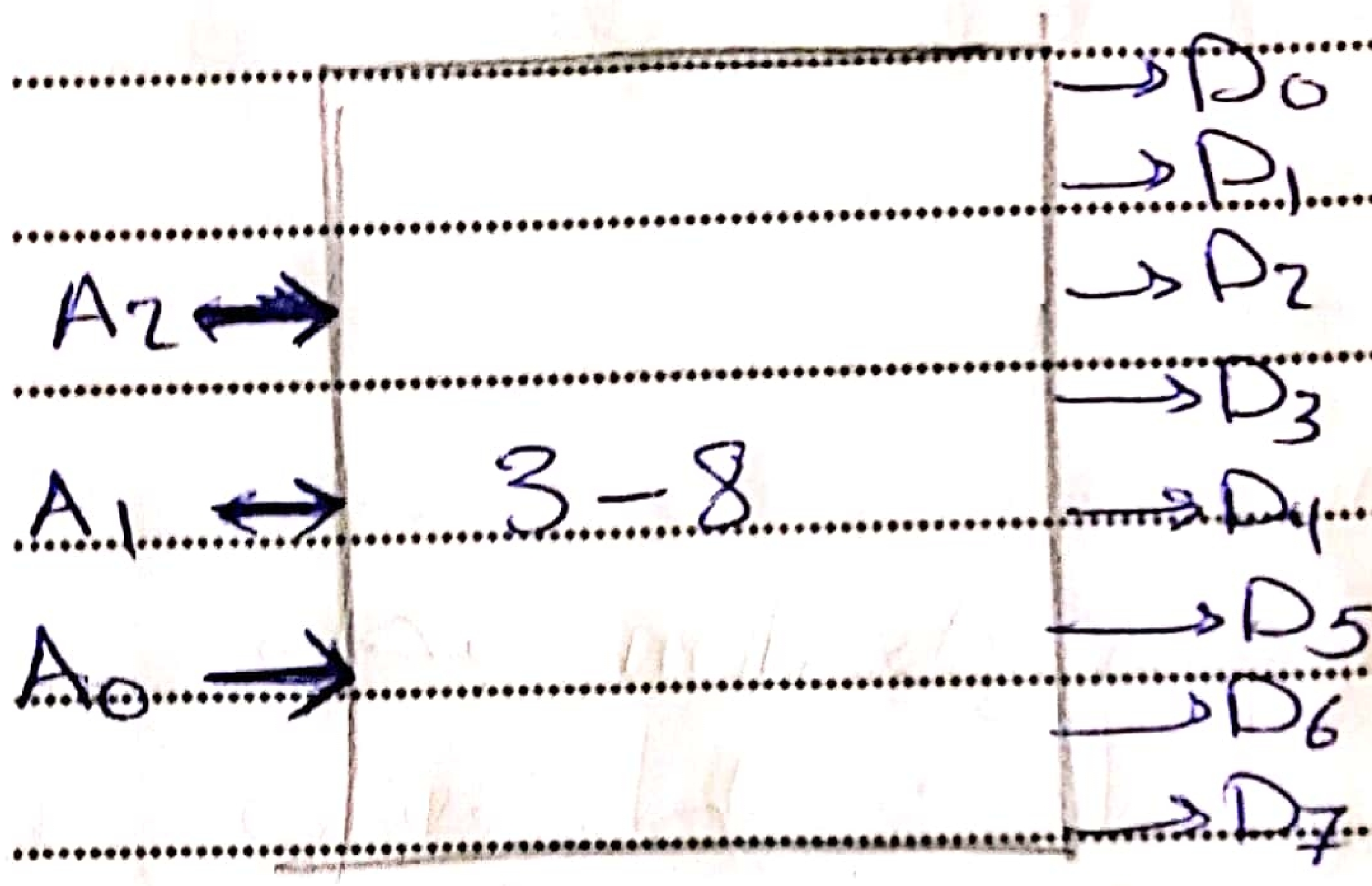
$D_3 = A_0 A_1$

واحد item دايمًا بيجي
والباقي بيطلع



SUBJECT:

3 – 8 decoder



→ truth table in slide 30

→ to draw the diagram
you need 8 And gates
and 3 inverters.

→ functions are written as

$D_0 = \bar{A}_0 \bar{A}_1 \bar{A}_2$

SUBJECT:

to design k-to-2^k decoder

→ you need k inverters

→ 2^k k-input And gates

to find GN of decoder:

$$GN = \underbrace{k}_{\text{inverters}} + \underbrace{2^k * k}_{\text{And gates}}$$

هاي الطريقة حل بناء الدارة و رسمها منطقية لما عدد k قليل،
لكن لما يصير كبير بتصير خطيرة ← اسمها The simple Approach

SUBJECT: .... / /

Decoder expansion approach

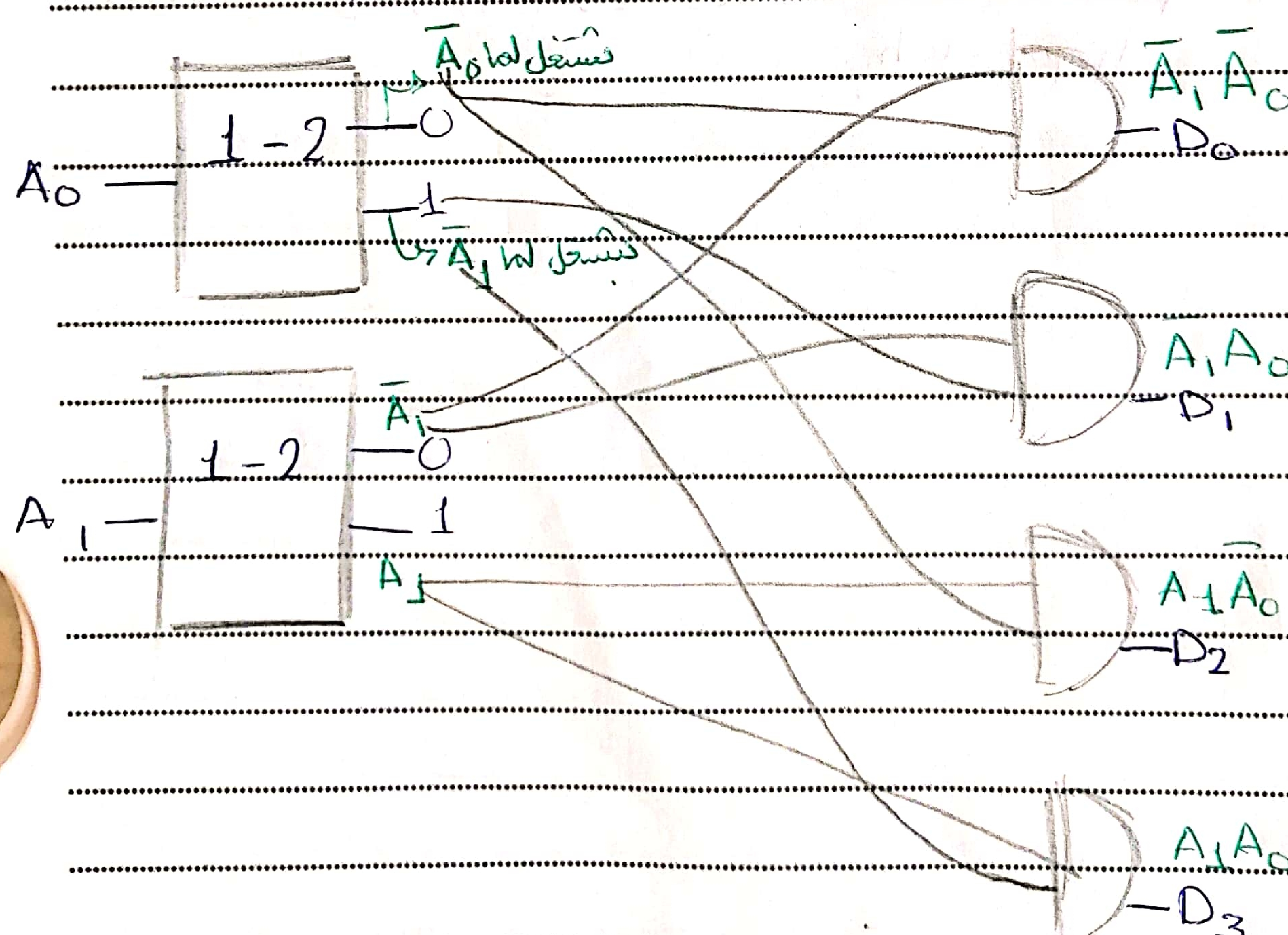
بنحطين GN أقل عدد لأول

بناخذ components صغار و بنعملها تركيب لحتى تعمل الـ circuit الكبيرة

(مقسوم على 2) ← اصغر decoders صغار و بضرب الـ output بيعرض عشان اي decoders كبار

2-4 [decoder]

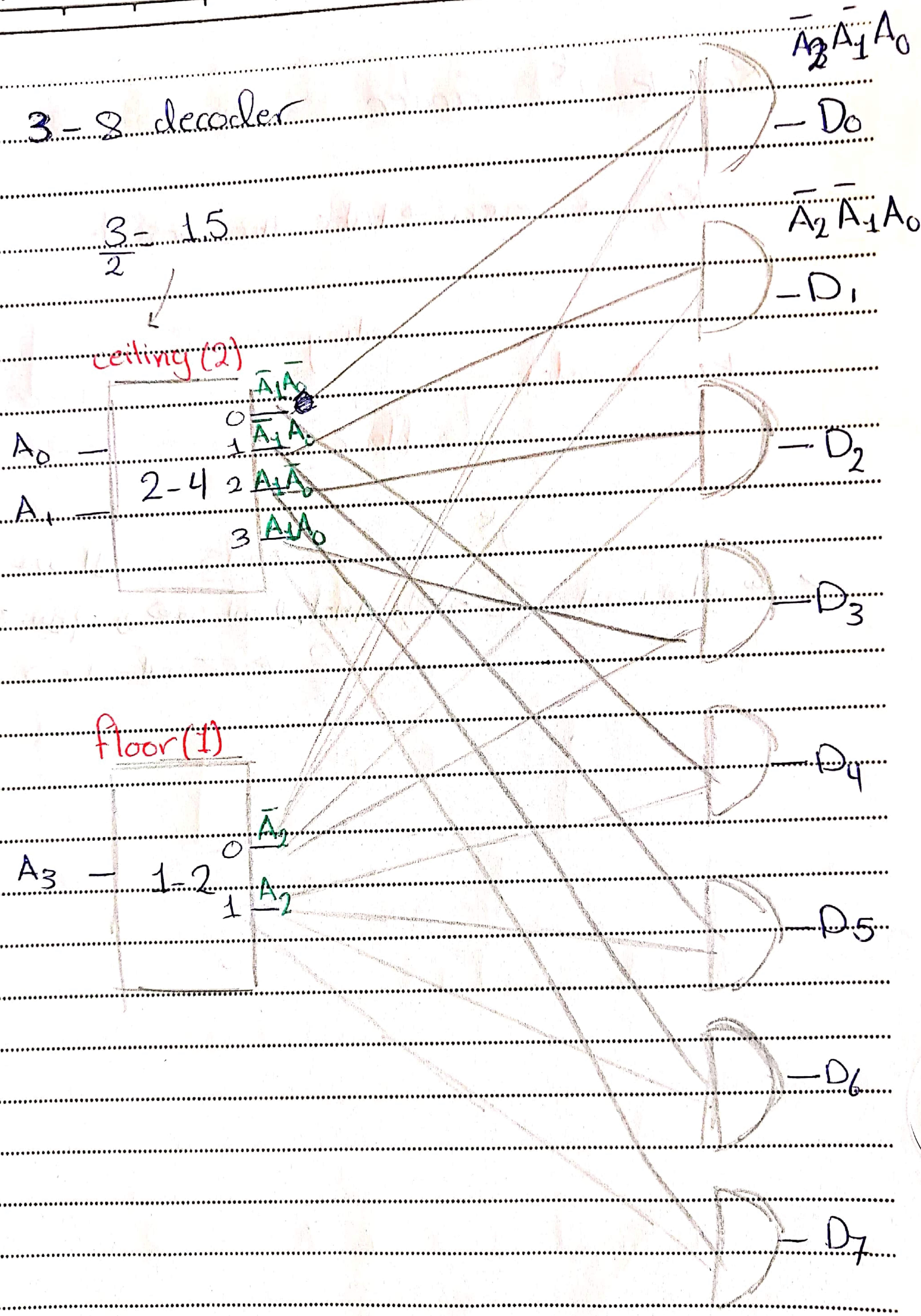
$$\frac{\text{الحجم الأصلي}}{2} = \frac{2}{2} = 1$$



SUBJECT:

3-8 decoder

$\frac{3}{2} = 1.5$



SUBJECT:

So $k - 2^k$ decoder

$k/2 \rightarrow$ even $\rightarrow$ take the result

$k/2 \rightarrow$ odd $\rightarrow$ ceiling / floor

عدد ال ands هو عدد 2^k

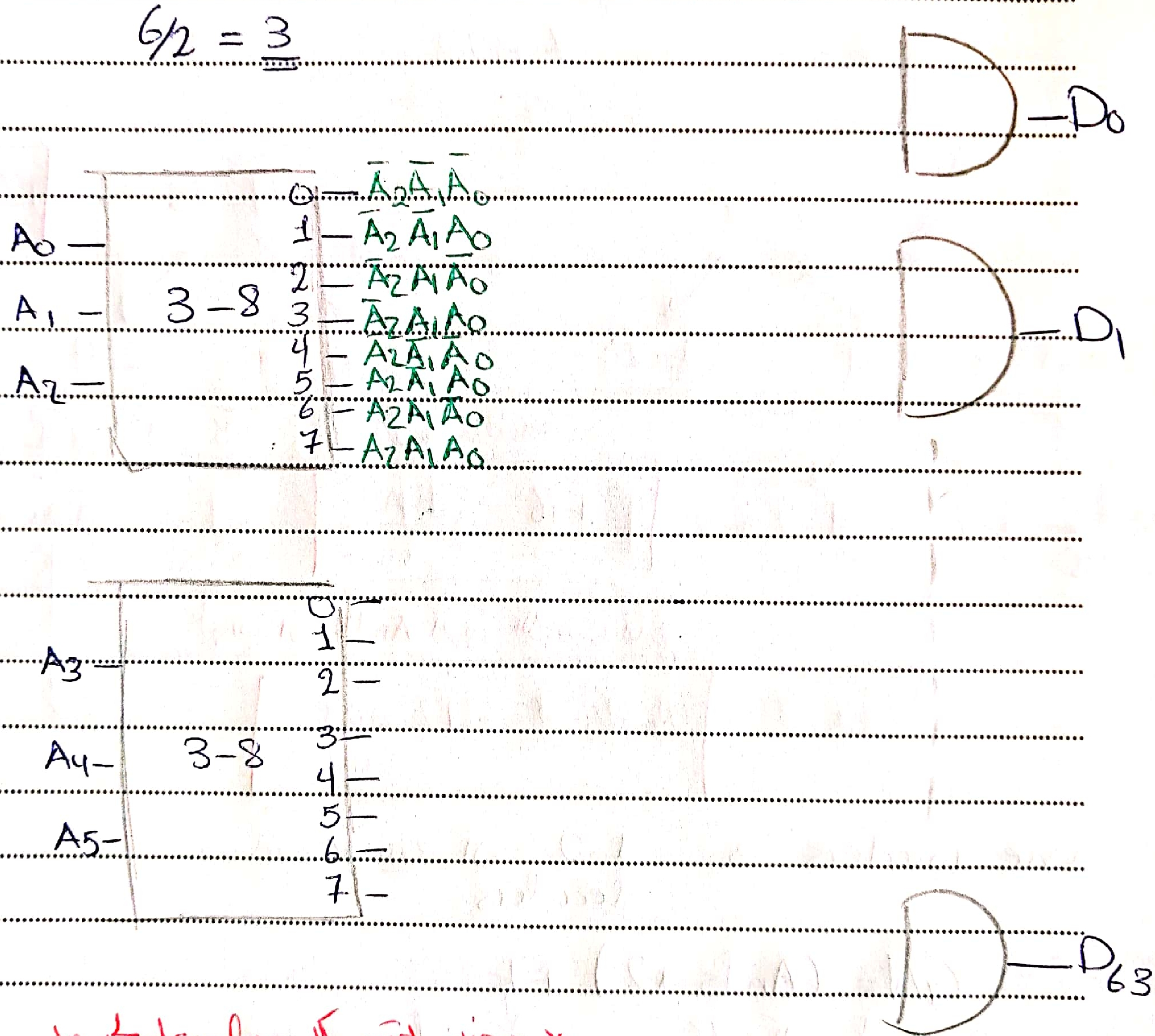
← بس لاحظو هون انه كل And راح يفوت عليها سلكين بس ✓

دايماً بيكون عددها 2

SUBJECT: .... / /

6-64 decoder

$6/2 = 3$

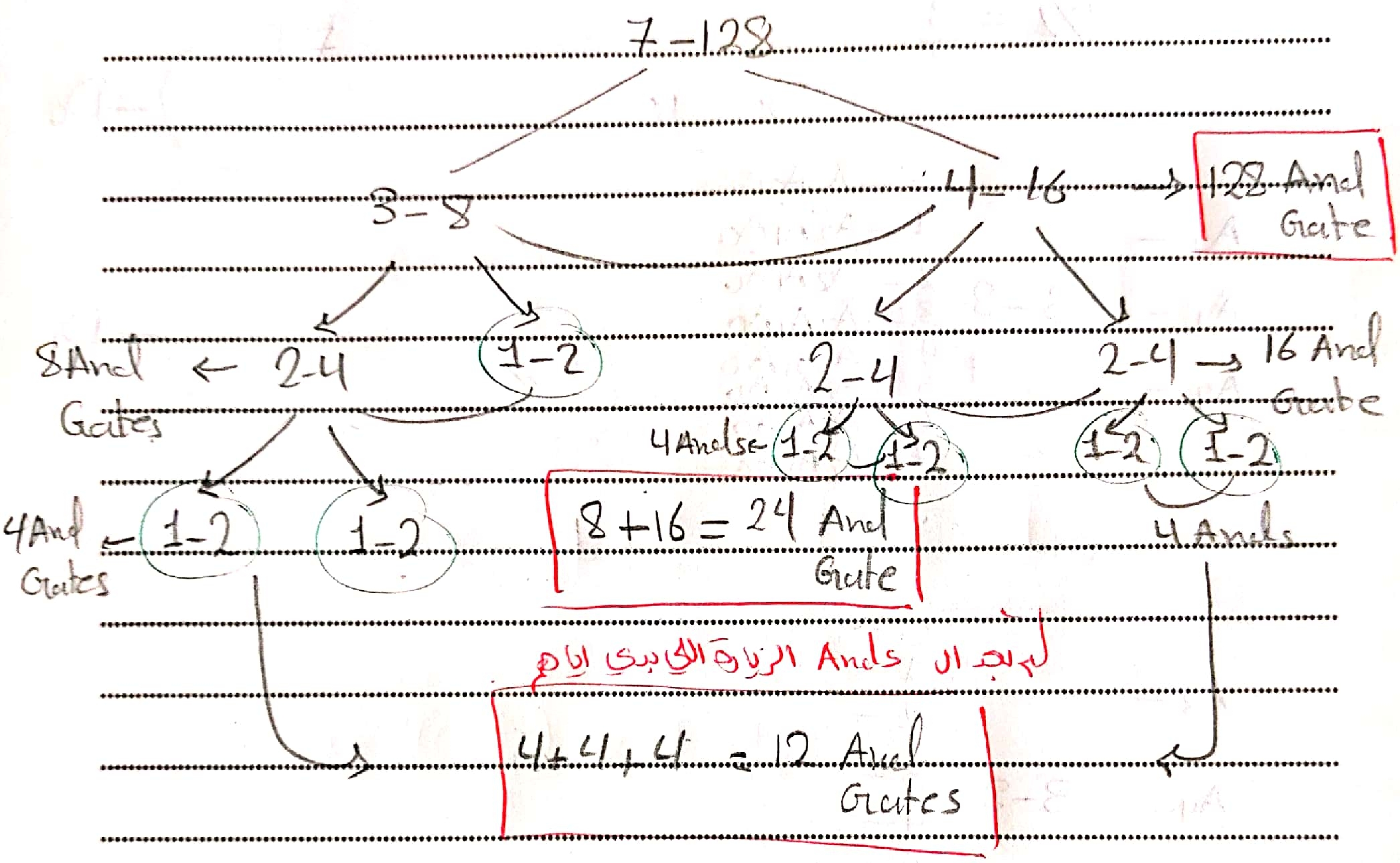


* بحطلي اكتر كل decoder لحد ما

decoder ← 1-2 اوصل

64 And gate

example: slide 38 [7-128 line decoder]



the inverters → موجودين في ال 1-2 decoders

GN = (Ands * 2) + k

لانه كل And منهم داخلها 2 واير عليها سلكين

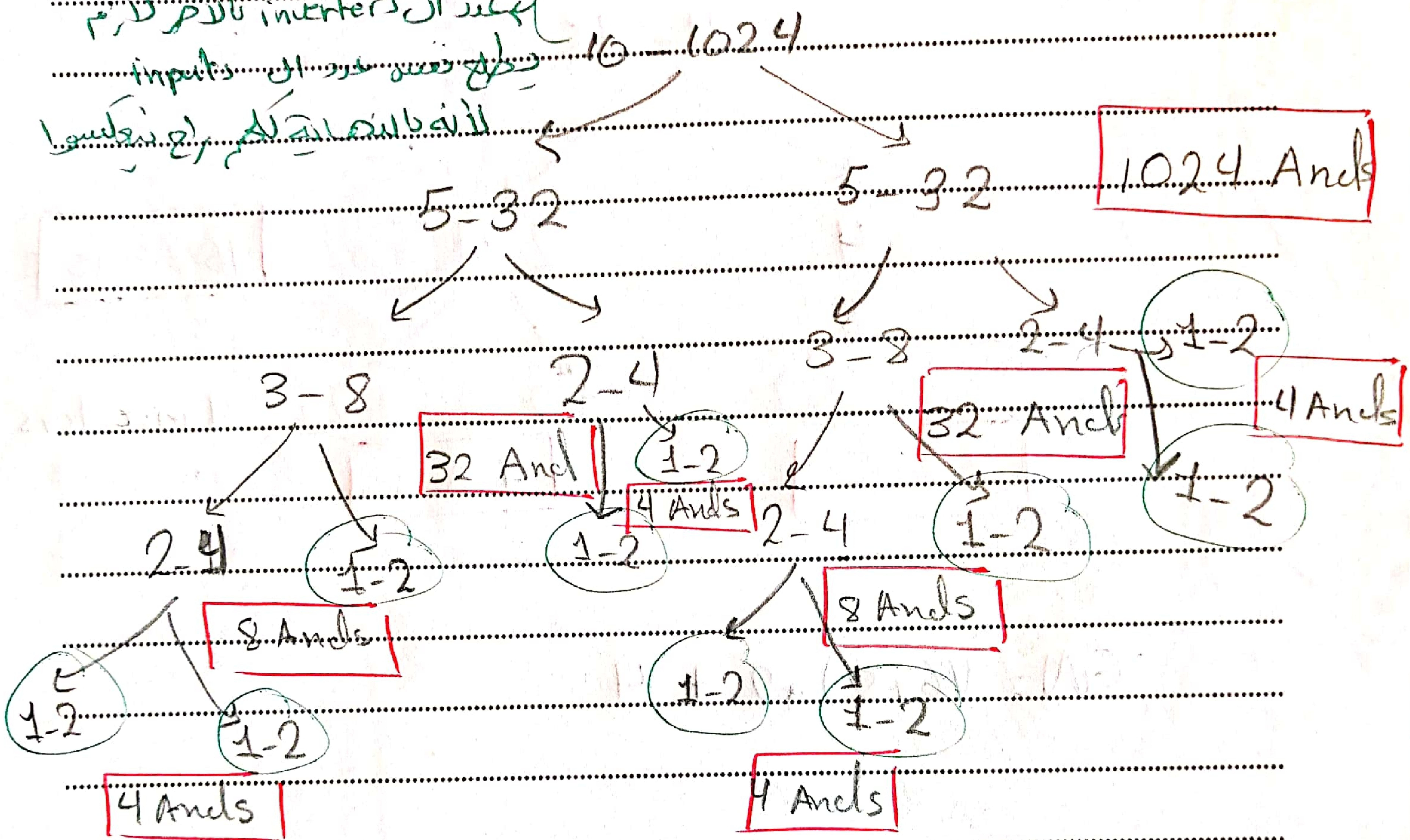
k → num of inverters

= (128 + 24 + 12) * 2 + 7 = 335

لو حسبتها بالطريقة الأولى تبع تقسيم ال (decoders) على معادلة ال GN
لاحظي راح تطلع = 903 ← التكلفة اعلى

example : find GN for $10-2^{10}$ decoder by expansion approach

بعدد ال inverters تالآخر لازم يطلع نفس عدد ال inputs لأنه بالنهاية كلهم رح ينعكسوا



$$GN = (1024 + 32 + 32 + 4 + 4 + 8 + 8 + 4 + 4) * 2 + 10$$

SUBJECT:

Example: find cost of 4-16 decoder by expansion approach

4-16

2-4 2-4

$\boxed{\text{16 Ands}}$

1-2 1-2 1-2 1-2

4 inverters

$\boxed{\text{4 Ands}}$ $\boxed{\text{4 Ands}}$

$GN = (16+8)*2 + 4$

SUBJECT: .... / /

Decoder with Enable

بنستخدم سلك الـ Enable عليها ، كل الـ outputs الي طالعين من الـ And Gate ، بربط بسلك كل وحدة مع And Gate جديدة وكل واحدة And Gate

بسلك جديد مع الـ EN (السلك يلي ا.و)

تصميم بسهّل الـ Enable ← Extra set of Ands

* الـ Enable بتعطيني مرونة أفكر بطريقة جديدة ، أصمم عن طريقها decoders أكبر.

إذاً في 3 طرق لتصميم الـ [decoder]

① تصميم الـ circuit بالطريقة العادية

② الـ expansion Approach ③ Using Enables

* وظيفة منوصلها بنفس الـ decoder
الـ decoder وظيفته الأصلية يترتب عليه دخل code ، item
بس لأنو دخل كل الـ combinations بس بيشتغل كل من واحد فيهم ، Enabler
(الـ) عدد البواقي أصفار

SUBJECT: / /

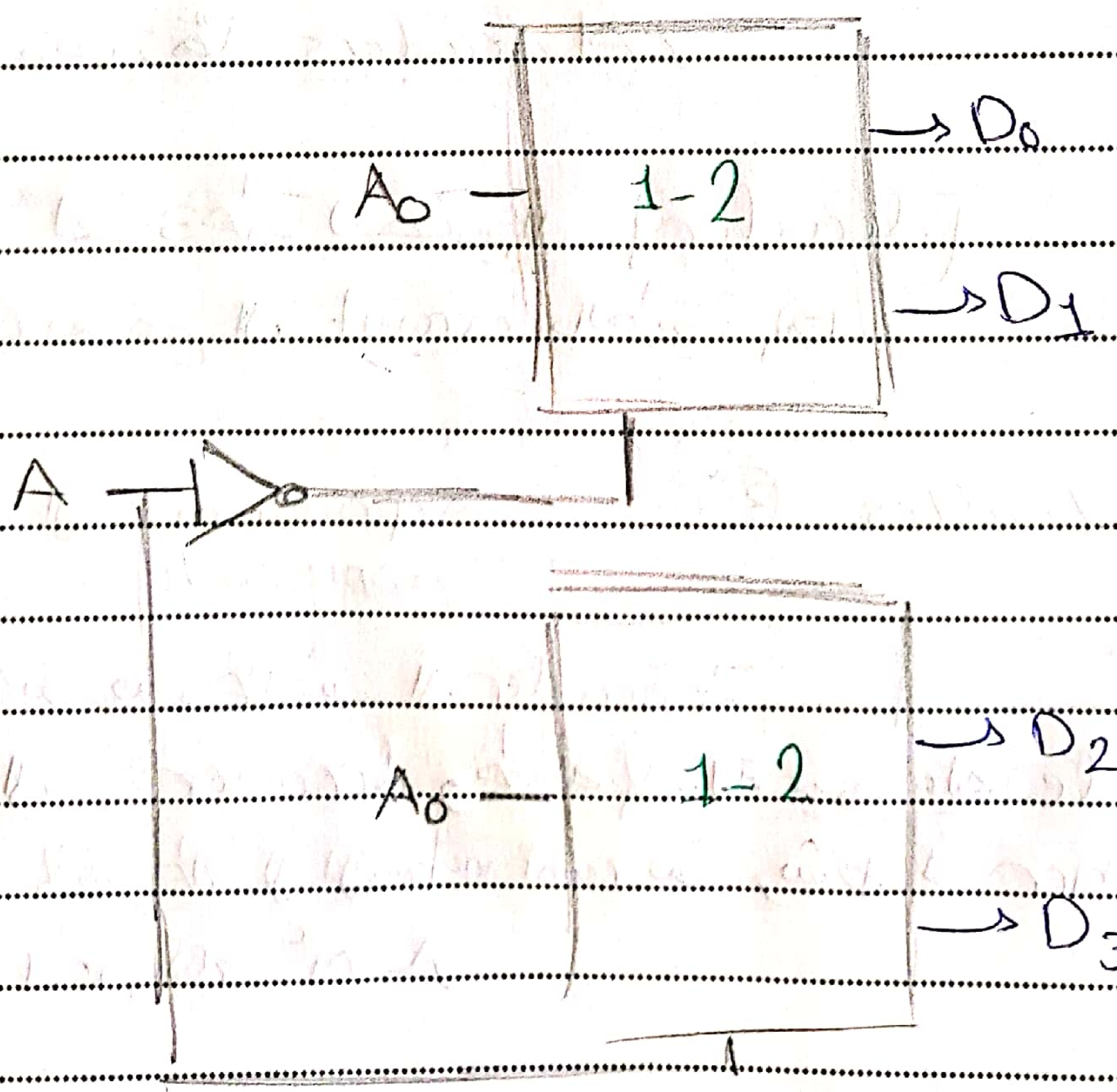
example : slide 41

2-4 line decoder

truth table

Enable

A_1	A_0	OUTPUT
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

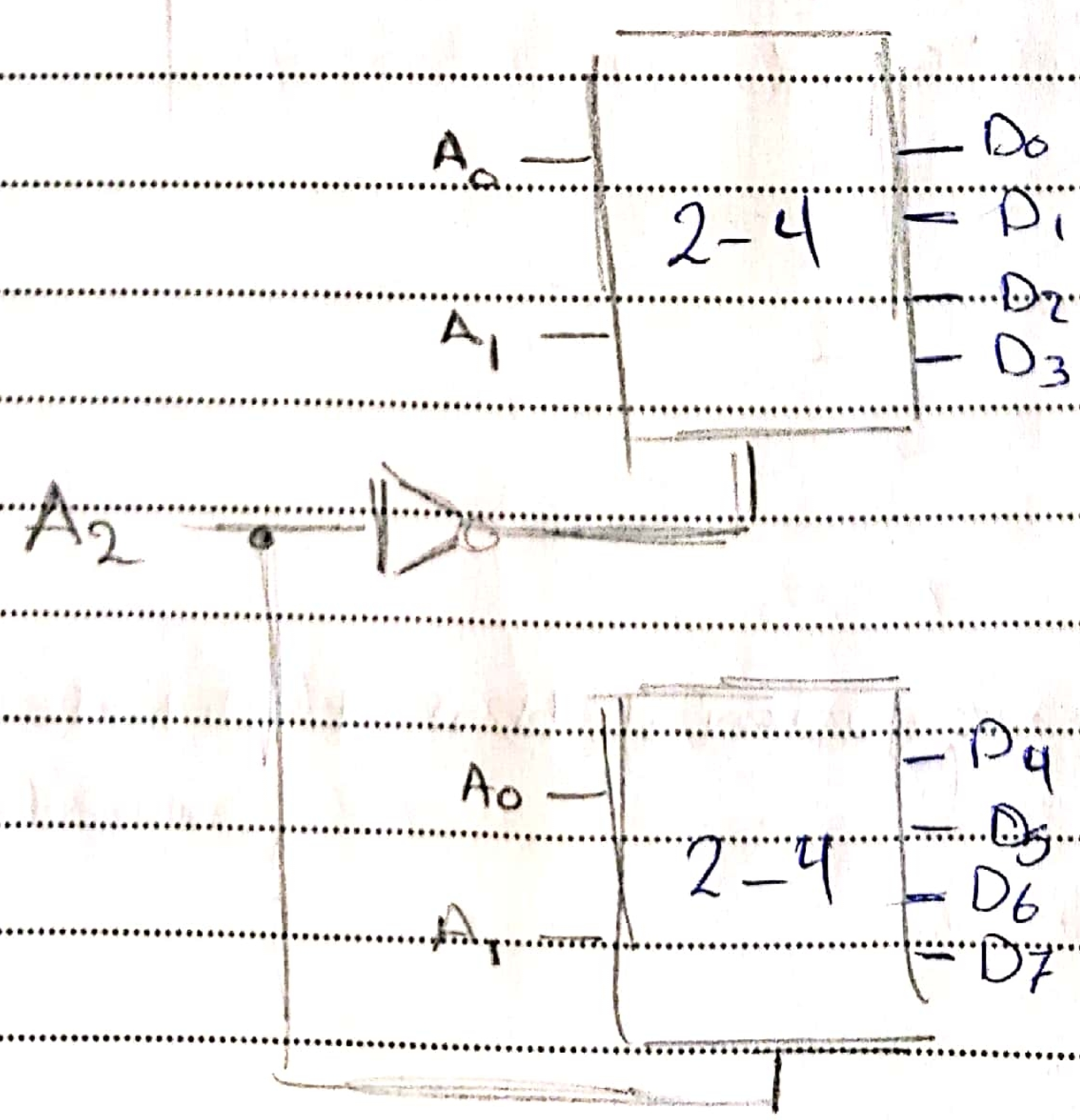


SUBJECT:

Example: slide 42

$3 \to 8$ decoders

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



Exercise: Slide 43

4 - 16 decoder using fire 2-4 line decoder ~~[illegible]~~ with enables

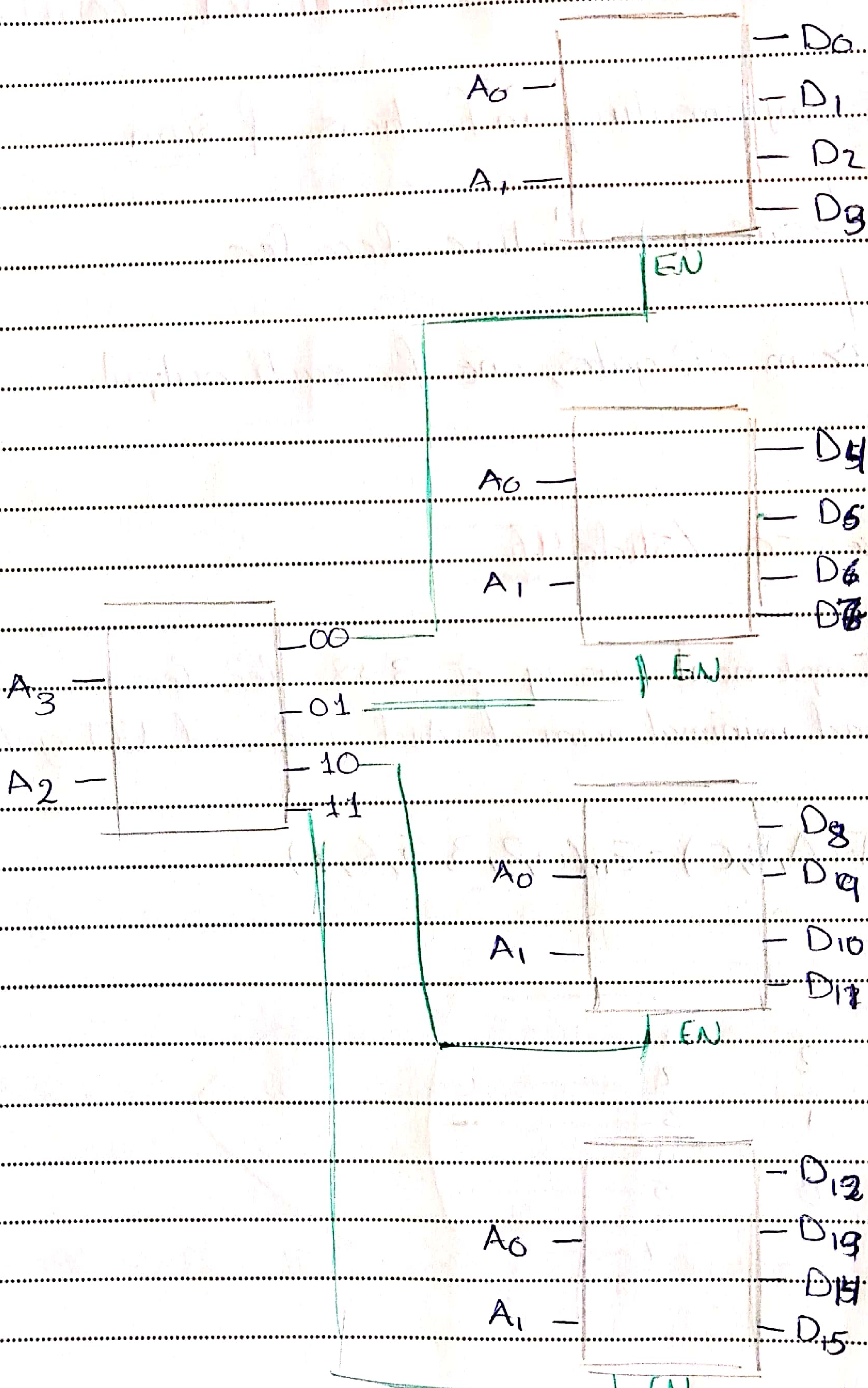
A_3	A_2	A_1	A_0	
0	0	0	0	D_0
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	D_{15}

نقسمها أربع أرباع
بحيث سلكين يلعبوا
دور ال enable

وسلكين يلعبوا دور ال
selection

نقسم ال truth table لأرباع او أنصاف، حسب شو معطيني
عدد encoders

SUBJECT:



A0
A1
D0
D1
D2
D3
EN
A0
A1
D4
D5
D6
D7
EN
A3
A2
00
01
10
11
A0
A1
D8
D9
D10
D11
EN
A0
A1
D12
D13
D14
D15
EN

SUBJECT:

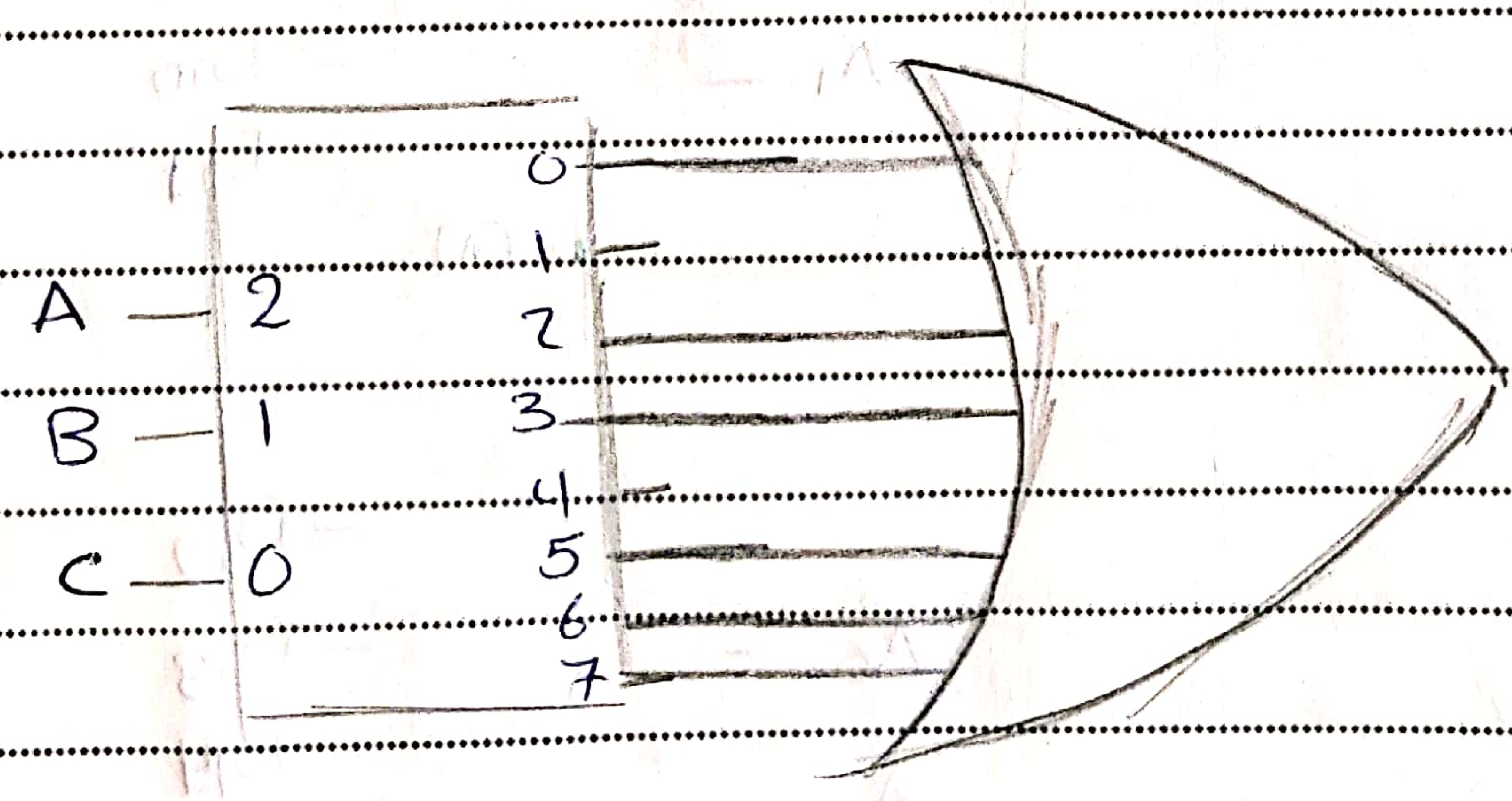
Decoder-Based combinational circuits

- implementing m functions of Sum
  - one n-to-2^n-line decoder
  - m OR gates, one for each output.

Exercise /Slide 46

Implement using a 3→8 decoder and minimal num of And, OR and NOT gates.

① $F(A,B,C) = \Sigma_m(0,2,3,5,6,7)$



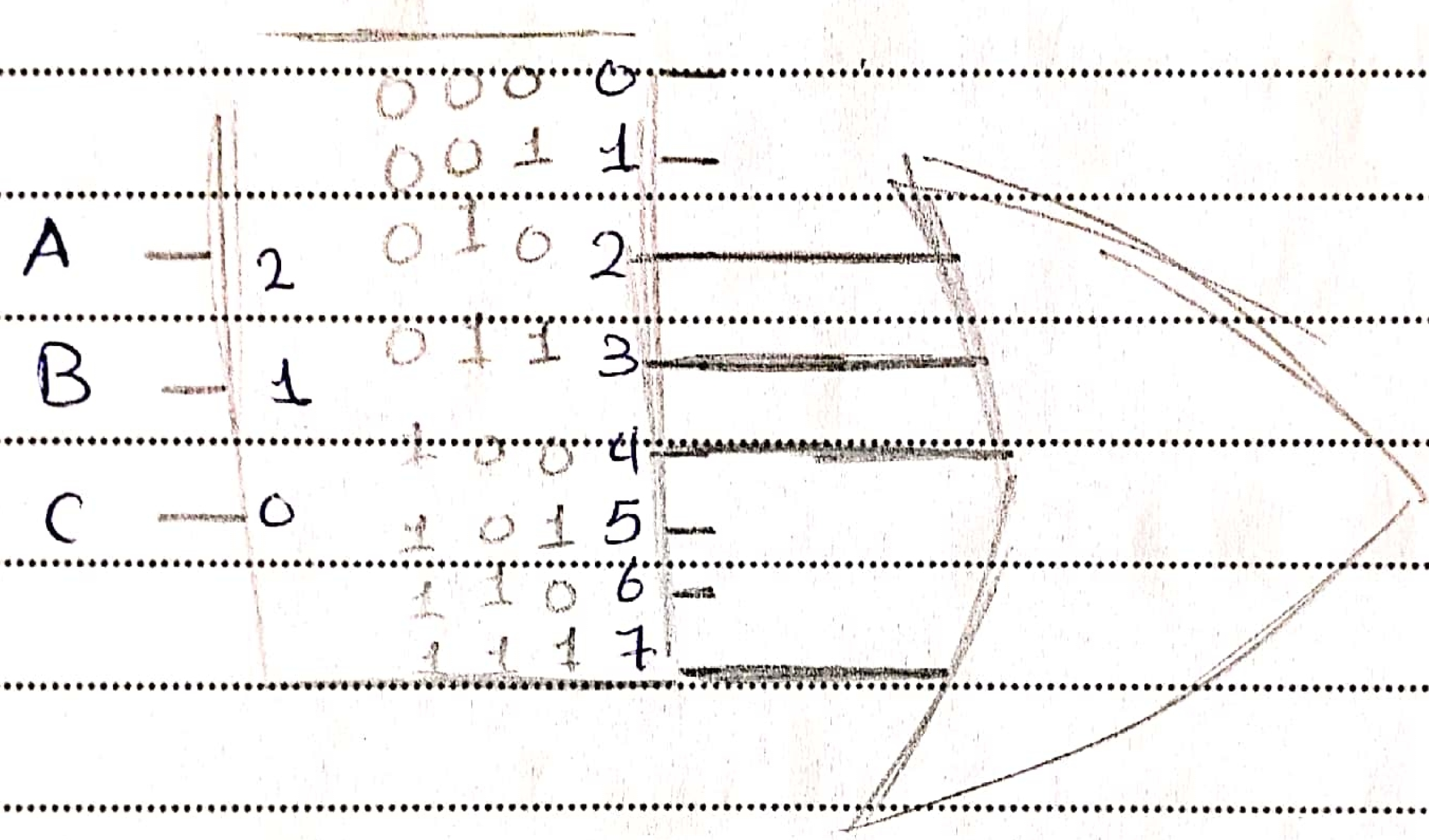
② $F_2(A,B,C) = A\bar{B} + BC$

↳ turn it into <u>SUM</u>

$= A\bar{B}(C+\bar{C}) + (A+\bar{A})BC$

$= A\bar{B}C + A\bar{B}\bar{C} + ABC + \bar{A}BC$

2 4 7 3



③ $F(A,B,C) = A + B + \bar{C}$

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

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

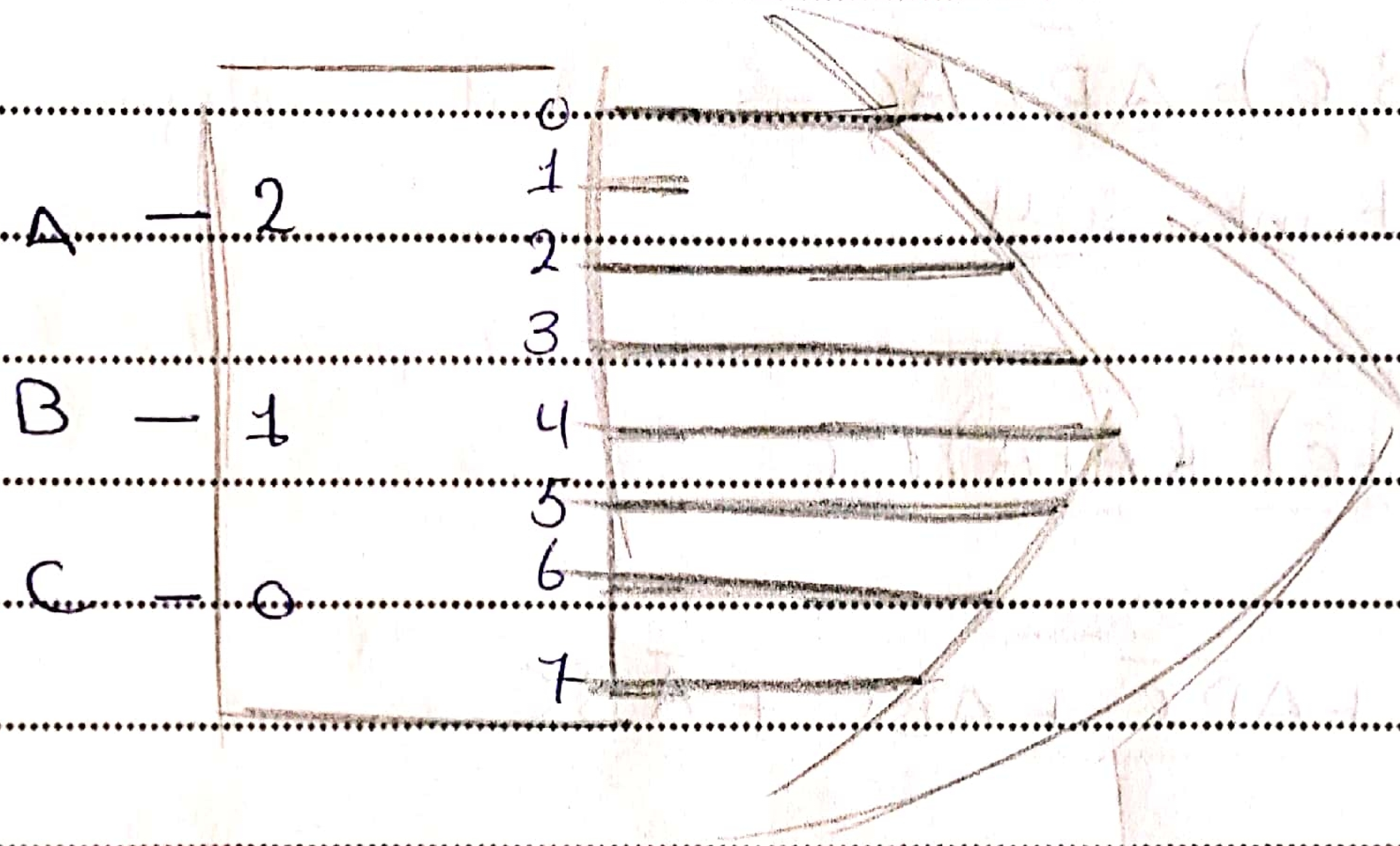
$AB\bar{C} + A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}\bar{C}$

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

111 110 100 010 101 011 110 000

7 6 4 2 5 3 6 0

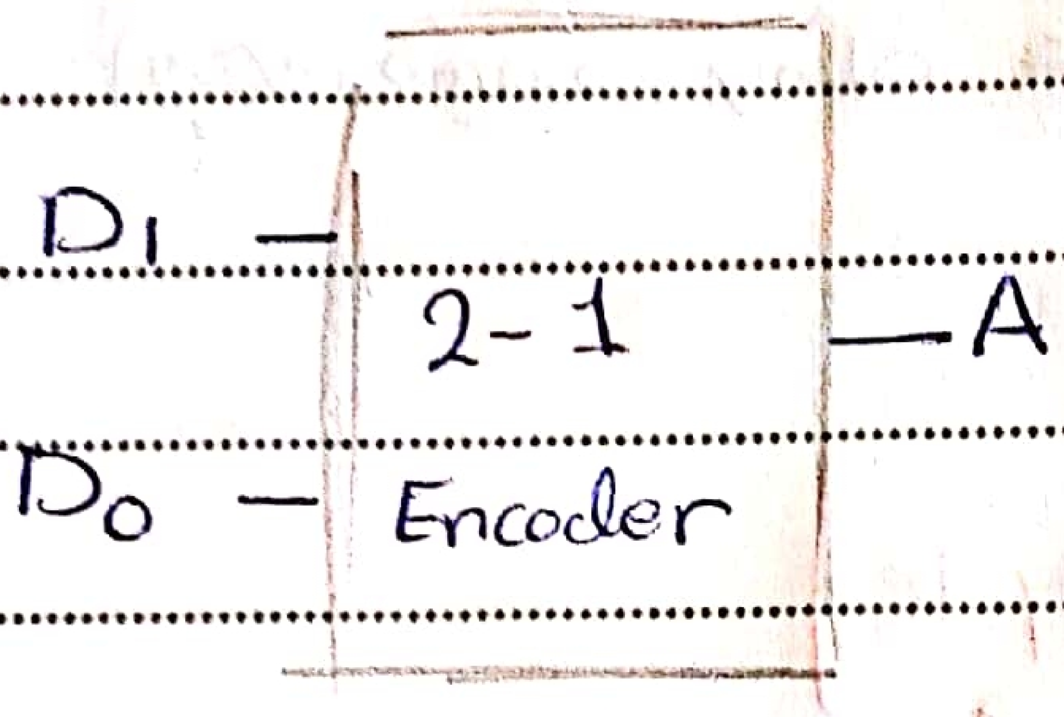
كلمطلها الا (1) ←


A — 2
B — 1
C — 0
0
1
2
3
4
5
6
7

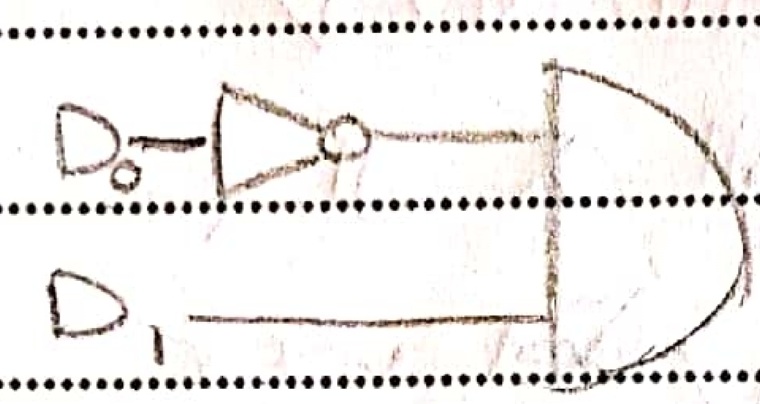
SUBJECT:

[Encoding]

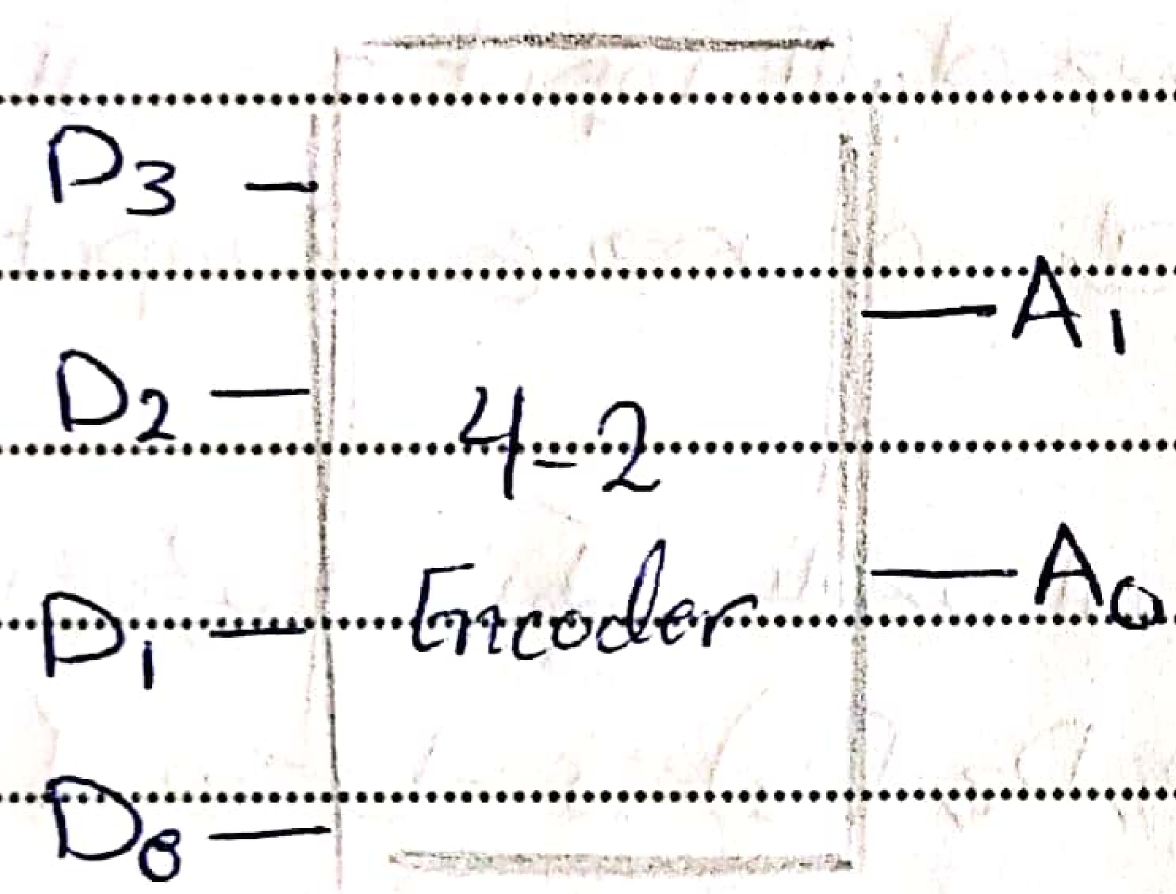
نحوّل عليه ال items
وبطلع ال codes



D_1	D_0	A
0	1	0
1	0	1



$A = D_1 \bar{D}_0$



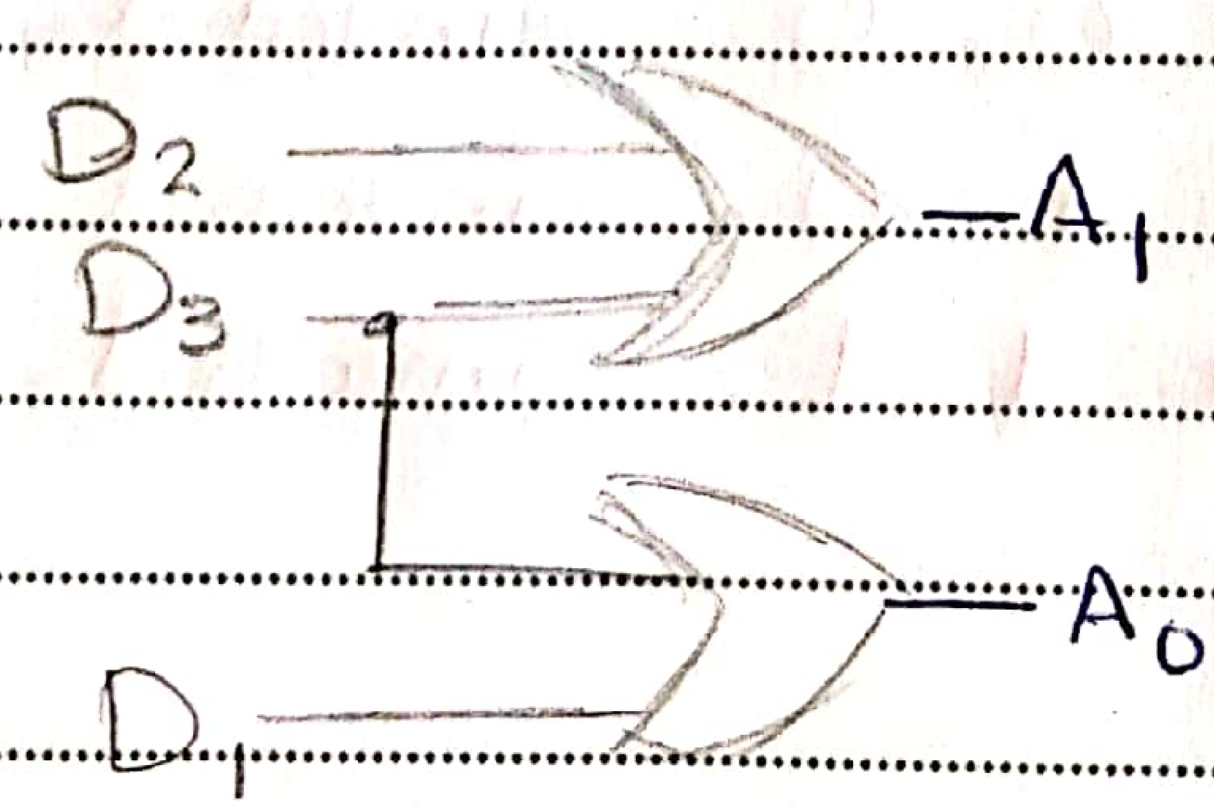
D_3	D_2	D_1	D_0	A_1	A_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

بم ال 1 صحابها انه دا بإجابة ال D_3
او جابة D_1

$A_1 = D_3 + D_2$

$A_0 = D_3 + D_1$

الكود اللي بطلع



SUBJECT: ....

→ Example / Slide 56

اخترت الي بدي ياه 4 bits من ال outputs لأنه ال 3
أقصى شي بتقدر تمثله ← 8 combinations

[Priority Encoder Example]

High priority Encoder ✓

*جدول slide 54 ← ال X الي في خانات ال input
معناها انه اذا ال user اختار رقم معين (عدد input)
كل ال inputs الي أصغر منه بطل الها قيمة، فلو خطأً اختار
2 inputs مع بعض، مرح ينظر الإلهم تلقائياً يهمل الي اقل منه

*بهاد الجدول انا بكير عالجت ال 16 حالة لل 16 combinations
الي ممكن ينتجوا عن (D0, D1, D2, D3)، لانه مثلاً ال
input 1 X X X
D_3 D_2 D_1 D_0

ممكن يمثل :

1	0	0	0	valid
1	1	0	0	invalid
1	1	1	0	invalid
1	1	1	1	invalid

فبستخدم ال priority
راح يختار اول وحدة
لانها valid ويهمل ال
bit أقل منه تلقائي
بحيث يكون ال input هو
فقط D_3

لنفس الجدول :

$V = D_3 + D_2 + D_1 + D_0$

$A_1 = D_3 + \bar{D}_3 D_2 = D_3 + D_2$ ← بعد التبسيط باستخدام $(X + \bar{X}Y)$

لما اكتب معادلات الجدول بيتضمن الـ validity ، اذا معطيت input لـ الي اعلى منها ، اجباري احط جزئي اعلى منها الي بكونو صفر كـ $(\bar{A})$ او $(\bar{D})$

$A_0 = D_3 + \bar{D}_3 \bar{D}_2 D_1 = D_3 + \bar{D}_2 D_1$ بعد التبسيط ✓ $(X + \bar{X}Y)$

← الـ High priority ، بتعطي اولوية للـ input الأعلى
فلازم دائماً اتأكد انه كل الي اعلى منها معطى = 0 عشان يختارها هي ما يختار اشي اعلى منها

← الـ low priority ، بتعطي اولوية للـ input الأقل
فلازم دائماً اتأكد انه كل الي اقل منها معطى = 0 عشان يختارها هي ما يختار اشي اقل منها

SUBJECT:

Example for low priority

D_3	D_2	D_1	D_0	A_1	A_0	V
0	0	0	0	X	X	0
X	X	X	1	0	0	1
X	X	1	0	0	1	1
X	1	0	0	1	0	1
1	0	0	0	1	1	1

$V = D_3 + D_2 + D_1 + D_0$

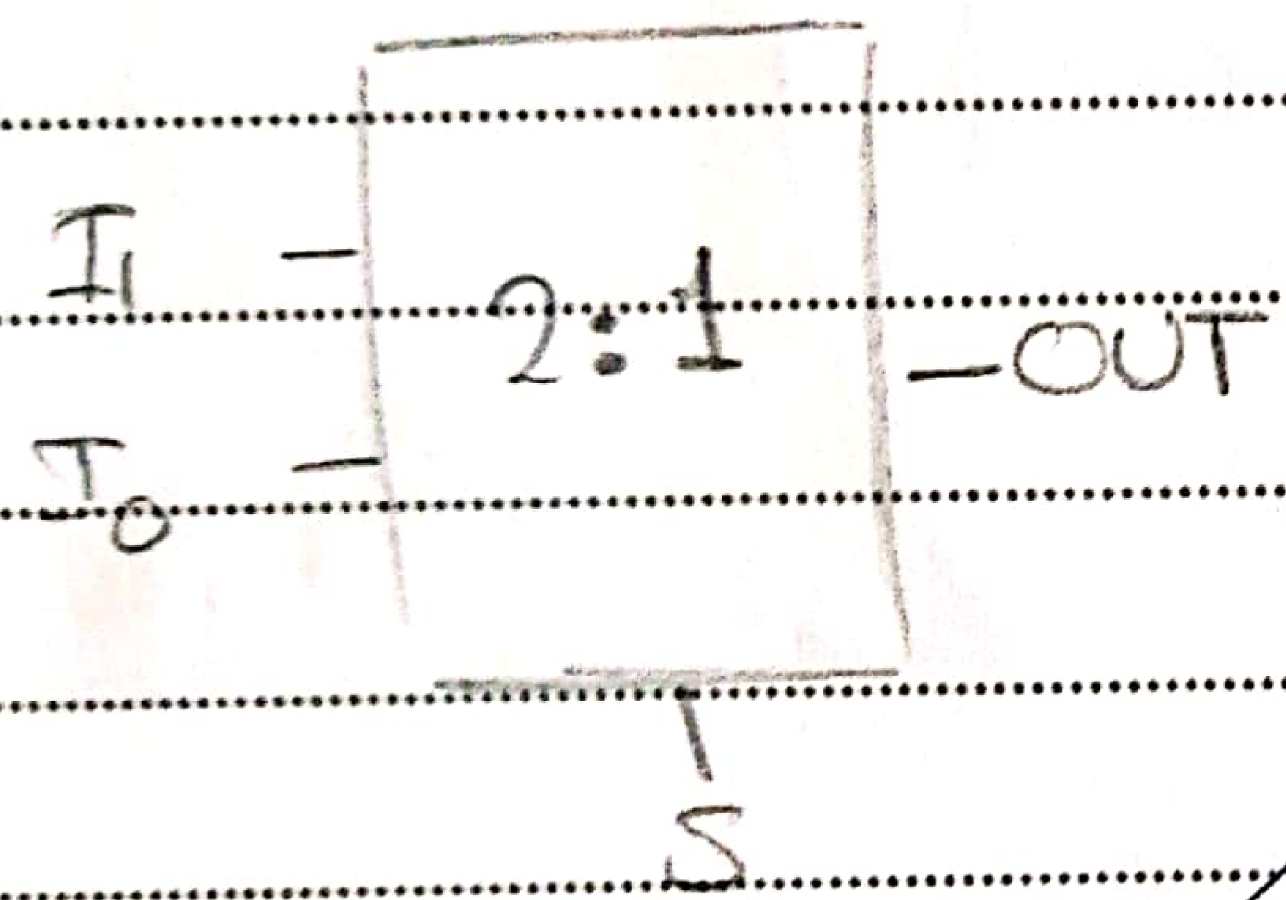
$A_1 = D_3 \overline{D}_2 \overline{D}_1 \overline{D}_0 + D_2 \overline{D}_1 \overline{D}_0$

$A_0 = D_3 \overline{D}_2 \overline{D}_1 \overline{D}_0 + D_1 \overline{D}_0$

[Multiplexers]

* Selection circuit depending on controller choices

example:



S	I_1	I_0	OUT
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

optimal منقدر اطلع بحركة

$OUT = \bar{S} I_0 + S I_1$

S	OUT
0	I_0
1	I_1

لما S=0
بنسخ I_0 ع ال OUT

condense truth table

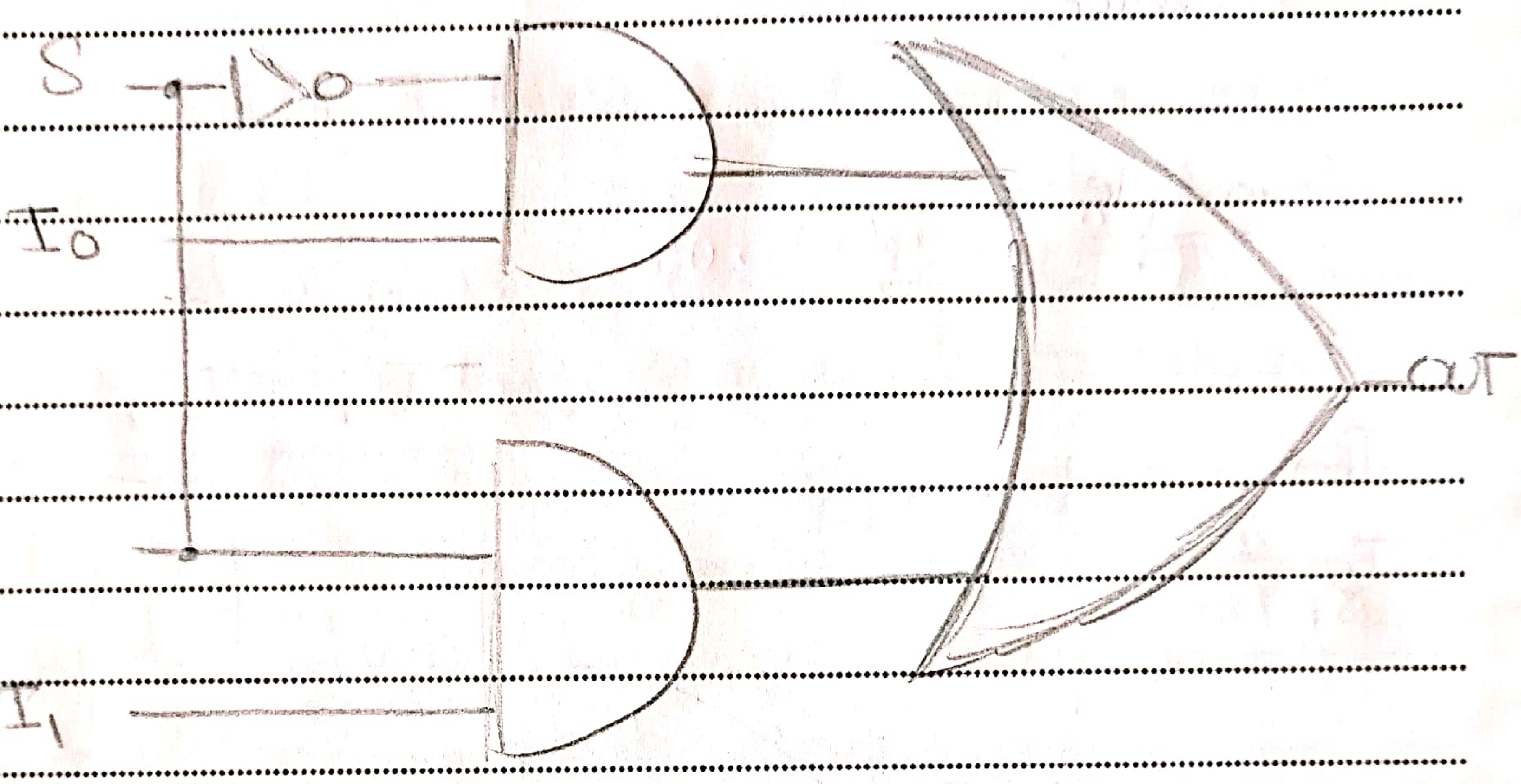
S \ $I_1 I_0$	00	01	11	10
0	0 (0)	1 (1)	1 (3)	0 (2)
1	0 (4)	0 (5)	1 (7)	1 (6)

I_1 / I_0 / S

from k-map ✓

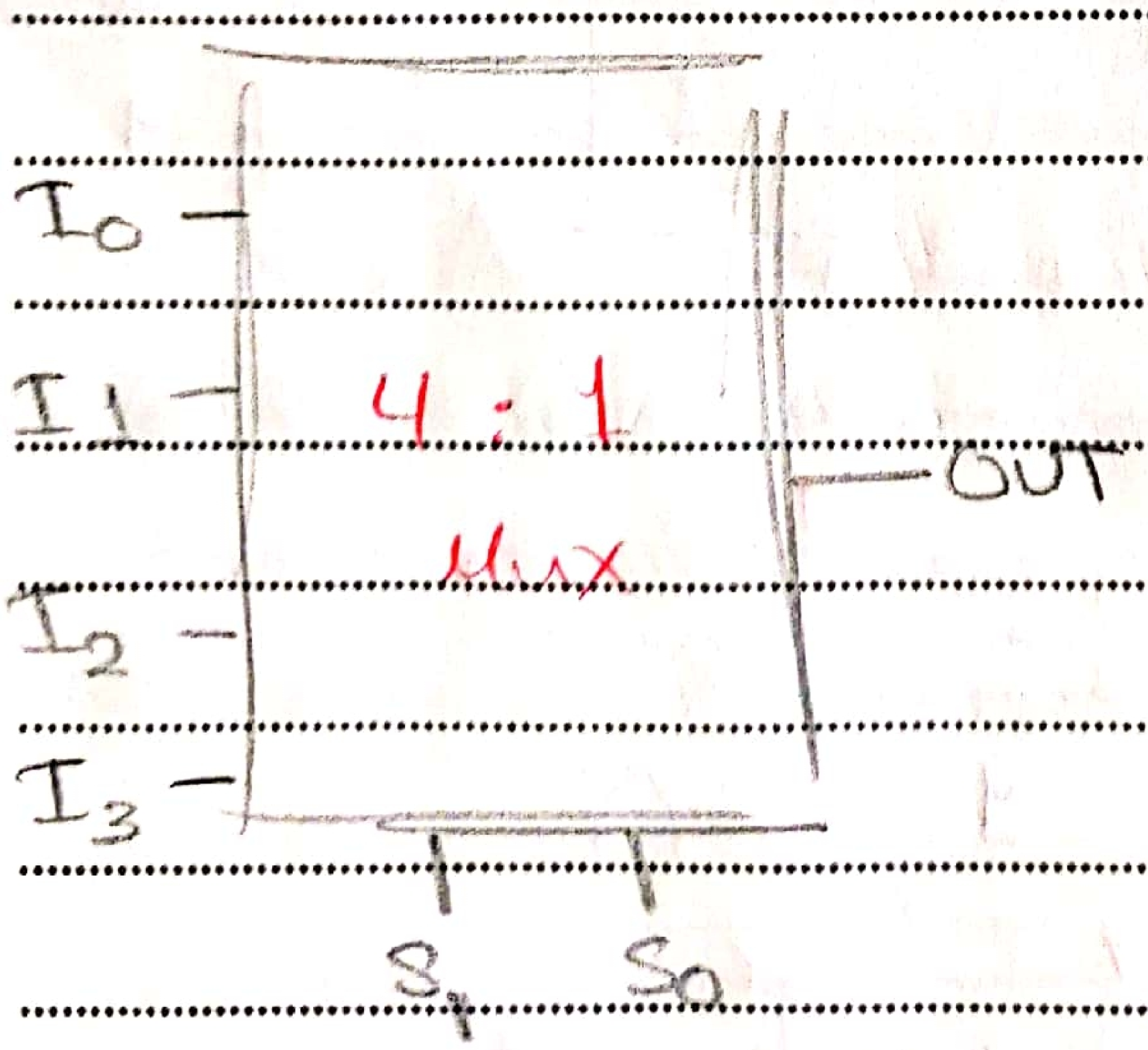
$OUT = \bar{S} I_0 + S I_1$

لو نفسرها باللغة بوصف ←
مين بيشتغل وليها بشغل ال multiplixer



← مصطلح الحالة optimal مباشرة من ال condensed truth table

مثلاً ←



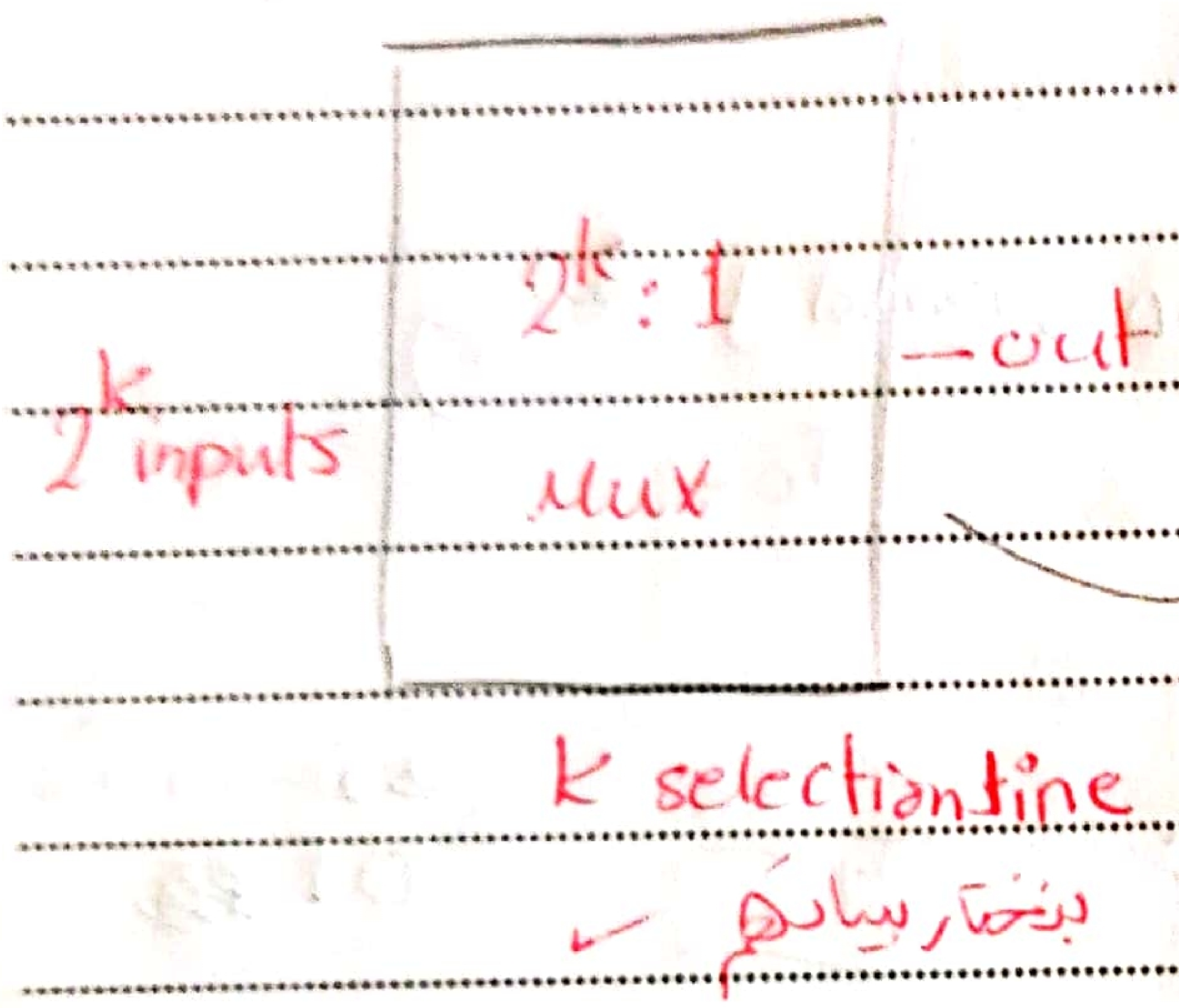
S_1	S_0	OUT
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

$OUT = \bar{S}_1 \bar{S}_0 I_0 + \bar{S}_1 S_0 I_1 + S_1 \bar{S}_0 I_2 + S_1 S_0 I_3$

SUBJECT:

ال Mux :-

← الطريقة ال optimal ← : بتطبق جميع حالات ال signals
الي ممكن تطلع ←



S_{k-1} , S_{k-2} ... S_0	out
0 ... 0	I_0
1 ... 1	

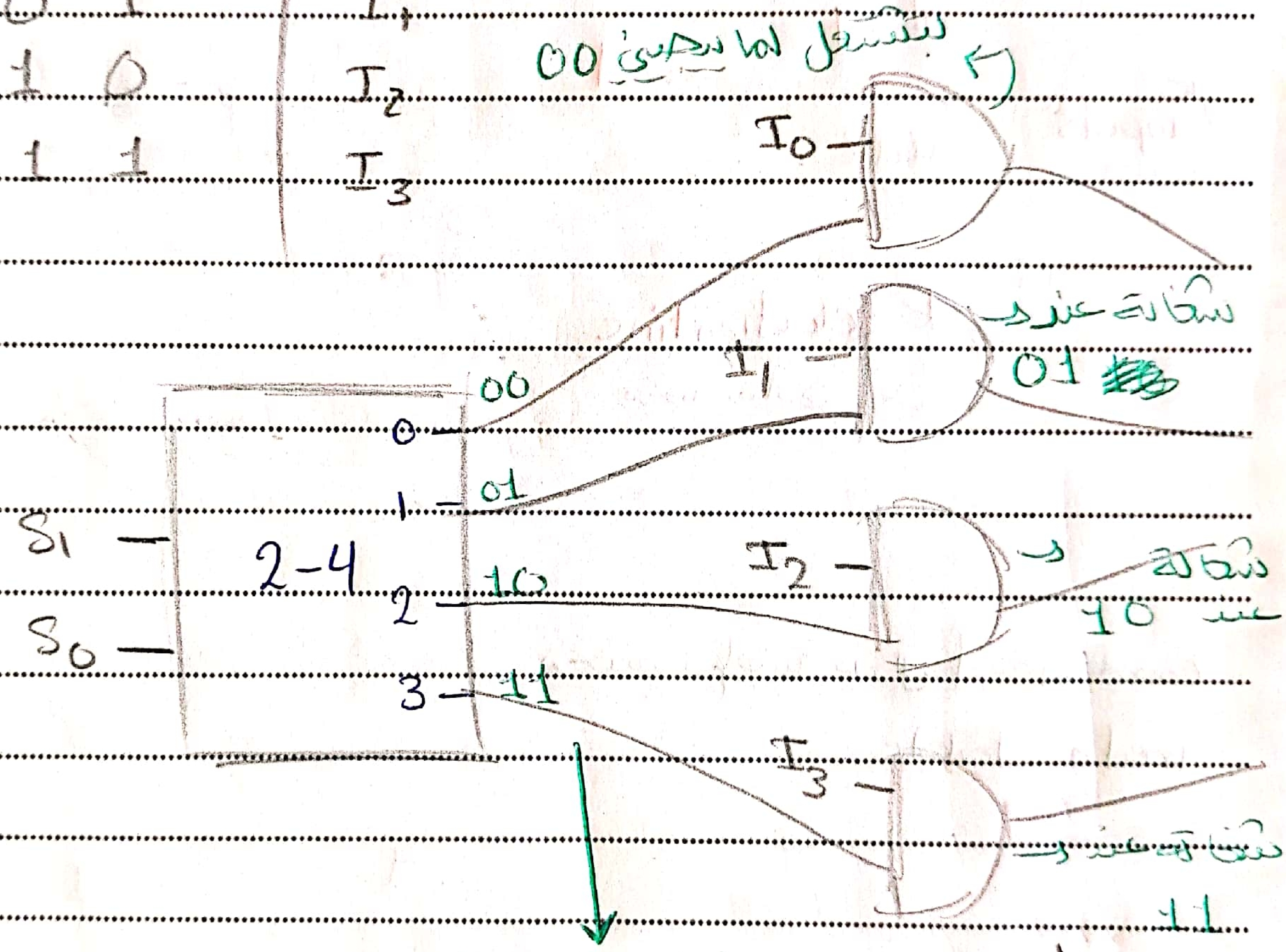
بنحسب بطريقة condensed
truth table

* ال Mux بتحوي decoder وراها Ands
← check slide 61

نفترض ارسم ال Mux باستخدام ال ~~Encoder~~ decoder

مثال :-

S_1	S_0	OUT
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



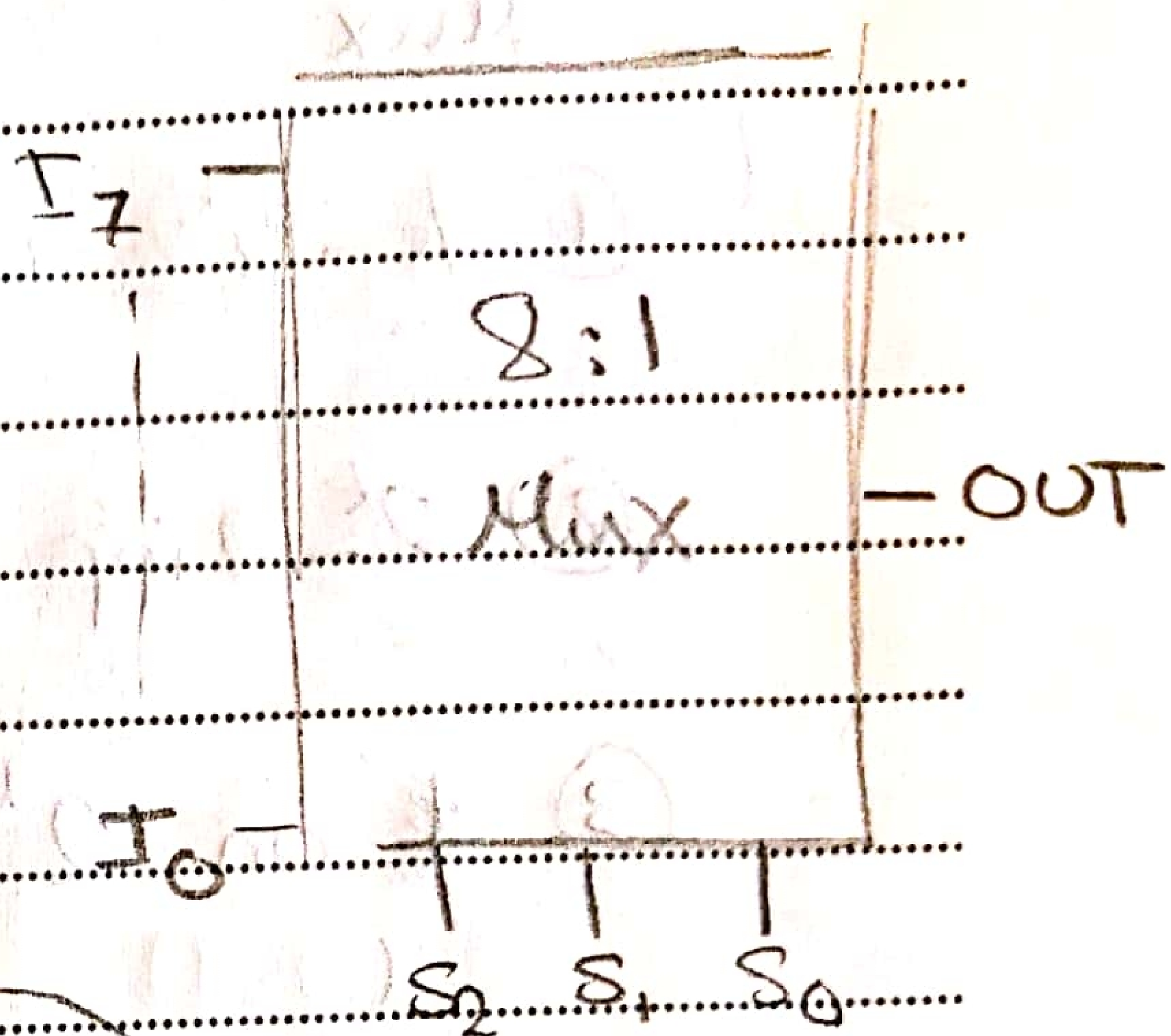
بتطلع كل الحالات ال output

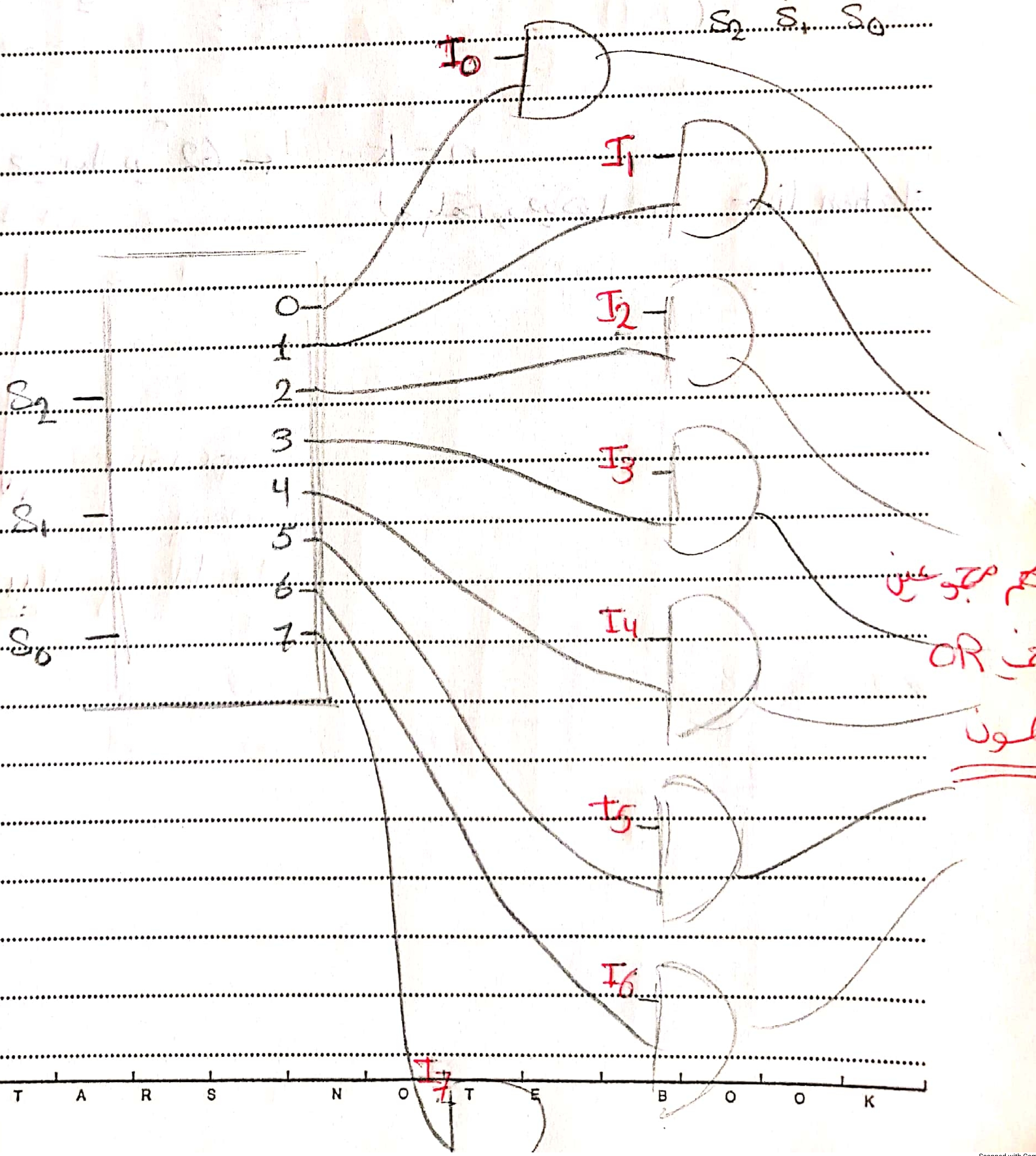
And ← كل احتمال اله
بيشغل ال input تبعه
وبالاخر كلهم بجمعهم على OR

بعدين بشبكهم OR

SUBJECT:

8 : 1 Mux





كلهم مجمعين عـ OR هون

SUBJECT: / /

Approach #1:-

عشان نرسم أي Mux باطريقة ← $2^k:1$ Mux

① $k-2^k$ Decoder (1 single Decoder)

② 2^k 2 input And Gates

③ on 2^k input OR ← حجمها كبير

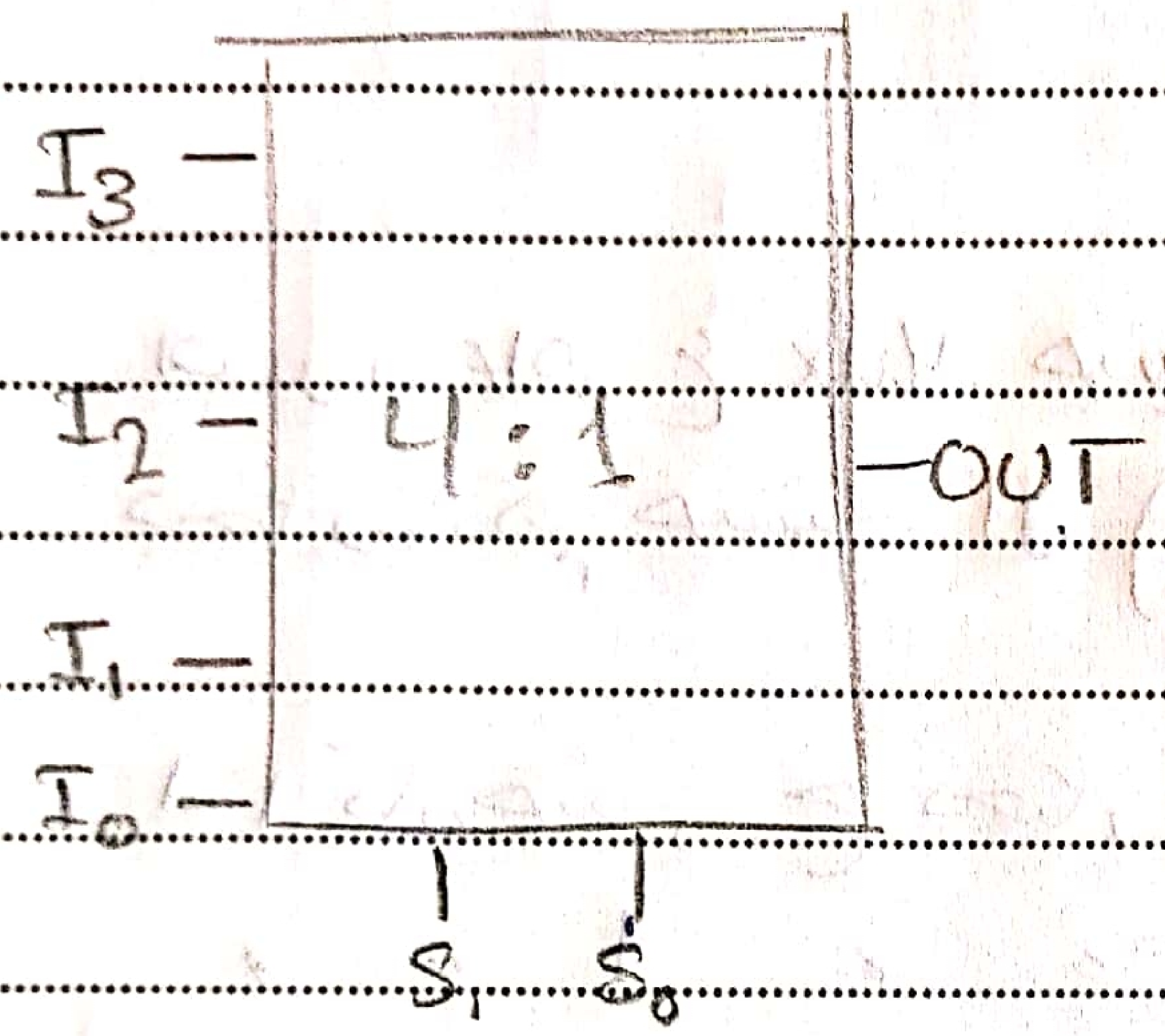
حيث سلايد 62 ← $n=k$

selection line لي هكتب فيها عدد

expensive way of building Mux

Approach #2:-
Another way of building the multiplexer

← يعني عملنا بدال decoder جبنا decoders صغار, و بنينا منهم decoder كبير

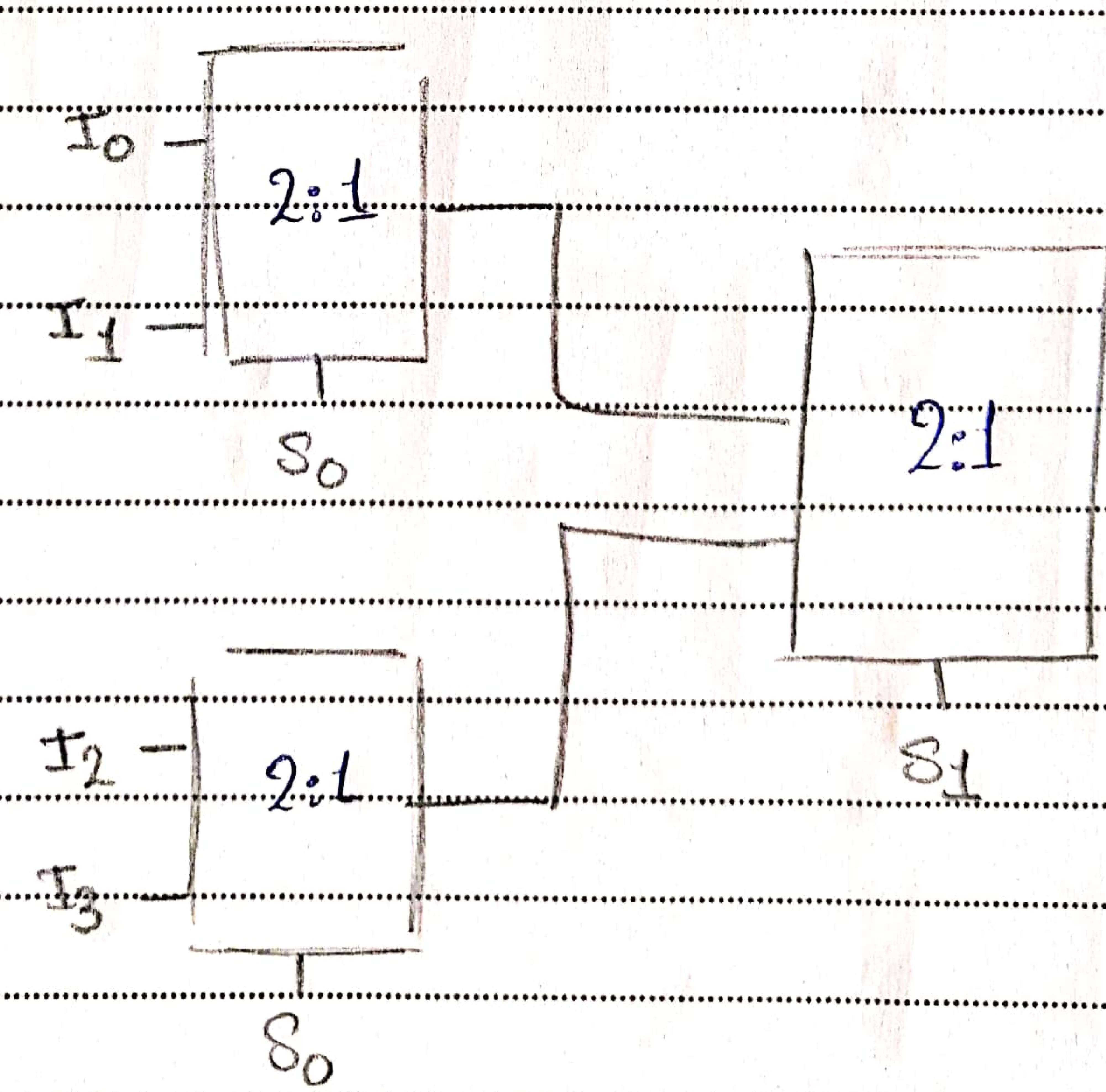


S_1	S_0	OUT
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

$S_1=0$

$S_1=1$

selects the winner



نترجم رسمة طماحنية بالكلام :-

I_0 و I_1 متنافسين بحجن حسب S_0 ← هو اللي S_0 بختار ويحدد الفائز

I_2 و I_3 " " " " حسب S_0 هو اللي S_0 بختار ويحدد الفائز

بعدين بحطهم ع MUX جديد عبارة عن (2:1)
بختار بيناتهم حسب ال ($\underline{\underline{S_1}}$) هي اللي بتحدد

اذا كان صفر ح يختار من فوق (I_0 او I_1)
" " 1 " " " تحت (I_2 او I_3)

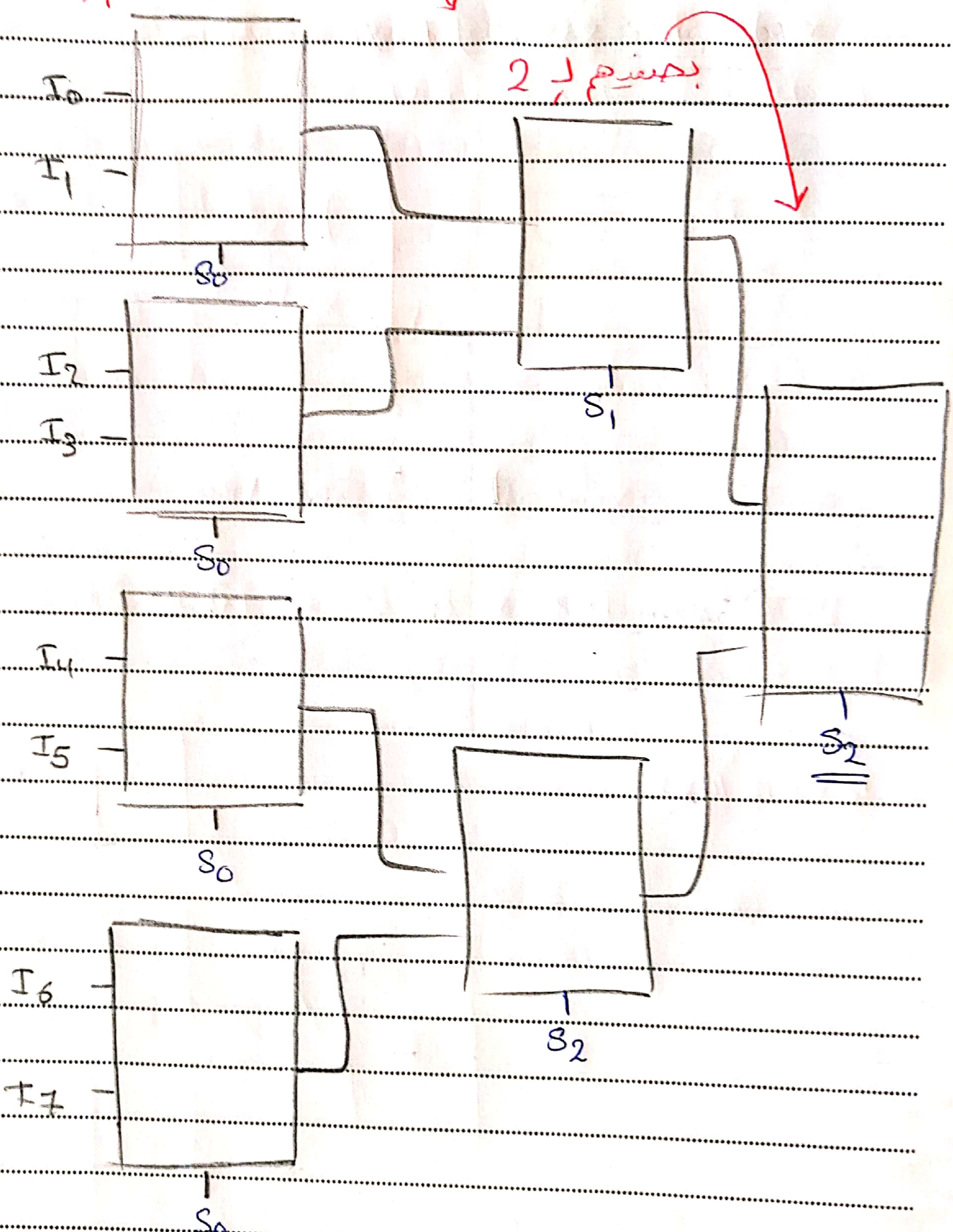
SUBJECT: .... / /

example: 8:1 mux

S_2	S_1	S_0	OUT
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

SUBJECT: .... / /

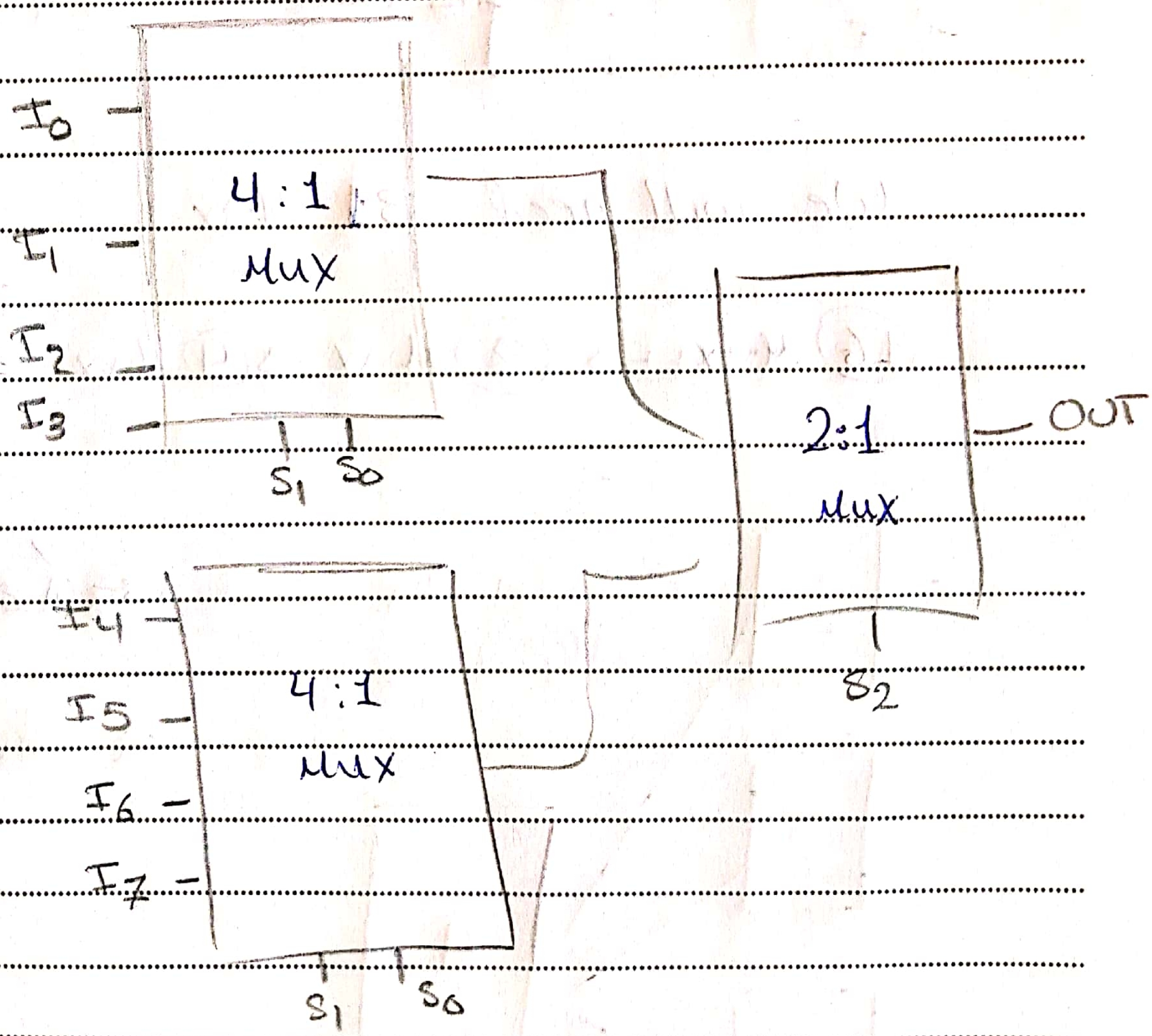
عنا 8 بدمجهم لـ 4



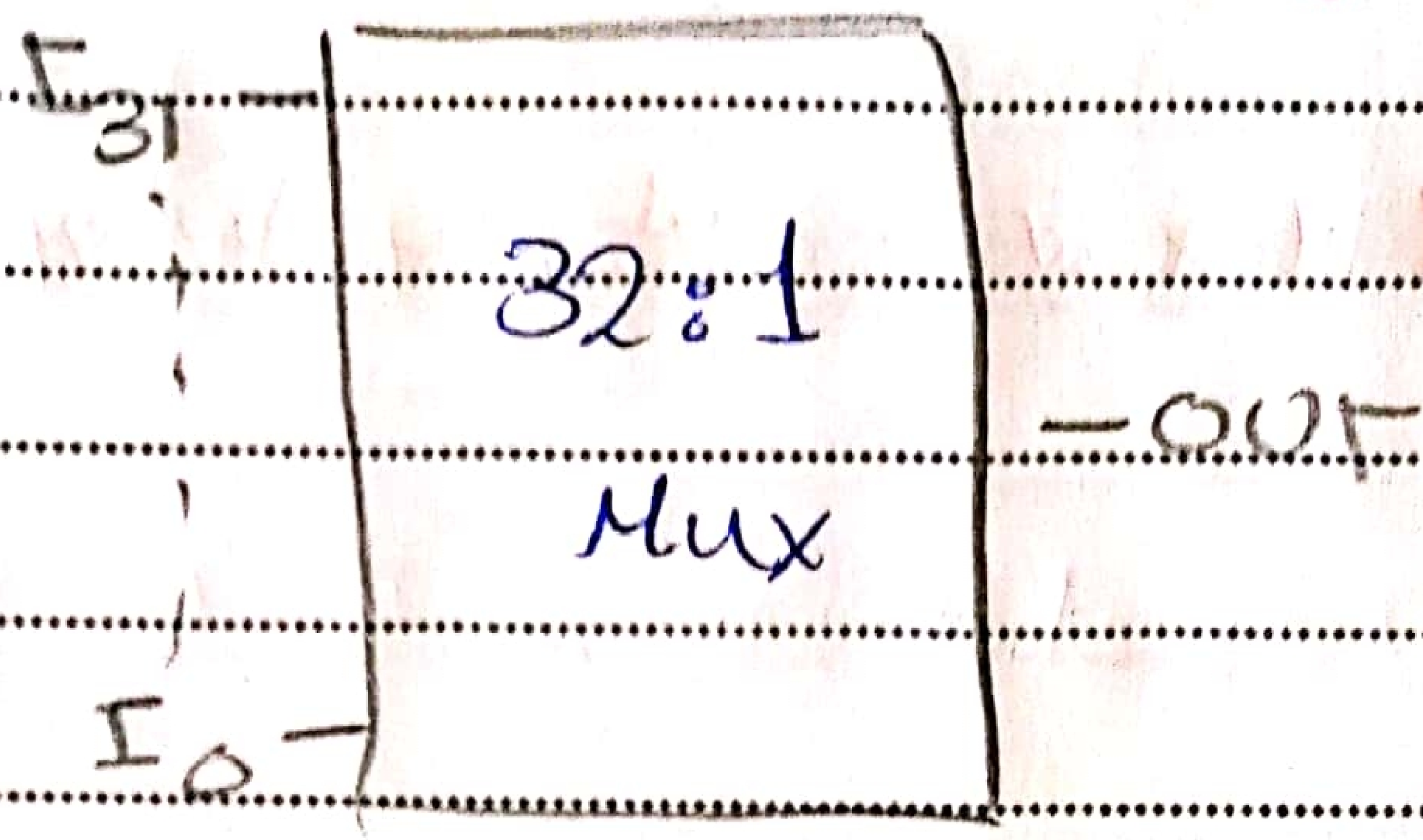
SUBJECT: .... / /

وحسب شو معطيني بالسؤال بختار الحجم ✓
مش شرط اعطيهم ثنائيات

اذا معطيني مثلاً عالحالة هاي 2 → 4:1 Mux و 4:1 Mux



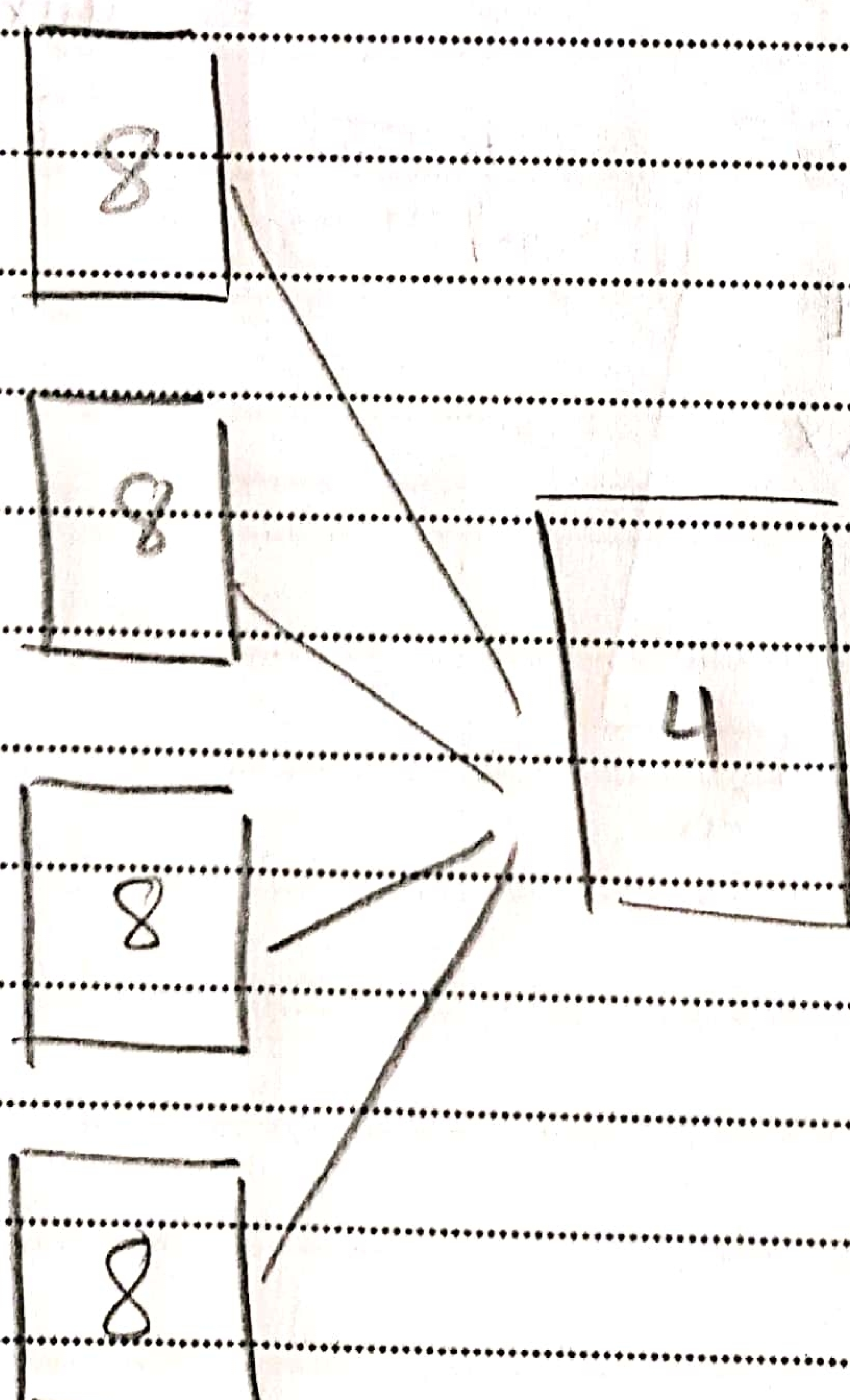
examples: Build ~~[crossed out]~~ 32 : 1 Mux using only 2:1 Mux



We will need 31 Mux

(16) Mux → (8) Mux → (4) Mux → (2) Mux → (1) Mux

أو مثلاً ع شكل ←



SUBJECT: .... / /

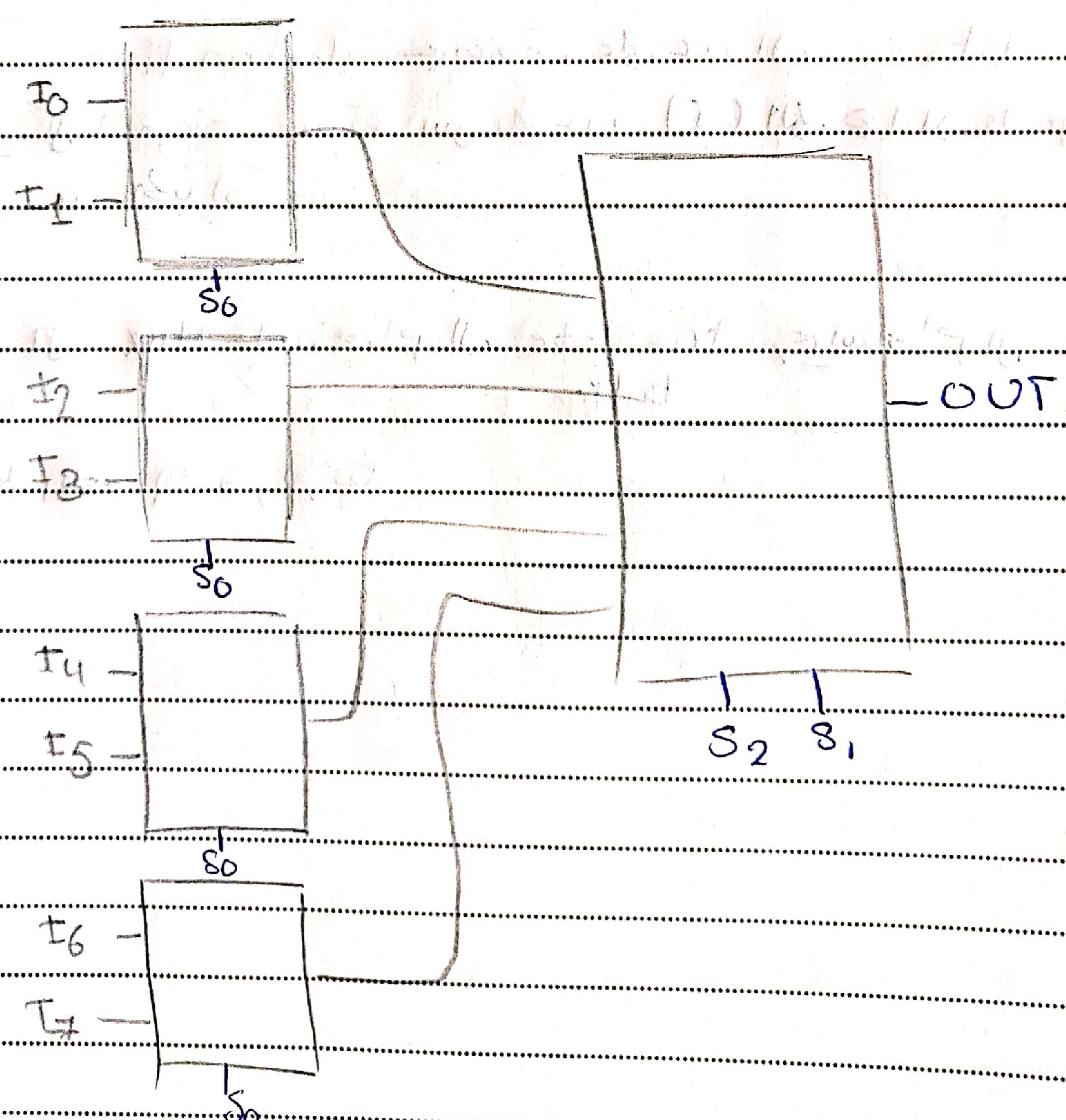
HomeWork :- Slide 66

Build 8:1 mux

① Two 4-1 Mux and one 2-1 Mux
Done in slide 65

③ only 2-1 Muxes (Done previously

② one 4-1 Mux and ~~4~~ Multiple 2-1 Muxes



SUBJECT: / /

في حجم كل signal فيها Quad Multiplexer ← (I) 4 bits

حجم كل (I) فيها ← Dual Multiplexer ← 2 bits

* وبرضو ممكن يكون ← 32-Bits Mux ← كل I ← 32 Bits

ملاحظة :-

ال Selection line ← بعتمد على عدد ال Bits

في كل ال signal بعتمد بس على عدد (I) اللي هي ال signals من عدد الخيارات

* بناء ال Mux باستخدام ال tri-state buffer زي استخدام ال

Muxes ~~...~~ لصفحة تقريباً صفحة 12

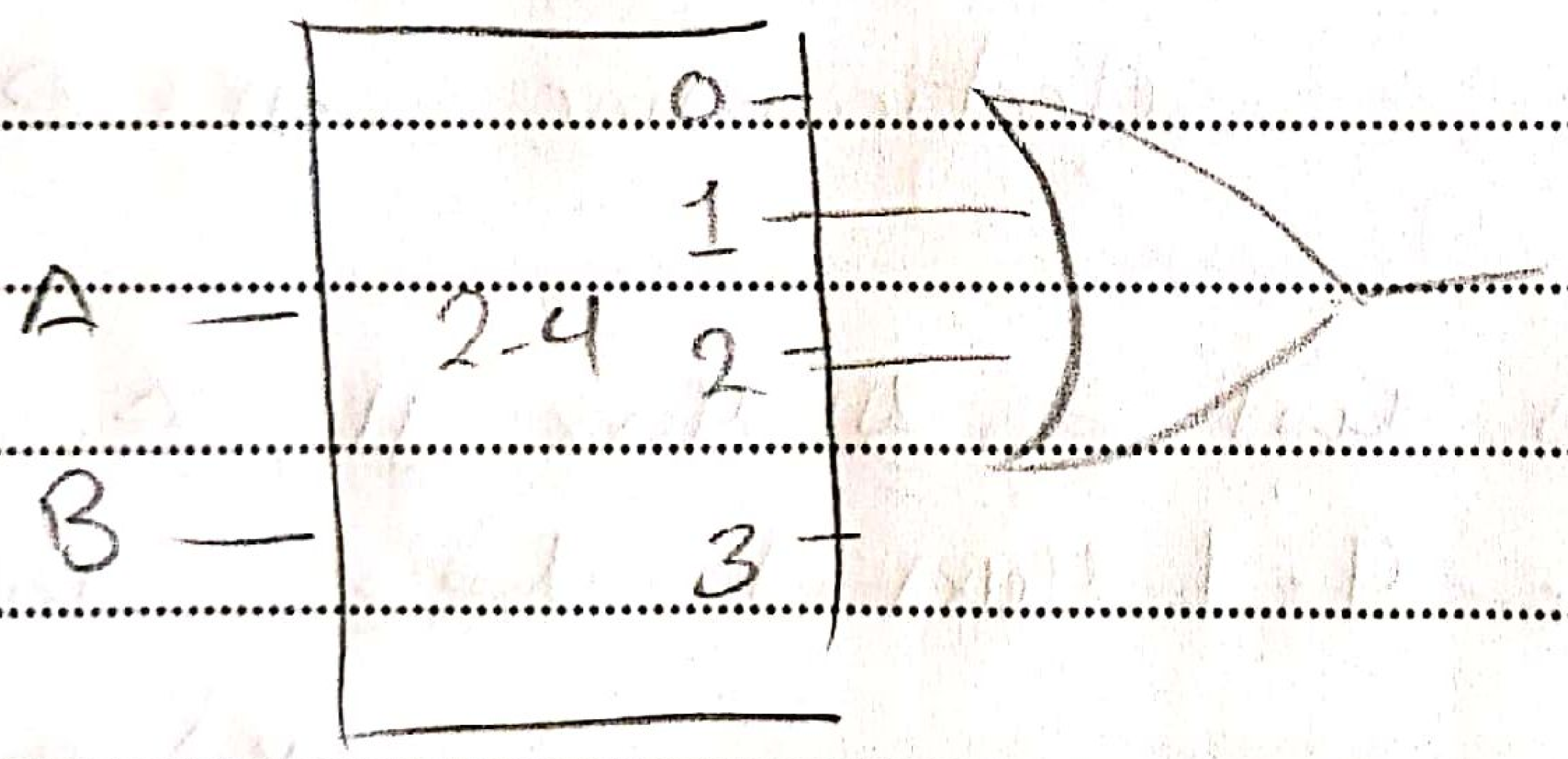
- Exercise 8:1 باستخدام buffer

لبناء أي combinational circuits من الـ truth table

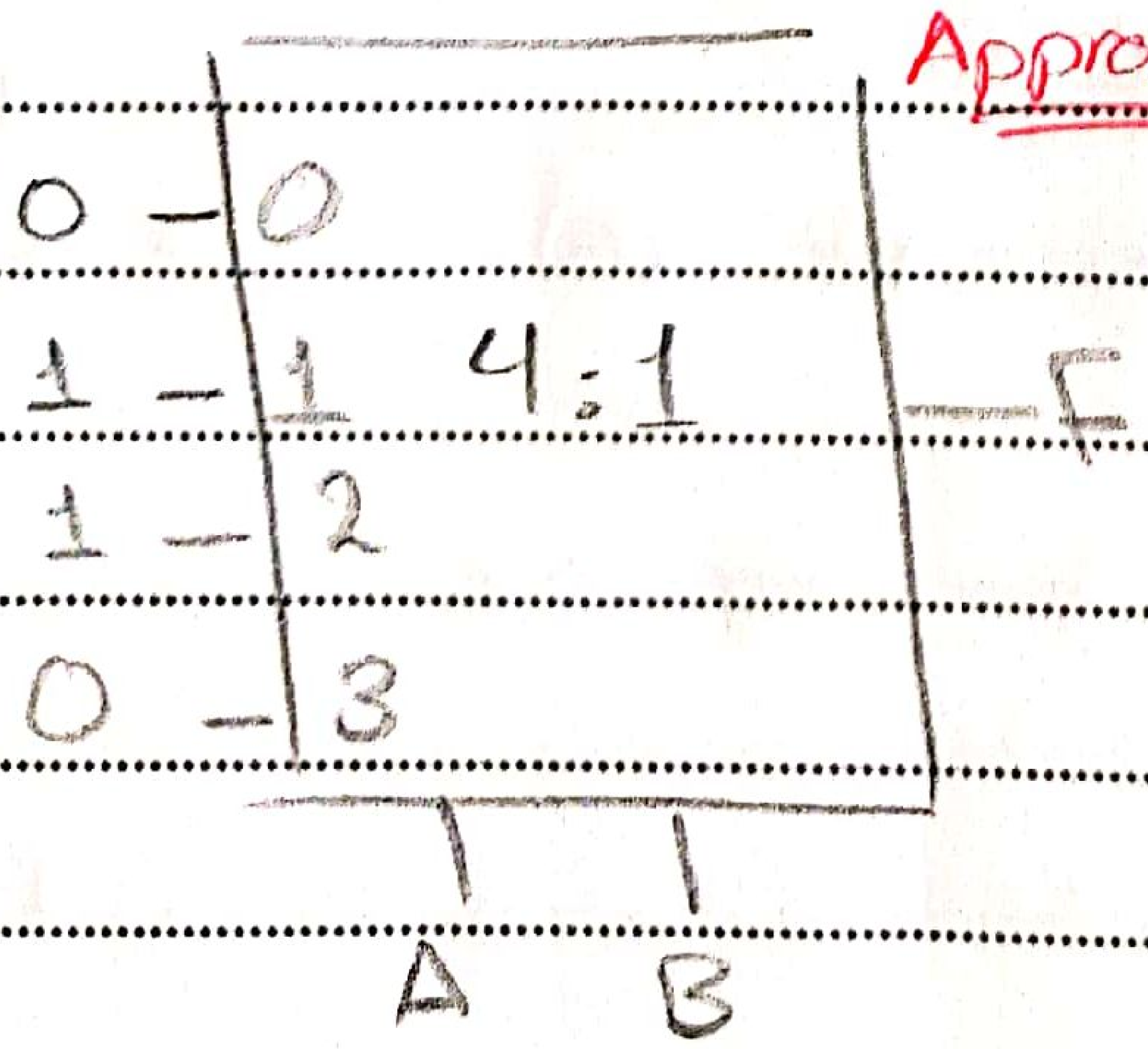
1. بنجيب الـ k-map و بنرسمها بال And و الـ OR والـ NOT

2. ممكن نرسمها بال And, OR, NOT ونحولهم ل NANDS و NORS

3. وممكن استخدم الـ Decoder ← بأني ادخل مجموعة inputs و اشوف وين الـ ones و ادخلهم على OR gate



4. باستخدام الـ Mux ← Approach#1



A	B	F	
0	0	0	I_0
0	1	1	I_1
1	0	1	I_2
1	1	0	I_3

اذا معناها بنقدر نرسم أي Mux بدون truth table ✓

✓ Approach #2 → كأنها tri-state buffer

Slide 72

ملاحظة:
← عند بناء أي Mux داخلياً، ال inputs بتكون ثابتة
مع ال selection line

← حسب حجم ال Mux الي بدي ابنيه بقسم ال truth table ← مثلاً اذا 4:1 Mux ← بقسم الجدول لأربع أرباع

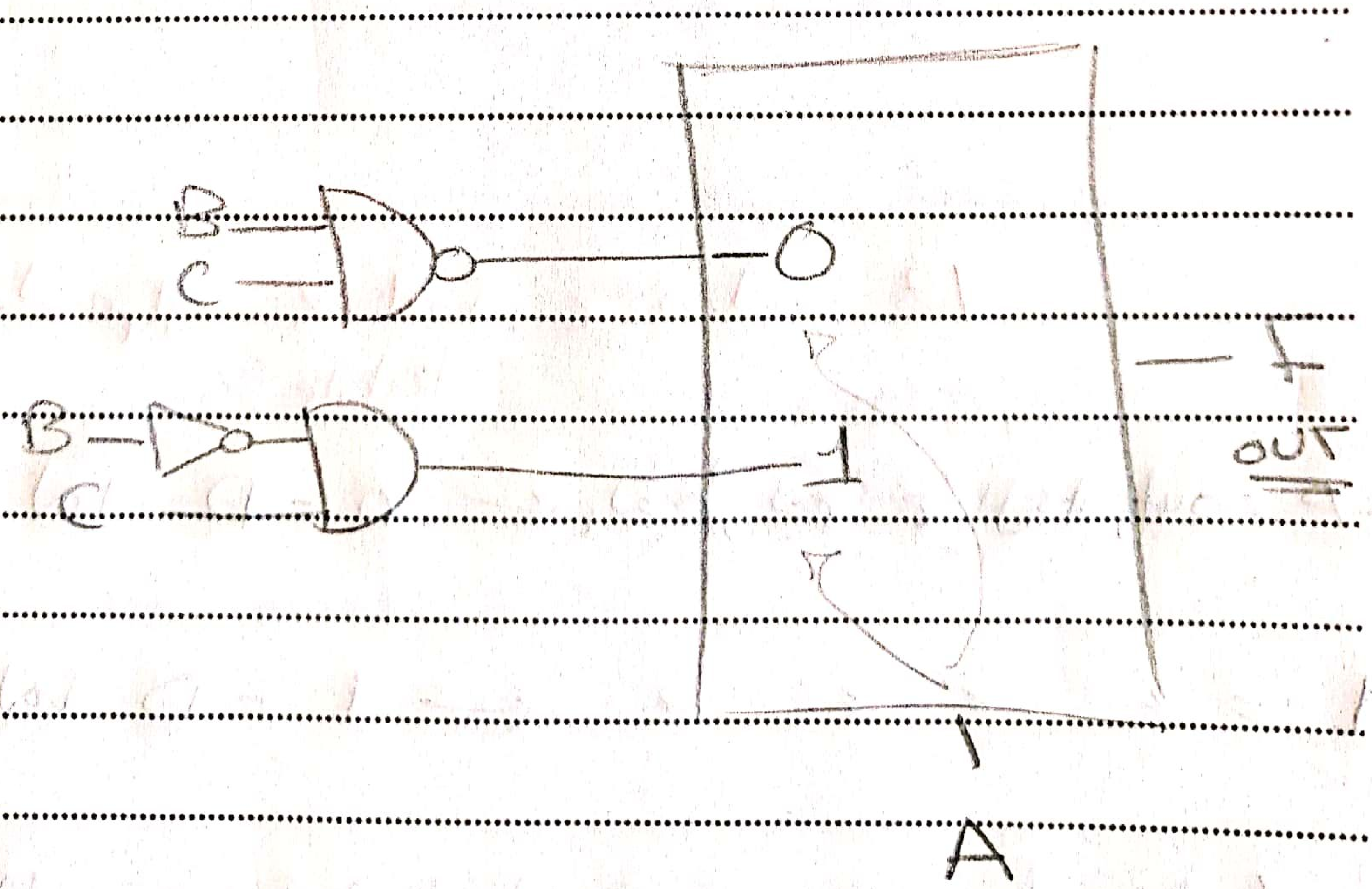
SUBJECT:

example:- Implement the function in slide 76 using 2:1 Mux

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

$\rightarrow F = \overline{BC}$ → احسب المقارنة

$F = \bar{B}C$



Demultiplexer :-

بياخذ input وبعرضو على اي signal بدو يطلعه

	S_1	S_0	D	D_0	D_1	D_2	D_3
0 = (بنتج D_0)	0	0	0 1	D	0	0	0
1 = (بنتج D_1)	0	1	0 1	0	D	0	0
2 = (بنتج D_2)	1	0	0 1	0	0	D	0
3 (بنتج D_3)	1	1	0 1	0	0	0	D

طريقة ثانية لرسم ال DMux table في سلايد 78

كانت فيه لما D = 0 ← لعطي بنتج ح ال outputs = صفر .

لما D = 1 ← 〃 〃 〃 〃 = 1

(بيشبه ال truthtable تبع ال decoder)

SUBJECT:

D	S_1	S_0	OUT			
0	X	X	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

D work as an Enable

So DMUX is as decoder with Enable

Decoders → Does the decoding

Demultiplixer → Does the distribution

the specification is compeletely different, but mathematically are equivalent

New chapter: Arithmetic function CH4

* كل عملية جمع بطلع منها sum وبطلع منها carry

لو الزائد من الجمع عن خانة واحدة (اللي بالايد)

* نفس الطريقة اللي بنجمع فيها الـ decimal بنجمع فيها الـ binary ✓

$$
\begin{array}{r}
1\ \ 1\ \ 0 \leftarrow \text{initial carry} \\
1\ 0\ 1\ 1 \\
1\ 0\ 0\ 1\ + \\
\hline
1\ 0\ 1\ 0\ 0
\end{array}
$$

$10 = 2 \leftarrow$ in binary $\leftarrow (10) = 1+1 \leftarrow$

(in Binary) 5 bits = (4 bits + 4 bits)

(in decimal ✓) 5 digits = (4 digits) + (4 digits)

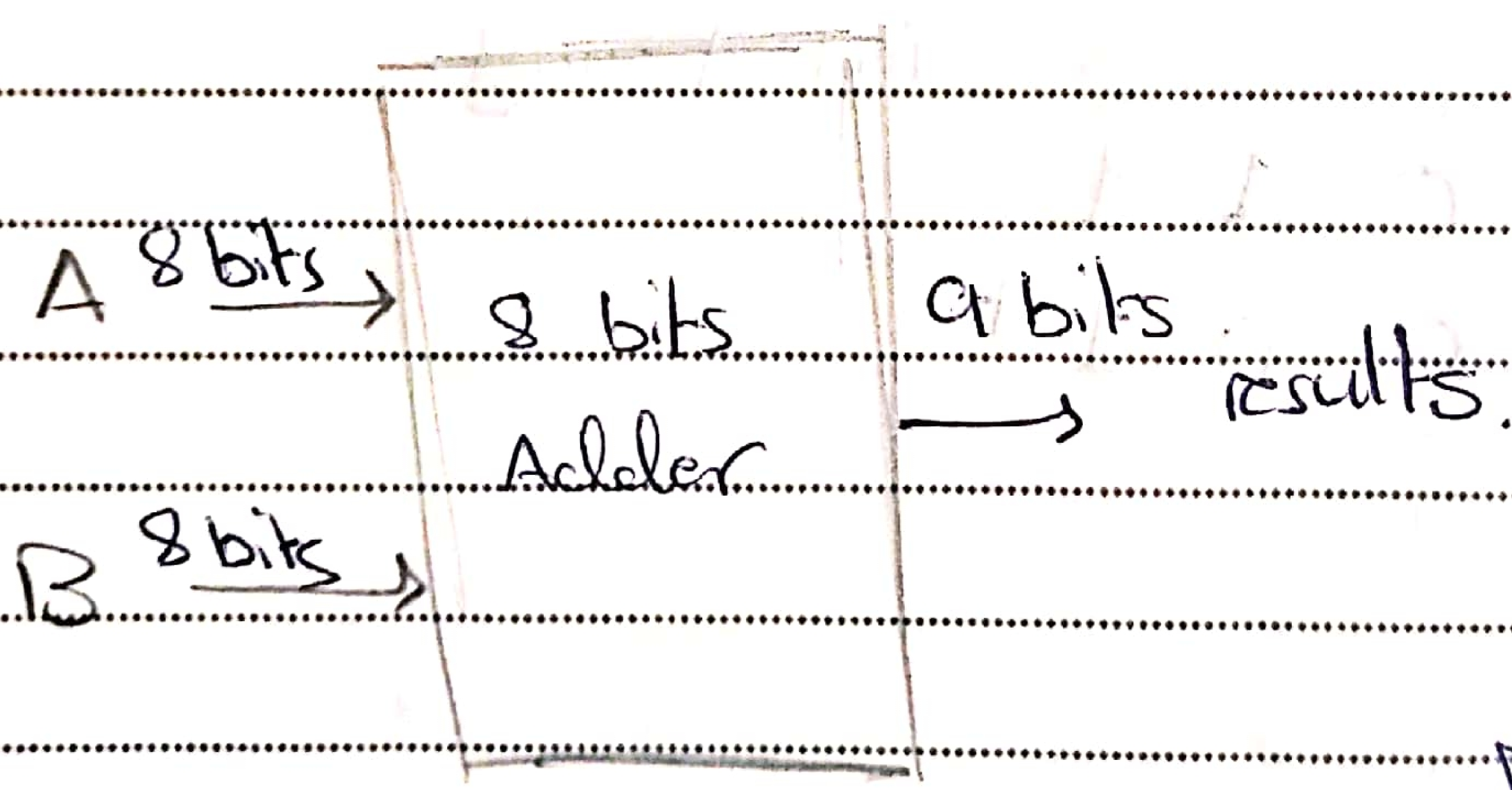
ال ← carry لازم يكون جزء من الجواب ✓

* بيعطيني تصميم أ board وارسم ال circuit تبعته

← بكتب ال Truth table حسب الوصف

← بعمل k-map كي أطلع المعادلة مبسطة ومثالية
وارسم ال circuit

بالطريقة التقليدية هيك بصمم ال [8 bit Adder]



16 Bit input

9 bits result

9 equations

A_7	A_6	---	A_0	B_7	B_6	---	B_0	R_8	R_7	...	R_0
0							0				
1							1				

2^{16} combination

هاي الطريقة عن عملية وعكسها (الطريقة السابقة)

الفكرة : ما في داعي نتبع الطريقة التقليدية لما يكون في تكرار لنفس العملية، زي عملية الجمع

في التصميم السابق او كل ما بيشتغل بس ال Adders بيشتغل كل تصميم فيه تكرار

← بنستخدم هاد التصميم اضطراريا (الطريقة sequential)

بنبسط ال Design لمجموعة full adders

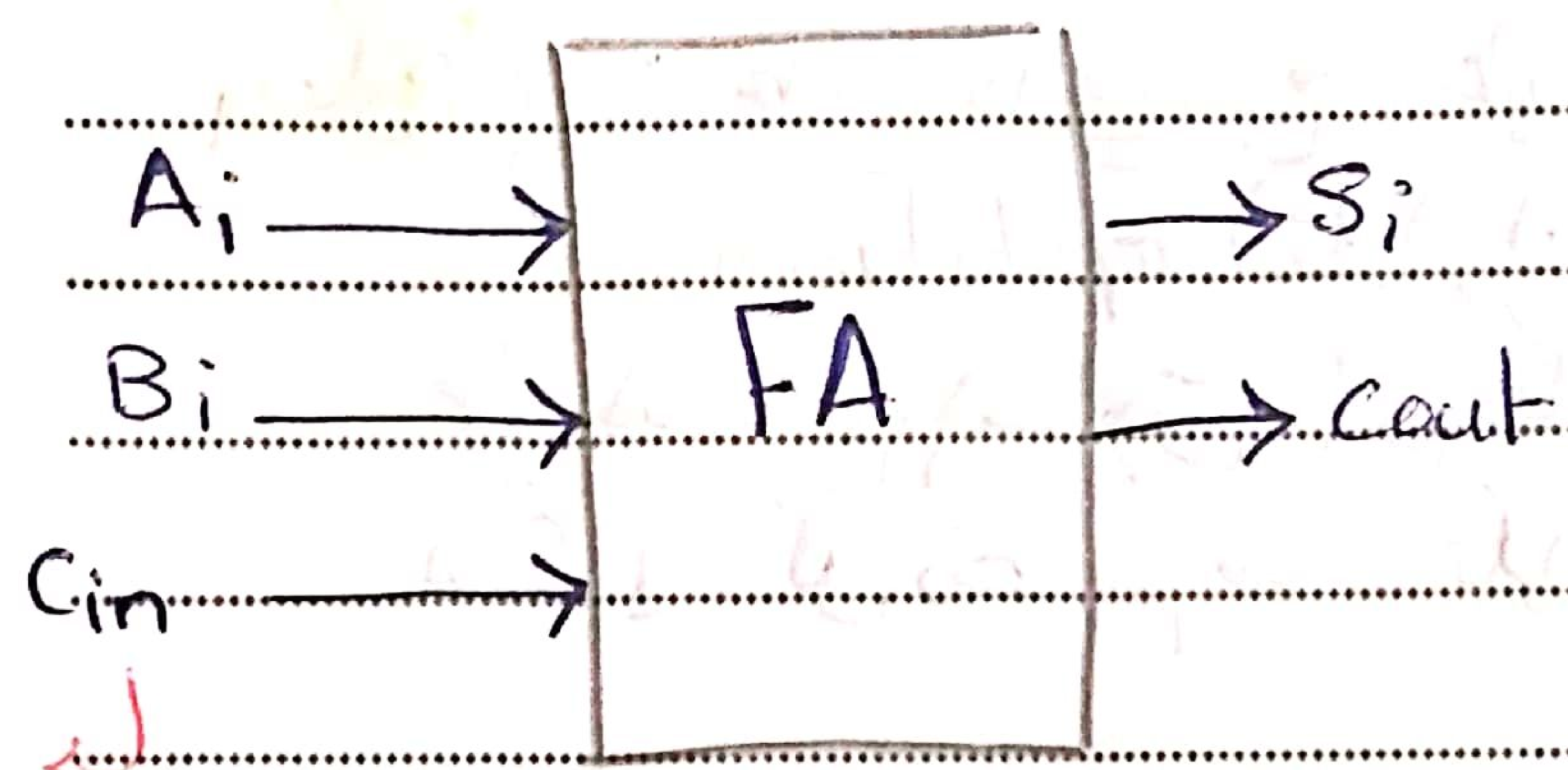
كل واحد منهم 1 bit، وبجيب مجموعة ال results

بطريقة sequential وبشبكهم مع بعض

سلبياتها: طريقة بطيئة بسبب ال Dependency

او ال serialization ✓

1 Bit Adder design (Full Adder)



the carry.

مجرد بنقرأ عليها
بطريقة تقليدية وإجمعها

C_{in}	A_i	B_i	S_i	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Then we make 2 k-maps

SUBJECT: / /

S_i

		A_i	A_i
0	1	0	1
1	0	1	0

(row 2: C_{in}; columns 2–3: B_i)

C_{out}

		A_i	A_i
0	0	1	0
0	1	1	1

(row 2: C_{in}; columns 2–3: B_i)

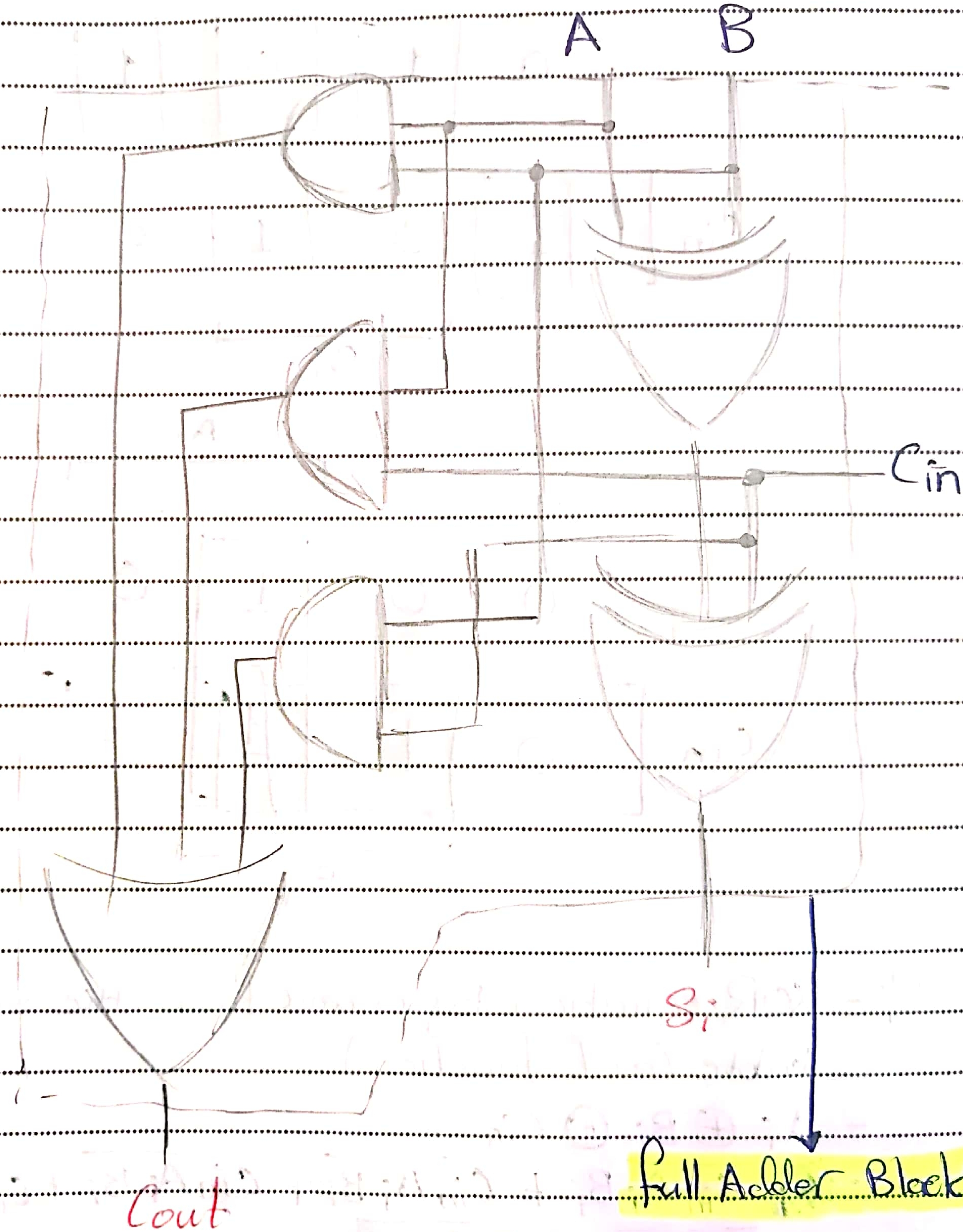
S_i = XOR equation by chance from the truth table (odd function)

$= A_i \oplus B_i \oplus C_{in}$

$= \overline{C_{in}}\,\overline{A_i}\,B_i + \overline{C_{in}}\,A_i\,\overline{B_i} + C_{in}\,\overline{A_i}\,\overline{B_i} + C_{in}\,A_i\,B_i$

$C_{in} = A_iB_i + A_iC_{in} + B_iC_{in}$

full Adder circuit :-

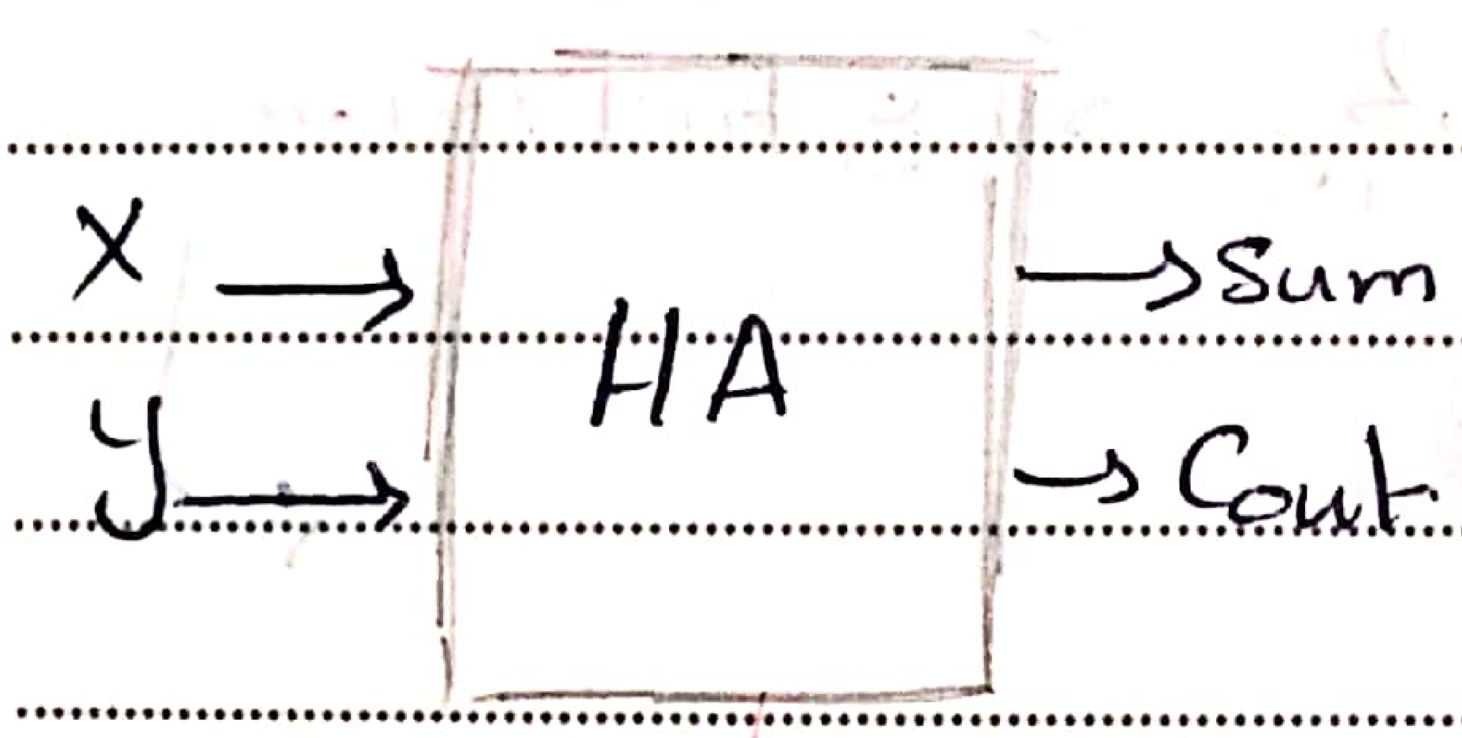


[The Ripple Carry Adder]

وبعدين بنشبك مجموعة ال (FA) مع بعضهم عشان
اعمل ال Adder المطلوب زي سلايد 13

⇒ Why is it called the [Full] Adder?

→ Half-Adder?



		0	0
1	0	1	1
0	1	1	0

+

بصير بجمعهم ثنائي بس ما بجمع ال 3 مرة وحدة
مثلاً بجمع ال carry مع ال A_i اول، بعدين باخذهم بجمعهم مع ال B_i

X	y	S_i	Cout
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$Sum = X \oplus y$] slide 8

$Cout = A$ → The And gate

ال Sum باخذ معادلته ←
XOR function
مش جديدة ✓

* من المقارنة بين تصميم الـ Full Adder في سلايد 11 والتصميم القديم، بنكتشف انه صار اقل تكلفة بسبب شكل المعادلة الجديد اللي سمحلي اشارك (XOR Gate) بحيث استخدم عدد (And gates) اقل

معادلتين متساويين رياضياً، لكن مختلفين كتابياً Cin

علاقة بال HA انه 2 Half Adders تشكلوا معادلة Full Adders في التصميم الجديد (وهو التصميم الاخير)

2HA + OR Gate

$\Downarrow$

1FA

Unsigned Subtraction

→ In subtraction you start from the LSF and you borrow

When you borrow in decimal you take (10)

→ when you borrow in binary you take (2) which is (10) in binary ✓

$$
\begin{array}{r}
\overset{0}{\cancel{1}}\ \overset{\overset{1}{\cancel{10}}}{\cancel{0}}\ \overset{10}{\cancel{0}}\ 1 \\
0\ 1\ 1\ 1\ - \\
\hline
0\ 0\ 1\ 0
\end{array}
$$

* But when subtract reversely :-

$$
\begin{array}{rl}
\cancel{1}\ \overset{10}{0}\ 1\ 1\ 1 & \leftarrow N \\
1\ 0\ 0\ 1\ - & \leftarrow M \\
\hline
1\ 1\ 1\ 0 &
\end{array}
$$

← استلاف وهمي

⇒ يعني كأنو رقم (0111) ضفت عليه 1 (10111) ال 1 هاي قيمتها ← (2^4) لأنها في خانة خامسة ← اللي هي 2^n لما كنا نحول من binary لـ decimal ✓

$$= \underbrace{(N-M) + 2^n}$$

← بيمثل معنى الطرح بال decimal

← هاد الناتج خطأ لعملية الطرح لكن

(الـ صحيح)

عشان اتخلص منها بطرحها اطرح :-

Do the correction:-

$\to 2^n - ((N-M) + 2^n)$

$2^n \to 2^4 = 16 \to 10000$; هادا من المثال السابق

$$
\begin{array}{r}
\ 1\ \ 1\ \ \\
0\ \cancel{10}\ \cancel{10}\ 10\ \\
\cancel{1}\ \cancel{0}\ \cancel{0}\ 0\ 0 \\
0\ 1\ 1\ 1\ 0 \\
\hline
-\ 0\ 0\ 0\ 1\ 0
\end{array}
$$

✓

final answer ✓

$\cancel{2^n} - (N-M) - \cancel{2^n} = -(N-M)$

$-10 = -(N-M)$

to summarize:-

$N - M$

→ end borrow = 0
positive answer $(N \geq M)$

→ end borrow = 1
negative answer
A correction should be applied ✓

SUBJECT: ....

Example:-

$$\begin{array}{r} 1\;0\;0\;0\;1\;0 \\ -\;0\;1\;1\;1\;0 \\ \hline 0\;0\;1\;0\;0 \end{array}$$

⇒ positive answer case

$$\begin{array}{r} 1\;0\;1\;1\;1\;0 \\ -\;1\;0\;0\;1\;0 \\ \hline 1\;1\;1\;0\;0 \end{array}$$

→ we added 2^5 in binary

$2^5 - ((N-M) + 2^5)$

$$\begin{array}{r} 1\;0\;0\;0\;0\;0 \\ -\;0\;1\;1\;1\;0\;0 \\ \hline -\;0\;0\;0\;1\;0\;0 \end{array}$$

correction ✓

final answer = - 100

Exercise slide 17:-

(1)

```
  10010110
  01100100 -
  --------
  00110010
```

(2)

```
 101100100
  10010110 -
  --------
  11001110
```

the correction:-

```
  100000000
  011001110 -
  ---------
- 000110010
```

final answer is (- 110010)

The Complements [م/ لؤي ملكه]

two types

① Diminished Radix $(r-1)$'s

② Radix r's

لما يطلب مني احسب ال comp. كأنه بيطلب قديش بدي
عشان كل digit يوصل ال max

example: [for $(r-1)$'s complement]

① solve the complement for $(34)_{10}$

9's complement ?

9's $\rightarrow (99)_{10}$

قديش بدي عشان اوصل ال
$(99)_{10} \leftarrow (34)_{10}$ لـ

$5 \leftarrow 4$

9's comp. $= (65)_{10}$ $\leftarrow$ $6 \leftarrow 3$

② 7's comp. for $(71)_8$

max $= (77)_8$ 71 ←

$(06)_8$

③ 9's comp. for $(114)_{10}$

max is $(999)_{10}$ → 9's comp. $= (885)_{10}$

④ 1's comp. for $(1010)_2$

max is $(1111)_2$ → 1's comp $= (0101)_2$

special case with binary
you always flip digits
to find (r-1)'s comp.

⟹

as you invert bits
so the comp. in binary
is found by inverters

⑤ 15's comp. for $(38A)_{16}$

max is $(FFF)_{16}$ → comp $= (C75)_{16}$

SUBJECT: .... / /

→ you can use the rule in slide 19 to solve the Diminished Radix comp.

بس هالقانون عشان اطبقه لازم تكون كل الارقام محوّلة بال decimal

حول الرقم N لـ $(2^n - 1 - N)$

من اي نظام لـ decimal ، بحسب بعدين برجع للنظام الاصلي ✓

For Radix Complement

بحسب ال diminished comp. ونضيف (1) بس ✓

examples:-

① $(34)_{10} \xrightarrow{10's} (66)_{10}$

$9's = (65)_{10} + 1$ ⇒ او طبق لقاعدة سلايد

② $(38A)_{16} \xrightarrow{16's} (C76)_{16}$

$15's = (C75)_{16} + 1$

③ $(1011)_2 \xrightarrow{2's} (0101)_2$

$1's = (0100)_2 + 1$

④ $(0111\ 0011)_2 \xrightarrow{2's} (1000\ 1101)_2$

$1's = (1000\ 1100)_2 + 1$

SUBJECT:

⑤ $(010100)_2 \xrightarrow{2's} (101100)_2$

$1's = (101011)_2 + 1$

ل راح نستخدم ال carries عشان ارجع

الطريقة لسريعة لحساب ال 2's ←

لمثلاً :-

1 0 0 1 0 1 0 0

2's 0 1 1 0 1 1 0 0

لمشي وبلش من اقصى اليمين لحد ما اوصل اول (1)

بثبت الارقام هاد لحد ما اوصلت اول (1)

بعد هيك بعكس كلشي بعدها

علاقة ال complement بالطرح

$N - M = N + 2's \text{ of } M$

ل هالشي بساعدني اصمم circuit عشان طرح من circuit للجمع

واذا طلع معاي carry زيادة اهمله

مثال

ال carry الزيادة هاد بيكون

$$
\begin{array}{r}
1011 \\
0111\ - \\
\hline
\end{array}
\longrightarrow
\begin{array}{r}
1\ 1011 \\
1001\ + \\
\hline
0100
\end{array}
$$

2's of M

① حالة ال positive answer ✓
بكون ال carry الزيادة 1
الكبير - الصغير

عشان نعمل circuit لطرح بدل circuit بيحول ل 2's بجيبه بجمع

العلاقة الي راح تساعدني احسب ال 2's

$$2\text{'s of } M = 1\text{'s of } M + 1$$

مثال

ال carry الزيادة

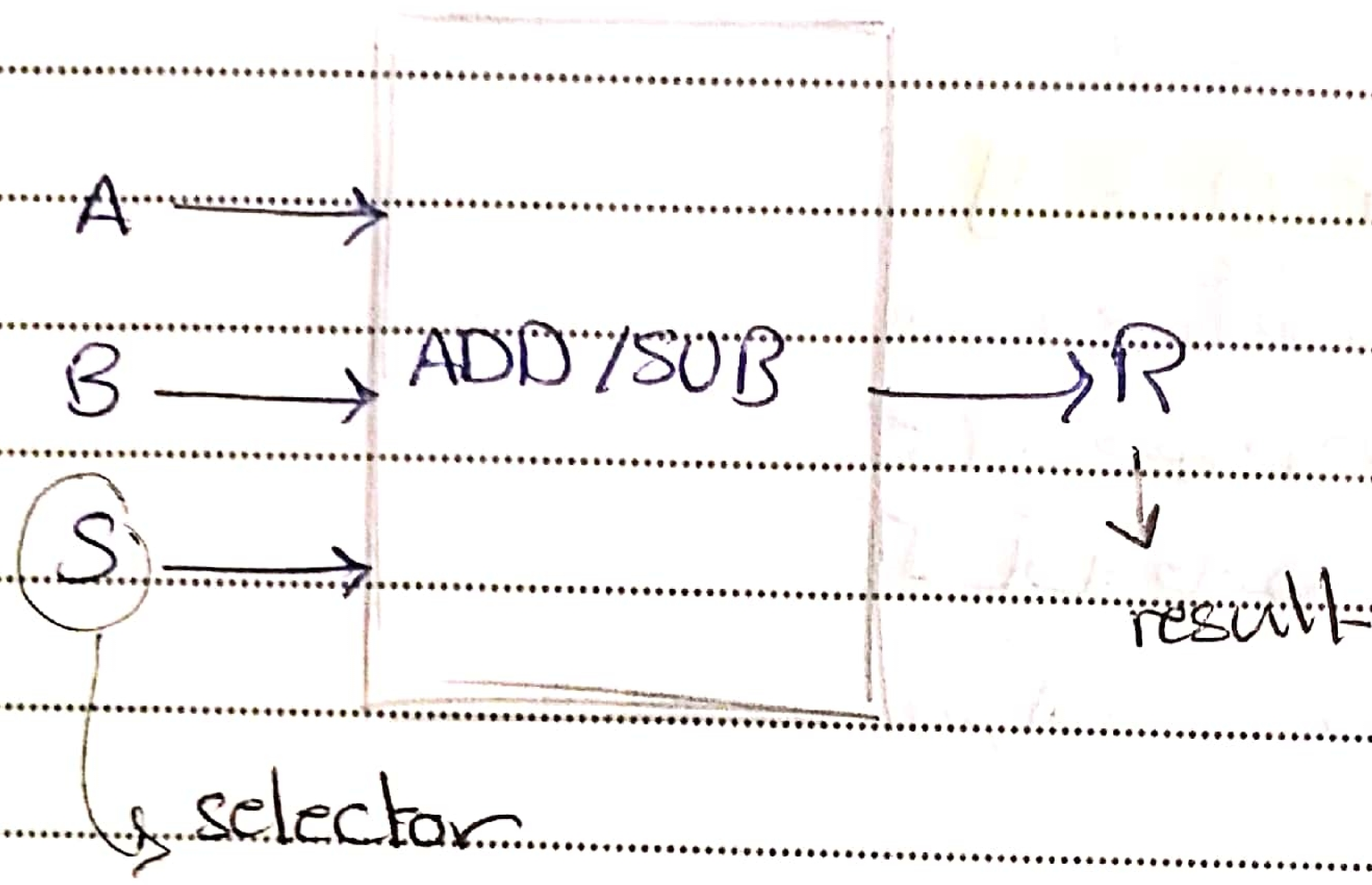
$$
\begin{array}{r}
0111 \\
1011\ - \\
\hline
\end{array}
\longrightarrow
\begin{array}{r}
0\ 0111 \\
0101\ + \\
\hline
1100
\end{array}
$$

لازم نعمله correction

لما حوله ال 2's وحطه سالب -(0100)

② حالة ال negative answer ✓
بكون ال carry الزيادة 0
الصغير - الكبير

to design ADD/SUB circuit:-



بصير التصميم رح يجي بسلايد [29]

تفسير التصميم :-

when S = 0 → implement addition
" S = 1 → " subtraction

رح نعتمد على ال XOR Gates ، وال Ripple-carry design

وحسب حابدخل قيمة لـ S بصير التنفيذ

لما $(S=0)$ ، بيعمل جمع

كيف تأثيره على ال XOR حسب المطابقة

$y \text{ xor } 0 = y$

يعني ملغي تأثير ال xor

كأنه مش موجود وبترجع السيركت

كأنه مافيها xor

سيركت Adder عادية

لما $(S=1)$ ، بعمل طرح

لما بعمل جمع احنا بنعمله هيك ← $A-B = A + 2\text{'s } B = A + 1\text{'s } B + 1$

$= A + \bar{B} + 1$

because $1\text{'s } B = \bar{B}$ ←

فهون بكون تأثير S عال XOR حسب المطابقة

$y \text{ xor } 1 = \bar{y}$

فتدخل ال $(S=1)$ على كل ال XOR وينتج عنها معكوس كل مدخلة

بتدخل معها ، يعني

$B_0 \oplus 1 = \bar{B}_0$ $\quad$ $B_2 \oplus 1 = \bar{B}_2$

$B_1 \oplus 1 = \bar{B}_1$ $\quad$ $B_3 \oplus 1 = \bar{B}_3$

← فانا هيك جبت اول جزء من المعادلة اللي هو 1's B
واللي بعادل $\bar{B}$ وبيدخل عال FA

← بيدخل ال A عليها (عال FA)

← ال 1 بيدخل عال (FA) من ال C_0 اللي بتعبر ال carry

وبتحقق المعادلة بالترتيب

$$
\begin{array}{ccccc}
A_i & + & 1\text{'s } B_i & + & 1 \\
\downarrow & & \downarrow & & \downarrow \\
A_i & & \bar{B}_i & & S
\end{array}
$$

$$
\begin{array}{ccccc}
 & A_3 & A_2 & A_1 & A_0 \\
 & \bar{B}_2 & \bar{B}_2 & \bar{B}_1 & \bar{B}_0 \\
 & C_3 & C_2 & C_1 & C_0 \\
\swarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
C_4 & S_3 & S_2 & S_1 & S_0
\end{array}
$$

C_4 controls whether the result is (+ / −)

→ if $C_4 = 1$, positive result (does not need correction)
→ and it will be discard

→ if $C_4 = 0$, the result needs correction, and will be negative

[SIGNED NUMBERS]

بالوضع الطبيعي لما كنا نحكي عن unsigned numbers كانت ال

[$max = 2^n - 1$] and [$min = 0$]

ولكن هالمرة راح نختلف مع ال Signed Numbers لأنه راح ناخذ bit تحدد هل هو موجب او سالب

Signed magnitude:-

→ take the most significant figure and turn it into 0 if you are representing positive number, and 1 if it is a negative number.

sign / magnitude

+3 → 0 0 1 1

-3 → 1 0 1 1

+7 → 0 1 1 1

-7 → 1 1 1 1

صار عندي ال 4 bits

min = -7 = 1111

max = +7 = 0111

لأنه في bit بطلت تمثل ال magnitude

So $max = +(2^{n-1} - 1)$

$min = -(2^{n-1} - 1)$

~~Binary~~

Signed 1's complement:-

ال 1's complement هي بيمثل الارقام السالبة

الارقام الموجبة تمثيلهم هو نفسه بأي نظام لتمثيل ال signed

حط 0 يمين ال magnitude ا

وللرقم السالب خذ ال 1's لل الرقم الموجب

+23 → 0 1 0 1 1 1

-23 → 1 0 1 0 0 0

1's

0 1 0 1 1 1

م بيصح اني استخدم نظام ذاتي

لكن ال 1 الي بتعبر عن السالب موجودة

Signed 2's complement:-

+23 → 0 1 0 1 1 1

-23 → 1 0 1 0 0 1 ⇒ 2's of (+23)

هاي هي الطريقة المستخدمة في الكومبيوترات الحديثة لتمثيل الارقام ال signed وهي احسن وحدة.

example :-

$(+30)_{10} \xrightarrow[\text{complement}]{\text{signed 1's}} (011110)_2$

$(-50)_{10} \xrightarrow[\text{magnitude with 8 bits}]{\text{signed}} (10110010)_2$

$(-50)_{10} \xrightarrow[\text{signed 2's comp.}]{\text{8 bits}} (11001110)_2$

$+50 = 00110010 \rightarrow$ 2's comp. of $(+50)$

$(1011)_2 \xrightarrow[\text{2's comp.}]{\text{signed}}$ هنا ال 2's كمان مرة عشان نرجع للقيمة الاصلية

رقم سالب

2's of $(1011) = (0101)_2$

$(-5)_{10} =$

$(0011)_2 \xrightarrow[\text{2's comp.}]{\text{signed}} (+3)_{10}$

رقم موجب

$(1011)_2 \xrightarrow[\text{magnitude}]{\text{signed}} (-3)_{10}$

رقم سالب

← حتى لما نحسب ال range بيتحدد بين السالب والموجب

من ال min وال max بنطلع عدد الcombinations بوصل زي ما هو

← اذا 4 bits ← # of combinations $= 2^4 = 16$

لكن بالحقيقة combination راح نخسرهم اللي هو (-0)

لأنه (-0 = +0) فالمجموع بيصيروا 15
↓
combination

← هادا لحكي لما احكي عن ال signed magnitued وال 1's comp.

لكن في ال 2's comp.

ما بنضيع ولا ال bit بنضل 16 ← combination

بسبب وجود ال (-8) حسب جدول سلايد 33

← بنصير ال ← Range [+7 — -8]

$max = +(2^{n-1} - 1)$

$min = -(2^{n-1})$

-8

لذلك هو احسن نظام

SUBJECT:

ال signed magnitude وال 1's comp الهم نفس قوانينا

ال min وال max

← بستخدم N-M = N + 2's of M

لأنه في ال 2's ما بضيع bit واحنا بنطلع الارقام

example: (addition)

①	0	0	0	0	0	1	1	0
ارقام موجبة ←	0	0	0	0	1	1	0	1 +
	0	0	0	1	0	0	1	1 =

حسب هاي الطريقة اعمل دائماً discard ل ال carry الاخير وراح يطلع الجواب دائماً صحيح.

②	0	0	0	0	0	1	1	0	← موجب
	1	1	1	1	0	0	1	1 +	← سالب
سالب ←	1	1	1	1	1	0	0	1 =	

in 2's comp.

③	1	1	1	1	1	0	1	0
discard the end carry.	1	1	1	1	0	0	1	1 +
	1	1	1	0	1	1	0	1 =

for subtraction use

$N - M = N + 2\text{'s of } M$

with the end carry discard

and the result taken always without correction.

وبنطبق هالحكي على ال circuit في سلايد 38

ال c_4 ← راح تكون دائماً discard

ولما نجمع او نطرح رقمين 4 bits راح يصير دائماً 4 bits

Overflow:-

الحكي الي قبل ممكن يخلينا بس نجمع رقمين موجبات يطلع سالب

لأنه آخر bit بتصير (1) وال end carry بنهمله (discard)

واحياناً اذا جمعنا رقمين سالبات بيطلع موجب

وصار هيك لأنه عدد ال bits عـ الجواب ما كفوا يعبروا عنه

← لما نجمع أرقام راح نحط ناتج جمع الكري مع ال bits المجموعة

```
  0 1 1                       1
  0 1 1 1   +ve            1 0 0 1   -ve
  0 1 1 0 + +ve            1 0 1 0 + -ve
  -------                  -------
  1 1 0 1   -ve            0 0 1 1   +ve
```

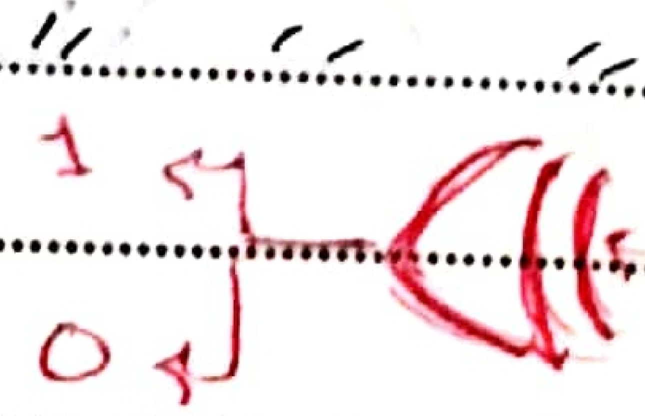
So the circuit should be able to detect the over flow

the computer makes a XOR for the 2 last carries

if the result is 1 → overflow

" " " = 0 → no overflow

[the programmer solve this problem.]




```
  0 1 1 1 1
   0 1 0 0 1 0
   0 0 1 1 1 1 +
   -----------
   1 0 0 0 0 1
```

here is the over flow

SUBJECT:

← لما نجمع رقم موجب مع سالب مستحيل يصير overflow

examples:

$$\begin{array}{r} {\scriptstyle 1\ 11\ \ 1} \quad \\ (11110011)_2 \\ +\ (11110000)_2 \end{array}$$

11100011 = (correct answer)

1 ⊕ 1 = 0 (no overflow)

what is overflow bit ? (0)

So always] →

→ N + M = N + M
→ N − M = N + 2's M
→ discard end carry
→ detect overflow by adding the XOR gate.
Only for signed numbers

example:-

$$
\begin{array}{r}
{\scriptstyle 1\ 1} \\
1011 \\
1000\ + \\
\hline
0011
\end{array}
$$

→ signed 2's comp.

0011 → answer

overflow bit = 1

for unsigned numbers

the overflow bit is always the end carry

if the bit = 1 → overflow ✓
" " " = 0 → no overflow

example:

signed numbers
is there an over flow?

```
 0 1 1 1
 0 0 1 1 0 1
 0 1 1 1 0 0  +
 ------------
 1 0 1 0 0 1
```

overflow bit = 1 → overflow ✓

Other Arithematic Function

multiplication and division in binary:-

①

```
   1011
    101 x
  ------
   1011
  00000 +
 101100 +
 ------
 110111
```

②

```
   111
   010 x
  -----
   000
  1110
 00000
 -----
 01110
```

③

```
        00101
    110|11110
        110   -
        ----
         110
         110
        ----
         000
```

answer = 101

4

```
         00101.11
    101|110111
        101    -
        ------
        00111
          101  -
        ------
        01000
         0101  -
        ------
         00110
           101 -
        ------
           001
```

answer is = 101.11

[special case] multiplying by powers of 2

زي لما اضرب اي رقم في ال decimal برقم $(10)^n$

$A \times 2^n = A\,\underbrace{00\ldots0}_{}$

لعدد من n

$101 \times 100 = 10100$

$100 \rightarrow 2^2$

$\Rightarrow$ this is called shifting left

[shifting right] نفس الشي بالعكس

if you are dividing by 2^n shift to right by n digits to design a multiplier:

a circuit that multiply any number by 5

$B_3 B_2 B_1 B_0 \rightarrow \boxed{\times 5} \rightarrow C_5 C_4 C_3 C_2 C_1 C_0$

كيف عرفنا انهم 6 bits

$15 \times 5 = 75$

max num of 4 bits represented by B

needs 6 bits 2^6

then design it by ripple carry adder using the following relationships

$$
\begin{array}{cccccc}
 & & B_3 & B_2 & B_1 & B_0 \\
 & & & 1 & 0 & 1 \quad \times \\
\hline
 & & B_3 & B_2 & B_1 & B_0 \\
 & 0 & 0 & 0 & 0 & \\
B_3 & B_2 & B_1 & B_0 & & \\
\hline
C_5 & C_4 & C_3 & C_2 & C_1 & C_0
\end{array}
$$

$C_0 = B_0$

$C_1 = B_1$

$C_5 C_4 C_3 C_2 = 00B_3B_2 + B_3B_2B_1B_0$

لما بدنا نكبّر رقم بنحط اصفار ع شماله ← ~~Zero fill~~

Zero fill

عملنا ذا ع يمينه بيتغير هي عمليه غير مسموحة
لأنها بتغير ع الأرقام

بس هاد حكي صحيح لو كان الرقم فقط unsigned
لأنه لو كان عندي رقم سالب وضفت
على يساره اصفار بحوله لرقم موجب وهاد غلط

For signed numbers:—

positive numbers : fill it with zeros

negative numbers : fill it with ones

SUBJECT: .... / /

[New Chapter: ch5]

next state → is called also new state
current/present state → is called also old state

Types of Sequential circuit ← اوطريقة لتصنيفهم

① Synchronous ⇒ متناسقة و متلاحقة.

← كل cycle جديدة حسب ال clock بحسب next state جديد

← كل ما تشوف rising edge بشتغل ، وبتعطي next state جديدة ، وبتشتغل ال circuit كاملة

← بقدر نخليها تشتغل عال rising edge او عال falling زي ما بدي

② Asynchronous ⇒ وقت عملها ودورية غير منتظم

← ما بنعرف ال rising edge الي جاي كثيراً بعد قديش راح تيجي لأنه مش منتظم

SUBJECT:

بشكل عام

output ↑ new state = f(input + old state)

↳ we call this type meely

output ↑ new state = f(old state)

↳ we call this type moore

Storing Information

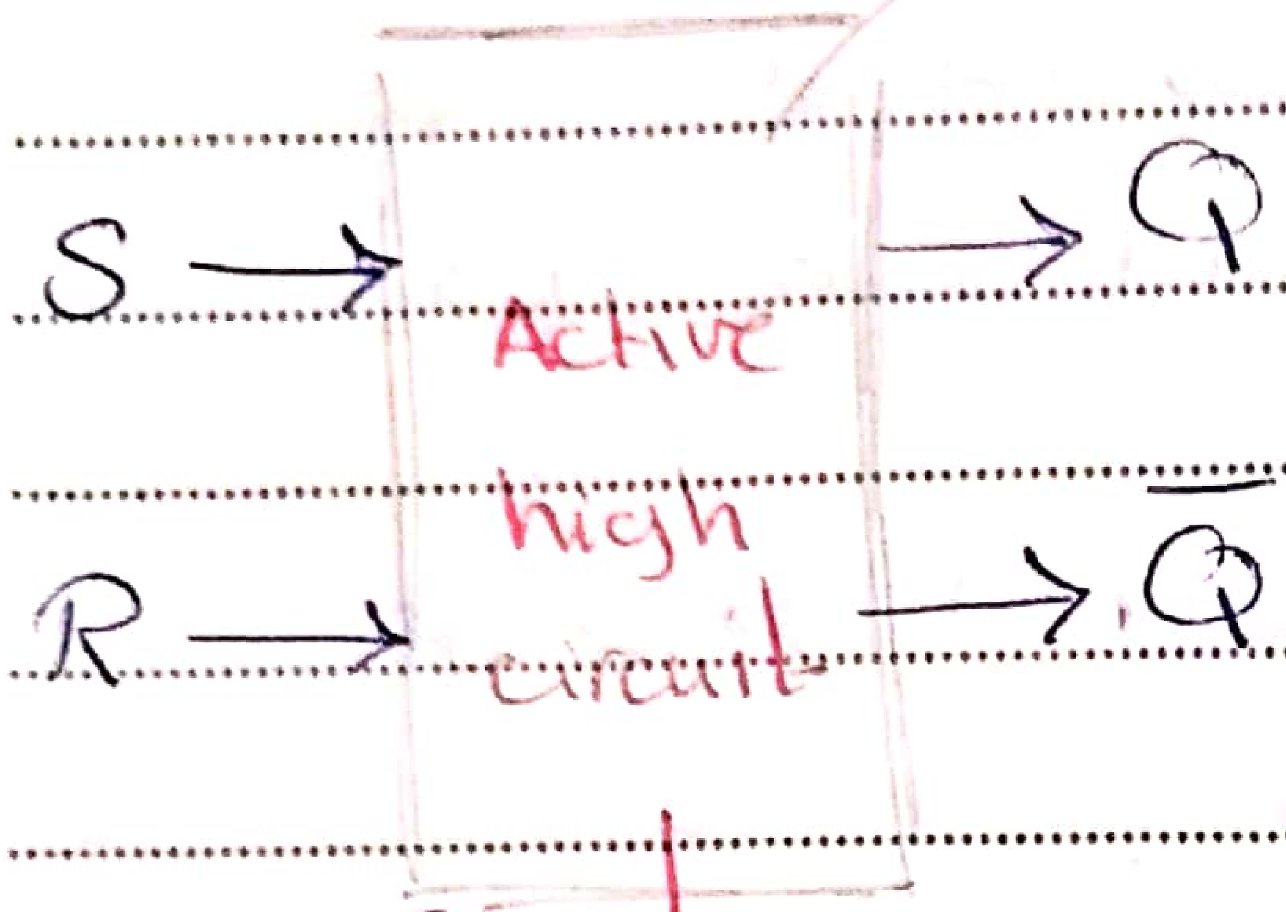
عشان نعمل circuit بتحتفظ بال ← loop

تعمل يعني refresh لنفس ال value المحفوظة كل مرة

SUBJECT: / /

[1] SR latch → first storage element ✓

القيمة اللي راح اخزنها فيها بتضل محتفظة فيها.



2 NOR Gates

first output is the complement of the second one

بدون inverter

بس هو هيك ال design واللي بطلع منه

R (reset) → zero وظيفتها تخزن

S (set) → one وظيفتها تخزن

أهم output هو اللي بعمل ال memory

S	R	
0	0	hold, remember $\underline{Q}$ ⇒ old value بتضل تحتفظ بال اللي بتخزن فيها آخر شي
0	1	→ reset → $Q = 0$ / $\overline{Q} = 1$
1	0	→ set → $Q = 1$ / $\overline{Q} = 0$
1	1	→ not allowed ✓

SUBJECT:

Clocked SR latch using And Gates
- 0 → hold mode
- 1 → regular S-R latch

C works as enable / disable

يعني إنه هي تستجيب للـ user أو هي تدخل بال hold state

حول ما هي (1) بيشتغل الـ circuit عادي ✓

بنستخدم And بين C و S، وبين C و R

S̄R̄ latch

G active low → set is (0)
reset is (1)

truth table بالعكس
using Nands.

S̄ → 0 بخزن
R̄ → 1 بخزن

S̄R̄ clocked latch

بنستخدم Nand بين C و S، وبين C و R

عشان نجبرها ترجع حالة الـ active high circuit الخارجية، لكن ما صحوا بين S̄ و R̄ بحكي active low

يعني من برا ما بين C، بتكون بشغالة زي الـ SR clocked latch ✓ العادية

SUBJECT: ....

$\overline{SR}$ clocked and SR clocked latches ← مهم

→ have the same truth table ✓
→ using Nands in $\overline{SR}$ does the job ✓
→ both work on active high ✓

D latch → ✓ مهم أكثر، وبيمسح الـ errors ✓
✓ بيستبدل الـ S,R بـ D وال inverter
✓ حاسس ع الـ truth table ← 2 hold states
Or only one (No change combination)

G clocked ✓ اوعى لازم تحفظوه

Flip-Flops

بحلوا مشكلة ال transparency ، وبيخزنوا storage

بتغير ال Q بحسب عند ال negative edges

وعكسهم بحيث بيشتغل عال positive

C	S	R	Q
↓	0	0	hold
↓	0	1	Reset
↓	1	0	set
↓	1	1	no change

otherwise → hold

الفرق بين ال flip flops وال latches هو كيف بتعاملوا مع ال clock

flip flops ← 2 latches

أول وحدة active على ← c=1 (The master)

ثاني وحدة ″ ″ ← c=0 (The slave)

SUBJECT: / /

← اول شي الـ Y هي الوحيدة اللي راح تتأثر بالـ user

بعدين بتثبت مش ممكن تتغير قيمتها (يعني بعد ما تاخذ الـ قيمها من الـ user مش ممكن الـ user يغير أيه)

← الـ slave بعدين بشتغل وقيمة Q راح تعتمد على الـ Y

وبشتغل عند الـ negative edge للـ f.f

↓

active high

وبشتغل عند الـ positive edge للـ ff

↓

active low

ما د يقلل احتمالية الـ (errors)

flip-flop Proplem / Error & it's catching

→ The reason is the redundancy in (hold) combination caused by (0,0) input of (S,R) → in (the master)

Direct Inputs

قادرين يغيروا بأي لحظة بصورة آنية ال Q بدون ينتظر من ال clock

S	R	C	D	Q	$\bar{Q}$
0	1	X	X	0	1
1	0	X	X	1	0
0	0	↑	0	0	1
0	0	↑	1	1	0

→ (active high) → with positive edge

ممنوع ال user يكبس الكبستين مع بعض

positive edge

active high → positive edge إجباري

active low → negative edge إجباري

SUBJECT: / /

Sequential Circuit Analysis

next state → A^+ or $A(t) = A(t+1) = D_A$

next cycle

Example on Analysis Slide (44)

Input : x, y — moore sync

Z : Output

state : A — next : A^+

equations :

$Z = A$ — $A^+ = x \oplus y \oplus A$

State	Inputs		Next state	output
A	x	y	A^+	Z
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

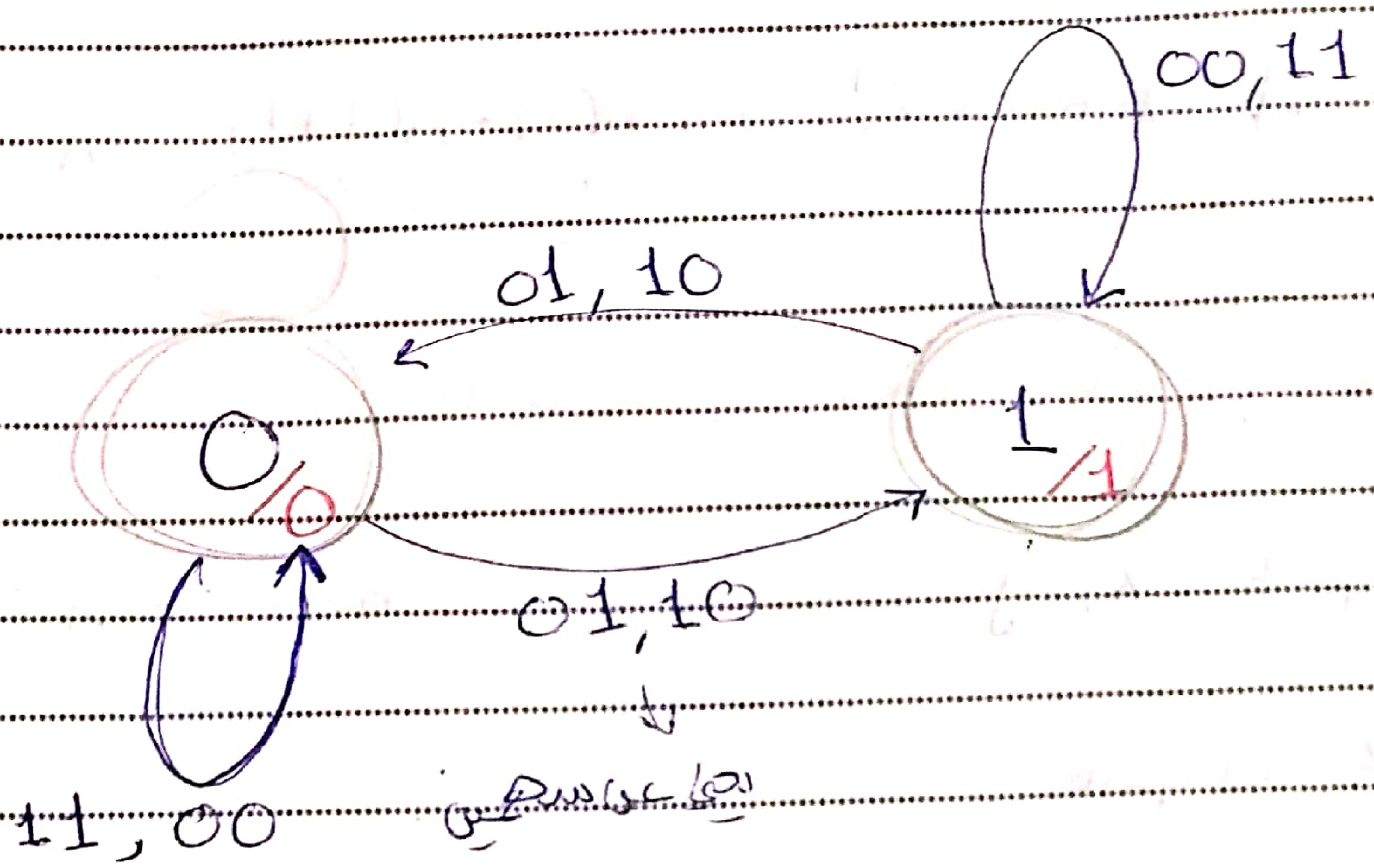
SUBJECT: ....

// flip flops

# of states = 2

#Inputs

state diagram # of transition = 2



← لما يكون moore ← بحطوا ال ~~Z~~ ع الدائرة لأنها ما بتتغير (هو نفسه ال A)

← لما يكون mealy ← بحطوا ال ~~Z~~ ع السهم لأنه عكس بتغير (كل تغير ع A)

SUBJECT: / /

Exercise slide (45)

(mealy)

input: X
outputs: y
states: A, B

طلعوا
الدوائر

التبسيط ساعد

equations:

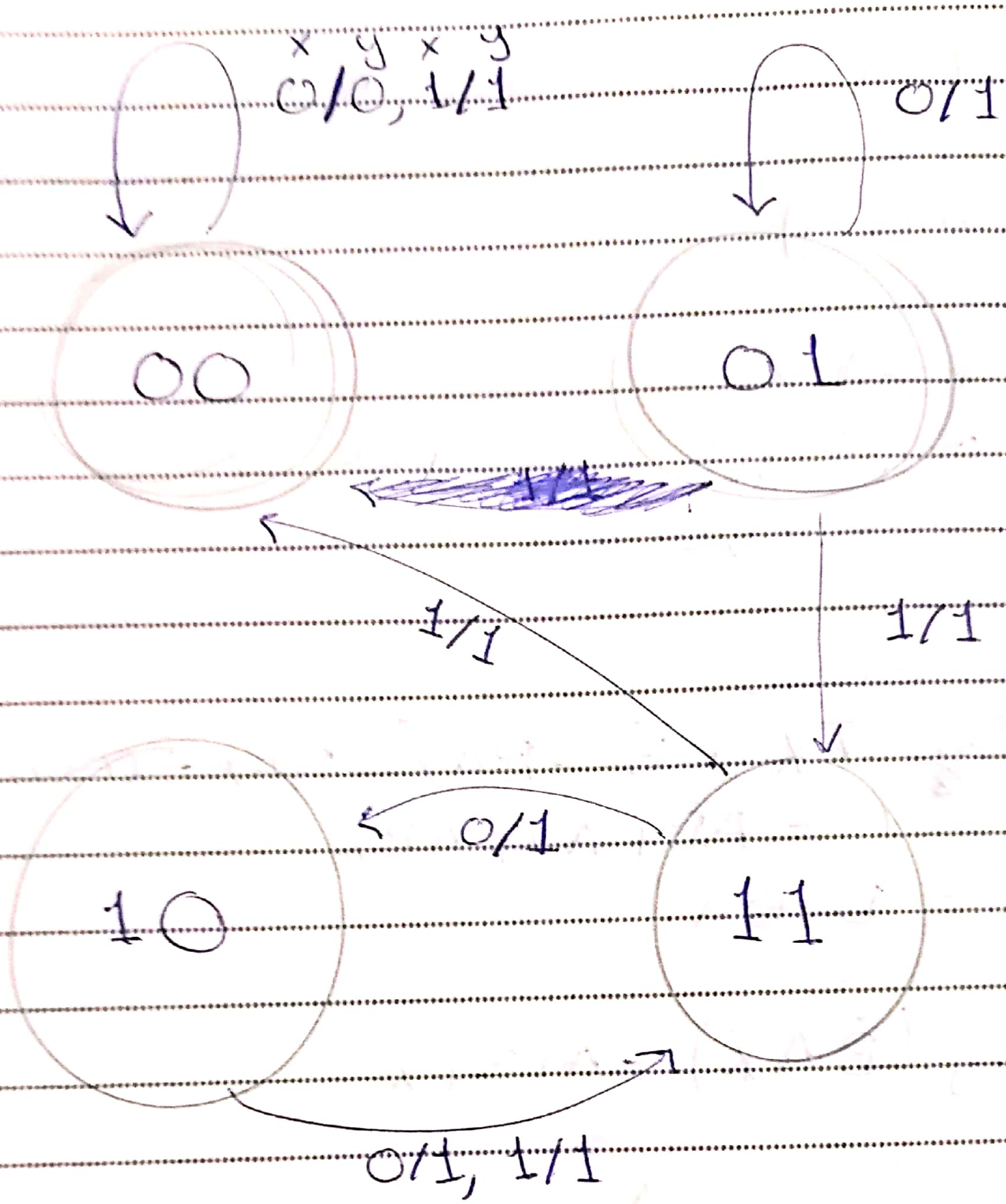
$A^+ = \bar{A}X \oplus \bar{B} = \bar{B}\bar{A}X + B\bar{A}X = \bar{B}(A+\bar{X}) + B\bar{A}X$
$= A\bar{B} + A\bar{X} + \bar{A}BX$

$B^+ = B \oplus A$

$y = \overline{(\bar{B}\bar{A}\bar{X})} = A + B + X$

states A	B	Inputs X	next states A^+	B^+	output y
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	0	0	1

SUBJECT:



Alternate State Table

الفكرة إنها هاي بالعرض عن ال state table العادي بس ^_^

Exercise slide (51)

when no input is available :
transition without condition

Reset ← Direct input حاطط
(active low)

وبتضل لآخر ال clock ، بيضل يشتغل ويلف
ورا بعض ، لحد ما يرجع ال user يكبس ال reset

111 → 110 → 101

Unused states
ما راح نوصلهم

Reset → high voltage → مش مكبوسة
Reset → low voltage → مكبوسة