تفريغ فيديوهات

لاب المنطق الرقمي

للدكتور وليد دويك

2022 - 2021

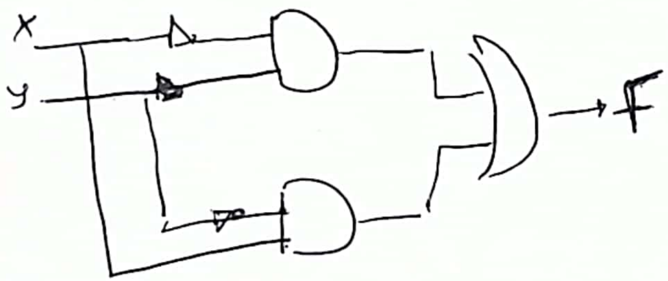
sondos abu iolaq *

Lab logic :

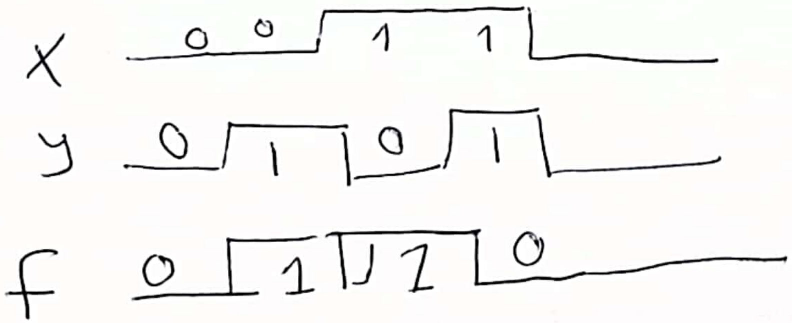
Exp 1 : cyclone II → الرابع من تحت*

new → design file → Blockdiagram

* $X \oplus y = \bar{X} \cdot y + X \cdot \bar{y}$

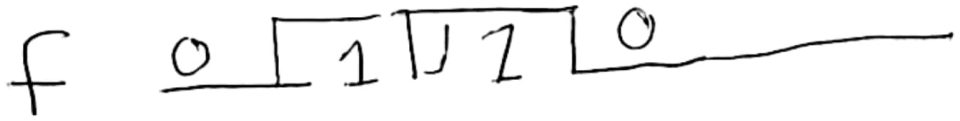


X	y	F
0	0	0
0	1	1
1	0	1
1	1	0



Wave Form :

newFile → new → other → vector wave formfile → save

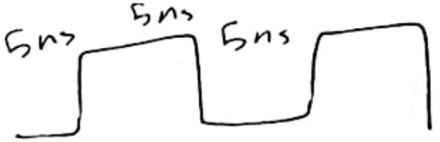


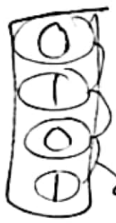
Wave Form :

Newfile → new → other → vector wave Formfile → save

* تحديد الزمن : كل كم بدي أختبر الـ (x) والـ (y)

طرق :



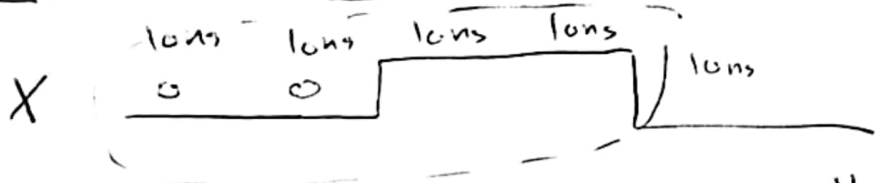


الزمن كشان أعطي كل الاحتمالات بدي

$4 * \underbrace{5}_{\text{time}} = 20\ ns$

① endtime ← Edit

② Clock ← Value ← double click



طول البيريود لـ X = 40ns

$4 * 10$

40 = endtime ← Clock ← Value * (X لـ)

Simulation → assignment → settings → functional → تأكد ان اسم الـ wvf موجود عند الـ input

- processing → generate functional.
- simulation tap → blue → ok
- fit in window

note: Timing → generate functional (process) → لا يحتاج
① لأنه يعمل كل الوقت ودايرکت
② يطوّل لتحويل الاوتبوت لـ 0 و 1 لانه يعمل اعتبار للـ delay (delay)

---

pins:

Implementation

- Full custom
- ASIC
- FPGA* ← يستخدم Lookup-tables

Pins :

Implementation

- Full custom
- ASIC
- FPGA *
  - ↓ تستخدم
  - Lookup-tables

- assignment → ~~set~~ assig editor → pin → switches تعباية

- بعد عملية ass pins ← بعمل فقط Compile

- Tools → programmer → start → kit شغل الـ
  - ↓
  - وضع: Hdrw: USB — mode: JTAG

Exp 2 : Verilog HDL is case sensitive

Example : module Nandgate (x_1, x_2, y);

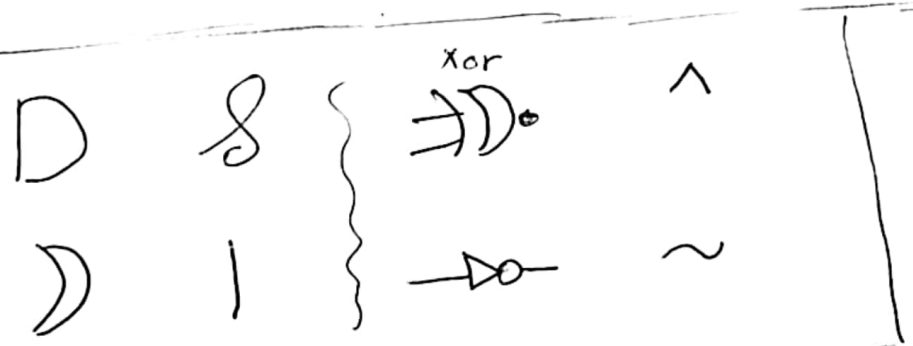
input x_1, x_2;

output y;

→ behaviorally → assign a = b & c | d ; [small modules]

→ structurally → logically : AND, or → and match them

[Top level Modules file]



Nandgate (3)(. . .)

↓

which means that there are 3 inputs

Exp 4 :- Decoder / Encoder

decoder → combinational circuits → / $n-2^n$ Line decoder

input		output			
	A B	D_0	D_1	D_2	D_3
0 ←	0 0	1	0	0	0
1 ←	0 1	0	1	0	0
2 ←	1 0	0	0	1	0
3 ←	1 1	0	0	0	1

→ active high ⇒ 1

$D_0 = \bar{A} \cdot \bar{B}$

$D_1 = \bar{A} \cdot B$

$D_2 = A \cdot \bar{B}$

$D_3 = A \cdot B$

behavior :

```
module decoder2_4 (A, B, D3, D2, D1, D0);
  input A, B;
  output D3, D2, D1, D0;
  assign D3 = A & B;
    //    D2 =   //
    //    D1 =   //
    //    D0 =   //
```

(inputs: A, B; outputs: D3, D2, D1, D0)

Priority - active high / active low

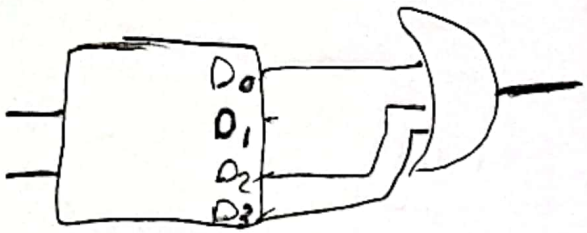
D0 = "

priority = active high / active low

active low → الجواب يصير صفر بدل الـ (1) [في جدول الـ Truth] → كيف بتعبر active low → behavior → not بنعط قبل كل معادلة كانت شغالة على high → structurally [بنعمل بدل الـ AND الـ NAND]

Decoder = minterms generator

$n = 2 \rightarrow 2^n = 4$ minterms / $f(A,B) = \Sigma(0,2,3) \Rightarrow m_0 + m_2 + m_3$



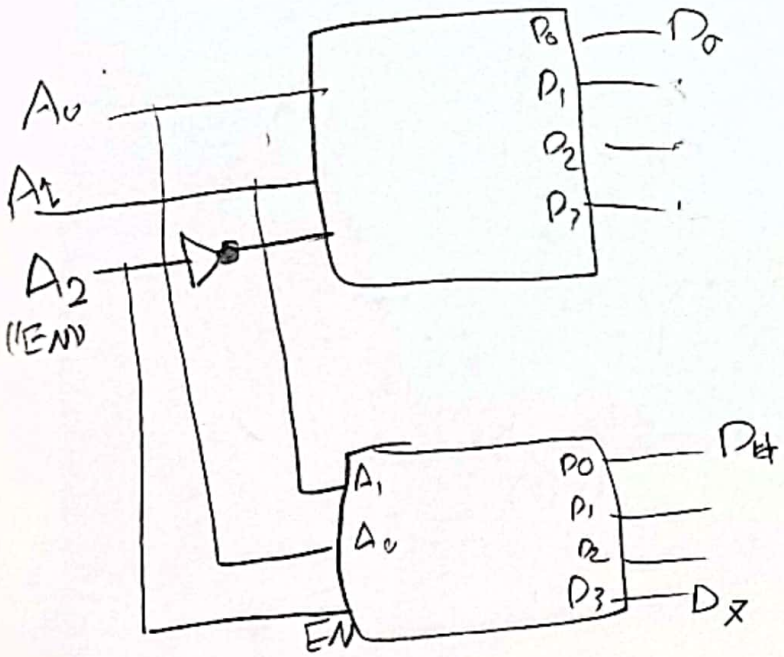
Enable 0 ⇒ All output of the decoder are 0's

1 ⇒ The Selected output of the decoder is 1 , all other outs are 0

« بشتغل الديكودر طبعين »

---

how to design 3-8 decoder by using 2-4 decoder



$D_0 = EN \cdot \bar{A} \cdot \bar{B} \Rightarrow \bar{A}_2\ \bar{A}_1\ \bar{A}_0$

0 0 0

$D_{(4)} = EN \cdot \bar{A} \cdot \bar{B} = A_2\ \bar{A}_1\ \bar{A}_0$

1 0 0 → dec = (4)

Encoder : 2^n to n line encoder

output

y_2	y_1	y_0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

$y_2 = I_4 + I_5 + I_6 + I_7$ = ones في ←

$y_1 = I_2 + I_3 + I_6 + I_7$

$y_0 = I_1 + I_3 + I_5 + I_7$

→ 6 → 110 → $y_2 y_1 y_0$ → y_2 ✓ y_1 ✓ y_0 0

decimal

توضيح

1 → 001 → y_0 ✓

decimal $y_2 y_1 y_0$

برضو

high priority / low priority

7 → أعلى أولوية

decimal

0 → أعلى أولوية

decimal

Exp 4] continue ...

I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	y_0	y_1	y_2
1	-	-	-	-	-	-	0			x
x	x	x	x	x	x	1				1
x	x	x	x	x	1	0				1
x	x	x	x	1	0	0				1
x	x	x	1	0	0	0				1
										0
										0
										0
										0

$y_2 \Rightarrow \underbrace{I_7 + I_6 + I_5 + I_4}$ وراء بعض موجودين

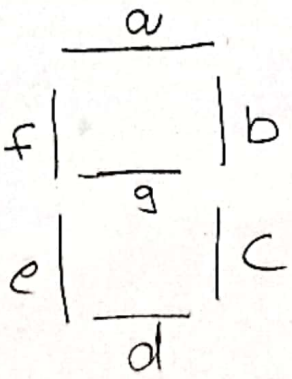
$y_0 = \bar{I}_7 + I_5 \bar{I}_6 + I_3 I_4 \bar{I}_6 + \bar{I}_1 \bar{I}_2 I_4 \bar{I}_6$

y_0
x
1
0
1
0
1
0
1

$y_1 = \bar{I}_7 + I_6 + \bar{I}_3 \bar{I}_4 \bar{I}_5 + I_2$ ...

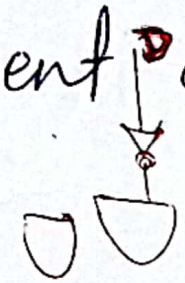
$\bar{I}_6$ فاطلعت هون لانه (I_6) موجودة بالمعادلة تم استخدامها مسبقاً

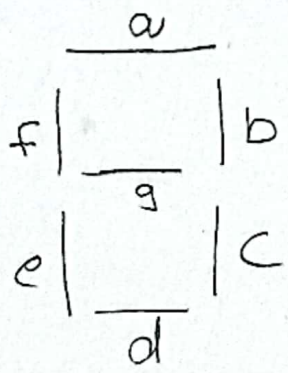
Segment display :



LEDs

Segment display :

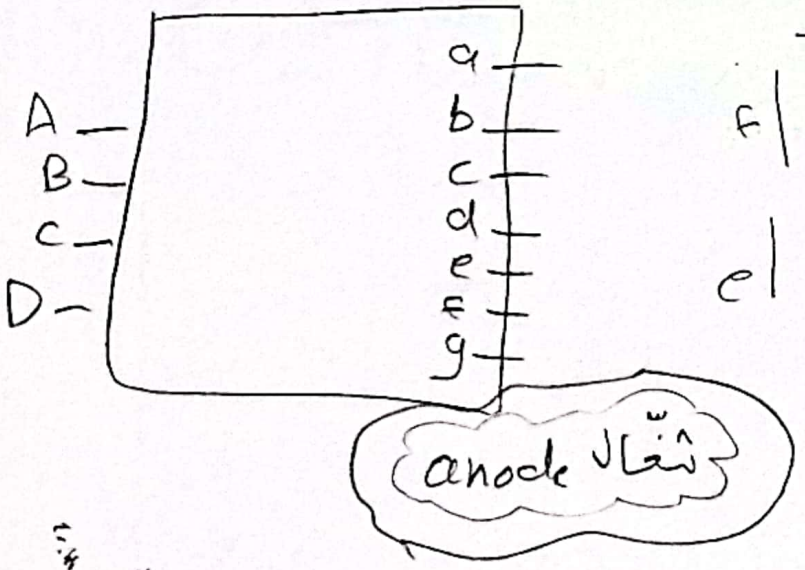


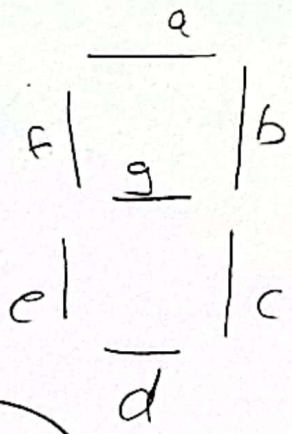


LEDs

* Common Cathode : Zero هشبوك عليه "low v"

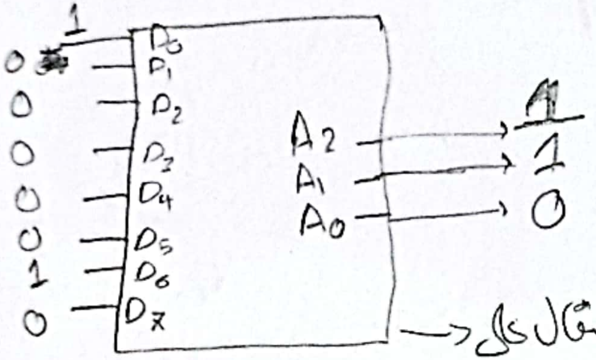
* Common anode : 1 هشبوك عليه (high voltage) → هوالي بشتغل 0
هوالي بطفي 1





ABcd	a b c d e f g
0 0 0 0	0 0 0 0 0 0 1
0 0 0 1	1 0 0 1 1 1 1
⋮	⋮

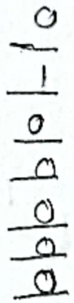
شتغالين

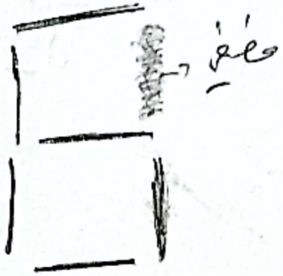


→ شغال لل
[high prioarity]

Encoder
8-3

لذلك 2
يصبح D6
كلاله وتبعون
وليه D0

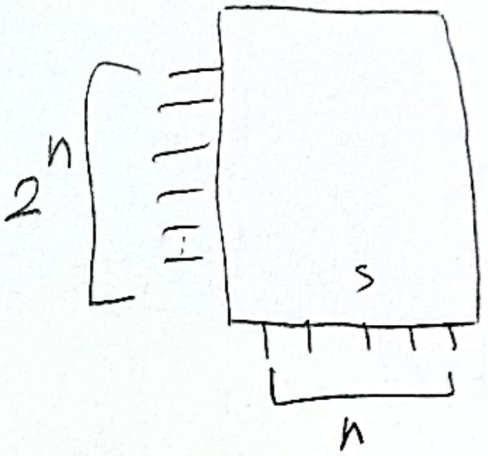




* طريقة عمل
7-seg
في التجربة
Lab sheet

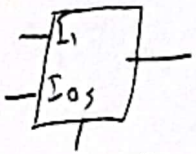
* Top level entity → لازم يكون اسم
الفايل والمودل
نفس الاشي

Exp 5 :- Mux



2^n to 1 Mux

n = 1



2 to 1 Mux

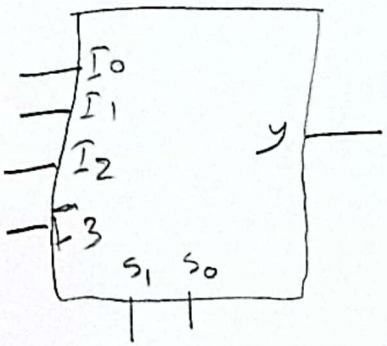
① Truth table

S	I_1	I_0	y
0	0	0	0
1	0	0	0
1	1	1	1

② equation

assign y = - - - - - -

4-to-1 Mux

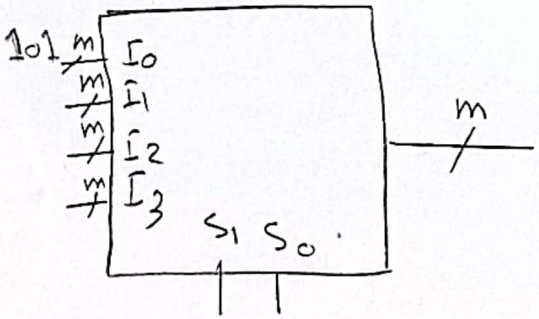


S_1	S_0	y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

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

AND, OR, INV
3 input 4 input

Width Expansion:



* الـ Select ما الها دخل
Line

بتحدد الـ bits للـ input و bits للخرجات كل
الاوتبوت

* m-bit 4-to-1 Mux
* 2-bit [dual] 4-to-1 Mux
* 4-bit [quad] 4-to-1 Mux

S_1	S_0	y_1	y_0
0	0	I_{01}	I_{00}
0	1	I_{11}	I_{10}
1	0	I_{21}	I_{20}
1	1	I_{31}	I_{30}

$$y_1 = \bar{S}_1 \bar{S}_0 I_{01} + \bar{S}_1 S_0 I_{11} + S_1 \bar{S}_0 I_{21} + S_1 S_0 I_{31}$$

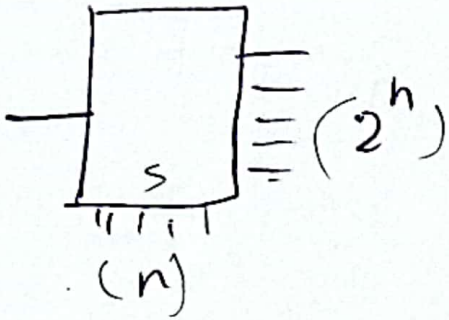
$$y_0 = \bar{S}_1 \bar{S}_0 I_{00} + \ldots + \ldots$$

assign ...

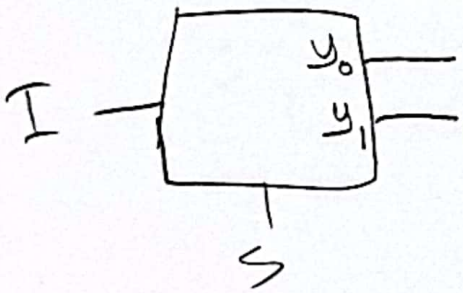
assign ...

module ... (inputs .... , outputs)

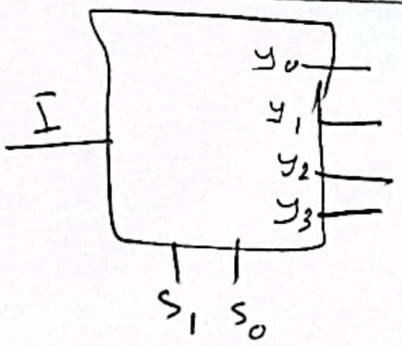
dMux



$n=1 \Rightarrow$ 2 to 1 DMux



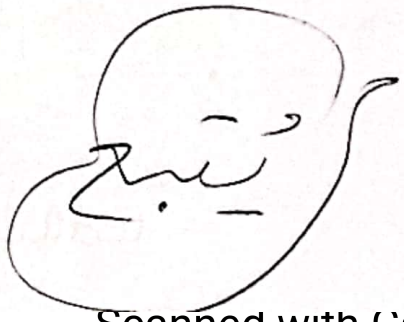
S	I	y_1	y_0
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	1



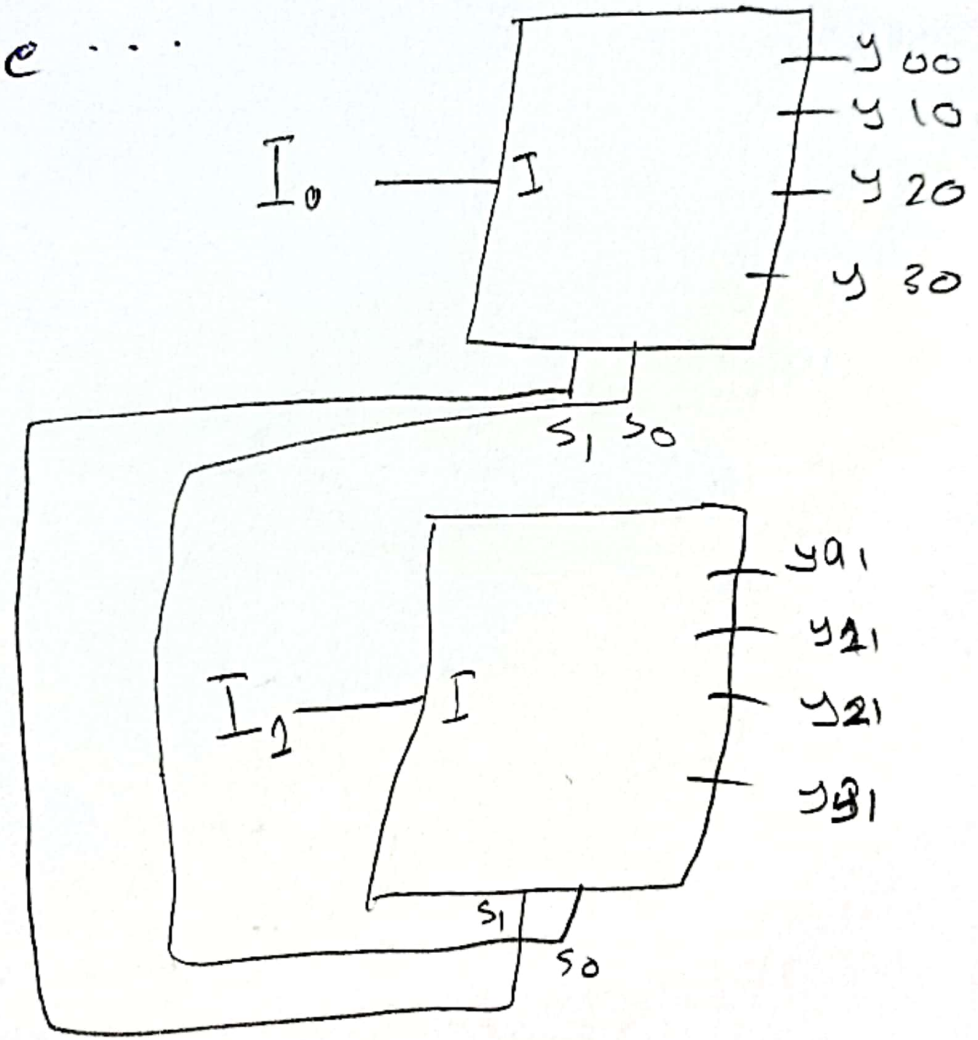
S_1	S_0	y_3	y_2	y_1	y_0
0	0	0	0	0	I
0	1	0	0	I	0
1	0	0	I	0	0
1	1	I	0	0	0

$y_0 = I \bar{S_1} \bar{S_0}$

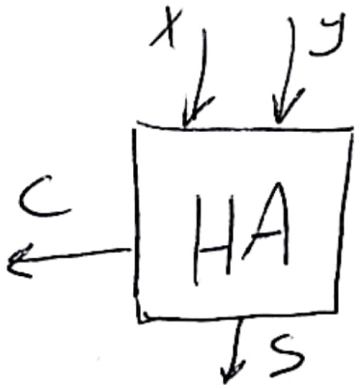
$y_2 = S_1 \bar{S_0} I$

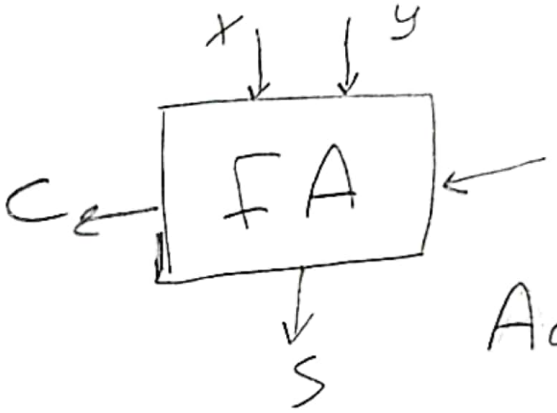


Exp 5 : Continue ...



Lab Exp 6]:





Adder

X	y	Z	carry C	sum 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

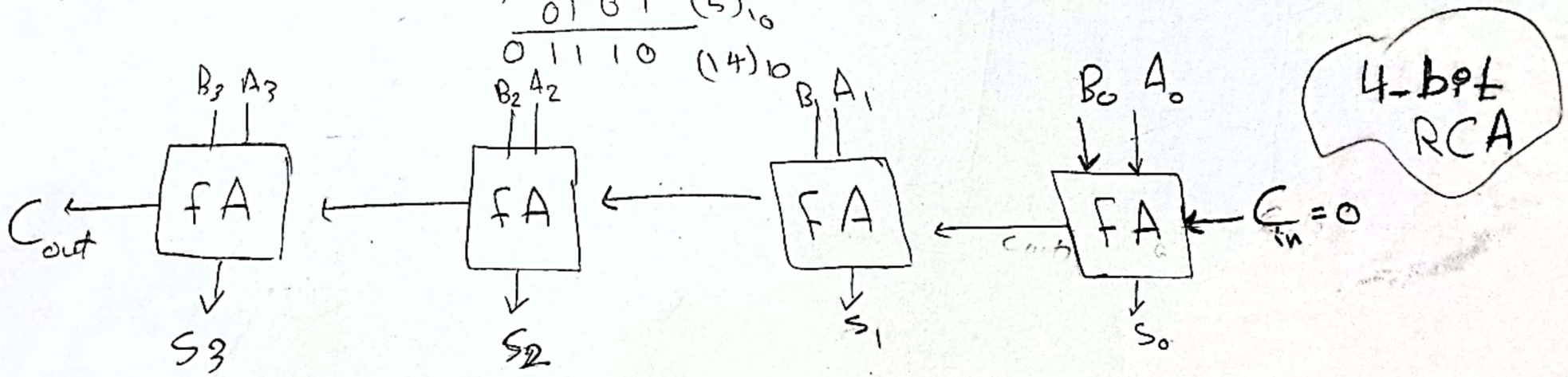
$$\Rightarrow S = \bar{X}\bar{y}Z + \bar{X}y\bar{Z} + X\bar{y}\bar{Z} + Xy Z$$

$$S = Z \oplus X \oplus y$$

$$C = (Xy) + (yZ) + (XZ)$$

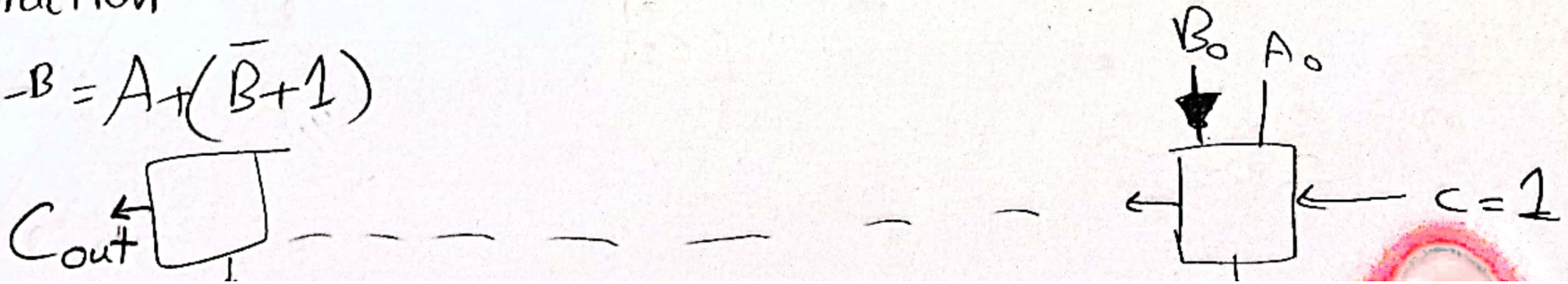
Ripple Adder :

$$
\begin{array}{rr}
 & 1\,0\,0\,1 \quad (9)_{10} \\
+ & 0\,1\,0\,1 \quad (5)_{10} \\
\hline
 & 0\,1\,1\,1\,0 \quad (14)_{10}
\end{array}
$$



* subtraction

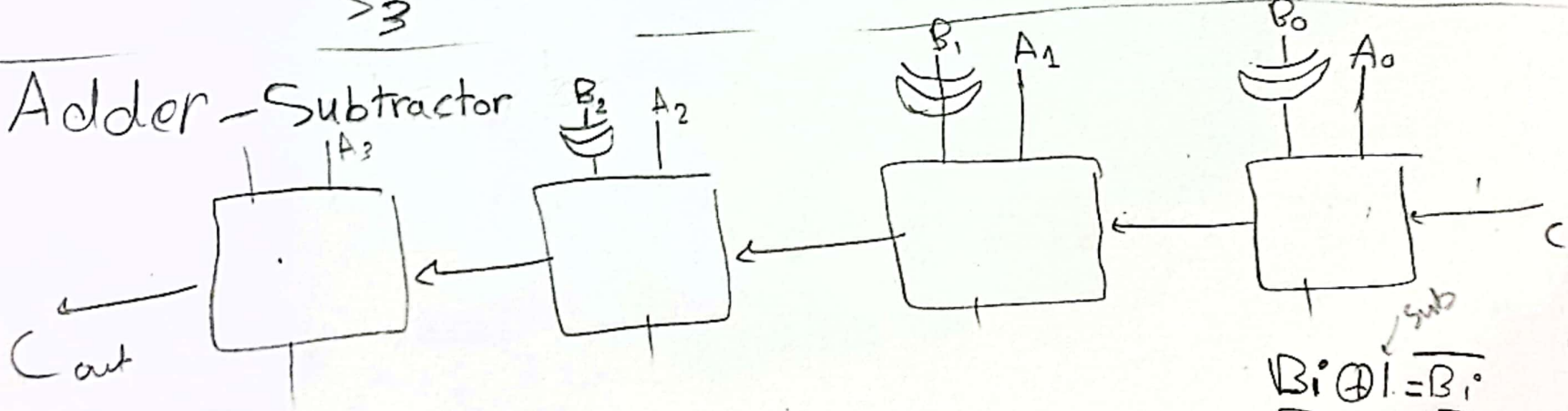
$$A - B = A + (\bar{B} + 1)$$



* subtraction

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

Adder - Subtractor



4-bit (adder-subtractor)

sub

$B_i \oplus 1 = \overline{B_i}$

$0 \oplus 1 = 1$

$1 \oplus 1 = 0$

Sub = 1

add

$B_i \oplus 0 = B_i$

$1 \oplus 0 = 1$

$0 \oplus 0 = 0$

Adder = 0

module Three_Bit_Adder_sub(A1, A2, A3, B1, B2, B3, S1, S2, S3, C4)

//FA(X, Y, Z, C, S)

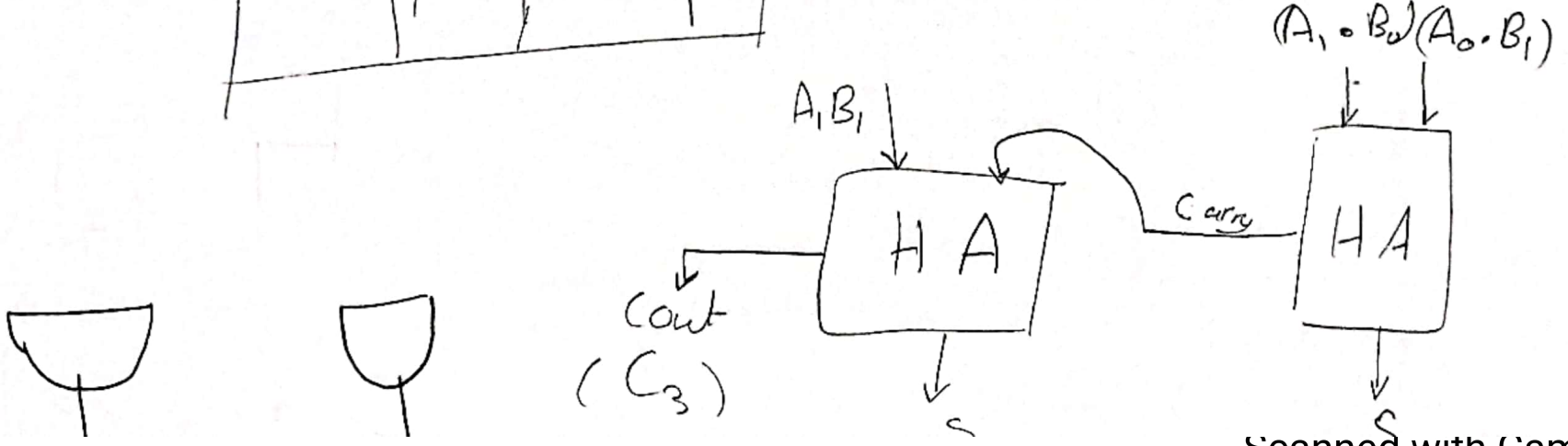
FA FA1(WB1, A, M, C1, S1); ← Xorgate xor1(B1, M, WB1)

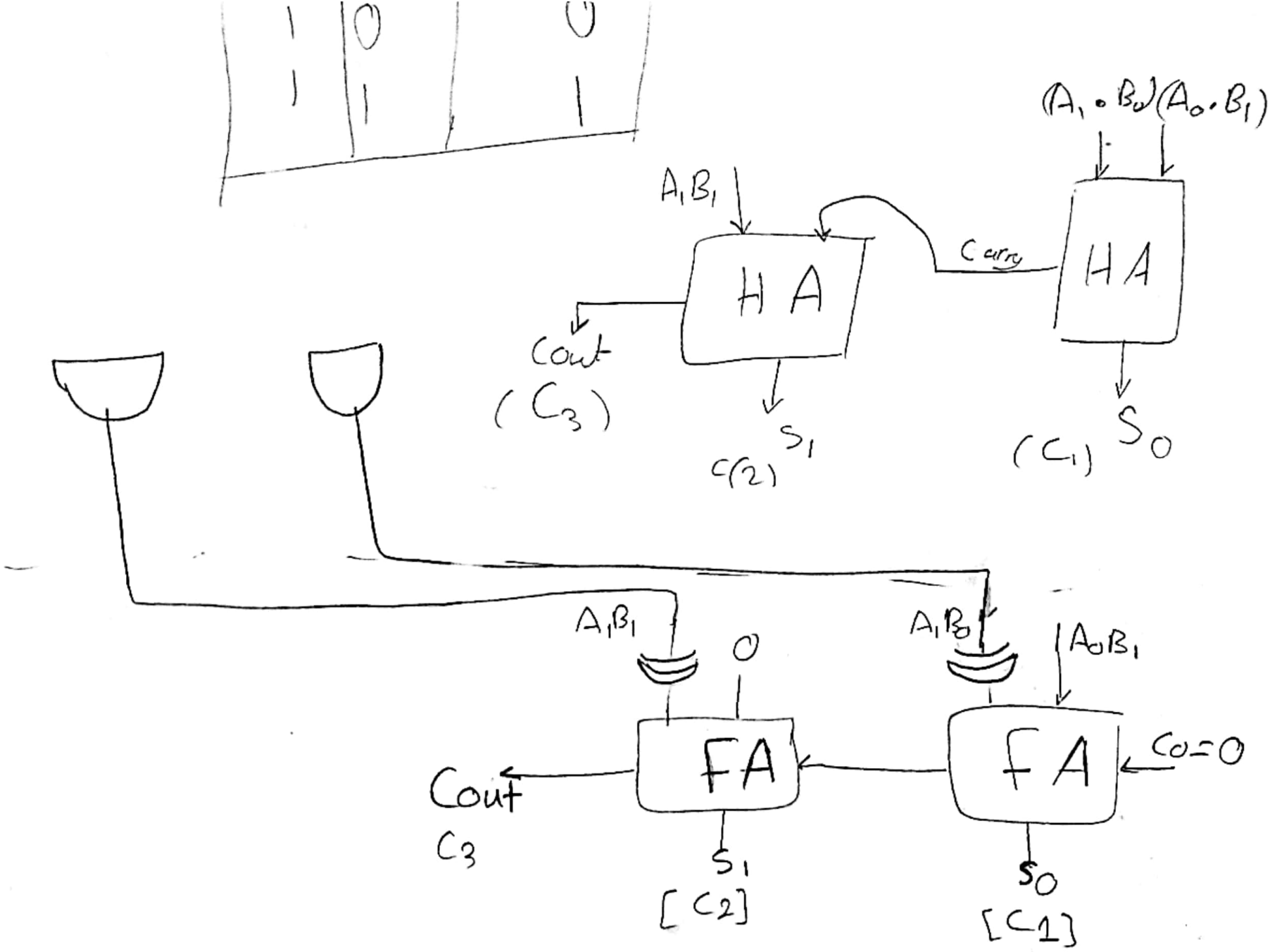
* مهم الترتيب والتشبيك العكسي

Multiplier :=

A	B	A X B
0	0	0
0	1	0
1	0	0
1	1	1

→ تشبه عملية AND



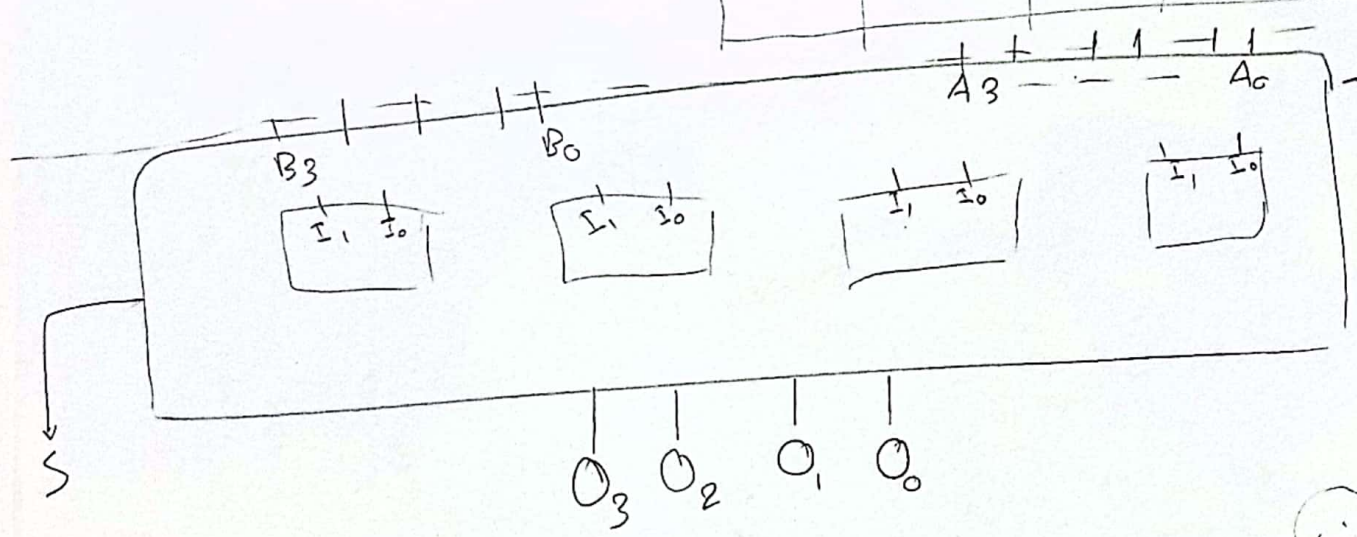


Lab 6: comparator

البرمجة في الدفتر

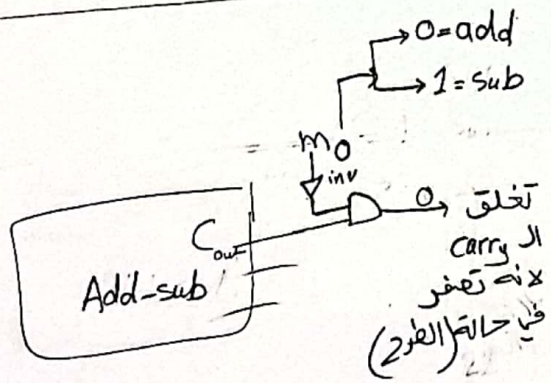
B	A	A>B	A=B	A<B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

التجربة :



* الديزاين structural

4 output
6 input

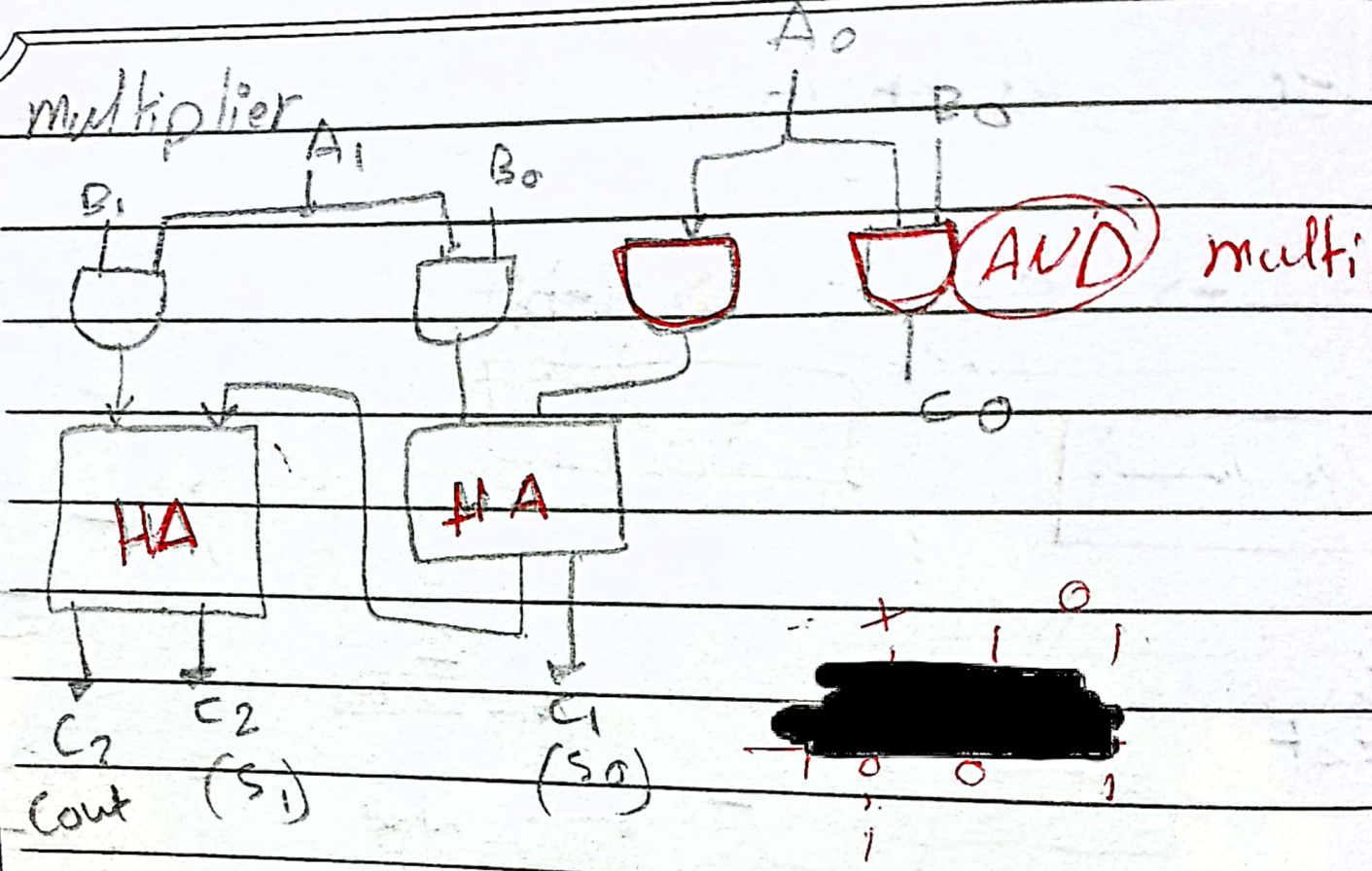


← في عملية الطرح قابلتم بالـ carry

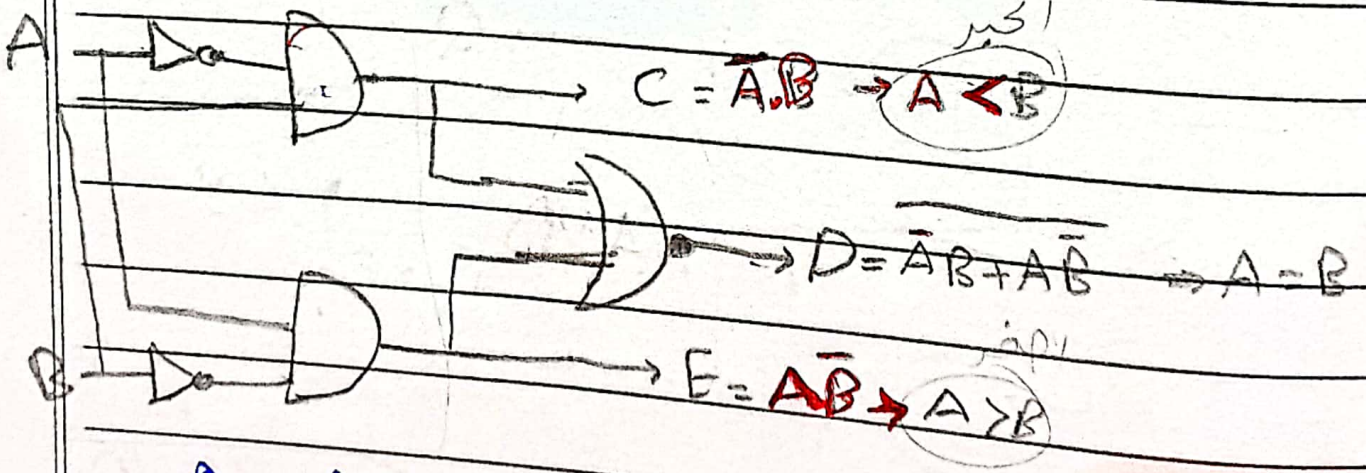
← " " الجمع بهتم

Multiple

multiplier



Comparator



A0 A1 A2 A3

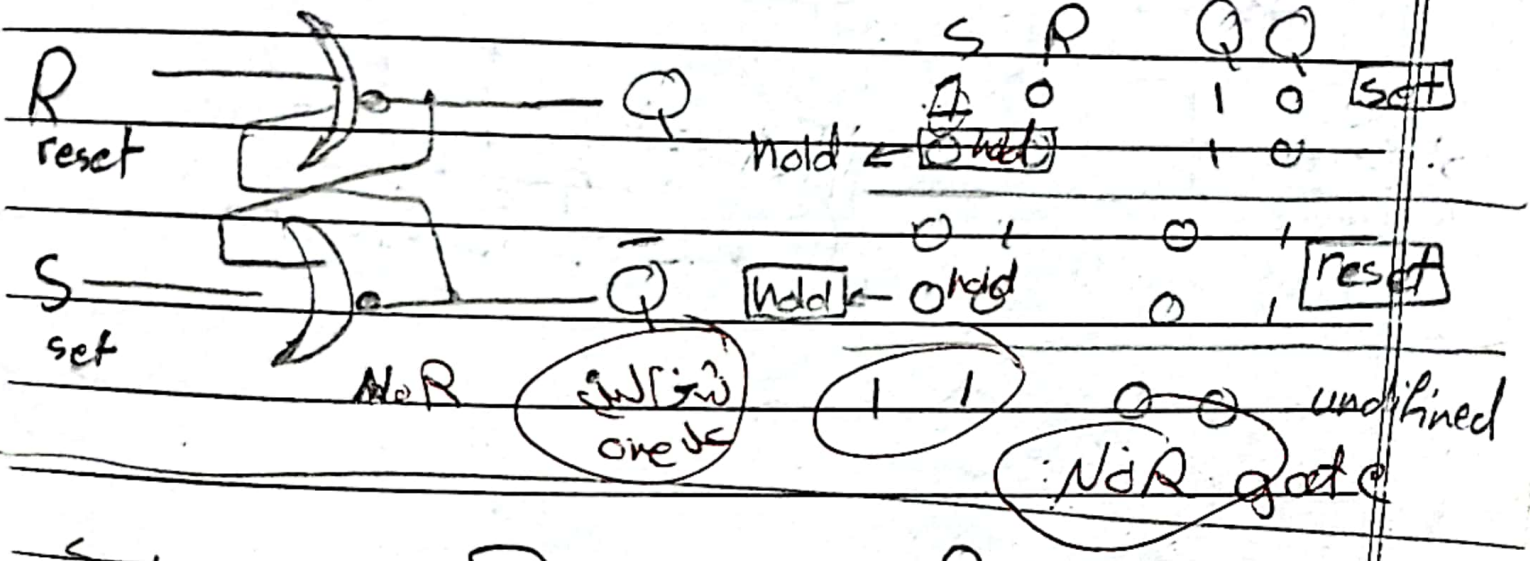
"Exp 3"

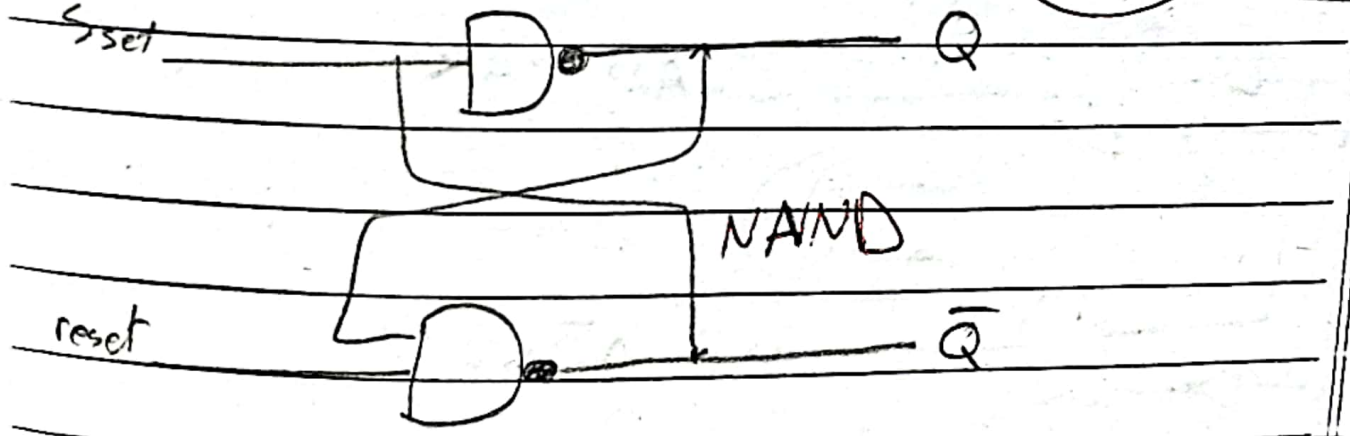
latches & flip flop registers

The S-R Latch ⇒ 1-bit Syncro

NAND gate

NOR based





S	R	Q	$\bar{Q}$	
0	1	1	0	
1	1	1	0	set
1	0	0	1	reset
1	1	0	1	

"exp 8"

3 stage shift register

A register thats used to store information

counter ⇒ synchronous Binary Counter

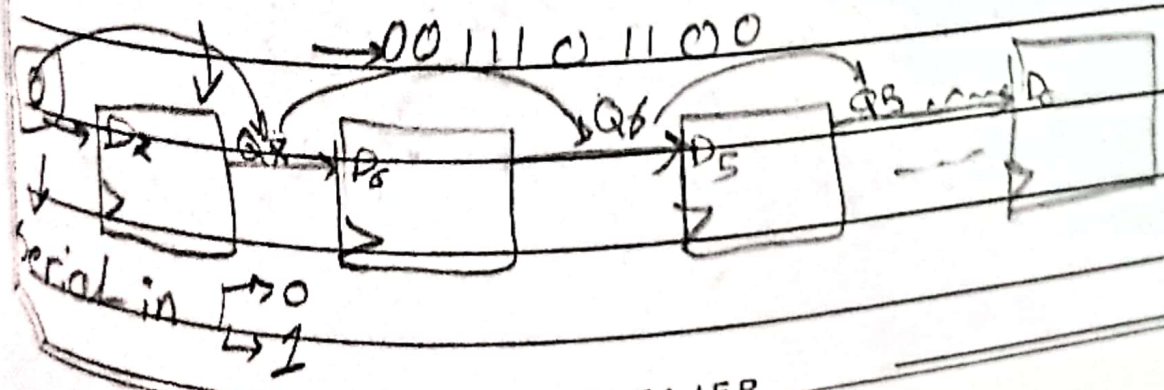
Flipflop هو Synch storage element ← هو

at reset = 1 all flips = zero

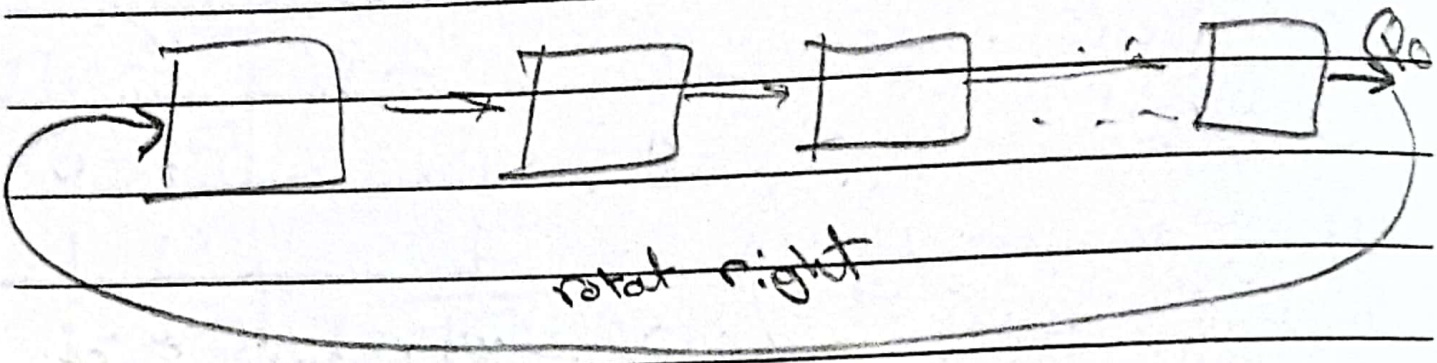
hold, load, rotate (left, right), shift (left/right) ⇒ بعملها الـ register

shift left ⇒ بتحرك ناحية اليسار ← 00110 1010 ← قبل الضرب في 2 unsi

shift right ⇒ بتحرك ناحية اليمين → 00111 0110 قبل القسمة على 2 unsi



rotate ⇒ بلف فقط ⇒ ما بدّل اشي

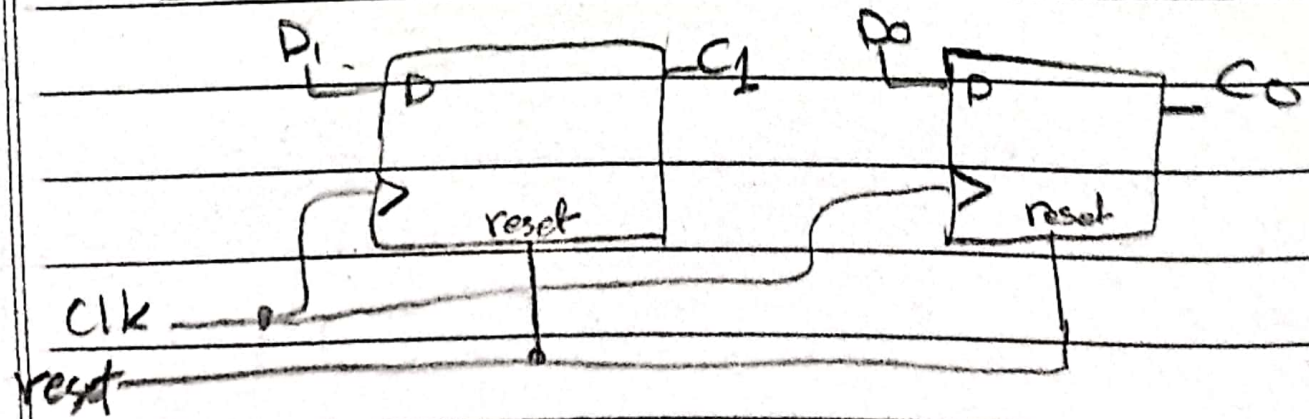


Universal Shift register

Counters ⇒ sequential Circuits
(عدادي عدّاد)

2-bit up counter

00, 01, 10, 11, 00 ....

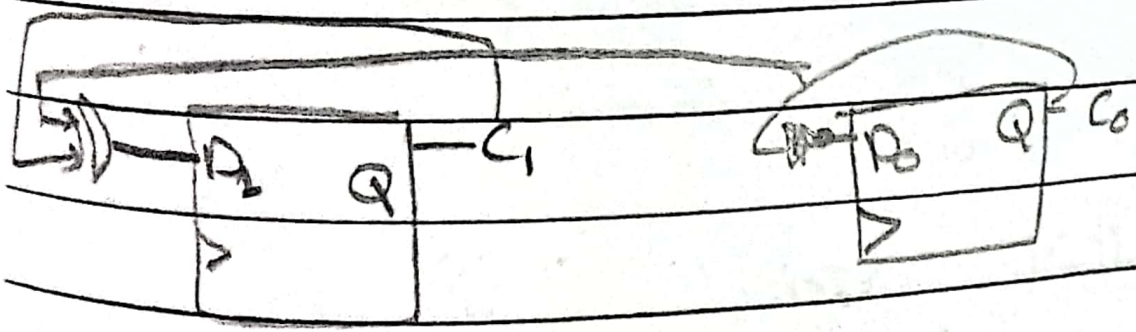


present state		Next state		Flip-flop inputs	
C_1	C_0	C_1	C_0	D_1	D_0
0	0	0	1	0	1
0	1	1	0	1	0
1	0	1	1	1	1
1	1	0	0	0	0

Result

$D_1 = C_1 \oplus C_0$

$D_0 = \bar{C}_1 \bar{C}_0 + C_1 \bar{C}_0 \Rightarrow [\bar{C}_0]$



التاريخ

exp # 3

design Entry

- Schematic
- HDL

Implementation

- FPGA
- Custom Circuit

Small integrated circuit ⇒ 10 gates

Medium " " ⇒ 10 - 100

large " " ⇒ 100 - 10000

Very large " " ⇒ 10000 gates

like: Complex microprocessor

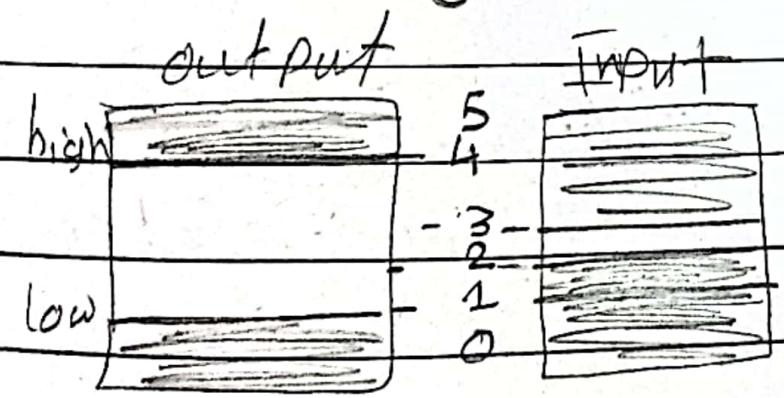
logic Families:

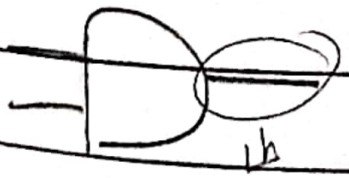
TTL	NMos
CMos	PMos
ECL	

TTL num ⇒ 74

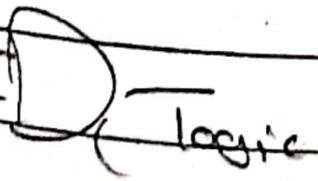
standard 74xx
high speed 74hxx
low power 74Lxx
shottky TTL 74Sxx
low powe shottky 74LSxx
Advanced shott 74ASxx
" low " 74ALSxx

* All TTL designed to operate from 5V power supply





logic zero ⇒ (0-1) volt



(logic one) ⇒ (4-5) volt

* اشارات لنقل السجنال داخل السلك الخارج من الاند

بيضعف السجنال او بتتغير لذلك الدال Inp على

(+1) لنتيج لحتى تتحرك الداتا براحتها وما توصل لمرحلة

ال Noise

* noise : منطقة مشوشة ما بتعرفها السيركت اشو قيمتها

* power needed by the gate delivered from supply = 20 mW per gate

"it will help portable equipment to know what type of battery might be needed"

* propogation delay speed (10.6 to 50) nano sec

* fan in ⇒ maxi 8 gates input

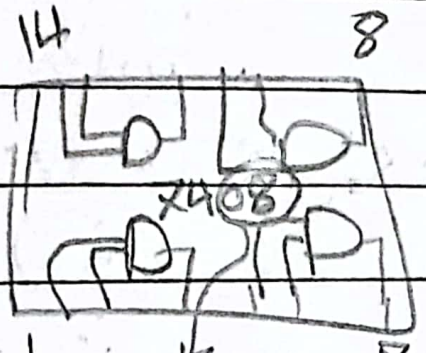
fanout ⇒ 12 gate

بعد لاوصل لبرة الاند

كل 12 gate مختلفة

+5V بيشتغل على

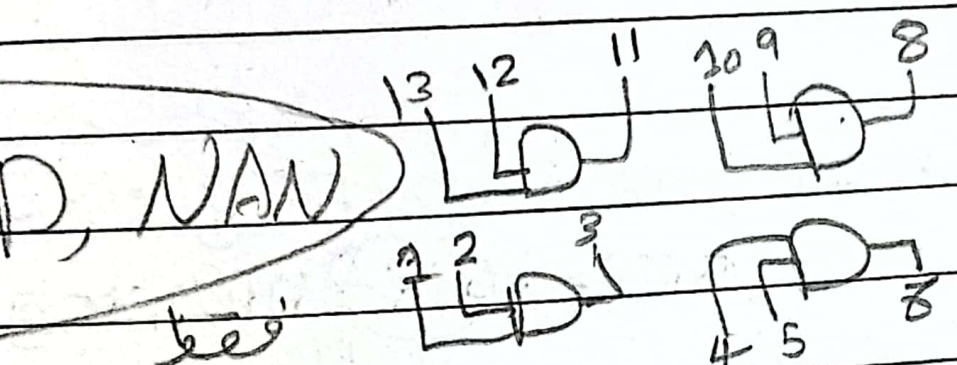
طريقة تقسيم الـpins سمي dual package inline



Total = 14 pin

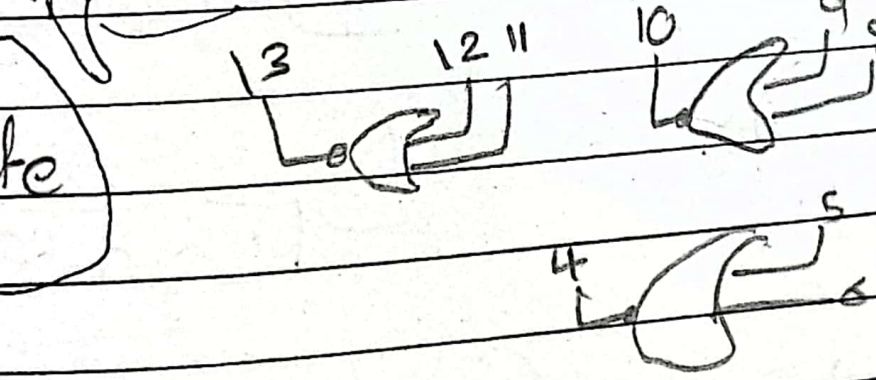
مشبوكة على ground = logic "zero"

OR, AND, NAN



فقط

NOR gate



04 → INV

08 → AND

00 ⇒ NAND

02 ⇒ NOR

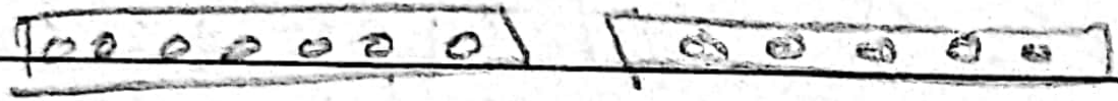
32 ⇒ OR

SN → Texas Instruments

J → Ceramic dual

short circuits 

Zero volt logic



5 volt