Chapter "6"
"Registers and Counters"

6.1

* CPU: Central processing:

Contains:

1. ALU: Arethmetic logic unit (+ −)
2. CU: Control Unit
3. Register: memory, group of Flip Flops.

1 Flip Flop → memory 1 bit → n bits → n-flipflops
2 Flip Flops → memory 2 bits.

* Counter: is a register goes through a Predetermined sequence of state

هو عدّاد يمشي بمتسلسلة محددة. يعني ممكن يبدأ بـ 000 ثم
عندما تشتغل ال clock بنتقل للنقطة الأخرى حسب الشروط.

* Clear = 1 → normal operation, $A_x = I_x$
Clear = 0 → دائماً $A_x = 0$. دفعة تعزيز مسح.

* parallel → يدخل على ال Circuit بيدخل أكثر من bit سوا
* serial → (يدخل فقط bit واحد ثم يليه الثاني وهكذا) (مشي مع بعض يدخلوا)

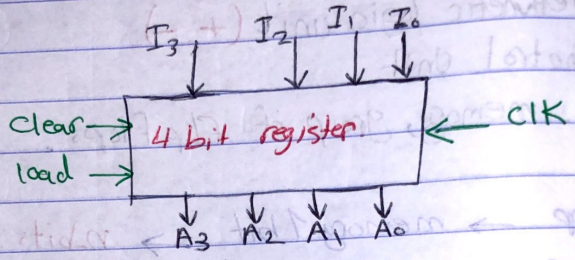
* أي اشي بلف يعني بطلع مدخلات لمخارجات هو P.S، وأي اشي بين كذا وكذا هو N.S
اشي واحد اثنين.

* Load = 0 , No change , N.S = P.S , A(t) = A(t+1)

يعني أيضاً إنو ال circuit تبقى محافظة على قيمتها الابتدائية بغض النظر عن قيم I

* Load = 1

A = I



$\boxed{1}$

if [clear = 0 , load X , CLK X] → $A_0 = 0$, $A_1 = 0$, $A_2 = 0$, $A_3 = 0$

التفسير: إذا كانت قيمة ال clear = 0، لا يهمنا قيمة ال load أو ال CLK ورح تصير قيم A[0:3] تكون تساوي 0.

$\boxed{2}$ clear = 1 , load = 0

∴ No change , A(t) = A(t+1)

التفسير: ال flip flop بضل بهاي الحالة مخزنة نفس القيمة الابتدائية اللي كانت عليها، يعني لو كان مخزن 0000 بضل 0000 لما ال load يساوي 0.

Scanned with CamScanner

$\boxed{3}$ clear = 1 , load = 1 → A = I

التفسير: القيم التي تدخل عم الـFlipFlop [قيم I] ستكون لفترة هي قيم A، يعني A تتبادل نفس قيمة I.

إذا $I_0 = 00 \rightarrow A_0 = 00$

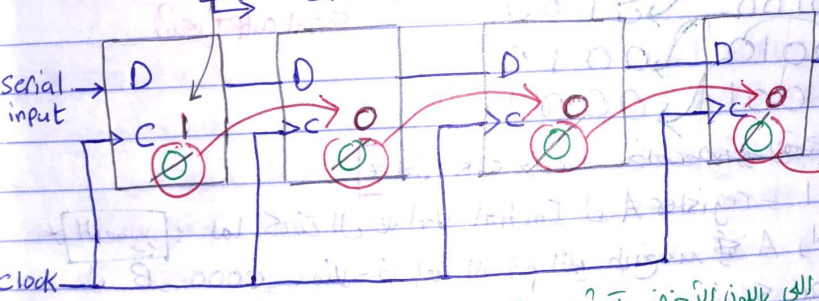
$I_1 = 10 \rightarrow A_1 = 10$ وهكذا...

$\boxed{6.2}$ Shift Registers: (إزاحة bits)

* لو عندي هاي الـCircuit:

* shift to the right:

SI = 101101 → أول bit

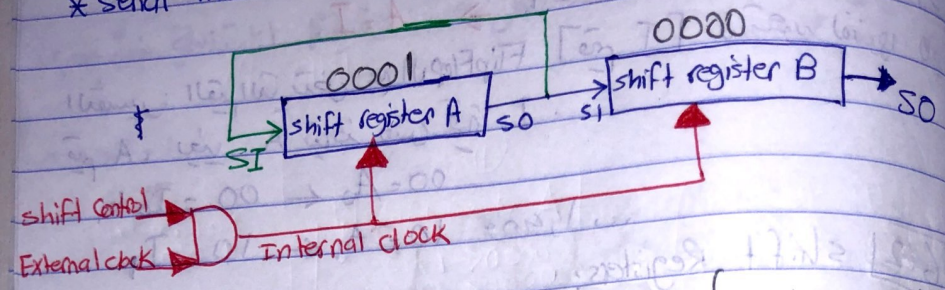


* الأصفار اللي باللون الأخضر بتمثل حالة الـFlipFlop قبل دخول الـSI يعني Initially (قيمة افتراضية)

* serial input → [يقوم بإدخال bit واحد وقتها كل مرة]

* تفسير الرسمة: لو دخل عندي serial input مقداره 101101 على أول D_FF رح يدخل أول bit منه وهو (1) وبالتالي رح ينتقل الـ(0) الموجود في الـD-FF الأولى إلى الـD-FF الثانية (اللي جمبها) وتحل مكانها (1) وهو أول bit في SI، وكذلك في DFF الثانية يحل محلها الـbit اللي موجود في D-FF الثانية وهكذا لحد ما يوصل الـbit لآخر DFF ويطلع عن طريق serial output، [بالمختصر، كل FF بتنقل الـbit اللي فيها للي بعدها وتتبادل الـbit الموجود في الـFF اللي قبلها.

* Serial Transfer From register A to register B.



* Initial Value for register A is 0001 (قيمة افتراضية)
* Initial Value for register B is 0000 (// //)

	0001	0000
*CLK1:	1000	1000
*CLK2:	0100	0100
*CLK3:	0010	0010
*CLK4:	0001	0001

* لما تصوي ال clock :
وهاد الصفر بطلع عن شكل Serial output

* بعد CLK4 وقفنا ، ~~خلص عدد العمليات متساوي~~

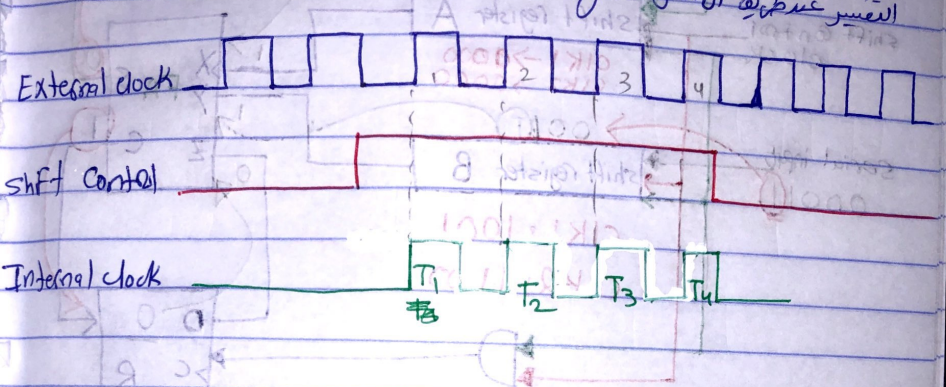
* التفسير : لما كانت ال Initial Value ل register A = 0001 و ل B = 0000 ، ملاحظ إنه السهم اللي طالع من A راجع عليها أساس إنه Serial input وكمان بروح نفسه بيدخل input S على B مشان يعمل shift to right بمقدار 1 bit وبالتالي ما تشتغل أول CLK رح تنقل ال (1) مرتين ، الأولى تروح عن أول منزلة من اليسار ل A والثانية لأول منزلة من اليسار ل B ، وبالتالي الإزاحة اللي بتصير بتخلي أول منزلة من اليمين ل B تنقل عن شكل SO يجي Serial output ومنضل هيك لحتى تصير عدد ال CLKs مساوي لعدد ال bits.

* Timing Diagram

* Notes about clocks:

→ after 4 clks in the previous example, the register stopped.

التفسير عن طريق الـ Timing diagram



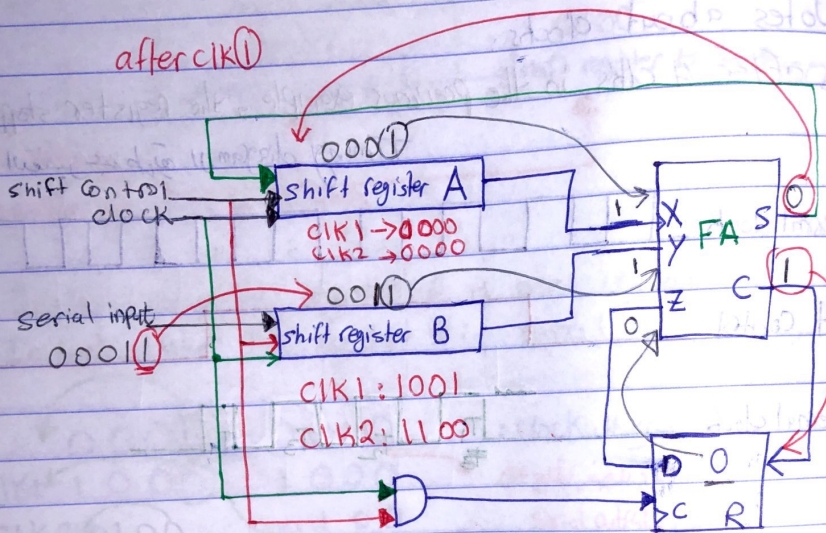
* الـ shift control اعتبرناه X (يعني أي قيمة) ودخلناه مع external clk على AND GATE وبما إنه الـ shift control أنا بتحكم فيه بأي قيمة تساوي (1) عندما تكون الـ CLK مساوية لعدد الـ bits يعني هون عندنا 4 bits بالفرض 4 clks والمعروف إن الـ CLK هي مستمرة و بتضل مشتغلة للانهاية، مشان هيك دخلناها على AND مع الـ shift control اللي بكون قيمته (1) فقط عند 4 clks أي عند 4 edges فقط.

$0 \cdot X = 0$

$1 \cdot X = X$

Scanned with CamScanner

* Serial Addition:



Initial Value for A = 0001

B = 0011

SI = 00011

D = 0

clk1 : 1

clk2 : 1

وهكذا حتى يبقى ال clocks

*التفسير: لما تدخل أوّل قيمة لـ A اللي هي 0001 وتشتغل ال clk، ال (1) رح يروح على X وكذلك B اللي = 0011، ال (1) رح يروح لـ Y، وكذلك D-FF حيث ال D رح يروح على Z وال FA رح يقوم بعملية الجمع، لذلك بطلع عنا sum بيرجع يروح على A و Carry بينزل على D-FF، أما B بيدخل عليها ال (1) اللي جاي من ال initial value لل serial input حيث = 0011، وهيك حتى باقي ال clks.

Register "A" + Register "B" = Register A + D

Sum ↓

النتيجة من الجمع بتروح على A

---

*الرسمة موجودة في السلايدات [سلايد رقم 15]: 4 bit Universal shift register

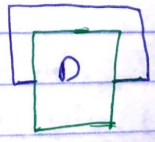
* منلاحظ من الرسمة إنّ ال mux مرتبط في ال D-FF

* when $S_1S_0 = 00$ → input = I_0 → لما عندي mux مرسوم، هيك بتكون D-FF

↳ مش بدّي A في D-FF مباشرة.

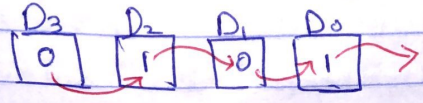
∴ input I_0 = A ∴ No change. (p.s = N.s)

No change ← يعني



* when $S_1S_0 = 01$ → input = I_1 → يمثل [serial input]: عملية إزاحة لليمين

101



تصبح 1

Scanned with CamScanner

* when $S_1S_0 = 10 \rightarrow I_2$ = input

وهو عمل إزاحة لليسار نفس I_1 ولكن بدل إزاحة لليمين،
إزاحة نحو اليسار.

* when $S_1S_0 = 11 \rightarrow I_3$ is the input

Mux مشبوك على I_3 يعني parallel load، يعني لو كانو القيم $I_3I_2I_1I_0$
1010 → in the D-FF, will be $\underset{D_3}{1}\ \underset{D_2}{0}\ \underset{D_1}{1}\ \underset{D_0}{0}$

Counters

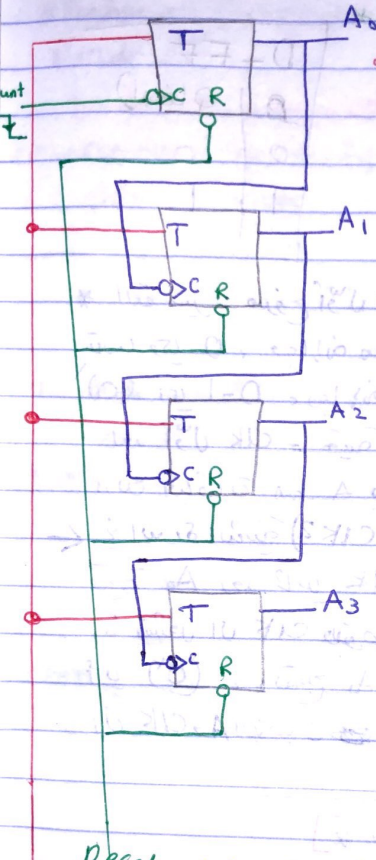
- Synchronous
  - ال FF كلهم إلهم نفس ال clk
- Ripple:
  - asynchronous:
    - كل FF إلها clk مختلفة، يعني ما في clk موحدة لكل ال FF

* Counters ex:
  - up Counter: 000 → 001 → 010 → ... 111
  - down Counter: 111 → 110 → ... → 000.

Scanned with CamScanner

6.3 Ripple Counters:

* أول شيء نسأل عنه هو ال edge ،
إذا انتقلت من 1 إلى 0 ث تشتغل ال CLK ،
غير هيك No change .
* ثاني شيء ، قيمة T .



T_FF	Q(t+1)
0	No change
1	Q'(t) [comp]

	A_3	A_2	A_1	A_0
initial values	0	0	0	0
CLK 1	0	0	0	1
CLK 2	0	0	1	0
CLK 3				
⋮				

* التفسير: أول CLK ، كانت الخانة الابتدائية لهم 0 وقيمة T هي (1) (مفعل) وبما أنها (1) ث تأخذ ال comp لـ A_0 وهي ستصبح 1 ، ولكن A_0 تمثل ال CLK لـ T-FF الثانية ، وبما أنها كانت 0 وأصبحت 1 يعني كان تعكس (1) وصارت (0) مش ث تشتغل ال CLK . أما ثاني CLK ، T=1 و A_0=1 ث تأخذ A_0' = 0 ، هنا أصبحت 0 بعدما كانت (1) ث تشتغل ال CLK وتنتقل إلى T-FF الثانية وقيمة T فيها = 1 ث تأخذ (A_1') فتصبح 1 فتكون A_1 قد تغيرت من 0 إلى 1 ∴ لا تشتغل ال CLK وقيمتها No change

رسمة ال D-FF (من السلايدات).

* من الرسمة قيمة D هي قيمة A'

D-FF	
D	Q(t+1)
0	0
1	1

	A_3	A_2	A_1	A_0
	0	0	0	0
CLK1)	0	0	0	1
CLK2)	0	0	1	0

* التفسير: صرح أول ما شي بشوف القيمة الابتدائية لـ A_0، هون تساوي 0، وبما إنه من الرسمة قيمة D هي A' ← قيمة D رح تصير $(0)'$ أي D = 1، وبما إنه ال N.S تساوي قيمة D، ← رح تصير قيمة A_0 بعد أول CLK = قيمة D = 1، ولكن A_0 ما تغيرت من 1 إلى 0 لذلك الباقيات من A بكونوا قيمتهم نفسها (No change).

← في الدورة الثانية (CLK2)، قيمة A_0 = 1، و D حتكون 0 وبالتالي قيمة A_0 بعد ثاني CLK رح تصير 0 زي D وانتقلت من 1 إلى 0 ← بتشتغل ال CLK وبتوصل لـ A_1، فتصبح قيمة D في A_1 تساوي 1 أي $(0)'$، وتصبح A_1 = 1، ولكنها لم تنتقل من 1 إلى 0 ← بتوقف ال CLK و A الباقيات باخذ نفس قيمتهم اللي كانو عليها.

JK

* بالرسمة محطين JK دائماً قيمتهم 11

يعني الحالة اللي بتصير هي comp

J	K	N.S
0	0	No change
0	1	0
1	0	1
1	1	Complement.

← عبارة عن حالة JKFF J بعد أول CLK.

	Q_3	Q_2	Q_1	Q_0
	0	0	0	0
CLK1)	0	0	0	1
CLK2)	0	0	1	0

← صارت (1) لأنه اشتغلت ال CLK وأعطت comp لل P.S

* بما أنه قيمة JK هي 11 من السؤال، ← N.S هي comp لل P.S حسب الجدول.

Scanned with CamScanner

* Synchronous Counters :

* slide 24

Synchronous → كل ال FF لهم نفس ال clk

أولاً حددنا إذا Ripple ، Synchronous ثم منشوف إذا positive أو negative edge

* في هذا المثال [positive edge]

* Count enable = 1 : في هذا المثال

* فرق بسيط : لما تكون ال Counter نوعه ripple ، كنا أول اشي نسأله نحنه هو قيمة ال clk ثم بعدها عند قيمة J أو D حسب نوع FF

بينما ال Synchronous منبدأ بالسؤال عن قيمة ال D أو JK أو T.

A_3	A_2	A_1	A_0		J	K	Q(t+1)
0	0	0	0	صفرون كلهم سواء بالجهاز	0	0	No chang
0	0	0	1	← (clk) comp.	0	1	0
0	0	1	0	→ 1.1 = 1 : (comp)	1	0	1
					1	1	$Q'(t)$

no change

No change

عندما تشتغل ال clk الأولى ينقل الانبوت (1) في ال Counter على أول FF فتأخذ كلاً من J, K قيمة 1 1 ، وحسب الجدول المعروف ، بتكون A_0 جديدة (N.S) مساوية للمتممة قيمتها السابقة أي A_0 بتصبح = $\bar{0}$ = 1

ثم تنقل بنفس اللحظة قيمة A_0 الابتدائية إلى ال FF الثانية ، [* مأخذ من القيمة الابتدائية مش اللي بعد ما تدق الساعة لأنه كلهم بشتغلوا مع بعض

4-bit Binary up-down Counters. Slide 25

T	Q(t+1)
0	No chang
1	complement

* up-down = 00

A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	0

→ T=0 ∴ No Change

[* طبعًا منعتمد على القيم الابتدائية عند أول CLK.]

* UP-down = 01

A_3	A_2	A_1	A_0
0	0	0	0
1	1	1	1
1	1	1	0

→ T=1 ∴ comp

No change

* up-down = 11

A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1

---

* Slide 26

Clear	clock	load	Count	function
0	X	X	X	Clear to 0, $A_x = 0$
1	↑	1	X	Load inputs (نفس الانبوتس بيدخلوا على الكاونتر)
1	↑	0	1	UP Count
1	↑	0	0	No change.

شرح الجدول :

Scanned with CamScanner

* شرح الجدول:

* Clear = 0

Count, load, clock → don't care.

أي عندما تكون قيمة Clear = 0 ، لا يهتم قيمة باقي المدخلات ، غير الناتج يكون 0.

* clear = 1 , load = 1

Count → don't care.

هون عندما ال Load = 1 دائماً الناتج يكون هو نفسه ال inputs ، يعني بروح بدخل قيم ال inputs نفسهم

* clear = 1 , Load = 0 , Count = 1

* don't care فش

هون يتم زيادة بمقدار (1) ع العدد الموجود ، يعني 0000 تصبح 0001 وهكذا.

* clear = 1 , load = 0 , Count = 0

* don't care فش

ما بصير تغيير ، N.S = P.S

* على حسب المثال الموجود في اللايدات:

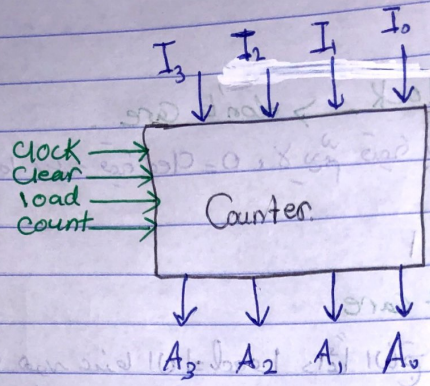
* clear = 1, Load = 1

if input = 0001 , يدخل ويخزنه 0001

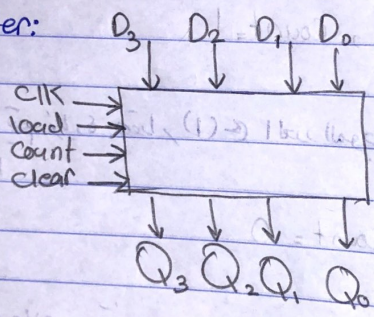
clear = 1 , load = 0 , count = 1 → up count

A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1

Scanned with CamScanner



*BCD Counter:



* Clear = 0 , load = 1 , count = 0 :
$Q_3 = 0$, $Q_2 = 0$, $Q_1 = 0$, $Q_0 = 0$

* لازم ال clear = 0

* Clear = 1 , En = 1 , load = 0 , count = 1
لح تزيد (1) عم الابو تس

* في الرسمة سلايد 27 :

Load بيدخل عليه (Q_0 . Q_3) ، طول ما هم قيمتهم صفر، بضل ال load = 0 وبالتالي بضل نزيد (1) على كل قيمة ، لحد ما $Q_0 = Q_3 = 1$ عندها يتوقف عن الزيادة .

AND Gate.

* لو كان

Q_3	Q_2	Q_1	Q_0	
0	0	0	0	← 0 = 0.0 ، 0 = ~~load~~ ، نزيد (1).
0	0	0	1	← 0 = 0.1 ، 0 = Load ، // //
0	0	1	0	← 0 = 0.0 ، 0 = Load ، // //
⋮				
1	0	0	1	← 1 = 1.1 ، 1 = Load ، ح يدخل قيمة ال input .
0	0	0	0	لذلك ح يرجع أصفار

* لو طلب منا يعد من 12 - 0 يعني من 1100 - 0000

← هون ال And ح تصير بين Q_3 و Q_2 مشان تعملنا قيمة (1) لحتى يتوقف ال Load ويصبح قيمته 1 ، مما يوقف عملية العد.

* لو طلب من 12 - 3 أي 1100 - 0011

Q_3	Q_2	Q_1	Q_0
0	0	0	0
0	0	0	1
1	1	0	0

ثم هنا تصبح قيمة Load = 1 وبالتالي يصبح الاوتبوت مساوي لقيمة $D_3 D_2 D_1 D_0$ وكان بده من 3 مش من 0 منذ خل قيم D هي 0011 على ال Counter

Q_3	Q_2	Q_1	Q_1	
0	0	0	0	
0	0	0	1	
⋮				
1	1	0	0	← 1.1 = 1 ، يتوقف عن العد.

Scanned with CamScanner