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 بمشي 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 $A_x = 0$, دخل كل مدخل صفر.

* parallel → بيدخل ع ال circuit بيدخل أكثر من bit سوا ←
* serial → (بيدخل فقط bit واحد ثم يليه الثاني وهكذا (مش مع بعض بيدخلوا)) ←

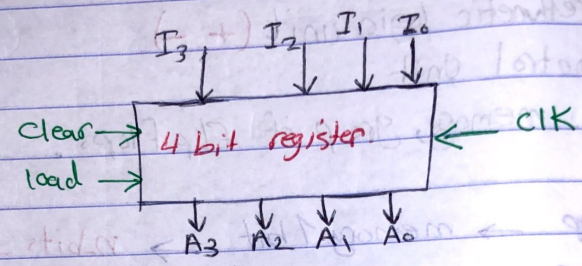
* أي اشي بلش يعني لطلع momentum لمدة ثاني هو P.S ، وأي اشي بين كذا وكذا هو N.S اشي طبيعي.

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

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

* Load = 1

A = I



[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 يهمّنا قيمة ال load أو ال clk وبعطي قيم $A[0:3]$ تكون تساوي 0.

[2] clear = 1 , load = 0

∴ No change , A(t) = A(t+1)

التفسير: ال flip flop بضل بحافظ الحالة مخزّنة نفس القيمة الابتدائية اللي كان عليها ، يعني لو كان مخزّن 0000 بضل 0000 طالما ال load يساوي 0.

Scanned with CamScanner

STUDENTS-HUB.com

3| clear = 1 , load = 1 → A = I

التفسير: القيم التي تدخل ع ال FlipFlop [قيم I] ستعود لنفسها هي قيم A، يعني A تأخذ نفس قيمة I .

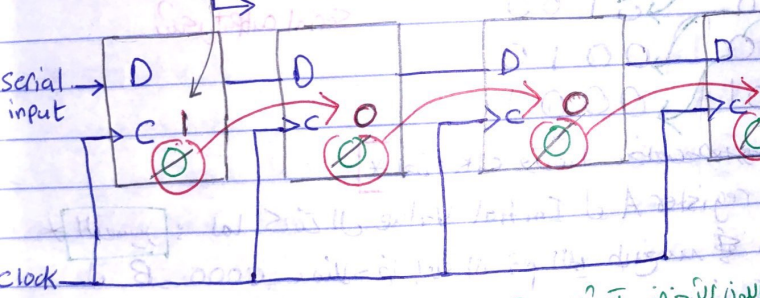
إذا $I_0 = 00 \leftarrow A_0 = 00$

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

6.2 Shift Registers :(إزاحة bits)

* لو عندي هاي ال Circuit :

* shift to the right :



SI = 101101 (أول bit)

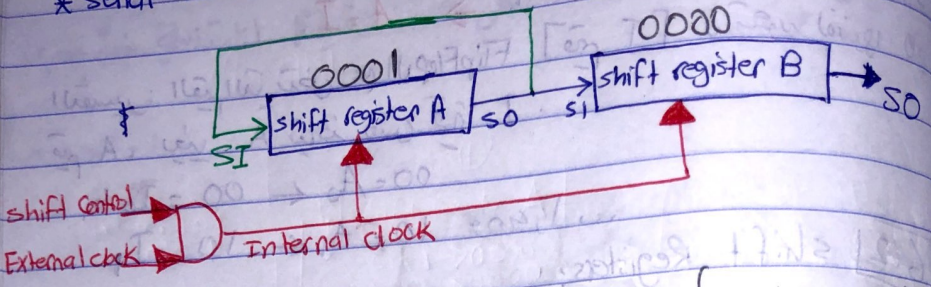
* الأصفار اللي باللون الأخضر بتمثل حالة ال FlipFlop قبل دخول ال SI يعني Initially (قيمة افتراضية)

[* serial input → بنقوم بإدخال bit واحدة كل مرة]

* تفسير الرسمة: لو دخل عندي serial input مقداره 101101 مع أول D_FF رح يدخل أول bit منه وهو (1) وبالتالي رح ينتقل ال(0) الموجود في ال D-FF الأول إلى ال D-FF الثانية (اللي جمبها) وتكون مكانها (1) وهو أول bit في SI ، وكذلك في DFF الثانية ع حل محلها ال bit اللي موجود في D-FF الثانية وهكذا لحد ما يوصل ال bit لآخر D-FF ويطلع ع هيئة serial output ، [باختصار، كل FF بتنقل ال bit اللي فيها للي بعدها وبتأخذ ال bit الموجود في ال FF اللي قبلها .

STUDENTS-HUB.com
Scanned with CamScanner

* serial Transfer From register A to register B.



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

* لما تضوي ال clock :

0001 0000

*clk1: 1000 1000

*clk2: 0100 0100

*clk3: 0010 0010

*clk4: 0001 0001

هاد الصفر بطلع ع شكل serial output

* بعد clk4 وقفنا ، ~~[illegible]~~

* التفسير : لما كانت ال Initial Value لـ register A = 0001 و ال B = 0000 ، ملاحظ إنه السهم اللي طالع من A راجع عليها أساس إنه serial input وكأنه بروح لنفسه بيدخل كـ input ع B مشان نعمل shift to right بمقدار 1bit ، وبالتالي لما تشتغل أول clk رح تنتقل ال (1) مرتين ، الأولى تروح ع أول منزلة من اليسار لـ A و الثانية لأول منزلة من اليسار لـ B ، وبالتالي الإزاحة اللي بتصير بتخلي أول منزلة من اليمين لـ B تنتقل ع شكل SO يعني Serial output ومنضل هيك لحد ما ~~[illegible]~~ تصير عدد ال clks مساوي لعدد ال ~~[illegible]~~ bits.

STUDENTS-HUB.com

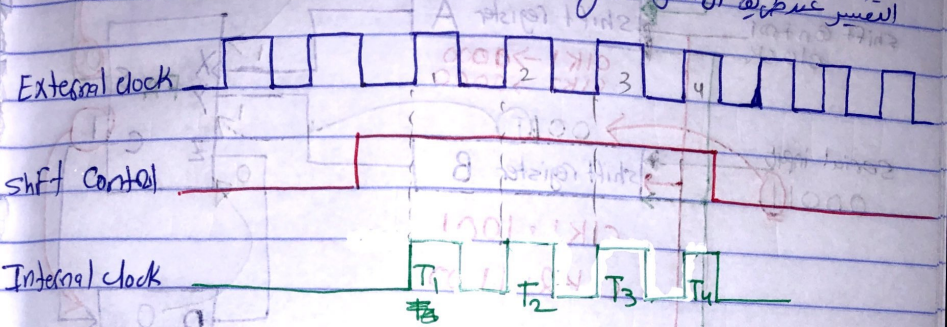
Scanned with CamScanner

*Timing Diagram

*Notes about clocks:

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

التفسير عن طريق ال Timing diagram



*ال shift control اعتبرناه X (يعني أي قيمة) ودخلناه مع ال external clk على AND GATE وبما إنه ال shift control أنا بتحكم فيه، بخلّي قيمته تساوي (1) عندما تكون ال clk مساوية لعدد ال bits، يعني هون عنا 4 bits، بلزمنا 4 clks، المعروف إنه ال clk هي مستمرة و بتضل مشتغلة للمالانهاية، مشان هيك دخلناها ع AND مع ال shift control اللي بكون قيمته (1) فقط عند 4 clks أي عند 4 edges فقط.

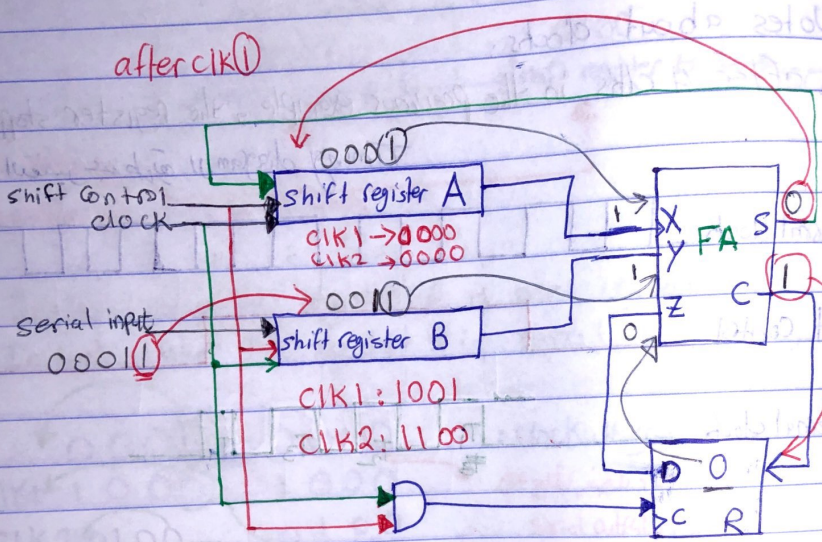
$0 \cdot X = 0$

$1 \cdot X = X$

Scanned with CamScanner

STUDENTS-HUB.com

* Serial Addition:



Initial Value for A = 0001 — clk1 : 1

B = 0011 — clk2 : 1

SI = 00011

D = 0 — وهكذا حتتابع ال clocks

Scanned with CamScanner

STUDENTS-HUB.com

* التفسير: لما تدخل أول قيمة ل A اللي هي 0001 وتشتغل ال clk ال (1) رح يروح على X، وكذلك B اللي = 0011، ال (1) رح يروح ل Y، وكذلك D-FF صفر ال 0 رح يروح على Z والـ FA رح تقوم بعملية الجمع، لذلك رح يطلع عنا sum بيرجع يروح ع A و carry بينزل ع D-FF، أما B بيدخل عليها ال (1) اللي جاي من ال initial value لل serial input حيث = 00011، وهكذا حتى باقي ال clks.

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

Sum

النتيجة من الجمع بتروح ع A

---

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

* ملاحظ من الرسمة إنه ال mux مرتبط في ال D-FF

* when $S_1S_0 = 00$ → input = I_0 → لو ما عندي mux مرسوم، صفين تكون D-FF

لـ مشبوك مع 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] متعلق بإزاحة لليمين

STUDENTS-HUB.com

Scanned with CamScanner

*When $S_1S_0 = 10 \rightarrow I_2$ = input.

وهو عمل إزاحة لليسار يعني I_1 ولكن بدل إزاحة لليمين.
إزاحة نحو اليسار

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

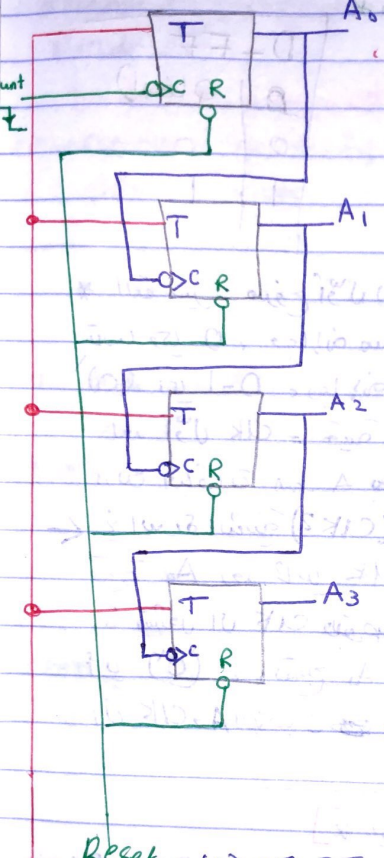
الـ Mux مشبوك على I_3 يعني parallel load ، يعني لو كانوا القيم $I_3 I_2 I_1 I_0$
1 0 1 0 → in the D-FF, will be $\underset{D_3}{1}\ \underset{D_2}{0}\ \underset{D_1}{1}\ \underset{D_0}{0}$

Counters

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

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

6.3 Ripple Counters:



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

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 الحالة الابتدائية لهم 0 وقيمة T الهم (1) (محطوط) وبما أنها (1) ⇐ 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 ⇐ No change يعني لازم تكون (1) وصارت (0) حتى تشتغل ال CLK . أما ثاني CLK ، T=1
و $A_0 = 1$ ⇐ نأخذ $A_0' = 0$ ، هنا أصبحت 0 بعدما كانت (1) ⇐ تشتغل ال CLK وتنتقل إلى T-FF الثانية وقيمة T فيها = 1 ⇐ نأخذ (A_1') فتصبح 1
فتكون A_1 قد تغيرت من 0 إلى 1 ⇐ لا تشتغل ال CLK وقتها ⇐ No change

STUDENTS-HUB.com

Scanned with CamScanner

رسمة ال D-FF (مع السلايدات).

* من الرسمة قيمة D هي قيمة A'

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: مروح أول اشي نشوف القيمة الابتدائية ل A_0 ، هو بده تساوي 0 ، وعا إنه من الرسمة قيمة D هي A' ، قيمة D رح تصير (0)' أي D=1 ، وعا 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 الحالة الثانية (Clk 2) ، قيمة A_0 = 1 ، و D حتكون 0 وبالتالي قيمة A_0 بعد ثاني clk رح تصير 0 زي D و انتقلت من 1 إلى 0 ، نشغل ال clk ونوصل ل A_1 ، فتصبح قيمة D في A_1 تساوي 1 أي (0)' ، وتصبح A_1 = 1 ، ولكنها عم تنتقل من 1 إلى 0 ، نتوقف ال clk و الباقيات A باخذو نفس قيمتهم اللي كانو عليها.

JK

* بالرسمة معطيني JK دائما قيمتهم 11

يعني الحالة اللي بتمثل هي comp

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

	Q_3	Q_2	Q_1	Q_0
	0	0	0	0
Clk1)	0	0	0	1
Clk2)	0	0	1	0

عبارة عن حالة ← JK FF بعد أول clk.

صارت (1) لانه اشتغلت ال clk وأعطت comp ل P.S

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

STUDENTS-HUB.com

Scanned with CamScanner

* Synchronous Counters :

* Slide 24

Synchronous → كل ال FF إلهم نفس ال clk

أولاً وجدنا إذا Ripple أو Synchronous، ثم منشوف إذا positive أو negative edge edge .

* في هذا الشكل [positive edge]

* Count enable = 1 : في هذا ال

* فرق بسيط: لما يكون ال Counter نوعه ripple، كنا أول اشي نسأله عنه هو قيمة ال clk ثم بعدها عن قيمة J K أو 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	
	0	0	0	0	← [illegible]
no change	0	0	0	1	→ comp. ← (clk 1
No change	0	0	1	0	→ 1.1 = 1 ∴ (comp)

J	K	Q(t+1)
0	0	No change
0	1	0
1	0	1
1	1	Q'(t)

عندما تشتغل ال clk (الأولى) يدخل الإنبوت (1) ع ال Counter ع أول FF فتأخذ كلاً من J,K قيمة 1 1، وحسب الجدول المعروف، ستكون A_0 الجديدة (N.S) مساوية المتممة لقيمتها السابقة أي A_0 ستصبح $= \bar{0} = 1$

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

STUDENTS-HUB.com

Scanned with CamScanner

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

[* طبعاً 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

شرح الجدول :

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

STUDENTS-HUB.com

Scanned with CamScanner

* شرح الجدول:

* Clear = 0

Count, load, clock → don't care.

أي عندما تكون قيمة clear = 0 ، لا يهمّ قيمة باقي المُدخلات ، دائماً الناتج بكون 0 .

* clear = 1 , load = 1

Count → don't care.

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

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

* فش don't care

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

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

* فش don't care

ما بصير تغيير ، N.S = P.S

* ع حسب المثال الموجود في السلايدات :

* clear = 1 , Load = 1

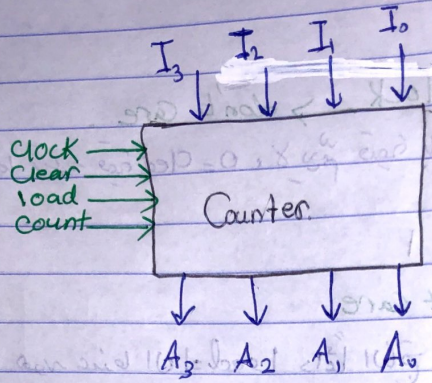
if input = 0001 , بيدخل وبخزنه 0001

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

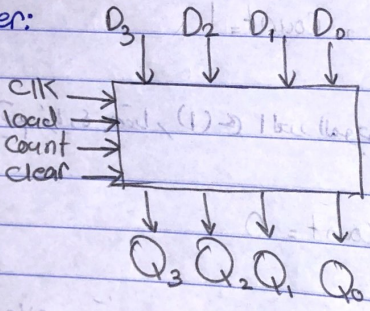
A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1

STUDENTS-HUB.com

Scanned with CamScanner



*BCD Counter:



* clear = 0 , load = 1 , count = 0:

$Q_3 = 0$, $Q_2 = 0$, $Q_1 = 0$, $Q_0 = 0$ *لأنه ال clear = 0

* clear = 1 , En = 1, load = 0, count = 1

رح تزيد (1) عم الابوتس

STUDENTS-HUB.com

Scanned with CamScanner

* في الرسمة سلايد 27:

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

AND Gate.

* لو كانو

Q_3	Q_2	Q_1	Q_0	
0	0	0	0	← 0.0 = 0 ∴ = 0 ∴ نزيد (1).
0	0	0	1	← 0.1 = 0 ∴ Load = 0 ∴ // //
0	0	1	0	← 0.0 = 0 ∴ Load = 0 ∴ // //
⋮				⋮
1	0	0	1	← 1.1 = 1 ∴ Load = 1 ∴ رح يدخل بعد ال input.
0	0	0	0	لذلك رح يرجع أصفار

* لو طلب منا بعد من 0 – 12 يعني من 0000 – 1100

← هون ال And رح تصير بين Q_3 و Q_2 مش ما نتحملنا قيمة (1) لحتى يتوقف ال Load ويصبح قيمته 1، مما يوقف عملية العدّ.

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 العدّ.

* لو طلب من 3 – 12 أي 0011 – 1100

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 $D_3 D_2 D_1 D_0$ ولأنه بده من 3 مش من 0 منّدخل قيم D هي 0011 ع ال Counter

STUDENTS-HUB.com

Scanned with CamScanner