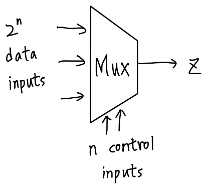
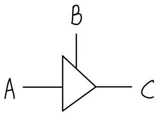
L9

Multiplexers：多選一



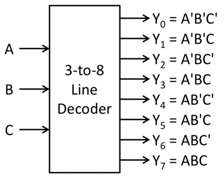
Three - State Buffer：連接 output



B	C
0	Z
1	A

Decoder：

generate all minterms of input variables



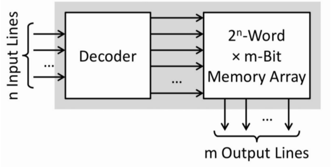
e.g. ABC = 010
Y_2 = 1 , other = 0

(priority) Encoder：

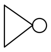
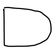
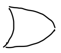
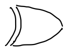
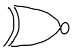
從 Y_7 開始看，如果有 1 就直接 output

ROM：stores an array of binary data

輸入 binary 輸出特定 word



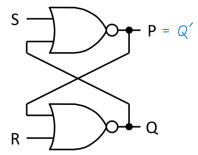
Appendix

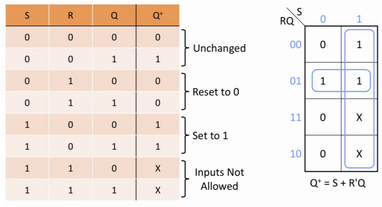
NOT		
AND		乘
OR		加
XOR		⊕ 不一樣為 1
XNOR		≡ 一樣為 1

B11902038

L11

S-R Latch



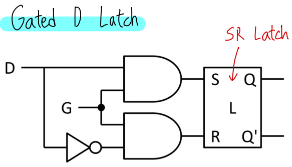


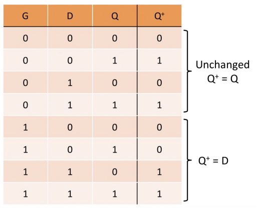
S	R	Q	Q^+	
0	0	0	0	Unchanged
0	0	1	1	
0	1	0	0	Reset to 0
0	1	1	0	
1	0	0	1	Set to 1
1	0	1	1	
1	1	0	X	Inputs Not Allowed
1	1	1	X	

RQ \ S	0	1
00	0	1
01	1	1
11	0	X
10	0	X

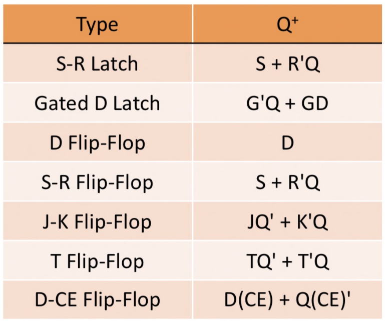
$Q^+ = S + R'Q$

Gated D Latch





G	D	Q	Q^+	
0	0	0	0	Unchanged $Q^+ = Q$
0	0	1	1	
0	1	0	0	
0	1	1	1	
1	0	0	0	$Q^+ = D$
1	0	1	0	
1	1	0	1	
1	1	1	1	

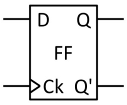


Type	Q^+
S-R Latch	S + R'Q
Gated D Latch	G'Q + GD
D Flip-Flop	D
S-R Flip-Flop	S + R'Q
J-K Flip-Flop	JQ' + K'Q
T Flip-Flop	TQ' + T'Q
D-CE Flip-Flop	D(CE) + Q(CE)'

Flip–Flop 就是 edge–triggered 的 Latch
edge–triggered：只有時脈變化時才作用
(rising–edged 0 –> 1)

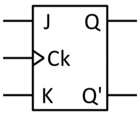
D Flip-Flop

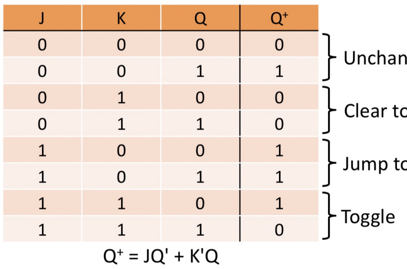
$Q^+ = D$ (rising-edge trigger)



D	CK	Q	Q^+
D	0→1	X	D
X	0、1	Q	Q

J-K Flip-Flop



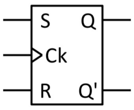


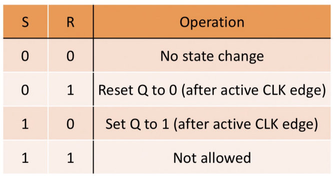
J	K	Q	Q^+	
0	0	0	0	Unchanged
0	0	1	1	
0	1	0	0	Clear to 0
0	1	1	0	
1	0	0	1	Jump to 1
1	0	1	1	
1	1	0	1	Toggle
1	1	1	0	

$Q^+ = JQ' + K'Q$

S-R Flip-Flop：

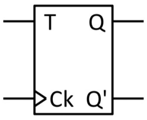
$Q^+ = S + R'Q$

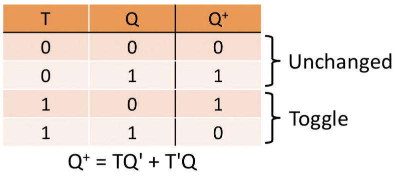




S	R	Operation
0	0	No state change
0	1	Reset Q to 0 (after active CLK edge)
1	0	Set Q to 1 (after active CLK edge)
1	1	Not allowed

T Flip-Flop





T	Q	Q^+	
0	0	0	Unchanged
0	1	1	
1	0	1	Toggle
1	1	0	

$Q^+ = TQ' + T'Q$

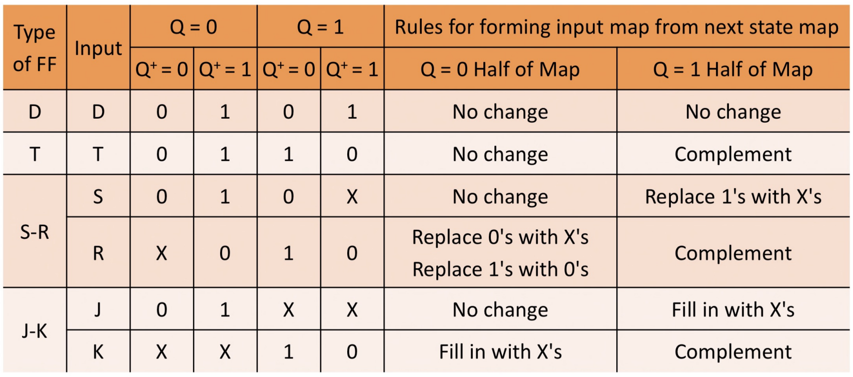
L12

Counter Design：

1. 畫 Next State table
2. 畫 K-map of $C^+ B^+ A^+$
3. 取決於 Flip-Flop 種類：當 Q → Q^+ 時 S.R 會是什麼?
4. 用 Excitation table
   結合 Next State table
   得出 Flip-Flop equation
   $C \rightarrow C^+$ 得 S_c R_c

Determine the flip flop input equations from the **next-state equations** using K-maps

➢ Always copy X's from next state maps onto input maps first



Type of FF	Input	Q = 0		Q = 1		Rules for forming input map from next state map	
		$Q^+ = 0$	$Q^+ = 1$	$Q^+ = 0$	$Q^+ = 1$	Q = 0 Half of Map	Q = 1 Half of Map
D	D	0	1	0	1	No change	No change
T	T	0	1	1	0	No change	Complement
S-R	S	0	1	0	X	No change	Replace 1's with X's
	R	X	0	1	0	Replace 0's with X's Replace 1's with 0's	Complement
J-K	J	0	1	X	X	No change	Fill in with X's
	K	X	X	1	0	Fill in with X's	Complement

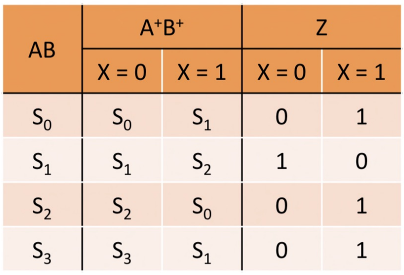
L13

State Table

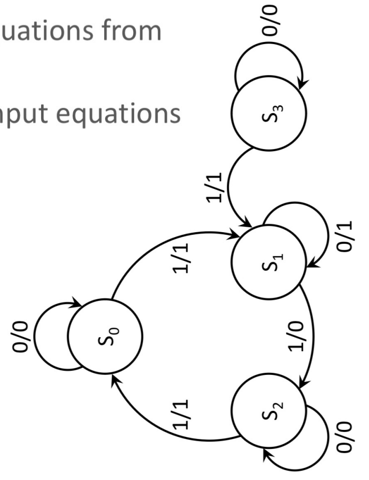
1. Determine the flip-flop input equations and the output equations from the circuit
2. Derive the next-state equation for each flip-flop from its input equations
3. Plot a next-state map for each flip-flop
4. Combine these maps to form the state table

Moore :
output depend on present state

Mealy :
depend on present state & input



AB	A^+B^+		Z	
	X = 0	X = 1	X = 0	X = 1
S_0	S_0	S_1	0	1
S_1	S_1	S_2	1	0
S_2	S_2	S_0	0	1
S_3	S_3	S_1	0	1



❑ Designing a sequential circuit

- ➢ Construct a state graph or state table (Unit 14)
- ➢ Simplify it (Unit 15)
- ➢ Derive flip-flop input equations and output equations (Unit 12)

Construct a state graph (L14)

❑ Steps

- ➢ Construct sample sequences to help you understand the problem
- ➢ Determine under what conditions it should reset
- ➢ If only one or two sequences lead to a nonzero output, construct a partial state graph
  - Another way, determine what sequences or groups of sequences must be remembered by the circuit and set up states accordingly
- ➢ Each time you add an arrow to the state graph, determine whether it can go to one of the previously defined states or whether a new state must added
- ➢ Check your graph to make sure there is one and only one path leaving each state for each combination of values of the input variables
- ➢ When your graph is complete, verify it by applying the input sequences formulated in step 1

Implication Table

- ❑ Draw an empty table, where each square represents a pair
- ❑ If outputs are different, give it an X (impossible!)
- ❑ Write down the implied pair in the square
- ❑ Delete self-implied pairs (redundant)

Present State	Next State X = 0	Next State X = 1	Present Output
A	D	C	0
B	F	H	0
C	E	D	1
D	A	E	0
E	C	A	1
F	F	B	1
G	B	H	0
H	C	G	1

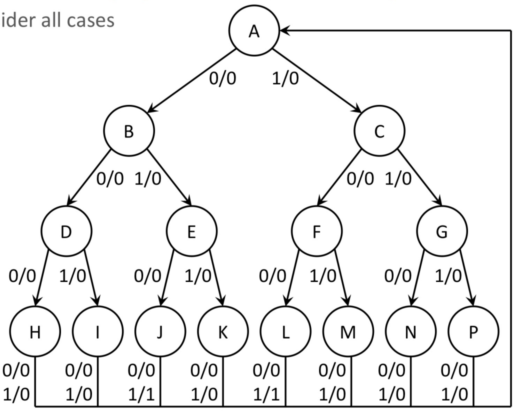
如果 A = B
那麼 D = F, C = H (不可能)

B	~~D-F C-H~~						
C	X	X					
D	~~A-D~~ C-E	~~A-F E-H~~	X				
E	X	X	~~C-E~~ A-D	X			
F	X	X	~~E-F B-D~~	X	~~C-F A-B~~		
G	~~B-D C-H~~	~~B-F~~	X	~~A-B E-H~~	X	X	
H	X	X	~~C-E D-G~~	X	~~A-G~~	~~C-F B-G~~	X
	A	B	C	D	E	F	G

簡化 (L15) Find equivalent state !

"Another" state graph for "0101/1001" sequence detector

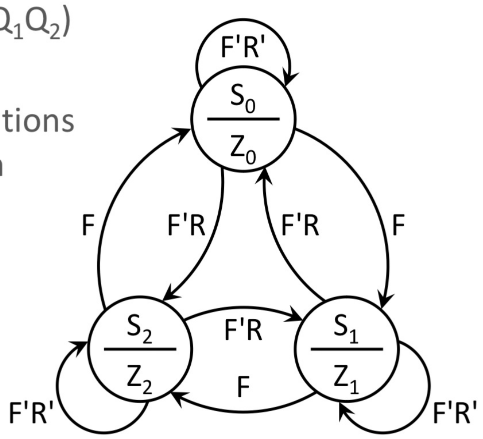
- ➢ Consider all cases



Input Sequence	Present State	Next State		Present Output	
		X = 0	X = 1	X = 0	X = 1
Reset	A	B	C	0	0
0	B	D	E	0	0
1	C	F → E	G → D	0	0
00	D	H	I → H	0	0
01	E	J	K → H	0	0
10	F	L → J	M → H	0	0
11	G	N → H	P → H	0	0
000	H	A	A	0	0
001	I	A	A	0	0
010	J	A	A	0	1
011	K	A	A	0	0
100	L	A	A	0	1
101	M	A	A	0	0
110	N	A	A	0	0
111	P	A	A	0	0

❑ One-hot state assignment: One flip-flop for each state

- ➢ Example: 3 flip-flops for 3 states ($Q_0Q_1Q_2$)
  - S_0 = 100, S_1 = 010, S_2 = 001
- ➢ Write next-state and output (Z) equations directly by inspecting the state graph
  - $Q_0^+ = F'R'Q_0 + F'RQ_1 + FQ_2$
  - $Q_1^+ = F'R'Q_1 + F'RQ_2 + FQ_0$
  - $Q_2^+ = F'R'Q_2 + F'RQ_0 + FQ_1$
  - $Z = Z_0Q_0 + Z_1Q_1 + Z_2Q_2$



B11902038