



# 序向電路設計

## 6-1 簡介

在組合邏輯電路中，電路的輸出只由目前的輸入決定；在序向(邏輯)電路 (Sequential circuit)，電路的輸出除了由目前的輸入決定外，也和先前的電路輸出有關。

換句話說，一個具有記憶的電路稱為序向電路。其主要記憶元件為正反器，因Flip-Flop具有記憶功能。其型式諸多，然各種不同正反器間最大不同，在於它們所擁有的輸入數目，以及這些輸入對他們原來狀態的影響，以下復習常用的幾種類型：

## 1. SR正反器：

特性表：分析及定義正反器的動作。並描述在已知輸入情況下的次態。

SR	Q(t+1)
00	Q(t)
01	0
10	1
11	?

由變數引用圖法得：

		R	
		0	1
S	0	Q(t)	0
	1	1	×

特性方程式：

$$Q(t+1) = S + R' Q(t)$$

$$SR = 0$$

或

Q(t)	S	R	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	?
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	?

卡諾圖化簡：

		SR			
		00	01	11	10
Q(t)	0	0	0	×	1
	1	1	0	×	1

$$Q(t+1) = S + R' Q(t)$$
$$SR = 0$$

特性方程式(or次一狀態方程式)：以目前狀態和輸入值表示出正反器次一狀態的方程式謂之。

然在序向電路設計過程中，通常我們都已知從現態到次態的變遷情況，而希望找出正反器的輸入條件以符合變遷條件的要求。因此我們需要一個表，它列出每一狀態變化所需的輸入條件。這個表稱之為激發表 (excitation table)。

SRFF激發表：

Q(t) $\rightarrow$ Q(t + 1)		S	R
0	0	0	×
0	1	1	0
1	0	0	1
1	1	×	0

## 2. JK 正反器：

特性表：

J K	$Q(t+1)$
0 0	$Q(t)$
0 1	0
1 0	1
1 1	$Q'(t)$

激勵表：

$Q(t) \rightarrow Q(t+1)$		J K
0	0	0 $\emptyset$
0	1	1 $\emptyset$
1	0	$\emptyset$ 1
1	1	$\emptyset$ 0

特性方程式：

		K		0	1
		J		0	1
Q(t+1)	0	Q(t)		0	
	1	1		Q'(t)	

$$Q(t+1) = J Q'(t) + K' Q(t)$$

(變數引入圖法)

		JK			
		00	01	11	10
Q(t+1)	Q(t)	00	01	11	10
	0	0	0	1	1
	1	1	0	0	1

或

Q(t)	J	K	Q(t+1)
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	0

### 3. D型正反器：

特性表：

D	$Q(t+1)$
0	0
1	1

特性方程式：

$$Q(t+1) = D$$

激發表：

$Q(t) \rightarrow Q(t+1)$		D
0	0	0
0	1	1
1	0	0
1	1	1



## 4. T型正反器

特性表：

T	$Q(t+1)$
0	$Q(t)$
1	$Q'(t)$

激發表：

$Q(t) \rightarrow Q(t+1)$		T
0	0	0
0	1	1
1	0	1
1	1	0

特性方程式：

$Q(t+1)$

T	
0	$Q(t)$
1	$Q'(t)$

$$\begin{aligned} Q(t+1) &= T' Q(t) + T Q'(t) \\ &= T \oplus Q(t) \end{aligned}$$

或

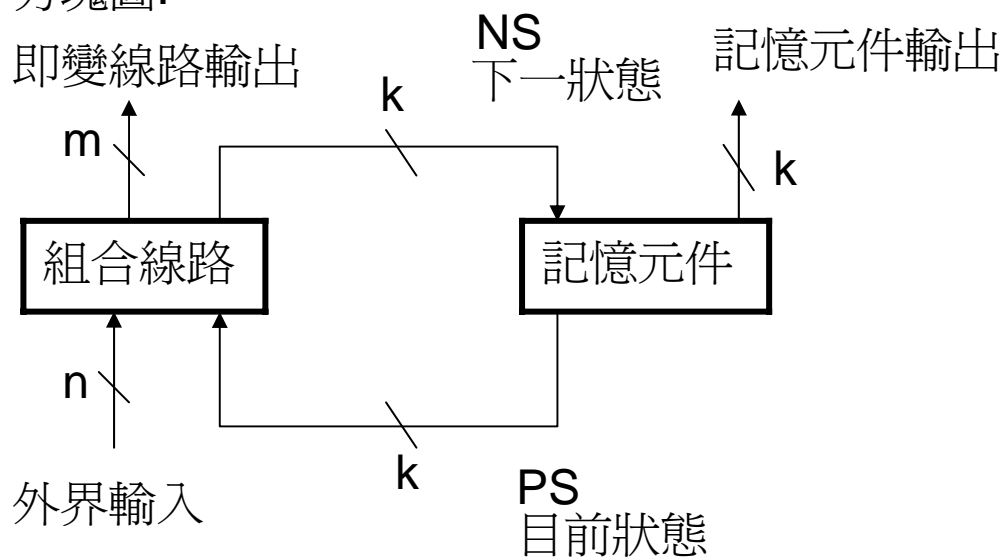
$Q(t) \rightarrow T$		$Q(t+1)$
0	0	0
0	1	1
1	0	1
1	1	0

$$\begin{aligned} Q(t+1) &= Q'(t) T + Q(t) T' \\ &= T \oplus Q(t) \end{aligned}$$

## 6-2 序向電路基本模式：

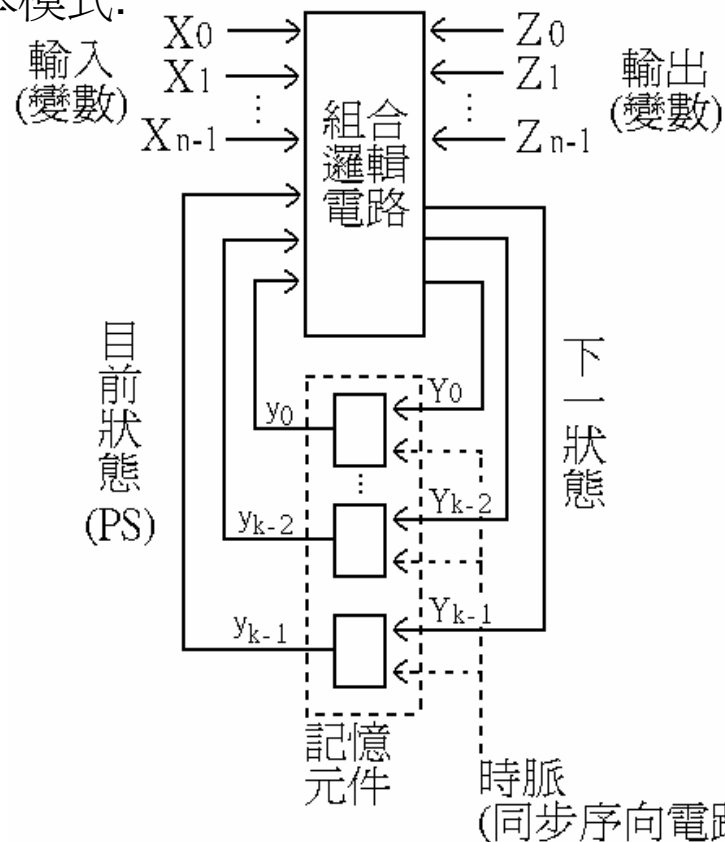
序向電路為具有計憶元件的邏輯電路，其方塊圖或基本模式如下圖：


方塊圖：




在序向電路中，當考慮電路的時序關係時，可以分成同步與非同步兩種類型。

基本模式：





同步序向電路（時脈式同步網路（clocked synchronous network））：在這種網路內，狀態只能夠隨著時脈的出現而改變，此種狀態改變如果需要數個正反器改變其狀態，則將會在同一時刻改變狀態，因為它們是根據相同的時脈同步運作。



非同步序向電路(網路): 此網路不根據共同時脈而同步運作；輸入信號改變以後，網路幾乎可以立刻改變狀態。此時如果有數個正反器必須改變狀態，則無法保證這些正反器會在同一時刻改變狀態。

亦即電路的行為只受輸入(變數)改變的次序影響而且可以在任何時刻發生。因此在這種電路中所用的記憶元件為非時脈元件，以便輸入改變時，即發生轉態。


常用的記憶元件為延遲元件(delay element)與SR—門閤(SR latch)。(最簡單的方法是限制輸入變數的改變方式，一次只允許一個而且連續地改變，必須等電路穩定後才發生。)

比較：由於非同步序向網路的設計方法牽涉到時序(timing)上的問題。因而在此我們只討論同步序向電路。設計良好的同步網路不會有時序上的問題，因為外部的輸入信號改變了以後，在時脈到達以前，我們已經準備了足夠長的時間，使得所有正反器的輸入都能夠到達穩定的值。

## 6-3 序向電路表現方式

序向電路的表示方式有：邏輯方程式(即交換函數)、狀態圖(state diagram)、狀態表(state table)、時序序列(timing sequences)，與時序圖(timing diagrams)。

利用邏輯方程式描述序向電路，並不容易瞭解電路的行為。因為這表示方式通常只止於理論上的探討，並不適合做電路的設計或分析。



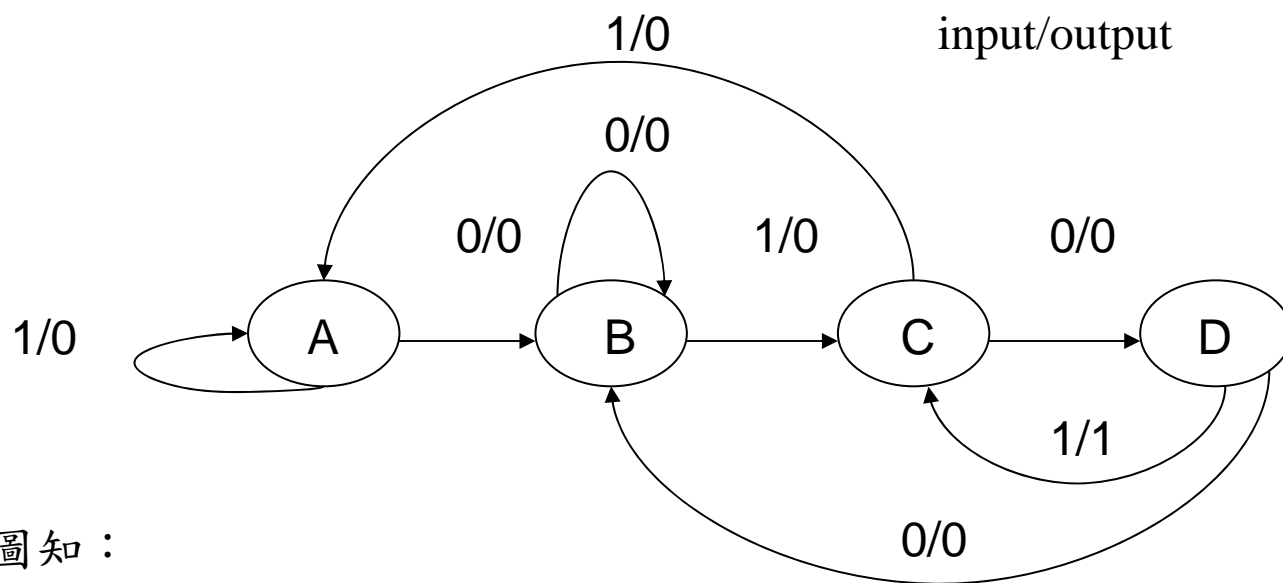
利用時序圖與時序序列兩種表示法，對序向電路的可能輸入序列相當多或是必須完全描述該電路的行為時，並不是很有用。因為這兩種表示方式，只在欲強調電路工作時的時序問題或是欲得之電路如何工作時，才有大的幫助。

因此，在實用上，序向電路的行為通常使用狀態圖或狀態表方式表示。



<例題一>：設一個序向電路有一個輸入變數與一個輸出變數，當此電路每次偵測到輸入序列為0101時，輸出一個為1的信號，否則輸出一個為0的信號。試求出此網路狀態圖。

sol:



如圖知：

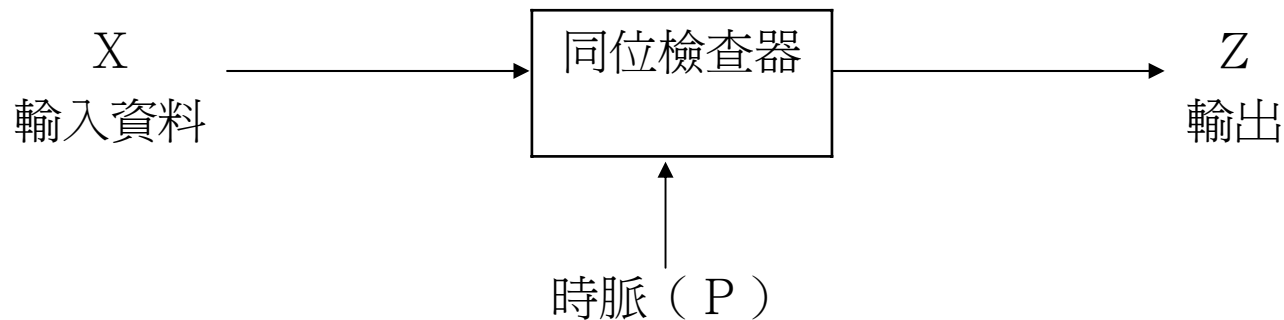
狀態 A 為初始狀態

狀態 B 為認知 “0”

狀態 C 為認知 “01”

狀態 D 為認知 “010”

<例題二>：有一序列式資料 (serial data) 的同位檢查器。網路方塊圖如下所示。當一串 0 或 1 加到其 X 輸入之後，如果接收到 1 的總數呈奇數，則輸出為1，否則為0。試求出此網路狀態圖。

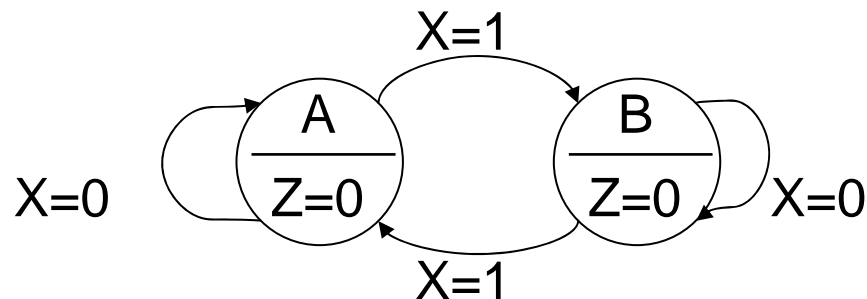


同位檢查器方塊圖

Sol:

此順序網路必須記住接收到的 1 數目是奇數或偶數；因此，我們只需要使用兩種狀態。分別記作 A 和 B，並分別與接收到偶數個 1 與奇數個 1 相對應。

(求出目而機狀態圖的方法和密利機相似，不同點在於輸出值要與狀態寫在一起，而不是寫在兩狀態之間的箭頭線旁。)



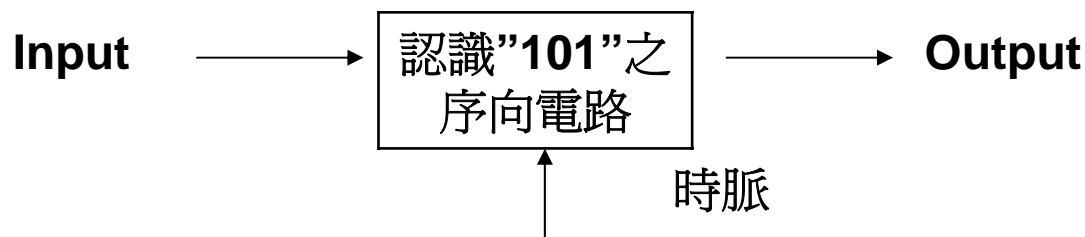
我們定義兩種序向電路：

目而機(Moore machine): 序向電路中的輸出只與目前狀態有關，亦即只為目前狀態的函數稱之。(如例題2)

密利機(Mealy machine): 序向電路中的輸出是目前狀態和輸出的函數稱之。(如例題1)

在理論上，Mealy 機與 Moore 機是等效的，(而且可以互相轉換)；在實用上，選擇 Mealy 機或 Moore 機則依實際問題而定。(如例1、例2)為方便討論，以下將以 Mealy 機為主。

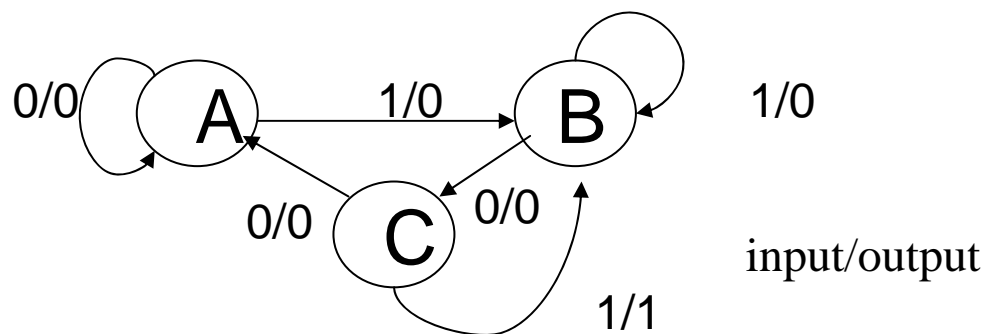
<pb.>：有一序向邏輯電路(sequential logic circuit)其方塊圖如下所示。它可由輸入線接受一連串的數位信號“0”或“1”(serial input)，如果它接受的信號依次是1 0 1時，則在接受1的同時，會由輸出線上產生一“1”的信號，其餘時間，輸出線上都是“0”的信號。試求出其 Mealy 機與 Moore 機狀態圖。



序列檢查器方塊圖

sol:

## Mealy machine

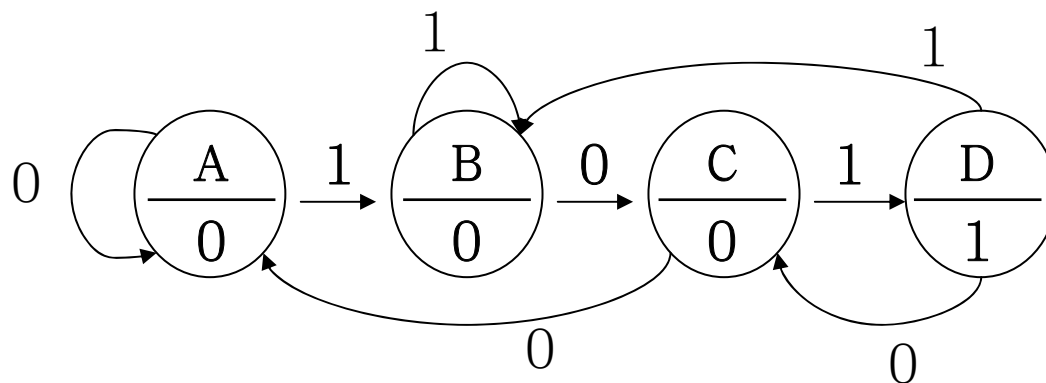


A：初始狀態

B：認知“1”

C：認知“10”

## Moore machine



A：初始狀態

B：認知“1”

C：認知“10”

D：認知“101”

因輸出值必須為1；

因此不能回到A

## 6-4 同步序向電路的設計與分析

序向電路分析：由線路圖著手而以狀態表或狀態圖的完成為止。

序向電路設計：則由一套規格說明著手，而以變成邏輯圖為止。

基本上兩個是相反的程序

## 1. 同步序向電路設計

步驟1、由問題定義導出狀態圖(或狀態表)。

步驟2、消去狀態表中多餘的狀態(狀態化簡)。

步驟3、做狀態指定(state assignment)。

步驟4、導出轉態表(transition table)與輸出表(output table)。

步驟5、選擇欲使用的記憶元件(即正反器)導出激勵表，並由此獲得激勵與輸出方程式。

步驟6、執行激勵與輸出方程式，並繪出邏輯電路。

現舉例說明這些設計步驟



<例題1>：某順序網路具有一個輸入(x)和一個輸出(z)，此網路的功能是檢查由連續4個輸入值組成的一組資料，如果輸入值的順序是0101或者1001，則會產生輸出值  $Z = 1$ 。每個輸入值通過之後，網路便必須重置成初始狀態。

Sol：步驟1、導狀態圖(或狀態表)

對狀態圖或狀態表的導出，書上(Roth)雖有列出原則以資參考，然因沒有一定的步驟，因此設計順序網路時，將問題的敘述轉換成狀態圖(或狀態表)是最困難也最具挑戰性的。

## 狀態表：

	輸入		次態		輸出	
	序列	現態	x=0	X=1	X=0	X=1
認知	起始 狀態	A	B	C	0	0
	0	B	D	E	0	0
	1	C	F	G	0	0
	00	D	H	I	0	0
	01	E	J	K	0	0
	10	F	L	M	0	0
	11	G	N	P	0	0

	輸入		次態		輸出	
	序列	現態	x=0	X=1	X=0	X=1
認知	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

## 步驟2、狀態化簡

<目的>：用以簡化閘數與正反器數，以降低成本，此乃任何 設計步驟所必須考慮問題。

<何謂狀態簡化>：即循序線路裡，正反器數目的縮減問題。其所要進行的工作是在外界輸入及輸出的要求皆無變動的情況下，減少狀態表中狀態數目的程序。

(狀態數目↓，正反器數目↓or↓，不過正反器數目↓，組合邏輯閘數有時也會有意外↑結果。)

在這例子裡，只有輸入輸出順序才是重要的，內在狀態不過用來產生所需順序而已。

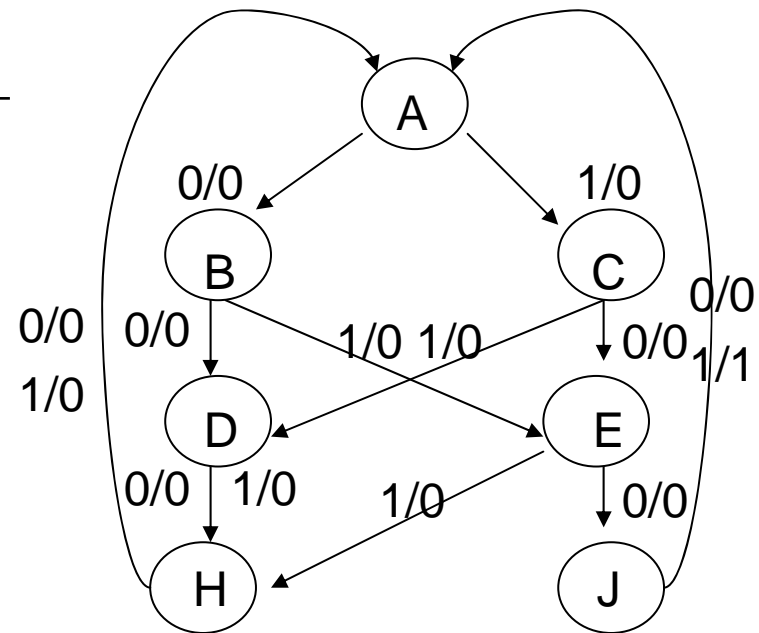


狀態化簡常用方法文獻上雖多有介紹，以下我們介紹列配合(row matching)法及利用隱意表(implication table)或稱合併表(merger table)來完成狀態化簡。

1. 列配合法如下：「對於每一輸入而言，兩個狀態下的輸出皆相同，並且使線路到達相同或相等的次態，則稱此二狀態為相等。而且兩個狀態為相等時，除去其中任一狀態均不改變原有的輸入輸出關係。」

現態	次態		輸出	
	X=0	X=1	X=0	X=1
A	B	C	0	0
B	D	E	0	0
C	F	G	0	0
D	H	I	0	0
E	J	K	0	0
F	L	M	0	0
G	N	P	0	0
H	A	A	0	0
I	A	A	0	0
J	A	A	0	1
K	A	A	0	0
L	A	A	0	1
M	A	A	0	0
N	A	A	0	0
P	A	A	0	0

現態	次態		輸出	
	X=0	X=1	X=0	X=1
A	B	C	0	0
B	D	E	0	0
C	E	D	0	0
D	H	H	0	0
E	J	H	0	0
H	A	A	0	0
J	A	A	0	1



狀態化簡後序列檢查器的狀態表和狀態圖

先直接求狀態圖

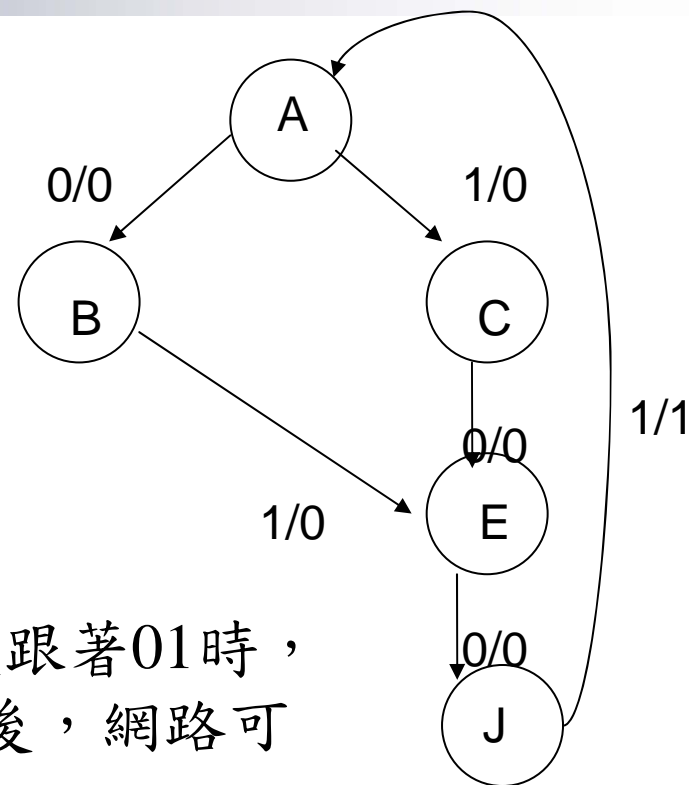
A：重置

B：認知“0”

C：認知“1”

E：認知“01”或“10”

J：認知“010”或“100”



注意：輸入值的序列是01或者10之後跟著01時，輸出值將是1；所以接收到01或10之後，網路可以查到相同的一個狀態。

在圖中加入 D 和 H 狀態的原因是網路在接收到 4 個輸入值之前不可以重置成 A 狀態。又接收到了 00 或 11 的輸入序列以後(D狀態)便不可能會產生1的輸出值。



## 2. 利用隱意表(或稱合併表)決定全等狀態

<b>B</b>	<div>B D C E</div>															
<b>C</b>	<div>B F C G</div>	<div>D F E G</div>														
<b>D</b>	<div>B H C I</div>	<div>B H E I</div>	<div>F H G I</div>													
<b>E</b>	<div>B J C K</div>	<div>D J E K</div>	<div>F J G K</div>	<div>H J I K</div>												
<b>F</b>	<div>B L C M</div>	<div>D L E M</div>	<div>F L G M</div>	<div>H L I M</div>	<div>J L K M</div>											
<b>G</b>	<div>B N C P</div>	<div>D N E P</div>	<div>F N G P</div>	<div>H N I P</div>	<div>J N K P</div>	<div>L N M P</div>										
<b>H</b>	<div>B A C A</div>	<div>D A E A</div>	<div>F A G A</div>	<div>H A I A</div>	<div>J A K A</div>	<div>L A M A</div>	<div>N A P A</div>									
<b>I</b>	<div>B A C A</div>	<div>D A E A</div>	<div>F A G A</div>	<div>H A I A</div>	<div>J A K A</div>	<div>L A M A</div>	<div>N A P A</div>	✓								
<b>J</b>	X	X	X	X	X	X	X	X	X							
<b>K</b>	<div>B A C A</div>	<div>D A E A</div>	<div>F A G A</div>	<div>H A I A</div>	<div>J A K A</div>	<div>L A M A</div>	<div>N A P A</div>	✓	✓	X						
<b>L</b>	X	X	X	X	X	X	X	X	X	✓	X					
<b>M</b>	<div>B A C A</div>	<div>D A E A</div>	<div>F A G A</div>	<div>H A I A</div>	<div>J A K A</div>	<div>L A M A</div>	<div>N A P A</div>	✓	✓	X	✓	X				
<b>N</b>	<div>B A C A</div>	<div>D A E A</div>	<div>F A G A</div>	<div>H A I A</div>	<div>J A K A</div>	<div>L A M A</div>	<div>N A P A</div>	✓	✓	X	✓	X	✓			
<b>P</b>	<div>B A C A</div>	<div>D A E A</div>	<div>F A G A</div>	<div>H A I A</div>	<div>J A K A</div>	<div>L A M A</div>	<div>N A P A</div>	✓	✓	X	✓	X	✓	✓		
	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>	<b>N</b>		

在表中，列與行的抬頭分別為狀態表中的狀態，而列與行的交點，則相當於相當（全等）狀態對。若兩個狀態不相等（容）時，則在其對應的格子中記上（×）符號；對於相等的狀態，則記上“√”符號；被隱含狀態對則置於對應的格子中。

由表中知： $D \equiv G$   
 $E \equiv F$   
 $H \equiv I, K, M, N, P$   
 $J \equiv L$

### 3. 比較：

一般來說，除了在網路接收到固定數目的輸入值便會重置成初始狀態這種特殊的情況之外，列配合法不一定能夠找出所有的全等狀態。此時則需利用隱意表來化簡。然而在作出隱意表之前，我們卻可以先利用列配合的方法先縮減狀態表中的一些狀態。

# <例題>：試化簡下列狀態表

現態	次態	輸出
	x = 0, 1	y
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

b	af ch						
c	×	×					
d	da ce	fa he	×				
e	×	×	ec da	×			
f	×	×	ef db	×	cf ab		
g	ab ch	fb	×	ab eh	×	×	
h	×	×	ec dg	×	ag bg	fc bg	×
	a	b	c	d	e	f	g

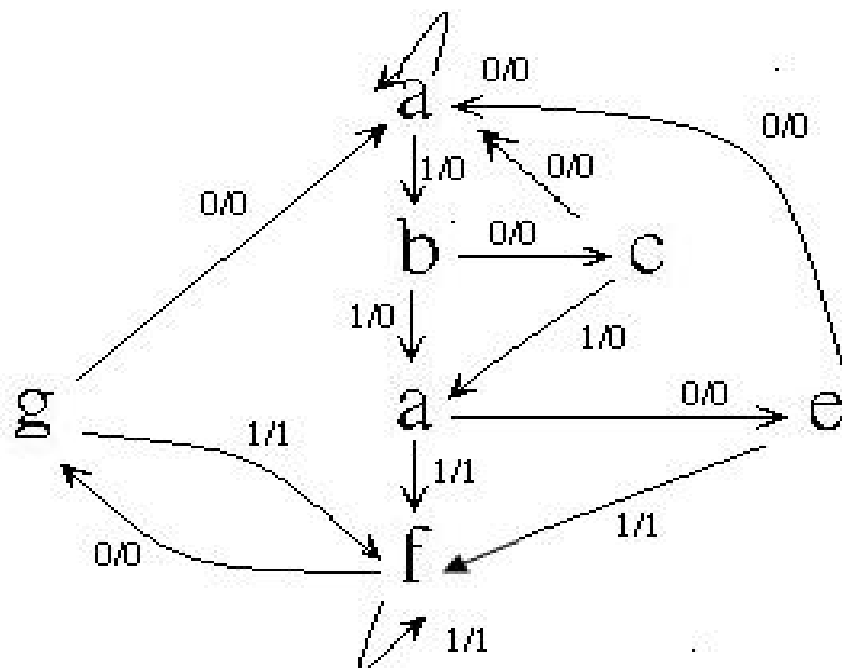
列配法：沒有相等狀態

由implication table知  
 $a \equiv d$  ,  $c \equiv e$

化簡後狀態表：

現態	次態	輸出
	$x = 0, 1$	$y$
a	a c	0
b	f h	0
c	c a	1
f	f b	1
g	b h	0
h	c g	1

<pb.> 試求下列狀態圖中所有的全等狀態。



( Ans :  $d \equiv f$  ,  $e \equiv g$  )


步驟3.狀態指定： $2^{n-1} < m \leq 2^n$

※適當選擇狀態指定，將可降低電路複雜性。

1.嘗試與改正方法（使用在狀態數少的情況）

2.（Roth）原則法（guideline method）：不一定能夠適用於所有的問題，即便適用，也不保證能夠得到最簡的解。

3.SHR技巧：能夠產生應用JK正反器的最佳解（1972.IEEE 文獻）


$$2^{n-1} < 7 \leq 2^n \implies n=3$$

A : 000      B : 001      C : 010

D : 011      E : 100      H : 101

J : 110



步驟4. 導出轉態表（變化表：將正反器的次狀態表示成目前狀態及輸出值的變數）與輸出表

現態			輸入	次態			輸出
$Q_A$	$Q_B$	$Q_C$	$x$	$Q_A$	$Q_B$	$Q_C$	$y$
0	0	0	0	0	0	1	0
0	0	0	1	0	1	0	0
0	0	1	0	0	1	1	0
0	0	1	1	1	0	0	0
0	1	0	0	1	0	0	0
0	1	0	1	0	1	1	0
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	0
1	0	0	0	1	1	0	0
1	0	0	1	1	0	1	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0
1	1	0	0	0	0	0	0
1	1	0	1	0	0	0	1

步驟5. 選擇記憶元件（FFs）導出激勵表，並求得激勵（正反器輸入）與輸出方程式。選擇 D type FFs，則激勵表與轉態表相同。

$D_A$ $Q_A Q_B$		$Q_C X$			
		00	01	11	10
00				1	
01	1			1	1
11				x	x
10	1	1			

$D_B$ $Q_A Q_B$		$Q_C X$			
		00	01	11	10
00			1		1
01			1		
11				x	x
10	1				


$$D_A = Q_B Q_C + Q'_A Q_C X + Q'_A Q_B X' + Q_A Q'_B Q'_C$$

$$D_B = Q'_A Q'_C X + Q'_A Q'_B Q_C X' + Q_A Q'_B Q'_C X'$$

$$Y = Q_A Q_B Q'_C X$$

$D_C$ \ $Q_C X$					
		00	01	11	10
$Q_A Q_B$	00	1			1
	01		1	1	1
	11			x	x
	10		1		

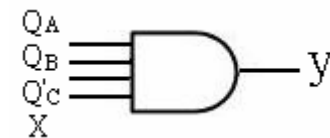
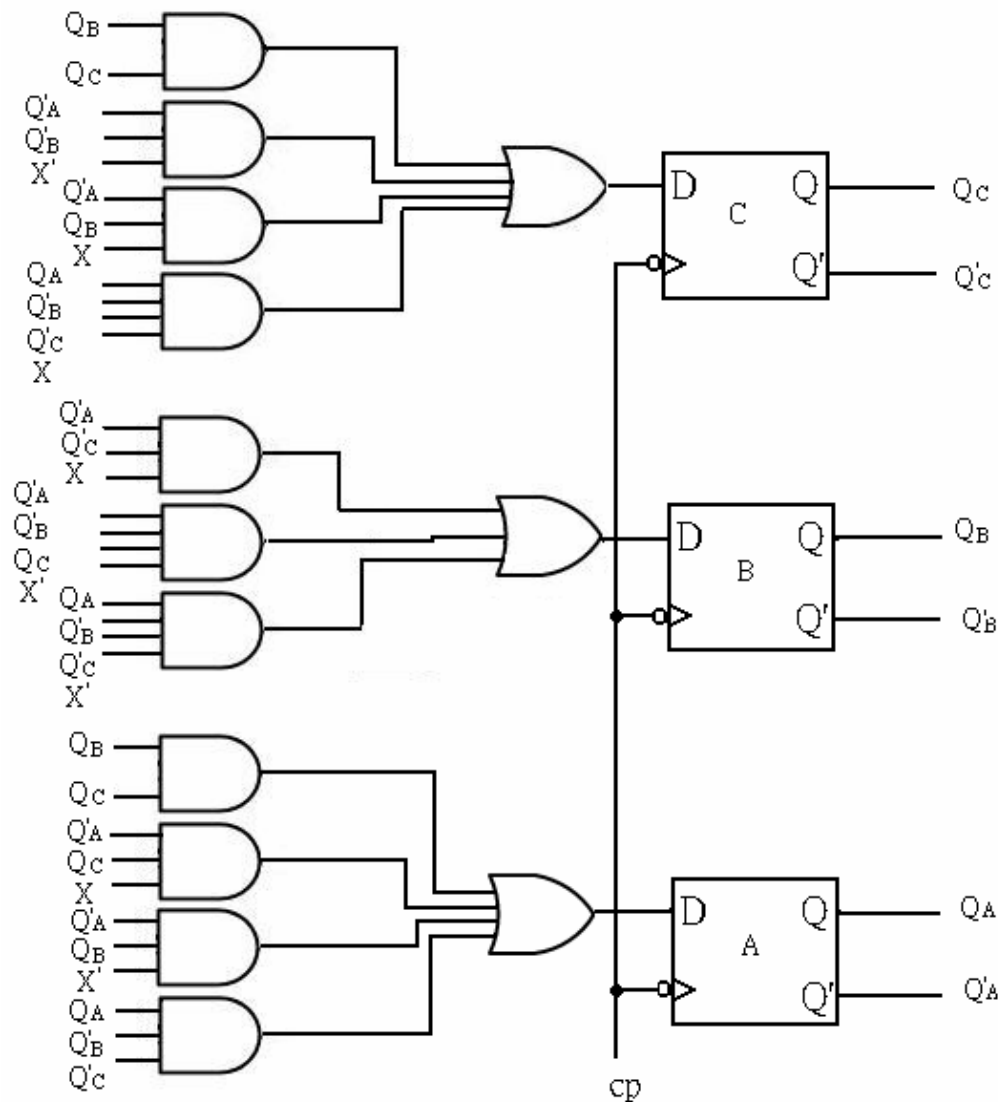
$$D_C = Q_B Q_C + Q'_A Q'_B X' + Q'_A Q_B X + Q_A Q'_B Q'_C X$$



◎以不用狀態設計，在嚴謹的設計中應分析線路的變數狀態，以保證設計步驟沒有錯誤。

- <1>提供一個當一重置輸入，在主時脈衝工作之前，不同步地預先設定系統中所有正反器的起始狀態。
- <2>當一個序向電路能夠由任何無效狀態開始，而最後（在一個或兩個主時脈衝之後）終究會回到正常的動作迴圈上時稱為自我啟動（self-starting）與自動修正（self-correcting）。
- <3>若線路永遠停留在無用狀態之間，則線路應該重新設計。重新設計時只要找出無效狀態之間造成循環的不用狀態，將它們指定一個有用狀態即可。

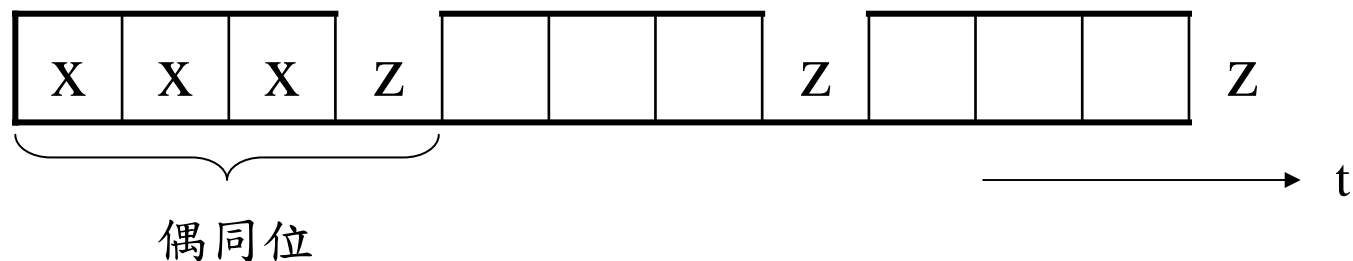
## 步驟6. 畫邏輯電路圖。



☆同步序向電路組合邏輯部份除了使用SSI執行外，亦可使用LSI（ex: PAL內含 Dtype FF）與MSI（ex: MUX）。

## <例題2> (同位產生器-使用JK正反器)

設計一個偶同位產生電路，它具有一個輸入端  $x$  與一個輸出端  $z$ ，當由  $x$  端連續接收三個位元的資料後，若這些位元的 1 總數為奇數，則電路產生 1 的輸出於  $z$  輸出端，否則產生 0 的輸出。



Sol:<步驟1>：導狀態圖  
(狀態表)

輸入 序列	現態	次態 X= 0, 1	輸出 X=0,1
	A	B C	0 0
0	B	D E	0 0
1	C	F G	0 0
00	D	H I	0 0
01	E	<del>I</del> / <del>J</del> K H	0 0
10	F	<del>I</del> / <del>L</del> M H	0 0
11	G	H <del>N</del> / <del>P</del> I	0 0
000	H	A A	0 0
001	I	A A	1 1
010	J	A A	1 1
011	K	A A	0 0
100	L	A A	1 1
101	M	A A	0 0
110	N	A A	0 0
111	P	A A	1 1

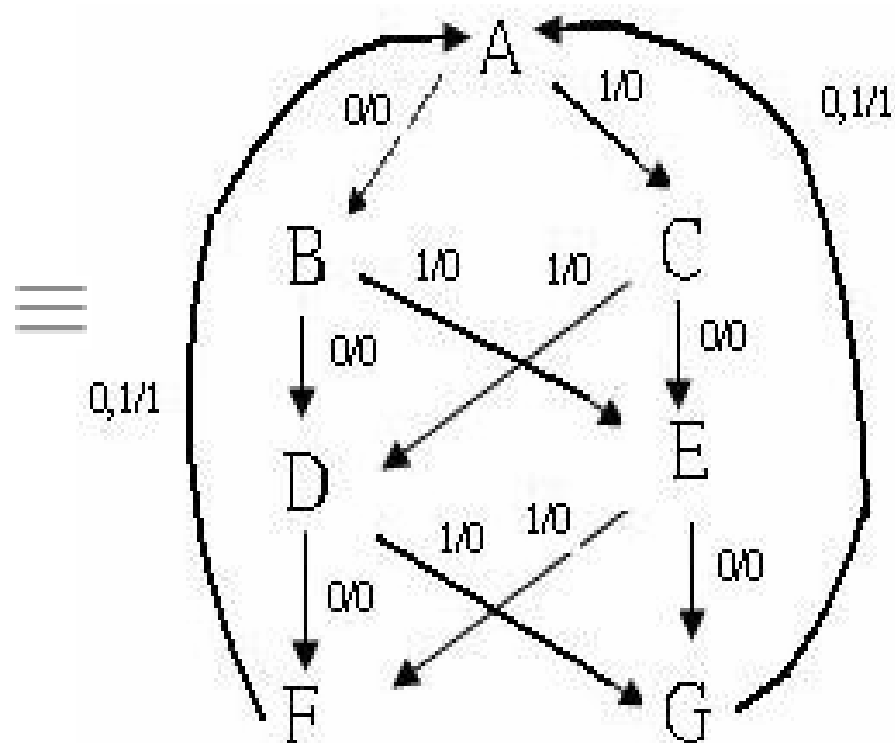
## <步驟2>：狀態化簡

現態	次態 X= 0 ,1	輸出 X=0,1
A	B C	0 0
B	D E	0 0
C	E <del>F</del> <del>G</del> D	0 0
D	H I	0 0
E	I H	0 0
<del>F</del>	<del>I H</del>	<del>0 0</del>
<del>G</del>	<del>H I</del>	<del>0 0</del>
H	A A	0 0
I	A A	1 1



## 重新整理

PS 現態	次態(NS) X= 0 ,1	輸出(Z) X=0,1
A	B C	0 0
B	D E	0 0
C	E D	0 0
D	F G	0 0
E	G F	0 0
F	A A	0 0
G	A A	1 1



### <步驟3>：狀態指定

A : 000 , B : 010 , C : 011

D : 110 , E : 111 , F : 100

G : 101

## < 步驟4 > : 導出轉態表與輸出表

PS      input				NS			Z	激勵表
Y <sub>1</sub>	Y <sub>2</sub>	Y <sub>3</sub>	X	Y <sub>1</sub>	Y <sub>2</sub>	Y <sub>3</sub>	output	J <sub>1</sub> K <sub>1</sub> J <sub>2</sub> K <sub>2</sub> J <sub>3</sub> K <sub>3</sub>
0	0	0	0	0	1	0	0	0 $\phi$ 1 $\phi$ 0 $\phi$
0	0	0	1	0	1	1	0	0 $\phi$ 1 $\phi$ 1 $\phi$
0	1	0	0	1	1	0	0	1 $\phi$ $\phi$ 0 0 $\phi$
0	1	0	1	1	1	1	0	1 $\phi$ $\phi$ 0 1 $\phi$
0	1	1	0	1	1	1	0	1 $\phi$ $\phi$ 0 $\phi$ 0
0	1	1	1	1	1	0	0	1 $\phi$ $\phi$ 0 $\phi$ 1
1	1	0	0	1	0	0	0	$\phi$ 0 $\phi$ 1 0 $\phi$
1	1	0	1	1	0	1	0	$\phi$ 0 $\phi$ 1 1 $\phi$
1	1	1	0	1	0	1	0	$\phi$ 0 $\phi$ 1 $\phi$ 0
1	1	1	1	1	0	0	0	$\phi$ 0 $\phi$ 1 $\phi$ 1
1	0	0	0	0	0	0	0	$\phi$ 1 0 $\phi$ 0 $\phi$
1	0	0	1	0	0	0	0	$\phi$ 1 0 $\phi$ 0 $\phi$
1	0	1	0	0	0	0	1	$\phi$ 1 0 $\phi$ $\phi$ 1
1	0	1	1	0	0	0	1	$\phi$ 1 0 $\phi$ $\phi$ 1

<步驟5>：選擇FFs，導出激勵表，並求激勵方程式（FF輸入方程式）及輸出方程式。

$Q_t$	$Q_{(t+1)}$	J	K
0	0	0	$\phi$
0	1	1	$\phi$
1	0	$\phi$	1
1	1	$\phi$	0

$J_1 \setminus y_3 X$		00	01	11	10
$y_1 y_2$	00	0	0	0	0
	01	1	1	1	1
	11	$\phi$	$\phi$	$\phi$	$\phi$
	10	$\phi$	$\phi$	$\phi$	$\phi$

$$J_1 = y_2$$

$K_1 \setminus y_3 X$		00	01	11	10
$y_1 y_2$	00	<del><math>\phi</math></del>	<del><math>\phi</math></del>	<del><math>\phi</math></del>	<del><math>\phi</math></del>
	01	$\phi$	$\phi$	$\phi$	$\phi$
	11	0	0	0	0
	10	<del>1</del>	<del>1</del>	<del>1</del>	<del>1</del>

$$K_1 = y'_2$$

$J_2 \backslash y_3 X$		00	01	11	10
$y_1 y_2$					
00		1	1	$\phi$	$\phi$
01		$\phi$	$\phi$	$\phi$	$\phi$
11		$\phi$	$\phi$	$\phi$	$\phi$
10		0	0	0	0

$$J_2 = y'_1$$

$K_2 \backslash y_3 X$		00	01	11	10
$y_1 y_2$					
00		$\phi$	$\phi$	$\phi$	$\phi$
01		0	0	0	0
11		1	1	1	1
10		$\phi$	$\phi$	$\phi$	$\phi$

$$K_2 = y_1$$

$J_3$ $y_3 X$					
$y_1 y_2$		$y_3 X$			
		00	01	11	10
00		0	1	$\phi$	$\phi$
01		0	1	$\phi$	$\phi$
11		0	1	$\phi$	$\phi$
10		0	0	$\phi$	$\phi$

$$J_3 = y'_1 x + y_2 x$$

$K_3$ $y_3 X$					
$y_1 y_2$		$y_3 X$			
		00	01	11	10
00		$\phi$	$\phi$	$\phi$	$\phi$
01		$\phi$	$\phi$	1	0
11		$\phi$	$\phi$	1	0
10		$\phi$	$\phi$	1	1

$$K_3 = x + y'_2$$

<步驟6>：畫邏輯電路圖

## 2.以狀態方程式(特性方程式)來設計

<例題>同上例，則由轉態表得

$Y_1 \backslash Y_3 X$					
		00	01	11	10
$y_1 y_2$	00	0	0	$\phi$	$\phi$
	01	1	1	1	1
	11	1	1	1	1
	10	0	0	0	0

$$Y_1 = y_2$$

$Y_2 \backslash Y_3 X$					
		00	01	11	10
$y_1 y_2$	00	1	1	$\phi$	$\phi$
	01	1	1	1	1
	11	0	0	0	0
	10	0	0	0	0

$$Y_2 = y'_1$$



$y_1 y_2$	$y_3 X$	00	01	11	10
00		0	1	$\phi$	$\phi$
01		0	1	0	1
11		0	1	0	1
10		0	0	0	0

$$\begin{aligned}
 Y_3 &= y'_1 y'_3 x + y_2 y'_3 x + y_2 y_3 x' \\
 &= (y'_1 x + y_2 x) y'_3 + y_2 x' y_3
 \end{aligned}$$

又JK FF特性方程式為

$$Q(t+1) = J Q'(t) + K' Q(t)$$


$$\text{且 } Y_1 = y_2 = y_2(y'_1 + y_1) = y_2 y'_1 + y_2 y_1$$

$$\text{Let } J_1 = y_2, K'_1 = y_2 \rightarrow K_1 = y'_2$$

$$Y_2 = y'_1 = y'_1(y'_2 + y_2) = y'_1 y'_2 + y'_1 y_2$$

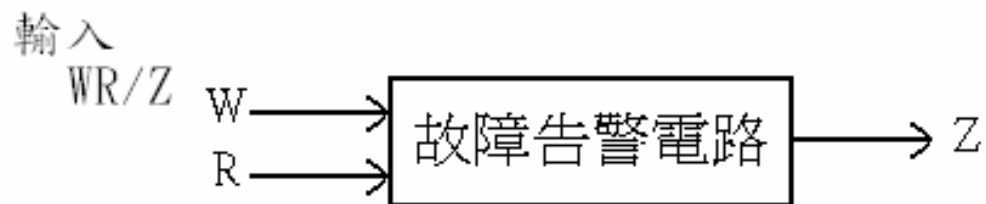
$$\text{Let } J_2 = y'_1, K'_2 = y'_1 \rightarrow K_2 = y_1$$

$$\text{Let } J_3 = y'_1 x + y_2 x, K'_3 = y_2 x' \rightarrow K_3 = y'_2 + x$$

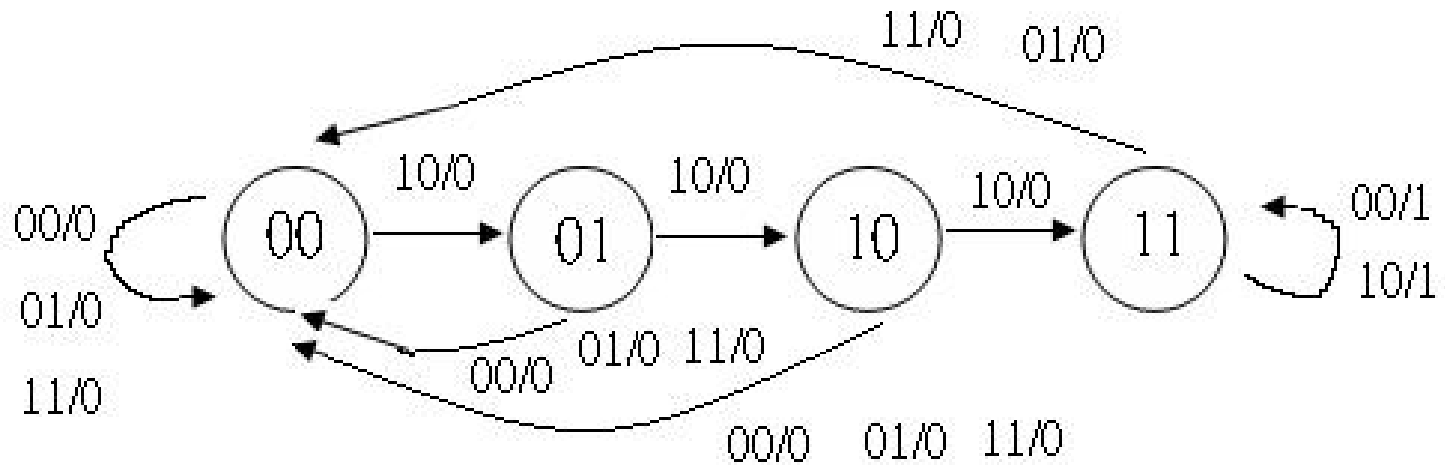


事實上，利用特性方程式求正反器的激勵方程式和利用激勵表是一樣的，只是前者以代數運算方式而後者以表格方式。一般而言，利用特性方程式求正反器的激勵方程式較使用激勵表的方式沒有系統而且困難。

<pb> 某系統中之故障告警電路，有二個輸入；偵錯信號W及重置信號R，有一輸出告警信號Z，如圖示。若偵測到錯誤時  $W = 1$ ；若連續錯三次以上就一直送告警信號（ $Z=1$ ），通知維護人員；在任何狀態按重置（ $R=1$ ）將使回覆到無錯狀態。現假設無錯狀態為00，錯一次狀態為01，連錯二次之狀態為10，連續三次以上之狀態為11，試利用D type FFs 設計此故障告警電路。



Sol :



$$D_A = BWR' + AWR' + ABR'$$

$$D_B = B'WR' + ABR'$$

$$Z = ABR'$$

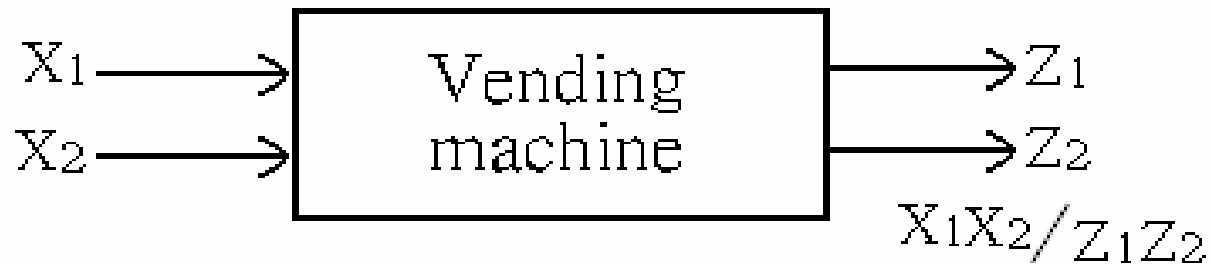
備註：由於目而網路只有在正反器狀態改變時才可能改變狀態，因此目而網路中不可能出現錯誤的輸出。

密利網路則由於網路已經變成了新狀態，但是屬於前一狀態的舊輸入值仍未消失。因此輸出可能有“錯誤的”輸出值。

如果此網路的輸出是饋入使用同一時脈的其他順序網路中，則“錯誤的”輸出將不會造成任何問題，因為其它的網路只會在時脈出現時改變狀態。

“錯誤的”輸出有兩種，一種是期望輸出值保持1時，輸出值暫時的變成0；另一種則反之。

- © Suppose the vending machine accepts only 5-and-10 dollar coins one by one



$X_1$  : 10元 ,  $X_2$  : 5元 ,  $Z_1$  : coke ,  $Z_2$  : change

