
Chapter
Chapter
Chapter
Chapter
Chapter
Chapter
Chapter
Chapter
8

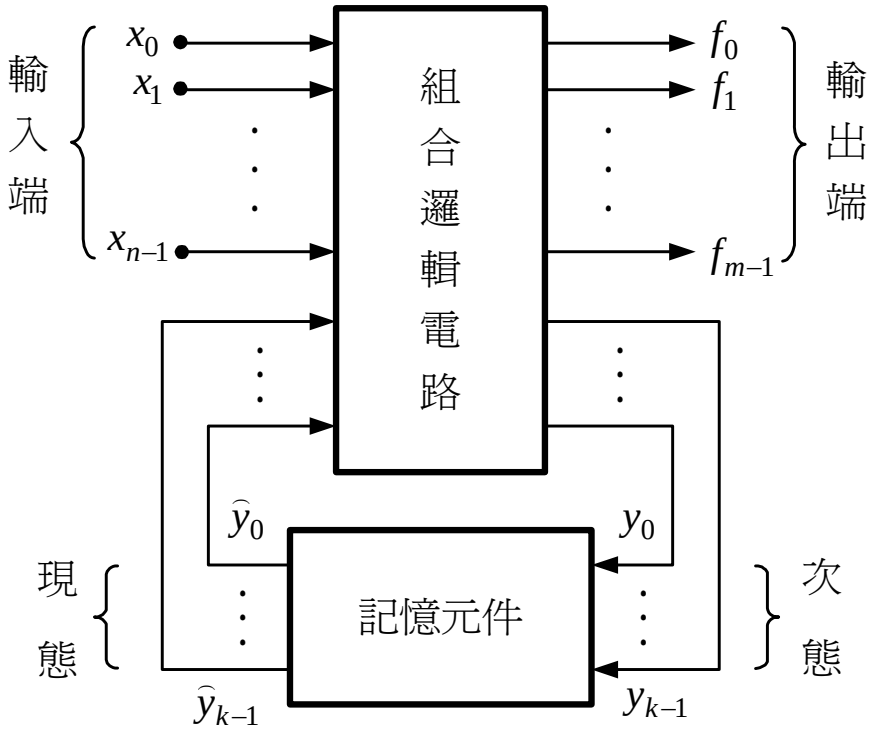
序向邏輯電路之設計

概　述

- 序向邏輯電路 (Sequential Logic Circuits) 的輸出狀態，不僅與目前輸入狀態有關，亦受前一個輸出狀態的影響，這意謂序向邏輯電路有使用記憶元件 (Memory Cell) 來保存前一個輸出之狀態，因此序向邏輯電路是組合邏輯電路與記憶元件所構成之電路。
- 時序控制之序向邏輯電路而言，當時序脈波發生變化時，才會使電路之輸出狀態產生轉變，而依時脈訊號之不同連接方式，序向邏輯電路可分為同步 (Synchronous) 與非同步 (Asynchronous) 序向邏輯電路兩種。
- 同步序向邏輯電路僅在時脈訊號 (*CP*) 觸發正反器之瞬間，方可改變電路之輸出狀態；而非同步序向邏輯電路可在任一時刻改變電路之輸出狀態。

序向邏輯電路基本觀念

◆ 序向邏輯電路是由 n 個輸入端 $(x_0, x_1, \ldots, x_{n-1})$、$m$ 個輸出端 $(f_0, f_1, \ldots, f_{m-1})$ 與 k 個記憶元件(k 個輸出端：$\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_{k-1}$與 k 個輸入端：$y_0, y_1, \ldots, y_{k-1}$)所組成，序向邏輯電路之結構方塊圖，如下圖所示。



- 當輸入端 $(x_0, x_1, ..., x_{n-1})$ 與記憶元件之輸出 $(\hat{y}_0, \hat{y}_1, ..., \hat{y}_{k-1})$ 等輸入訊號經過組合邏輯電路運算後，以產生輸出 $(f_0, f_1, ..., f_{m-1})$ 與記憶元件之輸入 $(y_0, y_1, ..., y_{k-1})$ 等訊號。而記憶元件之輸出訊號 $(\hat{y}_0, \hat{y}_1, ..., \hat{y}_{k-1})$ 稱為序向邏輯電路之現態 (Present State; *PS*)；記憶元件之輸入訊號 $(y_0, y_1, ..., y_{k-1})$ 稱為序向邏輯電路之次態 (Next State; *NS*)。

- 序向邏輯電路亦僅允許使用二元性訊號 (Binary Signal)，因此對具 k 個記憶元件 (k 個正反器) 之序向邏輯電路而言，因每一個記憶元件可用來表示一個變數，故有 k 個變數 (Variables) 可供使用，每個變數皆可以補數 (Complement) 或非補數 (Noncomplement) 之型態出現，故最多可產生 2^k 種二進位狀態。

- 序向邏輯電路有 k 個記憶元件，則最多具 2^k 種可能之二進位狀態組合，因此若需要 j 個狀態之序向邏輯電路，則電路所需之記憶元件數量 k 至少需要 $\log_2 j$ 個 (即 $k = \lceil \log_2 j \rceil$)，換句話說，具有 k 個記憶元件之序向邏輯電路，最多可用來處理 2^k 個不同之二進位狀態。

正反器之激勵表(概述)

- 在設計序向邏輯電路時，通常皆由已知所求電路之現態 (*PS*) 與次態 (*NS*) 之轉態情況開始，再將現態與外界輸入，透過組合邏輯電路運算後之輸出，作為正反器之輸入訊號，以滿足已知之轉態要求，即已知正反器輸出狀態之變遷情況，再求解應給予正反器何種輸入邏輯狀態，以符合所求之狀態轉變需求。

- 因此序向邏輯電路之設計，首先需由已知正反器之輸出狀態 (次態) 開始，求解可符合此輸出狀態 (次態)下，所有正反器輸入狀態 (現態) 之邏輯運算式為止。

- 正反器之真值表 (True Table) 可用來描述目前輸入狀態 (現態；*PS*) 下，正反器下一個輸出狀態 (次態；*NS*)；而正反器之激勵表(Excitation Table)是由已知之輸出狀態，以推導出之正反器輸入狀態。

- 本節將討論 *SR* 正反器、*D* 型正反器、*JK* 正反器與 *T* 型正反器等 4 種正反器之真值表，以推導出對應激勵表之方法。

SR 正反器之激勵表

◆ 首先列出 *SR* 正反器之真值表，如下表所示

現態輸出 Q_n	輸　入		次態輸出 Q_{n+1}
	S_n	R_n	
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	×（無效）
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	×（無效）

◆ 接著利用 *SR* 正反器之真值表作一個反向思考，便可獲得 *SR* 正反器之激勵表，如下表所示。

現態輸出 Q_n	次態輸出 Q_{n+1}	正反器之輸入 S_n	正反器之輸入 R_n	說明
0	0	0 0	0 1	$S_n = R_n = 0$時，$Q_{n+1} = Q_n = 0$； $S_n = 0$、$R_n = 1$時，$Q_{n+1} = 0$
0	1	1	0	$S_n = 1$、$R_n = 0$時，$Q_{n+1} = 1$
1	0	0	1	$S_n = 0$、$R_n = 1$時，$Q_{n+1} = 0$
1	1	0 1	0 0	$S_n = R_n = 0$時，$Q_{n+1} = Q_n = 1$； $S_n = 1$、$R_n = 0$時，$Q_{n+1} = 1$

現態輸出 Q_n	次態輸出 Q_{n+1}	正反器之輸入 S_n	正反器之輸入 R_n
0	0	0	×
0	1	1	0
1	0	0	1
1	1	×	0

D 型正反器之激勵表

◆ 首先列出 D 型正反器之真值表，如下表所示。

現態輸出(Q_n)	輸　入(D_n)	次態輸出(Q_{n+1})
0	0	0
0	1	1
1	0	0
1	1	1

◆ 接著利用 D 型正反器之真值表作一個反向思考，便可獲得 D 型正反器之激勵表，如下表所示。

現態輸出(Q_n)	次態輸出(Q_{n+1})	正反器之輸入(D_n)	說　　明
0	0	0	$D_n = 0$時，$Q_{n+1} = 0$
0	1	1	$D_n = 1$時，$Q_{n+1} = 1$
1	0	0	$D_n = 0$時，$Q_{n+1} = 0$
1	1	1	$D_n = 1$時，$Q_{n+1} = 1$

JK 正反器之激勵表

◆ 首先列出 *JK* 正反器之真值表，如下表所示。

現態輸出 Q_n	輸入		次態輸出 Q_{n+1}
	J_n	K_n	
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

◆ 接著利用 JK 正反器之真值表作一個反向思考，便可獲得 JK 正反器之激勵表，如下表所示

現態輸出 Q_n	次態輸出 Q_{n+1}	正反器之輸入 J_n	正反器之輸入 K_n	說明
0	0	0	0	$J_n = K_n = 0$時，$Q_{n+1} = Q_n = 0$；
		0	1	$J_n = 0$、$K_n = 1$時，$Q_{n+1} = 0$
0	1	1	0	$J_n = 1$、$K_n = 0$時，$Q_{n+1} = 1$；
		1	1	$J_n = K_n = 1$時，$Q_{n+1} = \overline{Q}_n = 1$
1	0	0	1	$J_n = 0$、$K_n = 1$時，$Q_{n+1} = 0$；
		1	1	$J_n = K_n = 1$時，$Q_{n+1} = \overline{Q}_n = 0$
1	1	0	0	$J_n = K_n = 0$時，$Q_{n+1} = Q_n = 1$；
		1	0	$J_n = 1$、$K_n = 0$時，$Q_{n+1} = 1$

現態輸出 Q_n	次態輸出 Q_{n+1}	正反器之輸入 J_n	正反器之輸入 K_n
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

T 型正反器之激勵表

◆ 首先列出 T 型正反器之真值表，如下表所示。

現態輸出(Q_n)	輸入(T_n)	次態輸出(Q_{n+1})
0	0	0
0	1	1
1	0	1
1	1	0

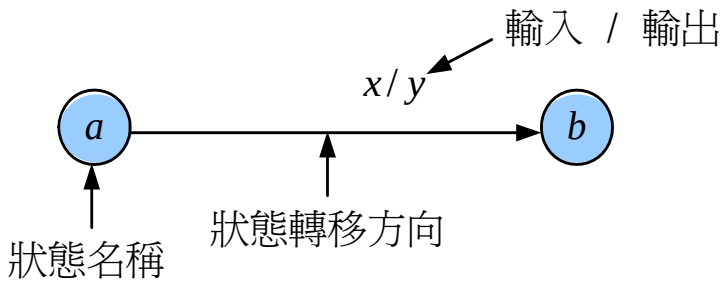
◆ 接著利用 T 型正反器之真值表作一個反向思考，便可獲得 T 型正反器之激勵表，如下表所示。

現態輸出(Q_n)	次態輸出(Q_{n+1})	正反器之輸入(T_n)	說　　明
0	0	0	$T_n = 0$時，$Q_{n+1} = Q_n = 0$
0	1	1	$T_n = 1$時，$Q_{n+1} = \overline{Q}_n = 1$
1	0	1	$T_n = 1$時，$Q_{n+1} = \overline{Q}_n = 1$
1	1	0	$T_n = 0$時，$Q_{n+1} = Q_n = 0$

序向邏輯電路之表示方法(狀態圖)

- 組合邏輯電路是一種無記憶性電路，即電路之輸出狀態完全由當時之輸入狀態來決定，而與過去之輸出狀態無關，故使用真值表 (True Table) 便足以描述組合邏輯電路之規格與功能。
- 序向邏輯電路是一種具記憶性電路，即電路之輸出狀態不僅與目前之輸入狀態有關，亦必須參考前一個輸出狀態，才可得到正確的輸出狀態，故必需使用狀態圖 (State Diagram) 與狀態表 (State Table) 或狀態方程式 (State Equation)，才可完整描述序向邏輯電路之規格與功能。
- 狀態圖 (State Diagram) 是以圖形 (Graph) 來描述序向邏輯電路之狀態轉換過程、輸入與輸出間之關係。而在狀態圖中，每一個頂點 (Vertex) 代表序向電路中之一個狀態，而連接頂點間之方向性分支 (Directed Edge)，則代表狀態轉換之方向，而連接頂點間之方向性分支上，以「/」之符號所隔開的數值，分別代表電路之現態輸入與現態輸出。

◆ **當輸入為 *x* (*x* 僅可為邏輯 1 或 0 兩個數值之一) 時，則表示狀態 *a* 會轉移至狀態 *b*，且電路之輸出為 *y* (*y* 亦僅可為邏輯 1 或 0 兩個數值之一)，如下圖所示。**



例題 8-1

若某序向邏輯電路具一個輸入 x 與輸出 y 之變數，當電路偵測到輸入訊號之序列為 1011 時，則電路會在接收到最後一個 1 之同時，在輸出端得到 1 之訊號，而在其他之情況，電路之輸出皆為 0，試繪出可實現上述序向邏輯電路之狀態圖。

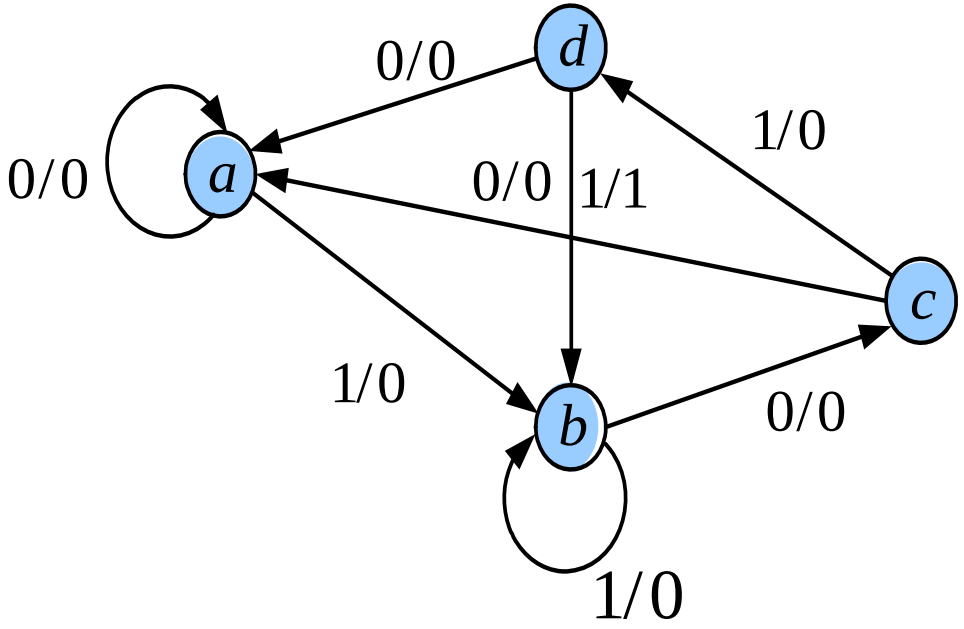
解

假設狀態 a 為初始狀態 (Initial State)，依題意可知：

1. 當狀態 a 之輸入 $x=0$ 時，則停留在 a 狀態，因未偵測到 $x=1011$ 之條件，故輸出為 $y=0$；當狀態 a 之輸入 $x=1$ 時，則轉換至 b 狀態，因尚未偵測到 $x=1011$ 之條件，故輸出亦為 $y=0$。
2. 當狀態 b 之輸入 $x=0$ 時，則轉換至 c 狀態，且此時得到 $x=10$，亦未符合 $x=1011$ 之條件，故輸出 $y=0$；若狀態 b 之輸入 $x=1$ 時，亦未偵測到 $x=1011$ 之條件，則停留在 b 狀態，且輸出亦為 $y=0$。
3. 當狀態 c 之輸入 $x=0$ 時，則轉換至 a 狀態 (回到初始狀態)，因尚未偵測到 $x=1011$ 之條件，故輸出亦為 $y=0$；若狀態 c 之輸入 $x=1$ 時，亦未偵測到 $x=1011$ 之條件，則轉換至 d 狀態，且輸出亦為 $y=0$。

4. 當狀態 d 之輸入 $x=0$ 時，則轉換至 a 狀態，亦未偵測到 $x=1011$ 之條件，故輸出為 $y=0$；若狀態 d 之輸入 $x=1$ 時，則轉換至 b 狀態（回到初始狀態），因已偵測到 $x=1011$ 之條件，且輸出為 $y=1$。

最後使用 a、b、c 與 d 等 4 個狀態，以描述所求序向邏輯電路之狀態圖，如下圖所示。



狀態表與狀態指定

- 將狀態圖轉換成為描述不同之輸入狀態 *x*，導致正反器之現態 (*PS*) 輸出與次態 (*NS*) 輸出之轉換過程與電路之輸出狀態 *y*，以表格之方式呈現出來，稱為狀態表 (State Table)。
- 狀態指定 (State Assignment) 是給予狀態圖中，每個狀態不同之二進位數字，以取代不同的字母符號 (狀態)。而所指定二進位數之位元數 (Bit Number)，必須與所求序向邏輯電路所需之正反器數目相同。

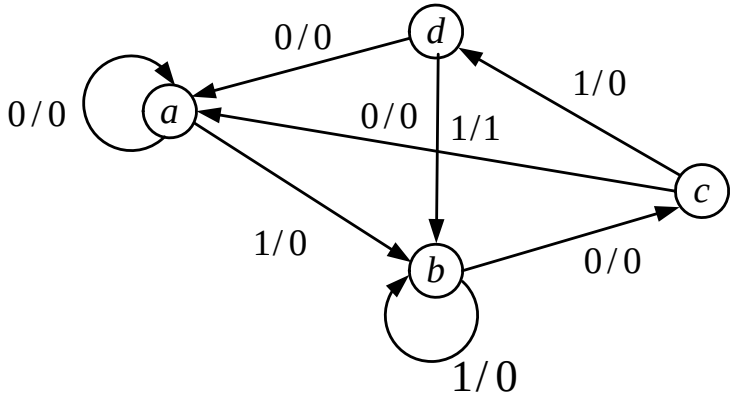
例題 8-2

試將例題 8-1 所得之狀態圖轉換為狀態表，並對所得之狀態表進行狀態指定，以描述所求之序向邏輯電路。

解

參考例題 8-1 之說明與狀態表之定義，即可得到描述所求序向邏輯電路之狀態表，如下表所示。

現　態	次　態		輸　出	
	$x=0$	$x=1$	$x=0$	$x=1$
a	a	b	0	0
b	c	b	0	0
c	a	d	0	0
d	a	b	0	1



◆ 因所求狀態圖計有 4 個狀態，故需要用 2 個正反器 $(n=\lceil \log_2 4 \rceil = 2)$ 來實現，首先將此兩個正反器分別標示為 A 與 B 之符號，以設計所求之序向邏輯電路。

◆ 若對狀態表進行下面之狀態指定後，即指定狀態 a 為「00」、狀態 b 為「01」、狀態 c 為「10」、狀態 d 為「11」，則可得狀態指定後之狀態表，如下表所示。

現　態		次　態				輸　出	
		$x=0$		$x=1$		$x=0$	$x=1$
A_n	B_n	A_{n+1}	B_{n+1}	A_{n+1}	B_{n+1}	y	y
0	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0
1	0	0	0	1	1	0	0
1	1	0	0	0	1	0	1

註 1：理論上，狀態指定以盡量降低所求序向邏輯電路所需之邏輯閘數目為最終目的，即對每一個狀態給予不同二進位數值，可能會造成使用不同複雜度之邏輯閘的結果。

註 2：然要得到使用最少邏輯閘之狀態指定（即最佳之狀態指定法），所造成之計算過程太過於繁瑣，且亦無法保證所得之狀態指定為最佳結果。為減化問題之討論，本書對於狀態指定，僅依二進位數順序來指定給每個字母符號（狀態），而不再詳細討論得到最佳狀態指定之方法。

狀態方程式

◆ **狀態方程式** (State Equation) 可用來描述正反器狀態轉換過程之數學表示式。狀態方程式之等式左邊，可用來表示正反器之次態 (*NS*)，而等式之右邊，則表示可使次態 (*NS*) 轉變為邏輯 1 之現態條件，而電路之輸出方程式，表示輸出 *y* 為邏輯 1 之現態條件。

例題 8-3

試將例題 8-2 所得之狀態表轉換為狀態方程式與輸出方程式，以描述所求之序向邏輯電路。

解

由所求狀態表可得正反器 *A*、*B* 之次態輸出 (狀態方程式等式之左邊表示正反器之次態 (*NS*)，而等式之右邊，則表示可使次態 (*NS*) 轉變為邏輯 1 之現態條件) 與電路之輸出方程式 (輸出 *y* 為 1 時，所對應之現態條件) 如下：

$$A_{n+1} = \bar{x} \cdot \overline{A}_n \cdot B_n + x \cdot A_n \cdot \overline{B}_n$$

$$B_{n+1} = (\overline{A}_n \cdot \overline{B}_n + \overline{A}_n \cdot B_n + A_n \cdot \overline{B}_n + A_n \cdot B_n) \cdot x = x$$

$$y = A_n \cdot B_n \cdot x$$

序向邏輯電路之設計步驟

- 一般而言，序向邏輯電路之設計，以功能描述開始，直到得到邏輯電路圖為止。因序向邏輯電路包含正反器(記憶元件)與組合邏輯電路，故其設計步驟會比組合邏輯電路複雜，且較無一定之規則可遵循。

- 序向邏輯電路又可分為同步與非同步序向邏輯電路兩種，且此兩種電路之設計方式亦稍有不同，因此僅可大致遵循下列幾個步驟：

  1. 對所求問題進行詳細分析，以繪出描述所求問題的序向邏輯電路之狀態圖 (State Diagram)。
  2. 針對所得之狀態圖進行狀態化簡，消除重複出現之狀態，以降低實際所需狀態數目，並依據化簡後之總狀態數目，以決定所需之正反器之數目。
  3. 選擇適當型式之正反器，並標示所有正反器輸入與輸出之符號。
  4. 將化簡所得狀態表之每一個狀態，給予狀態指定 (State Assignment)。

5. 決定實現序向邏輯電路之正反器型式，並使用狀態指定後之狀態表，配合所選用正反器之激勵表，以推導出轉態表 (Transition Table) 或狀態方程式 (State Equation)。

6. 利用卡諾圖來化簡每一個正反器輸入之最簡布林代數，或使用正反器之特性方程式 (Characteristic Equation) 配合狀態方程式，以求解每一個正反器輸入之最簡布林代數。

7. 依據所選用之正反器型式與化簡所得之每一個正反器輸入的最簡布林代數，以繪出所求之序向邏輯電路圖。

例題 8-4

試利用後緣觸發之 *JK* 正反器，實現例題 8-1 所示狀態圖之序向邏輯電路(使用轉態表)。

解

1. 所求狀態圖共有 4 個狀態，故需用 2 個正反器 ($n = \lceil \log_2 4 \rceil = 2$) 來實現所求之序向邏輯電路。

2. 因例題 8-2 之狀態表沒有次態與輸出相同之情況，故已為最簡之狀態表，故不需進行狀態化簡。

3. 依題意選擇後緣觸發 *JK* 正反器來實現所求之序向邏輯電路，並分別標示 J_A、K_A、J_B 與 K_B 為正反器之輸入符號，而分別標示 A_n 與 B_n 為正反器之輸出符號。

現　態	次　態		輸　出	
	$x=0$	$x=1$	$x=0$	$x=1$
a	a	b	0	0
b	c	b	0	0
c	a	d	0	0
d	a	b	0	1

4. 接著依序指定狀態 a 為「00」、狀態 b 為「01」、狀態 c 為「10」、狀態 d 為「11」後，可得一個新狀態表，如下表所示。

現　態		次　態				輸　出	
		$x=0$		$x=1$		$x=0$	$x=1$
A_n	B_n	A_{n+1}	B_{n+1}	A_{n+1}	B_{n+1}	y	y
0	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0
1	0	0	0	1	1	0	0
1	1	0	0	0	1	0	1

5. 依據步驟(4)所得之狀態表，配合 JK 正反器之激勵表，便可得到所求序向邏輯電路之轉態表，如下表所示。

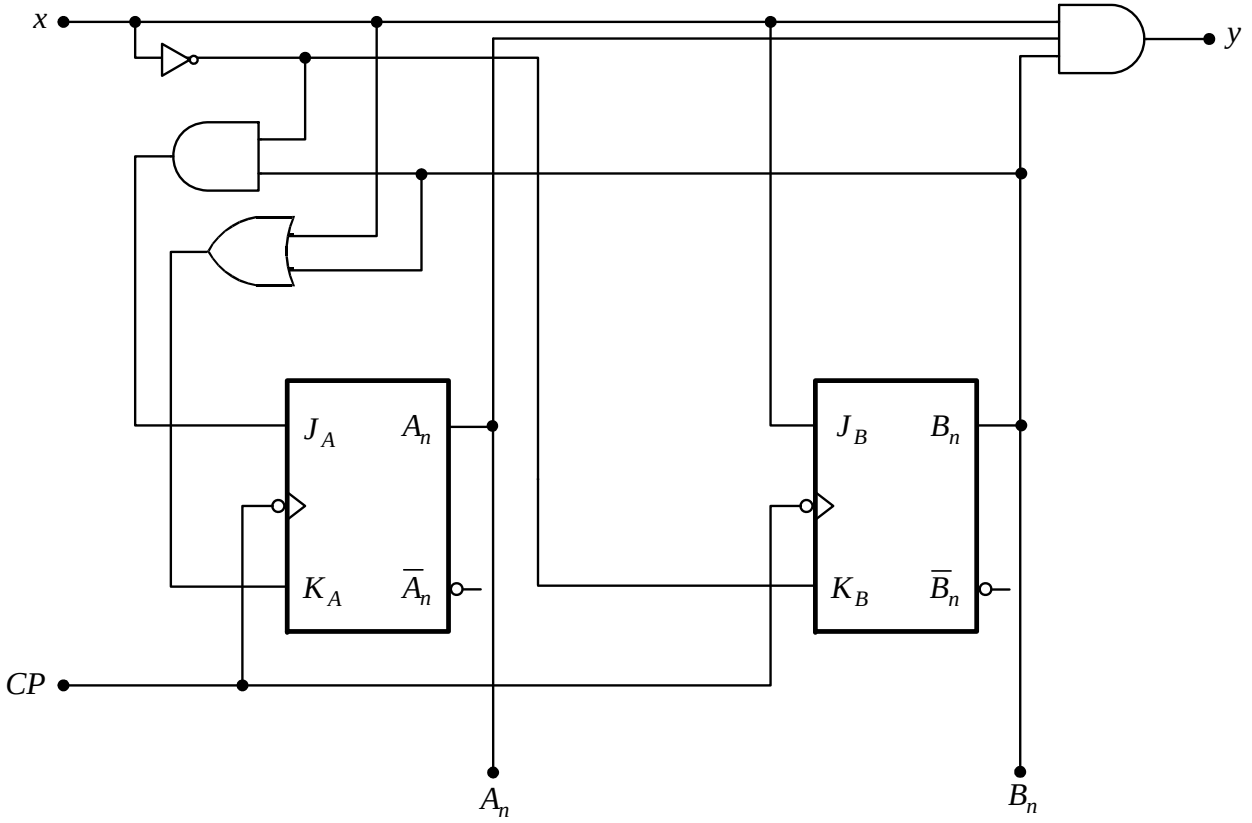
現　態			次　態		正反器之輸入				輸　出
x	A_n	B_n	A_{n+1}	B_{n+1}	J_A	K_A	J_B	K_B	y
0	0	0	0	0	0	×	0	×	0
0	0	1	1	0	1	×	×	1	0
0	1	0	0	0	×	1	0	×	0
0	1	1	0	0	×	1	×	1	0
1	0	0	0	1	0	×	1	×	0
1	0	1	0	1	0	×	×	0	0
1	1	0	1	1	×	0	1	×	0
1	1	1	0	1	×	1	×	0	1

6. 利用卡諾圖分別對上表之J_A、K_A、J_B與K_B等 *JK* 正反器之輸入與電路之輸出 y 進行邏輯化簡，即可得這些最簡之布林函數式如下：

$$J_A = \bar{x} \cdot B_n \qquad J_B = x \qquad y = x \cdot A_n \cdot B_n$$

$$K_A = \bar{x} + B_n \qquad K_B = \bar{x}$$

7. 使用 2 個後緣觸發之 *JK* 正反器，配合化簡所得之每一個正反器的輸入J_A、K_A、J_B、K_B與電路之輸出 y 的最簡布林代數，即可繪製所求之序向邏輯電路，如下圖所示。



例題 8-5

試利用後緣觸發之 *D* 型正反器，實現例題 8-1 所示狀態圖之序向邏輯電路(使用轉態表)。

解

1. **所求狀態圖共有 4 個狀態，故需用 2 個正反器 ($n = \lceil \log_2 4 \rceil = 2$) 來實現所求之序向邏輯電路。**

2. 因例題 8-2 之狀態表沒有次態與輸出相同之情況，故已為最簡之狀態表，故不需進行狀態化簡。

3. 依題意選擇後緣觸發 D 型正反器來實現所求之序向邏輯電路，並分別標示 D_A 與 D_B 為正反器之輸入符號，而分別標示 A_n 與 B_n 為正反器之輸出符號。

4. 接著依序指定狀態 a 為「00」、狀態 b 為「01」、狀態 c 為「10」、狀態 d 為「11」後，可得一個新狀態表，如下表所示。

現　態		次　態				輸　出	
		$x=0$		$x=1$		$x=0$	$x=1$
A_n	B_n	A_{n+1}	B_{n+1}	A_{n+1}	B_{n+1}	y	y
0	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0
1	0	0	0	1	1	0	0
1	1	1	0	0	1	0	1

5. 依據步驟(4)所得之狀態表，配合 D 型正反器之激勵表，便可得到所求序向邏輯電路之轉態表，如下表所示。

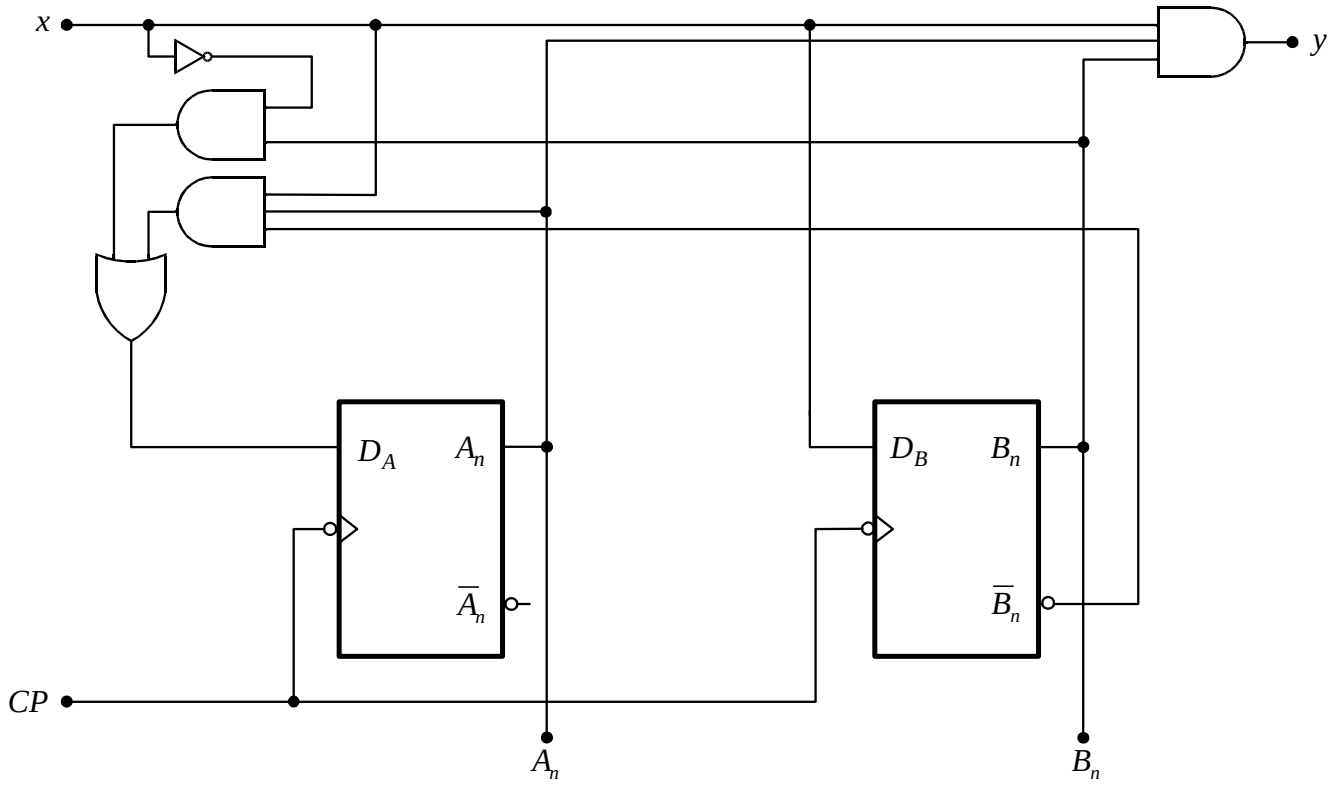
現　態			次　態		正反器之輸入		輸　出
x	A_n	B_n	A_{n+1}	B_{n+1}	D_A	D_B	y
0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	1	0
1	0	1	0	1	0	1	0
1	1	0	1	1	1	1	0
1	1	1	0	1	0	1	1

6. 利用卡諾圖分別對上表之D_A、D_B等 D 型正反器之輸入與電路之輸出 y 進行邏輯化簡，即可得到這些最簡布林代數式如下：

$$D_A = \overline{x} \cdot B_n + x \cdot A_n \cdot \overline{B}_n \quad 、 \quad D_B = x$$

$$y = x \cdot A_n \cdot B_n$$

7. 使用 2 個 D 型正反器與化簡所得之每一個正反器之輸入D_A、D_B與電路之輸出 y 的最簡布林代數，即可繪製所求之序向邏輯電路，如下圖所示。



例題 8-6

試利用後緣觸發之 *D* 型正反器，實現例題 8-1 所示狀態圖之序向邏輯電路 (請使用狀態方程式)。

解

1. 所求狀態圖共有 4 個狀態，故需用 2 個正反器 ($n = \lceil \log_2 4 \rceil = 2$) 來實現所求之序向邏輯電路。

2. 因例題 8-2 之狀態表沒有次態與輸出相同之情況，故已為最簡之狀態表，故不需進行狀態化簡。

3. 依題意選擇後緣觸發 D 型正反器來實現所求之序向邏輯電路，並分別標示 D_A 與 D_B 為正反器之輸入符號，而分別標示 A_n 與 B_n 為正反器之輸出符號。

4. 接著依序指定狀態 a 為「00」、狀態 b 為「01」、狀態 c 為「10」、狀態 d 為「11」後，可得一個新狀態表，如下表所示。

現　態		次　態				輸　出	
		$x=0$		$x=1$		$x=0$	$x=1$
A_n	B_n	A_{n+1}	B_{n+1}	A_{n+1}	B_{n+1}	y	y
0	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0
1	0	0	0	1	1	0	0
1	1	1	0	0	1	0	1

5. 依題意可得所求序向邏輯電路之狀態方程式(A_{n+1}與B_{n+1})與 D 型正反器輸出特性方程式(Q_{n+1})如下：

$$A_{n+1} = \overline{x} \cdot B_n + x \cdot A_n \cdot \overline{B}_n$$

$$B_{n+1} = x$$

$$Q_{n+1} = D_n$$

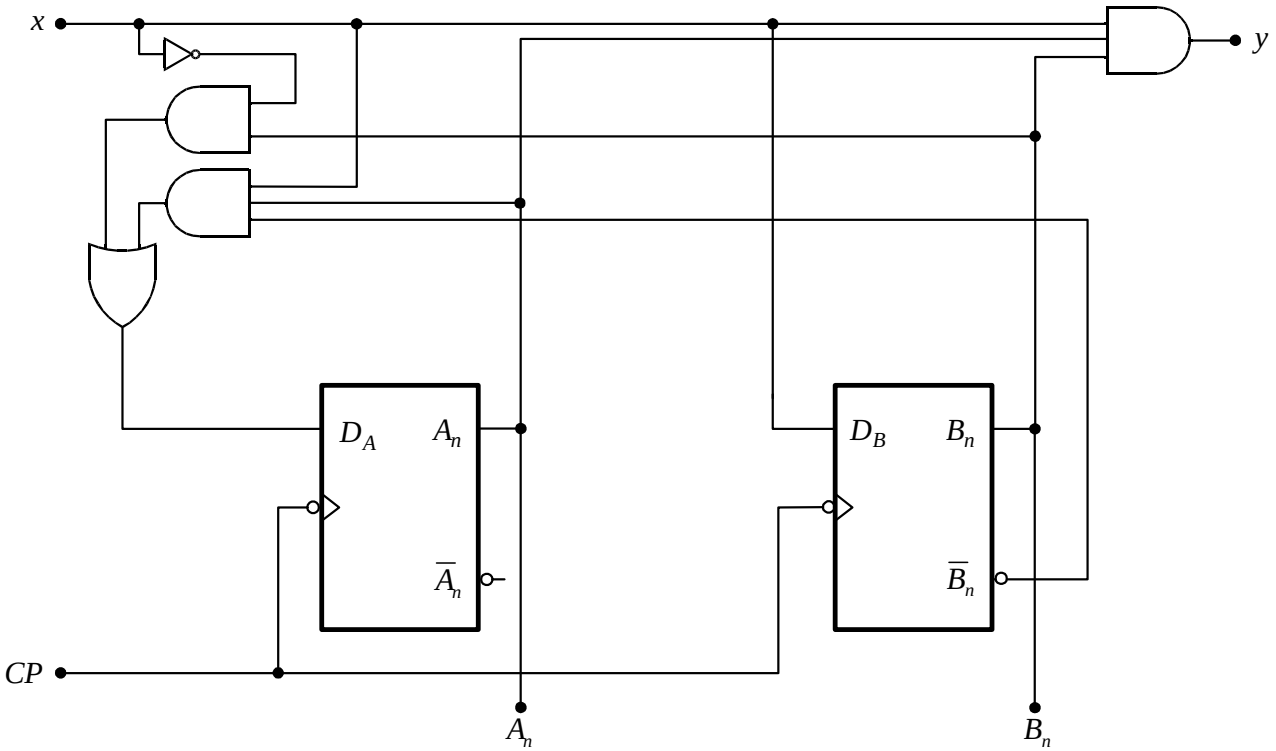
6. 令步驟 (5) 所得對應輸出之狀態方程式與特性方程式相匹配 (Q_{n+1}為正反器輸出之次態)，即可得 D 型正反器輸入 D_A 、 D_B 與輸出方程式 y 之布林函數式如下：

$$A_{n+1} = \overline{x} \cdot B_n + x \cdot A_n \cdot \overline{B}_n = D_A$$

$$B_{n+1} = x = D_B$$

$$y = x \cdot A_n \cdot B_n$$

7. 使用 2 個 D 型正反器與化簡所得之每一個正反器之輸入 D_A 、 D_B 與電路之輸出 y 的最簡布林代數，即可繪製所求之序向邏輯電路，如下圖所示。



註 1： 例題 8-5 與 8-6 分別使用正反器之**轉態表**與**狀態方程式**來求解序向邏輯電路，而所得之 D 型正反器之輸入 D_A、D_B 與輸出 y 之布林函數皆**相同**，故使用**不同之求解方法**，亦可得到**相同**之結果。

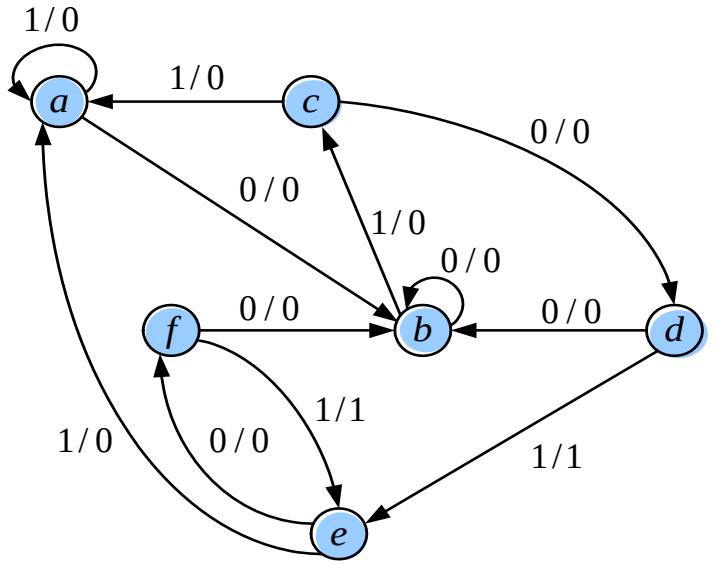
註 2： 使用**狀態方程式**，求解正反器之輸入端的最簡布林代數式，因牽涉到一些**複雜的代數運算**過程，往往比較容易發生**錯誤**之結果；而使用正反器之**轉態表**，求解正反器之輸入端的最簡布林代數式，以**卡諾圖**來進行化簡，故較**不容易發生錯誤**。

使用狀態化簡來設計序向邏輯電路

◆ 在狀態表中，若有兩個具有相同之次態與輸出之狀態，則表示此兩個狀態相等(Equivalent)，而兩個相等之狀態可合併成一個狀態，並將合併後之狀態給予一個新的狀態名稱，且必須將狀態表中，所有舊狀態名稱改變為合併後之新狀態名稱，此種處理過程稱為狀態化簡。

例題 8-7

試利用後緣觸發之 *T* 型正反器，實現下圖所示狀態圖之序向邏輯電路。



解

1. 觀察上圖可知，在進行狀態化簡前，此狀態圖共有 6 個狀態，故需要用 3 個正反器 ($n = \lceil \log_2 6 \rceil = 3$) 來實現所求之序向邏輯電路。

2. 接著將上圖之狀態圖轉換為狀態表，如下表所示，接著對下表次態與輸出皆相同之狀態(相等之狀態) 進行合併，即進行狀態化簡之工作。

現　態	次　態		輸　出	
	$x=0$	$x=1$	$x=0$	$x=1$
a	b	a	0	0
b	b	c	0	0
c	d	a	0	0
d	b	$e \rightarrow c$	0	1
e	$f \rightarrow d$	a	0	0
f	b	e	0	1

$e = c$

$f = d$

檢視上表可知，首先發現狀態 d 與狀態 f 之次態與輸出皆相同，故令狀態表中之所有 $f = d$ 後；接著又發現狀態 c 與狀態 e 之次態與輸出亦皆相同，故令狀態表中之所有 $e = c$ 後，而在刪除狀態 e 與狀態 f 後，可從新列出一個狀態化簡後之狀態表，如下表所示。

現　態	次　態		輸　出	
	$x=0$	$x=1$	$x=0$	$x=1$
a	b	a	0	0
b	b	c	0	0
c	d	a	0	0
d	b	c	0	1

觀察上表可知，經狀態化簡後，僅需使用 4 個狀態，即可用來表示所求之狀態圖，故僅用 2 個 T 型正反器 $(n = \lceil \log_2 4 \rceil = 2)$，便可用來實現所求之序向邏輯電路。

3. 依題意選擇後緣觸發之 T 型正反器來實現所求之邏輯電路，並分別標示 T_A 與 T_B 為正反器之輸入符號，而分別標示 A_n 與 B_n 為正反器之輸出符號。

4. 指定狀態 a 為「00」、狀態 b 為「01」、狀態 c 為「10」、狀態 d 為「11」，可得一個新狀態表，如下表所示。

現　態		次　態				輸　出	
		$x=0$		$x=1$		$x=0$	$x=1$
A_n	B_n	A_{n+1}	B_{n+1}	A_{n+1}	B_{n+1}	y	y
0	0	0	1	0	0	0	0
0	1	0	1	1	0	0	0
1	0	1	1	0	0	0	0
1	1	0	1	1	0	0	1

5. 並配合 T 型正反器之激勵表，便可列出所求序向邏輯電路之轉態表，如下表所示。

現　態			次　態		正反器之輸入		輸　出
x	A_n	B_n	A_{n+1}	B_{n+1}	T_A	T_B	y
0	0	0	0	1	0	1	0
0	0	1	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
1	0	0	0	0	0	0	0
1	0	1	1	0	1	1	0
1	1	0	0	0	1	0	0
1	1	1	1	0	0	1	1

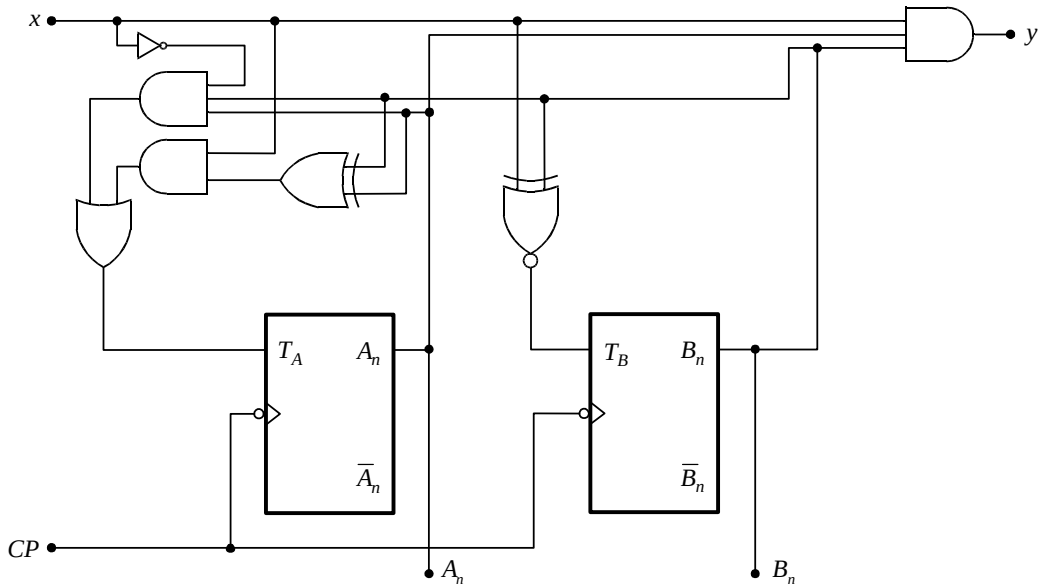
6. 利用卡諾圖分別對上表之T_A與T_B等 T 型正反器之輸入與電路之輸出 y 進行邏輯化簡，即可得到這些最簡布林代數式如下：

$$T_A = \bar{x} \cdot A_n \cdot B_n + x \cdot (\overline{A}_n \cdot B_n + A_n \cdot \overline{B}_n) = \bar{x} \cdot A_n \cdot B_n + x \cdot (A_n \oplus B_n)$$

$$T_B = x \cdot B_n + \bar{x} \cdot \overline{B}_n = x \odot B_n$$

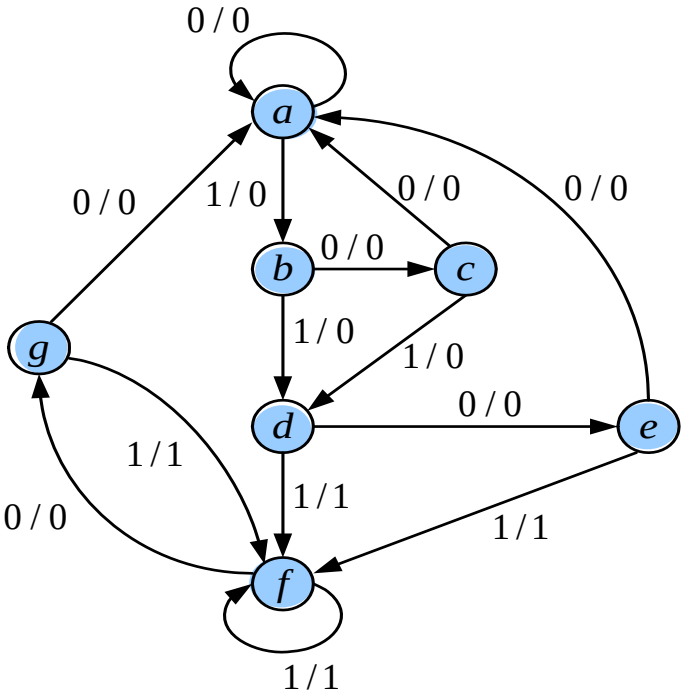
$$y = x \cdot A_n \cdot B_n$$

7. 最後使用 2 個 T 型正反器與化簡所得之每一個正反器輸入T_A、T_B與輸出 y 的最簡布林代數，即可繪出所求之序向邏輯電路圖，如下圖所示。



例題 8-8

試利用後緣觸發之 *JK* 正反器，實現下圖所示狀態圖之序向邏輯電路。



解

1. **觀察上圖可知，在進行狀態化簡前，此狀態有 7 個狀態，故需要用 3 個正反器** $(n = \lceil \log_2 7 \rceil = 3)$ **來實現所求之序向邏輯電路。**

2. 接著將上圖之狀態圖轉換為狀態表，並對次態與輸出皆相同之狀態(相等之狀態) 進行合併，即進行狀態化簡之工作。檢視下表可知，首先發現狀態 e 與狀態 g 之次態與輸出皆相同，故令狀態表中之所有 $e = g$；接著再發現狀態 d 與狀態 f 之次態與輸出亦皆相同，故令狀態表中之所有 $d = f$，因此於刪除狀態 f 與狀態 g 後，可從新列出一個狀態化簡後之狀態表，如下表所示。

現　態	次　態		輸　出	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	$f \rightarrow d$	0	1
e	a	$f \rightarrow d$	0	1
f	$g \rightarrow e$	f	0	1
g	a	f	0	1

$d = f$

$e = g$

接著將狀態化簡後之狀態表整理如下表所示：

現　態	次　態		輸　出	
	$x=0$	$x=1$	$x=0$	$x=1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

觀察左表可知，經狀態化簡後，僅需使用 5 個狀態，即可用來表示所求之狀態圖，故亦需用 3 個 *JK* 正反器 $(n=\lceil \log_2 5 \rceil = 3)$，才可用來實現所求之序向邏輯電路。

3. 依題意選擇後緣觸發之 *JK* 正反器來實現所求之邏輯電路，並分別標示 J_A、K_A、J_B、K_B、J_C 與 K_C 為正反器之輸入符號，而分別標示 A_n、B_n 與 C_n 為正反器之輸出符號。

4. 指定狀態 *a* 為「000」、狀態 *b* 為「001」、狀態 *c* 為「010」、狀態 *d* 為「011」與狀態 *e* 為「100」，可得一個新狀態表，如下表所示。

現　態			次　態						輸　出	
			$x=0$			$x=1$			$x=0$	$x=1$
A_n	B_n	C_n	A_{n+1}	B_{n+1}	C_{n+1}	A_{n+1}	B_{n+1}	C_{n+1}	y	y
0	0	0	0	0	0	0	0	1	0	0
0	0	1	0	1	0	0	1	1	0	0
0	1	0	0	0	0	0	1	1	0	0
0	1	1	1	0	0	0	1	1	0	1
1	0	0	0	0	0	0	1	1	0	1

5. 依據上表配合 *JK* 正反器之激勵表，便可列出所求序向邏輯電路之轉態表，如下表所示。

現　態				次　態			正 反 器 之 輸 入						輸出
x	A_n	B_n	C_n	A_{n+1}	B_{n+1}	C_{n+1}	J_A	K_A	J_B	K_B	J_C	K_C	y
0	0	0	0	0	0	0	0	×	0	×	0	×	0
0	0	0	1	0	1	0	0	×	1	×	×	1	0
0	0	1	0	0	0	0	0	×	×	1	0	×	0
0	0	1	1	1	0	0	1	×	×	1	×	1	0
0	1	0	0	0	0	0	×	1	0	×	0	×	0
1	0	0	0	0	0	1	0	×	0	×	1	×	0
1	0	0	1	0	1	1	0	×	1	×	×	0	0
1	0	1	0	0	1	1	0	×	×	0	1	×	0
1	0	1	1	0	1	1	0	×	×	0	×	0	1
1	1	0	0	0	1	1	×	1	1	×	1	×	1

6. 利用卡諾圖分別對上表之J_A、K_A、J_B、K_B、J_C與K_C等 JK 正反器之輸入與電路之輸出 y 進行邏輯化簡，即可得這些最簡之布林代數式如下：

$$J_A = \bar{x} \cdot B_n \cdot C_n$$

$$K_A = 1$$

$$J_B = C_n + x \cdot A_n$$

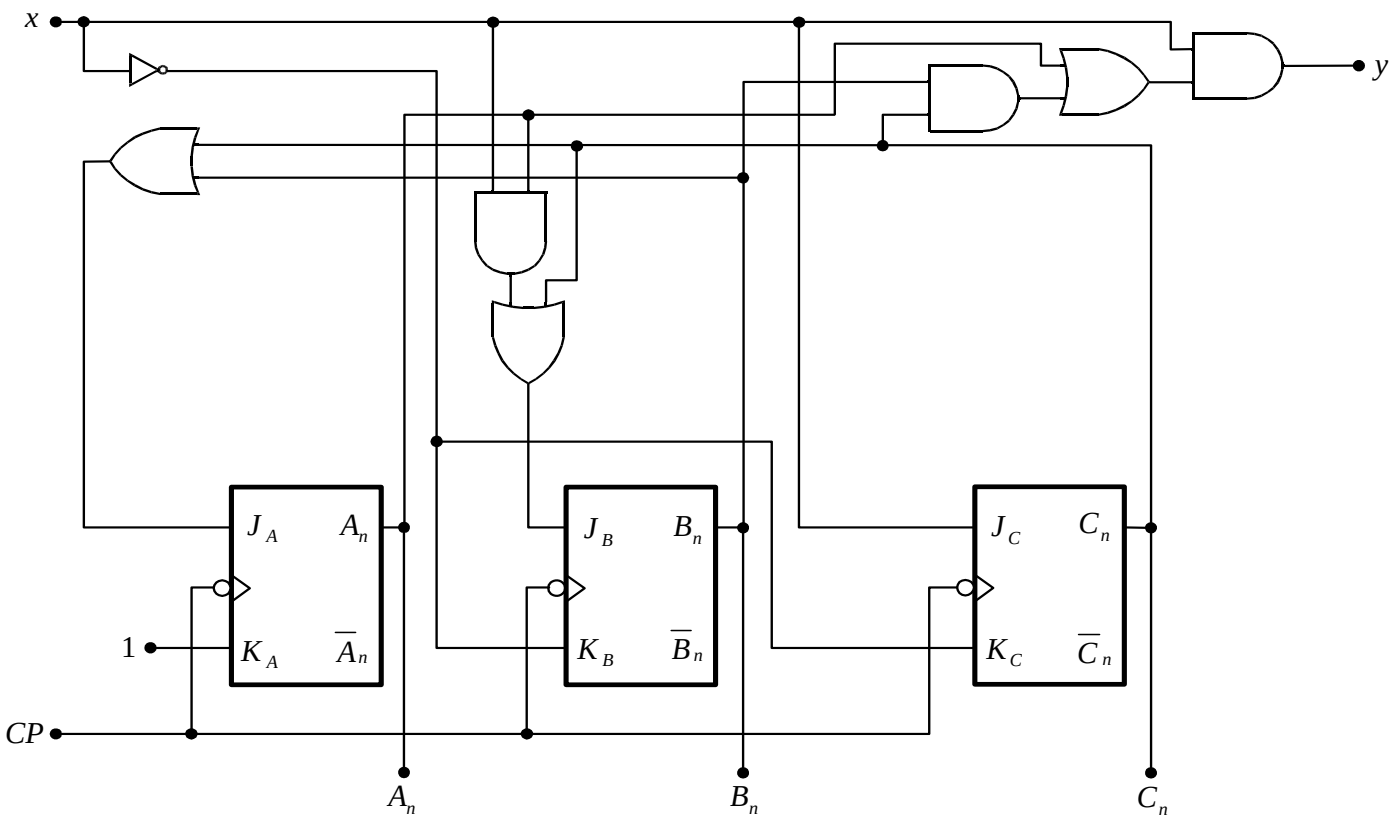
$$K_B = \bar{x}$$

$$J_C = x$$

$$K_C = \bar{x}$$

$$y = x \cdot (A_n + B_n \cdot C_n)$$

7. 最後使用 3 個 JK 正反器與化簡所得之每一個正反器輸入J_A、K_A、J_B、K_B、J_C、K_C與電路之輸出 y 的最簡布林代數，即可繪出所求之序向邏輯電路圖，如下圖所示。


x
y
J A
A n
1
K A
A n
J B
B n
K B
B n
J C
C n
K C
C n
CP
A n
B n
C n

例題 8-7 與例題 8-8 之綜合說明

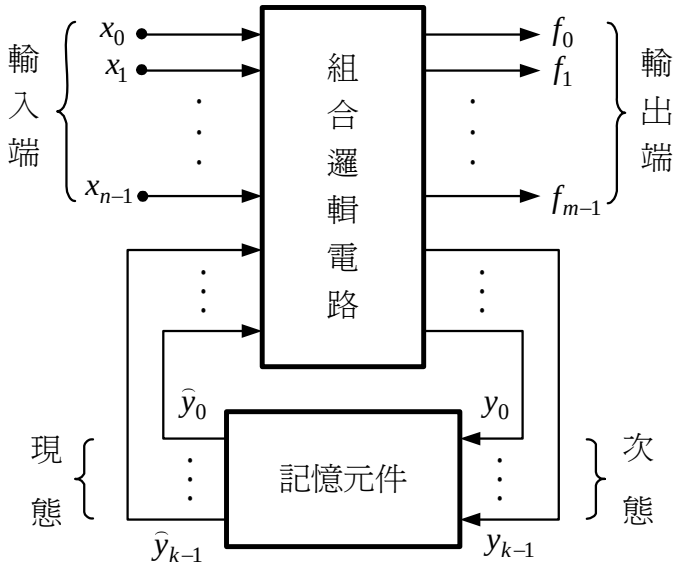
- 例題 8-7 之狀態圖，經過狀態化簡後，狀態數目可由 6 個減少為 4 個，因此設計圖 8-6 之序向邏輯電路時，可節省使用一個正反器，故可適度的降低實現序向邏輯電路之成本。
- 例題 8-8 之狀態圖，經過狀態化簡後，狀態數目可由 7 個減少為 5 個，因此設計此狀態圖之序向邏輯電路時，雖然無法減少正反器之使用數量，但在利用卡諾圖進行布林代數化簡時，以求取每個正反器輸入之最簡布林代數時，可增加不在意項 (Don't Care Term) 之數量，即可適度的簡化控制正反器輸入的組合邏輯之複雜度，故亦可降低實現邏輯電路之成本。
- 由以上之討論可知，在設計序向邏輯電路時，對所求之狀態圖進行狀態化簡，皆有助於降低實現序向邏輯電路之成本。

問題討論

1、試討論序向邏輯電路之基本觀念。

解：

當輸入端 $(x_0, x_1, ..., x_{n-1})$ **與記憶元件之輸出** $(\hat{y}_0, \hat{y}_1, ..., \hat{y}_{k-1})$ **等輸入訊號經過組合邏輯電路運算後，以產生輸出** $(f_0, f_1, ..., f_{m-1})$ **與記憶元件之輸入** $(y_0, y_1, ..., y_{k-1})$ **等訊號。而記憶元件之輸出訊號** $(\hat{y}_0, \hat{y}_1, ..., \hat{y}_{k-1})$ **稱為序向邏輯電路之現態 (Present State; *PS*)；記憶元件之輸入訊號** $(y_0, y_1, ..., y_{k-1})$ **稱為序向邏輯電路之次態 (Next State; *NS*)**



。即現態與輸入訊號經過組合邏輯電路之運算後，才可決定序向邏輯電路之輸出與次態，因此序向邏輯電路之狀態轉變 (State Transition) 不僅需考慮目前之輸入狀態，亦須考慮前一個輸出狀態，故序向邏輯電路是屬於一種記憶性電路 (Memory Circuits)。

2、試討論正反器之真值表與激勵表之關係。

解：

(1) 正反器之真值表 (Truth Table) 可用來描述目前輸入狀態 (現態；*PS*) 下，正反器下一個輸出狀態 (次態；*NS*)。

(2) 由已知之輸出狀態，以推導出之正反器輸入狀態，稱為正反器之激勵表 (Excitation Table)。

3、試討論序向邏輯電路之表示方法。

解：

因為序向邏輯電路是一種具記憶性電路，即電路之輸出狀態不僅與目前之輸入狀態有關，亦必須參考前一個輸出狀態，才可得到正確的輸出狀態，故必需使用狀態圖 (State Diagram) 與狀態表 (State Table) 或狀態方程式 (State Equation)，才可完整描述序向邏輯電路之規格與功能。

4、試討論使用激勵表與狀態方程式來設計序向邏輯電路之主要差異。

解：

(1) 使用激勵表來推導出轉態表後，再利用卡諾圖來化簡每一個正反器輸入之最簡布林代數表示式，以設計實際所需之序向邏輯電路。

(2) 使用正反器之特性方程式 (Characteristic Equation) 配合狀態方程式，以求解每一個正反器輸入之最簡布林代數表示式，以設計實際所需之序向邏輯電路。

5、試討論狀態化簡之基本原則與目的。

解：

(1) 在狀態表中，若有兩個狀態具有相同之次態與輸出，則表示此兩個狀態相等 (Equivalent)，而兩個相等之狀態可合併成一個狀態，並將合併後之狀態給予一個新的狀態名稱，且必須將狀態表中，所有舊狀態名稱改變為合併後之新狀態名稱，此為狀態化簡之基本原則。

(2) 經過狀態化簡後，若可適當減少狀態數量，則可節省使用正反器之數量；若經過狀態化簡後，無法減少正反器之使用數量，但在利用卡諾圖進行布林代數化簡時，以求取每個正反器輸入之最簡布林代數時，可增加不在意項 (Don't Care Term) 之數量，即可適度的簡化控制正反器輸入的組合邏輯之複雜度，故亦可降低實現邏輯電路之成本。由以上之討論可知，在設計序向邏輯電路時，對所求之狀態圖進行狀態化簡，皆有助於降低實現序向邏輯電路之成本。