

CH 6 數位邏輯

	布林代數基本運算子
☆	布林代數基本定理
	正規化表示方式 [Sum of min-terms Product of max-terms
☆ ⁵	電路化簡、卡諾圖之使用
☆ ⁴	Logical gate 邏輯單開元件種類、符號
☆ ³	Universal gate → 正反器
☆	組合電路、序向電路
☆	Pipeline 技術

1. 布林代數基本運算

- AND (•)

⇒ Input 皆為 1 的時候為 1，否則為 0

A	B	A•B
0	0	0
0	1	0
1	0	0
1	1	1

- OR(+)

⇒ Input 皆為 0 的時候為 0，否則為 1(有 1 個為 1 就 1)

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

- NOT

(3) NOT 否定 —, ' ,

Def: 1 變 0, 0 變 1.

A	\bar{A}
0	1
1	0

表3.2.2 基本邏輯閘與布林代數關係

邏輯閘	簡稱	布林代數	運算式
反向閘	NOT	NOT	$X = \bar{A}$
及閘	AND	AND	$X = A \cdot B$
或閘	OR	OR	$X = A + B$
反及閘	NAND	NOT-AND	$X = \overline{A \cdot B}$
反或閘	NOR	NOT-OR	$X = \overline{A + B}$
互斥或閘	XOR	EX-OR	$X = A \oplus B$
互斥反或閘	XNOR	EX-NOR	$X = \overline{A \oplus B}$

2. 布林代數基本定理(9 個)

(1) 單一律 $A \cdot A = A$
 $A + A = A$

(2) 基值律 $A \cdot 1 = A$ $A + 1 = 1$
 (基本律) $A \cdot 0 = 0$ $A + 0 = A$

(3) 補數律 $\bar{\bar{A}} = A$ $A + \bar{A} = 1$ $A \cdot \bar{A} = 0$

(4) 交換律 $A + B = B + A$ $A \cdot B = B \cdot A$

(5) 結合律 $(A \cdot B) \cdot C = A \cdot (B \cdot C)$
 $(A + B) + C = A + (B + C)$

(6) 分配律 $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$
 $A + (B \cdot C) = (A + B) \cdot (A + C)$

A	B	$(A+B)$	$\bar{A} \cdot \bar{B}$	$(\bar{A} \cdot \bar{B})$	$\bar{A+B}$
0	0	1	1	1	1
0	1	1	0	1	1
1	0	1	0	1	1
1	1	1	0	0	0

★(7) 笛摩根定理 對一個布林式子取補數(否定),
 則結果為:

$\left\{ \begin{array}{l} \circ \text{ 項次變補} \\ \circ \text{ 變} + \\ \circ \text{ 變} \cdot \end{array} \right.$

eg. $\overline{(A+B)} = \bar{A} \cdot \bar{B}$
 $\overline{(A \cdot B)} = \bar{A} + \bar{B}$

A	B	$A+B$	$\overline{A+B}$	\bar{A}	\bar{B}	$\bar{A} \cdot \bar{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

兩者所得結果相同

A	B	$A \cdot B$	$\overline{A \cdot B}$	\bar{A}	\bar{B}	$\bar{A} + \bar{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

兩者所得結果相同

(8) 吸收定理 eg. $x + xy = x$ pf: $x \cdot 1 + x \cdot y$
 $= x \cdot (1 + y)$
 $= x \cdot 1 = x$

eg. $x + xy + xz + x\bar{w} + x\alpha\beta\bar{r} = x \cdot (1 + y + z + \bar{w} + \alpha\beta\bar{r})$
 $= x \cdot 1 = x$

例: $A+B = A + \bar{A}B$

(6) 分配律

pf: $A+B = A+1 \cdot B = A+(A+\bar{A}) \cdot B = \underbrace{A+AB}_{(2) \text{ 基值律}} + \underbrace{\bar{A}B}_{(3) \text{ 補數律}} = \underline{A} + \bar{A}B$
(8) 吸收律

(9) 對偶律 用途: 已知某個布林式子成立, 則可以用對偶律轉換得到另一個也成立的布林式子。

轉換方式: $\begin{cases} -1 \text{ 變 } 0 & -0 \text{ 變 } 1 \\ -\cdot \text{ 變 } + & -+ \text{ 變 } \cdot \end{cases}$

eg. 已知成立

(1) $A + \bar{A} = 1 \xrightarrow{\text{對偶}} A \cdot \bar{A} = 0$

(2) $A \cdot 1 = A \xrightarrow{\text{對偶}} A + 0 = A$

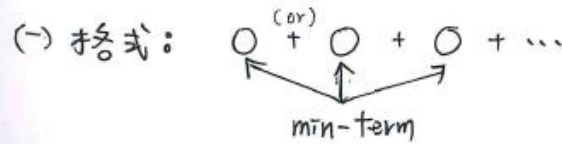
3. 正規化表示法

- 「最小項的和」(sum of minterms)

⇒ 將最小項以「OR」運算子結合, 也就是用「+」運算子

結合，便是「最小項的和」

⇒ N 個變數， 2^n 個組合



(=) Min-term: 包含所有輸入變數之乘積項

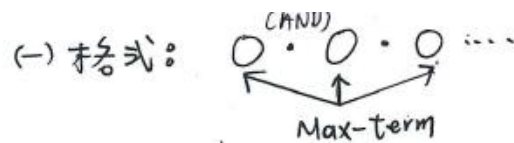
e.g. 2個變數(A,B) ⇒ 4個 min-terms: $AB, \bar{A}B, A\bar{B}, \bar{A}\bar{B}$.

e.g. 3個變數(A,B,C) ⇒ 8個 min-terms: $\bar{A}\bar{B}\bar{C}, AB\bar{C}, \bar{A}B\bar{C}, \dots, ABC$

- 「最大項的積」(product of maxterms)

⇒ 將最大項以「AND」運算子結合，也就是用「 \cdot 」運算子

結合，便是「最大項的積」



(=) Max-term: 包含所有輸入變數之和項

e.g. 2個變數(A,B) ⇒ 4個 Max-term: $(A+B), (\bar{A}+B), (A+\bar{B}), (\bar{A}+\bar{B})$

e.g. 3個變數(A,B,C) ⇒ 8個 Max-term: $(A+B+C), (\bar{A}+B+C), \dots, (\bar{A}+\bar{B}+\bar{C})$

Example

$F = ABC' + A'B' + BC'$ 3 variables 7 literals

Row No.	A	B	C	Mintems			Maxterms			f	f'
0	0	0	0	$A'B'C'$	=	m_0	$A+B+C$	=	M_0	0	1
1	0	0	1	$A'B'C$	=	m_1	$A+B+C'$	=	M_1	0	1
2	0	1	0	$A'BC'$	=	m_2	$A+B'+C$	=	M_2	0	1
3	0	1	1	$A'BC$	=	m_3	$A+B'+C'$	=	M_3	1	0
4	1	0	0	$AB'C'$	=	m_4	$A'+B+C$	=	M_4	1	0
5	1	0	1	$AB'C$	=	m_5	$A'+B+C'$	=	M_5	1	0
6	1	1	0	ABC'	=	m_6	$A'+B'+C$	=	M_6	1	0
7	1	1	1	ABC	=	m_7	$A'+B'+C'$	=	M_7	1	0

$$m_i' = Mi$$

$$\begin{aligned} \Rightarrow f &= A'BC + AB'C' + AB'C + ABC' + ABC \\ &= m_3 + m_4 + m_5 + m_6 + m_7 \\ \text{or } f(A, B, C) &= \sum m(3, 4, 5, 6, 7) \\ \text{min term} = 1 &\Rightarrow f = 1 \end{aligned}$$

$$\begin{aligned} f &= (A + B + C)(A + B + C')(A + B' + C) = M_0 M_1 M_2 \\ f(A, B, C) &= \prod M(0, 1, 2) \\ \text{maxterm} = 0 &\Rightarrow f = 0 \end{aligned}$$

4. 電路化簡

(一) 目的：用最少的邏輯閘完成相同的電路功能，以節省成本。

(二) 方法：有很多種，其中以“卡諾圖”最方便。

(三) 電路化簡，分2種

⇒ 化簡成最簡的 SOP (Sum of Product) (積) + (積) + (積)
 或化簡成最簡的 POS (Product of Sum) (和) · (和) · (和)

四 卡諾圖介紹。

(一) 以化簡最簡 SOP 為例：n 個變數 ⇒ 有 2^n 個方格，每個方格

Note: POS ⇒ 方格代表 Max-term

代表 Min-term (最小項)

- SOP(卡諾圖)

化簡的步驟：

1. 將在真值表中可產生 1 的每個基礎乘積項，對應的填入卡諾圖的空格中，並標記為 1，其他的空格則填入 0。
2. 依序圈出相鄰的 8 個 1、相鄰的 4 個 1、相鄰的 2 個 1，空格中的 1 可被重複圈選，以便消除最多的變數。
3. 如果還留下獨立的 1，也要個別圈選。
4. 觀察圈選的狀況，要讓所有 1 的空格都被圈到，而圈選的組數要愈少愈好。
5. 每一個圈選的結果是一個乘積項，將所有的乘積項 OR 起來即是化簡後的布林代數式。

$$(a) X = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C}$$

	\bar{C}	C
$\bar{A}\bar{B}$	0	0
$\bar{A}B$	1	0
AB	1	0
$A\bar{B}$	0	0

消去互補變數A → $X = B\bar{C}$

$$(a) X = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C}$$

	\bar{C}	C
$\bar{A}\bar{B}$	1	0
$\bar{A}B$	0	0
AB	0	0
$A\bar{B}$	1	0

消去互補變數A → $X = \bar{B}\bar{C}$

上下兩列亦屬於相鄰的方格

$$(c) X = \bar{A}\bar{B}CD + \bar{A}\bar{B}C\bar{D} + A\bar{B}C\bar{D} + A\bar{B}C\bar{D}$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	1	1
$\bar{A}B$	0	0	0	0
AB	0	0	0	0
$A\bar{B}$	1	0	0	1

消去互補的變數D → $X = \bar{A}\bar{B}C + A\bar{B}\bar{D}$

消去互補的變數C

左右兩列亦屬於相鄰的方格

利用卡諾圖化簡 $X = \bar{A}\bar{B}\bar{C} + \bar{B}C + \bar{A}B$ 之布林代數式

\bar{C}	C
$\bar{A}\bar{B}$	1
$\bar{A}B$	
AB	
$A\bar{B}$	

第一項 $\bar{A}\bar{B}\bar{C}$

\bar{C}	C
$\bar{A}\bar{B}$	
$\bar{A}B$	
AB	
$A\bar{B}$	1

第二項 $\bar{B}C$
因缺少變數A
所以只要卡諾圖
空格中有 $\bar{B}C$ 的
皆應填入1

\bar{C}	C
$\bar{A}\bar{B}$	
$\bar{A}B$	1
AB	1
$A\bar{B}$	

第三項 $\bar{A}B$
因缺少變數C
所以只要卡諾圖
空格中有 $\bar{A}B$ 的
皆應填入1

\bar{C}	C
$\bar{A}\bar{B}$	1
$\bar{A}B$	1
AB	
$A\bar{B}$	

化簡後 $X = \bar{A} + \bar{B}C$

④ 化成最簡的 POS (Product of Sum)

- 步驟相同，差別 \Rightarrow
- 每一方格代表最大項，填" ϕ "
 - 合併相鄰方格時，要寫出和項 (1 變補， ϕ 不補)
 - 最後再乘起來。

例: $F(A, B, C) = \pi_1(0, 1, 3, 4)$ 化成 Simplified POS

		BC							
		00	01	11	10	00	01	11	10
A	0	0	0	1	1	0			
	1	0							

$$\square: (A + \bar{C})$$

$$\therefore (A + \bar{C}) \cdot (B + C)$$

$$\square: (B + C)$$

例: $F(A, B, C, D) = (\bar{A} + \bar{B} + \bar{D}) \cdot (\bar{A} + C + \bar{D}) \cdot (A + \bar{B} + \bar{D}) \cdot (\bar{B} + C + D)$

		CD							
		00	01	11	10	00	01	11	10
AB	00								
	01	0	0	0					
11	11	0	0	0					
	10		0						

$$\square: (\bar{B} + C)$$

$$\square: (\bar{B} + \bar{D}) \quad \therefore (\bar{B} + C) \cdot (\bar{B} + \bar{D}) \cdot (\bar{A} + C + \bar{D})$$

$$\square: (\bar{A} + C + \bar{D})$$

- 隨意條件 (Don't care condition)

\Rightarrow 並非所有的輸入狀況皆會發生，其對輸出不重要

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	X
1	0	0	X
1	0	1	1
1	1	0	1
1	1	1	1

	\bar{C}	C
$\bar{A}\bar{B}$	0	0
$\bar{A}B$	0	X
$A\bar{B}$	X	1
AB	1	1

當成1 (pointing to $A\bar{B}$)

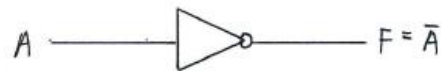
當成0 (pointing to $\bar{A}B$)

	\bar{C}	C
$\bar{A}\bar{B}$	0	0
$\bar{A}B$	0	0
$A\bar{B}$	1	1
AB	1	1

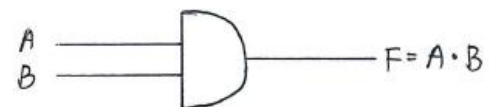
Y=A

5. 邏輯閘(Logic Gate)

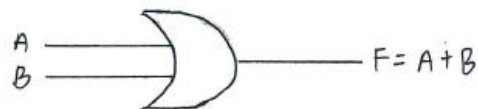
(1) NOT 否定



(2) AND gate



(3) OR gate



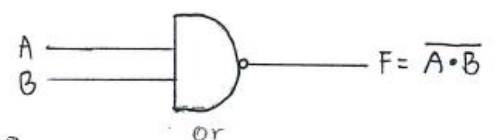
真值表.

A	B	$\bar{A}\bar{B}$
0	0	1
0	1	1
1	0	1
1	1	0



(4) NAND gate (NOT AND)

Def: AND 輸出再否定

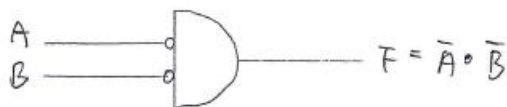
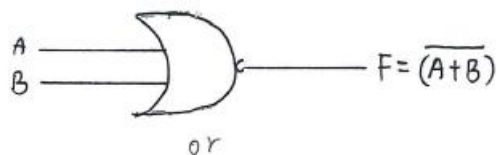


摩根定理



(5) NOR gate (NOT OR)

Def: OR 輸出再否定



真值表

A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

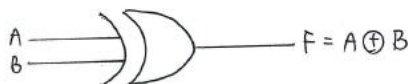
(6) XOR gate (exclusive-OR gate 互斥或)

• 相同為 0, 相異為 1。

• 真值表

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

• $A \oplus B \Leftrightarrow \overline{A}B + A\overline{B}$ 輪流補



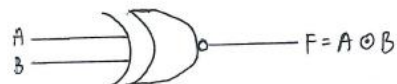
(7) XNOR gate (exclusive-NOR 互斥反或)

• 相同為 1, 相異為 0。

• 真值表

A	B	$A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1

• $A \odot B \Leftrightarrow \overline{A} \overline{B} + AB$ 同為補 + 同為不補



例: $\overline{A \oplus B} = A \odot B$

$$\begin{aligned}
 \text{Pf: } \overline{A \oplus B} &= \overline{\overline{A}B + A\overline{B}} = \overline{\overline{A}B} \cdot \overline{A\overline{B}} = (A + \overline{B}) \cdot (\overline{A} + B) \\
 &= A\overline{A} + AB + \overline{B}\overline{A} + \overline{B}B \\
 &= \overline{A}\overline{B} + AB = A \odot B
 \end{aligned}$$

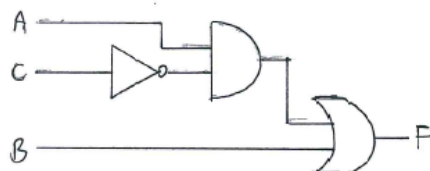
例 1: $F(A, B, C) = \overline{A}B + A\overline{B}\overline{C} + AB$ 化成 simplified SOP
並繪出電路圖 (Circuit Diagram)

Ans:

	BC	00	01	11	10
A	0			1	1
0				1	1
1		1		1	1

$\therefore B + A\overline{C}$

電路圖如下:



6. Universal Gate C Complete Gate 萬用閘/完整閘

Def: 單用此閘可以完成所有電路 (或某些 gates) 功能。

即 NAND 和 NOR gate 皆是。

7. XOR 與 XNOR 特性探討

(1) $A \oplus 1 = \bar{A}$ $A \odot 1 = A$
 $A \oplus 0 = A$ $A \odot 0 = \bar{A}$
 $A \oplus A = 0$ $A \odot A = 1$
 $A \oplus \bar{A} = 1$ $A \odot \bar{A} = 0$

XOR XNOR 結果差一個NOT

(2) ④ 互斥或是 Even-parity Bit generator (偶同位位元產生器) 跟 10 有關
即所有 Bits 值作互斥或, 即得 Even-Parity Bit \downarrow 即知。

γ : | 0 | 1 | 0 | □
| ⊕ 0 | 1 | 1 | 0 |
✓ — ○ — ✓ — ✓ — □

- input中若有偶数个"1" 作 \oplus 会得0
- input中若有奇数个"1" 作 \oplus 会得1

↓ 对偶律

- input中若有偶数个" \emptyset ", 作 \odot 得1
- input中若有奇数个" \emptyset ", 作 \odot 得 \emptyset

8. 電路種類

- 組合電路(Combinational Circuit)

四 組合電路之設計

(-) steps: ① 依據問題, 定出所需的 input 介叔, 及 output 介叔.

② 依問題(功能)畫出 Truth Table

③ 用卡諾圖實施電路化簡。

④繪出電路圖。

例1: 比較器. 若 $A \geq B$ 則 output 為 1. 否則為 0.

Ans: ① Input 2 J: (A, B)
output 1 J: (out)

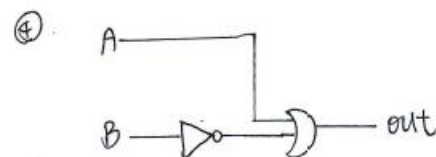
⑦

Z		
A	B	out
0	0	1
0	1	0
1	0	1
1	1	1

⑦

	B	0	1
A	0	1	
1		1	1

$$\therefore \text{OxT}(A, B) = A + \overline{B}$$



四 加法器.

- (1) 分爲二種
- I. 半加器 (Half Adder): { 輸入: A, B 2个
輸出: Sum (和) 與 Cout (進位)
 - II. 全加器 (Full Adder): { 輸入: A, B, C 3个
輸出: Sum 與 Cout

I. 半加器.

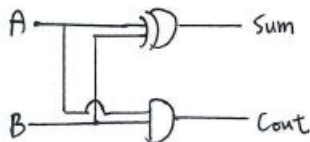
+ 進位			
A	B	Sum	Cout
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Sum:	B	0	1
A	0		1
1	1		

$$Sum = \bar{A}B + A\bar{B} = A \oplus B.$$

$$Cout: \underline{AB}$$

∴ 電路:



II. 全加器.

A	B	C	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Sum:	B	C	00	01	11	10
A	0		1			1
1	1			1		

$$Sum = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + AB\bar{C} = A \oplus B \oplus C$$

Cout:	B	C	00	01	11	10
A	0			1		
1		1			1	1

⇒ 在此不 follow 卡諾圖原則. 而是要再利用先前某些

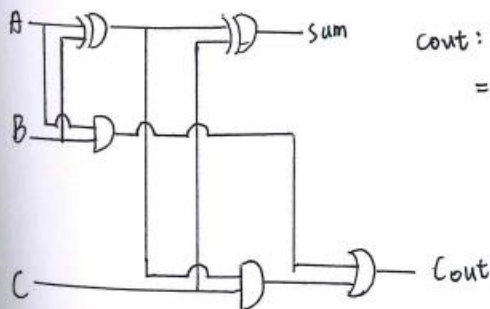
① 成果 of Sum.

$$Cout: \bar{A}B\bar{C} + A\bar{B}C + AB$$

$$= C \cdot (\bar{A}B + A\bar{B}) + AB = C \cdot (A \oplus B) + AB$$

相當於是 2 个半加器 及 1 个 OR.

∴ 電路圖.



減法器

分類

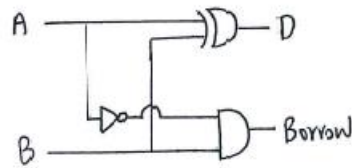
- I. 半減器 { 輸入: A, B (2個) 即 A-B
輸出: D 差值與 Borrow 借位.
- II. 全減器 { 輸入: A, B, C (3個) 即 A-B-C
輸出: D 與 Borrow

I. 半減器

A	B	D	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	1

$$\therefore D = \bar{A}B + A\bar{B} = A \oplus B$$

$$\text{Borrow} = \bar{A}B$$



II. 全減器

Inputs: A, B, C 3個 (即 A-B-C)

outputs: D 及 Borrow

A	B	C	D	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

D:

BC	00	01	11	10
A=0	0	1	1	1
A=1	1	0	0	0

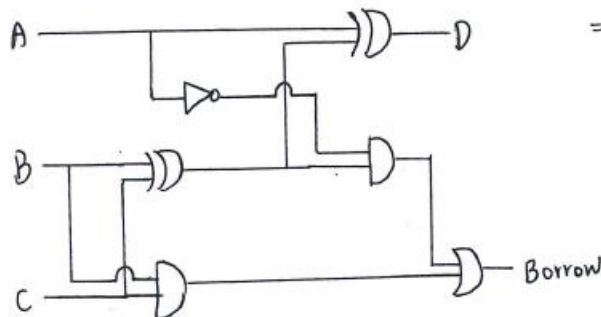
$$D = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC = A \oplus B \oplus C$$

Borrow:

BC	00	01	11	10
A=0	0	1	1	1
A=1	1	0	0	0

不遵守卡諾圖化簡原則. 想要再利用某些 ⊕ 成果 in D

電路圖:



$$\text{Borrow} = BC + \bar{A}BC + \bar{A}B\bar{C}$$

$$= BC + \bar{A} \cdot (B\bar{C} + BC)$$

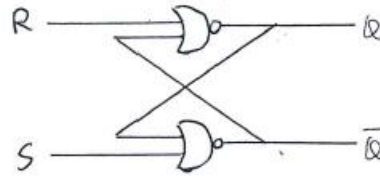
$$= BC + \bar{A} \cdot (B \oplus C)$$

- 序向電路(Sequential Circuit)

⇒ 正反器(Flip-Flop)

图: 以 NOR gate 为例.

R, S 为 inputs, Q 为输出 (\bar{Q} 为 Q 之补数)



R	S	上次输出 Q	本次输出 Q	\bar{Q}
1	0	0	0	1
1	0	1	0	1
1	1	0	0	0
1	1	1	0	0
0	0	0	0	1
0	0	1	1	0

★ NOR 特性: Input 只要有 1 个为 1, 输出必为 0

} 矛盾, 应避免之

R	S	Q	\bar{Q}
1	0	0	1
0	1	1	0
0	0	Q_t	\bar{Q}_t
1	1		

⇒ Reset (R)

⇒ Set (S)

⇒ Q_t : 上次输出, 即维持原态. (上次输出为 0 或 1) 即记忆功能.

1/1 矛盾, 不会发生, 宜避免之. 一般來說 undefined

• 激励表 Exciting Table

上次 Q_t	本次 Q	R	S
1	0	1	0
1	1	0	X
0	0	X	0
0	1	0	1

R, S: inputs.

⇒ 改变状态 Reset

⇒ { Set: S=1, R=0
维持原态: S=0, R=0

⇒ { Reset: S=0, R=1
维持原态: S=0, R=0

⇒ Set

R 一定要是 0, S 1 或 0 不在乎
"X": don't care.

- 比較

組合電路	序向電路
<ul style="list-style-type: none"> ◦ 電路之輸出只與輸入的組合相關與時序無關。與上一次輸出無關 	<ul style="list-style-type: none"> ◦ 輸出會與本次輸入組合。與上次輸出結果共同決定與時序 (Timing) 相關。
<ul style="list-style-type: none"> ◦ 相同的輸入組合必得相同 (一致) 之結果。 	<ul style="list-style-type: none"> ◦ 相同的輸入組合。不見得得到相同結果。
<ul style="list-style-type: none"> ◦ 不具備回饋 (feedback) 路徑之電路。 	<ul style="list-style-type: none"> ◦ 具備
<ul style="list-style-type: none"> ◦ 不具備記憶 (memory) 之能力 	<ul style="list-style-type: none"> ◦ 具備
<ul style="list-style-type: none"> ◦ 例: 比較器. 編碼器. 加法器. 解碼器. 減法器. 計數器. 多工器. (counting) 解多工器... etc. 	<ul style="list-style-type: none"> ◦ 例: 正反器 (flip-flop) 記憶體. 連波計數器. etc.

9. Pipeline(管線式)

- 公式

⇒ $(\text{Instruction count} + (\text{pipeline stage} - 1)) * \text{Clock cycle time}$

⇒ 執行到最後一條指令時，還需等 (stage - 1) 個 cycle，作後

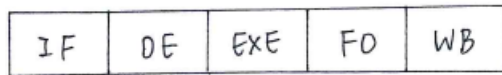
一個指令才會完全做完

(一) 目的

加速多条指令之执行 (缩短多条指令之执行时间)

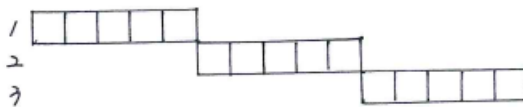
(二) Non-pipeline

假设机器指令有 5 个 stages. 如下:



每个 stage 各花 1 个 clock cycle. 执行 3 条指令. 共花 $3 \times 5 = 15$ 个 clock cycles. (循序执行)

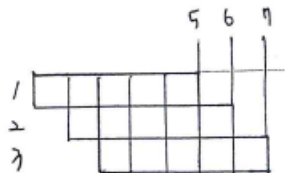
图:



(三) pipeline 技术

不同指令之不同 stage 可以 overlay execution (重叠 / 平行执行)

→ 加速整体指令之执行.



3 条指令共花 7 个 clock cycles.

☆ pipeline 之公式 (10)

令 N 为指令数目.

则 pipeline time

$$S: \text{指令 stage 数目.} = [N + (S - 1)] * \text{MAX} \{ \text{各 stage 之 Time} \} \leftarrow (\text{Instruction count} + (\text{pipeline stage} - 1)) * \text{Clock cycle time}$$

例: 机器指令有 5 个 stages 如下: Ans:



执行 10 条指令共花 ? ns

① Non-pipeline ② pipeline

① Non-pipeline

$$10 * (10 + 5 + 10 + 10 + 5) = 10 * 40 = 400 \text{ ns}$$

② pipeline

$$[10 + (5 - 1)] * \text{MAX} \{ 10, 5, 10, 10, 5 \} \\ = 14 * 10 = 140 \text{ ns.}$$

10. 命题演算

(一) $P \Rightarrow Q$ (唸法: 若 P 則 Q)

def: 等同於 $\sim P \vee Q$ ($\bar{P} + Q$)

(\sim : NOT, \wedge : AND, \vee : OR)

P	Q	$P \Rightarrow Q$
0	0	1
0	1	1
1	0	0
1	1	1

(二) $P \Leftrightarrow Q$ (P 若且唯若 Q)

def: 等同於 $(P \Rightarrow Q) \wedge (Q \Rightarrow P)$

$$(\sim P \vee Q) \wedge (\sim Q \vee P)$$

$$= (\bar{P} + Q) \cdot (\bar{Q} + P)$$

$$= \bar{P}\bar{Q} + \bar{P}P + Q\bar{Q} + QP$$

$$= \bar{P}\bar{Q} + PQ = P \oplus Q$$

(互斥或)

P	Q	$P \Leftrightarrow Q$	$P \Rightarrow Q$	$Q \Rightarrow P$	$(P \Rightarrow Q) \wedge (Q \Rightarrow P)$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	1	1	1	1

一致

(三) Tautology 恆真.

不論 input 之組合為何, output 一律為 True (1)

Note: 若以卡諾圖來看, 即每一個方格均填入 "1"