# 第四章 晶片 VHDL 模組單元設計

承第三章第一節，圖 11 與圖 12 所示，FRC 晶片設計共可分為五大單元，即資料流控制單元(Datapath Controller)，位址產生單元(Address Generation Unit, AGU)指紋前處理模組單元(包含 Binarization Unit 與 Thinning Unit)，編碼運算單元(包含 Ending Point Unit 與 Bifurcation Point Unit)與輸出入介面處理單元(Input/Output Interface Unit)。其中，Thinning Unit 與編碼運算單元，使用將演算法結果轉成查表的硬體化方式，所以硬體化後，只要修改查表內容，便可以適應不同演算法，符合可撓性的設計原則；而 AGU 的設計可適用於，產生 3x3 與 5x5 遮罩運算所需的位址，以及配合資料流控制單元設計，方便未來擴充為 7x7、9x9 及 13x13 等不同尺寸大小的影像遮罩，以符合可擴充性的設計；此外，資料流控制單元在上一章併同整體 FRC 晶片架構已詳談過，本章將針對其餘四個模組詳予介紹。

## 第一節 位址產生單元

位址產生單元(Address Generation Unit, AGU)主要負責的工作有兩部份，第一部份就是將以 P 點為中心，位址座標為(i,j)，其中 i 代表 Col，而則 j 代表 Row，產生 3x3 或 5x5 遮罩內的每一個點(pixel)的座標位址，以 P 點為中心之 3x3 與 5x5 對應位址如圖 24 所示，由圖中可以知道其所對應的硬體該如何設計，在圖 24(a)為 3x3 的遮罩與其所對應的位址座標，而圖 24(b)為 5x5 的遮罩，觀察其 i 與 j 的變化，可以知道在 3x3 的遮罩中的 P1’~P8’，對於 i 或 j 的變化量都是介於+/- 1 的，而在 5x5 遮罩中 P1~P24，其 i 與 j 的變化量則為+/- 2。

| | | |
|---|---|---|
| P1'<br>(i-1,j-1) | P2'<br>(i,j-1) | P3'<br>(i+1,j-1) |
| P8'<br>(i-1,j) | P<br>(i,j) | P4'<br>(i+1,j) |
| P7'<br>(i-1,j+1) | P6'<br>(i,j+1) | P5'<br>(i+1,j+1) |

| | | | | |
|---|---|---|---|---|
| P11<br>(i-2,j-2) | P12<br>(i-1,j-2) | P13<br>(i,j-2) | P14<br>(i+1,j-2) | P15<br>(i+2,j-2) |
| P10<br>(i-2,j-1) | P1<br>(i-1,j-1) | P2<br>(i,j-1) | P3<br>(i+1,j-1) | P16<br>(i+2,j-1) |
| P9<br>(i-2,j) | P8<br>(i-1,j) | P<br>(i,j) | P4<br>(i+1,j) | P17<br>(i+2,j) |
| P24<br>(i-2,j+1) | P7<br>(i-1,j+1) | P6<br>(i,j+1) | P5<br>(i+1,j+1) | P18<br>(i+2,j+1) |
| P23<br>(i-2,j+2) | P22<br>(i-1,j+2) | P21<br>(i,j+2) | P20<br>(i+1,j+2) | P19<br>(i+2,j+1) |

*其中i代表Col,j代表Row

(a) (b)

圖 24 以 P 為中心使用遮罩所對應座標
(a)3x3 遮罩
(b)5x5 遮罩

觀察圖 24 可以發現在圖 24(b) 5x5 遮罩中的 P1~P8 與圖 24(a) 3x3 遮罩中的 P1’~P8’的運算是相同的，也就是說，這兩個部份的硬體資源可以相互共用，因為 5x5 的遮罩內容包含了 3x3 遮罩的部份，如圖 25 所示。

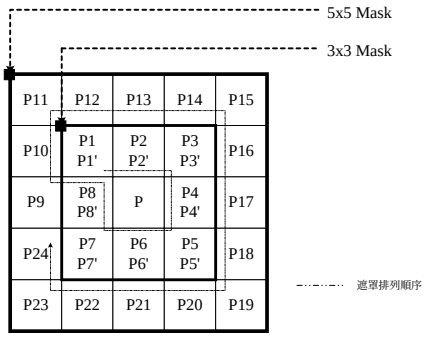



圖 25 5x5 遮罩與 3x3 遮罩共用關係

而 AGU 第二部份的工作，就是要讓 FRC 可以很容易的產生外掛式記憶體中的任一位址，也就是說可以任意取得外掛式記憶體中任一位址的資料。

目前所決定的指紋影像大小為 2D 影像 200(Row)x200(Col)x8=40KB，要如何產生這 40KB 中任一位址的資料呢？事實上 200(Row)x200(Row)的指紋影像在實際記憶體中是為 1D 的陣列排列，而 AGU 所做的動作就是將 2D 的陣列轉換成 1D 的陣列行為。可以透過公式(5)，瞭解在數學的轉換上如何將 200x200 的 2D 陣列轉成 0~39999 的 1D 陣列。

$$\text{Memory Address}[m]|_{m=0\sim39999} = 200 \times \text{Row}|_{\text{Row}=0\sim199} + \text{Col}|_{\text{Col}=0\sim199} \quad \text{.....(5)}$$

只要透過公式(5)代入 Row 與 Col 便可以得到在記憶體中實際的位址。其中 Col 可視為一幅影像上的 X 軸座標，Row 則可視為 Y 軸上的座標。

根據圖 25 可以清楚的知道 3x3 遮罩的 P1’~P8’與 5x5 遮罩的 P1~P8 相同，所以在硬體化時可將這兩部份的硬體結合在一起，可利用選擇線選擇 3x3 遮罩或 5x5 遮罩。因為在同一個時間，只能對外掛式的記憶體存取一個位址的資料，所以若要載入 3x3 遮罩所對應的位址，則必需花掉 8 個時脈(P1’~P8’)；同理，若要載入

5x5 遮罩所對應的位址，則必需花掉 24 個時脈(P1~P24)。

安排在 1~8 個時脈，用途為產生 3x3 遮罩所對應的位址，以及 5x5 遮罩中對應 P1~P8 位址，這個部份屬於共用的硬體資源；在 9~24 個時脈為產生 5x5 遮罩中剩餘的 P9~P24 的位址，對應時脈關係整理如圖 26 所示。可以根據圖 26 設計 MAG 的電路，MAG 的硬體電路如圖 27 所示。

圖中使用 8bit 的加減法器，用來產生+1，-1，+2，-2 的運算；利用 24 對 1 的多工器，將圖 25 中所對應的位址依序排列，透過 5bit 的 Counter 選擇所要產生的位址是 3x3 遮罩的內容或是 5x5 遮罩內容。

其中當 Mode Sel 為 0 時，5bit 的 Counter 會產生 0~7 的計數，也就是可選擇到 3x3 遮罩(P1~P8 的位址)，而當 Mode Sel 為 1 時，5bit 的 Counter 將產生 0~23 的計數，也就是可選擇到 5x5 遮罩(P1~P24 的位址)。

| 時脈個數 | 位址名稱 | 產生位址 | 時脈個數 | 位址名稱 | 產生位址 | 時脈個數 | 位址名稱 | 產生位址 |
|---|---|---|---|---|---|---|---|---|
| 1 | P1 | (i-1,j-1) | 9 | P9 | (i-2,j) | 17 | P17 | (i+2,j) |
| 2 | P2 | (i,j-1) | 10 | P10 | (i-2,j-1) | 18 | P18 | (i+2,j+1) |
| 3 | P3 | (i+1,j-1) | 11 | P11 | (i-2,j-2) | 19 | P19 | (i+2,j+2) |
| 4 | P4 | (i+1,j) | 12 | P12 | (i-1,j-2) | 20 | P20 | (i+1,j+2) |
| 5 | P5 | (i+1,j+1) | 13 | P13 | (i,j-2) | 21 | P21 | (i,j+2) |
| 6 | P6 | (i,j+1) | 14 | P14 | (i+1,j-2) | 22 | P22 | (i-1,j+2) |
| 7 | P7 | (i-1,j+1) | 15 | P15 | (i+2,j-2) | 23 | P23 | (i-2,j+2) |
| 8 | P8 | (i-1,j) | 16 | P16 | (i+2,j-1) | 24 | P24 | (i-2,j+1) |

圖 26 耗用時脈與其產生對應位址關係

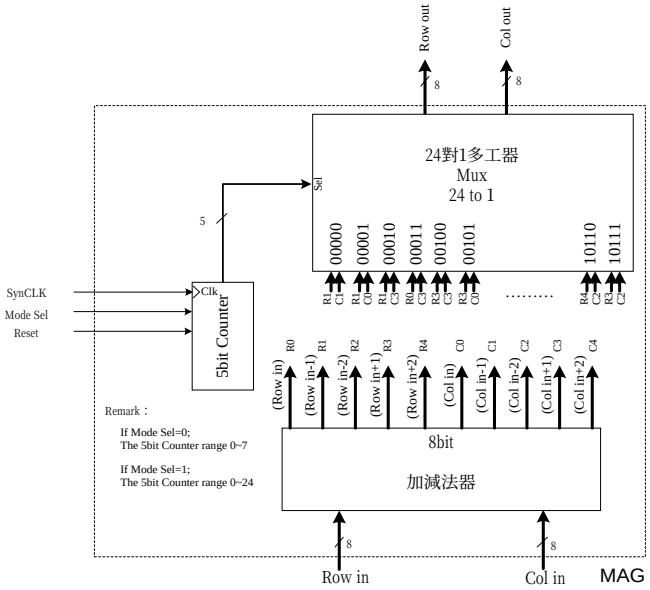


圖 27 AGU 硬體電路

## 第二節 指紋前處理模組單元

本模組單元分為兩個獨立子單元，即二值化單元(Binarization Unit)與細線化單元(Thinning)。這兩個子單元是指紋像進行前處理工作時，最浪費計算時間的部份，本論文提出有效的硬體化方法它們轉化為硬體模組，其中，細線化單元特別具獨創性[10]。

(一)二值化處理單元(Binarization Unit, BinU))

Binariztion Unit (BinU)為二值化單元，二值化的處理方法中，最常見的方式，就是選定一個固定的門檻值(threshold value)，當所判斷的像素灰階值大於等於其門檻值時，就令該像素為 255(白色)，反之，當所判斷的像素灰階值小於其門檻值時，就令該像素為 0(黑白)，如圖 28 所示。

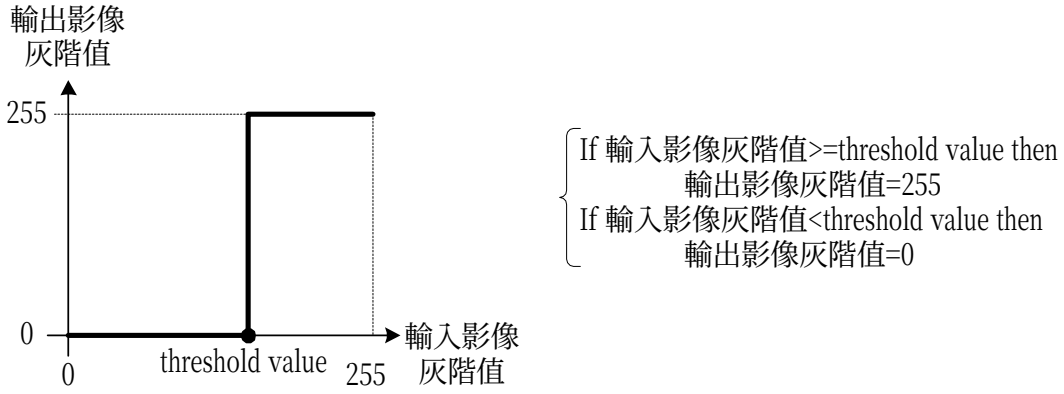



圖 28 常見二值化處理方法

若採用如圖 28 所示的方法，在處理的效果上，比較容易產生影像雜訊(noise)或將原有的影像資訊刪除，所以適應性比較差，所以本研究使用的二值化方法，就是利用 5x5 遮罩運算，動態產生 threshold value，這種做法的優點擁有較高的適應性，可以產生效果較好的二值化結果，其數學運算式如公式(6)所示。

$$\begin{cases} If\ \ P(i,j) \geq \dfrac{\sum_{\substack{r=-2\\ r\neq 0\ and\ s\neq 0}}^{\substack{r=2\\ s=2}} P(i+r, j+s)}{24} & then\ \ P(i,j) = 255 \\ If\ \ P(i,j) < \dfrac{\sum_{\substack{r=-2\\ r\neq 0\ and\ s\neq 0}}^{\substack{r=2\\ s=2}} P(i+r, j+s)}{24} & then\ \ P(i,j) = 0 \end{cases} \quad \cdots\cdots\cdots\cdots(6)$$

根據公式(6)可對二值化演算法進行其硬體化工作，其硬體化的方法如下所示：

對於 24 個灰度值的總和計算，需要 1 個累加器(accumulator)，若以最大灰度值來計算為 255 x 24 = 6120(十進制)，轉換為十六進制為：0x17E8，所以該累加器需要 13bit 寬；接著就是要處理除法的問題，所以需要一個除 24(十進制)的除法器，將所得平均與中心像素的灰度值做比較，故需要一個 8Bit 的比較器，硬體電路如圖 29 所示。

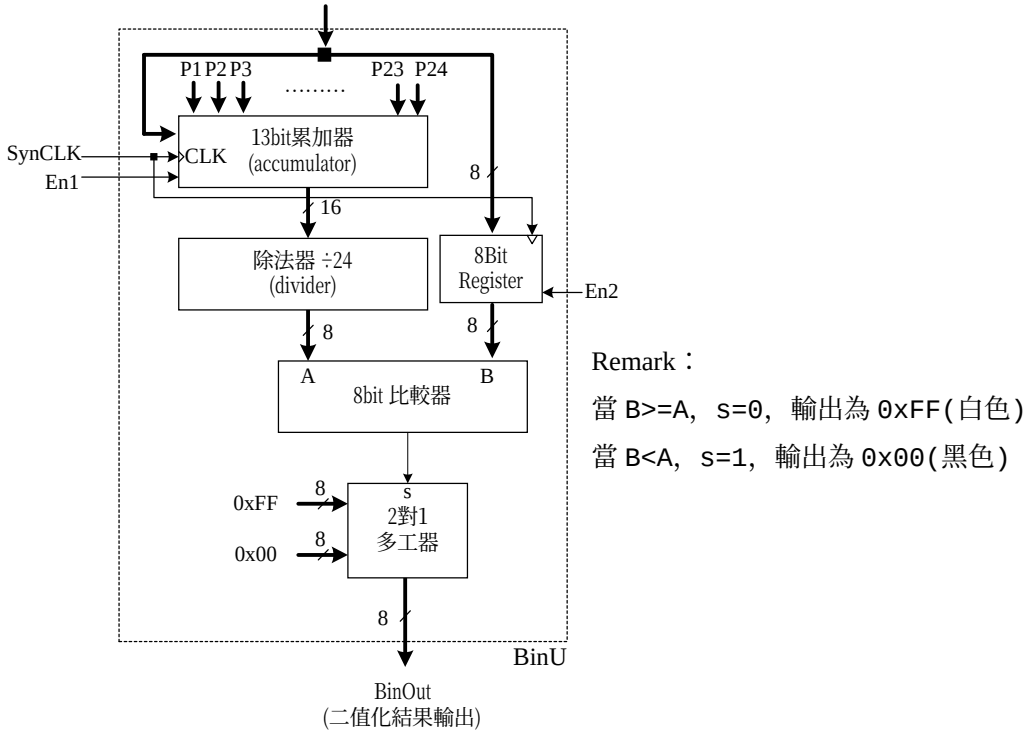



圖 29 BINU 硬體電路

根據圖 29 的硬體電路，使用 VHDL 將 BinU 硬體化，VHDL 硬體描述如圖 30 所示。

```
1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3 use IEEE.STD_LOGIC_ARITH.ALL;
4 use IEEE.STD_LOGIC_UNSIGNED.ALL;
5 use WORK.TestPack.ALL;
```

標準/自定函式庫的引用宣告

```
6
7 Entity BinU is
8       Port (
9                               DatIn  :        In      Std_logic_vector(DataWidth-1 downto 0);
10                              EC             :        In      Std_logic;
11                              CLK            :        In      Std_logic;
12                              Reset  :        In      Std_logic;
13                              BinRlt :        Out     Std_logic_vector(DataWidth-1 downto 0)
14                  );
15 End BinU;
16
```

BinU 的實體腳位宣告

```
17 architecture BinU _arch of BinU is
18
```

```
19 Signal ACCRlt                :       Std_logic_vector(12 downto 0);
20 Signal CompFlag              :       Std_logic;--Compare Flag
21 Signal CPXLx24               :       Std_logic_vector(12 downto 0);
22 Signal CNT                           :       Std_logic_vector(4 downto 0);
23 Signal CenterPXL             :       Std_logic_vector(DataWidth-1 downto 0);
24
25 Begin
26      Process(Reset,CLK)
27      Begin
28              If(Reset=Hi)Then
29                      ACCRlt<=(Others=>Lo);
30                      CNT<=(Others=>Lo);
31                      CenterPXL<=(Others=>Lo);
32              Elsif(CLK'Event)And(CLK=Hi)Then
33                      If(EC=Hi)Then
34                              CNT<=CNT+1;
35                              If(CNT=11)Then
36                                      CenterPXL<=DatIn;
37                              Else
38                                      ACCRlt<=ACCRlt+DatIn;
39                              End if;
40                      End If;
41              End if;
42      End process;
43 CPXLx24  <=      ("0"&CenterPXL&"0000")+("00"&CenterPXL&"000");
44 CompFlag  <=     Hi When(CPXLx24>=ACCRlt) Else
45                              Lo;
46 BinRlt                <=      x"00" When CompFlag=Lo Else
47                              x"FF";
48 End BinU_arch;
```

13bits 附有 Reset,CLK 以及 EC 的累加器(Accumulator)

13bits 的移位加法器

13bits 的比較器

二值化結果輸出

圖 30 BINU 硬體電路 VHDL 描述

(二)細線化單元(Thinning Unit, TU))

Thinning Unit，TU 為細線化單元，可選用的方法為 ZS algorithm 或是 LW algorithm，兩者的差異在於規則有些許不同，籍由圖 31 的 3x3 遮罩所對應的像素，分別將 ZS algorithm 與 LW algorithm 的規則分別如圖 32 與圖 33 所示。其中 A(P)為 P(i,j)周圍為 1 的個數，B(A)則為 P(i,j)周圍由 0 變化到 1 的次數。

| | | |
|---|---|---|
| P1 | P2 | P3 |
| P8 | P | P4 |
| P7 | P6 | P5 |

圖 31 3x3 遮罩所對應的像素

```
(1)Iteration 1 Rule：
    1)2<=A(P)<=6
    2)B(P)=1
    3)P2 x P4 x P6 =0
    4)P4 x P6 x P8 =0
(2) Iteration 2 Rule：
    1)’與 2)’同於(1)的 1)與 2)
    3)’ P2 x P4 x P8 =0
    4)’ P2 x P6 x P8 =0
```

圖 32 ZS ALGORITHM 細線化規則

```
(1)Iteration 1 Rule：
    1)3<=A(P)<=6
    2)B(P)=1
    3)P2 x P4 x P6 =0
    4)P4 x P6 x P8 =0
(2) Iteration 2 Rule：
    1)’與2)’同於(1)的1)與2)
    3)’ P2 x P4 x P8 =0
    4)’ P2 x P6 x P8 =0
```

圖 33 LW ALGORITHM 細線化規則

透過圖 31 即可進行將細線化演算法硬體化的動作。細線化電路硬體化採用的是查表法(Look-Up Table)的方式，使用圖 32 的細線化規則，轉換為查表方式的硬體設計上，共需使用兩個 256x1 bit 的記憶體單元，用以達成兩個迴圈(左上與右下)處理過程所需的運算。細線化硬體電路如圖 34 所示，其中將 P1~P8 分別接至為該記憶體的 A0~A7 位址。

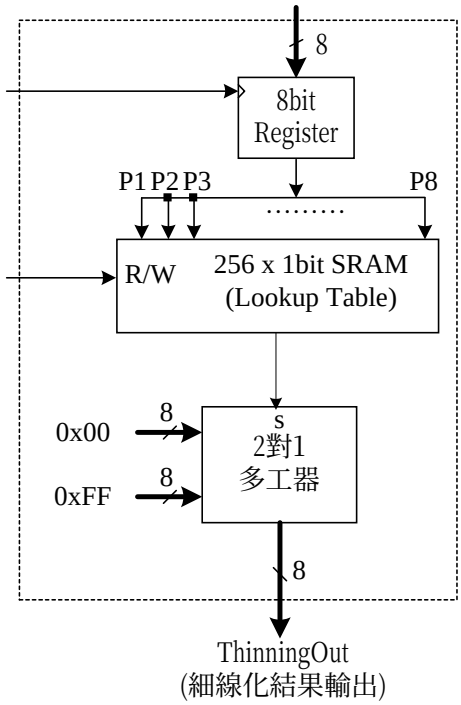



圖 34 TU 硬體電路

## 第三節 編碼運算單元

本運算單元分為兩個子單元，即端點找尋單元(Ending Point Unit)與分叉點找尋單元(Bifurcation Point Unit)。本論文提出的硬體化方法對這兩個單元採整合式設計處理如下。

(一)端點找尋單元(Ending Unit, EU)

Ending Unit(EU)為端點找尋處理單元，利用 3x3 遮罩來尋找端點，在 3x3 的所有情況中，屬於端點情況共有 16 種，如圖 35 所示：

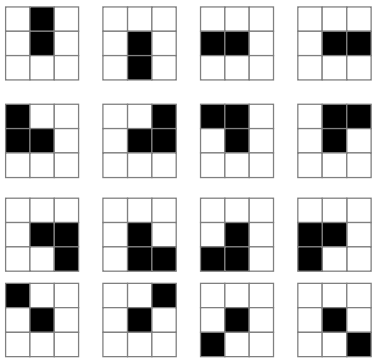


圖 35 3x3 端罩中屬於端點的 16 種情況

根據圖 35 所示，使用 3x3 遮罩排列運算，可獲得端點找尋的規則，並依端點找尋的規則使用查表法將之硬體化，端點找尋硬體化電路如圖 36 所示。

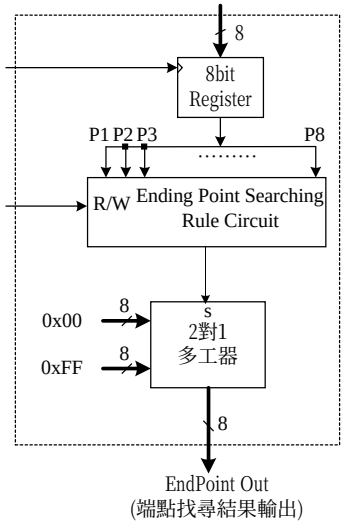



圖 36 EU 硬體電路

(二) 分叉點處理單元(Bifurcation Unit, BifU))

Bifurcation Unit(BifU)為分叉點找尋處理單元，利用 3x3 遮罩來尋找分叉點，在 3x3 的所有情況中，屬於分叉點的情況共有 37 種，如圖 37 所示：

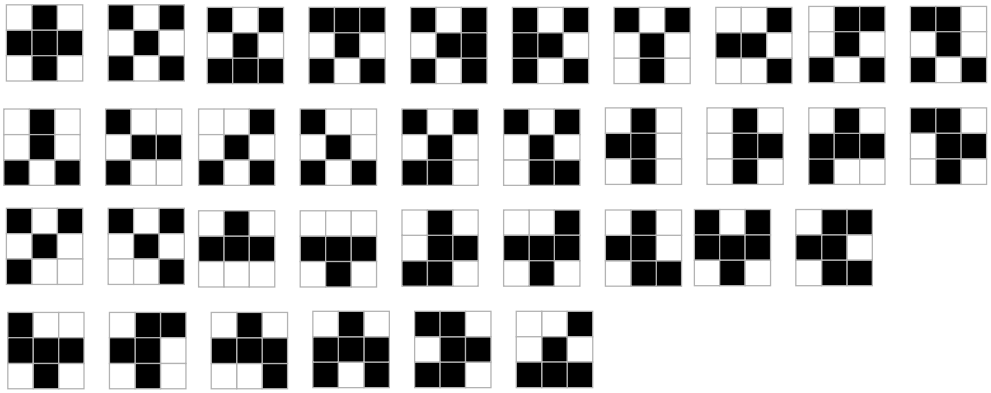


圖 37 以 3x3 端罩運算屬於分叉點的 35 種情況

根據圖 37 所示，使用 3x3 遮罩排列運算，可獲得分叉點找尋的規則，並依分叉點找尋的規則使用查表法將之硬體化，分叉點找尋硬體化電路如圖 38 所示

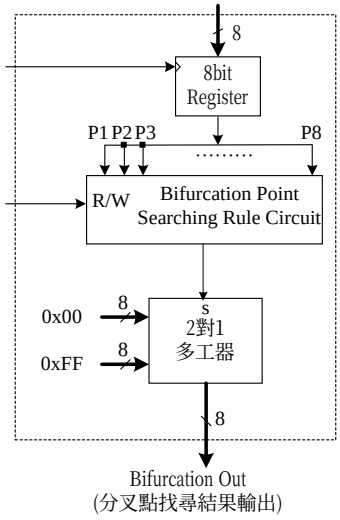


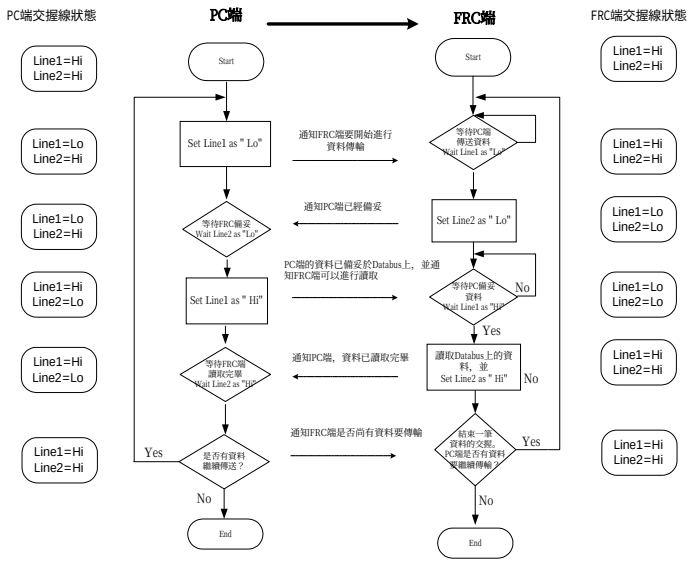

圖 38 BIFU 硬體電路

## 第四節 輸出入介面處理單元

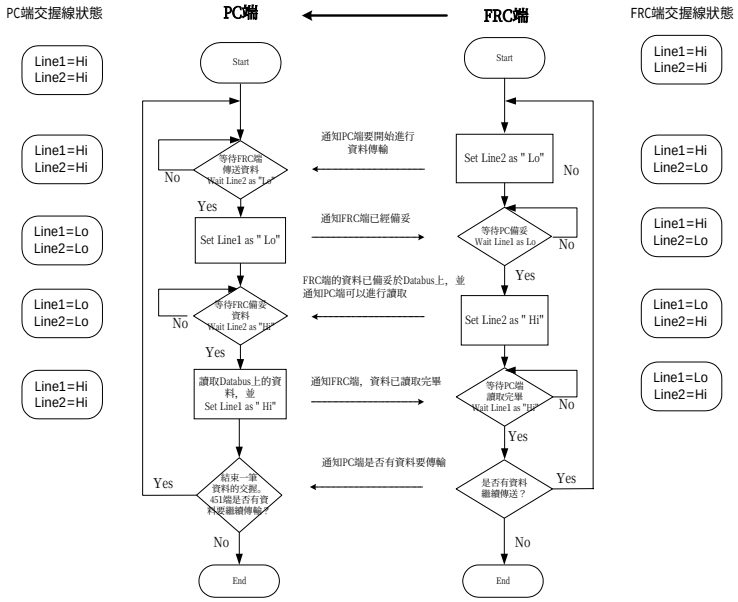
本單元細分為兩個子單元，即命令/資料交握傳輸控制器(SPP Controller, SPPCtrl)與外掛式記憶體控制器(SRAM Controller, SRAMCtrl)，前者負責與 PC 端與控制命令或影像資料傳輸交握的輸出入介面，後者則是負責外掛式記憶體控制信號、資料匯流排與位址匯流排實體訊號的硬體連線

(一)命令/資料交握傳輸控制器(SPP Controller, SPPCtrl)

SPP Controller 主要的功能就是 FRC 要從 PC 載入灰階資料資料，以及輸出運算結果至 PC 端，FRC 與 PC 間所使用的通訊協定就是採用標準並列埠通訊協定(Standard Parallel Protocol, SPP)，傳輸交握流程如圖 39(a)與(b)所示。其中，圖 39(a)為 PC 端傳送資料至 FRC 端的傳輸交握流程，圖 39(a)為 FRC 端傳送資料至 PC 端的傳輸交握流程。




(a) PC 端傳送命令/資料至 FRC 端




(b) FRC 端傳送影像資料至 PC 端

圖 39 PC/FRC 傳輸交握流程

依照圖 39 可以設計命令/資料交握傳輸控制器硬體電路，硬體電路如圖 40 所示。

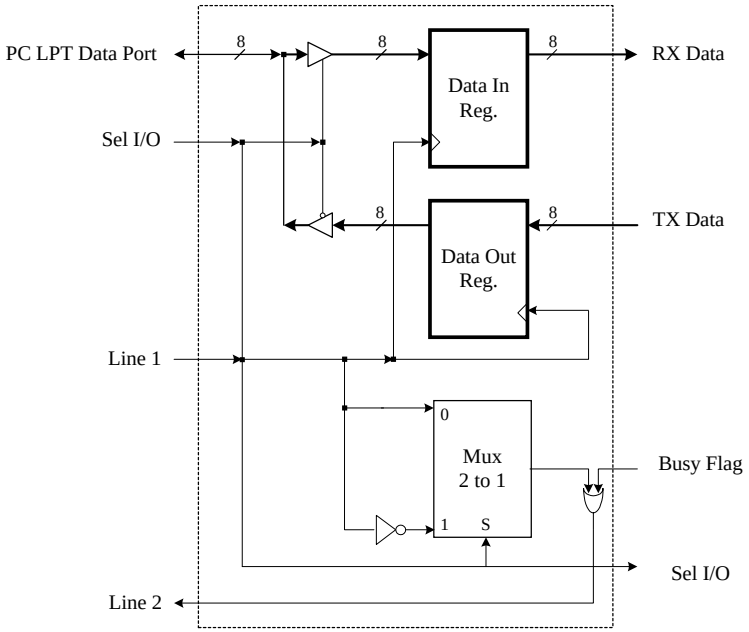



圖 40 SPPCTRL 硬體電路

(二)外掛式記憶體控制器(SRAM Controller, SRAMCtrl)

SRAM Controller 主要就是 FRC 實際對外掛式記憶體存取的控制器，外掛式記憶體可分為兩個模式(1)讀取模式及(2)寫入模式，詳細時序分別如圖 41 及圖 42 所示。對外掛式記憶體的存取，需特別注意訊號腳位轉態或保持的時間，以目前所選用的 BSI SRAM 而言，其延遲時間屬於 70ns，所以整個讀取或寫入週期必需大於 70ns(不可比 70ns 還小)，否則將造成資料讀出或寫入的錯誤。

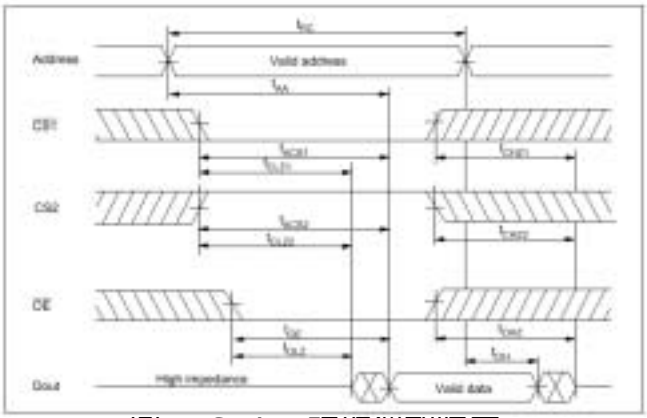


圖 41 SRAM 讀取週期時序

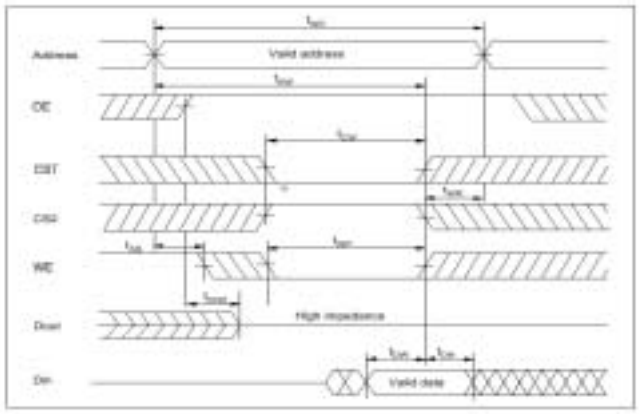


圖 42 SRAM 寫入週期時序

依照圖 41 及圖 42 可以發現/CS1 及 CS2 的準位都是固定的訊號，/CS1=低準位，CS2=高準位，而/WE 與/OE 剛好都是反向，所以可以將控制減少 1 支控制接腳，外掛式記憶體控制器硬體電路如圖 43 所示。

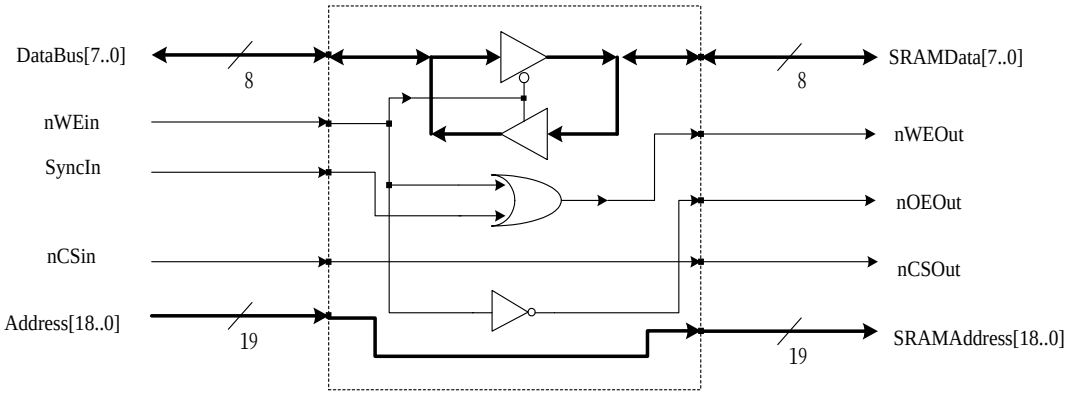



圖 43 SRAMCTRL 硬體電路