dsPIC Advanced Programming

Advanced Hands-on Programming with the dsPIC30F

課程目標 - 1

- 熟悉 Timer 的用法以及如何使用 MPLAB Language Tools Library 對 Timer 做初始化
- 學習 12-bit & 10-bit AD Module 的使用與設定 , 以及 10-bit ADC 的同步取樣/保持電路
- 學習 Motor Control PWM 模組的各項功能並用來產生 3 相 , AC 60 Hz 的前級驅動訊號
- 學習如何使用 PWM Override 的功能產生驅動 BLDC 的信號

Part - 1

General Purpose Timers

Timers / Counters Overview

- dsPIC 最多具有 5 個 16-bit General Purpose Timers / Counters (並非每一個 dsPIC 都具有 5 個)
  - 這 5 個 Timers 的功能都非常相近, 僅對外部 clock 的同步以及串接的設定上有些微差異
- 每個 Timer 有獨立的週期暫存器(Period Registers)
  - TMRx 和 PRx match 時可以中斷 CPU
  - TMRx 和 PRx match 時 ，TMRx 會自動被清除
- 每個 Timer 都可做 Gated Timer 的操作
  - GATE 信號的負緣可對 CPU 產生中斷
- 有 4 個 Timer 可以兩兩被串接成 32 bit 的Timers 或 counters
  - Timer 2,3串接 或 Timer 4,5串接

TIMERx 的控制暫存器 TxCON (x = 1..5)

- TON：用來啓用 Timer/Counter 的功能
- TCS：選擇 clock 的來源爲內部或外部

  TCS=0：Clock 來源爲內部； TCS=1：Clock 來源爲外部
- 輸入到 Timerx 的 clock 可以被做以下的預除 -> 1,8,64,256

  TCKPS<1:0> - Timerx 輸入 clock 預除量的設定
- 各 Timer 也可使用外部的 GATE 信號來控制是否要將內部的 clock 送至 TMRx

  TGATE=1：啓用 Gated time accumulation mode

TxCON Register

TON	-	TSIDL	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

-	TGATE	TCKPS<1:0>		-	-	TCS	-
bit7	6	5	4	3	2	1	bit0

TimerX 對外部 Clock 的處置方式

- 各 Timers 對外部 clock 的處置方式有所不同
  - TIMER1 可以選擇外部 clock 不與 CPU 時序同步而操作成 “asynchronous counter”
    - 條件：外部 clock 的頻率需 < 25MHz
  - TIMER2 and TIMER4 對外部 clock 的同步發生在 prescalar 之後
    - 外部 clock 的頻率需 < (prescale * 1/2 Fcy)
  - TIMER3 and TIMER5 對外部 clock 的同步發生在輸入端
    - 外部 clock 的頻率需 < 1/2 Fcy

Timer1 在非同步模式 (Asynchronous) 下的操作

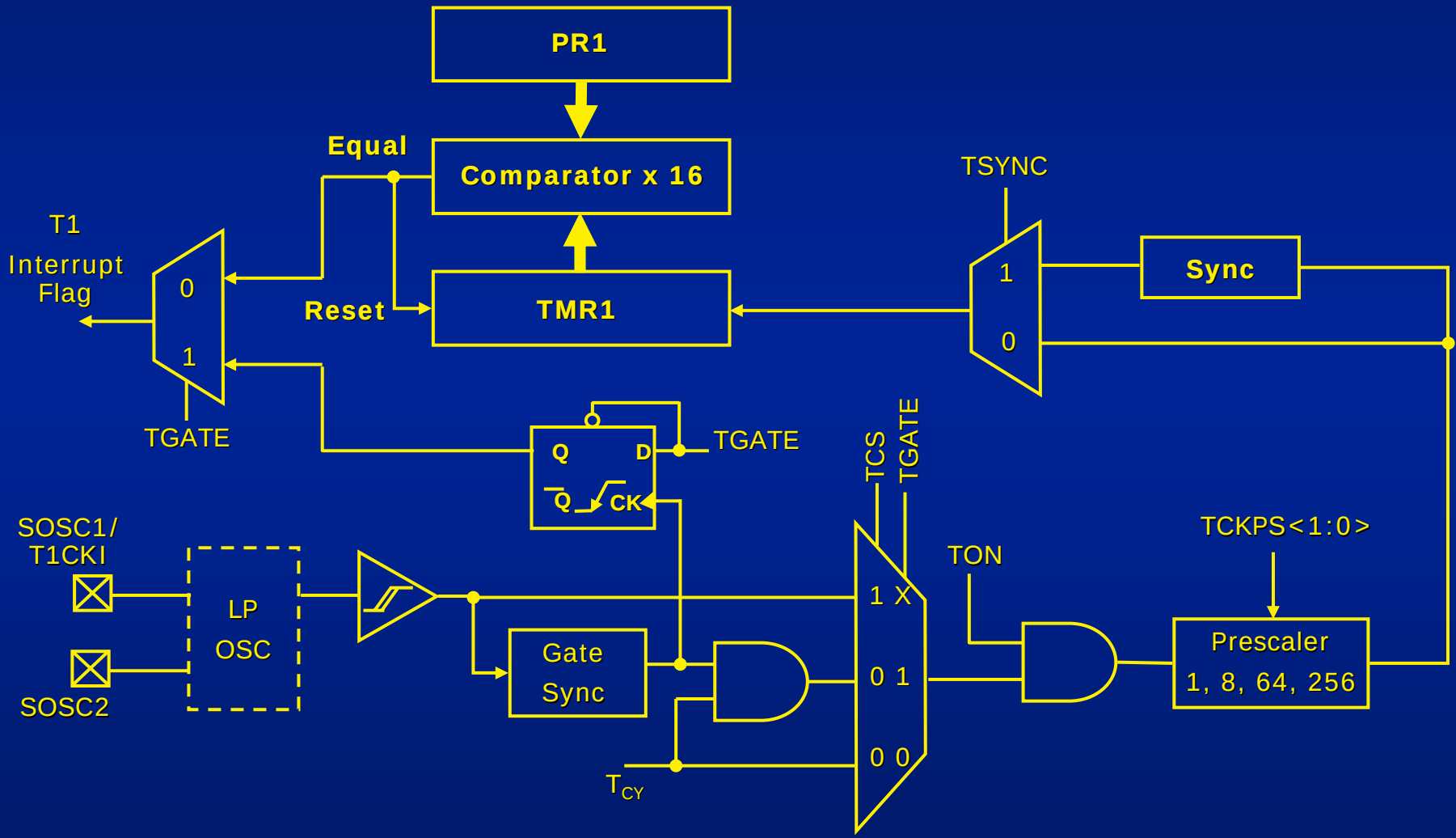
- Timer1 可以對非同步的外部 clock 做計數的工作 (counter)
- Timer1 可以接受頻率超過 Fcy 的外部頻率
- Timer1 在 CPU 處於 SLEEP 狀態時仍可以正常的接收外部 clock，進行 Counter 的動作
- Timer1 + LP Oscillator 可以做成 Real-Time Clock 的功能
  - Low-power operation (約7.5us @ 5V)
  - External 32KHz crystal oscillator
  - LP Oscillator can serve as system clock

T1CON Register

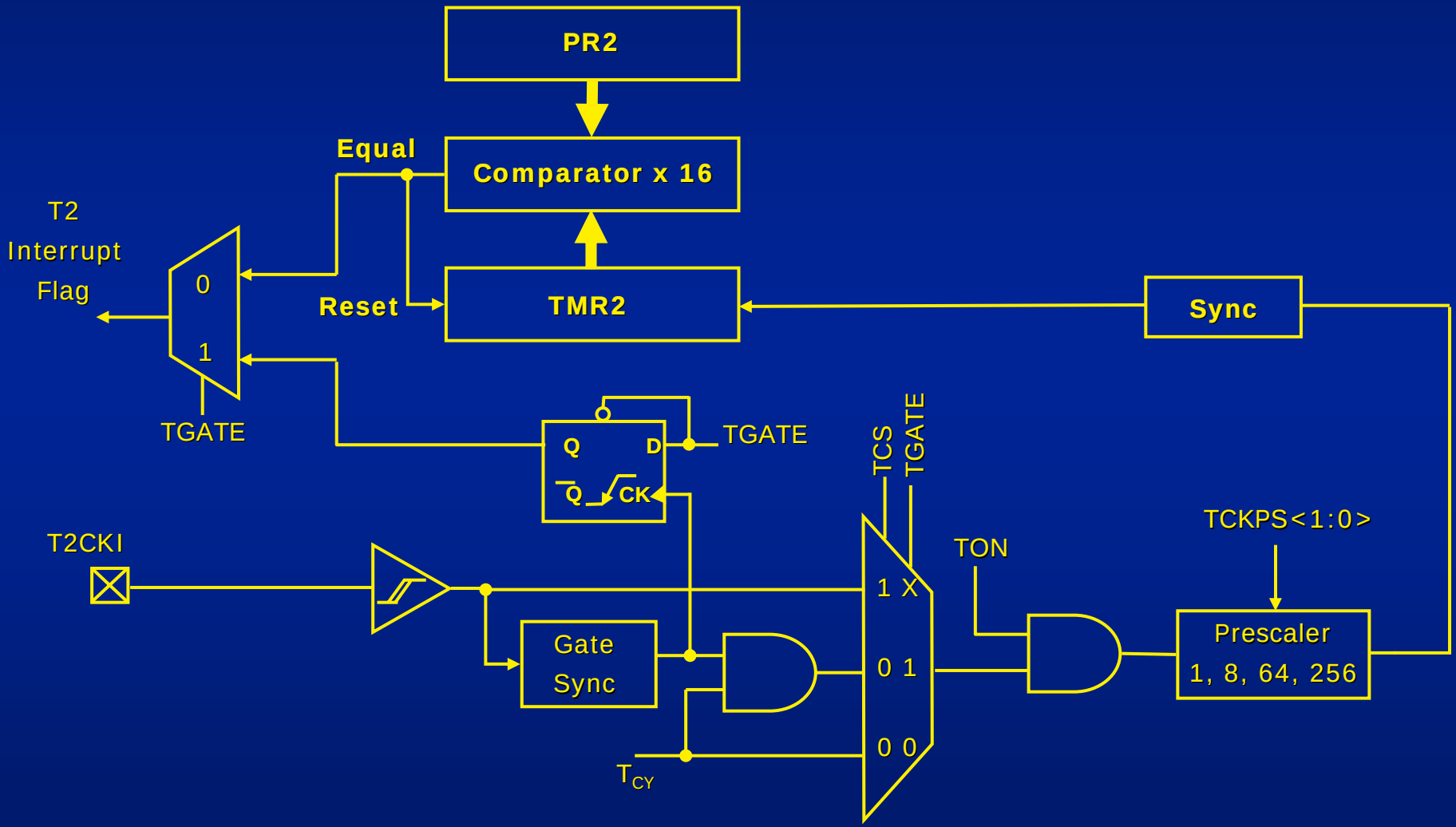
TON	-	TSIDL	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

-	TGATE	TCKPS<1:0>		-	TSYNC	TCS	-
bit7	6	5	4	3	2	1	bit0

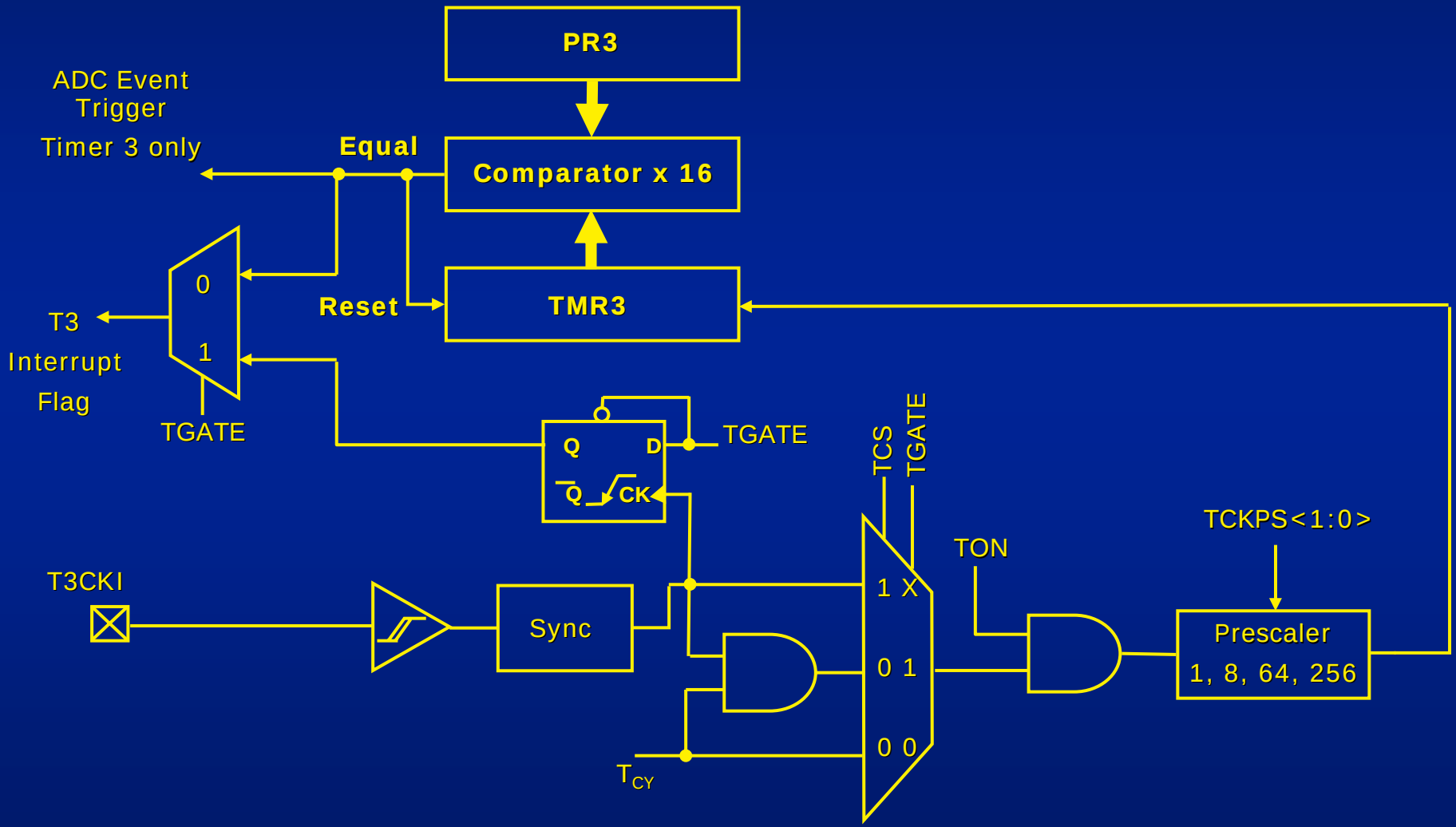
Timer 1 的方塊圖



Timer 2 及 Timer 4 方塊圖



Timer 3 和 Timer 5 方塊圖



將兩個 Timer 串接成 32-bit 的操作模式

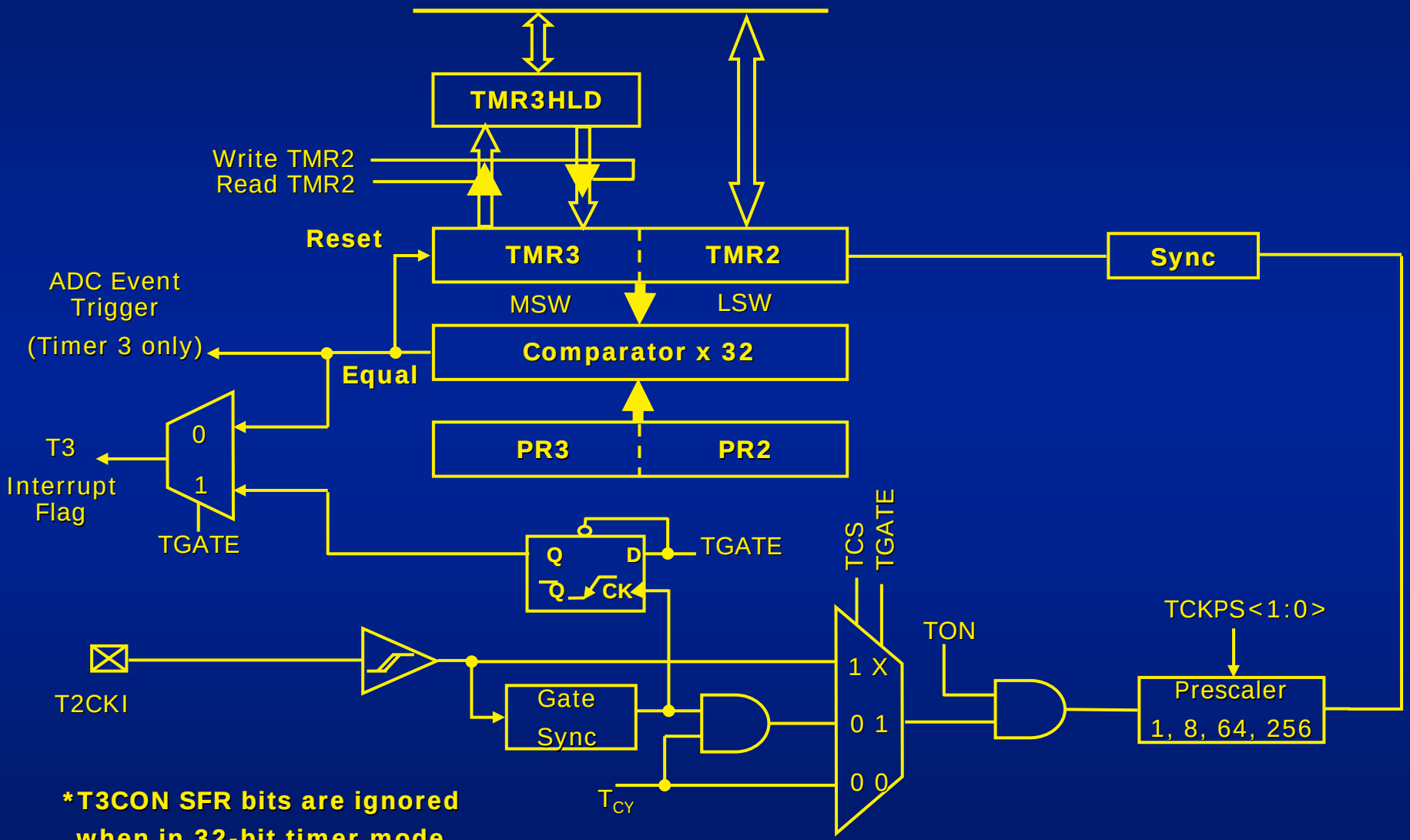
- Timer3/Timer2 可串接成 32-bit Timer
- Timer5/Timer4 可串接成 32-bit Timer
- 在此模式下, 有一個 16-bit 的寫入/讀取的 Latch Buffer 來完成 32-bit 的讀寫 (dsPIC 的 Data bus 寬度只有 16-bit)
- T32=1 : 32-bit Timer 的選擇位元
- 若將 Timer3/Timer2 串接，串接成的 32-bit Timer/Counter 的功能使用 Timer2 的相對應控制位元控制. Timer5/Timer4 則用 Timer4

T2CON Register

TON	-	TSIDL	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

-	TGATE	TCKPS<1:0>		T32	-	TCS	-
bit7	6	5	4	3	2	1	bit0

32-bit Timer 時的方塊圖



***T3CON SFR bits are ignored when in 32-bit timer mode**

Timers 的其他特殊用途

- Timer2 或 Timer3 被使用於 Capture 及 Compare 的 Time Base
- Timer3 可被用於觸發 ADC 的轉換動作
  - 在後續的練習中我們將練習如何使用 Timer 3 來觸發 AD 的轉換

練習 一
使用 Peripherals Library 來規劃 Timer1&3

- 用來設定及規劃 16 bit Timers 的 Peripheral Library
  - ConfigIntTimerx( )　　(x = 1 到 5)
  - OpenTimerx( )　　(x = 1 到 5)
  - 以上是將各 Timer 規劃成 16 bit Timer / Counter 時的函數
- 若是要設定串接的 32 bit Timers，則需要以下的函數
  - ConfigIntTimer23( ) or ConfigIntTimer45( )
  - OpenTimer23( ) or OpenTimer45( )

練習 一
使用 Peripherals Library 來規劃 Timer1&3

- ConfigIntTimerx( ) 可選用的參數
  - 以下兩種參數可使用 & 符號來連接
  - A. 中斷優先權的設定
    - Tx_INT_PRIOR_0　　(x = 1 到 5)
    - Tx_INT_PRIOR_1
    - Tx_INT_PRIOR_2
    - Tx_INT_PRIOR_3
    - Tx_INT_PRIOR_4
    - Tx_INT_PRIOR_5
    - TX_INT_PRIOR_6
    - TX_INT_PRIOR_7
  - B. 中斷致能或禁能的設定
    - Tx_INT_ON　　(x = 1 到 5)
    - Tx_INT_OFF

練習 一
使用 Peripherals Library 來規劃 Timer1&3

- OpenTimerx( ) 可選用的參數有兩組
  - 包括 config 及 period
  - config 的選項如下所列，各選項使用 & 連結
    - A. Timer Module ON or OFF
      - Tx_ON (x = 1 到 5)
      - Tx_OFF
    - B. Timer Module Idle ON or OFF
      - Tx_IDLE_CON (x = 1 到 5)
      - Tx_IDLE_STOP
    - C. 32-bit Timer Mode Disable (Only X = 2 or 4)
      - TX_32BIT_MODE_OFF
    - D. Timer Gate Time Accumulation Enable
      - Tx_GATE_ON
      - Tx_GATE_OFF

練習 一
使用 Peripherals Library 來規劃 Timer1&3

- config 的選項 (cont.)
  - E. Timer prescaler
    - Tx_PS_1_1 (x = 1 到 5)
    - Tx_PS_1_8
    - Tx_PS_1_64
    - Tx_PS_1_256 (Manual 有錯)
  - F. Timer synchronous clock enable
    - T1_SYNC_EXT_ON (x = 1 only)
    - T1_SYNC_EXT_OFF
  - G. Timer clock source
    - Tx_SOURCE_EXT
    - Tx_SOURCE_INT
- 第二組參數 period 則填入 0 to 65535 的值

練習 1
使用 Peripherals Library 來規劃 Timer1&3

- 練習一需要完成的程式功能
  - 規劃兩個 16-bit Timer，Timer1 & Timer3
  - 兩個 Timer 都使用 internal 的 Clock Source
  - Timer1 需設定爲 200 ms 中斷一次的 Timer
  - Timer3 需設定爲 1 ms 中斷一次的 Timer
  - Timer1 中斷時將 LED13 反向 (Toggle)
  - Timer3 中斷時設定旗號 Flags.T3OV
    - 主程式檢測 Flags.T3OV 並將 CounterT3 加一
    - 當 CounterT3 = 400 時將 CounterT3 歸零
    - CounterT3 被歸零的同時將 LED14 反向 (Toggle)

練習 一
使用 Peripherals Library 來規劃 Timer1&3

- 練習 一 需要注意的事項
  - 中斷發生的同時，相對應的中斷旗號會被設定爲 1
    - T1IF ,T3IF and .....
  - 大部分的中斷旗號必須以軟體清除，T1IF & T3IF 就必須清除
  - 注意 T3OV 旗號的設定與清除的時機
  - 注意 LED13 及 LED14 的正確位置 (已用 #define 宣告)
  - 要使用與周邊有關的函數，必須加入適當的 peripheral library 於 project 中
    - dsPIC30F4011 使用 libp30f4011.a
  - CPU 的 Oscillator 爲 7.3728 Mhz
  - 若 oscillator mode 設爲 XT w/PLL 8* ，則 Tcy = 1/(7372800*2) Second . (Fcy = 7372800 * 2)
    - dsPIC 的一個 Tcy 需 4 個 clock

第二單元

A/D 轉換器

10- 或 12-bit

高速，多輸入通道

自動掃描，交互輸入

A/D 轉換器

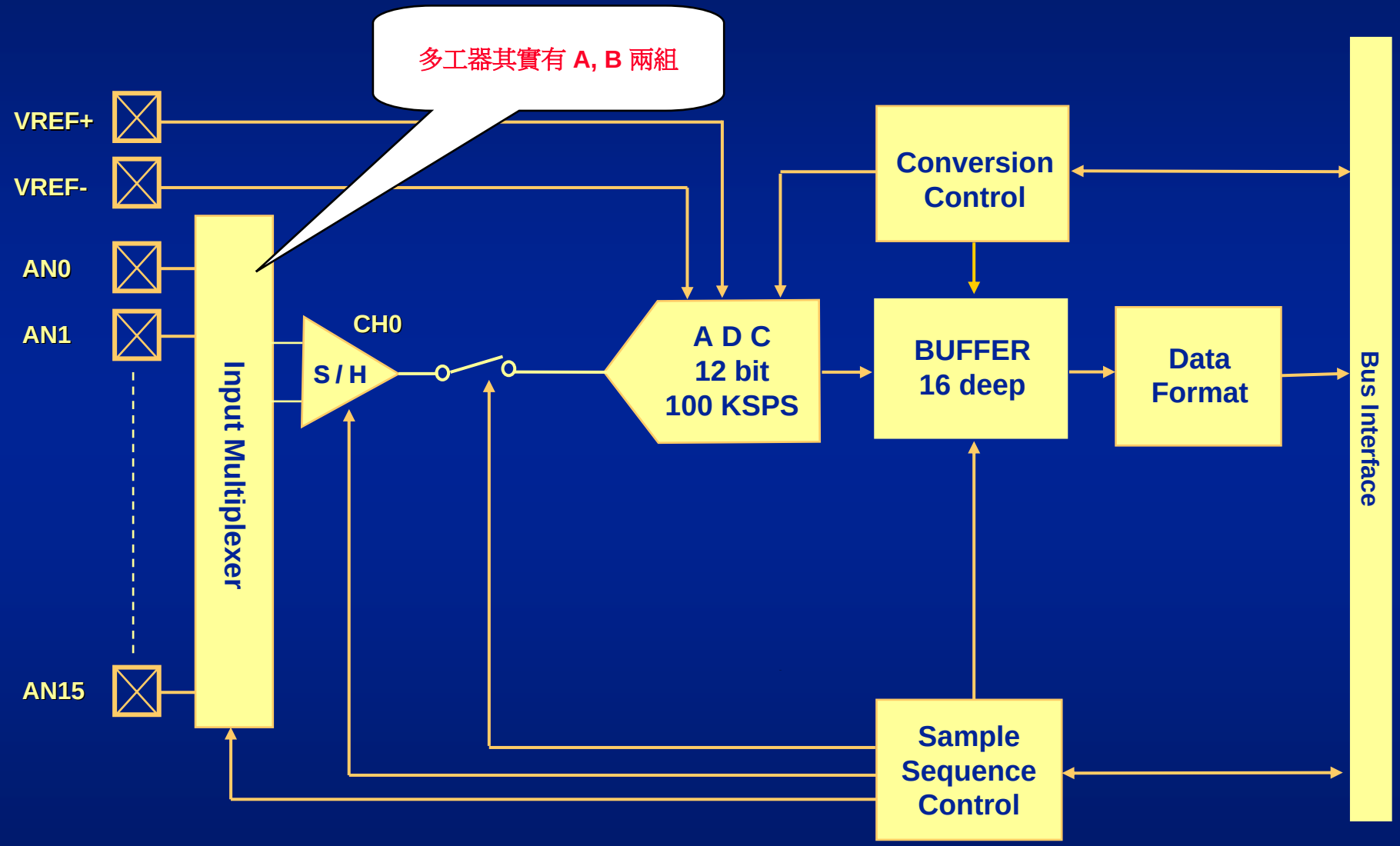
- 10-bit A/D
  - 10-bit 解析度，準確度 +/- 1 LSB
  - 500Ksps 轉換速率
  - 最多可有16 個類比輸入，4 個同步取樣電路

- 12-bit A/D
  - 12-bit解析度，準確度 +/- 1 LSB
  - 100Ksps 轉換速率
  - 最多可有16 個類比輸入，1個同步取樣電路

A/D 轉換器主要功能

- SAR 連續近似演算法
- 最多可有16 個類比輸入
- 可使用外部參考電壓源 VREF+，VREF-
- 單極性差動式輸入方式
- 可程式控制連續自動取樣
- 16 個轉換資料儲存區，可分爲兩組
- 多重觸發轉換模式
- 多種轉換排程的選項

12-bit A/D 轉換器



A/D 轉換器
取樣 / 轉換 的動作

A/D轉換
的結果是
這裡的值

類比
輸入

取樣
時間

轉換
時間

開始
取樣

取樣結束
開始轉換

轉換
結束

A/D 轉換器
ADPCFG 暫存器

定義：

類比輸入腳位設定

ADPCFG 暫存器

ADPCFG

每一類比輸入腳位都可以單獨設定

PCFG<15:0> - 選擇類比輸入腳位

- PCFGx = 1 –> 數位輸出入功能
- PCFGx = 0 –> 類比輸入功能

- 零件編號不同，類比輸入的接腳數也會不一樣
- 重置後**(Reset)**，預設腳位的功能爲類比輸入
- 每一類比輸入腳位都可以單獨設定
- 設成類比輸入腳以後，**I/O** 讀取的結果爲 **0**

A/D 轉換器
ADCON1 暫存器

定義：

取樣模式 & 轉換觸發模式 & 資料輸出格式

ADCON1 暫存器 (MSB)

ADCON1

ADON – 設爲 1 時，將 ADC 轉換功能致能

ADSIDL – 設爲 1 時，ADC 在 IDLE 模式下會關閉

FORM<1:0> - AD轉換後資料的輸出格式選擇

- 11 = 有號小數 (Signed fractional)
- 10 = 無號小數 (Fractional)
- 01 = 有號整數 (Signed integer)
- 00 = 整數 (Integer)

ADC 輸出格式
16-bit

	b15 b14 ←----------→ b1 b0
有號小數	$\overline{d11}$,d10,d09,d08,d07,d06,d05,d04,d03,d02,d01,d00, 0 , 0 , 0 , 0
小數	d11,d10,d09,d08,d07,d06,d05,d04,d03,d02,d01,d00, 0 , 0 , 0 , 0
有號整數	$\overline{d11}$,$\overline{d11}$,$\overline{d11}$,$\overline{d11}$,$\overline{d11}$,d10,d09,d08,d07,d06,d05,d04,d03,d02,d01,d00
整數	0 , 0 , 0 , 0 ,d11,d10,d09,d08,d07,d06,d05,d04,d03,d02,d01,d00

ADCON1 暫存器 (LSB)

ADCON1

SSRC<2:0> - 啓動AD轉換的觸發信號來源選擇

111 = 使用內部時序設定取樣時間及轉換時間(自動轉換)

(需參考到 ADCON3 暫存器的設定)

011 = 馬達控制 PWM 間隔結束時，結束取樣ADC開始轉換

(需參考到 SEVTCMP 暫存器的設定) 馬達專用的dsPIC才有此功能

010 = Timer 3 計時比較完成後，結束取樣ADC開始轉換

001 = INT0 腳位電位轉態時，結束取樣ADC開始轉換

(需參考到 INTCON2 <INT0EP> 位元的設定)

000 = 手動轉換， SAMP=1 時取樣，清除 SAMP 位元後轉換

ADCON1 暫存器 (LSB)

ADCON1

ASAM :

- 設爲 1 時，當上次AD轉完後成後，立即自動取樣 (SAMP位元會自動設爲 1)，
- 設爲 0 時，採手動取樣模式，將 SAMP位元設定爲 1 時取樣 .

SAMP :

- SAMP 寫入 1 時就開始取樣； SAMP 清爲 0 後，採樣工作結束，然後 AD 開始轉換 (當DONE=1或ADIF=1 時表示轉換完成)
- 當 SSRC 不爲 000 時，所指定到的 Trigger Source 會設定 SAMP = 1，並在取樣結束後自動的將 SAMP 清爲 “0”，並進行 AD 轉換

ADCON1 暫存器 (LSB)

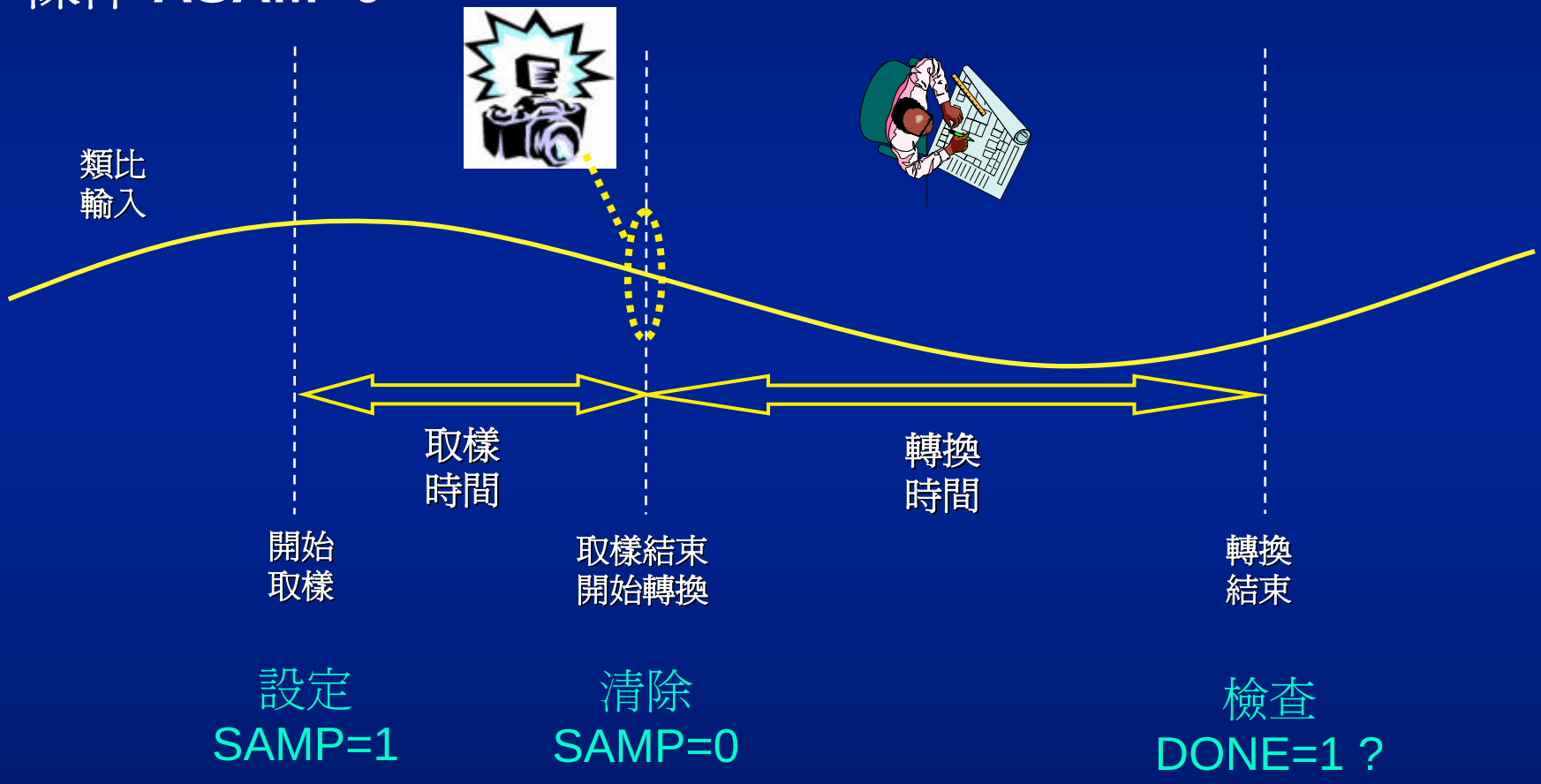
ADCON1

DONE：AD 轉換狀態指示位元

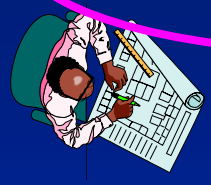
等於 1 時，A/D 轉換完成，該位元可以用軟體清除或新的轉換動作啓動時也會被清爲 0

A/D 轉換器
手動取樣 / 轉換步驟

條件 **ASAM=0**



A/D 轉換器
自動取樣 & Timer3 轉換



類比
輸入

取樣時間 | 轉換時間 | 取樣時間 | 轉換時間

Timer3 計時比較
完成後自動清除
SAMP開始轉換

上一次轉換完成後
硬體自動取樣
(SAMP 被設爲 1)
因 ASAM=1

Timer3 計時比較
完成後自動清除
SAMP開始轉換

ADCON1

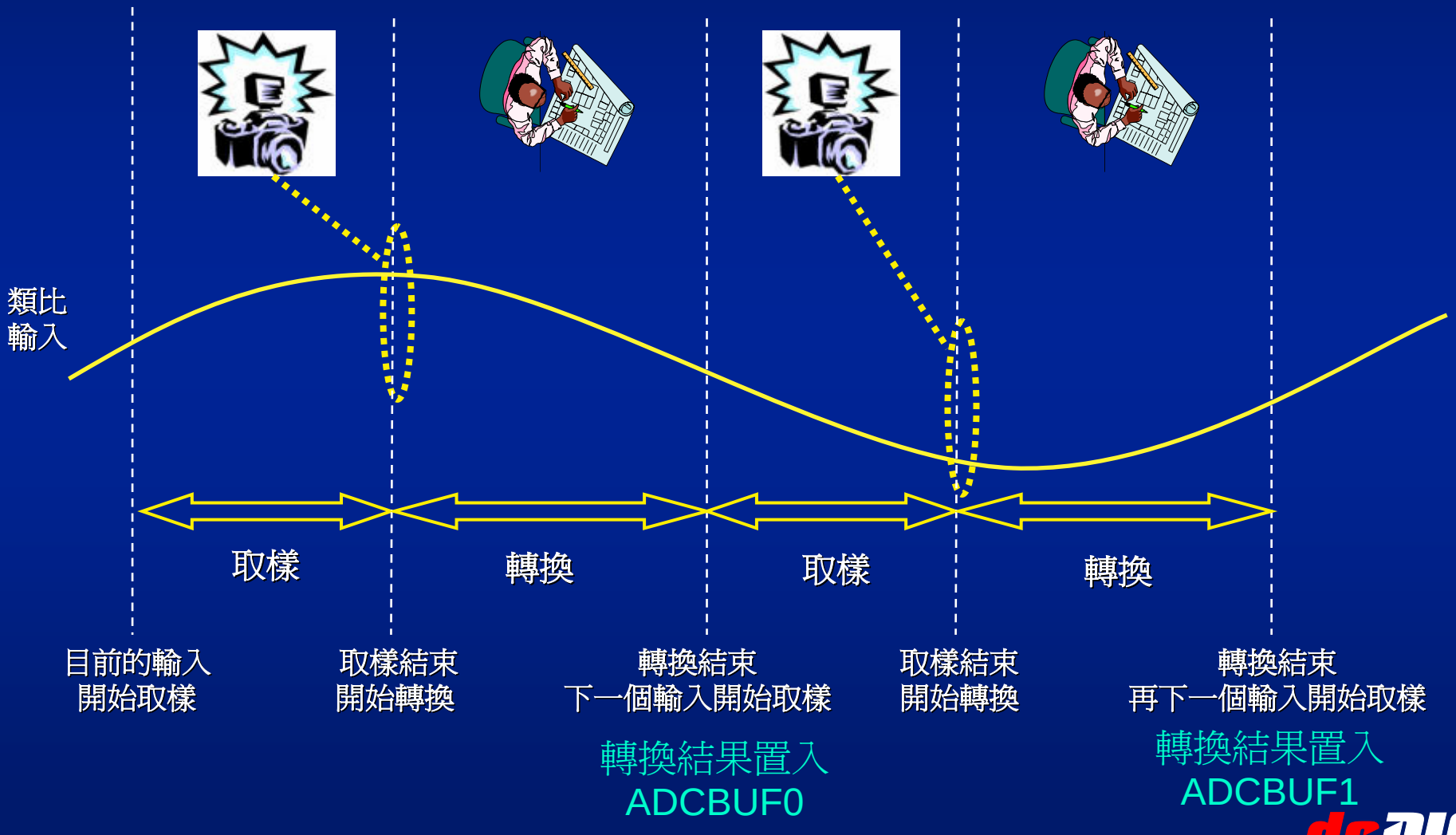
ADON=1	-	ADSIDL	-	-	-	FORM<1:0>

Bit 0

SSRC=010	-	SIMSAM	ASAM=1	SAMP	DONE

A/D 轉換器

多個類比輸入時的轉換方式



A/D 轉換器
ADCON2 暫存器

定義：

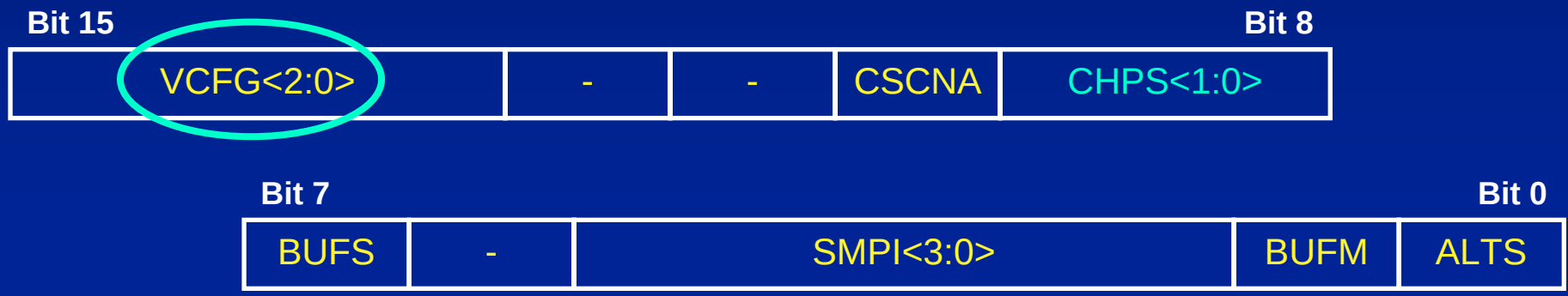
參考電壓

資料輸出緩衝器

多工器 & 輸入掃描

ADCON2 暫存器

ADCON2



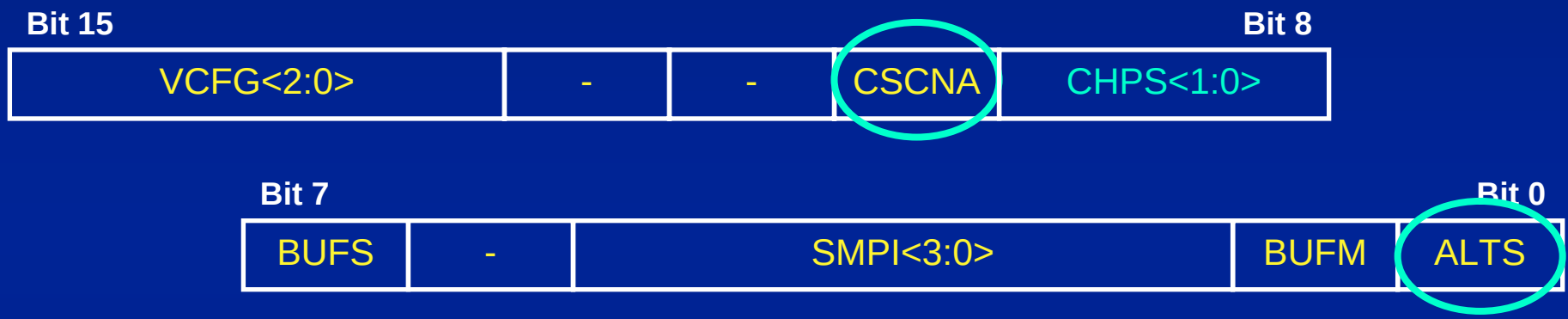
VCFG<2:0> - 參考電壓源的選擇

- 000 – AVDD ， AVSS
- 001 – 外接 VREF+ ， AVSS
- 010 – AVDD ， 外接 VREF-
- 011 – 外接 VREF+ ， 外接 VREF-
- 1xx – AVDD ， AVSS

轉換的輸入電壓範圍被限制在參考電壓範圍之間，在此範圍之外**(12-bit)**的電壓會以最大值**(0xFFF)** 或以最小值**(0x000)** 表示

ADCON2 暫存器

ADCON2



CSCNA : =1時，採用自動掃描方式自A組多工器輸入

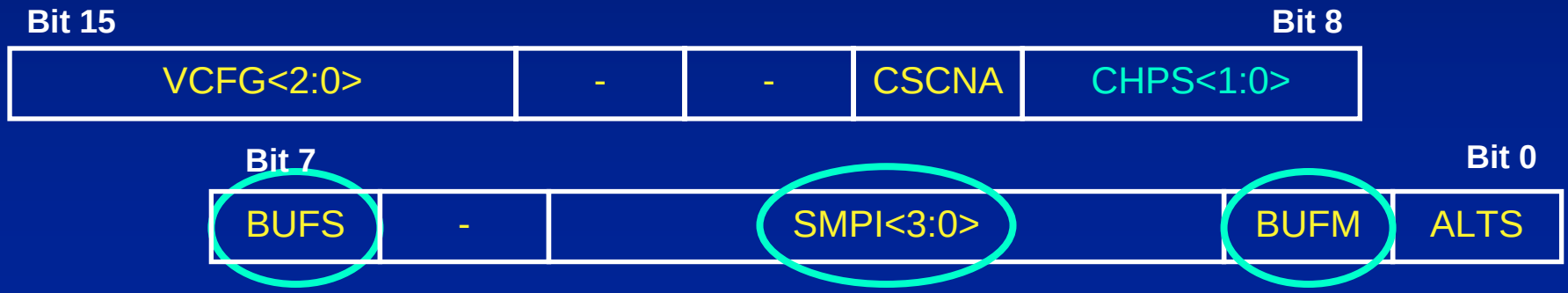
ALTS : = 1時，先選 A 組輸入再選 B 組輸入相互交換著選擇多工器 (A→B→A→B→A→B ….)

= 0時，輸入只選 A 組多工器

關於此項功能會在後面詳細說明使用方式

ADCON2 暫存器

ADCON2



SMPI<3:0> - 設定 AD 要轉換幾次後才產生一次中斷
(這些轉換後的資料會被存到 ADCBUFx 的暫存器列裡)

BUFM – ADCBUFx 暫存器列設定成單組或兩組模式

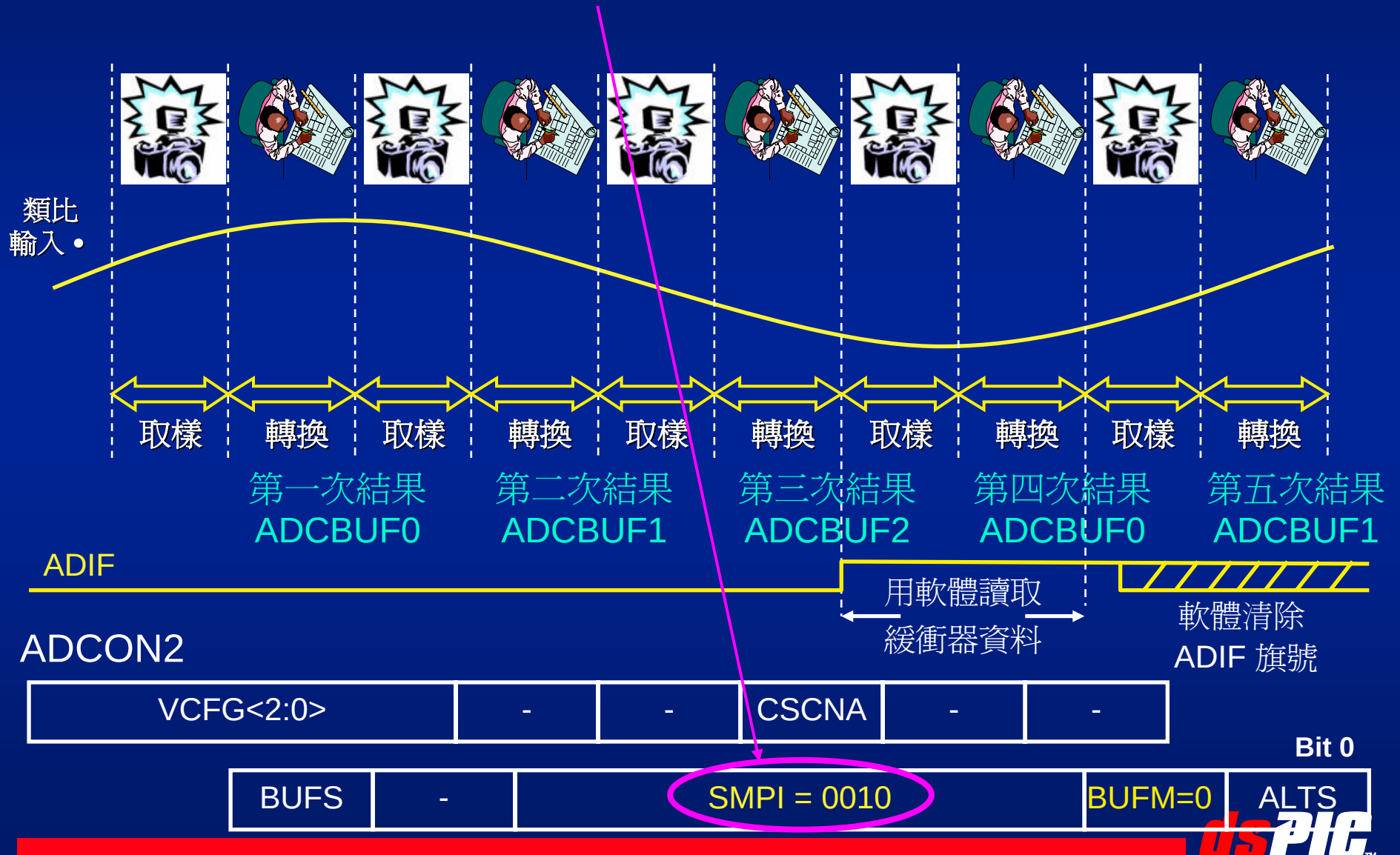
BUFS – ADCBUFx 採兩組模式時的狀態指示位元

轉換結果儲存規則

- AD 轉換的結果儲存到哪裡?
  - 當 BUFM 位元 = 0
    - 結果存入單一組16個緩衝器裡 - ADCBUF0,1,2...E,F
    - 每次 ADIF 中斷產生後，指標歸零指向 ADCBUF0
    - 每次 ADIF 中斷產生後，新的轉換資料會蓋掉上一次的 存在 ADCBUF0 的資料
    - 考慮中斷發生之後，軟體是否可在資料被下一輪的轉換結果覆蓋前對這一輪的資料做處置！
  - 當 BUFM 位元 = 1
    - 結果會分組自動存入兩組的8個緩衝器裡
    - BUFS 位元會指出目前 AD 轉換使用那組緩衝器
    - BUFS = 1, AD 目前填入第二組緩衝器 ADCBUF8 - F
    - 當使用此模式時，每一輪的轉換最多可在儲存 8 個資料後中斷.
    - 雖然單次可儲存資料量變少, 但 CPU 有更多時間間隔可以利用

A/D 轉換器

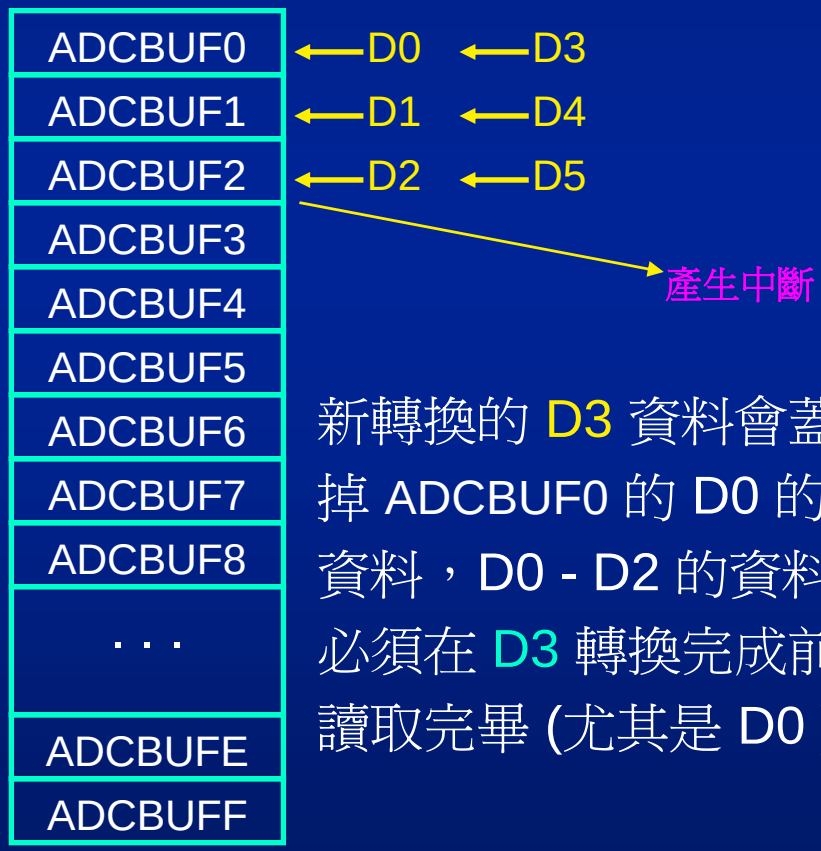
轉換三次後產生中斷的說明



A/D 轉換器
轉換儲存模式 & 轉換次數與中斷

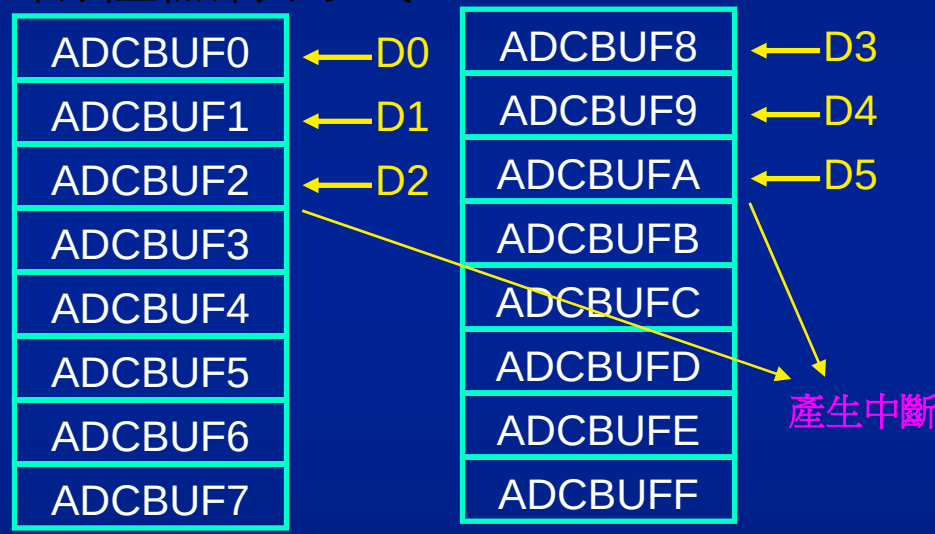
設定爲三次轉換後產生一次中斷 - **SMPI = 0010**

單組儲存方式 - BUFM = 0



新轉換的 D3 資料會蓋掉 ADCBUF0 的 D0 的資料，D0 - D2 的資料必須在 D3 轉換完成前讀取完畢 (尤其是 D0)

兩組儲存方式 - BUFM = 1



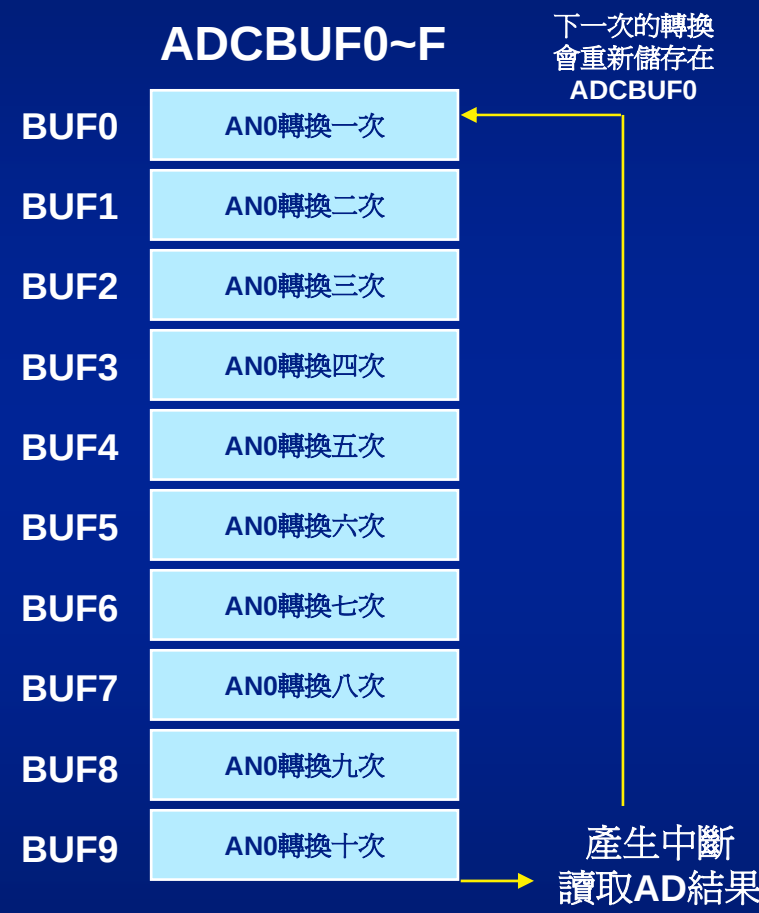
D6 的資料會蓋定 ADCBUF0 的 D0，所以 D0 - D2 的資料必須在 D6 轉換完成前讀取完畢

AD 中斷設定

設定需求：

AN0 爲輸入腳，轉換 10 次後產生一次中斷

- SMPI<2:0>=1001，轉換10次後中斷一次
- BUFM=0，採用單一 buffer (16-word)
- ALTS=0，只使用 MUX A 爲輸入
- CH0SA<3:0>=0000，AN0 for CH0+ 輸入
- CHONA=0， Vref- for CH0- 輸入
- CSCNA=0，輸入掃描禁能(Disable)
- CSSL<15:0>= N/A
- CH0SB<3:0>= N/A
- CH0NB=N/A



A/D 轉換器
ADCON3 暫存器

定義：

自動取樣時間

轉換時脈設定

ADCON3 暫存器

ADCON3

SAMC<4:0>：設定自動轉換時的取樣時間 (0 T_{AD} to 31 T_{AD})

ADRC : =1時，使用內建RC振盪時脈；=0時，使用系統時脈

ADCS<5:0> - AD 轉換時脈的時間 (T_{AD})

111111= (Tcy / 2) (ADCS<5:0>+1) = 32 Tcy

……

000000= Tcy/2

自動取樣轉換的時間計算

- 必需先設定爲自動取樣與轉換模式
  - 假設石英晶體用7.3728MHz，並使用16 倍的倍頻電路使工作頻率爲 117.9648MHz = 8.477nS (Fosc)
  - Tcy = (1 / Fosc) x 4 = 33.91nS
- 在規格書裡有規定
  - 最小的 TAD 需大於 713nS，最小的 轉換時間=10uS
  - 若 ADCON3 = 0x1F3F (SAMC=11111, ADCS=111111)

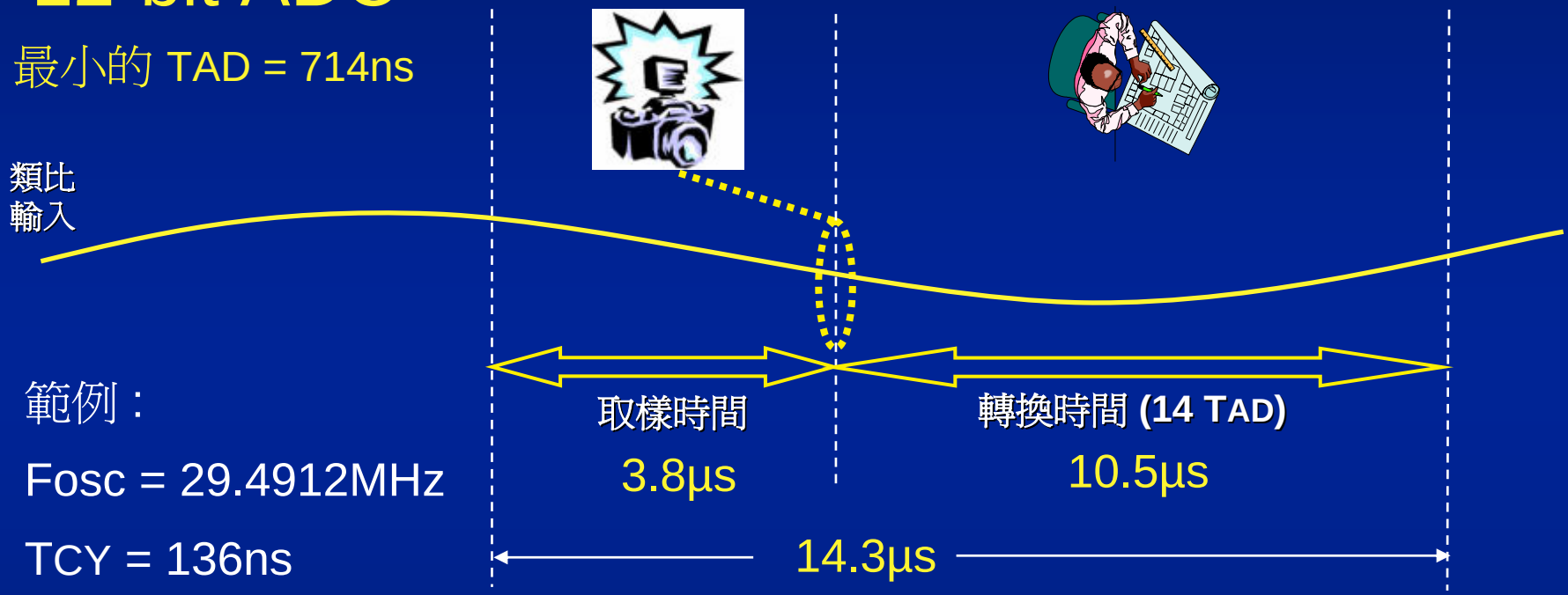
Tad = (ADCS<5:0>+1) Tcy / 2 = (63+1) Tcy / 2= Tcy * 32 = 33.91nS x 32 = 1.08512uS ,
轉換時間 Tconv = 14Tad ，取樣時間 = 31 Tad

A/D = 取樣時間 + 轉換時間 = 31 Tad + 14 Tad = (31 + 14) x 1.08512uS= 48.8304uS

轉換時間計算範例

12-bit ADC

最小的 TAD = 714ns



範例：

Fosc = 29.4912MHz

TCY = 136ns

TAD = TCY(ADCS+1)/2 = 748ns (Tconv. = 14 TAD = 10.5 µs)

TSAMP = TAD x SAMC = 748ns * 5 = 3.8µs

ADCON3

-	-	-	SAMC = 00101	ADRC = 0	-	ADCS = 001010

輸入阻抗與取樣時間

- 觀念！輸入阻抗越大取樣時間越長
  - 公式：T_{SMP} = 放大器穩定時間 (Tamp) + 取樣電容充電時間(Tc) + 溫度係素 (T_{COFF}) – 參考 dsPIC30F Family Reference Manual
    - Tamp 爲固定時間 = 0.5uS
    - Tc = - C_{HOLD} (RIC +RSS +Rin) In (1/2n)Secs
    - T_{COFF} = (Temp - 25°C) (0.05uS / °C) 溫度越高取樣時間變長

條件：

1. C_{HOLD} = 18Pf
2. Rin = 2.5Kohm
3. Vi = 4095 LSB , n=4096 for full scale input voltage
4. Rss = 1.2Kohm
5. Temp = 25°C

Tc = -18pF (250 + 1.2K + 2.5K) In (1/8192)
= 0.641uS

T_{SMP} = 0.5us + 0.641uS+ (25°C-25°C)(0.05/°C)
= 1.141uS (取樣時間)

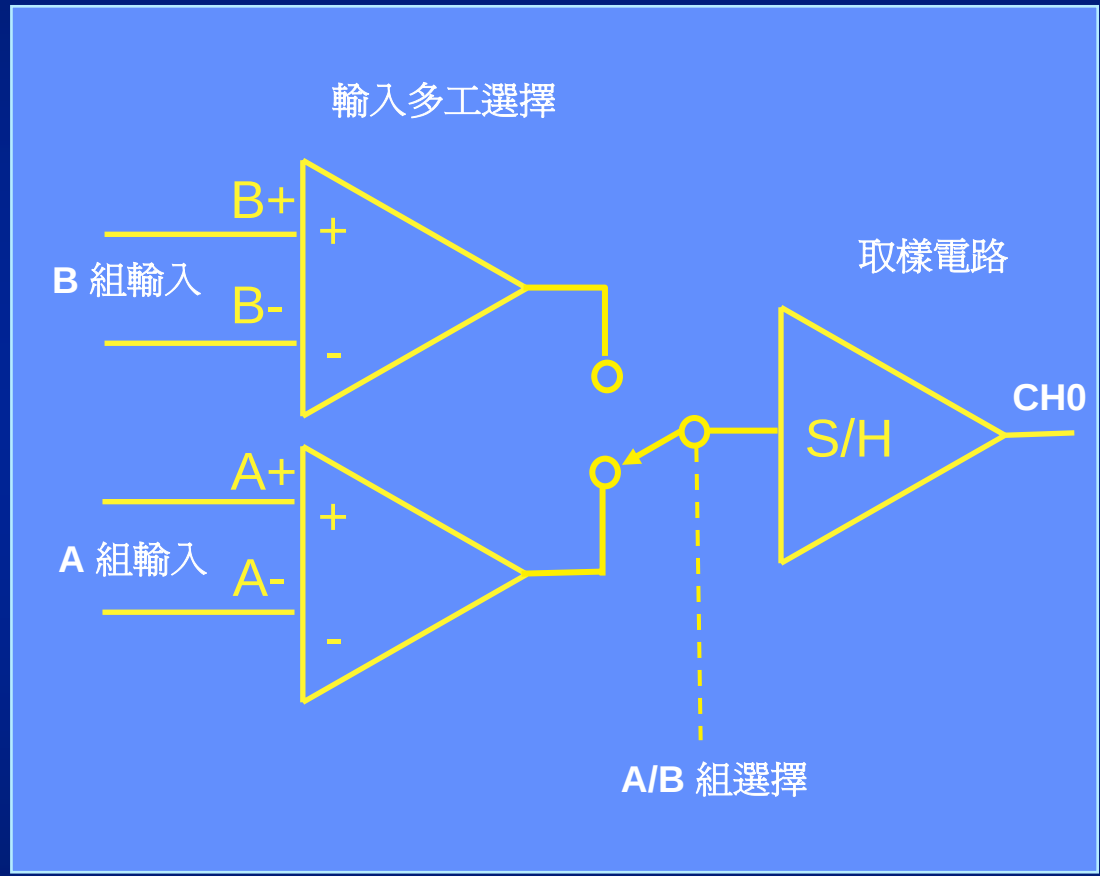
A/D 轉換器
ADCHS 暫存器

定義：

A、B 兩組多工輸入選擇

輸入多工器

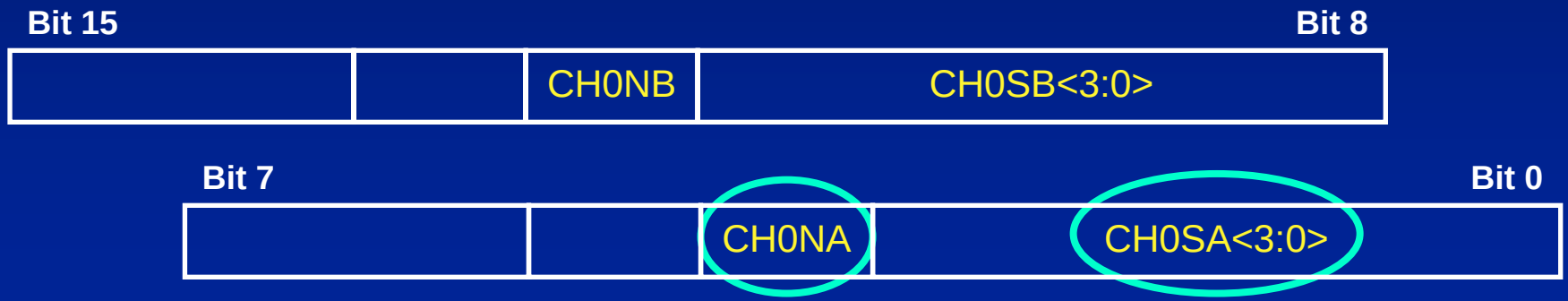
- 12-bit AD 轉換器有兩組多工輸入選擇(A及B)
- 每次的轉換僅能使用一組輸入
- 轉換的電壓值爲正輸入端電壓減去負端輸入端電壓



輸出值爲：[(VA+ - VA-)- Vref-] / (Vref+ - Vref-) * 4096 - 1

ADCHS 暫存器 MUX-A 的輸入選擇

ADCHS



CH0NA : A 組多工器負端輸入選擇

1 = 輸入選 AN1 , 0 = 輸入選 Vref-

CH0SA<3:0> : A 組多工器正端輸入選擇

1111 : 選擇 AN15爲輸入端

.....

0000 : 選擇 AN0爲輸入端

ADCHS 暫存器 MUX-B 的輸入選擇

ADCHS

CH0NB : B 組多工器負端輸入選擇

1 = 輸入選 AN1 , 0 = 輸入選 Vref-

CH0SB<3:0> : B 組多工器正端輸入選擇

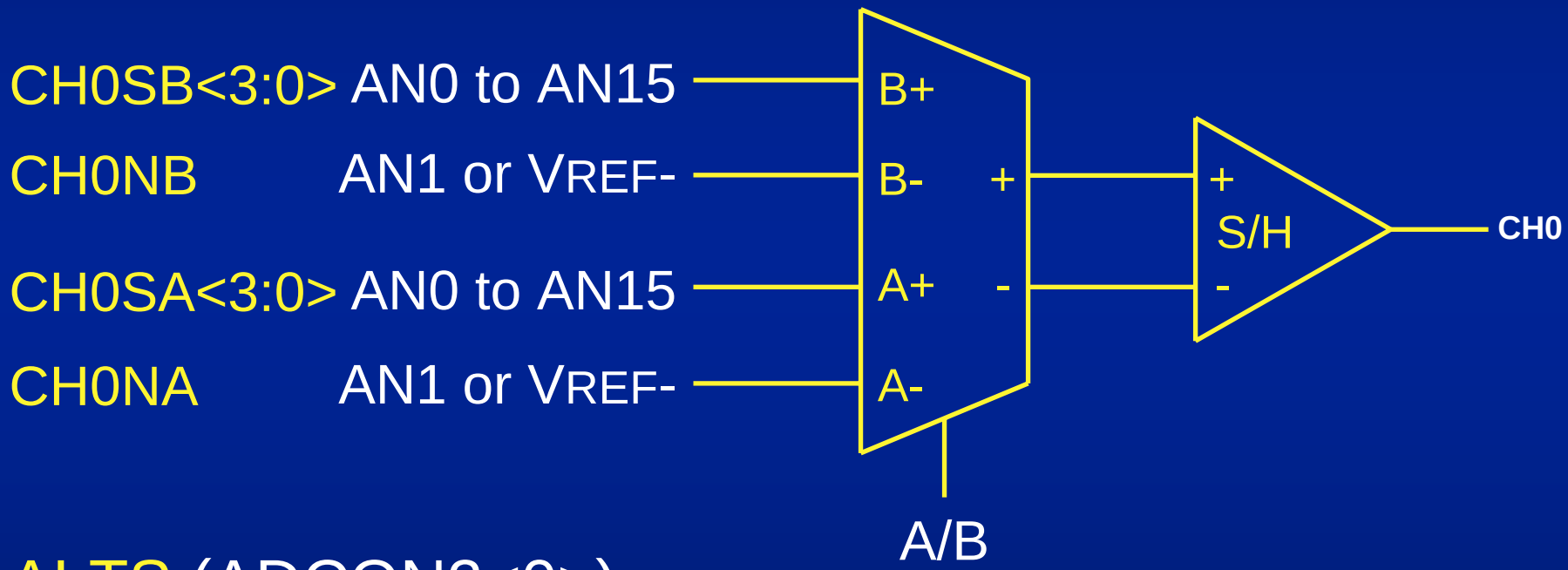
1111 : 選擇 AN15爲輸入端

……

0000 : 選擇 AN0爲輸入端

A/D 轉換器
多工 (交錯式) 輸入

多工器的取樣輸出 CH0



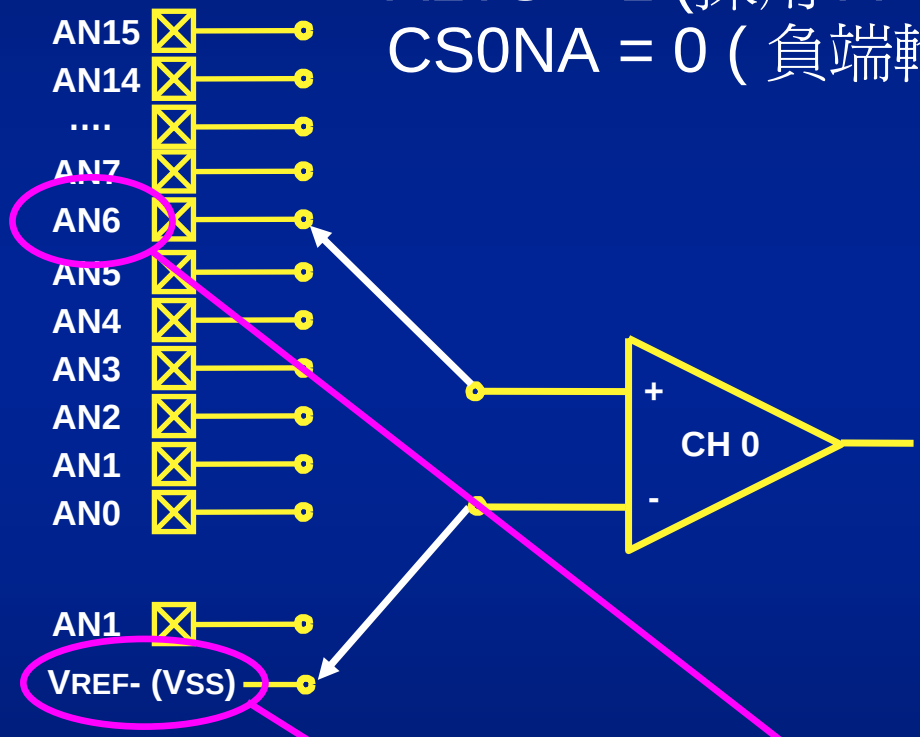
ALTS (ADCON2<0>)

= 0時，輸入只選 A 組多工器

= 1時，先選 A 組輸入再選 B 組輸入相互交替

A/D 轉換器
多工 (交錯式) 輸入說明 (一)

ALTS = 1 (採用 A、B 兩組交替模式)
CS0NA = 0 (負端輸入爲 Vref- 即接地)



ADRES Buffer

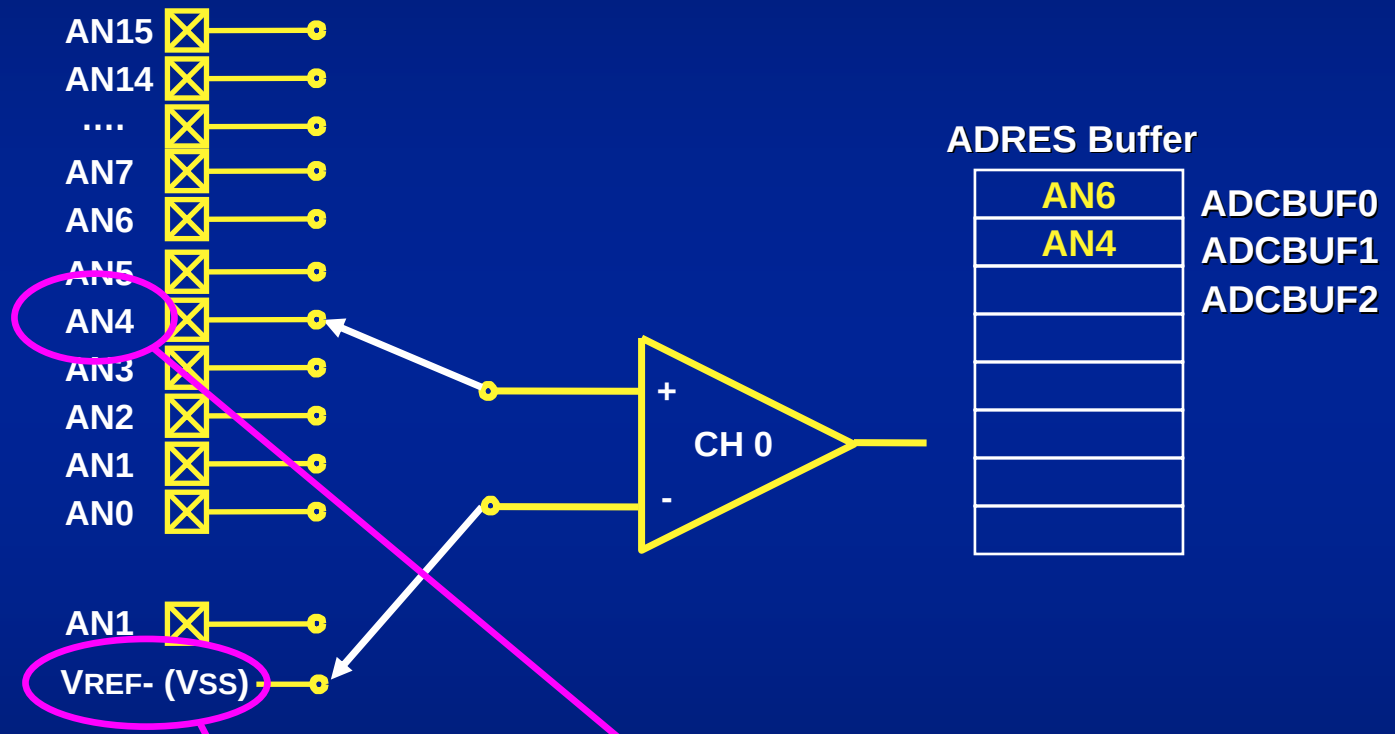
AN6	ADCBUF0
	ADCBUF1
	ADCBUF2

ADCHS

		CH0NB=0	CH0SB = 0100
		CH0NA=0	CH0SA = 0110

Bit 0

A/D 轉換器
多工 (交錯式) 輸入說明 (二)



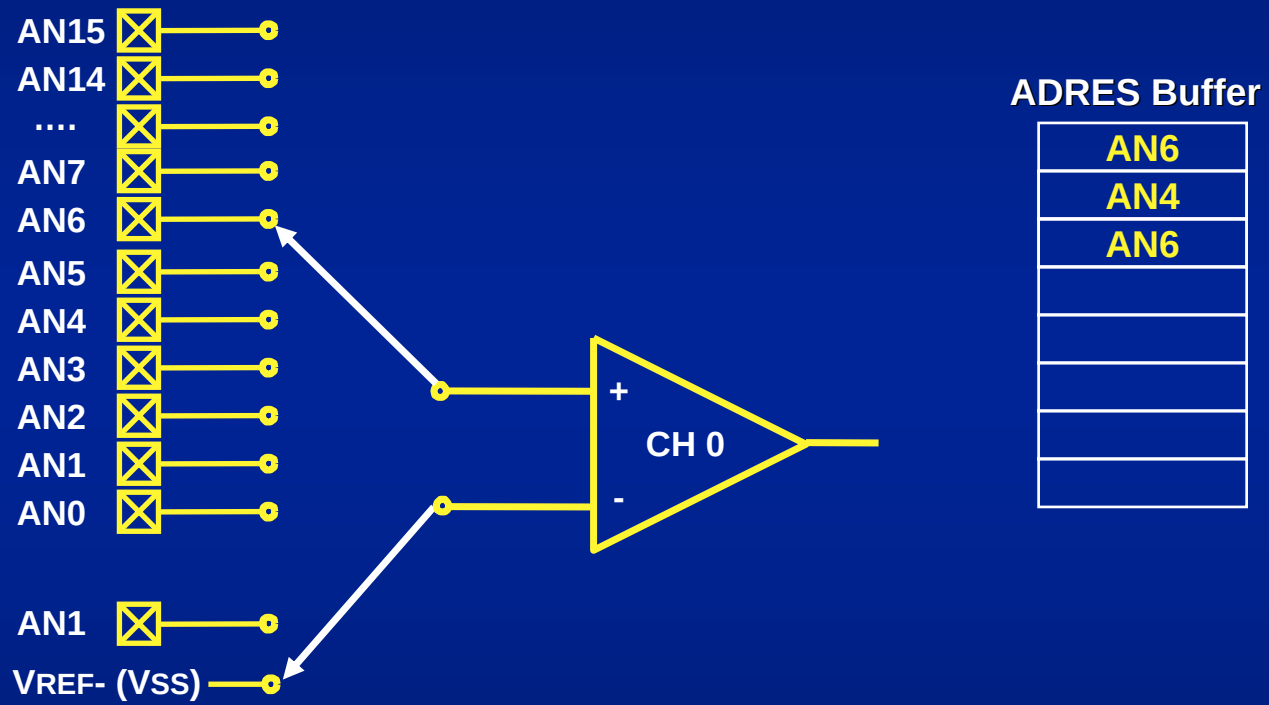
ADCHS

		CH0NB=0	CH0SB = 0100

Bit 0

		CH0NA=0	CH0SA = 0110

A/D 轉換器
多工 (交錯式) 輸入說明 (三)

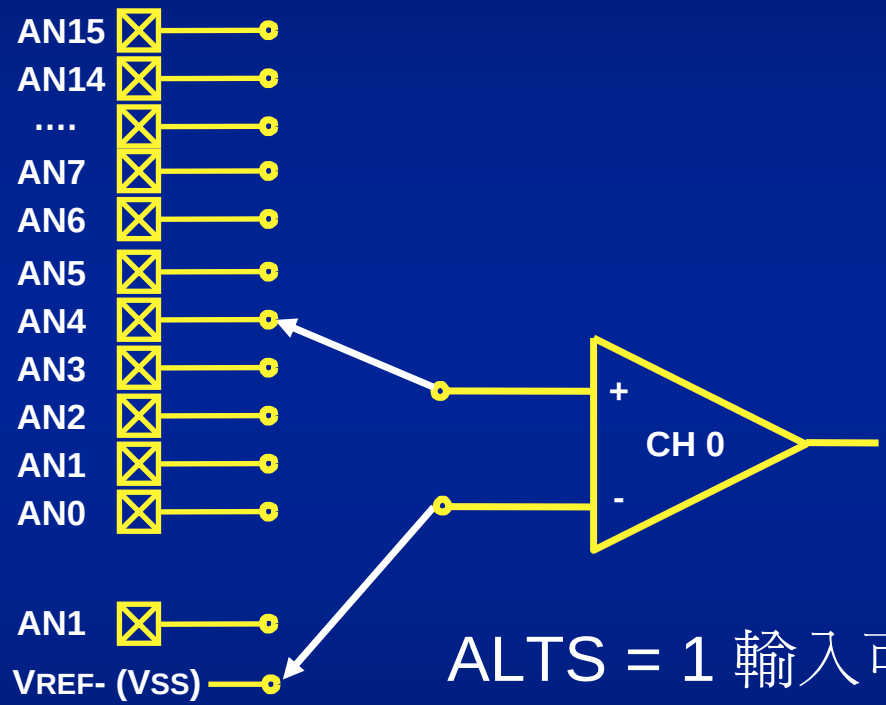


ADCHS

		CH0NB=0	CH0SB = 0100
		CH0NA=0	CH0SA = 0110

Bit 0

A/D 轉換器
多工 (交錯式) 輸入說明 (四)



ALTS = 1 輸入可自動交互更換

ADCHS

		CH0NB=0	CH0SB = 0100

Bit 0

		CH0NA=0	CH0SA = 0110

A/D 轉換器
ADCSSL 暫存器

定義：

類比輸入腳位掃描選擇

ADCSSL 暫存器

ADCSSL

CSSL<15:0> - 輸入掃描選擇

- 1 – 啓動掃描該腳位
- 0 – 不掃描

啓動掃描功能需將 CSCNA (ADCON2<10>) 位元設爲 1

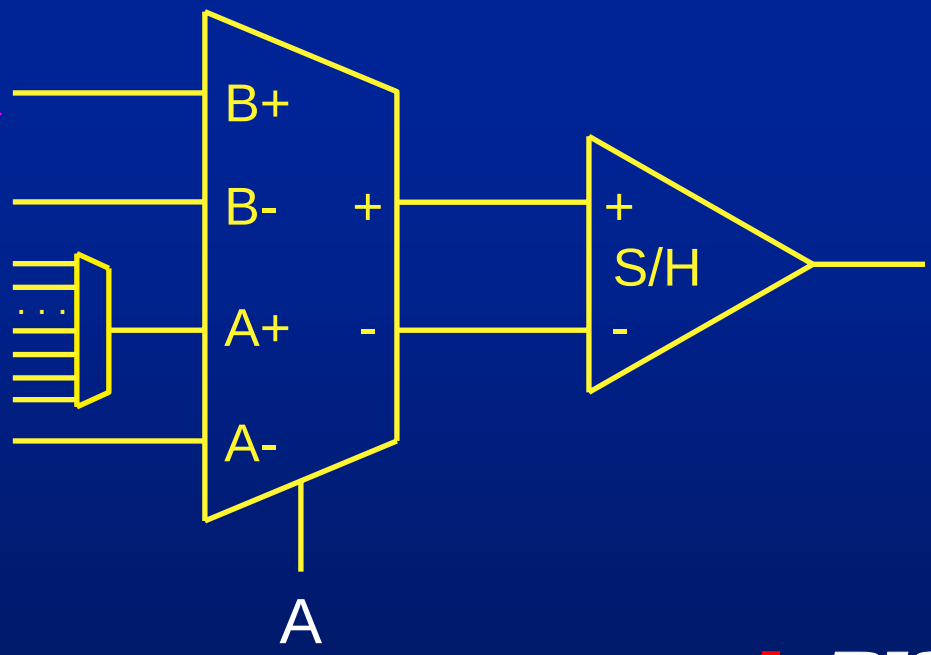
A/D 轉換器 - 掃描輸入

- 注意！只有 A 多工器的輸入端才有掃描之功能
- 這時 CH0SA<3:0> 的輸入被 CSSL<15:0> 的輸入取代
- 啓動掃描功能時，也可以將 ALTS 設爲 1 以啓動 A / B 交互切換功能

範例先討論 **ALTS=0** 的例子
即暫不使用 **B** 組多工器輸入

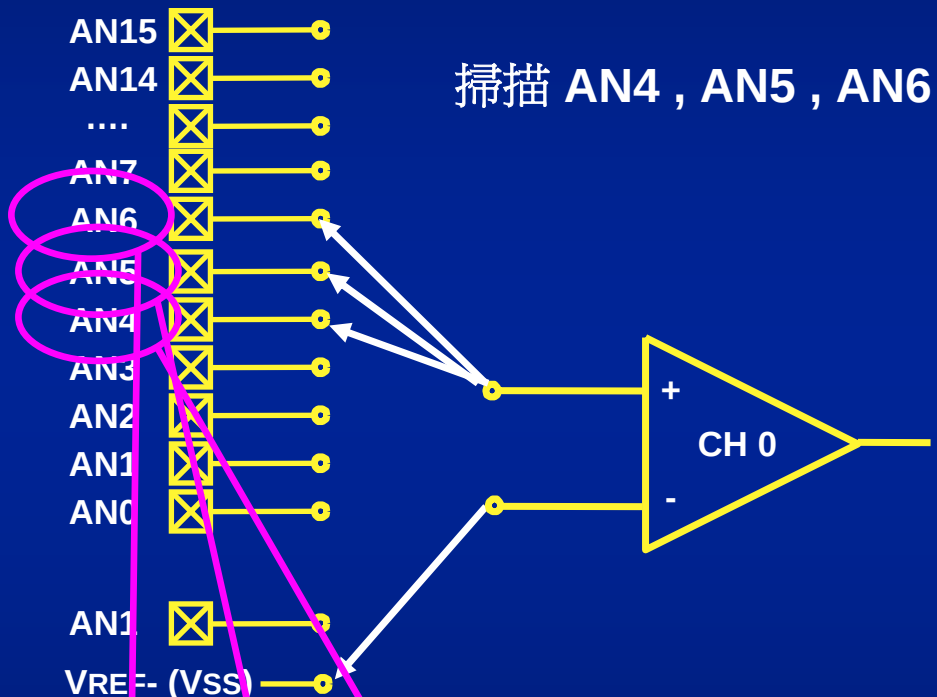
CSSL<15:0> AN0 to AN15

CH0NA AN1 or VREF-



A/D 轉換器
掃描輸入說明

ALTS = 0
CSCNA = 1



ADCSSL

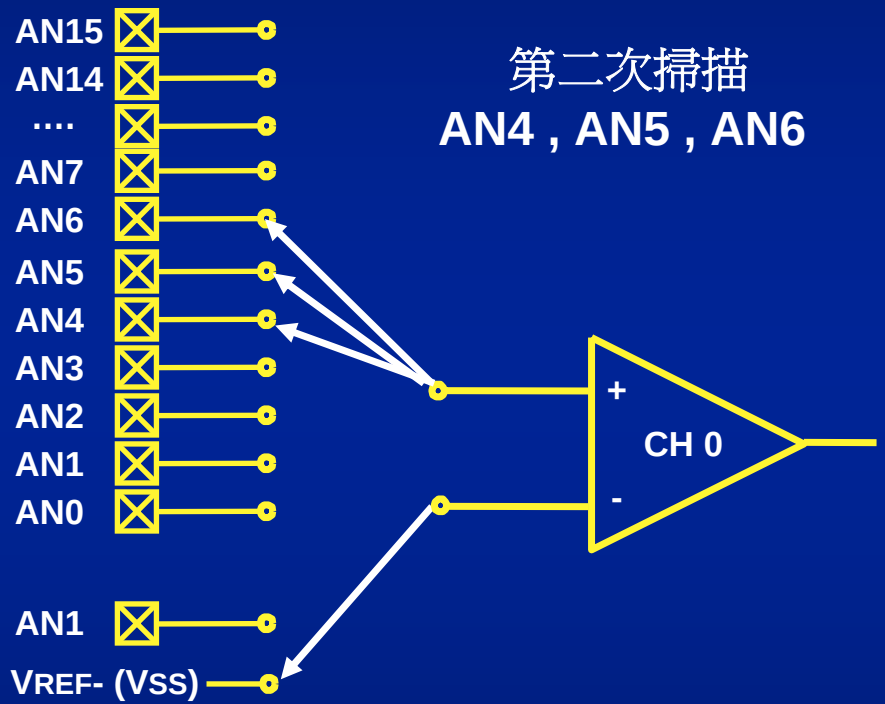
CSSL15	CSSL14	CSSL13	CSSL12	CSSL11	CSSL10	CSSL9	CSSL8

Bit 0

CSSL7	CSSL6=1	CSSL5=1	CSSL4=1	CSSL3	CSSL2	CSSL1	CSSL0

A/D 轉換器
掃描輸入說明

ALTS = 0
CSCNA = 1



ADRES Buffer

AN4
AN5
AN6
AN4
AN5
AN6

ADCSSL

CSSL15	CSSL14	CSSL13	CSSL12	CSSL11	CSSL10	CSSL9	CSSL8

Bit 0

CSSL7	CSSL6=1	CSSL5=1	CSSL4=1	CSSL3	CSSL2	CSSL1	CSSL0

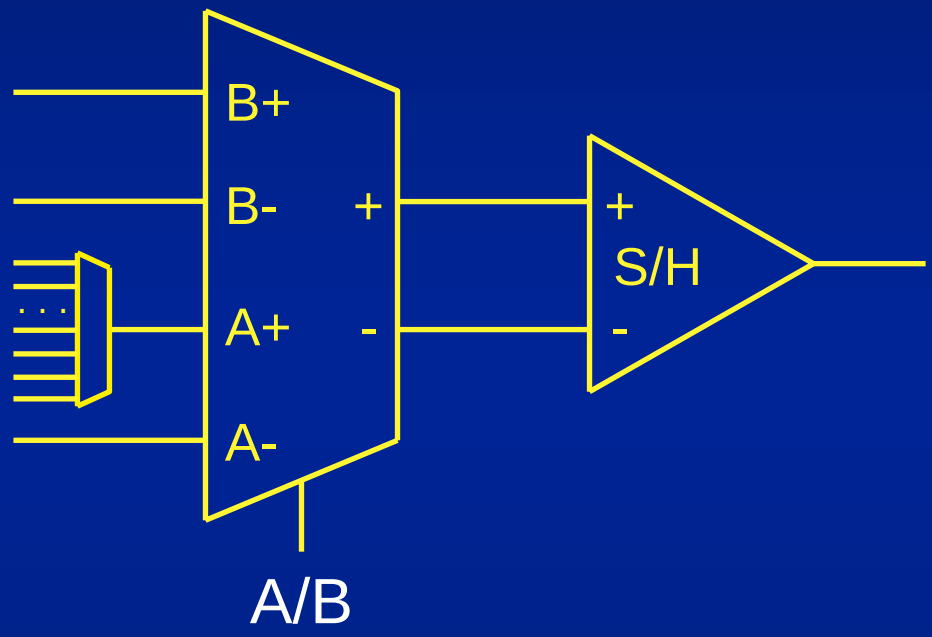
A/D轉換器
使用多工器與掃描輸入說明

CH0SB<3:0> AN0 to AN15

CH0NB AN1 or VREF-

CSSL<15:0> AN0 to AN15

CH0NA AN1 or VREF-



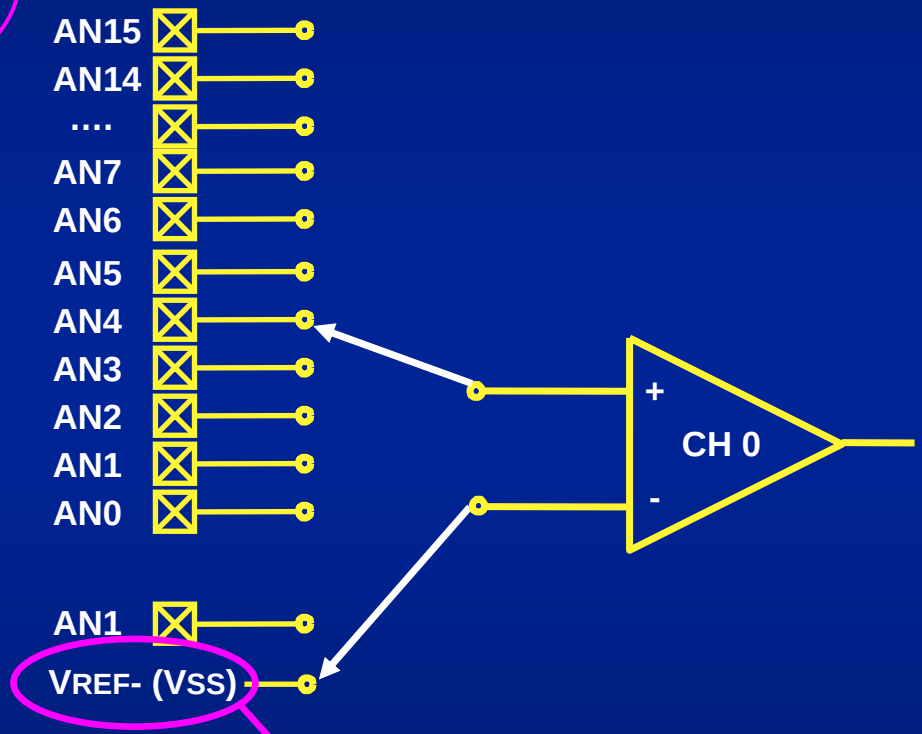
ALTS – 從 A 組輸入先掃描一個輸入再選 B 組輸入，
A / B 兩組之間相互交替做 AD 轉換

CSCNA – 啓動掃描多工器 A 的輸入功能

A/D轉換器
使用多工器與掃描輸入說明

ALTS = 1
CSCNA = 1

多工器 A 的輸入



ADCHS

		CH0NB=0	CH0SB = 0110

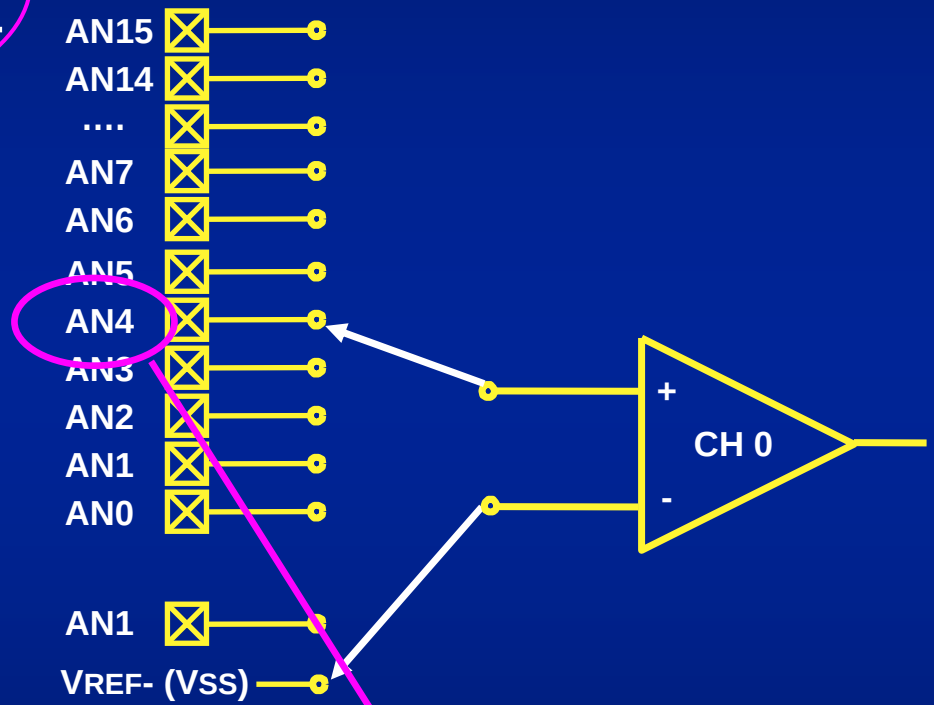
Bit 0

		CH0NA=0	CH0SA

A/D轉換器
使用多工器與掃描輸入說明

ALTS = 1
CSCNA = 1

多工器 A 的輸入



ADRES Buffer

AN4

ADCSSL

CSSL15	CSSL14	CSSL13	CSSL12	CSSL11	CSSL10	CSSL9	CSSL8

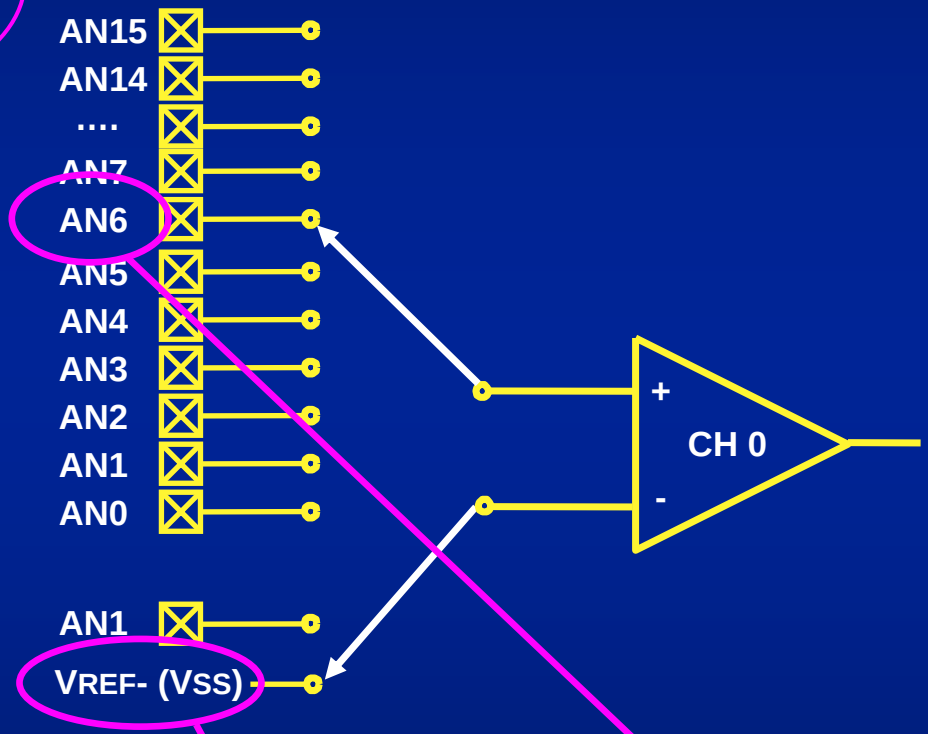
Bit 0

CSSL7	CSSL6	CSSL5=1	CSSL4=1	CSSL3	CSSL2	CSSL1	CSSL0

A/D轉換器
使用多工器與掃描輸入說明

ALTS = 1
CSCNA = 1

多工器 B 的輸入



ADRES Buffer

AN4
AN6

ADCHS

		CH0NB=0	CH0SB = 0110

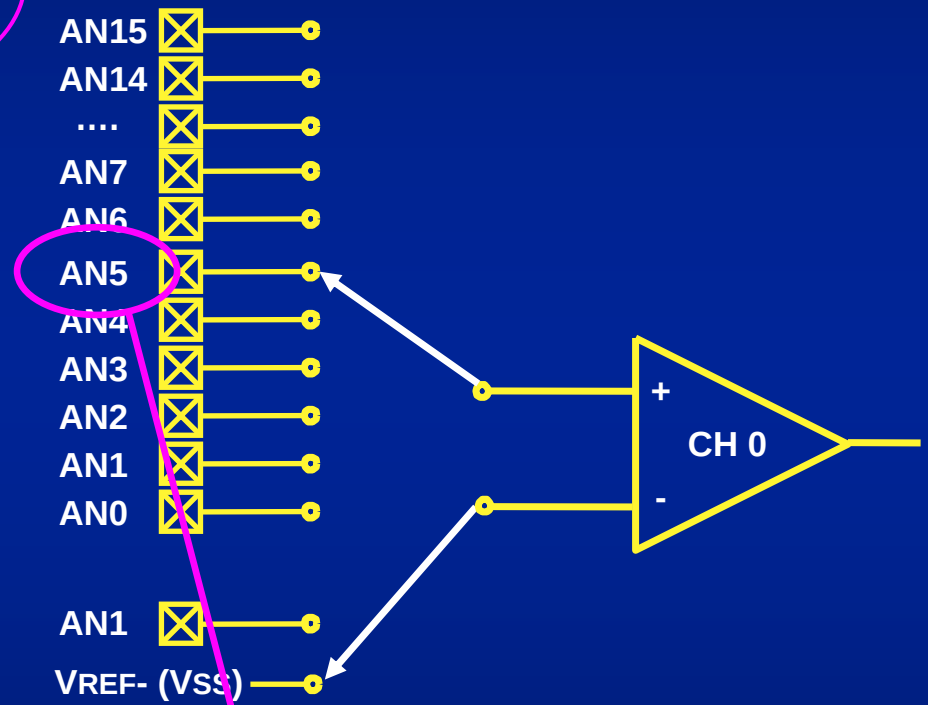
Bit 0

		CH0NA=0	CH0SA

A/D轉換器
使用多工器與掃描輸入說明

ALTS = 1
CSCNA = 1

多工器 A 的輸入



ADRES Buffer

AN4
AN6
AN5

ADCSSL

CSSL15	CSSL14	CSSL13	CSSL12	CSSL11	CSSL10	CSSL9	CSSL8

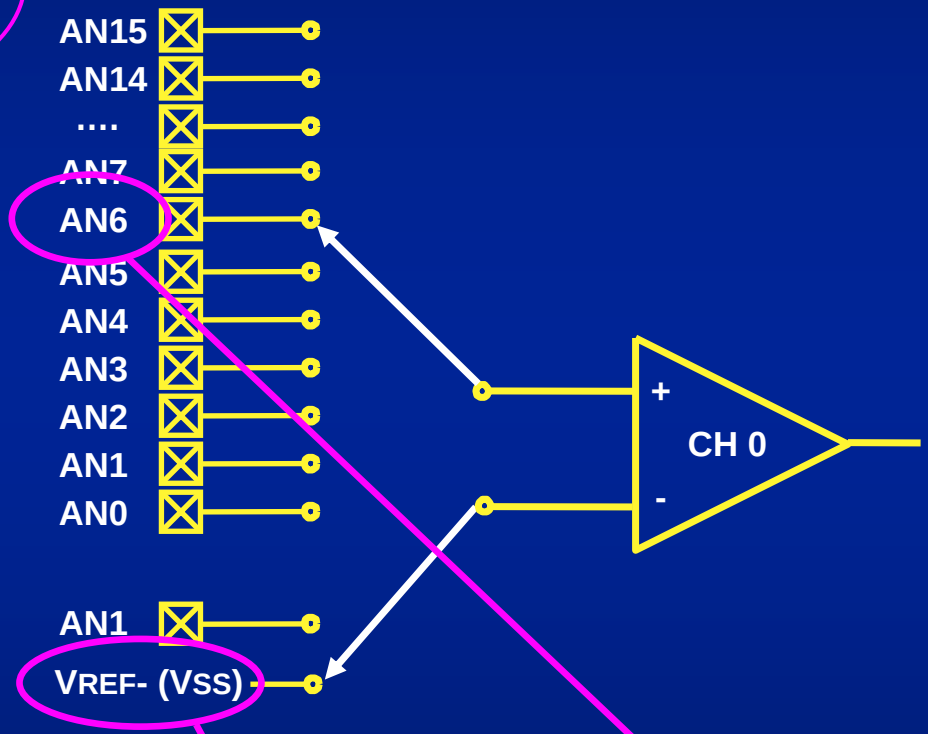
Bit 0

CSSL7	CSSL6	CSSL5=1	CSSL4=1	CSSL3	CSSL2	CSSL1	CSSL0

A/D轉換器
使用多工器與掃描輸入說明

ALTS = 1
CSCNA = 1

多工器 B 的輸入



ADRES Buffer

AN4
AN6
AN5
AN6

ADCHS

		CH0NB=0	CH0SB = 0110

Bit 0

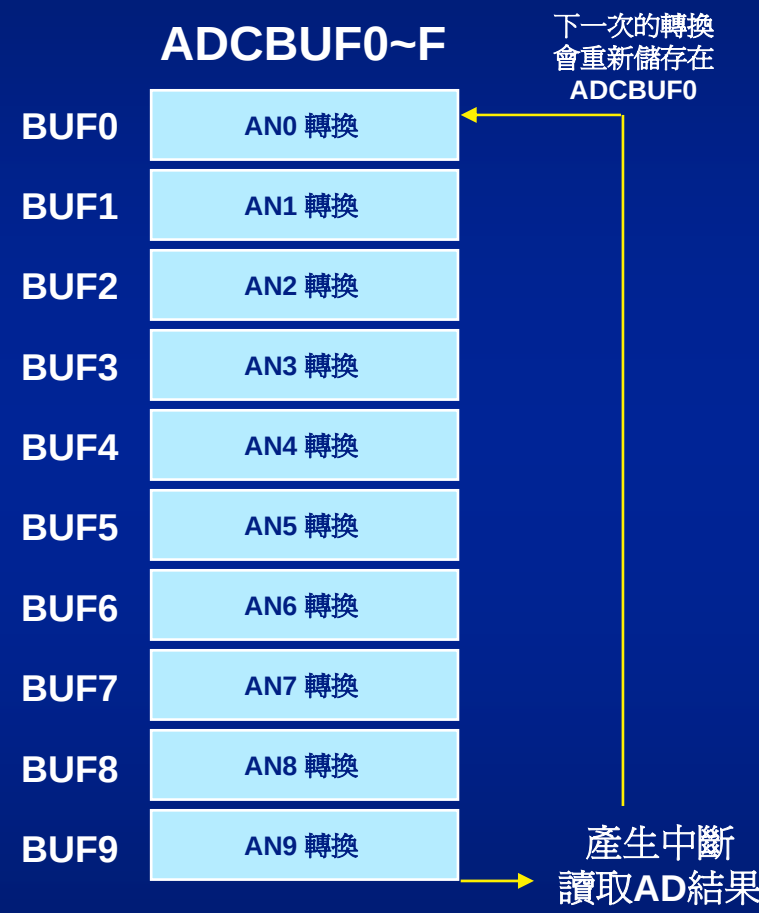
		CH0NA=0	CH0SA

AD 掃描輸入與中斷設定

設定需求：

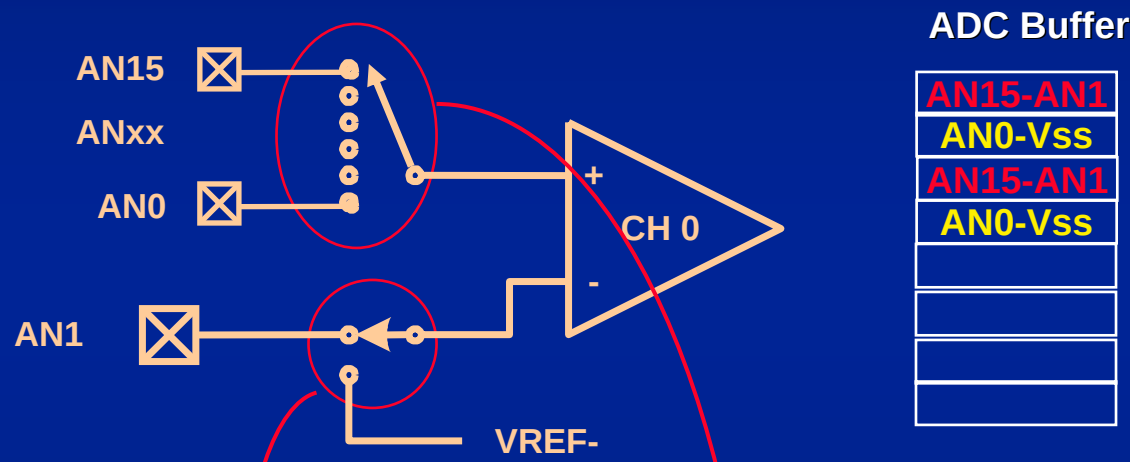
AN0-AN9 爲輸入腳，轉換 10 次後產生一次中斷

- SMPI<2:0>=1001，轉換10次後中斷一次
- BUFM=0，採用單一 buffer (16-word)
- ALTS=0，只使用 MUX A 爲輸入
- CH0SA<3:0>=N/A(掃描模式下該輸入無效)
- CHONA=0， Vref- for CH0- 輸入
- CSCNA=1，啓動輸入掃描功能
- CSSL<15:0>= 0000 0011 1111 1111，掃描輸入腳爲 AN0 ~ AN9
- CH0SB<3:0>= N/A
- CH0NB=N/A



差動輸入與交互掃描

- 輸入的設計是採用差動式輸入，正端的輸入可以減去負端的 AN1 或 VREF- 的電壓。



ADCHS

		Vref- = Vss	AN0
		CH0NB=0	CH0SB<3:0> = 0000

Bit 8

		AN1	AN15
		CH0NA=1	CH0SA<3:0>=1111

Bit 0

各種 AD 轉換觸發模式

利用 SSRC<2:0> 來選擇 AD 轉換的觸發信號來源

000 = 手動轉換， SAMP=1 時取樣，清除 SAMP 位元後轉換

111 = 使用內部時序設定取樣時間及轉換時間(自動轉換)

~~011 = 馬達控制 PWM 間隔結束時，結束取樣ADC開始轉換~~

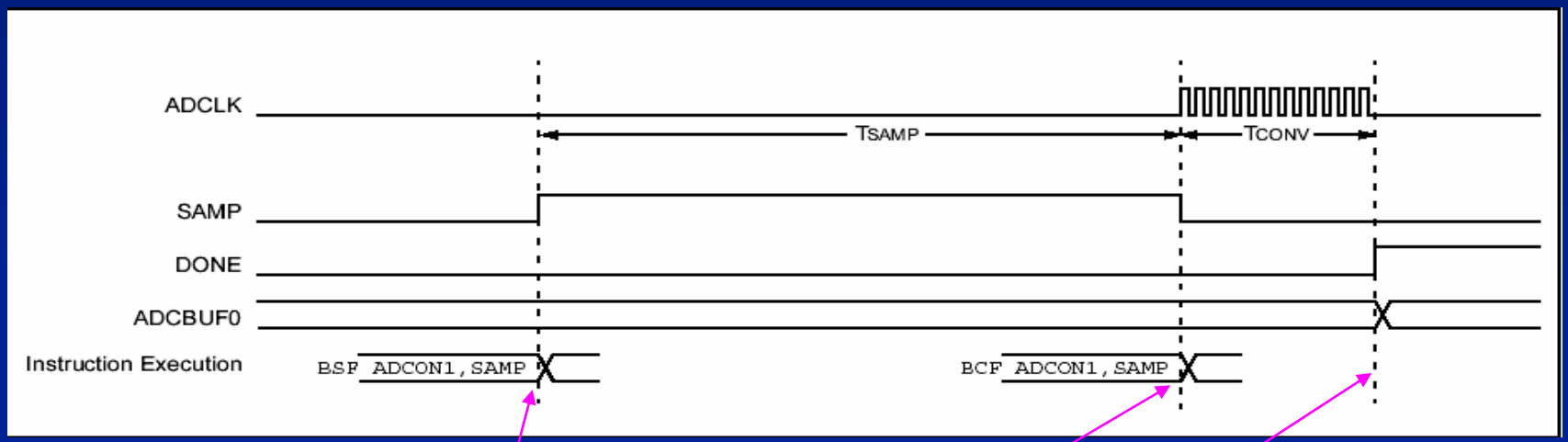
dsPIC30F Family Reference Manual 的 12-bit A/D Converter 書上的說明有誤

010 = Timer 3 計時比較完成後，結束取樣ADC開始轉換

001 = INT0 腳位電位轉態時，結束取樣ADC開始轉換

使用 AD 轉換器 – 手動取樣、手動轉換

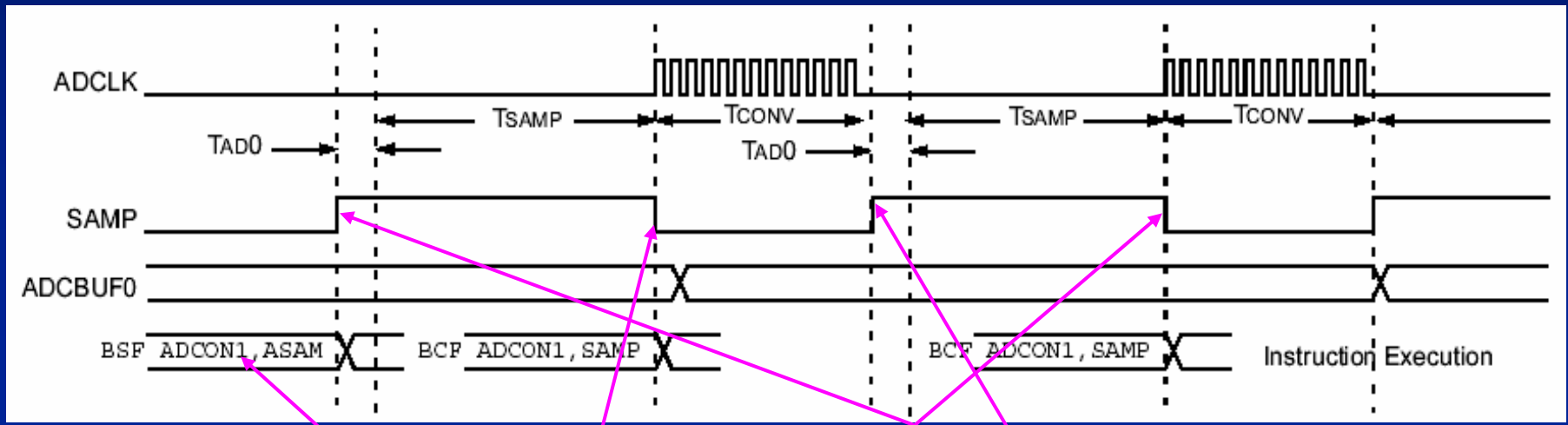
使用此模式的基本設定：SSRC<2:0> = 000 ，ASAM = 0




```
ADCON1bits.ADON=1;              //啓動AD轉換器
While(1)
{
  ADCON1bits.SAMP=1;            //開啓取樣
  DelayNuSec(100);              //延遲100us,取樣電容充電
  ADCON1BITS.SAMP=0;            //取樣結束，開始轉換
  while(!ADCON1bits.DONE);      //轉換完成了嗎 ?
  ADCvalue= ADCBUF0;            //儲存AD轉換結果
}
```

使用 AD 轉換器 – 自動取樣，手動轉換

使用此模式的基本設定：SSRC<2:0> = 000 ，ASAM = 1

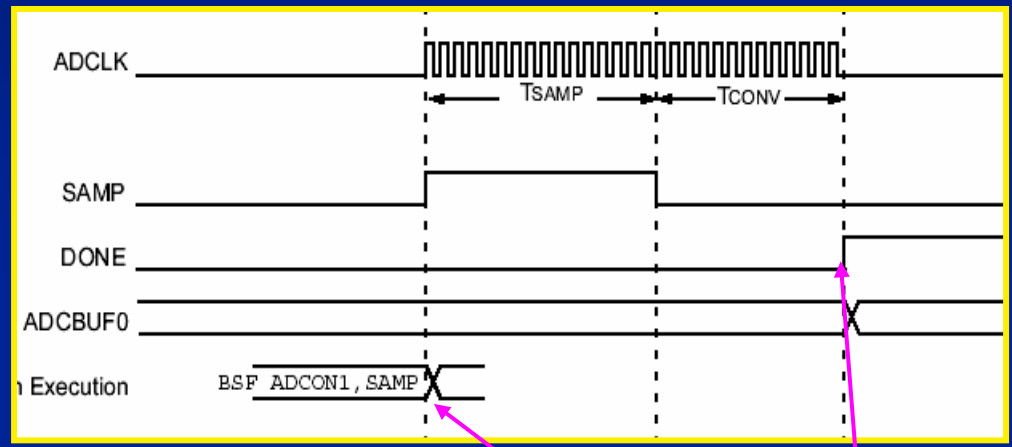



```
ADCON1bits.ADON=1;              //啓動AD轉換器
ADCON1bits.ASAM=1;              //啓動自動取樣，SAMP位元自動設爲 1
While(1)
{
  DelayNuSec(100);              //延遲100us,取樣電容充電
  ADCON1BITS.SAMP=0;            //取樣結束，開始轉換
  while(!ADCON1bits.DONE);      //轉換完成了嗎 ?
  ADCvalue= ADCBUF0;            //儲存AD轉換結果，SAMP位元自動設爲 1
}
```

使用 AD 轉換器 – 手動取樣，TAD 觸發轉換

使用此模式的基本設定：SSRC<2:0> = 111 ，ASAM = 0

- SAMC 位元設定幾個TAD的取樣時間 (SAMC=11111時設需31 TAD)
- 一但設定 SAMP=1，則會自動取樣再轉換

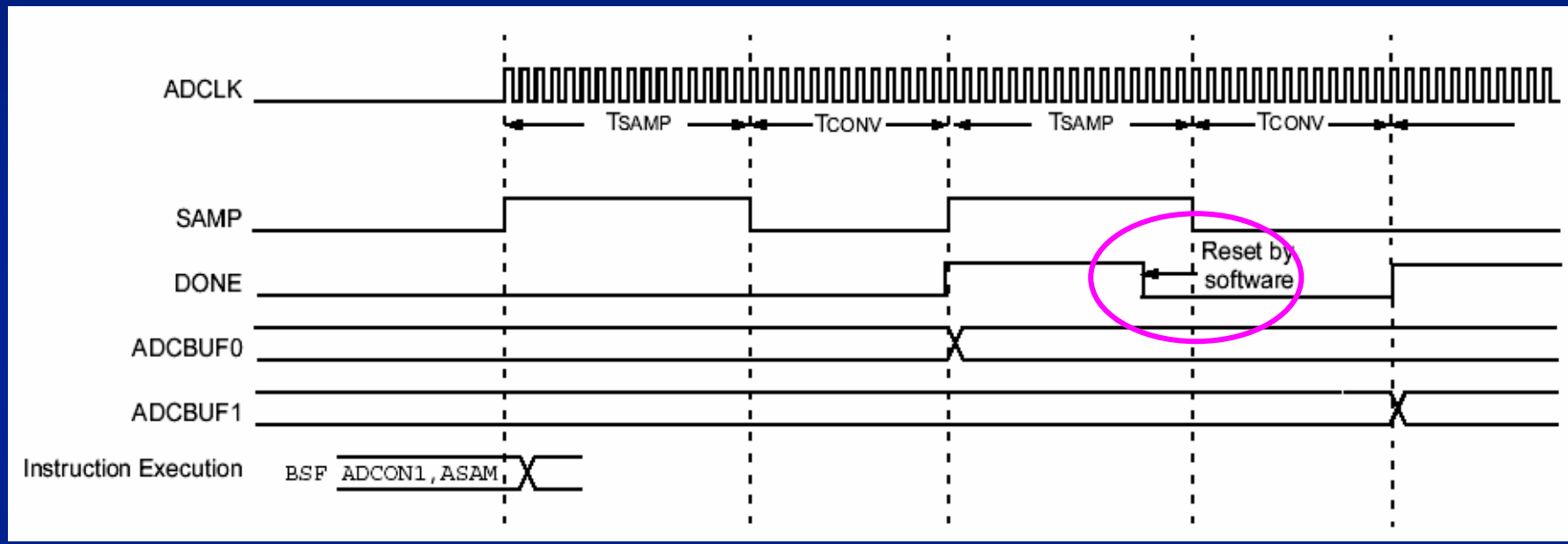



```
/* Initial the AD Module */
ADPCFG = 0xEFFF; // using AN12
ADCON1 = 0x00E0; // SSRC = 111
ADCHS = 0x000C;  // AN12 is CH0 input
ADCSSL = 0x0000; // None Sacn input
ADCON3 = 0x1F02; // 31 Tad , Tad=2Tcy
ADCON2 = 0x0000;
ADCON1bits.ADON=1; // Start AD module
```

```
While(1)
{
  ADCON1BITS.SAMP=1;

  while(!ADCON1bits.DONE);

  ADCvalue= ADCBUF0;
}
```

使用 AD 轉換器 –
自行運作模式 (自動取樣與轉換)

使用此模式的基本設定：**SSRC<2:0> = 111 ，ASAM = 1**



- SSRC<2:0> = 111 轉換觸發來自 A/D 的時脈，同時設定 ASAM=1 (自動取樣與轉換)
- 在此模式下，SAMP會自動設爲 1 進行取樣，完成取樣後(SAMC設定取樣時間) SAMP會自動清爲 0 開啓做轉換動作(需14 T_{AD}) 週而復始
- 詳細程式如下頁之說明

自動取樣與轉換 – 範例程式

```
#define FILTER_BLOCK_LENGTH 512
int SigIn[FILTER_BLOCK_LENGTH];

InitADC12( );
   ADCON1bits.ADON = 1;

   for ( i = 0; i < FILTER_BLOCK_LENGTH; i++ ) // Sample 512 A/D input
   {
        while ( !ADCON1bits.DONE );     // wait for end of conversion
        ADCON1bits.DONE = 0;            // reset DONE bit
        SigIn[i] = ADCBUF0;             // store sample in SigIn buffer
   }
```

```
void InitADC12(void)
{
IFS0bits.ADIF = 0;                  // clear A/D interrupt flag
IEC0bits.ADIE = 0;                  // disable A/D interrupt
ADCON1bits.ADON = 0;                // turn off the A/D converter

ADPCFG = 0xFFF7;  // PCFG3 = 0, only AN3 (RB3) input pin in analog mode
ADCHS = 0x0003;   // CH0NB = 0, CH0SB = 0000, CH0SB = Vref-, CH0SA = AN3
ADCON1 = 0x03E4;  // ADON=0, FORM=Frational, SSRC=AD clock Trigger
                  // ASAM=Auto Sampling & Conversion
ADCON2 = 0x0000;  // VCFG = AVdd and AVss are used, CSCNA = disable scanning
                  // SMPI = 0000 , SMPI = 16-word buffer , ALTS = use MUX A

ADCON3 = 0x1F3F;  // 0001 1111 0011 1111
    /*************************************************************/
    /* SAMC = 11111, sampling time is 31*Tad                     */
    /* ADRC = 0, A/D clock derived from system clock             */
    /* ADCS = 111111, Tad = (ADCS<5:0>+1)* Tcy/2, Tconv=14Tad    */
    /*-----------------------------------------------------------*/
    /*      Tad= 64*33.91nS/2 = 1.08512uS , Tconv= 15.1968uS     */
    /*  A/D = Tsample+Tconv= (32+14)1.08512= 49.915uS = 50uS(20KHz) */
    /*************************************************************/

ADCSSL = 0x0000;                // 0000 0000 0000 0000
}
```

使用 AD 轉換器 – Timer 3 觸發

使用此模式的基本設定 : **SSRC<2:0> = 010 , ASAM = 1 , SAMP=1**

- 使用 Timer 3 計時器觸發中斷
  - 使用Timer3中斷觸發可以使用在轉換速度需求較慢的應用,例如:音頻的取樣 (8K 的取樣時間)
    - 需較快的自動轉換可使用內部時脈驅動方式 (SSRC<2:0>=111)
  - 每次的 Timer 3 觸發會自動清除 SAMP=0 做 AD 的自動轉換，轉換完成後又會因 ASAM = 1 而設定 SAMP=1 做自動取樣，待下次的 Timer 3 觸發時再度清爲零做 AD 轉換，以上動作會週而復始的進行
  - 底下的範例是藉由 Timer 3 的下降緣發使 AD 的產生中斷 (Timer 3 並未設定中斷致能)

Timer 3 觸發 – 範例程式

```
void _ISR _ADCInterrupt(void)
{
        IFS0bits.ADIF = 0 ;      // 清除 ADIF 中斷旗號
        ADC_Buf = ADCBUF0 ; // 讀取 AD 的轉換值
        LED16 = !LED16 ;         // LED16 轉態一次供量取訊號除錯用
}
```

本範例是 **Timer 3** 觸發 **AD** 轉換，待 **AD** 轉換完成時立即產生中斷

注意！**SAMP** 位元硬體會自動設定(取樣)與清除(轉換)，軟體毋需設定此位元

Timer 3 & ADC 的初始設定

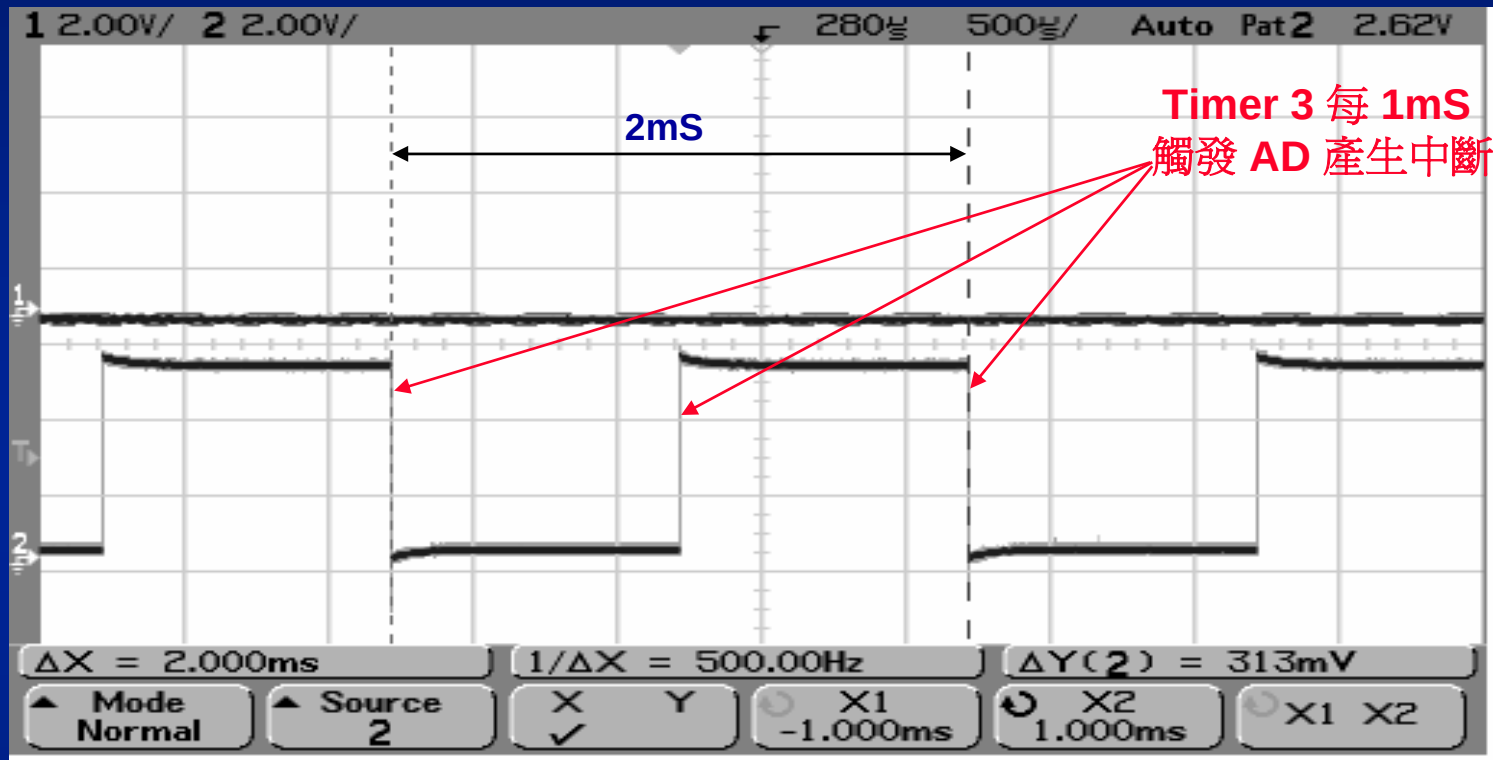
```
Void  Timer3_Initial ( void )
{
    ConfigIntTimer3 ( T3_INT_PRIOR_7 & T3_INT_OFF ) ;       //中斷禁能
    OpenTimer3 ( T3_ON & T3_IDLE_STOP & T3_GATE_OFF & T3_PS_1_1 &
                 T3_SOURCE_INT , (((long)FCY/1000 )) ) ;    // 計時=1mS, 啓動 Timer3
}

void ADC10_Initial (void)
{
    ADPCFG = 0xFF7F;          // AN7/RB7 爲類比電壓輸入腳，其它爲一般 I/O
    ADCON1 = 0x0046;          // 0b0000 0000 0100 0110
                              // 設定 Timer3 爲 AD 轉換的觸發來源
                              // A/D 採用自動取樣自動轉換模式
    ADCON2 = 0x0000;          // 參考電壓：Vref+ = Vdd, Vref- =Vss
                              //不採用輸入掃描方式， SMPI=000（每次轉換完成就產生中斷）
    ADCSSL = 0x0000;          // 不採用輸入掃描方式
    ADCON3 = 0x1F3F;          // TAD = 8 Tcy , SAMC = 15 TAD
    ADCHS =  0x0007 ;         // CH0正端輸入選擇AN7，負端輸入爲Vss
    IEC0bits.ADIE = 1 ;       // 打開AD的中斷
    IPC2bits.ADIP = 7 ;       // 中斷等級＝7 （最高優先權中斷等級）
    ADCON1bits.ADON = 1;      // 啓動 AD
}
```

練習二
使用 Timer3 來觸發 ADC 的轉換

- 練習二的動作 – 讀取 VR1 的位準
  - 將 Timer 3 規劃成週期爲 1ms
  - 將 ADC 規劃成使用 Timer 3 觸發的工作模式
  - 每次 ADC Interrupt 時
    - 將 ADCBUF0 讀進 MyADC
    - 設定旗標 Flags.ADC_DONE
    - 將 APP009 上的 LED16 反向一次
  - 主程式看到 ADC_DONE 旗標爲一時
    - 將 MyADC 的值顯示於 LCD
    - 清除 ADC_DONE

Timer 3 觸發波形量測

LED16



Timer-3 Period = 1ms

RF0 於每次 ADC interrupt 時 Toggle

於是波器可看到 RF0 每 1ms 轉態

使用 AD 轉換器 – 外部 INT0 觸發

使用此模式的基本設定：**SSRC<2:0> = 001 , ASAM = 1 , SAMP=1**

- 使用 INT0 外部觸發中斷
  - INT0 可設定上昇緣或下降緣來觸發 AD 的轉換
    - INTCON2<INT0EP>位元=1，設爲下降緣
    - INTCON2<INT0EP>位元=0，設爲上昇緣
  - 每次的 INT0 觸發會自動清除 SAMP=0 做 AD 的自動轉換，轉換完成後又會設定 SAMP=1 做自動取樣，待下次的 INT0 觸發時再度清爲零做 AD 轉換，以上動作會週而復始的進行
  - 底下的範例是藉由 INT0 的下降緣發使 AD 的產生中斷 (INT0 並未設定中斷致能)

外部 INT0 觸發 – 範例程式

```
void _ISR _ADCInterrupt(void)
{
        IFS0bits.ADIF = 0 ;      // 清除 ADIF 中斷旗號
        ADC_Buf = ADCBUF0 ; // 讀取 AD 的轉換值
        LED16 = !LED16 ;         // LED16 轉態一次供量取訊號除錯用
}
```

本範例是 **INT0** 觸發 **AD** 轉換，待 **AD** 轉換完成時立即產生中斷

注意！**SAMP** 位元硬體會自動設定(取樣)與清除(轉換)，軟體毋需設定此位元

INT0 & ADC 的初始設定

```
void INT0_Initial( void )
{
    TRISEbits.TRISE8 = 1 ;      // Set INT0 for a input
    INTCON2bits.INT0EP = 1 ;    // INT0 interrupt on negative edge
    IEC0bits.INT0IE = 0 ;
}
void ADC10_Initial(void)
{
    ADPCFG = 0xFF7F;            // AN7/RB7 爲類比電壓輸入腳，其它爲一般 I/O
    ADCON1 = 0x0026;            // 0b0000 0000 0010 0110
                                // 設定 INT0 爲 AD 轉換的觸發來源
                                // A/D 採用自動取樣自動轉換模式
    ADCON2 = 0x0000;            // 參考電壓：Vref+ = Vdd, Vref- =Vss
                                //不採用輸入掃描方式， SMPI=000（每次轉換完成就產生中斷）
    ADCSSL = 0x0000;            // 不採用輸入掃描方式
    ADCON3 = 0x1F3F;            // TAD = 8 Tcy , SAMC = 15 TAD
    ADCHS =  0x0007 ;           // CH0正端輸入選擇AN7，負端輸入爲Vss
    IEC0bits.ADIE = 1 ;         // 打開AD的中斷
    IPC2bits.ADIP = 7 ;         // 中斷等級＝7 （最高優先權中斷等級）
    ADCON1bits.ADON = 1;        // 啓動 AD
}
```

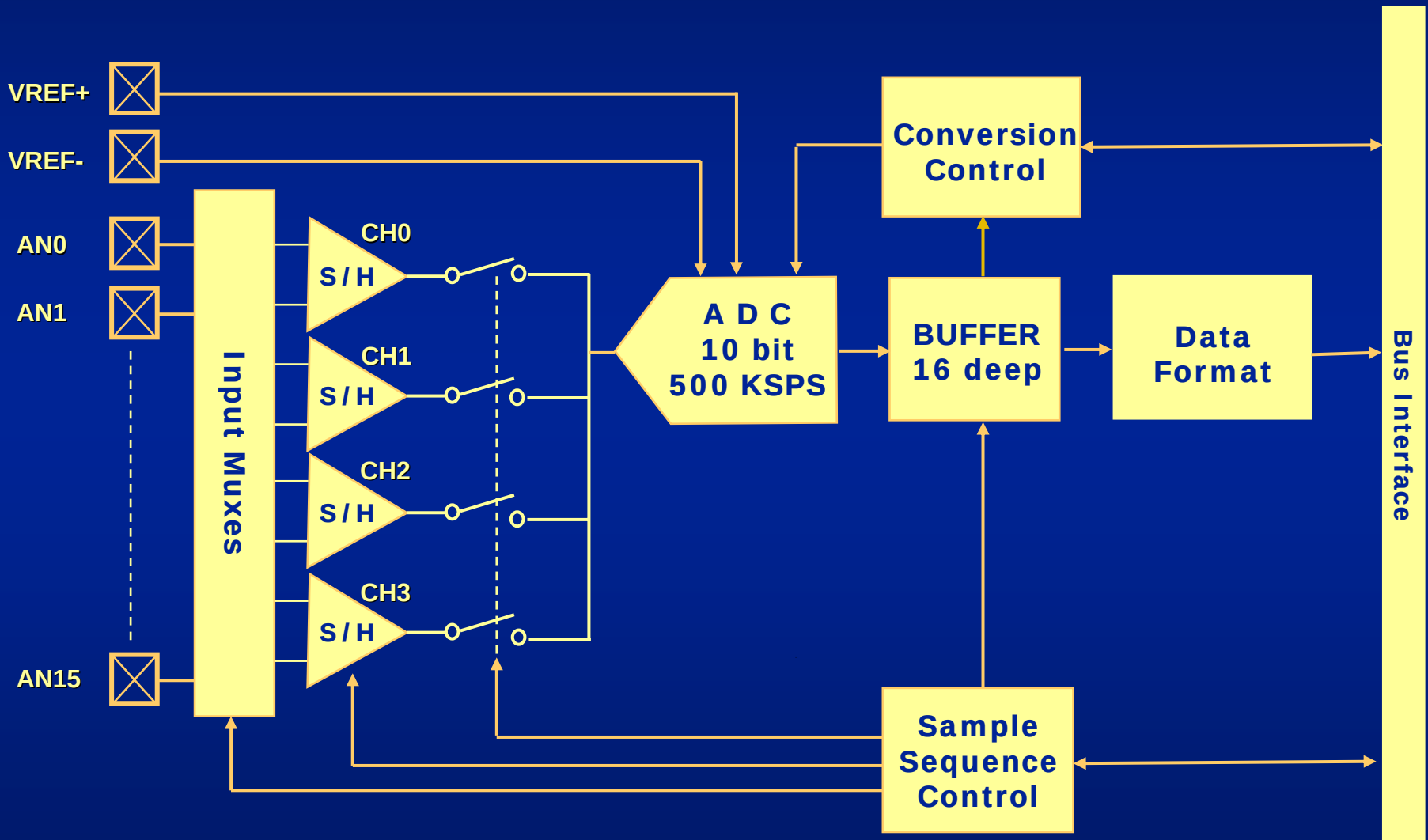
練習三
使用 INT0 來觸發 ADC 的轉換

- 練習三的動作 – 轉換 VR1 的輸入位準
  - 將 INT0/FLTA（按鍵 S8）規劃成下降緣觸發
    - 不要 Enable 其中斷
  - 將 ADC 規劃成使用 INT0 觸發的工作模式
  - 每次 ADC Interrupt 時
    - 將 ADCBUF0 讀進 MyADC
    - 設定旗標 Flags.INT_ACTIVE
    - 將 APP009 上的 LED16 反向一次
  - 主程式看到 INT_ACTIVE 旗標爲 “1” 時：
    - 將 MyADC 的值顯示於 LCD
    - 清除 INT_ACTIVE

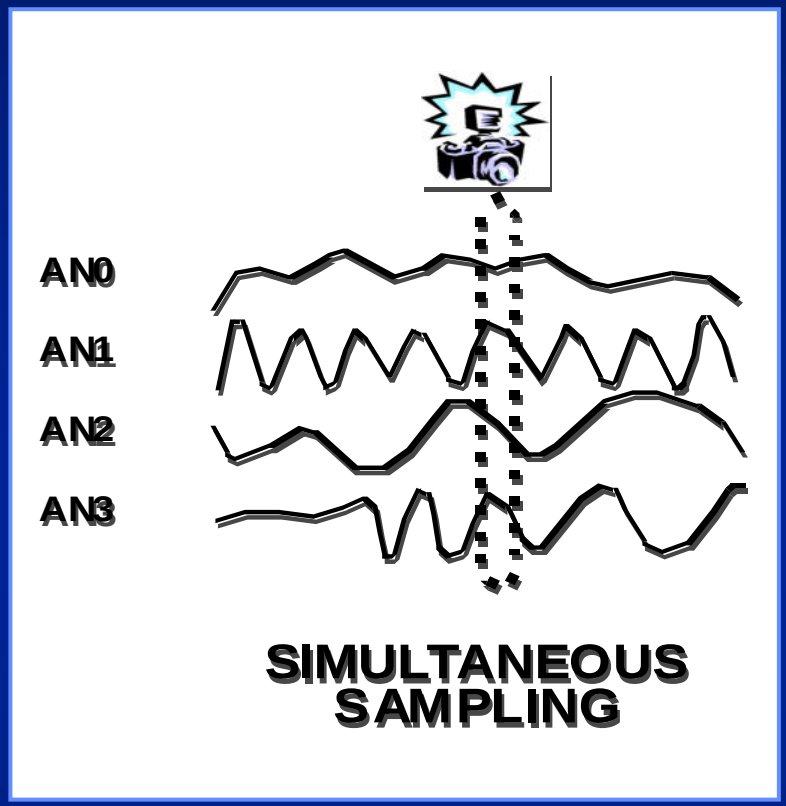
10-bit 高速 AD 轉換器

- 10-bit 與 12-bit 兩著差異
  - 10-bit 轉換速度可達 500Ksps(2uS)
  - 10-bit AD 輸入具有四個同步輸入取樣
  - 12-bit AD 只有一個CH0輸入，10-bit AD 共有四個 CH0，CH1，CH2，CH3
  - 10-bit AD 多了 CH1，CH2，CH3 的輸入選擇 (CH123SA，CH123NA & CH123SB，CH123NB)
  - 10-bit AD 其它的設定與 12-bit AD 相同

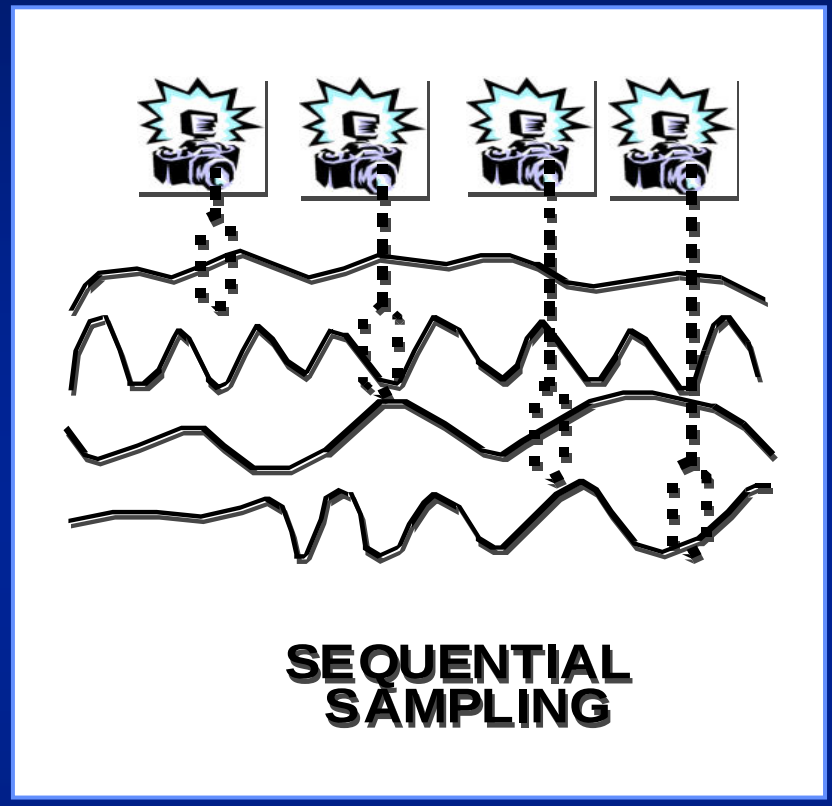
10-bit A/D 方塊圖



兩種取樣方式



同時取樣模式



依序取樣模式

設定同時取樣模式

ADCON1 Register

ADON	-	ADSIDL	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

SSRC<1:0>		-	SIMSAM	ASAM	SAMP	CONV
6	5	4	3	2	1	bit0

- SIMSAM – 同時取樣選擇位元
  - 1 = 同時取樣 CH0, CH1, CH2, CH3 (當 CHPS = 1x)，
    或 同時取樣 CH0 , CH1 (當 CHPS = 01)
  - 0 = 各個輸入單獨取樣，依序轉換 (普通模式)

同時取樣的CH選擇

ADCON2 Register

VCFG<2:0>			OFFCAL	-	CSCNA	CHPS<1:0>	
bit15	14	13	12	11	10	9	bit8

BUFS	-	SMPI<3:0>				BUFM	ALTS
bit7	6	5	4	3	2	1	bit0

- CHPS<1:0> - 選擇同時取樣的輸入數目 (1, 2, 或 4)
  - 1x = 同時轉換 CH0, CH1, CH2 和 CH3
  - 01 = 同時轉換 CH0 和 CH1
  - 00 = 轉換 CH0

CHPS<1:0> 與 SIMSAM 關係

CHPS<1:0>	SIMSAM	取樣 / 轉換 順序	取樣時間
00	x	取樣 CH0，轉換 CH0	1
01	0	取樣 CH0，轉換 CH0 取樣 CH1，轉換 CH1	2
1X	0	取樣 CH0，轉換 CH0 取樣 CH1，轉換 CH1 取樣 CH2，轉換 CH2 取樣 CH3，轉換 CH3	4
01	1	同時取樣 CH0 CH1 轉換 CH0 轉換 CH1	1
1X	1	同時取樣 CH0, CH1,CH2, CH3 轉換 CH0 轉換 CH1 轉換 CH2 轉換 CH3	1

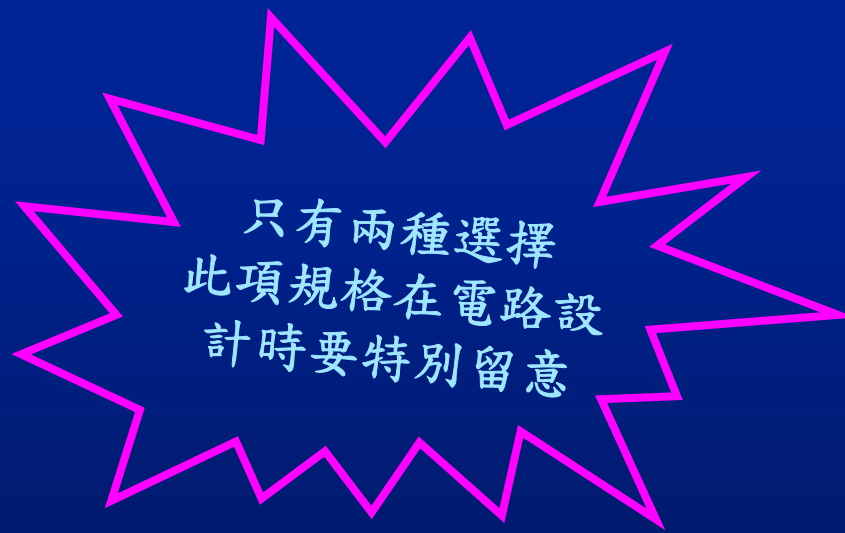
使用同時取樣注意事項 (一)

- CH0 (以 MUX A 爲例)
  - 正端輸入可從 AN0 ~ AN15 ➔(CH0SA<3:0>)
  - 負端輸入有 AN1，Vref- ➔(CH0NA)
- CH1(以 MUX A 爲例)
  - 正端輸入可從 AN0，AN3 ➔(CH123SA)
  - 負端輸入有 AN6，AN9，Vref- ➔(CH123NA<1:0>
- CH2(以 MUX A 爲例)
  - 正端輸入可從 AN1，AN4 ➔(CH123SA)
  - 負端輸入有 AN7，AN10，Vref- ➔(CH123NA<1:0>
- CH3(以 MUX A 爲例)
  - 正端輸入可從 AN2，AN5 ➔(CH123SA)
  - 負端輸入有 AN8，AN11，Vref- ➔(CH123NA<1:0>

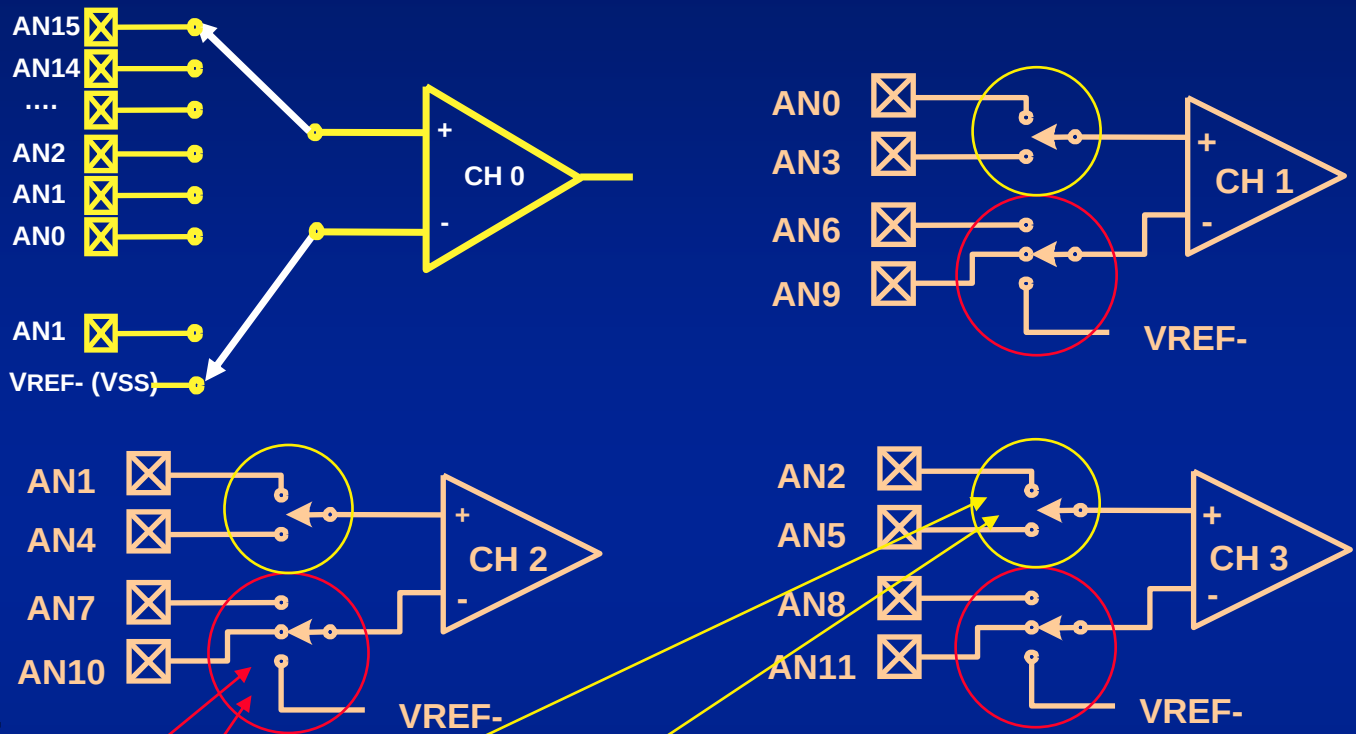
使用同時取樣注意事項 (二)

(以使用 MUX A 爲例)

- 若 SIMSAM=1 (設定同時取樣模式) & CHPS<1:0>= 11 (同時取樣 CH0~CH3)
  - CH123SA=0 :
    - CH1的輸入選 AN0
    - CH2的輸入選 AN1
    - CH3的輸入選 AN2
  - CH123SA=1 :
    - CH1的輸入選 AN3
    - CH2的輸入選 AN4
    - CH3的輸入選 AN5



10-bit A/D 轉換器



ADCHS Register

CH123NB<1:0>	CH123SB	CH0NA	CH0SB<3:0>				
bit15	14	13	12	11	10	9	bit8

CH123NA<1:0>	CH123SA	CH0NA	CH0SA<3:0>				
bit7	6	5	4	3	2	1	bit0

10-bit AD轉換速率

- 最快轉換速度爲 2uS (500K sps)
  - 此速度不包含取樣時間
- 最小 T_{AD} 的需求：167nS
  - 167nS * 12 Tad = 2 uS
- 使用內部 RC 振盪源時
  - Vdd > 3V, Tad 的典型值爲 300nS

$T_{CY} = (1 / F_{OSC}) * 4$

$T_{AD} = T_{CY} (ADCS +1) / 2$

範例:

120MHz速度執行，TAD=8TCY, TAD=266.64nS, 轉換速度=3.2uS

25MHz速度執行，TAD=2TCY, TAD=320nS, 轉換速度=3.86uS

A/D 轉換步驟

- 選擇 / 設定類比輸入腳 (ADPCFG<15:0>
- 設定參考電壓源 (ADCON2<15:12>)
- 選擇資料輸出格式 (ADCON1<9:8>)
- 設定AD轉換觸發模式 (ADCON1<7:5>)
- 設定轉換時脈的時間 (ADCON3<5:0>)
- 如需設定自動取樣 (使用自動觸發模式)
  - 設定自動取樣(ADCON3<12:8>)
- 選擇 A/D 輸入模式及多工群組 (ADCON2<ALTS,CSCNA>)
  - 設定 ALTS，如需使用多工群組 A 及群組 B
  - 是否需要輸入掃描功能
- 是否使用同時取樣模式
  - (ADCON1<SIMSAM>), (ADCON2<9:8>)
- 設定中斷模式 (ADCON2<5:2>, 啓動中斷，中斷優先權設定
- 啓動 AD 模組

A/D 轉換設定：範例一

- 自動掃描 AN0 - AN7，自動轉換，設成兩組輸出暫存區
  - 自動掃描 8 類比輸入 - CSCNA = 1, ADCSSL = 0x00FF
  - 自動轉換 - SSRC<2:0> = 111, ASAM = 1
  - 設定 SAMC<4:0> 自動取樣時間
  - 8 次轉換後產生中斷 - SMPI<3:0> = 0111
  - BUFM = 1 使用兩組資料緩衝器

	Buffer at 1st interrupt
ADCBUF0	CH0 - AN0
ADCBUF1	CH0 - AN1
ADCBUF2	CH0 - AN2
ADCBUF3	CH0 - AN3
ADCBUF4	CH0 - AN4
ADCBUF5	CH0 - AN5
ADCBUF6	CH0 - AN6
ADCBUF7	CH0 - AN7

	Buffer at 2nd interrupt
ADCBUF8	CH0 - AN0
ADCBUF9	CH0 - AN1
ADCBUFA	CH0 - AN2
ADCBUFB	CH0 - AN3
ADCBUFC	CH0 - AN4
ADCBUFD	CH0 - AN5
ADCBUFE	CH0 - AN6
ADCBUFF	CH0 - AN7

A/D 轉換設定：範例二

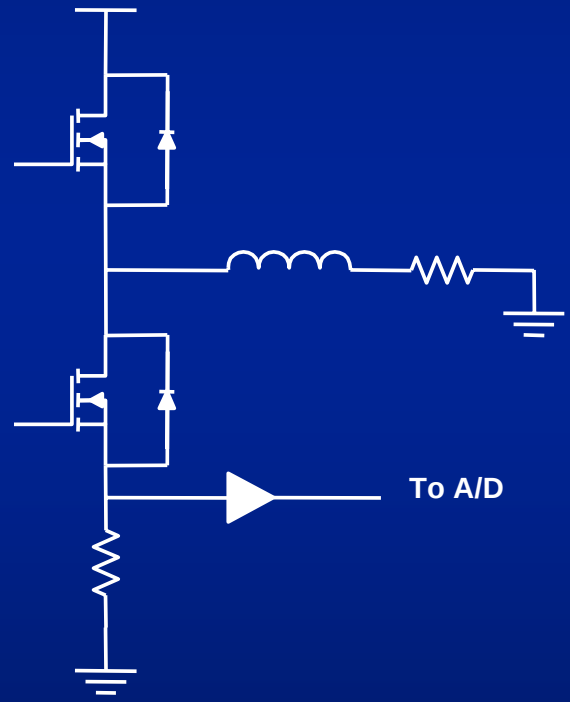
- 依序掃描 4 類比輸入(AN0,AN1,AN2,AN3)及高速掃描一個類比輸入AN4
  - 設定 A、B 兩組多工輸入模式：ALTS=1
  - 多工器 A 將使用四個類比輸入依序掃描 AN0-AN3, CSCNA = 1 ，ADCSSL = 0x000F
  - 多工器 B 永遠使用 AN4 爲輸入
  - 轉換八次後產生中斷：SMPI<3:0> = 0111
  - 連續取樣
  - 輸入只使用 CH0

ADCBUF0	CH0 - AN0
ADCBUF1	CH0 - AN4
ADCBUF2	CH0 - AN1
ADCBUF3	CH0 - AN4
ADCBUF4	CH0 - AN2
ADCBUF5	CH0 - AN4
ADCBUF6	CH0 - AN3
ADCBUF7	CH0 - AN4

使用 AD 轉換器 – PWM 觸發

使用此模式的基本設定 : **SSRC<2:0> = 011 , ASAM = 1 , SAMP=1**

- 使用 Motor PWM 觸發中斷
  - SEVTCMP 暫存器設定 A/D 觸發轉換的時間值
  - SEVTDIR<15> 設定在 PTMR 暫存器往上數或往下數時用 SEVTCMP 的內容值來觸發 ADC
  - 如此可確保 A/D 可正確地偵測 PWM Low 導通時的電流



Motor PWM 觸發 – 範例程式

```
void _ISR _ADCInterrupt(void)
{
        IFS0bits.ADIF = 0 ;      // 清除 ADIF 中斷旗號
        ADC_Buf = ADCBUF0 ; // 讀取 AD 的轉換值
        LED16 = !LED16 ;         // LED16 轉態一次供量取訊號除錯用
}
```

本範例是 **Motor PWM** 觸發 **AD** 轉換，待 **AD** 轉換完成時立即產生中斷

注意！**SAMP** 位元硬體會自動設定**(**取樣**)**與清除**(**轉換**)**，軟體毋需設定此位元

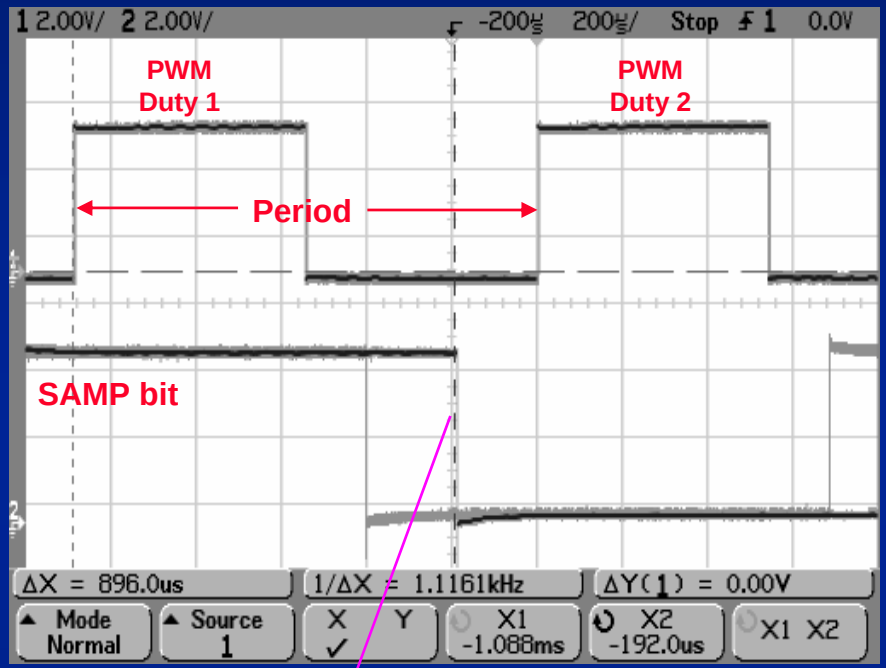
10-bit ADC 的初始設定

```
void ADC10_Initial(void)
{

    ADPCFG = 0xFF7F;        // AN7/RB7 爲類比電壓輸入腳，其它爲一般 I/O
    ADCON1 = 0x0066;        // 0b0000 0000 0110 0110
                            // 設定 Motor PWM 爲 AD 轉換的觸發來源
                            // A/D 採用自動取樣自動轉換模式
    ADCON2 = 0x0000;        // 參考電壓：Vref+ = Vdd, Vref- =Vss
                            //不採用輸入掃描方式， SMPI=000（每次轉換完成就產生中斷）
    ADCSSL = 0x0000;        // 不採用輸入掃描方式
    ADCON3 = 0x1F3F;        // TAD = 8 Tcy , SAMC = 15 TAD
    ADCHS =  0x07 ;         // CH0正端輸入選擇AN7，負端輸入爲Vss

    IEC0bits.ADIE = 1 ;     // 打開AD的中斷

    IPC2bits.ADIP = 7 ;     // 中斷等級＝7 （最高優先權中斷等級）

    ADCON1bits.ADON = 1;  // 啓動 AD
}
```

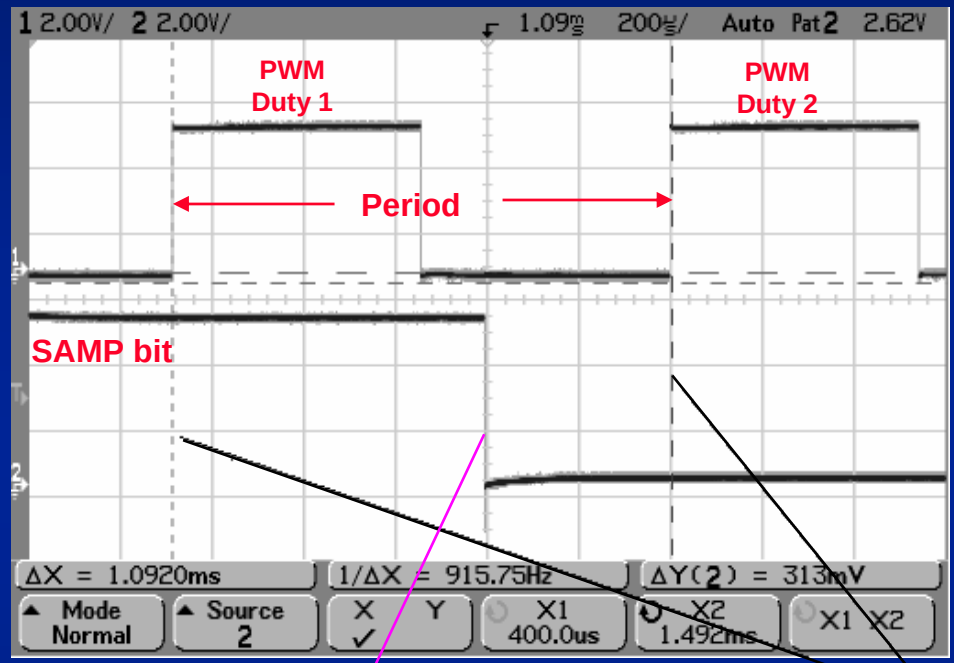
PWM1 的初始設定

```
void        MotPWM_Initial(void)
{
    IEC2bits.PWMIE = 0 ;        // Disable PWM Interrupt !!
    IEC2bits.FLTAIE = 0 ;
    OVDCON = 0xff00 ;           // Active all PWM OUTPUT !!
    TRISE = 0xffc0 ;
    PTCON = 0xa008 ;            // Configure as 0b1010 0000 0000 1000
                                // PWM Time Base Prescale = 1:16
                                // PWM Time Base OP in free running Mode
    PWMCON1 = 0x0077 ;          // Configure as 0b0000000001110111
                                // PWM I/O in complementary Mode and only PWM1L/H
                                // as PWM output
    PWMCON2 = 0x0000 ;          // Configure as 0b0000000000000000
    DTCON1 = 0x0101 ;           // Configure as 0b0000 0010 0000 0010 ;
    FLTACON = 0x0000 ;
    IPC9bits.PWMIP = 6 ;
    PTPER = 1000 ;              // PWM Time Base Period Register
    PDC1 = 1000 ;
    PDC2 = 1000 ;
    PDC3 = 1000 ;
}
```

Motor PWM 觸發波形量測



PWM Period = 1.092mS
PTPTR = 1000
SEVTCMP = 800



PWM Period = 1.092mS
PTPTR = 1000
SEVTCMP = 600

第三單元

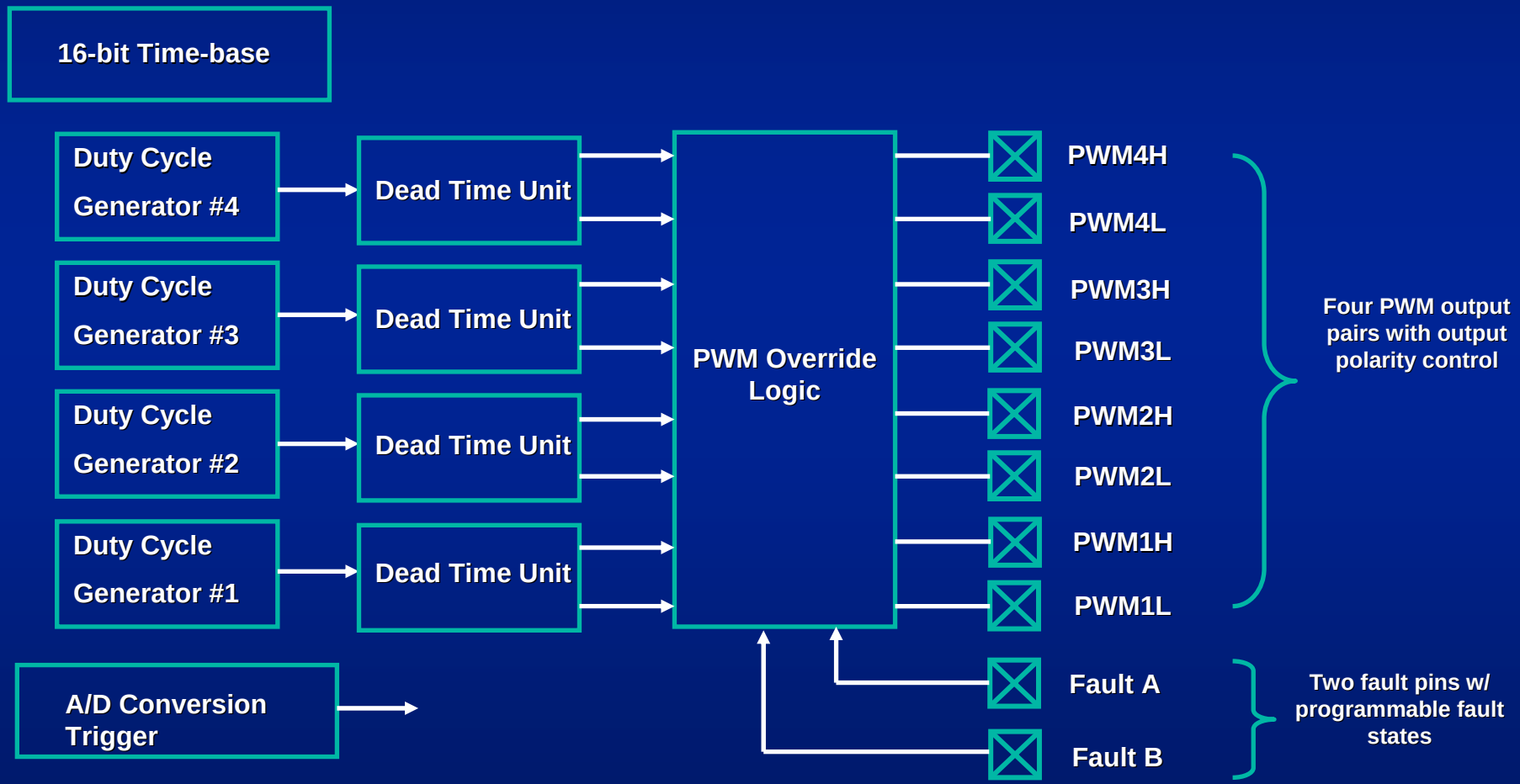
Motor Control PWM

Comprehensive Motor Drive

Motor PWM Features

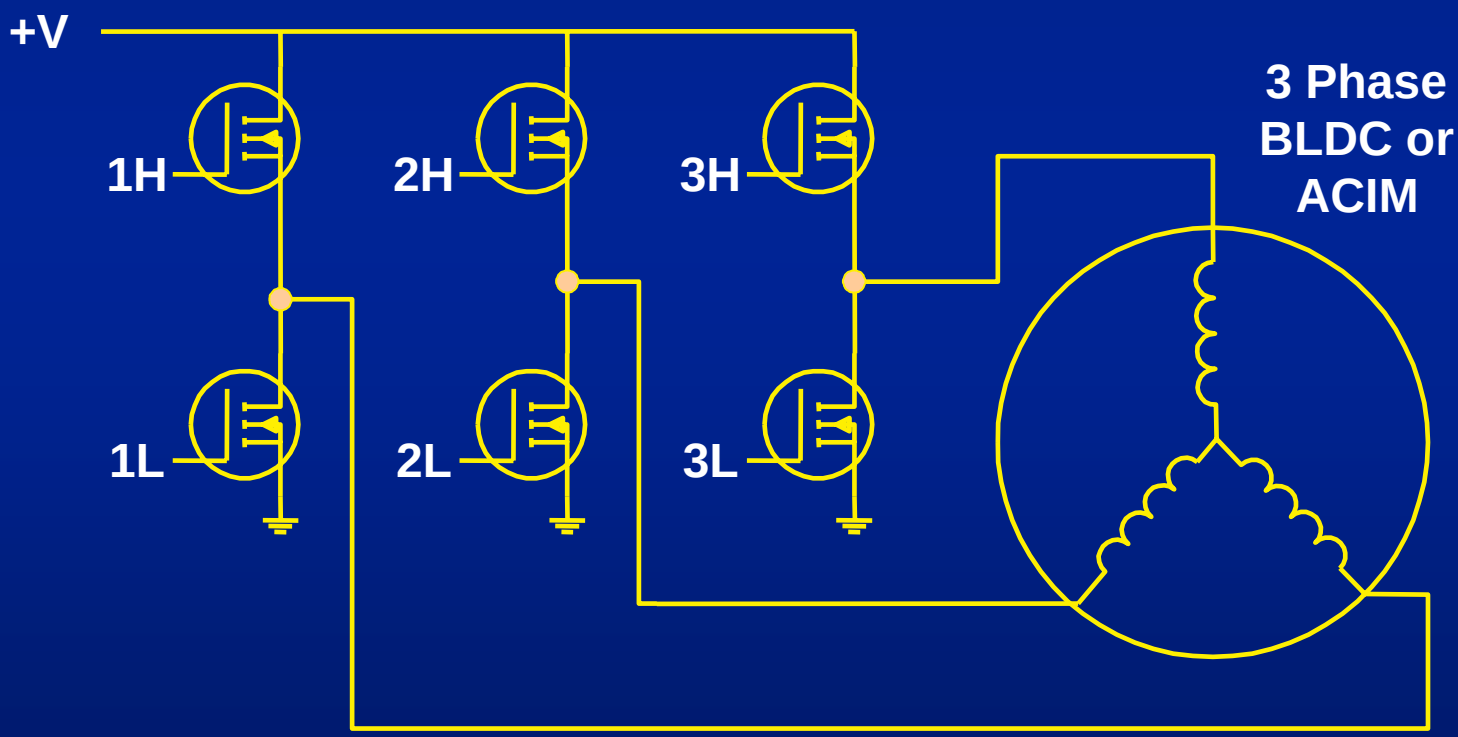
- Dedicated timebase
- Four PWM generators
  - Each PWM generator drives a pair of I/O pins
  - I/O can be complementary or independent
  - Programmable output polarity for I/O
- Dead time unit prevents power shoot-through
- Output override control
- Two hardware shutdown pins
  - Programmable shutdown state
  - Two operating modes

Motor Control PWM Block Diagram



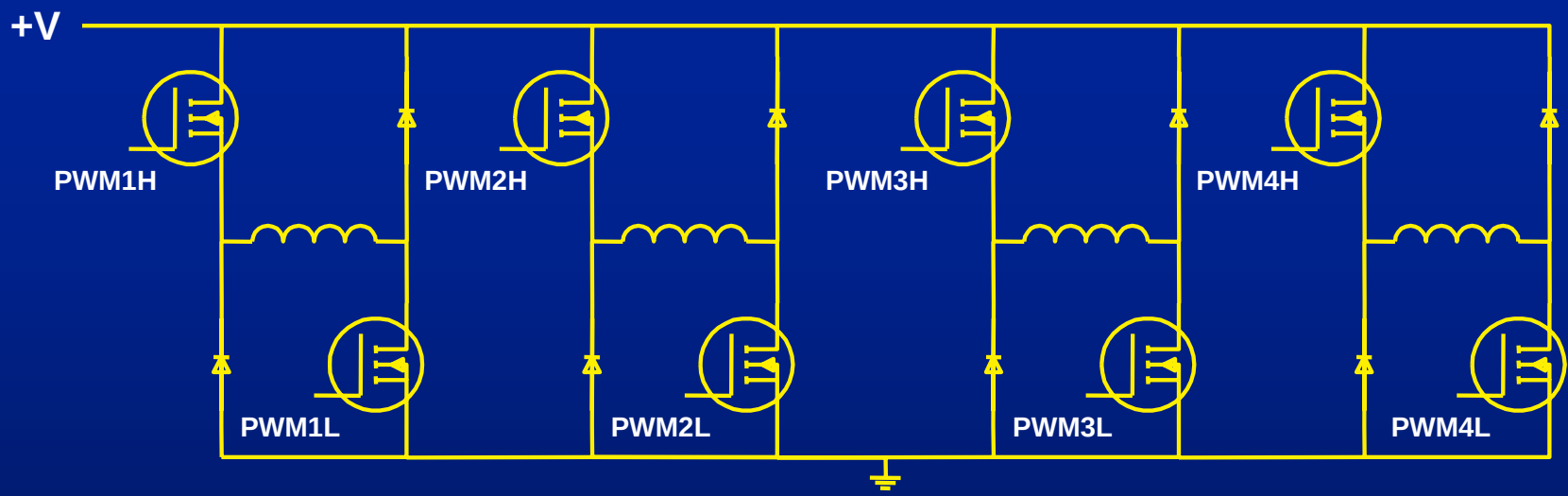
Motor Control PWM

- 3-Phase Inverter Application (Complementary Mode)



Motor Control PWM

- SR Motor Power Stage (Independent Mode)



MCPWM Timebase

- Dedicated 15-bit time-base, period register
  - Up count (edge align), up/down count (center align)
- PWM 的最高解析度爲：**Tcy / 2**
  - PDCx[1..15] 與 PTMR[0..14] 比較，PDC[0] 則與 prescaler 的 MSB 相比
  - 當 CPU 的執行速度 = 20 MIPS，若 PWM 設定爲 11-bit 解析度，PTPER 只需 10 bits 的值 !!
  - PWM 頻率爲 (20 * 10^6)/2^{10} = 19.5 kHz
  - 11 bit resolution 即可提供高於 audible frequencies 的切換頻率
  - PWM 可被設定爲 1-16 次 period match 時產生中斷
- PWM 的 prescaler 選項可爲 1:1, 1:4, 1:16, or 1:64 Tcy

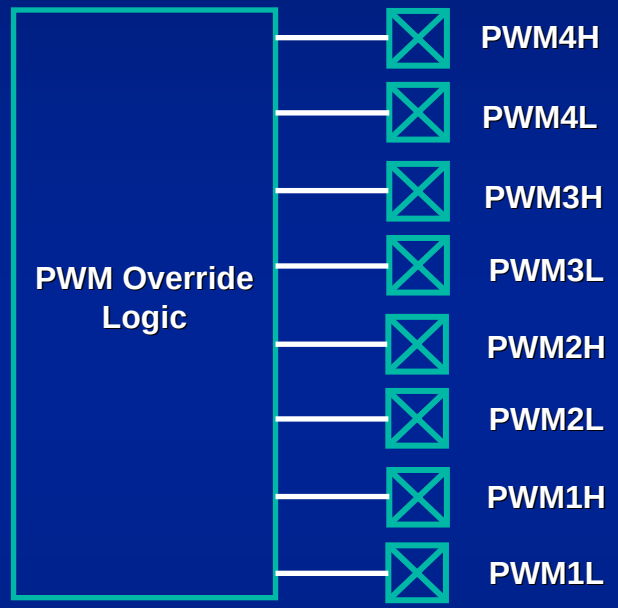
PTCON Register

PTEN	-	PTSIDL	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

PTOPS3	PTOPS2	PTOPS1	PTOPS0	PTCKPS1	PTCKPS0	PTMOD1	PTMOD0
bit7	6	5	4	3	2	1	bit0

MCPWM I/O Control

- 8 I/O pins / 4 pairs
- Pairs can be independent or complimentary mode
- Pins can be enabled as PWM or GPIO
- Configuration bits initialize this register at reset



PWMCON1 Register

-	-	-	-	PMOD4	PMOD3	PMOD2	PMOD1
bit15	14	13	12	11	10	9	bit8

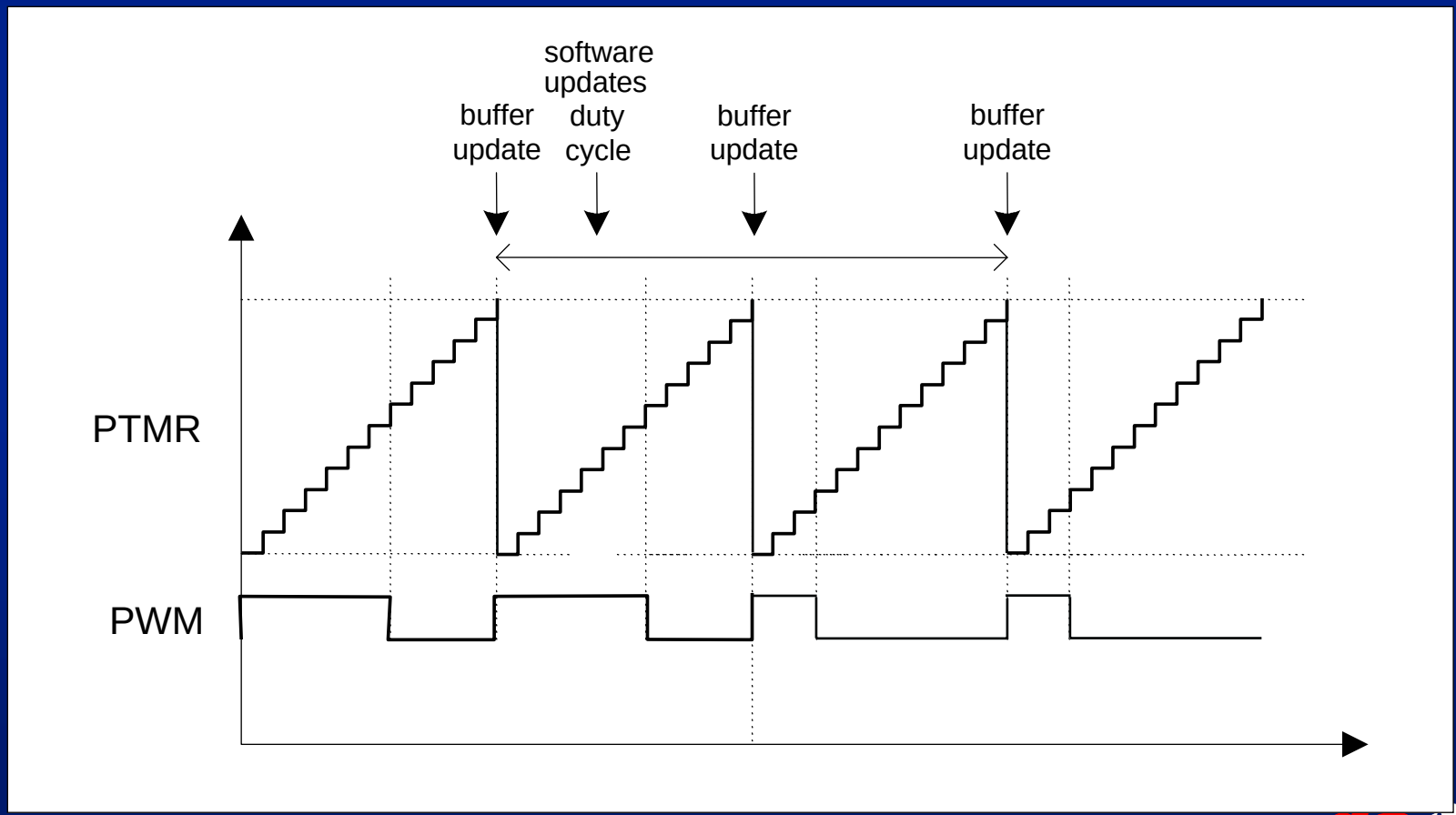
PEN4H	PEN3H	PEN2H	PEN1H	PEN4L	PEN3L	PEN2L	PEN1L
bit7	6	5	4	3	2	1	bit0

MCPWM Period Registers

- 有一個專用的 16-bit period register
- Period registers 具有一個 buffer
  - 下一週期的 period 可隨時被更新而不致影響到現在的週期設定
- Buffers 在 timebase 歸零或 rollover (optionally) 時將被更新

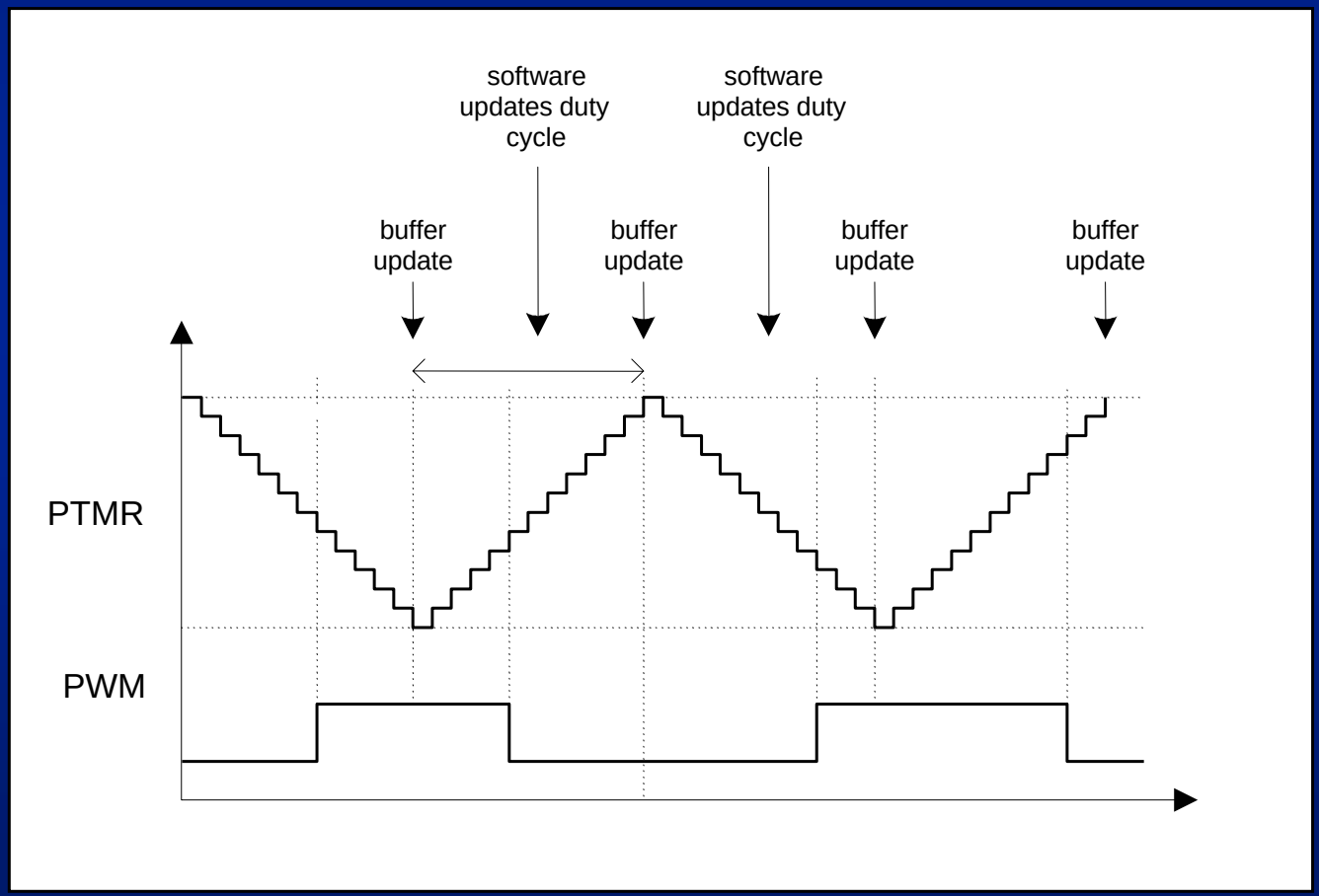
MCPWM Operating Modes

- MCPWM 有兩種對 period 和 duty 的操作模式
  - Edge Aligned PWM (PTPER = PTMR 就馬上歸零)



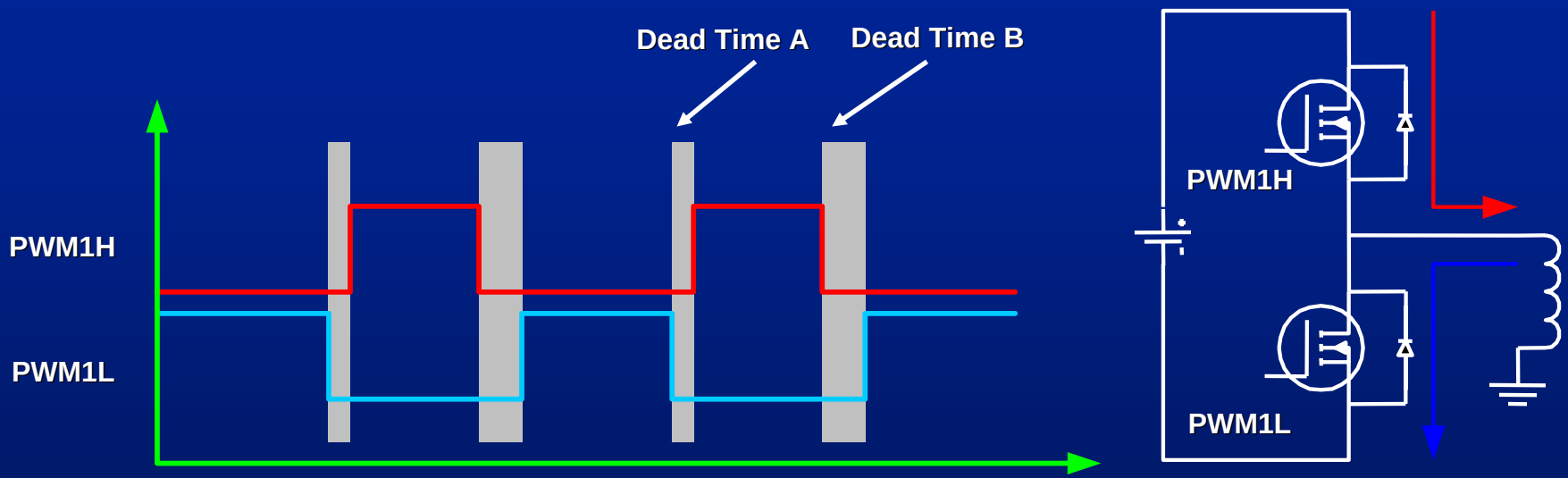
MCPWM Operating Modes

- Center Aligned PWM w/ Double Update
  - (PTPER = PTMR 後不馬上歸零而是向下數)



MCPWM Dead Time Insertion

- Applies only to pin pairs in complimentary mode
- Two programmable dead times
- One dead time per pair for multiple inverters OR
- Two dead times per pair for distortion optimization
- T_{cy} minimum resolution with four pre-scale options



Specifying Dead Time

- DTCON1 用來設定 A，B 兩組 dead time 的時間

DTBPS1	DTBPS0	DTB<5:0>					
bit15	14	13	12	11	10	9	bit8

DTAPS1	DTAPS0	DTA<5:0>					
bit7	6	5	4	3	2	1	bit0

- DTCON2 用來選擇當 PWM 輸出爲 active(A) 或 inactive(I) 的 edge 時要選用哪一組 dead time

-	-	-	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

DTS4A	DTS4I	DTS3A	DTS3I	DTS2A	DTS2I	DTS1A	DTS1I
bit7	6	5	4	3	2	1	bit0

MCPWM Output Override

- 使用於馬達控制時的換流（ motor commutation ）
- 可選擇的驅動方式：Drive PWM，Drive active，inactive
- POVD bits 決定相對應的 I/O pin 是否要自行控制（ 0- manual control) 或由 PWM module (1 – PWM Control)控制
- POUT bits 設定 manual control 時的輸出狀態（ 1=Active，0=Inactive ）
- 在需要 complementary mode 的輸出驅動時，PWM module 的其他控制暫存器可被正確設定以滿足 Dead time 的需求 (DTCON1，DTCON2)

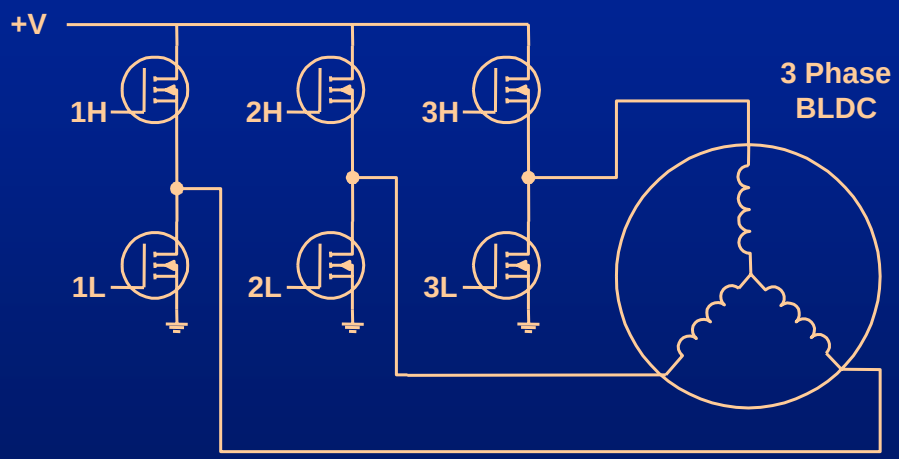
OVDCON Register

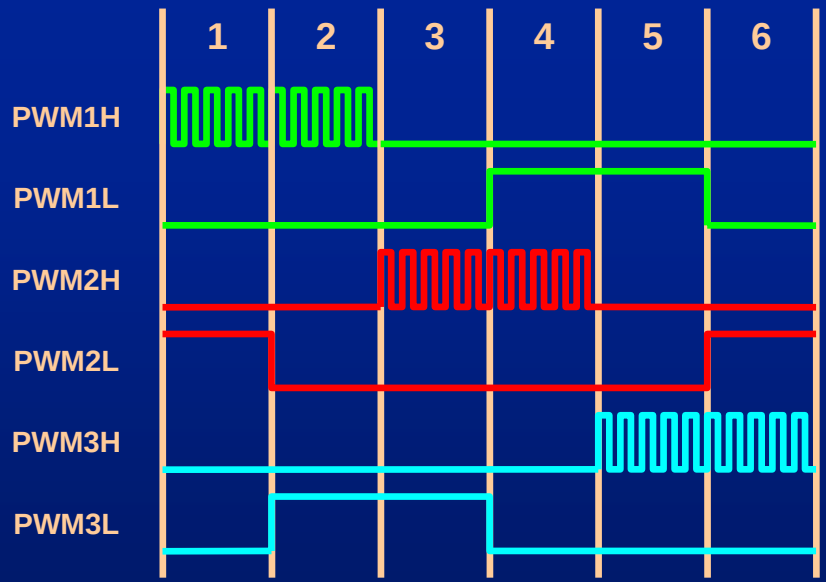
POVD4H	POVD4L	POVD3H	POVD3L	POVD2H	POVD2L	POVD1H	POVD1L
bit15	14	13	12	11	10	9	bit8

POUT4H	POUT4L	POUT3H	POUT3L	POUT2H	POUT2L	POUT1H	POUT1L
bit7	6	5	4	3	2	1	bit0

MCPWM Output Override (2)

- 當 PWM 使用於 6-step modulation 時
  - 依據轉子的位置來對 BLDC 的不同組線圈激磁 (Energize)
  - PWM override register (OVDCON) 用來控制哪些 transistors 要 active 或 inactive，然後使用 duty cycle 來控制電流





MCPWM Fault Inputs

- Two programmable fault pins: Fault A, Fault B
- Fault pins can be assigned to each output pair
- Fault pin asynchronously over-rides PWM output
- Fault state for each output is programmable
- Automatic or latched fault protection
- Interrupt vector for each fault input

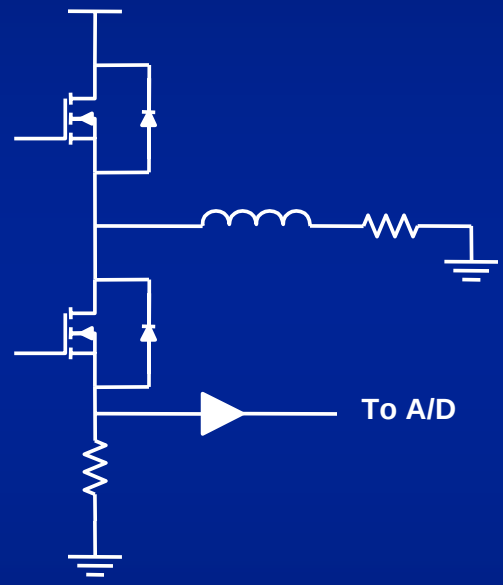
FLTACON Register

FAOV4H	FAOV4L	FAOV3H	FAOV3L	FAOV2H	FAOV2L	FAOV1H	FAOV1L
bit15	14	13	12	11	10	9	bit8

FLTAM	-	-	-	FAEN4	FAEN3	FAEN2	FAEN1
bit7	6	5	4	3	2	1	bit0

MCPWM A/D Synchronization

- SEVTCMP register sets A/D conversion start time
- SEVTDIR bit (bit 15) 設定在 PTMR up 或 down 時用 SEVTCMP 的內容來觸發 ADC
- 如此可確保 A/D 可正確地偵測 shunt current
- 減小 control loop 的 update delay



練習 四
練習設定 Motor Control PWM

- 練習 四 要完成的程式功能 (使用 ex4.c)
  - 完成一個名爲 MotPWM_Initial 的 function，以便檢驗 MCPWM 的基本功能
  - 使用 PWM 來作爲 AD 的觸發來源
    - 所有的 PWM 輸出都設爲 Complement Mode
    - 解析度爲 10 bits (PTPER = 512-1)
    - PWM 設定爲 Edge Alignment Mode
    - 所有 PWM 的 Active 用 dead time A
    - 所有 PWM 的 Inactive 用 dead time A
    - Dead time A 設爲 16 Tcy
    - (dsPIC30F4011 只有 dead time A)

練習 四
練習設定 Motor Control PWM

- 練習 四 要完成的程式功能 (Cont.)
  - 使用 PWM 觸發 ADC，將 VR1 的輸入位準顯示於 LCD
  - VR1 由 AN7 來轉換 (JP27 = 1&2)
  - APP009 的 Crystal 爲 7.3728 Mhz
  - Configuration Bits 中的 Osc. Mode 選爲 XT w/PLL 8x
  - 在示波器上將看到 28.8K 的 PWM 信號
    - (7372800 * 2) /512 = 28800
  - PTPER 的計算式：
    - PTPER = (Fcy/(FPWM*PTMR Prescaler)) - 1

練習 五
使用 Motor Control PWM

- 練習 五 要完成的程式功能
  - 使用 Motor Control PWM 產生能驅動 AC Motor 的信號
  - 解析度：最少 10-bit
  - 切換頻率：20 KHz
  - Target AC Frequency：60 Hz
  - 不考慮 加/減速 的操作

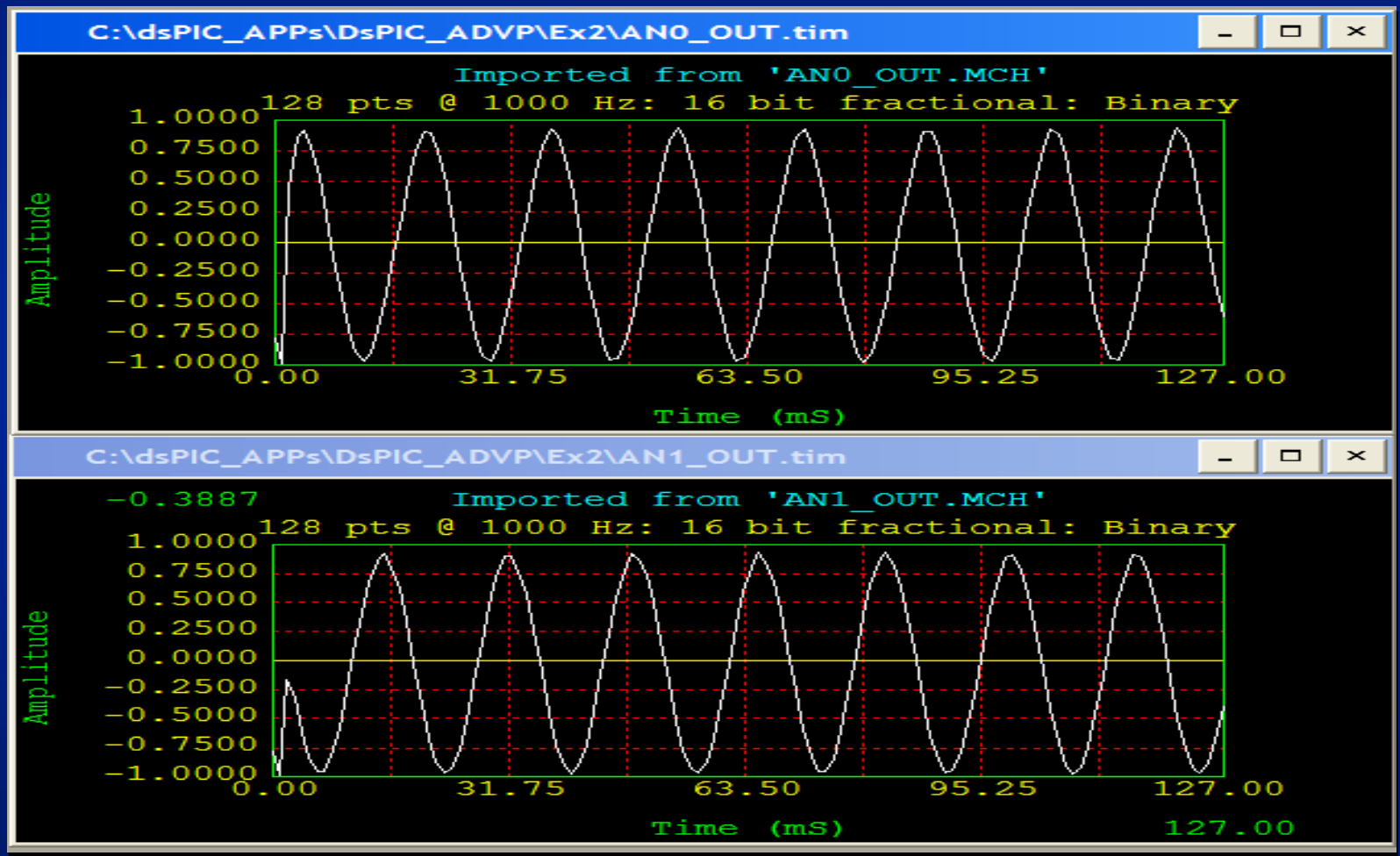
練習 五
使用 Motor Control PWM

- 有關練習 五 的一些提示
  - Fcy = 7372800 * 2 (XT w/PLL 8x)
  - PTPER = (Fcy/(FPWM*PTMR Prescaler)) – 1
    - PTPER = (14745600/(20000*1)) – 1 =736.28 (取整數值 736)
    - PDCx 的範圍：0 to 1473 (737 * 2)
    - 解析度將會 > 10-bit (0 to 1023)
  - 角度的計算
    - 60 Hz = 1S/60 = 1666.66 us
    - 每 1666.66 us 要完成 0^0 to 360^0 的 Duty Update
    - 1666.66 * 7.3728 * 2 = 245760 (Tcy 數量)
    - 245760 / 120 = 2048 (PR1 的設定值)
    - 每次 Timer1 interrupt 時，將輸出信號增加 3^0
  - 利用 APP012，將可於示波器上觀察到 3 組相差 120^0 的弦波輸出

練習 六
使用 10-bit ADC 提供的同步取樣/轉換功能

- 練習六
  - 使用 Timer3 爲 AD Conversion 的 Trigger Source
  - Timer3 需設定爲每 1ms 觸發 AD 轉換一次
  - 將 APP012 上的三個弦波信號用連接線與 AN0，AN1，AN2 相連
  - 使用 10-bit ADC 的同步取樣功能 !!
  - ADC 的中斷程式將 AN0，AN1，AN2 的轉換結果存入
    - AN0_Buf[ ]
    - AN1_Buf[ ]
    - AN2_Buf[ ]
  - 中斷點設於 Anx_Buf full 的地方（轉換 128 次）
  - 將結果 Export 並用 dspWORK 觀察結果

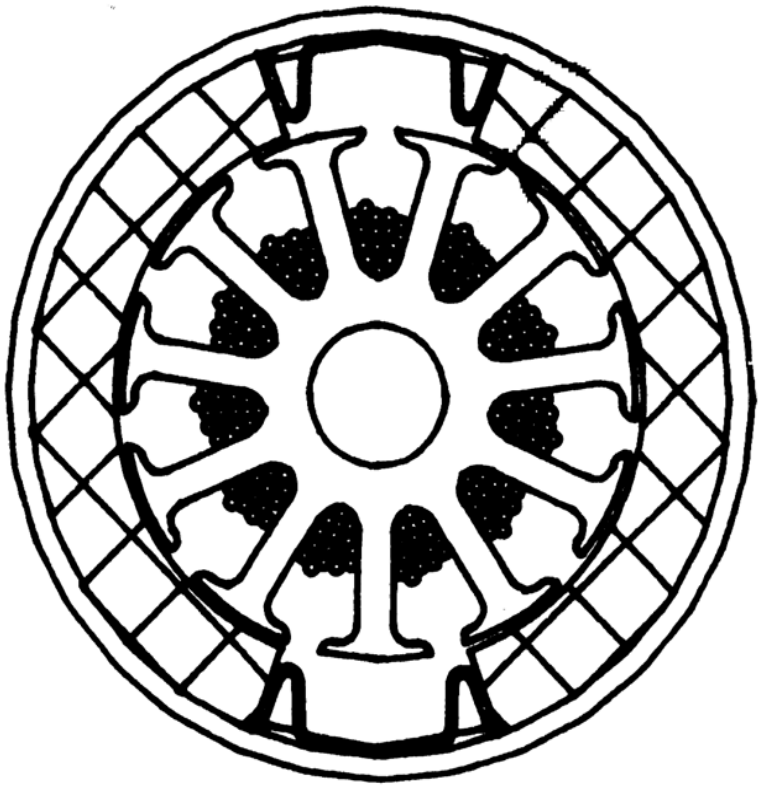
練習 六
使用 10-bit ADC 提供的同步取樣/轉換功能



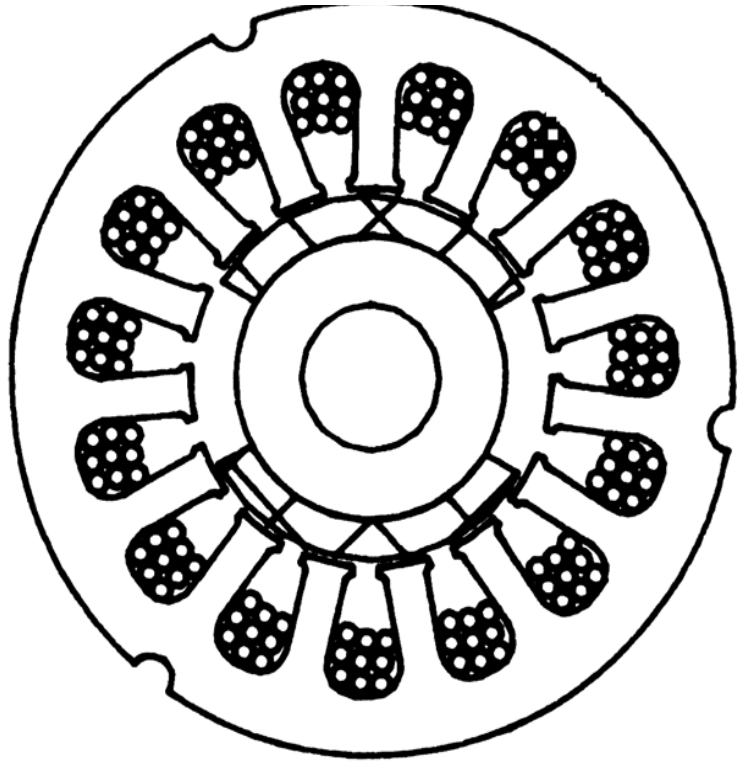
BLDC Motor Basics

Brushed & Brushless DC Motor Construction

PERMANENT MAGNET BRUSHED DC MOTOR



PERMANENT MAGNET BRUSHLESS DC MOTOR



BLDC Advantages Over Brushed DC Motor

- 在能提供一樣的輸出功率條件下，BLDC 的尺寸較小. 相對的電樞的散熱效果也較好
- 可以操作至較高的轉速.
- 因爲沒有轉子上的繞組以及所需的換向器，故慣性較低.
- 具有比較好的加速性
- 不需要維護換向所需的碳刷.
- 沒有因換向時產生的火花.

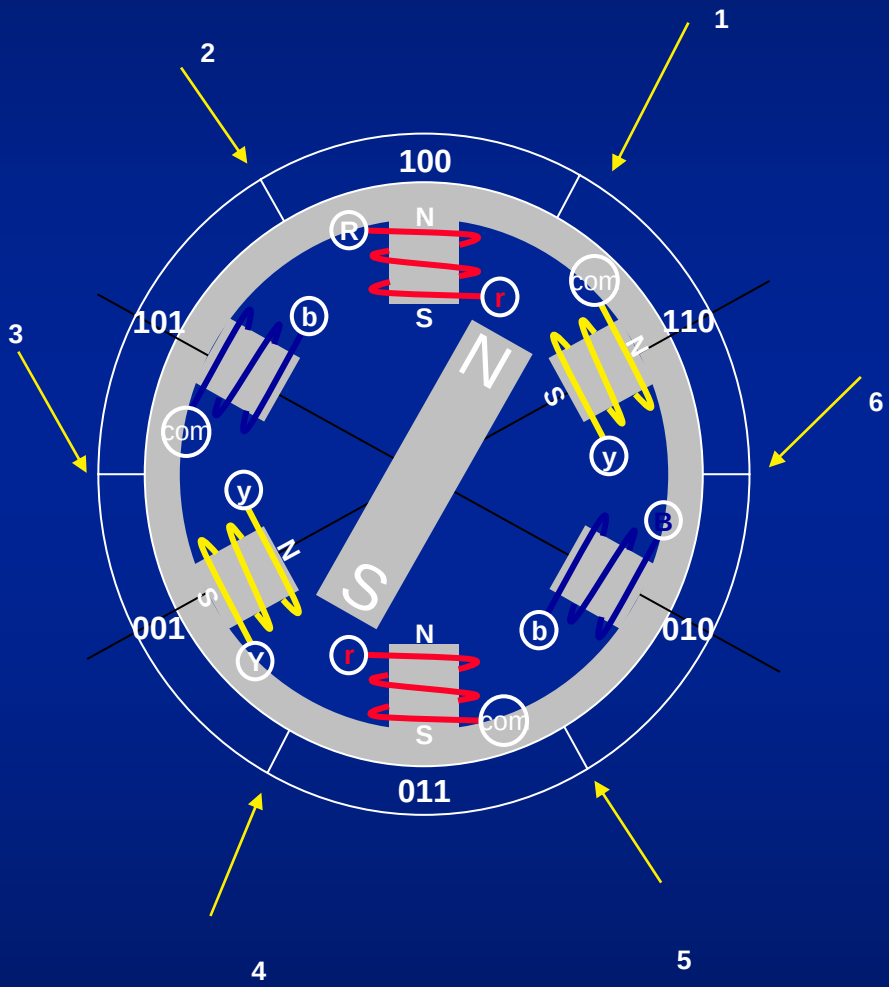
BLDC Control

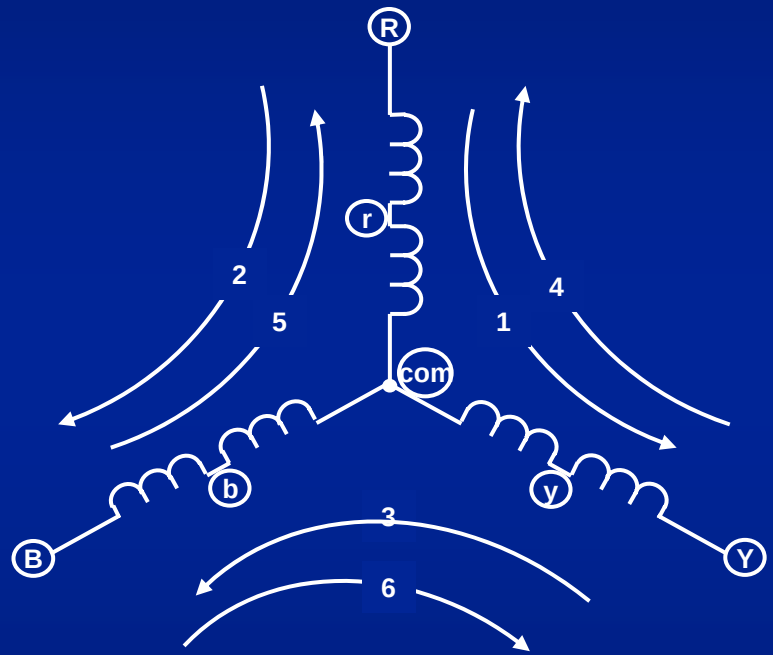
- 使用電力電子的換流方式來改變定子磁場方向, 取代在傳統 DC 馬達上使用機械式換流的方式.
- 磁場交換的動作必須正確地與轉子的位置同步以確保運轉的順暢與有效率地舛生轉矩BLDC 屬於同步式的馬達構造, 不需要額外的 damping 或 starting 的繞組.

Standard BLDC Position Sensing

- 通常將一個位置檢出用的圓盤附加於轉子上，使其提供與轉子磁極相對稱的且 duty 約爲 50% 的 pattern 輸出. 這些位置信號的重複率則與轉子的極數對稱
- 將此圓盤提供的 Pattern 由三個 Optical 或 Hall Sensoes 或來檢知進而轉換爲位置信號. 一般安排的方式有 120° 及 60° 間隔的排列於定子(Stator)上 .
- 若使用 Hall sensors, 轉子上磁極位置可被直接檢出.

BLDC 的結構與激磁順序 – 產生逆時針方向的旋轉磁場 (一組三相繞組的示意圖)

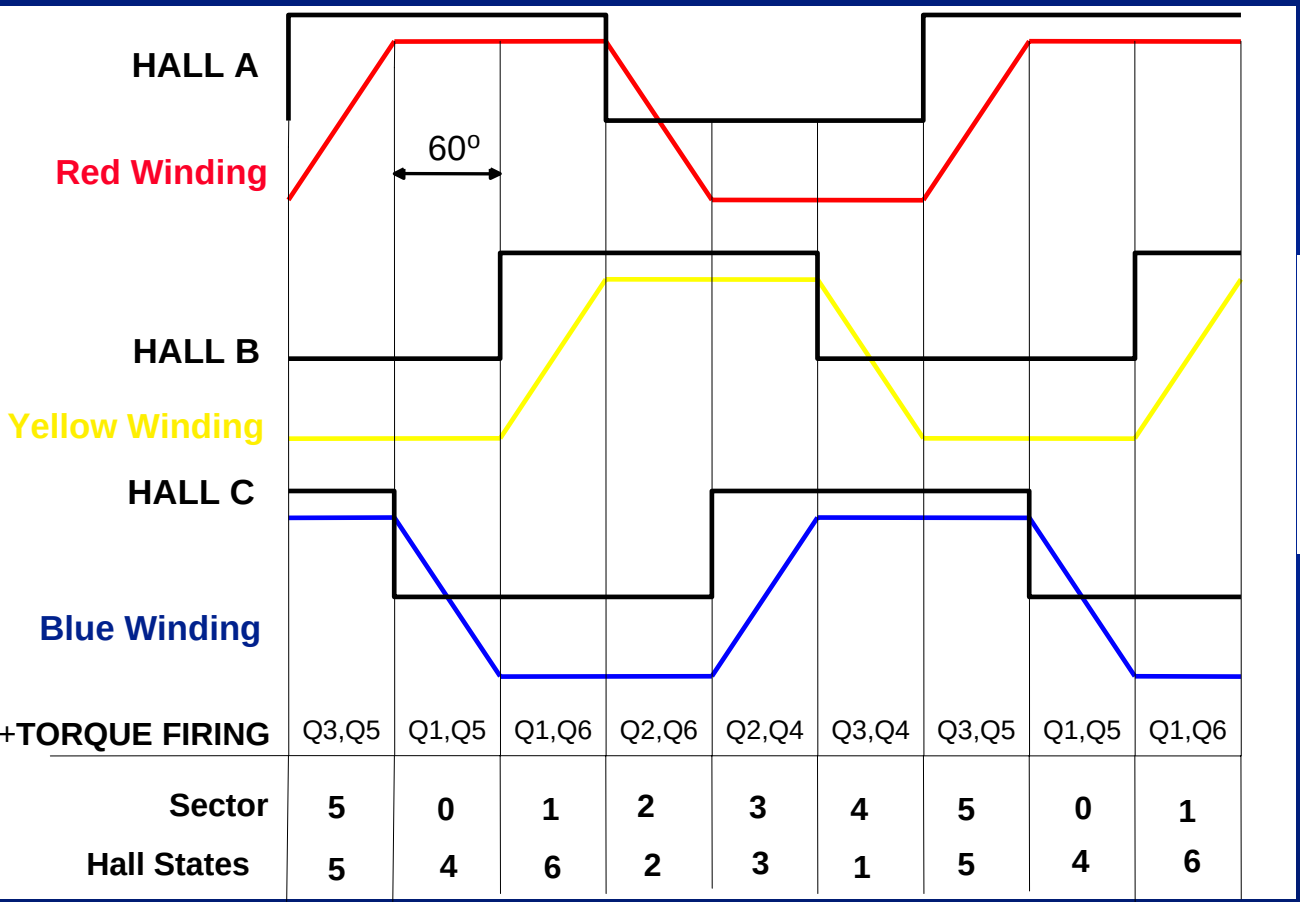




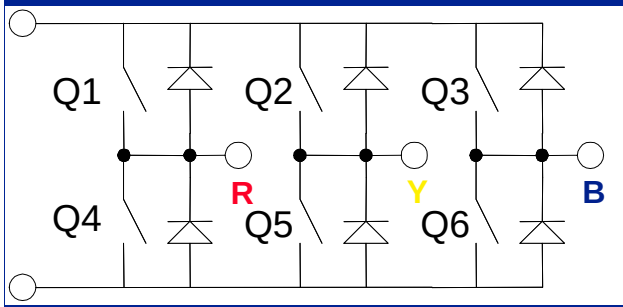
Standard Sensored BLDC Control

- 由前頁的說明，BLDC 的激磁方向與轉子位置必須配合
- 三個由 HALL Sensors 傳來的位置信號被用來決定線圈被激磁的方向與動作
- 由三個 Sensors 產生的組合,邏輯上因該有八種，但在實際的位置安排下, 有兩種狀態是不可能被產生的 (000, 111).
- 使用簡單的查表法, 即可決定哪些線圈要被驅動爲 DC+ ，DC- 或者是不激磁.
- 使用六個不同的有效狀態, 就直接對應到 6 個 60^0 的 Electrical Cycle sector (與機械角度的關係視極數決定).
- 這些 HALL Sensor 的狀態需要被正確的檢測而且順序必須正確，狀態的不正常置換及不合法狀態都必須被檢知以達到正確的控制

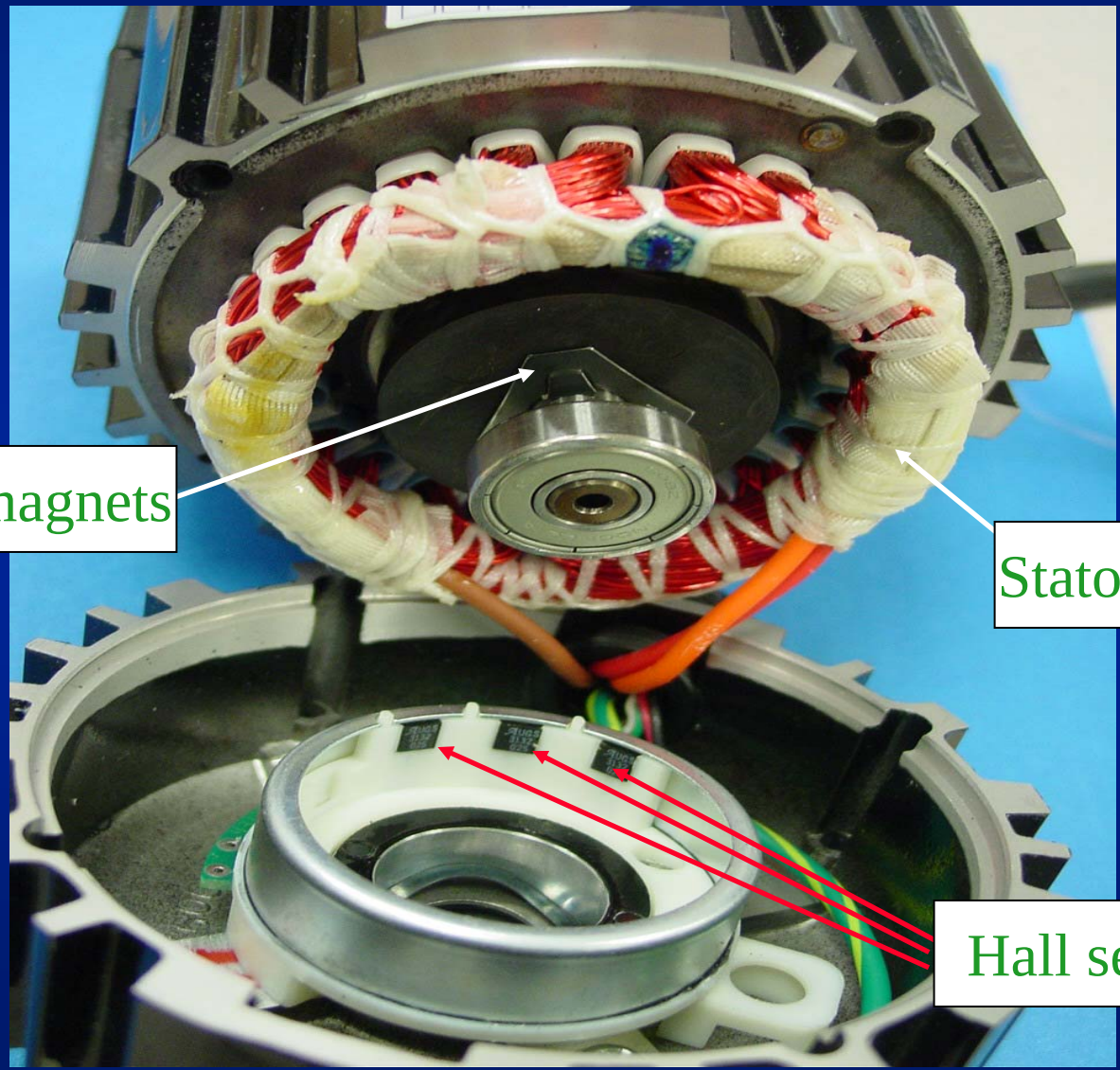
Standard BLDC Control (反轉)



+TORQUE FIRING	Q3,Q5	Q1,Q5	Q1,Q6	Q2,Q6	Q2,Q4	Q3,Q4	Q3,Q5	Q1,Q5	Q1,Q6
Sector	5	0	1	2	3	4	5	0	1
Hall States	5	4	6	2	3	1	5	4	6



BLDC Motor Construction



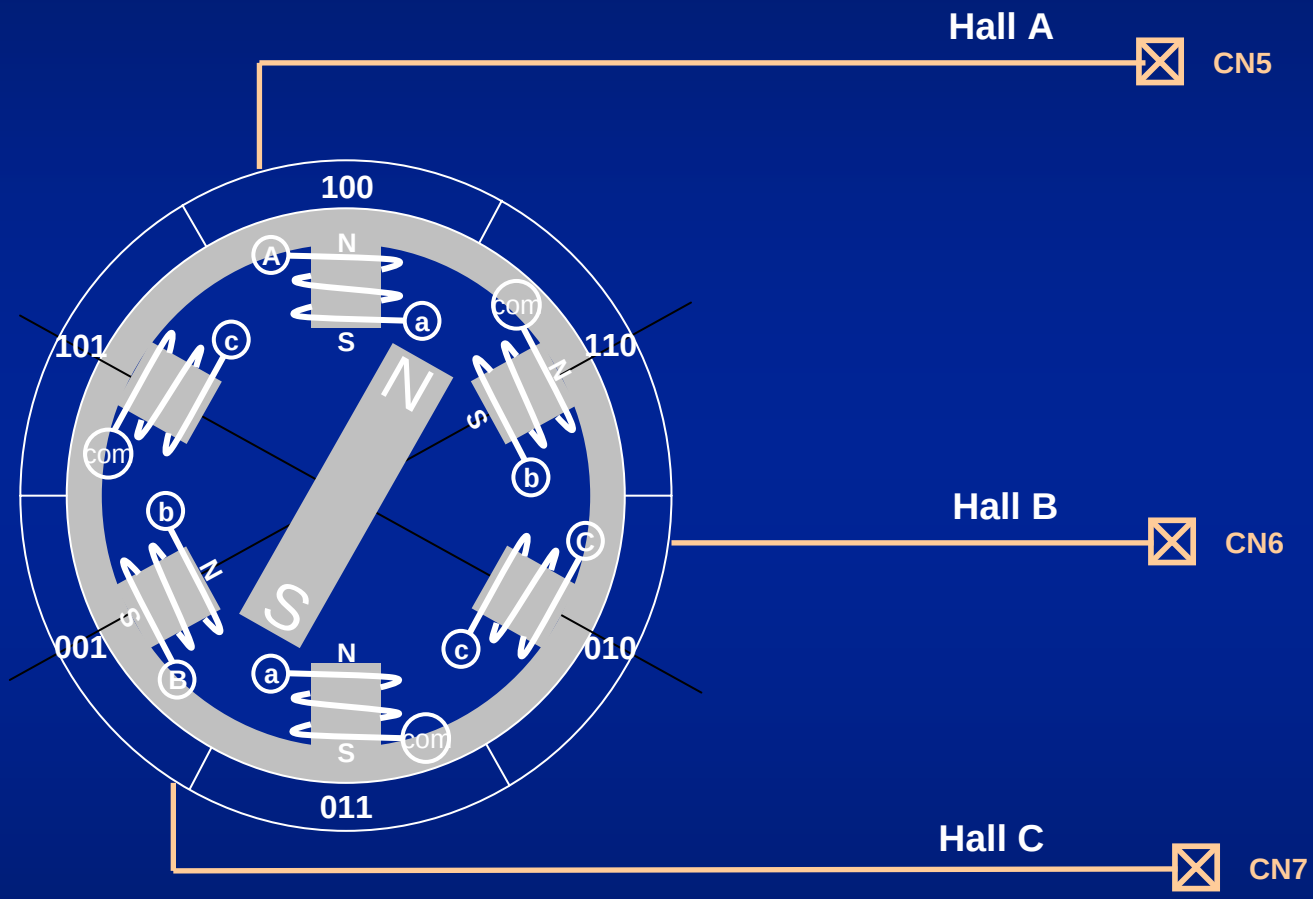
Control of Sensored BLDC

- 使用 HALL Sensor 控制 BLDC 的方式
  - 偵測位置的變化發生點
  - 讀取 HALL Sensor 來得知轉子的位置
  - 使用查表法來決定線圈換流的動作和方向
  - 使用 PWM 來達成速度控制
- 若要達到速度控制：
  - P , PI or PID
    - PWM_next = PWM_old + PID Error
    - Or PWM out = PID Error (KI != 0 or Kd != 0)
    - PID Error = Kp * E + KI * E(sum) + Kd * E(delta)

Change Notification (CN)

- dsPIC® DSC 具有偵測 Change Notification 的功能:
  - 偵測指定接腳上數位位準的變化並且產生中斷
    - CNInterrupt
  - Hall sensors A, B 和 C 分別被連接至RB3(CN5), RB4(CN6) 與 RB5(CN7).
  - 當 CNInterrupt 中斷發生, Hall inputs 可以被讀取然後用來作爲 lookup table 的索引值來取出要控制 BLCD 的信號組合

CN Hardware



MCPWM Output Override

- 使用於馬達控制時的換流（ motor commutation ）
- 可選擇的驅動方式：Drive PWM，Drive active，inactive
- POVD bits 決定相對應的 I/O pin 是否要自行控制（ 0- manual control) 或由 PWM module (1 – PWM Control)控制
- POUT bits 設定 manual control 時的輸出狀態（ 1=Active，0=Inactive ）
- 在需要 complementary mode 的輸出驅動時，PWM module 的其他控制暫存器可被正確設定以滿足 Dead time 的需求 (DTCON1，DTCON2)

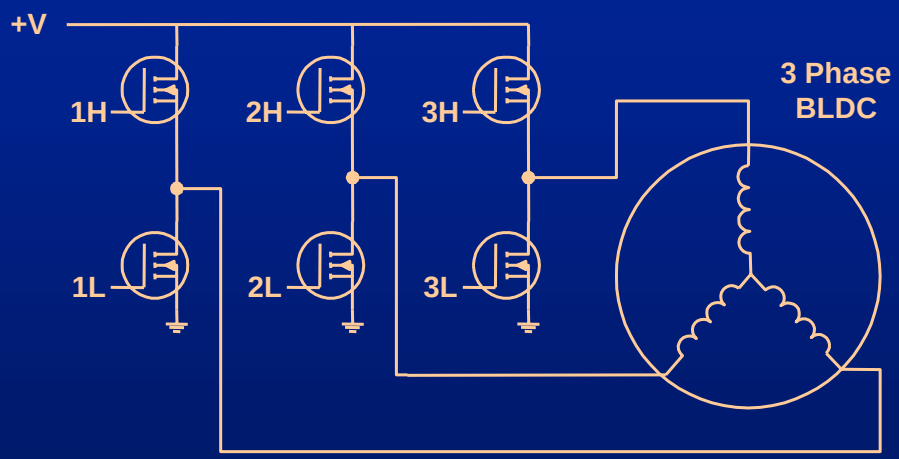
OVDCON Register

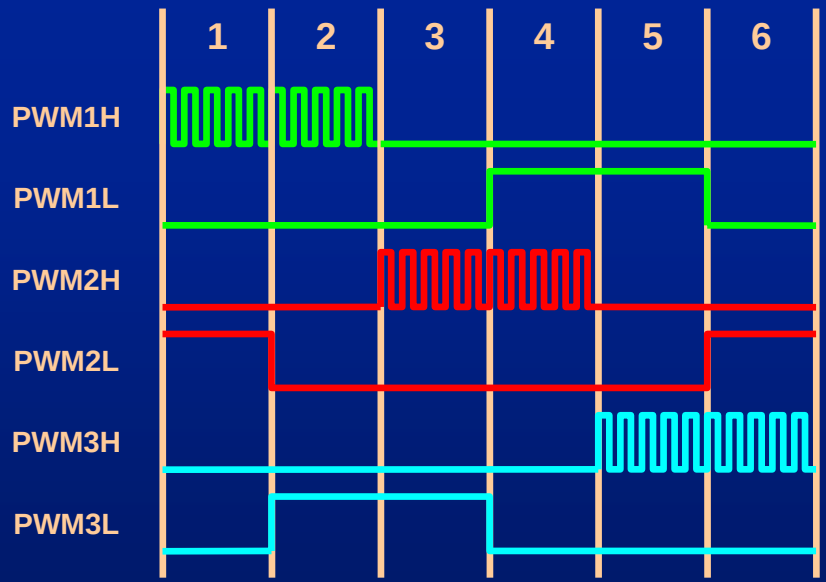
POVD4H	POVD4L	POVD3H	POVD3L	POVD2H	POVD2L	POVD1H	POVD1L
bit15	14	13	12	11	10	9	bit8

POUT4H	POUT4L	POUT3H	POUT3L	POUT2H	POUT2L	POUT1H	POUT1L
bit7	6	5	4	3	2	1	bit0

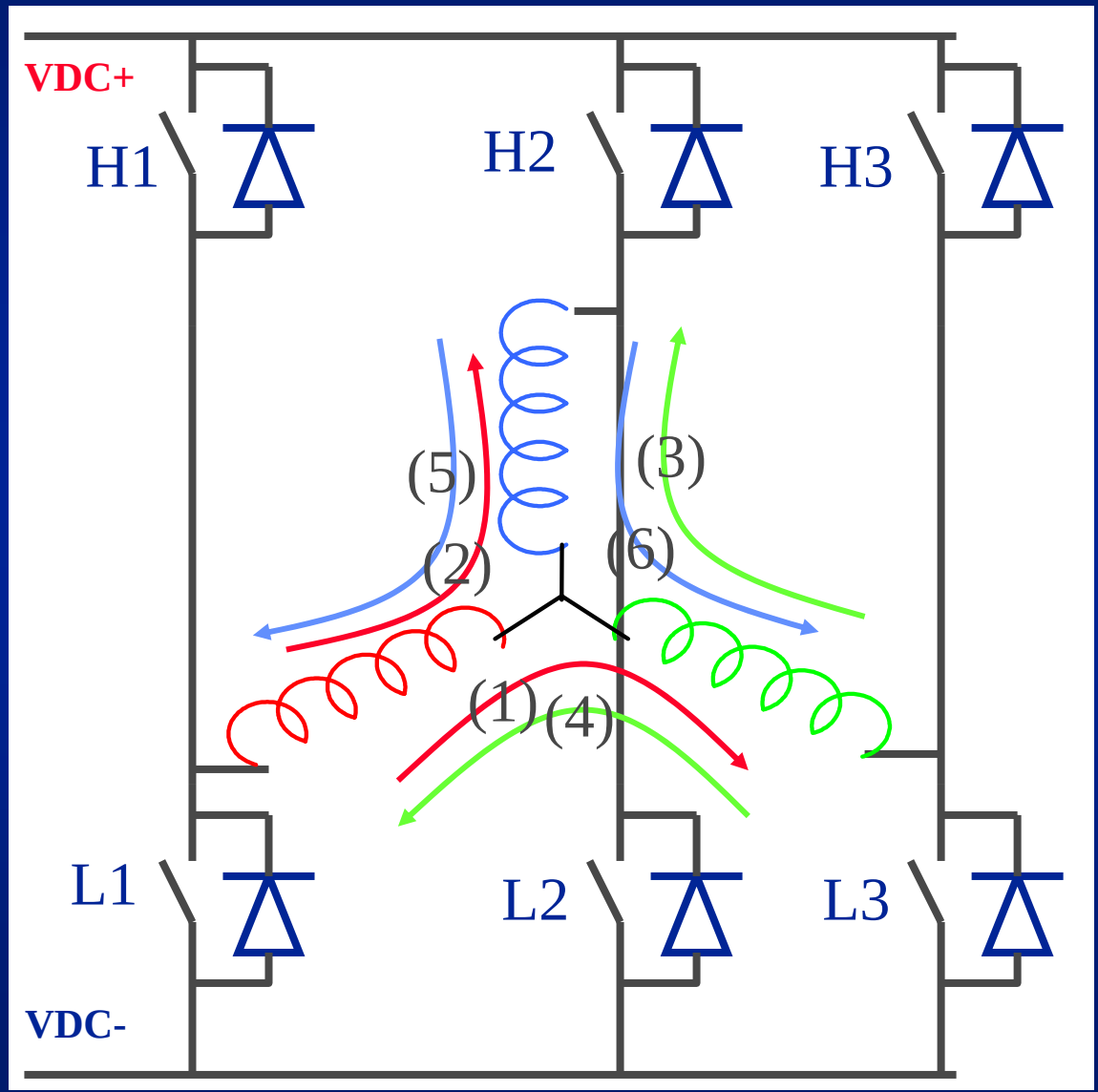
MCPWM Output Override (2)

- 當 PWM 使用於 6-step modulation 時
  - 依據轉子的位置來對 BLDC 的不同組線圈激磁 (Energize)
  - PWM override register (OVDCON) 用來控制哪些 transistors 要 active 或 inactive，然後使用 duty cycle 來控制電流

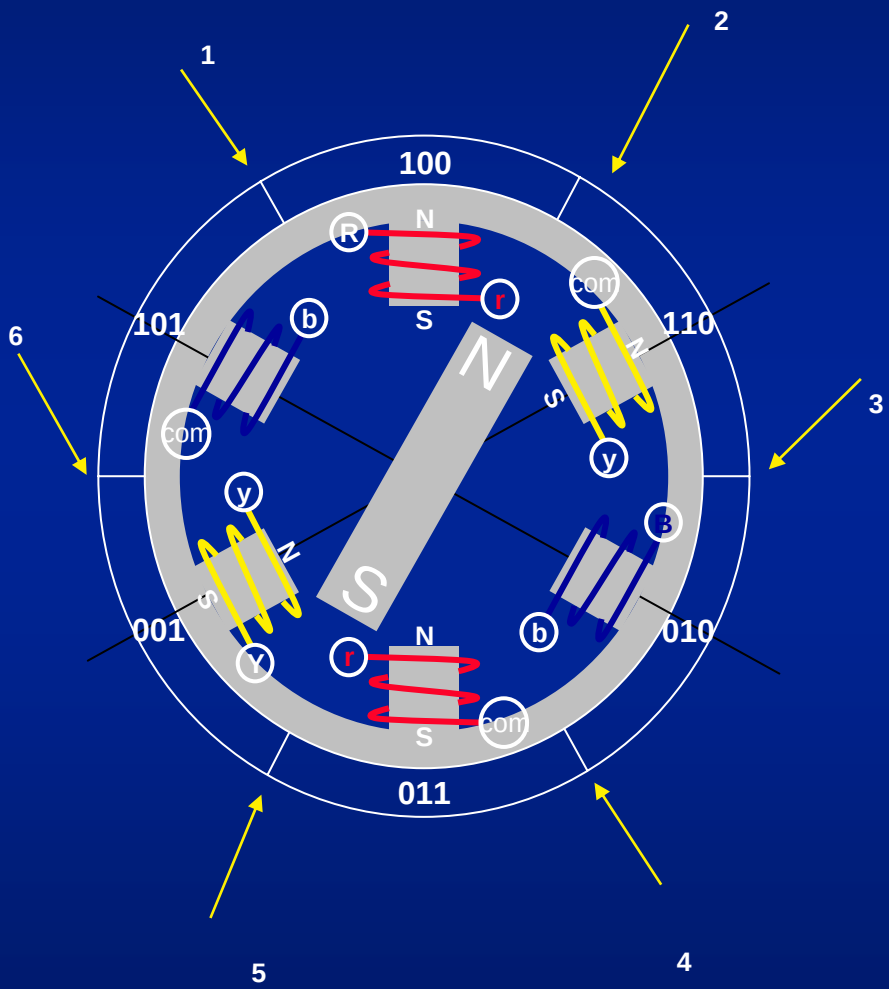


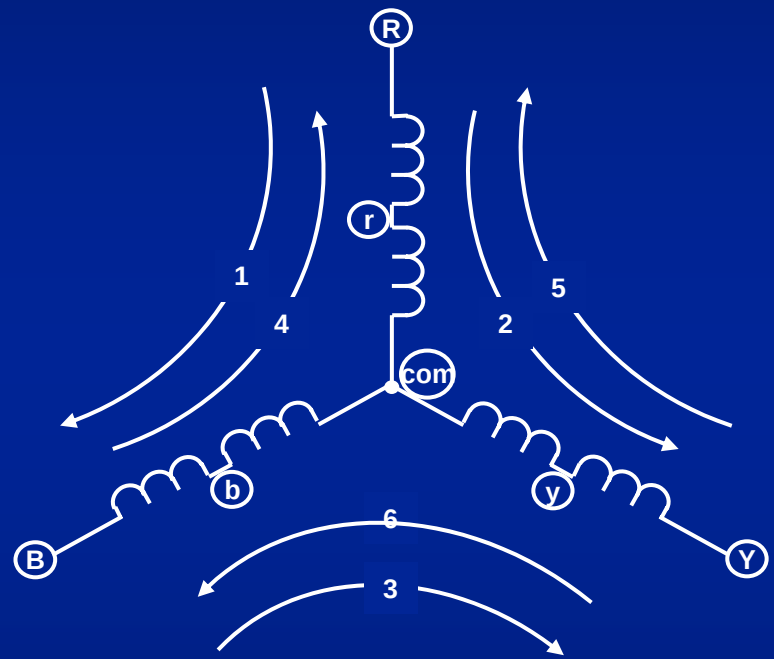


Drive Sequence (正轉)

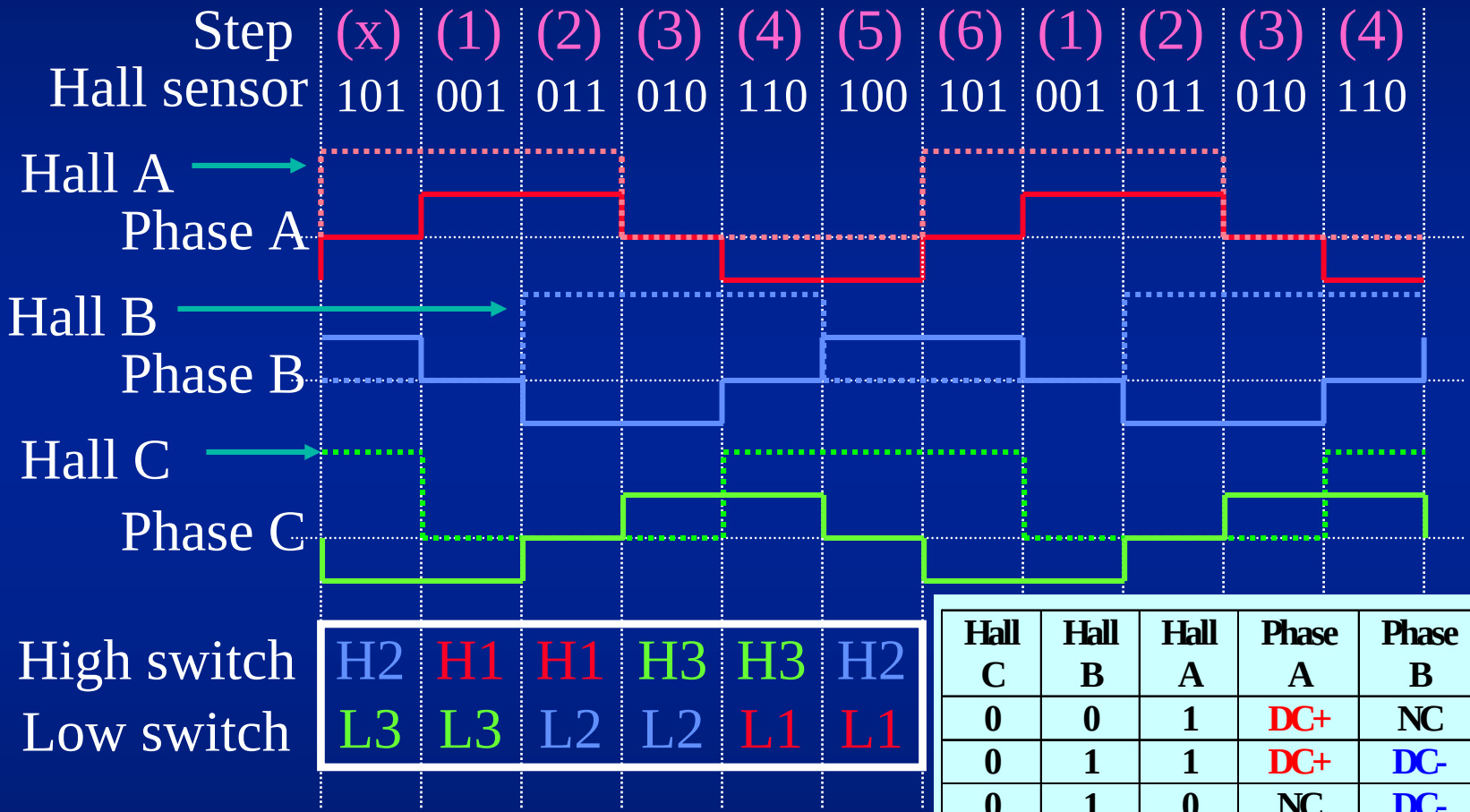


BLDC 的結構與激磁順序 – 產生順時針方向的旋轉磁場 (一組三相繞組的示意圖)





Switching Sequence



Hall C	Hall B	Hall A	Phase A	Phase B	Phase C
0	0	1	DC+	NC	DC-
0	1	1	DC+	DC-	NC
0	1	0	NC	DC-	DC+
1	1	0	DC-	NC	DC+
1	0	0	DC-	DC+	NC
1	0	1	NC	DC+	DC-

Motor Control PWM

- 使用 OVDCON 來控制 BLDC 換流的動作，只需一個指令即可完成一個 Step 的切換及控制

Step	**OVDCON** Value	
	POVD<7:0>	POUT<7:0>
1	00000010	00010000
2	00000010	00000100
3	00100000	00000100
4	00100000	00000001
5	00001000	00000001
6	00001000	00010000

1 Electrical Revolution

Hall C	Hall B	Hall A	Phase A	Phase B	Phase C
0	0	1	DC+	NC	DC-
0	1	1	DC+	DC-	NC
0	1	0	NC	DC-	DC+
1	1	0	DC-	NC	DC+
1	0	0	DC-	DC+	NC
1	0	1	NC	DC+	DC-

練習 七
以 HALL Sensors 的位置來設定 PWM 輸出信號

- 參考前面所提及之 HALL Sensor 與 BLDC 激磁的對應表，完成練習五的程式
  - PWM 的輸出狀態，皆設定為 Active “HIGH”
  - 使用 High Site Switching
  - 使用 CN5，CN6，CN7 來偵測 HALL Sensor 的變化
  - 當 HALL Sensor 轉態時，將適當的值填入 OVDCON 中
  - 要填入 OVDCON 的值存放於陣列 SW_HiTable[]
    - Programmer 要將正確值填入
  - 使用 HALL Generator 程式(另一片事先寫好程式的測試版）來測試產生的輸出是否正確

練習 七
以 HALL Sensors 的位置來設定 PWM 輸出信號

- 測試的步驟：
  - HALL Generator 會參考 S1 的輸入狀態，依序產生六個 HALL Sensor Position Signal
  - 練習五的程式依照 HALL Sensor 的位置產生對應的驅動訊號於 PWMxH & PWMxL
  - HALL Generator 將於 LCD 上顯示六個驅動訊號的狀態，依序爲：
    - PWM1L，PWM1H，PWM2L，PWM2H,PWM3L，PWM3H
  - 若顯示值爲 X，表示 Switching
  - 若顯示值爲 0，表示 Inactive
  - 若顯示值爲 1，表示 Active

練習 八
使用 LOW POWER MODULE 來驅動 BLDC

- 將修改過的練習七加入以下功能後，讓 APP009 能值驅動 BLDC
  - 使用 VR1 的轉速命令
  - 使用 S1 & S2 的啓動及停止命令
- BLDC Motor 信號線的顏色及定義
  - 請參考下一頁

練習 八
使用 LOW POWER MODULE 來驅動 BLDC

- BLDC Motor 信號線的顏色及定義
  - HALL Sensor Connection
    - RED : +5V
    - BLACK : GND
    - WHITE : HALL A
    - BROWN : HALL B
    - GREEN : HALL C
  - Winding Connection
    - RED : R (A or M1)
    - BLACK : Y (B or M2)
    - WHITE : B (C or M3)
    - GREEN : GND

Thank you

免付費技術服務專線電話

0800-717-718