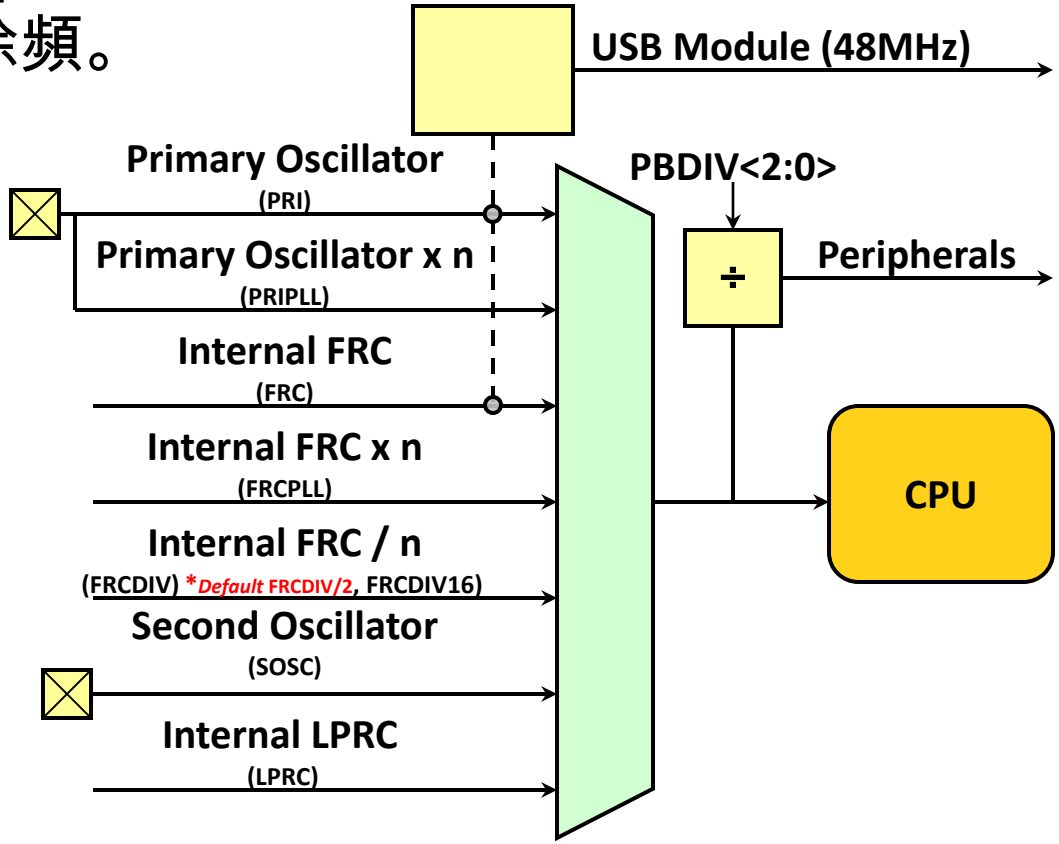
MICROCHIP

Regional Training Centers

Section 7
Oscillator and Configuration Bits

PIC32MX Oscillator Architecture

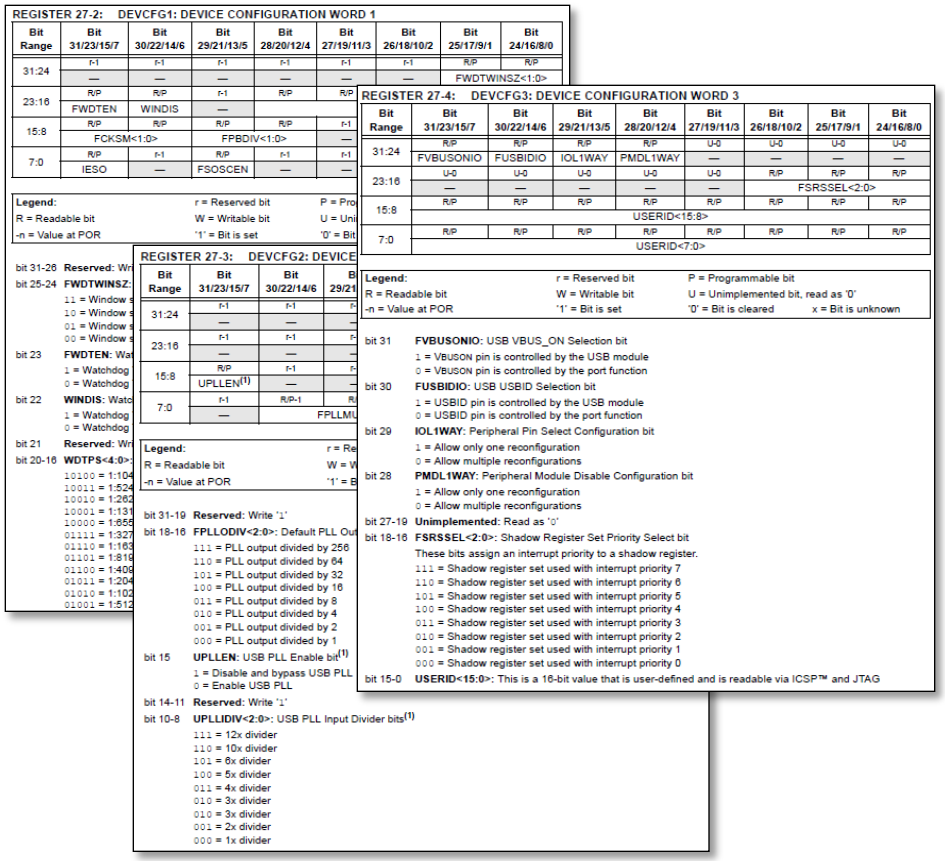
- 這圖應該還有印象, 前面有看過一次了。 PIC32具有多種時脈來源可選擇。可以使用內部RC或者外部的Crystal, Oscillator。輸入的時脈也可透過內部電路進行倍頻(PLL)或者除頻。
- 時脈來源的設定, 必須正確。如果設定錯誤, CPU會接收到錯誤的頻率, 導致程式運作上的時序不正確。或者完全無法運作。
- 三組時脈, 分別提供CPU, 周邊模組(Timer, UART, ADC, etc..)以及USB模組使用。



MICROCHIP
Regional Training Centers

Configuration Bit

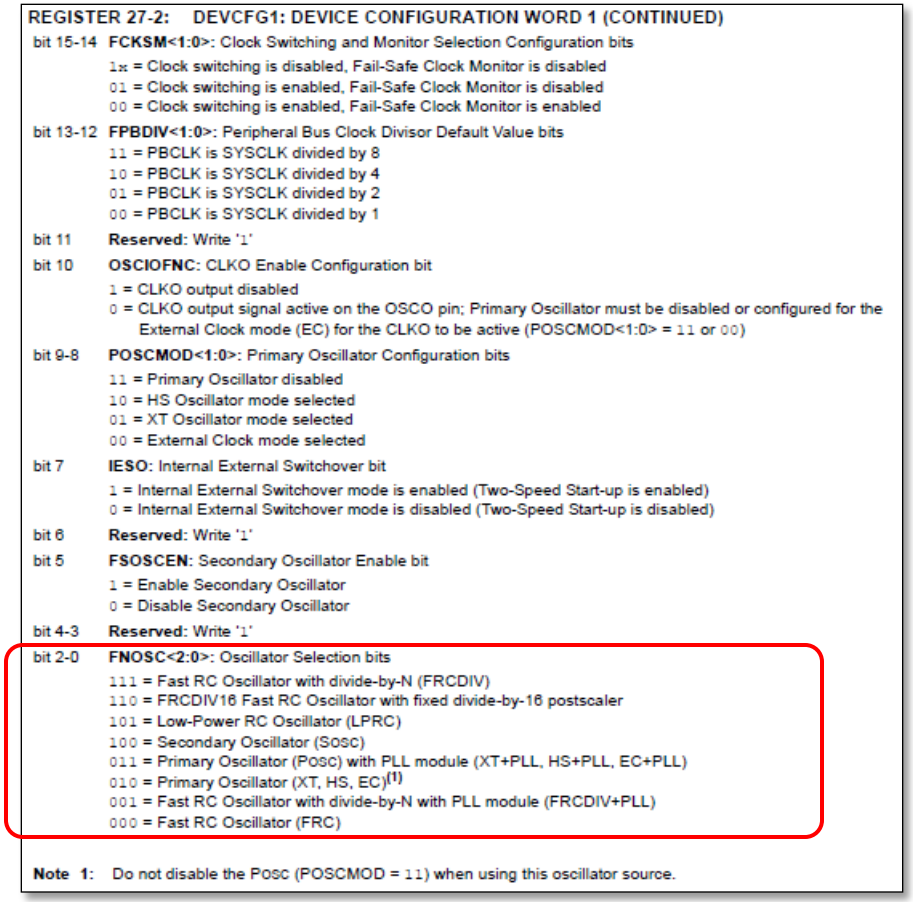
- Configuration Bit是用來設定MCU中一些重要的模式。其中包含如:CPU的時脈來源, 除錯工具使用的接腳,Watchdog, 程式碼保護, 程式碼防寫等設定。
- 其中又以CPU的時脈來源, 除錯工具使用的接腳設定最重要, 時脈來源設定錯誤CPU無法正常運作, 除錯工具使用的接腳設定錯誤, 則無法使用硬體除錯工具。
- 參考Datasheet的Special Feature章節可以看到完整的說明。



XC32 Configuration Bits Function

- 不同系列的Configuration Bits設定方式是一樣的,但使用的Function Name不同。必須搭配參照Datasheet及XC32的說明文件來設定。
- 以PICF32MX450F256H為例,要設定為時脈來源為內部FRC可使用以下方法:

  #pragma config FNOSC = FRC



REGISTER 27-2: DEVCFG1: DEVICE CONFIGURATION WORD 1 (CONTINUED)

bit 15-14 **FCKSM<1:0>**: Clock Switching and Monitor Selection Configuration bits
1x = Clock switching is disabled, Fail-Safe Clock Monitor is disabled
01 = Clock switching is enabled, Fail-Safe Clock Monitor is disabled
00 = Clock switching is enabled, Fail-Safe Clock Monitor is enabled

bit 13-12 **FPBDIV<1:0>**: Peripheral Bus Clock Divisor Default Value bits
11 = PBCLK is SYSCLK divided by 8
10 = PBCLK is SYSCLK divided by 4
01 = PBCLK is SYSCLK divided by 2
00 = PBCLK is SYSCLK divided by 1

bit 11 **Reserved**: Write '1'

bit 10 **OSCIOFNC**: CLKO Enable Configuration bit
1 = CLKO output disabled
0 = CLKO output signal active on the OSCO pin; Primary Oscillator must be disabled or configured for the External Clock mode (EC) for the CLKO to be active (POSCMOD<1:0> = 11 or 00)

bit 9-8 **POSCMOD<1:0>**: Primary Oscillator Configuration bits
11 = Primary Oscillator disabled
10 = HS Oscillator mode selected
01 = XT Oscillator mode selected
00 = External Clock mode selected

bit 7 **IESO**: Internal External Switchover bit
1 = Internal External Switchover mode is enabled (Two-Speed Start-up is enabled)
0 = Internal External Switchover mode is disabled (Two-Speed Start-up is disabled)

bit 6 **Reserved**: Write '1'

bit 5 **FSOSCEN**: Secondary Oscillator Enable bit
1 = Enable Secondary Oscillator
0 = Disable Secondary Oscillator

bit 4-3 **Reserved**: Write '1'

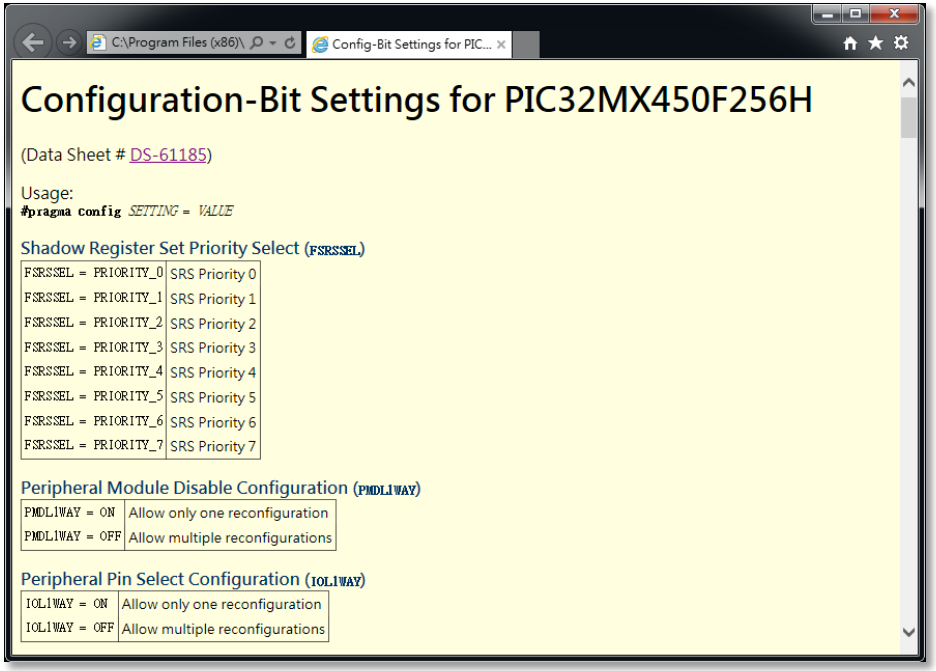
bit 2-0 **FNOSC<2:0>**: Oscillator Selection bits
111 = Fast RC Oscillator with divide-by-N (FRCDIV)
110 = FRCDIV16 Fast RC Oscillator with fixed divide-by-16 postscaler
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (SOSC)
011 = Primary Oscillator (POSC) with PLL module (XT+PLL, HS+PLL, EC+PLL)
010 = Primary Oscillator (XT, HS, EC)(1)
001 = Fast RC Oscillator with divide-by-N with PLL module (FRCDIV+PLL)
000 = Fast RC Oscillator (FRC)

Note 1: Do not disable the POSC (POSCMOD = 11) when using this oscillator source.

XC32 Configuration Bits Function

- MPLAB XC32可以在程式碼中使用特殊關鍵字*#pragma config*來設定Configuration Bit。
  利用 *#pragma config* 在程式裡來設定,語法如下例所示:
  #pragma config UPLLIDIV = DIV_2 , UPLLEN = ON
  #pragma config FPLLIDIV = DIV_2 , FPLLMUL = MUL_15 , FPLLODIV = DIV_1

- *config* 的有效定義字與意義可參考說明文件:
  C:\Program Files (x86)\Microchip\xc32\v1.30\docs\config_docs\32mx450f256h.html



XC32 Configuration Bits Function

- 要設定除錯介面，則使用以下方法設定:

  #pragma config ICESEL = ICS_PGx3

C:\Program Files (x86)\ Config-Bit Settings for PIC...

Watchdog Timer Window Size (FWDTWINSZ)

FWDTWINSZ = WINSZ_75	Window Size is 75%
FWDTWINSZ = WINSZ_50	Window Size is 50%
FWDTWINSZ = WINSZ_37	Window Size is 37.5%
FWDTWINSZ = WISZ_25	Window Size is 25%

Background Debugger Enable (DEBUG)

DEBUG = ON	Debugger is Enabled
DEBUG = OFF	Debugger is Disabled

JTAG Enable (JTAGEN)

JTAGEN = ON	JTAG Port Enabled
JTAGEN = OFF	JTAG Disabled

ICE/ICD Comm Channel Select (ICESEL)

ICESEL = ICS_PGx1	Communicate on PGEC1/PGED1
ICESEL = ICS_PGx2	Communicate on PGEC2/PGED2
ICESEL = ICS_PGx3	Communicate on PGEC3/PGED3

Program Flash Write Protect (PWP)

PWP = OFF	Disable
PWP = PWP4K	First 4K
PWP = PWP8K	First 8K
PWP = PWP12K	First 12K
PWP = PWP16K	First 16K

REGISTER 27-1: DEVCFG0: DEVICE CONFIGURATION WORD 0 (CONTINUED)

bit 24 **BWP**: Boot Flash Write-Protect bit
Prevents boot Flash memory from being modified during code execution.
1 = Boot Flash is writable
0 = Boot Flash is not writable

bit 23-20 **Reserved**: Write '1'

bit 19-12 **PWP<7:0>**: Program Flash Write-Protect bits
Prevents selected program Flash memory pages from being modified during code execution. The PWP bits represent the one's compliment of the number of write protected program Flash memory pages.
11111111 = Disabled
11111110 = 0xBD00_0FFF
11111101 = 0xBD00_1FFF
11111100 = 0xBD00_2FFF
11111011 = 0xBD00_3FFF
11111010 = 0xBD00_4FFF
11111001 = 0xBD00_5FFF
11111000 = 0xBD00_6FFF
11110111 = 0xBD00_7FFF
11110110 = 0xBD00_8FFF
11110101 = 0xBD00_9FFF
11110100 = 0xBD00_AFFF
11110011 = 0xBD00_BFFF
11110010 = 0xBD00_CFFF
11110001 = 0xBD00_DFFF
11110000 = 0xBD00_EFFF
11101111 = 0xBD00_FFFF
.
.
.
01111111 = 0xBD07_FFFF

bit 11-5 ~~**Reserved**: Write '1'~~

bit 4-3 **ICESEL<1:0>**: In-Circuit Emulator/Debugger Communication Channel Select bits
11 = PGEC1/PGED1 pair is used
10 = PGEC2/PGED2 pair is used
01 = PGEC3/PGED3 pair is used
00 = Reserved

bit 2 **JTAGEN**: JTAG Enable bit(1)
1 = JTAG is enabled
0 = JTAG is disabled

bit 1-0 **DEBUG<1:0>**: Background Debugger Enable bits (forced to '11' if code-protect is enabled)
1x = Debugger is disabled
0x = Debugger is enabled

Configuration Bits Setup

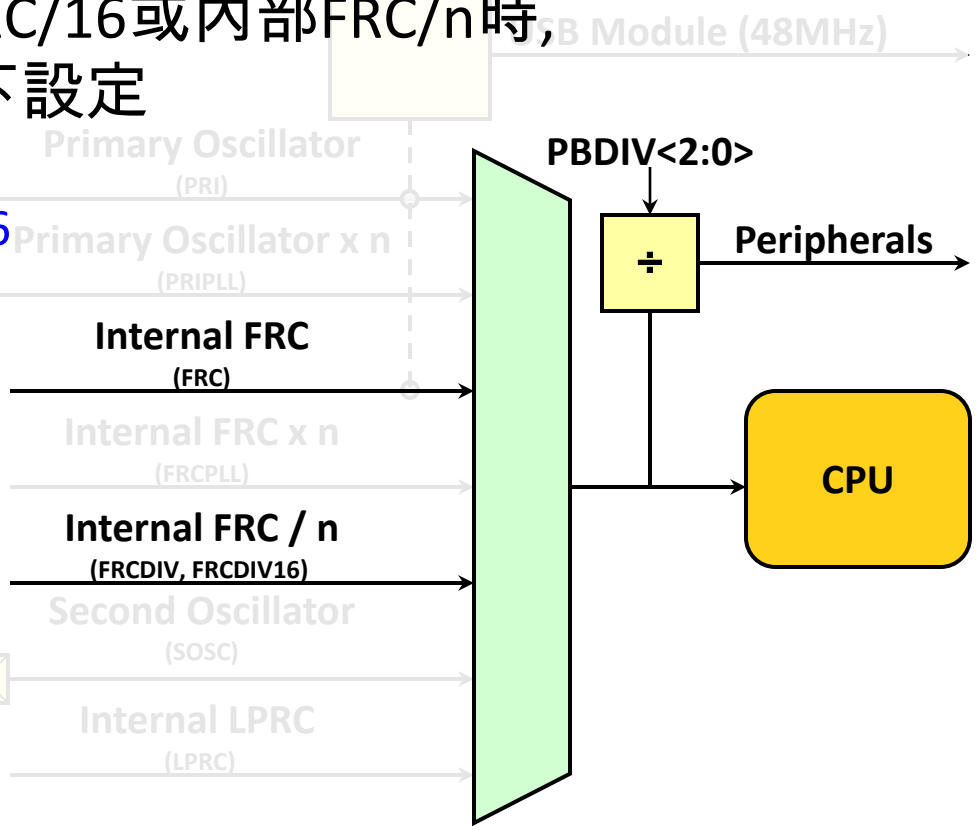
FRC & FRCDIV16 & FRCDIV

- PIC32MX的Oscillator來源可以選擇使用內部FRC,內部FRC/16或內部FRC/n。PIC32MX內部的FRC, 震盪頻率為8MHz。
- 要選擇使用內部FRC, 內部FRC/16或內部FRC/n時, Configuration Bits必須作以下設定

  #pragma config FNOSC = FRC
  #pragma config FNOSC = FRCDIV16
  #pragma config FNOSC = FRCDIV
- 如果選擇FRCDIV時, 還須透過設定OSCCONbits.FRCDIV決定n值(Default n = 2) 。

  Ex:OSCCONbits.FRCDIV = 0x00;

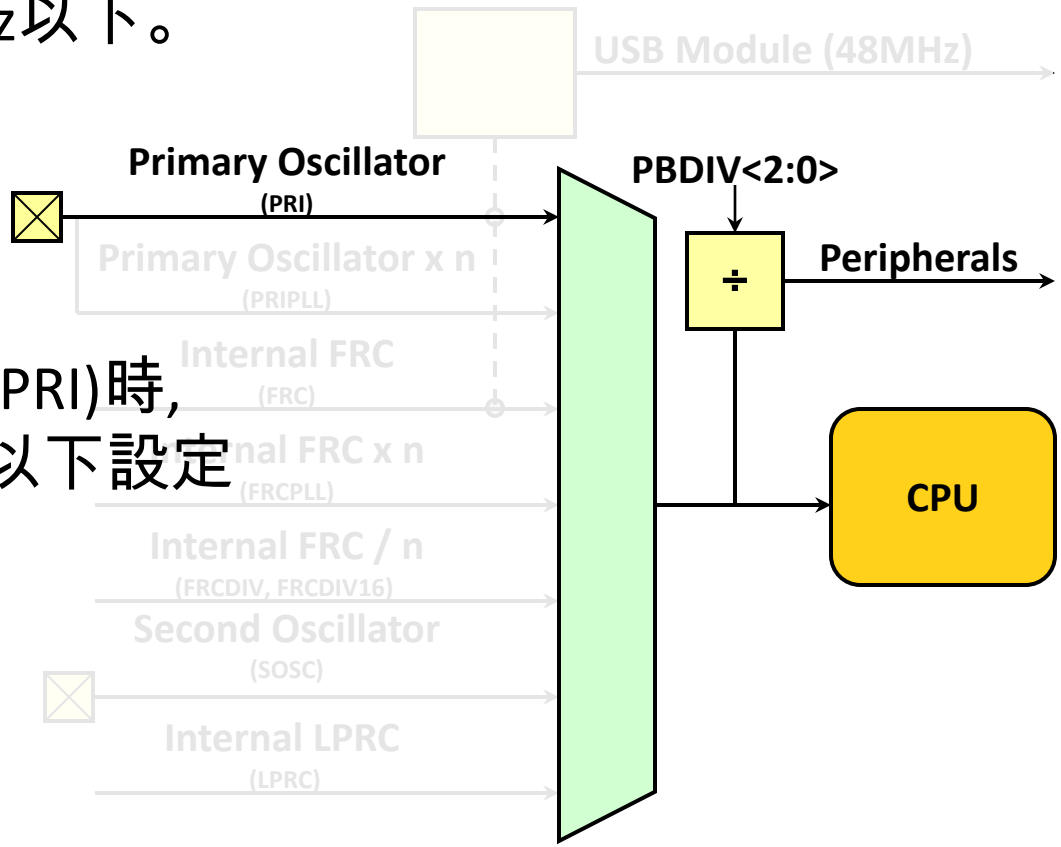
bit 26-24 **FRCDIV<2:0>**: Internal Fast RC (FRC) Oscillator Clock Divider bits
- 111 = FRC divided by 256
- 110 = FRC divided by 64
- 101 = FRC divided by 32
- 100 = FRC divided by 16
- 011 = FRC divided by 8
- 010 = FRC divided by 4
- 001 = FRC divided by 2 (default setting)
- 000 = FRC divided by 1



Configuration Bits Setup

PRI

- PIC32MX的Oscillator來源可以選擇使用主要外部時脈來源接腳(PRI)。 Primary OSC.通常是用來連接高速的Crystal。其最高的輸入頻率被限定在25MHz以下。



- 要選擇使用Primary OSC.(PRI)時, Configuration Bits必須作以下設定

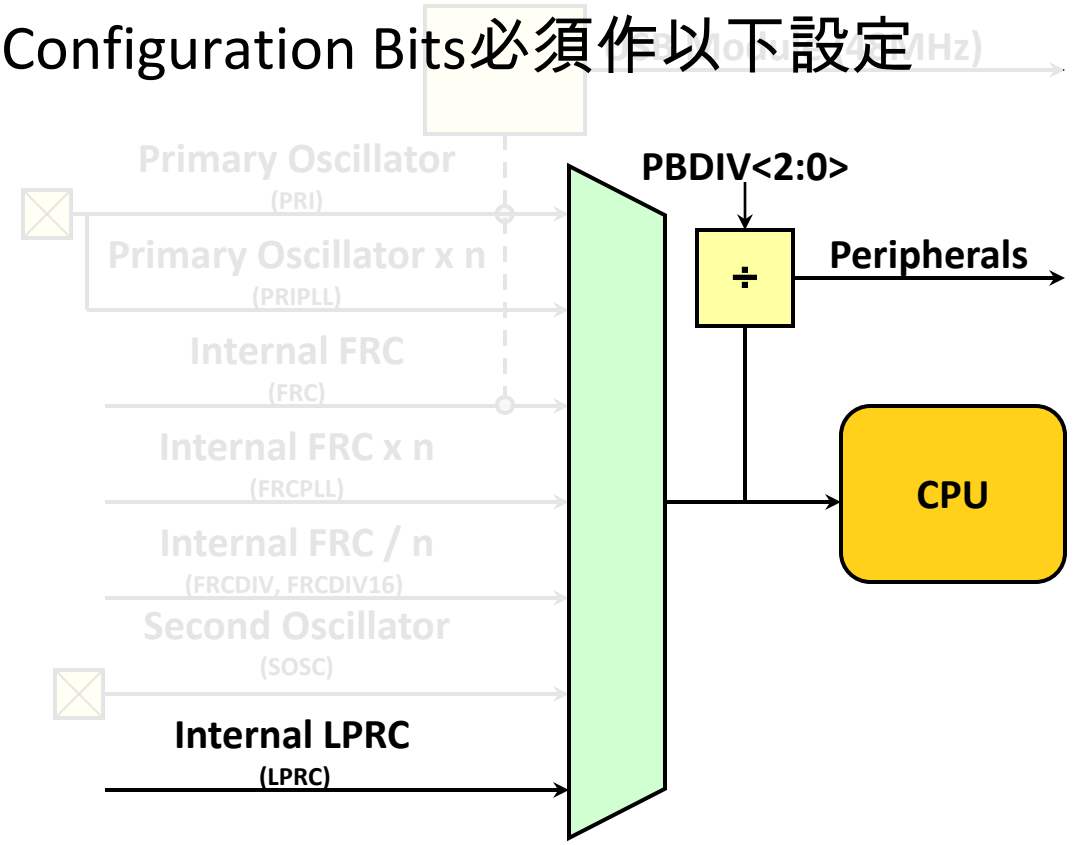
#pragma config FNOSC = PRI

Configuration Bits Setup

LPRC

- PIC32MX的Oscillator來源可以選擇使用內部LPRC, PIC32MX內部的LPRC, 震盪頻率為31.25KHz。
- 要選擇使用內部LPRC時, Configuration Bits必須作以下設定

  #pragma config FNOSC = LPRC

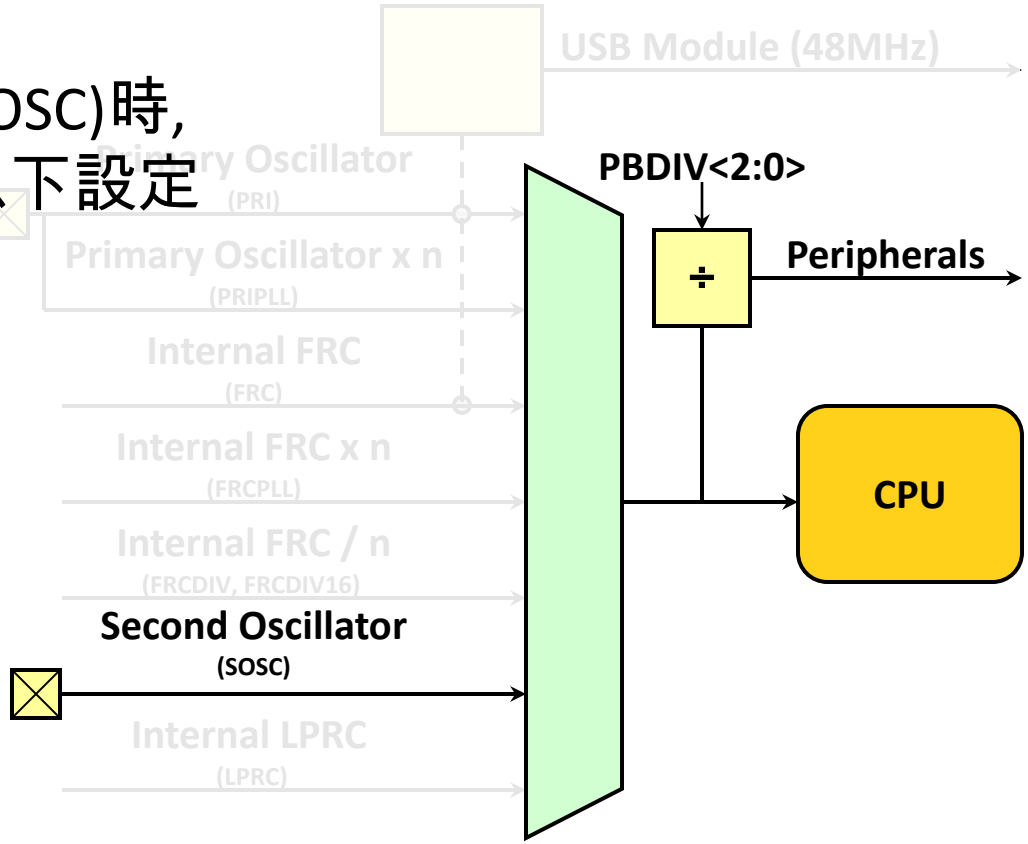


Configuration Bits Setup

SOSC

- PIC32MX的Oscillator來源可以選擇使用第二組外部時脈來源接腳(SOSC)。 Second OSC.通常是用來接低速的Crystal,如:32.768K Hz。
- 要選擇使用Second OSC.(SOSC)時, Configuration Bits必須作以下設定

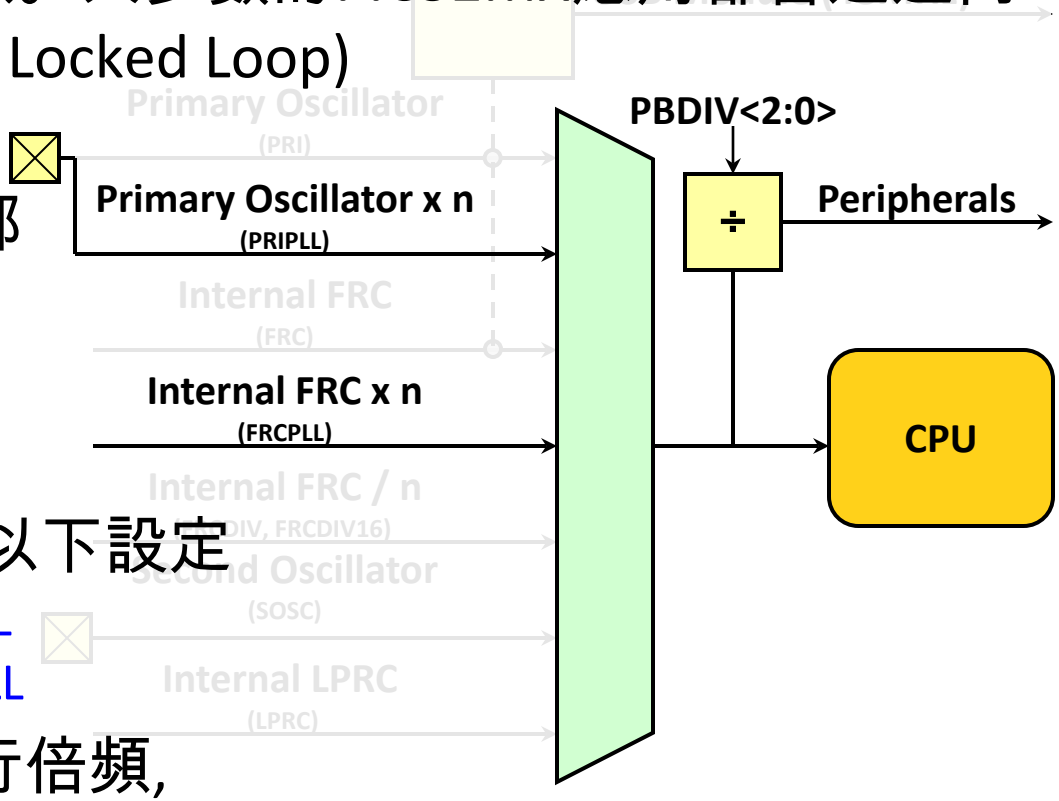
#pragma config FNOSC = SOSC



Configuration Bits Setup

PRIPLL, FRCPLL

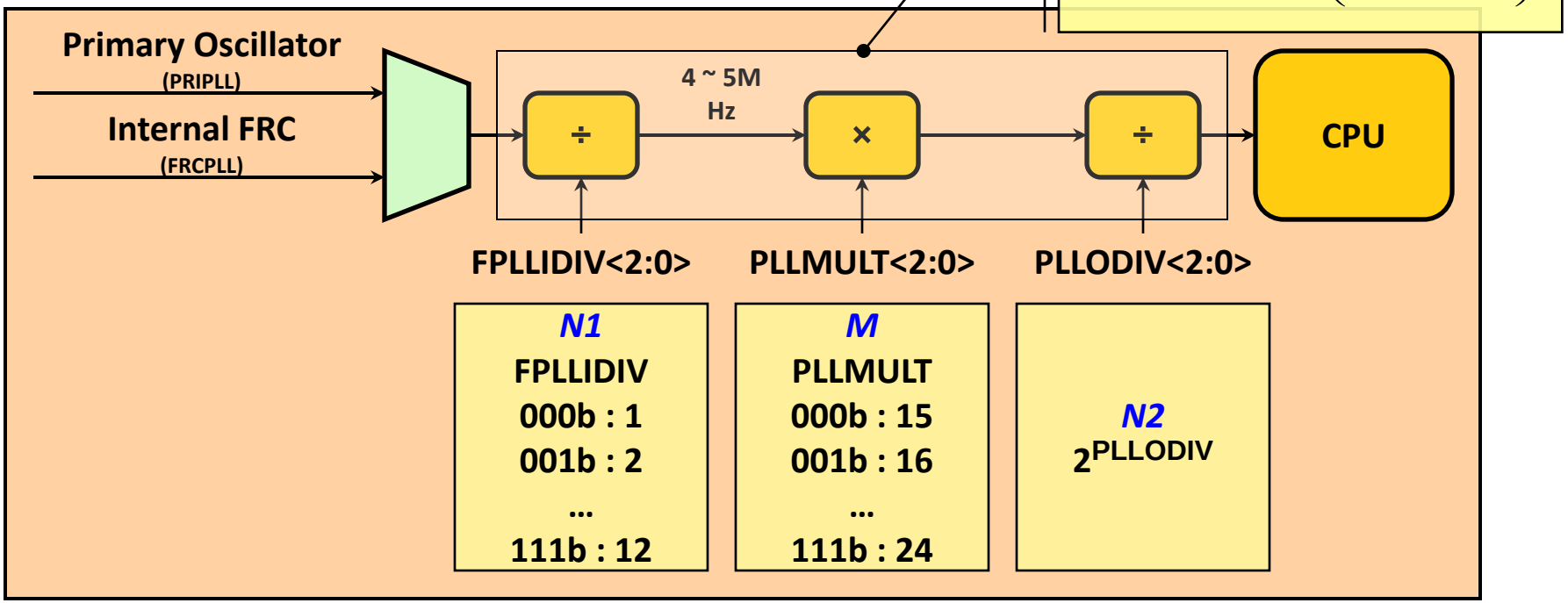
- PICMX32最高工作頻率可以達到80MHz, 但上述幾種時脈來源都無法提供80MHz的時脈。大多數的PIC32MX應用都會透過內部的倍頻線路(PLL, Phase Locked Loop)來提高頻率。
- 主要外部時脈來源跟內部FRC都可以透過倍頻線路來提高頻率。
- 要選擇使用時,
  Configuration Bits必須作以下設定
  #pragma config FNOSC = PRIPLL
  #pragma config FNOSC = FRCPLL
- PRIPLL, FRCPLL因為會進行倍頻,
  所以還必須額外設定倍頻線路(PLL, Phase Locked Loop)的參數。



Phase Locked Loop

- PIC32可以透過PLL線路,提高供給CPU的頻率。
- Primary Osc.跟FRC可以透過PLL線路倍頻。
- PLL的架構分為三個區塊:

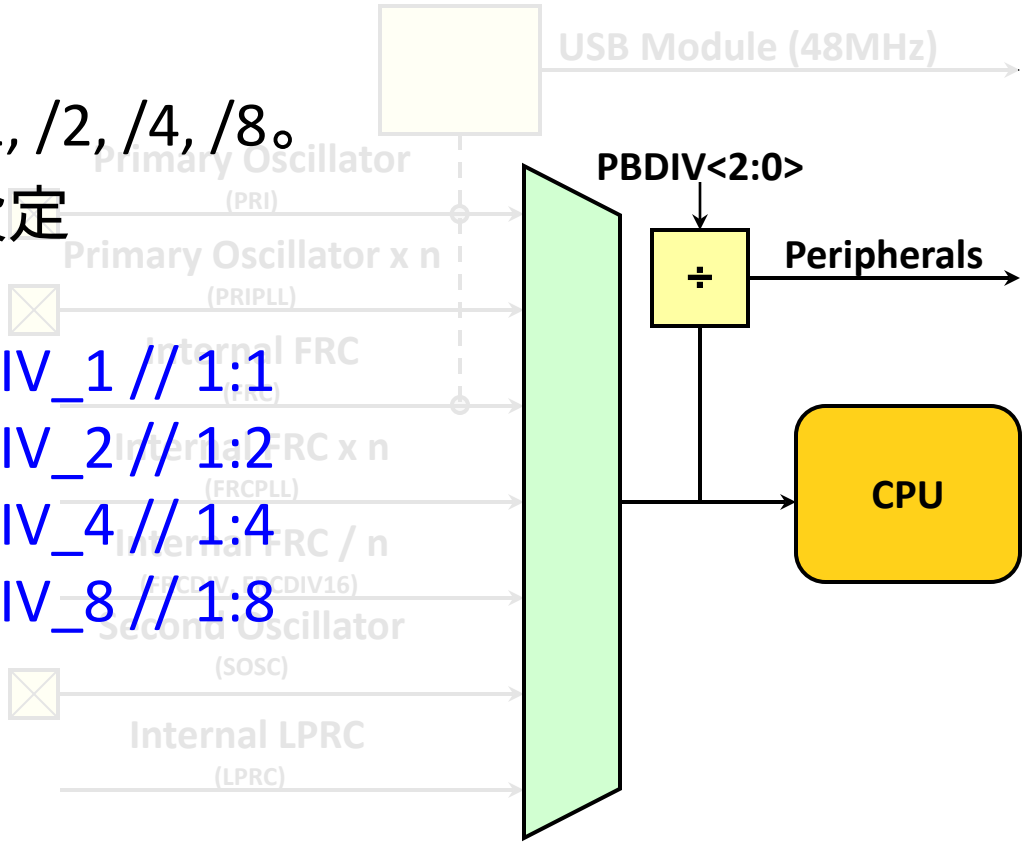
$$F_{OSC} = F_{IN} \times \left(\frac{M}{N1 \times N2} \right)$$



System Clock and PB Clock

- PIC32的系統時脈(System Clock)與周邊時脈(PB Clock)可以運作不同頻率,兩者間存在一比率關係。
- 周邊時脈(PB Clock) =
  系統時脈(System Clock) / 1, /2, /4, /8。
- Configuration Bits的比率設定
  可參考以下設定

```
#pragma config FPBDIV = DIV_1 // 1:1
#pragma config FPBDIV = DIV_2 // 1:2
#pragma config FPBDIV = DIV_4 // 1:4
#pragma config FPBDIV = DIV_8 // 1:8
```



MICROCHIP
Regional Training Centers

FRCPLL, PRIPLL Setup Example

- **FRC OSC (8MHz) with PLL. ((8MHz / 2) * 15) / 1 = 60MHz**
  System Freq. / 2 = PB Freq. = 30MHz
  #pragma config FNOSC = FRCPLL
  #pragma config FPLLIDIV = DIV_2 , FPLLMUL = MUL_15 , FPLLODIV = DIV_1
  #pragma config FPBDIV = DIV_2

- **Primary OSC (8MHz) with PLL. ((8MHz / 2) * 20) / 1 = 80MHz**
  System Freq. = PB Freq. = 80MHz
  #pragma config FNOSC = PRIPLL , POSCMOD = HS
  #pragma config FPLLIDIV = DIV_2 , FPLLMUL = MUL_20 , FPLLODIV = DIV_1
  #pragma config FPBDIV = DIV_1

- **Primary OSC (8MHz) with PLL. ((8MHz / 2) * 20) / 2 = 40MHz**
  System Freq. / 8 = PB Freq. = 5MHz
  #pragma config FNOSC = PRIPLL , POSCMOD = HS
  #pragma config FPLLIDIV = DIV_2 , FPLLMUL = MUL_20 , FPLLODIV = DIV_2
  #pragma config FPBDIV = DIV_8

MICROCHIP
Regional Training Centers

Oscillator On-Fly-Change

- PIC32MX支援系統時脈的動態切換。程式運作時因為各種需要必須改變時脈頻率時, 就可以使用動態切換。
- 動態切換有一套很複雜的流程, 參考Datasheet可知一二。要自己手工打造太辛苦了。好加在XC32有提供現成的函數可以支援。

  Microchip PIC32MX Peripheral Library.chm。

  C:\Program Files (x86)\Microchip\xc32\v1.30\docs\pic32-lib-help\

  OSCConfig(Clock Source , PLL Multiplier , PLL Postscaler , FRC Divisor);

  以PLLIDIV 設為除2, Primary Crystal = 8MHz為例: (*PLLIDIV無法動態修改)

```
OSCConfig( OSC_FRC , 0 , 0 , 0 );
// Sys. Clock =  FRC = 8 MHz
OSCConfig( OSC_FRC_DIV , 0 , 0 , OSC_FRC_POST_2 );
// Sys. Clock =  FRC / 2 = 4 MHz
OSCConfig( OSC_FRC_PLL , OSC_PLL_MULT_20 , OSC_PLL_POST_2 , 0 );
// Sys. Clock =  ((FRC / 2) * 20) / 2 = 40 MHz
OSCConfig( OSC_POSC_PLL , OSC_PLL_MULT_20 , OSC_PLL_POST_1 , 0 );
// Sys. Clock =  ((PRI / 2) * 20) / 1 = 80 MHz
```

MICROCHIP
Regional Training Centers

Oscillator On-Fly-Change

- 周邊時脈(PB Clock)與系統時脈(System Clock)的比率也可以動態切換。
- OSCSetPBDIV (PB Divider);
  Ex:
  OSCSetPBDIV(OSC_PB_DIV_8);
  // System Freq. / 8 = PB Freq.
  OSCSetPBDIV(OSC_PB_DIV_4);
  // System Freq. / 4 = PB Freq.
  OSCSetPBDIV(OSC_PB_DIV_2);
  // System Freq. / 2 = PB Freq.
  OSCSetPBDIV(OSC_PB_DIV_1);
  // System Freq. / 1 = PB Freq.

MICROCHIP
Regional Training Centers

Lab4 Oscillator On-Fly-Change

- 以Lab3的程式為基礎, 利用XC32提供的Oscillator On-Fly-Change來設定時脈。將CPU時脈來源轉為FRC (8MHz)。

- 透過Bootloader將程式實際燒錄到MCU中, 用肉眼觀察看看程式執行的情況。是否變成跟軟體模擬的時間相同?

Lab4 Oscillator On-Fly-Change

Step1

- 如何達成Oscillator On-Flay-Change?

  根據Lab3的設定, 應該調整8MHz, 程式的速度才符合期待。APP2013TM上只少有兩組8MHz, FRC跟PRI。請參考投影片跟說明文件Microchip PIC32MX Peripheral Library.chm。擇一設定即可。

- 為何需要Oscillator On-Flay-Change?

  因為Bootloader所設定的系統時脈, 不一定符合程式的需求, 所以必須透過動態切換, 來達成。

Lab4 Oscillator On-Fly-Change

Step2

- 注意! 每次調整頻率後, 必須重新修正PIC32的Wait State與Prefetch的設定, 才能讓系統達到最佳化的效能。
- MPLAB XC32提供的SYSTEM Config函數可以用來調整Wait State與Prefetch的設定

  SYSTEMConfig(Sys Clock , Flag);

  Ex:

  SYSTEMConfig(8000000 , SYS_CFG_WAIT_STATES | SYS_CFG_PCACHE);

- 所以記住必須在調整頻率的程式碼後面加入 SYSTEMConfig(Sys Clock , Flag);函數。