ATSAMC21G17A-101 eRTC

跟著Microchip使用APP-Nano-C21-D21-TW
為這個世界帶來更多的樂趣

MICROCHIP

A Leading Provider of Smart, Connected and Secure Embedded Solutions

SMART | CONNECTED | SECURE

APP-Nano-C21-D21-TW 初級開發者學習包

- Microchip台灣網站首頁選擇“Microchip Webinar 資料區”進入

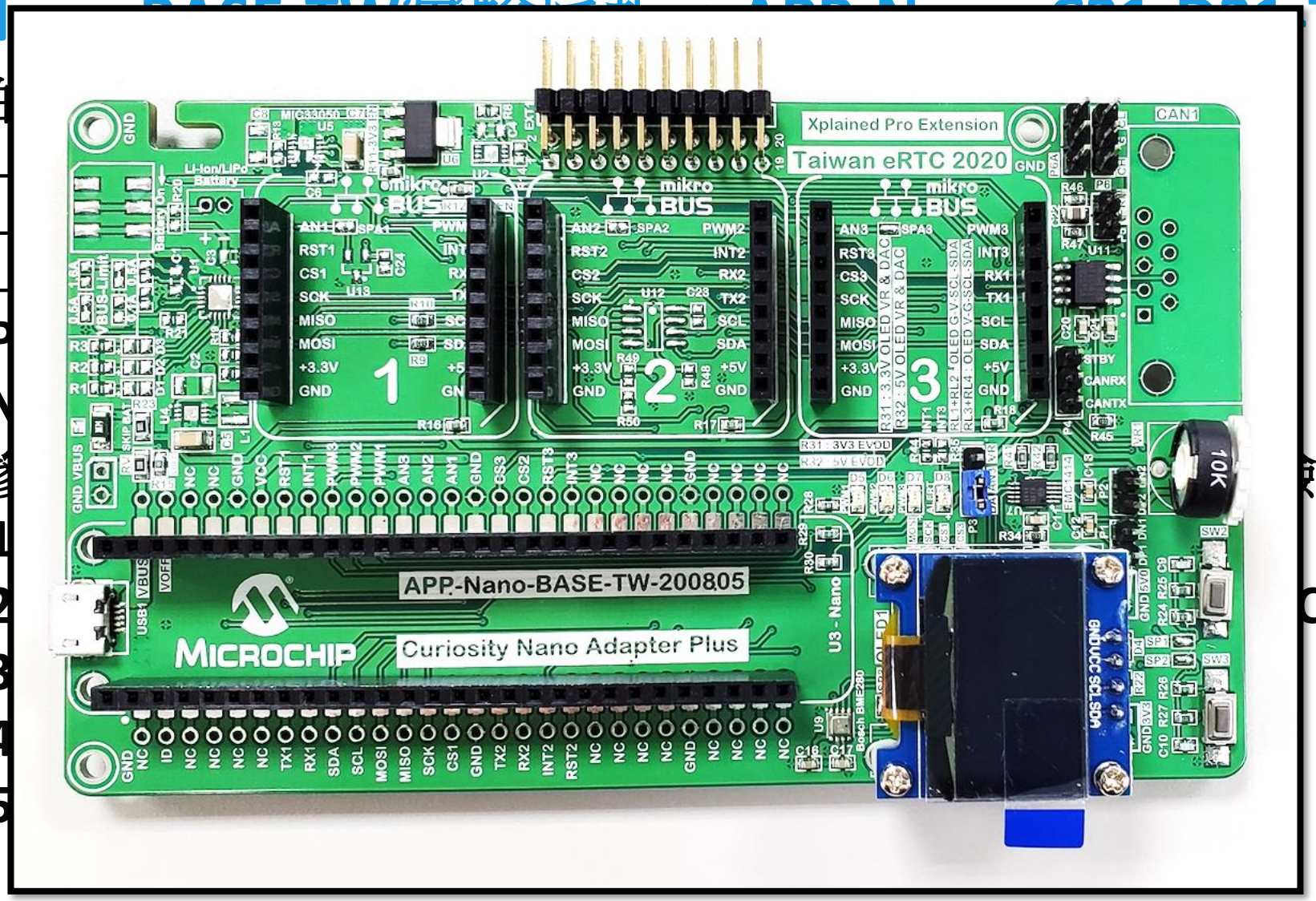
APP-Nano-C21-D21-TW

MCU: ATSAMC21G17A-AU

MICROCHIP

ATSAMC21G17A-101 eRTC課程內容

APP-N



- 使用
  - MPL
  - MPL
  - MPL
- 參考P
- **APP-N**
- 可以
- 練習1
- 練習2
- 練習3
- 練習4
- 練習5

Callback

MICROCHIP

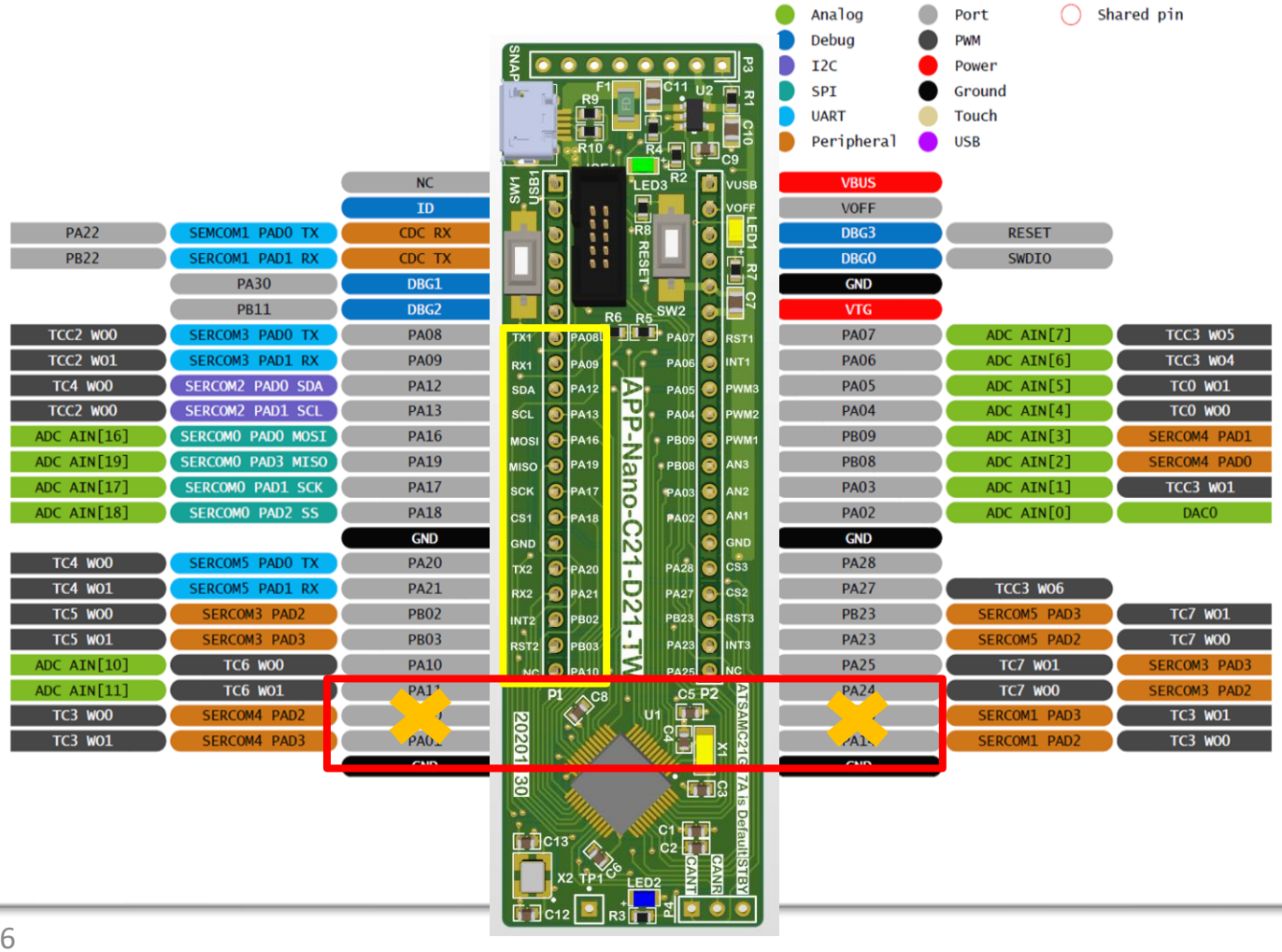
ATSAMC21G17A-101 eRTC課程內容

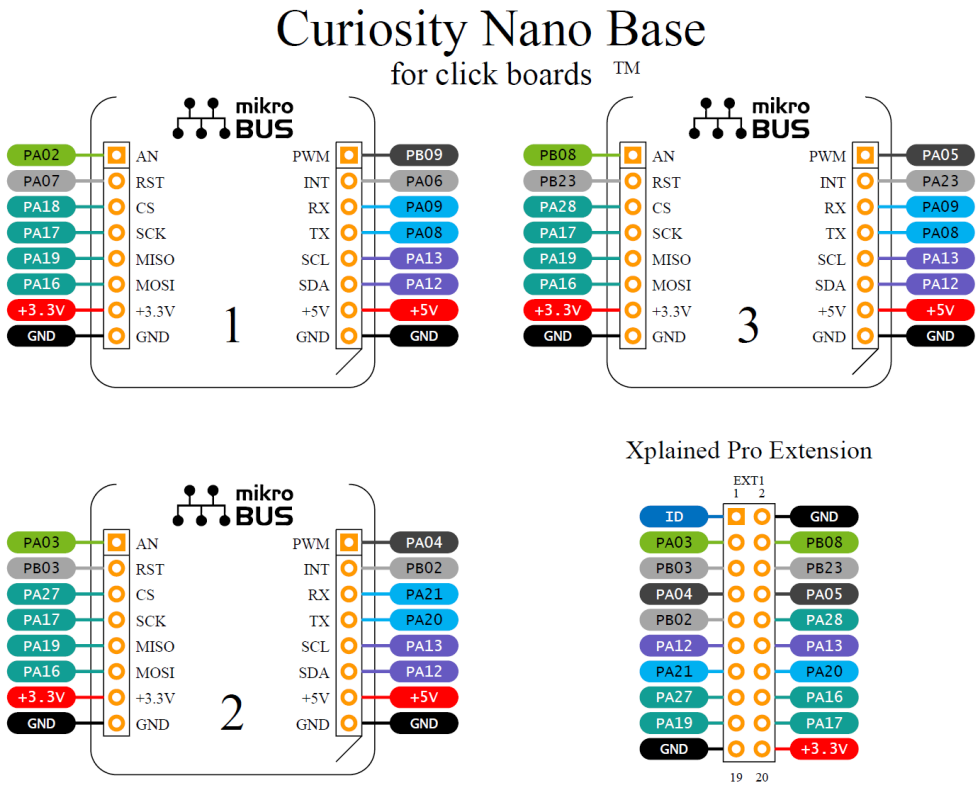
APP-Nano-BASE-TW實驗底板 + APP-Nano-C21-D21-TW

- 使用的軟體開發平台： **MPLAB® X IDE & MPLAB Harmony 3**
  - MPLAB XIDE V5.45
  - MPLAB Harmony 3 Configurator
  - MPLAB Harmony 3 Content Manager
- 參考**PLIB**相關的說明文件
- **APP-Nano-C21-D21-TW**設計及電路解說
- 可以參考https://microchipdeveloper.com/的範例來進行實驗
- **練習1：基本的I/O設定：按鍵與LED操作**
- **練習2：中斷實作：EIC以及TC的中斷操作法——使用PLIB & Callback**
- **練習3：練習使用APP-Nano-BASE-TW上的資源——OLED**
- **練習4：讀取APP-Nano-BASE-TW上的MCP9800溫度Sensor**
- **練習5：為你的ATSAMC21G17A加入CAN的傳送功能 ☺**

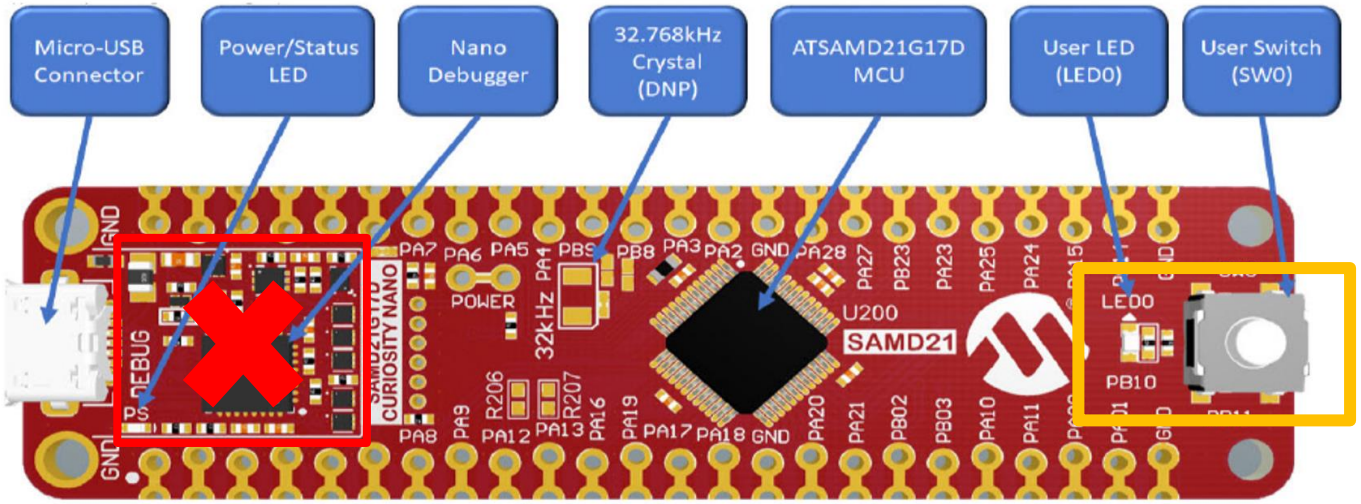
APP-Nano-C21-D21-TW Pinout vs. D21 Curiosity Nano
（沒有全部I/O都對應，但對Curiosity Nano Base 的3個mikroBUS都支援）

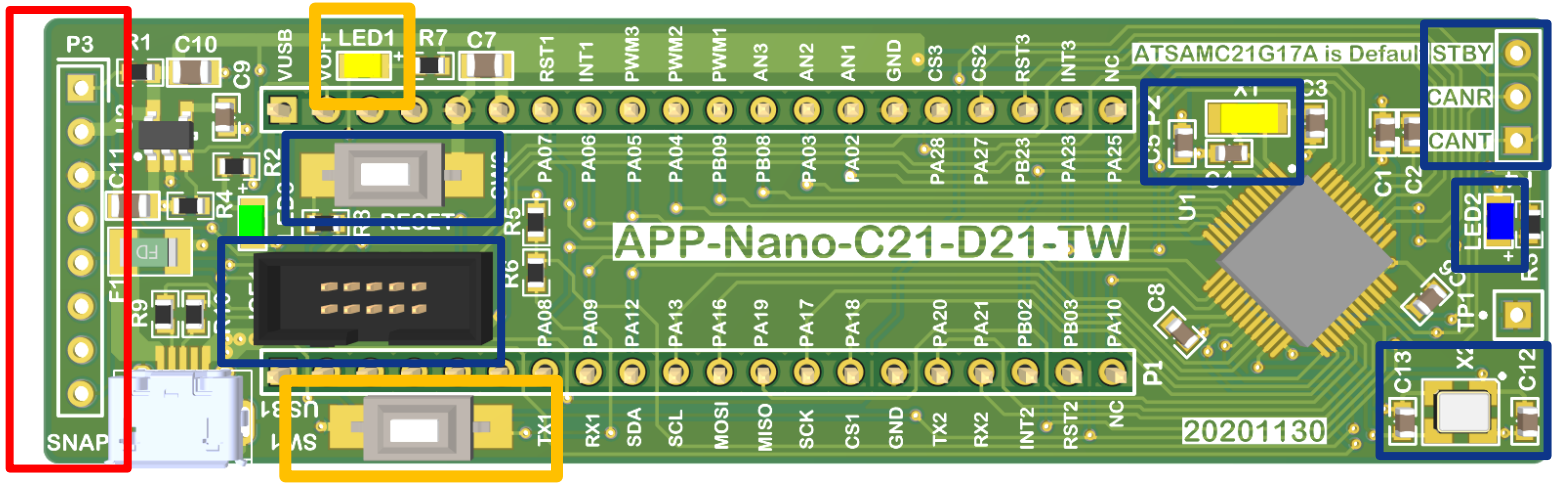
ATSAMC21G17A





APP-Nano-C21-D21-TW其他主要差異



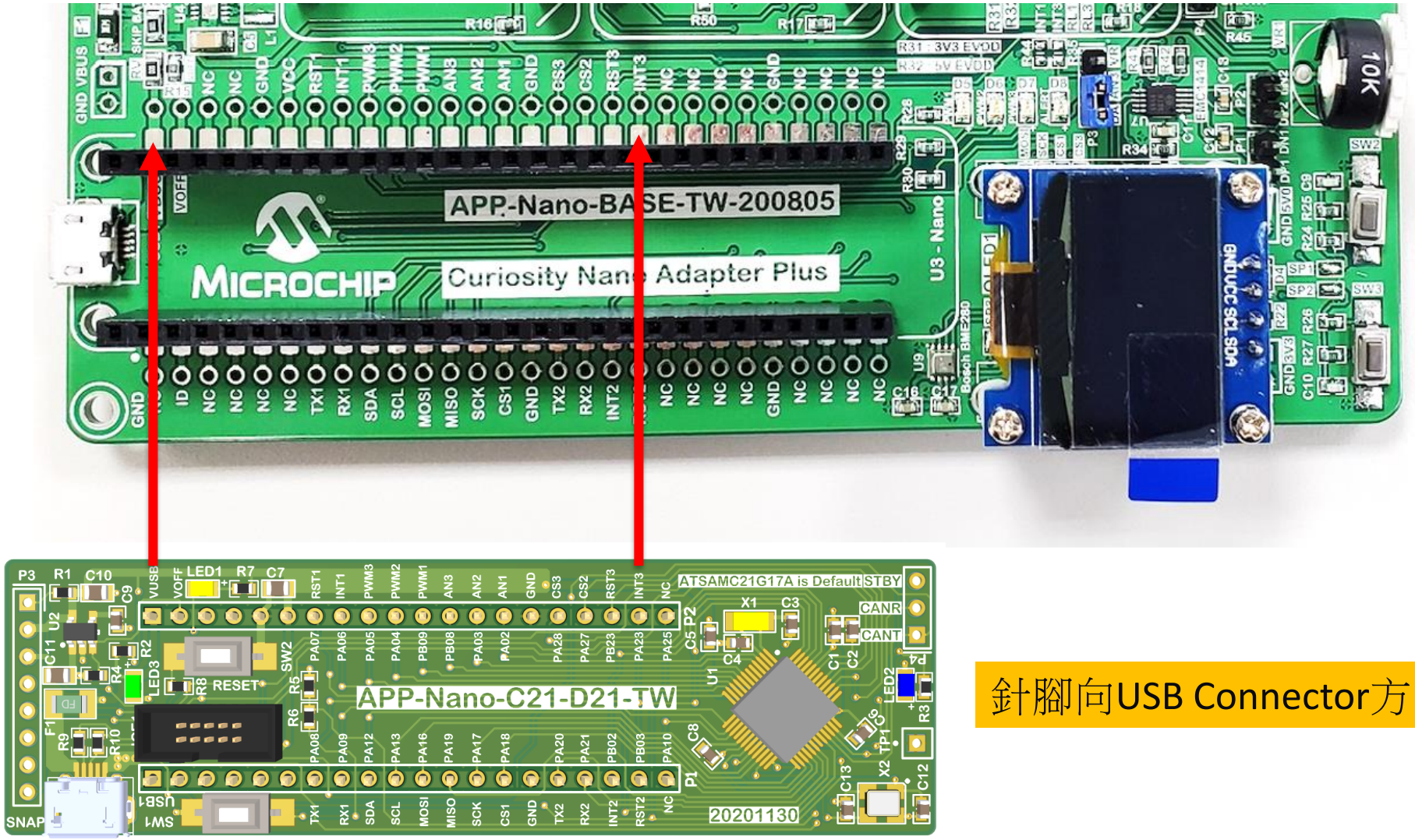


- 新增
  - X1 (32.768 kHz)
  - X2 (12 MHz)
  - LED2 (PA10)
  - SW2 (Reset SW)
  - P4 (CAN Connector)
  - ICE1 (JTAG signal for ATMEL ICE)

MICROCHIP

APP-Nano-C21-D21-TW

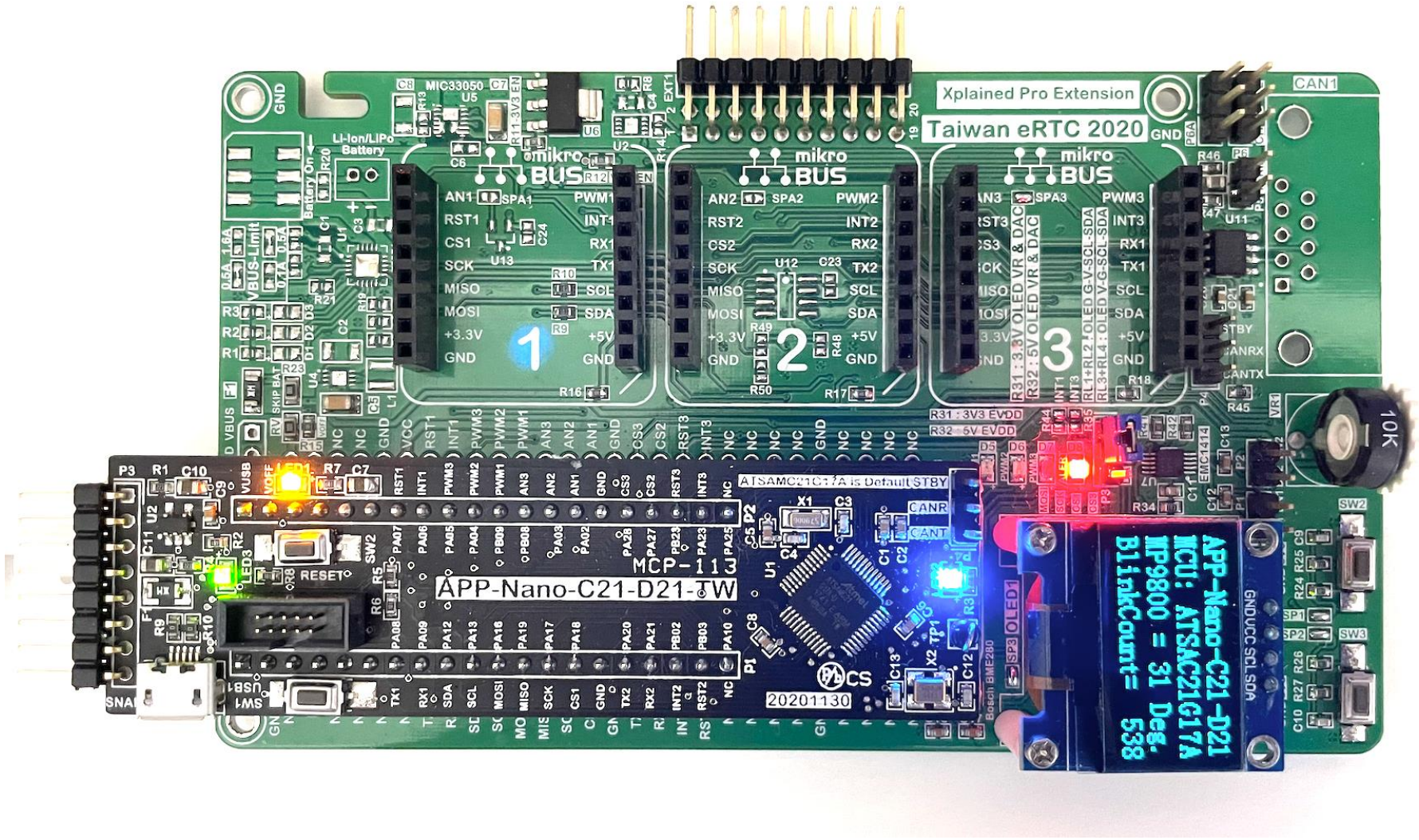
搭配底板： APP-Nano-BASE-TW Nano Board注意事項



針腳向USB Connector方向對齊

APP-Nano-C21-D21-TW

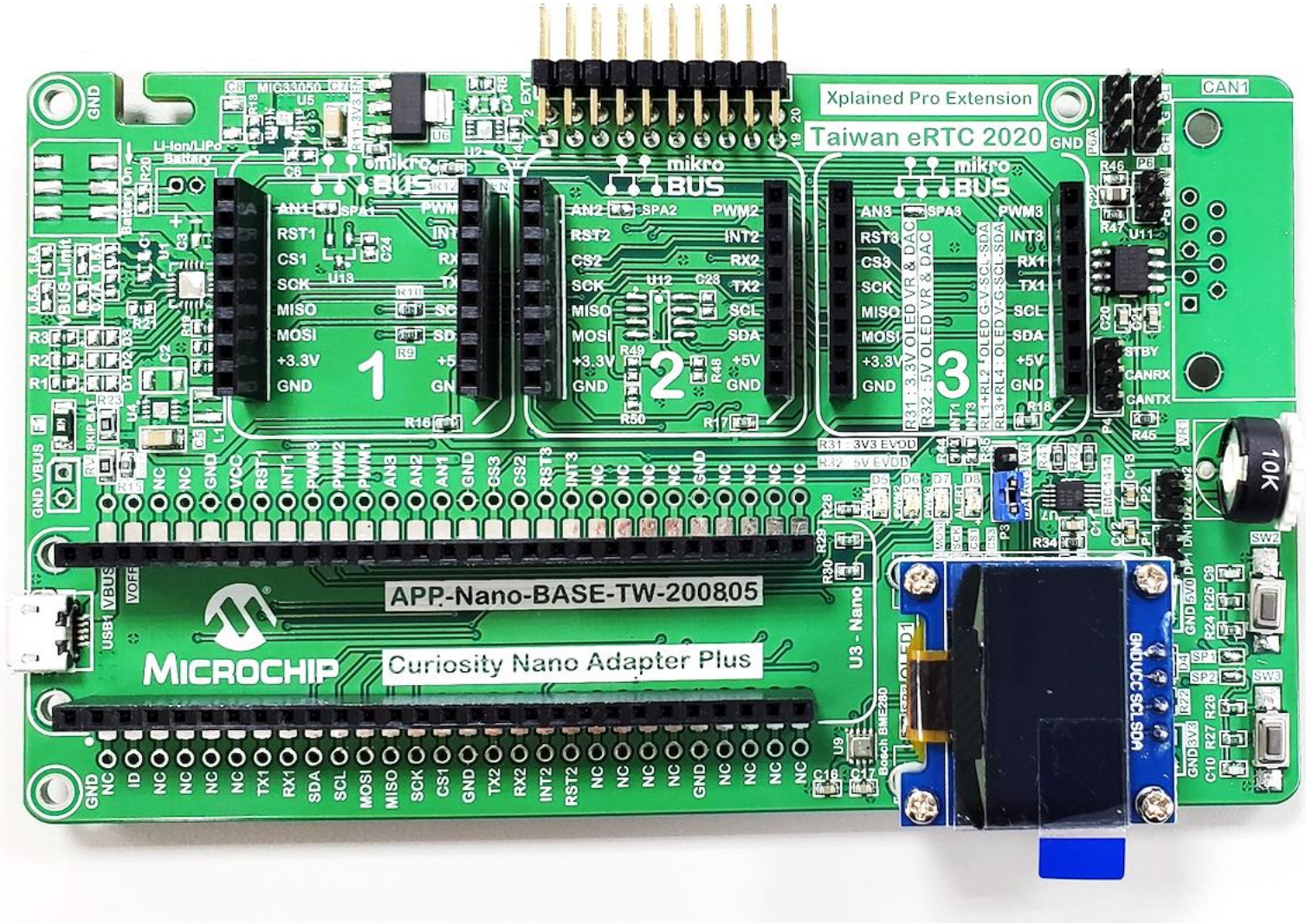
搭配底板： APP-Nano-BASE-TW Nano Board的實際狀況



MICROCHIP

APP-Nano-BASE-TW台灣加強版 – 相容於AC164162

目的：讓入門的學習更為方便、容易——增加許多基礎周邊



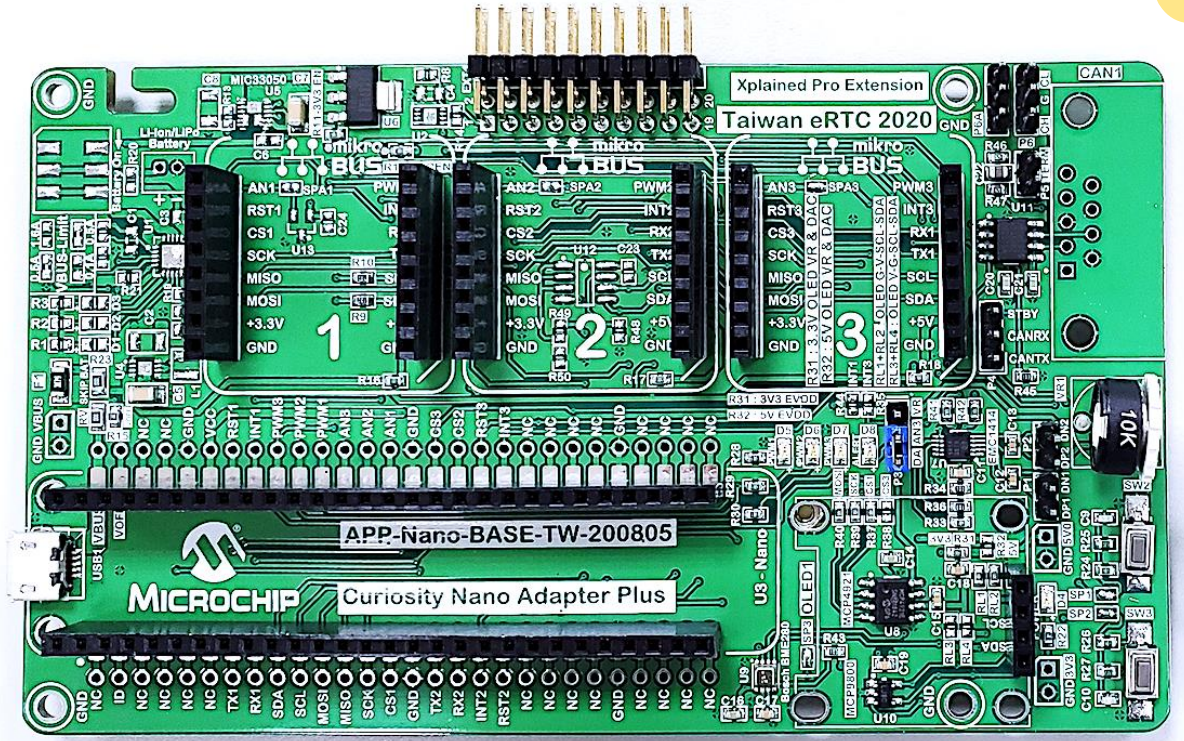
讓我們
複習一下 ！

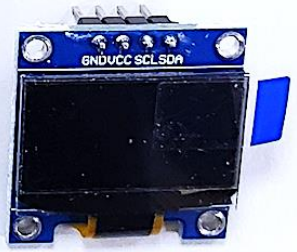
MICROCHIP

AC164162可以搭載超過10種以上的Microchip Curiosity Nano，但是APP-Nano-BASE-TW加上更多的預置功能

- **2個按鍵開關做信號輸入練習**
  - INT1 & INT2 Pin
- **4個LED進行狀態輸出指示**
  - PWM1 , PWM2 , PWM3, ALERT (TEMP)
- **1個VR作為可變的類比輸入**
  - AN3
- **1個使用I²C的OLED Display**
  - (SH1306 controller) SDA & SCL
- **2個I²C Temp Censor**
  - MCP9800 & EMC1414
- **1個SPI介面的DAC**
  - MCP4921
- **1個CAN Transceiver & Connector**
- **獨立的USB電源供應**
- **Bosch Sensor BME280**
  - Humidity sensor
  - Barometric pressure
  - Ambient Temperature

複習 ！



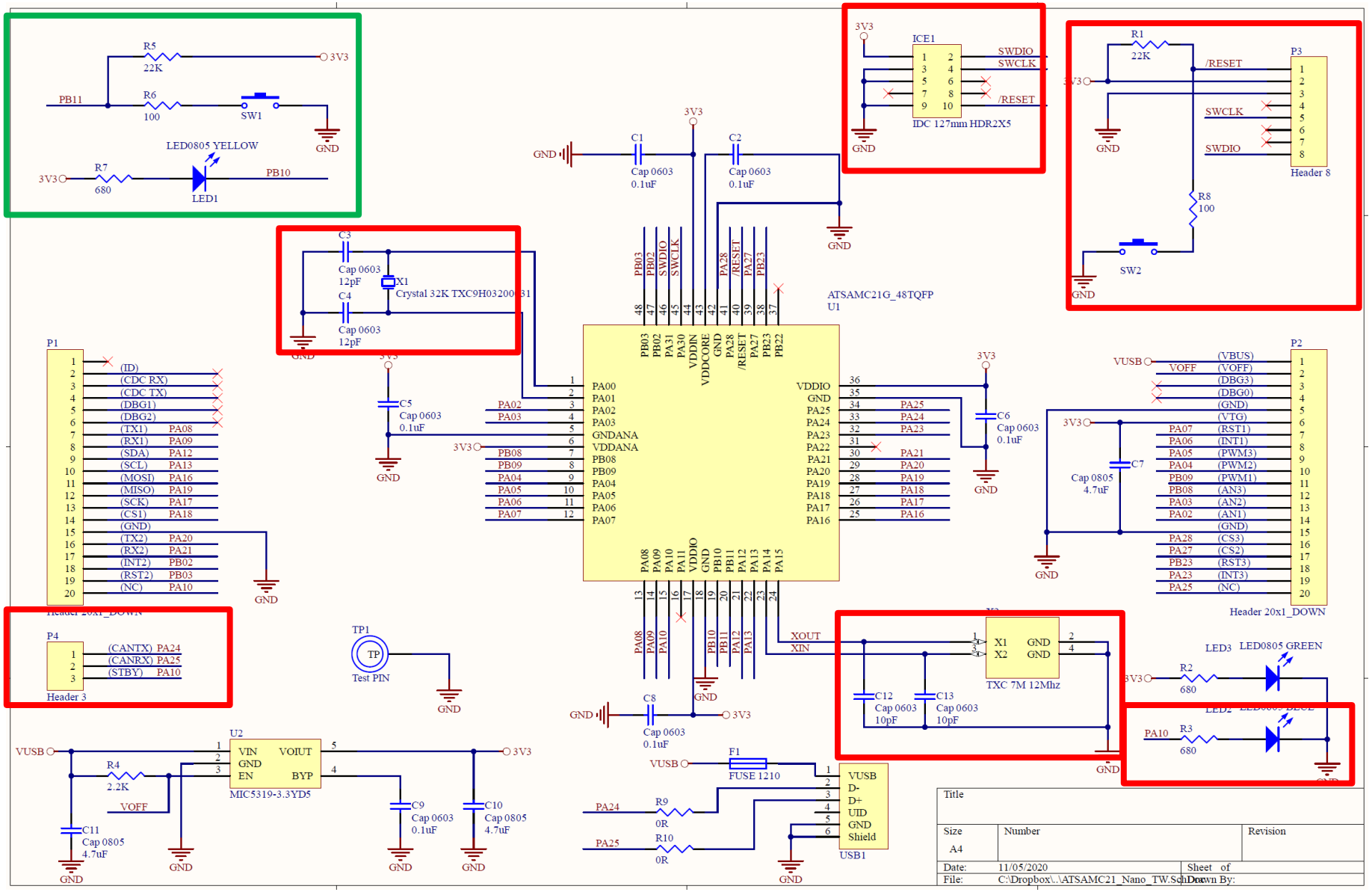


MICROCHIP

APP-Nano-BASE-TW 使用時的小小提醒

- **以下的IC或Connector在APP-Nano-BASE-TW並未安裝（1）**
  - U1, U2, U4, U5
  - SW1, J1,J14
  - L1
  - 它們著重於展示電源管理功能，並不影響基礎的程式練習
- **以下的IC或Connector在APP-Nano-BASE-TW並未安裝（2）**
  - CAN1： User 若要用 DB-9 connector for CAN 可自行安裝
  - U12：I^2C EEPROM相容腳位，User可安裝EEPROM or Crypto IC
  - U13： MCP9700A Analog-Out溫度sensor ， User可自行安裝

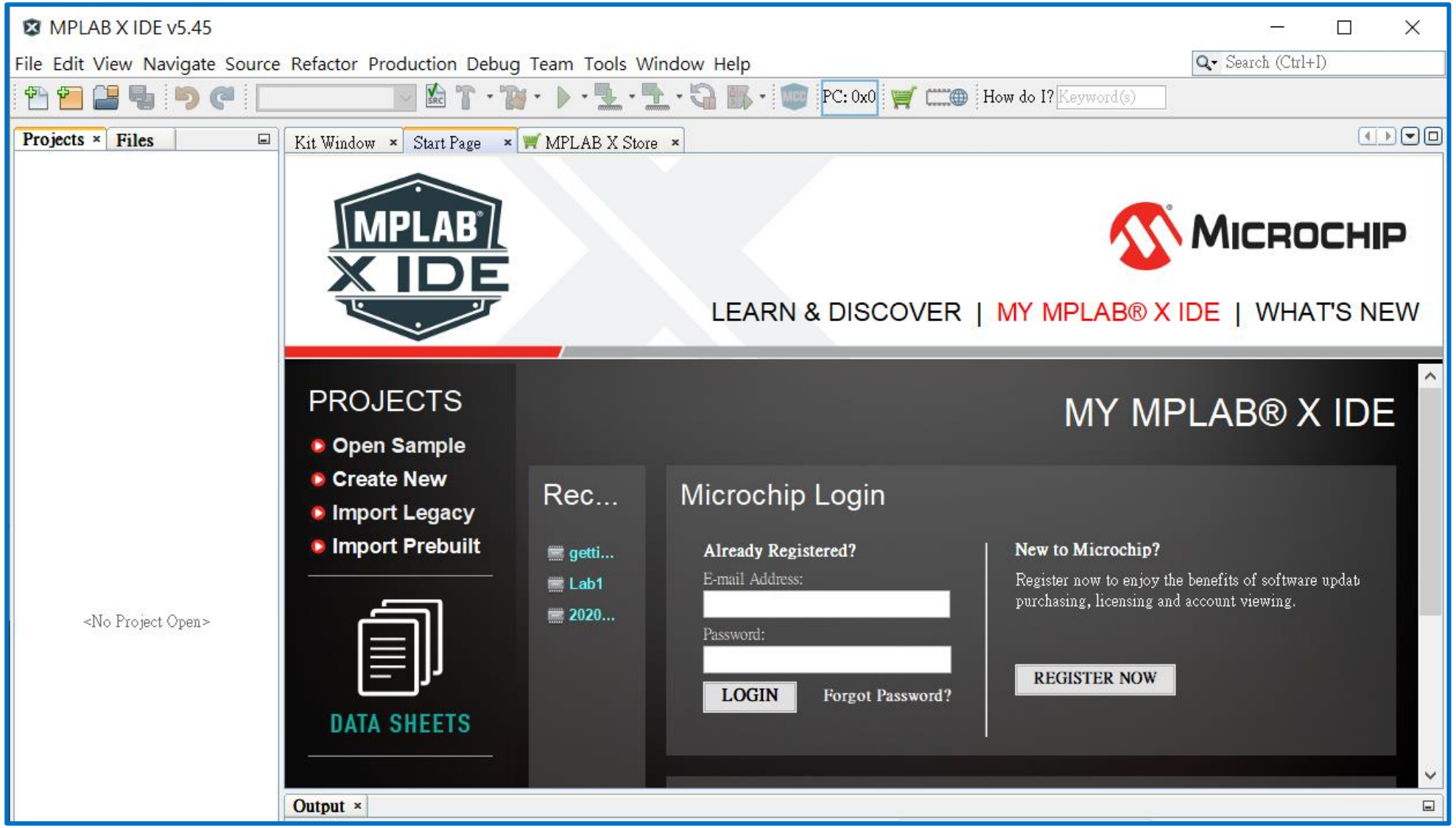
APP-Nano-C21-D21-TW線路圖



MICROCHIP

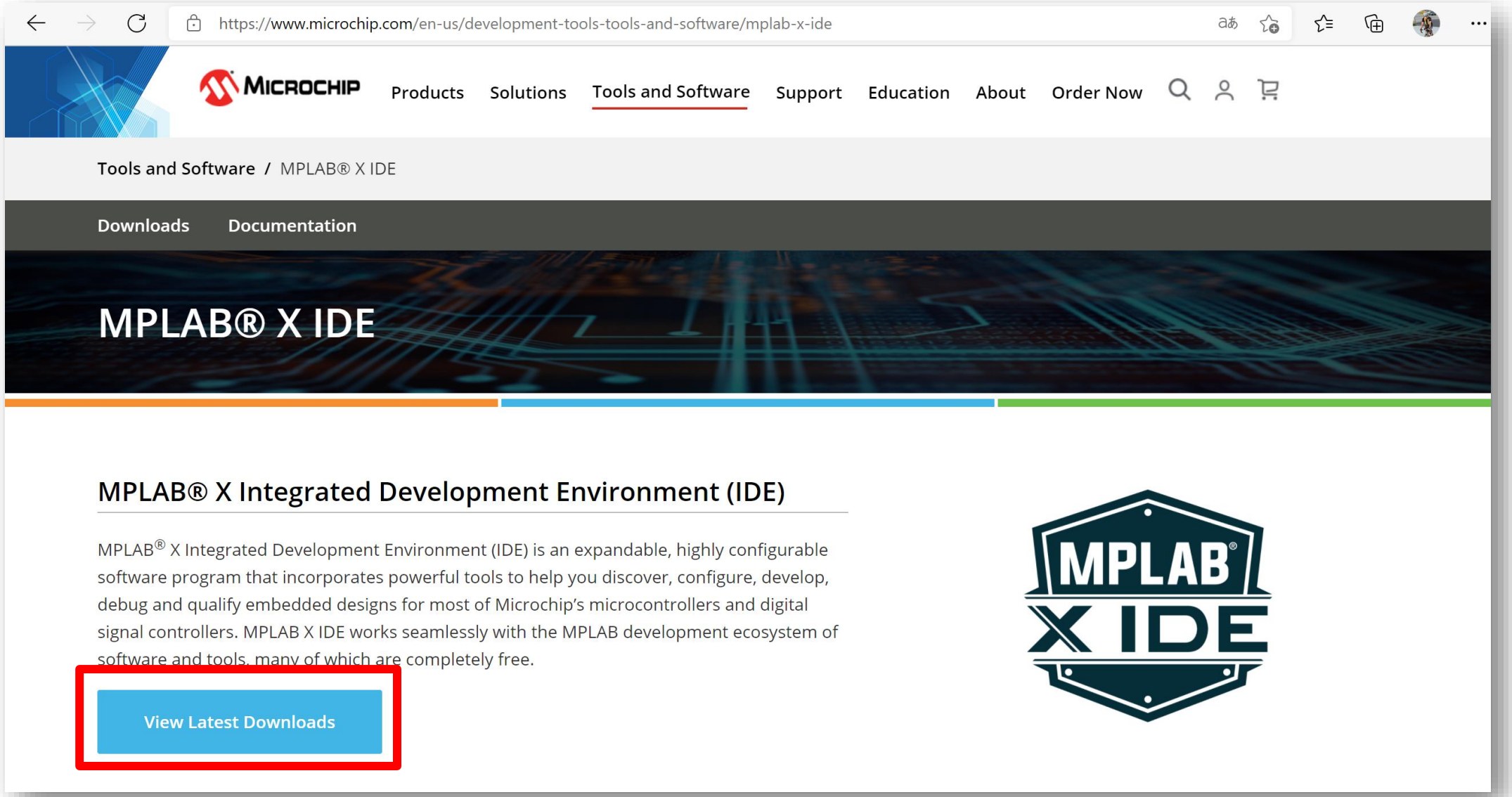
實驗前的準備: 檢查及安裝
MPLAB® X IDE
MPLAB Harmony 3
MPLAB XC32

MPLAB® X IDE版本：請安裝V5.45 or Higher

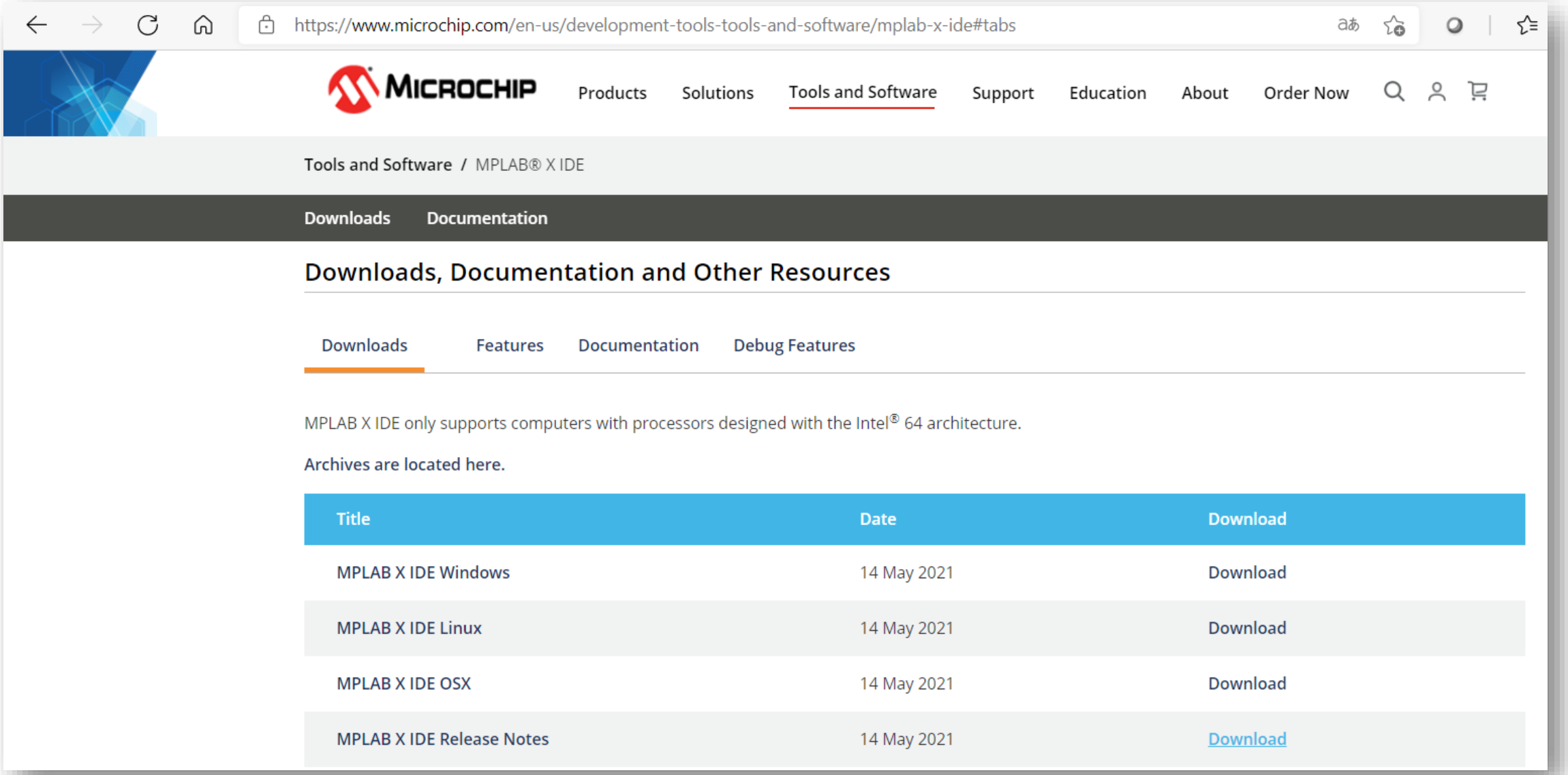


MICROCHIP

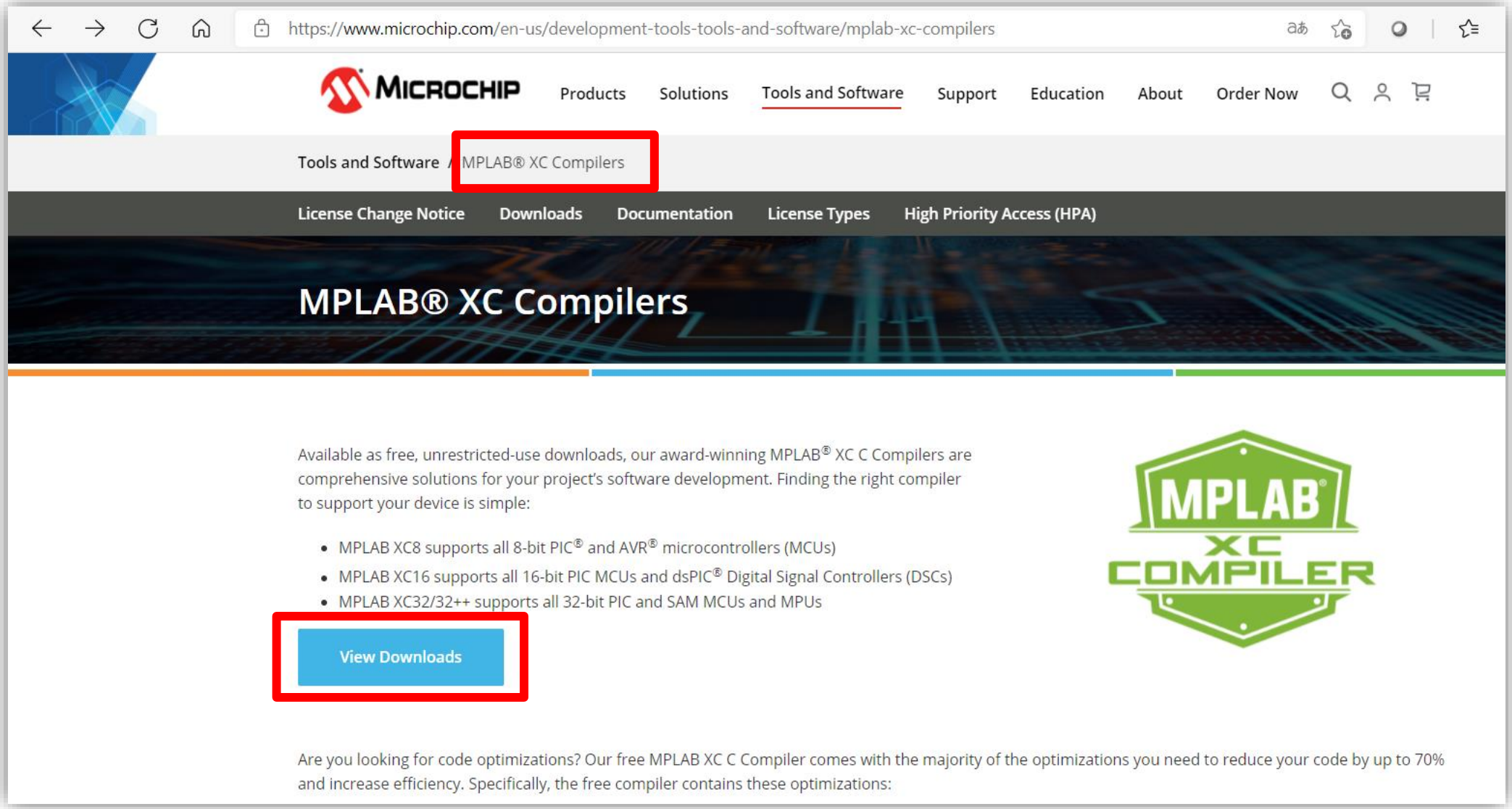
MPLAB® X IDE下載: Tools and Software 項目



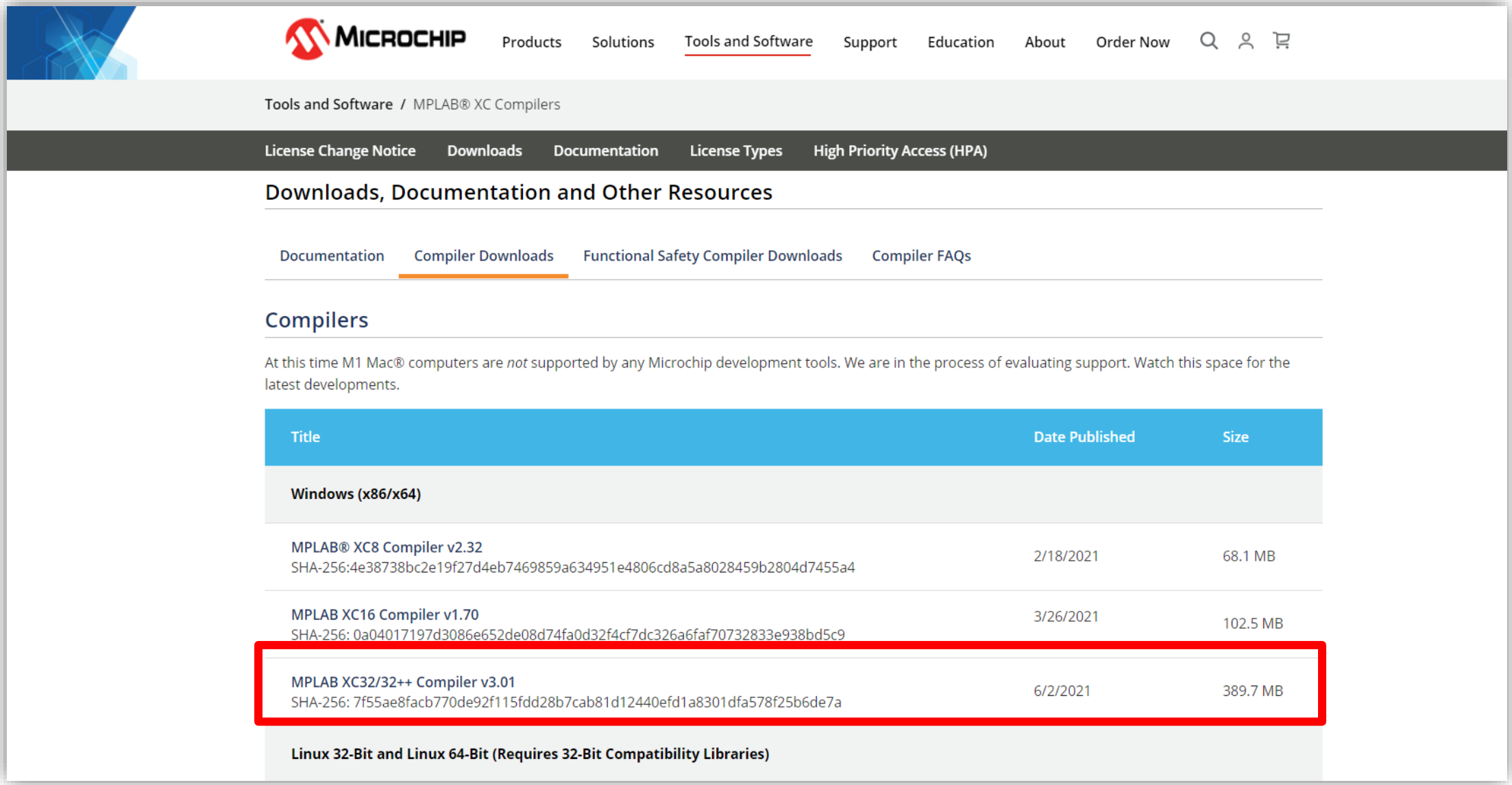
MPLAB® X IDE下載: 選擇適合你的版本



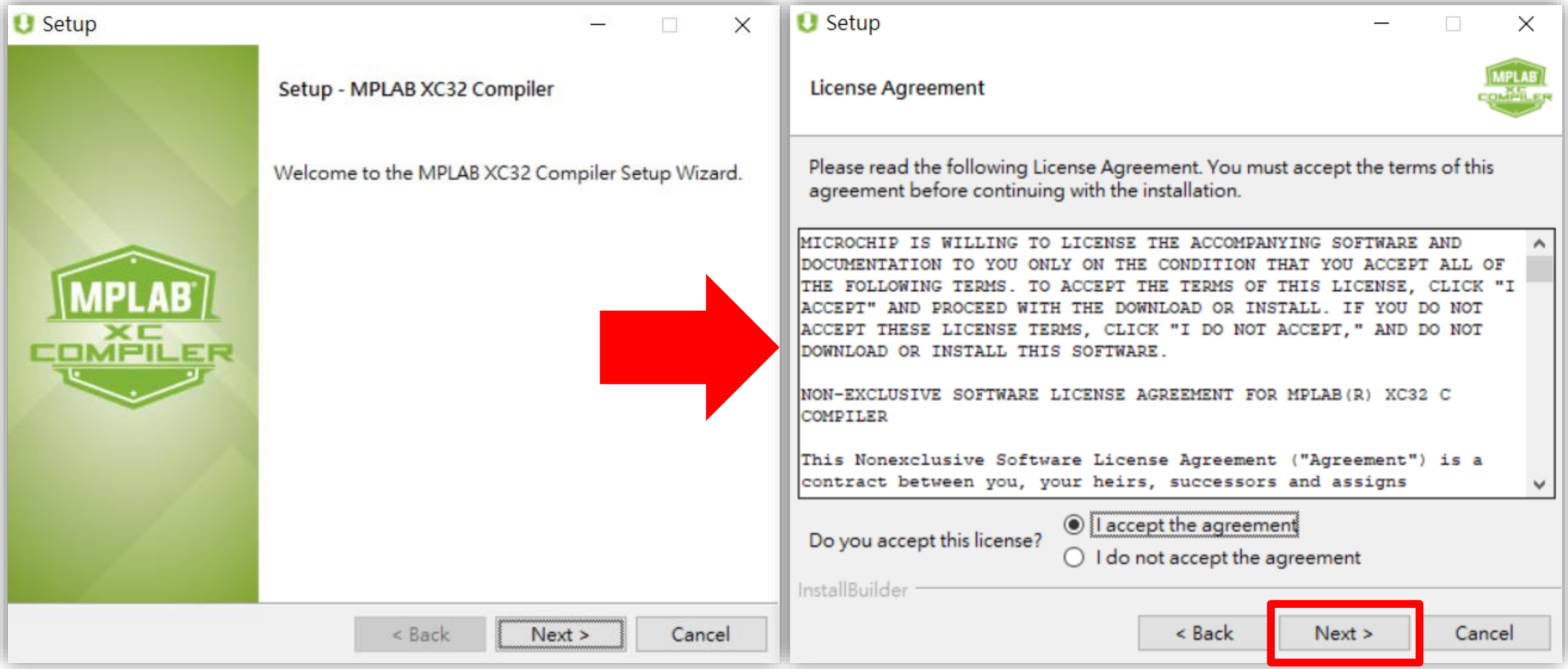
MPLAB® XC 下載: Tools and Software 項目



MPLAB® XC 下載：選擇您需要的作業系統與版本

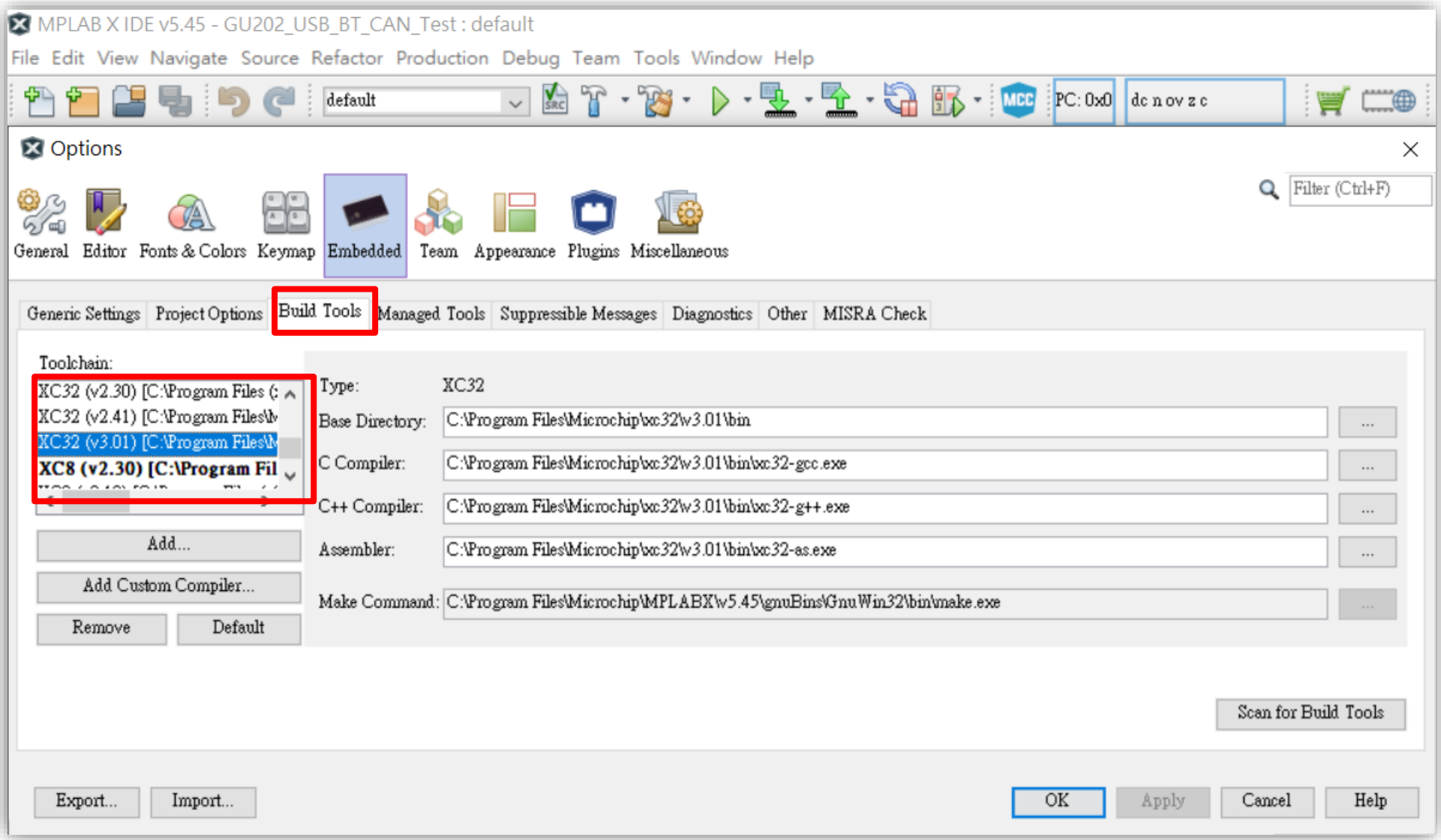


MPLAB® XC 安裝：安裝時要詳閱授權條款並須接受

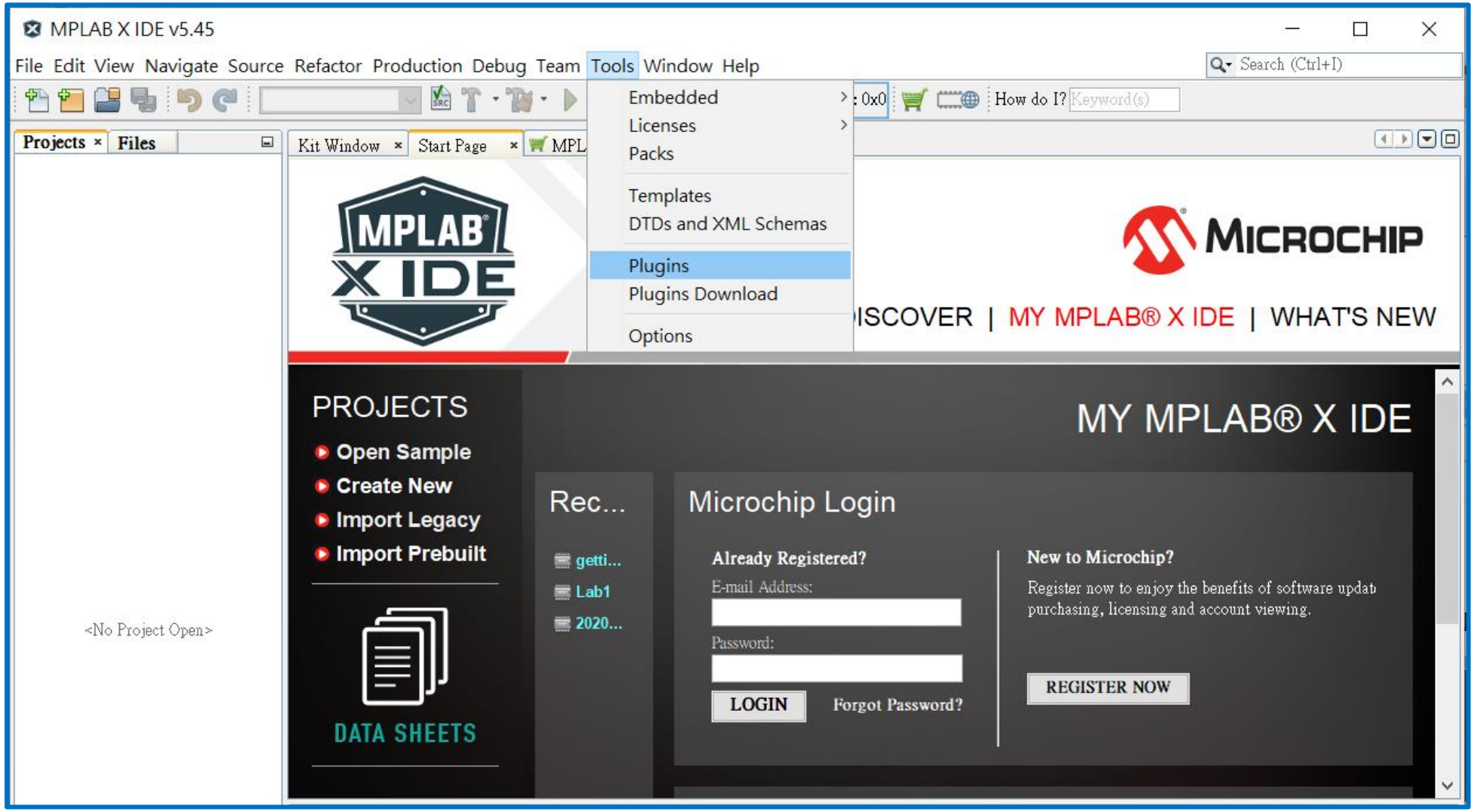


MPLAB® XC 安裝：確認安裝的有效性

- 使用 **Tools->Options->Embedded->Build Tools** 來檢查

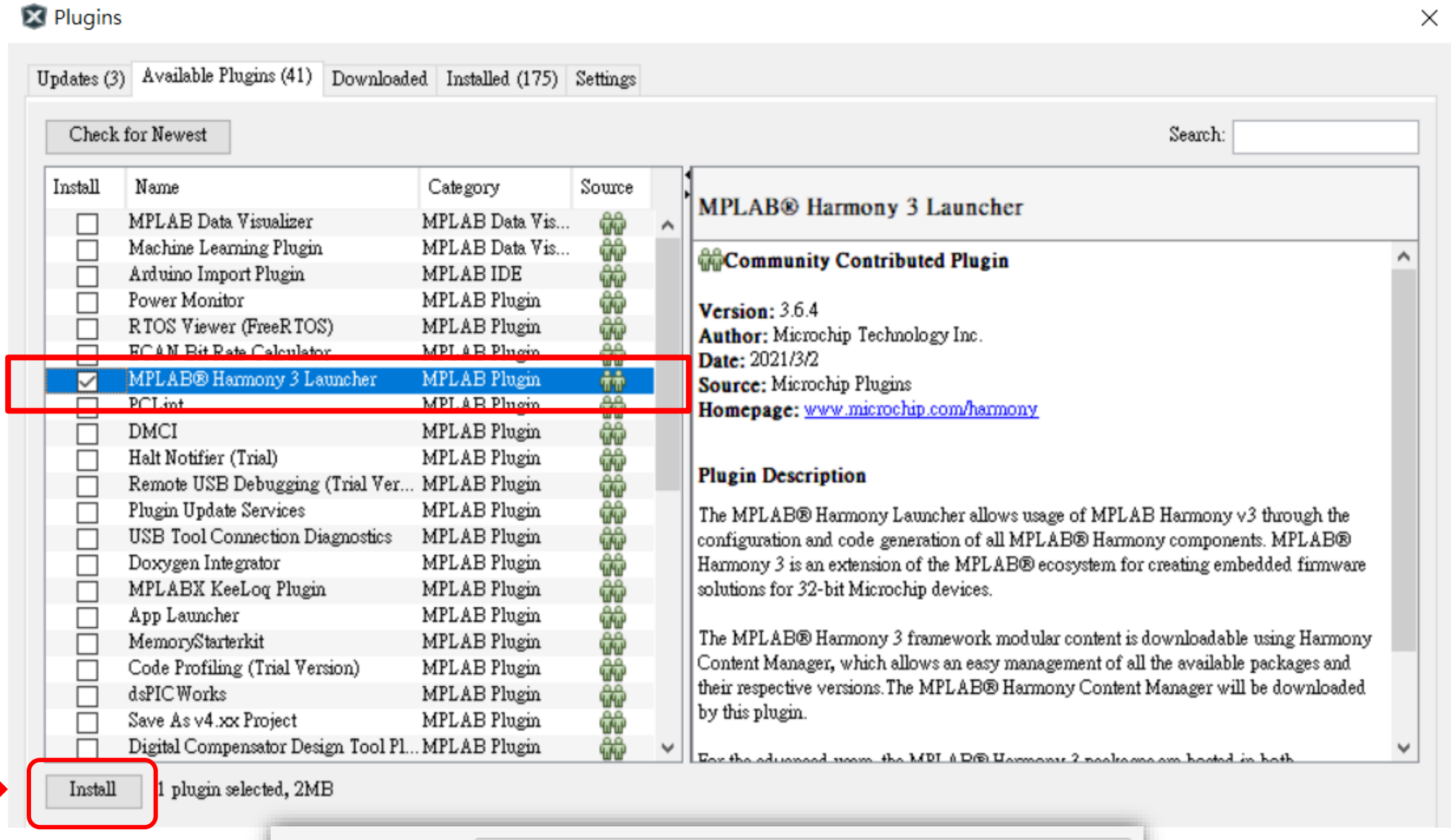


安裝Harmony 3： 選用Tools -> Plugins

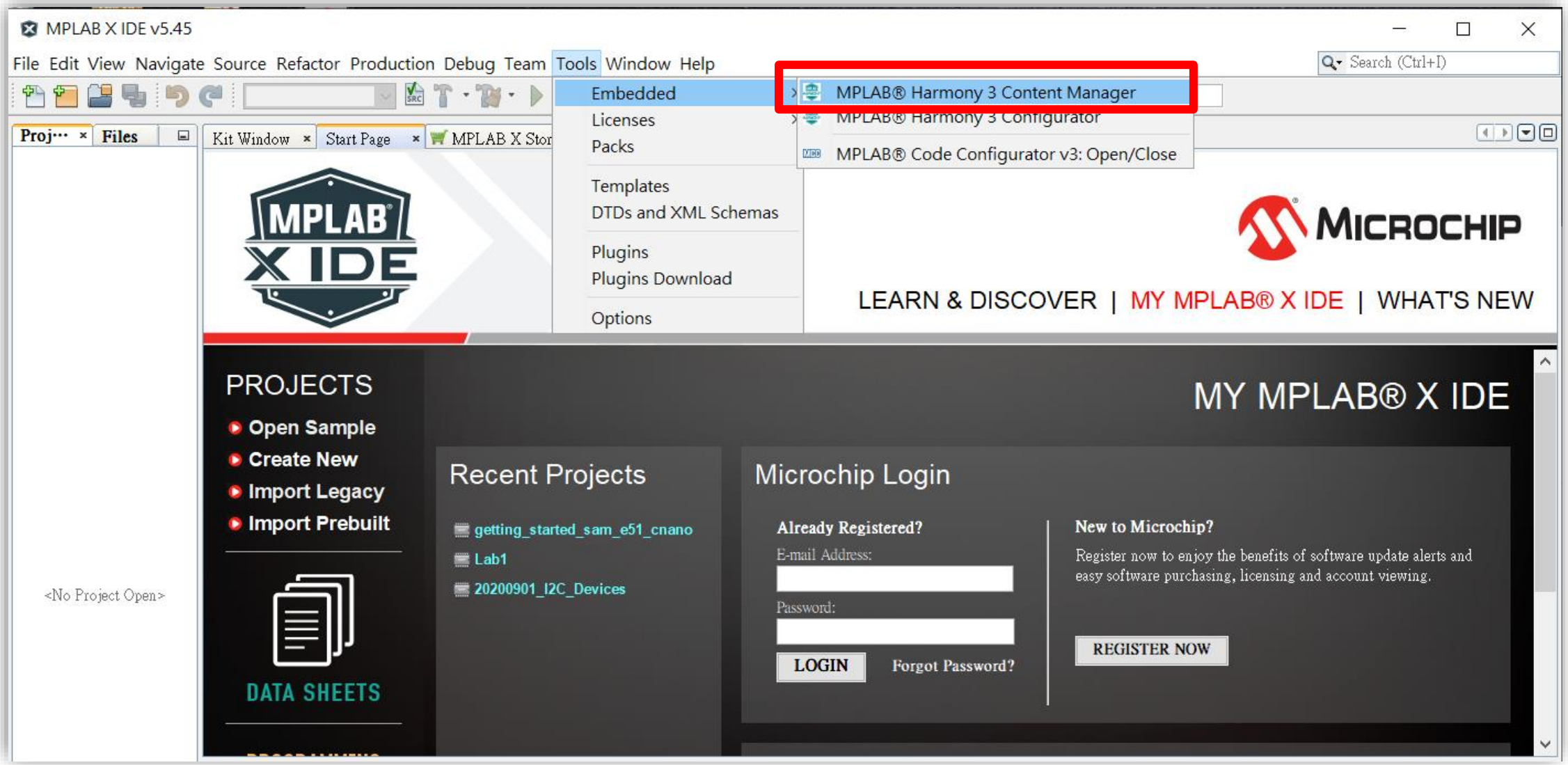


MICROCHIP

在Available Plugins中選擇MPLAB® Harmony 3 Launcher (新名稱)

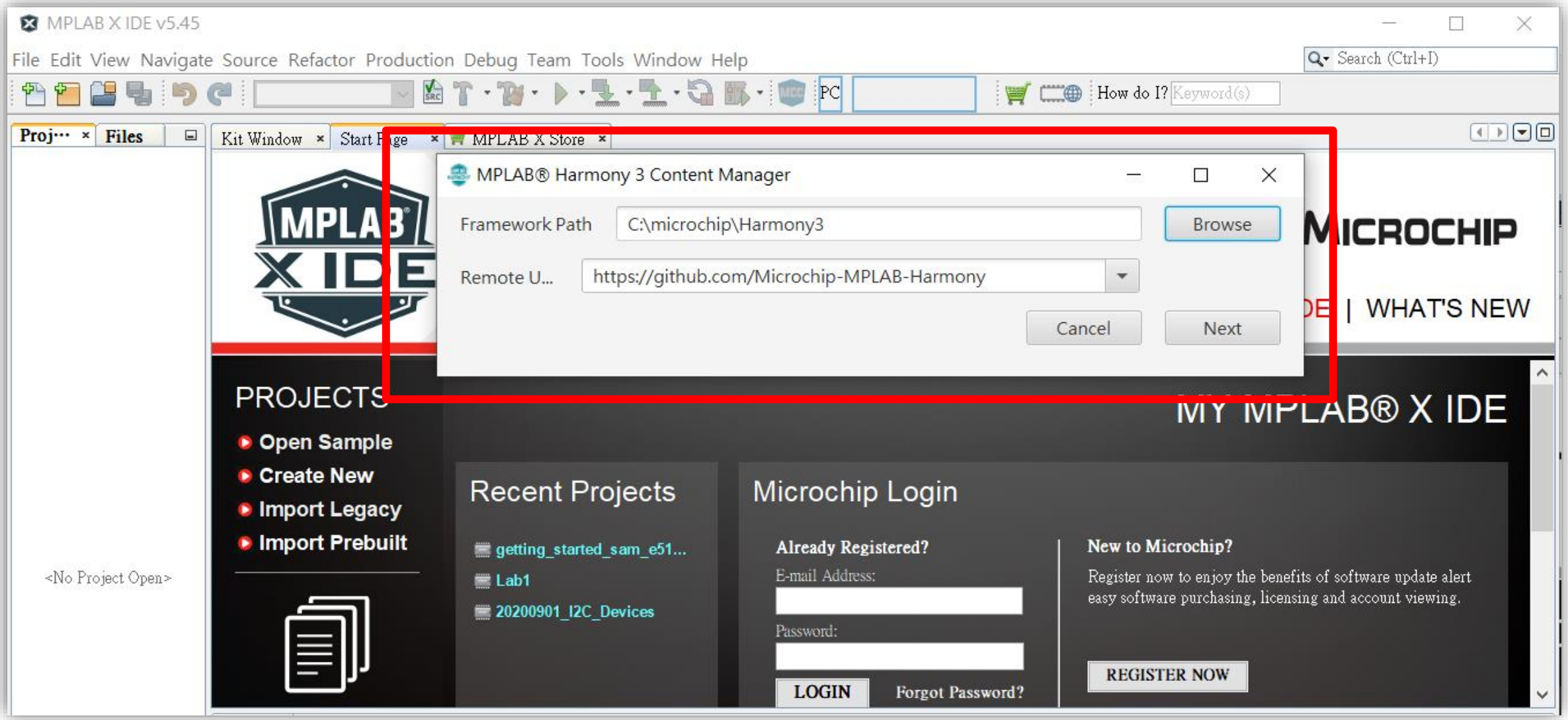


安裝Harmony 3之後以Content Manager來進行必要模組的安裝



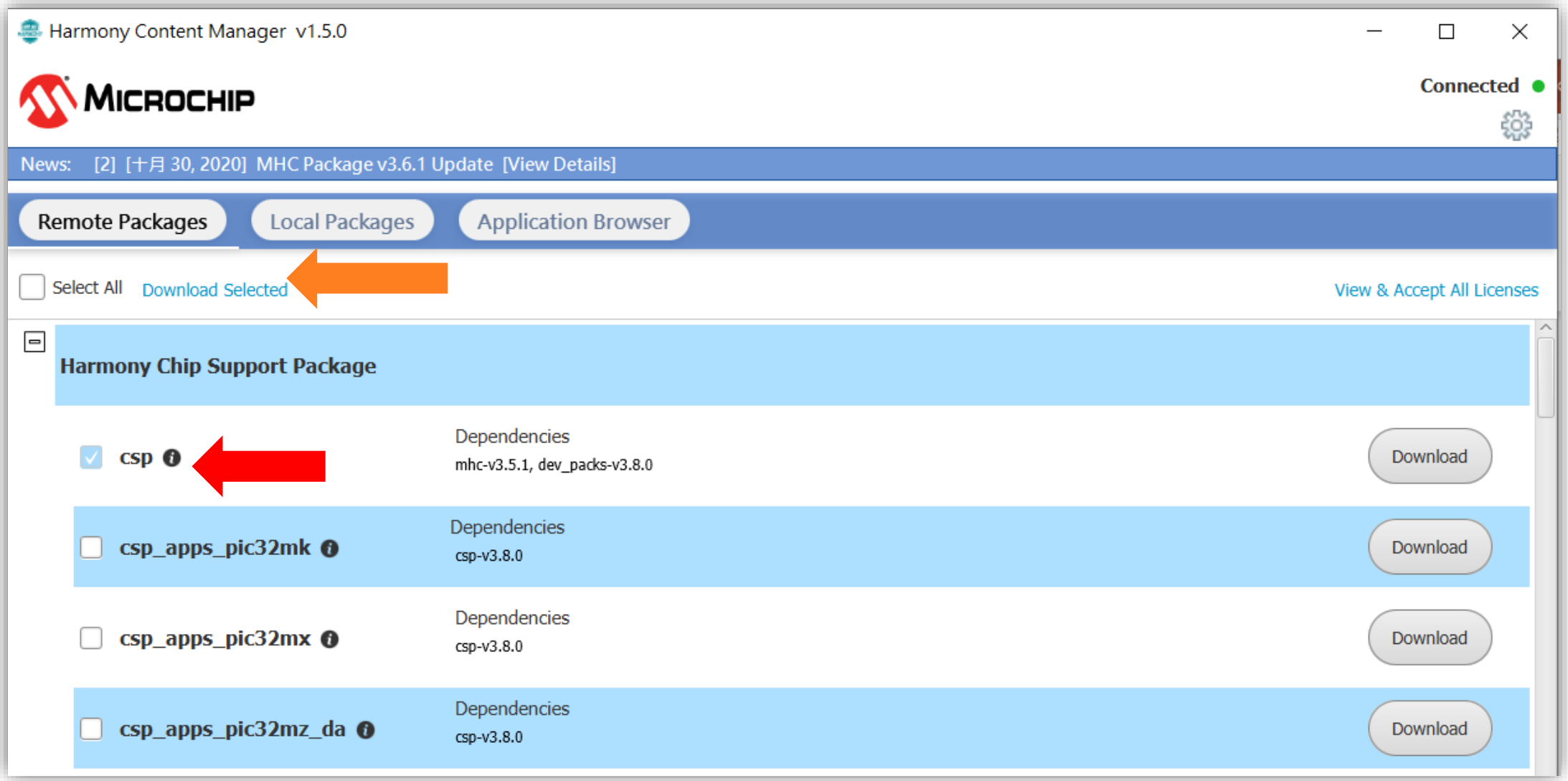
MICROCHIP

執行Content Manager要注意Harmony的安裝目錄

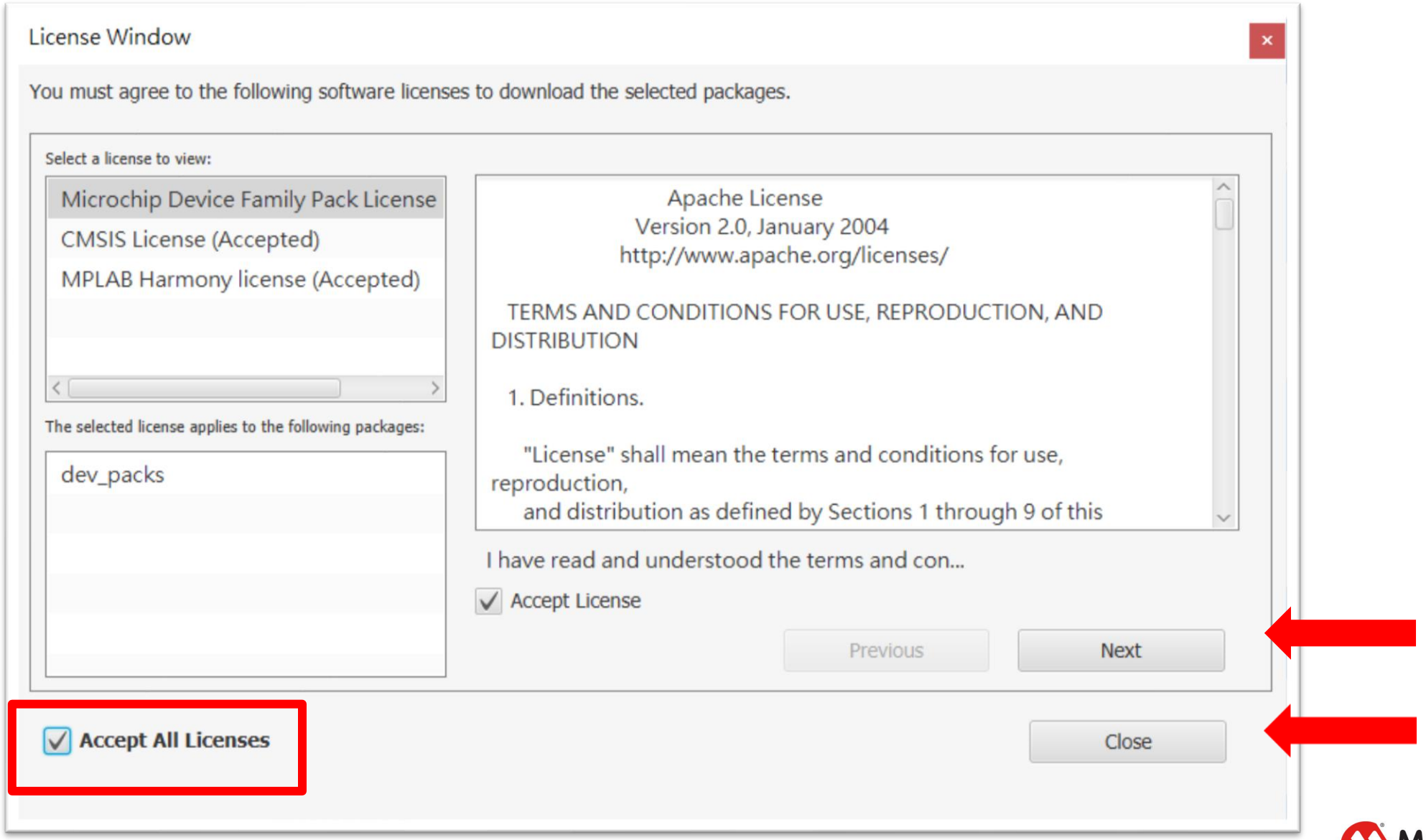


MICROCHIP

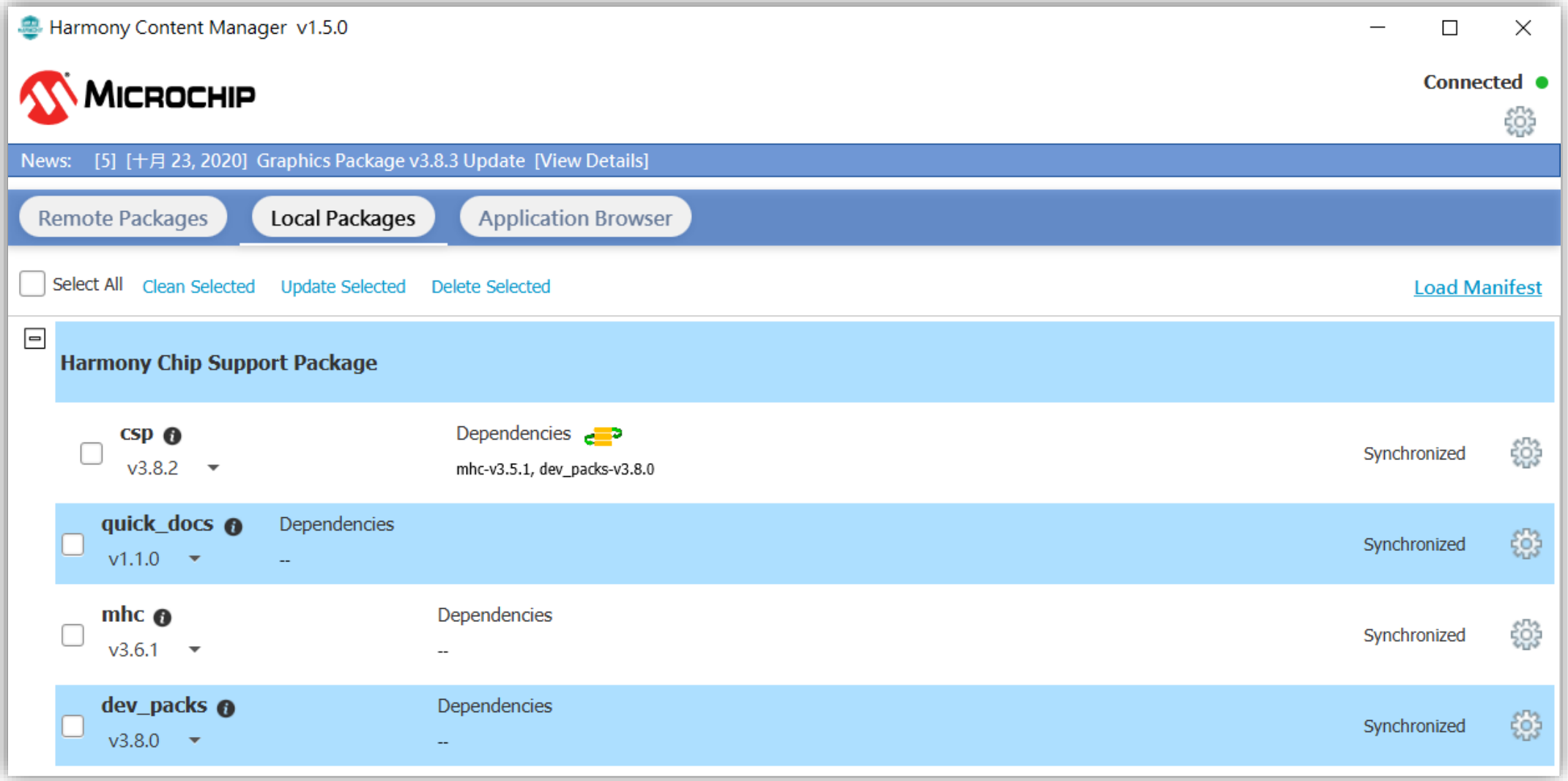
由Remote Package選擇必要安裝



勾選Accept All Licenses後開始安裝



安裝完成後可在Local Packages中檢視結果



MICROCHIP

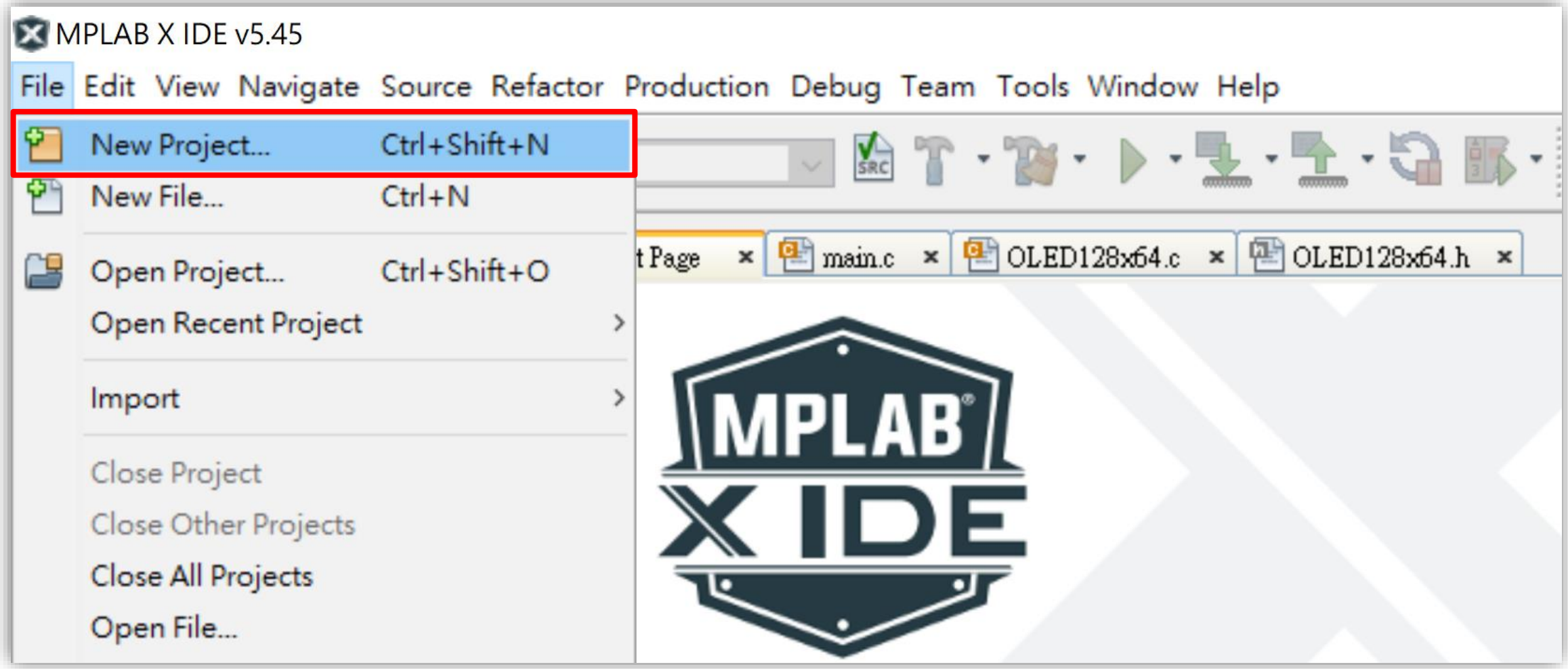
練習1：基本的I/O設定
按鍵與LED操作

練習1：基本的I/O設定：按鍵與LED操作

- 目標：
  - 練習使用SNAP or PICkit™ 4來Program ATSAMC21G17A-AU
  - 認識如何使用Harmony 3 Configurator來進行基礎的應用程式建構
    - Clock Configurator
    - Pin Configurator
  - 學習與ATSAMD21 Curiosity Nano相同的LED與Switch的操作
    - LED1 – PB10
    - SW1 – PB11
  - 動作要求：
    - 確認ATSAMC21G17A的CPU核心運行於48 MHz
    - LED1接受SW1的控制，SW1按下時，LED1做一次轉態的動作
    - → SW1放開時，LED1不要有任何變化

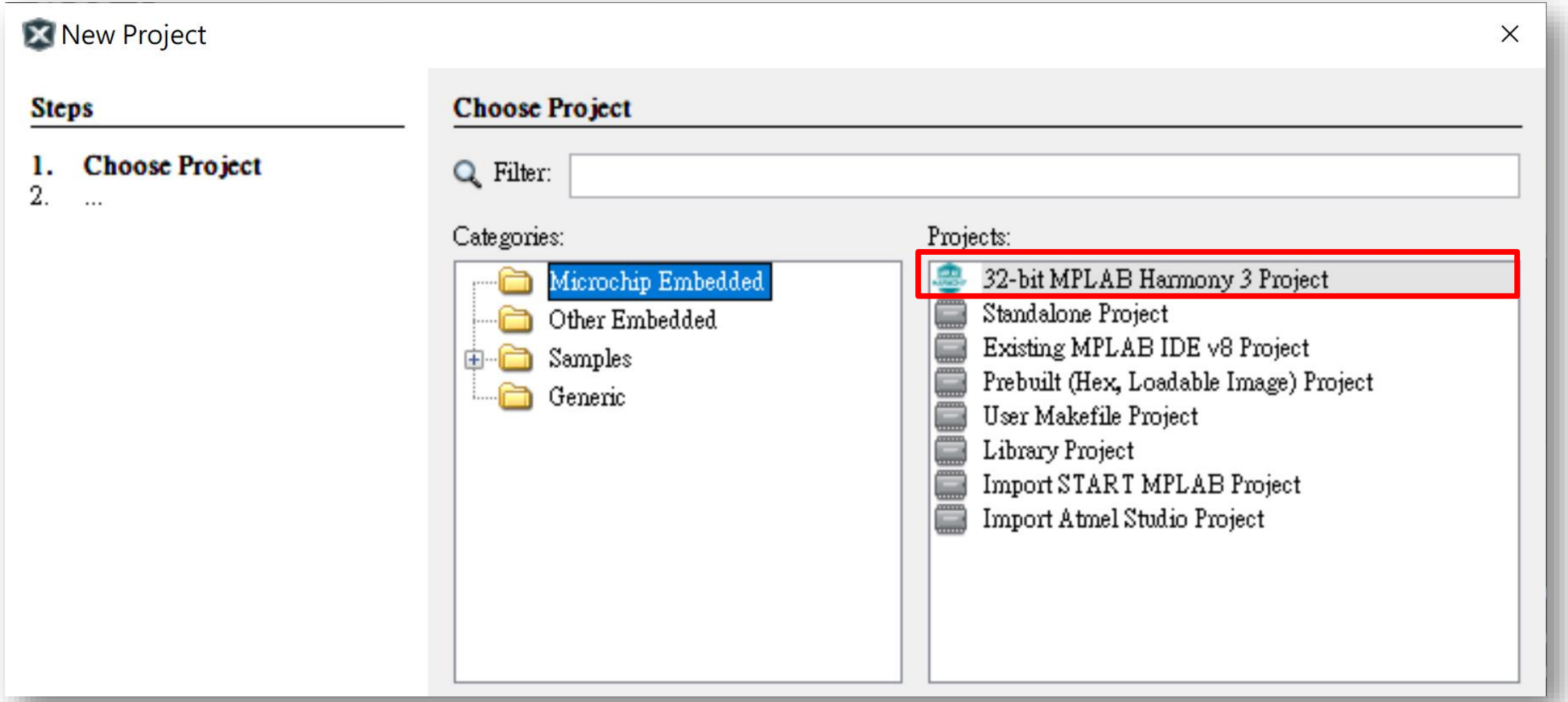
練習1：基本的I/O設定：按鍵與LED操作

- 開啟MPLAB® X IDE並且使用File-> New Project來創建新專案



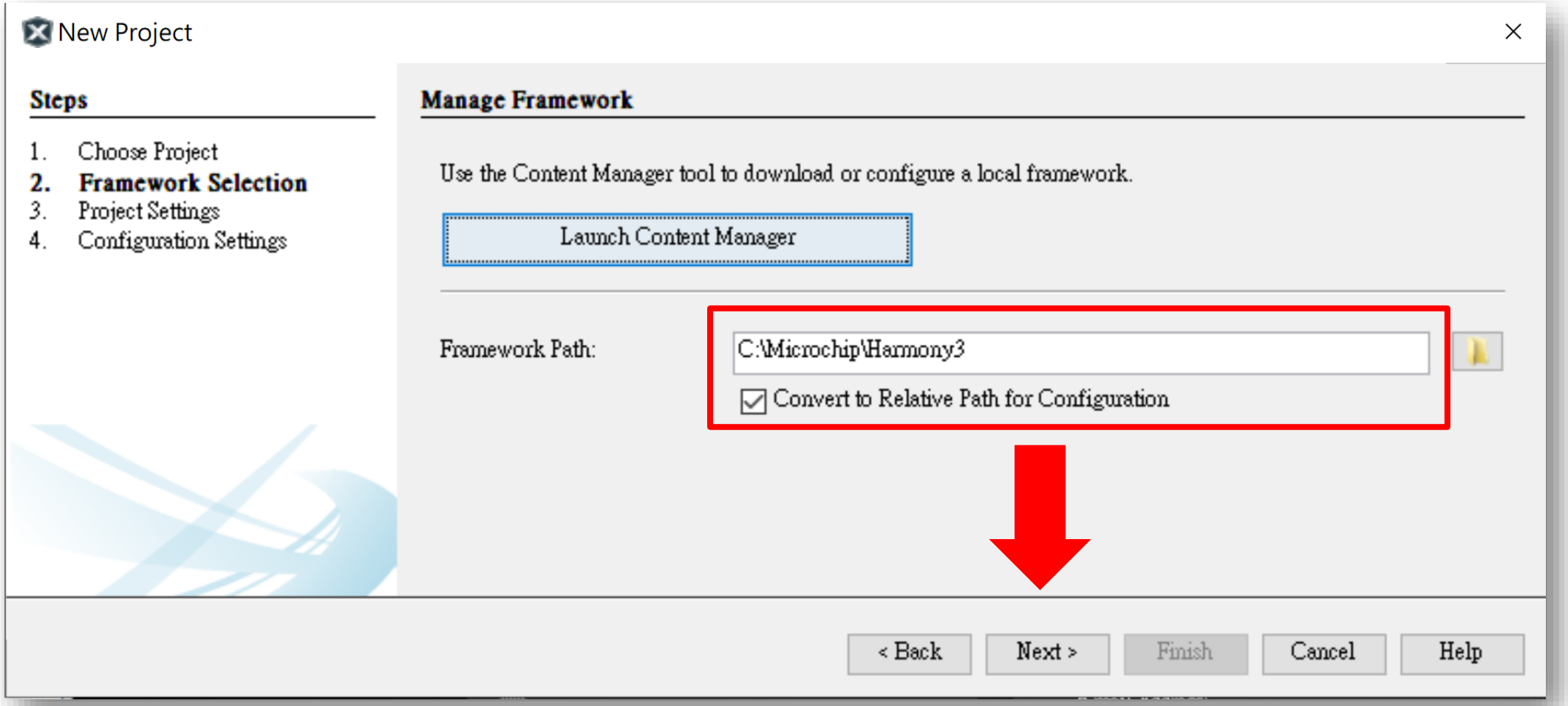
練習1：基本的I/O設定：按鍵與LED操作

- 在專案項目中選用“32-bit MPLAB Harmony 3 Project”



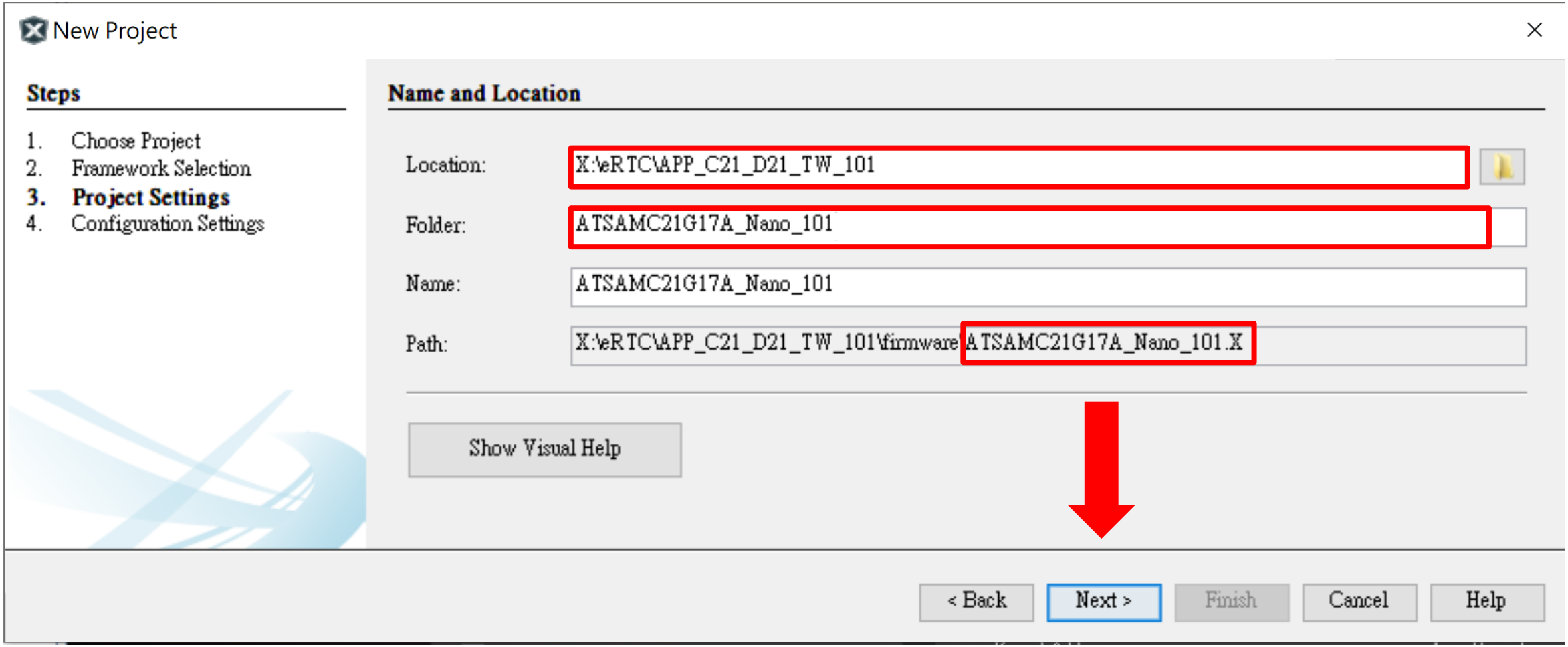
練習1：基本的I/O設定：按鍵與LED操作

- 確認Harmony 3安裝的路徑是對的



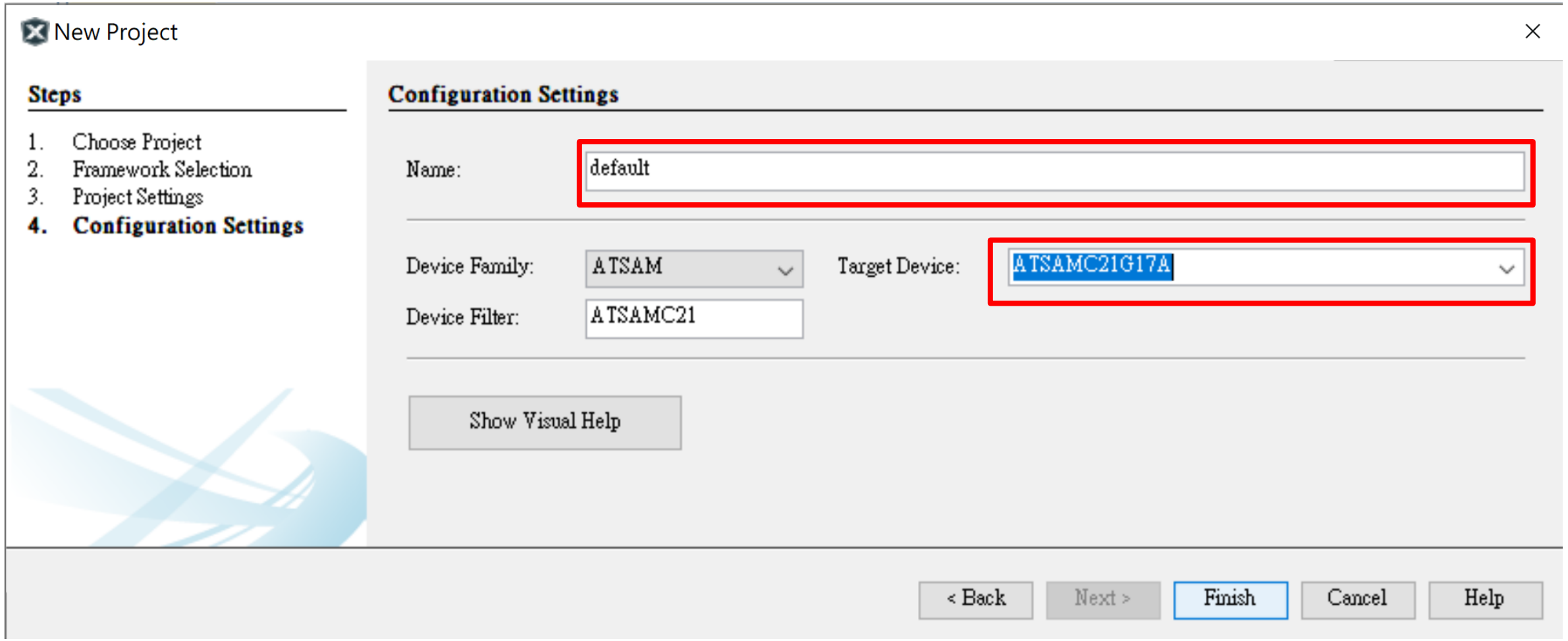
練習1：基本的I/O設定：按鍵與LED操作

- 將專案選定儲存位置、在firmware中的目錄名稱等



練習1：基本的I/O設定：按鍵與LED操作

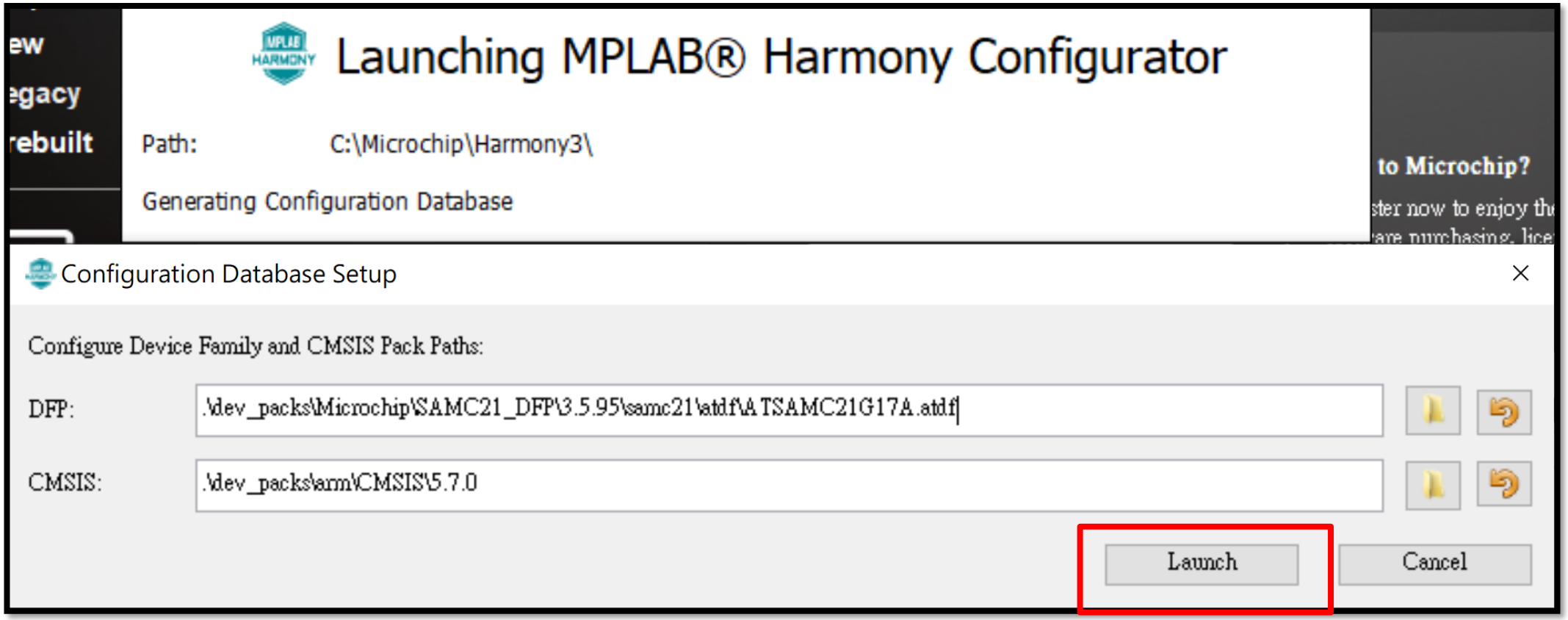
- **Configuration的名稱可用預設，但要選對MCU**



MICROCHIP

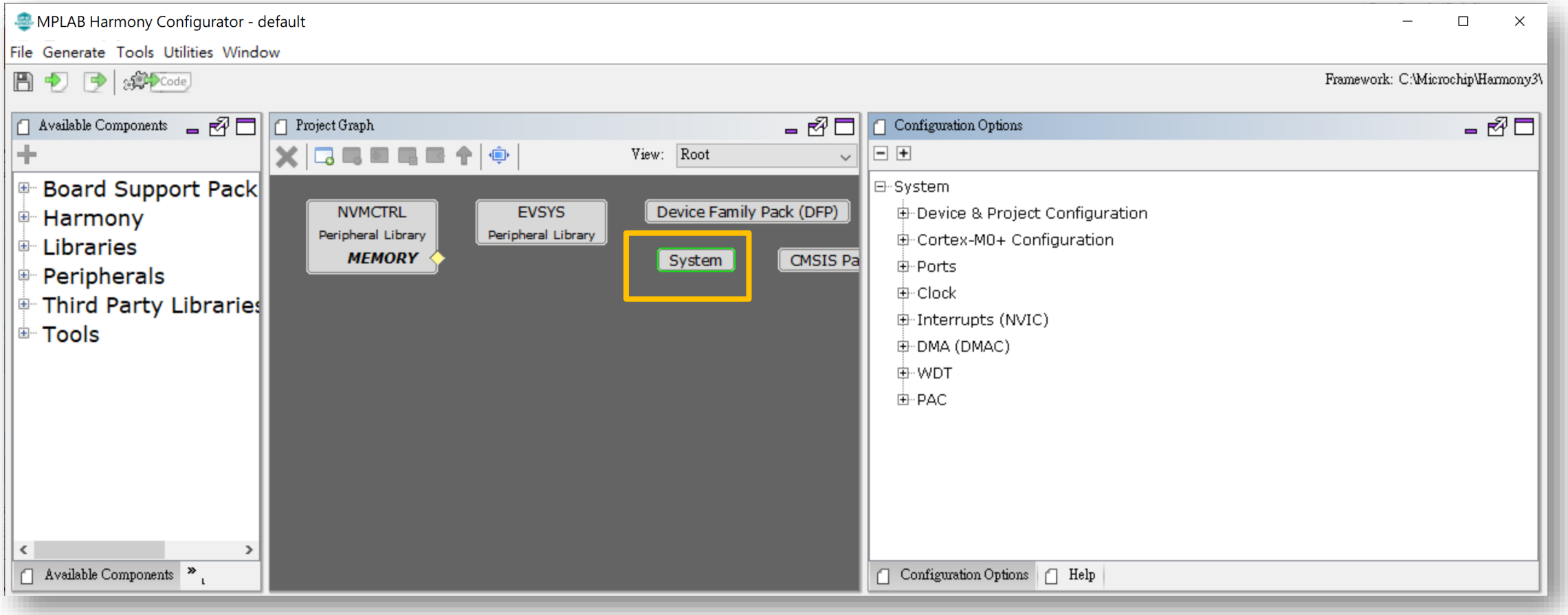
練習1：基本的I/O設定：按鍵與LED操作

- 最後，會提示使用的**Device Family Pack**以及**CMSIS Pack**位置及版本，按下**Launch**後就會帶出**Harmony 3 Configurator**



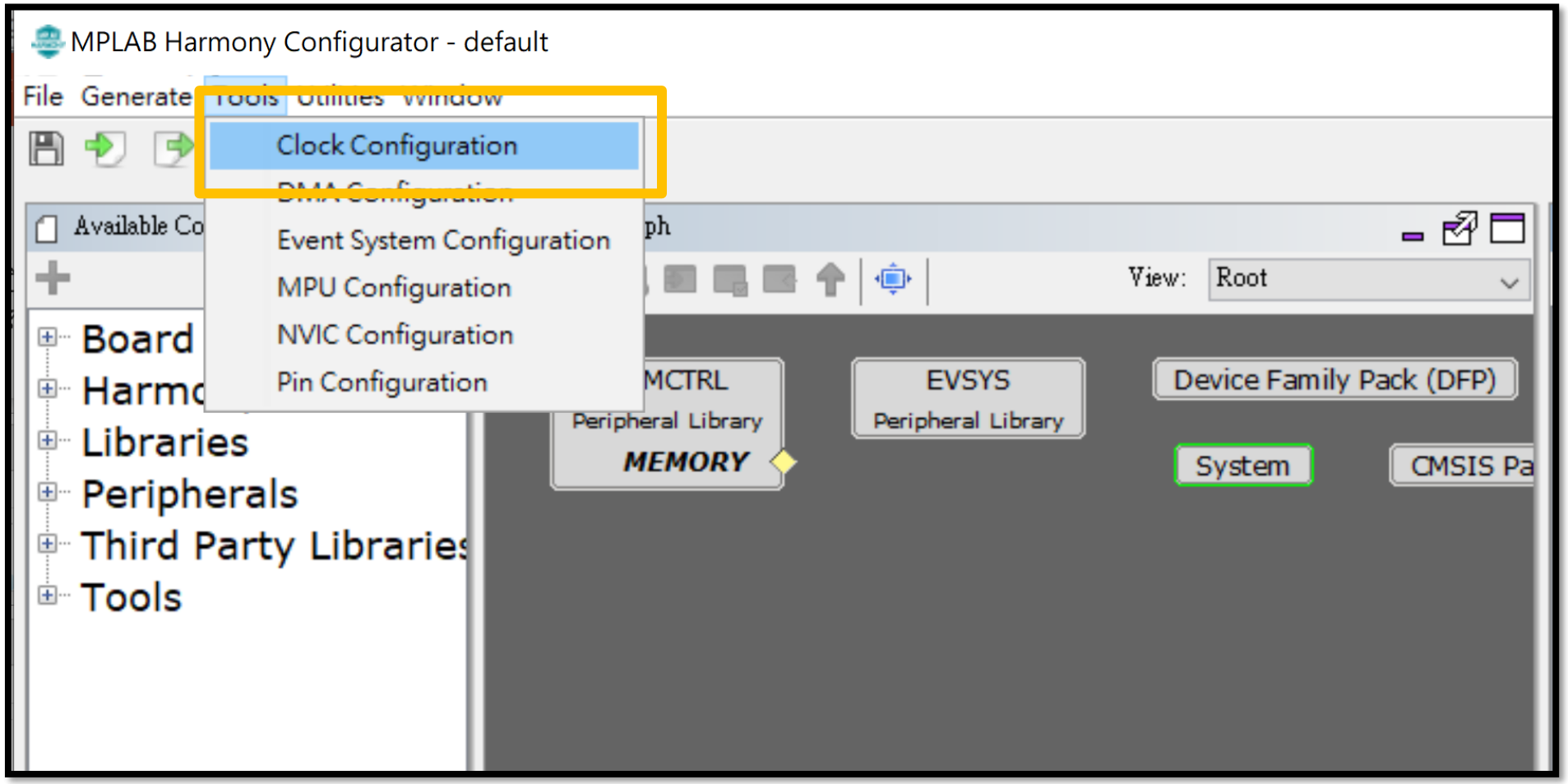
練習1：基本的I/O設定：按鍵與LED操作

- 載入Harmony 3 Configurator後會有幾個必須且預設的模組
- System模組是一般比較常需要規劃的



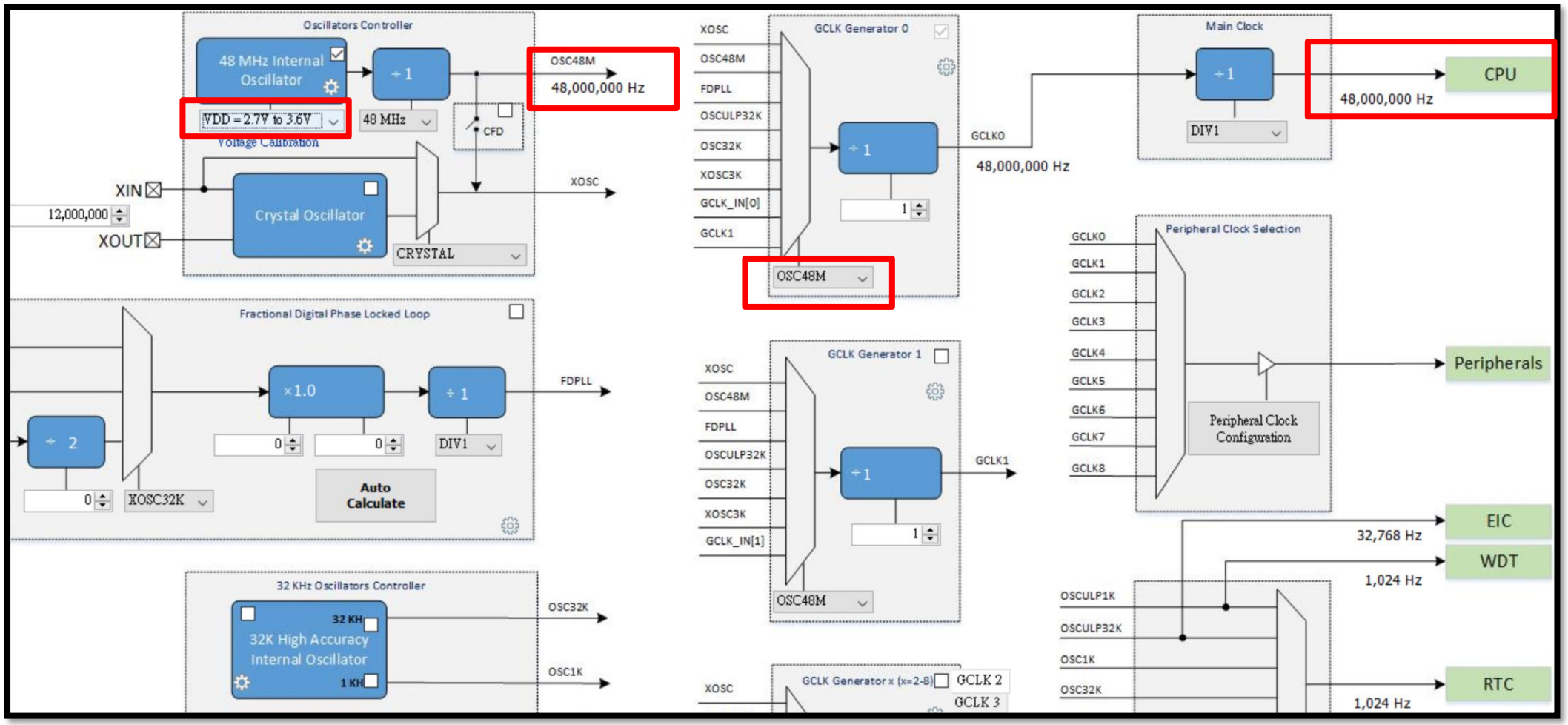
練習1：基本的I/O設定：按鍵與LED操作

- 檢查Clock的設定是否正確：目標 48 MHz



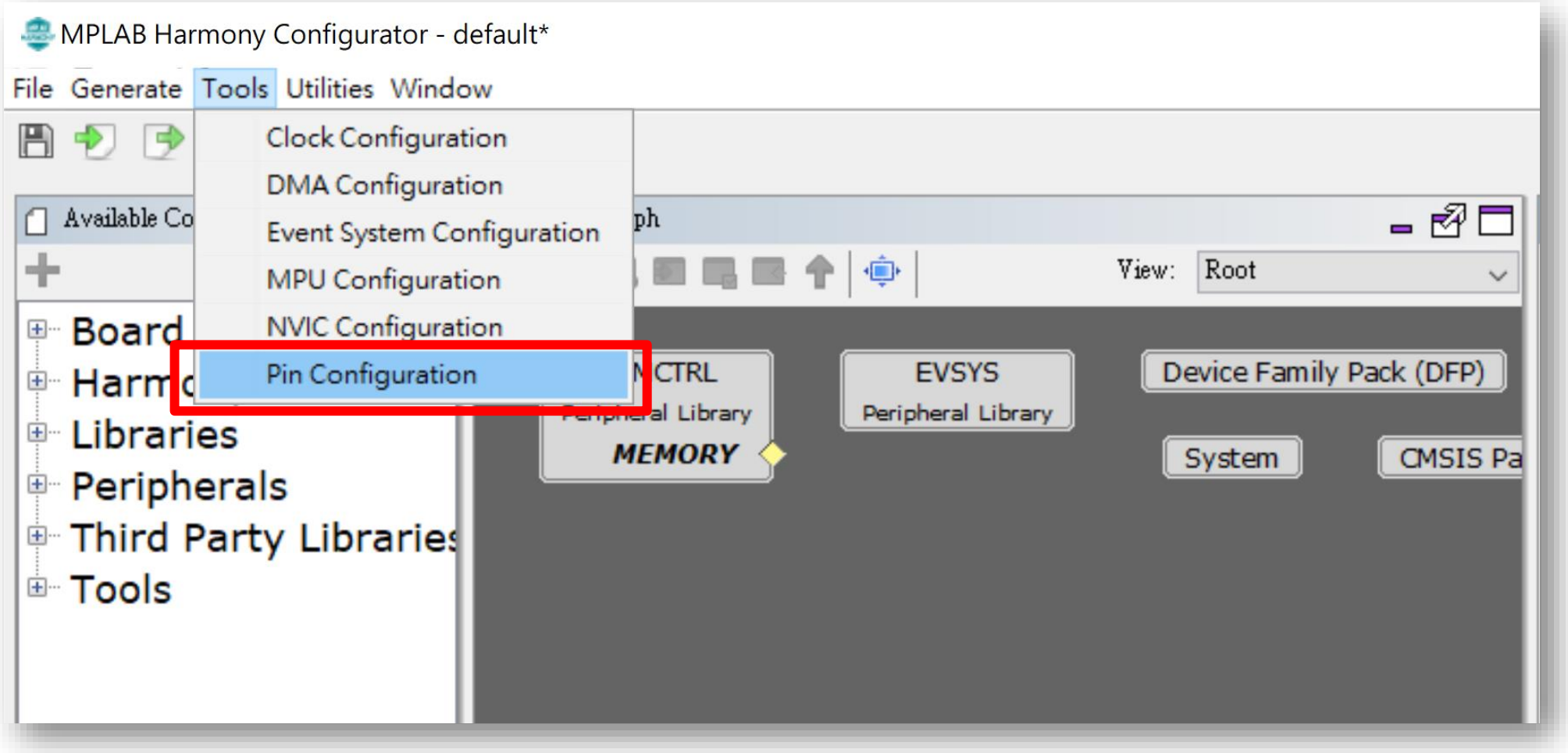
練習1：基本的I/O設定：按鍵與LED操作

- 可以使用預設值，但是把**48 MHz Internal OSC VDD**設對



練習1：基本的I/O設定：按鍵與LED操作

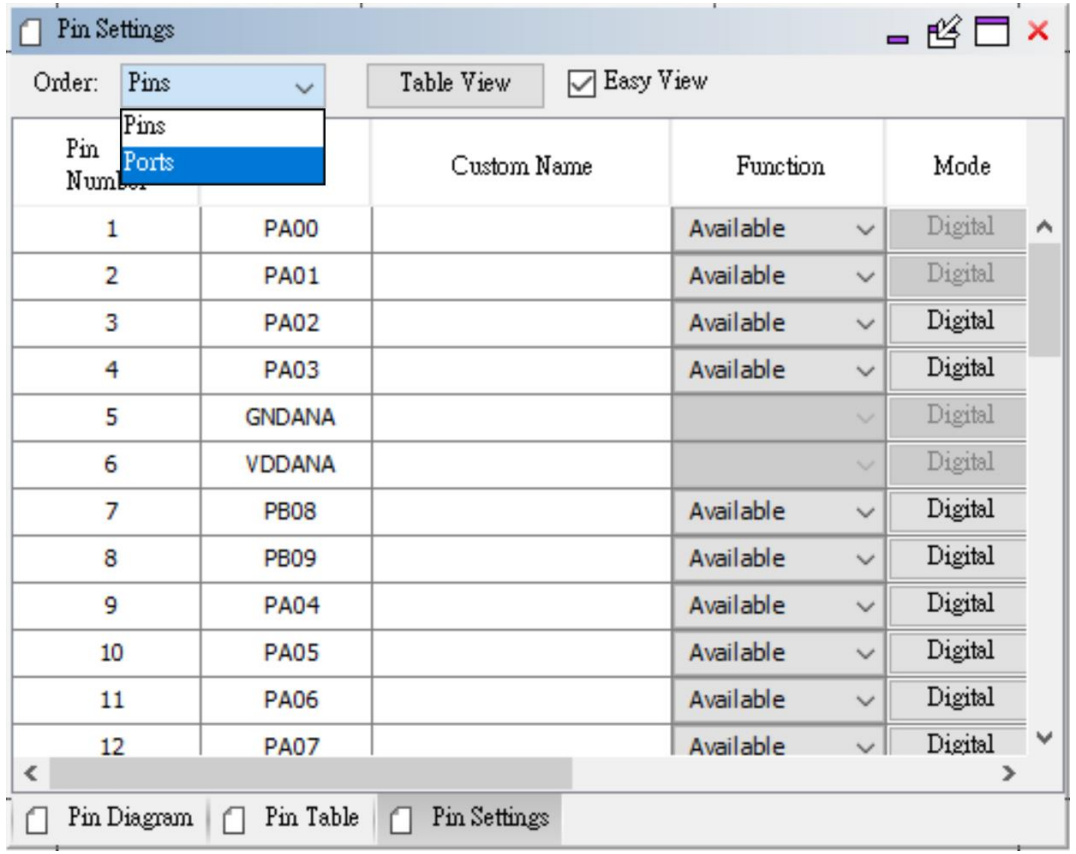
- 因為要做LED及按鍵的互動練習，所以一定要先規劃腳位

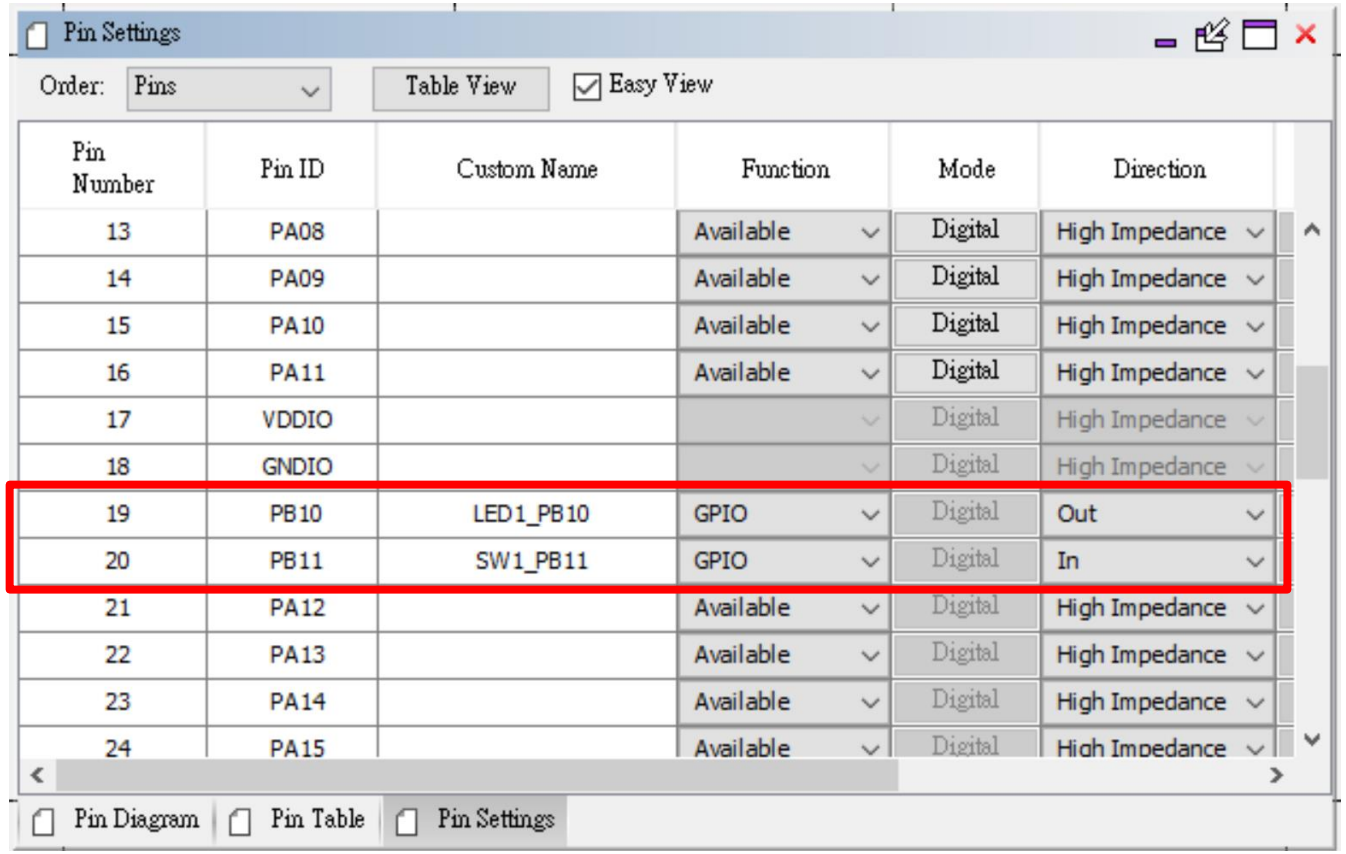


MICROCHIP

練習1：基本的I/O設定：按鍵與LED操作

- 不管事使用Pin或是Port來排列，將PB10 & PB11設定成GPIO。
  當然 … Custom Name可以設定成好記的名稱

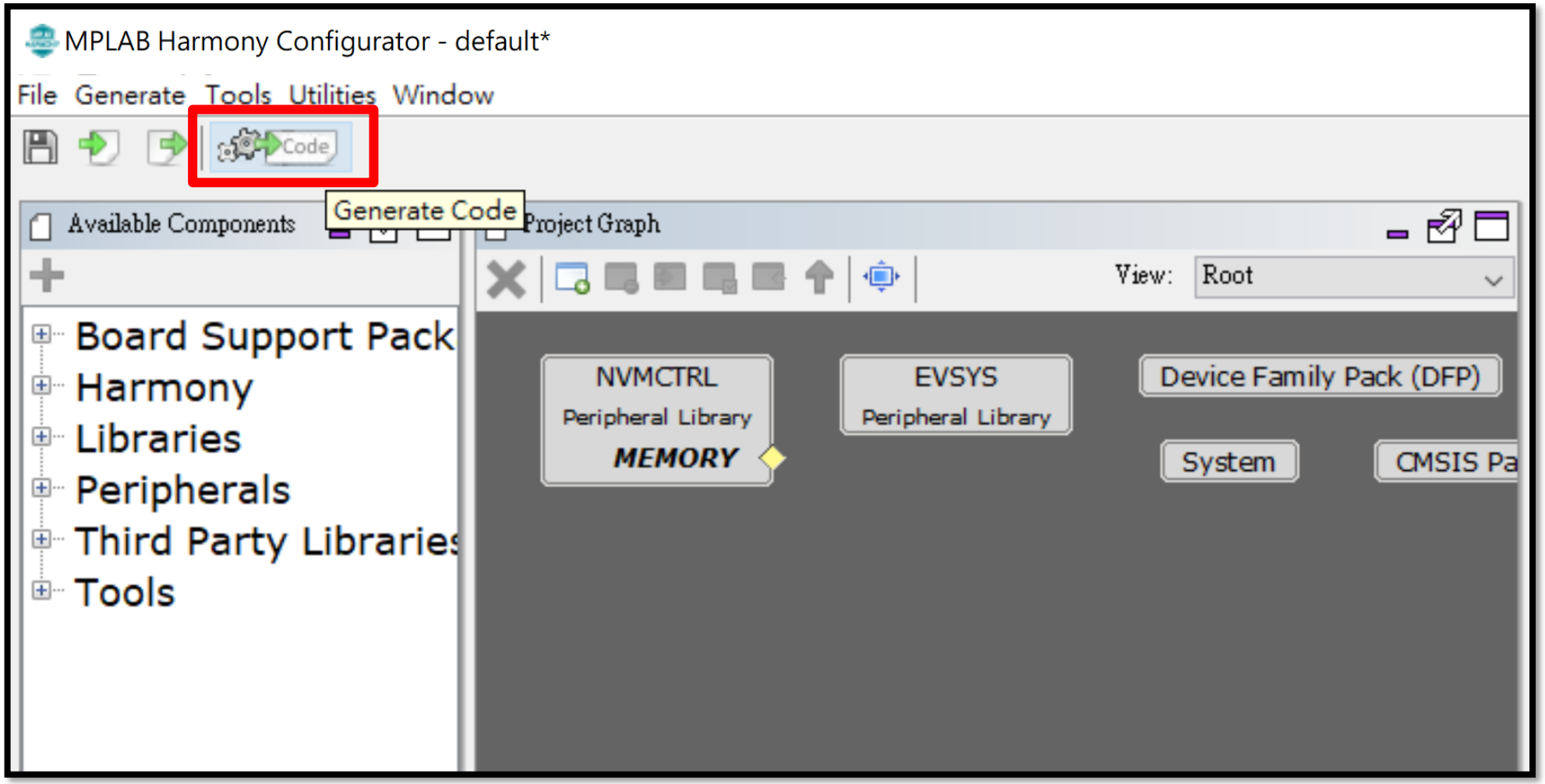




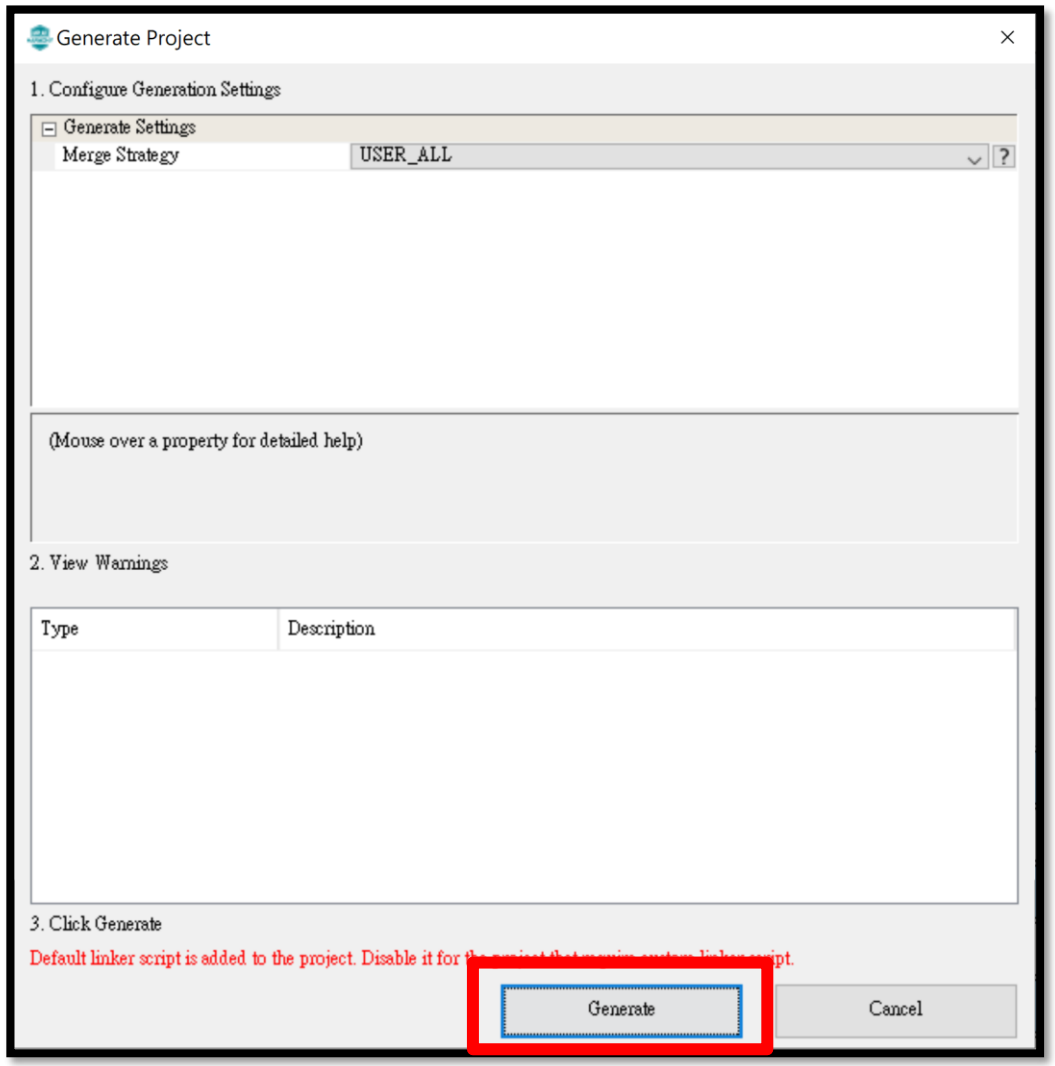
MICROCHIP

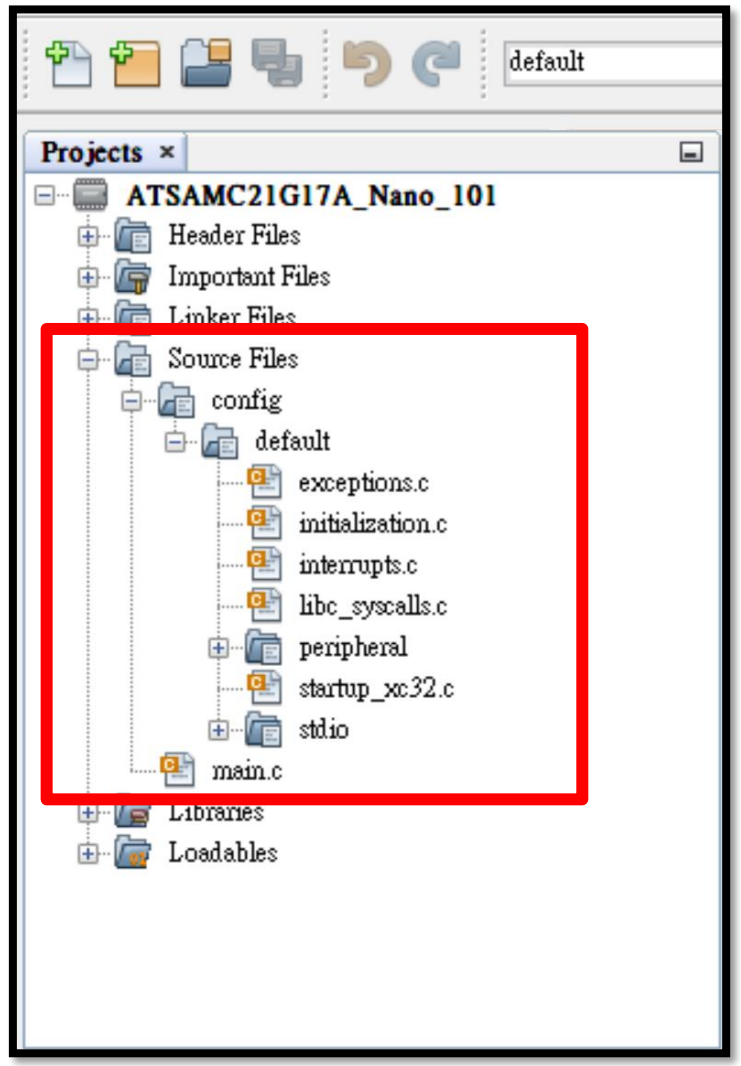
練習1：基本的I/O設定：按鍵與LED操作

- 當一切就緒，使用“**Generate Code**”按鈕來產生程式碼



練習1：Generate Code之後可在專案中看到相關檔案

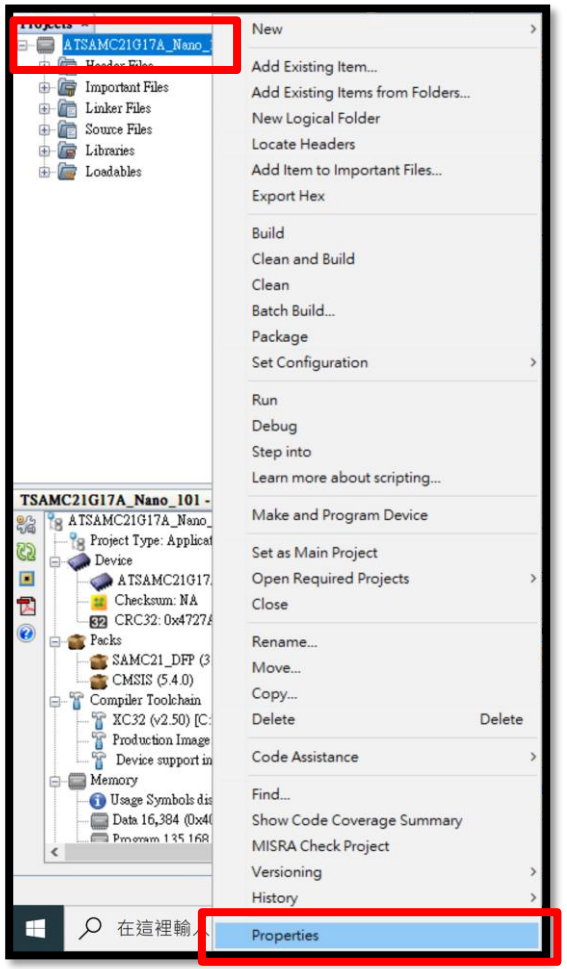


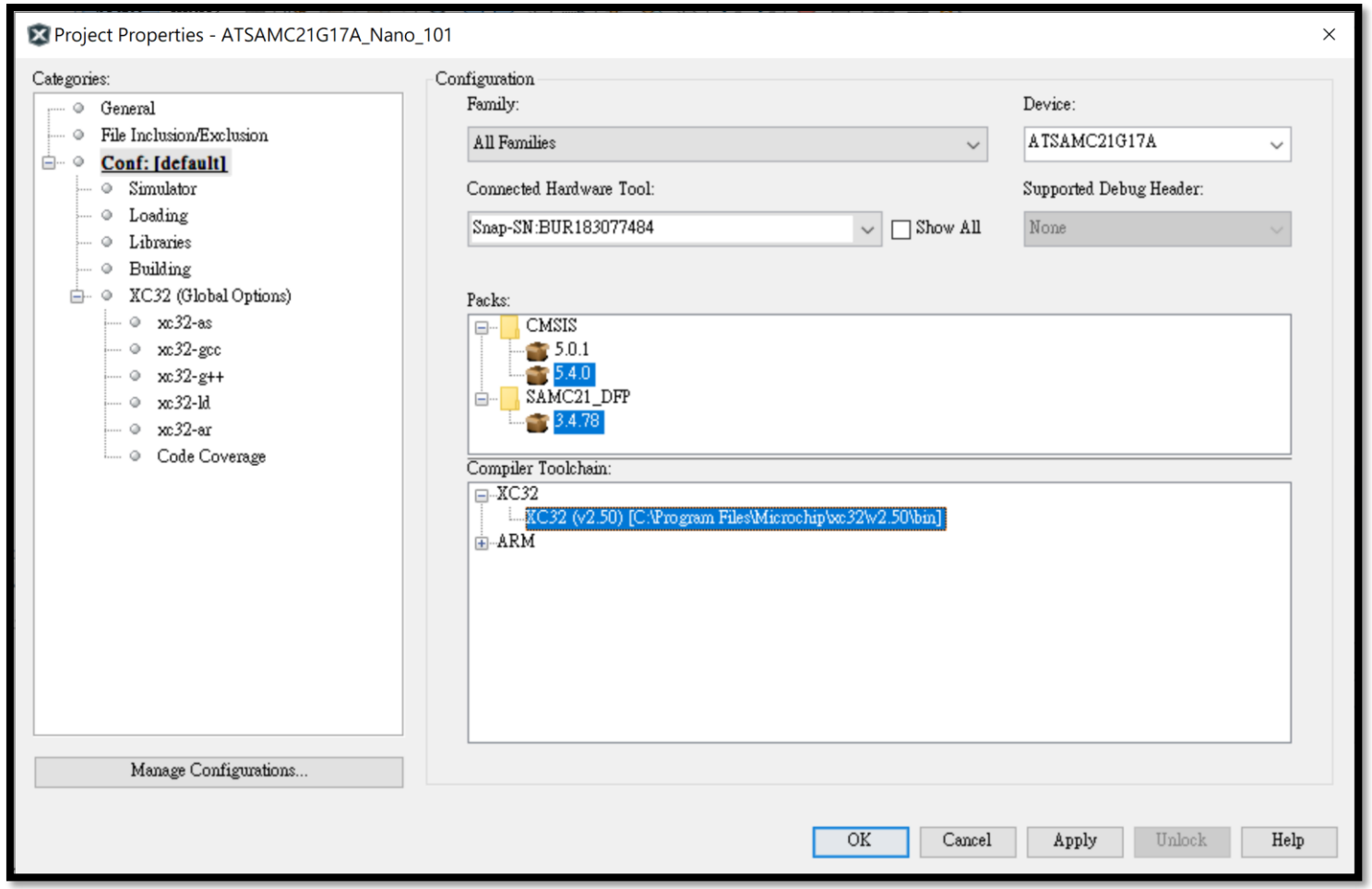


MICROCHIP

練習1：基本的I/O設定：按鍵與LED操作

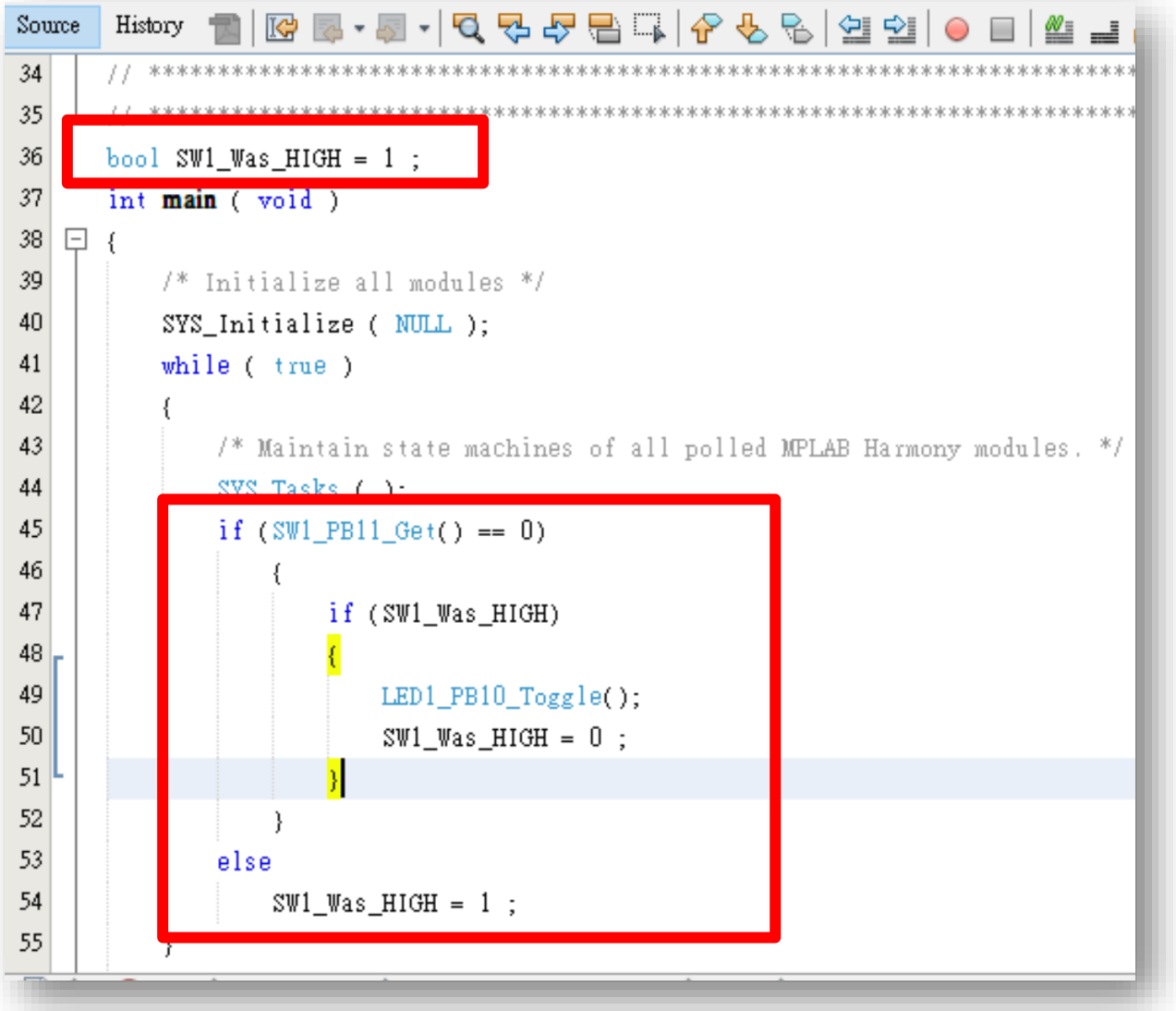
- 在專案名稱按下右鍵，選擇Properties來設定專案，選工具等





MICROCHIP

練習1：基本的I/O設定：按鍵與LED操作

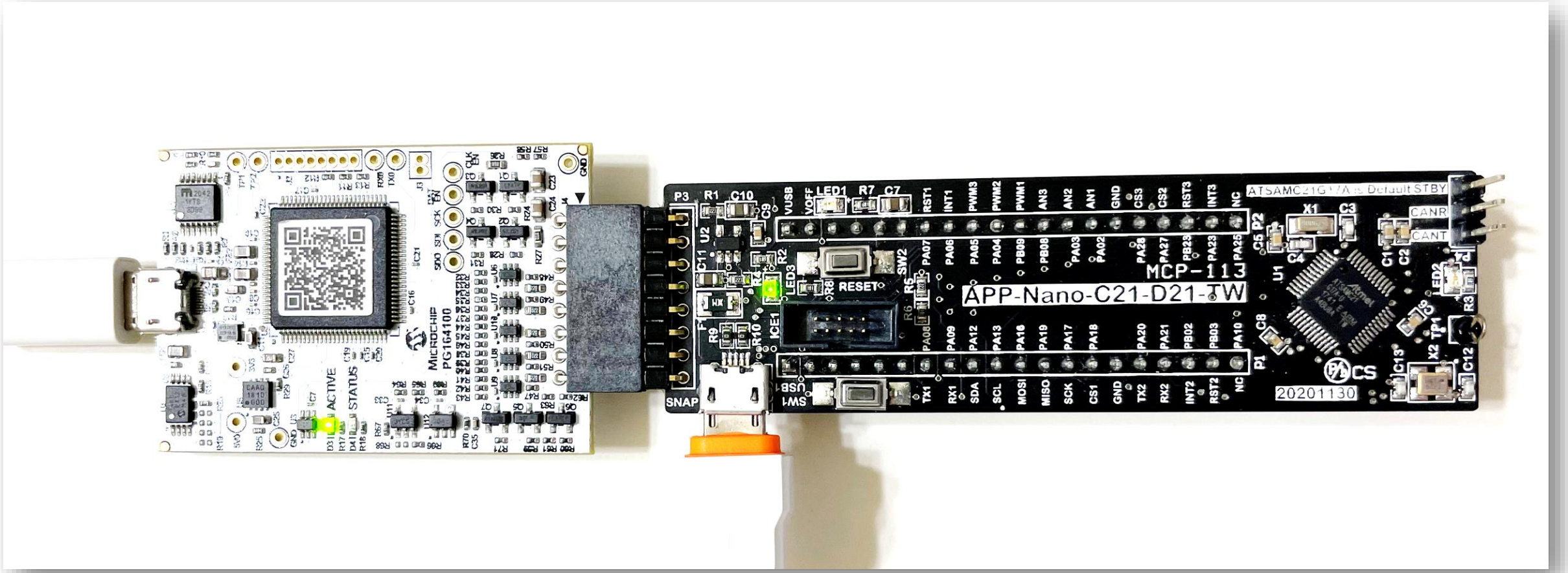


```
34  // ******************************************************************************
35  // ******************************************************************************
36  bool SW1_Was_HIGH = 1 ;
37  int main ( void )
38  {
39      /* Initialize all modules */
40      SYS_Initialize ( NULL );
41      while ( true )
42      {
43          /* Maintain state machines of all polled MPLAB Harmony modules. */
44          SYS_Tasks ( );
45          if (SW1_PB11_Get() == 0)
46              {
47                  if (SW1_Was_HIGH)
48                  {
49                      LED1_PB10_Toggle();
50                      SW1_Was_HIGH = 0 ;
51                  }
52              }
53          else
54              SW1_Was_HIGH = 1 ;
55      }
```

- 在main.c中鍵入必要程式來完成練習一要求功能
  - SW1按下一次，LED1就Toggle一次
  - LED1只有在按下時才能有變化
- 練習一看來不聰明，但是為了為練習2的EIC以及T0的PLIB操作鋪路

練習1：基本的I/O設定：按鍵與LED操作

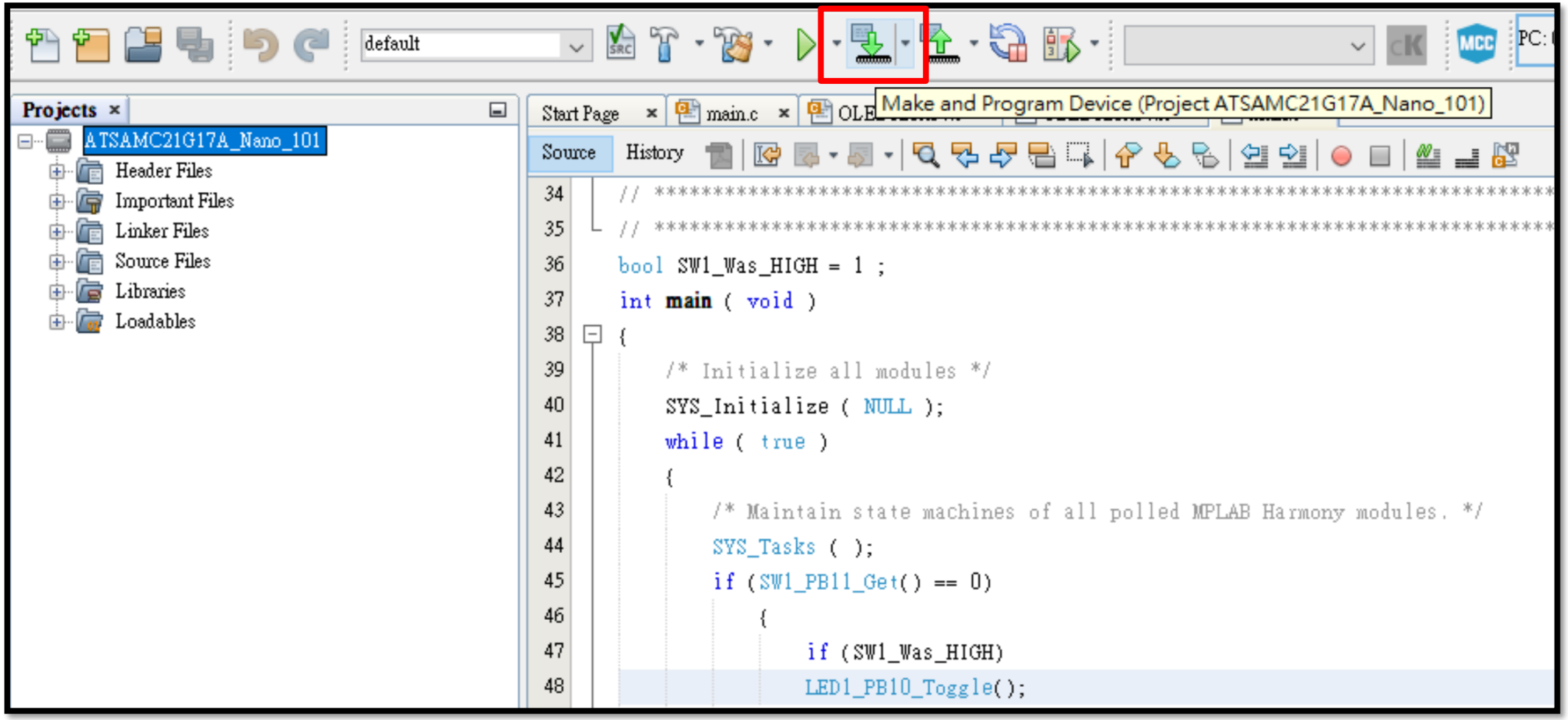
- **APP-Nano-C21-D21-Tw與SNAP or PICkit™ 4的正確連接**



MICROCHIP

練習1：基本的I/O設定：按鍵與LED操作

- 按下Make and Program Device按鈕來燒錄程式

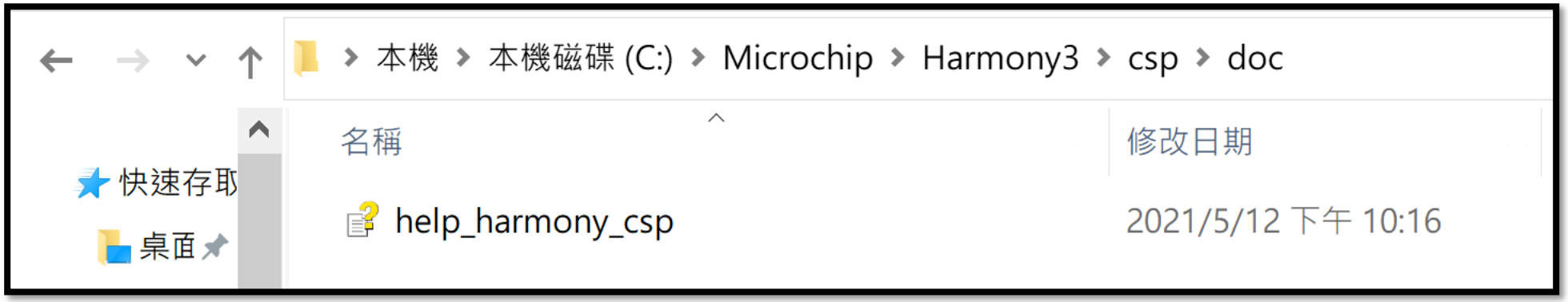


MICROCHIP

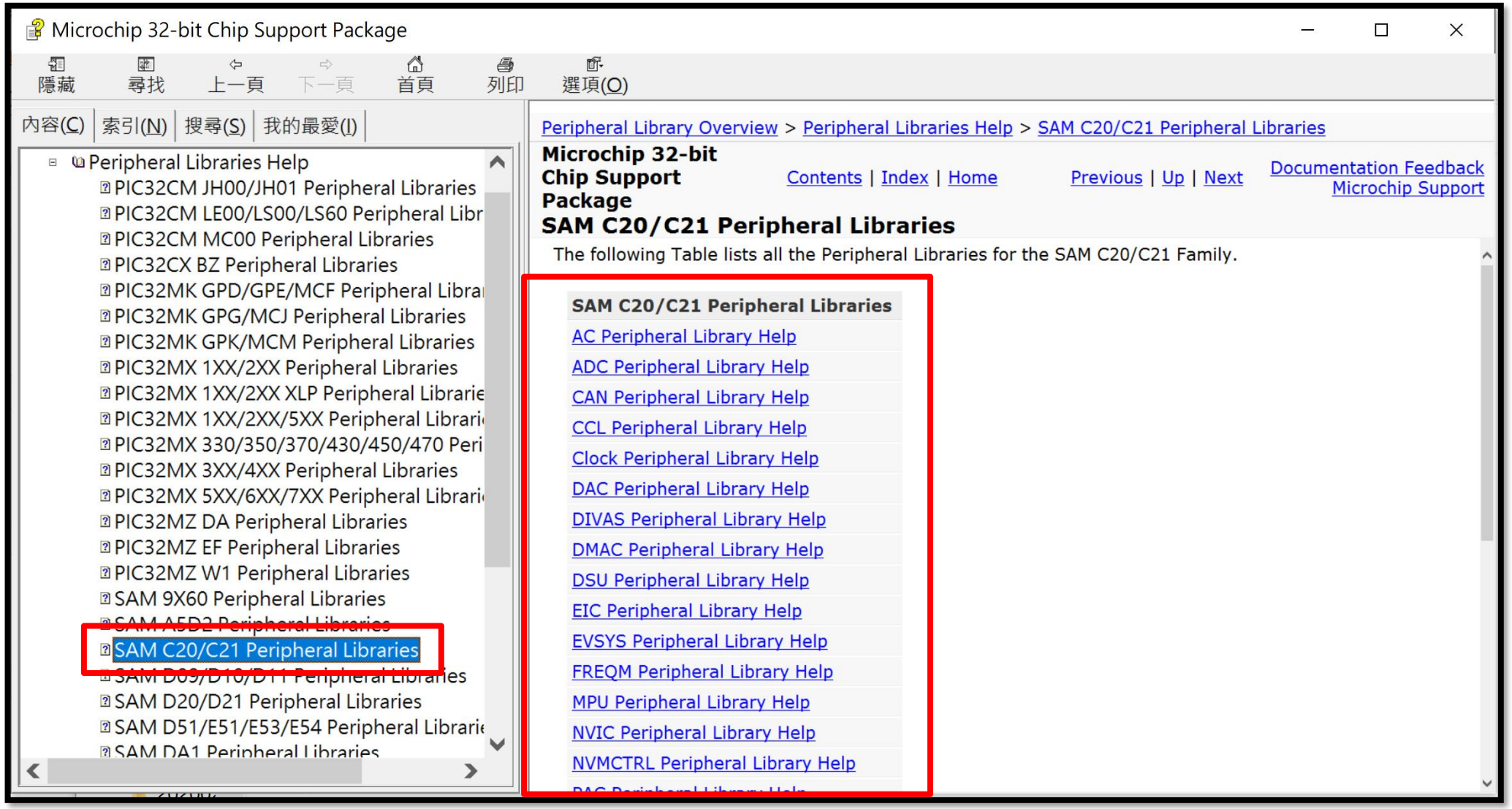
練習2：中斷實作——EIC以及TC的中斷操作法
使用PLIB & callback

要在哪裡找到PLIB的說明呢？

- 假設您的Harmony 3安裝於C:\Microchip\Harmony3
- 您可以在以下的目錄中找到
  - Help_harmony_csp這一個編譯過的HTML檔案



PLIB：每個被支援的MCU都有一個說明集合



MICROCHIP

PLIB: 檢閱 EIC Peripheral Library Help -1

EIC_CallbackRegister Function

C

```
void EIC_CallbackRegister(
    EIC_PIN pin,
    EIC_CALLBACK callback,
    uintptr_t context
);
```

Description

This function registers the callback function to be called when an interrupt condition has been sensed on the pin. A unique callback function can be reg

When an interrupt condition has been sensed on the pin, the library will call the registered callback function and will then clear the interrupt condition w

Preconditions

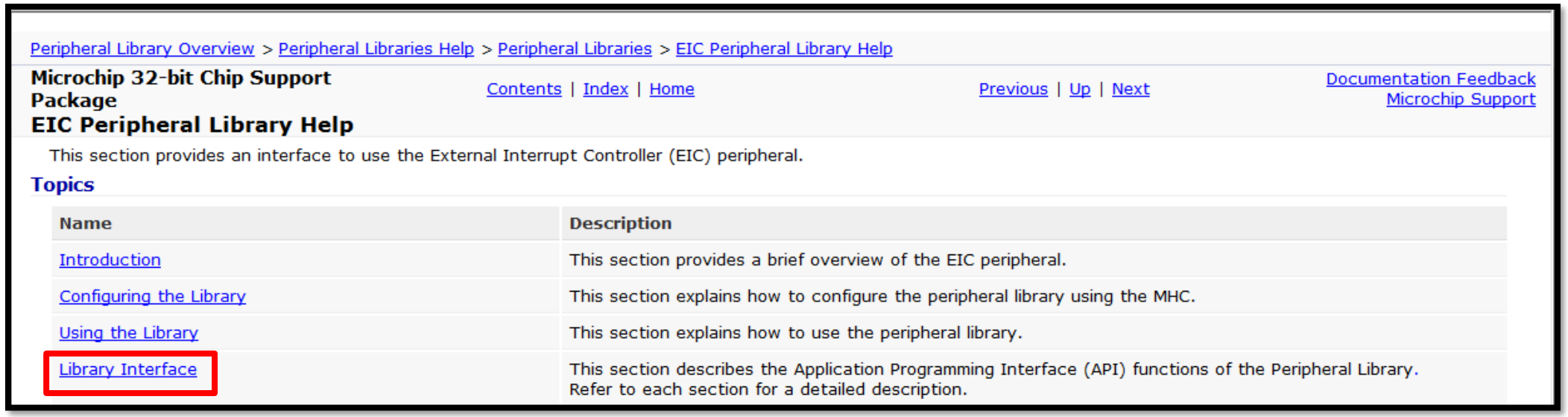
EIC_Initialize() must have been called first for the associated instance.

Parameters

Parameters	Description
pin	EIC Pin number
callback	callback function pointer. Setting this to NULL will disable the callback feature.
context	An application defined context value that will be passed to the callback function.

```
int main ( void )
{

    EIC_CallbackRegister(EIC_PIN_15,EIC15_Callback, 0);

}
```

PLIB: 檢閱 EIC Peripheral Library Help -2



Peripheral Library Overview > Peripheral Libraries Help > Peripheral Libraries > EIC Peripheral Library Help

Microchip 32-bit Chip Support Package

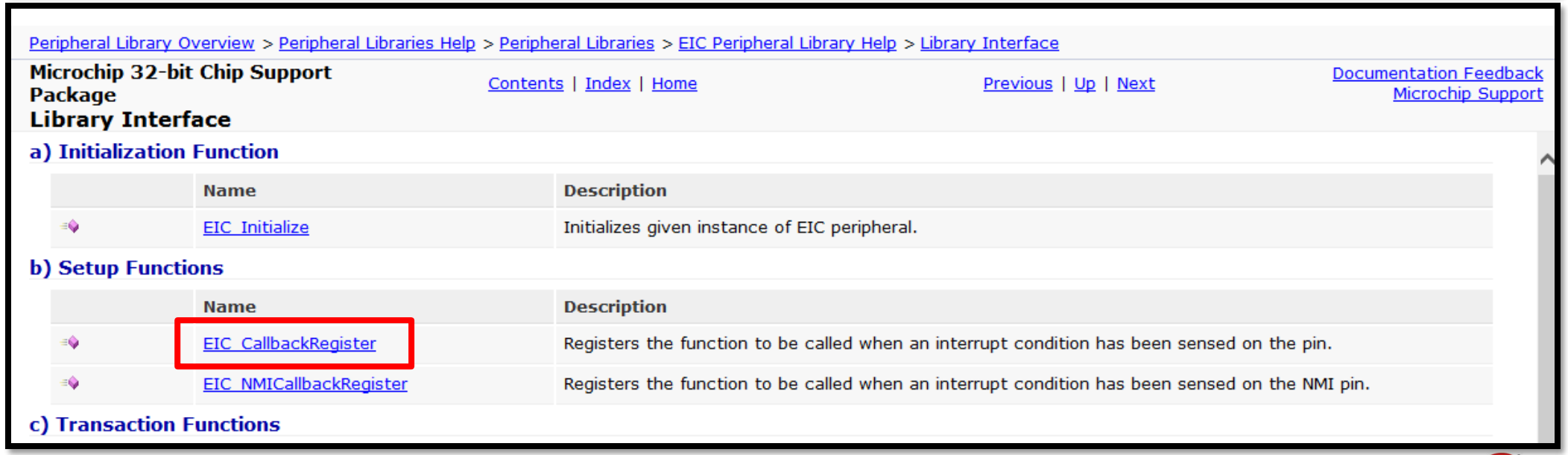
Contents | Index | Home Previous | Up | Next Documentation Feedback | Microchip Support

EIC Peripheral Library Help

This section provides an interface to use the External Interrupt Controller (EIC) peripheral.

Topics

Name	Description
Introduction	This section provides a brief overview of the EIC peripheral.
Configuring the Library	This section explains how to configure the peripheral library using the MHC.
Using the Library	This section explains how to use the peripheral library.
Library Interface	This section describes the Application Programming Interface (API) functions of the Peripheral Library. Refer to each section for a detailed description.



Peripheral Library Overview > Peripheral Libraries Help > Peripheral Libraries > EIC Peripheral Library Help > Library Interface

Microchip 32-bit Chip Support Package

Contents | Index | Home Previous | Up | Next Documentation Feedback | Microchip Support

Library Interface

a) Initialization Function

	Name	Description
	EIC_Initialize	Initializes given instance of EIC peripheral.

b) Setup Functions

	Name	Description
	EIC_CallbackRegister	Registers the function to be called when an interrupt condition has been sensed on the pin.
	EIC_NMICallbackRegister	Registers the function to be called when an interrupt condition has been sensed on the NMI pin.

c) Transaction Functions

Callbacks：一種允許了底層代碼呼叫在應用層定義的子程式之設計

何謂callback function，在google找到一篇相關的解釋

簡單的說，如果你使用了某個function，那麼你就是『call』了一個function。如果系統或是函式是要求你給一個function pointer，這個function pointer指到一個實際的函式(多半這個函式是你自己寫的)。然後它會在適當的時間

所謂的 callback function。因為這個function是被『

- **PLIB executes request from interrupt context**
- **PLIB notifies the status to application by calling the registered event handler**

TCx_TIMER_STATUS Enumeration

C

```
typedef enum {
  TC_TIMER_STATUS_NONE = 0,
  TC_TIMER_STATUS_OVERFLOW = TC_INTFLAG_OVF_Msk,
  TC_TIMER_STATUS_MATCH1 = TC_INTFLAG_MC1_Msk,
  TC_TIMER_STATUS_MSK = TC_TIMER_STATUS_OVERFLOW|TC_TIMER_STATUS_MATCH1,
  TC_TIMER_STATUS_INVALID = 0xFFFFFFFF
} TCx_TIMER_STATUS;
```

Description

Interrupt source mask for timer mode

This enumeration identifies TC timer mode interrupt source mask.

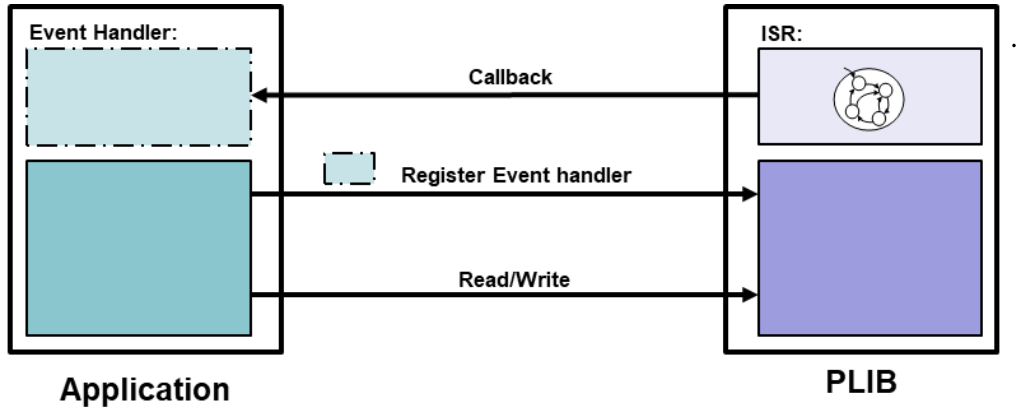
Members

Members	Description
TC_TIMER_STATUS_NONE = 0	No interrupt set
TC_TIMER_STATUS_OVERFLOW = TC_INTFLAG_OVF_Msk	overflow
TC_TIMER_STATUS_MATCH1 = TC_INTFLAG_MC1_Msk	match compare 1
TC_TIMER_STATUS_MSK = TC_TIMER_STATUS_OVERFLOW|TC_TIMER_STATUS_MATCH1	Mask of all timer interrupts
TC_TIMER_STATUS_INVALID = 0xFFFFFFFF	Force the compiler to reser

```
{
    bT
}

// ***
// ***
// Sec
// ***
// ***

int ma
{
    //
    SY

    //
    TC0_TimerCallbackRegister(TC0_CH0_TimerInterruptHandler, (uintptr_t)NULL);
```

Callbacks：一種允許了底層代碼呼叫在應用層定義的子程式之設計

- Implement an Event Handler routine in the application
- Register the Event Handler with the PLIB
- Application submits a Read/Write request to PLIB
- PLIB executes request from interrupt context
- PLIB notifies the status to application by calling the registered event handler



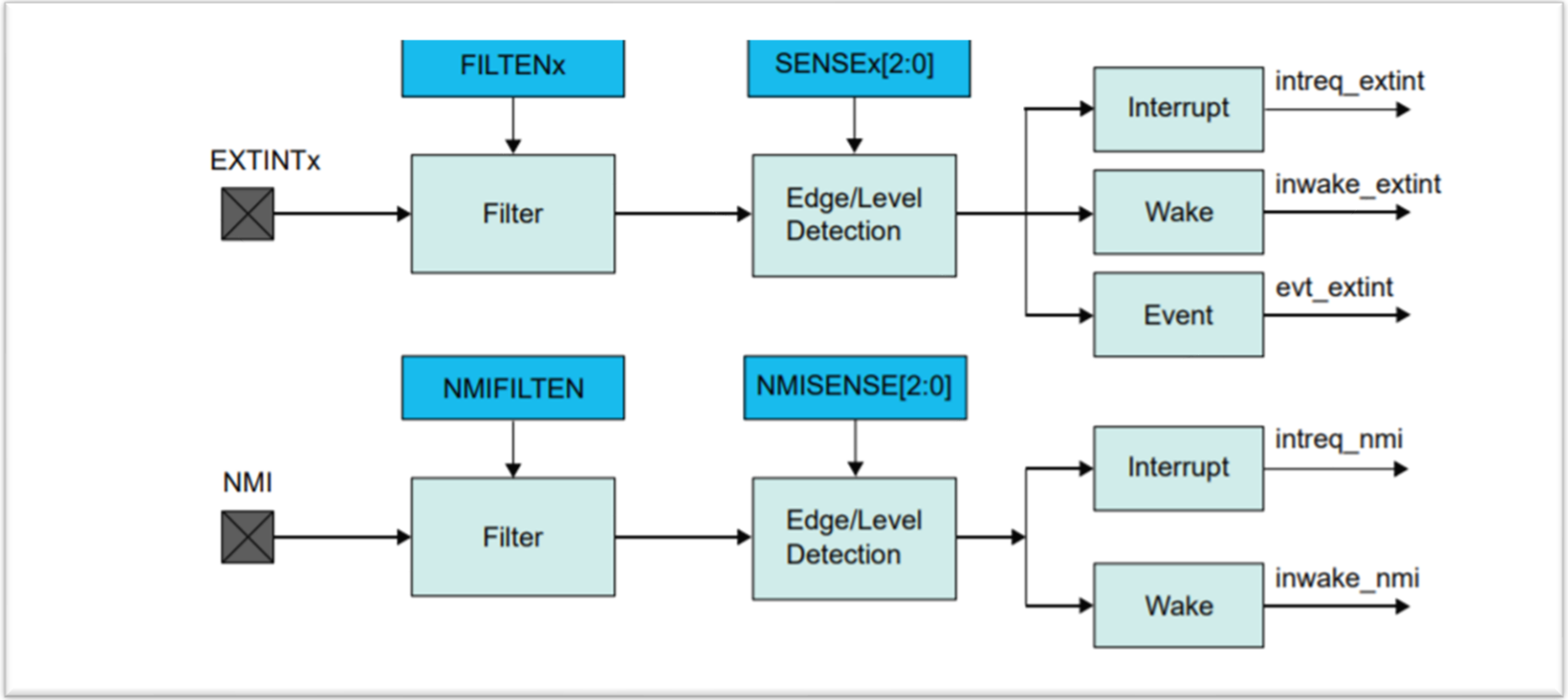

```
// This function is called after period expires
void TC0_CH0_TimerInterruptHandler(TC_TIMER_STATUS status, uintptr_t context)
{
    bToggleLED = true;
}

// *****************************************************************************
// *****************************************************************************
// Section: Main Entry Point
// *****************************************************************************
// *****************************************************************************

int main ( void )
{
    // Initialize all modules
    SYS_Initialize(NULL);

    // Register callback function for CH0 period interrupt
    TC0_TimerCallbackRegister(TC0_CH0_TimerInterruptHandler, (uintptr_t)NULL);
```

EIC: External Interrupt Controller 的方塊圖

Pin(1)			I/O Pin	Supply	A	
SAM C21E	SAM C21G	SAM C21J			EIC	REF
1	1	1	PA00	VDDANA	EXTINT[0]	
2	2	2	PA01	VDDANA	EXTINT[1]	
3	3	3	PA02	VDDANA	EXTINT[2]	
4	4	4	PA03	VDDANA	EXTINT[3]	ADC-DAC/VREFA
		5	PB04	VDDANA	EXTINT[4]	
		6	PB05	VDDANA	EXTINT[5]	
		9	PB06	VDDANA	EXTINT[6]	
		10	PB07	VDDANA	EXTINT[7]	
	7	11	PB08	VDDANA	EXTINT[8]	
	8	12	PB09	VDDANA	EXTINT[9]	
5	9	13	PA04	VDDANA	EXTINT[4]	SDADC / VREFB
6	10	14	PA05	VDDANA	EXTINT[5]	
7	11	15	PA06	VDDANA	EXTINT[6]	
8	12	16	PA07	VDDANA	EXTINT[7]	
11	13	17	PA08	VDDIO	NMI	
12	14	18	PA09	VDDIO	EXTINT[9]	
13	15	19	PA10	VDDIO	EXTINT[10]	
14	16	20	PA11	VDDIO	EXTINT[11]	
	19	23	PB10	VDDIO	EXTINT[10]	
	20	24	PB11	VDDIO	EXTINT[11]	
		25	PB12	VDDIO	EXTINT[12]	

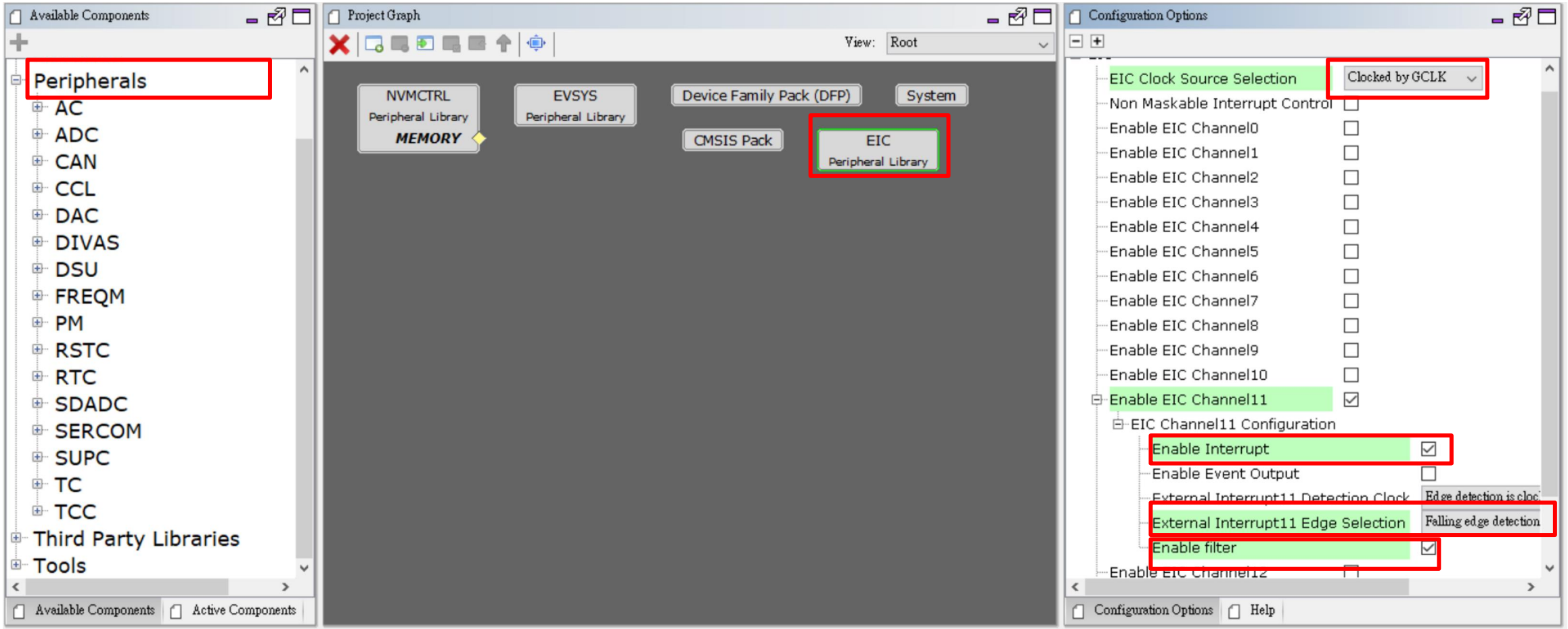
The External Interrupt Controller (EIC) allows external pins to be configured as interrupt lines. Each interrupt line can be individually masked and can generate an interrupt on rising, falling, or both edges, or on high or low levels. Each external pin has a configurable filter to remove spikes. Each external pin can also be configured to be asynchronous in order to wake up the device from sleep modes where all clocks have been disabled. External pins can also generate an event.

A separate non-maskable interrupt (NMI) is also supported. It has properties similar to the other external interrupts, but is connected to the NMI request of the CPU, enabling it to interrupt any other interrupt mode.



練習 2-1：使用EIC來簡化程式並提高可靠性

- 在Peripherals選取EIC
- 點選Project Graph中的EIC，完成必要的設定



MICROCHIP

練習2-1：使用EIC來簡化程式並提高可靠性

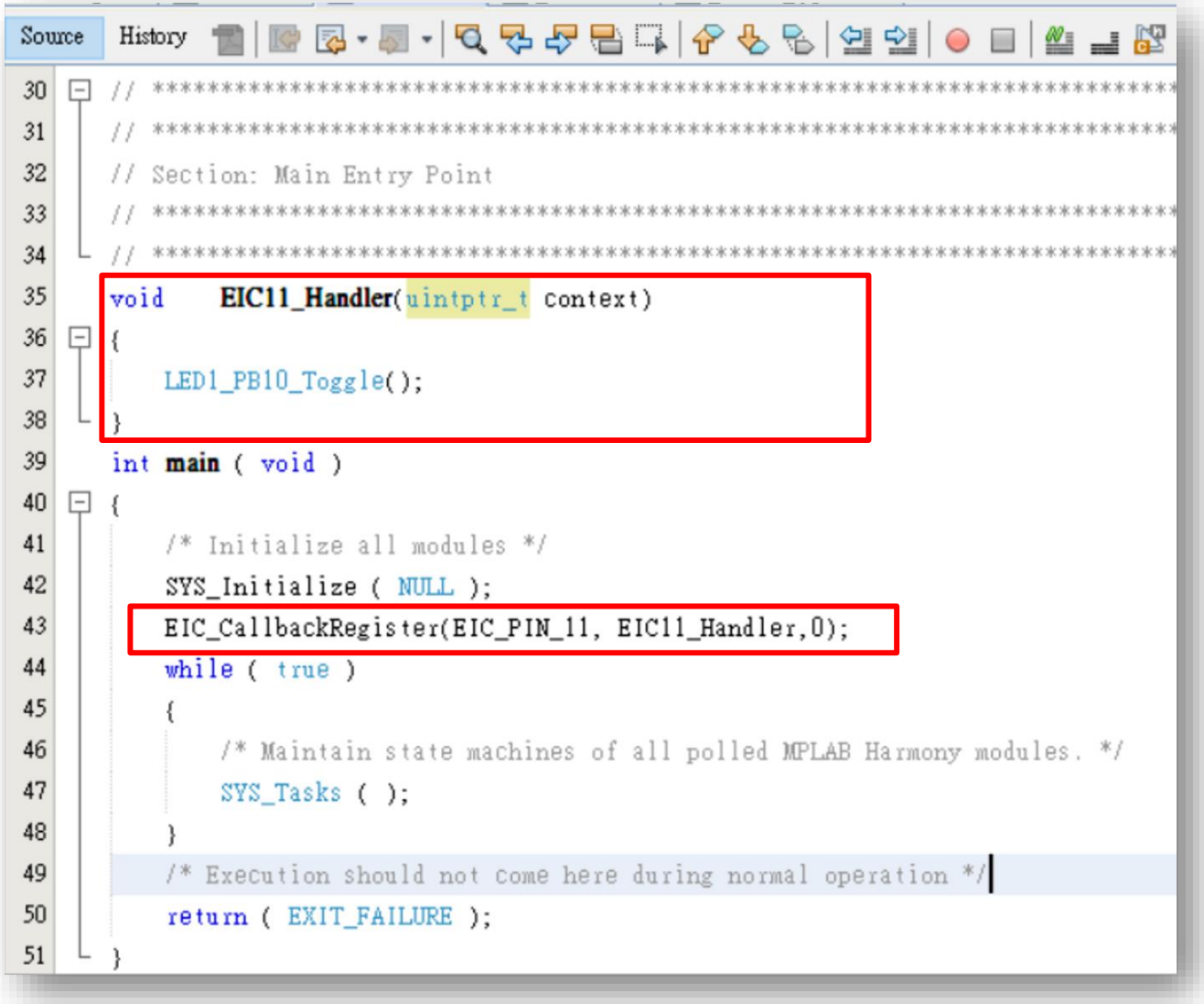
- **Pin Setting中，將PB11的Function改成EIC_EXTINT11**

Pin Settings

Order: Ports | Table View | Easy View

Pin Number	Pin ID	Custom Name	Function	Mode	Direction	Latch	Pull Up	Pull Down	Drive Strength
47	PB02		Available	Digital	High Impedance	Low	☐	☐	NORMAL
48	PB03		Available	Digital	High Impedance	Low	☐	☐	NORMAL
7	PB08		Available	Digital	High Impedance	Low	☐	☐	NORMAL
8	PB09		Available	Digital	High Impedance	Low	☐	☐	NORMAL
19	PB10	LED1_PB10	GPIO	Digital	Out	High			NORMAL
20	PB11		EIC_EXTINT11	Digital	In	n/a	☐	☐	NORMAL
37	PB22		Available	Digital	High Impedance	Low	☐	☐	NORMAL
38	PB23		Available	Digital	High Impedance	Low	☐	☐	NORMAL
5	GNDANA			Digital	High Impedance	Low			NORMAL
6	VDDANA			Digital	High Impedance	Low			NORMAL
17	VDDIO			Digital	High Impedance	Low			NORMAL
18	GNDIO			Digital	High Impedance	Low			NORMAL
35	GNDIO			Digital	High Impedance	Low			NORMAL

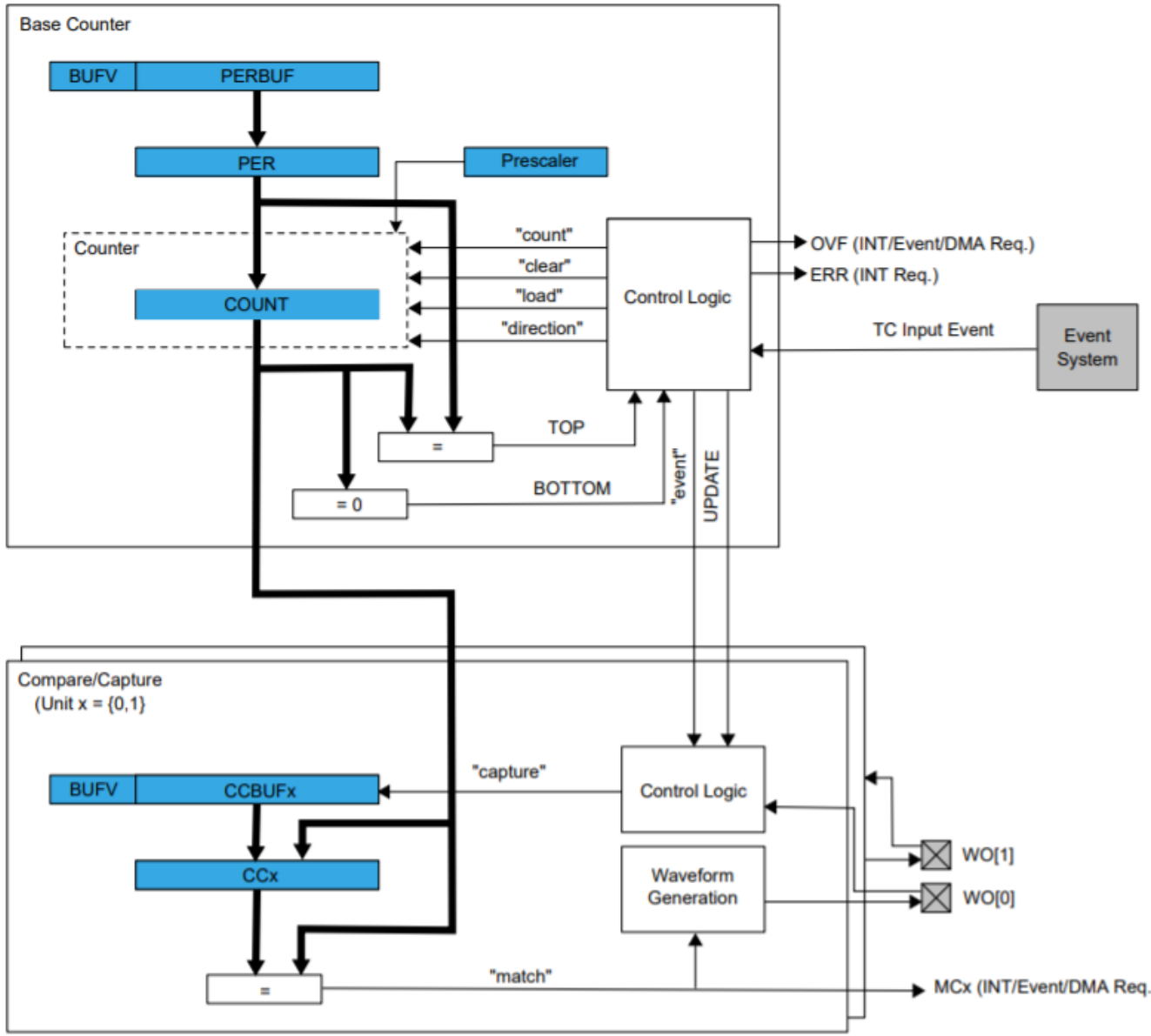
練習2-1：使用EIC來簡化程式並提高可靠性



```
30  // *****************************************************************************
31  // *****************************************************************************
32  // Section: Main Entry Point
33  // *****************************************************************************
34  // *****************************************************************************
35  void     EIC11_Handler(uintptr_t context)
36  {
37      LED1_PB10_Toggle();
38  }
39  int main ( void )
40  {
41      /* Initialize all modules */
42      SYS_Initialize ( NULL );
43      EIC_CallbackRegister(EIC_PIN_11, EIC11_Handler,0);
44      while ( true )
45      {
46          /* Maintain state machines of all polled MPLAB Harmony modules. */
47          SYS_Tasks ( );
48      }
49      /* Execution should not come here during normal operation */
50      return ( EXIT_FAILURE );
51  }
```

- EIC不但可自行偵測觸發條件，並且可以做訊號的過濾，防止彈跳發生。
- EIC_CallbackRegister()的作用為註冊EIC PIN 11事件的Call Back程式為何！
- 當EIC PIN 11的事件成立後，PLIB會呼叫指定的Call Back程式EIC11_Handler()繼續處理user要交辦的功能，之後再由PLIB做必要處置後，交回被中斷的程序繼續執行。

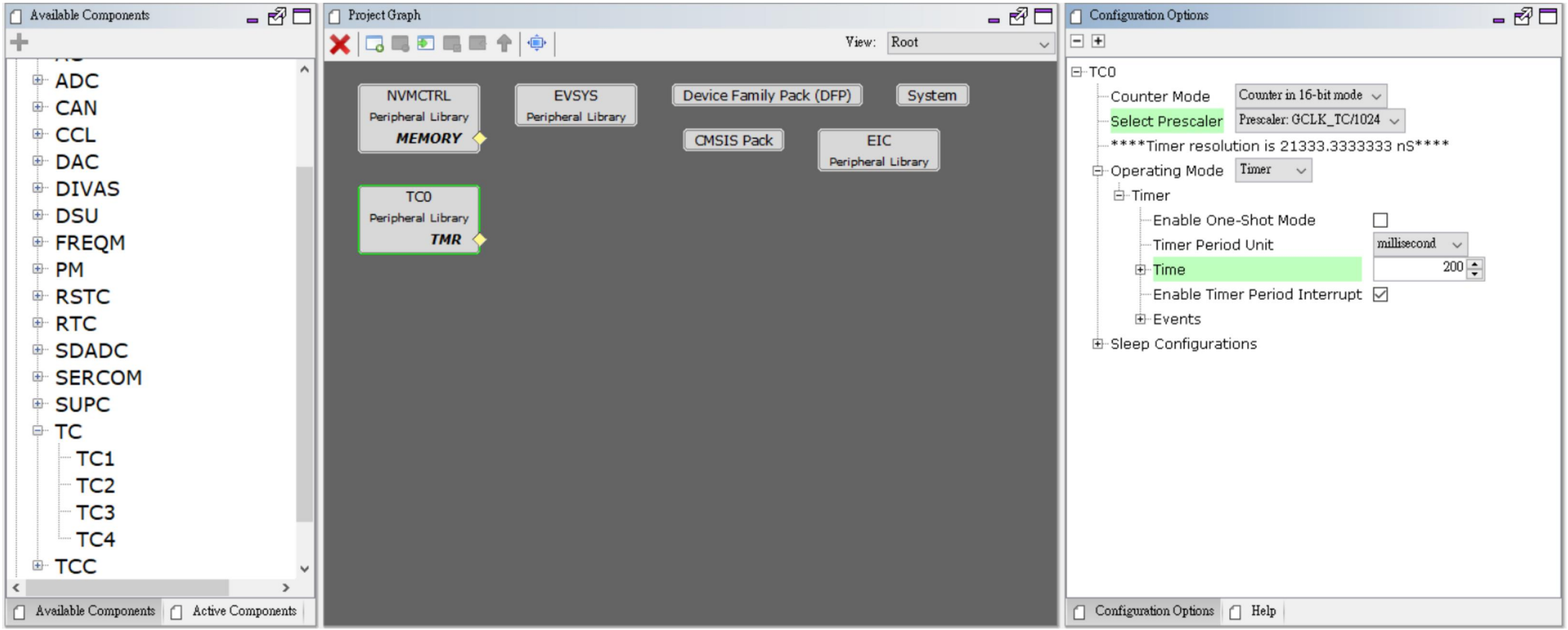
Timer/Counter (TC)



- **Each TC consists of :**
  - A counter, a prescaler, compare/capture channels and control logic.
- **The counter can be set to:**
  - count events, or clock pulses.
- **The counter, together with the compare/capture channels, can be configured to**
  - Timestamp input events or IO pin edges, allowing for capturing of frequency and/or pulse width.
- **Interrupts/output events on:**
  - Counter overflow/underflow
  - Compare match or capture

練習2-2：使用TC來提供精確的週期性時間觸發事件

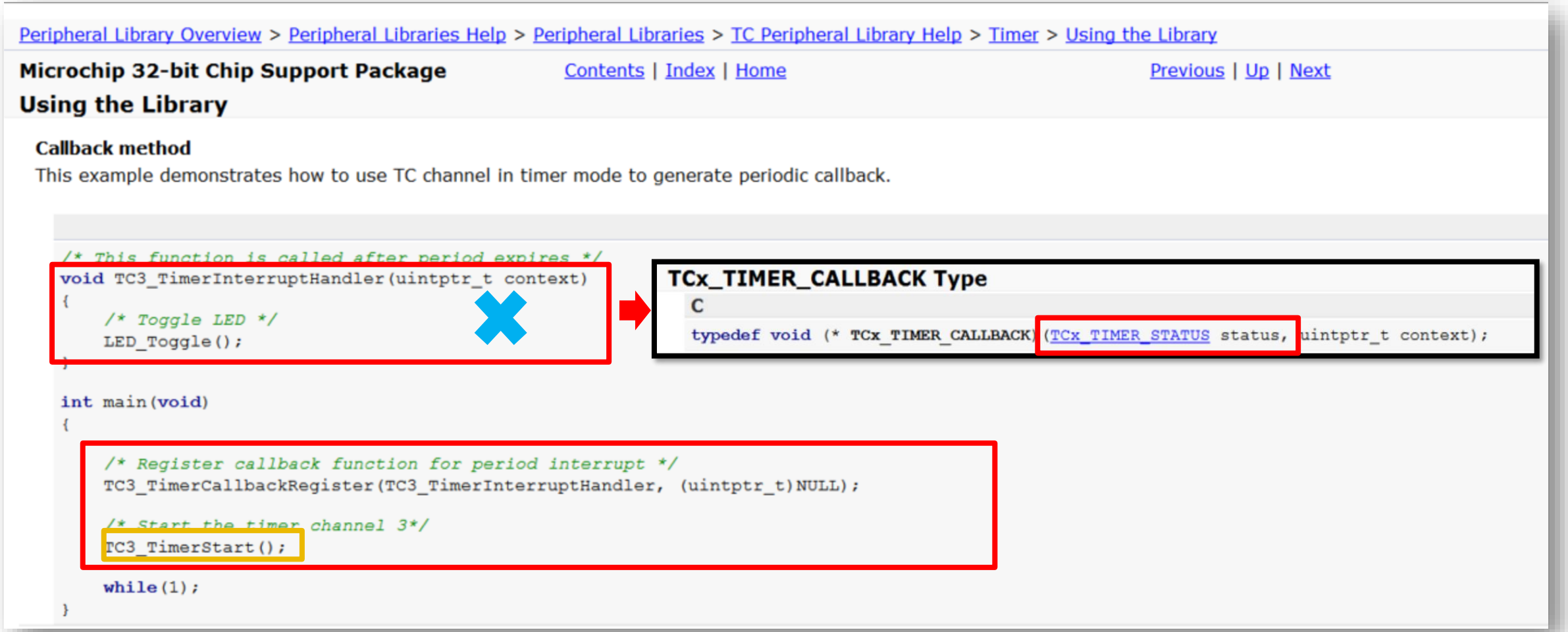
- 在Peripherals選取TC0
- 點選Project Graph中的TC0，完成必要的設定，設定時間為200 ms



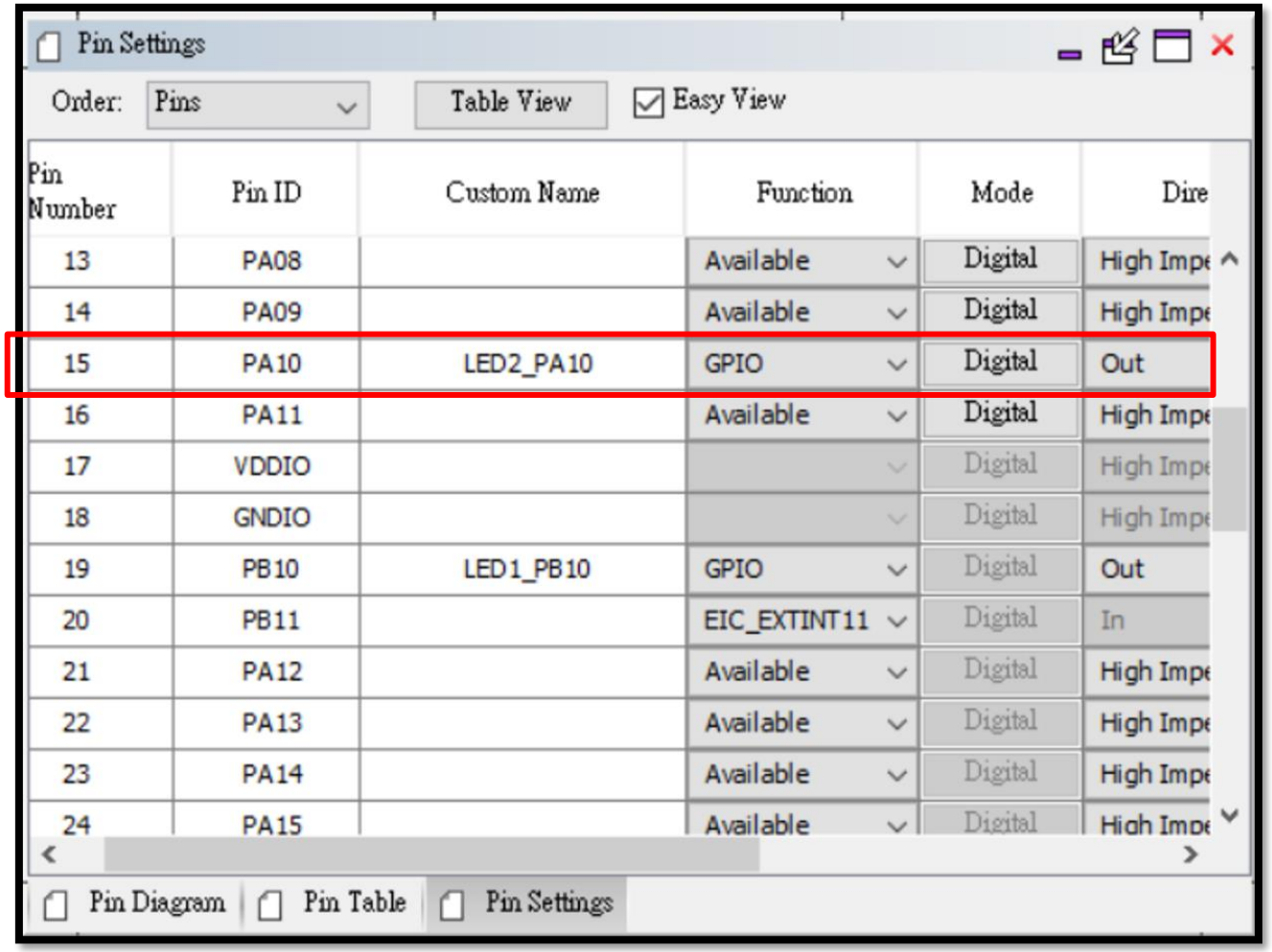
MICROCHIP

練習2-2：使用TC來提供精確的週期性時間觸發事件

- 在PLIB的說明文件中，找到針對TC周邊，以callback操作時相關資料
- TCx的PLIB在使用callback程式時會傳一個status的列舉值！



練習2-2：使用TC來提供精確的週期性時間觸發事件



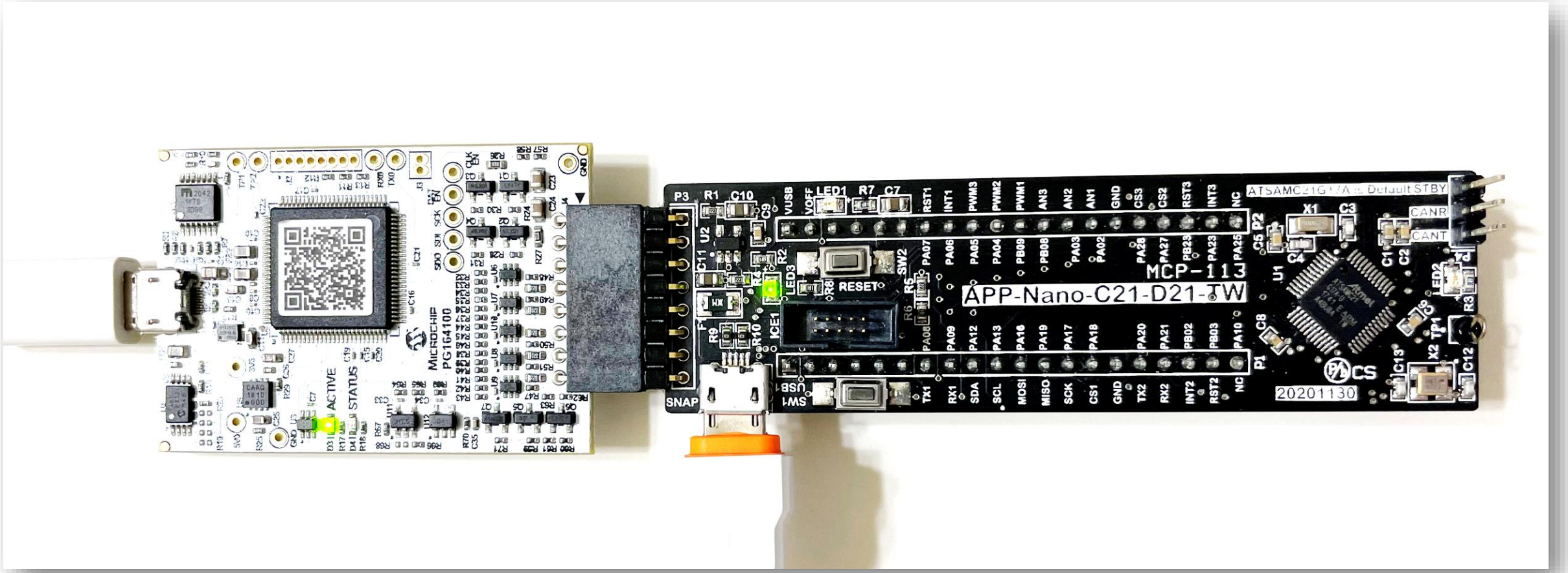
- 設定以另一個I/O腳位PA10來控制 LED2
- 練習1中利用SW1 - EIC 11（PB11）來控制LED1做Toggle的功能不變
- 加入的功能為： LED2每隔200 ms就轉態一次

練習2-2：加入TC0相關操作程式的內容

```
void    EIC11_Handler(uintptr_t context)
{
    LED1_PB10_Toggle();
}
void    TC0_EventHandler(TC_TIMER_STATUS Status,  uintptr_t context)
{
    LED2_PA10_Toggle() ;
}
int main ( void )
{
    /* Initialize all modules */
    SYS_Initialize ( NULL );
    EIC_CallbackRegister(EIC_PIN_11, EIC11_Handler,0);
    TC0_TimerCallbackRegister(TC0_EventHandler, (uintptr_t) NULL) ;
    TC0_TimerStart();
    while ( true )
    {
        /* Maintain state machines of all polled MPLAB Harmony modules. */
        SYS_Tasks ( );
    }
```

練習2-2：加入TC0相關操作程式的內容

- 在練習3之前的功能都只要單一的APP-Nano-C21-D21-TW即可

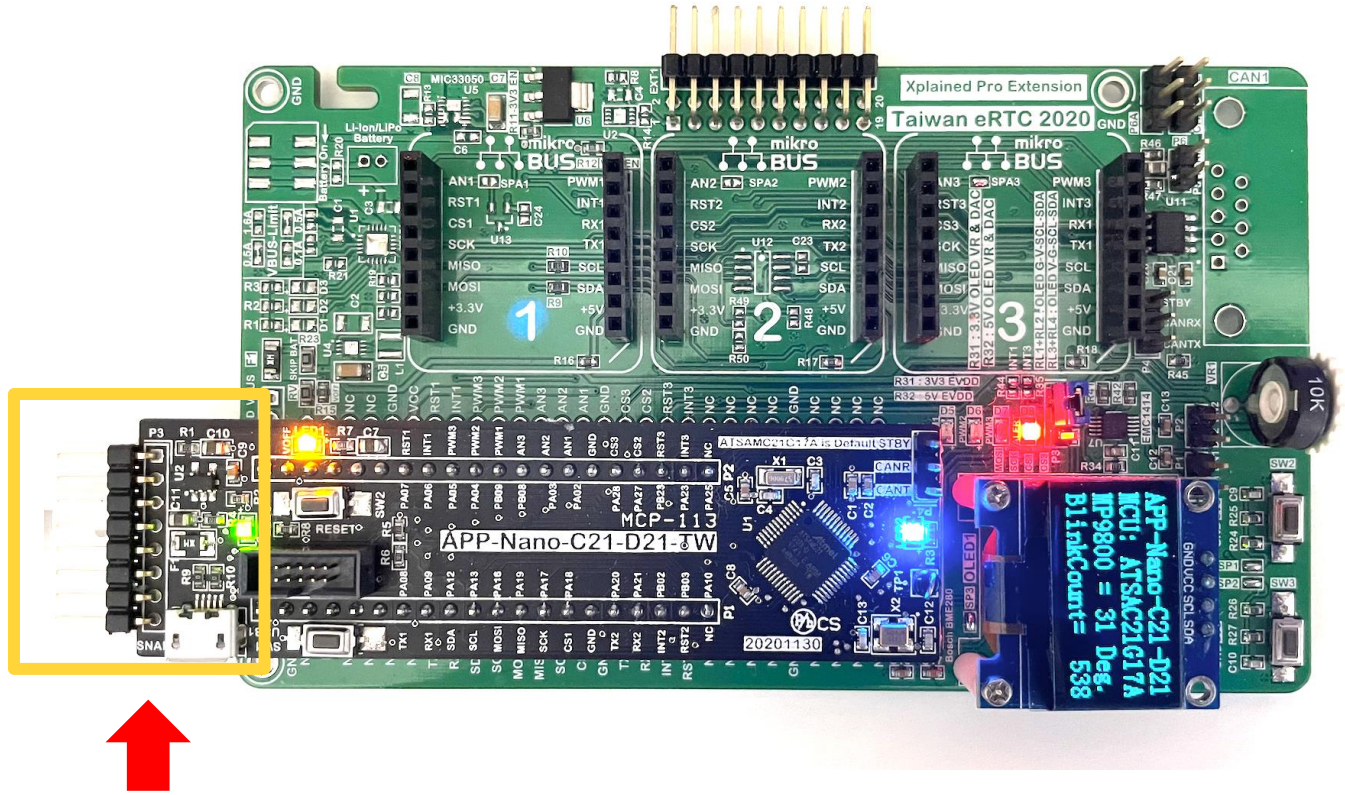


MICROCHIP

練習3：練習使用APP-Nano-BASE-TW上的資源使用OLED顯示資料

APP-Nano-C21-D21-TW on APP-Nano-BASE-TW
回顧APP-Nano-BASE-TW上的I²C Device位址

- **I²C OLED Display**
  - 128 * 64 Resolution
  - SH1306 controller organic / polymer light emitting diode
  - I²C Address : 0111100 = **0x3c**
- **I²C Temp Censor – MCP9800A5T-M/OT**
  - I²C Address : 1001101 = **0x4d**
- **I²C Temp Censor –** EMC1414-3-AIZL-TR
  - I²C Address : 0011000 = 0x18
- **Bosch Sensor BME280**
  - I²C Address : 1110110 = 0x76
  - Humidity sensor
  - Barometric pressure
  - Ambient Temperature



請由Micro-USB連接電腦並進行供電，並利用P3連接SNAP or PICkit™ 4

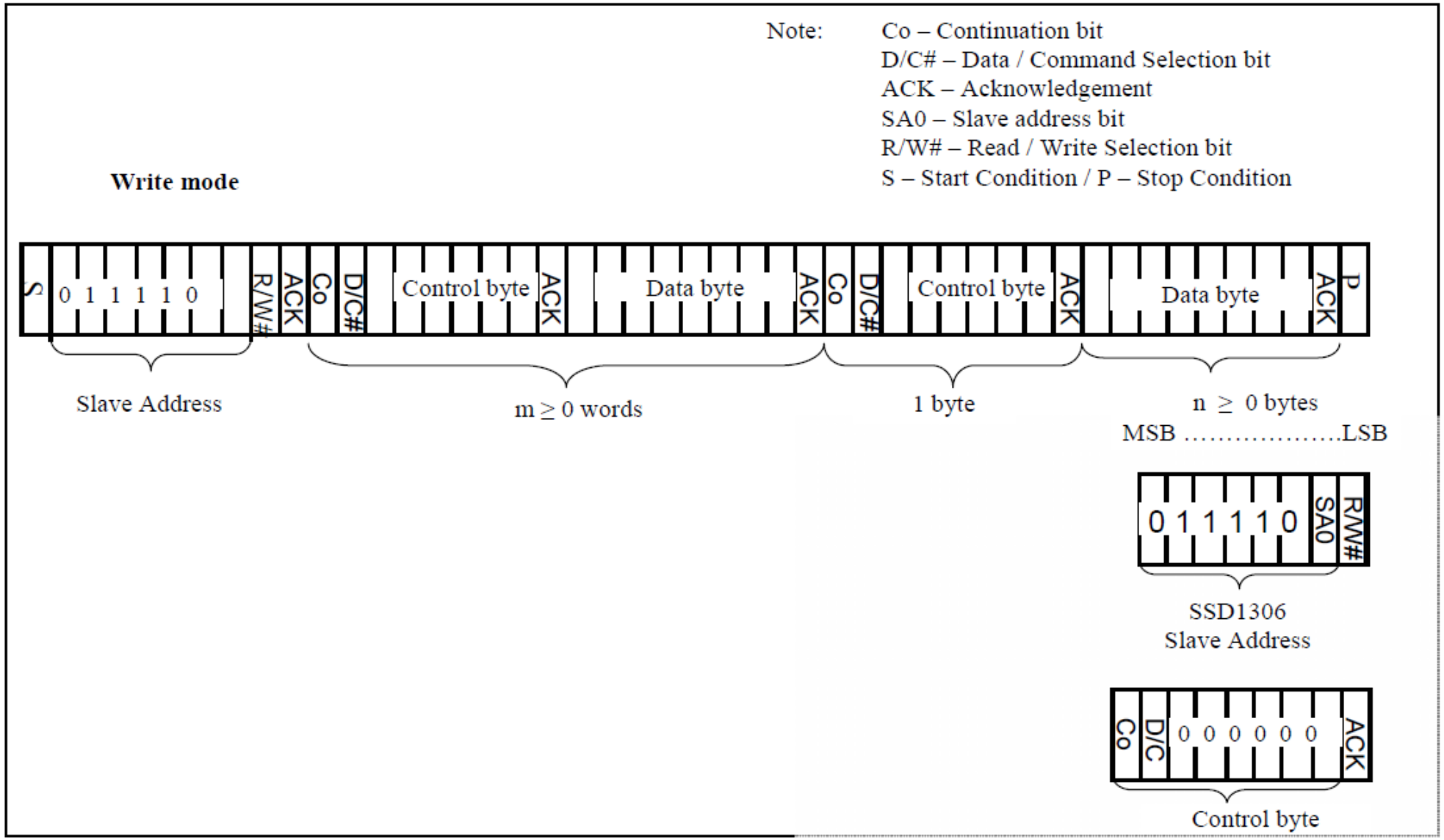
MICROCHIP

OLED Display簡介

- 使用驅動IC： SSD1306
- 0.96吋，解析度128 * 64
- 資料介面I²C
- VCC電源（3~5.5V）
- I²C位址

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0	1	1	1	1	0	SA0	R/W#

SSD1306 I²C Mode的資料格式

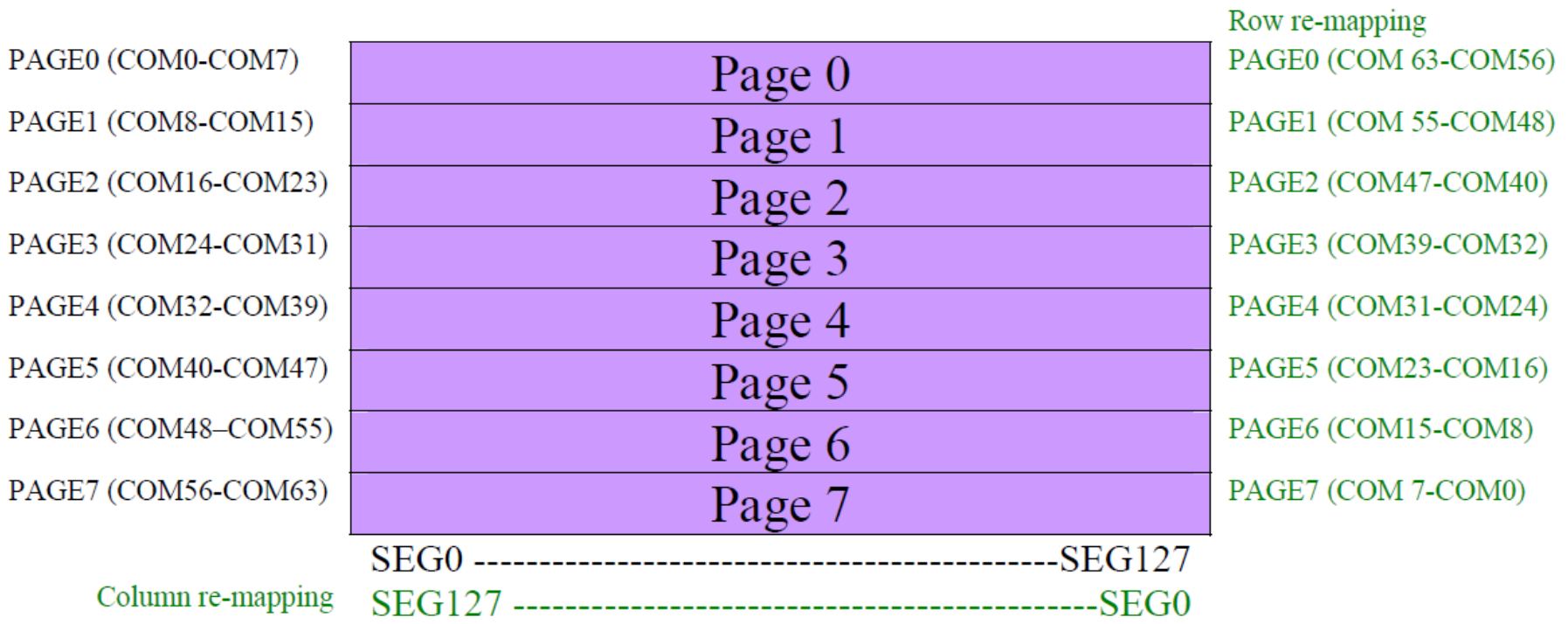


MICROCHIP

SSD1306内部的Display RAM

The GDDRAM is a bit mapped static RAM holding the bit pattern to be displayed. The size of the RAM is 128 x 64 bits and the RAM is divided into eight pages, from PAGE0 to PAGE7, which are used for monochrome 128x64 dot matrix display, as shown in Figure 8-13.

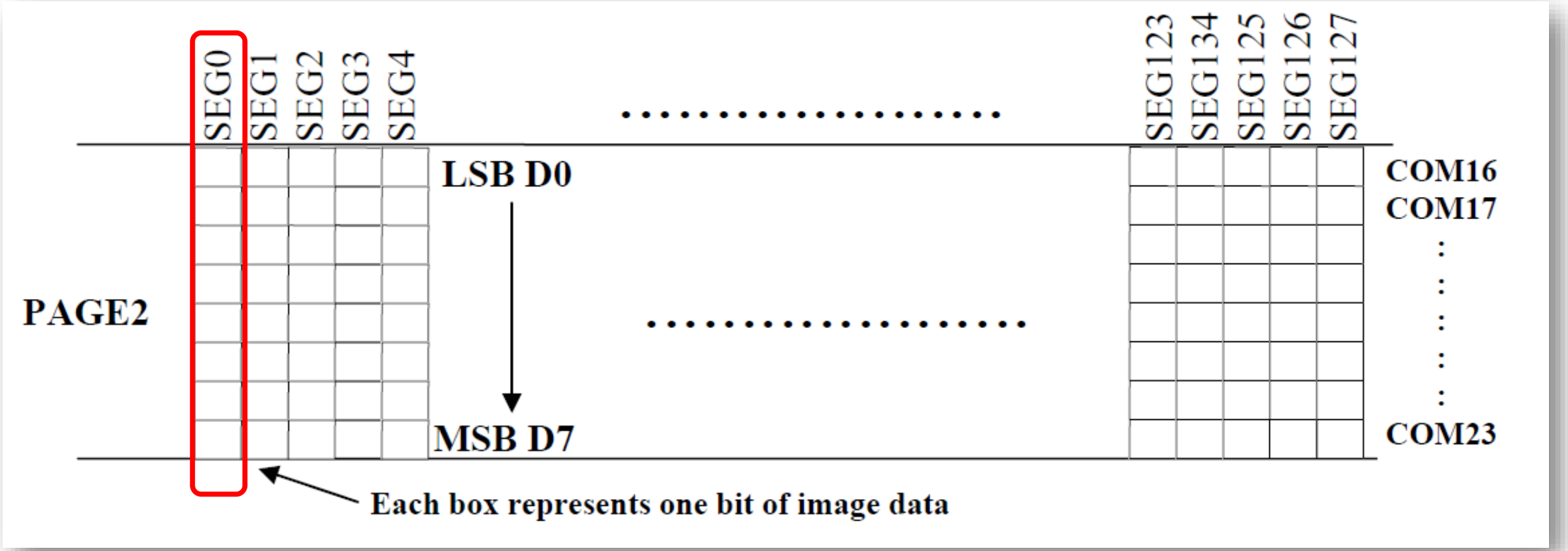
Figure 8-13 : GDDRAM pages structure of SSD1306



SSD1306 GDDRAM的放大圖

每一個Page有128個Bytes的資料，控制128*8個點

總共有8個Pages = 128 * 64點！



SSD1306的Library OLED128x64.h & OLED128x64.c

OLED128x64.h (1)

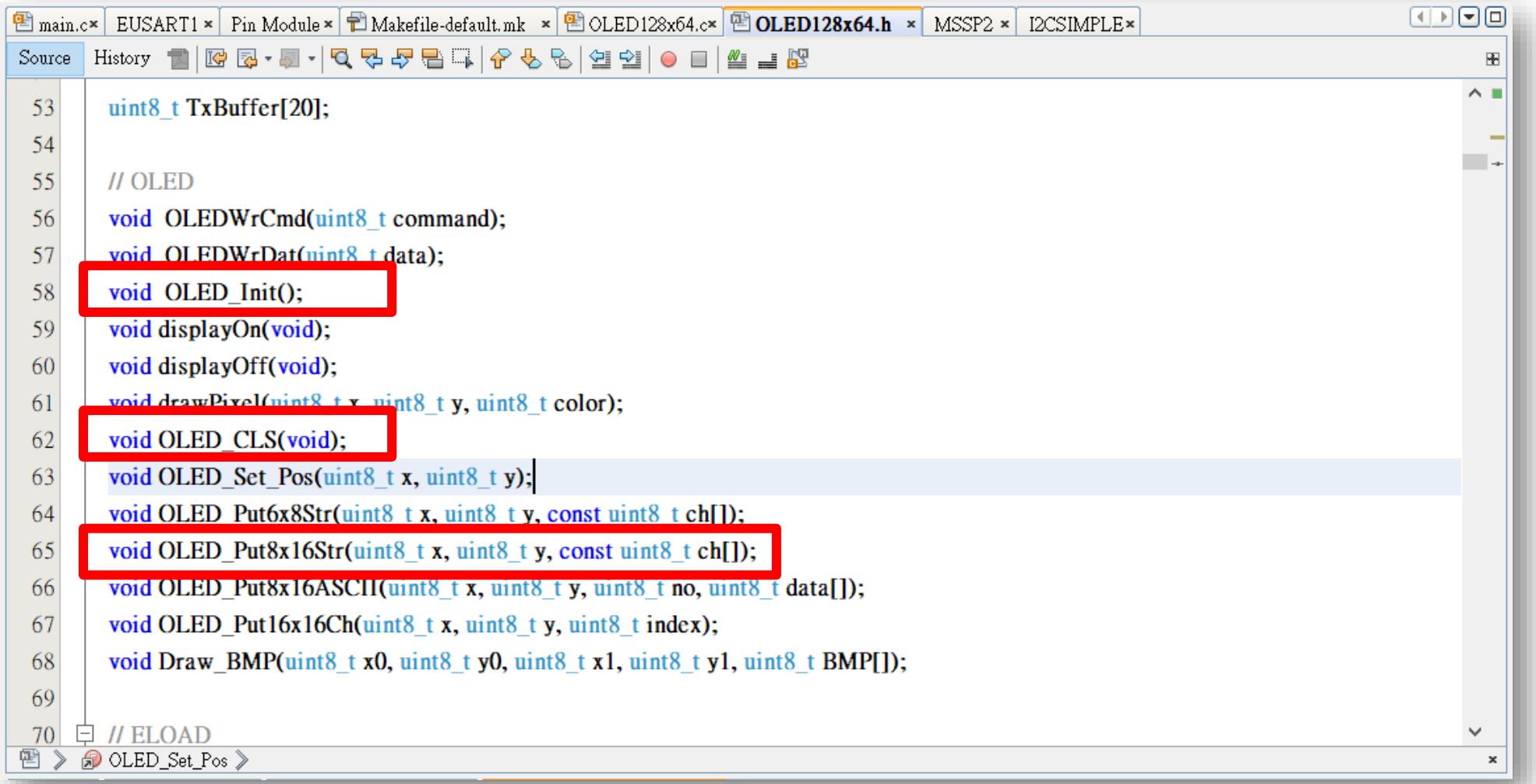
main.c × OLED128x64.h ×

Source History

```
39
40  #include "config/APP_Nano_C21_D21/definitions.h"
41
42  #define OLED_ADDRESS     0x3c          //3C -->  I2C address for OLED  0111100
43  #define X_WIDTH          128
44  #define Y_WIDTH          64
45  #define OLED_Command_Mode        0x80  // SSD1360
46  #define OLED_Data_Mode           0x40  // SSD1306
47  // #define _BV(x) (1 << x)
48  #define BLACK   0
49  #define WHITE   1
50  #define swap(a, b) { uint8_t t = a; a = b; b = t; }
```

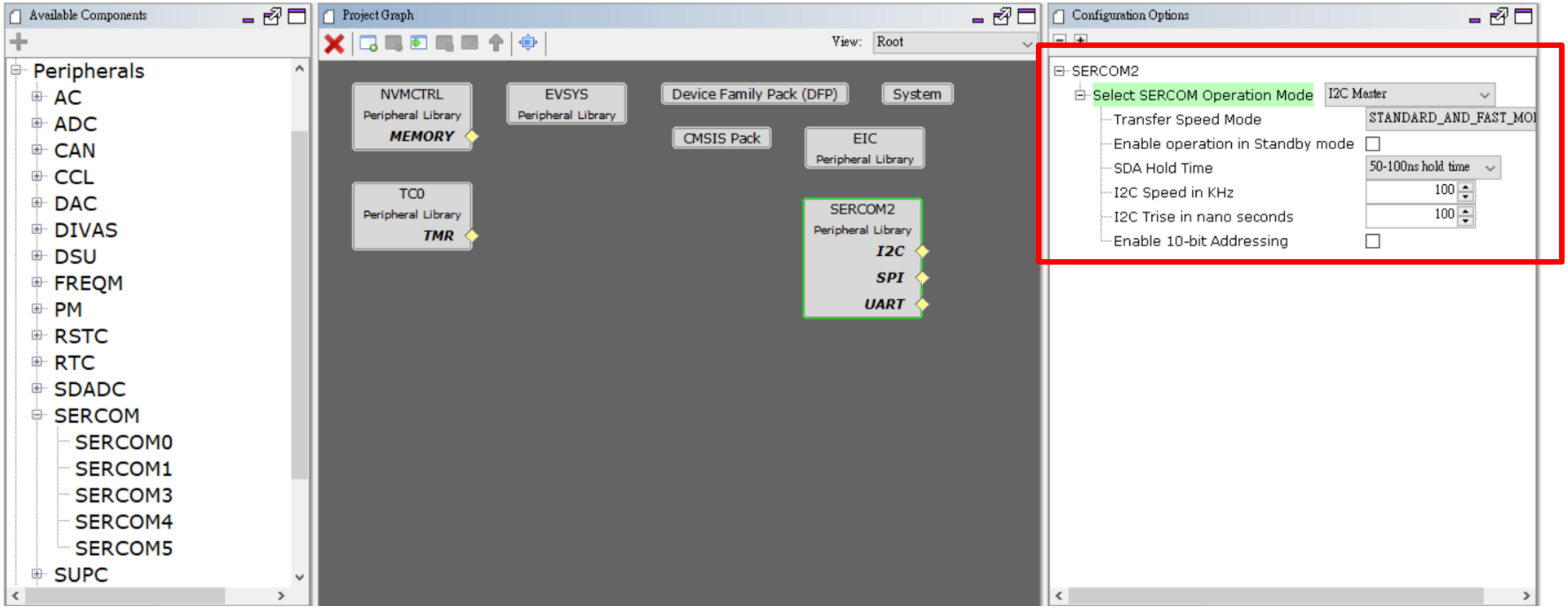
SSD1306的Library OLED128x64.h & OLED128x64.c

OLED128x64.h (2)



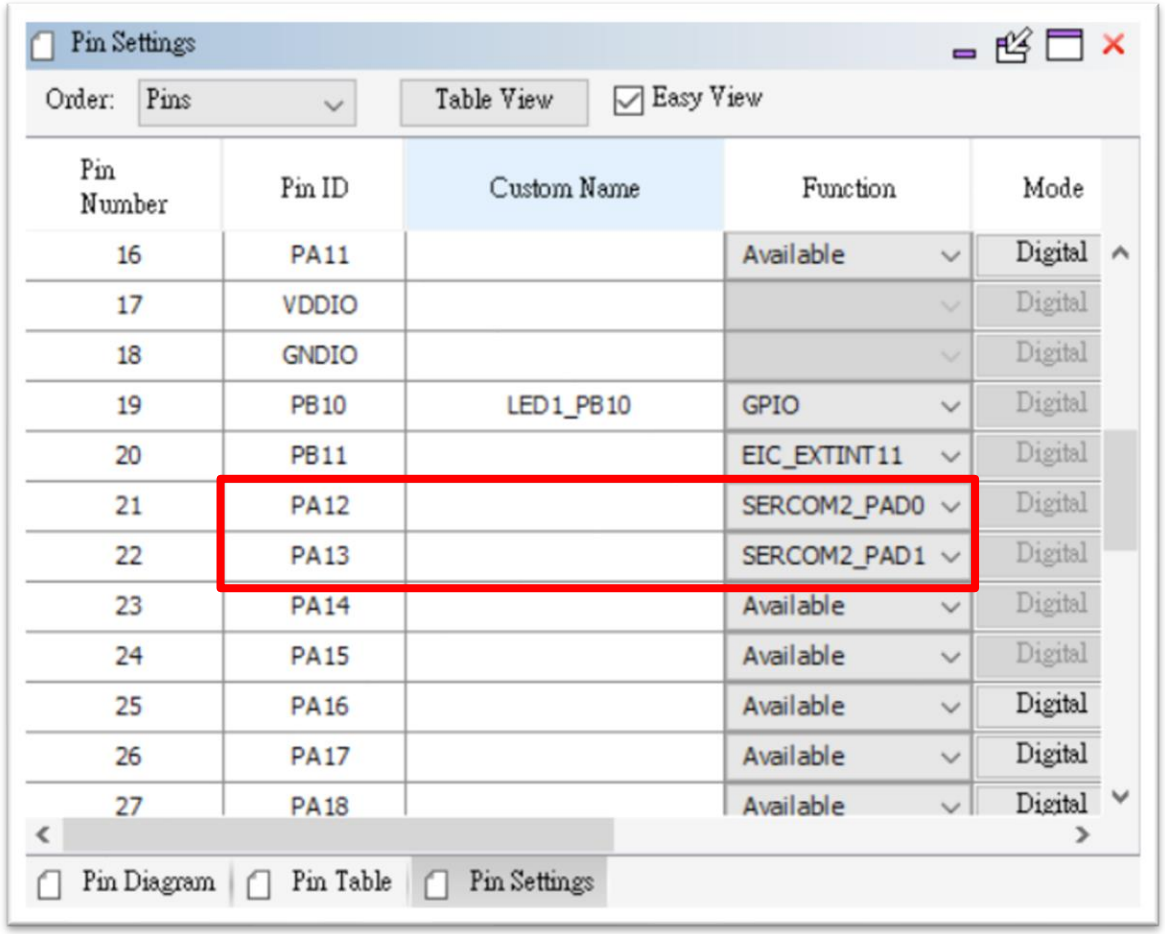
練習3：練習使用APP-Nano-BASE-TW上的資源 – OLED

- 在Peripherals選取SERCOM2
- 點選Project Graph中的SERCOM2，將其設定為I²C Master， 100 kHz



練習3：練習使用APP-Nano-BASE-TW上的資源 – OLED

- 在Pin Setting中將PA12 & PA13設為SERCOM2的SDA & SDL
- 請記得將有關OLED的3個相關檔案copy到你的專案中之src目錄



MICROCHIP

練習3：參考PLIB說明中對I2C callback的定義

Using the Library

The example code demonstrates write operation using callback method.

```
#define APP_SLAVE_ADDR 0x0057
#define NUM_BYTES      10

uint8_t myWriteData [NUM_BYTES] = {'1', '0', ' ', 'B', 'Y', 'T', 'E', 'S', '!', '!',};

void SERCOM0_I2C_Callback(uintptr_t context)
{
    if(SERCOM0_I2C_ErrorGet() == SERCOM_I2C_ERROR_NONE)
    {
        //Transfer is completed successfully
    }
    else
    {
        //Error occurred during transfer.
    }
}

int main(void)
{
    /* Register Callback function */
    SERCOM0_I2C_CallbackRegister(SERCOM0_I2C_Callback, (uintptr_t)NULL);

    /* Submit Write Request */
    SERCOM0_I2C_Write(APP_SLAVE_ADDR, &myWriteData[0], NUM_BYTES);
}
```

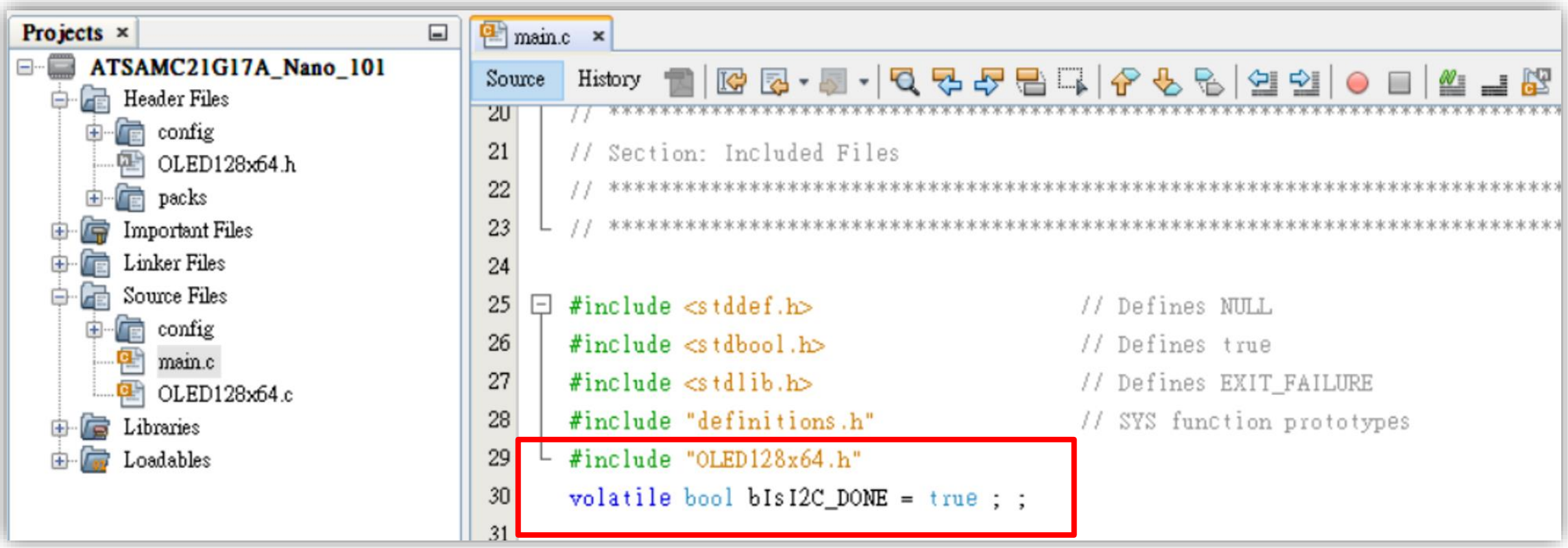
SERCOM_I2C_CALLBACK Type

C

```
typedef void (* SERCOM_I2C_CALLBACK)(uintptr_t contextHandle);
```

練習3：練習使用APP-Nano-BASE-TW上的資源 – OLED

- 記得將OLED128x64.c加入專案，並將OLED128x64.h加入main.c
- I²C功能的完成旗號請設定為bIsI2C_DONE ，在OLED128x64.c中也會參考此旗號



MICROCHIP

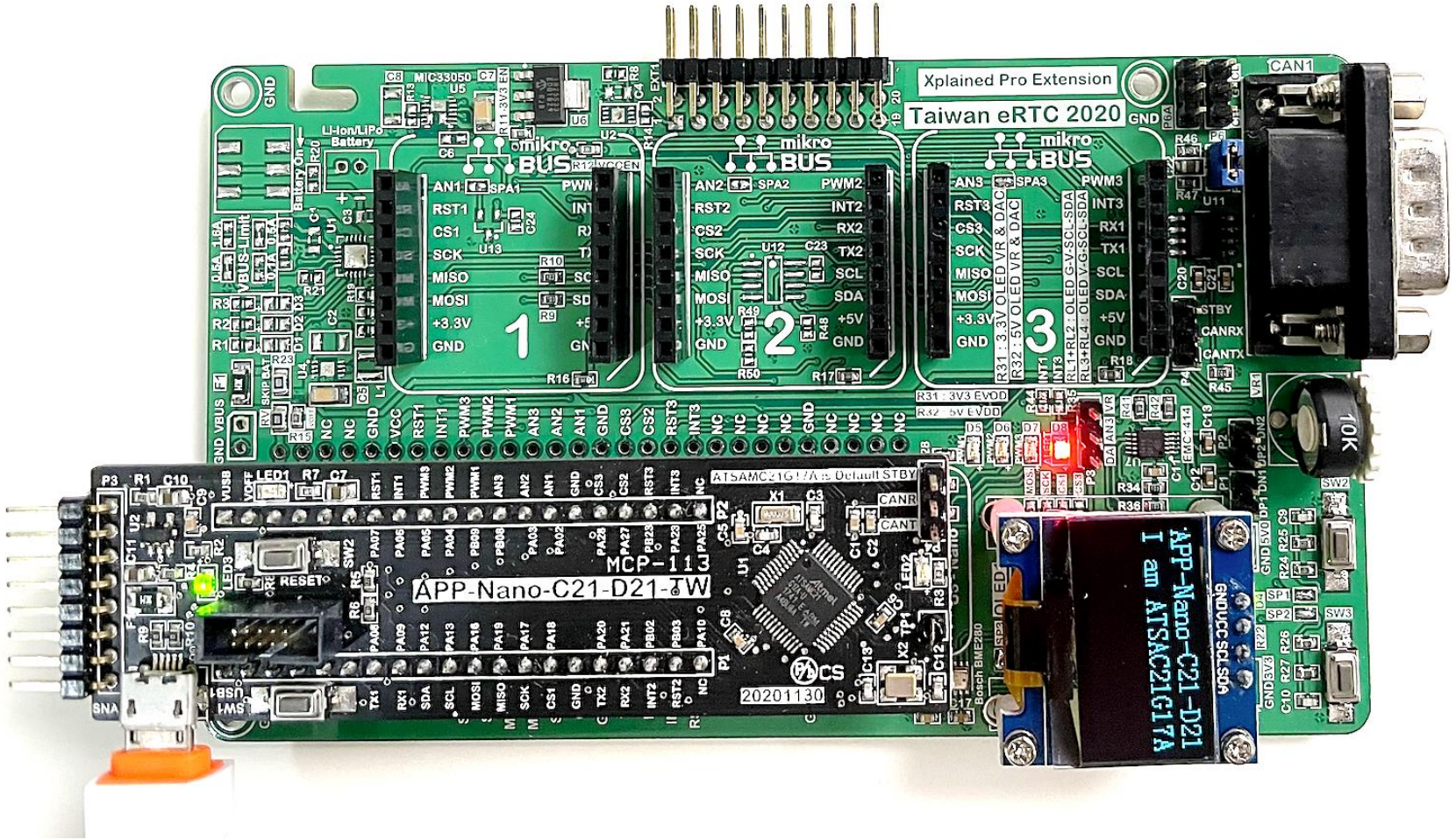
練習3：練習使用APP-Nano-BASE-TW上的資源 – OLED

- 鍵入SERCOM2的callback程式、註冊程式、及OLED程式

```
void I2C_EventHandler(uintptr_t context)
{
    if (SERCOM2_I2C_ErrorGet() == SERCOM_I2C_ERROR_NONE)
    {
        bIsI2C_DONE = true;
    }
}
int main ( void )
{
    /* Initialize all modules */
    SYS_Initialize ( NULL );
    EIC_CallbackRegister(EIC_PIN_11, EIC11_Handler,0);
    TC0_TimerCallbackRegister(TC0_EventHandler, (uintptr_t) NULL) ;
    SERCOM2_I2C_CallbackRegister(I2C_EventHandler, 0 ) ;
    TC0_TimerStart();
        OLED_Init();
        OLED_CLS();
        OLED_Put8x16Str(0, 0, (const unsigned char*) "APP-Nano-C21-D21") ;
        OLED_Put8x16Str(0, 2, (const unsigned char*) "I am ATSAC21G17A") ;

    while ( true )
    {
```

練習3的執行結果 – OLED, LED1, LED2, SW1可正常工作

- 雖然燒錄程式後可正常工作，但是USB電源插拔後卻不動!
- 可能原因：**OLED module的Reset時間太長 …. 解Bug …**



MICROCHIP

練習3：練習使用APP-Nano-BASE-TW上的資源 – OLED

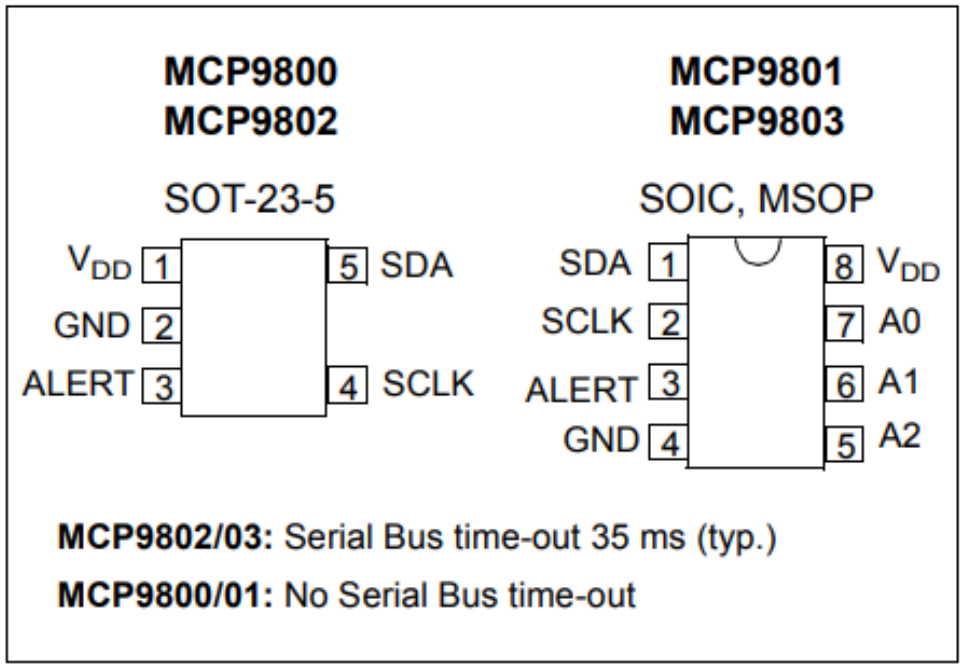
- 解決OLED啟動較晚的問題 – 方法很多，使用T0先delay 200 ms最快！

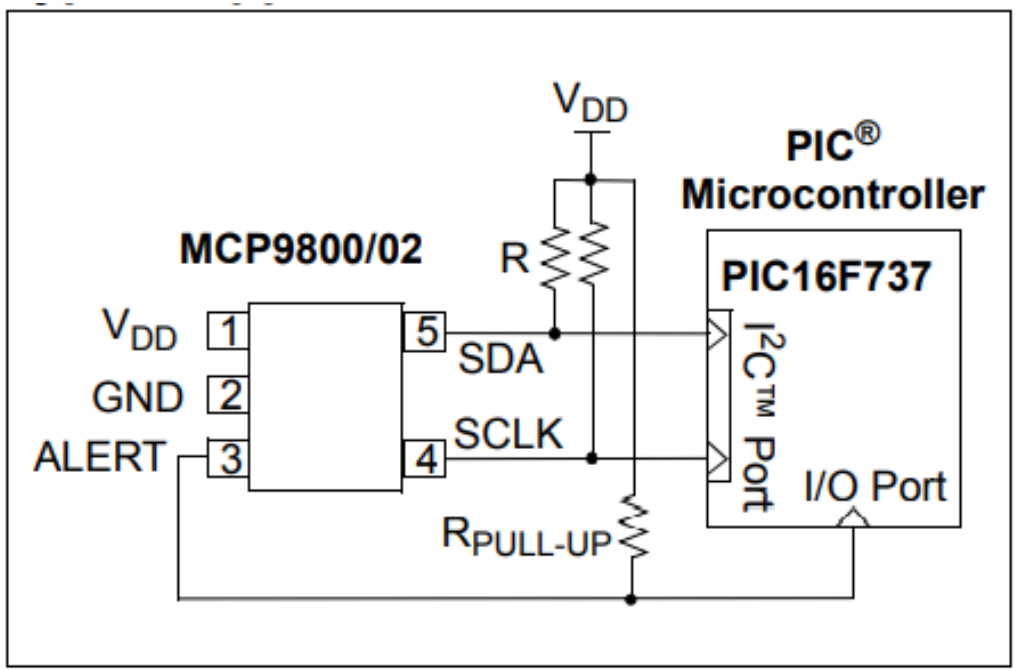
```
void     TC0_EventHandler(TC_TIMER_STATUS Status,  uintptr_t context)
{
    LED2_PA10_Toggle() ;
    bIsTC0_DONE = true ;
}
int main ( void )
{
    /* Initialize all modules */
    SYS_Initialize ( NULL );
    EIC_CallbackRegister(EIC_PIN_11, EIC11_Handler,0);
    TC0_TimerCallbackRegister(TC0_EventHandler, (uintptr_t) NULL) ;
    SERCOM2_I2C_CallbackRegister(I2C_EventHandler, 0 ) ;
    TC0_TimerStart();

    bIsTC0_DONE = false ;
    while(bIsTC0_DONE == false) {}

        OLED_Init();
        OLED_CLS();
```

練習4：加入MCP9800溫度Sensor的讀取並顯示於OLED第三行

MCP9800溫度感應IC簡介

- MCP9800為一個具備I²C介面並用其傳送資料的溫度Sensor
  - http://ww1.microchip.com/downloads/en/DeviceDoc/21909d.pdf
- MCP9800系列也提供不同包裝、Bus time-out 、位址選項



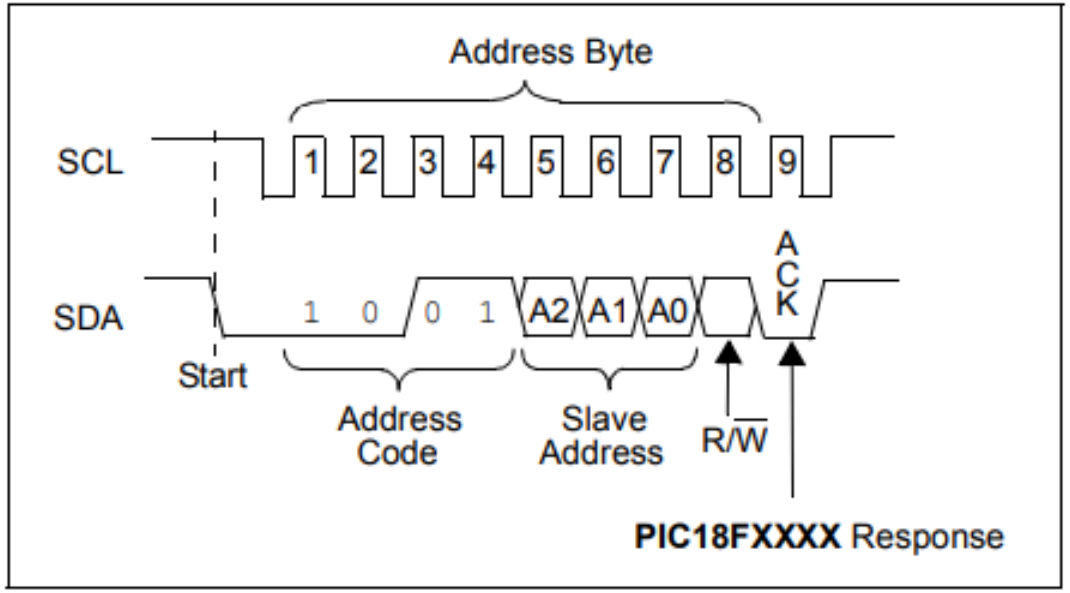


MICROCHIP

MCP9800系列溫度IC的定址

- **I²C Bus以7 or 10 bits的位址來區分並聯在一起的裝置**
  - MCP9800系列
- **MCP9800/MCP9802使用固定位址，有多種不同出廠位址可選購**
- **MCP9801 & MCP9803則因為有多餘腳位，提供A0~A2的位址接腳，故可以用這些接腳來自訂位址**

Device	A6	A5	A4	A3	A2	A1	A0
MCP9800/02A0	1	0	0	1	0	0	0
MCP9800/02A1	1	0	0	1	0	0	1
MCP9800/02A2	1	0	0	1	0	1	0
MCP9800/02A3	1	0	0	1	0	1	1
MCP9800/02A4	1	0	0	1	1	0	0
MCP9800/02A5	1	0	0	1	1	0	1
MCP9800/02A6	1	0	0	1	1	1	0
MCP9800/02A7	1	0	0	1	1	1	1
MCP9801/03	1	0	0	1	X	X	X



MICROCHIP

MCP9800的內部方塊圖與暫存器指標

使用P0、P1位元來指定要讀取或寫入的MCP9800暫存器

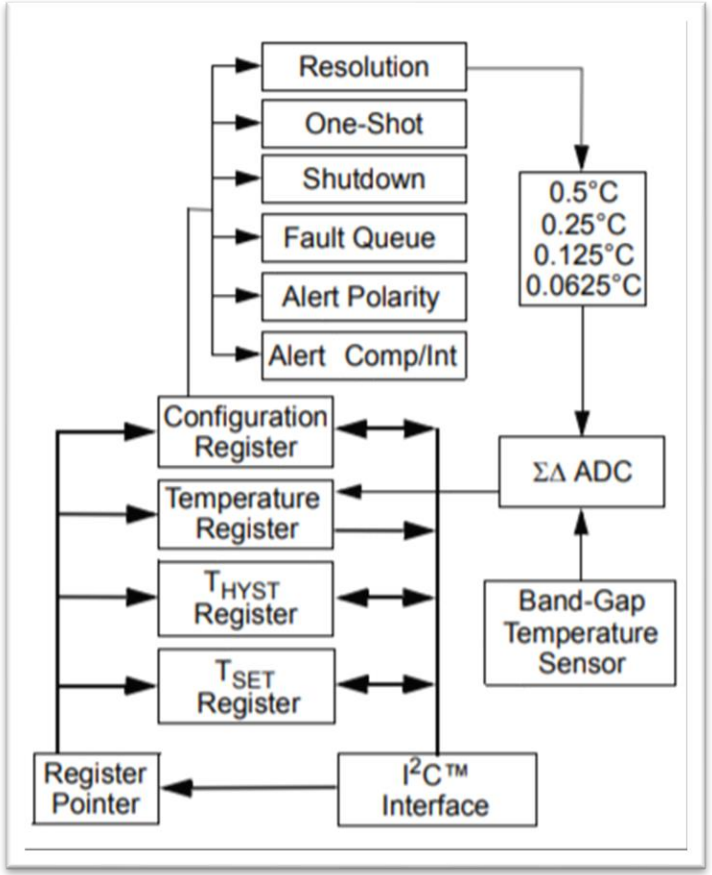
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
0	0	0	0	0	0	P1	P0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-2 **Unimplemented:** Read as '0'

bit 1-0 **Px<1:0>:** Pointer bits

00 = Temperature register (T_A)
01 = Configuration register (CONFIG)
10 = Temperature Hysteresis register (T_{HYST})
11 = Temperature Limit-set register (T_{SET})

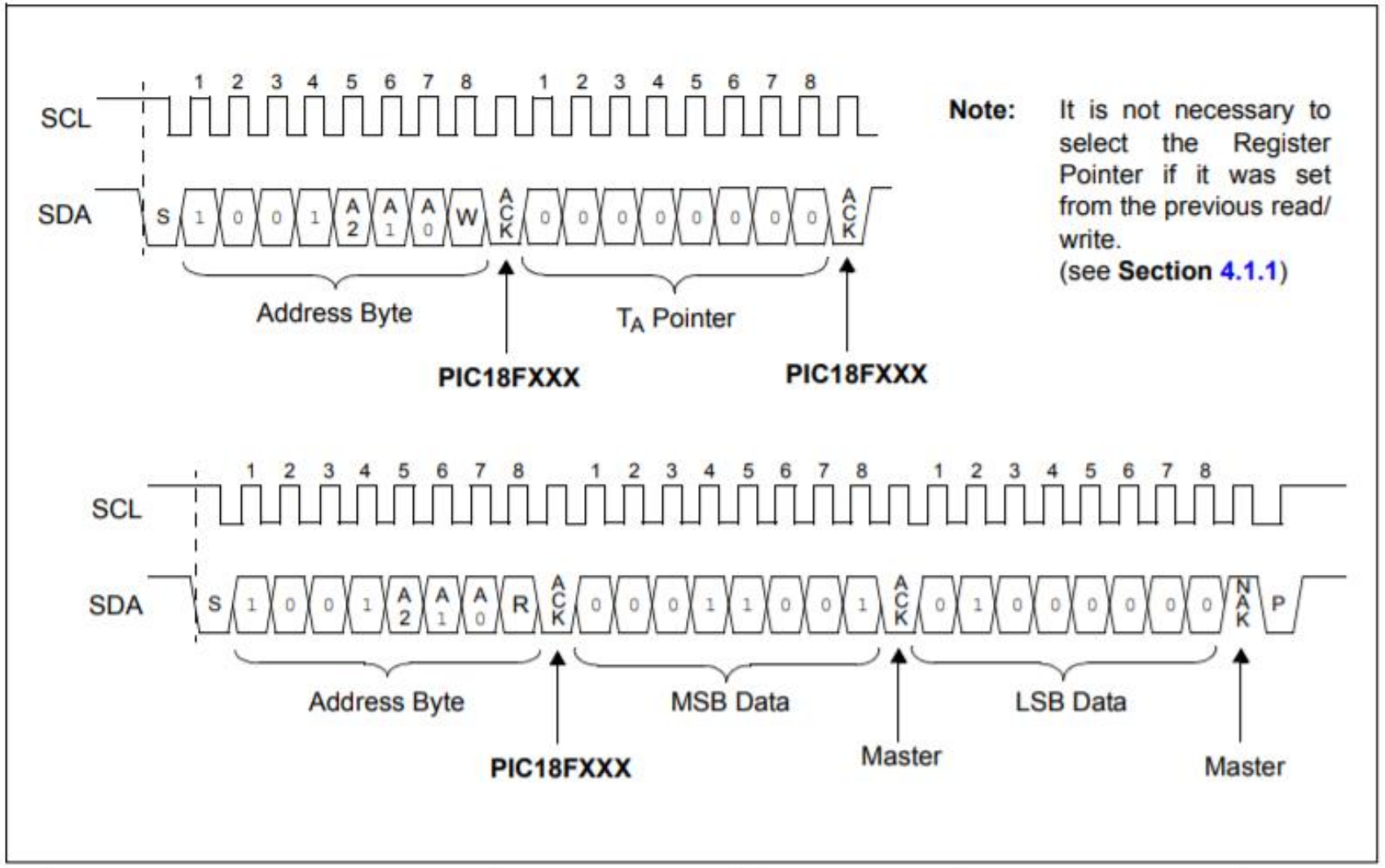


MICROCHIP

MCP9800個暫存器的位元定義

Register Pointer P1 P0	MSB/ LSB	Bit Assignment							
		7	6	5	4	3	2	1	0
Ambient Temperature Register (T_A)									
0 0	MSB	Sign	2^6°C	2^5°C	2^4°C	2^3°C	2^2°C	2^1°C	2^0°C
	LSB	2^{-1}°C	2^{-2}°C	2^{-3}°C	2^{-4}°C	0	0	0	0
Sensor Configuration Register (CONFIG)									
0 1	LSB	One-Shot	Resolution		Fault Queue		ALERT Polarity	COMP/INT	Shutdown
Temperature Hysteresis Register (T_{HYST})									
1 0	MSB	Sign	2^6°C	2^5°C	2^4°C	2^3°C	2^2°C	2^1°C	2^0°C
	LSB	2^{-1}°C	0	0	0	0	0	0	0
Temperature Limit-Set Register (T_{SET})									
1 1	MSB	Sign	2^6°C	2^5°C	2^4°C	2^3°C	2^2°C	2^1°C	2^0°C
	LSB	2^{-1}°C	0	0	0	0	0	0	0

MCP9800溫度暫存器讀取的範例



MICROCHIP

練習4：加入MCP9800溫度Sensor的讀取並顯示

- 增加新的變數、Buffer、.h檔來完成MCP9800的讀取及顯示

Source History

```
23  // ******************************************************************************
24
25  #include <stddef.h>                     // Defines NULL
26  #include <stdbool.h>                    // Defines true
27  #include <stdlib.h>                     // Defines EXIT_FAILURE
28  #include "definitions.h"                // SYS function prototypes
29  #include "OLED128x64.h"
30  #include <string.h>
31  #include <stdio.h>
32  volatile bool bIsI2C_DONE = true ;
33
34  volatile bool bIsTC0_DONE = true ;
35  #define     MCP9800_SLAVE_ADDR  0x4d
36  uint8_t     i2cWrData[8] ;
37  uint8_t     i2cRdData[8] ;
38  uint8_t     OLED_Buffer[20] ;
39  uint8_t     TC0_OverflowTime = 0 ;
```

練習4：加入MCP9800溫度Sensor的讀取並顯示

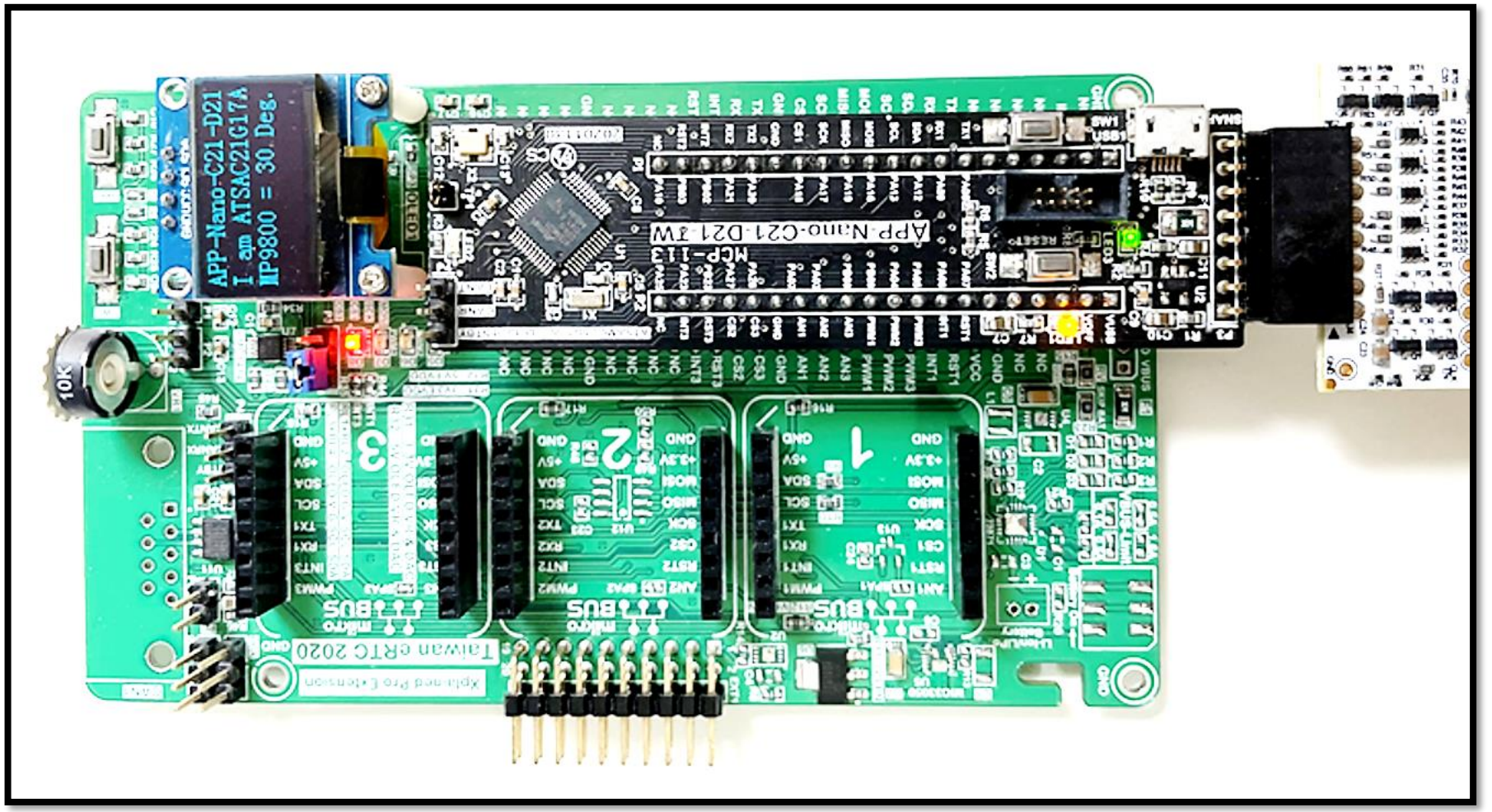
- 增加新的程式碼進行MCP9800的讀取（使用TC0_OverflowTime）
- 使用SERCOM2_I2C_WriteRead()、sprintf()、OLED_Put8x16ASCII()

Source History

```
83      while ( true )
84      {
85          /* Maintain state machines of all polled MPLAB Harmony modules. */
86          SYS_Tasks ( );
87          if (bIsTC0_DONE)
88          {
89              bIsTC0_DONE = 0 ;
90              if (++TC0_OverflowTime == 2)
91              {
92                  TC0_OverflowTime = 0 ;
93                  bIsI2C_DONE = false ;
94                  i2cWrData[0] = 0x00 ;
95                  SERCOM2_I2C_WriteRead(MCP9800_SLAVE_ADDR, i2cWrData, 1, i2cRdData, 2);
96                  while (bIsI2C_DONE == false) {};
97                  sprintf ((char*)OLED_Buffer,"MP9800 = %d Deg.%c", i2cRdData[0],'\0' ) ;
98                  OLED_Put8x16ASCII(0, 4, strlen((char*)OLED_Buffer), OLED_Buffer) ;
99              }
100         }
101     }
```

練習4：加入MCP9800溫度Sensor的讀取並顯示

- 執行結果MCP9800的溫度顯始於第三行



練習5：為你的ATSAMC21G17A加入CAN的傳送功能

練習5：為你的ATSAMC21G17A加入CAN的傳送功能

- 參考資料： **CAN-202D-ATSAMC21 CAN BUS eRTC**
  - http://elearning.microchip.com.tw/modules/tad_link/index.php?link_sn=12

CAN-202D-ATSAMC21
CAN BUS基礎& 利用
Harmony快速實現

網站連結：

http://www.microchip.com.tw/Data_CD/eLearning/eRTC/eRTC_CAN_202D_ATSAMC21_20201124.mp4

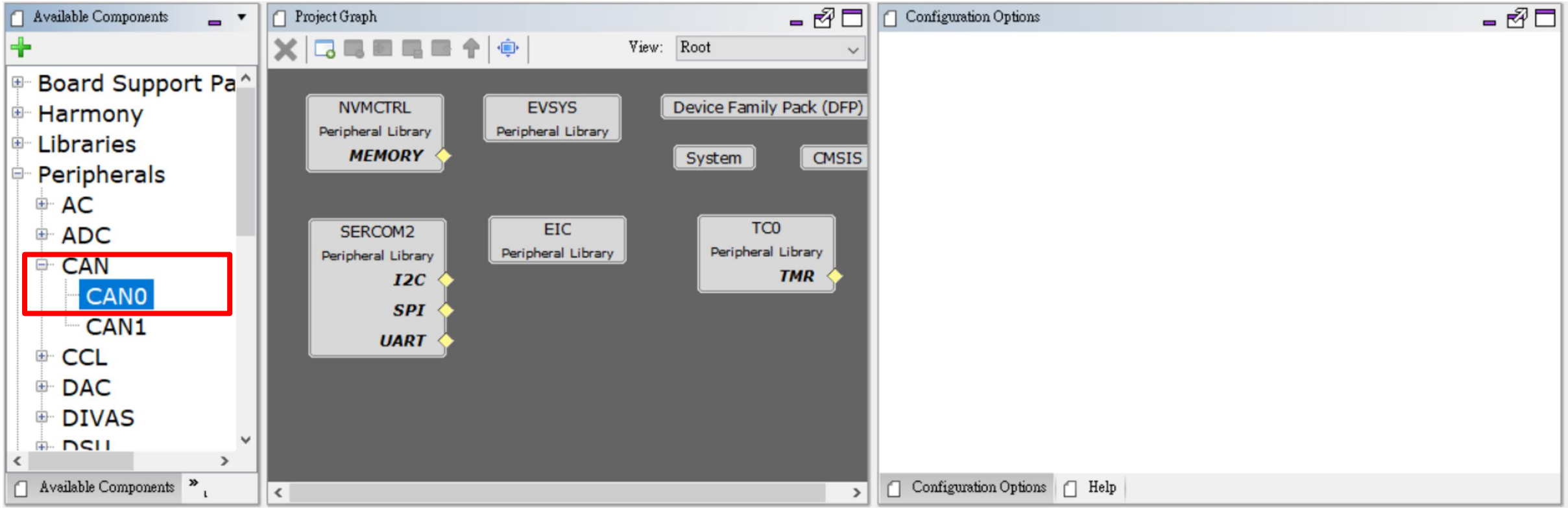
eRTC線上課程錄影 166　　列表

連至：http://www.microchip.com.tw/Data_CD/eL

MICROCHIP

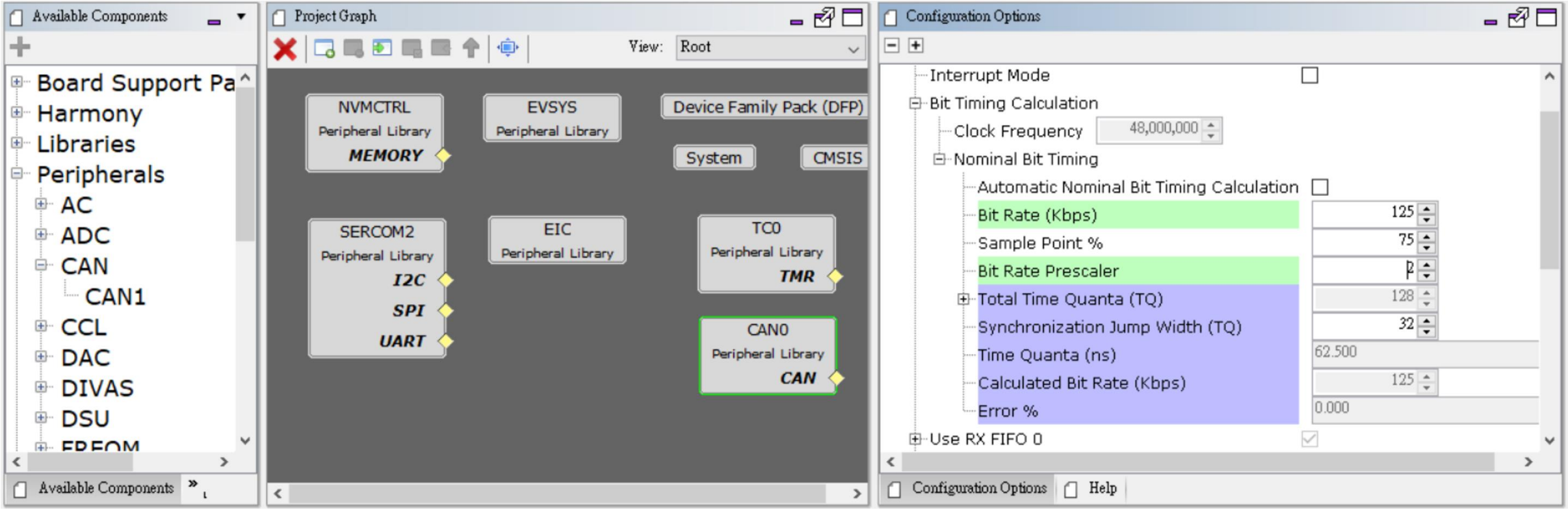
練習5：為你的ATSAMC21G17A加入CAN的傳送功能

- 在Available Component中、選取Peripherals中的CAN0模組



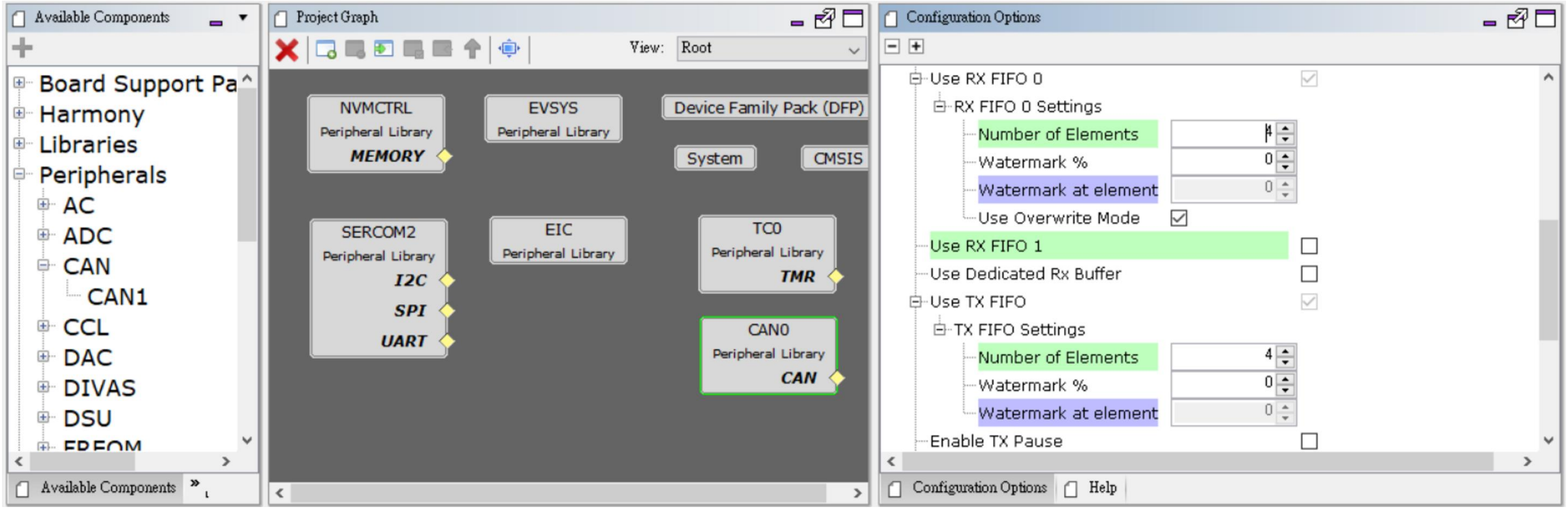
MICROCHIP

練習5：為你的ATSAMC21G17A加入CAN的傳送功能



MICROCHIP

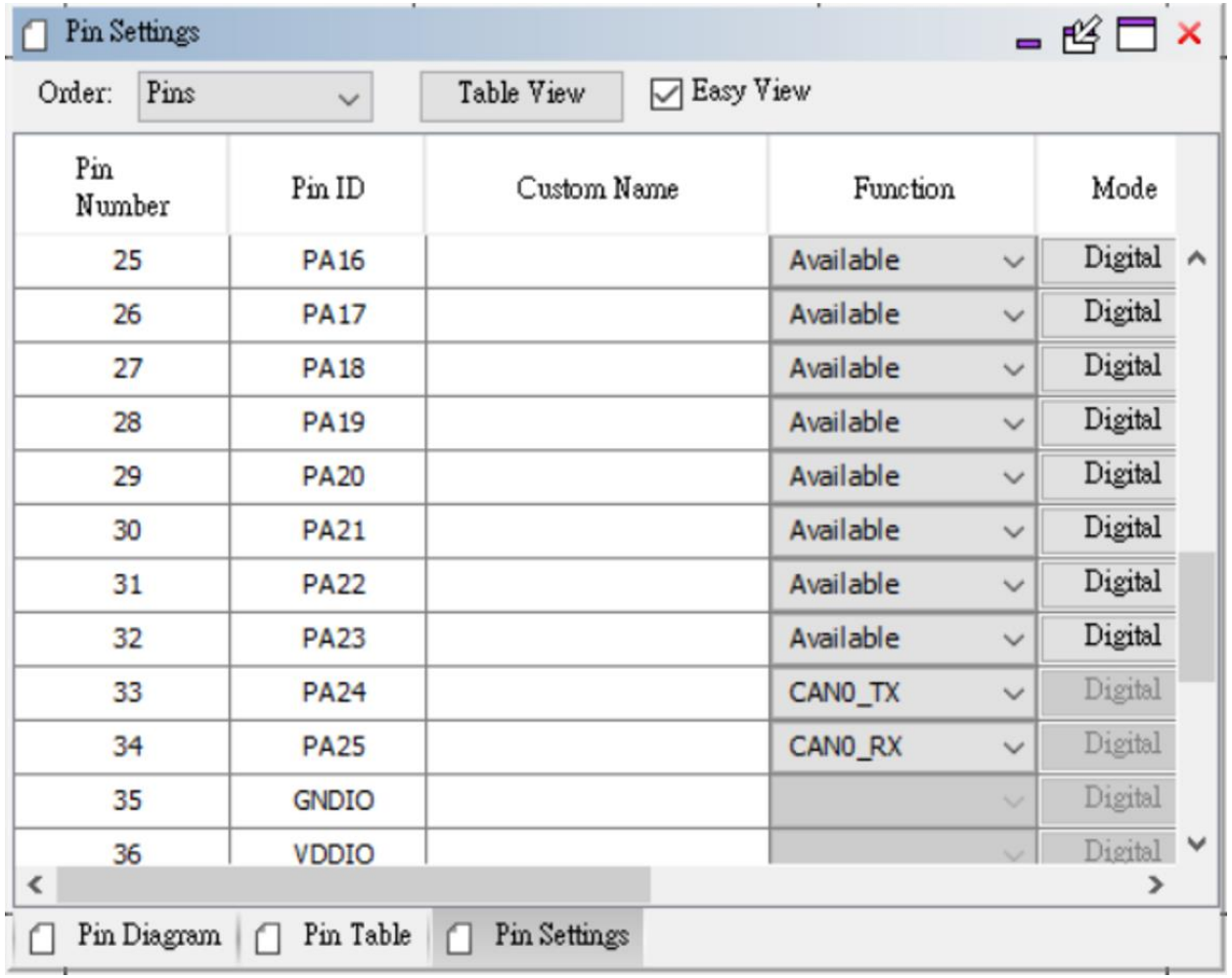
練習5：為你的ATSAMC21G17A加入CAN的傳送功能



MICROCHIP

練習5：為你的ATSAMC21G17A加入CAN的傳送功能

- **PA24 & PA25 要記得規劃為 CANTX & CANRX**



練習5：為你的ATSAMC21G17A加入CAN的傳送功能

- 本練習是為了讓APP-Nano-C21-D21-TW能夠實際工作
- CAN的詳細操作法請參考PLIB說明以及Microchip CAN-202D eRTC

Microchip 32-bit Chip Support Package
Using the Library

Contents | Index | Home Previous | Up | Next Documentation Feedback | Microchip Support

The CAN library supports the Normal and CAN-FD modes. The CAN Normal or CAN-FD mode can transfer message in a polling or an interrupt mode.

CAN Message RAM Configuration

Allocate CAN Message RAM Configuration in contiguous non-cacheable buffe as uint8_t Mcan0MessageRAM[CAN0_MESSAGE_RAM_CONFIG_SIZE] __attribute__((aligned (32))),
here additional attribute such as __attribute__((__section__(".region_nocache"))) or __attribute__((space(data), section (".ram_nocache"))) should be added if cache is enabled and non-cacheable section should be created in linker script.

CAN normal operation with polling

The following example shows the CAN normal mode operation with polling implementation.

```
int main(void)
{
/* Set Message RAM Configuration */
CAN0_MessageRAMConfigSet(Mcan0MessageRAM);

while (1)
{
/* Check if there is a receive FIFO 0 New Message */
if (CAN0_InterruptGet(CAN_INTERRUPT_RF0N_MASK))
{
CAN0_InterruptClear(CAN_INTERRUPT_RF0N_MASK);

/* Receive FIFO 0 New Message */
CAN0_MessageReceive(&messageID, &messageLength, message, 0, CAN_MSG_ATTR_RX_FIFO0, &msgFrameAttr);

/* Transmit back received Message */
CAN0_MessageTransmit(messageID, messageLength, message, CAN_MODE_NORMAL, CAN_MSG_ATTR_TX_FIFO_DATA_FRAME);
}
}
}
```

練習5：為你的ATSAMC21G17A加入CAN的傳送功能

- 請為CAN0的Message RAM依照PLIB說明安排必要的位置

Source History

```
28  #include "definitions.h"                // SYS function prototypes
29  #include "OLED128x64.h"
30  #include <string.h>
31  #include <stdio.h>
32  volatile bool bIsI2C_DONE = true ;
33
34  volatile bool bIsTC0_DONE = true ;
35  #define     MCP9800_SLAVE_ADDR  0x4d
36  uint8_t     i2cWrData[8] ;
37  uint8_t     i2cRdData[8] ;
38  uint8_t     OLED_Buffer[20] ;
39  uint8_t     TC0_OverflowTime = 0 ;
40
41  uint8_t             Can0MessageRAM[CAN0_MESSAGE_RAM_CONFIG_SIZE]  __attribute__((aligned (32))) ;
42  unsigned char       CAN_TxBuffer[64];
43  unsigned char       CAN_Counter = 0 ;
44
```

練習5：為你的ATSAMC21G17A加入CAN的傳送功能

- 參考PLIB 說明，以CAN0_MessageRAMConfigSet()來將CAN0的Message RAM指到對的指標（Can0MessageRAM）

Source History

```
67          bIsTC0_DONE = true ;
68      }
69
70      int main ( void )
71      {
72          /* Initialize all modules */
73          SYS_Initialize ( NULL );
74          EIC_CallbackRegister(EIC_PIN_11, EIC11_Handler,0);
75          TC0_TimerCallbackRegister(TC0_EventHandler, (uintptr_t) NULL) ;
76          SERCOM2_I2C_CallbackRegister(I2C_EventHandler, 0 ) ;
77          TC0_TimerStart();
78          CAN0_MessageRAMConfigSet( Can0MessageRAM) ;
79
80          bIsTC0_DONE = false ;
81          while(bIsTC0_DONE == false) {}
82
83          OLED_Init();
84          OLED_CLS();
85          OLED_Put8x16Str(0, 0, (const unsigned char*) "APP-Nano-C21-D21") ;
```

練習5：為你的ATSAMC21G17A加入CAN的傳送功能

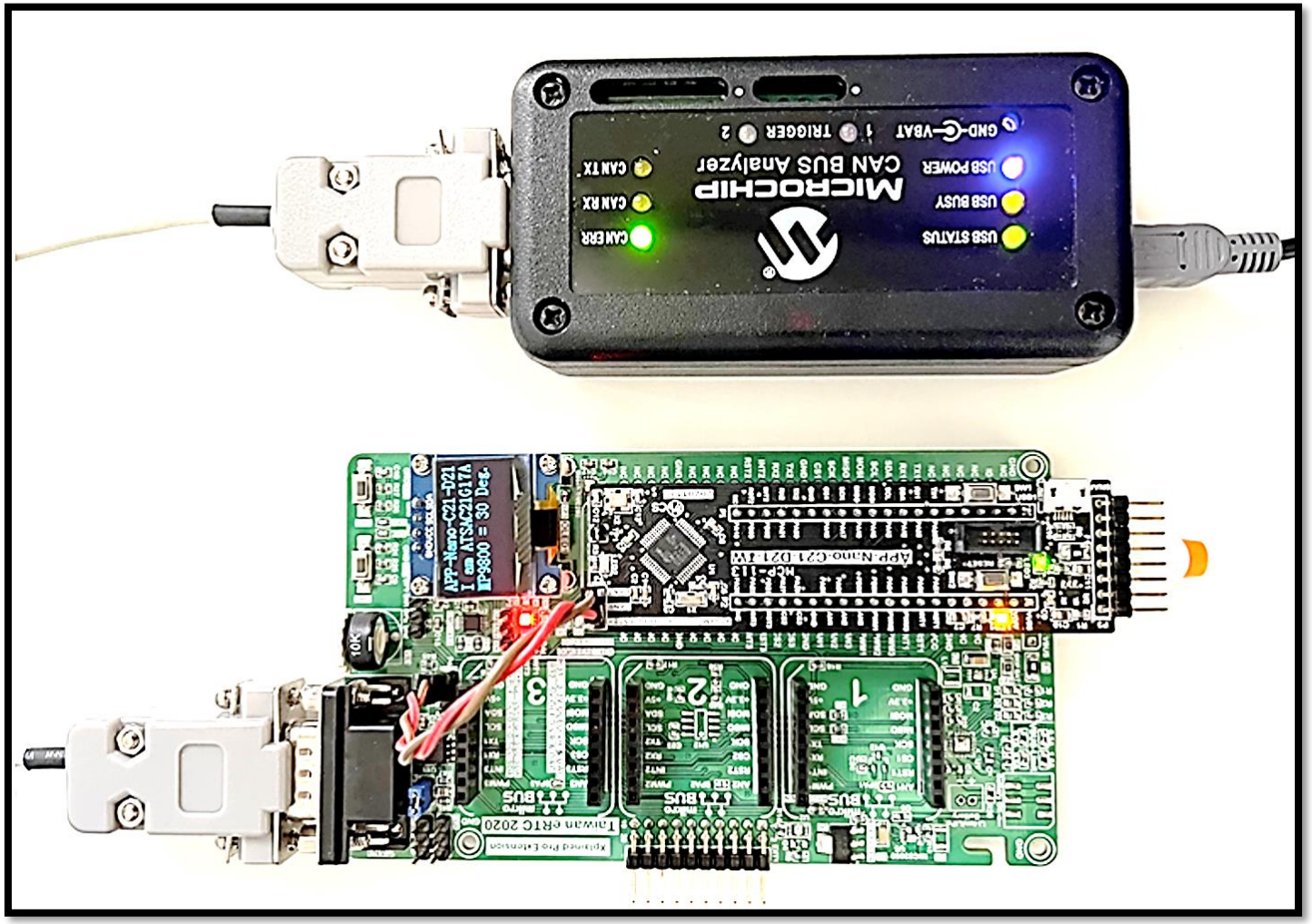
- 參考PLIB說明，使用CAN的相關Function來讀旗號、清旗號以及送資料 — CAN0_MessageTransmit()

Source History

```
92          if (bIsTC0_DONE)
93          {
94            bIsTC0_DONE = 0 ;
95            if (++TC0_OverflowTime == 2)
96            {
97                TC0_OverflowTime = 0 ;
98                bIsI2C_DONE = false ;
99                i2cWrData[0] = 0x00 ;
100               SERCOM2_I2C_WriteRead(MCP9800_SLAVE_ADDR, i2cWrData, 1, i2cRdData, 2);
101               while (bIsI2C_DONE == false) {};
102               sprintf ((char*)OLED_Buffer,"MP9800 = %d Deg.%c", i2cRdData[0],'\0' ) ;
103               OLED_Put8x16ASCII(0, 4, strlen((char*)OLED_Buffer), OLED_Buffer) ;
104
105               if (CAN0_TxFIFOIsFull() == false)
106               {
107                   CAN_TxBuffer[0] = i2cRdData[0] ;
108                   CAN_TxBuffer[1] = 88 ;
109                   CAN_TxBuffer[2] = 99 ;
110                   CAN_TxBuffer[3] = CAN_Counter ++ ;
111                   CAN0_MessageTransmit(0x100, 4, CAN_TxBuffer,CAN_MODE_NORMAL, CAN_MSG_ATTR_TX_FIFO_DATA_FRAME) ;
112               }
```

練習5：為你的ATSAMC21G17A加入CAN的傳送功能

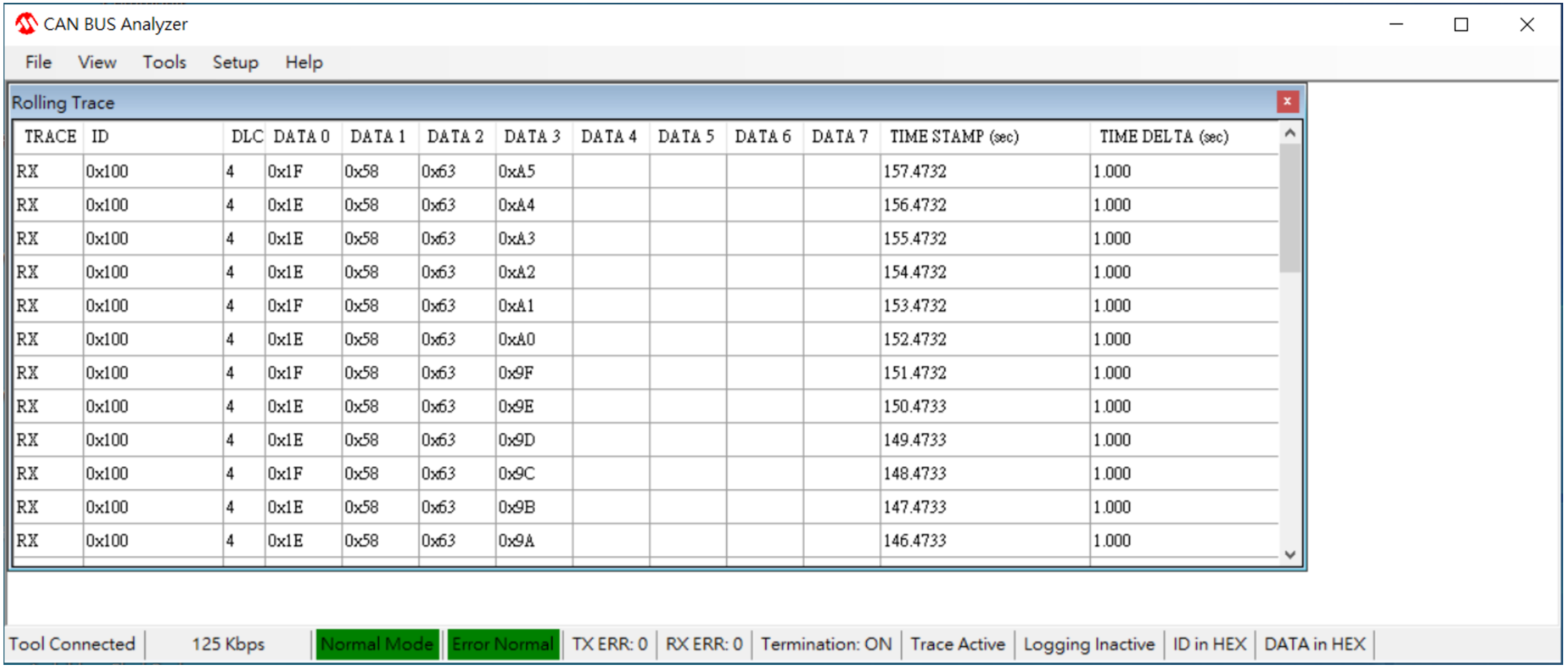
- 練習5的硬體連接（1）：使用CAN Bus Analyzer



MICROCHIP

練習5：為你的ATSAMC21G17A加入CAN的傳送功能

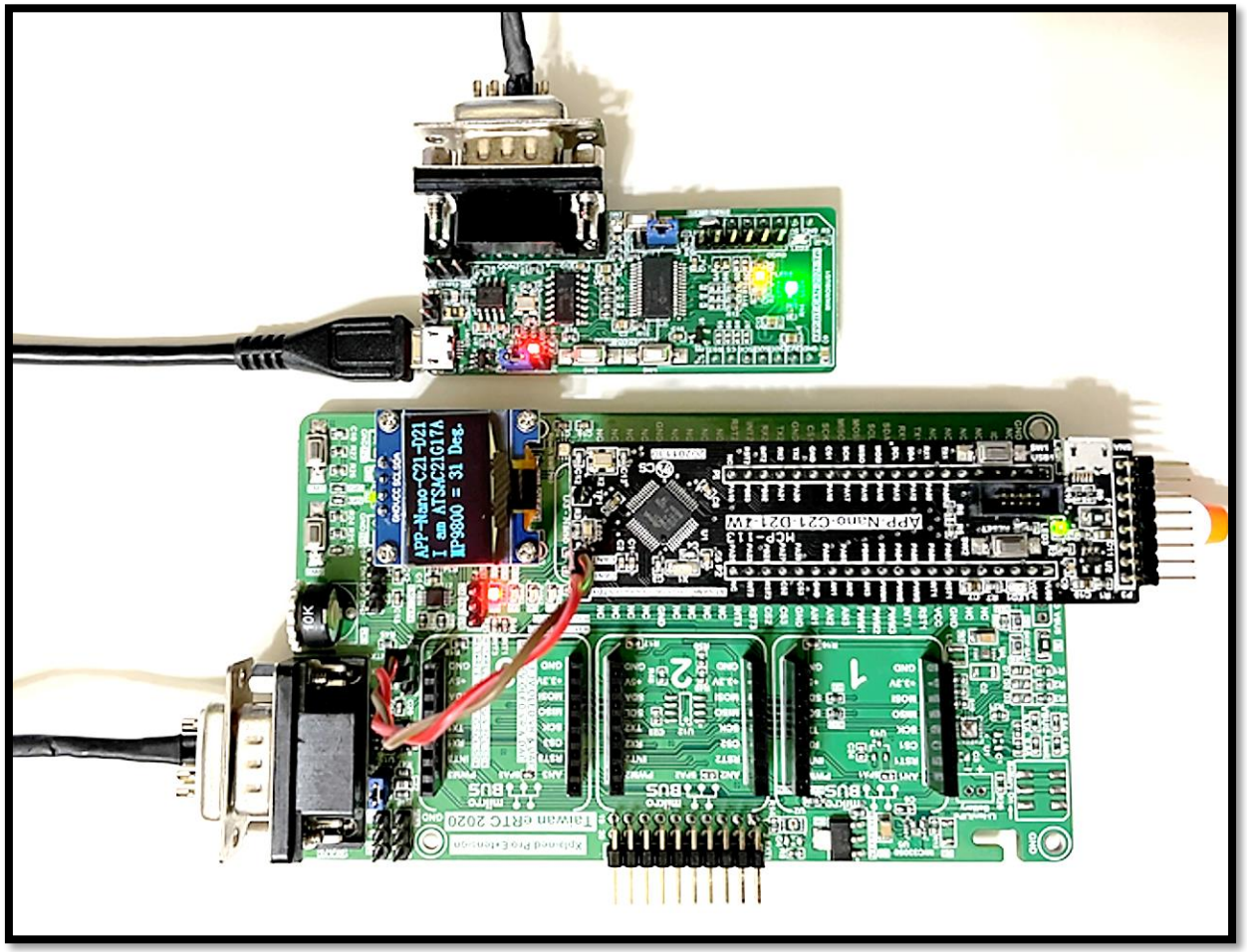
- 使用CAN Bus Analyzer的結果



TRACE	ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7	TIME STAMP (sec)	TIME DELTA (sec)
RX	0x100	4	0x1F	0x58	0x63	0xA5					157.4732	1.000
RX	0x100	4	0x1E	0x58	0x63	0xA4					156.4732	1.000
RX	0x100	4	0x1E	0x58	0x63	0xA3					155.4732	1.000
RX	0x100	4	0x1E	0x58	0x63	0xA2					154.4732	1.000
RX	0x100	4	0x1F	0x58	0x63	0xA1					153.4732	1.000
RX	0x100	4	0x1E	0x58	0x63	0xA0					152.4732	1.000
RX	0x100	4	0x1F	0x58	0x63	0x9F					151.4732	1.000
RX	0x100	4	0x1E	0x58	0x63	0x9E					150.4733	1.000
RX	0x100	4	0x1E	0x58	0x63	0x9D					149.4733	1.000
RX	0x100	4	0x1F	0x58	0x63	0x9C					148.4733	1.000
RX	0x100	4	0x1E	0x58	0x63	0x9B					147.4733	1.000
RX	0x100	4	0x1E	0x58	0x63	0x9A					146.4733	1.000

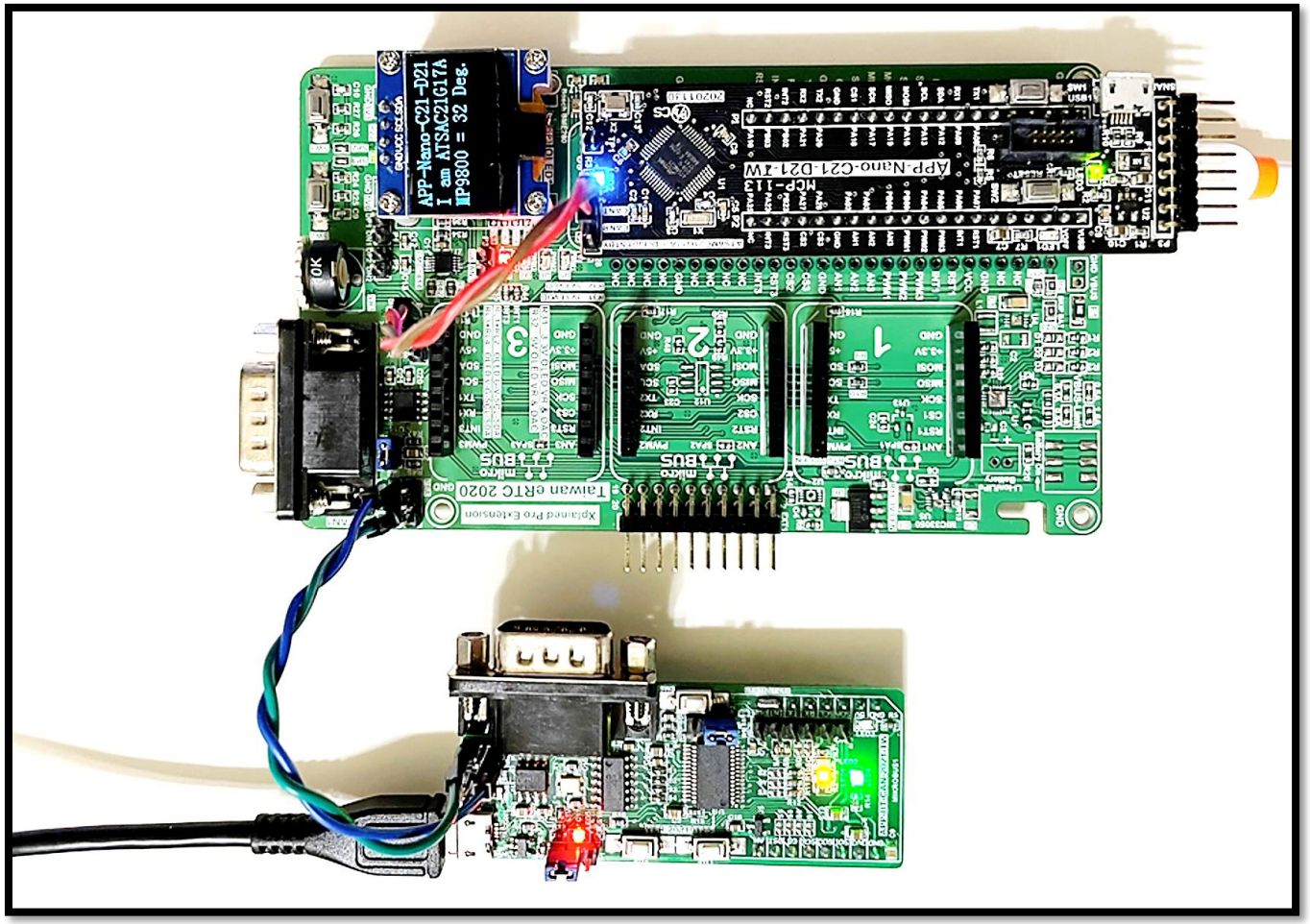
練習5：為你的ATSAMC21G17A加入CAN的傳送功能

- **練習5的硬體連接（2）：使用APP-BT-CAN-2021-TW + DB9**
  - APP-BT-CAN-2021-TW的Bus Monitor F/W (.Hex)會一起於此eRTC的範例程式包中



練習5：為你的ATSAMC21G17A加入CAN的傳送功能

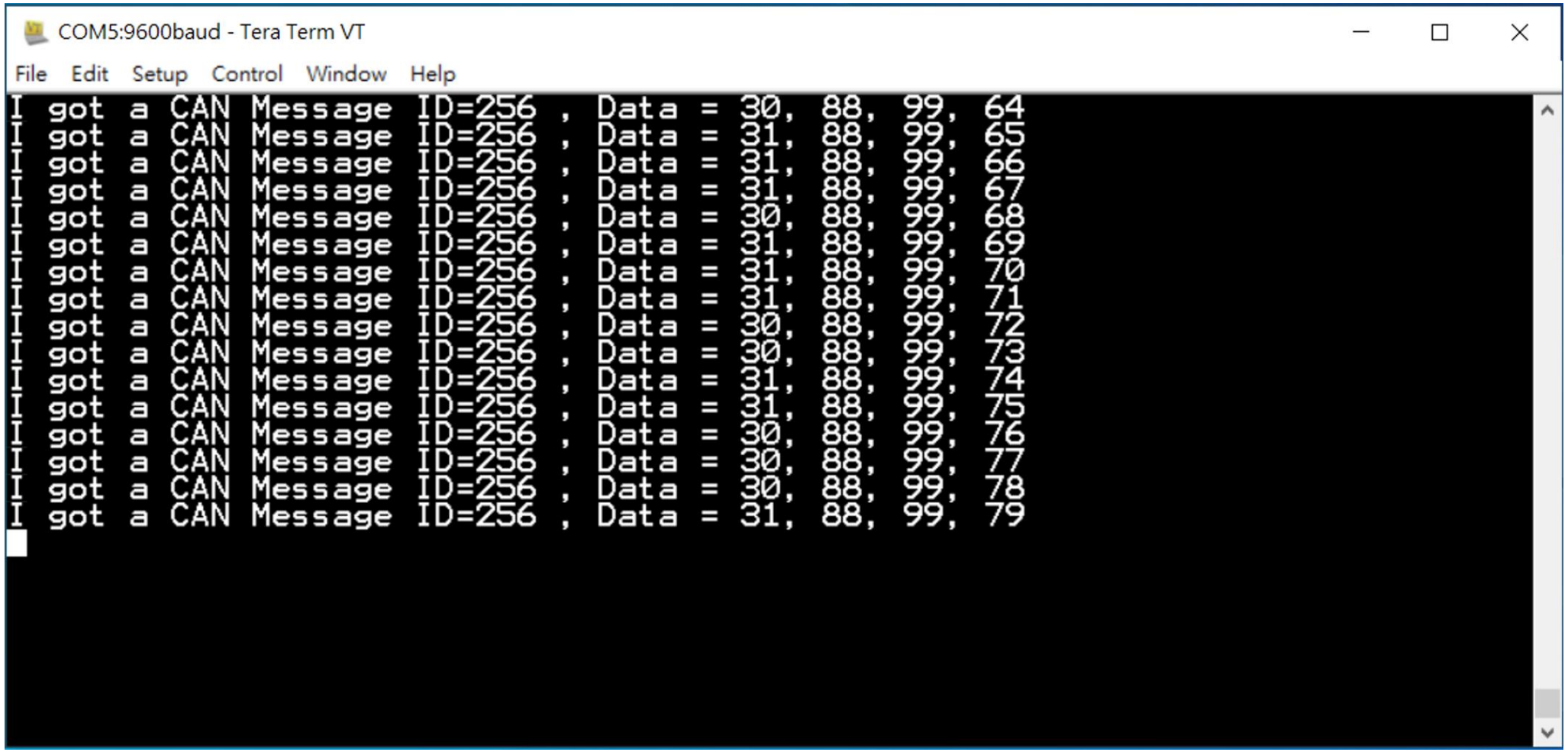
- 練習5的硬體連接（3）：使用APP-BT-CAN-2021-TW + 杜邦跳線



MICROCHIP

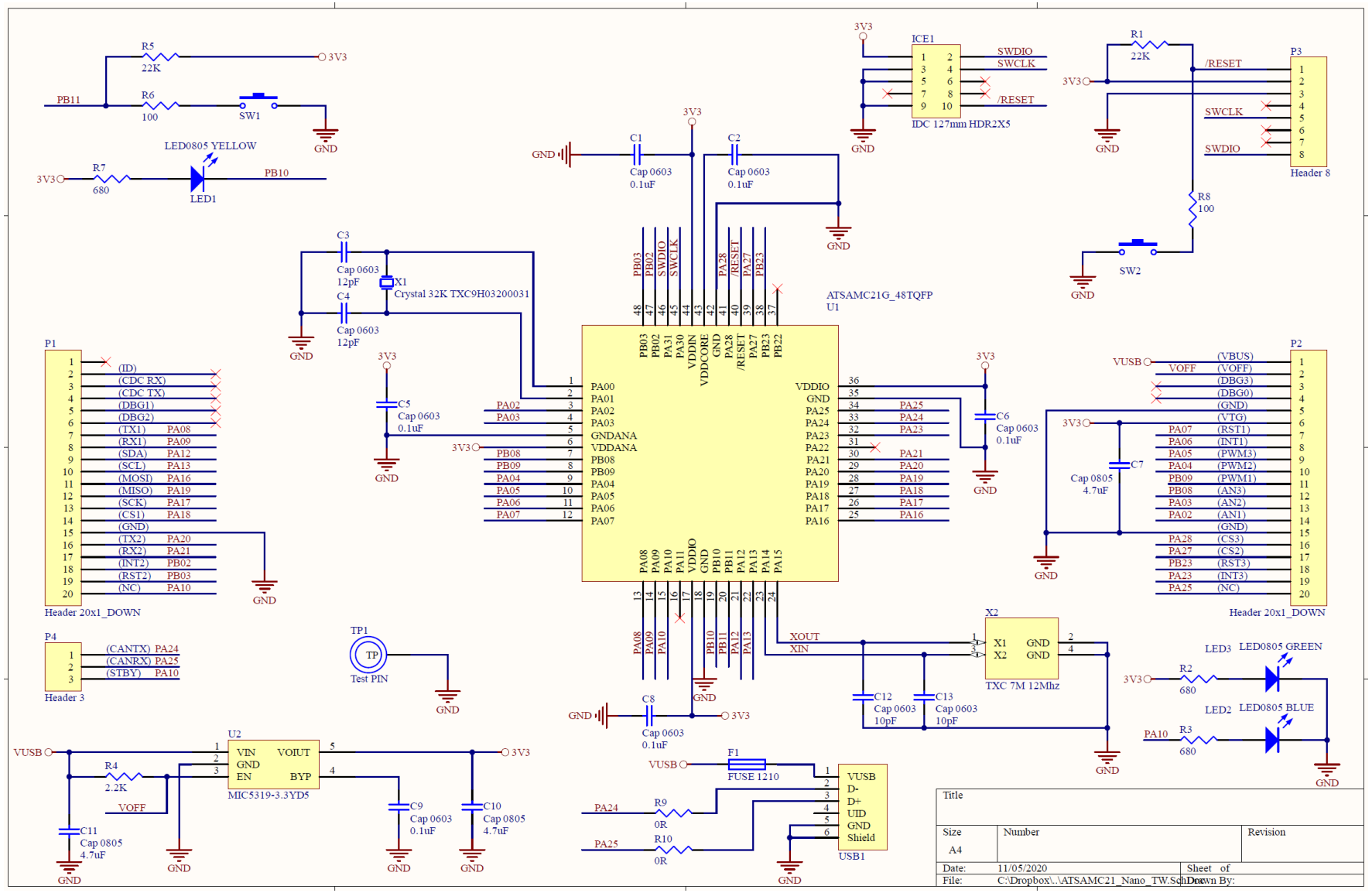
練習5：為你的ATSAMC21G17A加入CAN的傳送功能

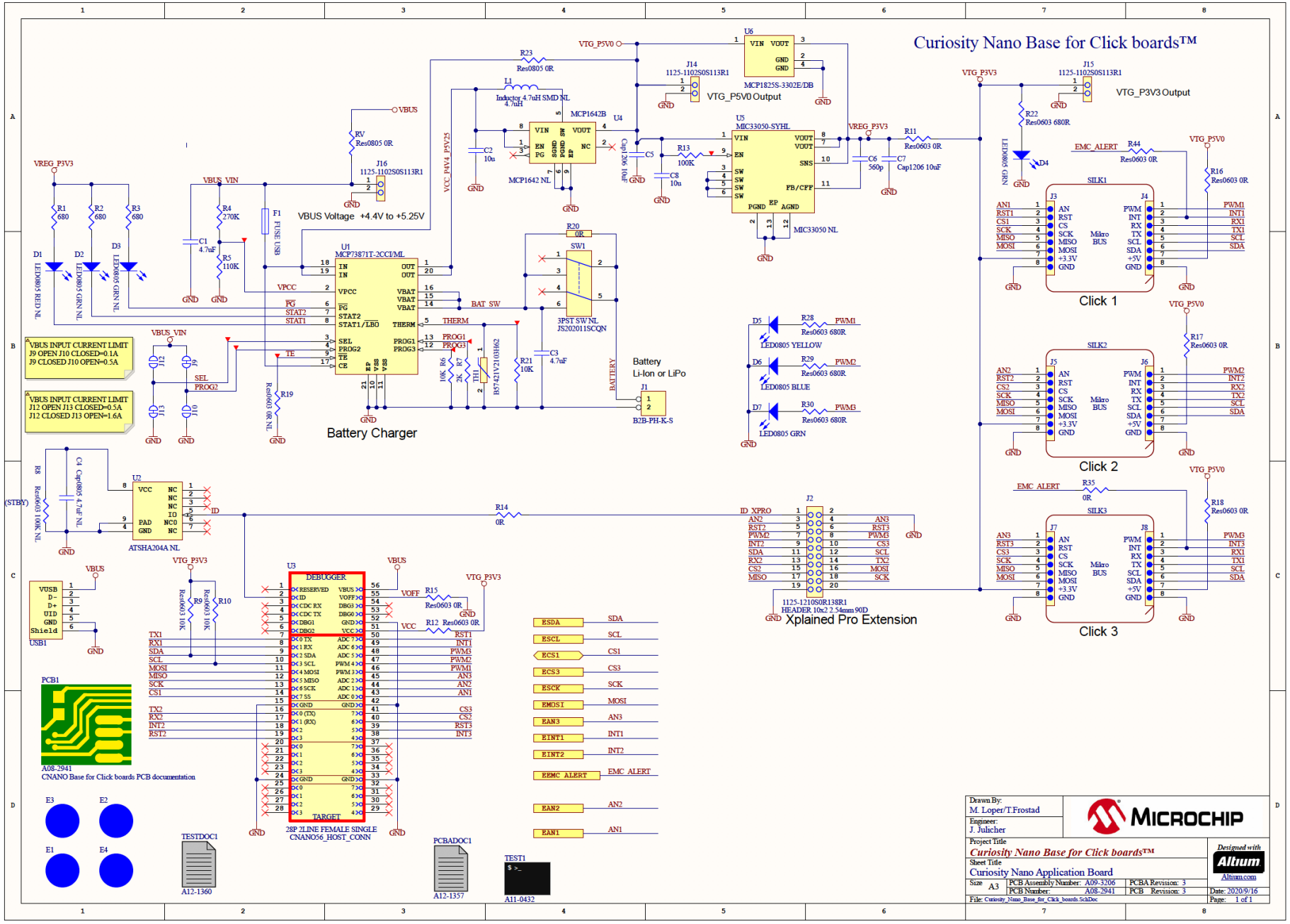
- 使用APP-BT-CAN-2021-TW的結果

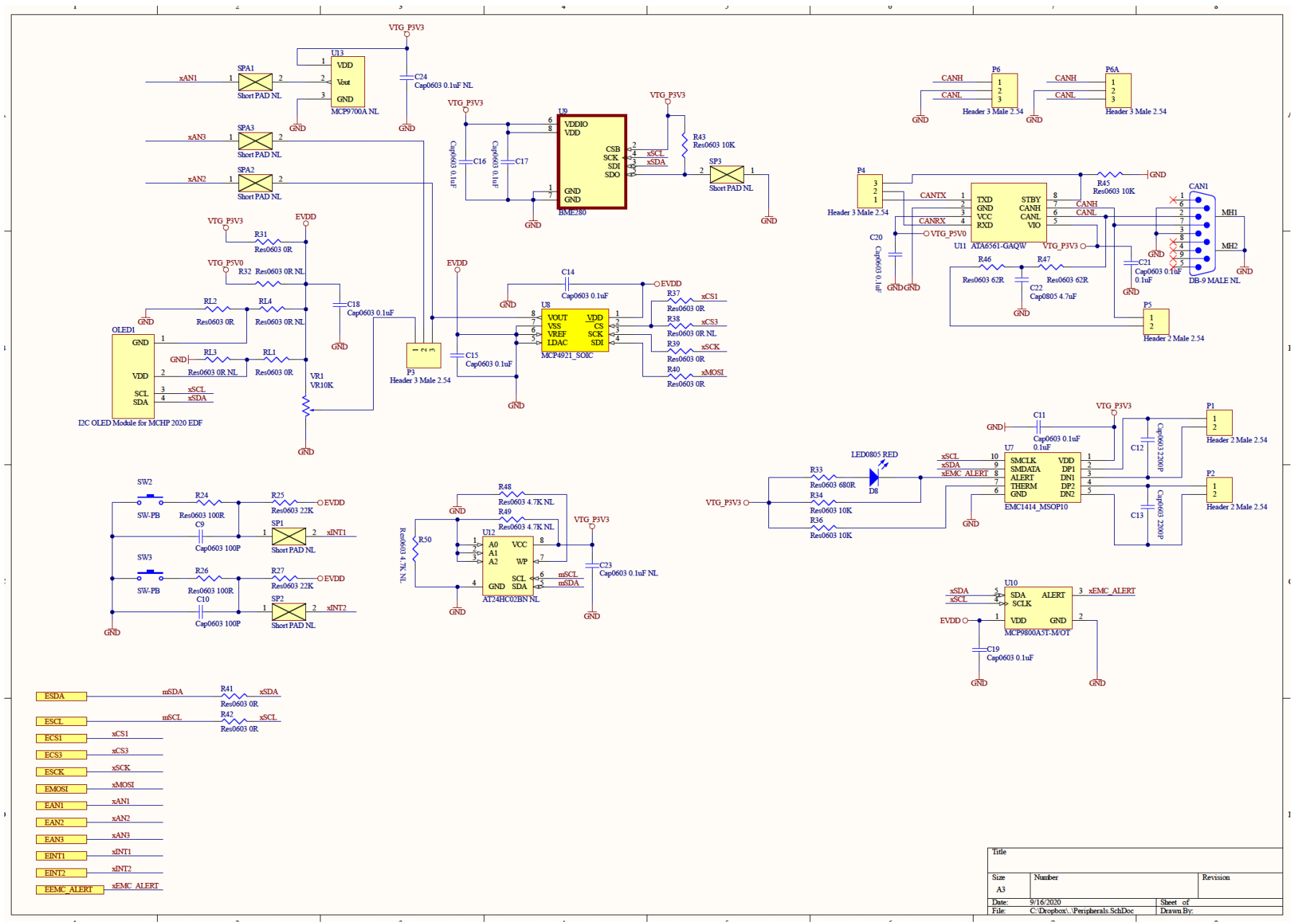


附錄： APP-Nano-C21-D21-TW APP-Nano-BASE-TW 線路圖

APP-Nano-C21-D21-TW線路圖




Curiosity Nano Base for Click boards™
VTG_P5V0 Output
VTG_P3V3 Output
VBUS Voltage +4.4V to +5.25V
VBUS INPUT CURRENT LIMIT
J9 OPEN J10 CLOSED=0.1A
J9 CLOSED J10 OPEN=0.5A
VBUS INPUT CURRENT LIMIT
J12 OPEN J13 CLOSED=0.5A
J12 CLOSED J13 OPEN=1.6A
Battery Charger
Battery
Li-Ion or LiPo
Click 1
Click 2
Click 3
Xplained Pro Extension
ATSHA204A NL
USB1
PCB1
A08-2941
CNANO Base for Click boards PCB documentation
TESTDOC1
A12-1360
PCBADOC1
A12-1357
TEST1
A11-0432
Drawn By:
M. Loper/T.Frostad
Engineer:
J. Julicher
Project Title
Curiosity Nano Base for Click boards™
Sheet Title
Curiosity Nano Application Board
Size A3
PCB Assembly Number: A09-3206
PCB Number: A08-2941
PCBA Revision: 3
PCB Revision: 3
Date: 2020/9/16
Page: 1 of 1
File: Curiosity_Nano_Base_for_Click_boards.SchDoc
Designed with Altium
Altium.com


VTG_P3V3
U13
VDD
Vout
GND
MCP9700A NL
C24
Cap0603 0.1uF NL
SPA1
xAN1
Short PAD NL
SPA3
xAN3
SPA2
xAN2
U9
VDDIO
VDD
CSB
SCK
SDI
SDO
GND
BME280
C16
C17
R43
Res0603 10K
SP3
EVDD
R31
Res0603 0R
R32 Res0603 0R NL
RL2
RL4
RL3
RL1
Res0603 0R
Res0603 0R NL
C18
Cap0603 0.1uF
OLED1
I2C OLED Module for MCHP 2020 EDF
VR1
VR10K
P3
Header 3 Male 2.54
C14
C15
U8
VOUT
VSS
VREF
LDAC
CS
MCP4921_SOIC
R37
R38
R39
R40
xCS1
xCS3
xSCK
xMOSI
SW2
SW3
SW-PB
R24
R25
R26
R27
Res0603 100R
Res0603 22K
C9
C10
Cap0603 100P
SP1
SP2
xINT1
xINT2
R48
R49
Res0603 4.7K NL
R50
U12
A0
A1
A2
WP
SCL
SDA
AT24HC02BN NL
C23
Cap0603 0.1uF NL
mSCL
mSDA
P6
P6A
CANH
CANL
Header 3 Male 2.54
P4
CANTX
CANRX
VTG_P5V0
U11 ATA6561-GAQW
TXD
GND
VCC
RXD
STBY
CANH
CANL
VIO
R45
R46
R47
Res0603 62R
C22
Cap0805 4.7uF
C20
C21
Cap0603 0.1uF
CAN1
DB-9 MALE NL
MH1
MH2
P5
Header 2 Male 2.54
LED0805 RED
D8
R33
R34
R36
Res0603 680R
Res0603 10K
C11
C12
C13
Cap0603 2200P
U7
SMCLK
SMDATA
ALERT
THERM
DP1
DN1
DP2
DN2
EMC1414_MSOP10
P1
P2
U10
SDA
SCLK
MCP9800A5T-M/OT
xEMC_ALERT
C19
ESDA
ESCL
ECS1
ECS3
ESCK
EMOSI
EAN1
EAN2
EAN3
EINT1
EINT2
EEMC_ALERT
R41
R42
Title
Size
A3
Number
Revision
Date: 9/16/2020
File: C:\Dropbox\..\Peripherals.SchDoc
Sheet of
Drawn By:

更多使用APP-Nano-BASE-TW完成的eRTC線上學習影片

- **PIC16F18446-101**
  - http://www.microchip.com.tw/Data_CD/eLearning/RAW_Video_PIC16F18446_V2.mp4
- **PIC16F18446-201**
  - http://www.microchip.com.tw/Data_CD/eLearning/eRTC_PIC16F18446_201_All.mp4

PIC16F18446-101

eRTC線上課程錄影 |
http://www.microchip.com.tw/Data_CD/eLearning/RAW_Video_PIC16F18446_V

PIC16F18446-201

eRTC線上課程錄影 |
http://www.microchip.com.tw/Data_CD/eLearning/eRTC_PIC16F18446_201_Al

MICROCHIP

APP_C21_D21_LearningKit.zip eRTC軟體包

- 可以由**Microchip台灣網站的Microchip Webinar資料區下載**
  http://www.microchip.com.tw/modules/tad_link/index.php?op=tad_link_form&link_sn=58

名稱	修改日期	類型	大小
APP_BT_CAN_SimpleMonitor.hex	2021/5/31 下午 07:52	HEX 檔案	192 KB
APP_C21_D21_TW_101_Lab_2	2021/6/1 上午 08:41	壓縮的 (zipped) ...	537 KB
APP_C21_D21_TW_101_Lab_All	2021/6/1 下午 01:06	壓縮的 (zipped) ...	729 KB
APP_C21_D21_TW_Discovery_101	2021/6/1 下午 01:26	Adobe Acrobat ...	17,807 KB
APP_Nano_C21_D21_TW_Schematic	2020/12/1 下午 12:25	Adobe Acrobat ...	666 KB
APP_C21_D21_TW_Manual_20210531	2021/5/31 上午 09:28	Adobe Acrobat ...	3,069 KB

謝謝
Thanks!