



25004 MCU1 Lab Manual



Lab 1 – Introduction Lab

MPLAB® X IDE project creation & Systick

Lab 1 Objectives



CREATE A PROJECT
IN MPLAB® X IDE



SET UP CLOCKS
USING MCC
HARMONY V3



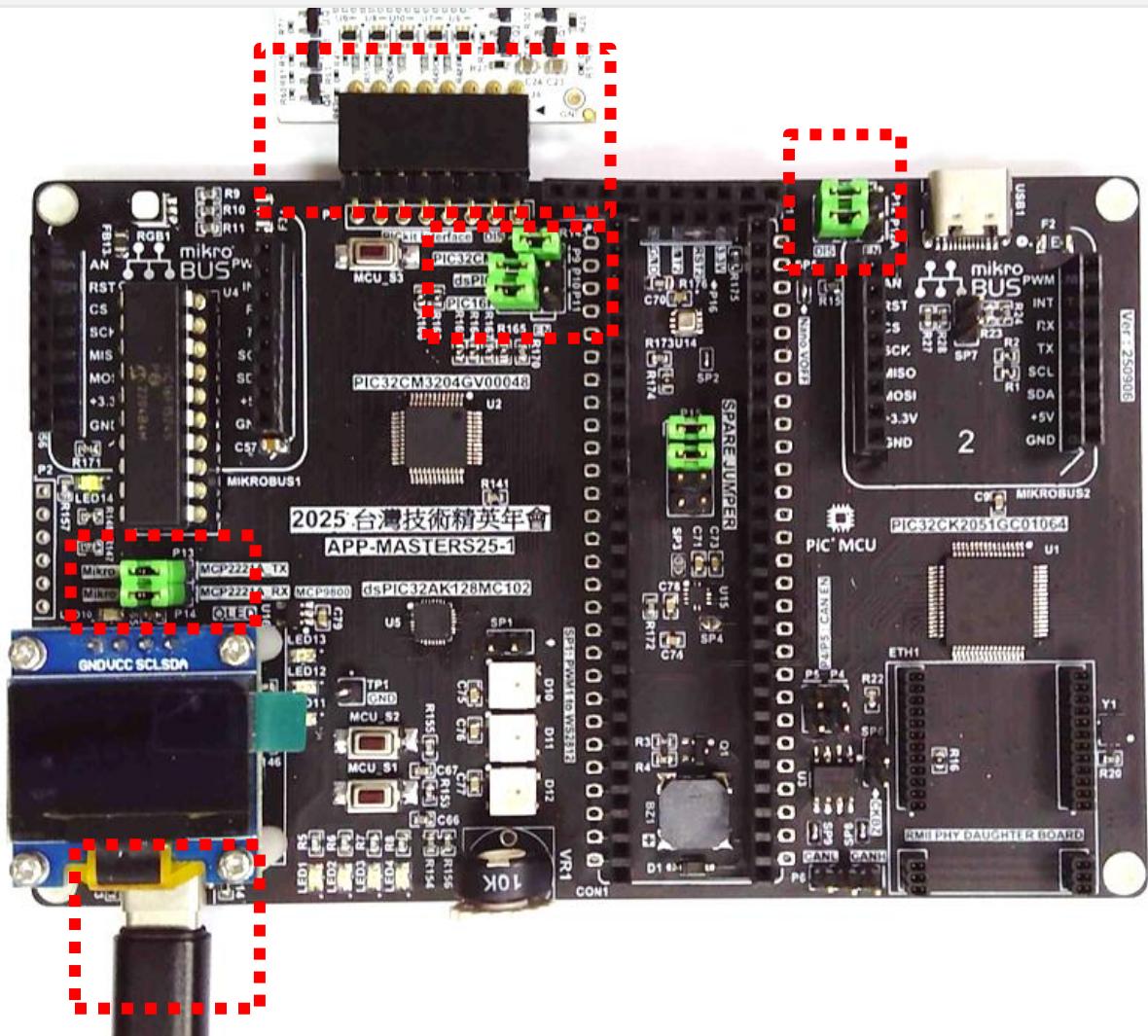
設定 SysTick 來獲得一個 100ms 的時間來源並使用其中斷來驅動 LED1



練習燒錄成是並且觀察實際執行結果與相關的檔案

Lab1 – MPLAB® X IDE project creation & Systick

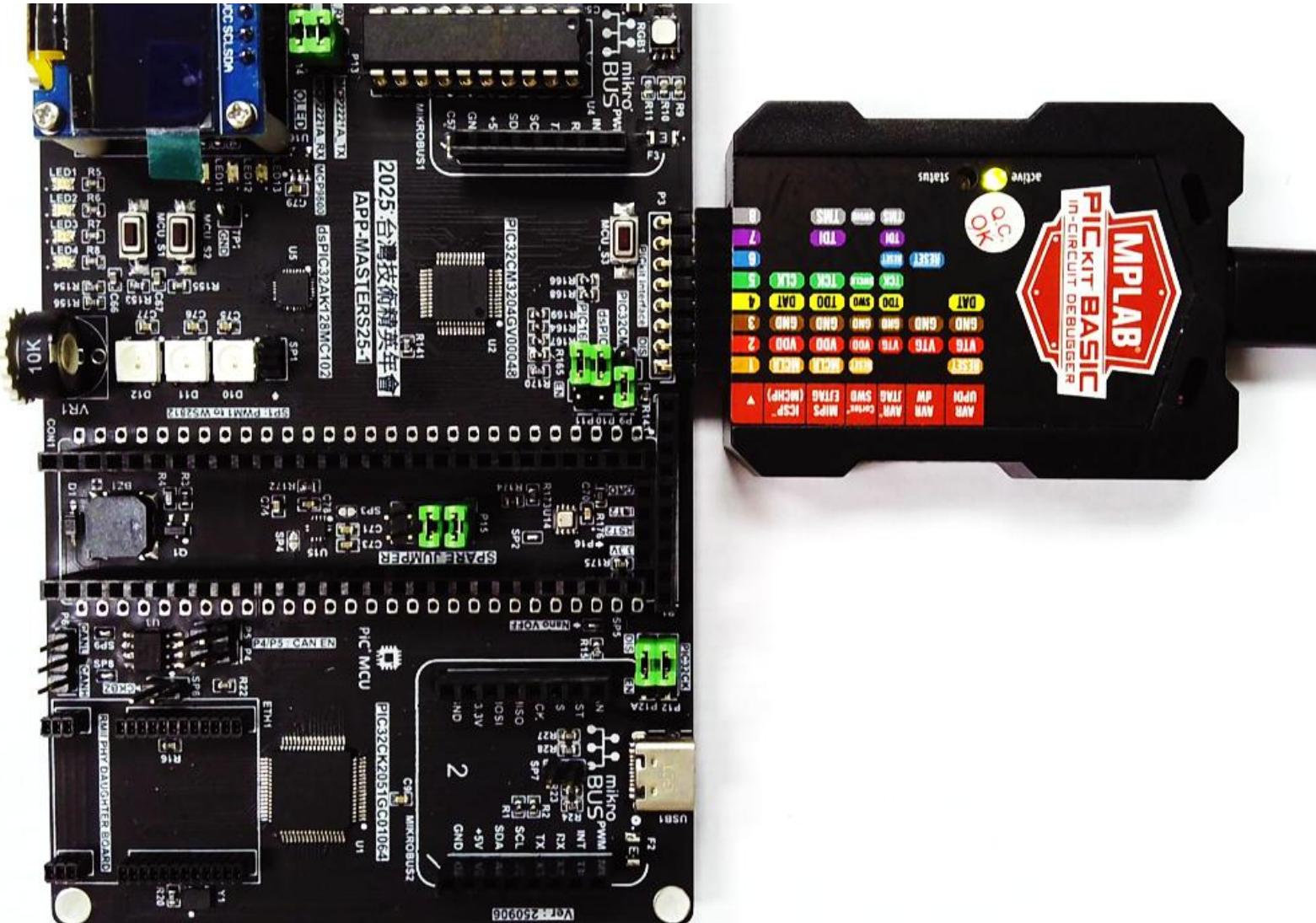
APP-MASTERS25-1 實驗板的 Jumper 設定以及燒錄器連接方式



- USB2 Connector 以 Type-C Cable 連接到電腦
- P3 : 連結到 Programmer
 - SNAP, PICkit-Basic, PICkit-5
- P9 : 插接 1&2 來 Enable PIC32CM3204GV00048
- P10, P11, P12, P12A 都插接在 2&3 的位置來 DISABLE 其它的 MCU
- P13 & P14 要 Close , MCP2221A 才能獲得 MCU 的 UART 信號

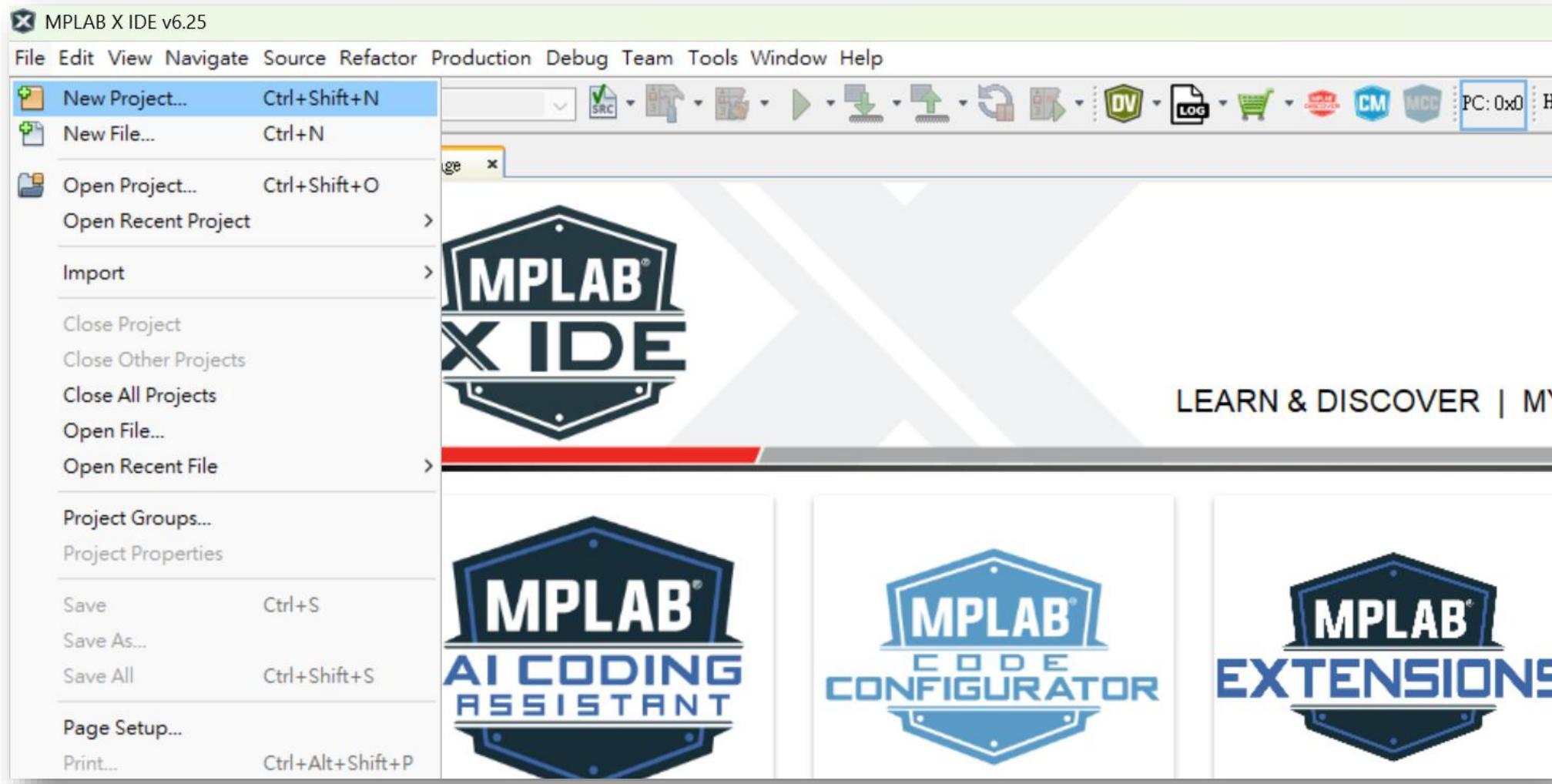
Lab1 – MPLAB® X IDE project creation & Systick

如果您有自己購買的 PICkit BASIC or PICkit-5 也能使用



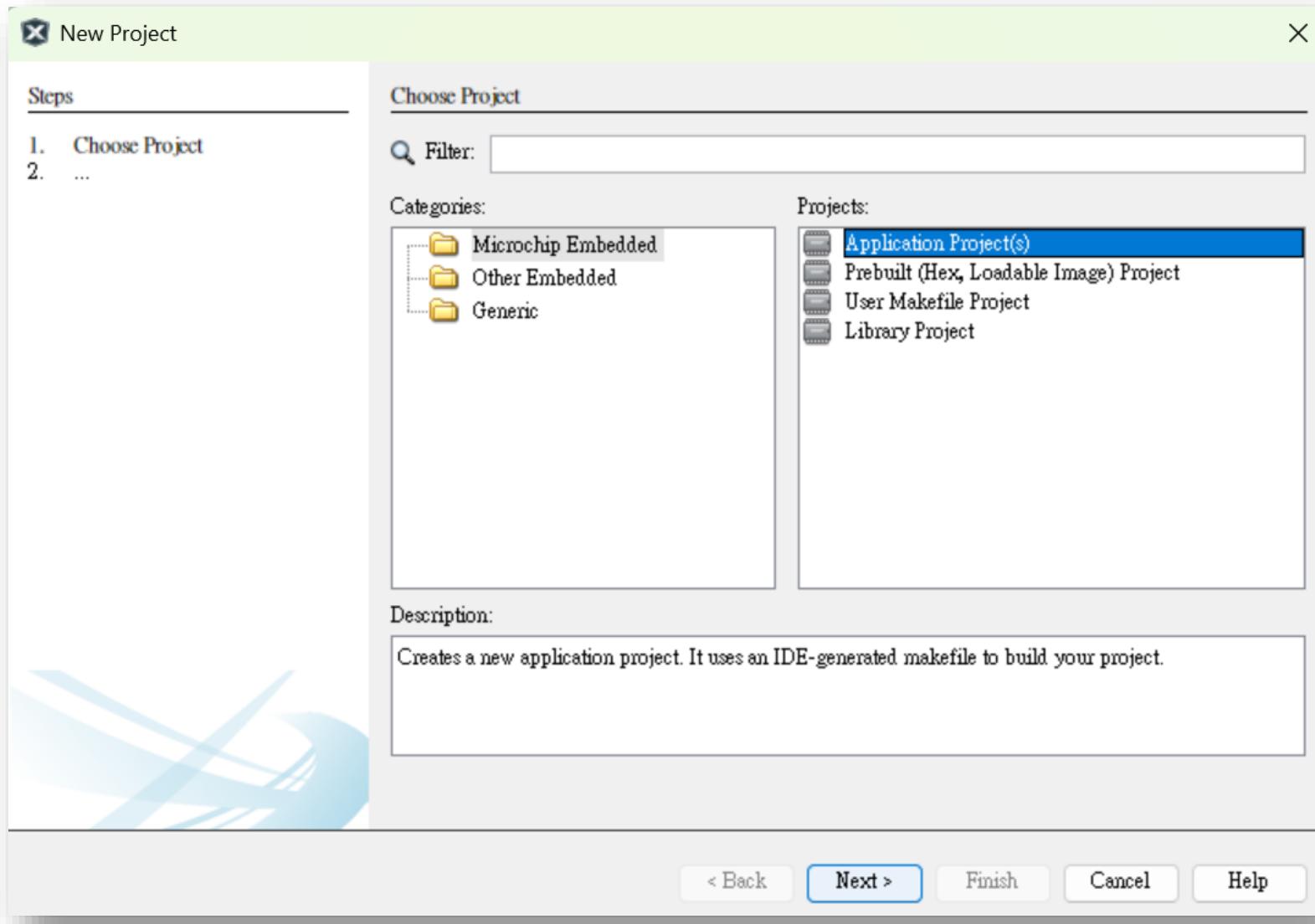
Lab1 – MPLAB® X IDE project creation & Systick

File -> New Project



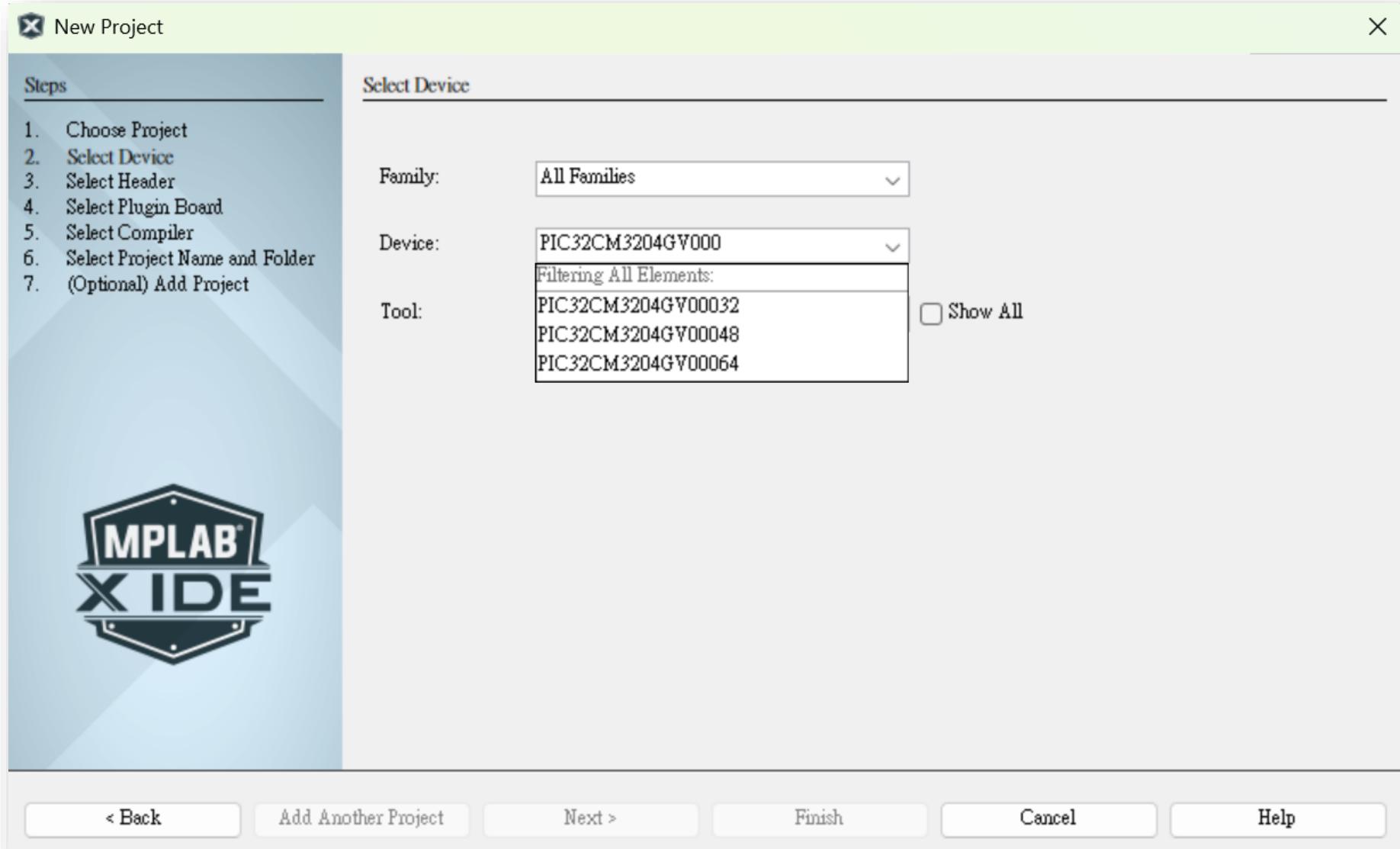
Lab1 – MPLAB® X IDE project creation & Systick

Select “Application Project(s)



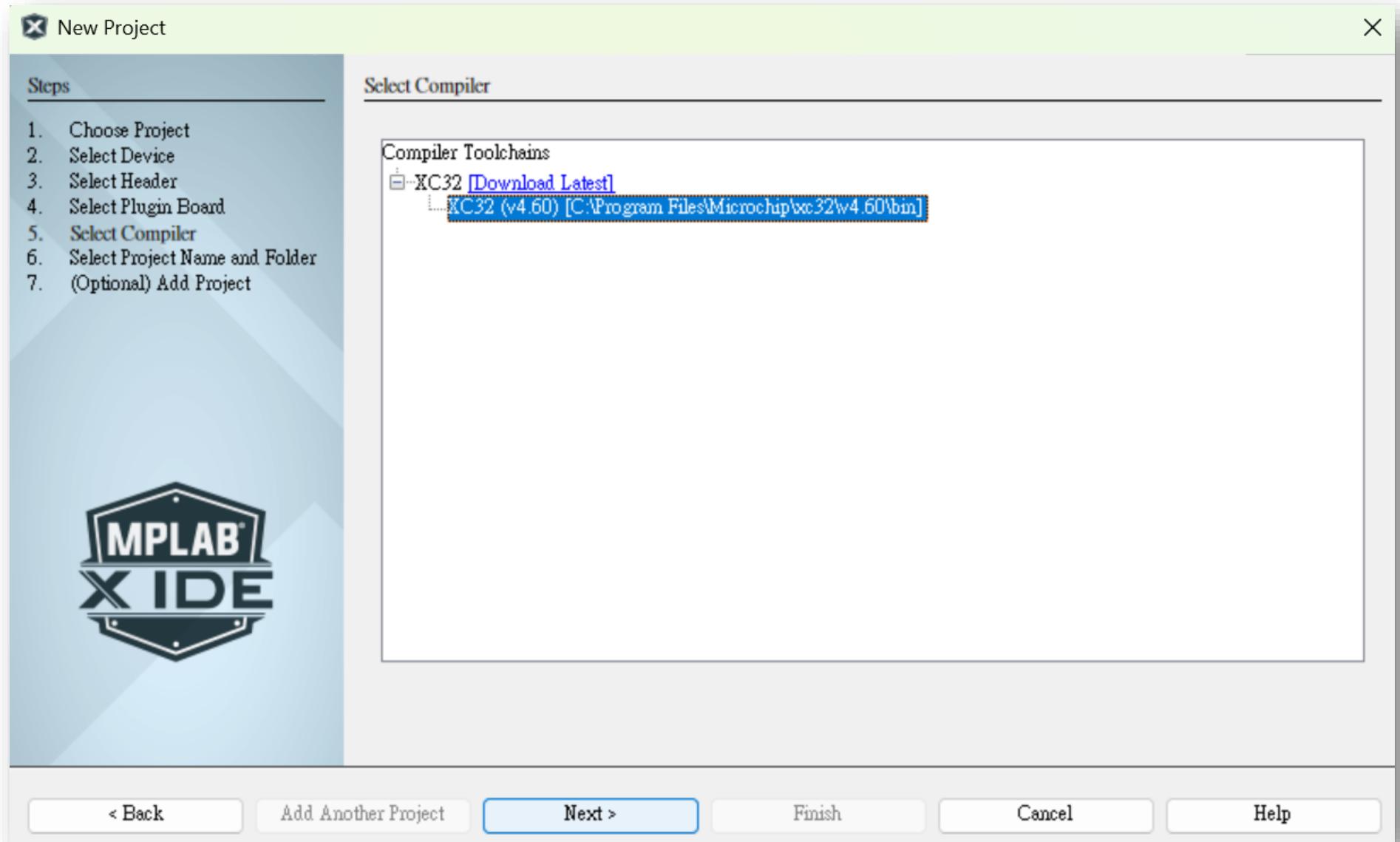
Lab1 – MPLAB® X IDE project creation & Systick

Select Device as PIC32CM3204GV00048 – You can select Tool later ..



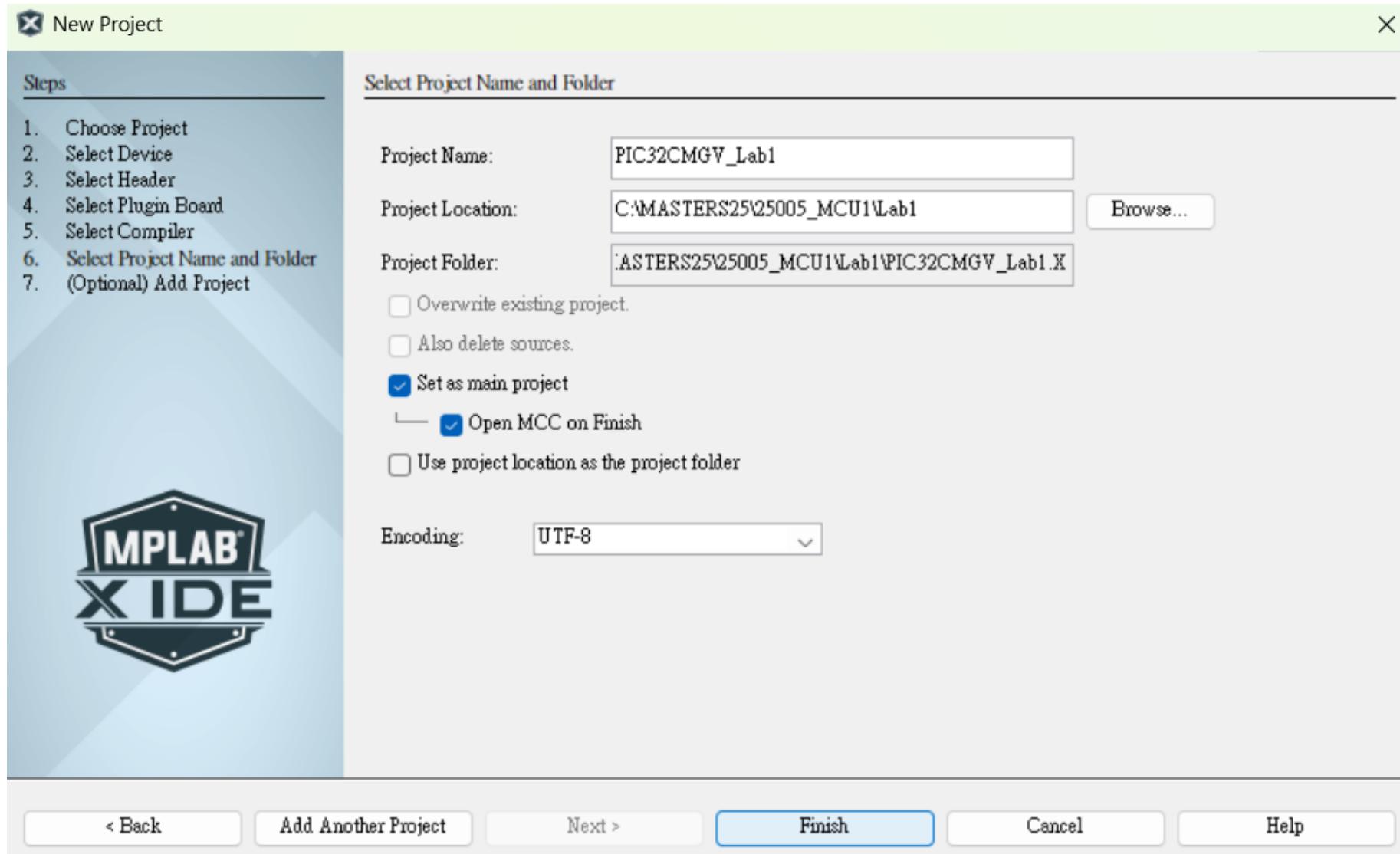
Lab1 – MPLAB® X IDE project creation & Systick

Select Compiler Toolchains – 使用 XC32 v4.60 版



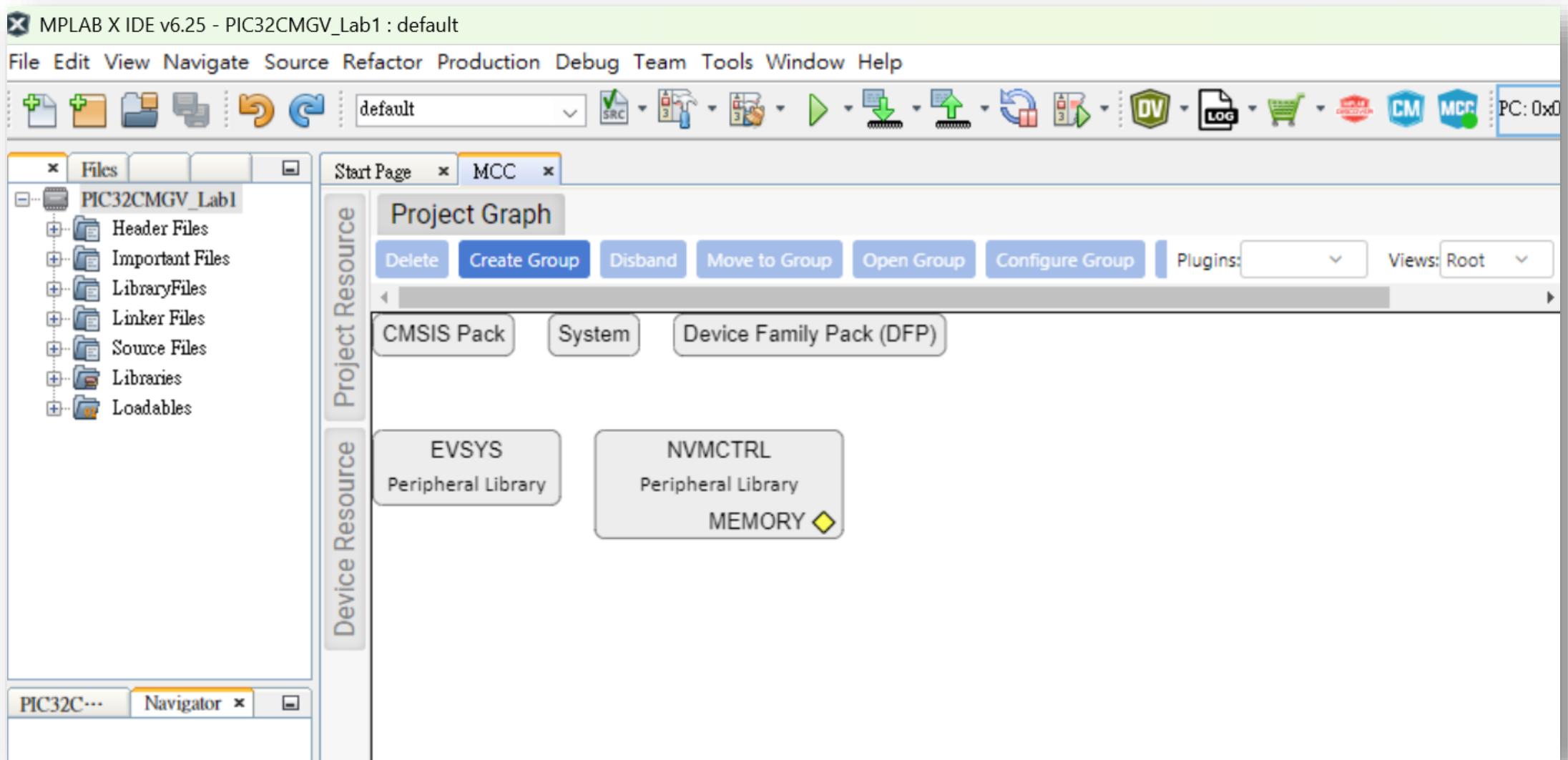
Lab1 – MPLAB® X IDE project creation & Systick

確定 Project Location & Name, 並且將編碼方式設定為 UTF-8 以便正確顯示中文



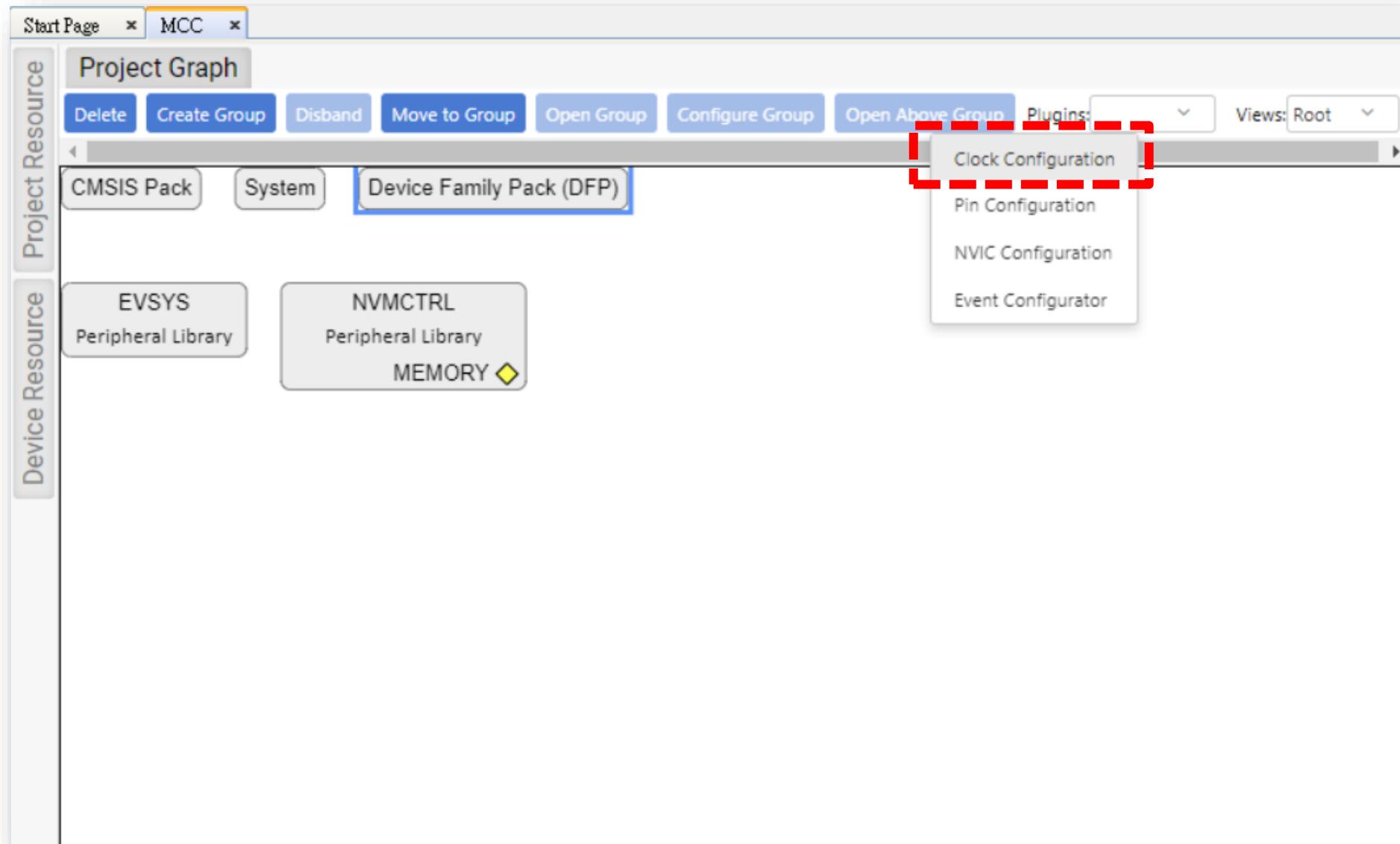
Lab1 – MPLAB® X IDE project creation & Systick

按下 Finish 之後 MCC 會被自動開啟並下載必要的軟體元件



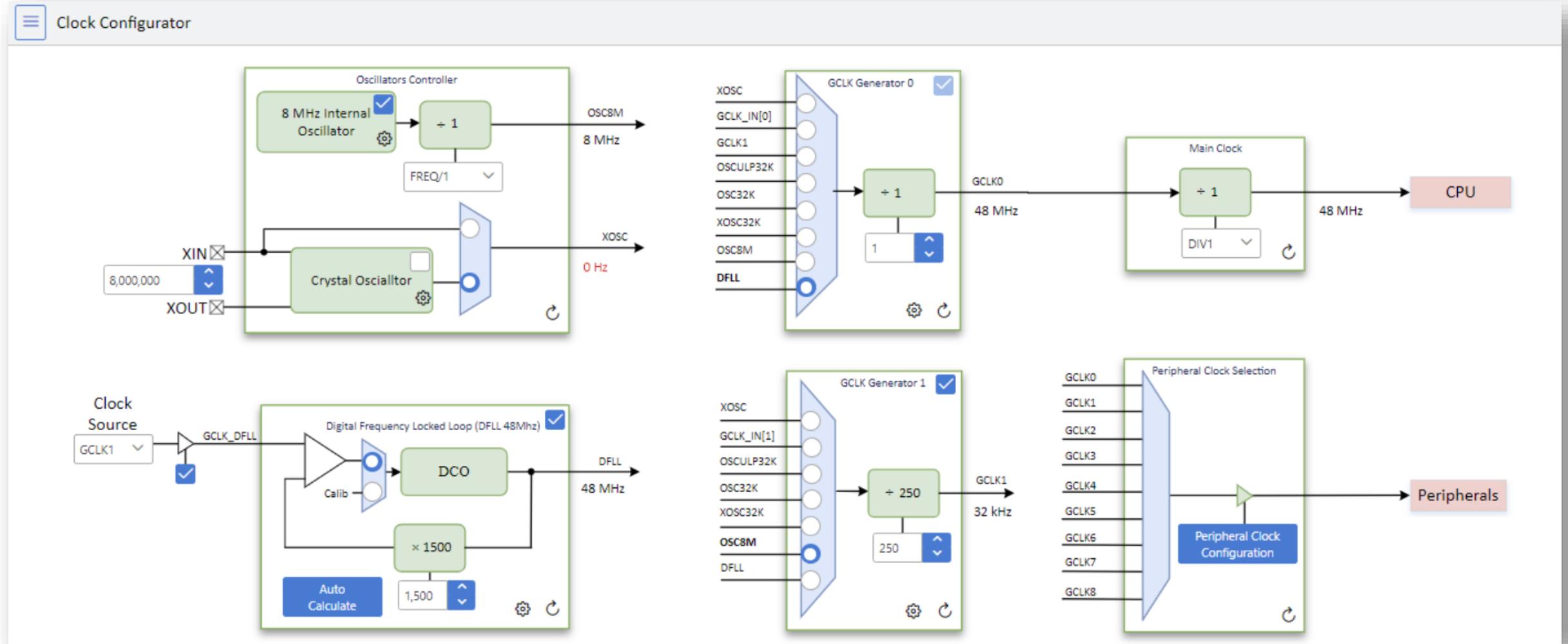
Lab1 – MPLAB® X IDE project creation & Systick

使用 “Clock Configuration” Plugin 來觀察及設定 Clock



Lab1 – MPLAB® X IDE project creation & Systick

預設的 CPU Clock 已經是 48 MHz，如有必要可議根據需求再調整



Lab1 – MPLAB® X IDE project creation & Systick

個別的周邊可以選用不同有被 Activated 的 Clock

The screenshot shows the MPLAB X IDE interface with the "Clock Configurator" window open. The window title is "Clock Configurator" and the sub-tab is "Peripheral Clock Configuration". The table lists various peripherals and their clock settings:

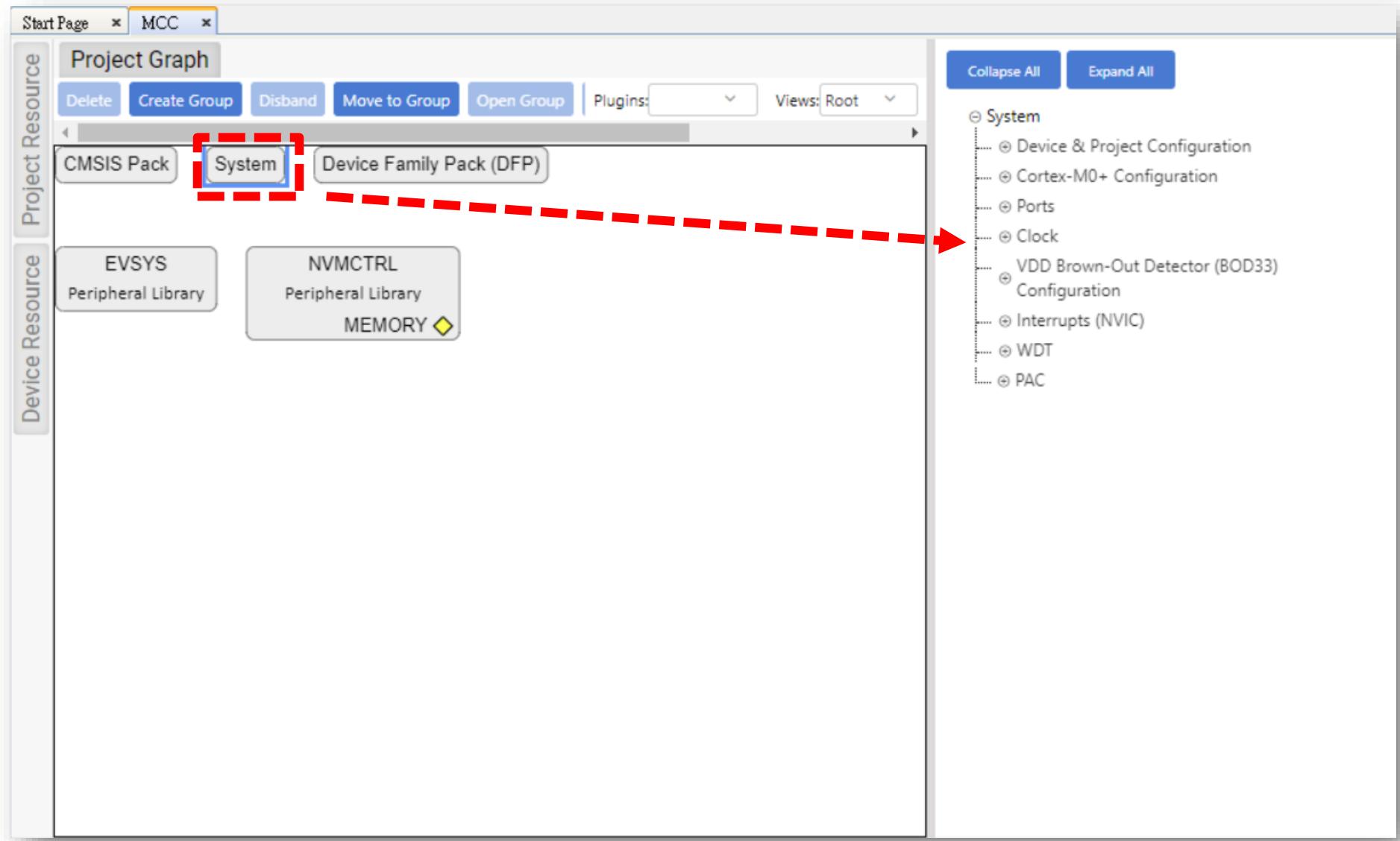
Peripheral	Enable	Source	Peripheral Clock Frequency
AC_ANA	<input type="checkbox"/>	GCLK0	--
AC_DIG	<input type="checkbox"/>	GCLK0	--
ADC	<input type="checkbox"/>	GCLK0	--
DAC	<input type="checkbox"/>	GCLK0	--
EIC	<input type="checkbox"/>	GCLK0	--
EVSYS_0	<input type="checkbox"/>	GCLK0	--
EVSYS_1	<input type="checkbox"/>	GCLK0	--
EVSYS_2	<input type="checkbox"/>	GCLK0	--
EVSYS_3	<input type="checkbox"/>	GCLK0	--
EVSYS_4	<input type="checkbox"/>	GCLK0	--
EVSYS_5	<input type="checkbox"/>	GCLK0	--

A red arrow points from the "Peripheral Clock Selection" section of the configuration dialog to the "Peripheral Clock Configuration" section, indicating the connection between the two.

On the right side of the interface, there is a clock tree diagram. It starts with a "Main Clock" source, which is divided by a "DIV1" block to produce a "48 MHz" signal for the "CPU". This same "48 MHz" signal also feeds into a "Peripheral Clock Selection" block. From this block, individual clock signals are distributed to the "Peripherals" listed in the configuration table.

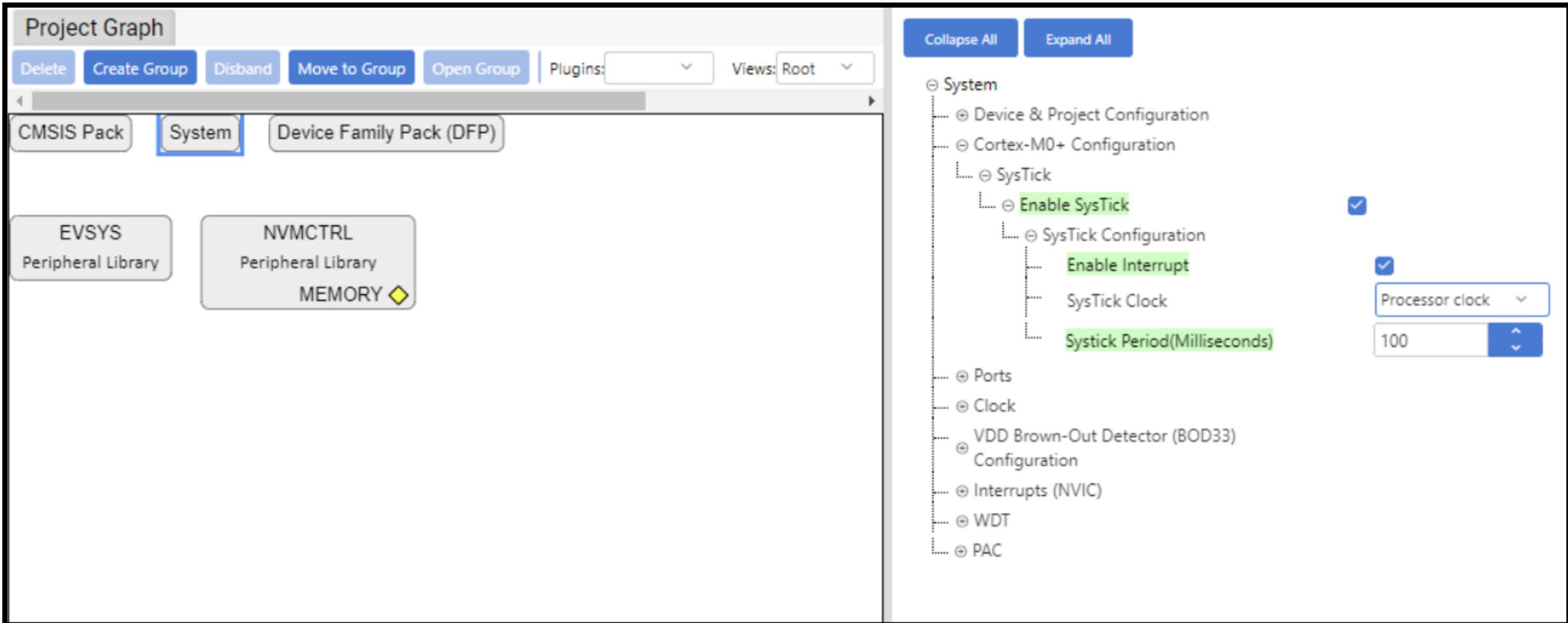
Lab1 – MPLAB® X IDE project creation & Systick

點選 System 來設定 SysTick 以及觀察其他重要選項



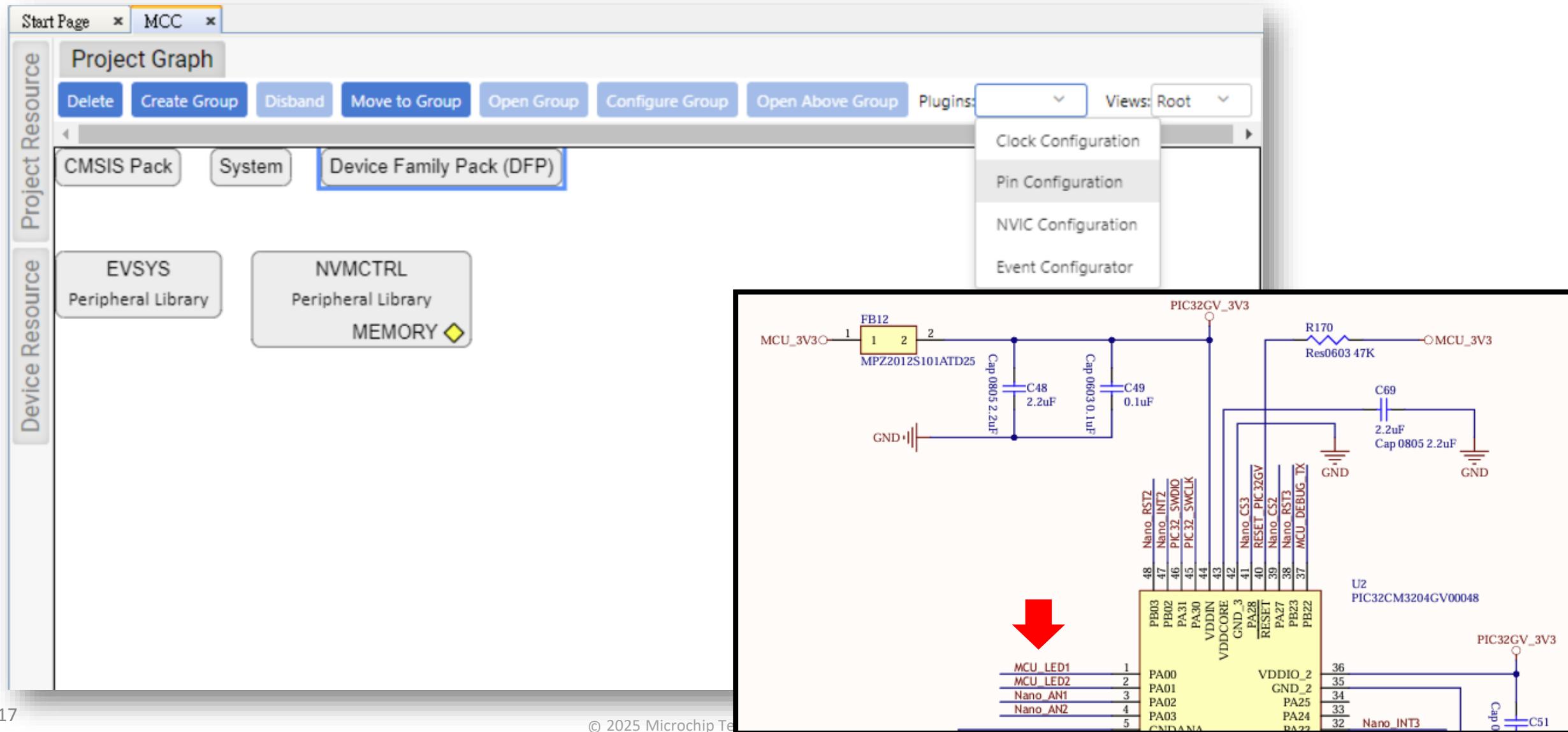
Lab1 – MPLAB® X IDE project creation & Systick

Enable SysTick 、啟用中斷並將 Systick Period 設定為 100 ms



Lab1 – MPLAB® X IDE project creation & Systick

I/O Pin 的設定：選擇 Pin Configuration 這個 Plugin (LED1 @PA00)



Lab1 – MPLAB® X IDE project creation & Systick

將 PA00 設定為可以做 Out 的 GPIO

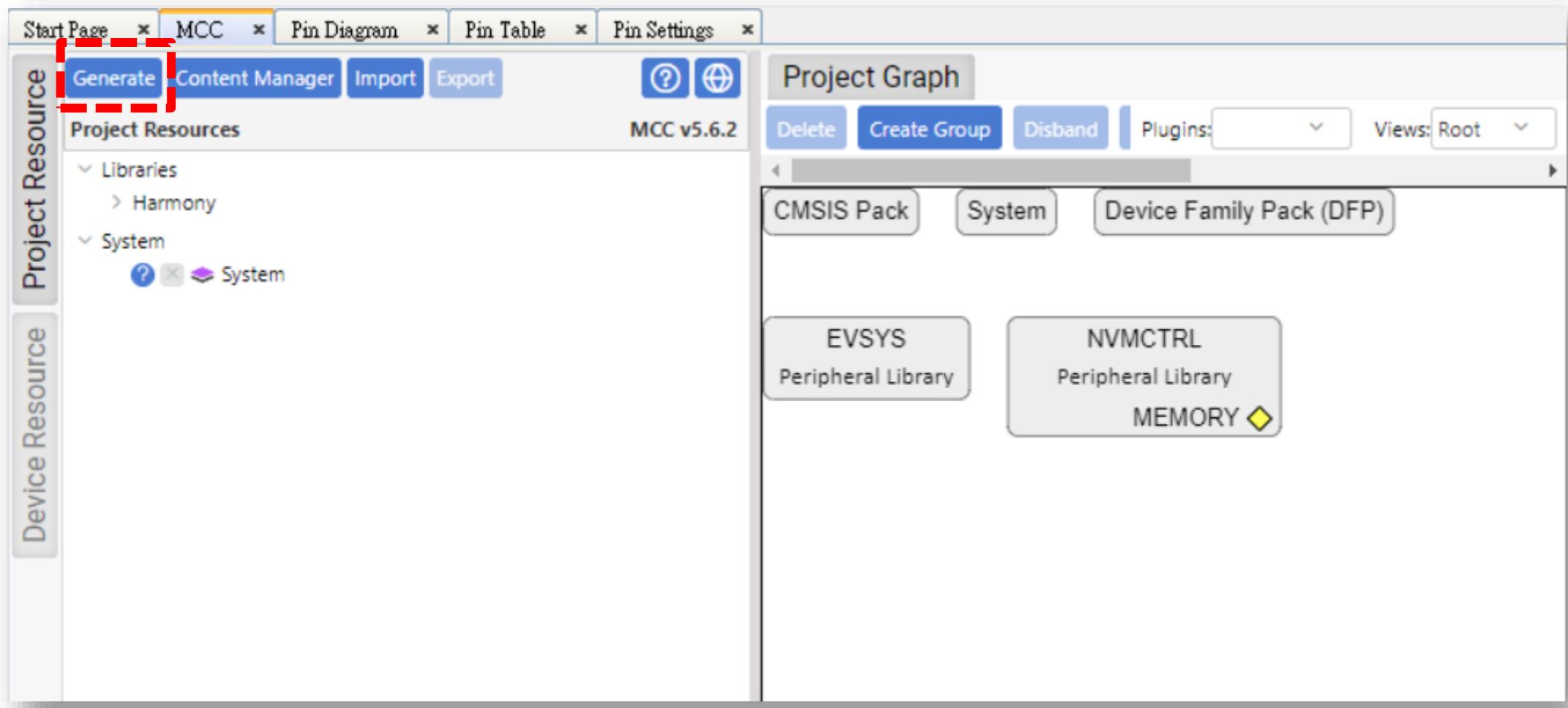
- Function
- Direction
- Custom Name (MCC 在產生 PLIB API 時會使用此名稱)

The screenshot shows the 'Pin Settings' tab in the MPLAB X IDE. The table lists pins from 1 to 7, with PA00 selected. The columns represent Pin Number, Pin ID, Custom Name, Function, Mode, Direction, Latch, Pull Up, Pull Down, and Drive Strength. PA00 is configured as a Digital output (Out) with Low as the default value for Latch, Pull Up, and Pull Down. The 'Custom Name' field contains 'LED1'. The 'Function' dropdown for PA00 shows 'GPIO' as the current selection, with other options like 'Available', 'EIC_EXTINT0', 'GPIO', 'SERCOM1_PAD0', 'SYSCTRL_XIN32', and 'TC2_WO0' listed below it.

Pin Number	Pin ID	Custom Name	Function	Mode	Direction	Latch	Pull Up	Pull Down	Drive Strength
1	PA00	LED1	GPIO	Digital	Out	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
2	PA01		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
3	PA02		EIC_EXTINT0	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
4	PA03		GPIO	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
5	GNDANA		SERCOM1_PAD0	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
6	VDDANA		SYSCTRL_XIN32	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
7	PB08		TC2_WO0	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
			Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL

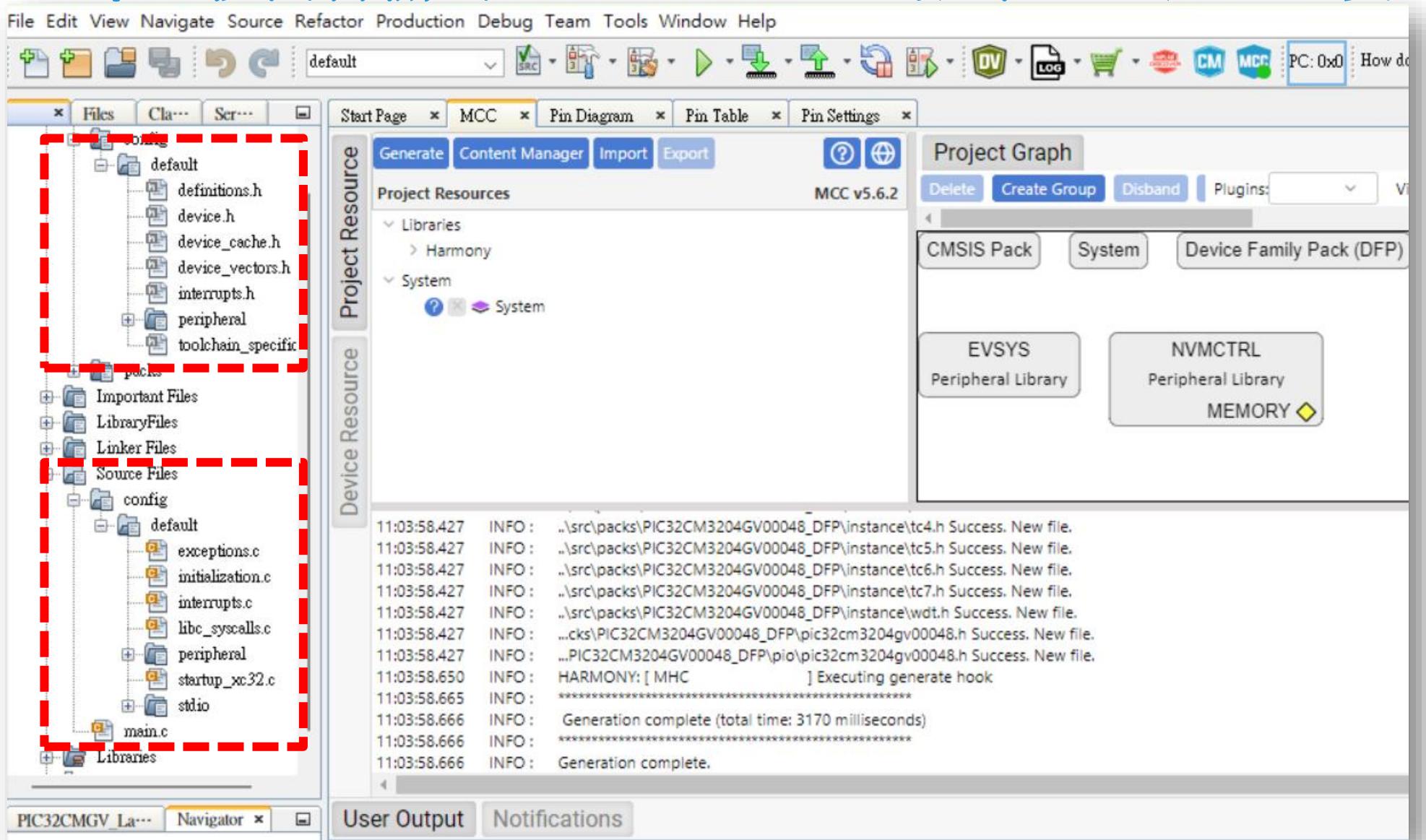
Lab1 – MPLAB® X IDE project creation & Systick

回到 MCC 頁面，點選 Project Resource 中的 **Generate**



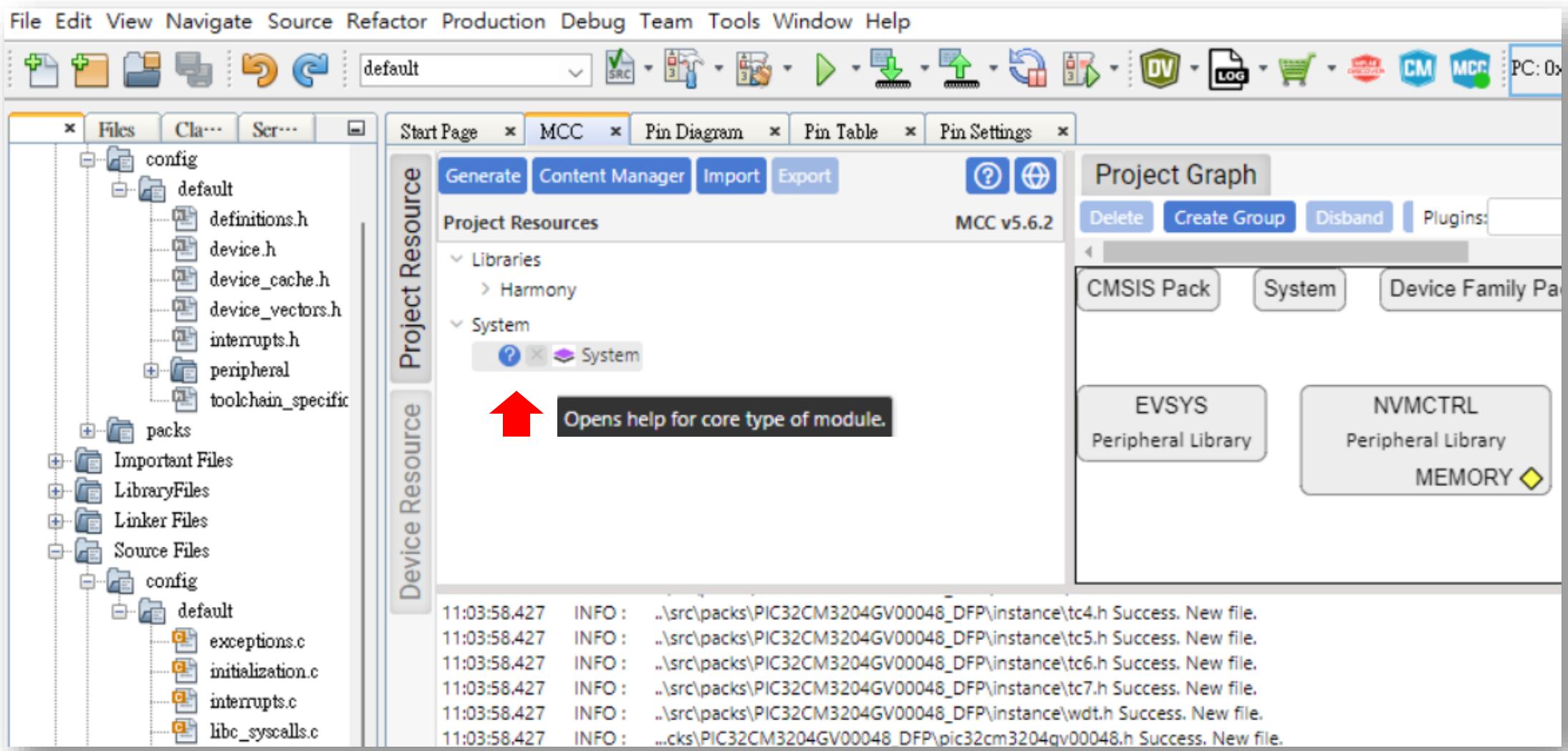
Lab1 – MPLAB® X IDE project creation & Systick

User Output 視窗會輸出 Generate Code 的過程並且產生必要的檔案



Lab1 – MPLAB® X IDE project creation & Systick

點選 System 模組左邊的 ? 即可連結到線上說明



Lab1 – MPLAB® X IDE project creation & Systick

點選 System timer 項目即可找到 SysTick 的應用說明

The screenshot shows a web browser window with the Microchip logo and the title "MPLAB® Harmony Peripheral Libraries". The URL in the address bar is "microchip.com". The page content is organized into two main sections: a sidebar on the left and a main table on the right.

Sidebar:

- Home / 2 API Documentation / 2.118 System
- > 2.109 Single-Edge Nibble Transmission (SENT)
- < > 2.110 Serial Communication Interface (SERCOM)
- > 2.111 Shutdown Controller (SHDWC)
- > 2.112 I2C SMBUS
- > 2.113 Static Memory Controller (SMC)
- > 2.114 Serial Peripheral Interface (SPI)
- > 2.115 Serial Quad Interface (SQI)
- 2.116 STDIO
- > 2.117 Supply Controller (SUPC)
- 2.118 System
- > 2.119 System timer (SysTick)
- > 2.120 Timer Counter (TC)

Main Content:

<ul style="list-style-type: none">Cache Controller (Cache)Cortex-M Cache Controller (CMCC)	Cache controller peripheral library for the target device.
<ul style="list-style-type: none">Direct Memory Access Controller (DMA)Direct Memory Access Controller (DMAC)Extensible DMA Controller (XDMAC)	DMA controller peripheral library for the target device.
<ul style="list-style-type: none">MIPS Core Timer (CORETIMER)System timer (SysTick)ARM Cortex A Generic timer (GENERIC_TIMER)	Core specific timer peripheral library for the target device.
<ul style="list-style-type: none">Watchdog Timer (WDT)Dual Watchdog Timer (DWDT)Dead Man Timer (DMT)	Watchdog timer peripheral library for the target device.

Lab1 – MPLAB® X IDE project creation & Systick

Lab1 希望使用中斷，所以參考 callback method

Callback method

This example demonstrates how to use SysTick to generate periodic callback.

```
/* This function is called after Timer expires */
void SYSTICK_EventHandler(uintptr_t context)
{
    /* Toggle LED */
    LED_Toggle();
}

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

    /* Register Callback */
    SYSTICK_TimerCallbackSet(SYSTICK_EventHandler, (uintptr_t) NULL);

    /* Start the Timer */
    SYSTICK_TimerStart();
}
```

Copy 

Lab1 – MPLAB® X IDE project creation & Systick

修改後的 main.c -> 請注意 LED1 的 Custom Name 已被用上

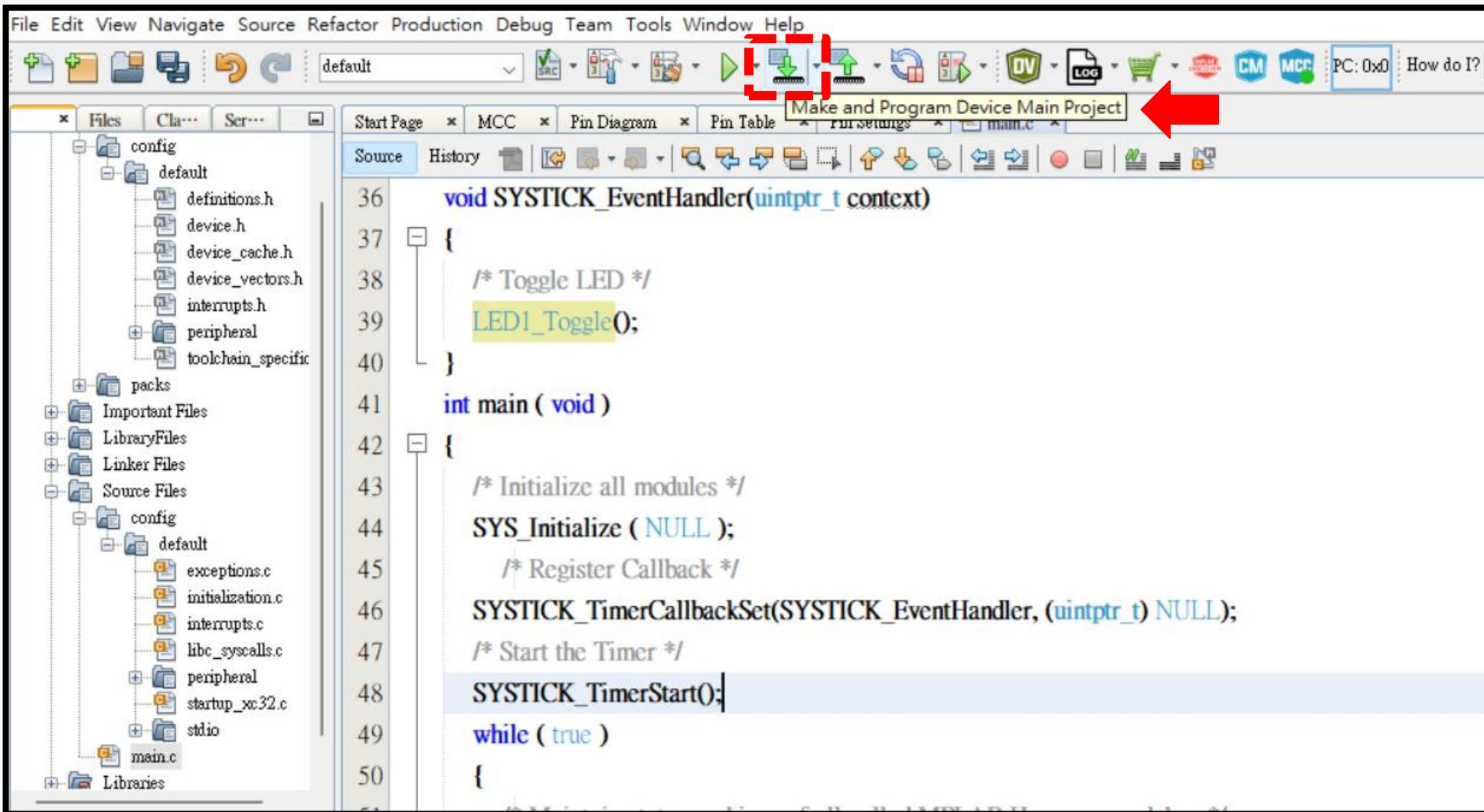
The screenshot shows the MPLAB X IDE interface. The left pane displays the project structure under 'Source Files'. The 'main.c' file is open in the right pane. The code implements a Systick interrupt handler and the main loop of the application.

```
void SYSTICK_EventHandler(uintptr_t context)
{
    /* Toggle LED */
    LED1_Toggle0;
}

int main ( void )
{
    /* Initialize all modules */
    SYS_Initialize ( NULL );
    /* Register Callback */
    SYSTICK_TimerCallbackSet(SYSTICK_EventHandler, (uintptr_t) NULL);
    /* Start the Timer */
    SYSTICK_TimerStart();
    while ( true )
    {
        /* Maintain state machines of all polled MPLAB Harmony modules. */
        SYS_Tasks ();
    }
}
```

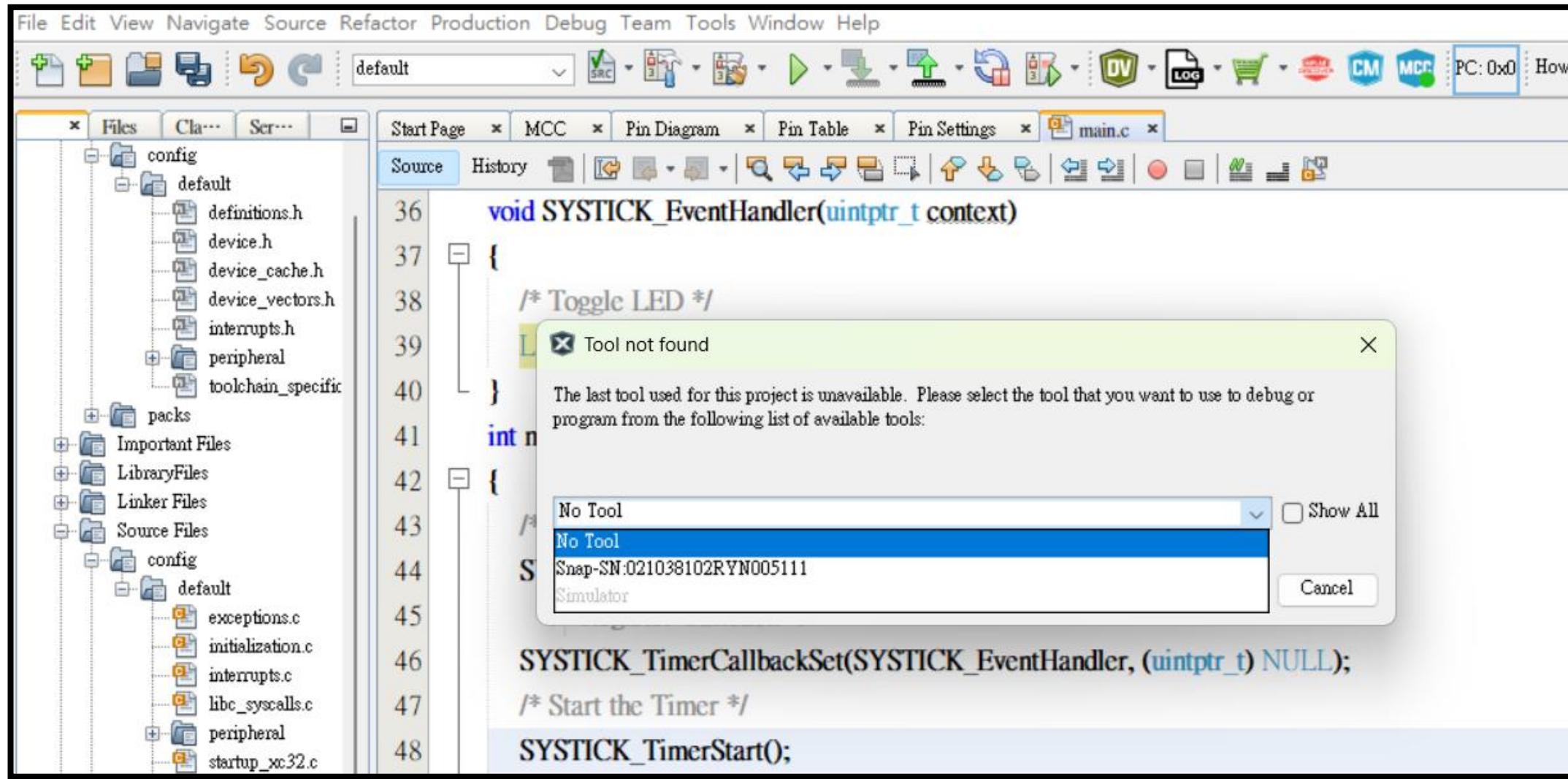
Lab1 – MPLAB® X IDE project creation & Systick

點選  Make and Program Device Main Project 來進行燒錄作業



Lab1 – MPLAB® X IDE project creation & Systick

如果連接的 Tool 之前未被辨識過，MPLAB XIDE 會讓適用者確認並選擇

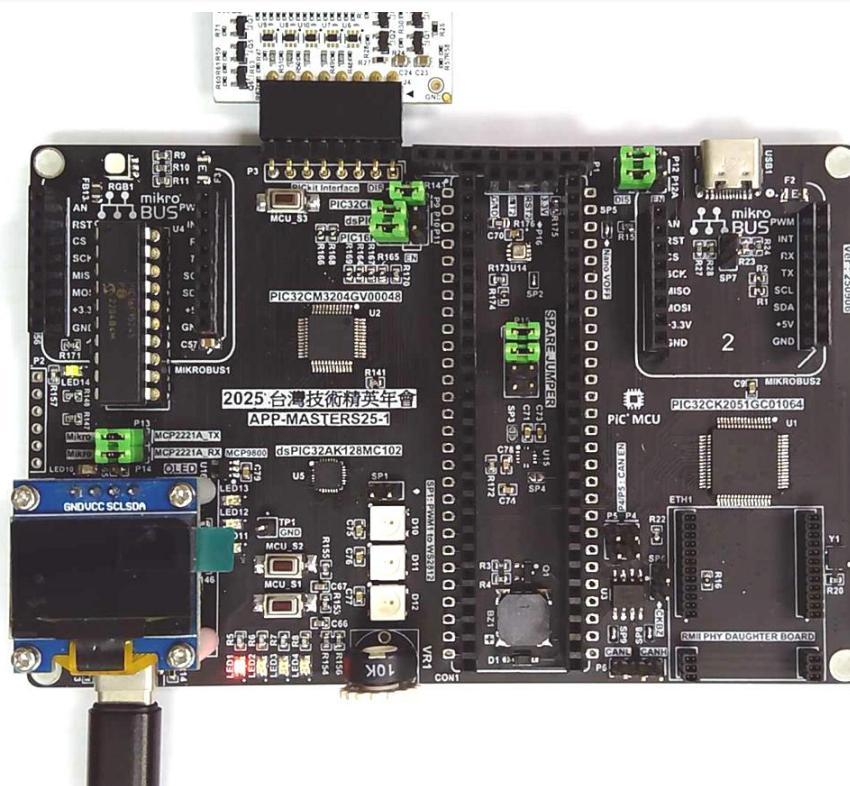


Lab1 – MPLAB® X IDE project creation & Systick

燒錄完成後的執行結果 – LED1 每 100 ms 轉態一次

```
Output ×
Scripting × MPLAB® Code Configurator × PIC32CMGV_Lab1 (Build, Load, ...) × Snap × Snap-PIC32CMGV_Lab1 ×
The following memory area(s) will be programmed:
program memory: start address = 0x0, end address = 0x4ff
configuration memory
User Id Memory

Due to the large memory ranges on this device, only the areas of memory that have been loaded with code (via the build process
Programming/Verify complete
```



Lab 1 Summary



Successfully created a MPLAB® X IDE project using MCC Harmony v3



Check and Config Clock – Make sure CPU runs at 48 Mhz
Config SysTick and learn how to implement “Interrupt”
Config I/O Pins and use Custom Name for easy access



Program the device by using Microchip Tool
Viewed the main device and component header files



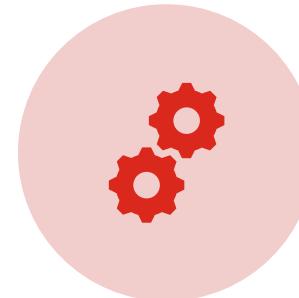
Lab 2 – Interface to the world

ADC read and UART communication

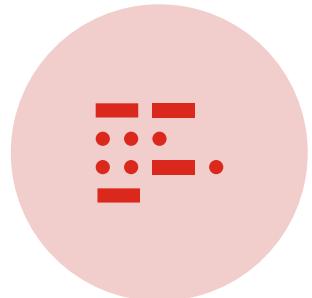
Lab 2 Objectives



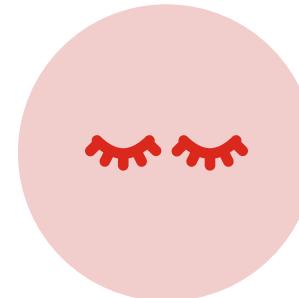
Continue setup from
Lab 1



Set up UART
(115200,N,8,1) & ADC

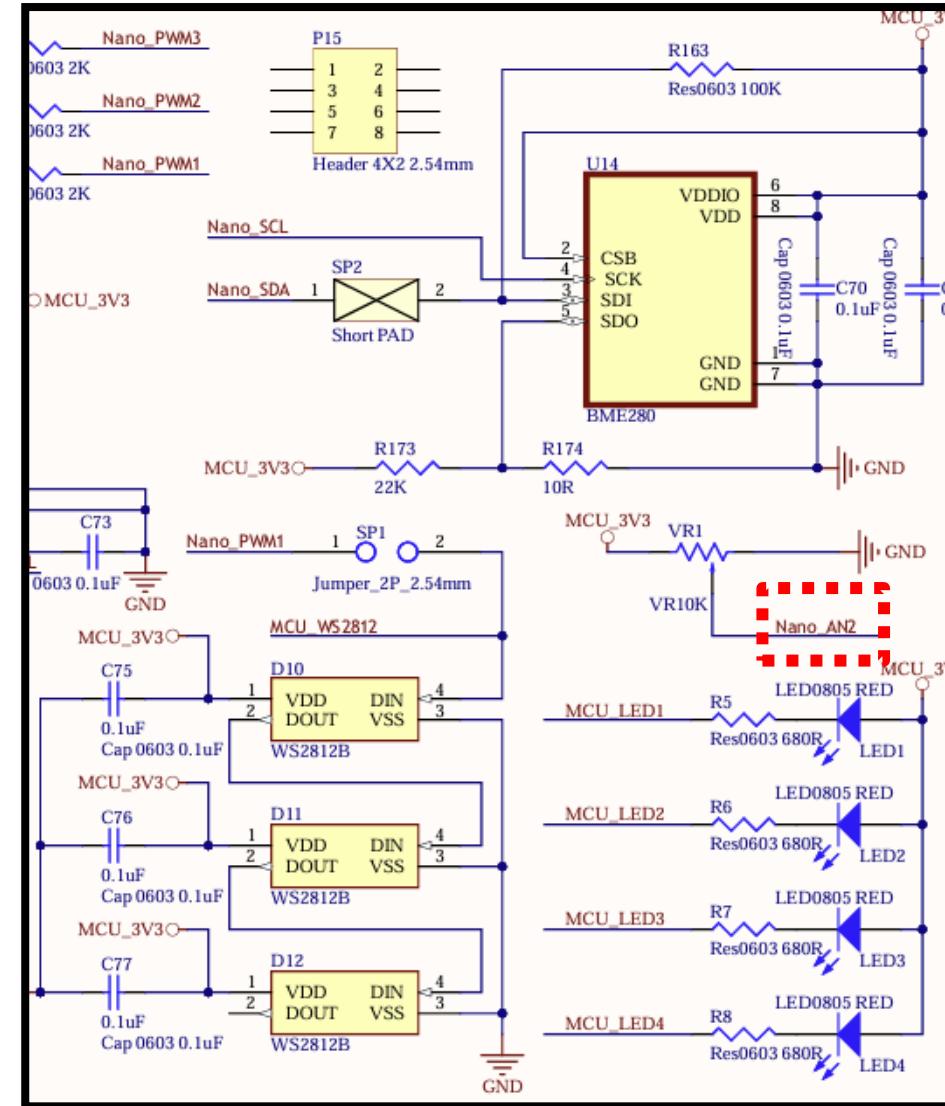
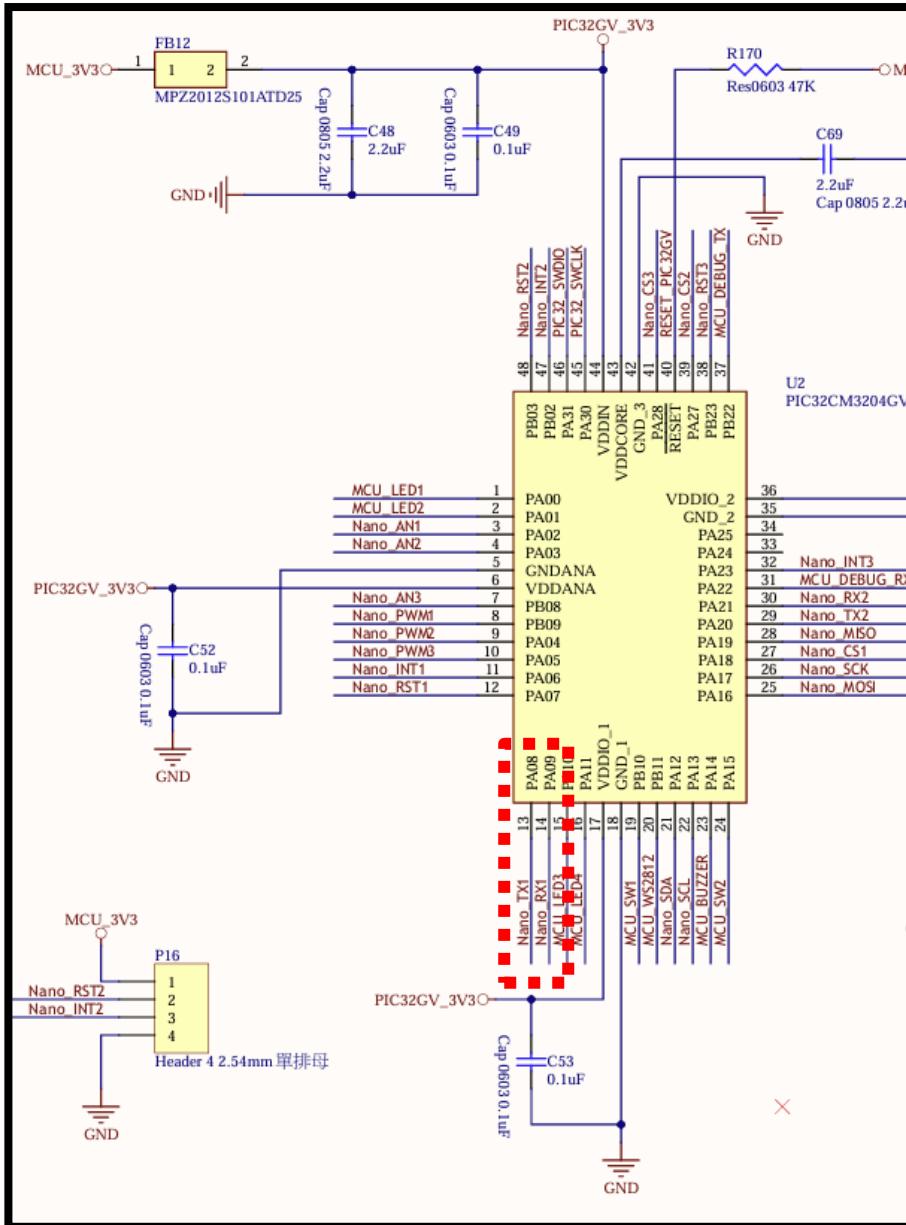


Review the generated
PLIBs



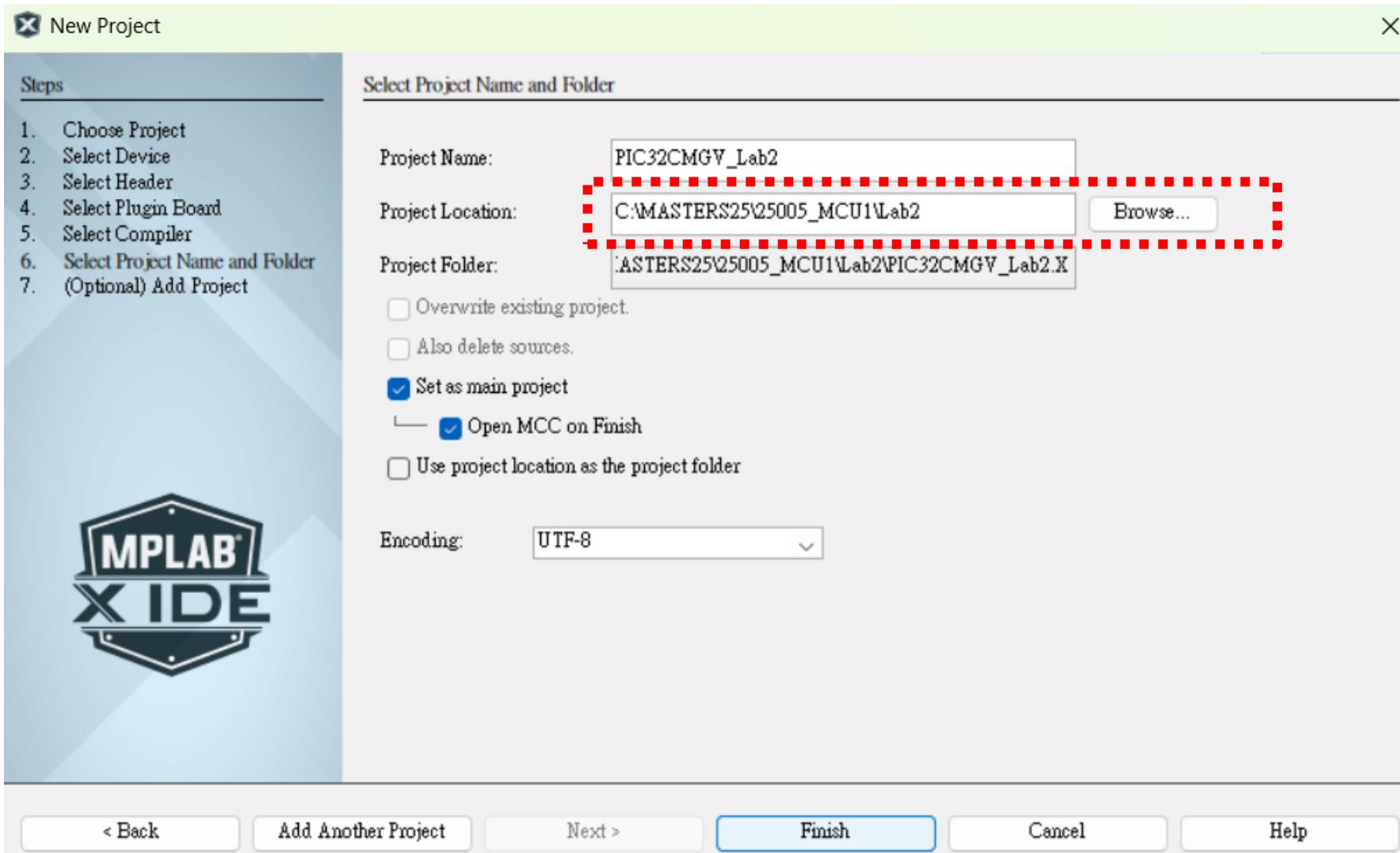
Send VR(ADC) result
through UART to
MPLAB® Data Visualizer

Lab2 – Interface to the World -> 使用到的接腳



Lab2 – Interface to the World

建立 Lab2 保留 Lab1 的部分, 讓 SysTick 以 200ms 的時間來中斷 MCU



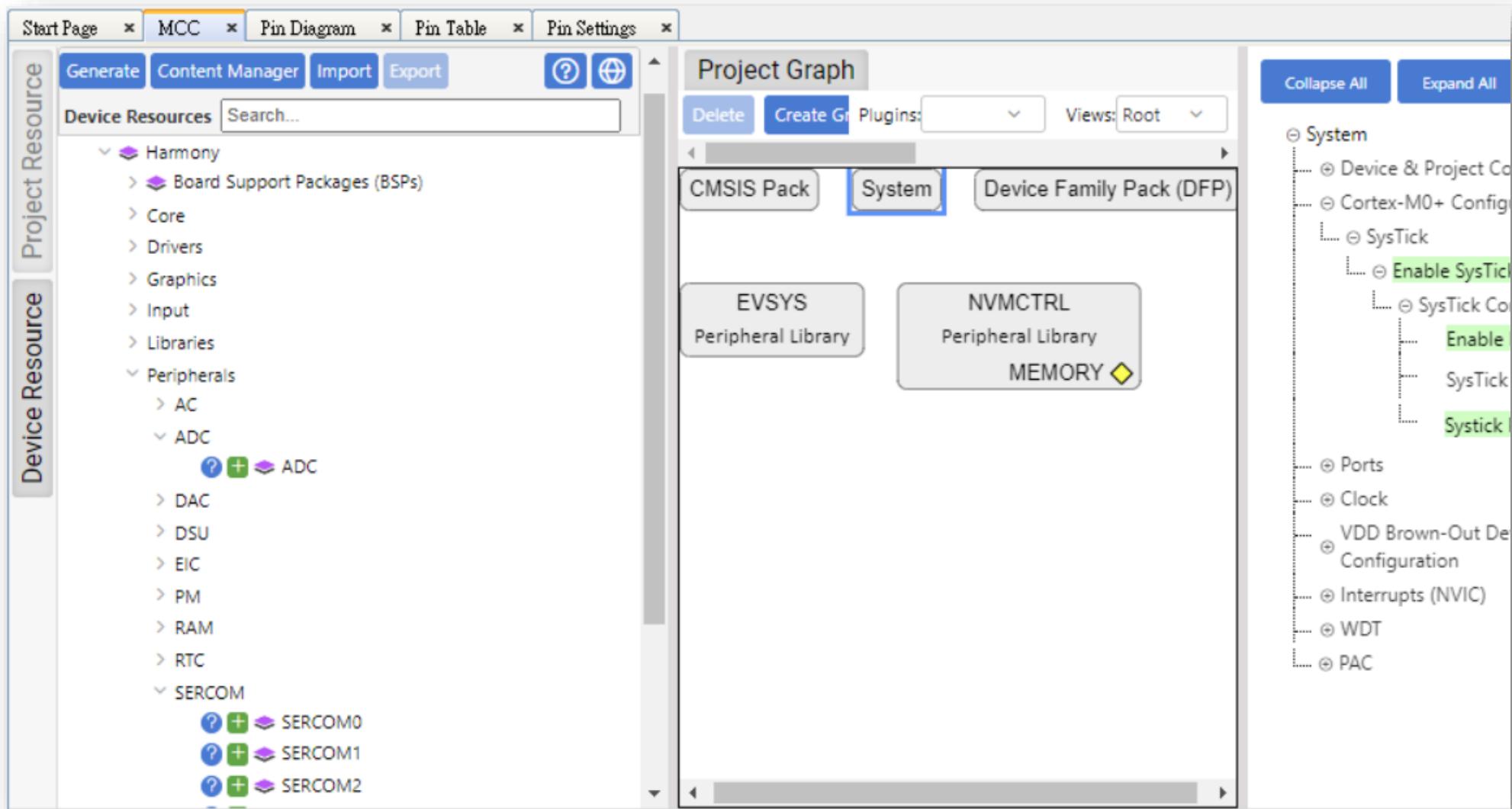
Lab2 – Interface to the World

增加對 VR (PA03) 以及 UART (PA08&PA09) Pin 腳的設定

Pin Settings									
Order:	Pins	Table View	<input checked="" type="checkbox"/> Easy View						
Pin Number	Pin ID	Custom Name	Function	Mode	Direction	Latch	Pull Up	Pull Down	Drive Strength
1	PA00	LED1	GPIO	Digital	Out	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
2	PA01		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
3	PA02		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
4	PA03	VR1	ADC_AIN1	Analog	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
5	GNDANA			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
6	VDDANA			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
7	PB08		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
8	PB09		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
9	PA04		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
10	PA05		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
11	PA06		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
12	PA07		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
13	PA08		SERCOM0_PAD0	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
14	PA09		SERCOM2_PAD1	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
15	PA10		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
16	PA11		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
17	VDDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
18	GNDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
19	PB10		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL

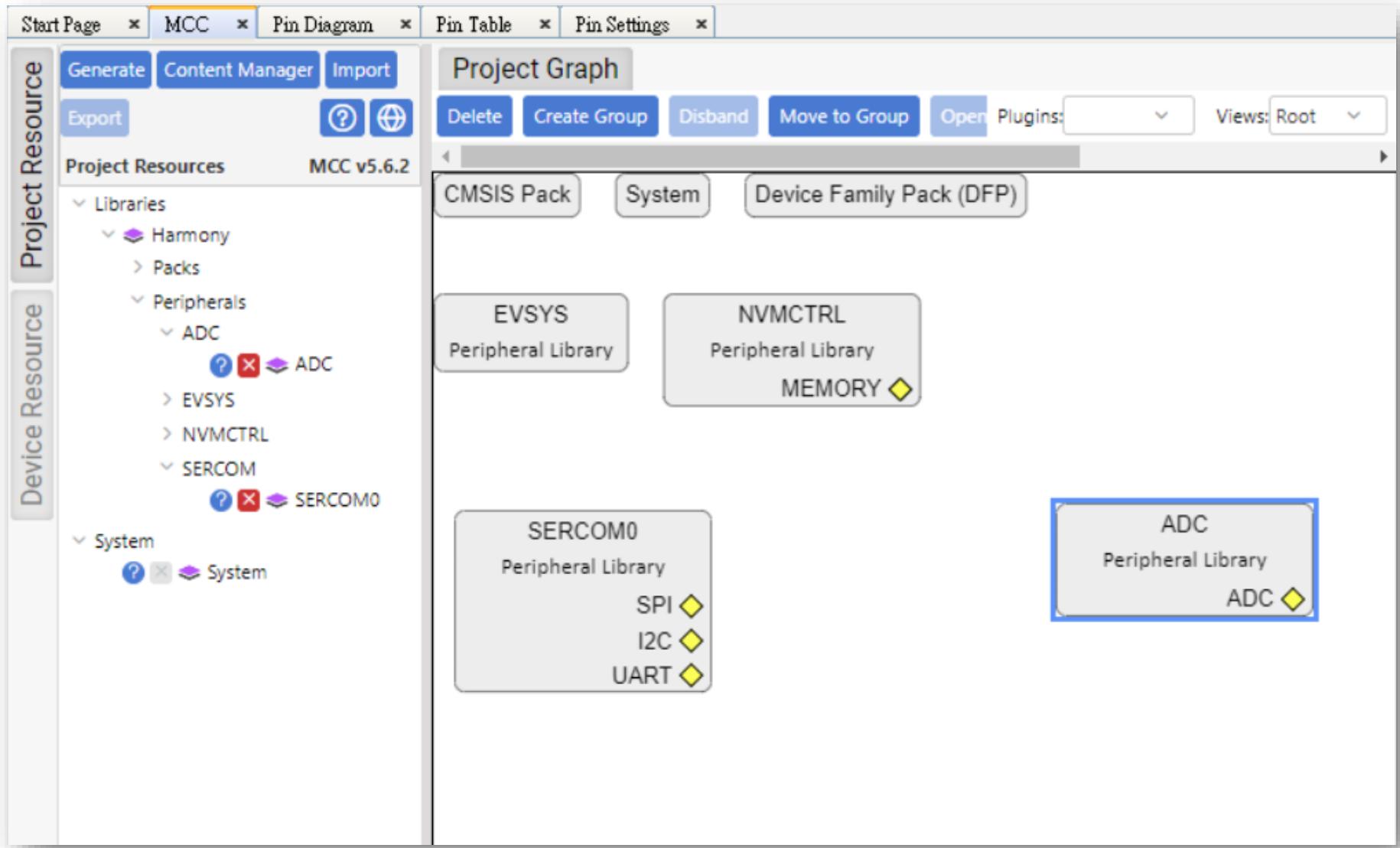
Lab2 – Interface to the World

在 Device Resource 中加入 ADC & SERCOM0



Lab2 – Interface to the World

在 Project Resource 中以及 MCC 視窗可以看到被加入的物件



Lab2 – Interface to the World

ADC 的規劃 – 針對單一 channel 的規劃 (VR的上下限為 0~3.3V)

The screenshot shows the Microchip Studio interface with the Project Graph on the left and the ADC configuration on the right.

Project Graph:

- Buttons: Delete, Create Group, Disband, Move to Group, Open, Plugins: [dropdown], Views: Root [dropdown]
- Groups:
 - CMSIS Pack
 - System
 - Device Family Pack (DFP)
 - EVSYS Peripheral Library
 - NVMCTRL Peripheral Library
 - MEMORY ◀
 - SERCOM0 Peripheral Library
 - SPI ◀
 - I2C ◀
 - UART ◀
 - ADC Peripheral Library (highlighted with a blue border)

ADC Configuration:

- ADC** (Collapsible section)
 - Select Prescaler: Peripheral clock divided by 32 ▾
Value: 4
 - Select Sample Length (half ADC clock cycles)
**** Conversion Time is 6.66666666667 uS ****
 - Select Gain: 1/2x ▾
 - Select Reference: 1/2 VDDANA (only for VDDANA > 2.0V)
 - Select Conversion Trigger: SW Trigger ▾
- Channel Configuration** (Collapsible section)
 - Select Positive Input: ADC AIN1 Pin ▾
 - Select Negative Input: Internal ground ▾
 - Number of inputs to scan: 0
- Result Configuration** (Collapsible section)
 - Select Result Resolution: 12-bit result ▾
 - Left Aligned Result:
 - Enable Result Ready Interrupt:
 - Enable Result Ready Event:

Lab2 – Interface to the World

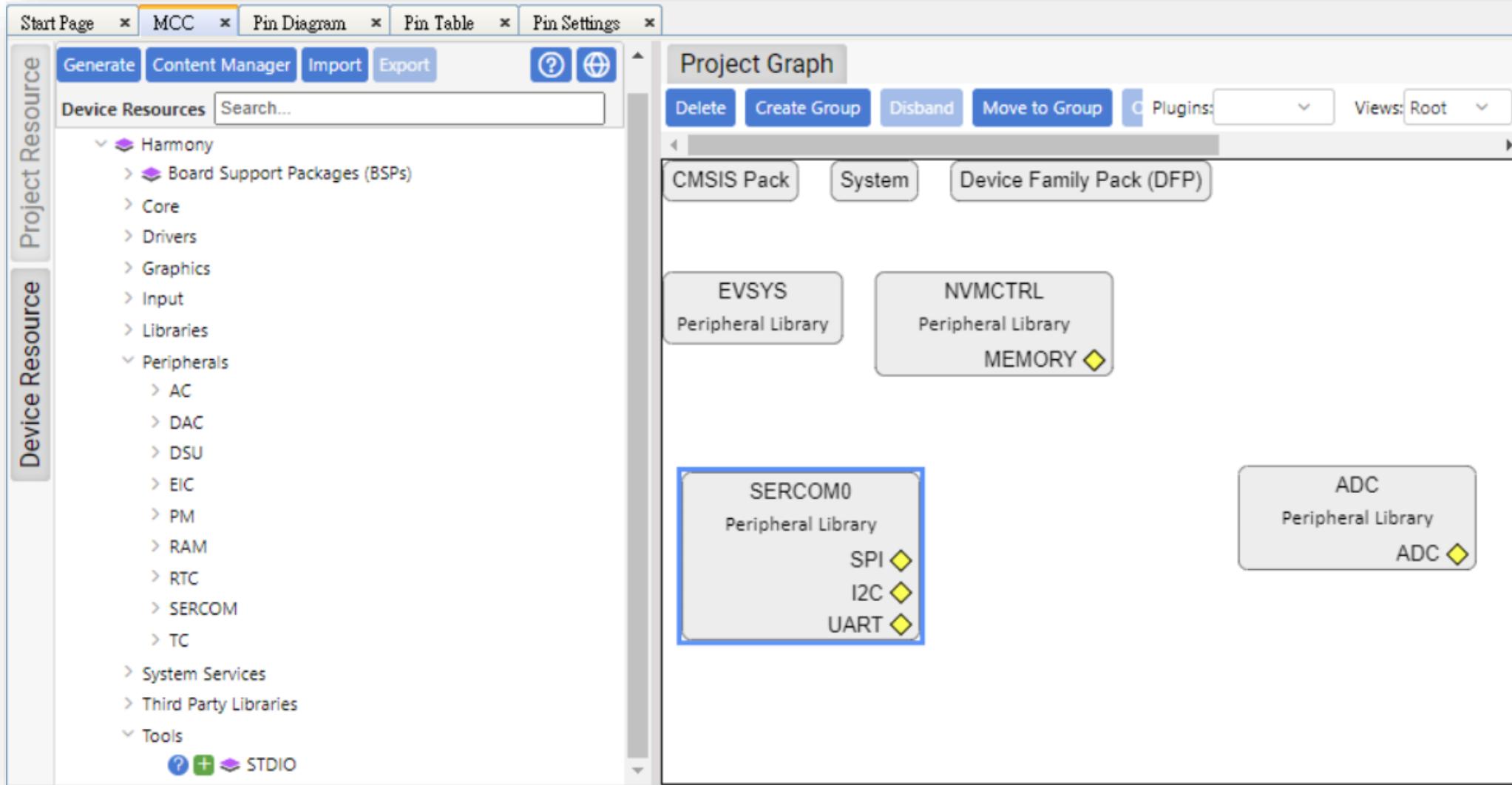
UART 的規劃

The screenshot shows the Microchip Studio Project Graph interface. On the left, the 'Project Graph' sidebar includes buttons for 'Delete', 'Create Group', 'Disband', 'Move to Group', 'Open', 'Plugins', 'Views', and 'Root'. Below these are buttons for 'CMSIS Pack', 'System', and 'Device Family Pack (DFP)'. The main workspace contains several peripheral library components: 'EVSYS Peripheral Library', 'NVMCTRL Peripheral Library', 'MEMORY ◆', 'SERCOM0 Peripheral Library' (which is selected and highlighted with a blue border), 'ADC Peripheral Library', and 'ADC ◆'. The 'SERCOM0 Peripheral Library' component is expanded to show its internal structure. On the right, the configuration pane for 'SERCOM0' is displayed, featuring 'Collapse All' and 'Expand All' buttons. The 'Select SERCOM Operation Mode' section is set to 'USART with internal Clock' and 'Non-blocking mode'. Other configuration options include:

- Operating Mode: Selected
- Receive Enable: Selected
- Transmit Enable: Selected
- Frame Format: USART frame
- Baud Rate in Hz: 115,200
- Parity Mode: No Parity
- Character Size: 8 Bits
- Stop Bit Mode: One Stop Bit
- Start-of-Frame Detection: Enabled (checkbox checked)
- Receive Pinout: SERCOM PAD[1] is used for data reception
- Transmit Pinout: PAD[0] = TxD; PAD[1] = XCK
- Enable Run in Standby: Enabled (checkbox checked)

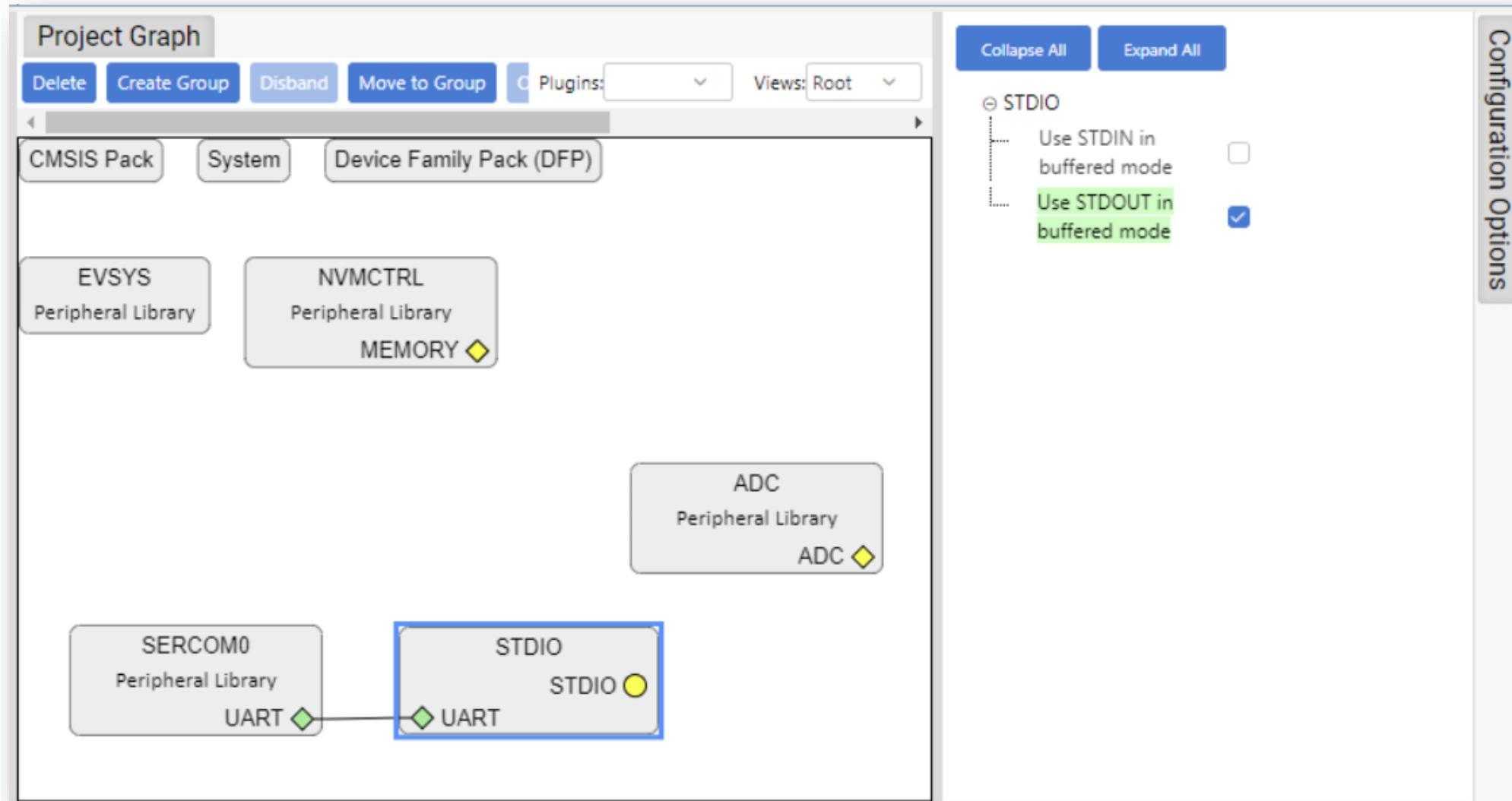
Lab2 – Interface to the World

將 STDIO 加入以便於將 UART 的資料導向 stdout



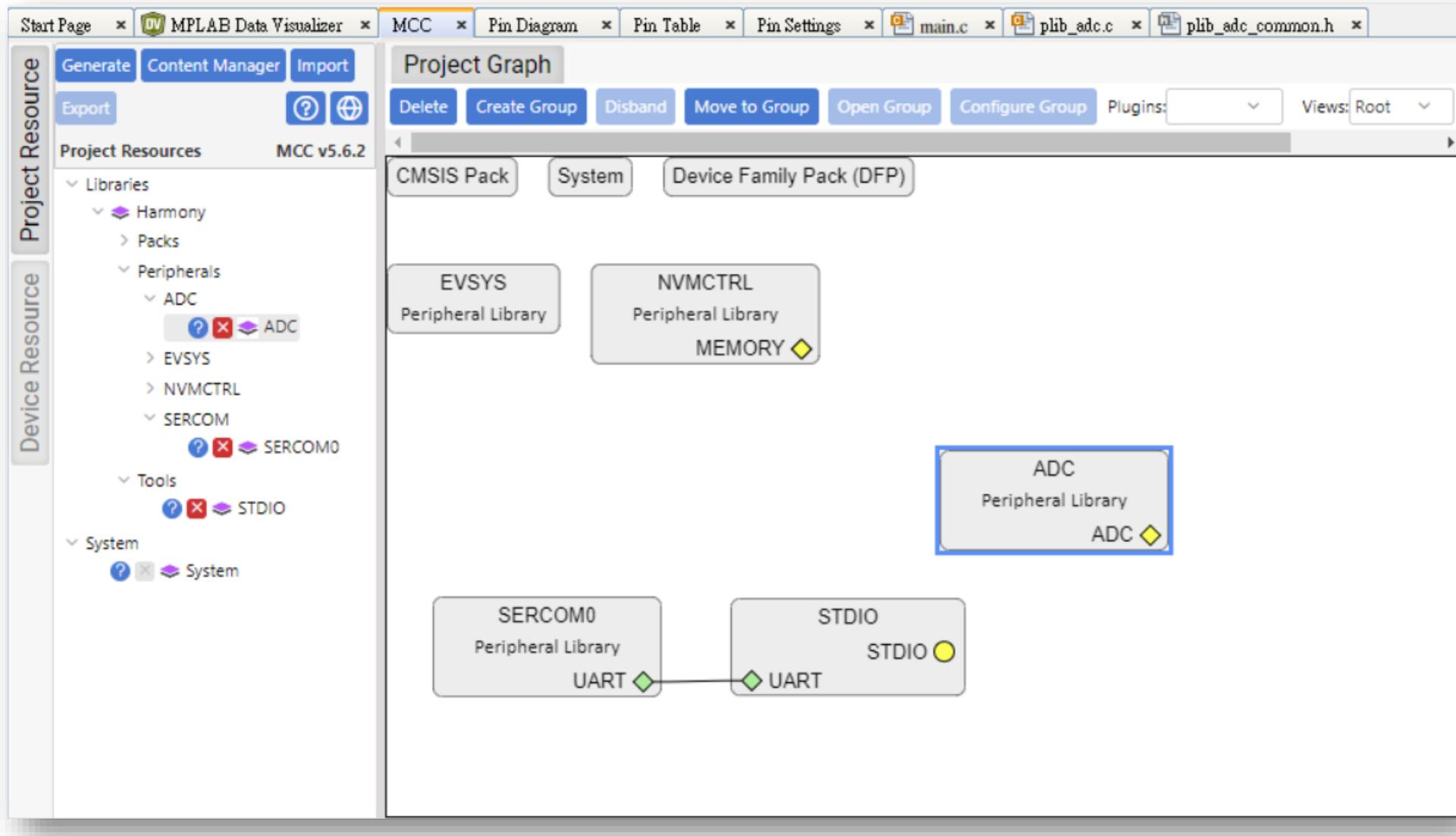
Lab2 – Interface to the World

將 SERCOM0 以滑鼠與 STUDIO 做連結即可使用 printf 來輸出資料



Lab2 – Interface to the World

點選 ADC module 左邊的問號    ADC 一樣可以連結到線上的說明文件



Lab2 – Interface to the World

ADC Module 的線上說明文件



MPLAB® Harmony Peripheral Libraries

[Home](#) / [2 API Documentation](#) / [2.3 Analog Digital Converter \(ADC\)](#)

1 MPLAB® Harmony Peripheral Libraries

- < [2 API Documentation](#)
 - > [2.1 Analog Comparators \(AC\)](#)
 - > [2.2 Analog Comparator Controller \(ACC\)](#)
 - ✓ [2.3 Analog Digital Converter \(ADC\)](#)
 - [2.3.1 ADCx_CallbackRegister Function](#)
 - [2.3.2 ADCx_ChannelSelect Function](#)
 - [2.3.3 ADCx_ComparisonWindowSet Function](#)
 - [2.3.4 ADCx_ConversionResultGet Function](#)
 - [2.3.5 ADCx_ConversionSequencelsFinished Function](#)
 - [2.3.6 ADCx_ConversionStart Function](#)
 - [2.3.7 ADCx_ConversionStatusGet Function](#)
 - [2.3.8 ADCx_Disable Function](#)
 - [2.3.9 ADCx_Enable Function](#)
 - [2.3.10 ADCx_Initialize Function](#)
 - [2.3.11 ADCx_InterruptsClear Function](#)
 - [2.3.12 ADCx_InterruptsDisable Function](#)
 - [2.3.13 ADCx_InterruptsEnable Function](#)

2.3 Analog Digital Converter (ADC)

This Plib implements software abstraction for ADC Peripheral.

Library Interface

Analog Digital Converter peripheral library provides the following interfaces:

Functions

Name	Description
ADCx_Initialize	Initializes ADC peripheral
ADCx_Enable	Enable (turn ON) ADC module
ADCx_Disable	Disable ADC module
ADC_SamplingStart	Starts the sampling
ADCx_ChannelSelect	Selects ADC input channel
ADCx_ConversionStart	Starts the ADC conversion of all the enabled channels with the

Lab2 – Interface to the World

ADCx_ConversionResultGet Function 的使用範例

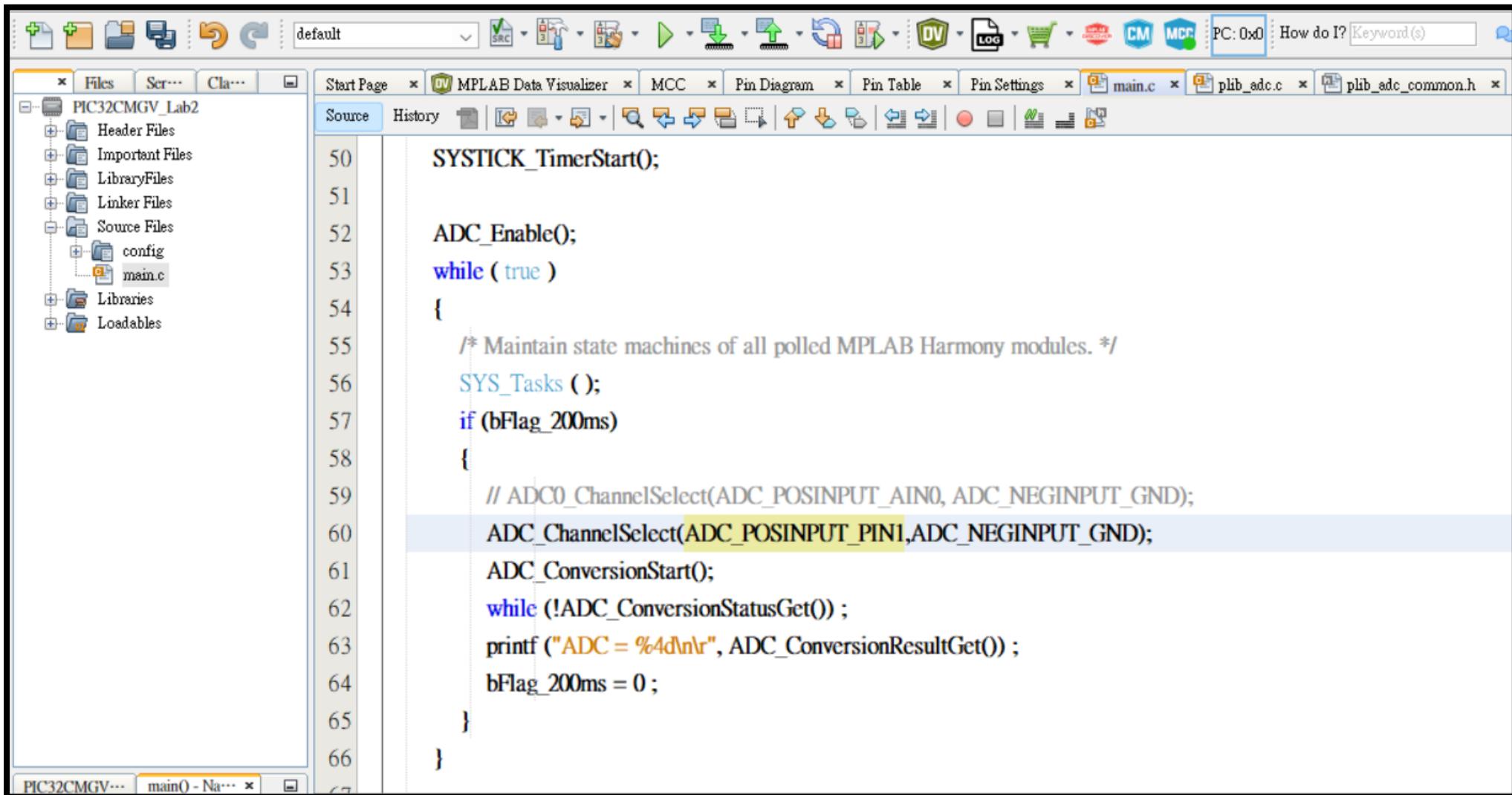
- 因為 PIC32CM3204 GV 系列只有一個 ADC，所以產生的 API 名稱為 **ADC_ConversionResultGet()**

Example

```
uint16_t adcResult = 0;  
ADC0_Initialize();  
ADC0_ConversionStart();  
while(!ADC0_ConversionStatusGet());  
adcResult = ADC0_ConversionResultGet();
```

Lab2 – Interface to the World

修改後的 main.c 主體，注意 bFlag_200ms 由 SysTick 的中斷設定

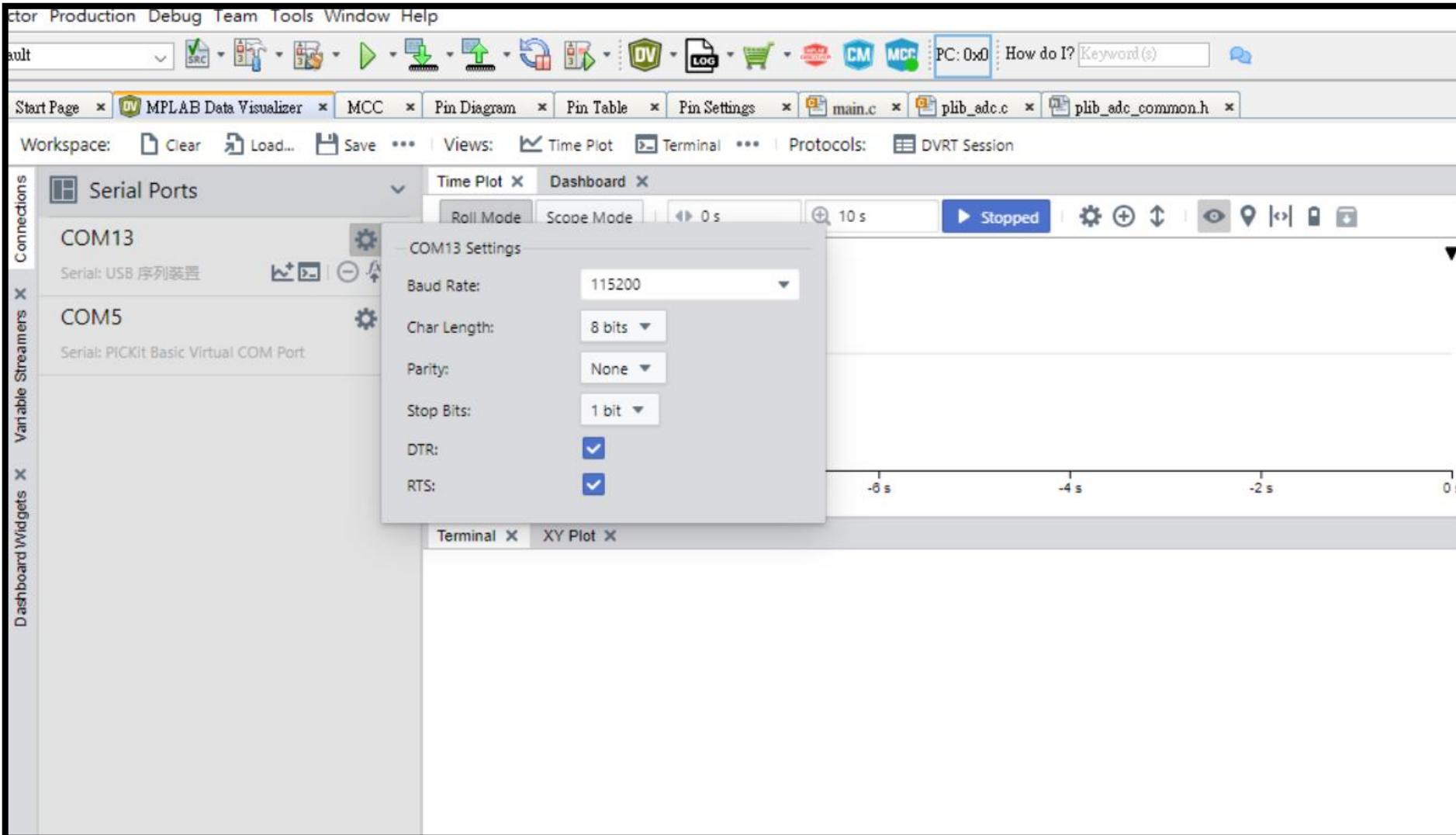


The screenshot shows the MPLAB X IDE interface with the main.c file open in the editor. The code implements a periodic ADC conversion task using the SysTick timer. It includes calls to SYSTICK_TimerStart(), ADC_Enable(), SYS_Tasks(), and ADC_ConversionStart(). A conditional block checks for the bFlag_200ms flag and performs an ADC conversion if it is set.

```
50     SYSTICK_TimerStart();
51
52     ADC_Enable();
53     while ( true )
54     {
55         /* Maintain state machines of all polled MPLAB Harmony modules. */
56         SYS_Tasks ();
57         if (bFlag_200ms)
58         {
59             // ADC0_ChannelSelect(ADC_POSINPUT_AIN0, ADC_NEGINPUT_GND);
60             ADC_ChannelSelect(ADC_POSINPUT_P1, ADC_NEGINPUT_GND);
61             ADC_ConversionStart();
62             while (!ADC_ConversionStatusGet());
63             printf ("ADC = %4d\n", ADC_ConversionResultGet());
64             bFlag_200ms = 0;
65         }
66     }
67 }
```

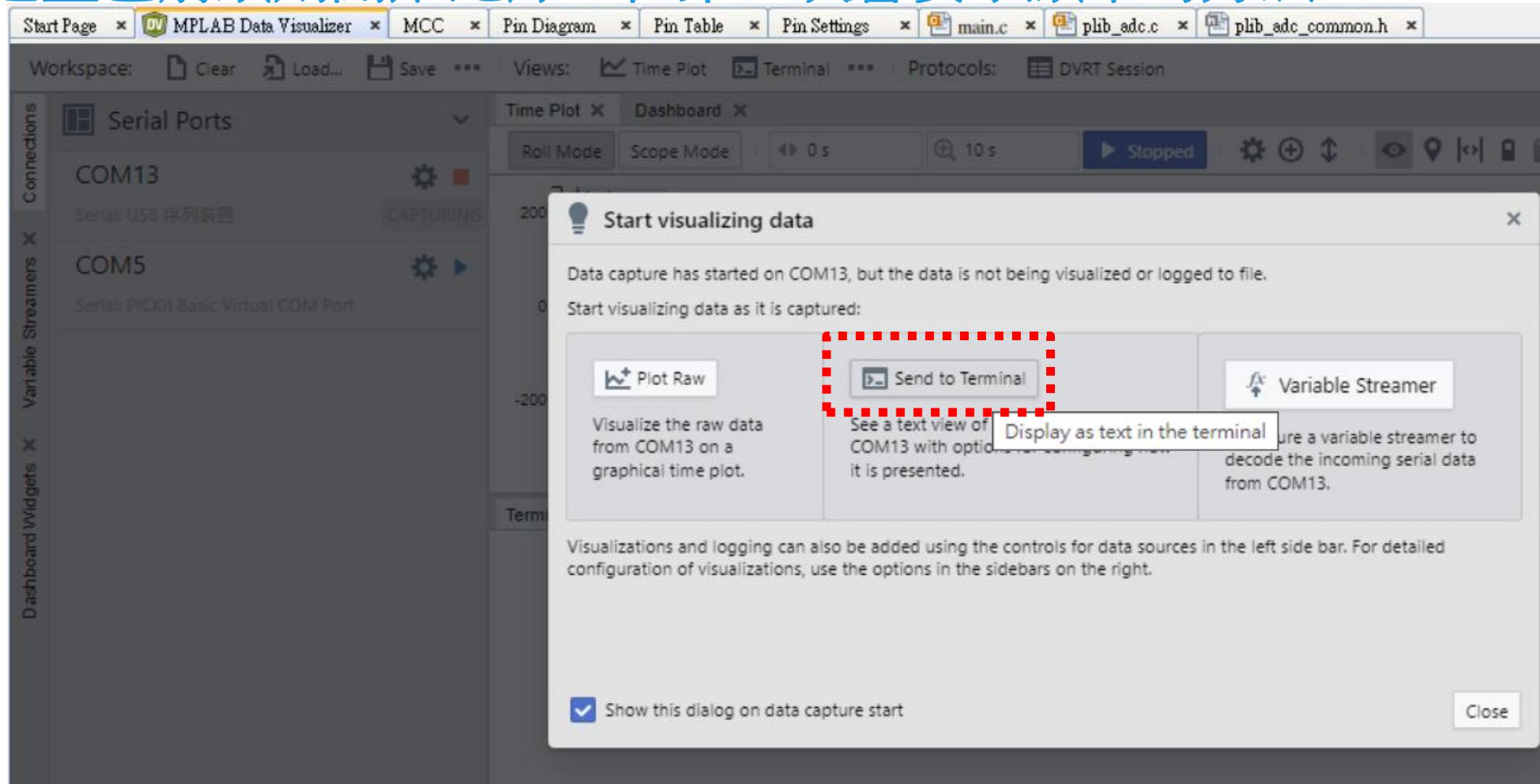
Lab2 – Interface to the World

點選 Data Visualizer，選擇正確的 COM 並確定通信速率及格式正確



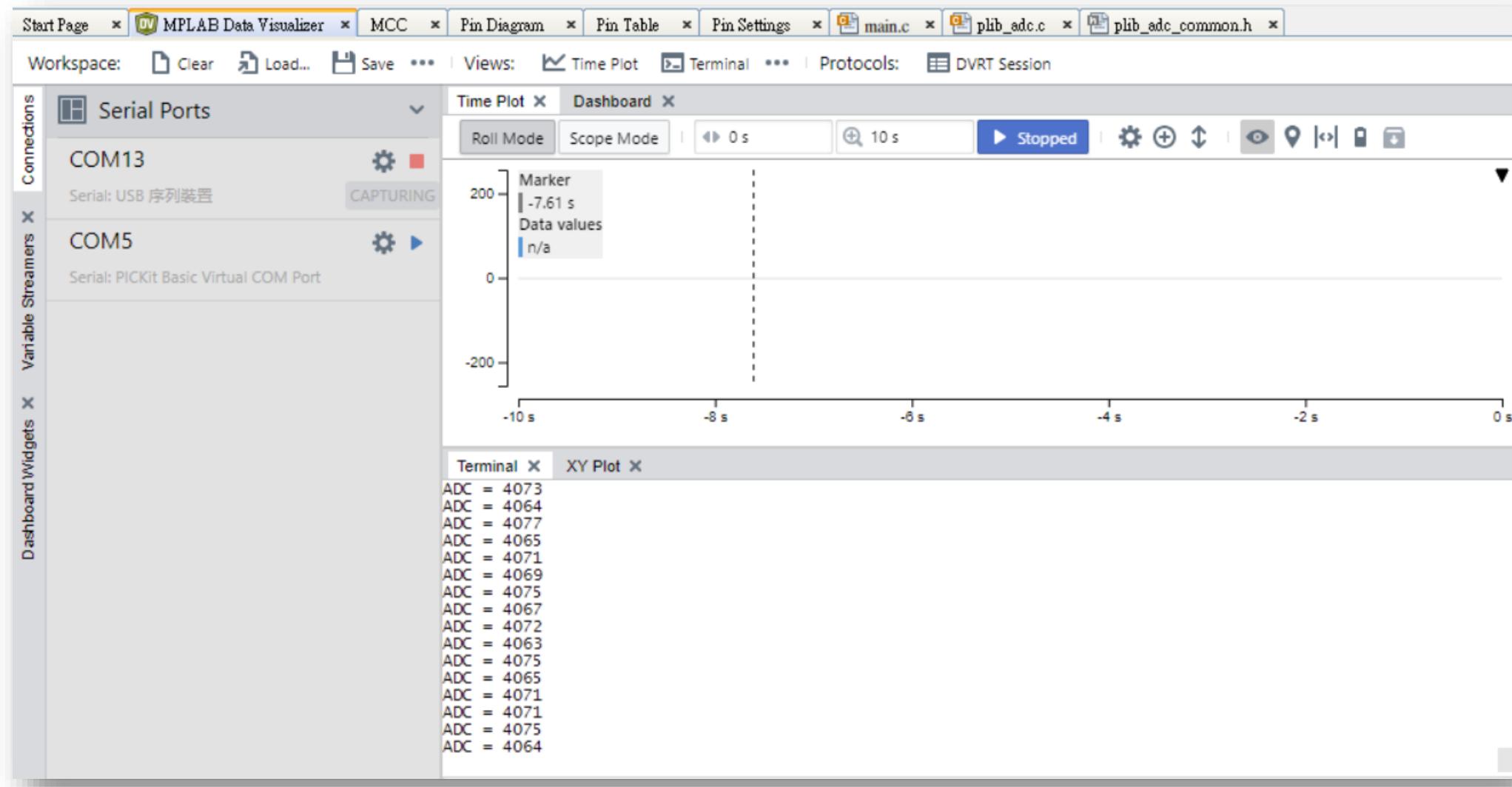
Lab2 – Interface to the World

點選藍色箭頭則開始記錄，但第一次會要求顯示的方法



Lab2 – Interface to the World

ADC 的讀值被 Serial Port 輸出的結果 (轉動 VR 會看到變化)





Lab 3 – OLED Display for ADC Result

Lab 3 Objectives



Continue setup from
Lab 2 (**Reserve all
function**)



Set up SERCOM for I²C
Master support



Initialize and operate
OLED display by using
Library



Additionally update
ADC result to OLED
periodically (200ms)

Lab 3 – OLED Display for ADC Result

 **New Project** X

Steps

1. Choose Project
2. Select Device
3. Select Header
4. Select Plugin Board
5. Select Compiler
6. Select Project Name and Folder
7. (Optional) Add Project

Select Project Name and Folder

Project Name:

Project Location:

Project Folder:

Overwrite existing project.

Also delete sources.

Set as main project

Open MCC on Finish

Use project location as the project folder

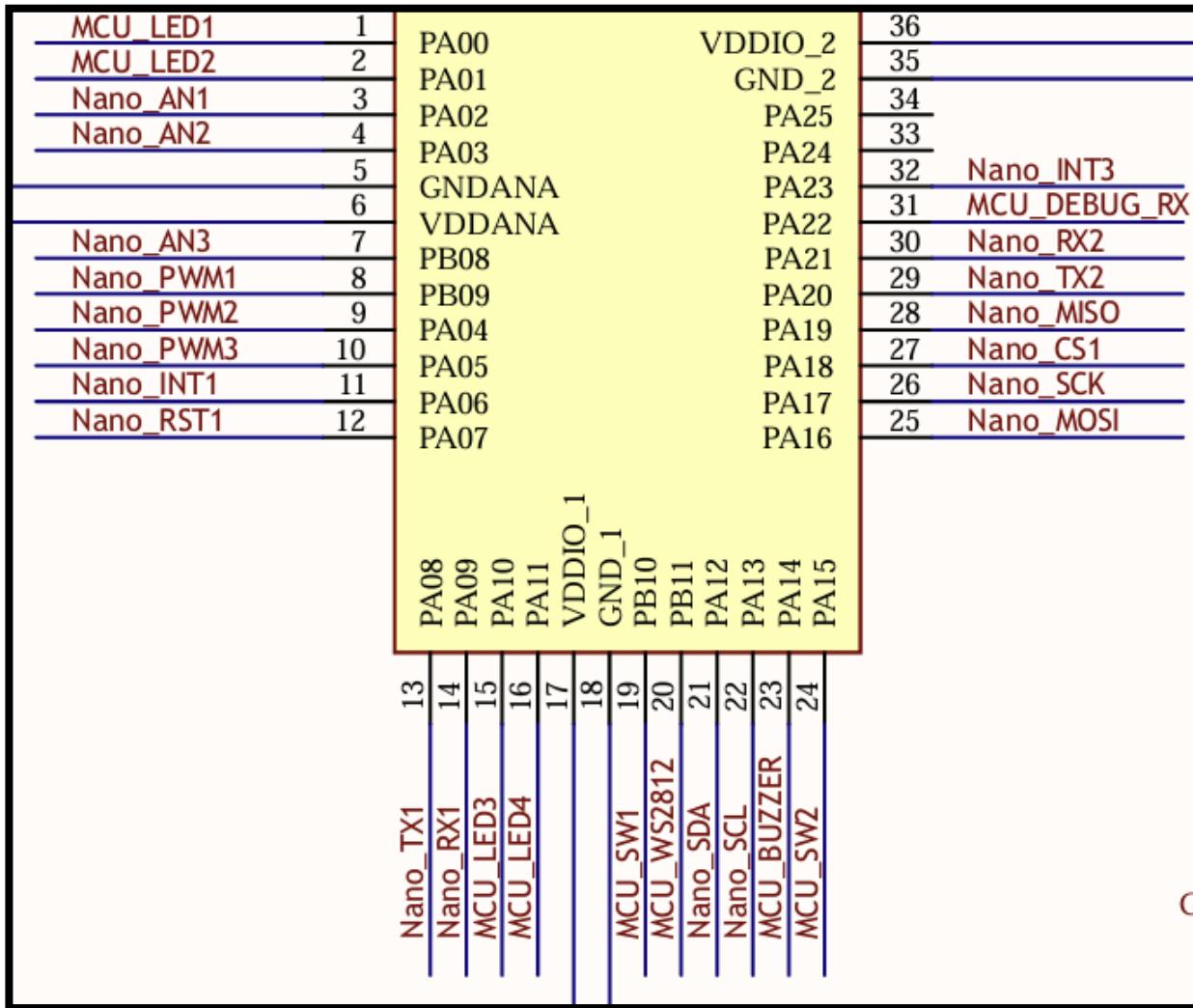
Encoding:

< Back Add Another Project Next > Finish Cancel Help



Lab 3 – OLED Display for ADC Result

SDA & SCL are located @ PA12 & PA13



Signal Name	Type	Description
PAD[0]	Digital I/O	SDA
PAD[1]	Digital I/O	SCL
PAD[2]	Digital I/O	SDA_OUT (4-wire operation)
PAD[3]	Digital I/O	SCL_OUT (4-wire operation)

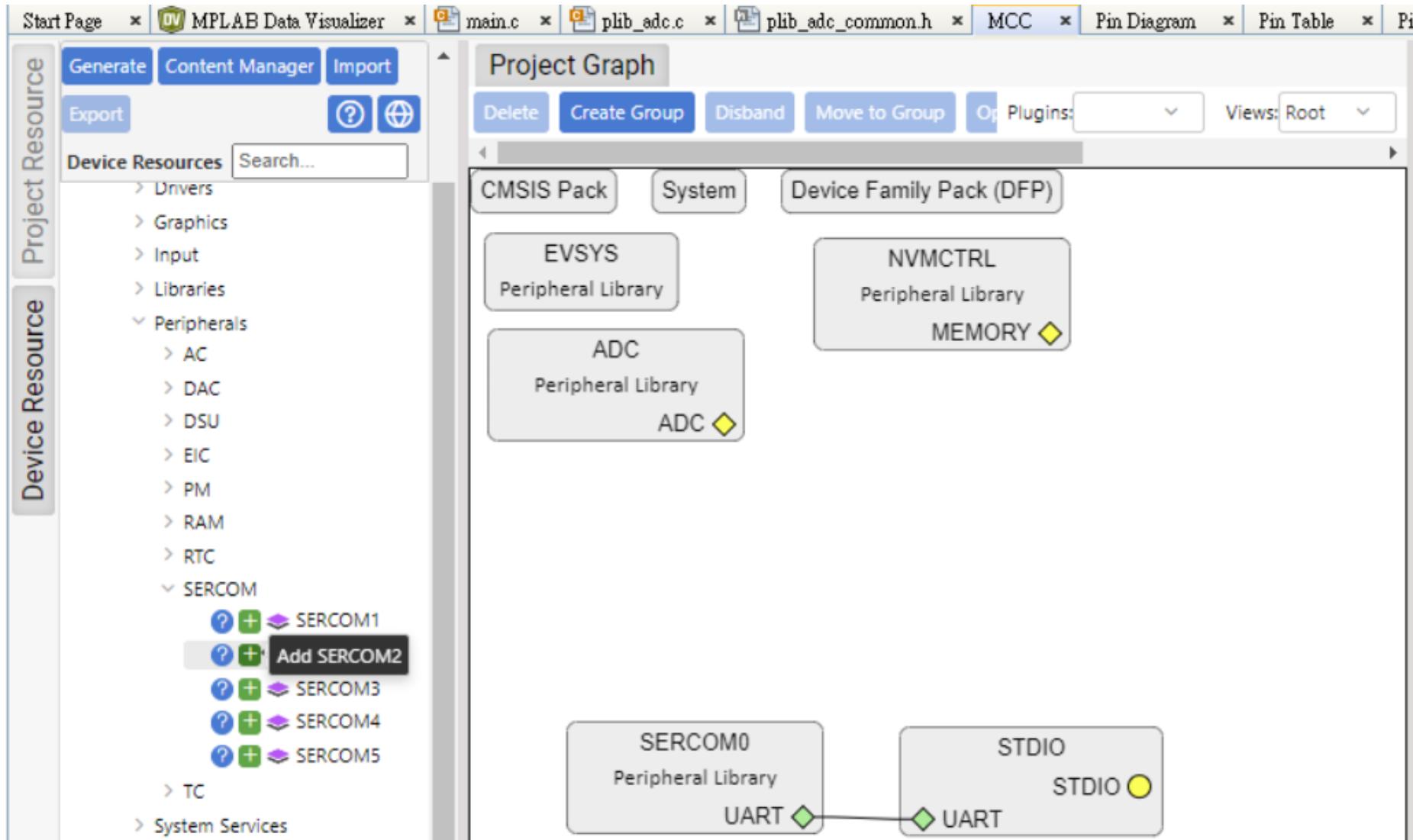
Lab 3 – OLED Display for ADC Result

Set SERCOM2 for I2C function (PAD0 = SDA, PAD1=SCL)

Pin Number	Pin ID	Custom Name	Function	Mode	Direction	Latch	Pull Up	Pull Down	Drive Strength
10	PA05		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
11	PA06		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
12	PA07		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
13	PA08		SERCOM0_PAD0	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
14	PA09		SERCOM0_PAD1	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
15	PA10		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
16	PA11		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
17	VDDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
18	GNDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
19	PB10		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
20	PB11		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
21	PA12		SERCOM2_PAD0	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
22	PA13		SERCOM2_PAD1	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
23	PA14		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
24	PA15		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
25	PA16		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
26	PA17		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
27	PA18		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL

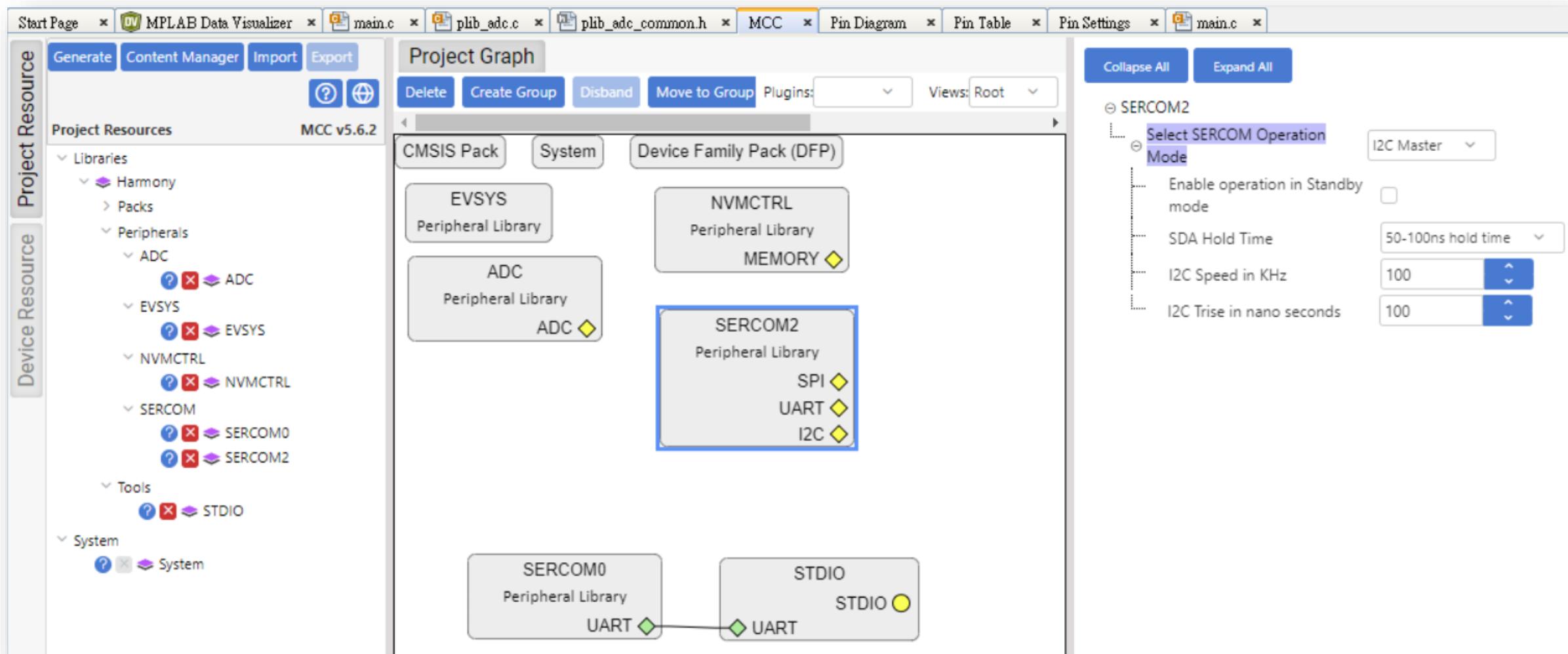
Lab 3 – OLED Display for ADC Result

加入 SERCOM2



Lab 3 – OLED Display for ADC Result

Config SERCOM2 as I2C Master



Lab 3 – OLED Display for ADC Result

Copy all library files to \src folder and add OLED128x64.c to project

The screenshot shows the MPLAB X IDE interface. The title bar reads "MPLAB X IDE v6.25 - PIC32CMGV_Lab3 : default". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Production, Debug, Team, Tools, Window, Help. The toolbar has various icons for file operations like Open, Save, Build, and Run. The project tree on the left shows a project named "PIC32CMGV_Lab3" with subfolders Header Files, Important Files, LibraryFiles, Linker Files, Source Files (containing config, main.c, and OLED128x64.c), Libraries, and Loadables. The main code editor window displays C code. The code starts with a function definition:

```
43
44     void SYSTICK_EventHandler(uintptr_t context)
45     {
46         /* Toggle LED */
47         LED1_Toggle();
48         bFlag_200ms = 1;
49     }
50
51     int main ( void )
52     {
53         /* Initialize all modules */
54         SYS_Initialize ( NULL );
55         /* Register Callback */
```

The file "OLED128x64.c" is highlighted in the project tree. The status bar at the bottom shows "OLED_Put16x16Ch > OLED_Init0(); >".

Lab 3 – OLED Display for ADC Result

Add required code and definition to main.c (1)

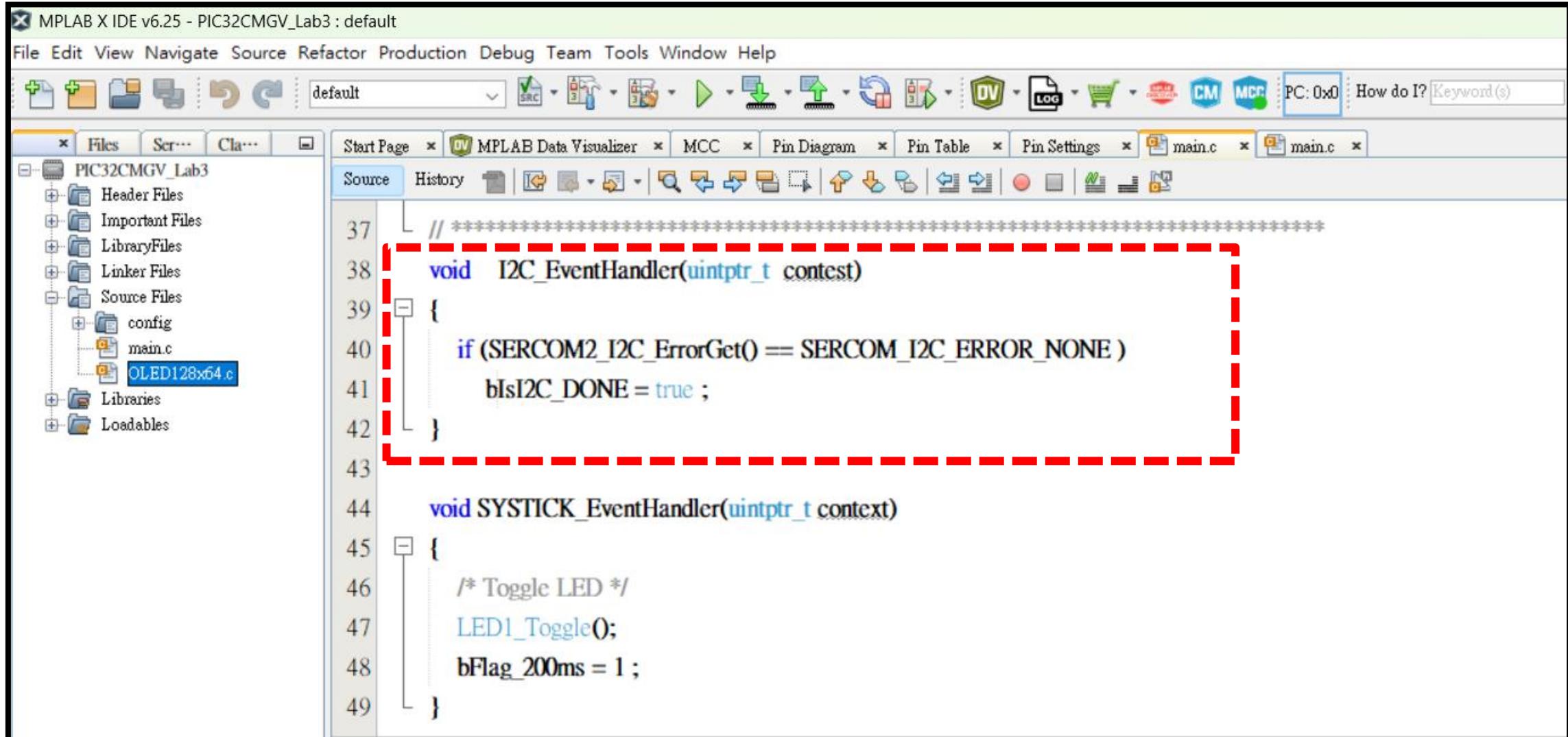
The screenshot shows the MPLAB X IDE interface with the project 'PIC32CMGV_Lab3' open. The 'main.c' file is selected in the tab bar. The code editor displays the following C code:

```
22 // ****
23 // ****
24 #include <stddef.h>           // Defines NULL
25 #include <stdbool.h>          // Defines true
26 #include <stdlib.h>            // Defines EXIT_FAILURE
27 #include "definitions.h"        // SYS function prototypes
28 #include "OLEDI28x64.h"         // Line 28 is highlighted with a red dashed box
29
30 volatile bool    bFlag_200ms = 0;
31 volatile bool    bIsI2C_DONE = true;
32 uint8_t          ASCII_Buffer[24];
33 // ****
34 // ****
```

A red dashed rectangular box highlights the line '#include "OLEDI28x64.h"' at line 28. Another red dashed rectangular box highlights the declarations for 'bFlag_200ms', 'bIsI2C_DONE', and 'ASCII_Buffer' from lines 30 to 32.

Lab 3 – OLED Display for ADC Result

Add required code and definition to main.c (2)



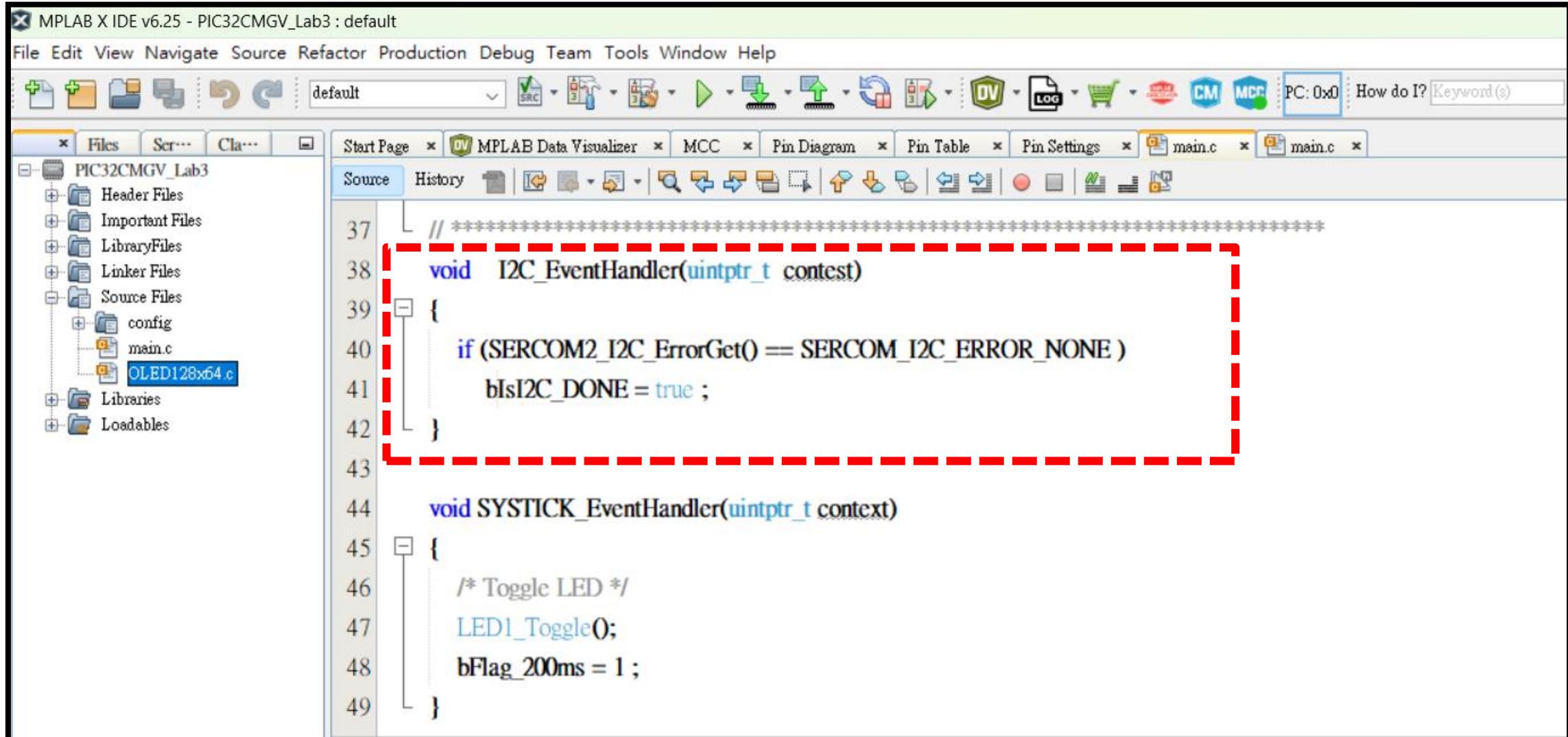
The screenshot shows the MPLAB X IDE interface with the project "PIC32CMGV_Lab3" open. The main window displays the "main.c" source code. Two specific sections of the code are highlighted with red dashed boxes:

```
// *****
void I2C_EventHandler(uintptr_t context)
{
    if (SERCOM2_I2C_ErrorGet() == SERCOM_I2C_ERROR_NONE )
        bIsI2C_DONE = true ;
}

void SYSTICK_EventHandler(uintptr_t context)
{
    /* Toggle LED */
    LED1_Toggle();
    bFlag_200ms = 1 ;
}
```

Lab 3 – OLED Display for ADC Result

Add required code and definition to main.c (3)



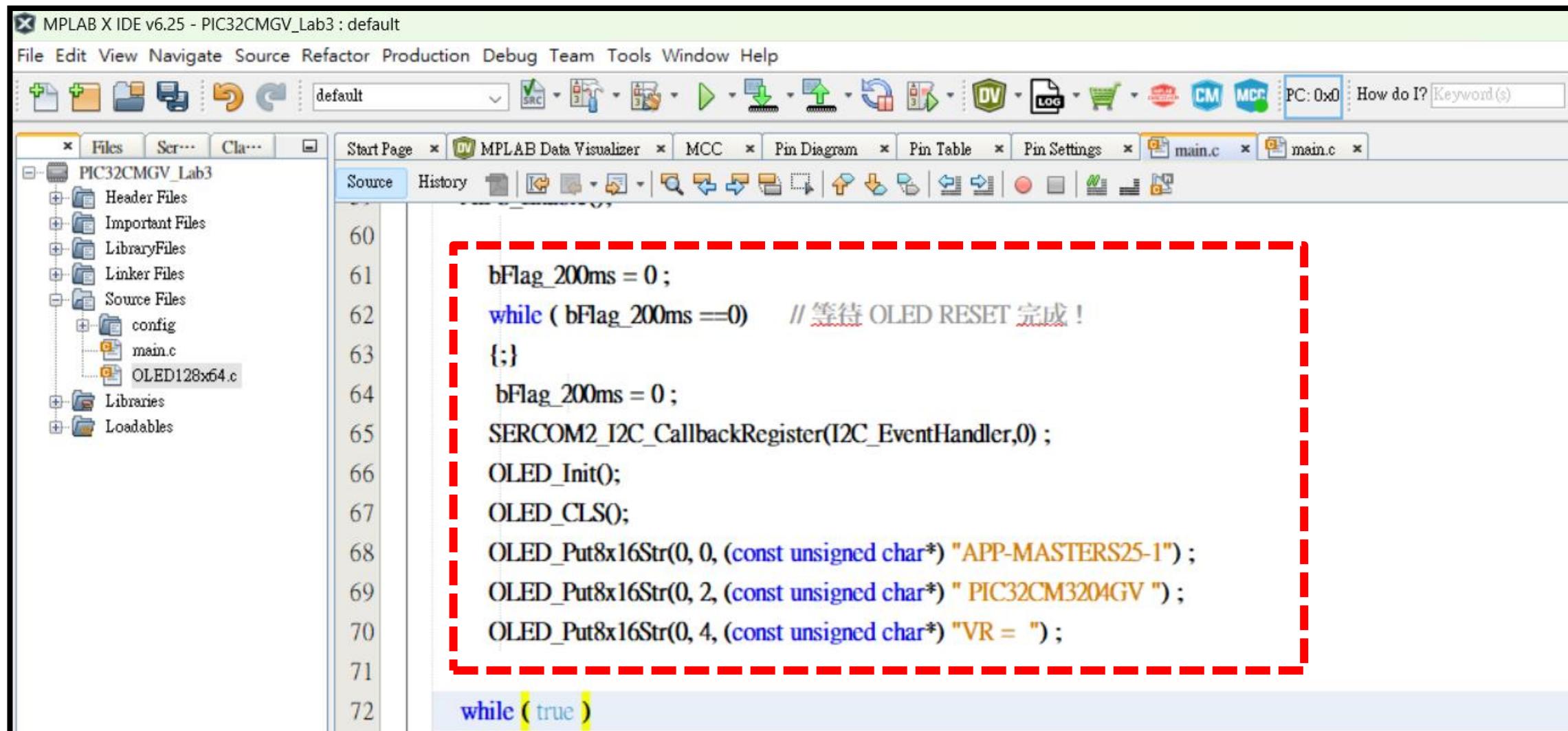
The screenshot shows the MPLAB X IDE interface with the project "PIC32CMGV_Lab3" open. The main window displays the "main.c" source code. Two specific sections of code are highlighted with red dashed boxes:

```
// *****
void I2C_EventHandler(uintptr_t context)
{
    if (SERCOM2_I2C_ErrorGet() == SERCOM_I2C_ERROR_NONE )
        bIsI2C_DONE = true ;
}

void SYSTICK_EventHandler(uintptr_t context)
{
    /* Toggle LED */
    LED1_Toggle();
    bFlag_200ms = 1 ;
}
```

Lab 3 – OLED Display for ADC Result

Add required code and definition to main.c (4)



The screenshot shows the MPLAB X IDE interface with the project 'PIC32CMGV_Lab3' open. The left sidebar displays the project structure, including 'Source Files' containing 'config', 'main.c', and 'OLED128x64.c'. The main editor window shows the 'main.c' file with the following code:

```
60
61     bFlag_200ms = 0 ;
62     while ( bFlag_200ms ==0) // 等待 OLED RESET 完成 !
63     {};
64     bFlag_200ms = 0 ;
65     SERCOM2_I2C_CallbackRegister(I2C_EventHandler,0) ;
66     OLED_Init();
67     OLED_CLSO;
68     OLED_Put8x16Str(0, 0, (const unsigned char*) "APP-MASTERS25-1") ;
69     OLED_Put8x16Str(0, 2, (const unsigned char*) " PIC32CM3204GV ") ;
70     OLED_Put8x16Str(0, 4, (const unsigned char*) "VR = ") ;
71
72     while ( true )
```

A red dashed rectangle highlights the code from line 61 to line 71, which initializes the OLED display by setting up the SERCOM2 I2C callback and calling the OLED_Init() and OLED_CLSO() functions. The code then prints the strings "APP-MASTERS25-1", " PIC32CM3204GV ", and "VR = " to the OLED screen.

Lab 3 – OLED Display for ADC Result

Add required code and definition to main.c (5)

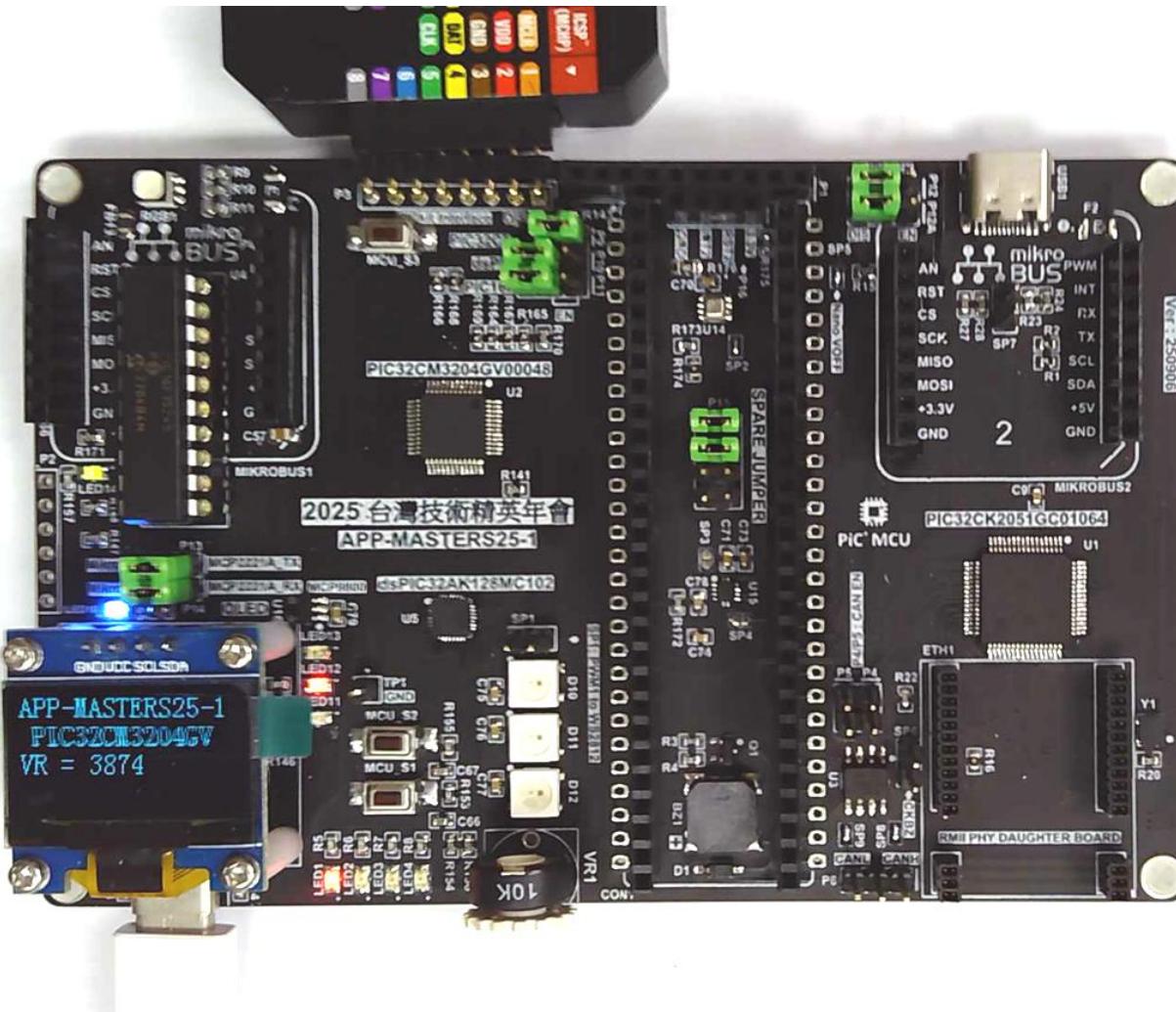
The screenshot shows the MPLAB X IDE interface with the title bar "MPLAB X IDE v6.25 - PIC32CMGV_Lab3 : default". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Production, Debug, Team, Tools, Window, and Help. The toolbar has various icons for file operations like Open, Save, Build, and Run. The project tree on the left shows "PIC32CMGV_Lab3" with subfolders Header Files, Important Files, LibraryFiles, Linker Files, Source Files, Libraries, and Loadables. The main editor window displays the "main.c" file with the following code:

```
75 SYS_Tasks ();
76 if (bFlag_200ms)
77 {
78     // ADC0_ChannelSelect(ADC_POSINPUT_AIN0, ADC_NEGINPUT_GND);
79     ADC_ChannelSelect(ADC_POSINPUT_PIN1, ADC_NEGINPUT_GND);
80     ADC_ConversionStart();
81     while (!ADC_ConversionStatusGet());
82     printf ("ADC = %4d\n", ADC_ConversionResultGet());
83     sprintf((char*)ASCII_Buffer, "%4d", ADC_ConversionResultGet());
84     OLED_Put8x16Str(40, 4, ASCII_Buffer);
85     bFlag_200ms = 0;
86 }
87 }
```

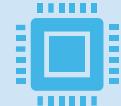
A red dashed box highlights the section of code from line 82 to line 84, which prints the ADC result to a buffer and then displays it on the OLED screen.

Lab 3 – OLED Display for ADC Result

Lab3 實際執行結果



Lab 3 Summary



Properly set up SERCOM to implement I²C MASTER Function



Add Library into your project



Config OLED Display by using Library



Display ADC read result to Line-3 of OLED



The End

LEGAL NOTICE

MASTERs 2025

SOFTWARE:

Microchip software provided to you during the MASTERs 2025 conference is provided "AS IS", without warranty. Microchip assumes no liability or responsibility for your use of any Microchip software. You must use Microchip software exclusively with Microchip products. Further, use of Microchip software is subject to copyright and the disclaimers and license terms accompanying such software, whether set forth at the install of each program or posted in a header or text file.

Notwithstanding the above, certain components of software offered by Microchip and 3rd parties may be covered by "open source" software licenses, including licenses that require that the distributor of such software make the software available in source code format. To the extent required by any such open-source software license, the terms of such license govern.

Any transfer of Microchip product, technology, or software may be subject to export controls. Microchip encourages any entity transferring product, technology, or software to seek appropriate legal advice and/or consult the applicable governmental agencies prior to transfer. The information provided herein is subject to change at any time and without notice.

NOTICE & DISCLAIMER:

Materials, software and accompanying information (including, for example, any references to 3rd party companies and 3rd party websites) are for informational purposes only and are provided "AS IS." Microchip assumes no responsibility for statements made by 3rd party companies, or materials or information that such 3rd parties may provide.

MICROCHIP DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, INCLUDING ANY IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY DIRECT OR INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND RELATED TO THESE MATERIALS OR ACCOMPANYING INFORMATION PROVIDED TO YOU BY MICROCHIP OR OTHER THIRD PARTIES, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR THE DAMAGES ARE FORESEEABLE. PLEASE BE AWARE THAT THE IMPLEMENTATION OF INTELLECTUAL PROPERTY PRESENTED IN MASTERs 2025 MAY REQUIRE A LICENSE FROM MICROCHIP OR THIRD PARTIES.

TRADEMARKS:

The Microchip name and logo and the Microchip logo are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. Information on and a listing of Microchip trademarks is available at <https://www.microchip.com/en-us/about/legal-information/microchip-trademarks>.

All other trademarks mentioned herein are property of their respective companies.

COPYRIGHT:

© 2025 Microchip Technology Incorporated, All Rights Reserved.