



# **MICROCHIP**

---

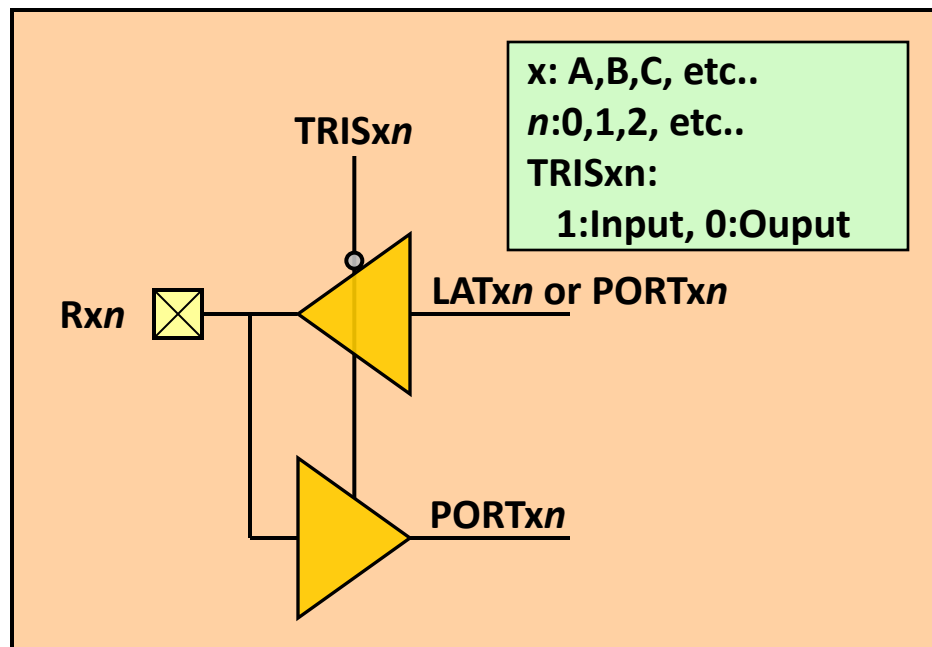
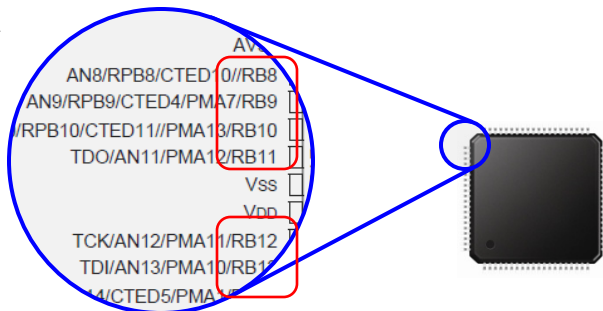
***Regional Training Centers***

## **Section 5**

### **IO Port Architecture**

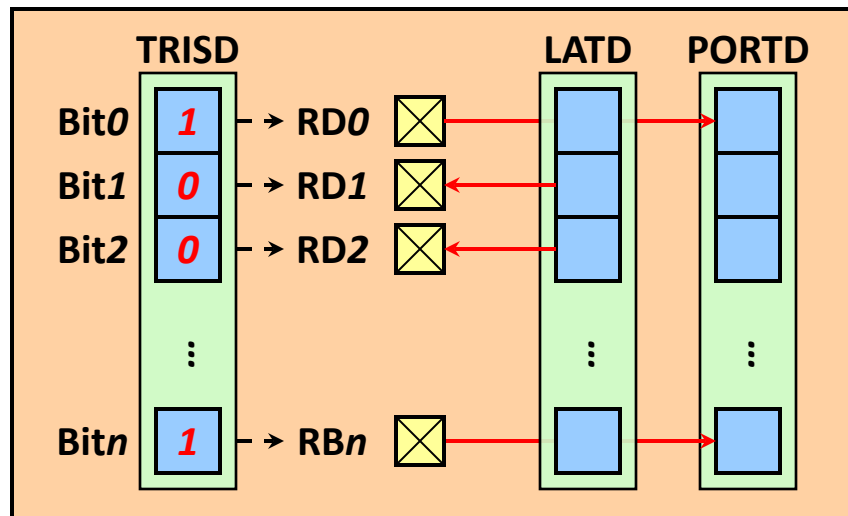
# I/O Port Block Diagram

- 16 Bits MCU的IO Port示意圖, 如圖所示。
- 所有的IO Port都具有 $TRIS_x$ ,  $LAT_x$ 跟 $PORT_x$ 三個特殊功能暫存器。  
 $TRIS_x$ 用來設定輸出或輸入。  
**設定為"0", 表示輸出;設定為"1", 表示輸入。**
- 設為輸出時, 欲輸出的狀態填入 $LAT_{xn}$ 或 $PORT_{xn}$ , 對應的接腳 $R_{xn}$ 就會有輸出高準位或低準位。
- 設為輸入時, 可以讀取 $PORT_{xn}$ 取得外部的實際狀態。



# I/O Port Manipulation

- TRISx, LATx跟PORTx特殊功能暫存器都是一個16 Bits的暫存器, 暫存器中每個Bit都控制著對應的I/O接腳。
- 以PortD為例, TRISD, LATD, PORTD的Bit 8, 控制RD8接腳。
- 要用RD8控制LED時, 必須先把TRISD的Bit 8設為0(輸出), 然後把要輸出的狀態, 填入LATD的Bit 8。
- 要用RD8讀取按鍵狀態時, 則必須把TRISD的Bit 8設為1(輸入), 然後就可以從PORTD的Bit 8取得外部的狀態。



# Lab1 – BasicIO

## 目標


- 嘗試透過MCC的設定, 加入IO Port的控制程式。
  - 透過MCC的設定過程, 了解MCC的各項基本功能。
  - 請嘗試控制RD8(D4), 讓RD8可以不斷的轉態(Toggle, 1<->0)。
- 
- 該如何開始？

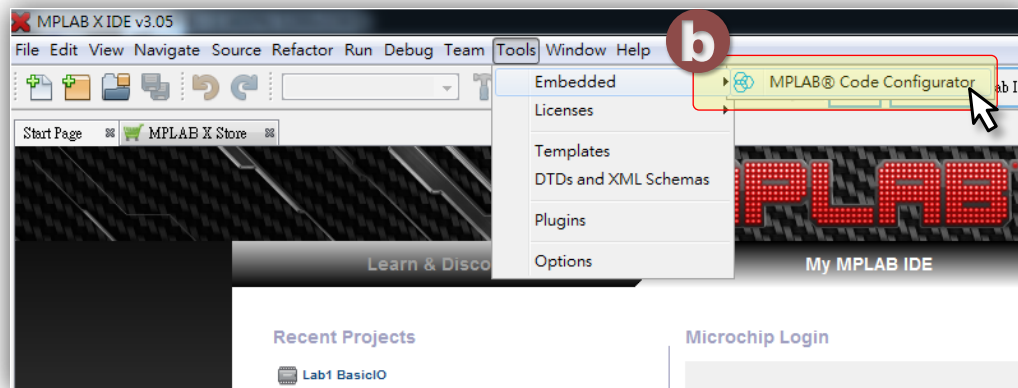
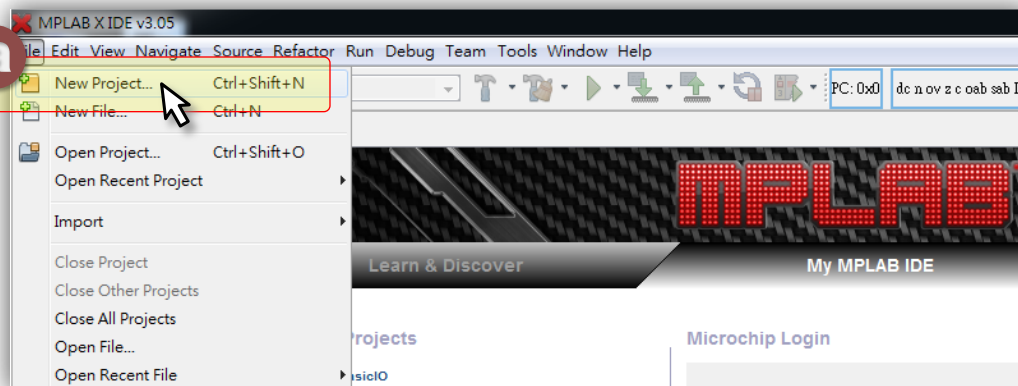
# Lab1 – BasicIO Step



## 步驟 A

- a** 建立新專案  
(C:\Exercises\Lab1 BasicIO.X)
- b** 開啟MCC  
功能表 ▶ Tools ▶ Embedded  
▶ MPLAB Code Configurator

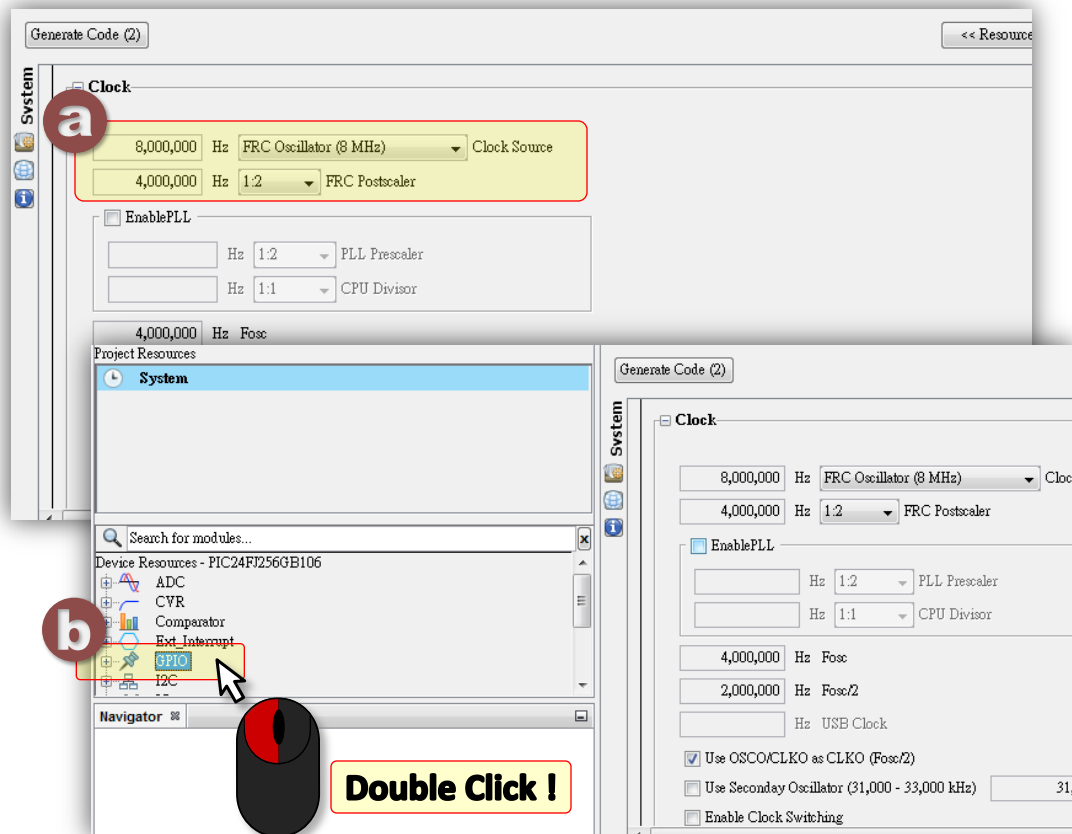
 建立空白專案後，不需要建立任何檔案。直接執行MCC即可。所有需要的檔案，MCC都會協助產生。



# Lab1 – BasicIO Step

## 步驟 B

- a** 設定系統時脈  
點選 ► **System** 設定  
系統時脈為  
Clock Source: **FRC**  
Postscaler: **1:2**
- b** 新增專案需求資源  
► **GPIO**



# Lab1 – BasicIO Step

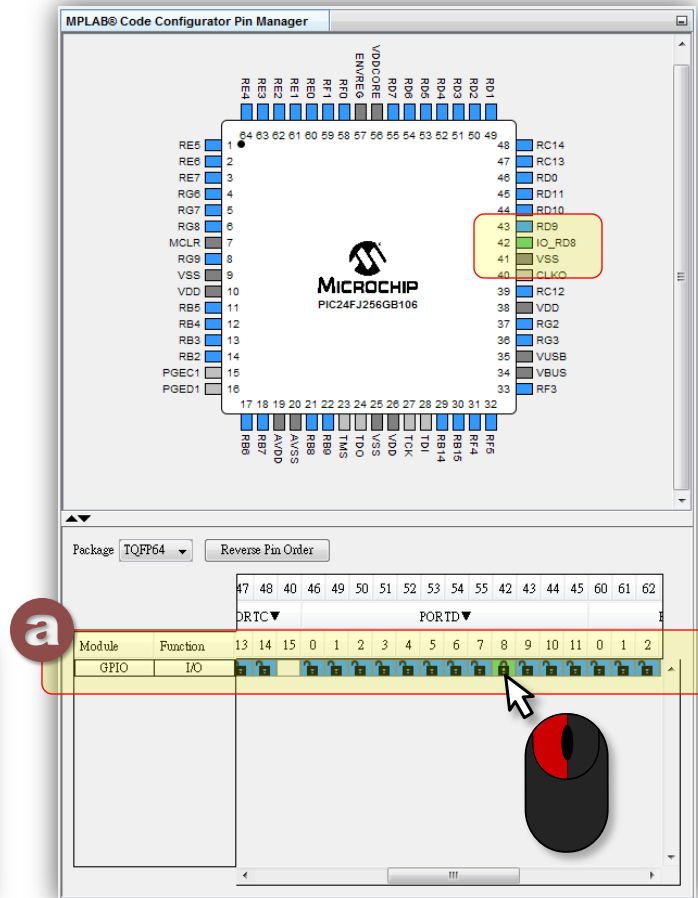
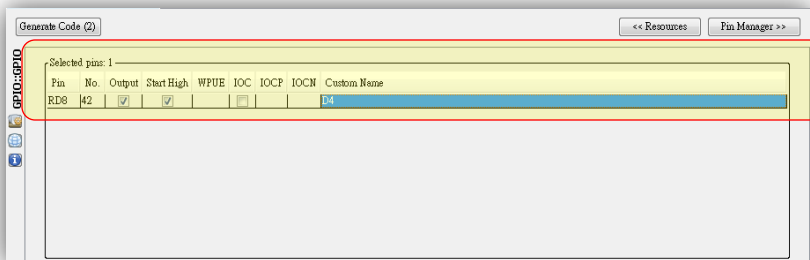


## 步驟 C

**a** 從接腳管理區指定  
▶RD8接腳。



指定接腳時, 可以觀察下封裝圖示的變化, 以及建構區域的變化。也可以嘗試取消指定, 觀察變化。



# Lab1 – BasicIO Step



## 步驟 D

**a** 從建構區域設定  
**GPIO RD8**的各項屬性。

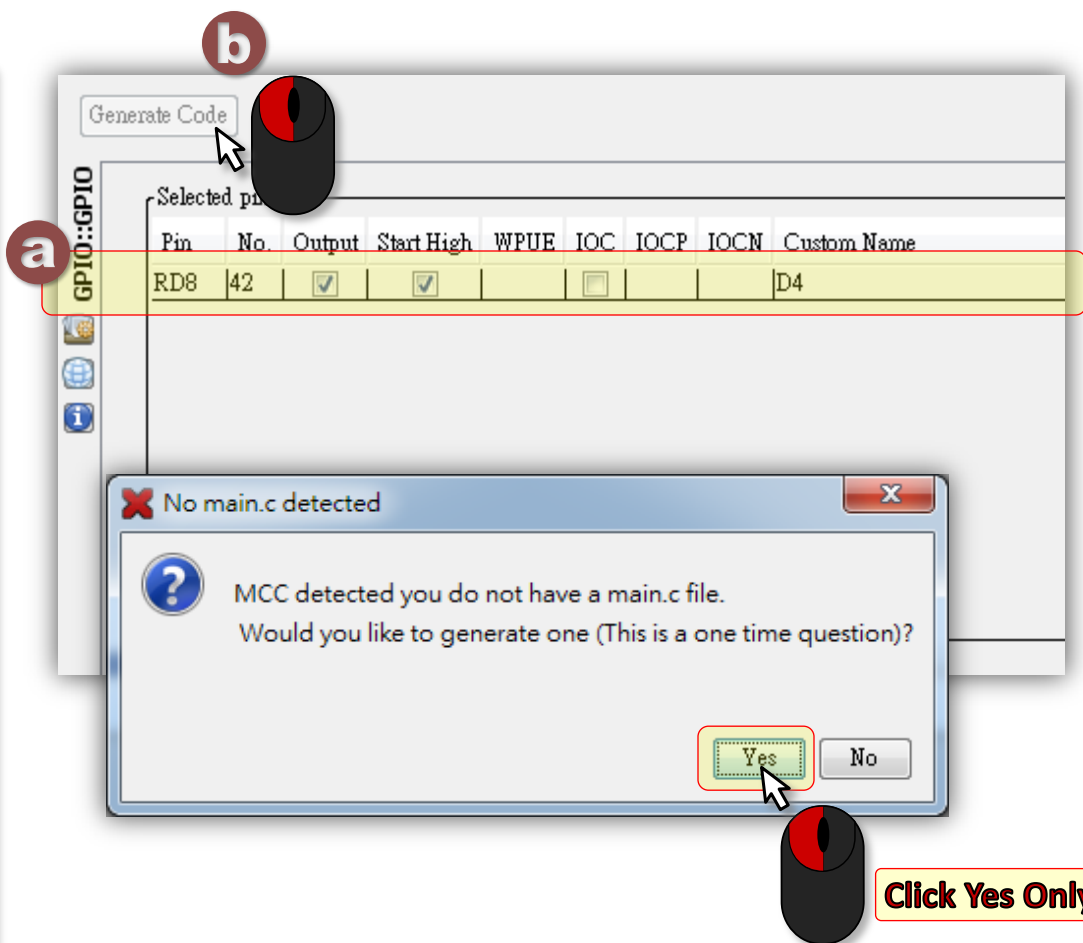
- ▶ **Output** ☒
- ▶ **Start High** ☒
- ▶ **Custom Name : D4**

**b** 選擇

- ▶ **Generate Code(2)**



針對空白專案, 第一次產生程式碼時, 會詢問是否自動建立 **main.c** 以及內部的必要程式。如果選否, 之後無法再自動產生。



**Click Yes Only!**



# Lab1 – BasicIO Step



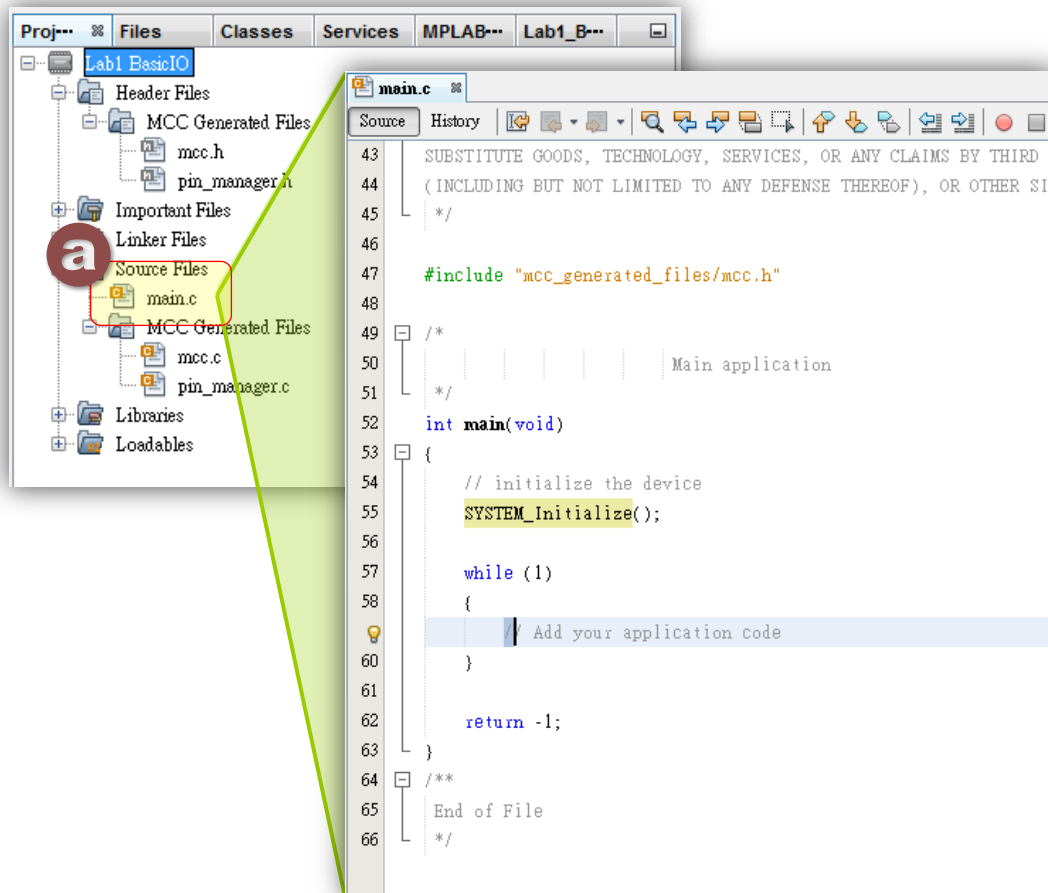
## 步驟 E



回到專案視窗, 觀察專案變化。所有的檔案都由MCC預先完成。



開啟 **main.c**, 開始撰寫程式。



# Lab1 – BasicIO Step



步驟 F

**a** 在 **main.c** 中加入以下  
片段:

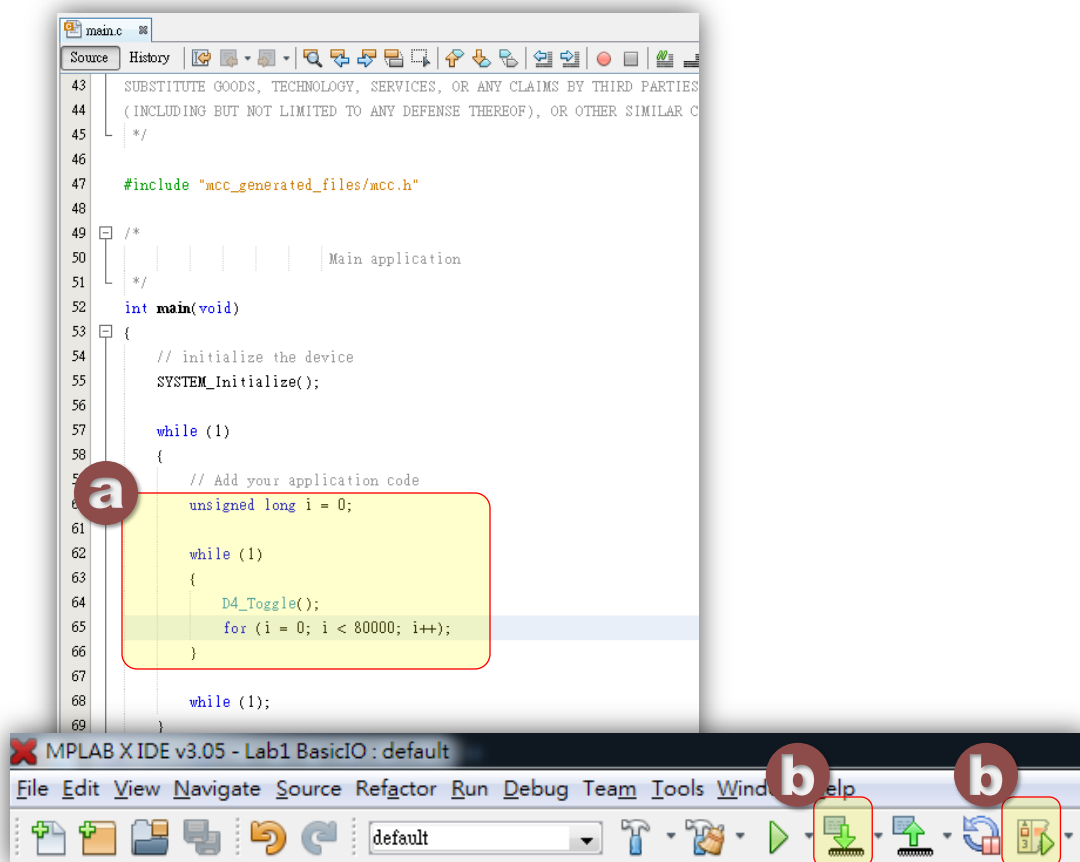
```
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();

    while (1)
    {
        // Add your application code
        unsigned long i = 0;

        while (1)
        {
            D4_Toggle();
            for (i = 0; i < 80000; i++);
        }

        while (1);
    }
}
```

**b** 使用  或  執行程式  
並驗證結果。



# Lab1 – BasicIO Step



觀察

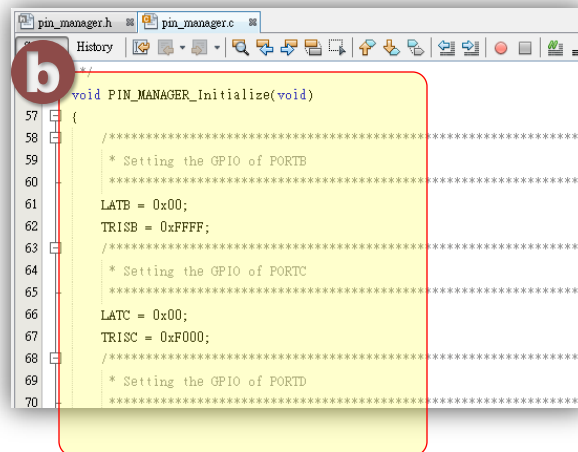
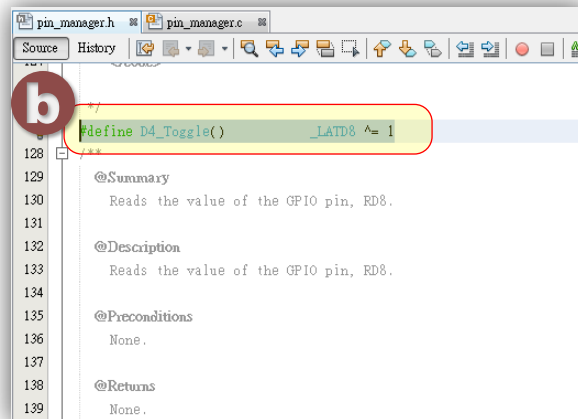
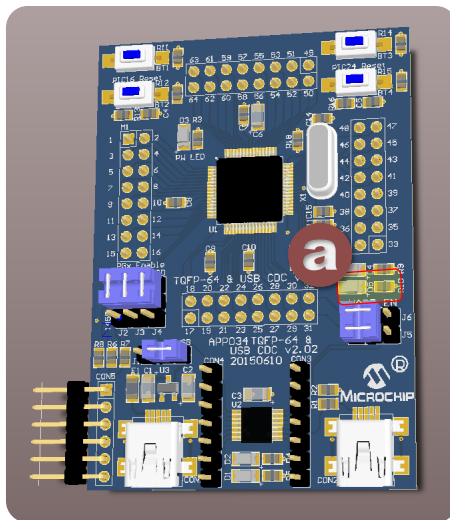
**a** 觀察實驗板上**RD8(D4)**的狀態。是否有閃爍?

**i** 透過專案視窗觀察MCC實際上產生了那些檔案, 檔案的內容又為何?

**b** 開啟

- ▶ **pin\_manager.h**
- ▶ **pin\_manager.c**

觀察有關RD8(D4)接腳的相關定義。



# Lab1 – BasicIO

## MCC's Setting & Code Example

Package: TQFP64 Reverse Pin Order

Module	Function	2	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	0	1	2	3	4	5	6	7	0	1	3	4	5	2	3	6	7	8	9
GPIO	I/O																																			

```
main.c
while (1)
{
    unsigned long i = 0;

    D4_Toggle();
    for (i = 0; i < 80000; i++);
}
```

Generate Code

Selected pins: 1

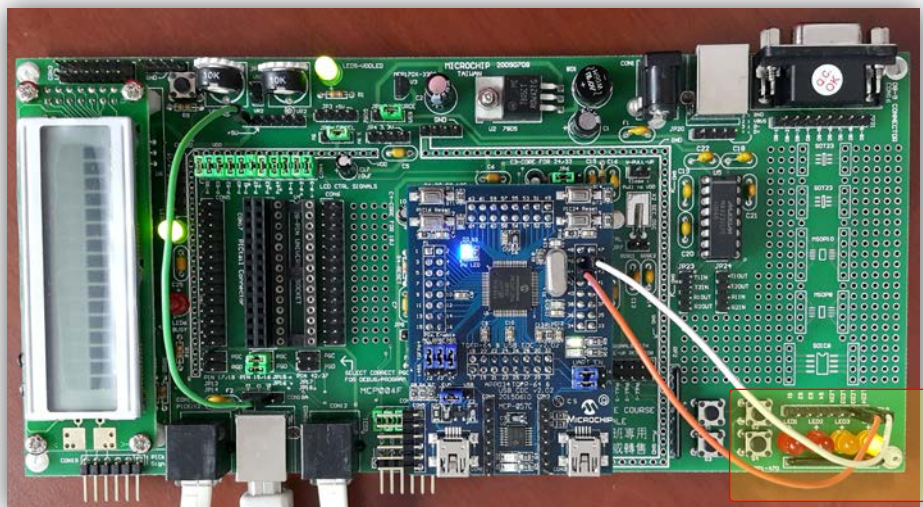
Pin	No.	Output	Start High	WPUE	IOC	IOCP	IOCN	Custom Name
RD8	42	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	D4

# Lab2 – IO Control



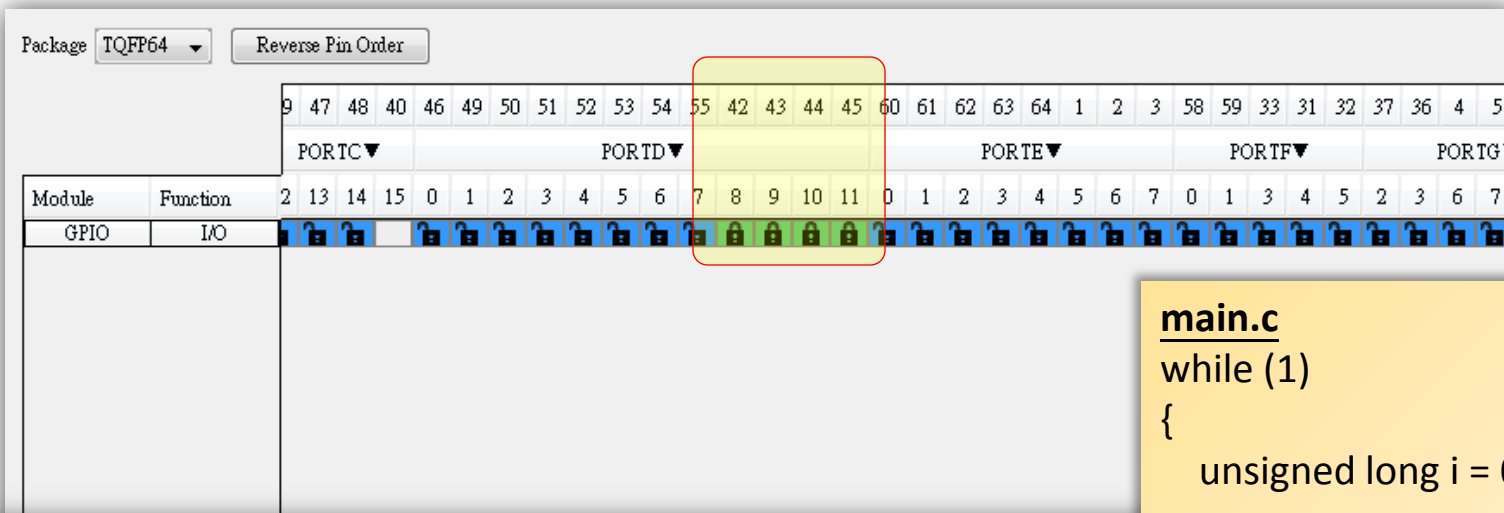
## 目標

- 嘗試在Lab1的程式架構上, 增加**RD9(D5)**的控制程式, 讓D4與D5交錯閃爍。
- 嘗試使用杜邦線連接APP026-3x上的LED, 並增加控制程式, 讓APP026-3x上的LED3, LED4加入閃爍。  
**(RD8(D4), RD9(D5), RD10(LED3, *Pin44*), RD11(LED4, *Pin45*))**



# Lab2 – IO Control

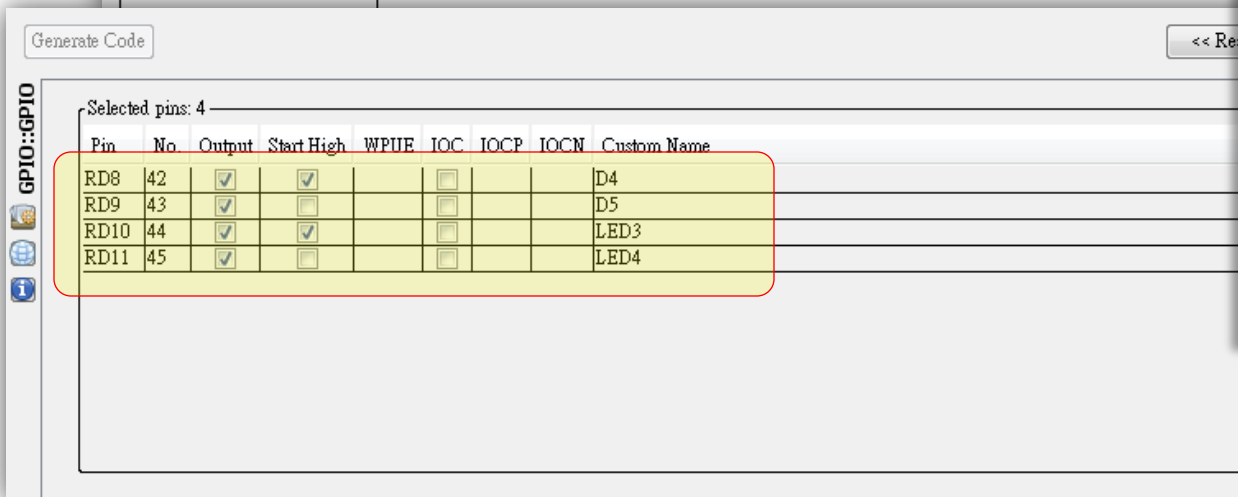
## MCC's Setting & Code Example



### main.c

```
while (1)
{
    unsigned long i = 0;

    D4_Toggle();
    D5_Toggle();
    LED3_Toggle();
    LED4_Toggle();
    for (i = 0; i < 80000; i++);
}
```



# Analog or Digital Mode

- 有些GPIO接腳( $Rxn$ )的功能跟類比輸入( $ANn$ )是共用的。這些接腳在MCU Reset/Power On後的預設值為Analog Mode, 因此無法立即做為數位輸出入使用。如果碰到這類接腳, 在使用時必須先取消Analog Mode改為Digital Mode, IO才能正常動作。
- PIC24FJ256GB106透過ADxPCFGL and ADxPCFGH來設定。  
**1為Digital mode, 0為Analog mode**。

