

# 處理器設計與實作

## 實習講義

編撰者

助教：邱健鳴、蔡森至、張瑞元、曾柏翔、許元杰  
暨成大電通所計算機架構與系統研究室CASLAB

國立成功大學電機系與電腦與通信工程研究所

## Lab7-實驗大綱與目的

- ⊕ Project Introduction
- ⊕ 實作一 (GPIO experiment)
- ⊕ Appendix :溫度感測器模組測試

# Project introduction

- ⊕ 使用助教提供或是自行購買的模組完成一個期末作品。
- ⊕ 助教可提供LCD、temperature module、open source code(裡面包含許多API)和教學講義
- ⊕ 需繳交demo 作品以及final project report(期限為最後一次實驗課 1/4)
- ⊕ Report 至少需有題目構想、軟體流程圖、程式碼解析、心得

# Ex: Project Report

⊕ 題目構想: 我們想做一個溫度計，會依照目前的溫度顯示在lcd上，並且依照不同溫度，LED會有不同的閃爍速率.....

# Ex: Project Report

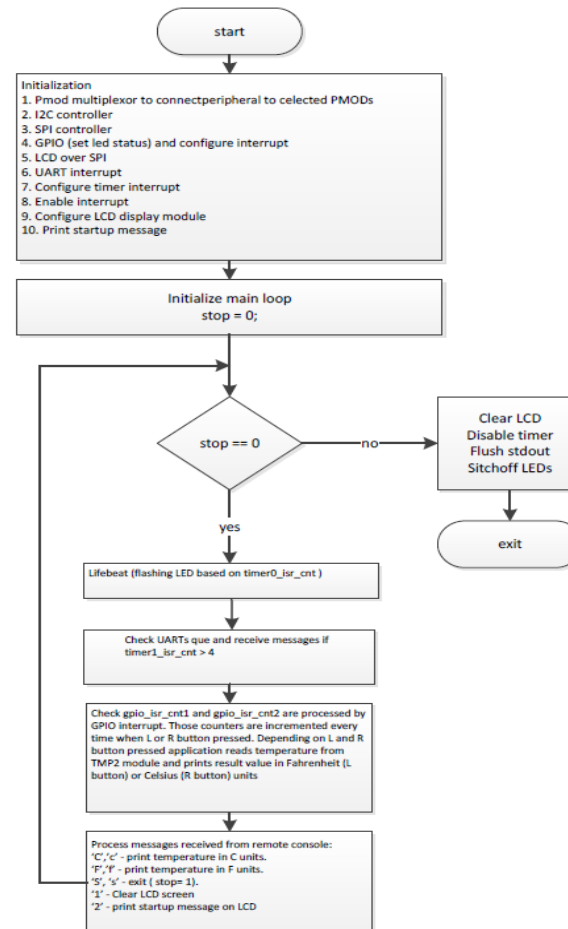
## 軟體流程圖:

**Timer interrupt**  
Increments sw timers each 1000 CPU tics:

```
timer0_isr_cnt++;  
timer1_isr_cnt++;
```

**UART interrupt**  
Receive messages form remote console (messages are processed in main loop)

**GPIO interrupt**  
Every time when pressed onboard button this interrupt increment corresponding sw counter:  
gpio\_isr\_cnt1 for 'L' button  
gpio\_isr\_cnt2 for 'R' button  
gpio\_isr\_cnt3 for 'A' button



# What is API

API就是皮卡丘 提供 鋼鐵尾巴 跟 雷電 兩種技能給你呼叫 基本上 你不用研究皮卡丘為什麼會發電 也不用研究為什麼尾巴會變鋼鐵 反正你只要說: 上吧皮卡丘 使用雷電!!

# What is API

- ⊕ API將許多硬體register設定包在一個function內
- ⊕ 使用者不需知道是如何實現此function，只要知道function功用



```
uart_print(uart, "\n\r*****\n\r");
uart_print(uart, "  ARC EM Starter kit v2.0 *\n\r");
uart_print(uart, "  Flashing LEDs(GPIO) demo *\n\r");
uart_print(uart, "*****\n\r");
```

**API**

```
// simple debug print
void uart_print(DWCREG_PTR uartRegs, const char * pBuf) {
    unsigned int i = MAX_DEBUG_MSG;

    unsigned char byte = *pBuf++;
    while(byte && i--) {

        // wait if FIFO is full
        while(!(uartRegs[U_USR] & U_USR_TFNF));

        // transmitt data byte
        uartRegs[U_THR] = byte;
        byte = *pBuf++;
    }
}
```

# GPIO 介紹

- ⊕ GPIO 是種具有彈性且可以藉由軟體控制 (software-controlled) 的數位訊號
- ⊕ 常見於開發版邊緣, 以針腳 (Pin) 的形式呈現
- ⊕ 這些針腳即是開發版與外界溝通的重要橋樑
- ⊕ 簡單例子, 想像成是開關, 使用者可以打開或關閉 (input), 或由開發版來打開或關閉 (output)

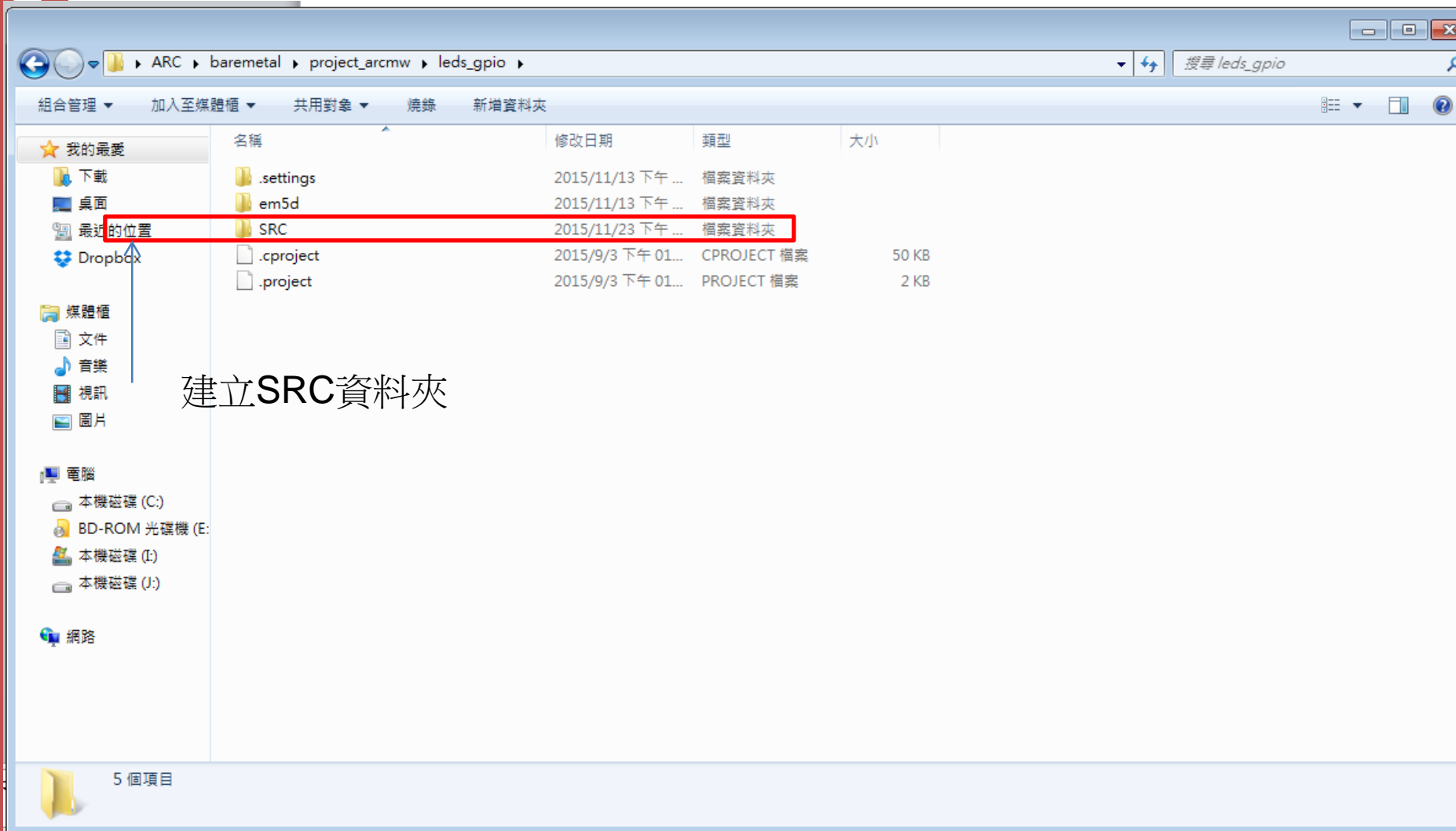


## 實作過程

- ⊕ 步驟與上個實驗一模一樣 run leds\_gpio project

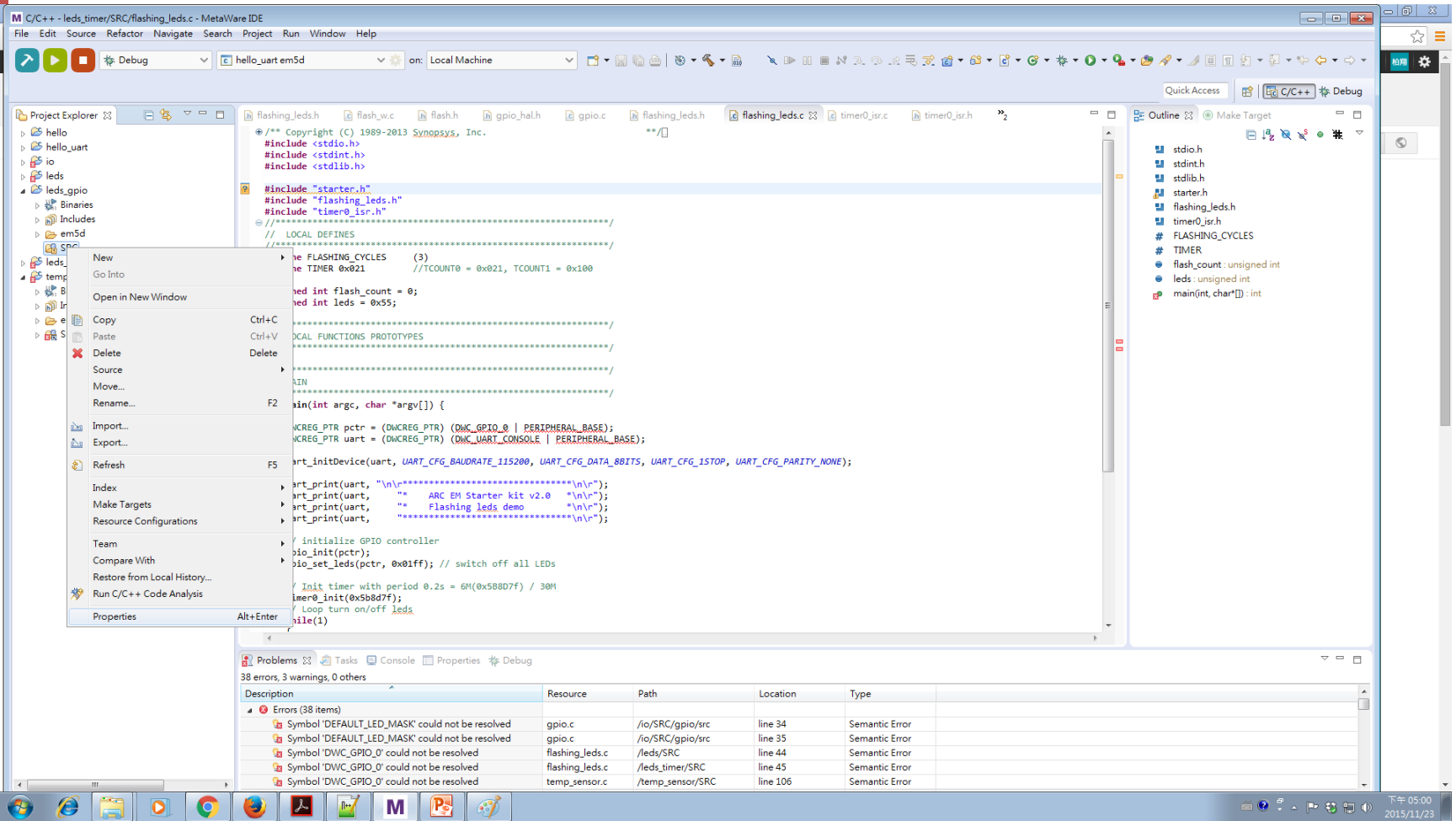
⊕ 如果src裡面本身沒有gpio\_interrupt.c等檔案，可參照下幾頁投影片執行

# Step1: create SRC 資料夾

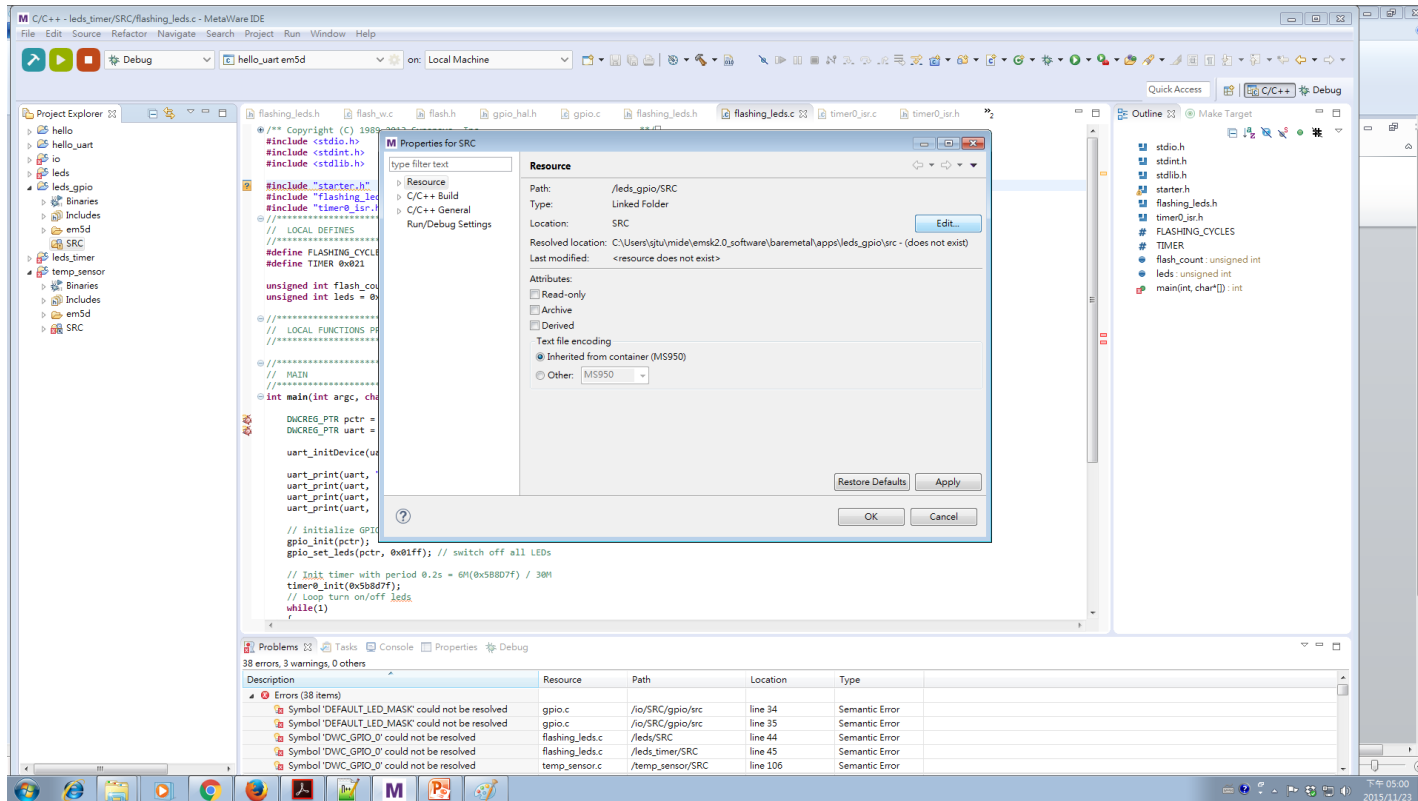


建立SRC資料夾

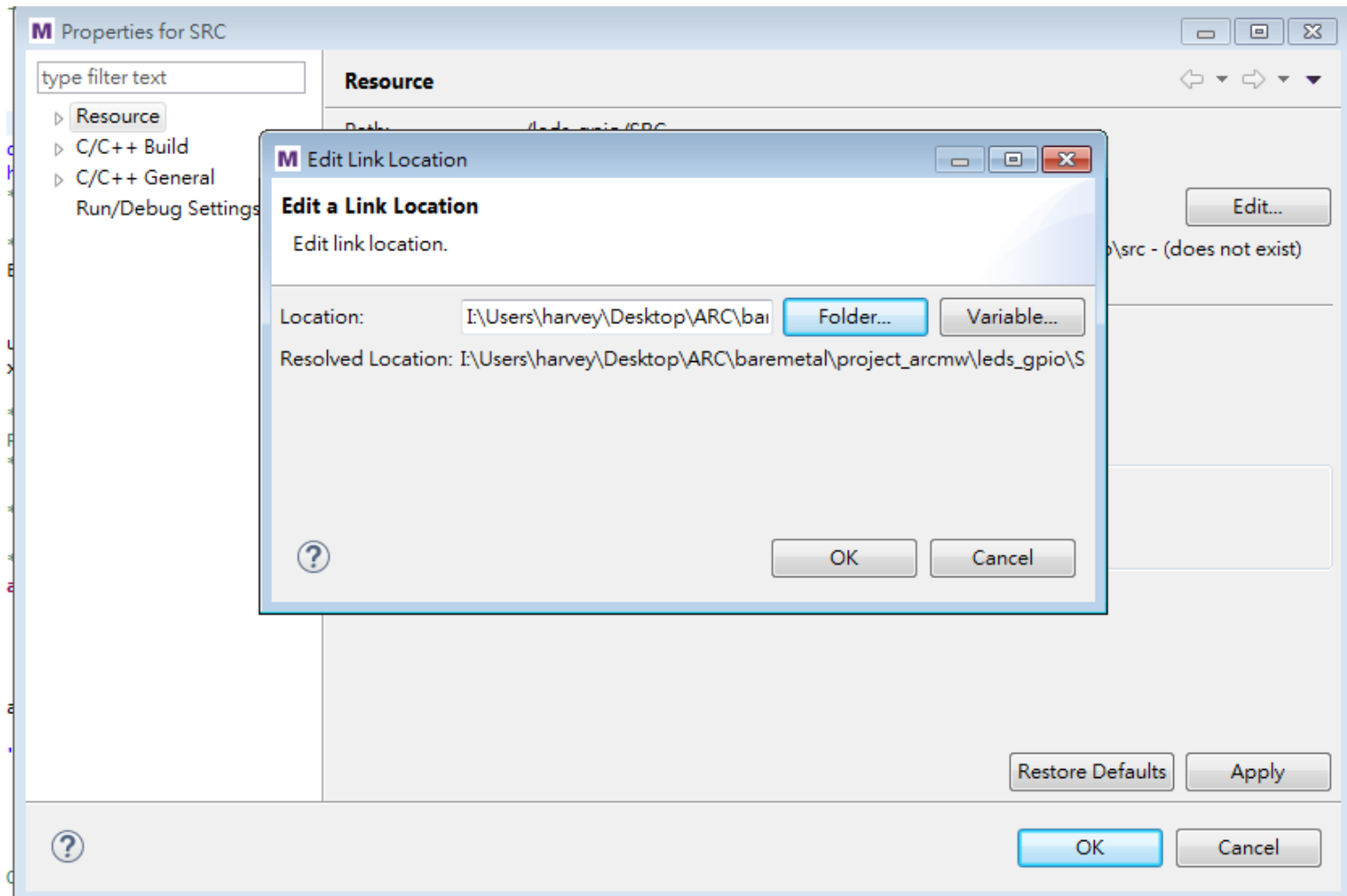
# STEP2: 點選SRC右鍵->PROPERTIES



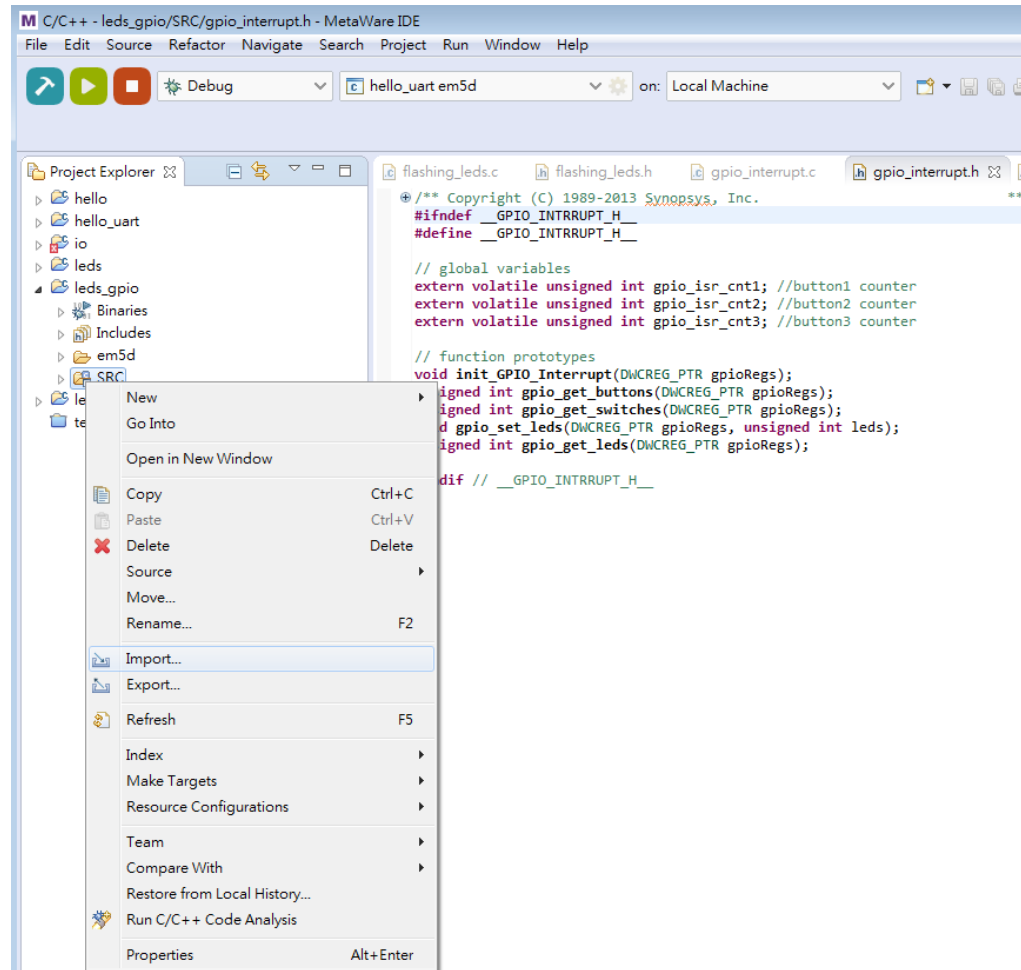
# Step3: 按下edit



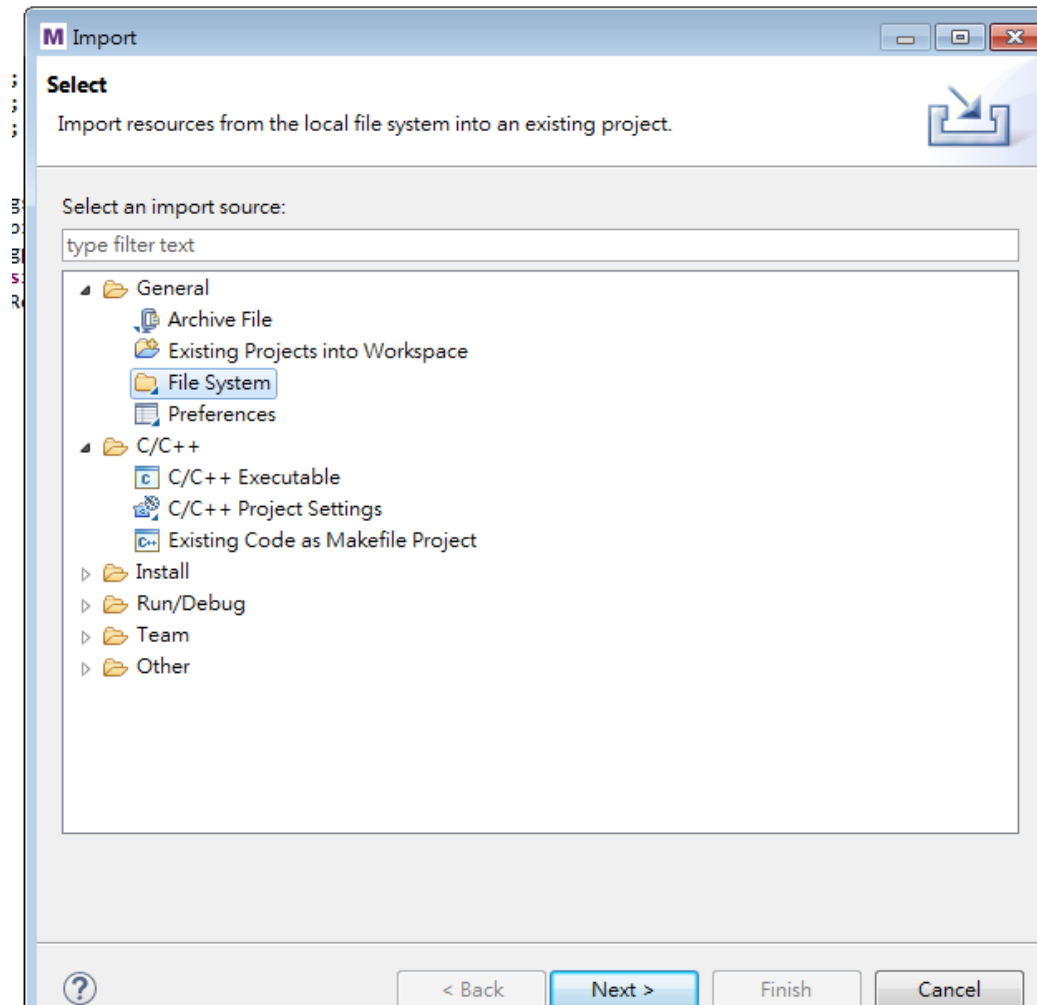
# Step4:選剛才建立的src



# Step5:import file system



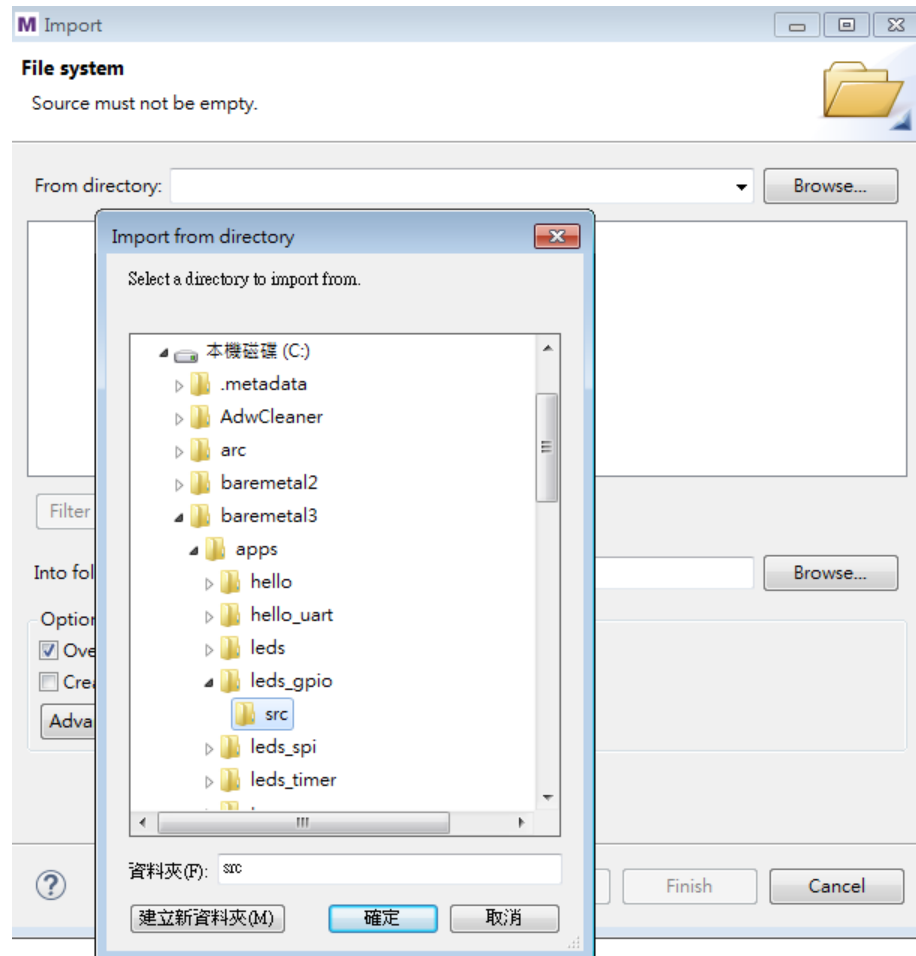
# Step6: choose file system



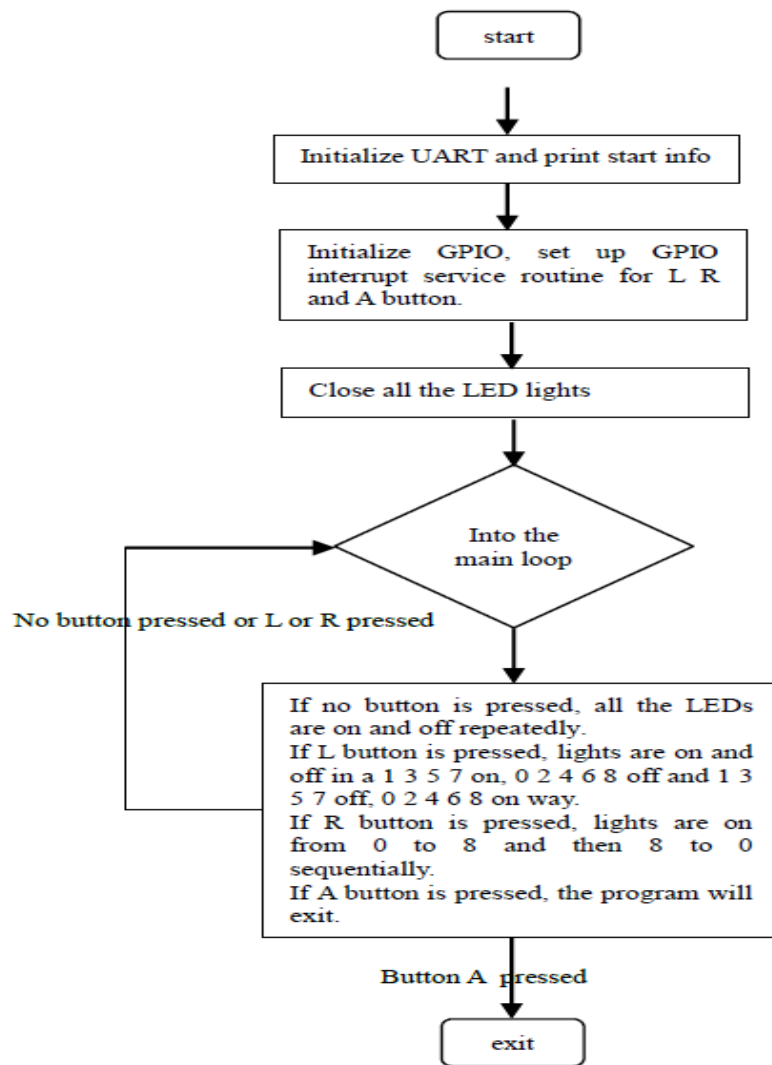


# Step7: Choose file

✦ Source path: /baremetal/apps/leds\_gpio/src



# 實作一 流程圖



# Gpio\_init

⊕ 初始化gpio的value(包括button、led、switches)

```
void gpio_init(DWCREG_PTR gpioRegs) {  
  
    //initilize buttons  
    gpioRegs[SWPORTA_DR] = 0;        //Data to PortA  
    gpioRegs[SWPORTA_DDR] = 0;      //PortA - input  
  
    //LED[7:0]  
    gpioRegs[SWPORTB_DR] = DEFAULT_LED_MASK;  
    gpioRegs[SWPORTB_DDR] = DEFAULT_LED_MASK; //PortB - output  
  
    //Switches  
    gpioRegs[SWPORTC_DR] = 0;        //Data to PortC  
    gpioRegs[SWPORTC_DDR] = 0;      //PortC - input  
}
```

# init\_GPIO\_Interrupt

⊕ 註冊gpio的interrupt,讓按下button時會觸發gpio ISR

```
⊖ /*****  
init_GPIO_Interrupt - Initialize GPIO controller and setup GPIO ISR  
*****/  
⊖ void init_GPIO_Interrupt(DWCREG_PTR gpioRegs) {  
    _clri();                // Turn off interrupts  
  
    gpioRegs[DEBOUNCE]      = 0x00;    // Debounce disable for buttons  
    gpioRegs[INTTYPE_LEVEL] = 0x07;    // edge sensitive interrupt for buttons  
    gpioRegs[INT_POLARITY]  = 0x07;    // interrupt polarity for buttons  
    gpioRegs[PORTA_EOI]     = 0xFFFF; // clear any pending interrupts  
    gpioRegs[INTEN]         = 0x07;    // interrupt enable for buttons  
  
    // save GPIO base address for access to GPIO registers from ISR  
    gpioBaseAddress = gpioRegs;  
  
    gpio_isr_cnt1 = gpio_isr_cnt2 = gpio_isr_cnt3 = 0;  
  
    // initialize interrupt vector  
    target_setvect( GPIO_INT_VECT, Gpio_ISR);  
  
    _seti(0x10);           // Enable interrupt for priority 0  
}
```

# gpio\_set\_leds

## ⊕ 設定GPIO LED腳位



Bit 8

Bit 7

Bit 0

gpioRegs[SWPORTB\_DR]控制每個LED燈亮暗  
1 是暗 0是亮

```
// switch off all LEDs  
gpio set leds(ptr, 0x01ff);
```

```
void gpio_set_leds(DWCREG_PTR gpioRegs, unsigned int leds) {  
    gpioRegs[SWPORTB_DR] = leds;  
}
```

# Exercise1: Implement the interrupt service routine

```

[ ] /*****
Gpio_ISR  -GPIO interrupt service routine; It increments gpio_isr_cnt* counter
          if corresponding button pushed.
*****/
target_interrupt void Gpio_ISR ()
[ ] {
    //read status
    unsigned int reg;
    reg = gpioBaseAddress[INTSTATUS];

    //clear interrupts
    gpioBaseAddress[PORTA_EOI] = reg;
    ///###Insert code here###
}

```

# hint

## ⊕ 1. gpioBaseAddress[INTSTATUS]為記錄gpio interrupt的狀態

按下L鈕 gpioBaseAddress[INTSTATUS]的BIT0為1

按下R鈕 gpioBaseAddress[INTSTATUS]的BIT1為1

按下A鈕 gpioBaseAddress[INTSTATUS]的BIT2為1

```
unsigned int reg;
reg = gpioBaseAddress[INTSTATUS];
```

## ⊕ P.S. C code

```
//clear interrupts
gpioBaseAddress[PORTA_EOI] = reg;
```

condition去判斷

## hint

⊕2.用三個COUNTER去判斷，當按下BUTTON L 將 gpio\_isr\_cnt1=1;以此類推，在此ISR只需紀錄是哪一個按鈕被觸發，我們把led閃爍過程移到main function去實現

```
volatile unsigned int gpio_isr_cnt1 = 0; //button1 counter  
volatile unsigned int gpio_isr_cnt2 = 0; //button2 counter  
volatile unsigned int gpio_isr_cnt3 = 0; //button3 counter
```



# Exercise2: Implement the main function

```
// switch off all LEDs
gpio_set_leds(pctr, 0x01ff);

while (1)
{
    // default behaviour
    ///###Insert code here###
    //Button "L" is pressed
    ///###Insert code here###
    //Button "R" is pressed
    ///###Insert code here###
    //Button "A" is pressed
    ///###Insert code here###
}
return 0;
```

## hint

- ⊕ 用剛才的gpio\_isr\_cnt1、gpio\_isr\_cnt2、gpio\_isr\_cnt3判斷去做相對應的動作
- ⊕ gpio\_set\_leds(pctr, 0x0); 控制led燈閃爍
- ⊕ delay(PULSE\_DELAY); 利用delay讓led不要閃爍太快，可讓肉眼看到。

# 實驗結報

## ⊕ 結報格式

➤ **這次需要繳交結報**，請各位可以開始動手做期末Project。

P. S. 助教可提供的模組包括溫度感測、LCD模組。

## ⊕ TA Contact Information:

➤ 助教信箱：[a2215689@gmail.com](mailto:a2215689@gmail.com)

➤ Rm 92617

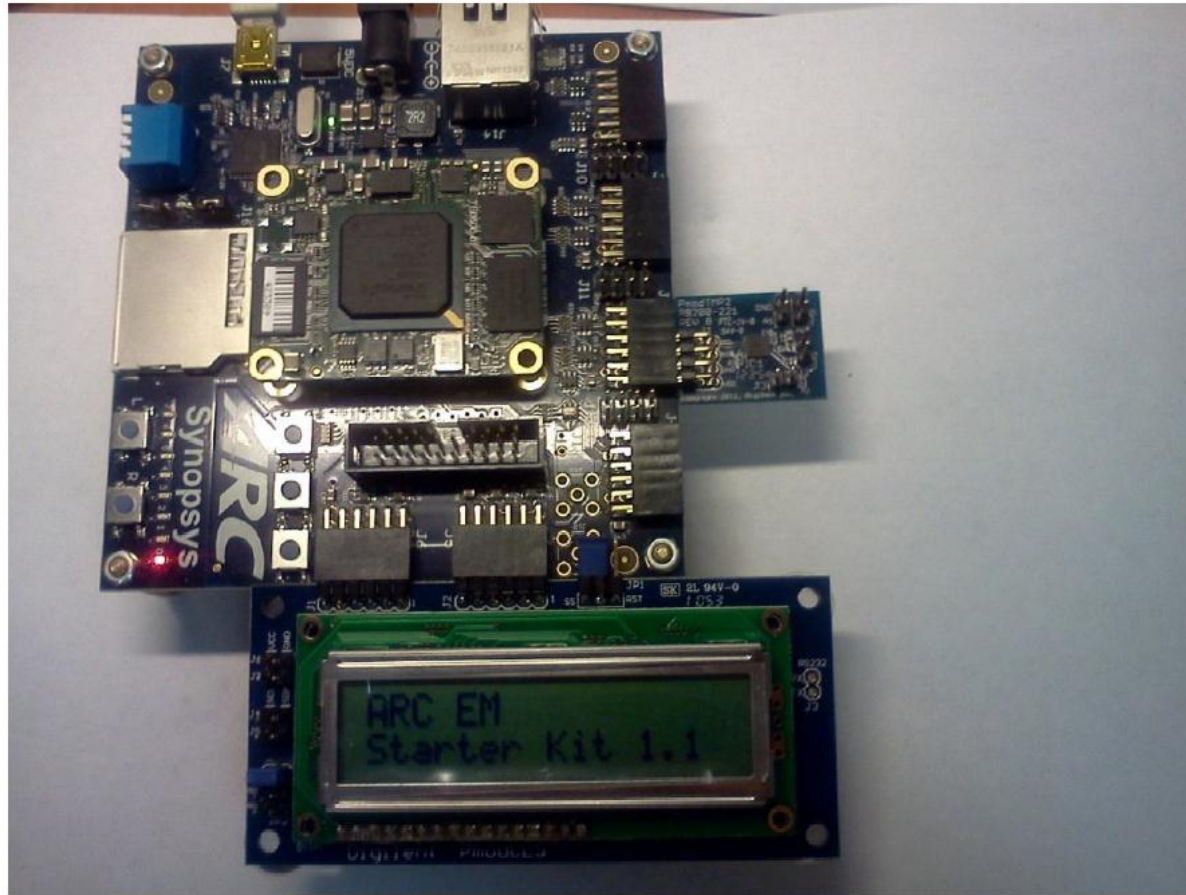
➤ Office hour：(Monday)14:00pm~16:00pm

# Appendix1:Temp sensor project

⊕希望大家在做project前，可先做此實驗，此實驗可以熟悉如何操作溫度感測器和LCD燈

⊕Reference: [Temperature\\_sensor\\_demo.pdf](#)

# Appendix1:Temp sensor LCD 接法

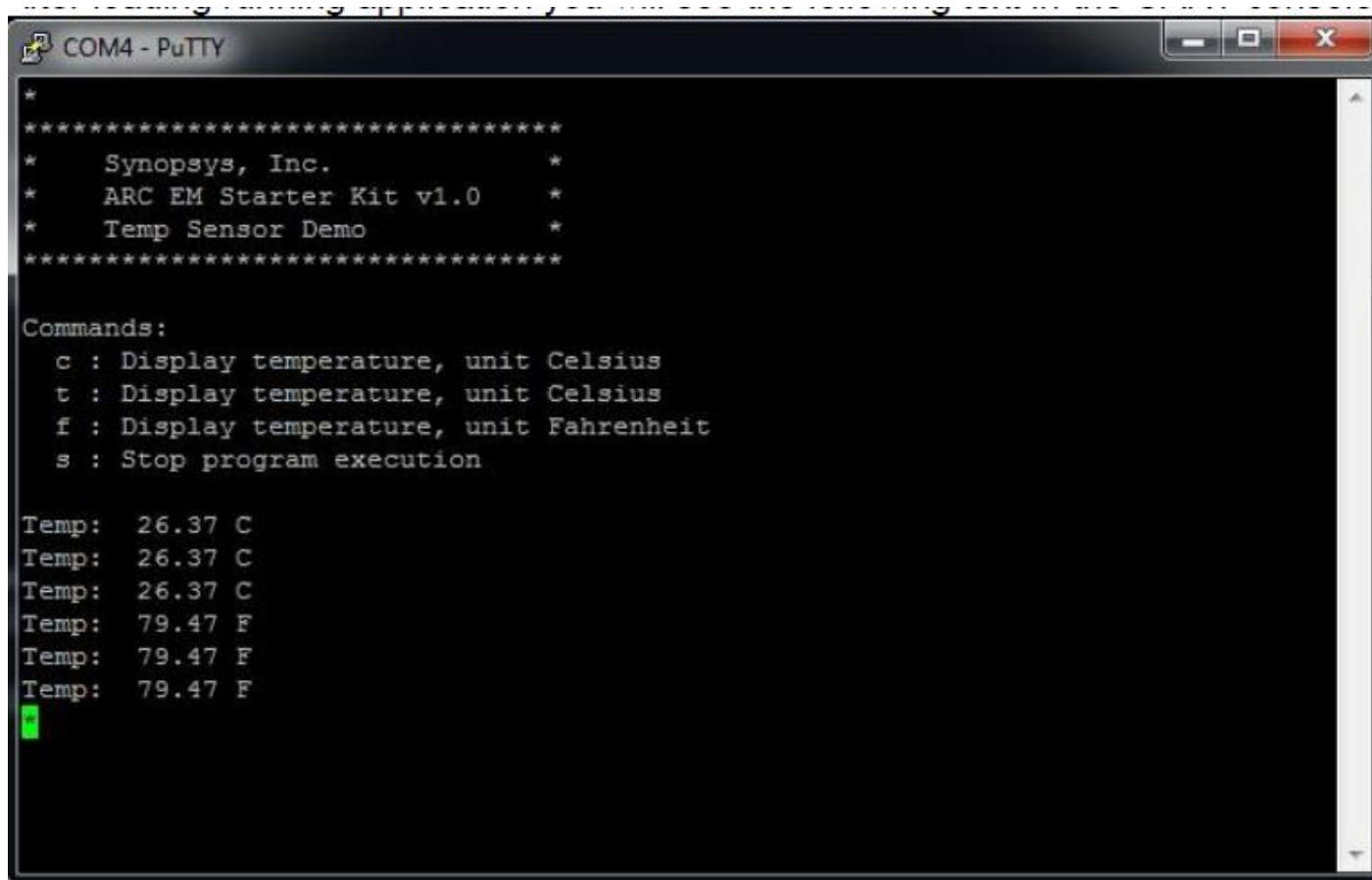


# Appendix1:Temp sensor application

⊕ 步驟與上個實驗一模一樣 run temp\_sensor project

# Appendix1:Temp sensor application

- ⊕ 按下鍵盤 c、t 可顯示攝氏溫度



```
COM4 - PuTTY
*
*****
*   Synopsys, Inc.           *
*   ARC EM Starter Kit v1.0 *
*   Temp Sensor Demo        *
*****

Commands:
  c : Display temperature, unit Celsius
  t : Display temperature, unit Celsius
  f : Display temperature, unit Fahrenheit
  s : Stop program execution

Temp: 26.37 C
Temp: 26.37 C
Temp: 26.37 C
Temp: 79.47 F
Temp: 79.47 F
Temp: 79.47 F
█
```

## Appendix2: ARC API Introduction

⊕ `int Read_Temp (int tmp_units, DWCREG_PTR console, DWCREG_PTR uart, DWCREG_PTR i2c)`

⊕ Implement in the `/apps/temp_sensor/src/temp_sensor.c`

⊕ 可看此function學習如何把溫度從I2C Temperature Sensor PmodTmp2 讀出，並且想想如何在project將它做修改以及應用



# Gpio API function

- ⊕ Implement in `baremetal/io/gpio/src/gpio.c`
- ⊕ 裡面function 包括如何初始gpio，如何設定led燈，和讀led燈、button、switch狀態

# LCD API function

- ⊕ Implement in baremetal/io/gpio/src/lcd.c
- ⊕ 裡面function包括如何初始lcd、印字串在螢幕、清除螢幕等等基本功能

# UART API function

- ⊕ Implement in baremetal/io/gpio/src/uart.c
- ⊕ 較常用的是 `void uart_print(DWCREG_PTR uartRegs, const char * pBuf)`
- ⊕ Ex: `uart_print(uart, "Hello world\n\r");` 會印 hello world 在 putty 上