Lab1~Lab5 共用規定

- 上課時間: 14:10~17:00；地點 @新館一樓 65105
- 每一個 lab 最晚都會在上課當天中午12:00前上傳投影片到 moodle，為避免教室網路訊號不好，請同學在14:00上課前先下載投影片至電腦中。
- 每一個 lab 佔總分 8%, 獨立計分. (Final Project 佔總分 60%)
- Lab 完成後, 要在 7 天內寫好 lab report 上傳moodle。
- 要來上 lab 課簽到，我們才會為你的 lab 成果評分。
- 若 lab 下課前有做完，我們會現場幫你評分。
- 若 lab 下課前沒做完，會有補交機制 (各 lab 規定方式可能不同)，期限內有完成就不會扣分 (期限為 7 天內，超過不計分)。

Lab5規定

- Lab5 補交機制 (各 lab 規定方式可能不同)
  本次lab不開放補交
- 寫lab report (上傳moodle)

Lab5 fat file system with SPI to micro SD card adaptor device

TA : 鄭煦霖、徐健翔

Email : p76131571@gs.ncku.edu.tw 、 p76131686@gs.ncku.edu.tw

Outline

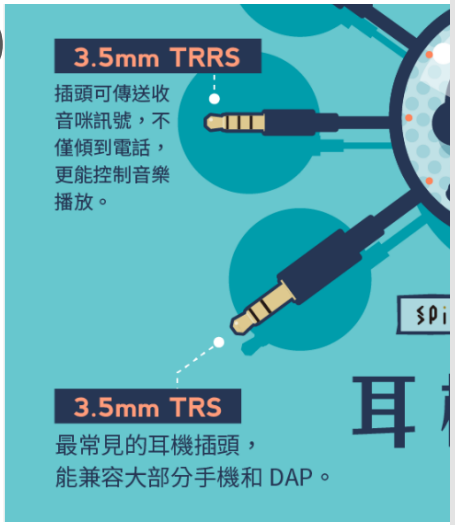
- Requirement
- Preparations
- Import the project
- Lab5
- Grading

Requirement

- 在開發板上面外接模組，讀取micro SD卡內的WAV檔案並播放
- 對micro SD卡作寫入

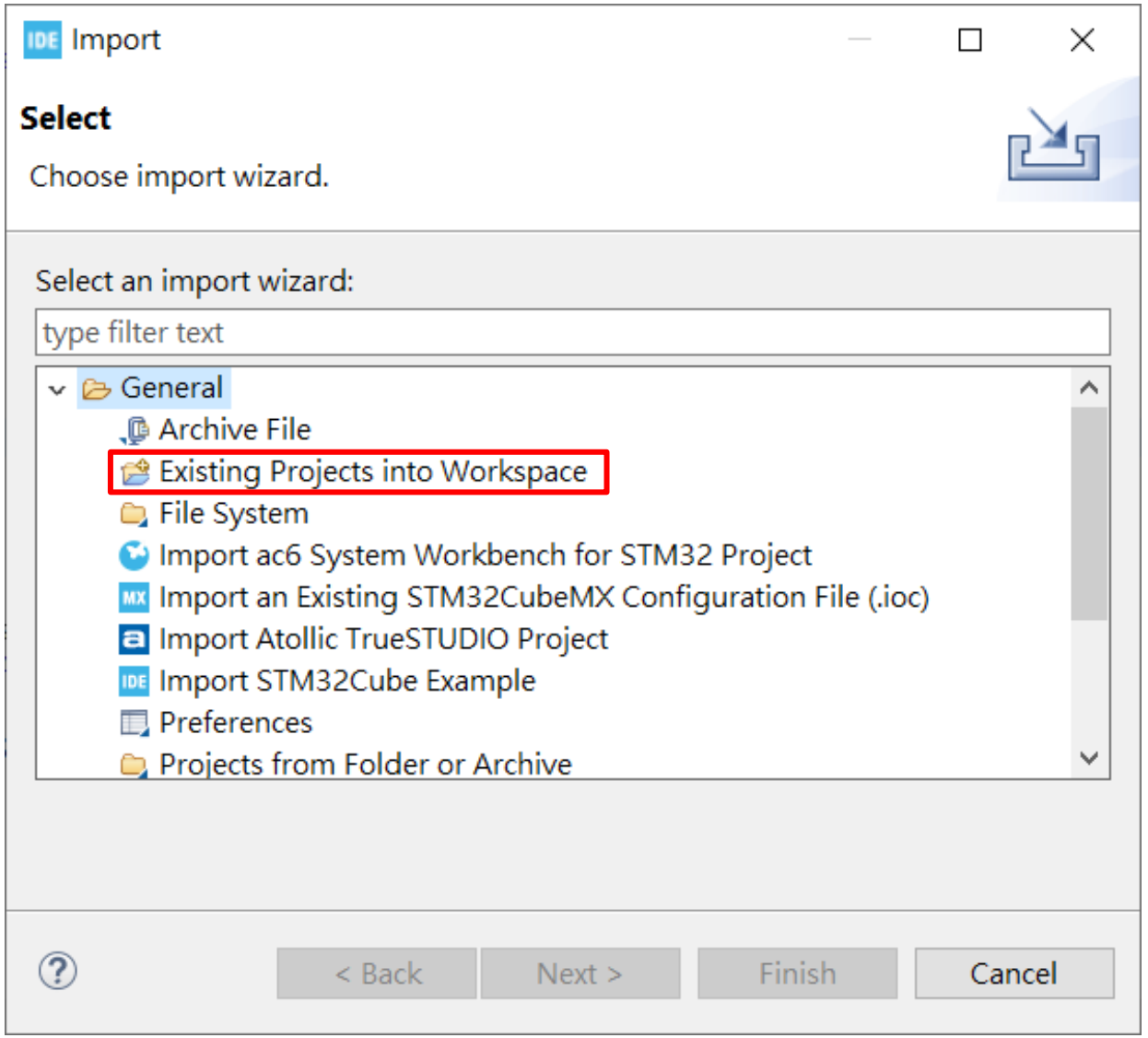
Preparations

- **ttl to usb adaptor** (USART傳輸要用)
- 32gb micro sd card(目前只測試過16gb.32gb可以work, 64gb不行，所以**不要買太大容量的**)
- **micro sd card adaptor**(要外接的模組)
- sd card reader(將檔案放入sd卡時用 如果laptop已經有讀卡機則不用)
- 有線耳機(手機用)(可有麥克風功能，右圖兩種皆可)
- 六條杜邦線(兩邊皆母) (連結spi裝置到開發板用)
- 多個.wav音樂檔案
- 同學們可以利用這網站來轉換成wav檔
- convert-to-wav



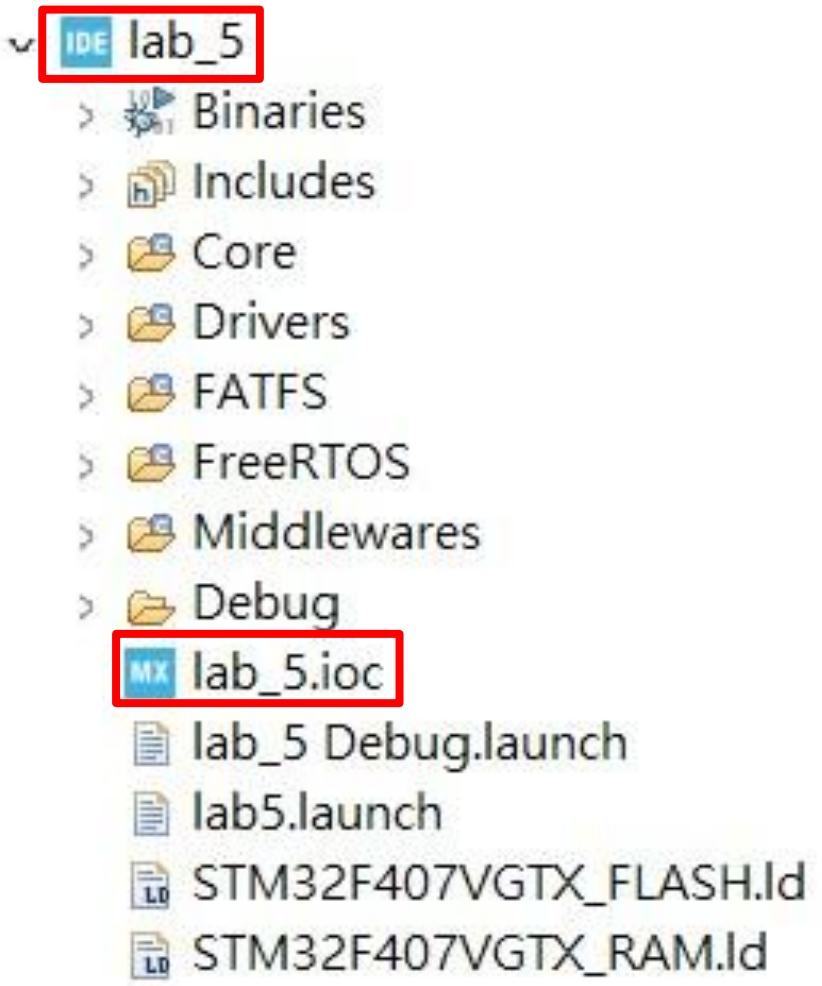
Import the project

- 這裡實作比較麻煩，需要的時間比較久一點，所以我們有先寫好一部份了。有興趣想要自己做的同學可以參考我們整理好的文件。對於這次的lab可以直接下載project下來作更改就好了
- 下載 project lab5
- 下載project 的壓縮檔案並解開 選擇 file -> import (如右圖)



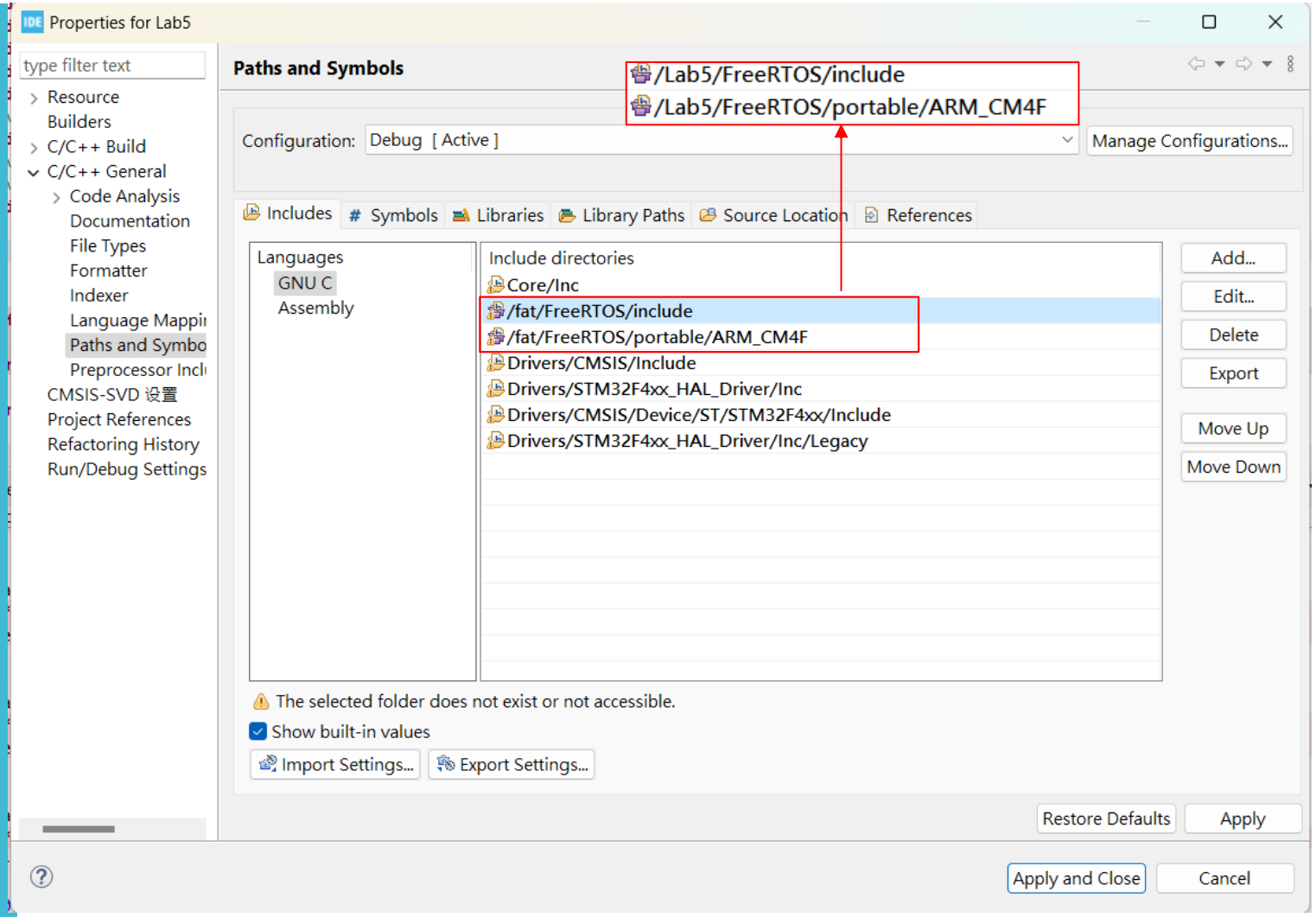
Rename project

- 若想更改專案名稱，要先更改專案資料夾名稱，再點進去更改ioc檔名(名稱都要一樣)，再import進來
- 範例: lab5 ->lab_5



Path & Symbols

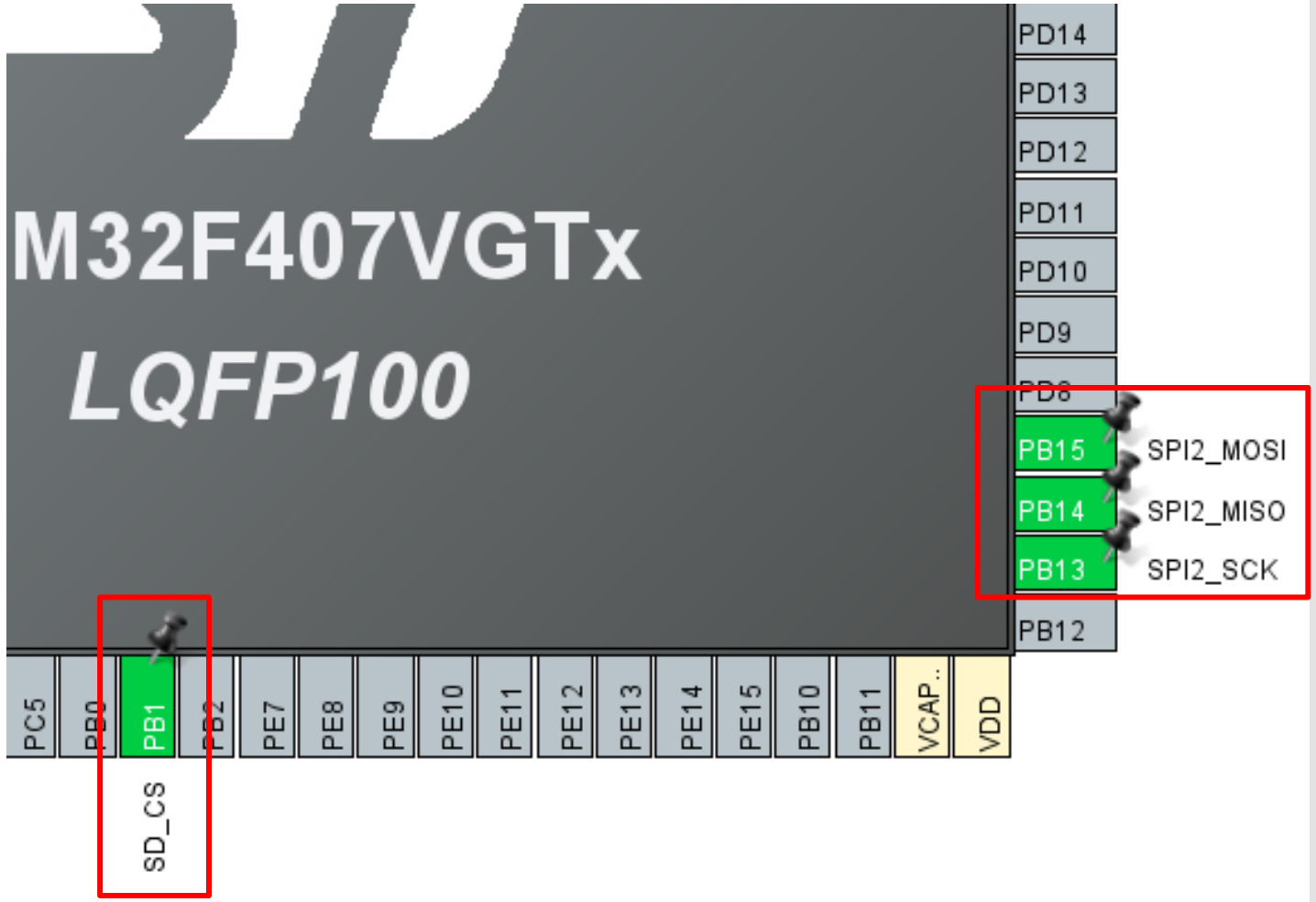
- 正常應該可以直接build，如果不行的話，檢查一下path
- 對專案檔點右鍵，選擇properties
- 再選擇Paths and Symbols
- 更改includes 裡的部分path，跟lab0一樣



Install sd card device

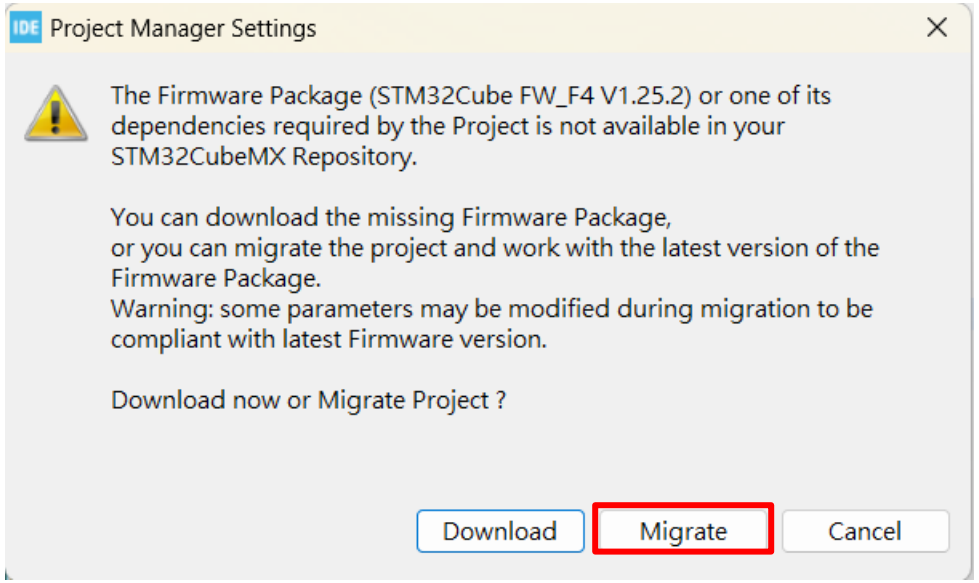
- 依照右圖和下表接線
- sd card module 上的 CS 接到 PB1
- Sd card module 上的5V 接到 5V
- GND 記得接

Name	pin
CS	PB1
SCK	PB13
MOSI	PB15
MISO	PB14
VCC	5V
GND	GND



SD和板子腳位的對應表

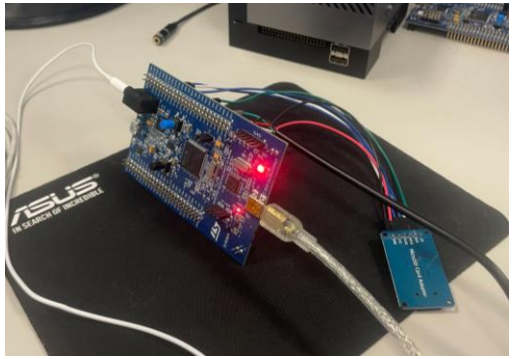
- 若開啟ioc會跑出右圖的警示，點擊“Migrate”
- 保存之後會再需要regenerate c code，一樣記得註解掉stm32f4xx_it.h跟stm32f4xx_it.c裡面的部分函式



Playback the wav file

```
SD card mounted Successfully.!
Parsing SD card for WAV files...
Found WAV file: KUI.WAV
Found WAV file: TANYA.WAV
Found WAV file: VICKY.WAV
Found WAV file: MIXER.WAV
Total WAV files found: 4
```

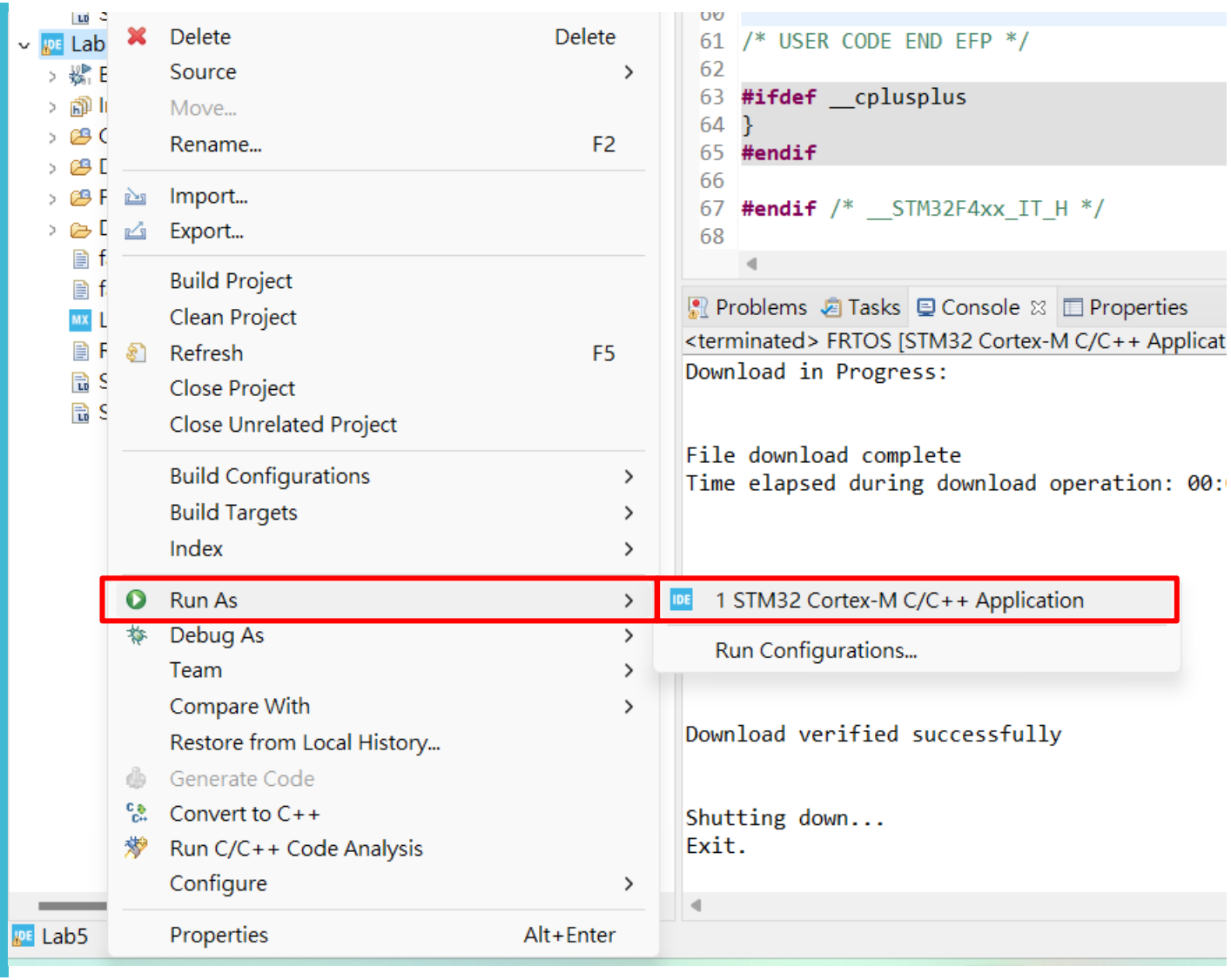
- 請先註解掉其他task，只留下AudioPlayer task就好
- 請放入多個wav檔案(建議先準備3個以上，Lab要使用到).
- 請接上耳機，會需要從耳機外放出來，結果應該要是循環撥放所有音樂，以及USART會傳輸如上圖的資訊(有正確mount到SD卡以及解析出所有的音樂檔案)，USART腳位設定如lab 2。
- 若是有播放不出來或是有雜音的狀況，可以把sd卡拔出再插入、杜邦線插緊(建議插在版子後面)，或是放到laptop的讀卡機重整
- **播放不出來的請先自己檢查，也有可能是SD卡或是外接模組型號的問題。請先確定播的出來再往下做。**



執行 project(第一次)

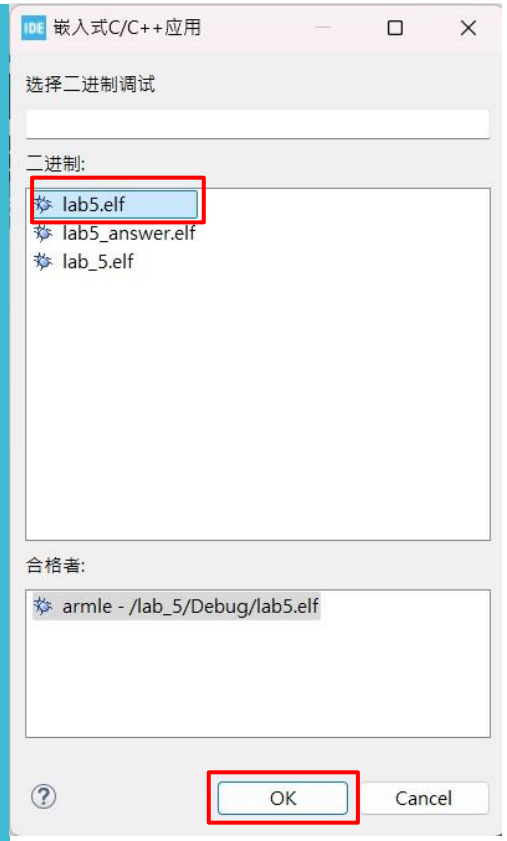
- 對專案點右鍵，選擇

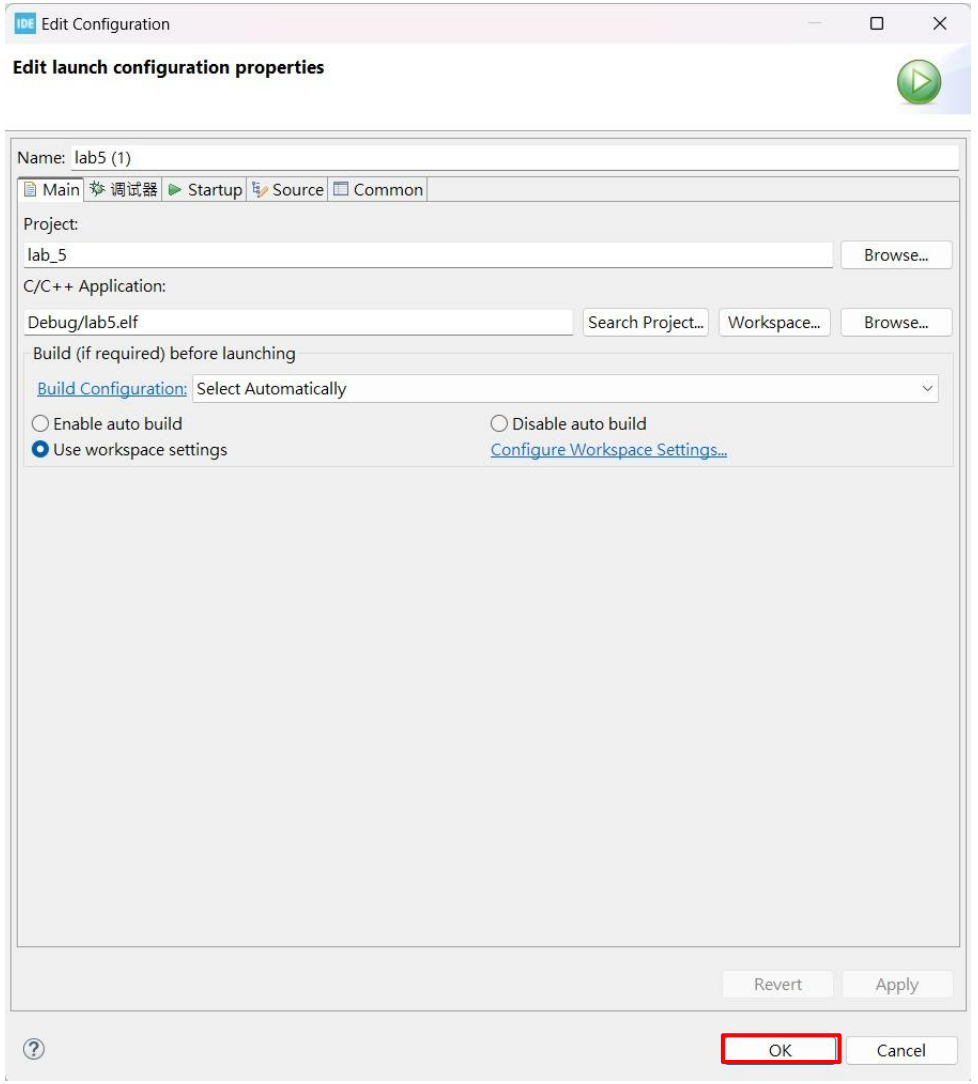
Run As -> STM32 Cortex-M C/C++ Application



執行 project(第一次)

- 接下來會跑出左邊的視窗，選擇lab5.elf，再點擊ok
- 之後會出現右圖，點擊ok即可
- 以後要跑就可以照原本的方式就好了~





若是會遇到**音樂撥到一半斷掉**的情況，可以嘗試調低品質和採樣率(板子可能對高品質音效處理不夠快)

Button task

- 當藍色按鈕被按下時，會觸發中斷，執行 callback function 並通知 button task。
- button task 內會用到GetButtonEvent() 這個函式，會回傳這次user是單擊、雙擊還是長按，挖空的地方需要完成這個判斷。

```
eButtonEvent getButtonEvent()
{
    static const uint32_t DOUBLE_GAP_MILLIS_MAX = 250;
    static const uint32_t LONG_MILLIS_MIN = 800;

    static uint32_t button_down_ts = 0;
    static uint32_t button_up_ts = 0;
    static bool double_pending = false;
    static bool long_press_pending = false;
    static bool button_down = false;
    static bool long_press_fired = false;

    static eButtonEvent pending_event = NO_PRESS;
    uint32_t now = HAL_GetTick();

    /*write your code here*/
    /*Determine whether it's a single press, double press, or long press.*/

    eButtonEvent event_to_return = pending_event;
    pending_event = NO_PRESS;
    return event_to_return;
}
```

Button task

- 完成GetButtonEvent() 之後，根據按下的模式，做出相對應的事情(根據下一頁的流程圖)。

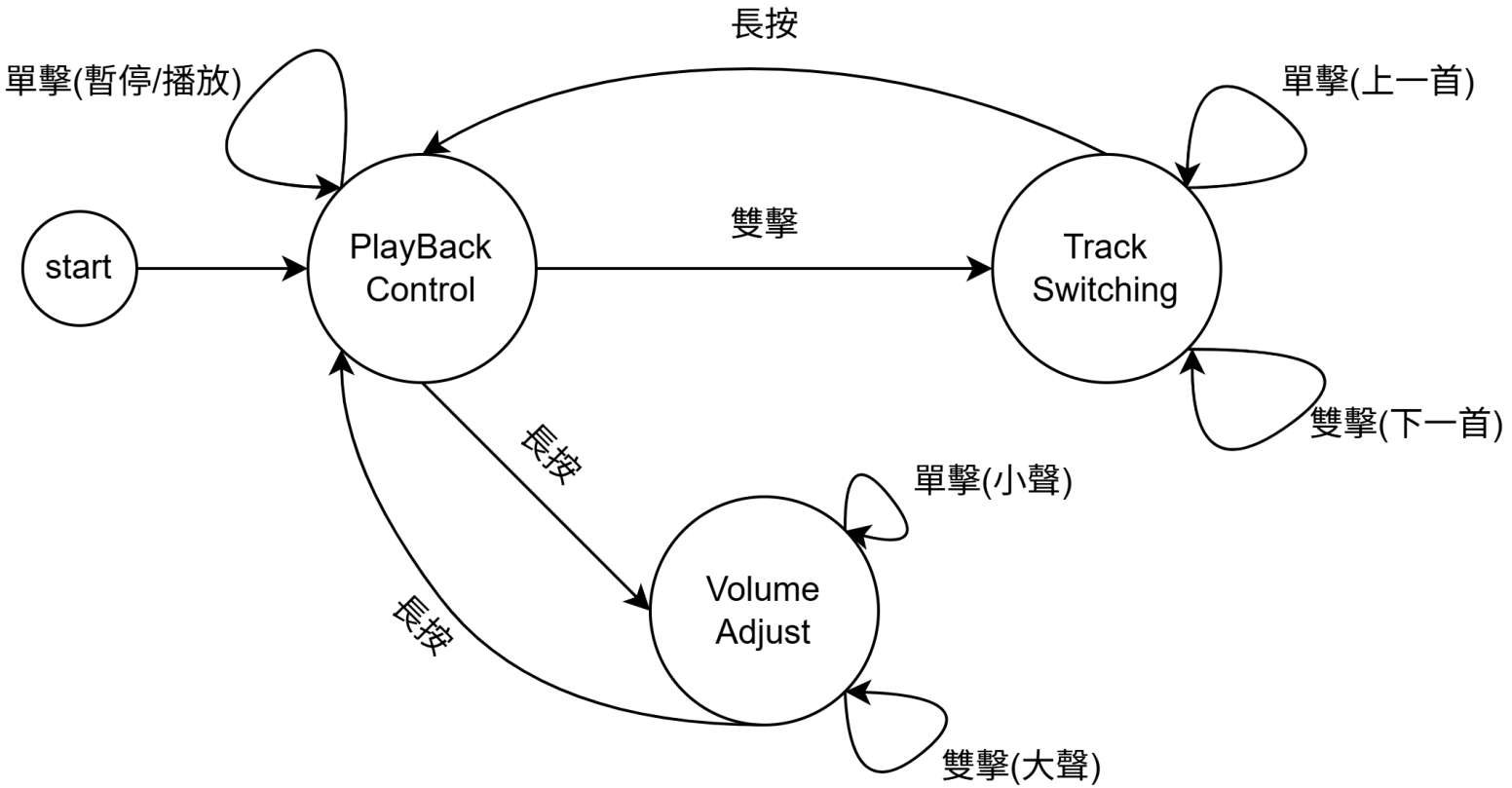
```
void ButtonTask(void *pvParameters){
  while (1){
    xTaskNotifyWait(0, 0, NULL, pdMS_TO_TICKS(20));

    eButtonEvent event = getButtonEvent(); →知道按下的模式
    if (event != NO_PRESS) {
      switch (currentState) {
        case PLAYBACK_CONTROL:
          if (event == SINGLE_PRESS) {
            myprintf("PLAYBACK_CONTROL: Single Press Detected. Toggling Play/Pause.\r\n");

            if (AudioState == AUDIO_STATE_PLAY){
              LogOperation("AUDIO_PAUSE\r\n");
              //Write your code here
            }
            if (AudioState == AUDIO_STATE_WAIT){
              LogOperation("AUDIO_RESUME\r\n");
              //Write your code here
            }
          }
          else if (event == DOUBLE_PRESS) {
            myprintf("PLAYBACK_CONTROL: Double Press Detected. Entering Track Switching Mode.\r\n");
            //Write your code here
          }
          else if (event == LONG_PRESS){
            myprintf("PLAYBACK_CONTROL: Long Press Detected. Entering Volume Adjust Mode.\r\n");
            //Write your code here
```

↱單擊要做的事

Button task


長按
單擊(暫停/播放)
單擊(上一首)
start
PlayBack Control
雙擊
Track Switching
雙擊(下一首)
長按
單擊(小聲)
Volume Adjust
長按
雙擊(大聲)

Button task

- 根據欲作的操作，去更改變數AudioState的值。定義如下:

```
typedef enum {
  AUDIO_STATE_IDLE = 0,
  AUDIO_STATE_WAIT,
  AUDIO_STATE_INIT,
  AUDIO_STATE_PLAY,
  AUDIO_STATE_RECORD,
  AUDIO_STATE_NEXT,        ->下一首
  AUDIO_STATE_PREVIOUS,    ->上一首
  AUDIO_STATE_FORWARD,
  AUDIO_STATE_BACKWARD,
  AUDIO_STATE_STOP,
  AUDIO_STATE_PAUSE,       ->暫停
  AUDIO_STATE_RESUME,      ->恢復播放
  AUDIO_STATE_VOLUME_UP,   ->音量增加
  AUDIO_STATE_VOLUME_DOWN, ->音量減少
  AUDIO_STATE_ERROR,
}AUDIO_PLAYBACK_StateTypeDef;
```

Log task

- Log task 會先開啟 SD card 內的 log.txt，並清空它。如果檔案不在，就創建一個新的。
- 每當按下按鈕，有操作時，button task內會用LogOperation()函式把要記錄的字串傳過來，此時log task就要把這行字串寫進去log檔裡，最後用PrintLogFile()輸出現在log檔內的所有內容。
- 用f_open, f_close() (), f_read(), f_write()函式去實作，傳入的參數要注意。

Log task

- 要印出來的東西如下，看你做了什麼操作，log file就會記錄什麼操作

```
LogTask: Wrote log entry.
Log file contents:
AUDIO_PAUSE
AUDIO_RESUME
VOLUMN_DOWN
VOLUMN_UP
PREVIOUS_SONG
NEXT_SONG
```

Lab5 grading

- (2%)分辨出單擊.雙擊和長按
- (1%)按鈕按下所對應的功能都正確
- (2%)完成log檔的開檔.讀檔及寫入
- (3%) Lab report(一定要交)