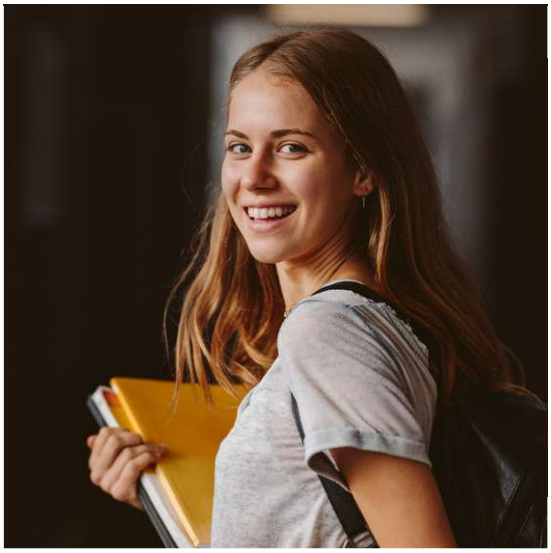
音樂撥放器



控制二乙
09
賴品媛
指導老師:張漢民

大安高級工業職業學校

完成日期 2025.02.27

目錄

摘要

這堂課要結合以前學的按鈕開關LED和蜂鳴器，利用以前的所學加上今天新寫的內容，才能夠順利完成這次實習作業，因此我們除了要學會這次tone()的程式碼，更要能把上次按鈕開關和LED應用出來。

動機

學習按鈕控制： 透過 OneButton Library，學習如何使用按鈕的單擊和長按事件。
音樂播放： 學習如何使用 Arduino 控制蜂鳴器播放音樂。
LED 指示燈應用： 學習如何使用 LED 指示燈顯示程式的運行狀態。
整合應用： 將按鈕控制、音樂播放和 LED 指示燈整合到一個專案中，提高綜合應用能力。
實際應用： 將所學知識應用於實際專案，例如製作一個簡單的音樂播放器。

總說

這份作業不僅僅是一個簡單的練習，它是一個綜合性的學習過程，涵蓋了硬體接線、軟體編程、庫的使用、狀態機設計等多個方面。通過完成這個作業，我們可以為未來更複雜的項目打下堅實的基礎，並提升自己的技術能力。

主題

1. 主題核心

- 硬體與軟體的整合：利用Arduino開發板連接按鈕開關、蜂鳴器和LED指示燈，並通過OneButton Library實現按鈕操作
- 音樂播放控制：通過按鈕的單擊和長按操作，實現音樂的播放、暫停和重播功能
- 視覺與聽覺反饋：使用LED指示燈的狀態來反映音樂的播放狀態

2. 功能描述

- 開機初始化：系統啟動時，音樂處於停止狀態，LED指示燈保持常亮
- 單擊按鈕：每次單擊按鈕，音樂會在播放與停止之間切換
- 長按按鈕：長按按鈕時，音樂會從頭開始播放。
- LED狀態指示：當音樂播放時，LED指示燈會閃爍。

當音樂停止時，LED指示燈停止閃爍並保持常亮。

3. 技術實現

- 硬體部分：
  - 使用按鈕開關作為輸入設備，控制音樂的播放狀態。
  - 使用蜂鳴器作為輸出設備，播放簡單的音樂。
  - 使用LED指示燈作為狀態顯示設備，提供視覺反饋。
- 軟體部分：
  - 使用OneButton Library檢測按鈕的單擊和長按事件。
  - 使用Arduino的tone()函數控制蜂鳴器播放音樂。
  - 使用millis()函數實現非阻塞的LED閃爍效果。

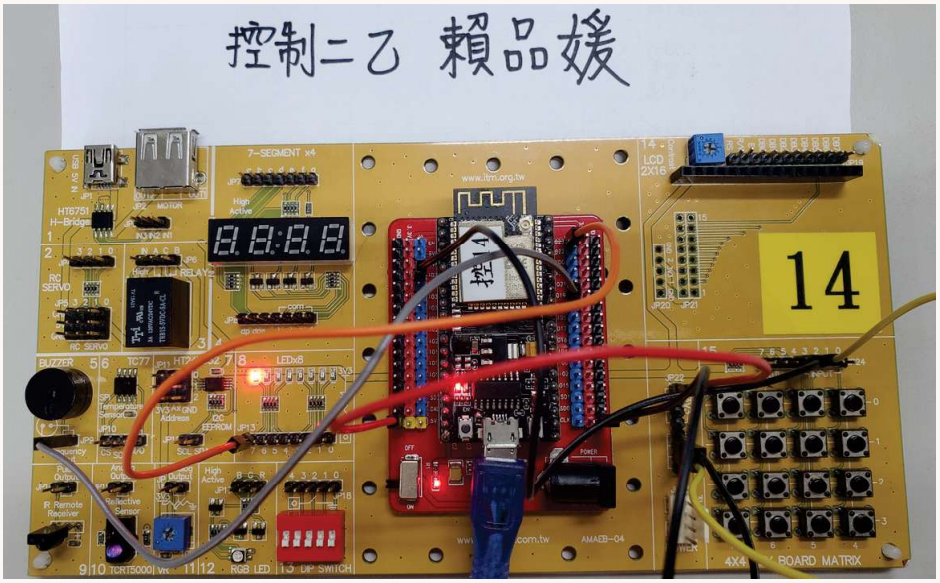
實習步驟

硬體接線

蜂鳴器

按鈕開關

LED



ESP32接線

3V3----------5V

GND----------GND

按鈕開關

INPUT3----------GND

ONPUT4----------34

LED

LED7----------23

蜂鳴器

蜂鳴器----------19

實習步驟
程式設計

```
// 低音區（C3 ~ B3）
#define C3  131
#define Cs3 139  // C#3 / Db3
#define D3  147
#define Ds3 156  // D#3 / Eb3
#define E3  165
#define F3  175
#define Fs3 185  // F#3 / Gb3
#define G3  196
#define Gs3 208  // G#3 / Ab3
#define A3  220
#define As3 233  // A#3 / Bb3
#define B3  247

// 中音區（C4 ~ B4，包含標準 C4）
#define C4  262
#define Cs4 277  // C#4 / Db4
#define D4  294
#define Ds4 311  // D#4 / Eb4
#define E4  330
#define F4  349
#define Fs4 370  // F#4 / Gb4
#define G4  392
#define Gs4 415  // G#4 / Ab4
#define A4  440
#define As4 466  // A#4 / Bb4
#define B4  494

// 休止符
#define REST 0  // 無聲音

// 音符序列
int melody[] = {D3,G3,B3,A3,G3,  B3,A3,G3,E3,D3,D3,G3,B3,A3,G3,
B3,A3,B3,D4,REST,B3,D4,B3,D4,B3,G3,D3,E3,G3,G3,E3,D3,D3,G3,B3,
G3,A3,B3,A3,G3,D3,G3,B3,A3,G3,B3,A3,G3,E3,D3,D3,G3,B3,A3,G3,
B3,A3,B3,D4,REST, B3,D4,B3,D4,B3,G3,D3,E3,G3,G3,E3,D3,D3,G3,B3,
G3,A3,B3,A3,G3 };
```

```
// 每個音符的持續時間，單位是 base_duration 的倍數
float lengths[] = {2,4,1,1,1,4,2,4,2,4,2,4,1,1,1,4,2,1,8,2,2,4,1,1,1,
4,2,4,1,1,1,4,2,4,1,1,1,4,2,8,2,4,1,1,1,4,2,4,2,4,2,4,1,1,1,
4,2,1,8,2,2,4,1,1,1,4,2,4,1,1,1,4,2,4,1,1,1,4,2,8};

const int ledPin = 23;                  // LED接腳
unsigned long previousMillisa = 0;      // 前一次的millis()時間

const long interval = 1000;             // 預設計時的時間
bool state = LOW;                       // LED狀態，初始值為HIGH（OFF）

int pin = 19;    // 蜂鳴器接腳
int base_duration = 200;  // 音符基準時間
int note_interval = 100;  // 間隔時間

int i = 0;  // 當前播放的音符索引
unsigned long previousMillis = 0;  // 上一次切換音符的時間
bool isPlaying = false;            // 是否正在播放音符
int note_duration;                 // 當前音符的持續時間

#include <OneButton.h>          // 引用OneButton函式庫
#define pb 34                   // 定義按鈕引腳
OneButton button(pb, true);     // 建立OneButton物件，名稱為button，按鈕為低態動
作
boolean startPlay = 0;          // 0：停止播放，1：開始播放

void setup() {
  pinMode(pin, OUTPUT);  // 宣告蜂鳴器接腳為輸出

  Serial.begin(9600);  // 啟用串列埠監看視窗

  button.attachClick(singleClick);                // 單擊
  button.attachDoubleClick(doubleClick);          // 雙擊
  button.attachLongPressStart(longPressStart);    // 長按開始
  button.attachDuringLongPress(longPress);        // 長按過程中
  button.attachLongPressStop(longPressStop);      // 長按結束

  pinMode(ledPin, OUTPUT);            // 宣告ledPin為輸出腳
  digitalWrite(ledPin, 0);        // 輸出狀態
}

void loop() {
   button.tick(); // 定期偵測按鈕狀態

  unsigned long currentMillis = millis();  // 取得當前時間
  unsigned long currentMillisa = millis();          // 當前的millis()時間

  if (startPlay == 1)  // 開始播放
  {
    if (currentMillisa - previousMillisa >= interval)  // 若達到預設計時的時間
    {
```

```
        // 每經過一個interval的時間，要做的事
  //---------------------------------------------------------------------------
        state = !state;                     // 改變LED狀態
        digitalWrite(ledPin, state);        // 輸出狀態
  //---------------------------------------------------------------------------
        previousMillisa = currentMillisa;   // 更新前一次的millis()時間
      }
  //---------------------------------------------------------------------------
      if (i < sizeof(melody) / sizeof(int))     // 檢查是否還有音符未播放
      {
        note_duration = base_duration * lengths[i];    // 當前音符的持續時間

        if (!isPlaying) // 若處於"未播放"狀態，則開始播放當前音符
        {
          tone(pin, melody[i], note_duration);  // 發出聲音
          previousMillis = currentMillis;       // 記錄開始播放的時間
          isPlaying = true;                     // 標記為正在播放
        }
        else if (currentMillis - previousMillis >= note_duration +
  note_interval)
        {
          // 當前音符播放完畢，切換到下一個音符
          noTone(pin);          // 停止發聲
          i++;                  // 切換到下一個音符
          isPlaying = false;    // 標記為未播放
        }
      }
      else  // 曲目播放完畢
      {
        startPlay = 0;  // 停止播放
        i = 0;          // 重置音符索引
        previousMillis = currentMillis;  // 重置時間
        digitalWrite(ledPin, 0);      // 輸出狀態
      }

    }

      else if (startPlay == 0)
      {
        digitalWrite(ledPin, 0);      // 輸出狀態
      }

    }

  // 單擊處理

void singleClick() {
  Serial.println("Button Single Click.");
  startPlay = !startPlay;     // 開始播放或停止播放
}
```

```
// 雙擊處理
void doubleClick() {
  Serial.println("Button Double Click.");
}

// 長按開始
void longPressStart() {
  Serial.println("Button Long Press Start.");
}

// 長按過程中
void longPress() {
  Serial.println("Button Long Press ......");
  startPlay = 0;  // 停止播放
  isPlaying = 0;  // 標記為未播放
}

// 長按結束
void longPressStop() {
  Serial.println("Button Long Press Stop.");
  i = 0;          // 重置音符索引
}
```

程式碼說明

音符定義：

- 使用 #define 定義低音區和中音區的音符頻率。
- 定義休止符REST為0。

音樂資料：

- melody[]陣列儲存音符序列。
- lengths[]陣列儲存每個音符的持續時間。

硬體接腳：

- ledPin： LED 接腳。
- pin： 蜂鳴器接腳。
- pb：按鈕接腳。

狀態變數：

- startPlay：控制音樂播放狀態。
- isPlaying：標記當前是否正在播放音符。
- i：當前播放的音符索引

按鈕事件處理：

- 使用OneButton Library 處理單擊、雙擊和長按事件。

LED閃爍與音樂播放：

- 使用millis()實現非阻塞的LED閃爍和音樂播放控制。

實作成果

https://youtu.be/fWfRAtv7mMA?si=-JVMY4RV_OHoqIfN

↑ 點擊連結進入影片介紹

心得

我覺得這堂課最吸引我的是我們不用從頭到尾都要自己寫，可以像在拼拼圖那樣，把以前所拼好的一部分在加上今天新拼的，就能有個新的功能。過程中，我遇到了LED閃爍的問題，我按暫停時它的功能要恆亮，但它卻會根據我按按鈕當下的狀態決定亮或不亮，一開始嘗試在每一個動作內都加入LED的狀態，但結果依然沒變，經過多次嘗試，最後利用在括號外加入播放音樂時輸出為恆亮，解決了問題。

省思

這次的作業中，我原本的想法是只需要加入LED的控制，結果卻發現寫在哪裡和怎麼寫都是難題。在解決問題的過程中，我學會寫程式固然重要，但如何寫引用看得懂程式，才是真正的重點，同時詢求同學的幫助也是不二法門，透過了解對方的想法，進而想出解決問題的方法。我也發現耐心是學習不可或缺的態度，試一次不對，可以多試幾次，不放棄，成功就不會離你而去。

參考資料

https://wp.640629.xyz

https://chat.deepseek.com/a/chat/s/0b0d113f-5421-42db-8a51-acdd819e6e1d

https://gemini.google.com/app/927063e54b1e01fd?hl=zh-TW