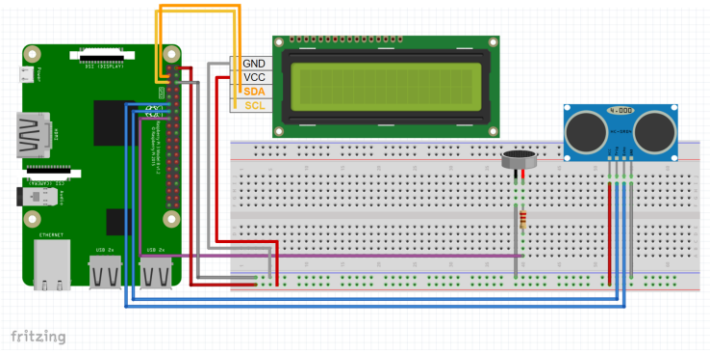
實驗三: LCD and Ultrasonic

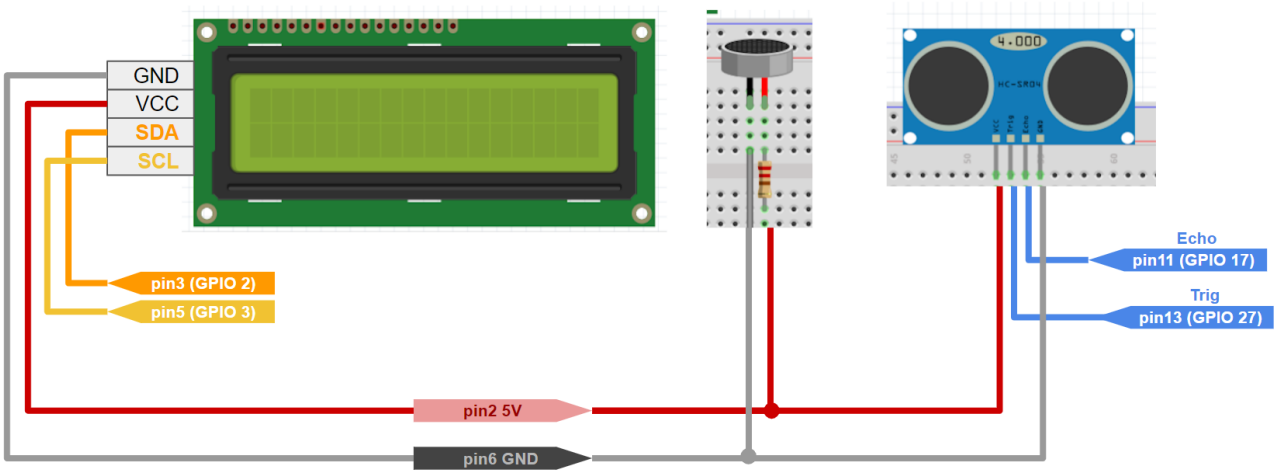
班級: 資科三甲

姓名: 謝妤婕

學號: 110816032

一、電路圖





蜂鳴器圖例使用大小較相似的麥克風代替。

二、程式

需求
1. 即時將 ultrasonic 偵測到的障礙物距離顯示在 LCD 上，包含 cm 和 inch 2. 當障礙物距離小於 5cm 播放音樂 3. ctrl+c 結束程式
程式說明
主程式包含兩條 thread: 一個即時將 ultrasonic 偵測到的障礙物距離顯示在 LCD 上、判斷距離是否大於 5 公分； 一個則是根據上述判斷結果決定是否撥放音樂或停止撥放。 完整程式碼: https://colab.research.google.com/drive/1VnnKBSRmvi67gN_Qd4MrEBAymkH_Hn0T?usp=sharing

程式(宣告變數及部分初始化)

```
import RPi.GPIO as GPIO
import sys
import smbus2
import threading
from time import time
from time import sleep
from RPLCD.i2c import CharLCD

pin ultraTrig = 27
pin ultraEcho = 17
pin buzzer = 22

sys.modules['smbus'] = smbus2
lcd = CharLCD('PCF8574', address=0x27, port=1, backlight enabled=True)
print("========== LCD setup done ==========")

musicPlay = False
stopThread = False
print("====== variable setup done ==========")

musicNoteDict = {
  "dot-Do": 262, "dot-Re": 294, "dot-Mi": 330, "dot-Fa": 349, "dot-So": 392, "dot-La": 440, "dot-Si": 494,
  "Do": 523, "Re": 587, "Mi": 659, "Fa": 698, "So": 784, "La": 880, "Si": 988,
  "Do-dot": 1046, "Re-dot": 1175, "Mi-dot": 1318, "Fa-dot": 1397, "So-dot": 1568, "La-dot": 1760, "Si-dot": 1976,
}

musicLittleStar = [
  "Do-dot", "Do-dot", "So-dot", "So-dot", "La-dot", "La-dot", "So-dot",
  "Fa-dot", "Fa-dot", "Mi-dot", "Mi-dot", "Re-dot", "Re-dot", "Do-dot",
  "So-dot", "So-dot", "Fa-dot", "Fa-dot", "Mi-dot", "Mi-dot", "Re-dot",
  "So-dot", "So-dot", "Fa-dot", "Fa-dot", "Mi-dot", "Mi-dot", "Re-dot",
  "Do-dot", "Do-dot", "So-dot", "So-dot", "La-dot", "La-dot", "So-dot",
  "Fa-dot", "Fa-dot", "Mi-dot", "Mi-dot", "Re-dot", "Re-dot", "Do-dot"
]
```

程式(除錯用的函式)

```
def printException(e, funcName):
  print("now at state %s()"%(funcName))
  print(e)
```

程式(初始化函式)

```
def stepInit():
  try:
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(pin buzzer, GPIO.OUT)
    GPIO.setup(pin ultraTrig, GPIO.OUT)
    GPIO.setup(pin ultraEcho, GPIO.IN)
    GPIO.output(pin ultraTrig, False)
    print("========== GPIO.setup done ==========")

    print("\n\n")
    print("====================================")
    print("====================================")
    print("==== press keyboard ^C to finish ====")
    print("====================================")
    print("====================================")
    print("\n\n")
  except:
    print("now at state stepInit()")
```

程式(蜂鳴器判斷條件撥放音樂) - buzzerThread 函式

```
def runBuzzer():
  try:
    while True:
      global musicPlay, stopThread
      p = GPIO.PWM(pin buzzer, 100)
      p.start(80)
      for note in musicLittleStar:
        if (not musicPlay) or stopThread:
          p.stop()
          if stopThread:
            print('======= buzzer thread stopped =======')
            return "Thread end"
          break
        p.ChangeFrequency(musicNoteDict[note])
        sleep(0.2)
      p.stop()
  except Exception as exception:
    printException(e=exception, funcName="runBuzzer")
    return "Thread end"
```

程式(LCD 顯示距離數值)

```
def runLCD(d cm, d in):
  try:
    lcd.clear()
    lcd.cursor pos = (0, 0)
    lcd.write string(" "+str(d cm)+" cm")
    lcd.cursor_pos = (1, 0)
    lcd.write string(" "+str(d in)+" in")
  except Exception as exception:
    printException(e=exception, funcName="runLCD")
    return "Thread end"
```

程式(超音波收發)

```
def ultrasonicTrig():
  try:
    GPIO.output(pin ultraTrig, True)
    sleep(0.001)
    GPIO.output(pin ultraTrig, False)
  except Exception as exception:
    printException(e=exception, funcName="ultrasonicTrig")


def ultrasonicEcho(value, timeout):
  try:
    count = timeout
    while GPIO.input(pin_ultraEcho) != value and count > 0:
      count = count - 1
  except Exception as exception:
    printException(e=exception, funcName="ultrasonicEcho")
```

程式(計算距離、呼叫 LCD 函式) - ultraThread 函式

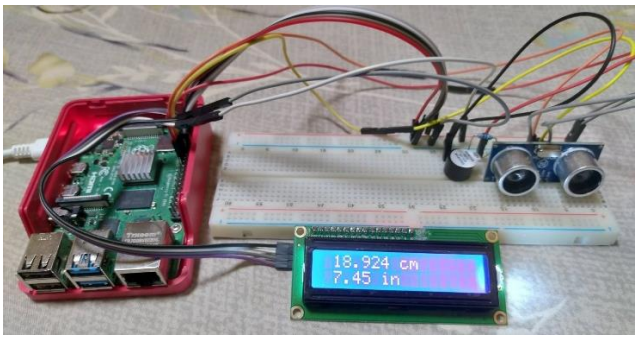
```
def ultrasonicDistance():
  while True:
    try:
      if stopThread:
        lcd.clear()
        print('======= ultra thread stopped ========')
        return "Thread end"
      ultrasonicTrig()
      ultrasonicEcho(True, 5000)
      start = time()
      ultrasonicEcho(False, 5000)
      end = time()
      distance_cm = round(((end-start) * 34000 / 2), 3)
      distance_in = round(distance_cm*0.3937, 3)
      runLCD(distance_cm, distance_in)
      global musicPlay
      if distance_cm > 5:
        musicPlay = False
      else:
        musicPlay = True
      sleep(0.5)
    except Exception as exception:
      printException(e=exception, funcName=ultrasonicDistance)
```

程式(主函式)

```
def main():
  stepInit()
  try:
    buzzerThread = threading.Thread(target=runBuzzer)
    buzzerThread.start()
    ultraThread = threading.Thread(target=ultrasonicDistance)
    ultraThread.start()
    while True:
      print("raspi working...")
      sleep(1)
  except KeyboardInterrupt:
    print("\n\n\n")
    print("============= user STOP =============")
  except Exception as exception:
    print("\n\n\n")
    print("========== other Exception ==========")
    printException(e=exception, funcName="main")
  finally:
    global stopThread
    stopThread = True
    GPIO.cleanup()
    print("========== GPIO cleanup done ========")
    print("============ end of main() ==========")


main()
```

三、影片連結



YouTube 影片連結 ： https://youtu.be/DvN1YU5uvRo

四、困難

第一個遇到的困難是在執行 *sudo i2cdetect -y 1* 的時候**找不到 i2c 的位址**，如附圖一個都沒有(朋友則是每個都有)。在老師的建議下把線拔掉重插後就正常了(朋友的也是 0u0)，猜測可能是接觸不良或沒有獻祭綠色乖乖給樹梅派。

~~第二個遇到的困難是蜂鳴器的聲音很虛，差點以為他確診了。~~

第二個遇到的困難是讓**無窮迴圈 thread 在主函式被使用者以 *ctrl+C* 中斷後能正常結束**。

一開始我採用「將蜂鳴器函式交給 thread、超音波測距函式由主函式呼叫」的方法執行，測距小於五公分會更新全域變數布林值 musicPlay 為 True(相反則為 False)，而蜂鳴器函式則會透過這個變數決定是否撥放音樂或從樂譜 for 迴圈中 break。但結果就是 *ctrl+C* 能終止超音波，但 LCD 會拋出例外出現亂碼、蜂鳴器函式依舊無法終止，整個程式會跟殭屍一樣失控，最後只能強行斷電(或運氣好連續好幾次 *ctrl+C* 可以終止)。

我首先決定處理的是讓蜂鳴器函式終止的問題，因為元件個別控制開 thread 是必然的。再上網找了各種關於 python 提供的 thread 函式後，我得到了非常不幸的消息，python 並沒有提供直接結束 thread 的方法，只能自己寫一個 class 定義方法，或是利用 join 暫停主函式等 thread 結束再正常結束整個程式(當然還有強制 kill，但副作用太多我決定不嘗試)。

前者由於我在新的.py 檔隨意寫的 class 總是無法正常運作，所以這個解法留到以後研究；後者的作法我嘗試後的結果就是世紀大悲劇，不論主函式或 thread 都結束不了，整個程式跑到升天。

(當時參考的網站: https://www.geeksforgeeks.org/python-different-ways-to-kill-a-thread/)

最後我放棄找資料了，直接回歸原本的問題，因為 thread 是執行指定的函式，所以讓函式正常結束也就能讓 thread 正常結束，而讓無窮迴圈結束不外乎就是指定條件讓他 break，但由於在這個函式裡有兩個迴圈，為了提升程式可讀性，這裡我採用的不是這個方法，而是「能讓任何函式都乖乖結束的 return」。

再來要解決的是 LCD 出現亂碼的問題，我猜想應該是 *ctrl+C* 時 LCD 剛好顯示到一半被強迫終止，只要先確保 lcd.clear 有確實執行再結束程式就行了。

就當我以為一切都皆大歡喜時，悲劇來了，程式執行下去 *ctrl+C* 會讓主函式和 thread 停止沒錯，但只會停一個，還是隨機的，另一個要再按一次 *ctrl+C* 才會停，至今原因不明，我決定就讓他成為懸案，因為有超直觀的解決方法——把主函式超音波的功能交給另一條 thread，主函式負責加油打氣接收 *ctrl+C* 的中止通知就好。

以上，就是這次多災多難的實驗。

至於我發現其他人都是直接讓整首歌播完根本沒寫 thread 也沒遇到那麼多問題，已經是兩天後的事了。

五、心得

還是忍不住把快要散掉的杜邦線撕開了。整齊的杜邦線讓人看了很舒服，但撕杜邦線也是很療癒。世界上怎麼會有這麼美好的發明呢(*´ω`*)

拿到樹梅派這麼多週，終於大卡關了，有點小感動。

……拚死拚活搞定讓無窮迴圈的 thread 正常結束，才發現其他同學都是讓整首歌正常播完 QuQ 也因此爬了不少 thread 的教學文章，而且修完 bug 的成就感真的滿滿滿!!!

雖然繞了遠路但收穫更多，超開心啦 0u0!!!