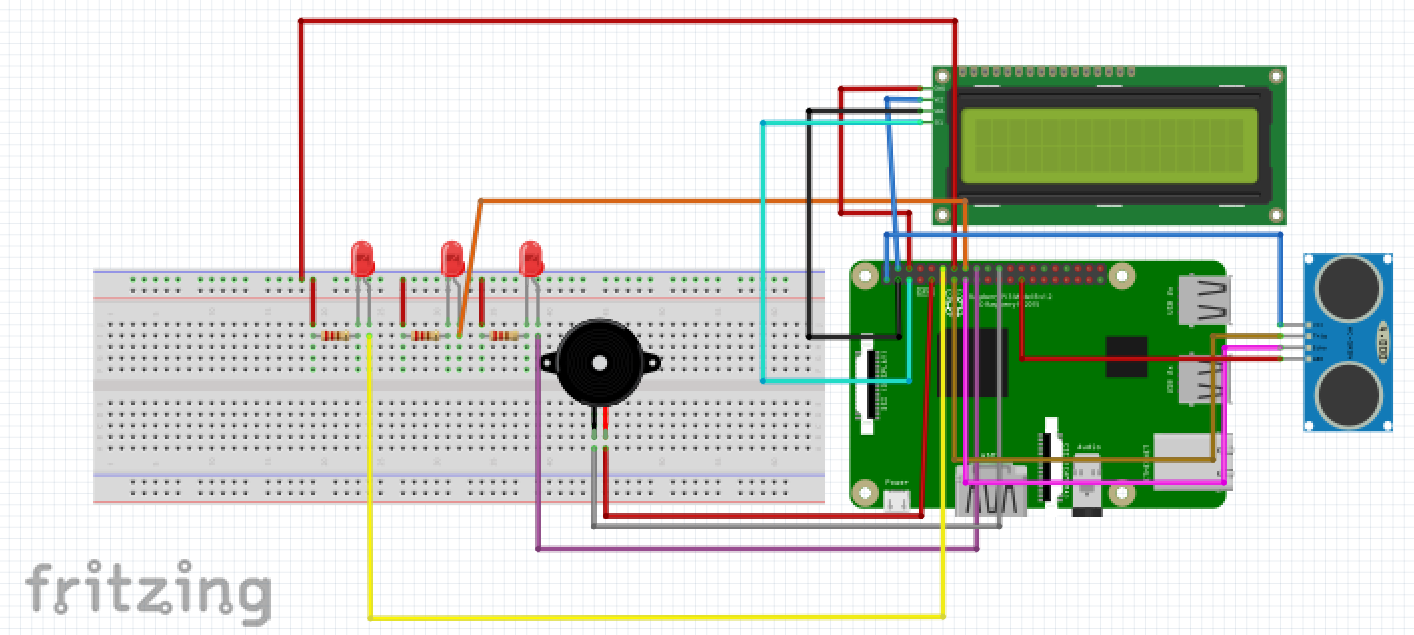
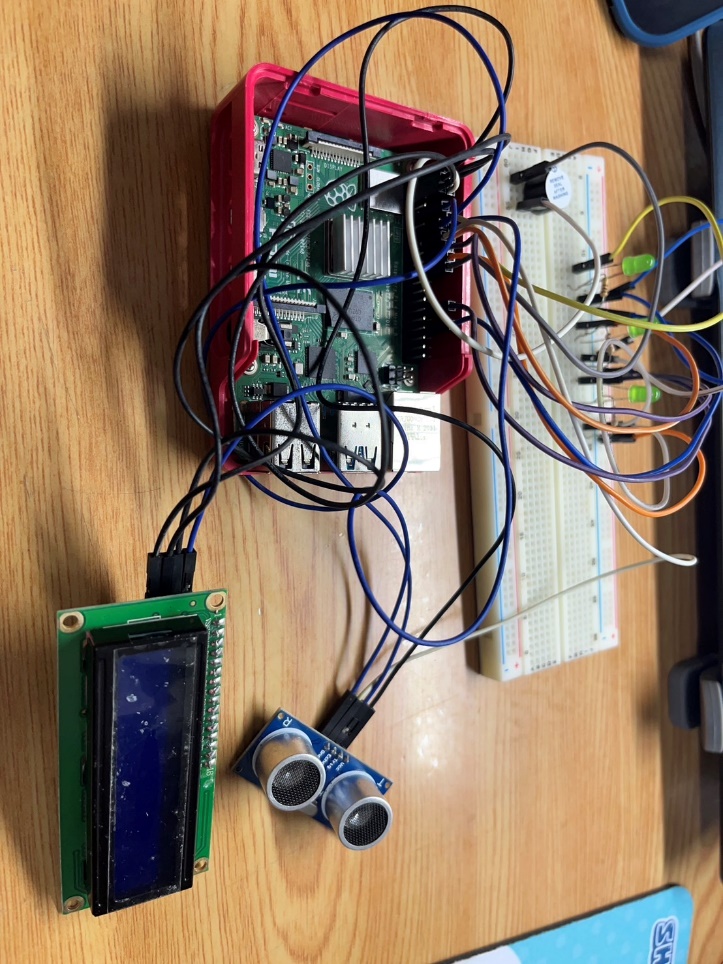
「嵌入式系統」期末專題報告

**作者: 林翡 班級:資四甲 學號:110916015 姓名:林翡**

* 目的：**(至少包含3個以上課堂教授的元件)**
* 本實驗旨在實作一個簡易的倒車雷達，模擬現實生活中的汽車倒車雷達功能。實驗中使用的元件包括超音波模組進行測距，當障礙物距離越來越近時，蜂鳴器 (Buzzer) 會發出警示聲，同時LED燈會閃爍，而測量的距離將顯示在LCD螢幕上。
* 使用元件：
* 蜂鳴器(ZEETEK-1205)\*1
* LED燈\*3
* 超音波模組(HC-SR04)\*1
* I2C LCD\*1
* 220Ω電阻\*3
* 杜邦線\*數條
* 方塊圖(或是連接圖、或是硬體線路圖) (可以手繪)
* **連接圖**



* **實際連接圖**
* 程式碼

1. import RPi.GPIO as GPIO
2. from time import sleep
3. import  time
4. import smbus2
5. from RPLCD.i2c import CharLCD
6. import sys
7. GPIO.setmode(GPIO.BCM)
8. GPIO.setwarnings(False)
9. # 定義GPIO腳位
10. led1\_pin = 18
11. led2\_pin = 23
12. led3\_pin = 24
13. trig\_pin = 27
14. echo\_pin = 22
15. buzzer\_pin = 25
16. #設定LCD
17. sys.modules['smbus'] = smbus2
18. lcd = CharLCD('PCF8574', address=0x27, port=1, backlight\_enabled=True)
19. # 初始化GPIO
20. def setup():
21. GPIO.setwarnings(False)
22. GPIO.setmode(GPIO.BCM)
23. GPIO.setup(led1\_pin, GPIO.OUT, initial=0)
24. GPIO.setup(led2\_pin, GPIO.OUT, initial=0)
25. GPIO.setup(led3\_pin, GPIO.OUT, initial=0)
26. GPIO.setup(trig\_pin, GPIO.OUT)  # Trig
27. GPIO.setup(echo\_pin, GPIO.IN)  # Echo
28. GPIO.output(trig\_pin, False)  # Set trigger to False (Low)
29. GPIO.setup(buzzer\_pin, GPIO.OUT)  # Set pins' mode is output
30. global Buzz  # Assign a global variable to replace GPIO.PWM
31. Buzz = GPIO.PWM(buzzer\_pin, 440)  # 440 is initial frequency.
32. Buzz.start(40)  # Start Buzzer pin with 40% duty ratio
33. v = 343  # 331+0.6T, T=Celsius
34. # 超音波測距函式
35. def detection():
36. GPIO.output(trig\_pin, GPIO.HIGH)
37. sleep(0.00001)  # 10u 秒的TTL觸發信號
38. GPIO.output(trig\_pin, GPIO.LOW)
39. pulse\_start = 0
40. # ---Response level output have proportional with detection to the distance---
41. while GPIO.input(echo\_pin) == 0:  # 等待echo\_pin變高電平
42. pulse\_start = time.time()  # 記錄echo引腳變高前的時間點
43. while GPIO.input(echo\_pin) == 1:  # 等待echo\_pin變低電平
44. pulse\_end = time.time()  # 記錄echo引腳變低前的時間點
45. # -----------------------------------------------------------------------------
46. # 計算距離
47. t = pulse\_end - pulse\_start
48. d = (t \* v) / 2
49. return d \* 100
50. # 計算三次測量的平均距離
51. def detection\_average():
52. d1 = detection()
53. sleep(0.065)
54. d2 = detection()
55. sleep(0.065)
56. d3 = detection()
57. sleep(0.065)
58. distance = (d1 + d2 + d3) / 3
59. return distance
60. # 根據距離的不同，控制Buzzer和三顆LED燈的行為來發出警報
61. def alarm(distance):
62. if distance > 12:
63. warning = 0
64. Buzz.ChangeDutyCycle(0) # 關閉Buzzer，不發出聲音
65. GPIO.output(led3\_pin, 0)
66. GPIO.output(led1\_pin, 1) # 亮第一顆LED燈
67. sleep(0.6)
68. GPIO.output(led1\_pin, 0)
69. elif distance <= 12 and distance > 7:
70. warning = 0
71. Buzz.ChangeDutyCycle(0)
72. GPIO.output(led3\_pin, 0)
73. Buzz.ChangeDutyCycle(50) # 開啟蜂鳴器，設置50%的占空比
74. GPIO.output(led2\_pin, 1) # 亮第二顆LCD燈
75. time.sleep(0.3)
76. Buzz.ChangeDutyCycle(0)
77. GPIO.output(led2\_pin, 0)
78. elif distance <= 7 and distance > 2:
79. GPIO.output(led3\_pin, 1) # 亮第三顆LED燈
80. Buzz.ChangeDutyCycle(50)
81. # 將距離顯示在LCD上
82. def run\_lcd(distance\_cm):
83. try:
84. lcd.clear()
85. lcd.cursor\_pos = (0, 0)
86. lcd.write\_string( "Distance:" )
87. lcd.cursor\_pos = (1, 0)
88. lcd.write\_string(str(distance\_cm) + " cm" )
89. except Exception as exception:
90. printException(e=exception, funcName="run\_lcd")
91. return "Thread end"
92. def destory():
93. Buzz.stop()                 # Stop the buzzer
94. GPIO.output(buzzer\_pin, 1)      # Set Buzzer pin to High
95. GPIO.cleanup()
96. if \_\_name\_\_ == '\_\_main\_\_':
97. try:
98. setup()
99. while True:
100. for i in range(0, 4, 1):
101. distance = detection\_average()
102. alarm(distance)
103. distance1 = round(distance,3) # distance取到小數點第三位
104. run\_lcd(distance1)
105. except KeyboardInterrupt:
106. print("\nException: KeyboardInterrupt\n")
107. finally:
108. destory()

* 程式說明

1. 初始化(setup())和設定：程式開始時會定義及初始化GPIO的腳位並設置它們的模式。設置了三個LED燈腳位、一個蜂鳴器腳位和超音波模組的Trig和Echo腳位。LCD也會初始化以顯示測量結果。[11-36]
2. 定義音速v為343，根據公式331+0.6T，T設定為20度。[38]
3. 定義偵測距離的函式(detection())，監聽 Echo pin 的變化並計算距離。[41-55]
4. 測量三次距離取平均函式(detection\_average())：將三次測量的結果取平均值，以提高測量的穩定性。[58-69]
5. 根據不同距離的判斷，觸發Buzzer和三顆LED燈的行為來發出警報的函式(alarm())：如果距離超過12公分，亮第一顆LED燈；距離在7到12公分之間，亮第二顆LED燈並間歇性鳴叫；距離在2到7公分之間，亮第三顆LED燈並持續鳴叫。 [72-91]
6. 顯示距離在LCD的函式(run\_lcd())。[94-103]
7. 清理函式：當程式結束時，會停止蜂鳴器並清理所有GPIO設定。[105-108]
8. 主函式：呼叫setup初始化GPIO，利用外層迴圈while True持續偵測距離並發出相應的警示。它會一直執行，直到程式被手動中斷（例如按下 Ctrl+C）或發生其他例外情況；內層迴圈是延長顯示距離間的週期，避免LCD顯示距離會太快；最後停止就清空GPIO、停止Buzzer。[110-123]

* 影片連結處

<https://youtu.be/cyDy3Y7eNCg>

* 心得
* 這是我第一次接觸到樹莓派這個硬體設備，一開始光是燒錄OS跟連上無線網路就困難重重的，因為手機是Apple的，那時還以為自己這門課要悲劇了，燒錄OS也重燒了2、3次吧!但後來聽到學姐說用Apple手機可以開最大相容性就可以順利連上網路，最後還真的成功。

　　接下來的作業就是一連串的接電路、找bug、debug，其實我還蠻喜歡動手實作的課，大一、大二還覺得有點排斥，因為那時都覺得好難懂，但到大四該有的基礎都有了，所以接電路、debug也當成是一種基本過程，有成功找出錯誤的地方，完成老師指派的作業就會有成就感，還記得當時的搖桿讓我卡了超級久，最後加了電阻才讓搖桿可以順利分辨方向。

　　這次的期末專題我的發想就是想做一個現實生活中常見到的東西，剛好倒車雷達就是，實作起來也不會太複雜，網路上的資源也蠻多的，但我自己又多加了LCD元件來顯示距離，至於這次遇到的困難是一開始接led燈的電阻都沒有接地，導致Buzzer會叫，但led都不會亮，還有接觸不良的問題，雖然可能跟電阻有關，因為正常是要用220Ω的電阻，但因為盒子只有2個，所以我第一顆led燈就改接100Ω的電阻 ，就導致第一顆led有點接觸不良的感覺，但不曉得是不是因為這個原因。

　　總之，這堂課我認為蠻有趣又好玩的，也不會因為是開在碩班的課就特別難，還蠻容易上手的。

* 其他(補充、參考資料)
* <https://northxhpc.blogspot.com/2016/10/raspberry-pi-3-model-b-project1-parking.html>
* 畫電路圖軟體:fritzing