|  |
| --- |
| 基本資料 |
| * 姓名：王豪逸 * 標題：B024020014\_Assignment 4(基本) |
| 作業目的與設計 |
| * 目標：模擬垃圾車 * 電路設計說明(總分1分)：   車子使用的GPIO：  左輪[前(6),後(13)],右輪[前(19),後(26)]  方向燈的GPIO:  左(20)，右(21)  蜂鳴器(18)   電路設計照片(總分1分)：     * 程式邏輯設計(總分1分)：   Socket server基本上就要有兩種thread，等待用戶連接的跟用戶連接後處理資料的thread。  SocketThread: 等待使用者連接的thread，會卡在accept。使用者連接後會產生ConnectThread。  ConnectThread: 處理使用者資料的thread。會生成控制車子的物件並根據使用者的輸入來使用物件的函式。  車子的部分有音樂跟方向燈這兩個跟時間有關係的動作。我分別弄成兩個thread class  AutoCar: 車子的主物件能夠使用已經寫好的函式來控制他，本身也是方向燈閃爍的thread。也有個音樂盒LittltStar來撥放聲音。  LittltStar: 播放小星星的thread，有簡單的函式能夠讓他執行與暫停撥放。  另外傳送的指令讓他有意義並且配合另外製作的簡單APP改成:  前進: up, 後退: down, 左轉: left, 右轉: right, 停止: stop   * 影片URL：https://youtu.be/wobk3mB69VY |
| 心得(總分1分) |
| * 遇到哪些困難：  1. 車子的馬達一直無法轉動。 2. 一直無法使用CTRL+C完整的中止這個程式  * 如何解決：  1. 做了很多交叉的測試，再找看看是不是L298N壞掉。但是測試了一陣子後發現有些時候一邊能動又有些時候是另一邊。也不是螺絲那部分的接觸不良。最後發現杜邦線接L298N那邊很鬆常常沒有接觸好，拿張紙卡處那些線這個問題就解決了。 2. 試過了try except也試過了擷取signal好不容易在window能夠運作了，但是換到樹梅派也是會出問題。社團上有人有提供好的處理方法是thread上處理不當。   這份作業在socket的部分如果不用thread處理的話這程式很容易出問題，但是很多人為了作業要求或是要簡潔程式碼，只用少少的一兩個甚至不想用，這樣有意義嗎。另外也讓我有點好奇，為甚麼要簡潔這些線路。使用的線材多一點但是看起來清楚明瞭不是很好嗎。 |
| 程式碼 |
| 程式碼：  ====基本題===(總分1分)  import socket  import threading  import signal  import sys  import time  import RPi.GPIO as GPIO  def main():  GPIO.setmode(GPIO.BCM)    sockThread = SocketThread()  sockThread.start()    try:  while(True):  time.sleep(10)  except KeyboardInterrupt:  sockThread.stop()  exit()  class SocketThread(threading.Thread):  def \_\_init\_\_(self):  threading.Thread.\_\_init\_\_(self)  self.ip = "192.168.2.4"#socket.gethostbyname(socket.gethostname())  self.port = 8089  self.threads = []  def run(self):  self.sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  self.sock.bind( (self.ip, self.port) )  self.sock.listen(5)  self.running = True  print('Server start at: %s:%s' %(self.ip, self.port))  while self.running:  try:  conn, addr = self.sock.accept()  print ('Connected by ', addr)  ct = ConnectThread(conn, addr)  self.threads.append(ct)  ct.start()  except socket.error:  pass  for ct in self.threads:  ct.stop()  print("Sock stop")    def stop(self):  self.running = False    for ct in self.threads:  ct.stop()  self.sock.close()    class ConnectThread (threading.Thread):  def \_\_init\_\_(self, conn, addr):  threading.Thread.\_\_init\_\_(self)  self.conn = conn  self.addr = addr  self.car = AutoCar()  self.car.start()  def run(self):  self.running = True  try:  self.conn.send("Hello\n".encode())  while self.running:  data = self.conn.recv(1024).decode()  print (data)  self.car.move(data)  self.conn.send("server received you message.\n".encode())  except socket.error:  print("Disconnect", self.addr)  self.stop()  def stop(self):  self.running = False  #self.conn.close()  self.car.stop()    class AutoCar (threading.Thread):  def \_\_init\_\_(self):  threading.Thread.\_\_init\_\_(self)  self.musicBox = LittltStar()  self.musicBox.start()    self.carGPIO = [6,13,19,26]  self.leftLedGPIO = 20  self.rightLedGPIO = 21  self.leftLed = False  self.rightLed = False  self.flash = True  self.running = False    for i in self.carGPIO:  GPIO.setup(i, GPIO.OUT)  GPIO.output(i, False)  GPIO.setup(self.leftLedGPIO, GPIO.OUT)  GPIO.setup(self.rightLedGPIO, GPIO.OUT)  GPIO.output(self.leftLedGPIO, False)  GPIO.output(self.rightLedGPIO, False)    def run(self):  self.running = True  while self.running :  GPIO.output(self.leftLedGPIO, self.leftLed)  GPIO.output(self.rightLedGPIO, self.rightLed)  time.sleep(0.5)  if(self.flash):  GPIO.output(self.leftLedGPIO, False)  GPIO.output(self.rightLedGPIO, False)  time.sleep(0.5)    def move(self, way):  if(way == "right"):  self.leftLed = False  self.rightLed = True  self.flash = True  self.musicBox.play()  self.setWheel([True, False, False, False])  elif (way == "left"):  self.leftLed = True  self.rightLed = False  self.flash = True  self.musicBox.play()  self.setWheel([False, False, True, False])  elif (way == "up"):  self.leftLed = False  self.rightLed = False  self.flash = True  self.musicBox.play()  self.setWheel([True, False, True, False])  elif (way == "down"):  self.leftLed = True  self.rightLed = True  self.flash = True  self.musicBox.play()  self.setWheel([False, True, False, True])  elif (way == "stop"):  self.leftLed = True  self.rightLed = True  self.flash = False  self.musicBox.mute()  self.setWheel([False, False, False, False])    def setWheel(self, mode):  for i in range(0, len(self.carGPIO)):  GPIO.output(self.carGPIO[i], mode[i])  def stop(self):  self.running = False  self.musicBox.stop()    class LittltStar (threading.Thread):  def \_\_init\_\_(self):  threading.Thread.\_\_init\_\_(self)  self.tone = [0, 262, 294, 330, 349, 392, 440]  self.chord = [1,1,5,5,6,6,5,4,4,3,3,2,2,1,  5,5,4,4,3,3,2,5,5,4,4,3,3,2,  1,1,5,5,6,6,5,4,4,3,3,2,2,1]  self.buzzGPIO = 18  GPIO.setup(self.buzzGPIO, GPIO.OUT)  self.buzz = GPIO.PWM(self.buzzGPIO, 440)  self.buzz.start(60)    self.playMusic = False  self.running = False    def run(self):  temple = 0  self.running = True  while self.running :    if(self.playMusic):  self.buzz.ChangeFrequency(self.tone[self.chord[temple]])  self.buzz.ChangeDutyCycle(60)  temple = (temple+1)%len(self.chord)  if temple % 7 == 0:  time.sleep(0.9)  else:  time.sleep(0.4)  else:  time.sleep(0.4)  self.buzz.ChangeDutyCycle(0)  time.sleep(0.1)    def play(self):  self.playMusic = True  def mute(self):  self.playMusic = False  def stop(self):  self.running = False  main() |