Mechatronics: Progress Report 3

ECOR1044 C: Mechatronics

Group 1044-C2-23

Group Member 1: 101241887 Uchenna Obikwelu

Performed on: 23/03/2022

Submitted on: 26/03/2022

2a.

An H-bridge is an electronic circuit that can switch the polarity of the voltage applied to a load[1]. An H-bridge is basically 4 switches that are connected in series with each other and in parallel with a load (e.g DC motor). By closing two adjacent switches, you can change the polarity of the voltage being applied to the load. In the case of DC motors, H-bridges are sometimes used in their operations because by changing the polarity of the voltage being applied to the motor, you can effectively change the direction of rotation of the motor [1].

2b.

In an industrial setting, the Raspberry Pi(Figure 1.1) would be the central computer through which engineers would program the mechanical machinery to complete its intended tasks. The LCD screen(Figure 1.1) would be a display that would show the state of the machinery at any given time. The H-bridge would be the electronic mainframe that would receive instructions from the central computer in order to operate the machinery. By adding the equivalent of try/except you're sure the whole system does not crush down or stop working from an expected error. Try/Except allows you to provide specific instructions to the machinery in the case where a foreseen error occurs, so the machinery isn't damaged in any way from that error. Adding a second motor to act as a fail safe in the event that the first motor stopped working, would be a good additional functionality.

2c.

Attach an annotated hardware diagram of the lab setup, this should be referenced when discussing the system or components (block diagrams or annotated photographs)

LAB SETUP

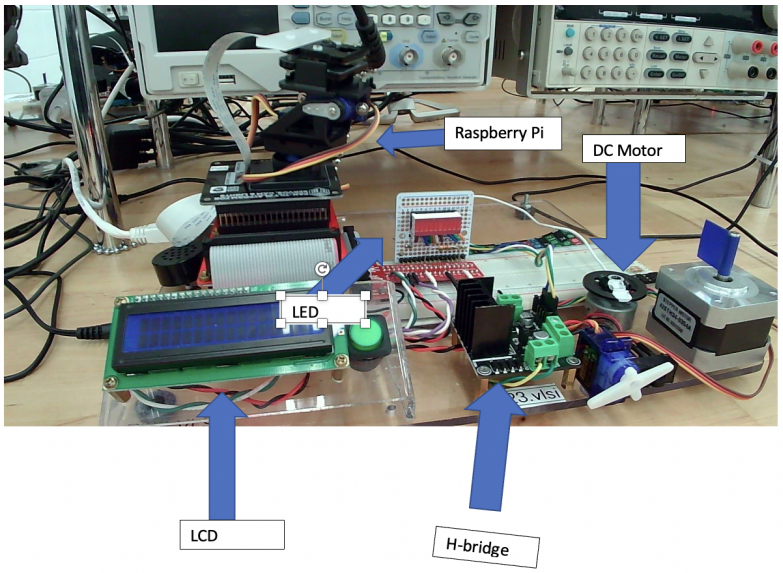


Figure 1.1 Diagram of Hardware used for Lab

2d.

This lab required us to call on our knowledge of python programming and also the knowledge gained from the first two labs such as how to use the GPIO library and the LCD_i2c library, to run a DC motor being operated by an H-bridge. The Lab was exciting, interesting. It was really a great opportunity to work and learn how the raspberry pi works. These labs made be believe that anything is possible electronically and programming wise. We just have to understand the concept behind things and work things out.

Appendix A-Experiment Scripts

6.1

```
import RPi.GPIO as GPIO #Library for GPIO pins
import time #Library for time-related tasks
print('setup')
GPIO.setmode(GPIO.BCM) #Sets a way to reference the GPIO pins
GPIO.setup(20,GPIO.OUT) #Sets GPIO Pin 20 to an output pin
GPIO.setup(21,GPIO.OUT) #Sets GPIO Pin 21 to an output pin

# Main Script

try:
    GPIO.output(20, GPIO.HIGH) #Sets the voltage of Pin 20 'HIGH' (3.3V)
    time.sleep(1)
    GPIO.output(20, GPIO.LOW) #Sets the voltage of Pin 20 'LOW' (0V)
    time.sleep(2)
    GPIO.output(21, GPIO.HIGH) #Sets the voltage of Pin 21 'HIGH' (3.3V)
    time.sleep(1)
    GPIO.output(21, GPIO.LOW) #Sets the voltage of Pin 21 'LOW' (0V)

except KeyboardInterrupt:
    pass
GPIO.cleanup()#Resets the GPIO Pins that we used
print('Done') #Lets the user know the program is done running
```

6.2

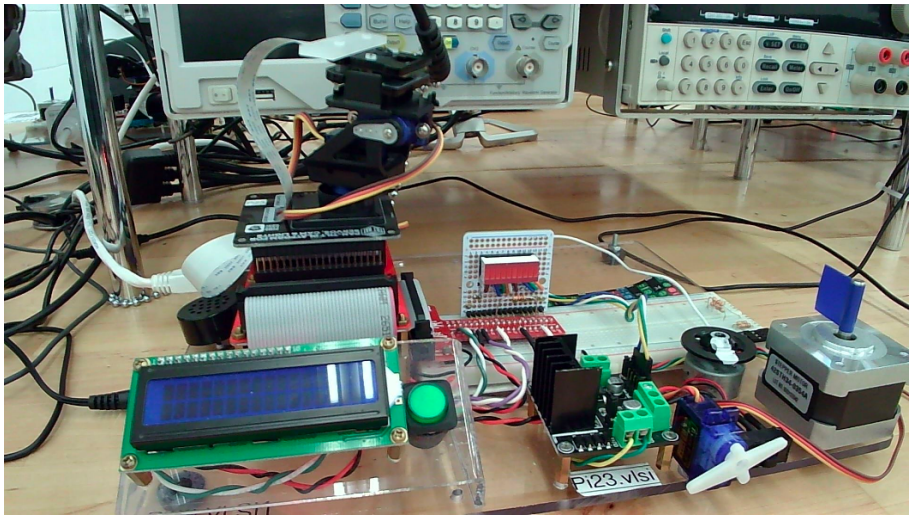
```
import RPi.GPIO as GPIO #Library for GPIO pins
import time #Library for time-related tasks
import lcd_i2c
```

```
print('setup')
GPIO.setmode(GPIO.BCM) #Sets a way to reference the GPIO pins
GPIO.setup(20,GPIO.OUT) #Sets GPIO Pin 20 to an output pin
GPIO.setup(21,GPIO.OUT) #Sets GPIO Pin 21 to an output pin
GPIO.setup(23,GPIO.OUT) #Sets GPIO Pin 23 to an output pin
#PWM Signal Setup
pin_1 = GPIO.PWM(20, 50) #set pin 20 as a PWM output, with a frequency of 50 Hz
pin_2 = GPIO.PWM(21, 50) #set pin 21 as a PWM output with a frequency of 50Hz
pin_1.start(0) #sets the starting duty cycle of the PWM signal to 0% and initialises the signal
pin_2.start(0) #sets the starting duty cycle of the PWM signal to 0% and initialises the signal
time.sleep(1) #sleep for a second to ensure signal is initialised properly
lcd_i2c.lcd_init() #calls setup function of the LCD
lcd_i2c.lcd_string('Hello',1) #Prints a string to the lcd display
time.sleep(2)

print("Begin") #informs user the main function of the program is beginning
try:
   while True: #Continuously running loop
      userinput = input('Enter a Command: \n')
      if userinput == 'DirA':
         pin_2.ChangeDutyCycle(0)
         GPIO.output(23, GPIO.HIGH) #Sets the voltage of Pin 23 'HIGH' (3.3V)
         for dc in range(19):
            lcd_i2c.lcd_string('DirA', 1)
            pin_1.ChangeDutyCycle(dc) #changes the duty cycle to dc
   #print(dc)
            time.sleep(1)
         GPIO.output(23, GPIO.LOW) #Sets the voltage of Pin 23 'LOW' (0V)
         time.sleep(3)
#print('Changing Direction')
#dc = 0
      elif userinput == 'DirB':
         pin_1.ChangeDutyCycle(0)
         GPIO.output(23, GPIO.HIGH) #Sets the voltage of Pin 23 'HIGH' (3.3V)
         for dc in range(19):
            lcd_i2c.lcd_string('DirB',2)
            pin_2.ChangeDutyCycle(dc) #changes the duty cycle to dc
      #print(dc)
            time.sleep(1)
         GPIO.output(23, GPIO.LOW) #Sets the voltage of Pin 23 'LOW' (0V)
         time.sleep(3)
      elif userinput == 'stop':
         print('Cleaning Up')
         lcd_i2c.lcd_string('Goodbye',1)
         time.sleep(2)
         pin_1.stop() #stops the pin initialization
         pin_2.stop() #stops the pin initialization
         lcd_i2c.cleanup() #LCD cleanup
         GPIO.cleanup() #cleanup all of the GPIO pins used within the script
         break
      else:
         print('Invalid Input \n Try again')
except KeyboardInterrupt:
   pass
lcd_i2c.lcd_string('Goodbye',1)
```

```
time.sleep(2)
if userinput == 'stop': #checks if user manually stops the program and performs a cleanup if this is False
    pass
else:
    pin_1.stop() #stops the pin initialization
    pin_2.stop() #stops the pin initialization
    lcd_i2c.cleanup() #LCD cleanup
    GPIO.cleanup() #cleanup all of the GPIO pins used within the script
print("Done") #informs the user the program is finished running
```

Appendix B-Experiment Photos



References

[1] M. Feuerherm and T. Karamat, 'ECOR1044 Lab 3 Instructions' *Bright Space ECOR 1044* [Online] Available: https://brightspace.carleton.ca/d2l/le/content/57180/viewContent/2550925/View.[Accessed: 26-Mar-2022].