Advanced embedded Systems

Lab 8	Name:
Slides to command RC servomotors	Shane Randell

Materials required.

Quantity	item
1	Raspberry Pi
1	microSD memory card (4GB to 32GB)
1	Power supply USB connector (1.5Amp)
1	PC
1	Enable network connection
1	USB Mouse and keyboard
1	Monitor and cable HDMI or VGA + adaptor
1	LED
1	Resistor 220 Ohms
1	Push button
1	10K resistor
2	RC servomotors
1	74HC14 (optional)
1	Arduino board (NANO, UNO, Mega, etc.)
2	Potentiometers

Table 1. Components

The first part of this lab shows the GUI slide function to command the position of two RC-servos. The second part, the student must implement the same the Arduino microcontroller.

First part:

a) Design a GUI program that includes two slides like the following Figure:

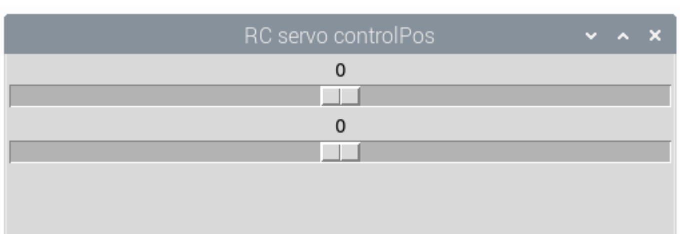


Figure 1 App with two slides

b) complete the code for commanding the Servomotors using two sliders:

```
from guizero import App, Slider
from gpiozero import AngularServo
```

```
from guizero import App, Slider,TextBox
from gpiozero import AngularServo
import time

maxPW = 2/1000
minPW = 1/1000


def slider_read(slider_value):
    textbox.value = slider_value
    print(slider_value)
    servo1 = AngularServo(20, min_pulse_width = minPW, max_pulse_width = maxPW, initial_angle = 0, min_angle = -90, max_angle = 90)
    servo1.angle = int(slider_value)
    time.sleep(0.5)

def slider2_read(slider_value):
    textbox.value = slider_value
    print(slider_value)
    servo2 = AngularServo(21, min_pulse_width = minPW, max_pulse_width = maxPW, initial_angle = 0, min_angle = -90, max_angle = 90)
    servo2.angle = int(slider_value)
    time.sleep(0.5)

app = App()
slider1 = Slider(app, start=-90, end=90, width="fill",command = slider_read)
slider2 = Slider(app, start=-90, end=90, width="fill",command = slider2_read)
textbox = TextBox(app)

print(textbox.value)

app.display()
```

I didnot have motors to test my code. but I sent it to a class mate and they confirmed It worked. I will attach photos of my wave forms.

Table 1. First program

c) Mount the two servomotors as shown in the following photo:

Figure 2 Two RC-servomotors attached

d) Connect control RCservo1 to pin GPIO20, and RCservo2 to pin GPIO21, Vcc and GND of the servomotor from external power supply. Note that GND should be connected to the external power supply and the GND of the Raspberry Pi.

Optionally you could connect the following gate to protect your Raspberry Pi (for large using periods)

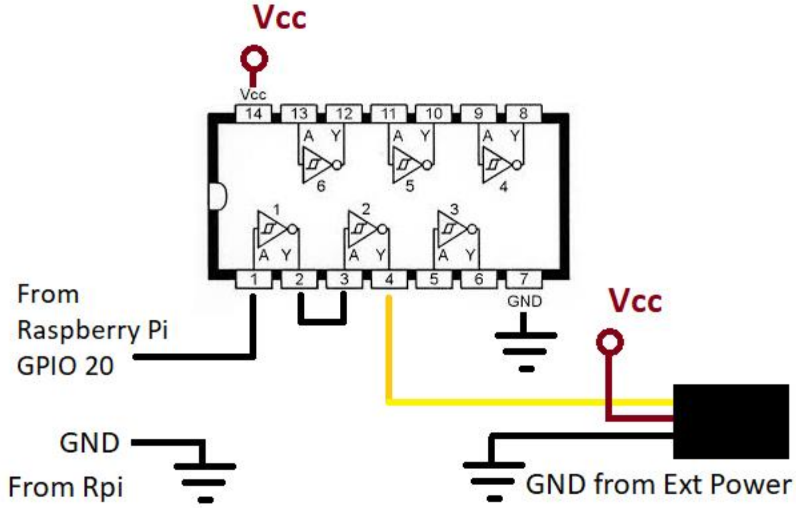


Figure 3. Optional circuit to connect the command line from the raspberry Pi to the servomotor cables. Note that GPIO 21 also require to be connected, you can use the other inverters for that, you would need two inverters for the other servomotor to avoid altering the logic.

Second part:

Implement the same system using Arduino microcontroller by using potentiometers to command de position and then compare the performance of the previous application

Write the Arduino code here:

```
#include <servo.h>

Servo servo_test;
int pot = 0, angle = 0;

Void setup ()
{ servo_test.attach(9);
}

Void loop() {
  angle = analogread(pot);
  angle = map(angle,0,1023,0,179);
  servo_test.write(angle);
  delay(5); }
```

I did not have any servos so I wrote code just to give me an output wave form to compare to my raspberry pi.

Questionnaire

1. did you experience a difference of command the position of the RC servomotors using the Raspberry Pi vs the Arduino Board? If yes, explain the difference.

2. Did you calibrate the width of the pulses with the Raspberry Pi program? If yes, explain why and how you did that.

3. Using analog discovery studio or the oscilloscope in the lab plot the PWM signals produced by the Raspberry Pi for 0 to 180 degrees:

Shaft Position	Raspberry Pi	Arduino
0°	T_{on}= 0 T_{off}= 0	T_{on}= 576µS T_{off}= 19.31ms
45°	T_{on}= 1.08ms T_{off}= 19.08mS	T_{on}= 836µS T_{off}= 19.137ms
90°	T_{on}= 1.22ms T_{off}= 18.70ms	T_{on}= 1.04ms T_{off}= 18.83ms
135°	T_{on}= 1.59ms T_{off}= 18.40ms	T_{on}= 1.48ms T_{off}= 18.44ms
180°	T_{on}= 1.87ms T_{off}= 18.26ms	T_{on}= 1.89ms T_{off}= 18.134ms

Table 1

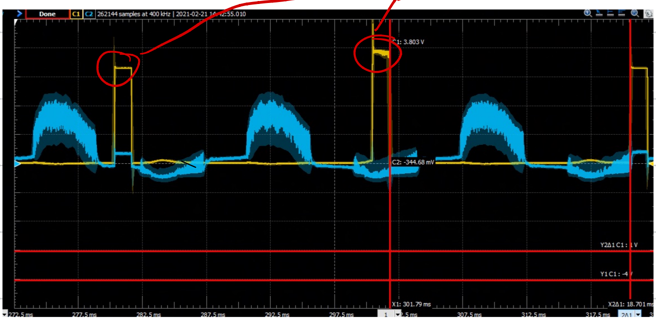
Conclusions

Describe all the findings of using slides and the hardware considerations to implement this system and upload the programs to Github; and record the videos and share the links to the Dropbox.

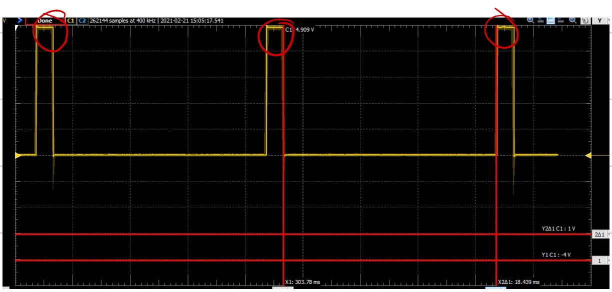
1) I did not have a motor for this lab but based on the wave forms generated the ardinuo was likely much better at controlling the servos motor. The wave form was much smoother and had alot less noise.

Raspberry Pi

Noise on signal



Arduino → No noise



2.) I did not have motors to calibrate. But after talking to class mates, you need to calibrate the moter. You do this by using the defult max and min value for the library, once you find these you compare it to the actually servo moving. By stepping the correction up incrimentally, you can calibrate the motor to move within its full range.

Conclusions

This lab went well for me, but due to my motor breaking, I had to do this lab based on wave fourms, not the motor moving. This was tricky becacuse not having a reference point for the measurements became frustrating. Leaving me to compare duty cycles rather than motor positions.