Team 4

Date Due: 11/17/2023

Course: ELEC 553

Project 3: Motor Control

Team Members : Shaun Lin (hl116), Eric Lin (el38),

Yen-Yu Chien (yc185), Saif Khan (sbk7)

Inspiration:

The project for this team took inspiration from the following project , Autonomous Lane Keeping Car Using Raspberry Pi and OpenCV [1].

Objective:

The objective for the graduate students (ELEC 553) is to program and control an autonomous RC car using a BeagleBone AI-64 (BBAI64) [2]. The project focuses on implementing motor control for precise speed and steering adjustments, utilizing Pulse-width Modulation (PWM) for enhanced accuracy. We are tasked with integrating the BBAI64 with the RC car's motor systems, modifying existing Python scripts for compatibility, and demonstrating the car's motor functionalities. A significant aspect of the project is to setup the python scripts to be able to operate the motor of the RC car as well as the documentation and reporting process, which includes submitting well-commented code and learning outcomes.

Challenges:

Challenge 1- Network Configuration : This was the most challenging part of the project in order to connect the BeagleBone AI-64 to WIFI. In the instruction, we were told that the BeagleBone was configured and could be connected to the internet. However, it was not the case. It took us a long time to set and configure the WiFi. As a result, we wrote a script so that the BeagleBone connects to WiFi without additional settings.

Challenge 2- PWM Setup: The challenging aspect of the project was to set up network configuration on the BeagleBone AI-64. The PWM configuration to control the motors took quite a bit of time to completely adjust the Pulse width modulation duty cycle and successfully control the ESC motor and steering servo. The project mentioned teams we used for inspiration [3].

Interesting Application of Approach: An interesting approach of this application would be to create a GUI where when the motors start it would show the speed of the RC Car and as the acceleration increases the GUI would show the current speed of the motors. This can be done using matplotlib in python [4].

Full Code for Motor Control:

```python
'''
ELEC 424/553
Project 3
Authors: Eric Lin(el38), Shaun Lin(hl116), Yen-Yu Chien (yc185), Saif Khan (sbk7)
'''
import time
import math

# stop: 7.5
# forward min: 8.0
# forward max: 9.0
# ESC motor speed function
def ESC(percentage):
    # P9_14 - Speed/ESC
    with open('/dev/bone/pwm/1/a/period', 'w') as filetowrite:
        filetowrite.write('20000000')
    with open('/dev/bone/pwm/1/a/duty_cycle', 'w') as filetowrite:
        filetowrite.write(str(int(percentage/100*20000000)))
    with open('/dev/bone/pwm/1/a/enable', 'w') as filetowrite:
        filetowrite.write('1')
    return

# turn left max: 11
# turn right max: 3
# steering angle of the servo function
def Servo(percentage):
    # P9_16 - Steering
    with open('/dev/bone/pwm/1/b/period', 'w') as filetowrite:
        filetowrite.write('20000000')
    with open('/dev/bone/pwm/1/b/duty_cycle', 'w') as filetowrite:
        filetowrite.write(str(int(percentage/100*20000000)))
    with open('/dev/bone/pwm/1/b/enable', 'w') as filetowrite:
        filetowrite.write('1')
    return

# Initialize the servo
servo_percentage = 7.5
Servo(servo_percentage)
# Initialize the ESC motor
motor_percentage = 7.5
ESC(motor_percentage)
time.sleep(0.5)

# Start the motor
print("Starting the motor")
for x in range(75, 91):
    ESC(x/10)
    time.sleep(0.1)

# Stop the motor
print("Stopping the motor")
ESC(7.5)
time.sleep(0.5)

# Turn from left to right
print("Turning from left to right")
for x in range(3, 11):
    Servo(x)
    time.sleep(0.1)

# Turn Servo back to center
print("Turning Servo back to center")
Servo(7.5)
time.sleep(0.5)
```

Figure 1: Screenshot all of pwm_test.py code

In figure 1 , This script pwm_test.py (attached with this document) controls the speed and steering of an autonomous RC car using a BeagleBone AI-64 (BBAI64). It does this by manipulating the Pulse Width Modulation (PWM) signals to control an Electronic Speed Controller (ESC) for the motor and a servo motor for steering. The functions perform the following :

1) ESC (percentage): Controls the speed of the car.
2) Servo(percentage): Controls the steering direction of the car.

ESC (percentage): This function controls the speed of the RC car's motor. It takes a percentage value as input, which it converts to a duty cycle for the PWM signal. This signal is then used to set the speed of the motor, with the duty cycle determining how fast the motor spins. The function writes to character device drivers called period, duty_cycle and enables them to set the PWM period, duty cycle, and enable the signal, thereby controlling the motor's speed respectively. The 20,000,000 is in nanoseconds which is 20 milliseconds. The filetowrite.write(‘1’) enables PWM signal.

Servo (percentage): This function manages the steering of the RC car by controlling a servo motor. It receives a percentage value, translating it into a duty cycle for the PWM signal that dictates the servo's position. The function adjusts the steering angle by writing to character device drivers called period, duty cycle and enable to set the PWM period, duty cycle, and enable the signal, allowing precise control over the car's steering mechanism.

Hardware Setup:

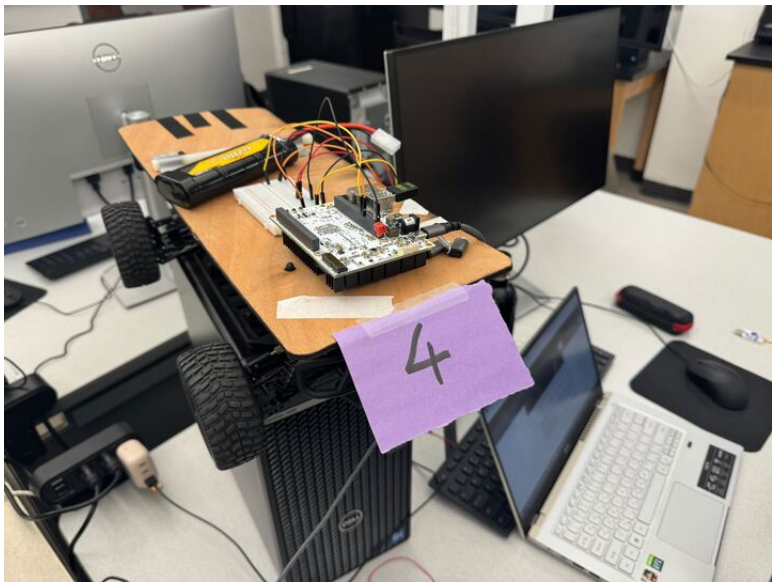


Figure 2: Hardware setup.

The hardware setup in Figure 2 includes the following items to be able to operate the RC car motors, all of the materials were provided for by the course instructor. The Microcontroller used for the project is the beagle bone black AI-64 [2].

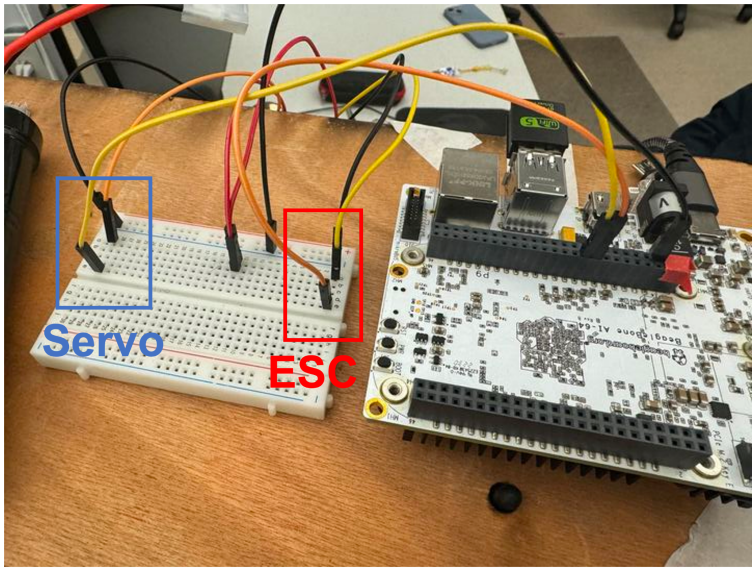
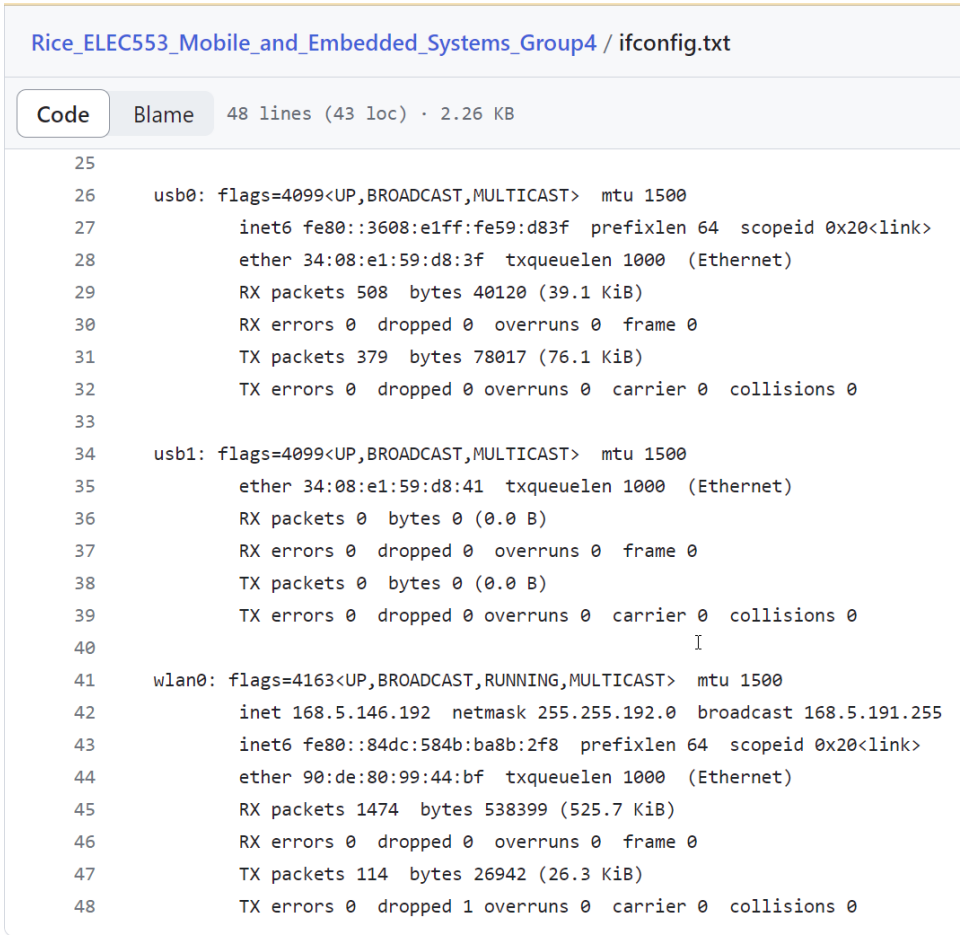


Figure 3: BeagleBone AI-64 Motor Connection Circuit.

The yellow, red, brown wires , as stated in the project document are the wires that control the connection from the Beaglebone board to the servo motors on the car. The servo motors are operated using PWM on P9_14 and P9_16 pins.



Rice_ELEC553_Mobile_and_Embedded_Systems_Group4 / ifconfig.txt

Code Blame 48 lines (43 loc) · 2.26 KB

```
25
26    usb0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
27            inet6 fe80::3608:e1ff:fe59:d83f  prefixlen 64  scopeid 0x20<link>
28            ether 34:08:e1:59:d8:3f  txqueuelen 1000  (Ethernet)
29            RX packets 508  bytes 40120 (39.1 KiB)
30            RX errors 0  dropped 0  overruns 0  frame 0
31            TX packets 379  bytes 78017 (76.1 KiB)
32            TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
33
34    usb1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
35            ether 34:08:e1:59:d8:41  txqueuelen 1000  (Ethernet)
36            RX packets 0  bytes 0 (0.0 B)
37            RX errors 0  dropped 0  overruns 0  frame 0
38            TX packets 0  bytes 0 (0.0 B)
39            TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
40
41    wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
42            inet 168.5.146.192  netmask 255.255.192.0  broadcast 168.5.191.255
43            inet6 fe80::84dc:584b:ba8b:2f8  prefixlen 64  scopeid 0x20<link>
44            ether 90:de:80:99:44:bf  txqueuelen 1000  (Ethernet)
45            RX packets 1474  bytes 538399 (525.7 KiB)
46            RX errors 0  dropped 0  overruns 0  frame 0
47            TX packets 114  bytes 26942 (26.3 KiB)
48            TX errors 0  dropped 1 overruns 0  carrier 0  collisions 0
```

Figure 4: Screenshot of the MAC Address.

Figure 4 shows the MAC address **168.5.146.192** which allows us to wirelessly connect ssh to the Beaglebone black board, this file is attached with this document in the submission.

References:

[1] R. (n.d.). *Autonomous Lane-Keeping Car Using Raspberry Pi and OpenCV*. Retrieved November 14, 2023, from https://www.instructables.com/Autonomous-Lane-Keeping-Car-Using-Raspberry-Pi-and/

[2]R. (n.d.). *BeagleBone AI-64*. Beaglebone.org. Retrieved November 14, 2023, from https://www.beagleboard.org/boards/beaglebone-ai-64

[3]R. (n.d.). COVID Debuff (Semi-Autonomous RC Car Platform). Hackster.io. Retrieved November 14, 2023, from https://www.hackster.io/covid-debuff/covid-debuff-semi-autonomous-rc-car-platform-75b072

[4] M. (n.d.). *MatplotLib Documentation*. Retrieved November 17, 2023, from https://matplotlib.org/stable/index.html