LESSON D-11

ACTIVITY #1:
MOTOR DIRECTION TEST

In the last lesson you confirmed that both motors would turn in both directions with the robot held above your work surface. This is a good test, but there could still be problems that exist with your motor wiring that could be causing reversed commands to the drive motors. In this activity, you will modify the program from Lesson D-10 to drive both motors in forward and then reverse to confirm the motors propel the robot in the expected direction when using the GPIO pin numbers below:

High on GPIO Pin	Motor Movement
GPIO23	Right Motor Reverse
GPIO24	Right Motor Forward
GPIO19	Left Motor Forward
GPIO26	Left Motor Reverse

Note:

The enable lines required for these motor movements have been omitted from this chart for clarity.

We will first be powering the robot using AC wall power to modify the program, and then switching to battery power for mobile testing once the program is complete.

Step #1

If the Pi is currently up and running, shut the Pi down before proceeding.

Disconnect battery power input from the Pi by disconnecting the micro USB power connector. Connect the 5V wall power adapter to wall power and then to the micro USB power connection of the Pi. The Pi will power up and automatically connect to your WiFi network.

Step #2

Connect to the VNC server of the Raspberry Pi using a desktop computer or laptop connected to your WiFi network. Once connected, you should be viewing the Desktop of the Pi just as if you had a monitor, keyboard, and mouse connected directly to the Pi.

Open the folder named **`robot`** on the Desktop and double-click on the file named **`motor_test.py`** to open the file in Thonny.

Step #3

Save a new copy of this program in Thonny by selecting **`File`**, and then **`Save As`** from the upper-left menu. Double-click on the **`robot`** folder in the displayed list of files to save your file in the same location as the original. Type **`motor_fwd_rev.py`** in for the file name and click on the **`OK`** button in the lower-right of the Thonny window.

You will now be working in a new copy of the file so the original can be used again later, if needed.

Step #4

The first modification will be to allow the function named **drive_motor** to allow for multiple arguments. The arguments that we will use will be pin1, pin2, and state:

Pin1 – A pin to control

Pin2 – A pin to control

State – The desired state of high or low

This will save quite a bit of code since we will be controlling multiple motor pins to drive the motors. Change the word **pin** after the function name to **pin1, pin2, state**:

```
def drive_motor(pin):
```

will become

```
def drive_motor(pin1, pin2, state):
```

Step #5

The next modification will be to replace the code the function will run when called. The new function behavior will be to change the high/low state of **pin1** and **pin2** to the **state** specified in the arguments when the function is called. Remove the current content from the **drive_motor** function and replace it with the following highlighted lines of code:

```
def drive_motor(pin1, pin2, state):
    GPIO.output(pin1, state)
    GPIO.output(pin2, state)
    time.sleep(1)
```

This will cause **pin1** and **pin2** to pushed high or pulled low based on the value of the state argument, and then a **1** second delay will occur before returning to the main program.

Step #6

It's now time to modify the main program motor command to include the additional pin and state information that was added to the function. Here are the motor commands that will be used:

1. The **r_for** and **l_for** pins will be pushed high to drive the robot forward.
2. The **r_for** and **l_for** pins will be pulled low to stop the robot.
3. The **r_rev** and **l_rev** pins will be pushed high to drive the robot in reverse.
4. The **r_rev** and **l_rev** pins will be pulled low to stop the robot.

Remove the existing **drive_motor** function calls and replace them with the highlighted lines of code below:

```
    time.sleep(1)

drive_motor(r_for, l_for, 1)
drive_motor(r_for, l_for, 0)
drive_motor(r_rev, l_rev, 1)
drive_motor(r_rev, l_rev, 0)

GPIO.cleanup()
```

The program is now fully modified, and all changes have been highlighted below:

```
import RPi.GPIO as GPIO
import time

motors = [13,19,26,12,23,24]
r_enable = 12
r_for = 24
r_rev = 23
l_enable = 13
l_for = 19
l_rev = 26

GPIO.setmode(GPIO.BCM)
GPIO.setup(motors, GPIO.OUT)
GPIO.output(motors, GPIO.LOW)

GPIO.output(r_enable, 1)
GPIO.output(l_enable, 1)

def drive_motor(pin1, pin2, state):
    GPIO.output(pin1, state)
    GPIO.output(pin2, state)
    time.sleep(1)

drive_motor(r_for, l_for, 1)
drive_motor(r_for, l_for, 0)
drive_motor(r_rev, l_rev, 1)
drive_motor(r_rev, l_rev, 0)

GPIO.cleanup()
```

Step #7

Save the program and shut down the Raspberry Pi. Save your program, exit Thonny, and shut down the Pi. Once the SD card activity LED is no longer flashing green, remove micro USB power cable from the Pi.

Connect the micro USB power cable from the on-board 5V converter to the Pi. Next, connect the 9-volt battery pack to the 5V converter, powering up the robot.

Caution – When running on power from the battery pack, the motors will be powered up and ready to move. **Make sure that your robot is on the floor or in some location where driving off the edge of a desk or tabletop is not a possibility**, as this could result in damage to the robot that cannot be easily repaired.

Do not continue to run the robot if the red power LED on the Pi is flashing or turning off. Discontinue use of the robot immediately and charge or replace the batteries. Running the robot in a low battery pack could result in corruption or data loss on the SD card in the Pi.

Step #8

Connect to the VNC server of the Raspberry Pi using a desktop computer or laptop connected to your WiFi network. Once connected, you should be viewing the Desktop of the Pi just as if you had a monitor, keyboard, and mouse connected directly to the Pi.

Open the folder named robot on your Desktop and double-click on the program named **`motor_fwd_rev.py`** to open it in a new Thonny window.

Ensure the area around the robot is clear of any obstacles and safe for driving. Run the program to confirm that the robot drives forward for one second, stops, reverses for one second, and stops. If the robot deviates from this expected behavior, power the robot down and check the list of possible wiring causes below:

- Robot drives in reverse and then forward – Both motors have swapped connections, check connections of both motors from GPIO pin, through motor drive IC, and onto motor per wiring connections made in Lesson D-10.
- Robot drives in circles instead of forward/reverse – One motor is reversed, determine which motor drives in reverse when program starts. Check wiring for this motor per Lesson D-10 and correct reversed wiring as needed

Do not proceed to the next activity until running this program results in the robot driving forward and then backward.