MEMORANDUM

To:	Charlie Refvem, Department of Mechanical Engineering, Cal Poly SLO crefvem@calpoly.edu	
From:	Antonio Ventimiglia ventimig@calpoly.edu	Caiden Bonney clbonney@calpoly.edu
Date:	10/2/2025	
RE:	**Writing Hardware Drivers**	

In order to test the functionality of the Motor and Encoder classes, a test was administered to the Romi to satisfy the following six parameters:

1. Each motor can be enabled and disabled individually and does not start moving when enabled.
2. Each motor can spin forward and backward independently with varying speed.
3. The encoder and motor objects are paired together properly in code so that spinning a motor causes its associated encoder to count.
4. The encoders each count upward when the motors have a positive duty cycle applied, and the wheels rotate such that Romi would drive forward.
5. Each encoder position can grow in the positive and negative directions.
6. Timer reloads have no effect on the encoder speed or position.

The test comprised of measuring the position and velocity of each motor after enabling or setting a motor's effort. A snippet of the code can be seen in Figure 1.

```
# 1. Each motor can spin forward and backward independently with varying speed.
print("************************** Test 1 **************************")
print("Zeroing right and left encoders before enabling motors")
right_encoder.zero()
left_encoder.zero()
wait_and_print_info()

print("Enabling right and left motors")
right_motor.enable()
left_motor.enable()
wait_and_print_info()

print("left motor effort: 20    -  Right Spin: None      Left Spin: Forward")
left_motor.set_effort(20)
wait_and_print_info()

print("right motor effort: -20  -  Right Spin: Reverse   Left Spin: Forward")
right_motor.set_effort(-20)
wait_and_print_info()

print("left motor effort: -20   -  Right Spin: Reverse   Left Spin: Reverse")
left_motor.set_effort(-20)
wait_and_print_info()

print("right motor effort: 20   -  Right Spin: Forward   Left Spin: Reverse")
right_motor.set_effort(20)
wait_and_print_info()

print("left motor effort: 100   -  Right Spin: Forward   Left Spin: Forward")
left_motor.set_effort(100)
wait_and_print_info()

print("right motor effort: 0    -  Right Spin: None      Left Spin: Forward")
right_motor.set_effort(0)
wait_and_print_info()

print("Disabling both motors    -  Right Spin: None      Left Spin: None")
left_motor.disable()
right_motor.disable()
wait_and_print_info()
```

Figure 1. Snippet of code from testing

As seen in Figure 1, the “wait_and_print_info()” function is called after each action is done to the motor. This function waits some time while updating the encoder drivers and then prints the position and velocities of both encoders. The full terminal output can be seen in Figure 2.

```
************************** Test 1 **************************
Zeroing right and left encoders before enabling motors
waited
right pos: 0
left  pos: 0
right vel: 0
left  vel: 0

Enabling right and left motors
waited
right pos: 0
left  pos: 0
right vel: 0
left  vel: 0

left motor effort: 20    -  Right Spin: None      Left Spin: Forward
waited
right pos: 0
left  pos: 9207
right vel: 0
left  vel: 0.0006923077

right motor effort: -20  -  Right Spin: Reverse   Left Spin: Forward
waited
right pos: -11154
left  pos: 18194
right vel: -0.0008461538
left  vel: 0.0006923077

left motor effort: -20   -  Right Spin: Reverse   Left Spin: Reverse
waited
right pos: -22343
left  pos: 8950
right vel: -0.0008461538
left  vel: -0.0007692308

right motor effort: 20   -  Right Spin: Forward   Left Spin: Reverse
waited
right pos: -11083
left  pos: -141
right vel: 0.0008461538
left  vel: -0.0007692308

left motor effort: 100   -  Right Spin: Forward   Left Spin: Forward
waited
right pos: -546
left  pos: 63864
right vel: 0.0008461538
left  vel: 0.005076923

right motor effort: 0    -  Right Spin: None      Left Spin: Forward
waited
right pos: 9753
left  pos: 129799
right vel: 0.0007692308
left  vel: 0.005076923

Disabling both motors    -  Right Spin: None      Left Spin: None
waited
right pos: 9822
left  pos: 131200
right vel: 0
left  vel: 0
```

Figure 2. Terminal outputs from testing

After analyzing the terminal output from Figure 2, all the success can be observed. Since one motor can be enabled without changing the position of the other (action 3), so we satisfy parameter 1. Since the left and right motors can be set to unique effort values independently (meaning setting one motor’s effort had no effect on the other motor’s velocity) parameter 2 is satisfied. Since the position is shown to have been zero after enabling/zeroing then non-zero once the motor’s effort was set, parameter 3 is satisfied. Since the position shows an increase with a positive effort and decrease with negative effort, parameter 4 is

satisfied. It was also visually confirmed that the positive effort corresponded to the forward direction. As seen in Figure 2, the position of each encoder was able to be both a positive and negative value, therefore parameter 5 is satisfied. Lastly, there are instances in which the encoders reach a position value over 65,535 which means the auto-reload set at 0xFFFF does not affect data collection, satisfying parameter 6. This quick analysis shows how the code in the python files "main.py", "encoder.py", and "motor.py" provide the required functionalities to the Romi device.

References:

- Refvem, C. (2025). *Lab 0x01 Interrupt Callbacks and ADC Reading* [Unpublished lab instructions]. Department of Mechanical Engineering, Cal Poly. Retrieved from https://canvas.calpoly.edu/courses/161863/assignments/1368600?module_item_id=4790186
- Refvem, C. (2025). *Lab 0x01 Lecture 5 – Decoding Quadrature Encoders* [Unpublished lecture notes]. Department of Mechanical Engineering, Cal Poly.
- Ventimiglia, A., & Bonney, C. (2025). *main.py, encoder.py, motor.py* [Unpublished computer software]. Department of Mechanical Engineering, Cal Poly.