

ECE 476/676 - Homework #4

Timing, Analog I/O, Motors with Binary Inputs

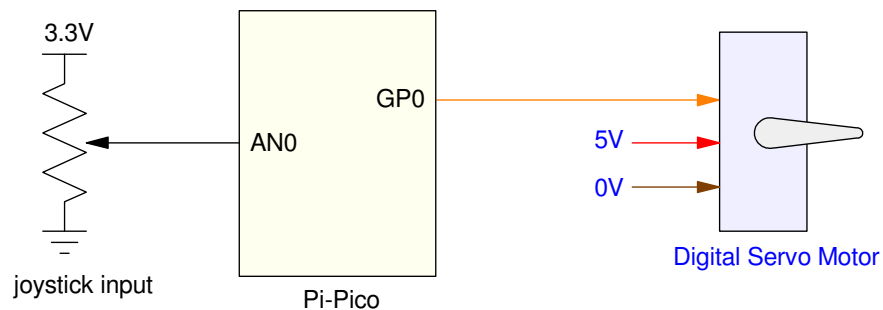
Digital Servo Motor Angle Control

1) Hardware: Connect your Pico to your digital servo motor (blue motor with a three wire interface).

- Yellow: Signal (0.5ms to 2.5ms pulse @ 50Hz)
- Red: +5V
- Brown: 0V

Write a test program to verify you can

- Read the analog input from your joystick
- Drive the motor from 0 degrees (0.5ms pulse) to 180 degrees (2.5ms pulse)



Test Code

```
# Probl 1
from machine import Pin, PWM
from time import sleep_ms

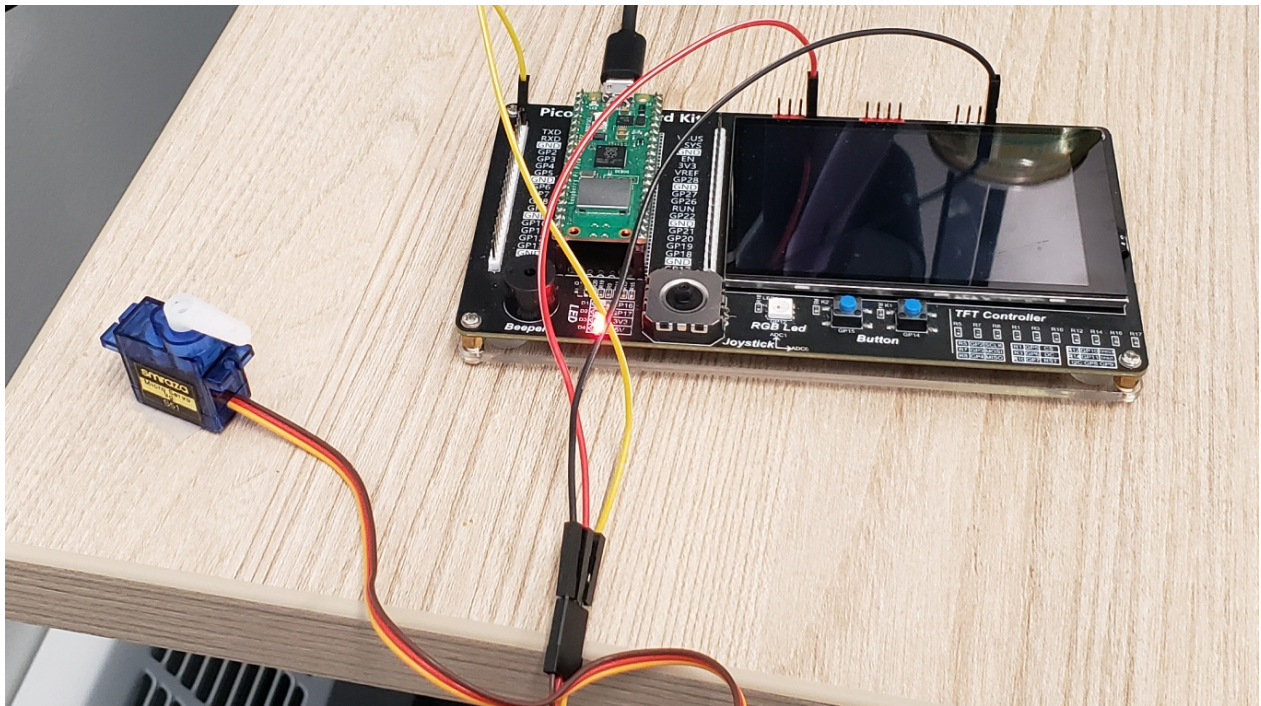
Control = PWM(Pin(0))
Control.freq(50)

x = 500_000
dx = 1_000

while(1):
    if(x <= 500_000):
        dx = abs(dx)
        print('forward')
    if(x >= 2_500_000):
        dx = -abs(dx)
        print('reverse')
    x += dx
    Control.duty_ns(x)
    sleep_ms(1)
```

shell

```
MPY: soft reboot
forward
reverse
forward
reverse
```



Hardware Connections: Power, Ground, GP0 = signal

Angle Control: Static System

2) Write a python program so that the angle of the servo motor is proportional to the joystick position:

$$PW = a \cdot A/D + b$$

such that

- Pulse width = 0.5ms when A/D = 0x0000
- Pulse width = 2.5ms when A/D = 0xFFFF
- Pulse width is proportional to the A/D reading in between

Code

```
from machine import Pin, PWM, ADC
from time import sleep_ms

Control = PWM(Pin(0))
Control.freq(50)

a2d0 = ADC(0)
a2d1 = ADC(1)

k = 100 / 0xFFFF
kV = 3.3 / 0xFFFF
x = 500_000
dx = 1_000

while(1):
    a0 = a2d0.read_u16()
    pct = k * a0
    V = kV * a0

    y = int( 500_000 + 20_000 *pct )

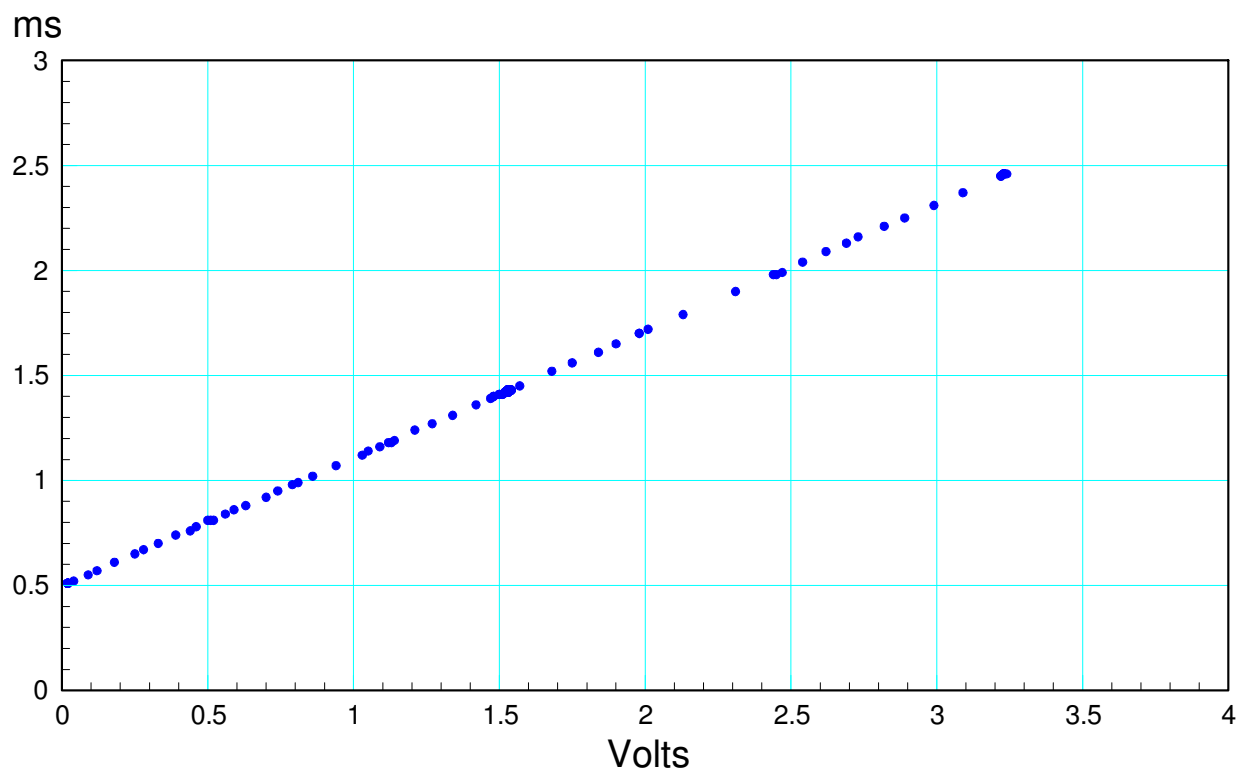
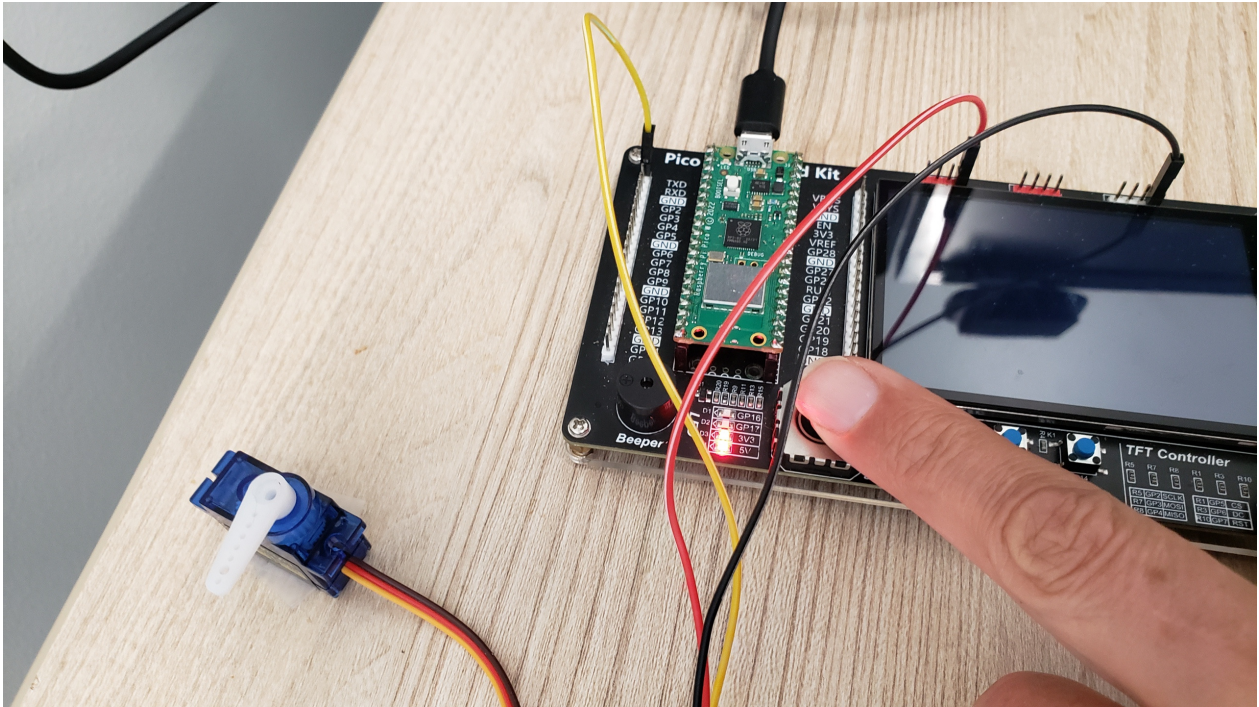
    Control.duty_ns(y)
    print('{: 10.2f}'.format(V), '{: 10.2f}'.format(y / 1_000_000))
    sleep_ms(10)
```

shell

```
MPY: soft reboot
    Volts      pulse  (ms)
    1.53        1.43
    1.53        1.43
    1.54        1.43
    1.53        1.43
    1.53        1.43
    1.53        1.43
    1.53        1.43
    1.53        1.43
    1.53        1.43
```

3) Verify your code works

- Short video works
- Print out time, analog input, and pulse width in the shell window also works



Angle Control: 1st-Order Differential Equation

4) Write a python program so that the angle of the servo motor is related to the joystick position according to a 1st-order differential equation with a time-constant of one second:

$$y' + y = x$$

where

- x is the analog input from the joystick from 0% to 100% and
- y is the angle of the digital servo motor, ranging from 0% (0.5ms pulse) to 100% (2.5ms pulse)

```
from machine import Pin, PWM, ADC
from time import sleep

Control = PWM(Pin(0))
Control.freq(50)

a2d0 = ADC(0)

k = 100 / 0xFFFF

x = 500_000
dx = 1_000
dt = 0.01
y = 0
N = 0

while(1):
    x = k * a2d0.read_u16()
    dy = -y + x
    y += dy * dt

    duty = int( 500_000 + 20_000 * y )

    Control.duty_ns(duty)

    N = (N + 1) % 10
    if(N == 0):
        print('{: 10.3f}'.format(x), '{: 10.3f}'.format(y))
        sleep(dt)
```

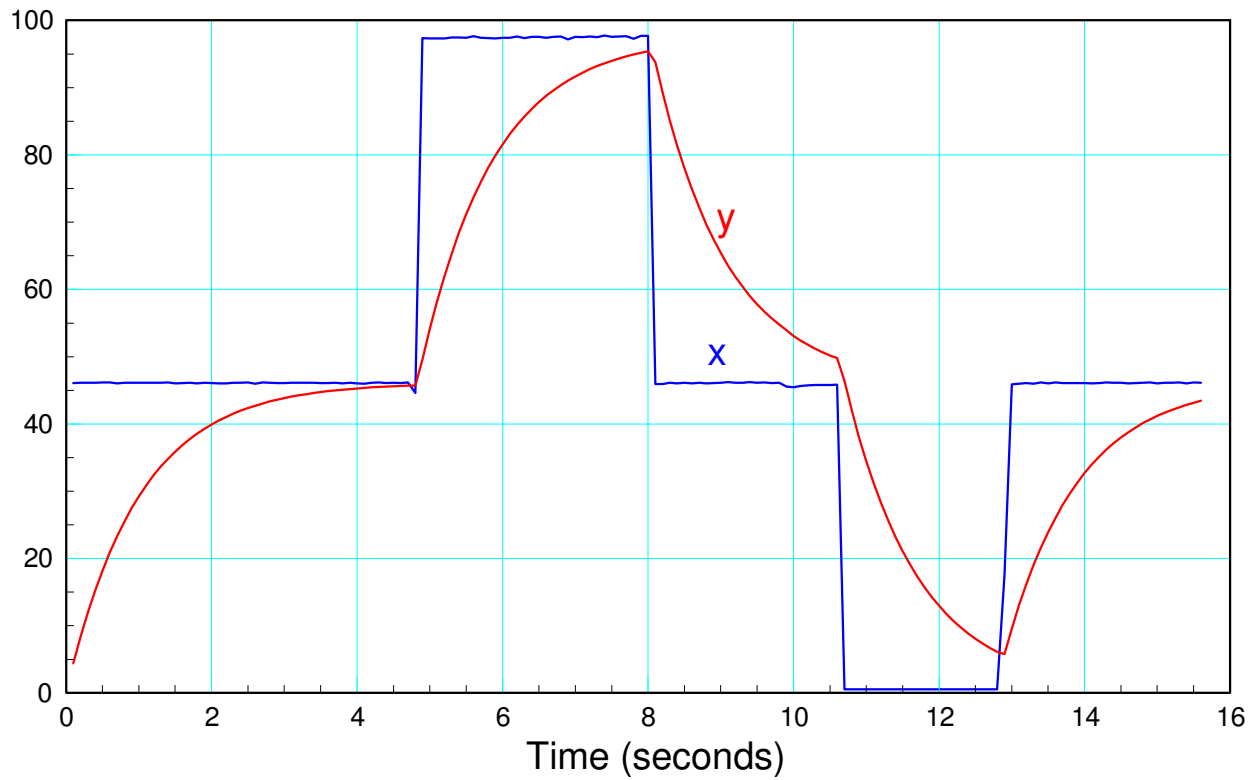
shell

x (%)	y (%)
46.081	4.406
46.105	8.398
46.130	12.002
46.130	15.263
46.154	18.212
46.178	20.877

5) Verify your code works

- Short video works
- Print out time, analog input, and pulse width in the shell window also works

The input on a percentage basis (x) is recorded along with the output of the filter (y)



Joystick position (x = blue) and filtered position (y = red)

Angle Control: 2nd-Order Differential Equation

6) Write a python program so that the angle of the servo motor is related to the joystick position according to a 1st-order differential equation with a time-constant of one second:

$$y'' + 2y' + 10y = 7x$$

where

- x is the analog input from the joystick from 0% to 100% and
- y is the angle of the digital servo motor, ranging from 0% (0.5ms pulse) to 100% (2.5ms pulse)

```
from machine import Pin, PWM, ADC
from time import sleep

Control = PWM(Pin(0))
Control.freq(50)

a2d0 = ADC(0)

k = 100 / 0xFFFF

dt = 0.01
y = dy = 0
N = 0

while(1):
    x = k * a2d0.read_u16()
    d2y = 7*x - 10*y - 2*dy
    dy += d2y * dt
    y += dy * dt

    duty = int( 500_000 + 20_000 *y )

    Control.duty_ns(duty)

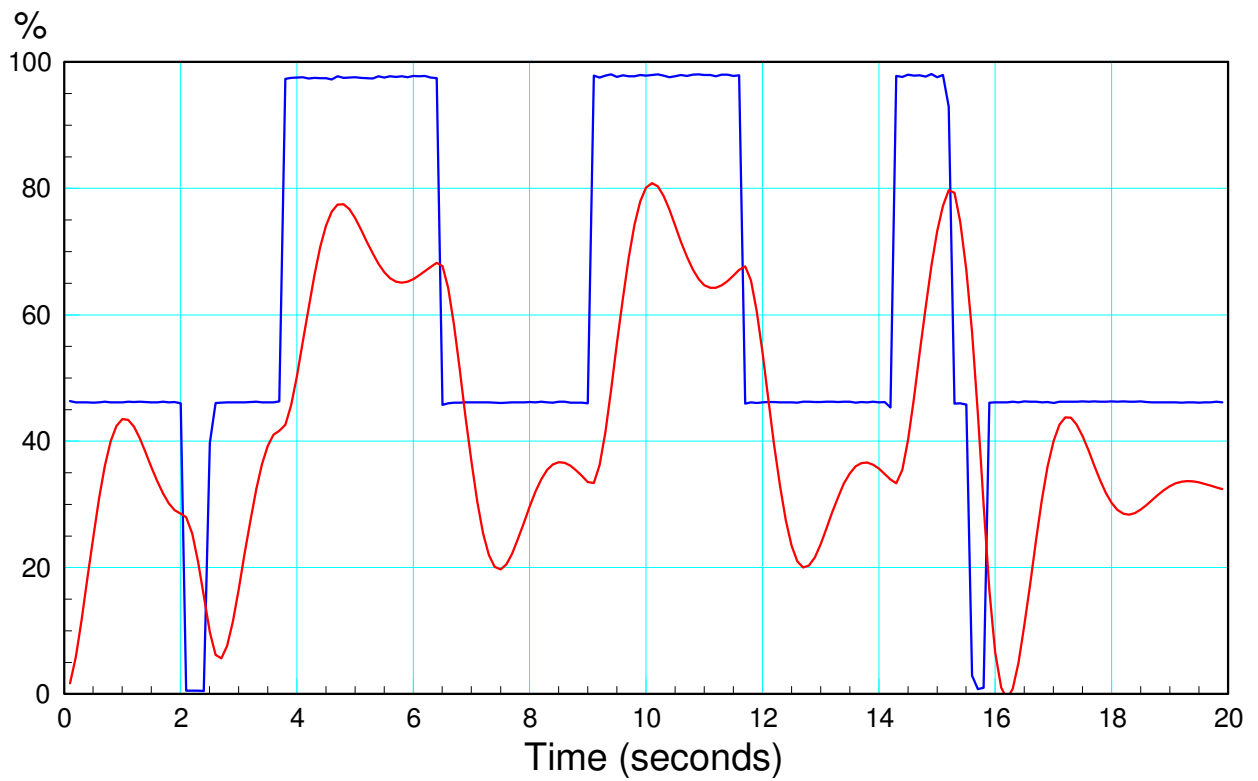
    N = (N + 1)% 10
    if(N == 0):
        print('{: 10.3f}'.format(x), '{: 10.3f}'.format(y))
        sleep(dt)
```

shell

x (%)	y (%)
46.325	1.659
46.130	5.795
46.154	11.581
46.105	18.188
46.056	24.867
46.130	30.999
46.203	36.126
46.130	39.969

7) Verify your code works

- Short video works
- Print out time, analog input, and pulse width in the shell window also works



Joystick position (x = blue) and filtered position (y = red)