UNSW SYDNEY

MTRN3100

Tutorial 3 – Motors and Control

Motors and Encoders

UNSW
SYDNEY

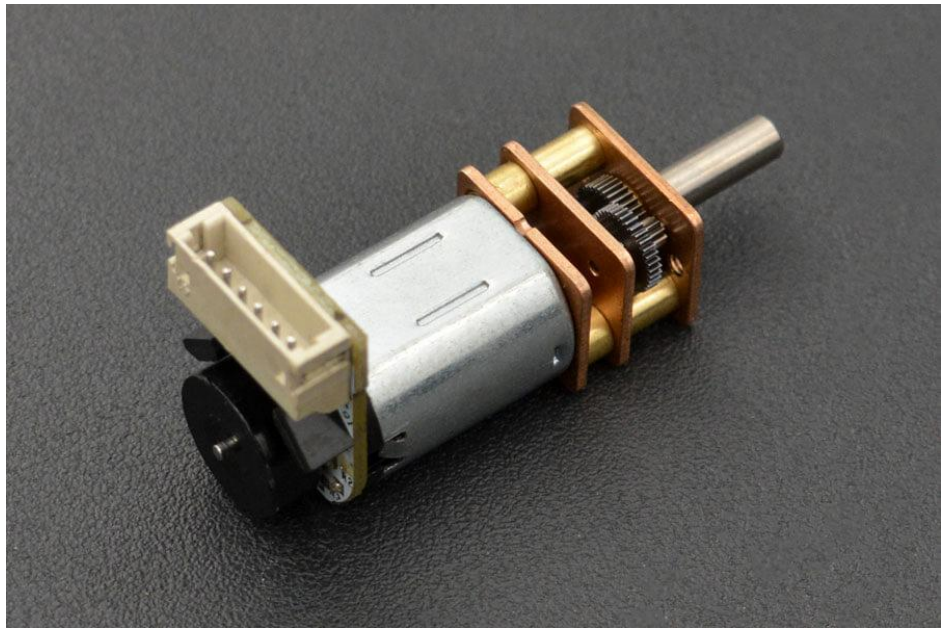
DC Motors

Wiring:

M1,M2 – Motor terminals

GND, A, B, VCC – Encoder

When using the motors be gentle with the gearbox. Make sure not to restrict the encoder disk on the back of the motor.

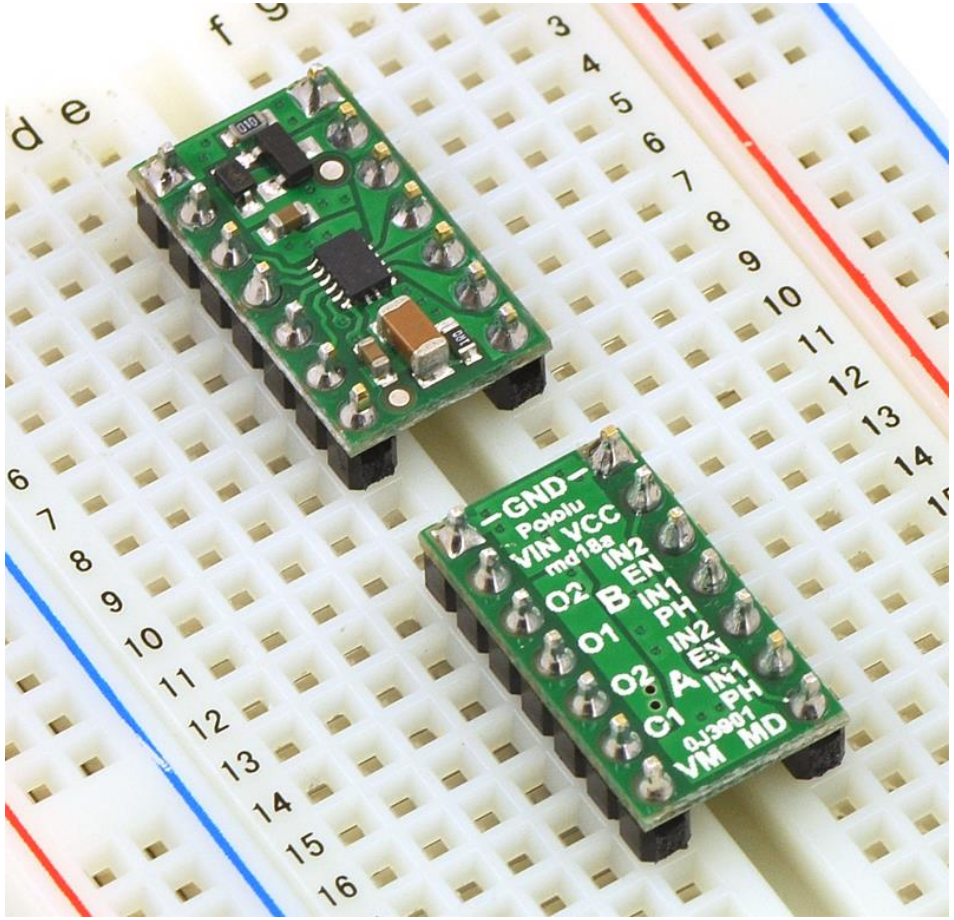


Motor controller

Different motor controllers can take in different signal such as PWM, CAN, UART, I2C.

The motor controllers used in this project use a speed pin and a direction pin. This mode can be set by bringing the MD pin on the motor controller to +5v.

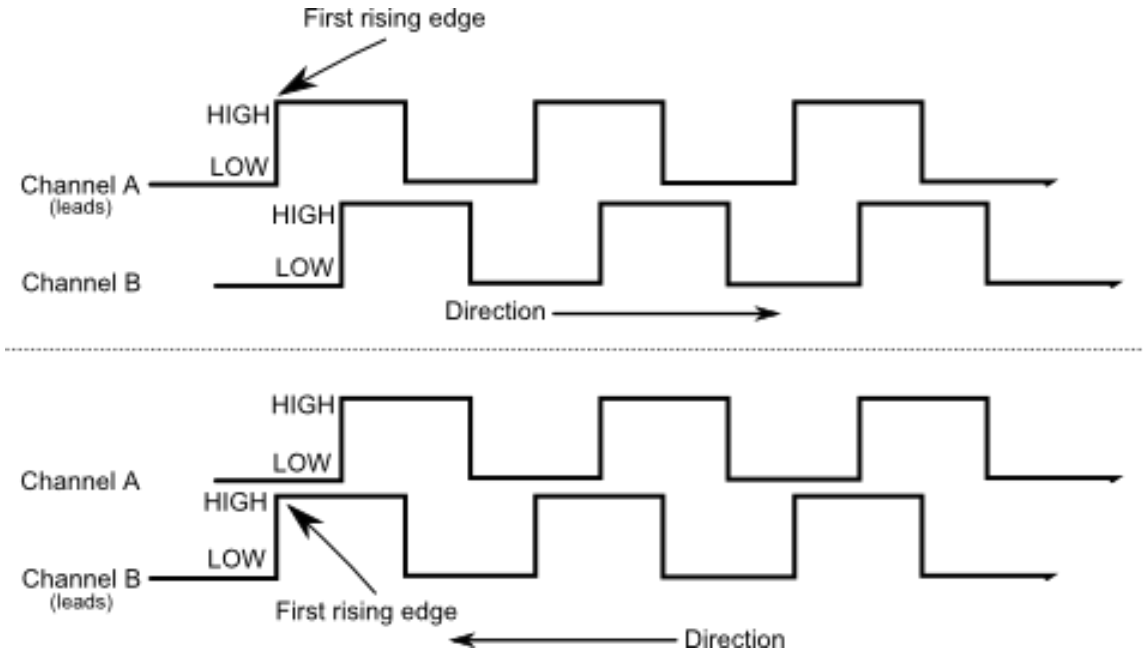
The speed pin (PWM) is used to control the current across the motor. The direction pin (Boolean) controls the flow of current (Clockwise - Anticlockwise).



Quadratic Encoders

Quadratic encoders are incremental encoders which trigger a hall effect sensor with rotation.

They do not store absolute position, only relative. As such there is no initial known rotation on start up.



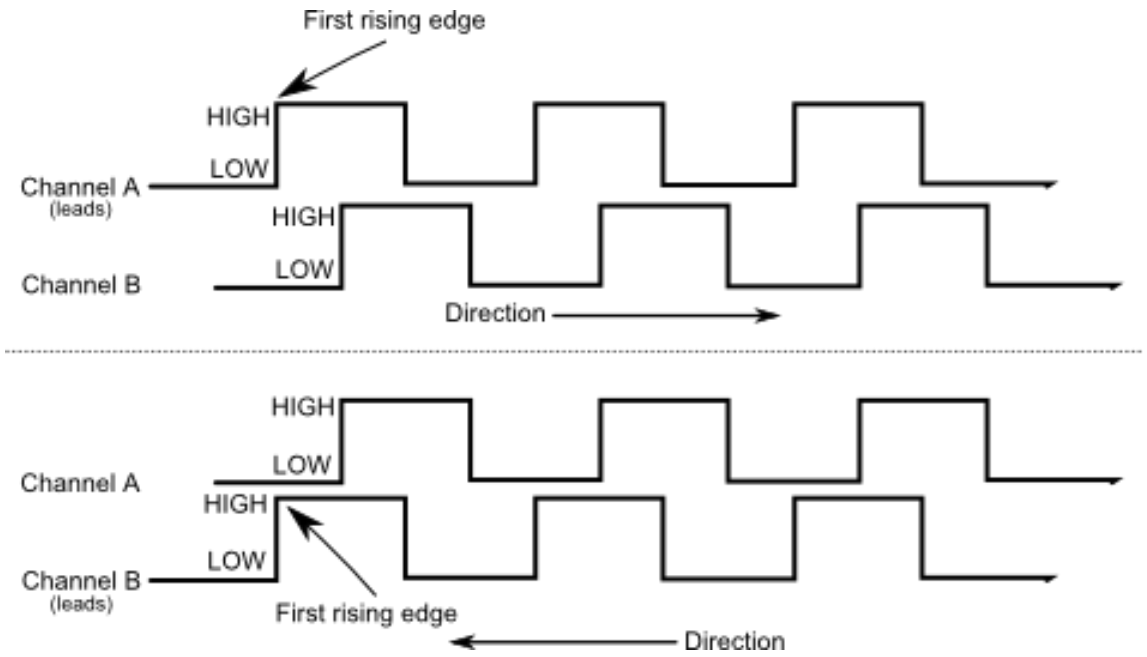
https://docs.wpilib.org/en/stable/docs/software/hardware-apis/sensors/encoders-software.html

UNSW SYDNEY

Reading Quadratic Encoders

At the rising edge of channel A, Channel B can be measured. The direction of rotation can be derived if Channel B is high or low.

Angular speed can be derived by comparing the time between two rising edges.



https://docs.wpilib.org/en/stable/docs/software/hardware-apis/sensors/encoders-software.html

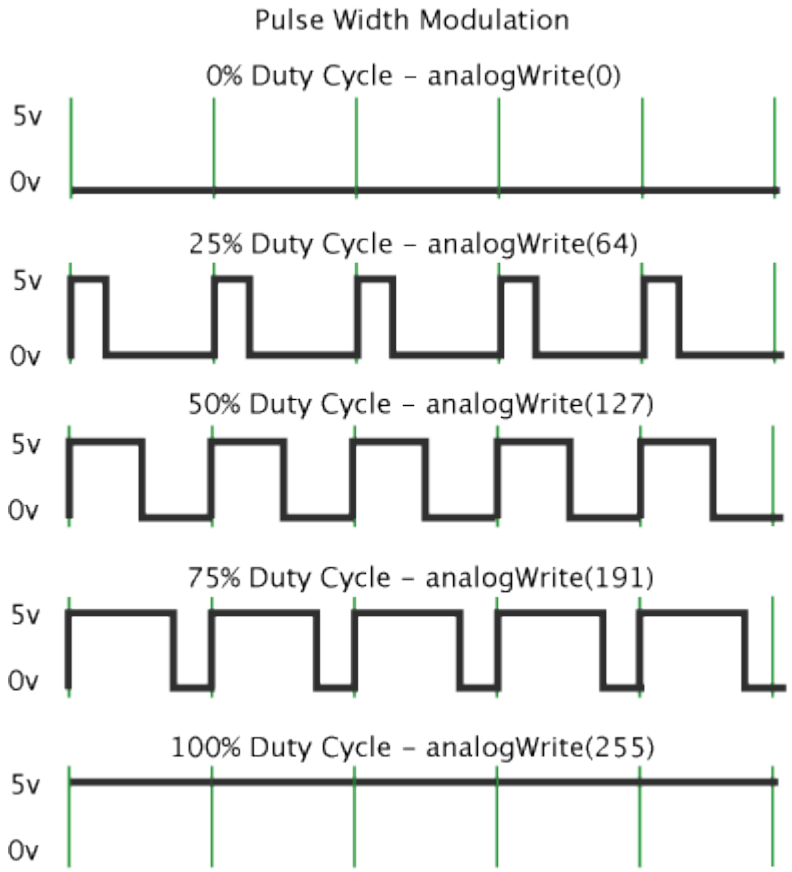
Interrupts and PWM

Pulse Width Modulation (PWM) is a digital signal method which can be used to output a variable voltage. This is achieved through time varying pulses of a digital output.

Use analogWrite(pin,speed); on a PWM pin.

<u>Arduino Nano</u>
Interrupt Pins – 2, 3

PWN Pins – 3,5,6,9,10,11



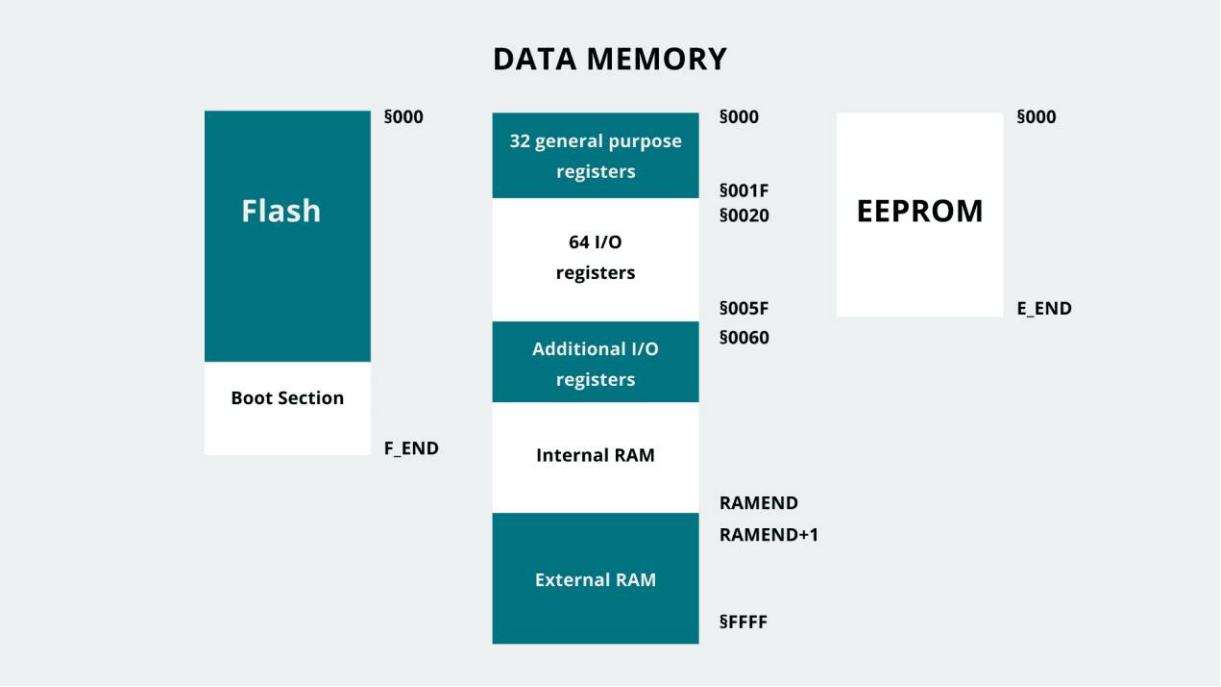
https://docs.arduino.cc/learn/microcontrollers/analog-output/

Volatile Flags

The volatile flag allows to variable to be updated faster than a standard variable.

Volatile variables are stored in ram not cached in a storage register.

This is required for the encoder interrupts.



https://docs.arduino.cc/learn/programming/memory-guide/

Control

UNSW
SYDNEY

Control

Open Loop:

No feedback. As such disturbances can not be detected and accounted for.

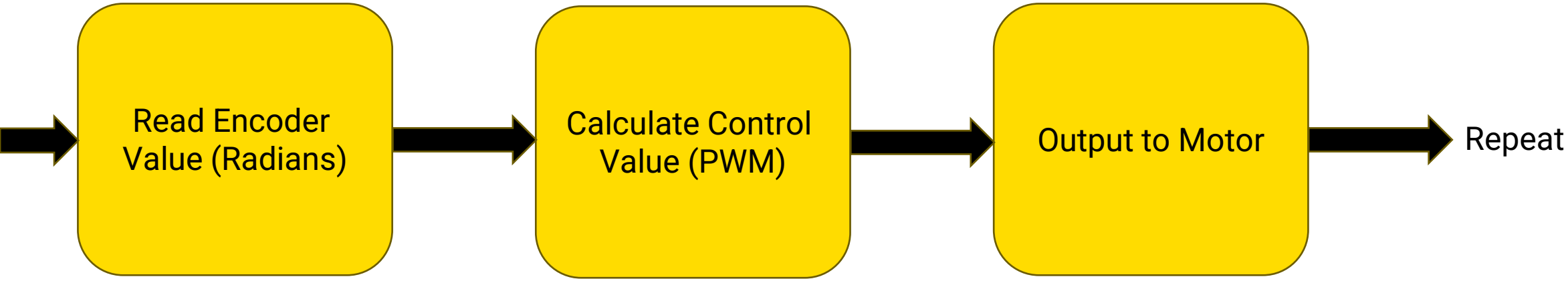
Eg - To move 1 rotation turn the motor on at the maximum speed for 2.5 seconds. After the 2.5 seconds the motor could have moved only 0.8 rotations if there is friction.

Closed Loop:

Used external feedback (sensors) to provide information to the controller. Can adjust to account for disturbances.

Eg - To move 1 rotation turn the motor on. When we get close to the goal start slowing down. If there is some error from friction adjust the torque to overcome it.

Control Loop

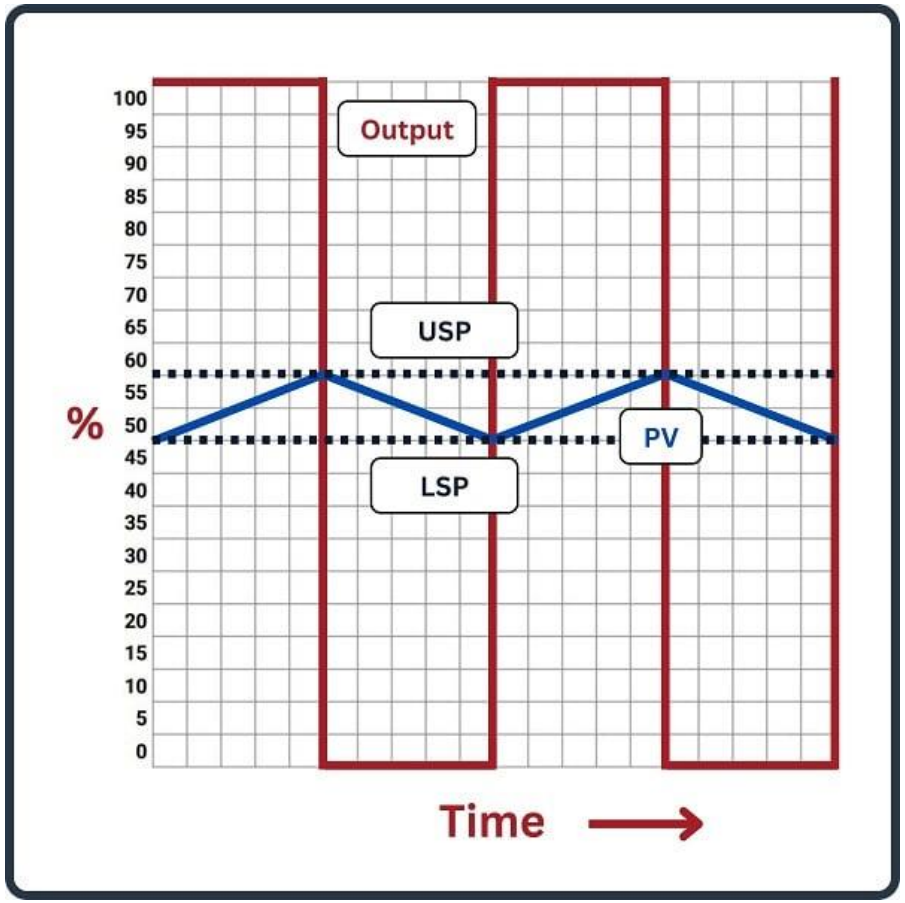


Ideally your control loop operates at a high frequency (No delays)

Bang-Bang Controller

Bang-Bang controllers set a constant output based on the systems relation to the setpoint.

Rotational inertia can cause oscillations around the set point which cannot be corrected without external influence.



https://control.com/textbook/closed-loop-control/onoff-control/

Bang-Bang Pseudocode

```
1  loop:
2      error = setpoint - measured_value
3      if (error < deadband):
4          output = speed
5      else if (error > deadband):
6          output = -speed
7      else:
8          output = 0
9
```

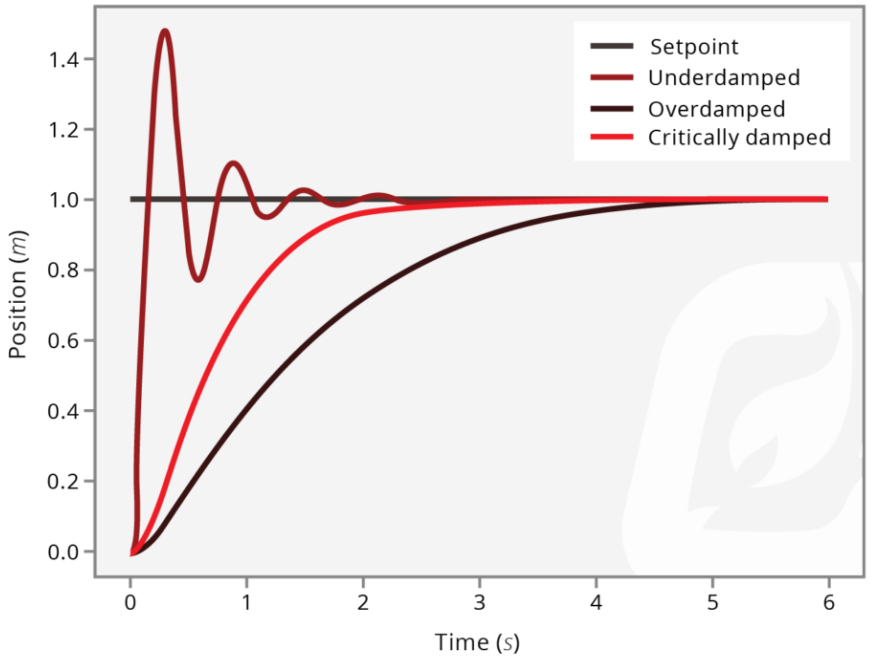
PID Controller

PID controllers use the error between the setpoint and the measured value to influence the control value.

P – If you're not at the set point get there.

I – If you haven't been at the setpoint for a while get there.

D – If you're approach the setpoint too fast slow down.



PID Controller Response Types

https://profireenergy.com/what-is-a-pid-controller/

PID Pseudocode

```
loop:
    error = setpoint - measured_value
    dt = 0.02 # Time since last control loop

    proportional = error
    integral = integral + error × dt
    derivative := (error - previous_error) / dt
    output := Kp × proportional + Ki × integral + Kd × derivative
    previous_error := error


```

How to tune PID?

1. Set all terms to zero.
2. Increase P until systems starts to oscillate.
3. Increase D until oscillations calm down and response is fast.
4. Keep increasing P and D if a faster response is desired.
5. If steady state error is present increase I.
6. Increase P and loop if a faster response time is desired.

Sometimes you may not need all 3 terms. For certain applications P, PI, PD, or PID controllers may all work.

Lab tasks

UNSW
SYDNEY

C++ code

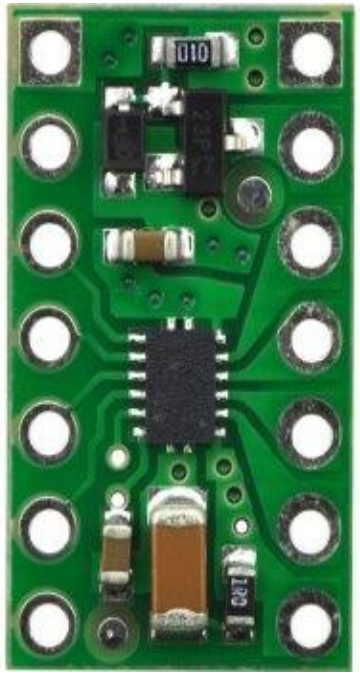
The following syntax is used to call a function in a class:

motor.setPWM(120);

C++ code (Arduino) is assumed knowledge for this course given MTRN2500. If you need a touch up:

Refer to: https://www.w3schools.com/cpp/default.asp

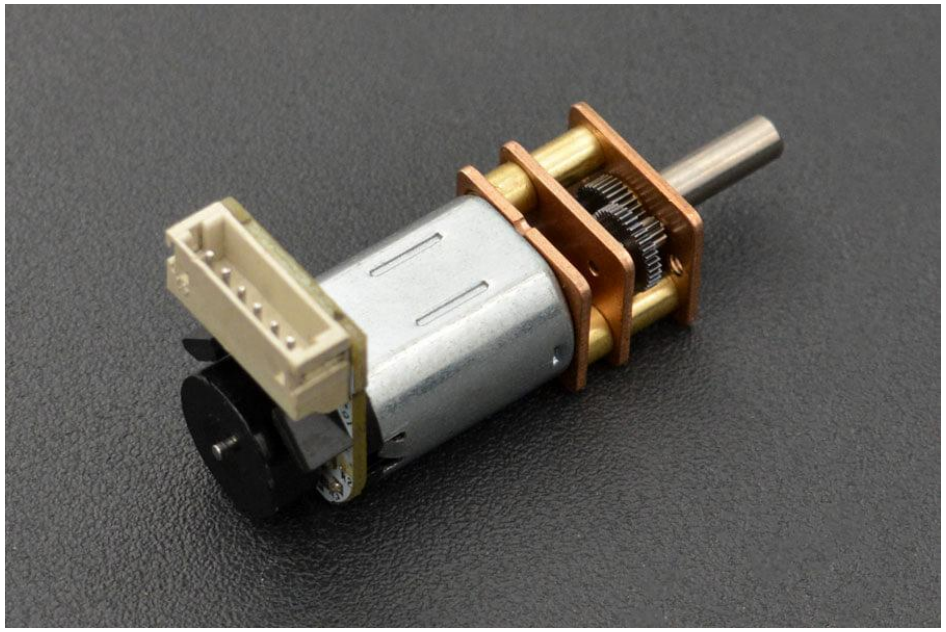
Wiring Motor Controller

Arduino GND	GND		GND	**Arduino GND**
Arduino +5V	VCC		VIN	**Arduino +5V**
Arduino Pin 9	BIN2/BENABLE		BOUT2	**Motor M1/M2**
Arduino Pin 10	BIN1/BPHASE		BOUT1	**Motor M1/M2**
No connection	AIN2/AENABLE		AOUT2	**No connection**
No connection	AIN1/APHASE		AOUT1	**No connection**
Arduino + 5V	MODE		VMM	**No connection**

The motor controllers used in this course are dual drivers, however for this lab you will only need one channel.

Wiring Encoder

Encoder	Colour	Arduino
GND	Red *	GND
A	White	Pin 2 (Interrupt)
B	Purple	Pin 4 (Digital)
VCC	Green	+5V



*We am aware this is a horrid colour choice, sadly it is the only cable which we could source.

Controller debugging

Does your controller just spin and spin?
Here are some steps to debug:

- Verify that your motor class works. Pass through 120,-120 and ensure it turns both ways.
- Print out your error.
- If the error is increasing and not decreasing to zero your motor may be turning the wrong way.
- In this case try inverting your signal/switching your motor direction.

Print statements save minds, use them.

Coding Tips with Interrupts

Never place a print statement inside the interrupt function. Print statements can block the ISR causing unexpected behavior.

Do not call the quadratic encoder interrupt function manually. It only functions correctly if triggered with an interrupt, otherwise it can cause unexpected behaviour.

If you want to get the encoder counts call getRotations() or encoder.count.