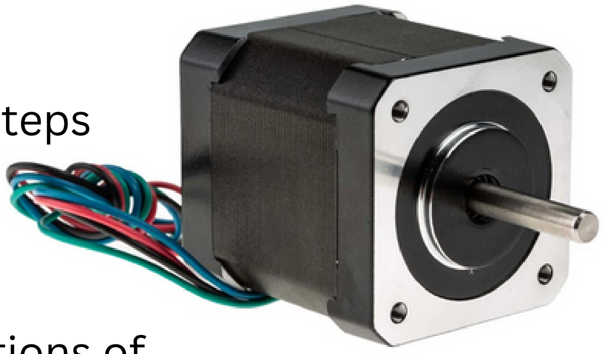
Drawing Robot

Same concepts as "gears and motion motorized"

What is a stepper motor?

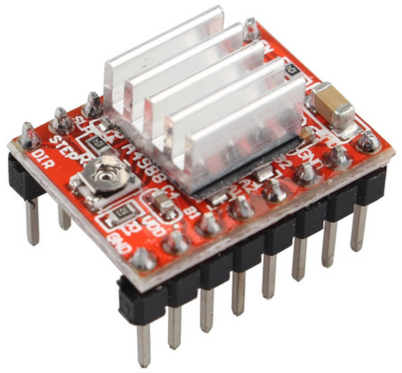
- Unlike convensional dc motors that spin continuously, stepper motors rotate in steps where each step is a combination of electromagnets being activiated.
- to get it to rotate, you activate combinations of the electromagnets in a specific order.



Nema 17 Stepper motor
200 steps per revolution

How to control it with a pico?

- Directly connecting any motor to a pi pico (apart from small servos) is a bad idea because of how much power they require in comparison to other electrical components.
- To work around this limitation, "driver boards" are used as an intermediary between motors and microcontrollers, combining the power supplied from a seperate source and the data signals provided by a micro controller (Pi Pico)
- The driver board simplifies the logic we have to implement in software to 3 main controls: motor direction, when to step, resolution of each step.
- Motor direction is self explainatory, to step you simply provide a high value to a pin on the driver board, and specifying the resolution allows you to do 1/2, 1/4, 1/8 and 1/16 steps for finer movement control.



A4988 Driver board

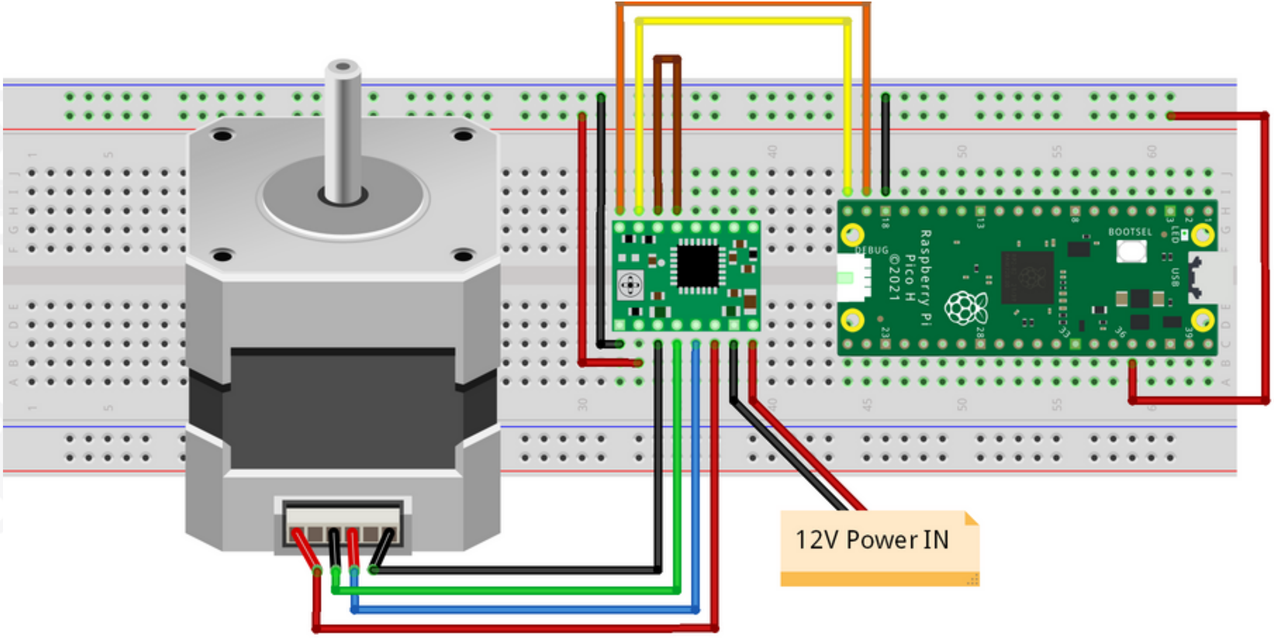
Remember to add a capacitor between the 12V power and ground pins to prevent the motor from jittering

Part 1: Making a stepper motor spin

Before we start making some art, lets remember how to get a single motor spinning.

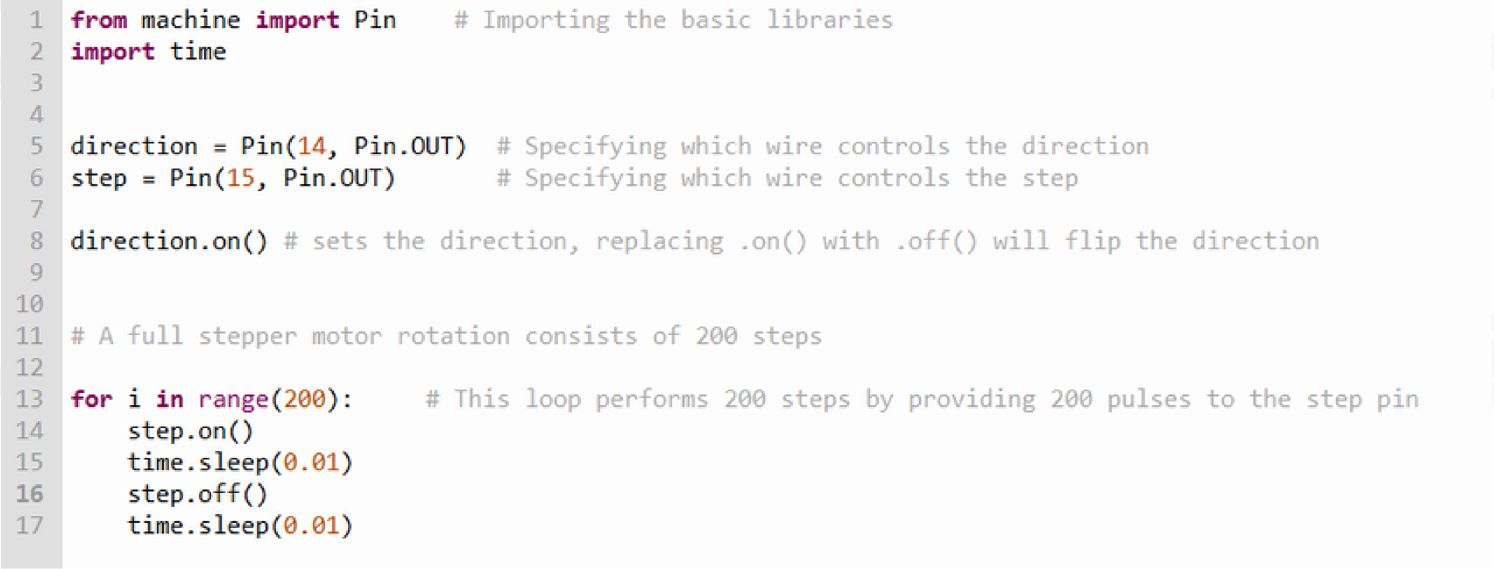
- the diagram bellow shows you what needs to be wired up where with the Stepper motor on the left, driver board in the middle and Pi Pico on the right.
- Because motors require much more power that what a usb port or Pi Pico can provide, we need to give the driver board an extenal supply.

To prevent accidental short circuits, do not connect the 12V supply until you are ready to test your code and confident with the wiring, if you are unsure please ask



You can add some masking tape to the motor to better visualize rotation

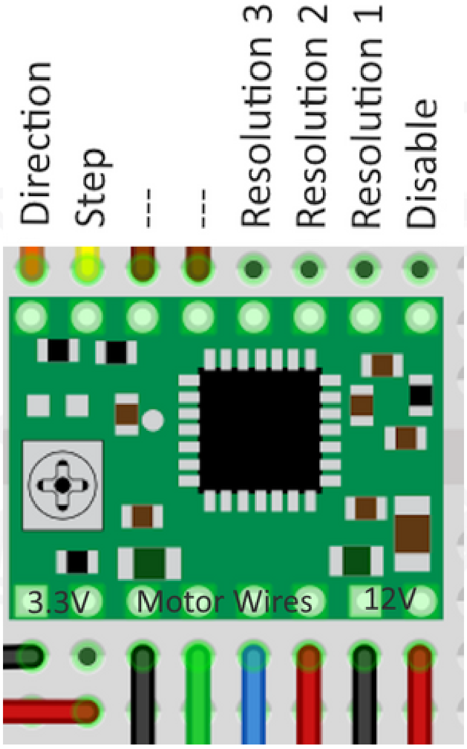
- Once everything is wired up (except 12V power) you can connect your Pi Pico to a computer and implement the following code:



```python
from machine import Pin     # Importing the basic libraries
import time


direction = Pin(14, Pin.OUT)  # Specifying which wire controls the direction
step = Pin(15, Pin.OUT)       # Specifying which wire controls the step

direction.on() # sets the direction, replacing .on() with .off() will flip the direction


# A full stepper motor rotation consists of 200 steps

for i in range(200):      # This loop performs 200 steps by providing 200 pulses to the step pin
    step.on()
    time.sleep(0.01)
    step.off()
    time.sleep(0.01)
```

- As stated earlier it is possible to specify the resolution of a step by connecting up some extra wires.
- For a 1/2 step connect **Resolution 1** to **V++**
- For a 1/4 step connect **Resolution 2** to **V++**
- For a 1/8 step connect **Res 1 and 2** to **V++**
- And for 1/16 step connect **Res 1, 2 and 3** to **V++**

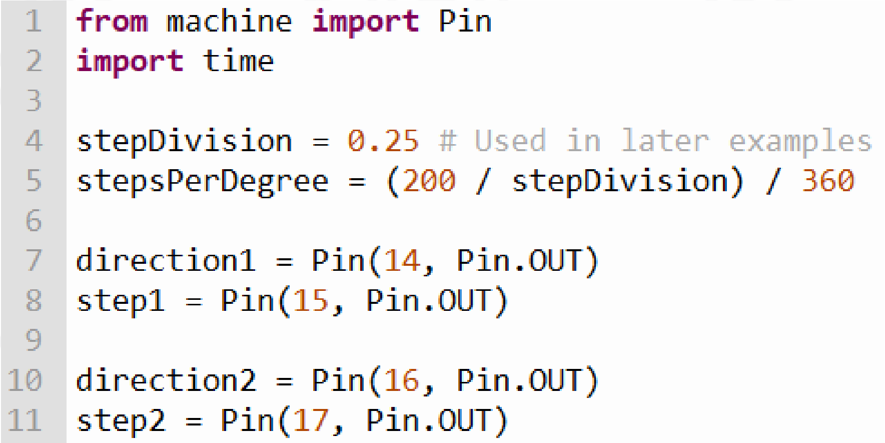
- Try out different resoltions and see the effect it has on the motor.
- You may need to ground unused **Resolution** pins if your motor is behaving oddly



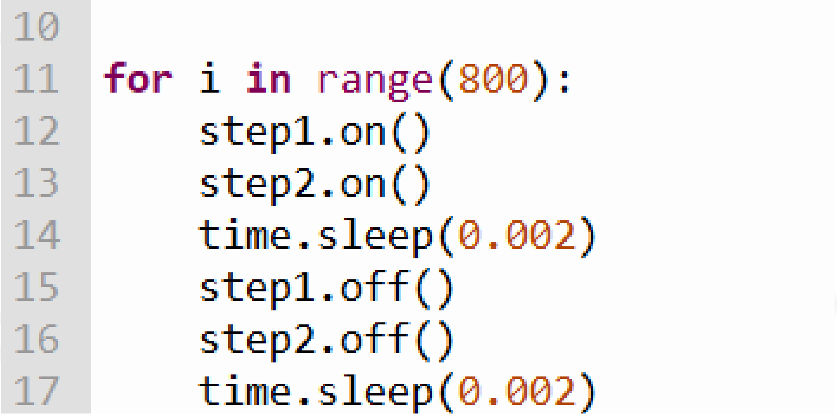
Part 2: Making 2 motors spin

By now you should have a single motor spinning, now duplicate the circuit and code to allow 2 motors to spin.

To help, your code should look like this:

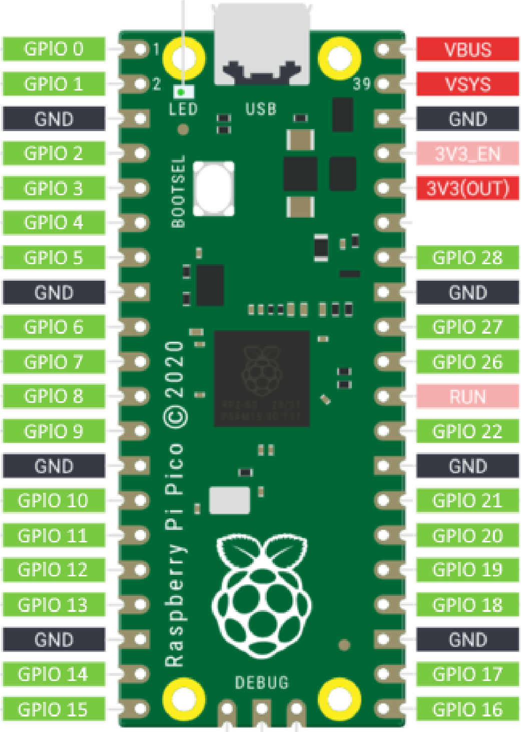


```
from machine import Pin
import time

stepDivision = 0.25 # Used in later examples
stepsPerDegree = (200 / stepDivision) / 360

direction1 = Pin(14, Pin.OUT)
step1 = Pin(15, Pin.OUT)

direction2 = Pin(16, Pin.OUT)
step2 = Pin(17, Pin.OUT)
```



```
for i in range(800):
    step1.on()
    step2.on()
    time.sleep(0.002)
    step1.off()
    step2.off()
    time.sleep(0.002)
```

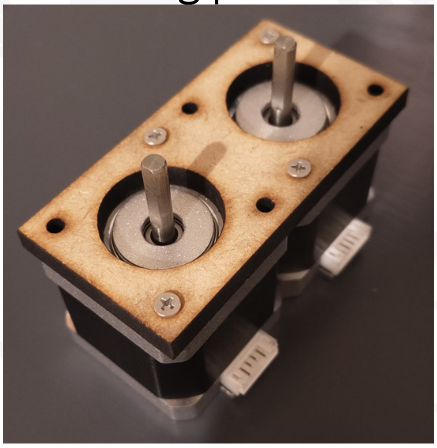
Remember to add a capacitor between the 12V power and ground pins to prevent the motor from jittering



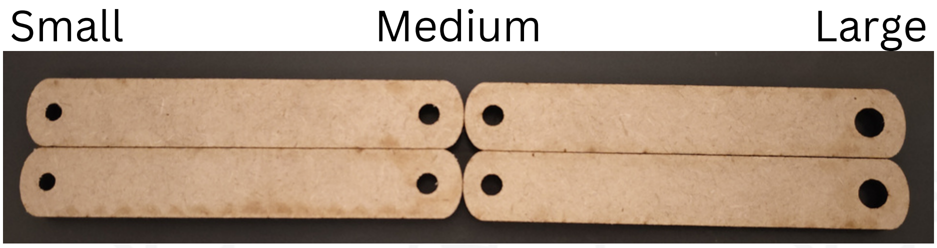
make sure the MS2 pin is connected to 3.3V before continuing!

Task 3: Making a robot draw

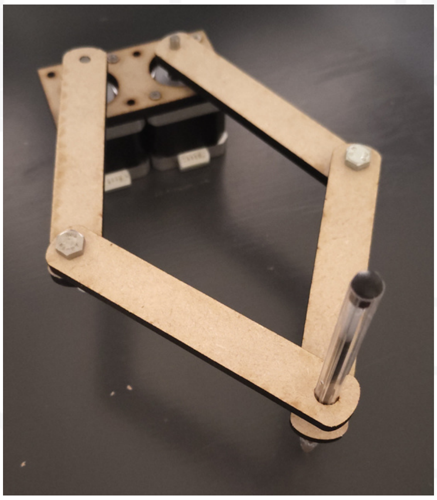
1. Connect the two motors to a mounting plate as shown below



2. Then connect each pair of arms with a bolt through the medium holes

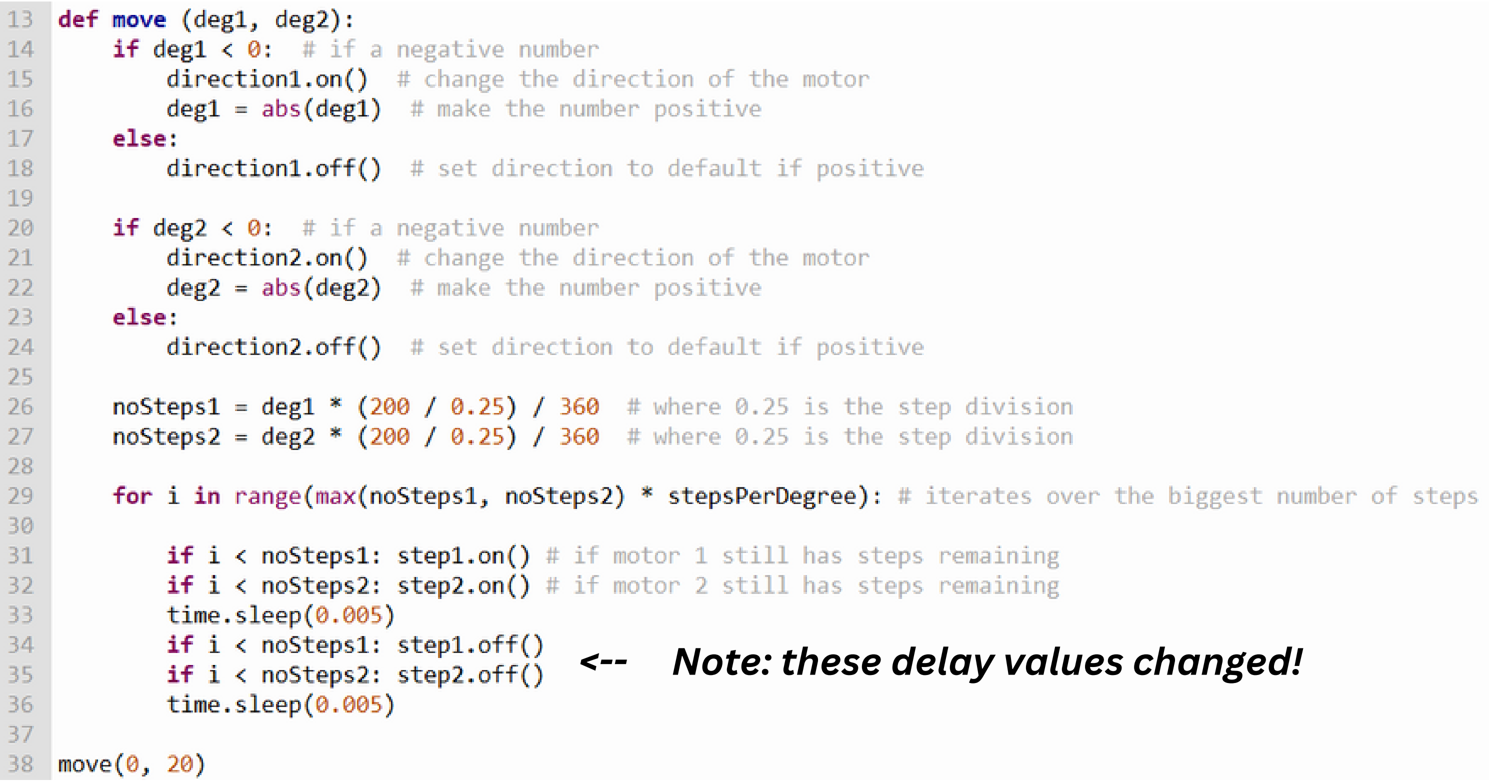


3. Then insert the small hole of each arm on to the motor shafts, **beware the hole is not completely circular and should be aligned with the shape of the motor shaft.** And insert a pen into the other ends.



Make sure the motor doesnt spin beyond the limits of the arms by only making small movements in your code (less than 20 degrees)

4. Implement the following code to make a move function that we will use later:



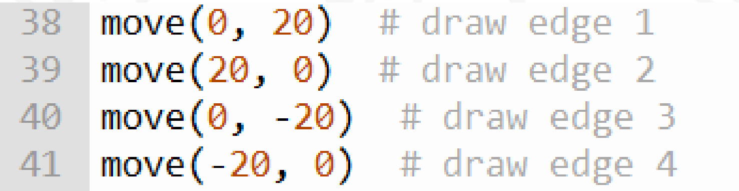
```python
def move (deg1, deg2):
    if deg1 < 0:  # if a negative number
        direction1.on()  # change the direction of the motor
        deg1 = abs(deg1)  # make the number positive
    else:
        direction1.off()  # set direction to default if positive

    if deg2 < 0:  # if a negative number
        direction2.on()  # change the direction of the motor
        deg2 = abs(deg2)  # make the number positive
    else:
        direction2.off()  # set direction to default if positive

    noSteps1 = deg1 * (200 / 0.25) / 360  # where 0.25 is the step division
    noSteps2 = deg2 * (200 / 0.25) / 360  # where 0.25 is the step division

    for i in range(max(noSteps1, noSteps2) * stepsPerDegree): # iterates over the biggest number of steps

        if i < noSteps1: step1.on() # if motor 1 still has steps remaining
        if i < noSteps2: step2.on() # if motor 2 still has steps remaining
        time.sleep(0.005)
        if i < noSteps1: step1.off()
        if i < noSteps2: step2.off()
        time.sleep(0.005)

move(0, 20)
```

<-- ***Note: these delay values changed!***

You may need to flip a motor connector on the breadboard to ensure both rotate in the same direction.

IMPORTANT: Before running more complex drawing instructions, make sure the arms are in a neutral position where they can move in both directions without colliding (from a top view the arms should be make a square)

Now you should have have a robot with code to make it draw a wonky line, lets see if we can make it draw a sqaure with the following code:



```
move(0, 20)   # draw edge 1
move(20, 0)   # draw edge 2
move(0, -20)   # draw edge 3
move(-20, 0)   # draw edge 4
```

You may need to alter the exaact values to compensate for slack

Review and extra challenges:

Congrats! Today you will have acheived robotic art in the form of a wonky square!

Challenge 1: Add a while loop to your sequence of move commands to draw over previous cycles and created a clearer shape.

Challenge 2: Using a combination of different move() statements get the robot to draw some text, a circle, or anything you can think of.

Thinking point: How could we compensate for the lines being curved in order to produce a perfectly straight line?

Pi Pico Code Docs

Worksheet produced by Lukas Hastings for use by HackSussex
https://github.com/supersand21/Robotics-Workshop