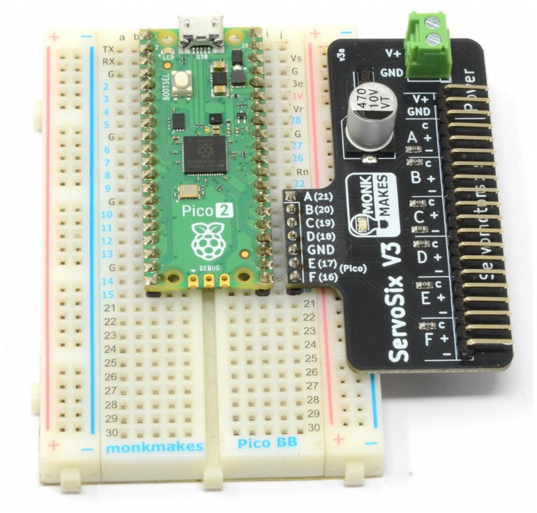
Instructions:

ServoSix V3



The ServoSix V3 is ideal for connecting servomotors to any breadboard-compatible microcontroller. However, it is especially suited to the Raspberry Pi Pico.

Document version: 1a

Table of Contents

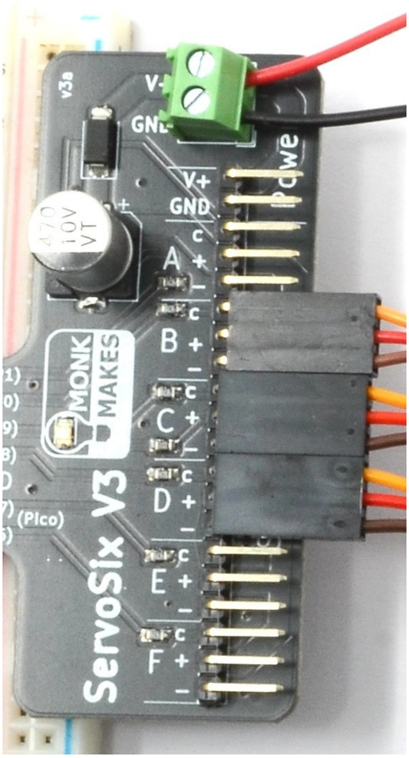
Connecting Servomotors

When connecting the servomotors to the ServoSix V3 board, make sure that you get them the right way around, as shown below.

Here, you can see servomotors connected to positions B, C and D.

For each servo, the orange control wire goes to the connection marked 'c', the middle red connection to '+' and the brown negative connection to '-'.

Sometimes these leads are a different color, so refer to the documentation for your servomotor, as connecting them the wrong way around might damage them.



Powering the Servomotors

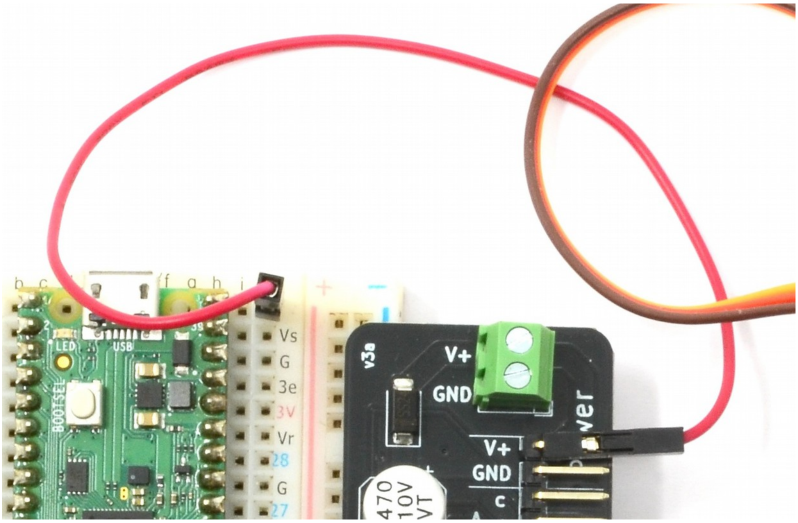
There are two options when it comes to providing power to the ServoSix V3 board, which in turn distributes power to the servomotors.

For small servo motors and some microcontroller boards you can power the ServoSix V3 direct from your microcontroller board. However, if you are using larger servomotors, then you may find that the electrical load of the servomotors moving makes your microcontroller board unreliable. It may reset at random. You may also find that the servomotors exceed the maximum current that your microcontroller can provide.

Whether your microcontroller board can provide enough power to the servomotors also depends on how the microcontroller board is powered. If the board is connected to USB, then it should have enough 5V power from the USB connection to power small servomotors. But, if the microcontroller is being powered by batteries or some other means, then it may not.

It's often better to use a separate power supply for the servomotors, such as a 6V AA battery box.

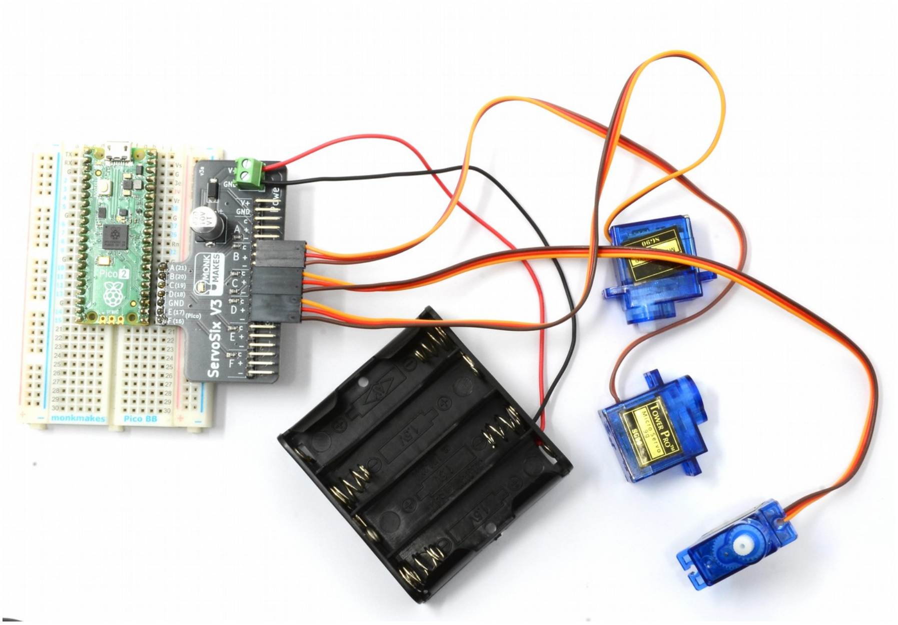
Using a Microcontroller Board's 5V supply



If you decide to power your servomotors from the power supply of your Pico (or other microcontroller board), you will need to connect the V+ pin (or screw terminal) of the ServoSix to the 5V supply of your microcontroller using a jumper wire. In this example, the 5V power supply of the Raspberry Pi Pico is used (top right pin on the Pico). This pin takes it's power from the USB connection.

Note that some boards such as the Arduino Uno have both 5V and 3.3V pins but the 3.3V pins can't supply enough current to power to drive a servomotor, so it is usually better to use the 5V supply.

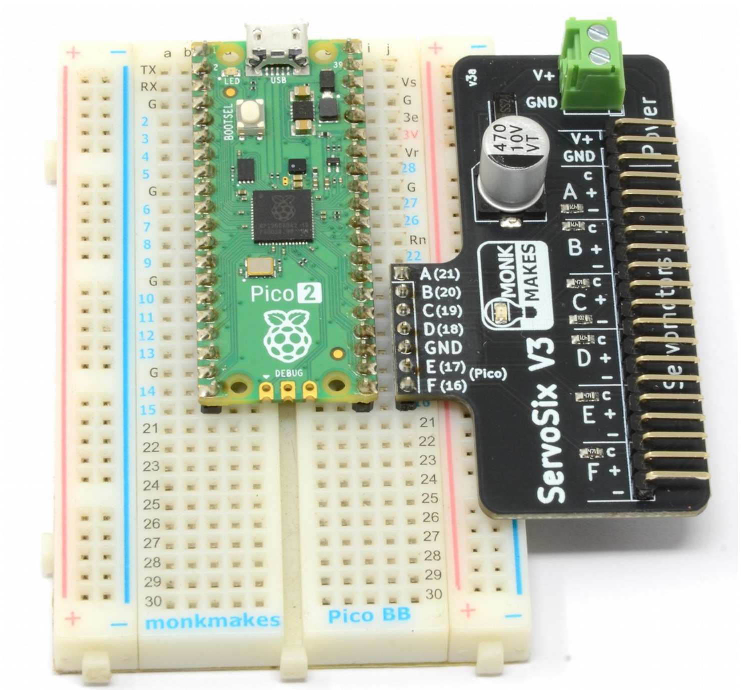
Using a Battery Box



To use a battery box, such as the one shown above, first make sure that you choose one that supplies a voltage in the allowed range for your servomotors. Many servomotors will work quite happily at 6V, in which case a 4xAA battery box is ideal.

Raspbery Pi Pico

Although you can use this kit with most microcontroller board, it is particularly well suited to the Raspberry Pi Pico and newer boards like the Pico 2 and Pico 2W. In fact the slightly unusual order of control pins, with the GND pin a little way up the connector, allow the ServoSix V3 to line up with pins on the Pico when used with solderless breadboard.



Here, a Pico 2 is fitted into a MonkMakes Breadboard for Pico which marks the pin names (https://monkmakes.com/pico_bb) making it easy to make connections to the Pico.

The servo motors A to F are controlled by Pico pins 21, 20, 19, 18, 17, 16 respectively. Remember, you also need to provide power to the ServoSix board in one of the ways described earlier.

Controlling Servomotors in MicroPython

The MicroPython PWM (Pulse Width Modulation) feature can be used to generate pulses to control a servomotor.

The following example for Raspberry Pi Pico will cause the servomotor arm to rotate through roughly 180 degrees and then return back to its starting point.

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

servoA = PWM(Pin(21))
servoA.freq(50) # pulse every 20ms

def set_angle(servo, angle, min_pulse_us=500,
                                max_pulse_us=2500):
    us_per_degree = (max_pulse_us - min_pulse_us) / 180
    pulse_us = us_per_degree * angle + min_pulse_us
    # duty 0 to 1023. At 50Hz,
    # each duty_point is 20000/65535 = 0.305 µs/duty_point
    duty = int(pulse_us / 0.305)
    servo.duty_u16(duty)

min_angle = 0
max_angle = 180
period = 0.02

while True:
    for angle in range(min_angle, max_angle):
        set_angle(servoA, angle)
        sleep(period)
    for angle in range(max_angle, min_angle, -1):
        set_angle(servoA, angle)
        sleep(period)
```

The PWM frequency is set to 50Hz (pulses per second) (`servo.freq(50)`), as that is the frequency of pulses that the servomotor expects.

Most of the work in generating the pulses to control the servomotor is contained in the function `set_angle`. This means that, if you want to create your own projects using servomotors, you can just copy this function into your own code.

The first parameter to `set_angle` is the servo to be used. In this case servo channel A connected to pin 21. The second parameter is the angle that you want to set the servomotor's arm to, which should be between 0 and 180. The other parameters set the minimum and maximum pulse width -- you should not need to change these unless you have an unusual servomotor. The function starts by working out the number of microseconds of pulse which will be required for each degree of angle. It then calculates the total pulse length in microseconds for the angle required. Finally, it calculates the duty (the PWM value) between 0 and 65535, and uses `servo.duty_u16` to set that pulse width on the servomotor’s control pin.

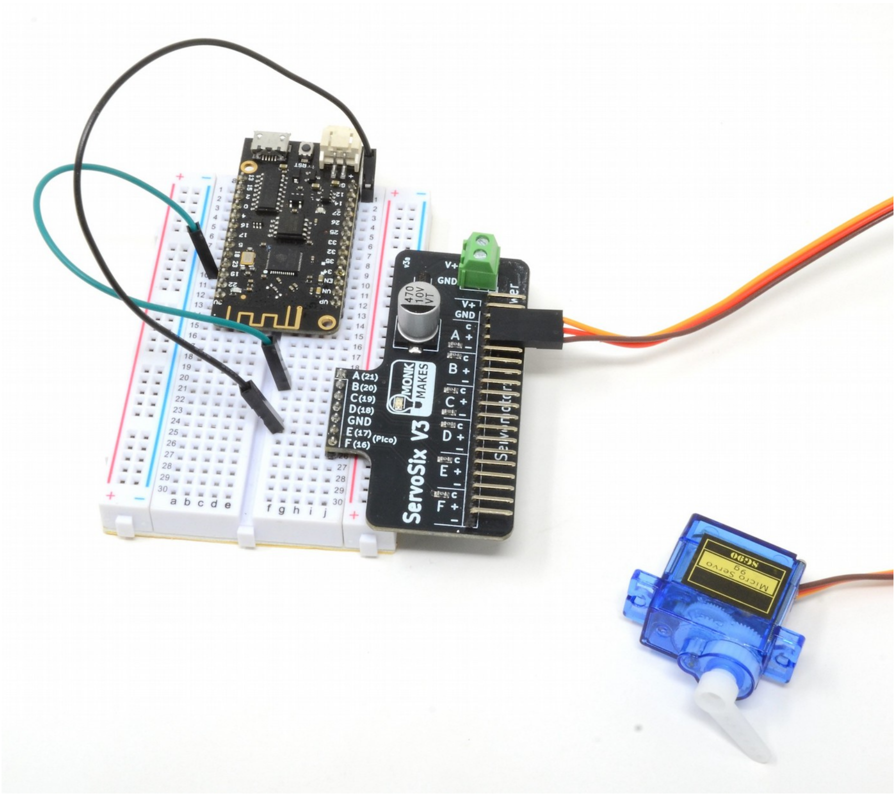
The main loop steps the angle from 0 to 180 and then back down from 180 to 0. Note that many servomotors sweep through less than the full 180 degrees.

The example above is intended for the Raspberry Pi Pico. If you want to use it for other boards, such as the ESP32, you will need to change the pin number above to match the pin connected to the servomotor's control pin.

The program above can be downloaded from:
https://github.com/monkmakes/servosix_v3_examples

Other Microcontrollers

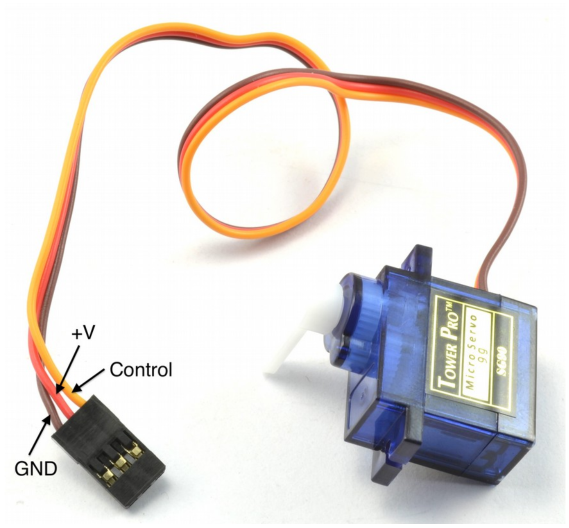
Although the ServoSix V3 is designed for the Raspberry Pi Pico, you can also use it as an easy way to connect servomotors to other boards such as ESP32 and Arduino boards. The pins won't line-up in breadboard like a Pico, so you will need to use jumper wires to link the control pins and GND of your microcontroller to the ServoSix as shown below, for a common ESP32 board.



You will need to also provide power to the ServoSix using a battery box or connection from the microcontroller board.

The MicroPython example for Pico in an earlier section should work fine, but you may need to use a different pin number.

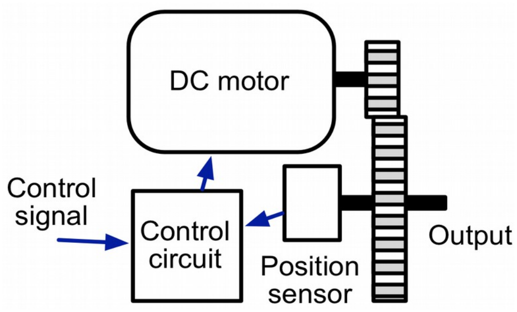
Servomotors



Servomotors have a 3-way connector. Two of these supply power to the servo and the third is a control signal that sets the position of the servomotor's arm. Wiring color can vary with different servomotors, but the most common are as follows:

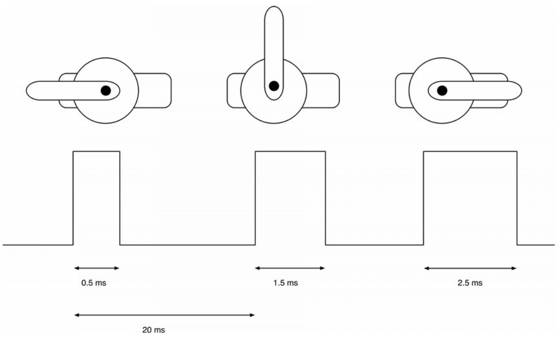
Lead color	**Function**
Brown	Ground (GND)
Red	+V (3 to 6V)
Orange	Control signal

Despite their small size, servomotors are surprisingly complex little machines, controlled using signal pulses. Here's a diagram of how a servo motor works.



A DC motor (normal motor that can rotate continuously) is attached to a gear box, which drives the servo arm, but is also coupled to a position sensor that provides feedback to an electronic control circuit that keeps the servo arm at the correct angle.

Control of the position uses a series of pulses arriving every 20 milliseconds.



The length of each of these pulses will determine the position of the servo arm. So a short pulse of just 0.5 milliseconds will put the arm at one end of its travel. A pulse of 1.5 milliseconds will put the arm at its center position and a maximum pulse length of 2.5 milliseconds will put it at the other end of its travel.

Troubleshooting

Problem: The servomotors don't move, and the ServoSix V3 board's orange LED is NOT lit.

Solution: This means that the ServoSix V3 board is not getting power. This could be for a number of reasons:

- The batteries are depleted
- The battery box is connected the wrong way around
- One or more of the batteries are the wrong way around

Problem: The servomotors don't move, and the ServoSix V3 board's orange LED is lit.

Solution: Power is getting to the ServoSix V3 board, so this means that the cause is probably one of the following:

- The connections between your microcontroller board and the ServoSix V3 board are incorrect. Check your connections.
- The servomotors are connected the wrong way around. See Page XX.
- Your microcontroller has not been programmed to move the servomotors.

Problem: When I set the servo angle to 0 or 180 the servo judders.

Solution: This is quite normal. Many servomotors will judder when they get to the ends of their range of angles. If you get this problem, then try reducing the range of angles that you try to set your servomotor to. Perhaps 10 to 170 rather than 0 to 180.

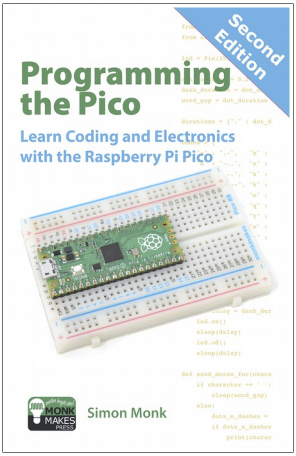
Problem: When I move the servo angle between 0 and 180 in the software, the actual angle moved my the servo is more than (or – more likely – less than) 180 degrees.

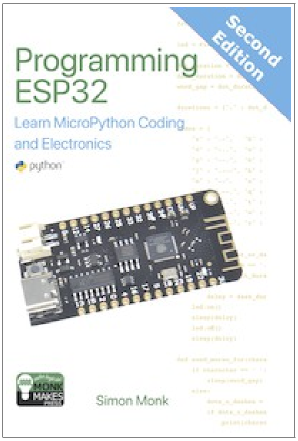
Solution: This is quite normal. Most servomotors do not exactly match the angle the are commanded to by their control pulses. See also the previous problem.

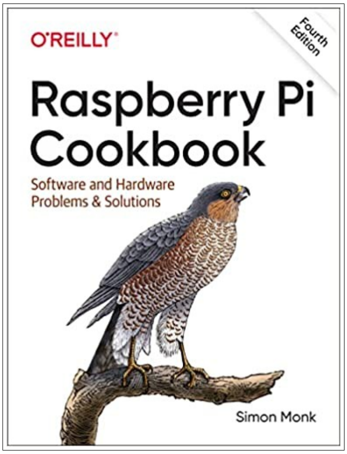
For other support questions, please check out the product's web page (http://monkmakes.com/servosix3_kit) and you can send support questions to support@monkmakes.com

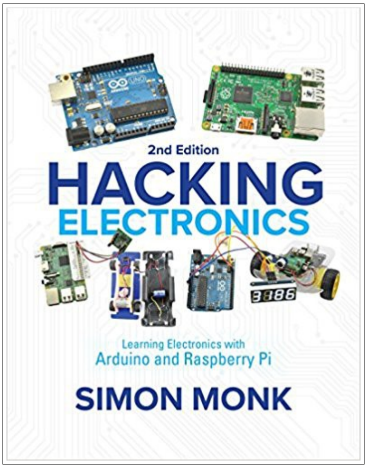
LEARNING

You can find out more about books by Simon Monk (the designer of this kit) at: http://simonmonk.org.





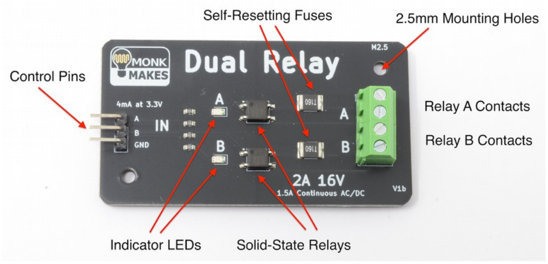


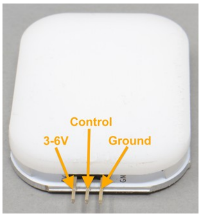


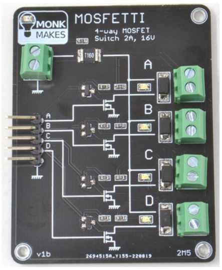
MonkMakes

For more information on this kit, the product's home page is here: https://monkmakes.com/servosix3_kit

As well as this kit, MonkMakes makes all sorts of kits and gadgets to help with your maker projects. Find out more, as well as where to buy here: https://monkmakes.com you can also follow MonkMakes on Instagram as @monk_makes_ltd







MonkMakes products: Applified Speaker 2, Dual Relay, Illuminata, LED Arc, Mosfetti, Plant Monitor.

See: https://monkmakes.com/products