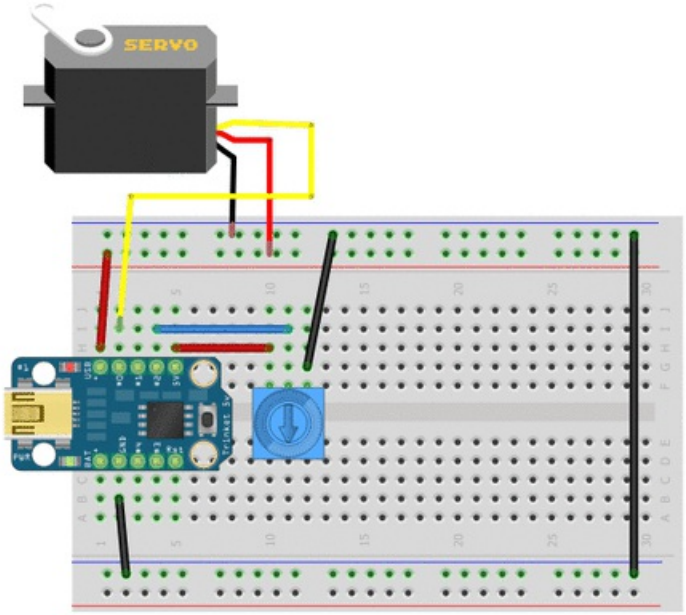
adafruit learning system

Trinket (& Gemma) Servo Control

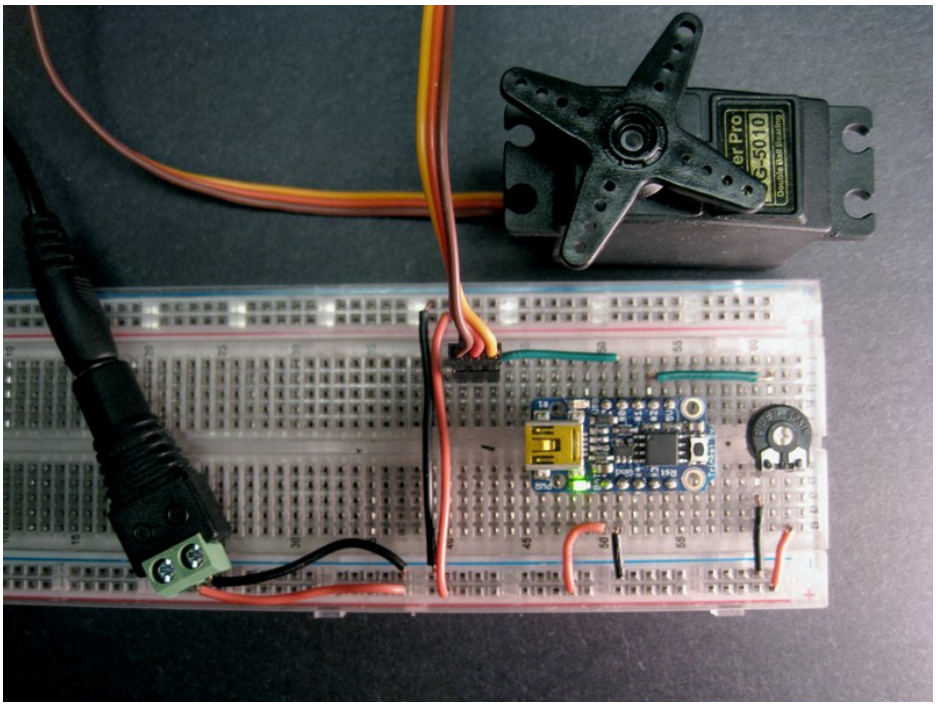
Created by Mike Barela



Last updated on 2018-10-17 09:58:35 PM UTC

Guide Contents

Overview



This guide was written for the Trinket Mini and Gemma v2 boards. It has been updated to also support the Trinket M0 and Gemma M0 using CircuitPython. We recommend the Trinket M0 or Gemma M0 as they are easier to use and are more compatible with modern computers!

The Adafruit Trinket's small size makes it ideal for lightweight or small projects including robotics. This project demonstrates the use of a standard hobby servo with the Trinket.

The standard Arduino IDE servo library will not work with 8 bit AVR microcontrollers like the ATtiny85 on the Trinket and Gemma due to differences in available timer hardware. Fortunately the Adafruit_SoftServo library (https://adafru.it/cFs) works well on any available pin (a hardware PWM (pulse width modulated) pin is not required). The library is not ideal, in that servos must be refreshed periodically. A true hardware library would be best if one could be coded, although it would be limited to pins capable of hardware PWM (GPIO #1 and #4).

If you turn the potentiometer, the servo will rotate from zero to 180 degrees. The circuit can be expanded into a number of useful projects.

Parts used:

- **Adafruit Gemma M0 (https://adafru.it/ytb), Trinket M0 (https://adafru.it/zya), Trinket Mini** (https://adafru.it/egk) or **Gemma v2** (http://adafru.it/1222) microcontroller board (if Trinket, either the 3.3V (http://adafru.it/1500) or 5V (http://adafru.it/1501) type works). We recommend using the **Trinket M0 (https://adafru.it/zya).**
- USB cable for power and reprogramming
- Standard 5 volt hobby servo (https://adafru.it/caH)(several are available in the Adafruit shop)
- Potentiometer (http://adafru.it/356) (variable resistor), anything from 1K Ω to 10 KΩ, value not critical
- Breadboard, jumper wires/hookup wire

- Power supply and connector - if you want to have it connected to a battery pack or computer-independant

Wiring

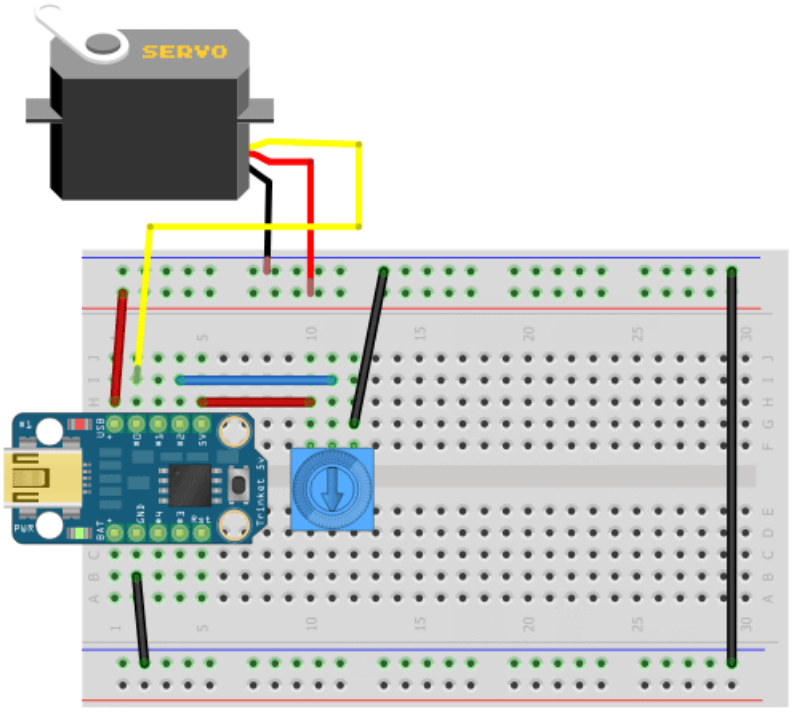
Wiring up the servo and trimpot is easy. You can use EITHER a 5V or 3V Trinket (or Gemma!)

The fastest way to get started is to have the servo powered by **USB**+ which is the 5V line from the USB port. This lets you get up to 500mA without going through the onboard voltage regulator. Then ground to **GND**. The control line of the servo goes to **#0**.

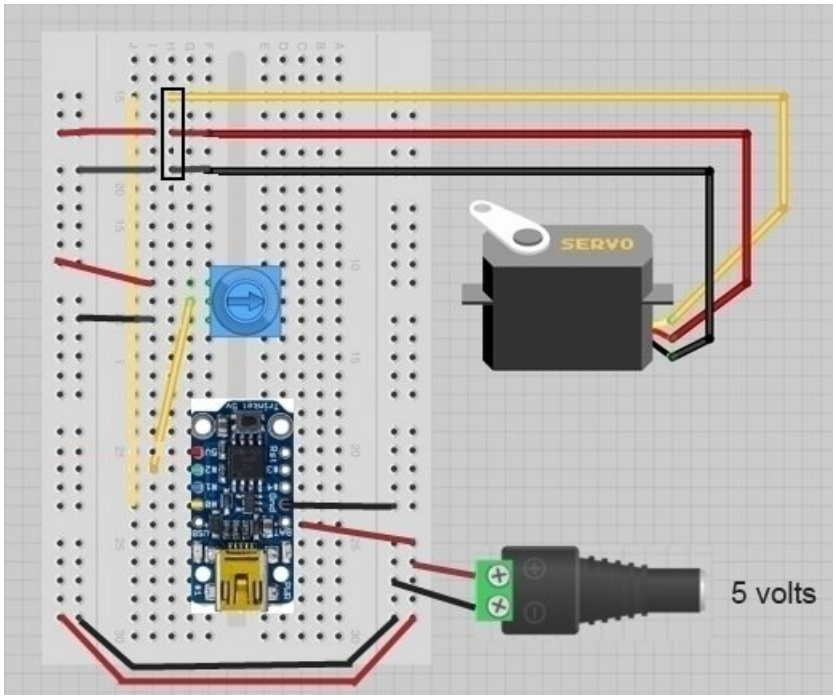
The outer edges of the trimpot go to **GND** and **5V** or **3V** (whichever is available on the Trinket) and the middle pin goes to **#2**

Connect the Trinket to the computer USB for power and programming!

These diagrams use the Trinket Mini but you can also use the Trinket M0 or Gemma M0.



Alternatively, you can power the Trinket from an external 4.5-6VDC battery pack as shown below. Instead of **USB**+ use **BAT**+ for Trinket/servo power.



You may solder the included header pins onto the Trinket to aid in attaching the board to a breadboard. The Trinket connects to power and ground (via the BAT pin) as well as the red and black (or red and brown) on the servo. The outer legs of the potentiometer also connect to power and ground (one to each, it is not polarity sensitive). The center leg of the potentiometer is connected to Trinket GPIO #2. The signal wire (yellow or orange) on the servo connects to Trinket Pin #0.

Normal or extra-long header pins can be placed on the breadboard to help you connect the servo to the board.

It is suggested you use an external wall or battery supply and not power the servo via the regulator. Servos can draw up to 500mA and the Trinket regulator can only source 150 milliamps (USB power generally 500 milliamps). The female DC adapter is helpful in connecting power supplies with barrel connectors to breadboards.

Arduino Code

Trinket M0 & Gemma M0

The Trinket / Gemma M0 can use the default Servo library. The Arduino Code for 'Knob (https://adafru.it/CcP)' is a good place to start.

The following pins on each controller can be used for driving servos with the default Servo Library:

- Trinket M0 - Use PWM pins D0, D2, D3, D4 (pin D1 cannot be used)
- Gemma M0 - Use PWM pins D0, D2 (pin D1 cannot be used)

Trinket & Gemma with ATtiny85

To control servos with the tiny microcontroller on the Trinket, we'll need a Servo library. The default Arduino Servo library is really only good for Uno/Leonardo/Due and similar beefy processors that can drive servos 'standalone'. Sadly, the Attiny85 can't quite do that as it does not have 16bit timers.

So instead we'll use a simpler servo library. Luckily, we wrote one that's perfect! Download the Adafruit_SoftServo library from https://github.com/adafruit/Adafruit_SoftServo (https://adafru.it/cFs) by clicking the button below

https://adafru.it/cFt

https://adafru.it/cFt

Install the library into the Arduino libraries directory. See All About Installing Arduino Libraries (https://adafru.it/aYM) for a guide.

Please ensure your Arduino IDE is augmented to support Trinket per the Introducing Trinket Guide (https://adafru.it/cEu).

The code below may be copied-and-pasted into a new project window in the Arduino IDE.

The Arduino code presented below works well on the Trinket Mini and Gemma v2. If you have an M0 board you can use the CircuitPython code on the next page of this guide.

```
/*****************************************************************
  SoftServo sketch for Adafruit Trinket.  Turn the potentiometer knob
  to set the corresponding position on the servo
  (0 = zero degrees, full = 180 degrees)

  Required library is the Adafruit_SoftServo library
  available at https://github.com/adafruit/Adafruit_SoftServo
  The standard Arduino IDE servo library will not work with 8 bit
  AVR microcontrollers like Trinket and Gemma due to differences
  in available timer hardware and programming. We simply refresh
  by piggy-backing on the timer0 millis() counter

  Required hardware includes an Adafruit Trinket microcontroller
  a servo motor, and a potentiometer (nominally 1Kohm to 100Kohm

  As written, this is specifically for the Trinket although it should
```

```
  As written, this is specifically for the Trinket although it should
  be Gemma or other boards (Arduino Uno, etc.) with proper pin mappings

  Trinket:        USB+   Gnd   Pin #0  Pin #2 A1
  Connection:     Servo+  -    Servo1   Potentiometer wiper

 ************************************************************************/

#include <Adafruit_SoftServo.h>  // SoftwareServo (works on non PWM pins)

#define SERVO1PIN 0   // Servo control line (orange) on Trinket Pin #0

#define POTPIN   1   // Potentiometer sweep (center) on Trinket Pin #2 (Analog 1)

Adafruit_SoftServo myServo1, myServo2;  //create TWO servo objects

void setup() {
  // Set up the interrupt that will refresh the servo for us automagically
  OCR0A = 0xAF;            // any number is OK
  TIMSK |= _BV(OCIE0A);    // Turn on the compare interrupt (below!)

  myServo1.attach(SERVO1PIN);   // Attach the servo to pin 0 on Trinket
  myServo1.write(90);           // Tell servo to go to position per quirk
  delay(15);                    // Wait 15ms for the servo to reach the position
}

void loop()  {
  int potValue;  // variable to read potentiometer
  int servoPos;  // variable to convert voltage on pot to servo position
  potValue=analogRead(POTPIN);               // Read voltage on potentiometer
  servoPos = map(potValue, 0, 1023, 0, 179);  // scale it to use it with the servo (value between 0 and 1
  myServo1.write(servoPos);                   // tell servo to go to position

  delay(15);                             // waits 15ms for the servo to reach the position
}

// We'll take advantage of the built in millis() timer that goes off
// to keep track of time, and refresh the servo every 20 milliseconds
// The SIGNAL(TIMER0_COMPA_vect) function is the interrupt that will be
// Called by the microcontroller every 2 milliseconds
volatile uint8_t counter = 0;
SIGNAL(TIMER0_COMPA_vect) {
  // this gets called every 2 milliseconds
  counter += 2;
  // every 20 milliseconds, refresh the servos!
  if (counter >= 20) {
    counter = 0;
    myServo1.refresh();
  }
}
```

From the **Tools→Board** menu, select **Adafruit Trinket 8 MHz** (or **Gemma**). Connect the USB cable between the computer and Trinket, press the reset button on the board, then quickly click the upload button (right arrow icon) in the Arduino IDE.

If you get an error message (or a huge list of them), it's usually one of the following:

- Is the Arduino IDE properly configured for Trinket use? Try compiling and uploading a simple sketch (like the Blink example, set for pin #1).
- Is the Adafruit_SoftServo library properly installed? It must be correctly named and in the right location (the Arduino libraries folder - see All About Installing Arduino Libraries (https://adafru.it/aYM) for a guide).
- If the code compiles but does not upload, press the reset button and try the upload again.

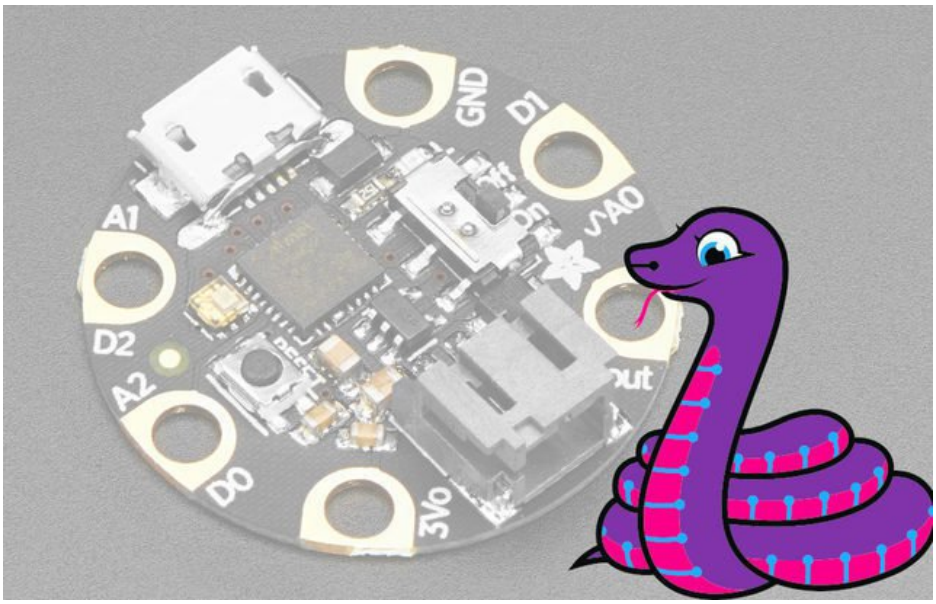
The code compiles to 1678 bytes of 5310 maximum.

Now you can try twisting the potentiometer to watch the servo spin!

> You can connect servos to pins #0, #1 and #2 but if you connect to #3 or #4 it will interfere with the USB bootloader. So if you want to use #3 or #4, unplug the servos while uploading!

Check this video for what you will see!

CircuitPython Code



Trinket M0 boards can run **CircuitPython** — a different approach to programming compared to Arduino sketches. In fact, **CircuitPython comes factory pre-loaded on Trinket M0**. If you've overwritten it with an Arduino sketch, or just want to learn the basics of setting up and using CircuitPython, this is explained in the **Adafruit** (https://adafru.it/zxF)**Trinket M0 guide (https://adafru.it/zxF).**

> These directions are specific to the "M0" boards. The original Trinket and Gemma with an 8-bit AVR microcontroller doesn't run CircuitPython...for those boards, use the Arduino sketch on the "Arduino code" page of this guide.

Below is CircuitPython code that works similarly (though not exactly the same) as the Arduino sketch shown on a prior page. To use this, plug the Trinket M0 into USB...it should show up on your computer as a small **flash drive**...then edit the file "**main.py**" with your text editor of choice. Select and copy the code below and paste it into that file, **entirely replacing its contents** (don't mix it in with lingering bits of old code). When you save the file, the code should **start running almost immediately** (if not, see notes at the bottom of this page).

If Trinket M0 doesn't show up as a drive, follow the Trinket M0 guide link above to prepare the board for CircuitPython.

```
# Trinket Gemma Servo Control
# for Adafruit M0 boards

import board
import pulseio
from adafruit_motor import servo
from analogio import AnalogIn

# servo pin for the M0 boards:
pwm = pulseio.PWMOut(board.A2, duty_cycle=2 ** 15, frequency=50)
my_servo = servo.Servo(pwm)
angle = 0

# potentiometer
trimpot = AnalogIn(board.A1)  # pot pin for servo control

def remap_range(value, left_min, left_max, right_min, right_max):
    # this remaps a value from original (left) range to new (right) range
    # Figure out how 'wide' each range is
    left_span = left_max - left_min
    right_span = right_max - right_min

    # Convert the left range into a 0-1 range (int)
    value_scaled = int(value - left_min) / int(left_span)

    # Convert the 0-1 range into a value in the right range.
    return int(right_min + (value_scaled * right_span))


while True:
    angle = remap_range(trimpot.value, 0, 65535, 0, 180)
    my_servo.angle = angle
```

This code requires an additional library be installed:

1. adafruit_motor

If you've just reloaded the board with CircuitPython, create the "lib" directory and then download the Adafruit CircuitPython Bundle (https://adafru.it/uap). You can copy 'adafruit_motor' folder into the lib directory.

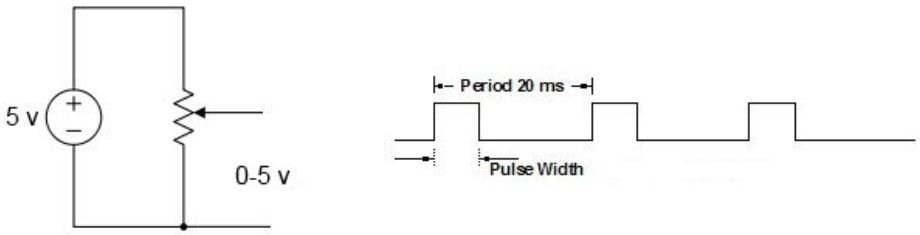
```
$ mkdir /Volumes/CIRCUITPY/lib
$ cp -pr adafruit_motor /Volumes/CIRCUITPY/lib
```

https://adafru.it/uap

https://adafru.it/uap

Review and Going Further

How It Works



The potentiometer creates a voltage divider, providing a voltage from zero to five volts depending on how you turn the shaft. The voltage is read by the analog input on the Trinket. The Trinket calculates an angle from zero to 180 degrees in proportion to the voltage on the potentiometer. The Adafruit_SoftServo library sends a pulse width modulated signal to the Trinket Pin 0, which is interpreted by the servo as a specific angle to move to depending on the pulse width.

The Adafruit_SoftServo Library

You may define multiple Adafruit_SoftServo objects and control them on different pins. In theory, all five pins should be capable or servo use, **but if using #3 and #4 remove the servo(s) while USB uploading!**

The trickiest part of using this library is the constant software refresh to keep signals going to the servos. We take advantage of the Arduino IDE's built in timer (commonly known as **millis()**) and piggyback on top of it to create the 50-times-a-second update a servo requires.

As previously stated, it would be preferable to have an AVR 8 bit hardware timer based library. One on the web, **Servo8Bit**, is billed as ATtiny85 compatible.

Going Further

Two servos can create an x-y axis controller or a two wheeled robot. This would still leave 3 pins for other functions.