Universitat de Girona
Escola Politècnica Superior

Treball final de grau

Estudi: Grau en Enginyeria Mecànica

Títol: Disseny, fabricació i muntatge d'un braç robòtic de 4 graus de llibertat

Document: Plec de condicions

Alumne: Joan Luque Barrull

Tutor: Lluis Ripoll Masferrer

Departament: Enginyeria mecànica i de la construcció industrial

Àrea: Enginyeria mecànica

Convocatòria (mes/any): Juny 2022

Índex

1. INTRODUCCIÓ

1.1 Objecte i abast

En aquest document es recullen cadascuna de les especificacions dels materials i dels equips de fabricació a emprar, així com el procés a seguir en el seu muntatge en cas que es volgués repetir aquest projecte.

Així doncs, aquest document assenyala les obligacions i responsabilitats existents a l'hora de fabricar i materialitzar el projecte per tal que satisfaci els requeriments projectats.

D'altra banda, amb aquest document el projectista queda exempt de culpa en cas de trencament, fallida o error derivats d'un incorrecte seguiment de les instruccions descrites a continuació.

1.2 Documents contractuals i informatius

El projecte està format per diferents blocs, separats físicament per documents, com són la memòria i annexos, els plànols, el plec de condicions, l'estat d'amidaments, i per últim el pressupost. Entre tots aquests documents es defineix el projecte en la seva totalitat.

- Memòria: Es detalla la solució proposada.
- Plànols: S'hi defineix el sistema projectat, així com els detalls constructius.
- Plec de condicions: S'hi troben les prescripcions tècniques.
- Estat d'amidaments: Hi apareixen les medicions de tots els components que intervenen dins el projecte.
- Pressupost: S'hi detalla el cost del sistema, així com les partides que en formen part.

D'aquests documents que composen el projecte complet, en són documents contractuals únicament el conjunt de plànols, el plec de condicions i el pressupost. La memòria i l'estat d'amidaments no deixen de ser documents de caràcter informatiu. Qualsevol modificació que impliqui un canvi important en el projecte comportarà la notificació al projectista per tal de ser aprovada, i en cas que ho sigui caldrà que es redacti el projecte de nou amb els corresponents canvis.

1.3 Compatibilitat entre documents

Els documents mencionats anteriorment tenen un ordre d'importància, per tal de seguir en cas de que apareguin discordances entre ells o confusions per part del lector. Aquest ordre és el següent:

- Plànols
- Plec de condicions
- Memòria i annexos
- Estat d'amidaments
- Pressupost

El conjunt de Plànols i el Plec de Condicions són els documents vinculants. En cas de possibles discrepàncies aquests prevaldran sobre la resta de documents del projecte.
Si entre aquests dos documents també apareixen variacions, pren sempre més importància el document dels Plànols, on s'hi defineixen les especificacions tècniques.

2. PRESCRIPCIONS TÈCNIQUES

2.1 Condicions dels materials

El braç robòtic ha estat dissenyat i comprovat per a ser fabricat amb uns materials de característiques concrets per a cada peça. Aquests materials estan especificats tant en el document *Plànols*, com en el document *Estat d'amidaments*, i serà obligatòria la seva utilització per obtenir un disseny vàlid.

En cas que algun material o element dels indicats no es trobi en estoc i es necessités amb urgència, la decisió de canvi per un que presenti les mateixes característiques o semblants que els originals requereix l'aprovació del projectista.

2.2 Condicions de fabricació

Per tal d'implementar el procés de fabricació adient, cal seguir els passos indicats a *l'Annex C. Fabricació*. És important que els elements impresos en els quals van acoblats elements comercials com guies o rodaments quedin ben impresos i amb una tolerància adequada per tal d'assegurar un bon funcionament del mecanisme i evitar problemes en la seva utilització i possibles jocs entre peces.

El disseny del braç incorpora una sèrie de peces que requereixen unes característiques específiques, tan dimensionals com superficials. Aquestes es veuen representades en els plànols de cadascuna de les peces i s'han de seguir de manera estricta durant la seva fabricació. Així mateix i una vegada acabada la fabricació s'ha de realitzar un control dimensional i visual per assegurar que les peces compleixen amb les característiques especificades.

Per altra banda, és obligatori utilitzar eines adequades i calibrades correctament per a la fabricació de totes aquestes peces.

2.3 Condicions de muntatge

Durant el procés de muntatge és imprescindible seguir l'ordre indicat en els plànols per tal de garantir el correcte funcionament del prototip i evitar futurs problemes.

En aquest procés s'ha de fer especial èmfasi en la correcta col·locació i la fixació de tots els elements, així com la correcta instal·lació dels rodaments i guies.

Per a la instal·lació electrònica, cal seguir el diagrama i la programació de *l'Annex C. Electrònica i control.*

Finalment i una vegada acabat el muntatge es realitza un control funcional i visual per assegurar que el conjunt compleix amb les característiques especificades.

2.4 Condicions per l'usuari

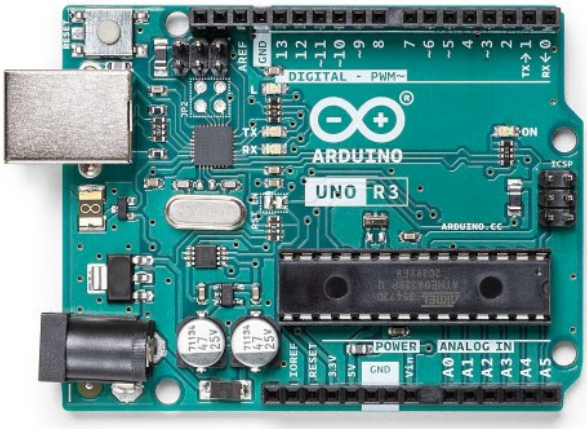
En el moment en el que l'usuari es disposi a utilitzar el braç robòtic, prèviament està obligat a llegir i entendre el contingut del apartat *Annex E: Manual d'usuari i manteniment* del *document 1. Memòria i annexos.*

Així mateix, l'usuari haurà de procurar no utilitzar elements que puguin malmetre el prototip i fer-ne un ús sempre sobre una superfície anivellada.

FITXES TÈCNIQUES

Controladora: Arduino UNO

Product Reference Manual
SKU: A000066



Description

The Arduino UNO R3 is the perfect board to get familiar with electronics and coding. This versatile microcontroller is equipped with the well-known ATmega328P and the ATMega 16U2 Processor.
This board will give you a great first experience within the world of Arduino.

Target areas:

Maker, introduction, industries

Features

- **ATMega328P** Processor
  - **Memory**
    - AVR CPU at up to 16 MHz
    - 32KB Flash
    - 2KB SRAM
    - 1KB EEPROM
  - **Security**
    - Power On Reset (POR)
    - Brown Out Detection (BOD)
  - **Peripherals**
    - 2x 8-bit Timer/Counter with a dedicated period register and compare channels
    - 1x 16-bit Timer/Counter with a dedicated period register, input capture and compare channels
    - 1x USART with fractional baud rate generator and start-of-frame detection
    - 1x controller/peripheral Serial Peripheral Interface (SPI)
    - 1x Dual mode controller/peripheral I2C
    - 1x Analog Comparator (AC) with a scalable reference input
    - Watchdog Timer with separate on-chip oscillator
    - Six PWM channels
    - Interrupt and wake-up on pin change
  - **ATMega16U2 Processor**
    - 8-bit AVR® RISC-based microcontroller
  - **Memory**
    - 16 KB ISP Flash
    - 512B EEPROM
    - 512B SRAM
    - debugWIRE interface for on-chip debugging and programming
  - **Power**
    - 2.7-5.5 volts

CONTENTS

1 The Board

1.1 Application Examples

The UNO board is the flagship product of Arduino. Regardless if you are new to the world of electronics or will use the UNO as a tool for education purposes or industry-related tasks.

First entry to electronics: If this is your first project within coding and electronics, get started with our most used and documented board; Arduino UNO. It is equipped with the well-known ATmega328P processor, 14 digital input/output pins, 6 analog inputs, USB connections, ICSP header and reset button. This board includes everything you will need for a great first experience with Arduino.

Industry-standard development board: Using the Arduino UNO board in industries, there are a range of companies using the UNO board as the brain for their PLC's.

Education purposes: Although the UNO board has been with us for about ten years, it is still widely used for various education purposes and scientific projects. The board's high standard and top quality performance makes it a great resource to capture real time from sensors and to trigger complex laboratory equipment to mention a few examples.

1.2 Related Products

- Starter Kit
- Tinkerkit Braccio Robot
- Example

2 Ratings

2.1 Recommended Operating Conditions

Symbol	Description	Min	Max
	Conservative thermal limits for the whole board:	-40 °C (-40°F)	85 °C (185°F)

NOTE: In extreme temperatures, EEPROM, voltage regulator, and the crystal oscillator, might not work as expected due to the extreme temperature conditions

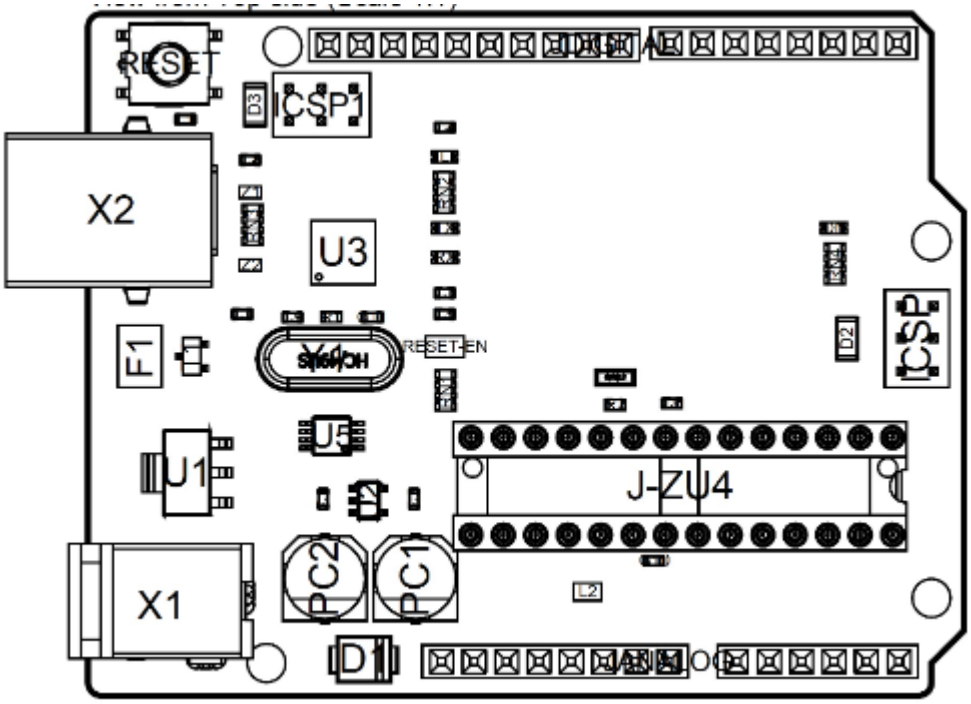
2.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
VINMax	Maximum input voltage from VIN pad	6	-	20	V
VUSBMax	Maximum input voltage from USB connector		-	5.5	V
PMax	Maximum Power Consumption	-	-	xx	mA

3 Functional Overview

3.1 Board Topology

Top view



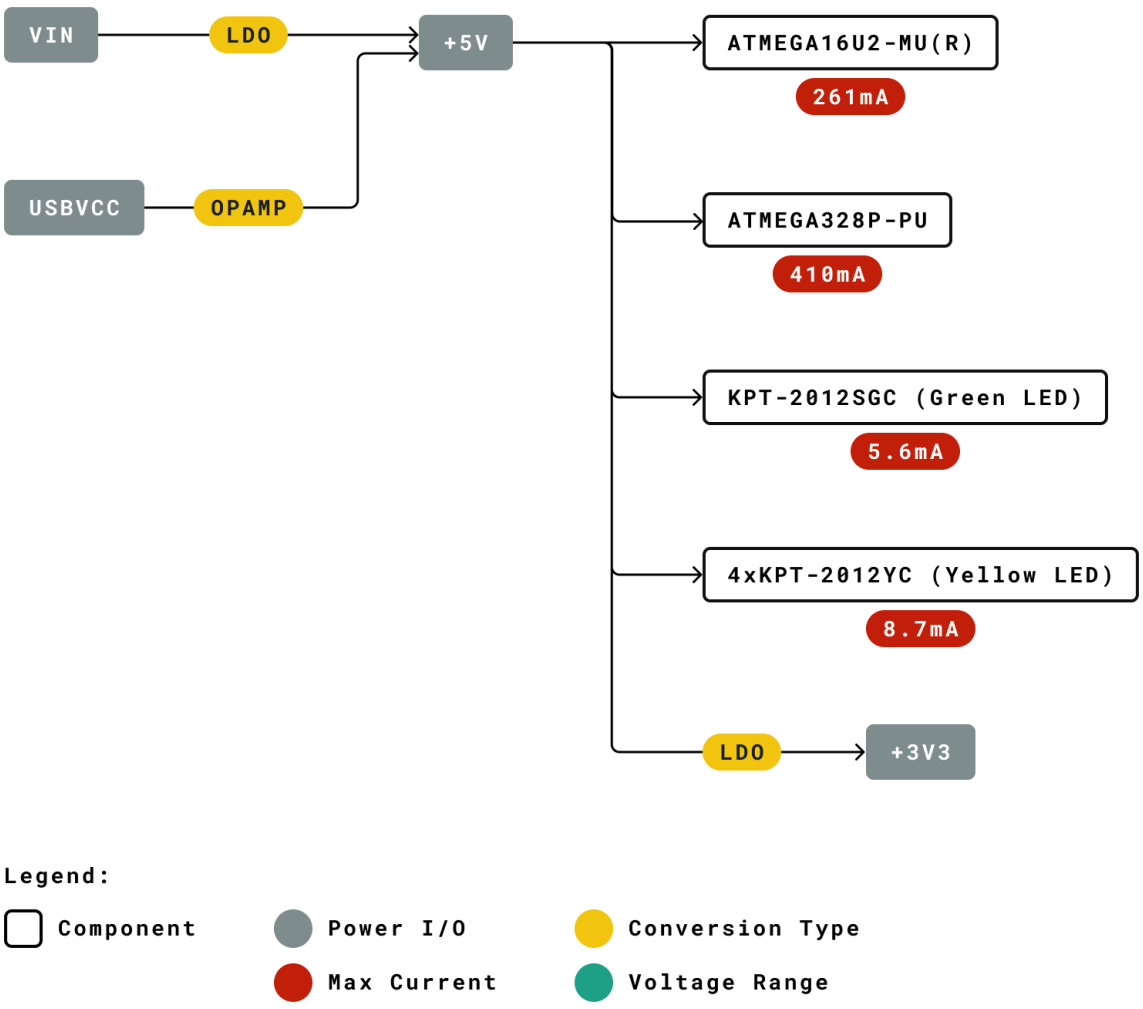
Board topology

Ref.	Description	Ref.	Description
X1	Power jack 2.1x5.5mm	U1	SPX1117M3-L-5 Regulator
X2	USB B Connector	U3	ATMEGA16U2 Module
PC1	EEE-1EA470WP 25V SMD Capacitor	U5	LMV358LIST-A.9 IC
PC2	EEE-1EA470WP 25V SMD Capacitor	F1	Chip Capacitor, High Density
D1	CGRA4007-G Rectifier	ICSP	Pin header connector (through hole 6)
J-ZU4	ATMEGA328P Module	ICSP1	Pin header connector (through hole 6)
Y1	ECS-160-20-4X-DU Oscillator		

3.2 Processor

The Main Processor is a ATmega328P running at up tp 20 MHz. Most of its pins are connected to the external headers, however some are reserved for internal communication with the USB Bridge coprocessor.

3.3 Power Tree



Power tree

4 Board Operation

4.1 Getting Started - IDE

If you want to program your Arduino UNO while offline you need to install the Arduino Desktop IDE [1] To connect the Arduino UNO to your computer, you'll need a Micro-B USB cable. This also provides power to the board, as indicated by the LED.

4.2 Getting Started - Arduino Web Editor

All Arduino boards, including this one, work out-of-the-box on the Arduino Web Editor [2], by just installing a simple plugin.

The Arduino Web Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

4.3 Getting Started - Arduino IoT Cloud

All Arduino IoT enabled products are supported on Arduino IoT Cloud which allows you to Log, graph and analyze sensor data, trigger events, and automate your home or business.

4.4 Sample Sketches

Sample sketches for the Arduino XXX can be found either in the "Examples" menu in the Arduino IDE or in the "Documentation" section of the Arduino Pro website [4]

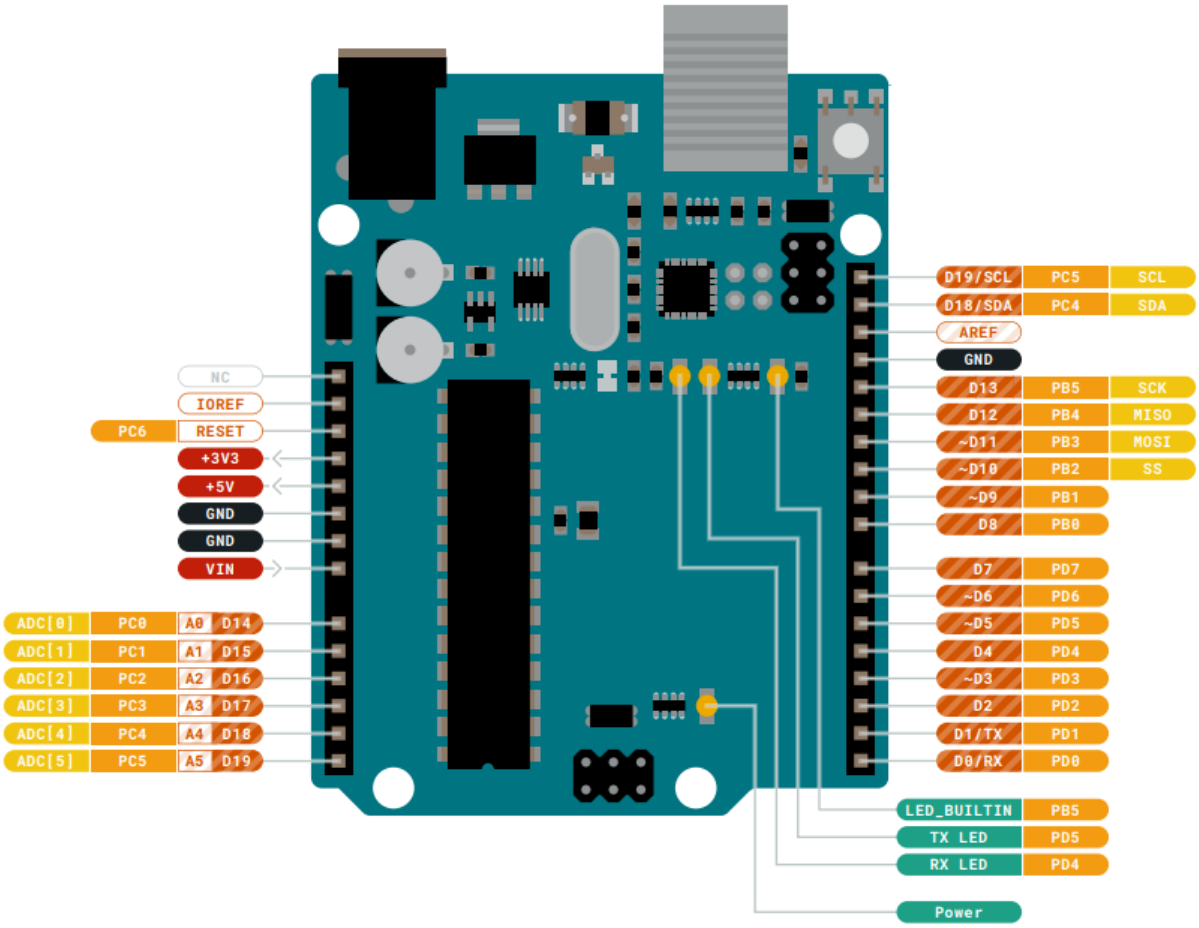
4.5 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on ProjectHub **[5]**, the Arduino Library Reference **[6]** and the online store **[7]** where you will be able to complement your board with sensors, actuators and more

4.6 Board Recovery

All Arduino boards have a built-in bootloader which allows flashing the board via USB. In case a sketch locks up the processor and the board is not reachable anymore via USB it is possible to enter bootloader mode by double-tapping the reset button right after power up.

5 Connector Pinouts



Pinout

5.1 JANALOG

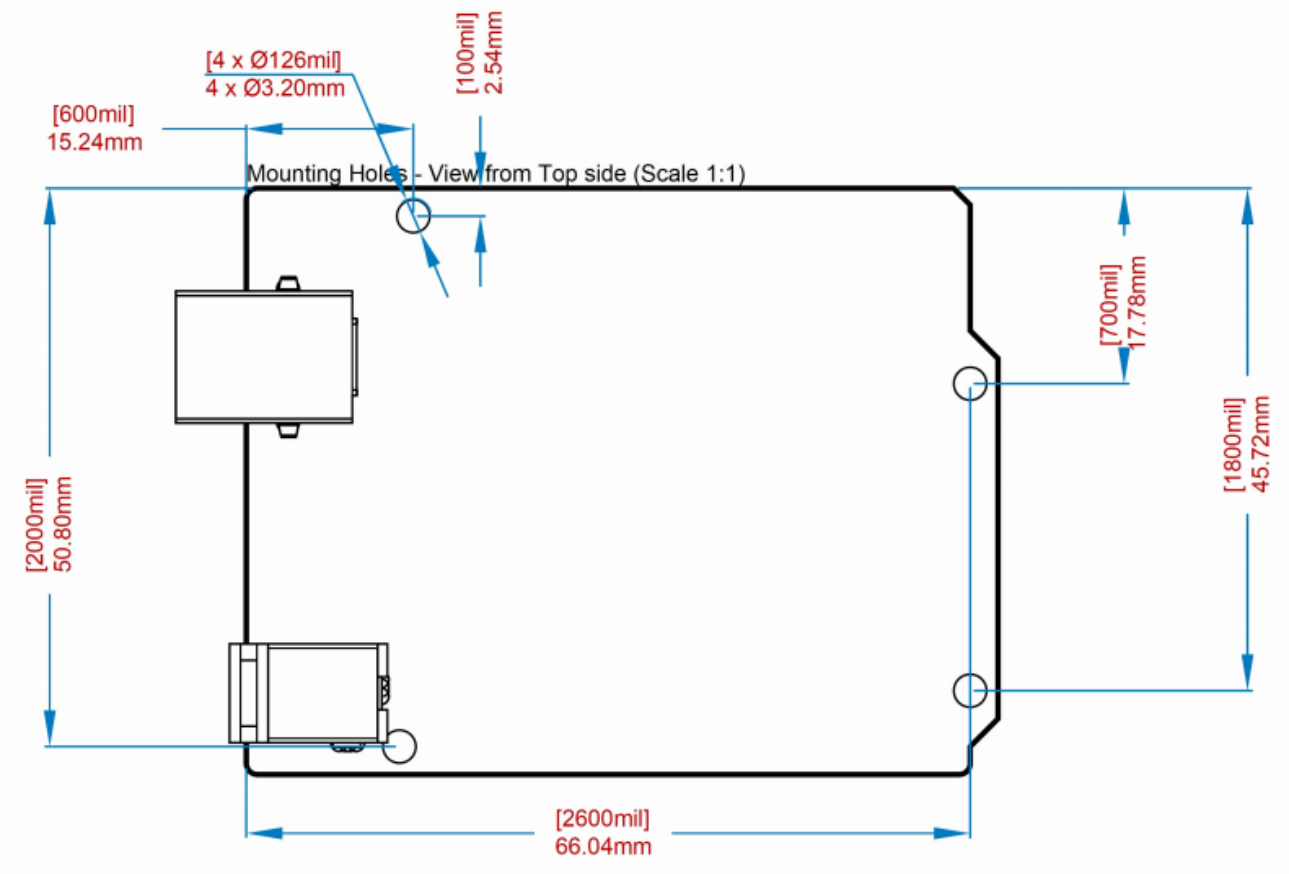
Pin	Function	Type	Description
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

5.2 JDIGITAL

Pin	Function	Type	Description
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SD4	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

5.3 Mechanical Information

5.4 Board Outline & Mounting Holes



Board outline

6 Certifications

6.1 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

ROHS 2 Directive 2011/65/EU	
Conforms to:	EN50581:2012
Directive 2014/35/EU. (LVD)	
Conforms to:	EN 60950-1:2006/A11:2009/A1:2010/A12:2011/AC:2011
Directive 2004/40/EC & 2008/46/EC & 2013/35/EU, EMF	
Conforms to:	EN 62311:2008

6.2 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	Maximum limit (ppm)
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions: No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.

6.3 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

7 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.

Important: The operating temperature of the EUT can't exceed 85°C and shouldn't be lower than -40°C.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 2014/53/EU. This product is allowed to be used in all EU member states.

8 Company Information

Company name	Arduino S.r.l
Company Address	Via Andrea Appiani 25 20900 MONZA Italy

9 Reference Documentation

Reference	Link
Arduino IDE (Desktop)	https://www.arduino.cc/en/Main/Software
Arduino IDE (Cloud)	https://create.arduino.cc/editor
Cloud IDE Getting Started	https://create.arduino.cc/projecthub/Arduino_Genuino/getting-started-with-arduino-web-editor-4b3e4a
Arduino Pro Website	https://www.arduino.cc/pro
Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/
Online Store	https://store.arduino.cc/

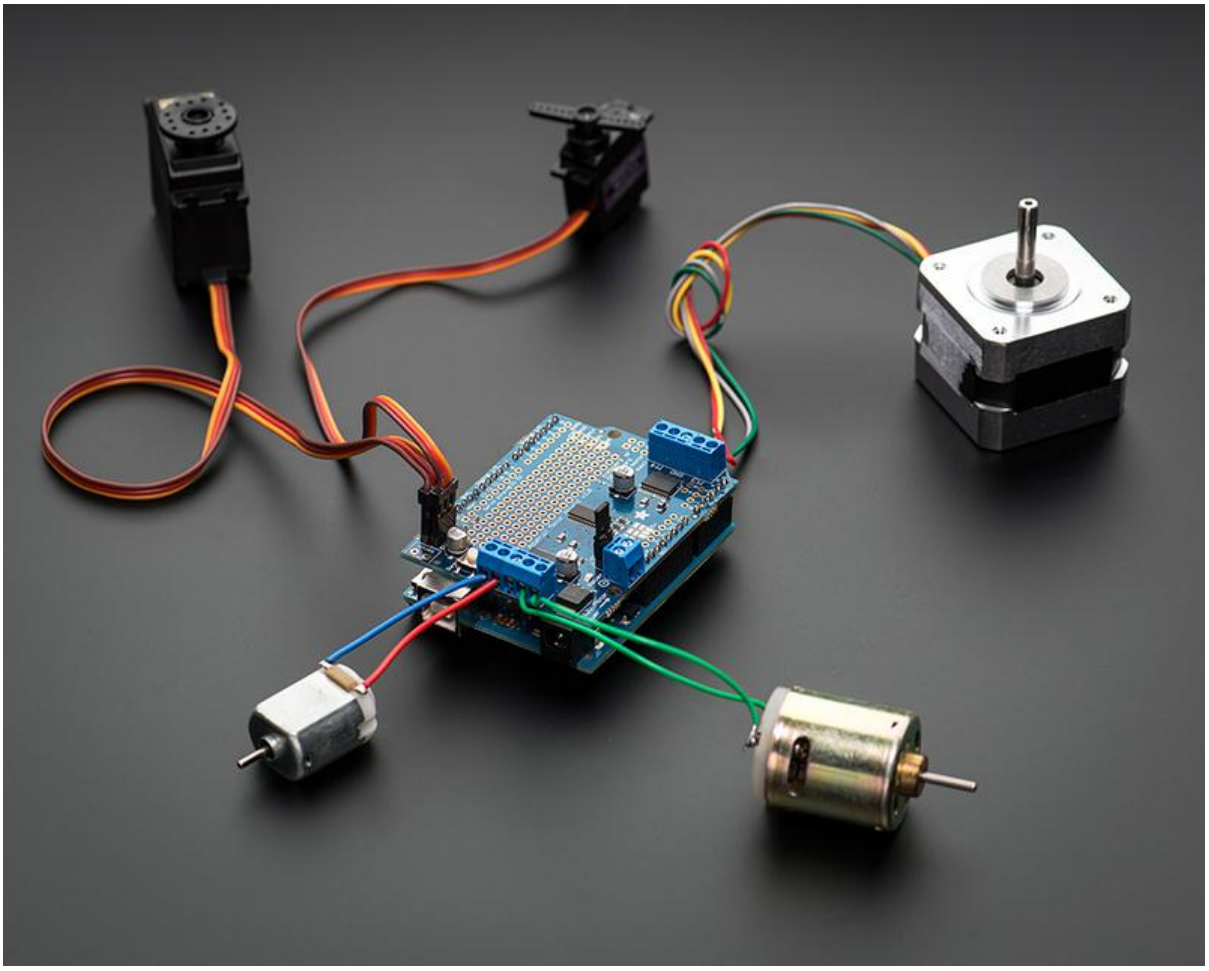
10 Revision History

Date	Revision	Changes
xx/06/2021	1	Datasheet release

Placa: Motorshield v2

Adafruit Motor Shield V2

Created by lady ada

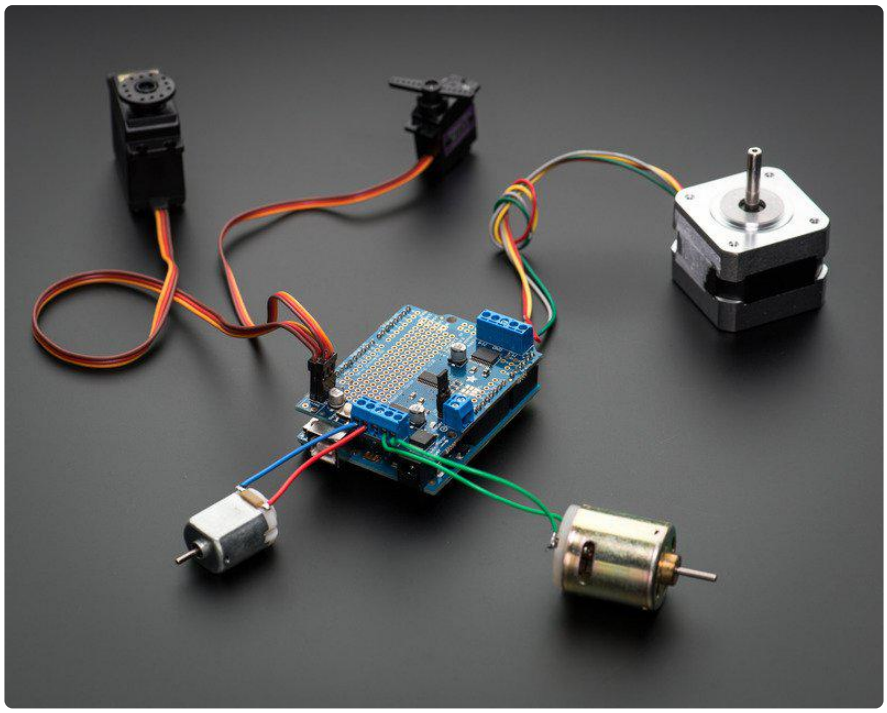


https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino

Last updated on 2021-11-22 09:40:54 AM EST

Table of Contents

Overview



The original Adafruit Motorshield kit is one of our most beloved kits, which is why we decided to make something even better. We have upgraded the shield kit to make the bestest, easiest way to drive DC and Stepper motors. This shield will make quick work of your next robotics project! We kept the ability to drive up to 4 DC motors or 2 stepper motors, but added many improvements:

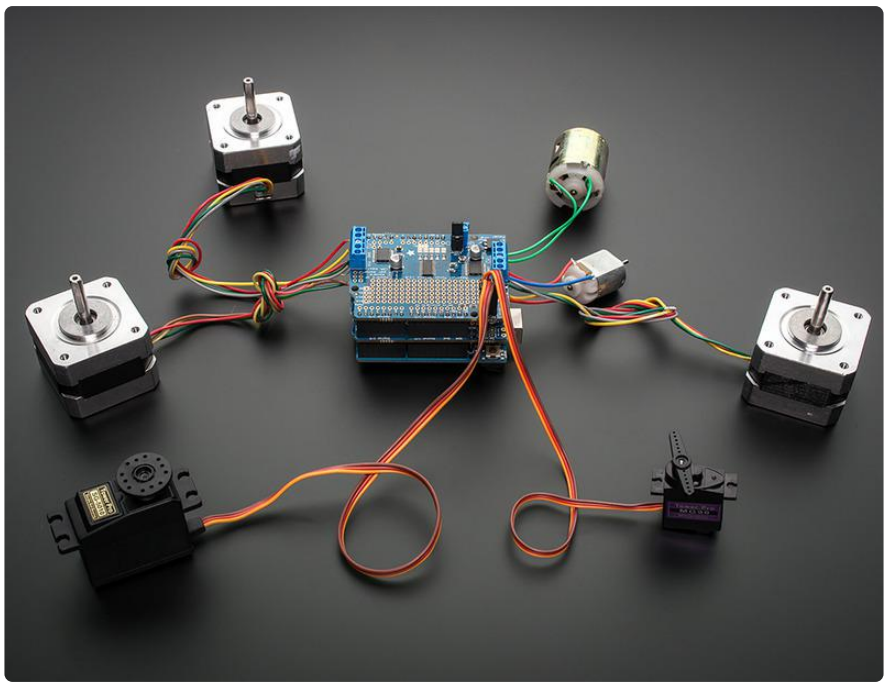
Instead of a L293D darlington driver, we now have the TB6612 MOSFET drivers with 1.2A per channel current capability (you can draw up to 3A peak for approx 20ms at a time). It also has much lower voltage drops across the motor so you get more torque out of your batteries, and there are built-in flyback diodes as well.

Instead of using a latch and the Arduino's PWM pins, we have a fully-dedicated PWM driver chip onboard. This chip handles all the motor and speed controls over I2C. Only two GPIO pins (SDA & SCL) plus 5v and GND. are required to drive the multiple motors, and since it's I2C you can also connect any other I2C devices or shields to the same pins. This also makes it drop-in compatible with any Arduino, such as the Uno, Leonardo, Due and Mega R3.

Completely stackable design: 5 address-select pins means up to 32 stackable shields: that's 64 steppers or 128 DC motors! What on earth could you do with that many steppers? I have no idea but if you come up with something send us a photo because that would be a pretty glorious project.

Lots of other little improvements such as a polarity protection FET on the power pins and a big prototyping area. And the shield is assembled and tested here at Adafruit so all you have to do is solder on straight or stacking headers and the terminal blocks. Lets check out these specs again:

- 2 connections for 5V 'hobby' servos connected to the Arduino's high-resolution dedicated timer - no jitter!
- 4 H-Bridges: TB6612 chipset provides 1.2A per bridge (3A for brief 20ms peaks) with thermal shutdown protection, internal kickback protection diodes. Can run motors on 4.5VDC to 13.5VDC.
- Up to 4 bi-directional DC motors with individual 8-bit speed selection (so, about 0.5% resolution)
- Up to 2 stepper motors (unipolar or bipolar) with single coil, double coil, interleaved or micro-stepping.
- Motors automatically disabled on power-up
- Big terminal block connectors to easily hook up wires (18-26AWG) and power
- Arduino reset button brought up top
- Polarity protected 2-pin terminal block and jumper to connect external power, for separate logic/motor supplies
- Tested compatible with Arduino UNO, Leonardo, ADK/Mega R3, Diecimila & Duemilanove. Works with Due with 3.3v logic jumper. Works with Mega/ADK R2 and earlier with 2 wire jumpers.
- Download the easy-to-use Arduino software library, check out the examples and you're ready to go!
- 5v or 3.3v compatible logic levels - jumper configurable.



As of Arduino 1.5.6-r2 BETA, there is a bug in the Due Wire library that prevents multiple Motor Shields from working properly with the Due!

FAQ

How many motors can I use with this shield?

You can use 2 DC hobby servos that run on 5V and up to 4 DC motors or 2 stepper motors (or 1 stepper and up to 2 DC motors) that run on 5-12VDC

Can I connect more motors?

Yes, by stacking shields! Every shield you stack on will add 4 DC motors or 2 stepper motors (or 1 more stepper and 2 more DC motors).
You will not gain more servo connections as the servo contacts go to pin #9 and #10 on the Arduino.

What if I also need some more servos?

Check out our lovely servo shield, also stackable with this motor shield and adds 16 free-running servos per shield http://learn.adafruit.com/adafruit-16-channel-pwm-slash-servo-shield (https://adafru.it/ciQ)

What Arduinos is this shield compatible with?

It is tested to work with Duemilanove, Diecimila, Uno (all revisions), Leonardo and Mega/ADK R3 and higher.

It can work with Mega R2 and lower if you solder a jumper wire from the shield's SDA pin to Digital 20 and the SCL pin to Digital 21

For use with the Due or other 3.3v processors, you must configure the board for 3.3v logic levels. Find the set of 3 pads labeled "Logic". Cut the small trace between the center pad and 5v and add a jumper from 3.3v to the center.

As of Arduino 1.5.6-r2 BETA, there is a bug in the Due Wire library that prevents multiple Motor Shields from working properly!

I get the following error trying to run the example code: "error: Adafruit_MotorShield.h: No such file or directory...."

Make sure you have installed the Adafruit_MotorShield library

---

How do I install the library?

Check the tutorial page on the subject here http://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino/install-software (https://adafru.it/ciO)

What stepper motors can I used with the shield?

The TB6612B driver chip are simple H-bridge drivers with a 1.2A continuous current limit. There is no active current limiting, so you need to choose a stepper motor that will not try to pull more than that. For a detailed explanation, see this guide (https://adafru.it/r2d)

A simple rule of thumb is to use a motor with a phase resistance of 10 ohms or more. This will be safe to use with supply voltages up to 12v.

The NEMA 17 motor we have in the shop has a phase resistance of about 35 ohms, so it is a good match for the shield.

Can I use this NEMA-17 motor?

NEMA-17 is just a motor frame-size designation. It tells us that the motor body is 1.7" square. It tells us nothing about the electrical characteristics. You will need to know at least the motor's phase resistance in order to determine compatibility.

See this guide for details: Matching the Driver to the Stepper (https://adafru.it/r2d)

HELP! My motor doesnt work! - HELP! My motor doesnt work!...But the servos work FINE!

Is the power LED lit? The Stepper and DC motor connections will not work if the onboard green Power LED is not lit brightly!
You must connect 5-12VDC power to the shield through the POWER terminal blocks or through the DC barrel jack on the Arduino and VIN jumper.

What is the green Power LED for?

The LED indicates the DC/Stepper motor power supply is working. If it is not lit brightly, then the DC/Stepper motors will not run. The servo ports are 5V powered and does not use the DC motor supply

---

What pins are/are not used on the motor shield?

GND and either 5v (default) or 3.3v are required to power the logic on-board. (5v or 3v operation is selectable via jumper)

The shield uses the SDA and SCL i2c pins to control DC and stepper motors. On the Arduino UNO these are also known as A4 and A5. On the Mega these are also known as Digital 20 and 21. On the Leonardo these are also known as digital 2 and 3. Do not use those pins on those Arduinos with this shield with anything other than an i2c sensor/driver.

Since the shield uses I2C to communicate, you can connect any other i2c sensor or driver to the SDA/SCL pins as long as they do not use address 0x60 (the default address of the shield) or 0x70 (the 'all call' address that this chip uses for group-control)

If you want to use the servo connections, they are on pins #9 and #10. If you do not use the connector then those pins are simply not used.

You can use any other pins for any other use

Note that pins A4 and A5 are connected to SDA and SCL for compatibility with classic Arduinos. These pins are not available for use on other processors.

How can I connect to the unused pins?

All pins are broken out into 0.1" spaced header along the edges of the shield

---

My Arduino freaks out when the motors are running! Is the shield broken?

Motors take a lot of power, and can cause 'brownouts' that reset the Arduino. For that reason the shield is designed for seperate (split) supplies - one for the electronics and one for the motor. Doing this will prevent brownouts. Please read the user manual for information about appropriate power supplies.

I'm trying to build this robot and it doesn't seem to run on a 9V battery....

You cannot power motors from a 9V battery. You must use AA batteries or a lead acid battery for motors.

Can this shield control small 3V motors?

Not really, its meant for larger, 5V+ motors. It does not work for 3V motors unless you overdrive them at 5V and then they will burn out faster

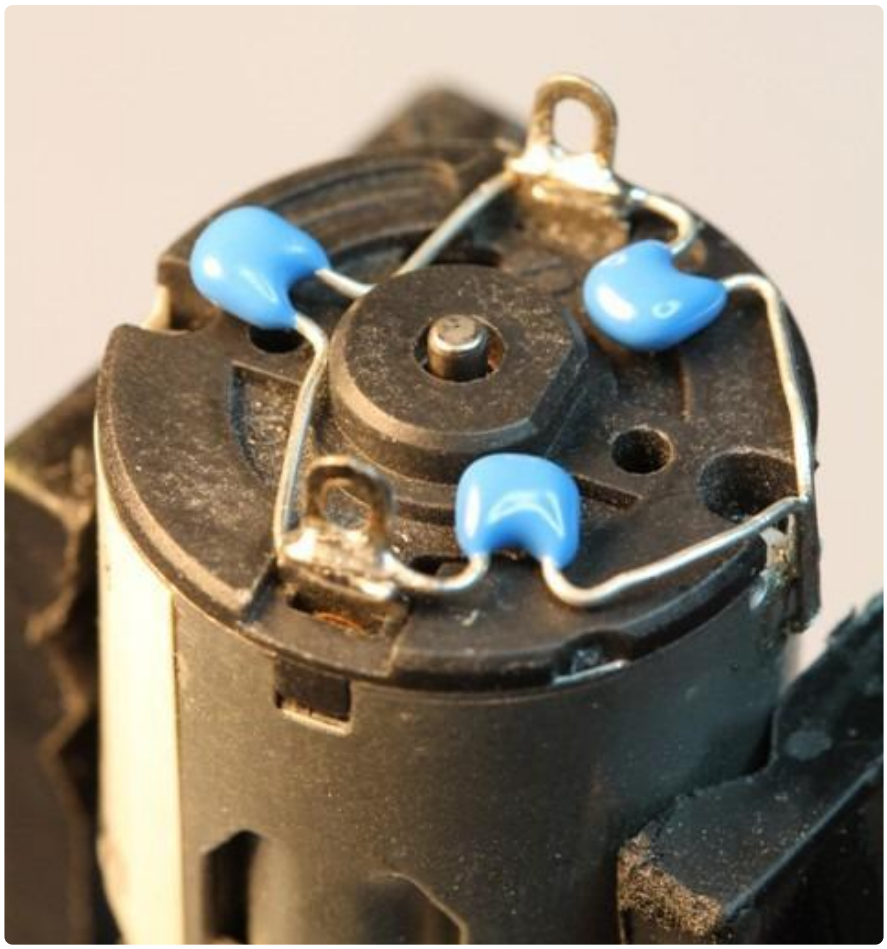
I have good solid power supplies, but the DC motors seem to 'cut out' or 'skip'.

Try soldering a ceramic or disc 0.1uF capacitor between the motor tabs (on the motor itself!) this will reduce noise that could be feeding back into the circuit (thanks macegr (https://adafru.it/clc)!)

When the motors start running nothing else works.

Many small DC motor have a lot of "brush noise". This feeds back into the Arduino circuitry and causes unstable operation. This problem can be solved by soldering some 0.1uF ceramic noise suppression capacitors to the motor.

You will need 3 total. 1 between the motor terminals, and one from each terminal to the motor casing.



But my motor already has a capacitor on it and it still doesn't work.

These motors generate a lot of brush noise and usually need the full 3-capacitor treatment for adequate suppression.

Why don't you just design capacitors into the shield?

They would not be effective there. The noise must be suppressed at the source or the motor leads will act like antennae and broadcast it to the rest of the system!

Why won't my stepper motor go any faster?

Since the shield is controlled by i2c, the maximum step rate is limited by the i2c bus speed. The default bus speed is 100KHz and can be increased to 400KHz by editing the library file in your Arduino installation folder. The file can be found in hardware/libraries/wire/utility/twi.h.

Find the line with: "#define TWI_FREQ 100000L"
and change it to "#define TWI_FREQ 400000L"

Or, you can add the following code to your setup() function: (Note: this line must be inserted after the call to begin())

```
TWBR = ((F_CPU /400000l) - 16) / 2; // Change the i2c clock to 400KHz
```

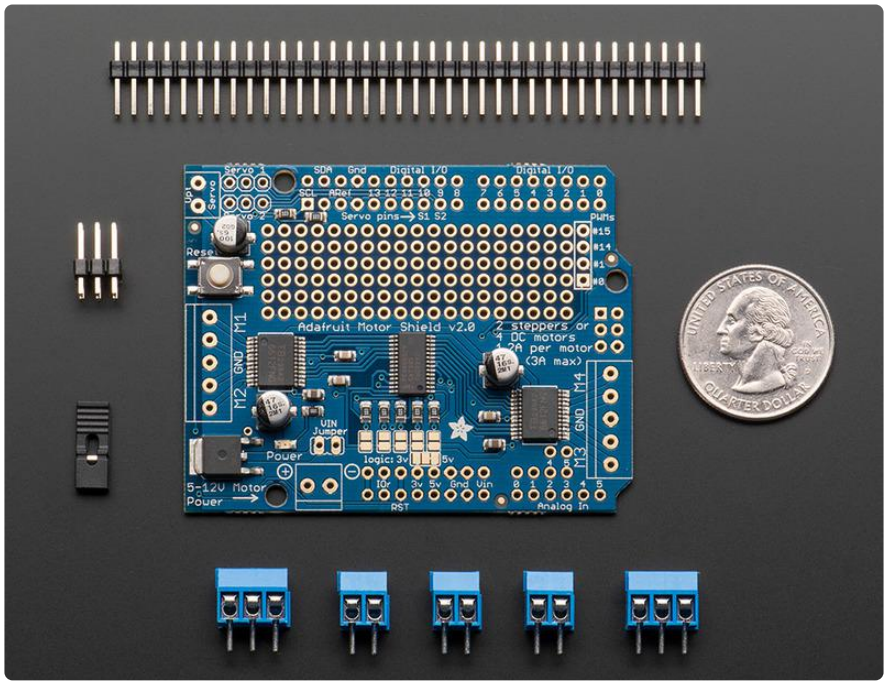
What I2C addresses are used by this shield?

The shield is addressable from 0x60-0x7F. 0x70 is an "all call" address that all boards will answer to.

My shield doesn't work with my LED backpack.

Some backpacks have a default address of 0x70. This is the "all call" address of the controller chip on the motor shield. If you re-address your backpack, it will work with the shield.

Install Headers & Terminals



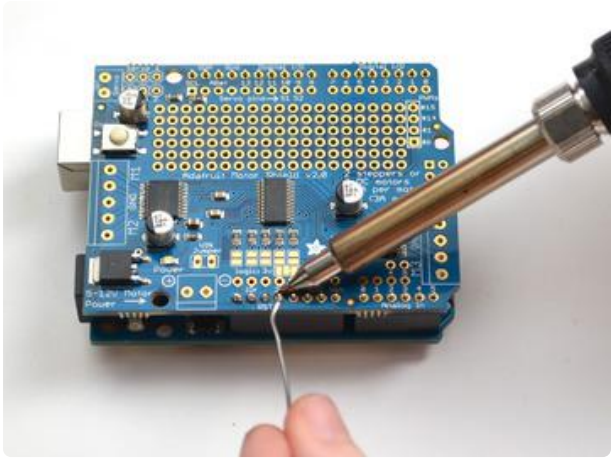
Installing Standard Headers

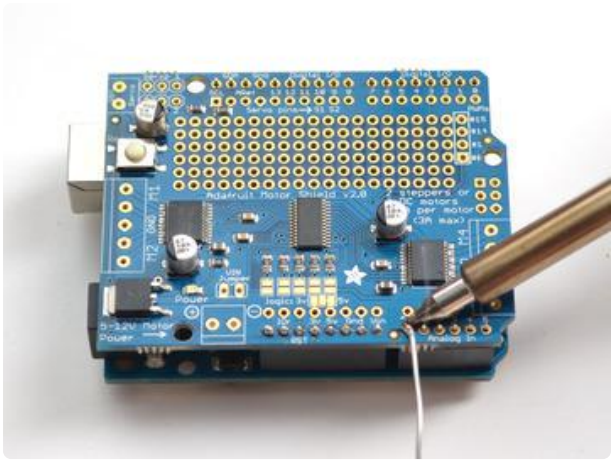
The shield comes with 0.1" standard header. Standard header does not permit stacking but it is mechanically stronger and they're much less expensive too! If you want to stack a shield on top, do not perform this step as it is not possible to uninstall

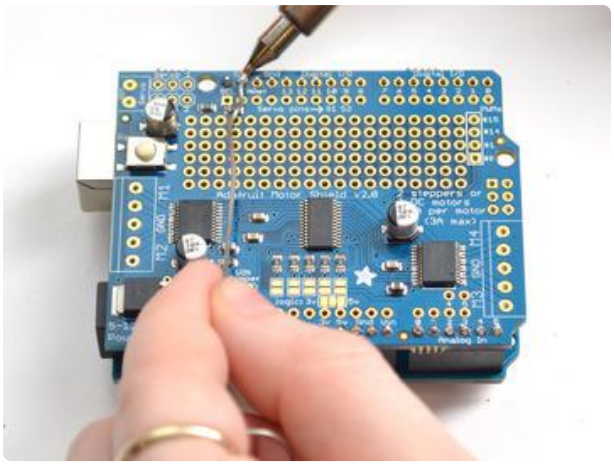
the headers once soldered in! Skip down to the bottom for the stacking tutorial

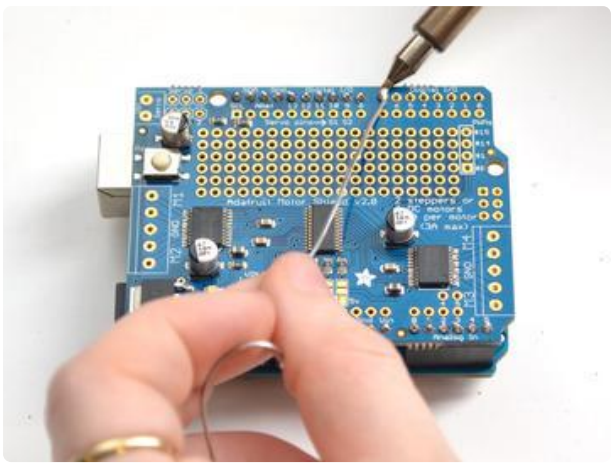
Break apart the 0.1" header into 6, 8 and/or 10-pin long pieces and slip the long ends into the headers of your Arduino

Place the assembled shield on top of the header-ed Arduino so that all of the short parts of the header are sticking through the outer set of pads



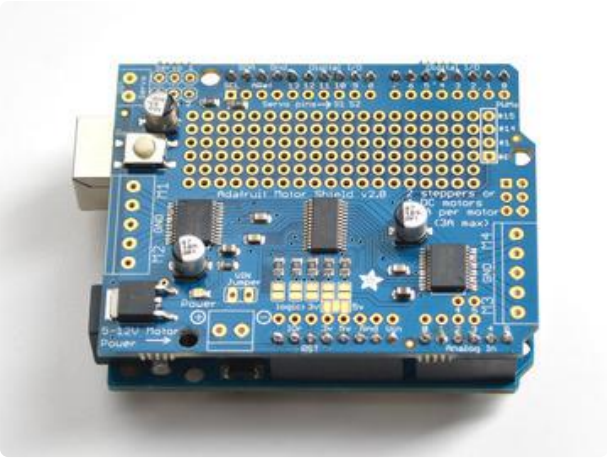






Solder each one of the pins into the shield to make a secure connection

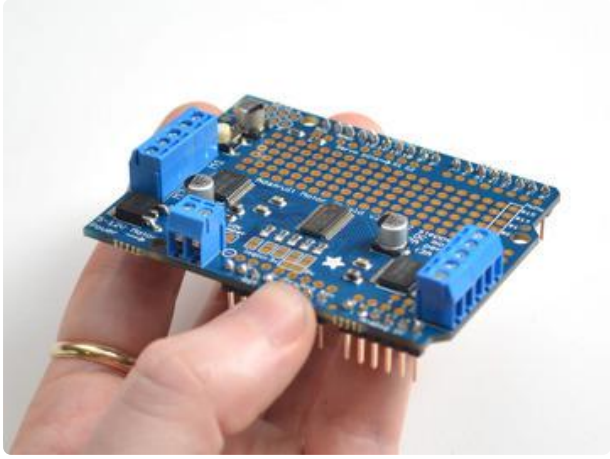
Next, you will attach the terminal blocks, power jumper and servo connections



That's it! Now you can install the terminal blocks and jumper...

Installing Terminal Blocks and more

After you have installed either normal or stacking headers, you must install the terminal blocks.

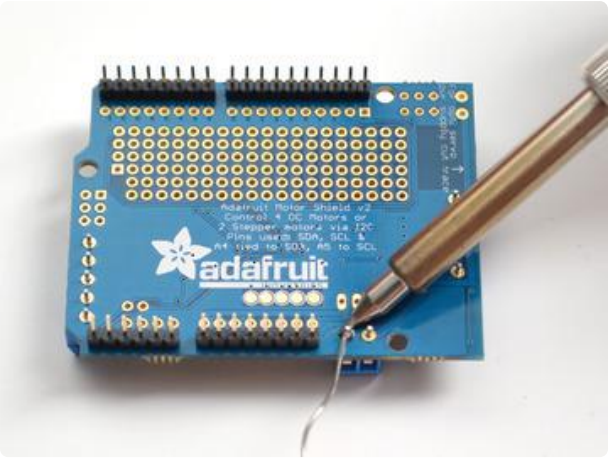


Next we will install the terminal blocks. These are how we will connect power and motors to the shield. They're much easier to use than soldering direct, just use a small screwdriver to release/attach wires!

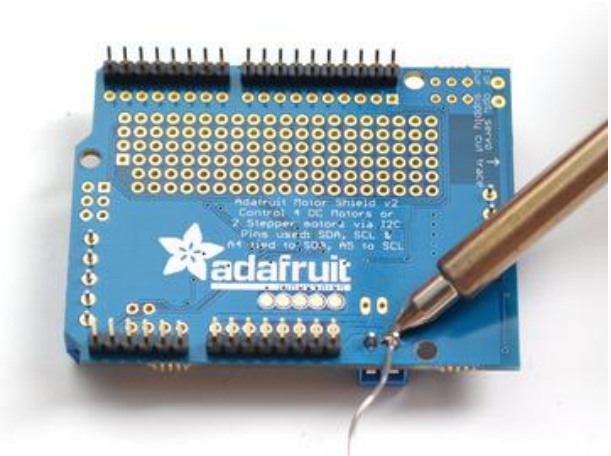
First, though, we must solder them in.

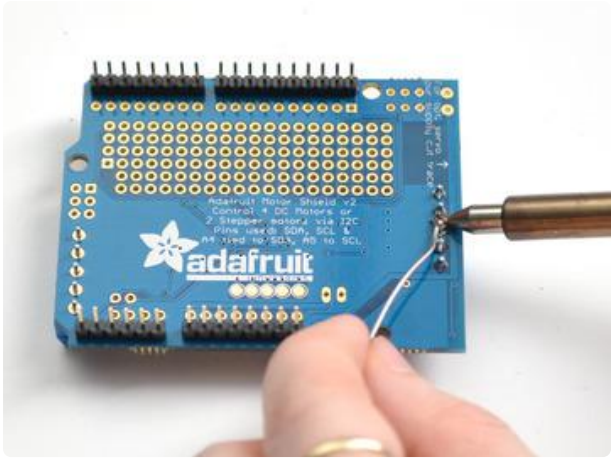
Slide the 3-pin terminal blocks into 2-pin terminal blocks so that you have 2 x 5-pin and 1 x 2-pin blocks. The two 5-pin sets go on either side. The 2-pin piece goes near the bottom of the shield. Make sure that the open holes of the terminal blocks face out!

Flip the board over so that you can see & solder the pins of the terminal blocks

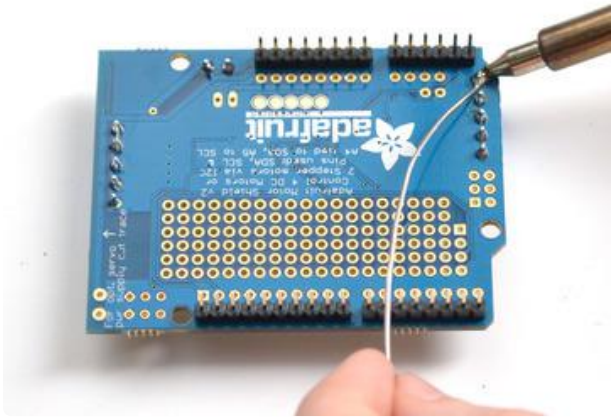


Solder in the two pins of the external power terminal-block

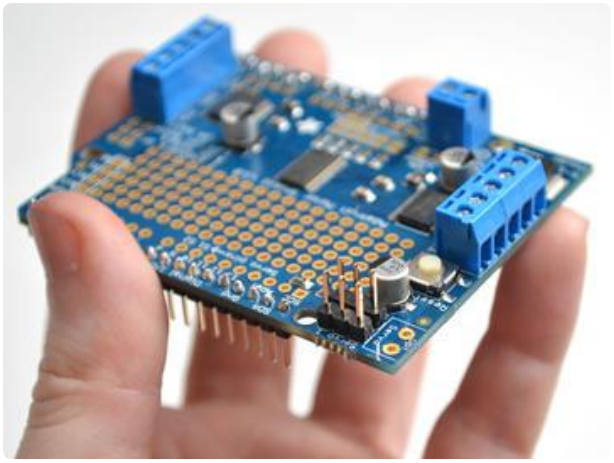




Solder in both motor blocks, 5 pads each

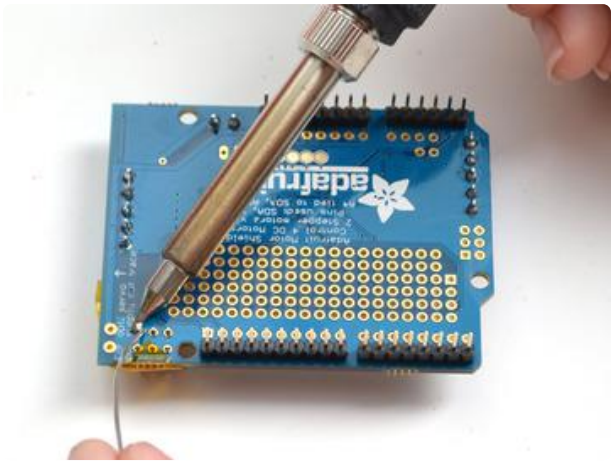


That's it for the terminal blocks. Next up, servo connections.

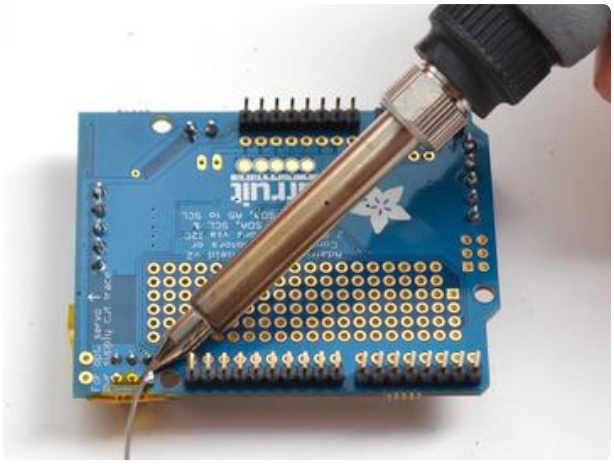


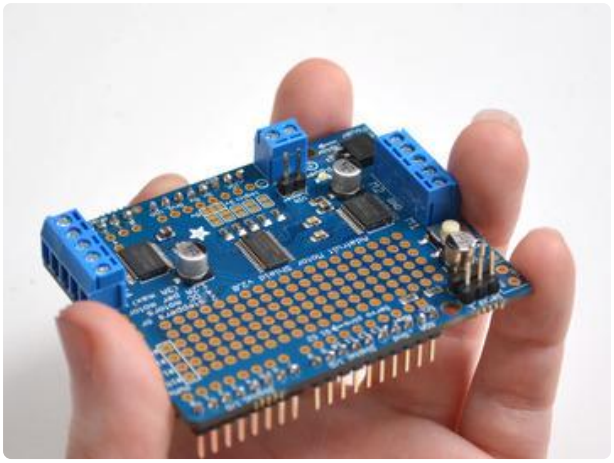
OK next up take the 2x3 pin header and place it with the short legs down into the top corner where it says SERVO 1 and SERVO 2

You might have to sort of angle the part a little to get it to fit into both sets of 3-pin holes. we did this so it wont fall out easily when you turn it over!

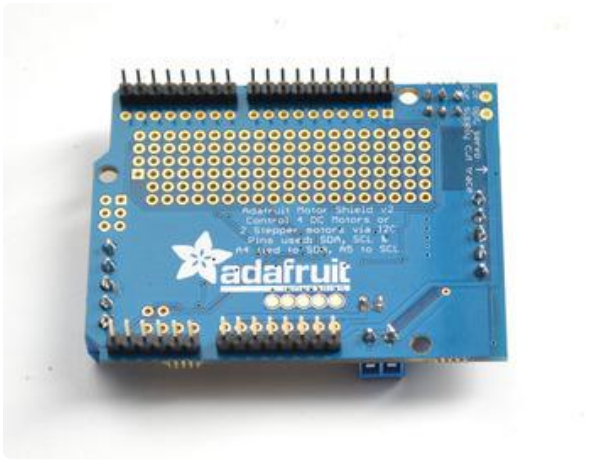


Then flip the board over and solder the 6 pins

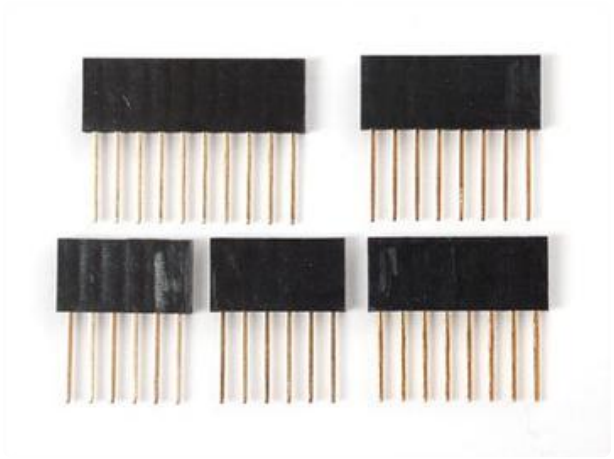




Finally, break off a 2-pin piece of header and place it next to the POWER terminal block, short legs down, tape it in place if necessary and solder it in.

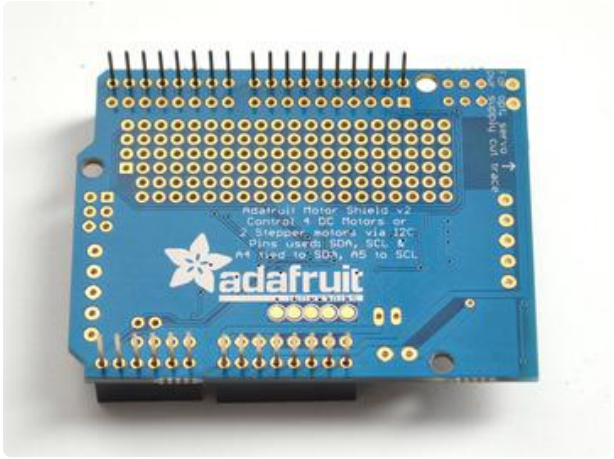


Installing with Stacking Headers

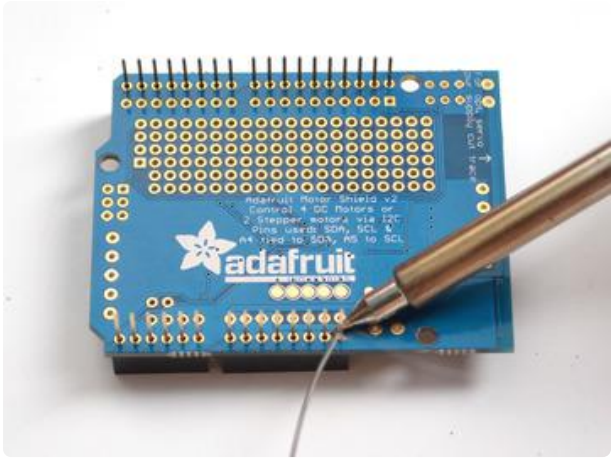


You will need to purchase Arduino stacking headers for this step, the shield does not come with them. (http://adafru.it/85)

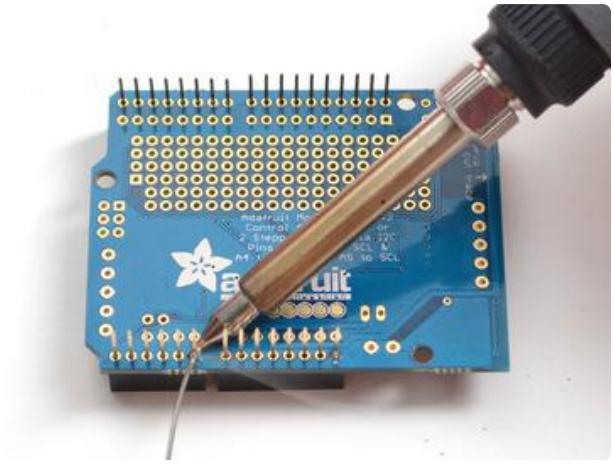
We don't show soldering in the 2x3 stacking header but you should solder that in as well - even though this shield does not use it, the one above may need those pins!

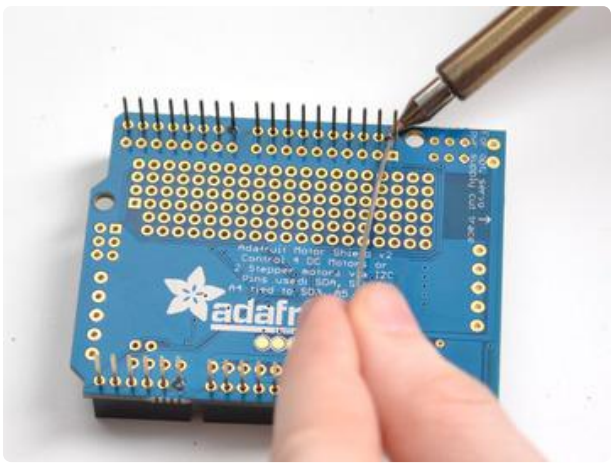


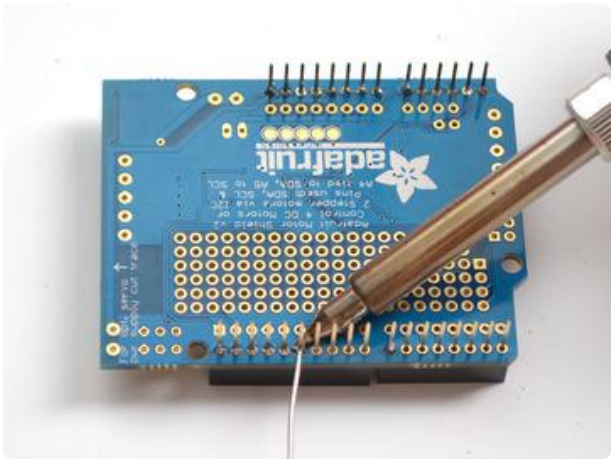
Start by sliding the 10 pin, 2 x 8 pin and 6-pin stacking headers into the outer rows of the shield from the top. Then flip the board over so its resting on the four headers. Pull on the legs if necessary to straighten them out.

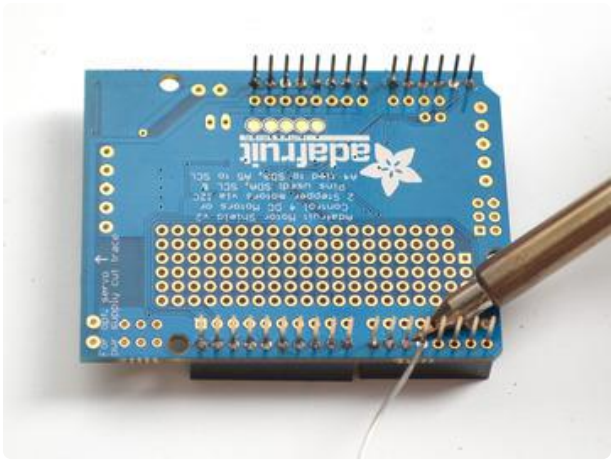


Tack one pin of each header, to get them set in place before more soldering. If the headers go crooked you can re-heat the one pin while re-positioning to straighten them up









Once you've tacked and straightened all the headers, go back and solder the remaining pins for each header.

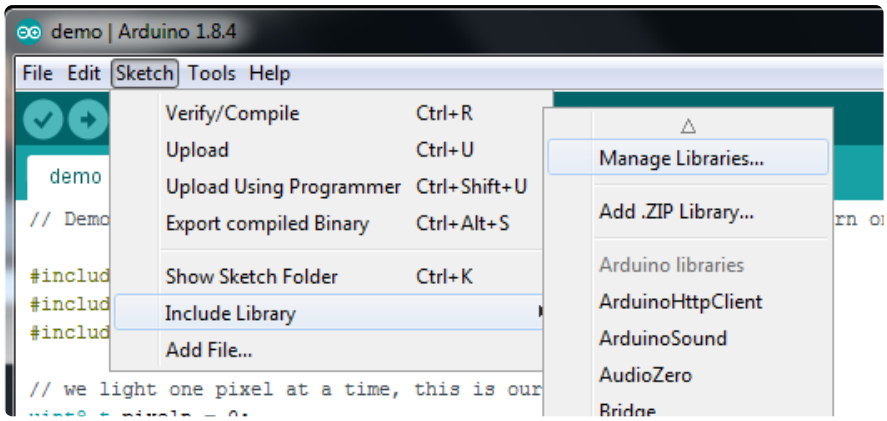
Install Software

Install Adafruit Motor Shield V2 library

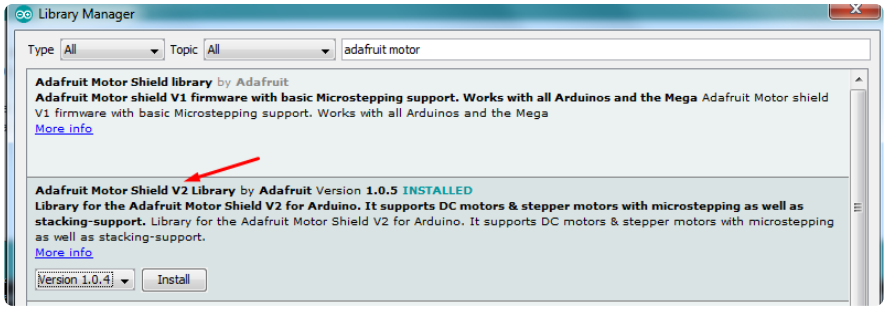
To use the shield on an Arduino, you'll need to install the Adafruit Motorshield v2 library. This library is not compatible with the older AF_Motor library used for v1 shields. However, if you have code for the older shield, adapting the code to use the new shield isn't difficult. We had to change the interface a little to support shield stacking, & we think its worth it!

To begin controlling motors, you will need to install the Adafruit_Motor_Shield_V2_Library library (code on our github repository) (https://adafru.it/ciN). It is available from the Arduino library manager so we recommend using that.

From the IDE open up the library manager...



And type in adafruit motor to locate the library. Click Install



If you plan to use AccelStepper for acceleration control or for simultaneous control of multiple stepper motors, you will also need to download and install the AccelStepper library:

AccelStepper Library

https://adafru.it/lpB

For more details on how to install Arduino libraries, check out our detailed tutorial! (https://adafru.it/aYM)

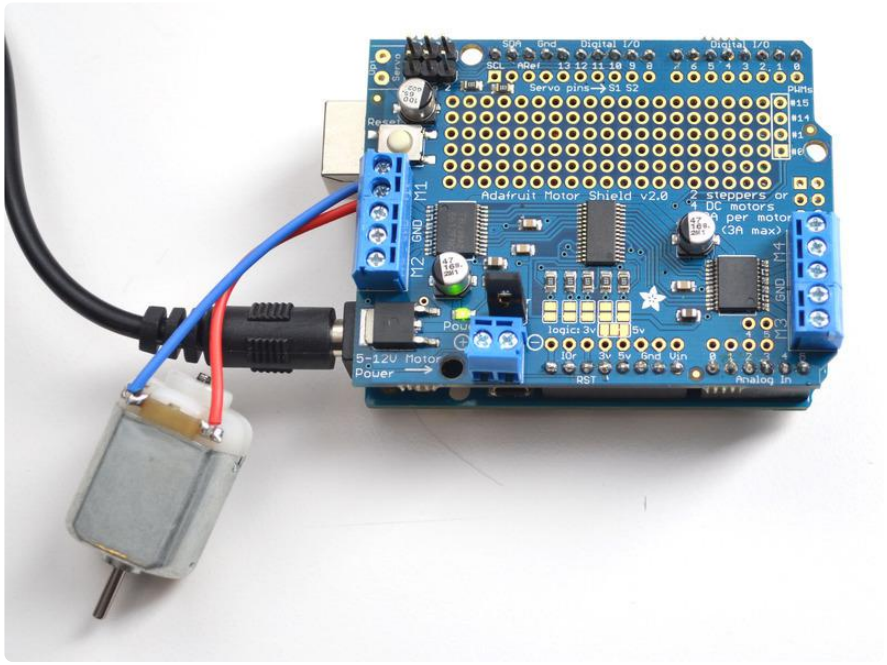
Running the Example Code

DC Motor

The library comes with a few examples to get you started up fast. We suggest getting started with the DC motor example. You can use any DC motor that can be powered by 6V-12VDC

First, restart the IDE to make sure the new library is loaded.

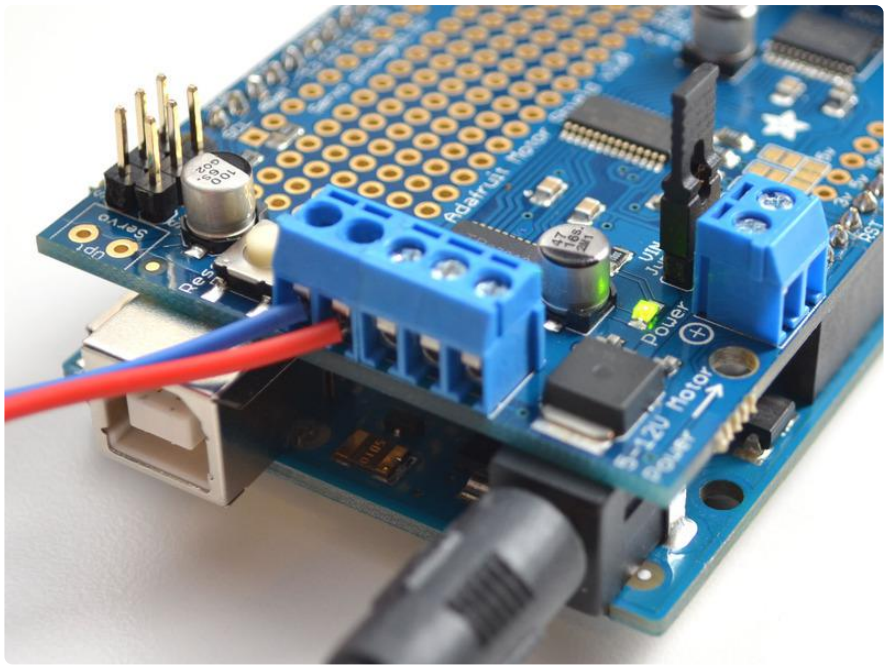
Plug the shield into the Arduino and connect a DC motor to motor port 1 - it does not matter which wire goes into which terminal block as motors are bi-directional. Connect to the top two terminal ports, do not connect to the middle pin (GND) See the photo below for the red and blue wire example. Be sure to screw down the terminal blocks to make a good connection!



You must also supply 5-12VDC to power the motor. There are two ways to do this

1. You can power the Arduino via the DC Barrel Jack and insert the VIN Jumper shown as the tall black handle right next to the green Power LED below
2. You can power the Arduino via the DC Barrel jack or USB port. Then Power the shield via the 5-12VDC motor power terminal port, the double terminal block next to the green Power LED and remove the VIN jumper

If the Green LED next to the power terminal block isn't lit up brightly do not continue!



Once you have verified the motor is connected properly and you have the power LED lit up brightly, we can upload our code.

In the IDE, load File->Examples->Adafruit_MotorShield->DCMotorTest

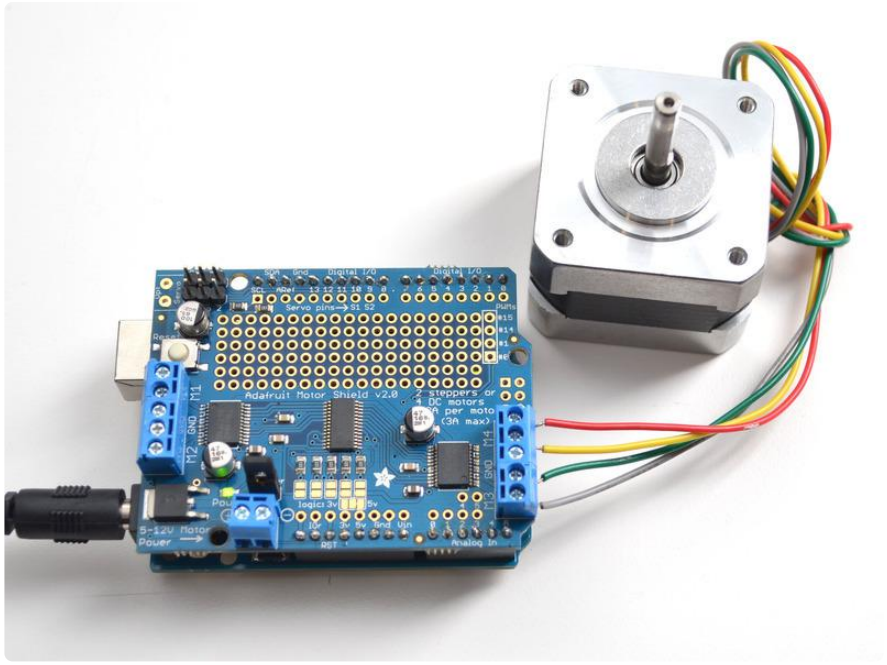
You should see and hear the DC motor turn on and move back and forth, attaching a slip of paper or tape as a 'flag' can help you visualize the movement if you have trouble seeing the movement

Stepper Motor Test

You can also test a stepper motor connection with the shield. The shield can run unip olar (5-wire and 6-wire) and bipolar (4-wire) steppers. It cannot run steppers with any other # of wires! The code is the same for unipolar or bipolar motors, the wiring is just slightly different.

Plug the shield into the Arduino and connect a stepper motor to motor port 2 - unlike DC motors, the wire order does 'matter'. Connect to the top two terminal ports (coil #1) and the bottom two terminal ports (coil #2).

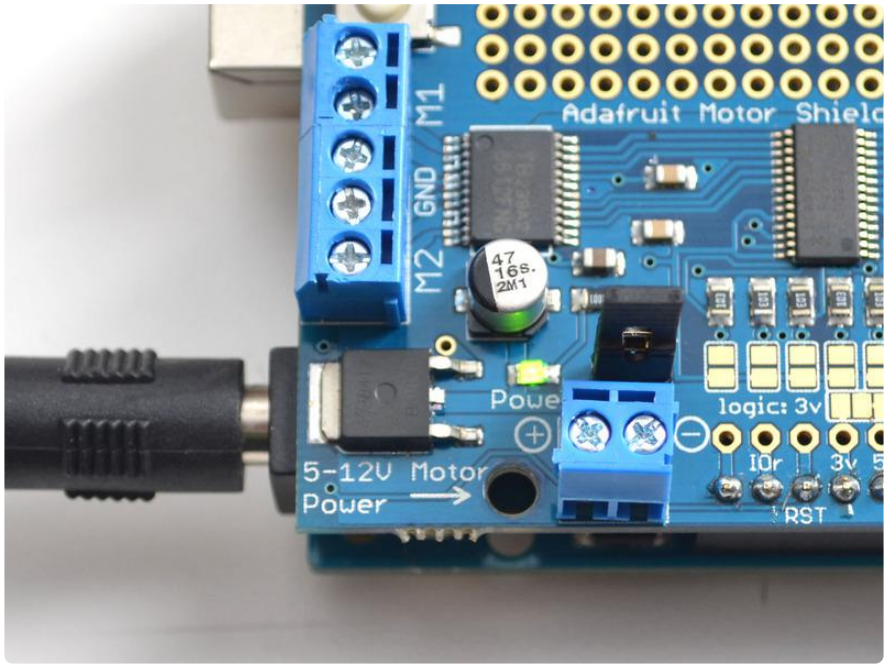
- If you have a bipolar motor, do not connect to the middle pin (GND).
- If you are using a unipolar motor with 5 wires, connect the common wire to GND.
- If you are using a unipolar motor with 6 wires, you can connect the two 'center coil wires' together to GND



You must also supply 5-12VDC to power the motor. There are two ways to do this

1. You can power the Arduino via the DC Barrel Jack and insert the VIN Jumper shown as the tall black handle right next to the green Power LED below
2. You can power the Arduino via the DC Barrel jack or USB port. Then Power the shield via the 5-12VDC motor power terminal port, the double terminal block next to the green Power LED and remove the VIN jumper

If the Green LED isn't lit up brightly do not continue - you must power it via the VIN jumper or the terminal block

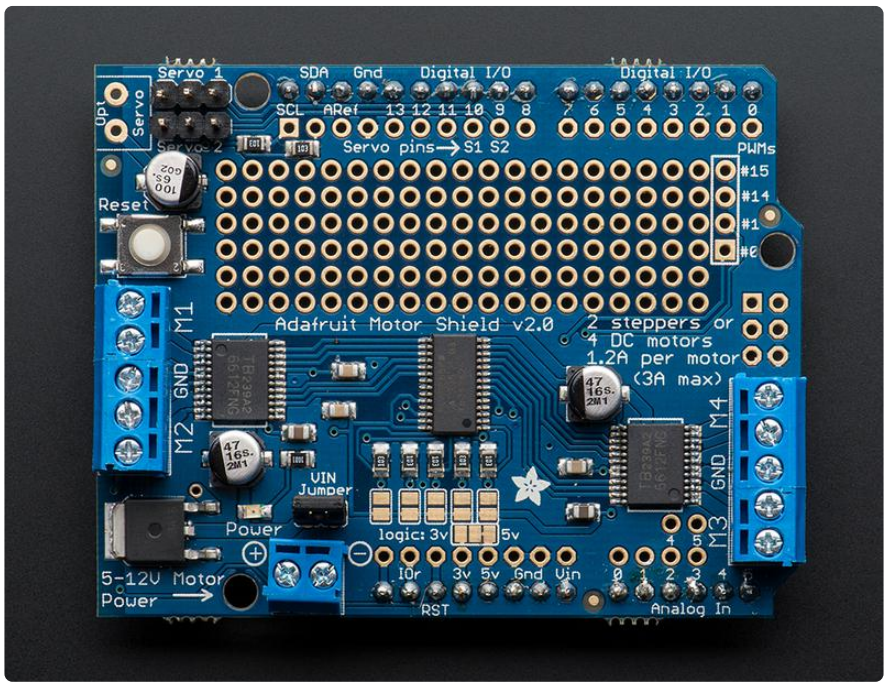


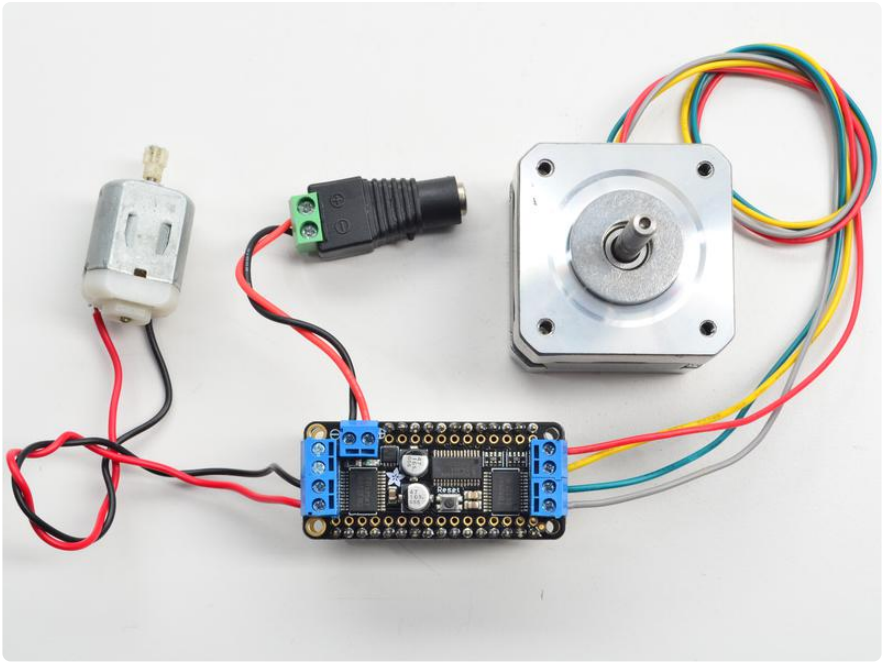
Once you have verified the motor is connected properly and you have the power LED lit up brightly, we can upload our code.

In the IDE, load File->Examples->Adafruit_MotorShield->StepperTest

You should see and hear the stepper motor turn on and move back and forth, attaching a slip of paper or tape as a 'flag' can help you visualize the movement if you have trouble seeing the movement. There are four ways to move a stepper, with varying speed, torque and smoothness tradeoffs. This example code will demonstrate all four.

Library Reference





class Adafruit_MotorShield;

The Adafruit_MotorShield class represents a motor shield and must be instantiated before any DCMotors or StepperMotors can be used. You will need to declare one Adafruit_MotorShield for each shield in your system.

Adafruit_MotorShield(uint8_t addr = 0x60);

The constructor takes one optional parameter to specify the i2c address of the shield. The default address of the constructor (0x60) matches the default address of the boards as shipped. If you have more than one shield in your system, each shield must have a unique address.

void begin(uint16_t freq = 1600);

begin() must be called in setup() to initialize the shield. An optional frequency parameter can be used to specify something other than the default maximum: 1.6KHz PWM frequency.

Adafruit_DCMotor *getMotor(uint8_t n);

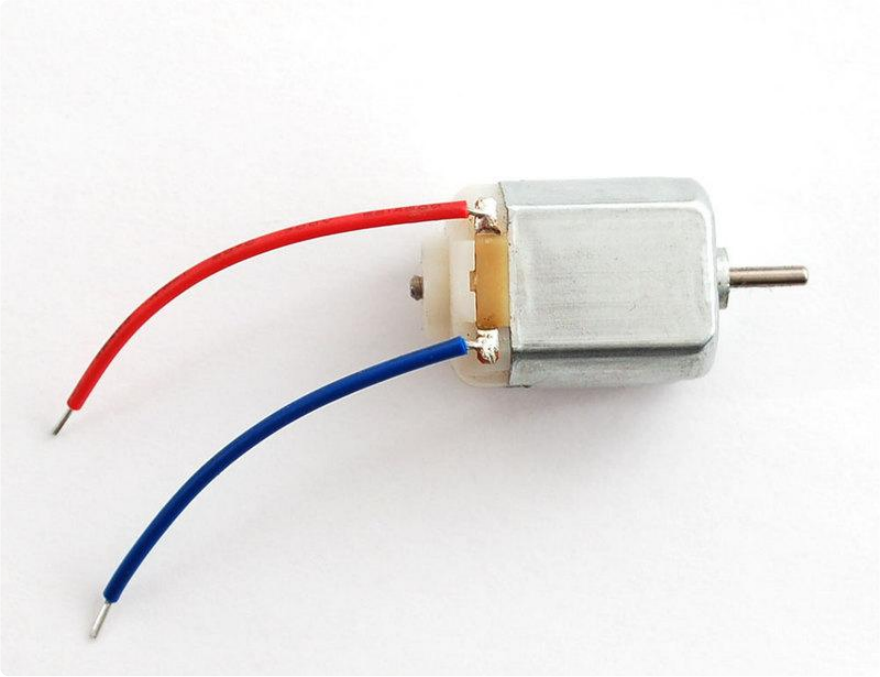
This function returns one of 4 pre-defined DC motor objects controlled by the shield. The parameter specifies the associated motor channel: 1-4.

Adafruit_StepperMotor *getStepper(uint16_t steps, uint8_t n);

This function returns one of 2 pre-defined stepper motor objects controlled by the shield.
The first parameter specifies the number of steps per revolution.
The second parameter specifies the associated stepper channel: 1-2.

void setPWM(uint8_t pin, uint16_t val);
void setPin(uint8_t pin, boolean val);

These are low-level functions to control pins on the on-board PWM driver chip. These functions are intended for internal use only.



class Adafruit_DCMotor

The Adafruit_DCMotor class represents a DC motor attached to the shield. You must declare an Adafruit_DCMotor for each motor in your system.

Adafruit_DCMotor(void);

The constructor takes no arguments. The motor object is typically initialized by assigning a motor object retrieved from the shield class as below:

```
// Create the motor shield object with the default I2C address
Adafruit_MotorShield AFMS = Adafruit_MotorShield();

// Select which 'port' M1, M2, M3 or M4. In this case, M1
```

```
Adafruit_DCMotor *myMotor = AFMS.getMotor(1);
// You can also make another motor on port M2
Adafruit_DCMotor *myOtherMotor = AFMS.getMotor(2);
```

void run(uint8_t);

The run() function controls the motor state. The parameter can have one of 3 values:

- FORWARD - Rotate in a forward direction
- BACKWARD - Rotate in the reverse direction
- RELEASE - Stop rotation

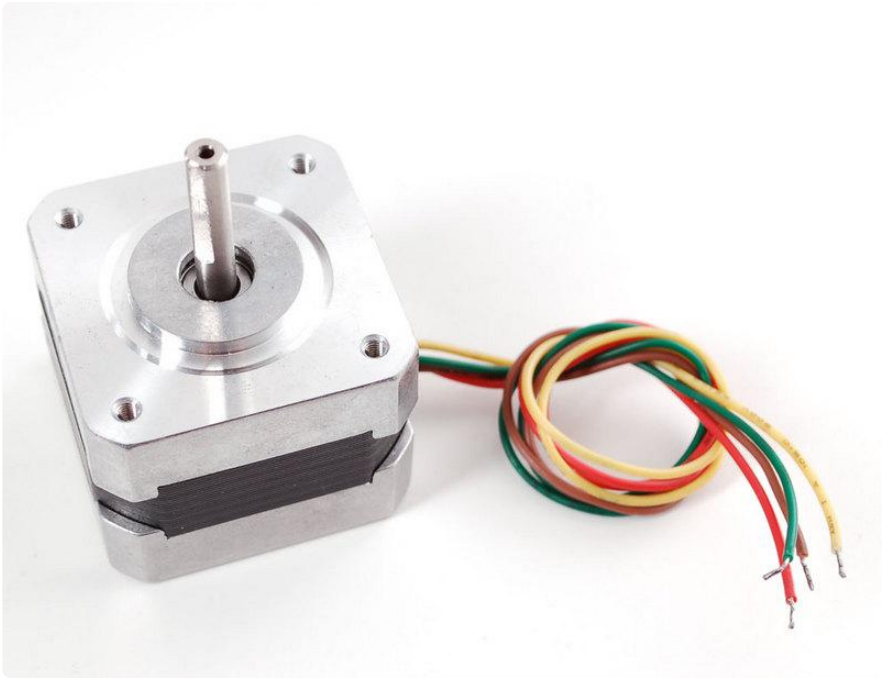
Note that the "FORWARD" and "BACKWARD" directions are arbitrary. If they do not match the actual direction of your vehicle or robot, simple swap the motor leads.

Also note that "RELEASE" simply cuts power to the motor. It does not apply any braking.

void setSpeed(uint8_t);

The setSpeed() function controls the power level delivered to the motor. The speed parameter is a value between 0 and 255.

Note that setSpeed just controls the power delivered to the motor. The actual speed of the motor will depend on several factors, including: The motor, the power supply and the load.



class Adafruit_StepperMotor

The Adafruit_StepperMotor class represents a stepper motor attached to the shield. You must declare an Adafruit_StepperMotor for each stepper motor in your system.

Adafruit_StepperMotor(void);

The constructor takes no arguments. The stepper motor is typically initialized by assigning a stepper object retrieved from the shield as below:

```
// Create the motor shield object with the default I2C address
Adafruit_MotorShield AFMS = Adafruit_MotorShield();

// Connect a stepper motor with 200 steps per revolution (1.8 degree)
// to motor port #2 (M3 and M4)
Adafruit_StepperMotor *myMotor = AFMS.getStepper(200, 2);
```

void step(uint16_t steps, uint8_t dir, uint8_t style = SINGLE);

The step() function controls stepper motion.

- The first parameter specifies how many steps to move.
- The second parameter specifies the direction: FORWARD or BACKWARD
- The last parameter specifies the stepping style: SINGLE, DOUBLE, INTERLEAVED or MICROSTEP

The ste() function is synchronous and does not return until all steps are complete. When complete the motor remains powered to apply "holding torque" to maintain

position.

void setSpeed(uint16_t);

The setSpeed() function controls the speed of the stepper motor rotation. Speed is specified in RPM.

uint8_t onestep(uint8_t dir, uint8_t style);

The oneStep() function is a low-level internal function called by step(). But it can be useful to call on its own to implement more advanced functions such as acceleration or coordinating simultaneous movement of multiple stepper motors. The direction and style parameters are the same as for step(), but onestep() steps exactly once.

Note: Calling step() with a step count of 1 is not the same as calling onestep(). The step function has a delay based on the speed set in setSpeed(). onestep() has no delay.

void release(void);

The release() function removes all power from the motor. Call this function to reduce power requirements if holding torque is not required to maintain position.

Arduino Library Docs

Arduino Library Docs (https://adafru.it/AvQ)

Powering Motors

Motors need a lot of energy, especially cheap motors since they're less efficient.

Voltage requirements:

The first important thing to figure out what voltage the motor is going to use. If you're lucky your motor came with some sort of specifications. Some small hobby motors are only intended to run at 1.5V, but its just as common to have 6-12V motors. The motor controllers on this shield are designed to run from 5V to 12V.

MOST 1.5-3V MOTORS WILL NOT WORK

Current requirements:

The second thing to figure out is how much current your motor will need. The motor driver chips that come with the kit are designed to provide up to 1.2 A per motor, with 3A peak current. Note that the 'peak' rating is for very brief peaks such as during startup. Peak current levels can only be tolerated for a few milliseconds. If you will be pushing the 1.2A continuous limit you'll probably want to put a heat-sink on the motor driver, otherwise you will get thermal failure, possibly burning out the chip.

You can't run motors off of a 9V battery so don't waste your time/batteries!

Use a big Lead Acid or NiMH battery pack. Its also very much suggested that you set up two power supplies (split supply) one for the Arduino and one for the motors. 99% of 'weird motor problems' are due to noise on the power line from sharing power supplies and/or not having a powerful enough supply! Even small DC motors can draw up to 3 Amps when they stall.

Setting up your shield for powering Hobby Servos

Servos are powered off of the same regulated 5V that the Arduino uses. This is OK for the small hobby servos suggested. Basically, power up your Arduino with the USB port or DC barrel jack and you're good to go. If you want something beefier, cut the trace going to the optional servo power terminal and wire up your own 5-6V supply!

Setting up your shield for powering DC and Stepper Motors

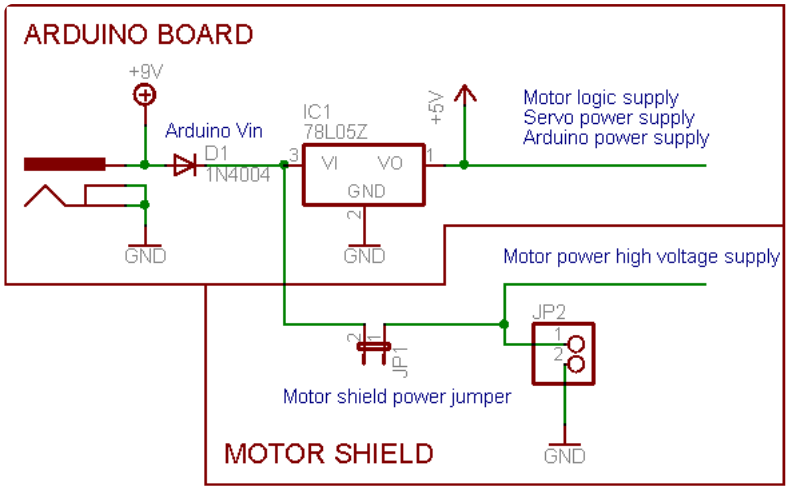
The motors are powered off of a 'high voltage supply' and NOT the regulated 5V. Do n't connect the motor power supply to the Arduino's 5V power pin. This is a very very very bad idea unless you are sure you know what you're doing! You could damage your Arduino and/or USB port!

There are two places you can get your motor 'high voltage supply' from.

1. One is the DC barrel jack on the Arduino board
2. The other is the 2-terminal block on the shield that is labeled DC Motor Power 5-12VDC.

The DC Jack on the Arduino has a protection diode so you won't be able to mess things up too bad if you plug in the wrong kind of power. The terminal block has a protection FET so you will not damage the arduino/shield if you wire up your battery supply backwards, but it wont work either!

Here's how it works:



If you would like to have a single DC power supply for the Arduino and motors

Say a wall adapter or a single battery pack with 6-12VDC output, simply plug it into the DC jack on the Arduino or the 2-pin power terminal block on the shield. Place the power jumper on the motor shield.

Note that you may have problems with Arduino resets if the battery supply is not able to provide constant power, so it is not a suggested way of powering your motor project. You cannot use a 9V battery for this, it must be 4 to 8 AA batteries or a single /double lead acid battery pack.

If you would like to have the Arduino powered off of USB and the motors powered off of a DC power supply

Plug in the USB cable. Then connect the motor supply to the power terminal block on the shield. Do not place the jumper on the shield.

This is a suggested method of powering your motor project as it has a split supply, one power supply for logic, and one supply for motors

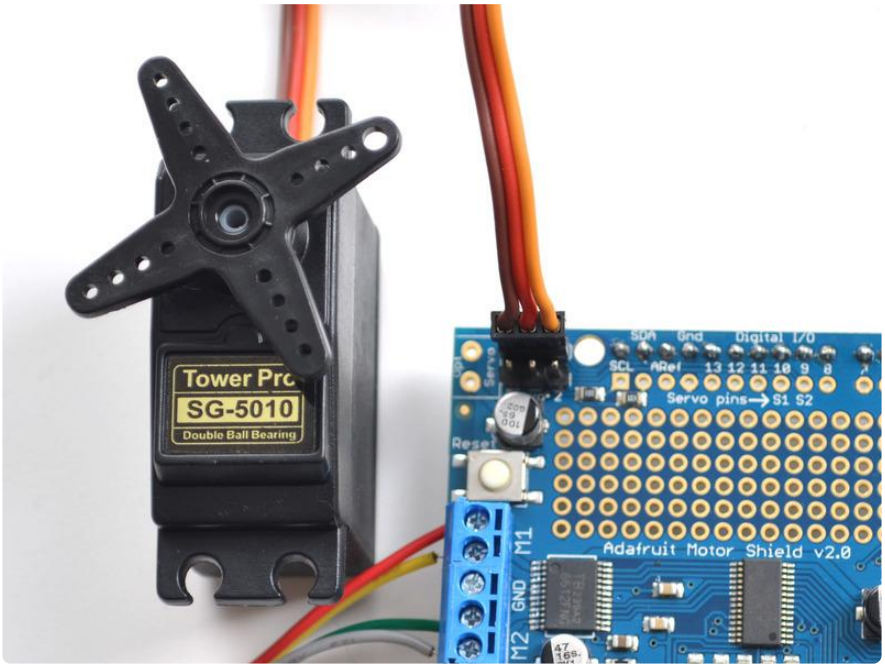
If you would like to have 2 separate DC power supplies for the Arduino and motors.

Plug in the supply for the Arduino into the DC jack, and connect the motor supply to the power terminal block. Make sure the jumper is removed from the motor shield.

No matter what, if you want to use the DC motor/Stepper system the motor shield LED should be lit indicating good motor power

---

Using RC Servos



Hobby servos are the easiest way to get going with motor control. They have a 3-pin 0.1" female header connection with +5V, ground and signal inputs. The motor shield simply brings out the PWM output lines from Arduino pins 9 and 10 to two 3-pin headers so that its easy to plug in and go. They can take a lot of power so a 9V battery wont last more than a few minutes!

The nice thing about using the onboard PWM is that its very precise and goes about its business in the background. You can use the built in Servo library

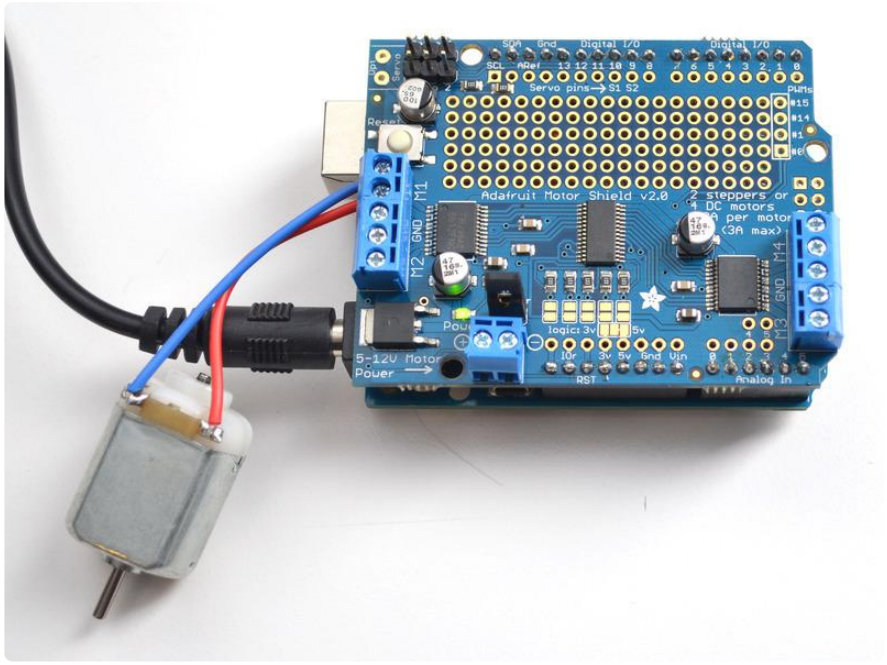
Using the servos is easy, please read the official Arduino documentation for how to use them and see the example Servo sketches in the IDE (https://adafru.it/aOD).

Powering Servos

Power for the Servos comes from the Arduino's on-board 5V regulator, powered directly from the USB or DC power jack on the Arduino. If you need an external supply, cut the 5v trace on the bottom of the board and connect a 5V or 6V DC supply directly to the Opt Servo power input. Using an external supply is for advanced users as you can accidentally destroy the servos by connecting a power supply incorrectly!

When using external servo power, be careful not to let it short out against the USB socket shell on the processor board. Insulate the top of the USB socket with some electrical tape.

Using DC Motors



DC motors are used for all sort of robotic projects.

The motor shield can drive up to 4 DC motors bi-directionally. That means they can be driven forwards and backwards. The speed can also be varied at 0.5% increments using the high-quality built in PWM. This means the speed is very smooth and won't vary!

Note that the H-bridge chip is not meant for driving continuous loads of 1.2A, so this is for small motors. Check the datasheet for information about the motor to verify its OK!

Connecting DC Motors

To connect a motor, simply solder two wires to the terminals and then connect them to either the M1, M2, M3, or M4. Then follow these steps in your sketch

Include the required libraries

Make sure you #include the required libraries

```
#include &lt;Wire.h&gt;
#include &lt;Adafruit_MotorShield.h&gt;
#include "utility/Adafruit_MS_PWMServoDriver.h"
```

Create the Adafruit_MotorShield object

```
Adafruit_MotorShield AFMS = Adafruit_MotorShield();
```

Create the DC motor object

Request the DC motor from the Adafruit_MotorShield:

```
Adafruit_DCMotor *myMotor = AFMS.getMotor(1);
```

with getMotor(port#). Port# is which port it is connected to. If you're using M1 its 1, M2 use 2, M3 use 3 and M4 use 4

Connect to the Controller

In your `setup()` function, call `begin()` on the Adafruit_MotorShield object:

```
AFMS.begin();
```

Set default speed

Set the speed of the motor using setSpeed(speed) where the speed ranges from 0 (stopped) to 255 (full speed). You can set the speed whenever you want.

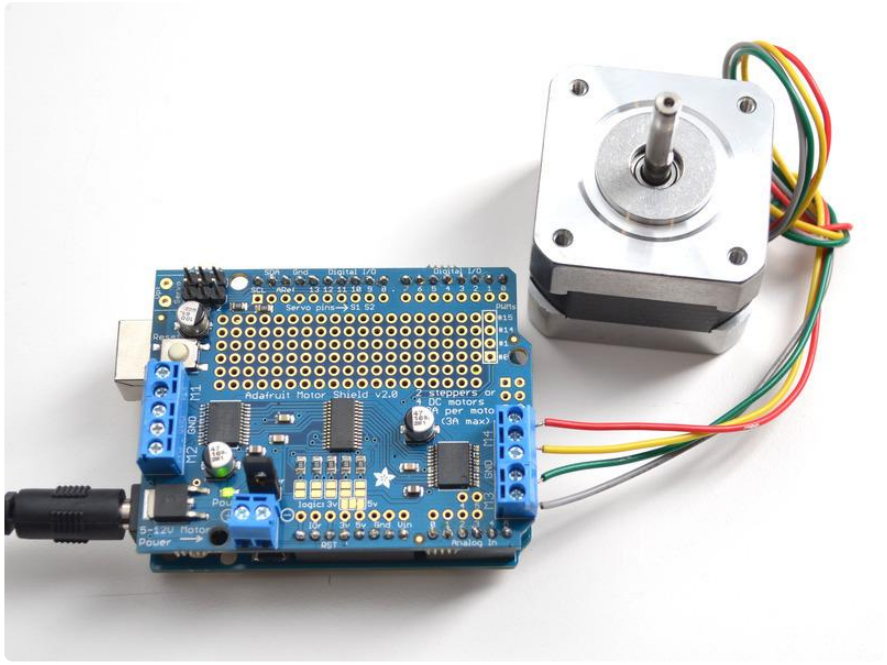
```
myMotor-&gt;setSpeed(150);
```

Run the motor

To run the motor, call run(direction) where direction is FORWARD, BACKWARD or REL EASE. Of course, the Arduino doesn't actually know if the motor is 'forward' or 'backward', so if you want to change which way it thinks is forward, simply swap the two wires from the motor to the shield.

```
myMotor-&gt;run(FORWARD);
```

Using Stepper Motors



Stepper motors are great for (semi-)precise control, perfect for many robot and CNC projects. This motor shield supports up to 2 stepper motors. The library works identically for bi-polar and uni-polar motors

Before connecting a motor, be sure to check the motor specifications for compatibility with the shield (https://adafru.it/r2d).

For unipolar motors: to connect up the stepper, first figure out which pins connected to which coil, and which pins are the center taps. If its a 5-wire motor then there will be 1 that is the center tap for both coils. Theres plenty of tutorials online on how to reverse engineer the coils pinout. (https://adafru.it/aOO) The center taps should both be connected together to the GND terminal on the motor shield output block. then coil 1 should connect to one motor port (say M1 or M3) and coil 2 should connect to the other motor port (M2 or M4).

For bipolar motors: its just like unipolar motors except theres no 5th wire to connect to ground. The code is exactly the same.

Running a stepper is a little more intricate than running a DC motor but its still very easy

Include the required libraries

Make sure you #include the required libraries

```
#include &lt;Wire.h&gt;
#include &lt;Adafruit_MotorShield.h&gt;
#include "utility/Adafruit_PWMServoDriver.h"
```

Create the Adafruit_MotorShield object

```
Adafruit_MotorShield AFMS = Adafruit_MotorShield();
```

Create the stepper motor object

Request the Stepper motor from the Adafruit_MotorShield:

```
Adafruit_StepperMotor *myMotor = AFMS.getStepper(200, 2);
```

...with `getStepper(steps, stepper#)`.

`Steps` indicates how many steps per revolution the motor has. A 7.5 degree/step motor has 360/7.5 = 48 steps.

`Stepper#` is which port it is connected to. If you're using M1 and M2, its port 1. If you're using M3 and M4 indicate port 2

Set default speed

Set the speed of the motor using `setSpeed(rpm)` where rpm is how many revolutions per minute you want the stepper to turn.

Run the motor

Then every time you want the motor to move, call the `step(#steps, direction, steptype)` procedure. #steps is how many steps you'd like it to take. direction is

either FORWARD or BACKWARD and the step type is SINGLE, DOUBLE, INTERLEAVE or MICROSTEP.

- "Single" means single-coil activation
- "Double" means 2 coils are activated at once (for higher torque)
- "Interleave" means that it alternates between single and double to get twice the resolution (but of course its half the speed).
- "Microstepping" is a method where the coils are PWM'd to create smooth motion between steps.

Theres tons of information about the pros and cons of these different stepping methods in the resources page.

You can use whichever stepping method you want, changing it "on the fly" to as you may want minimum power, more torque, or more precision.

By default, the motor will 'hold' the position after its done stepping. If you want to release all the coils, so that it can spin freely, call `release()`

The stepping commands are 'blocking' and will return once the steps have finished.

Because the stepping commands 'block' - you have to instruct the Stepper motors each time you want them to move. If you want to have more of a 'background task' stepper control, check out the AccelStepper library (https://adafru.it/X1e) . There are several AccelStepper examples included with the motor shield library.

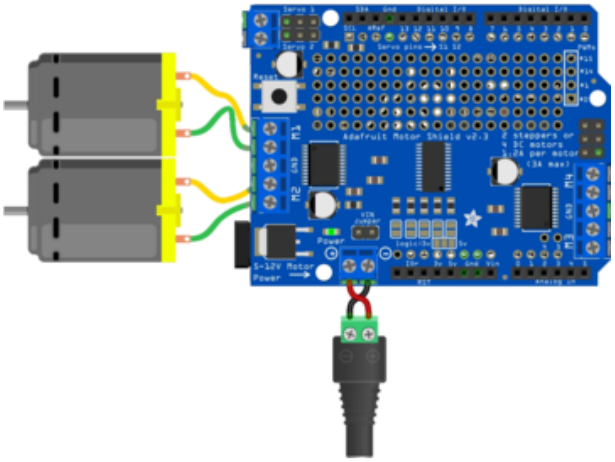
Python & CircuitPython

We've written a handy CircuitPython library for the various DC Motor and Stepper kits called Adafruit CircuitPython MotorKit (https://adafru.it/DdU) that handles all the complicated setup for you. All you need to do is import the appropriate class from the library, and then all the features of that class are available for use. We're going to show you how to import the `MotorKit` class and use it to control DC and stepper motors with the Adafruit Stepper + DC Motor Shield.

CircuitPython Microcontroller Wiring

First assemble the Shield exactly as shown in the previous pages. There's no wiring needed to connect the Shield to the Metro. The example below shows wiring two DC motors to the Shield once it has been attached to a Metro. You'll want to connect a

barrel jack to the power terminal to attach an appropriate external power source to the Shield. The Shield will not function without an external power source!



- Connect the two motor wires from the first motor to the M1 terminal on the Shield.
- Connect the two motor wires from the second motor to the M2 terminal on the Shield.
- Connect the positive side of the power terminal to the positive side of the barrel jack.
- Connect the negative side of the power terminal to the negative side of the barrel jack.

CircuitPython Installation of MotorKit and Necessary Libraries

You'll need to install a few libraries on your Metro board.

First make sure you are running the latest version of Adafruit CircuitPython (https://adafru.it/Amd) for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (https://adafru.it/uap). Our CircuitPython starter guide has a great page on how to install the library bundle (https://adafru.it/ABU).

If you choose, you can manually install the libraries individually on your board:

- adafruit_pca9685
- adafruit_bus_device
- adafruit_register
- adafruit_motor
- adafruit_motorkit

Before continuing make sure your board's lib folder or root filesystem has the adafruit_pca9685.mpy, adafruit_register, adafruit_motor, adafruit_bus_device and adafruit_motorkit files and folders copied over.

Next connect to the board's serial REPL (https://adafru.it/Awz) so you are at the CircuitPython >>> prompt.

CircuitPython Usage

To demonstrate the usage, we'll initialise the library and use Python code to control DC and stepper motors from the board's Python REPL.

First you'll need to import and initialize the MotorKit class.

```
from adafruit_motorkit import MotorKit
kit = MotorKit()
```

DC Motors

The four motor spots on the Shield are available as `motor1`, `motor2`, `motor3`, and `motor4`.

In this example we'll use `motor1`.

Note: For small DC motors like sold in the shop you might run into problems with electrical noise they generate and erratic behavior on your board. If you see erratic behavior like the motor not spinning or the board resetting at high motor speeds this is likely the problem. See this motor guide FAQ page for information on capacitors you can solder to the motor to reduce noise (https://adafru.it/scl).

Now to move a motor you can set the `throttle` attribute. We don't call it speed because it doesn't correlate to a particular number of revolutions per minute (RPM). RPM depends on the motor and the voltage which is unknown.

For example to drive motor M1 forward at a full speed you set it to `1.0`:

```
kit.motor1.throttle = 1.0
```

To run the motor at half throttle forward use a decimal:

```
kit.motor1.throttle = 0.5
```

Or to reverse the direction use a negative throttle:

```
kit.motor1.throttle = -0.5
```

You can stop the motor with a throttle of `0`:

```
kit.motor1.throttle = 0
```

To let the motor coast and then spin freely set throttle to `None`.

```
kit.motor1.throttle = None
```

That's all there is to controlling DC motors with CircuitPython! With DC motors you can build fun moving projects like robots or remote controlled cars that glide around with ease.

Stepper Motors

Similar DC motors, stepper motors are available as `stepper1` and `stepper2`. `stepper1` is made up of the M1 and M2 terminals, and `stepper2` is made up of the M3 and M4 terminals.

We'll use `stepper1` in our example.

The most basic function (and the default) is to do one single coil step.

```
kit.stepper1.onestep()
```

You can also call the onestep function with two optional keyword arguments. To use these, you'll need to import stepper as well.

```
from adafruit_motor import stepper
```

Then you have access to the following options:

- `direction`, which should be one of the following constant values:
  - `stepper.FORWARD` (default)
  - `stepper.BACKWARD`.

- `style`, which should be one of the values:
  - `stepper.SINGLE` (default) for a full step rotation to a position where one single coil is powered
  - `stepper.DOUBLE` for a full step rotation to position where two coils are powered providing more torque
  - `stepper.INTERLEAVED` for a half step rotation interleaving single and double coil positions and torque
  - `stepper.MICROSTEP` for a microstep rotation to a position where two coils are partially active.
- `release()` which releases all the coils so the motor can free spin, and also won't use any power

The function returns the current step 'position' in microsteps which can be handy to understand how far the stepper has moved, or you can ignore the result.

To take a double-coil step backward call:

```
kit.stepper1.onestep(direction=stepper.BACKWARD, style=stepper.DOUBLE)
```

You can even use a loop to continuously call `onestep` and move the stepper, for example a loop of `200` microsteps forward for smooth movement:

```
for i in range(200):
    kit.stepper1.onestep(style=stepper.MICROSTEP)
```

That's all there is to controlling a stepper motor from CircuitPython! Steppers are handy motors for when you need smooth or precise control of something--for example 3D printers and CNC machines use steppers to precisely move tools around surfaces.

Full Example Code

For DC motors:

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

"""Simple test for using adafruit_motorkit with a DC motor"""
import time
import board
from adafruit_motorkit import MotorKit

kit = MotorKit(i2c=board.I2C())
```

```
kit.motor1.throttle = 1.0
time.sleep(0.5)
kit.motor1.throttle = 0
```

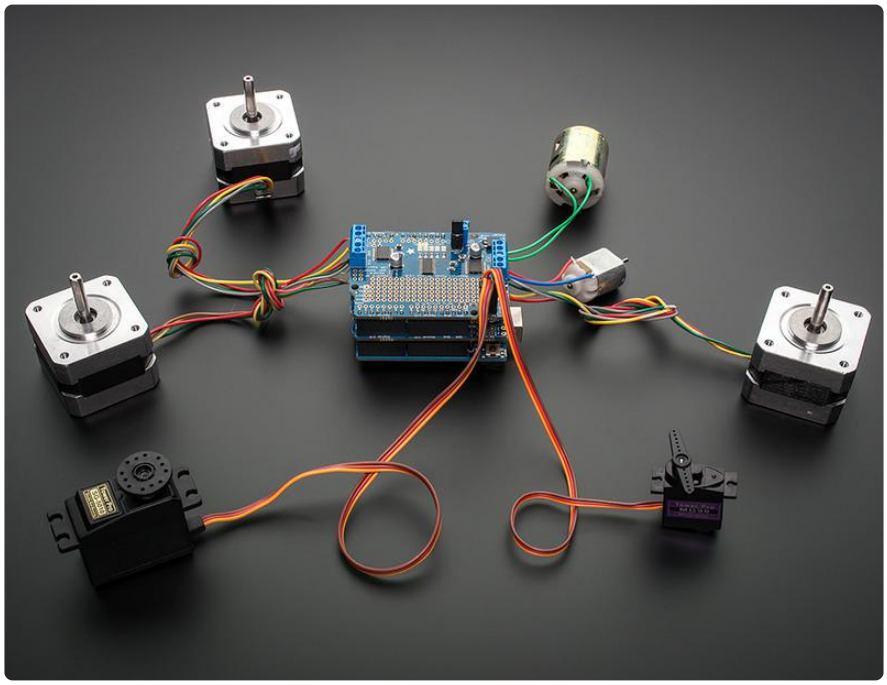
For stepper motors:

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

"""Simple test for using adafruit_motorkit with a stepper motor"""
import time
import board
from adafruit_motorkit import MotorKit

kit = MotorKit(i2c=board.I2C())

for i in range(100):
    kit.stepper1.onestep()
    time.sleep(0.01)
```

Python Docs

Python Docs (https://adafru.it/DeE)

Stacking Shields



One of the cool things about this shield design is that it is possible to stack shields. Every shield you stack can control another 2 steppers or 4 DC motors (or a mix of the two)

You can stack up to 32 shields for a total of 64 steppers or 128 DC motors! Most people will probably just stack two or maybe three but hey, you never know. (PS if you drive 64 steppers from one of these shields send us a photo, OK?)

Note that stacking shields does not increase the servo connections - those are hard-wired to the Arduino digital 9 & 10 pins. If you need to control a lot of servos, you can use our 16-channel servo shield and stack it with this shield to add a crazy large # of servos. (http://adafru.it/1411)

Stacking shields is very easy. Each shield you want to stack on top of must have stacking headers installed. Check our instructions for how to do so. (https://adafru.it/ciP) The top shield does not have to have stacking headers unless you eventually want to put something on top of it.

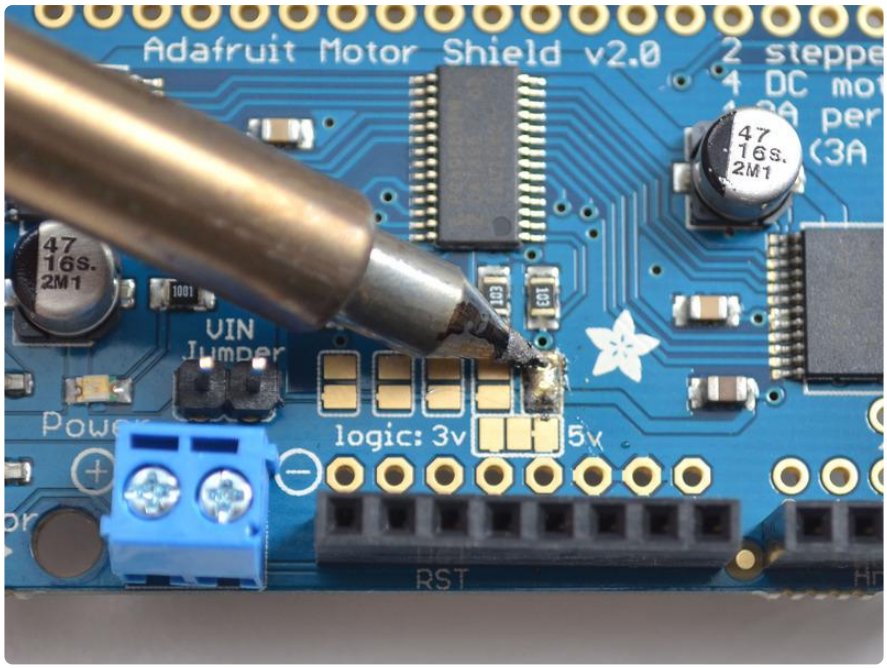
The only thing to watch for when stacking shields is every shield must have a unique I2C address. The default address is 0x60. You can adjust the address of the shields to range from 0x60 to 0x7F for a total of 32 unique addresses.

Addressing the Shields

Each board in the chain must be assigned a unique address. This is done with the address jumpers on the lower edge of the board. The I2C base address for each board is 0x60. The binary address that you program with the address jumpers is added to the base I2C address.

To program the address offset, use a drop of solder to bridge the corresponding address jumper for each binary '1' in the address.

The right-most jumper is address bit #0, then to the left of that is address bit #1, etc up to address bit #4



Board 0: Address = 0x60 Offset = binary 0000 (no jumpers required)
Board 1: Address = 0x61 Offset = binary 0001 (bridge A0 as in the photo above)
Board 2: Address = 0x62 Offset = binary 0010 (bridge A1, to the left of A0)
Board 3: Address = 0x63 Offset = binary 0011 (bridge A0 & A1, two rightmost jumpers)
Board 4: Address = 0x64 Offset = binary 0100 (bridge A2, middle jumper)

etc.

Note that address 0x70 is the "all call" address for the controller chip on the shield. All boards will respond to address 0x70 - regardless of the address jumper settings.

Writing Code for Multiple Shields

The Adafruit_MotorShield library has the ability to control multiple shields, unlike the older AF_Motor library. First we must create a Motor Shield Controller for each shield, with the assigned address.

```
Adafruit_MotorShield AFMSbot(0x61); // Rightmost jumper closed
Adafruit_MotorShield AFMStop(0x60); // Default address, no jumpers
```

One motor shield is going to be called AFMSbot (bottom shield, so we remember) and one is AFMStop (top shield) so we can keep them apart. When you create the shield object, specify the address you set for it above.

Then we can request the motors connected to each one

```
// On the top shield, connect two steppers, each with 200 steps
Adafruit_StepperMotor *myStepper2 = AFMStop.getStepper(200, 1);
Adafruit_StepperMotor *myStepper3 = AFMStop.getStepper(200, 2);

// On the bottom shield connect a stepper to port M3/M4 with 200 steps
Adafruit_StepperMotor *myStepper1 = AFMSbot.getStepper(200, 2);
// And a DC Motor to port M1
Adafruit_DCMotor *myMotor1 = AFMSbot.getMotor(1);
```

You can request a stepper or DC motor from any port, just be sure to use the right AFMS controller object when you call getMotor or getStepper!

Then, both shields must have begin called, before you use the motors connected

```
AFMSbot.begin(); // Start the bottom shield
AFMStop.begin(); // Start the top shield
```

You can try out this code for yourself by setting up two shields and running the File->Examples->Adafruit_MotorShield->StackingTest example

---

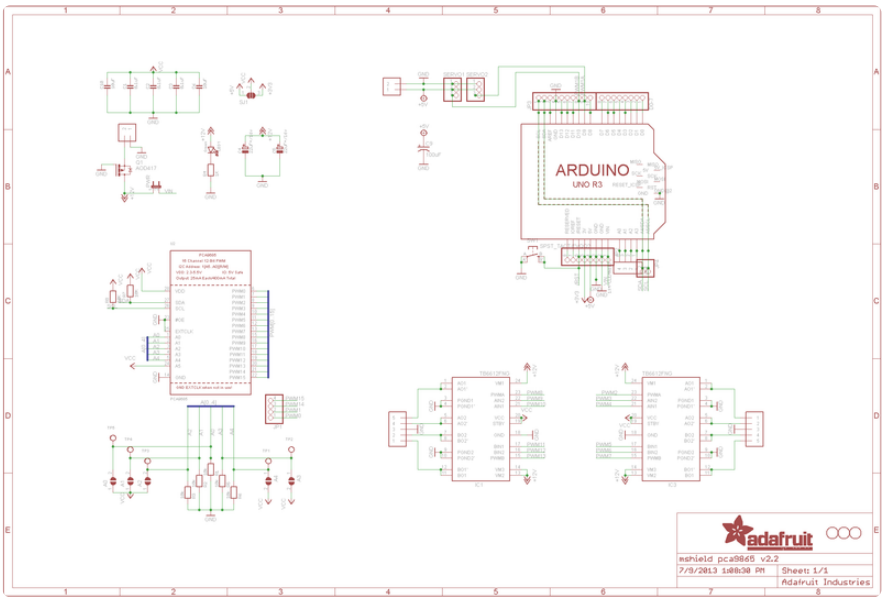
Resources

Motor ideas and tutorials

- Wikipedia has tons of information (https://adafru.it/aOF) on steppers
- Jones on stepper motor types (https://adafru.it/aOH)
- Jason on reverse engineering the stepper wire pinouts (https://adafru.it/aOI)

PCB files are on GitHub (https://adafru.it/DeF)

Schematic, click to embiggen


ARDUINO
UNO R3
adafruit
mshield pca9865 v2.2
7/9/2013 10:30 PM
Sheet: 1/1
Adafruit Industries

Driver: A4988

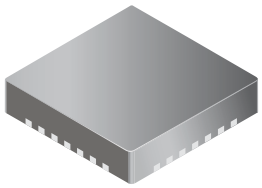
DMOS Microstepping Driver with Translator and Overcurrent Protection

Features and Benefits

- Low $R_{DS(ON)}$ outputs
- Automatic current decay mode detection/selection
- Mixed and Slow current decay modes
- Synchronous rectification for low power dissipation
- Internal UVLO
- Crossover-current protection
- 3.3 and 5 V compatible logic supply
- Thermal shutdown circuitry
- Short-to-ground protection
- Shorted load protection
- Five selectable step modes: full, $^1/_2$, $^1/_4$, $^1/_8$, and $^1/_{16}$

Package:

28-contact QFN
with exposed thermal pad
5 mm × 5 mm × 0.90 mm
(ET package)



Approximate size

Description

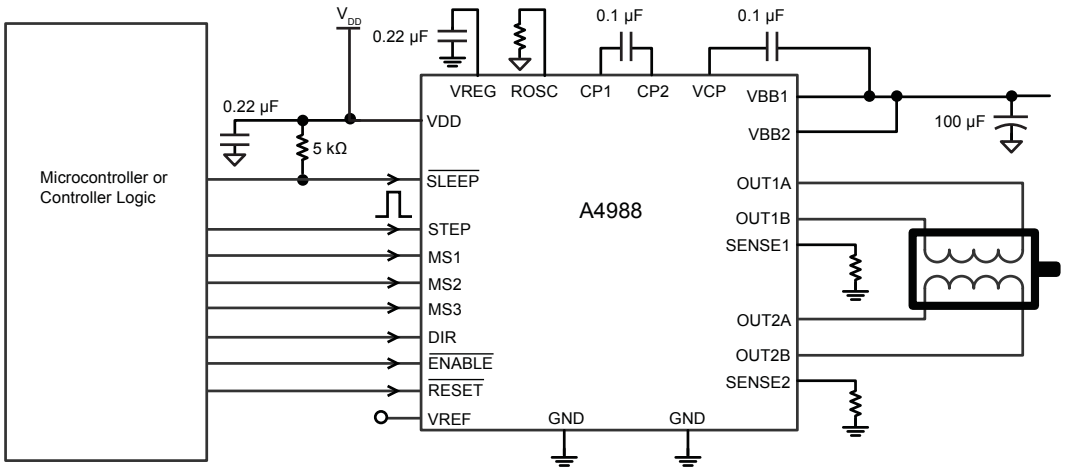
The A4988 is a complete microstepping motor driver with built-in translator for easy operation. It is designed to operate bipolar stepper motors in full-, half-, quarter-, eighth-, and sixteenth-step modes, with an output drive capacity of up to 35 V and ±2 A. The A4988 includes a fixed off-time current regulator which has the ability to operate in Slow or Mixed decay modes.

The translator is the key to the easy implementation of the A4988. Simply inputting one pulse on the STEP input drives the motor one microstep. There are no phase sequence tables, high frequency control lines, or complex interfaces to program. The A4988 interface is an ideal fit for applications where a complex microprocessor is unavailable or is overburdened.

During stepping operation, the chopping control in the A4988 automatically selects the current decay mode, Slow or Mixed. In Mixed decay mode, the device is set initially to a fast decay for a proportion of the fixed off-time, then to a slow decay for the remainder of the off-time. Mixed decay current control results in reduced audible motor noise, increased step accuracy, and reduced power dissipation.

Continued on the next page…

Typical Application Diagram



Description (continued)

Internal synchronous rectification control circuitry is provided to improve power dissipation during PWM operation. Internal circuit protection includes: thermal shutdown with hysteresis, undervoltage lockout (UVLO), and crossover-current protection. Special power-on sequencing is not required.

The A4988 is supplied in a surface mount QFN package (ES), 5 mm × 5 mm, with a nominal overall package height of 0.90 mm and an exposed pad for enhanced thermal dissipation. It is lead (Pb) free (suffix –T), with 100% matte tin plated leadframes.

Selection Guide

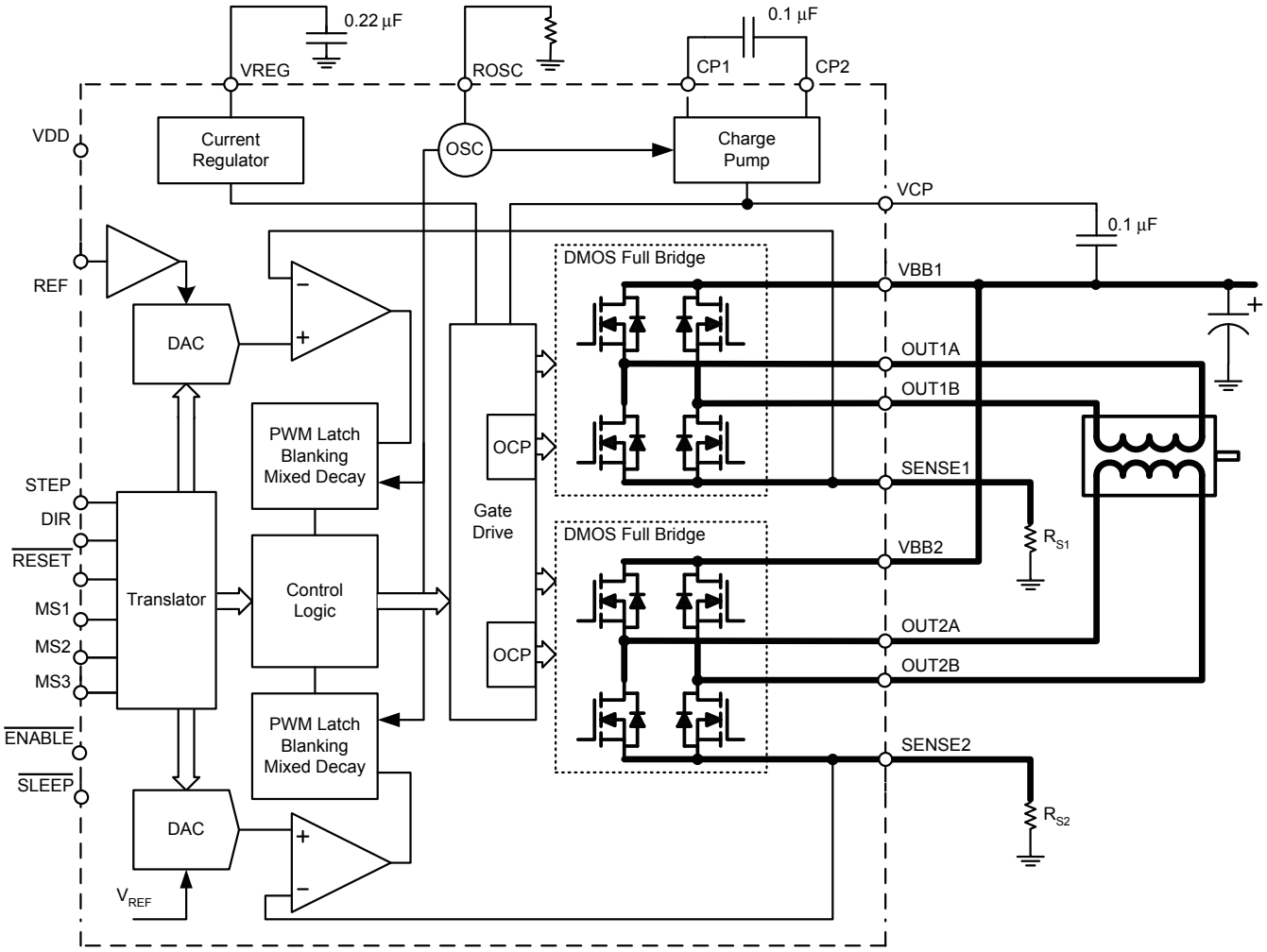
Part Number	Package	Packing
A4988SETTR-T	28-contact QFN with exposed thermal pad	1500 pieces per 7-in. reel

Absolute Maximum Ratings

Characteristic	Symbol	Notes	Rating	Units
Load Supply Voltage	V_{BB}		35	V
Output Current	I_{OUT}		±2	A
Logic Input Voltage	V_{IN}		–0.3 to 5.5	V
Logic Supply Voltage	V_{DD}		–0.3 to 5.5	V
VBBx to OUTx			35	V
Sense Voltage	V_{SENSE}		0.5	V
Reference Voltage	V_{REF}		5.5	V
Operating Ambient Temperature	T_A	Range S	–20 to 85	ºC
Maximum Junction	T_J(max)		150	ºC
Storage Temperature	T_{stg}		–55 to 150	ºC

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

Functional Block Diagram



Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

ELECTRICAL CHARACTERISTICS[1] at T_A = 25°C, V_{BB} = 35 V (unless otherwise noted)

Characteristics	Symbol	Test Conditions	Min.	Typ.[2]	Max.	Units
Output Drivers						
Load Supply Voltage Range	V_{BB}	Operating	8	–	35	V
Logic Supply Voltage Range	V_{DD}	Operating	3.0	–	5.5	V
Output On Resistance	R_{DSON}	Source Driver, I_{OUT} = –1.5 A	–	320	430	mΩ
		Sink Driver, I_{OUT} = 1.5 A	–	320	430	mΩ
Body Diode Forward Voltage	V_F	Source Diode, I_F = –1.5 A	–	–	1.2	V
		Sink Diode, I_F = 1.5 A	–	–	1.2	V
Motor Supply Current	I_{BB}	f_{PWM} < 50 kHz	–	–	4	mA
		Operating, outputs disabled	–	–	2	mA
Logic Supply Current	I_{DD}	f_{PWM} < 50 kHz	–	–	8	mA
		Outputs off	–	–	5	mA
Control Logic						
Logic Input Voltage	$V_{IN(1)}$		V_{DD}×0.7	–	–	V
	$V_{IN(0)}$		–	–	V_{DD}×0.3	V
Logic Input Current	$I_{IN(1)}$	V_{IN} = V_{DD}×0.7	–20	<1.0	20	µA
	$I_{IN(0)}$	V_{IN} = V_{DD}×0.3	–20	<1.0	20	µA
Microstep Select	R_{MS1}	MS1 pin	–	100	–	kΩ
	R_{MS2}	MS2 pin	–	50	–	kΩ
	R_{MS3}	MS3 pin	–	100	–	kΩ
Logic Input Hysteresis	$V_{HYS(IN)}$	As a % of V_{DD}	5	11	19	%
Blank Time	t_{BLANK}		0.7	1	1.3	µs
Fixed Off-Time	t_{OFF}	OSC = VDD or GND	20	30	40	µs
		R_{OSC} = 25 kΩ	23	30	37	µs
Reference Input Voltage Range	V_{REF}		0	–	4	V
Reference Input Current	I_{REF}		–3	0	3	µA
Current Trip-Level Error[3]	err_I	V_{REF} = 2 V, %$I_{TripMAX}$ = 38.27%	–	–	±15	%
		V_{REF} = 2 V, %$I_{TripMAX}$ = 70.71%	–	–	±5	%
		V_{REF} = 2 V, %$I_{TripMAX}$ = 100.00%	–	–	±5	%
Crossover Dead Time	t_{DT}		100	475	800	ns
Protection						
Overcurrent Protection Threshold	I_{OCPST}		2.1	–	–	A
Thermal Shutdown Temperature	T_{TSD}		–	165	–	°C
Thermal Shutdown Hysteresis	T_{TSDHYS}		–	15	–	°C
VDD Undervoltage Lockout	V_{DDUVLO}	V_{DD} rising	2.7	2.8	2.9	V
VDD Undervoltage Hysteresis	$V_{DDUVLOHYS}$		–	90	–	mV

[1]For input and output current specifications, negative current is defined as coming out of (sourcing) the specified device pin.

[2]Typical data are for initial design estimations only, and assume optimum manufacturing and application conditions. Performance may vary for individual units, within the specified maximum and minimum limits.

[3]$V_{ERR} = [(V_{REF}/8) – V_{SENSE}] / (V_{REF}/8)$.

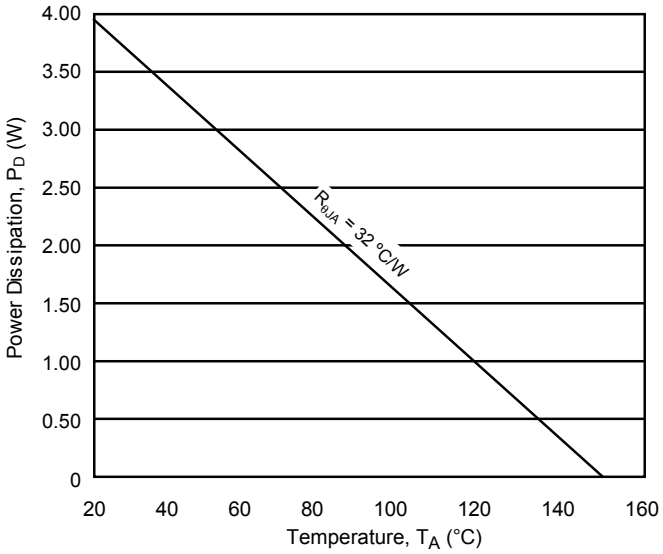
Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

THERMAL CHARACTERISTICS

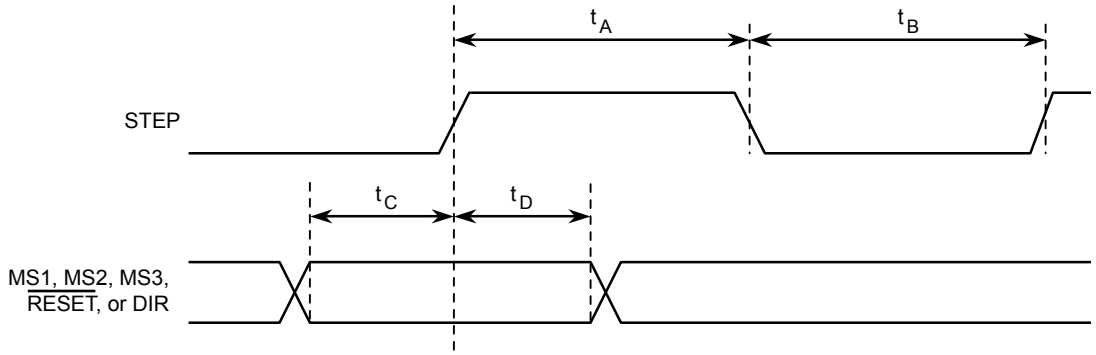
Characteristic	Symbol	Test Conditions*	Value	Units
Package Thermal Resistance	$R_{\theta JA}$	Four-layer PCB, based on JEDEC standard	32	ºC/W

*Additional thermal information available on Allegro Web site.

Power Dissipation versus Ambient Temperature



Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com



Time Duration	Symbol	Typ.	Unit
STEP minimum, HIGH pulse width	t_A	1	µs
STEP minimum, LOW pulse width	t_B	1	µs
Setup time, input change to STEP	t_C	200	ns
Hold time, input change to STEP	t_D	200	ns

Figure 1. Logic Interface Timing Diagram

Table 1. Microstepping Resolution Truth Table

MS1	MS2	MS3	Microstep Resolution	Excitation Mode
L	L	L	Full Step	2 Phase
H	L	L	Half Step	1-2 Phase
L	H	L	Quarter Step	W1-2 Phase
H	H	L	Eighth Step	2W1-2 Phase
H	H	H	Sixteenth Step	4W1-2 Phase

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

Functional Description

Device Operation. The A4988 is a complete microstepping motor driver with a built-in translator for easy operation with minimal control lines. It is designed to operate bipolar stepper motors in full-, half-, quarter-, eighth, and sixteenth-step modes. The currents in each of the two output full-bridges and all of the N-channel DMOS FETs are regulated with fixed off-time PWM (pulse width modulated) control circuitry. At each step, the current for each full-bridge is set by the value of its external current-sense resistor (R_{S1} and R_{S2}), a reference voltage (V_{REF}), and the output voltage of its DAC (which in turn is controlled by the output of the translator).

At power-on or reset, the translator sets the DACs and the phase current polarity to the initial Home state (shown in figures 8 through 12), and the current regulator to Mixed Decay Mode for both phases. When a step command signal occurs on the STEP input, the translator automatically sequences the DACs to the next level and current polarity. (See table 2 for the current-level sequence.) The microstep resolution is set by the combined effect of the MSx inputs, as shown in table 1.

When stepping, if the new output levels of the DACs are lower than their previous output levels, then the decay mode for the active full-bridge is set to Mixed. If the new output levels of the DACs are higher than or equal to their previous levels, then the decay mode for the active full-bridge is set to Slow. This automatic current decay selection improves microstepping performance by reducing the distortion of the current waveform that results from the back EMF of the motor.

Microstep Select (MSx). The microstep resolution is set by the voltage on logic inputs MSx, as shown in table 1. The MS1 and MS3 pins have a 100 kΩ pull-down resistance, and the MS2 pin has a 50 kΩ pull-down resistance. When changing the step mode the change does not take effect until the next STEP rising edge.

If the step mode is changed without a translator reset, and absolute position must be maintained, it is important to change the step mode at a step position that is common to both step modes in order to avoid missing steps. When the device is powered down, or reset due to TSD or an over current event the translator is set to the home position which is by default common to all step modes.

Mixed Decay Operation. The bridge operates in Mixed decay mode, at power-on and reset, and during normal running according to the ROSC configuration and the step sequence, as shown in figures 8 through 12. During Mixed decay, when the trip point is reached, the A4988 initially goes into a fast decay mode for 31.25% of the off-time, t_{OFF}. After that, it switches to Slow decay mode for the remainder of t_{OFF}. A timing diagram for this feature appears on the next page.

Typically, mixed decay is only necessary when the current in the winding is going from a higher value to a lower value as determined by the state of the translator. For most loads automatically-selected mixed decay is convenient because it minimizes ripple when the current is rising and prevents missed steps when the current is falling. For some applications where microstepping at very low speeds is necessary, the lack of back EMF in the winding causes the current to increase in the load quickly, resulting in missed steps. This is shown in figure 2. By pulling the ROSC pin to ground, mixed decay is set to be active 100% of the time, for both rising and falling currents, and prevents missed steps as shown in figure 3. If this is not an issue, it is recommended that automatically-selected mixed decay be used, because it will produce reduced ripple currents. Refer to the Fixed Off-Time section for details.

Low Current Microstepping. Intended for applications where the minimum on-time prevents the output current from regulating to the programmed current level at low current steps. To prevent this, the device can be set to operate in Mixed decay mode on both rising and falling portions of the current waveform. This feature is implemented by shorting the ROSC pin to ground. In this state, the off-time is internally set to 30 µs.

Reset Input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input sets the translator to a predefined Home state (shown in figures 8 through 12), and turns off all of the FET outputs. All STEP inputs are ignored until the $\overline{\text{RESET}}$ input is set to high.

Step Input (STEP). A low-to-high transition on the STEP input sequences the translator and advances the motor one increment. The translator controls the input to the DACs and the direc-

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

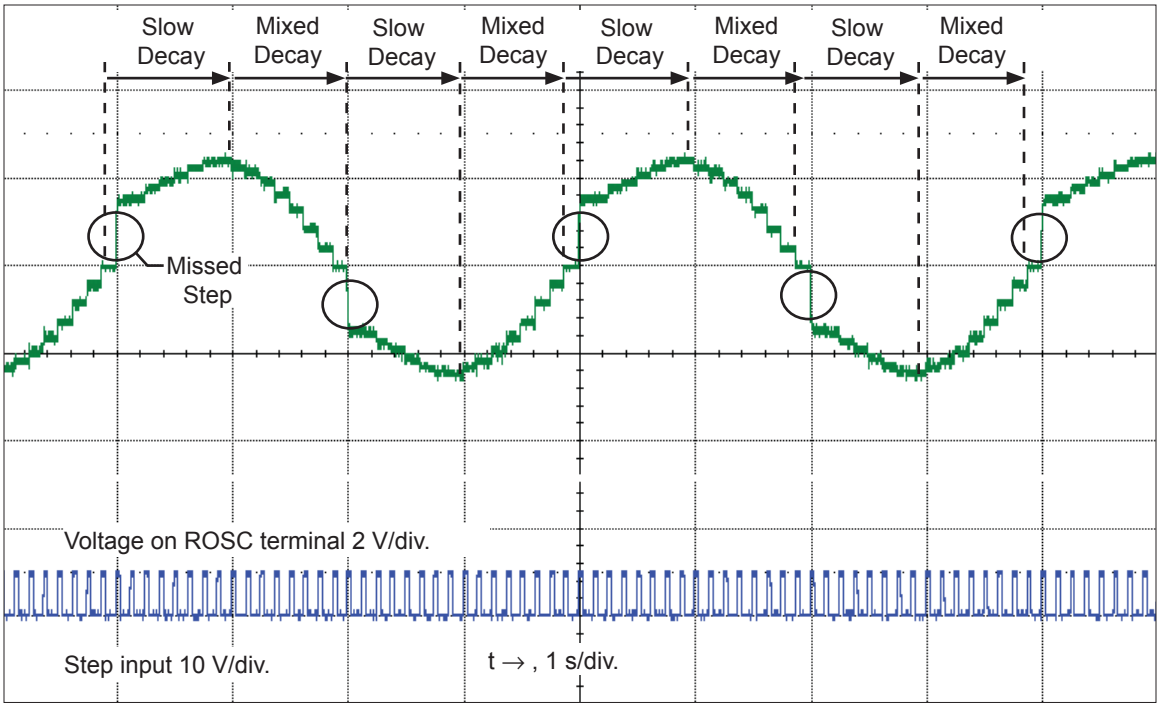


Figure 2. Missed steps in low-speed microstepping

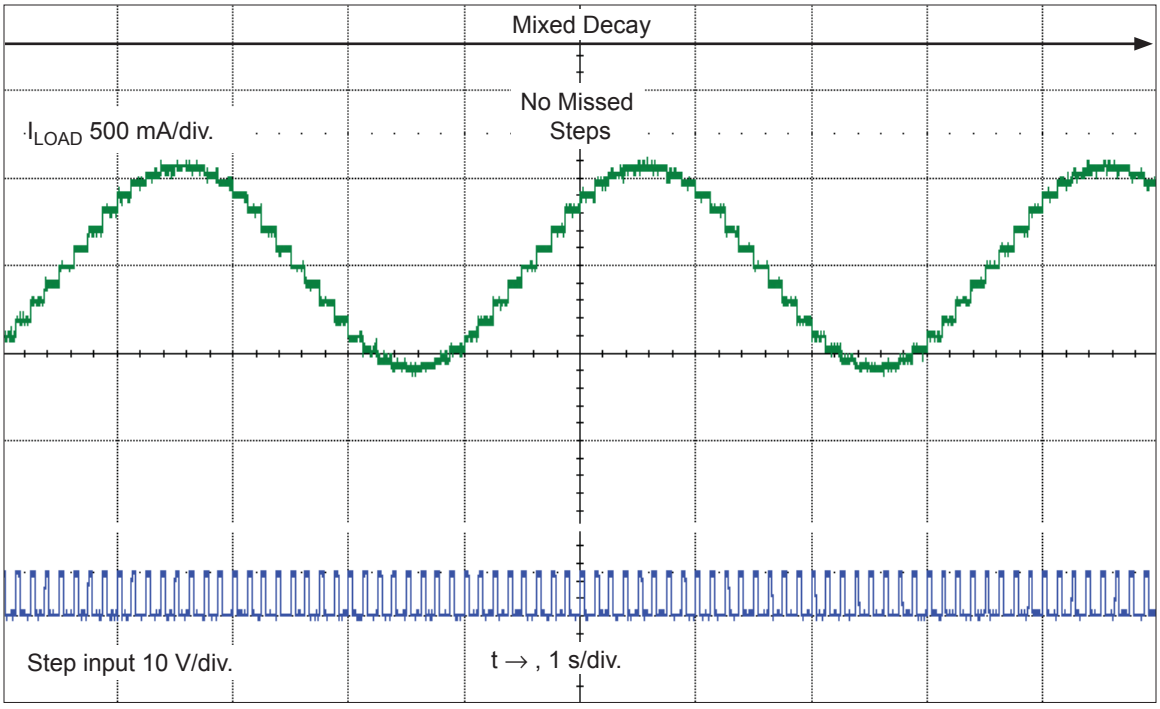


Figure 3. Continuous stepping using automatically-selected mixed stepping (ROSC pin grounded)

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

tion of current flow in each winding. The size of the increment is determined by the combined state of the MSx inputs.

Direction Input (DIR). This determines the direction of rotation of the motor. Changes to this input do not take effect until the next STEP rising edge.

Internal PWM Current Control. Each full-bridge is controlled by a fixed off-time PWM current control circuit that limits the load current to a desired value, I_{TRIP}. Initially, a diagonal pair of source and sink FET outputs are enabled and current flows through the motor winding and the current sense resistor, R_{Sx}. When the voltage across R_{Sx} equals the DAC output voltage, the current sense comparator resets the PWM latch. The latch then turns off the appropriate source driver and initiates a fixed off time decay mode

The maximum value of current limiting is set by the selection of R_{Sx} and the voltage at the VREF pin. The transconductance function is approximated by the maximum value of current limiting, $I_{TripMAX}$ (A), which is set by

$$I_{TripMAX} = V_{REF} / (8 \times R_S)$$

where R_S is the resistance of the sense resistor (Ω) and V_{REF} is the input voltage on the REF pin (V).

The DAC output reduces the V_{REF} output to the current sense comparator in precise steps, such that

$$I_{trip} = (\%I_{TripMAX} / 100) \times I_{TripMAX}$$

(See table 2 for $\%I_{TripMAX}$ at each step.)

It is critical that the maximum rating (0.5 V) on the SENSE1 and SENSE2 pins is not exceeded.

Fixed Off-Time. The internal PWM current control circuitry uses a one-shot circuit to control the duration of time that the DMOS FETs remain off. The off-time, t_{OFF}, is determined by the ROSC terminal. The ROSC terminal has three settings:

- ROSC tied to VDD — off-time internally set to 30 µs, decay mode is automatic Mixed decay except when in full step where decay mode is set to Slow decay
- ROSC tied directly to ground — off-time internally set to 30 µs, current decay is set to Mixed decay for both increasing and decreasing currents, except in full step where decay mode is set to Slow decay. (See Low Current Microstepping section.)
- ROSC through a resistor to ground — off-time is determined by the following formula, the decay mode is automatic Mixed decay for all step modes.

$$t_{OFF} \approx R_{OSC} / 825$$

Where t_{OFF} is in µs.

Blanking. This function blanks the output of the current sense comparators when the outputs are switched by the internal current control circuitry. The comparator outputs are blanked to prevent false overcurrent detection due to reverse recovery currents of the clamp diodes, and switching transients related to the capacitance of the load. The blank time, t_{BLANK} (µs), is approximately

$$t_{BLANK} \approx 1 \text{ µs}$$

Shorted-Load and Short-to-Ground Protection. If the motor leads are shorted together, or if one of the leads is shorted to ground, the driver will protect itself by sensing the overcurrent event and disabling the driver that is shorted, protecting the device from damage. In the case of a short-to-ground, the device will remain disabled (latched) until the $\overline{\text{SLEEP}}$ input goes high or VDD power is removed. A short-to-ground overcurrent event is shown in figure 4.

When the two outputs are shorted together, the current path is through the sense resistor. After the blanking time (≈1 µs) expires, the sense resistor voltage is exceeding its trip value, due to the overcurrent condition that exists. This causes the driver to go into a fixed off-time cycle. After the fixed off-time expires the driver turns on again and the process repeats. In this condition the driver is completely protected against overcurrent events, but the short is repetitive with a period equal to the fixed off-time of the driver. This condition is shown in figure 5.

During a shorted load event it is normal to observe both a positive and negative current spike as shown in figure 3, due to the direction change implemented by the Mixed decay feature. This is shown in figure 6. In both instances the overcurrent circuitry is protecting the driver and prevents damage to the device.

Charge Pump (CP1 and CP2). The charge pump is used to generate a gate supply greater than that of VBB for driving the source-side FET gates. A 0.1 µF ceramic capacitor, should be connected between CP1 and CP2. In addition, a 0.1 µF ceramic capacitor is required between VCP and VBB, to act as a reservoir for operating the high-side FET gates.

Capacitor values should be Class 2 dielectric ±15% maximum, or tolerance R, according to EIA (Electronic Industries Alliance) specifications.

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

V_{REG} (VREG). This internally-generated voltage is used to operate the sink-side FET outputs. The VREG pin must be decoupled with a 0.22 µF ceramic capacitor to ground. V_{REG} is internally monitored. In the case of a fault condition, the FET outputs of the A4988 are disabled.

Capacitor values should be Class 2 dielectric ±15% maximum, or tolerance R, according to EIA (Electronic Industries Alliance) specifications.

Enable Input ($\overline{\text{ENABLE}}$). This input turns on or off all of the FET outputs. When set to a logic high, the outputs are disabled. When set to a logic low, the internal control enables the outputs as required. The translator inputs STEP, DIR, and MSx, as well as the internal sequencing logic, all remain active, independent of the $\overline{\text{ENABLE}}$ input state.

Shutdown. In the event of a fault, overtemperature (excess T_J) or an undervoltage (on VCP), the FET outputs of the A4988 are disabled until the fault condition is removed. At power-on, the UVLO (undervoltage lockout) circuit disables the FET outputs and resets the translator to the Home state.

Sleep Mode ($\overline{\text{SLEEP}}$). To minimize power consumption when the motor is not in use, this input disables much of the internal circuitry including the output FETs, current regulator, and charge pump. A logic low on the $\overline{\text{SLEEP}}$ pin puts the A4988 into Sleep mode. A logic high allows normal operation, as well as start-up (at which time the A4988 drives the motor to the Home microstep position). When emerging from Sleep mode, in order to allow the charge pump to stabilize, provide a delay of 1 ms before issuing a Step command.

Mixed Decay Operation. The bridge operates in Mixed Decay mode, depending on the step sequence, as shown in figures 8 through 12. As the trip point is reached, the A4988 initially goes into a fast decay mode for 31.25% of the off-time, t_{OFF}. After that, it switches to Slow Decay mode for the remainder of t_{OFF}. A timing diagram for this feature appears in figure 7.

Synchronous Rectification. When a PWM-off cycle is triggered by an internal fixed-off time cycle, load current recirculates according to the decay mode selected by the control logic. This synchronous rectification feature turns on the appropriate FETs during current decay, and effectively shorts out the body diodes with the low FET $R_{DS(ON)}$. This reduces power dissipation significantly, and can eliminate the need for external Schottky diodes in many applications. Synchronous rectification turns off when the load current approaches zero (0 A), preventing reversal of the load current.

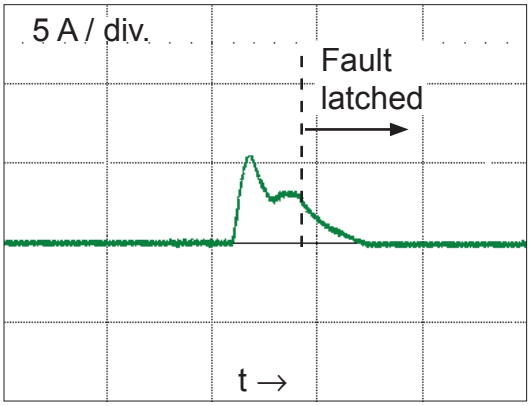


Figure 4. Short-to-ground event

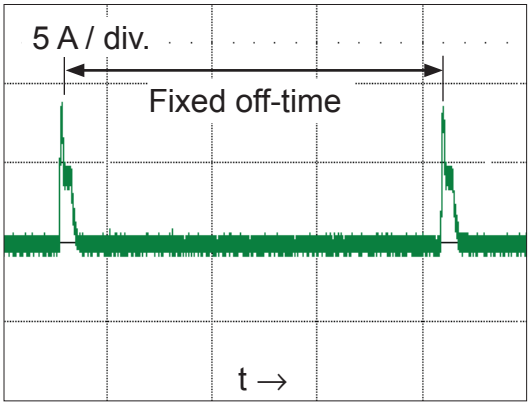


Figure 5. Shorted load (OUTxA → OUTxB) in Slow decay mode

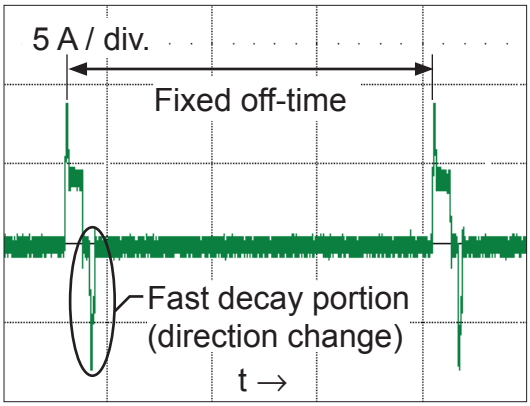
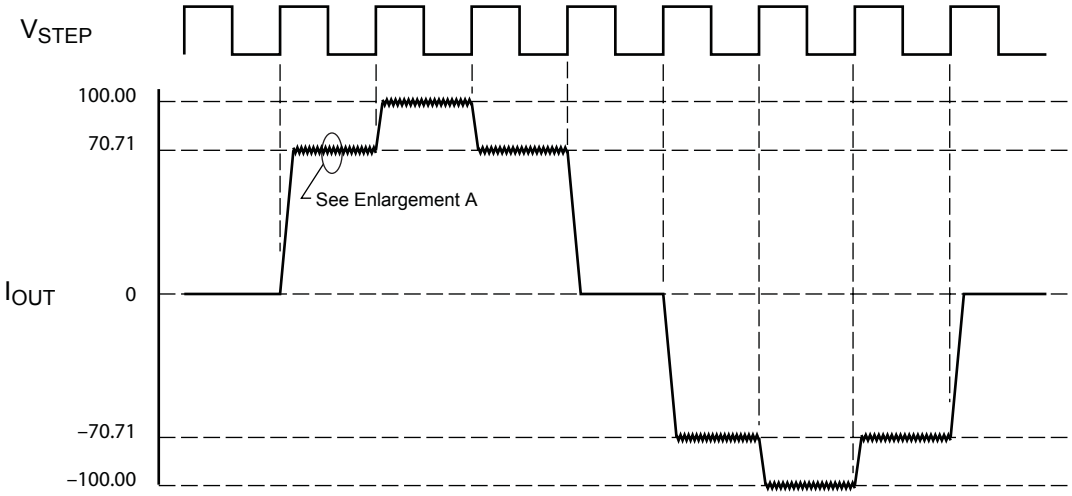
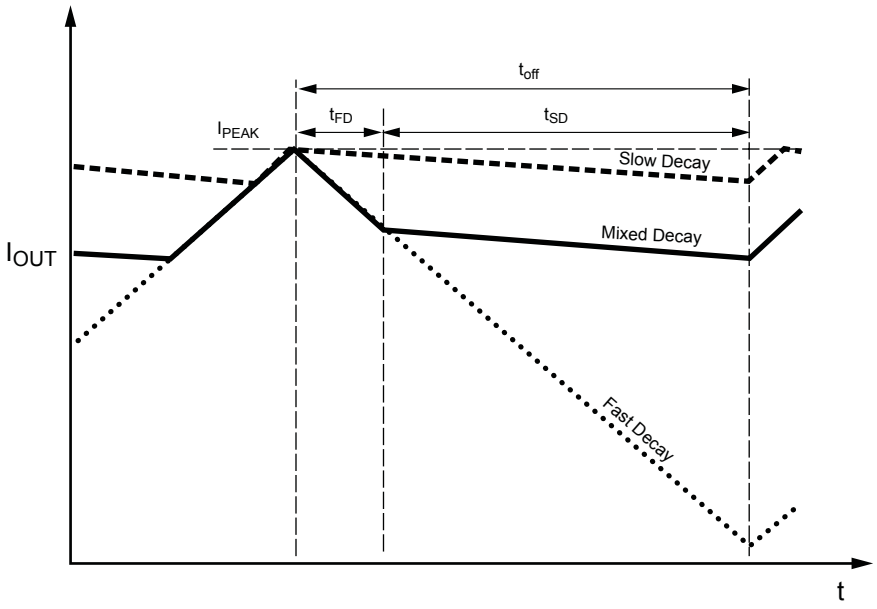


Figure 6. Shorted load (OUTxA → OUTxB) in Mixed decay mode

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com



Enlargement A



Symbol	Characteristic
t_{off}	Device fixed off-time
I_{PEAK}	Maximum output current
t_{SD}	Slow decay interval
t_{FD}	Fast decay interval
I_{OUT}	Device output current

Figure 7. Current Decay Modes Timing Chart

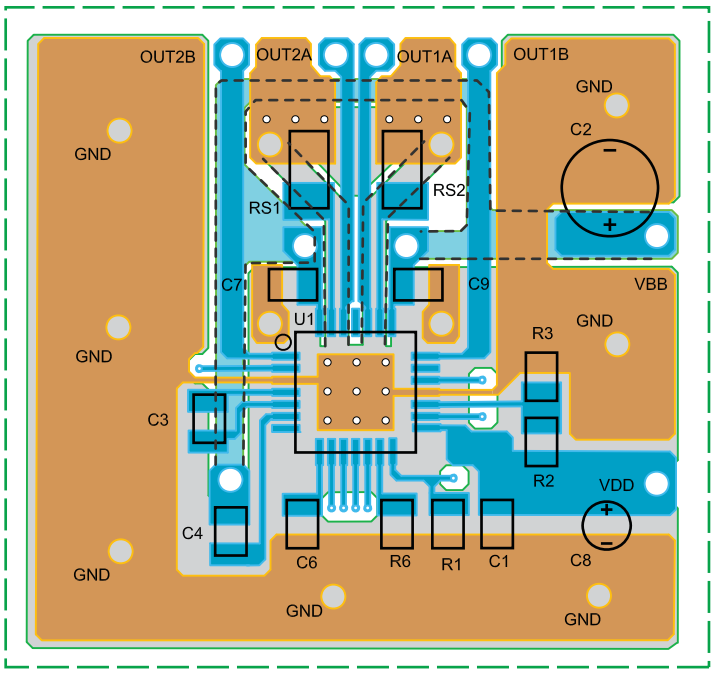
Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

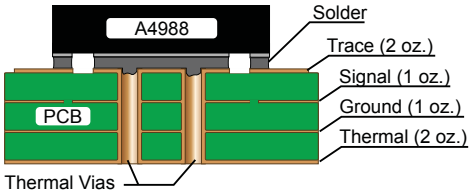
Application Layout

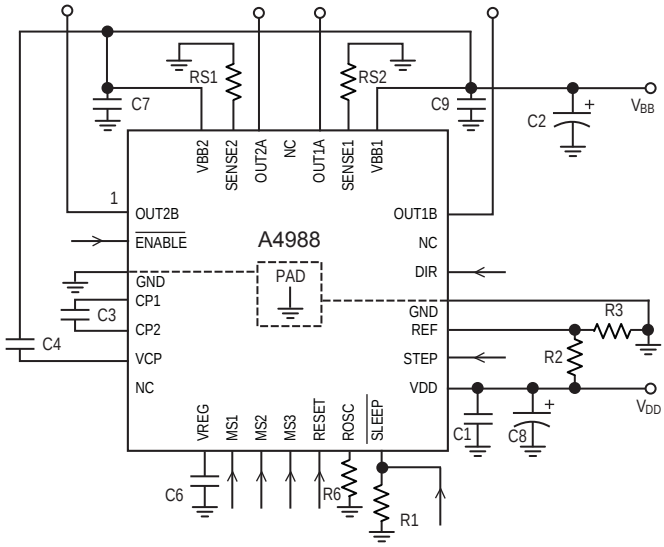
Layout. The printed circuit board should use a heavy ground-plane. For optimum electrical and thermal performance, the A4988 must be soldered directly onto the board. Pins 6, 7, 18, and 19 are internally fused, which provides a path for enhanced thermal dissipation. Theses pins should be soldered directly to an exposed surface on the PCB that connects to thermal vias are used to transfer heat to other layers of the PCB.

In order to minimize the effects of ground bounce and offset issues, it is important to have a low impedance single-point ground, known as a *star ground*, located very close to the device. By making the connection between the pad and the ground plane directly under the A4988, that area becomes an ideal location for a star ground point. A low impedance ground will prevent ground bounce during high current operation and ensure that the supply voltage remains stable at the input terminal.

The two input capacitors should be placed in parallel, and as close to the device supply pins as possible. The ceramic capacitor (CIN1) should be closer to the pins than the bulk capacitor (CIN2). This is necessary because the ceramic capacitor will be responsible for delivering the high frequency current components. The sense resistors, RSx , should have a very low impedance path to ground, because they must carry a large current while supporting very accurate voltage measurements by the current sense comparators. Long ground traces will cause additional voltage drops, adversely affecting the ability of the comparators to accurately measure the current in the windings. The SENSEx pins have very short traces to the RSx resistors and very thick, low impedance traces directly to the star ground underneath the device. If possible, there should be no other components on the sense circuits.

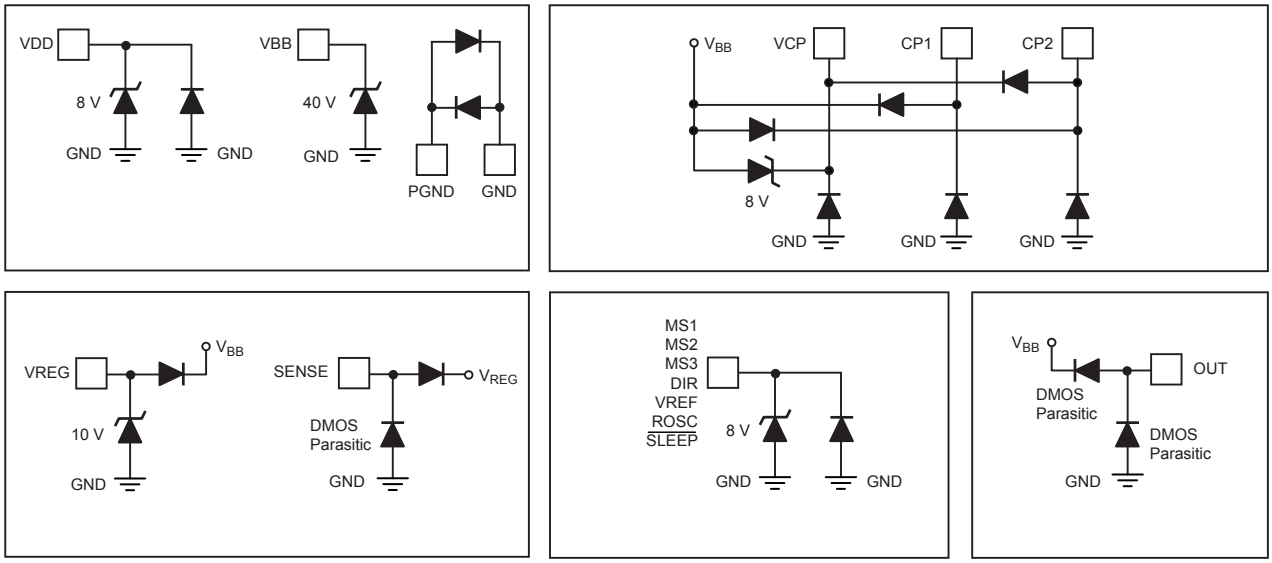






Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

Pin Circuit Diagrams



Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

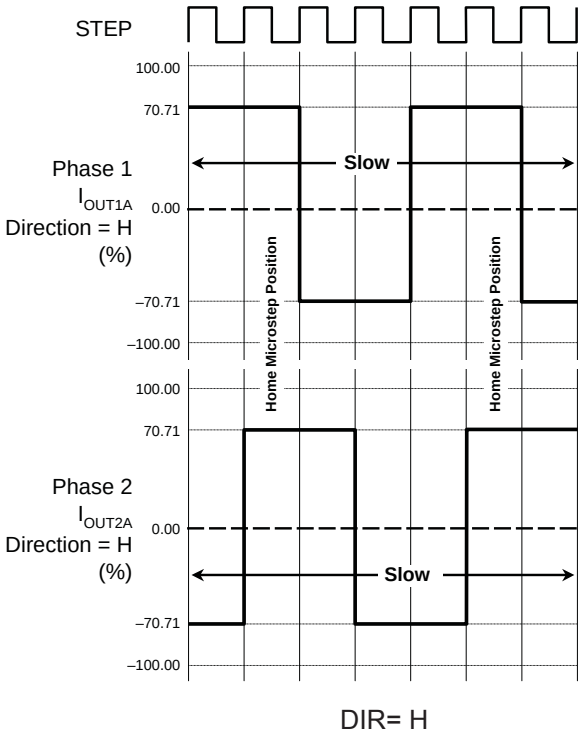


Figure 8. Decay Mode for Full-Step Increments

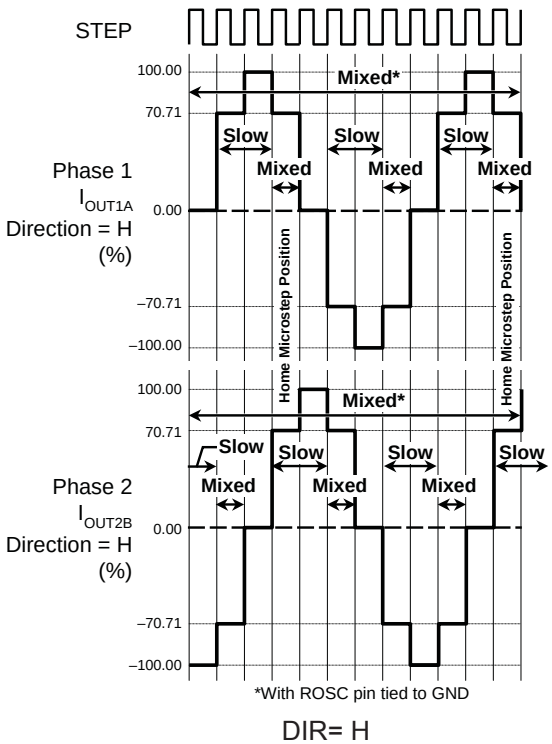


Figure 9. Decay Modes for Half-Step Increments

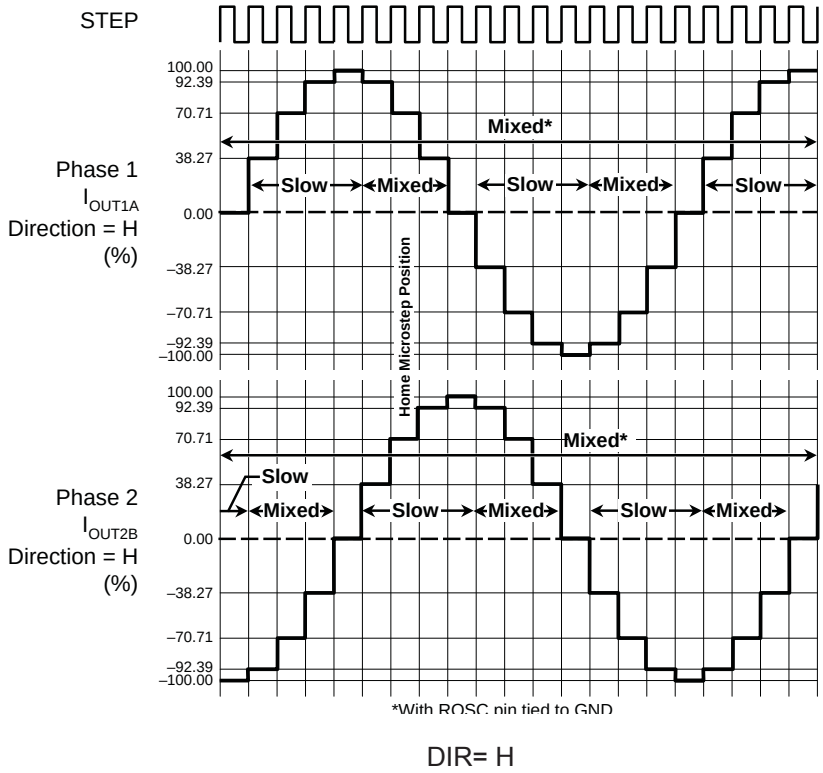


Figure 10. Decay Modes for Quarter-Step Increments

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

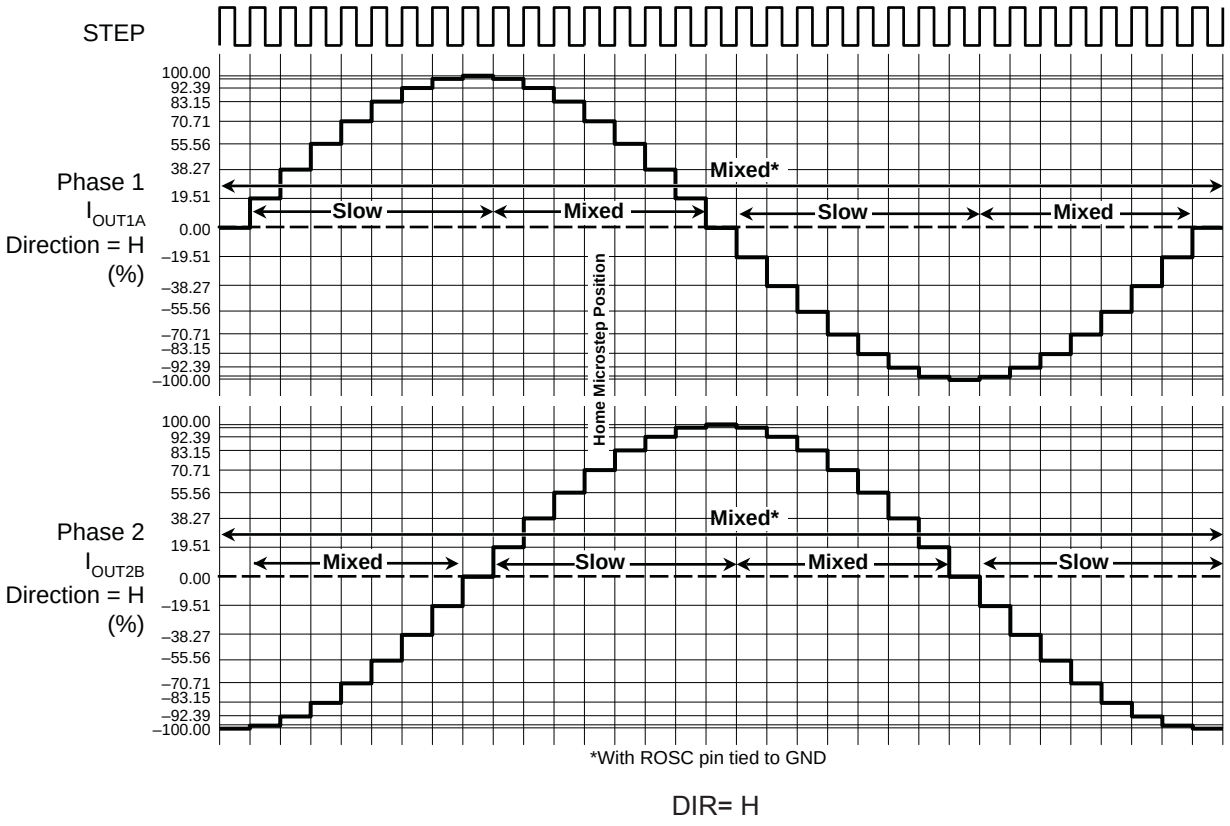


Figure 11. Decay Modes for Eighth-Step Increments

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

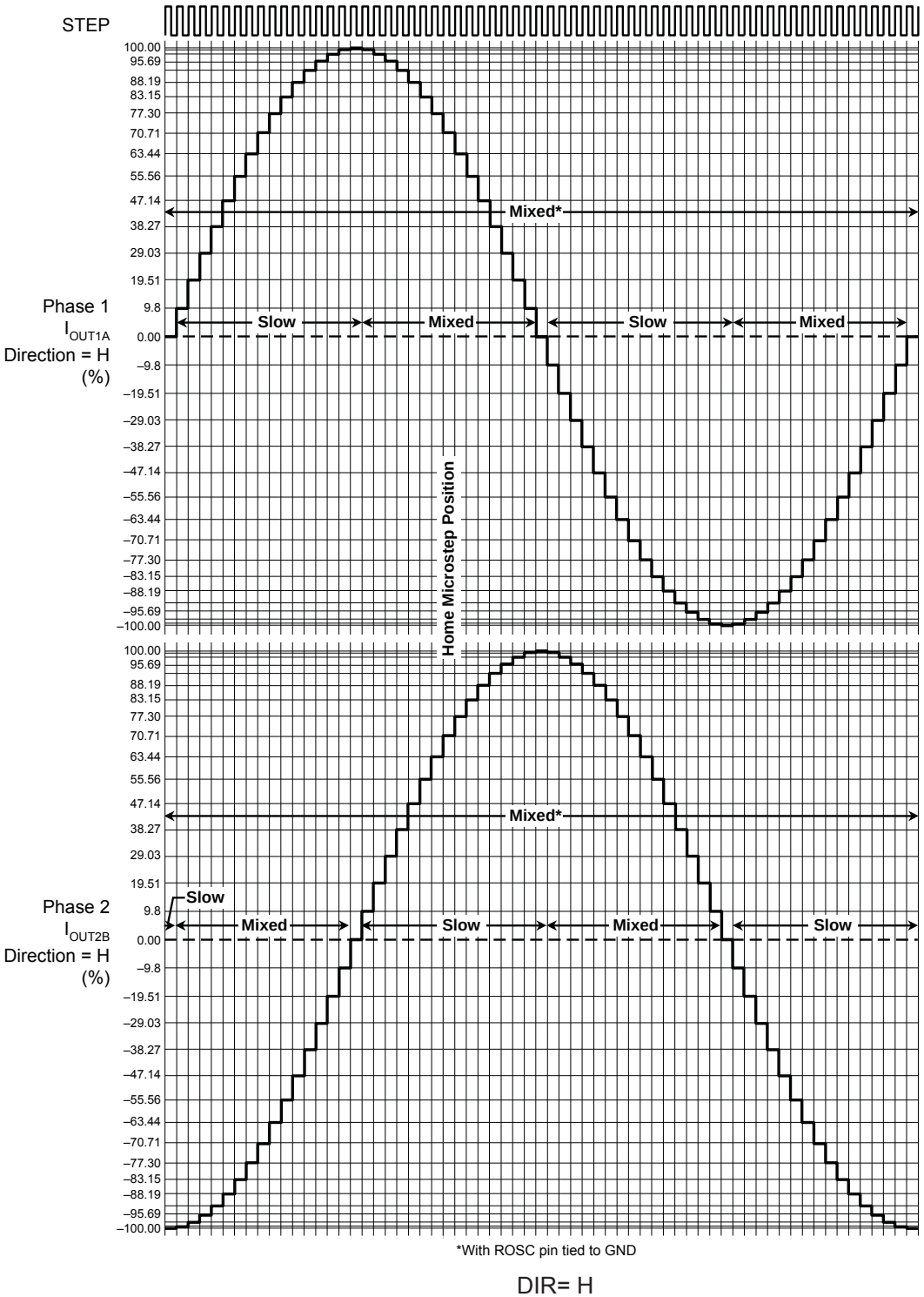


Figure 12. Decay Modes for Sixteenth-Step Increments

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

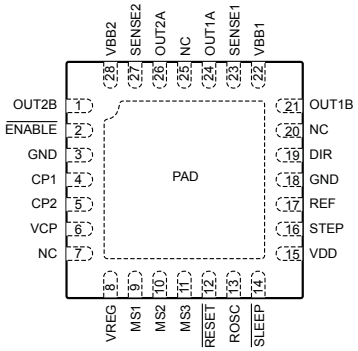
Table 2. Step Sequencing Settings

Home microstep position at Step Angle 45º; DIR = H

Full Step #	Half Step #	1/4 Step #	1/8 Step #	1/16 Step #	Phase 1 Current [% $I_{tripMax}$] (%)	Phase 2 Current [% $I_{tripMax}$] (%)	Step Angle (º)	Full Step #	Half Step #	1/4 Step #	1/8 Step #	1/16 Step #	Phase 1 Current [% $I_{tripMax}$] (%)	Phase 2 Current [% $I_{tripMax}$] (%)	Step Angle (º)
	1	1	2	1	100.00	0.00	0.0		5	9	17	33	–100.00	0.00	180.0
				2	99.52	9.80	5.6					34	–99.52	–9.80	185.6
			2	3	98.08	19.51	11.3				18	35	–98.08	–19.51	191.3
				4	95.69	29.03	16.9					36	–95.69	–29.03	196.9
		2	3	5	92.39	38.27	22.5			10	19	37	–92.39	–38.27	202.5
				6	88.19	47.14	28.1					38	–88.19	–47.14	208.1
			4	7	83.15	55.56	33.8				20	39	–83.15	–55.56	213.8
				8	77.30	63.44	39.4					40	–77.30	–63.44	219.4
1	2	3	5	9	70.71	70.71	45.0	3	6	11	21	41	–70.71	–70.71	225.0
				10	63.44	77.30	50.6					42	–63.44	–77.30	230.6
			6	11	55.56	83.15	56.3				22	43	–55.56	–83.15	236.3
				12	47.14	88.19	61.9					44	–47.14	–88.19	241.9
		4	7	13	38.27	92.39	67.5			12	23	45	–38.27	–92.39	247.5
				14	29.03	95.69	73.1					46	–29.03	–95.69	253.1
			8	15	19.51	98.08	78.8				24	47	–19.51	–98.08	258.8
				16	9.80	99.52	84.4					48	–9.80	–99.52	264.4
	3	5	9	17	0.00	100.00	90.0		7	13	25	49	0.00	–100.00	270.0
				18	–9.80	99.52	95.6					50	9.80	–99.52	275.6
			10	19	–19.51	98.08	101.3				26	51	19.51	–98.08	281.3
				20	–29.03	95.69	106.9					52	29.03	–95.69	286.9
		6	11	21	–38.27	92.39	112.5			14	27	53	38.27	–92.39	292.5
				22	–47.14	88.19	118.1					54	47.14	–88.19	298.1
			12	23	–55.56	83.15	123.8				28	55	55.56	–83.15	303.8
				24	–63.44	77.30	129.4					56	63.44	–77.30	309.4
2	4	7	13	25	–70.71	70.71	135.0	4	8	15	29	57	70.71	–70.71	315.0
				26	–77.30	63.44	140.6					58	77.30	–63.44	320.6
			14	27	–83.15	55.56	146.3				30	59	83.15	–55.56	326.3
				28	–88.19	47.14	151.9					60	88.19	–47.14	331.9
		8	15	29	–92.39	38.27	157.5			16	31	61	92.39	–38.27	337.5
				30	–95.69	29.03	163.1					62	95.69	–29.03	343.1
			16	31	–98.08	19.51	168.8				32	63	98.08	–19.51	348.8
				32	–99.52	9.80	174.4					64	99.52	–9.80	354.4

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

Pin-out Diagram



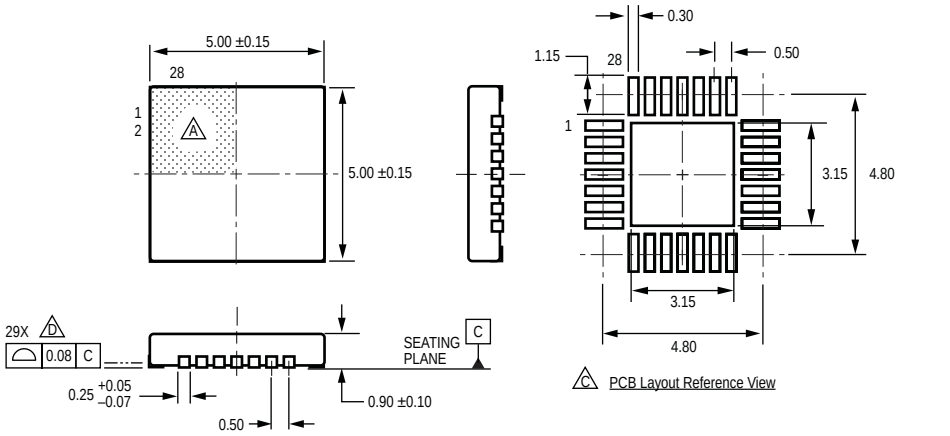
Terminal List Table

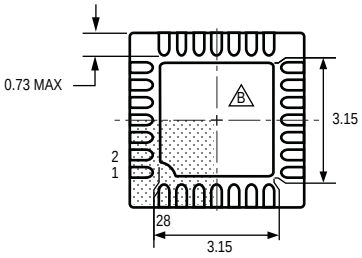
Name	Number	Description
CP1	4	Charge pump capacitor terminal
CP2	5	Charge pump capacitor terminal
VCP	6	Reservoir capacitor terminal
VREG	8	Regulator decoupling terminal
MS1	9	Logic input
MS2	10	Logic input
MS3	11	Logic input
$\overline{\text{RESET}}$	12	Logic input
ROSC	13	Timing set
$\overline{\text{SLEEP}}$	14	Logic input
VDD	15	Logic supply
STEP	16	Logic input
REF	17	G_m reference voltage input
GND	3, 18	Ground*
DIR	19	Logic input
OUT1B	21	DMOS Full Bridge 1 Output B
VBB1	22	Load supply
SENSE1	23	Sense resistor terminal for Bridge 1
OUT1A	24	DMOS Full Bridge 1 Output A
OUT2A	26	DMOS Full Bridge 2 Output A
SENSE2	27	Sense resistor terminal for Bridge 2
VBB2	28	Load supply
OUT2B	1	DMOS Full Bridge 2 Output B
$\overline{\text{ENABLE}}$	2	Logic input
NC	7, 20, 25	No connection
PAD	–	Exposed pad for enhanced thermal dissipation*

*The GND pins must be tied together externally by connecting to the PAD ground plane under the device.

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

ET Package, 28-Pin QFN with Exposed Thermal Pad





For Reference Only; not for tooling use
(reference JEDEC MO-220VHHD-1)
Dimensions in millimeters
Exact case and lead configuration at supplier discretion within limits shown

A Terminal #1 mark area

B Exposed thermal pad (reference only, terminal #1 identifier appearance at supplier discretion)

C Reference land pattern layout (reference IPC7351 QFN50P500X500X100-29V1M);
All pads a minimum of 0.20 mm from all adjacent pads; adjust as necessary to meet application process requirements and PCB layout tolerances; when mounting on a multilayer PCB, thermal vias at the exposed thermal pad land can improve thermal dissipation (reference EIA/JEDEC Standard JESD51-5)

D Coplanarity includes exposed thermal pad and terminals

For the latest version of this document, visit our website:
www.allegromicro.com

Allegro MicroSystems, Inc.
115 Northeast Cutoff
Worcester, Massachusetts 01615-0036 U.S.A.
1.508.853.5000; www.allegromicro.com

Motor: Usongshine US-117HS4401S

Data Specs

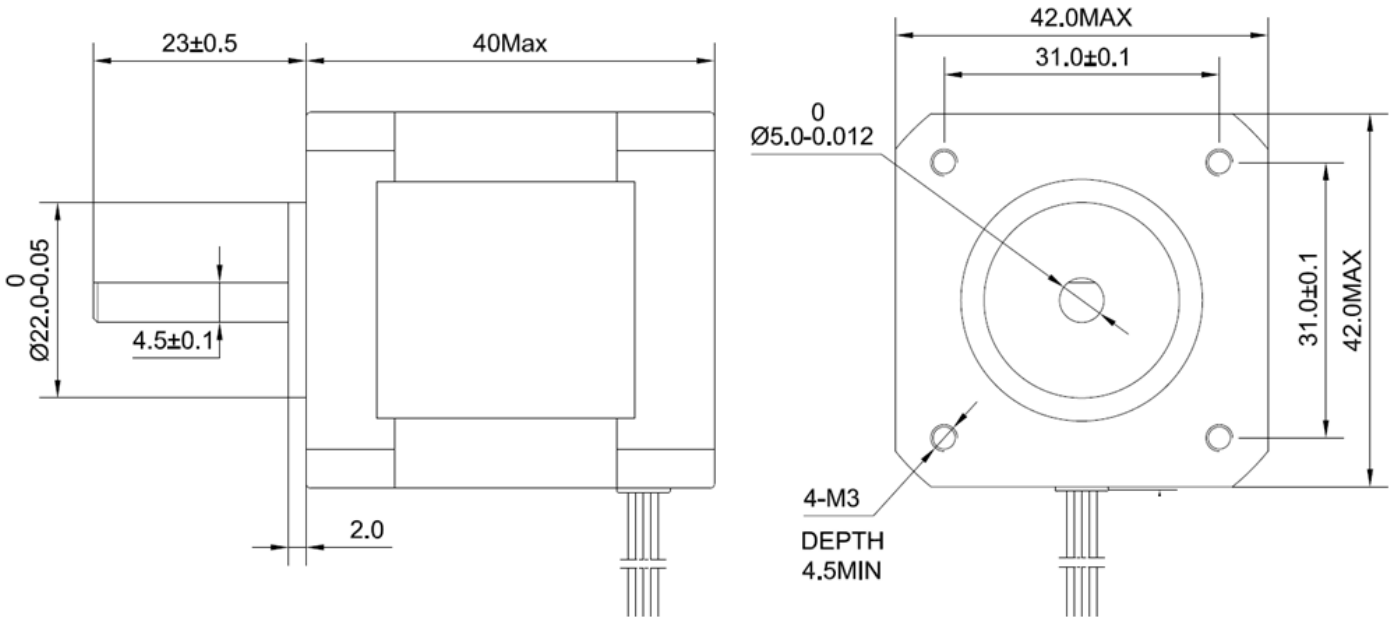
17HS4401S 1.7A Torque:43N.cm Stepper Motor

A stepper motor to satisfy all your 3D-Printer, robotics, Linear Motion projects needs! This 4-wire bipolar stepper has 1.8° per step for smooth motion and a nice holding torque. The motor was specified to have a max current of 1.7A/phase so that it could be driven easily with common motor shield for Arduino (or other motor driver) and a wall adapter or lead-acid battery. The motors are supplied with a 50cm long power cable with a 4-pin Harwin female connector already fitted - ready to plug and print!

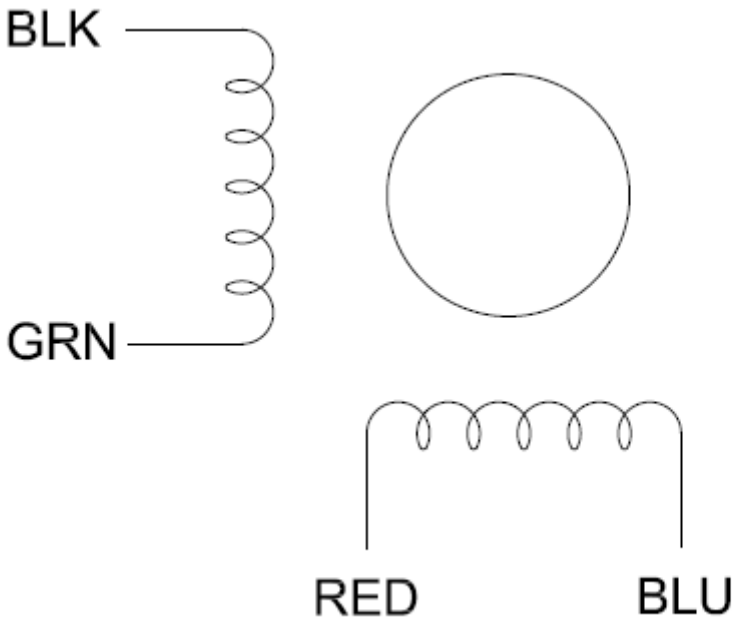
Brief Data:

- Nema17 Bipolar.
- Number of Phase: 2.
- Step Angle: 1.8°.
- Phase Voltage: 2.6Vdc.
- Phase Current: 1.7A.
- Resistance/Phase: 1.5Ω ±10%.
- Inductance: 2.8mH ±20% (1KHz).
- Number of Wire: 4 (100cm Length).
- Holding Torque: 43Ncm.
- Shaft Diameter: Ø5mm.
- Motor Length: 40mm.
- Rotor Inertia: 54gcm^2.
- Temperature rise: 80°C Max.
- Insulation Class: B.
- Dielectric Strength: 500VAC/1-minute.
- Mass: 280g.

Mechanical Dimensions:



Connection:



Nema-17 Stepper Motor Body Length Comparison:

Servo-motor: MG995

31150-MP

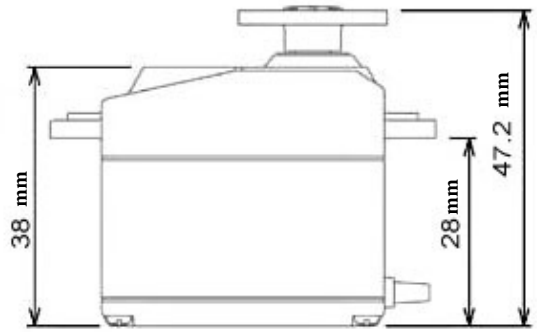
MG995 High Speed Servo Actuator

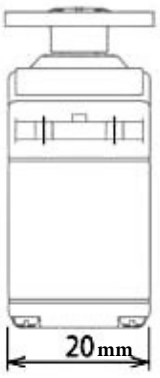
The unit comes complete with color coded 30cm wire leads with a 3 X 1 pin 0.1" Pitch type female header connector that matches most receivers, including Futaba, JR, GWS, Cirrus, Blue Bird, Blue Arrow, Corona, Berg, Spektrum and Hitec.

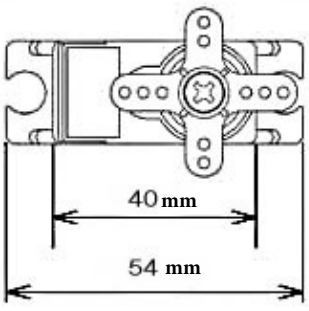
This high-speed servo actuator is not code dependant; You can use any servo code, hardware or library to control them. The MG995 Actuator includes arms and hardware to get started.

Specifications

- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 8.5 kgf·cm (4.8 V), 10 kgf·cm (6 V)
- Rotation Angle: 120deg. (+- 60 from center)
- Operating speed: 0.2 s/60º (4.8 V), 0.16 s/60º (6 V)
- Operating voltage: 4.8 V to 7.2 V
- Dead band width: 5 µs
- Stable and shock proof double ball bearing design
- Metal Gears for longer life
- Temperature range: 0 ºC – 55 ºC

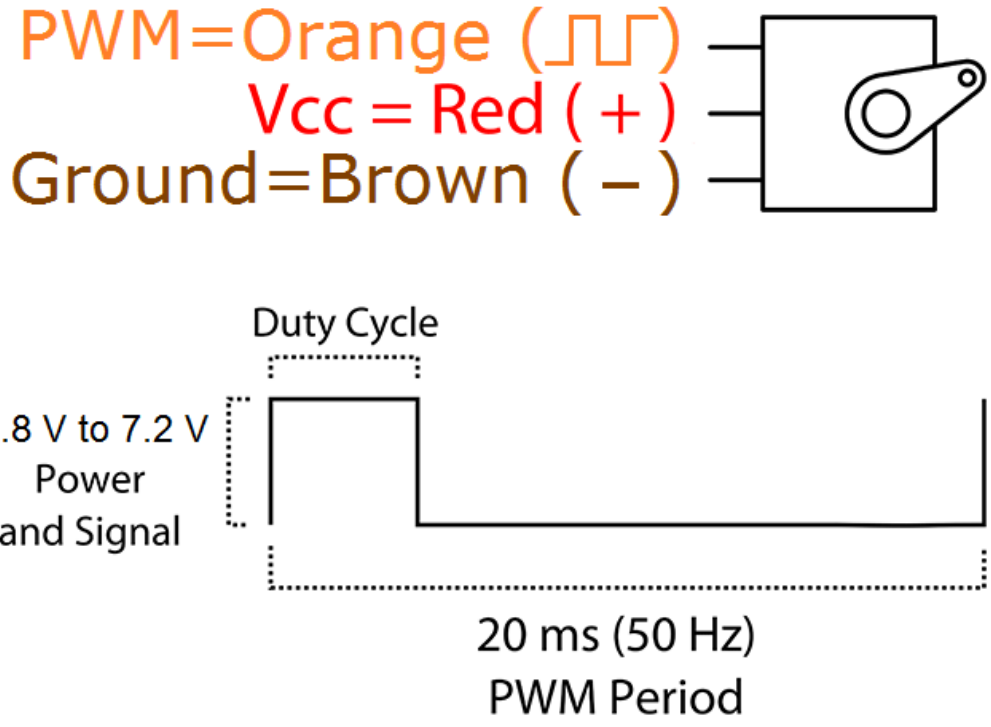






31150-MP

MG995 High Speed Servo Actuator



MARLIN P. JONES & ASSOC., INC.
P.O. Box 530400 Lake Park, Fl 33403
800-652-6733 FAX 561-844-8764
WWW.MPJA.COM

Fi de carrera: MICRO SWITCH ZM

Honeywell

MICRO SWITCH™ Standard Subminiature Snap-Action Z Series

Snap-Action Switches

DESCRIPTION

The industry-defining name in snap-action switches, Honeywell MICRO SWITCH™ standard subminiatures are designed for repeatability and enhanced product life. The MICRO SWITCH™ Z Series combines small size and light weight with ample electrical capacity, low cost, and enhanced life.

The MICRO SWITCH™ Z Series consists of six product families with unique features that can drop right into an application.

These reliable and rugged switches offer a variety of actuators, terminations, circuitry configurations, electrical ratings, contact materials, operating characteristics, and sealing allows them to be utilized in numerous potential applications.

Carefully manufactured and thoroughly inspected, the MICRO SWITCH™ Z Series standard subminiatures are a great value for applications requiring sensing presence or absence of an object.

FEATURES

- Small size and light weight switches lend themselves to numerous potential applications
- Choice of low energy or power-duty electrical ratings allow the switch to be specified in more types of applications
- Broad range of amp ratings (from 0.1 A to 10.1 A)
- Watertight IP67 sealing available on some listings allows the switch to be used where sealing and presence/absence detection is required
- UL/CSA, cUL, ENEC, and CE approvals

POTENTIAL APPLICATIONS

- **Industrial:** Appliances, communication equipment, computers, electromechanical timers, mechanical cam assemblies (timers), office equipment, electric tools, HVAC wall controls, instrumentation, valves, vending machines
- **Transportation:** Automotive, truck, and boat wire harnesses; sub-assemblies for convertible roofs; lock modules for tail-gate/trunk; tank and hood latch detection
- **Medical:** Medical and hospital beds, foot pedal controls, and chair lifts
- Applications where a pre-wired sealed on/off switch is required

MICRO SWITCH™ Standard Subminiature Snap-Action Z Series

SPECIFICATIONS

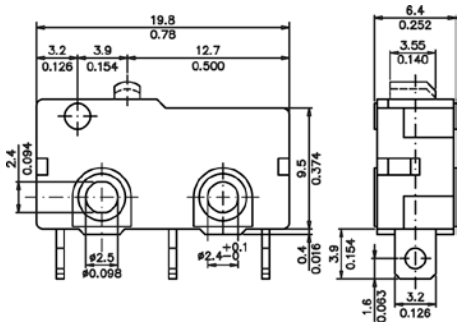
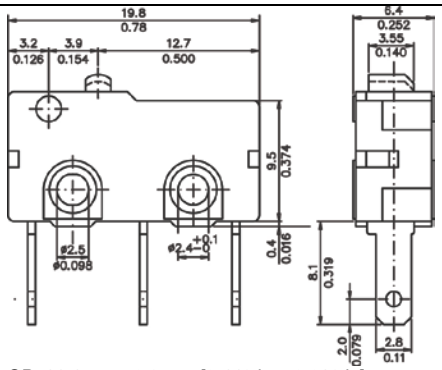
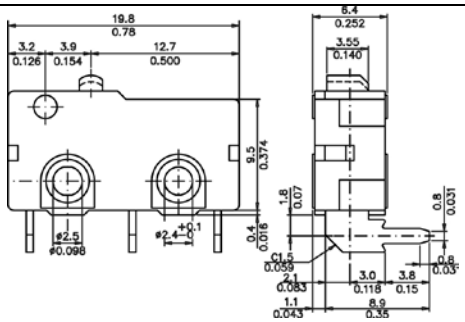
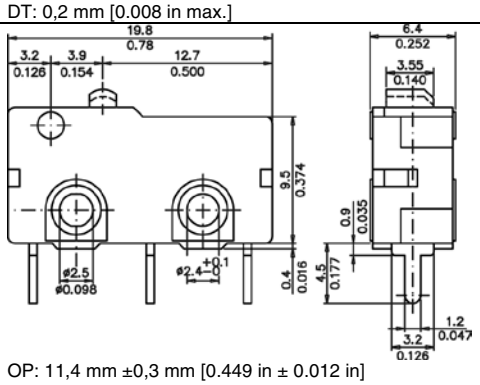
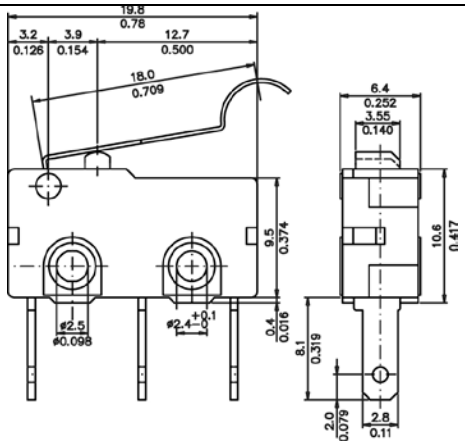
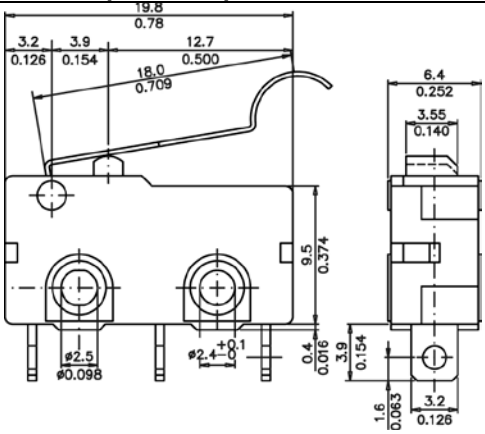
			
SERIES	**ZM (coil internal spring)**	**ZM1 (flat internal spring)**	**ZV (coil spring)**
Differentiator	Integral lever, no ENEC, and an internal coil spring	Integral lever, ENEC, and a flat internal spring	Snap-on lever, ENEC, and coil spring
Use	Use when ENEC is not required and the lever needs to be better secured to the switch	Used when added forces of a flat snap spring, ENEC, and a secured lever are required	Use when ENEC and a snap-on lever are required
Potential applications	alarms, computers, food processors, gas detectors, humidifiers, joysticks, money sorters, water pumps	air conditioners, consumer electronics, gas detectors, humidifiers, telephones, time recorders, toys	air conditioners, computers, consumer appliances, gas detectors, joysticks, money sorters, telephones, toys
Ampere rating	0.1 A, 5 A, 10.1 A	0.1 A, 3 A, 6 A, 10.1 A	0.1 A, 6 A, 10.1 A
Circuitry	SPDT, SPNO	SPDT, SPNO, SPNC	SPDT, SPNO, SPNC
Operating force	0.18 oz to 8.78 oz	12 gf to 355 gf	0.78 oz to 11.01 oz
Termination	Quick connect, solder, pcb	Quick connect, solder, pcb	quick connect, solder, pcb
Actuator	Pin plunger, straight, roller, sim. roller, L-shaped	Pin plunger, straight, roller, sim. roller, L-shaped	pin plunger, straight, roller, sim. roller
Voltage	125 Vac, 250 Vac, 30 Vdc	125 Vac, 250 Vac	125 Vac/125 Vdc 6(2) A 250 Vac
Agency approvals	UL, CE, CSA	UL, cUL, ENEC	UL, CE, CSA, ENEC
Agency file info	CE: 61058-1; UL: E12252; CSA: LR212438	UL: E12252; c-UL: E12252	CE: 61058-1; UL:12252; c-UL: E12252
Operating temperature	-40 °C to 120 °C [-40 °F to 248 °F]	-40 °C to 120 °C [-40 °F to 248 °F]	-40 °C to 120 °C [-40 °F to 248 °F]
Contacts	Silver, gold-plated silver, gold-plated brass, silver-tin-indium oxide	Silver, gold-plated silver, gold-plated brass, silver-tin-indium oxide	Silver, gold-plated silver, silver-tin-indium oxide
Housing	Polyamide (nylon)	Polyamide (nylon)	Polyamide (nylon)
Sealing	None		
Storage humidity	85 % RH max. at 40 °C [104 °F]		
Dielectric strength	1000 Vac (50 Hz to 60 Hz) between contacts, between terminals and ground, for one minute	1000 Vac (50 Hz to 60 Hz)/min	1000 Vac (50 Hz to 60 Hz) between contacts, between terminals and ground, for one minute
Contact resistance	300 mOhm max.	300 mOhm max.	300 mOhm max.
Insulation resistance	100 mOhm min. (at 500 Vdc/min)	100 mOhm min. (at 250 Vdc/min)	100 mOhm min. (at 500 Vdc/min)
Vibration	10 Hz to 55 Hz, displacement 0,75 mm (p-p)		
Expected mechanical life	10 million min.	10 million min. @ <10 A; 1 million min. @ 10 A	10 million min.
Electrical service life	Min. 1,000,000 operations on resistive load current 0.1 A at 125 Vac; 0.1 A at 30 Vdc; Min. 6,000 operations on resistive load 5 A at 125/250 Vac	Min. 10,000 operations	Min. 1,000,000 operations @ 0.1 A; Min 10,000 operations on resistive and motor load current 6(2) A 250 Vac
Electrical operating frequency	0.1 A – 120 operations/min other – 10 to 30 operations/min	10 to 30 operations/min	0.1 A – 120 operations/min; Other – 10 to 30 operations/min
Mechanical operation frequency	120 operations/min.		

Snap-Action Switches

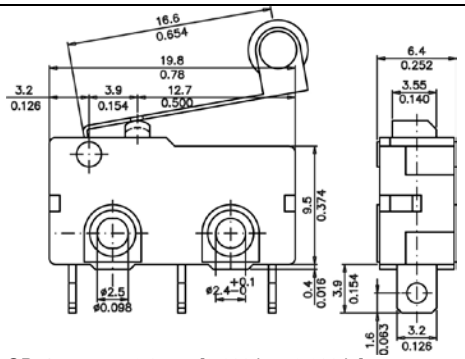
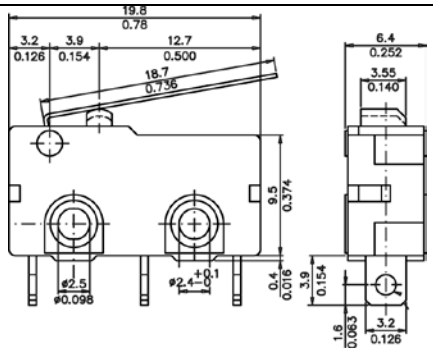
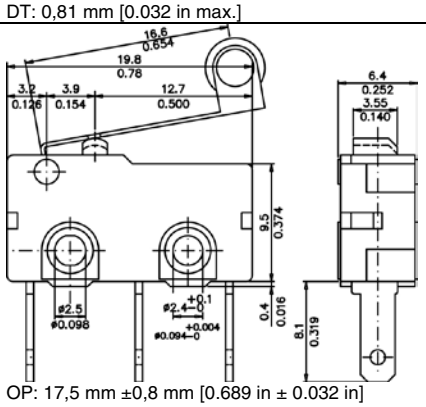
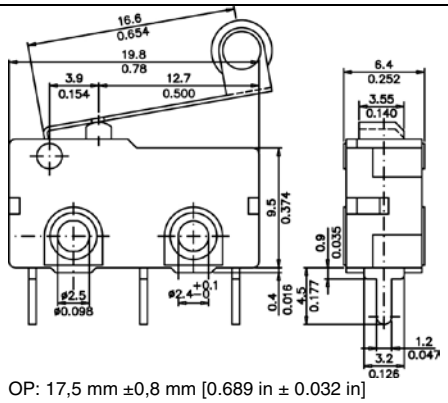
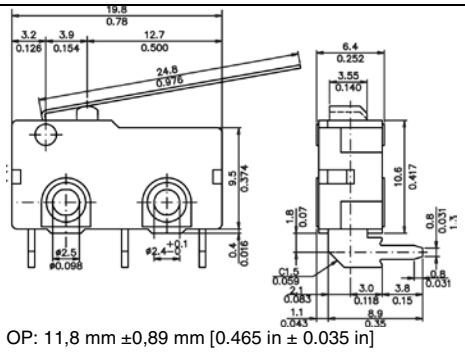
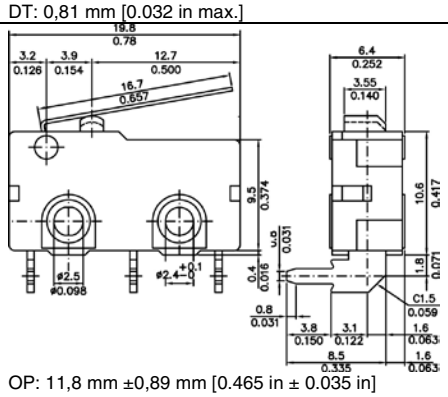
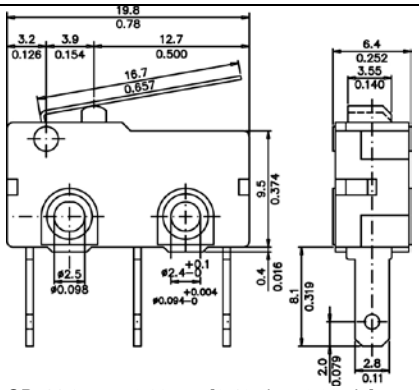
			
SERIES	**ZW (water-tight)**	**ZD (water-tight)**	**ZX**
Differentiator	IP67 rating with lead wires; snap-on lever, coil spring, and ENEC	Smaller sized (like the ZX), sealed to IP67 (with leadwires only); plunger travel can be restricted, offers side-post quick mounting	Two-thirds the size of the ZM Series; unsealed, integral lever, and coil spring
Use	Use when a sealed position switch in a small and cost-effective package is required	Use for automotive applications due to sealing and quick mounting option	Use when a much smaller unsealed position switch is required
Potential applications	air conditioners, computers, consumer appliances, gas detectors, joysticks, money sorters, telephones, toys	automotive (operation systems and engine area interior), air conditioners, communication, electric toothbrushes, toys	calculators, computer mouse, cordless phones, electric knife & stapler, tester machines, walkie-talkies
Ampere rating	0.1 A. 5 A	0.1 A, 3 A	0.1 A. 3 A
Circuitry	SPDT, SPNO, SPNC	SPDT	SPDT
Operating force	1.94 oz to 7.16 oz	130 gf to 195 gf	0.53 oz to 5.3 oz
Termination	quick connect, solder, cable bottom exit, cable side exit	Solder, pcb straight, pcb left angle, pcb right angle, pre-wired	solder, pcb snap-in, pcb left angle, pcb right angle
Actuator	pin plunger, straight, roller, sim. roller	Pin plunger, straight, sim. roller	pin plunger, straight, roller, special
Voltage	125 Vac, 250 Vac	125 Vac, 12 Vdc	125 Vac , 48 Vdc
Agency approvals	UL, cUL, CE, ENEC	UL, cUL, CE, ENEC	UL, CE, CSA
Agency file info	CE: 61058-1; UL: E12252; c-UL: E12252	UL: E12252; c-UL: E12252	CE: 61058-1; UL:12252; CSA: LR212438
Operating temperature	-40 °C to 120 °C [-40 °F to 248 °F] (w/o wires) -40 °C to 105 °C [-40 °F to 221 °F] (w/ wires)	-40 °C to 120 °C [-40 °F to 248 °F]	-40 °C to 120 °C [-40 °F to 248 °F]
Contacts	silver, gold-plated silver	Silver, gold-plated silver	silver, gold-plated silver
Housing	PBT polyester thermoplastic	PBT polyester thermoplastic	Polyamide (nylon)
Sealing	IP67 (with leadwires only)	IP67 (with leadwires only)	None
Storage humidity	85 % RH max. at 40 °C [104 °F]		
Dielectric strength	1000 Vac (50 Hz to 60 Hz) between contacts and 1250 Vac (50 Hz to 60 Hz), between terminals and ground, for one minute	150 Vac (50 Hz to 60 Hz)/minute between contacts, 500 Vac (50 Hz to 60 Hz)/minute between live parts and dead metal parts	1000 Vac (50 Hz to 60 Hz) between contacts, between terminals and ground, for one minute
Contact resistance	30 mOhm max.	100 mOhm max.	100 mOhm max.
Insulation resistance	100 mOhm min. (at 500 Vdc/min)	100 mOhm min. (at 250 Vdc/min)	100 mOhm min. (at 500 Vdc/min)
Vibration	10 Hz to 55 Hz, displacement 0,75 mm (p-p)		
Expected mechanical life	2 million min.	500,000 min.	1 million min.
Electrical service life	Min. 10,000 operations	Min. 500,000 operations on resistive load current 10 mA; Min. 6000 operations on resistive load current 3 A	Min. 1,000,000 operations on resistive load current 0.1 A at 48 Vdc; Min. 10,000 operations on resistive load current 3 A at 125 Vac
Electrical operating frequency	10 to 30 operations/min	10 mA – 120 operations/min 3 A – 10 to 30 operations/min	0.1 A – 120 operations/min 3 A – 10 to 30 operations/min
Mechanical operation frequency	120 operations/min.		

MICRO SWITCH™ Standard Subminiature Snap-Action Z Series

ZM AND ZM1 STANDARD LEVER OPTIONS & DIMENSIONS mm/in

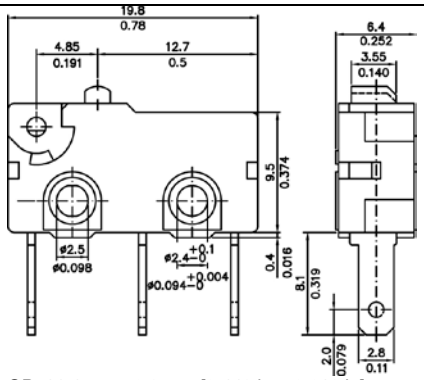
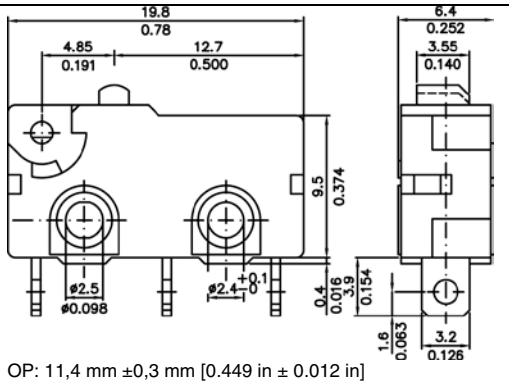
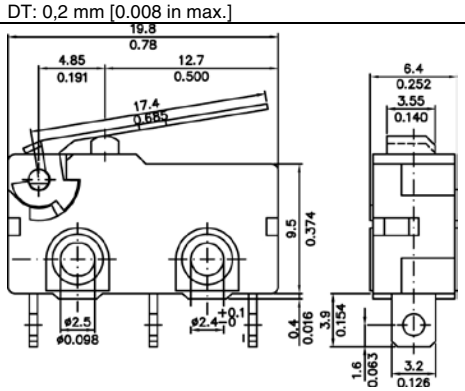
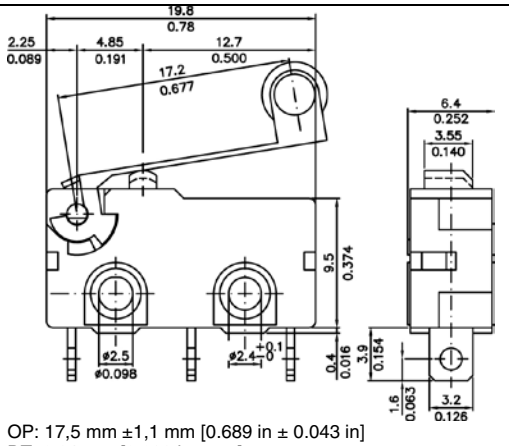
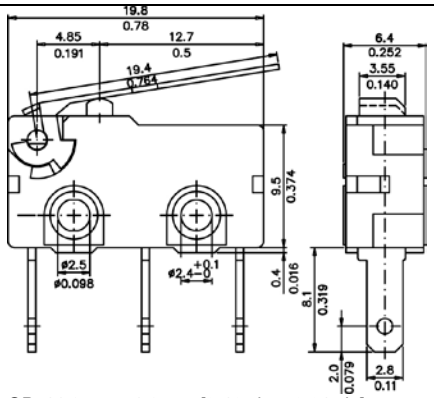
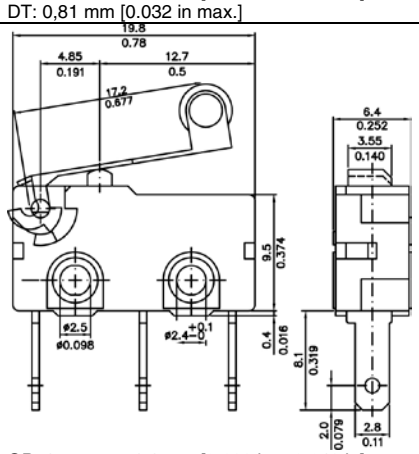
Lever/ Terminals	Dimensions	Lever/ Terminals	Dimensions
Pin plunger/ solder	 OP: 11,4 mm ±0,3 mm [0.449 in ± 0.012 in] DT: 0,2 mm [0.008 in max.]	Pin plunger/ quick connect	 OP: 11,4 mm ±0,3 mm [0.449 in ± 0.012 in] DT: 0,2 mm [0.008 in max.]
Pin plunger/ PCB right	 OP: 11,4 mm ±0,3 mm [0.449 in ± 0.012 in] DT: 0,2 mm [0.008 in max.]	Pin plunger/PCB	 OP: 11,4 mm ±0,3 mm [0.449 in ± 0.012 in] DT: 0,2 mm [0.008 in max.]
Simulated roller/quick connect	 OP: 15,1 mm ±1,5 mm [0.591 in ± 0.059 in] DT: 0,9 mm [0.035 in max.]	Simulated roller/solder	 OP: 15,1 mm ±1,5 mm [0.591 in ± 0.059 in] DT: 0,9 mm [0.035 in max.]

Continued – **ZM AND ZM1 STANDARD LEVER OPTIONS & DIMENSIONS mm/in**

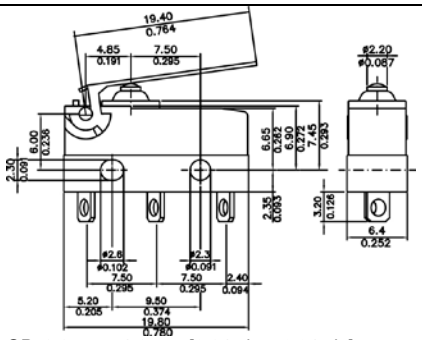
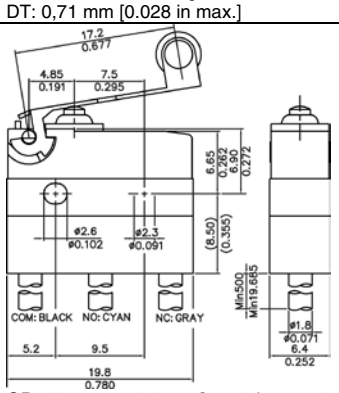
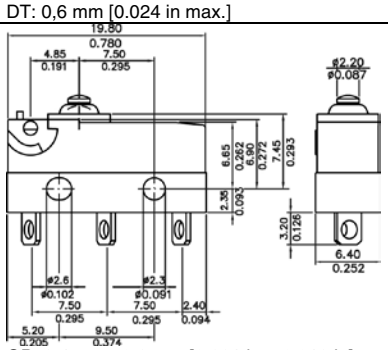
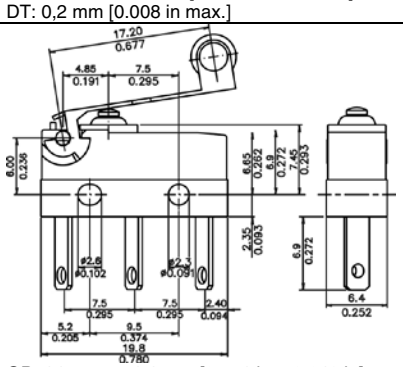
Lever/ Terminals	Dimensions	Lever/ Terminals	Dimensions
Roller/solder	 OP: 17,5 mm ±0,8 mm [0.689 in ± 0.032 in] DT: 0,81 mm [0.032 in max.]	Straight/ solder	 OP: 11,8 mm ±0,89 mm [0.465 in ± 0.035 in] DT: 0,81 mm [0.032 in max.]
Roller/ quick connect	 OP: 17,5 mm ±0,8 mm [0.689 in ± 0.032 in] DT: 0,81 mm [0.032 in max.]	Roller/PCB	 OP: 17,5 mm ±0,8 mm [0.689 in ± 0.032 in] DT: 0,81 mm [0.032 in max.]
Straight/PCB right	 OP: 11,8 mm ±0,89 mm [0.465 in ± 0.035 in] DT: 0,81 mm [0.032 in max.]	Straight/PCB left	 OP: 11,8 mm ±0,89 mm [0.465 in ± 0.035 in] DT: 0,81 mm [0.032 in max.]
Straight/ quick connect	 OP: 11,8 mm ±0,89 mm [0.465 in ± 0.035 in] DT: 0,81 mm [0.032 in max.]		

MICRO SWITCH™ Standard Subminiature Snap-Action Z Series

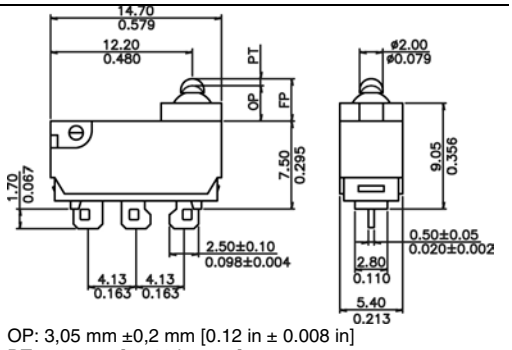
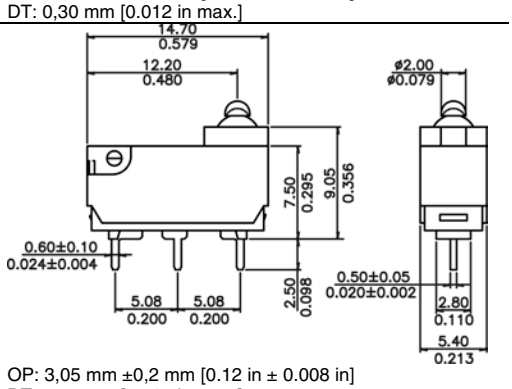
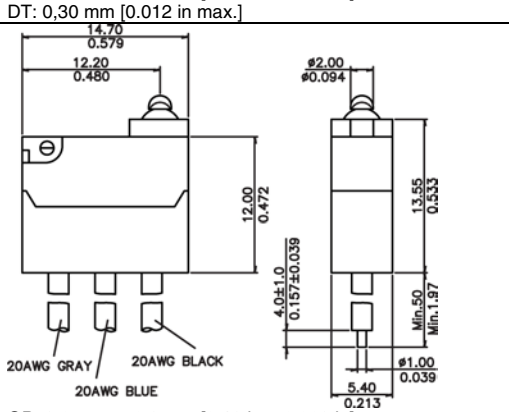
ZV STANDARD LEVER OPTIONS & DIMENSIONS mm/in

Lever/ Terminals	Dimensions	Lever/ Terminals	Dimensions
Pin plunger/ quick connect	 OP: 11,4 mm ±0,3 mm [0.449 in ± 0.012 in] DT: 0,2 mm [0.008 in max.]	Pin plunger/ solder	 OP: 11,4 mm ±0,3 mm [0.449 in ± 0.012 in] DT: 0,2 mm [0.008 in max.]
Straight/ solder	 OP: 11,8 mm ±1,6 mm [0.465 in ± 0.063 in] DT: 0,81 mm [0.032 in max.]	Roller/solder	 OP: 17,5 mm ±1,1 mm [0.689 in ± 0.043 in] DT: 0,81 mm [0.032 in max.]
Straight/ quick connect	 OP: 11,8 mm ±1,2 mm [0.465 in ± 0.047 in] DT: 0,81 mm [0.032 in max.]	Roller/ quick connect	 OP: 17,5 mm ±1,1 mm [0.689 in ± 0.043 in] DT: 0,81 mm [0.032 in max.]

ZW STANDARD LEVER OPTIONS & DIMENSIONS mm/in

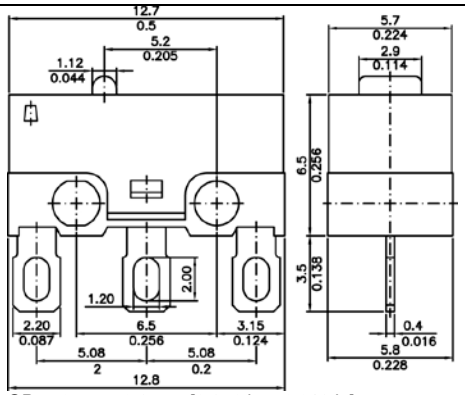
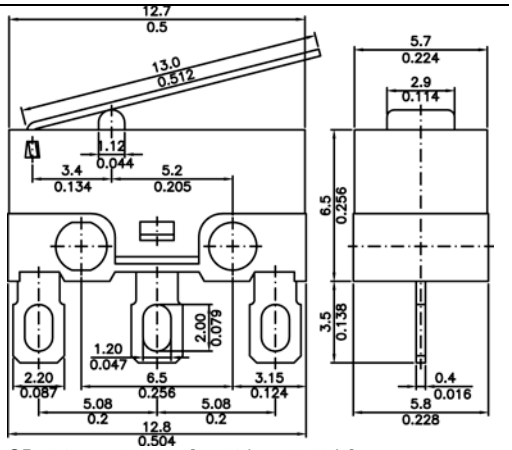
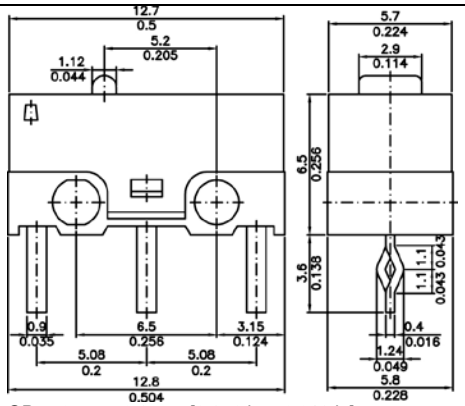
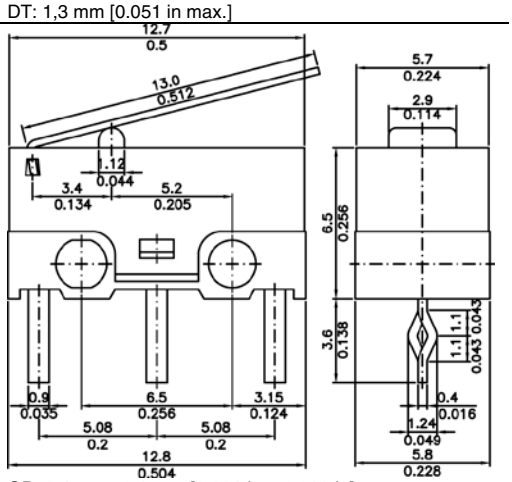
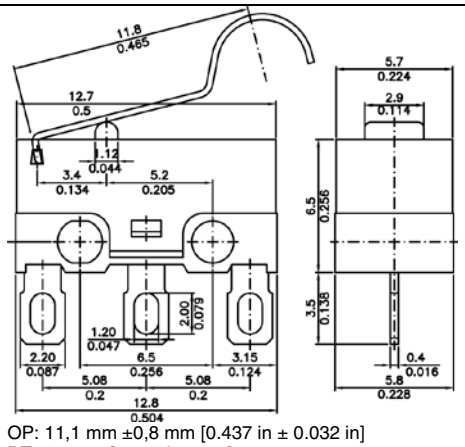
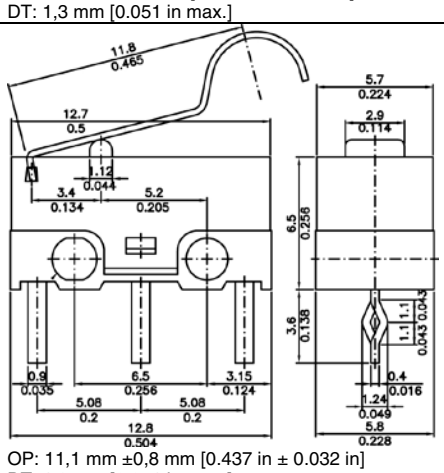
Lever/Term.	Dimensions
Straight/ solder angled	 OP: 8,8 mm ±1,2 mm [0.347 in ± 0.047 in] DT: 0,71 mm [0.028 in max.]
Roller/cable bottom exit	 OP: 14,5 mm ±1,1 mm [0.571 in ± 0.043 in] DT: 0,6 mm [0.024 in max.]
Pin plunger/ solder angled	 OP: 8,4 mm ±0,3 mm [0.331 in ± 0.012 in] DT: 0,2 mm [0.008 in max.]
Roller/quick connect	 OP: 14,5 mm ±1,1 mm [0.571 in ± 0.043 in] DT: 0,6 mm [0.024 in max.]

ZD STANDARD LEVER OPTIONS & DIMENSIONS mm/in

Lever/ Terminals	Dimensions
Pin plunger/ solder terminals	 OP: 3,05 mm ±0,2 mm [0.12 in ± 0.008 in] DT: 0,30 mm [0.012 in max.]
Pin plunger/PCB straight terminals	 OP: 3,05 mm ±0,2 mm [0.12 in ± 0.008 in] DT: 0,30 mm [0.012 in max.]
Pin plunger/wire leads	 OP: 3,05 mm ±0,2 mm [0.12 in ± 0.008 in] DT: 0,30 mm [0.012 in max.]

MICRO SWITCH™ Standard Subminiature Snap-Action Z Series

ZX STANDARD LEVER OPTIONS & DIMENSIONS mm/in

Lever/ Terminals	Dimensions	Lever/ Terminals	Dimensions
Pin plunger/ solder	 OP: 7,0 mm ±0,3 mm [0.276 in ± 0.012 in] DT: 0,30 mm [0.012 in max.]	Straight/ Solder	 OP: 8,4 mm ±0,8 mm [0.331 in ± 0.032 in] DT: 1,3 mm [0.051 in max.]
Pin plunger/PCB	 OP: 7,0 mm ±0,3 mm [0.276 in ± 0.012 in] DT: 0,30 mm [0.012 in max.]	Straight/PCB	 OP: 8,4 mm ±0,8 mm [0.331 in ± 0.032 in] DT: 1,3 mm [0.051 in max.]
Simulated roller/solder	 OP: 11,1 mm ±0,8 mm [0.437 in ± 0.032 in] DT: 1,3 mm [0.051 in max.]	Simulated roller/PCB	 OP: 11,1 mm ±0,8 mm [0.437 in ± 0.032 in] DT: 1,3 mm [0.051 in max.]

Snap-Action Switches

ZM SERIES NOMENCLATURE TREE

ZM

Switch type (Notes 1, 4)

ZM Series Subminiature

50

Current rating

10	0.1 A: 125 Vac (gold-plated)
50	5 A: 125/250 Vac
90	10.1 A: 125/250 Vac (Note 2)

E

Operating force *(at pin plunger)*

B	60 g max.
D	104 g max.
E	146 g max.
G (ONLY)	249 g max.

10

Terminal type

10	Solder straight
20	PCB straight
50	PCB right angle
60	PCB left angle
70	Quick connect 0.110 in
99	Special (Note 3)

A

Actuator type (Note 5) *(integral levers)*

A	Pin plunger
B	Short straight lever 16,7 [0.66]
C	Standard straight lever 18,7 [0.74]
D	Long straight lever 24,8 [0.98]
K	Longer straight lever 35,2 [1.39]
J	Longest straight lever 55,2 [2.17]
F	Roller lever 16,6 [0.65]
H	Small simulated roller lever 17,9 [0.71]
E	Standard simulated roller lever 18,0 [0.71]
M	Large simulated roller lever 21,1 [0.83]
L	L-shaped lever 31,5 [1.24]
S	Special lever (Note 3)

01

Circuitry

01	SPDT UL, CE, CSA
03	SPNO UL, CE, CSA
04	SPNC UL, CE, CSA

Special Designator

A special designator letter is used only when terminal type is "99" or actuator type is "S" to specify that the termination or the actuator of the switch is special. Review product specification to determine the exact differences.

NOTES

(1) Nomenclature is for identification purposes only, not all combinations are possible. Variations not set up would require minimum volumes to establish.
(2) Switches with 10.1 A rating are only available with "G" operating force.
(3) Terminal type "99" or actuator type "S" designates a special and therefore requires a special designator letter at the end of the listing.
(4) Establishing new nomenclature may require notification to UL and European approvals agencies.
(5) Lever length dimension is measured as follows: straight lever - from the center line of the pivot to the end of the plunger; roller lever or simulated roller lever - from the center line of the pivot point to the center line of the roller diameter.

ZM1 SERIES NOMENCLATURE TREE

ZM1

Switch type (Note 1)

ZM1 Series Subminiature

50

Current rating

10	0.1 A: 125 Vac (gold-plated)
50	3 A: 125/250 Vac
60	6 A: 125/250 Vac
90	10.1 A: 125/250 Vac (Note 2)

C

Operating force *(at pin plunger)*

C	150 g max.

10

Terminal type

10	Solder straight
20	PCB straight
50	PCB right angle
60	PCB left angle
70	Quick connect 0.110 in
99	Special (Note 3)

A

Actuator type *(integral levers)*

A	Pin plunger
B	Short straight lever 14,6 [0.57]
C	Standard straight lever 16,6 [0.65]
D	Long straight lever 22,7 [0.89]
E	Standard simulated roller lever
F	Roller lever
H	Small simulated roller lever
J	Longest straight lever 52,9 [2.08]
K	Long straight lever 33,1 [1.30]
L	L-shaped lever
M	Large simulated roller lever
S	Special lever (Note 3)

01

Circuitry

01	SPDT UL, cUL, ENEC
03	SPNO UL, cUL, ENEC
04	SPNC UL, cUL, ENEC

Special Designator

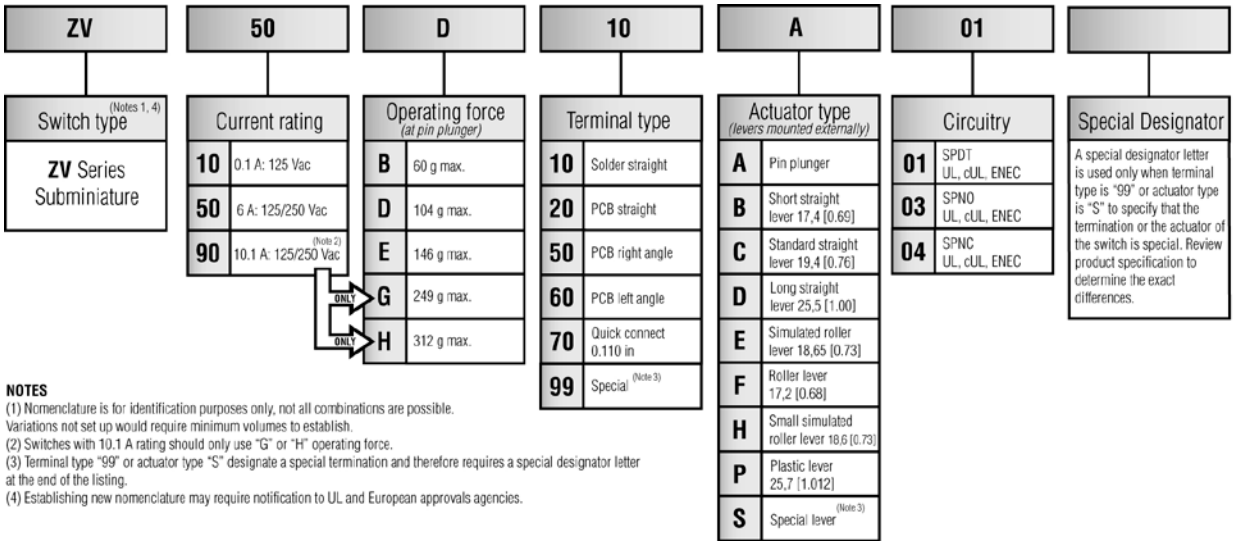
A special designator letter is used only when terminal type is "99" or actuator type is "S" to specify that the termination or the actuator of the switch is special. Review product specification to determine the exact differences.

NOTES

(1) Nomenclature is for identification purposes only, not all combinations are possible. Variations not set up would require minimum volumes to establish.
(2) Switches with 10.1 A rating are only available with "G" operating force.
(3) Terminal type "99" or actuator type "S" designates a special and therefore requires a special designator letter at the end of the listing.

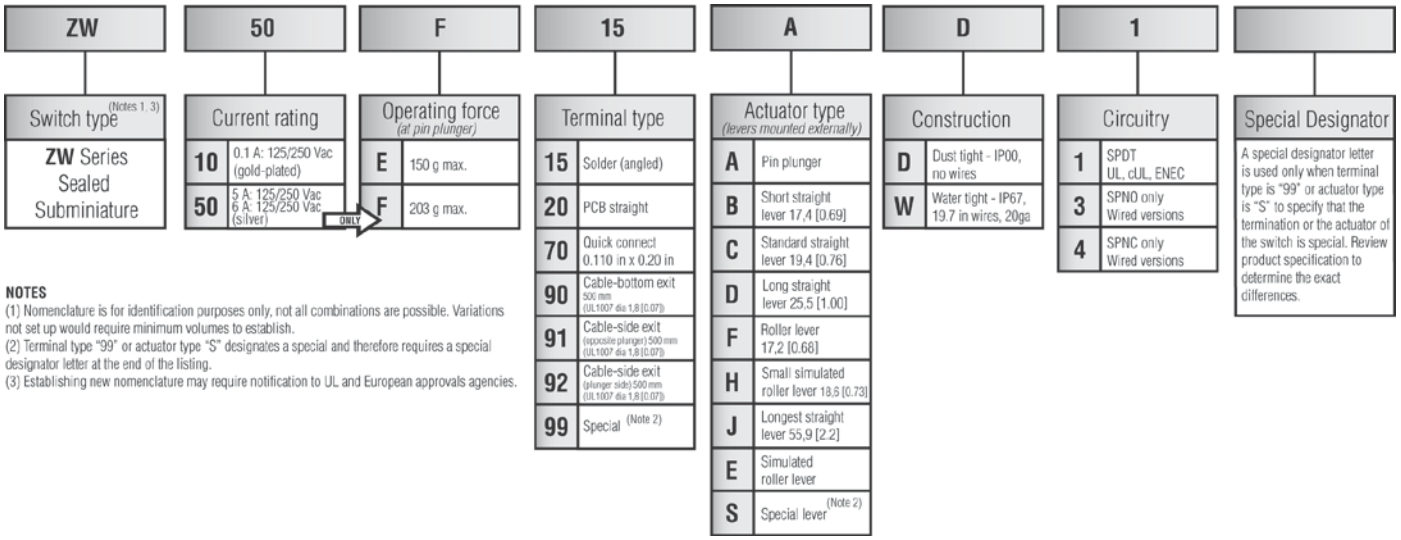
MICRO SWITCH™ Standard Subminiature Snap-Action Z Series

ZV SERIES NOMENCLATURE TREE



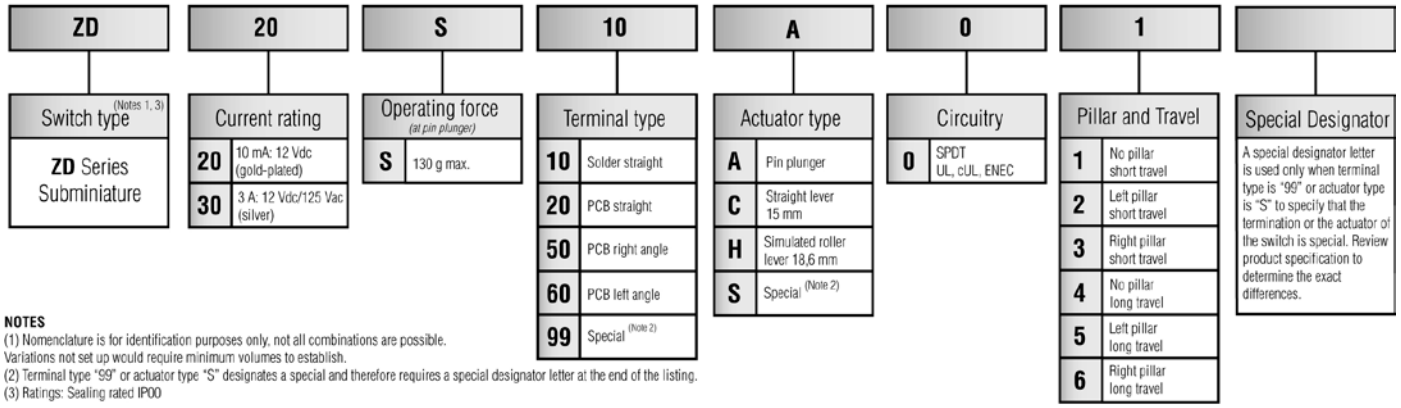
NOTES
(1) Nomenclature is for identification purposes only, not all combinations are possible. Variations not set up would require minimum volumes to establish.
(2) Switches with 10.1 A rating should only use "G" or "H" operating force.
(3) Terminal type "99" or actuator type "S" designate a special termination and therefore requires a special designator letter at the end of the listing.
(4) Establishing new nomenclature may require notification to UL and European approvals agencies.

ZW SERIES NOMENCLATURE TREE



NOTES
(1) Nomenclature is for identification purposes only, not all combinations are possible. Variations not set up would require minimum volumes to establish.
(2) Terminal type "99" or actuator type "S" designates a special and therefore requires a special designator letter at the end of the listing.
(3) Establishing new nomenclature may require notification to UL and European approvals agencies.

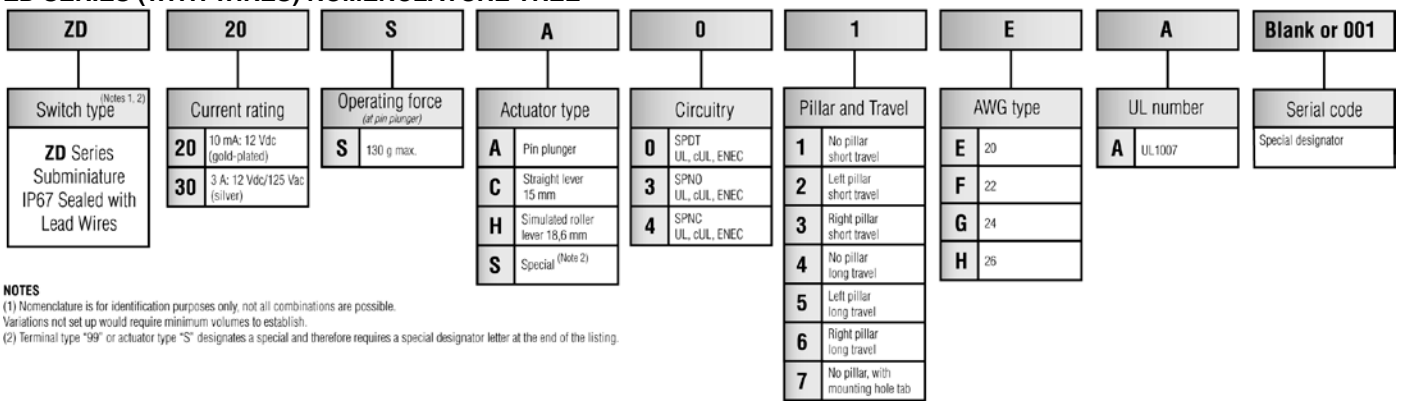
ZD SERIES (NO WIRES) NOMENCLATURE TREE



NOTES
(1) Nomenclature is for identification purposes only, not all combinations are possible. Variations not set up would require minimum volumes to establish.
(2) Terminal type "99" or actuator type "S" designates a special and therefore requires a special designator letter at the end of the listing.
(3) Ratings: Sealing rated IP00

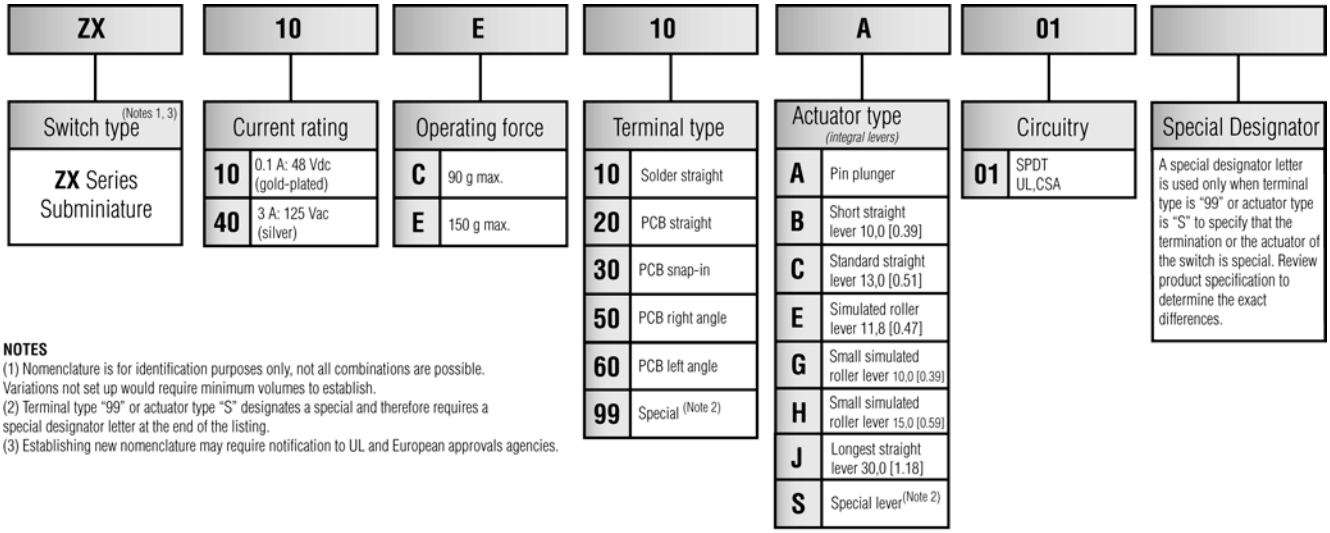
Snap-Action Switches

ZD SERIES (WITH WIRES) NOMENCLATURE TREE



NOTES
(1) Nomenclature is for identification purposes only, not all combinations are possible.
Variations not set up would require minimum volumes to establish.
(2) Terminal type "99" or actuator type "S" designates a special and therefore requires a special designator letter at the end of the listing.

ZX SERIES NOMENCLATURE TREE



NOTES
(1) Nomenclature is for identification purposes only, not all combinations are possible.
Variations not set up would require minimum volumes to establish.
(2) Terminal type "99" or actuator type "S" designates a special and therefore requires a special designator letter at the end of the listing.
(3) Establishing new nomenclature may require notification to UL and European approvals agencies.

⚠ WARNING

PERSONAL INJURY

DO NOT USE these products as safety or emergency stop devices or in any other application where failure of the product could result in personal injury.

Failure to comply with these instructions could result in death or serious injury.

⚠ WARNING

MISUSE OF DOCUMENTATION

- The information presented in this product sheet is for reference only. Do not use this document as a product installation guide.
- Complete installation, operation, and maintenance information is provided in the instructions supplied with each product.

Failure to comply with these instructions could result in death or serious injury.

WARRANTY/REMEDY

Honeywell warrants goods of its manufacture as being free of defective materials and faulty workmanship. Honeywell's standard product warranty applies unless agreed to otherwise by Honeywell in writing; please refer to your order acknowledgement or consult your local sales office for specific warranty details. If warranted goods are returned to Honeywell during the period of coverage, Honeywell will repair or replace, at its option, without charge those items it finds defective. **The foregoing is buyer's sole remedy and is in lieu of all other warranties, expressed or implied, including those of merchantability and fitness for a particular purpose. In no event shall Honeywell be liable for consequential, special, or indirect damages.**

While we provide application assistance personally, through our literature and the Honeywell web site, it is up to the customer to determine the suitability of the product in the application.

Specifications may change without notice. The information we supply is believed to be accurate and reliable as of this printing. However, we assume no responsibility for its use.

SALES AND SERVICE

Honeywell serves its customers through a worldwide network of sales offices, representatives and distributors. For application assistance, current specifications, pricing or name of the nearest Authorized Distributor, contact your local sales office or:

E-mail: info.sc@honeywell.com

Internet: www.honeywell.com/sensing

Phone and Fax:

Asia Pacific	+65 6355-2828
	+65 6445-3033 Fax
Europe	+44 (0) 1698 481481
	+44 (0) 1698 481676 Fax
Latin America	+1-305-805-8188
	+1-305-883-8257 Fax
USA/Canada	+1-800-537-6945
	+1-815-235-6847
	+1-815-235-6545 Fax

Sensing and Control
Honeywell
1985 Douglas Drive North
Golden Valley, MN 55422
www.honeywell.com/sensing

Honeywell

004954-1-EN IL50 GLO Printed in USA
July 2010

Filament: PLA

tonerplastics

PLA 3D Filament Data Sheet

Polylactide (PLA)

Polylactide, also known as Polylactic Acid, is a biodegrade thermoplastic. Synthesized from organic sugars, PLA has become the most common material for 3D filaments due to its eco-friendliness and ease of use. PLA maintains several desirable properties for 3D printing such as a low melting temperature and glass transition temperature. As a result, PLA offers a high level of detail and exceptional print quality

Mechanical Properties		Test Method
Tensile Strength @ Break, PSI	7700	ASTM D638
Yield Strength, PSI	8700	ASTM D638
Tensile Elongation, %	6.0	ASTM D638
Notched Izon Impact, ft-lb/in	0.3	ASTM D256
Size Specifications		
Nominal Outer Diameter, mm	1.75/2.88	-
OD Tolerance, mm	±0.08	-
Ovality, mm	< 0.05	-

Applications

- General purpose 3D printing
- Fine detail prints
- Applications where strength is not critical

Recommended Printer Settings

- Extruder Temperature: 190-230 °C
- Printing Speed: 50-90+ mm/s
- Bed Temperature: 60-70 °C
- Bed Adhesion: Blue Painters Tape

Additional Information

- Biodegradable
- Standard Sizes Available: 1.75/2.88mm
- Custom packaging methods available upon request
- Spool Weight: 1kg., 1lb., 5lb., and 10lb
- All filaments are sealed with desiccants

Regulatory Compliance

- RoHS
- REACH
- Toy Safe

Disclaimer:

The above information is provided in good faith. It is solely the customer's responsibility to determine if the product and information in this document are appropriate for the customer's end use. Customers are always advised to consult their 3D printer manufacturer before using Toner Plastic's filaments.

Filament: PETG

FICHA TÉCNICA

Prusament PETG de Prusa Polymers

PETG es uno de los filamentos más utilizados. Es una excelente opción para imprimir piezas estresadas mecánicamente. En comparación con el PLA, es más resistente al calor, más flexible y menos frágil.

APLICACIONES: El uso típico del PETG es la impresión de piezas funcionales y mecánicas. Gracias a la buena adhesión de la capa, también es adecuado para impresiones impermeables.

NO ES ADECUADO PARA: No apto para piezas pequeñas.

POSTPROCESAMIENTO: Cuando se postprocesa el PETG, es posible usar lijado seco y húmedo.

IDENTIFICACIÓN:

Nombre comercial	Prusament PETG
Nombre químico	Copolyester
Uso	FDM Impresión 3d
Diámetro	1.75 ± 0.02 mm
Fabricante	Prusa Polymers, Praga, Republica Checa

PARÁMETROS DE IMPRESIÓN RECOMENDADOS:

Temperatura del Nozzle [°C]	250 ± 10
Temperature de la Base Calefactable [°C]	80 ± 10
Velocidad de Impresión [mm/s]	hasta 200

PROPIEDADES MATERIALES TÍPICAS:

Propiedades Físicas	Valor Típico	Método
Gravedad Específica [g/cm3]	1.27	ISO 1183
Absorción de humedad en 24 horas [%](1)	0.2	Prusa Polymers
Absorción de humedad 7 días [%](1)	0.3	Prusa Polymers
Absorción de humedad 4 semanas [%](1)	0.3	Prusa Polymers
Temperatura de Deflexión Térmica (0,45 MPa) [°C]	68	ISO 75
Resistencia a la Tracción del Filamento [MPa]	46 ± 1	ISO 527

PROPIEDADES MECÁNICAS DE LAS MUESTRAS DE PRUEBAS IMPRESAS(2):

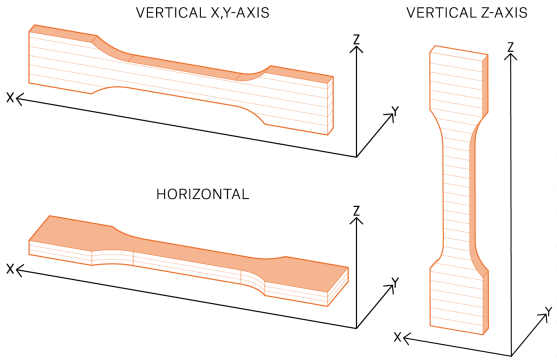
Propiedad / dirección de impresión	Horizontal	Vertical Eje X,Y	Vertical Eje Z	Método
Resistencia a la Tracción [MPa]	47 ± 2	50 ± 1	30 ± 5	ISO 527-1
Módulo de Tracción [GPa]	1.5 ± 0.1	1.5 ± 0.1	1.4 ± 0.1	ISO 527-1
Elongación en el Límite de Elasticidad [%]	5.1 ± 0.1	5.1 ± 0.1	2.5 ± 0.5	ISO 527-1
Fuerza de impacto Charpy(3) [kJ/m2]	NB(C)(4)	NB(4)	5 ± 1	ISO 179-1

(1) 30 °C; humedad 30 %

(2) La impresora 3D Original Prusa i3 MK3 fue utilizado para hacer muestras de prueba. Slic3r Prusa Edition 1.40.0 se utilizó para crear códigos G con la siguiente configuración: Prusa PETG Filament; Ajustes de impresión 0,20mm FAST (capas de 0,2mm); Capas sólidas superiores:0 Inferiores:0; Relleno 100% Rectilíneo, velocidad de relleno 100mm/s; multiplicador de extrusión 1.07; temperatura del extrusor 260°C todas las capas; temperatura de la base calefactable 90°C todas las capas; resto de parámetros por defecto

(3) Charpy sin muesca - Dirección de golpe de borde según ISO 179-1

(4) NB (sin rotura); C (rotura completa) entre paréntesis el segundo tipo de fallo más frecuente > 1/3



Renuncia

Los resultados presentados en esta hoja de datos son solo para su información y comparación. Los valores dependen significativamente de la configuración de impresión, las experiencias de los operadores y las condiciones del entorno. Todos deben considerar la idoneidad y las posibles consecuencias del uso de piezas impresas. Prusa Polymers no puede asumir ninguna responsabilidad por lesiones o pérdidas causadas por el uso del material de Prusa Polymers.