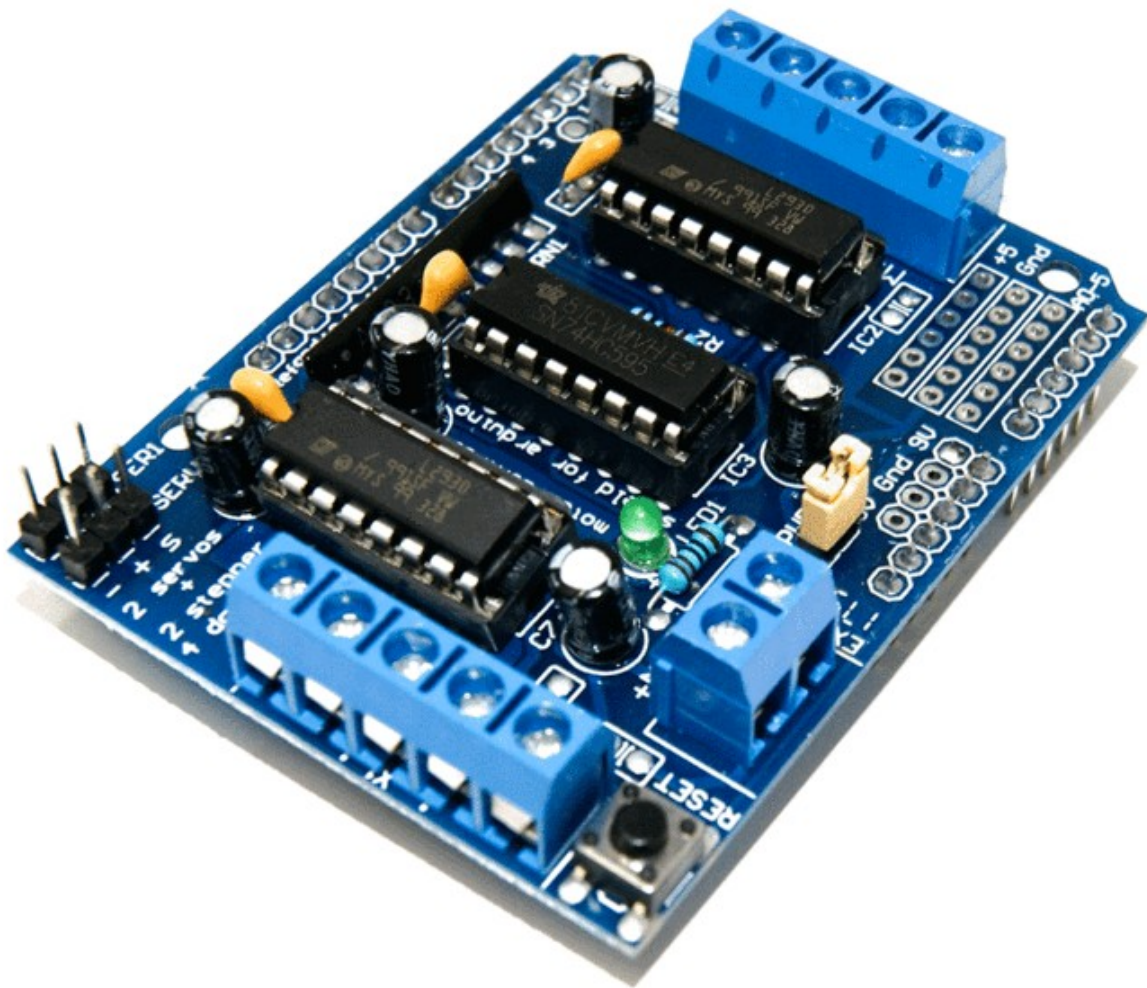




Welcome!

Thank you very much for purchasing our AZ-Delivery 4-channel L293D motor shield for Arduino. On the following pages, we will introduce you to how to use and setup this handy device.

Have fun!



Areas of application

Education and teaching: Use in schools, universities and training institutions to teach the basics of electronics, programming and embedded systems. Research and development: Use in research and development projects to create prototypes and experiments in the fields of electronics and computer science. Prototype development: Use in the development and testing of new electronic circuits and devices. Hobby and Maker Projects: Used by electronics enthusiasts and hobbyists to develop and implement DIY projects.

Required knowledge and skills

Basic understanding of electronics and electrical engineering. Knowledge of programming, especially in the C/C++ programming language. Ability to read schematics and design simple circuits. Experience working with electronic components and soldering.

Operating conditions

The product may only be operated with the voltages specified in the data sheet to avoid damage. A stabilized DC power source is required for operation. When connecting to other electronic components and circuits, the maximum current and voltage limits must be observed to avoid overloads and damage.

Environmental conditions

The product should be used in a clean, dry environment to avoid damage caused by moisture or dust. Protect the product from direct sunlight (UV)

Intended Use

The product is designed for use in educational, research and development environments. It is used to develop, program and prototype electronic projects and applications. The Sensor product is not intended as a finished consumer product, but rather as a tool for technically savvy users, including engineers, developers, researchers and students.

Improper foreseeable use

The product is not suitable for industrial use or safety-relevant applications. Use of the product in medical devices or for aviation and space travel purposes is not permitted

disposal

Do not discard with household waste! Your product is according to the European one Directive on waste electrical and electronic equipment to be disposed of in an environmentally friendly manner. The valuable raw materials contained therein can be recycled become. The application of this directive contributes to environmental and health protection. Use the collection point set up by your municipality to return and Recycling of old electrical and electronic devices. WEEE Reg. No.: DE 62624346

electrostatic discharge

Attention: Electrostatic discharges can damage the product. Note: Ground yourself before touching the product, such as by wearing an anti-static wrist strap or touching a grounded metal surface.

safety instructions

Although our product complies with the requirements of the RoHS Directive (2011/65/EU) and does not contain any hazardous substances in quantities above the permitted limits, residues may still be present. Observe the following safety instructions to avoid chemical hazards: Caution: Soldering can produce fumes that can be harmful to health. Note: Use a solder fume extractor or work in a well-ventilated area. If necessary, wear a respirator mask. Caution: Some people may be sensitive to certain materials or chemicals contained in the product. Note: If skin irritation or allergic reactions occur, stop use and, if necessary, consult a doctor. Caution: Keep the product out of the reach of children and pets to avoid accidental contact and swallowing of small parts. Note: Store the product in a safe, closed container when not in use. Attention: Avoid contact of the product with food and drinks. Note: Do not store or use the product near food to prevent contamination. Although our product complies with the requirements of the RoHS Directive (2011/65/EU) and does not contain any hazardous substances in quantities above the permitted limits, residues may still be present. Observe the following safety instructions to avoid chemical hazards: Caution: Soldering can produce fumes that can be harmful to health. Note: Use a solder fume extractor or work in a well-ventilated area. If necessary, wear a respirator mask. Caution: Some people may be sensitive to certain materials or chemicals contained in the product. Note: If skin irritation or allergic reactions occur, stop use and, if necessary,

consult a doctor. Caution: Keep the product out of the reach of children and pets to avoid accidental contact and swallowing of small parts. Note: Store the product in a safe, closed container when not in use. Attention: Avoid contact of the product with food and drinks. Note: Do not store or use the product near food to prevent contamination. The product contains sensitive electronic components and sharp edges. Improper handling or assembly can result in injury or damage. Observe the following safety instructions to avoid mechanical hazards: Attention: The product's circuit board and connectors may have sharp edges. Use caution to avoid cuts. Note: Wear appropriate protective gloves when handling and assembling the product. Caution: Avoid excessive pressure or mechanical stress on the board and components. Note: Only mount the product on stable and flat surfaces. Use appropriate spacers and housings to minimize mechanical stress. Attention: Make sure the product is securely fastened to prevent accidental slipping or falling. Note: Use appropriate support or secure mounting in enclosures or on mounting plates. Caution: Make sure all cable connections are connected securely and correctly to avoid strain and accidental unplugging. Note: Route cables so that they are not under tension and do not pose a tripping hazard. The product operates with electrical voltages and currents that, if used improperly, can result in electric shocks, short circuits or other hazards. Observe the following safety instructions to avoid electrical hazards: Attention: Use the product only with the specified voltages. Note: The performance limits of the product can be found in the associated data sheet Caution: Avoid short circuits between the connectors and components of the product Note: Make sure that no conductive objects touch or bridge the circuit board. Use insulated tools and pay attention to the arrangement of connections. Caution: Do not perform any work on the product when it is connected to a power source. Note: Disconnect the product from power before making any circuit changes or connecting or removing components. Caution: Do not exceed the specified current ratings for the product's inputs and outputs. Note: The performance limits of the product can be found in the technical specifications or in the data sheet Attention: Make sure that the power sources used are stable and correctly sized. Note: Only use tested and suitable power supplies to avoid voltage fluctuations and overloads. Attention: Maintain sufficient distance from live parts to avoid accidental contact. Note: Ensure that the cabling is arranged safely and clearly according to the voltage used. Caution: Use insulating housings or protective covers to protect the product from direct contact. Note: Place the product in a non-conductive case to avoid accidental touching and short circuits. The product and the components on it may become warm during operation. Improper handling or overloading the product can result in burns, damage or fire. Observe the following safety instructions to avoid thermal hazards: Caution: Make sure the product is used within recommended operating temperatures. Note: The recommended operating temperature range is typically between -40°C and +85°C. Check the specific information in the product data sheet. Attention: Do not place the product near external heat sources such as radiators or direct sunlight. Note: Ensure that the product is operated in a cool and well-ventilated area. Attention: Make sure the product is well ventilated to avoid overheating. Note: Use fans or heat sinks when operating the product in a closed enclosure or in an environment with limited air circulation. Attention: Mount the product on heat-resistant surfaces and in heat-resistant housings. Note: Use enclosure materials that can withstand high temperatures to avoid damage or fire hazard. Caution: Implement temperature monitoring when using an enclosure and, if necessary, protection mechanisms that shut down the product if it overheats. Note: Note: Use temperature sensors and appropriate software to monitor the temperature of the product and shut down the system if necessary. Caution: Avoid overloads that can cause excessive heating of components. Note: To prevent overheating, do not exceed the specified current and voltage limits. Caution: Short circuits can generate significant heat and cause fires. Note: Make sure that all connections are correct and secure and that no conductive objects can accidentally cause short circuits.



The 4-channel L293D motor driver shield is simply plugged onto an Arduino board and allows up to 4 DC motors, 2 stepper motors or 2 servo motors to be controlled. The powerful and reliable L293D chip that distributes the load serves as an H-bridge. This allows you to easily use DC motors and power supplies up to 36V.

The proven design allows motors to be easily connected and controlled by an Arduino and is particularly suitable for fast prototyping. This shield is also suitable for beginners, there are numerous libraries, instructions and example sketches that make controlling the direction or speed of motors child's play. We especially recommend the Adafruit Motor Shield Library (AFM), which you can download from the Arduino IDE.



The most important information in a nutshell

- » Labeled connections for easy connection
- » Compatible with Arduino Mega, Diecimila, Duemilanove and Uno R3
- » 2 connections for 5V servo motors with connection to the Arduino timer for smooth control
- » For controlling 4 DC motors, 2 stepper motors or 2 servo motors
- » Up to 4 bidirectional DC motors with individual 8-bit control
- » Up to 2 stepper motors (unipolar or bipolar) with single coil, double coil or interleaved stepping
- » 4 H-bridges: 0.6A (1.2A peaks) with thermal protection for motors from 4.5V to 36V DC
- » Pull-down resistors to stop the motors when switched on
- » 2 connections for external power supply, separate for logic and motor supply
- » Status LED for operation indication
- » Reset button
- » Dimensions: 70*55mm



All links at a glance

Programming interfaces

- » Arduino IDE: <https://www.arduino.cc/en/Main/Software>
- » Web-Editor: <https://create.arduino.cc/editor>
- » Arduino extension for SublimeText: <https://github.com/Robot-Will/Stino>
- » Arduino extension "VisualMicro" for Atmel Studio or Microsoft Visual Studio: <http://www.visualmicro.com/page/Arduino-for-Atmel-Studio.aspx>

Arduino Tutorials, References, Community

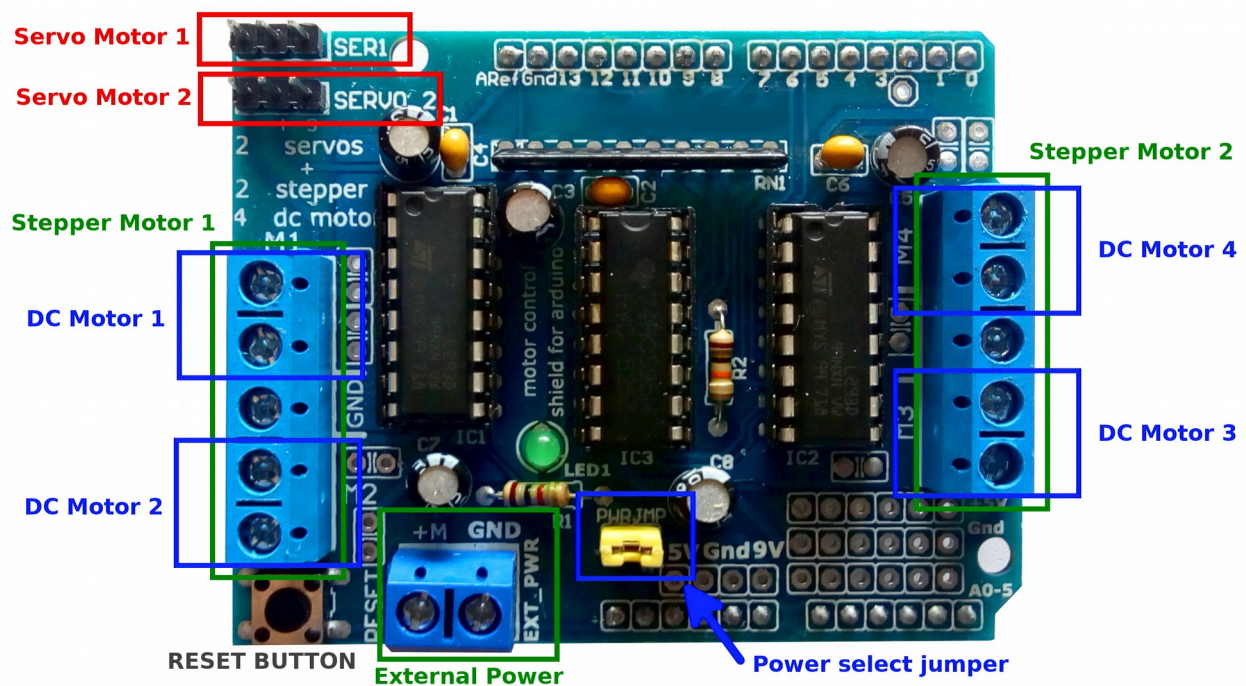
- » <https://www.arduino.cc/en/Tutorial/HomePage>
- » <https://www.arduino.cc/en/Reference/HomePage>

Interesting from AZ-Delivery

- » Arduino Accessories:
<https://az-delivery.de/collections/arduino-zubehor>
- » AZ-Delivery G+Community:
<https://plus.google.com/communities/115110265322509467732>
- » AZ-Delivery on Facebook:
<https://www.facebook.com/AZDeliveryShop/>

Overview

4 DC motors or 2 stepper motors can be connected, the voltage range is 4.5 - 13.5 V. Additionally there are 2 connections for standard servo motors for 5V.



The shield uses the standard PWM pins of the Arduino board to control the connected motors and servos. The Shield is compatible with every Arduino (e.g. Uno, Leonardo, Due, Mega). One Arduino board can use only Motor Shield at a time.



Used pins of the Arduino

PIN 11: DC Motor 1 / Stepper motor 1 (Activation / Speed)

PIN 3: DC Motor 2 / Stepper motor 1 (Activation / Speed)

PIN 5: DC Motor 3 / Stepper motor 1 (Activation / Speed)

PIN 6: DC Motor 4 / Stepper motor 1 (Activation / Speed)

PINs 4, 7, 8 and 12 are used to control the DC / stepper motors via the 74HC595.

These PINs are only used for the servo motors.

PIN 9: Servo motor 1 control

PIN 10: Servo motor 2 control

The six analog inputs (PINs 14 to 19) and the digital inputs (PIN 2 and 13) are not used.

Selection of motors; hardware

Motor voltage

Most motors require voltages from 6V to 12V. These can be operated with the Arduino Shield.

Motors with voltages from 1.5V to 3V cannot be operated.

Hardware motor voltage

Most motors require voltages from 6V to 12V. These can be operated with the motor shield. Motors with voltages from 1.5V to 3V can not be used.

Current range

The Arduino Shield is designed for 0.6 A per motor. The peak value may be up to 1.2 A for a short time. If a large amount of current is required, the ICs used become so hot and must be cooled.

NiMH batteries are the best power supply. An operation on e.g. 9V battery blocks is not recommended. It is best to disconnect the power supply of the motors from the power supply of the Arduino (2 power supplies).

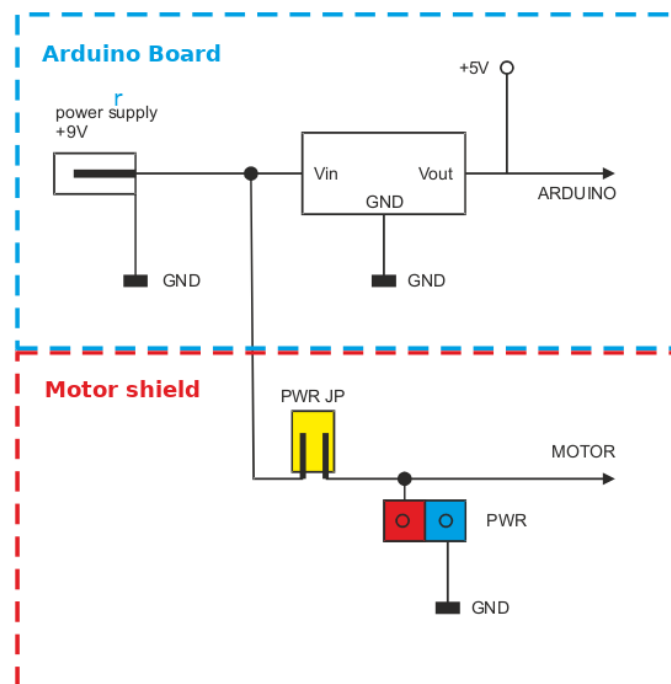
Many problems during the operation of an Arduino system come from disturbances generated by motors on a common power supply. Large fluctuations in power consumption caused by different loads on the motors result in voltage fluctuations that can "confuse" an Arduino program.

Separation of power supplies can be done by removing Power select jumper, and connecting external power supply on motor shield (image above). This way, Arduino board has its own power supply, and motor shield has external power supply.

If jumper is not removed, power for motors and driver, will be used from Arduino board (Arduino and Motor shield have the same power supply).

Power supply of motors

DC motors must be supplied with their own power supply, as some of them draw high currents. They must not be connected to the 5V pins of the Arduino board. This could destroy the Arduino board or the USB port.



There are two possibilities for connection:

1. The DC plug (power supply) on the Arduino board.

There is a protective diode on the plug. This protects the Arduino against incorrect voltage.

2. The 2-pole screw terminal PWR on the Motor Shield.

There is also a protective circuit on the screw terminal which prevents damage to the motor shield.



Option 1: Same power supply for Arduino Board and Motor Shield (1 battery)

A voltage source 6 - 12V e.g. battery is connected to the plug of the Arduino board or to the screw terminals of the motor shield.

IMPORTANT: The jumper Power selection jumper must be plugged into the motor shield! This can lead to interference, as the voltage can fluctuate, depending on the current consumption of the motors. This operating mode is only recommended if a strong battery pack is used.

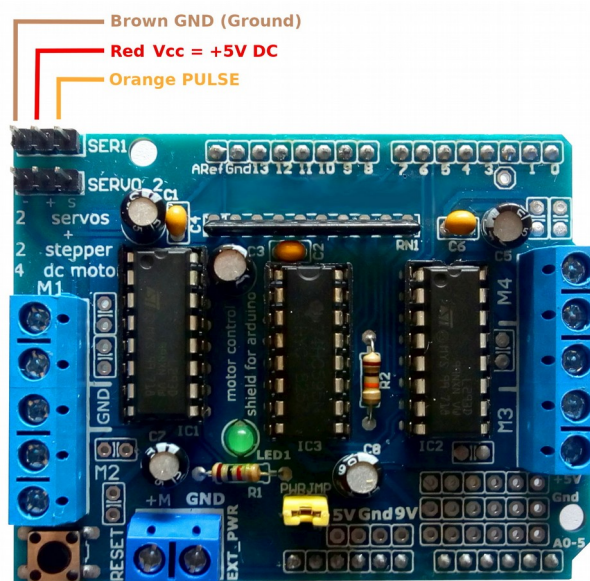
Option 2: Different power supply for Arduino Board and Motor Shield (2 batteries)

The Arduino is supplied either via the USB cable or via the power connector. An additional power supply is connected to the screw terminals of the Motor Shield.

IMPORTANT: The jumper Power selection jumper on the Motor Shield must not be plugged in! This is the recommended method when motors or larger loads are used. Logic voltage and load voltage are decoupled.

Connection for servo motors

The connectors of the Arduino Shields are designed for small model servos. They are powered directly by the 5V of the Arduino Board. If larger servos are used (more power), then the tracks have to be disconnected from the servo connections to the pin header and cables have to be soldered to the external 5V supply for servo motors.



Model servo motors are very easy to use.

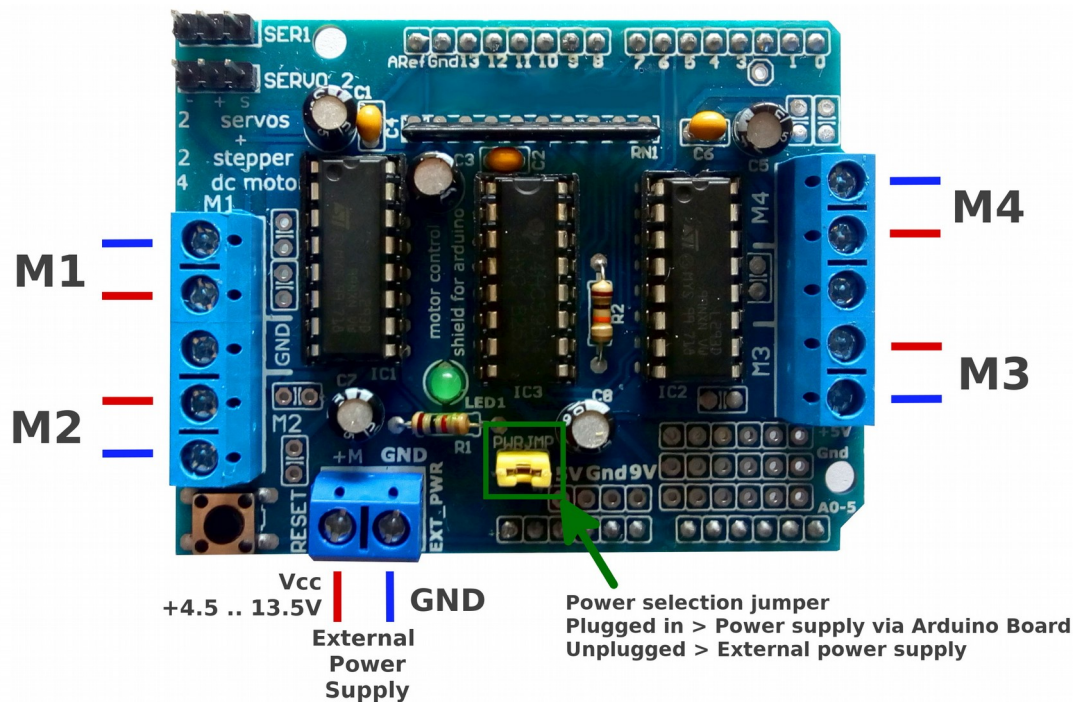
They have a 3-pin female cable. The signals are transmitted as PWM signals, there is a PULS connector on 3-pin cable for this purpose (often designed as orange or white wire). Power is supplied via the Arduino Board. The other two wires of 3-pin cable are for the power supply: Red = Vcc (+5V) and Black or Brown = GND.

PIN Assignment

Arduino Pin 9	PWM Servo 1
Arduino Pin 10	PWM Servo 2

Connection of DC motors (2 wire)

4 bidirectional DC motors can be connected: M1, M2, M3, M4. The speed can be changed in steps of 0.5%. The motors can run in both directions depending on the control.



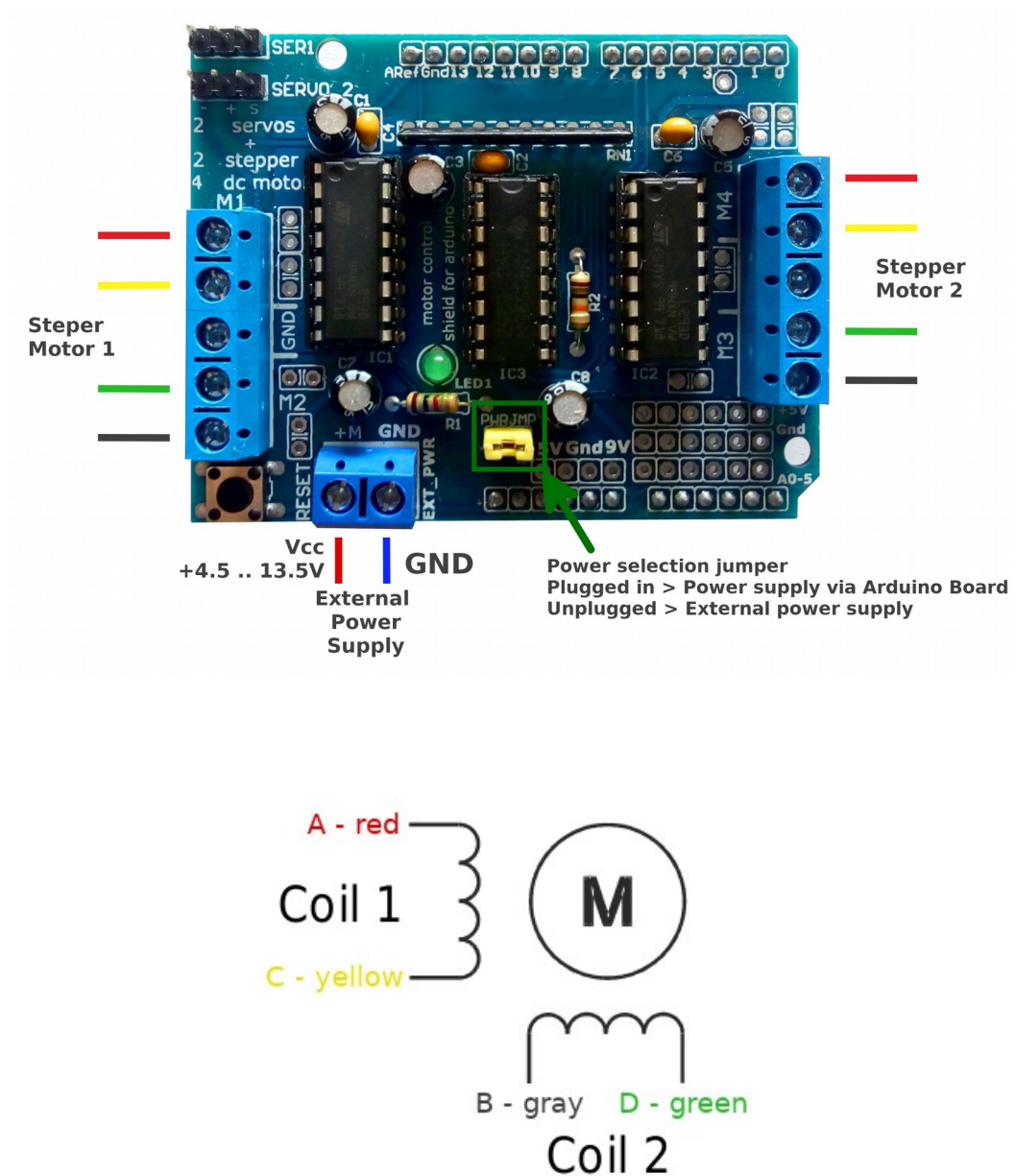
The Motor Shield does not have a heat sink. It is not designed for a constant motor current of 0,6 A. For larger motors or long running times it is recommended to glue a heat sink onto the driver IC.

Both wires of the motor are connected to the corresponding terminals.

The direction of rotation can be changed by exchanging the wires. (e.g. all motors should run in the same direction.)

Connection of stepper motors

2 stepper motors can be connected: both uni-polar and bi-polar.



Az-Delivery

When connecting 2 stepper motors, make sure that the same coils are always connected to one DC Motor connection (the colours used are used just for example, they are different for each stepper motor).

Coil 1 of stepper motor 1 > M1

Coil 2 of stepper motor 1 > M2

Coil 1 of stepper motor 2 > M3

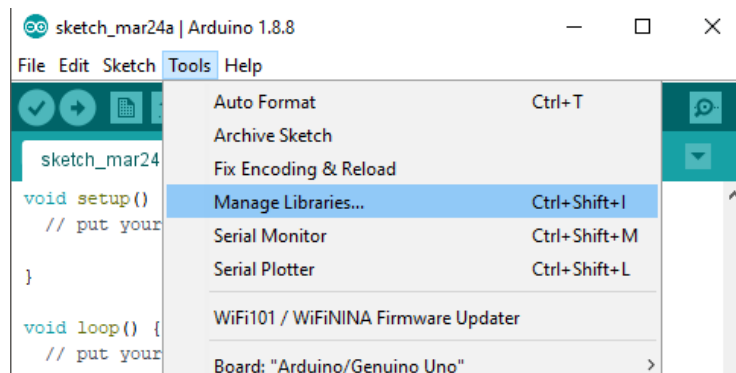
Coil 2 of stepper motor 2 > M4

Az-Delivery

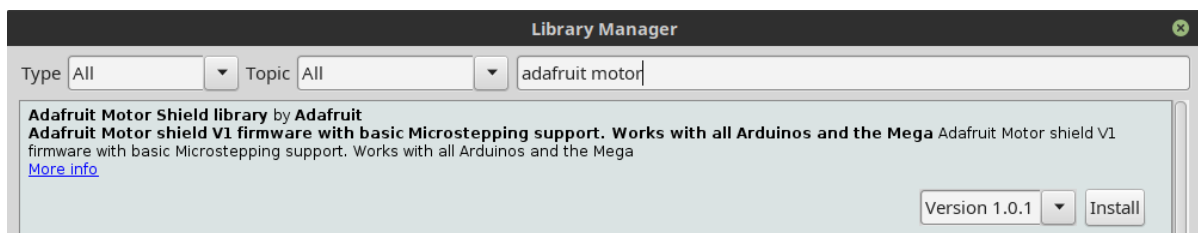
Loading the library

We recommend the "Adafruit Motorshield v1 library" to be used to program the motor shield. We can download it in arduino IDE by going to:

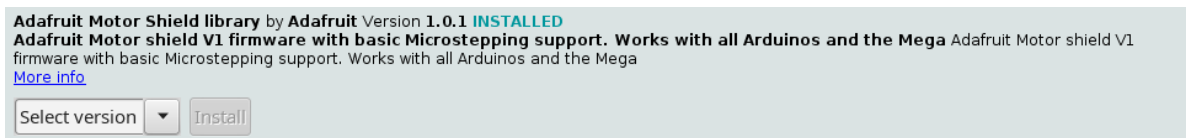
Tools > Manage library



New window will open and in the search box type "*adafruit motor*"



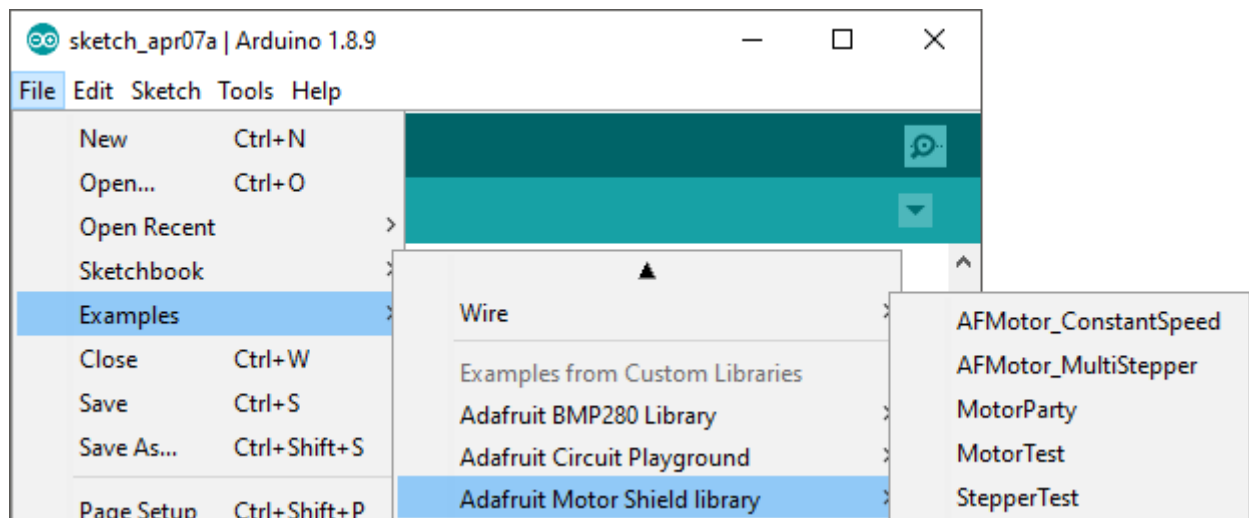
Click on install button and when it is installed "INSTALLED" will be appear near the of library name .



Az-Delivery

ATTENTION: Do not load the library for the Motor Shield V2. It is not compatible with the Arduino Motor Shield.

After loading you can have a look at the example sketches.





Example sketch for the control of a servo motor

```
// Adafruit Motor shield library copyright
// Adafruit Industries LLC, 2009
// this code is public domain, enjoy!

#include <AFMotor.h>
#include <Servo.h>
Servo servo1;                      // DC hobby servo
void setup() {
    Serial.begin(9600);            // set up Serial library at 9600 bps
    Serial.println("Servo Test!");
    servo1.attach(9);              // servo switch on
}
int i;
void loop() {
    for(i = 0; i < 255; i++) {
        servo1.write(i);
        delay(3);
    }
    for(i = 255; i != 0; i--) {
        servo1.write(i-255);
        delay(3);
    }
}
```



Functions for the control of a DC motor

```
#include <AFMotor.h>
```

To create the DC motor object:

```
AF_DCMotor motor(number, frequency)
```

This command has 2 arguments:

- » number - Motor connection 1,2,3 or 4.
- » frequency – Rotation speed control signal

Rotation speed for motors 1 and 2 these frequencies can be selected:

```
MOTOR12_64KHZ,  
MOTOR12_8KHZ,  
MOTOR12_2KHZ or  
MOTOR12_1KHZ.
```

And for motors 3 and 4, only 1KHz can be selected.

All other settings are ignored.

Hints: High speeds e.g. 64KHz are not used. The choice of a low speed e.g. 1KHz leads to a lower energy consumption.

To set the speed of the motor, we use:

```
motor.setSpeed(speed)
```

- » speed - is integer number which limits are from 0 (motor stop) to 255 (maximum speed).

Az-Delivery

To move the motor we use:

```
motor.run(direction)
```

» direction - can be one of three values:

FORWARD

BACKWARD

RELEASE

The directions "*forwards*" and "*backwards*" are not fixed. They depend on the wiring of the individual motors. They can be changed by simply exchanging the connections.



Example sketch for the control of a DC motor

An object with the name "*motor*" is created. All commands refer to <motor>. The sketch is taken from the examples of the "Adafruit Motor shield library".

```
// Adafruit Motor shield library copyright
// Adafruit Industries LLC, 2009
// this code is public domain, enjoy!
#include <AFMotor.h>
AF_DCMotor motor(2, MOTOR12_64KHZ);    // Creating an object of
                                        // <motor> class:
                                        // port 2, 64KHz pwm

void setup() {
    Serial.begin(9600);
    Serial.println("Motor test!");
    motor.setSpeed(200);                // Speed 200 (maximal 255 possible)
}

void loop() {
    Serial.print("tick");
    motor.run(FORWARD);                // Motor goes forward
    delay(1000);
    Serial.print("tock");
    motor.run(BACKWARD);               // Motor goes backwards
    delay(1000);
    Serial.print("tack");
    motor.run(RELEASE);                // Motor stops
    delay(1000);
}
```



Functions for the control of a stepper motor

```
#include <AFMotor.h>
```

To create the stepper motor object we use:

```
AF_Stepper name(steps, stepper)
```

- » name - name of the motor object, e.g. "*motor*" is specified for all other commands.

- » steps - resolution of the stepper motor

 - e.g. 7.5 degree stepper motor has $360/7.5 = 48$ steps resolution

 - e.g. 1.8 degree stepper motor has $360/1.8 = 200$ steps resolution

- » stepper - number for connection of the stepper motor

 - Connection 1 = Terminals M1 and M2

 - Connection 2 = Terminals M3 and M4

To set the speed of the motor we use:

```
motor.setSpeed(rpm)
```

- » rpm - number of revolutions per minute of the stepper motor

To start the motor we use:

```
motor.step(steps, direction, steptype)
```

- » steps - number of steps

- » direction - rotation direction of motor, can be one of these values:

 - FORWARD

 - BACKWARD

Az-Delivery

» steptype - type of step:

SINGLE - only one coil is active

DOUBLE - both coils are simultaneously active (higher torque)

INTERLEAVE - Switching between SINGLE and DOUBLE to double the resolution, thus halving the speed

MICROSTEP - The coils are controlled with a PWM signal.

This results in a "*soft*" movement between the individual steps.

Stop the engine

By default, the motor keeps its position active after the step is completed. (high power consumption). If you want the motor to move freely (low power consumption), you must order it separately.

```
motor.release()
```

Program flow

The step commands BLOCK the program flow, they are not executed in the background. Program flow waits until the motor movement is completed. (There is no multitasking!)



Example sketch for the control of a stepper motor

An object with the name "*motor*" is created. All commands refer to <motor>. The sketch is taken from the examples of the "Adafruit Motor shield library".

```
// Adafruit Motor shield library copyright
// Adafruit Industries LLC, 2009
// this code is public domain, enjoy!
#include <AFMotor.h>
AF_Stepper motor(48, 2);    // 48 Steps / revolution, port 2, M3+M4
                             // Object name: motor

void setup() {
    Serial.begin(9600);      // Serial Interface with 9600 bps
    Serial.println("Stepper test!");
    motor.setSpeed(10);      // 10 revolutions / Minute
    motor.step(100, FORWARD, SINGLE); // 100 Steps, forward, single
    motor.release();         // Motor released, low energy consumption
    delay(1000);             // wait 1000ms = 1s
}

void loop() {
    // 100 steps each in different modes
    motor.step(100, FORWARD, SINGLE);
    motor.step(100, BACKWARD, SINGLE);
    motor.step(100, FORWARD, DOUBLE);
    motor.step(100, BACKWARD, DOUBLE);
    motor.step(100, FORWARD, INTERLEAVE);
    motor.step(100, BACKWARD, INTERLEAVE);
    motor.step(100, FORWARD, MICROSTEP);
    motor.step(100, BACKWARD, MICROSTEP);
}
```

You've done it, you can now use your module for your projects.



Now it is time to learn and make the Projects on your own. You can do that with the help of many example scripts and other tutorials, which you can find on the internet.

If you are looking for the high quality products for Arduino and Raspberry Pi, AZ-Delivery Vertriebs GmbH is the right company to get them from. You will be provided with numerous application examples, full installation guides, eBooks, libraries and assistance from our technical experts.

<https://az-delivery.de>

Have Fun!

Impressum

<https://az-delivery.de/pages/about-us>