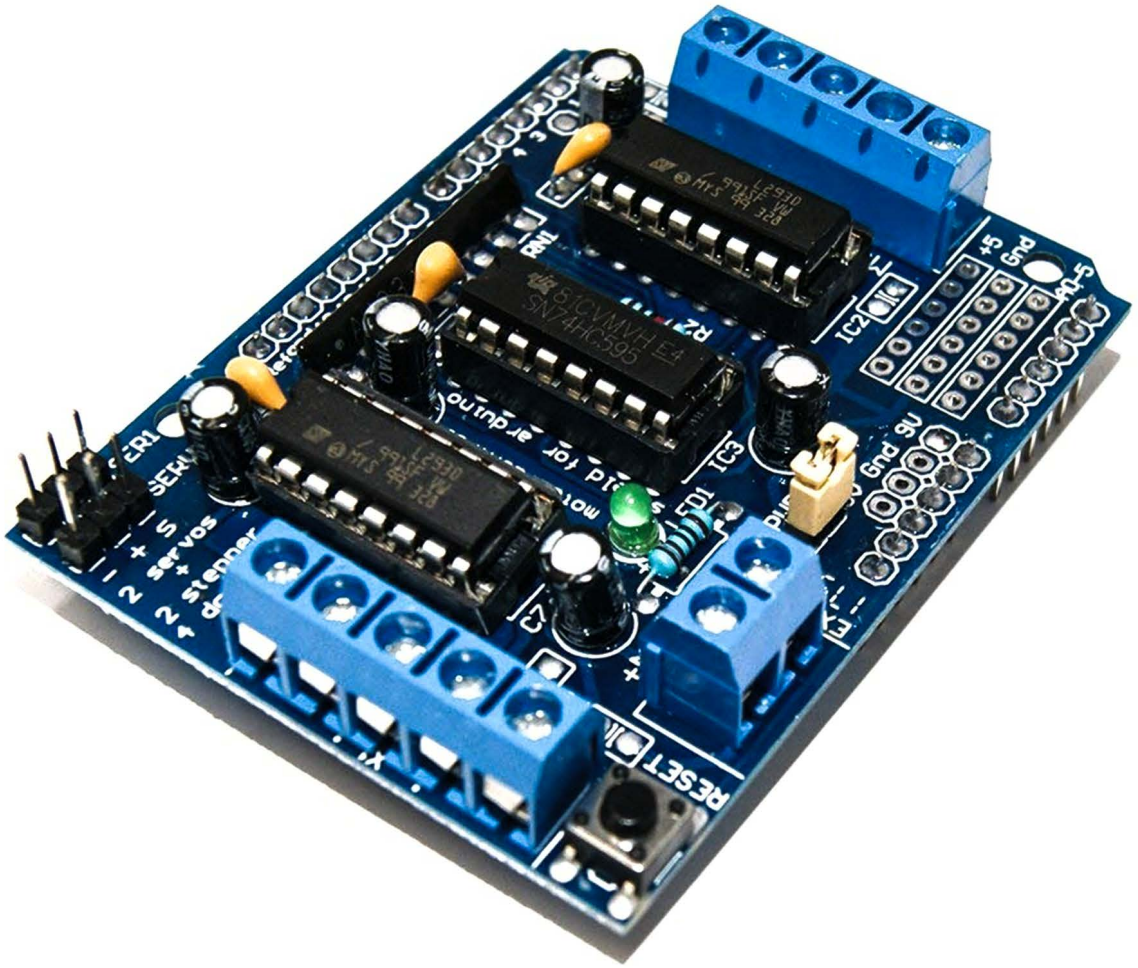
AZ-Delivery
Ihr Experte für Mikroelektronik!

L293D Motortreiber Shield Datenblatt



Contents:

1. Description

L293D is a monolithic integrated, high voltage, high current, 4-channel driver. Basically this means using this chip you can drive DC motors with power supplier up to 36 Volts, and the chip can supply a maximum current of 600mA per channel. L293D chip is also known as a type of H-Bridge.
The H-Bridge is typically an electrical circuit that enables a voltage to be applied across a load in either direction to an output, e.g. motor.

2. Features

• 2 connections for 5V 'hobby' servos connected to the Arduino's high-resolution dedicated timer - no jitter!

• 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.

• 4 H-Bridges: L293D chipset provides 0.6A per bridge (1.2A peak) with thermal shutdown protection, 4.5V to12V

• Pull down resistors keep motors disabled during power-up

• Big terminal block connectors to easily hook up wires (10-22AWG) and power

• Arduino reset button brought up top

• 2-pin terminal block to connect external power, for separate logic/motor supplies

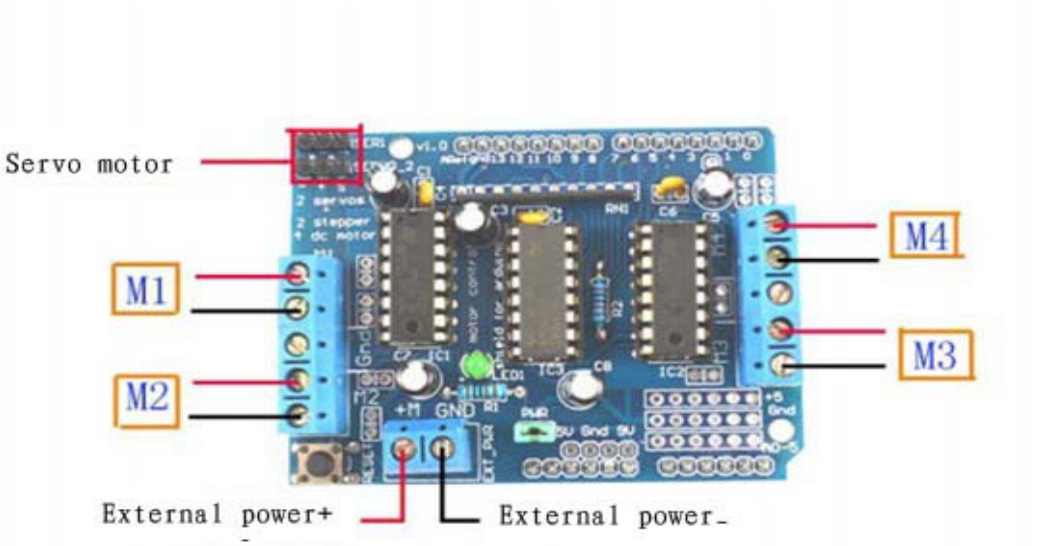
• Tested compatible with Mega, UNO& Duemilanove

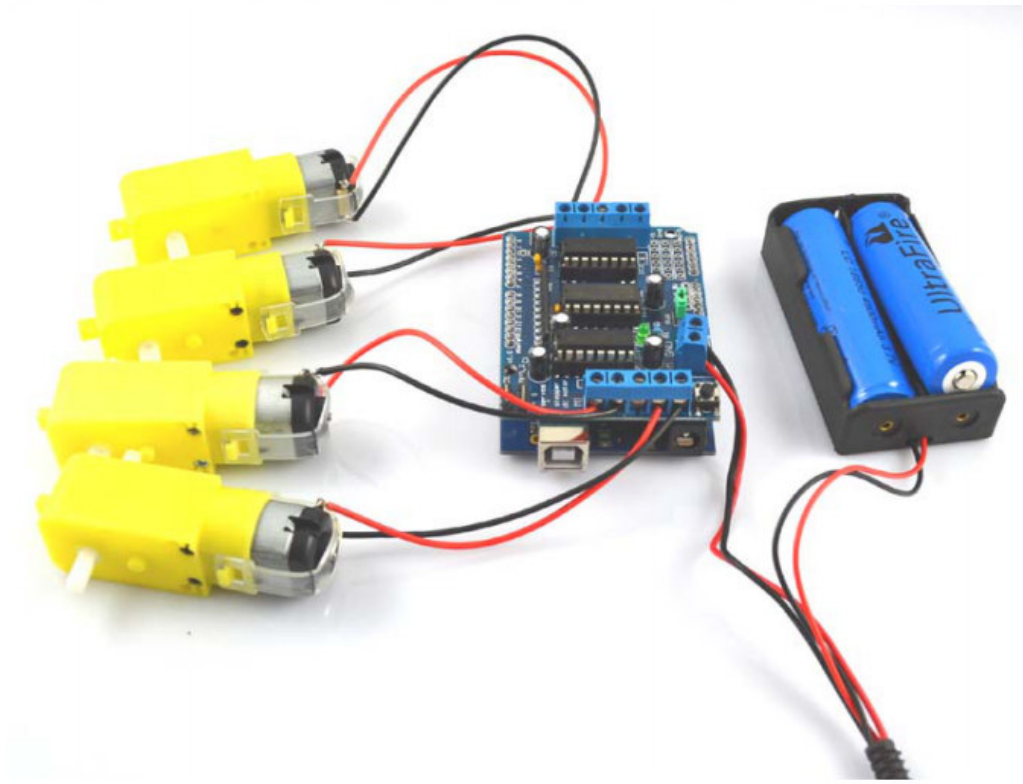
• Dimensions: 69 mm x 53 mm x 14.3 mm (2.7 in x 2.1 in x 0.6 in)

3. Operation

Arduino controller: 1 pcs
L293D: 1 pcs
DC motor: 4 pcs
Power supplier 9V: 1 pcs

Please connect the devices according to the following drawing:





Program source code is as follows: (Adafruit Sketch for Motor Shield)

```
#include <Servo.h>
#define MOTORLATCH 12
#define MOTORCLK 4
#define MOTORENABLE 7
#define MOTORDATA 8
#define MOTOR1_A 2
#define MOTOR1_B 3
#define MOTOR2_A 1
#define MOTOR2_B 4
#define MOTOR3_A 5
#define MOTOR3_B 7
#define MOTOR4_A 0
#define MOTOR4_B 6
#define MOTOR1_PWM 11
#define MOTOR2_PWM 3
#define MOTOR3_PWM 6
#define MOTOR4_PWM 5
#define SERVO1_PWM 10
#define SERVO2_PWM 9
#define FORWARD 1
#define BACKWARD 2
#define BRAKE 3
#define RELEASE 4
Servo servo_1;
Servo servo_2;
void setup() {
  Serial.begin(9600);
  Serial.println("Simple Adafruit Motor Shield sketch");
  servo_1.attach(SERVO1_PWM);
  servo_2.attach(SERVO2_PWM);
}
void loop() {
  motor(1, FORWARD, 255);
  motor(2, FORWARD, 255);
  motor(3, FORWARD, 255);
  motor(4, FORWARD, 255);
  delay(2000); // Be friendly to the motor: stop it before reverse.
  motor(1, RELEASE, 0);
  motor(2, RELEASE, 0);
  motor(3, RELEASE, 0);
  motor(4, RELEASE, 0);
  delay(100);
```

```
  motor(1, BACKWARD, 128);
  motor(2, BACKWARD, 128);
  motor(3, BACKWARD, 128);
  motor(4, BACKWARD, 128);
  delay(2000);
  motor(1, RELEASE, 0);
  motor(2, RELEASE, 0);
  motor(3, RELEASE, 0);
  motor(4, RELEASE, 0);
  delay(100);
}
void motor(int nMotor, int command, int speed) {
  int motorA, motorB;
  if (nMotor >= 1 && nMotor <= 4) {
  switch (nMotor)
{
  case 1:
  motorA = MOTOR1_A;
  motorB = MOTOR1_B;
  break;
  case 2:
  motorA = MOTOR2_A;
  motorB = MOTOR2_B;
  break;
  case 3:
  motorA = MOTOR3_A;
  motorB = MOTOR3_B;
  break;
  case 4:
  motorA = MOTOR4_A;
  motorB = MOTOR4_B;
  break;
  default:
  break;
}
```

```
switch (command) {
  case FORWARD:
  motor_output (motorA, HIGH, speed);
  motor_output (motorB, LOW, -1); // -1: no PWM set
  break;
  case BACKWARD:
  motor_output (motorA, LOW, speed);
  motor_output (motorB, HIGH, -1); // -1: no PWM set
  break;
  case BRAKE:
  motor_output (motorA, LOW, 255); // 255: fully on.
  motor_output (motorB, LOW, -1); // -1: no PWM set
  break;
  case RELEASE:
  motor_output (motorA, LOW, 0); // 0: output floating.
  motor_output (motorB, LOW, -1); // -1: no PWM set
  break;
  default:
  break;
}
}
}
void motor_output (int output, int high_low, int speed) {
  int motorPWM;
switch (output){
  case MOTOR1_A:
  case MOTOR1_B:
  motorPWM = MOTOR1_PWM;
  break;
  case MOTOR2_A:
  case MOTOR2_B:
  motorPWM = MOTOR2_PWM;
  break;
  case MOTOR3_A:
  case MOTOR3_B:
  motorPWM = MOTOR3_PWM;
  break;
```

```
  case MOTOR4_A:
  case MOTOR4_B:
  motorPWM = MOTOR4_PWM;
  break;
  default:
  speed = -3333;
  break;
}
if (speed != -3333) {
  shiftWrite(output, high_low); // set PWM only if it is valid
if (speed >= 0 && speed <= 255) {
  analogWrite(motorPWM, speed);
}
}
}
void shiftWrite(int output, int high_low) {
  static int latch_copy;
  static int shift_register_initialized = false;
// Do the initialization on the fly,
// at the first time it is used.
if (!shift_register_initialized) {
// Set pins for shift register to output
  pinMode(MOTORLATCH, OUTPUT);
  pinMode(MOTORENABLE, OUTPUT);
  pinMode(MOTORDATA, OUTPUT);
  pinMode(MOTORCLK, OUTPUT);
// Set pins for shift register to default value (low);
  digitalWrite(MOTORDATA, LOW);
  digitalWrite(MOTORLATCH, LOW);
  digitalWrite(MOTORCLK, LOW);
// Enable the shift register, set Enable pin Low.
  digitalWrite(MOTORENABLE, LOW);
// start with all outputs (of the shift register) low
  latch_copy = 0;
  shift_register_initialized = true;
}
// The defines HIGH and LOW are 1 and 0.
// So this is valid.
bitWrite(latch_copy, output, high_low);
shiftOut(MOTORDATA, MOTORCLK, MSBFIRST, latch_copy);
delayMicroseconds(5); // For safety, not really needed.
digitalWrite(MOTORLATCH, HIGH);
delayMicroseconds(5); // For safety, not really needed.
digitalWrite(MOTORLATCH, LOW);
}
```