Grove - I2C Motor Driver V1.1

|  |
| --- |
| **Contents**   [[hide](javascript:toggleToc())]   * [1 Introduction](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Introduction) * [2 Feature](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Feature) * [3 Specifications](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Specifications) * [4 Interface Function](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Interface_Function) * [5 Application Ideas](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Application_Ideas) * [6 Cautions](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Cautions) * [7 Usage](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Usage) * [8 Version Tracker](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Version_Tracker) * [9 Resources](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Resources) * [10 Support](http://www.seeedstudio.com/wiki/Grove_-_I2C_Motor_Driver_V1.1#Support) |

Introduction

The Twig I2C motor driver is a new addition to the TWIG series with the same easy-to-use interface. Its heart is a dual channel H-bridge driver chip that can handle current up to 2A per channel, controlled by an Atmel ATmega8L which handles the I2C communication with for example an Arduino. Both motors can be driven simultaneously while set to a different speed and direction. It can power two brushed DC motors or one 4-wire two-phase stepper motor. It requires a 6V to 15V power supply to power the motor and has an onboard 5V voltage regulator which can power the I2C bus (selectable by jumper). All driver lines are diode protected from back EMF.

The easy software interface is not the only easy-to-use feature because the TWIG I2C motor driver is designed to get you up and running in notime. It features a LED for power and four LED's to indicate if and to which direction each motor is running. Screw terminals facilitate motor and power connections, and the GROVE system plug and I2C interface enables you to daisy-chain the driver with many other devices.

**Model:**[**ROB72212P**](http://www.seeedstudio.com/depot/twig-i2c-motor-driver-p-907.html)



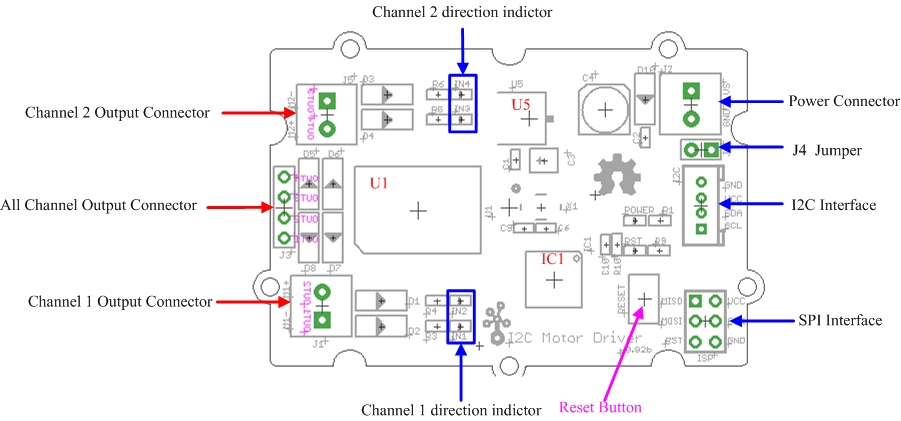
Feature

* Grove compatible
* I2C interface
* Motor's speed and direction can control
* Number of channels: 2
* Easy to use

Specifications

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Min** | **Typical** | **Max** | **Unit** |
| **Working Voltage** | 6 | - | 15 | VDC |
| **Max Output Current per channel** | 0.7 | | | A |
| **Input/output voltage on I2C bus** | 5 | | | V |
| **Communication protocol** | I2C | | | / |

Interface Function

[](http://www.seeedstudio.com/wiki/File:I2C_Motor_Driver_Interface1.jpg)

**U5:** AMS1117 IC,5v voltage regulator  
**U1:** L298 IC,dual full bridge driver   
**IC1**: Atmega8 IC, Control Motor Rotate.

**NOTE:** Input voltage on screw terminals is regulated to 5v and connected to I2C +5v via a jumper (J4). Remove jumper if both external power via the screw terminals and power via the I2C header is used. Use jumper if 5v should be supplied to the I2C bus.

Application Ideas

This motor driver can be used to drive any brushed electronic motor as long as it doesn't consume more than 2A at 5v. Two motors can be driven simultaneously while set to a different speed and direction. The speed can be set fully proportional and is controlled by the ATmega8 on the board using PWM. It is set by I2C commands sent from e.g. an Arduino. It is perfect for applications like robots, homebuilt RC cars, case fans, high power LED illumination or any other project that involves proportional load control.

Cautions

* The board will be very hot if while operating over 1Amps. Do keep off your hands!
* Different Arduino IDE may have difference. I use arduino-0019 and it works fine, but when i use arduino - 0022, I need to add some delay() at the end of Wire.endTransmission().

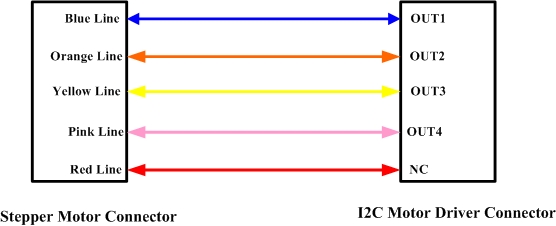
Usage

The I2C Motor Driver can control motor which is based on the chip L298, The L298 isn’t just a dual motor driver, it is a dual H-bridge. An h-bridge is basically a specific setup of transistors that allow you to switch direction of current. So hooked up to a motor, that means you can have it spin in both directions, and with PWM input, you can use your arduino to make them spin at any speed. Because the L298 has 2 H-bridges, you can not only make a robot go forwards and backwards, but also turn around by having each wheel spin in a different direction.  
Now, let's use the I2C Motor Driver to control two motors rotating in the positive or opposite direction.   
(图片)  
**Note:**

The first thing to notice however, is that you do need an external power source for your motors, the 5v pin on the arduino just can not source enough power to drive 2 motors, and you could damage your arduino if you do.

It also should be reminded that you need burn firmware for Atmega8 using ISP if you have not done.

When control step motor, To take 24BYJ48 for an example, Connection method as show blow:

[](http://www.seeedstudio.com/wiki/File:I2C_Motor_Driver_Connector.jpg)

The below is an example program to be used with an Arduino. The code for this is very basic. We created a function for you that makes controlling the L298 from your arduino easier, but you can also change it up and do it your own way.

/\*

Grove- i2C motor driver demo v1.0

by: http://www.seeedstudio.com

\*/

// Author:LG

#include <Wire.h>

#define MotorSpeedSet 0x82

#define PWMFrequenceSet 0x84

#define DirectionSet 0xaa

#define MotorSetA 0xa1

#define MotorSetB 0xa5

#define Nothing 0x01

#define EnableStepper 0x1a

#define UnenableStepper 0x1b

#define Stepernu 0x1c

#define I2CMotorDriverAdd 0x0f // Set the address of the I2CMotorDriver

// set the steps you want, if 255, the stepper will rotate continuely;

void SteperStepset(unsigned char stepnu)

{

Wire.beginTransmission(I2CMotorDriverAdd); // transmit to device I2CMotorDriverAdd

Wire.write(Stepernu); // set pwm header

Wire.write(stepnu); // send pwma

Wire.write(Nothing); // send pwmb

Wire.endTransmission(); // stop transmitting

}

// enanble the ic motor driver to drive a 4-wire stepper. the i2c motor driver will

//driver a 4-wire with 8 polarity .

//direction: stepper direction ;

//motor speed: defines the time interval the i2C motor driver change it output to drive the stepper

// the actul interval time is : motorspeed \* 4ms. that is , when motor speed is 10, the interval time

//would be 40 us

void StepperMotorEnable(unsigned char Direction, unsigned char motorspeed)

{

Wire.beginTransmission(I2CMotorDriverAdd); // transmit to device I2CMotorDriverAdd

Wire.write(EnableStepper); // set pwm header

Wire.write(Direction); // send pwma

Wire.write(motorspeed); // send pwmb

Wire.endTransmission(); // stop transmitting

}

//function to uneanble i2C motor drive to drive the stepper.

void StepperMotorUnenable()

{

Wire.beginTransmission(I2CMotorDriverAdd); // transmit to device I2CMotorDriverAdd

Wire.write(UnenableStepper); // set unenable commmand

Wire.write(Nothing);

Wire.write(Nothing);

Wire.endTransmission(); // stop transmitting

}

void MotorSpeedSetAB(unsigned char MotorSpeedA , unsigned char MotorSpeedB) {

MotorSpeedA=map(MotorSpeedA,0,100,0,255);

MotorSpeedB=map(MotorSpeedB,0,100,0,255);

Wire.beginTransmission(I2CMotorDriverAdd); // transmit to device I2CMotorDriverAdd

Wire.write(MotorSpeedSet); // set pwm header

Wire.write(MotorSpeedA); // send pwma

Wire.write(MotorSpeedB); // send pwmb

Wire.endTransmission(); // stop transmitting

}

void MotorPWMFrequenceSet(unsigned char Frequence) {

Wire.beginTransmission(I2CMotorDriverAdd); // transmit to device I2CMotorDriverAdd

Wire.write(PWMFrequenceSet); // set frequence header

Wire.write(Frequence); // send frequence

Wire.write(Nothing); // need to send this byte as the third byte(no meaning)

Wire.endTransmission(); // stop transmitting

}

void MotorDirectionSet(unsigned char Direction) { // Adjust the direction of the motors 0b0000 I4 I3 I2 I1

Wire.beginTransmission(I2CMotorDriverAdd); // transmit to device I2CMotorDriverAdd

Wire.write(DirectionSet); // Direction control header

Wire.write(Direction); // send direction control information

Wire.write(Nothing); // need to send this byte as the third byte(no meaning)

Wire.endTransmission(); // stop transmitting

}

void MotorDriectionAndSpeedSet(unsigned char Direction,unsigned char MotorSpeedA,unsigned char MotorSpeedB) { //you can adjust the driection and speed together

MotorDirectionSet(Direction);

MotorSpeedSetAB(MotorSpeedA,MotorSpeedB);

}

void stepperrun()

{

Serial.println("sent command to + direction, very fast");

SteperStepset(255);

StepperMotorEnable(1, 1);// ennable the i2c motor driver a stepper.

delay(5000);

Serial.println("sent command to - direction, slow");

SteperStepset(255);

StepperMotorEnable(0, 20);

delay(5000);

Serial.println("sent command to - direction, fast");

StepperMotorEnable(0, 2);// ennable the i2c motor driver a stepper.

delay(5000);

Serial.println("sent command to + direction,100 steps, fast");

SteperStepset(100);

StepperMotorEnable(1,5);

delay(3000);

Serial.println("sent command to shut down the stepper");

StepperMotorUnenable();

delay(1000);

Serial.println("sent command to - direction, slow, and 10 steps then stop");

SteperStepset(10);

StepperMotorEnable(0,40);

delay(5000);

Serial.println("sent command to shut down the stepper");

StepperMotorUnenable();

delay(5000);

// StepperMotorUnenable();// unenable the stepper

// delay(1000);

// StepperMotorEnable(1, 5);// ennable the i2c motor driver a stepper.

// delay(3000);

// StepperMotorEnable(0,10);

// delay(3000);

// StepperMotorUnenable();// unenable the stepper

// delay(5000);

}

void setup() {

Wire.begin(); // join i2c bus (address optional for master)

delayMicroseconds(10000);

Serial.begin(9600);

Serial.println("setup begin");

stepperrun();

}

void loop() {

// the following code sent commands to motor driver to drive DC motor

// while (1)

// {

// stepperrun();

// }

//

while(1) {

MotorSpeedSetAB(255,255);//defines the speed of motor 1 and motor 2;

delay(10); //this delay needed

MotorDirectionSet(0b1010); //"0b1010" defines the output polarity, "10" means the M+ is "positive" while the M- is "negtive"

// make sure M+ and M- is different polatity when driving DC motors.

delay(1000);

MotorDirectionSet(0b0101); //0b0101 Rotating in the opposite direction

delay(500);

}

}

Upload it to Arduino. Please click [here](http://www.seeedstudio.com/wiki/Upload_Code) if you do not know how to upload.  
Then the motors will rotating in the positive or opposite direction in cycle.  
**Note:** Each motor has 3 control pins, 2 for direction, and one for speed. When one direction pin is HIGH and the other is LOW the motor will spin one direction, flip them and it spins the other direction (both HIGH or both LOW and the motor stops). The PWM pin allows you to analogWrite to this pin to control the speed of that one motor. andlogWrite 0 and the motor stops, 255, and it will go full speed.

Version Tracker

|  |  |  |
| --- | --- | --- |
| **Revision** | **Descriptions** | **Release** |
| v0.9b | Initial public release | July 21th, 2011 |
| v1.1 | Initial public release | May 17th, 2012 |
| v1.2 | Initial public release | June 15th, 2012 |

Resources

* [L298 datasheet](http://www.st.com/stonline/books/pdf/docs/1773.pdf)
* [AMS1117 Datasheet](http://www.advanced-monolithic.com/pdf/ds1117.pdf)
* [I2C motor driver Eagle File](http://garden.seeedstudio.com/images/7/74/Eagle_files_of_I2C_motor_driver_v0.92b.zip)
* [File:Grove-I2C Motor Driver firmware.zip](http://www.seeedstudio.com/wiki/File:Grove-I2C_Motor_Driver_firmware.zip)

Support

If you have questions or other better design ideas, you can go to our [forum](http://www.seeedstudio.com/forum) or [wish](http://wish.seeedstudio.com/) to discuss.