SEED TECHNOLOGY INC (SEEEDUINO)
Grove - I2C Motor Driver
Model: ROB72212P

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

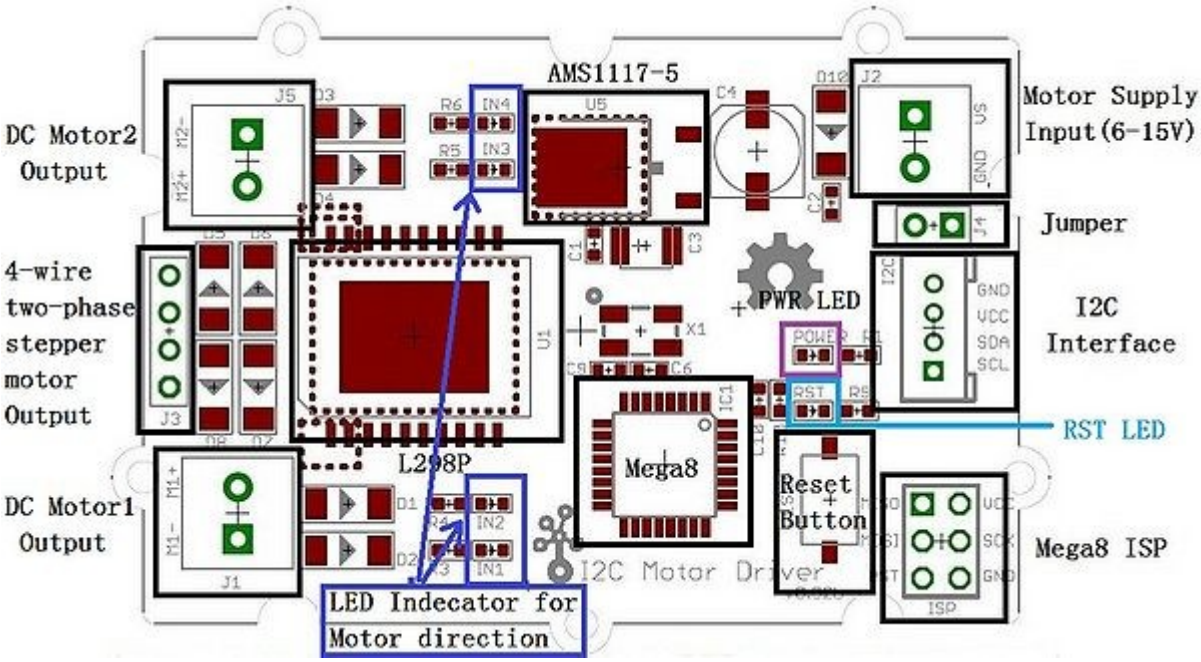
plug and I2C interface enables you to daisy-chain the driver with many other devices.

Specifications

- Default address: 0x28
- Grove compatible
- I2C interface
- Changable slave address
- Speed control: proportional 8-bit
- Number of channels: 2
- Maximum output per channel: 2A
- Input/output voltage on I2C bus[1]: 5v
- Input voltage on screw terminals: 6v-15v

[1] 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.

Function Block



Pin definitions

Pad Type	Pin Status	Description
GND	Input	Ground port
VS	Input	Motor Power supply(6-15V), also for 1117-5 regulator
SDA	Input and Output	I2C Serial Data(+5V Logic)
SCL	Input	I2C Serial Clock(+5V Logic)
VCC	Output or NC(not connected)	When the jumper is on, VCC as a +5V output for external MCU(Xduinos);NC while jumper is off.
M1+,M1-,M2+, M2-	Output	DC motor output relevent to VS

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

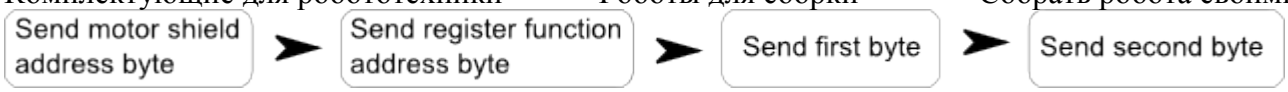
- The board will be very hot if while operating over 1Amps. Do keep off your hands!
- Different Arduino IDE may have difference. I use arduino-0019 and it works fine, but when i use arduino - 0022, I need to add some delay() at the end of Wire.endTransmission().

Schematic

Eagle files of I2C motor_driver.

Communication block diagram

This I2C motor driver communicates using I2C. The protocol is as follows: Address > register > arguments Two argument bytes always need to be sent, even if the second has no meaning (and is therefore 0). The motor address can be changed. The register addresses are described below.



Example: 0x28 > 0x82 > 255 > 255

Motor shield address > Set PWM AB > Motor A speed byte > Motor B speed byte

User register summary

Address	Function	Argument 1, 1 byte	Argument 2, 1 byte
0x28	Motor shield standard address		
0x82	Set PWM AB	byte motor A speed	byte motor B speed
0x84	Set frequency	Prescaler	0 (need to send, no meaning)
0x83	Change address	New address	Save or not ('S' or 'N' char)
0xAA	Channel set, set IN1-IN4 of H-bridge chip, only for advanced users	Byte: 0 0 0 0 I4 I3 I2 I1	0 (need to send, no meaning)
0xA1	Motor 1 speed	1 Forward or 2 Back rotation	Motor 1 speed byte
0xA5	Motor 2 speed	1 Forward or 2 Back rotation	Motor 2 speed byte

Mechanic Dimensions

Hardware Installation

Programming

You need to use an I2C communication library. Wire.h can be used when working with Arduino. Then send the commands described above.

The below is an example program to be used with an Arduino. It includes all functions and functions can simply be copypasted in your own program.

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().

```
#include <Wire.h>

//#define MOTORSHIELDaddr   0x0f
#define SETPWMAB          0x82
#define SETFREQ           0x84
#define CHANGEADDR        0x83
#define CHANNELSET        0xaa
#define MOTOR1            0xa1
#define MOTOR2            0xa5
#define SAVEADDR          'S'
#define NOTSAVEADDR       'N'

static unsigned char MOTORSHIELDaddr = 0x28;

void speedAB(unsigned char spda , unsigned char spdb)
{
  Wire.beginTransmission(MOTORSHIELDaddr); // transmit to device MOTORSHIELDaddr
  Wire.send(SETPWMAB);        //set pwm header
  Wire.send(spda);              // send pwma
  Wire.send(spdb);              // send pwmb
```

```
  Wire.endTransmission();    // stop transmitting
}

void fre_pre(unsigned char pres)
{
  Wire.beginTransmission(MOTORSHIELDaddr); // transmit to device MOTORSHIELDaddr
  Wire.send(SETFREQ);        // set frequence header
  Wire.send(pres);              //  send prescale
  Wire.send(0);              //  need to send this byte as the third byte(no meaning)
  Wire.endTransmission();    //
}

void change_adr(unsigned char new_adr, unsigned char save_or_not)
{
  Wire.beginTransmission(MOTORSHIELDaddr); // transmit to device MOTORSHIELDaddr
  Wire.send(CHANGEADDR);        // change address header
  Wire.send(new_adr);              //  send new address
  Wire.send(save_or_not);              //   save the new address or not
  Wire.endTransmission();    //
  delayMicroseconds(100); //this command needs at least 6 us
}

void channel(unsigned char i4) //0b 0000 I4 I3 I2 I1
{
//  delayMicroseconds(4);
  Wire.beginTransmission(MOTORSHIELDaddr); // transmit to device MOTORSHIELDaddr
  Wire.send(CHANNELSET);        // channel control header
  Wire.send(i4);              // send channel control information
  Wire.send(0);              // need to send this byte as the third byte(no meaning)
  Wire.endTransmission();    //
}
void motorAndspd( unsigned char motor_s,unsigned char Mstatus, unsigned char spd)
{
  Wire.beginTransmission(MOTORSHIELDaddr); // transmit to device MOTORSHIELDaddr
  Wire.send(motor_s);        // motor select information
  Wire.send(Mstatus);        // motor satus information
  Wire.send(spd);            //  motor speed information
  Wire.endTransmission();    //
}

void setup()
{
  Wire.begin(); // join i2c bus (address optional for master)
  delayMicroseconds(10); //wait for motor driver to initialization
}

void loop()
{

  motorAndspd(0xa1,0b01,225);
  motorAndspd(0xa5,0b01,30);
  delay(2000);

  motorAndspd(0xa5,0b10,225);
  motorAndspd(0xa1,0b10,30);
  delay(2000);

  change_adr(0x12, NOTSAVEADDR );
  MOTORSHIELDaddr = 0x12;

  for( int i = 0;i< 16;i++)
  {
    fre_pre(i);
    channel(i);
    delay(100);
  }
  speedAB(0 , 0);
  change_adr(0x28,NOTSAVEADDR );
  MOTORSHIELDaddr = 0x28;
```

```
  while(1)
  {
    channel(0b00001010);
    delay(500);
    channel(0b00000101);
    delay(500);
  }
}
```

Example

The projects and application examples.

Bill of Materials (BOM) /parts list

Part	Value	Device	Package	Description
C1	100nf	006-C30*35	0603_S	Ceramic Capacitors
C2	100nf	006-C30*35	0603_S	Ceramic Capacitors
C3	10UF	C_TAN_B	C_TAN_3X3.5	
C4	100U	PANASONIC_D260	SANYO-OSCON_SMD_C6	
C6	22PF	006-C30*35	0603_S	Ceramic Capacitors
C9	22PF	006-C30*35	0603_S	Ceramic Capacitors
C10	100nF	006-C30*35	0603_S	Ceramic Capacitors
D1	IN4007	010-D2010	D2010	
D2	IN4007	010-D2010	D2010	
D3	IN4007	010-D2010	D2010	
D4	IN4007	010-D2010	D2010	
D5	IN4007	010-D2010	D2010	
D6	IN4007	010-D2010	D2010	
D7	IN4007	010-D2010	D2010	
D8	IN4007	010-D2010	D2010	
D10	m4	010-D2010	D2010	
I2C	TWIG_2.0	TWIG_2.0	2.0_1X4	
IC1	MEGA8-AI	MEGA8-AI	TQFP32-08	MICROCONTROLLER
IN1	red	007-LED0603	D0603	
IN2	green	007-LED0603	D0603	
IN3	red	007-LED0603	D0603	
IN4	green	007-LED0603	D0603	
ISP	AVR-ISP-6	AVR-ISP-6	AVR-ISP-6	AVR ISP-6 Serial Programming Header
J1	CON2	CON2	CON2	
J2	CON2	CON2	CON2	
J3	2.54_1X4P_DD	2.54_1X4P_DD	CK_1X4	
J4	HEADER_1X2_2.54_2.54	HEADER_1X2_2.54_2.54	65*35	
J5	CON2	CON2	CON2	
LOGO2	LOGO-OSHW-FILLX0200-NT	LOGO-OSHW-FILLX0200-NT	OSHW_FILLX200_NOTEXT	

POWER	Green	007-LED0603	D0603	
R1	1k	005-R0603	0603-1	
R3	1k	005-R0603	0603-1	
R4	1k	005-R0603	0603-1	
R5	1k	005-R0603	0603-1	
R6	1k	005-R0603	0603-1	
R9	1k	005-R0603	0603-1	
R10	10K	005-R0603	0603-1	
RESET	008-BOTTON_2P_SMD	008-BOTTON_2P_SMD	BOTTON_1	
RST	Red	007-LED0603	D0603	
U$1	U23N	U23N	U2*3N	
U$3	TWIGLOGOTWIGLOGO_SMALL	TWIGLOGOTWIGLOGO_SMALL	TWIGLOGO_SMALL	
U1	L298N	L298_BRIDGE_DRIVER	POWERSO20	
U5	1117-5	1117-5	TO252	
X1	011-XTAL_SMD-4P_8M	011-XTAL_SMD-4P	CRYSTAL-SMD-5X3	

FAQ

Please list your question here:

Support

If you have questions or other better design ideas, you can go to our forum or wish to discuss.

Version Tracker

Revision	Descriptions	Release
v0.9b	Initial public release	date

Bug Tracker

Bug Tracker is the place you can publish any bugs you think you might have found during use. Please write down what you have to say, your answers will help us improve our products.

Additional Idea

The Additional Idea is the place to write your project ideas about this product, or other usages you've found. Or you can write them on Projects page.

Resources

L298 dual full bridge driver - http://www.st.com/stonline/books/pdf/docs/1773.pdf

AMS 1117CD 5v voltage regulator - http://www.advanced-monolithic.com/pdf/ds1117.pdf

Eagle files of I2C motor_driver.

See Also

Other related products and resources.

Licensing

Relevant datasheets

L298 dual full bridge driver - http://www.st.com/stonline/books/pdf/docs/1773.pdf

AMS 1117CD 5v voltage regulator - http://www.advanced-monolithic.com/pdf/ds1117.pdf