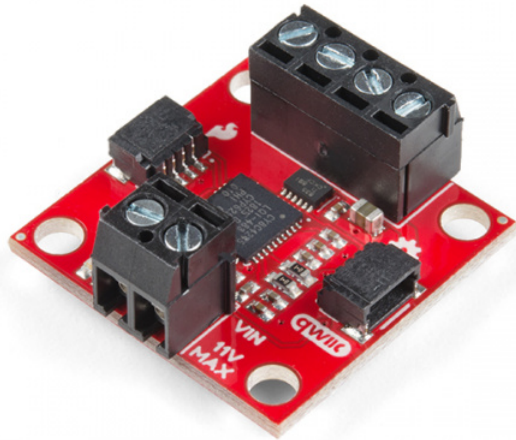


Hookup Guide for the Qwiic Motor Driver

Introduction

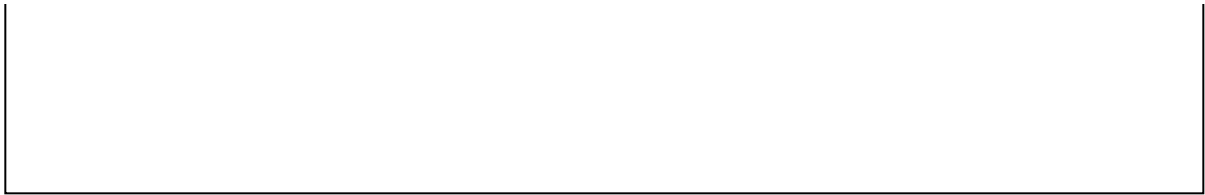
The Qwiic Motor Driver takes all the great features of the Serial Controlled Motor Driver and mini-sizes them, adding Qwiic ports for plug and play functionality. Boasting the same PSOC and 2-channel motor ports, the QWIIC Motor Driver is designed to communicate over I²C, but UART is also available.



SparkFun Qwiic Motor Driver
● ROB-15451

Product Showcase: SparkFun Qwiic Motor Driver





Required Materials

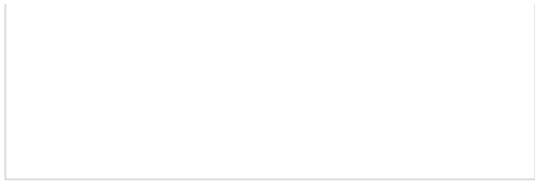
To follow along with this tutorial, you will need the following materials. You may not need everything though depending on what you have. Add it to your cart, read through the guide, and adjust the cart as necessary.

Qwiic Motor Driver Wish List SparkFun Wish List	
	USB Mini-B Cable - 6" CAB-13243 This is a USB 2.0 type B to Mini-B 5-pin black cable. You know, the mini-B connector that usually comes with USB ...
	SparkFun FTDI Basic Breakout - 3.3V DEV-09873 This is the newest revision of our [FTDI Basic](http://www.sparkfun.com/commerce/product_info.php?products_id=...
	SparkFun Qwiic Motor Driver ROB-15451
	SparkFun RedBoard Qwiic DEV-15123
	Hobby Gearmotor - 140 RPM (Pair) ROB-13302 These are a pair of hobby gearmotors from DAGU. These gearmotors are the same ones recommended for use in ...
	Qwiic Cable - 100mm PRT-14427
	SparkFun Traveler microB Cable - 1m CAB-14741
	Jumper Wires Premium 6" M/M - 20 AWG (10 Pack) PRT-11709 Jumper wires are awesome. Just a little bit of stranded core wire with a nice solid pin connector on either end. The...
	Lithium Ion Battery - 1Ah PRT-13813 These are very slim, extremely light weight batteries based on Lithium Ion chemistry. Each cell outputs a nominal 3...

Suggested Reading

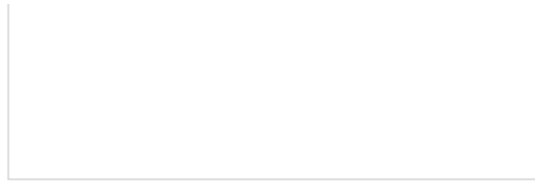
If you aren't familiar with the following concepts, we recommend you read over these tutorials before continuing.





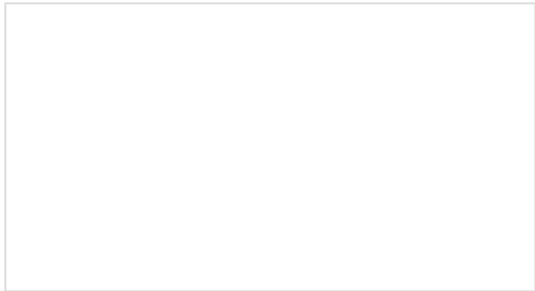
Serial Communication

Asynchronous serial communication concepts: packets, signal levels, baud rates, UARTs and more!



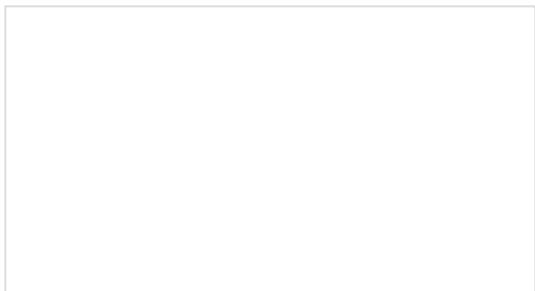
I2C

An introduction to I2C, one of the main embedded communications protocols in use today.



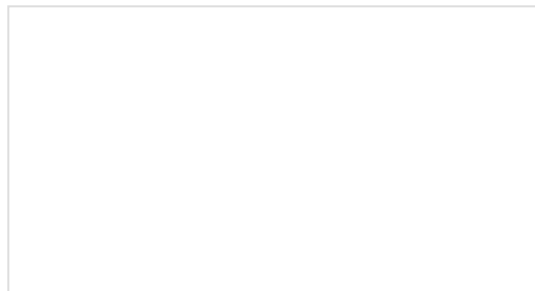
Motors and Selecting the Right One

Learn all about different kinds of motors and how they operate.



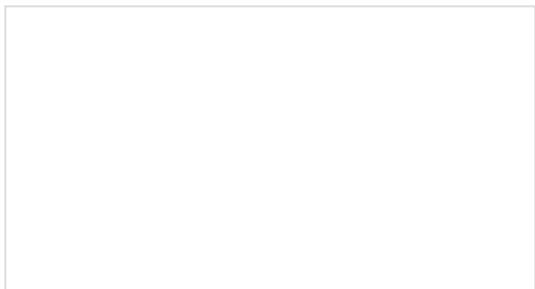
Logic Levels

Learn the difference between 3.3V and 5V devices and logic levels.



Bi-Directional Logic Level Converter Hookup Guide

An overview of the Bi-Directional Logic Level Converter, and some example circuits to show how it works.



Serial Controlled Motor Driver Hookup Guide

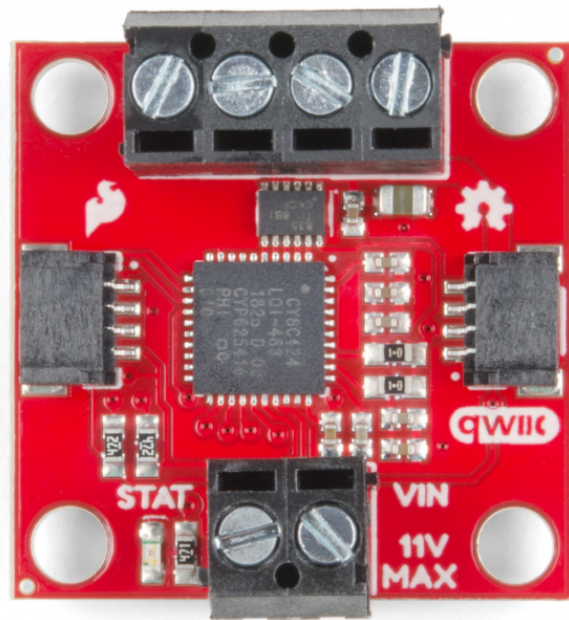
Hookup guide for the Serial Controlled Motor Driver

Hardware Overview

Let's look at some of the various features of the hardware.

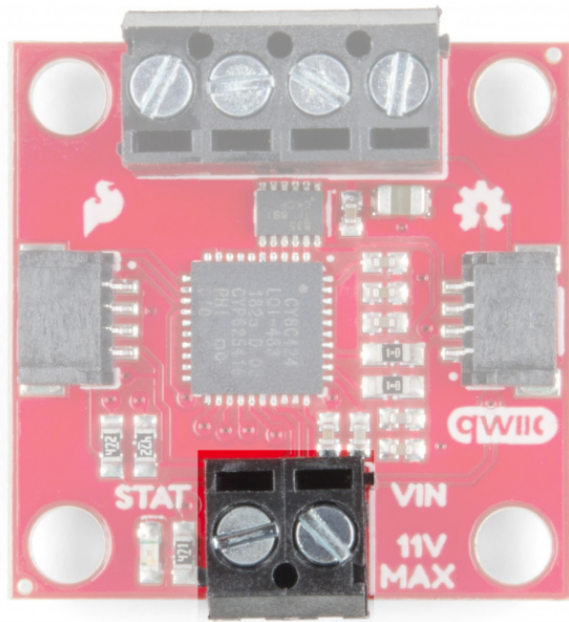
Features:

- 1.5 A peak drive per channel, 1.2 A steady state
- Operates from **3 to 11 volts** with 12v absolute max
- **3.3V** default VCC and logic
- 127 levels of DC drive strength.
- Controllable by I²C or TTL UART signals
- Direction inversion on a per motor basis
- Global Drive enable
- Exposed small heat sink shape
- Several I²C addresses, default UART bauds available



Power

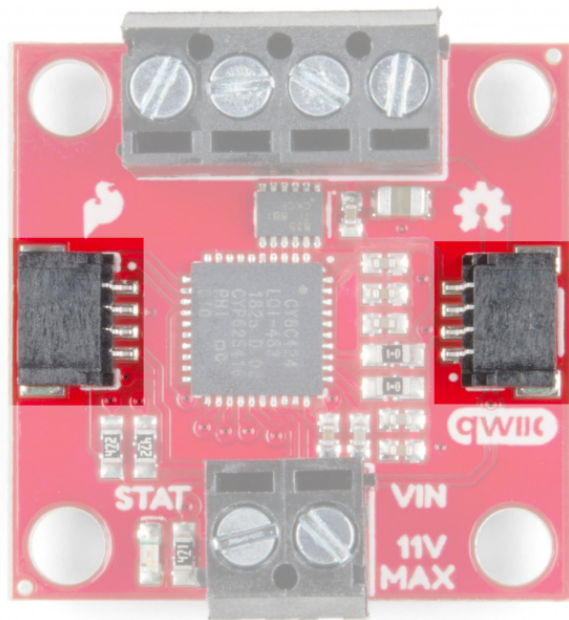
There are two separate power circuits on this board . Power for the motors is supplied through the **VIN Connectors** - you can provide anywhere from **3.3V to 11V** to the "MAX 11V" and "GND" connections. Power for the PSOC and logic circuits is provided by the 3.3V inputs on the Qwiic connectors. Both are needed for proper functioning.



Motor Power Ports: Ground (Left) - VIN(Right)

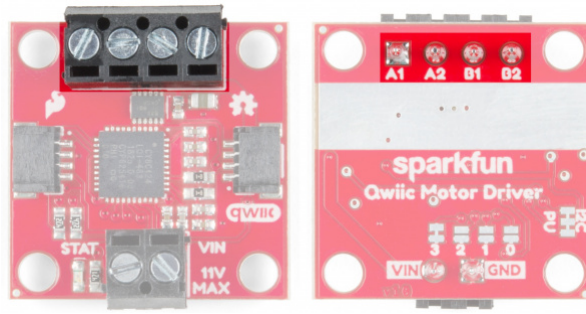
Qwiic Connectors

There are two Qwiic connectors on the board such that you can provide power or daisy-chain the boards should you choose to do so. If you're unfamiliar with our Qwiic system, head on over to our Qwiic page to see the advantages!



Motor Ports

The screw pin terminals at the top of the board allow for two motor connections. They are labeled on the backside of the board.



From left to right on the front: B2-B1-A2-A1

				Function / Connection	
Group	Name	Direction	Description	UART	I ² C
Motor Port	A1	O	Winding of first addressable location	Motor A winding	
	A2	O	Winding of first addressable location	Motor A winding	
	B1	O	Winding of second addressable location	Motor B winding	
	B2	O	Winding of second addressable location	Motor B winding	

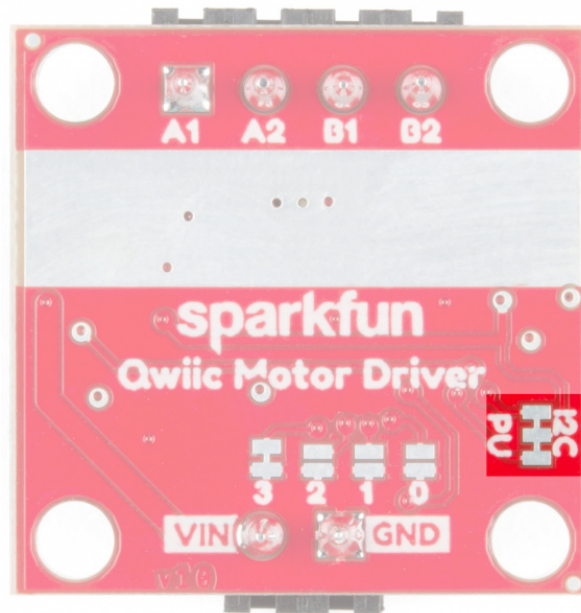
Jumpers

Jumper Usage Table

There are 2 sets of jumpers to configure on this board. There are pull-up enables for I²C and 4 **config** bits that select operational mode.

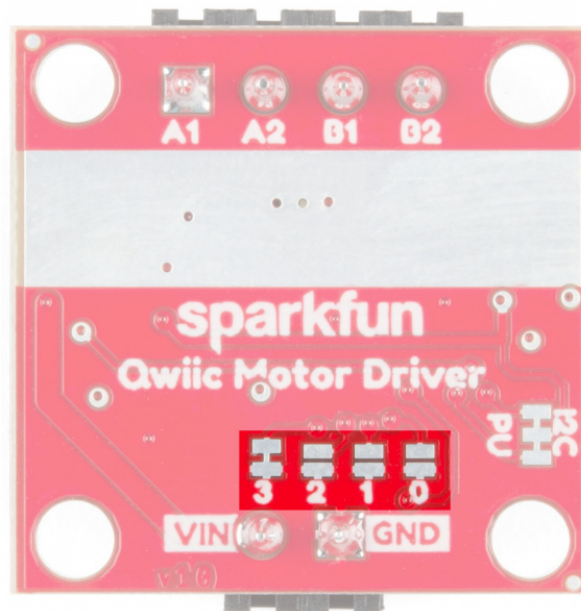
Name	Description	Usage
I ² C Jumpers	I ² C pull-up enable	Opening these disables the I ² C pull-up resistors used for I ² C communication. If multiple I ² C devices are being used, these pull-ups should be disabled on all but one device. If UART is being used, the pull-up resistors should be disabled.
Address Jumpers	Serial and function selection	The config bits are 4 bits that form a configuration nybble. A closed jumper is a '1' and an open jumper is a '0'. See config table for more information.

I²C Pull-Up Jumpers



Address Bits

The configuration is set by encoding a number into the 4 config bits on the bottom of the board. Close a jumper to indicate a 1, or leave it open to indicate a 0.



Use this table to see what the user port, address, and expansion port will become in each configuration:

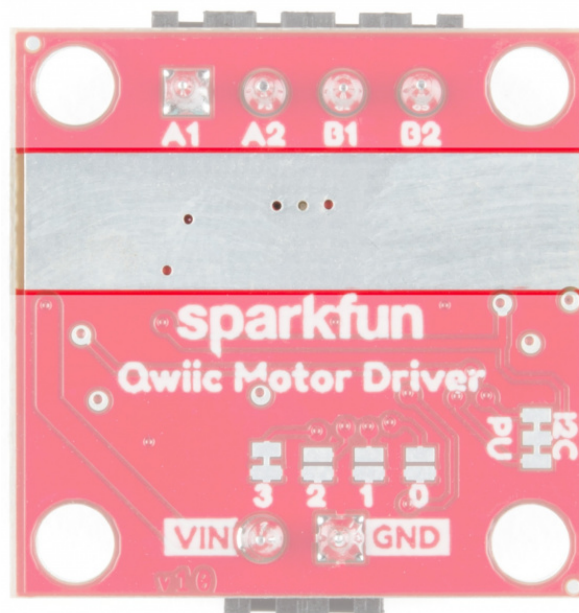
Pattern	Mode	User Port	User Address	Expansion Port
0000	UART at 9600	UART	N/A	Master
0011	I ² C	I ² C	0x58	Master
0100	I ² C	I ² C	0x59	Master
0101	I ² C	I ² C	0x5A	Master
0110	I ² C	I ² C	0x5B	Master

0111	I ² C	I ² C	0x5C	Master
1000	I²C	I²C	0x5D	Master
1001	I ² C	I ² C	0x5E	Master
1010	I ² C	I ² C	0x5F	Master
1011	I ² C	I ² C	0x60	Master
1100	I ² C	I ² C	0x61	Master
1101	UART at 57600	UART	N/A	Master
1110	UART at 115200	UART	N/A	Master
1111	N/A	Reserved	N/A	N/A

Bold text is the default setting for the Qwiic Motor Driver

Thermal Conduction Area

The Qwiic Motor Driver is designed to operate small robot drive motors without a heatsink; we were able to run up to about 1.1A continuous current without going above 100°C. If you find that you need a heatsink, you can use our Theragrip Thermal Tape to attach three Small Heat Sinks across the thermal conduction area on the back of the board.



If you need more information on how to determine whether or not you need a heat sink, kick on over to the [Serial Controlled Motor Driver Hookup Guide](#) and scroll down to **Typical Application Motors and Heat Sinking**.



Installing Arduino IDE

MARCH 26, 2013

A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.

Getting the Arduino Library

The Qwiic Motor Driver uses the same Arduino Library as the Serial Controlled Motor Driver (hereafter referred to as SCMD). To get the Arduino library, either download and install it from Github or use the Arduino Library Manager.

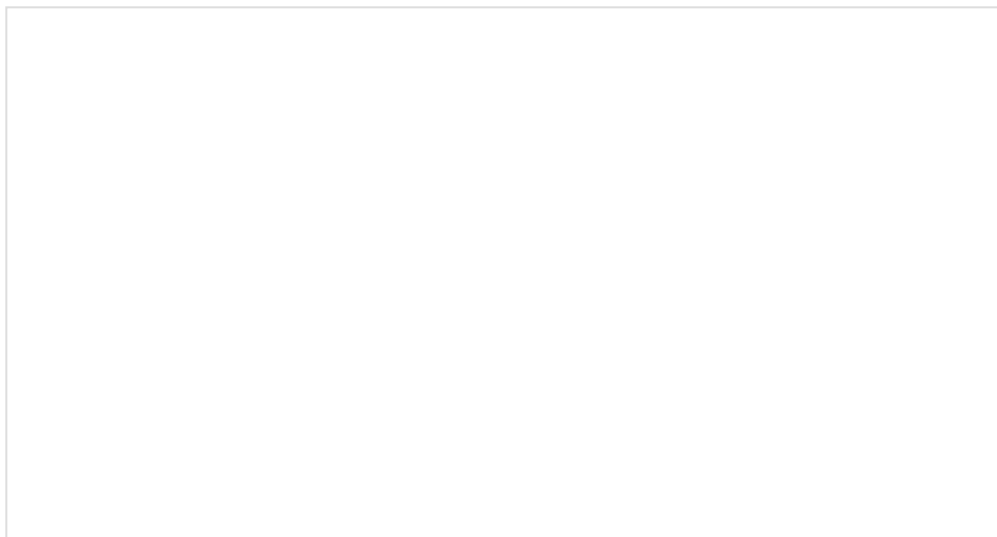
Download the Github repository

Visit the GitHub repository to download the most recent version of the library, or click the link below:

[DOWNLOAD THE ARDUINO LIBRARY](#)

Use the library manager or install in the Arduino IDE

In the Library Manager, search for ***Serial Controlled Motor Driver***. For help installing the library, check out our [How To Install An Arduino Library](#) tutorial.



Installing an Arduino Library

JANUARY 11, 2013

How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not

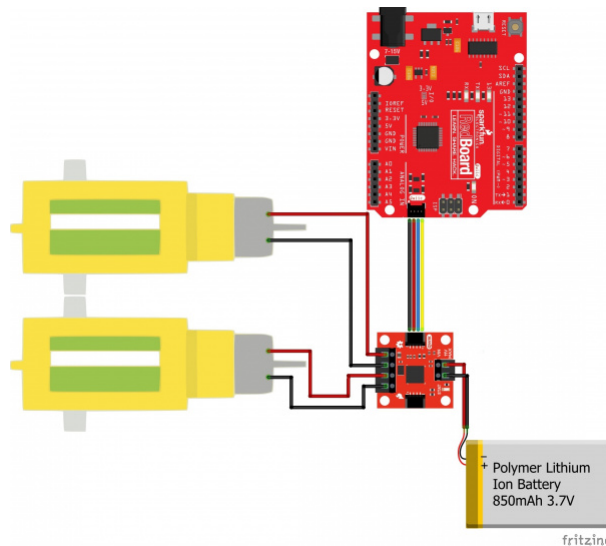
linked with the Arduino IDE, we will also go over manually installing an Arduino library.

Experiment 1: Testing the Motors

Let's start by hooking up some motors and making sure they're running. Since the Qwiic Motor Driver uses the same PSOC as the Serial Controlled Motor Driver, the same examples will work with minor modifications.

✂ **Note:** In lieu of using the external LiPo battery, it is also possible to use the **5V** and GND pins from the RedBoard Qwiic.

Hardware Hookup



Click the image for a closer look

Testing the Motors

The following test is essentially the *TwoMotorRobot.ino* example from the SCMD library, but with a few minor changes to account for the defaults of the Qwiic Motor Driver.

Copy and paste the following code into your Arduino browser and upload.

```

//This example drives a robot in left and right arcs, driving in an overall wiggly course.
// It demonstrates the variable control abilities. When used with a RedBot chassis,
// each turn is about 90 degrees per drive.
//
// Pin 8 can be grounded to disable motor movement, for debugging.

#include <Arduino.h>
#include <stdint.h>
#include "SCMD.h"
#include "SCMD_config.h" //Contains #defines for common SCMD register names and values
#include "Wire.h"

SCMD myMotorDriver; //This creates the main object of one motor driver and connected slaves.

void setup()
{
    pinMode(8, INPUT_PULLUP); //Use to halt motor movement (ground)

    Serial.begin(9600);
    Serial.println("Starting sketch.");

    //***** Configure the Motor Driver's Settings *****/
    // .commInter face is I2C_MODE
    myMotorDriver.settings.commInterface = I2C_MODE;

    // set address if I2C configuration selected with the config jumpers
    myMotorDriver.settings.I2CAddress = 0x5D; //config pattern is "1000" (default) on board for address 0x5D

    // set chip select if SPI selected with the config jumpers
    myMotorDriver.settings.chipSelectPin = 10;

    //*****initialize the driver get wait for idle*****/
    while ( myMotorDriver.begin() != 0xA9 ) //Wait until a valid ID word is returned
    {
        Serial.println( "ID mismatch, trying again" );
        delay(500);
    }
    Serial.println( "ID matches 0xA9" );

    // Check to make sure the driver is done looking for slaves before beginning
    Serial.print("Waiting for enumeration...");
    while ( myMotorDriver.ready() == false );
    Serial.println("Done.");
    Serial.println();

    //*****Set application settings and enable driver*****/

    //Uncomment code for motor 0 inversion
    //while( myMotorDriver.busy() );
    //myMotorDriver.inversionMode(0, 1); //invert motor 0

    //Uncomment code for motor 1 inversion

```

```

while ( myMotorDriver.busy() ); //Waits until the SCMD is available.
myMotorDriver.inversionMode(1, 1); //invert motor 1

while ( myMotorDriver.busy() );
myMotorDriver.enable(); //Enables the output driver hardware
}

#define LEFT_MOTOR 0
#define RIGHT_MOTOR 1
void loop()
{
    //pass setDrive() a motor number, direction as 0(call 0 forward) or 1, and level from 0 to 255
    myMotorDriver.setDrive( LEFT_MOTOR, 0, 0); //Stop motor
    myMotorDriver.setDrive( RIGHT_MOTOR, 0, 0); //Stop motor
    while (digitalRead(8) == 0); //Hold if jumper is placed between pin 8 and ground

    //***** Operate the Motor Driver *****//
    // This walks through all 34 motor positions driving them forward and back.
    // It uses .setDrive( motorNum, direction, level ) to drive the motors.

    //Smoothly move one motor up to speed and back (drive level 0 to 255)
    for (int i = 0; i < 256; i++)
    {
        myMotorDriver.setDrive( LEFT_MOTOR, 0, i);
        myMotorDriver.setDrive( RIGHT_MOTOR, 0, 20 + (i / 2));
        delay(5);
    }
    for (int i = 255; i >= 0; i--)
    {
        myMotorDriver.setDrive( LEFT_MOTOR, 0, i);
        myMotorDriver.setDrive( RIGHT_MOTOR, 0, 20 + (i / 2));
        delay(5);
    }
    //Smoothly move the other motor up to speed and back
    for (int i = 0; i < 256; i++)
    {
        myMotorDriver.setDrive( LEFT_MOTOR, 0, 20 + (i / 2));
        myMotorDriver.setDrive( RIGHT_MOTOR, 0, i);
        delay(5);
    }
    for (int i = 255; i >= 0; i--)
    {
        myMotorDriver.setDrive( LEFT_MOTOR, 0, 20 + (i / 2));
        myMotorDriver.setDrive( RIGHT_MOTOR, 0, i);
        delay(5);
    }
}
}

```

What You Should See

The code goes through and establishes communication with the motor driver and then runs each motor in arcs, resulting in a "wiggly pattern".

Things to note:

- `Serial.begin` is periodically run until the returned ID word is valid.
- Setup waits for `isReady()` to become true before going on to the drive section
- One motor is inverted by command at setup. Do it here so you don't have to mess with it later.
- `enable()` is called to connect the drivers to the PWM generators.
- `LEFT_MOTOR` and `RIGHT_MOTOR` are defined to ease use of the `setDrive( ... )` function.

See the Arduino Library Reference section of the Serial Controlled Motor Driver Hookup Guide for more information on the functions defined in the Arduino library.

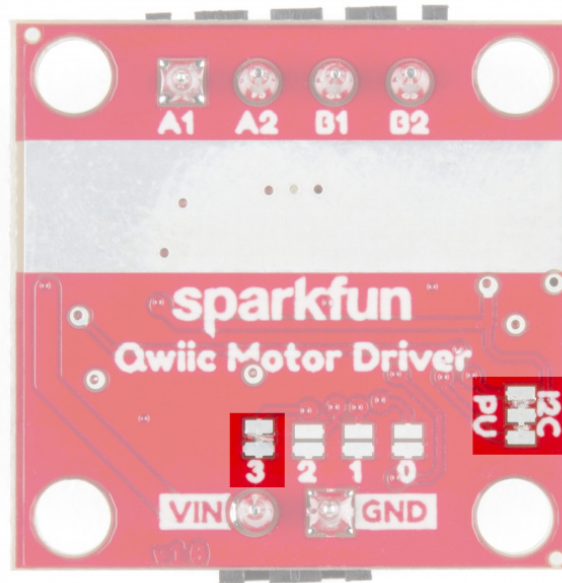
Experiment 2: Interactive Commands with UART

This example demonstrates the basic commands, plus some direct register access possible with only a UART available. This type of program could be easily run from a script from a more classic PC where I²C isn't available.

Interactive UART

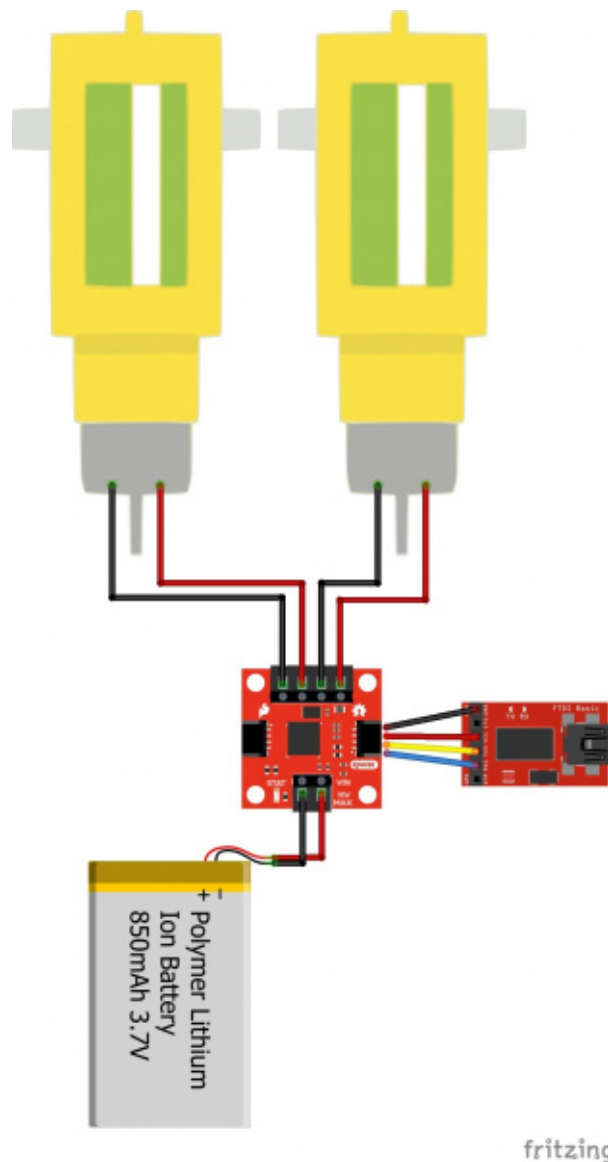
Requirements

- Computer serial terminal set to 9600 baud.
- Terminal set to send CR and LF (Carriage return and line feed).
- Config jumpers set to '0000', or all open.
- I²C PU jumper fully open
- FTDI Basic or Serial Basic - either will work but ensure you have the **3.3V** version



Make sure the Address Jumper 3 and I²C Pullup Jumpers are cut as you see here.

Connect the FTDI to the Qwiic Motor Driver as you see in the Fritzing diagram below. Attach two motors to the driver, one between A1 and A2, and the other between B1 and B2.



Click the image for a closer look

Example Commands

When you're ready, make sure you have the correct COM port selected in your Arduino IDE, open a Serial Monitor, and send the following commands:

"R01"

This will read the ID register and return 0xA9

"M0F50"

This will tell motor 0 to drive at half speed, forward -- But nothing will happen yet!

"E"

This will enable all drivers. Motor 0 should begin spinning at half speed.

"M1R100"

This will tell motor 1 to drive at full speed in reverse. Now both should be spinning opposite directions.

"D"

D will disable both motors, which will stop spinning.

See the section UART Commands in the Serial Controlled Motor Driver Hookup Guide for a full command listing.

Troubleshooting

Need help?

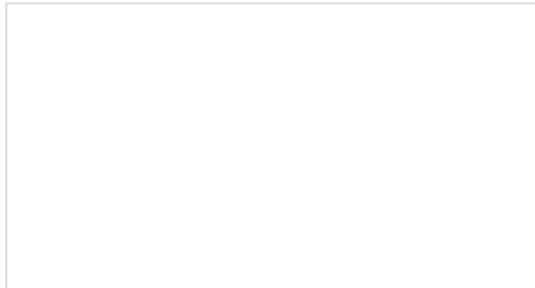
If your product is not working as you expected or you need technical assistance or information, head on over to the SparkFun Technical Assistance page for some initial troubleshooting.

If you don't find what you need there, the SparkFun Forums are a great place to find and ask for help. If this is your first visit, you'll need to create a Forum Account to search product forums and post questions.

Resources and Going Further

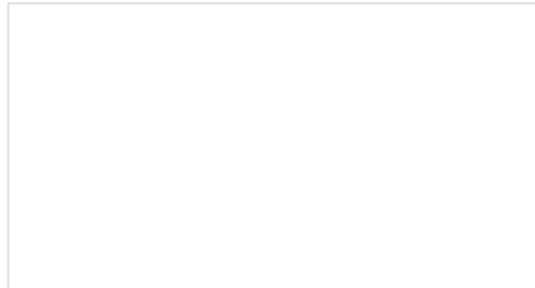
- Schematic (PDF)
- Eagle Files (ZIP)
- Github (Hardware)
- Github (Arduino Library)
- Serial Controlled Motor Driver DataSheet (PDF)
- Python Library GitHub
- Qwiic Motor Driver Board Dimensions

Need inspiration? Check out some of the Qwiic or motor related tutorials!



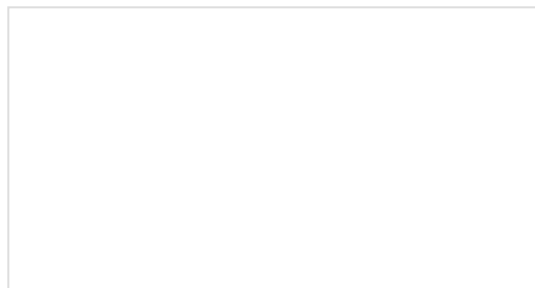
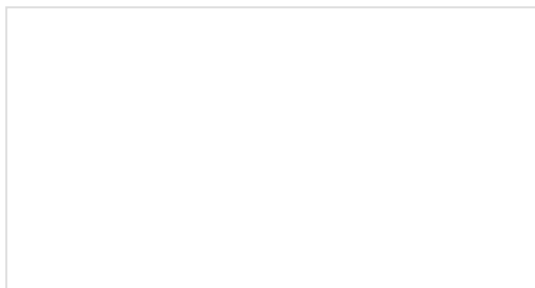
SparkFun LoRa Gateway 1-Channel Hookup Guide

How to setup and use the LoRa Gateway 1-Channel in Arduino.



Using SparkFun Edge Board with Ambiq Apollo3 SDK

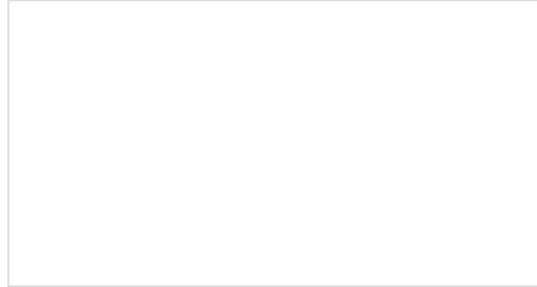
We will demonstrate how to get started with your SparkFun Edge Board by setting up the toolchain on your computer, examining an example program, and using the serial uploader tool to flash the chip.



Build a Qwiic Jukebox that is Toddler Approved!

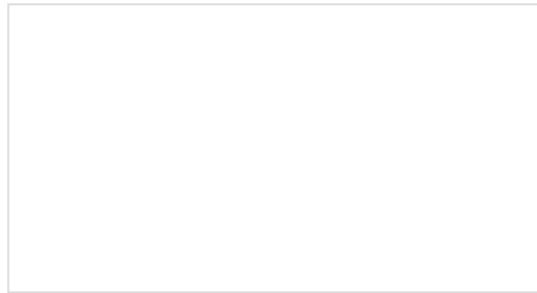
Follow this tutorial to build your own custom jukebox.

Note, this is designed simple and tough for use primarily with toddlers. It's also a great introduction to SparkFun's Qwiic products!



SparkFun Inventor's Kit for micro:bit Experiment Guide

This guide contains all the information you will need to explore the twelve circuits of the SparkFun Inventors Kit for micro:bit.

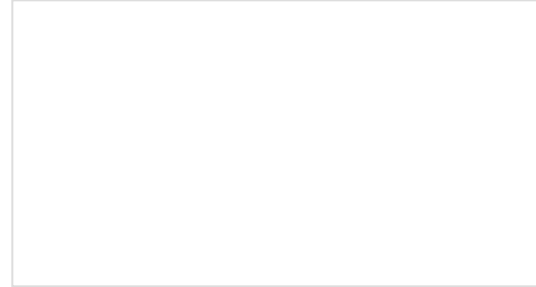


ReconBot with the Tessel 2

Build a robot with the Tessel 2 that you can control from a browser on your phone or laptop.

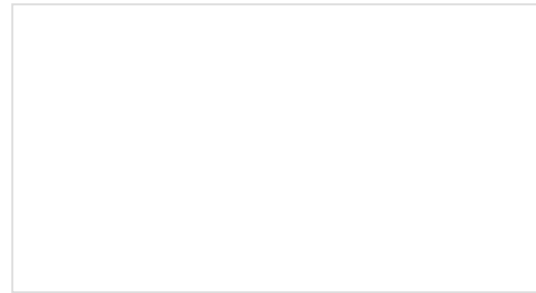
Capacitive Touch Slider (CAP1203) Hookup Guide

An easy and Qwiic way to add capacitive touch to any of your projects using the CAP1203! In this guide, we go over how to connect and set up your Capacitive Touch Slider so you can start playing with it right away.



How to Build a Remote Kill Switch

Learn how to build a wireless controller to kill power when things go... sentient.



Activity Guide for SparkFun Tinker Kit

This activity guide will take you through the basics of building 11 different circuits with the SparkFun Tinker Kit and how to program them using the Arduino IDE.