



Hookup Guide - TMC6300 BLDC Motor Driver

none

Table of contents

1. Getting Started	4
1.1 Introduction	4
1.2 Hardware Overview	12
1.3 Hardware Assembly	22
1.4 Software - Arduino IDE	30
2. Resources	45
2.1 Hardware Resources	45
2.2 Background Information	47
3. Support	49
3.1 Troubleshooting Tips	49

1. Getting Started

1.1 Introduction

Brushless Motor Driver - 3 Phase (TMC6300)

SKU: ROB-21867



The TMC6300 from [ADI + Trinamic](#) is a powerful and easy to use three phase motor driver that was designed to control our [Brushless Gimbal Motor](#). However, it can be used to control any 3-phase [BLDC](#) or [PMSM](#) motor with up to 2A ($1.4A_{RMS}$) of total drive current. Separate high-side and low-side control of the three half-bridges allows for incredible control of each phase of the [motor commutation](#). The driver also provides temperature and short circuit protections; and a diagnostic output to indicate system faults. With a 1.8V regulated power output and an operating voltage down to 2V, the TMC6300 is suitable for low-power microcontroller and battery powered designs (min. 2 AA/NiMh cells, or 1-2 Li-Ion cells).

Our board layout has been designed with the LEDs and labels facing up, IC down. This allows the thermal pad on the board to be access if cooling is required. Additionally, the breakout pins are specially aligned to fit perfectly onto a breadboard and hold the headers more perpendicular to facilitate assembly.

Controlling 3-phase motors is not trivial and this board requires 6 PWM signals to fully control one motor. We've found the [Arduino Simple Field Oriented Control](#) library to work well with the board; however, there are some hardware limitations such as [supported microcontrollers for 6PWM Mode](#). With additional considerations, for integrating [position sensors](#) into the feedback control loop.



Purchase from SparkFun 

1.1.1 Required Materials

To get started, users will need a few items. Now some users may already have a few of these items, feel free to modify your cart accordingly.

- Computer with an operating system (OS) that is compatible with all the software installation requirements.
- A compatible microcontroller/Arduino board; we recommend the [SparkFun RedBoard Plus](#).

Warning

The recommended Arduino library for the TMC6300 motor driver is not compatible with all microcontrollers or boards. For a complete list of compatible microcontrollers and boards, please refer to the [documentation](#) for the Simple Field Oriented Control Library.

- [USB 3.1 Cable A to C - 3 Foot](#) - Used to interface with the RedBoard Plus (1)
 - a. If your computer doesn't have a USB-A slot or your microcontroller/Arduino board has a different USB connector, then choose an appropriate cable or adapter.
- [SparkFun Brushless Motor Driver - 3 Phase \(TMC6300\)](#)
- [BLDC Motor](#) (1)
 - a. This gimbal motor requires a 6 to 8V power supply. However, for zero-load, low-speed testing, we have found that users can get away with utilizing the 5V power from a RedBoard Plus.
- [Power Supply](#)
- [Soldering Tools](#) (1)
 - a. Check out the beginner tool kit below; otherwise, click here for a full selection of our available [soldering tools](#).
- [Headers](#)
- [Small Breadboard](#)
- [Jumper Wires](#)

Product Thumbnail



[USB 3.1 Cable A to C - 3 Foot](#)

CAB-14743



[SparkFun RedBoard Plus](#)

DEV-18158



[SparkFun Brushless Motor Driver - 3 Phase \(TMC6300\)](#)

ROB-21867

[Breadboard - Self-Adhesive \(White\)](#)

PRT-12002



SparkFun Beginner Tool Kit

TOL-14681



Break Away Headers - Straight

PRT-00116



**Jumper Wires Premium 6" M/M
Pack of 10**

PRT-08431



**Jumper Wires Premium 6" M/F
Pack of 10**

PRT-09140



**Three Phase Brushless Gimbal
Stabilizer Motor**

ROB-20441



**Power Supply - 80W DC Switching
Mode**

TOL-09291

Charge Pump Capacitor

The datasheet recommends a 0.1 μ F capacitor for the charge pump pin, to stabilize the input power supply when there are large swings in the voltage to the motor. Below are links to our 0.1 μ F capacitors:



Capacitor Ceramic 0.1uF

COM-08375



SparkFun Capacitor Kit

KIT-13698

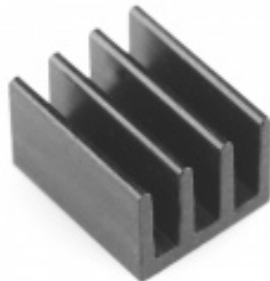
Heat Sink Accessories

To attach a heat sink to dissipate excess heat, users will want to check out the following components:



Small Heatsink

PRT-11510



Thermal Tape 4x4" Square

PRT-17054



Heatsink Compound

PRT-09599

Current Sensing

To perform current sensing from the low-side MOSFETs, users will need to amplify the output signal. Here are some products that users may be interested in:



SparkFun Configurable OpAmp Board - TSH82
BOB-14874

Break Away Headers - Straight
PRT-00116

SparkFun Beginner Tool Kit
TOL-14681

Hobby Knife
TOL-09200

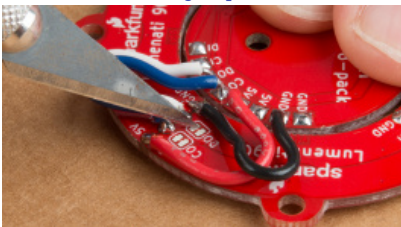
Jumper Modification

To modify the jumper, users will need [soldering equipment](#) and/or a [hobby knife](#).

New to jumper pads?

Check out our [Jumper Pads and PCB Traces Tutorial](#) for a quick introduction!

[How to Work with Jumper Pads and PCB Traces](#)



Solder Lead Free - 100-gram Spool
TOL-09325

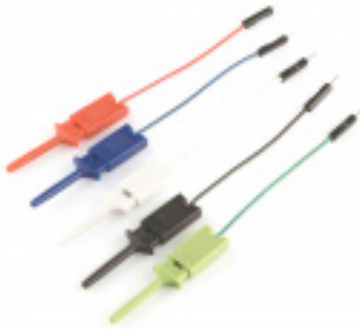
Weller WLC100 Soldering Station
TOL-14228

Chip Quik No-Clean Flux Pen - 10mL
TOL-14579

Hobby Knife
TOL-09200

Alternative Motor Connections

For users with less precise soldering skills, we recommend these wiring options as they are less dependent on how the leads to the motor are tinned. Additionally, users may find the alligator clips useful for motors with thicker wires.



IC Hook with Pigtail

CAB-09741



Alligator Clip with Pigtail (4 Pack)

CAB-13191

Alternative Power Source

Here are alternative battery options to power the TMC6300 motor driver. These batteries should be in the operation voltage range for the gimbal motor (just set the max charge to 8V). Additionally, they are capable of supplying more than the maximum current draw of the TMC6300.



SkyRC IMAX B6 V2 Professional Balance
Charger / Discharger

PRT-16793



Lithium Ion Battery - 2200mAh 7.4v

PRT-11856



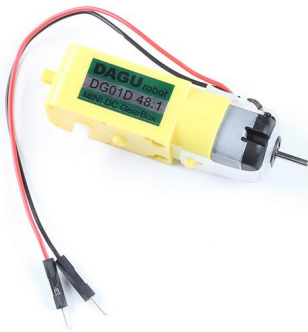
Lithium Ion Battery - 1000mAh 7.4v

PRT-11855

DC Motor Example

For users who wish to try out the H-bridge/DC motor example, they will also need a DC motor:

Hobby Gearmotor - 140 RPM
ROB-21245



1.1.2 Suggested Reading

As a more sophisticated product, we will skip over the more fundamental tutorials (i.e. **Ohm's Law** and **What is Electricity?**). However, below are a few tutorials that may help users familiarize themselves with various aspects of the board.

Electric Power



Connector Basics



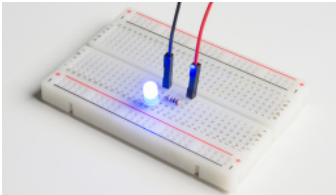
Working with Wire



How to Power a Project



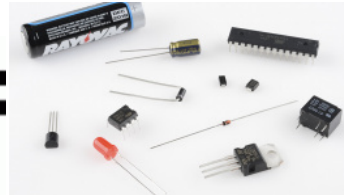
What is a Circuit?



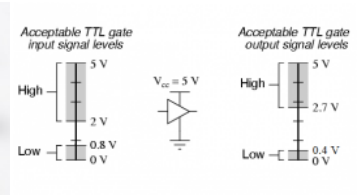
Alternating Current (AC) vs. Direct Current (DC)



Polarity



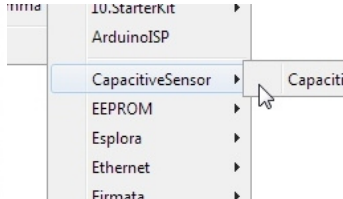
Logic Levels



Installing the Arduino IDE



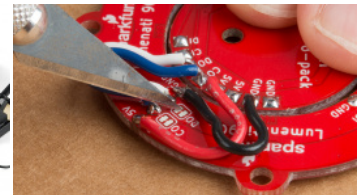
Installing an Arduino Library



How to Solder: Through-Hole Soldering



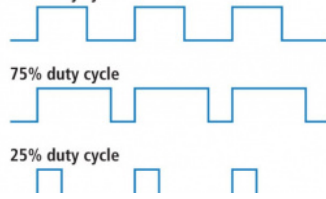
How to Work with Jumper Pads and PCB Traces



Motors and Selecting the Right One



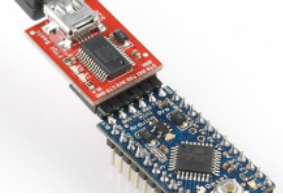
Pulse Width Modulation



How to Use an Oscilloscope



Serial Communication

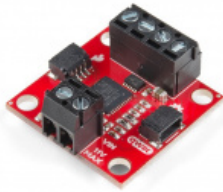


i Need to control a different type of motor?

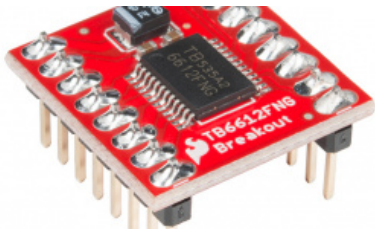
This tutorial is primarily focused on utilizing the TMC6300 motor driver to control a 3-phase brushless DC (BLDC) motor. While the versatility of this chipset allows for the control of other motor types, we would recommend less experienced users to explore products designed for those specific motors or actuators. Below, are additional product tutorials and resources for our other actuator and motor types:

- [Brushed DC Motors](#)
- [Stepper Motors](#)
- [Servos](#)

Hookup Guide for the Qwiic Motor Driver



TB6612FNG Hookup Guide



🕒 2023-08-21

🕒 2023-08-21

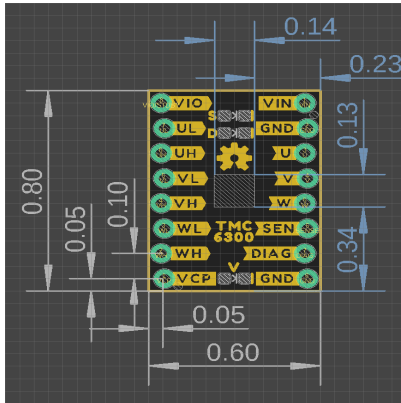
👤 [santaimpersonator](#)

🌐 [GitHub](#)

1.2 Hardware Overview

1.2.1 Board Dimensions

The board dimensions are illustrated in the drawing below; the listed measurements are in inches.



Board dimensions (PDF) for the TMC6300 motor driver breakout board, in inches.

Need more measurements?

For more information about the board's dimensions, users can download the [eagle files](#) for the board. These files can be opened in Eagle and additional measurements can be made with the dimensions tool.

Eagle - Free Download!

Eagle is a [CAD](#) program for electronics that is free to use for hobbyists and students. However, it does require an account registration to utilize the software.

Download from

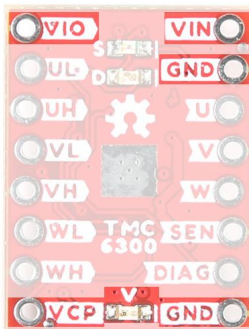
Dimensions Tool

This video from Autodesk demonstrates how to utilize the dimensions tool in Eagle, to include additional measurements:



1.2.2 Power

Users are provided with [PTH](#) to connect their external power supply, I/O logic-level voltage, and the regulated 1.8V output. The TMC6300 motor driver has an input voltage range of **2.0V** to **11.0V**.



The power connections on the TMC6300 motor driver.

Below, is a general summary of the circuitry on the board:

- **VIN** - Power supply input
 - Range: **2 to 11V**
- **VIO** - I/O supply voltage
 - Range: **1.8 to 5.25V**
 - **NSTDBY** - IC goes to standby mode and resets when this pin is pulled to **GND**
- **VCP** - Charge pump voltage
- **GND** - The common ground or the 0V reference for the board

Info

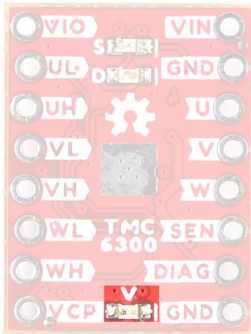
For more details, users can reference the [schematic](#) and the [TMC6300 datasheets](#).

Motor Voltage

Even though the input voltage range for the motor driver goes down to 2V, users will need to provide the minimum operating voltage for their motor. Our *Gimbal Stabilizer Motor* has an operating voltage range of 6 - 8V.

Power LED

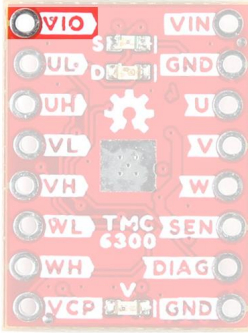
The red, power (**PWR**) LED will light up when a power supply is connected to the board. However, the LED can be disabled for low-power applications by cutting the [jumper](#).



The PWR status LED indicator for the TMC6300 motor driver.

VIO /Standby Pin

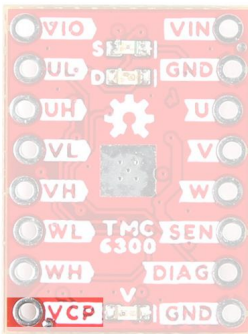
In it's default configuration, the **VIO** pin is used to enable the motor driver and set the logic level voltage (**1.8 to 5.25V**) for the I/O pins. However, the **VIO** pin also operates as a standby pin when it is pulled **LOW**. In standby, the TMC6300 resets and sits in standby mode.



vio pin on the TMC6300 motor driver.

VCP Pin

The `vcp` pin is broken out for users to include an external charge pump capacitor. Adding an external capacitor would help stabilize the supply voltage, from the large voltage swings (dV/dt) of the motor driver's operation. A 1nF to 100nF capacitor rated at 10V, is recommended by the [datasheet](#).



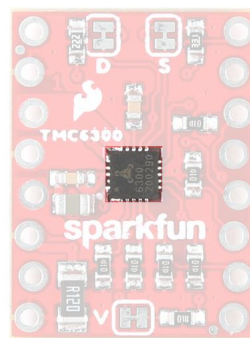
vcp pin on the TMC6300 motor driver.

1.2.3 TMC6300

The **TMC6300** from [Trinamic Motion Control](#), part of Analog Devices, is a low voltage, 3-Phase **BLDC/PMSM** motor driver utilizing separate high-side and low-side control signals for its three half-bridges.

Features:

- VIN: 2.0V to 11.0V
 - Operating current: 7mA
 - Standby current: 30nA
- VOUT: 1.8V
- 3 Half-Bridges
 - 3 High-side MOSFETs
 - 3 Low-side MOSFETs
- I/O Supply Voltage Input
- Diagnostic Output



TMC6300 chip on the TMC6300 motor driver.

- Overtemperature Protection
 - Shutdown Temperature: 150°C
 - Typical Power Dissipation: 1W
- Short Protection

Info

For more details, please refer to the [TMC6300 datasheet](#) and [application note](#).

Chip Protections

The TMC6300 features the overtemperature and short protections:

- The overtemperature protection feature implements a two temperature thresholds; however, the datasheet warns users that this should only be relied on as an emergency precaution and may not prevent the destruction of the IC. This is due to several factors including that excess heat can generate quickly before the overtemperature sensor can react. Therefore, users should prevent this situation from occurring by design with methods such as adequate heat dissipation.
- The short protection feature guards the motor commutation channels by monitoring the current flowing through each of the power stage MOSFETs. Once a short condition (short to `GND` or `VS`) is safely detected, all driver bridges become switched off, and the `DIAG` output becomes set. In order to restart the motor, the users must disable and restart the TMC6300. As with the overtemperature protection, the datasheet warns users that this feature should only be relied on as an emergency precaution and may not prevent the destruction of the IC or detect all possible short events.

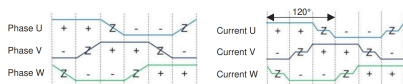
By monitoring the current draw through the `SEN` pin, users can also implement an over current protection scheme in their software. This can also aid in preventing a trigger in the overtemperature protection and validating a short detection.

Maximum Load Current

When pushing the maximum load current of 2A, users should monitor the current draw through the `SEN` pin and add a heat sink to provide additional heat dissipation. These precautions can help to avoid damaging the IC.

Motor Commutation

The TMC6300 relies on an electrical [commutation](#) sequence/signal to drive the motor phases to a [BLDC](#) or [PMSM](#) motor. The commutation signals for these motors are trapezoidal for BLDC motors and sinusoidal for PMSM motors.



Trapezoidal motor commutation.



Sinusoidal motor commutation.

(Source: *Brushless-DC Motor Driver Considerations and Selection Guide* application note)

For a trapezoidal signal, the high-side (HS) and low-side (LS) MOSFETs, can just be driven high or low. However, in order to approximate a sinusoidal signal, a progressively varying PWM signal must be provided from the microcontroller and all six signal should be in sync with each other.



Sinusoidal PWM signal.

(Source: *Demystifying BLDC motor commutation: Trap, Sine, & FOC*)

i PMSM vs BLDC Motor

A BLDC motor (sometimes referred to as a BLDM) and a PMSM (sometimes referred to as an AC synchronous motor) for the most part will appear the same in their internal construction. The difference between these types of motors is in their stator windings. This means that their commutation (the electrical signals used to drive the motor) is different.

- In a BLDC motor, the windings are concentrated on salient poles, requiring a voltage waveform that's more trapezoidal than sinusoidal.
- In a PMSM, the windings are distributed over several poles, requiring a voltage waveform that's more sinusoidal.

🔥 Gimbal Motor

Based on measuring the output from one of the coils, our gimbal motor is a PMSM and would require a sinusoidal waveform to drive the motor. It should be noted that a trapezoidal waveform can probably be used; however, users may notice effects such as cogging.

🔥 Alternative Use Cases

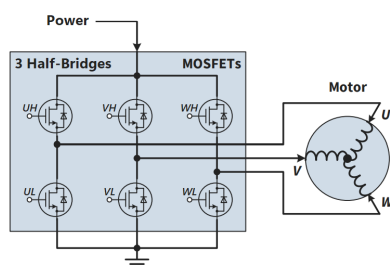
While this IC is intended to be used to drive 3-Phase BLDC/PMSM motors, users can easily adapt their hardware and software to work with other motors. For example, users could use two half-bridges to form an H-bridge and adapt their control software to drive a brushed, DC motor.

1.2.4 I/O Pins

There are several I/O pins for the TMC6300. Some of the pins are detailed in the [power section](#) above; the `vOUT` pin is omitted because it isn't broken out.

Half-Bridges

The TMC6300 features high-side and low-side MOSFET pairs of the three available half-bridges which control the commutation of the three motor phases.

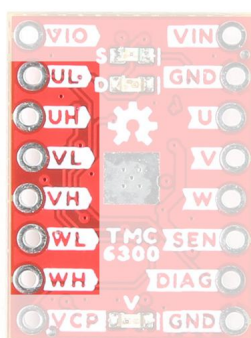


6 PWM control of a 3-phase motor commutation.

(Source: Modified from the [Block commutation vs. FOC in power tool motor control application note](#))

Input Output

The electronic commutation sequence of these pins will depend on the motor that is connected. For most cases, users will provide a PWM signal to each of the pins. *These are active-high pins with logic levels controlled by the VIO pin.*



The six control pins (*UL / UH*, *VL / VH*, and *WL / WH*) for the three half-bridges of the TMC6300 motor driver.

i Active High ▾

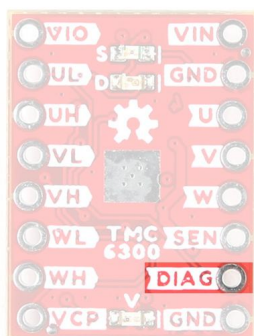
By pulling the pin high, the MOSFET will enable power to flow through that section of the half-bridge.

🔥 Microcontroller Limitations ▾

Users will need to use pins capable of providing a PWM signal. In addition, for the recommended [Simple FOC Arduino library](#), users will also need to consider the [supported microcontroller](#) as well as the configuration for the [6PWM mode](#).

Diagnostic Pin

The diagnostic pin is triggered based on different faults (*i.e. shorts and overtemperature*) detected by the IC. By default, the status will be indicated by the, green diagnostic, **D** LED and will remain **LOW** until triggered. Once triggered, users will need to disable and reset the TMC6300 or power cycle the board.

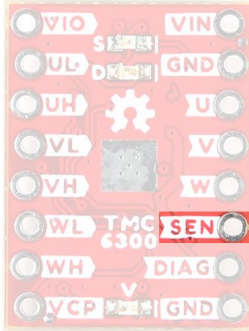


The *DIAG* pin on the TMC6300 motor driver.

Users can also monitor the *DIAG* pin, so their microcontroller knows when to reset the TMC6300 to clear the fault.

Current Sense Pin

The current sense pin is the foot point of the *u* and *v* half-bridges, with a 0.12Ω resistor attached. Users can measure the voltage across the *SENSE* and *GND* pins to determine the current flowing to the motor; however, it is recommended that an op-amp be attached to amplify the signal.

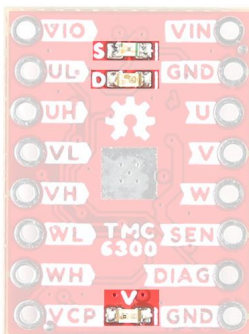


The *SEN* pin on the TMC6300 motor driver.

1.2.5 LEDs

There are three status indicator LEDs on the TMC6300 motor driver:

- *v* - Power (*Red*)
 - Turns on once power is supplies to the *VIN* pin
- *D* - Diagnostics (*Green*)
 - Turns on to indicate a fault (*see diagnostic pin section*)
- *s* - Standby (*Blue*)
 - Turns on when the motor driver is enabled
 - Turns off, when the IC has been reset and the motor driver is in standby mode



The status indicator LEDs on the TMC6300 motor driver.

1.2.6 Heat Sink Pad

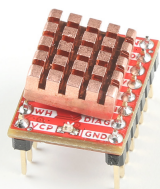
A 0.13" x 0.14" plated copper pad is provided on the top of the board, where users can add a heat sink to dissipate excess heat generated by the TMC6300. The pad can accommodate the [small heatsink](#) in our catalog.

i Thermal Shutdown Temperature

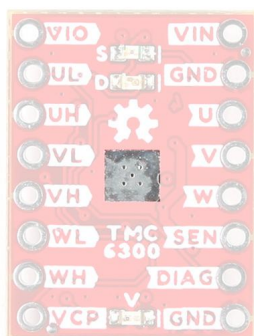
The AP329A has a 160°C (320°F) thermal shutdown temperature. The TMC6300 will restart automatically when the junction temperature decreases to +130°C.

⚠ Copper Heat Sink

Users may be tempted to use our [copper heatsink](#) on their board. However, we highly advise against using the copper heatsink because it barely fits and will likely cause a short across one of the pins.



Copper heatsink not fitting on the TMC6300 motor driver.



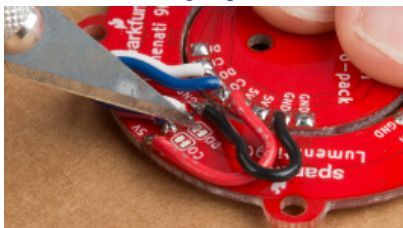
Heat sink pad on the top of the TMC6300 motor driver.

1.2.7 Jumpers

Never modified a jumper before?

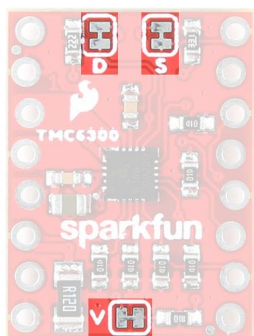
Check out our [Jumper Pads and PCB Traces tutorial](#) for a quick introduction!

[How to Work with Jumper Pads and PCB Traces](#)




There are three jumpers on the back of the board that can be used to easily modify a hardware connections on the board.

- **V** - This jumper can be used to remove power from the red, power LED.
- **S** - This jumper can be used to remove power from the blue, standby LED.
- **D** - This jumper can be used to remove power from the green, diagnostic LED.




The LED jumpers on the back of the TMC6300 motor driver.

 2023-08-21

 2023-08-21

 [santaimpersonator](#)

 [GitHub](#) 

1.3 Hardware Assembly

1.3.1 Hardware Components

TMC6300 Motor Driver

HEADERS

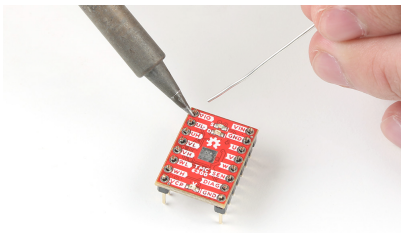
New to soldering?

If you have never soldered before or need a quick refresher, check out our [How to Solder: Through-Hole Soldering](#) guide.

[How to Solder: Through-Hole Soldering](#)

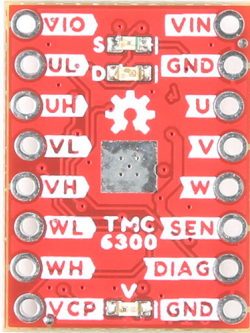


The pins on the SparkFun TMC6300 motor driver are broken out to 0.1"-spaced pins on the outer edges of the board. When selecting headers, be sure you are aware of the functionality and board orientation required.

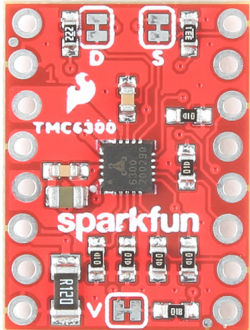


*Soldering headers to the TMC6300 motor driver.***Tip**

Please be aware that the side of the board with the silkscreen label for the pins and the heat sink pad is, technically the top side of the board when in use.



Top side of the TMC6300 motor driver.



Bottom side of the TMC6300 motor driver.

Staggered PTH Pins

The pins on the board may appear to be offset or crooked; this is by design, we stagger the holes along a specific center alignment. This reduces the geometric tolerance between the holes and header pins along a single axis, which helps to hold the header in place and keeps the pins more orthogonal to the board when soldering.

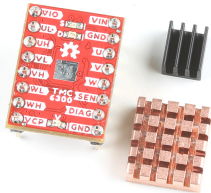
HEAT SINK

Tip

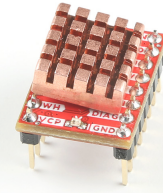
With larger heat sinks, we recommend a test fit and attaching it last to avoid conflicts with other parts of the board. For example, the heat sink could block the PTH pins/slots or access to the jumper pad.

⚠ Copper Heat Sink

Users may be tempted to use our [copper heatsink](#) on their board. However, we highly advise against using the copper heat sink because it barely fits and will likely cause a short across one of the pins.



Different heat sinks next to the TMC6300 motor driver.



Copper heat sink not fitting on the TMC6300 motor driver.

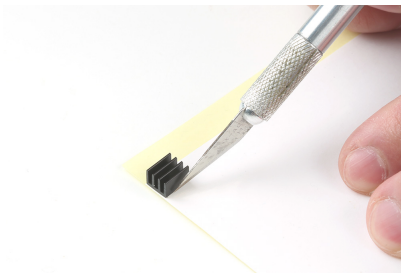
To attach a [heat sink](#) to the board, users will also need a piece of [thermal tape](#). We recommend the following procedure:

1. Cut out a piece of [thermal tape](#) to fit the bottom of the [heat sink](#).

Tip

Covering the entire bottom of the heat sink can insulate the electrical contacts on the board from shorting.

- For a perfect fit, users can place the heat sink over the tape and trace the outline to cut with scissors.
- For a perfect fit, users can also place the heat sink over the tape and cut the outline with a [hobby knife](#).



Cutting the thermal tape to fit the heat sink.

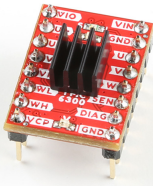
2. Place the piece of thermal tape on the bottom of the heat sink.

Tip

We recommend peeling off just one side of the backing sheet or [release liner](#) to place the thermal tape on the heat sink. Users can then peel the other side off when they are ready to place the heat sink on their board.

3. Attach the heat sink to the board.

- Make sure to make any jumper modifications and/or solder any connections before placing the heat sink on the board.
- Make sure to avoid any electrical contact with the sides of the heat sink.



Heat sink attached to the TMC6300 motor driver.

BLDC Gimbal Motor

Advanced Skills Required

To connect the gimbal motor to the TMC6300 motor driver board, some advanced soldering and wire stripping skills are necessary. The wire leads from the motor are only about 2" long, which is not a lot to work with. Users may only have two attempts at stripping the wires before they run out the available wire length.

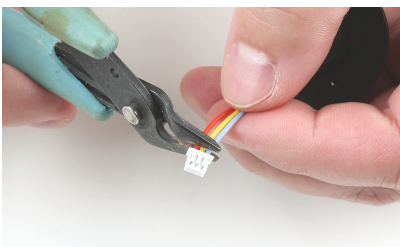
New to soldering?

If you have never soldered before or need a quick refresher, check out our [How to Solder: Through-Hole Soldering](#) guide.

[How to Solder: Through-Hole Soldering](#)



To connect the gimbal motor to the TMC6300 motor driver, users will need to expose the wiring. First, remove the JST connector and make sure to cut as close to the plastic housing as possible.

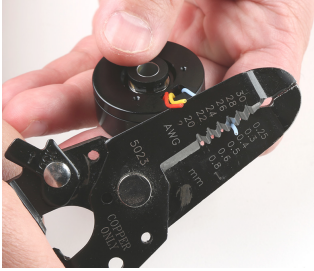


Cut off the plastic JST connector on the motor's wire leads.

i Alternative Connections

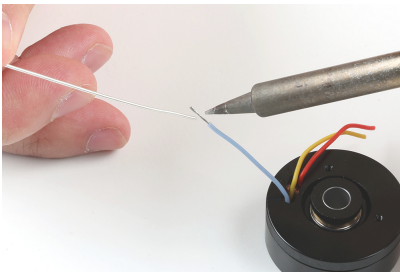
If users are utilize the alligator or IC hook pigtails, they may be able to crack the plastic housing of the connector to expose the crimped wire terminals.

Next, with as much care as possible, strip off some of the electrical insulation of each of the leads. On our [wire strippers](#), the 26AWG notch worked the best.

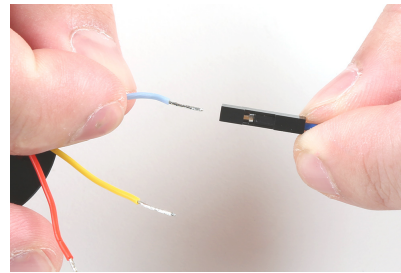


Strip off some of the insulation from the motor's wire leads.

The last step is to twist and tin the wires, so that they can be inserted into some jumper wires. Make sure to keep the ends straight and avoid adding to much solder, so that the wire ends can still fit into the female jumper wire terminals.



Twist and tin the exposed wire leads, so that they can be inserted into the female end of a jumper wire.



The leads should be straight and clear of bulges fit the into female terminals.

🕒 2023-08-21

🕒 2023-08-21

👤 [santaimpersonator](#)

🐙 [GitHub](#)

1.3.2 Example Setups

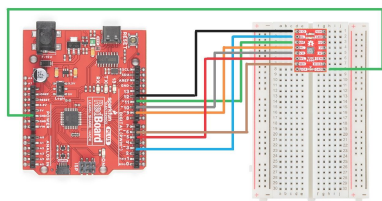
Assembly for Examples

The following instructions are for wiring up the RedBoard Plus, TMC6300 motor driver, and motor for the examples provided in this tutorial.

REDBOARD PLUS TO TMC6300

For ease of use, connect the TMC6300 motor driver with headers attached to the center of breadboard. The pin layout should perfectly align, so that users will have three breadboard pins available on each side of the TMC6300 breakout board. Then, following the table below, connect the pins from the RedBoard Plus to breadboard pins associated with the motor driver.

RedBoard	3	5	6	9	10	11	12	GND
Motor Driver	UL	WL	WH	VL	VH	UH	VIO	GND



A graphical representation of the connections between the RedBoard Plus and a breadboard with the TMC6300 motor driver attached.

i Timer Pins

As mentioned in the [Simple FOC library documentation](#),

ARDUINO UNO SUPPORT

Arduino UNO and all the atmega328 based boards have only 6 PWM pins and in order to use the `BLDCDriver6PWM` we need to use all of them. Those are 3, 5, 6, 9, 10 and 11. Furthermore in order for the algorithm to work well we need to use the PWM pins that belong to the same timer for each high/low side pair of each phase. So Atmega328 pins belonging to the timers are:

TIM0	TIM1	TIM2
5, 6	9, 10	3, 11

Therefore it is important that `phA_h` and `phA_l` belong to one timer, `phB_h` and `phB_l` to second timer and `phC_h` and `phC_l` to the last timer. If we decide that phase A belongs to the timer `TIM0` we can set `phA_h` either to pin 5 or pin 6.

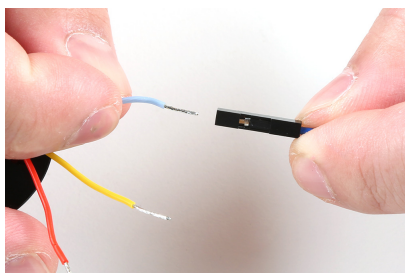
MOTOR TO TMC6300

Connecting a motor to the TMC6300 motor driver straight forward. Users, just need to connect the half-bridge drive channels with the ends of the motor's stator coils.

- Utilizing two half-bridges to drive a brushed DC motor with a full H-bridge
- Utilizing three half-bridges to drive a brushless DC motor

BLDC Gimbal Motor

Connecting a 3-phase, BLDC motor to the motor driver is relatively simple as the sequence of the wires connection doesn't matter. Using jumper wires, connect the prepared ends of the gimbal stabilizer motor to the `u`, `v`, and `w` pins of the TMC6300 breakout board.



A prepared end of the gimbal motor being inserted into the female terminals of a jumper wire.

Tip

Users may want to prop up the base of the gimbal motor, as the magnetic end of the motor shaft protrudes below its base plate. Users can see a demonstration in the video below:

 [Video with Gimbal Motor](#)

Reversing the Rotation of the Motor

For a 3-phase brushless motor, the connection sequence of the wires to a BLDC motor doesn't necessarily matter; the direction that the motor spins can be controlled through the software. However, for reference, switching two of the output channels to the motor will automatically reverse the direction that the motor was spinning.

- If users swap just the u and v connections, leaving the w connection alone, the motor will now spin in the opposite direction of the original configuration.

DC Hobby Motor

If this was a H-bridge motor driver, the connections to the motor wouldn't matter. However, as there are three half-bridges, users will need to note which half-bridges the DC motor is connected to. These connections will dictate how the motor is driven by the software. For the example, connect the motor to the v and w output channels of the TMC6300 motor driver.

Reversing the Rotation of the Motor

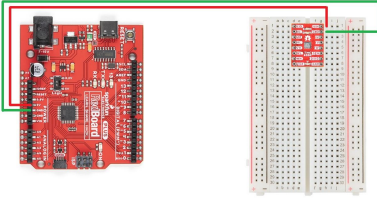
For a single phase DC motor, the direction that the motor spins can be controlled through the software. However, for reference, switching the polarity of the motor's wires will also reverse the direction that the motor was spinning.

POWERING THE TMC6300

Enough power should be provided to the TMC6300 to drive the motor connected to it. Therefore, the drive current and voltage range of the motor should be taken into consideration. Additionally, users should monitor the drive current to prevent overheating of the TMC6300 motor driver.

Ideally, if users have access to a variable power supply, it would be the most convenient solution for adjusting the voltage and source current parameters. Other power supply alternatives include a 6V (4xAA) battery pack or dual-cell LiPo battery. However, if users are unable to find a suitable power source, we have found that the 5V power output from the RedBoard Plus is sufficient to drive the gimbal motor, under a no load condition at low speeds.

RedBoard	5V	GND
Motor Driver	VIN	GND



A graphical representation of the connections between the *RedBoard Plus* and a *breadboard* with the *TMC6300 motor driver* attached.


Current Monitoring

For the examples in this tutorial, the motor will be driven with a no-load condition and the motor drive current shouldn't need to be monitored. (An exception would be when the motor is hindered from spinning, in which case the torque and drive current will spike.)

 2023-08-21

 2023-08-21

 [santaimpersonator](#)

 [GitHub](#) 

1.4 Software - Arduino IDE

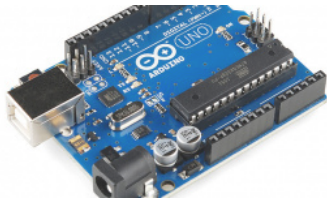
1.4.1 Installation & Setup

Arduino IDE

Tip

For first-time users, who have never programmed before and are looking to use the Arduino IDE, we recommend beginning with the [SparkFun Inventor's Kit \(SIK\)](#), which is designed to help users get started programming with the Arduino IDE.

Most users may already be familiar with the Arduino IDE and its use. However, for those of you who have never heard the name *Arduino* before, feel free to check out the [Arduino website](#). To get started with using the Arduino IDE, check out our tutorials below:



WHAT IS AN ARDUINO?



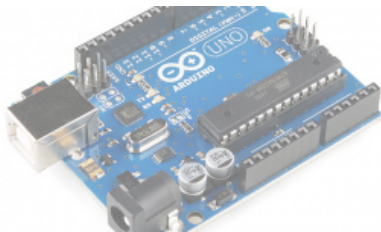
INSTALLING ARDUINO IDE



INSTALLING AN ARDUINO LIBRARY



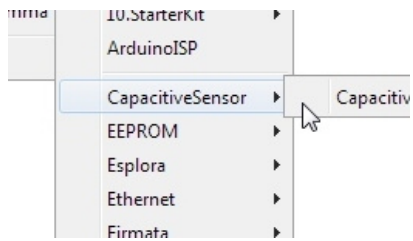
INSTALLING BOARD DEFINITIONS IN THE ARDUINO IDE



What is an Arduino?



Installing the Arduino IDE



Installing an Arduino Library



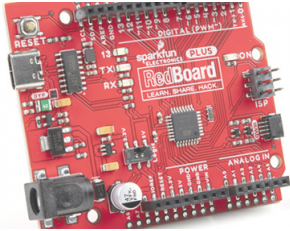
Installing Board Definitions in the Arduino IDE



Need help setting up the RedBoard Plus?

REDBOARD PLUS

The following instructions should help users get started with the RedBoard Plus. For more information about the board, please check out our hookup guide below:

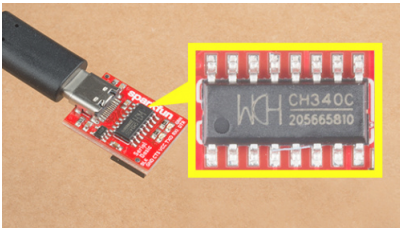


[RedBoard Plus Hookup Guide](#)

CH340 Driver

Users will need to install the appropriate driver for their computer to recognize the serial-to-UART chip on their board/adapter. Most of the latest operating systems will recognize the CH340C chip on the board and automatically install the required driver.

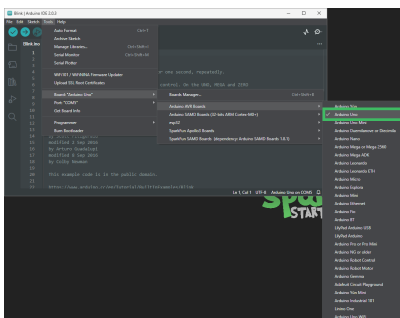
To manually install the CH340 driver on their computer, users can download it from the [WCH website](#). For more information, check out our [How to Install CH340 Drivers Tutorial](#).



[How to Install CH340 Drivers](#)

Arduino IDE

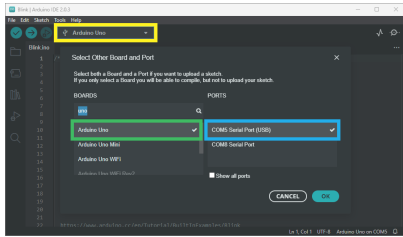
When selecting a board to program in the Arduino IDE, users should select the **Arduino Uno** from the **Tools** drop-down menu (*i.e.* **Tools > Board > Arduino AVR Boards > Arduino Uno**).



Select the **Arduino Uno** from the **Tools** drop-down menu in the **Arduino IDE**.

Arduino IDE 2.x.x - Alternative Method

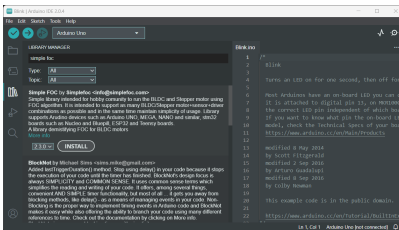
In the newest version of the Arduino IDE 2.x.x, users can also select their board (green) and port (blue) from the Select Board & Port dropdown menu (yellow).



Selecting the **Arduino Uno** and **COM5** port from the **Select Board & Port** drop-down menu in the Arduino IDE (v2.0.3).

SimpleFOCLibrary

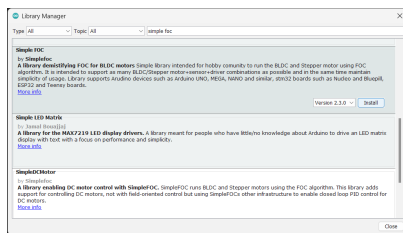
The [Simple Field Oriented Control Library](#) can be installed from the library manager in the Arduino IDE.



SimpleFOCLibrary in the library manager of the Arduino IDE.

Arduino IDE (v1.x.x)

In the Arduino IDE v1.x.x, the library manager will have the following appearance for the SimpleFOC library:



SimpleFOCLibrary in the library manager of the Arduino IDE (v1.x.x).

This library utilizes a motor control scheme called field oriented control (FOC), which can utilize a feedback control loop to drive a motor with a higher power efficiency and precision characteristics, like evenly distributed torque control.

Info

For more details about the library, check out the [online documentation](#).

Supported Hardware

For a detailed and up-to-date list of the hardware supported by this library, check out the [library's documentation](#). The following are excerpts taken from the library's documentation page:

Microcontrollers Motor Drivers Motors

Arduino SimpleFOCLibrary supports:

MCU	2 PWM mode	4 PWM mode	3 PWM mode	6 PWM mode	pwm frequency config
Arduino (8-bit)	✓	✓	✓	✓	✓ (either 4kHz or 32kHz)
Arduino DUE	✓	✓	✓	✗	✓
stm32	✓	✓	✓	✓	✓
esp32 MCPWM	✓	✓	✓	✓	✓
esp32 LEDC	✓	✓	✓	✗	✓
esp8266	✓	✓	✓	✗	✓
samd21/51	✓	✓	✓	✓	✓
teensy	✓	✓	✓	✓	✓
Raspberry Pi Pico	✓	✓	✓	✓	✓
Portenta H7	✓	✓	✓	✗	✓
nRF52	✓	✓	✓	✓	✓

From this table you can see that if you need the 6 PWM mode for your application you should avoid using Teensy and Arduino DUE boards for now.

Info

For more details, please refer to the [SimpleFOC Arduino library documentation](#).

6PWM MOTOR DRIVER

Users will need to utilize the `BLCDriver6PWM` class to provide the six PWM signals required for the TMC6300 motor driver.

BLDCDriver6PWM

This class provides an abstraction layer for most of the common BLDC drivers, which require six PWM signals. This method offers more control than a three PWM motor drivers, since each of the 6 half-bridges MOSFETs can be controlled independently.

To create the interface to the BLDC driver you need to specify the 6 PWM pin numbers for each motor phase and optionally enable pin.

```
// BLDCDriver6PWM( int phA_h, int phA_l, int phB_h, int phB_l, int phC_h, int phC_l, int en)
// - phA_h, phA_l - A phase pwm pin high/low pair
// - phB_h, phB_l - B phase pwm pin high/low pair
// - phB_h, phC_l - C phase pwm pin high/low pair
// - enable pin - (optional input)
BLDCDriver6PWM motor = BLDCDriver6PWM(5,6, 9,10, 3,11, 8);
```

Microcontroller Considerations

Arduino Uno STM32 ESP32

Arduino UNO and all the atmega328 based boards have only 6 PWM pins and in order to use the `BLDCDriver6PWM` we need to use all of them. Those are `3, 5, 6, 9, 10` and `11`. Furthermore in order for the algorithm to work well we need to use the PWM pins that belong to the same timer for each high/low side pair of each phase. So Atmega328 pins belonging to the timers are:

TIM0	TIM1	TIM2
5, 6	9, 10	3, 11

Therefore it is important that `phA_h` and `phA_l` belong to one timer, `phB_h` and `phB_l` to second timer and `phC_h` and `phC_l` to the last timer. If we decide that phase A belongs to the timer `TIM0` we can set `phA_h` either to pin 5 or pin 6.

Info

For more details about the `BLDCDriver6PWM` class, check out the [online documentation](#).

BLDC MOTOR

All BLDC motors are handled with the `BLDCMotor` class.

BLDCMotor

This class implements the BLDC FOC algorithm, motion control loops, and monitoring.


To instantiate the BLDC motor we need to create an instance of the `BLDCMotor` class and provide it the number of pole pairs of the motor.

```
// BLDCMotor(int pp, (optional R, KV))
// - pp - pole pair number
// - R - phase resistance value - optional
// - KV - motor KV rating [rpm/V] - optional
BLDCMotor motor = BLDCMotor(11, 10.5, 120);
```

 **Motor Considerations**

While, the datasheet for our [gimbal motor](#), indicates that there are **8 pole pairs**, we have found that the motor operates more smoothly if the `BLDCMotor` class is instantiated with **7 pole pairs** instead.

```
BLDCMotor motor = BLDCMotor(7);
```

 **Info**

For more details about the `BLDCMotor` class, check out the [online documentation](#).

MOTION CONTROL

Unless a feedback loop is incorporated when driving the motor (*i.e. with [position sensors](#) or [current sensing](#)*), users should implement the SimpleFOC library using the [open-loop control](#).

Open-Loop Motion Control Types:

- [Velocity](#)
- [Position](#)

 2023-08-21

 2023-08-21

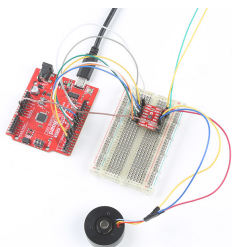
 [santaimpersonator](#)

 [GitHub](#) 

1.4.2 Example - Basic

Hardware Assembly

Users should already have followed the instructions from the [component assembly](#) and [example setups](#) sections to setup their hardware for this example.

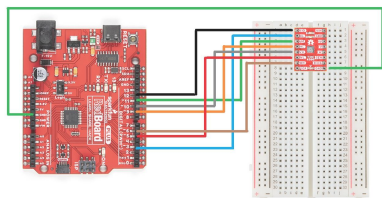


A graphical representation of the connections between the RedBoard Plus and a breadboard with the TMC6300 motor driver attached.

REDBOARD PLUS TO TMC6300

Connect the following pins from the RedBoard Plus to the TMC6300.

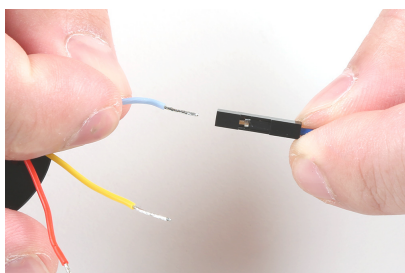
RedBoard	3	5	6	9	10	11	12	GND
Motor Driver	UL	WL	WH	VL	VH	UH	VIO	GND



A graphical representation of the connections between the RedBoard Plus and a breadboard with the TMC6300 motor driver attached.

CONNECTING THE GIMBAL MOTOR

Connecting a 3-phase, BLDC motor to the motor driver is relatively simple as the sequence of the wires connection doesn't matter. Using jumper wires, connect the prepared ends of the gimbal stabilizer motor to the **u**, **v**, and **w** pins of the TMC6300 breakout board.

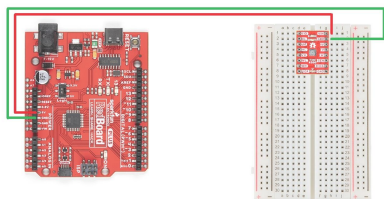


A prepared end of the gimbal motor being inserted into the female terminals of a jumper wire.

POWERING THE TMC6300

If users are unable to find a suitable power source, we have found that the 5V power output from the RedBoard Plus is sufficient to drive the gimbal motor, under a no load condition at low speeds.

RedBoard	5V	GND
Motor Driver	VIN	GND



A graphical representation of the connections between the [RedBoard Plus](#) and a [breadboard](#) with the [TMC6300 motor driver](#) attached.

Example Code

After installing and setting up the Arduino IDE and the Simple FOC Arduino library, users will need to upload the following example code to the RedBoard Plus. This code can be copied or downloaded below:

[Download `BLDC.ino` Example Code](#)



Example Code

BLDC.ino

```

// Open loop motor control example
#include <SimpleFOC.h>

// BLDC motor & driver instance
// BLDCMotor motor = BLDCMotor(pole pair number);
BLDCMotor motor = BLDCMotor(7);
// BLDCDriver3PWM driver = BLDCDriver3PWM(pwmA, pwmB, pwmC, Enable(optional));
BLDCDriver6PWM driver = BLDCDriver6PWM(5, 6, 9, 10, 3, 11);

// Stepper motor & driver instance
//StepperMotor motor = StepperMotor(50);
//StepperDriver4PWM driver = StepperDriver4PWM(9, 5, 10, 6, 8);

//target variable
float target_velocity = 6;

// // instantiate the commander
Commander command = Commander(Serial);
// void doTarget(char* cmd) { command.scalar(&target_velocity, cmd); }
// void doLimit(char* cmd) { command.scalar(&motor.voltage_limit, cmd); }

void setup() {
  // driver config
  // power supply voltage [V]
  driver.voltage_power_supply = 5;
  // limit the maximal dc voltage the driver can set
  // as a protection measure for the low-resistance motors
  // this value is fixed on startup
  driver.voltage_limit = 5;
  // pwm frequency to be used [Hz]
  // for atmega328 fixed to 32kHz
  // esp32/stm32/teensy configurable
  driver.pwm_frequency = 32000;

  driver.init();
  // link the motor and the driver
  motor.linkDriver(&driver);

  // limiting motor movements
  // limit the voltage to be set to the motor
  // start very low for high resistance motors
  // current = voltage / resistance, so try to be well under 1Amp
  motor.voltage_limit = 3; // [V]

  // open loop control config

```

RUNNING THE MOTOR

By default, the motor should spin automatically. However, if users wish to control the speed of the motor, they can uncomment lines **21-22** and **56-57** of code and reprogram the RedBoard Plus.

Changes Highlighted

Uncomment the following lines of code (**21-22** and **56-57**):

BLDC.ino

```
// Open loop motor control example
#include <SimpleFOC.h>

// BLDC motor & driver instance
// BLDCMotor motor = BLDCMotor(pole pair number);
BLDCMotor motor = BLDCMotor(7);
// BLDCDriver3PWM driver = BLDCDriver3PWM(pwmA, pwmB, pwmC, Enable(optional));
BLDCDriver6PWM driver = BLDCDriver6PWM(5, 6, 9, 10, 3, 11);

// Stepper motor & driver instance
//StepperMotor motor = StepperMotor(50);
//StepperDriver4PWM driver = StepperDriver4PWM(9, 5, 10, 6, 8);

//target variable
float target_velocity = 6;

// // instantiate the commander
Commander command = Commander(Serial);
// void doTarget(char* cmd) { command.scalar(&target_velocity, cmd); }
// void doLimit(char* cmd) { command.scalar(&motor.voltage_limit, cmd); }

void setup() {

  // driver config
  // power supply voltage [V]
  driver.voltage_power_supply = 5;
  // limit the maximal dc voltage the driver can set
  // as a protection measure for the low-resistance motors
  // this value is fixed on startup
  driver.voltage_limit = 5;
  // pwm frequency to be used [Hz]
  // for atmega328 fixed to 32kHz
  // esp32/stm32/teensy configurable
  driver.pwm_frequency = 32000;

  driver.init();
  // link the motor and the driver
  motor.linkDriver(&driver);

  // limiting motor movements
  // limit the voltage to be set to the motor
  // start very low for high resistance motors
  // current = voltage / resistance, so try to be well under 1Amp
  motor.voltage_limit = 3; // [V]

  // open loop control coeff
```

In order to drive the motor, users will need to access the [serial monitor](#) and provide the [commands](#) necessary to drive the motor. A full list of the available commands can be found in the [Simple FOC Arduino library documentation](#). However, only the `T` and `L` commands are enabled in the example code.

- Sending a `T` command will set the target motor velocity in rads/s
 - Example - Entering `T6` into the serial monitor, will set the target motor velocity to 6 radians/s.
- Sending a `L` command will set the voltage limit of the motor driver and in turn, the current limit (*voltage_limit / motor_resistance*)
 - Example - Entering `L5` into the serial monitor, will set the voltage limit to 5V and the current limit to .5A ($5V/10\Omega$).

Baud Rate

The serial monitor baud rate should be configured to 115200bps.

 2023-08-21

 2023-08-21

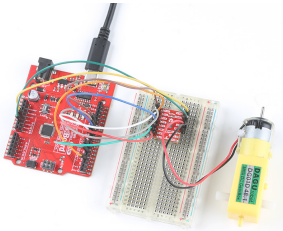
 [santaimpersonator](#)

 [GitHub](#) 

1.4.3 Example - H-Bridge

Hardware Assembly

Users should already have followed the instructions from the [component assembly](#) and [example setups](#) sections to setup their hardware for this example.

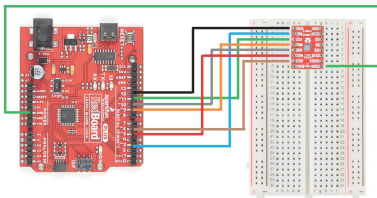


A graphical representation of the connections between the [RedBoard Plus](#) and a [breadboard](#) with the [TMC6300 motor driver](#) attached.

REDBOARD PLUS TO TMC6300

Connect the following pins from the RedBoard Plus to the TMC6300.

RedBoard	3	5	6	9	10	11	12	GND
Motor Driver	UL	WL	WH	VL	VH	UH	VIO	GND



A graphical representation of the connections between the [RedBoard Plus](#) and a [breadboard](#) with the [TMC6300 motor driver](#) attached.

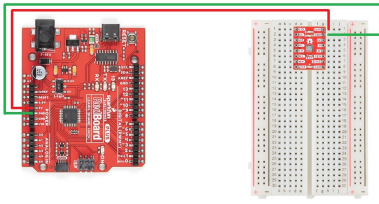
CONNECTING THE DC MOTOR

If this was a H-bridge motor driver, the connections to the motor wouldn't matter. However, as there are three half-bridges, users will need to note which half-bridges the DC motor is connected to. These connections will dictate how the motor is driven by the software. For the example, connect the motor to the `v` and `w` output channels of the TMC6300 motor driver.

POWERING THE TMC6300

If users are unable to find a suitable power source, we have found that the 5V power output from the RedBoard Plus is sufficient to drive the gimbal motor, under a no load condition at low speeds.

RedBoard	5V	GND
Motor Driver	VIN	GND



A graphical representation of the connections between the *RedBoard Plus* and a *breadboard* with the *TMC6300 motor driver* attached.

Example Code

After installing and setting up the Arduino IDE and the Simple FOC Arduino library, users will need to upload the following example code to the RedBoard Plus. This code can be copied or downloaded below:

[Download `DC.ino` Example Code](#)

Example Code

```

DC.ino

// BLDC driver standalone example
#include <SimpleFOC.h>

// BLDC driver instance
BLDCDriver6PWM driver = BLDCDriver6PWM(5, 6, 9, 10, 3, 11);

void setup() {

  // pwm frequency to be used [Hz]
  // for atmega328 fixed to 32kHz
  // esp32/stm32/teensy configurable
  driver.pwm_frequency = 32000;
  // power supply voltage [V]
  driver.voltage_power_supply = 5;
  // Max DC voltage allowed - default voltage_power_supply
  driver.voltage_limit = 5;
  // daad_zone [0,1] - default 0.02f - 2%
  driver.dead_zone = 0.05f;

  // driver init
  driver.init();

  // enable driver
  driver.enable();

  _delay(1000);
}

void loop() {
  driver.setPwm(5, 0, 0);
  _delay(1000);

  driver.setPwm(0, 5, 0);
  _delay(1000);
}

```

RUNNING THE MOTOR

By default, the motor should spin automatically. However, if users wish to control the motor, they can modify lines **30** and **33** of code and reprogram the RedBoard Plus. These lines control the high and low-side MOSFETs of the H-bridge directly through pins 5, 6, 9, and 10.

Changes Highlighted

Modify the following lines of code (**30** and **33**):

DC.ino

```
// BLDC driver standalone example
#include <SimpleFOC.h>

// BLDC driver instance
BLDCDriver6PWM driver = BLDCDriver6PWM(5, 6, 9,10, 3, 11);

void setup() {

  // pwm frequency to be used [Hz]
  // for atmega328 fixed to 32kHz
  // esp32/stm32/teensy configurable
  driver.pwm_frequency = 32000;
  // power supply voltage [V]
  driver.voltage_power_supply = 5;
  // Max DC voltage allowed - default voltage_power_supply
  driver.voltage_limit = 5;
  // daad_zone [0,1] - default 0.02f - 2%
  driver.dead_zone = 0.05f;

  // driver init
  driver.init();

  // enable driver
  driver.enable();

  _delay(1000);
}

void loop() {
  driver.setPwm(5, 0, 0);
  _delay(1000);

  driver.setPwm(0, 5, 0);
  _delay(1000);
}
```

- The PWM voltage value `driver.setPwm(voltage_value, 0, 0)` affects the speed of the motor.
- The position of this value in relation to the PWM drive channel `driver.setPwm(channel_v, channel_w, 0)` affects the direction.

Tip

Users may also be interested in the [SimpleDC Motor library](#) to drive DC motors with the Simple FOC Arduino library.

 2023-08-21

 2023-08-21










 [santaimpersonator](#)

 [GitHub](#) 

2. Resources

2.1 Hardware Resources

2.1.1 Product Resources





-  [Product Page](#)
- Component Documentation
 -  [TMC6300 Datasheet](#)
-  Design Files:
 -  [Board Dimensions](#)
 -  [Schematic](#)
 -  [Eagle Files](#)
- Arduino Library:
 -  [Simple FOC](#)
 -  [Documentation](#)
 - [Installation](#)
 - [Supported Hardware](#)
 - [Code Overview](#)
-  [SFE Product Showcase](#)
-  [Hardware Repo](#)

Additional Resources

- [Motors and Motor Driver Product Category](#)
-  [SparkFun Technical Assistance](#)

2.1.2 Manufacturer's Resources

Analog Devices + Trinamic also provides great resources for the TMC6300 motor driver:

- [TMC6300 Product Page](#)
 - [Block Diagram/Pinout](#)
 -  [Datasheet](#)
 -  [Application Notes](#)
-  [Trinamic: TMC6300 Product Training Module](#)
-  [Technical Support Page](#)



 2023-08-21




 2023-08-21

 [santaimpersonator](#)

 [GitHub](#) 

2.2 Background Information

Below, are several articles, application notes, and other technical resources on 3-phase motors and utilizing a field oriented control (FOC) scheme:

- Microchip Technology
 - [AN885: Brushless DC \(BLDC\) Motor Fundamentals](#)
 - [AN2757: Sensored \(Encoder-Based\) Field Oriented Control of Three-Phase Permanent Magnet Synchronous Motor \(PMSM\)](#)
 - [AN1078: Sensorless Field Oriented Control of a PMSM](#)
 - [AVR32723: Sensor Field Oriented Control for Brushless DC motors with AT32UC3B0256](#)
- Diodes Incorporated
 - [AN1164: Introduction to Brushless DC Motors](#)
- Monolithic Power Systems
 - [AN047: Brushless DC Motor Fundamentals](#)
- Texas Instruments
 - [Demystifying BLDC motor commutation: Trap, Sine, & FOC](#)
 - [Sensored Field Oriented Control of 3-Phase Permanent Magnet Synchronous Motors](#)
 - [Sensorless Field Oriented Control of 3-Phase Permanent Magnet Synchronous Motors](#)
 - [Brushless-DC Motor Driver Considerations and Selection Guide](#)
 - [High Performance Brushless DC Motor Control](#)
 -  [Field oriented control of permanent magnet synchronous motors](#)
 -  [Field Oriented Control of Permanent Magnet Motors](#)
 -  [Field Oriented Control of Permanent Magnet Motors](#)
- MATLAB
 -  [Motor Control, Part 4: Understanding Field-Oriented Control](#)
- Analog Devices + Trinamic
 - [FOC As Hardware Building Block](#)
- Infineon Technologies
 - [Motor Handbook](#)
 - [AN204469 - FM3 Family 3-Phase PMSM FOC Control](#)
 - [Block commutation vs. FOC in power tool motor control](#)
 -  [Motor control for BLDC: block commutation vs. field-oriented control](#)
- ST Microelectronics
 - [AN5397: Current Sensing in Motion Control Applications](#)
 - [AN4220: Sensorless Six-Step BLDC Commutation](#)

 2023-08-21

 2023-08-21

 [santaimpersonator](#)

 [GitHub](#) 

3. Support

3.1 Troubleshooting Tips

Need Help?

If you need technical assistance or more information on a product that is not working as you expected, we recommend heading on over to the [SparkFun Technical Assistance](#) page for some initial troubleshooting.

[SparkFun Technical Assistance Page](#)

If you can't find what you need there, the [SparkFun Forums](#) is a great place to search product forums and ask questions.

Account Registration Required

If this is your first visit to our forum, you'll need to create a [Forum Account](#) to post questions.

3.1.1 ESP32 Compile Error

If users run into an error similar to, `fatal error: soc/soc_caps.h: No such file or directory`, it may be due to an issue with the version of the ESP32 Arduino core that is installed in the **Boards Manager**. Users should make sure they have the latest version of the ESP32 Arduino core installed; or at least a version later than `v2.0.1`.

Info

For more information, please reference this [forum post](#) for the Simple FOC Arduino library.

 2023-08-21

 2023-08-21

 [santaimpersonator](#)

 [GitHub](#) 