

# 2A DC Motor Phidget

## Contents

### **1 WELCOME**

### **2 USING THE DCC1000**

#### 2.1 Phidget Control Panel

##### 2.3.1 Windows

##### 2.3.2 macOS

#### 2.2 First Look

#### 2.3 DC Motor

#### 2.4 Encoder

#### 2.5 Position Controller

### **3 TECHNICAL DETAILS**

#### 3.1 Control Loop Parameters

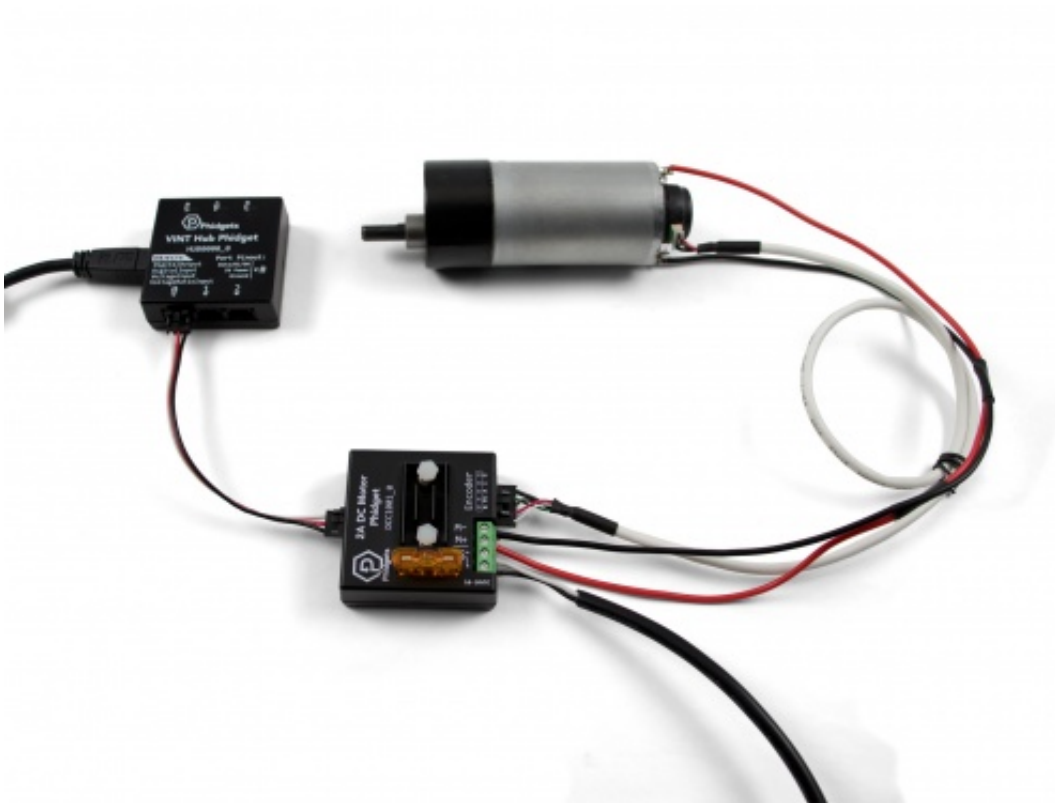
## 4 WHAT TO DO NEXT

# Getting Started

Welcome to the DCC1001 user guide! In order to get started, make sure you have the following hardware on hand:

- DCC1001 - 2A DC Motor Phidget
- VINT Hub
- Phidget cable
- USB cable and computer
- Power supply (8-30V DC)
- DC motor

Next, you will need to connect the pieces:



TO  
TOP

1. Connect the DCC1001 to the VINT Hub using the Phidget cable.
2. Connect the motor to the Phidget's output terminals.
3. Connect the VINT Hub to your computer with a USB cable.
4. (Optional) If your motor has an encoder, connect it to the encoder port on the DCC1001.
5. Connect the power supply to the DCC1001's power terminals.

Now that you have everything together, let's start using the DCC1001!

## Using the DCC1001

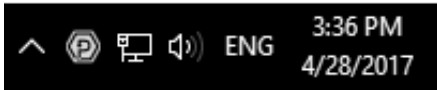
### Phidget Control Panel

In order to demonstrate the functionality of the DCC1001, the Phidget Control Panel running on a Windows machine will be used.


The Phidget Control Panel is available for use on both macOS and Windows machines.

# Windows

To open the Phidget Control Panel on Windows, find the  icon in the taskbar. If it is not there, open up the start menu and search for Phidget Control Panel



# macOS

To open the Phidget Control Panel on macOS, open Finder and navigate to the Phidget Control Panel in the Applications list. Double click on the  icon to bring up the Phidget Control Panel.

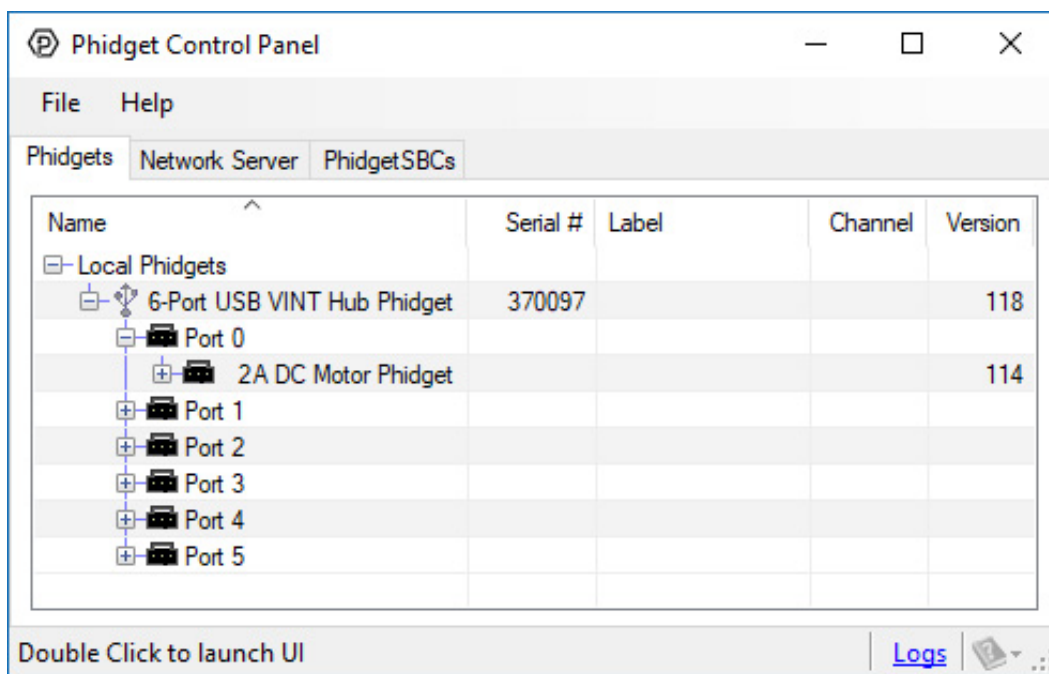
For more information, take a look at the getting started guide for your operating system:

- Getting started with Windows
- Getting started with macOS

Linux users can follow the getting started with Linux guide and continue reading here for more information about the DCC1001.

# First Look

After plugging the DCC1001 into your computer and opening the Phidget Control Panel, you will see something like this:



The Phidget Control Panel will list all connected Phidgets and associated objects, as well as the following information:

- **Serial number:** allows you to differentiate between similar Phidgets.
- **Channel:** allows you to differentiate between similar objects on a Phidget.
- **Version number:** corresponds to the firmware version your Phidget is running. If your Phidget is listed in red, your firmware is out of date. Update the firmware by double-clicking the entry.

The Phidget Control Panel can also be used to test your device. Double-clicking on an object will open an example.

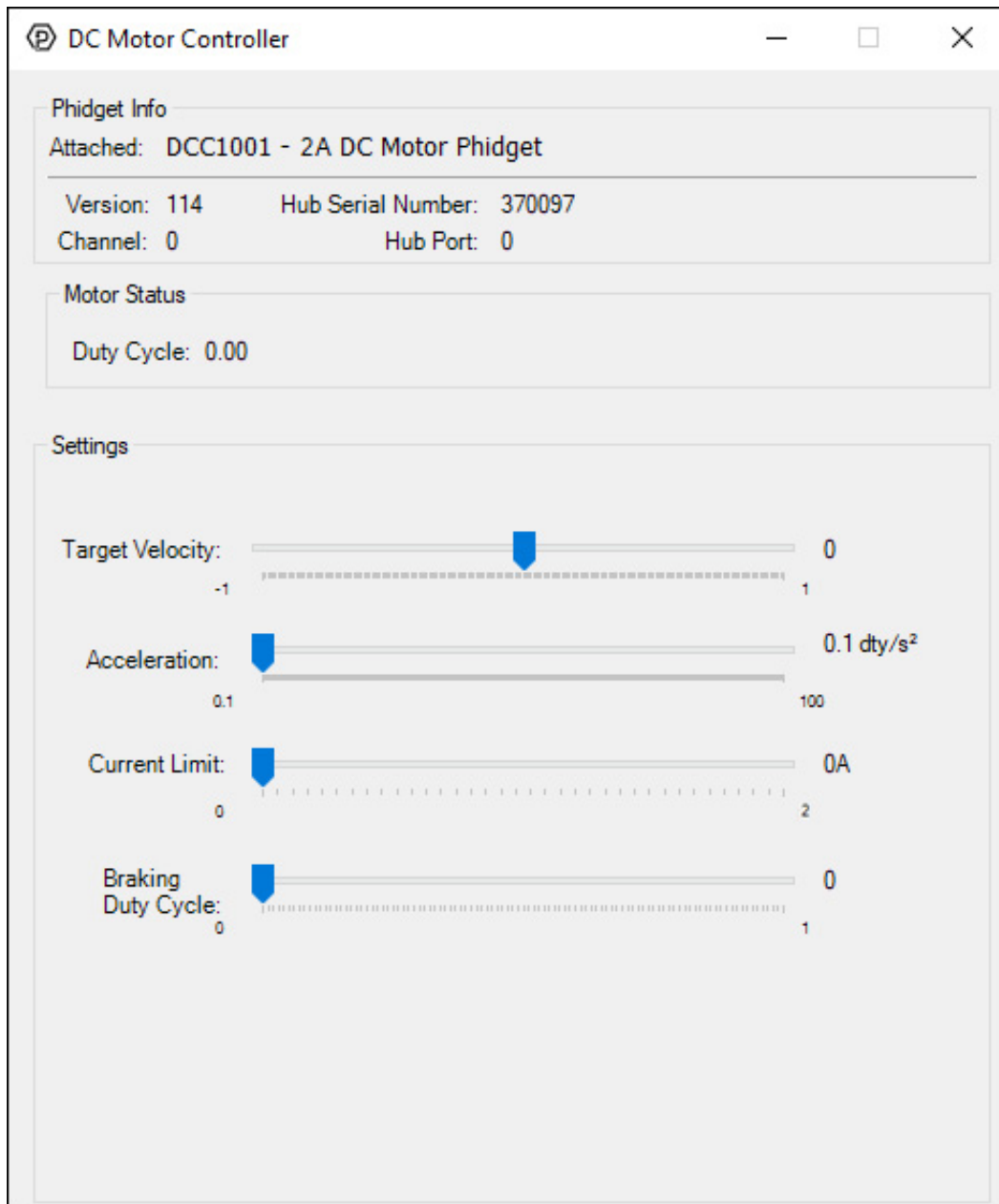
The objects associated with the DCC1001 are as follows:

- **DC Motor Controller:** Controls the velocity and current of the motor.
- **Encoder Input:** Reads encoder input so you can implement closed-loop control of the motor.

- **Position Controller:** A built-in position controller.

## DC Motor

Double-click on the DC Motor object, labelled DC Motor Phidget, in order to run the example:

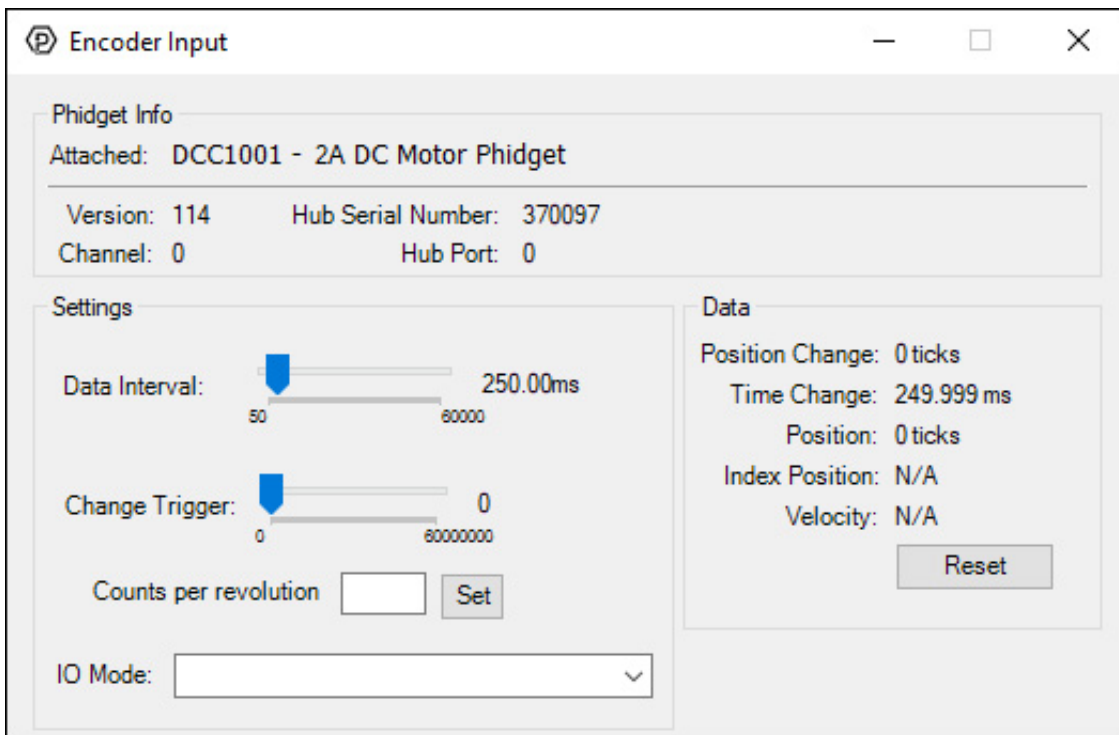


General information about the selected object will be displayed at the top of the window. You can also experiment with the following functionality:

- Drag the Target Velocity slider from -1 (full reverse) to 1 (full forward) to make the motor move.
- Manipulate the Acceleration slider to increase/decrease the amount of time it takes the DC Motor to reach a target velocity.
- Manipulate the Current Limit slider to limit the amount of current provided to the motor. Higher current means more torque, but more power consumption.
- Manipulate the Braking Duty Cycle slider to change how hard the motor brakes.
- Manipulate the Current Regulator Gain: see the technical section for details on this.
- Turn the fan on and off by selecting the fan mode. Auto mode will have the fan turn on whenever the controller starts to heat up.

## Encoder

Double-click on the Encoder object, labelled Encoder Input, in order to run the example:

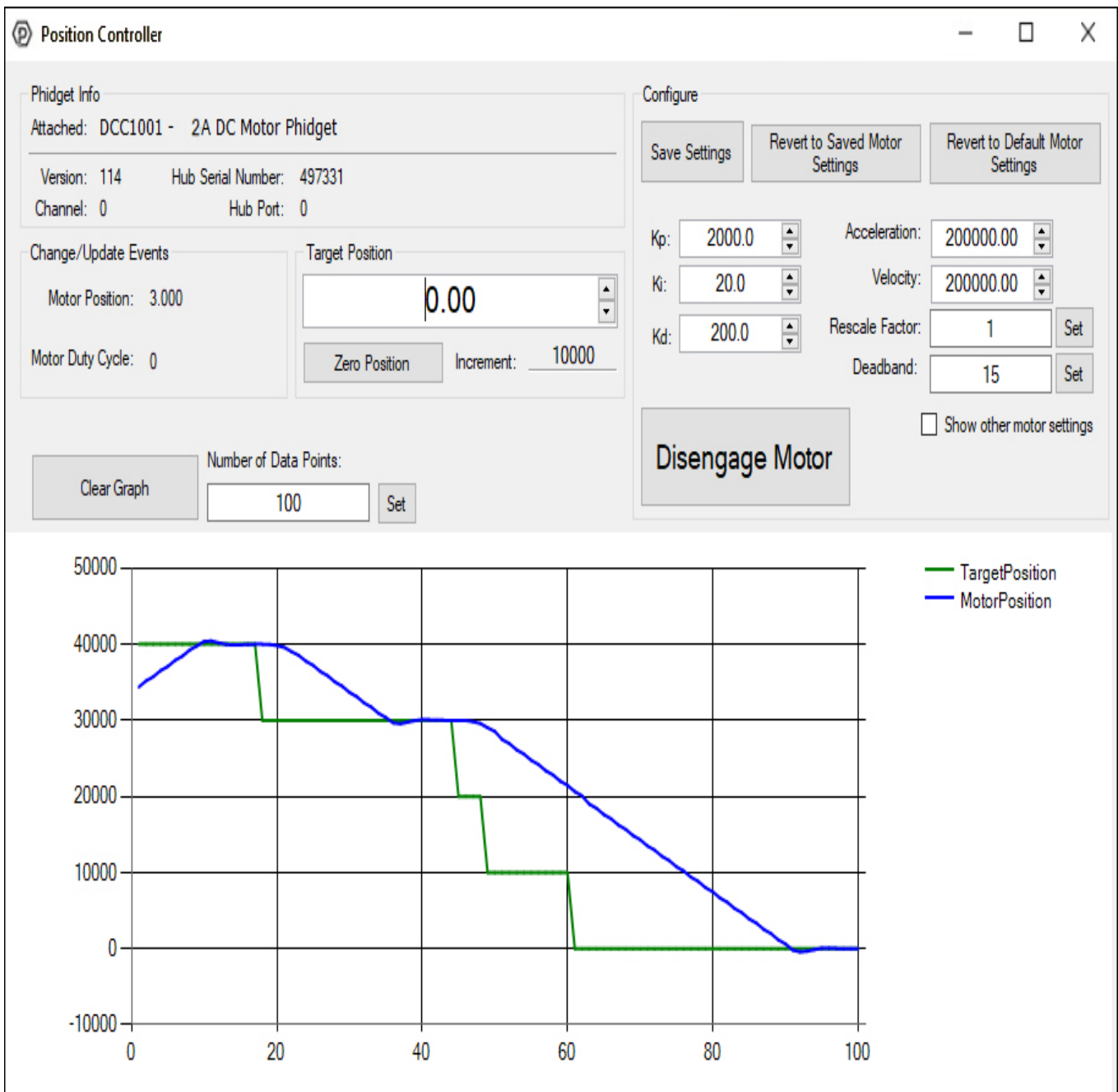


General information about the selected object will be displayed at the top of the window. You can also experiment with the following functionality:

- **Position Change:** the number of ticks (or quadrature cycles) that have occurred since the last change event.
- **Time Change:** the amount of time in milliseconds that has elapsed since the last change event.
- **Position:** the total position in ticks relative to where the encoder was when the window was opened.
- **Index Position:** the position where the index channel was last encountered. Some encoders do not support index, check your encoder's datasheet for more information.
- **Velocity:** the average velocity in rotations per second. A CPR must be specified to enable this functionality.
- Specify a counts per revolution (CPR) value to enable velocity calculation.
- Modify the change trigger and/or data interval value by dragging the sliders. For more information on these settings, see the data interval/change trigger page.

## Position Controller

Double-click on the Position Controller object in order to run the example:



General information about the selected object will be displayed at the top of the window. You can also experiment with the following functionality:

**Note: a video describing the use of this program is available below in the Technical Details section.**

### Configuration

- It is recommended to set the Rescale Factor first. This will change the units of your controller. For information on the rescale factor, visit the technical details section below.
- You can set the control parameters Kp, Ki and Kd in order to change the behavior of the control loop. You can save these variables into the program so you don't have to re-enter them manually (NOTE: This does not store the settings on the DCC1001, it simply saves them inside the control panel program, so you'll have to re-enter them if it's used on another computer).
- The Velocity Limit and "Acceleration can be set in the top right. These values will be used to create a motion profile that the controller will try to track.
- Use the Deadband to determine how exactly the controller will try to reach the target position. View the API for a detailed description of how Deadband works.
- Click on Show Other Motor Settings' to access Stall Velocity. This is a safety feature which protects your hardware. View the API for a detailed description of how Stall Velocity works.

### Other

- Change the Target Position to select which position you want the motor to try to reach.

- Press the Engage Motor button to allow the motor to start, and press again to stop it.
- You can Zero Position to add a position offset which will return the motor's position to 0.

#### Graph:

- You can view the control data on the graph. The green line is the selected target position, and the blue line is the motor's actual position as reported by the Hall Effect sensors. When the two lines meet, the motor will stop moving and attempt to hold that position, even if moved by an external force. You can Clear the graph, and customize the number of data points shown at any given time.

## Technical Details

### Control Loop Parameters

In order to get the desired behavior from your controller, you will have to tune your control parameters. This video explains the tuning procedure and gives information on how the controller works.

#### Motor Position Controller



### Further Reading

For more information, have a look at the DC Motor and Controller Primer.

### What to do Next

- Software Overview - Find your preferred programming language here to learn how to write your own code with Phidgets!
- General Phidget Programming - Read this general guide to the various aspects of programming with Phidgets. Learn how to log data into a spreadsheet, use Phidgets over the network, and much more.
- Phidget22 API - The API is a universal library of all functions and definitions for programming with Phidgets. Just select your language and device and it'll give you a complete list of all properties, methods, events, and enumerations that are at your disposal.

