RoboJackets Electrical/Firmware Soldering Training Guide

Bernardo Perez and Andrew Roach

November 15, 2021
v1.0

Contents

1 Background

The purpose of this lab is to give you experience soldering common components onto a PCB and debugging the soldered board.

For the first part of the lab, all the components and the board have been provided for you to work with. Additionally, to speed up the process, some components have been soldered onto the provided boards beforehand. Ideally, you should be able to finish the soldering session within the given time frame. However, if at the end of the soldering time frame you have not finished soldering components onto your board, you may continue soldering during the debugging section or ask an instructor to help you get to a point where you can continue with debugging the board.

For the second part of the lab, you will be debugging the provided board. One of the pre-soldered components are guaranteed to have an issue with them, so you will be responsible for determining which of the components are at fault.

1.1 Arduino IDE

If you already have the Arduino IDE installed and are familiar with using it, you can skip the rest of this section. If you have never used the Arduino IDE before, that's OK! Read on to learn all you need to know for this lab.

Download the Arduino IDE here.

We can write programs to the Arduino by creating 'sketches' which are written in the C++ programming language. These have a .ino extension and can best be edited within the Arduino IDE. The IDE consists of an editor and console which will serve as your primary tools to edit and check your program.

The two most important buttons in the Arduino IDE are the "Verify" and the "Upload" buttons.

- The checkmark is the "Verify" button. Pressing this button will compile your code. However, the "Verify" button does NOT flash your program onto the Arduino Uno.
- The arrow pointing to the right is the "Upload" button. Pressing this button will upload your code onto the Arduino Uno. Once your computer finishes uploading your compiled program to the Arduino, the Arduino should automatically run the uploaded program! Just to be sure your uploaded code is running, you can press the reset button on the Arduino.

However, before you try uploading code to the Arduino, make sure that the Arduino Uno is connected to your computer and that the correct port is selected under Tools → Port.

A typical program is split into two parts: the setup() function and the loop() function:

- setup(): This function is called when the program starts. It'll run once when you power on or reset the Arduino board. Here is where you will initialize variables, set pin modes, etc.
- loop(): This function loops consecutively so the code you place here will constantly run. You can have other functions in your program but they will not run unless they are called in setup() or loop().

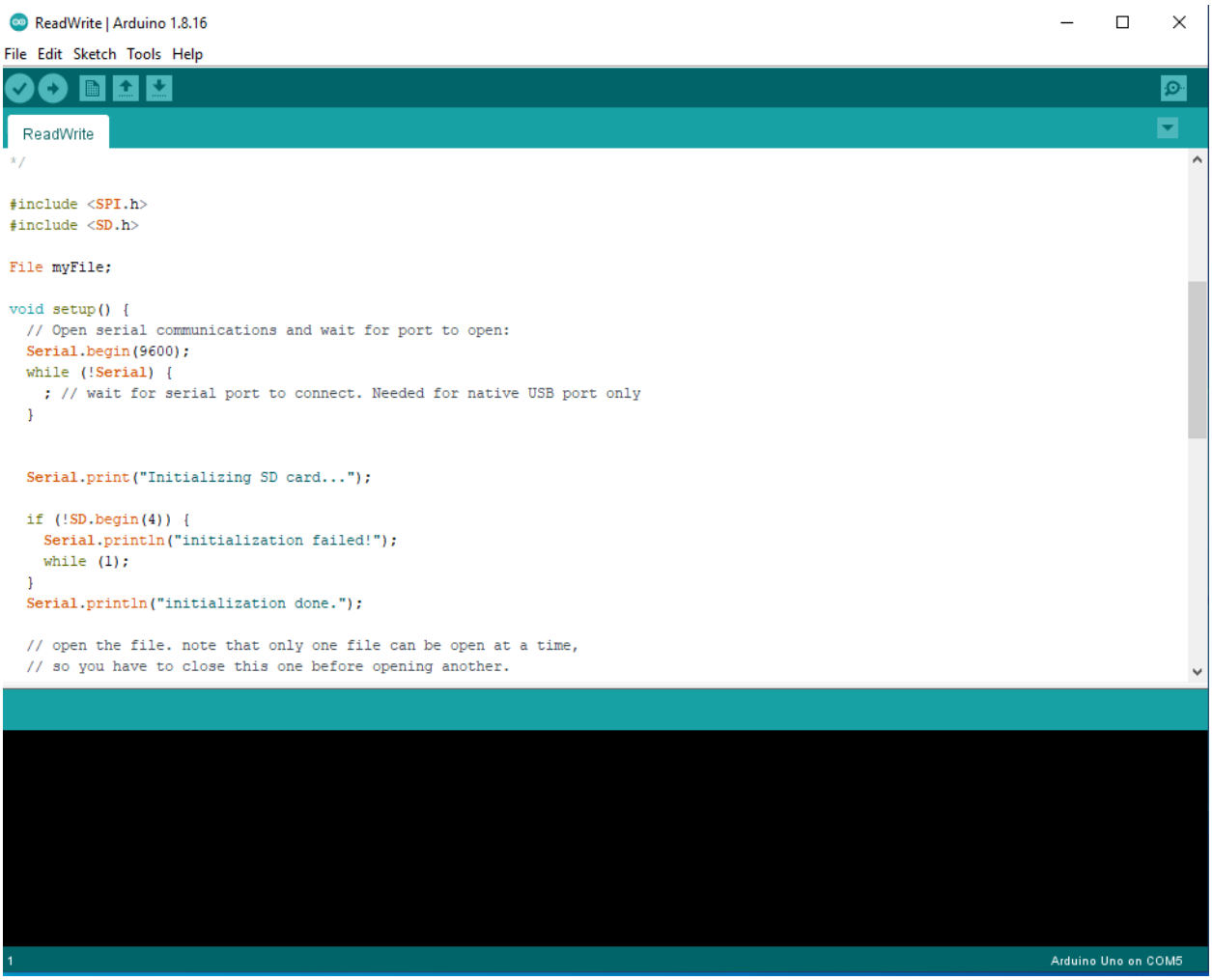


Figure 1: The Arduino IDE

Figure 2: The Verify and Upload buttons are pictured on the left-hand side of the blue bar

2 Objective

This lab is split into two parts: a soldering section and a debugging section. We will give presentations on tips and tricks to tackle each section and, of course, you can always ask the instructors if you need help!

2.1 Soldering

The soldering section will have you soldering a set of predetermined components onto the board. Some components will be optional. The required and optional components are listed in section 4.1 and visualized in 4.2. Try to solder all of the components labeled as "required" by the end of the soldering period.

2.2 Debugging

The debugging portion of the lab will require you to find the one pre-selected component that is soldered incorrectly. Additionally, you may have to correct any mistakes you may have made in the soldering process.

3 Materials

The full list of materials is as follows (*already soldered):

- C1 0.1uF
- C2 1uF
- C3 0.1uF (SHOULD NOT BE SOLDERED)
- F1 350 mA*
- J1 PJ-037A*
- LED1 ORANGE
- LED2 GREEN*
- LED3 ORANGE
- LED4 GREEN*
- OUT 1X2-3.5MM
- Q1 DMN2056U
- R1 1K
- R2 2.4K*
- R3 1K
- R4 10K*
- R5 120K*
- R6 10K*
- R7 2.4K*
- S1 EG1903-ND
- SW1 PTS645
- U1 DRV8871
- Arduino Uno

You may have to look at the data sheet for any of these components if you are unsure about their internal connection.

4 Guided Lab

4.1 Downloading the Arduino IDE

In this lab, you will be using an Arduino Uno to interface with the soldering training board. See Section 1.1 if you don't have the Arduino IDE installed and/or you are unfamiliar with the Arduino IDE.

4.2 Required Soldering Parts

The components you are **required to solder** are listed as follows (If you have EAGLE installed, you can reference the parts with EAGLE, listed as: *EAGLE PART NAME* - *PART DESCRIPTION*)

- OUT - 1x2 3.5mm Terminal Block
- Q1 - MOSFET
- S1 - Slide Switch
- SW1 - Button
- U1 - DRV8871 (DC Motor Driver IC)
- Male-Male through hole pins (Leftmost set of pins on top and bottom of the board)

Additionally, there are a few optional components you may solder onto the board, those being:

- R1 - 1k ohm resistor
- R3 - 1k ohm resistor
- LED1 - orange LED
- LED3 - orange LED
- Male-Male through hole pins (Remaining set of pins on top and bottom of the board)

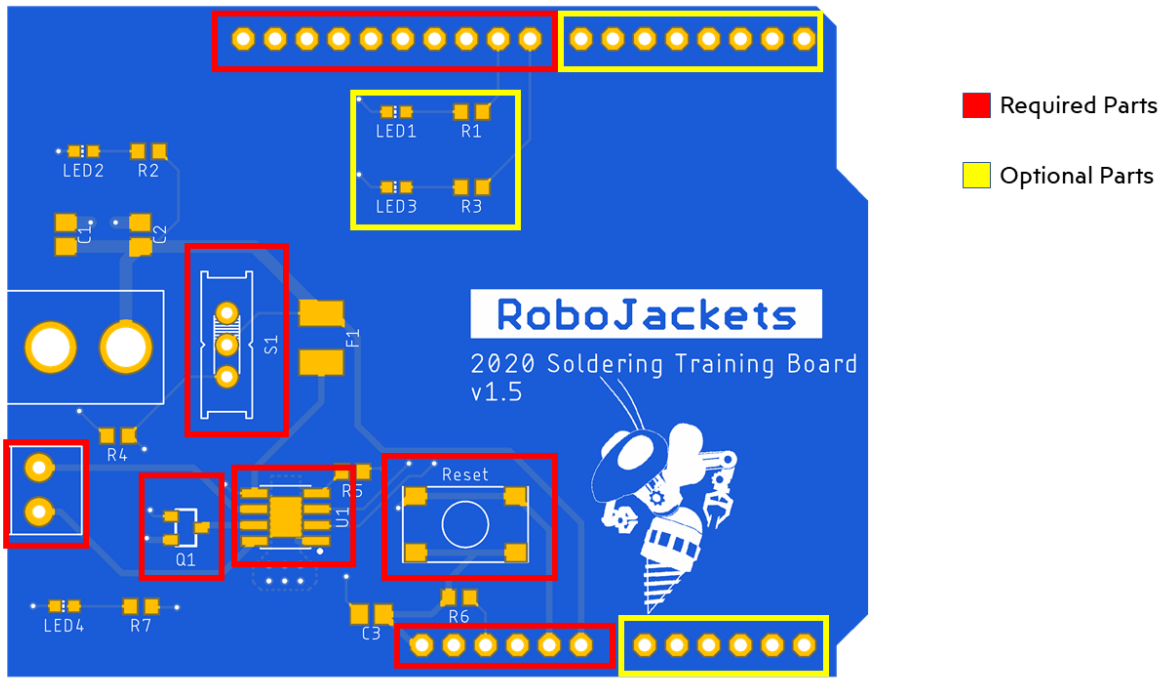


Figure 3: Features required parts to solder and their locations for reference.

4.3 Soldering Tips

- Solder the surface mount components first so that you are working on a level surface. Save the Male-Male through hole pins, slide switch, and terminal block for the end.
- Use tweezers to hold smaller parts or to make fine adjustments to the part's placement.
- For surface mount components, you should place solder onto one pad prior to placing a component down. Then, hold the soldering iron on this pad to melt the solder and place your component. Remove the soldering iron, and the part is now held in place by this solder. Continue soldering the other joints.
- The orientation of U1 (DC Motor Driver IC) is important. The line on the device should be on the same side as the dot on the soldermask of the PCB.
- When soldering the LEDs, ensure that they have the proper direction. Direction on surface mount LEDs is indicated by a small arrow inscribed on the part itself. That arrow should be on the same side as the white indicator on the PCB soldermask.

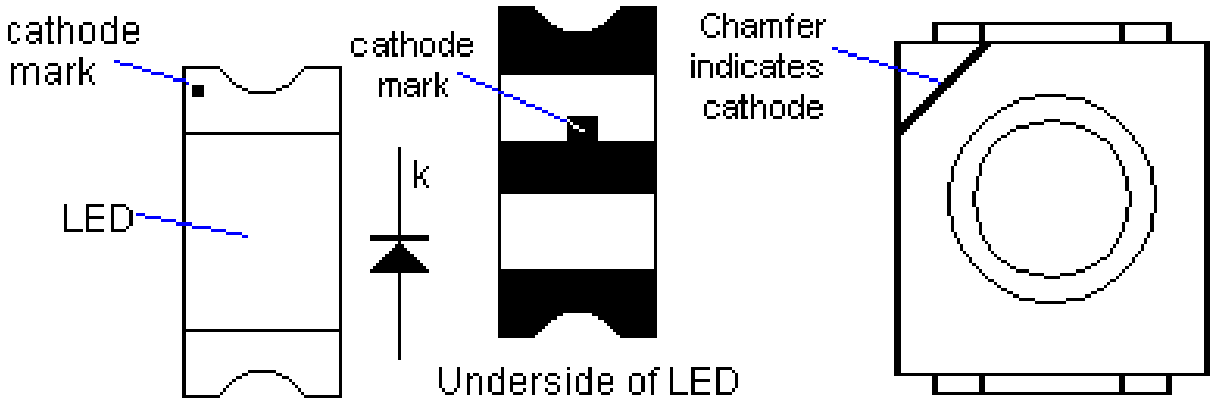


Figure 4: Examples of polarity markings on surface-mount LEDs

4.4 Debugging

Debugging can be a challenging process; in fact, when you go on to work on your own projects, you will likely spend most of your time fixing problems you thought you got right on your first try. In this section of the lab, you will get experience finding errors and fixing them.

We have intentionally placed a hardware bug on each board and a software bug in the Arduino sketch we provided you! In addition to finding these errors, you should also rigorously check your own soldering job for potential errors.

4.5 Debugging Power Issues

Our circuit needs a 12V power supply to work. We will use a bench top power supply to supply the needed 12V. To connect the bench top power supply to the training board, use the provided barrel jack connector and jumper wires. However, do NOT plug your board in yet! You first need to visually inspect your board and check for short circuits.

First, visually check your solder connections. Ensure that the solder is clearly securing each component to the solder pads. Next, check for solder bridges, as shown in Figure 4. A solder bridge may cause a short circuit, so you must fix these before proceeding!

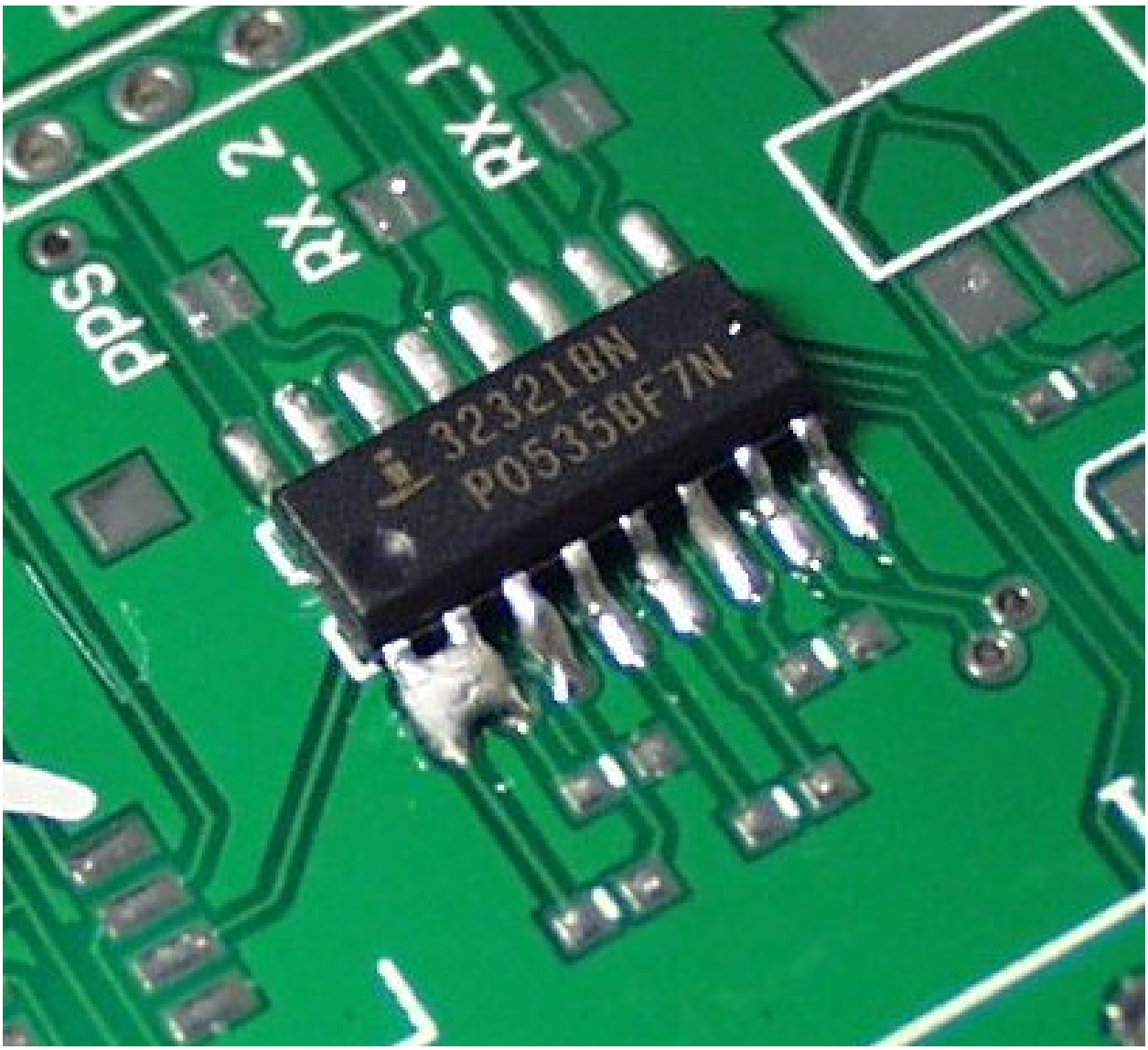


Figure 5: Example of a solder bridge. Watch out for these!

Figure 6: Continuity symbol on a multimeter.

In addition to a visual inspection, you should also use your multimeter to check for power-ground continuity. First, put your multimeter in continuity mode; the multimeter should beep if you touch the two leads together. Put one lead on ground and probe the power rails with the other lead. If the multimeter beeps, you have a short to ground, which is no good!

If you do have a short to ground, you need to thoroughly investigate the cause. Try probing around the area that is shorting to ground and give it a thorough visual inspection. Through trial and error, narrow down the area where the fault could be occurring. If you are completely stumped, ask for help!

If you have done a thorough visual inspection of your board and can't seem to find any shorts to ground, you should try powering your board on. When doing so, connect the board to the power supply IN THIS ORDER.

- Turn on the power supply. Adjust the voltage to 12V.
- Turn off the power supply.
- Connect the board to the power supply.
- Turn on the power supply.

You should always connect circuits to benchtop power supplies in this order. Otherwise, you run the risk of turning on the power supply to discover that the power supply is supplying a much higher voltage than you intended, thus ruining all of your hard work.

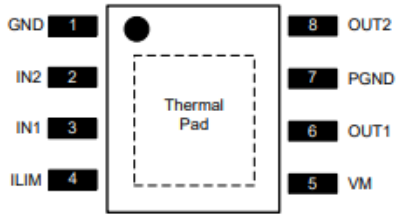
Once you have power, you can check that everything on the board that should be getting power is getting power. Switch your multimeter into DC voltage mode. Put one lead on ground and probe the power lines and pins at various points. If the multimeter is reading a low voltage on something that should be getting a lot more power, something isn't connected properly! If this is the case, probe around until you find the source of problem. Turn off the power supply and disconnect the board, and then go about fixing the problem.

Please note that the power LED might be soldered on incorrectly; therefore, do not use it for debugging purposes!

One of the most crucial parts to test is the DRV8871 IC. The pinout of the IC is shown in Figure 6. Take note of pin 5: the power supply voltage. If this chip isn't getting power, the motor isn't gonna spin. If there is 0V on pin 5, follow the PCB traces and use your multimeter in voltage mode to find where you rediscover the supply voltage.

5 Pin Configuration and Functions

DDA Package
8-Pin HSOP
Top View



Pin Functions

PIN		TYPE	DESCRIPTION	
NAME	NO.			
GND	1	PWR	Logic ground	Connect to board ground
ILIM	4	I	Current limit control	Connect a resistor to ground to set the current chopping threshold
IN1	3	I	Logic inputs	Controls the H-bridge output. Has internal pulldowns (see Table 1).
IN2	2			
OUT1	6	O	H-bridge output	Connect directly to the motor or other inductive load.
OUT2	8			
PGND	7	PWR	High-current ground path	Connect to board ground.
VM	5	PWR	6.5-V to 45-V power supply	Connect a 0.1-µF bypass capacitor to ground, as well as sufficient bulk capacitance, rated for the VM voltage.
PAD	—	—	Thermal pad	Connect to board ground. For good thermal dissipation, use large ground planes on multiple layers, and multiple nearby vias connecting those planes.

Figure 7: The Pin Descriptions of the DRV8871 Integrated Circuit.

4.6 Debugging Firmware Issues

Once everything on the board seems to be getting power, attach the DC motor to the training board. Next, fire up your computer and download the provided .ino file. In the Arudino IDE, go to File → Open, then navigate to where you downloaded the .ino file. With the training board connected to the Arduino, Verify and Upload your program to the Training board.

We intentionally placed a bug in the firmware code for you to find and fix, so even if your board's hardware is working perfectly fine, it will still not work because there is a bug in the firmware. In general, since you usually don't have access to a debugger and since the electrical hardware is another possible point of failure, debugging firmware can be very challenging. However, if you didn't rush to plug you board into the Arduino and carefully checked your board using your multimeter, you can be pretty confident that your remaining errors must lie in the firmware.

Here are some tips relevant to this lab:

- Try different outputs. Comment out the original code and create a simpler output (Tip: if you select several lines of code with your mouse, you can comment out multiple lines using CTRL + / on Windows, CMD + / on Mac). For example, try running the motor only forward or only in reverse. By reducing the number of variables affecting the firmware bug, you will have an easier time spotting it.
- Trace through the code. Find where in the code the Arduino is sending out signals and what pins those outputs correspond to.
- Do a "reality check"; is the Arudino actually outputting to the correct pins? Use an oscilloscope to view the signals the pins are sending. Basically, for each oscilloscope probe, hook the oscilloscope probe onto the Arduino pin in question and clip the black alligator clip to ground. With the board powered on and the Arudino running, press the "Auto" button.

5 Troubleshooting

This document should cover most questions you should have. If you need additional help, call over a trainer! If you need help with soldering, do not hesitate to call over a trainer. Seeing someone demonstrate various soldering techniques can help immensely.

Don't rush through the debugging portion; take your time and think! Use the techniques discussed above to convince yourself that your board must work! If you carefully eliminate each potential problem using the tools at you disposal, your board will (eventually) work!