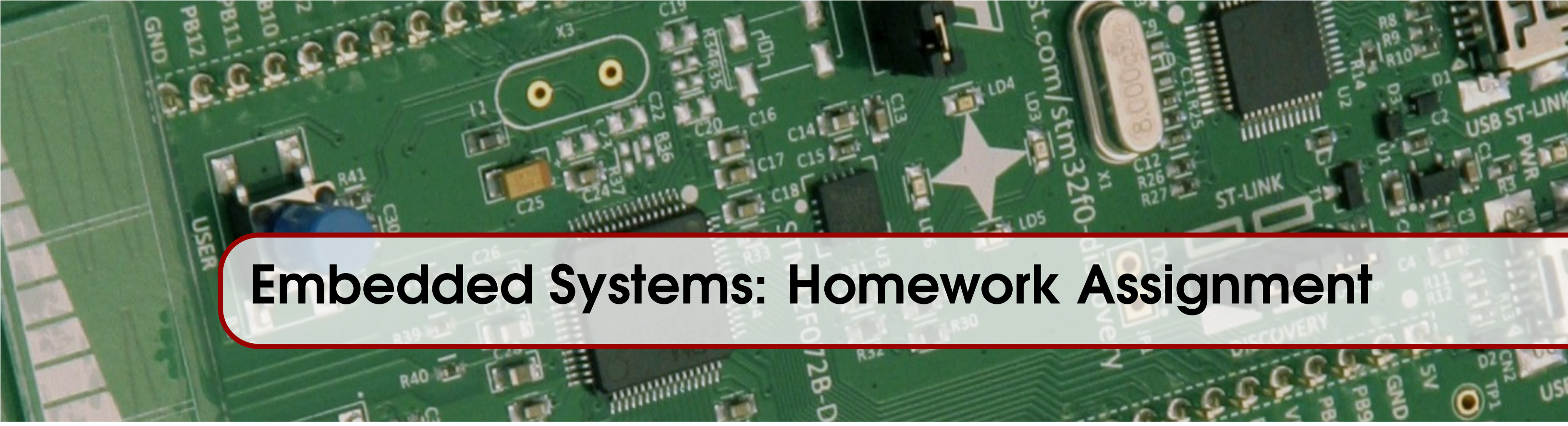


Embedded Systems: Homework Assignment

0.1 Overview

The principal design challenges in embedded systems stem from their interaction with physical processes, and are therefore much more diverse field than computer programming or electrical engineering alone. To become a truly competent embedded systems designer, one must have an intimate understanding of many engineering fields, including: programming and software design, hardware architecture, control theory, circuit theory, PCB design, *etc*. The homework and labs in this course are specifically taylored to cover a wide range of disciplines familiar to an embedded systems engineer. Specifically, you will guided through the design and control of a DC motor driver, using the components shown in Figure 1. The system will consist of (1) an STM32F072 DISCOVERY development board, (2) a custom-made extension board that contains all the required power interface between the DC motor and the STM microcontroller, and (3) the brushed DC motor itself. In the labs you will learn about the STM microcontroller, and will primarily focus on software development. In the homework you will design, fabricate, and assemble your own custom motor driver board. By the end of the semester, you will acquire the skills needed to realize a complete cyber-physical system.

0.2 Homework Objectives

The goal of this homework assignment is to provide you with hands-on experience with PCB (Printed Circuit Board) design and assembly of a custom DC motor driver. You will learn to read and interpret datasheets, draw detailed schematics using modern CAD software, design PCB layouts, and hand-solder components onto the PCB. The homework will be split into three distinct assignments. Part 1 will cover the schematic drawing, Part 2 will cover the design of the PCB layout, and Part 3 will cover the final assembly and testing your custom PCB. The homework is purposefully sectioned with three consecutive due dates so that your design may be periodically verified for correctness and completeness throughout the process, as it is essential that you produce a functional board. Figure 2 shows the process flow as you move through the homeworks.

To realize your PCB schematic and layout, we will use a common software suite for Electronic Design Automation (EDA) called EAGLE. EAGLE is available on Windows, Linux, and Mac OS X as freeware – so feel free to install it on your personal machines. It is also available on the lab computers in CADE, Engman, and the MEB Digital Lab.

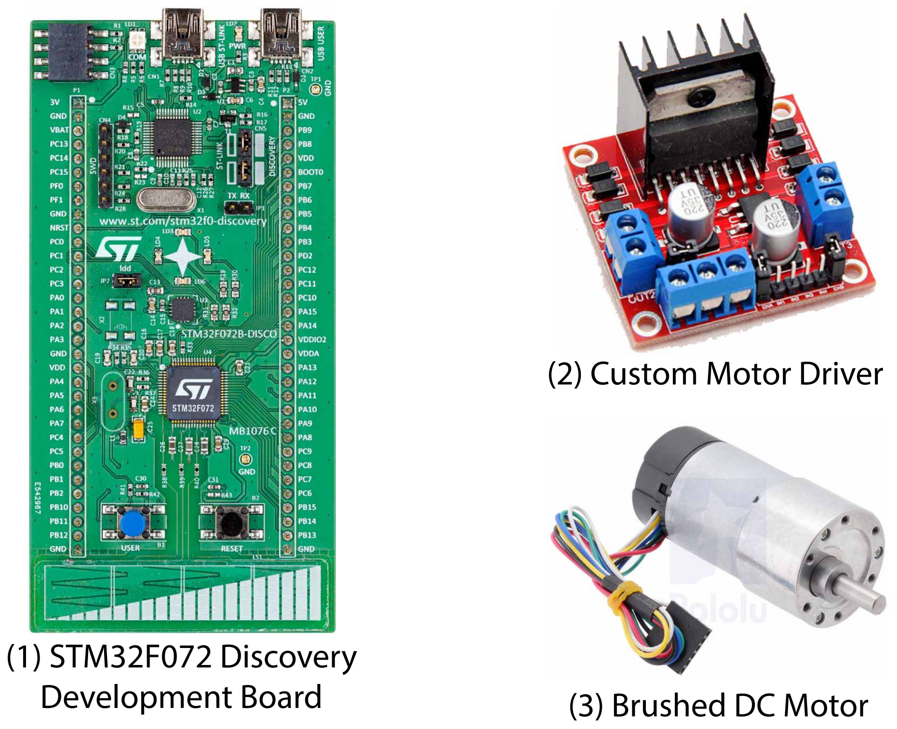


Figure 1: Components used to realize complete cyber-physical system.

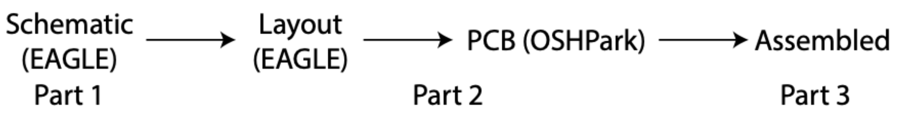


Figure 2: Production flow of custom motor driver PCB and assembly.

0.3 Introduction to PCB Design

If you have never designed a PCB before, which is true for the majority of the class, it may seem quite overwhelming. You might have tons of questions and not even know where to begin. We therefore provide you with this introduction to PCB design to answer your most pressing questions and hopefully mitigate your initial anxiety regarding the topic.

As an embedded systems engineer you are required to pertain an intimate understanding of many engineering fields, from computer architecture, to circuit theory, to computer programming. Something that many students do not initially understand is that in order to appropriately program a system, it is necessary to know how the physical pieces function and interact, which is not always true for higher level computer programming. One piece of knowledge that is expected from every embedded systems engineer is the ability draw electrical schematics of physical electrical systems, as well as the resulting implementation. We are referring of course to Printed Circuit Board (PCB) design.

From Wikipedia, a PCB is described as follows:

> A PCB mechanically supports and electrically connects electrical components using conductive tracks, pads, and other features etched from one or more sheet layers of copper laminated onto and/or between sheet layers of a non-conductive substrate. Components are generally soldered onto the PCB to both electrically connect and mechanically fasten them to it.

In the specific case of this homework assignment, we would like to design a PCB that will mechanically support and electrically connect an integrated circuit (IC) used to provide power and control to a small DC motor. We will send electrical signals from the microcontroller (the STM32F072 DISCOVERY development board) to the IC, which will be soldered to your custom PCB. This IC will in turn generate much more powerful signals (with the help of an external power supply or battery) to power and control a DC motor. Your PCB will also house a digital temperature sensor IC, which you can again communicate with using the microcontroller I^{2}C interface.

There are several basic, fundamental steps every engineer will follow when designing a PCB. The first step is defining a clear objective for the design. In our case, we would like to design a board that will translate low-power commands from the microcontroller into powerful electrical signals capable of controlling the speed and direction of a DC motor. With a good definition of the objective in mind, we can begin looking for the necessary components. Luckily, we provide you with a full list of the components to incorporate, which is known as a Bill of Materials (BOM). This brings us to the next (and most difficult) step in the design process: connecting the components. For every component you find online, there will be an accompanying datasheet that gives you all the necessary information about this component. Inside of the datasheet, you will find information about power consumption and ratings, necessary supporting circuitry, physical dimensions, and other important information. It is crucial that you, as an engineer, know how to interpret these documents in order to appropriately and safely implement these components.

Interpreting Datasheets. The first time you open a datasheet, you'll be stricken by the sheer amount of information they provide. However, you'll soon notice a structured, universal format that will help you quickly parse them for helpful information. Some of the most useful bits of information are:

- **The device pinout.** This is a simple image of the chip with the pins numbered, and a table showing the pin numbers, names, and a description for each pin. This is essential for knowing the function of each pin, as well as how to connect them in your schematic.

- **Electrical Characteristics and Maximum Ratings.** These tables detail (among many other characteristics) the maximum voltage each pin can tolerate, as well as the necessary current draw they will source. This information is necessary so that you do not damage the pins by using too much voltage and other electrical nuances. These tables also give information regarding the *nominal* voltage, which refers to the recommended voltage for the device. For example, a pin might be *tolerant* of 5V, but have a nominal voltage of 3.3V. Sourcing 5V to the pin probably won't damage it instantly, but it puts the device at much higher risk of becoming damaged over time.
- **Application Note/Proposed Uses.** A well written datasheet will provide some intended uses for the device, and details for controlling the system. These will typically be shown alongside schematics, and possibly some example code/psuedo-code.
- **Proposed Schematics.** Almost every IC you use will require some additional components, such as resistors and capacitors, to help smooth out and filter electrical signals.
- **Mechanical Dimensions and Recommended Land Pattern.** This information is used to design the physical layout of the pads on your PCB, so that the component will have a clean solder joint and electrical connection.

A datasheet may contain less information than this (if it is not well written), but in most cases will provide a great deal more information than we have given here.

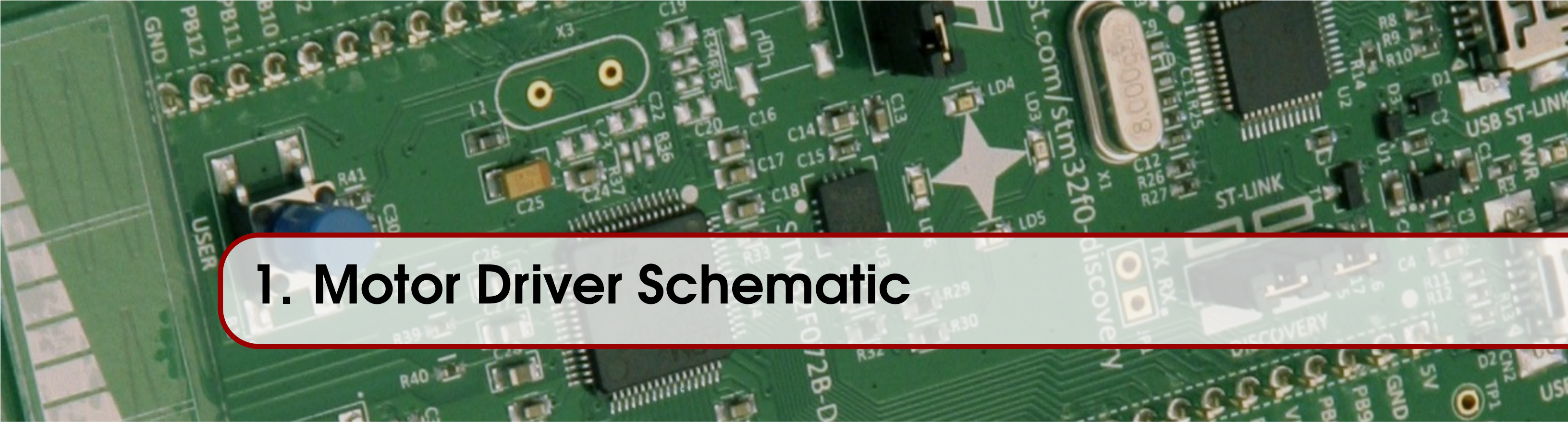
Using the component datasheets and the recommended schematics, you have all the information you need to design the schematic of your PCB. Using EAGLE, you can draw these components (or import someone else's component designs) and connect them to create your device schematic.

From the schematic, it is very straightforward to design the actual PCB layout. For simple designs, such as this homework, layout is more of an art than it is engineering. Your main challenge will be to organize the components in a way to minimize wire crossings and to ensure the final PCB is aesthetically pleasing. Indeed, as with many natural system, a simple and aesthetic organization is often efficient. In the case of PCB design, it will lead to smaller loses, parasitic, crosstalk, etc.

Once our PCB layout is designed, we can collect the CAD files and send them to a PCB manufacturer. There are thousands of companies that will fabricate your PCB around the world, and, for our course, we will be using OSHPark, a company located in Oregon, USA. Manufacturers will usually take 2-3 weeks to fabricate your design and ship it back. During this time, you'll typically order all the components to solder on the PCB. Vendors such as Mouser and Digikey are the most popular suppliers in the US, and these are where we get almost all our components. We have already purchased all the components listed in the BOM and created a kit for sale at the stockroom, so you don't have to worry about ordering them online.

The final step is soldering the components to your PCB. If you're doing a mass production of these PCBs, you'll likely have an external company do the soldering with automated robots, but prototypes will typically be soldered by hand. We cover soldering in detail in Chapter 3 of this paper.

Hopefully this short introduction has dispelled some of the initial questions you have about the basic concepts of the homework.



1. Motor Driver Schematic

In Part 1, you will draw the schematic for your motor driver using EAGLE PCB Design Software. Listed below are some tutorials and introductions to EAGLE. We highly recommend you make use of them as you complete the homework assignment if you haven't used EAGLE before, and especially if you have never used a PCB design tool. Readings and Tutorial Videos EAGLE is very popular with the hobbyist and educational communities and has a lot of online tutorials. These come in a variety of formats and depending on your personal preference you may chose to read or watch example projects.

Most of the following sections are taken from a series of tutorials developed and generously shared under a creative commons license by Sparkfun Electronics. Unless you are already familiar with PCB design, it is strongly recommended that you read the following sections and visit the full tutorials online.

EAGLE User Manual:

- Available from the EAGLE control panel (main window) after expanding the "Documentation" tab in the left hand list.
- Manuals can be found under the "doc" subfolder of the EAGLE install directory.

EAGLE Tutorial Videos:

- CadSoft EAGLE guided tour [https://cadsoft.io/tour/]
- Jeremy Blum & Element14 EAGLE Tutorials [http://www.jeremyblum.com/category/eagle-tutorials/]
- Tangentsoft EAGLE and Soldering Tutorials [https://tangentsoft.net/elec/movies/]

Sparkfun EAGLE Tutorial List:

- Basic theory of PCB structure [https://learn.sparkfun.com/tutorials/pcb-basics]
- How to read schematics, EAGLE schematic tips [https://learn.sparkfun.com/tutorials/how-to-read-a-schematic]
- Install and configure EAGLE [https://learn.sparkfun.com/tutorials/how-to-install-and-setup-eagle]
- Designing a schematic [https://learn.sparkfun.com/tutorials/using-eagle-schematic]
- Converting a schematic to board layout [https://learn.sparkfun.com/tutorials/using-eagle-board-layout]
- PCB layout with surface-mount parts [https://learn.sparkfun.com/tutorials/designing-pcbs-advanced-smd]
- Creating custom part symbols and footprints [https://learn.sparkfun.com/tutorials/designing-pcbs-smd-footprints]

1.1 Instructions

The overall system block diagram, shown in Figure 1.1, illustrates the different components that make up the complete motor driver system, including the microcontroller, custom motor driver board and all of its components, and DC motor. Carefully read the description of each component, as they contain crucial information for your design.

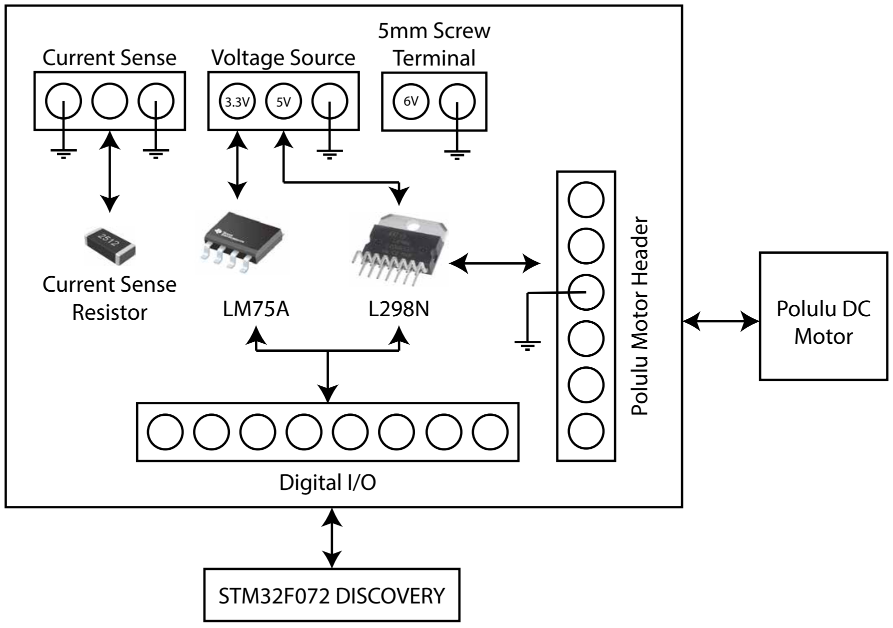


Figure 1.1: System block diagram.

STM32F072 DISCOVERY: This board is the heart of this course, and is the device you will learn about and program throughout the labs. It contains an ARM Cortex-M0 based microcontroller whose pins are broken out to pin headers. Carefully review the datasheet at the URL below and in particular, understand the maximum voltage rating for the different pins. You will eventually write code to control the functions of the pins, and more specifically send commands from the DISCOVERY to your custom motor driver board to control the DC motor. Connections between the discovery board and the extension board will be made through jumper wires. You will use the linked datasheet many times throughout the semester, so note that it is not necessary to know the entire datasheet immediately. [DISCOVERY Datasheet]

Brushed DC Motor: The DC motor that your custom motor driver board will be controlling is a Pololu, item #2824. We will power the motor with 6V from an external power supply in the lab, and the stall current (maximum current) at this voltage is 2500 mA. The 6V power supply will run into the custom motor driver board, where it will be manipulated using signals from the microcontroller, and outputted to the motor on the red and black wires to control the speed and direction of rotation. The four remaining wires (green, blue, yellow, white) are used for the rotary encoder, and are signals coming from the motor, passed straight through the custom motor driver board, and sent to the microcontroller. A rotary encoder is a type of sensor used to measure the speed of rotation and angle of the motor. We will learn more about the encoder during the Timers lab (Lab 6), but for the moment it is sufficient to know that the encoder signals are an output from the motor, an input to the motor driver board, where they are passed straight through and output, and then finally an input to the microcontroller dev board.

It is recommended to keep the 6V value in mind while wiring the H-bridge (described below) and while sizing this current sense resistor (also described below) to obtain the maximum resolution of the STM32F0 Analog-to-Digital Converter (ADC). Recall that Ohm's Law states that the voltage across the resistor is equal to the current running through the resistor multiplied by the resistance ($V = I \cdot R$). [Pololu Motor Datasheet]

H-Bridge: The H-Bridge motor driver (please read about the principle of an H-Bridge) will serve as the power interface between the Polulu motor and the control system (DISCOVERY). We will use a ST L298N chip (MultiWatt package). Carefully study its datasheet, available at the URL below, and understand how this chip is controlled and wired. Note that the 6V power supply connection will be made with the 5mm screw terminal (listed on the BOM), and the remaining control signals will be made with standard 2.54mm pin headers. Please also be careful of the different decoupling capacitances required with this device to operate properly. [L298N Datasheet]

Current Sense Resistor: The current sense resistor is used to measure the current running through the motor. You will see the resistor shown on the H-Bridge schematic datasheet, and the exact resistance and dimensions can be found in the BOM. The resistor is sized according to the STM32F0 ADC range and the maximum current of the motor. Using Ohm's Law, the voltage across the resistor is proportional to the current through the resistor (V = I * R). By feeding this voltage value into one of the STM32F0 DISCOVERY ADC pins, we can read the voltage and consequently the motor current. In order to shield the sensitive voltage value from ambient electromagnetic radiation as it travels from your custom PCB to the DISCOVERY, please put a ground pin on either side of the voltage pin, resulting in a 3-pin header.

Temperature Sensor: In order to monitor the temperature around the H-bridge, we will add a Texas Instruments LM75A Digital Temperature Sensor. It is strongly recommended to read its datasheet at the URL below. This device will communicate through I2C (a serial communication protocol) with the microcontroller. The I^{2}C signals, along with the power supply and digital control of the H-bridge, will be on a standard 2.54mm pin header. [LM75A Datasheet]

Table 1.1 shows the pins required in your design. This table may be used as a reference to help learn about the functionality of the ICs as well as the schematic and physical layout.

1.2 Creating The Schematic

This section covers the initial steps required to start designing a schematic within EAGLE. It is a condensed version of the SparkFun tutorials "How to read a schematic" and "Using EAGLE: Schematic." It assumes you have a basic idea about electric circuits and common components such as resistors and capacitors. If not then you will want to review a basic circuit tutorial, such as https://learn.sparkfun.com/tutorials/what-is-a-circuit, and let us know of any questions you may have.

1.2.1 Creating a New Project

After loading, EAGLE always starts with the "Control Panel" window. The control panel provides a quick access point to all previously opened projects as well as parts libraries, documentation and user scripts. The following image shows the control panel with a new project selected. You can access the built-in user manuals by opening the "Documentation" tap in the left-hand list.

- Create a new project by using (File $\rightarrow$ New $\rightarrow$ Project)
  - EAGLE projects are simply folders that will contain all of your schematic and board design files. You should give the new project a simple name to describe its contents.

Table 1.1: Custom Motor Driver Minimum Required Pinout

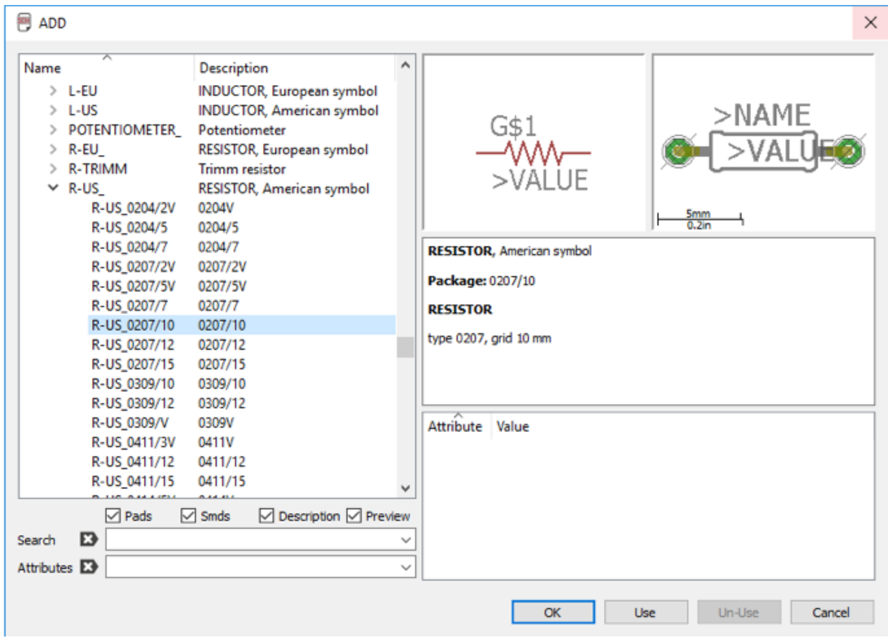
Pin Name	I/O	Termination	Description
ENABLE	I	L298N	PWM from STM32 to control motor speed
INPUT1	I	L298N	Signal from STM32 to control motor direction
INPUT2	I	L298N	Signal from STM32 to control motor direction
I_SENSE	O	L298N	Analog value going to ADC to determine motor current. Remember this pin is on its own header with a ground pin on either side
OS	O	LM75A	Over-Temperature bit, can be used by STM32
SDA	IO	LM75A	I2C data line, be sure to pull up!
SCL	IO	LM75A	I2C clock line, be sure to pull up!
ENC_OA	O	Motor Header	encoder signal, output pin from motor ENC_IA
ENC_OB	O	Motor Header	encoder signal, output pin from motor ENC_IB
MTR_BLK	O	L298N	Motor power. Part of 6-pin motor header
MTR_RED	O	L298N	Motor power. Part of 6-pin motor header
GND	O	-	Motor ground. Part of 6-pin motor header
5V	O	-	Motor encoder power. Part of 6-pin motor header. Sourced from 5V input pin.
ENC_IA	I	Pass-through	Motor encoder. Pass through to output pin. Part of 6-pin motor header.
ENC_IB	I	Pass-through	Motor encoder. Pass through to output pin. Part of 6-pin motor header.
Motor Header	IO	Polulu Motor	The 6-pin motor header. Be sure to order the pins correctly!
6V	I	L298N	Provides motor power (5mm screw terminal). Sourced from lab power supply.
5V	I	L298N	Logical voltage supply for L298N signals. Will be sourced from STM32F0 DISCOVERY.
3V3	I	LM75A	Logical 3.3V voltage supply for LM75A power and I2C pullups. Will be sourced from STM32F0 DISCOVERY.
Ground	IO	-	Multiple GND pins sourced from power supply and STM32.

  - After creating the new project, notice that there is a green circle next to the list entry for it. This indicates that the project is loaded and any new schematics or board design files will be associated with it.

- Add a schematic to the new project either by right-clicking on the project entry and selecting (New $\rightarrow$ Schematic) or by using the (File $\rightarrow$ New $\rightarrow$ Schematic)
- Once the schematic window opens, you can save the new empty schematic and it should appear under the project entry in the control panel.
  - Schematic files have the extension ".sch" but may not automatically identify with EAGLE when clicked in a folder. You will always want to load your files through the control panel window.
  - Name your schematic such that we can identify it once you submit the assignment. A good name would be "HW1_NAME_UNID.sch"

1.2.2 Adding Parts to a Schematic

- An empty schematic isn't all that useful. You will want to add components to the schematic by selecting the "Add Component" button on the left-hand toolbar. (Edit $\rightarrow$ Add)
- The first time you add a component to a schematic may take a while as EAGLE loads all of the built-in part libraries.
  - These part libraries contain hundreds of existing schematic symbols and PCB footprints for use. Unfortunately this makes them somewhat difficult to navigate.
  - Many of the online tutorials use their own component libraries. Companies such as Adafruit or Sparkfun often publish libraries containing the parts they sell.
- For basic parts such as resistors, capacitors and connectors you are allowed to use existing component libraries.
  - For parts such as the motor driver and temperature sensor you should create your own library and part symbols.
  - Sparkfun has a good tutorial for creating your own parts.
  - For including libraries from other sources, see the Setup Tutorial.



Name	Description
L-EU	INDUCTOR, European symbol
L-US	INDUCTOR, American symbol
POTENTIOMETER_	Potentiometer
R-EU_	RESISTOR, European symbol
R-TRIMM	Trimm resistor
R-US_	RESISTOR, American symbol
R-US_0204/2V	0204V
R-US_0204/5	0204/5
R-US_0204/7	0204/7
R-US_0207/2V	0207/2V
R-US_0207/5V	0207/5V
R-US_0207/7	0207/7
R-US_0207/10	0207/10
R-US_0207/12	0207/12
R-US_0207/15	0207/15
R-US_0309/10	0309/10
R-US_0309/12	0309/12
R-US_0309/V	0309V
R-US_0411/3V	0411V
R-US_0411/12	0411/12
R-US_0411/15	0411/15

RESISTOR, American symbol

Package: 0207/10

RESISTOR

type 0207, grid 10 mm

Attribute	Value

- Once the ADD window opens, navigate in the left-hand part of the window until you find a parts library called “rcl”
  - Once you expand “rcl” you will find a number of sub-categories, expand the “R-US” category and select a random entry.
  - Once selected, the right-hand parts of the window show the schematic symbol, PCB footprint and other attributes of the selected part.
  - Notice that all the parts in the “R-US” category are resistors and share the same schematic symbol. Why so many duplicate parts? Although all these parts are resistors and behave identically in the schematic, they are all physically different when placed on the actual PCB.
  - When selecting parts from a library, you will need to make sure to select the appropriate version to match the physical parts you’ll be using.

1.2.3 Selecting Parts offered in the Stockroom

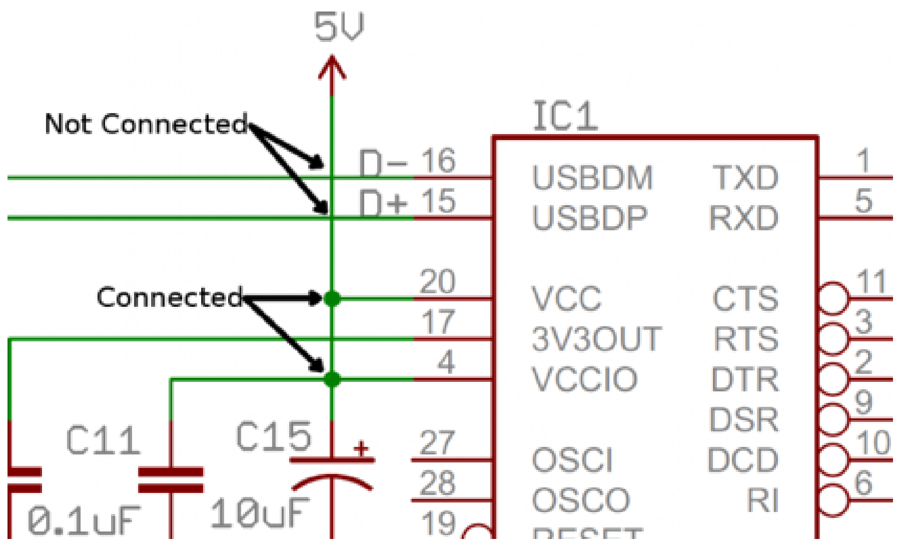
The stockroom offers kits containing all of the parts you’ll need to build the motor driver. These kits use mostly surface-mount parts so you’ll need to select SMT (surface-mount) PCB footprints that match the size of parts offered. Please refer to the attached Bill of Materials for detailed information about every component included in the kits.

Making Schematic Connections

- After selecting a part in the ADD window, press OK to return to the schematic.
  - You should have a copy of the selected part on your cursor, you can rotate the orientation of the part before placing with the right-mouse button.
  - You can continue to add copies of that part by clicking until you press ESC or click the red stop button on the top toolbar.
- To create circuit connections between the pins of the parts you placed, click the “Net” button on the left-hand toolbar.

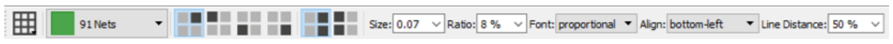
  - Using the “Net” tool you can click on a pin of a component and then a net/wire connection will follow the cursor.
  - You can click on the schematic to make corners and other geometry for routing
  - Click on another pin to end the current net.

- You can change the routing style (right-angle, 45-degree, or direct) by clicking on the right-mouse button while routing.
- You can join multiple nets by clicking on the middle of an existing line while placing a new net.
  - When joining nets, EAGLE should always place a junction “node” on the intersection point.
  - Without a junction, there is no electrical connection in the schematic even if the nets intersect.



Preparing The Schematic for Submission

- While not strictly necessary, a page frame makes the schematic look a lot neater.
  - Page frames are added like a circuit component, for a standard letter-size paper use the frames→FRAME_A_L component.
- You can document the value of passive components such as resistors with the “Value” tool. This doesn’t have any real effect on the schematic, but makes it much clearer to anyone attempting to understand it.
- You can place arbitrary text using the “Text” tool. The default text size is pretty small, after typing your text in the edit box, press enter or click OK. Your text should be at the cursor, waiting for you to place it.
  - While the text tool is active there is a secondary top toolbar that allows you to select the text layer, size and font style.


91 Nets Size: 0.07 Ratio: 8 % Font: proportional Align: bottom-left Line Distance: 50 %

- When submitting the first part of the assignment, you will want to upload the “.sch” file. Eventually in the later parts of the assignment you will create board layouts which are contained in “.brd” files.

1.3 Adding Peripherals

You can add whatever peripherals, chips, or sensor you like to the board. The following are some standard ones that you will need on almost any embedded system. Add at least one 0.1µF and one 1 µF capacitor between your power rails. These are called decoupling capacitors (decap) and are used to smooth the voltage and absorb spikes (so connect it between VCC and GND). If you use LEDs, then add another 10 µF and 1 µF capacitor onto the power rail for the LEDs. Also make sure of adding test pins. It will be extremely useful during debugging to grip a scope probe or a pin of your logic analyzer.

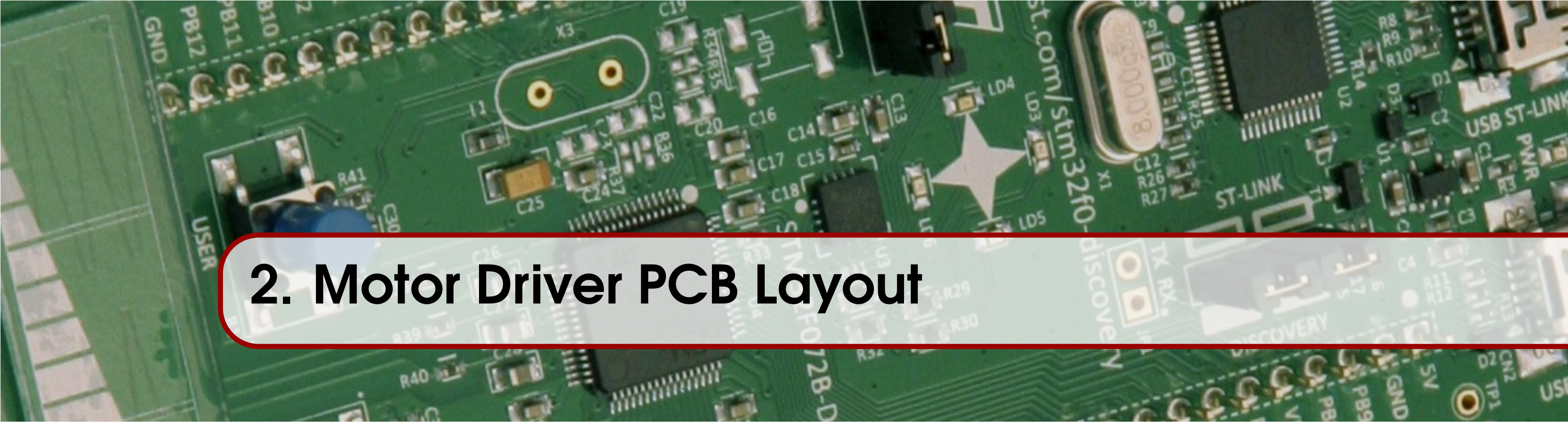
1.4 Deliverables

Please include the ".sch" schematic file as well as a high quality PDF of the schematic. The files are to be submitted through CANVAS. Please refer to Table 1.2 for the grading rubric.

All files you submit to canvas should follow the convention **Section_AssignmentName_FileName**.

Table 1.2: Homework 01 Grading Rubric

Description	Points
All components in BOM implemented	40
All connections made correctly	30
Design is not overly messy	15
Schematic border	5
Author and description in frame	5
PDF of schematic included	5



2. Motor Driver PCB Layout

2.1 Introduction

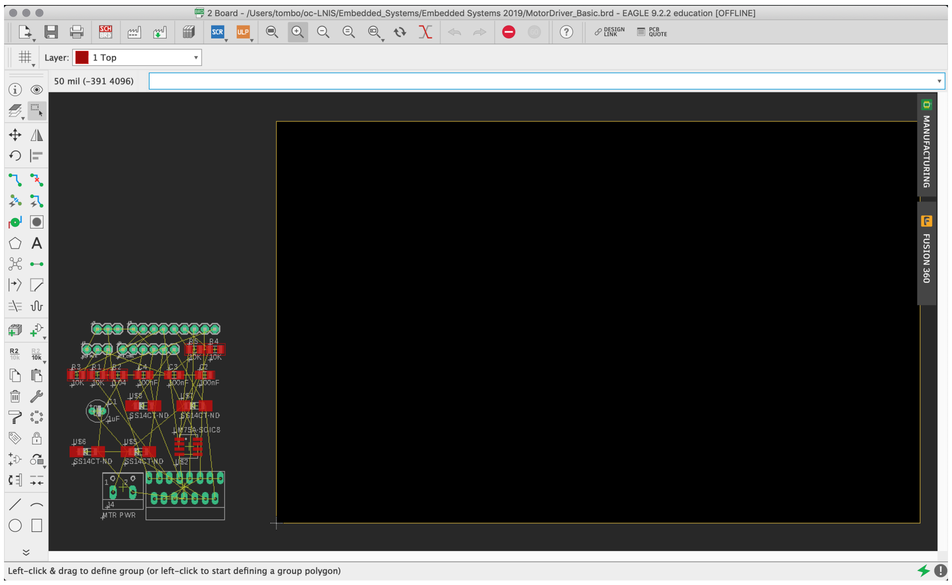
Congratulations! You successfully created your first schematic using EAGLE and you are now ready to move to the next step – the layout of the board! For this purpose, we'll mainly use the PCB design tool included in EAGLE and called: Generate/Switch to board. This is usually done in 3 steps: (1) Open the schematic and switch from the schematic editor to the related board; (2) the placement of the components – where the different devices are physically organized on the PCB surface to minimize the wire routing complexity (you'll need to use your brain); and (3) the wire routing itself. **Try to make the smallest possible PCB (smaller = cheaper). A good goal to shoot for is a 1x2 inch outline.**

2.2 Generate the Board

The schematic is now finished. We can now create PCB (printed circuit board) layout. For that, simply click the Generate/Switch to board command (on the top toolbar, or under the File menu) – which should prompt a new, **board** editor window to open.

2.3 Placing Components

After the board editor window is opened, it will look like the following. All your components will be in a clump over to the left of the origin.

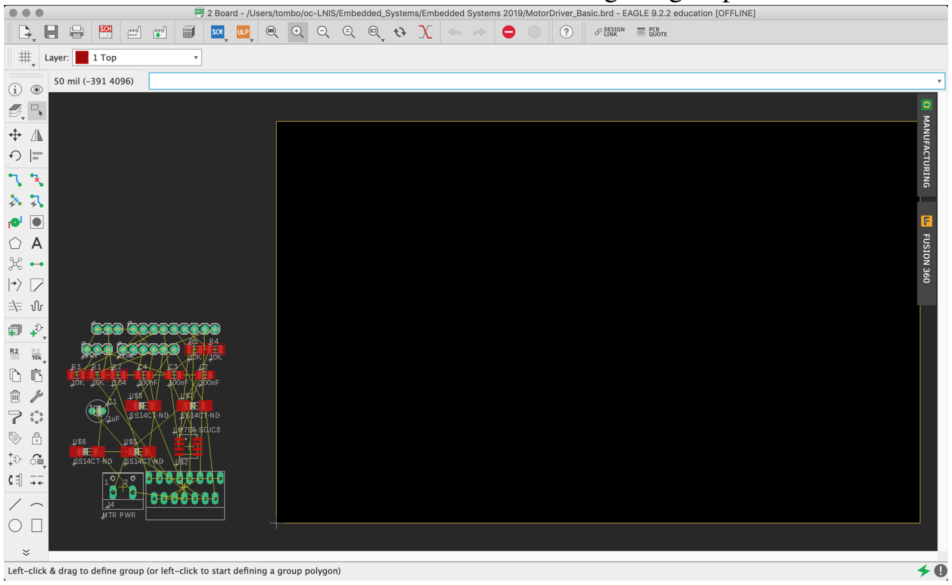

2 Board - /Users/tombo/oc-LNIS/Embedded_Systems/Embedded Systems 2019/MotorDriver_Basic.brd - EAGLE 9.2.2 education [OFFLINE]

Layer: 1 Top

50 mil (-391 4096)

Left-click & drag to define group (or left-click to start defining a group polygon)

The first thing you want to do is move at least some components into the legal board area where you can work with them. If you have a particularly large board with many components, you might want to do this a section at a time. You can move them all at once, using the group-move feature.


2 Board - /Users/tombo/oc-LNIS/Embedded_Systems/Embedded Systems 2019/MotorDriver_Basic.brd - EAGLE 9.2.2 education [OFFLINE]

Layer: 1 Top

50 mil (-391 4096)

Left-click & drag to define group (or left-click to start defining a group polygon)

Select the GROUP icon, then click and drag to make a rectangle that goes all the way around the components. Then select the MOVE icon and RIGHT click (right clicking selects the group instead of a single component) and drag the set into the board outline. Use the ZOOM button to tighter the view.

The full legal side of the board is bigger than you need. Shrink the outline by using the MOVE tool. Click on the center of the top horizontal line (which selects the whole line instead of an endpoint) and move it down, then click on the center of the rightmost vertical line and move it leftward.

Now you need to move the components to (near) where you want them on the final board. OR you want to move them to sensible places that will make the placement of traces easier. A lot of the "ART" of making PCBs (and especially Single Sided Boards) lies in finding "good" places for the components. In general, you can start by placing the components similar to how they appear on the schematic. Another common practice is to put the pin headers on the edge of the board.

After placing all components in the board, click the Design Rule Check icon for checking DRC errors.

2.4 Routing

Routing is probably the hardest part of designing a PCB, but if the placement has been done in an efficient way, the routing shouldn't be too complex. Usually, one routes different nets according to their importance like this:

1. Analog
2. High-speed digital and differential signals
3. Digital signals
4. Power/Ground

Do not use the autorouter! The autorouter doesn't know the Importance of your signals by default. You should first route critical traces by hand and then set the autorouter to do the boring bits. Only if necessary, use the autorouter to do the digital signals. Though, this is highly discouraged, as it often times messes up your design!

PCB Manufacturers usually have specific limits on how small of a trace or drill hole you can make. Our manufacturer has a minimum trace width of 8 mil (8/1000 inches) and has a minimum spacing requirement of 8 mil (i.e. 8 mil trace and space). Also try not to use a drill smaller then 15 mil with an annular ring of 30 mil diameter.

2.5 Artwork

Artwork: Once you finished routing your board, it is time to add some artwork. Use the "Draw→Text" tool to add text onto the "Top Overlay" layer. The most common things one adds are: Name of the board; Version and date; Your name and maybe company logo. Try adding some artwork to your board. Signatures and logos can be imported from bit map pictures. This won't be covered here, but there are many tutorials online that show you how to do it.

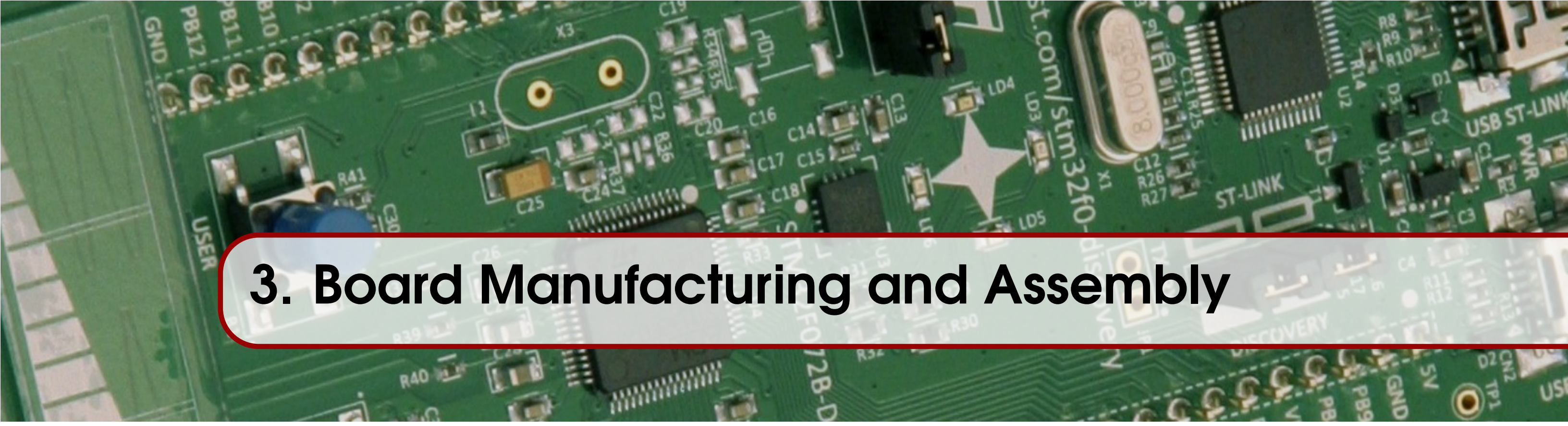
Board Outline: Your board can have any outline. However, we will keep to a rectangular shape. If you don't provide a board outline to the manufacturer, then they will just assume you want a rectangular board around all your components. However, to make sure that they know what the boarder is, it is better to provide one. Therefore, draw a rectangle around your board on Mechanical Layer 3. Use an 8mil trace to do so. We will export this later.

2.6 Deliverables

Create a PDF document as you did for homework 1.1 with your board layout and submit it on Canvas along with a complete zipped version of your EAGLE project. Please use ZIP, not RAR, TAR, *etc*. Submit the board PDF and the zipped EAGLE project folder. Table 2.1 shows the rubric we will be using to grade your submission.

Table 2.1: Homework 02 Grading Rubric

Description	Points
All connections made correctly	20
Correct land pattern dimensions	20
Design is not overly messy	15
All components in BOM implemented	10
Appropriately labelled components	10
Ground plane	10
Name printed on PCB	5
Area under 2 in^2	5
PDF of layout included	5



3. Board Manufacturing and Assembly

3.1 Manufacturing

It’s time to fab your board! As you already know, we will use a PCB manufacturer called OSHPark (https://oshpark.com). They are reasonably cheap with 2 layer boards for $5 per square inch, provide 3 copies of your board included in that price, and ship in under 12 calendar days from ordering. This is why we need to place the order ASAP to make sure we get the boards for the final labs. You already have all the necessary parts from Homework 2.

Once your board has been approved by the grader, all what you have to do is go through the submission page: https://oshpark.com and submit the fabrication files according to their needs: https://oshpark.com/guidelines. Make sure to clean all the possible errors the system will detect. Once everything is ready to go, please checkout and pay the fees. Note that to reduce the fabrication costs, you are allowed to team up and send a unique board per group of three. If you prefer to minimize risks (or in case your board has many flaws identified by the TAs), please feel free to use any of the solutions available on Canvas.

3.2 Assembly

Once you’ve received your PCBs from OSHPark, it’s time for the fun, hands-on part–soldering the components onto the board! All of the parts listed in the BOM are available as a kit that you can purchase in the stockroom. Most people haven’t soldered before, so we like to do a demonstration during lab time before you dive in. You can also purchase soldering practice kits for about $1 from the stockroom if you want to get a feel for it before you solder your PCB. In addition to the in-class demo, we have some references and tutorials on Canvas that you can look at before you begin.

3.3 Testing and Verification

Even if your PCB was designed well, it’s possible for something to go wrong while soldering that keeps your device from working. For example, applying too much heat to a component with the soldering iron or heat gun can damage the component, or you may accidently short a connection with solder. It is therefore very important to probe your finished board before you power it up. Get a multimeter and put it in continuity mode (it beeps when the leads are shorted), and start poking around the board. For example, put the leads across the 3.3V, 5V, 6V, and Ground rails to make sure you haven’t shorted any power supplies. Check that the I^2C data and clock lines aren’t connected, the motor control pins aren’t connected, *etc.* It is very important that you catch any problems now so that you don’t damage the board.

3.4 Deliverables

For this deliverable you'll need to check off your board with a TA. Show them that you can make the motor spin both clockwise and counter-clockwise. There is no need to demonstrate speed control, as you will be doing that in the lab.