miniEngine v2.0

Documentation

Software version 2.0 alpha 5
Hardware revision D

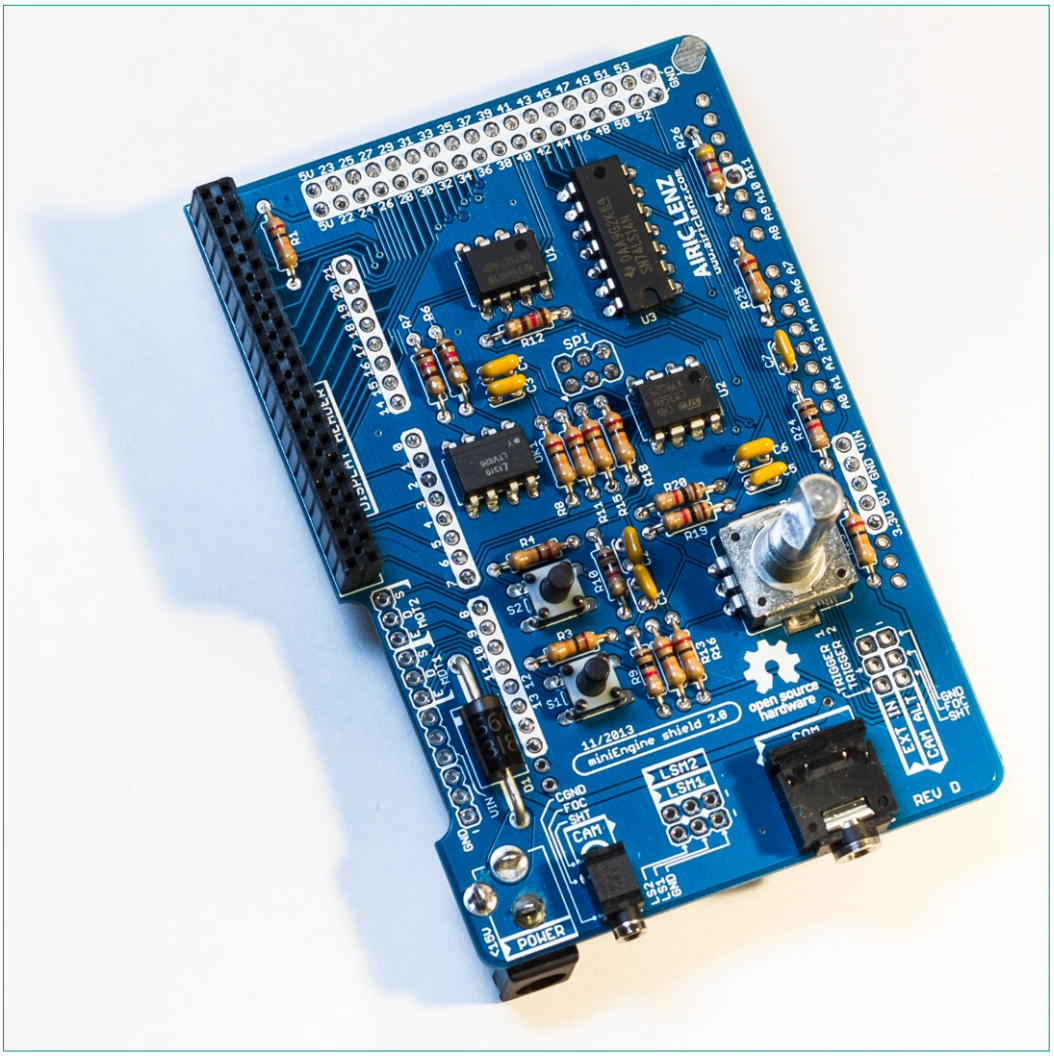
Airic Lenz

Content

1 Introduction

The miniEngine is open-source motion-control-system for time-lapse as well as video photography. The current version provides all functions needed to smoothly and precisely control 2 stepper-motors and one camera. It is Arduino DUE based and comes with a shield, designed to hold all required hardware components. These components allow user input as well as connecting 2 stepper-motor-drivers for then connecting the motors.

This is how a fully assembled miniE (short for miniEngine) shield for the Arduino DUE looks like (the display is not plugged into the shield for a better view):



The miniEngine 2 Arduino shield

These function are currently supported by the system:

- Simultaneous control of two stepper motors
- Control of one photo-camera (needs to be remote trigger-able)
- Time-lapse recording in shoot-move-shoot mode
- Time-lapse recording in continuous mode (also usable for video shoots)
- System is fully adjustable to the hardware used, e.g. stepper motor calibration to be able to set up precise moves in cm / °
- User input via one rotary encoder and two buttons
- 320 x 240 pixel colour TFT display
- All settings / program data stored on an SD card for easy and fast reconfiguration using multiple cards holding different set-ups
- Fully open-source

These are the major components that are required for the system:

1	Arduino DUE with the miniEngine 2 software installed
1	ITDB02-2.4E Display (320x240 Pixel, SD-Card, Touch) from ITEAD Studio
1	miniEngine 2 Arduino shield (can be bought on www.airiclenz.com)
1 or 2	Stepper Motor Drivers (BigEasyDriver is recommended and supported)
1 or 2	Stepper Motors
1	7-15V Power supply capable of supplying up to 4A

If you have any questions regarding this documentation, the described processed or the system itself, please get help in the forum before you possibly destroy your system:

www.forum.airiclenz.com

| 2 | Hardware

The following chapter explains how to assemble the miniEngine shield and which components are needed in detail.

There is an additional power input on the miniEngine shield. This input is secured up to 5A by a beefy diode and can easily handle the amount of energy the stepper motors might need. USE THIS POWER-JACK and NOT the original power-jack of the Arduino!!! Otherwise you might destroy the Arduino!

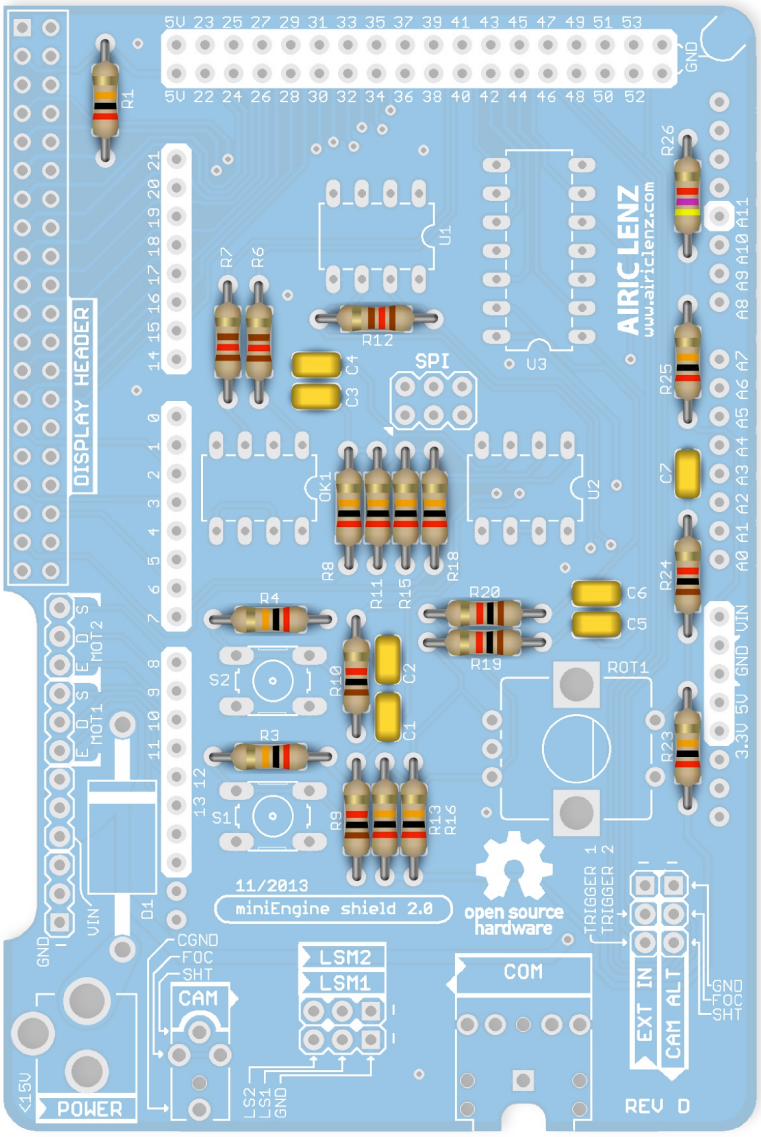
This guide requires soldering. If you have no or little experience with soldering, please ask someone to help you with the assembly. This will reduce the risk for hurting yourself as well as improve your soldering quality and make sure everything works as intended.

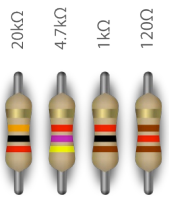
| 2.1 | *Bill of Material - miniEngine 2 shield*

Amount and Part	Board designator(s)	Description / package
11x 20kΩ Resistor	R1, R3, R4, R8, R11,R13, R15, R16, R18, R23, R25	1/4W, 5%, THT Resistor
5x 1kΩ Resistor	R9, R10, R19, R20, R24	1/4W, 5%, THT Resistor
3x 120Ω Resistor	R6, R7, R12	1/4W, 5%, THT Resistor
1x 4.7kΩ Resistor	R26	1/4W, 5%, THT Resistor
7x 100nF Capacitor	C1, C2, C3, C4, C6, C6, C7	2.54 mm pitch Capacitor
1x 5A Diode	D1	DO201AD
1x 2.5MM Stereo Jack	CAM	2.5mm Headphone jack
1x 3.5MM Stereo Jack	COM	3.5mm Headphone jack
1x 2-Channel Optocoupler	OK1	LTV826, DIP-8
1x Differential Bus Transceiver	U1	SN75176B, DIP-8
1x 2-Channel Op-Amp	U2	LM358, DIP-8

1x Hex Inverting Schmitt Trigger	U3	SN74HC14N, DIP-14
2x Tactile Switch	S1, S2	Tactile Button, THT
2x 1x18 Male Pin Header	Arduino Pins (marked white)	Male Pin Header with 2.54mm pitch, THT
4x 1x8 Male Pin Header	Arduino Pins (marked white)	Male Pin Header with 2.54mm pitch, THT
1x 2x20 Female Pin Header	DISPLAY HEADER	Female Pin Header with 2.54mm pitch, THT
1x 2x3 Female Pin Header	SPI	Female Pin Header with 2.54mm pitch, THT
1x Rotary encoder	ROT1	PEC11, 25mm shaft, 24 steps, 24 indent, switch
2x Limit switch	LSM1, LSM2	Generic 2x3, THT, 2.54mm pitch header
1x Trigger inputs & alternative Camera output	EXT IN, CAM ALT	Generic 2x3, THT, 2.54mm pitch header

| 2.2 | *Assembly Steps - miniEngine 2 shield*





Step 1

Solder all Resitors and all Capacitors to the board:

20kΩ: R1, R3, R4, R8, R11,R13, R15, R16, R18, R23, R25

1kΩ: R9, R10, R19, R20, R24

120Ω: R6, R7, R12

4.7kΩ: R26

100nF: C1, C2, C3, C4, C5, C6 ,C7

Step 2

Solder all medium sized top side components:

U1: SN75176B (Communication)

U2: LM358 (Backlight amplifier)

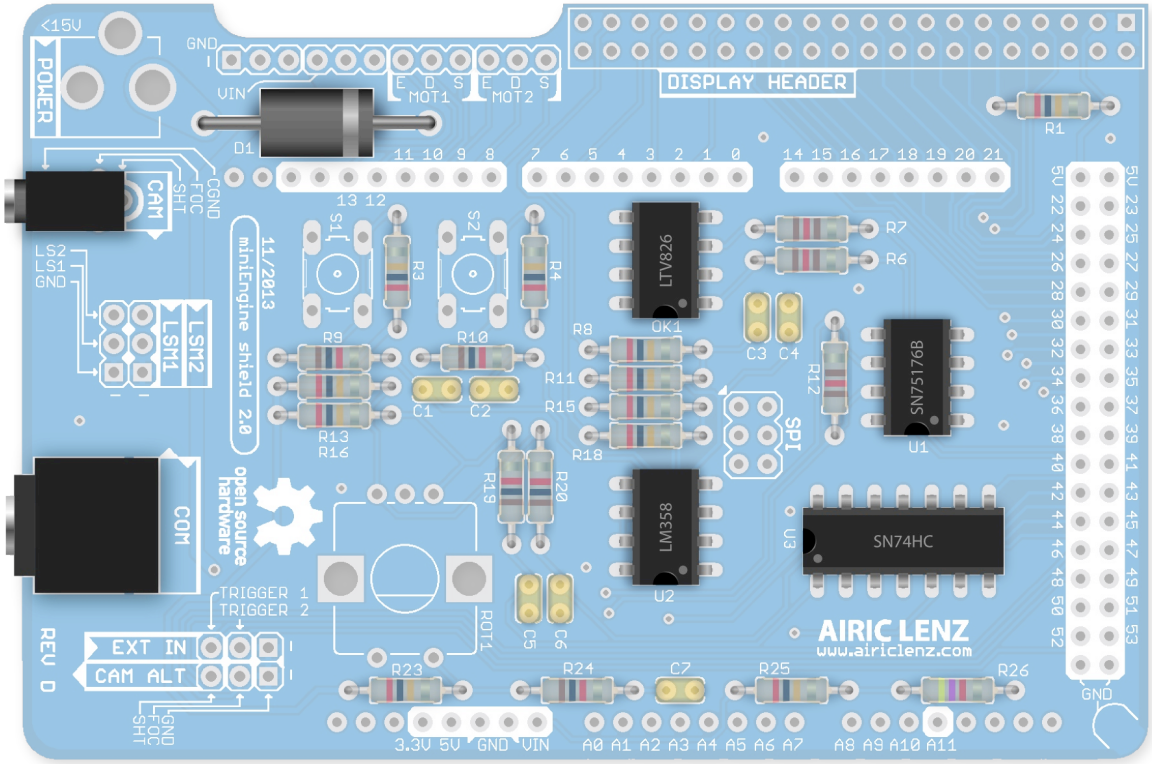
U3: SN74HC (Invert. Schmitt Trigger)

OK1: LTV826 (Optoisolator)

D1: 5A Rectifier Diode

CAM: 2.5 mm Stereo Jack

COM: 3.5 mm Stereo Jack



Step 3

Solder all BOTTOM side components:
MUST BE SOLDERED TO THE BOTTOM SIDE!

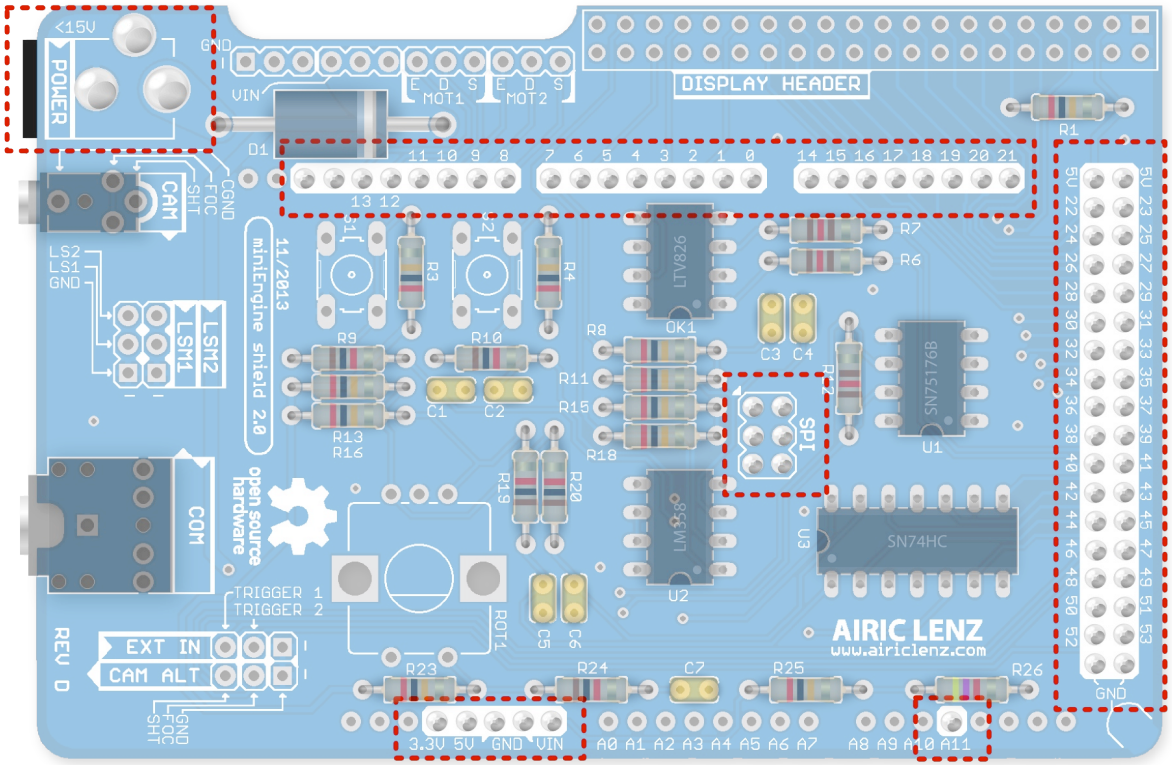
POWER: 2.1mm Power Jack (inner pin positive)

Arduino Pins: Male Pin Headers need to be soldered to the white marked holes! You can solder umarked pins too but white marked ones must be soldered.

4 headers with 8pins
(one needs to be broken for Power Pins and Analog input pin 11)

2 headers with 18pins

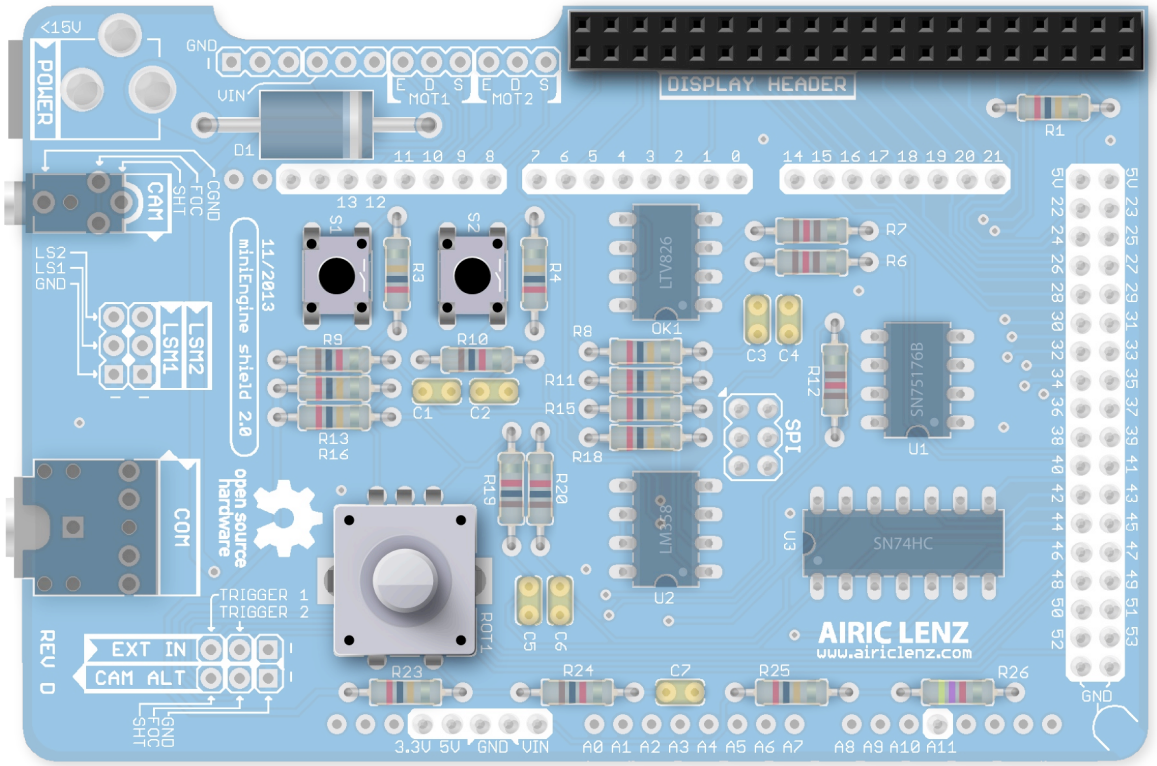
SPI Port: 3x2 Female Header



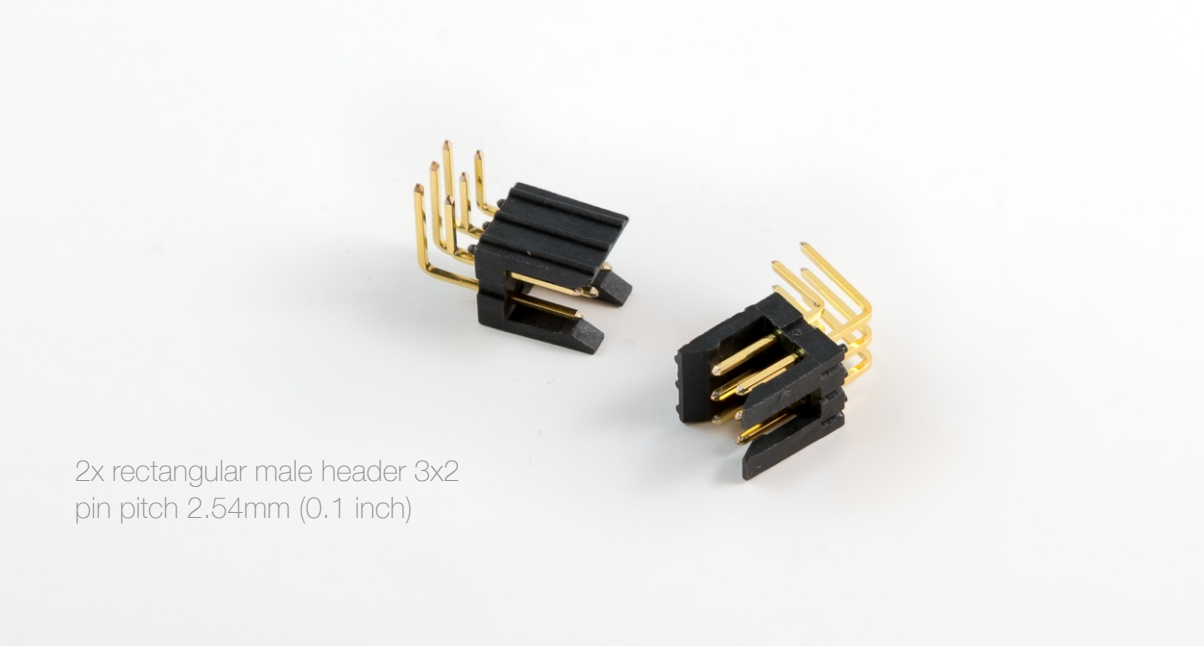
Step 4

Solder the big top side components:

S1, S2:	Tactike Switches
DISPLAY HEADER:	20x2 Female Header
ROT1:	Rotary Encoder

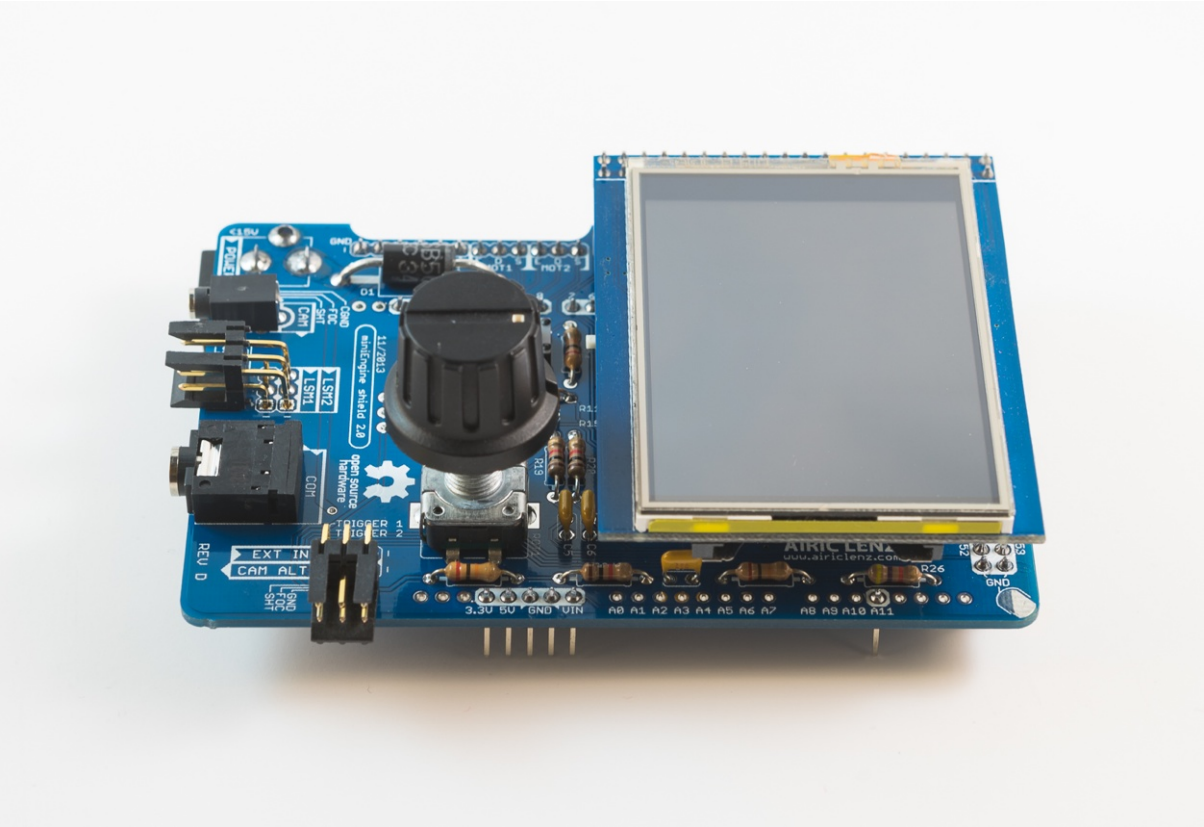


For connecting the limit switches as well ass the external trigger inputs to the system I recommend this type of connector (altho you can use whatever you like here as the footprint on the board uses a standard pitch). You need 2 of them - one for the Limit Switches and one for the Triggers:



2x rectangular male header 3x2
pin pitch 2.54mm (0.1 inch)

Please set your display to the 16 bit mode before plugging it into the shield.



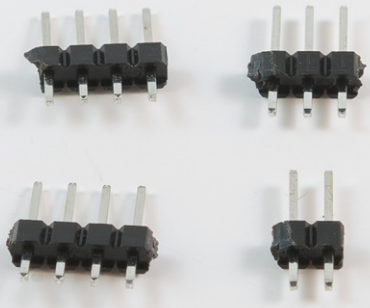
| 2.3 | *Bill of Material - BED Board*

Amount and Part	Description / package
5x M3 Screw 12mm	Use screws with a flat head as you most probably need to file off a bit of the head of two of the screws
1x M3 Screw 8mm	-
5x M3 Standoff Nut 5mm	You can use 2 regular M3 nuts for one Standoff nut
6x M3 Standoff (about 10mm)	The length is determined by your needs and heavily dependant on your enclosure
1x Male Pin Header 13x1 Pitch 2.54mm (0.1 inch)	13 pins needed per Big Easy Driver
1x or 2x Angled Male Headers 4x1 Pitch 2.54mm (0.1 inch)	one for each BED
1x or 2x Big Easy Driver	also called BED
4x Terminal Screw Headers 6x1 Pitch 2.54mm (0.1 inch)	*This is optional. You can also solder wires directly between the miniEngine shield and the BED board.*
12x Wire (~40mm)	*This is optional. You can also solder wires directly between the miniEngine shield and the BED board.*
1x or 2x Modular Plug 4P4C	*This is optional (one for each BED).*
1x or 2x DIP Switch 3x (DIP-6)	*This is optional (one for each BED).*

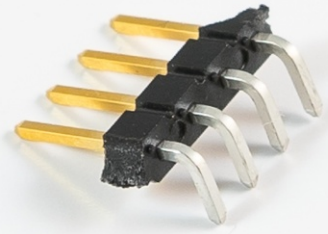
1X M3 Screw 8mm
5x M3 Screw 12mm
5x M3 Standoff nut 5mm
6x M3 Standoff

13x1 Male Pin Header
broken into:

2x 4 Pins
1x 3 Pins
1x 2 Pins

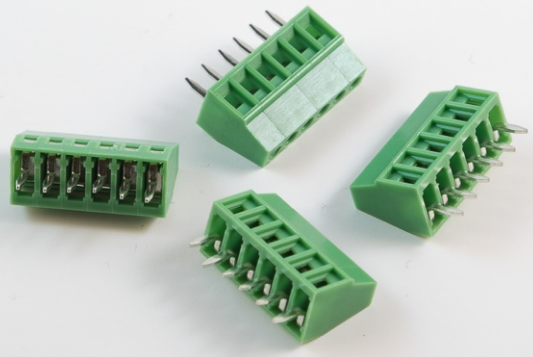


1x or 2x Angled Male Header 4x1

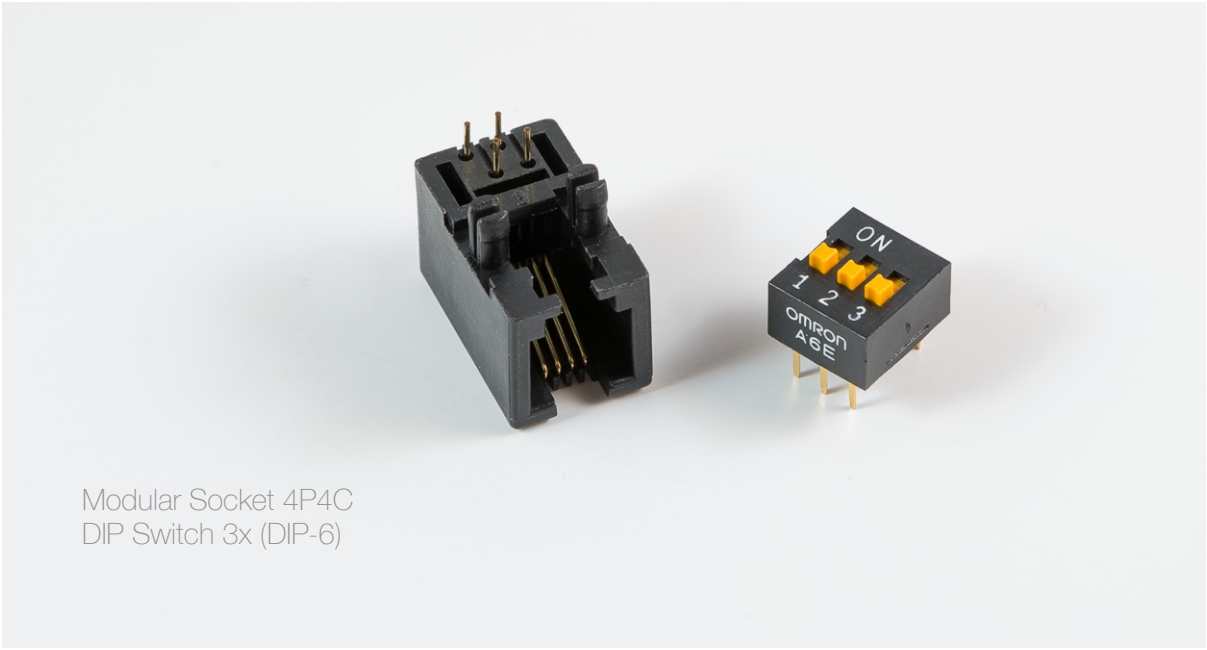


These are optional items:

4x Terminal Screw Headers 6x1



12x Wires (~40mm)

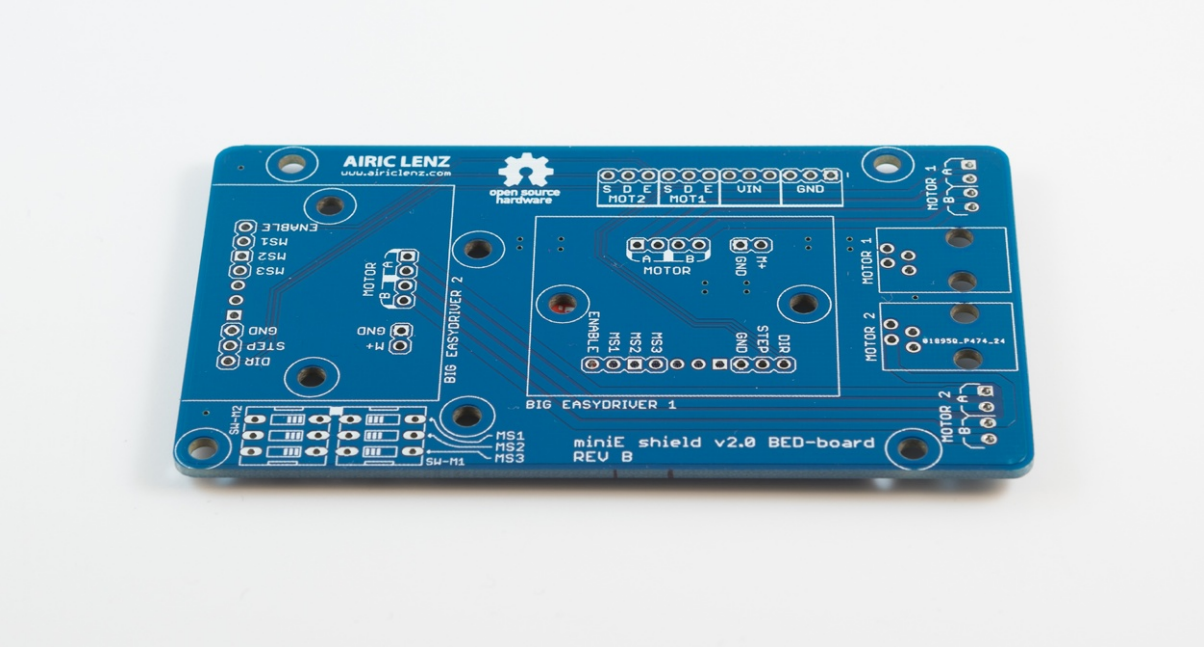


Modular Socket 4P4C
DIP Switch 3x (DIP-6)

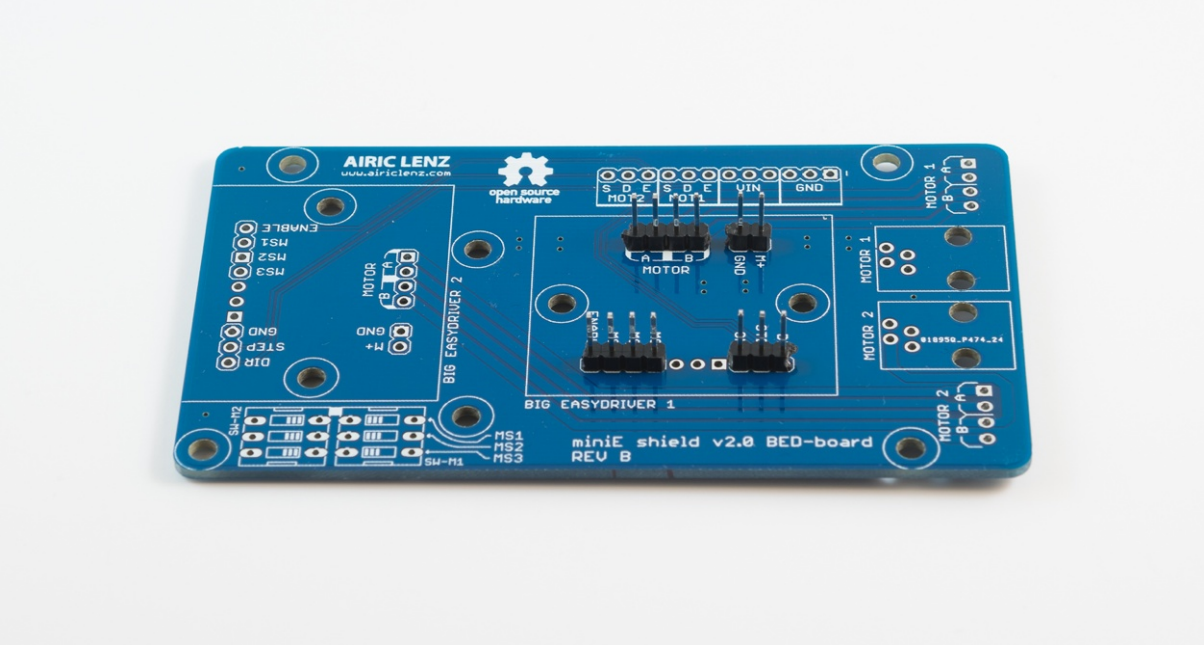
| 2.4 | *Assembly Steps - BED Board*

The following guide describes a mounting solution that solders the Big EasyDriver board(s) permanently to the BED board because this is the most space saving solution. If you with to use a detachable solution, please don‘t follow this guide.

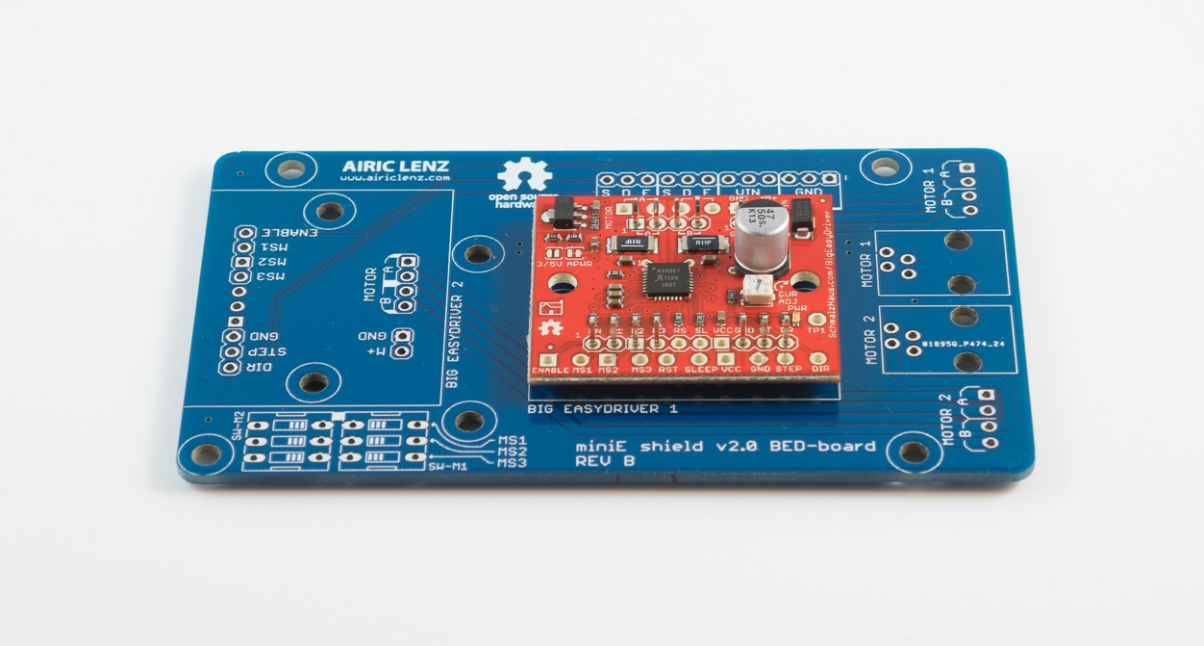
First take your BED board PCB...



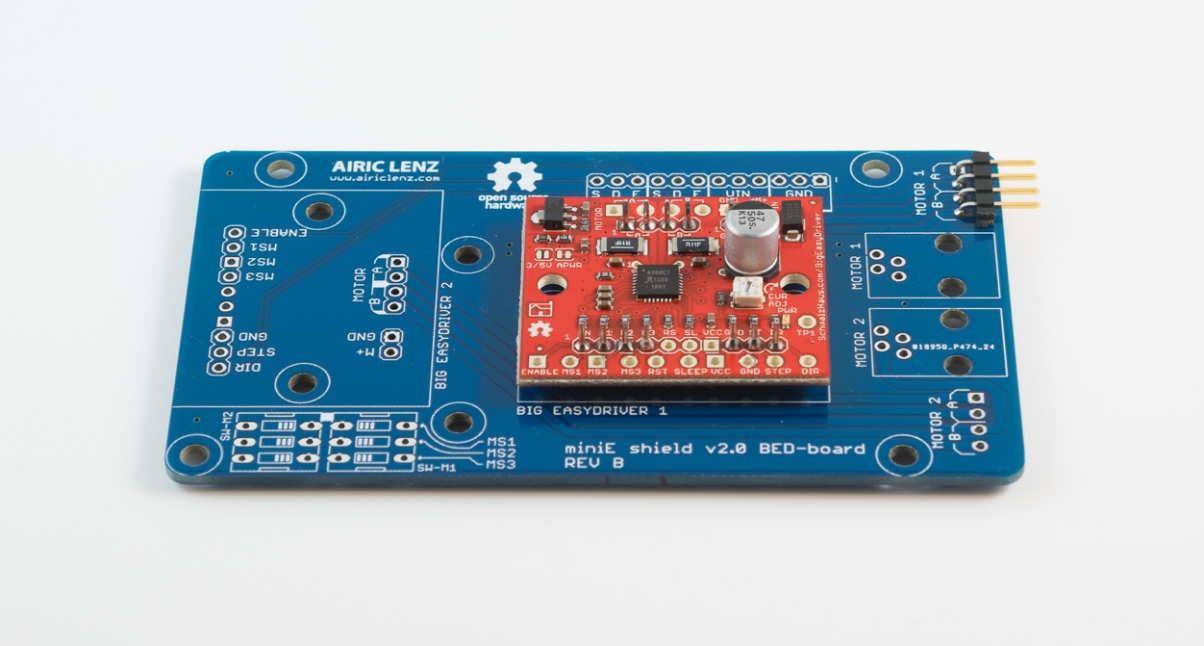
...as well as your Pins and solder them with the short end to the BED board like this (soldered from the bottom side). Make sure they are soldered straight! If you want to use 2 Big EasyDrivers repeat all the following steps for the other one as well.



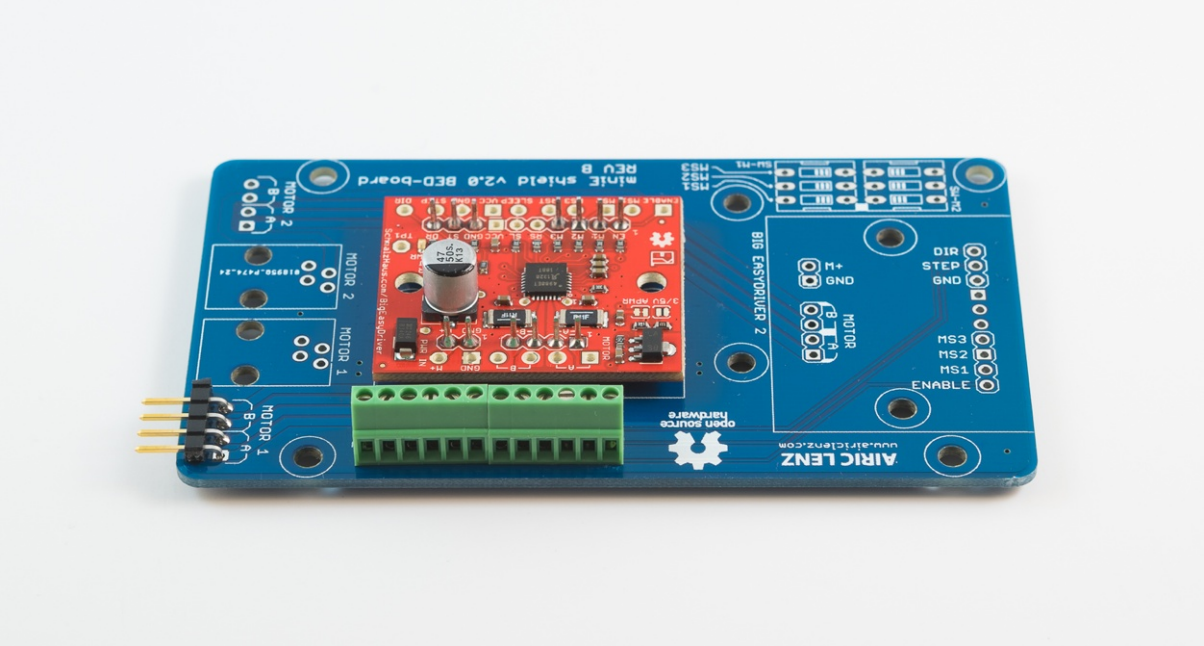
Now place your Big EasyDriver on these pins like this.



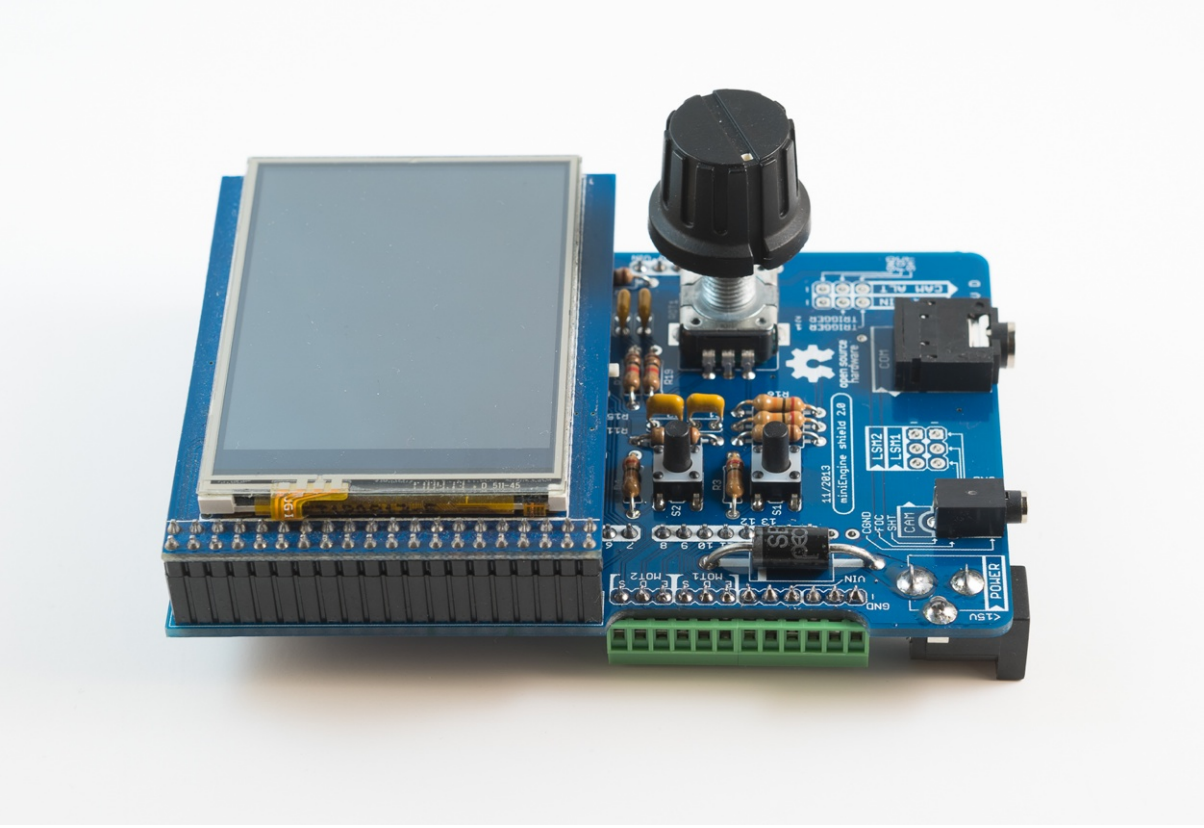
Take your soldering iron and solder the Big EasyDrivers to the pins. Also solder the motor connector to the BED board like this:



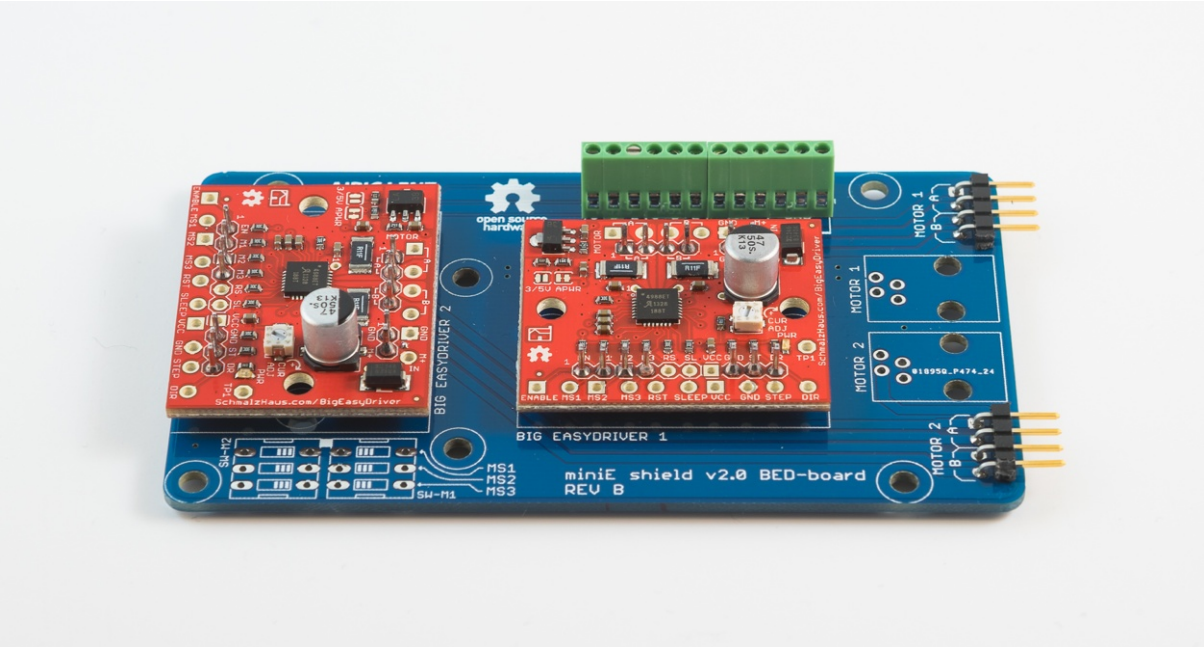
Take 2 of the Screw Terminals and solder them to the BED board:



Solder the other two Screw Terminals to the minEngine 2 shield but to the bottom side:



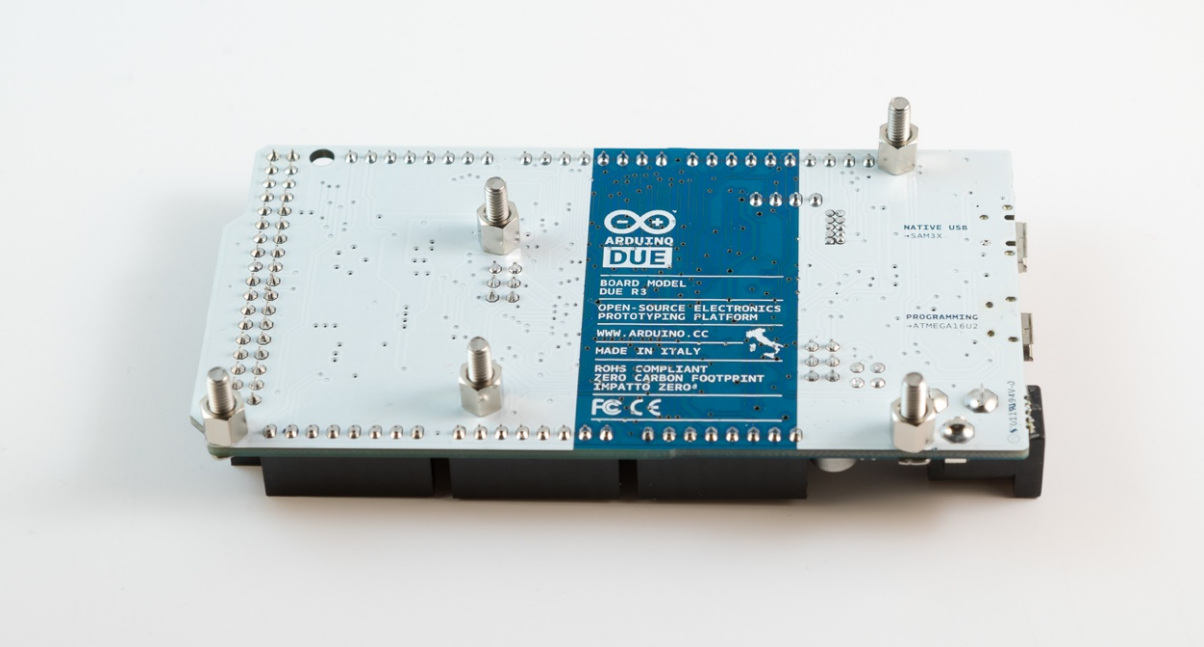
Your assembly of the BED board is done now! Congratulations! Here is how the board looks like, if you assembled it with both possible Big EasyDrivers:



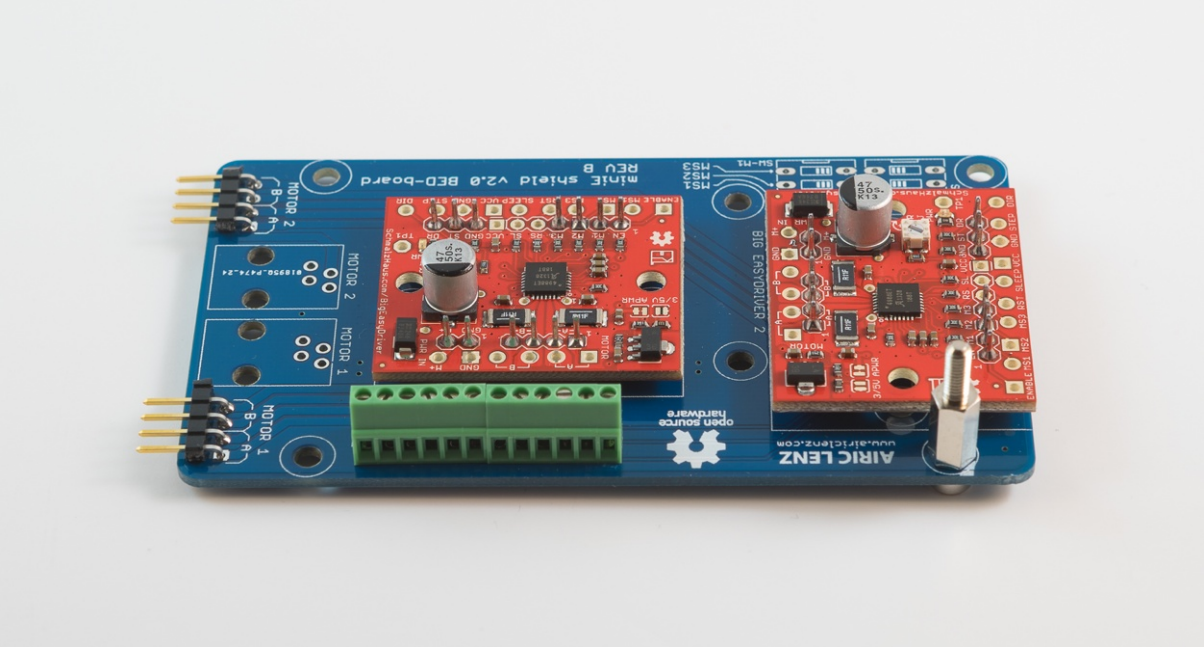
If you want to use the DIP Switches or the optional motor connectors, please solder them to the board now.

| 2.5 | *Final assembly (putting it all together)*

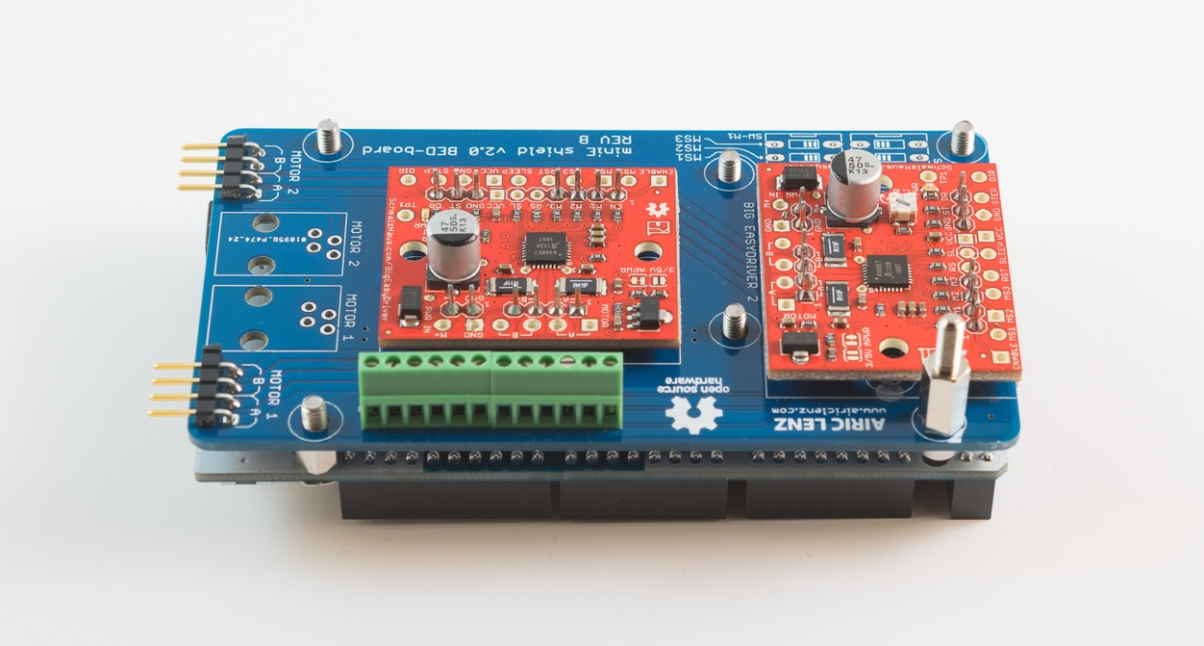
Take the 5 M3 Screws that are 12mm long as well as 5 of the 5mm Standoff nuts and attach them to your Arduino DUE as displayed here. Make sure that the screws do not make contact to any of the metal parts of the Arduino DUE. I needed to file off a bit of two of the screws to make sure that this does not happen!



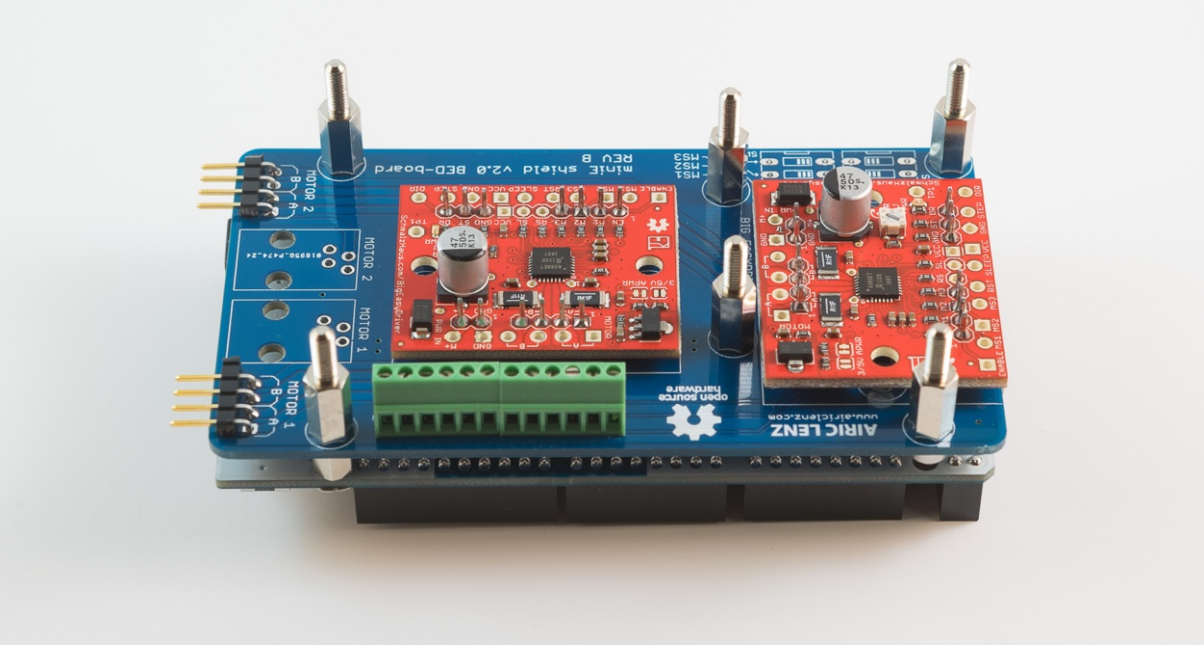
Screw the 8mm M3 screw with the last Standoff to the BED board:



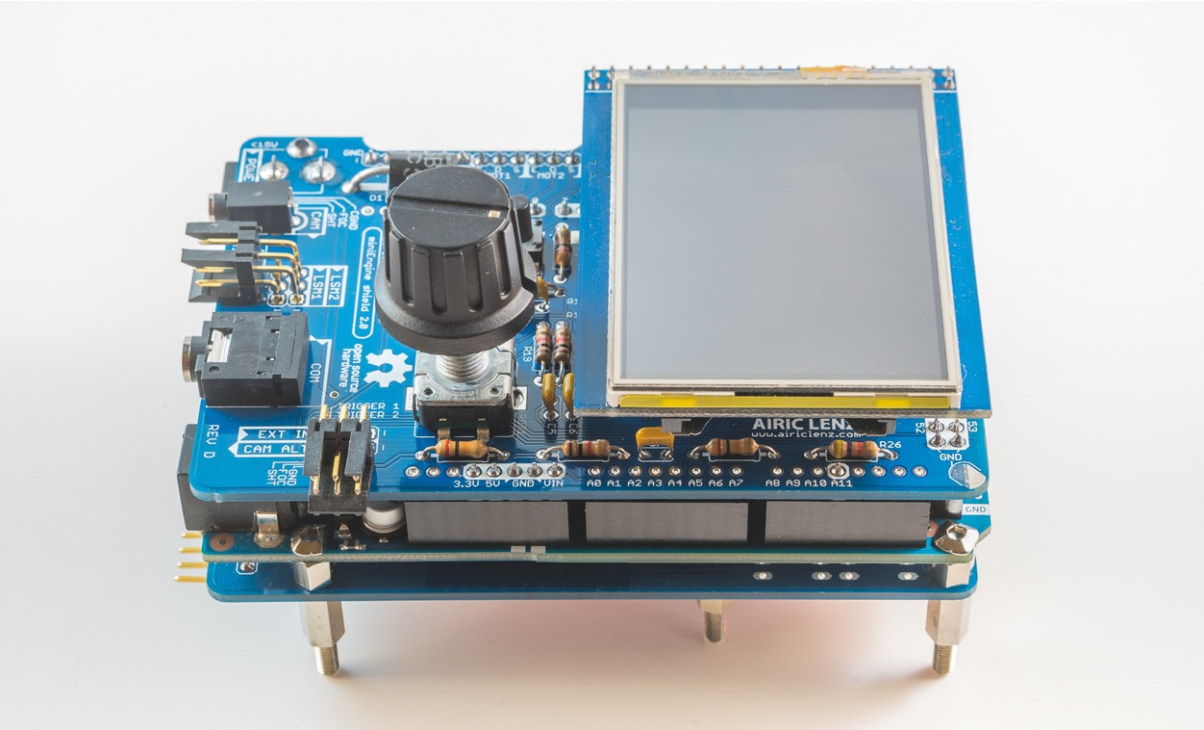
Place the BED board on top of the Arduino DUE like so:



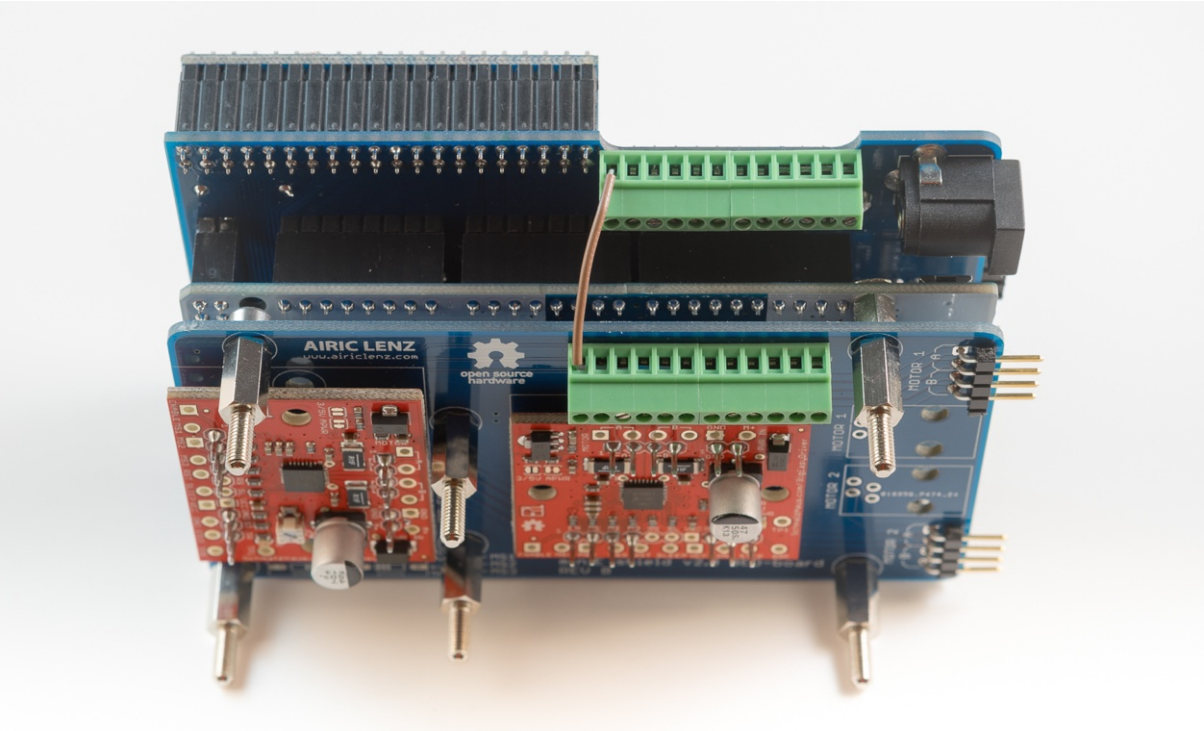
Then attach all the remaining Standoffs:



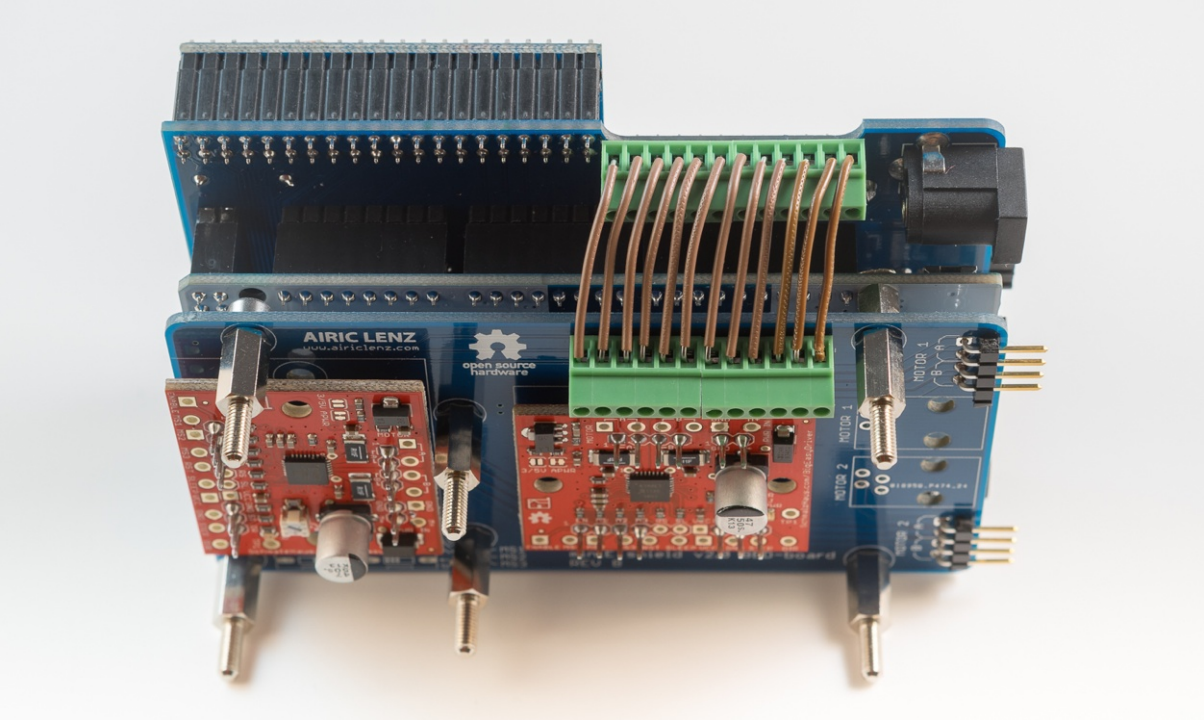
Plug your miniEngine 2 shield carefully into the Arduino DUE:



Now screw the wires into the screw terminal to connect the two boards. Begin on the left hand side:

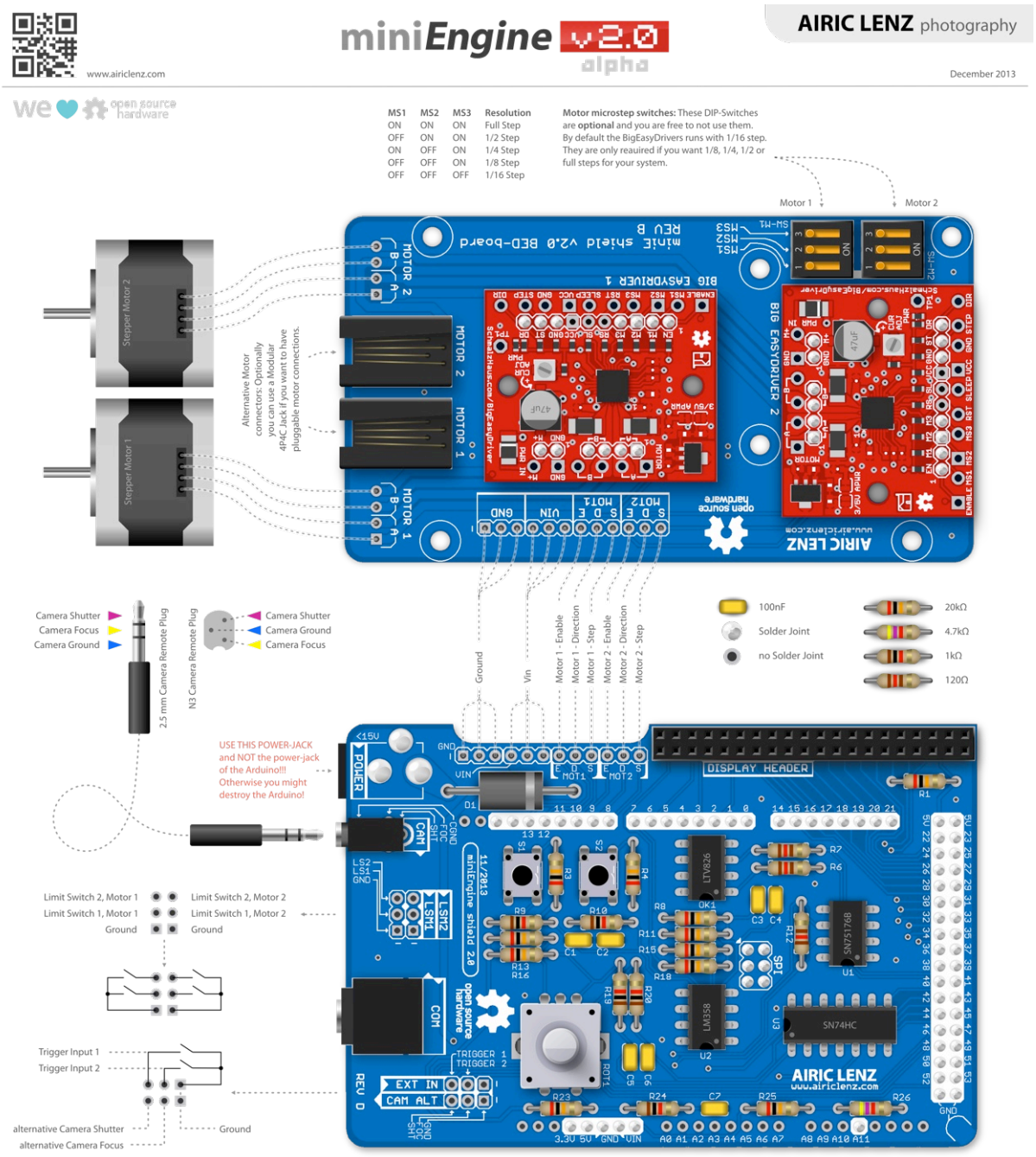


Here is the assembly with all wires connected:



Your miniEngine 2 is now ready for some awesome time-lapse and video photography!

| 2.6 | *Shield functions*



| 3 | Software

| 3.1 | *Installation & Update*

The installation process as well as the Update process are identical. For installing and updating the software on your Arduino DUE please follow this guide.

If you don‘t already have installed the Arduino Development Suite (for the Arduino DUE the beta version 1.5.x is needed), please download it from:

http://arduino.cc

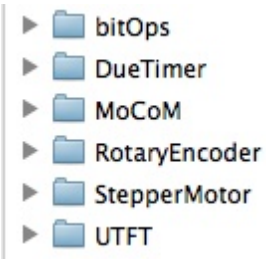
and install it. The latest miniEngine 2 software can be downloaded from here:

http://github.com/airiclenz/miniEngine2

Now, after you downloaded the project, please unzip (if you downloaded a ‚ZIP-archive) or copy it to any location you want. Please open the new folder and then open

*Software \ libraries *

You should then see the following folder content:



2 of the 6 required libraries were developed by me. Two of them were not.

The library ***UTFT*** (here in version 2.0) was developed by Henning Karlsen (*http://henningkarlsen.com/electronics/library.php?id=52*).

The ***DueTimer*** library was developed by Ivan Seidel (*https://github.com/ivanseidel/DueTimer*).

A big „Thank you“ to both of you for making these libraries available!

To successfully compile the miniEngine 2 code, you need to install these 6 libraries within your Arduino development environment. To do so, open the Arduino folder. This folder

should be placed in your user accounts documents folder (You can check the path in the Arduino settings). On a Mac this folder usually can be found here:

/Users/USERNAME/Documents/Arduino

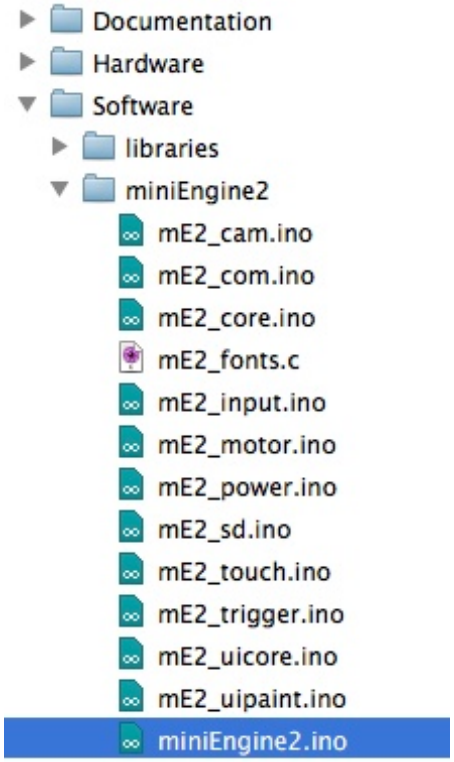
On a Windows system this folder should be placed here:

Drive:\Users\USERNAME\My Documents\Arduino

In this folder you should find a sub folder called *libraries* (if not, create a folder called *libraries* in the Arduino folder). Now copy all 6 library folders from your download folder into this Arduino-library folder. If you are doing an update, simply replace the old folders.

Now re- start the Arduino programming environment. You just installed the needed libraries and should now be able to compile and install the miniEngine software on your Arduino Due. In the project folder you downloaded, go back to the sub-folder *Software\miniEngine2*:

Open the project in your Arduino 1.5.x by double clicking the main project file *miniEngine2.ino* (all needed files will then be loaded automatically).



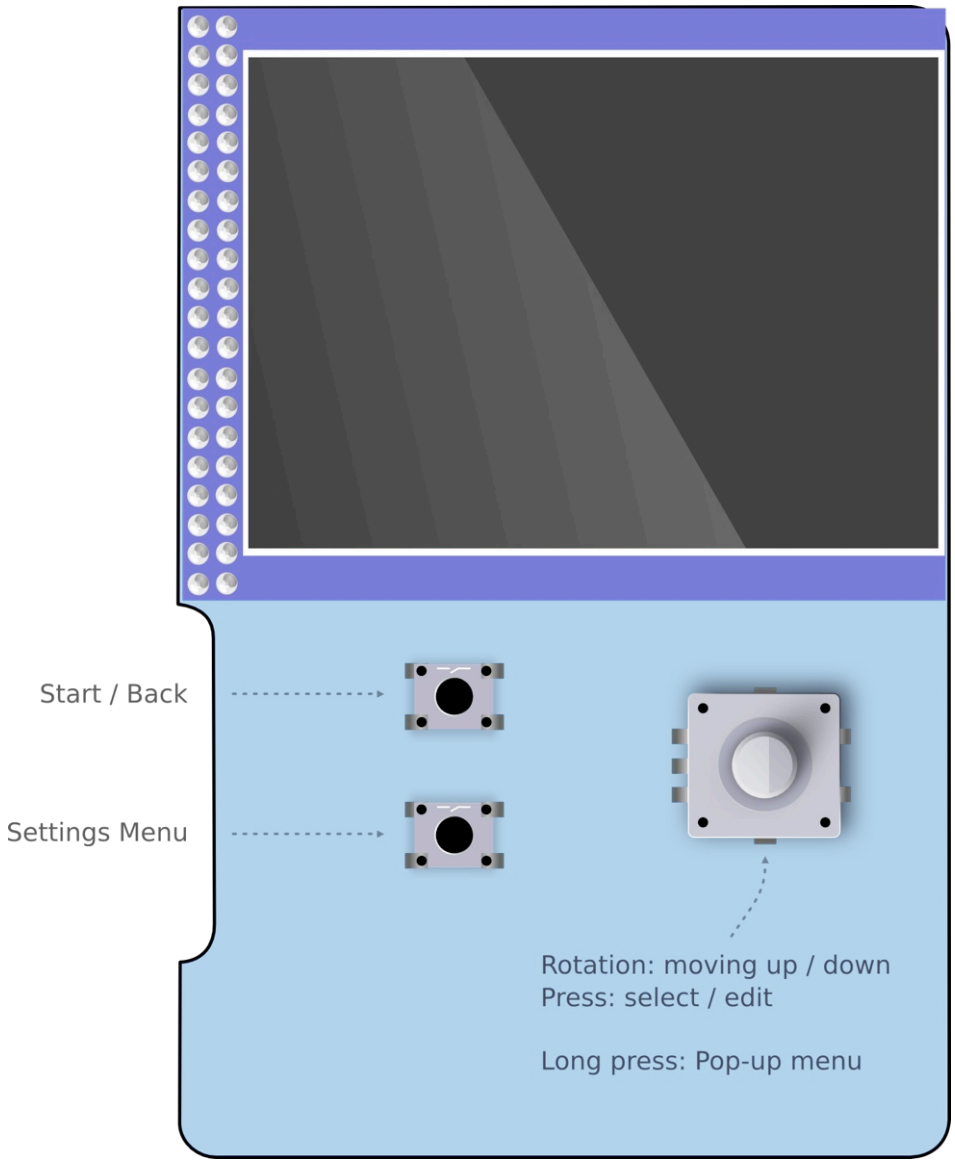
For uploading the software to your Arduino DUE board, connect your Arduino DUE with the „Programming port (the right one) to your computer (for details about connecting the Arduino DUE, please check the Arduino website and the *Arduino documentation*).

Now press the upload button:

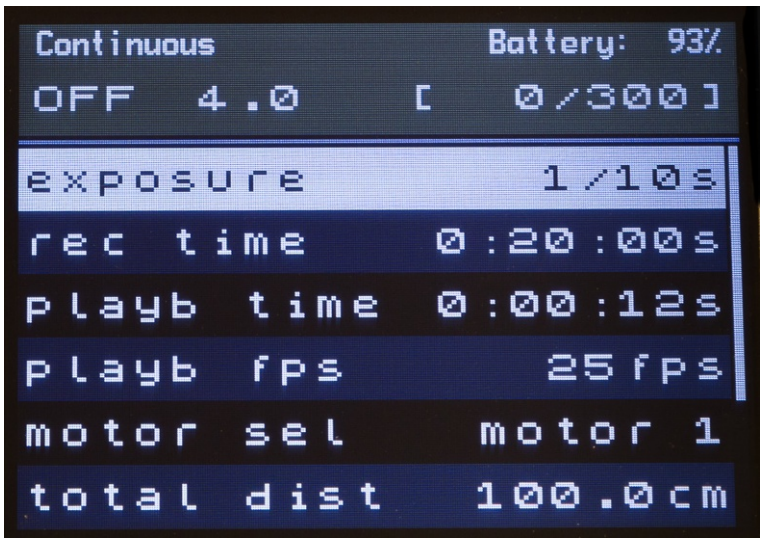
After the code was successfully uploaded, your Arduino board is ready for doing timelapse photography for you.

| 3.2 | *How to use the system*

The graphic below describes the functions of the 3 main input elements of the miniEngine 2 shield.

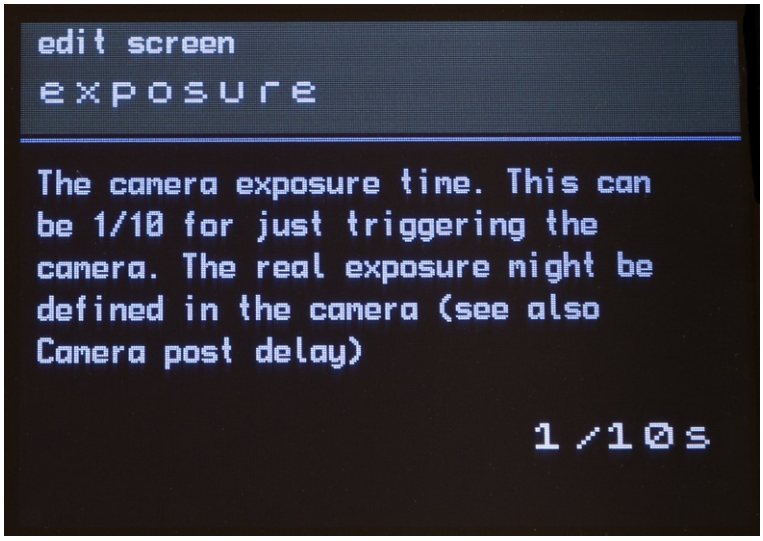


The following pictures show the main screens of the system. You will also find a brief introduction on how to navigate through the system:



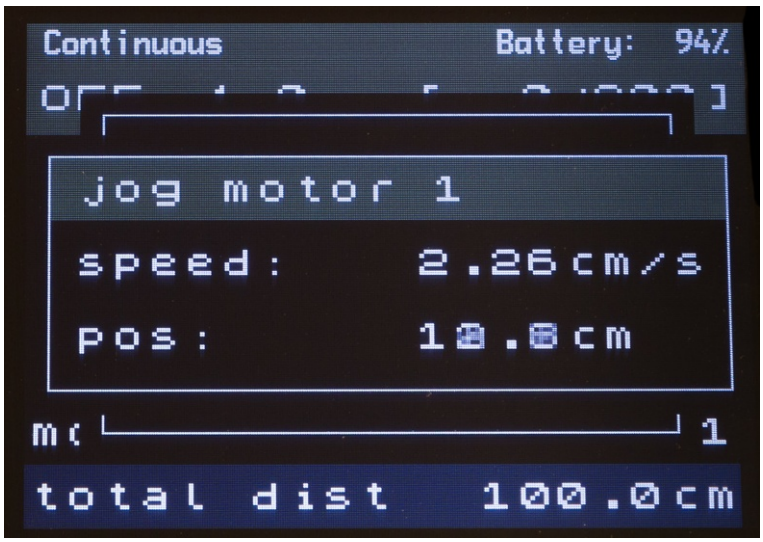
This is the main screen of the miniEngine 2. In the header it shows in which mode the system is at the moment (here: System is not running, „Continuous mode“, The interval is 4 seconds and the programmed run will execute 300 pictures of which zero are done).

Pressing the rotary switch („Select / Edit key“) will lead you to the following screen where you can edit the selected setting (in this case the Camera exposure):



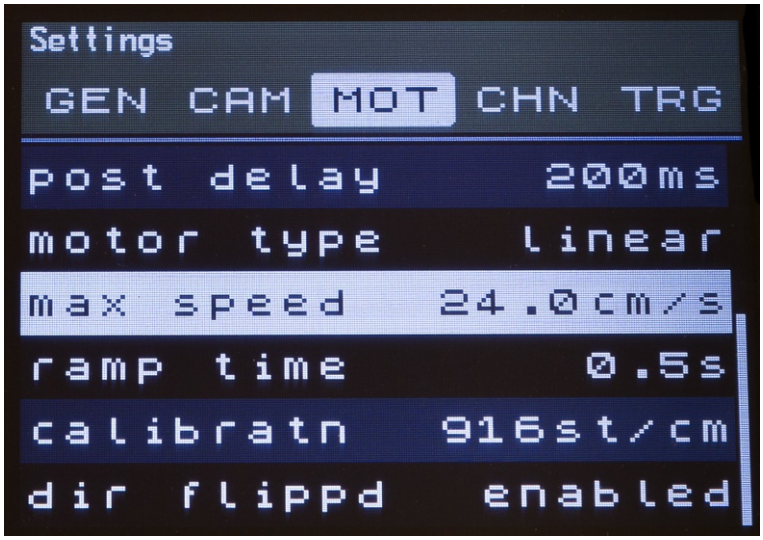
All of the edit screens show a short description of the setting you can change there - in this case the Camera exposure. The setting can be changes with the rotary encoder. Pressing the „ Start / Back key“ will bring you back to the previous screen.

Back on the main screen you can press your rotary encoder down for about half a second to come to the popup menu. From there you can access different often needed functions such as manual motor movement (jog function) by pressing the rotary encoder down („Select / Edit key“):



Pressing the „ Start / Back key“ will bring you back to the main screen.

If you are on the main screen, pressing the „Settings menu key“ will open the Settings screens. There are 5 different settings screens and repeatedly pressing the „Settings menu key“ will browse through the different screens.



Pressing the „Start / Back key“ will bring you back to the main screen.

| 3.3 | *Motor calibration*

The miniEngine 2 offers the possibility to calibrate the motors you are using. This calibration takes the motor and its specifications itself as well as the rig it is driving into account. This chapter is aiming at helping you finding the correct calibration value for your motor(s). The calibration values needed, should be given with the following units:

- steps per centimeter for linear movements
- steps per degree for radial movements

As stepper motors come in a variety of form factors and specifications we need to add the most important of these specs to our calibrations. This is the degrees been moved by one full motor step. The most common one is 1.8°. This means that such a motor needs 200 full steps to do one full revelation:

$$\frac{360°}{1.8°} = 200$$

Because we are using multi-stepping to achieve a finer resolution, we need to multiply this full-step-value with the amount of sub-steps done. In this example we are using 1/16th stepping (which is the default for the recommended Big Easydriver):

$$200 \times 16 = 3200$$

Ok - so we need 3200 steps to do a full revelation with the stepper motor in micro-stepping-mode. Now let‘s add the rig that this motor is driving to the calculation. For now we are assuming we are driving a toothed belt with the stepper. The gearwheel has a circumference of 3.5cm. This means that 1 stepper revelation moves the belt 3.5cm. Here is how we get how much steps are needed for 1cm - which is also our final calibration value:

$$\frac{1cm}{3.5cm} \times 3200 = 914.29$$

So we need to enter 914.29 steps / cm as the calibration value into the system.

If we had a motor that would do a radial move, the calibration-calculation would just differ in the last step. Assuming you are not using the motor to rotate your axis 1:1 but with a reduction of 1:4 (four revelations of the motor rotate the axis one time) , the final calculation-step would look like this:

$$4 \times \frac{3200}{360} = 35.56$$

So the final calibration value for the radial setup is 35.56 steps / degree.

Software License

The miniEngine software is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

The miniEngine software and all belonging material is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see *http://www.gnu.org/licenses/*.

Hardware License

All hardware designs and schematics provides along with the miniE software are free. You can redistribute it and/or modify it under the terms of the CERN Open Hardware Licence 1.1, or (at your option) any later version.

The hardware documentation and hardware files are distributed in the hope that they will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the CERN Open Hardware Licence for more details.

You should have received a copy of the CERN Open Hardware Licence along with these files. If not, see *http://www.ohwr.org/projects/cernohl/wiki/*.