Arduino Based THC

For the Everlast Power Plasma 50

User Manual

*Open source hardware and software free to non-commercial use.*

[www.github.com](http://www.github.com), project: regeg/ArduinoTHC

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Changes** |
| 0.1 | 7/11/13 | Initial version w/ pictures and both Shield & Micro |
| 0.2 | 7/20/13 | Added BOM and white wire directions for Micro board |
| 0.3 | 7/22/13 | Added section on programming the Arduino |

[1. CNC/Plasma System Requirements 5](#_Toc362285791)

[Everlast Power Plasma 50 5](#_Toc362285792)

[Mach 3 CNC Driver 5](#_Toc362285793)

[2. User Skill Level Requirements 5](#_Toc362285794)

[Electronics Skills 5](#_Toc362285795)

[Software Skills 5](#_Toc362285796)

[3. Theory of Operation 6](#_Toc362285797)

[Torch Voltage Behavior 6](#_Toc362285798)

[Torch Height Controller Functions 6](#_Toc362285799)

[THC Block Diagram 6](#_Toc362285800)

[THC Interfaces 7](#_Toc362285801)

[Torch On Interface 7](#_Toc362285802)

[Arc Okay Signal 8](#_Toc362285803)

[Plasma Voltage Interface 8](#_Toc362285804)

[A Word on Electrical Isolation 8](#_Toc362285805)

[A Word on Plasma Torch Voltage Polarity 8](#_Toc362285806)

[Plasma Torch Voltage Dividers 9](#_Toc362285807)

[Voltage Divider 9](#_Toc362285808)

[High Frequency Op Amp Filter 11](#_Toc362285809)

[Low Frequency RC Filter 11](#_Toc362285810)

[CNC Interfaces 12](#_Toc362285811)

[Serial Command Interface 12](#_Toc362285812)

[Power Supply Subsystem 13](#_Toc362285813)

[Arduino 13](#_Toc362285814)

[4. THC Boards 14](#_Toc362285815)

[Connecting to the Plasma 15](#_Toc362285816)

[Connecting to the CNC 16](#_Toc362285817)

[Connecting the Serial Port - Shield 16](#_Toc362285818)

[Connecting the Serial Port – Micro 17](#_Toc362285819)

[5. Board Assembly 18](#_Toc362285820)

[Shield 18](#_Toc362285821)

[Rev E-M Required Modifications / White Wires 19](#_Toc362285822)

[Torch Voltage (-) Not Tied to Ground 19](#_Toc362285823)

[+5 Volts Not Tied Relay and Arduino Micro Module 20](#_Toc362285824)

[6. Board Checkout 21](#_Toc362285825)

[Program the Arduino 21](#_Toc362285826)

[Get the Serial Port Working 21](#_Toc362285827)

[Test the Plasma Connections 21](#_Toc362285828)

[Test the CNC Connections 22](#_Toc362285829)

[7. Programming the Arduino 22](#_Toc362285830)

[Download the Arduino Tool Set 22](#_Toc362285831)

[Download the Source Code 22](#_Toc362285832)

[Building and Downloading the Source Code 22](#_Toc362285833)

[8. First Operational Test 24](#_Toc362285834)

[9. THC Operation 25](#_Toc362285835)

[Disabled Mode 25](#_Toc362285836)

[Bypass Mode 25](#_Toc362285837)

[Cruise Mode 25](#_Toc362285838)

[Operating Mode 25](#_Toc362285839)

[10. Shield Schematic 26](#_Toc362285840)

[11. Shield Layout 27](#_Toc362285841)

[12. Micro Schematic 28](#_Toc362285842)

[13. Micro Layout 29](#_Toc362285843)

[14. Bill Of Materials / Parts List 30](#_Toc362285844)

|  |  |
| --- | --- |
|  | This icon denotes a significant piece of information. |

|  |  |
| --- | --- |
|  | This icon denotes a very important warning. |

# CNC/Plasma System Requirements

This THC system was designed with specific requirements in mind. The target requirements are:

* Everlast Power Plasma 50 (PP50) plasma cutter
* Mach3 CNC Driver, licensed version required for THC use

## Everlast Power Plasma 50

The PP50 was selected as the initial plasma system to be supported for two reasons:

* It is a good performing and economical plasma cutter with a CNC port and a pilot arc start
* It’s the cutter I have

One aspect of the PP50 CNC interface is that it includes a 100K ohm resistor on both the + and – poles of the arc voltage signal. This plays a role in determining the voltage divider resistor values. While this THC uses undivided voltage, it is not recommended that it be connected directly to the torch voltages (if the plasma unit has no CNC interface). This would require hardware design changes.

## Mach 3 CNC Driver

The Mach 3 CNC driver application is recommended because:

* It is the configuration the system was designed to use
* The long term goal Is to integrate the THC controller directly into Mach

Mach is not required. A Windows PC application can be used to control the THC.

# User Skill Level Requirements

This system is not intended to be a turn-key system for people who are willing or able to build electronics or compile software source code and download the result.

## Electronics Skills

The following skills are required to be able to build the THC:

* Read a schematic
* Solder (and probably de-solder our mistakes)
* Order, identify and know how to handle discrete electronic components
* The ability to troubleshoot/debug any errors made during assembly of the system

## Software Skills

The following software skills are required to be able to build the THC:

* Install and use the Arduino development environment
* Compile and download an Arduino software project
* Read C/C++ with a very basic understanding to allow “tweaking” of key performance variables

# Theory of Operation

## Torch Voltage Behavior

The way a THC works is very simple. When using a plasma cutter the voltage used to cut will increase and decrease in relation to the height of the torch from the metal surface. (Plasma cutters have constant amperage and variable voltage.) Cutting voltages are typically under 200 volts DC. The PP50 cuts in the range of 90 volts to 120 volts (roughly).

|  |  |
| --- | --- |
|  | ***When the torch goes higher above the cutting surface, the voltage increases.***  ***When the torch goes closer to the cutting surface, the voltage decreases.*** |

One volt difference in the cutting voltages translates to about 15 thousandths of an inch. The target voltage to use when cutting is dependent on:

* Metal thickness
* Speed of torch motion

If the torch slows down while cutting (for example: when cutting a corner or changing direction), the distance the arc has to travel will increase. This will result in a higher voltage even though the torch is at the proper height. Since the THC has no way of knowing the speed of the torch, a setting in Mach 3 is used to instruct it to ignore the THC if the speed drops below a user set percentage of the cutting speed.

## Torch Height Controller Functions

The basic function of the THC is very simple. It compares the actual torch cutting voltage to the user set target voltage. If the torch cutting voltage is higher than the target voltage, it signals Mach to move the torch down. If the torch cutting voltage is lower than the target voltage, it signals Mach to move the torch up.

There are lots of special cases and other behaviors that will be described later, but that’s the basics of a THC.

## THC Block Diagram

The following is a high level block diagram of the THC.



As can be seen from the diagram, the fundamental components of the THC hardware are:

* Voltage Monitoring
* Torch On Relay
* RS-232 Opto-Isolator
* CNC Opto-Isolator
* 5 Volt Voltage Regulator

The “Voltage Monitoring” consists of:

* + - Voltage divider
    - High frequency op-amp filter
    - Low frequency Resistor-Capacitor (RC) filter

As seen, there are three different interfaces. They are:

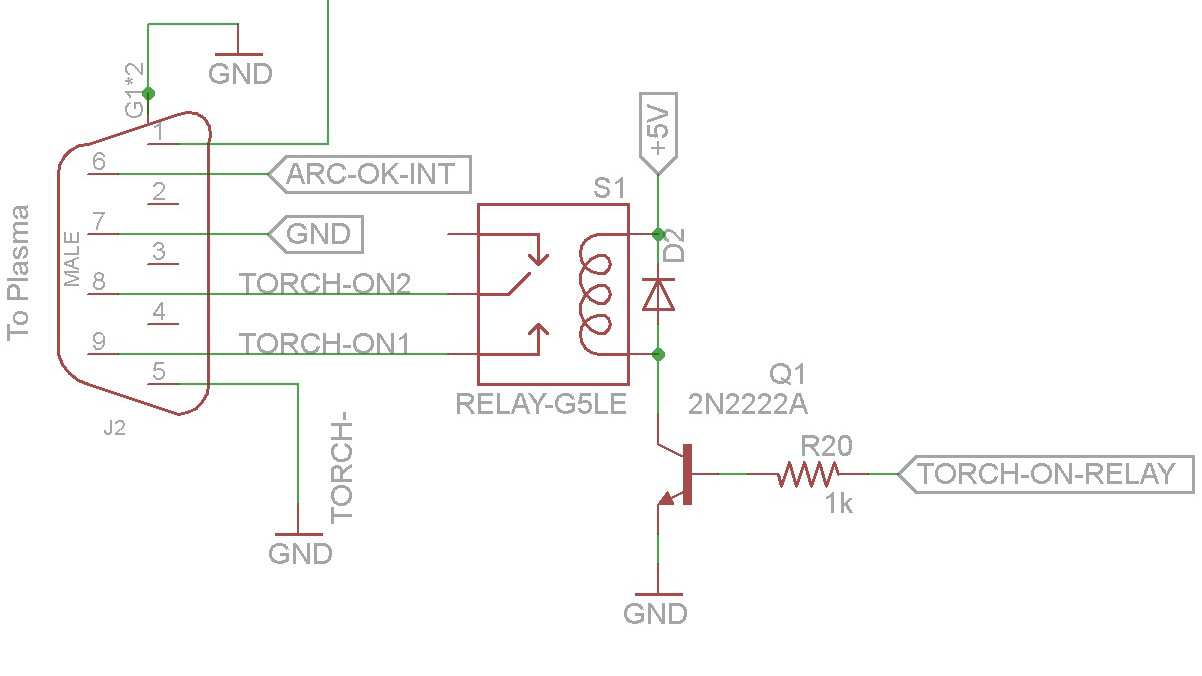
* Plasma Inteface
  + Torch On Output
  + Arc Good Input
  + Torch Voltage Input
* CNC Interface
  + Torch On Signal (input from CNC)
  + Arc Good Signal (output to CNC)
  + Torch Up Signal (output to CNC)
  + Torch Down Signal (output to CNC)
* Serial Control Interface (bi-directonal RS-232)

## THC Interfaces

### Torch On Interface

To use a THC you must have the ability to turn the torch on and off under computer control. This is typically not a digital interface. Generally plasma cutters require a relay (sometimes referred to as dry contacts) to close a circuit to turn the torch on, and open it to turn the torch off.

The torch on circuit uses a transistor to boost the output of an Arduino DigitalOut to sufficient current to control the relay. The relay circuit is connected directly to the Normally Open (NO) contacts of the relay.



### Arc Okay Signal

To use a THC the CNC system must know when the cutting arc is good so that it can start cutting. If, while cutting, the arc okay signal goes away, the CNC will stop motion.

The arc okay signal on the plasma is typically a relay (dry contact). To read this, an Arduino DigitalInput is connected to one of the two arc okay signals, and the ground from the Arduino is connected to the other.

The Arduino input is “tied high”. This means that it will read 5 volts if there is nothing connected to it. So, when the plasma closes the relay, the ground is connected to the input and it changes from 5 volts to 0 volts. This is commonly referred to as an “active low” signal.

|  |  |
| --- | --- |
|  | Electrical noise, or interference, can cause voltages to show up in places they wouldn’t normally be. If you have a signal configured for “active high”, or to go on when there is voltage present, electrical noise can cause a false signal to occur. |

|  |  |
| --- | --- |
|  | *This is a big safety issue.* You should always use “active low” to avoid a situation where the system is stopped and you’re in the process of working around the area of the torch and electrical noise causes it to turn on unexpectedly. |

### Plasma Voltage Interface

The plasma voltage interface is the most complex of all the electronics.

#### A Word on Electrical Isolation

Electrical isolation is having subsystems that interconnect that do not share a common ground. This prevents the transmission of noise between systems and the chance of unexpected voltages that can damage components/systems (ground loops).

Addressing grounds loops and isolation can be very complex. I took a very simple approach:

* the interface signals to the CNC are opto-isolated
* the USB/RS-232 interface to the PC is opto-isolated
* the Arduino is powered by a two-prong wall wart and the board is not tied to the common or A/C ground

Because the CNC system and the PC are opto-isolated, they cannot be damaged by problems with the torch or THC. I believe this approach isolates the THC sufficiently, bug I figured that if I was wrong the worst I could do is fry an Arduino and THC board (which relatively cheap to replace).

|  |  |
| --- | --- |
|  | This worked for me, but I’m not an electrical engineer and it may not be the best way to do this. It was very easy though. |

#### A Word on Plasma Torch Voltage Polarity

Plasma torches use a negative voltage to cut. The ground clamp is at 0 volts and the torch is at -120 volts (or whatever). If you take a battery powered volt meter (isolated from the plasma’s ground) and connect the negative lead to ground and the positive lead to the torch, you’ll see a negative voltage.

However, if you connect the positive lead to ground and a negative lead to the torch – you see a positive voltage.

If the CNC port on the plasma unit provides an isolated output (I don’t know if the PP50 output is isolated), or if you provide isolation by some other means, you can just read the voltage normally by swapping polarities.

#### Plasma Torch Voltage Dividers

The cutting voltage used as an input is a full voltage signal (up to 200 volts) with the potential for lots of electrical noise on it. This signal has to be dropped to a level the Arduino can read without being damaged (0 to 5 volts). Additionally, electrical noise on the signal must be reduced/removed.

The PP50 does provide a divided voltage signal that can either be 1/16th or 1/50th of the voltage. The following table shows how this works:

|  |
| --- |
| **Torch Voltages and Dividers** |
| |  |  |  | | --- | --- | --- | | **Actual Voltage** | **Divider** | **Output Voltage** | | 120 volts | None | 120 volts | | 120 volts | 1/16th | 7.5 volts | | 120 volts | 1/50th | 2.4 volts | |

What I had found in my testing is that I had about 2 volts of noise on the plasma signal. What this means is that no matter what voltage output is used, an extra two volts will be added on top of it.

If using the 1/16th divider the added noise would be more than 25% of the signal, if using the 1/50th divider the added noise would be almost as much as the signal. For this reason, the full voltage is used so that when it is divided down, the noise is divided down with it.

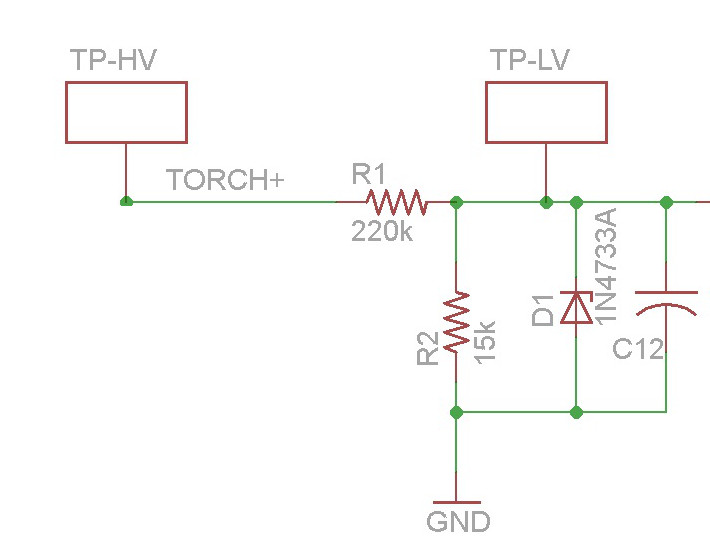
#### Voltage Divider

A simple voltage divider is used to step the voltage down from the cutting range to a maximum of 5 volts. While the torch voltage is specified to be up to 200 volts, in practice I have not seen a voltage greater than about 130 while cutting. As a result, I targeted a voltage divider value for a lower input voltage.

This gives better voltage resolution, but can result in the voltage input exceeded 5 volts and damaging the electronics. To address this, a Zener Diode is incorporated in the circuit to ensure the voltage does not exceed 5 volts.

C12 is a “spare” capacitor that I put in to make it easier to experiment with different filtering arrangements. The “TP-“ symbols are for optional Test Point connectors. These allow easy monitoring of voltages. “TP-HV” is for High Voltage (full torch voltage) and “TP-LV” is for the low voltage (divided and unfiltered voltage) , TP-FLT is for the final filtered voltage and TG-GND is for the ground.

The voltage divider circuit with Zener diode is shown below.



|  |  |
| --- | --- |
|  | For the PP-50 the value for R2 in the diagram above can bet either 15K or 18K. 15K Ohms will give 7 counts for volt. 18K Ohms will give almost 8 counts per volt. 18K will give better resolution, but less voltage range. |

The voltage reduction is determined by the following formula:

VOut = ( (R1) / (R1+R2) ) / VIn

So, for a 100 volt input, the formula would be:

VOut = ( 15,000 / (220,000 + 15,000) ) \* 100

VOut = 0.06383 \* 100

VOut = 6.383 Volts

**NOTE:** one thing to keep in mind is that the 100K ohm resistors on both the positive and negative legs of the voltage signal change the voltage that the circuit sees.

So, a 120 volt cutting voltage is seen as (XX) volts at the CNC port and the voltage divider. The voltage divider results in this being changed to (XX) volts at the output of the voltage divider.

The Arduino has a 10 bit Analog to Digital converter. This means that a 0 volt signal is seen as 0 “counts” and a 5 volt signal is seen as 1023 “counts”.

This results in the following formula to convert cutting voltage to analog counts (keep in mind that the tolerance, or accuracy, of the resistor adds a bit of slop to this calculation).

(insert overall formula)

So, the output of the voltage divider, along with all its noise is passed to the high frequency op amp filter.

#### High Frequency Op Amp Filter

Using an oscilloscope I found that the environment was fairly noisy. The noise seemed to be mostly from the PC (older commercial PC). Because that was there to start with, I didn’t pursue it further to see what each component of the system (steppers, stepper drivers, plasma) introduced.

The op-amp filter was designed using Texas Instruments free filter design application “FilterProDesktop”. A friend who is an electrical engineer advised on the optimum values to use.

Op-Amp selection is critical. While none is specified in the schematic, I started with a TI part that was a general purpose rail-to-rail op-amp. This resulted in never went below 1.8 volts and never exceeded 3.8 volts. Using a Linear Technologies op-amp that had almost the same specifics I got satisfactory results.

Other op-amps can be explored (the op-amp selected is around $7), but I did not do that since I had something that worked.

It’s worth noting that Analog Device’s circuit simulator (LTSpice) is an excellent tool for examining the behavior of circuits. I used this extensively to understand and tune the circuit design. It also helped me identify an op-amp that would work.

#### Low Frequency RC Filter

The op-amp filter will filter frequencies above (1K hz?). When examining the actual output of the filter there was a significant 60 hz noise component.

To remove the 60 hz noise I used an RC filter. It’s important to understand the difference between the (filter frequency) and the (stop band frequency).

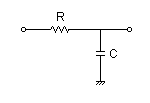
The (filter frequency) (is the frequency at which the filter function starts to affect the signal.

The (stop band frequency) is the frequency at which all signals above that are filtered out.

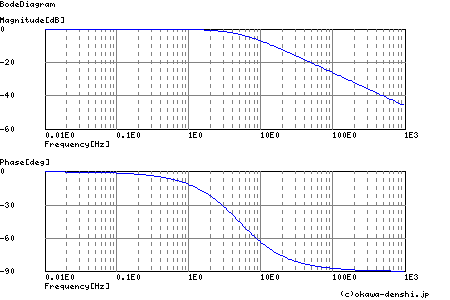
To filter out 60 hz noise, the (filter frequency) must be significantly lower than 60 hz. However, the lower frequency that the filter is, the slower the signal will show changes. If there is too much delay, you won’t be able to do accurate torch height control.

A good RC Low Pass Filter design tool can be found at: <http://sim.okawa-denshi.jp/en/CRlowkeisan.htm>

The schematic and formula for calculating the filter frequency are shown below:



Using the calculator and experimenting with different filter cut off frequencies, I ended up with an optimum value of 470 ohm resistor and 68 uF capacitor. The response chart for this is shown below.



In the first graph above, 1E0 indicates 0 Hz and, 10E0 is 10 Hz, etc.

The values used were a trade-off of noise reduction and signal response. The filter can be adjusted for an optimum behavior based on your goals. I wanted a fairly fast response so that I could detect when the torch was crossing a kerf (already cut area) to disable torch height control. This prevents the torch from being sent lower and “crashing’ into the metal at the other side of the kerf.

## CNC Interfaces

The CNC interfaces are straight-forward. There is one input signal to the THC and three output signals. These are all opto-isolated. For opto-isolation, the CNC must provide it’s 5 volt supply to power the opto-isolator.

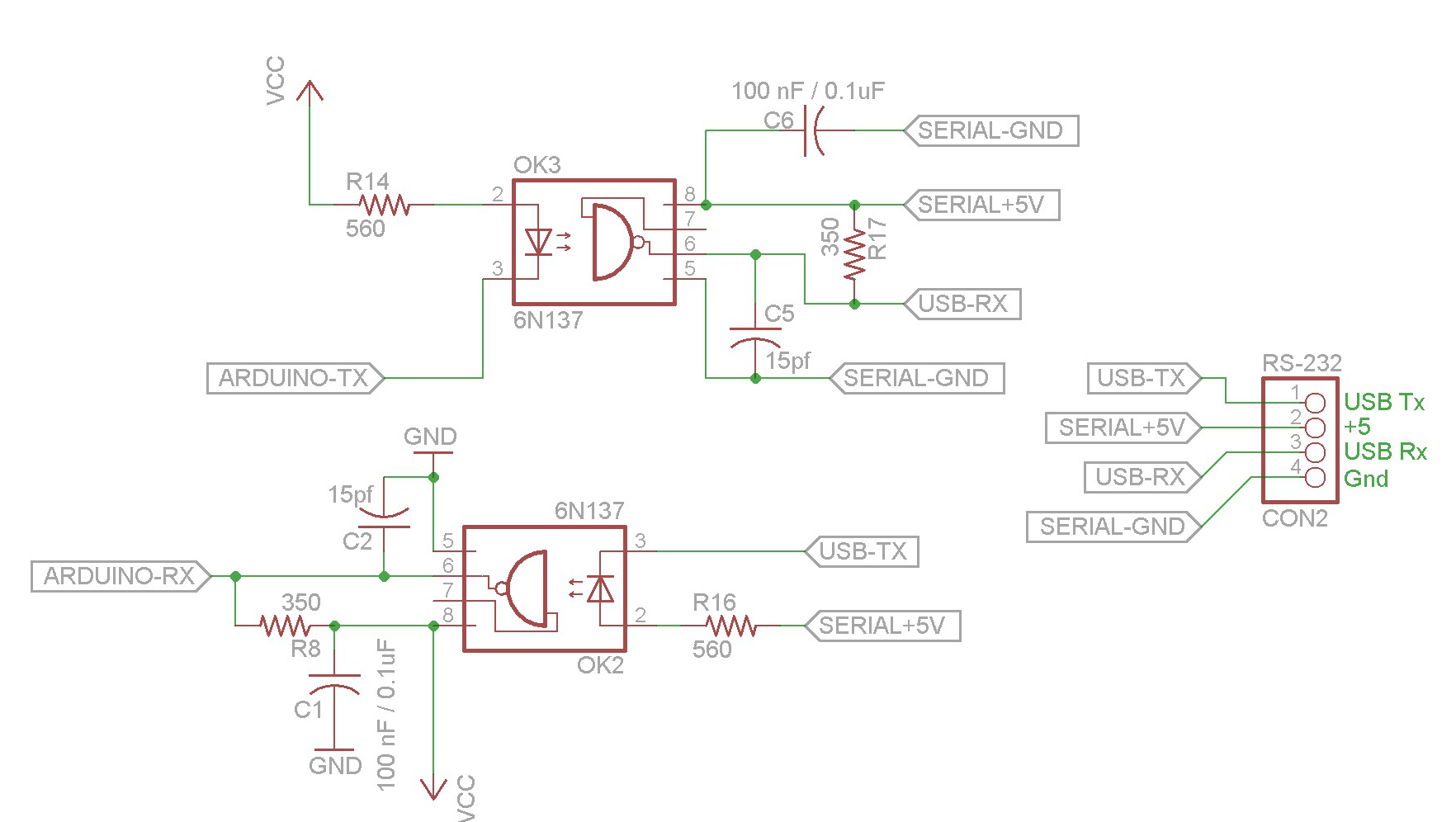
## Serial Command Interface

The initial design of the THC was “stand-alone”. It had push buttons and a 2 line LCD display. There were a number of issues with this:

* There was no way to capture the cutting voltages (needed to develop and debug the control algorithms)
* The LCD library seemed to have some performance issues
* Wiring the buttons was a rat’s nest
* The THC was controlled by its interface, but it was near the plasma/CNC but the system was controlled by the PC’s screen and keyboard (it was very difficult to interact with both)

As a result, I dumped the LCD/button interface and whet to a high speed serial interface. (The original source code is available is anyone wants to pursue that approach.)

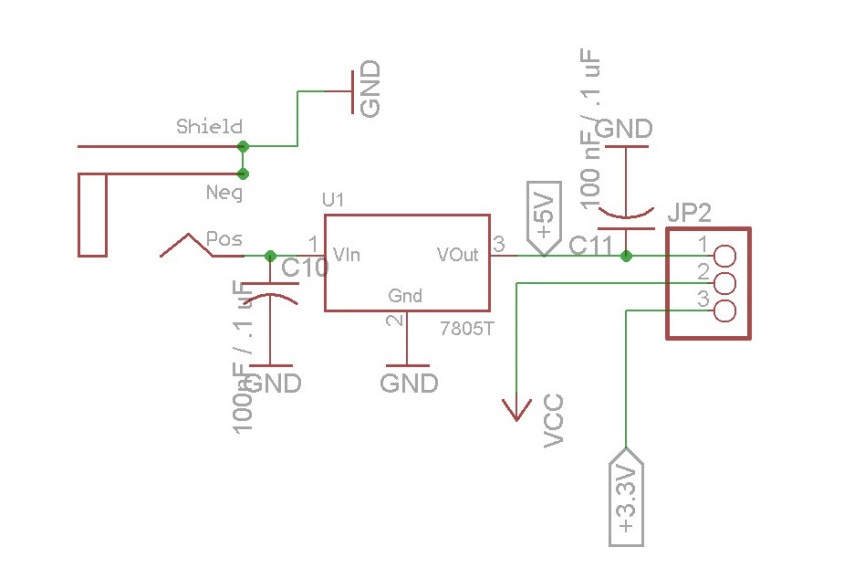
The Arduino provides TTL level serial data signals, not RS-232 levels. This allows the signals to be opto-isolated. However, the other size must provide power for its opto-isolators and level conversion from TTL to RS-232. I did this by using a cheap (FTDI) TTL to USB converter purchased from E-bay ($3).



|  |  |
| --- | --- |
|  | The serial cable must be shielded or the noise will make the system inoperative. |

## Power Supply Subsystem

An Arduino-compatible power input jack and voltage regulator are provided on the board. These are not necessary as the power can be provided by the Arduino. I put these on the board so that in the future I could move the THC to a Freescale Freedom board, which does not have a wall-wart connector.



## Arduino

I started the development with an Arduino Mega. I selected this processor because it is one of the fastest of the Atmel Arduinos (16 MHz) and had the most program (flash) and data (RAM) space.

Once the LCD was removed and the library was no longer needed, the memory requirements dropped significantly. The THC will now work with an UNO – but that has not been tested.

An Arduino Micro Pro version has also developed. ***The Arduino Micro Pro must be a 5 volt version.*** The 3 volt version is not currently supported.

# THC Boards

There are two versions of the THC. One is a shield style “S” and one uses a Micro Pro “M”.

The schematics and board layouts are at the end of this document.

The schematics and board layouts were done using EagleSoft CAD. There is a free version available that will allow you to open and examine both the schematic and the board layout. You can also generate a Bill Of Materials (BOM) that is a list of all parts necessary.

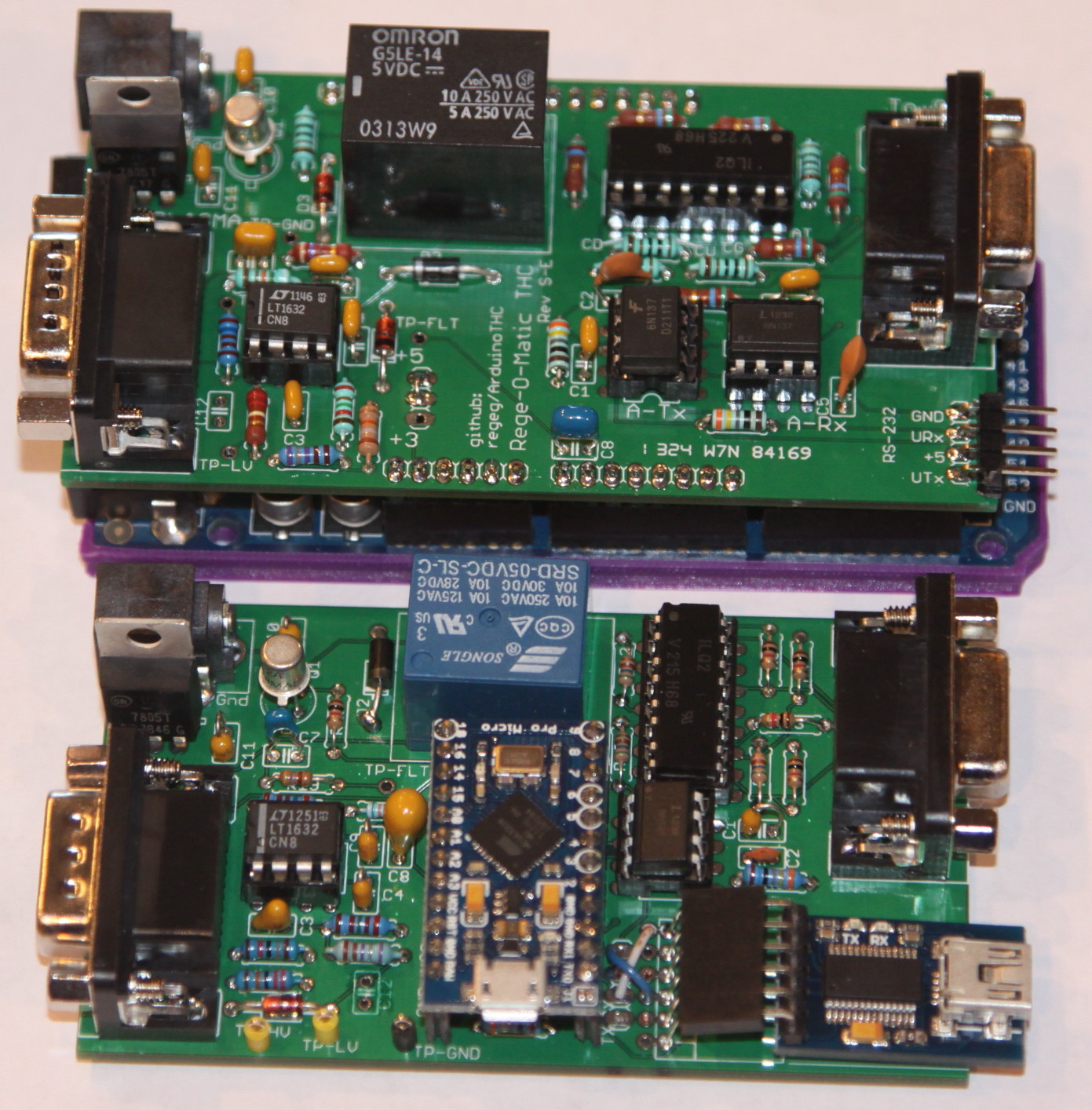
The design assumes that all capacitors are non-polarized. However, the capacitor for he RC voltage filter is marked with a + side in the event you use a polarized capacitor (i.e. tantalum or electrolytic).

|  |  |
| --- | --- |
|  | The power inlet jack, voltage regulator and associated capacitor are optional and are not needed for normal use with an Arduino system. |

|  |  |
| --- | --- |
|  | The test points are optional but assist in reading voltages or taking scope captures during operation. |

|  |  |
| --- | --- |
|  | Sockets for all IC’s are optional, but reduce the chance of heat damage when soldering and allow easier insertion and removal/replacement. |

Below are the completed THC’s. The top board in the image is the “Shield” version of the THC and it is mounted on an Arduino Mega board (the purple is a board protector made with a 3D printer). The bottom board in the image below is the “Micro” version of the THC.

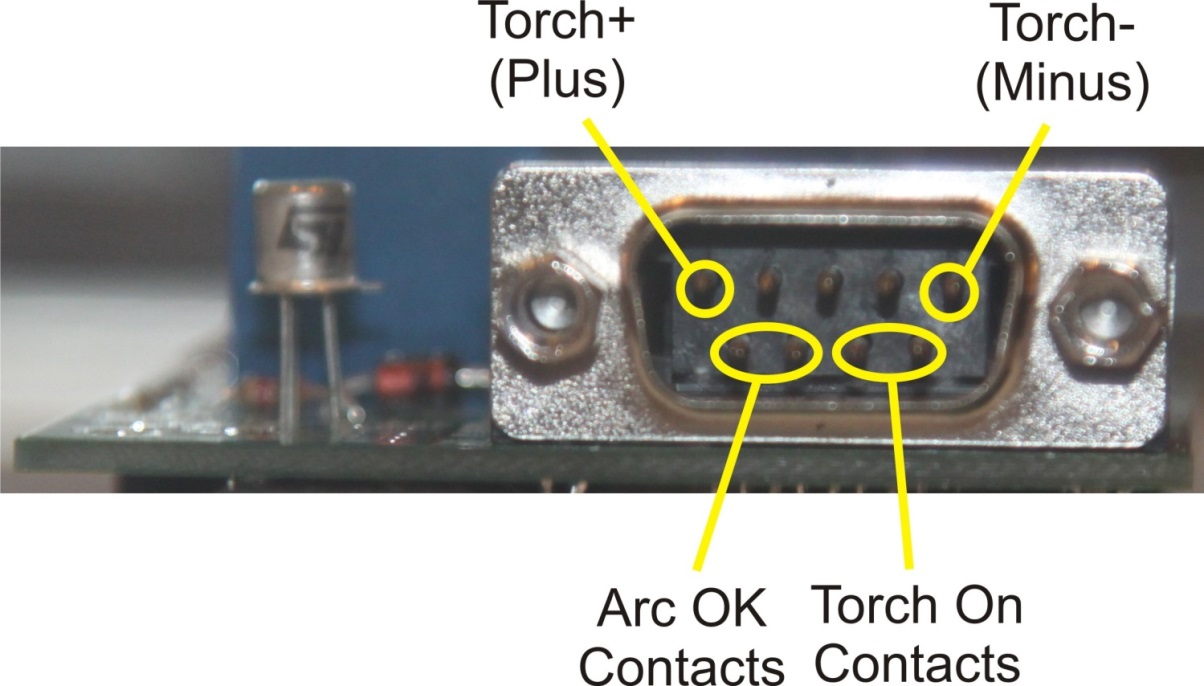


## Connecting to the Plasma

The board is designed to use right angle mount DB-9 connectors to provide for ease of connection and disconnection. If you choose, you can wire your cables directly to the board.

You must use shielded wire. And using a ferrite wouldn’t be a bad idea. (I connected the shield to the case of the Plasma Unit but left it unconnected on the THC side).

If you use the DB-9, the signals are located as shown.

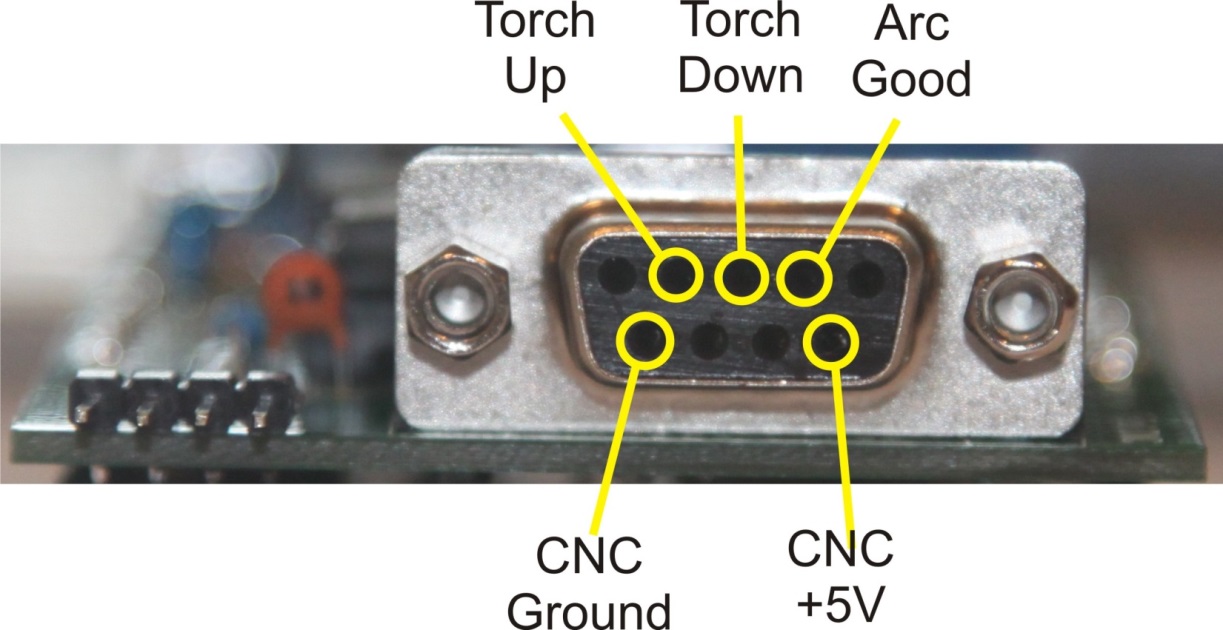


## Connecting to the CNC

The board is designed to use right angle mount DB-9 connectors to provide for ease of connection and disconnection. If you choose, you can wire your cables directly to the board.

You must use shielded wire. And using a ferrite wouldn’t be a bad idea. (I connected the shield to the case of the Plasma Unit but left it unconnected on the THC side).

If you use the DB-9, the signals are located as shown, if you are looking at the connector from the side the cable attaches to.



|  |  |
| --- | --- |
|  | The CNC must provide the 5 volt supply to operate the opto-isolators. |

## Connecting the Serial Port - Shield

The board is designed to use right angle .1” pin connectors (standard on the Arduino). However, you can solder your serial cable directly to the board.

You must use shielded wire. In my system I used shielded wire but did not need to connect the shield to anything. (Without the shielded wire there was too much noise for reliable data communication.)

NOTE: the PC/level converter must provide the 5 volt supply to operate the opto-isolators.

## Connecting the Serial Port – Micro

I used head pin sockets with extra long pings (the sockets used on shields with long pins to go into the Arduino board) and bent the legs. This is then used to plug the RS-232 to USB adapter into.

Since there are multiple configurations of RS-232 to USB adapters, all with different pin-outs, you must put jumpers in to connect the board signals to the adapter signals (+5, Gnd, Rx, Tx).

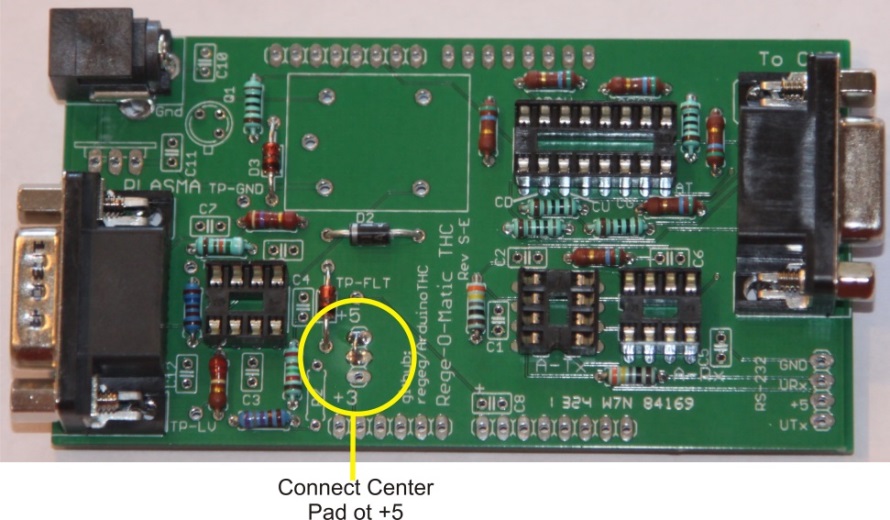
# Board Assembly

The following is the suggested order of assembly. While any order will work, after building multiple versions of the prototypes, I found this sequence to be easiest.

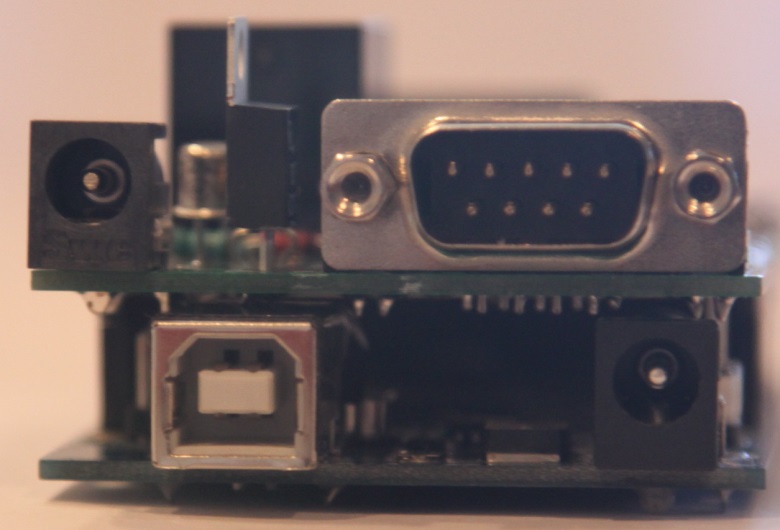
## Shield

1. Resistors
2. Diodes
3. 5 volt jumper (See below)
4. IC sockets (if desired)
5. Test points (if desired)
6. DB-9’s
7. Right angle header pin for serial to USB (best to solder from the top of the board)
8. Power inlet (if desired)
9. Header pins (This is best done by putting them in an Arduino board and placing the shield on top. Solder one pin on each end of each header. Then remove the shield from the Arduino board and solder the rest of the pins)
10. Relay
11. Voltage Regulator (if desired)
12. Transistor
13. Capacitors

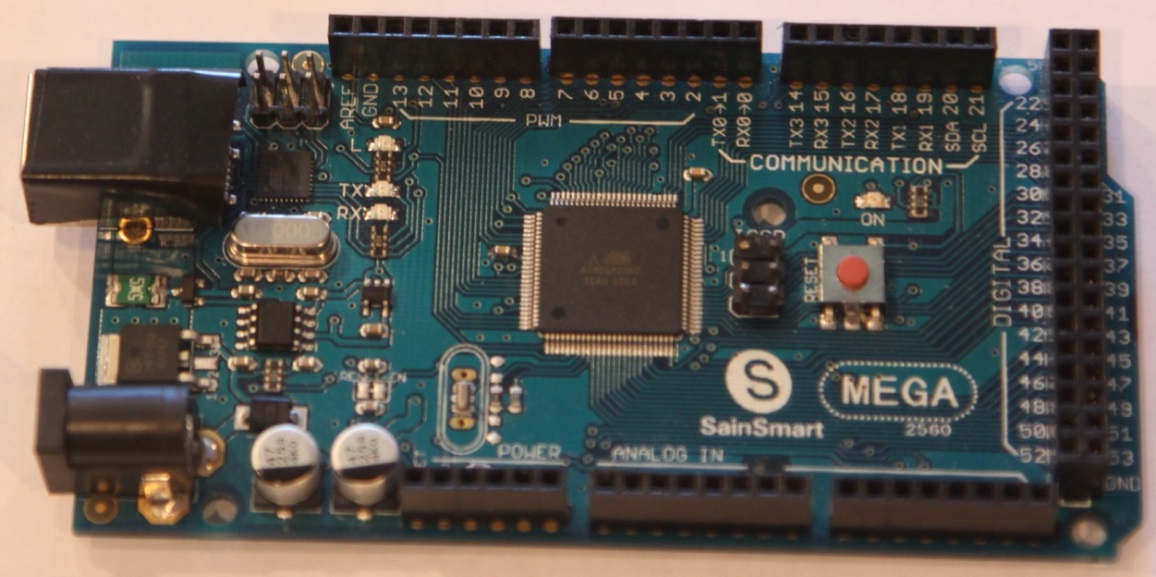
**Step 3:** The shield was designed so that it could be used with a 3V Arduino or Arduino compatible board (Freescale Freedom). This has not been tested! However, for the board to work – you must put a jumper between the center and +5V pad as shown in the drawing.



In the photo below you can see how the bottom side of the components of the power circuit contact the outside shield of the USB connector.



|  |  |
| --- | --- |
|  | *DON’T FRY YOUR BOARD!* One drawback of the design is that there are soldered connections that come in contact with the metal shield on the Arduino’s USB port. You will reduce this if you don’t build the power supply portion of the board. In either case (building the power supply or not) you should cover the USB shield in at least 2 or 3 layers of electrical tape. |



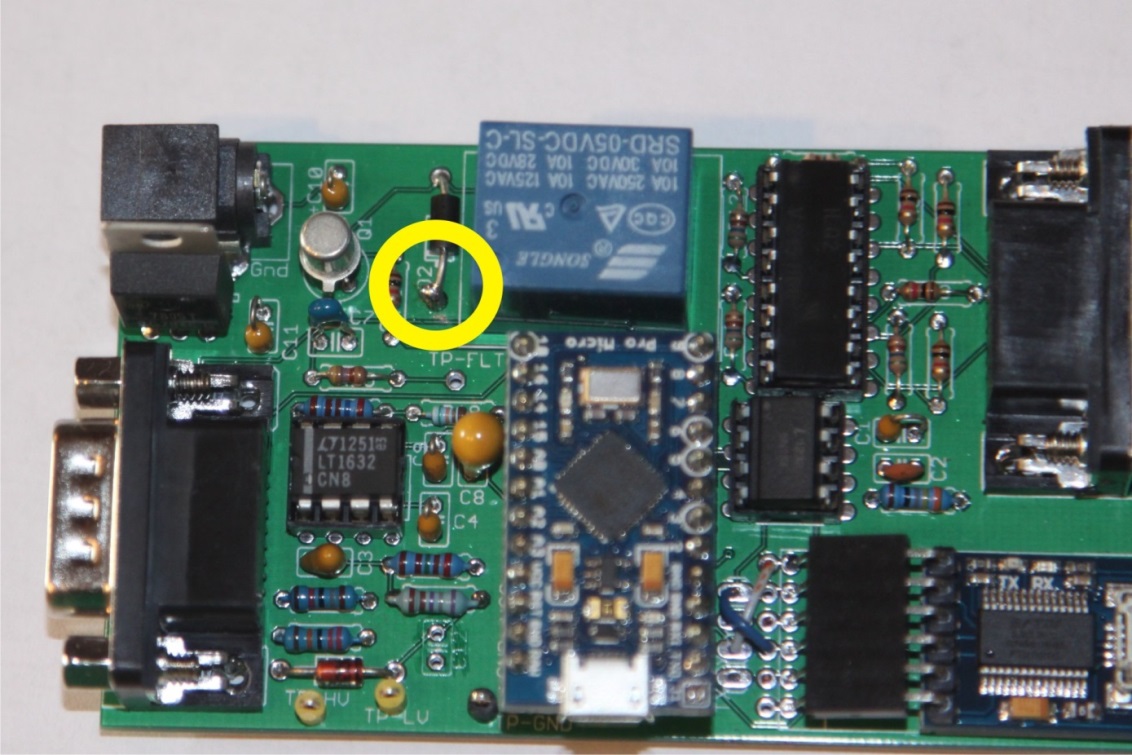
## Rev E-M Required Modifications / White Wires

The design has gone through many iterations but the Rev E version introduced the Micro design. As a result of being the first pass, there are two missing connections that require additional work for proper operation.

### Torch Voltage (-) Not Tied to Ground

At the DB-9 connector to the plasma unit, the Torch Voltage (-) signal is not tied to ground. The bottom copper plane of the board (all copper areas that are not traces) is a ground plane. The input signal can be connected to that by carefully scraping the solder mask away to allow soldering to the copper underneath. Alternately, you can run a wire from the DB-9 pin to another pin wired to ground.

The picture below shows the bottom of the board where solder mask was removed from the ground plane and a solder bridge was added between the ground plane and the pin.

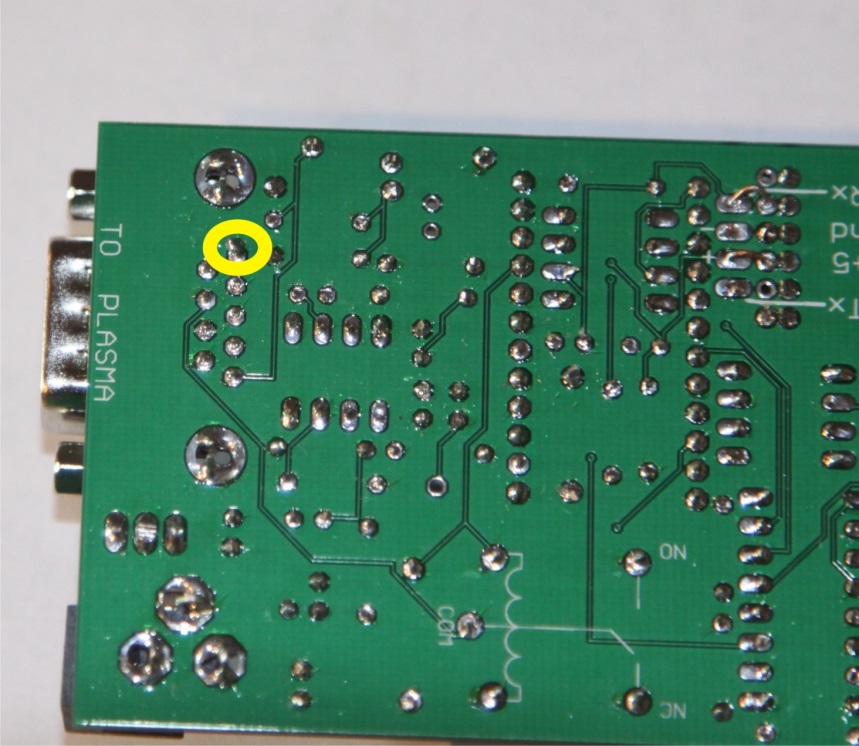


### +5 Volts Not Tied Relay and Arduino Micro Module

The Arduino Micro module was built to be single voltage (+5 volts) only. When the jumper was removed one net that connects the diode on the torch-on relay and the +5 volt supply to the Arduino Micro was not connected to the +5 volt supply.

The top copper plane of the board (all copper areas that are not traces) is a +5 volt plane. The net connecting the diode and Micro can be connected to +5 volts by carefully scraping the solder mask away to allow soldering to the copper underneath. Alternately, you can run a wire from the diode to a +5 pin.

The picture below shows the top of the board where solder mask was removed from the ground plane and a solder bridge was added between the +5 volt plane and the diode.



# Board Checkout

If you build the board properly, it should all just work. That probably won’t be the case, so you’ll need to troubleshoot your hardware. Trouble shooting is beyond the scope of this manual, but the following order is suggested.

1. If you build the power supply, without plugging in the Arduino, power up the shield and measure voltages to ensure it is operating correctly.
2. Program the Arudino
3. Get the serial port working
4. Test the relay
5. Test the CNC connections
6. Test the Plasma connections

## Program the Arduino

You need to program the Arduino with the most current source code. When no THC is connected and the Arduino is running the THC software, you should see the serial port LED flashing regularly.

Remove power and connect the THC. Use the wall-wart to provide power. Verify that the serial port LED is flashing. If all LED’s go off – you have +5 shorted to ground somewhere on your board.

**NOTE:** When running the Arduino with the THC attached, you must use a wall-wart for power. Using the USB to power the system will prevent the serial port from operating properly and can cause a ground loop that can destroy your computer.

## Get the Serial Port Working

By getting the serial port working first, you can use the stand-alone windows application to ensure the system is running and debug the connections to the Plasma and CNC. If it stand-alone app doesn’t work right away, you may want to use a terminal program to send characters and look at what’s received as you’re debugging (TeraTerm is a good choice).

## Test the Plasma Connections

Once the serial port is working the stand-alone Windows app can be used to test the Plasma connections.

Before connecting the torch, make sure the relay clicks on and off when commanded from the Windows app.

Next, short the two Arc Good signals and verify that the signal status on the Windows app is updated appropriately.

|  |  |
| --- | --- |
|  | Make sure the torch is anchored in a safe location before attempting to turn it on. |

**NOTE:** You may need to turn the torch on frequently and run it on pilot arc for a while to debug. This is very hard on consumables. If you have old crappy consumables – they are good to use for this.

Once you know that the relay works and the Arc Good work, connect the torch and verify it turns on and off when the relay is flipped.

If the torch control is working you should see the voltage display updated. If it is not, you need to run the torch while debugging the voltage divider and filter circuits.

|  |  |
| --- | --- |
|  | You will be debugging a circuit that can have up to 200 volts on it. Be careful. |

## Test the CNC Connections

|  |  |
| --- | --- |
|  | The CNC connections should initially be debugged with the plasma disconnected. |

First, use the test capability of the Windows app to generate the Arc Good, Torch Up and Torch Down signals. You can use the “Diagnostics” screen in Mach to verify if they are working correctly.

If they are not, it can either be a problem with your board and/or cables or with your Mach configuration. Double check connection of your CNC output signals to the pins on the Break Out Board (BOB) and verify that is how the “Outputs” are configured in Mach.

The main screen of the default plasma screen set in Mach provides the ability to turn the torch on and off. You can use this and monitor the signal status on the Windows app to see if this signal is correct.

# Programming the Arduino

## Download the Arduino Tool Set

The Arduino compiler and tool set are constantly being updated. The currently supported version is 1.05. The software should work on new versions, but changes in the compiler can sometimes cause compile errors that require minor changes.

To download the Arduino tool set:

1. Go to [www.arduino.cc](http://www.arduino.cc)
2. Click the “Download” label in the top title bar
3. You can use either the “Windows Installer” or the “Windows (ZIP File)”.
   1. If you use the “Windows Installed”, run the program after downloading
   2. If you use the “Windows ZIP file”, open the file after downloading and copy to a convenient location (usually “C:\Arduino-1.0.5” (or something similar)

Run the Arduino IDE and make sure it works.

If you’ve not used the Arduino before, try some of the example software to get the hang of the software and downloading.

## Download the Source Code

To build the software and load it on the Arduino, you need to have the source code. All source code is maintained on GitHub ([www.github.com](http://www.github.com)).

1. Go to the Git Hub site
2. Enter “regeg/ArduinoTHC” in the search box on the top menu bard of the window.

|  |  |
| --- | --- |
|  | The Git hub name you search on is case sensitive! You must match the case: “***regeg/ArduinoTHC***” |

1. Once the project page comes up, click the “Download ZIP” on the right size of the page.
2. Save the ZIP file somewhere convenient
3. Open the zip file and copy the directories to a convenient location

## Building and Downloading the Source Code

Now that you have the source code downloaded, you must build it and program the Arduino.

1. Start the Arduino development environment
2. Enter the “ArduinoTHC” directory (in the location you previously downloaded it)
3. Enter the “ArduinoSource” Directory
4. Enter the “THC\_REV-EM sketch”
5. Open the “THC\_sketch.ino” file
6. The project should now be opened with a number of tabs across the top (all the files in the project)
7. Select the “Platform.h” file.
8. Set the “define” for the hardware you are using
   1. If you are using an Arduino Micro Pro, make sure the line with “#define MICRO\_THC” **IS NOT** commented out (a “//” denotes a comment
   2. If you are using an Arduino Mega, make sure the line with “#deifne MICRO\_THC” **IS** commented out.
9. Plug in your Arduino (if using an Arduino Mega, unplug the shield before programming)
10. Select the “Board” Menu
11. Select the board type you are using, either
    1. Arduino Mega 2560, or
    2. Arduino Leonardo (for the Micro Pro)
12. Select the serial port that the board is on
13. In the tool bar, click the icon with a check mark. The software should compile without errors.
14. Click the icon with the arrow pointing to the right to download the code to the board

Once programmed successfully, you should be ready to go.

# First Operational Test

For the first operational test of the THC, you should set the THC to “Bypass” mode. This should allow the system to make cuts without any THC control. While cutting you should verify that the signal statuses are updated (Torch On, Arc Good, Torch Up, Torch Down), the voltage is updated and the system properly cuts material.

# THC Operation

The THC software provides for four modes of operation. They are shown in the state diagram below.

(put in state diagram)

It should be noted that mode changes are not possible when the Torch On or Arc Good signal are active (except for Cruise Mode to Enabled Mode).

## Disabled Mode

Disabled mode will not pass any control signals through so the plasma torch will be non-operational.

|  |  |
| --- | --- |
|  | **DO NOT** use Disabled Mode as an alternative to turning the plasma off while working on the system. |

The only way to exit Disabled Mode is through the mode change button.

## Bypass Mode

Bypass mode will pass all signals through to allow cutting and will display cutting voltage. However, it will not perform any torch height control.

The only way to exit Bypass Mode is through the mode change button.

## Cruise Mode

Cruise mode will pass all signals through to allow cutting. When commanded to “go”, the system will go into Enabled mode and start height control using the actual voltage when the “go” command is received as the target voltage.

This mode is intended to be used with a touch-and-go torch holder when cutting a new material and you don’t know the desired cut voltage. The idea is that the touch-and-go controller gets the torch to the right height and once you verify that, you can have it control the height based on that initial cut height.

The Cruise Mode can be exited by the mode change button, or using the “go” command while cutting.

## Operating Mode

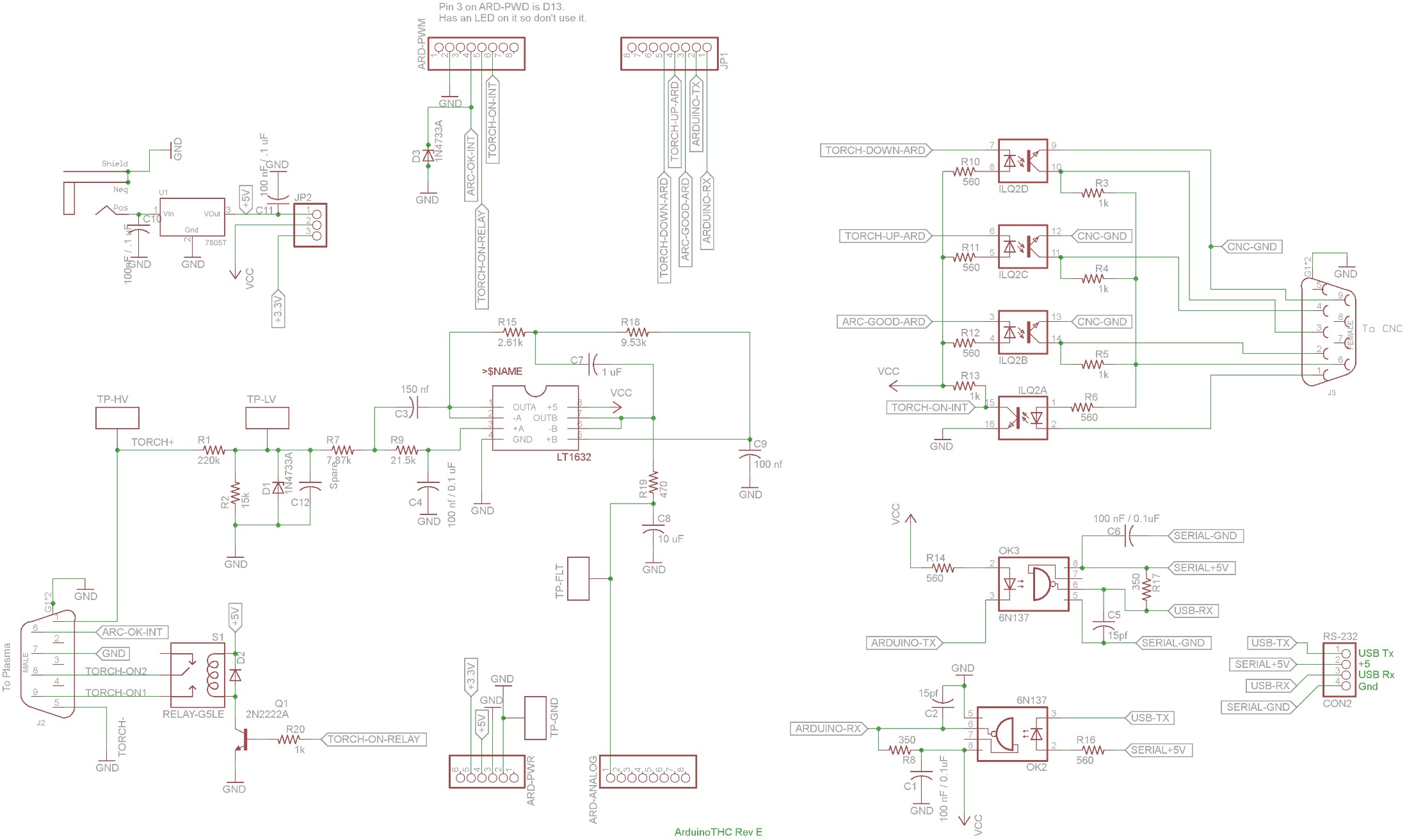
This is the normal mode for cutting with torch height control. Within the Operating Mode, there are two states. They are:

* Enabled
* Cutting

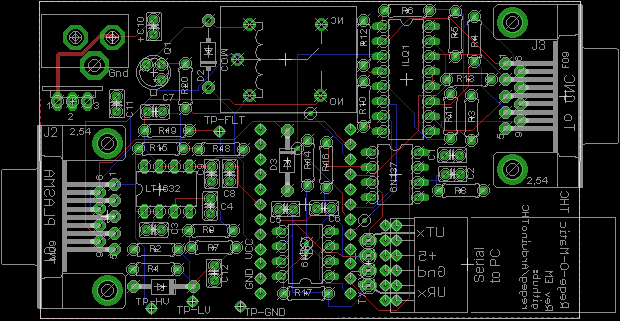
Enabled means that the system is ready to cut, but idle.

Cutting is entered once the Torch On signal is received from the CNC and left when Torch On and Arc Good are no longer active.

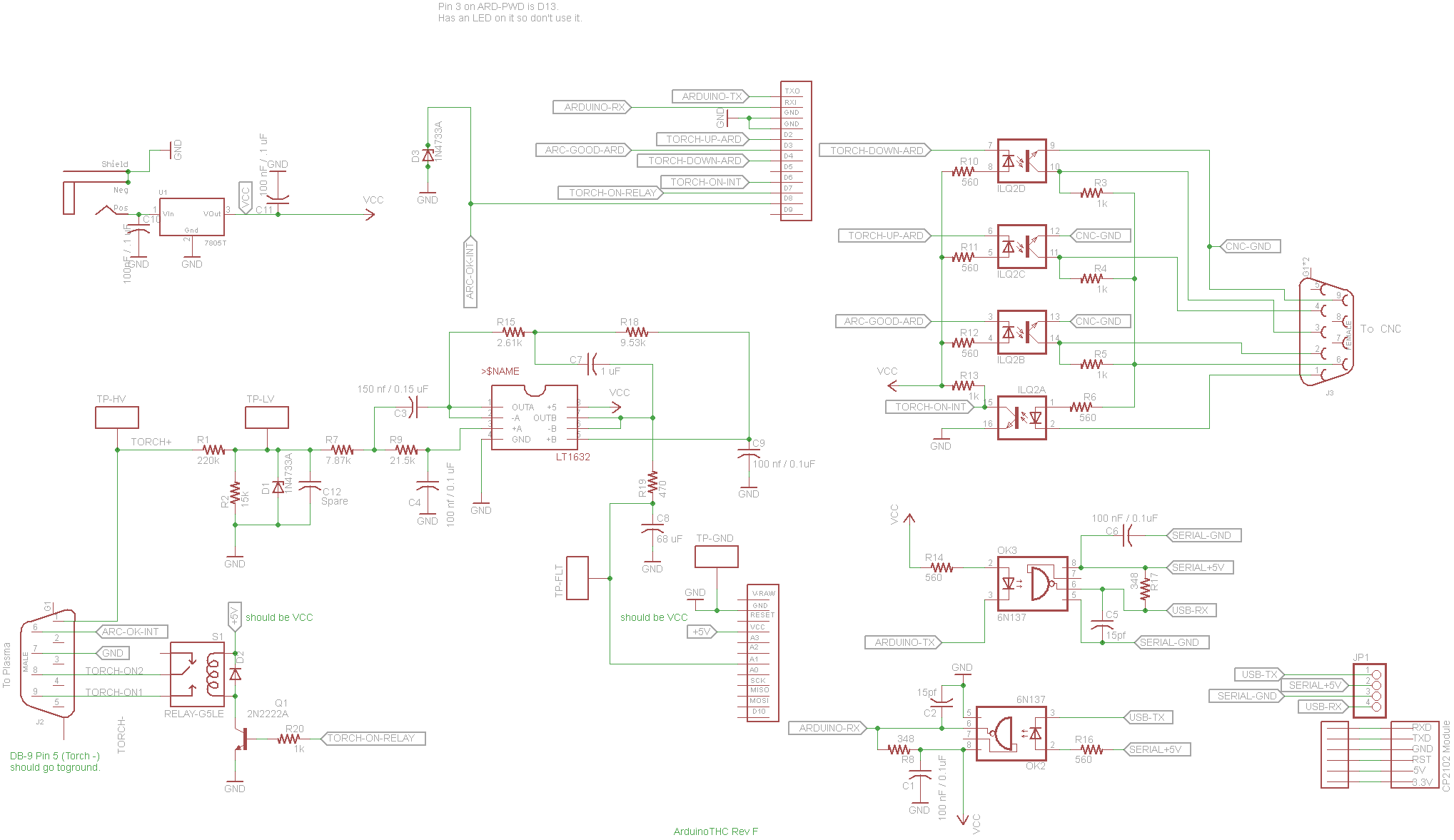
# Shield Schematic



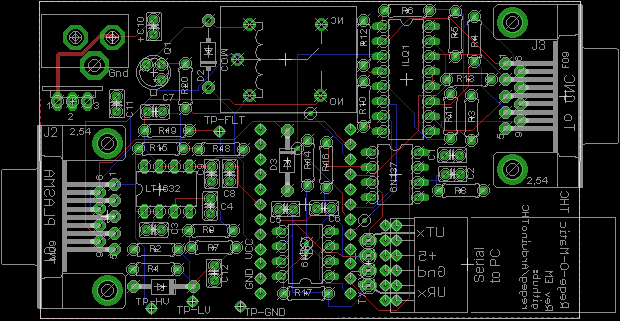
# Shield Layout



# Micro Schematic



# Micro Layout



# Bill Of Materials / Parts List

The following is a parts list (Bill Of Materials) that covers both the Shield and the Micro version. The specified part numbers are meant as a guide only.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Headers (Male & Female)** | |  |  |  |  | |  | |  |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | | **Price** | | **Notes** |
| 1 | RS-232 Connector | 1x4 Pin Header |  |  | 0.1" Male Header - Right Angle | |  | | Shield Only |
| 1 | ARD-PWR | 1x6 Pin Header |  |  | 0.1" Male Header - PCB mount, Straight | |  | | Shield Only |
| 3 | ANALOG, PWM, JP2 | 1x8 Pin Header |  |  | 0.1" Male Header - PCB mount, Straight | |  | | Shield Only |
| 2 | Arduino Micro Connector | 1x20 |  |  | 0.1" Female Header/Socket | |  | | Micro Only |
| 1 | USB Module Header | 1x6 |  |  | 0.1" Female Right angle, or 0.1" Long Pin Female Header | |  | | Micro Only |
|  |  |  |  |  |  | |  | |  |
| **Capacitors** | |  |  |  | Note: 2.5 mm spacing ideal (not all caps are that spacing | | | | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 4 | C1, C4, C6, C9 | 100 nF / 0.1uF | 399-9785-ND | $0.24 | Capacitor | $0.96 | |  | |
| 2 | C2, C5 | 15pf | DigiKey: 1427PH-ND | $0.30 | Capacitor | $0.60 | |  | |
| 1 | C3 | 150 nf / 0.15uF | 399-4278-ND | $0.36 | Capacitor | $0.36 | |  | |
| 1 | C7 | 1 uF | 478-7666-ND | $0.73 | Capacitor | $0.73 | |  | |
| 1 | C8 | 68 uF | 445-8249-ND | $0.86 | Capacitor | $0.86 | |  | |
|  |  |  |  |  |  |  | |  | |
| **Diodes & Transistor** | |  |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 2 | D1, D3 | 1N4733A | 1N4733AFS-ND | $0.23 | Zener Diode | $0.46 | |  | |
| 1 | D2 | 1N4001 | 1N4001-E3/54GICT-ND | $0.48 | Diode | $0.48 | |  | |
| 1 | Q1 | 2N2222A | 497-2598-ND | $1.26 | NPN Transistor | $1.26 | |  | |
|  |  |  |  |  |  |  | |  | |
| **IC's and Sockets** | |  |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 1 | ILQ2 | ILQ1 | 751-1330-5-ND | $2.47 | 4 Channel Opto-Isolator | $2.47 | |  | |
| 2 | OK2 | 6N137 | 160-1791-ND | $0.90 | 1 Channel Opto-Isolator | $1.80 | |  | |
| 1 | U$1 | LT1632 | LT1632CN8#PBF-ND | $6.66 | Opamp | $6.66 | |  | |
| 1 | Socket for ILQ2 |  |  |  | 12 Pin DIP IC Socket |  | | Optional, but recommended | |
| 3 | Socket for Optos & Op Amp |  |  |  | 8 Pin DIP IC Socket |  | | Optional, but recommended | |
|  |  |  |  |  |  |  | |  | |
| **DB-9 Connectors** | |  |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 1 | J2 | DB9-MALE | 609-4003-ND | $0.95 | DB9M-RIGHT-ANGLE | $0.95 | | Can wire directly instead of using a connector | |
| 1 | J3 | DB9-FEMALE | 609-4009-ND | $0.95 | DB9F-RIGHT-ANGLE | $0.95 | | Can wire directly instead of using a connector | |
|  |  |  |  |  |  |  | |  | |
| **Resistors** | |  |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 5 | R3, R4, R5, R13, R20 | 1k | PPC1.00KYCT-ND | $0.26 | Resistor | $1.30 | | resistors 6.5mm x 2.5mm | |
| 2 | R8, R17 | 348 | PPC348YCT-ND | $0.26 | Resistor | $0.52 | |  | |
| 1 | R19 | 470 | PPC470W-1CT-ND | $0.37 | Resistor | $0.37 | |  | |
| 6 | R6, R10, R11, R12, R14, R16 | 560 | PPC560W-1CT-ND | $0.37 | Resistor | $2.22 | |  | |
| 1 | R15 | 2.61k | 2.61KXBK-ND | $0.10 | Resistor | $0.10 | |  | |
| 1 | R7 | 7.87k | 7.78KXBK-ND | $0.10 | Resistor | $0.10 | |  | |
| 1 | R18 | 9.53k | PPC9.53KYCT-ND | $0.26 | Resistor | $0.26 | |  | |
| 1 | R2 | 15k | PPC15.0KYCT-ND | $0.26 | Resistor | $0.26 | | Alternate 18k / S18KHCT-ND | |
| 1 | R1 | 220k | PPC220KW-1CT-ND | $0.37 | Resistor | $0.37 | |  | |
| 1 | R9 | 21.5k | PPC21.5KYCT-ND | $0.26 | Resistor | $0.26 | |  | |
|  |  |  |  |  |  |  | |  | |
| **Relay** |  |  |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 1 | S1 | RELAY-G5LE | Z1011-ND | $1.48 | Relay | $1.48 | |  | |
|  |  |  |  |  |  |  | |  | |
|  |  |  |  |  | **Total (for priced parts above)** | **$25.78** | |  | |
|  |  |  |  |  |  |  | |  | |
| **Optional - Shield or Micro** | |  |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 4 | TP-FLT, GND, HL, LV | TEST-POINT-.040-HOLE | 5001K-ND | $0.33 | Test Point | $1.32 | |  | |
|  |  |  |  |  |  |  | |  | |
| **Required on Mirco / Optional on Shield** | | |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
| 1 | J1 | N/A | SC1313-ND | $1.57 | 2.1MM-PLUG | $1.57 | |  | |
| 1 | U1 | N/A | NCP7805TGOS-ND | $0.46 | 7805T Linear voltage regulator | $0.46 | |  | |
| 2 | C10,C11 | 100 nF / 0.1uF | 399-9785-ND | $0.24 | Capacitor | $0.48 | |  | |
| 0 | C12 | N/A | N/A | $0.00 | Capacitor, Unused | $0.00 | |  | |
|  |  |  |  |  |  |  | |  | |
| **Shield Only - Optional** | |  |  |  |  |  | |  | |
| **Count** | **Part** | **Value** | **Part Number** | **Unit Cost** | **Description** | **Price** | | **Notes** | |
|  | JP2 | 0.1" Pin Header |  |  | PINHD-1X3 Voltage jumper |  | |  | |
|  | for JP2 | 0.1" Jump |  |  | Jumper |  | |  | |