# Bennington College Small Radio Telescope Operations Manual

*Andrew Cencini – Hugh Crowl*
*William Buchanan, Alexander Curth, Erick Daniszewski, Evan Gall, Clemente Gilbert-Espada, Chernoh Jalloh, Brendon Walter*

*with guidance from*
**MIT Haystack Observatory's**
Haystack Small Radio Telescope Project

# Contents

# 1 Introduction

This manual serves as a guide following the work done at Bennington College during the spring of 2014 to build and operate a radiotelescope, following the procedure layed out by MIT Haystack and their Small Radio Telescope (SRT) project. This manual contains information regarding both the hardware and software components of the project which should allow for a simple and successful build and operation of a SRT.

# 2 Parts List

## 2.1 LNA to Dongle Interface

| **Part** | **Qty.** | **Manufacturer** | **Cost** |
|---|---|---|---|
| SMA-M Crimp Connector Straight | 1 | Amphenol/Connex | $5.95 |
| LMR-400 | 100ft | | $102.99 |
| Type-F Male Crimp Connector | 1 | Amphenol/Connex | $4.95 |
| In-Line Amplifier | 1 | Blonder Tongue | $8.22 |
| BNC Female Jack to Type-F Male Plug | 3 | | $10.99 |
| BNC-M to BNC-M Patch Cable | 3 | Amphenol RF | $12.55 |
| Power Injector | 1 | Channel Vision | $14.71 |
| BNC-F to SMA-M Adapter | 1 | Vitelec / Emerson Connectivity Solutions | $12.09 |
| Bandpass Filter | 1 | Mini-Circuits | |
| SMA-F to BNC-F Adapter | 1 | AIM-Cambridge / Emerson Connectivity Solutions | $4.21 |
| BNC-F to SMB Plug Adapter | 1 | | $8.49 |
| AC-to-DC Power Supply | 1 | | $20.18 |
| Breadboard | 1 | Vector Electronics | $5.75 |
| DC Jack | 1 | CUI Inc | $1.00 |
| F-Type Plug | 1 | Bomar Interconnect/ Winchester Electronics | $4.16 |

## 2.2 Feed & LNA

| **Part** | **Qty.** | **Manufacturer** | **Cost** |
|---|---|---|---|
| Ultra Low Noise Amplifier | 1 | Mini-Circuits | |
| Coaxial Bias-Tee | 1 | Mini-Circuits | |
| Coaxial Cable 086-6SM+ | 1 | Mini-Circuits | |
| Coaxial Cable 086-4SM+ | 1 | Mini-Circuits | |
| SMA-F to SMA-F Adapter | 1 | Mini-Circuits | |
| SMA-F to Solder Pin Bulkhead Connector | 1 | Emerson Network Power Connectivity Johnson | $8.06 |
| SMA-M to SMA-M Right Angle Adapter | 1 | | |
| 3M Adhesive Copper Foil Tape | 1 | 3M | |
| 22 Gauge Copper Wire | 1 | | |
| Semi-Rigid Coaxial Cable | 1 | Micro-Coax | $4.62 |
| F-Type Male to BNC Female | 1 | | $0.89 |
| SMA-M to SMA-M Coupler | 1 | | $7.49 |

## 2.3 Mount

| **Part** | **Qty.** | **Manufacturer** | **Cost** |
|---|---|---|---|
| 1/4” (3’ x 2’) cold rolled steel plate | 1 | | Found scrap metal |
| 2 1/2” (11’) long steel pipe | 1 | | Found scrap metal |
| 1/8” 2” (6’) angle iron | 1 | | Found scrap metal |
| Black spray paint | 1-3 | Any | $4 - $12 |

# 3 Tools List

## 3.1 Assembling the LNA and Feed

| **Tool** | **Description** |
|---|---|
| Soldering iron + solder | soldering components as specified by Haystack documentation |
| Wire cutter/ strippers | cutting 22 gauge copper wire |
| Screwdriver | assembling box of LNA |
| Labeler | labeling LNA |
| Hacksaw | cutting styrofoam |
| Metric measuring tape | measuring styrofoam, and helix position |
| Drill press and 1/4" bit | drilling hole into styrofoam, LNA baseplate, cake pan, pc board |
| X-acto knife | cutting copper tape |
| Straight edge | cutting copper tape |
| Marker | marking position of helix |

## 3.2 Assembling the Dish

| **Tool** | **Description** |
|---|---|
| Socket wrench set | bolting arms onto center |
| Hammer and tap | marking ends of arms and outer band to make drilling holes easier |
| Handheld Riveter/ rivets | riveting the outer band and strips over mesh on the arms. |
| Hacksaw | |
| Power drill + Drill bit | drilling holes to place rivets connecting outerband, mesh, strips, and dish arms |
| Safety Gloves | for your fingers |
| Pliers | Useful for bending mesh when wrapping around edge |
| Tin Snips | cutting mesh |
| Pen/Marker | mark hole locations pre-drilling |

### 3.3 Building the Mount

| Tool | Description |
|---|---|
| Safety glasses | Eye protection for grinding |
| Welding Gloves | Hand protection |
| Welding helmets | Eye protection |
| Respirators | To mitigate inhalation of steel dust |
| MIG welder | For welding all the components together |
| Grinders with various heads | a steel brush head for cleaning surfaces to be welded and a carbide head for fabrication |
| Hydraulic metal Band saw | For cutting the pipe |
| Plasma Cutter | For cutting steel sheeting |
| C Clamps | For holding components while welding |
| Large roofing square | Useful for making sure things are perpendicular |

### 3.4 Attaching Arms

| Tool | Description |
|---|---|
| Rivet gun | For attaching cake pan receiver to arms |
| Band Saw | For making custom L brackets to attach cake pan to dish arms |
| Drill Press | For making custom L brackets to attach cake pan to dish arms |
| 5mm crescent wrench | For tightening arm to dish fasteners |
| Flexible tape measure | For spacing arms evenly around a dish |

### 3.5 Mounting Dish

| Tool | Description |
|---|---|
| Socket Wrench + socket set | For bolting the rotor to the mount and the dish to the rotor |
| Ladder | Somewhere to stand while tightening fasteners |
| Tap and die | For minor adjustments to mounting plate if customized holes are required |

# 4 Installing SRTN Software

The original SRT source can be found on the MIT Haystack Observatory website, here: http://www.haystack.mit.edu/edu/undergrad/srt/ . This source appears to work with ArchLinux and CentOS, and reportedly works on REHL (Red Hat Enterprise Linux). The source will not work immediately with Ubuntu, however a modified version of the source can be found here: https://github.com/BenningtonCS/Telescope-2014 , which will work on Ubuntu, as well as CentOS and ArchLinux (It may work with other linux-based operating systems, however, those listed are the only ones we have tested so far).

## 4.1 MIT Haystack Observatory SRT

The MIT Haystack Observatory SRT website includes the manuals, documentation, and program code. The code base we are looking at, SRT Source Code ver 3, can be found here.

## 4.2 Ubuntu

These instructions were developed as a guide from a clean download of source files from the MIT Haystack Observatory website. If using the code pulled from the repository, some of the steps (modifying the srtnmake script for example) may not be necessary, but the guide should still hold, overall.

Note that these instructions were written for Ubuntu 12.10

```
$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 12.10
Release:    12.10
Codename:   quantal
```

### 4.2.1 Getting the Source Code

To get the source code, either go to the website, linked above, and download SRT Source Code ver 3, or download directly from the link above. Once downloaded, the gzipped tarball should be in your Downloads directory:

```
~/Downloads/newsrtsource_ver3.tar.gz
```

*Move and Unzip File in New Directory*

To create a new directory, move the gzipped tarball into it, and then unzip it, follow the commands below:

```
~$ mkdir radio
~$ mv Downloads/newsrtsource_ver3.tar.gz radio
~$ cd radio
~/radio$ tar -xzvf newsrtsource_ver3.tar.gz
~/radio$ cd srtnver3
```

#### 4.2.2 Adding Library Dependencies

The reason the code would not compile is because SRT uses libraries which the computer you are using does not have. To get them, use the command below:

```
~/radio/srtnver3$ sudo apt-get install libgtk2.0-dev libusb-1.0-0-dev
```

NOTE: If you are unable to download using apt-get install, try using the command apt-get update first, wait for all updates to be installed, and then retry the install command above.

Modifying the Compile Script In order for the program to compile with gtk+-2.0, it must know that it has to use it/where to get it. (This was the problem that prevented it from compiling in class). To remedy this, simply /radio/srtnver3$ sudo nano srtnmake to edit the bash script. Scroll to the bottom of the file to the gcc line, which should look like: gcc -W -Wall -03 $CFLAGS $LIBS main.c vspectra_four.c disp.c plot.c cat.c geom.c time.c outfile.c sport.c map.c cmdfl.c cal.c srthelp.c velspec.c four.c librtlsdr.c tuner_r820t.c -lm ‘pkg-config –libs –cflags libusb-1.0‘ and change it so the end section is now ... ‘pkg-config –libs –cflags gtk+-2.0 libusb-1.0‘ Now you have all the libraries and the script should be able to compile successfully.

#### 4.2.3 Compiling the Source Code

The script to compile the code (the one you just edited) should take care of all the compilation, so it is just a matter of running it /radio/srtnver3$ ./srtnmake Once you execute this command, a few warnings should appear, with lines ending with [-Wunused-but-set-variable]. These errors are fine, and there should be relatively few of them as compared to before (running the compile script without the libraries or modification to the script). The output I got from running the compile script is shown below as a reference for which errors are acceptable.

Now the source code is successfully compiled!

#### 4.2.4 Running the Program

If you look in the srtnver3 directory, you will notice an executable file called srtn. This is what we just compiled with srtnmake. To run: /radio/srtnver3$ ./srtn Once executed, this should open a window which looks similar to the image below.

## 4.3 CentOS

These instructions were made using CentOS release 6.5 with kernel version:

```
$ uname -r
2.6.32-431.el6.x86_64
```

on a machine with an AMD processor.

### 4.3.1 MIT Haystack Observatory SRT

The MIT Haystack Observatory SRT website includes the manuals, documentation, and program code. The code base we are looking at, SRT Source Code ver 3, can be found here.

### 4.3.2 Getting the Source Code

From the website, linked above, or the download link provided, download the srtnver3 source files from the MIT Haystack website. Once the download is complete, the gzipped tarball should be in your Downloads directory.

```
$ cd Downloads
$ tar xzvf newsrtsource_ver4.tar.gz
If you want to move the source code out of the Downloads directory, and into the Home di
$ mv srtnver3 ~
$ cd ~
```

### 4.3.3 Getting project dependencies

From a clean install of CentOS, you will be missing some of the packages needed to run the software.

*Getting sudo permissions*

Before you do this though, you need to add your profile to the sudoers file, if you do not already have sudo permissions. To do this:

```
$ su
$ chmod +w /etc/sudoers
$ gedit /etc/sudoers
```

Down by the bottom of the file will be the line:

```
\#\# Allow root to run any commands anywhere
root    ALL=(ALL)    ALL
```

Below this add the line " ALL=(ALL) ALL", where is the username logged in to the computer, in our case 'radio telescope'

```
\#\# Allow root to run any commands anywhere
root    ALL=(ALL)    ALL
radiotelescope    ALL=(ALL)    ALL
```

Save the file and exit gedit, then in the terminal, type exit to return to your user status:

```
[root@localhost radiotelescope]\# exit
exit
[radiotelescope@localhost ~]\$
```

Now your user account should have sudo permission, so we can now install package dependencies.

*Getting dependencies*

The packages needed are gtk+2, gcc, and libusb. These can be acquired using the CentOS package manager, yum:

```
\$ sudo yum install gtk2-devel gcc libusb1-devel
```

If successful, you should now have all the packages you need to run the software.

#### 4.3.4 Compiling and running

Now, move into the srtnver3 directory $ cd /srtnver3 and run srtnmake: $ ./srtnmake This should complete successfully, but may show some warning messages. Now, to run the software: $ ./srtn

#### 4.3.5 Running with unsimulated dongle

To run with the dongle, receiver simulation must be turned off. To do this, simply $ gedit srt.cat In the edit window, find the line that says SIMULATE RECEIVER and change it to *SIMULATE RECEIVER Save the file and exit gedit. Now, we can run srtn using the dongle with $ sudo ./srtn And it should work! NOTE: Notice how before to run, it was just ./srtn and this time it is sudo ./srtn. If you try./srtn after enabling the dongle you will get a message similar to: $ ./srtn Found 1 device(s) 0: , SN: ????? Using device 0: ezcap USB 2.0 DVB-T/DAB/FM dongle usb_open error -3 Please fix the device permissions, e.g. by installing the udev rules file rtl-sdr.rules Failed to open rtlsdr device #0. As we see above, it is possible to run srtn without sudo, but to do so, we must do a little extra work in installing the provided udev rules. A guide to installing the udev rules will be added later.

### 4.3.6 Running with unsimulated controller/rotator

To run with the controller/rotator, antenna simulation must be turned off. To do this, simply $ gedit srt.cat In the edit window, find the line that says SIMULATE ANTENNA and change it to *SIMULATE ANTENNA Save the file and exit gedit. Now, we can run srtn using the controller/rotator with $ sudo ./srtn To test that the rotator works, try manually changing the az or el in-program using the 'azel' button. (Note: If the the rotator motion is moving in a direction you do not want it to go, you can press the 'S' button on the controller to stop movement.) If you are unable to run the program successfully, look at the controller setup page to troubleshoot and make sure the controller was set up correctly.

## 4.4 Arch

The MIT Haystack Observatory SRT website includes the manuals, documentation, and program code. The code base we are looking at, SRT Source Code ver 3, can be found here.

### 4.4.1 Getting the Source Code

From the website, linked above, or the download link provided, download the srtnver3 source files from the MIT Haystack website. Once the download is complete, the gzipped tarball should be in your Downloads directory.

```
$ cd Downloads
$ tar xzvf newsrtsource_ver3.tar.gz
```

### 4.4.2 Getting Project Dependencies

From a clean install of Archbang, there are a few necessary things missing, namely make andpatch. From the command line, run:

```
$ sudo pacman -S make patch
```

### 4.4.3 Install gcc-4.4

Unfortunately, the program will not run properly using a current version of gcc, so it is necessary to download and install version 4.4. As gcc-4.4 is not in pacman, gcc-4.4 must be downloaded from the AUR. You can find gcc-4.4 in the Arch User Repository here Download the tarball. It should appear in your Downloads folder. Untar it by using:

```
$ cd ~/Downloads
$ tar xzvf gcc44.tar.gz
```

and move into the new folder that was created.

```
$ cd gcc44
```

Use the command makepkg by itself to make the package. NOTE: It might be necessary to make the package as root. If this is the case, run:

```
$ sudo makepkg --asroot
```

As gcc is a large program, it might take a very long time to download. Once it's downloaded, run:

```
$ sudo pacman -U gcc44-4.4.7-6-x86_64.pkg.tar.xz
```

Once again, as gcc is a large program, this will take a long time to complete.

#### 4.4.4 Compile and Run Source

Move into the srtnver3 folder that you downloaded and untared from MIT. Assuming it's in your Downloads folder, use: $ cd Downloads/srtnver3

Edit srtnmake To make the makefile use gcc-4.4 instead of the default version, you will have to edit the make file. $ nano srtnmake Change gcc to gcc-4.4 Save and exit out of nano.

Compile srtnmake Run: $ ./srtnmake If everything is done right, the program should compile with few errors.

Running srtn To run the program: $ ./srtn Running with unsimulated dongle NOTE: It might be necessary to blacklist the driver in order to run properly. Please look at 'Blacklist Driver' under the troubleshooting section. To run with the dongle, receiver simulation must be turned off. To do this, simply $ nano srt.cat In the edit window, find the line that says SIMULATE RECEIVER and change it to *SIMULATE RECEIVER Save the file and exit nano. Now, we can run srtn using the dongle with $ sudo ./srtn And it should work! NOTE: Notice how before to run, it was just ./srtn and this time it is sudo ./srtn. If you try./srtn after enabling the dongle you will get a message similar to: $ ./srtn Found 1 device(s) 0: , SN: ????? Using device 0: ezcap USB 2.0 DVB-T/DAB/FM dongle usb_open error -3 Please fix the device permissions, e.g. by installing the udev rules file rtl-sdr.rules Failed to open rtlsdr device #0. As we see above, it is possible to run srtn without sudo, but to do so, we must do a little extra work in installing the provided udev rules. A guide to installing the udev rules will be added later.

#### 4.4.5 Running with unsimulated controller/rotator

To run with the controller/rotator, antenna simulation must be turned off. To do this, simply $ nano srt.cat In the edit window, find the line that says SIMULATE ANTENNA and change it to *SIMULATE ANTENNA Save the file and exit nano. Now, we can run srtn using the controller/rotator with $ sudo ./srtn To test that the rotator works, try manually changing the az or el in-program using the 'azel' button. (Note: If the the

rotator motion is moving in a direction you do not want it to go, you can press the 'S' button on the controller to stop movement.) If you are unable to run the program successfully, look at the controller setup page to troubleshoot and make sure the controller was set up correctly.

### 4.4.6 Troubleshooting

Blacklist driver It may be necessary to blacklist the driver so it will not already be in use when we try to call on it: $ sudo -i # echo "blacklist dvb_usb_rtl28xxu" ¿ /etc/modprobe.d/librtlsdr-blacklist.conf This will require a reboot before the effects can be seen. $ sudo reboot

# 5 RAS SPID Rotor

## 5.1 Setup

## 5.2 Wiring

The document that came with the rotator can be found here for additional information on the rotator and wiring it.

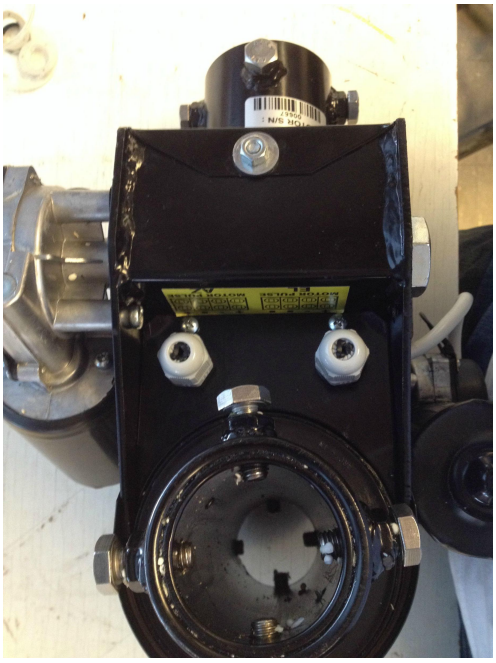
### 5.2.1 Tools Needed

- Screwdriver (Phillips)
- Utility Knife
- Adjustable Wrench
- Small Screwdrivers (Phillips & Flathead)
- Soldering Iron
- Solder
- Electrical Tape
- Wire Strippers

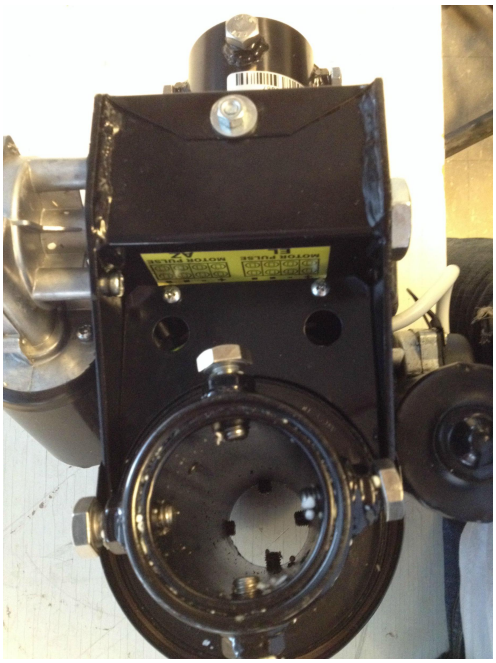
### 5.2.2 Instructions

The rotator we used was an Alpha Spid RAS rotor (http://www.spid.alpha.pl/english/11.php). These instructions are a guide to accessing the power/control terminals to the rotator in order to control it with our rotor controller (http://www.spid.alpha.pl/english/05.php).

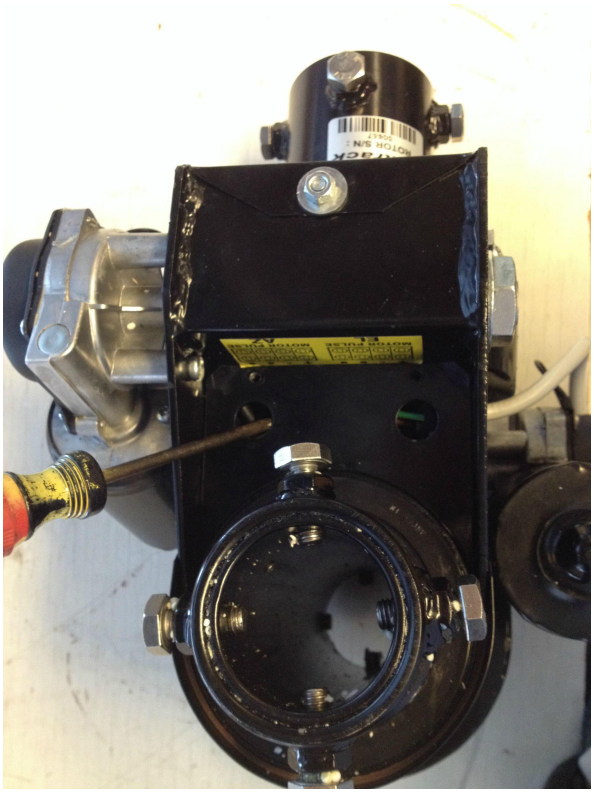
On the rotor, there is a plate with two screws and two waterproofed wire outputs (see pictures below). On the face next to the plate should be a sticker detailing the EL AZ wiring.



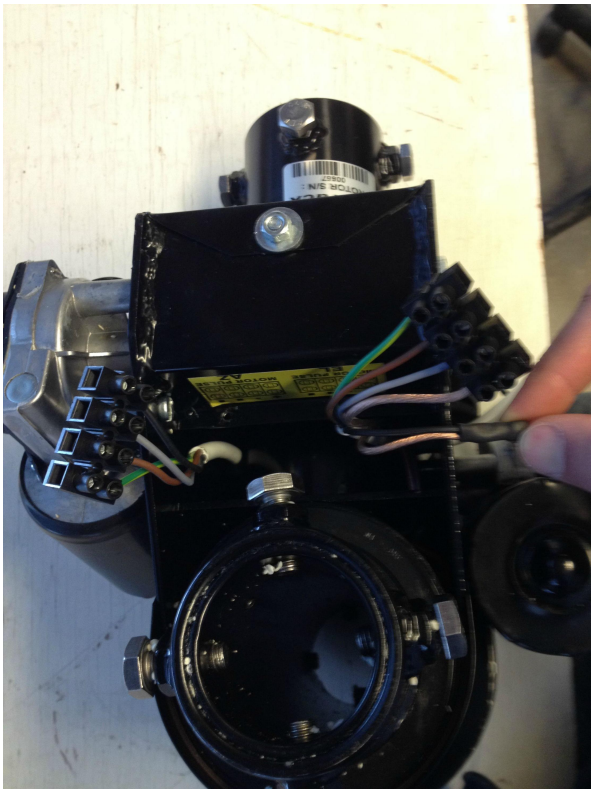
First, unscrew/remove the two waterproof outlets using the adjustable wrench and unscrew the two screws on the plate.



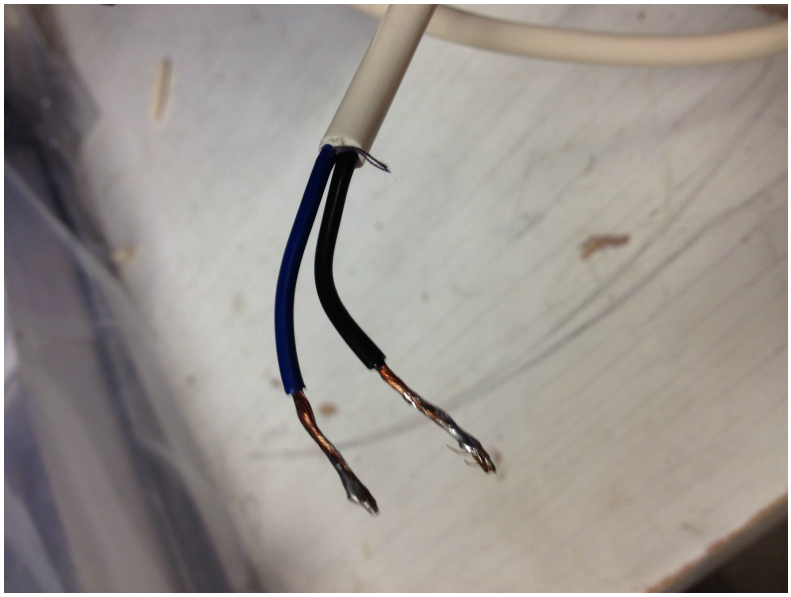
Note that the edge of the plate may be painted over, so use a utility knife to edge the plate. The first time you remove the plate, it may be difficult, so, using the back of a screwdriver, tap around the corners of the plate to loosen it.
Using a screwdriver (or some other means of leverage), insert one end into a hole left from unscrewing the waterproof outlet, and carefully pry off the plate. This may take some time since the plate has a tight fit initially.



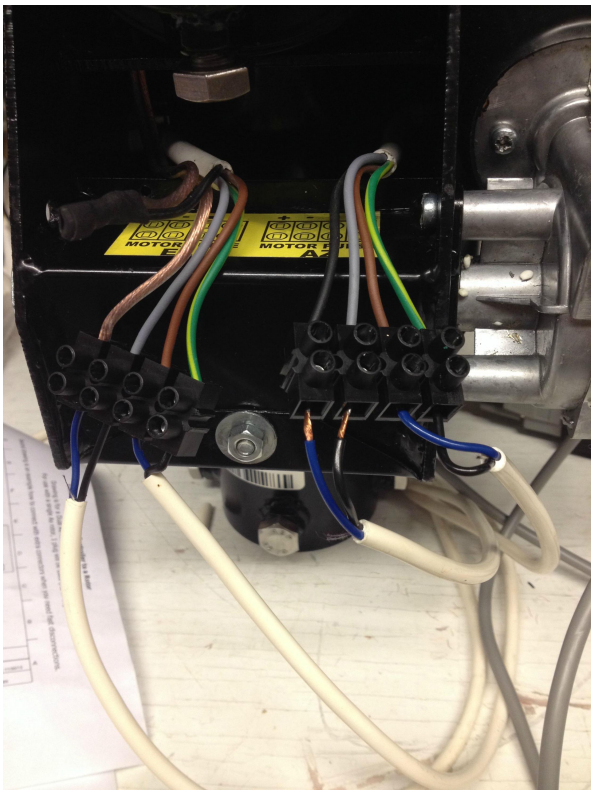
Underneath the plate are two wiring terminals, one for AZ and one for EL. (See the sticker on the rotor to determine which terminal corresponds to AZ and which corresponds to EL).



For testing purposes, a set of test cables were made, approx. 75 cm long. (Later, these wires were replaced with more permanent ones to fit our mounting strategy). Our test cables consisted of four cables, each cable containing two wires, so there were two cables per terminal, or four wires per terminal. The ends of the wires were stripped and tinned.



The ends of the wires were secured in their respective rotator terminal.



The free end was then soldered to a head which connects to the controller. When doing this, note the order of the wires. On the head, the pins are labeled 1-4, which

will correspond sequentially to the wires coming from the terminal. Electrical tape was wrapped around the wire near the connection to ensure each pin and wire were isolated from one another. Stripping a small amount of wire on this end may diminish the need to wrap the connection in electrical tape.

The head was re-assembled and connected to the controller.

Once everything was secure and connected, the power supply and controller were turned on, with the mouse attached to the controller.


Adjusting the the controller to be in 'A' (automatic mode), we were able to control the rotor with the mouse.

## 5.3 Controller Settings

Full documentation can be found:
( http://www.spid.alpha.pl/download/AlfaSpid_Rotator_RAS_EN.doc ).
Much of this documentation was pulled directly from the source linked above.

### 5.3.1 RAS SPID rot2 Settings

In our setup, we are using the RAS SPID rot2 Rotator. We want to have our settings such that PS (program simulation) is configured to SP. If the mode is not SP, the controller will not work. With the rotor we are using (1 degree per pulse accuracy), we also want our rotor transmission, P, to be set to 1.0. To have the computer software or mouse control the rotor, the mode will need to be set to A (automatic). For more detailed information, see the "Controller Setup" section, below.

### 5.3.2 Controller Reset

Since there are no mechanical limits in the rotor, it may be installed with the antenna pointing in any direction. There is no reason to locate "TRUE NORTH" until you are ready to calibrate the control box. Use the controller to position the antenna to physically point north, then reset the controller as follows:

- Turn the unit OFF.
- While holding the F button down, turn control unit back on.

- The display should then read 0.0 0.0

This feature can be used if, for any reason, the direction of the antenna becomes incorrect. This may be caused by antenna to mast slippage or incorrect initial alignment.

**Important Note:**

The SPID rotator is now set at the counter-clockwise end of its normal rotation range. Normal rotation range is in a clockwise direction for 360 degrees.

From the reset position, you can rotate counter-clockwise an additional 180 degrees in over-travel, as well 360 degrees clockwise, plus an additional 180 degrees into clockwise over-travel.

Counter-clockwise over-travel is indicated by a steady dot above the over-travel icon [<->]. [<->] Rotation past 359 degrees into the clockwise over-travel is indicated by a blinking dot above the over-travel icon.[<->]

**Technical Note:**

You will need to leave sufficient coax length to accommodate the additional 180 degrees of over-travel on each end of normal rotation. Failure to do so can cause damage to your coax and/or antennas.

### 5.3.3 Controller Operation

There are multiple modes of operation for the SPID controller. The two main modes, F and S are described below, along with all of their sub-modes.

### 5.3.4 F - Function Mode

The F button steps through the function menus. The leftmost character on the display indicates the function mode you are currently in.

**(no letter) = Normal Operations Mode**

In Normal Operations Mode, the up, down, left, right buttons cause rotation as long as the buttons are pressed. Pressing S while in normal operations mode will take you to setup mode.

**H = Half-Auto Mode**

In Half Auto Mode, the up, down, left, right buttons can be used to pre-select the desired beam heading. The heading displayed on the controller will rapidly change in the direction of desired rotation. Once the desired beam heading is shown on the display, release the key. Approximately $\frac{1}{2}$ of a second after no key presses have been detected,

the display will revert back to the actual beam heading, and rotation towards the desired heading will take place. Pressing any key while in transit to the desired heading will cancel the action.

**A = Auto Mode**

In Auto Mode, the controller will respond to commands from control software running on an attached computer. The up, down, left, right buttons can still be used, but pressing of any of them will cause canceling the data from software.

#### 5.3.5 S - Setup Mode

The S button steps through the setup menu.

**P = Rotor Transmission**

This value defines the accuracy of rotator operation. 1.0 means operating with up to 1 degree per pulse from rotator accuracy.

**PS = Program Simulation**

Program Simulation allows the user to set the serial communication protocol used by the rotator. When set to emulate another brand of rotator, the Spid will respond to commands, and send responses back to the computer as if it were the rotator brand selected. If your favorite software supports a rotator, chances are, the Spid will be able to interface to your software. There are 2 modes available:

- PS SP = SPID
- PS 4A = Yaesu (GS232 protocol)

(RS232: 600N1, 8 bits)

(data rate bound 600, 1 STOP bit, no even parity bit)

Operating mode can be changed using left, right.

**PH = Programmable High Limit**

The Programmable High Limit is a user adjustable clockwise travel limit value. By reducing this value, the maximum clockwise rotation travel can be restricted. Use the buttons:

- *left* and *right* to adjust the azimuth value,
- *up* and *down* to adjust the elevation value.

**PL = Programmable Low Limit**

The Programmable Low Limit is a user adjustable counter-clockwise travel limit value. By increasing this value, the minimum counter-clockwise rotation travel can be restricted. Use the buttons:

- *left* and *right* to adjust the azimuth value,
- *up* and *down* to adjust the elevation value.

**PP = Heading Adjust**

This setting can be used to make minor heading adjustments without causing the rotator to turn. If you notice that the heading displayed on the controller to a known signal source is out by a few degrees, you can change the heading displayed on the LED readout to match the known heading, rather than having to turn back to North and reset the controller. These settings are made by up, down, left, right buttons.

## 5.4 Controller Setup

The antenna controller will likely need a bit of setup in order to work properly with the software. To properly set up the controller, three things must be done:

1. Make sure the controller has no limits set
2. Set the controller to SPID mode
3. Set the controller to Automatic mode so it is able to communicate with a computer via serial

### 5.4.1 Setting Limits

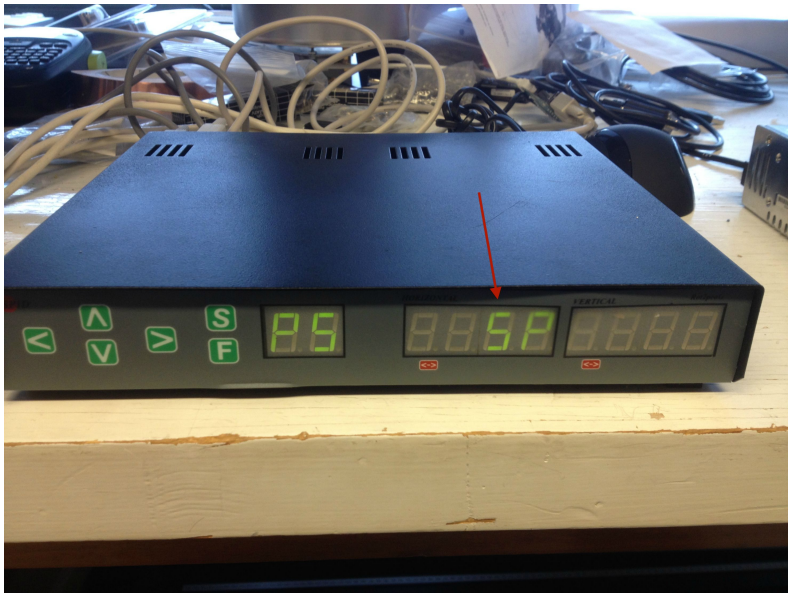
The easiest way to be sure there are no limits set on the controller is by doing a controller reset. To do this, turn off the controller, hold down the 'F' button, and while still holding down the button turn the controller on again. When this happens the display should look like this:


To change the remaining settings, we will use the buttons on the controller, designated by the red arrows. (F, S, *left*, *right*, *up*, *down*)

If there are limits set, it is possible that when the software is run, a buffer overflow may occur, and that the GUI logic breaks, leading the program to be ineffective. If no radio data is being displayed on the UI and large values/NaN appear on the readout, having limits set is a potential cause.

### 5.4.2 Setting to SPID

In the controller's program simulation setting, two options are available, SPID or Yaesu. Since the controller we used is SPID, the controller should be set to SPID. To set, press the 'S' button until the readout says PS. Then, if the readout does not say PS SP, press the $<$ or $>$ arrows to change it to SP. It should look like:



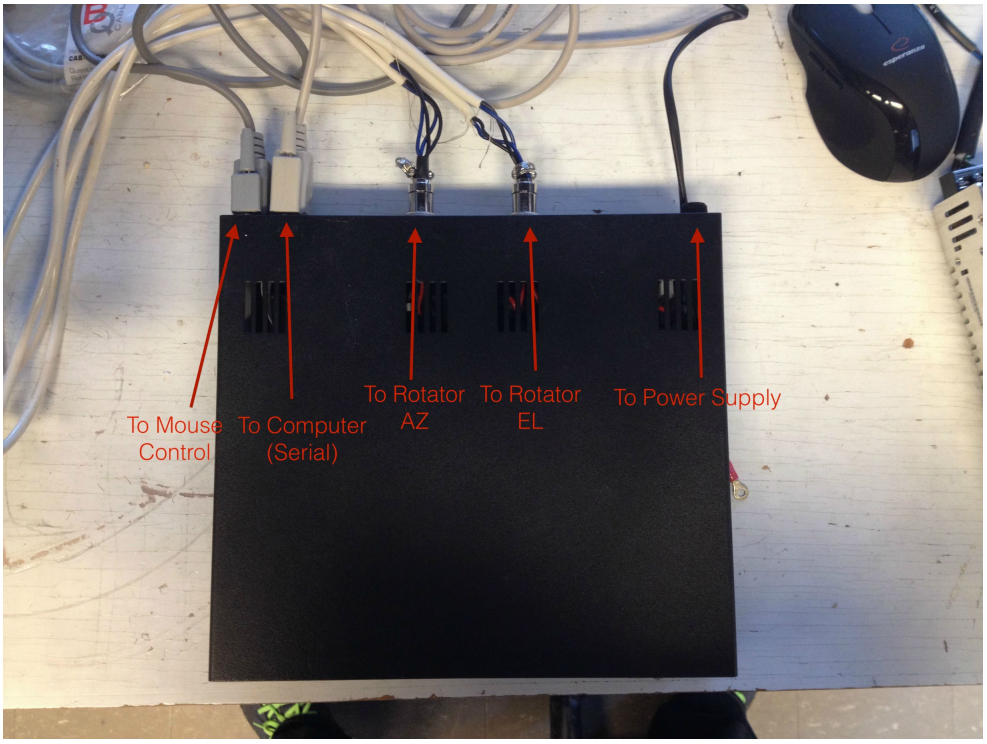
If the controller is not in SPID mode, nothing will happen and the program may hang, as it is unable to communicate with the controller.

### 5.4.3 Set to Automatic

In order for the computer to be able to communicate with the controller, it must be in automatic mode. To do this, you would press the 'F' button until an A shows up on the readout. The readout should look like:

### 5.4.4 Controller Wiring

While the controller does not have many/complicated connections, and the documentation is fairly good for what connects where, below is a picture detailing our connection setup, for reference.



# 6 Building Components

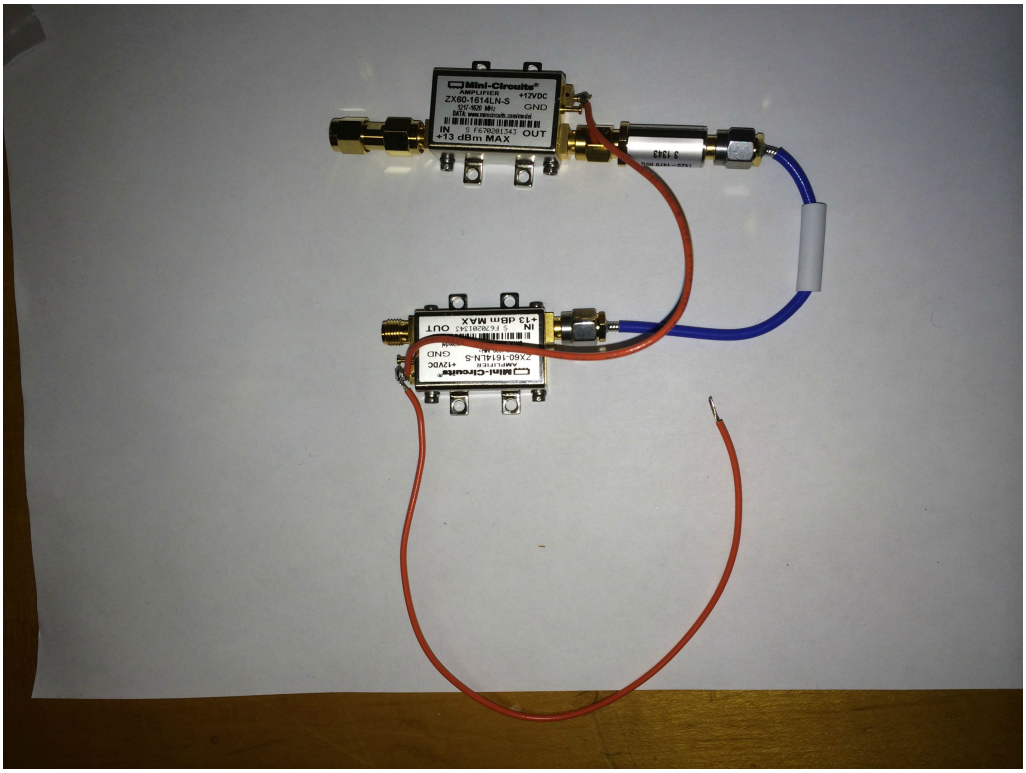
## 6.1 LNA

Attached the coupler to the "In" side of Amp 1. Then attached the bandpass filter to the "Out" side of Amp 1.

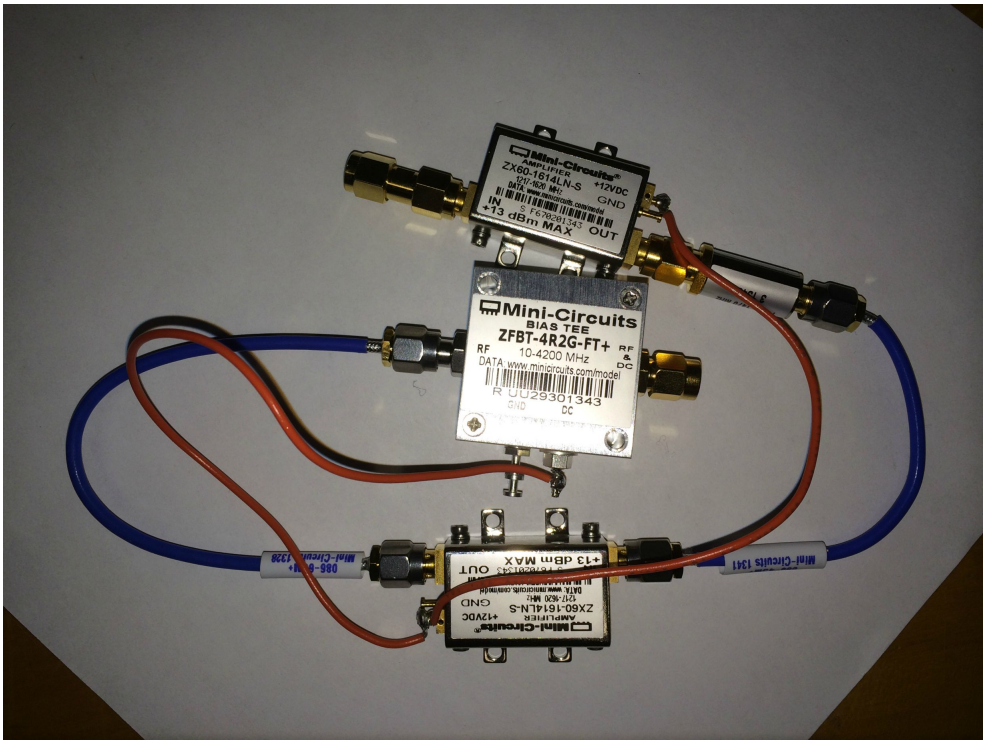

Connected one coaxial cable to bandpass.


Connected opposite end of coaxial cable to "In" side of Amp 2.


Cut two lengths of 15-20 centimeter long 22 gauge copper wire. Soldered one wire to both Amps. Soldered the other wire to Amp 2.



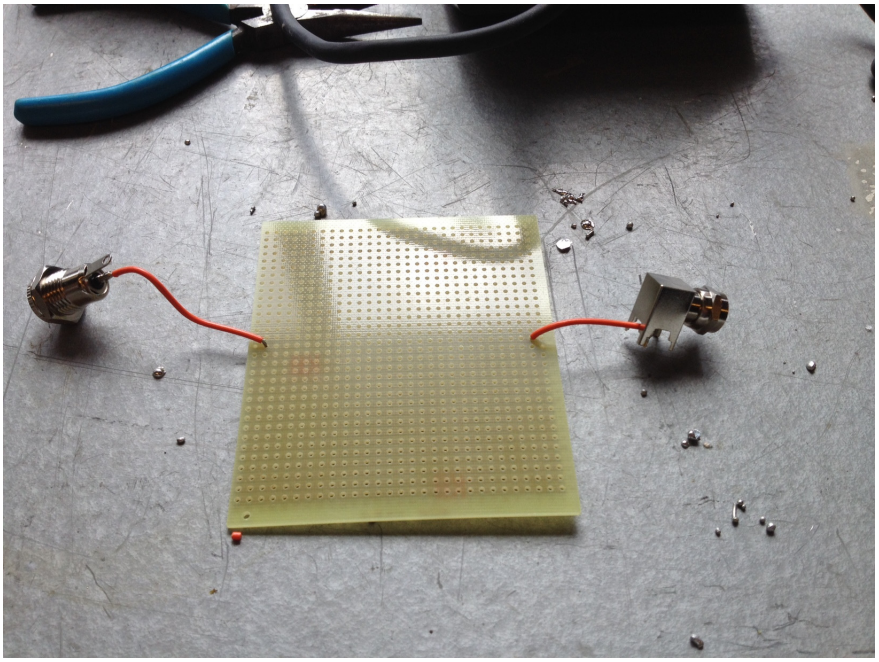

Soldered the other end of the second wire to the 12 volt pin on the bias tee. Connected "RF" side of the bias tee and "out" side of Amp 2 with coaxial cable.



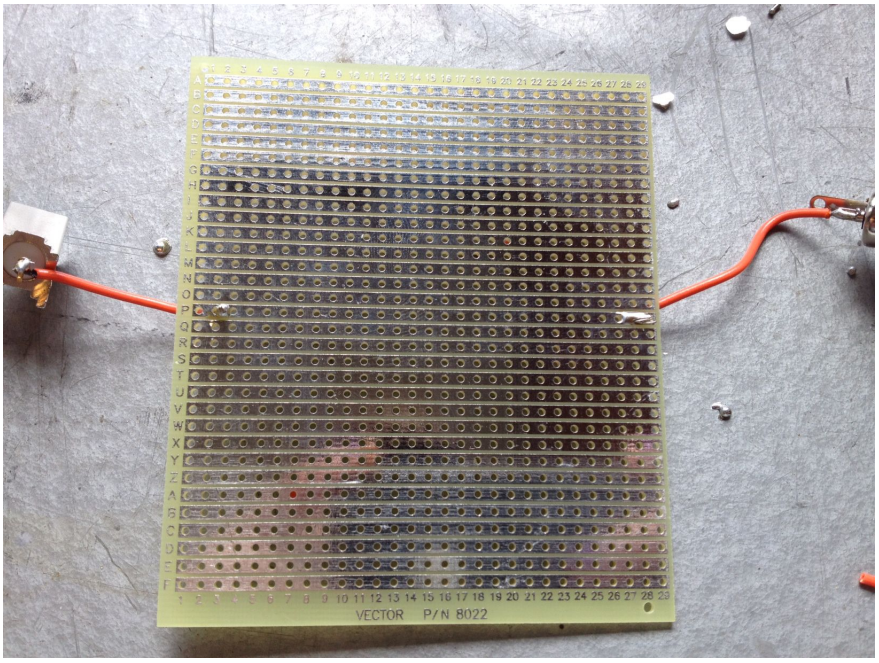

Placed this group into the case, with coupler and "RF DC" side of the bias tee protruding from the holes.


Placed on rubber lining, and case cover, wrote LNA on top, and screwed into place.

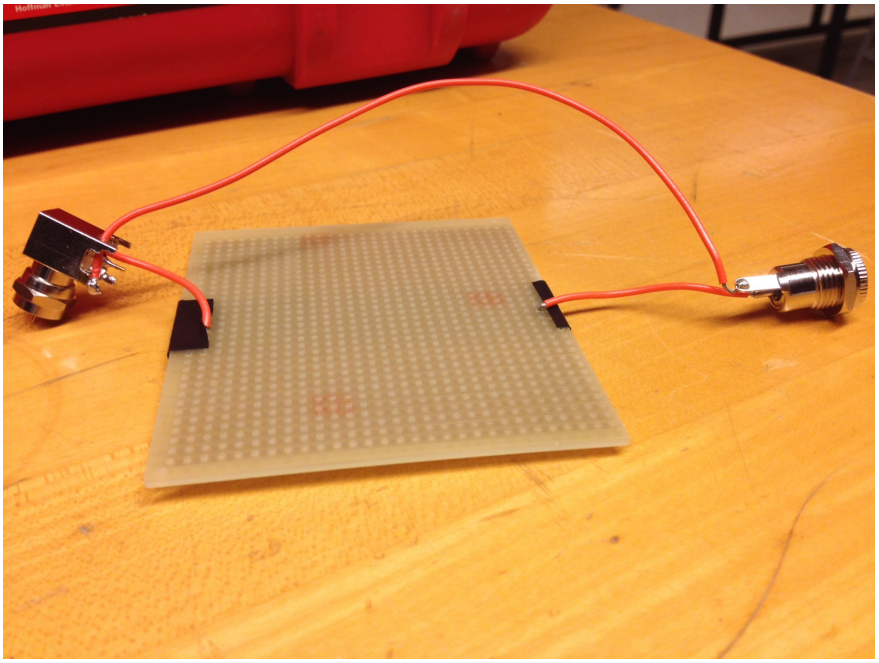

Pics of assembly of dc jack connector to f type coax adapter. Solder middle pin of dc plug connector to perf board. Solder middle pin of f type plug connector to same conductive strip of perf board.



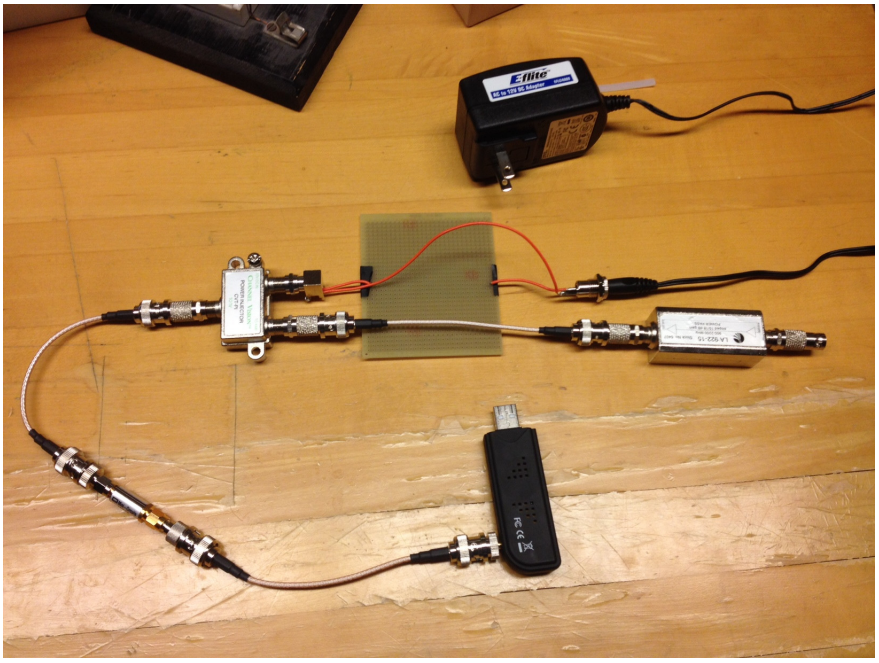
Under side.




Solder ground pin on dc jack connector directly to one of the ground pins on the f type plug connector.



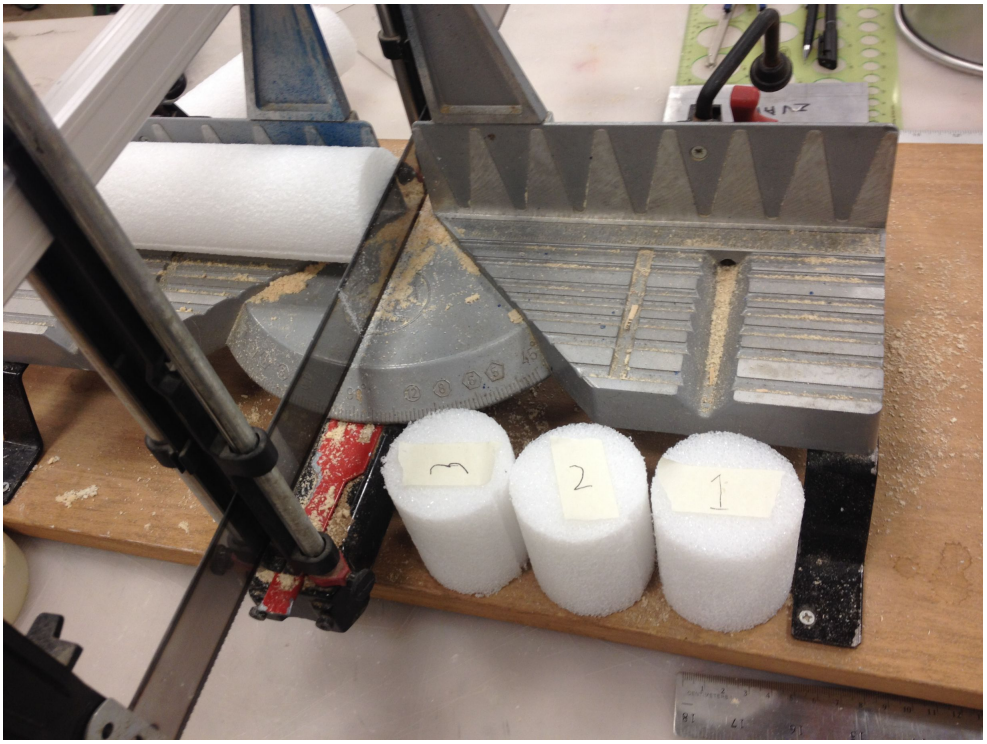
Back End b/t Dongle and LNA Put Together




## 6.2 Feed

### 6.2.1 Cutting Styrofoam Rod

We used a hack saw in a jig to get perfectly straight cuts in the 2.5” Styrofoam rod.



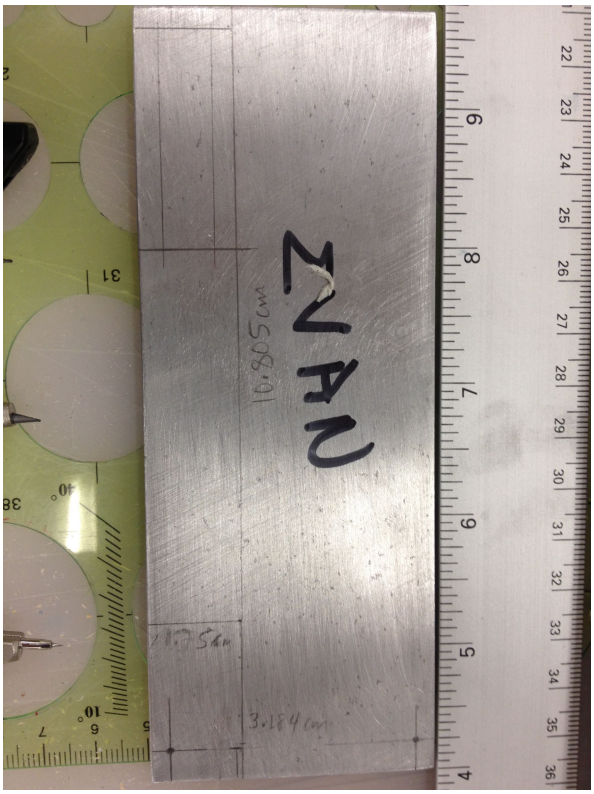

The Styrofoam was measured from multiple sides to determine the center. The rod we have ranges from 2 5/16" to 2 1/2" in diameter.

We used a drill press with a 1/4" bit. The Styrofoam was held by a vice on the drilling table. (The Styrofoam is a bit messy so we put down a rag under the vice)

Note: we made three rods to provide room for error in the application and soldering of the copper tape later.

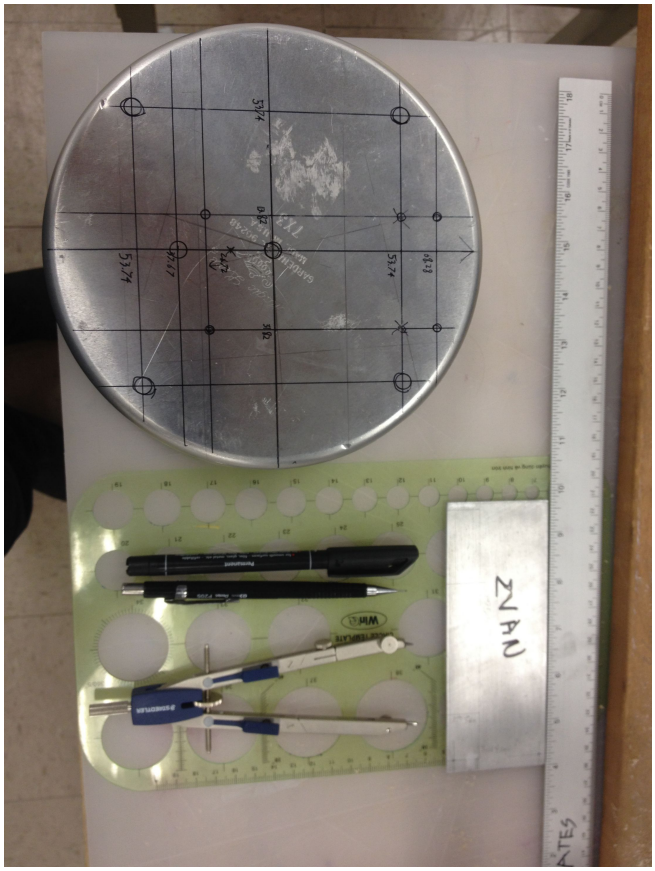
### 6.2.2 Drilling Aluminum LNA Base Plate

We put the plate in a vice which we secured on the drill press table with some heavy steel blocks.



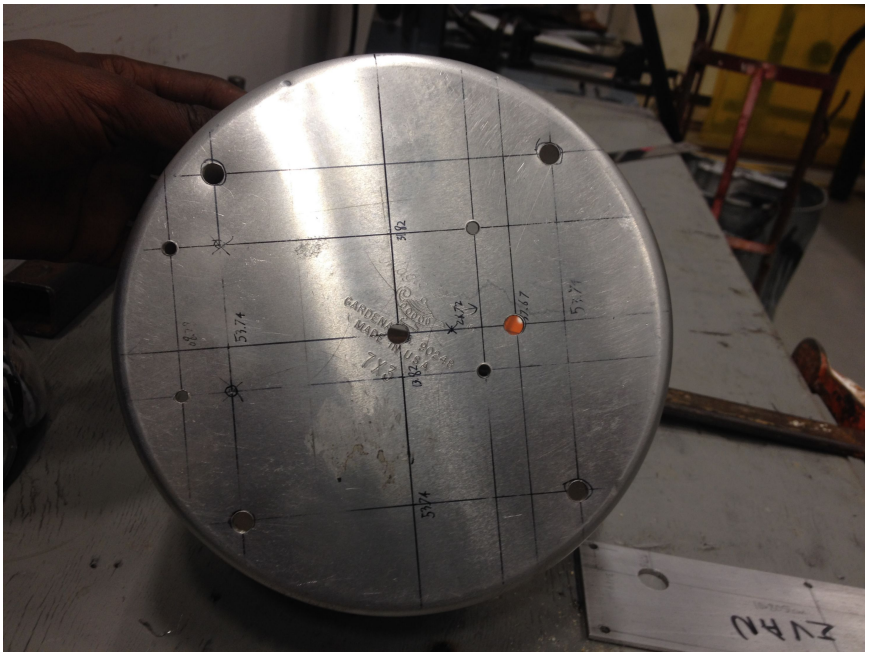

### 6.2.3 Drilling Aluminum Cake Pan

We made the measurements of the cake pan working out from the center which we found by transecting the circle and then making perpendicular rays from the center of each line cutting across the arc. This proved to be adequately precise.



The cake pan was more of a challenge to drill than the base plate as it was tricky to secure to the drill press table so we could make accurate holes. We ended up putting a 4" piece of angle iron on the pan and then clamping that down to the edges of the table.


Finished drilling

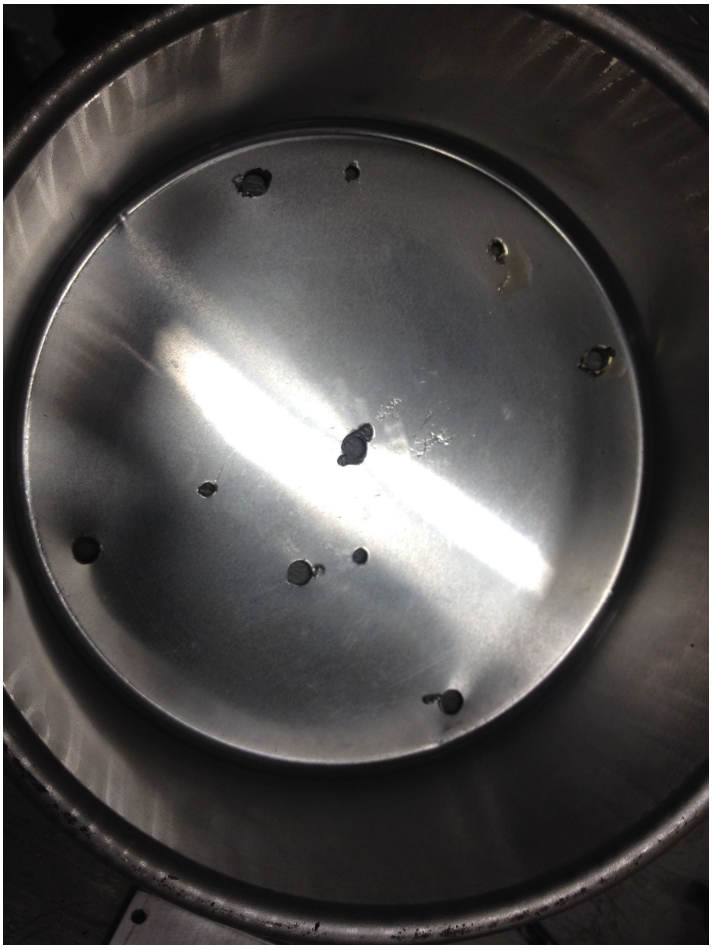


### 6.2.4 Minor concerns

The drilling instructions are precise to a 100th of a millimeter. We did the drilling with a drill press, not a CNC machine, so I would say our precision was closer to a 10th of a millimeter.

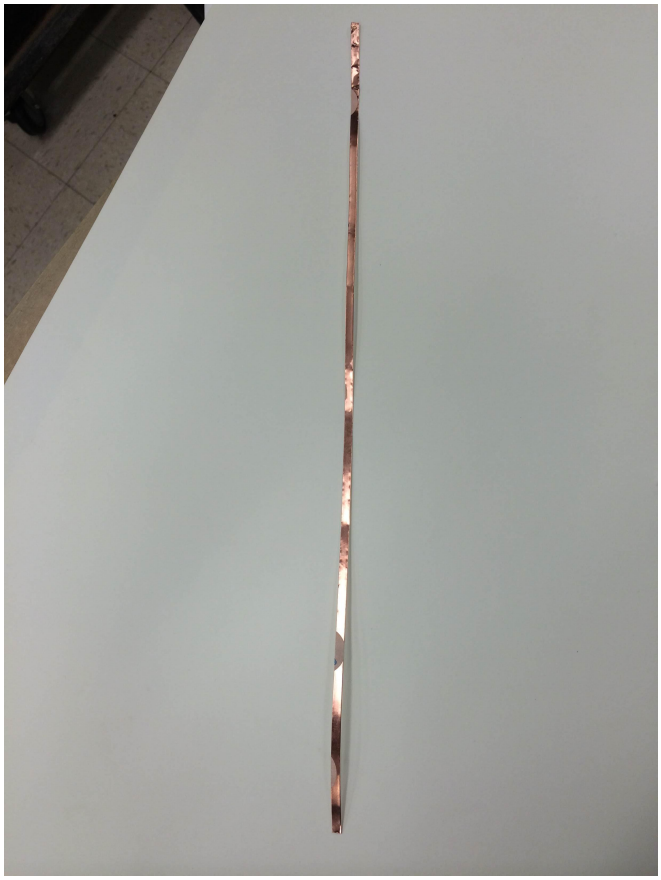
Also, the cake pan is really not a precisely made object, it is certainly not as precise as the schematics given by MIT. I think that indicates that our slight reduction in precision will be fine.

The cake pan is made of what seems to be a softer aluminum than the LNA base plate. This made the metal not cut cleanly on the inside of the pan (the bit went through the outside first). The holes are the correct dimensions but there is some extra material around the holes. I have tried to clean it up with an x-acto knife and been relatively successful. I think the metal is so soft that these small bits of extra material will be squished when we insert the hardware, making their effect negligible.



### 6.2.5 Making helical antenna

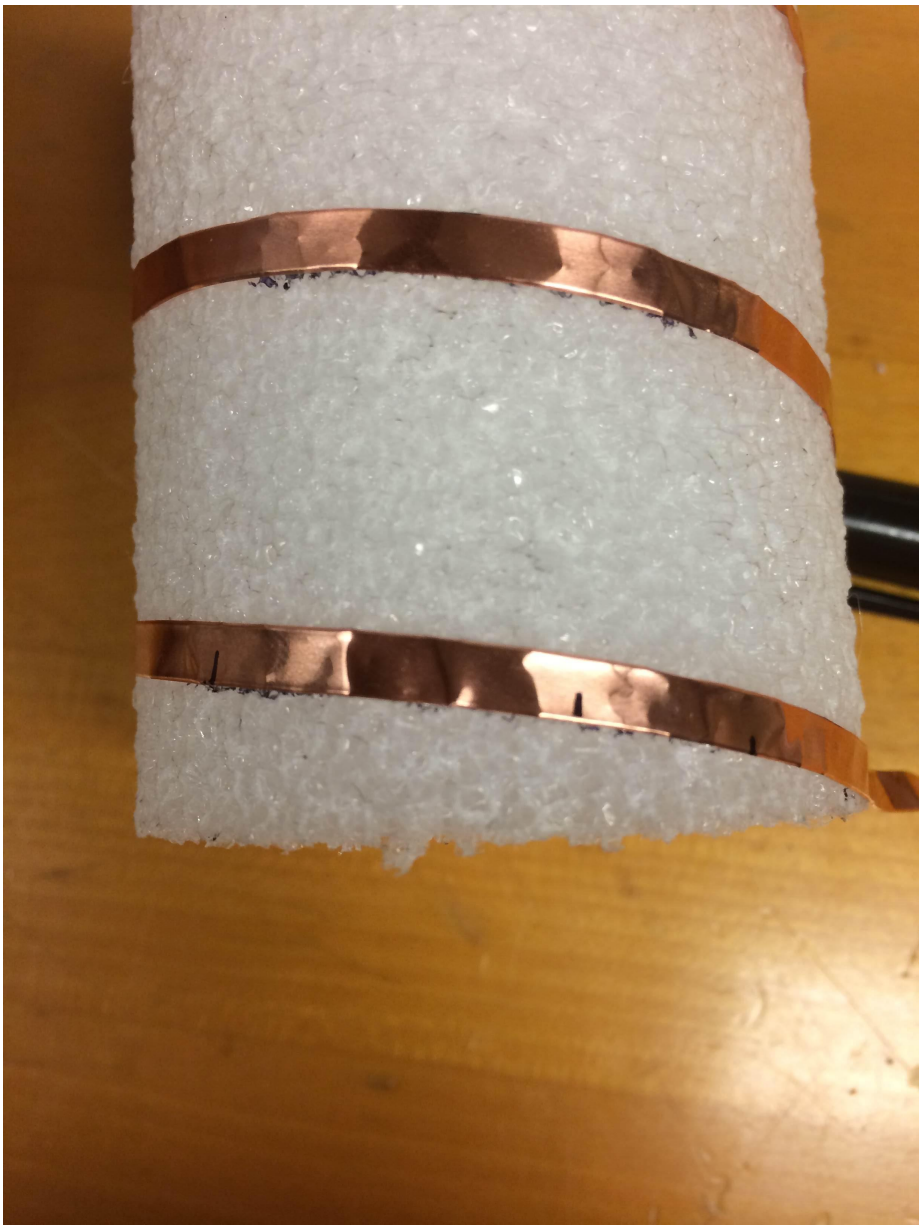
We cut out strips of copper tape with an X-acto knife and a straight edge.

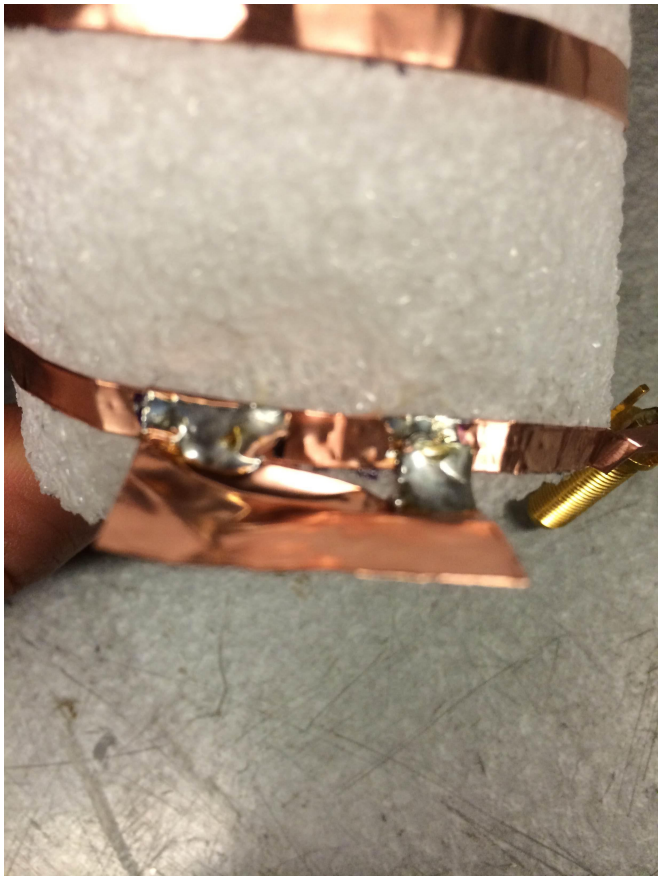


Then we measured marked the styrofoam rod at even intervals as specified by the manual to get a consistent helix with the tape.

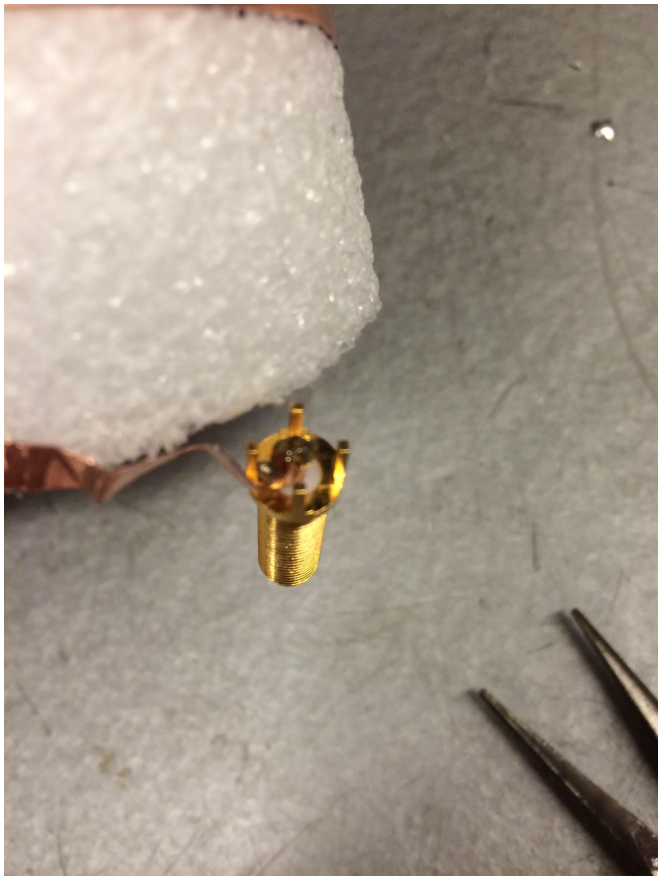
Once the tape was applied we soldered on 26.25mmx13mm piece of copper tape for impedance matching section. Note: It is important to include an extra 3-4mm on the long side of the rectangle so that it can be attached to the helix.

In the process we melted some of the rod. This should be ok as long as the tape stays in a helix.



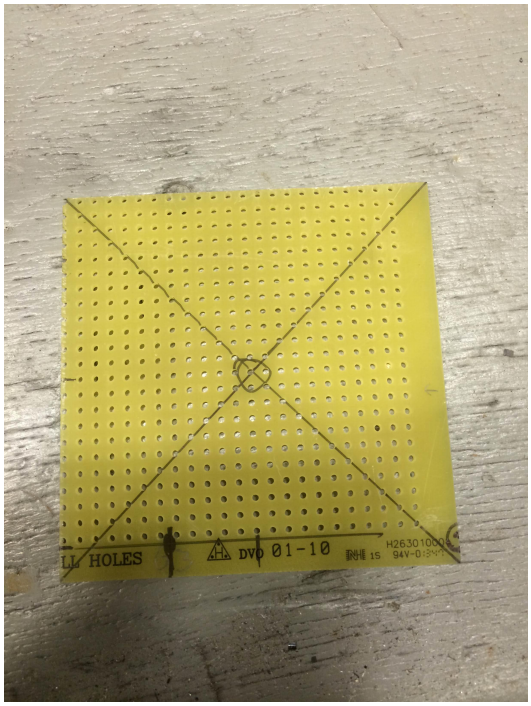


The we soldered the end of the helix to the SMA connector. MIT cut off some of the ground pins to make that easier. We left them on and it worked out the same.

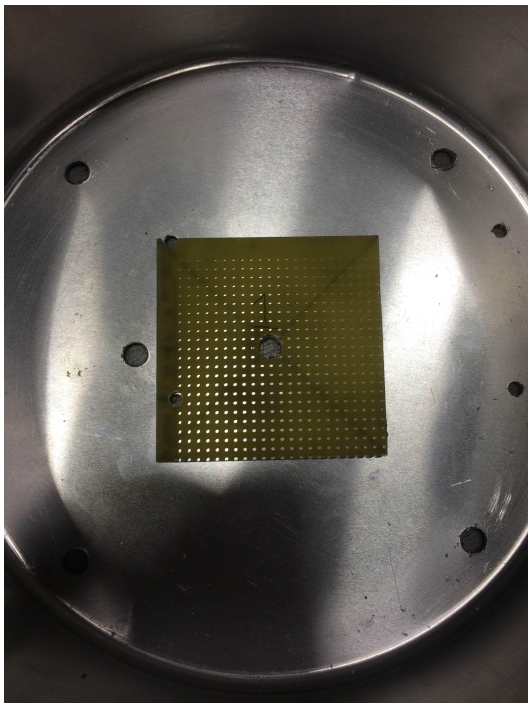


### 6.2.6 Drilling/cutting PC board

We cut the PC board into 63mm squares (We had enough PC board to make a couple back ups).

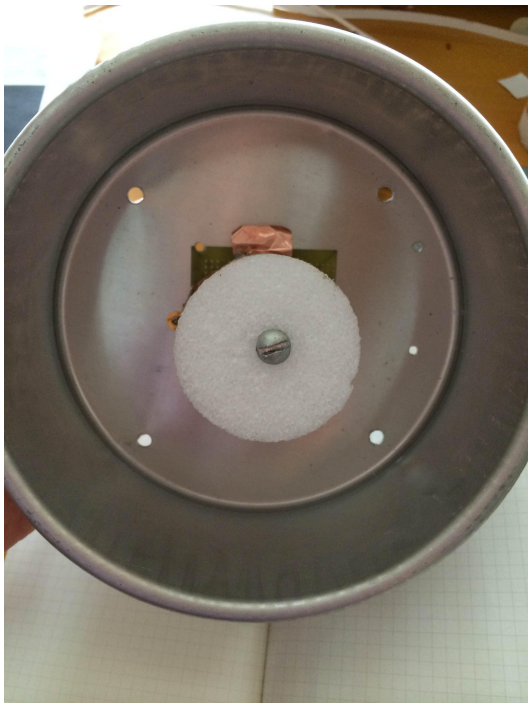


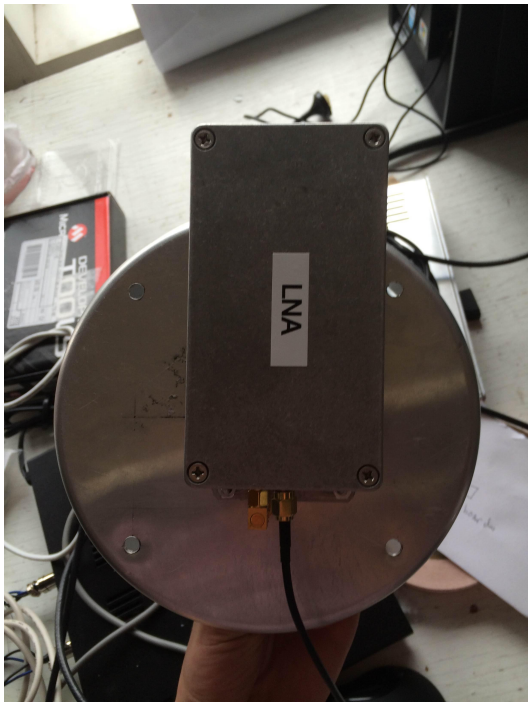

Then we drilled holes in it as specified by the manual using a drill press. It was relatively easy to secure the board in a vice for drilling.

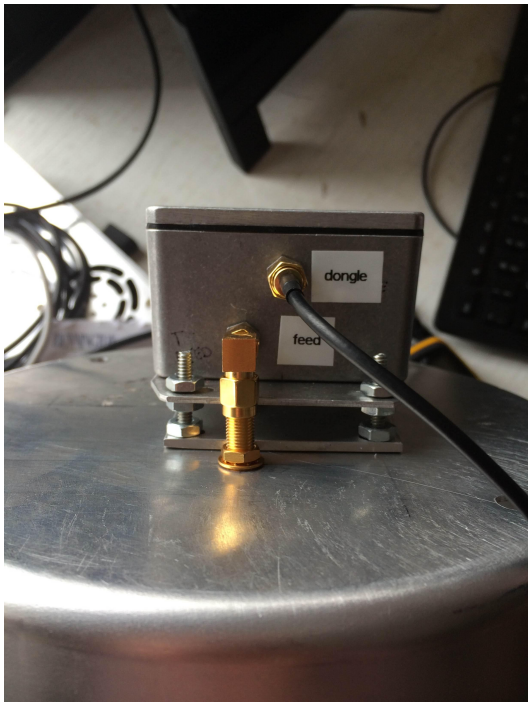


### 6.2.7 Assembly

All the components were then bolted together.




LNA


dongle
feed

## 6.3 Roof Mount

We used scrap metal to make our dish mount so we had to start by cleaning up the pieces of metal we were going to use and make them the right size.

We plasma cut a 1.5' x 3' rectangle of 5 gauge steel (about 3/16" thick, it later turned out this is to thin) and cleaned up the edges with a grinder.

We cut an 11.5’ length of 2.5” (exterior diameter) pipe and cleaned in using a steel brush head on a grinder.

We then MIG welded the pipe to the plate. We reinforced the weld with a couple extra beads because most of the force in the system will be directed to this point.

The real trick was getting the pipe perpendicular to the plate.

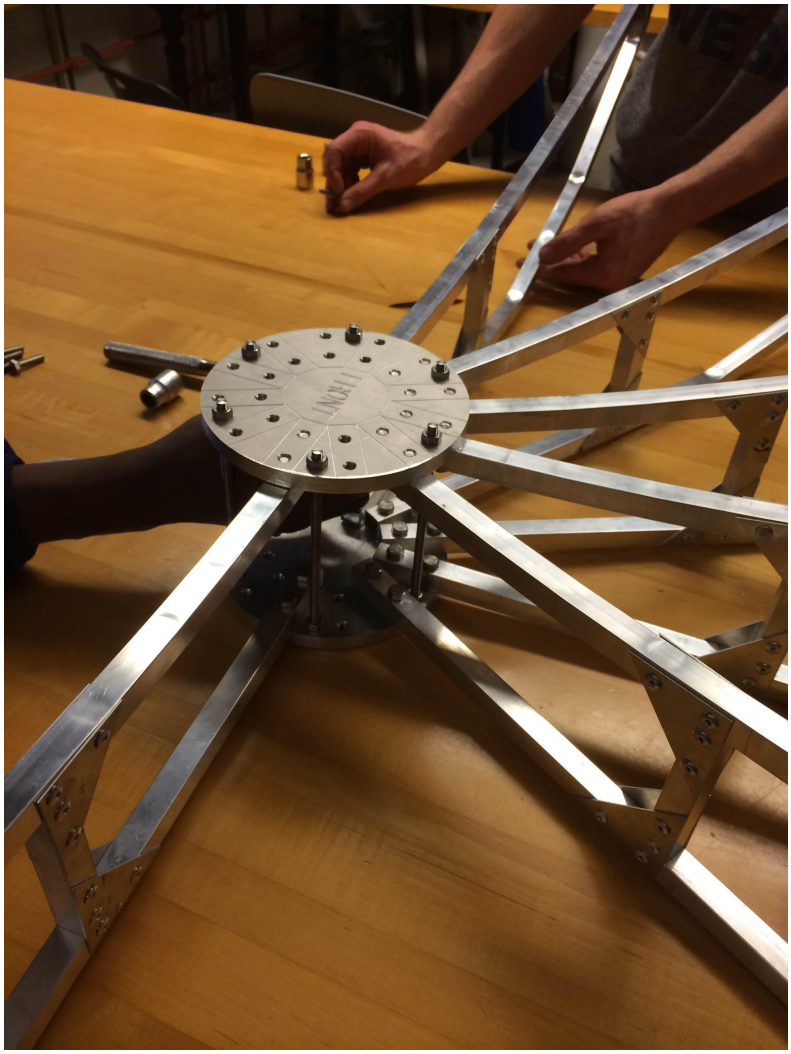
We ended up clamping the plate on one side of the table and then using the corner of the table as a perpendicular straight edge in two dimensions.

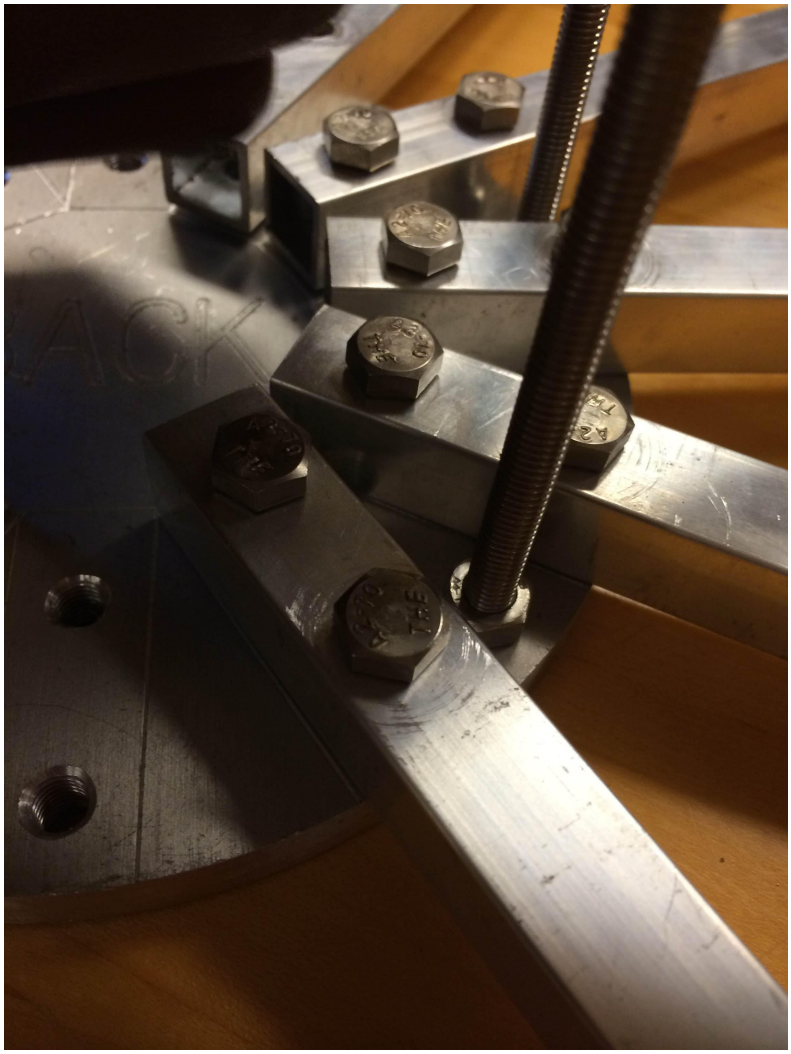
After some tests we realized the weight of the pipe was warping the plate so we successfully reinforced it with some angle iron.

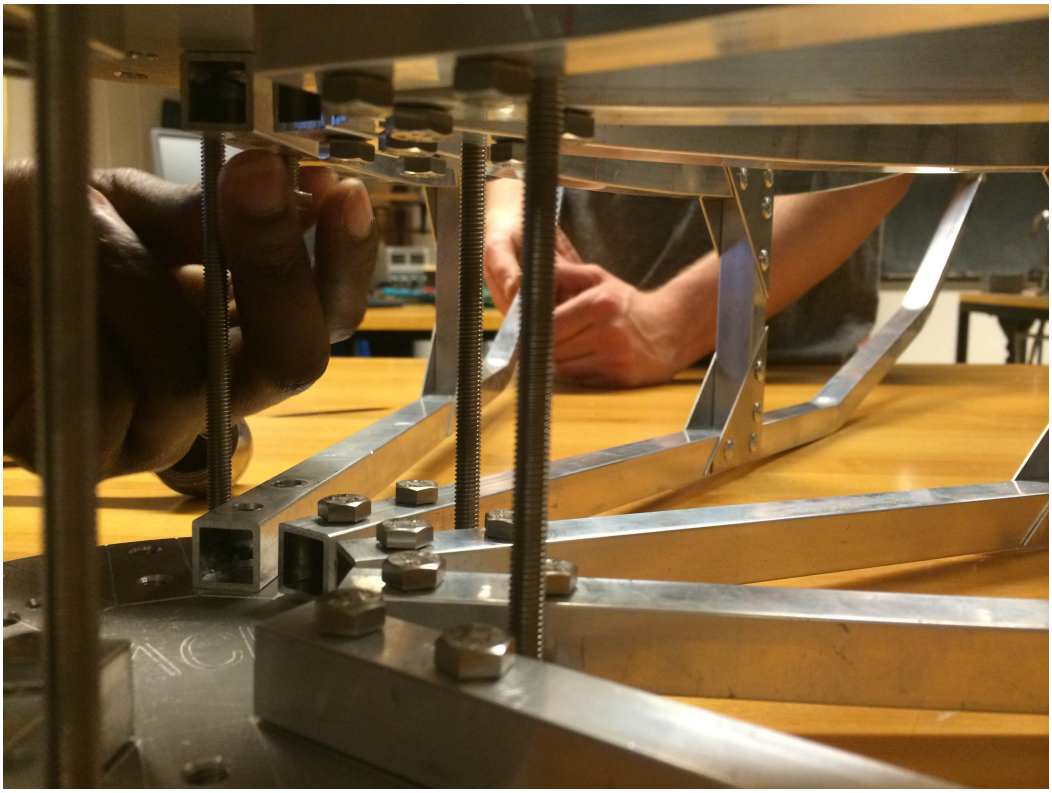
## 6.4 Dish

We bolted together the central plates, trying to make sure they are parallel to each other.

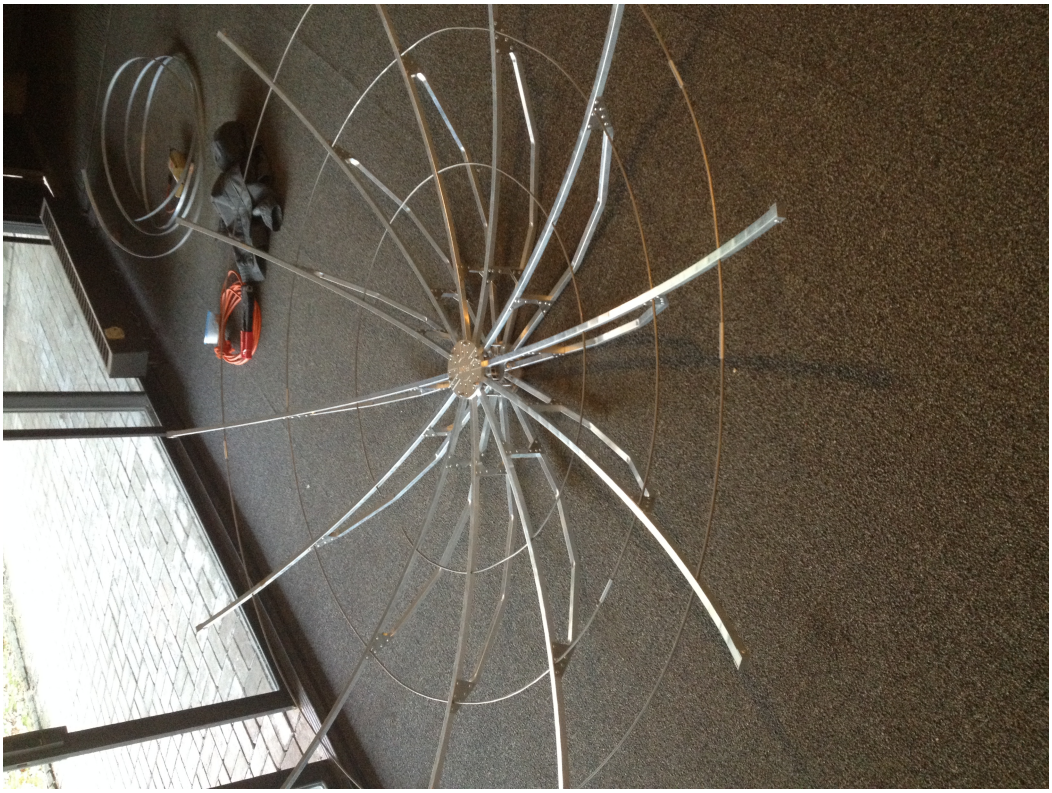
Then we bolted on half of the dish arms which confirmed the accuracy of our central plate construction. This is as much of the dish as will fit out the door to the roof we will be mounting the dish on. The rest of the assembly happened on site.



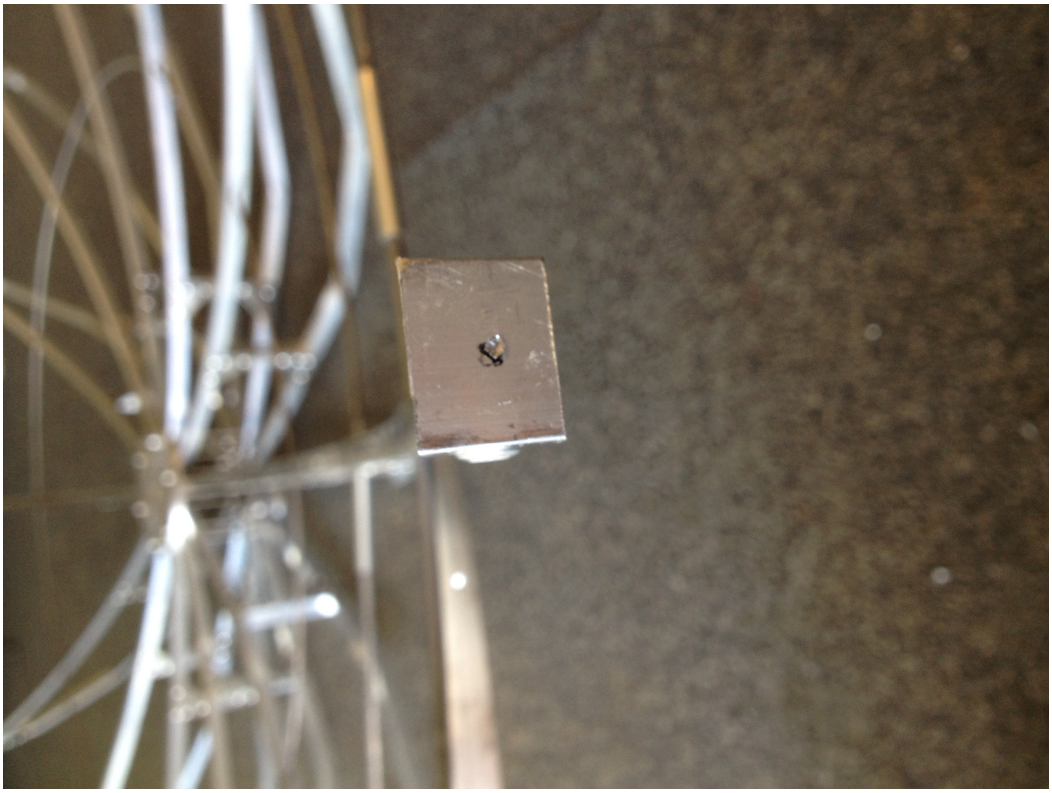




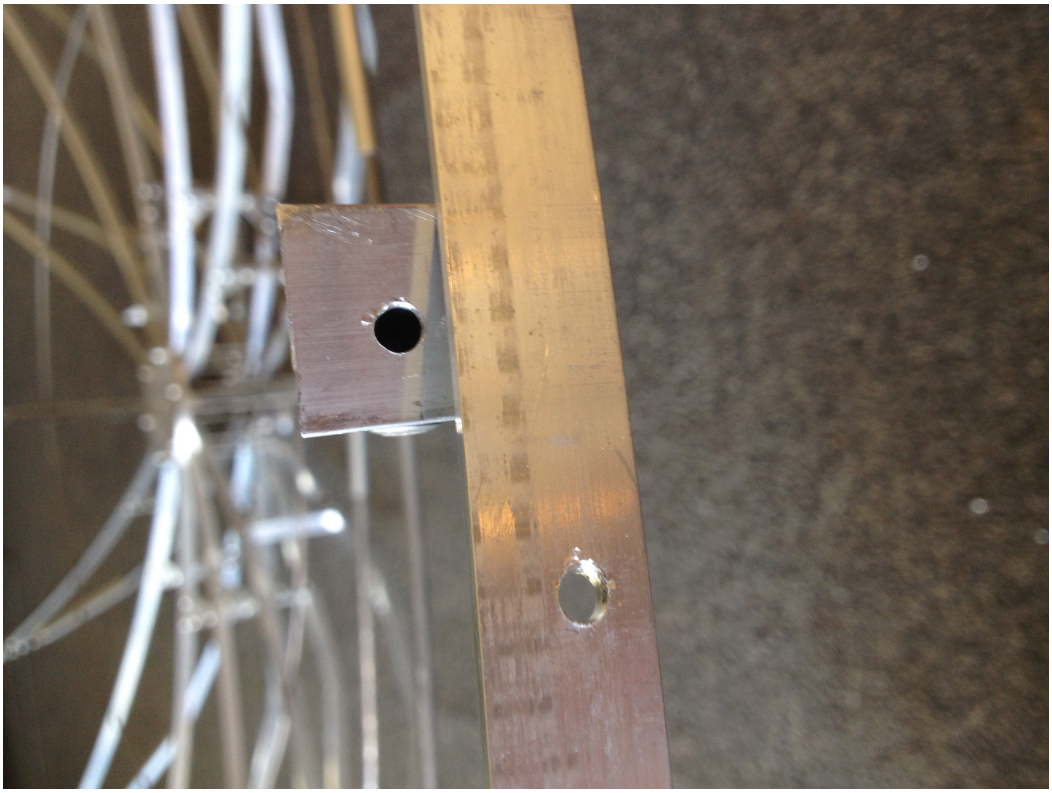
Once the ribs are put together and the support rings are in place, we are ready to put on the outer band.



To do this, a hammer and tap were used to make dents in the end of the rib and the band



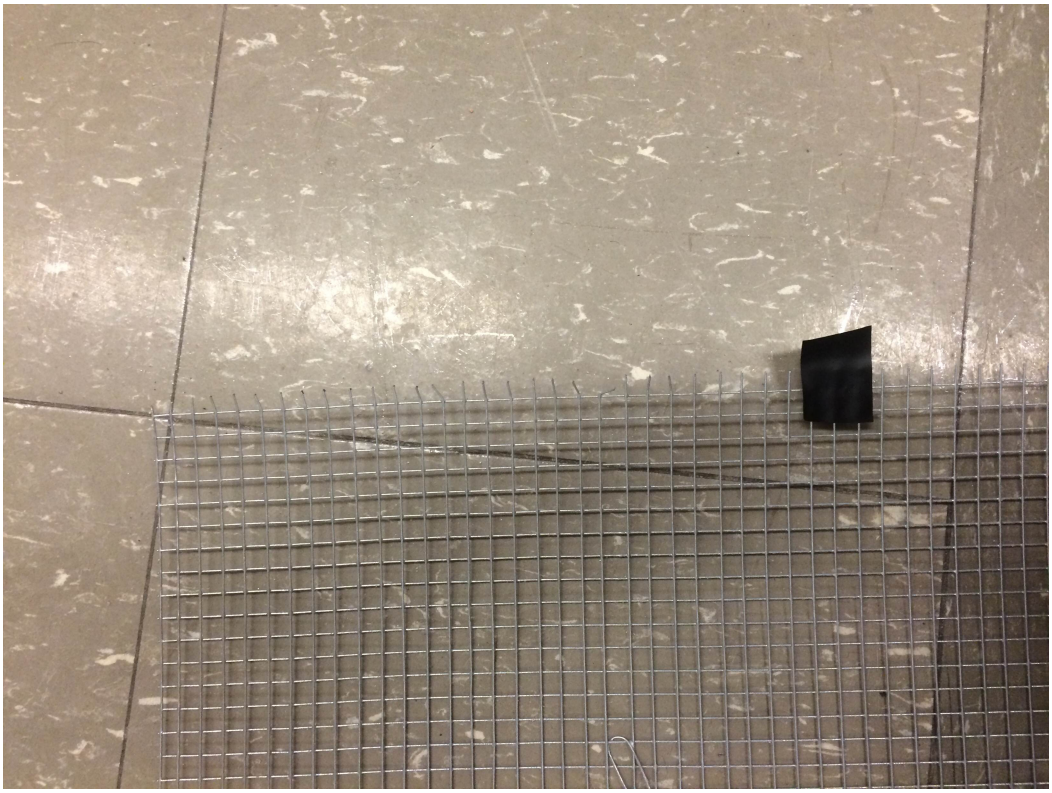
Then, using a power drill, holes were cut in both so they could be lined up and riveted.

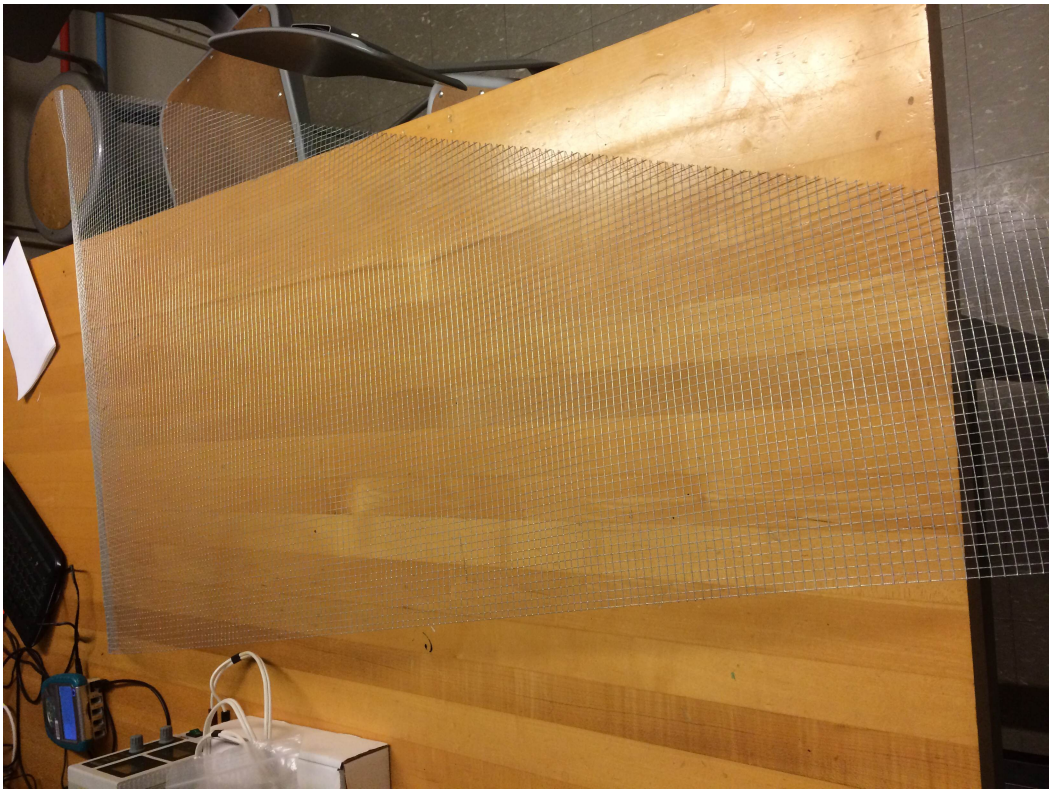


This process of tapping, drilling, and riveting was repeated for each of the 12 ribs of the dish. Once complete, we were ready to start putting the mesh on.

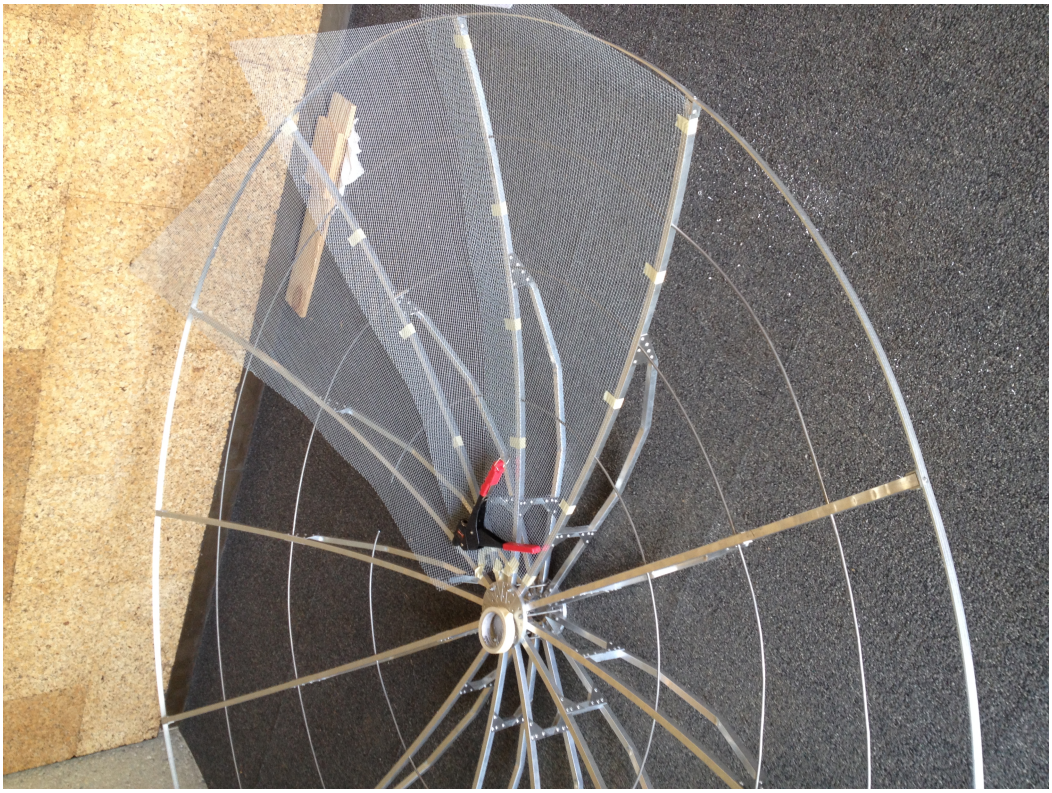
Make sure all the ends of the support rings are together. We did not crimp them, since they seemed secure enough so that they would not fall out.

Cutting the mesh!
Below are some pictures that detail the mesh cutting process.





After the Mesh was cut to specifications, it was laid out on the skeleton of the dish, to make sure there was enough material and it would come together properly



After we were sure that we had enough material all cut to the right size to allow for some extra, all the sheets were removed except for one. One end of the sheet was secured to a rib using tape, while the other end was trimmed down to be flush with the next rib

Then, the next sheet is placed on top of the end of the first sheet so they are overlapping, and a strip with pre-drilled holes (which came with the kit) were lined up on top of the rib, and holes were drilled through the rib at the places where the holes in the strip were

The strip, with the overlapping pieces of mesh underneath, was then riveted to the rib.

This process was repeated for all 12 sections of the dish until it was complete.

Then, the edges were trimmed, leaving about 1/4 to 1/2 inch mesh past the outer band (enough to wrap around the band a bit)

It was then wrapped around the band

And secured using zipties ( 4 zipties per section * 12 sections = 48 zipties)