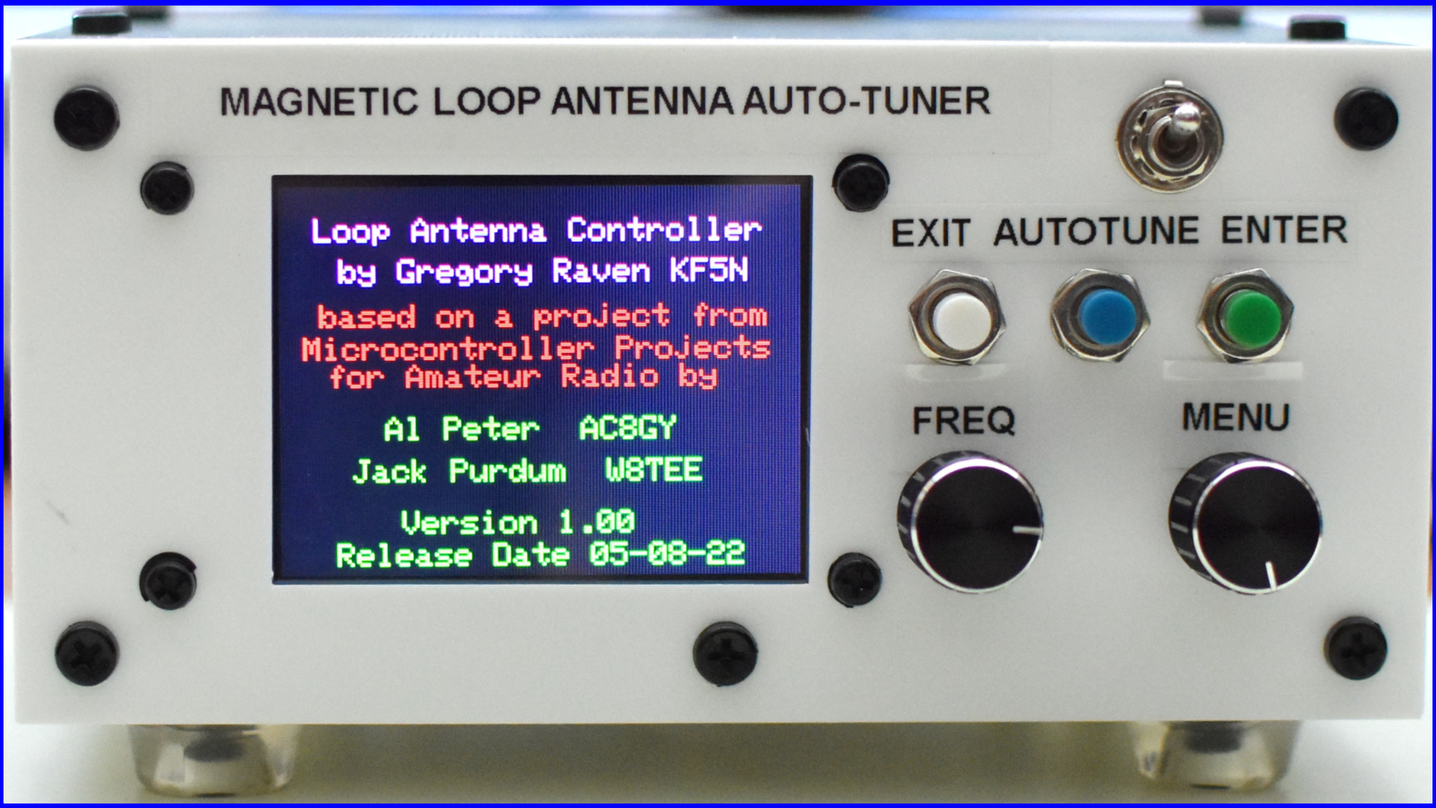
HIGH FREQUENCY MAGNETIC LOOP ANTENNA AUTO-TUNER FOR AMATEUR RADIO



GREGORY RAVEN
KF5N

High Frequency Magnetic Loop Antenna Auto-Tuner for Amateur Radio

Gregory Raven KF5N

May 2022

Contents

1 Introduction **1**

2 Controller Hardware **3**
2.1 Microcontroller 5
2.2 DDS Module, RF Amplifier and SWR Bridge 6
2.3 Stepper Motor Driver 7
2.4 External RF Relay 8
2.5 ILI3941 Display 8
2.6 Butterfly Capacitor 9
2.7 Audio Muting 9

3 Controller Schematic **11**

4 Printed Circuit Board **13**
4.1 Printed Circuit Board 13

5 Firmware **15**
5.1 Organization of the Code 15
5.2 Data Class 16
5.3 EEPROM Class 16
5.4 ILI9341 Driver 16
5.5 Stepper Motor Library 17
5.6 Rotary Encoder Library 17
5.7 De-bouncing 17
5.8 Limit Switch Protection 17

6 Loop Antenna Construction **19**
6.1 3D Printed Parts 21
6.1.1 Left and Right Loop Supports 21
6.1.2 Top Support 22
6.1.3 N Connector Brackets 22
6.1.4 Coupling Loop and BNC Connector Bracket 23
6.1.5 Tuner Assembly Clamps 25
6.1.6 Limit Switch Bracket 27
6.1.7 Ethernet Connector Bracket 27
6.1.8 Stepper Motor Bracket 28

6.1.9 Pulleys and Belt Drive 29
6.1.10 Plexiglas Tuner Assembly Base 30
6.2 Comments on Threads in 3D Printed Parts 31
6.3 Butterfly Capacitor 31
6.4 Capacitor Protective Enclosure 32

7 Using Ethernet Cable for Stepper Motor Connection 35

8 Controller Enclosure 37

9 User Interface 41

10 Speaker Audio Muting 43

11 Construction 45
11.1 PCB Assembly 45
11.2 Ribbon Cables 46
11.3 JST Cables 46
11.4 Power and Fuse Wiring 47

Appendix 49
11.5 Github Repositories 49

Chapter 1

Introduction

This is the construction manual and ”user guide” for a magnetic loop antenna controller which is patterned after a design published in the book ”Microcontroller Projects for Amateur Radio” by Jack Purdum W8TEE and Albert Peter AC8GY. The author’s firmware for the antenna controller was developed in the Arduino IDE. The book includes a detailed information and a detailed construction guide for both the antenna and the controller. The book is available here (click on the link):

Microcontroller Projects for Amateur Radio

This derivative design replaces the STM32 ”Blue Pill” controller module with a Raspberry Pi Pico. I noticed the Pi Pico was very similar in size to the STM32 board, and had the necessary GPIOs and ADC (Analog to Digital Converter) to perform the same function as the Blue Pill board. Another reason to use the Pi Pico is that it has not been affected by “supply chain” problems which have plagued other electronic endeavors. The official vendors are listed in the “Buy now” button, however, the boards can also be purchased at Amazon and eBay.

`https://www.raspberrypi.com/products/raspberry-pi-pico/`

The firmware was developed in the Visual Studio Code text editor which is provisioned by a script provided by Raspberry Pi. This script installs and configures everything required to develop firmware for the Pi Pico. The C/C++ “Software Development Kit” (SDK) provided by Raspberry Pi has a wealth of functionality to control the peripheral components of the Pi Pico. The documentation for the Pi Pico is available here:

Raspberry Pi Pico Documentation

One of the prime reasons to chose the Pi Pico is that the development kit includes a “debugger”, which is a feature the Arduino IDE lacks. However, this may change as the Arduino project migrates to the 2.0 IDE.

As the original project was developed as an Arduino project, a lot of “hacking” of the code was required to adapt it to the Raspberry Pi SDK. The first experiment was to hack the display library since this was the most complex library in the project.

After a few days of experimenting with both the library code and Cmake, the hacked display library was demonstrated functioning! The other libraries were much easier to translate to the Pi Pico SDK.

In the future, a version of firmware may be provided which uses the Arduino IDE. It should be possible, however, right now I can't support it.

Although this project follows the same pattern as the original, there are other significant hardware changes:

- Replaced large external stepper driver with A4988 stepper driver module. The module is much smaller and is integrated into the PCB.
- Replaced MC1458 op amps with LM358. This eliminates the requirement for a negative supply voltage, which in turn eliminates the negative DC supply module.
- A BNC RF connector is added.
- An Ethernet connector is added. This is used to connect the controller to the antenna tuner motor and limit switches.
- A coaxial power connector is added for the 12 volt power required.
- Electronic switches were added to save standby power consumption.
- A +12 Volt relay control output was added.
- The number of pushbuttons required was reduced from seven to three. This was allowed due to simplification of the user interface.
- An audio muting board can be added to mute speaker audio during the auto-tuning process.
- A shunt diode plus fuse style reverse polarity protection was added.

The buttons and menu system are de-bounced.

The PCB is 100mm by 100mm and can be fabricated for approximately US $1 (not including shipping cost).

The enclosure is 3D printed and is similar to the design of the original project.

The controller should function with the original project's "double-double" loop antenna. I decided to go a simpler route with a basic single-loop design. The loop is resonated with an eBay sourced butterfly capacitor. The antenna uses a combination of PVC plumbing and 3D printed parts.

The remainder of this document should provide an experienced homebrewer with enough information to duplicate the controller and antenna. The Github repositories containing the project information will be periodically updated as the firmware and components continue to evolve.

Chapter 2

Controller Hardware

Figure 2.1: Top view of the fully assembled loop antenna controller.

2.1 Microcontroller

The original design used an STM32-based board commonly known as the ”Blue Pill”. This is a single-core ARM microcontroller with 64kb of Flash memory, and 20kb of RAM. It has an Analog to Digital Converter (ADC) which is required by the SWR measuring function of the project. The board is supported in the Arduino IDE and has a large range of supporting libraries.

A drawback of the ”Blue Pill” in the Arduino IDE is that there is more than one source for the supporting libraries. These libraries are not necessarily interchangeable. Configuration of the IDE and libraries can be confusing.

Compounding the ”Blue Pill” problem is the numerous variants of the board with microcontrollers sourced from various vendors. There are geniune ST Micro devices, unauthorized clones, and authorized clones. The officially supported Arduino support software will reject the unauthorized clone devices. Alternative support libraries may or may not work with the clones.

When you buy a ”Blue Pill”, you may not be able to determine exactly what you are getting.

In early 2021 Raspberry Pi made a surprise announcement of a new microcontroller product called the ”Pi Pico”. The microcontroller is a dual-core ARM system-on-chip with an ADC, and enough general purpose input-output (GPIO) to support the magnetic loop controller. Flash memory is a much larger 2MB and 260kB of RAM.

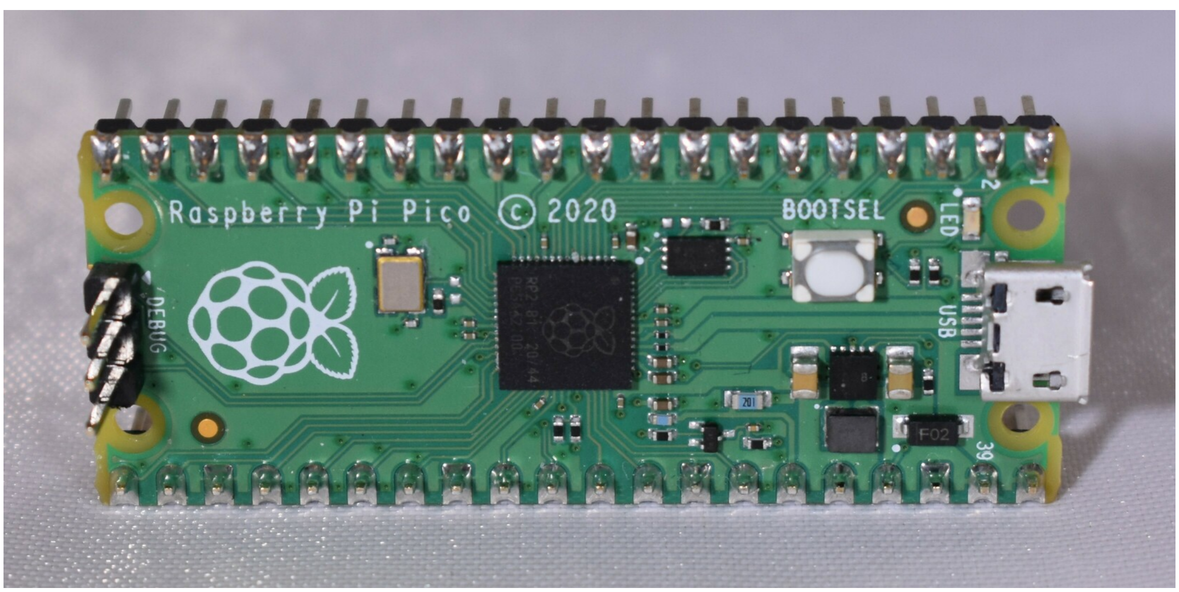


Figure 2.2: The Raspberry Pi Pico board. Note the three-pin debug header on the left side of the board.

The Pi Pico form factor is quite similar to the Blue Pill. The price is also similar, with an advertised cost of $4 (not including shipping). Pi Pico boards have been consistently available in spite of the semiconductor supply chain issues seen in recent months.

Making the Pi Pico greatly attractive is the expansive and well-supported Software Development Kit (SDK) along with numerous examples. The downside of this is new device is that there are not many libraries available, although the ecosystem is continuing to expand.

Arduino support is available, however, this author has not explored this option yet.

2.2 DDS Module, RF Amplifier and SWR Bridge

The DDS Module, RF Amplifier, and SWR bridge circuits are direct copies of the original project.

The DDS Module is based on the AD9850 integrated circuit. The module can be ordered from Amazon or eBay.

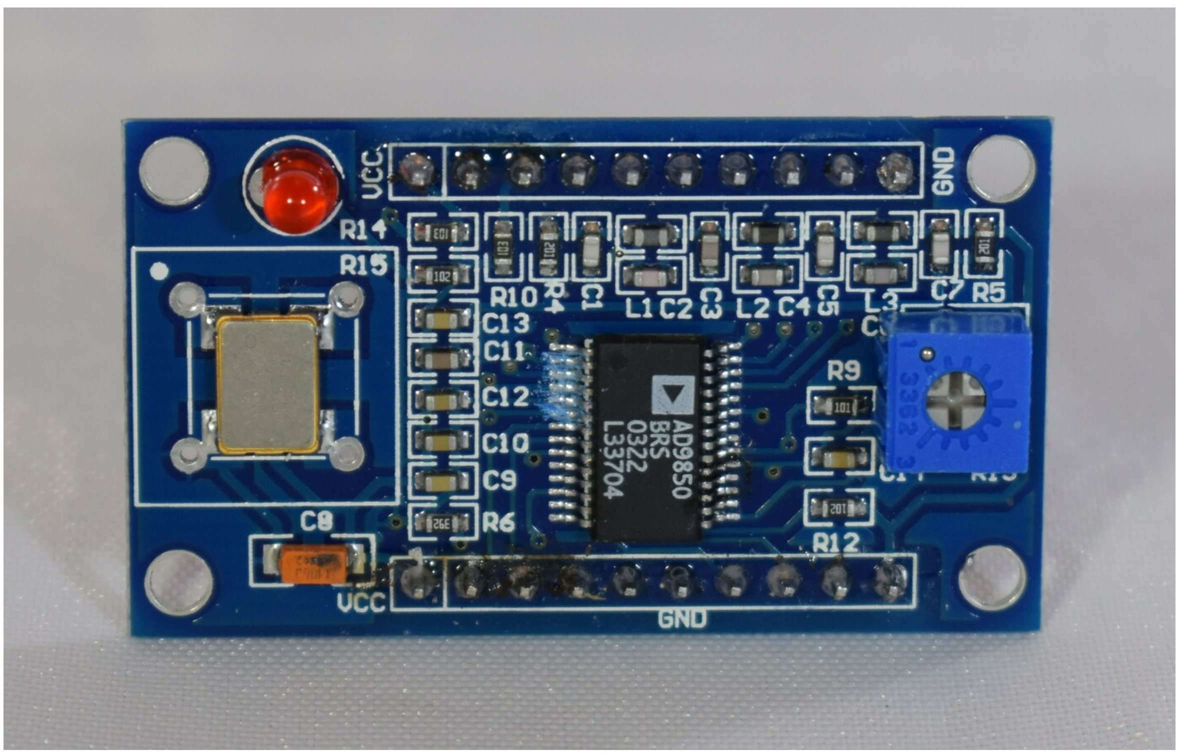


Figure 2.3: AD9850 DDS Module

The RF amplifier consumes significant current, so it was decided to add a PMOS switch connected to a GPO so that the amplifier can be deactivated when not required.

One quirk of the SWR bridge circuit is that it is sensitive to the forward voltage drop of the 1N5711 diodes used. This will not affect the ability of the controller to tune to minimum SWR, but it will affect the accuracy of the SWR reported on the display. I observed improved accuracy when the forward voltage drop of the 1N5711 diodes is lower. Diodes ordered from Amazon by “Chanzon” performed the best.

2.3 Stepper Motor Driver

The original project used a large external stepper motor driver called the TB6600. This is a robust device with DIP switches to control step size and current drive. It is capable of 1/32 microsteps. The main problem with this device is that it is larger than the entire PCB and needs to stand up vertically. So the project’s case must be much larger to accommodate this stepper driver.

Meanwhile, I had taken the cover off the controller of my 3D printer, and discovered that it used very small plug-in stepper driver boards. These driver boards are about the size of a postage stamp! They are based on the A4988 driver device. However, these drivers are capable of only 1/16 microsteps. This requires an external DIP switch or resistors to select. The current is adjusted by a very tiny potentiometer on the board. The authors of the original project stated they used 1/16 microsteps, so perhaps this small board would be adequate. So far it has! The size reduction of the finished device in the case is substantial. The A4988 cost is about $2, versus $12 for the TB6600, so it is a nice cost reduction as well. The modules can be purchased from Amazon or eBay.

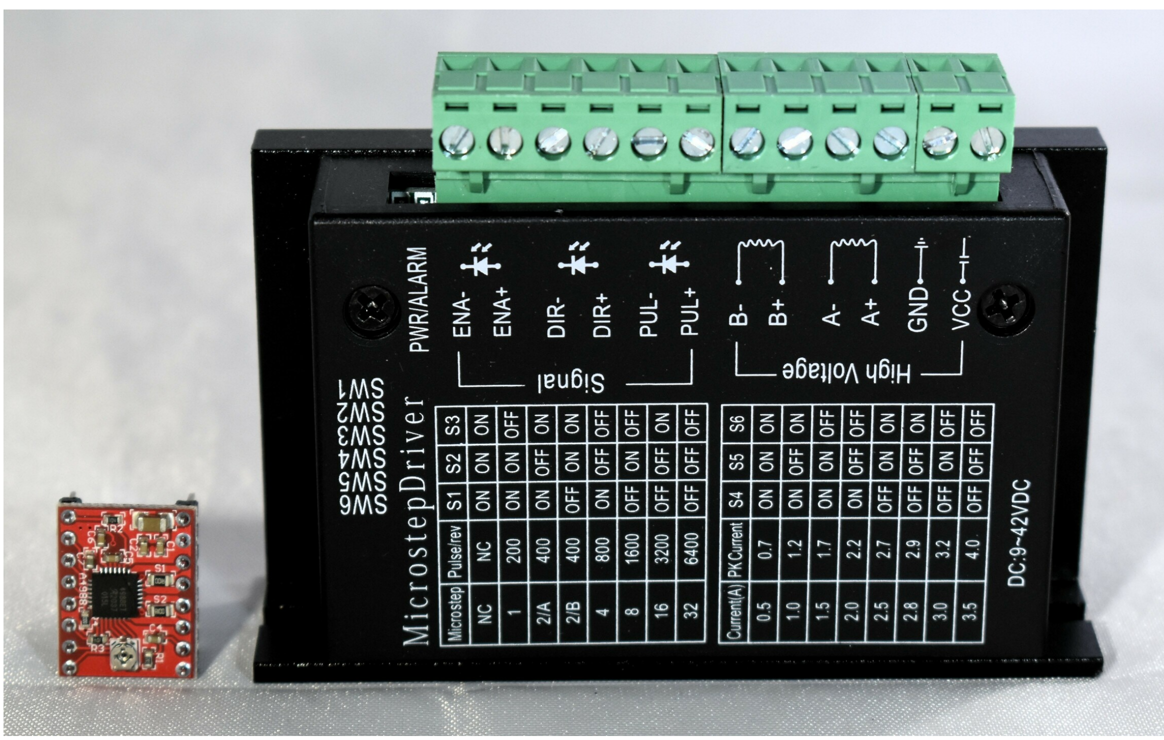


Figure 2.4: Size comparison: A4988 Module on the left, and the TB6600 driver on the right

2.4 External RF Relay

In order for the controller to tune the antenna to resonance and minimize SWR, it must be directly connected to the antenna. After tuning, the antenna must be switched over to the RF transceiver. An electromagnetic relay is required. A PMOS switch is used in the controller to switch +12 Volts to a coaxial power connector on the rear of the controller. This should be connected to the coil of the RF relay. The coil is de-energized during normal operation of the transceiver. The coil is energized only when the controller is auto-tuning. Thus when controller power is off, the transceiver is connected to the antenna.

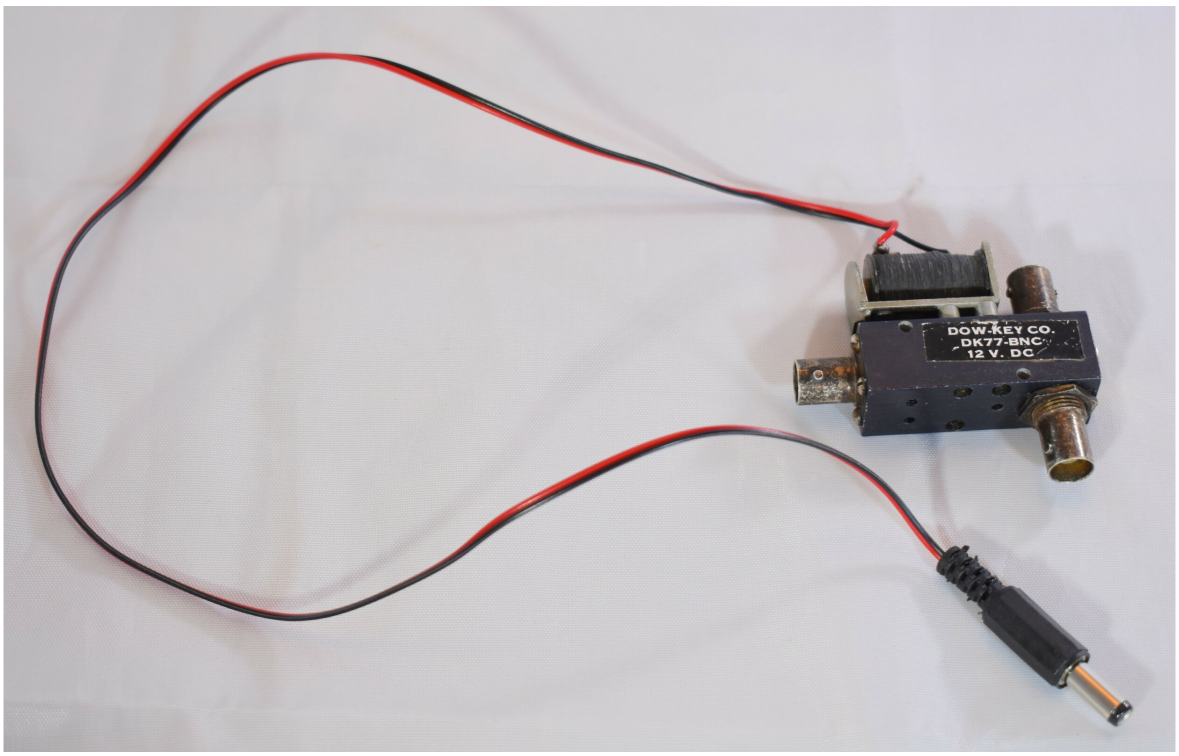


Figure 2.5: A "Dow Key" type external RF relay with coaxial DC connector

2.5 ILI3941 Display

The ILI3941 is a 320x200 pixel color display module. The module mounts to the front panel, and it is connected to the controller board using a ribbon cable and an 8 pin IDC style connector.

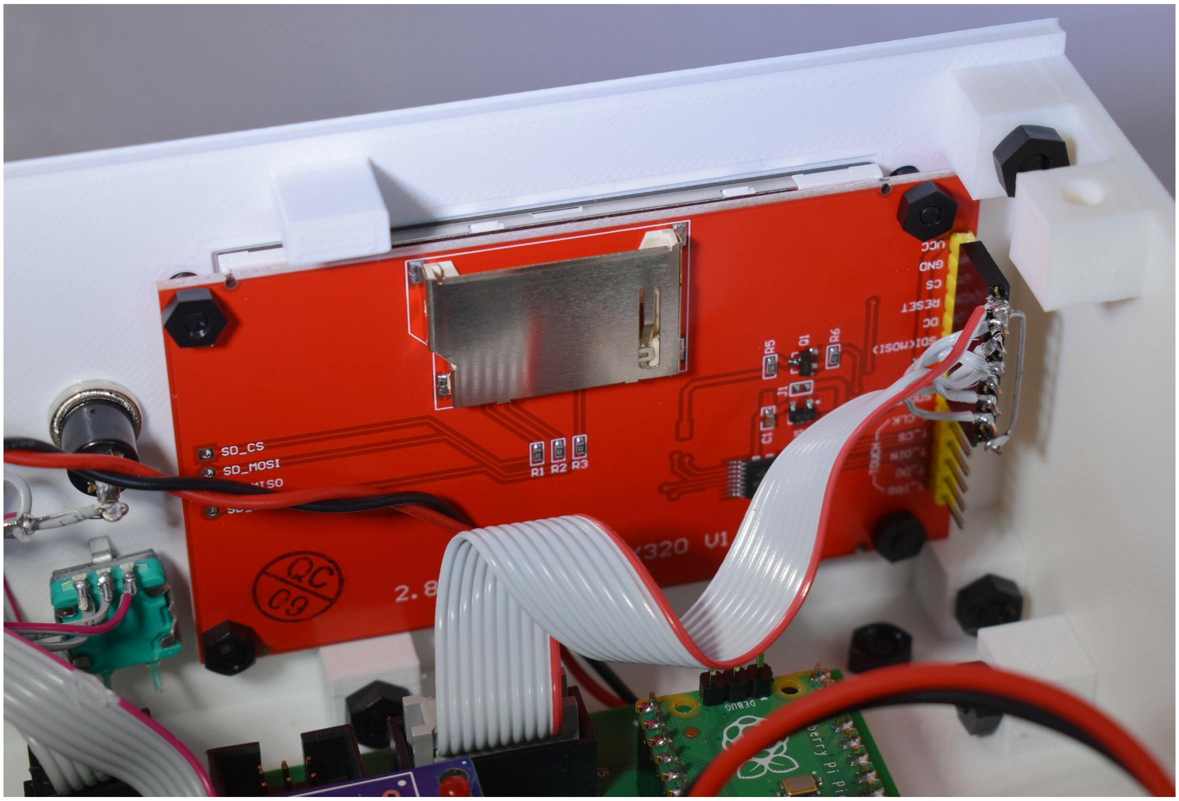


Figure 2.6: The ILI9341 display mounted to the back side of the front panel.

2.6 Butterfly Capacitor

The "butterfly capacitor" is a critical component! The capacitor determines the frequency range of the antenna. It also determines the power handling capability via the spacing of the capacitor plates.

The capacitor can be ordered from eBay from the TA1LSX store:

`https://www.ebay.com/str/ta1lsxstore?_trksid=p2047675.m3561.l2563`

2.7 Audio Muting

Please refer to the chapter on the audio muting board.

Chapter 3

Controller Schematic

The RF circuitry from the DDS to the output of the SWR bridge was a direct copy of the original project.

The forward and reverse SWR amplifiers were changed significantly. These two-stage operational amplifiers have sufficient gain to amplify the SWR bridge output to a level high enough to utilize close to the full input range of the microcontroller's ADCs. Another function of these amplifiers is to linearize the output of the SWR bridge, which is a pair of diode detectors. A diode feedback circuit is used in the input stage, which linearizes and improves the dynamic range of the SWR measuring system.

The original system used 741 operational amplifiers with a dual $\pm$5volt supply. This requires a negative 5 volt power supply board.

The 741 op-amps are replaced by LM358s, which can operate with a single supply and have a useful input range to ground. Since the SWR detectors output positive voltage only, this is acceptable. These op-amps have low input offset voltage, however, it is enough to disrupt the SWR measurement accuracy. Input offsets are effectively removed in software. The circuit is very similar to the original with the exception of the single supply replacing the dual.

Two PMOS transistors controlled by GPOs are added to reduce current drain during stand-by. The +5 Volt PMOS switch also provides a signal to the audio muting board during auto-tune. A third PMOS outputs +12 Volts to an RF relay to switch the antenna to the controller during auto-tune.

A 1N4004 diode is in shunt with the +12 Volt DC power input. This diode is used with a fuse mounted in the rear panel for reverse-polarity protection.

An A4988 stepper motor driver module is added. This replaces the large external motor driver used in the original project. The A4988 module is connected to an Ethernet connector on the rear of the PCB. Limit switch inputs are also routed to the Ethernet connector.

The Raspberry Pi Pico module was wired according to the directions in the documentation:

https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf

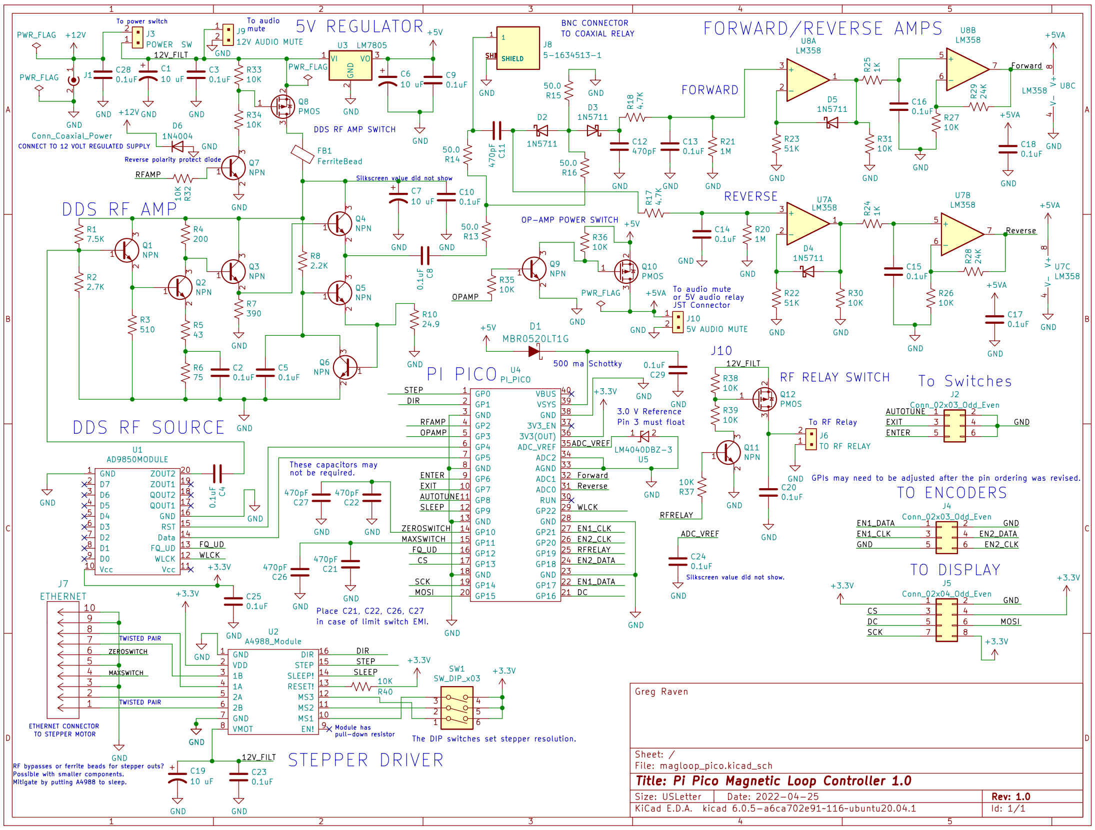


Figure 3.1: Magnetic Loop Controller Schematic

Chapter 4

Printed Circuit Board

4.1 Printed Circuit Board

The first version PCB was derived from the original STM32 project. Since the Pi Pico is approximately the same size, it was dropped in place.

The first Pi Pico board used a through-hole wire terminal for the stepper motor and limit switch connections. These are connections which must be made through a long cable to the remote antenna tuner.

Cheap lawn sprinker cable was used to connect the controller board to the tuner via the terminal strip. This would function fine with a short cable. However, upon trying a 50 foot long cable the controller went nuts!

What happened is that the very square pulses output from the stepper driver were coupling to the limit switch inputs. This causes big problems as the controller senses an incoming pulse train on the limit switch inputs, rather than a single step which is what should happen when a limit switch closes.

So I took this problem to the Groups.io Software Controlled Amateur Radio group, which is frequented by builders and users (and the authors of the book) of the original magnetic loop controller. Similar problems were seen, with the best solution to change to a shielded CAT7 Ethernet cable.

After successful testing breadboards of the Ethernet cable “fix” for the limit switch problem, it was decided to make a new PCB with an Ethernet connector mounted to the board.

This version of the board added PMOSFET switches for an antenna relay, the 5 volt supply to the op-amps, and 12 volt supply to the RF amplifier. Unfortunately there was only one spare GPIO, so all of these switches operated simultaneously.

The new external RF relay switch is required to disconnect the controller from the antenna during normal transmit/receive operation after autotuning is complete. The other switches are mainly intended for current drain reduction when the circuits are not required in normal operation of the transceiver.

The second prototype board was a success, however, the Ethernet connector was located on the side of the board. This means an Ethernet jumper would be required to connect to another Ethernet connector on the rear of the case. Not good!

So it was decided to fundamentally rearrange the main components on the PCB for the third prototype. The Ethernet connector was moved to the rear of the board, adjacent to the power connector. This worked well, as the stepper driver was also moved to the back and this made the routing of 12 volts to the stepper driver much shorter.

Both the Pi Pico and the DDS boards were moved to new locations. When the re-routing was complete, the board routing was simplified, and it should perform better than the second prototype.

Routing of signals to the IDC connectors for display, encoders, and pushbuttons was done such that making the ribbon cables is as easy as possible.

Gerber and drill hole files for fabrication of the board are located in the magloop_pico_pcb repository.

Chapter 5

Firmware

The firmware for the project was developed using a Raspberry Pi 4 single-board computer. Raspberry Pi has done an outstanding job providing a script which installs everything required to develop code for the Pi Pico, including Visual Studio Code, which is an advanced text editor. It is essentially an IDE.

Debugging is done via Serial Wire Debugger interface (SWD). The SWD is connected to the Pi Pico via three jumpers from the Raspberry Pi 4 GPIO connector to a three pin header on the Pi Pico. No special “dongle” is required!

The supported debugger is the main reason this project was developed using the Pi Pico SDK. Debugging is only recently available in the Arduino IDE 2.0. However, an initial look at the Arduino debugger indicated it was not mature compared to the Pi Pico SDK. In the future, the Arduino IDE may be revisited if the Pi Pico support includes a solid debugging feature.

5.1 Organization of the Code

The code is organized differently than the original project. The original project is an Arduino style project, with the use of C++ classes (libraries) and multiple files with associated C-style functions in each. It is the typical “setup” and “loop” structure of Arduino.

This new project based on the Pi Pico is more strongly C++. Rather than files with numerous C-style functions, these functions were gathered up and put into C++ classes. The main function instantiates these classes in the form of several objects. I found this “C++ embedded programming” helped me to create easier to understand code. The fundamental programming principle is called “separation of concerns”:

https://en.wikipedia.org/wiki/Separation_of_concerns

“Inheritance” was used where it naturally fell into place. For example, the “StepperManagement” functions were organized into a C++ class, and this class inherits from the AccelStepper class library.

The “state machine” pattern was used in several places including the GUI menu in the main function.

The CMake build system is used to compile, link, and create various associated files. This is one disadvantage of using the Pi Pico SDK. You don’t have to deal with this in Arduino. After a bit of struggle with CMake, I got it all working together smoothly.

5.2 Data Class

There is a new “Data” class which stores all of the constants used to configure the controller. This class is intended to replace all or most of the #define statements which appear in the original design. A data object is instantiated in the main program at start-up. The data object is referenced by the other objects and is used to access the relevant constants.

5.3 EEPROM Class

The EEPROM class performs the same function as the original project. It provides non-volatile storage for the band limit stepper motor positions after initial calibration. Like the original project, it uses FLASH memory to mimic EEPROM functionality. The Pi Pico SDK FLASH functions were used to create this class.

5.4 ILI9341 Driver

As mentioned before, the library support for the Pi Pico is in the early stages. I needed a library for driving the ILI9341 TFT display. There is an Arduino driver, and I set off on a mission to hack it to work with the Pi Pico.

The problem with hacking an Arduino library is that the code is designed to work with multiple architectures. The code is littered with #if statements which select blocks of code depending on which architecture is in use. This was painful to work with, on top of the fact that it is a complex driver.

After hours and hours of hacking, I got the ILI9341 working with the Pi Pico. If a driver specifically intended for the Pi Pico appears, I will certainly consider replacing the monstrous hack I created. Having stated that, it has been working reliably and consistently with all of the prototypes. Having the display is also great for debugging because you can print anything you want to the display. That combined with the debugger made the rest of the code development quite pleasant.

The original Arduino library:

https://github.com/adafruit/Adafruit_ILI9341

5.5 Stepper Motor Library

The original “AccelStepper” library was modified to work with the Pi Pico. The modifications for this library were minimal.

The original Arduino library:

https://www.airspayce.com/mikem/arduino/AccelStepper/

5.6 Rotary Encoder Library

The original “Rotary” library was modified to work with the Pi Pico. The modifications for this library were minimal.

The original Arduino library:

https://github.com/buxtronix/arduino/tree/master/libraries/Rotary

5.7 De-bouncing

De-bouncing functions are included in the Button class.

The original project used simple time-delays for debouncing the buttons. This project has state-machine based de-bounce. This was challenging to implement, as it also had to include de-bounce when moving up and down the menu levels. An auxiliary function was required along with careful placement of the debounce functions in the code.

5.8 Limit Switch Protection

The low and high limit switches are connected to GPIs via the CAT7 cable. These GPI states are continuously monitored during stepper motor movements. If either switch closes, the stepper motor will be stopped. This is to protect the limit switches from damage due to over-rotation of the stepper motor. This is most likely to happen during “bring up” of the controller as connections could be accidentally reversed. It is also possible to hit the high limit if the controller is commanded to go to a frequency beyond the limit of the upper limit of antenna resonance.

In case of an upper limit event, the user is notified of an ”Upper Limit Hit” via the display. Note that the primary function of the low limit switch is for calibration of the stepper motor. Its secondary function is to protect itself from destruction.

Chapter 6

Loop Antenna Construction

The original project uses a ”double-double” loop structure formed from copper plumbing pipe with 3D printed supports and PVC plumbing pipe.

That structure was a bit too much for my limited ship capabilities, so I opted for the common single-loop design. I had a chunk of semi-rigid coaxial cable lying around. So that became the “loop” part of the antenna. It is possible to use copper or aluminum tubing, and I am especially interested in building a version which uses lightweight aluminum refrigeration tubing.

The 3D printed parts for the double-double inspired the parts used in the single-loop antenna. The mechanics of the stepper motor drive for the butterfly capacitor are also similar to the double-double.

The stepper motor, bracket, limit switches, and pulley parts were sourced from Amazon. With regards to the pulley, you must make sure to order the pulleys with the same mounting hole diameter as the motor and capacitor shafts. A list of links to these parts is provided in the BOM file.

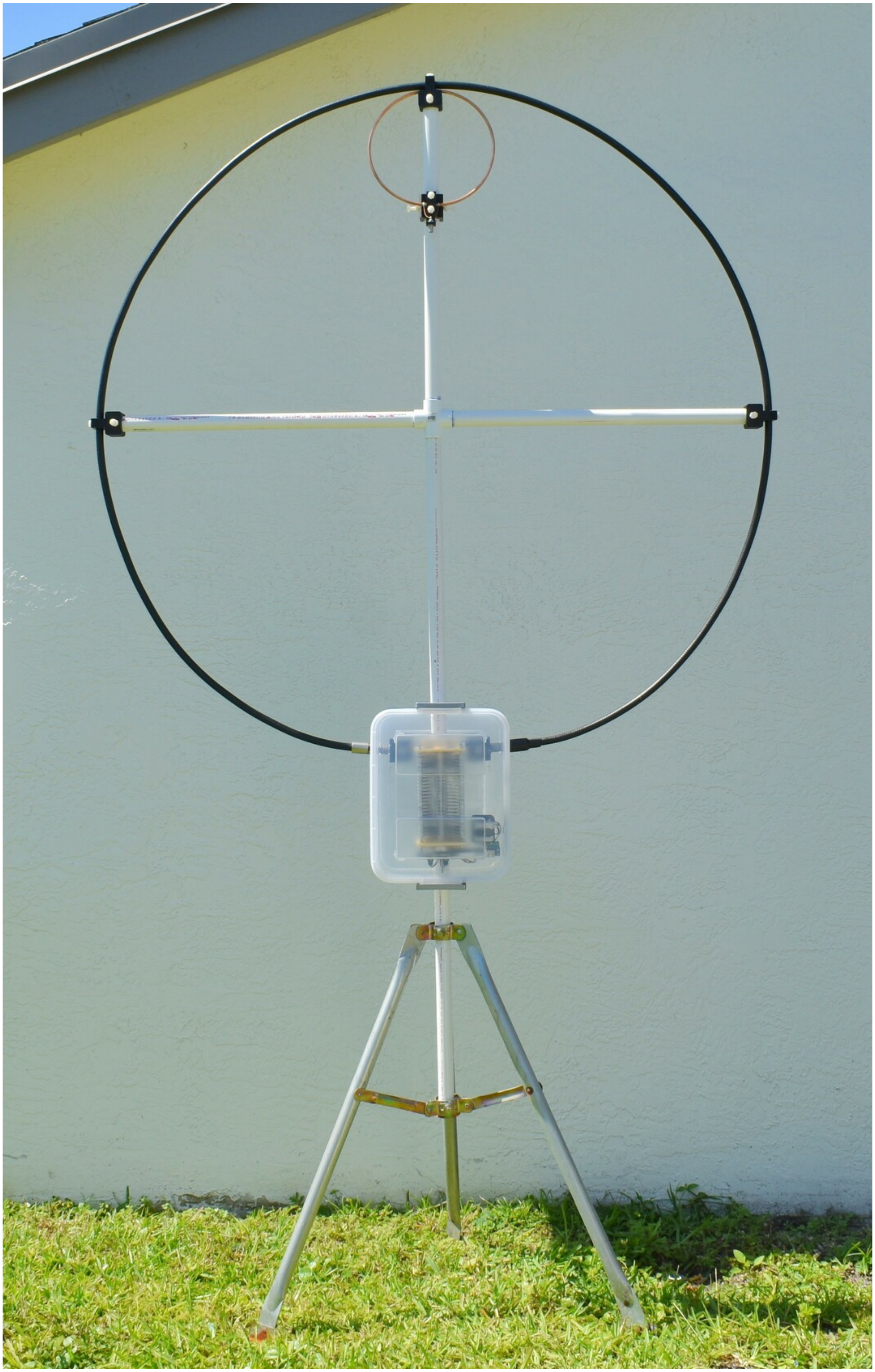


Figure 6.1: The complete loop antenna.

6.1 3D Printed Parts

I am going to warn you that the 3D printed parts for the antenna are to be considered ”prototypes”. You will probably need to change them to fit your material and requirements. But that is the beauty of 3D printing! I tried to keep the structure as simple as possible.

The antenna structure is supported by one-inch schedule 40 PVC pipe. 3D printed “clamps” are used to hold the loop in place. Another set of clamps holds the “tuner” assembly in place at the bottom of the loop. The antenna tuner components are bolted to a rectangular piece of Plexiglas. Nylon bolts are used throughout to keep the weight down.

6.1.1 Left and Right Loop Supports

These are relatively simple parts which clamp the loop to the ends of the horizontal PVC pipes. Nylon bolts are used for both the loop clamp and also to secure the support to the PVC pipe. It is also possible to use glue to secure the supports to the PVC.

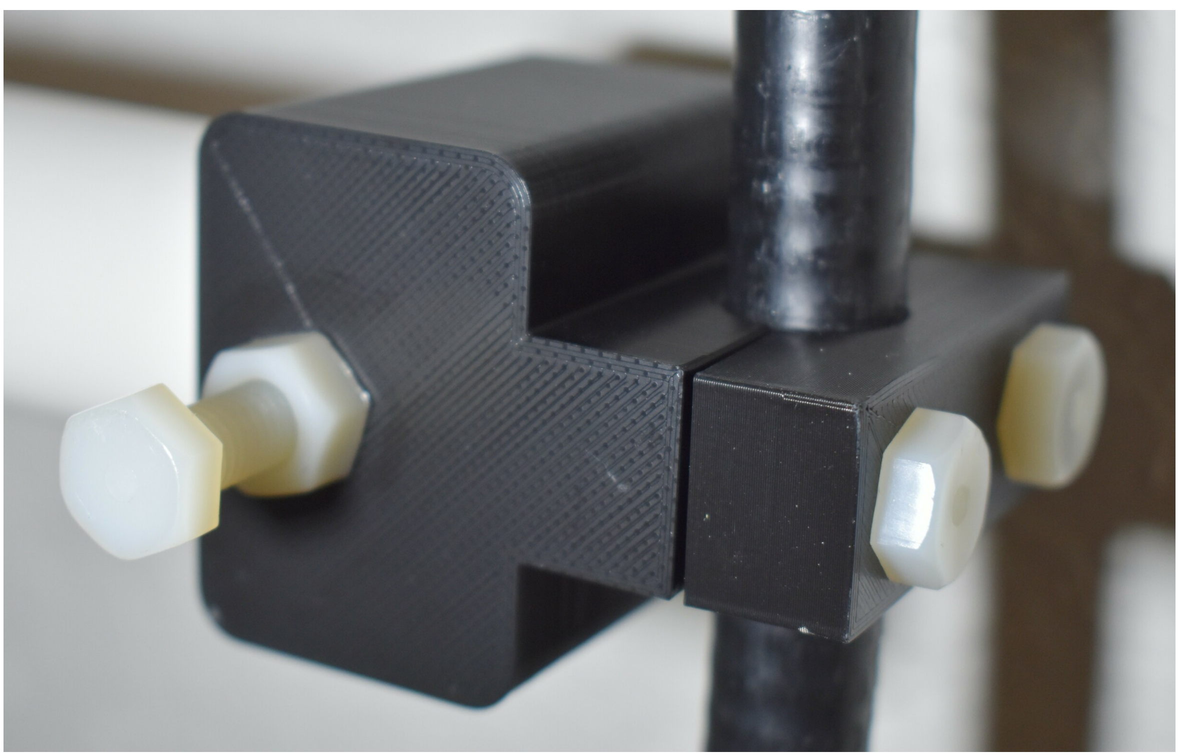


Figure 6.2: Side support for the loop conductor. The support is the same for left and right.

6.1.2 Top Support

The top support is similar to the left/right supports, with the additional feature of a slot for the coupling loop.

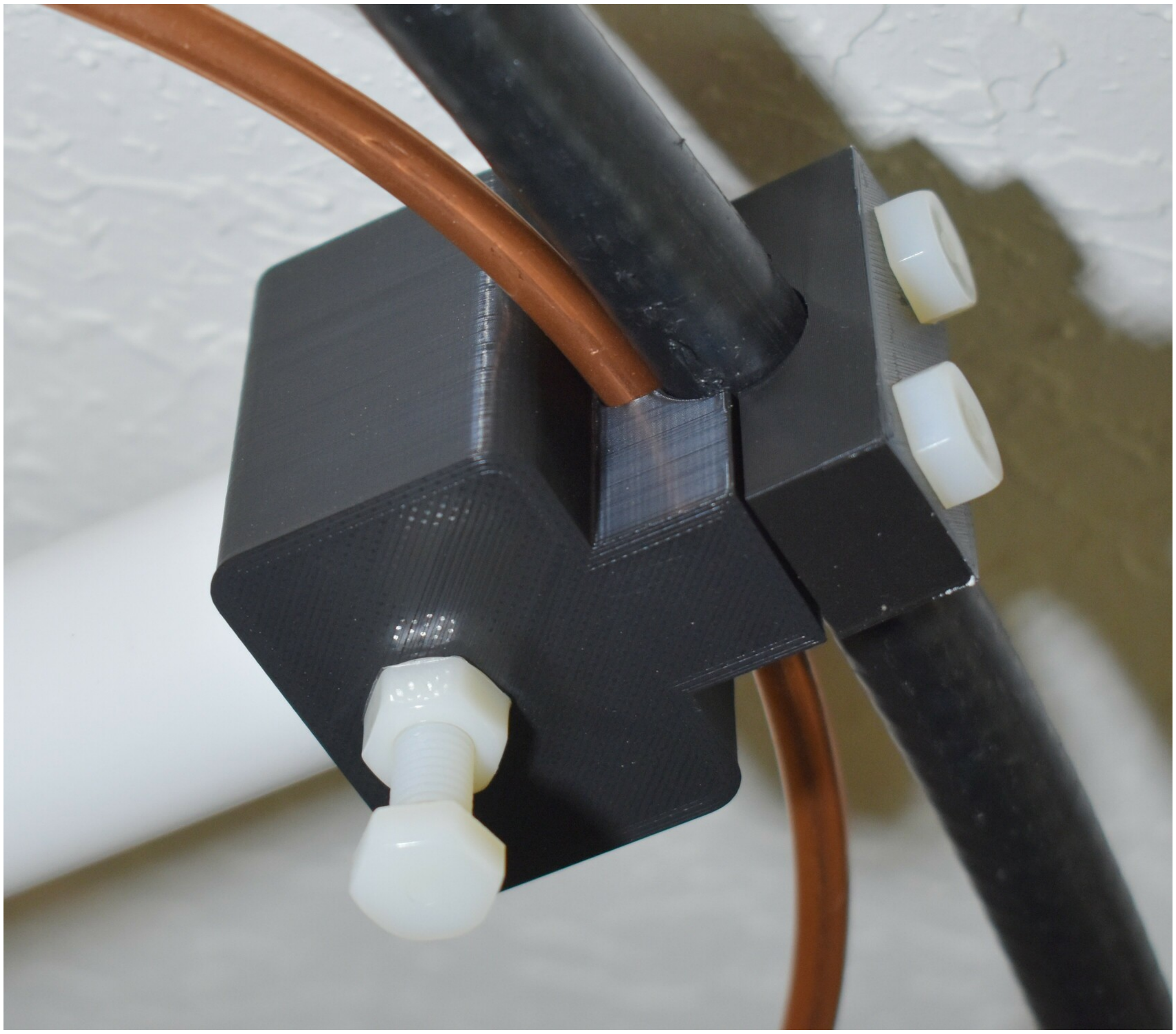


Figure 6.3: Top support for the loop conductor. Note the slot for the coupling loop.

6.1.3 N Connector Brackets

The cable I used for the loop has male N connectors. I designed simple brackets for chassis-mount style female connectors. Note that the center conductor is not used and is floating. Soldered terminals and solid copper wire connect the outer "shield" of the coax to the butterfly capacitor.

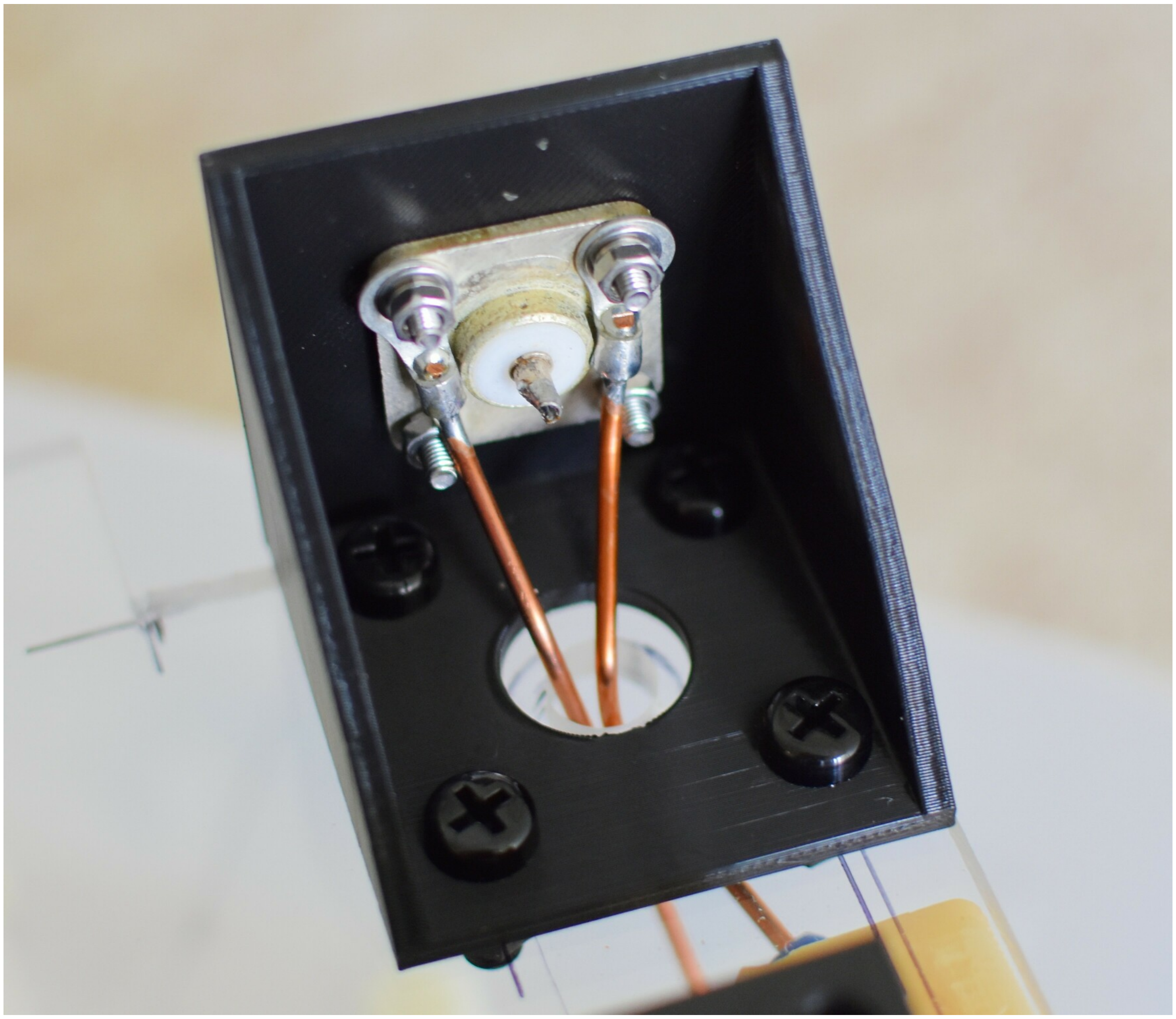


Figure 6.4: N connector bracket.

6.1.4 Coupling Loop and BNC Connector Bracket

The coupling loop is formed from quarter-inch copper plumbing pipe. The diameter was experimentally derived by making loops of three different diameters. A "NanoVNA" was used to check the SWR over the desired frequency range.

The coupling loop diameter is 26 centimeters. It is possible it could be further optimized, but the improvement if any would be minimal.

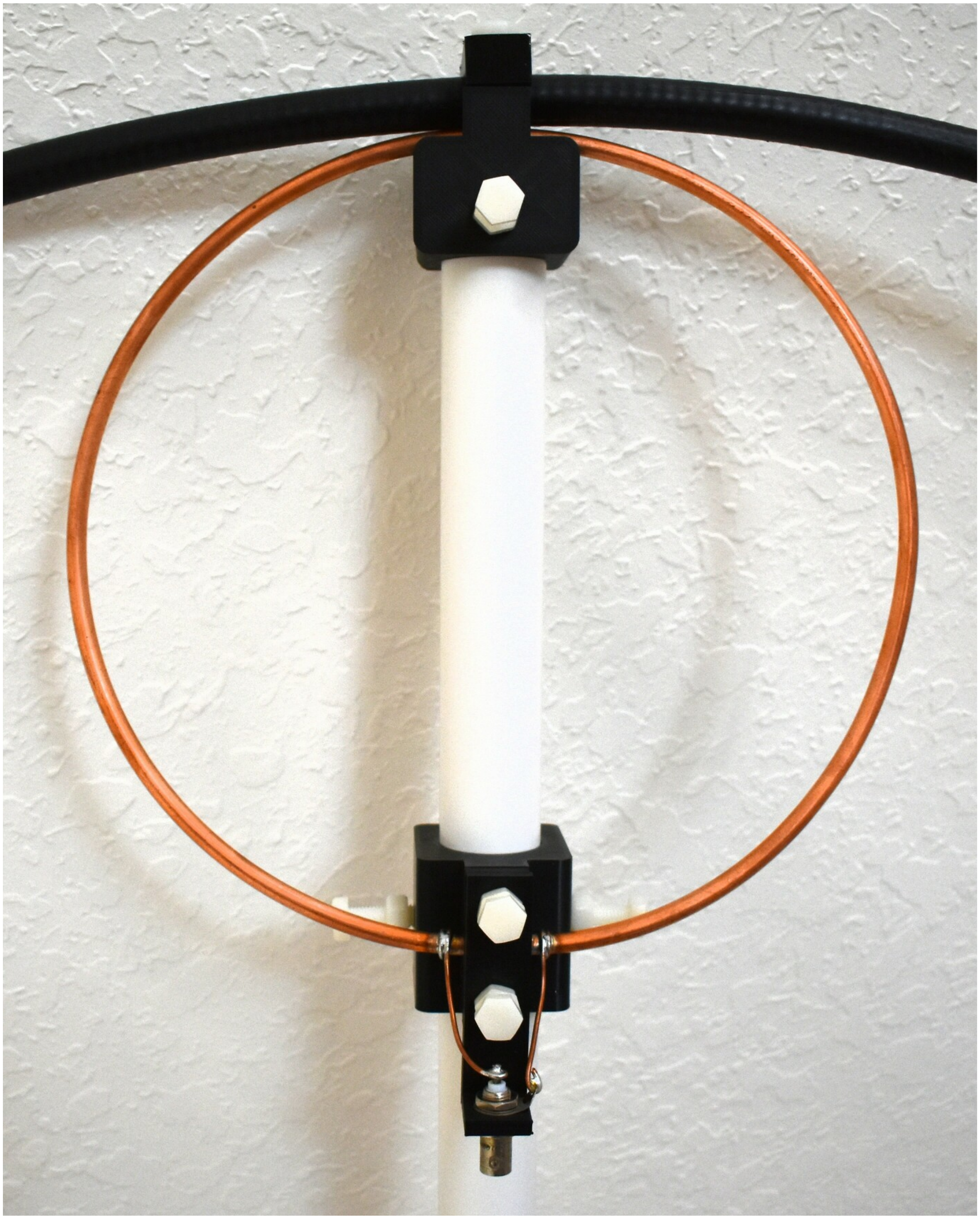


Figure 6.5: The coupling loop assembly.

The coupling bracket slides over the one-inch PVC pipe and is secured in place with nylon bolts. A 3D printed bracket holds the BNC chassis-mount female connector, and it also bolts to the bracket to secure the ends of the coupling loop. Solid copper wire is used to connect the coupling loop to the BNC connector.

The lower bracket for the coupling loop is composed of three parts:

1. The main bracket piece which slides over the PVC pipe.

2. A copper tube clamping piece which is sandwiched between parts 1 and 3.

3. The angle bracket for the BNC connector

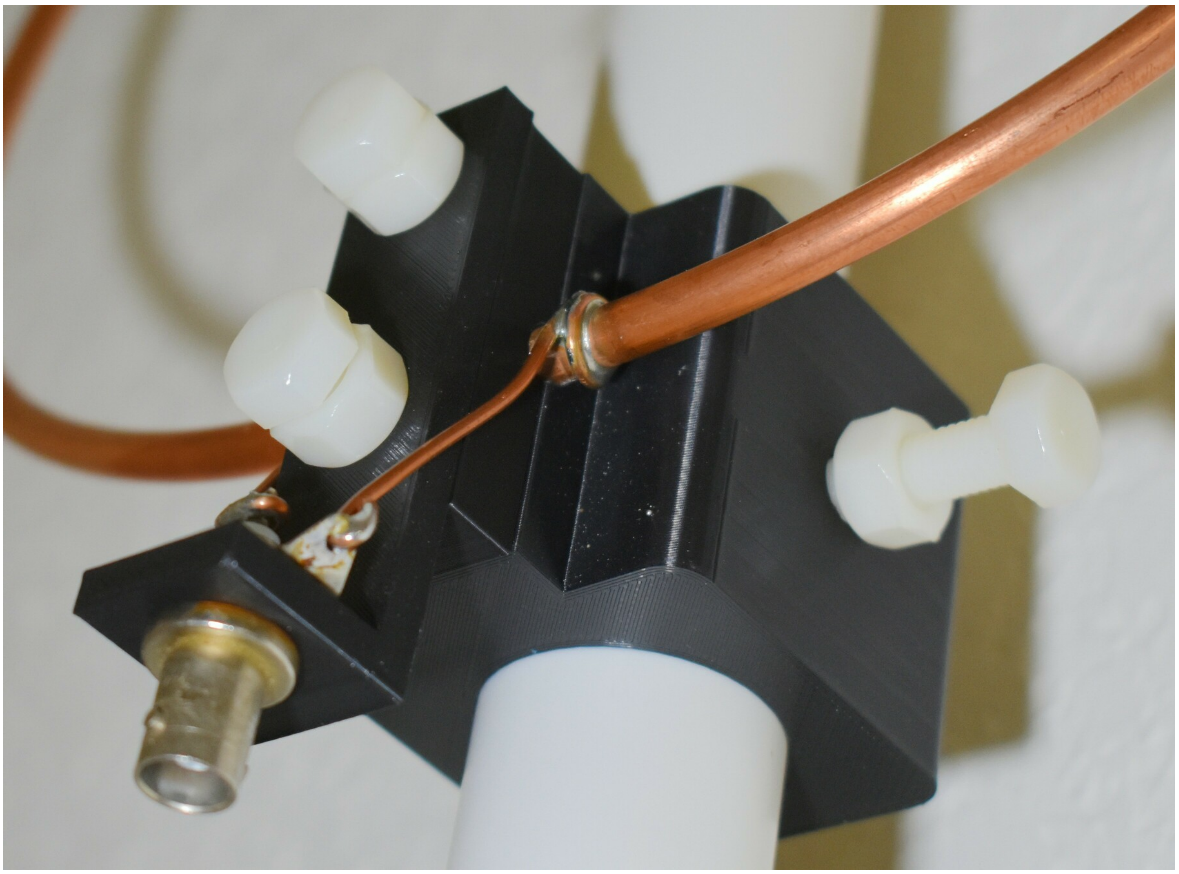


Figure 6.6: The lower coupling loop bracket with BNC connector bracket "sandwich" assembly.

6.1.5 Tuner Assembly Clamps

Four clamp pieces are required to secure the tuner assembly to the vertical PVC pipe. These clamps grip the pipe and hold the tuner assembly via holes drilled in the Plexiglas tuner assembly base. Thus there are a total of six bolts required for each set of clamps, for a total of twelve bolts. The photos below will clarify the construction of this assembly.

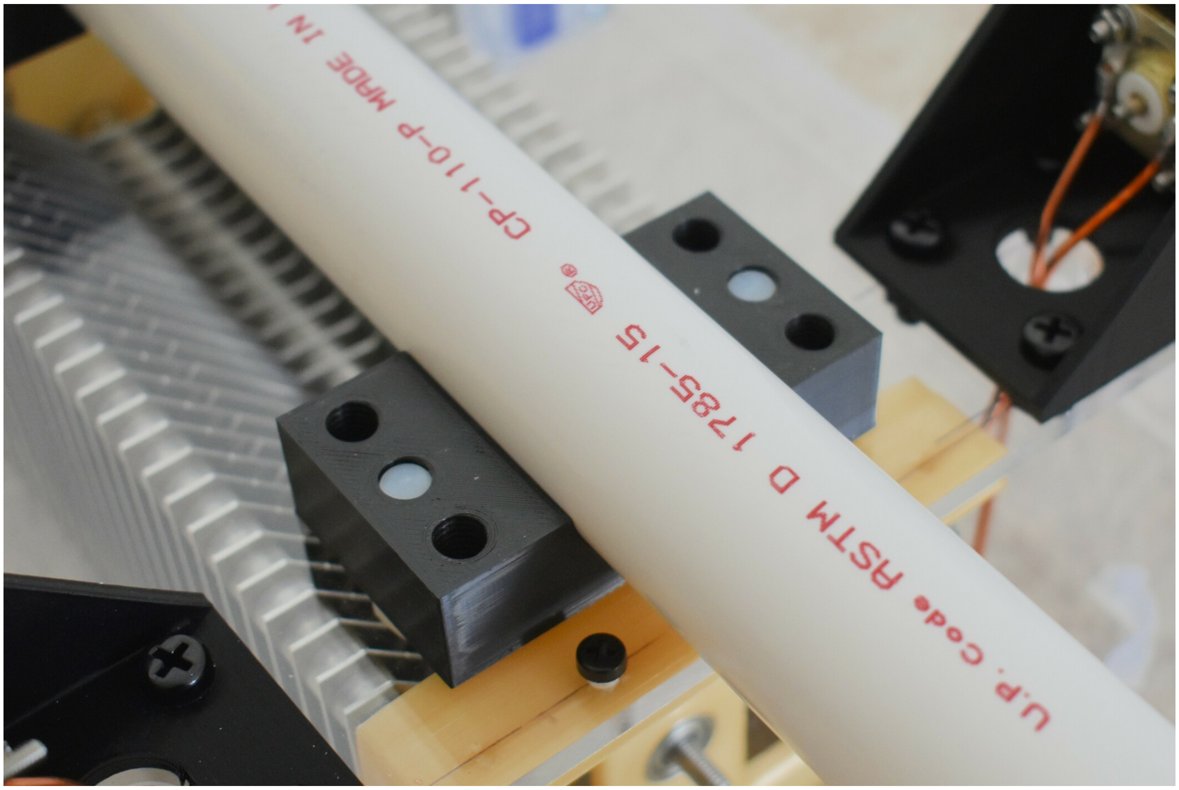


Figure 6.7: One of two pipe brackets before adding the clamp piece.

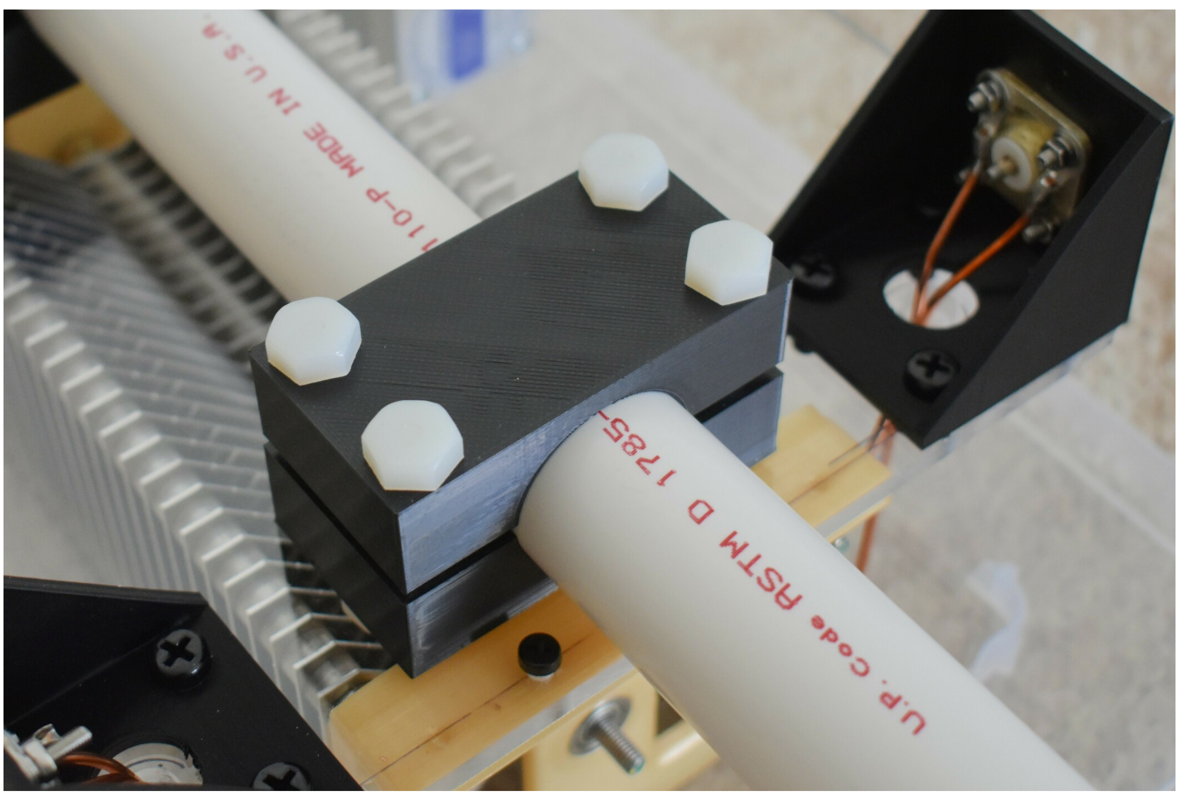


Figure 6.8: One of two pipe brackets after adding the clamp piece.

6.1.6 Limit Switch Bracket

The limit switches are small modules ordered from Amazon. The modules include the plug and wiring. The switches must be held at the proper height such that the switches close at the high and low capacitance limits of the capacitor. A 3D printed bracket was designed to accomplish this. The height may require adjustment depending on the exact mechanics used. However, only the "zero" limit is critical. This is the switch which closes on the high-capacitance end of the rotation. This switch provides the primary calibration which is used by the controller to steer and auto-tune the resonant frequency of the antenna.

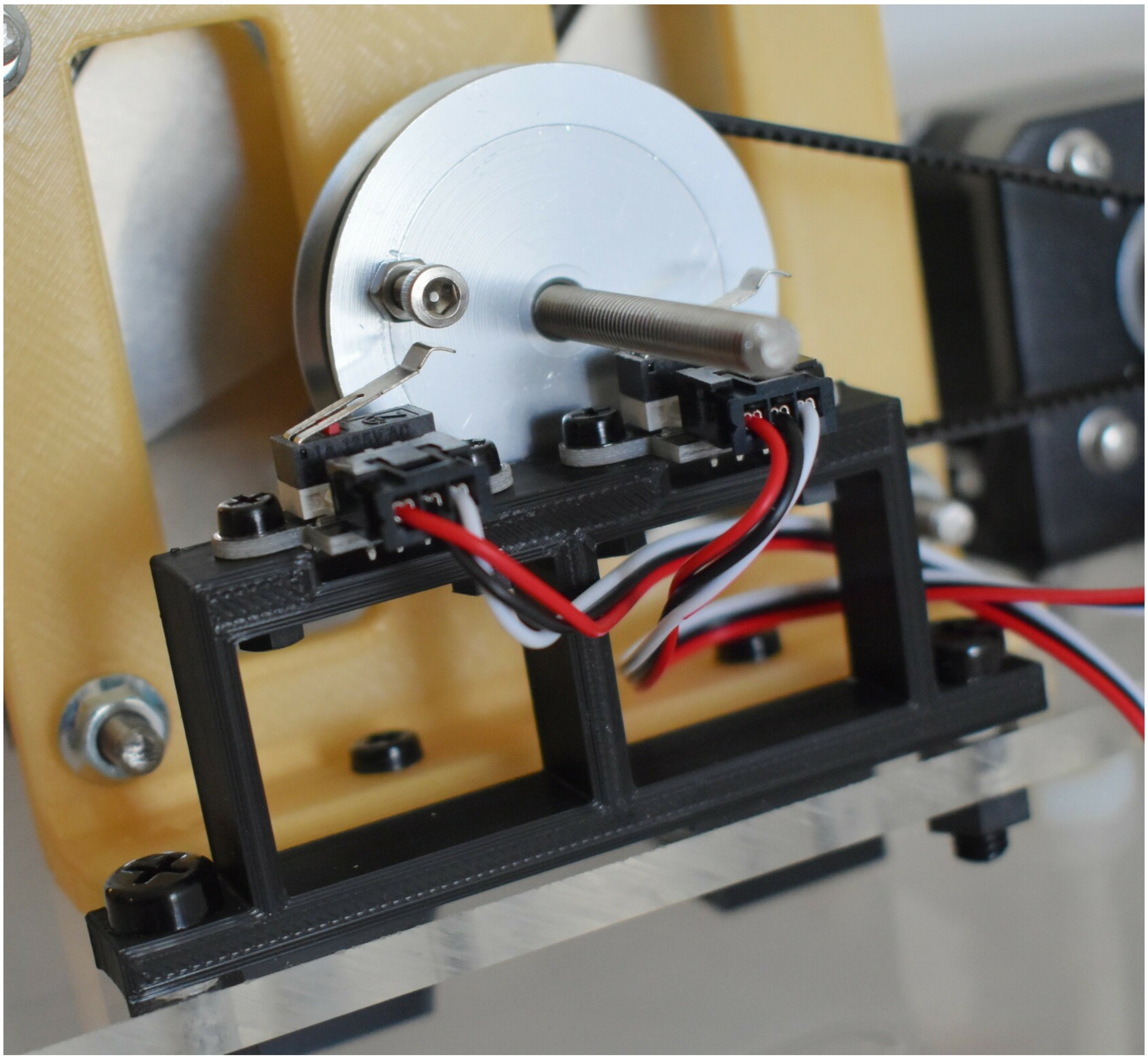


Figure 6.9: The limit switch bracket.

6.1.7 Ethernet Connector Bracket

The stepper motor and limit switch connections use a "breakout" board ordered from Amazon. This board is a simple Ethernet to terminal block. A 3D printed bracket and four bolts attaches the board to the Plexiglas tuner base. The motor limit switch wires connect to the terminal block.

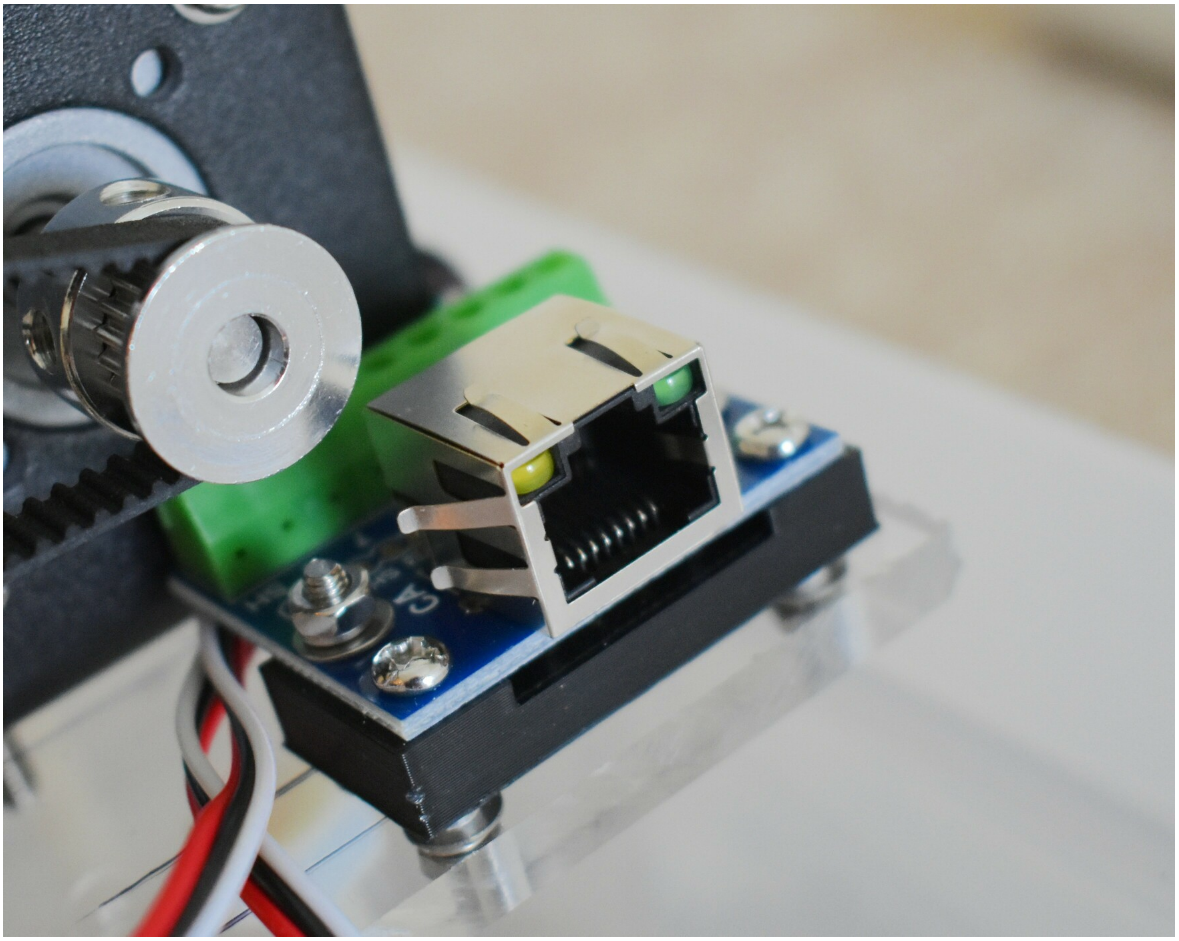


Figure 6.10: The Ethernet connector bracket.

6.1.8 Stepper Motor Bracket

The motor mounting bracket was purchased from Amazon along with the stepper motor. This bracket has slotted mounting holes, which allow for adjustment of the motor location, which is a good feature.

The motor is a NEMA 17 1.5 Amp rated motor. The torque required to turn the butterfly capacitor is low, so it is possible for a smaller motor to work, but this has not been tested.

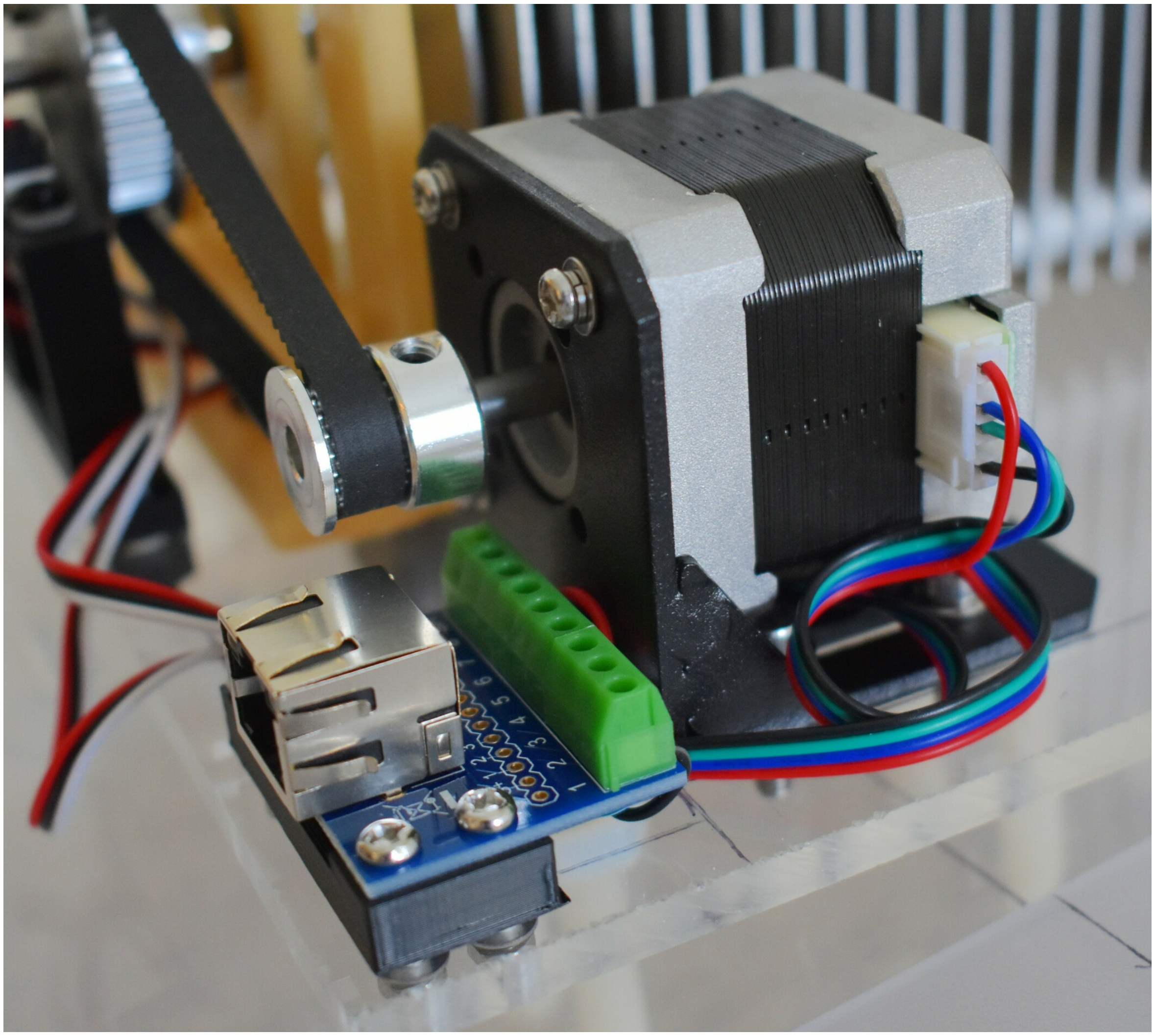


Figure 6.11: The stepper motor and bracket.

6.1.9 Pulleys and Belt Drive

The pulleys and fiber belt are 3D printer parts ordered from Amazon. Be sure to get the correct pulley hole diameter to match the shafts of the motor and capacitor.

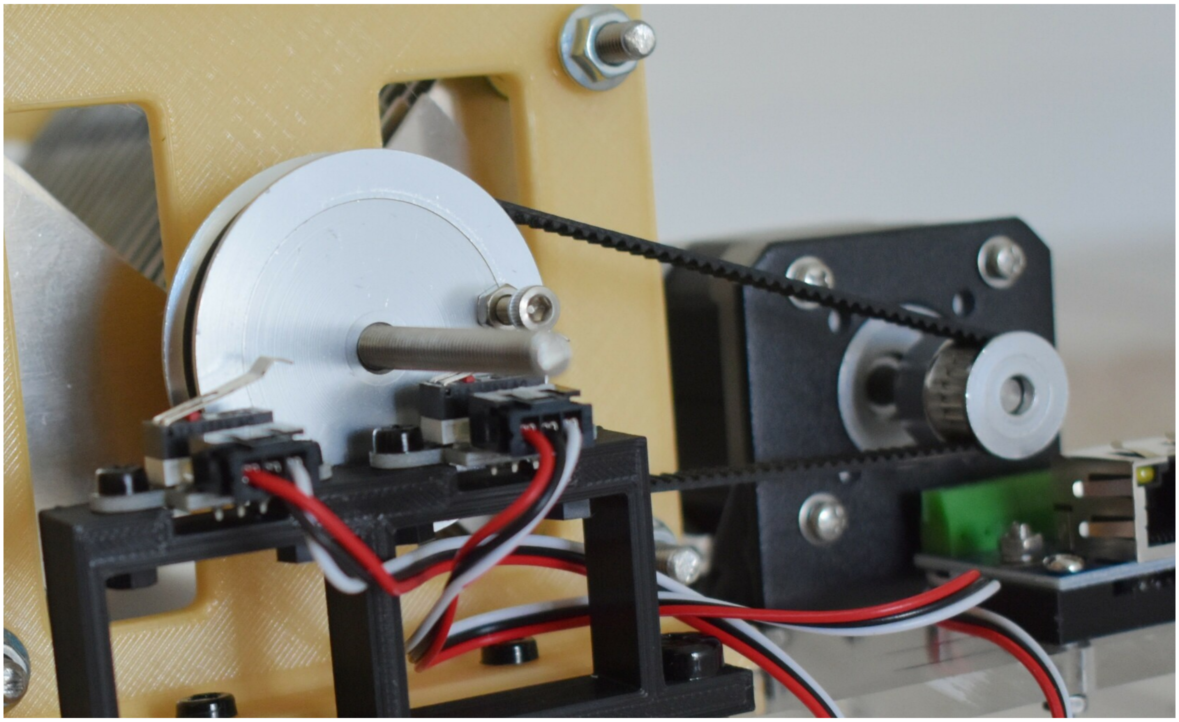


Figure 6.12: The pulley and belt.

6.1.10 Plexiglas Tuner Assembly Base

A rectangular piece of Plexiglas was ordered from Amazon. This Plexiglas should be reasonably thick for mechanical stability. A link to the part used is provided in the BOM.

A lot of hole drilling is required to mount the various pieces of the tuner assembly.

It is best to get all of the parts ready, including the 3D printed parts, before proceeding with preparation of the base. Otherwise it is easy to make a mistake. Arrange all of the parts on the base, and make sure all clearances make sense, and that there is room for all of the components. Make very careful measurements, and mark out the hole locations on the Plexiglas.

Measure twice, cut once!

Drilling Plexiglas requires very sharp drill bits. Go slowly or the Plexiglas will melt!

I recommend a little slop in the motor mount holes. This will allow some adjustment of the motor position in order to tighten the belt. The belt needs to be snug, but not overly tight.

I used a combination of nylon and steel hardware for mounting the components to the base. Amazon has good hardware “kits” of both nylon and steel hardware. I used exclusively metric hardware, and I am happy with the completed tuner assembly.

6.2 Comments on Threads in 3D Printed Parts

Getting the 3D printed threads to work well required experimentation. I recommend designing a simple rectangular block with a threaded hole to test the quality of the threads. I used Freecad 0.20 which includes a threading tool. This tool has a "clearance parameter" which can be adjusted to adjust the tightness of the threads. This will have to be adjusted to get the threads "just right".

It is better to use larger and coarse threaded bolts. I used M8-1.25 nylon bolts. In future designs, larger bolts may be used. Links to the parts ordered from Amazon are included in the BOM spreadsheet.

6.3 Butterfly Capacitor

The butterfly capacitor is assembled from a kit which can be ordered from an eBay store. The URL to the store is included in the BOM spreadsheet.

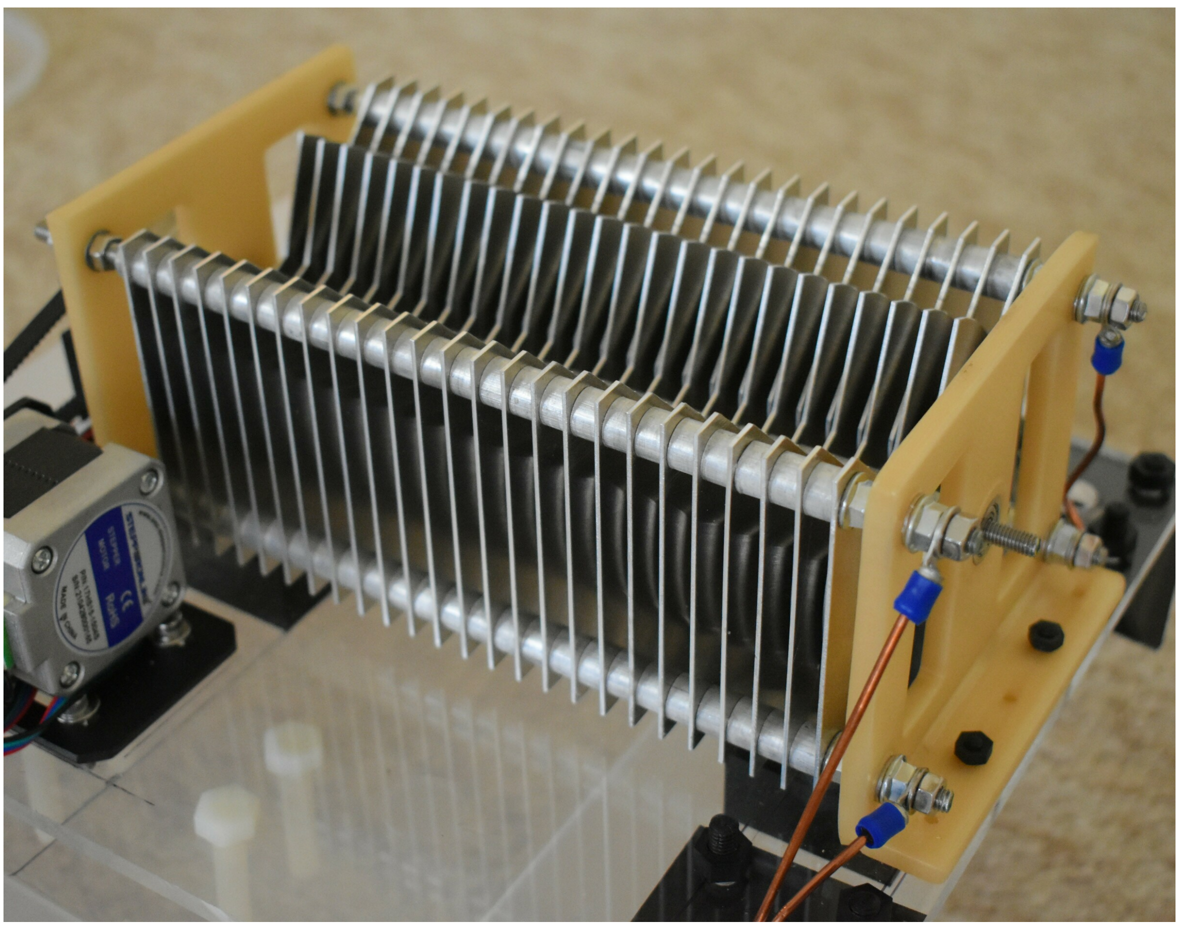


Figure 6.13: The "butterfly capacitor".

The butterfly capacitor can be ordered with more or less plates in order to adjust the capacitance range. This is critical in order to cover the desired amateur radio bands.

Online calculators were used to determine the number of plates required for operation from 40 to 20 meters.

The calculators will get close, but include a bit of margin, especially for the lowest frequency. In my case, the maximum number of plates was ordered.

6.4 Capacitor Protective Enclosure

An inexpensive plastic storage box was fitted onto the tuner assembly for some protection from moisture, insects, and animals. It is by no means waterproof, so don't leave it outside in bad weather!

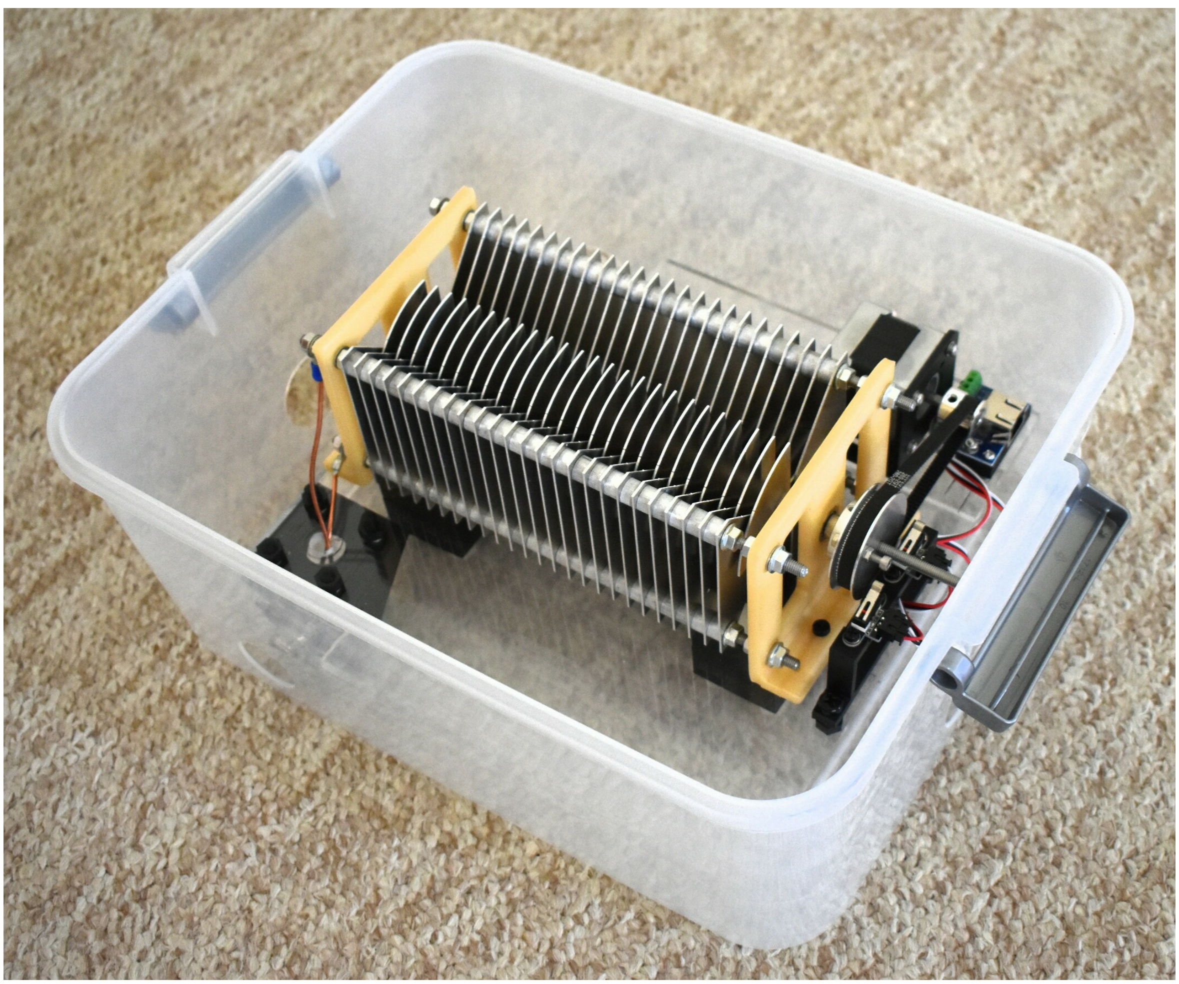


Figure 6.14: The tuner's plastic enclosure without top.

Figure 6.15: The tuner's plastic enclosure with top.

Chapter 7

Using Ethernet Cable for Stepper Motor Connection

The controller works by rotating a stepper motor connected to a tuning capacitor. As the antenna approaches resonance, there is a sharp “dip” in the SWR, which is continually monitored by the controller via the antenna coaxial cable.

Thus it is required to drive a stepper motor from the controller to whatever remote location the antenna is located.

The challenge with this is that the stepper motor drive is a series of sharp pulses! This means the cable connecting the stepper motor to the controller must not have excessive resistance, parasitic inductance, and capacitance. Fortunately, the pulses are very low frequency so it does not have to be perfect. I was skeptical at first that such remote operation of a stepper would work reliably, but this scheme has proven robust enough to rotate the variable capacitor. The fact that this is a low-torque application of a stepper motor helps make it possible.

In addition to the four wires for the stepper motor, there must be wires for two “limit switches”. There is a “low” switch and a “high” switch. Both of these switches are used to protect from over-rotating the capacitor. More importantly, the low switch is important for calibration of the system. The controller will automatically “zero” out the stepper when the controller is powered on. The zero step point coincides with the fully-meshed, maximum capacitance setting of the variable capacitor. All further operations of the controller are dependent on the zero step point being accurately established at power on.

A major problem can occur as the limit switch wires are bundled in the same cable as the stepper wires. Stray capacitance in the wires causes the sharp pulses to be coupled to the low and high limit switch wires, which are connected to the Pi Pico’s GPIs. This will cause “falsing” of the limit switch detect, and the result is erratic operation of the controller.

At first, I was using cheap utility wire from the local home improvement store. The pulses on the limit switch wires were extreme, so I decided to consult with other users of the controller who frequent the Software Controller Ham Radio group (groups.io). Some were using capacitors on the limit switches, and others were using Ethernet cables. The Ethernet cable idea is suggested in the book, but somehow I didn't notice that.

After looking at the various forms of Ethernet cable, I decided to try Cat 7. This has shielded twisted pairs, which will minimize cross-talk. There is also a shield around the entire cable assembly which can be grounded.

There are a total of four twisted pairs. This is perfect, as the stepper motor signals can be connected to two of the twisted pairs, and the limit switches can be connected to the other two.

Note that the shield around the entire cable assembly should be grounded. This requires an Ethernet connector which makes this connection. This connector must be carefully selected.

The stepper driver module is located adjacent to the Ethernet connector to allow short and simple routing. The Ethernet cable plugs into the back of the controller.

Note that shunt capacitors were added to the board in case of limit switch problems. I haven't detected problems so far, however, my Ethernet cable is only 50 feet in length. Others have reported successful operation with cables of 100 feet.

An Ethernet connector to screw terminal adapter board is used at the antenna. This board is available from Amazon or eBay. The BOM has a link to this board. An example 50 foot length Cat7 cable is also in the BOM. The cable is claimed to be waterproof and outdoor rated.

Chapter 8

Controller Enclosure

The enclosure components are 3D printed. I'm a beginner in 3D printing, so what I did was print out some of the components of the original controller enclosure, which were published as STL files. I have adapted these designs for this new controller, and published the design files in FreeCad and STL format. The builder should probably optimize these files for their particular printer using a "slicer" application like Cura.

I have used nylon hardware for assembling the enclosure, although typical steel or brass hardware should be OK. There is a good selection of nylon hardware in a kit available from Amazon. A link to this kit is in the BOM.

The enclosure was adjusted for minimal convenient size to fit the PCB and display. The rear panel has several holes for the various connectors exiting the rear of the controller.

There are four holes in the base for attaching silicone feet. You will need to drill holes in the adhesive silicone feet to allow them to be attached with bolts. The adhesive is not strong enough for this application. The feet will not stay in place. The feet were purchased from Amazon. A link to them is in the BOM.

One problem with the style of enclosure construction is that the last piece is impossible to bolt on because you can't get your fingers inside a six-sided enclosure. There are two possibilities:

1. Leave the top unbolted. Insert the bolts, but they won't be secured. But they will keep the top cover from moving around.

2. Use "melt in" threaded inserts. These are commonly used in 3D printed parts. A threaded brass cylinder is melted into the plastic using a soldering iron with a special tip. I haven't tried this yet, but it is probably what I will do for the final assembly step.

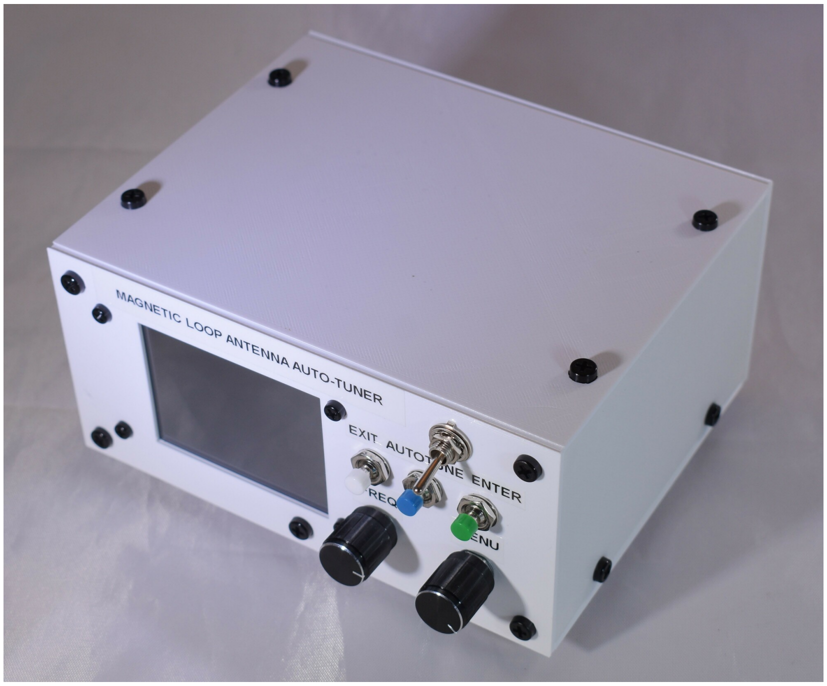


Figure 8.1: Front view of the controller enclosure.

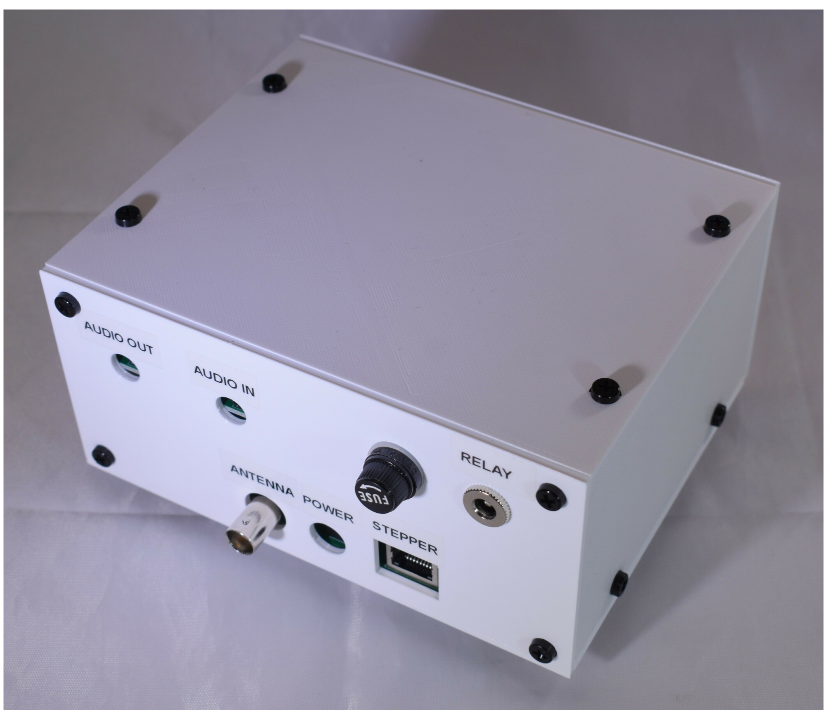


Figure 8.2: Back side view of the controller enclosure.

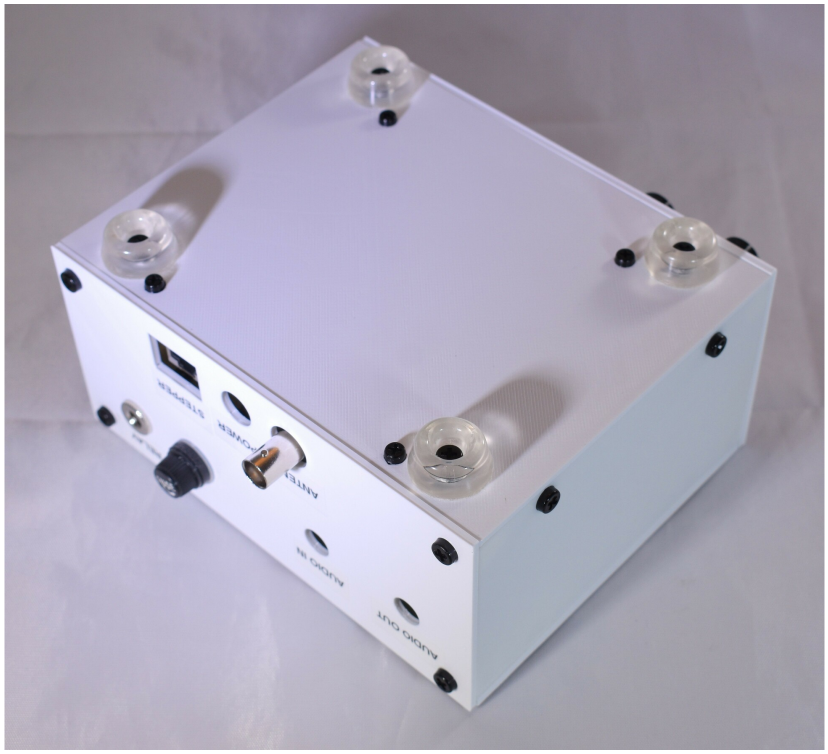


Figure 8.3: Bottom view of the controller enclosure.

The design files for the enclosure components are located in the magloop_pico_enclosure repository.

Chapter 9

User Interface

The user interface is both simplified and expanded compared to the original project. The original project includes a total of seven pushbuttons: five discrete pushbuttons, and two buttons integrated into the encoders.

The force required to push the encoder buttons is enough to make the controller slide around. Those were the first priority for elimination.

Changing the menu system to use “Enter” and “Exit” buttons eliminated two more buttons. The only remaining dedicated button is “AutoTune”, which is central to the mission of the controller.

A “menu function map” follows.

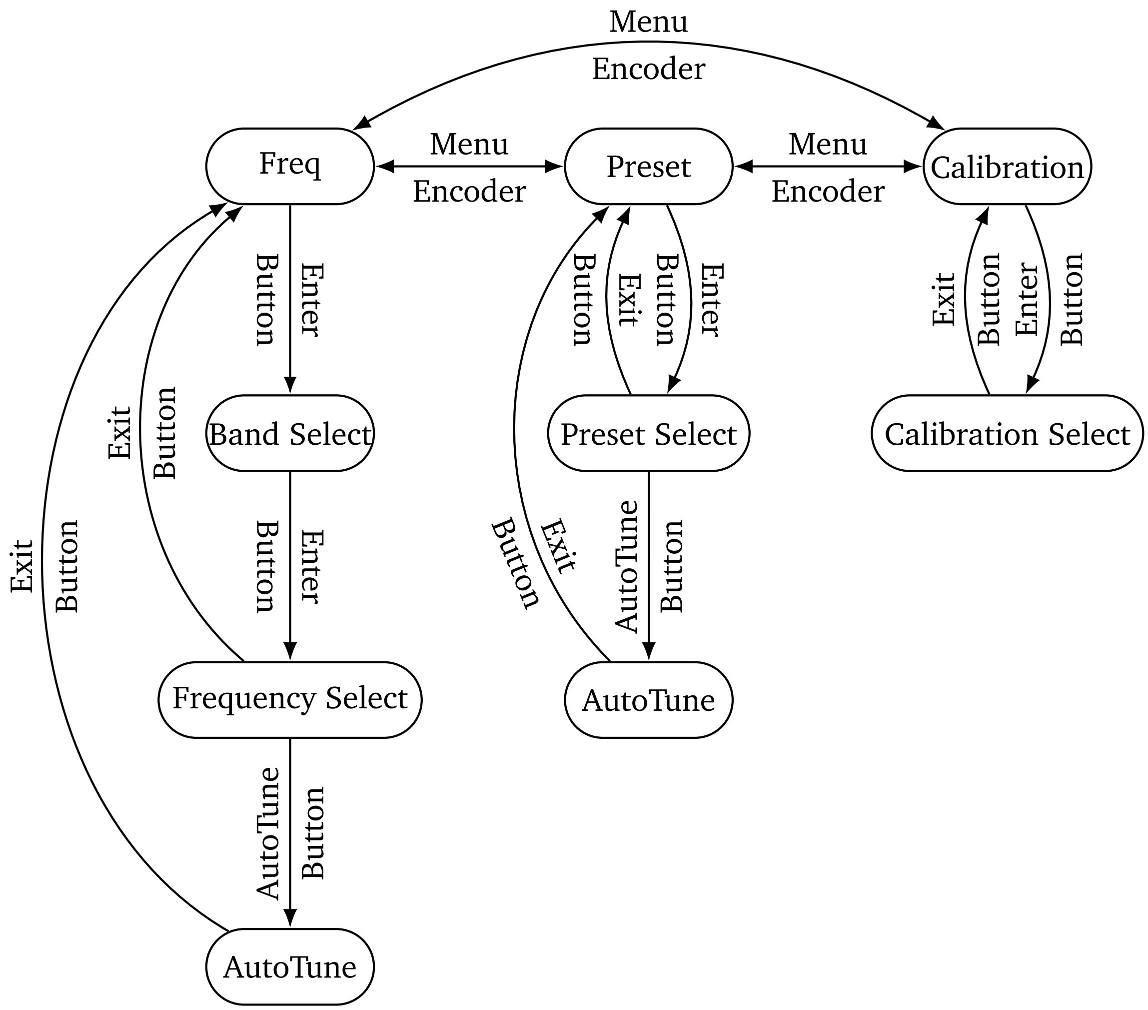


Figure 9.1: Magnetic Loop Antenna Controller Menu Function Map

Chapter 10

Speaker Audio Muting

One annoying problem became apparent during system testing of the controller with antenna and radio attached. The architecture of this controller is that it provides the RF signal for measuring SWR during auto-tuning. This is the purpose of the AD9850 DDS module.

However, this large signal is going to be heard in the receiver if it is tuned to the desired frequency! It is a very loud tone, and although brief, it is very irritating.

The solution to this problem is simple, but it requires additional circuitry. There was not enough room to implement this on the controller board, so an auxiliary PCB was designed to take care of this function.

The circuit functions by simply muting the receiver audio during auto-tuning. The audio muting PCB is designed so that it can be built as a "line level" muting circuit or a simple electromagnetic relay to connect-disconnect the speaker output.

I didn't know the simplest way to mute audio (besides a relay), but an internet search yielded this web page:

https://www.sound-au.com/articles/muting.html

The line-level muting circuit is based on information from this web page. The simple relay circuit is sort of intertwined with the line-level circuit, but it should be easy to see how to build the board depending on the user's preference.

The board requires two simple two-wire cables to connect to the controller board. Because these connections have power and ground, the connectors had to be polarized. JST type connectors were used.

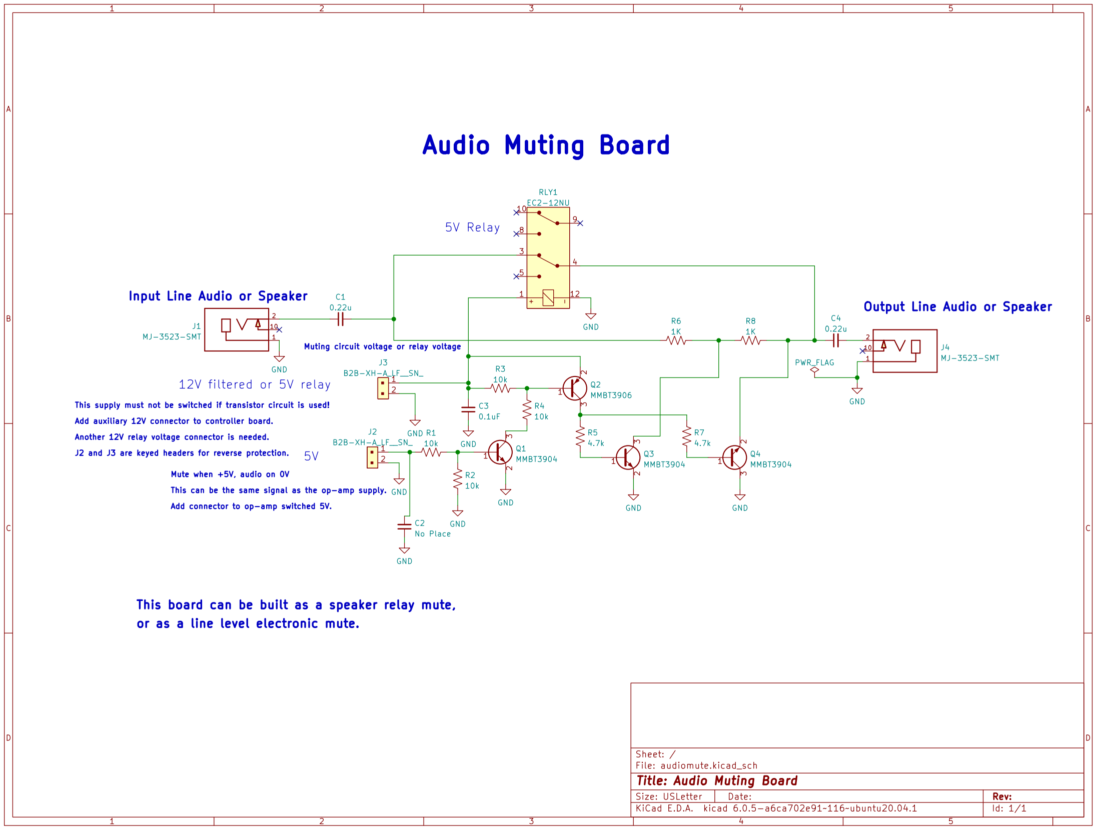


Figure 10.1: Audio Muting Board Schematic

Chapter 11

Construction

In this chapter I will throw out a few ideas to make the assembly of the controller faster and easier.

11.1 PCB Assembly

First, get all of the parts ready and reasonably sorted. Get all of the surface-mount resistors and capacitors into a bin, semiconductors in a bin, the leaded parts in another bin, the modules in another, etcetera.

It is a good idea to assemble everything on a grounded static mat. I did not adhere to this 100% of the time and nothing has failed yet, although latent defects can show up later. Just try to keep from zapping your parts, OK?

In general, you should start with the lowest height parts and then the next height, and the next, etcetera.

Start with the passive surface-mount parts. These are all very low height on the board, and it is easier to solder them without taller leaded parts obstructing the soldering iron. Get all of the surface-mount resistors and capacitors soldered to the PCB. Use a good clean fine-point soldering iron time, and fine solder.

Up next are the transistors and diodes. Be especially careful with the Schottky diode which routes power to the Pi Pico. The cathode marking is hard to see, so you might want to check with the diode function on a VTVM to make sure you are orienting the diode in the correct direction. The SOT-23 NPN and PMOS transistors are marked on the PCB. There is also a SOT-23 3.0 volt regulator device which is located underneath the Pi Pico.

Next, solder the four (DO35) Schottky detector diodes, making sure to insert with the correct polarity ("K" on the PCB means cathode).

Next are the through-hole LM358 operational amplifiers and DIP SW1.

Next are the JST connectors. Be sure to make the orientation of these connectors consistent with respect to the ground pin. These cables carry supply voltage, so do not cross-wire them to ground!

Next, there are six headers for the modules. Be sure to get these soldered in straight, as this makes insertion of the modules easier.

Next, the IDC connectors for the display, switches, and pushbuttons.

Next, the BNC and coaxial power connector. The BNC connector has a lot of thermal mass, so you should change to a larger soldering iron tip. You may need to use a larger soldering iron, or use two soldering irons.

Next, the four leaded electrolytic capacitors, with care to insert them with the correct polarity.

Next, the Ethernet connector, and make sure the ground tabs are properly soldered.

Next, the LM7805 regulator, making sure it is inserted with the correct orientation. Add an aluminum heatsink to the LM7805.

11.2 Ribbon Cables

images of latest and greatest ribbon cables here

11.3 JST Cables

There are two JST cables. JST was used because they are polarized connectors. Thus it is important to carefully construct them in order to avoid short circuits.

You can search and find youtube videos which show how to use a wire stripper and the crimping tool to make the cables. It does take a little bit of practice to get the stripped wire length just right. But once learned you will have a skill you can apply to other projects.

Figure 11.1: A typical crimping tool which can be used to make JST cables.

11.4 Power and Fuse Wiring

The coaxial power plug, fuse, and power switch are wired in series. The wires to the power switch on the front panel can be tucked underneath the PCB for neatness.

Don't forget to install a 1 amp fuse in the fuse housing.

Appendix

11.5 Github Repositories

The project requires a total of six Github repositories:

1. magloop_pico_project. Documentation and five submodules.
2. magloop_pico_pcb. Controller PCB design files.
3. magloop_pico_antenna. 3D printer files for antenna components.
4. magloop_pico_enclosure. 3D printer files for the controller enclosure.
5. magloop_pico_code. C++ code for the controller.
6. magloop_pico_audiomute. PCB files for the audio mute board.

The entire project can be downloaded with a few Git commands as follows:

```
git clone https://github.com/Greg-R/magloop_pico_project
cd magloop_pico_project
git submodule update --init
```

The PDF documentation is located in magloop_pico_project. Five other folders in the same directory complete the list above.

To obtain the latest changes to any or all repositories, use these commands:

```
cd magloop_pico_project
git pull
git submodule update
```

The submodules can be cloned individually if desired. Here are the URLs:

- https://github.com/Greg-R/magloop_pico_pcb
- https://github.com/Greg-R/magloop_pico_antenna
- https://github.com/Greg-R/magloop_pico_enclosure
- https://github.com/Greg-R/magloop_pico_code
- https://github.com/Greg-R/magloop_pico_audiomute

BUILD YOUR OWN COMPACT MAGNETIC LOOP ANTENNA FOR THE HF AMATEUR RADIO BANDS!

A "Magnetic Loop Antenna" is a compact and portable antenna for the High Frequency Amateur Radio bands. This is an effective antenna for those who have limited space to deploy a large HF antenna or for use on portable expeditions.

This book provides the information you need to build your own controller and antenna. Commercially available equivalents cost hundreds or thousands of dollars. Using your own tools and equipment, you can manfacture your own loop antenna system for much less.

This design is based on a project from the book "Microcontroller Projects for Amateur Radio" by Al Peter and Jack Purdum. This new project uses the Raspberry Pi Pico microcontroller, which can be purchased for $4.

The controller enclosure and some antenna components are manufactured using a 3D printer.

All design files are available in a Github repository.