Steve Harvey -
VE3EZB

Building a WiFi enabled Remote Antenna Tuner

Situation

VE3OJN (Mel) has an 80/160m vertical antenna in a field beside his home. This antenna requires the use of an Antenna Tuning Unit (ATU) at the base of the antenna to match the antenna to the feed line. In this case, the ATU is nothing more than a couple of inductors (one for each band) and a relay to switch between the two inductors, all installed in a box at the base of the antenna. The 12v dc power required for the relay is provided from a power supply located in the shack and fed via a ~36m (120ft) cable buried under the driveway and lawn.

The Problem

For the 80m band, the inductor (the coil on the left in Figure 5) can be tuned by soldering a tap to the inductor at a specific point. This setup should remain unchanged unless there are modifications made to the feedline or radials. However, at 160m, in order to cover the entire band, the antenna requires periodic "re-matching" or "re-tuning" when the operating frequency shifts across the band. Here lies the challenge. Using a manually adjusted inductor via the solder tap method becomes impractical, as it necessitates constant readjustment as the operating frequency is moved up/down the band.

Ideas

The basic solution was a roller inductor but how to tune it from the shack? Having someone at the antenna to manually rotate the inductor until SWR is satisfactory is not very practical. Running new wire and installing a motor and end stop switches to adjust the inductor from the shack was certainly a workable idea but that involved tearing up the driveway and lawn to install more cables. Not ideal because the RF transmission cable (LDF-4 1/2" heliax) and the 12v dc relay power as well as the many ground radials buried around the antenna and digging another trench for a new wire, risked damaging the existing cable.

As hams, we live in the world of wireless (it's kinda our thing) so why not a wireless solution. I like to tinker with and program Arduinos and other "System on a Chip" (SoC) microprocessors, and I had a few in my "tickle trunk" 'cause you never know when you will need one. Those of you who are of a particular age will get the reference to the tickle trunk - that thing had everything.

The Components

For the microprocessor, I decided on the NodeMCU 12E (figure 1) which comes with a built-in ESP8266 WiFi capable SoC all on one board. I needed a stepper motor, so I chose the Nema-17 stepper motor. (figure 6)

Figure 1 – NodeMCU 12E - notice the WiFi antenna on the left (squiggly line).

I also needed a motor driver to control the motor and the L298N Motor Drive Controller Board (figure 2) fit the job nicely and finally, rounding out the hardware list are a couple common micro switches. These micro switches will be used to detect when the roller has reached the end of the inductor, stopping the motor, preventing damage to the motor and/or the roller inductor. Now, with all the required items gathered, so began the project to control the roller inductor at the ATU via WiFi capable SoC and a stepper motor.

Now yes, I appreciate that you can purchase a commercially available tuner but those are quite pricey and as far as I am aware, there are no WiFi capable tuners, and besides, where is the fun in buying something when you can build it yourself.

Implement the Solution

The ATU needed to have the roller inductor and stepper motor installed (figure 6). This was taken care of by Mel - VE3OJN, leaving the end stop switches and the programming up to me. The ATU has a 12v dc source for the relay so that voltage source was used to power the L298N motor driver.

One of the nice things about the L298N motor driver is that it has an on-board 5v dc supply and since the NodeMCU 12E can operate on 5v dc, no extra wires are needed.

As far as I know, the NodeMCU 12E was not designed to operate in a higher-than-normal RF environment so it was decided to use the sole available wire in the ATU to connect it to a physical "Hard Reset" switch within the shack. This decision would prove to be beneficial as I'll explain later.

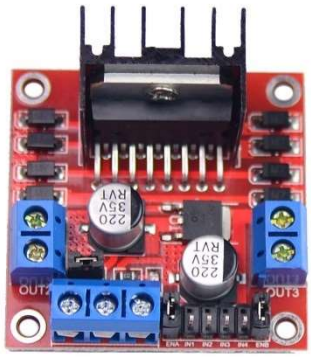

Figure 2 – L298N motor drive control board.

Wiring the Hardware

The micro switches were hot glued (figure 3) to the inductor roller to hold them in place and to ensure the switch pins did not contact the metal frame of the inductor. Easy.

The stepper motor was mounted in the box and coupled to the 160m inductor using an isolation coupler (Figure 11).

The diagram in Figure 4 shows how the components are connected. The motor controller (L298N) draws power from the 12V source, which also supplies power to the relay used to toggle between the two inductors and their corresponding bands. The NodeMCU receives power from the L298N's on-board 5v dc supply. The connection between NodeMCU pins D1-D4 (set as outputs) and L298N pins IN1-IN4 enables control over the stepper motor's movement.

A Ground (GND) pin from the NodeMCU is wired to one side of both microswitches, while Pins D5 and D6 (set as inputs) from the NodeMCU connect to the opposite sides of these microswitches. These microswitches serve as the clockwise (cwes) and counter-clockwise (ccwes) end-stops.

Each coil of the motor (A+ & A- coil and B+ & B-) are connected to the L298N motor controller. The A+

and A- coils connect to the L298N Out1 and Out2, while the B+ and B- coils connect to the L298N Out4 and Out3, respectively. It is important to note that the wiring configuration may be different for each

Figure 3 – Close-up of the roller inductor with the end stop switches hot-glued into place. What? It worked.

stepper motor so check the wiring diagram for your motor if you decided to build this.

Not shown in the diagram at figure 4 is the "Hard Reset" pin/wire. This wire is connected from the RST pin on the NodeMCU and connects to a momentary switch located back in the shack. The RST pin on the NodeMCU is the pin located between the GND and EN pin that you can see at the bottom right of Figure 1. When the button in the shack is pressed, it will connect the RST pin to the power supply ground and cause the NodeMCU to reset.

Writing the Code

Now, for coding (programming). For the coding platform, I chose the VS Code/PlatformIO environment. It has lots of nice features to help make coding easier. However, all the coding can be done using the Arduino IDE. I've used the Arduino IDE for many of my projects and even though the Arduino IDE doesn't have all the bells and whistles of other programming environments, it works well. If you are new to programming, I suggest you download the latest version of the Arduino IDE which you can find here. https://www.arduino.cc/en/software

Since this was to be a wireless solution, a webserver had to be set up. As mentioned earlier, the beauty of using the NodeMCU 12E is that it has a built-in ESP8266 that has WiFi and limited webserver capabilities. While the webserver capabilities of the ESP8266 may be limited, they are more than enough to handle this application comfortably. In the interest of brevity, I won't bore you with all the code design issues that needed to be managed but after several attempts, and some help from ChatGPT, I finally had a working code. I uploaded (flashed) the code to the NodeMCU that was connected to the L298N motor control board, crossed my fingers, and pointed my browser to the webserver on the ESP8266 … Success!

User interface

The software must be configured to the SSID and password of the wireless network that you want the Remote Tuner to connect to. Since the user interface is a simple webserver, it can be accessed by pointing the browser of your PC, laptop or smartphone to http://remote_tuner. [Note **http** and not https and note there is an **underscore** (not a space) between remote and tuner.] If you are connected to the same WiFi network as the tuner, you should see something like figure 8.

Pressing the CCW button will cause the motor to (as the name suggests) move counterclockwise, thus causing the roller of the inductor to move from the bottom towards the top of the inductor (figure 3); the green bar will move to the right to indicate the position of the roller and the numbers above the green bar will change to indicate the number of steps motor has taken which is a numeric indication of the position of the roller. If the CCW end stop microswitch is reached and activated, the CCW End Limit "LED" will turn RED (figure 10) and the motor

will stop moving. You will also get a message that the Motor STOPPED. The motor/inductor will now only move CW. Pressing the CW button will cause the green bar to move left, the numbers to count down and the motor/inductor will move clockwise until you press STOP or until the clockwise end stop is reached. The STOP button stops the motor once you are satisfied with the SWR. The Soft-Reset button will cause the NodeMCU to reset. (See comments about the Soft and Hard Reset elsewhere in this article.) The numbers you see above the green position bar are there to help quickly get back to the correct motor position for a particular frequency. Once the antenna is matched, you can record this number and the next time you want to use the same frequency, all you need to do is move the motor/inductor to the same number and after a quick SWR check (always check) you should be tuned.

Development, bench testing

As mentioned earlier, the decision to employ a “Hard Reset” for the NodeMCU would prove to be beneficial, and I'll explain how. Initially, my plan was to integrate a software-based reset using the “ESP.restart” function, which I did (Soft Reset). However, as is often the case, reality unfolded differently than anticipated. I discovered that rebooting the server using this “Soft Reset” method does not remove the power from the L298N motor drive controller board. The combined power draw of the stepper motor and controller board is approximately 800mA during movement and around 900mA when stopped. I believe the increased current when stationary is attributed to the “brake feature”, which, while useful in applications like 3D printing to maintain motor position, proves unnecessary in this context.

Extended operation in “brake-mode” results in heat buildup in both the motor and the control board. To tackle this issue and decrease the circuit's resting current, a “Hard Reset” is implemented. The “Hard Reset” switch, situated in the shack, is a simple toggle switch that connects the NodeMCU's reset switch to the power supply ground, causing the microcontroller to reset. Until an alternate solution can be found for this issue, the microcontroller must be “Hard Reset” after each tune is complete. Still better than going out to the antenna each time you want to re-tune. It's noteworthy that, despite the theoretical equivalence of the “Hard-Reset” and “Soft-Reset” pins, this specific application has proven otherwise.

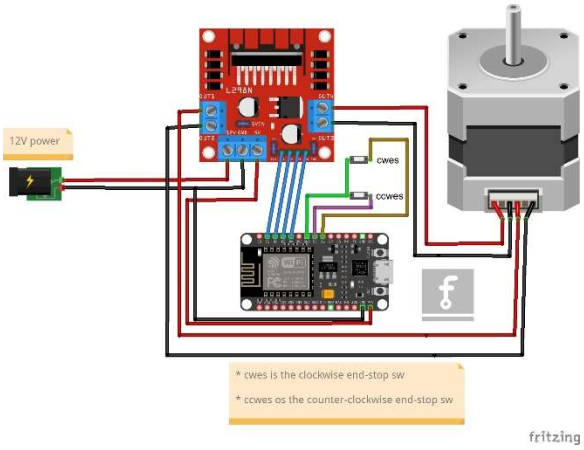

Figure 4 – Wiring diagram.

Field testing

Finally, the day came for the field test. Unfortunately, I couldn’t participate in the field test (stupid work … always interfering with my fun). Mel - VE3OJN took the ATU box and installed it at the base of the antenna (figure 5). He then connected to it via his WiFi network and tested to ensure the motor was moving as expected … it did. Test one complete. Next was the big test. From inside his shack, (where it was warm) Mel checked the tuning using his SWR meter and adjusted the inductor until the SWR was flat. Success. Another project in the books.

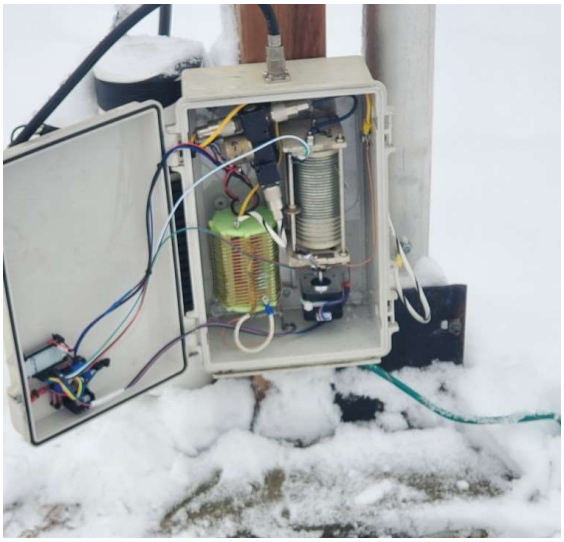

Figure 5 – ATU installed at the base of the antenna.

Figure 6 – Nema-17 stepper motor.

Figure 7 – ATU. On the top left-ish is the relay for switching in/out the 160m inductor. The 80m inductor can be seen on the bottom left and the 160m inductor on the right.

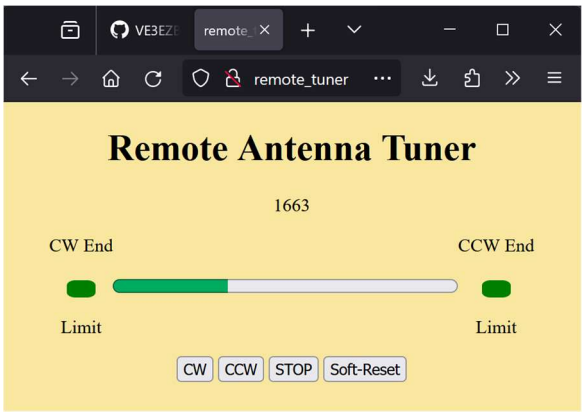

Figure 8 – This is a screenshot of what you see when you point your web browser to the ESP8266 server. (http://remote_tuner) The number above the "green bar" also indicates the position of the roller on the inductor and can be used to quickly tune/match the antenna.

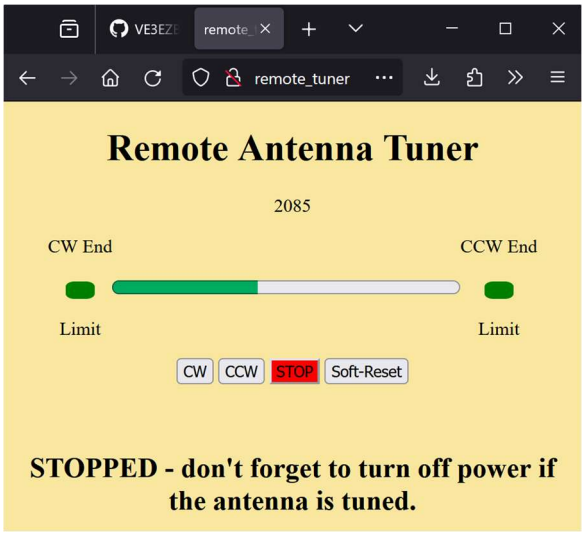

Figure 9 – Screenshot with STOP button pressed and end stop switches not reached.

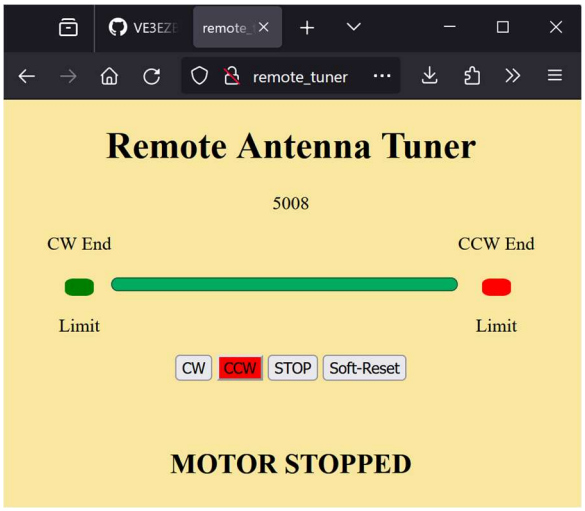

Figure 10 – Screenshot when the CCW End Stop has been reached. The "LED" will turn red to indicate that the CCW End Stop has been reached. The motor/roller will only move CW.

Follow-up

As expected, reality often differs from expectations, and this scenario is no different. When applying RF from the amp, even at a low output, the NodeMCU ceases to function during tuning. Although this isn't a problem once the circuit is tuned, it poses an inconvenience during the tuning process, requiring manual motor adjustments and checks. To address this issue, we propose to implement a Faraday cage-like shield around the NodeMCU, enclosing it on all sides except the one facing the house, where the controller's Wi-Fi antenna is situated. I'll provide an update if we are successful in getting this to work … if I remember :)

73—Steve—VE3EZB

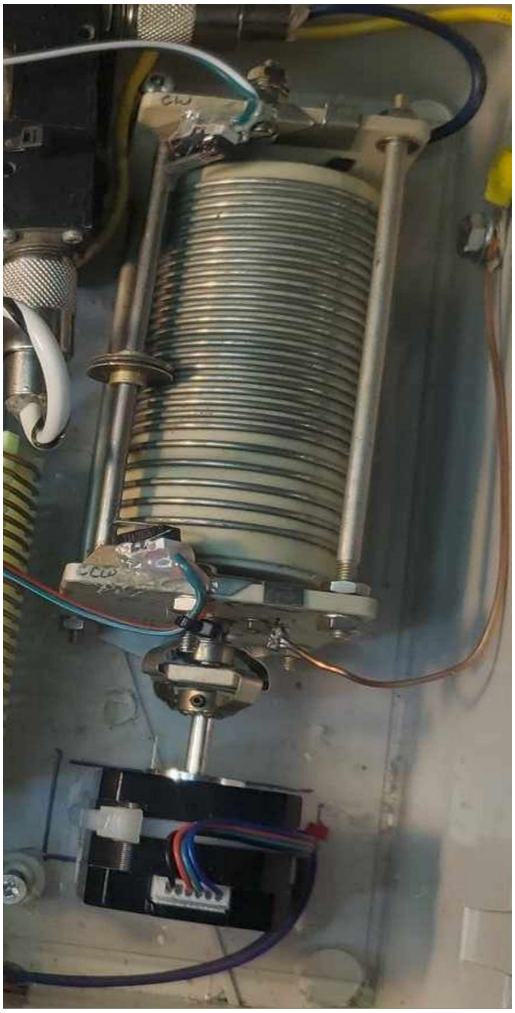

Figure 11 – Notice the isolation coupler now installed between the motor and the inductor.

Note: If you are interested in building this project, you can find the software files on Steve's Github at https://github.com/VE3EZB/Remote-AE-Tuner