Upgraded Software for the W8TEE/K2ZIA Antenna Analyzer - Version 3.8

User Manual

John Price – WA2FZW

License Information

Introduction

The W8TEE/K2ZIA antenna analyzer was originally developed as a club project for the Milford Amateur Radio Club. The original author of the software, Jack Purdum, published the design and code online on the Yahoo SoftwareControlledHamRadio group (which has now been moved to the SoftwareControlledHamRadio group on Groups.io). Jack also published an article about the project in the November 2017 issue of QST.

My main objective in modifying the software was to make it work on the 6 meter band (which requires replacing the AD9850 DDS with the higher frequency AD9851). In the process of going through the original code to figure out how to accomplish this, I did find a number of potential and actual bugs in the code (Definition: Working Software – Software with only undiscovered bugs) and came up with some enhancements to make it easier to use. Then I got a little carried away!

One might question why the initial release is Version 03.0. This software went through a couple months of testing and enhancement before it was deemed fit for public consumption.

Here, I will describe the operation of the analyzer using this software.

Acknowledgements

First of all, thanks to Jack (W8TEE) and Farrukh (K2ZIA) for the outstanding initial work on the project. Thanks to Edwin (PE1PWF) for the initial software work to add 6 meters and the modifications necessary to use the AD9851 DDS and for doing some really thorough analysis of the performance and investigation of ways to improve the hardware. Thanks to Jim (G3ZQC) and Dick (K2RH) for testing the new software as development progressed (it's still not finished). Thanks to Glenn (VK3PE) for his contribution of the AD8307 detector circuit. This greatly improves the accuracy of the SWR readings, and also for a lot of testing and debugging.

Bill Blackwell (AB1XB) contributed the modifications in Version 03.4 to allow the debugging code to be turned on or off via a menu option under the "Maintenance" menu.

The feature that allows users to skip over the band and frequency selection on startup was suggested on the SoftwareControlledHamRadio group by Dave (M0WID) along with the idea of displaying the band and frequency range information at the bottom of the menu screens.

George (KB1HFT) contributed modifications in Version 03.6 that calculate and log statistics on the readings taken during a scan.

Hardware Modifications

There are four optional hardware modifications that can be made to the unit to take full advantage of the capabilities of this software.

Replace the AD9850 DDS with the AD9851

The AD9851 DDS module is pin compatible with the AD9850 Type II module used in the original design. It is alleged to be capable of operating at higher frequencies than the AD9850. If you desire to use the analyzer on 6 meters, you should consider making the change.

The only real hardware change other than swapping out the board that needs to be made is to put a jumper between the anode of diode D5 and the cathode of diode D6 on the board. This increases the voltage to the DDS from around 3.5V to around 5V.

Instead of installing a permanent jumper on my own board, I ran the wires to a connector mounted on the edge of the board so a jumper plug can be installed or removed to undo the change. This really isn’t necessary, however, as it was later discovered that the AD9850 modules also seem perfectly happy running at 5 volts.

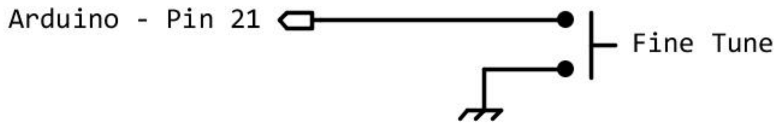
The “Fine Tune” Option

In the original software, the band limits could only be set in 100KHz steps and the fixed frequency for the single frequency SWR reading could only be adjusted in 5KHz steps.

With the addition of a push button, the adjustment increments can be stepped through 100KHz, 10KHz or 1KHZ. For the 6 meter and “Custom” bands which have much wider ranges than the HF bands, the frequency can also be tuned in 1MHz steps.

Starting with Version 03.6 of the software, the button serves dual purposes. Besides being used to change the tuning increments, it can also be used to toggle the DDS on and off on the “Frequency” and “SWR Calibrate” displays

To modify the hardware, another switch is required; a push button type connected between pin 21 of the Arduino and ground as shown here:



Wiring Diagram for Fine Tuning Button

If the modification is not installed, the software will function as it did in earlier versions, except that you will see a carat character (‘^’) displayed under the digit that will be changed when you rotate the encoder knob. If the modification is installed, you can move the carat with the button.

The button is enabled or disabled based on the definition of “FT_INSTALLED” in the “My_Analyzer.h” file. If the button is installed, un-comment the line reading:

```
          #define    FT_INSTALLED
```

If the button is not installed, the line should be commented out.

Better SWR Readings

Because of a number of factors in the original design of the hardware, although the analyzer will do a good job of showing you the frequency where the minimum SWR occurs, the accuracy of the actual numbers is not so good. We tried a few software solutions to improve the data with little success.

There are 2 modifications that can be made to improve the accuracy of the readings. The first, and easiest to implement is described in the document "PE1PWF improve on SWR readings 1224.doc" and just involves some fairly minor changes to the original hardware. This software supports his modifications if they are done exactly as described in his instructions. A number of people have partially implemented his changes and have had mixed results.

The 2nd approach which probably offers better results was developed by Glenn (VK3PE). He designed an add-on circuit board to replace the entire detector circuit, and later, a complete replacement for the main circuit board. The details can be found on the SoftwareControlledHamRadio group web page. His modification does require some software changes, which have been included in this software.

In Version 03.3, we found that smoother scan plots result if the analog reference voltage in the Arduino is set to 1.1V as opposed to the 2.56V reference previously used if the PE1PWF hardware modifications are installed (as per Edwin's document). If you have the VK3PE modifications installed, the reference voltage must be 5V, and if you have an unmodified unit (using either DDS), the reference voltage must be 2.56V. The software, as distributed includes these changes.

Battery Check Function

Version 03.8 provides the capability of including a "Battery Check" function. Details of how to enable it in the software can be found in the "Hacker's Guide".

In order to use the capability, a minor hardware change is required. As there are so many different ways folks are powering the analyzer and so many variations of the basic configuration, it is impossible for me to provide a standard modification.

But, what you need to add is something like a voltage divider with the center point connected to one of the analog pins on the Arduino. You have to make sure that the voltage on that pin NEVER exceeds 5V or you run the risk of damaging the Arduino. Maybe you could even use a zener diode as the bottom half of the divider. Make sure also that the resistor values you choose are large enough that the measuring circuit doesn’t add a significant current drain on the supply, particularly if using battery power.

You would probably also want to design the divider so that the low limit reading on the pin falls somewhere around the midpoint of the reading range on the pin (0 – 1023).

The instructions for modifying the associated software are in the “Hacker’s Guide”.

Software Changes in Version 3.8

I fixed 2 bugs.

One was that when doing an SWR calibration, the software was picking the midpoint of the default frequency limits for the selected band as opposed to those set by the user. This was not a major issue on the regular ham bands, but when using the “Custom” band, the calibration frequency would sometimes be outside the range selected by the user.

The second bug was in the code to set the lower band limit frequency. If the “FT” button had been used to set the cursor on the MHz position and the current frequency was less than 1 MHz and you tried to lower the frequency, the selected frequency got corrupted.

There are two new features in this release; The first is that now when using the “Repeat Scans” function, the current and maximum scan counts will be displayed at the top of the scan graph (where the “LIVE” label usually appears, if that is enabled) as “nn/nnn”. Note that the counts will be displayed regardless of whether or not the “LABEL_SCAN” symbol is defined in the header file or not.

The second modification was to add a prototype “BatteryCheck” function as described above.

Setting up The Arduino IDE

When I originally wrote the earlier versions of this manual, it didn't occur to me that this might be the first Arduino project for some people, and thus I didn't say too much about how to set up the Arduino IDE. I still won't do that myself, but the online document "Getting Started with Arduino and Genuino products" describes how to install the IDE and gives a decent overview of how to use it.

More than one person has contacted me about one the following compile errors:

```
'INTERNAL2V56' was not declared in this scope
'INTERNAL1V1' was not declared in this scope
```

These are caused by not having the Board type under the "Tools" menu in the IDE set to "Arduino/Genuino Mega or Mega 2560".

Installing Non-Standard Libraries

Many of the libraries needed for the analyzer software are built into the IDE when it is installed, however, there are four that are not part of the standard IDE, and must be added before this software will compile and load properly.

Those libraries and where to find them are:

Adafruit_GFX.h	https://github.com/adafruit/Adafruit-GFX-Library
AD985XSPI.h	Zip file is in the directory with this software
MCUFRIEND_kbv.h	https://github.com/prenticedavid/MCUFRIEND_kbv
Rotary.h	https://github.com/brianlow/Rotary

Make sure you use the versions listed here. There are libraries out there with similar names, but they are not compatible with this software.

The instructions for how to add them are also on the Arduino web site in the document "Installing Additional Arduino Libraries". The links are also found in the "My_Header.h" file included with this software.

Using the Arduino IDE Serial Monitor

Several of the functions in the analyzer make use of the Arduino IDE's serial monitor as an output device: "Plot>Serial", "EEPROM>Serial", "Debug", and, new in Version 03.6, logs prepared by optional statistics calculations.

As distributed, the baud rate for the serial monitor output is set to 9,600 in the program file, so the serial monitor should be set for that also. Having said that, however, if you plan to make use of the dynamic debugging feature added in Version 03.4, you might want to set the baud rate in the program and serial monitor to a much higher rate as at 9,600, it really slows down the scans. Also new in Version 03.6 is a menu pick under the Maintenance menu that allows you to change the analyzer's serial baud rate. If you change the analyzer's serial output baud rate, you must change it in the IDE's Serial Monitor's window. This is done via a dropdown in the Monitor window's lower right.

The serial monitor can be started by the key combination "ctrl-shift-m" pressed all at one time.

Installation on Your Computer

The program (why do they call them sketches?) consists of **4 files**; the ".ino" file (whatever its current name is), a header file, "My_Analyzer.h", and, new in Version 03.6, two files that implement statistics calculations and their logging: "AAStats.cpp" and "AAStats.h". **All four files must be installed in a directory with the same name as the .ino file. No other files should be in that directory.**

Due to the software changes, this new version will not work with files saved on the SD card that were created with the original (W8TEE) software, therefore, you should delete any old files before trying to read them with the new code.

Also, the structure of the data stored in the EEPROM has changed from the W8TEE versions and must be erased prior to installing this software. I have provided a separate "Erase_Eprom" program that *MUST* be used to accomplish the task before loading this software.

If you are using a virgin Arduino (i.e. one that has never had any software installed, you should also run the “Erase_Eprom” program before installing the analyzer software. All of the Arduinos that I’ve personally used came with the EEPROM initialized to all zeroes, however we’ve found that some come initialized to all ones or, worse random trash. Either of these later conditions will cause problems in the analyzer code.

If you’ve already loaded a previous version of this software, there is no need to erase the EEPROM, unless noted otherwise.

There is also a “Read_Eprom” program that will display the contents of the EEPROM on the Arduino IDE’s serial monitor. These functions are also built into the program however the built-in “Erase EEPROM” function does not reset the file sequence number in the EEPROM.

Note that the “Erase_Eprom.ino” and “Read_Eprom.ino” files have to be put in folders separate from the actual analyzer program file and header file. Those folders must be named “Erase_Eprom” and “Read_Eprom”. As noted above, the program file and header file must be in a folder with the same name as the program file (without the “.ino”).

Pre-Installation Edits

For All Users

Before you compile and install the software on your analyzer, there are some edits to be made in the “My_Analyzer.h” file. BTW, I sometimes use the program “Notepad++” for editing “.C” and “.h” files, although you can just as easily use the Arduino IDE.

There are 4 “#define” statements that determine which hardware options you have installed. You must select between the AD9850 DDS and the AD9851 DDS, and make sure the one for the DDS you are using is un-commented, and that the other one is commented out.

The 3rd definition (“#define” AD8307_SWR”) should be un-commented if you have Glenn’s AD8307 modification installed. Make sure it is commented out if the modification is not installed.

The 4th definition (“#define” PE1PWF_MOD”) should be un-commented if you are using Edwin’s modifications. Make sure it is commented out if the modification is not installed.

DO NOT enable both the “AD8307_SWR” and “PE1PWF_MOD” options. There is no telling what kind of havoc that will cause in the code!

Also associated with the DDS are settings for the default calibration constants for the AD9850 and AD9851 DDS modules. These are set to the standard settings of 125MHz for the AD9850 and 180MHz for the AD9851.

Normally, you don’t need to do anything with these however, we have found that there are some AD9851 modules that will not work correctly with the 180MHz setting. If you have one of these, setting the value of “CAL_9851” to 120MHz will usually fix the problem.

Another valid reason to change the value for whichever DDS module you are using is to set the default calibration constant for the specific DDS module you have installed in your hardware. When you do the “Freq Cal” function under the “Maintenance Menu” (*which you should do before attempting to use the analyzer*) you can change the default value to the calibration factor for your specific unit.

If you want to enable 6 meter operation, you may need to modify the line that reads:

```
#define ADD_6_METERS
```

If you want to use the analyzer on 6 meters un-comment this line. If you don’t want 6 meter operation, this line needs to be commented out. Note that 6 meter operation will only be enabled if the AD9851 is also enabled (although, you can fudge this).

You can also change the 6 meter band limits. I have them set for use from 50MHz to 54MHz now. If you want to change the limits change the numbers on the lines that read:

```
#define    LOW_6M_EDGE     50000U
#define    HIGH_6M_EDGE    54000U
```

Make sure to leave the “U” at the end of the numbers, or all kinds of strange behaviors will be observed!

There are edits that have to do with the capability to add a “Custom” band, which you can set for any frequency range you desire. To enable or disable this feature, comment or un-comment the line that reads:

```
#define ADD_CUSTOM
```

As was the case for the 6 meter band, you can modify the band limits by changing the values on the lines:

```
#define    LOW_C_EDGE         100U
#define    HIGH_C_EDGE      65500U
```

Again, make sure to leave the “U” at the end of the number. The limits are currently set to scan from 100KHz to 65.5MHz (changed in Version 03.2). Do not attempt to set the lower band edge to less than 100KHz and do not attempt to set the upper band edge to more than 65.5MHz

The lower frequency limit was changed so that the analyzer could also be used as a signal generator for doing things like tweaking IF chains.

Be aware that when you use this capability, you will be transmitting a low power signal across all frequencies in the set range, including frequencies that are not amateur radio frequencies.

There are a few other changes that you can (or may need to) make. One that might be needed is to flip-flop the definitions of “PINA” and “PINB”. If you find that your encoder seems to work backwards, reverse the definition.

Another set of edits that can be made is to alter the parameters for the “Repeat Scans” function.

There are currently six other definitions that can be enabled or disabled at your choosing:

```
#define FT_INSTALLED
#define AUTO_EXAMINE
#define HORIZ_INDEX
#define LABEL_SCANS
#define SKIP_BAND_SELECT
#define VIEW_PIN_DATA
```

These are explained in the comments in the header file and in the sections of this manual where they apply.

Once you have made the changes to the “.h” file and if upgrading from Jack’s original software, run the “Eprom_Erase” program, you can compile and load the program just as you would any other Arduino program.

If you’re updating from a prior release of this software, you need not run the “Eprom_Erase” program, unless it was so specified in the release announcement.

Edits for Non-USA Users

The frequency limits for each band are set in the software based on the USA’s frequency allocation chart. Other countries may have different allocations, and thus you should modify the frequency table for your country.

The band limit definitions are near the top of the “.ino” file:

```
uint32_t bandEdges[][2] = { {  1800,  2000 },    // 160 Meters
                            {  3500,  4000 },    //  80 Meters
                            {  5330,  5404 },    //  60 Meters
                            {  7000,  7300 },    //  40 Meters
```

I didn’t show the whole table here. The frequency definitions are in MHz divided by 1000. To change the upper or lower limit for any particular band, simply change the number. Be careful though to leave the commas and braces where they are, or you’re going to have problems compiling the code.

A Note About SWR Readings

Before we start the analyzer, please note one change that was made in Version 03.6 that applies throughout all of the functions. Because the “Frequency” and “SWR Calibrate” functions now allow the DDS to be turned off, some divide by zero issues showed up in some of the SWR calculations.

To deal with this, I had to add a bunch of tests for bogus SWR values.

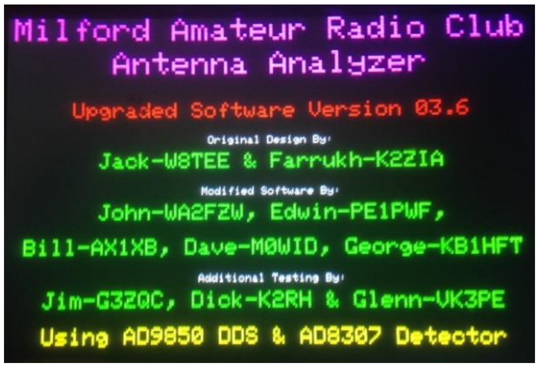
Effective with Version 03.6, if the calculated SWR is less than 1.0 (impossible) or greater than 20.0, instead of displaying the actual value, the SWR fields will contain “N/A”. This applies to any place where SWR values are shown.

Startup

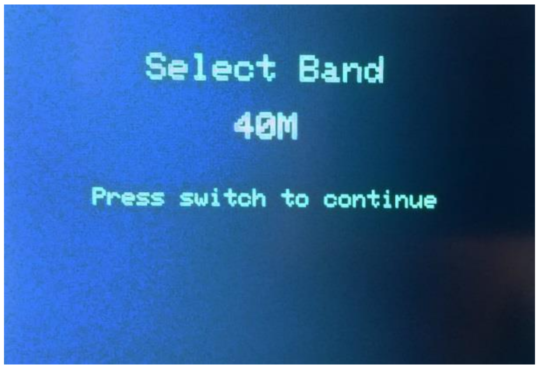
When you turn the analyzer on, you will be greeted by a screen showing all the credits for 3 seconds.

Note, at the bottom of the startup screen, it tells you which DDS you are using and whether or not you are using the VK3PE detector circuit or the PE1PWF modifications.

Also note, if you failed to enable one of the DDS types in the "My_Analyzer.h" file, an error message will start flashing on the display and program execution will stop.



Milford Amateur Radio Club
Antenna Analyzer
Upgraded Software Version 03.6
Original Design By:
Jack-W8TEE & Farrukh-K2ZIA
Modified Software By:
John-WA2FZW, Edwin-PE1PWF,
Bill-AX1XB, Dave-M0WID, George-KB1HFT
Additional Testing By:
Jim-G3ZQC, Dick-K2RH & Glenn-VK3PE
Using AD9850 DDS & AD8307 Detector



Select Band
40M
Press switch to continue

The startup screen will display for three seconds, after which, you may or may not be asked to select a band to be used.

The first time you turn the analyzer on with this software (or if you used the "Erase EEPROM" function) the "Select Band" screen will display "40M" in the selection box. After the analyzer has been used once, the band displayed will be the last one that you used.

If you enabled the definition of the "SKIP_BAND_SELECT" symbol in the header file, and there is valid band and frequency data already saved in the EEPROM, you will not be asked for this information and the next screen that will be displayed will be the main menu screen.

If asked for the operating band, you can rotate the encoder knob right or left to select a different band. Pushing the encoder switch will save the selection.

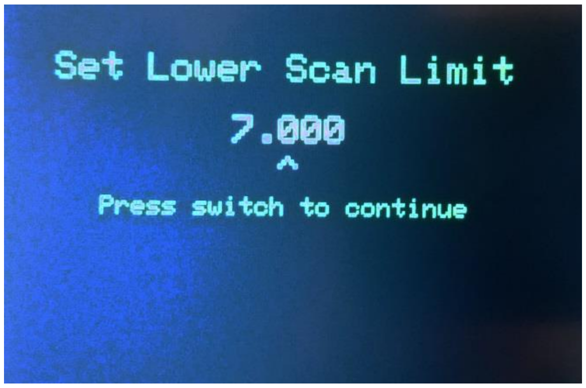
The next screen will ask you to “Set Lower Scan Limit”. The selection box will display the lower (US) frequency limit for the band selected in the previous step (or the limit you previously set for the selected band if you didn’t change the active band in the previous step). Again, rotating the encoder knob to the right or left will raise or lower the frequency in 100KHz steps if the “Fine Tune” hardware modification has not been made. If the “Fine Tune” option is installed, the frequency can be changed in 100KHz, 10KHz or 1KHz steps by using the “Fine Tune” push button to move the cursor. For 6 meters and the “Custom” band, the frequency may also be changed in 1MHz steps.

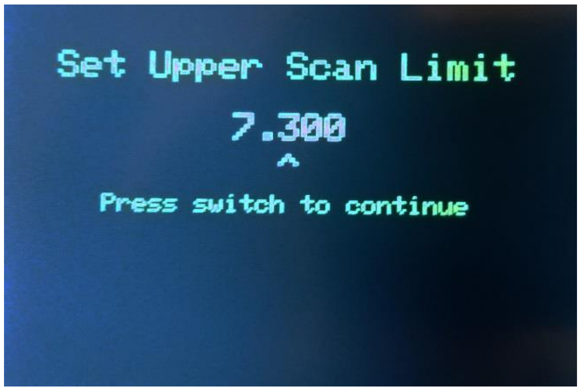
The software will allow you to set it equal to or less than the selected lower band limit, but it will not allow you to set it equal to or higher that the currently set “Upper Scan Limit”

Pushing the encoder switch will save the frequency.

The third screen asks you to “Set Upper Scan Frequency”. This works just like setting the low frequency except it won’t allow you to set it to less than or equal to the low frequency setting.

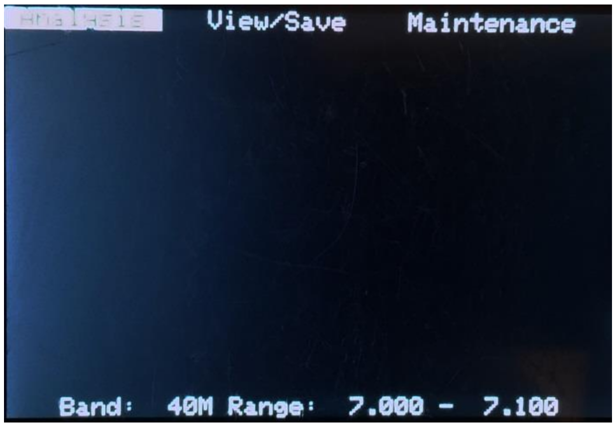
It’s also a good idea to not make the scan range less than 100KHz.


Set Lower Scan Limit
7.000
^
Press switch to continue


Set Upper Scan Limit
7.300
^
Press switch to continue

Main Menu

Once the band information has been set and saved (or if you configured the software to skip asking for it), the main menu will be displayed. Currently there are three items in the main menu, “Analysis”, “View/Save” and “Maintenance”. At the bottom of the screen the current selected band and scan frequency limits are displayed.



Rotating the encoder knob one way or the other will move the selection highlight and pushing the encoder button will select it.

In the following sections, we will discuss the selections under each heading.

Analysis Menu Items

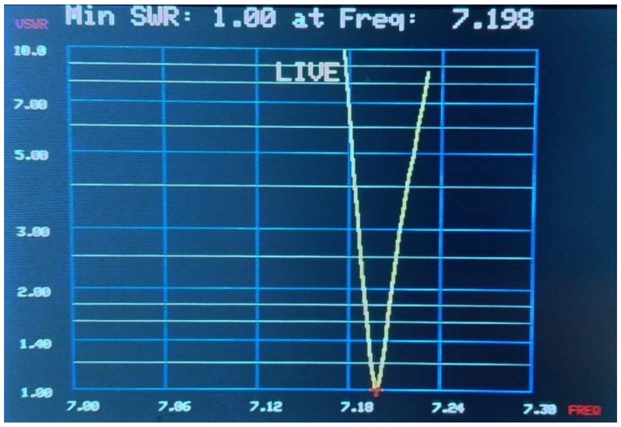
Currently, there are either five or six choices under the “Analysis” menu item (not counting the choice to go back to the “Main Menu”). The following sections will describe what each does.

Single Scan

This option will scan the antenna from the lower frequency limit previously set to the upper frequency previously set and produce a graph of the SWR versus frequency.

The SWR values are plotted on the vertical axis, and the frequencies on the horizontal axis. A red ‘+’ marks the minimum SWR on the graph, and the minimum SWR value and the frequency at which it occurred are shown in the heading. Each point on the graph represents the mean value of a sample of 75 readings (by default) and averaged to determine the point to be plotted.

This is a scan of my 40 meter magnetic loop done with the band limits set for 7.0MHz to 7.3MHz.



Notice, that we changed the vertical axis from a linear scale to a logarithmic scale, which is how SWR readings should really be plotted. This change also allows a maximum SWR reading of 10:1 to be plotted as opposed to the 3:1 limit in the original software.

Also, in Version 03.2 we added a label. “LIVE” shows the plot is from a real-time scan that was just done. The labels can be turned on and off by changing the definition of “LABEL_SCANS” in the “My_Analyzer.h” file.

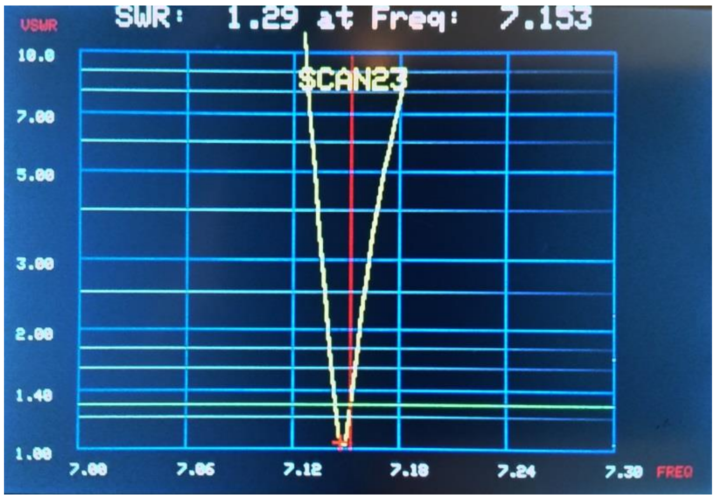
If the symbol “AUTO_EXAMINE” is defined in the “My_Analyzer.h” file, once the scan has completed, a vertical red line will appear at the minimum SWR point on the graph. As shown in the picture below, in Version 03.5, an optional horizontal cursor line was added.

The horizontal cursor can be enabled or disabled based on the definition of the symbol “HORIZ_INDEX” in the header file.

If enabled, the horizontal cursor will display as a green line if the SWR at the point being examined is less than 1.5:1; a yellow line if the SWR is less than 2:1 and a red line if the SWR is 2:1 or greater.

Moving the encoder will move the line(s) and show the SWR and frequency at the point where the line(s) cross the curve. If the “AUTO_EXAMINE” symbol is not defined (the line commented out in the header file), the cursor(s) can be brought up by simply moving the encoder one click in either direction.

The same thing can be done after using the “Repeat Scans” or “View Plot” options.



Pushing the encoder switch will return you to the main menu. If you don’t want to use the “Examine” feature, simply pushing the encoder switch upon completion of the scan will take you back to the main menu.

Statistics Collection & Output

New in Version 03.6, thanks to George (KB1HFT), are features that log basic statistics on the scan readings to the IDE’s Serial Monitor.

This quantitative data can be useful when debugging your hardware or software modifications. It is especially useful for comparing the effects of different hardware & software configurations on induced or mitigated noise that might affect the SWR readings.

Key among the calculated statistics are the standard deviations of the sample readings taken for plotting each scan point. Standard deviations provide a quantitative measure of the distribution of values in a set of sample readings.

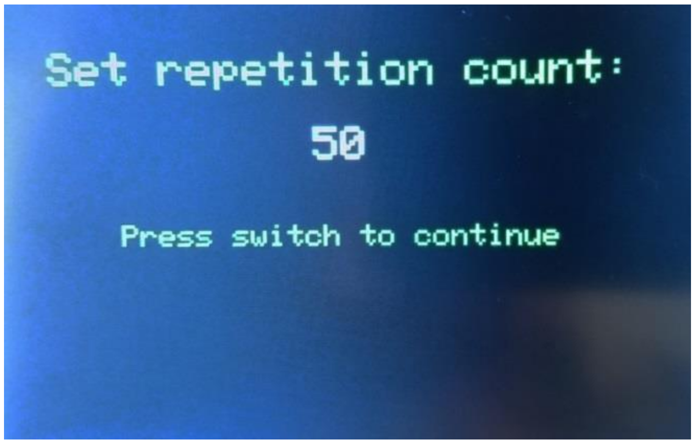
> From Wikipedia: “In statistics, the sample standard deviation (represented by the Greek letter sigma, σ) is a measure that is used to quantify the amount of variation or dispersion of a set of data values. A low standard deviation indicates that the data points tend to be close to the mean of the set, while a high standard deviation indicates that the data points are spread out over a wider range of values.”
>
> The sample standard deviation is defined as:
>
> $$\sigma = \sqrt{\frac{\sum_{n=1}^{n=max}(s_n - \bar{s})^2}{n-1}}$$
>
> Where s_n is the nth sample, and $\bar{s}$ is the mean of all the samples in the group of readings. If the distribution is a “normal” or “bell shaped” distribution, 99.9+% of the samples in the set will fall within +/- 6 sigmas from the mean value.

To use the new statistics logging features, attach a resistive load, and set Maintenance >Options, and select the Logging level desired.

Logging levels are hierarchical: for a level to activate, the level above it must be activated (although the UI in Version 03.6 should be tweaked to reflect this). Activating logging levels is explained in more detail under “Options” later in this document.

Repeat Scans

Added in Version 02.8 is the ability to “Repeat Scans” any number of times you like (up to a maximum of 100 times). When used, the function will ask you to set the number of repetitions to run.



Set repetition count:

50

Press switch to continue

The first time after startup, the default will be set to 50. The number can be adjusted from 10 to 100 in steps of 10 by rotating the encoder. On subsequent requests, the initial setting will be the same as the last time you used the function (since turning the analyzer on).

Once the number of cycles has been set, the analyzer will proceed to start scanning. The scans are not continuous; there is a 1 second pause between scans to allow you to read the header.

Pressing the encoder button during a scan will cause the function to terminate upon completion of the current scan. After the last scan, the encoder button must be pressed again to return to the main menu just like when you do a “Single Scan”, or moving the encoder knob will activate the “Examine” feature just as can be done after a “Single Scan”.

The “Save Scan” option can be used after the last scan completes. The scan saved will be the last one run.

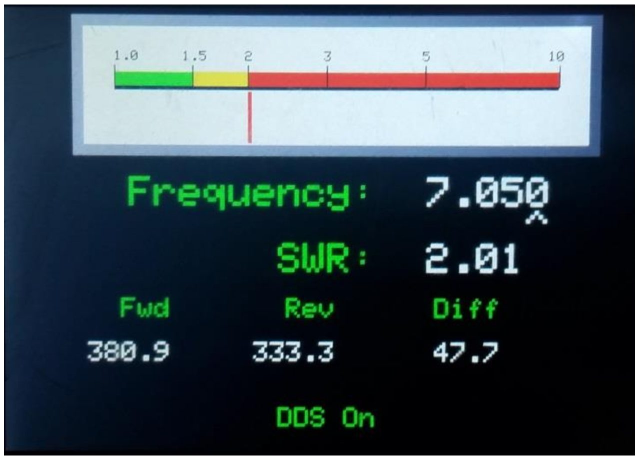
The pause time and other related parameters that control the allowed number of cycles are in the “My_Analyzer.h” file and can be modified if you desire.

Frequency

This selection allows you to measure the SWR at specific frequencies. The display will show the frequency being tested and the SWR for that frequency, and we even added an “analog” SWR meter to the display!

The starting frequency is set to the mid-point of saved frequency range.

Rotating the encoder knob one way or the other will change the frequency up or down in 5KHz steps if the “Fine Tune” hardware modification is not installed. The frequency can be moved past either of the pre-established frequency range settings, but not past the band edges. Note, however that you can use the “Custom” band to set up any range of frequencies you want to look at from 100KHz to 65.5MHz.



```
Frequency: 7.050
      SWR: 2.01
Fwd      Rev      Diff
380.9    333.3    47.7
        DDS On
```

This function underwent some major changes in Version 03.6. Most notably, you can now optionally display the actual Arduino forward and reverse pin readings and the difference between them. This can be enabled or disabled based on the definition of the symbol “VIEW_PIN_DATA” in the header file.

You will also notice there is a notation at the bottom of the screen that the DDS is on.

The “Fine Tune” button now performs two functions on this screen. A long push (more than 1 second) of the button will toggle the DDS on and off. You might ask why you would want to do that. At least one person was having trouble getting accurate SWR readings on his antenna. The problem was discovered to be as a result of a very strong signal nearby that was overcoming the reverse signal from the DDS.

If you are using an unmodified version of the analyzer, or if you have the PE1PWF modifications installed, when you turn the DDS off, you should see readings of zero on both the forward and reverse pins. If you still see a reverse reading, the analyzer is picking up a signal from somewhere else.

Unfortunately, this doesn’t work with the VK3PE board, as the readings never go to zero.

The “Fine Tune” button can be used to cycle through which digit in the number will be changed when the encoder is moved. The digit that will change when you operate the encoder is denoted by a carat character (‘^’) under the digit. Short pushes (less than ½ second) of the “Fine Tune” button will cycle the carat through the digits that you are allowed to change.

If the carat is under the 1KHz digit, moving the encoder will change the frequency in 1KHz steps if the “Fine Tuning” option is installed and 5KHz if the “Fine Tuning” option is not installed (this can be adjusted by changing the definition of “FIXED_FREQ_INCR” in the “.ino” file).

Change Band

This option allows the operator to change to a different band and set the frequency range exactly like was done at startup.

Set Limits

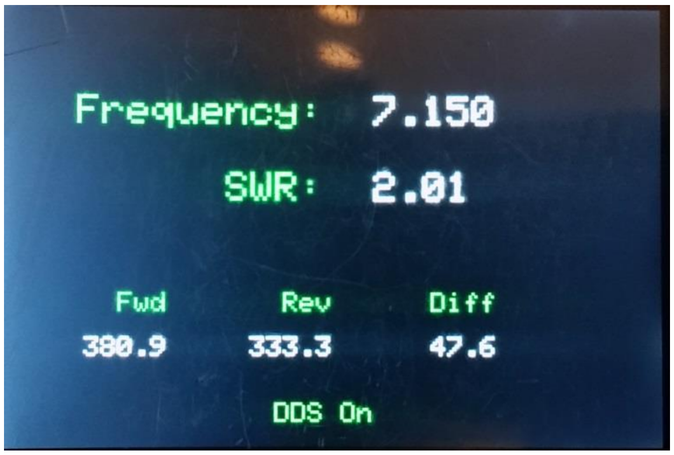
This option allows the operator to change the lower and upper frequency limits without changing the band. It works just like the frequency selection process done during startup.

If the “Fine Tune” option is installed in the hardware, short pushes (less than ½ second) of the “Fine Tune” button can be used to cycle through which digit in the number will be changed when the encoder is moved. The digit that will change is denoted by a carat character (‘^’) under the digit.

SWR Calibrate

IN version 03.6, the “SWR Calibrate” function is available for analyzers with the PE1PWF modifications installed and those using the VK3PE boards. Unfortunately, we still haven’t come up with a good way of calibrating the unmodified version of the analyzer.

The display looks like the “Frequency” display without the SWR meter.


Frequency: 7.150
SWR: 2.01

Fwd	Rev	Diff
380.9	333.3	47.6

DDS On

Although the screens for the PE1PWF and VK3PE versions look the same, the calibration process is a little different.

To use this function for the PE1PWF version, you need to connect a known value resistor to the antenna pins on the analyzer card with very short leads or to the antenna connector. Good choices are 33Ω (SWR should be 1.52:1), 100Ω (SWR should be 2:1) or 150Ω (SWR should be 3:1). *DO NOT* attempt to calibrate the SWR with a 50Ω resistor or 50Ω dummy load; it won’t work!

With the resistor connected, rotate the encoder knob until you get the appropriate reading for the resistor you are using. Pushing the encoder switch will save the calibration factor in the EEPROM and it will be applied to all future SWR computations.

That part of the process is the same for the VK3PE version, however you have to first use the trimpot on the board to balance the AD8397 logarithmic amplifiers. That can be done using a voltmeter and test points on the circuit board, or by making the adjustment so that the forward and reverse readings on the display are equal with nothing connected to the antenna connector.

Once the amplifiers are balanced, connect a known value resistor to the antenna connector and adjust the encoder until the correct SWR shows on the display. Glenn suggests using one that will give you a high SWR reading such as 470Ω, which should result in a 9.4 to 1 SWR.

Note that as was the case with the "Frequency" function, the display of the pin data is optional based on definition of the symbol "VIEW_PIN_DATA" in the header file.

The View/Save Menu

The selections under the "View/Save" menu allow one to save scan data on the SD card or to view the saved data in a number of ways.

Save Scan

This option will save the scan range, and the SWR/frequency pairs for the most recent scan in a file on the SD card.

The files are all named "SCANnn.CSV", where the sequence number "nn" is assigned by the software. Note that the maximum file sequence number is 99. If more than 100 files are created, the sequence number will be reset to zero, and files will be overwritten on the SD card.

The files are "comma separated variable" (CSV) files suitable for reading into Microsoft Excel or other spread-sheet programs.

If no scan has been run since the analyzer was turned on, the operator will see an error message indicating that there is no active scan, and thus, no new file will be created. However, it should be noted that if you used the "View Plot" or "View Table" options, there will be an active scan in memory which you can save. Of course, you will now have 2 different files with the same data!

View Plot

This selection will display a list of the saved scan files on the SD card (up to 20) and allow the operator to use the encoder to select one.

The saved data will be displayed in graphical format exactly as it was displayed when the scan was first performed (picture above), except that the label will show the name of the file from which the plot was loaded. Again, the labels can be turned on or off by changing the definition of “LABEL_SCANS” in the header file.

If the symbol “AUTO_EXAMINE” is defined in the “My_Analyzer.h” file, once the scan has completed, a vertical red line will appear at the minimum SWR point on the graph. If the symbol “HORIZ_INDEX” is defined, a horizontal cursor will also appear.

Moving the encoder will move the line and show the SWR and frequency at the point where the cursor(s) cross the curve. If the “AUTO_EXAMINE” symbol is not defined (the line commented out in the header file), the cursor(s) can be brought up by simply moving the encoder one click in either direction.

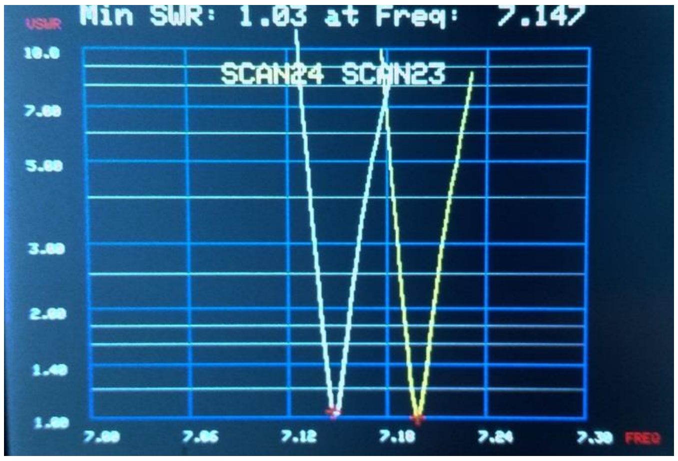
The same thing can be done after using the “Single Scan” or “Repeat Scan” options.

Overlay

This option can be used to overlay the graphical results of a previous scan over the current one. The SWR curve for the current scan is displayed in yellow, and the overlay scan curve is in white.

In order to use this capability, there must be a valid scan already in the analyzer’s memory. That can be done by doing a “New Scan” or by using the “View Plot” or “View Table” functions first.

Here’s an example:



The minimum SWR shown in the heading is that associated with the overlay.

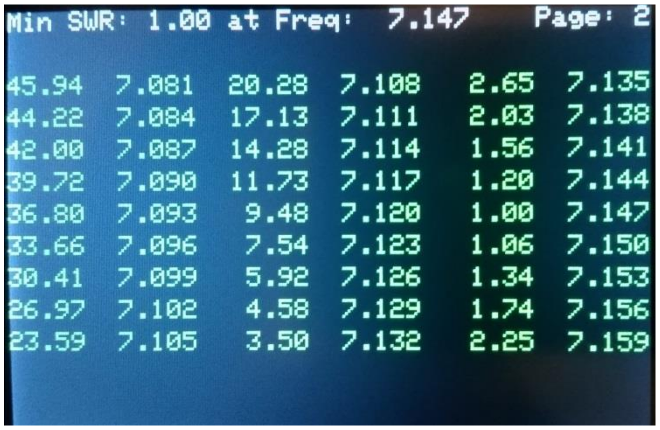
If there is no active scan already in memory, an error message will be displayed. Similarly, if the current scan and the overlay scan have different frequency ranges, an error message will be displayed. In either case, the “Overlay” will not be done.

Both scans are labeled. It’s hard to see in the picture, but the label colors match the colors used for the curves; yellow for the original plot and white for the overlay. If the main plot came from a file, its label will be the file name. If it’s a live scan, the label will show “LIVE”.

View Table

This selection allows the saved data to be displayed in a table format showing the SWR/frequency pairs for each of the 101 scan points. The output is a multi-page affair. Rotating the encoder will display the next or previous page.

The minimum SWR for the saved scan and the frequency at which it occurred are shown in the header.



```
Min SWR: 1.00 at Freq:  7.147     Page: 2

45.94  7.081  20.28  7.108   2.65  7.135
44.22  7.084  17.13  7.111   2.03  7.138
42.00  7.087  14.28  7.114   1.56  7.141
39.72  7.090  11.73  7.117   1.20  7.144
36.80  7.093   9.48  7.120   1.00  7.147
33.66  7.096   7.54  7.123   1.06  7.150
30.41  7.099   5.92  7.126   1.34  7.153
26.97  7.102   4.58  7.129   1.74  7.156
23.59  7.105   3.50  7.132   2.25  7.159
```

Plot>Serial

This allows the contents of a saved scan file to be sent to the Arduino IDE’s serial monitor provided the analyzer is connected to a PC and the IDE is running.

The output is in CSV format. The first line of the output contains the low and high frequency limits (divided by 1,000) for the scan and the remaining lines are the SWR/frequency pair numbers. The output is an exact replica of the data in the file.

It should be noted that there is no way to detect whether or not the USB connector on the Arduino Mega is actually connected to something or not, so if there is no connection to a PC, the function will appear to complete successfully. In other words, there is no way to display an error message indicating that there is no connection established.

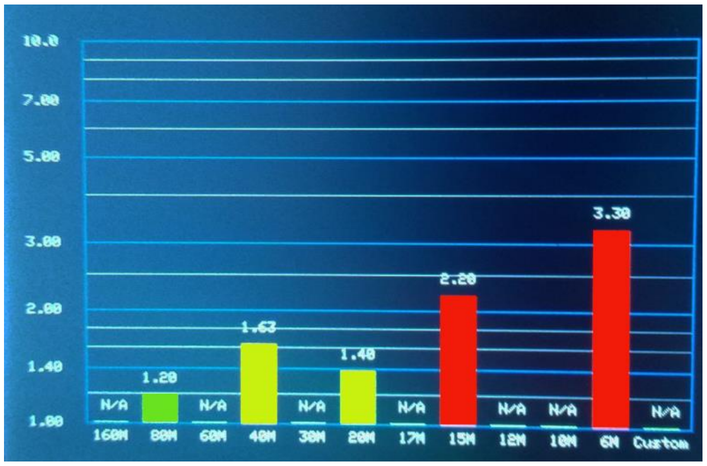
And, again, make sure the Arduino’s serial monitor bit rate is set to 9600.

The data displayed in the serial monitor can be copied and pasted into a text file to be saved as a “.csv” file, which can then be opened with Microsoft Excel or other spreadsheet programs for further analysis.

View Mins

Every time a scan is performed, the program saves the minimum SWR for the scan in the EEPROM. The "View Mins" function allows you to display a bar graph showing the most recent minimum SWR readings for all bands that have been scanned. Note the ability to display a bar graph for a single band was eliminated in this software.

Here's what the bar graph display looks:



The bars are painted in different colors for different SWR readings; a green bar indicates that the SWR is less than 1.5:1. A yellow bar indicates that the SWR is between 1.5:1 and 2:1. A red bar indicates an SWR of 2:1 or more. Those colors correspond to the colors used on the "analog" SWR meter in the "Frequency" function.

The bars are labeled with the band below the bar and the actual SWR at the top of the bar.

The picture shows the display with the 6 meter and "Custom" bands enabled. If either or both of them are not enabled, there will be blank spots where their bars should be. It would have taken a lot of extra code to recalculate the bar widths based on the actual number needed (although maybe I'll change that in the future).

Maintenance Menu

The following choices can be found under the “Maintenance” menu:

Delete File

This allows you to delete a single file from the SD card. It displays a list of the files (only 20 of them if you have more than that on the card; it’s an internal limit in the software right now).

Moving the encoder knob will allow you to select the file to be deleted and pushing the button will bring up a screen asking to either actually delete the file or cancel the operation.

Delete All

This choice allows you to delete all the files on the card. It will again display a list of the first 20 files on the card for 3 seconds than take you to a screen to allow you to either perform the operation or cancel it.

If there are more than 20 files on the card, it will inform the operator that only the first 20 files are listed, and if the operator chooses to delete the files, it will only delete the 20 files listed.

It will briefly display the name of each file being deleted.

Reset Seq#

This allows you to reset the file name sequence number to zero as long as there are no scan files remaining on the SD card.

If there are still scan files on the card, you will be informed of that fact, and the operation will be cancelled.

Clear Mins

This allows you to zero out all the saved SWR minimum readings.

Erase EEPROM

As noted in the setup instructions, using this version of the software requires running a program to zero out the EEPROM memory prior to installing this version. As that also might be required in future releases, I figure it would be a good idea to provide the option under the “Maintenance” menu.

The program detects that the EEPROM is not initialized and takes care of setting things up correctly itself during startup.

Note that the function does *NOT* reset the next file sequence number.

EEPROM>Serial

This function reads the data stored in the EEPROM and sends it to the Arduino IDE’s serial monitor in a fairly readable format. For those locations that have symbolic addresses defined in the software, the contents are explained.

Mount SD

Formerly if the analyzer was started with no SD card installed, simply putting one in would not allow it to have been used, as the only check for it was in the startup sequence. This function allows the card to be installed at any time of the operator’s choosing.

If no card is actually installed, a message indicating that fact will be displayed. If the card is mounted successfully, a message to that effect will be displayed.

There is still a slight problem with this function that I have not figured out how to solve yet. It is described below in the “Known Bugs & Glitches” section of this document.

Freq Cal

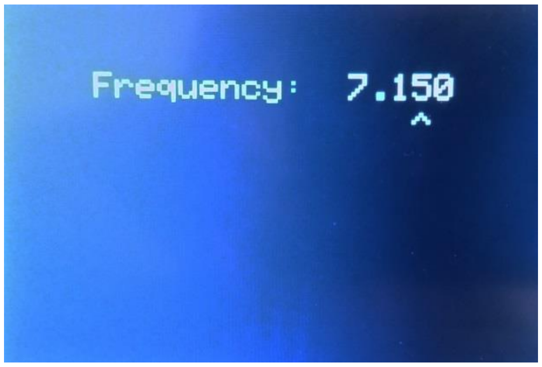
You should perform this function before trying to do any meaningful analysis with the device.

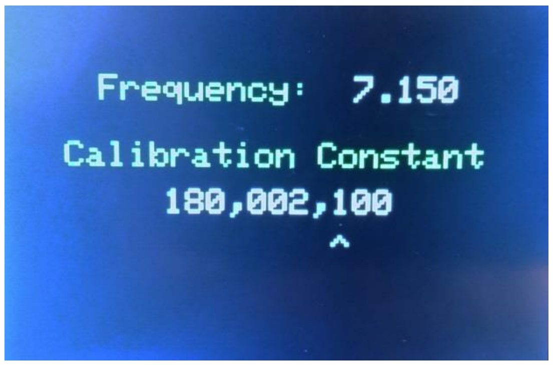
Added in Version 03.3 is the ability to calibrate the DDS frequency. If the frequency is off by a few KHz or so, it’s not really a big deal for tuning antennas (except maybe really narrow band ones such as my magnetic loop), however Edwin (PE1PWF) came up with the idea of using the analyzer as a simple frequency generator that can be used for such tasks as tuning IF chains. For that, the frequency needs to be correct.

To perform this procedure, you’ll need an accurate frequency counter capable of operating on the band on which you select to do the calibration or an accurate receiver for that band.

First, use the “Frequency” option under the “Analysis” menu to transmit a known frequency and using your receiver or counter, determine how far the analyzer is off frequency and in which direction.

Next, start the “Freq Cal” function under the “Maintenance” menu. When you start it up, it will ask you for the frequency to use; it works just like the “Frequency” option. Once you have set the frequency to use, the screen will display the default calibration constant for the DDS in use.


```
Frequency:  7.150
                ^
```


```
Frequency:  7.150
Calibration Constant
   180,002,100
           ^
```

On the second screen, the carat (^) indicating which digit will be changed when the encoder is moved will be set under the 100Hz digit. If the “Fine Tune” option is installed, the carat can be moved with short pushes of the “Fine Tune” button.

Finally, use the encoder to adjust the calibration constant until the analyzer is transmitting on the desired frequency. It takes a fairly significant change in the calibration factor to move the frequency by a significant amount. Also note that in order to raise the frequency, the calibration constant needs to be lowered and vice-versa. A change of 2KHz to 2.5KHz in the calibration factor results in approximately a 100HZ change in the transmitted frequency.

Once you have the calibration factor set, pushing the encoder button will save it for future reference in the EEPROM. When the analyzer is restarted the next time, the saved calibration factor will be used to initialize the DDS. You can also change the default value for your specific unit by editing the value of "CAL_9850" or "CAL_9851" as described earlier.

Note that if you perform the calibration a 2nd time, the calibration factor displayed will again be the default value for your DDS, not the already saved value. However, if you don't change it, the saved value will not be updated when you exit the function. It was done this way to handle the special case of someone having to change the default AD9851 calibration factor from 180MHz to 120MHz.

Options

Added in Version 03.4, thanks to Bill (AB1XB), we added the capability to turn the debugging output on or off via an option under the "Maintenance" menu. This option allows "Serial.print" statements in the code to be enabled or disabled without having to re-compile and reload the software. In Version 03.6, Bill has expanded this feature so that you can select statistics logging levels and the serial monitor baud rate without recompiling.

Selecting Maintenance > Options will give you a screen telling you the current state of several option settings and giving you the ability to toggle the state or just cancel out of the request.

Turn the encoder to select an option to be set, push the encoder button to move the highlight to the setting's value, and then turn the encoder to toggle between settings. Finally push the encoder's button to store the selected setting. Settings are saved in EEPROM and thus are persistent across operating sessions.

If Maintenance > Options > Enable Debug Mode is set to "yes", a red "DB" will be displayed in the upper righthand corner on most of the display screens (but not all of them).

During a scan, and depending on the logging level(s) selected, the statistics code will:

1. Calculate and, after the scan is finished, log the averages of the minimums, maximums, and standard deviations of the distributions of forward and reflected readings for all groups of point samples for all samples in the scan.

   To activate this feature:

   > Set Maintenance > Options > Log Statistics Summary to “yes”.

2. Log the minimum, maximum, and standard deviation of the distribution of forward and reflected readings in the sample of readings for each scan point in a scan.

   To activate this feature:

   > Set Maintenance > Options > Log Statistics Summary to “yes”, and
   >
   > > Set Maintenance > Options > Log SWR Point Stats to “yes”.

3. Log the 75 (by default) sample readings for each individual scan point in a scan.

   To activate this feature:

   > Set Maintenance > Options > Log Statistics Summary to “yes”, and
   >
   > Set Maintenance > Options > Log Indiv Readings to “yes”.

Also on the Maintenance > Options screen is a means to set the baud rate of the data sent to the IDE’s serial monitor.

To activate this feature, select Maintenance > Options > Serial Baud Rate, and select a desired baud rate for debugging output. You must also set the matching baud rate in the serial monitor’s window via a dropdown in the lower right of the monitor’s window, and restart the IDE.

Some General Notes about Menu Choices

In most cases, after an operation such as a scan or display of data in a saved file is performed, you will need to push the encoder button to take you back to the main menu. In some cases, such as deleting files, you are simply returned to the main menu automatically upon completion of the process.

One thing that was missing from the original code was a lot of the error handling that should have been included. I've added a lot of it, but there are still places where you may just be sent back to the main menu with no explanation. Most of these errors have to do with the SD card not being installed or not having any files on it.

On a related note, when you start the analyzer, if there is no SD card installed, you will be informed that any operations having to do with the SD card will be disabled. That's not exactly true right now. You can still try to perform file related operations, but you will usually get another error message explaining the problem. In the future, I plan to completely remove the menu options that require the file system if no card is installed.

It is also important to note that if the analyzer is started without the card installed, simply inserting one won't work. The analyzer must be restarted, or the "Mount Card" option in the "Maintenance" menu can be used.

Known Bugs and Glitches & Troubleshooting

Mount SD Card

There is one known issue that can be considered a real "bug". If you don't have an SD card installed when you turn the analyzer on, you can use the "Mount SD" command in the "Maintenance" menu to enable it, and that works. But, if you remove the card and reinsert it while the analyzer is running, it won't work. The "Mount SD" command will tell you that it was successfully mounted, however, it is not. I'm still trying to figure this one out.

Exiting the Single Frequency (or SWR Calibrate) Function

Another slight inconvenience is that when using the single frequency measurement function (or the "SWR Calibrate" function if activated), the analyzer sometimes won't respond to pushing the encoder button to exit the function. This is due to the fact that there is a 300mS delay between when it takes readings, during which the software won't see the encoder button operated. If you're too quick on the button, the software doesn't see it. Just holding the switch in a little longer takes care of it.

Slow Scan

It's an optical illusion! When using this software, you will notice that the curve for a "New Scan" plots very slowly as compared to the original code. In the original software, the software painted the graph axes, then ran a loop to get the readings, then ran a loop to find the minimum SWR, and finally another loop to plot the results. In this software, all three processes were put in a single loop. In the old software, there was a 5 to 6 second delay between when the axes were plotted and when the curve was plotted while the data was being gathered.

Now, the data gathering time is done in small increments between plotting the data points. It actually takes the same amount of total time. It just looks slow!

Maximum File Number

The maximum file sequence number currently allowed is 99. If more than 100 files are saved on the SD card, the sequence number will be reset to zero. This will cause the program to overwrite earlier saved files.

My Encoder Is Backwards

Many of the encoders out there are wired backwards from the documentation.

I moved the definitions of the encoder pins into the header file in Version 03.2 to make them easier to find. Simply flipping the numerical pin assignments on the definitions of “PINA” and “PINB” will fix the problem.

The Analyzer Reports Bad SWR Values

A number of folks have complained that the results they get when using the analyzer on a real antenna don’t match the readings that their other analyzers or SWR meters give them, or they report that it works just fine when using various values of resistors on the output, but doesn’t work on a real antenna.

Some of these problems have either been traced to bad DDS modules or other hardware problems in the units.

The rest are generally caused by the fact that the owner failed to use the frequency calibration function to set the unit up properly. Resistors (hopefully) are not frequency sensitive; antennas (hopefully) are.

If the analyzer’s output frequency is off by a few KHz, it’s not going to matter much when working on an antenna unless it’s perhaps something with a very narrow bandwidth like a magnetic loop. In many cases, we’ve seen some units that for whatever reason are way off frequency.

If you’re trying to set up a 40 meter dipole and the analyzer is putting out a 6MHz signal, you’re not going to get good results.

Initial state of Debug and Statistics Options

When upgrading to version 3.7, you could see unusual behavior due to the initial state of the debug options in EEPROM. Just go to the Maintenance > Options menu and turn off debugging and statistics, then start over and set whatever options you want.

Suggestion Box

We welcome any suggestions for further improvements. Please feel free to post on the “SoftwareControlledHamRadio” group, or email me at WA2FZW@ARRL.net.