# Handheld 2-meter SmartPhone Transceiver

*Figure 1 - SmartPhone HT*

Not long ago, a fellow Club member, **Lenny Rust W2LJR**, told me about a SmartPhone hack that he thought might hold some interest for me. The idea was to turn an Android-based SmartPhone into a two-meter handheld transceiver (Figure 1). I thought that the idea might have some mileage in it, so I went ahead and built the contraption.

I started out by going to the website offered by **Vance Vagell KV4P**, an Amateur Extra-rated ham from Durham, NC. Vance's quite helpful website can be found online at https://www.kv4p.com/index.html. The very simplicity of his idea and its implementation make this an attractive project for any ham who wants to build something just a little bit different or quirky.

The entire build requires a total of only eight or nine parts:

- a custom printed circuit board (PCB);
- an SA818S-V (or DRA818V) radio module;
- an ESP32 development board;
- an SMA connector, 90° PCB mount;
- an appropriate antenna with male SMA connector;
- a USB-C x USB-C data cable 6" long, or a USB-C wye-connector;
- an adhesive gel pad; and
- a 3-D printed case and optional stand.

Before I go into the actual build, I need to explain that there are some options here. At the time that I began this project, the builder had to gather all of the parts and pieces from various sources such as those identified on the KV4P website. Now, however, there are at least three suppliers (*New DIY*, *Chelegance*, and *Halibut Electronics*) who have kitted the parts, making the "gathering" much simpler.

If gathering the parts yourself, you must make some choices as you progress. For example, you must decide which radio module to fit, the SA818S-V or the DRA818V. Then you need to decide which ESP32 development board to purchase, which is a function of the design level that you choose to build. Design level v1.8a uses a 38-pin, 1-inch width pin row spacing ESP32 board. Design level v1.8b uses a 38-pin, 0.9-inch pin row width ESP32 board. Finally, design level v1.8c uses a 30-pin ESP32 board. In each case, it is important to also source the matching PCB, which also then requires a choice to be made between the PCB sources. In all of the v1.8 designs, the PCB can be purchased through either *PCBWay* or *JLCPCB.* The main parts of this build are shown in Figure 2.

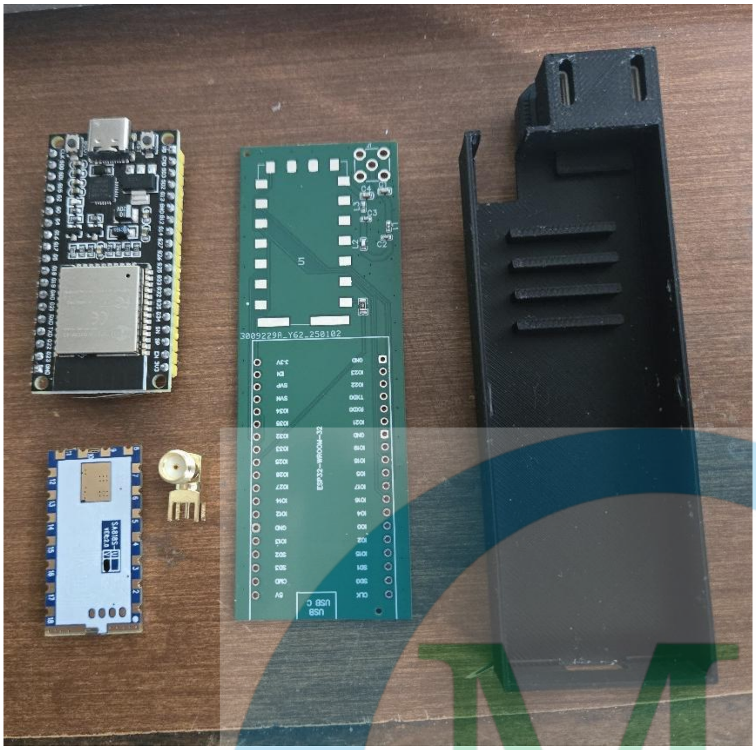


*Figure 2 - Major assembly parts*

As if all of that wasn't confusing enough, there is now also a v2.0 design level, which takes a slightly different approach. This is the version offered by Halibut Electronics in the kit that they sell, but it is also available for the builder in the form of Github files used to produce the PCB with its integrated ESP32 functionality.

A choice must also be made as to the antenna to be used with this project. Many builders will choose the small pocket-size flexible "stubby" antenna, which in fact I did. However, for a boost in performance, I later decided to purchase a Nagoya NA-771 antenna.

In my build, I downloaded the necessary files for board production and population by JLCPCB, and then sent them off to that company to get the boards into production. There is a five-board minimum at JLCPCB, so I now have four surplus boards, which I will make available at cost to Club members on a first-come basis. So, let's talk about the board cost for a minute.

*Figure 3 - Printed stand from 3-D parts set*

The bare boards would be produced at a price of forty cents per board, as the standard JLCPCB price for boards of this size is $2.00 (two dollars) for a five-board minimum order. The shipping would add an additional eighteen dollars or so for DHL express carriage from Asia to my door. However, in this case, I decided to try out the JLCPCB board assembly option. With this option, JLCPCB will place and solder any components that you decide you want them to install. In order for this to happen, JLCPCB must be provided with some component identification and placement information, which is contained in one of the files that I downloaded from the KV4P website and then uploaded to JLCPCB as a part of the board order process.

In this situation, the delivered price of the assembled PCB's from JLCPCB was $48.72. As a result, the actual per-board cost was $48.72 ÷ 5, or $9.744 each. OK – let's start running a tab on this project.

I will begin by listing each of the various parts and components that make up this project, in an effort to truly determine the actual bottom-line cost for this deal. The KV4P website touts a $35.00 price tag, so let's see just how close I came to that figure.

The tabulated total cost, as shown in the table at right, comes out to $73.88, but I think that the $22.37 cost of the Nagoya antenna can justifiably be deducted, as it was an after-the-fact modification and so was not a part of the minimal build cost.

**Project Cost Summary**

| Item | Cost |
|---|---|
| printed circuit board | $9.74 |
| ESP32 development board | 6.18 |
| SA818S-V radio module | 8.69 |
| USB-C x Micro USB cable | 6.81 |
| USB port adapter | 1.57 |
| antenna (original) | 5.33 |
| antenna (Nagoya NA-771) | 22.37 |
| SMA 90° connector | 1.16 |
| 3-D printed case & stand | 5.50 |
| gel adhesive pad | 1.06 |
| USB tee | 5.47 |
| **TOTAL** | **$73.88** |

No matter how one looks at it, the build cost here was almost double that suggested by the developer and published on his website. That fact notwithstanding, the project nonetheless easily produced a quite functional two-meter hand-held radio based on the Android SmartPhone.

It must be understood that as already mentioned, the SmartPhone must be an Android-based device. Furthermore, the SmartPhone must also be of a type that uses a USB Type C port for external communications. In addition, the Android app requires that the OS be at least version 8, which would most likely include any Android phone newer than about a 2017 model.

The build consisted of soldering three items to the prepared PCB. I say *"prepared PCB"* because in this case, the PCB is *not* a bare board; instead, it has several passive components already installed, in this case by JLCPCB. The installed passives include three inductors (L1, L2, L3), four capacitors (C1, C2, C3, C4), and a single non-identified resistor at one megohm.

Those items to be installed at build time include the SA818S-V radio module integrated circuit (IC), the ESP323 development board, and the 90° PCB-mount SMA female connector.

While *not* specified or even suggested in the build guidelines, I found it to be quite advantageous to drill three 1/8" holes in the 3-D printed case into which the completed PCB will be snapped. The reason for the existence of these three small holes is quite evident if one is at all familiar with the ESP32 development board. Two of the holes are near the lower end of the case, in alignment with the two momentary switches (*RESET* and *BOOT*) on the ESP32 board. The third hole is in alignment with the multi-function indicator LED on the board. The three holes simply make it possible to reset and reboot the ESP32 board as well as making it possible to monitor the board status by viewing the indicator LED. Without these holes, it becomes necessary to remove the board from the case to access the switches or to view the LED.

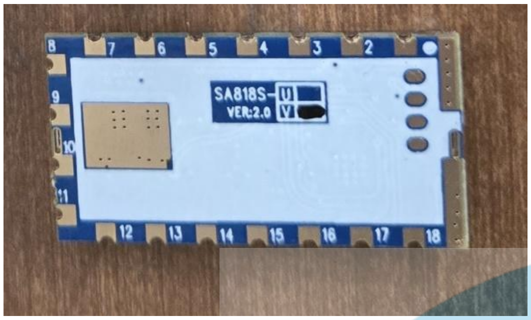



*Figure 4 - SA818S-V Radio module*

The build first step is to install the SA818S-V radio module. This module is a twenty-pad surface-mount device without pins or leads. In this situation, start by tinning all twenty of the pads on the PCB. Then, place the module onto the PCB in proper alignment with the PCB pads. Hold it there while applying heat to one of the corner pads. This will cause the tinning solder on that pad to flow and allow the module to become bonded to the PCB at that point. Verify the module's alignment, and then repeat the heat-and-set process at the diagonal corner from the first pad heated. The design of the module provides a notch at each of the eighteen pads around three edges of the module (Figure 4). The two long pads on one short end of the module do not have the notches, but they are easily heated by applying the soldering iron to the PCB pad. Once all twenty of the pads have been reheated and the solder flowed into place, the module should be secure to the PCB. Touch up the soldering by adding additional solder to those pads where it may be required.

*Figure 5 - 90° SMA connector*

Now move on to the 90° SMA connector (Figure 5). Insert the connector pins into the corresponding holes in the PCB so that the connector extends straight out from the end of the PCB. It is important that the connector extend from the end and *not from the side* of the PCB in order for the PCB to properly fit into the plastic case later. Solder the pins carefully, ensuring that no solder bridges exist to the center pin, which is the signal pin.

Finally, install the ESP32 development board (Figure 6) to the PCB in such a manner that the USB port on the development board is nearest the end of the PCB. Be careful not to bend any of the 38 pins on the ESP32 board while inserting them into the PCB. It is a snug fit, but it *does* fit perfectly. Solder each of the ESP32 pins to the PCB, again taking care to avoid any solder bridges between adjacent pins. Check your work carefully, and when you are satisfied, it is time to move on. Clip the ESP32 board pins short, as the solder side of the PCB will be against the SmartPhone case when the unit is installed to the phone. Now, install the antenna that you chose to the SMA port on the radio unit.

It is now time to download and install the *KV4P HT* software to your Android phone. The software is readily available at the Google Play Store, and it is safe to download and install.

Once the software installation is complete, connect the completed PCB to the SmartPhone using the USB cable or the USB wye connector, whichever you decide to use. Launch the KV4P HT software. On initial startup, the software should detect the attached radio and will then download and install the firmware to the radio unit. That process can take as long as twenty minutes, so be patient. The project documentation says that the firmware install should take five to ten minutes, but mine took just short of eighteen minutes by actual count.

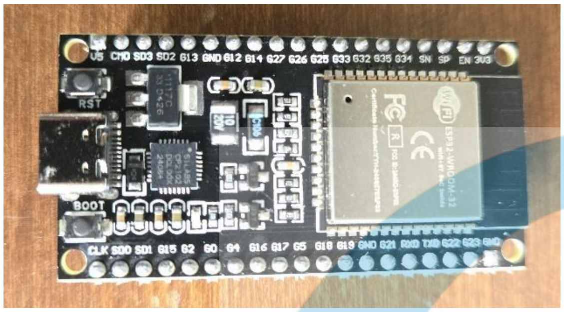


*Figure 6 - ESP32 development board*

When the firmware installation is complete, it is time to test the radio's operation. It is a simple and intuitive matter to set up a local repeater or two, as well as to set up a couple of simplex channels in the unit's memory slots. Repeater configuration includes the ability to set the direction of offset as well as the CTCSS tone frequency, and you can name the stored channel in almost any manner that suits your needs.

In the radio settings area, the user can choose from a "sticky" PTT setup or a "toggle" PTT setting. I prefer the toggle method, so that is what I set up. By trial and error, I also determined that medium or high mic gain was required, and that the OTA audio was best with pre-and de-emphasis active, the low pass filter active, and the high pass filter disabled. There are some other settings that the user can test out as time permits, but it is important to set the Bandwidth to "WIDE" and to set the squelch to a comfortable level.

My OTA performance was considerably better with the Nagoya NA-771 antenna than it was with the stubby flexible antenna recommended in the build instructions, so it was well worth the additional money spent.

I used the radio for an entire day with no problems either with the radio or with the phone's basic operation. I mounted the radio unit to the SmartPhone using the adhesive gel pads recommended on the KV4P website. It took only a 3/4"-wide strip cut from one of the pads, which are sold ten to a box. These pads are easily removed by twisting the radio, and they are reusable by simply rinsing them under running water, so the box of ten pads will last a *long* time.

A few words about the 3-D printed case are called for here. The KV4P project site offers a link (https://www.thingiverse.com/search?q=kv4p&page=1) to a Thingiverse search for cases for this radio. Select a case that suits your fancy, but make sure that it is specified for the board version (either 1.*x* or 2.*x*) on which you built your radio. I chose a design set produced by the project creator, which includes a stand for the assembled phone and radio. This set includes two different cases – one for the cable connection and one for the USB wye (Figure 7) connection. Both of these cases have inlaid lettering set into the outer surface and a set of letter cutouts to be placed in there. The idea is to print the lettering, which reads *"kv4p HT"* in a contrasting color. Depending upon the printer capability, this can possibly be printed all as one job with a color change after the lettering is printed. I cheated, however, and modified the designs to remove the lettering altogether, giving me a pristine surface without any decorative printing.

*Figure 7 - USB wye connector*

I had both cases printed, as I was going to try out both methods of connection, so that I could write about both methods from a position of experience. The printing was done for me by another Club member, **Rich Subers W2RHS**, who did an excellent job, as always, with the printing. I asked for black or dark grey and received the black, with which I am quite happy. So… what about the two connection methods?

It turns out that most builders of this radio will probably end up with the cable connection, as I ultimately did. This is due to the fact that when installed in a protective case, as most of do with our phones, the USB wye does not quite fit. It fits perfectly with the phone out of its protective casing, but when protected, the spacing between the phone's USB-C port and the radio's USB-C port is just a little bit too wide.

As to the cable connection… for some reason, the project designer specifies a USB-C x Micro-USB cable plus a Micro-USB to USB-C adapter. I don't really like using adapters if I can avoid it, so I set out to find a USB-C x USB-C data cable in a six-inch length. It turns out that it was sort of difficult to find the right cable. I went through three different cable options from Amazon before I found one that was truly data-capable, though *all* of them claimed to be *"fast-charge high-speed data"* type cables. Who would have thought that it would be so difficult to get a data cable in that size and type?

Because the phone with radio attached will not fit in my belt holster where I customarily carry my phone, I pulled the radio off and stored it in an anti-static zip-lock type pouch, together with the connecting cable. I can carry that and the antenna in my laptop case (which is too heavy by far already), making them as readily available as my computer is.

End result – a working two-meter hand-held radio based on an Android SmartPhone. It lives up to its advance billing, with more than a watt of output. I was able to reach the Club's two-meter repeater from the Clubhouse with a usable signal. Next up was the performance and spectral testing.

To do some spectral testing, I used my Siglent SVA1015X Spectrum/Vector Network Analyzer, with the signal brought in two different ways. First, I did over-the-air testing to an antenna mounted to the RF input port of the SA. After that, I coupled the radio to the SA via an attenuator and an SMA tee connector, with a dummy load on the second port of the tee.

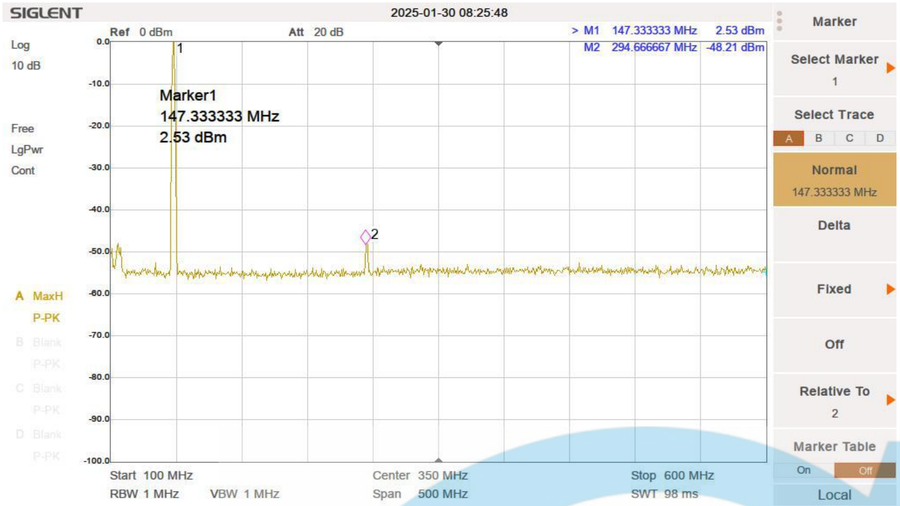



*Figure 8 - Over-the-air test transmission at 147.540MHz*

Needless to say, two very different pictures were painted with the two different test methods. Figure 1 is the over-the-air test. Both tests were made using a 147.540MHz transmission. The OTA test was seen by the SA, as is evident by the data shown in Figure 8, as a 147.333,333MHz fundamental and a single 294.666,667MHz harmonic. The harmonic came in at 48.21dBm below the fundamental, which was at 2.53dBm above the reference level. Attenuation of 20dB was applied by the SA, the transmitting antenna was the Nagoya NA-771, and the receiving antenna was a seven-section telescopic mast. The test frequency span was 500MHz, from a starting point of 100MHz to a stopping point of 600MHz.

By way of comparison, Figure 9 shows the results of a cable-connected transmission test. As was mentioned earlier, this test too was at 147.540MHz. The setup for this test had the radio coupled to the SA via an SMA tee to an inline 20dB attenuator, with a 50Ω dummy load on the second branch of the SMA tee. Note that the external 20dB attenuator was in addition to the 20dB internal attenuation afforded by the SA. The same 500MHz test span was set, again from 100MHz to 600MHz. This test yielded the same 147.333,333MHz fundamental, indicating good consistency in the transmitter. The fundamental was 2.14dBm above the reference level. In this test, however, three harmonics were seen within that 500MHz span, one of them at 295.333,333MHz, one at 442.666,667MHz and finally the third at 590.666,667MHz. As to the harmonic attenuation, the first harmonic was 37.86dBm below the fundamental, with the second harmonic a bit lower at 48.23dBm under the fundamental, and the third at 51.01dBm under the fundamental level.

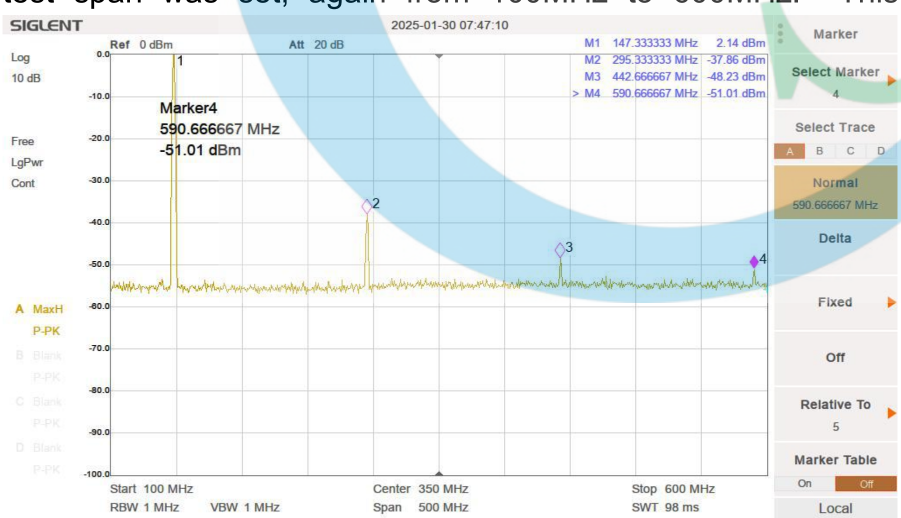



*Figure 9 - Cable-connected test transmission at 147.540MHz*

It is rather interesting that while the fundamental came in at a little bit lower than the nominal transmit frequency, the harmonics were actually closer to true harmonics of the nominal frequency than they were to simple multiples of the measured fundamental.

Next up was the power output testing. This was done with my Bird 43 directional wattmeter, using a *"5C"* element, which is rated at five watts and for 100 to 250MHz. According to the KV4P website, this radio should output a one-watt RF signal. However, in actual testing, the output was just north of 1.5W (see Figure 10). The test conditions were the same 147.540MHz signal output through the Bird and into a dummy load


*Figure 10 - Output power testing at 147.540MHz into Bird 43 directional wattmeter*

The final testing that I did was a quick look at the output frequencies as seen by a layman's frequency counting tool, the Latnex FC-2800M hand-held OTA frequency counter. This tool is somewhat useful as a quick-check of the rough output frequency. What makes this tool especially useful is its ability to decode the embedded coded audio tones that are being sent by the transmitter. I did two separate tests here, one at the by-now common frequency of 157.540MHz, and the other at the W2MMD 2-meter frequency pair of 141.180MHz and 147.780MHz with the CTCSS set properly to 131.8Hz. Figures 11 and 12 show the results of these tests.

As can be seen from these photos, this low-tech frequency counter found the output signals to be very close to their nominal values. In the case of the repeater transmission, it was clearly offset by the +600kHz difference, and the CTCSS tone was clearly set to its proper 131.8Hz value.


*Figure 11 - 141.180MHz repeater output*

As is by now quite evident, this little radio is a decent performer, though it is *not quite* rock-solid at its nominal or indicated frequency. This was clearly evident from the various tests performed and illustrated herein, but it is also somewhat evident when actually using the radio on the air. I have received two comments about very slightly distorted audio when transmitted from this unit. I attribute that distortion to being slightly off-frequency.

The build was extremely simple, with the most difficult part – which was not difficult at all – being the soldering of the two long pads on the SA818S-V radio module. That is only because the specific placement of the module, where the side pads are centered on their PCB footprint pads, places the module in such a position that only a tiny sliver of the wide PCB pads for the long module pads remain visible and therefore accessible to the soldering iron. In my case, I had an extremely fine chisel-point tip in my iron, so I had no real trouble in reaching those two pads to solder them properly. A builder with a soldering iron tip that is more blunt would definitely have difficulty in soldering those pads correctly.

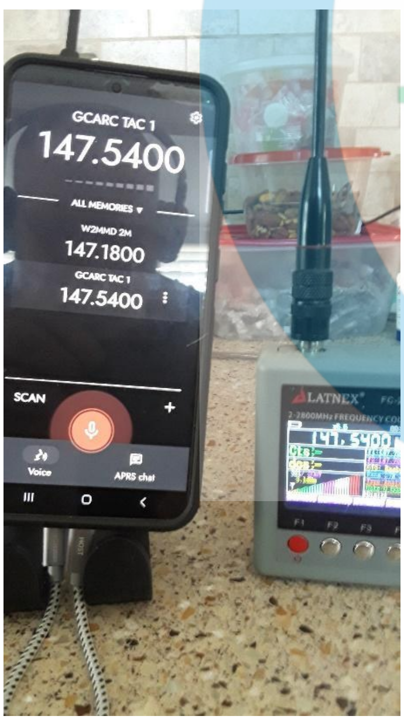



*Figure 12 - 147.540MHz simplex output*

The testing of the radio went well and was quite straight-forward in nature. As illustrated and discussed, the accuracy of the radio is acceptable with a caution, which is that there is a slight frequency miss. However, I only received two complaints out of the extensive use and testing that I have given this radio, so I figure that it is acceptable. The actual difference is 0.14007% -- less than a quarter of a percent of the nominal frequency.

Although 47 CFR §90 specifies frequency stability limits for general land mobile radio usage, it does not address the amateur radio two-meter band, as that part does not regulate the amateur frequency ranges. The amateur radio two-meter band is, of course, regulated under 47 CFR §97, which is curiously silent on frequency stability across all of the amateur bands. Nonetheless, if we consider the 150-174MHz band for land mobile radio as being somewhat similar, we would probably be reasonably safe in using the 50ppm figure assigned for that frequency range in §90.213. Thus, we should calculate what 50ppm would be for a 147.540MHz signal, which comes in at 7,377Hz. This means that a 147.540MHz signal at a tolerance of 50ppm should fall between 147.532,623MHz and

147.547,377MHz.  We have already seen that this radio displayed a frequency error of 206,667Hz, which is over 28 times the 50ppm limit imposed on similar frequencies in 47 CFR §90.213.

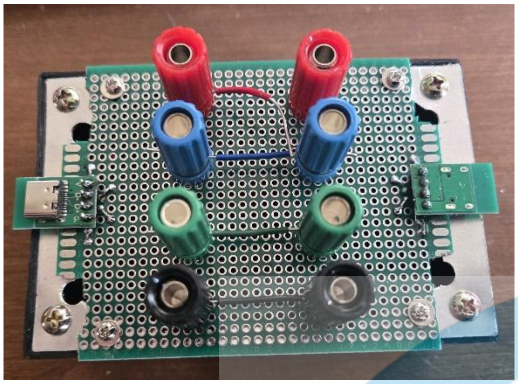


*Figure 13 - USB-C breakout box*

I find it quite interesting that §97 does not address frequency stability at all.  Normally, when something is not addressed in a certain piece of Federal regulation, it falls back to some over-riding regulation in some other section of the regs.  I have been completely unable to find any such over-riding regulation that would apply to the amateur bands as related to frequency stability.  Does that mean that it is wide open and not regulated?  That does not make much sense to me… but I cannot find anything to cover this issue.

I decided to take a look at some handheld radios for which I have manuals in my library, to see what kind of frequency stability they claim.  Here is what I found:

- Anytone D878UVII+… ±2.5ppm
- Azden AZ-11… not specified
- Baofeng BF-F8HP… ±2.5ppm
- Connect Systems CS600/CS700 family… ±1ppm
- Icom IC-U86/IC-V86… ±2.5ppm
- IcomID-52A… ±2.5ppm
- Kenwood TH-F6A… not specified
- Kenwood TH-D71… ±5ppm
- Yaesu FT-60… ±5ppm


*Figure 14 - Standby current draw*

As is evident from the above data, all of which came directly from the manufacturers' published manuals, the above-listed radios show a range of stability claims from a low of 1ppm to a high of 5ppm.  It seems, however, that 2.5ppm is the prevalent figure in this type of radio.  With this in mind, and then taking that value as a standard, our KV4P SmartPhone HT looks even worse, as the comparison done earlier was to a standard of 50ppm, twenty times greater than the 2.5ppm limit so common in handheld radios.

*Figure 15 - Receive current draw*

This brings me to the specifications of the SA818S-V radio module. There are no specifications listed as to output frequency stability. The only thing even remotely helpful is the fact that the board uses a 1ppm TCXO crystal from KDS Daishinku. I would think that with that kind of base, the radio *should* have better frequency stability than it actually does, though that performance really is quite dependent upon the peripheral components used in conjunction with the crystal. The point is that they started with a good – not the best, of course, but good nonetheless – crystal and they *could* have designed a radio module with tighter frequency control. Of course, that would also have driven the cost upward. I have seen these modules online ranging from a low of what I paid – eight dollars and change – up to as much as twenty-three dollars. Still, I would have paid a couple of bucks more for a more precise frequency synthesization capability.

*Figure 16 - Transmit current draw*

This is a good project for someone who wants to build a radio that is unique and completely portable without dependence on any battery *except the cell phone battery,* from which it *sucks* current.

A while back, I built a USB-C breakout box (Figure 13) that gives me access to each of the four main lines in the USB bus. The breakout box uses a pair of SMT USB-C ports with each one mounted to a small adapter PCB with four pins. The port PCB pins are then installed to the project board and connected to a set of binding post pairs, one for each of the four USB lines (+5V, D+, D-, GND). The whole thing is built up on a 70mm x 90mm perforated project board (PCB) with plated through holes. The holes are in 26 rows by 31 columns, for a total of (26 x 31 =) 806 holes. The number of holes is not important; the box could have been built with a blank piece of PCB material. I simply selected convenient locations to install the binding posts, and then wired them to the pin headers that would support the port PCB's. This allows me to monitor any or all of the four lines quite easily.

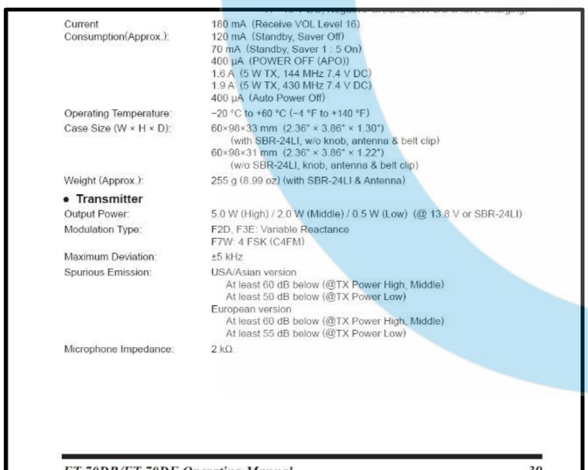
By connecting jumper wires between three of the post sets and an ammeter between the +5V post pair, I could easily read the current drawn by the radio, which came in at a surprising set of values:

- 139mA when idle (Figure 14);
- 350mA during RX operations (Figure 15); and
- 491mA during TX operations (Figure 16).

If we weight-correct the current draw based on a 70% idle time, 18% RX time, and 12% TX time, a typical solid hour's worth of current draw might be something like this:

- 139mA x 0.70 = 97.3mA idle draw;
- 350mA x 0.18 = 63mA RX draw;
- 491mA x 0.12 = 58.92mA TX draw; thus
- Total draw...... = 219.22mA.

My own cell phone is a Samsung S24+, which uses a 4900mAh battery. If the HT was the *only* thing drawing current, which of course, it is *not,* the battery would last for 22.352 hours. However, the *ViserMark* testing figure (https://www.visermark.com/post/samsung-galaxy-s24-plus-smartphone-battery-performances) shows an average useful battery charge life of 41 hours for this cell phone. If we take the 4900mAh battery and extend that over 41 hours, we derive a typical hourly draw of 119.5mA. Thus, by taking the 219.22mA and adding it to the 119.5mA basic draw, we get a new composite draw of 338.72mA, which results in a charge life of only 14.46 hours with the radio in operation.



| | |
|---|---|
| Current Consumption(Approx.): | 180 mA (Receive VOL Level 16)<br>120 mA (Standby, Saver Off)<br>70 mA (Standby, Saver 1 : 5 On)<br>400 µA (POWER OFF (APO))<br>1.6 A (5 W TX, 144 MHz 7.4 V DC)<br>1.9 A (5 W TX, 430 MHz 7.4 V DC)<br>400 µA (Auto Power Off) |
| Operating Temperature: | −20 °C to +60 °C (−4 °F to +140 °F) |
| Case Size (W × H × D): | 60×98×33 mm (2.36" × 3.86" × 1.30")<br>(with SBR-24LI, w/o knob, antenna & belt clip)<br>60×98×31 mm (2.36" × 3.86" × 1.22")<br>(w/o SBR-24LI, knob, antenna & belt clip) |
| Weight (Approx.): | 255 g (8.99 oz) (with SBR-24LI & Antenna) |

- **Transmitter**

| | |
|---|---|
| Output Power: | 5.0 W (High) / 2.0 W (Middle) / 0.5 W (Low) (@ 13.8 V or SBR-24LI) |
| Modulation Type: | F2D, F3E: Variable Reactance<br>F7W: 4 FSK (C4FM) |
| Maximum Deviation: | ±5 kHz |
| Spurious Emission: | USA/Asian version<br>At least 60 dB below (@TX Power High, Middle)<br>At least 50 dB below (@TX Power Low)<br>European version<br>At least 60 dB below (@TX Power High, Middle)<br>At least 55 dB below (@TX Power Low) |
| Microphone Impedance: | 2 kΩ |

*FT-70DR/FT-70DE Operating Manual* 39

*Figure 17 - Yaesu FT-70D current draw specs*

The measured current draws were considerably higher than I expected them to be, and indeed are quite higher than are the current draws of typical handheld radios with their more complex support and peripheral circuitry.

By nature of comparison, I selected the Yaesu FT-70D handheld, and looked at its current draw specifications from the Yaesu Technical Supplement for that radio (Figure 17). The radio is a nominal five-watt transmitter with a moderate set of more advanced features. While not, by far, the most complex handheld in Yaesu's lineup, it is nonetheless a fairly complex radio. Its features include both analog and digital operational capability, C4FM mode, WIRES-X capability, and it offers mode and status indicators. In other words, there is a *lot* more going on here than in the SmartPhone HT, so one might expect the current draw to be higher, right? Well, it seems as if that is an incorrect assumption. The actual specifications are as follows:

- 70mA idle (with 1:5 battery saver active);
- 120mA idle (without battery saver active);
- 180mA RX operations; and
- 1.6A TX operations (5W output on VHF band).

All of these values are less than their counterparts on the SmartPhone HT, *including the TX operations current draw.* I know – 1.6A is considerably more than 491mA, but, let's compare apples to apples for a minute and see where that takes us.

The FT-70D boasts a 5W RF output as compared to the 1W of the SmartPhone HT. Even if we are generous and give the SmartPhone HT credit for its actual 1.5W output, the Yaesu unit still comes in lower. Five watts divided by one and a half watts equals $3.33\overline{33}$. If we then multiply the 491mA TX draw of the SmartPhone HT by $3.33\overline{33}$ we come up with a value of $1.63\overline{66}$A as the projected 5W draw, which is actually very slightly higher than the TX draw specification of the FT-70D. In reality, the TX operational current draw is the only one that seems to be in line.

While I could probably keep going and find something else to talk about here, everything that really matters has been covered. I found this project to be an interesting build, and I would recommend it to anyone who wants something that is just that little bit quirky and off the beaten track. Sure, it could be better, but that can be said about almost every piece of equipment out there. In this case, we get just exactly what we expect to get when we assemble this unit. It will be fun to use for a little while, and then the novelty will wear off and we are left with a radio that works, but will almost certainly not be our first grab when we reach for a radio to take out into the field. Still… we *might* just think about grabbing it…