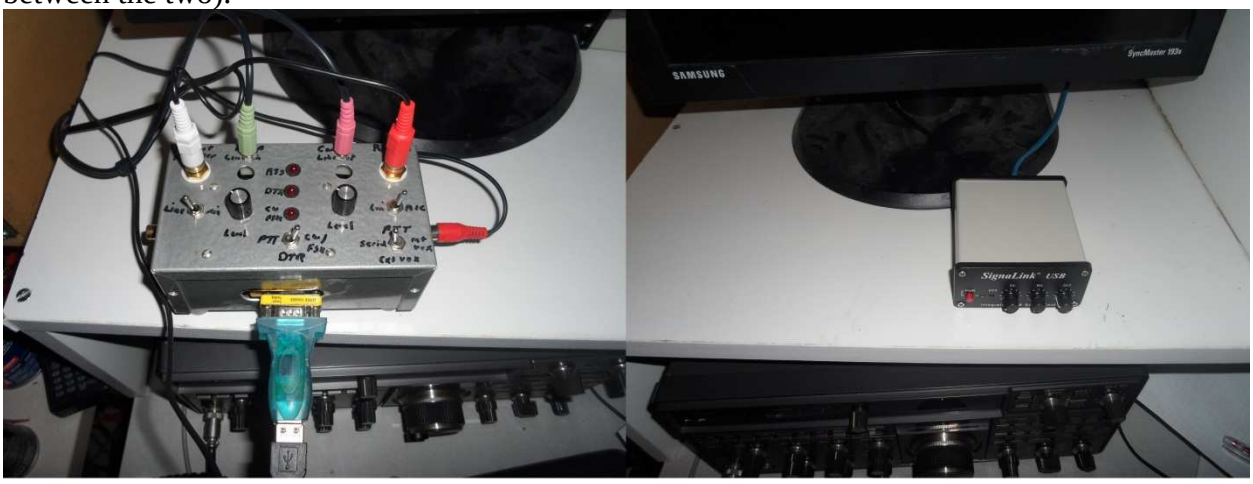
Rettysnitch Corner: When good enough just isn't … Part II

(Signalink USB® Mods)

Mat Breton - AB8VJ

Sometimes you buy something only to discover that it isn't as good as it might be, or as good as you had envisioned. It could be shoddy design, shoddy manufacturing, misleading marketing, the fact the company just plain missed the mark, or perhaps you had unrealistic expectations. You are faced with a tough choice: you can try to return it for another, you can try to get your money back, you can give up and throw it away, or you can try to make lemonade out of your lemon.

I recently purchased a used Signalink USB® digital interface. Digital interfaces are used by hams to interface their rig audio signals (mic and speaker lines) into/out of their PC. This allows us to use digital modes such as PSK31, RTTY, slow-scan TV, etc. I had been using a homemade digital interface I had made out of re-purposed modem transformers and optical isolators. It worked fine, except that it still had some extra noise in it from unintentional coupling, it looked like a rat's nest because of all the cables, and it wasn't easily transportable between rigs and computers due to all the settings that had to be tweaked. For these reasons I decided to investigate an external sound card solution. In theory this should fix the issues I had. The Signalink USB® is an outstanding concept: place everything into one box so that you have just one cable to the rig, and one cable to the computer. The on-board soundcard makes signal adjustment much easier. The built in digital-optimized VOX means you don't have to constantly readjust the VOX on your rig, or worry about a serial-port PTT function. And this device galvanically isolates your computer from your rig (reduces the chances of RF interference between the two).



(A) Homemade Interface (B) Signalink USB Interface

When I got it home, the first thing I did was hook it up to verify basic operation: great, it worked. I also verified that the computer and rig truly were galvanically isolated (many of its competition are not). But I'm not satisfied with "it seems to work" … I need to know how well it works. Using a second laptop as a sound generator, and using analyzer software on the primary computer, I ran it through some basic tests. Its frequency response from 50Hz to 3kHz was relatively flat (within 3dB) across a wide input dynamic range on both receive and send. Unfortunately I discovered that it had significant (50dB) low-frequency broadband noise (between 10Hz and 700Hz; the noise peaked at about 100Hz, and followed an approximate 1/f curve). 50dB is huge: more than a 300 times the electrical noise at low frequencies. There were also a couple of other quirks (less critical but interesting and/or annoying): when the device was turned off it continue to send sound to the computer, but at a 10 to 15dB reduced level. Also,

when the device was turned off it generated a 30dB noise spike at approximately 1 kHz (and even harmonics). In my opinion, while the device was functional it didn't meet my high expectations.

I took the device apart to see what was in it: the electrical design was quite simple. It consisted of an audio USB converter IC, two isolation transformers, four inverting AC voltage op-amp circuits, and a PIC microcontroller for the internal VOX function and control of the PTT relay (using a built in comparator). The build quality was generally very good: there was only one poor solder joint (a leaded LED), which I touched up. The PCB layout was acceptable; although I wouldn't have done the high-speed differential USB traces as it was executed (one differential leg had an extra/unnecessary throughole via, and was run on the other side of the PWB for a short length). After reverse engineering the circuit, it was easy to see that there were a couple of electrical design flaws contributing to the problems I had noted during testing, as well as a few others that were easily spotted:

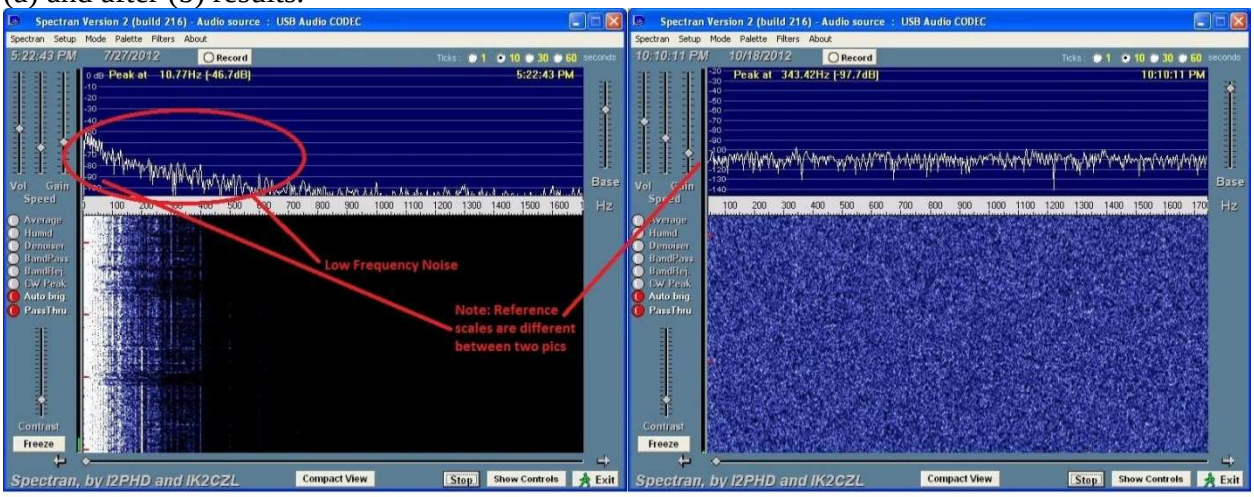
- The bias circuits for the op-amps were directly driven from the USB power, with only some minor capacitive decoupling: This resulted in an extremely low circuit PSRR (Power Supply Rejection Ratio) in spite of the low-noise high-PSRR op-amps used in the design, with the noise from the USB power feeding directly into the audio stream.
- When the on/off switch was in the "off" position, it turns out that the voltage powering the PIC and op-amps does not drop to zero volts. Instead it hovers around an average of 2.5Volts. This is caused by a "sneak" path between the HAAT (hot at all times) USB IC circuit, and the switched PIC microcontroller and op-amps when unpowered. In reality the voltage is actually oscillating as the voltage rises until the PIC microcontroller tries to power-up, collapsing the rail voltage and turning back off, and then repeating.
- There is minimal RF filtering on the USB power/ground. It is typical in USB applications to use ferrite beads/chips or chokes in addition to capacitive decoupling due to the high-frequency nature of this coupled noise. I did not detect any faults in my shack, but it could possibly occur if I moved the device between various computers, rigs, and locations. Note that you should not add additional capacitive decoupling to the Signalink USB® 5V USB power/ground lines, since the USB 1.1 spec limits the maximum capacitance to less than 10uF total to the downstream device.
- There is no ESD protection on the USB data pins and USB power line … they go right from the connector to the USB Audio and PIC ICs. Although not required, TVS (Transient Surge Suppression) is often added to create robust solutions using USB that directly drive CMOS ICs.
- The Aux and Speaker outputs are not buffered, meaning that they can couple noise back into the audio circuit, and that a change in impedance at these outputs will affect the audio stream. As long as nothing is plugged into these outputs, this does not matter of course. I was able to use an aux speaker plugged in as a microphone … any noise in the room was turned into an electrical signal and passed to the computer as noise. The Monitor output can likewise couple noise back into the microphone stream (do not directly couple it to earbuds or a speaker).
- The case was only grounded through contact points on the edges of the circuit board. These contact points were "floating" in the aluminum slides. Unfortunately aluminum will form an aluminum-oxide barrier when exposed to air, and aluminum-oxide is a pretty good insulator (~35KOhms or so when formed in humid air). Unless the oxide is broken and pressure is maintained to keep an air-tight metal-metal connection, this is not a

reliable method of grounding a case. In addition, the front and back covers were non-conductive plastic: this essentially negated any advantage of having an aluminum case in the first place. I measured very high impedance between the case and the USB shield when it was assembled.

- When the device is turned off, the USB Audio IC is kept fully alive. This has the advantage of not causing the windows driver to "lose" its settings, but on the other hand the IC will also continually send data to the PC. The IC will only suspend itself (and turn off the internal 3.3V supply) if the USB bus becomes inactive for a period of time (such as when the PC is turned off). Because the op-amps are "off", sound is passed through to the sound card through existing capacitors/traces unamplified, so the PC will continue to "see" it … just at a very reduced level.

Tigertronics publishes a minimal (incomplete in my opinion) set of specifications for their device, so technically one can't say that it *doesn't* meet spec. I looked over the issues, and decided that the ones that mattered to me could be fixed for less than $0.20, and I that would have fun doing it. While doing research on the device, I discovered Peter Frenning's (OZ1PIF) website, which is *the* place to go about the Signalink USB's® noise issue. After researching everything I decided on a game plan to modify the interface.

The most important modification involves removal of the internal electrical noise. There are several well-known ways to buffer the bias op-amp bias voltage from the noisy 5V-USB line: insert an op-amp unity gain amp, install a massive decoupling network, use a voltage tracker or voltage regulator, etc. I chose to use regulated supply … because there is already an on-board low-noise supply at 3.7V for the comparator in the PIC microcontroller and the Audio USB IC A/D reference. I only had to add one jumper and replace one resistor (to put the bias back at 2.5V), and the noise was gone. This solution only cost $0.02. The figure below shows the before (a) and after (b) results.



(A) Noise Floor Before Mod (B) Noise Floor After Mod

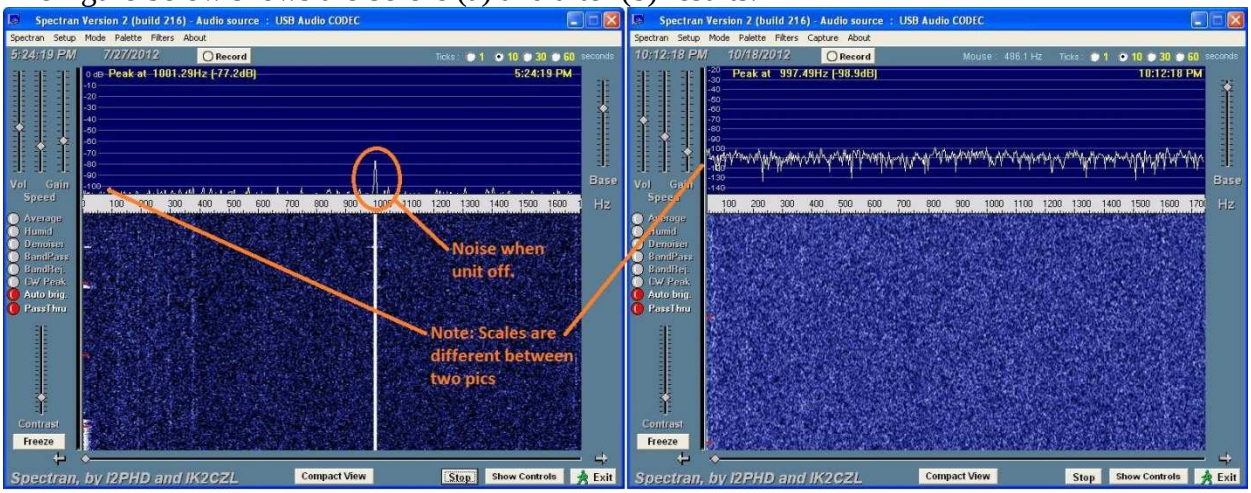
Alternatively, OZ1PIF chose to increase the decoupling capacitance on the voltage divider, while KJ7HJ chose to put a 3.0V regulator to replace the 2.5V divider supply. I also tried both of these methods and I can verify that all three will result in various amounts of reduction in noise. The approximately 10dB difference between them is probably moot for most people.

Sometimes when you fix one problem you uncover others (like peeling the layers off a stinky onion). There was still some noise being generated at 60Hz and its third multiple (180Hz), probably due to power-line interference. This is *only* 12dB up from the noise floor, and is undetectable by the fifth harmonic (300Hz). It couldn't previously be seen due to all the noise

generated by the poor bias supply design. I decided not to try to fix this, as at this low frequency it would be a lot of work without a lot of benefit. It turns out this would not be a problem at all, which I'll address in a minute.

As for the current sneak path (causing the 1000Hz noise when the unit was switched off), the most elegant solution would be to identify the paths and eliminate the current flows. This can often be done with series high impedance (resistors), buffer circuits, or diodes. However, there appeared to be at least three different possible sneak paths that I could immediately identify (plus the possibility of more that I missed). That is a lot of work to identify and cure each one.

An alternative is to treat the symptoms instead. The switch on the Signalink USB® is wired as a single-throw, but it is capable of being a double-throw. I cut two traces and added three jumpers. When the switch is now turned off, it will ground the 5VSW line, eliminating the possibility that the sneak path(s) will power up the rails. Now the device is quiet even when power is turned off. The figure below shows the before (a) and after (b) results.



(A) Noise when unit off, before mod

(B) Noise when unit off, after mod

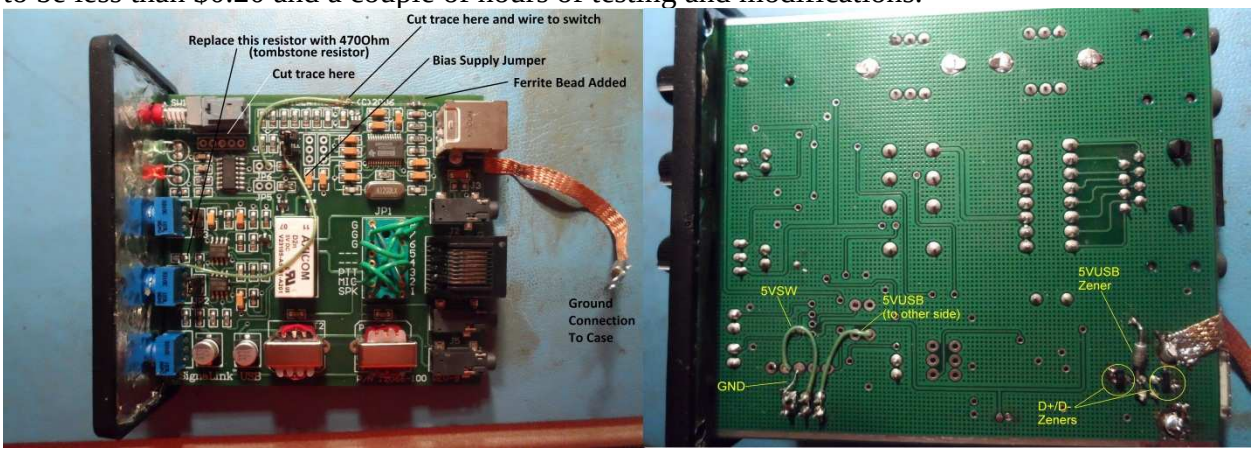
These following modifications were "optional" for me, since I didn't really have any problems with them with my current setup. But I did them anyway:

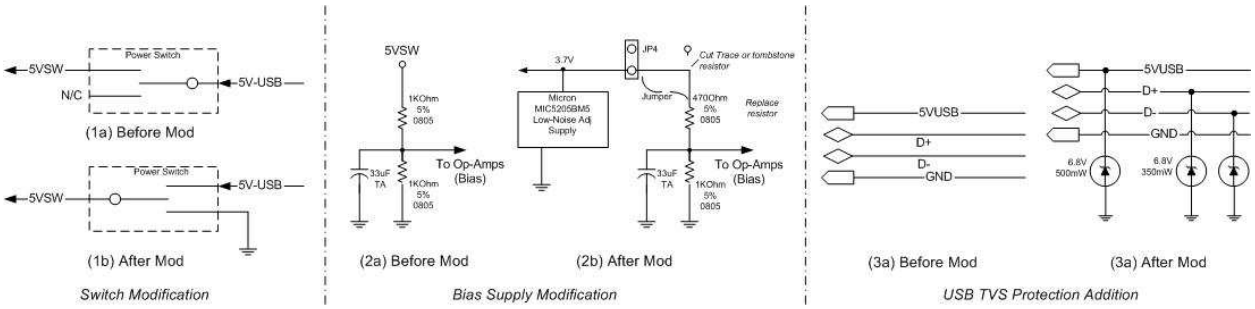
1. I put an appropriately sized ferrite chip of the right composition (for HF frequencies) on the USB line as a Just-In-Case (good design practice). This will attenuate RF between the computer and interface to some degree. Technically you should put them on both power and ground, but ground is a bit problematic with this particular layout. Alternatively you could use chokes (inductors) as OZ1PIF did.
2. To reduce the chances of issues when operating in an RF-rich environment (field day, for example) I put a short wire lead from ground directly to the case (to ensure good case shielding) and used aluminum foil to fully shield the plastic end caps (with a little glue to stick the foil to the inside plastic). The picture below shows the improved shielding solution.
3. I put 350mW 6.8V Zener diodes between the USB data lines and ground. I put a 500mW 6.8V Zener between the 5VUSB and ground. Note that adding Zeners on the data lines is only recommended for USB1.1 devices (not 2.0 or 3.0, which require lower capacitance solutions).

I did not bother to buffer the aux and speaker outputs (since I won't be using them). I also did not do anything to eliminate the fact that the device convert/send "sound" even when its power switch is off since this does not bother me.

After the shielding was completed, the 60Hz noise (and harmonics) disappeared into the noise floor. This confirms my assumption that it was likely radiated power-line noise. It was "accidentally" fixed when I was trying to improve overall RF shielding: sometimes it's better to be lucky than good!

With the changes mentioned above (with schematic changes shown below), the Signalink USB® ceases to be a sub-optimal solution and becomes an excellent sound interface (in my opinion), rivaling it's much more expensive competitors as well the alternative solutions (using a pro-sound card, for example). The new setup is much neater than my old, I can easily take the device with me on trips, I am not constantly tweaking settings, and the overall noise level is lower. In this case, "Good enough" just wasn't, but the difference between that and "Excellent" turned out to be less than $0.20 and a couple of hours of testing and modifications.





Notes:

1. I used two "free" tools during the analysis and testing. One is a SW program called "Spectran": this turns your sound card into a spectrum analyzer (technically an FFT analyzer). The other is a SW program called "Virtual MR1": this will allow your sound card to generate white noise, pink noise, sine waves, sweeps, etc.
2. All dB figures in this article are referenced only to full scale of the Signalink USB® A/D, and not to 1V (dBV) or "decibels unloaded" (dBu).
3. The Signalink USB® comes with a 90-day warranty from date of purchase. Performing these modifications before the 90-day period is over will likely void your warranty. Modify your gear at your own risk: I assume no liability if you break something.
4. All viewpoints are based on my personal experiences and opinions, and do not necessarily represent those of the GCARC, Tigertronics, or anyone else for that matter.
5. The reverse-engineered schematics are available from the author upon request.