Cheap and Easy SDR

Get your feet wet with the latest technology for the price of a night at the movies.

Robert Nickels, W9RAN

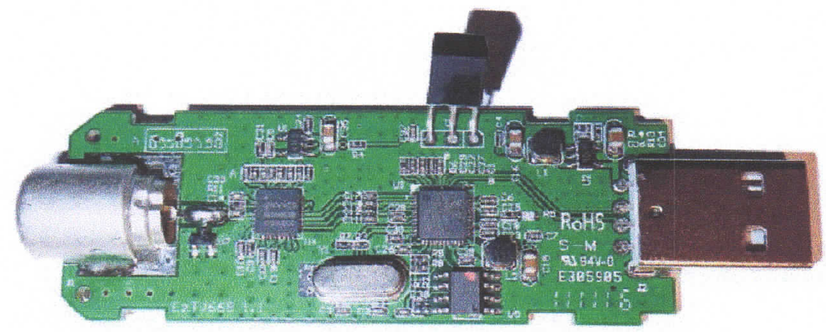

Figure 2 — Inside view of a typical DVB-T dongle. The RF connection is on the left, a USB connector on the right, and in between is an SDR.

There's little doubt that software defined radio (SDR) will be a big part of ham radio for a long time. While many experimenters and homebrewers enjoy working at the "bleeding edge" of this new blend of RF and software, SDR is still a mystery to many hams. This article describes how you can put together an all-mode software defined receiver that covers nearly from dc to daylight for less than $50, and a VHF/UHF version for about half that.

Digital Broadcasting — A Game Changer

The television broadcast world has undergone a momentous change from analog to digital broadcasting. DVB-T (digital video broadcast — terrestrial) has emerged as the standard for digital broadcast transmission used by nearly 100 countries throughout the world (mostly in Europe, Asia, Australia, and parts of Africa). North American TV stations now broadcast using ATSC (Advanced Television Systems Committee, North American Standard). But what does this have to do with cheap and easy SDR?

The answer is what's become known as the "DVB-T Dongle" — a small, inexpensive USB "stick" (flash drive) that was developed to allow DVB-T viewing on a standard PC or laptop (see Figure 1).

Figure 1 — DVB-T dongles are all you need for a V/UHF SDR. They come in various shapes and sizes.

A *Linux* developer in Finland named Antti Palosaari, who was working on digital television drivers for *Linux*, noticed something very interesting buried in the instruction set of the Realtek RTL2832 quadrature decoder chip that's inside every DVB-T dongle.

A special *Radio* mode, which was intended to allow the dongle to be used for FM broadcast reception, actually caused the device to output a stream of 8 bit I/Q samples at a rate of up to 2,000,000 samples per second. He quickly saw the potential for a cheap and easy SDR, and with the enthusiastic help of other developers from the Osmocom (Open Source Mobile Communications) group, a set of basic drivers and utilities to communicate with the complex ICs in the dongle was developed for both *Linux* and *Windows*.

To understand why this is a big deal, we'll need to delve slightly into how a SDR works. Many hams have played with a simple direct conversion receiver, in which a local oscillator very near the desired frequency is mixed with the RF signal to produce an audio signal without any intermediate conversion or IF stages. Since one type of SDR also uses this *zero frequency IF* approach, the Osmocom developers soon had the DVB-T dongle spitting out a stream of I/Q data in the same format used by existing SDR applications. While other inexpensive front ends such as the SoftRock rely on a PC sound card to convert this stream of data from analog to digital form, the RTL2832 quadrature sampling decoder performs this task at speeds up to 20 times faster, and without using a sound card at all.

This means that rather than being able to see, for example, a 96 kHz wide segment of an amateur band using the sound card approach, the DVB-T chip can digitize a slice of the RF spectrum more than 2 MHz wide. While each sample has only 8 bits of resolution, it turns out that this is not a limitation for most uses.

This SDR starts at VHF

Recalling that the DVB-T dongle was created for watching digital TV, it's not surprising to learn that it doesn't cover the HF bands. But it does include a synthesized tuner chip (initially the Elonics E4000, see Figure 2) that provides continuous coverage from 64 through 1700 MHz (with a gap between 1100 and 1250 MHz). The *RTL_TEST* utility can be used to measure the actual coverage range of each dongle, as exact coverage can vary a bit.

A few pieces of open source software turn the DVB-T dongle into an all mode SDR that covers all VHF and UHF bands, and an easy to build HF converter can extend the coverage range from the AM broadcast band up through 6 meters.

Build Your Own Cheap and Easy SDR

First, you'll need to get a DVB-T dongle. Since they aren't compatible with the North American TV standard, they aren't easily found in North America. Most of us have obtained ours through online sources such as eBay. Search terms like "RTL2832," "DVB-T" or "SDR" should produce a number of suppliers, just be sure to get one that uses the RTL2832 and E4000 combination for greatest coverage range, although drivers for the Raphael Micro R820T tuner have recently been added. The price range runs from $20 up. A partial list of dongles known to work can be found online (see the references).

Gathering the Rest of the Pieces

While you're shopping, pick up at least one coaxial adapter to fit your dongle that will accept a regular TV type F connector. Most will require what is described as "PAL Male to F Female," although technically it's the "Belling-Lee" connector that's been used for decades in areas where the PAL video standard was supported (see, for example, Radio Shack Model: 278-261, shown in Figure 3).

The DVB-T has good sensitivity, but there's no substitute for a good antenna. Outdoor antennas such as the discone, collinear vertical and log periodic (or even old TV antennas) will do a good job on the VHF and UHF bands, especially if connected via RG-6/U or similar low-loss coax. The system connectivity is shown in Figure 4.

An SDR needs a PC

It would hardly be "cheap and easy" if you had to go out and buy a new lightning-fast PC to play with SDR. The system described runs well on older hardware. In my experience, it typically runs at about 10% CPU utilization on my 3.4 GHz Pentium D machine, and 12% on a 5 year old Core Duo laptop. You should be okay with a modest PC running *Windows XP* or newer OS and with one available USB port.

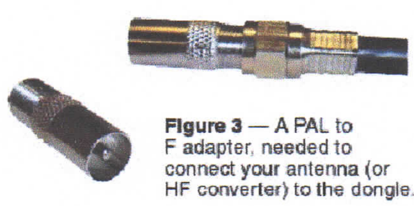

Figure 3 — A PAL to F adapter, needed to connect your antenna (or HF converter) to the dongle.

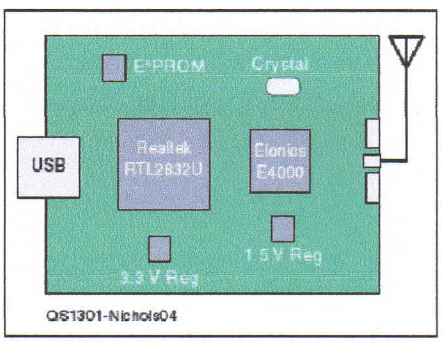

Figure 4 — Block diagram of the DVB-T dongle based SDR.

And the PC Needs Software

The first step is to get the dongle running on VHF/UHF. I suggest you read through the complete procedure, then use the ***Install*** script shortcut that will automate much of the procedure (you will still have to run the driven utility manually at least one time to install the proper *WinUSB* driver, but the script will automate the download of *Zadig* for you).

Driver Installation

You will need two pieces of software to install the proper driver. First, download *Zadig* from Sourceforge at **sourceforge.net/projects/libwdi/files/zadig/**. If necessary, download the *7Zip* archiver needed to install *Zadig* from **www.7-zip.org/**. Use *7Zip* to open the archive and install *Zadig*.

Plug in the DVB-T dongle, then run *Zadig*, the utility to install the custom driver. Abort the NEW HARDWARE FOUND dialog if it comes up. Now install the correct driver with *Zadig*. In *Zadig*, click on OPTIONS and LIST ALL DEVICES.

You should now see a screen that looks like Figure 5. Click on the item that says BULK-IN, INTERFACE (INTERFACE 0). This is your RTL device.

Make sure WINUSB is shown in the box to the right of the green arrow — not LIBUSB or LIBUSBK.

Click on REPLACE DRIVER. A few seconds later you should receive a success message. This completes the driver installation, although you may need to repeat the above sequence if you use a different USB port, or for other reasons the driver linkage is lost. It should be a one-time event, however.

To double-check, open *Windows Device Manager* and you should see an entry that includes the words LIBUSB WINUSB BULK INTERFACE DEVICE (0). If so — great, you're done. If not, *Windows Plug-and-Play* may have automatically installed TV drivers under "Sound Video and Game Controllers." If so, these must be removed before the dongle will work as an SDR. Remove the driver using *Device Manager*, then re-run *Zadig* as described above. (There may be slight variations in syntax due to different versions of *Windows*).

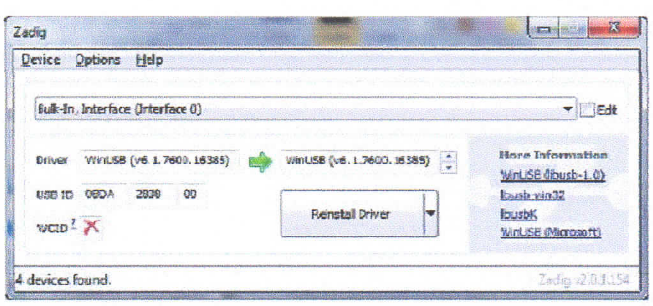

Figure 5 — *Zadig* main screen.

This completes DVB-T driver installation but we need one more thing: A *Radio* application or app!

A "Sharp" Approach to SDR Software — Radio Apps

A number of talented programmers have developed very comprehensive and powerful SDR applications (apps), and several of them have been modified to work with the DVB-T dongle as the RF front-end. While you may wish to try others, I will describe how to get started using *SDR#* (pronounced and sometimes spelled *SDRSharp*) — an SDR app that I've found to be simple, powerful and easy to use. It's also free!

SDR# is an open source software defined radio application for *Windows* created by Youssef Touil in Paris, with collaborative assistance from other volunteer software engineers around the world. *SDR#* is written in *C#*, a modern, general-purpose, object-oriented programming language developed by Microsoft. *SDR#* is intended as a DSP application for use with a wide range of RF hardware, including SoftRock, FiFiSDR, FUNcube Dongle, SDR-4, LazyDog's LD-1, SDR-IQ, SDR-14, RTL2832U / RTLSDR, any sound card based SDR front end and any external input/output based SDR front end. A script provides for one-click installation — see the link below. The following procedure is provided for reference.

Integrating the Software

The following steps must be followed to integrate *SDR#* and the necessary RTL drivers from Osmocom. Refer to the instructions at **rtlsdr.org/softwarewindows** for screenshots and a more detailed step-by-step procedure (all this is done automatically by the *Install* script).

- Download a copy of the "Continuous Integration" or "Dev" version of *SDR#* from the *SDR#* website: **www.sdrsharp.com** and unzip it into its own directory.

- Download the pre-built *Windows* zipfile RTLSDR binaries and libraries from Osmocom at **sdr.osmocom.org/trac/raw-attachment/wiki/rtl-sdr/RelWithDebInfo.zip**.

- From the zipfile directory **RelWithDebInfo.zip\rtl-sdr-release\x32** copy **rtl_sdr.dll** and **libusb-1.0.dll** to the directory. *SDR#* is unzipped to open the file *SDRSharp.exe. Config* in a text editor. Scroll down to the line that looks like <!-- <add key="RTL-SDR / USB" value="SDRSharp.RTLSDR.RtlSdrIO,SDRSharp.RTLSDR" /> --> and remove the leading <!-- and trailing --> so that it looks like:

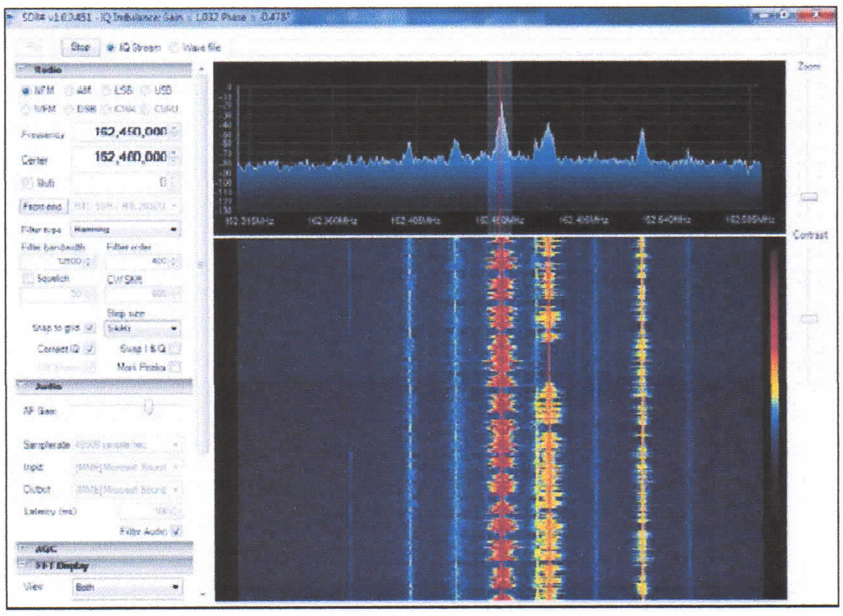

Figure 6 — *SDR#* installed screen showing broadcast FM spectrum.

<add key="RTL-SDR / USB"value="SDRSharp.RTLSDR.RtlSdrIO,SDRSharp.RTLSDR" /> and save the file.

- Download the *SDR# RTL Plugin* and copy the contents of the zip file to the SDR# directory.

This may seem like a lot of work, but it goes much faster than hours of metalworking, soldering and testing required to build a receiver from hardware. There is a shortcut that automates the above process, making it even easier. Download the install script from: **sdrsharp.com/downloads/sdr-install.zip** and run it, that should be all there is to it. Refer to the manual steps and websites above if you need further assistance.

The *SDR#* executable will be found in a new directory. Since the registry is not modified, *SDR#* files can be deleted, moved and reinstalled as desired.

Making it Play

Double click on SDRSHARP.EXE to launch the application — you should see a screen like that in Figure 6. Once you have completed driver installation, and have attached a VHF-UHF antenna to your dongle, go ahead and launch the *SDR#* program. Set RADIO MODE to WFM (wideband FM). From the topmost box, change OTHER — SOUND CARD to RTLSDR / USB. You'll notice the frequency and center boxes are filled in with a default frequency in the FM broadcast band. Click on PLAY. You should be greeted with a brightly colored spectrum display on top and a waterfall display below. As you change the frequency (or click and drag the spectrum display chart) you should be able to hear FM broadcast audio. Congratulations! You've just "built" your first software defined radio!

Using SDR#

The *SDR#* human interface is intuitive and flexible to use (see Figure 7). The PLAY/STOP button is found at the top of the screen, along with the ability to select live data from a connected IQ stream such as the DVB-T dongle, or you can play back stored files.

The balance of the *SDR#* screen consists of three functional areas. There are expandable control panels along the left-hand side, tuning controls and spectrum display at the top and a waterfall display at the bottom. In addition, there are controls for spectrum ZOOM and waterfall CONTRAST along the rightmost edge.

Mode selection is at the top of the RADIO control panel (see Figure 9), with TUNING and FILTER controls below. Default settings for everything else should be okay — go ahead and play — you can't hurt anything and this is the only way to learn what various features do.

Tuning with SDR#

SDR tuning is a little different, but for those who came up through the radio ranks with a *general-coverage* or *two-dial* receiver, it will seem familiar. The center box is analogous to the MAIN TUNING in that it determines which 2 MHz-wide slice of the RF spectrum is selected and visible in the spectrum display. The FREQUENCY entry shows the exact frequency the SDR is tuned to within this window, much like a BANDSPREAD control on a two-dial receiver.

Tuning can also be performed by using the mouse to click or drag within the SPECTRUM DISPLAY area of the main screen. Dragging the black background area will change the

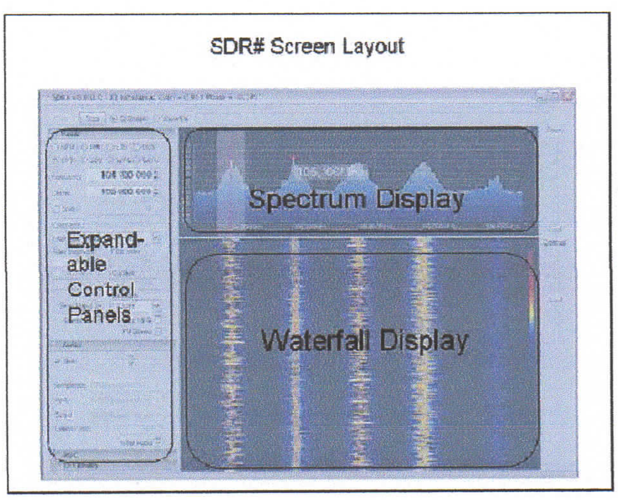

Figure 7 — *SDR#* main screen layout.

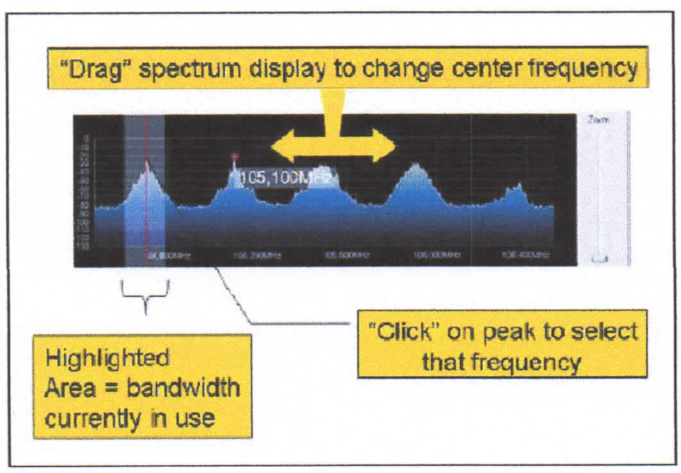

Figure 8 — Tuning with *SDR#*.

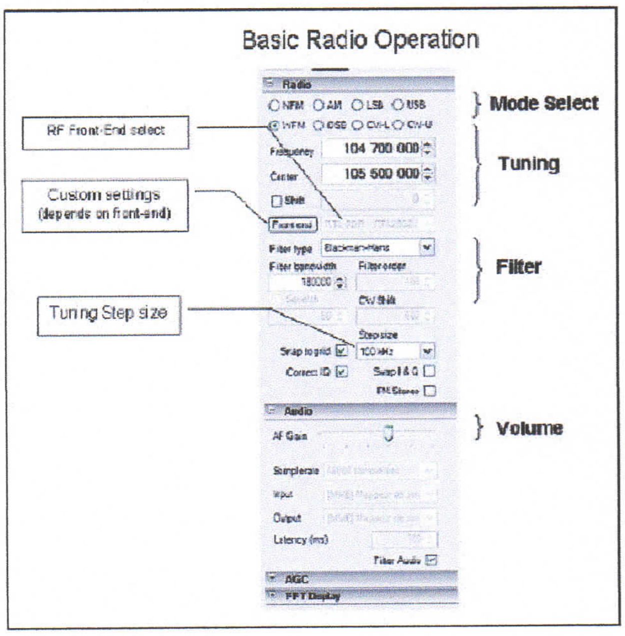

Figure 9 — *SDR#* radio control panel.

Cheap and Easy

Every new technology in Amateur Radio represents a step forward, and each time there are pioneers who help pave the way by making it possible for average hams to get involved in the latest technology. My personal interest in the early days of single sideband provides a perfect example — the March 1956 *QST* article "Cheap and Easy SSB" by Anthony Vitale, W2EWL, in which he described his easy to make SSB transmitter.

W2EWL based his design on the same phasing-type circuit that had been published before, and was in commercial production. He made use of a ubiquitous World War II *command set* transmitter, which was widely available as surplus for a few dollars at that time, and he simplified the circuit and found ways to use off-the-shelf components. The effect of this approach was to make it more attractive for an average ham who was interested in SSB to give it a try.

My hope is that by combining similar low cost hardware and proven designs, along with open source software and the ubiquitous *Windows*-based PC, this 21st century *Cheap and Easy* will encourage more amateurs (and potential future amateurs) to get their feet wet with the SDR, which I believe will have as much impact on the future of Amateur Radio as SSB did 60 years ago.

center frequency (see Figure 8), moving the 2 MHz window up and down in frequency. Clicking any signal peak within the SPECTRUM DISPLAY will instantly tune to that signal. The ZOOM slider on the right allows you to spread out signals and see more detail. Resolution is determined by the FFT (fast Fourier transform) setting selected in the FFT DISPLAY control panel.[1] The frequency manager provides unlimited memory capacity, and the ability to store and edit groups of favorite frequencies. Other options are available in experimental versions of *SDR#* and, if you're a *C#* programmer, there's a long wish list of future enhancements.

The Waterfall "Time Machine"

The most striking feature of any SDR screen is the colorful *waterfall* display. Previously used in applications such as sonar, speech processing and seismology, the waterfall (also known as a *spectrogram*) display gives us a brief look back in time as signals appear at the top and then scroll down and off the bottom of the display. The frequency of each signal is the same as in the spectrum display above, so the waterfall will show *wiggles* as we click and drag the spectrum display to change frequency. Since the waterfall is a two-dimensional view (frequency vs time), the amplitude of each signal is depicted through the horizontal width and color of the line painted under each signal in the waterfall. After a few minutes of adding the visual sense to your radio listening, you'll wonder how you ever got along without it.

Black indicates no signal, and shades of blue, yellow and red correspond to increasingly higher received signal levels. A very strong signal will be almost solid red, while a weak signal will consist of mostly yellow and blue. Modulation type can be determined from the waterfall display as well, in fact the dits and dahs of a CW signal can usually be read vertically as the display scrolls.

The rest of the radio controls are pretty self-explanatory. It's great to be able to tailor the filter as you wish and gain familiarity with the magic of using software to do what most of us still think of as the domain of capacitors and resistors. And if you find that the performance isn't quite up to the level of a commercial rig, just remember that the RF part of this receiver can be easily lost amongst your pocket change.

Adding HF Coverage

The first thing any ham does when confronted with a new radio is to spin the tuning dial. Unfortunately, the lower end of the DVB-T dongle is around 64 MHz, well short of six meters and the popular HF ham bands. But that is easily fixed, using technology that has been an essential part of radio for over 100 years!

The HF Converter

A radio frequency *mixer* is a device that accepts two different frequencies as inputs, and creates the sum and difference of them as outputs. It is shown schematically in Figure 10. Think of it as mixing two colors of paint. If we combine yellow and blue, for example, the result will be green. A good RF mixer will combine two radio frequencies so completely that we see only the *green* output, with no yellow or blue left. For our HF converter, the INPUT signal will come from an HF antenna, such as a longwire or dipole. For the local oscillator signal I've chosen 125 MHz because a preassembled CMOS oscillator provides a "cheap and easy" solution, with the 0-30 MHz HF bands moving up to 125-155 MHz. This frequency eliminates the possibility of interference from FM radio stations.

We can use the heterodyne mixer to build a frequency converter for the HF range (and beyond). Table 1 shows the relationship between the LO, Input and Output for frequencies in the 80 and 10 meter ham bands.

The SDR software will automatically take care of the arithmetic when the converter is in use so we can have a direct frequency display.

HF Converter Design

There are many ways to create a heterodyne mixer, but the NE-602 and the improved NE-612 integrated circuits were created for exactly this purpose. They have been proven in countless designs. It provides high sensitivity, a low noise figure and low-cost in a single device that uses a *Gilbert cell* circuit that cancels unwanted signals and produces clean output. I found that using a standard CMOS oscillator module was the best way to provide the 125 MHz local oscillator signal.

[1]Greater resolution requires more CPU horsepower, so the basic rule is to set the resolution as high as needed, but no more. For most uses, 4096 is sufficient.

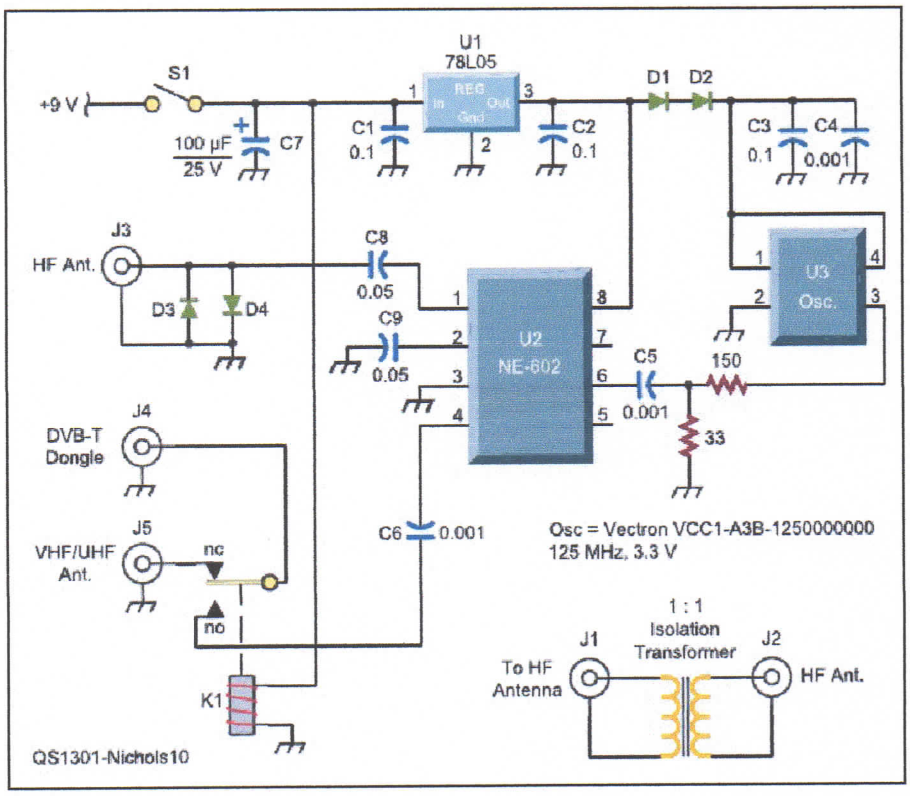

Figure 10 — Schematic diagram and parts list for the HF upconverter for SDR operation down to below the AM broadcast band.

C1-C3 — 0.1 µF, capacitor.
C4-C6 — 0.1 µF, capacitor.
C7 — 100 µF, 25 V electrolytic capacitor.
C8, C9 — 0.05 µF, capacitor.
D1-D4 — 1N4007 silicon diode.
J1-J3 — RCA type panel jack.
J4, J5 — F-type coax panel jack.
K1 — SPDT relay, 5 V coil. Omron G5LA-14-DC5 or equivalent.
R1 — 150 Ω, ¼ W resistor.
R2 — 33 Ω, ¼ W resistor.
S1 — SPST miniature toggle switch.
T1 — RF isolation transformer, 2 turns each side on BN-43-2402 binocular ferrite core.
U1 — LM7805 voltage regulator IC.
U2 — NE-602 mixer IC.

The converter needed for SDR has one big difference when compared with a typical converter hams are familiar with. Normally we would add tunable filters on the input to pass only the signals of interest and reject everything else. Since we want to view the entire HF band, I found that a simple wide-band untuned input and output circuit worked well, at least in my location. These design decisions greatly simplify the converter, but those who are located near strong RF sources (such as broadcast transmitters) may need to add some external filtering.

Although the AGC in the tuner works well, it can be overloaded by strong signals. Click CONFIGURE to turn off the tuner AGC and manually adjust the RF GAIN slider as needed for optimal reception.

Figure 10 shows the schematic and the parts list of the HF converter using an NE-602 mixer IC. Separate antenna connections are provided for HF and VHF-UHF antennas. A relay is used to automatically connect the converter if HF operation is desired. If converter power is off, the normally-closed contact routes VHF/UHF signals directly to the DVB-T dongle for normal operation. This makes it possible to locate the converter any desired distance from the DVB-T dongle and select between HF and normal operation by simply switching the power to the converter on or off. The converter can be built on a small PC board, or using perforated project board.

Table 1
Relationship Between the Local Oscillator (LO), Input and Output Frequencies for the 80 and 10 Meter Amateur Bands

Input Frequency (MHz)	LO Frequency (MHz)	Output Frequency (MHz)
3.58	125	128.58
28.5	125	153.50

Figures 11 and 12 show the completed upconverter board and the board in its cabinet. Additional details are on the QST In Depth website (**www.arrl.org/qst-in-depth**).

Inexpensive RG-6U coax and F type connectors are used, as they provide low loss through the UHF range. Input signals from the antenna are coupled to pin 1 of the NE-602 through a broadband coupler and back-to-back diodes that provide isolation and protect the input from static discharges.

Through hole components are used for everything but the 125 MHz CMOS oscillator module, which is only available in a surface mount package, but this component is easily attached with a small-tipped soldering iron. Simply apply a small amount of solder to the pad area, then hold the part in place while reflowing the solder to one pad to hold it in place, then add a bit of solder and reflow the remaining pads. The 125 MHz output from the module is reduced by means of a voltage divider to the proper level required by the NE-602. PC boards are available from FAR Circuits at **www.farcircuits.net**.

Any source of dc power from 9-12 V can be used, but I strongly urge the use of a linear rather than switching type power supply to minimize the potential for switching noise that can cause interference in the converter's output. The 5.0 V output from a 7805 regulator powers the NE-602 and is reduced by approximately 1.4 V by two series diodes to provide the 3.3 V needed by the oscillator. The oscillator draws approximately 50 mA, so a well regulated supply is needed.

Interconnecting wires between the converter PC board and the connectors must be kept short to prevent undesired pick-up of undesired signals. Use fully shielded coax such as RG-6U for antenna interconnections. A USB extender cable may be used to move the DVB-T dongle away from the RFI-noisy PC. As with any receiver, a suitable antenna should be chosen for the type of reception desired on the VHF/UHF bands. For HF, a simple longwire or dipole located away from noise sources will produce good results.

Operating the SDR

Once you experience the fun of having spectral and waterfall displays, it will be hard to go back to using a conventional radio. With just a few hours of use it becomes easy to identify various types of signals, even to the point of recognizing which sideband is being transmitted. It will become easy to click the SDR exactly on frequency.

Using the SDR on HF is no different than on the higher direct frequency range, except everything is shifted up by 125 MHz. While it's

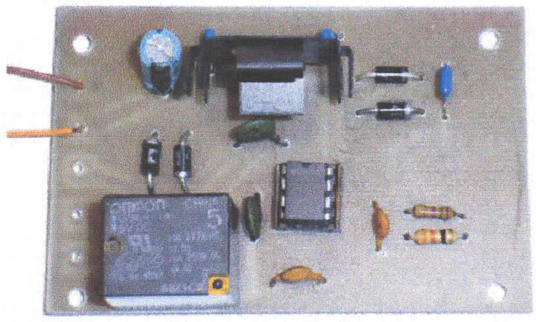

Figure 11 — Assembled upconverter board — ready for final assembly.

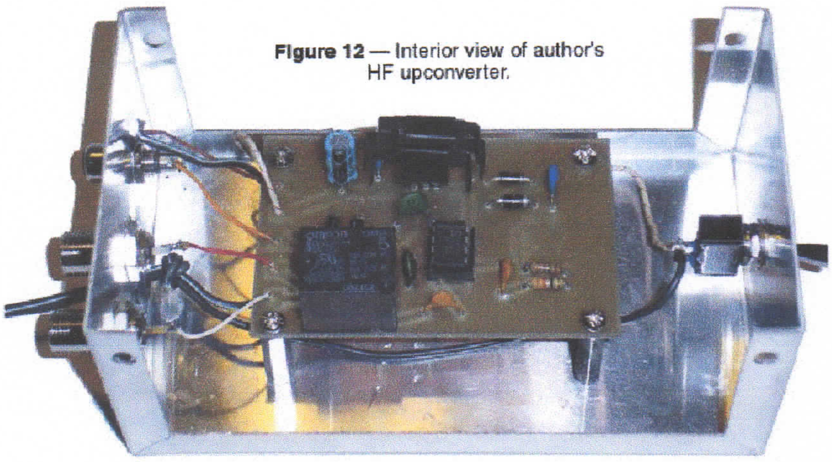

Figure 12 — Interior view of author's HF upconverter.

pretty easy to "do the math." *SDR#* also includes a provision for a frequency shift that will automatically be added or subtracted from the frequency that is displayed and stored in *Frequency Manager*. So, if we enter "–125000000" as the shift, the frequency will be displayed correctly (we use a negative offset because this is the amount to be subtracted from the displayed reading).

As with normal SDR listening with the DVB-T dongle, a slice of radio spectrum up to 2 MHz wide is presented at one time, equally centered about the selected center frequency. So to begin with, Enter "135000000" in the CENTER FREQUENCY box (note the offset is not applicable to center frequency). In normal operation, we would be tuned to 135 MHz in the aircraft band, but with the HF converter switched on, the center frequency is now 10 MHz. With the ZOOM slider all the way down, most North American listeners should see a fairly tall spike near the center of the display. Make sure the AM mode is selected and click on the spike — if you're lucky you will be hearing the sounds of the US NIST time and frequency standard station, WWV from Fort Collins, Colorado.

I've written a guide that explains more about the features of *SDR#* (see references).

The SDR Difference

Few have ever actually seen the entire HF spectrum all at once, but a panoramic view of the whole HF spectrum is possible. It is fascinating to visually see the HF spectrum unfold, and to identify signal types by sight. While I'm not ready to give up my tuning knob, I'd be lost without this added dimension while cruising the bands. You will quickly become adept at visually identifying signals and modulation types, and using the spectrum display to spot unknown signals or open frequencies.

The core of every SDR is the digital signal processing (DSP) software that is used to demodulate and filter the incoming I/Q stream. *SDR#* gives you a choice of several filter algorithms and infinite control of bandwidth to suit your taste or band conditions. Likewise, the waterfall and spectrum displays can be customized as desired.

I suspect that hidden features of the RTL2832 mark the beginning of what promises to be a long journey into the world of SDR. We all remember the idiosyncrasies of our first Novice rig, but also the thrill of learning, and how much fun it was to eke the most performance out of inexpensive and simple gear. That experience was invaluable when it came time to upgrade, as we had a good idea of what features and capabilities to look for in our next rig. Through the power of software, new features and capabilities are only a download away.

For More Information

The following is a partial listing of compatible DVB-T device and Osmocom references.

Osmocom: **sdr.osmocom.org/trac/wiki/RTL-sdr**.

The official *SDR#* website: **sdrsharp.com.**

Shortcut to official download page: **sdrsharp.com/index.php/downloads**. (Note: "Continuous Integration" or "nightly" versions contain the latest enhancements and new features that are not described in this article.)

RTL2832 DVB-T dongle installation information: **RTLsdr.org/softwarewindows**. (Note: The above procedure must be followed before RTL devices can be used with *SDR#*.)

Bob Rich's standing download site for the latest version of his experimental code (including *Auto Tuner* and *Trunking*): **public-xrp.s3.amazonaws.com/Release-latest.zip**.

##RTLsdr and #sdrsharp on Freenode IRC

Yahoo *SDR#* group: **uk.groups.yahoo.com/group/SDRSharp/**.

My own collection of SDR-related info on Google Docs: **tinyurl.com/bisg2or**.

My *SDRSharp* user guide and other information can be found at **goo.gl/suS2w**.

ARRL member and Amateur Extra class licensee Robert Nickels, W9RAN, was first licensed at age 14 in 1965 as WN0OHO in Nebraska. He has a BS from Fort Hays State University in Kansas, and credits ham radio as a major influence during his 35 year career in the electronics manufacturing industry. A holder of three US patents, Bob recently retired from Honeywell, where he held positions as a principal engineer, engineering manager, and strategic marketing director. He currently heads up RAN Technology Inc, a business and technology consulting firm. An avid cyclist and cross-country skier, he enjoys ham radio history and homebrewing, in addition to his main interest — collecting, restoring and operating a growing collection of vintage electronics and boat anchor radios from the last five decades.

You can reach Robert at 2645 East Dr, Freeport, IL 61032 or at w9ran@arrl.net.

For updates to this article, see the *QST* Feedback page at www.arrl.org/feedback.