

# Low-Cost 2.4-GHz Spectrum Analyzer

*Forget dropping big bucks on a microwave spectrum analyzer. You can build your own for a fraction of the cost. Scott walks you through the process of building an ATmega48-based 2.4-GHz spectrum analyzer.*

If you've shopped around for a microwave spectrum analyzer, you know that a basic unit costs around \$3,000. The prices shoot into the stratosphere from there. Wouldn't it be great if you could build a basic spectrum analyzer for, say, \$50 in parts? Well, that's exactly what I'll be describing in this article (see Photo 1).

## WHY A SPECTRUM ANALYZER?

As UHF and microwave RF designs become more popular, you frequently need to "see" what's being transmitted or received. Most of us are used to observing signals in the time domain with an oscilloscope; however, most oscilloscopes don't have the bandwidth to see microwave signals. In addition, they aren't much help in determining what a signal looks like in the frequency domain. This is where spectrum analyzers come in.

In a perfect world, all radio transmissions would be free from interference from other devices operating on the same frequency. In reality, though, many wireless devices must coexist in the industrial, scientific, and medical (ISM) bands. These portions of the frequency spectrum are allocated for unlicensed, low-power, short-range operation. Examples of devices operating on ISM bands are cordless phones, Wi-Fi networks, Bluetooth devices, and cordless mice for PCs. If you're designing a wireless system that must coexist with these potential sources of interference, you must under-

stand how these transmissions relate to your own communications. If you've ever had the experience of your Wi-Fi network going down while you were talking on your 2.4-GHz cordless phone, you know what I mean.

If you're like me, you've had the experience of designing wireless systems without a means to debug transmissions. Is the transmitter sending its signal at all? Is it transmitting on the correct frequency? Using your 100-MHz oscilloscope on this 2.4-GHz signal will get you nowhere. My spectrum analyzer can go a long way in helping sort out these problems.

If you need to carry out calibrated, traceable measurements, or if you need to look for harmonics of 2.4 GHz, there's no substitute for a high-end spectrum analyzer with its high-end price tag. However, many aspects of wireless design and the process of conducting site surveys don't call for these features.

Even if you have more modest requirements, you must dig pretty deep into your wallet to acquire the right equipment. In the past, if you attempted to build low-cost, hobbyist-grade spectrum analyzers, you probably ended up with a bulky systems based on off-the-shelf modules like cable TV tuners, which cost several hundred dollars.

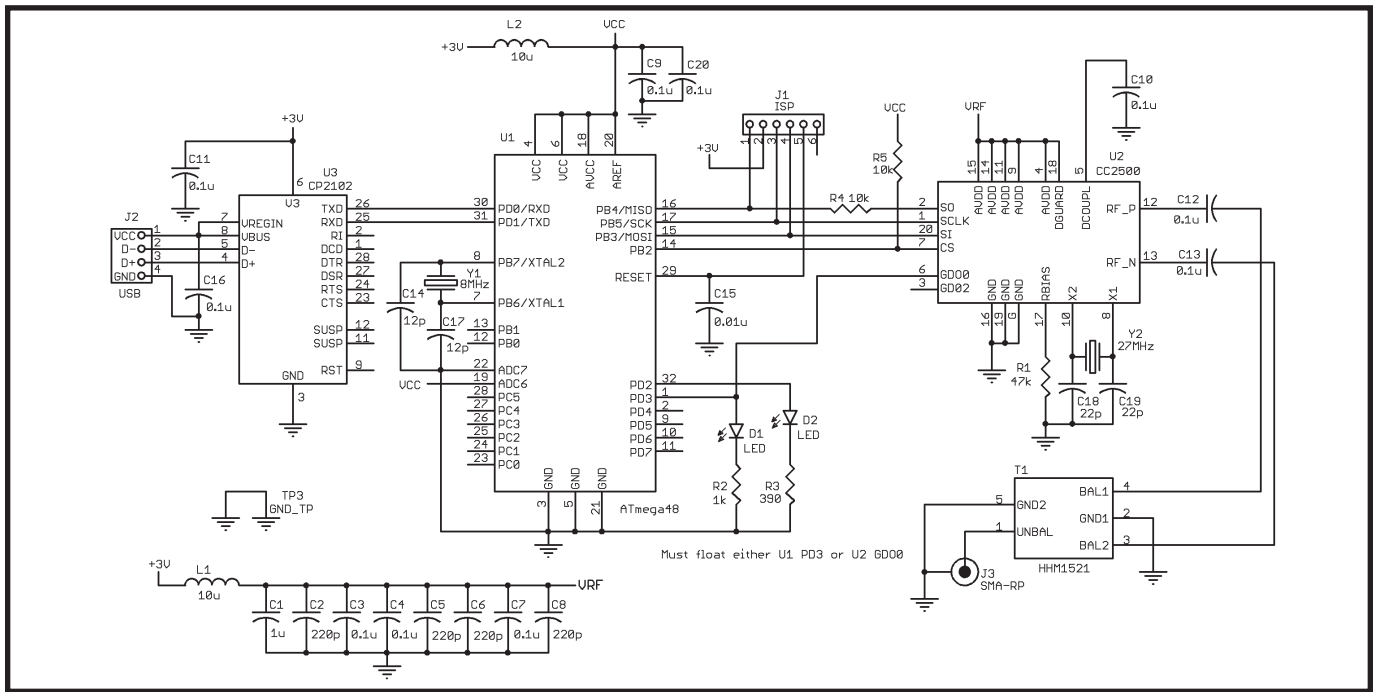
## PC CHIPS TO THE RESCUE

You can thank the Pentium processor line (and other similar chips) for some interesting new capabilities in wireless circuits. How does computer technology affect RF circuits? In the constant push for smaller feature sizes and faster clock speeds, chips for high-volume products like PCs set the standard for the semiconductor fabrication processes used throughout the industry. This benefits other chip product lines because smaller features mean more capabilities at lower costs. In addition, the faster circuits needed for today's PCs allow microwave frequency circuits to be integrated onto a single chip that in the past required many external components.

Examples of state-of-the-art low-power ISM band circuits are the recent offerings from companies like Xemics, Nordic Semiconductor, and Chipcon. If you're familiar with conventional radio circuits, you'll look at Figure 1 and ask, "Where are all the tuned circuits?" Unlike conventional tuners like the venerable superhet with its big IF coils, these new frequency-synthesized chips require virtu-



**Photo 1**—The spectrum analyzer consists of a small RF receiver circuit connected to a PC via a USB port. A custom Windows software application controls the hardware and displays the spectrum.



**Figure 1**—The ATmega48 microcontroller (U1) controls the RF receiver's (U2) functions. U1 is connected via a UART connection to USB bridge chip U3. J1 enables the microcontroller to be programmed in-circuit.

ally no external parts other than bypass capacitors, a crystal, and (sometimes) an antenna matching network.

In spite of these semiconductor advances, there are no dedicated, low-cost spectrum analyzer chips on the market right now. But if a chip has the right combination of features, you can use a radio chip intended for other purposes in your spectrum analyzer design. The basic approach for a swept-tuned spectrum analyzer is to tune a single-chip radio receiver to a given frequency, measure the signal strength, tune to the next frequency, and so on until you've acquired the signal strength at all frequencies in the spectrum. Plotting the spectrum is relatively straightforward after you have the data.

## WHICH COMPONENTS?

To build my spectrum analyzer, you need a receiver that's sensitive to frequencies higher than your range of interest (see Photo 2). If you're like me, you're interested in the 2.4-GHz ISM band, which extends from 2.40 to 2.485 GHz. The chip must be digitally tunable. You also need a way to know the strength

of the signal it's finding at each frequency (usually referred to as received signal strength indicator, or RSSI). Chipcon's CC2500 chip meets all of these requirements.

The CC2500 is a low-power 2.4-GHz transceiver with a sophisticated packetized data transfer and forward error correction. Using the CC2500 is serious overkill for this application, because you only need a receiver (not a transceiver). You don't even need to demodulate and receive data. But, the benefit is that it has a sensitive receiver with digital tuning and digital RSSI.

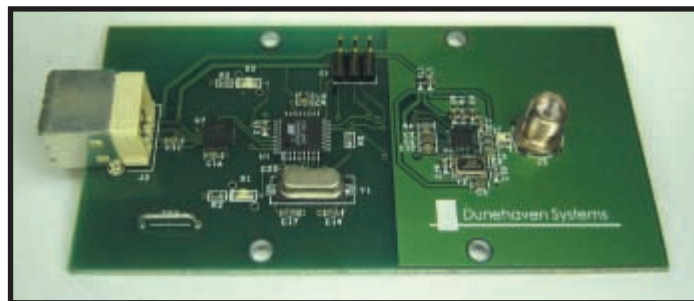
In case you're wondering if such a powerful chip will fit your budget, how does \$2 per chip sound? You can thank all the people demanding cheap PCs for the semiconductor technology

that makes this chip possible.

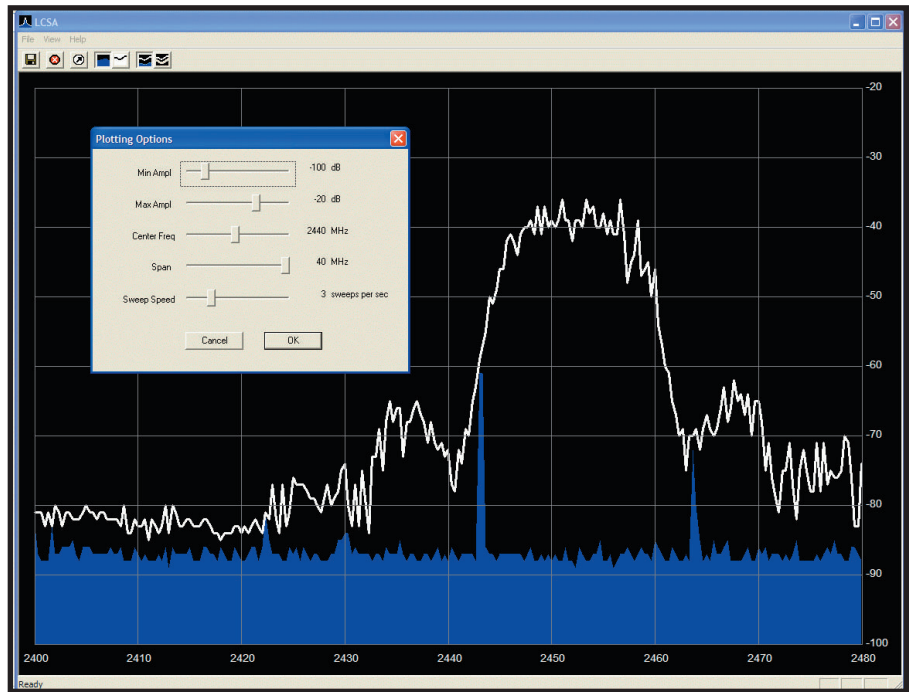
In addition to the CC2500, you need an embedded processor to control it. I chose Atmel's ATmega48, which has 4 KB of flash memory and 0.5 KB of RAM (see Figure 1). Its SPI interface communicates with the CC2500. Its UART allows data to be sent to and from the ATmega48.

Now that you've identified the hardware that will acquire the spectrum, you need a user interface to control the hardware and to display the results of your spectrum analysis. Rather than trying to reinvent an embedded system keyboard and display, I decided to use a commercial PC. You could use a hand-held device such as a Pocket PC or a larger machine running a desktop operating system. I used a laptop running Windows.

You also need a way to connect the PC to your microcontroller. Fortunately, this is easier than it was a few years ago. Silicon Laboratories (formerly Cygnal) makes a USB-to-UART bridge chip called the CP2102. Like the radio receiver part of this design, the CP2102 has a high level of integration, requiring only a couple bypass



**Photo 2**—The spectrum analyzer circuit board includes a Chipcon CC2500 transceiver chip. An Atmel ATmega48 processor and a CP2102 USB interface are also on board.



**Photo 3**—The accompanying Windows application displays the 2.4-GHz ISM band spectrum. The display updates continuously. You can adjust the horizontal and vertical scaling as well as the center frequency and sweep speed.

capacitors. Just connect the CP2102 to the ATmega48's UART and send serial data back and forth to the PC. In addition, the CP2102 has an on-chip voltage regulator that you can use to power the rest of your circuit, so no batteries or power supplies are needed. Everything is powered from the PC's USB port.

## PUT IT ALL TOGETHER

The PCB layout for this design is trickier than those for low-frequency circuits. Pay special attention to the layout of the RF section containing the CC2500 chip.

Chipcon provides a useful reference design on its web site. I adapted Chipcon's layout into a PCB that also incorporates the USB and microcontroller functionality. The RF portion of the board has topside and bottom-side ground planes for a solid RF ground. This enabled me to use a conventional two-layer PCB. Like many other modern circuits, most of the chips I used were only available in surface-mount packages. No wire-wrapping allowed in this project!

The CC2500 is configured for a 200- $\Omega$  balanced antenna interface. This can be directly connected to a suitable PCB loop antenna. For better performance, I used a whip antenna. In conjunction with my 50- $\Omega$  whip antenna,

my circuit includes a 200- to 50- $\Omega$  balun to couple it to the CC2500.

The simple software that runs on the ATmega48 passes commands from the PC to the CC2500 and sends responses back to the PC. This code, written in C, compiles to just 1 KB, well within the 4-KB flash memory size in the microcontroller. I used an ImageCraft ICCAVR compiler along with Atmel's AVR Studio programming tool, although other vendors' tools would work too. There is a six-pin header on the board that's configured to use Atmel's standard ISP programming interface. This allows the ATmega48's flash memory to be programmed in-circuit with a tool like the STK500 or the AVRISP, both of which are available from Atmel.

To help deal with the somewhat complex 60-plus registers in the CC2500, Chipcon offers free SmartRF configuration software, which helps determine the settings to program into its registers. For this project, I fixed the receiver bandwidth at 843 kHz (the maximum that is supported). The CC2500 allows the center frequency of its receiver to be set easily by defining channel spacing and channel registers. The center frequency is equal to the base frequency plus the channel

number multiplied by the channel spacing. For this application, I set the base frequency to 2,400 MHz and the channel spacing to 333 kHz. By setting the channel to a value between 0 and 255, I can tune the CC2500 over the range of 2,400 to 2,485 MHz. This covers the entire range of the 2.4-GHz ISM band.

Slightly complicating things is the fact that you must recalibrate the CC2500's signal-processing path each time you change frequencies in your sweep. Rather than take the time to do this during the sweep, my software pre-calibrates and stores the calibration constants. Later, during the actual measurement sweep, the software transfers the calibration settings back to the CC2500. This enables the system to jump from one frequency to the next in as little as 100  $\mu$ s. A complete sweep of 250 measurements can be done in 25 ms. In many cases, it's preferable to sweep more slowly than this so you don't miss intermittent signals.

The CP2102 USB interface has drivers that make it look like a built-in serial port to any PC it is plugged into. Any PC-based software that can connect to a serial port can connect to the CP2102 just by pointing it to the correct COM port number. You need software on the PC side to control this serial port and graph the results.

I used Microsoft Visual C++ 6.0 to create this PC software. Like its more expensive test equipment counterpart, my spectrum analyzer software enables control of the sweep speed, center frequency, frequency span, and peak hold modes. There is also a way to export captured spectra to a spreadsheet for further analysis or plotting. You can also write serial port code in Excel or some other data collection software instead of using the PC application I designed.

## PUT IT IN PLAY

The Low-Cost Spectrum Analyzer (LCSA) PC software is a 32-bit Windows application that should run on any reasonably modern PC (see Photo 3). It has modest memory requirements. There is no special installation process. Just double-click the LCSA icon.

Prior to using the software for the first time, you must install a driver for the CP2102 USB chip. This is easy in Windows XP. Let Windows's Add Hardware wizard search the Internet for a driver after plugging the spectrum analyzer into the USB port. Alternately, you can tell Windows to use the Silicon Laboratories driver, which you may download from the *Circuit Cellar* FTP site.

After plugging in the spectrum analyzer, installing the CP2102 driver, and running LCSA, the software should immediately begin acquiring data and showing you the 2.4-GHz ISM band spectrum in real time. You can open the Options dialog box to adjust the amplitude and frequency scales. Toolbar buttons allow you to turn on and off Peak Hold, color amplitude bars, and the white and black backgrounds. The Save menu item will export a comma separated value (CSV) file of the currently displayed spectrum.

In the simplest situation, you can perform a single sweep of the band and see the spectrum almost instantly on the screen. However, this works well only for continuous wave (CW) signals in which the spectrum is always the same from one instant to the next.

In most ISM communication protocols, the carriers are turned on and off and/or hop to different frequencies during a transmission. Table 1 shows

Technology	Modulation	Channel center frequency	Channels	Channel bandwidth	Channel spacing
Wi-Fi (802.11b)	DSSS	2,412–2,472 MHz	13	22 MHz	5 MHz
Bluetooth (802.15.1)	FHSS	2,402–2,480 MHz	79	1 MHz	1 MHz
ZigBee (802.15.4)	DSSS	2,405–2,480 MHz	16	3 MHz	5 MHz
Wireless USB	DSSS	2,402–2,479 MHz	78	1 MHz	1 MHz
Cordless phones	Varies	2,400–2,483 MHz	Varies	Varies	Varies
Microwave oven	Pulsed CW	~2,400–2,500 MHz	–	–	–

**Table 1**—Check out how these transmitters operating in the 2.4-GHz ISM band (DSSS, FHSS, and CW) compare to each other.



some of the differences between popular 2.4-GHz protocols. For protocols labeled as direct-sequence spread spectrum (DSSS) or frequency-hopping spread spectrum (FHSS), you must use special techniques to get a true picture of the frequency spectrum.

If a transmitter is hopping between different frequencies as you are scanning across the frequency spectrum, you might miss a transmission at a particular frequency if you are monitoring a different frequency at that moment. For this reason, the microcontroller can be instructed to over-sample several times at each frequency and save the highest value it finds at each frequency. This is controlled by the Sweep Time parameter in the PC software application. Slower sweep speeds mean more samples are acquired at each frequency in the hopes of monitoring that frequency when the transmitter is transmitting on that frequency.

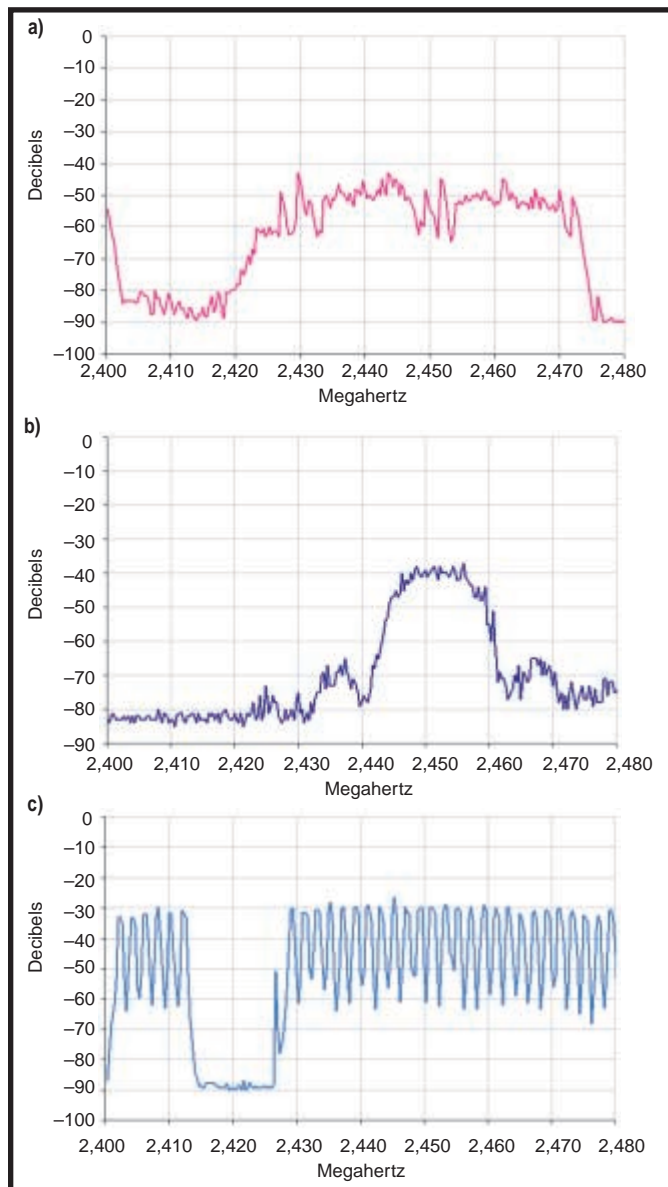
In practice, you'll try to strike a balance between sweeping so fast that transmissions are missed and sweeping so slowly that the transmission ends before the sweep is complete or that the screen updates become too sluggish. Typically, you'll need to capture the spectrum for somewhere between a few seconds and a minute to completely acquire a pulsed spectrum. Alternately, the spectrum analyzer can be put into Zero Span mode, during which it rapidly refreshes the amplitude at one particular frequency.

In addition to adjusting the sweep speed, you can remember the highest value you have found at each frequency. This is done by using the Peak Hold feature, which allows the true

shape of the spectrum to be accumulated after many sweeps of the frequency band. The maximum amplitudes at each frequency in the spectrum are saved until Peak Hold mode is turned off.


## RESULTS

I made the charts in Figure 2 in Excel after exporting the spectrum data from the Windows software application. Hopefully, you can appreciate how useful scans like these can be when you are trying to



**Figure 2a**—Even at a distance of 30', the sensitive receiver can detect a signal leaking from a microwave. Such signals can disrupt Wi-Fi communications in the area. **b**—The spectrum of a 802.11b access point operates on channel 9. The 22-MHz transmission centered at 2,452 MHz is visible, as are some side lobes. **c**—Take a look at the spectrum of a Bluetooth-enabled PDA searching for nearby devices (a so-called "inquiry sequence"). During an inquiry, a subset of the 79 possible Bluetooth channels are used. The channels are visible as peaks in the spectrum.

make your microwave radios work properly in the presence of other signals in a crowded spectrum.

As I explained, my circuit is designed for 2.4 GHz. However, Chipcon recently released a pin-compatible RF transceiver (C1100) that operates from 300 to 900 MHz. You can adapt my circuit to perform spectrum analysis for the UHF ISM bands using the CC1100. 

*Author's note: You may order a bare circuit board for this project or a completed and tested spectrum analyzer at [www.dunehaven.com/LCSA.html](http://www.dunehaven.com/LCSA.html).*

Scott Armitage ([scotta@dunehaven.com](mailto:scotta@dunehaven.com)) owns Dunehaven Systems, which is based in the Minneapolis area. He provides hardware and software consulting services to many companies, primarily in the areas of medical product design and embedded systems. To e-mail Scott, type "Circuit Cellar" in the subject line to get past the spam filter.

## PROJECT FILES

To download the code and additional files, go to [ftp://ftp.circuitcellar.com/pub/Circuit\\_Cellar/2006/189](ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/189).

## SOURCES

### ATmega48 Microcontroller

Atmel Corp.  
[www.atmel.com](http://www.atmel.com)

### CC2500 Multichannel RF transceiver

Chipcon  
[www.chipcon.com](http://www.chipcon.com)

### ANT-2.4-CW-RCT-RP Antenna

Linx Technologies, Inc.  
[www.linxtechnologies.com](http://www.linxtechnologies.com)

### CP2102 USB-to-UART Bridge

Silicon Laboratories, Inc.  
[www.silabs.com](http://www.silabs.com)