

# Compact Spectrum Analyzer

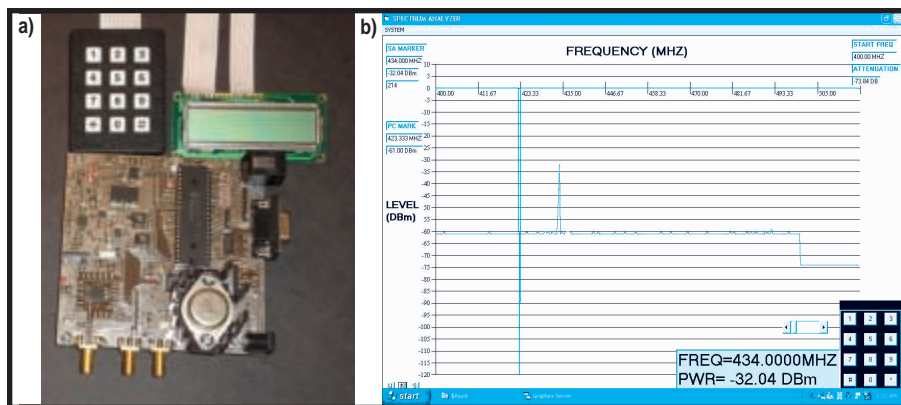
*If RF testing is in your future, you'll need a spectrum analyzer. But they don't come cheap. Your best bet is to follow Neal's lead by building your own 4" x 4" system.*

It is funny how simple beginnings can lead you down convoluted paths of learning and discovery. About a year and a half ago, I was playing around with my automatic garage door remote control unit. I was experimenting with signal encoding at the time, and I wanted to look at the scheme used in my remote. One thing led to another, and before I knew it, I was trying to determine the RF frequency at which the transmitter was operating.

I then built RF filters to look at signal levels and RF generators to aid in prototyping and testing. Next came RF mixing, frequency doubling, and so on. You can see where I'm going here. I became hooked on RF and all of the art that goes along with getting RF circuits working properly. Currently, all of my design-related pursuits are focused on circuits and test equipment at radio frequencies.

When it comes to RF test equipment, the spectrum analyzer is the Holy Grail. Unfortunately, both new and used RF spectrum analyzers are extremely expensive. There are a variety of home-brew spectrum analyzers available on the 'Net. Some are more sophisticated than others because all of the circuit elements are built from scratch. They feature complex filter constructions, have precise PCB layouts that turn PCB traces into inductors, and include a lot of discrete components. Other somewhat simpler designs feature integrated modules borrowed from the TV/VCR tuner world. These are easier to build, but they lack sophistication and functionality.

As I was browsing the 'Net for an alternative to a home-brew spectrum analyzer, I stumbled across the Maxim MAX3550, which seemed almost too good to be true. Not only



**Photo 1a**—My spectrum analyzer's PCB is shown here with an optional keypad and LCD, which I unplugged for clarity. The 4" x 4" PCB includes the complete spectrum analyzer with a 5-V power supply. **b**—The spectrum analyzer can be controlled as a stand-alone device from the keypad or from a virtual keypad using a mouse. The output spectrum can be a censored trace on an oscilloscope or a more elaborate PC screen display with a few extra whistles and bells. Alternatively, the power level at a single frequency can be displayed on the LCD.

does it perform much of the processing required to construct a spectrum analyzer, it's also highly integrated, simple to lay out, and easy to control. When I found this part, I knew I had to try to design yet another spectrum analyzer. The result of my effort is the single 4" x 4" PCB shown in Photo 1a.

You can use my spectrum analyzer board as a stand-alone system with its own keypad and display, or you can connect it to a PC. If you do the latter, the output will be displayed on the PC's screen and all of the keypad's functionality will be available with a mouse. You can also connect the spectrum analyzer

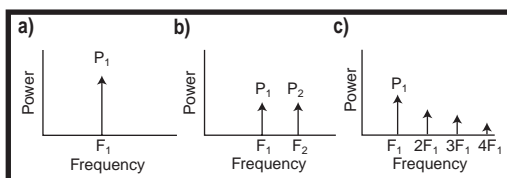
board directly to an oscilloscope to display the output spectrum. All of these combinations are possibilities. The circuit is relatively easy to build, and construction is pretty forgiving when it comes to the layout and part selection processes. Read on if you want a low-cost spectrum analyzer for your future RF endeavors.

## ANALYZER FUNDAMENTALS

A spectrum analyzer is used to display the power distribution of a signal as a function of frequency. This type of display is said to be in the frequency domain as opposed to being in the time domain as displayed on the screen of an oscilloscope.

To illustrate the value of a spectrum analyzer, let's look at some typical signals being analyzed with spectral analysis.

If the input to a spectrum analyzer is a pure sine wave, the output spectrum might look like what you see in Figure 1a. The arrowed line at  $F_1$  indicates that the frequency of the sine wave is



**Figure 1**—Compare the spectrum of a pure sine wave (a) to the spectrum of two equal power pure sine waves (b) and the spectrum of a slightly distorted sine wave (c).

$F_1$  and the power level is  $P_1$ . If you have an oscilloscope that operates at RF frequencies, you can get the same information from its screen. But oscilloscopes that operate at up to 1 GHz are extremely expensive.

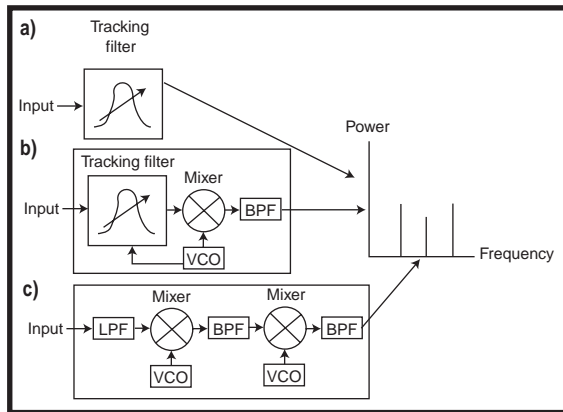
Figure 1b illustrates the spectrum analyzer output that occurs if your input signal is the sum of two sine waves of different frequencies and equal amplitude. There are two components to the signal, and the frequencies and power levels of each component are displayed. Even if you were to use a high-frequency oscilloscope, this information would be difficult to discern from the oscilloscope's display.

Yet another example of spectrum analyzer output is shown in Figure 1c. This is the output that results when a distorted sine wave is the input signal. The components at the various frequencies show the amount of harmonic distortion contained in the signal.

A spectrum analyzer facilitates other operations too. You can use one to determine filter responses, measure field strength, tune antennas, locate noise sources, and debug RF circuits. As soon as you have a working spectrum analyzer, you'll wonder how you ever accomplished anything without one.

## ARCHITECTURE CHOICES

Let's look at some spectrum analyzer architectures so you can better understand my design. Figure 2 illustrates three different spectrum analyzer archi-



**Figure 2a**—The center frequency of a narrowband filter is swept across the frequencies of interest. **b**—The VCO shifts the frequency of interest into the passband of the BPF. **c**—The two mixers shift the frequency into the passband of the BPF without needing a tracking filter.

tectures: swept filter, heterodyne with tracking filter, and double conversion.

The swept filter analyzer varies the passband of a band-pass filter (BPF) over the frequencies to be covered (see Figure 2a). It produces an output voltage that's proportional to the amplitude levels of the various frequency components. Although it appears simple, the direct implementation of narrowband filters variable across 1 GHz is difficult.

The heterodyne with a tracking filter approach is much easier to implement (see Figure 2b). It has been the basic approach used in radio receivers for years. Basically, a voltage-controlled oscillator (VCO) shifts the frequency of interest into the passband of the fixed-frequency narrowband BPF located at the output. The purpose of the tracking filter is to eliminate any signals located at the image frequency before the signal enters the mixing process. This is necessary because the

mixing process produces both the sum and difference frequencies, which would result in unwanted signals being shifted into the BPF's passband without the tracking filter. Because this tracking filter doesn't have to be too narrow to reject the image frequency component, it lends itself to an easier implementation than the tracking filter in the swept filter approach.

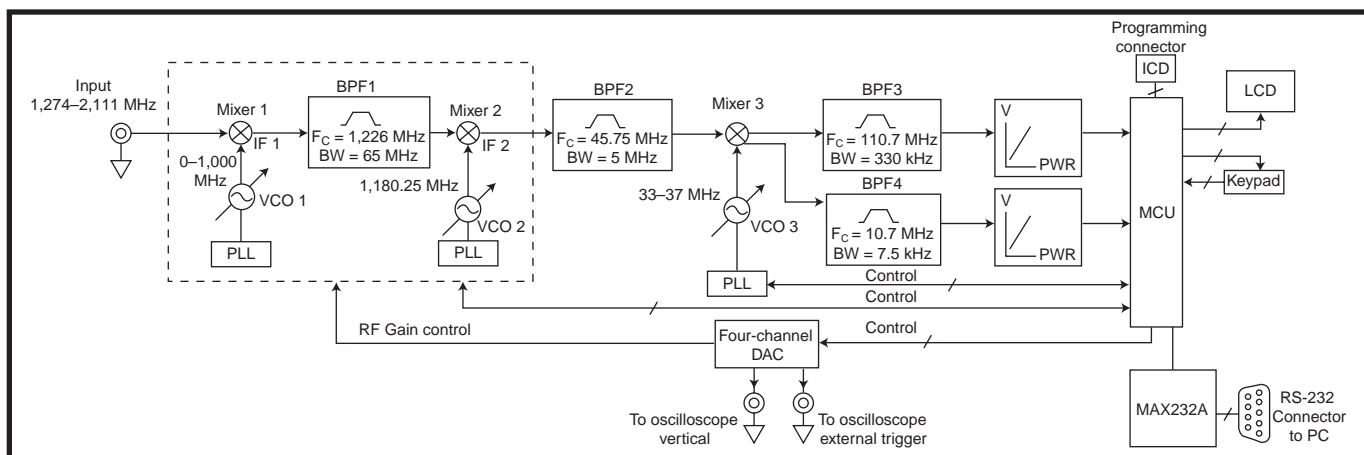
Modern high-frequency receivers use a double-conversion architecture (see Figure 2c). The beauty of this approach is that there are no tracking filters to deal with. The added complexity is a second mixing stage. The front end fixed LPF

removes components outside the highest frequency range of interest. The first mixing stage moves the signal frequency component of interest up in frequency into the fixed passband of the first BPF. The shift up in frequency causes the image frequency to be above the fixed passband of the input low-pass filter (LPF), so there is no image frequency content in the passband of the first BPF. The second mixing stage brings the frequency of interest back down. This allows extremely narrowband final filters to be used for maximum selectivity.

I used the double conversion architecture for my spectrum analyzer design. Why? Because it is relatively easy to implement fixed filters and variable-frequency VCOs.

## SIGNAL PROCESSING

My spectrum analyzer is fairly simple (see Figure 3). Diagrams showing



**Figure 3**—Three frequency conversion stages translate the frequency of interest into the passband of the final narrowband filters and log amplifiers. The PIC18F4520 microcontroller constructs the spectrum and drives the various output devices. The PIC18F4520 microcontroller also controls all of the conversions and gains.

what the frequency spectrum looks like at various stages of signal processing (assuming the analyzer is locked on an 800-MHz input sine wave) are posted on the *Circuit Cellar* FTP site. The front end of the process is basically the double conversion architecture. The first mixer/VCO moves the frequency of interest up to 1,226 MHz, the passband of the BPF1. For the 800-MHz hypothetical input signal, VCO1 is set to 2,026 MHz.

Notice that there is no LPF at the front end of the spectrum analyzer like that in the double conversion approach. The LPF needs to be only 2 GHz to reject the image frequency components. I didn't include an LPF because my applications typically don't have content higher than 2 GHz. If your application is different, you can add an external BNC-connected LPF. Also note that the VCOs are phase-locked-loop (PLL) controlled for

high stability and ease of control. More on PLLs later.

The second mixing stage shifts the frequency of interest down to 45.75 MHz, the center of the BPF2's passband. The BPF2's output is a 6-MHz wide region of the spectrum centered at 45.75 MHz. As you'll learn when I describe the hardware, many of the center frequency and filter bandwidth choices were dictated by the parts I selected.

I chose two resolutions for the spectrum analyzer: one for broad frequency looks and one for close frequency examination. The coarse spectrum resolution bandwidth is 330 kHz. The finer resolution bandwidth is 7.5 kHz. These resolutions are readily available in band-pass filters with center frequencies of 10.7 MHz. The problem is that the signal of interest is 45.75 MHz, so another conversion stage is required to bring the frequency of interest

down to 10.7 MHz. Mixer 3 provides the required down conversion.

The 10.7 MHz centered content is now delivered to BPF3 and BPF4, each with their respective bandwidths. The final task is to convert the output voltages of the two filters to power levels. Log amplifiers provide this functionality by producing an output voltage proportional to the log of the input voltage. The log amplifier outputs are delivered to the microcontroller for display. I used a Microchip Technology PIC18F4520 microcontroller for this project.

The PIC18F4520 microcontroller controls the frequency scanning of the RF portion of the spectrum analyzer. A coarse frequency stepping of 330 kHz is accomplished by varying the VCO1's frequency. The finer-resolution 7.5-kHz stepping operation is accomplished by varying VCO3. There are high- and low-sensitivity operating modes.

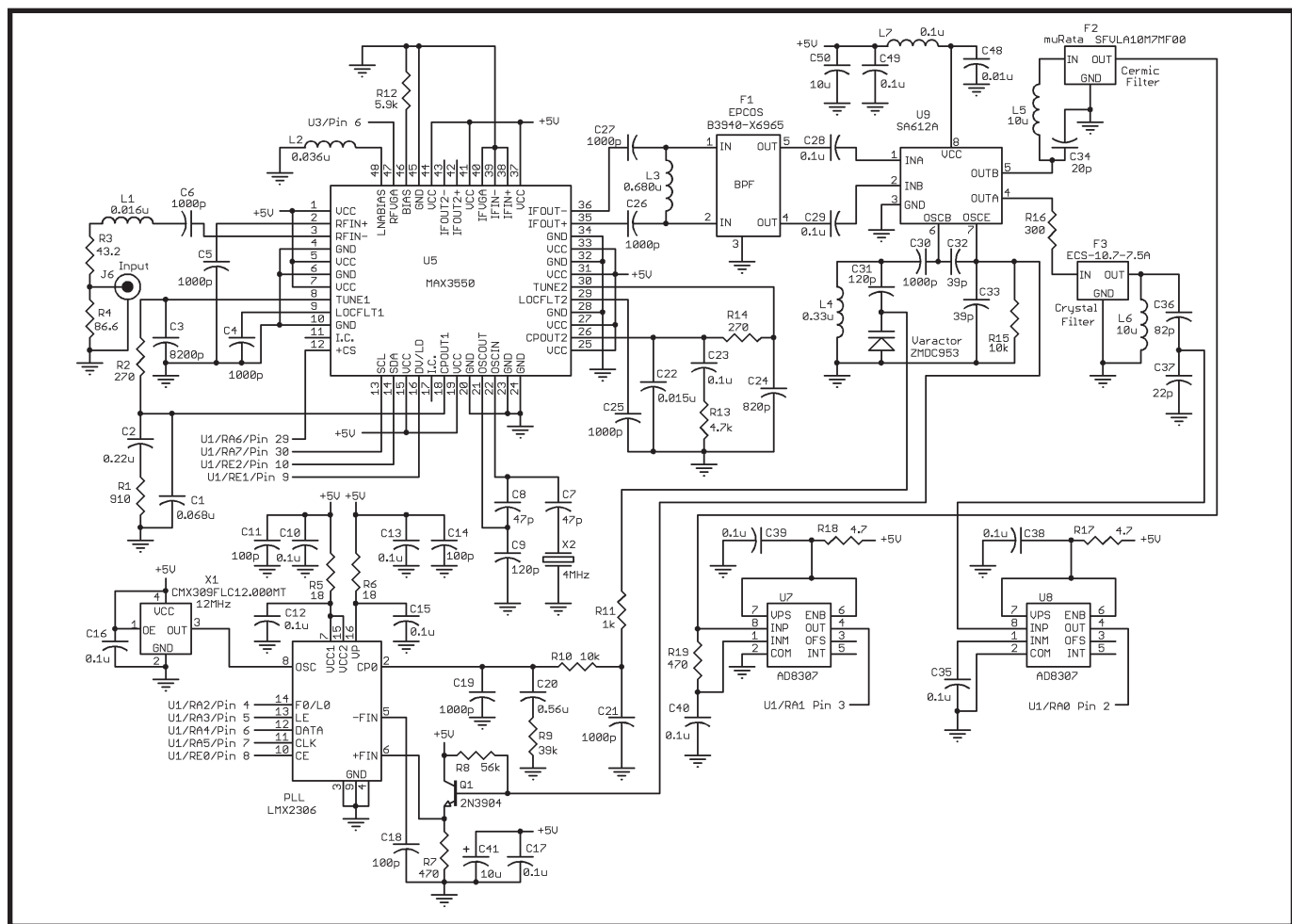


Figure 4—The MAX3550 integrates many of the spectrum analyzer's front-end functions. The Philips Semiconductors SA612A mixer shifts the signal of interest to 10.7 MHz for filtering. The Analog Devices ADB307 amplifier performs the conversion to decibels relative to 1 milliwatt (dBm).

These two modes of operation are controlled by varying the RF gain in the first mixing stage (also under the control of the PIC18F4520 microcontroller).

My spectrum analyzer can display its output spectrum on an LCD, an attached PC, or directly on an oscilloscope. The PIC18F4520 microcontroller uses a DAC to generate the signals to drive the oscilloscope. An RS-232 interface is included to communicate with a PC. The PIC18F4520 microcontroller also responds to commands from the optional keypad and drives the LCD to display the power level of a user-selected frequency. The system includes a connector so you can use a firmware development system to in-circuit program the microcontroller.

### HARDWARE FRONT END

Figure 4 (p. 62) shows the RF portion of the spectrum analyzer. The front end centers on the MAX3550 broadband TV tuner IC, which is a 48-pin QFN, 7 mm × 7 mm IC. This

is the IC that got me interested in taking a crack at a home-brew spectrum analyzer.

There are several ICs on the market that are said to have single-chip TV functionality, but I haven't found any with the level of integration and ease of use of the MAX3550. The chip contains all of the components necessary to perform double conversion tuning. It has the necessary variable gain RF amplifiers, double-balanced mixers, a band-pass filter, and PLL frequency synthesizers.

Other one-chip solutions for TV tuning require external VCO tuning inductors and external varactors. The MAX3550 has eight digitally selectable tank circuits and the necessary varactors on the chip. Maxim has somehow found a way to integrate the required inductors and capacitors on the chip, which isn't an easy task over the wide frequency operating range. The operating frequency range extends from 50 to 878 MHz, the standard TV frequency space, while providing 60-dB RF gain control. The response across

this frequency range is flat, typically 0.3 dB. The high-quality local oscillators have superior phase noise performance of -86 dBc/Hz at 10 kHz. The integrated filter achieves 68 dBc of image rejection. Device programming and configuration are easily accomplished with a standard three-wire interface to the microcontroller.

I am actually using the chip from 10 to 956 MHz, sacrificing some accuracy at the very high and very low frequencies. All that's needed to support this part are a handful of SMD resistors and capacitors to form the PLL loop filters and to do normal bypassing. A 4-MHz crystal is also needed. The MAX3550 requires a single 5-V supply.

Another thing that attracted me to the MAX3550 is that Maxim offers an evaluation kit for testing it. The beauty of this was that I could copy the circuit design in the kit, which ensured that the part would perform as specified. The evaluation kit also came with a piece of software to exercise the part using a PC. This enabled me

## Easy Embedded Linux

\$169

Qty 1



16MB FLASH / 32MB RAM

200Mhz Arm9 CPU

16 Digital I/O

Watchdog

10/100 Ethernet

Battery backed Clock/Calendar

Audio In/Out

2 USB

2 Serial Ports

*We brought you the world's easiest to use DOS controllers and now we've done it again with Linux. The **OmniFlash** controller comes preloaded with Linux and our development kit includes all the tools you need to get your project up and running fast.*

*Out-of-the-box kernel support for USB mass storage and 802.11b wireless, along with a fully integrated Clock/Calendar puts the **OmniFlash** ahead of the competition.*

Call **530-297-6073** Email [sales@jkmicro.com](mailto:sales@jkmicro.com)  
 On the web at [www.jkmicro.com](http://www.jkmicro.com)

## JK microsystems

# KEIL™

An ARM® Company

## Microcontroller Tools

### µVision® IDE

```

graph TD
    subgraph IDE
        A[µVision Project Manager] --> B[C/C++ Compiler]
        A --> C[Macro Assembler]
        B --> D[C/C++ Libraries]
        C --> D
        D --> E[Linker / Locator]
        E --> F[µVision Debugger]
        F --> G[Device Simulation]
        F --> H[Target Hardware]
    end
            
```

### RTOS

- RTOS Kernel
- TCP/IP Suite  
TCP, UDP, PPP, SLIP  
ARP, DNS,  
Ethernet, DHCP,  
HTTP, FTP, SMTP
- Flash File System
- USB Device Interface
- CAN Interface

Professional Tools for Over 1,000 Devices

- 8-bit: 8051 and Extended 8051 Variants
- 16-bit: C16x, XC16x, and ST10
- 32-bit: ARM7, ARM9, and Cortex-M3

New! RealView®  
ARM Compiler

800-348-8051
[www.keil.com/xd](http://www.keil.com/xd)

to copy the values that were used for setting up the various PLL divider registers and to ensure specified performance once again.

The BPF in the chip actually has a movable center frequency that's used to avoid a problem that can occur in the double-conversion process. It turns out that in the process, unwanted beat frequencies are generated from harmonics of the two local oscillators. The MAX3550 allows the center frequency of its internal BPF to be shifted slightly when these beat frequencies come into play. This moves the beat frequencies outside the output BPF's passband. I used the software that came with the evaluation kit to decide how to control the internal BPF's center frequency.

The MAX3550's output is fed into a 6-MHz wide EPCOS band-pass SAW filter centered at 45 MHz. This filter, which is contained in a five-pin SIP package, has an out-of-band rejection of higher than 50 dB, which is excellent. I used it

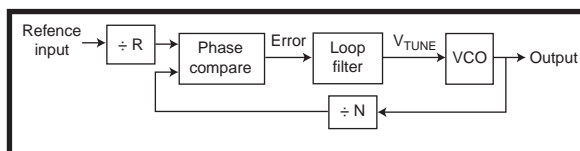
because it's the same filter used in the evaluation kit. I wanted to ensure there weren't any impedance-matching issues. The filter has a steep response and is designed for use with standard TV channel spacing.

### CONVERSION DOWN

The conversion of the 45.75-MHz centered signal down to 10.7 MHz is accomplished with a Philips Semiconductor SA612A double-balanced mixer/oscillator and a National Semiconductor LMX2306 PLL frequency synthesizer. The SA612A contains a differential input mixer with an input impedance that closely matches that of the EPCOS filter output, so no impedance-matching

components are required. The SA612A, which can receive inputs at -119 dBm, has a reasonable third-order intercept that's typically -13 dBm. It also contains a high-frequency common collector transistor that can be configured as a local oscillator for the mixing process.

Before I describe the actual circuit, let's look at what makes up a PLL. Figure 5 shows a PLL-controlled VCO. As you can see, the voltage  $V_{TUNE}$  controls an oscillator's frequency of operation. The VCO output is divided by factor  $N$ . A fixed frequency reference is divided down by factor  $R$ . These two divided-down signals are fed into the phase comparison box whose output is an error signal proportional to the difference in frequencies of the two divided-down signals. The error signal is passed through a loop filter.  $V_{TUNE}$  is changed until the divided-down VCO frequency matches the divided-down reference frequency. At this point, the loop is said to be in lock, and the VCO output remains stable locked to



**Figure 5**—An error signal is generated in the comparison of divided-down versions of the VCO output and a reference input. The loop filter conditions the error signal. It corrects the VCO tuning voltage until the output gets to the desired frequency.

## The New Rabbit

Designed For The Most Demanding Embedded Control Applications

- Speeds up to 58.98 MHz
- Ethernet with royalty-free TCP/IP
- Supports 16-bit memory devices
- 8 Channel 12 bit A/D
- Up to 6 serial ports
- Large NAND Flash
- 512K Flash
- 40 GPIO




**Next Generation Rabbit 4000**

**Kits From \$259**



Order Online At [rabbit4000.com](http://rabbit4000.com)

06160



# CIRCUIT CELLAR®

back issues available as

## Searchable Archives on CD-ROM

**NOW SHIPPING:**

CD-ROM #10 2005 Issues 174-185

CD-ROM #9 2004 Issues 162-173

CD-ROM #8 2003 Issues 150-161

Order Online:

[www.circuitcellar.com](http://www.circuitcellar.com)

or call 860.875.2199

the frequency dictated by the values set in  $N$  and  $R$ .

In the spectrum analyzer implementation of a PLL, the frequency dividing and phase comparison is accomplished by the LMX2306. The fixed reference is provided by a 12-MHz oscillator clock source. A 2N3904-based buffer delivers the VCO output signal to the LMX2306. This buffer is necessary because the LMX2306's +FIN input contains a prescaler that may produce some transients. The buffer keeps these transients away from the SA612A mixer. The LMX2306 contains several internal registers that control how it operates, all of which are easily interfaced to the PIC18F4520 microcontroller. For example, three registers ( $A$ ,  $B$ , and  $R$ ) control the PLL frequency. The frequency of operation is set as follows:

$$\text{Frequency} = (8B + A) \times \frac{12 \text{ MHz}}{R}$$

The phase error measured by the LMX2306 produces current pulses at its CP0 output that have a duty cycle proportional to the amount of frequency error. This CP0 output is fed into the loop filter made up of three capacitors and two resistors that are connected as shown to the CP0 output.

This loop filter's design is critical. It can make or break the quality of your PLL system. It influences the speed at which the loop responds to requests to change frequency (loop bandwidth). It also minimizes the effects of spurs generated by things like component leakage and current pulsing (spur gain). Lastly, it influences the output sine wave's stability after it's locked (phase margin/phase noise). There are books on the market that describe how to optimize the loop filter design.

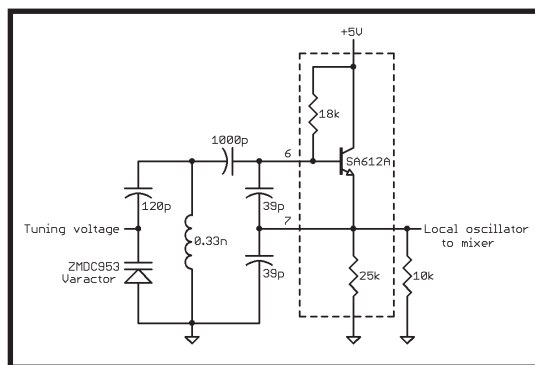
Fortunately, there is a much easier way to come up with a design for this loop filter. National Semiconductor's WEBENCH online electrical simulation tool will do all of the heavy lifting for you. Simply enter a few key parameters like the frequency range, the frequency step size, the reference frequency, and the power supply that you want to use, and

WEBENCH will essentially create the design. It will generate a schematic and a detailed parts list that includes vendor part numbers. WEBENCH will even tell you which VCO and PLL chips to order.

After the tool completes the design, it allows you to change part values and examine the effects on performance. All critical response, gain, and stability values for your design are calculated and graphed. As a final offering, you can even order a National PCB that contains the design in hardware form. This is one of the best online tools for a complex task that I have ever seen. National, give the hard-working engineer who put this thing together a big raise!

Unfortunately, the WEBENCH program didn't have a standard VCO to recommend that satisfied the frequency requirements for my spectrum analyzer. This required me to design a custom VCO that operated from 34 to 36 MHz. This tool enabled me to input the parameters describing my custom VCO. I then proceeded to design the loop filter as before.

The oscillator is a classic common collector Colpitts design that uses the biased high-frequency transistor internal to the SA612A as the active element (see Figure 6). It employs a Zetex Semiconductors ZMDC953 varactor to provide variable capacitance to tune the overall LC tank that sets the operating frequency. The capacitance of the varactor varies from approximately 26 to 84 pF as the tuning voltage varies from 4 to 1 V.



**Figure 6**—The spectrum analyzer uses a high-frequency transistor internal to the SA612A mixer to implement a VCO. The total  $C_{EQ}$  and the 330-nH inductor determine the oscillating frequency. Changing the voltage driving the varactor varies  $C_{EQ}$ .

The overall equivalent capacitance  $C_{EQ}$  resulting from the 120-pF capacitor in series with the varactor and the paralleling of the two 39-pF series capacitors (in conjunction with the inductor) determines the circuit's oscillating frequency:

$$F = \frac{1}{2\pi \times \sqrt{L \times C_{EQ}}}$$

When the  $C_{EQ}$  is around 62 pF, the circuit oscillates at the nominal 35 MHz.

Rather than adding more rigorous loop gain and start-up criteria to the design process, I went with the first-order look and used parts I saw in a similar design. The oscillator worked fine when I made the prototype, although the output amplitude was lower than the 200 mV<sub>pp</sub> needed to drive the SA612A mixer and PLL inputs. I just used brute force and added the 10-kΩ resistor from the emitter to ground. As a result, the output amplitude increased as needed. This resistor increases the circuit's transconductance ( $g_M$ ), thus increasing amplitude and ensuring start-up.

There's an important gotcha to consider when you're dealing with frequency synthesizers. So far, you've been under the assumption that the reference input (12 MHz in this case) is exact. In fact, if you use a standard reference source like the one used here, the frequency is typically off by several parts per million. The clock source I used was actually 12,000,080 Hz. Because a divided-down version of this is what's compared to a divided-down version of the VCO output, the VCO output frequency will be slightly off. If the reference signal is divided by  $R$  and the VCO output is divided by  $N$ :

$$\text{Frequency error} = (\text{Reference}_{IDEAL} - \text{Reference}_{ACTUAL}) \times \frac{N}{R}$$

This is relatively easy to account for in the firmware, so don't panic. Incidentally, this same analysis applies to the PLLs internal to the MAX3550. Any error in the 4-MHz clock driving the MAX3550 must be accounted for when coming up with the final frequency spectrum.

## FINAL SIGNAL PROCESSING

The 10.7-MHz centered signal at the output of the SA612A is simultaneously fed to the high- and low-resolution paths of the spectrum analyzer. An inexpensive ceramic filter accomplishes the lower-resolution 330-kHz bandwidth filtering. The more difficult higher-resolution 7.5-kHz bandwidth filtering is done with a more expensive crystal filter.

The passive components connected to the inputs and outputs of these filters are there for impedance matching. In the case of the crystal filter, this is critical because the response of a crystal filter can change dramatically if there are mismatches. (To learn more about the online tool I used to design the matching networks, refer to the web sites listed in the Resources section of this article.)

The output levels of the two filters now need to be converted to logarithmic power levels for the final step of

spectrum generation. In RF, the common unit for power is decibels relative to 1 milliwatt (dBm):

$$\text{Power dBm} = 10 \log \frac{\text{Power}}{1 \text{ mW}}$$

The Analog Devices AD8037 logarithmic amplifier is perfectly suited to perform this required conversion. Its output varies 25 mV/dBm over a 90-dB dynamic range. It responds to signals as low -75 dBm. In addition, it's highly stable and runs off a single power supply.

## CONTROLLER HARDWARE

Figure 7 shows the controller portion my spectrum analyzer. A PIC18F4520 serves as the controlling microprocessor. I used the PIC18F4520 because it has an internal clock source. Using the internal PLL and no external parts, the chip runs at 32 MHz. Crosstalk to the RF section is minimized because this is all internal to the microcontroller. I

also wanted to minimize the number of support parts around the microcontroller, so the 40-pin version provided the necessary I/O pins needed to directly control the spectrum analyzer without the use of external bus-sharing hardware. All of the I/O pins are used except one.

The spectrum analyzer does a lot of arithmetic, so the 8 × 8 hardware multiplier in the PIC18F4520 is highly attractive. The PIC18F4520 has 1,536 bytes of RAM onboard, which is needed to hold a single pass of the spectrum analyzer without any external storage. The PIC18F4520 has 32 KB of internal flash memory, relatively fast ADCs, and a wide word architecture that makes for fast execution speeds.

The parts around the PIC18F4520 are fairly standard. There is hardware for an RS-232 interface for running the spectrum analyzer in PC Connected mode. A standard 2 × 16 LCD and a standard 4 × 3 keyboard are included for

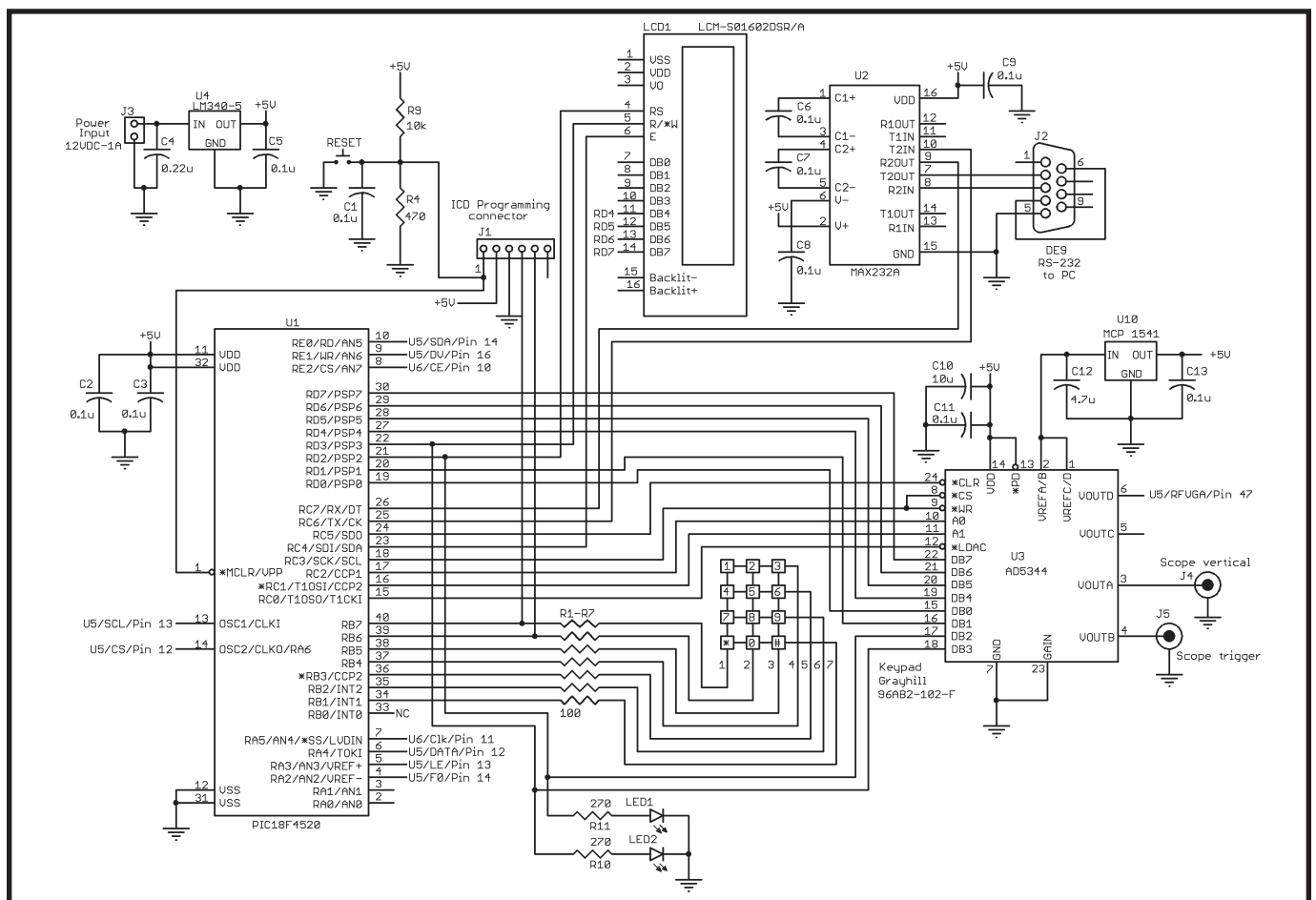


Figure 7—The PIC18F4520 microcontroller is the heart of the controller. A minimum amount of support hardware is necessary. Multiple display and control options are interfaced to the microcontroller. A single 5-V supply is required.

running the system in Stand-Alone mode. The PCB also has a place to mount a six-pin modular connector that you can use to interface with the Microchip ICD2 code development system.

Three analog signals are generated in the spectrum analyzer and controlled by the PIC18F4520. Two of these are for driving an oscilloscope when it's chosen for output display. The third signal gives the PIC18F4520 precision control of the MAX3550's RF gain. I chose a four-channel Analog Devices AD5334 DAC to accomplish these tasks because it's a fast, easy-to-interface, low-noise part. I added a Microchip MCP1541 to provide a stable 4.096-V reference voltage for the digital-to-analog process.

A final nice feature is that everything runs from a single 5-V power supply. Normally, RF circuits containing amplifiers and PLL require other than 5 V to operate. But because of the MAX3550's high integration level, and because the SA612A conversion step was designed assuming only a 5-V supply, the power supply requirements remain basic. Every RF component is also meticulously bypassed to keep the supply line clean. This allows the 5-V supply to be implemented with an LM3405 voltage regulator and a few local bypass capacitors. A heatsink is required to supply the spectrum analyzer's 375-mA load.

## SOFTWARE & FIRMWARE

My spectrum analyzer has a lot of functionality, and I will continue to evolve some of the things that it can do. The code for the PIC18F4520 is written almost entirely in PBASIC, making it easy to follow and readily modifiable. The program for operating in PC mode is written entirely in Visual Basic (again, making it easy to read and evolve). Although there is a lot of code to perform all the functions required by the PIC18F4520 and the PC, there's nothing too unusual that's worth noting. You may download the code and Visual Basic files from the *Circuit Cellar* FTP site.

There are several key operating specifications for the spectrum analyzer. There is about 1 GHz of frequency covered at two resolution bandwidths.

The spectrum analyzer is capable of seeing signals at -93 dBm with good power level and frequency accuracies.

The spectrum analyzer's frequency range is from 10 to 956 MHz. The resolution bandwidth is 330 kHz at low resolution; it's 7.5 kHz at high resolution. The system's sensitivity is -93 dBm at low and high resolution, and its maximum input level is 0 dBm.

The spectrum analyzer's power accuracy at low and high resolution is less than 1 dB, and the internal attenuation is 0 and 40 dB. The frequency accuracy at low resolution is less than 166,666 Hz; it's less than 2,500 Hz at high resolution.

The spectrum analyzer's interfaces include RS-232, an LCD, a keypad, and an oscilloscope vertical/trigger. The spectrum analyzer requires 5 V and 375 mA at low and high resolutions.

The unit is extremely sensitive to low-level signals, and it has good frequency accuracy and resolution. Also, as I explained earlier, there are several ways to display the analyzer's output and command its operations (see Photo 1b). It can be controlled as a stand-alone device from the keypad or from a virtual keypad on a PC screen using a mouse. The output spectrum can be a censored trace on an oscilloscope or a more elaborate PC screen display with a few extra whistles and bells. Alternatively, the power level at a single frequency can be displayed on the LCD.

## GREAT RF TOOL

I love working on a project that requires me to learn new things. I like it even better when I end up with a product that's useful and fun to use. My spectrum analyzer exceeded my expectations in terms of performance and usefulness, and it has become a part of my everyday set of RF prototyping tools.

If you're a little nervous about taking on a project of this complexity, you might want to consider implementing only the 330-kHz low-resolution bandwidth portion of the spectrum analyzer. This will enable you to eliminate the LMX2306 PLL, the 10.7-MHz crystal filter (and the AD8307 connected to it), and the varactor-tuned circuit attached to the SA612A. You'd

just have to replace the components connected to the SA612A oscillator pins with a fixed 35-MHz X'tal. You could also leave out the oscilloscope and keypad interface hardware. If you do, you'd use the spectrum analyzer in only its PC mode. This would still be a useful tool that would serve you well in many of your RF pursuits. Good Luck! ☑

*Neal Martini (nealmartini@cableone.net) holds an M.S.E.E. degree from the University of Missouri, Rolla. He is retired after working 24 years for Hewlett-Packard in the LaserJet and InkJet printing businesses. In addition to being involved with a variety of boards, Neal works independently in product development in several application areas. In his spare time, he enjoys spending time with family, woodworking, racquetball, golf, and playing the piano.*

## PROJECT FILES

To download the code and additional files, go to [ftp://ftp.circuitcellar.com/pub/Circuit\\_Cellar/2006/192](ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/192).

## RESOURCES

Impedance Matching, [www.hoflink.com/~mkozma/match19c.html](http://www.hoflink.com/~mkozma/match19c.html).

Loop Filter Design, <http://webench.national.com/appinfo/webench>.

## SOURCES

**AD5334 DAC and AD8307 amplifier**  
Analog Devices, Inc.  
[www.analog.com](http://www.analog.com)

**MCP1541 Voltage reference and PIC18F4520 microcontroller**  
Microchip Technology, Inc.  
[www.microchip.com](http://www.microchip.com)

**LMX2306 Synthesizer and WEBENCH electrical simulator**  
National Semiconductor Corp.  
[www.national.com](http://www.national.com)

**SA612A Mixer**  
Philips Semiconductors  
[www.semiconductors.philips.com](http://www.semiconductors.philips.com)

**ZMDC953 Diode**  
Zetex Semiconductors  
[www.zetex.com](http://www.zetex.com)