FEATURE ARTICLE

by Larry Foltzer (USA)

Sweep-Frequency Generator Design

DDS-Based Test Equipment from Sweat Equity

A proper design lab needs an accurate, programmable frequency source. And if you take a lot of repetitive frequency-domain measurements on filters, an accurate sweeper is a must-have tool. Here you learn how to use your high-speed circuit design expertise to build a digital sweep-frequency generator (SFG) and a digital reference-frequency generator (RFG).

Talk about timing! I had just completed the design and prototype implementation of two frequency-domain test instruments when I received *Circuit Cellar* 243, which featured Robert Lacoste's article, "A Tour of the Lab (Part 2): The Frequency Domain" (October 2010). Robert's article prompted me to start writing about my designs so that others might benefit from them.

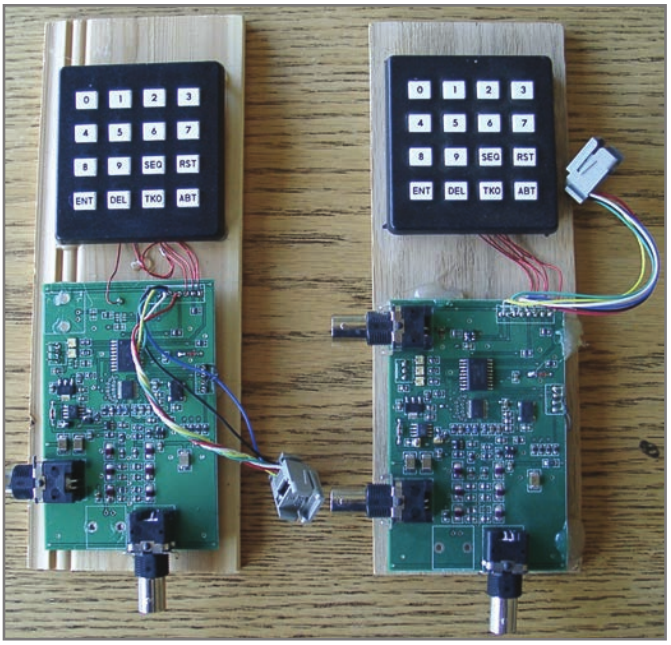

Photo 1—The completed frequency-domain test instruments. An Analog Devices AD9834-based RFG is on the left. An AD5930-based SFG is on the right. The ICSP interface used to program a Microchip Technology PIC16F627A microcontroller is provided by a dangling RJ connector socket.

As you are probably aware, Analog Devices offers a family of direct digital synthesis (DDS) devices designed to fulfill a multitude of sophisticated and important electronic functions, such as a programmable frequency reference, sweep-frequency generators, and amplitude, frequency, and phase modulators. At minimum, a well-equipped lab needs an accurate, programmable frequency source. Furthermore, if one anticipates making a lot of repetitive frequency-domain measurements on filters, an accurate sweeper is worth its weight in silver. So, armed with high-speed circuit design expertise, three pieces of silver, a Microchip Technology 8-bit microcontroller, and Assembly language programming experience, I set out to modernize and enhance the accuracy of my home laboratory with a digital sweep-frequency generator (SFG) and a digital reference-frequency generator (RFG).

From a spectral point of view, most of my home projects fall below 10 MHz. So, I selected the Analog Devices 50-MHz AD5930 waveform generator for my SFG and the 75-MHz AD9834 waveform generator for the RFG implementation (see Photo 1).

Early in the project design phase, I discovered that the signal pinout assignments of the AD9834 and the AD5930 were nearly identical, making it possible to use a single printed circuit board (PCB) design for both projects (more on that later). Therefore, I decided to start with the SFG design project since the RFG project could be viewed as a subset of the SFG from both a physical and code point of view.

The objectives of the AD5930-based SFG design were a 2- to 10-MHz frequency range, a 2-Hz frequency resolution,

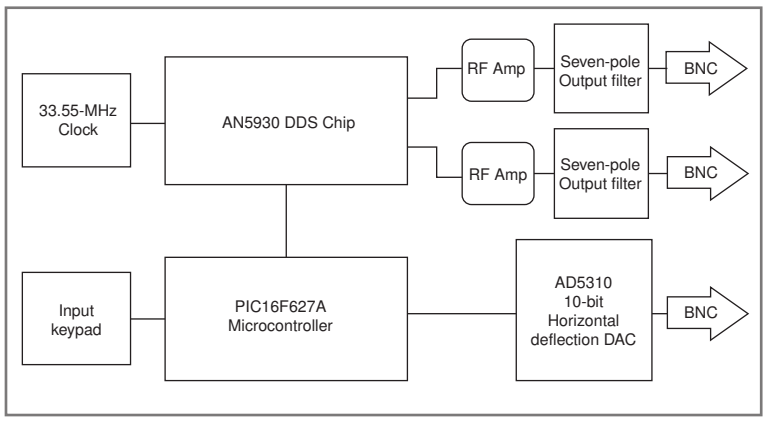

Figure 1—The functional block diagram of the sweep-frequency generator (SFG)

and an output signal level greater or equal to +7 dBm into 50 Ω. I also wanted the generator to provide a synchronized horizontal sweep signal in support of amplitude versus frequency display. The user interface had to remain simple to keep size, cost, and code complexity to a minimum.

Figure 1 shows a functional block diagram of the SFG. A description of the key features and functions follows.

KEYPAD INPUT & COMMUNICATIONS

The keypad shown in Photo 2 is a common fixture in my workshop, providing custom I/O signal functionality for numerous projects. For simplicity, the SFG uses a single LED as a handshake mechanism to signal when the system is ready for data input. When the SFG is running steady state, the handshake LED is illuminated and waits for the first key press of an input sequence. If a sweep is in progress, the LED remains lit until the current sweep is complete and the first key input is recognized. At this point, the LED is extinguished indicating that the SFG is ready for the rest of the command sequence. The LED turns on again after the appropriate parameter command key is pressed.

Data/parameter entry uses reverse-Polish notation (RPN) format. Parameter values are input in base_10 format first, followed by a character keystroke signifying what parameter is to be assigned the preceding input value. Only two input parameters are "required" to control the operation of the SFG: sweep center frequency and frequency step size. An optional third parameter may be used to increase the length of time each frequency making up the sweep (500 total) is present at the output. This duration is inversely proportional to the center frequency (F_C) and scaled/increased appropriately by the value entered here. A typical SFG programming sequence is shown in Table 1.

The white space between the first and second parameter key entry sequence shown in Table 1 is intentional. It indicates that some sweep speed and position-dependent delay may be required after the first character of the command sequence is entered in order to comply with the I/O handshake protocol. This behavior is due to the fact that the microcontroller uses inline code to control and synchronize the output frequency dwell time and horizontal output deflection signals. New input is only recognized after full sweep completion.

To set the center frequency to 4.5 MHz, enter:	4 500000 (ENT)
To set the frequency step to 2,000 Hz, enter:	2 000 (DEL)
To set the frequency step duration to 1.0 ms, enter: (This represents 200 µs overhead + 16 × 50 µs)	16 00 (TKO)

Table 1—A typical SFG programming sequence

The keypad scanner is based on a Microchip Technology PIC16F84 microcontroller. It is tasked with scanning the keypad for input and communicating with the SFG's microcontroller over a unidirectional serial interface consisting of the following signal and handshake lines: Data Available (/DAV), Serial Data, and Acknowledge (/ACK).

When a key is pressed, the PIC16F84 places the first bit of the keycode on the data line and pulls /DAV low. If the SFG's microcontroller detects /DAV low (/DAV sampled at beginning of retrace time), it reads the serial data line and pulls /ACK low, telling the keypad processor the data bit has been captured. The keypad then raises /DAV and the process is repeated until all the keycode bits have been communicated.

DDS CLOCK CONSIDERATIONS

The AD5930 is built around a 24-bit core. According to the datasheet, it can be clocked at rates up to 50 MHz. The 24-bit core size means the DDS machinery of the AD5930 can break a single sine wave into (2^{24}) discrete "phase-slices," where each phase-slice's amplitude is quantized into a 10-bit word stored in a look-up table. Sinusoidal frequency synthesis takes place by reading successive values from the phase-to-amplitude look-up table and feeding them to a high-speed DAC. The step size (SS) between successive values read from the look-up table is proportional to the desired output frequency. The following equation shows how to calculate SS. The magnitude of the phase step needed to produce F_{OUT}:

$$SS = \left(2^N\right) \times \left(\frac{F_{OUT}}{F_{DDS}}\right) \text{ for } N = 24 \quad [1]$$

Rearranging this equation forms a ratio of constants on the left side of the equation, and a constant ratio of variables on the right side of the equation:

$$\frac{F_{DDS}}{\left(2^N\right)} = \left(\frac{F_{OUT}}{SS}\right) \quad [2]$$

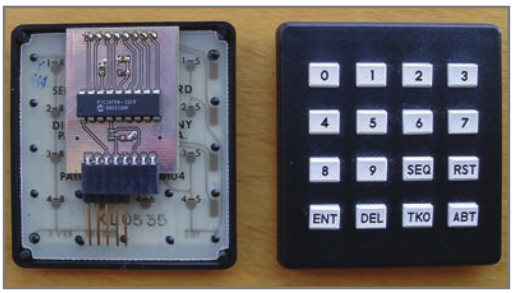

Photo 2—The keypad which features a Microchip Technology PIC16F84 microcontroller

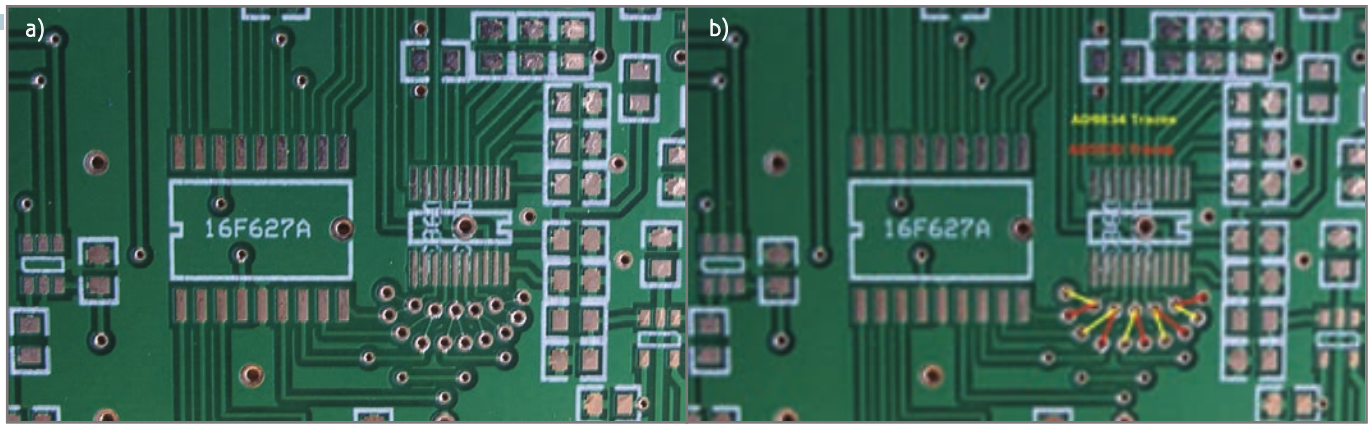

Photo 3—Magnified views of the printed circuit board (PCB). **a**—A virgin PCB with both sets of configuration traces intact. **b**—The two sets of traces used to configure the PCB are color coded to show the different traces.

For a ratio of two, $F_{DDS} = 2 \times 2^{24} = 2^{25} = 33.554432$ MHz, which is both greater than twice the Nyquist frequency and below the maximum specified operating frequency of 50 MHz. Furthermore, SS, which is the frequency programming word, must be half the desired output frequency for the ratio of two to be preserved:

$$SS = F_{OUT} \times \left[\frac{(2^{24})}{(2^{25})}\right] = \frac{F_{OUT}}{2} \quad [3]$$

It is important to note that by employing this strategy, all you have to do to calculate the appropriate phase SS to generate a specific output frequency is divide that frequency in half. This task couldn't be easier in a binary machine. With a cleared carry bit, simply shift the frequency number to the right one bit and you're done. This makes an Assembly language programmer very happy, indeed.

Furthermore, from the equation above, you see that for a unity change in SS, F_{OUT} increases by 2, which is the frequency resolution of the system. The question is, from where do you get a 33.554432-MHz clock to drive the AD5930? The answer is that obtaining a clock source at virtually any specific frequency is easy these days, given the availability of programmable frequency references, such as the Epson SG-8002JF-PHB-ND oscillator available for just a few dollars from distributors such as Digi-Key.

DDS-TO-MCU PHYSICAL CONTROL INTERFACE

During schematic capture, I realized that Analog Devices had made an error in pinning out the programming and control signal lines on the AD5930 and AD9834. That is, the S_{DATA}, S_{CLK}, and F_{SYNC} are on different pins on these parts. Since I wanted to use a single PCB design for my two projects, I would have to provide a way to remap the necessary signals to correct this unfortunate pinout error.

Photo 3 shows magnified views of the area of the PCB where my solution is implemented. Photo 3a shows a virgin PCB with both sets of configuration traces intact. Photo 3b color codes the two sets of traces used to configure the PCB for the appropriate application. Cut the yellow set of traces for AD5930 compatibility or cut the red set of traces for AD9834 compatibility. For best results, use a number 11 surgical blade to sever the appropriate unused trace connections. If you make a mistake, through-holes are provided to facilitate a repair.

I selected a Microchip Technology PIC16F627A microcontroller to orchestrate the operation of these designs. There wasn't any particular feature that was required of these microcontrollers other than sufficient port count, low cost, and availability. And, since I already had them in my inventory, the choice was clear. You can easily port the code to other Microchip Technology parts if you are so inclined. Furthermore, you can alter the code as you see fit

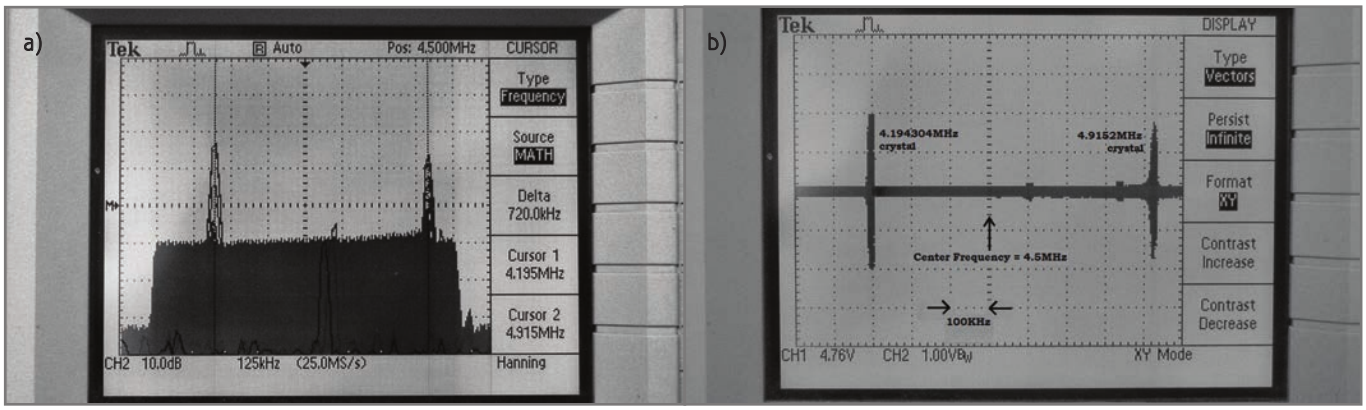

Photo 4—Sweeps configured with a width of 1.0 MHz spanning 5 to 4 MHz. The signal transmission characteristics of two parallel connected crystals connected in series with a 50-Ω terminated scope input are also shown.

and reprogram the microcontroller through the in-circuit serial programming (ICSP) interface, the dangling six-position RJ connector shown in Photo 1.

SFG DISPLAY OPTIONS

I happen to be lucky enough to own a Tektronix TDS 220, digital real-time oscilloscope that provides all the support features needed as a display for the SFG: fast Fourier transform (FFT) analysis, variable persistence, and X-Y display capability.

In FFT mode, only a single input to the scope is required to generate an X-Y display of amplitude versus frequency. The FFT capability takes care of filling the appropriate frequency bins and displaying their contents logarithmically and sequentially along the X/frequency axis. However, while this mode offers some convenience, it suffers from poor resolution, especially in light of the spectral purity typical of a DDS source. More on this later.

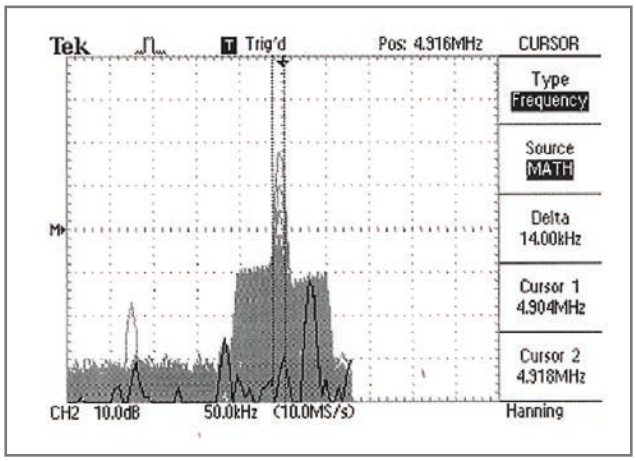

Photo 5—The crystal bandpass measuring roughly 14 kHz (about three bins wide)

In what I call the "direct approach," the TDS 220 is placed in X-Y Display mode while feeding the output of the horizontal deflection DAC to the X input channel of the scope. The sweep is always composed of 500 individual frequency steps starting at the highest frequency (F_C + (250 × dF)) and ending at (F_C – (250 × dF)). On an oscilloscope screen with 10 graduations, frequency display density is 50 steps per graduation. For the "4500000 (ENT), 2000 (DEL)" case, total sweep width is 1.0 MHz (i.e., 500 × 2,000 Hz), spanning the range 5 to 4 MHz. Photo 4 shows sweeps configured in this way, showing the signal transmission characteristics of two parallel connected crystals (4.192304 MHz and 4.9152 MHz) connected in series with a 50-Ω terminated scope input.

The difference in frequency resolution capability of the two display options is substantial. TDS 220 FFT mode resolution tracks the sampling rate (F_S) necessary to accommodate the frequencies of interest, per Nyquist. For example, to observe the transmission through a 4.9152-MHz crystal, the greatest frequency resolution is obtained when the sampling rate is set

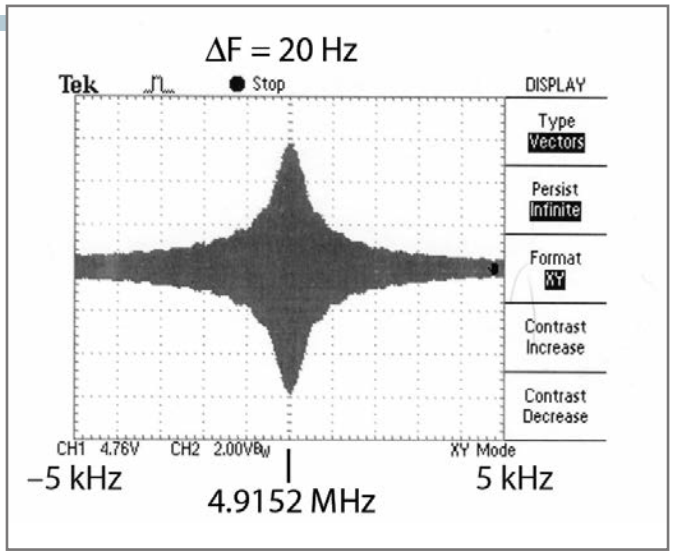

Photo 6—The transmission bandwidth measurement of the 4.9152-MHz crystal using a 20-Hz frequency step

to 10 Msps. And since the TDS 220 FFT record length is fixed at 2,048, the scope's best resolution for this signal is approximately 4.9 kHz (i.e., 10 MSps/2,048). Under these limitations, crystal bandpass measures about three bins wide or about 14 kHz (see Photo 5).

Frequency resolution using the "direct approach," is in stark contrast to the above. The direct approach, in effect, passes the signal through 1-Hz bandpass filters spaced SS Hz (effective bin width) apart. The resulting signal displays the amplitude of the RF envelope over the designated frequency span. Photo 6 shows the transmission bandwidth measurement of the 4.9152-MHz crystal using a 20-Hz frequency step (1,000 Hz per cm). Bandwidth measures roughly 30 times narrower (about 400 Hz) than obtained with the scope's built-in FFT mechanism.

The direct approach requires two inputs. The horizontal sweep deflection generator produces a linear voltage output that is decremented in sync with DDS output frequency steps. This horizontal sweep signal is generated from an Analog Devices AD5310 10-bit digital-to-analog converter (DAC). At the start of the sweep, the DAC input starts with a count of 1,012 and is decremented, two at a time, until the DAC input count reaches 12 after 500 frequency steps. Mathematically, DDS output frequency steps from $F_C + (250 \times dF)$ to $F_C - (250 \times df)$ while the sweep voltage steps from:

$$V_S = 5\ V \times \left(\frac{1{,}012}{1{,}023}\right) \quad [4]$$

to:

$$V_S = 5\ V \times \left(\frac{12}{1{,}023}\right) \quad [5]$$

Synchronization of the horizontal deflection signal DAC with the output frequency is provided by the connection of the DDS sync signal from pin 9 to microcontroller pin 11, via I/O_2 and I/O_3 shown in Figure 2.

Finally, variable persistence provides memory of the spectrum amplitude profile. This is especially useful for low-frequency measurements when the period of a single cycle could be measured in the tens of milliseconds, where the total trace

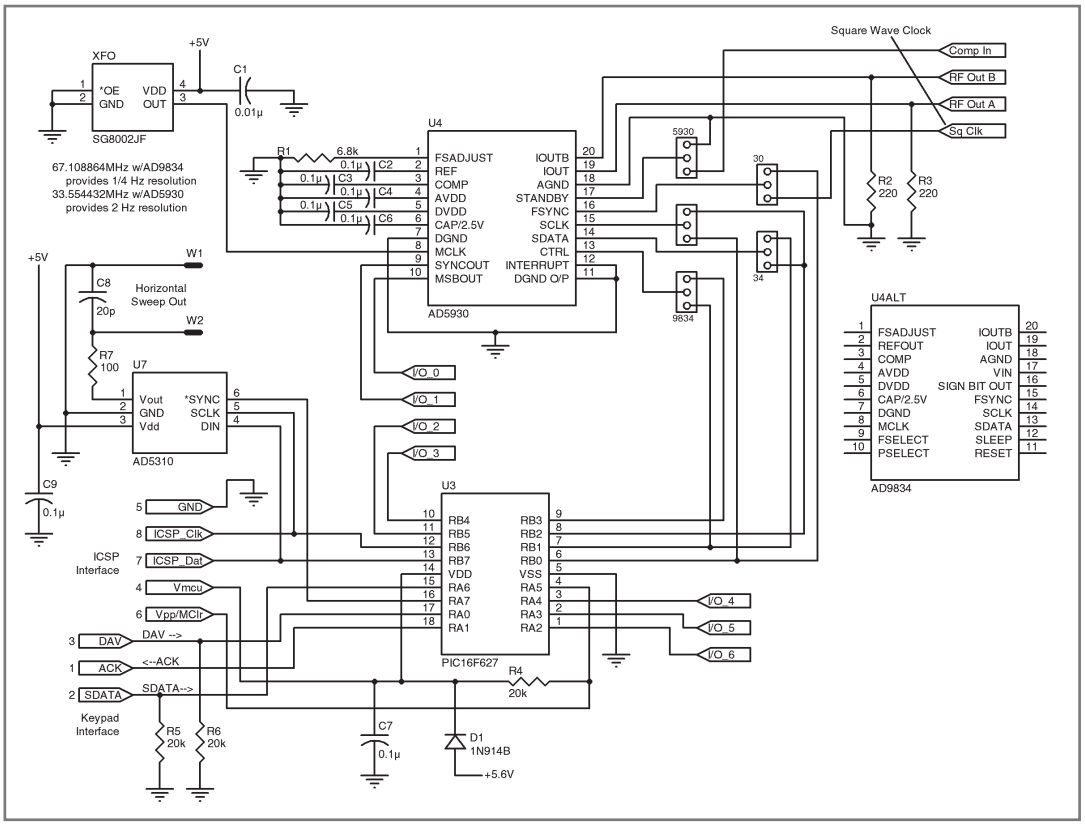

Figure 2—The synchronization of the horizontal deflection signal digital-to-analog converter (DAC) with the output frequency provided by the connection of the DDS sync signal from pin 9 to the microcontroller pin 11 via I/O_2 and I/O_3

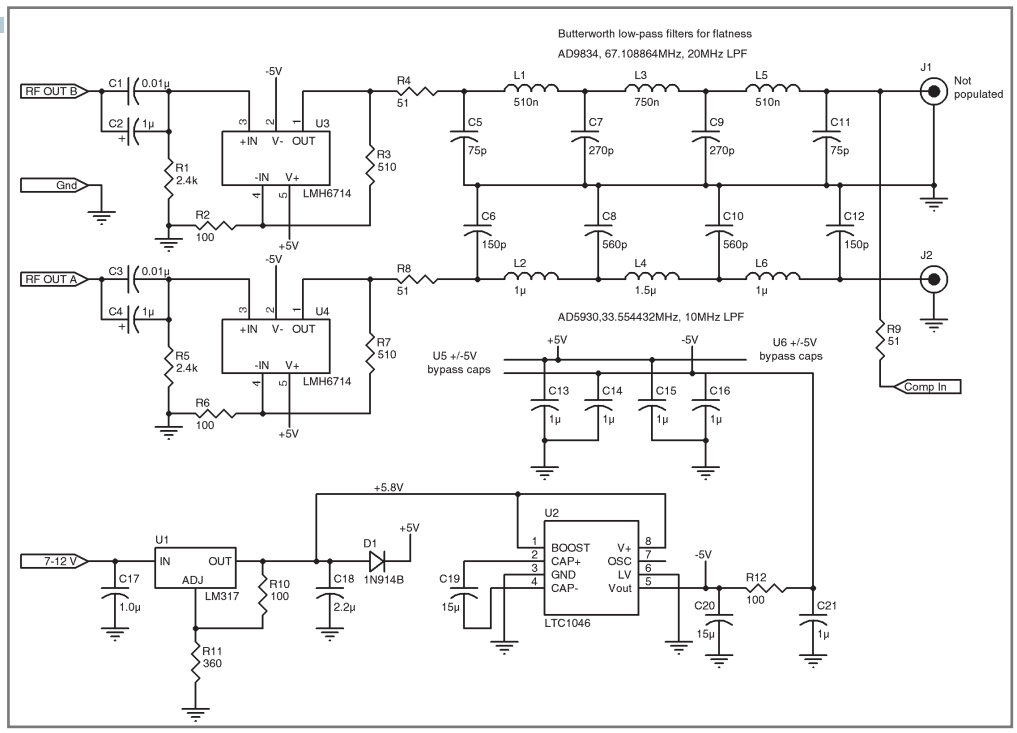

Figure 3—The operating voltages for the printed circuit board (PCB) circuits that originate from a single 7- to 9-V input via a National Semiconductor on-board LM317 regulator and a Linear Technology LT1046 converter

time for low-frequency sweeps could exceed 1 minute.

POWER CONDITIONING

Operating voltages for the PCB circuits are derived from a single 7- to 9-V input by way of a National Semiconductor on-board LM317 regulator and a Linear Technology LT1046 converter (see Figure 3). Complementary positive and negative supply voltages from the LT1046 inverter supply power to a pair of National Semiconductors LMH6714 video op-amps that subsequently drive seven-pole, low-pass output filters.

In my prototypes, I did not have the component values I had calculated for the filters shown in Figure 3, so I used the closest values I had on hand. Simulations indicated I would only pay about a 6-dB stop-band penalty for my substitutions, so I went with it. For my purposes, so far, so good!

THE INTERFACE

The SFG potentially provides two 50-Ω characteristic impedance RF outputs. In my prototype, I terminated only one RF signal in a BNC connector, but terminated the second output port with a 50-Ω resistor at the filter's output. The two RF ports are side-by-side along the bottom edge of the PCB. These outputs are capable of supplying 5 MW + 7 dBm, 1.45 V peak-to-peak into a 50-Ω load.

The BNC connector on the lower left side of the PCB, which can be turned on by software, is driven by the most-significant bit of the phase-amplitude look-up table address, producing a square wave output that is in phase with the sine wave outputs. I prefer harmonic free output, so I have it turned off.

Finally, the BNC on the upper left edge of the PCB is the output of a 10-bit DAC that generates a linear horizontal deflection signal to drive the "X" coordinate channel of an oscilloscope in sync with frequency stepping. This is a low-speed signal and intended to drive into a high input impedance. The TDS 220 enables a fine adjustment of the channel sensitivity, so that the sweep width can be adjusted to the full-scale deflection of the scope for the best display accuracy.

SFG FIRMWARE

The firmware for the SFG is available on the *Circuit Cellar* FTP site. I'm not going to elaborate on it here, except to say that I enjoy algorithm development and implementation in machine language. Even simple tasks that work well give me pleasure. Keep it simple and effective is my motto.

RFG OVERVIEW

As I stated previously, the RFG is a subset of the SFG in terms of support circuitry, firmware, and design equations. However, the faster 75-MHz AD9834 employs a 28-bit core, providing higher output frequency and frequency resolution while requiring different code to compute the frequency control word.

With the RFG based on the 75-MHz AD9834, operation to 20 MHz is possible but will require the redesign of the low-pass output filter. I did not have the appropriate parts for the 20-MHz low-pass filter design, so I used the same filter components as used on the SFG, limiting the output to 10 MHz.

The RFG and AD9834 can produce continuous wave (CW), frequency-shift keyed (FSK), and phase-shift keyed (PSK) signals with modulation externally or

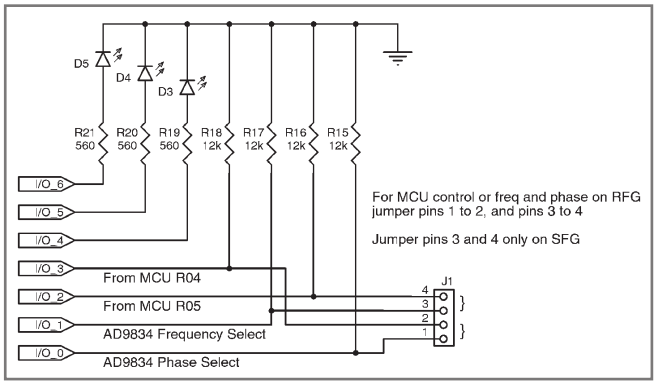

Figure 4—Jumper J1 configuration for RFG or SFG operation

by the RB4 and RB5 I/O ports on the PIC16F627A, if you write the code. Figure 4 shows how to configure J1 for modulation applications.

The output power objective is the same as for the SFG: +7-dBm output power into 50-Ω. There is no need to populate the horizontal sweep DAC, unless you are willing to write code for it for a special application (e.g., to plot the gain at a specific frequency of an AGC loop versus AGC voltage).

CLOCKING THE RFG

For convenience, you repeat Equation 2 here for N = 28, where 2^{28} is the depth of the phase-to-amplitude look-up table. Amazing!

$$\frac{F_{DDS}}{(2^{28})} = \frac{F_{DDS}}{268,435,456} = \left(\frac{F_{OUT}}{SS}\right) \quad [6]$$

Anyway, since the maximum operating frequency of the AD9834 is 75 MHz, the resolution of the RFG, F_{OUT}/SS, will be less than one.

To maximize the output frequency range while remaining true to integer math, select 67.108864 MHz (i.e., $F_{DDS} = 2^{26}$), which sets the resolution to 0.25 Hz—that is, an incremental change of SS by four is required to shift F_{OUT} by 1 Hz. Expressed mathematically, the frequency control word is:

$$SS = F_{OUT} \times \left[\frac{(2^{N})}{F_{DDS}}\right] = F_{OUT} \times \left[\frac{(2^{28})}{2^{26}}\right] = 4 \times F_{OUT} \quad [7]$$

Now all you have to do to is the following. Calculate SS by changing the base-10 input string from the keypad to hexadecimal. Perform two PIC "RLF SS,F" instructions to multiply the desired frequency by four. Perform an "ANDLW 0xFC" instruction on the LSB to mask off bits B1 and B0, ensuring the result is ((N) – (N modulo 4)). And then program the result into F0 or F1, as appropriate.

QED!

TO MODULATE OR NOT

While the RFG and the AD9834 are capable of generating angle-modulated signals, my interest, and the current implementation, is in producing CW output for potential use as a local oscillator for radio applications. The firmware listed here configures the RFG for CW output, with static levels from microcontroller ports RB4 and RBS, driving the P_{SELECT} and F_{SELECT} lines of the AD9834. This interconnection of the microcontroller and DDS is made by connecting pin 1 to pin 2 of J1, and pin 3 to pin 4 of J1, as shown in Figure 4.

The RFG firmware contains no code to introduce phase or frequency modulation using the microcontroller. However, the AD9834 is initialized to accept externally generated TTL-level modulation input by way of I/O jumper pins I/O_0 for phase register selection and I/O_2 for frequency register selection. If you write your own modulation generator code, you can have the microcontroller drive DDS register select pin 10 (P_{SELECT}) and pin 9 (F_{SELECT}) directly and achieve modulation rates as high as 234 Kbps.

(SEG)	"Shift Key" for functional expansion
(RST)	Full RFG reset
(ENT)	Switch output to frequency specified in (F0) register
(DEL)	Switch output to frequency specified in (F1) register
(TKO)	Change output signal phase to that specified in (P0) register
(ABT)	Change output signal phase to that specified in (P1) register
(PARAM)(ENT)	Store PARAM in F0, F_{OUT} = F0
(PARAM)(DEL)	Store PARAM in F1, F_{OUT} = F1
(PARAM)(TKO)	Store PARAM in P0, Phase = P0
(PARAM)(ABT)	Store PARAM in P1, Phase = P1
(PARAM)(SEQ)(ENT)	Save PARAM in EEPROM address F0
(PARAM)(SEQ)(DEL)	Save PARAM in EEPROM address F1
(PARAM)(SEQ)(TKO)	Save PARAM in EEPROM address P0
(PARAM)(SEQ)(ABT)	Save PARAM in EEPROM address P1
Example:	Programming F0 and F1, in that order, and switching back to F0 for F0 = 4.096 MHz, F1 = 4.9152 MHz Command Sequence: 4096000 (ENT) 4915200 (DEL) (ENT)

Table 2—Alphanumeric keystroke sequences used to control the operation of the RFG

The AD9834 can be modulated by way of bit 10 (PSEL) or bit 11 (FSEL) in the DDS control register if bit 9 is cleared in the control register. However, that modulation path is slow since you have to send a 16-bit control word just to change one bit, for a maximum speed of roughly 17 Kbps.

RFG COMMANDS & CODE

Table 2 is a list of the alphanumeric keystroke sequences used to control the operation of the RFG. (PARAM) signifies a purely numeric entry representing frequency step and phase offset parameters. And, as with the SFG, RFG input command sequence is RPN-like.

The firmware/code for this project is simple and mostly static in nature. The code is available on the *Circuit Cellar* FTP site.

THAT'S A WRAP, ALMOST

The PCB is a two-sided board and was designed using ExpressPCB CAD tools. For the most part, this design worked well, although I wish I had provided large pads for some of the higher-value capacitors since 0805 versions of them are expensive and not in my inventory. I had to tombstone larger caps on small pads to avoid bankruptcy.

My original design targeted the AD5310 DAC for the horizontal displacement signal generator. But, it turned out to be in short supply, and a direct pin-compatible version was not found. I found a solution, however, in the AD5611, whose pinout matched the PCB when mounted upside down.

Finally, I have a pet peeve: What's with all these super-small packages? Isn't the PIC16F627A SOIC-18 small enough? Is the real estate savings of the DDS packages worth the handling difficulty? I don't think so. Look at the microcontroller size versus that of the DDS. Now try soldering these devices by hand. Yes, it can be done. Then again, I've been doing this stuff for a long time. But, as my eyes get older, it becomes more difficult to play in the sandbox, with or without a microscope in the workshop. No wonder kids don't go into this business of ours; they are intimidated straight out by the smallness. I wish manufactures would offer hobby packaging so the new generation would have some way to get their feet wet.

Bring back the Heathkit!

Larry Foltzer (lefoltzer@dishmail.net) was employed for 30 years in the fiber optics communication industry. First, he cofounded Optelecom, Inc. A decade later, his focus shifted to data and telecommunications, where for Motorola he participated in the IEEE-802 MAC subcommittee on Token-Passing access control methods. Later, he was the System Architect of the Raychem/Raynet Passive Optical Network System.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2011/254.

RESOURCE

R. Lacoste, "A Tour of the Lab (Part 2): The Frequency Domain," *Circuit Cellar* 243, 2010.

SOURCES

AD5930 and AD9834 Waveform generators and AD5310 DAC
Analog Devices, Inc. | www.analog.com

Epson SG-8002JF-PHB-ND oscillator
Digi-Key Corp. (distributor) | www.digikey.com

LT1046 Converter
Linear Technology Corp. | www.linear.com

PIC16F84 and PIC16F627A Microcontrollers
Microchip Technology, Inc. | www.microchip.com

LM317 Regulator and LMH6714 Video op-amps
National Semiconductor Corp. | www.national.com

TDS 220 Digital real-time oscilloscope
Tektronix, Inc. | www.tek.com

NEED-TO-KNOW INFO

Knowledge is power. In the computer applications industry, informed engineers and programmers don't just survive, they *thrive* and *excel*. For more need-to-know information about some of the topics covered in this article, the *Circuit Cellar* editorial staff recommends the following content:

—

Programmable Network Analyzer
by James Gaston & Peter Hiscocks
***Circuit Cellar* 194, 2006**

James and Peter use an oscilloscope and a waveform generator to create a programmable network analyzer. The simple program will be a great addition to your workbench. Topics: Network Analyzer, Oscilloscope, Waveform Generator, DSO-101, Transfer Function, WGM-101, Tcl/Tk

—

Go to *Circuit Cellar*'s webshop to find this article and more: www.cc-webshop.com