QEX

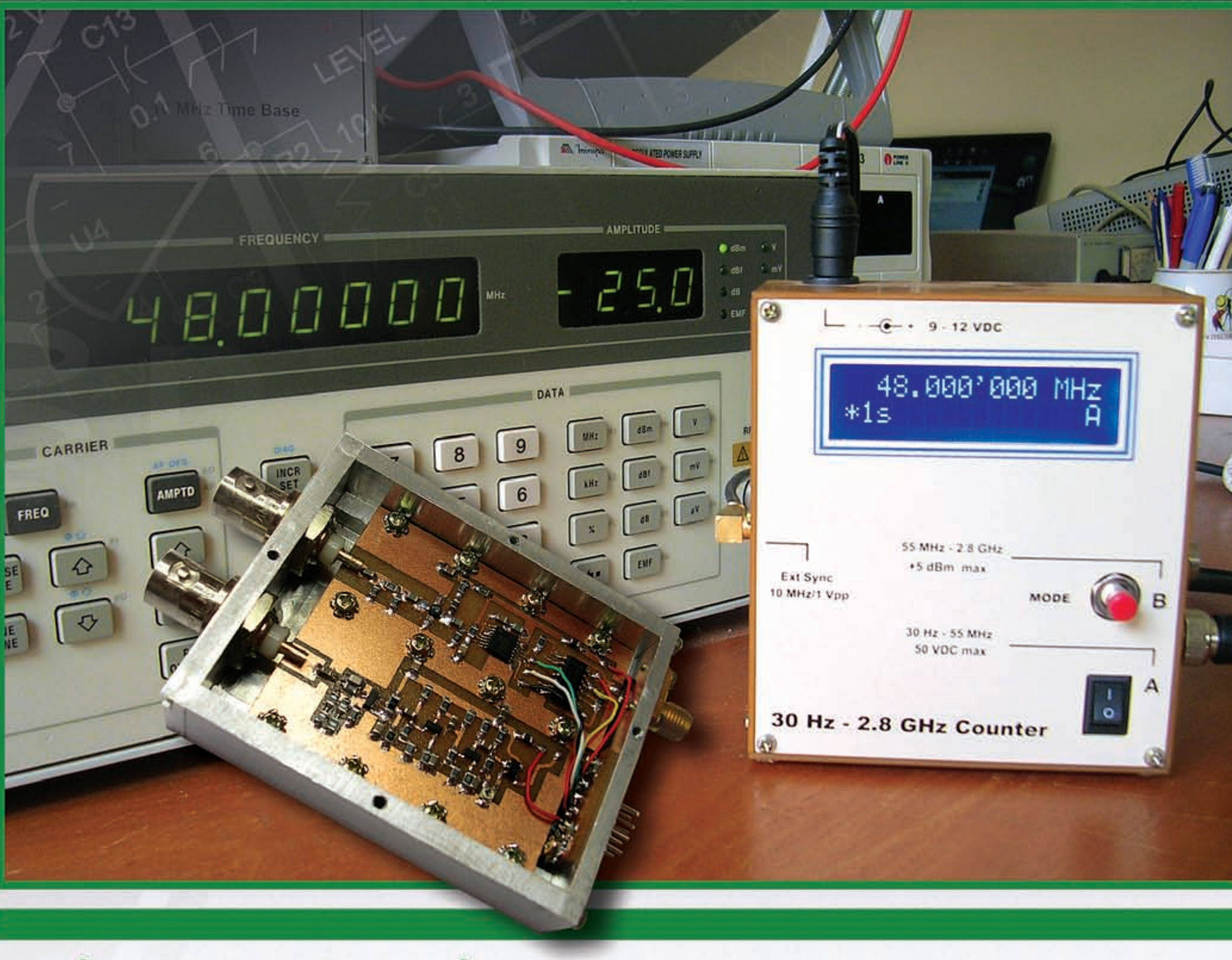
$5

May/June 2010

www.arrl.org

A Forum for Communications Experimenters

Issue No. 260



Rubens Ramos Fernandes uses a PIC16F876A microcontroller and the prescaler in a National Semiconductor LMX2326 RF synthesizer to produce a handy frequency counter that covers from 30 Hz to 2.8 GHz.

ARRL The national association for AMATEUR RADIO™
225 Main Street
Newington, CT USA 06111-1494

KENWOOD

Listen to the Future

EchoLink® Ready

TM-V71A

Large dual color amber or green thirteen segment LCD.

With the Kenwood TM-V71A you have a choice of where you want your speaker, on the top or on the bottom of the radio. Simply remove the faceplate and flip the main body, then reattach the face, it's that simple! Yet another Kenwood **1st**, this dual band transceiver has ten dedicated EchoLink® memory channels as well as EchoLink sysop-mode operation. EchoLink connection to your PC via the optional PG-5H cable kit is easy with no expensive interface needed.

EchoLink® is a registered trademark of Synergenics, LLC. For more information please see: www.echolink.org.

KENWOOD U.S.A. CORPORATION
Communications Sector Headquarters
3970 Johns Creek Court, Suite 100, Suwanee, GA 30024
Customer Support/Distribution
P.O. Box 22745, 2201 East Dominguez St., Long Beach, CA 90801-5745
Customer Support: (310) 639-4200 Fax: (310) 537-8235

www.kenwoodusa.com
ADS#08610

JQA-1205 091-A
ISO9001 Registered
Communications Equipment Division
Kenwood Corporation
ISO9001 certification

QEX (ISSN: 0886-8093) is published bimonthly in January, March, May, July, September, and November by the American Radio Relay League, 225 Main Street, Newington, CT 06111-1494. Periodicals postage paid at Hartford, CT and at additional mailing offices.

POSTMASTER: Send address changes to: QEX, 225 Main St, Newington, CT 06111-1494
Issue No 260

Harold Kramer, WJ1B
Publisher

Larry Wolfgang, WR1B
Editor

Lori Weinberg, KB1EIB
Assistant Editor

Zack Lau, W1VT
Ray Mack, W5IFS
Contributing Editors

Production Department

Steve Ford, WB8IMY
Publications Manager

Michelle Bloom, WB1ENT
Production Supervisor

Sue Fagan, KB1OKW
Graphic Design Supervisor

David Pingree, N1NAS
Senior Technical Illustrator

Advertising Information Contact:

Janet L. Rocco, W1JLR
Business Services
860-594-0203 – Direct
800-243-7768 – ARRL
860-594-4285 – Fax

Circulation Department

Cathy Stepina, *QEX Circulation*

Offices

225 Main St, Newington, CT 06111-1494 USA
Telephone: 860-594-0200
Fax: 860-594-0259 (24 hour direct line)
e-mail: **qex@arrl.org**

Subscription rate for 6 issues:

In the US: ARRL Member $24, nonmember $36;

US by First Class Mail:
ARRL member $37, nonmember $49;

International and Canada by Airmail: ARRL member $31, nonmember $43;

Members are asked to include their membership control number or a label from their QST when applying.

In order to ensure prompt delivery, we ask that you periodically check the address information on your mailing label. If you find any inaccuracies, please contact the Circulation Department immediately. Thank you for your assistance.

May/June 2010

About the Cover

Rubens Ramos Fernandes (formerly PY2QE), designed and built a handy frequency counter for experimenters. He used a PIC16F876A microcontroller and the prescaler in a National Semiconductor LMX2326 RF synthesizer to build his counter. It covers 30 Hz to 55 MHz and 55 MHz to 2.8 GHz.

In This Issue

Features

Columns

Index of Advertisers

The American Radio Relay League

The American Radio Relay League, Inc, is a noncommercial association of radio amateurs, organized for the promotion of interest in Amateur Radio communication and experimenta-tion, for the establishment of networks to provide communications in the event of disasters or other emergencies, for the advancement of the radio art and of the public welfare, for the representation of the radio amateur in legislative matters, and for the maintenance of fraternalism and a high standard of conduct.

ARRL is an incorporated association without capital stock chartered under the laws of the state of Connecticut, and is an exempt organization under Section 501(c)(3) of the Internal Revenue Code of 1986. Its affairs are governed by a Board of Directors, whose voting members are elected every three years by the general membership. The officers are elected or appointed by the Directors. The League is noncommercial, and no one who could gain financially from the shaping of its affairs is eligible for membership on its Board.

"Of, by, and for the radio amateur," ARRL numbers within its ranks the vast majority of active amateurs in the nation and has a proud history of achievement as the standard-bearer in amateur affairs.

A *bona fide* interest in Amateur Radio is the only essential qualification of membership; an Amateur Radio license is not a prerequisite, although full voting membership is granted only to licensed amateurs in the US.

Membership inquiries and general corres-pondence should be addressed to the administrative headquarters:

ARRL
225 Main Street
Newington, CT 06111 USA
Telephone: 860-594-0200
FAX: 860-594-0259 (24-hour direct line)

Officers

President: KAY C. CRAIGIE, N3KN
570 Brush Mountain Rd, Blacksburg, VA 24060

Chief Executive Officer: DAVID SUMNER, K1ZZ

The purpose of *QEX* is to:

1) provide a medium for the exchange of ideas and information among Amateur Radio experimenters,

2) document advanced technical work in the Amateur Radio field, and

3) support efforts to advance the state of the Amateur Radio art.

All correspondence concerning *QEX* should be addressed to the American Radio Relay League, 225 Main Street, Newington, CT 06111 USA. Envelopes containing manuscripts and letters for publication in *QEX* should be marked Editor, *QEX*.

Both theoretical and practical technical articles are welcomed. Manuscripts should be submitted in word-processor format, if possible. We can redraw any figures as long as their content is clear. Photos should be glossy, color or black-and-white prints of at least the size they are to appear in *QEX* or high-resolution digital images (300 dots per inch or higher at the printed size). Further information for authors can be found on the Web at **www.arrl.org/qex/** or by e-mail to **qex@arrl.org**.

Larry Wolfgang, WR1B

lwolfgang@arrl.org

Empirical Outlook

A Request for Help

I am using my editorial space in this issue of *QEX* to ask you, my readers, for some help. I am always pleased to hear from authors, whether new or seasoned writers, who would like to submit an article for publication. Obviously, that is great. I know many more of you have ideas for articles, or have projects that you may not have even thought about writing up for publication. I encourage you to take that step of sharing your knowledge (or questions/frustrations) with your fellow *QEX* readers.

I need your help in a number of ways when you submit your work for possible publication. We prefer to receive manuscripts in electronic format, normally submitted by e-mail. You can reach me at a number of ARRL e-mail addresses; perhaps the easiest to remember is **qex@arrl.org**. Of course you can also send manuscripts to ARRL Headquarters through the mail, and if you submit your article that way, please include in the package a CD with the article files and any related materials.

We use Microsoft *Word* to prepare articles for publication, but you can submit files using other word processing programs. Many of you are using Open Office. I've learned that Open Office has a very effective file converter, so it is easy to open those files and save them as "article.docx" or "article.doc" files. Could you do me the favor of converting your article and sending me the *Word* file? If you aren't confident that the file will translate correctly, send both the ".odt" and ".doc" or ".docx" files. That way I can look at the original Open Office file if something doesn't look right in *Word*.

That's just one example. When you prepare graphics for your article, you might be using a schematic capture program or a variety of CAD or drawing programs. While our graphics folks can deal with most formats, we find that most of those drawing programs are able to produce "image.jpg," "drawing.bmp" or "chart.gif" files. Adobe PDF files are also easy to use. A few months ago it seemed like everyone was sending Microsoft *Visio* graphics files. We don't use *Visio*, and don't have a current version on our computers. I learned that *Visio* has a nice file conversion utility. Please use those conversion utilities to save your graphics files in one of the preferred file formats.

If your article includes photos (and what article doesn't look better with a few nice photos?) please keep in mind that we need at least 300 dots per inch of resolution at the printed size, for publication. Most modern digital cameras will easily produce images with suitable resolution. In general, use the highest resolution setting available on your camera, and use the jpg format. When you are setting up your photos, take a look at the background and consider how that will look when it is printed in *QEX*. If your workbench or operating table is as cluttered as mine, you may want to clean off a space or shoot the photo somewhere else. A bed sheet draped over a card table or other surface can provide a nice clean background, and perhaps give you better contrast to show off your project. Setting the project on your bathroom (or kitchen) floor tiles may not be the best idea, either. The wild pattern in your carpet may also be distracting!

Proper lighting is a challenge to capturing good photos. An on-camera flash often creates excess glare or "hot spots" reflected in a display or glossy chassis. You can use a couple of flood lights to light everything evenly from both sides, although incandescent bulbs can give everything a red tint. Also be careful of how shadows will affect your picture. Natural sunlight often provides the best color, and although we don't print *QEX* in color, you might end up with a possible cover shot. Direct sunlight can also cast harsh shadows, so you should be careful of that, too. You don't need to have a professional photographer take your pictures, and we don't expect you to provide perfect photos, but with a little care, and attention paid to potential problems, you can produce excellent shots.

Many *QEX* articles involve showing oscilloscope or spectrum analyzer screens. Most modern oscilloscopes and spectrum analyzers provide some way to store the screen image to a digital file. Still, for many of us, the only way to capture those images is to take a photo of the display screen. This leads to a number of problems, including finding a way to align the image properly in the viewfinder, and getting a nice square image. Too often these photos come out blurred on one side, or with excessive "pin cushion effect" in which the sides and/or top and bottom bulge outward.

In this issue, Sam Green, WØPCE, describes the oscilloscope camera mount that he built. With this simple device made from scrap wood, Sam found a way to position his camera at the proper distance from the screen to fill the image area with the display while keeping the camera lens centered on the screen and parallel to it. Sam has produced some excellent oscilloscope photos. Read his article and think about how you could make something similar to ensure the proper alignment of your camera for scope photos.

Many authors want to submit their work in an article layout format, complete with drawings and photos embedded into the file. While this makes a nice presentation, it is not what we need for the typesetting process. Even if you want to place your graphics into your word processing file, please send us separate graphics files in one of the common graphics file formats. Once you embed a photo into a *Word* document, and allow that program to save the file, we will never be able to recover a photo as good as the one with which you started! Oh, and please include Figure captions for all of the graphics in your article.

With your help, we continue to strive to make *QEX* the best that it can be. Thanks!

David Bern, W2LNX

8809 Cold Spring Rd, Potomac, MD 20854; w2lnx@arrl.net

A PS/2 Keyer: Using a Keyer Paddle to Emulate a PS/2 Keyboard and Mouse

Would you like to operate your computer without a keyboard? How about RTTY or PSK-31 without a keyboard? Now you can.

This article describes my first programming project with a Microchip PIC® microcontroller. The program for this project, written in *C*, emulates a PS/2 keyboard and a PS/2 mouse using a CW keyer paddle for input.

Note: an earlier version of this article was published in *The 28th Annual ARRL and TAPR Digital Communications Conference Proceedings*. The 28th Annual DCC was held in Chicago, Illinois on September 25-27, 2009.

Over the past several years, I became interested in learning how to program Microchip PIC microcontrollers and I began to look for an interesting project. I am an experienced *C* programmer, but I knew nothing about PIC microcontroller programming. My criteria for the project was that it needed to have a well defined input, a well defined output, and the circuit needed to consist solely of a PIC microcontroller with some light emitting diodes (LED). And, importantly, it needed to be written in *C*.

At the 26th DCC, Milt Cram, W8NUE and George Heron, N2APB introduced their NUE-PSK digital modem.[1] The full details of the NUE-PSK modem were published in the Mar/Apr 2009 issue of *QEX* (NUE-PSK Digital Modem Enables PSK31 Field Operation Without Using a PC!), along with a summary article in the March 2008 issue of *QST*. The NUE-PSK modem is a small device that provides portable PSK31 (and now RTTY) operation without the use of a personal computer. It does, however, require a PS/2 keyboard for entering text. It occurred to me that it could be more portable if the large PS/2 keyboard were replaced with a CW keyer paddle. A PIC microcontroller would translate CW input sent on the paddle into the output from a standard PS/2 keyboard. Hence, I found the idea for the project for which I was looking. I would write a program that runs on a PIC, and that emulates a PS/2 keyboard using a keyer paddle for input. Later on in the project, I wondered if it was also possible to emulate a PS/2 mouse with only two switch contacts and no other moving parts.

Background

Morse Code

Morse code input is defined in the ITU recommendation on the international Morse code and is further described in an article in Wikipedia.[2, 3] The CW character and word timings needed for this project are the length of time of a *dash* relative to a *dot*, the length of time between *dot*s and *dashe*s in a letter, the length of time between letters and the length of time between words.

The PS/2 Protocol

The output of a PS/2 keyboard and a PS/2 mouse follow the PS/2 protocol, which was originally described in the *IBM Personal System/2 Hardware Technical Reference*, in the sections on the "101 and 102 Key Keyboard" and the "Keyboard and Auxiliary Device Controller" sections of the computer reference manual, respectively.[4, 5] Articles on the Internet about the PS/2 protocol, the PS/2 keyboard protocol, and the PS/2 mouse protocol were most helpful.[6, 7, 8] More recently, an article describing a keyboard-game interface using the PS/2 protocol appeared in *Nuts and Volts* magazine.[9]

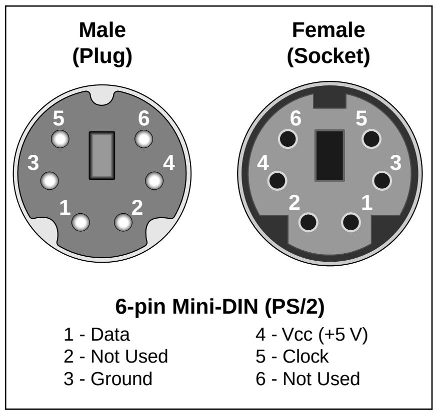


Figure 1 — The PS/2 ports for the keyboard and mouse on personal computers use six pin mini-DIN connectors.

The physical PS/2 interface, shown in Figure 1, used for the keyboard and mouse connectors uses the PS/2 protocol. The PS/2 protocol is a two way synchronous protocol used to communicate between a host and a device. (See Note 6.) A *host*, typically, is a personal computer and a *device*, typically, is a keyboard or a mouse. In addition to a clock line and a data line, there is a 5 V line and a ground line. The PS/2 protocol uses the clock line and the data line for sending data. The frequency of the clock is within the range of 10 to 16.7 kHz. Data is sent as an 11 bit frame, starting with a zero start bit, the eight data bits, with the least significant bit first, an odd parity bit and one stop bit. The device always generates the clock sig-

[1]Notes appear on page 8.

nal. The host can request to communicate with the device by pulling the clock line low for at least 100 µs, then pulling the data line low and then releasing the clock line back high. The device responds by generating the clock signal for the host to send its data to the device. Finally, the device acknowledges that it has received the complete data from the host.

The PS/2 Keyboard Protocol

Each key on a PS/2 keyboard device is identified by a unique scan code. Scan code set 2 is commonly used today and it was originally developed by IBM for the AT keyboard.[10] The keyboard device sends to the host the scan code of a key that is pressed; a break code and its scan code is then sent when that key is released. The host computer can communicate with the keyboard device. For example, the host commands the keyboard to light its Caps Lock LED when the Caps Lock key is pressed.

The PS/2 Mouse Protocol

The standard PS/2 mouse device sends the host its movement and button information as a three byte packet. Mouse movement information is sent as a relative position change and is a signed nine bit two's complement binary number. Also, a standard PS/2 mouse has a left, a middle, and a right button. Initially, the host communicates with the mouse to configure it after the host has determined whether the mouse is a standard PS/2 mouse or an enhanced PS/2 mouse.

The Project

Learning the PS/2 Protocol

To gain an understanding of the PS/2 protocol, I needed to see the data and the clock lines at the wire level on a PS/2 cable.

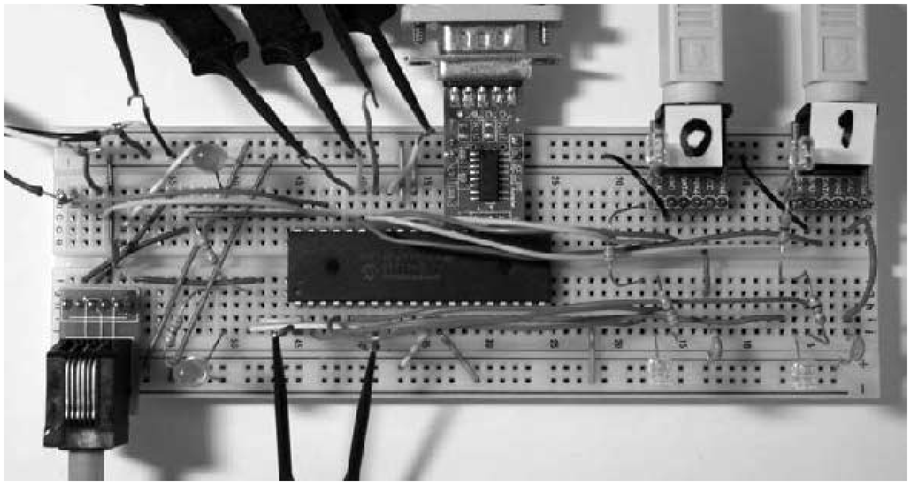


Figure 2 — This was my development breadboard. At the top of the photo, the four logic analyzer probes are connected to the clock and data lines of the keyboard and mouse connectors. The other two logic analyzer probes are connected to the –MCLR pin and a mark pin.

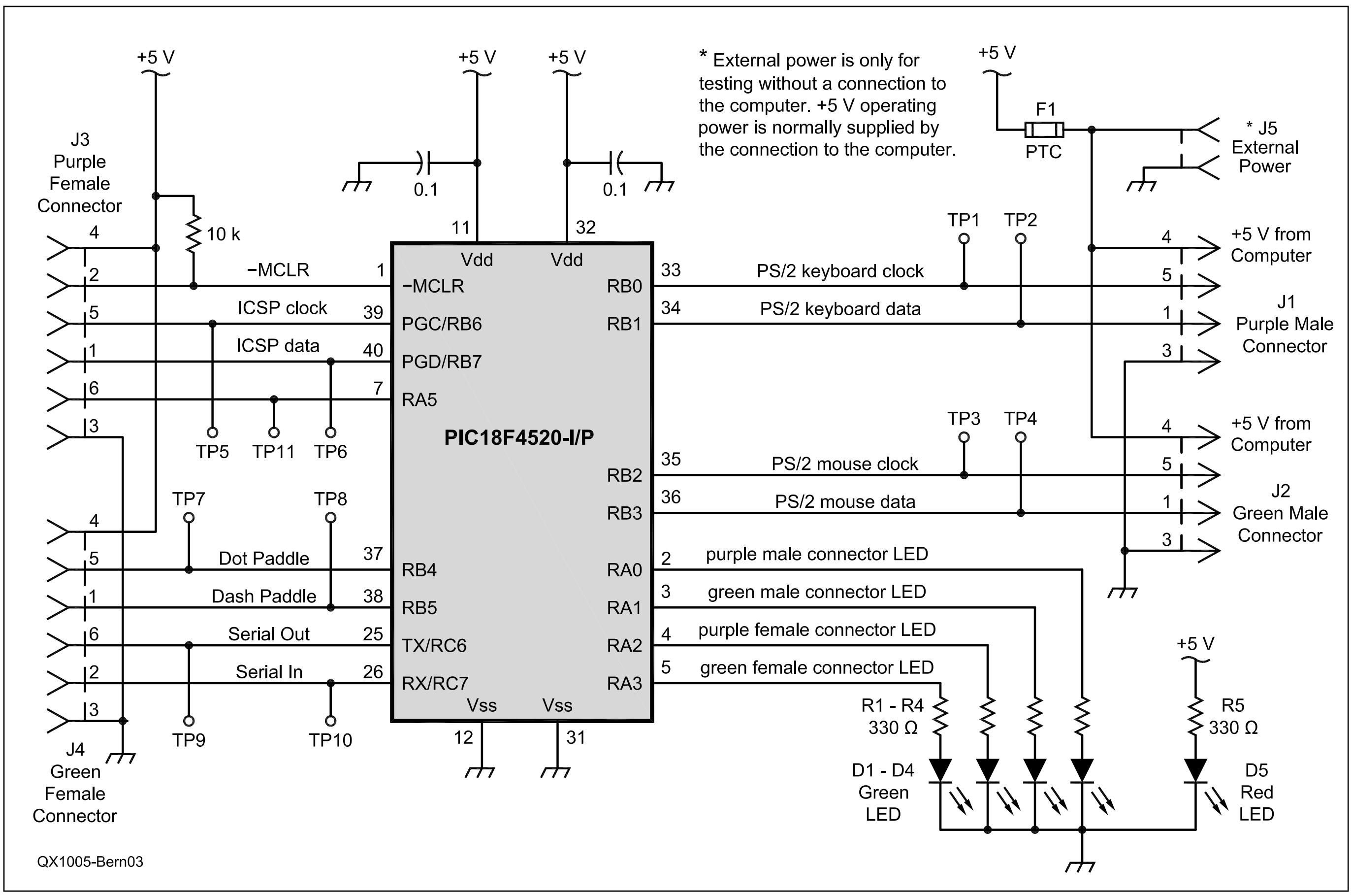


Figure 3 — This is the schematic diagram of the PS/2 Keyer.

I assembled a six pin mini-DIN connector breakout board from parts obtained from SparkFun Electronics, a wonderful resource for digital electronics hobbyists.[11, 12] Sample PS/2 protocol data came from an original IBM AT computer PS/2 keyboard and a Logitech PS/2 three button mouse. I purchased an inexpensive logic analyzer from Saleae.[13] Since the PS/2 data bits are sent in "reverse" order — least significant bit first — I created a paper form to record the start bit, eight data bits, parity bit, stop bit and any acknowledgment bit. Later in the project, I purchased a USBee SX logic analyzer, since it can decode the PS/2 protocol for the host and the device.[14]

PIC18F4520 Prototyping Boards

I needed to choose a PIC microcontroller for my project. Rick Hambly, W2GPS, develops and sells time related products using PIC microcontrollers and he recommended that I use the 18F series PIC microcontrollers. I purchased the PIC18F4520 development kit from CCS: it includes a prototyping board, and most valuably, an exercise booklet to help get started.[15] Note that the PIC18F4520 prototyping board and exercise booklet can be purchased separately from the complete kit. Other prototyping and demonstration boards I have tried that use the PIC18F4520 are the PICDEM 2 Plus demonstration board from Microchip, the Dem2PLUS demonstration board from Sure Electronics, and the Olimex 40 pin bare prototyping board with RS-232 from SparkFun.[16, 17, 18] A USB version of the Olimex 40 pin bare prototyping board is also available.[19]

I assembled a prototyping board as shown in Figure 2 by using a breadboard purchased from Beginner Electronics, a PIC18F4520 microcontroller in the PDIP form factor, two six pin mini-DIN connectors, an ICD programming connector and a serial TTL to RS-232 adapter.[20, 21, 22, 23] I connected the 5 V power supply to the prototyping board with a positive temperature coefficient (PTC) resettable fuse device, available from Spark Fun, at both mini-DIN connectors and at the ICD programming connector.[24] These devices protect the prototyping board and power supply sources against an accidental short circuit. The two mini-DIN connectors are connected to a personal computer using a PS/2 keyboard-and-mouse-to-USB adapter and two PS/2 male-to-male cables.[25, 26] The ICD connector is connected to a PIC programmer with a short cable. As luck would have it, the receive and transmit pins of the RS-232 adapter match the corresponding input/output pins on the PIC18F4520. Note the probes connected to the logic analyzer in Figure 2.

The circuit schematic diagram shown in Figure 3 consists of essentially one component — the PIC18F4520 microcontroller. The PS/2 keyboard connector, the PS/2 mouse connector and the CW keyer paddle is connected to port B pins of the PIC to take advantage of the internal pull-up resistors provided within the PIC. No external clock crystal or resonator is needed since the internal 8 MHz clock within the PIC is used instead.

PIC Development Tools

For the PIC programmer, I used the Microchip PICkit™ 2 programmer/debugger.[27] The PICkit 2 is inexpensive and was adequate for this project. It has a convenient power management feature: it provides power to the development board if it detects that the board has no power. Also, power to the board can be turned on and off from a software menu item.

After evaluating free demonstration versions of several PIC *C* compilers, I chose the CCS *C* compiler since it appears to best hide hardware details of PIC microcontrollers with library functions.[28] The port input/output library functions are easy to use and interrupt service routines are easy to write. Also, Rick Hambly, W2GPS, recommended the CCS *C* compiler since he uses it extensively for his work.

For the actual development environment, I preferred to use the Microchip MPLAB® integrated development environment (IDE) instead of the one provided by CCS with their *C* compiler suite.[29] Note that the less expensive CCS PCH *C* compiler instead of the full CCS *C* compiler suite is sufficient for this project since PCH integrates with the MPLAB IDE.[30]

Learning PIC *C* Programming

My first goal was to repeatedly blink an LED on and off and to display "hello, world" on a *HyperTerminal* serial terminal window. I read several introductory books on PIC *C* programming to get me started.[31, 32, 33] There are several other good books to read, as well.[34, 35, 36] *Nuts & Volts* and *Circuit Cellar* magazines also have good articles on PIC microcontroller development.[37, 38]

For program testing and debugging, I learned to use LEDs and timing pulses on an output pin to observe with a logic analyzer. I used the `printf()` function to print configuration and debugging information during program development and testing. I also quickly learned *not* to introduce a timing error in the code by putting a `printf()` in the wrong place.

Writing the Program

This program is organized into three software modules. The first module decodes CW characters keyed in with the paddle. The invalid CW character *di di dah dah* is used to indicate that the next character entered is a command code. Some command codes are keyboard Enter, keyboard Caps Lock, an

Table 1
The Parts List to Build the PS/2 Keyer on a Circuit Board

Part	*Quantity*	*Part Name*	*Vendor*	*Part Number*
C1, C2	2	0.1 µF 50 V 10% PC-Mount Capacitor	Digi-Key	BC1148CT-ND
D5	1	Red round diffused lens LED	Digi-Key	P589-ND
R1-R5	5	330 Ω ¼ W resistor	Digi-Key	P330BACT-ND
R6	1	10 kΩ ¼ W resistor	Digi-Key	10KQBK-ND
U1	1	40 pin IC socket	Digi-Key	3M5471-ND
U1	1	PIC18F4520-I/P	Digi-Key	PIC18F4520-I/P-ND
	4	Self-Stick Rubber Feet	Digi-Key	SJ5012-0-ND
J1, J3	1	PS/2 keyboard extension cable with purple connectors	Micro Center	SKU: 133314
J2, J4	1	PS/2 mouse extension cable with green connectors	Micro Center	SKU: 133272
J4	1	¼ inch In-Line Stereo Audio Jack	RadioShack	Catalog #: 274-141
D1-D4	4	Green rectangular clear lens LED	SparkFun	SKU: COM-08532
		Microchip PICkit 2 microcontroller programmer	Digi-Key	PG164120-ND
		Saleae Logic logic analyzer	SparkFun	SKU: TOL-08938
		Pro USB Converter: USB to PS/2 Keyboard and Mouse	Micro Center	SKU: 919712

error code and a switch to mouse mode. The error code generates the appropriate number of backspace characters to the beginning of a word just entered. CW is translated into ASCII text characters using a lookup table.

The paddle *dot* and *dash* contacts are connected to pins RB4 and RB5, respectively, of port B to take advantage of the change-on-input interrupt feature. This allows the PIC to go to sleep and consume practically no power while waiting for an interrupt to occur when a paddle lever is pressed.

Keyer paddle switch contact bounce was probably the most challenging problem of this project. A paper on switch bounce convinced me that a keyer paddle without any switch contacts, such as the Touchkeyer paddle is necessary for this project.[39, 40] An internal timer interrupt is used to measure time between *dot*s and *dashe*s. Using port B input interrupts and timer interrupts simplified the code.

The second module emulates a PS/2 keyboard using the PS/2 protocol. ASCII characters are translated into PS/2 keyboard scan codes using a lookup table. The lookup table has every character of a PS/2 keyboard, including those not found in Morse code. It was a thrill to first see the letter Q generated by the PIC microcontroller appear in a *NotePad* text editor window!

The third module emulates a PS/2 standard three-button mouse. It generates PS/2 mouse button clicks and mouse pointer movement from the keyer paddle input. Clicking — briefly pressing — the left paddle lever generates a left mouse button click and, correspondingly, clicking the right paddle lever generates a right mouse button click. Mouse pointer movement is controlled by pressing and holding the paddle lever: the left contact controls the left-right mouse pointer movement; the right contact controls the up-down mouse pointer movement. Pressing both sides moves the mouse pointer along one diagonal direction or the other diagonal direction. Hence, there are eight possible mouse movement directions, and it is possible to move the mouse pointer in an

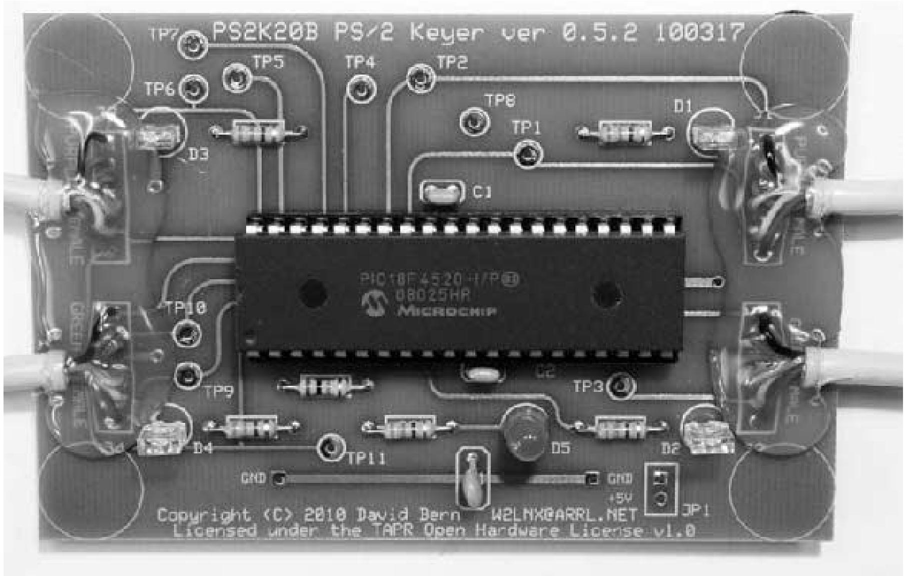


Figure 4 — On the printed circuit board, there is a green LED in a transparent plastic lens next to each cable connection. Hot glue is used to protect and secure the connections from the PS/2 cables to the circuit board. The circles in the corners indicate where rubber feet are placed under the circuit board.

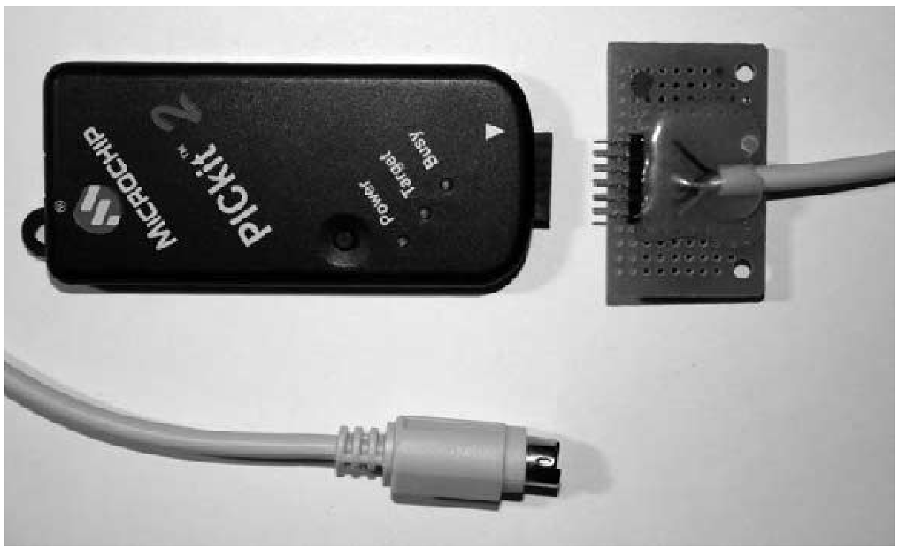


Figure 5 — This photo shows a PICkit 2 programming cable wired using the connections shown in Table 2. Hot glue protects the cable connections to a small prototyping board. The dark colored dot on the top left corner of the perf board is oriented with the white triangle on the PICkit 2.

Table 2A

PS/2 Mouse Extension Cable with Green Connectors and PS/2 Keyboard Extension Cable with Purple Connectors

Pin	*Wire Color*
1	Black
2	Violet
3	Red
4	Green
5	Yellow
6	Blue

Table 2B

Connections from the PICkit 2 Programmer Male Connector to a PS/2 Male Connector PICkit 2 Programming Cable

Pin	*PICkit 2 Male Connector*	*Pin*	*PS/2 Male Connector*
1	–MCLR	2	–MCLR
2	V_{DD}	4	V_{DD}
3	ground	3	ground
4	ICSP PGD	1	ICSP data
5	ICSP PGC	5	ICSP clock
6	not connected	6	not connected

Table 3A
Cable From Bencher CW Paddle to a ¼ Inch Stereo Phone Plug, RadioShack Catalog no.: 274-139.

Paddle Contact		*¼ Inch Stereo Phone Plug*
Dot	Left	Tip
Dash	Right	Ring
Ground		Sleeve

Table 3B
Connections Between a ¼ Inch Stereo In-Line Jack, RadioShack Catalog no.: 274-141, and a PS/2 Male Connector.
CW Paddle Cable

¼ Inch Stereo In-Line Jack		*PS/2 Male Connector*
Dot	Brass Tab	5
Dash	Silver Tab	1
Ground		3

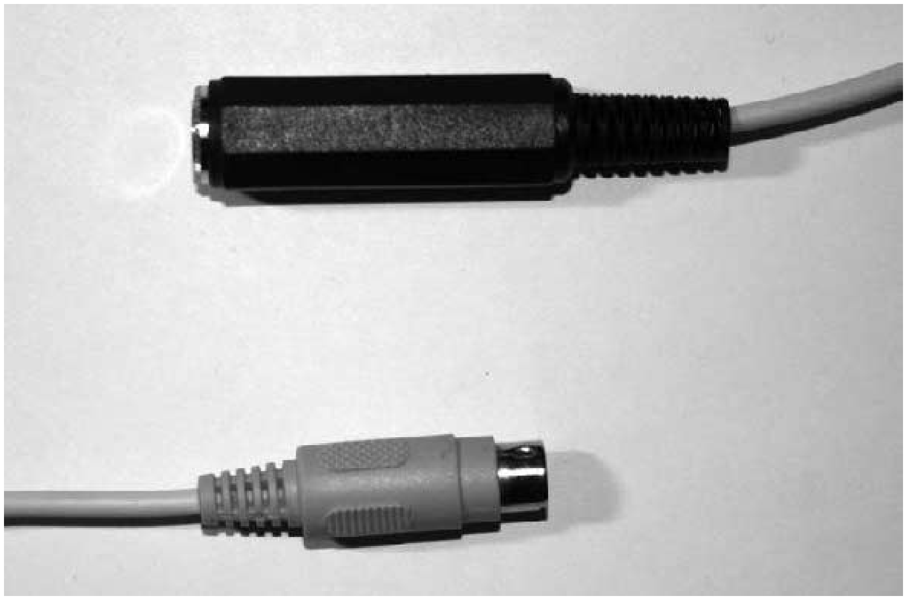


Figure 6 — This shows a CW paddle cable wired using Table 3B.

octagon. Pressing for longer than about three seconds causes the mouse pointer to move faster on the screen. A command code of three left clicks switches the program back to keyboard mode. Again, it was fun to see the mouse pointer move by itself in a continuous circle on the screen during initial testing. It took about two weeks of testing and experimenting with different mouse pointer movement policies to get something reasonable to control mouse pointer movement.

Testing during software development was not difficult. A PS/2 keyboard and mouse to USB adapter, a logic analyzer and LEDs were used. During program startup, the PIC would send a PS/2 keyboard and a PS/2 mouse device reset code via the PS/2 to USB adapter to the host computer. The computer would then respond with PS/2 device configuration command codes. This made it unnecessary to repeatedly plug and unplug the adapter USB connector in and out of the USB port on the computer. I used a logic analyzer to help my understanding of the host-to-device initial configuration dialog and for debugging my generated PS/2 keyboard scan codes and PS/2 mouse packets. I connected the logic analyzer probes to the PS/2 keyboard and the PS/2 mouse clock and data lines. Also, additional mark pulses were generated at points of interest in the code to identify logic analyzer points of interest. LEDs connected to pins on port A indicated program activity. A general status LED, an LED for the PS/2 keyboard port, and an LED for the PS/2 mouse port were used.

Building a Circuit Board

After the 2009 *Digital Communications Conference,* I designed a printed circuit board as shown in Figure 4. A PS/2 mouse extension cable with green connectors and a PS/2 keyboard extension cable with purple connectors were cut in half and attached to the circuit board with hot glue. The male green and purple connectors are connected to the corresponding purple and green female connectors of the PS/2 keyboard-and-mouse-to-USB adapter.

Table 1 provides the parts list of readily available components to construct a circuit board. Figure 5 shows a PICkit 2 programming cable wired according to the information in Table 2.

Figure 6 shows a CW paddle cable. On one end of a connecting cable, I wired an in-line ¼ inch stereo phone jack and on the other end I used a PS/2 male connector, wired according to Table 3B.

A sample program is shown in Figure 7. It continuously blinks all the port LEDs. This program is useful for doing a basic test of the circuit board after its construction.

Future Enhancements

The current version uses a PS/2 keyboard-and-mouse-to-USB adapter to connect the PIC18F4520 microcontroller to the host computer. My next version will use a PIC18F4550 microcontroller, since it contains an internal USB interface, eliminating the need for a PS/2-to-USB adapter.

Availability

The source code and the Express PCB circuit board files for this project are available for download from the ARRL *QEX* Web page.[41] The source code is released under the GNU General Public License, version 2 (GPLv2); the files for the printed circuit board are released under The TAPR Open Hardware License Version 1.0 (May 25, 2007).

Conclusion

This is an interesting and enjoyable project. I hope that this article inspires you to start a Microchip PIC microcontroller programming project. You can incrementally develop a microcontroller project of moderate complexity by systematically testing at every step of the way. Microchip PIC microcontrollers, programming tools and software development tools are readily available and are relatively inexpensive.

This project may be of possible use by people who have limited physical mobility who cannot use a conventional keyboard or a mouse. This could allow them to type on a computer or to manipulate a mouse pointer using just two switch contacts. Coupled with a "Puff and Sip Key," described in the March 2004 issue of *QST,* the PS/2 keyer could allow hands free operation of a computer.[42]

Acknowledgements

I would like to thank the following people for helping me with this project in some way: Adam Bern, KB3KVD, for being my keyer paddle mouse tester, Jim Johns, KAØIQT, for enlightening conversations about Microchip tools, Joe Julicher, N9WXU, for suggesting the PICkit 2 programmer, Larry Wolfgang, WR1B, for convincing me to work on this project, Milt Cram, W8NUE, and George Heron, N2APB, for developing their NUE-

PSK modem, Rick Hambly, W2GPS, for giving me guidance, and Steve Bible, N7HPR, for encouraging me to write up this project for the 2009 Digital Communications Conference.

Notes

[1]Milton Cram, W8NUE, and George L. Heron, N2APB, "NUE-PSK31: A digital modem for PSK31 Field Operation...Without Using a PC!," 2007 *TAPR and ARRL 26th Digital Communications Conference Proceedings*, Hartford, Connecticut: ARRL.

[2]International Morse code, "RECOMMENDATION ITU-R M.1677," International Telecommunication Union, 2004; available at **www.godfreydykes.info/international morse code.pdf**

[3]Morse code, Wikipedia, The Free Encyclopedia; **http://en.wikipedia.org/wiki/Morse_code**

[4]Keyboard 101- and 102-Key, *IBM Personal System/2 Hardware Interface Technical Reference — Common Interfaces*, 1990; **www.mcamafia.de/pdf/pdfref.htm**

[5]Keyboard and Auxiliary Device Controller, *IBM Personal System/2 Hardware Interface Technical Reference — Common Interfaces*, 1990, **www.mcamafia.de/pdf/pdfref.htm**

[6]Adam Chapweske, *The PS/2 Mouse/Keyboard Protocol*, 2003; **www.computer-engineering.org/ps2protocol/**

[7]Adam Chapweske, *The PS/2 Keyboard Interface*, 2003; **www.computer-engineering.org/ps2keyboard/**

[8]Adam Chapweske *The PS/2 Mouse Interface*, 2003; **www.computer-engineering.org/ps2mouse/**

[9]Jochen Jahn, "Keyboard-Game Interface," *Nuts & Volts Magazine*, Dec 2009, pp 30-34.

[10]Adam Chapweske, *Keyboard Scan Codes: Set 2*; **www.computer-engineering.org/ps2keyboard/scancodes2.html**

[11]SparkFun Electronics, MiniDIN 6-Pin Connector, SKU: PRT-08509; **www.sparkfun.com/commerce/product_info.php?products_id=8509**

[12]SparkFun Electronics, MiniDIN 6-Pin Connector Breakout, SKU: PRT-08651; **www.sparkfun.com/commerce/product_info.php?products_id=8651**

[13]Saleae LLC, Saleae logic analyzer; **www.saleae.com/logic/**

[14]CWAV, Inc, USBee SX logic analyzer; **www.usbee.com/sx.html**

[15]CCS, Inc., PIC18F4520 Development Kit; **www.ccsinfo.com/product_info.php?products_id=18F452kit**

[16]Microchip Technology Inc., PICDEM 2 Plus, Part Number: DM163022; **www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en010072**

[17]Sure Electronics Co, Hybrid of PIC18F4520 Dem2PLUS and Low Power Demo Board; **www.sure-electronics.com/product/goods.php?id=24** and **www.sureelectronics.net/goods.php?id=24**

[18]SparkFun Electronics, 40 Pin PIC Development Board, SKU: DEV-00021; **www.sparkfun.com/commerce/product_info.php?products_id=21**

[19]SparkFun Electronics, 40 Pin PIC


```
#include <18F4520.h>
#use delay(internal=8MHZ)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)

#define PORT0_LED PIN_A0
#define PORT1_LED PIN_A1
#define PORT2_LED PIN_A2
#define PORT3_LED PIN_A3

#define DELAY 250

void blink_port_LEDs(void);

/* -------------------------------------------------- *
 * this program continuously blinks all the port LEDs *
 * -------------------------------------------------- */

void main(void)
{
  delay_ms(DELAY);
  printf("Figure 7 sample program... hello\r\n");

  for (;;) {
    blink_port_LEDs();
  }
}


/* --------------------------------- *
 * this routine blinks each port LED *
 * --------------------------------- */

void blink_port_LEDs(void)
{
  output_high(PORT0_LED); delay_ms(DELAY);
  output_low(PORT0_LED); delay_ms(DELAY);

  output_high(PORT1_LED); delay_ms(DELAY);
  output_low(PORT1_LED); delay_ms(DELAY);

  output_high(PORT3_LED); delay_ms(DELAY);
  output_low(PORT3_LED); delay_ms(DELAY);

  output_high(PORT2_LED); delay_ms(DELAY);
  output_low(PORT2_LED); delay_ms(DELAY);
}
```

Figure 7 — This is a sample *C* program for the PIC18F4520 using the CCS *C* compiler.

Development Board with USB, SKU: DEV-00022; **www.sparkfun.com/commerce/product_info.php?products_id=22**

[20]Beginner Electronics, Breadboard and Wire Kit; **www.beginnerelectronics.com/beginner/Products.php**

[21]Microchip Technology Inc, PIC18F2420/2520/4420/4520 Data Sheet: 28/40/44-Pin Enhanced Flash Microcontrollers with 10-Bit A/D and nano-watt Technology (document DS39631E), 2008; **www.microchip.com**

[22]SparkFun Electronics, Adapter board for Microchip ICD and ICD2; **www.sparkfun.com/commerce/product_info.php?products_id=193**

[23]HVW Technologies, RS-232 Driver Module — DCE; **www.hvwtech.com/products_view.asp?ProductID=289**

[24]SparkFun Electronics, PTC Resettable Fuse, SKU: COM-08357; **www.sparkfun.com/commerce/product_info.php?products_id=8357**

[25]Inland Pro USB Converter USB to PS/2 Keyboard and Mouse, SKU: 919712; **www.microcenter.com/single_product_results.phtml?product_id=0230515**

[26]Cable Club, 6 ft PS/2 Keyboard and Mouse Interface Cable (Male/Male), part: BC20277-6; **www.cableclub.com/keyboard-mouse-interface-cable-malemale-p-797.html**

[27]Microchip Technology Inc, PICkit 2 Development Programmer/Debugger; **www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en023805**

[28]CCS, Inc, Compiler Exclusively for Microchip PIC® MCUs, **www.ccsinfo.com/content.php?page=compilers**

[29]Microchip Technology Inc, MPLAB Integrated Development Environment; **www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002**

[30]CCS, Inc, PCH Command-line Compiler for PIC18 MCU parts; **www.ccsinfo.com/product_info.php?cPath=Store_Software&products_id=PCH_full**

[31]Benson, David, C *What Happens, Using PIC® Microcontrollers and the CCS* C *Compiler*, Hayden, Idaho; Square 1 Electronics, 2008; **www.sq-1.com/cwhtoc.html**

[32]Chuck Hellebuyck, *Beginner's Guide To Embedded* C *Programming: Using the PIC® Microcontroller and the HITECH PICC-Lite™* C *Compiler*, Milford, Michigan; Electronic Products, 2008, ISBN: 978-1438231594; **www.elproducts.com/embeddedcbook.htm**

[33]Martin P Bates, *Programming 8-bit PIC Microcontrollers in* C*: with Interactive Hardware Simulation*, Oxford, UK: Newnes Press, 2008; ISBN: 9780750689601

[34]Chuck Hellebuyck, *Beginner's Guide to Embedded* C *Programming — Volume 2: Timers, Interrupts, Communication, Displays and More*, Milford, Michigan: Electronic Products, 2009; ISBN: 978-1448628148, **www.elproducts.com/embeddedcbook2.htm**

[35]Dogan Ibrahim, *Advanced PIC Microcontroller Projects in* C*: From USB to RTOS with the PIC 18F Series*, Oxford, UK: Newnes Press, 2008; ISBN: 978-0750686112

[36]Richard H. Barnett, Sarah Cox, Larry O'Cull, *Embedded* C *Programming and the Microchip PIC*, Clifton Park, New York: Delmar Learning, 2004; ISBN: 978-1401837488

[37]*Nuts & Volts Magazine, The Magazine for the Electronics Hobbyist*, ISSN 1528-9885; **www.nutsvolts.com/**

[38]*Circuit Cellar, The Magazine for Computer Applications*, ISSN 1528-0608; **www.cuitcellar.com/**

[39]Jack G. Ganssle, *A Guide to Debouncing*, Rev 3: June, 2008; **www.ganssle.com/debouncing.pdf**

[40]CW Touchkeyer touch paddle, model P1PADW; **www.cwtouchkeyer.com/P1PADW.htm**

[41]The program source code file and circuit board pattern files in ExpressPCB format are available for download from the ARRL *QEX* Web site. Go to **www.arrl.org/qexfiles** and look for the file **5x10_Bern.zip**.

[42]Gary Gordon, K6KV, "Build a Puff-and-Sip Key," March 2004 *QST*, pp 31-32.

David Bern, W2LNX, was first licensed in 1979 as N2AER with an Advanced Class and upgraded to an Amateur Extra Class in 2000. Later, he obtained his W2LNX vanity call sign since he is also an avid Linux enthusiast. As a high school student, he earned his First-Class Radiotelephone Operator License with Ship Radar Endorsement. In 1977, he earned a BS in Computer Science from City College of New York and then in 1983 earned an MS in Computer Science from New York University. He is a professional software developer and also an adjunct professor of engineering at Montgomery College, Rockville, Maryland. He prefers building and experimenting with ham radio projects to operating but enjoys operating QRP digital modes in the summer with his son Adam, KB3KVD, in the Virginia mountains. Currently, he is learning microcontroller programming and digital signal processing.

QEX

Rubens R. Fernandes, ex-PY2QE

37 Borodino Ct, Greenwith, SA 5125, Australia; ruferna@gmail.com

A Frequency Counter for the Experimenter

Here is a classic workbench general purpose instrument, with better than 26 mVrms sensitivity from 30 Hz to 2.2 GHz.

Frequency counters are considered indispensable tools for experimenters and basic models can be found on the market for reasonable prices nowadays. Nevertheless, besides basic features like good sensitivity and variable gate times, I wanted something more, like the possibility of synchronizing to external high-stability time bases. (I have an old HP 10811-60164 time base and one of these days I will gather enough courage to build a GPS receiver and lock this unit to the 1 part per second cesium-based reference signal.) I also wanted a counter that could do all the boring math when connected to the VFOs of the QRP rigs I have been building for years, independently of the mixing strategy. It should also measure lower frequencies, down to audio range. As I didn't find such an instrument having a reasonable price, I started designing my own.

The final result was a two-channel counter, the first channel ranging from 30 Hz to 55 MHz, with high impedance, and the second channel from 55 MHz to 2.8 GHz, with a 50 Ω impedance. Four gate times were included: 0.01, 0.1, 1 and 10 seconds. It can measure period too, up to 25 kHz, and display the result as time or frequency, at your choice. We can also introduce an offset — there are pre-defined offset frequencies you can choose (based on the most common IFs found on homebrew and basic commercial rigs) or you can just input the frequency of your choice and it will be memorized. Although the main objective was sinusoidal signals, it will work with other waveforms, like square or triangular waves. A low-battery alarm was also included.

The frequency counter in this project is connected to a commercial RF generator.

Construction

Both channels must be very well shielded. Photo A shows the front end enclosure that I built. Another important thing is to provide a path as close as 50 Ω as possible from the Channel B connector (the high frequency channel) to the prescaler (LMX2326). I used a 0.1 inch trace on the circuit side. FR4 (fiber glass) material is a good choice for the board (although I have used phenolic material), with a ground plane on one side and the circuit on the other. Use surface mount devices (SMD) for both channels.

The original unit has four printed circuit boards: front end, power supply, time base and control board, and all boards were assembled using 0805 surface mount technology (SMT) components, except for the processor and some components here and there. I suggest a shielded enclosure for the unit, to reduce electromagnetic interference (EMI) problems. The main box was built using one sided phenolic boards. The lead photo shows the assembled unit.

It would be wise to orient the external sync connector to the outside — a quick connection type would be great, like a subminiature B (SMB) connector. I have used ordinary (though good quality) BNC connectors for the channel inputs.

Circuitry

Figure 1 shows the schematic diagram

of the front end circuit board. On the low frequency channel (Channel A), the FET provides high input impedance. It is followed by two amplifiers and a Schmitt trigger inverter that "digitizes" the signal. The high frequency channel (Channel B) input has a 3 dB pad that helps to define the 50 Ω path between the connector and the monolithic amplifier. The LMX2326 is a low power phase locked loop (PLL), but only its prescaler is used here. Channel A has an overall sensitivity of about 25 mV RMS from 30 Hz to 55 MHz, (from 0.2 Hz for square waves). Channel B has roughly the same sensitivity, from 55 MHz to 2.2 GHz. Above 2.2 GHz, the sensitivity is degraded to about 150 mV RMS at 2.8 GHz. Careful layout for the Channel B input could probably improve this behavior. The power supply of the channel not selected is switched off, to avoid interaction and to minimize power consumption. Maximum levels are 50 V peak for Channel A and +16 dBm for Channel B.

The power supply (see Figure 2) was provided with a protection circuit at the power

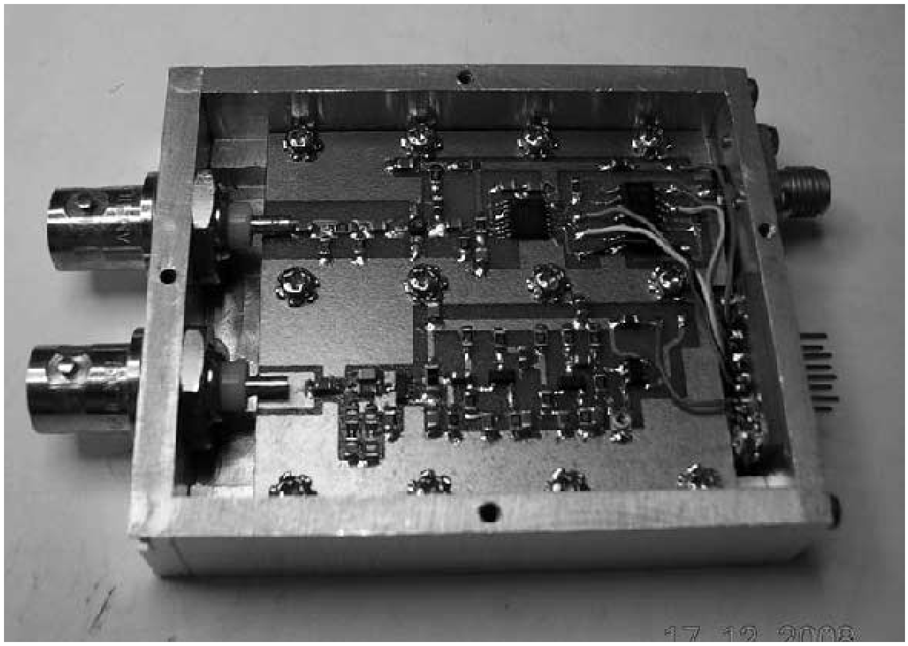


Photo A — It is best to build the counter front end circuitry into a shielded enclosure.

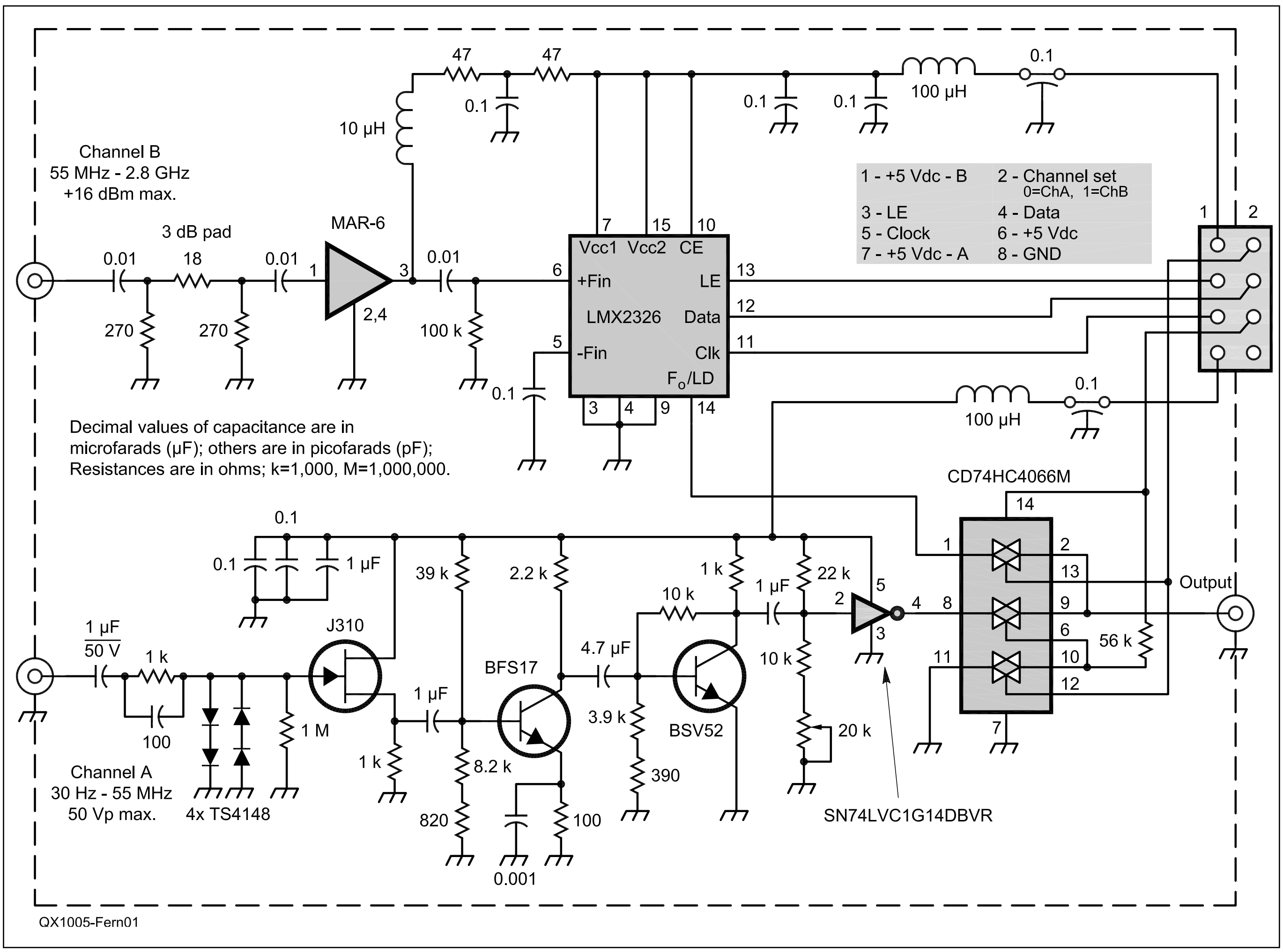


Figure 1 — Here is the schematic diagram of the counter front end circuitry.

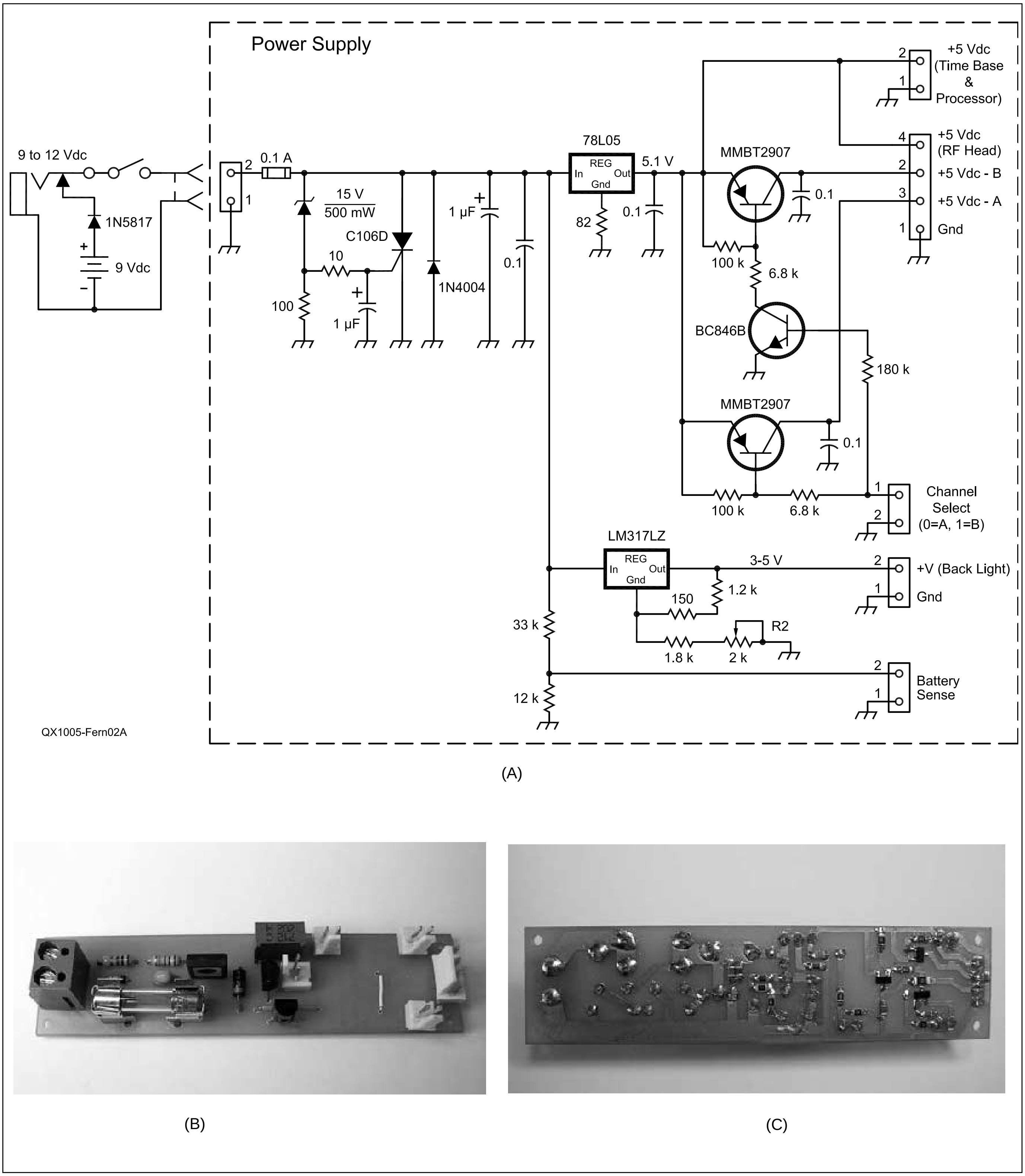


Figure 2 — The schematic diagram at Part A shows the counter power supply. Part B shows the top of the supply circuit board and Part C shows the bottom of the board, with circuit traces and SMT components.

input (9-12 V dc) that can be omitted if you have a tidy workbench and well behaved friends. The fuse will blow if you have an internal short circuit, if the supply leads are inverted or in the case of an overvoltage (>15 V dc). The power supply module generates 5 V dc for general use, switchable 5 V dc for the front end channels and an adjustable voltage for the display back light.

Figure 3 shows the schematic diagram of the internal time base, along with photos of the top and bottom of my circuit board. In the original counter, I used a commercial adjustable oscillator, disassembled from a piece of old NEC equipment. I suggest the two-transistor circuit of Figure 5 for the internal time base, noting that the capacitors that are part of the oscillator are all NPØ, including the trimmer — SMT fixed capacitors would be fine. I have tested this circuit with an ordinary crystal, and the circuit worked very well. It would be a very good idea to enclose the circuit with a small shielding box. When you connect an external time base signal with 1 Vpp, the internal reference is automatically switched off.

The PIC16F876A processor is the heart of the counter and it also controls the display, which requires much less power for its blue back light than the older green ones.

Adjustments

There are only four adjustments in the counter. First of all, adjust the trimpot on the processor board for the best display contrast.

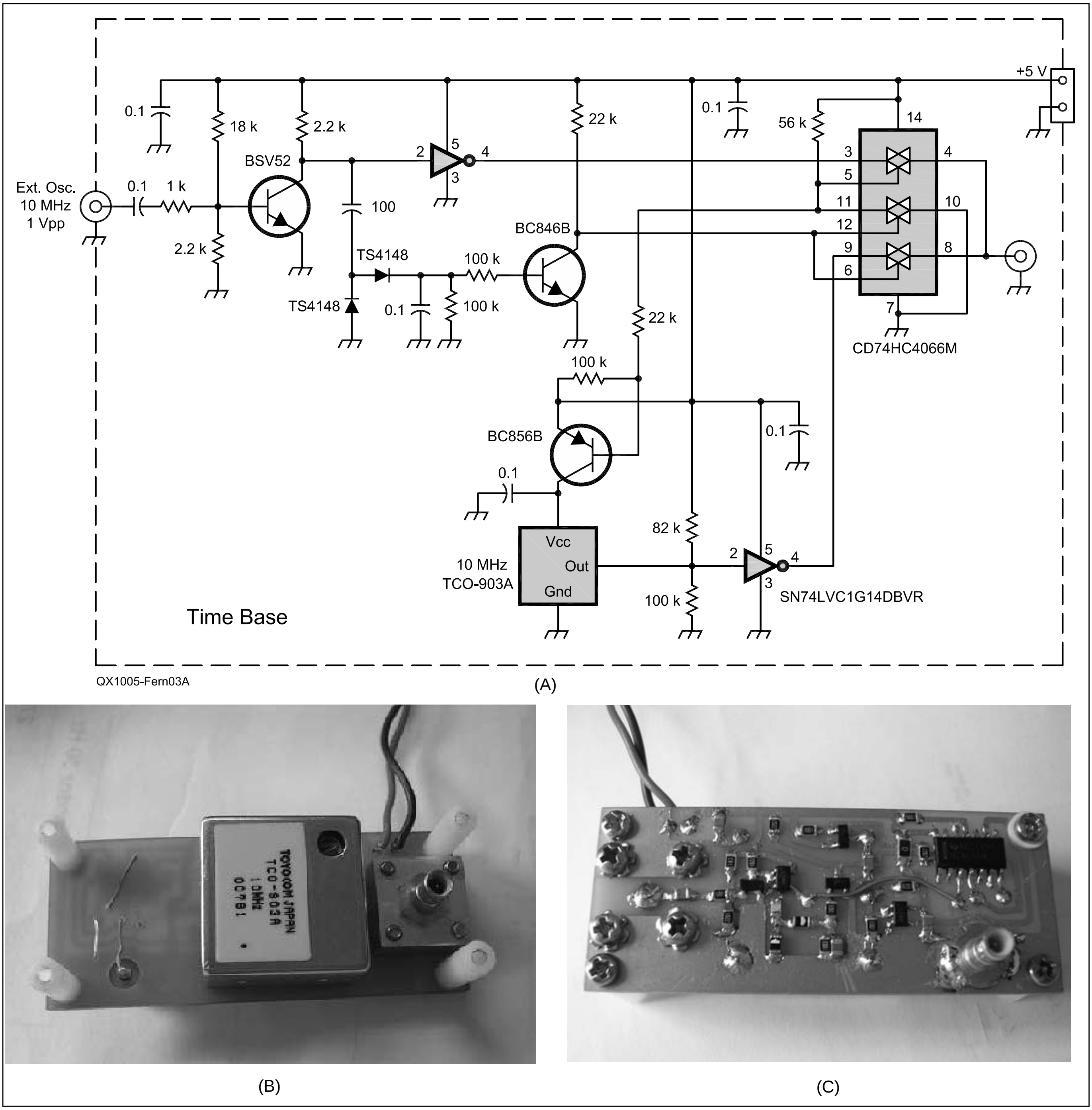


Figure 3 — Part A is the time base schematic diagram for the circuit I used. Part B shows the top of my circuit board, with the commercial 10 MHz module. Part C shows the bottom of my circuit board, with the circuit traces and SMT components.

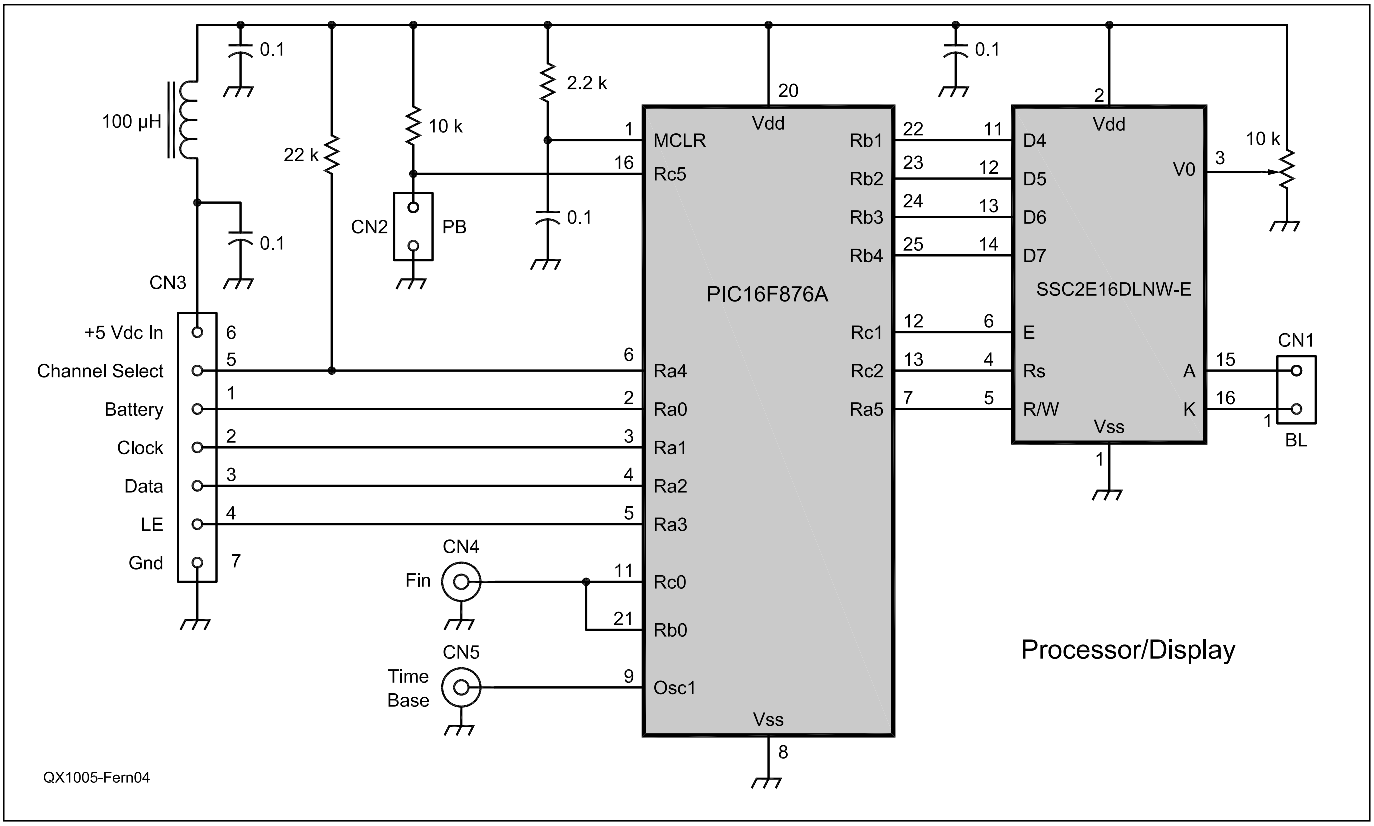


Figure 4 — The PIC 16F876A controller, shown in this schematic diagram, is the heart of the counter.

Now connect a 25 mV RMS, 50 MHz signal to Channel A and adjust the trimpot at the input of the inverter until this frequency (or close) is steadily shown on the display. Increase the frequency up to 55 MHz, always tweaking this trimpot.

Now connect a reference signal (10 MHz, for instance) to the Channel A input. Adjust the internal time base trimmer until the display shows the exact frequency, down to units of Hz. Note that the precision of this calibration adjustment will depend on the precision of this external reference signal.

Finally adjust the power supply module trimpot for a comfortable display back light — if you intend to use this counter frequently with batteries, it would be wise to adjust to the minimum voltage that will still give good readability.

Software and Operation

Excluding the **ON-OFF** switch, this is a one-button-only instrument — pressing and releasing the **MODE** button will give the SCROLL command, and pressing and holding the button will give the ENTER command. If you are in the main display, scrolling will toggle between Channel A

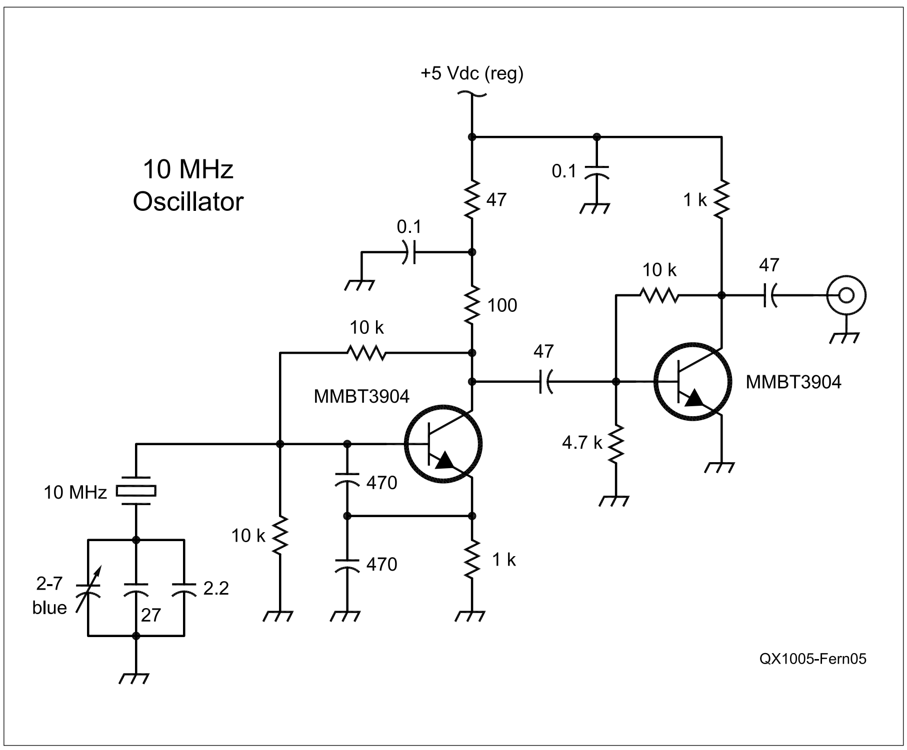


Figure 5 — This 10 MHz reference oscillator can be used as a reference oscillator for the counter.

and B and ENTER will take you to the main menu. The following choices will be shown: gate options, period measurements and offset.

Gate options: Select the gate time — 0.01, 0.1, 1 or 10 seconds. This is the time the gate will be open for the measurement. Faster measurements will give less precision.

Period: When you measure frequency, the gate will be kept open for a defined extension of time and the events will be counted — on the other hand, when you measure period, two subsequent events will respectively open and close the gate, and the time between the events will be measured. For this reason, measuring the period of low frequency signals will lead to more accurate results. You can measure very low frequency signals using this technique, and the result can be shown as frequency or as time. Remember that Frequency = 1 / Period. To leave the Period mode and start the Frequency mode, just choose a gate time.

Offset: Here you can add or subtract a frequency to the signal present at any input. There are some predefined frequencies you can choose, related to the IF channel of simple rigs, or you can use the "Extern" option: just input the offset frequency and choose this option. Example 1: suppose you have a 40 m (7 MHz to 7.1 MHz) CW rig, a 4 MHz IF filter and a 3 to 3.1 MHz VFO. In this case, connect the counter to the VFO and choose plus 4 MHz.

Example 2: Now you have the same band but a 10.7 MHz IF and the VFO ranging from 3.7 to 3.6 MHz. This time choose a –10.7 MHz offset, and everything will be just fine. The software will take care of the negative values and will show the correct frequency of the signal at the antenna.

There are many options you can use to load the hex file into the processor. Some of them (both software and hardware options) have been shared among Internet users. Particularly, I have downloaded the MPLAB IDE software from the Microchip Web site (IDE stands for Integrated Development Environment), because it has interesting tools for the developer and I have a compatible programmer.

A Word of Caution

This counter was designed for amateur use and I have purposely pushed the limits of the processor and prescaler to reach a desired performance. I have tested the circuit with samples I had, analyzing the circuit behavior with frequencies even higher than those stated here and I would expect the same performance for average chips. As a matter of fact, two more prototypes were assembled with other chips, leading to same results. Nevertheless, if you don't want to step into the dark side of the force and/or don't care about the world above 550 MHz, replace the LMX2326 with an LMX2306 (same pinout). The software, in this case, has to be changed and the channel B sensitivity will be around 5 mV RMS, from 25 MHz up. I suggest, in this case, that you replace the original front end pad with a 9 dB one, resulting in an overall sensitivity better than 10 mV RMS. The hex program files for both versions are available for download from the ARRL *QEX* files Web site. (See Note 1.)

Final words

I have been using this counter for over a year, and it is extremely useful, together with other home built equipment I have made. Many concepts used here were derived from good technical literature and research on the Internet, especially concerning component data sheets.[2, 3] The design of this counter was also possible due to good friends: Delson, PY2DME, helped me with measurements above 1 GHz. Eduardo, PY2GNZ, assembled and tested prototypes and Jorge, PY2PVT, made the circuit boards.

During the final stages of preparing this issue of *QEX* for the printer, I discovered that National Semiconductor recently discontinued production of the LMX2306/2316/2326 family. While these chips are still available from some sources, stock will be limited and they may become scarce in the near future. I learned that the Analog Devices ADF4116/4117/4118 family of ICs appear to be equivalent, in terms of specifications, functions, pinouts and packages, but I can't be sure that they will work with the same software, and provide the same performance without assembling a new set of counter boards and testing them. The Analog Devices ICs are available from Digi-Key. I would like to note that last year the LMX family of parts were easy to find at very good prices, but the market is very dynamic, and the manufacturers only want to produce parts for which there is a very large market.

Notes

[1]The author's hex code program file is available for download from the ARRL Web site *QEX* files area. Go to **www.arrl.org/qexfiles/** and look for the file **5x10_Fernandes.zip**.

[2]Wes Hayward, W7ZOI, Rick Campbell, KK7B, Bob Larkin, W7PUA, *Experimental Methods in RF Design*, ARRL, ARRL Order No. 9239, $49.95. ARRL publications are available from your local ARRL dealer or from the ARRL Bookstore. Telephone toll free in the US: 888-277-5289, or call 860-594-0355, fax 860-594-0303; **www.arrl.org/shop**; **pubsales@arrl.org**.

[3]Bob Okas, W3CD, "The Norcal Frequency Counter – FCC-1," ARRL, *QST*, Sep 2006, pp 28-32.

Rubens R. Fernandes earned a BS in Electrical Engineering in 1970, and worked for about 31 years in the telecommunications industry, in research, design and production. He retired about 5 years ago. He has been a licensed Amateur Radio operator since 1979, and held the call sign PY2QE while living in Brazil. He is a CW enthusiast. He now dedicates his spare time to home brewing small transceivers and test equipment. He has a small workshop that includes mechanical, electrical, software and silk-screening facilities.

QEX

Byron Blanchard, N1EKV

16 Round Hill Rd, Lexington, MA 02420; Byron@n1ekv.org

Adjusting BJT Oscillator Amplitude

Here are a few simple techniques to adjust a bipolar transistor oscillator, along with an explanation of why those techniques work.

Most oscillator circuits are published with no explanation of how component values were chosen or what oscillation amplitude to expect. This article will use a published bipolar junction transistor (BJT) crystal oscillator circuit as an example, to help you understand these designs. First, I will show how to set the oscillation amplitude using only dc measurements. You don't need any RF measuring instruments such as an oscilloscope or RF voltmeter. You set the amplitude by adjusting the dc bias current of the transistor while listening to hear when the oscillation just starts and stops. Later, I will explain why this simple procedure works.

Oscillation amplitude is important for any oscillator, but particularly so for a crystal oscillator. Excess crystal power dissipation can cause frequency drift or even damage the crystal. For any oscillator, the amplitude must be compatible with transistor limitations, power supply voltages, and desired output level.

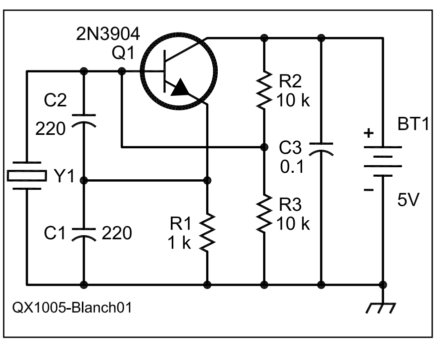


Figure 1 — Example oscillator circuit.
Y1 = 10 MHz (R_S = 30 Ω max)
Q1 = 2N3904
BT1 = 5 V
C1 = 220 pF (Z = –*j* 72 Ω)
C2 = 220 pF (Z = –*j* 72 Ω)
C3 = 0.1 µF bypass
R2 = 1 kΩ (but will be adjusted)
R3 = 10 kΩ
R4 = 10 kΩ

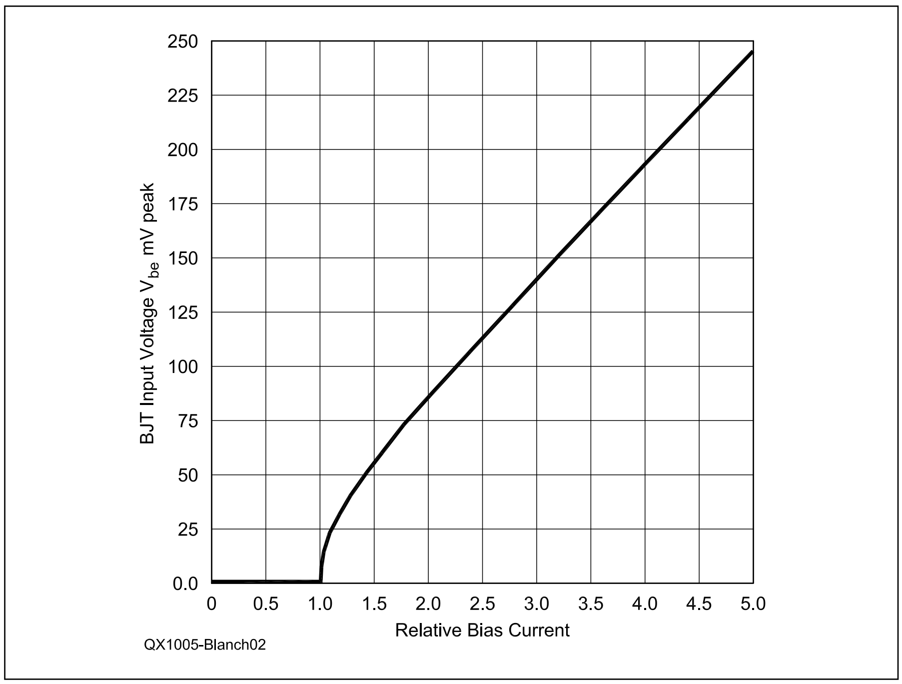


Figure 2 — BJT input voltage (oscillation amplitude) versus relative bias current.

You can set the oscillation amplitude of the oscillator circuit in Figure 1 by adjusting the value of R1. R1 sets the dc bias current, which directly controls transistor gain (transconductance). The method requires no RF instrumentation, only dc measurements. You do need some way, such as listening with a receiver, to tell if the circuit is oscillating. The example is a Pierce (Colpitts-like) crystal oscillator with component values from an arbitrarily-chosen published circuit. The method also applies to most common BJT LC oscillator circuits such as the Colpitts, Clapp, Seiler, Vackar, and Hartley types. I will explain why this procedure works later in the article.

Figure 2 shows how oscillation amplitude depends on transistor dc bias, the current through R1. The current scale is *relative* to the current at which oscillation just starts, which is defined as *one* on the current scale. For a relative current value less than one, the oscillation amplitude is zero — the circuit isn't oscillating. For relative current values greater than one the circuit does oscillate with amplitude shown by the curve of Figure 2. For relative current of two the amplitude is about 86 mV peak, measured between the base and emitter of the transistor.

To set oscillation amplitude you need first to determine the actual or absolute value of bias current corresponding to the

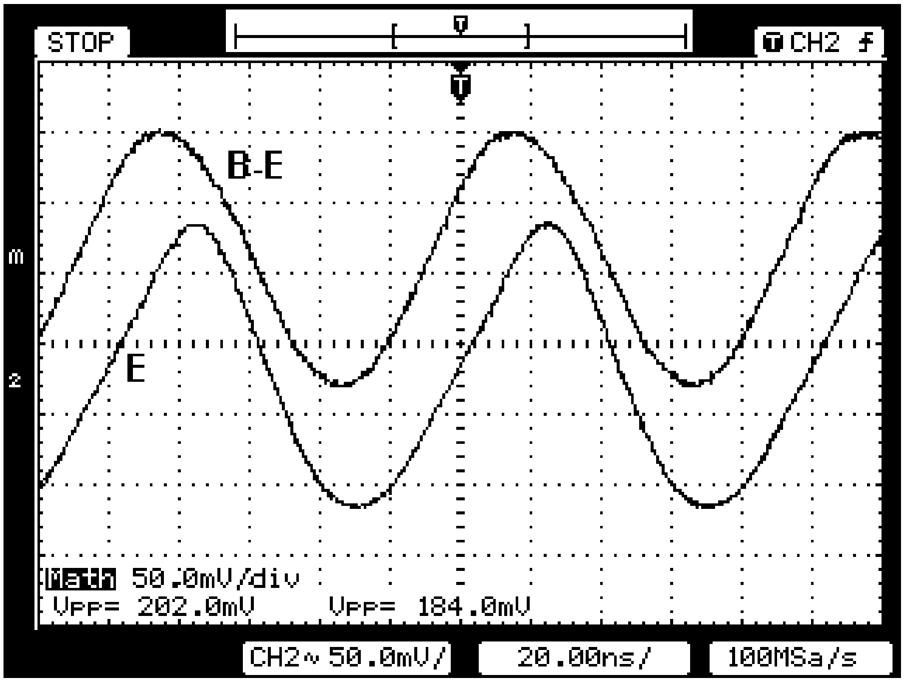


Figure 3 — Waveforms for 324 µA bias current.

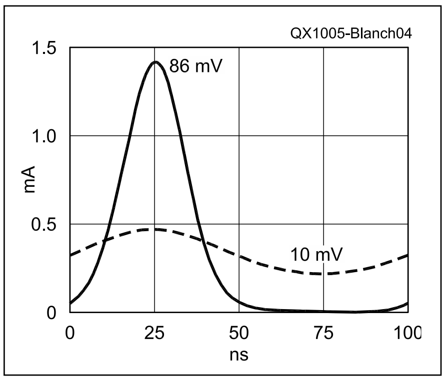


Figure 4 — Simulated collector current waveforms. The 86 mV and 10 mV values refer to the peak values of the two V_{be} waveforms.

relative bias current value of *one* in Figure 2. Adjust R1 to the value at which the oscillator just barely starts and stops. Measure the resulting direct current in R1, which is the *absolute* current value corresponding to the *relative* current value of one. Then adjust R1 to increase the current by the factor required for the desired oscillation amplitude. The new bias current is the previously measured just-starting bias current multiplied by a factor from the horizontal axis of Figure 2. The relationship plotted in Figure 2 is valid for any bipolar junction transistor (BJT) because it derives from the basic physics of a BJT.

Now let's see if this works. I made a breadboard of the circuit in Figure 1. I replaced R1 with a potentiometer plus a 100 Ω current sense resistor and measured the voltage across the sense resistor with a digital voltmeter. I connected 10× scope probes to the base and emitter of Q1 in order to see waveforms and measure voltages. I initially set R1 to the 1 kΩ value specified in the published circuit.[1] I tuned a receiver (in CW mode) to the oscillator frequency then increased the value of R1 until I heard the oscillation just stop. The bias current through R1 was then 162 µA, found by measuring 16.2 mV across the current sense resistor and dividing by the 100 Ω value of the resistor. I then adjusted R1 for a *relative* bias current of 2 by setting the current to 162 µA × 2 = 324 µA. Figure 2 predicts an oscillation voltage of 86 mV peak across C2 at the input of the BJT. Using a relative bias current of two was an arbitrary choice, a nice round number, but not a general rule of thumb. I recommend choosing a relative bias current no higher than five in order to limit peak collector currents.

Figure 3 shows the resulting voltage waveforms. The E waveform is the measured voltage at the emitter of Q1. The B-E waveform is the base-emitter voltage, V_{be}, computed by subtracting the E (emitter) voltage from the B (base) voltage. This base-emitter voltage. V_{be} measures 184 mV peak-to-peak or 92 mV peak compared to the predicted 86 mV peak. Not bad!

Waveform B-E, the transistor input voltage, V_{be}, is a low-distortion sine wave because it is filtered by the crystal and C2. The E waveform is the transistor output voltage, V_{ce}, which is the voltage across C1 (inverted because the ac ground is at the collector). This E waveform has a slightly larger magnitude than, and is phase shifted from, the B-E waveform because the impedance at that point has a reactive component. The distortion in the E waveform results because the transistor output current is badly distorted, as seen in Figure 4. The only harmonic filtering for the voltage at the emitter is by C1, since the crystal looks like an open circuit at harmonic frequencies. The peak collector current, 1.4 mA, is four times the average current.

Now let's calculate the crystal power dissipation. The crystal current is the same as the current in C2. The voltage across C2 is the 86 mV peak base-emitter voltage of Q1 set previously. From Ohm's Law, the current is that voltage divided by the impedance of C2, Z = 1 / (2 × pi × f × C) = 72 Ω. I = E / Z = 86 mV / 72 Ω = 1.2 mA pk. Crystal power is I^2R / 2, where R is the crystal resistance. The resistance spec for this crystal is 30 Ω maximum, so maximum power is $(1.2 \text{ mA})^2$ × 30 Ω / 2 = 22 µW. A typical crystal would probably have a resistance of half of the maximum spec and so only half the power dissipation. If you want to drive the crystal harder by increasing the crystal current you should increase the value of C2, not the relative bias current.

You now have an oscillator with known values of both oscillation amplitude and crystal current (and hence power dissipation) but have not connected any load to it. The most obvious place to connect a load is to the emitter of Q1, but be careful. Since transistor Q1 is an emitter follower for dc, you might expect that it would have a low output impedance at the oscillator frequency. Wrong! At the oscillation frequency Q1 is acting as a common-emitter amplifier. If Q1 was instead acting as an emitter follower, the ac magnitude of V_{be} would be small, only that needed to supply the load current. But in an oscillator, V_{be} is a large sine wave whose amplitude is defined by the resonant circuit and transistor, not by the load current (except indirectly). Another way of looking at it is to say that Q1 is busy keeping the oscillation going and so can't also work as an output buffer. Any loading of the circuit degrades oscillator performance so it is best to use a separate high-input-impedance buffer amplifier between the oscillator and any load.

Whether or not you use a buffer amplifier, be sure to repeat the amplitude adjusting procedure in the final configuration with the actual load connected. Even connecting a scope probe makes a measurable difference.

The buffer amplifier can be as simple as an emitter follower or source follower. For demanding applications a buffer amplifier

[1]Notes appear on page 20.

with high reverse isolation may be needed so disturbances at its output do not feed back into the oscillator and disturb oscillation purity. High reverse isolation can be provided by one or more grounded-base BJT, grounded-gate FET or dual-gate FET stages. When an oscillator must drive more than one load you may even need a separate buffer amplifier for each load. An example is when you need one or more very clean outputs while also driving a noisy load such as digital logic. Buffer amplifiers deserve their own separate article.

Why it Works

The first part of this article showed you how to easily adjust the amplitude of a BJT oscillator using only dc measurements. The reason it was so easy was because a BJT is so predictable. The nonlinear behavior of a BJT is determined by bias current and the basic physics of the device, and is little affected by manufacturing tolerances. The following explanations use the simplest ideal model of the BJT, which works adequately for oscillators at frequencies up to 50 or 100 MHz. Except for the section on BJT nonlinearity the explanations also apply to field effect transistors (FET) and vacuum triodes or pentodes. FET and tube nonlinearities and biasing are quite different and subject to large sample-to-sample variation.

Figure 5 shows a feedback oscillator consisting of an amplifier and a feedback network connected in a loop. For steady oscillation the loop gain must be exactly one at the frequency at which the phase shift is exactly zero. Loop gain is the gain of the amplifier times the gain of the feedback network. If the loop gain is greater than one, the amplitude will increase, while if the loop gain is less than one, oscillations will die out. So we must find a device whose gain for small inputs is large enough for oscillation to start and for large inputs is small enough to stabilize oscillation amplitude.

The gain of a BJT, an FET, or a vacuum triode is usually specified as a transconductance, g_m, (output current divided by input voltage). If we characterize the feedback network gain by its transimpedance, z_f, (output voltage divided by input current) then when we multiply amplifier gain and feedback network gain to get loop gain, the units cancel so loop gain is a pure number.

A practical oscillator is designed with a loop gain greater than one for small signals so it will start. The oscillation amplitude will rapidly grow. The amplifier will eventually overload or limit enough to reduce the loop gain to one. The amplifier overload behavior controls the oscillation amplitude. The feedback network phase shift controls the oscillation frequency.

Understanding the Circuit

Figure 1 shows the example circuit as it is usually drawn. It doesn't look much like the diagram in Figure 5. Figure 6 shows the example circuit redrawn to more clearly show its function. It has three major parts, the transistor itself, the feedback network, and the dc bias network for the transistor. Notice that it doesn't have a ground symbol. Whether and where it is grounded is an important detail in practical implementations, but has no effect on basic circuit functioning. If you used an FET or vacuum tube instead of a BJT the circuit would look the same, except for a quite different dc bias network.

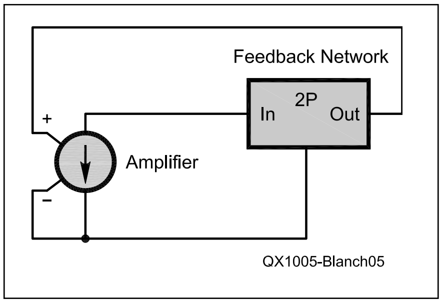


Figure 5 — Feedback oscillator block diagram.

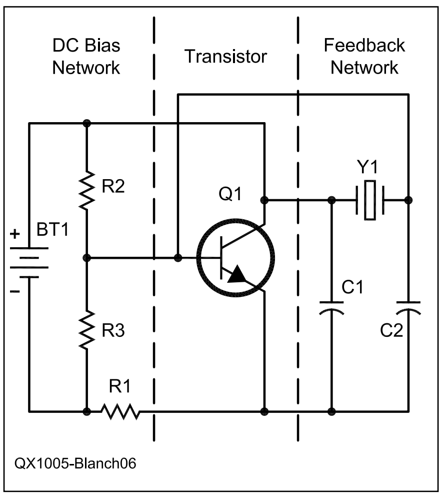


Figure 6 — Example crystal oscillator circuit of Figure 1 redrawn.

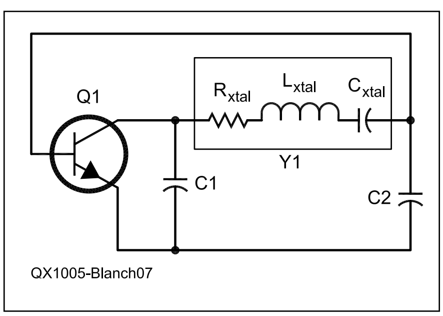


Figure 7 — Example circuit reduced to basics.

The dc bias network is just the customary way of biasing a BJT at a stable dc operating point. The bias current is set by R1 and its value is the voltage across R1 divided by the resistance of R1. For most of the following figures the dc bias network will not be shown. We will assume that the transistor is properly biased.

Figure 7 shows the oscillator circuit stripped to its essentials, a transistor driving a pi network whose output drives the transistor input. The equivalent circuit (instead of just the symbol) of the crystal is shown to remind us that it includes capacitance, inductance and resistance. The oscillator circuit is like a transistor power amplifier with a pi network matching circuit, but with one important difference — the external load is missing. The output of the oscillator pi network drives only the transistor input, which takes negligible power. In a power amplifier the pi network is a matching network designed to maximize power to an external load and minimize power lost in the pi network. In an oscillator, the pi network *contains* the load. The load is the series resistance in the equivalent circuit of the crystal. The other function of a pi network in both a power amplifier and an oscillator is to filter out harmonics in the (very distorted) transistor output current. In both cases, the pi network output is a sine wave with fairly low harmonic content.

The discussion so far has been with a pi network feedback network. So it applies directly to the example Pierce (Colpitts-like) crystal oscillator and Colpitts and Clapp/Gouriet LC oscillators. It also applies to more complex networks, such as the Seiler and Vackar circuits and any other that meet the following conditions. The feedback network must 1) invert the phase of its input voltage and 2) filter out harmonic components present in the BJT output current.

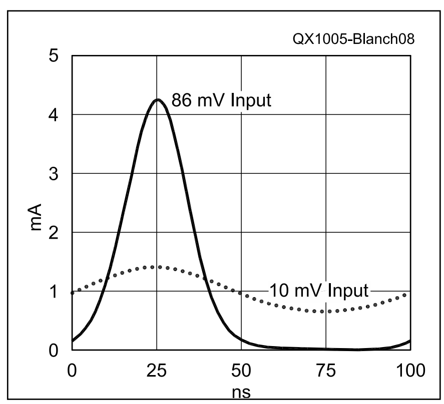


Figure 8 — Transistor collector current simulated waveforms.

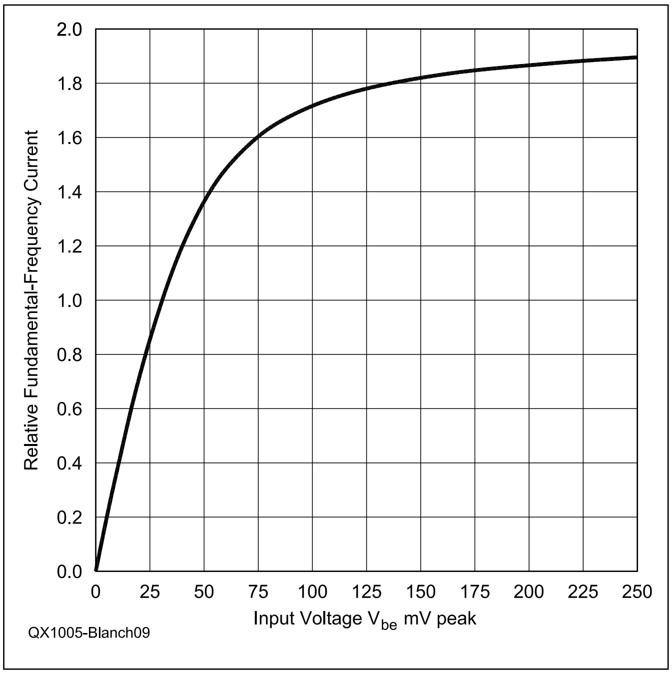


Figure 9 — BJT overload or limiting behavior.

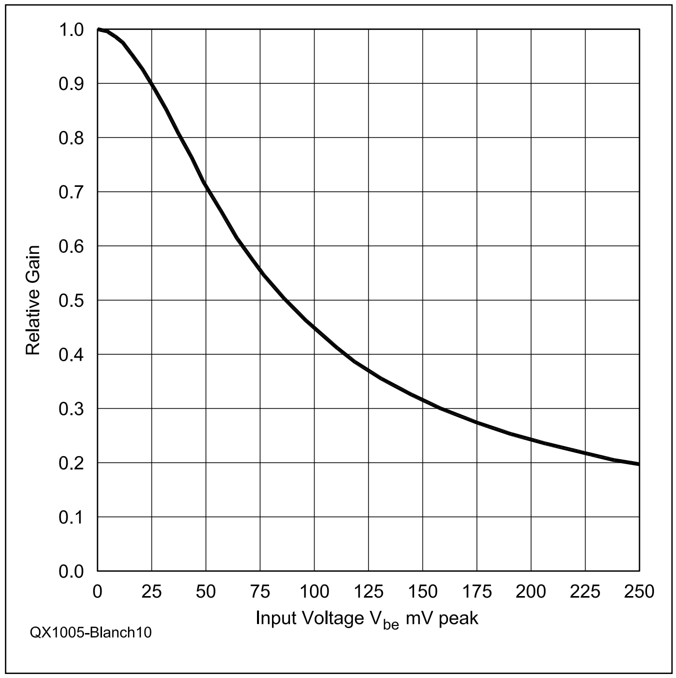


Figure 10 — BJT relative gain versus input voltage.

It needs to invert the phase in order for the feedback to have the correct sense, since the BJT is an inverting amplifier. It needs to filter out the harmonics not because the oscillator wouldn't work but rather because the amplitude prediction curves presented below are valid only when the BJT input voltage is a sine wave. The Hartley circuit does invert the phase but, unless the parts of its inductor are perfectly coupled, doesn't filter out the harmonics. For the Hartley, the bias current setting procedure will work but the resulting amplitude may be in considerable error and so should be verified by measurement.

The above circuit discussion applies equally to oscillators using field-effect transistors or vacuum triodes or pentodes, except for the way the active device is biased and its gain adjusted. Oscillator circuits such as the Butler, with the crystal in series with the BJT emitter, function in a very different way so this analysis doesn't apply to them.

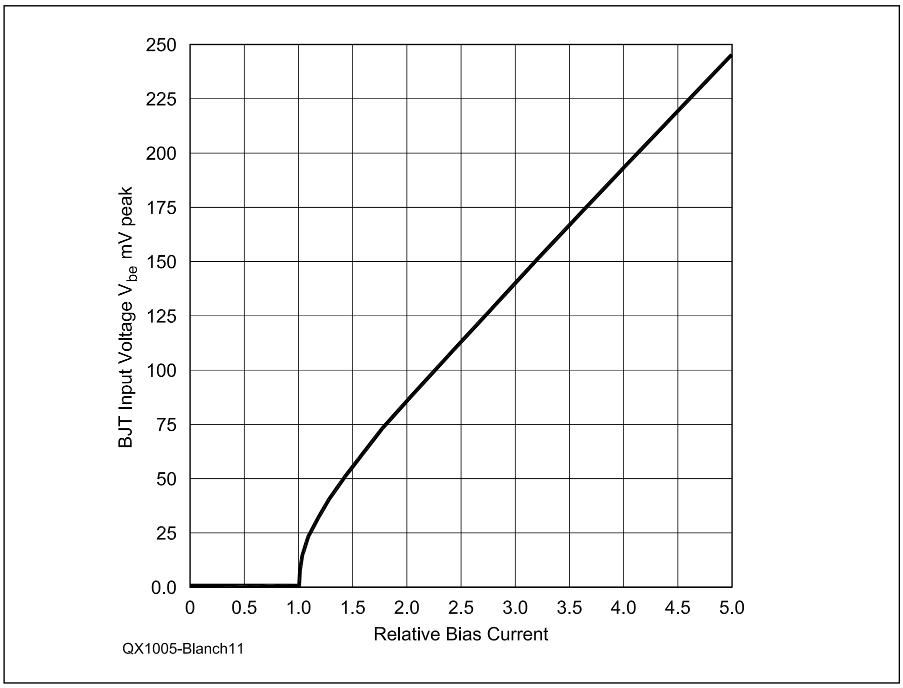


Figure 11 — BJT input voltage vs relative bias current..

BJT Nonlinearity

Amplifier gain and loop gain are linear system parameters. In a linear system the oscillation amplitude would grow to infinity if the loop gain was even the slightest bit greater than one. Stable oscillation depends on the loop gain being exactly one. The "method of equivalent linearization" allows us to pretend that the very nonlinear BJT is a linear amplifier whose gain depends on input amplitude. The method is to drive the BJT input with a sine wave and then measure only the fundamental-frequency component of the BJT collector current. We can ignore the harmonic currents because the feedback network filters them out. The equivalent linear gain of a BJT is the fundamental-frequency component of collector output current divided by the base-emitter input voltage.

A BJT is a voltage-controlled current-output device — a transconductor. The current from emitter to collector is controlled by the voltage between base and emitter. The output current is exponentially related to input voltage and changes by a factor of ten for each 60 mV change in input voltage. Because the current is so sensitive to input voltage, a BJT is almost always current biased as shown in Figure 1 and Figure 6, where the bias current is defined by R1. The input voltage then assumes the value necessary to support that current. The small-signal gain g_m, is directly proportional to the bias current, and is equal to 38 × I_{bias}. For example, in the circuit described earlier, with a bias current of 324 µA has a g_m = 38 × 0.000324 A = 12 milliSiemens (mS). This gain is a transconductance with units of amperes/volt or Siemens. (A Siemens used to be called a mho, for older folks who remember tube transconductance being given in micromhos.)

For input voltage of even a few tens of millivolts, the output current becomes very distorted. Figure 8 shows the output currents for 10 mV and 86 mV peak sine wave input voltages and 1 mA average bias current. For 10 mV, the current is a slightly distorted sine wave. For 86 mV input voltage (corresponding to a relative bias current of two) the current is cut off for half of the cycle and its peak current is four times its average current. The current pulses become still narrower and taller as input voltage increases further.

The overload or limiting behavior of a BJT with constant current bias is shown in Figure 9. The horizontal axis is the amplitude of the sine wave base-emitter input voltage. The vertical axis is the amplitude of the fundamental-frequency component of the collector current relative to the bias current. The curve flattens out at a value of two for large input voltage as the collector current is then only narrow pulses. The fundamental-frequency component of an impulse train is twice the average current.

Figure 10 shows the large-signal linear-equivalent gain compared to small-signal gain versus input voltage. The large-signal gain is reduced to half of the small-signal gain when the input voltage is 86 mV peak. So, for an oscillation amplitude of 86 mV, the small-signal loop gain must be set to two. BJT gain is linearly proportional to bias current so the bias current must be set to twice the current at which oscillation just starts. There isn't anything magical about the choice of 86 mV — relative bias current of two is just a nice round number.

Figure 11 is the Figure 10 data plotted a different way to make it easier to see how oscillation amplitude depends on relative bias current. The values on the horizontal axis of Figure 11 are the reciprocal of those on the vertical axis of Figure 10. The value of 0.5 on the vertical axis of Figure 10 corresponds to the value of 2.0 on the horizontal axis of Figure 11. The vertical axis of Figure 11 is identical to the horizontal axis of Figure 10 and represents oscillation amplitude.

This part of the article explains why the simple adjustment procedure described in the first section works. It showed how to adjust the value of *one* component, R1, in the example circuit. The more general question is how to choose all the *other* component values in an oscillator you are designing from scratch. Answering that question is left for another article.

Acknowledgments

This article is based on concepts from many publications. It is much influenced by Eric Hafner's 1963 paper, which is the clearest explanation I have seen.[2] Hafner summarizes work by and for the Signal Corps after WW II. The Clarke-Hess 1971 textbook treatment of BJT nonlinearities was the other essential reference.[3] Additional resources are listed on the author's Web site.[4]

I much appreciate the help and encouragement from Paul Wade, W1GHZ. He asked good questions and critiqued several drafts. He even did computer simulations, which helped me see the limitations of some of my earlier ways of thinking.

Byron Blanchard, N1EKV, was first licensed as WNØFRR/WØFRR in 1951. An ARRL Member, he holds an S.B in Electrical Engineering from MIT. Byron retired from a career as an electronics engineer doing analog circuit design, system design and embedded microprocessor programming for measuring instruments. His projects mostly involved dc and low frequency precision measurements rather than RF design.

He designed a few oscillators the way most folks do, by copying published circuits. After he retired he decided to learn how oscillators really work. He collected books, articles and test equipment, and performed experiments. After giving a few talks at ham club meetings and conferences, his goal now is to explain the basics to a wider audience.

Notes

[1]The 1 kΩ value for R1 is way too small. Oscillation amplitude was 650 mV peak and waveforms were badly distorted.

[2]Erich Hafner, "Theory of Oscillator Design," *Proceedings of the 17th Annual Frequency Control Symposium*, May 1963, pp. 508-536. This article is available on the Internet at **http://www.ieee-uffc.org/frequency_control/teaching/pdf/s6310508.pdf**

[3]Kenneth K. Clarke and Donald T. Hess, *Communication Circuits: Analysis and Design*, Addison-Wesley, 1971, ISBN 0-201-01040-2. Reprinted 1994 by Krieger Publishing Company, ISBN-13: 9780894648632.

[4]I have posted more information and links about oscillators on my Web page: **http://n1ekv.org/**.

QEX

Mark Spencer, WA8SME, ARRL Education and Technology Program Coordinator

774 Eastside Rd, Coleville, CA 96107; mspencer@arrl.org

Solving Random Noise Issues in TRM-433-LT Data

A number of manufacturers include this tiny transceiver in their UHF data link modules. Here is a way to make it more useful.

I use a number of UHF, Part 15-type, data link modules in ARRL Education and Technology Program projects. These data links are based on Linx RF transmitter, receiver and transceiver ICs that operate on 433 MHz, have an advertised range of 300 feet (optimistic) and are designed to send and receive digital data.

I started a number of years back with using a simple data link to connect a Morse code bug to a transceiver without the interconnecting cables. Since then, I have used these data links in robotics (the Mars Lander simulator) to connect a free-roving robot with students at a ground station in the classroom. I've put them to work in handheld seismometers that students can take on amusement park rides. The links are great for relaying acceleration data to students on the ground. I have also used the data links to receive data collected by Sudden Ionospheric Disturbance (SID) receiver systems that have been detailed in a previous *QEX* article ("SID: Study Cycle 24, Don't Just Use It," Sept/Oct 2008 *QEX*). Lately, I have integrated one of these data links into a CubeSat Simulator to connect a working model of a satellite sending telemetry data to a simulated ground station (Figure 1). This latest project emphasized one limitation of these data links and inspired me to develop a solution.

The Linx series of RF modules use On-Off Keying (OOK) AM modulation to transmit data. Basically, OOK means that when the carrier is present, the receiver output is high to represent the binary state 1 (one), and when the carrier is absent, the receiver output is low to represent the binary state 0 (zero). The early Linx TXM-433-LC/RXM-433-LC transmitter/receiver pair of modules performed well. In an apparent attempt to increase range, the Linx developed the more sensitive TXM-433-LR/RXM-433-LR, but these devices had an irritating flaw: there was no squelch function in the receiver. If no signal was present, the receiver generated a considerable amount of random noise at its output. Software can be written to overlook and negate the random noise, but if you want to send Morse code or simply send control codes to turn a remote switch on and off, the random noise is devastating because it can masquerade as valid data. (The CubeSat Simulator project included sending data via Morse code, just like the real satellites. The random noise made that mode of transmission virtually impossible.)

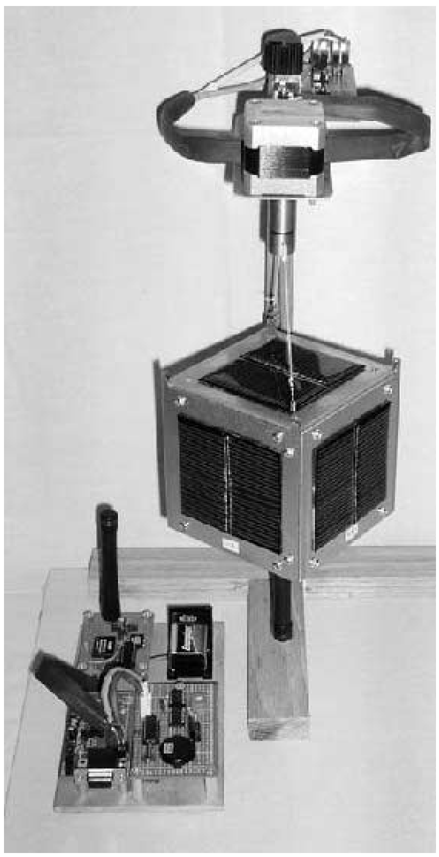


Figure 1 — The CubeSat simulator. A UHF data link connects the satellite to the ground station.

Over time I developed a fairly complex circuit that combined the TX/RX modules along with an antenna changeover device, a squelch circuit, and the gate logic to control all the bits and pieces (Figure 2). I found that it worked fairly well. But as always seems to be the case, just when you get a handle on one technology, the next generation is marketed and it makes the older versions obsolete. Such is the case with the new Linx TRM-433-LT module.

The TRM-433-LT

The TRM-433-LT is a transceiver on a chip that includes virtually all of the features that I had incorporated in my transceiver design at reduced cost (Figure 3)! The random noise issue is still a problem, but there is on-board squelch capability to deal with it. Unfortunately, Parallax, a major manufacturer of RF modules designed around these Linx ICs, ignored the new squelch capability. That meant I had to improvise my own solution, which is detailed in the schematic in Figure 4. Refer to the portion of the schematic that illustrates the TRM-433-LT pinout during the following discussion.

First of all, the TRM-433-LT device is a 12-pin, surface mount device only. The functions of most of the device pins are pretty self explanatory. An external 433 MHz antenna is connected to pin 1, the antenna pin. Voltage (V_{cc} = 3.3 V) is applied to pin 11. The ground pins are pins 2 and 10. The device can be powered down to conserve current through the PDN at pin 9. The TR (transmit/receive) pin is the device PTT; a logic "high" on pin 8 switches the transceiver module into the transmit mode. The DATA line, pin 7, is bidirectional and serves as data in for transmit and data out for receive. The DATA line has a Schmidt triggering circuit that shapes the data output to a square wave. The other pins will require a bit more explanation.

The TRM-433-LT is an FCC Part 15 device, which means that the power output of the transmitter has to be reduced below certain levels to operate in the unlicensed service. The transmitter output power is controlled by the L ADJ at pin 12. The device documentation has a graphic that helps determine a resistance value attached between pin 12 and V_{cc} to control the power. Direct connection of this pin to V_{cc} will produce full power. I chose to use a resistor of 750 Ω between V_{cc} and this pin to bring the power level within Part 15 compliance. The RSSI (pin 4) is the received signal strength indicator line that presents a voltage proportional to the received signal strength. The RSSI line was used in conjunction with the data line to form a squelch circuit in the previous design (Figure 2). Pin 6 outputs the raw received data. The A REF (pin 5) is an added feature of the TRM-433-LT and allows the use of an internal squelch circuit. The threshold level of the squelch is set by varying resistance

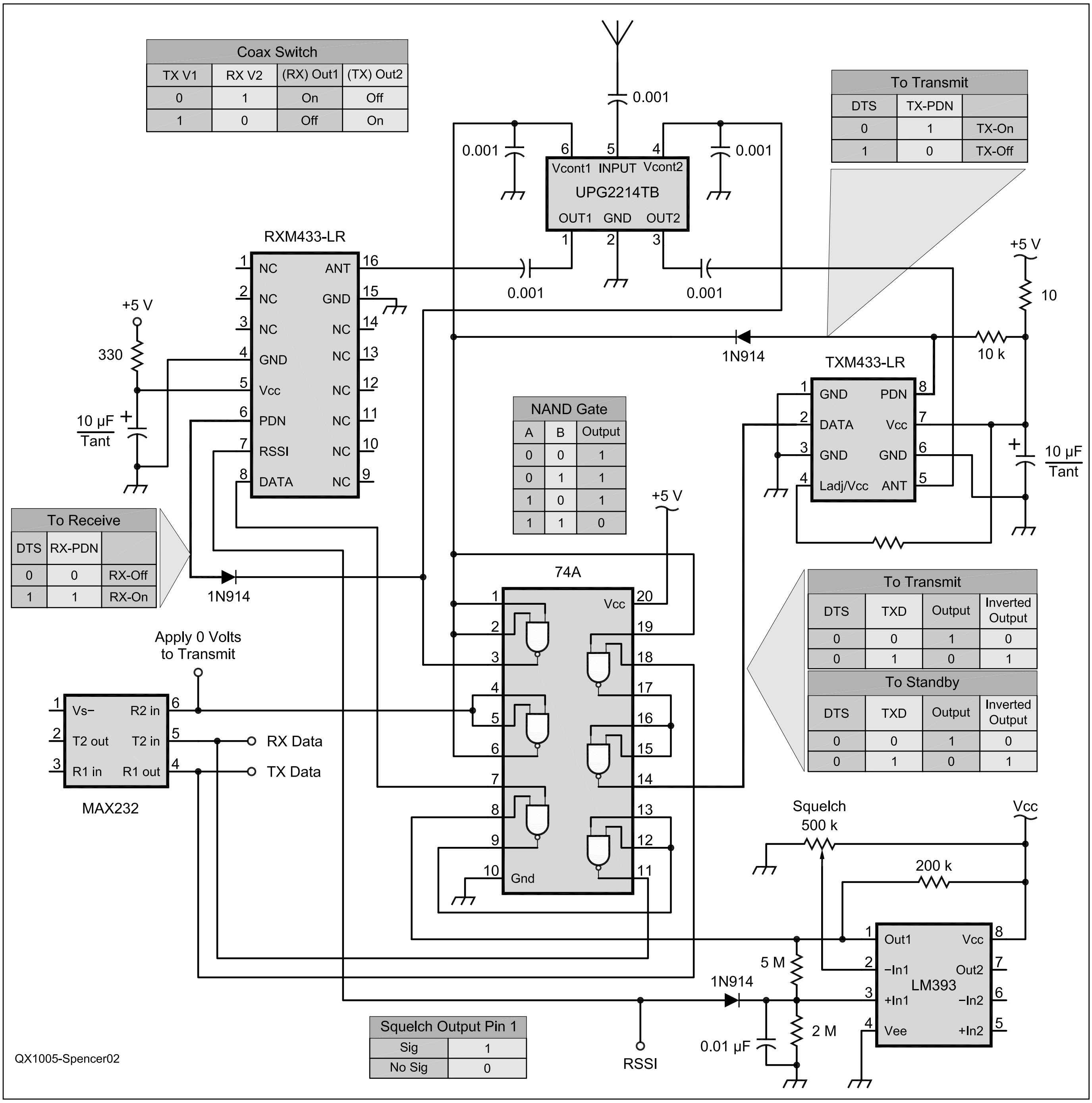


Figure 2 — Data link system circuit diagram using separate transmitter and receiver modules.

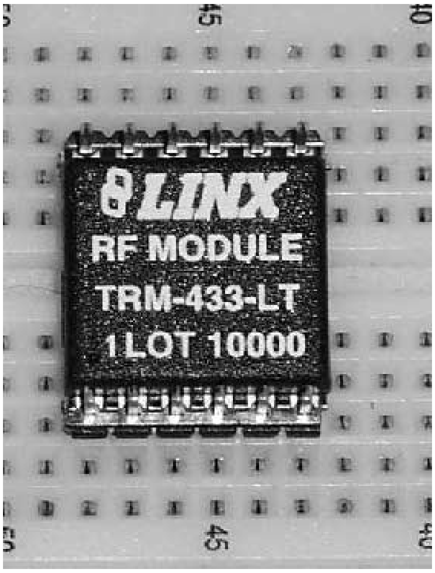


Figure 3 — The Linx TRM-433-LT transceiver module.

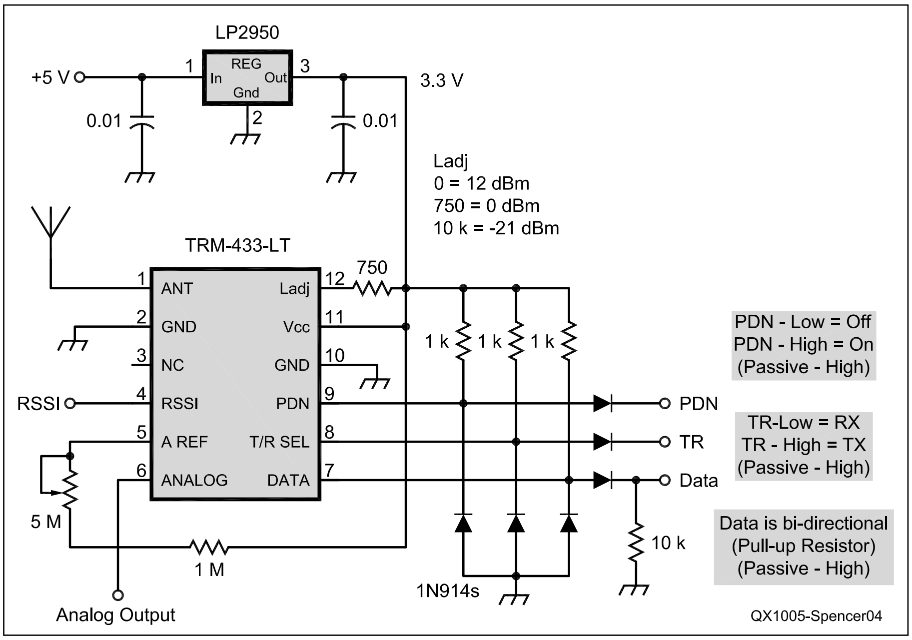


Figure 4 — A TRM-433-LT based data link circuit.

value between this pin and V_{cc}.

The TRM-433-LT requires CMOS level voltages for V_{cc} as well as the other interconnection pins. The 3.3-V regulator steps down the applied voltage to the appropriate level. The 1 kΩ resistors serve as pull-up resistors to keep the associated I/O pins at V_{cc}. The 1N914 diodes attached to the I/O pins serve two purposes. First, the diodes provide some static discharge protection. Second, the diodes provide isolation between non-CMOS level voltages applied to the pins and the TRM-433. For example, in the static state of the TR pin, it is held high. This puts the transceiver in the transmit mode. If a microcontroller were used to control this device, the controlling pin of the microcontroller would apply 5 V to put the transceiver in transmit mode. The in-line diode isolates the 5 V from the module and allows the pull-up resistor to supply the requisite current. Alternatively, if the transceiver is put in the receive mode, the microcontroller would bring the controlling pin low (ground) and the in-line diode would conduct the pull-up resistor current to ground, grounding the module TR pin.

Testing and Tweaking the Squelch Function

The added squelch appears to solve the random noise issue with these devices. However, there are some unspecified quirks that need to be considered. While the squelch will reduce or eliminate random noise on the data line when there is no signal present, the squelch circuit also affects the Schmidt trigger in the data line. Let me illustrate these quirks with some oscilloscope images under various operating conditions.

In the first test, 100-ms-wide pulses were sent over the data link. The oscilloscope image in Figure 5 depicts what happens. The light gray trace, for reference, is the pulse that is generated by the microcontroller and sent by the transmitter. The receive module data line output, with no squelch circuitry or resistor installed, is the Channel 1 trace. Notice that the receiver data line output is high when the transmitted pulse is high. But also notice that about 40 ms after the pulse goes low, random noise on the receiver data line kicks in. If sending Morse code, or controlling switches, the noise could be a problem.

Alternatively, the ANALOG line could be used to mitigate the noise issue. The output of the ANALOG pin follows the raw received signal level without going through the Schmidt trigger circuit in the data line. In Figure 6, the upper trace, Channel 2, shows the action of the ANALOG pin. Note that the ANALOG pin follows the transmitted pulse (the gray reference trace) religiously. One consideration in dealing with the ANALOG pin is that the voltage delta goes from approximately 690 mV when low up to 1.6 V when high (not 0 and 5 V as would be expected). Most comparator or microcontroller devices can distinguish between these two states (voltages). Resistors were added to the squelch circuit to configure the receiver in this test (depicted in the circuit diagram in Figure 4). Note that with the added squelch circuit that the pulse width coming out of the data line pin has been truncated, even though the squelch was opened up to un-squelch the receiver (as indicated by the presence of noise). The manufacturer's documentation alludes to this characteristic this way:

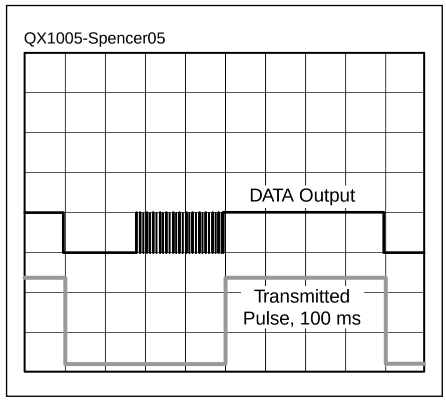


Figure 5 — 100 µs pulses with random noise on receiver data line.

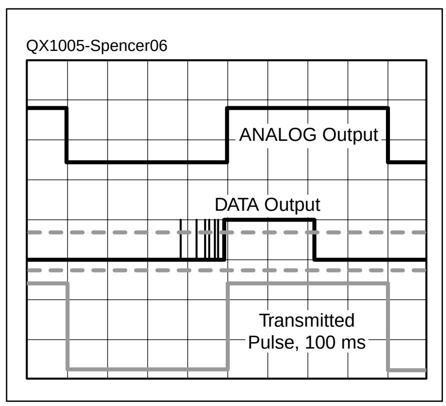


Figure 6 — The transceiver with the squelch circuit active. ANALOG output above, DATA output in the middle and transmitted pulses at the bottom for reference.

"It should also be noted that squelching will cause some bit stretching and contracting, which could affect PWM-based protocols."

In Figure 7, the receiver is squelched. Note that the noise is gone, but the pulse width out of the data line is further truncated while the ANALOG pin faithfully follows the transmitted pulse width.

In Figure 8, serial data was transmitted at 4800 baud. Again, the gray trace is the reference and depicts the data that is being generated for transmission by a microcontroller. The Channel 1 trace is the output of the receiver presented on the data line. Other than a minor delay, the waveform is religiously reproduced. The upper trace is the output of the ANALOG pin. Notice that the output waveform is not shaped. Although it may be usable, it is not as well formed as the Schmidt trigger shaped wave on the data line. In the tests that I did with 9600 baud, the waveform contraction that was mentioned in the documentation started to be evident and could make the use of 9600 baud and above marginal when the receiver is squelched.

The characters were sent in Figure 8 with 5 ms spacing. I stretched the time between characters to 50 ms in Figure 9. Notice that the noise that was evident in the unsquelched configuration is gone, but also notice that the mark state is lost. This data stream has been corrupted due to the loss of the start bit. This problem was also alluded to in the device documentation: "This prevents low amplitude noise from causing the DATA line to switch, reducing the hash during times that the transmitter is off or during transmitter steady-state times which exceed 15 ms." Translating this sentence, it means that if the transmit state is static too long (on or high too long), the Schmidt trigger will shut down and bring the data line to the low state. This characteristic effectively sets the minimum usable baud rate of these modules.

So what's the bottom line? If you are going to send control pulses or Morse-code-like data, use the ANALOG line. If you are going to send serial data, use baud rates between 300 and 4800 baud and the DATA line. Slower baud rates will be affected by the transmitter "steady-state" caveat and baud rates faster will be affected by the bit "contraction" issue.

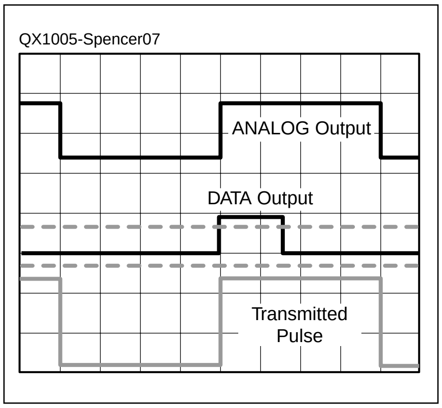


Figure 7 — The squelched transceiver results in truncated DATA pulses.

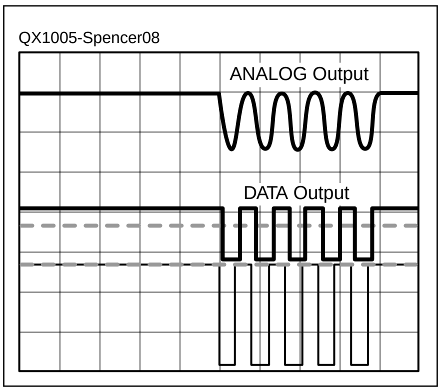


Figure 8 — 4800 baud data. The DATA line is being shaped by the Schmidt trigger. The ANALOG line is raw data.

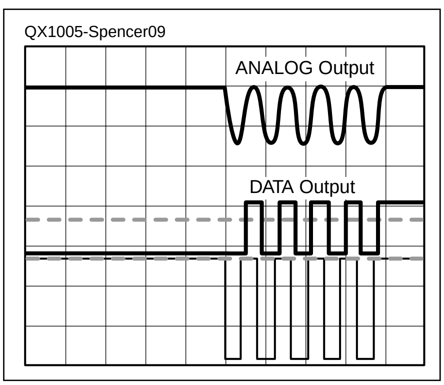


Figure 9 — 4800 baud data with 50 ms between bytes. Notice that the "mark" state is lost in the data line.

Circuit Construction

The data link transceivers were initially built on prototyping boards. Later, formal circuit boards were designed and procured to make circuit duplication easier. I mentioned that the TRM-433-LT is available only as a surface-mount device. It seems that as my eyesight gets less acute, and my grip gets a bit shakier, the components get smaller! At least this device is still large enough that you have a chance to hand wire it. One technique I found effective is to hold header pins in a proto-board and solder the device to the header pins as illustrated in Figure 10 and Figure 3. The rest of the components for the circuit are through-hole components (Figure 11). I use wire wrap wire to make the interconnection between components.

The final circuit board version of the transceiver is shown in Figure 12. It makes a tiny, yet flexible, package.

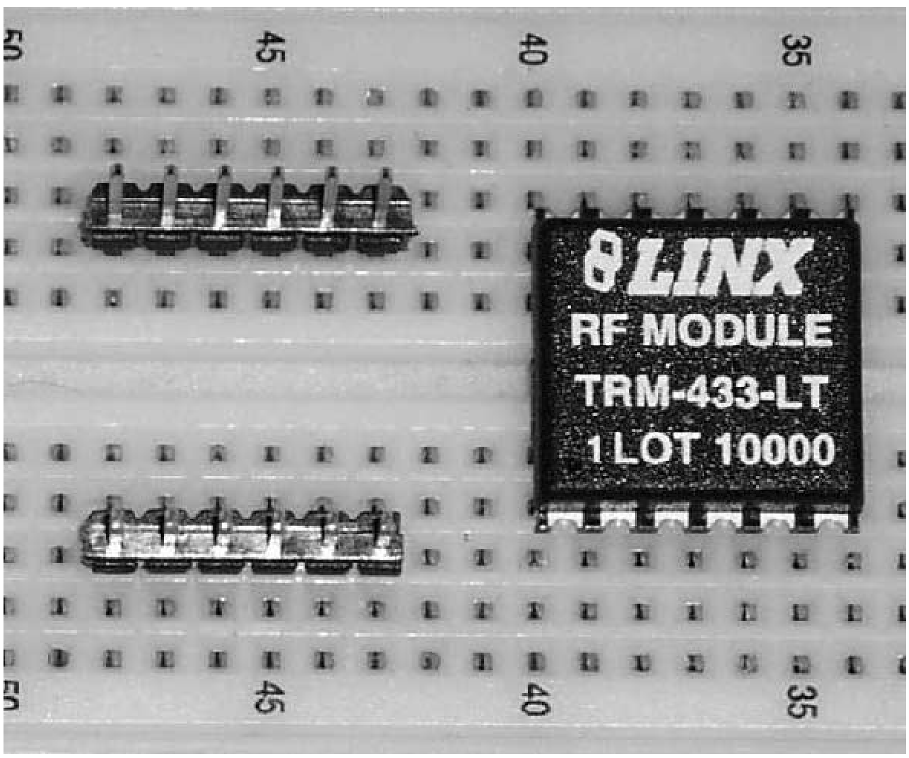


Figure 10 — Use header pins soldered to the SMD device to make it manageable.

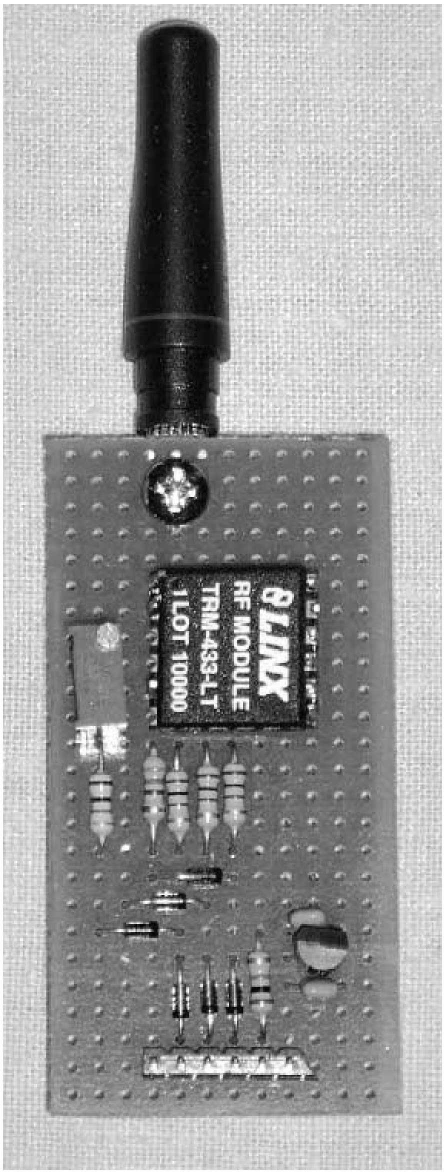


Figure 11 — The hand-wired data link module on a prototyping board.

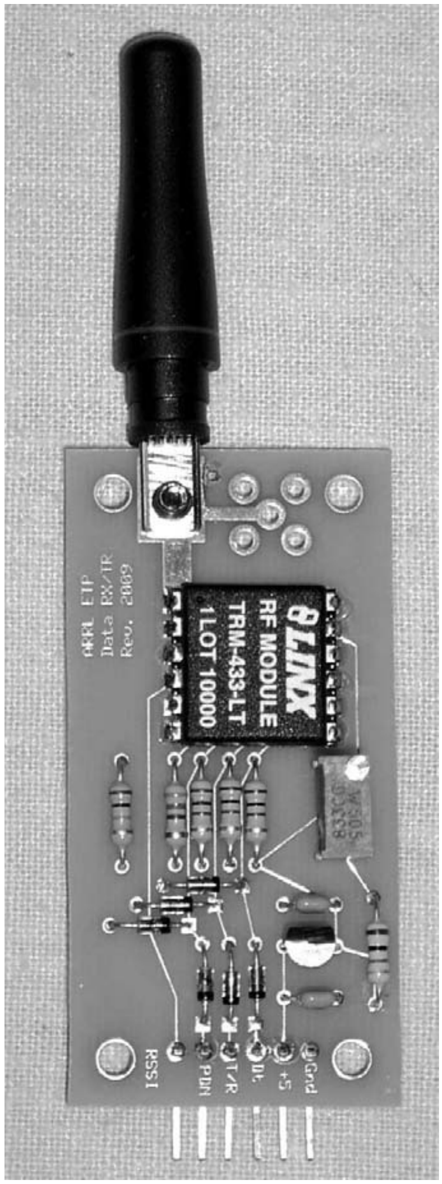


Figure 12 — The formal circuit board module.

Summary

The Linx TRM-433-LT transceiver is easy to adapt to data link applications. With a minimum of ancillary components, you can control the power output of the transmitter section and squelch the receiver noise when no signal is present. There are, however, some trade-offs when squelching the receiver due to the effect the squelch circuit has on the output waveform at the DATA pin. But knowing these limitations, you can easily work around them. Take a look around your shack and I'll bet you will find a number of applications for use these handy modules.

Mark Spencer, WA8SME, is an ARRL member and the ARRL Education and Technology Program Coordinator.

Gary Steinbaugh, PE, AF8L

9534 Appomattox Court, Loveland, OH 45140-7140; gsteinbaugh@yahoo.com

An Inexpensive Laboratory-Quality RF Wattmeter

For less than $100, you can assemble a laboratory-quality RF wattmeter.

This calorimetric wattmeter design has many advantages: it uses inexpensive and readily available components, it can be calibrated with ordinary equipment, and it is capable of better than ±1% accuracy. It does take a long time to make a measurement, and a few calculations are required to obtain a reading, but this will be an acceptable tradeoff for most users.

Calorimeters measure the change in temperature that occurs when thermal energy evolves from another form of energy, in this case, the conversion of radio frequency energy into heat. The beauty of the calorimetric approach is that it is insensitive to frequency, so this wattmeter can be calibrated with easily measured low-frequency line voltage or even a direct current source.

A temperature sensor is added to an oil-filled resistive dummy load, which is enclosed in an insulating container. By measuring the oil temperature at the beginning and end of a measurement and the length of time that the RF energy was applied, the average power in watts can be determined. Although slow, this design avoids the expense and complexity of laboratory wattmeters like the Hewlett-Packard Model 434A, with its bolometer, oil pump, heat exchangers and fan, flow and level meters, etcetera.

I will begin by describing the construction, use, and calibration of this wattmeter; the inquisitive reader may continue to the theory of operation, and on to the derivation of the pertinent equations.

Constructing the Wattmeter

I have an MFJ-250 Versaload, but any similar dummy load, like a Heathkit HN-31 Cantenna, or even a homemade equivalent, may be used. Add a connector for the temperature-sensing thermistor by making a ¼ inch hole (I used a step drill) in the lid midway between the center and the edge.[1] Install a rubber grommet (3⁄16 inch inside diameter), and insert a 0.001 µF feedthrough capacitor (which helps to shunt any voltage that might be induced in the thermistor leads by stray RF). Separate two conductors from about 75 mm of ribbon cable, carefully pull out the strands of wire, and slip the remaining insulation over the thermistor leads. Solder the thermistor leads to the feedthrough capacitor as shown in Figure 1. (The thermistor data sheet cautions: "Use heat sinks when soldering or welding to thermistor leads.")

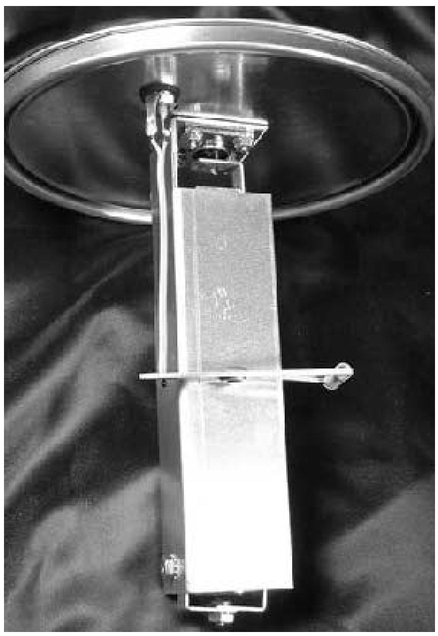


Figure 1 — This photo shows the thermistor that will measure the temperature inside the oil-filled dummy load can. Note the clip-on heat sink used to protect the thermistor while soldering. The dummy load resistor is inside the rectangular metal shield under the paint can lid.

While a dummy load is designed to dissipate heat, a calorimeter must capture heat instead, so a container with high thermal insulation is required. My dummy load (in a one-gallon paint can) fit nicely in a 14 × 11 × 12 inch foam drink cooler.[2] I added seven slabs of 1½ inch thick expanded polystyrene insulation (readily available at home improvement stores), shaped to fit inside the drink cooler as shown in Figure 2. This foam is easily cut with a fine tooth razor saw; with a little care, it can be smoothed nicely with sandpaper. Wrap the dummy load in a thin blanket of fiberglass insulation to discourage surface airflow, and place it in the container as shown in Figure 3. Add oil, close the can lid, and put the drink cooler lid into position.[3]

Using the Wattmeter

There are three steps to making a power measurement: first, record the thermistor resistance, then note the time and apply RF energy. Second, watch the thermistor resistance drop to some predetermined value (I chose 1800 Ω, corresponding to 50°C, for reasons that are explained later), then remove the RF energy and record the time. Third, come back two hours after power application (t_1 = 120 minutes) and record the thermistor resistance.

To obtain the power reading, convert the initial thermistor resistance to a temperature using Equation 2, the Steinhart-Hart equa-

[1]Notes appear on page 32.

Figure 2 — Pieces of foam insulation were cut to fit snugly around the dummy load paint can and fill the inside of a Styrofoam cooler. Note that the logo on the cooler can be one of your choice.

Figure 3 — Here is the paint can and insulation inside the foam cooler. The can is filled with transformer oil before the resistor and thermistor assembly is lowered into the can and the lid closed.

tion, and record this temperature as T_{ON}. Similarly, convert the later thermistor resistance to a temperature and record this as T_1. Call the time (in minutes) when the RF was removed t_{off}, and plug these values into the Equation 1:

$$P \text{ (in watts)} = \frac{\left(T_1 - T_{ON}\right) \mathrm{K} e^{\frac{(t_1 - t_{OFF})}{\tau}}}{60 t_{OFF}}$$

[Eq 1]

The values of constants τ and K are determined by a one-time calibration as described later in the article.

The Steinhart-Hart equation is generally accepted as a mathematical model of a thermistor. Using the parameters for a YSI No. 44034 thermistor, it states that with R (in ohms):

$$T \left(\text{in } ^\circ\text{C}\right) = -273.15 + \frac{1}{1.285 \times 10^{-3} + 2.362 \times 10^{-4} \times \ln\left(R\right) + 9.285 \times 10^{-8} \times \left[\ln\left(R\right)\right]^3}$$

[Eq 2]

It is handy to have this equation stored in a programmable calculator or a spreadsheet.

Example Power Measurement

Measured:
T_{ON} = 22.69°C
t_{OFF} = 7.25 minutes (7 minutes, 15 seconds)
t_1 = 120.0 minutes
T_1 = 56.06°C
Known (see below):
τ = 464 minutes
K = 5126 J/°C
Calculation:

$$P \text{ (in watts)} = \frac{\left(T_1 - T_{ON}\right) \mathrm{K} e^{\frac{(t_1 - t_{OFF})}{\tau}}}{60 t_{OFF}}$$

$$P \text{ (in watts)} = \frac{\left(56.06 - 22.69\right) 5126 e^{\frac{(120.0 - 7.25)}{464}}}{60 \times 7.25}$$

$$P \text{ (in watts)} = \frac{\left(33.37\right) \times 5126 e^{0.243}}{435}$$

$$P \text{ (in watts)} = \frac{171055 \times 1.275}{435}$$

P = 501.37 W

Calibration

The calibration procedure is a one time process. You will never need to repeat it unless you make a change to the wattmeter (addition of oil, new container, or something like that). It is much like making a power measurement, but with an additional temperature measurement.

The most convenient way to calibrate the wattmeter is to use ac line power, adjusted with a variable transformer; my calibration setup is shown in Figure 4. Initially, I planned to calculate power as V^2/R, but experiments revealed that the MFJ-250 load resistance is not constant — it changes with temperature. I made a spreadsheet (see Figure 5) showing combinations of voltage and current that would produce a chosen dissipation (200 W, for reasons that are explained later), moni-

Figure 4 — This photo shows the completed calorimeter along with the various meters and variable transformer that the author used to calibrate the system.

tored the voltage and current, and adjusted the variable transformer to maintain one of these combinations.

To begin, record the thermistor resistance, note the time, and apply 200 W of ac power. While fine-tuning the variable transformer, watch the thermistor resistance drop to 1800 Ω, then turn off the ac power and record the time. Come back two hours (120 minutes) after the time when you turned the power on, and record the thermistor resistance, then return in another hour (180 minutes) and again record the thermistor resistance.

To obtain the time constant, τ, convert the initial thermistor resistance to a temperature and record this temperature as T_{ON}. Convert the two-hour (t_1 = 120 minutes) thermistor resistance and record this as T_1, then convert the three-hour (t_2 = 180 minutes) thermistor resistance and record this as T_2. Plug these values into Equation 3:

$$\tau = \frac{-\left(t_2 - t_1\right)}{\ln\left(\frac{T_2 - T_{ON}}{T_1 - T_{ON}}\right)} \quad \text{[Eq 3]}$$

To obtain the calibration constant, K, call the time (in minutes) when ac power was turned off t_{OFF}, and plug these values into Equation 4:

$$K = \frac{60 \times P \times t_{OFF}}{\left(T_1 - T_{ON}\right) e^{\frac{t_1 - t_{OFF}}{\tau}}} \quad \text{[Eq 4]}$$

Obviously, overall accuracy will be limited by the quality of the calibration. For best results, your calibration should be performed in a temperature-controlled room, out of direct sunlight or other factors that could affect the calorimeter. I am not a metrology expert, but I understand that a calibration standard should be four times as accurate as the instrument being calibrated. Since power is integrated over time, the measurement is an average, and if the settings vary about the desired point, any errors tend to cancel. The τ and K for every wattmeter will be unique, of course, but as a point of reference, my τ measured 464 minutes, and my K measured 5126 J/°C. Here are the sample calculations for my data.

Example Calibration Procedure

Measured:
P = 200 W
T_{ON} = 22.70°C
t_{OFF} = 14.67 minutes

Wattmeter Calibration Data

Power: 200 Watts

Ohms	*Volts = √(watts × ohms)*	*Amps = √(watts / ohms)*
42.8	92.520	2.162
42.9	92.628	2.159
43.0	92.736	2.157
43.1	92.844	2.154
43.2	92.952	2.152
43.3	93.059	2.149
43.4	93.167	2.147
43.5	93.274	2.144
43.6	93.381	2.142
43.7	93.488	2.139
43.8	93.595	2.137
43.9	93.702	2.134
44.0	93.808	2.132
44.1	93.915	2.130
44.2	94.021	2.127
44.3	94.128	2.125
44.4	94.234	2.122
44.5	94.340	2.120
44.6	94.446	2.118
44.7	94.552	2.115
44.8	94.657	2.113
44.9	94.763	2.111
45.0	94.868	2.108
45.1	94.974	2.106
45.2	95.079	2.104
45.3	95.184	2.101
45.4	95.289	2.099
45.5	95.394	2.097
45.6	95.499	2.094
45.7	95.603	2.092
45.8	95.708	2.090
45.9	95.812	2.087
46.0	95.917	2.085
46.1	96.021	2.083
46.2	96.125	2.081
46.3	96.229	2.078
46.4	96.333	2.076
46.5	96.437	2.074
46.6	96.540	2.072
46.7	96.644	2.069
46.8	96.747	2.067
46.9	96.850	2.065
47.0	96.954	2.063
47.1	97.057	2.061
47.2	97.160	2.058
47.3	97.263	2.056
47.4	97.365	2.054
47.5	97.468	2.052
47.6	97.570	2.050
47.7	97.673	2.048
47.8	97.775	2.046
47.9	97.877	2.043
48.0	97.980	2.041

Figure 5 — This chart shows the calibration data for the calorimeter.

$t_1 = 120.0$ minutes
$T_1 = 50.07$°C
$t_2 = 180.0$ minutes
$T_2 = 46.75$°C
Calculate the time constant, τ:
$\tau = -(t_2 - t_1) / \ln((T_2 - T_{ON}) / (T_1 - T_{ON}))$
$\tau = -(180.0 - 120.0) / -\ln((46.75 - 22.70) / (50.07 - 22.70))$
$\tau = -60 / \ln(24.05 / 27.37)$
$\tau = -60 / \ln 0.879$
$\tau = -60 / -0.129$
$\tau = 464$ minutes
Calculate the calibration constant K:
$K = 60\, P\, t_{OFF} / (T_1 - T_{ON})\, e^{(t_1 - t_{OFF}) / \tau}$
$K = 60\, (200)\, 14.67 / (50.07 - 22.70)\, e^{((120 - 14.67) / 464)}$
$K = 12000 \times 14.67 / 27.37\, e^{0.227}$
$K = 176040 / (27.37 \times 1.255)$
$K = 5126$ J/°C

Wattmeter Performance

I made measurements at several power levels, producing the family of curves shown in Figure 6. Using τ and K calculated from the 200 W measurement, these readings resulted in the data shown in Table 1. Accuracy begins to suffer below 50 W for reasons that will be discussed below.

Making Choices

Accurate direct-reading temperature sensors (thermometers) are convenient but expensive, so I used a thermistor, which is a temperature-sensitive resistor that exhibits a steep drop in resistance as temperature increases. I chose a YSI 44034 thermistor with a resistance of 5000 Ω at 25°C, because my Fluke 111 digital multimeter has a resolution of four digits below 6000 Ω.

For better temperature accuracy, it is desirable to produce a large temperature excursion, but this increases measurement times, and I did not want to risk overheating the calorimeter, so I chose to turn off power at 1800 Ω (corresponding to 50°C) as a tradeoff. Of course, the oil temperature should be returned to ambient before making another reading.

The calculated results will be most accurate near the power level used for calibration, with errors increasing for higher or lower power. I wanted to make measurements in the range of 50 W to 1000 W; 224 W is their geometric average, so I rounded this to 200 W to calibrate and measure τ and K for my wattmeter.

It is desirable to minimize the long measurement times, but all thermal transients must have settled out before taking a reading, and 2 hours (120 minutes) seems to be a good compromise.

Table 1
Wattmeter Measurements

Applied Power, W	*Calculated Power, W*	*Relative Error, %*
500	501.37	+ 0.27
200	200.00	0.00
100	99.64	– 0.36
50	49.76	– 0.48
20	18.61	– 6.95

Theory of Operation and Its Verification

The principle of conservation of energy dictates that electrical energy applied to the load resistor must be converted to an equivalent quantity of thermal energy, or heat. As heat from the load resistor is added to the oil, its temperature increases. Now, a perfect calorimeter would respond as shown in the upper curve of Figure 7, and the only measurement required would be ΔT. Because of time delays, convective currents, and loss of heat through the container walls, however, a practical calorimeter responds as shown in the lower curves of Figure 7. At first glance, it would seem impossible to determine ΔT, but by knowing the calorimeter time constant τ and its calibration constant K, ΔT may actually be reverse-calculated. Mathematical physics tells us that the resulting exponential decay curves, once the initial transients die out, must all have the same shape (as proven in the Appendix). Thus, if we scale the decay curve with τ and K, ΔT may be accurately extrapolated.

Verification tests comparing power sources at 2 MHz, 60 Hz, and dc (see the setup in Figure 8) confirmed that the wattmeter is frequency insensitive. Both a Bird wattmeter and an oscilloscope were used to measure RF power (note that RMS values must be used when comparing to dc, not peak or peak-to-peak values).

To calibrate the wattmeter, four variables must be measured:

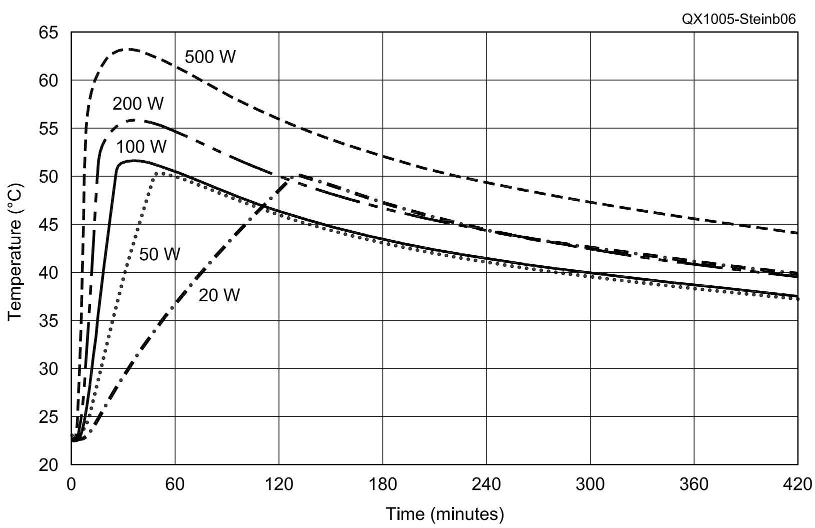


Figure 6 — This family of curves shows the temperature versus time for various amounts of input power.

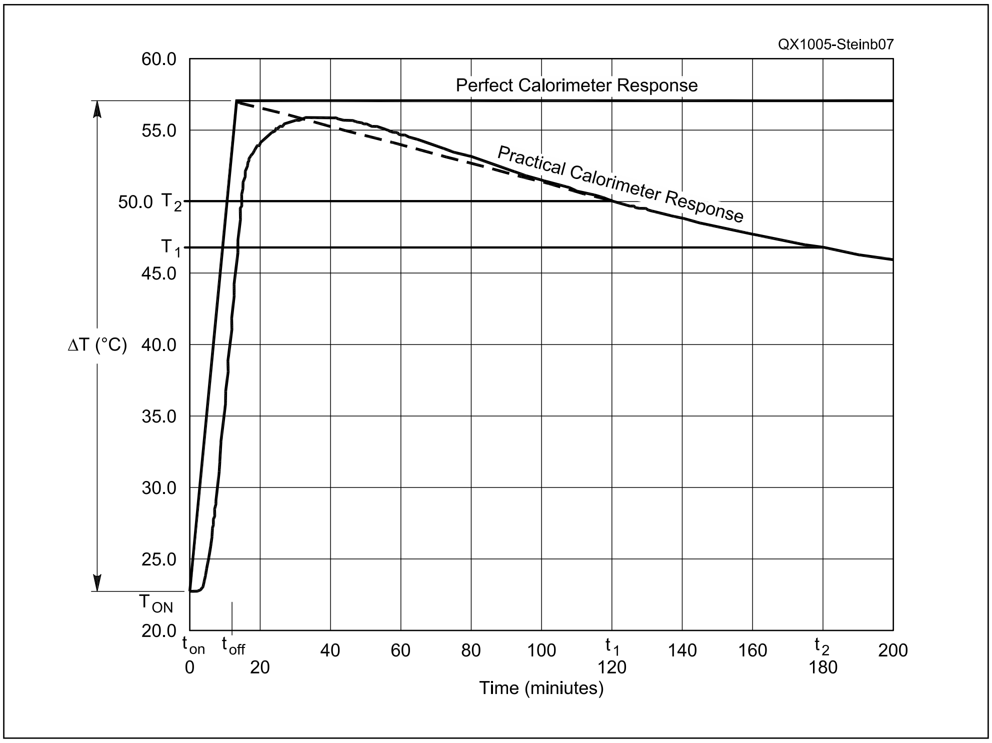


Figure 7 — This graph plots the temperature versus time for a perfect calorimeter and for a the practical calorimeter made for the wattmeter.

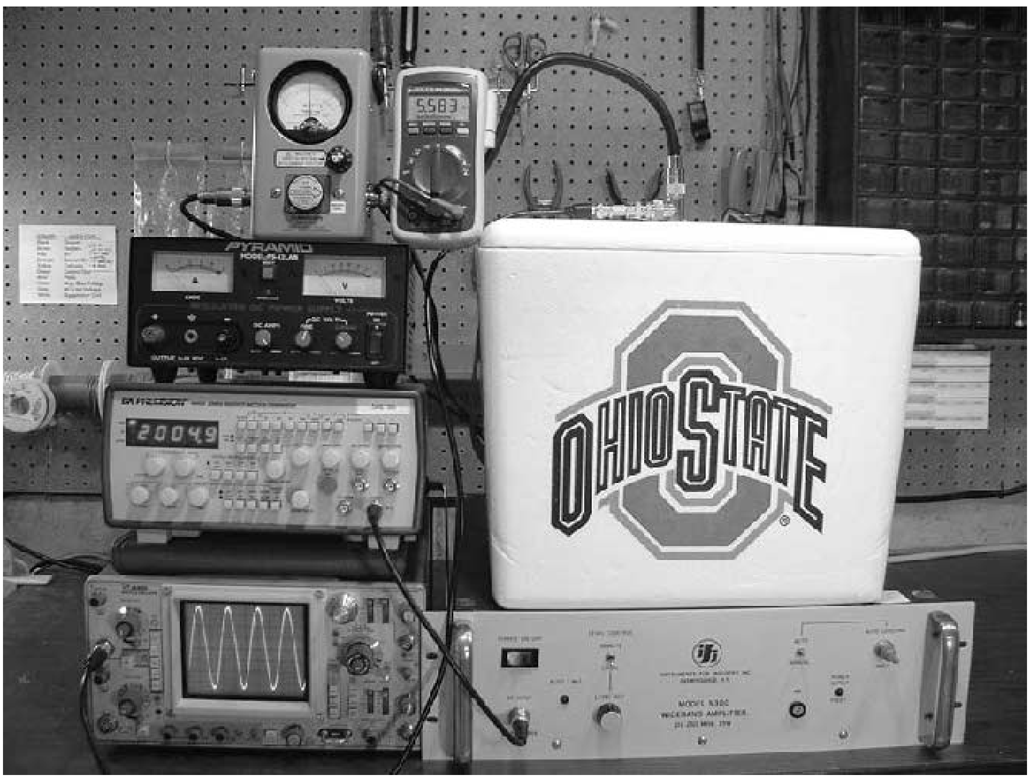


Figure 8 — This photo shows the equipment set-up used to verify the wattmeter operation and accuracy.

time, temperature, voltage, and current. Time is the easiest of these to measure accurately; a stopwatch with a seconds display should be adequate.

Measuring temperature to a fraction of a degree Celsius is not a trivial task. The YSI 44034 is interchangeable to ±0.1°C, and since only temperature *differences* are significant, I estimate my differential readings to be better than ±0.03°C.

Because of the load resistance changes mentioned earlier, power was measured as the product of voltage and current: voltage was measured with a Fluke 179 digital multimeter, and current was measured with a Wavetek Meterman 27XT digital multimeter (a better ac ammeter would have been desirable). My variable transformer appears to exhibit some graininess and thus is not infinitely variable, but the power measurement is averaged, and errors tend to cancel as the transformer is adjusted around the desired point.

Errors increase for power inputs less than 50 W because it takes so long for the oil temperature to reach 50°C that a significant amount of heat escapes through the container walls, resulting in low readings. For QRP use, I might suggest assembling a second, faster-reacting calorimeter using a smaller can (for example, one quart or even one pint) and an appropriately sized noninductive load resistor. This calorimeter should be wrapped in an insulating blanket that fills up the existing cavity in the container.

A final thought: if a faster wattmeter is available, this wattmeter might be used as a calibration standard.

Derivation of Equations

Notation (Refer to Figure 7)

ΔT is extrapolated temperature increase in °C at power off
τ is time constant in minutes
t_{ON} is time at power on ($\equiv 0$)
T_{ON} is temperature at power on, in °C (ambient temperature)
t_{OFF} is time at power off in minutes
t_1 is time at first reading in minutes
T_1 is temperature at first reading in °C
t_2 is time at second reading in minutes
T_2 is temperature at second reading in °C
P is power in watts (W)
q is added heat in joules (J)
C_V is specific heat at constant volume (J/(g ×°C))
m is mass in grams (g)
K is calibration constant ($\equiv C_V \times m$) in joules per °C (J/°C)

Calculating τ

From physics:

$$T_2 = T_{ON} + \left(T_1 - T_{ON}\right)e^{\frac{-(t_2-t_1)}{\tau}} \qquad \text{[Eq 5]}$$

Rearranging Equation 5,

$$T_2 - T_{ON} = \left(T_1 - T_{ON}\right)e^{\frac{-(t_2-t_1)}{\tau}} \qquad \text{[Eq 6]}$$

$$\frac{\left(T_2 - T_{ON}\right)}{\left(T_1 - T_{ON}\right)} = e^{\frac{-(t_2-t_1)}{\tau}} \qquad \text{[Eq 7]}$$

$$\ln\left(\frac{T_2 - T_{ON}}{T_1 - T_{ON}}\right) = \frac{-\left(t_2 - t_1\right)}{\tau} \qquad \text{[Eq 8]}$$

$$\tau = \frac{-\left(t_2 - t_1\right)}{\ln\left(\frac{T_2 - T_{ON}}{T_1 - T_{ON}}\right)} \qquad \text{[Eq 9]}$$

(Note: Equation 9 is the same as Equation 3.)

Calculating K

From physics:

$$\Delta T = (T_1 - T_{ON})\, e^{(t_1 - t_{OFF})/\tau} \qquad \text{[Eq 10]}$$

and

$$q = C_V\, m\, \Delta T \qquad \text{[Eq 11]}$$

Substituting K into Equation 11 (note that this eliminates having to measure C_v and m),

$$q = K\, \Delta T \qquad \text{[Eq 12]}$$

Rearranging Equation 12,

$$K = q / \Delta T \qquad \text{[Eq 13]}$$

Also,

$$q = \int_0^{t_{OFF}} P\,dt \qquad \text{[Eq 14]}$$

Solving Equation 14 for constant P:

$$q = P\,(t_{OFF} \times 60 \text{ seconds / minute}) \qquad \text{[Eq 15]}$$

Substituting Equations 10 and 15 into Equation 13:

$$K = \frac{60 \times P \times t_{OFF}}{\left(T_1 - T_{ON}\right)e^{\frac{t_1 - t_{OFF}}{\tau}}} \qquad \text{[Eq 16]}$$

(Note: Equation 16 is the same as Equation 4.)

Calculating P

Rearranging Equation 16:

$$P \text{ (in watts)} = \frac{\left(T_1 - T_{ON}\right)K e^{\frac{(t_1 - t_{OFF})}{\tau}}}{60 t_{OFF}} \qquad \text{[Eq 17]}$$

(Note: Equation 17 is the same as Equation 1.)

Appendix: A Little Mathematical Physics

Isaac Newton determined that the rate at which the temperature of a mass changes is proportional to the difference between its own temperature and the surrounding (ambient) temperature. Stated as an equation:

$$\frac{d}{dt}T(t) = r\left(T(t) - T_a\right) \qquad \text{[Eq A-1]}$$

To solve this differential equation, first rearrange it as:

$$\frac{dT(t)}{T(t) - T_a} = r\ dt$$

then integrate both sides:

$$\int \frac{dT(t)}{T(t) - T_a} = \int r\ dt \qquad \text{[Eq A-2]}$$

$$\ln(T(t) - T_a) + C_1 = r\,t + C_2 \qquad \text{[Eq A-3]}$$

Combine the constants of integration:

$$\ln(T(t) - T_a) = r\,t + C_2 - C_1 = r\,t + C_3 \qquad \text{[Eq A-4]}$$

Exponentiate:

$$e^{\ln(T(t) - Ta)} = e^{rt+C3} = e^{rt}\,e^{C3} = e^{rt}\,C_4 \qquad \text{[Eq A-5]}$$

$$T(t) - T_a = C_4\,e^{rt} \qquad \text{[Eq A-6]}$$

Since e is a constant, the shape of the curve will be the same for all heating or cooling masses.

Incorporate the initial conditions:

At $t = 0$: $T(0) > T_a$ (the mass is cooling);

$$T(0) - T_a = C_4\,e^0 = C_4 \qquad \text{[Eq A-7]}$$

Therefore:

$$T(t) - T_a = (T(0) - T_a)\,e^{rt} \qquad \text{[Eq A-8]}$$

Rearrange:

$$T(t) = T_a + (T(0) - T_a)\,e^{rt} \qquad \text{[Eq A-9]}$$

It is customary to replace r with the time constant $\tau = -1 / r$ (negative since $T(0) > T_a$), so:

$$T(t) = T_a + (T(0) - T_a)\,e^{-t/\tau} \qquad \text{[Eq A-10]}$$

Notes

[1]Omega Engineering (**www.omega.com**) is a source for accurate thermistors.

[2]Wattmeter operation appears to be unaffected by the logo available on the drink cooler.

[3]Underfilling can result in a burned-out resistive load, while overfilling can result in hot oil surging out of the pressure relief. Inside my dummy load can are two metal disks, part of the wire bail attachments; I added oil to just submerge these disks.

Continuously licensed since 1964, Gary Steinbaugh, AF8L, is an ARRL Life Member. Holding a BSEE from Case Institute of Technology and several patents, he is a licensed Professional Engineer, and the author of many technical articles. He is Senior Electronic Engineer for AtriCure, Inc., a manufacturer of RF electrosurgical instruments used in the treatment of soft tissue ablation. Gary is also a Certified Flight Instructor and a semi-pro musician. He may be reached at **gsteinbaugh@yahoo.com**.

Larry Wolfgang, WR1B

QEX Editor; qex@arrl.org

Reader's Page

Hello Larry,

Here are two photos of my "Controlled Noise Source." This project combines the calibrated noise source described by William Sabin, WØIYH, (in the May 1994 issue of *QST*, and in the Technical Correspondence column in the January 2008 issue of *QST*) and a voltage controlled attenuator described in Agilent Technologies Applications Note 1048. This attenuator comprises 4 PIN diodes in a pi network configuration and uses surface mount devices. Various levels of attenuation are achieved by applying preset dc voltages selected by a rotary switch. A relay is included to bypass the noise source, making the attenuator accessible for other uses. Photo 1 shows the operating controls and Photo 2 shows a view inside the project case.

The noise source is a NoiseCom NC302L diode specifically designed for this purpose but unfortunately not within the budget of most hams. I have a few available at $10. A satisfactory alternative is a 1N5235B small signal Zener, available for less than a $1 from Mouser Electronics (**www.mouser.com**).

— 73, Ron Skelton, W6WO, 4221 Gull Cove Way, Capitola, CA 95010; **w6wo@k6bj.org**

Photo 1 — Ron Skelton, W6WO, built this "Controlled Noise Source," with a calibrated noise source and switched PIN diode attenuator.

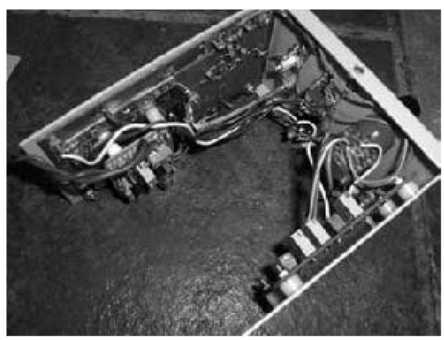


Photo 2 — Here is an internal view of Ron's project.

Jon Wallace
111 Birden St, Torrington, CT 06790; fjwallace@snet.net

Richard Flagg, AH6NM
1721-1 Young St, Honolulu, HI 96826; rf@hawaii.rr.com

Amateur Radio Astronomy Projects — Radio Signals from Jupiter

You can receive signals directly from the planet Jupiter, possibly by using equipment you already own.

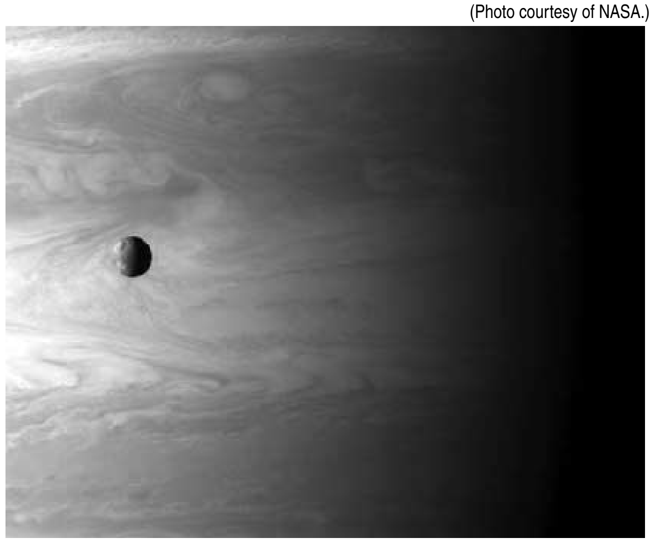
In 1955 mysterious signals from space were discovered by radio astronomers at the Carnegie Institution of Washington, DC. Some thought the signals were local interference, perhaps a noisy ignition system of a pickup truck whose driver was returning home from a late night date. However, analysis revealed that the planet Jupiter was in the beam of the Mills Cross antenna each time that signals were heard. Unlike many radio astronomy dish antennas, the huge Mills Cross comprised over 100 dipoles strung between wooden poles planted in a Maryland field. The dipoles were phased to produce a narrow, steerable, pencil-thin beam some 2.5º in width. That is an amazingly narrow beam considering the operating frequency was 22.2 MHz. Ever since this accidental discovery, researchers have aimed shortwave antennas at Jupiter as they attempted to understand the source of these powerful signals.

Giant Jupiter, some 500 million miles from Earth, is a huge ball of gas with a small core of solid hydrogen, a strong magnetic field, and a retinue of over 60 moons. The largest moons, Io, Europa, Ganymede and Callisto, were first viewed by Galileo over 400 years ago using his primitive telescope. Large enough to hold 1000 Earths, Jupiter rotates about its axis every 10 hours — a rotating speed demon compared to tiny Earth and its 24-hour day.

Signals Detected

The so-called *decametric* radio signals from Jupiter are not on the air all the time but seem to be linked to three longitude regions around the planet, cleverly named the A, B, and C source regions. If one of these source regions is facing Earth, we have an increased probability of receiving signals. If the Jovian moon Io is in the right place in its orbit, the probability of receiving signals is greatly enhanced. The moon Io happens to be within the tidal force's limit of Jupiter and it is literally being torn apart by gravitational forces with tides as large as 100 meters (about 300 feet!). Io crosses the magnetic field of Jupiter and is thus able to release charged particles into the field. These charges are accelerated to very high speed and spiral along magnetic field lines and generate synchrotron radiation, which manifests itself as the radio signals detected here on Earth. There is additional data that suggests that Ganymede and Europa may also contribute to the radio emissions. Earth's

(Photo courtesy of NASA.)



Jupiter and Io.

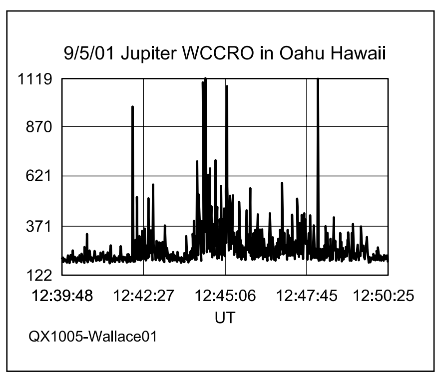


Figure 1 — Jupiter radio noise bursts received at 20.1 MHz using a Jove receiver and dual dipole antenna. Signals are displayed using *SkyPipe* software. The vertical axis is proportional to voltage.

Figure 2 — The Jove RJ1.1 direct conversion receiver is simplicity itself with only two controls – a POWER SWITCH/VOLUME CONTROL and a TUNING knob. Low level audio from the receiver is sent to a computer sound card and also may drive an amplified speaker. The receiver uses a J310 grounded gate RF amplifier followed by the traditional NE602 mixer/oscillator and high gain audio amplifiers. A varicap tunes the receiver over a 400 kHz range centered on 20.1 MHz.

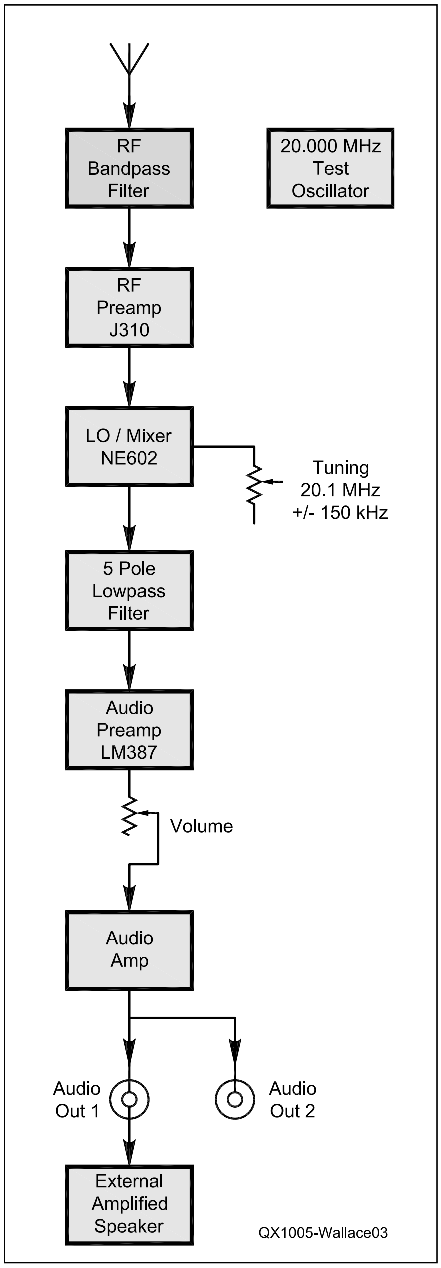


Figure 3 — Here is a block diagram of the Radio Jove receiver.

ionosphere limits ground based reception below about 15 MHz and Jupiter itself does not emit these signals above 39.5 MHz — an upper limit determined by the strength of the Jovian magnetic field.

So what do the signals sound like? There are two distinct types: *L-bursts* sound like ocean waves breaking up on a beach, and *S-bursts*, which can occur at rates of tens of bursts per second, sound like popcorn popping or a handful of gravel thrown onto a tin roof.

Have you heard them? Late at night is the best time, when the ionosphere has become transparent and most terrestrial signals have disappeared on the 15 meter band. The quiet hiss in your headphones comes mostly from relativistic electrons spiraling in the galactic magnetic field. L-bursts and S-bursts are heard above this background noise. A radio noise storm of L- or S-bursts can last from a few minutes to a couple of hours (Figure 1).

Do you need a giant antenna spread out over several acres? Fortunately not — a ham band Yagi will do very nicely. Even if Jupiter is 30° or 40° above your horizon, a low-mounted Yagi aimed toward the azimuth of Jupiter will probably have adequate gain. And you don't need a cryogenically-cooled front end either; your favorite ham-band receiver is plenty sensitive. Just be sure to turn the AGC off, as AGC can severely distort the Jovian noise bursts. Probably the best frequency range is between 18 and 22 MHz, so if you are using a ham-band only receiver, try the 15 or 17 meter bands. Either AM or SSB modes will work. Just tune for a quiet spot between the stations.

During a good storm, Jovian signals can be easily heard, often several dB above the background noise. Of course, the bigger your antenna the stronger the signals. The 640-dipole, 26.3 MHz, phased array antenna at the University of Florida would yield signals well over 20 dB above the background.

Radio Jove

Ten years ago a group of (mostly) University of Florida graduates working at NASA conceived an educational outreach program known as Radio Jove. The idea was to build an inexpensive radio telescope kit suitable for detecting signals from Jupiter. The Jove receiver (Figure 2) is a simple direct-conversion design operating over a few hundred kilohertz range centered at 20.1 MHz. The block diagram is shown in

Sources for Jupiter Radio Supplies and Information

Radio Jove Web site: **http://radiojove.gsfc.nasa.gov/**

Jim Sky's Radio Jupiter Central site: **www.radiosky.com/rjcentral.html**

Jim Sky's Radio Sun Central site: **www.radiosky.com/suncentral.html**

The Society of Amateur Radio Astronomy (SARA): **www.radio-astronomy.org/**

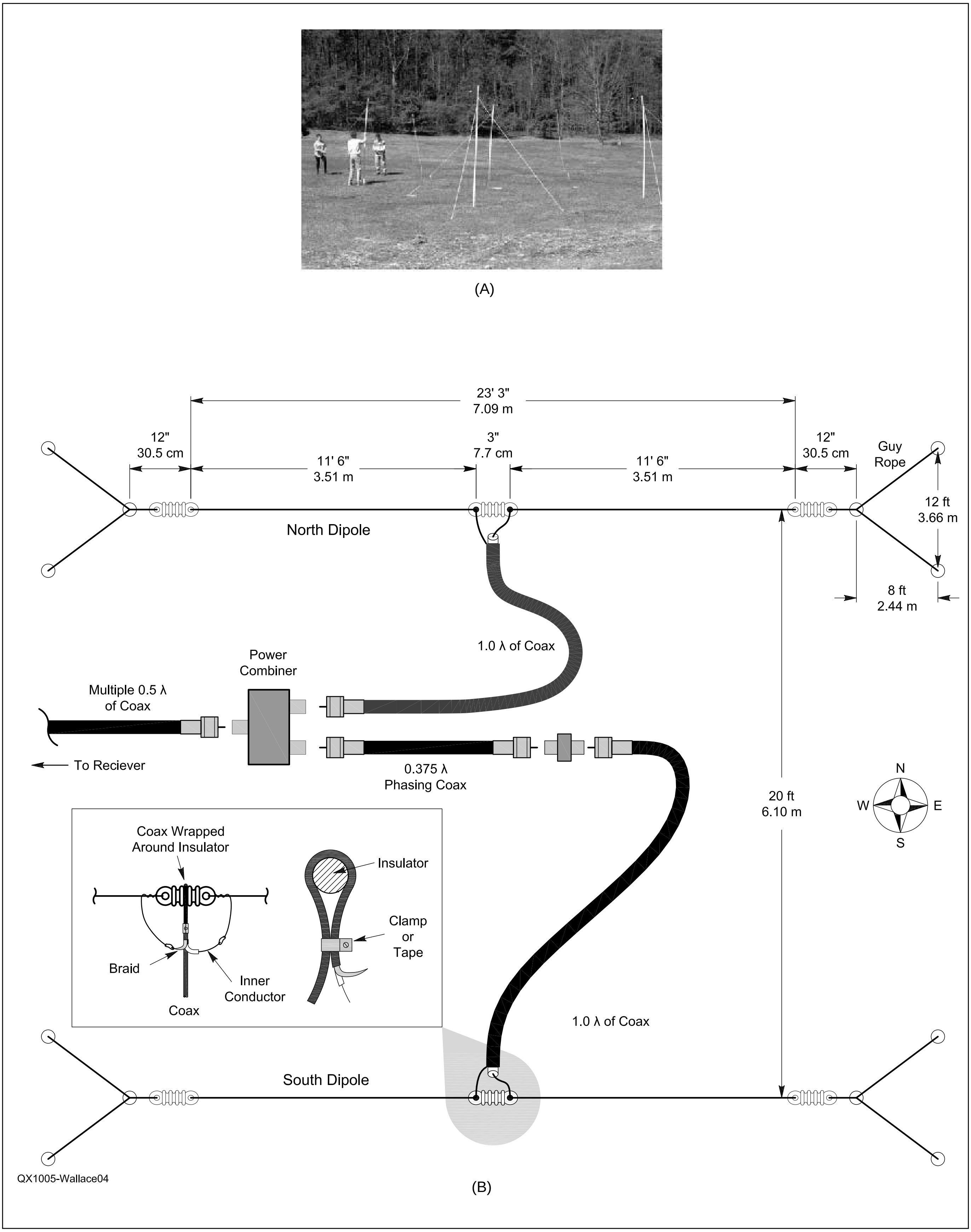


Figure 4 — The Jove dual dipole array. Dipoles are suspended between PVC masts. Signals from the dipoles go to a power combiner and then to the receiver.

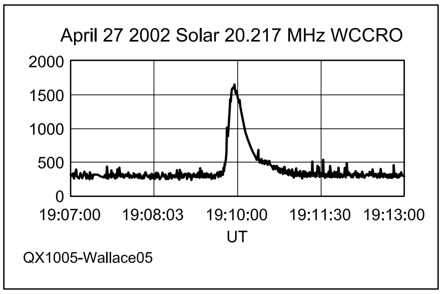


Figure 5 — A single shark fin shaped solar burst. Activity has been sparse during the long solar minimum but as we head toward maximum there will be many opportunities to hear solar bursts. On occasion, following a strong burst the background noise will decrease, indicating increased absorption in the earth's ionosphere due to enhanced solar X-ray and UV flux.

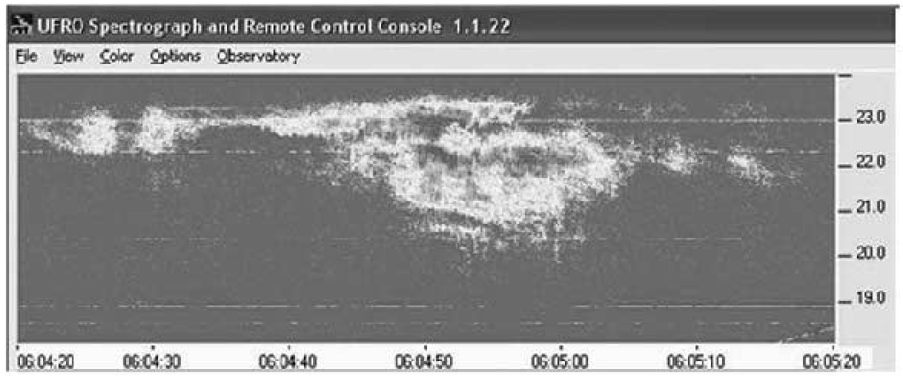


Figure 6 — Radio spectrogram of Jupiter radio noise storm activity. Horizontal lines are weak stations while Jovian signals are seen as the enhanced region between 20.2 and 23.5 MHz. Any resemblance between this emission region and the shape of the starship *Enterprise* is purely coincidental.

Figure 3. The antenna is a dual dipole array (Figure 4). Audio signals from the receiver are sent to a computer sound card where they are processed and displayed as a strip-chart record using *SkyPipe* software. In addition to simply displaying signal strength, *SkyPipe* will stream your data over the Internet to other observers who can then display your results in real time on their computer screens.

Radio Jupiter Pro (RJP) software carries out the fairly complex calculations necessary to predict when a radio noise storm is likely, taking into account the longitude of Jupiter facing the Earth, the position of Io in its orbit, and where Jupiter is in your local sky.

Many observers are also using the Jove equipment to monitor radio noise bursts from the Sun. Some solar bursts can be very strong (over 25 dB above the background) and can easily be received with a single dipole. Individual solar bursts often start abruptly, and then trail off in intensity over tens of seconds (Figure 5). On a strip-chart record they sometimes look like a shark fin. Since you will be receiving during the daytime, the ionosphere can be an issue (as discussed in previous articles) and you need to tune between stations to avoid false signals.

Solar radio bursts are classified as follows (from **www.radiosky.com/suncentral.html**):

Type I Short, narrow band events that usually occur in great numbers together with a broader band continuum. May last for hours or days.

Type II Slow drift from high to low frequencies. Often show fundamental and second harmonic frequency structure.

Type III Rapidly drift from high to low frequencies. May exhibit harmonics. Often accompany the flash phase of large flares.

Type IV Flare-related broad-band continua.

Type V Broad-band continua that may appear with Type III bursts. Last 1 to 2 minutes, with duration increasing as frequency decreases.

The Jove program operates two radio spectrographs. These instruments normally sweep through 200 channels between 18 and 28 MHz and produce a visual display of signal strength at different frequencies. Spectrograms are streamed to observers in real-time and are useful in seeing what frequencies are currently active (**http://radiojove.gsfc.nasa.gov/software/index.html**). Jupiter noise storms often drift up and down the spectrum, and the spectrograms help observers confirm their signals at 20.1 MHz (Figure 6).

To date over 1400 Radio Jove kits have been sold to enthusiastic observers all over the world. The kit has been successfully assembled by middle school and high school students, giving them a lesson in electronics and the opportunity to participate in scientific studies.

Radio Jove offers a great opportunity for ham radio operators to become involved, perhaps with local schools, helping them to get on the air and make observations. The radio telescope kit costs under $200 and includes the receiver kit, most of the antenna hardware, as well as *SkyPipe* and *RJP* software. To learn more about the program visit the Jove Web site at **http://radiojove.gsfc.nasa.gov**.

Jon Wallace has been a high school science teacher in Meriden, Connecticut for over 28 years. He is past president of the Connecticut Association of Physics Teachers and was an instructor in Wesleyan University's Project ASTRO program. He has managed the Naugatuck Valley Community College observatory and run many astronomy classes and training sessions throughout Connecticut. Jon has had an interest in 'non-visual' astronomy for over twenty-five years and has built or purchased various receivers as well as building over 30 demonstration devices for class use and public displays. He is currently on the Board of the Society of Amateur Radio Astronomers (SARA) and developed teaching materials for SARA and the National Radio Astronomy Observatory (NRAO) for use with their Itty-Bitty radio Telescope (IBT) educational project. Other interests include collecting meteorites, raising arthropods ("bugs") and insectivorous plants. Jon has a BS in Geology from the University of Connecticut; a Master's Degree in Environmental Education from Southern Connecticut State University and a Certificate of Advanced Study (Sixth Year) in Science from Wesleyan University. He has been a member of ARRL for many years but is not a licensed Amateur Radio operator.

Richard Flagg has an educational background in physics, astronomy, and electrical engineering. He spent many years at the University of Florida studying radio emissions from the planet Jupiter and has worked as a telemetry engineer on the Eastern Test Range, developing and testing antennas and low noise receiving systems. Richard was also the principal engineer for the State of Hawaii commercial spaceport development project. For the last decade he has been involved with the NASA radio Jove educational outreach project. As AH6NM he has participated in over 75 SAREX and ARISS telebridge contacts and enjoys satellite tracking and chasing DX – mostly on 20 meter CW. QEX

Dr. Sam Green, WØPCE

10951 Pem Rd, Saint Louis, MO 63146; w0pce@arrl.net

An Oscilloscope Camera Mount

Take perfect waveform photos with your digital camera.

I make a lot of measurements with an oscilloscope, which I record in order to present the results. In decades past when my employer provided a laboratory and paid for the materials, I went through a lot of Polaroid film packs. These were never inexpensive, but such film now costs upwards of $1.50 per shot.

Up-to-date oscilloscopes are computers and record screen images to USB memory sticks. A decade ago, some oscilloscopes had floppy disk drives to record screen images.

I do a lot of work at home and still need to record results with older equipment that requires photographic recording. The modern digital camera makes this fairly easy, but if you've ever tried to hold a camera steady in front of an oscilloscope screen, you will appreciate the stability that a purpose-made mount provides. I built this unit with available parts and little care for looks. I didn't purchase anything. It isn't pretty, but the mount in Figure 1 really does the job.

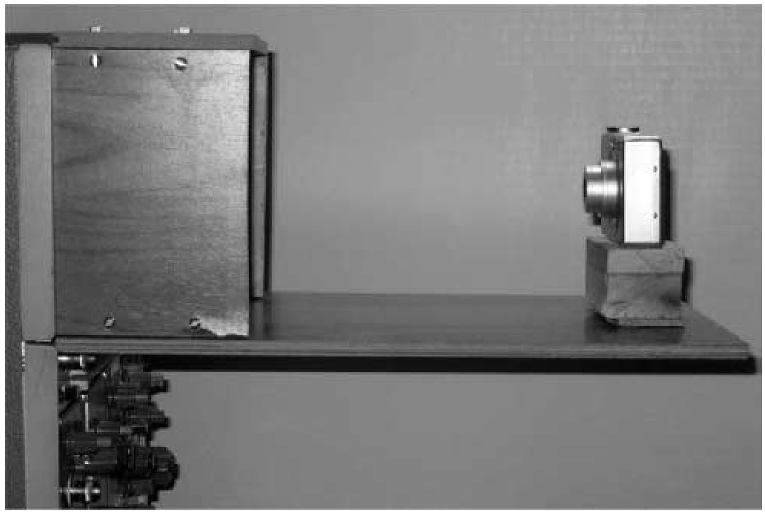


Figure 1 — The completed camera mount attaches to the front of a Tektronix oscilloscope and positions a digital camera with the lens centered in front of the screen.

Make your own

I only provide guidelines here, because each design is custom. Various oscilloscopes have different size screens, and various cameras have different heights to the center of the lens and different horizontal spacing from the center of the lens to the tripod screw mount.

Your oscilloscope must be less than 40 years old to have a rectangular bezel around the CRT. This excludes old Dumont and Eico and really old Heathkit oscilloscopes with circular bezels.

The material in my example is ¼ inch thick flooring material that a colleague at work discarded, so there must be ¼ inch clearance around the bezel. Figures 2 and 3 show two more overall views. If there are any knobs or bolt heads close to the bezel that interfere, consider how you might accommodate these.

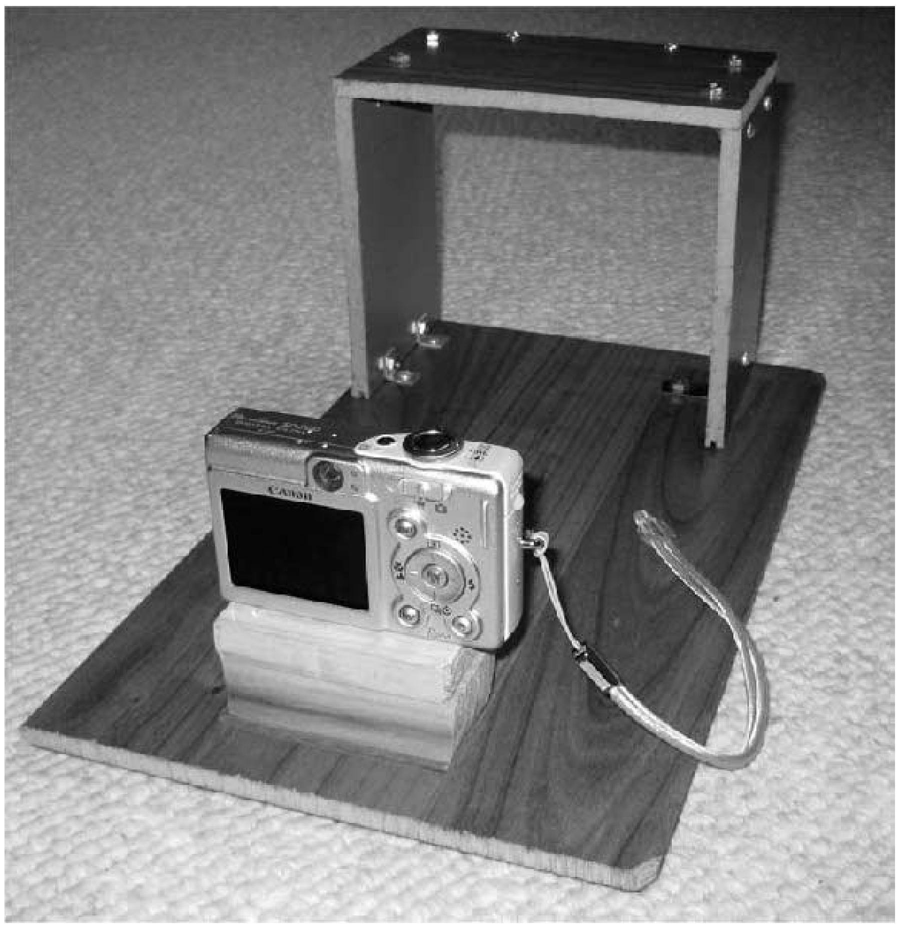


Figure 2 — Here, the completed camera mount is ready to be placed on the oscilloscope. Note that simple angle brackets hold the pieces of the frame together.

My unit is about 12 inches long with the camera front almost 11 inches from the screen. I use the zoom feature to increase the size of the screen image within the format. If your digital camera doesn't zoom, find the optimum screen to camera lens experimentally beforehand. If your digital camera does not have a macro short range focus option, stop here or make sure the screen is in focus at a range that fills a good portion of the view screen.

Build a snug fitting frame around the rectangular bezel. If you're lucky enough to have a Tektronix oscilloscope, the bezel has a slot on top as shown in Figure 4, to accommodate the original oscilloscope camera adapter. I ran a couple of short wood screws into the top of my frame so that the screw points penetrate into this slot to hold the camera mount onto the instrument. Figure 5 shows these screws at the top, and Figure 6 shows the mount held to the slot by these screws. My Kikusui oscilloscope has a similar a slot, as do some other brands. With no slot, you must find another way to hold the camera mount in place, or perhaps set the oscilloscope on the floor with the screen facing up so the camera mount can simply rest in place.

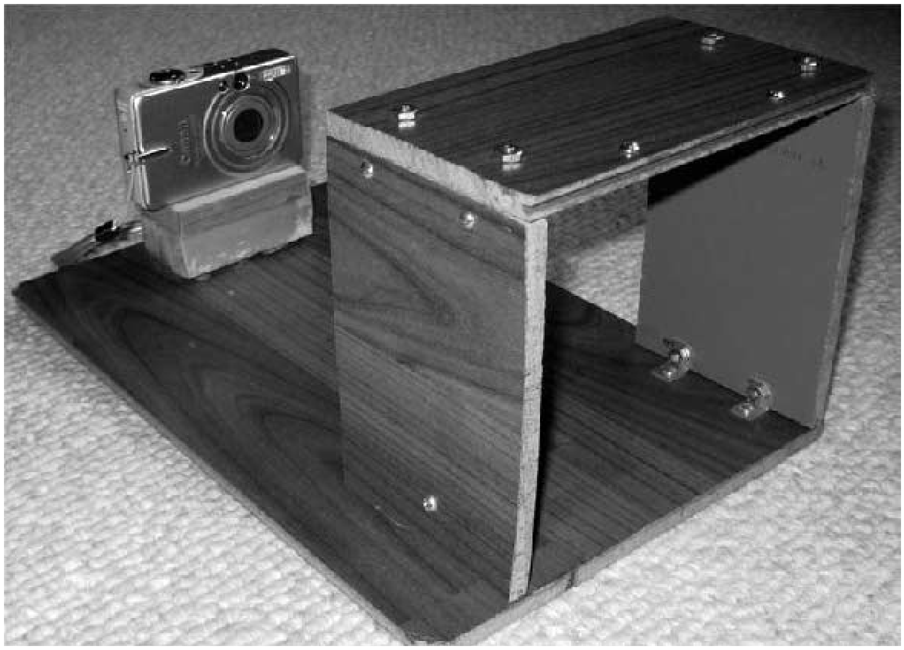


Figure 3 — This photo shows the frame from the oscilloscope end.

Figure 4 — The top edge of the frame around this Tektronix oscilloscope has a V groove designed to hold a camera mount. The author drove two wood screws through the top of his camera mount box so the points of the screws protrude just enough to engage in this groove.

Measure and Calculate

Now you have a platform that fits to the bezel. Next calculate where to position the camera so that the center of the lens is at the horizontal and vertical center of the screen.

First determine where to place the ¼ inch camera mount hole to center the lens horizontally. Double check this before drilling. Then determine how much you must elevate the camera to place the center of the lens at the same height as the center of the screen. Shim to this height with blocks of wood. You don't need to plane a shim to fine-tune this height adjustment, though. It's more important to keep the camera horizontal that to get it exactly in the center. Then drill a hole for the camera mounting bolt through the elevating blocks. Figure 7 shows the details of the assembly.

The Hard Part

Here is the hard part. You must find and perhaps modify a ¼-20 bolt of length sufficient to pass through the base board and shim blocks plus a few threads to screw into the camera. I found some "swivel pad clamps" in a couple sizes for $7.48 plus shipping.[1] These bolts have a 1 inch diameter head designed for clamping something in place with simple finger tightening. You may be able to find something similar, or use an ordinary Allen

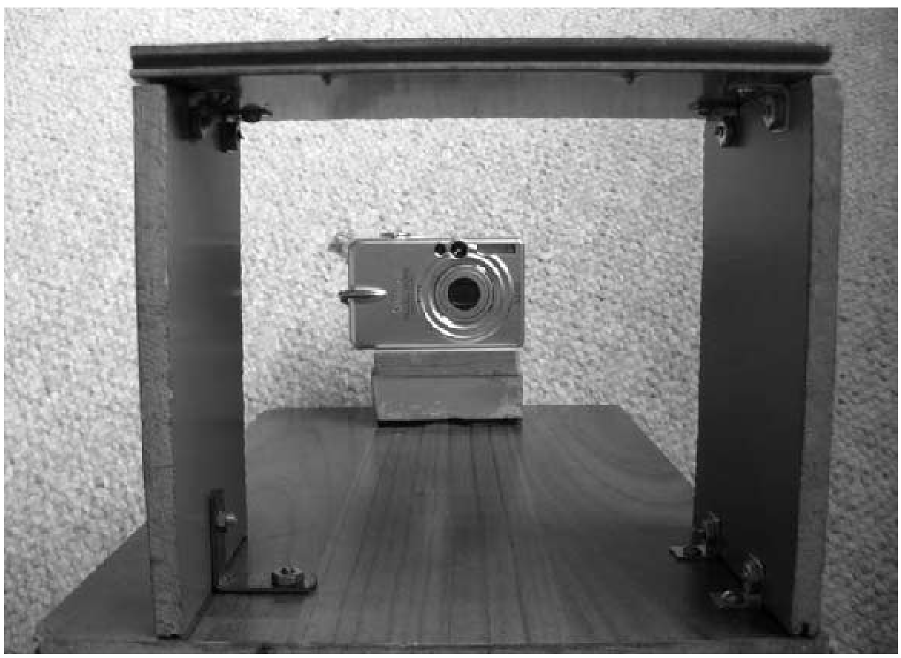


Figure 5 — Looking into the camera from the oscilloscope end of the camera mount frame, you can see the points of the screws protruding slightly through the top board.

[1]I found long ¼-20 "swivel pad clamps" via mail order at Reid Supply. **www.reidsupply.com/Detail.aspx?itm=AM-2**.

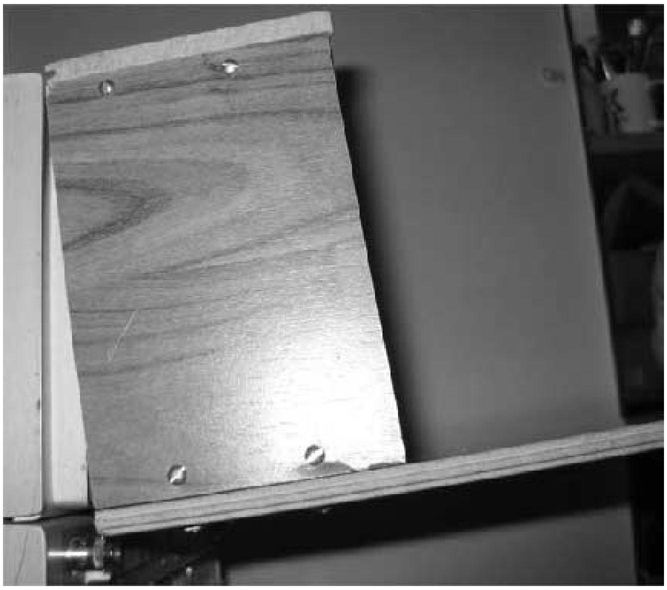


Figure 6 — The screw points in the top board fit into the V groove of the oscilloscope frame, and the mounting system hinges down into position.

Figure 7 — This photo shows the spacer boards used to shim the camera to the proper height so the lens will be centered on the oscilloscope display.

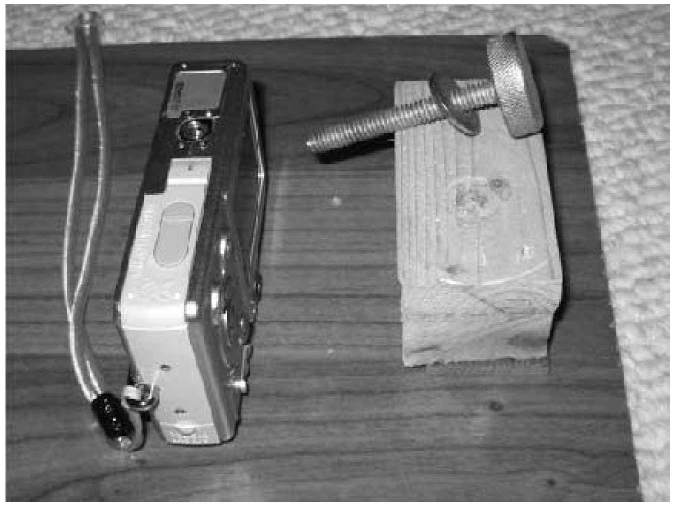


Figure 8 — Here you can see the spacer block and the long "swivel pad clamp" screw as well as the tripod mounting hole on the bottom of the camera.

Figure 9 — Viewed from the bottom of the camera mounting system, you can see that there are several holes to accommodate the "swivel pad clamp" screw for attaching various cameras for proper positioning of the camera lens.

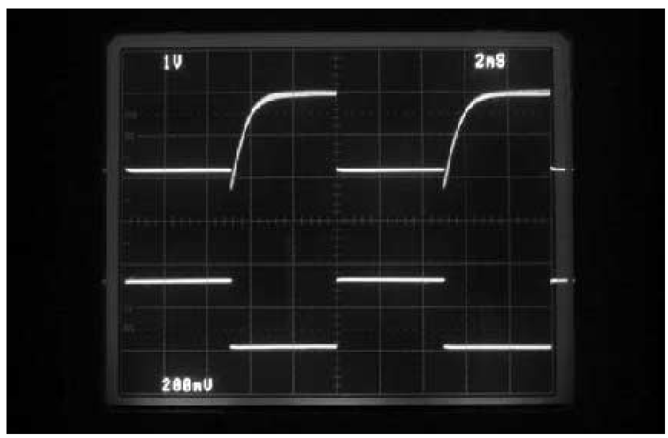


Figure 10 — This photo of the oscilloscope screen shows the clear photos that the author produces with his camera mounting system. Notice especially that the screen frame is rectangular, level and square, with no "pin cushion effect" along the edges of the screen.

or hex head bolt with large-diameter washers. If the bolt is too long, grind it down or use washers to take up some length. I did both. Use at least one washer to keep a smaller bolt head from grinding into the wood. Do not use a wrench to force a bolt that is too long into the camera base.

Figures 8 and 9 show the mounting hardware detail. You can see additional holes in the bottom view of Figure 9 from when this mount supported a different camera with different positioning requirements.

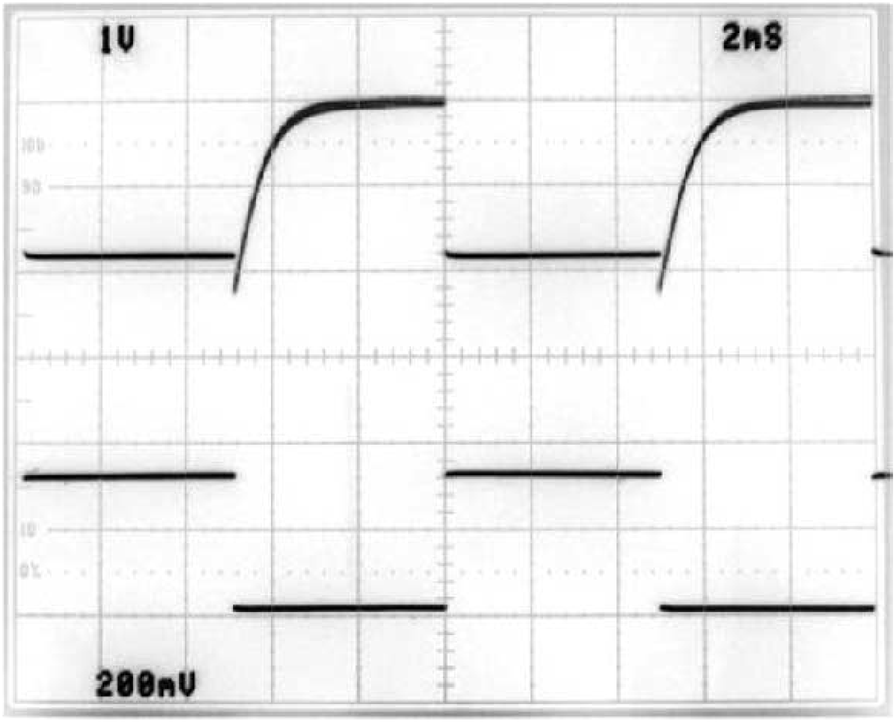


Figure 11 — By using a photo editing program, the author is able to convert the photo to a "negative" image and then produce a gray scale image for publication.

Using the Camera Mount

For simplicity, this mount does not completely enclose the camera to screen path. You get to see the screen while you make pre-photo adjustments, but you must turn off the ambient lighting when taking photos. I turn off nearby lamps and one across the basement to prevent reflections from the screen.

Adjust the zoom so that the camera does not crop the screen image. If you were careful, the image on the back of your digital camera is perfectly horizontal. Turn off the flash and enable the macro short range focus option in the menu by selecting the flower rather than the mountain. Snap some screen images. Zoom into a saved image to be sure it is in correct focus. The lowest possible camera resolution is perfectly satisfactory for oscilloscope waveform photos. I use 640 × 480 pixel image size, but I keep the jpeg resolution at "fine" because I hate jpeg artifacts. You can trade-off a smaller image on the view screen by using an image size with more pixels like 1600 × 1200 before you crop as I describe below.

Figure 10 is a raw image that shows I didn't elevate my camera quite high enough. I didn't have a sufficiently long bolt.

Get It to the Computer

Finally, transfer the images from the camera to a personal computer to process the images in a program like PhotoShop or other editing software. I use Paint Shop Pro. Select the portion of the photo that corresponds to the desired screen image and discard the rest. Finally, change the color to negative. This reduces the file size typically by one third to around 30 kilobytes, saves a lot of ink or toner when printing, makes dim traces more apparent, and looks better.

Figure 11 shows a cropped image ready to publish in grey scale format with contrast enhancement.

In Conclusion

Try building this simple photographic aid. Spending an evening to improve your waveform recording will be well worth the effort.

Dr. Sam Green, WØPCE, is an aerospace engineer in Saint Louis, Missouri, with degrees in Electronic Engineering from Northwestern University and the University of Illinois at Urbana. Sam specializes in optical data communications and photonics. Sam became KN9KEQ and K9KEQ in 1957 while a high school freshman in Skokie, Illinois, and held a Technician class license for 36 years before finally upgrading to Amateur Extra Class in 1993. Sam is a Registered Professional Engineer in Missouri, a life senior member of IEEE, an ARRL Member, and a member of BEARS, the Boeing Employee Amateur Radio Society. Sam holds fourteen patents, and three more are pending. QEX

Next Issue in *QEX*

Dick Jansson, KD1K, shares some of his thermal design expertise with us as he describes the operation and applications of heat pipes. Dick has done thermal design work on a number of AMSAT OSCAR satellites, including his use of heat pipes in the AO-40 (Phase 3D) satellite. Today you might find heat pipe technology used to help cool the CPU in your computer. Dick explains how these devices operate, and even suggests an Amateur Radio application — cooling a transistor power amplifier with a CPU heat pipe.

Ray Mack, W5IFS

17060 Conway Springs Ct, Austin, TX 78717; w5ifs@arrl.net

SDR: Simplified

SDR: Simplified Web Page

I have obtained the web address **www.dsp-radio-resources.info** so that I can communicate updates and corrections between installments of the column. As I write this, the page is empty, but it should have content soon.

As promised, I worked on our software development tools over the holidays and the ensuing months. Once again, it was quite an adventure. I have finally gotten to the point of being able to decompress archives, upload and download files, and build binary images. This installment chronicles what I learned.

Uncompressing *Linux* Archives

The latest source code for *uCLinux* comes in the file **uClinux-dist-2009R1.1-RC4.tar.bz2**. The last extension is the compression type (bzip) and the next indicates that it is a *Linux* TAR file. This file is called a *tarball* (after the ugly things that float in the ocean). TAR is an abbreviation for Tape ARchive from the old days when software was physically moved from computer to computer on massive reels of magnetic tape. *Windows* archive utilities do not understand tar files, so we need a *Linux* tool to decompress the file into its component files for use on *Windows*. I found a free *Linux* archive utility at **the freecountry.com**. Use the URL **thefree country.com/utilities/archivers.shtml** and get PeaZIP. This utility understands all of the archive types on the Stamp distribution CD and on the Blackfin Koop Web site.

Transferring Files to *uBoot*

A quick and easy way to load binary files into memory for execution is called tftp (trivial file transfer protocol). One of the commands that is part of the *uBoot* debug monitor on the Stamp os tftp. The Blackfin Web pages give a link to *TFTP Turbo* from Weird Solutions. They have a personal version that you can download from **corporate.weird-solutions.com/products/tftp-turbo**. This is both a server and a client so you can use tftp to send and receive files from other computers. *TFTP Turbo* works on *Windows XP* and *Windows 2000* but not on *Windows 7*. If you are running *Windows 7*, you will need to get the TFTP server from SolarWinds. Go to **solarwinds.com/downloads/** to find the free down load. You will need to register with them to download the file, but it is free.

The Blackfin Web site has a topic in the documentation that describes how to set up a tftp server. The documentation is at **docs.blackfin.uclinux.org/doku.php?id=setting_up_a_tftp_server**. The information for *Windows* is at the bottom. One thing the Blackfin Web site does not tell you is that the *Windows* firewall (especially *Windows 7*) is very good at stopping access. TFTP uses UDP packets and port 69 (an arcane part of Internet operation) to connect between your Stamp and the computer. You need to go into the network configuration/firewall utility in Control Panel and allow Port 69 to have access to your computer.

TFTP is an extremely unsecure method of file transfer. You should leave your TFTP server disabled when not in use and probably disconnect from the Internet when transferring files. Definitely keep your virus scan software up to date! Unfortunately, *uBoot* does not have a more robust file transfer mechanism unless you want to use Y Modem or Kermit serial protocols. This is another place where *Windows 7* is not your friend. All *Windows* versions before *7* came with Hyperterm from Hilgraeve Systems and this allowed Y modem and Kermit. *Windows 7* does not come with a terminal program and Hilgraeve no longer gives away a free version. A special note to Hyperterm users with *Windows XP*: the scroll back buffer is terribly broken in the program as shipped by Microsoft. Only the window with the first 25 lines has a correct display. I also have an older version of *Procomm* that I loaded on my *Windows 7* Netbook. It does not handle file transfers. There is likely something special that makes the older version not play well with *Windows 7*.

Once your TFTP server is installed, you need to set it up to run. The SolarWinds server is set up under the File\Configure menu. You need to set the place where the files are stored. You also need to start the server. The SolarWinds server has a Start button on the configuration dialog. You need to click that button to start up the server.

Once you have tftp running on the *Windows* machine, you are ready to load a file into memory on the Stamp. I configure my tftp server to use c:\temp as the source directory for my files. I build a binary (but you can test with a text file) and place it in the c:\temp directory. I tested using the example program "blink" that is in the *Windows* compiler directory. The next step is to set the Stamp so it knows where to find your computer. Run the command "printenv" and it will show you all of the environment variables for *uBoot*. One of those is "serverip". You need the IP address of your *Windows* machine to do the transfer. Open a command prompt window (in the start menu under accessories) and type "ipconfig". The IP address will be listed. It is generally something like 192.168.1.102. Now go to the command window for the Stamp and type "set serverip 192.168.1.102" (assuming the IP address you found was 192.168.1.102). The last set up step is to connect your Stamp to the network. Type "dhcp" so that the Stamp gets its own IP address. When you type "tftp 1000000 blink" the Stamp will contact the computer and transfer the file "blink" into the address 0x1000000 in memory.

Linux and Software Development Computers

I built a *Linux* machine from an old desktop PC I had around the house, but I can't use it to work over lunch at the office. WalMart had a sale on Netbooks for $228 over the holidays, so I picked one up for use during lunchtime. This replaced my ten year old laptop that finally became too crippled to use. You might want to try WalMart in case they still have some at that price.

There are a few things I really like about using a netbook as a development system. The first is that it is ultimately portable. It weighs just a couple of pounds and is about the size of a text book. Even better is that it will hook to a USB keyboard, USB mouse, and full size monitor to give 1920 by 1080 screen resolution. I essentially have a full featured desktop computer in 2 pounds.

We talked last year about using *coLinux* as a virtual computer on top of *Windows*. At least for me, this was not terribly useful. I found a way to use *Ubuntu Linux* without modifying my *Windows* machine in any way. *Linux* fully understands *Windows* disks, so you can boot a second drive to give you *Linux* and still get to all of your *Windows* working directories.

All the computers I have seen over the past three years or so allow you to boot from either a CD or a USB disk in addition to the internal hard disk. In searching for tools, I found a way to boot *Linux* from a USB Flash drive or USB hard drive. Booting from a Flash drive is important because it allows us to have a *Linux* machine that will seamlessly deposit files on our *Windows* hard drive as well as allowing building a new image of *Linux*. I tried to build a USB hard drive system, but that was painful and not successful. The Flash system on a 2 GB drive works adequately, though. I will continue to work on getting an external hard drive to boot *Linux*, since that will allow us to build device drivers and the entire operating system.

The first step in the process is to go to the Ubuntu Web site (**www.ubuntu.com/GetUbuntu/download**) and download the CD image for the install CD. This will place the file **ubuntu-9.10-desktop-i386.iso** (690 MB size) on your *Windows* hard drive. Note where your browser saved the file. Next you need to go to the boot utility site (**www.pendrivelinux.com/create-a-ubuntu-9-10-live-usb-in-windows/**). You really need to start at step 2 of the instructions since you have already downloaded *Ubuntu Linux* from the Ubuntu site. Step 2 downloads the file **USB-Installer-for-Ubuntu-v0.2.exe** onto your system. Insert your 2 GB or larger Flash Drive in your *Windows* machine. Use the *Windows Explorer* application to format the Flash drive as a FAT32 volume. It is important that you use this format. **Once this process is complete you should not attempt to use this drive with *Windows*!** After the drive is formatted, run the *USB-installer-for-Ubuntu-v0.2* program. It will prompt you for the location of your Ubuntu ISO file, and then you can let the program create your boot drive. This takes about 15 minutes, so be patient while it works.

The last step is to modify the boot sequence for your computer. This is a place where every computer is different depending on which company produced the BIOS. Your computer documentation should tell you how to interrupt the boot sequence and start the BIOS configuration. It is almost universal that continually pressing the space bar immediately after turning on the computer will send you to a point where you can get to the configuration utility. You will have to search the menus in order to find the step that allows you to set up the system to boot from the USB drive first and then boot from the internal hard drive if the USB boot does not show a bootable drive.

Do not try to use your new *Linux* Flash drive in more than one computer. This confuses the boot process and will likely result in the need to reload *Linux* on your flash drive. You should not store any critical data on your new *Linux* Flash drive just in case you need to reformat it. Using a Flash drive is not perfect. At some point you will encounter a problem such as not being able to connect to your *Windows* C: drive or the user interface just never comes up. The solution is to reformat the Flash drive.

Now, you can choose to run your computer as a *Linux* machine by inserting your USB Flash drive and turning on the power. Turning on power without the USB Flash drive will start *Windows* normally.

The system will ask you if you want to run *Linux* from the USB drive as well as presenting a few other options. Choose to run from the USB drive. *Linux* is a little different from *Windows* when it starts. You will see hundreds of commands go by on the screen as it starts up. These are diagnostic messages that the *Linux* developers use to ensure it boots correctly. The last thing that happens is that the system will log you on as the default user and bring up the user interface.

The user interface is very similar to that of *Windows*. The menu bar across the top is similar to the *Windows* Start menu. The items we want are in the "Places" menu. This is similar to *Windows Explorer* and lists several "disk drive" locations on the computer. The one we want is the C: drive on your *Windows* computer. There is a Volume Identifier on the hard drive. In my case, this is "eMachines." Another computer shows C: as "80.0GB Drive." You can double click that icon and open your C: drive. Let's assume you went to the Blackfin *Linux* site and downloaded the tar ball to the root of "C:" You will see an icon labeled "**uClinux-dist-2009R1.1-RC4.tar.bz2**." Double click that icon and the archive utility will launch, read the archive, and show you the contents. The menu at the top of the application has the option to decompress the archive. It is important to store archive files in the root of C: or in working directories in the root. *Linux* does not allow you to go into the "My Documents" area of the drive.

When you are finished doing a *Linux* operation, you can shut down the system by clicking the red On/Off button in the upper right of the screen. Just like *Windows*, it lets you shut down, restart, or hibernate. It is important that you allow the system to completely finish its job of shutting down before removing the Flash drive. *Linux* stores your configuration to the Flash drive as part of its shut down.

DSP Software Execution on the Stamp

There are two methods to run code on the Stamp. The first is called bare metal. In this mode, your program takes complete control of the CPU and all of its resources. Your program must handle all of the operations and the only practical way to debug is using a JTAG debugger (expensive). The second way is to run a program as a *Linux* application. In this case, *Linux* stops you from getting directly to large parts of the Stamp board hardware. You must use device drivers to access the hardware but you can debug your code using the software debugger called "gdb." (There is a kernel debugger for debugging device drivers, but that is a more advanced topic.)

There are two versions of the *C* cross compiler. The one for bare metal is "**elf\bin\bfin-elf-gcc.exe**." The one for *Linux* is "**linux-uclibc\bfin-linux-uclibc\bin\gcc.exe**." The compilers are actually identical. They will generate identical code. The difference is that the linker (the program that makes the binary image) is set to link your code to different libraries.

The libraries in the elf branch implement stub versions of functions like read and write of files. So, if you want to debug a bare metal application by sending messages to the serial port, you will have to implement the low level functions to manage the hardware. You must stay away from any functions that manipulate the I/O system when implementing a bare metal system unless you can implement any functions that touch the hardware.

The libraries in the "linux-uclibc" branch implement connections to the I/O subsystem of *Linux*. *Linux* implements a wall between your programs and the hardware. That wall requires that all of your connections to hardware call a *Linux* I/O function. In general, that means calling read(), write(), or ioctl(). The downside to using *Linux* is that it requires a device driver for every piece of hardware in the system. We ran into this problem when we tried to use the AD7476 with *Linux* as shipped on the Stamp board. We had problems using the SPI driver that is part of the image on the board.

Run Time Loading of Drivers in *Linux*

We can build a new version of the operating system that includes a new SPI driver, but we need a *Linux* computer for that process. The other option is to build just the SPI driver and set up the *Linux* on the Stamp to load the driver from your *Windows* computer every time you want to run the Stamp board. As you can imagine, that is pretty cumbersome. The best solution is to create a new operating system image and install it in the Flash memory. The danger, and the reason I haven't done it yet, is that you could wipe out the flash memory in a way that would require a JTAG debugger to recover. I am not quite ready for that leap of faith.

The process of installing a driver at runtime is straightforward even though it is time consuming. It is time consuming because files are stored in a RAM disk and they disappear when you turn off the power or reboot. The driver is called a kernel module because it runs with full privilege of the operating system (the kernel).

The first step of the process is to load the file onto a running *Linux* image using FTP. Once the file is loaded into the RAM disk in the proper place, we run a command called "insmod." The program reads the file header and connects the new module into the kernel. There are two other commands that we will use when we do run time loadable modules. "lsmod" shows which modules are installed, and "rmmod" removes a module. We will look at this process in more detail in a later installment.

Simple Hardware Testing

There is a third and intermediate level of control that will allow you to talk directly to the hardware using a method similar to BASIC's peek and poke. The boot monitor, *uBoot*, runs after the internal BF537 boot ROM runs. You can stop it from booting *Linux* by pressing the space bar immediately after *uBoot* starts. There are a number of commands in *uBoot* that allow reading and writing memory. Type "help" at the command prompt to see all of the available commands.

The memory read and write commands have multiple sub commands. It is not clear from the help you get in *uBoot* that you ***must*** use the word wide forms of the commands. The full 32 bit and byte wide forms give you garbage and can cause the system to reboot. The registers of the Blackfin processors are on 4 byte boundaries, but they are only 16 bits wide. You will see the data of each register duplicated when you do a memory dump. Changing registers can be done with commands such as "mm.w". You can change the SPI registers, as an example, by typing "mm.w ffc00500<Enter>." It prompts you for each 16 bit value. The board comes up with the first two SPI registers set to 0400 and ff00. You can write those values back as a test by typing 0400 and "Enter" twice, then ff00 and "Enter" twice.

This method is especially useful for testing new hardware since you can almost always read and write registers to do a single input or output operation. I used this method to verify that the combination ADC/DAC board was functional.

A Bare Metal Test Program

I finally managed to puzzle through all of the documentation for the Stamp, *uBoot*, *Linux*, and the compiler. The compiler comes with an example program called blink. You will find it in the "GNU_Toolchain\examples\standalone\bf537-stamp" directory. You will open a Command window and change to the directory with the blink application source code. Simply type "make" and the system will build all of the files necessary for the program. Copy the file "blink" to your tftp directory (C:\temp on my machine). Start your terminal emulator program and power up the Stamp. Press the space bar to stop *uBoot* from starting *Linux*. Type the command "dhcp" on the Stamp. Run "ipconfig" on your computer to determine its IP address. Type "set serverip IPADDRESS" where IPADDRESS is your computer's IP address. Type "tftp 1000000 blink" on the Stamp. You should see the transfer occur and about 67000 bytes transferred. Type "bootelf 1000000" on the Stamp. You will see several messages on the console that are the results of loading the file from address 0x1000000 up to address 0xffa00000. When you look at the Stamp board you should see the LEDs illuminating in a moving pattern.

On the Web and Next Installment

Now that we can write code, compile it, and load it on the Stamp to run, we will do the experiments that we looked at in theory during 2009. These include a hardware verification program for the combination ADC/DAC board and an AM receiver. I will post the results and code on both my **dsp-radio-resources.info** Web site and also send them for posting on the ARRL Web site at **www.arrl.org/qexfiles**.

The next installment will look at some more ambitious programs to do receiving and transmitting operations. With the steep learning curve of the software tools behind us, we can embark on some fun experiments and work towards a usable Software Defined Radio. QEX

Bob Kopski, K3NHI

25 W End Dr, Lansdale, PA 19446; kopskib@gmail.com

Tech Notes

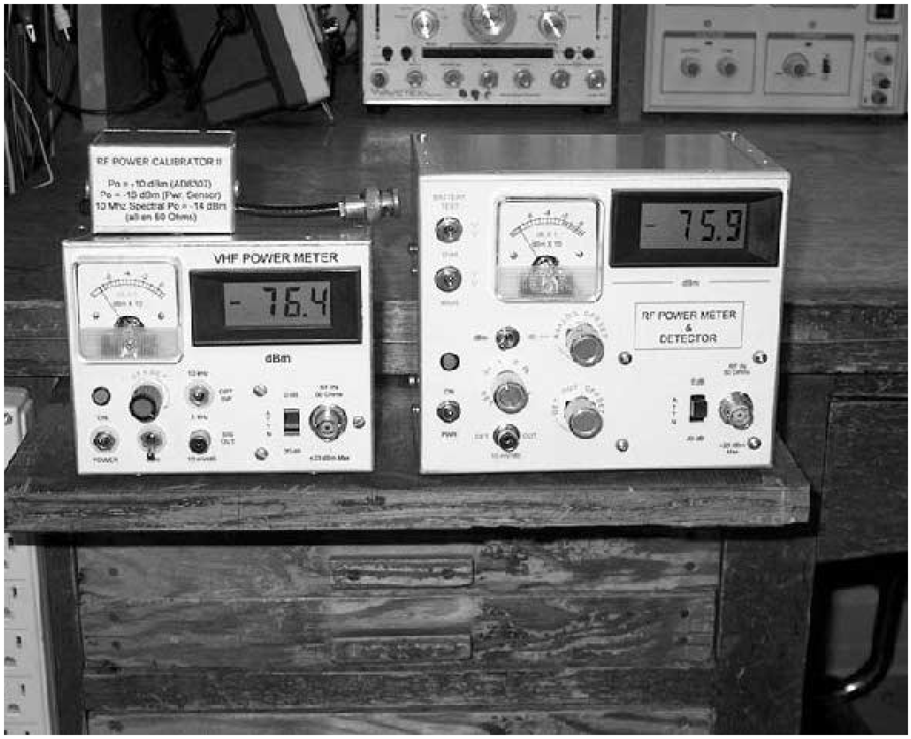


Figure 1 — This photo shows two power meters built by the author. The RF Power Calibrator II is on top of the VHF Power Meter, shown on the left.

The Analog Devices 8307 logarithmic amplifier has long been a popular component among home brew equipment builders. I first saw reference to its use in a simple power meter application in a June 2001 *QST* article by Wes Hayward, W7ZOI and Bob Larkin, W7PUA.[1] Later, with encouragement from Wes, I offered some embellished versions in *QEX* for May/June 2002 and Sept/Oct 2003.[2, 3] Still later, in the May/June 2005 issue of *QEX*, Roger Hayward, KA7EXM, offered a PIC-based variant.[4] In the Jan/Feb 2004 issue of *QEX*, I described “A Simple RF Power Calibrator” as an easy, low cost, and accurate way to calibrate any of the above instruments, plus spectrum analyzers and the more conventional sensor-type power meters as well.[5] This simple home brew device filled a missing spot in many home labs.

The Calibrator used an easily measured 10 MHz square wave as a reference signal source. Anyone with a DVM could set it up and then apply it as a known source of RF power. Its utilization in this regard was based on an Analog Devices assured performance feature of the AD8307, namely that a square wave having a peak to peak amplitude of ½ the peak to peak value of a sine wave produced the same response. That is, the 8307 saw these different waveforms as “the same.” Not only did I get this as personal assurance from Analog Devices (AD) years ago, but reference was made to this in the 8307 Rev A Data Sheet at the time. (You can download this data sheet at: **www.datasheetcatalog.org/datasheet/analogdevices/297929515AD8307_a.pdf**. See page 10, left column, at the bottom of the page.) The same statement still appears in the latest data sheet (Rev D) available at **www.analog.com/static/imported-files/data_sheets/AD8307.pdf**. See page 12 under “Intercept Calibration.” As designed and published, the Calibrator produced a –20 dBm level on a 50 Ω input 8307-based power meter. The same signal provided a –24 dBm reference for a spectrum analyzer, and a –23 dBm level for a conventional sensor-type power meter. (See Note 5 for the associated math.) Regrettably, the 8307 waveform-related performance has changed.

Recently I completed another variant of an 8307 power meter — some 6 years after building my first. In the process of calibrating it, I found something amiss in the application of my trusty Calibrator. It took weeks of effort and back-and-forth with AD, but eventually I learned that “newer” 8307s *do not* respond to non-sinusoids as do “older” 8307s. The reference paragraph in the 8307 Data Sheet is no longer applicable over the normal dynamic range of the IC. Evaluation of four device date code samples, beginning with but past (my original) 0140 up to 0713 demonstrate this.

This departure from the early performance is illustrated in two Microsoft *Excel* spreadsheet files that I created. See PM1PM2_4.xls and PM1PM2_5.xls, which are available for download from the *QEX* files Web site.[6] It is clear that, whereas one power meter responds nicely over the dynamic range of applied square wave power, the second power meter does not — it basically “falls apart” at about –20 dBm — exactly the design reference level of the Calibrator. Yet both power meters seem to work fine with sine wave inputs. I have verified this same behavior for several “newer,” variously date-coded 8307s, as has AD. (I have suggested to AD that a revision of their data sheet may be in order.)

While it is not clear exactly what has changed, it is clear that the Calibrator, with its design value of –20 dBm relative to 8307 applications, cannot be used with confidence with contemporary ICs. Maybe I should say contemporary ICs cannot be used with confidence with the Calibrator as designed. In any case, all data I’ve studied for square wave equivalent power levels greater than about –20 dB shows quite acceptable performance — as depicted in the *Excel* spreadsheet files mentioned earlier.

In consideration of the above I have revised the circuitry of the *QEX* Simple RF Power Calibrator to increase its output to –10 dBm for 8307 power meter applications. The new level then becomes –13 dBm as seen by power sensors, and –14 dBm (fundamental) on a spectrum analyzer.

[1]Notes appear on page 45.

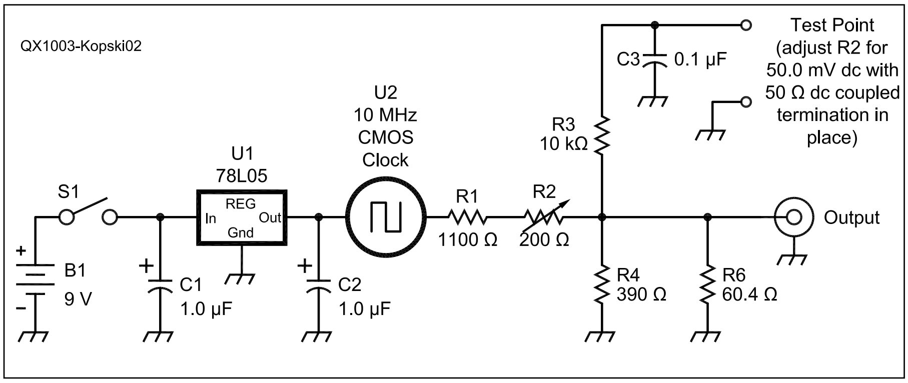


Figure 2 — Here is the schematic diagram for the Power Calibrator II. With a 50 Ω termination connected to the output, the circuit will produce a –10 dBm reading on the AD8307 based power meter. A laboratory power meter that is based on thermal measurements will read –13 dBm. With a spectrum analyzer you would read –14 dBm on the 10 MHz fundamental signal.

C1, C2 — 1.0 µF, 35 V tantalum
C3 — 0.1 µF ceramic
R1 — 1100 Ω
R2 — 200 Ω trim pot, Mouser 652-3306P-1-201 or equivalent
R3 — 10 kΩ
R4 — 390 Ω ¼ W, ±1% film
R6 — 60.4 Ω ¼ W, ±1% film
U1 — 5 V regulator, LM78L05
U2 — Oscillator, 10 MHz, CTX 045, Digi Key CTX114-ND or equivalent
Battery snap-on connector, Mouser 12BC016 or equivalent
Housing, LMB, Mouser 537-M00-P or as desired
¼ inch threaded standoffs, Mouser 534-8712 or equivalent
Output cable/connector as desired. The author used a coax pigtail with a BNC connector
Assorted screws, solder, tie wraps, circuit board material (using "ugly construction") and assorted hardware

Figure 1 is a photo of two of my power meters, with the revised power calibrator on top of the VHF Power Meter on the left. The revised circuit and parts values for the "RF Power Calibrator II" is shown in Figure 2. A comparison between the original and revised circuits would show the minor resistor value and wiring changes. The alignment procedure remains the same, but the new test point value is 50.0 mV dc as noted on the schematic. The output still looks like 50 Ω, and, when properly adjusted and then loaded with 50 Ω the output signal will be 100 mV_{pp}. Recall that a 100 mV_{pp} square wave "looks like" a 200 mV_{pp} sine wave, or –10 dBm to this IC.

As an aside, do note that all AD 8307 ICs respond very well to true sine waves. This can be seen in the *Excel* spreadsheets. All of these parts have degraded behavior with so-called "sine" inputs that are really not clean (undistorted) sine inputs. This means that the 8307 may not obey the published log linearity performance specification for these cases — something to be aware of.

Notes

[1]Wes Hayward, W7ZOI, and Bob Larkin, W7PUA, "Simple RF Power Measurement," June 2001 *QST*, pp 38-43.

[2]Bob Kopski, K3NHI, "An Advanced VHF Wattmeter," May/June 2002 *QEX*, pp 3-8.

[3]Bob Kopski, K3NHI, "A Simple Enhancement for the 'Advanced VHF Wattmeter,' " Sep/Oct 2003 *QEX*, pp 50-52.

[4]Roger Hayward, KA7EXM, "A PIC Based HF/VHF Power Meter," May/June 2005 *QEX*, pp 3-10.

[5]Bob Kopski, K3NHI, "A Simple RF Power Calibrator," Jan/Feb 2004 *QEX*, pp 51-54.

[6]The author's Microsoft Excel spreadsheets are available for download from the ARRL *QEX* Web site. Go to **www.arrl.org/qexfiles** and look for **3x10_Kopski.zip**.

QEX

Upcoming Conferences

Society of Amateur Radio Astronomers

July 4-7, 2010
National Radio Astronomy Observatory
Green Bank, West Virginia

2010 Annual Meeting

The Society of Amateur Radio Astronomers (SARA) will hold its 2010 Annual Meeting and Technical Conference, July 4 through 7, 2010, at the National Radio Astronomy Observatory (NRAO), Green Bank, West Virginia.

The last three years' Conferences were landmark accomplishments for our organization. Please help the Society of Amateur Radio Astronomers to make the 2010 Conference even better!

44th Annual Central States VHF Society Conference

July 22–24, 2010,
Doubletree Inn St Louis at Westport
1973 Craighshire
St Louis, MO 63146

Papers for publication, either for presentation at the conference or simply for inclusion in the Proceedings, are solicited for the Central States VHF Society Conference. The deadline for submissions to be included in the *Proceedings* is May 25, 2010. Poster displays are also solicited for the Conference. The deadline for notifying the Society that you plan to bring a poster for display is June 28, 2010.

Presentations aren't necessarily technical — they cover the breadth of the VHF/UHF ham radio hobby. Highlights in past years have been demonstrations of Software Defined Radio and LASER Communication beyond line-of-sight. Presentations generally vary from 15 to 45 minutes and step you through the highlights of the topic at hand, with complete texts published as articles in the *Proceedings*. To submit papers or indicate that you will have a poster for display, or with other questions about the Conference, contact Ron Ochu, KOØZ, No. 5 Cricklewood Ln, St. Peters, MO 63376; **ko0z@arrl.net**. There is more information about this year's conference available on the Central States VHF Society Web site at **www.csvhfs.org/conference/**.

The 29th Annual ARRL and TAPR Digital Communications Conference

September 24-26, 2010
Portland, Oregon

Mark your calendar and start making plans to attend the premier technical conference of the year, the 29th Annual ARRL and TAPR Digital Communications Conference to be held September 24-26, 2010, in Portland, Oregon. The conference location is The Heathman Lodge, 7801 NE Greenwood Dr, Vancouver, WA 98662. Phone: 1-888-475-3100; **www.heathmanlodge.com**. The Heathman Lodge is across the Columbia River from Portland.

We recommended that you book your room prior to arriving. TAPR has reserved a block of rooms at the special DCC room rate of $89.00 single/double. This special rate is good until September 1, 2010. After that you will pay the regular room rate. To book your room, call the hotel directly and mention the group code Tucson Amateur Packet Radio when making reservations. *Be sure to book your rooms early!*

The ARRL and TAPR Digital Communications Conference is an international forum for radio amateurs to meet, publish their work, and present new ideas and techniques. Presenters and attendees will have the opportunity to exchange ideas and learn about recent hardware and software advances, theories, experimental results, and practical applications.

Topics include, but are not limited to: Software defined radio (SDR), digital voice (D-Star, P25, WinDRM, FDMDV, G4GUO), digital satellite communications, Global Position System (GPS), precision timing, Automatic Position Reporting System (APRS), short messaging (a mode of APRS), Digital Signal Processing (DSP), HF digital modes, Internet interoperability with Amateur Radio networks, spread spectrum, IEEE 802.11 and other Part 15 license-exempt systems adaptable for Amateur Radio, using TCP/IP networking over amateur radio, mesh and peer to peer wireless networking, emergency and Homeland Defense backup digital communications, using Linux in Amateur Radio, updates on AX.25 and other wireless networking protocols.

This is a three-Day Conference (Friday, Saturday, Sunday). Technical and introductory sessions will be presented all day Friday and Saturday.

Join others at the conference for a Friday evening social get together. A Saturday evening banquet features an invited speaker and concludes with award presentations and prize drawings.

The ever-popular Sunday Seminar focuses on a topic and provides an in-depth four-hour presentation by an expert in the field. Check the TAPR Web site for more information: **www.tapr.org**.

Call for Papers

Technical papers are solicited for presentation and publication in the *Digital Communications Conference Proceedings*. Annual conference proceedings are published by the ARRL. Presentation at the conference is not required for publication. Submission of papers are due by 30 July 2010 and should be submitted to:

Maty Weinberg, ARRL, 225 Main Street, Newington, CT 06111, or via the Internet to **maty@arrl.org**.

14th INTERNATIONAL EME CONFERENCE

It is with great pleasure that Al Ward, W5LUA; Barry Malowanchuk, VE4M; Tony Emanuele, WA8RJF, and the North Texas Microwave Society officially announce the 14th International EME Conference. We invite you to Dallas, Texas on August 12-14, 2010 for the premier EME event of the year. Whether you are a new or an experienced EMEer, the conference will offer a wide range of technical, social and sightseeing activities for everyone.

The hallmark of past EME conferences has been the excellent technical presentations and the 14th EME Conference promises an outstanding technical program. Presentations include:

- Low Noise Amplifiers
- Tracking the Moon and other celestial bodies
- SSPAs/Tube Power Amplifier/ TWT Workshop
- Receiving with Software Defined Radios
- Big Dish EME
- Highlights from several DXpeditions.
- Getting started with a small dish on 1296 EME
- Feed design and construction
- EME Propagation
- Software
- Live EME demo on 1296 MHz by WA5WCP/5

Of course no technical portion of a Conference is complete without the Technical Proceedings. If you would like to present or submit a technical article for inclusion in the Proceedings please contact Al, Barry or Tony as soon as possible.

We also have several electronic vendors signed up to showcase their wares. The vendor rooms will be open all day Friday and Saturday. If you are interested in obtaining a vendor table, please contact us as soon as possible before the conference.

We will have a hospitality suite and registration on Wednesday evening so we can get acquainted. The Thursday family activities will include air conditioned bus tours in the Dallas-Ft Worth area. A Saturday night banquet at the hotel with a special guest speaker is also planned. A short session on Sunday morning will wrap things up.

The Conference Hotel is The Westin at the Dallas Fort Worth Airport, which offers first class conference amenities at an excellent conference price of $89 USD per night plus taxes. We have guest rooms booked for Wednesday night through Saturday night. Hotel registration is now available online at **www.starwoodmeeting.com/StarGroupsWeb/res?id=0902108016&key=7BF3D**.

If you have any difficulties in using the online Westin hotel registration, please call reservations at 888-627-8617 and reference the "EME Conference" to get the conference rate. If all else fails please drop Al a note at **w5lua@sbcglobal.net**.

Whether this will be your first EME Conference or your 14th we are looking forward to seeing you in Dallas in August. — *73 and good DX (via the Moon), Al Ward, W5LUA; Barry Malowanchuk, VE4MA, and Tony Emanuele, WA8RJF*

Letters to the Editor

EXPERIMENTAL DETERMINATION OF GROUND SYSTEM PERFORMANCE FOR HF VERTICALS (JAN/FEB 2009 THROUGH JAN/FEB 2010) LETTERS (MAR/APR 2010)

Larry,

With regard to the Letter from Rudy Severns, N6LF, in the Mar/Apr 2010 issue, his "Equation 18" is the same as "Scale (4)" (Reflection Loss, dB) of "Table I" in Phillip H. Smith's article on his "Mega-Rule" in *QST*, March 1969. [That article is available for download on the Members Only section of the ARRL Web site. — Ed.] An extensive table of "Reflection Loss, dB" and nine other function values appears in the Mega-Rule instruction manual as "Table II" and as "Table 14.1" in Smith's *Electronic Applications of the Smith Chart*, Noble Publishing, Atlanta, GA, 1995.

Further, with reference to Rudy's excellent summary article in *QST*, March 2010, I've already heard two enthusiastic, on-the-air references to it.

— 73, James L Olsen Jr, W3KMN, 5905 Landon Ln, Bethesda, MD 20817; **w3kmn@aol.com**

Hi James,

Thanks for the kind words. I think too many people are taking the *QST* article as gospel when it should be viewed only as an interesting set of experiments, which shed some light on a few questions. Even the *QEX* series, which is much more detailed, raises far more questions than it answers.

I was really hoping to encourage others to expand on my work (as I expanded on Sevick's) by showing how it might be done.

— 73, Rudy Severns, N6LF, PO Box 589, Cottage Grove, OR 974249; **n6lf@arrl.net**

A QUESTION FOR QEX READERS

Dear Larry,

It has been some years now since publishing my Dirodyne and Bedford receiver articles in *QEX*.[1, 2] I am still developing equipment that is radical in design, and as such I do tend to come across problems that few ever see. I wonder if the readers of *QEX* can assist me with a problem that I have never seen answered in the literature before, but which must be seen from time to time. Anyway here it is.

I have been using a balanced mixer with the Fairchild quad switch type FST3125 IC in a direct conversion receiver. It consists of four FETs, which simply act as four SPST switches. I have used these in direct conversion receivers and at the lower frequencies — say below 20 MHz — they work very well. From about 25 to 30 MHz, however, they introduce weak drifting birdies and other noise into the audio.

I recently went all out to find and fix the problem. I used two of the FETs as a series switching demodulator. The clock is a very clean square wave from a frequency divider as used commonly in ARRL literature. I decided to look at the antenna port of the receiver operating at about 28 MHz, and there was nothing unusual there. The normal carrier leak of about –50 dBm was evident and as clean as can be determined on my spectrum analyser (analyzer for your USA readers). No other signal was fed to the demodulator for this test.

When looking from about 300 to 500 MHz, between the harmonic content of the clock there were many spurs that were not quite stable but responded to temperature changes by a small amount. Typically they lay in multiples of 60 kHz at 300 MHz. See Figure 1. They were not evident below say 200 MHz at all.

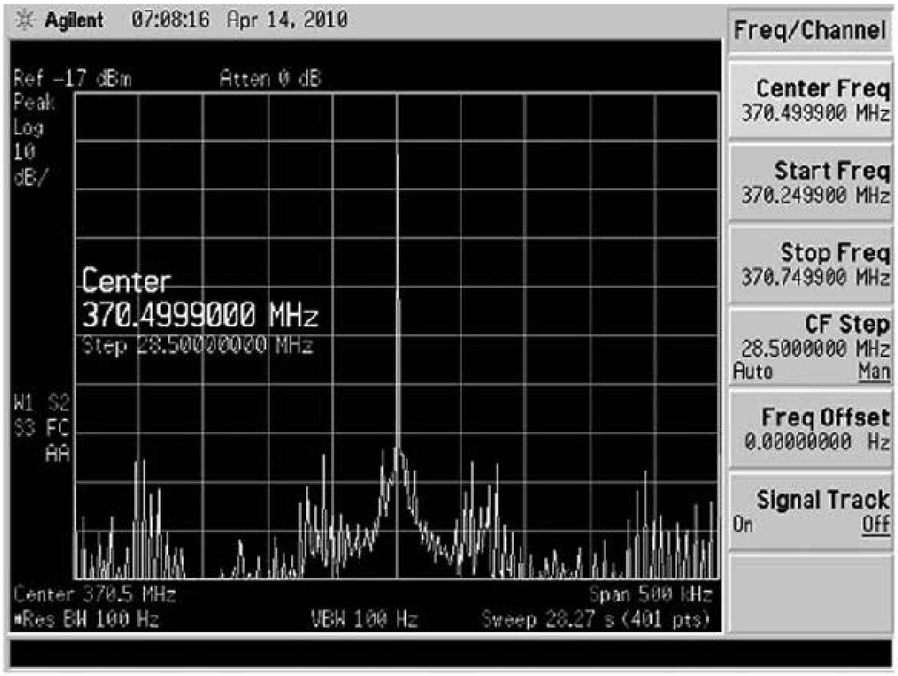


Figure 1—A spectral display of Rod Green's mixer output.

These spurs were not on the oscillator signal but were generated by the mixer itself. I can only suggest that the FETs in the mixer, which have a finite turn on time of about 5 ns must contribute to the problem in some way. It is baffling and I have never seen anything about this in print. It has occurred in every mixer using FET switching that I have built. It only occurs at the upper end of the HF spectrum. The mixer loss only rises by perhaps 1 dB or so, compared to lower frequency operation.

Has anyone ever had this problem, and what is more, solved it? I have done some experiments by placing an RC filter to limit the rise time of the clock signal. There is evidence that it has improved the situation at the antenna terminal but the audio noise is still present so I am not yet convinced that I have accomplished anything.

In closing, I enjoy your magazine and try to read every issue. Perhaps you may be able to start a questions from readers section, as I am sure that I am not the only one doing radical design.

— 73, Rod Green, VK6KRG, 106 Rosebery St, Bedford, WA 6052, Australia; **vk6krg@yahoo.com.au**

Hi Rod,

Let's see what our readers come up with. I hope we both hear from anyone who has experience using the FST3125 switches in a mixer circuit. It has been a while since you have published an article in *QEX*. Perhaps you would like to consider writing an article about your mixer experiments.

— 73, Larry Wolfgang, WR1B, QEX Editor; **lwolfgang@arrl.org**

Notes

[1]Rodney Green, VK6KRG, "The Bedford Receiver: A New Approach," *QEX*, Sep/Oct 1999, pp 9-23.

[2]Rodney Green, VK6KRG, "The Dirodyne: A New Radio Architecture?," *QEX*, Jul/Aug 2002, pp 3-12.