QEX

$5

March/April 2009

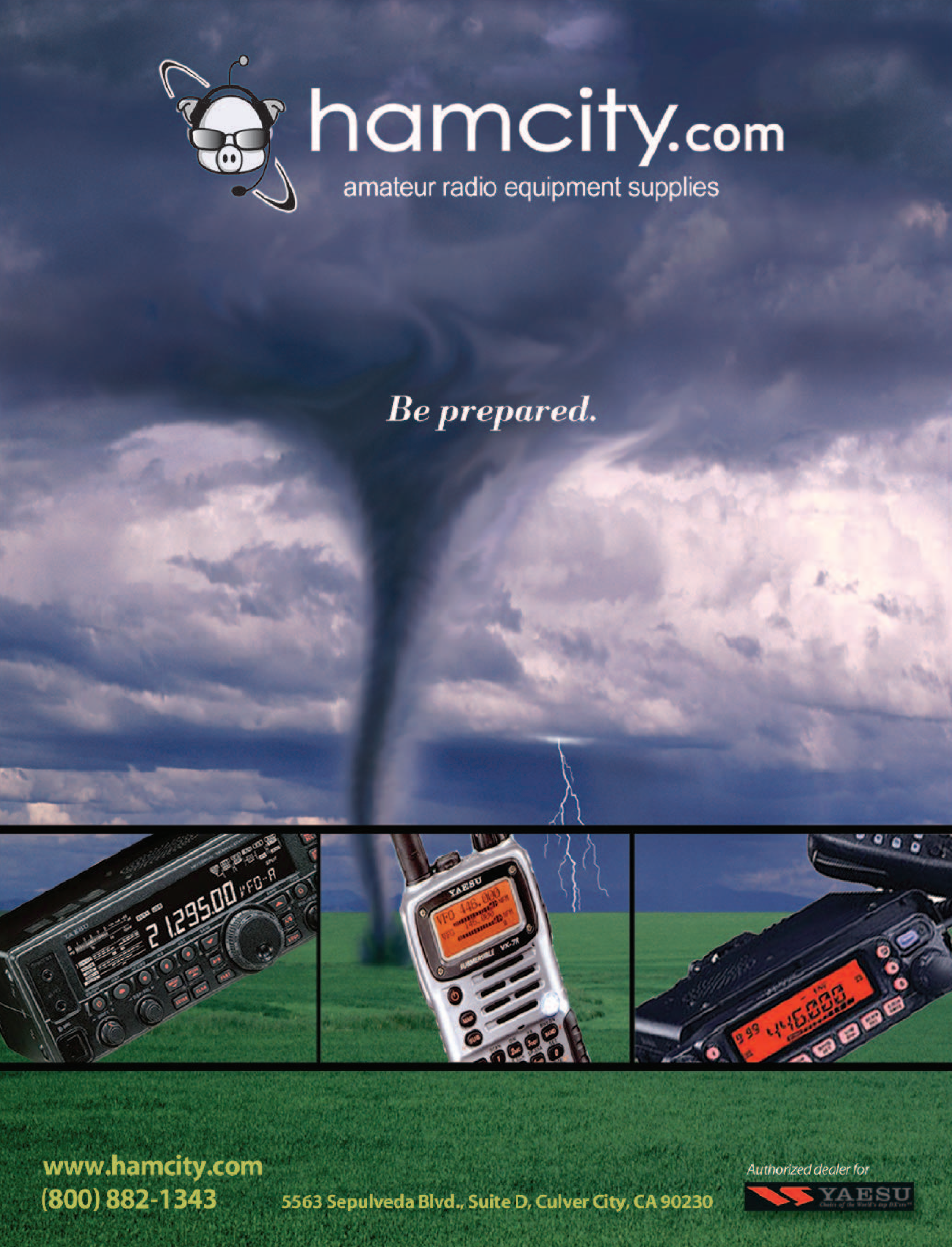
www.arrl.org

A Forum for Communications Experimenters

Issue No. 253

AF8L introduces his Cybernetic Sinusoidal Synthesizer. The modules in this system put automatic control theory into practice to generate high-purity, frequency and amplitude-stabilized sine waves.

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


hamcity.com
amateur radio equipment supplies
Be prepared.
www.hamcity.com
(800) 882-1343
5563 Sepulveda Blvd., Suite D, Culver City, CA 90230
Authorized dealer for
YAESU

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 253

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.

About the Cover

Gary Steinbaugh, AF8L, presents Part 1 of his Cybernetic Sinusoidal Synthesizer. Gary puts automatic control theory into practice to generate high-purity, frequency and amplitude-stabilized sine waves. The project includes an oven-stabilized crystal oscillator, a variable gain RF amplifier with ALC, a PLL frequency synthesizer and an RF power meter with digital readout in dBm as well as an analog voltage output.

In This Issue

Features

3 **A Cybernetic Sinusoidal Synthesizer**
By Gary Steinbaugh, PE, AF8L

6 **Some Assembly Required—How to Write a Simple Microcontroller Program**
By Roger Monroe, K7NTW

16 **NimbleSig III—Part 2**
By Thomas M. Alldread, VA7TA

25 **A Versatile Two-Tone Audio Generator for SSB Testing**
By Ken Grant, VE3FIT

29 **Experimental Determination of Ground System Performance for HF Verticals—Part 3**
By Rudy Severns, N6LF

33 **Exploring Near-End-Fed Wire Antennas**
By Ron Skelton, W6WO

Columns

35 **Tech Notes**
By Mark Spencer, WA8SME

41 **SDR: Simplified**
By Ray Mack, W5IFS

45 **Letters to the Editor**

46 **In the Next Issue of *QEX***

47 **Reader's Page**
By Larry Wolfgang, WR1B

48 **Upcoming Conferences**

Index of Advertisers

American Radio Relay League:...... Cover III
Atomic Time:..28
Down East Microwave Inc:........................32
hamcity.com.................................... Cover II
Kenwood Communications:............Cover IV
National RF, Inc: 32
Nemal Electronics International, Inc:........28
RF Parts 45, 47
Teri Software:..48
Tucson Amateur Packet Radio: 40

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: JOEL HARRISON, W5ZN
528 Miller Rd, Judsonia, AR 72081

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**.

Any opinions expressed in *QEX* are those of the authors, not necessarily those of the Editor or the League. While we strive to ensure all material is technically correct, authors are expected to defend their own assertions. Products mentioned are included for your information only; no endorsement is implied. Readers are cautioned to verify the availability of products before sending money to vendors.

Larry Wolfgang, WR1B

lwolfgang@arrl.org

Empirical Outlook

Be Prepared!

Some readers will recognize that statement as the Boy Scout Motto. What does that have to do with Amateur Radio? Well, maybe nothing, and maybe everything. That motto has been on my mind recently for a variety of reasons. As an active Scouter, I look for ways to reinforce its importance to Scouts. It is also something I try to apply to most areas of my life.

An often quoted story involves a question posed to Lord Robert S S Baden-Powell on the meaning of that motto. Someone asked, "Prepared for what?" and the founder of Scouting is reported to have answered, "Why for any old thing!"

Young Scouts most often think about being prepared when they are packing for a camping trip. "Will I have everything I really need for this trip? What weather conditions and other challenges might I face?" With more experience, however, we learn that being prepared involves much more than packing for a campout.

My oldest son is preparing for a summer canoe trip with his Troop on the Allegash Wilderness Waterway in northern Maine. He is concerned with exercising to lose some weight and building muscle and endurance for the trip. He understands the need to "be prepared" for the conditions he knows he will face on that trip. My Troop is considering a canoe trip in the Delaware Water Gap National Recreation Area this summer, and we are talking about the necessary preparations for that trip. Young Scouts need to learn and practice safe river canoeing skills as well as being prepared for a week of "wilderness camping."

What does all this have to do with Amateur Radio? Well, for one thing, any "wilderness camping plan" should include an emergency communications plan. There are many options to consider. Today, many people carry a cell phone and expect to be able to call for help in an emergency. That could be a serious mistake, however, in an area where there will be no cell phone coverage, or where that coverage may be spotty at best. A small QRP rig, battery pack and lightweight antenna can be one good option to consider. As an added benefit, Scouts can be introduced to the fun of ham radio as well as proper radio communications procedures.

How prepared are you? What would you do if you lost power and your town or city became isolated from outside resources, as happened to some parts of our country several times this past winter? ARES members are asked to have a "go kit" of radio equipment and even clothes, food and water ready to go at a moment's notice. How long could you be self sufficient, even in your own home? Of course everyone needs to be concerned with meeting the needs of their own families first, but as Amateur Radio operators we should also be prepared to step in with communications support for friends and neighbors. I am sure there will be significant challenges in any of these situations, and no doubt there will be lessons learned about what could have been done differently to be better prepared.

A question I often pose to Eagle Scout candidates is "How can you be prepared for something that you have no idea is going to happen?" Part of the best answers is that you can think about how you would respond to situations even though you may never face them. You can learn from every experience and think about how to draw on those experiences to solve new problems. You can also learn to have confidence in your ability to evaluate a situation and then develop a solution.

Do you practice for emergency communications conditions? Have you participated in Field Day and similar activities, where you have to learn how to solve problems on the spot? I am sure you have heard some variation of the statement, often applied to sports teams, "You play like you practice, so practice the way you want to play."

"Learn by doing" is another good mantra for us to consider. I am reminded of a Scouting event where a fairly new ham and I were doing a ham radio demonstration at a Scout Camp. When his ac power supply stopped working, my friend tried to switch to a back-up battery he had brought to the camp. Isn't it great to be prepared? Then a power supply connector pulled off of one wire to his radio. He also didn't have a secure way to attach the power cable to the battery. Frustrated, he was nearly ready to just pack up and go home because everything seemed to be going wrong. I had a toolbox stocked with items that allowed us to get his station on the air again in a few minutes. I may not have been much better prepared than he was, but over the years I had gained some experience with "emergency repairs." So, Scouting and Amateur Radio have a lot in common when it comes to being prepared "for any old thing."

Gary Steinbaugh, PE, AF8L

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

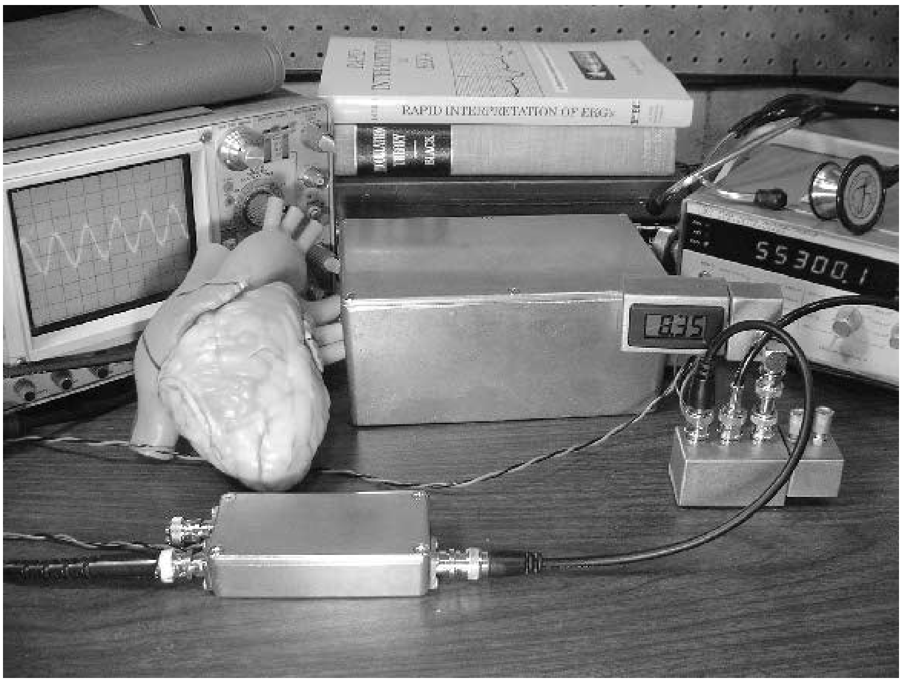
A Cybernetic Sinusoidal Synthesizer

A system of four modules puts automatic control theory into practice to generate high purity, frequency and amplitude stabilized oscillations.

This series of articles presents an introduction to automatic control theory as applied to communications systems, and employs this theory in four independent construction projects:

1. An oven-stabilized crystal-controlled oscillator (OXCO) with a high-purity output,
2. An RF power meter with both a digital readout in dBm and an analog voltage output,
3. A variable-gain RF amplifier for automatic power level control, and
4. A phase locked loop (PLL) frequency synthesizer with a low phase noise sinusoidal output.

With 50 Ω RF input and output impedances, these bite-size modules are useful separately, and together they produce a frequency-stabilized, amplitude-stabilized sine wave with a quality not possible from a simple crystal oscillator: a *Cybernetic Sinusoidal Synthesizer*.



"The heart of this system presents with no arrhythmias."

A Little History, A Little Theory

The concept of feedback has always held a great fascination for me. Corrective feedback is probably most familiar to Amateur Radio operators in automatic gain control circuits, but it appears all around us, from wall thermostats to robotic surgery equipment … and also within us. Your body is full of homeostasis: temperature, blood pressure and pH, cardiac and pulmonary rates, and many other life processes depend on corrective feedback. Consciousness is thought to spring from closed neurological loops, and there is even a theory of consciousness that is based on the interaction of electromagnetic fields produced in the brain.[1, 2] Although Norbert Wiener coined the English word *cybernetics* for a 1946 Macy Foundation conference, its ancient Greek root, κυβερνητησ, was used by Plato

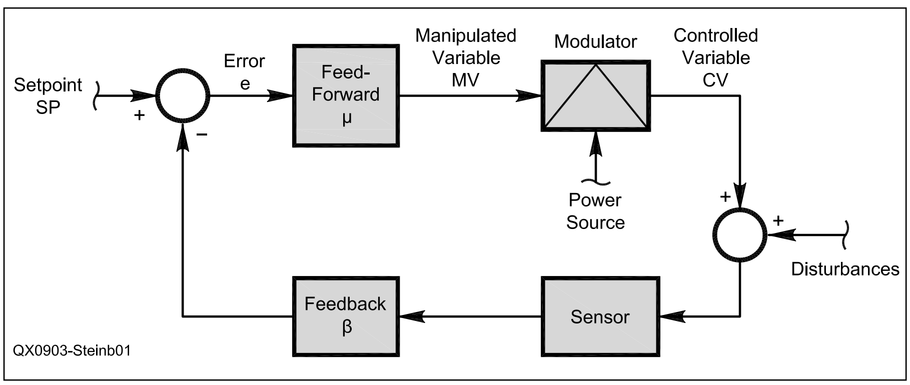


Figure 1 — This diagram of a generalized feedback loop illustrates the use of a sample taken at the loop output and fed back to the loop input, where it is subtracted (corrective feedback) to modify the input signal.

[1]Notes appear on page 5.

to refer to government (the Latin cognate is *gubernator*).[3] [*Merriam-Webster's Collegiate Dictionary*, Tenth Edition, defines cybernetics "pilot, governor" and as "the science of communication and control theory that is concerned especially with the comparative study of automatic control systems (as the nervous system and brain and mechanical-electrical communication systems)." — Ed.] It appears in the Bible, where it is translated as *shipmaster*, and our own André-Marie Ampère (1775-1836) wrote of *la cybernétique* (with Plato's connotation).[4] The first thorough mathematical study of feedback in governors and moderators was written in 1868 (about the time that young Tom Edison was a telegraph operator in Cincinnati) by none other than James Clerk Maxwell (1831-1879).[5]

Now, the first practical use of electricity was in telegraphy (Maxwell died in the year that Edison invented the incandescent light). Signal losses in long lines were overcome by using sensitive mechanical relays, but mechanical repeaters were impractical for telephony. The introduction of Edison / Fleming / De Forest electron-tube amplifiers made long-distance telephony possible; early tube repeaters, however, were marginally stable and produced distortion that was multiplied with each repeater. An improved amplifier was greatly desired.

Western Electric / Bell Laboratories must have been an interesting place to work in the early 1900s! In 1919, Edwin Henry Colpitts (1872-1949), the research branch chief, modified the design of the inductively coupled oscillator that his colleague Ralph Vinton Lyon Hartley (1888-1970) invented in 1915; it was patented in 1920 as an *Oscillation Generator* (Colpitts eventually became vice president of Bell Labs). Another of Hartley's associates was Harold Stephen Black (1898-1983), who was struck with an idea for solving the repeater amplifier problem while riding a ferry to work in mid-1927 (so he relates).[6] The plan was to compare a sample of the amplifier's output to the input signal, and to correct instantly for any inaccuracies in the output, trading unused gain for fidelity and stability. His invention was derided for several (invalid) reasons:

1. Signals would go around and around within such a device, resulting in "singing" and rendering the scheme useless;

2. It was unthinkable to throw away any of the hard-earned gain of the newly-available vacuum tube;

3. Such esoteric ideas came only from the doctors of the Research Department, not simple bench engineers.

With reluctant support from management, Black was permitted to work on stabilized feedback amplifiers, and by the end of the year he had a working design whose distortion was 50 dB lower than conventional repeaters. He applied for a patent in 1928, but the patent was not granted until 1937, partly because his application was treated as if it had been one for a perpetual motion machine. (See Note 6.) Black retired from Bell Labs in 1963, having received many honors, awards, and nearly 300 patents.

Automatic control engineering was further advanced by two of Black's colleagues at Bell Labs, Harry Nyquist (1889-1976) and Hendrik Wade Bode (1905-1982). Nyquist diagrams determine if a loop will be stable or not, and they have similarities to the early charts of another Bell Labs staff member, Phillip Hagar Smith (1905-1987); Bode's magnitude / phase plots predict how close a loop is to instability. Like Smith Charts, these graphical methods are still in use.

The advent of World War II found Bell Labs working on an analog computer that aimed antiaircraft guns by calculating and predicting the trajectories of incoming enemy aircraft. The amplifiers in their T10 gun director, however, were too slow for real-time use, and improvements were again needed. George A. Philbrick (1913-1974), a physicist at MIT, became involved with the T10 amplifiers, and enlisted Loebe Julie, a brand new electrical engineer working at Columbia University, to develop a compact differential input (Black's amplifiers had only an inverting input) amplifier fast enough to perform the necessary mathematical operations. Thus, Julie became the inventor of the operational amplifier, and the improved M9 gun director, coupled with MIT's SCR-584 radar, saved many lives.[7]

In 1952, George A. Philbrick Researches, Inc. unveiled the world's first commercial operational amplifier, the K2-W, which used two 12AX7 high-mu dual triodes and dissipated about 5 W. The Bardeen / Shockley / Brattain transistor (1947 … Bell Labs, again), an outgrowth of wartime radar crystal detectors, made possible more compact, lower power op amps like Philbrick's modular PP65. Jack Kilby's integrated circuit (1958) led to monolithic op amps like Bob Widlar's µA702 and µA709, and Dave Fullagar's 1968 landmark µA741. The circuitry about to be described is greatly indebted to all of the aforementioned!

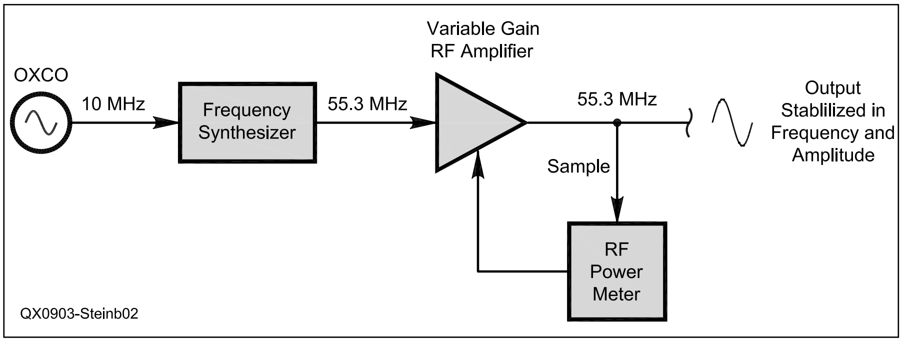


Figure 2 — This block diagram shows one way the four modules that will be described in this series of articles can be interconnected.

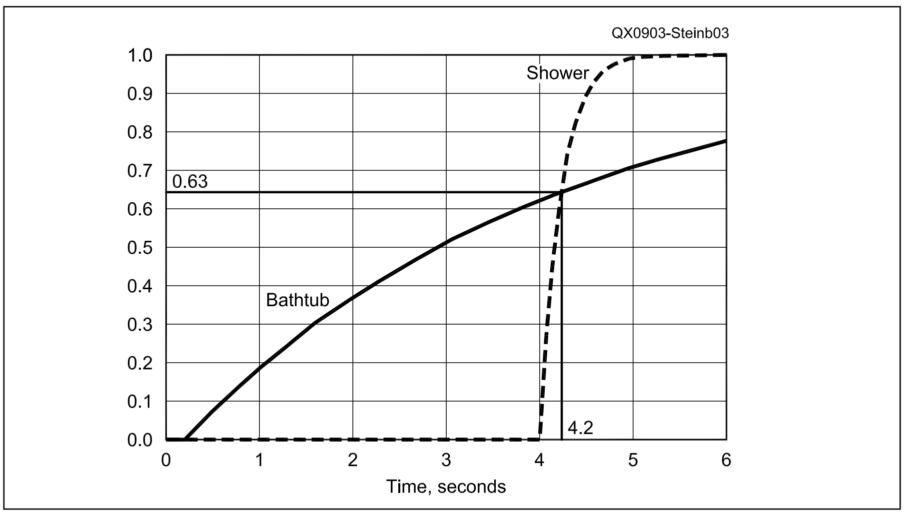
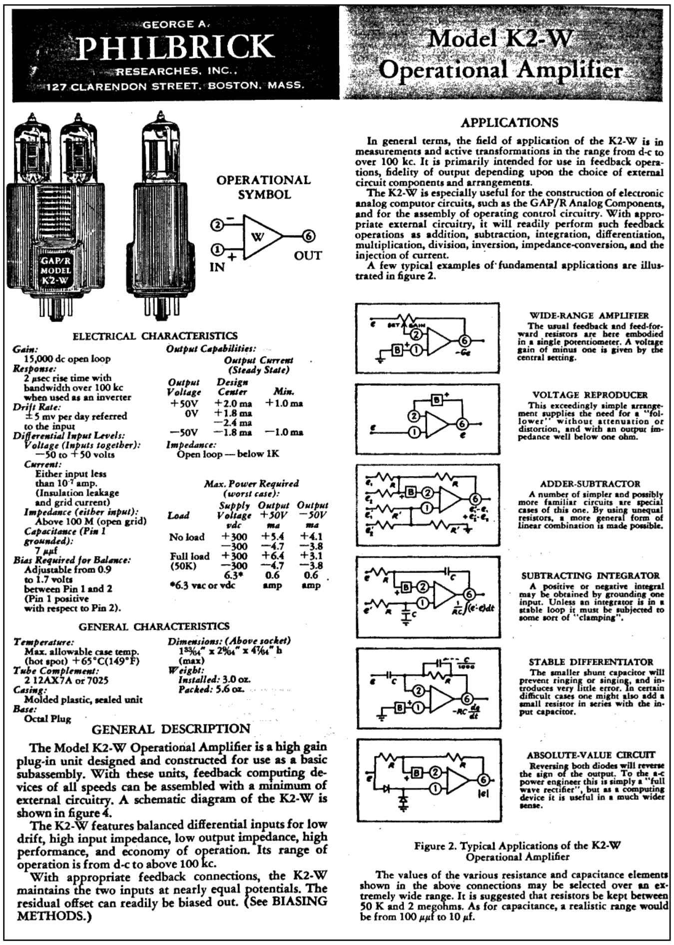


Figure 3 — This graph illustrates the concept of feedback by considering the process by which the water temperature changes in a bathtub full of water when the hot water is turned on full. This is compared to the water temperature in a shower when the same change is made.



GEORGE A.
PHILBRICK
RESEARCHES, INC.,
127 CLARENDON STREET, BOSTON, MASS.

Model K2-W
Operational Amplifier

OPERATIONAL
SYMBOL

② – W ⑥ OUT
① + IN

GAP/R MODEL K2-W

ELECTRICAL CHARACTERISTICS

Gain:
15,000 dc open loop
Response:
2 μsec rise time with bandwidth over 100 kc when used as an inverter
Drift Rate:
±5 mv per day referred to the input
Differential Input Levels:
Voltage (Inputs together):
−50 to +50 volts
Current:
Either input less than 10^{-7} amp. (Insulation leakage and grid current)
Impedance (either input):
Above 100 M (open grid)
Capacitance (Pin 1 grounded):
7 μμf
Bias Required for Balance:
Adjustable from 0.9 to 1.7 volts between Pin 1 and 2 (Pin 1 positive with respect to Pin 2).

Output Capabilities:

Output Current (Steady State)

Output Voltage	*Design Center*	*Min.*
+50V	+2.0 ma	+1.0 ma
0V	+1.8 ma	
	−2.4 ma	
−50V	−1.8 ma	−1.0 ma

Impedance:
Open loop — below 1K

Max. Power Required (worst case):

Load	*Supply Voltage vdc*	*Output +50V ma*	*Output −50V ma*
No load	+300	+5.4	+4.1
	−300	−4.7	−3.8
Full load (50K)	+300	+6.4	+3.1
	−300	−4.7	−3.8
	6.3*	0.6 amp	0.6 amp

*6.3 vac or vdc

GENERAL CHARACTERISTICS

Temperature:
Max. allowable case temp. (hot spot) +65°C(149°F)
Tube Complement:
2 12AX7A or 7025
Casing:
Molded plastic, sealed unit
Base:
Octal Plug

Dimensions: (Above socket)
1 33/64" x 2 9/64" x 4 7/64" h (max)
Weight:
Installed: 3.0 oz.
Packed: 5.6 oz.

GENERAL DESCRIPTION

The Model K2-W Operational Amplifier is a high gain plug-in unit designed and constructed for use as a basic subassembly. With these units, feedback computing devices of all speeds can be assembled with a minimum of external circuitry. A schematic diagram of the K2-W is shown in figure 4.

The K2-W features balanced differential inputs for low drift, high input impedance, low output impedance, high performance, and economy of operation. Its range of operation is from d-c to above 100 kc.

With appropriate feedback connections, the K2-W maintains the two inputs at nearly equal potentials. The residual offset can readily be biased out. (See BIASING METHODS.)

APPLICATIONS

In general terms, the field of application of the K2-W is in measurements and active transformations in the range from d-c to over 100 kc. It is primarily intended for use in feedback operations, fidelity of output depending upon the choice of external circuit components and arrangements.

The K2-W is especially useful for the construction of electronic analog computor circuits, such as the GAP/R Analog Components, and for the assembly of operating control circuitry. With appropriate external circuitry, it will readily perform such feedback operations as addition, subtraction, integration, differentiation, multiplication, division, inversion, impedance-conversion, and the injection of current.

A few typical examples of fundamental applications are illustrated in figure 2.

WIDE-RANGE AMPLIFIER
The usual feedback and feed-forward resistors are here embodied in a single potentiometer. A voltage gain of minus one is given by the central setting.

VOLTAGE REPRODUCER
This exceedingly simple arrangement supplies the need for a "follower" without attenuation or distortion, and with an output impedance well below one ohm.

ADDER-SUBTRACTOR
A number of simpler and possibly more familiar circuits are special cases of this one. By using unequal resistors, a more general form of linear combination is made possible.

SUBTRACTING INTEGRATOR
A positive or negative integral may be obtained by grounding one input. Unless an integrator is in a stable loop it must be subjected to some sort of "clamping".

STABLE DIFFERENTIATOR
The smaller shunt capacitor will prevent ringing or singing, and introduces very little error. In certain difficult cases one might also add a small resistor in series with the input capacitor.

ABSOLUTE-VALUE CIRCUIT
Reversing both diodes will reverse the sign of the output. To the a-c power engineer this is simply a "full wave rectifier", but as a computing device it is useful in a much wider sense.

Figure 2. Typical Applications of the K2-W Operational Amplifier

The values of the various resistance and capacitance elements shown in the above connections may be selected over an extremely wide range. It is suggested that resistors be kept between 50 K and 2 megohms. As for capacitance, a realistic range would be from 100 μμf to 10 μf.

Here is the data sheet for one of the first operational amplifiers — the K2-W by George A. Philbrick Researches, Inc.

A K2-W operational amplifier and some more modern op amps are shown along with Gary Steinbaugh's homebrew spectrum analyzer.

Providing a thorough explanation of the modeling and design of control systems is, as they say, beyond the scope of this paper. Manipulating the differential equations that result whenever a loop is closed is far from a trivial task, even when streamlined by transforming into Pierre-Simon Laplace's notation. Nevertheless, automatic control is a fascinating subject that I would encourage you to investigate further. I hope that the following discussion will help to familiarize you with some of its high points.

Figure 1 diagrams a generalized feedback loop (in homage to Black, it uses his original μ-β symbology; the alert reader will find similar subtle tributes to others within these articles). By feeding back a sample of the output and subtracting it from the original signal (the "setpoint"), an error signal results, which is used to modulate the output and thus correct it for any inaccuracies. Note the intrusion of disturbances, the primary reason that feedback is necessary. Disturbances (also called noise) may invade at multiple places, but they can be lumped mathematically into one point, as shown.

Figure 2 illustrates one way that the modules described in these articles may be interconnected. This system is chock full of feedback loops (ergo the "cybernetic" appellation), and the main loop in each module will be diagrammed to show its relationship to Figure 1.

Not every system (a control engineer would use the term "process," or, more picturesquely, "plant") is as easy to control as another (some are simply not controllable by any practical means). As an example, imagine that you are soaking in a bathtub, with a constant inflow of warm water. You have determined that the water is not warm enough for your liking, and you give the hot water knob a sharp clockwise twist. The water begins warming after a short time delay, and the temperature rises slowly (the time constant), tapering off to a new level; this response to a step change is shown in Figure 3. Note the time delay of 0.2 seconds, and the time constant (the time to reach 63% of the ultimate change) of 4 seconds:[8] the controlled variable (temperature) of this process is easy to control.

Now, instead of a bathtub, imagine standing in a shower that is not warm enough. You twist the knob as before, but no temperature change is produced during the time that the hot water travels up a pipe to the shower head, but when the hot water reaches your skin, the temperature changes with painful abruptness. In Figure 3, note the time delay of 4 seconds, and the time constant of 0.2 seconds: the controlled variable of this process is hard to control (the ratio of time delay to time constant is representative of the level of difficulty). If you react to this scalding by twisting the knob completely counterclockwise, you will have no relief for another 4 seconds, at which time the chilly spray compels you to spin the knob clockwise again, and so it continues. You now have an intimate understanding of the mechanism of oscillation!

In the second part of this series, both the shower and bathtub responses will be employed to implement the first construction project, a low distortion, ovenized, crystal-controlled oscillator.

Notes

[1]Douglas R. Hofstadter, *I am a Strange Loop*, New York: Basic Books, 2007.

[2]Quantum mechanics has determined that the awareness of an observer affects electron motion. Probably unrelated, but who knows?

[3]Norbert Wiener, *Cybernetics or Control and Communication in the Animal and the Machine*, Cambridge: The MIT Press, 1947.

[4]Revelation 18:17.

[5]James Clerk Maxwell, "On Governors," London: *Proceedings of the Royal Society*, No. 100, 1868.

[6]Harold S. Black, "Inventing the Negative Feedback Amplifier," *IEEE Spectrum*, Vol 14, December 1977. (Personally, I prefer the term corrective feedback; negative feedback sounds so … negative.)

[7]Robert Buderi, *The Invention that Changed the World*, New York: Simon & Schuster, 1996.

[8]Evaluating Newton's Law of Cooling / Heating at $t = \tau$, $1 - e^{-1} = 0.632$.

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 for soft tissue ablation. Gary is also a Certified Flight Instructor and a semi-pro musician.

QEX

Roger Monroe, K7NTW

PO Box 236, Clearlake, WA 98235; k7ntw@wavecable.com

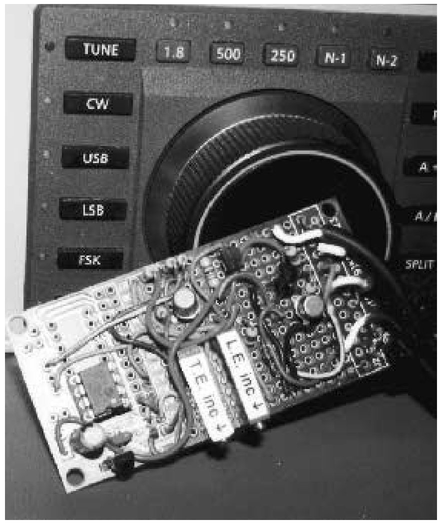
Some Assembly Required — How to Write a Simple Microcontroller Program

Interested in learning how to program a microprocessor? This introduction will help you get started.

Every month when *QST* arrives, I look forward to reading the construction and technical articles. What a great hobby we share, that we can be that involved with any aspect of it. Truly a community of folk dedicated to building up every Amateur Radio enthusiast. Learning by doing has many benefits; it is fun and not least is, what we learn from our first QSO or first building project is carried to the next and so on.

While a microcontroller has been the central component of many past *QST* articles, the complexity of their internal programs required at least moderate to proficient programming skills in order to understand the code. This article will attempt to point the amateur wanting to learn programming on a path that will aid being successful using a microcontroller to do a simple albeit useful chore in his or her station. The microcontroller is a good choice, if not the best choice, for even a simple application. This is due in large measure to the compactness of the device. That compactness includes not only physical size but also, processing speed, input-output logic, analog to digital converters (ADC), level comparators and a list of features that keeps going and growing. These added peripherals in the device differentiate them from microprocessors. Many cost less than the popular LM555 timer. The microprocessor in your desktop computer has a generalized CPU and can run many different programs. The embedded system does one task with a microcontroller selected for its special features.

Past articles have focused on getting the program (operating instructions) for the processor into the device so it is usable. See Figure 1. This makes it possible to duplicate the original design. These articles have not dealt with actually writing a program in detail. My background with programming microcontrollers has been as a hobbyist. Getting started programming is this article's purpose. Most information on assembly language assumes that you have had some previous experience with a microprocessor or microcontroller, but this may not be true for you. My hope is that I can explain the microcontroller to the point that what you read about programming makes sense. One of the neat things about modern electronics is the building block concept. Once the builder



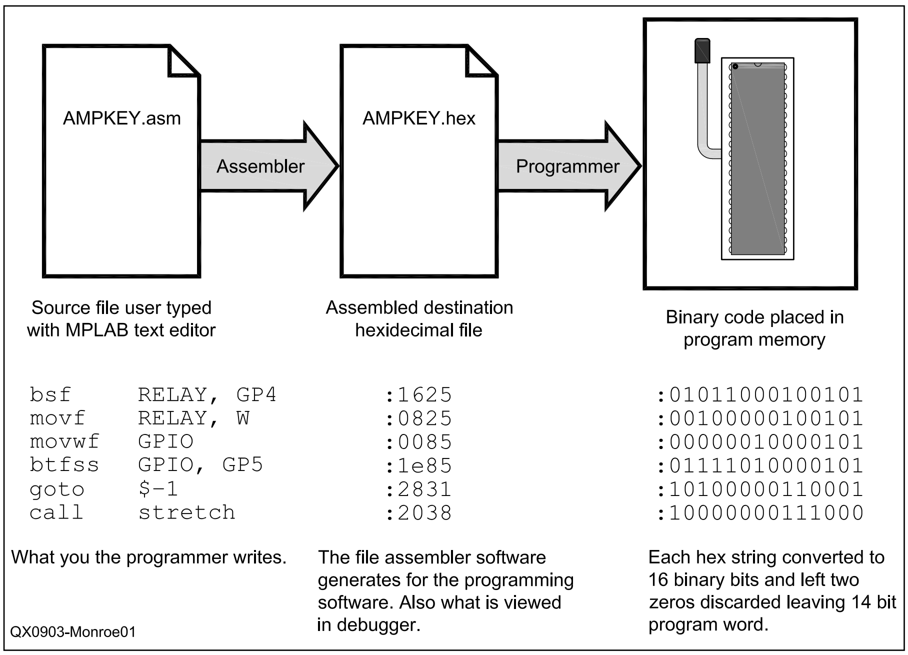


Figure 1 — This diagram shows the process of an assembly language program being converted to a hexadecimal program file, which is then loaded into the microprocessor as binary code. The colon before each set of numbers in the list show that the numbers are strung end to end.

understands the rules for interconnecting the blocks, he or she no longer has to understand every detail in every circuit. The microcontroller is just such a device.

Putting the program into the microcontroller, although important, is only one step in the process of implementing a microcontroller. The process begins with a need or idea for something for which we think a microcontroller would be suitable. Call this the high-level design phase. Here we consider all the things we think we may need, and set goals for our project. Since a microcontroller typically handles information and data in sequence it would make sense to draw a flow chart. We may have to juggle tasks, so everything looks like it is happening at once. From the flow chart and our list of goals we can write some instructions for our project. Then we can assemble those into the machine code. It is wise to test code using debugging software first, but if we wanted, we could "burn a chip" and see if it worked. Chances are it is not going to work like you expected the first time. Then you go back and make changes and repeat the testing. It may take you several tries to get things right. You will be learning quickly, and with good software tools, you will find why your code doesn't run as expected, and then you will be able to fix it. Just understand that this is normal for programming a microcontroller.

Let's all take a deep breath and dive in! There are a large number of ideas, skills, and materials to be pulled together in one place in order to start the writing process. Furthermore, there is a vast range of experiences and learning backgrounds, making it more difficult to identify each reader's status as a beginning programmer. It is useful to have some exposure to a number of varied subjects related to electronics as you begin the foray into microcontroller programming. Some that may be helpful are identified in Figure 2.

For the beginner, some encouraging words are needed. You will be exposed to manufacturer's material used by embedded systems designers who have had the benefit of formal courses of study and/or extensive applied experience. If you understand a fraction of what you read from those documents then take it as a positive sign. New introductions of microcontroller products point to it being state of the art technology. and I hope you will enjoy taking part in this technology.

There is not room enough in one article to describe how to design and use a microcontroller circuit in every detail. What we can well cover, however, is an overview of the steps involved in getting a simple electronic function built into one typical device. The following may not be the best way to do the job at hand but it will, I hope, convey the steps involved. If what you build and program accomplishes your goal, then that is all that matters. The good news is the cost. There is a wealth of free information on the Internet. The hardware does not need to be expensive. In fact, you don't even need any hardware to start learning to program. A bit of study before buying extensive amounts of hardware will allow you to select what is best for your intended purpose.

Perhaps you don't have any ideas yet for a microcontroller project, but want a more detailed understanding of how they work. Once you have your first program running, you will no doubt want to write more. When you learn what microcontrollers can do and how to write programs for them, you will

Nice to know subjects to ease learning microcontrollers and making them useful in your shack.

- Using Microsoft Windows
  - ✓ File manipulation (saving, save as, extensions, naming, moving between files and folders
  - ✓ Text editors cut, copy, and paste within and between documents.
  - ✓ Desktop control
  - ✓ Task bar, drop down menus, Mouse selecting, Keyboard short cuts
- DC circuit theory
- Digital techniques
  - ✓ Number systems, binary and hexadecimal
  - ✓ TTL and CMOS
  - ✓ Logic gates, truth tables
  - ✓ Flip-flops, registers, counters
  - ✓ Interfacing between logic families
- Diode and transistor
  - ✓ Biasing and conduction state (cutoff linear saturated)
  - ✓ As buffering and level conversion elements
- High level computer language such as BASIC

Recent editions of *The ARRL Handbook* cover circuit subjects in the detail needed.

Figure 2 — It is helpful to have some exposure to, and knowledge of computers, electronics theory and perhaps some experience programming in a higher level language such as BASIC.

Transceiver to Amplifier Relay Interface

This sidebar chronicles my efforts to interface a Ten-Tec OMNI VI+ transceiver and a Heathkit SB-220 linear amplifier for quick transmit receive (QSK) operation. The OMNI VI+ is a popular transceiver, but as you will see, is a bit more difficult to connect for QSK control of an HF amplifier than are some other transceiver brands.

When February 2008 *QST* arrived I read "External Full Break-In Transmit-Receive Switch for HF Amplifiers" by Phil Salas, AD5X, with particular interest.[1] I had intended to install high-speed vacuum relays in my SB-220 amplifier, but the project had stalled. His article prodded me into action. Read on to find out how you can add QSK to your OMNI VI+ station. I, like Phil, am interested in working DX on CW but even for SSB the modifications are worth the effort. With QSK relay keying, VOX operation becomes silky smooth. When the relays cannot be heard and the VOX delay is shortened, the result is like talking on a cellular telephone. Even if you don't have a Ten-Tec radio perhaps you will be able to use a microcontroller in your next station control project.

If hams had unlimited budgets for new radios the problem could be easily solved by purchasing the manufacturer's matching linear amplifier along with the new QSK ready transceiver. Often the amplifier costs more than the transceiver itself, however. It therefore

[1]Notes appear on page 15.

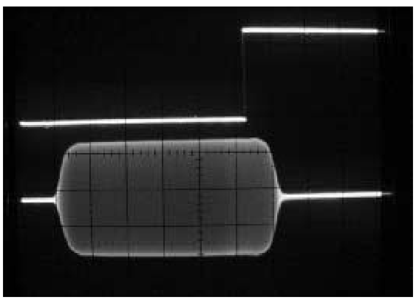


Figure A — The oscilloscope display of 22 wpm dots shows the amplifier key line and transmitted output from a Ten-Tec Omni VI+ transceiver. Notice that there is about a 10 ms delay after the line is keyed until the transmitter output begins, but the transmitter output also continues for about 10 ms after the line is unkeyed. The TX OUT-TX EN pulse (test point A) is connected to Channel 1, The transceiver RF output is connected to Channel 2. Oscilloscope horizontal sweep is set to 10 ms/cm. The Channel 1 Trigger is set for negative slope.

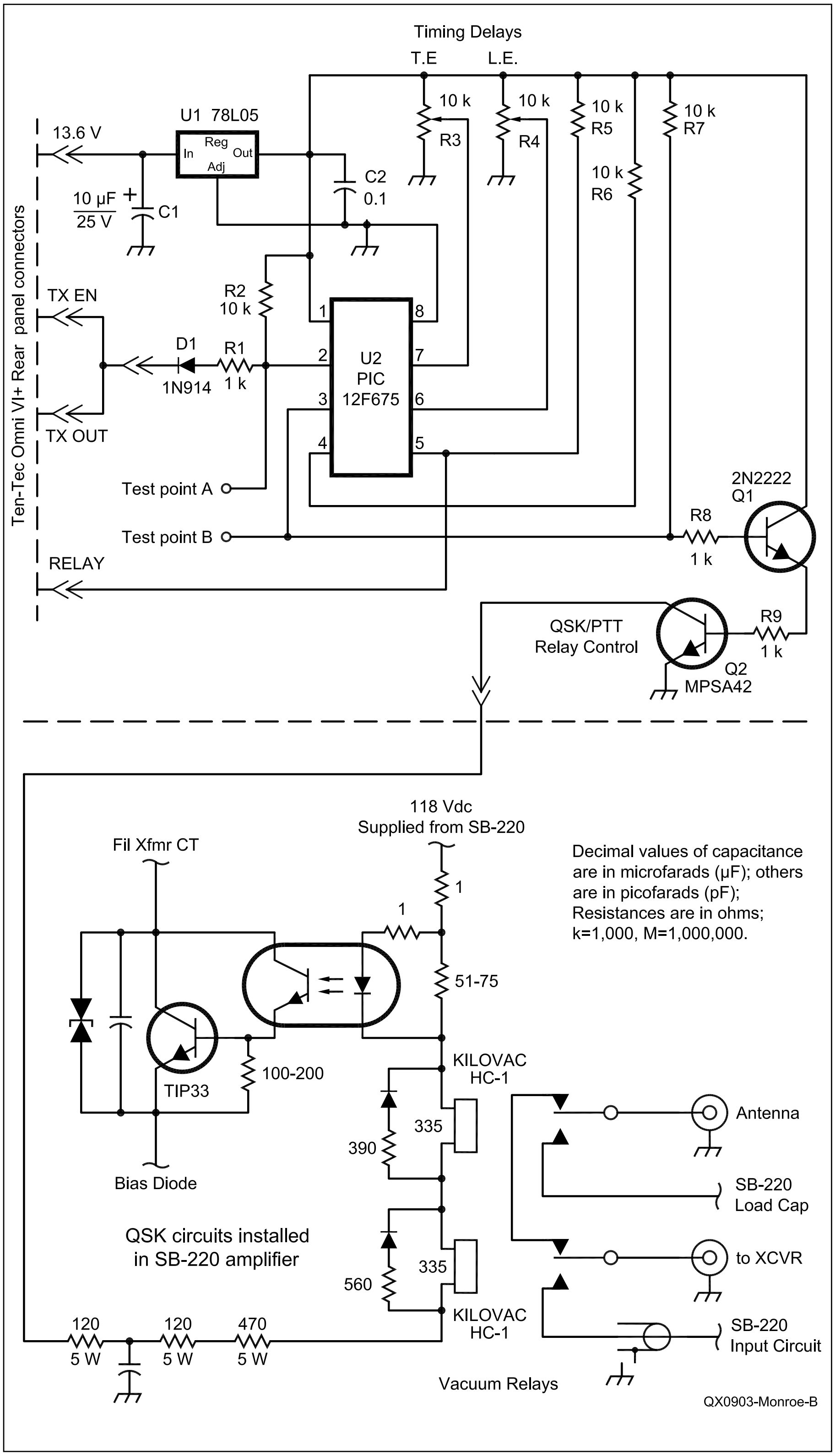


Figure B — Here is the QSK timing circuit that I built for my Ten-Tec Omni VI+ transceiver, to key my Heath SB-220 amplifier.

is not uncommon for hams to wire their existing amp for push to talk operation (PTT) or sideline the amplifier and run barefoot. Why shouldn't we be able to have our QSK and QRO it too? If we look at the problem in a technical and practical manner some possible solutions appear.

Inside the transceiver, the task of switching from receive to transmit or vice versa (T/R) quickly has been meticulously thought out by the design engineers if they want their design to be sought out by the discriminating ham. Likewise, in an interest to sell a matching amplifier, the engineer has designed the two to work together harmoniously. Lighting fast T/R response (QSK) has always been a premium feature in a ham station. I think partially because of manufacturing costs and partially because of compatibility issues, after market amplifiers seldom have high speed T/R circuitry.

To break down the process of going QSK we have to look at three things. First is the transceiver QSK capable? Second does the amplifier have high speed relay switching? The third question becomes how to hook them together. This article is going to assume the transceiver is QSK ready. Richard Measures, AG6K, has great information on converting many amplifiers to QSK operation using vacuum and reed relays along with high speed bias switching.[2] Phil Salas' Feb 2008 *QST* article shows you how to do this without "popping the hood" on the amplifier. In Phil's article he was able to use his transceivers through an Ameritron ARB-704 interface. I recommend you read Phil's article, as well as one by Richard Measures, AG6K; "The Nearly Perfect Amplifier," *QST*, Jan 1994. In addition, both these hams have excellent Web sites.

I was about to hook up my OMNI VI+ and SB-220 amplifier for QSK operation, but first thought, I'd better see how it looked on an oscilloscope. See Figure A. In the OMNI the TX out jack is jumpered to the TX EN jack when the radio is used without a Ten-Tec amplifier. When the transceiver is keyed, the TX OUT line is grounded through a transistor, bringing the positive TX EN low. After passing through further circuitry the RF is increased smoothly to control unwanted clicks and splatter. But what startled me was that RF did not begin to flow for 10 ms after TX EN went low. Even more surprising was RF flowing for another 10 ms after the TX EN line went high. It was clear to me that this pulse would not work to directly key the amplifier, because the radio and amplifier would be producing gobs (an unscientific, unstable quantity) of RF when the relays dropped.

The pulse would work if it were both delayed and stretched. I thought briefly about making a timing circuit using 4001 CMOS gates. I almost immediately thought of using a microcontroller instead, when I realized that the gate timer approach would take at least two ICs. I had used Microchip's PICmicro microcontrollers before, so I was eager to see how one would work controlling the high-speed vacuum relays I was considering putting in my venerable SB-220 amplifier.

When designing something that is going to be interconnected with various pieces of equipment, spend some time thinking about how it is all going to work together. All the modes and combinations of modes need to be considered. How about powering up and down sequences? What transient (quick changes) might occur between the vacuum tube amplifier and the solid state transceiver output, as well as relay coil inductance? How would I address the different voltage levels between the transceiver keying circuits, my keying adapter, and the amplifier? I decided that using the A/D converters inside the 12F675 PIC device would make it possible to precisely set the transceiver QSK relay times. The internal clock in the PIC is 1% accurate. The program was only going to be reading two potentiometer values, delaying and stretching pulses, so a single main loop of code would do the job.

Refer to schematic diagram of Figure B. The vacuum relay coils are

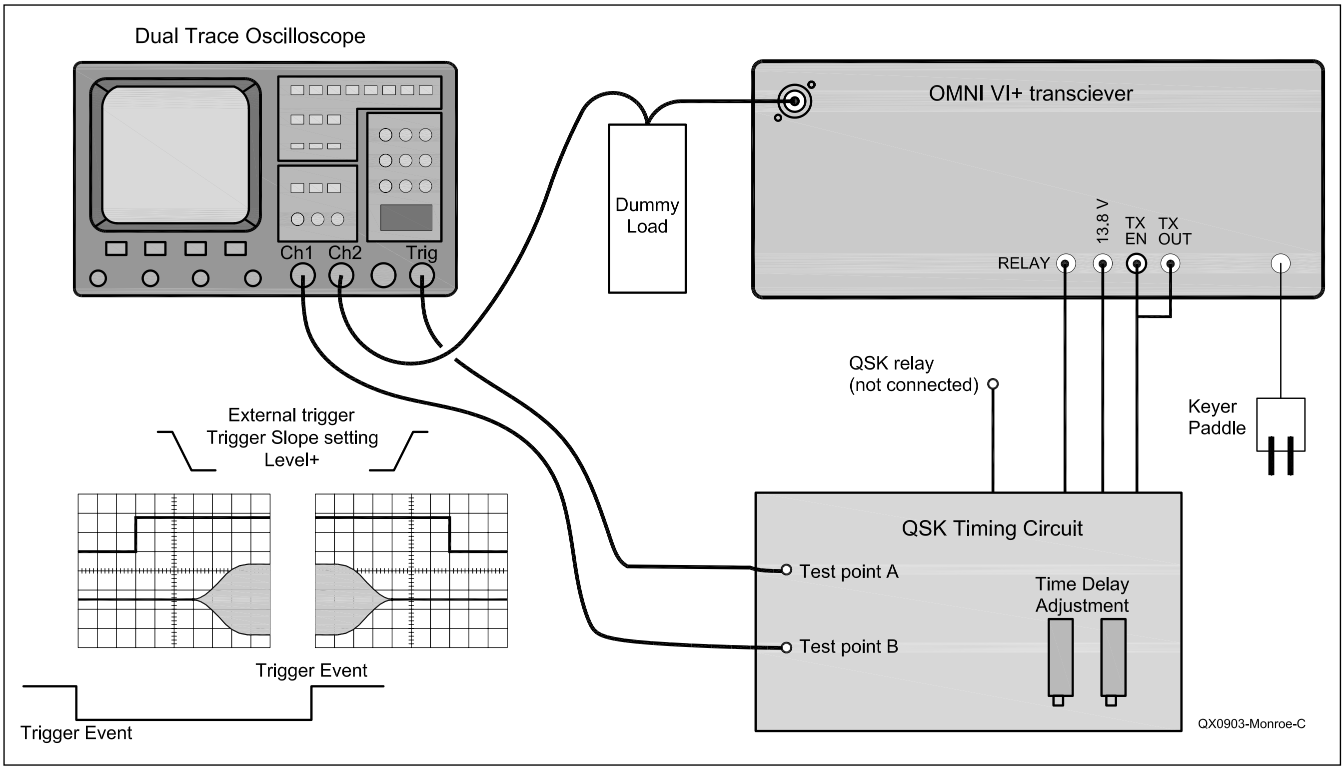


Figure C — This diagram shows the test setup I used to verify the correct operation of the timing circuit.

connected in series with resistors to a positive voltage as high as 110 to 150 V. When the series circuit is grounded through Q2 HV switch transistor, 80 mA of current activates the relays. I selected an MPSA42 transistor for this task. Q1 buffers the 5 V output of the PIC to forward bias Q2. The diode between the OMNI TX OUT/ TX EN jumper and U2 input (pin 2) is reverse biased, while the transceiver is receiving, and is needed, because the OMNI internal keying transistor is a 13 V circuit while the PIC has a 5 V CMOS logic level. When it is pulled to ground in the transceiver the diode conducts, dropping the PIC input to logic low. The controller program has been waiting for the input to go low. Now it marks time until the leading edge (L.E.) value established by the analog to digital converter has expired. Then the output (U2 pin 4) is brought high. This logic high from pin 3 of U2 energizes the QSK relays in the amplifier. Now the input waits for un-key (TX OUT) to go high. Then the controller waits using the value established by the trailing edge (T.E.) potentiometer. After this added time, RF has stopped flowing from the transceiver, and it is time to de-energize the relays. The program stays in a loop waiting for the next TX event. The push to talk (PTT) relay in the OMNI brings pin 5 of the PIC low. QSK and PTT are a logic OR function in the PIC program.

Building the Circuit

I used a PICPROTO 8 prototyping board in developing this project.[3] I used cables with RCA plugs pre-attached on one end and the other end soldered to a DB9 connector on the PICPROTO 8 board, giving a neat appearance. Before adjusting the QSK timing circuits for the amplifier, QSK relays should be measured for speed. I used the method recommended and detailed in Phil Salas' Feb 2008 article. See Figure C for QSK timing adjustments. Adjust the trim pots using a triggered oscilloscope, connect channel 1 on test point B (pin 3 U2) and the channel 2 to sample the transceiver RF output with the power turned down to QRP level and feeding a dummy load. Connect an external trigger probe to test point A. My Tektronix scope wanted to false trigger with the OMNI on 40 m. Changing to 160 m cured that problem. Do not connect the amplifier yet. Adjust the leading edge to provide for the pull-in time plus a margin of safety before the RF starts. The trailing edge is set so the drop out will occur shortly after RF stops flowing. See Figure D. The leading edge delay is adjustable up to 7 ms and the trailing edge stretch up to 13 ms. Now go ahead and connect the amplifier and the QSK relay output to the QSK relays. Start with RF power at minimum while confirming proper relay action. When you are satisfied everything is working smoothly, advance to full power and enjoy some smooth QRO QSK operation.

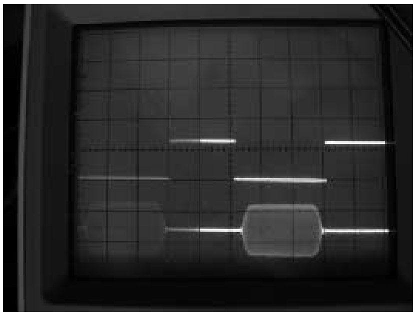


Figure D — A photo of the oscilloscope screen shows the correct operation of the QSK timing circuit. The AMP RELAY pulse (test point B) is connected to Channel 1 and the RF output is connected to Channel 2. The oscilloscope horizontal sweep is set to 10 ms/cm, with Channel 1 triggering set to positive slope.

As a consequence of what I found and have read, I think a ham wanting to add QSK capability should have access to a triggered oscilloscope to confirm levels and timing before applying RF to the amplifier input.

As with any project involving significant voltages, safety should be paramount. If you question your abilities to complete a project such as this I would recommend you enlist the help of a more experienced ham to check your work. In this same light, I am indebted to Phil Salas, AD5X, for encouraging me to write this article.

Programmed 12F675s are available from the author for this project. The PIC program file is available for download from the ARRL *QEX* Web site.[4]

think of many possible uses for a microcontroller. We will pick one microcontroller and use it to explain how it functions and the steps involved in writing a program for it. For your first program, a simple task such as controlling a light emitting diode (LED), or reading some switches will build confidence while you learn how each instruction works. When I build hardware, I like to build a section and test it, and get it working before moving on to the next. I find this approach works for me when writing assembly language programs too.

We will develop our embedded system using Microchip PIC programming software.[5] It is called MPLAB IDE, and it runs on a Microsoft *Windows* computer The IDE stands for integrated development environment, which means the text editor, assembler, and device programming software, are combined into one program. It is available free from the Microchip Web-site. It has many good features, not the least is free upgrades as new devices are added to Microchip's product line. The Microchip Web site is a gold mine of information (**www.microchip.com**).

A big part of learning to use microcontrollers involves getting to the information you need from the very large body of data that exists. Online at Microchip are "Web Seminars," viewable on line or by download. Even if you have other microcontroller experience you will benefit from Microchip's presentation. A good voiced presentation is more fun for me than just reading about it. I recommend you view the Introduction to MPLAB IDE and Introduction to Microchip Development tools. These are accessed by going to Training, then E-learning and search for English-Development tools. You may want to download MPLAB IDE onto your computer so you can start learning right away. There is an excellent help window on the MPLAB IDE main screen. Knowing what is available from the help menu will save the time and effort of digging through a data sheet for a needed detail while writing or debugging code.

Assembly language has an immediate relationship to what is happening inside the microcontroller, which helps us see how the hardware and software come together for an understanding of how the program works. There are other good reasons for using assembly language. First we will use it in our description of the internal operation of the PIC. Secondly, computers love numbers! Give a computer a few million binary bits and it will be waiting for more in milliseconds. Give me a few dozen binary bits, however, and I'll be snoozing shortly, thank you! *QST* Technical Editor Joel Hallas, W1ZR, suggests keeping the Scientific Calculator (MS *Windows* Accessories) on the computer desktop for quick number base conversions.

It is necessary to get a clear picture in our head, so things are going to have names so we can relate to their function and stay awake. Being able to call a part of the controller by a name and letting the assembler translate that into numbers, you start to have a "handle" on the code. For example, suppose we needed a digital number that looked like 00101101 to send to the I/O port, and we needed that several places in the program. It would difficult to find our error if we typed it one digit wrong in one place. But if we had assigned it a name

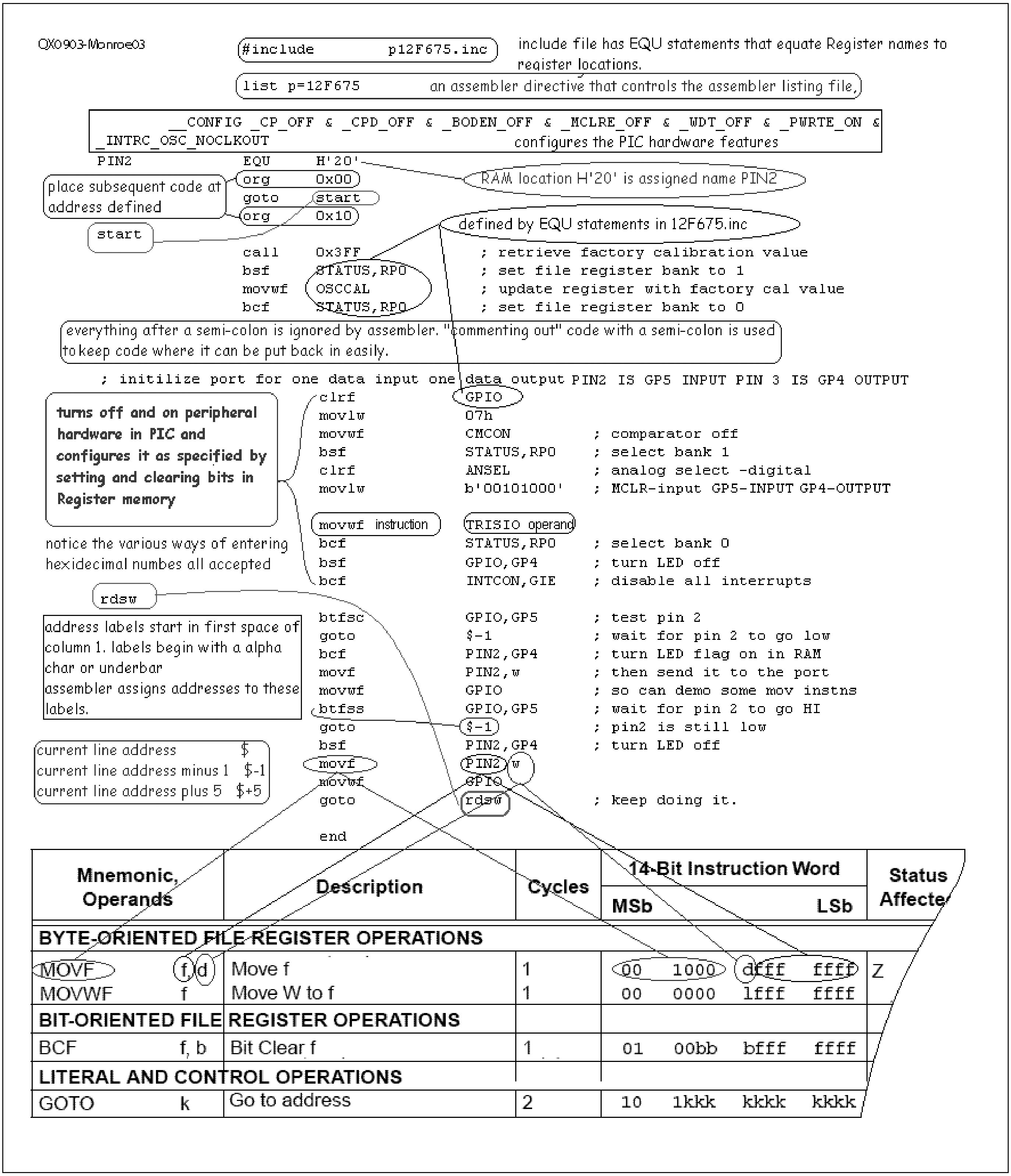


Figure 3 — This drawing identifies some of the Assembly language instructions and explains their operation in a program. This short program controls an LED in response to a switch as described in the text.

such as PORTCF the assembler would get it right every time and if we typed PORTCT by mistake the assembler would tell us where we had erred. To do this we can assign names to numbers in our program using the EQU directive. A directive tells the assembler how to handle data to generate machine code. Include files (.inc) are lists of EQU assignments already for use in MPLAB. There is one for each model device. You then just use the name of the register without worrying about its actual address.

There are several families each having a range of complexity. We will look at a mid-range part. The *PICmicro Mid-Range Reference Manual* gives a general explanation of every section in the family. There are 35 sections to this document, a lot of reading to be sure, but it is useful and understandable. I find it helpful to read through first to just gain a sense of the things to be learned. Then I follow up reading and taking notes, to separate things so they make sense to me.

All processors in the family will share the same instruction set. Instructions are English-like abbreviations, converted to numbers by the assembler, which tell the central processing unit (CPU) what to do. The Mid-range PIC has only 35 instructions, and you can sometimes get by with less! The programmer will use instruction mnemonics that are easier to remember than 14-bit words. Operands are ROM or RAM information with which the instruction works. Look over the example source code (Figure 3) of a very simple program. The first instructions set up the microcontroller by selecting and configuring the peripherals we want. When starting out, if using a number helps you understand what is going on, then that is what you should use. Writing programs is a learning by doing extravaganza; trying out new and different things are the keys to success. So is maintaining a high fun factor.

A typical instruction looks like
MOVF f,d
as it appears in an instruction set list in Figure 3. It tells the assembler to write machine code to tell the processor to move data from the memory register addressed at f and place that data in register d. Look closer at the instruction. We see d is only one bit. That gives two choices; put the data in the working register if the bit is a 0 or back into the register it came from if it is a 1. In a line of source code it looks like.

```
LOD    movf    PIN2,w
```

LOD would be the address in the program code where the instruction is located. MOVF would be the instruction code bits (001000) at that location, PIN2 was previously defined as an address in Register memory where the data to be moved is located, and w was defined as zero in the include file p12F675. This should make more sense once we look at the inside of a microcontroller.

We type instructions and operands using the text editor that is part of MPLAB IDE. It has a nice feature that when you type a valid instruction the text changes color.

In addition to the *Mid-Range MCU Manual*, the programmer needs the Data sheet for the specific device being programmed. There are detailed descriptions, which when read in concert with the *Mid-Range Manual*, fully explain the function of all blocks of the PIC, all electrical, temperature, and timing environments — in short, everything you wanted to know and then some! The 12F675 "data sheet" is 130 pages long. Look through this to get an overview of the information it contains. You may want to refer to the *Mid-Range Manual* as you read the following description.

Inside the Microcontroller

Since the hardware and software work so closely together, it is necessary to relate the two as we go along. We have an idea what program instructions look like. Now is the time to peek inside our microcontroller. See the block diagram in Figure 4. The microcontroller has a lot of circuits that are tied together to process information. The PICmicro is called a reduced instruction set computer (RISC) processor. The processor in a personal computer, because it has a big set of instructions, does a lot with one instruction that a PIC cannot do. The PIC performs many simple instructions, so it is very dependent upon computer memory to place and retrieve program instructions and data. To facilitate this, program memory and data memory are separate. This is called "Harvard Architecture." One advantage is that it allows for a longer program word while using a byte width (8 bits) data word. At the very center of what is happening inside the microcontroller is the arithmetical and decision making logic of the CPU, but because it's totally dependent on data, we will look at those main sources first.

Memory

One of the first lines in a PIC program contains the Configuration Bits; a 14 bit word assigned a memory location well above user program memory, and configured at program time. It is here that, Code Protection, Brownout Detect, Master clear pin function, Power-up timer, Watchdog timer, and the type of oscillator for clock timing is selected. It configures the PIC internal hardware for the specialized function of this application.

It is good to have a clear mental picture of the different memory and registers in the PICmicro before moving on. When we get to the CPU, you'll have a good sense of what the differences are and where stuff is going. We are going to use names to keep things straight. First we will look at program memory. It is called that because it's where the computer instructions reside. It is the memory that we write to when putting the program inside the chip. We write to this memory only when we program it, and we will only read it when the PIC is operational. On our PIC12F675 the program memory is 1024 words long and each word is 14 bits wide. (1 K is computer shorthand for 1024.) It represents a bank of memory. Its data is accessed by a 13 bit address from the program counter. Larger PICs have more than one bank of program memory, and need some additional instructions in the program to move around the different banks. Even with just one bank it is sometimes necessary to learn and use special registers to address different locations in program memory. You will learn the different addressing modes in your study of PIC programming.

The 12F675 uses FLASH program memory as indicated by the F in its part number. Its memory is nonvolatile and retains what we program into it even when its power is disconnected. FLASH memory can be electrically reprogrammed thousands of times, even in the circuit in which they reside. You will see that called in circuit serial programming (ICSP). Parts like the 12C672 (older technology) come in packages with a UV erasing window or without a window (which means they cannot be erased). This type of memory is sometimes referred to as EPROM. A FLASH technology advantage is not buying several expensive windowed devices, so you can be erasing while reprogramming in the debug phase. I have to admit that I liked looking in the tiny window at the silicon I was programming!

The second block of memory is not as big. It is called the File Register. It is static random addressable read write memory (SRAM). In the 12F675 it contains the Special Function Registers (SFR) plus 64 bytes of User RAM. The program instructs the CPU to read and write to this memory. Its address comes from program memory or the File Select Register. SRAM is where variable data is placed. File Register memory is in two definable parts, Special Function and General Purpose. This memory is static memory, which means that when it is powered up the contents in many of its registers will contain random data. Some of the registers are dedicated to specific functions and they can have preselected data written to them at power up. It would be desirable for example to disable interrupts while the PIC is booting up. The peripherals also change the contents of the SFR directly and that is one of the features that make the PIC powerful. What is important is that these

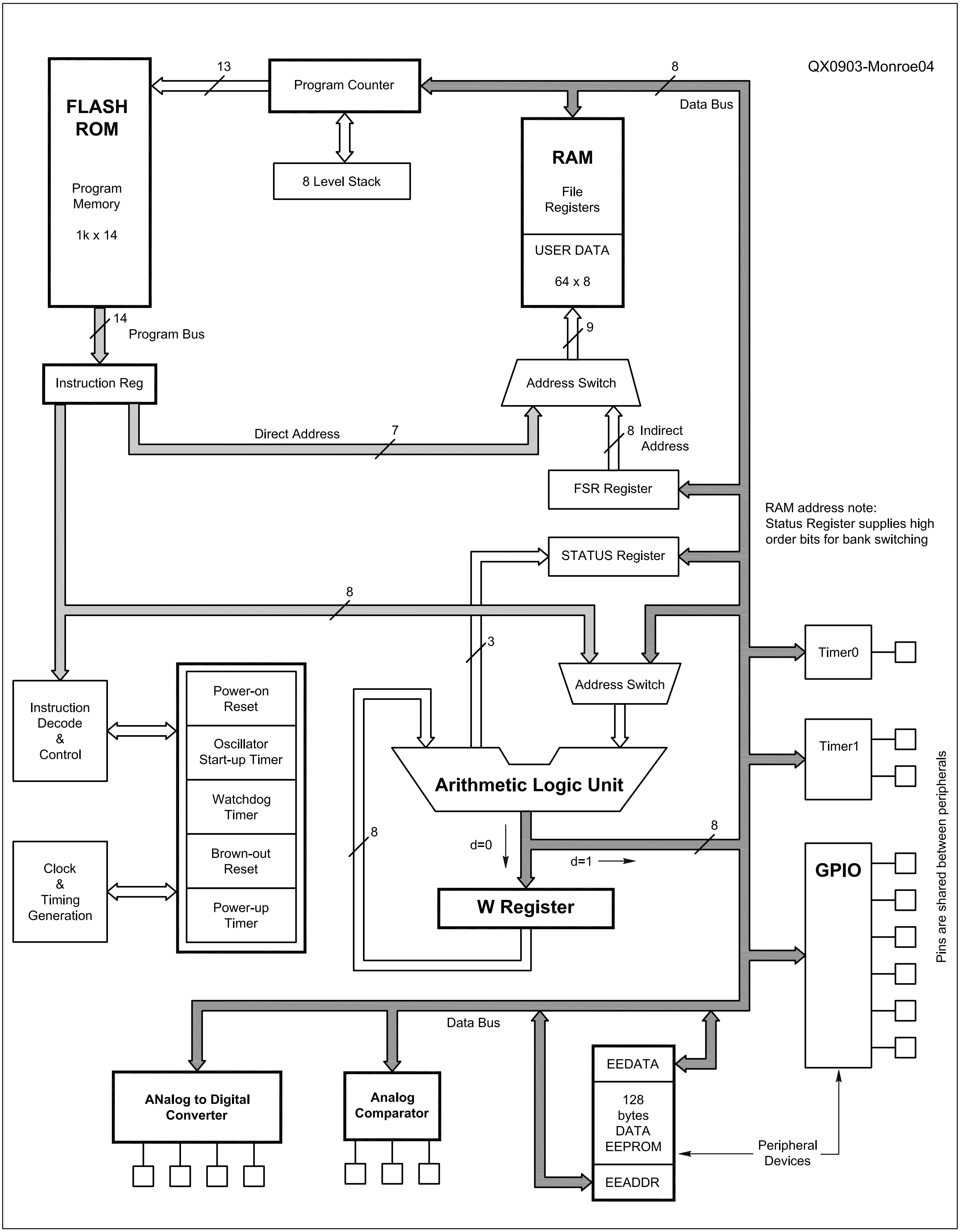


Figure 4 — This block diagram shows the basic operation of the PIC12F675 microprocessor.

registers' content can be affected by the program and by electrical activity on the pins of the PIC itself.

There is another block of memory called EEPROM that can be written to at programming time or while operational. It is a peripheral. It requires a specific series of instructions to access its data. EEPROM can be used for calibration data or anything else that needs to be retained when the PIC power is turned off.

The Oscillator

The PIC contains an internal timing reference or oscillator that connects to timing generating circuits that control the flow of data through the central processing unit (CPU). Each instruction requires 4 or 8 oscillator cycles. Microchip calls these Q cycles to differentiate them from instruction cycles. One instruction cycle will then take one microsecond to complete for a 4 MHz oscillator.

It is nice to know how an instruction works, but not essential in understanding how the program works. By thinking in logical layers we can concentrate on the task at hand, without getting bogged down in the details.

The CPU

The CPU in the PIC consists of three main sections, control logic unit, arithmetic logic unit (ALU) and temporary storage registers. Upon reset, the control will set the program counter to read the first location in program memory. That will access the 14 bit word that contains the first instruction. The instruction decoder will interpret and control the flow of data between the registers and ALU process, in step with the Q cycles. The instruction decoder will examine the contents of the special function registers to further dictate what to do. Data is acted upon in the ALU and then placed in the working (W) register or file register memory as directed by the destination bit in the operand of the instruction word. The program counter may be incremented depending on what instruction was used.

The different blocks of the microcontroller need temporary locations for passing data between them. It's helpful for me to think of a register as a row of toggle switches, the handles remaining where you last put them. The PIC uses them the same way, a sort of memory with a hardwired location (no address) that is shared between the logic blocks. There are registers specific to the processing of instructions. The most obvious is the Instruction Register, with its contents used by the CPU. Programmed memory data appears here, one word at a time. The CPU performs logical functions and passes data as directed by the instruction decoder. What is not instruction is data, and gets directed to where it is needed. Another important register is the W or working register. The name working register says it perfectly, as its data is constantly changing as the program executes. Sometimes programmers refer to this register as the accumulator.

The program counter (PC) is another busy dedicated register. The program counter addresses the program memory. The PC is connected to the Program Counter Stack (Stack) and Program Memory. The Stack is used for returning from an interrupt request or a sub-routine call to continue sequential processing of program instructions, a sort of program place saver. The Status Register, although it has a physical address, and its contents can be read that way, is changed as a result of the computational process. An instruction, whose logic is affected by the three logic bits in Status, will not overwrite those bits. Two other bits in Status are only readable using instruction mnemonics. You will be keeping a close eye on the Status register as you write and debug your programs. Peripheral blocks such as I/O and A/D are connected to their corresponding Special File Registers directly, in addition to being addressable. Other File Registers will be covered later. Because the purpose of the PIC is generally to accept input, process that input, and drive an output, the GPIO registers will be used in almost every program you will write. One of the great strengths of the PIC is the close connection between processing and physical I/O. Now you have a basic understanding of what a PIC is.

Writing the Program

The first program you write, should demonstrate that the assembler software works, and that you understand the rules that the assembler program requires you to obey.

MPLAB IDE is built up around the idea of a "Project" so that all the resources you need are kept organized for your ease of use. The alternative is that you manage different files yourself within MPLAB. The Project Wizard makes using the project method even easier. You are taken step by step through the wizard when you view the "Introduction to MPLAB IDE" Web seminar. The template file contains a lot of comments that explain what the assembler needs to translate your program instructions into machine code. One of the "dirty little secrets" you discover is that, although we only have to learn 35 instructions, we have to learn and use some assembly instructions called directives. This is another excellent reason for starting off using the Project Wizard and adding a template file. The initial directives are all in place and you can start adding your own instructions and building your own simple code and running it in the Simulator right away.

Let us assume you now have MPLAB on your desktop and have created a "First Project" using Project Wizard and a template source file. If that statement is unclear view the Microchip "Introduction to MPLAB IDE" previously mentioned. On the assembler source file (.asm) scroll down to where it says "; remaining code goes here" and type: *here GOTO here*.

Make sure the first "here"' is in the first column with no space in front of it. Go to the File drop down and save the source file changes. Then Build All in the project drop down menu. You have just built an assembly program! Okay, so it doesn't do anything, but what can we expect from a one-line program?

What you see is exactly how easy it is to use some development software. You can use the simulator to run your program on the desktop computer as outlined in the above mentioned Web seminar. Practice setting up "watch windows." View the File registers and don't forget the status bar at the bottom of the screen for the current program counter, Status flags, and W register contents. See if you can figure out what each instruction does and what registers are affected. Read the description for each instruction, under what conditions the Status register contents affect the instruction and what instruction actions do to the Status register in the *Mid Range Manual*. You will learn not only how the instructions work together; but also become a proficient user of MPLAB in the bargain. It's good to print out a copy of the processor specific include (.inc) file so you can relate its *<label EQU value>* statements as you examine program and register memory contents. Notice how bit equate is used in a bit-oriented file register operation.

If you are new to programming, keep your first programs simple. You can use the GOTO instruction to make your program loop. Inside the loop you can have the controller perform different tasks one after the other then return to repeat the same tasks again and again. No doubt you can see how that could be useful if you wanted to generate a custom wave shape. Another useful technique is a subroutine. Suppose you need to provide a delay of 50 microseconds several places in your program. Instead of repeating the code every time it is needed, you can write one routine that uses a timer, and call it when you want it. Did I just hear someone say "Call and Return?" If you can think of breaking down large tasks into a sequence of tiny steps you will have no problem using microcontrollers.

If you can't decide which of two instructions is going to work in your program, type them both in, but put a semicolon in front of one. Assemble your program, and test it. If it doesn't work move the semicolon and try again. The semicolon makes it easier to find that spot in code again. Everybody eventually gets caught changing code in the wrong line. Don't take my word for it. See if it confuses you, when it happens to you! Another trick is to keep unused lines of code after the end directive. The assembler ignores everything after the end directive. If you need them, you won't have to retype them.

Now we will try a short program written as absolute code. No linker was used as in relocatable code. Note the ORG directives have actual physical addresses associated with them. This is how the programmer tells the assembler where in program memory to put his code if not using linker script. If you want, you can write a simple routine in absolute code and build it in MPLAB using the Quick-build command (in the Project drop down menu). All you need are lines of instruction and an end directive. This is handy if you are trying out a few lines of code to see how it will assemble and run on the simulator. MPLAB uses the device selected in the Configure menu. Try it. Open up MPLAB and on File drop down click on New file. Click in the new window and Save As UPIC.asm or whatever name you like, the file extension must be .asm. Type the following lines of code in the open window:

```
LOC1  equ    020h     ; EQU assembler directives
LOC2  equ    021h     ; name two user RAM addresses
      clrw            ;clear W register
      movwf  LOC1     ;zero RAM location named LOC1
      movwf  LOC2
      incf   LOC1,f   ;watch register contents in simulator
      incf   LOC1,f   ;notice it increments
      incf   LOC2,w   ;but when we don't put it back
      incf   LOC2,w   ;in register it stays the same
      movf   LOC1,f   ;watch Z flag and see how a
      movf   LOC2,f   ;register contents affects it.
here  goto   here     end
```

Save File. Go to Project drop down and click on Quickbuild UPIC.asm. Go to the Debugger drop down and Mouse over Tool select and click on MPLAB SIM if it is not on the screen already. Single step through the program, watching the File registers selected from the View drop down along with the W register and Status flags on the status bar at bottom of the main window.

It's not so important to know how the microcontroller handles an instruction as it is to know what it does to the data and where it ends up. You are also becoming familiar with the programming suite and its features as you build a solid understanding of the instructions. You may want to try out more instructions using MPLAB SIM and compare what you observe to the description in the Mid-Range Users Manual.

Perhaps you are eager to actually use some hardware, program a chip, and see it work. In order for us to witness a PIC doing something useful we need some contact with the world outside the PIC. Look at the block diagram in Figure 4; see how many I/O devices are in the device and how few pins are available for your use. The data sheet shows that the peripherals share pins. It is necessary to configure the peripherals you would like to use and disable the others through program instructions. I will use the 12F675 again as an example because it has a comparator, A/D converters, digital I/O and timers. See Figure 3. We will use the one input port on pin 2, GP5, and one output on pin 3, GP4. Note: The data sheet for the 12F675 shows that pin 4 is input only. When setting up peripherals pay close attention to the data sheet. By setting the RP0 bit in the Status register we can access Bank1 of the Special Function Register, and by clearing that same bit we can switch back to Bank 0. In Bank 0, GPIO is cleared. Writing 07h to CMCON turns off the comparator. Switching to Bank 1 gives us access to ANSEL, where we select digital I/O. Writing 28h or binary 00101000 to TRISIO selects GP5 as input along with MCLR. All other I/O pins are output. Clearing the RP0 bit in Status takes us back to Bank 0. The remainder of the program looks at a switch that controls the LED. Just think, at the very low cost of small microcontrollers, one could be used in a flashlight! If your imagination is leading you to add features then you are going to enjoy microcontrollers. Also a program like this can be useful to test hardware on a new circuit. This program will run on a 12F675. Connect the anode of an LED to pin 3, the cathode goes to ground through a 150 Ω resistor. Connect a switch between ground and pin 2.

Now you have gone through the mechanics of writing and assembling a microcontroller program. Let's step back into the PIC again and look at Addressing Modes. There are two associated with program memory. Those are Immediate and Relative. Immediate addressing gets its data, along with the program instruction, right from program memory. Relative addressing is used by the skip instructions and computed addresses loaded into the program counter by the program via the PCL and PCLATH registers. A Relative mode may call the immediate mode as in a lookup table in program memory. An example would be an LED array pattern arranged in a predetermined order in program memory. Then using a computed value the converted value could be looked up.

Two modes associated with File Register memory are Direct and Indirect. Direct specifies the RAM address in the instruction. Indirect addressing uses an offset to point to a RAM data location using the FSR file select register. It could be used to pass variable data, say the result of a frequency measurement to a liquid crystal display (LCD) using several user RAM locations. After you have used the different modes it will easier to see how each one is useful for your purpose.

There is much more to be learned about programming than can fit into a magazine article, but now you should be able look at some program code and decipher its contents. Take advantage of the programs that are archived on the ARRL Web site for projects featured in *QST* and *QEX* articles. Try borrowing code and adapting it for your needs.

The American QRP Club has lessons available for down load on their Web site as the PIC ELMER 160 course by John McDonough, WB8RCR.[6] It is written by a ham, for hams, using many ideas a ham can put to practical use while learning.

Refer to the sidebar QSK Interface for Ten-TEC OMNI-VI+ and Heath SB-220 Amplifier to see how I used a PIC microcontroller to solve a timing problem.

Notes

[1]Phil Salas, AD5X, "External Full Break-In Transmit-Receive Switch for HF Amplifiers," *QST*, Feb 2008, pp 76 – 79. Also see Phil's Web page, **www.ad5x.com/**.

[2]Richard L. Measures AG6K has a lot of good information about converting amplifiers to QSK operation on his Web site at **www.somis.org/**.

[3]PICPROTO8 prototyping boards are available from MicroEngineering Labs, Inc, Box 60039, Colorado Springs, CO 80960; **www.melabs.com**.

[4]The program files associated with this article are available for download from the ARRL *QEX* Web site. Go to **www.arrl.org/qexfiles** and look for the file **3x09_Monroe.zip**.

[5]Microchip Technology Inc, 2355 West Chandler Blvd, Chandler, AZ 85224-6199; PICmicro and MPLAB IDE; **www.picmicro.com**.

[6]John McDonough, WB8RCR, American QRP Club PIC Elmer 160 Course Lessons, **www.amqrp.org/elmer160/lessons/index.html**.

An ARRL Life Member, Roger Monroe, K7NTW, learned about Amateur Radio while in grade school, when a teacher had the class build crystal radios. He was hooked! He received his Novice license in 1960, and lost the "N" the following year. Roger earned an AA degree in electronics technology, and served 4 years in the US Navy. He has also held the Amateur call signs TF2WLR and GM5DVL. He enjoyed 36 years of continuous employment servicing consumer electronics. He is now retired. Roger is married, and has two children who are still in school. His father-in-law, an electrical engineer, was influential in helping him reach for a more refined understanding of radio. Roger also enjoys reading about early electrical pioneers. It is still a thrill when something he built works!

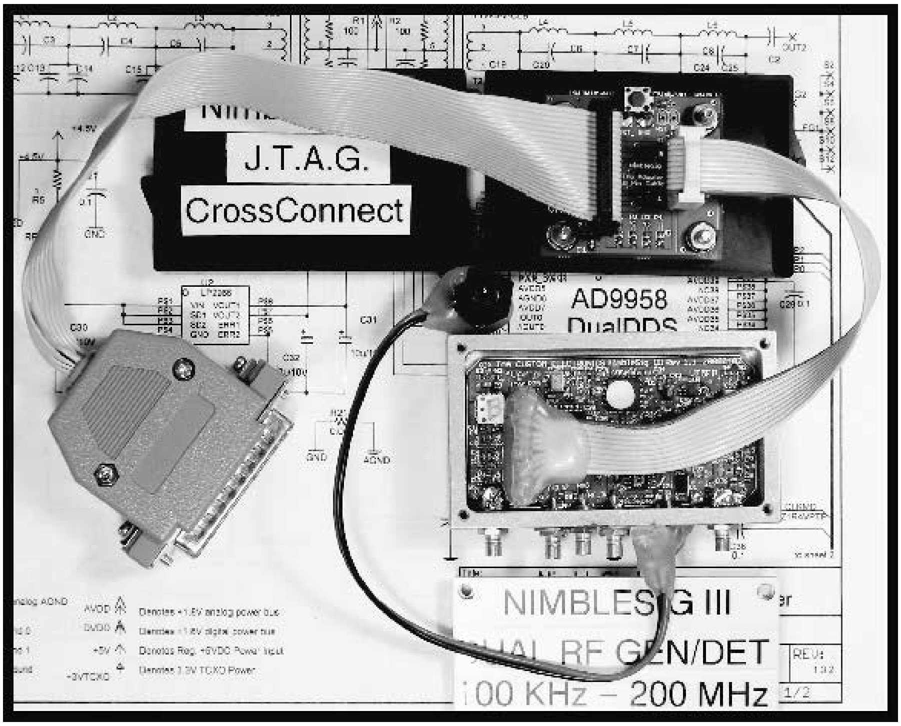
Thomas M. Alldread, VA7TA

7056 Railway Ave, Courtenay, BC V9J 1N4, CANADA; va7ta@telus.net

NimbleSig III—Part 2

Build this dual output DDS RF generator and low-level RF power meter.

Part 1 of this article described the hardware design and construction details for NimbleSig III (NS3). NS3 is a frequency agile, direct digital synthesizer (DDS) based RF signal source with independent dual outputs that have a frequency range of 100 kHz to 200 MHz. Included within the module is a wide dynamic range RF level meter. In Part 2 of this article, I will describe the software design, computer interfacing, MPU programming and initial testing procedure aspects of this project. Part 3 will describe the NimbleSig III command set and calibration procedures in detail.



The NimbleSig III connection daisy chain shows the parallel port JTAG interface connected to the 20 pin side of the cross-connection board and the NimbleSig III 10 pin JTAG programming cable connected to the 10 pin side.

Firmware

The majority of the firmware is written in "*C*" using Rowley Associates' *CrossWorks for ARM* professional Integrated Development Environment (IDE), for which there is a relatively low cost license available for the hobbyist. The vast majority of the code is written for the GCC compiler, which is used by the *CrossWorks* IDE. The only assembler code used is for the very short (just a dozen instructions) and quick, high modulation rate, fast interrupt service routine (FIQ ISR). The program code, which uses some *CrossWorks* IDE proprietary routines, is available for download in hex format, which is suitable for directly programming the NXP LPC2138 microcontroller using one of the methods described in this article.[1] At the time this article is being written the size of the NS3 source code is roughly 5000 lines, and the compiled program code size is about 100 KByte, which leaves about 80% of the program memory free for future enhancements. The RAM usage is just under 50% and only 15% of the external EEPROM is currently used.

Firmware Flow Chart

Figure 1 is a simplified flow chart that illustrates the firmware organization. Upon power up, the LPC2138 MPU internal boot loader does some housekeeping chores and continues on to execute the startup code. In general, the startup code configures the MPU to run the NimbleSig program. Among other things, the execution of the startup code configures the interrupt table vectors, sets up the 50 MHz system clock PLL to run from the 25 MHz reference clock, sets the peripheral clock to the maximum 50 MHz rate and provides pointers to exception handlers.

Once the MPU is configured, control is passed to the main NimbleSig program. The main initialization block does the following:

- Configures the general purpose input/output pin directions to match the NS3 hardware design.
- Sets up the UART for 115,200 Baud, 8 bit, no parity bit, one stop bit (8N1) operation with FIFO double buffers and TX/RX interrupts enabled.
- Sets the ADC input select to the ADC2 pin, which is connected to the RF level detector output and places the ADC into continuous operation at a rate of about seventeen thousand 10 bit conversions per second.
- Configures the I2C peripheral into master mode and enables it for 400 kbit/s EEPROM communications.
- Assigns the modulation timer1 interrupt to the Vector Interrupt Controller (VIC) vectored IRQ priority 2 slot. Note that the highest priority 0 slot is reserved for the fast interrupt (FIQ) mode that is used for the higher modulation rates. The VIC priority slot 1 was left unassigned and reserved for possible future use.

[1]Notes appear on page 24.

• Assigns the timing tick timer0 to the VIC priority 3 slot. This timer uses three of the four available timer0 match registers to generate 1 ms, 10 ms and 100 ms timing ticks, which are used for timing reference where needed throughout the program.

• The host communications UART0 interrupt is assigned to the vectored IRQ priority 4 time slot.

• Upon completing the VIC assignments, the interrupts are enabled.

• Timer0 is configured to generate the timing ticks mentioned above.

• Timer1 is configured to generate the start up default 400 Hz modulation.

• Configures the SSP to Bus Master, SPI mode, 25 Mbit/s DDS serial communications.

• Acquires the DDS reference frequency from the EEPROM and stores it into RAM. If the EEPROM is blank 500,000,000.0 Hz is assigned by default.

• Acquires RF level detector and frequency response calibration values from EEPROM and stores them in RAM.

• Acquires RF output level calibration values from EEPROM for both generators A and B, which are also stored in RAM for fast access.

• Loads the initial startup values into the DDS registers.

• Presets Generator A to 40 MHz and Generator B to 50 MHz as startup frequency defaults.

• Provides confirmation of the successful completion of initialization by printing the sign-on message shown in Figure 2. Figure 3 is a screen shot of the DDS register dump, showing the default initialization values.

Once the initialization is complete the program enters the multiple decision, repetitive forever loop. It will remain within this loop, scanning for the need to complete tasks unless stopped by one of the interrupt assignments.

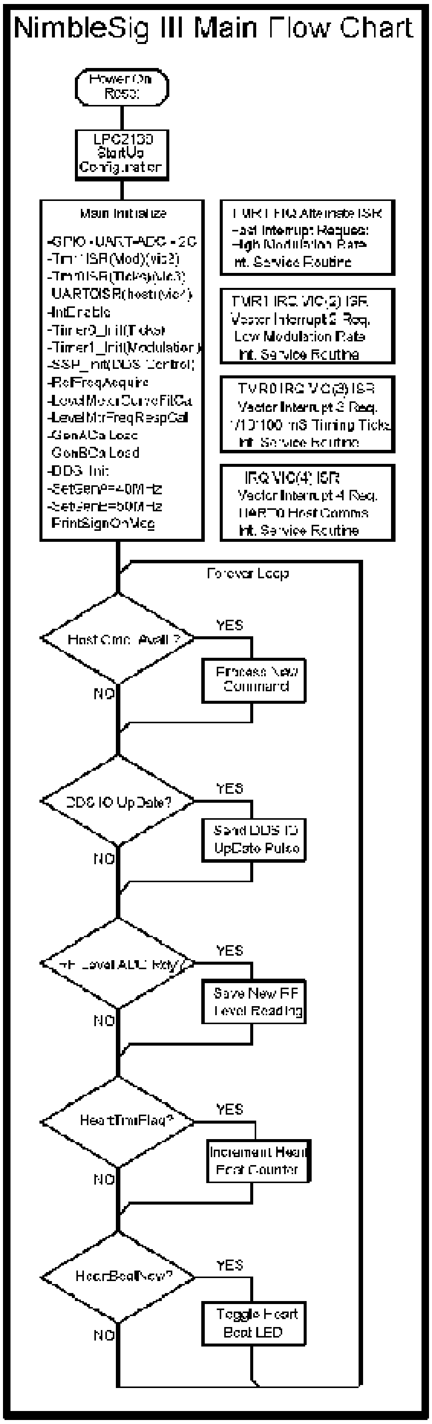


Figure 1 — This Flow Chart illustrates the main program control steps for the NimbleSig III operation.

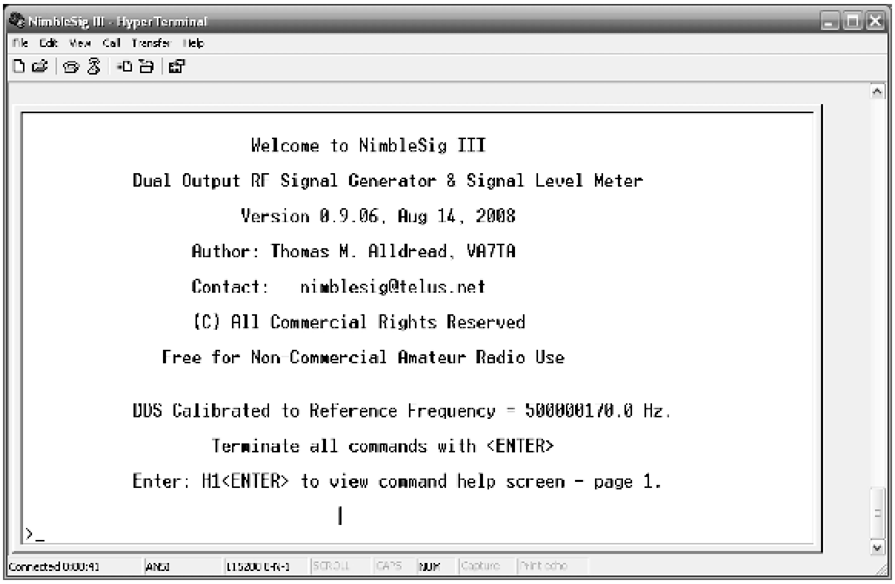


Figure 2 — The NimbleSig III Sign On Screen indicates initialization has been completed.

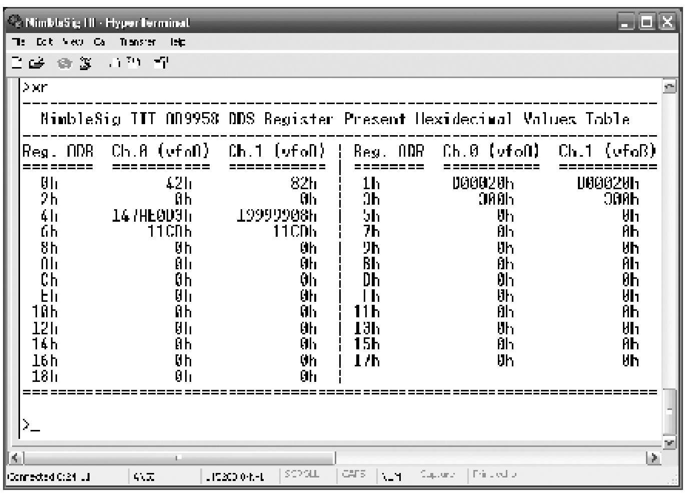


Figure 3 — This DDS Register Dump shows the default initialization values.

Upon entering the forever loop, NS3 first checks for the availability of a new host command. In the event of a new command, the program flow will branch out of the loop, decode and implement the command and then return to the forever loop.

The next decision test detects for the completion of a fresh SSP command data transmission to the DDS. When an SSP data transmission to the DDS has been completed an I/O update pulse must follow to activate the new data within the DDS engine.[2] If a fresh SSP command completion is detected and a corresponding I/O update pulse has not already been sent by a subroutine then a new I/O update pulse is sent. This I/O update function is needed here because the program flow often returns to the forever loop while the independently functional SSP peripheral is still in the process of sending the most recent command data to the DDS.

Next, a check is made to see if there is new ADC data available. As mentioned above, the ADC that measures the RF level detector output runs continuously. If it is found that there is a new ADC value available, the data is stored within a variable. This ensures a very recent RF level reading is always available.

The final two decisions within the forever loop maintain the flashing of the heartbeat LED. Thus, when flashing at the normal 72 pulses per minute rate, the LED indicates the program is idling along within the forever loop. Some of the calibration commands that need to wait for user input will interrupt the forever loop, but during normal operation the heartbeat LED should be always flashing.

The four blocks to the right of the main initialization block in the flow chart represent the very short interrupt service routines (ISRs). These ISRs remain idle until called upon by a corresponding interrupt request. When an interrupt occurs, the forever loop program flow is stopped for a short time while the MPU branches out to execute the instructions in the appropriate ISR.

Personal Computer Serial Interface

A personal computer (PC) will be needed to program, test and calibrate the NimbleSig III module. The PC may be connected to the module 3.3 V UART interface via either a legacy 9 pin serial port or a USB port. The legacy serial (COM) port option requires an RS232C to 3.3V data level conversion adapter. The USB port alternative requires a USB protocol to 3.3V UART serial data adapter. At the time of this writing both adapter types could be purchased on eBay already assembled on tiny PC boards at a price similar to the procurement cost of the parts in single unit quantities.[3]

In the past I have used the legacy 9 pin serial port with my favorite RS232C adapter design variation, which derives its power from the serial port.[4] More recently, however, I have been using the more modern alternative USB-RS 232 port adapter, which is based on the preassembled adapter board mentioned above. It uses a popular IC manufactured by Future Technologies Devices International Ltd (FTDI). I first soldered the FTDI board via the provided header connectors onto a 0.1 inch perforation grid prototype board, which I then mounted onto the base of a somewhat over-sized (but convenient and free) wall-wart shell. As shown in Figure 4, the pigtail-style TX/RX data coaxial cables are soldered to the perf board and are anchored to the shell base with hot glue.

This versatile USB adapter board uses the FT232RL chip, which offers a means for setting the logic voltage level for the UART data interface.[5] As the board provides for various powering and logic level options, a couple of soldered jumpers are needed to configure the board for this application. The desired logic level is set by placing +3.3 V on the VCC-IO pin, which is conveniently extended on this particular circuit board to a solder pad labeled "VIO." The FT232RL has an internal LDO regulator that provides a regulated 3.3 V output, which I used (in accordance with the data sheet recommendation for 3.3 V logic levels) to feed the VCC-IO pin. I discovered the 3.3 V LDO output connected to a chip capacitor near the USB connector. It was easy to solder a jumper made from no. 30 AWG wire wrap wire from that point over to the VIO pad. To select the PC USB port 5 V supply as the power source for the adapter

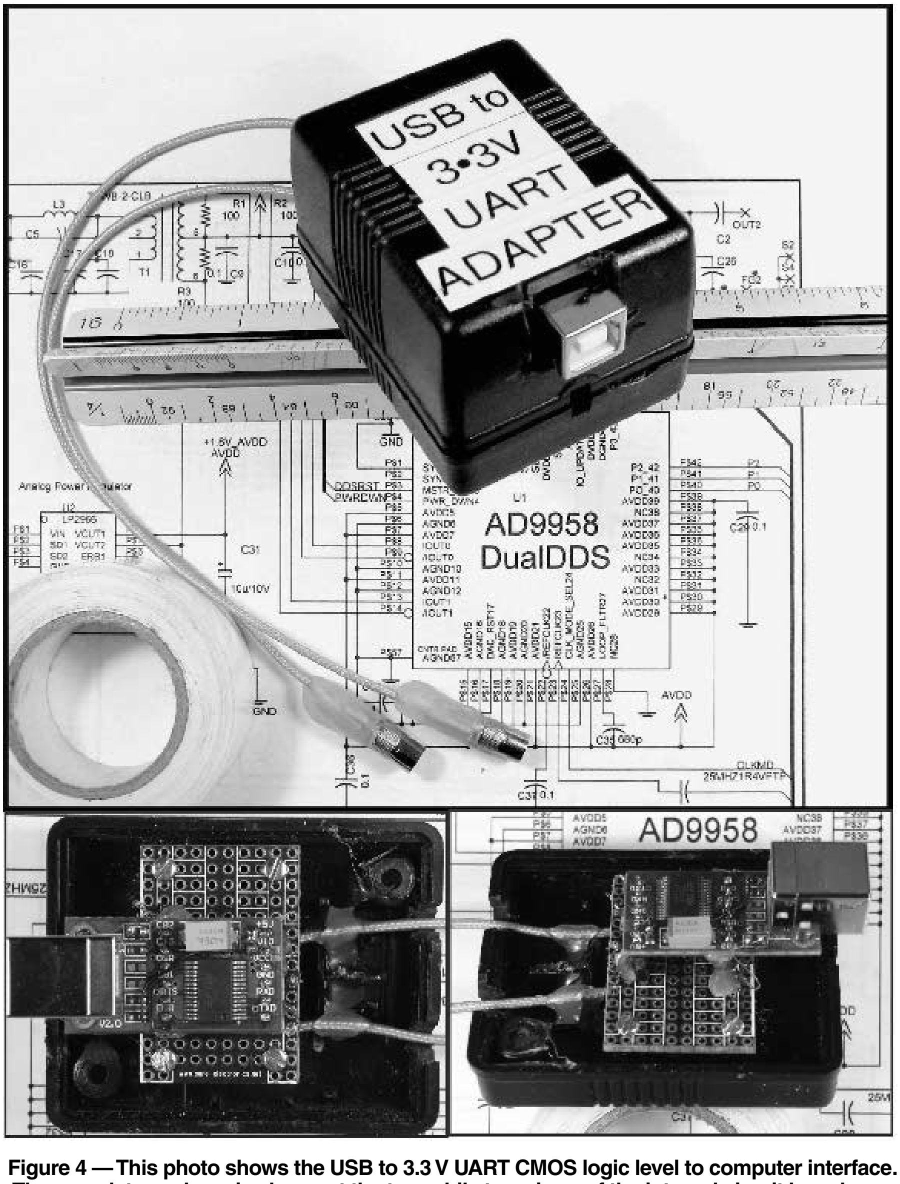


Figure 4 — This photo shows the USB to 3.3 V UART CMOS logic level to computer interface. The complete package is shown at the top, while two views of the internal circuit boards are shown at the bottom.

I also needed to solder two circuit board pads together. These were side by side and nicely labeled, so this connection was also easy. To use this adapter, a *Windows* driver is available for download from the Internet.[6] This driver is needed to detect the USB FTDI device and to make it appear to software applications as a new *Windows* virtual COM port in Device Manager.

MPU Programming Procedures

Prior to attempting to program the MPU the voltages on the power buses should be confirmed as normal and the 25 MHz clock input to the MPU should be checked. Please note that the MPU cannot be programmed unless it is driven by a proper (in this case 3.3 $V_{P\text{-}P}$, 25 MHz square wave) clock signal. Note that I have encountered difficulty soldering the lead-less TCXO chip. If the clock signal is missing try pressing down on the TCXO chip. If the 25 MHz pulse stream suddenly appears, the TCXO needs to be resoldered to obtain good solder flow to the hidden connections under the package.

There are two methods that I will describe for loading the NimbleSig III hex format code into the FLASH program memory of the LPC2138 MPU. The first method uses one of the UART serial ports while the second uses the JTAG debugger interface. Both methods require the running of special, but fortunately free, programming software that can be downloaded from the Internet.

Serial Port "In-System Programming" (ISP) Method

The ISP method which takes about 2 minutes to program the MPU is about ten or more times slower than programming via the JTAG port. Unless you plan to reprogram the MPU repetitively (as is necessary during software development) this tardiness is probably more than offset by the fact that you do not need any special programming hardware to use the ISP method. I think this ISP alternative will be the most appealing to those that do not already have or want to acquire the JTAG interface that is needed for writing and debugging embedded programs.

The ISP method involves uploading the hexadecimal formatted program code file through the UART0 serial port interface at a rate of 38 KBaud (the maximum recommended rate for this application). The internal ISP loader firmware is permanently factory installed. This firmware receives and converts the hexadecimal formatted program code into binary machine code prior to writing it into the FLASH program memory. Although the ISP method requires either the USB or the RS232 serial port interface described above, that interface is also required later for the calibration and routine PC keyboard control of NS3. Thus, there is no extra hardware needed for the ISP programming method except for the improvisation of a simple grounding probe made from a needle, as described below.

Since I am all set up for the much faster JTAG programming method, unfortunately I didn't have this relatively slow method of programming in mind when I designed the circuit board artwork. This resulted in an ISP mode start up procedure that is not as convenient as it could be. I intend to provide a more convenient means for entering ISP mode if I find the need to order another batch of circuit boards. In order to get the MPU to start up in the ISP mode with the current circuit board artwork design, you must carefully place a logic low on the MPU I/O port 0, bit 14 pin (P0.14) during reset. The P0.14 I/O bit connection is assigned to the LPC2138 MPU package pin number 41.

In preparation for ISP programming, first install the free Internet download "FlashMagic" *Windows* software package onto the host computer.[7] FlashMagic, which is designed to interface a *Windows* PC platform to the LPC2000 ISP protocol, will be used to program the NS3 module. FlashMagic also provides tools for viewing and erasing the MPU memory in addition to programming. A screen capture of the FlashMagic main window is shown in Figure 5.

Download the hex program code from the Web site listed in Note 1 or from the ARRL *QEX* files download Web site.[8] Save the latest version of the "NimbleSig3_1r3r*.hex" program file to a convenient location (such as the root directory of a USB removable drive or to a dedicated hard drive directory created for this project) within the host PC folder organization structure.

Prepare a ballast probe made from a needle, flexible lead, current limiting resistor and a small alligator clip for placing a logic low on pin 41. Solder a 6 to 8 inch flexible lead to the needle head. Connect the opposite end of the lead to a 100 to 1000 Ω, current limiting, pull-down resistor, which in turn needs to be connected to ground. It is important to current limit this probe with the resistor to prevent possible damage to the NS3 circuitry should a wrong pin be touched accidentally. As an added precaution the needle body and the resistor should be insulated with fine gage heat shrink tubing, leaving only the tip of the needle exposed. I used a sewing machine needle with a square shank connected via a flexible, stranded, light gauge wire to a 330 Ω, ¼ W resistor mounted on the stub of a small alligator clip. My probe, which is insulated and supported at both ends with multiple layers of shrink tubing can be seen on my Web site.[9] There is also an alternative reset method described on my Web site.[10]

Using the pin diagram from the LPC2138 data sheet "Pin Configuration" chapter, identify the location of pin 41. With the power

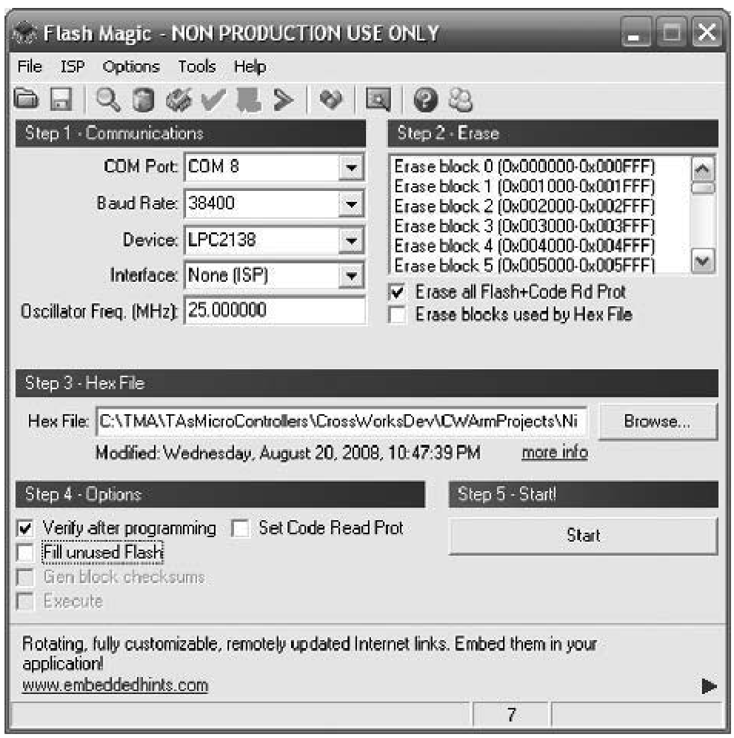


Figure 5 — This FlashMagic screen shot shows the settings for programming NimbleSig III.

off and the use of magnification as necessary, establish confidence by practicing the connection of the current limited grounding probe tip to pin 41. Note that the 10 kΩ resistor R24 is the pull up resistor for pin 41, so the probe may alternatively be connected to the side of R24 that connects to pin 41. It will be necessary to apply power while pin 41 is held low, so prepare a convenient method for applying power to the module. During power up initialization, the LPC2138 MPU always senses the logic level on pin 41 and will enter ISP mode if it senses a logic low (<0.8 V). Once powered up, the pull-down needle probe may be removed from pin 41. Do not expect the heartbeat LED to flash while in ISP mode.

The ISP programming procedure is as follows:

- Execute the FlashMagic program. When it boots you should see a window similar to Figure 5.
- Select the COM port that is used for the NimbleSig interface.
- Set the Baud Rate to 38,400, Device LPC2138, Interface ISP, Oscillator 25 MHz as shown in Figure 5.
- Use the "Browse" button and point to the NS3 hex file name.
- Select the "Erase all Flash+Code Rd Prot" and "Verify after programming" options.
- Now boot NS3 into ISP mode by applying power to NS3 while pulling down the LPC2138 package pin 41 with the special probe assembly as described above.
- You may view the program memory contents with the ISP menu "Display Memory" Option. A blank memory is full of hexadecimal "FF" characters, which is what you should see if the MPU has never been programmed before.
- Click the "Start" button and you should see the options area become disabled (grayed out) and the programming and verify status reported in the status bar at the bottom of the window. After a couple of minutes you should get a "Finished" message in the status bar.
- You may wish to again view the program memory contents with the ISP menu "Display Memory" Option. Now the memory contents should show random hexadecimal characters corresponding to the NS3 program code as shown in Figure 6.
- Finally, power cycle NS3 to reset the MPU (without application of the P0.14 pull down) and the heartbeat LED should start flashing indicating the MPU is executing the NS3 program code.

If FlashMagic fails to communicate with the ISP the first time, try the startup procedure again. If you still encounter difficulty establishing connection with the LPC2138 some tips are given below in the "Initial Testing" section for checking the functionality of the interface.

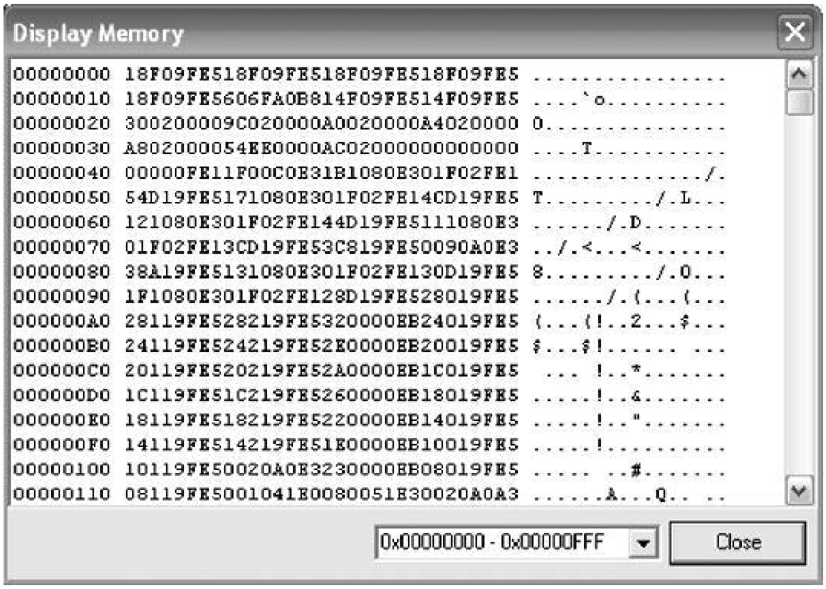


Figure 6 — A FlashMagic screen shot of the LPC2138 Memory Dump Display, after programming.

JTAG Programming Method

The JTAG method programs the MPU significantly faster than the ISP method. The downside of the JTAG method is that it requires the use of a computer set up with an ARM JTAG interface and the construction of an adapter cable assembly to connect to the NS3 JTAG connector. I have designed a small, passive PC board for interfacing the 20 pin ARM JTAG standard to the more space efficient 10 pin connector used here. I think this programming method will appeal to those who already have an ARM JTAG interface or those interested in getting set up for developing their own embedded applications for ARM microcontrollers.

There are numerous types of ARM JTAG programming/debugging interfaces available. Some are proprietary and only work with certain development software packages. If you already happen to have a JTAG based ARM development system that will program LPC2000 series MPUs from hexadecimal format files, then you may already have what you need to program this LPC2138 using the NS3 cross-connection board and cable described here. Figure 7 shows the cross-connection board schematic and Figure 8 is a view of the board and associated ribbon cable.

The board simply cross-connects the necessary JTAG signals from the 20 pin JTAG header to the appropriate pins on the NS3 JTAG 10 pin SIP connector cable. In consideration of the longer JTAG ribbon cable length, I provided pull-up, pull-down resistors and a power rail bypass capacitor. I decided to also provide mounting pads for a reset switch, which is a nice-to-have convenience during program development. The adapter cable, which connects between the PC board and the NS3 module, simply consists of a 10 conductor ribbon cable that has a standard 10 pin (2 rows by 5pins, 100 mil spacing) female-header press-fit connector on one end and a 10 pin SIP strip plug on the other end. The cross-connections done on the PC board match up the order of the 10 ribbon cable wires directly to the order of the SIP connector pins, which makes the ribbon cable wires relatively easy to attach and solder to the SIP pins. I used hot glue to form a cap for the SIP strip. The lead photo shows the overall JTAG connection daisy chain.

There are a variety of JTAG interfaces available that will adapt either a PC USB port or a legacy PC parallel port for programming/debugging an ARM MPU. Some will only work with specific IDE development software packages. Usually a given IDE software package will perform the best with the custom designed interface that is designed specifically for it. Interfaces can cost as much as several hundred dollars. I find the relatively low cost "CrossConnect Lite" interface from Rowley works very well for me with their *CrossWorks for ARM* IDE package. They also offer a more expensive, higher speed interface for commercial enterprises.

Probably the most universal and certainly the least expensive JTAG ARM interface adapters are clones of the parallel port ARM

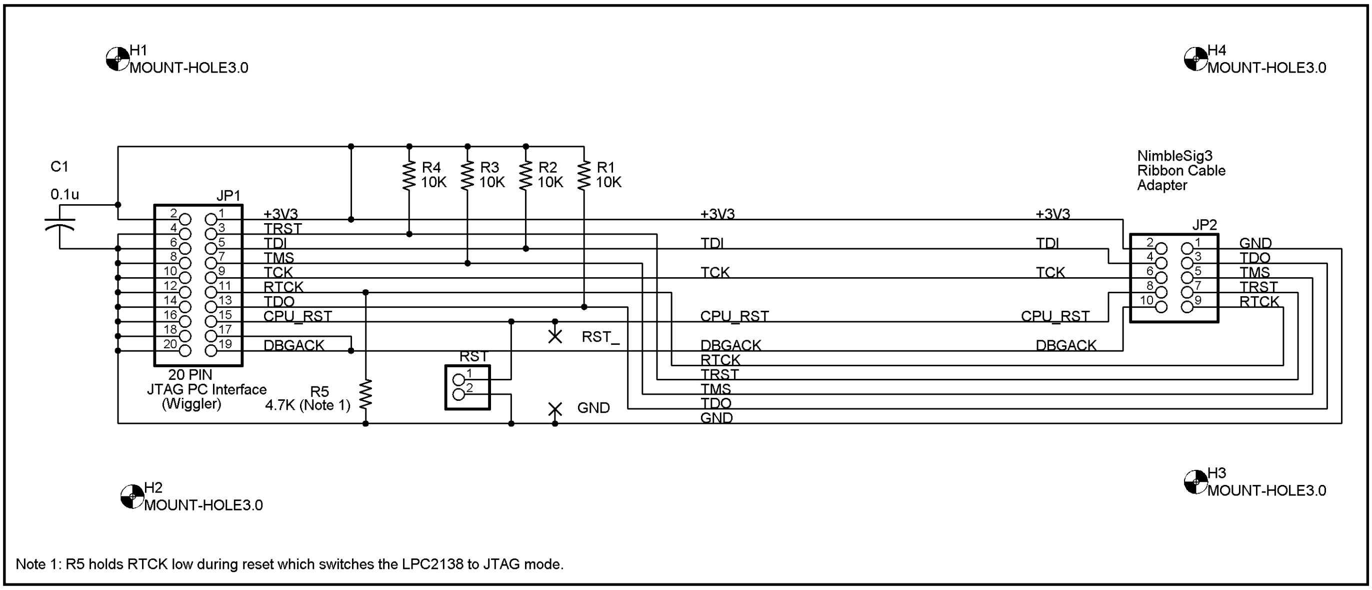


Figure 7 — This schematic diagram shows the NimbleSig III JTAG Adapter Board.

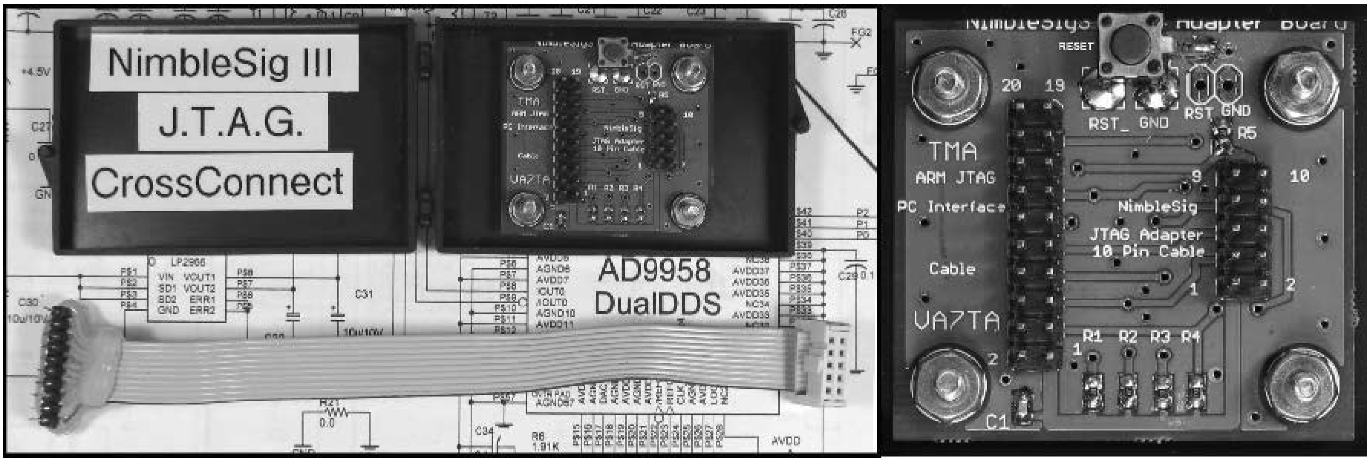


Figure 8 — On the left is the cross-connection circuit board with the NS3 JTAG cable, A close-up view of the circuit board is shown on the right.

JTAG interface named "Wiggler." My understanding is that Macraigor Systems originally designed and at the time of this writing still manufacture this product.[11] The Wiggler interface design will work with several development software packages. It is now manufactured by many after market companies. I have noticed these interfaces sold on eBay for less than $10. The particular unit I have has been in production for some time and is widely used. It is manufactured by Olimex Ltd. and is available from SparkFun Electronics in the USA for roughly $21.[12] The Wiggler interface does, however, require a computer with a legacy parallel port. I installed an inexpensive ($10), PCI slot, 2 serial / 1 parallel multi-port communications card into my modern day desktop computer. This card worked out well for my Olimex interface, and as a bonus, I find the legacy ports are useful to have available in my ham shack for other applications.

As shown in the lead photo, the Wiggler JTAG adapter is packaged into a 25 pin D connector shell. The 20 conductor ribbon from the interface connects to the 20 pin header on the NS3 JTAG cross-connection board. The 10 conductor JTAG SIP connector cable described above completes the connection to the NimbleSig III board JTAG connector. Please note that because I failed to leave sufficient room in the original cross-connect board artwork designs for alignment pins or shrouds, care is needed to ensure that both of the ribbon-cable connectors are plugged onto the circuit board headers with proper orientation. The usually tracer-marked conductor number ones of the ribbon cables must connect to the corresponding pin number ones of the header connectors.

I have also noticed an error on the silk-screen pin numbering for the 10 pin (2 × 5) header—the odd and even rows are reversed. Pin 1 is actually pin 2 and vise versa. I suggest the use of an ohmmeter with reference to the schematic diagrams and connector pin 1 references, to first ensure proper orientation. Then mark the connectors and boards (a dab of nail polish works well) for future reference. After connecting the JTAG interface to your PC parallel port (as shown in the lead photo) the next step is to get ready to run the JTAG software for uploading the NS3 program into the microcontroller program memory. A free *Windows* program called "H-JTAG" that could be downloaded from the Internet performed nicely for me.[13] The H-JTAG/Wiggler interface combination is apparently capable of programming a wide range of JTAG featured semiconductor devices.I found that once I finally got the options set correctly that H-JTAG running on my *Windows XP* desktop computer has proven to work very reliably for this application. I did, however, have problems getting it to work initially because I didn't have any definitive reference for setting up the options for my Wiggler compatible interface. Hopefully the instructions that follow will make it a much simpler task for others.

The first step is to download and install H-JTAG (I downloaded version V0.7.0) on to your *Windows* PC platform.

When H-JTAG is started, the "H-JTAG Server" window shown in Figure 9 should pop up. If the H-JTAG options are not set properly for your interface and/or parallel port it may not successfully detect the LPC2138, and may consequently show the device ID as "UNKNOWN" in red text.

To get H-JTAG to work with my Olimex parallel port JTAG interface I needed to select the "LPT JTAG Setting" option under the "Settings" menu item, which popped up the window shown in Figure 10. The only setting here that I needed to change from the provided defaults was nSRST, which needed to be changed from "NO SYS RST" to "Pin6 D4." Then, since my parallel port is assigned to LPT3 address 0xA400, I needed to open the "LPT Port Setting" option that is also found under the "Settings" menu item. There I selected Lpx and inserted the address of my parallel port. The parallel port address setting in H-JTAG must match that assigned to your parallel port by *Windows*. You can find the address in Control Panel>System>Hardware>Device Manager>Ports. Then right click parallel port, select properties and then the Resources tab, where the parallel port address is shown.

Once all the options are set correctly, H-JTAG should detect the LPC2138 and display the "ARM7TDMI 0x4F1F0F0F" chip ID in blue text as shown in the center of Figure 9. The chip detection may be re-tested by clicking on the "Detect Target" option under the "Operations" menu item.

Once H-JTAG detects the MPU correctly, as shown in Figure 9, you are ready to take the programming plunge. Click the "Flasher" menu item and select the "Start H-Flasher" drop down option. You should get a new Window titled "H-Flasher" similar to that shown in Figure 11.

The procedure is as follows:

• Select no. 1 "Flash Selection" menu, then select NXP in the "Flash Selection" list and finally select LPC2138 from the drop down list.

• Select no. 2 "Configuration" and enter the clock frequency of 25 into the "XTAL(MHz):" window.

• Select no. 4 "Pgm Options" and tick "VERIFICATION" and "NXP LPC2000" options.

• Select no. 5 "Programming" option and you should get a new internal window titled ">> Programming – LPC2138" as shown in Figure 11. Set the file type to "Intel Hex Format" and point (via the "..." button) to the "Src File:" by browsing through your file structure to the location of the NimbleSig III program code hex file. To make this easier for the next time, you can save all these settings if you wish. I chose to save them in a file called ns3 and H-Flasher added the "hfc" extension to making the full file name ns3.hfc.

• Finally, to initiate the programming operation click the "Program" button and the NS3 program should upload into the LPC2138 flash program memory in about 10 seconds. When it completes, it should indicate the upload verified successfully.

• Reset the MPU with the reset switch on the NS3 JTAG cross-connection circuit board and the heartbeat LED should start to flash, indicating the NS3 firmware is running.

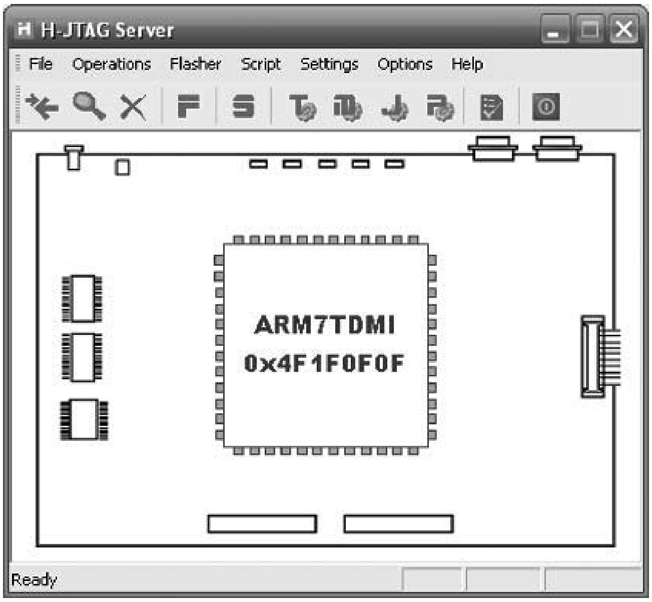


Figure 9 — Here is a screen shot of the H-JTAG Initial Screen.

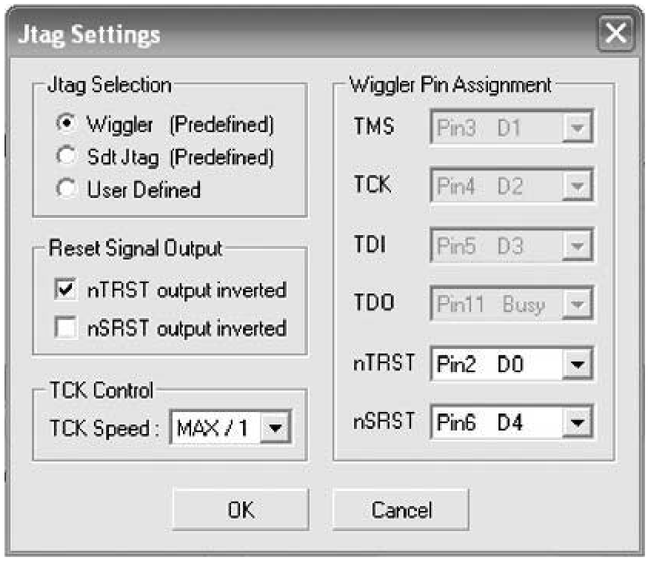


Figure 10 — The H-JTAG NimbleSig III programming JTAG settings are shown in this screen shot.

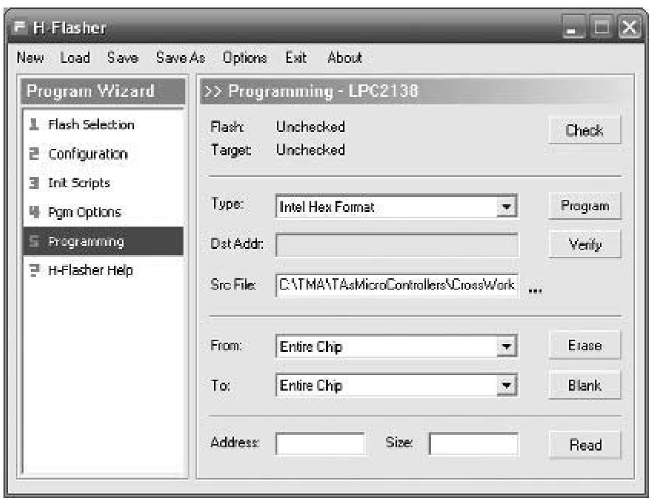


Figure 11 — This screen shows the H-Flasher NimbleSig III programming settings.

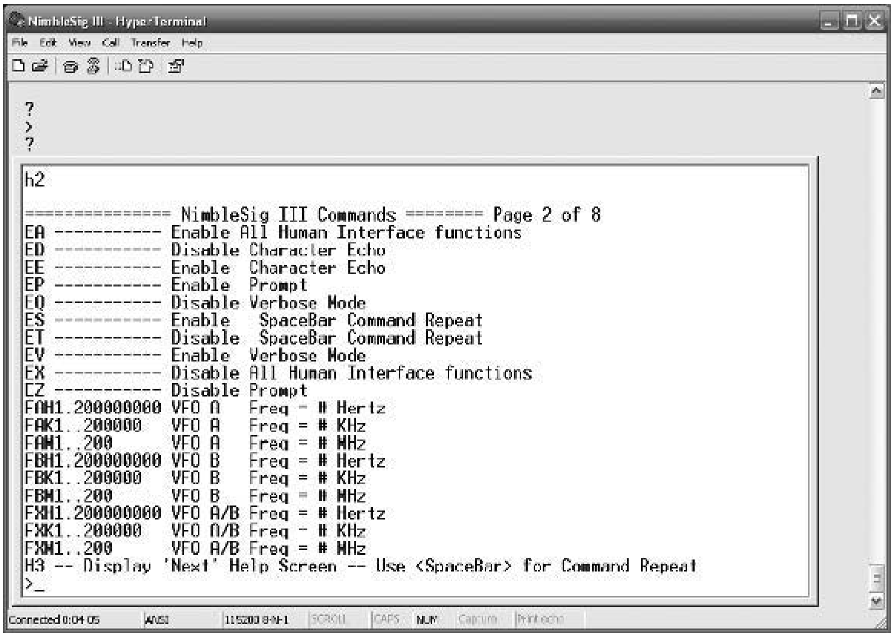


Figure 12 — An example of the NimbleSig III Command List Help Screen — page 2 of 8.

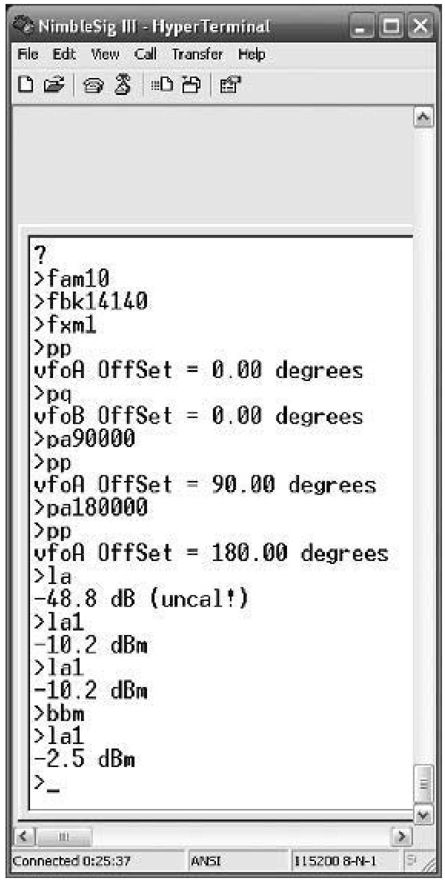


Figure 13 — This screen shot shows the NimbleSig III preliminary testing sequence of commands.

Initial Testing

Once the heartbeat LED is flashing, the next step is to establish computer communications with the NS3 module. Connect the computer-to-NS3 serial UART data COM interface of your choice (see above) to the NS3 data TX/RX connectors (if you programmed the chip with the ISP method this connection is probably already done). Using your favorite terminal program set the serial data options to a speed of 115,200 bits per second (Baud), 8 data bits, no parity and 1 stop bit (8N1) and set the flow control option to "None." When you first power up the NS3 module you should get the welcome screen shown in Figure 2. If you press "Enter" repeatedly, NS3 should echo a command error "?" followed by the ">" ready prompt on the next line.

If you have trouble establishing communications and need to troubleshoot, start by looping the interface back on itself. You should be able to type text to yourself without errors. If not, you will need to troubleshoot your terminal software setup and/or NS3 data interface until the loop back works. If the loop back works but you are still unable to communicate with the module check the serial data options mentioned above and confirm that the TX/RX data leads to the NS3 module are not reversed. As shown in the schematic (Figure 4 in Part 1) the data path hardware within the NS3 module is minimal. There is nothing except a pair of current limiting resistors in series with the TX/RX data paths, so there is not much to check other than the connections. NS3 software execution can be confirmed by observing the flashing heartbeat LED.

Once communications is established, enter a few commands to confirm normal operation. Note that all NS3 commands must be terminated with the "Enter" character and that either upper or lower case text is okay. First try the "xr" command. You should get a register dump similar to that shown in Figure 3. The values in the registers are the initialization values. The values at address 4h are the frequency tuning words (FTW) for the DDS that result in output "A" operating at 40 MHz and output "B" at 50 MHz. There will be a slight difference in the actual FTW values due to frequency calibration differences from unit-to-unit.

If the register dump looks reasonably normal the next step is to check for signal outputs. Using suitable test gear (scanner, frequency counter, oscilloscope or best of all a spectrum analyzer) check for a 40 MHz signal on output "A" and a 50 MHz signal on output "B." The signal amplitudes should be roughly –5 dBm or 0.3 $V_{P\text{-}P}$. The most common problem I have encountered with the loss or intermittent failure of either output is with the hidden-from-view, lead-less solder connections of the broadband output transformers. I find these connections difficult to manually solder with confidence. I make a point of applying a little force to the transformers from various angles while monitoring the corresponding output to ensure that the signal remains steady.

Next, try paging through the eight command help pages by entering "h1" through "h8." You should get help page screens similar to Figure 12, which list the command syntax for the various commands.

To pre-calibrate the NS3 power meter with typical calibration data enter the "xe" command and type 'c' to continue when requested. To pre-calibrate the output response of both generators with typical values enter the "xk" command and press 'c' to continue when asked.

The following is a list of steps to follow to check out the basic functionality of the NS3 module and in the process gain some insight into the use of some of the commands. The Figure 13 screen capture illustrates this sequence of commands:

- Try changing the output A frequency to 10 MHz with the "fam10" command and confirm the frequency shift with your test gear.
- Similarly, try changing the output B frequency to 14.140 MHz by entering "fbk14140."
- Do a phase comparison of the two signals on a dual channel oscilloscope if you have one available. This is best done with equal length cables terminated with 50 Ω at the oscilloscope input connectors.
- Enter "fxm1" to set both A and B outputs to 1 MHz with a defined phase offset. Check that the locked phase A and B output

offsets are 0.00° by entering the "pp" and "pq" commands respectively. Observe the two in phase signals on the oscilloscope.

- Enter "pa90000" to offset the phase of generator A by 90,000 millidegrees. Then enter pp to get confirmation that VFO A is offset by 90.00°. Observe the 90° phase difference on the oscilloscope.
- Enter "pa180000" to offset the phase of VFO A by 180,000 millidegrees. Then enter pp to get confirmation that VFO A is offset by 180.00°. Observe the 180°, phase opposite signals on the oscilloscope.
- Enter "la" to check the quiescent level from the power level detector. It should be around –50 dBm, but it depends somewhat on the RF noise in the room (see below).
- Connect the generator A output to the RF detector input, then enter the "la1" command and NS3 should report a level of about –10 dBm.
- Connect generator B output to the RF detector input, then enter the "la1" command and NS3 should again report a level of about –10 dBm.
- Enter "bbm" to set output B to maximum level with no level correction. Enter the "la1" command and NS3 should report a level within the –2 to –4 dBm range. To do the same test at 10 MHz (not shown in Figure 13) enter "fbm10" and then enter "la10" to measure with the 10 MHz calibration factor. Note that this maximum output level will vary with frequency, but should remain higher than –11 dBm throughout the 100 kHz to 200 MHz range once the power detector has been properly calibrated. The 200 MHz low pass filters should start to roll off above 180 MHz. Note that there is a new design for a 230 MHz low pass filter described on my Web site, which I believe is an improvement, because it does not start to roll off until about 200 MHz.

Figure 13 is a screen shot of the above sequence of tests done with a fully calibrated module. Note that both the A and B output levels, which measured exactly the same, are very close to –10 dBm. This is because this unit had already been calibrated prior to this test sequence. It is unlikely that a fresh, uncalibrated module would perform to this level of accuracy with just typical calibration data — but it should be reasonably close.

The power detector noise floor can vary significantly depending on the RF environment. For example there was a three inch long, 93% shielded coaxial cable with SMB connectors on each end, connected to the detector input when I measured the –48.8 dBm noise floor shown in Figure 13. When I removed the cable, the noise floor dropped to –65 dBm. I suspect the short, open ended input cable was picking up 2.4 GHz RF from my WiFi network wireless router, which was only about 3 feet away.

If you get similar results with this series of initial tests, you are ready to move on to final calibration of the NS3 module. Final calibration will require access to a well calibrated RF signal generator and a receiver capable of receiving WWV. Part 3 of this series will describe the NS3 command set in detail and will provide the final calibration procedures.

NimbleSig III Corrections, Updates and Errata

The value of C51, the MPU clock coupling capacitor shown in the Figure 4 schematic of Part 1 of this article, is much too high and in accordance with Murphy's Law may, in some cases, prevent the start up of the LPC2138. The value of C51 shown as 0.1 µF in Part 1, Figure 4 should be changed to 100 pF. Revised schematics can be downloaded from my Web site.[14]

The LPC2138 MPU used here could possibly be more clearly identified as an LPC2138/01 which is the revised, post 2006 version. Although it is unlikely that an older version LPC2138 would be supplied now, it is possible that there may be some older devices still in distribution or obtainable by special order. It is important that the revised LPC2138/01 version be specified for NimbleSig III, because there were unresolved issues with the original device that could affect NS3 operation.

A new design for the low pass filters, which extends the corner frequency from the original 200 MHz design described in Part 1 to 230 MHz has been tested. This modification extends the calibration range of the generators to 200 MHz with room to spare. The frequency response simulations, schematic and parts information can be found on my Web site.[15]

During tests of the 230 MHz low pass filters, a 125 MHz spur (¼ the DDS clock frequency) was noticed on the "A" generator output at a relative amplitude of –50 dB or lower. This spur does not appear on the "B" output and may be related to the slight difference in the ground plane between sides A and B of the circuit board. I have no reason to believe that it is related to the 230 MHz LPF modification. I noticed in general the "B" output is the cleanest output of the two, so I recommend it be used for the more critical signal need in an application. This seems to be consistent with units equipped with either the 200 MHz or 230 MHz filter. I also noticed the 500 MHz clock spur is between 40 and 50 dB down. It may be possible to reduce this spur by trial and error tuning the 500 MHz traps in the filters. I have not tried to do this at the time of this writing. I will post any further information on this subject on my Web site.[16]

Notes

[1]See **www3.telus.net/ta/NimbleSig%20III/NS3%20HEX%20Code/**.
[2]ADI AD9958 data sheet URL: **www.analog.com/static/imported-files/data_sheets/AD9958.pdf**.
[3]The 3.3 V UART adapters are described on eBay as "USB to 1.8~5V TTL Serial UART (TTL) Adapter" seller id: yliu73.
[4]See **www3.telus.net/ta/NimbleSig%20III/RS232C_3.3VUART_DataConverter/RS232_UART_Sch.html**.
[5]See **www.ftdichip.com/Documents/DataSheets/DS_FT232R_v104.pdf**.
[6]See **www.ftdichip.com/Drivers/VCP.htm**.
[7]Flash Magic download: **www.flashmagictool.com/**.
[7]See **realterm.sourceforge.net/index.html#downloads_Download.**
[8]The hex program code and other files associated with NimbleSig III, current as of the publication date of this article, are available for download from the ARRL *QEX* Web site at **www.arrl.org/qexfiles**. Look for the file **3x09_Alldread.zip**.
[9]See **www3.telus.net/ta/NimbleSig%20III/ISP%20Mode%20StartUp%20Probe/ISP_Mode_StartUp_Probe.jpg**.
[10]See **www3.telus.net/ta/NimbleSig%20III/NS3_rev1r3_ISP_BootUpMod/NS3_ISP_Re-Start_Mod.html**.
[11]See **www.macraigor.com/wiggler.htm**.
[12]See **www.sparkfun.com/commerce/product_info.php?products_id=275**.
[13]See **www.hjtag.com/download.html**.
[14]See **www3.telus.net/ta/NimbleSig%20III/NS3%20Schematics/**.
[15]See **www3.telus.net/ta/NimbleSig%20III/NS3_230%20MHz%20LowPassFilter/**.
[16]See **www3.telus.net/ta/NimbleSig%20III/NS3%20Errata%20and%20Updates/NimbleSigIII_Errata.html**.

Tom Alldread, VA7TA, became interested in electronics when still in grade school, repaired radio and television sets in his teens, obtained his Amateur Radio ticket at the age of 19, and went on to obtain commercial radio operator certification a year later. He subsequently graduated from the Capitol Radio Engineering Institute Engineering Technology program. His career was in the telecommunications industry where he worked in microwave and VHF radio equipment maintenance, training, engineering standards and design, and long distance network operations management.

Tom is now retired and lives on Vancouver Island with his wife Sylvia, VA7SA. Tom has a history in repeater building and emergency communications, and he enjoys operating mobile CW, building equipment and is currently the net manager for the 20 meter Trans Canada Net. His other interests include microcontroller programming, circuit design, computing, hobby farming, digital photography, bicycling and sailing.

QEX

Ken Grant, VE3FIT

5 Windrush Trail, West Hill, Ontario, Canada M1C 3Y5, ve3fit@arrl.net

A Versatile Two-Tone Audio Generator for SSB Testing

Here is a useful piece of lab gear that you can easily build yourself!

I've been working on restoring a 25-year-old SSB transceiver. As things tend to do, I soon found that this restoration, if it was to be done properly, required several pieces of test equipment that I didn't own or have access to (at least at that time).

One of the adjustments called for in the service manual requires that the outputs of two audio generators, operating at 300 Hz and 2700 Hz and at equal amplitudes, be summed and applied to the transceiver's microphone input. Monitoring the transmitted output on an oscilloscope should show the classic *cat's-eye* waveform (**Figure 1**) if everything is adjusted properly. Carrier feed through, overdriving, or misaligned filters would produce a less than perfect cat's-eye. By the way, the period of the cat's-eye is the period of the difference in frequency between the two tones.

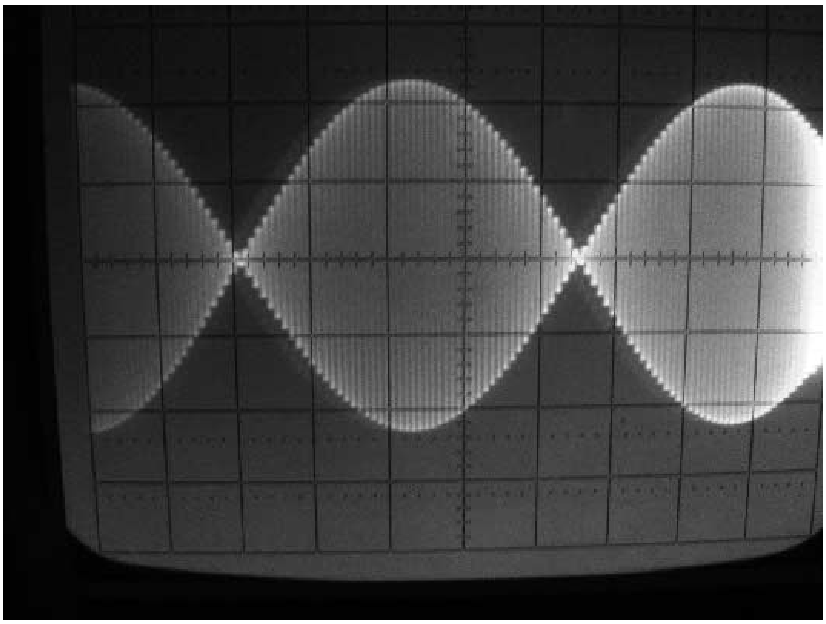


Figure 1 — Cat's-eye waveform seen when summing two equal-amplitude tones.

Lacking the two-tone test generator necessary to perform this adjustment, I decided to build my own. I designed a stand-alone instrument with two independent frequency-variable audio oscillators. One covers roughly 300 to 1100 Hz and the other covers roughly 800 to 3200 Hz. The overlap in ranges allows for any frequency pair that you're likely to need. The device can also be used around the shack as a general purpose audio oscillator. The output flatness for each tone is ± 0.1 dB as measured on an HP 403B AF voltmeter.

The frequency of each oscillator is adjusted by the front panel TONE A or TONE B controls. (See the front and rear panels in **Figures 2** and **3**.) A fixed amplitude (1 Vp-p) signal is made available at the rear-panel COUNTER socket in order to read the exact frequency of the chosen tone on an external frequency counter.

TONE SELECT lets you select either Tone A alone or Tone B alone, or to sum them. The OUTPUT LEVEL control sets the signal level at the rear-panel OUTPUT and SCOPE sockets (they're connected in parallel). The output to the transmitter can be adjusted from 0 to

Figure 2 — Front panel view of the two-tone generator. Note the individual controls for Tones A and B.

Figure 3 — Rear panel view of the generator.

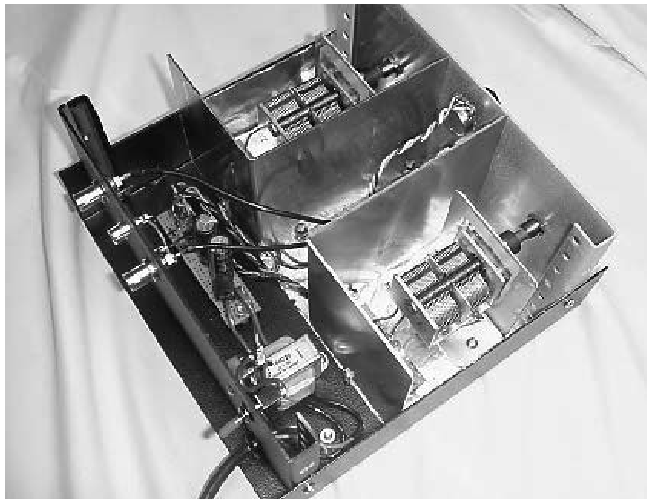


Figure 4 — A look inside the two-tone generator.

30 mVp-p (single tone) or 0 to 60 mVp-p (two-tone) at an impedance of 600 Ω.

As you can see in **Figures 4** and **5**, each oscillator is housed in a separate compartment containing a variable capacitor and a small circuit board located behind the front panel. It's important to keep each oscillator very well shielded due to its sensitivity to cross-talk and pickup.

Between the oscillator compartments are the OUTPUT LEVEL and TONE SELECT controls and a circuit board containing the summing and output amplifiers.

The fuse, power switch, power transformer and power supply circuit board are located along the rear of the case.

Circuit Description

The schematic diagram of the oscillator, summer and output amplifier circuitry is shown in **Figure 6**. Only three TL082 operational amplifiers are used. They are very high input impedance devices and each consumes less than 10 mA at a supply voltage of ± 12 V.

Each oscillator, U1A and U2A, uses the Wien Bridge (or RC) configuration with an incandescent lamp in the negative feedback loop to stabilize the output amplitude. This scheme was popularized back in 1939 in Bill Hewlett's HP 200A audio oscillator.[1] It provides a very clean sine wave with near-constant amplitude over a wide frequency range. The output of each oscillator is buffered by a unity-gain amplifier, U1B and U2B.

The frequency of operation of each oscillator is determined by fixed values of resistance and an AM broadcast-band receiver dual-section variable capacitor. Since both the stator and the rotor of each capacitor is floating above ground (not the usual hook-up in its intended use), the entire capacitor must be insulated from ground.

Variable capacitors C1 and C2 have been padded to provide about a 4:1 frequency range for each oscillator.

Resistors R1 and R2 are both made up of series connected 1 MΩ and 301 kΩ resistors. R5 and R6 are standard parts values. All are 1%, metal-film types.

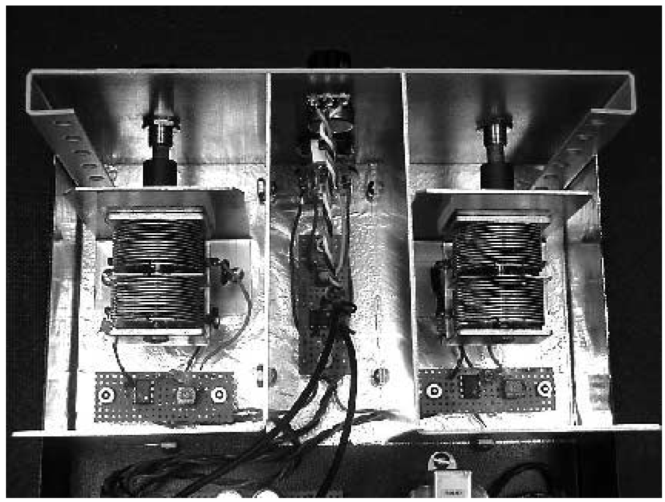


Figure 5 — A close-up view showing the oscillator compartments and summer circuit board.

The exact values of the resistors and capacitors in the positive feedback loop aren't too critical as long as they're the same. Buy extra parts and pick the closest pairs using a resistance-capacitance meter. Capacitors C3 and C5 are smaller in value than their counterparts, C4 and C6 respectively, in order to compensate for the input capacitance of the op-amps.

The trimpots in the negative feedback loops, R3 and R7, are preset for about 250 Ω and ultimately adjusted for a signal of exactly 10 Vp-p at the output of the buffer amps.

A two-pole, three position rotary switch, S1, is used to apply the desired tone (A, B or both) to the summing amplifier, U3A.

The OUTPUT LEVEL control, R14, unity gain amplifier, U3B, and voltage divider, R15 / R16, provide suitable microphone drive to most transceivers.

The full-wave power supply (**Figure 7**) is configured using a center-tapped transformer, T1 and bridge rectifier, U6. The large-value filter capacitors, C7 and C8, and high-gain voltage regulator ICs, U4 and U5, produce very clean and stable supply voltages for the op-amps. No heat-sinking is required.

Construction Notes

I housed the instrument in a Hammond 1458D4 instrument case, measuring 4 × 8 × 8 inches. I removed a bit of paint from the non-visible areas of the case screw holes so that the top, bottom and ends were all electrically bonded. I also lined the case halves above and below each oscillator compartment with household aluminum foil for better shielding.

The oscillator compartment walls both measure 3 ⅞ × 3 × 4 ½ inches. I used spade bolts to secure them to the bottom of the case. The capacitor mounting brackets both measure 2 ¾ × 2 ½ × 1 ¼ inches. These parts are made from 1⁄16-inch thick sheet aluminum.

[1]See page 6 of the PDF at **www.hparchive.com/Manuals/HP-200-IRE-Article.pdf**. This was the start of Hewlett-Packard.

Insulating the variable capacitors is achieved by using a pair of 1 ½ × 1 ½ × ¼ inch thick spacer plates (cut from scrap plastic), nylon mounting screws and phenolic shaft-extenders between the capacitor shafts and the tuning knobs.

Be very precise in laying out the capacitor mounting brackets and the shaft bushings. This will ensure smooth rotation of the capacitor shafts.

The phenolic shaft-extenders and front panel brass shaft-bushings are Abbatron (formerly H. H. Smith) part numbers 130 and 119, respectively.

All four circuit boards are Veroboard copper-clad board mounted on ½ inch standoffs. Any perf board would be equally suitable.

The power and signal leads were brought out of each oscillator compartment by filing a small "mouse-hole" on the bottom edge of the compartment wall.

The front and rear panel markings were done using a Brother P-Touch label maker.

Testing

The various circuits can be individually bench-tested prior to final assembly.

Start with the power supply. The outputs should both be around 12 Vdc. Use a 270 Ω, 1 W resistor as a dummy load to make sure that each side of the supply holds up. Each oscillator can then be individually tested and adjusted using a bench power supply (or the instrument's own supply) and an oscilloscope. Slowly turn the variable capacitor through its range. You may notice a slight "bobble" in the output signal's amplitude for a second or so as you change frequency. What's important is that oscillation continues at every position of the shaft. Otherwise, it's quite likely that, somehow, a pair of plates

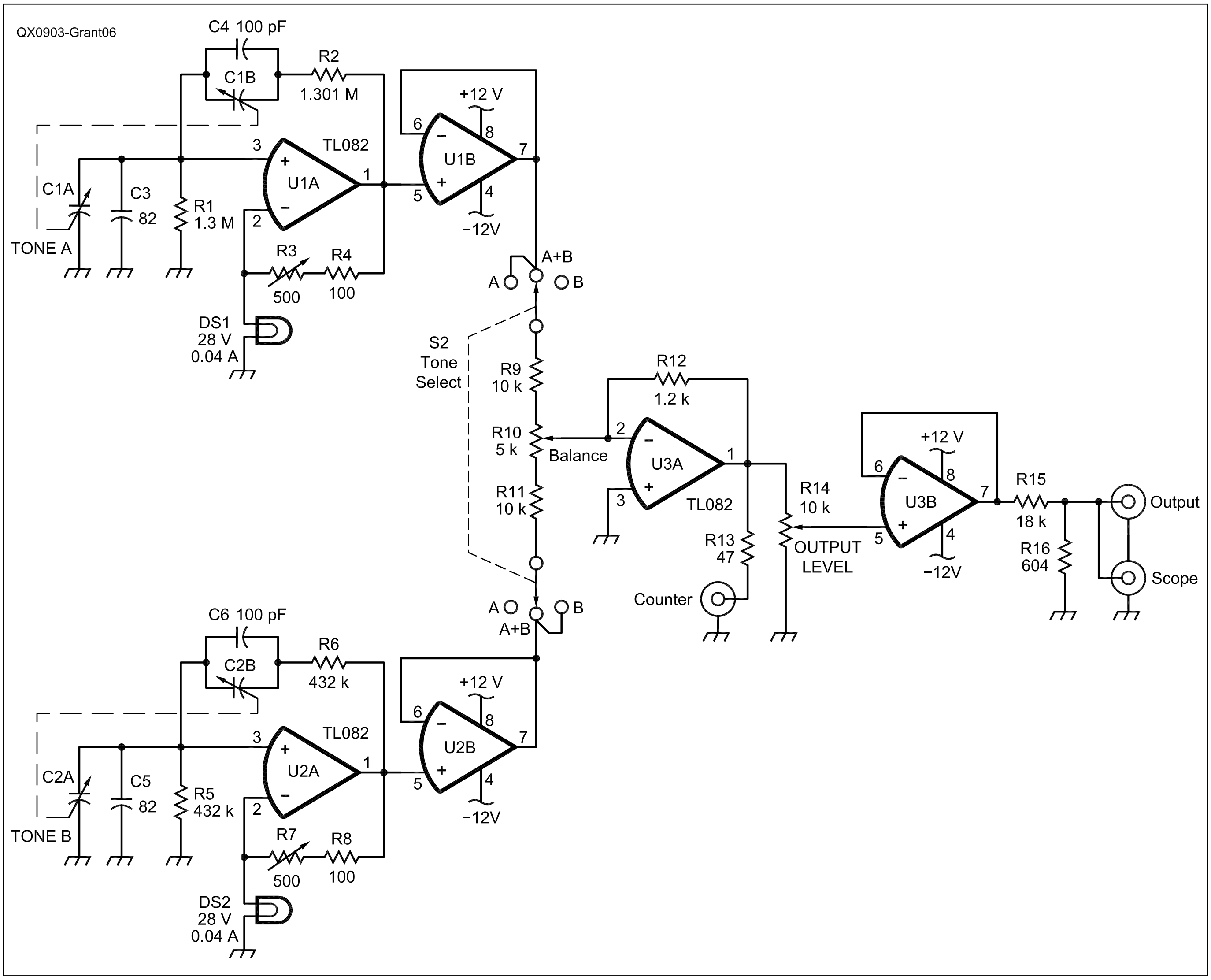


Figure 6 — Schematic diagram of the oscillators, summer and output amplifier. Most parts are available from Ocean State Electronics (OSE) at www.oselectronics.com/; tel 401-596-3080; fax 401-596-3590. Ocean State part numbers shown where applicable.

C1, C2 -- dual 13-380 pF air variable capacitors, OSE BC13380.
C3, C5 -- 82 pF silver mica capacitors, OSE CSM82.
C4, C6 -- 100 pF silver mica capacitors, OSE CSM100.
Lamp1, Lamp2 -- 28 V, 40 mA, OSE L-327 or equivalent.
R1, R2 -- 1.301 MΩ, 1% resistors, OSE RM1.00M in series with RM301K.
R4 – 100 Ω resistor, OSE RM100.
R5, R6 – 432 kΩ, resistors 1%, OSE RM432K.
R9, R11 – 10 kΩ resistors, OSE RQ10K.
R12 -- 1.2 kΩ resistor, OSE RQ1.2K.
R13 – 47 Ω resistor, OSE RQ47.
R15 – 18 kΩ resistor, OSE RQ18K.
R16 – 604 Ω resistor, OSE RM604.
R3, R7 – 500 Ω trimpot, OSE OFA52.
R10 – 5 kΩ trimpot, OSE OFA53.
R14 – 10 kΩ potentiometer, OSE PQ10K.
S2 -- rotary switch, 2 pole, 3 position, OSE 30-15203.
U1, U2, U3 -- dual op-amps, OSE TL082CN.
Abbatron #130 insulated shaft extenders and #119 panel bushings, or equivalents.

are shorting. This may be due to dirt, wire strands, solder debris or mechanical damage to a plate. Regardless of the cause, it must be corrected.

When you're satisfied that each oscillator is working properly, set the variable capacitors at half mesh and adjust R3 and R7 for 10 Vp-p signals at the outputs of U1B and U2B respectively.

Next, connect the oscillator outputs to the summer and output amplifier board and check for proper operation. The balance trimpot, R10, is then adjusted for equal signal amplitudes at the COUNTER output when switching between tones A and B.

When all is well, proceed with the final assembly. I used shielded wire made of RG-174/U coax to feed the output sockets and 24 gauge hook-up wire for everything else.

Conclusion

This unit has been a welcome addition to my test bench. It's compact and quite versatile. The best part is that the heart of the design, the variable capacitors, can be found in countless old, discarded AM radios.

An active ARRL member, Ken Grant, VE3FIT, was first licensed in 1962 at the (then minimum) age of 15 and holds the Canadian Advanced Amateur Certificate. He resides in West Hill, Ontario, a suburb of Toronto, with his wife, Marlene, and two of their three sons. Ken graduated as an electronics technician in 1967 and has worked ever since in Electronics Development and Biomedical Engineering at the Princess Margaret Hospital (for cancer treatment and research) in Toronto. His Amateur Radio interests include home-brewing, chasing DXCC on 6 meters and rag-chewing on both SSB and CW.

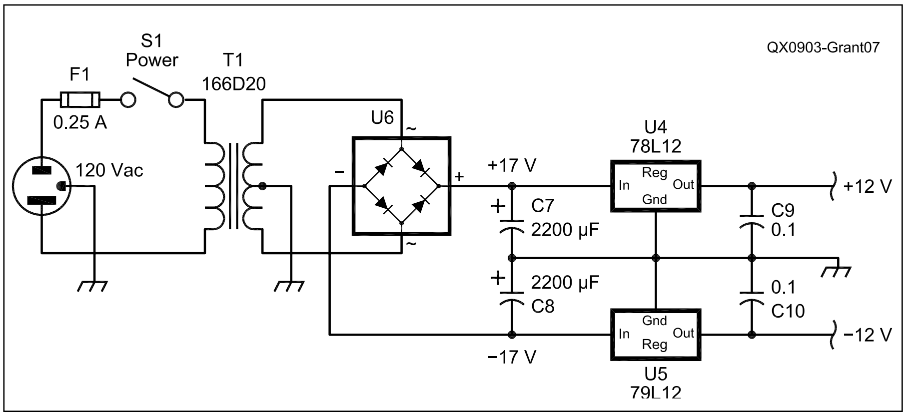


Fig. 7 — Schematic diagram of the generator power supply. Most parts are available from Ocean State Electronics (OSE) at www.oselectronics.com/; tel 401-596-3080; fax 401-596-3590. Ocean State part numbers shown where applicable.

C7, C8 -- 2200 µF, 25 Vdc electrolytic capacitors, OSE CER2200-25.
C9, C10 -- 0.1 µF tantalum capacitors, OSE CT.1-50.
F1 – 1/4 A fuse, OSE FSM14.
S1 -- toggle switch, OSE 10002.
T1 – 20 V, 0.1 A, center tapped transformer. Hammond 166D20 or equivalent.
U4 -- +12 V regulator, OSE 78L12.
U5 -- −12 V regulator, OSE 79L12.
U6 -- bridge rectifier, 2A, 50V, OSE RC201.

QEX

Rudy Severns, N6LF

PO Box 589, Cottage Grove, OR 97424; n6lf@arrl.net

Experimental Determination of Ground System Performance for HF Verticals Part 3 Comparisons Between Ground Surface and Elevated Radials

Experimental results from another of the author's antenna experiments.

Over the years there has been a great deal of discussion regarding the relative merits of a vertical antenna with a few elevated radials versus one with a large number of radials either lying on the ground or buried just below the surface. *NEC* modeling predicts that as few as four radials, a few feet above ground, will provide as efficient a ground system as a large number of on-ground radials. Whether this prediction is valid is a matter of some dispute. Resolving this issue is important for amateurs using HF vertical antennas.

The first segment of the experiment was a comparison of the performance of a ¼-wavelength vertical antenna with a large number of ground surface radials (64) to one with only four elevated radials. From the results in segment one it appeared that elevated radial systems for HF verticals have some merit. But there are a number of different ways to implement an elevated radial system. The purpose of the second segment of the experiment was to evaluate the relative performance of several different elevated radial schemes.

Segment One

All measurements were made at 7.2 MHz using a 33.5 foot tubular aluminum vertical antenna. The experiment began with sixty four, 33 foot no. 18 AWG insulated wire radials lying on the ground surface. The antenna was insulated from ground and used a common mode choke (balun) in the feed line. With a height of 33.5 feet and 64 radials, the vertical was close to resonance at 7.2 MHz.

During the experiment, |S21| (magnitude of the transmission gain, see Part I of this series)[1] and the input impedance at the feed point (Z_i) were measured and recorded as the radial system was changed. The experiment began with 64 radials lying on the ground surface. Without changing the height of the vertical, |S21| and Z_i were measured as the radial number was reduced in the following sequence: 64, 32, 16, 8, 4. The next step was to make a series of measurements, beginning with the four radials on the ground and then elevating the radials and the base of the vertical to 6 inches, 12 inches and finally 48 inches. At the 48 inch height, a measurement of the current division between the radials was made.

This entire sequence was repeated three times on different days. The results did not change significantly between test runs.

[1]Notes appear on page 32.

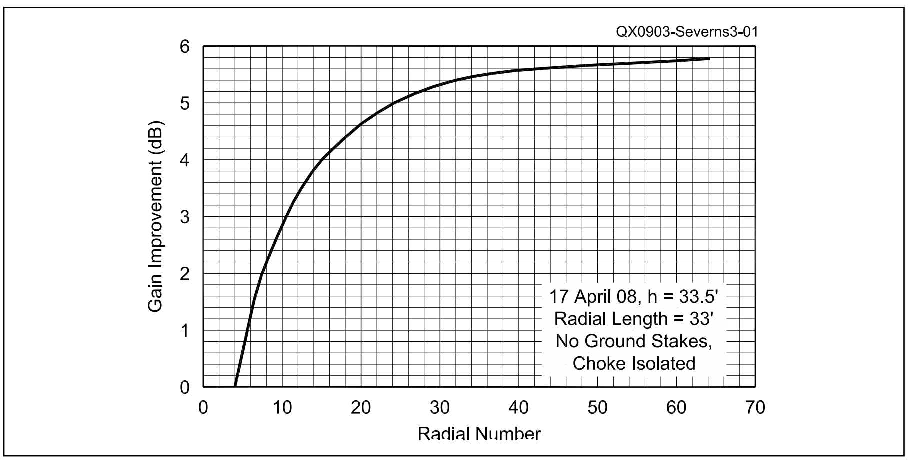


Figure 1 — |S21| as a function of radial number. All radials are lying on the ground surface.

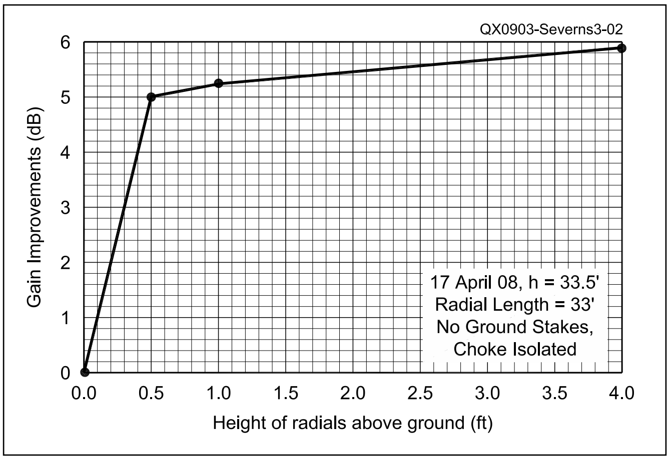


Figure 2 — |S21| with 4 radials and the antenna base at different heights.

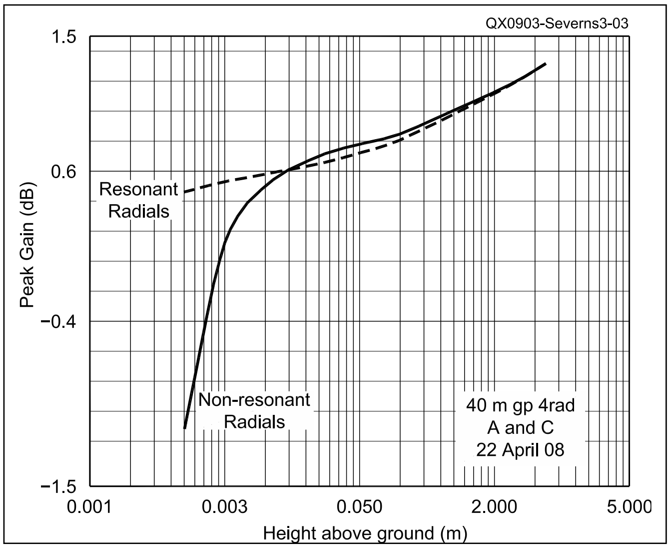


Figure 3— *NEC* prediction of peak gain versus radial height for 4 radials.

Experimental Results

The observed variations in |S21| as radial number and height were changed are shown in Figures 1 and 2. In the graphs, |S21| has been normalized (0 dB) to the value for 4 radials lying on the ground surface, so that the graphs show the *improvement* in dB as either the radial height or number were increased.

From Figure 1, we see that with 64 radials lying on the ground surface |S21| = +5.8 dB. From Figure 2, for four radials and the base of the antenna elevated 48 inches above ground, we see that |S21| = +5.9 dB. The difference is only 0.1 dB. For any practical purpose, the two ground systems are equivalent, which is in accord with *NEC* predictions.

The large change in |S21| with radial number in Figure 1, which is predicted by *NEC*, is mostly the result of additional loss caused by resonances present in sparse radial screens. This effect was discussed in Part 2 of this series.[2]

The very large change between 0 inches and 6 inches in elevation shown in Figure 2 was also predicted by *NEC*. A typical prediction from *NEC* of peak gain versus radial height is shown in Figure 3.

The data line labeled "nonresonant radials" corresponds to constant length (33 feet) radials, which are not shortened to compensate for the effect of the soil characteristics on the radial resonant frequency. The other data line shows the effect of adjusting the length of the radials to re-resonate the antenna as the height above ground is altered.

Typical measured values for Z_i during the experiment are given in Table 1.

The measured current division between the radials, normalized to 1 A of total base current, is given in Table 2.

The radial current asymmetry was small

Table 1
Experimental Values for Feed Point Impedance.

Number of Radials	*Radial Height (Inches)*	*Z_i (Ω)*
64	0	39.7 – *j* 1.2
32	0	42.9 + *j* 2.1
16	0	56.1+ *j* 6.2
8	0	85.5 + *j* 8.0
4	0	137 + *j* 14.9
4	6	43 + *j* 6.4
4	12	40.6 + *j* 0.08
4	48	34.8 – *j* 9.7

Table 2
Current Distribution in the Radials When Elevated to 48 Inches.

Radial Number	*Relative Current (A)*
1	0.235
2	0.271
3	0.247
4	0.247

Table 3
Gain Comparisons With One and Four Radials.

Radial Number (dB)	*Azimuth (Degrees)*	*Peak Gain (dBi)*	*Elevation (Degrees)*	*Delta from 4 Radial Case (dB)*	*Delta from 4 Radial Case*
4	0	+1.15	21.4	0	X
4	0	–1.12	8	X	0
1	0	+0.38	22.8	–0.77	X
1	0	–2.04	8	X	–0.92
1	90	–0.36	22.8	–1.51	X
1	90	–2.79	8	X	–1.67
1	180	–2.19	19.8	–3.34	X
1	180	–4.59	8	X	–3.47

enough to not have any meaningful effect on |S21|. Earlier measurements on radial systems with 64 radials, lying on the ground surface, also showed little asymmetry in the current division.

Effect Of Radial Current Division Asymmetry

As shown by Weber, it is very common for the current division between the radials in an elevated radial system to be unequal, especially if there are only a few radials.[3] This asymmetry can affect the radiation pattern, and may possibly explain some of the variation in earlier comparisons. For this reason, I was very careful to minimize that asymmetry.

To get worst case estimates of the effect of current asymmetry on the pattern, I did some *NEC* modeling. Two models, the first with four radials and the second with one radial, are shown in Figures 4 and 5.

Comparisons between the peak gain and the gain at 8° elevation are given in Table 3. I have shown the peak gain and its associated angle, and also the gain at 8°, which corresponds to the angle to the test range receive antenna. As Table 3 shows, that makes little difference in the magnitude of the pattern distortion.

The worst case signal reduction from the four-radial case is at the 180° azimuth, with one radial. If all the current were in the radial pointing away from the receive antenna, the signal strength would be a bit over –3 dB from the case where all four radials had the same current. I examined models with 1, 2, 3 and 4 radials, but the worst case is for a single radial. That is hardly surprising.

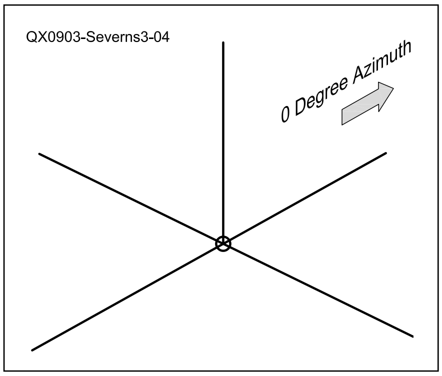


Figure 4 — Four elevated radials, 48 inches above 0.015/30 soil.

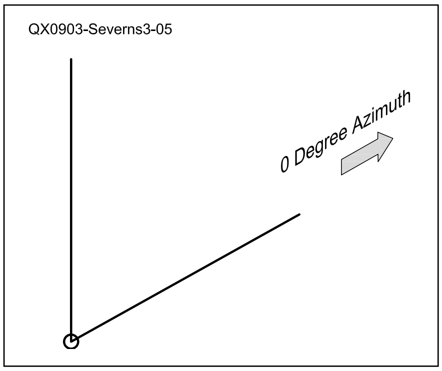


Figure 5 — One elevated radial, 48 inches above 0.015/30 soil.

Segment 2

The "standard" elevated radial scheme has four or more radials elevated above ground by 4 feet to 10 feet, with the base of the vertical antenna also elevated so that the radial fan is essentially flat. For a variety of practical reasons, however, somewhat different radial configurations are often used and it is of some interest to see what effect these variations have on the performance of the antenna.

Description of the Experiment

All the experimental runs were done with four 35 foot radials (except as noted), the length of the vertical set to 34 feet and a test frequency of 7.2 MHz. The antenna, including radials, was isolated from ground with a common mode choke (balun) in the feed line. Measurements of |S21| and Z_i were made for each test configuration.

The following configurations were tested:

1) Radials and antenna base elevated at 48 inches above ground.

2) The far end of the radials at 48 inches sloping down to the base at ground level.

3) A "gullwing" configuration as suggested by Dean Straw, N6BV, and later extensively modeled by Al Christman, K3LC.[4] The base was at ground level with the radials rising from the base at a 45° angle until they reached 48 inches above ground. The rest of the radials beyond this point were kept at 48 inches above ground from this point out to the far ends.

4) Radial lengths cut to 17.5 feet (≈ ⅛-wavelength). Radial and base height set to 48 inches. Antenna resonated with a 2.2 μH inductor.

5) For reference purposes, a run was made with the radials lying on the ground surface and the antenna base at ground level. This was done as a check because segment one of this experiment had been done earlier and ground conditions at the site had changed. Also a slightly different radial length was used (35 feet versus 33.5 feet).

Experimental results

The experimental results are summarized in Table 4. The values for |S21| were normalized by setting the value for configuration 1 to 0 dB and the rest to the difference between them and configuration 1. A line of data from an earlier experiment has been added for comparison. (See Note 2.)

As a check, for configuration 1, the current division between the radials was measured. Those results are summarized in Table 5.

Comments on Segment Two

The most important observation is that radically changing the radial geometry does not seem to have a major impact on performance (|S21|).

Table 5
Measured current division between radials, normalized to 1A total base current.

Radial number	*Normalized Current (A)*
1	0.249
2	0.269
3	0.260
4	0.221

Table 4
Experimental Results

Configuration Number	*|S21| Normalized (dB)*	*Z_i (Ω)*	*Test Configuration*
1	0	39 + *j* 6.3	Base and 4 radials elevated at 48 inches
2	–0.47	36 + *j* 6.2	Base at ground level, radials ends at 48 inches
3	–0.65	29 – *j* 11	Gullwing, base at ground level radial ends at 48 inches
4	–0.36	39 + *j* 0.9	Base and radials at 48 inches radial length = 17.5 feet 2.2 μH inductor to resonate
5	–5.19	132 + *j* 22	Base and radials on ground surface, four 35 foot radials
Earlier Experiment (See Part 2)	–1.79	51 + *j* 1	Base and radials on ground surface, Four 21 Foot Radials

Cutting the radial lengths in half (configuration 4) and adding a small loading inductor reduced the gain by only –0.4 dB. The use of shorter radials has been suggested by Weber (see Note 3) and Moxon to either make the radial screen footprint smaller and/or reduce asymmetry in the current division between radials.[5]

I was surprised to see that the gain reduction for the gullwing configuration (configuration 3) was slightly worse than simply running the radials straight up to the far end (configuration 2). It may have something to do with the higher feed point impedance in configuration 2. In the case of the gullwing, the radials rise close to the vertical element, resulting in some cancellation between the vertical element and radial currents depressing the feed-point resistance. We see a similar effect in top-loaded antennas with sloping wires. From the standpoint of keeping the radials above head height for safety reasons, the gullwing is more attractive than just sloping up the radials.

It would seem that anything done to get the radial wires away from ground makes a great improvement as you can see from configuration 5, where the radials are lying directly on the ground surface. Even using shorter, resonant radials on the ground surface is not as effective as simply elevating the radials. Modeling and experimental work shows that you don't have to get very high to make a substantial improvement but greater heights are used for safety reasons to keep the radials above head height.

One thing missing from this experiment was the use of more than four radials. An earlier experiment which compared four elevated radials to eight in configuration 1, showed very little difference in |S21| (about + 0.2 dB). The advantage of more radials is not so much improved efficiency but rather reduced chances for radial current asymmetry and a lower Q, which can improve the SWR match bandwidth.

Summary

The experiments seem to show that a few elevated radials can work well as a replacement for a large number of ground radials. The experiments also show that alternate elevated radial geometries can work nearly as well as the "standard" and may have practical advantages.

Certainly this set of experiments does *not* completely resolve the debate regarding a large number of ground radials versus a few elevated radials, but it does lend some credence to the *NEC* modeling. To finally resolve these questions we need other experimenters to repeat these and/or similar experiments. We should also recognize that these experiments were done at a particular site, which has good to very-good soil. Repeating the tests over other soils, particularly poor ones, would be of considerable interest. It is at least possible that larger differences between the ground surface and elevated radials might be seen.

Even if these tests and *NEC* modeling are in fact correct and a few elevated radials can, in principle, provide equivalent performance to a large number of ground radials, this does not mean we should dash out and convert all our ground systems to four elevated radials. Because of their much higher Q, elevated radial systems are subject to a number of ills. They are very sensitive to details of layout, soil characteristics, nearby conductors, coupling to feed lines, and other factors. Like ground radials, elevated radial systems work much better if the screen is not too sparse: in other words, try to use 12 or more radials. You will be much happier.

Notes

[1]Rudy Severns, N6LF, "Experimental Determination of Ground System Performance for HF Verticals, Part 1," *QEX*, Jan/Feb 09, pp 21 - 25.

[2]Rudy Severns, N6LF, "Experimental Determination of Ground System Performance for HF Verticals, Part 2," *QEX* Jan/Feb 09, pp 48 - 52.

[3]Dick Weber, K5IU, "Optimum Elevated Radial Vertical Antennas," *Communication Quarterly*, Spring 1997, pp 9 - 27.

[4]R. Dean Straw, N6BV, "Antennas Here Are Some Verticals On The Beach," *ARRL Antenna Compendium, Vol 6*, pp 216 - 225.

[5]L. Moxon, G6XN, "Ground Planes, Radial Systems and Asymmetric Dipoles," *ARRL Antenna Compendium Vol 3*, pp 19- 27.

Rudy Severns, N6LF, was first licensed as WN7WAG in 1954 and has held an Extra class license since 1959. He is a consultant in the design of power electronics, magnetic components and power-conversion equipment. Rudy holds a BSE degree from the University of California at Los Angeles. He is the author of two books and over 80 technical papers. Rudy is an ARRL Member, and also an IEEE Fellow.

QEX

Ron Skelton, W6WO

4221 Gull Cove Way, Capitola, CA 95010; w6wo@k6bj.org

Exploring Near-End-Fed Wire Antennas

The author investigated near-end-fed wire antennas by modeling his design, then built a 40 m version to verify the modeled performance.

This article describes wires fed close (about 10%) to one end in contrast to center-fed, end-fed and off-center-fed antennas. Center-fed resonant antennas present a typical impedance in the range of 50 to 75 Ω, so coaxial cable is almost universally used as the transmission line. End-fed (Zepp) antennas presenting an input impedance of about 1000 Ω are usually fed with open wire line and justify an antenna tuner. Between these extremes, antennas fed roughly 30% from one end are derived from a design introduced by Lauren Windom, W8GZ, in his famous *QST* paper on "Ethereal Adornments."[1]

The feed-point impedance chosen by some modern off-center-fed (OCF) designs is approximately 300 to 450 Ω. These are fed with either balanced line or by coax using a 6:1 or 9:1 transformer at the feed point. Other designs, such as the one presented in the June 2008 issue of *QST* by Rick Littlefield, K1BQT, focus on a 140 Ω point that requires a 2.8:1 impedance match.[2]

My current interest is in wires fed about 10% from one end at a point where the impedance is 800 Ω. I distinguish these as Near-End-Fed (NEF) antennas. The idea came from realizing that it was simple to construct a wide-band transformer with a turns ratio of 4:1 (an impedance ratio of 16:1, to transform the 800 Ω to 50 Ω). Having achieved rather poor results from transformers using transmission line principles I began exploring binocular ferrite cores with a single turn primary and 4 turn secondary. Such transformers have a long history in matching the low output impedance of transistor amplifiers to 50 Ω.[3] I was delighted to measure transmission loss <0.3 dB, return loss >25 dB, and common-mode impedance well in excess of 500 Ω across the range of 3 to 30 MHz. This performance is only slightly degraded from 1 to 50 MHz.

[1]Notes appear on page 34.

The primary consists of a copper tube with a diameter of 0.25 inch, inserted into each of the binocular cores, joined at one end and open at the other as shown in Figure 1. The four small copper tubes in each larger tube were intended to increase the coupling between primary and secondary but they did not make a significant difference in performance. The secondary is simply 4 turns looped through the large tubes.

Ferrite transformers are liable to become lossy and saturate when used at high current locations such as the center point of a dipole. In a NEF design, lower currents and higher voltages are present. Therefore, core saturation is less likely and certainly not an issue at 100 W with the cores shown here. With Teflon wires used in the secondary, voltage breakdown should also not be a concern.

Figure 2 shows the transformer mounted in its plastic box, with suspension and strain relief eyebolts. Note the primary connection to the coax connector on the right and the four-turn secondary wires exiting the connected cores to the left

To learn what I could about feed-point impedance, I browsed through many decades of *QST* and found one extremely useful technical article by William Wrigley, W4UCW, in the February 1954 issue.[4] There is also a rich history on the evaluation of OCF designs and shield currents, notably by L. B. Cebik, W4RNL(SK) and by Jack Belrose, VE2CV.[5, 6] Because I found no articles on practical NEF designs, I am tempted to claim that they are a novel variety.

Whenever a wire balanced with respect to ground is fed off-center or via an unbalanced cable, the feed line carries unbalanced (common-mode) current and will radiate as part of the antenna. This may be useful as there will be both vertical and horizontal radiation. Feed line radiation is generally undesirable, however, because it can cause "RF in the Shack," more local noise ingress, alter the impedance seen by the radio and modify the directivity and takeoff angle of the antenna.

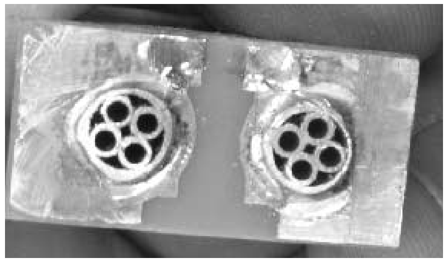


Figure 1 — This is an end view of the transformer core.

Because of their unbalanced nature, practical OCF or NEF antennas may be less predictable than well-balanced antenna systems. It is worth noting, however, that well-balanced antennas and their feeders are the exception rather than the rule. The first line of defense in mitigating feed line currents is a balun at the feed point. To be effective, a balun must introduce sufficient impedance in the path of common-mode currents and it is curious that vendors do not usually specify what that figure is for their products.

This impedance can be measured with some form of impedance analyzer, and as a general rule we need "choking" impedance at least 10× greater than the feed line — for example, 500 Ω for a typical coax cable. The transformer used in the NEF experiments exceeded 500 Ω across the HF range.

I enjoy modeling antennas (especially in the depth of winter or in the heat of summer) and NEF half wave antennas for all bands from 20 to 160 m were explored both as horizontal and sloping configurations. The outer surface of the coax outer conductor was modeled as a third wire, grounded at the base of the tower. Promising results were obtained from modeling single band antennas from 160 to 17 m.

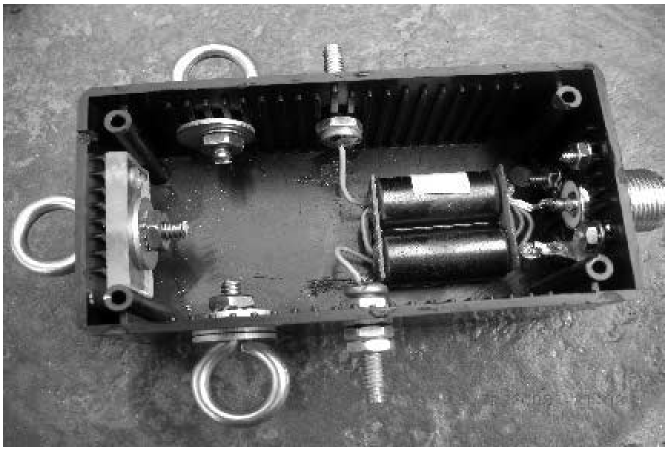


Figure 2 — The 4:1 turns ratio transformer housing.

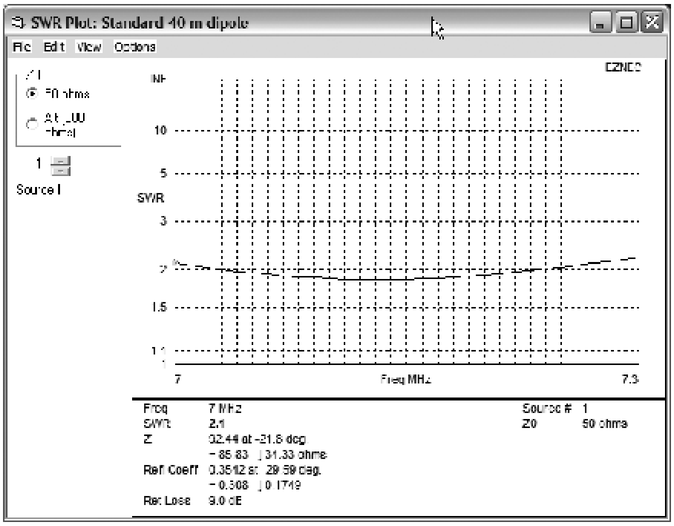


Figure 4 — This is a conventional center-fed dipole SWR plot.

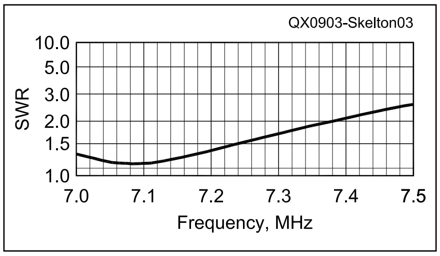


Figure 3 — This is an SWR plot for the NEF40, 40 ft above ground.

Multi-band operation, however, was generally disappointing. Naturally, there is no guarantee a real antenna will perform just like its model, but models do provide a great starting point.

The results shown in Figure 3, from modeling a 40 m NEF half wave dipole, are typical of those obtained for other bands. This model was 68 ft long at 40 ft elevation, with the feed point 10.5 ft from one end. This antenna as a sloper, produced a similar SWR and its vertical pattern showed slightly more energy radiated at low angles

Results similar to the NEF40 are obtained at 30, 20 and 17 m. For the 60, 80 and 160 m designs, the bandwidth over which a low SWR is achieved is progressively reduced, as is the case for conventional dipoles. SWR for the NEF designs were as good as, or better than, conventional center-fed dipoles. Radiation patterns were the same for both types. Figure 4 shows SWR for the equivalent center fed dipole.

Even with an effective balun, currents induced on the coax shield can cause it to radiate. I wanted to investigate the extent of this shield current and how it may be minimized. Figure 5 is the model for the NEF40 without a balun and with the coax shield modeled as a vertical wire grounded at the base. The high degree of shield current guarantees that this coax will become a significant part of the antenna. Note also that the feed point current has been slightly distorted and this affects the input impedance.

I found the most effective way of reducing shield current is to have a balun/transformer at the feed point and an in-line ferrite choke close to ground. Incidentally, I consider all baluns to be transformers in a sense, but transformers may or may not be used as baluns. Figure 6 illustrates the reduced shield current

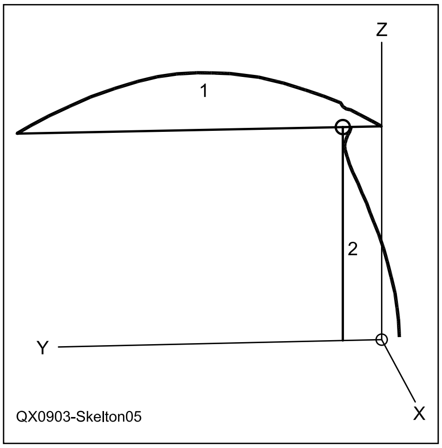


Figure 5 — NEF40 with coax shield modeled as a vertical wire to ground.

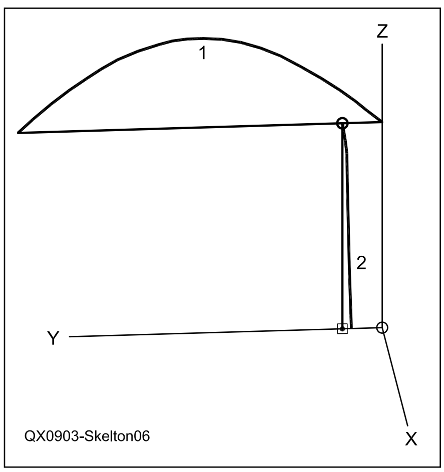


Figure 6 — NEF40 showing reduced shield

I believe that antenna models are very valuable as a starting point, and the results of an actual 40 m NEF antenna came close to the model. It was, however, sensitive to the proximity of my tower and for the best SWR it was necessary to suspend the feed point away from my tower by 20 ft.

In conclusion, the concept and usefulness of a NEF antenna fed at an 800 Ω point has been substantiated to my satisfaction. Getting the best results required spending more time than usual in positioning the antenna with respect to its environment. The prospect of multi-band antennas using multiples of half wavelengths and parallel dipoles deserve further attention.

Notes

[1]Loren G. Windom, W8GZ, "Notes on Ethereal Adornments," *QST*, Sep1929, pp 19-22, 84.

[2]Rick Littlefield, K1BQT, "A No-compromise OCF Dipole for Four Bands," *QST*, June 2008, pp 32-34.

[3]Norm Dye, Helge Granberg, *Radio Frequency Transistors: Principles and Practical Applications*, (Motorola Series in Solid State Electronics), Newness, 1993.

[4]W. B. Wrigley, W4UCW, "Impedance Characteristics of Harmonic Antennas," *QST*, Feb 1954, pp 11-14.

[5]L. B. Cebik, W4RNL, "The Isolated OCF Antenna, Some Less Explored Facts."

[6]Jack Belrose, VE2CV, and P. Bouliane, "The Off-Center-Fed Dipole Revisited: A Broadband, Multi-Band Antenna," *QST*, Aug 1990, pp. 28-33.

Ron Skelton, W6WO, was first licensed as G3IHP in 1951. Ron held various DX call signs in British colonies over the years. Ron is an ARRL Member as well as a Life Member of IEEE and a Fellow of the IEE. His areas of Amateur Radio interest include weak signal and digital mode communications, homebrewing and Elmering.

QEX

Mark Spencer, WA8SME, ARRL Education and Technology Program Coordinator

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

Tech Notes

Following up on his Sep/Oct 2008 *QEX* article, "SID: Study Cycle 24, Don't Just Use It," Mark Spencer, WA8SME, has been sending me some of the data he has collected. He also sent the artwork for a redesigned VLF receiver system that he plans to use with the 2009 Teachers' Institute program in the ARRL Education and Technology Project. I thought this information was worth passing along to interested readers, so this Tech Notes column is a collection of the information Mark has sent me. — Larry Wolfgang, WR1B; **lwolfgang@arrl.org**.

Sep 26, 2008

Figure 1 is a graphic of the sudden ionospheric disturbance (SID) data that I have been collecting. The data focuses on the period just before and after local dawn (–7 hours UTC for daylight savings time here in California). It appears that the SID data will be highly dependent on the time of year, mainly because of the presence of thunderstorms in the continental US and the resulting spikes due to lightning strikes. That dependence will become more clear as I gather more data, I just started collecting data in April, and during last spring and early summer the plots were relatively clean.

Of particular interest are the consistent spikes I observe during the dawn null of the plot. You would think that thunderstorms would not be that prevalent during the dawn hours. This is one thing I will have to watch for over the fall and winter months to see if these spikes are still there (they were not as numerous in the spring/early summer collections, but they were there periodically). I would suspect the spikes to be caused by thunderstorms to my east, but since they appear just in the null, that kind of rules that out.

The shift in the nulls is following the change in the sunrise time, and that is expected. I thought that the spikes might be some sort of electrical light that is triggered by sunrise; that could be a possibility, but due to my remote location, that would require a neighbor to have one heck of a dirty yard light (I do not have any yard lights or automatic devices triggered by sunrise or timers, so I am confident it is not a local signal). The closest street light is about 3 miles away so I don't think that is the cause of the signal. One of these days I am going to have to get up at sunrise to see if I can find a local yard light that turns off at sunrise and correlate the time to data.

I still am wondering if the spikes just might be due to some perturbation (holes) in the D layer as it is being ionized. Another thing I am going to try is to gather more detailed data (more data points) during the period to see what the curve looks like at higher resolution. Of course, what would be really helpful is if someone else had a plot during the period for comparison. That will come in time when more people get their stations up and running.

I am sure there is more to these SID plots than meets the eye!

Nov 7, 2008

I thought I'd pass along the data that I collected here on my SID system, shown in Figure 2. These ionospheric reactions would not be classified as SIDs but they did produce some interesting blips on the graphs. You would have to have some database experience with interpreting the graphs, because the changes are subtle.

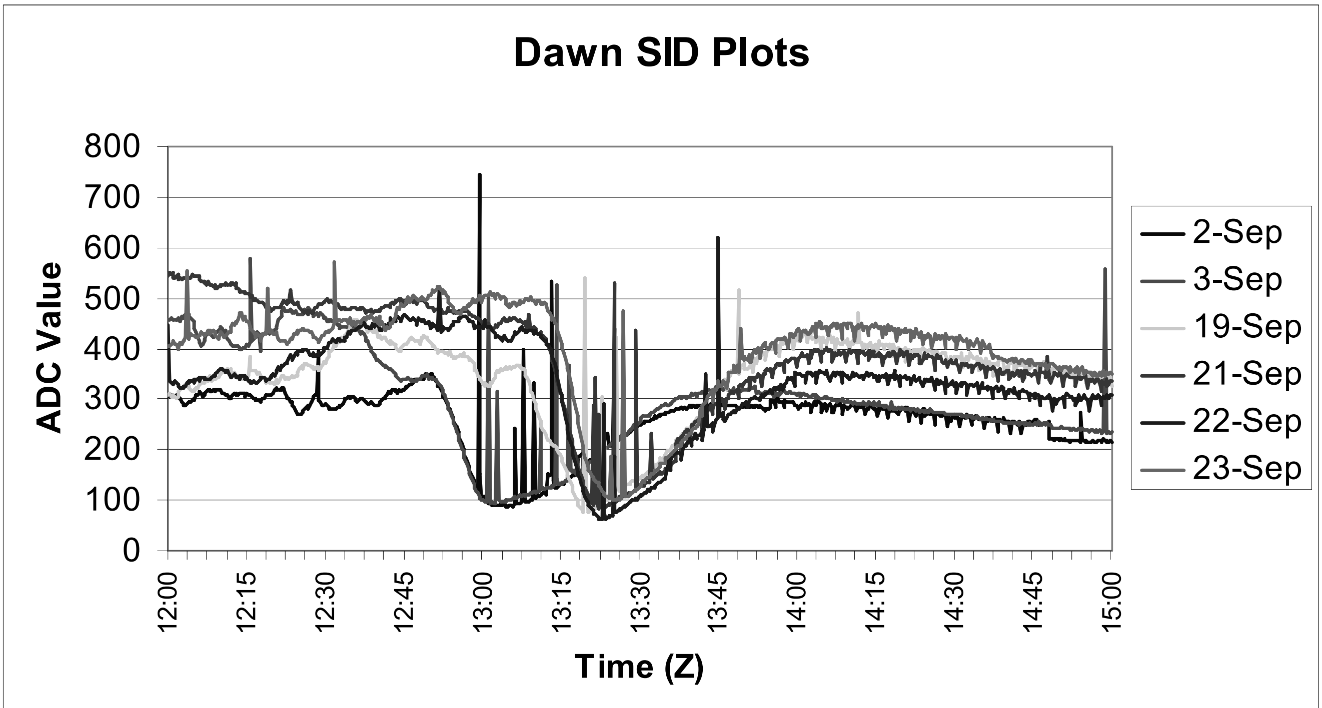


Figure 1 — This composite graph shows the analog to digital converter (ADC) output values from the SID receiver around local sunrise over several days in Sep 2008.

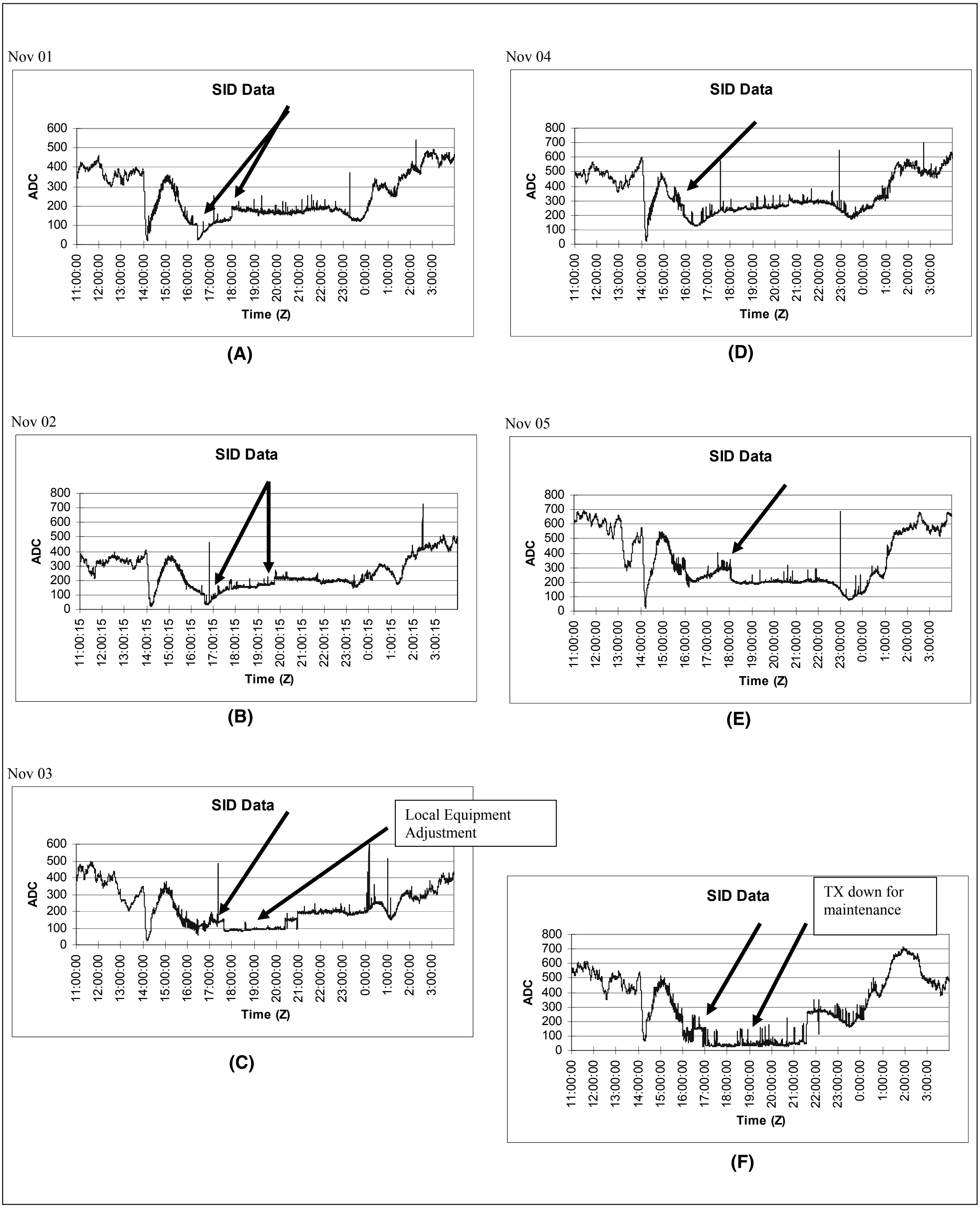


Figure 2 — These graphs show the data I collected from Nov 1 through Nov 6, 2008.

That is why scientific inquiry is mostly grunt work, collecting mounds of data so that those odd-ball changes stand out.

The arrows point to some anomalies that were probably caused by recent sunspot activity. I tried to note when the transmitter in Washington state is shut down for maintenance each Thursday. I also am experimenting with some changes to my SID station that affected one of the graphs. My graphs have those small and frequent spikes because I have the sensitivity of my system set a little high, because I am also looking at lightning strikes to see what is going on there. Others have the gain of their systems depressed to focus only on SIDs and their graphs would look cleaner. I also am sampling at a high rate to get higher resolution, a rate probably much higher than needed to study SIDs, sort of like using a microscope to look at mountain scenery. Too much resolution sometimes creates a fog that covers the freight train headed at you.

I am about to abandon my experiments with using the system for lightning strikes, because there is just too much data for me to handle with so little time!

One of the important points of all this is that there is more to ham radio than contacts and contests. Contacts and contests for schools have a place, but frankly those other aspects of the hobby/service, are in many cases more important.

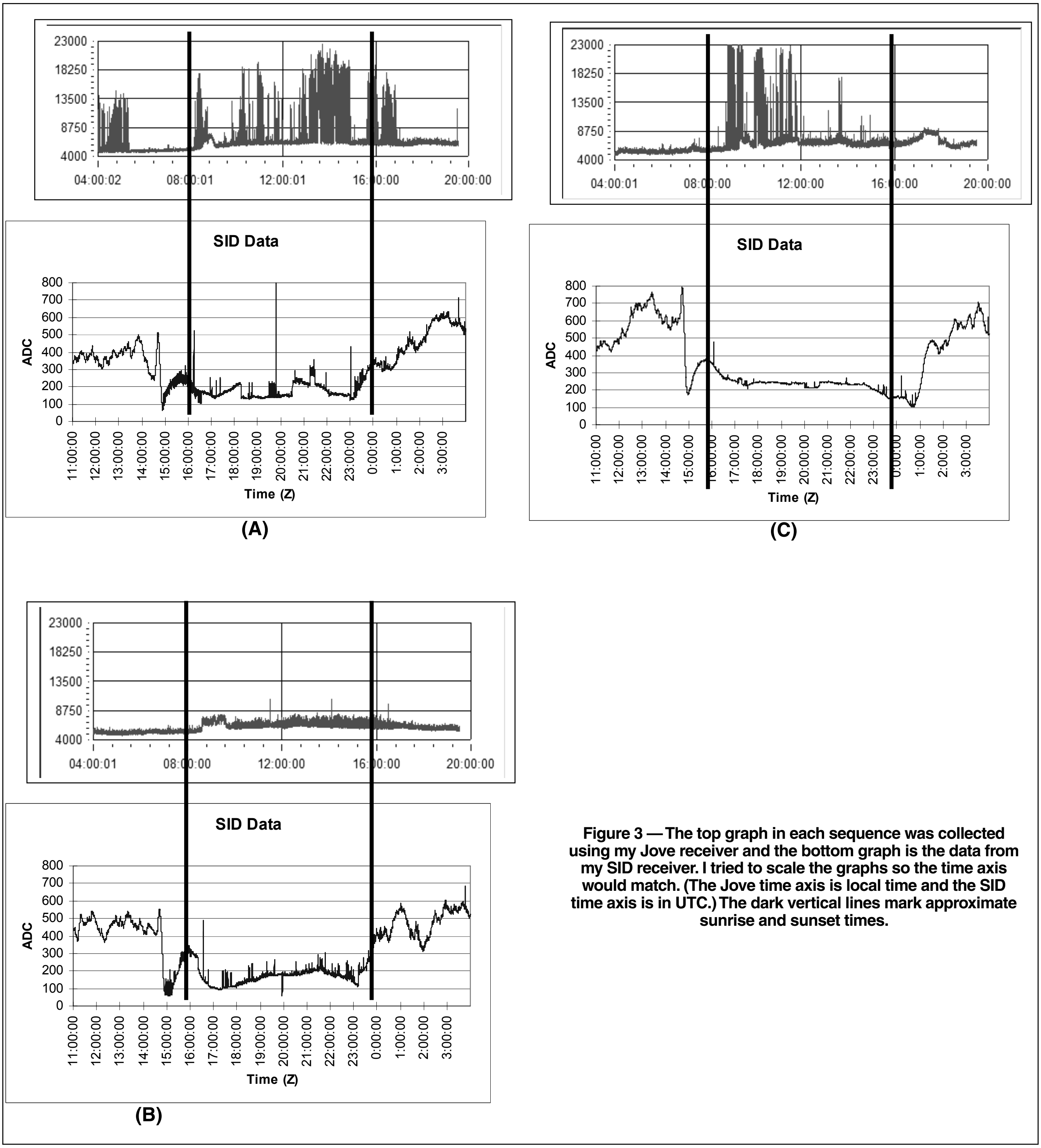


Figure 3 — The top graph in each sequence was collected using my Jove receiver and the bottom graph is the data from my SID receiver. I tried to scale the graphs so the time axis would match. (The Jove time axis is local time and the SID time axis is in UTC.) The dark vertical lines mark approximate sunrise and sunset times.

One point to make is that I don't think anyone really knows what is going on with the data or how to interpret it. My commentary is just my thoughts on what I suspect is going on and is by no means a professional judgment. I have found that lightning strikes are probably the cause of the high spikes and those strikes could be just about anywhere on the globe (more likely within the continental US). I am also detecting what appear to be gross seasonal changes in the shapes of the curves. I haven't collected enough data to see any real trends there.

Dec 6, 2008

I've been working with the SID data for a while, trying to tweak the station to make it more reliable and less labor intensive to gather the data. I also have been working with how to exploit the data using some standard software tools. I created a Microsoft *PowerPoint* presentation to step through the graphs, and perhaps make it easier to see the changes. I generate the graphs from the *Excel* spreadsheet that I use to collect the data, then cut and paste a standardized graph into *PowerPoint*. I have a *PowerPoint* presentation set up to auto advance, with fade-in transitions as fast as the computer will go. There is about 1 month of data and I will be able to add to the data daily as it comes in. [This *PowerPoint* presentation is available for download from the ARRL *QEX* Web site.[1] — *Ed.*]

There are some strange things going on, of course, that are interesting. I haven't a clue yet what it means, particularly about 1.5 hours after sunrise and the period

[1]Notes appear on page 40.

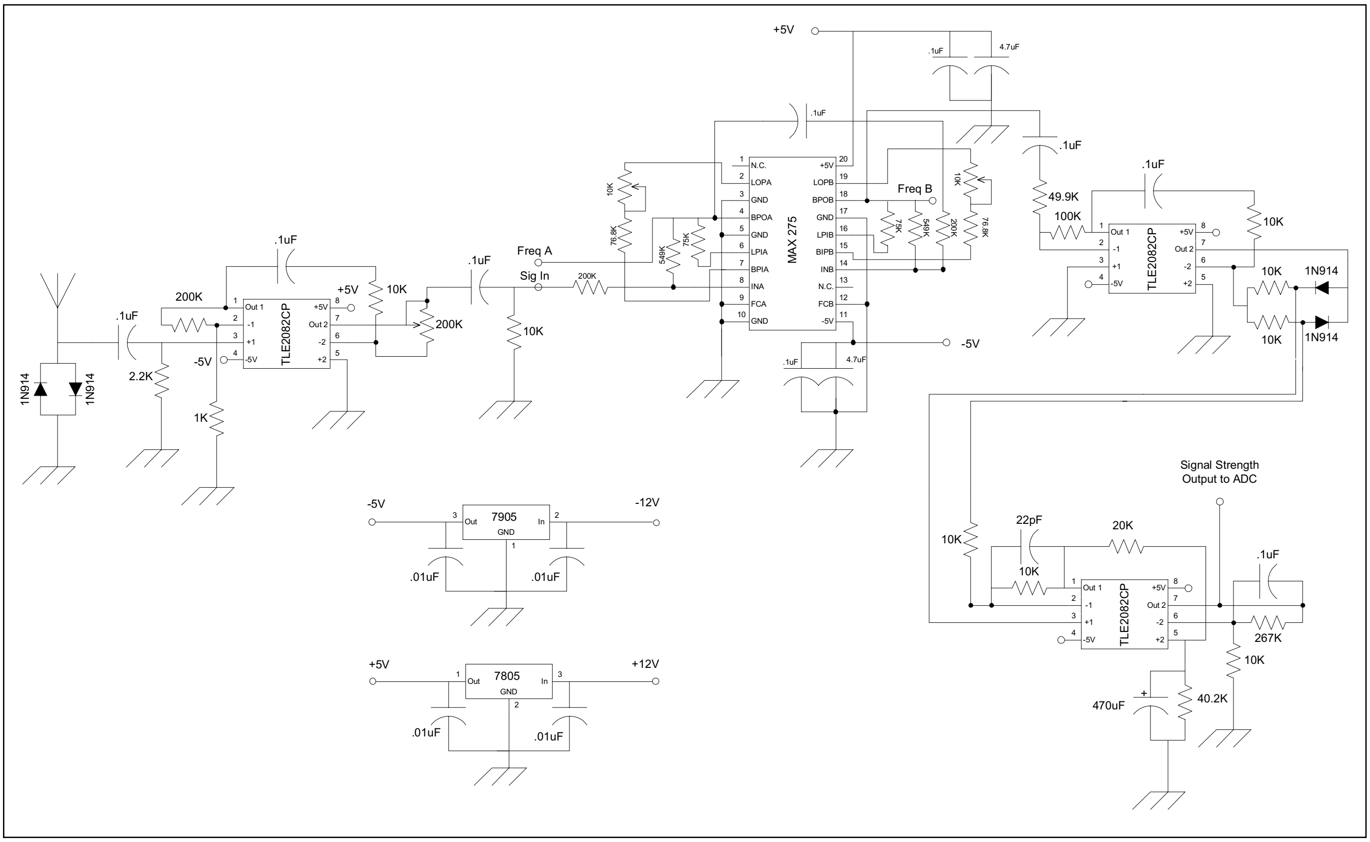


Figure 4 — This schematic diagram shows the SID2 circuit.

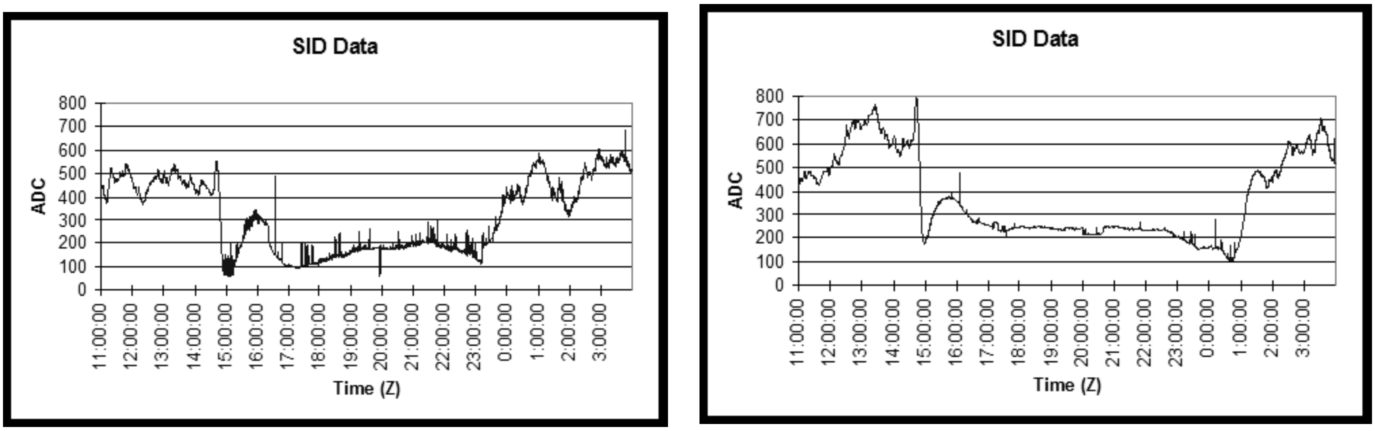


Figure 5 — Part A shows the SID data received using the older Gyrator receiver. Part B shows the SID data received on the following day using the new SID2 receiver.

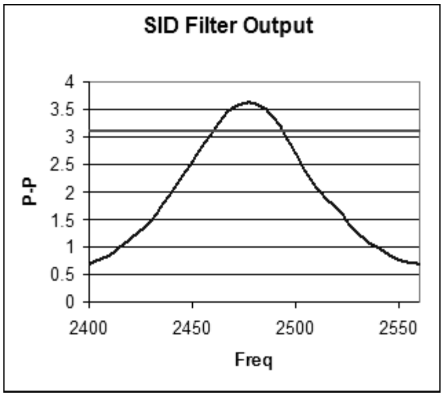


Figure 6 — This graph shows the performance of the MAX275 filter of the SID2 receiver. The horizontal line marks the –3 dB points.

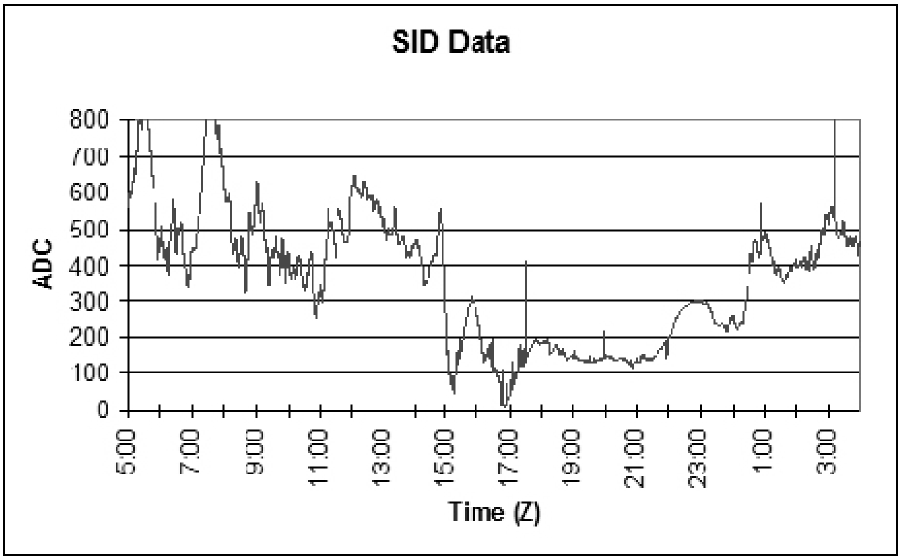


Figure 7 — This graph shows data collected Dec 30, 2008 using the SID2 receiver.

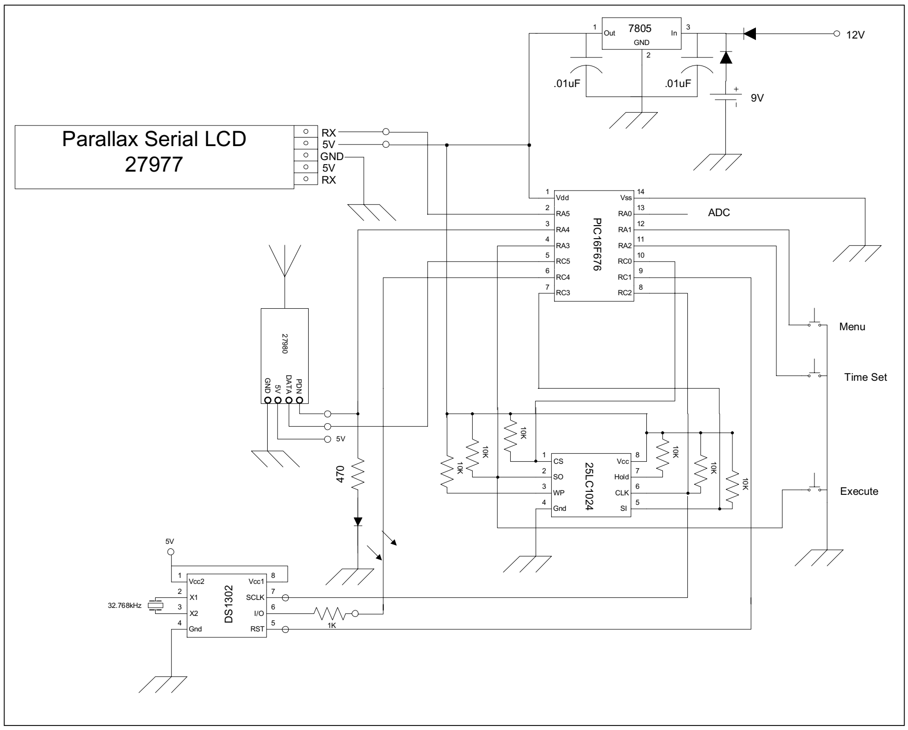


Figure 8 — This is the schematic diagram of the new interface circuit.

around sunset. The steep dip at sunrise is consistent, and behaves as you also would expect. It drifts later in the morning with the later sunrise time. The trough after the sunrise dip is a bit unpredictable. I have anecdotally correlated that variation with observations on the bands. The wide variation at sunset kind of perplexes me. On one hand, sunrise has a very definite and predictable trend and behavior. You would expect that sunset would have the same behavior, but perhaps not as pronounced. It is almost as if ionization occurs like a trip wire reaction; once it starts, it goes to completion. The de-ionization however is not as rapid and complete. I expected that the ionization/de-ionization mechanisms would have similar, if not identical, behaviors. I also am wondering if the fall and rise and fall again around the time (particularly at sunrise) that ionization and de-ionization starts is kind of a "ringing bell" effect, a rapid stimulus causes the system to overshoot the steady state level of ionization and it takes some time for the system to settle down to the steady state. I would be pleased to hear other ideas and explanations.

17 Dec 2008

I did an experiment to compare the results of a few days of Jove data to the SID data. [Jove refers to the NASA Radio Jove solar and planetary radio telescope program for schools. It is a program for monitoring Jupiter and IO. See **http://radiojove.gsfc.nasa.gov/**. — Ed.] The Jove receiver is tuned to 20.05 MHz and I use an NVIS 2-element phased array (pointed up). The system is designed to detect radio signals from Jupiter, but in the daytime it can be used to monitor cosmic background noise and radio energy generated by the sun. There doesn't appear to be a correlation that jumps out at me, but I will do a few more checks to see if there is any connection. See Figure 3. I am trying to explain the sudden changes in the SID data, particularly those sharp drops and "floors" and sharp rises and "ceilings." The only connection that I am seeing is that the rise in the Jove plot around 1700 UTC follows the post-sunrise trough in the SID data shortly before that time.

The Jove data comes from a simple station I built up for that purpose, similar to what schools would use. The radio receives AM at 20.05 MHz. The antenna is a phased two element dipole array about 6 ft off the ground and focused up. The display software is *Radio SkyPipe*. This software is available for free download from **www.radiosky.com/skypipeishere.html**.

I completed the new receiver design on a breadboard and ran it yesterday. The circuit diagram is shown in Figure 4. The receiver is much better behaved than the Gyrator design described in the original *QEX* article. It is more stable and does not break into oscillation. It also appears to handle the spikes from lightening strikes a bit better (which may or may not be good depending on your interest). Figure 5 compares the Gyrator receiver described in the original *QEX* article with the new receiver on consecutive days. Figure 6 shows the filter plot, and marks the –3dB points. That filter circuit is performing well. The next step is to finish the redesign of the interface circuit, because I have not been very happy with the Propeller-based design.

31 Dec 2008

I have the new interface up and running on a proto board. The plot shown in Figure 7 is a 23 hour plot from yesterday. The new interface is based on the PIC16F676. I have external memory to collect the data and a real time clock for time control. The software is set up for the operator to set the time (that is maintained by backup battery power), set the start collection time and the stop collection time. At the start collection time it starts collecting data points at 5 second intervals (dah). At the stop collection time, it automatically dumps the data to the computer via the data link and starts collection again at the set time. I have it set up so that as long as the computer is running and connected to the data link, it will receive the data. It takes about 35 minutes to send the data via the data link. So, I have it set up to collect 23 hours of data.

6 Jan 2009

I have also completed a SID2 redesign. The receiver and interface have been redesigned and made more easily duplicated and easier to tune and operate. Figure 4 shows the SID2 schematic diagram and Figure 8 is the new interface schematic diagram. The receiver and interface all fit on one small board. The board size is small enough that you can order 3 prototyping boards for around $60 from ExpressPCB. The ExpressPCB file that I used to have the board made is available for download from the ARRL *QEX* Web site.[2] The assembly language program file for the PIC is also available for download from that Web site.[3]

The data is actually pretty interesting though I am not sure what it all means yet. The data provides some good stuff for statistical analysis, probably Algebra II level stuff and an opportunity for students to do some real-world research with a simple, inexpensive system.

Notes

[1]The *PowerPoint* presentation showing graphs of the author's data from late October through early December 2008 is available for download from the ARRL *QEX* Web site. Go to **www.arrl.org/qexfiles** and look for the file **3x09_Spencer_Tech_Notes_PPT.zip**.

[2]The circuit board files are available for download from the ARRL *QEX* Web site. Go to **www.arrl.org/qexfiles** and look for the file **3x09_Spencer_Tech_Notes_PCB.zip**.

[3]The assembly language files for the PIC are available for download from the ARRL *QEX* Web site. Go to **www.arrl.org/qexfiles** and look for the file **3x09_Spencer_Tech_Notes_asm.zip**.

QEX

Ray Mack, W5IFS

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

SDR: Simplified

In this issue, we will look at some of the fundamentals of sampling theory and how the theory works in practical situations. First, let's look at the results of our lab from last month.

Lab 1 Results

There are a few changes necessary to the schematic diagram of Figure 5 in the first column, in the Jan/Feb 2009 issue.[1] First, I missed that pin 6 appears twice on the op-amp when I proofread my copy. The inverting input on the top op-amp (U2A) should be pin 2. Second, the circuit operates better if R3 and R4 are 10 kΩ instead of 100 Ω.

Figure 1 shows the oscilloscope output of the circuit as I presented it in the last issue. The top trace of the scope plot is the modulated waveform from the sound card and the bottom trace is the demodulated signal. Once the values are tweaked for optimum performance, you see three levels in the bottom trace. You see distinct levels for the one and the zero, with a blip for the dc level in between. This blip can be used to synchronize the data bits in this particular modulation scheme.

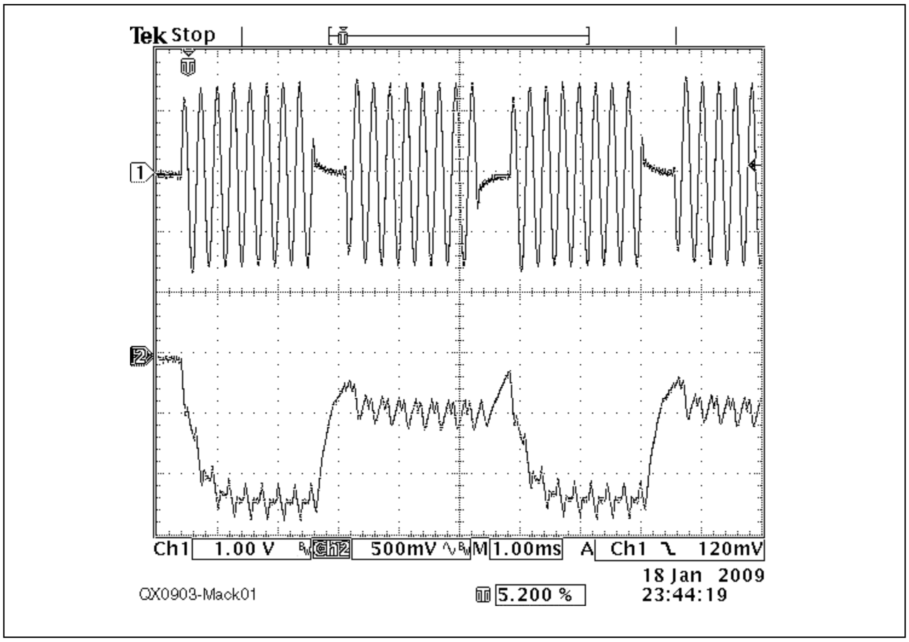


Figure 1 — The top trace of this oscilloscope plot is the modulated waveform from the sound card and the bottom trace is the demodulated signal.

Time Domain and Frequency Domain

There are two ways to look at electrical signals. The first way we learned was to measure voltage or current versus time. Oscilloscopes measure voltage versus time and chart recorders measure current versus time. This is a time domain representation. The second way to look at signals is voltage versus frequency. Panadapters and spectrum analyzers show voltage versus frequency. This is a frequency domain representation. Both time domain and frequency domain tools are used with digital signal processing.

The Nyquist Criterion

One of the most common applications of digital signal processing involves sampling a continuous analog signal using an analog to digital converter, processing the samples using a digital computer, and converting the samples back into a new continuous analog signal. Both conversion processes can lose information if not done correctly. In the analog world, we call loss of information "distortion."

The Nyquist criterion describes what is necessary so that information is not lost in the initial analog to digital conversion pro-

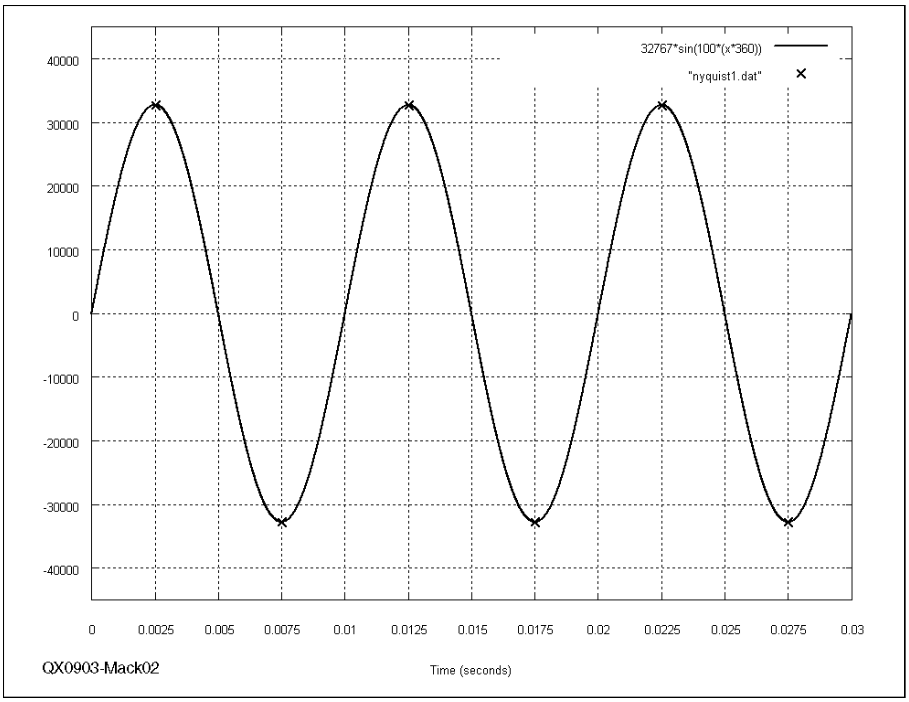


Figure 2 — This sine wave represents a 100 Hz signal. The x marks at the crests and troughs represent one set of sampling points when the sampling rate is 200 samples per second (200 Hz). These samples can define a 100 Hz sine wave with an amplitude equal to the original signal — the ideal situation.

[1]Notes appear on page 44.

cess. Nyquist is probably the best known and most misquoted of the foundations of digital signal processing. Nyquist states that a signal must be sampled at a rate ***greater than*** twice the highest frequency component of the input information. It is totally incorrect to say that it must be greater than ***or equal***, which is how many authors describe Nyquist. In fact, practical systems require the sampling frequency to be much more than twice the highest input frequency. The Nyquist criterion also states that a signal that is sampled at a rate greater than twice the highest input frequency can be completely reconstructed as an analog signal without loss of information. There are some impractical assumptions in this second half of the Nyquist criterion, however.

Let's look at an example of why the "equal to" condition is just plain wrong. We'll take a 100 Hz sine wave and sample it 200 times per second. We can do this manually by simply plugging in values to the formula for a sine function and evaluating for a sequence of times that are 5 milliseconds apart. We will look at three different cases, however, for the position of our first sample relative to where the sine function begins. Our first case samples the sine function at times 2.5 ms, 7.5 ms, 12.5 ms, 17.5 ms, 22.5 ms and 27.5 ms. Figure 2 is a *Gnuplot* of the input sine and the samples. You can see that you capture all of the maximum and minimum values of the sine function, so you should be able to exactly reconstruct the waveform with the proper filtering. This reconstructed sine will have the input frequency of 100 Hz and have the exact same phase. It will also have a peak amplitude equal to the value of the samples.

The second sample sequence is for times 0, 5 ms, 10 ms, 15 ms, 20 ms, 25 ms and 30 ms. You can see from Figure 3 that this sample sequence of the very same sine function yields a sample result indistinguishable from a dc value of zero! This result is called aliasing. Aliasing is the name of the effect in which sampling converts one frequency to a lower frequency. In this case, 100 Hz is converted to zero hertz.

Our third sample sequence samples the sine function at 1.25 ms, 6.25 ms, 11.25 ms, 16.25 ms, 21.25 ms, and 26.25 ms. Figure 4 shows the sample sequence. You can reconstruct this signal and create a 100 Hz sine with proper filtering, but notice that you will get a sine wave that has an amplitude only 0.707 of the original, and the result will be out of phase with the original by 45°.

So, from now on we will be certain to sample our input signals at some frequency greater than two times the highest input signal. Let's look at a 10 kHz sine wave, and see what happens if we sample it at a rate of 22,050 samples per second. Figure 5 is an oscilloscope plot of the 10 kHz signal sampled at 22.05 kHz and played by our sound card. The sample rate is 10% above

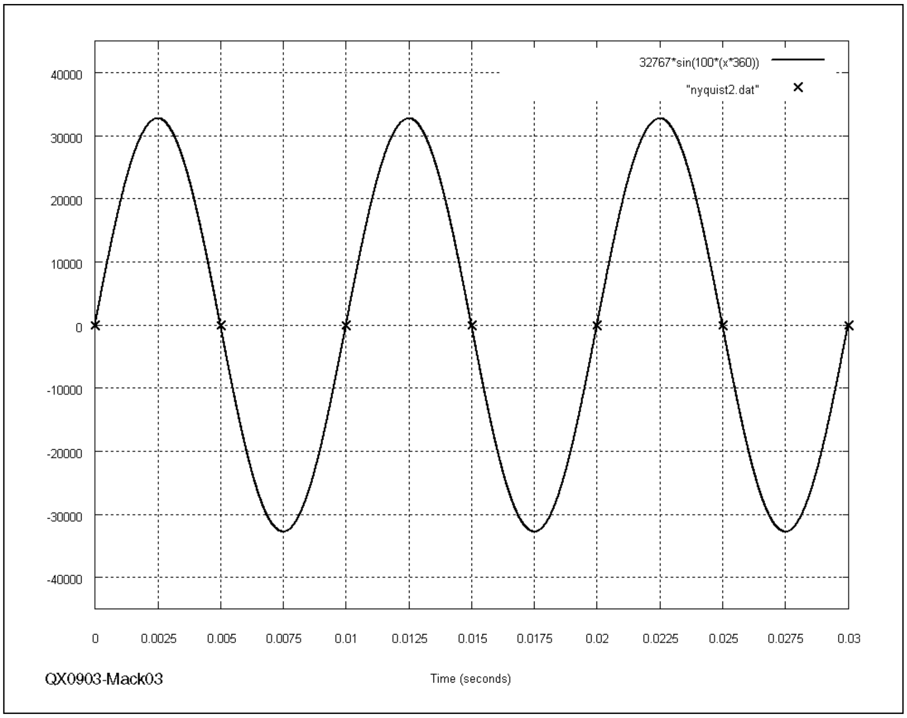


Figure 3 — In this oscilloscope plot, the same 100 Hz signal from Figure 2 is again sampled at a rate of 200 Hz, but this time the samples are taken to match the zero crossing points. This time the samples appear to define a dc signal with zero amplitude.

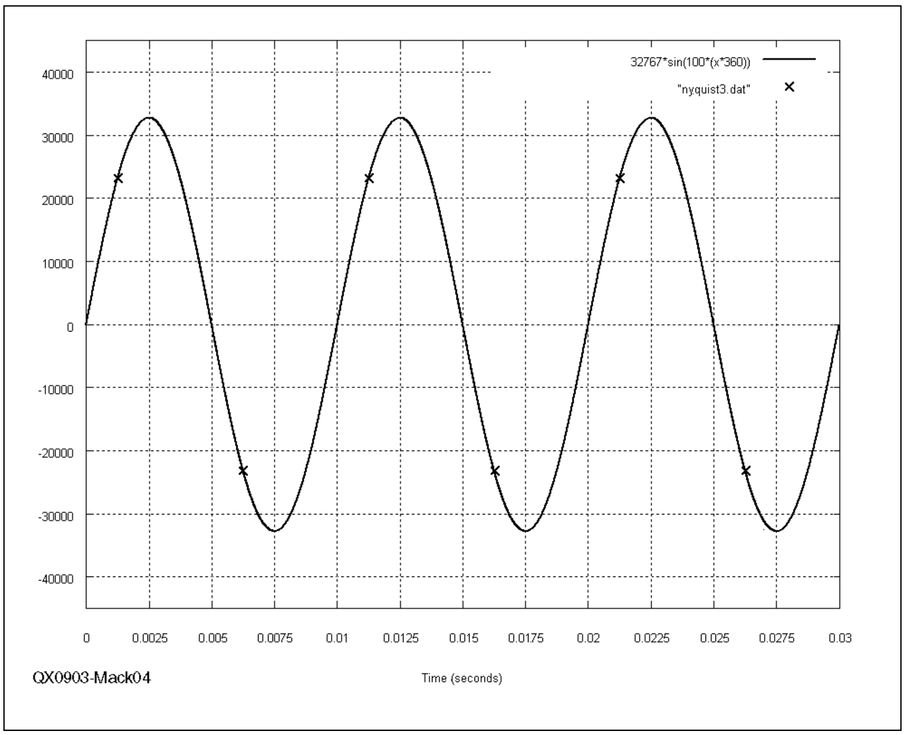


Figure 4 — Here the 100 Hz sine wave is sampled at a rate of 200 Hz, with the samples selected to fall 45° before the crests and troughs. With proper filtering you can create a 100 Hz sine wave from these samples, but the resulting signal will have a maximum amplitude of only 0.707 times the maximum amplitude of the original sine wave. In addition, the new signal will be shifted 45° in phase.

the required amount. You will notice that the signal created from the samples is not a clean sine wave. The difference from a true sine wave illustrates a failure to follow the second requirement of Nyquist: sampling at a rate a little above two times the highest input frequency requires a brick wall filter.

A brick wall filter is an ideal (and physically impossible) filter that has a passband response that is an exactly rectangular shape (Figure 6). The filter on my Dell laptop is nowhere near a brick wall filter. The next two waveforms (Figure 7) are the output of my HP computer for a 10 kHz signal sampled at 22.05 kHz and 20 kHz sampled at 44.1 kHz. Notice that once the initial ramp occurs, then the output signal is a very good sine wave at 10 kHz and 20 kHz respectively. Even though the filter is almost a brick wall, the difference is enough that the first eleven cycles are not a true reconstruction of the input data.

All of the issues we have looked at with respect to Nyquist presume that you are interested in digitizing a signal that spans from dc all the way up to some sample rate less than the Nyquist limit. As we move to looking at RF signals and how to modulate them or demodulate them, we will take advantage of properties of bandpass and other filters, mathematics of sampling, and sample rates that are chosen to keep far enough away from the desired signals so reasonable filters will do the job.

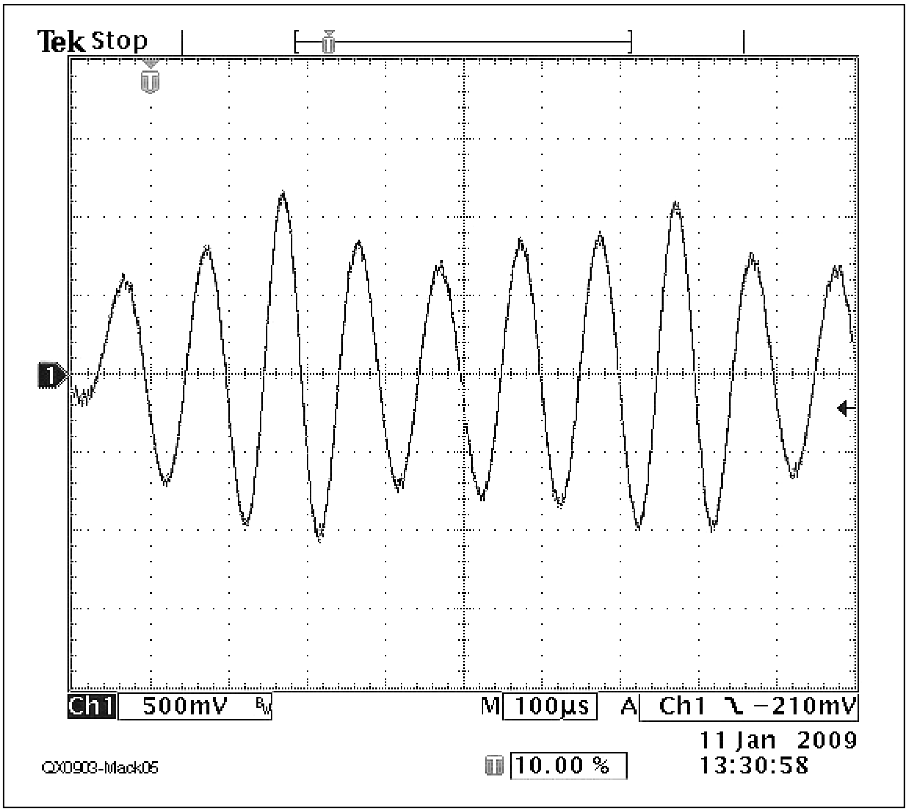


Figure 5 — This oscilloscope plot shows a 10 kHz sine wave that was sampled at 22.05 kHz and played through the sound card on my Dell laptop computer. The reconstructed signal isn't a very good sine wave. The filter on this sound card is not close to a "brick wall" filter.

Fourier and Negative Frequencies

The Fourier transform and negative frequencies are two concepts where the "real world" doesn't seem to square with the mathematics we use. The Fourier transform converts a continuous signal in the time domain to a different continuous function in the frequency domain. Fourier actually discovered two sets of functions. The one we will use for filtering and spectrum display is the Fourier transform. It applies to continuous and non-repetitive waveforms. The other is the Fourier Series, which describes how to create a periodic waveform by adding up a sequence of cosine and sine waveforms that are harmonically related.

We need the concept of I and Q again when dealing with the Fourier transform. Mathematicians call a signal that contains both I and Q a complex signal. That is because they play games with the imaginary value ***i*** (square root of –1) in order to use complex mathematics to generate values that have both magnitude and phase. Electronics folks use ***j*** instead if ***i***, but it is the same thing. We saw in the last column that complex math is really as simple as creating two electronic signals that are a sine wave and a cosine wave, so there is no real magic involved. A Fourier transform takes a complex input in the time domain (I and Q waveforms) and transforms them into a new complex waveform in the frequency domain. This new complex waveform is also just an in phase and quadrature set of data. The combination of the transformed I and Q signals can be used to generate a pair of plots that either show magnitude and phase or we can work directly with just the I and Q components.

The math functions for cosine and sine start at a time of negative infinity and go to positive infinity. The functions that do a Fourier transform also operate from negative infinity to positive infinity. The choice of a real world "zero" time is arbitrary for both the trig functions and the Fourier functions. When we measure a signal with an oscilloscope, we usually place our "zero" time as the left edge of the display and measure time as a positive value relative to that left edge. It is equally reasonable, however, to place "zero" at the middle of the screen and measure both positive and negative time for the data on the screen. If our oscilloscope is set for 1 second per division, our view is limited to –5 seconds up to +5 seconds. All of our digital signal processing will work with signals with positive and negative times and limited amounts of data. We use positive and negative times because all of the functions require the numbers to be symmetrical about "zero." When we do a Fourier transform on signals that cover time before and after zero, we generate a frequency domain representation that contains both positive and negative frequencies.

Now is a good time to look at two more important trig identities:

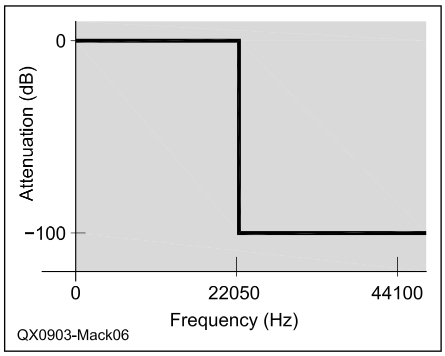


Figure 6 — This graph represents the transition between the pass band and stop band for an ideal "brick wall" Nyquist filter for a 44.10 kHz sample rate.

Sin(–x) = –sin (x) = sin (x +180°) and cos (–x) = cos (x)

What these two identities mean for us is that a sine wave of –10 Hz (based on our reference cosine wave from last issue) is identical to a sine wave of 10 Hz with a 180° phase shift. The cosine wave has identical phase for both negative and positive frequencies.

The analog image reject mixer uses the property of negative frequencies and a couple of phase shifters to add one set of frequencies and cancel the other set. The Weaver method of single sideband

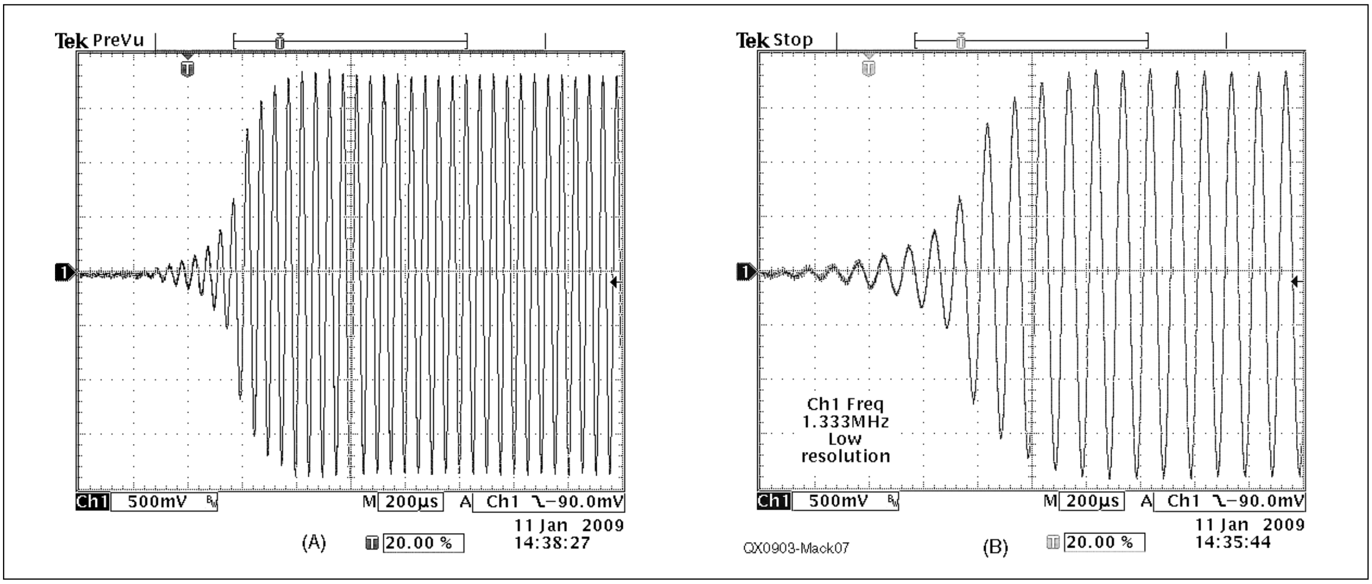


Figure 7 — The oscilloscope plot at A is the output signal from a 10 kHz signal sampled at 22.05 kHz, as processed on my HP computer, with a sound card that provides a much better approximation of a brick wall filter. The plot at B is the output signal from a 20 kHz signal sampled at 44.1 kHz, and played through the sound card in my HP computer.

generation also uses the phase differences between positive and negative frequencies to perform its functions.[2]

You are probably familiar with the Fourier series for a square wave, which consists of the fundamental frequency and all of the odd harmonics. The concept of "zero" changes the Fourier series, however. Here is what I learned as the classic Fourier series for a 50% duty cycle square wave:

$$F(t) = \sin(x) + 1/3 \sin(3x) + 1/5 \sin(5x) + 1/7 \sin(7x) + 1/9 \sin(9x) + \ldots$$

Figure 8 shows a plot of the resulting waveform. Let's look at a similar Fourier series that I first saw in a recent *QEX* article that has all of the same frequencies with the same amplitudes but with different phases:

$$F(t) = \cos(x) - 1/3 \cos(3x) + 1/5 \cos(5x) - 1/7 \cos(7x) + 1/9 \cos(9x) \ldots$$

Figure 9 shows a plot of the resulting waveform for this series. We see that it is the same 50% duty cycle square wave, but with zero in the middle of the high phase. These waveforms are a good example of how changes of where we place zero in the time domain can affect how the frequency domain is presented.

Next Time

In the next column we will start working in earnest with a real application of the Nyquist criterion and Fourier transforms. We will look at how Fourier and Nyquist can be used for an under sampling receiver to receive the time standard WWVB at 60 kHz, using a sound card sampling at 48 kHz. We will also go through the steps necessary to set up your computer to use the Blackfin Stamp product.

Since I started working on this column, DigiKey has decided not to stock the AD7476-DBRD board. Analog Devices still makes the board and has some in stock, so you can order a board directly from their Web site. You will need to set up an account. Start from the main page at **www.analog.com/** and select "Buy Online" in the upper right corner. Select "Place Credit Card Order Now" and select "No I am a New User." You will need to fill out the information to register with the site and set your password. Then you can select "Buy Online" again, and place an order for the AD7476-DBRD.

Notes

[1]Ray Mack, W5IFS, "SDR: Simplified," Jan/Feb 2009 *QEX*, pp 53 – 56.

[2]For information about the Weaver method of SSB signal generation, see the Wikipedia entry at **http://en.wikipedia.org/wiki/Single-sideband_modulation**.

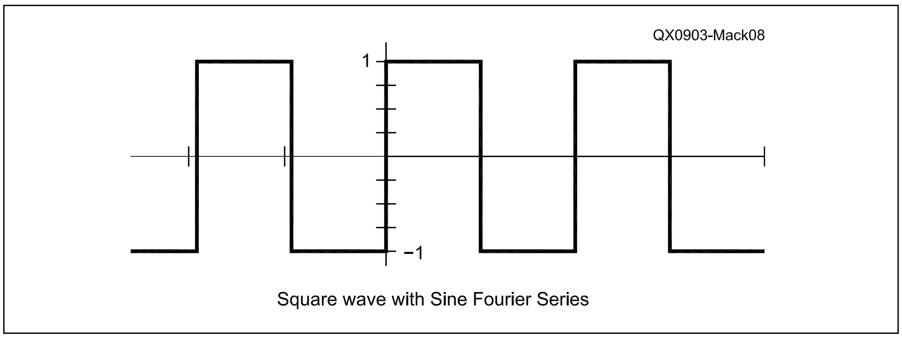


Figure 8 — This is the square wave signal that results from a sine Fourier series. Note the 180° phase shift around the zero frequency axis.

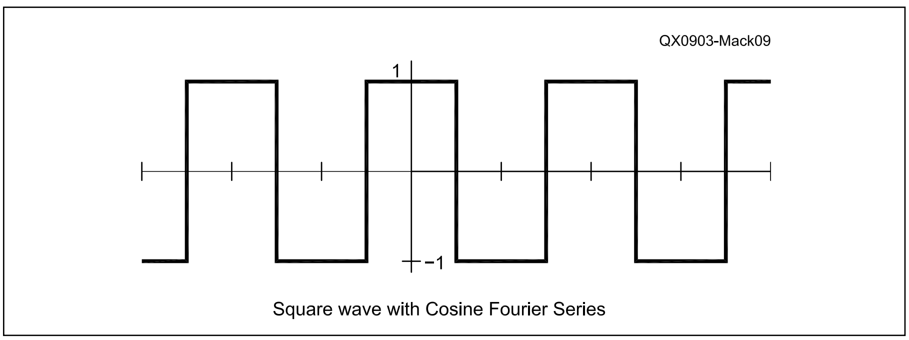


Figure 9 — This is the square wave signal that results from a cosine Fourier series. Note the 0° phase shift around the zero frequency axis.

QEX

Letters to the Editor

VHF Frequency Multiplication Using the SA602 IC (Jul/Aug 2008)

Dear *QEX* Editor,

This is a response to a letter in the Jan/Feb issue of *QEX*, page 58, about "VHF Frequency Multiplication using the SA602 IC." Linearity is not necessary to generate the second harmonic in a mixer, but a phase shift or time delay between the two inputs to the mixer is necessary.

Start with a square wave, and make a copy delayed by 90°. Multiply these two square waves together, and the product is a square wave at twice the frequency. Varying the delay, varies the duty cycle of the output wave.

This is a convenient way to double the frequency of a digital clock using a delay line (or string of inverters to provide the delay) and an exclusive-or gate. The output has a rising edge for every rising or falling edge of the input, with the width of each output pulse approximately equal to the delay:

```
Input ---+-----(delay)---
         |              XOR-------output
  -------- + ---------------
```

— 73, Peter Traneus Anderson, KC1HR, 42 River St, Andover, MA 01810-5908;
traneus@verizon.net

Broadband Impedance Matching (Nov/Dec 2008)

Dear Frederick and Larry,

In reference to the article "Broadband Impedance Matching" in the Nov/Dec 2008 issue of *QEX* I would like to bring to the attention of interested readers some work I have done to augment the work of Mr. Frederick Hubler. I have found a way to obtain the g(n) values using a method and a computer program described in a book by Thomas R. Cuthbert, Jr, *Circuit Design Using Personal Computers*, John Wiley & Sons, 1983. The program in this text, Program B6-3, has been rewritten in C++ and modified to suit the nomenclature in Mr. Frederick Hubler's article. This computer program, as well as supporting files are available for download from the ARRL *QEX* Web site.[1]

The program runs under *Windows XP* and *Windows Vista*. The program is compiled as a Win32 Console Application, and so it may run under earlier *Windows* operating systems. Source code is provided so readers are able to modify and compile as they wish. Program output is displayed at the console and written to both text and csv files for use with spreadsheet programs such as Microsoft *Excel*.

Three types of inputs may be provided to the program:

1) Load Decrement

2) Load Parameters

3) Load Q

In the first type of input, the Load Decrement, which is defined in Mr. Hubler's article is input into the program. In the second type, load parameters are input into the program and some simple calculations are performed to obtain the Load Decrement. Load parameters are also defined in Mr. Hubler's article. In the third type, load Q is input into the program as was done in the original "Program B6-3" found in the text by Thomas R. Cuthbert, Jr, and this is scaled by the fractional bandwidth within the program to obtain the Load Decrement.

The significant data inputs of the program are:

1) Result file name

2) Order N

3) Low and high frequency

4) Type of input and the corresponding data

5) Select serial or parallel load realization.

The significant data outputs of the program are:

1) Center Frequency

2) Match Bandwidth

3) Fractional Bandwidth

4) Return Loss Min and Max

5) SWR Min and Max

6) g(n) prototype values

7) Scaled Load and Source Resistance

8) Scaled L(n) and C(n) values.

When you download the file **3x09_hubler-schultz.zip** you will find the following:

Readme.pdf
This is a description of files contained in this zip file.

Cuthbert_B6_3n._xe

[1]Steven Schultz's C++ program file and related data files to accompany Frederick Hubler's article "Broadband Impedance Matching: from the Nov/Dec 2008 issue of QEX are available for download from the ARRL QEX Web site. Go to **www.arrl.org/qexfiles** and look for the file **3x09_hubler-schultz zip**.

The program executable (which needs to be renamed to .exe).

Cuthbert_B6_3n.cpp
The program source code.

Interpolated 4.xls
Impedance data for antenna.

Series Load
Folder that contains files related to modeling series loads

Parallel Load
Folder that contains files related to modeling parallel loads

I have extensively tested program and reviewed the accompanying documentation. I will be interested in hearing from readers who try this program, and have any comments or suggestions.

— Sincerely, Steven Schultz, WB8WGY, 5 Kinsman St, Lowell, MA 01852; **steven.m.schultz@ieee.org**

Steven,

Thank you for the e-mail and the program. I tried it out, and it works fine for the n = 4 networks in the article.

I had a request from Ron Skelton, W6WO, inquiring about another way to obtain the g-parameters since the book I referenced is pretty expensive. Although I don't have a copy of Tom Cuthbert's book, I knew him from working at Rockwell Collins, and I had forgotten that he had written that book.

— Sincerely, Frederick Hubler, 9422 Deer Ridge Dr, Cedar Rapids, IA 52411; **fhubler@msn.com**

An Automatic Noise-Figure Meter (May/June 2007)

Larry,

I've just discovered that there was an error in the C code file for my article "An Automatic Noise-Figure Meter," which appeared in the May/June 2007 issue of *QEX*. I received an e-mail from Hannes, SM6PGP, who found that line 546 in the panfi.c file should really have read:

ENR = ((T_hot_RAM - 290.0) / 290);

I have attached the corrected .zip file to replace the one that is presently archived on the ARRL *QEX* Web site; **5x07_koehler.zip**. I would appreciate it if you could publish the correction in the next convenient issue of *QEX* giving due credit to Hannes, SM6PGP. Thanks.

— 73, Jim Koehler, VE5FP, 2258 June Rd, Courtenay, BC V9J-1X9, Canada; **jark@shaw.ca**

Jim,

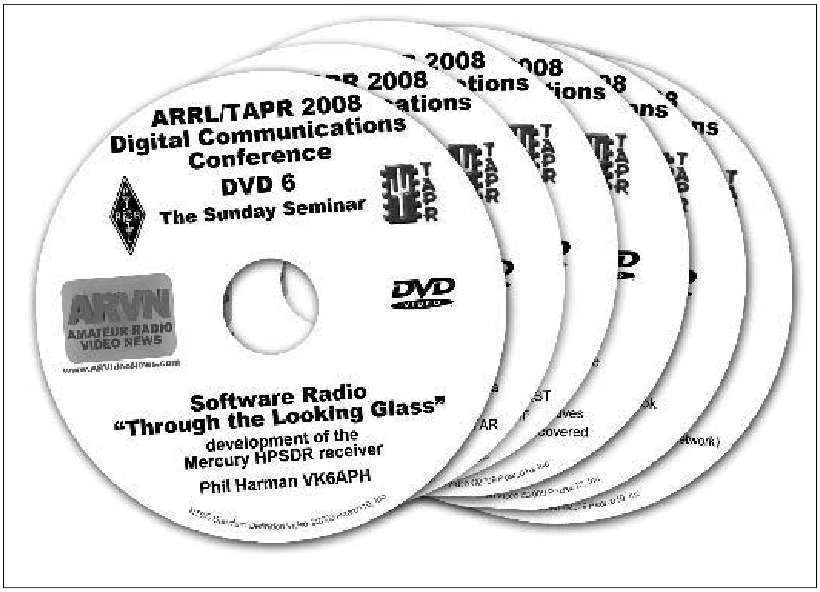
Thanks for letting us know about the error in that line of code, and for sending the updated file. I will replace the original **5x07_koehler.zip** file on our server with this new file. I will also include a link to the file under a heading for "Letters" in the Mar/Apr 2009 section of the files list.

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

Ray and Larry,

Thank you for the new SDR column! The first installment contained the first explanation of I and Q that I felt like I could even begin to understand.

— 73, Robert Morris, AB1HL, 28 Arlington St, Cambridge, MA 02140; **rtm@csail.mit.edu**



ARRL/TAPR 2008 Digital Communications Conference DVD Set

Hi Larry,

The set of six DVDs from the 2008 ARRL/TAPR DCC is now available. DCC attendees and those who missed the conference will be interested in the DVDs. The full weekend of presentations are included on these DVDs. There is more information about each DVD available at the Amateur Radio Video News Web site: **www.ARVideoNews.com**. They can be purchased for $15 each (plus shipping), or the complete set for $75 (plus shipping).

Every minute of each presentation is included, and there are four presentations on each DVD. They're not in sequential order as given at the DCC. Rather, I grouped them together by subject as best I could. DVD 1 is all about packet radio; the talks on DVD 2 are about Software Defined Radio. DVD 3 includes presentations about Winlink, APRS and D-STAR. Your talk about writing for ARRL publications, Joe (N6CL) Lynch's talk about writing for CQ-VHF, as well as SuitSat and AMSAT Eagle talks are included on DVD 4. All the introductory talks are together on DVD 5, and the **Sunday Seminar** — a half-day talk on the Mercury high-performance software defined radio project by designer Phil Harman, VK6APH, fills DVD 6. Most of the talks run about 45 minutes, so they'd be good for club meeting programs.

TECHNICAL NOTE: ARVN DVDs are produced in standard definition, NTSC format. They have no geographic restriction, but International playback requires multi-standard equipment. They should also play okay on computers around the world.

— 73, Gary Pearce, KN4AQ, 508 Spencer Crest Ct, Cary, NC 27513; **kn4aq@arvideonews.com**

Hi Gary,

Thanks for letting us know that the DVDs are now available. This was a big project and we appreciate your efforts to fully document the DCC.

— 73, Larry, WR1B QEX

Next Issue in *QEX*

Bob Melvin, W6VSV, shows off his "Backyard Antenna Test Range." UHF and microwave experimenters around Hillsborough, CA have been meeting in Bob's backyard for years. There, he conducts computerized UHF and microwave antenna tests on his test range. Bob built the test gear and wrote the computer software that performs the tests, and plots 8.5 by 11 inch charts that display the measured results. Bob has tested many small antennas on all of the ham bands from 432 MHz up through 10 GHz! Now he is sharing his handiwork with *QEX* readers.

Larry Wolfgang, WR1B

QEX Editor; qex@arrl.org

Reader's Page

Hello Larry,

After reading your editorial in the Jan/Feb 2009 issue, I thought I might get the chance to show off my implementation of Bertrand (VE2ZAZ) Zauhar's GPS-derived frequency standard, published in the Sep/Oct 2006 issue of *QEX*. Bertrand has indeed done an excellent design job with much attention to detail, and the results have been very satisfying. My version has an average frequency offset that lies around 0.5×10^{-9}, which is more than adequate for my needs.

I used the Motorola OnCore GT+ GPS module. A defunct uninterruptible power supply donated the project's enclosure — and it even already had all the cutouts I needed! I designed and built the multi-voltage power supply and outer oven controller thermostat circuits for the HP10811 OCXO I used. The project worked right off the bat, and — low and behold — locked within minutes (which left me almost wishing for some good old troubleshooting!). Now I am confident that my HP5335A frequency counter indications are spot-on!

Thanks, Bertrand! Thanks, QEX! Keep up the good work, guys!
Tasos Thomaidis, SV8YM, Zakynthos, Greece

Dear Larry,

As you requested in the Jan/Feb issue of *QEX*, here is my homebrew masterpiece. It is a GPS disciplinizer for a quartz oscillator, as described by Brooks Shera, W5OJM, in his article, "A GPS Based Frequency Standard," in the July 1998 issue of *QST*. My version has been running for over 10 years, and I don't see any reason to change it. It is still my best homebrew project. It not only provides a 10 MHz signal synchronized with GPS, but also keeps computer records of the GPS phase difference comparison indications.

On the front panel you can see the meter and LED indication of PLL control voltage,

Photo 2 — This photo shows the front panel of Taso's GPS-derived frequency standard. He used two LEDs for the FLL status indication, instead of a single bicolor LED.

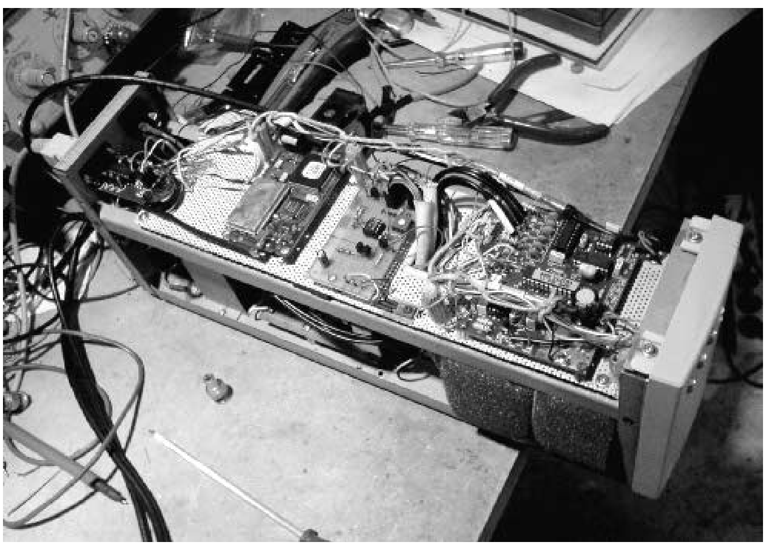


Photo 1 — The nearly completed project by SV8YM. Bertrand's circuit board is on the front (right) side, with the HP10811 OCXO under it, wrapped in insulating foam.

Photo 3 — Here is the back panel of Taso's project. Using 50 Ω terminations at the unused outputs proved beneficial.

Photo 4 — Andrea Daretti, IZ2OUK, built the GPS-Based Frequency Standard from the July 1998 issue of *QST*. The top box is the "disciplinizer," and the quartz crystal oscillator is under that.

as well as the switch for the PLL filter time constant. The "disciplinized" quartz oscillator is underneath.

It would be very nice to see my project on the *QEX* "Readers' Page."

Andrea Daretti, IZ2OUK, Milan, Italy

Upcoming Conferences

35th Eastern VHF/UHF Conference

April 17, 18, 19, 2009
Enfield, CT

The 35th Eastern VHF/UHF Conference will be held April 17, 18, 19 2009 at the Crowne Plaza Hotel, 1 Bright Meadow Blvd, Enfield, CT 06082. Phone (860)-741-2211.

The Friday evening Hospitality Suite opens at 6 PM and extends throughout the evening. Registration opens Saturday at 7:30 AM, and the presentations from 8:30 AM to 4:30 PM, with a lunch break at noon. The dinner banquet Saturday evening is from 7 to 10 PM, including a trivia quiz, banquet prizes and more. Sunday there is an outdoor flea market (weather permitting) from 8 AM to noon.

The conference is handicapped accessible, with onsite dining and parking. See the North East Weak Signal Group Web site for more information and online PayPal registration. (**www.newsvhf.com/vhfconf.html**) For additional information, contact Conference Chairman Bruce Wood, N2LIV, 3 Maple Glen Ln Nesconset, N.Y. 11767-1711; **n2liv@optonline.net**

13th Annual Southeastern VHF Society Conference

April 24, 25 2009
Charlotte, NC

The 13th Annual Southeastern VHF Society Conference will be held April 24 and 25, 2009 at the Doubletree Hotel Charlotte Airport, 2600 Yorkmont Rd, Charlotte NC 28208. Phone 704-357-9100 or their national toll free reservation line, 800-222-TREE.

Informal activities start Thursday evening with a get-together in the hospitality rooms. The conference is on Friday and Saturday. There are technical and operating presentations both days. The Annual Banquet is Saturday night, with a keynote speaker. For more information and to register on-line go to the Society Web site: **www.svhfs.org/**. The Conference Chair is Bill Fisher, W4GRE; **wvfisher@gmail.com**

TM-G707A

TM-V7A

TM-D700A

With the supplied accessories the RC-D710 is a full upgrade to the TM-V71A. The TM-V71A will have full functionality of the TM-D710A by exchanging the TM-V71A panel with the RC-D710.

This is where it gets interesting!

PG-5J connection kit makes the RC-D710 a complete standalone APRS/TNC for your current radio. This option allows connectivity with previous and current Kenwood models* as an external modem.

**Compatible models include: TM-D710A / TM-V71A / TM-D700A / TM-G707A / TM-V7A / TM-733A / TM-255A / TM-455A*
SmartBeaconing™ from HamHUD Nichetronix

KENWOOD U.S.A. CORPORATION
Communications Sector Headquarters
3975 Johns Creek Court, Suite 300, Suwanee, GA 30024-1298
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#42908

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