N2ADR describes his approach to designing a software-defined HF transceiver using Field Programmable Gate Arrays.

QEX

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

January/February 2011

About the Cover

This month's *QEX* cover showcases the
HF digital transceiver designed by James
Ahlstrom, N2ADR. See his article "An All-Digital
Transceiver for HF." The actual circuit board and
enclosure appear in the cover foreground. A
screenshot of the author's Quisk control software
is displayed in the background.

In This Issue

Features

Index of Advertisers

Raymond Mack, W5IFS

Empirical Outlook

Larry is still recovering from an illness. He is fine, but unable to write for this issue, so it falls to me again.

HSMM Losses

The HSMM activity lost a pioneer on November 12, 2010. John Champa, K8OCL, lost his battle with cancer. Many people knew his work through the ARRL HSMM working group and he was working on new topics until just recently. You can read more at **www.arrl.org/news/john-champa-k8ocl-sk**. John was a visible part of the HSMM effort through his committee work and articles in *CQ VHF Magazine* and *QEX*.

Only a fortnight later, we lost a tireless contributor to the work being done in Austin on the HSMM Mesh project – Lloyd Crawford, N5GDB. Lloyd was a driving force behind the scenes in our efforts to deploy wireless routers throughout the Austin area as well as other metropolitan areas of Texas. Lloyd was responsible for the software installation and hardware modifications for a significant number of Mesh nodes throughout Texas. Lloyd's passing is also a loss for the VHF and microwave community in Texas. Lloyd was a founding member of the Roadrunner Microwave Group and was an officer in some capacity for its entire history. I will miss our frequent lunches during the work week. Lloyd is survived by his wife, Ibbie, two brothers, a son, two daughters, and many grandchildren.

Writing for *QEX*

We haven't mentioned writing for *QEX* in this space in a while. We are always interested in articles about what you are working on that would be of interest to others. This is the Experimenters Exchange, after all! It is not necessary that English is your first language or that you are an expert at writing even if English is your first language. You will notice European and Asian call signs on recent articles. While a well-written submission has a better chance for acceptance, your work doesn't have to be letter perfect. Get down your experiences as best you can, and we'll work with you. The main qualification we require of authors is that they know their topic. We need material at all skill levels. We actively support a wide variety of topics, from simple construction projects to the advancement of theory. The rate is $50.00 per printed page plus you get to see your name in print. You can find the author guide at **arrl.org/qex-author-guide.** It gives a more detailed description of the process of writing and submitting an article. I look forward to seeing your article in *QEX*.

Looking Back/Looking Forward

I am writing this at the end of the year 2010 and it seems a good time to look back at the year behind and ponder the year ahead. This was the 100 year anniversary of the Boy Scouts of America. An every four year event called National Jamboree was moved to coincide with this year and was held at Camp AP Hill, VA. The ARRL had a significant presence (including our editor, Larry Wolfgang) to help Scouts learn about amateur radio and earn their license. The Scouts also brought back four merit badges from 100 years ago that a Scout could only earn during 2010. One of those was signaling merit badge. This required a Scout to learn the Morse code as used for radio, light and wigwag flags. Hopefully, this exposure will encourage more young people to join the amateur ranks.

July 1 found the end to arguably the worst BPL installation with respect to interference to amateur bands. The end of the Manassas, VA system came due to a flawed financial model rather than due to compliance with interference regulations, but at least it is gone. Hopefully, it stands as an example of the significant flaws in both the financial and technical aspects of BPL. On another front, ReconRobotics seems to be stalled in deploying systems that are likely to interfere with operations on 70 cm. This delay is due in large part to ARRL spectrum defense efforts to convince the FCC to follow proper procedures.

Looking forward, the 900 MHz band appears to be moving toward use only as a strong local signal band. Kent Brittain, WA5VJB, of the North Texas Microwave Society reports measurements of RFID devices near retailers like WalMart along with other ISM uses of the band have raised the noise floor to the point that weak signal amateur work will be pushed to the band edges near 902.00 MHz to 902.05 MHz. In Texas, those uses also include RFID tags used by the State's toll road automatic billing systems.

My crystal ball is a little fuzzy on what is likely to appear in electronic devices for us to use in experiments. It seems reasonable to expect that data converters for SDR use will continue to drop in price and increase in frequency. Today, you can buy data converters below $200 each that operate above the 70 cm band. That is still a pretty steep price. Perhaps this could be the year for a VHF and UHF transceiver with only level shifting amplifiers between the antenna and data converters. The power amplifier companies are hard at work on new materials. Gallium Nitride (GaN) has reached the mainstream in commercial use. Now, they are working on Indium Arsenide materials that could produce even more powerful high frequency power transistors.

James Ahlstrom, N2ADR

221 Pleasant Plains Rd, Stirling, NJ 07980; jahlstr@gmail.com

An All-Digital Transceiver for HF

Build your own HF transceiver using an FPGA and software.

In a previous article, I described a digital SSB exciter that used a personal computer (PC), a field programmable gate array (FPGA), and software.[1] That design used an Ethernet controller connected to a microcontroller (MCU) and then connected to the FPGA. The microcontroller ran the Ethernet and IP code, leaving the FPGA free to do the digital signal processing. The microcontroller was a bottleneck that slowed down the data speed to barely acceptable levels. I wanted to work at higher sample rates and clock speeds, so I started this new project.

At first, I tried to connect the Ethernet controller chip to both the MCU and the FPGA, but this proved awkward. In the end, I decided to eliminate the MCU and connect the Ethernet controller directly to the FPGA. This resulted in a high data rate but meant that I had to write new Ethernet and IP code for the FPGA and throw my MCU code away. This was worrisome, as I had never written a large FPGA program before. I purchased some new Verilog books that proved to be highly useful, thought about it for a while, and then dove in.[2,3]

With the MCU gone, I could change to a faster 10/100 megabit per second Ethernet controller. The FPGA would now be at the center of the design with the digital to analog converter (DAC) transmitter chip connected to it. Then I realized I could just add an analog to digital converter (ADC) for receive, and I would have a complete transceiver.

The Basics

For background, you might want to read my original article. For digital signal processing basics, be sure to see the book by Lyons[4] and the ARRL digital radio Web site[5]. But if that is not handy, here is a crash course in digital radio.

An ADC is an analog to digital converter; it converts your antenna voltage to a number. I am using a 14-bit ADC running at

122.88 MHz, and the numbers it measures range from –8192 to +8191. These numbers will be crunched by software to ultimately result in received audio. To transmit, you need a DAC: a digital to analog converter. You take microphone audio, convert it to numbers with a (different) ADC, crunch those numbers and send them to the DAC for direct conversion to RF. Then, amplify and filter the RF and send it to your antenna. The numbers we are talking about are actually pairs of numbers, an in-phase and a quadrature number, or an I/Q pair. These are most easily thought of as a complex number in the form $x + jy$. Or you could think of them as representing both the amplitude and phase in one number.

The software mostly consists of filters and mixers just like in an analog radio. A digital mixer is the multiplication by $\cos(2\pi ft) + j \sin (2\pi ft)$. You need to gen-

erate this I/Q number somehow, and the CORDIC algorithm is a good choice.[6] To tune numbers with a digital mixer you feed the I/Q pairs of numbers into a CORDIC module, and get new I/Q pairs out the other end. The new numbers represent the original signals shifted in frequency by a constant frequency; that is, they have been "mixed". The advantage that I/Q number pairs have over plain numbers is best seen in a digital mixer. Analog mixers always have images; they produce sum and difference frequencies. These digital mixers have no images. They produce either a sum or a difference frequency but not both.

The digital filters we are most interested in are low-pass FIR and CIC filters. The FIR (finite impulse response) filters can have very nice pass-band and stop-band responses, but they require hardware multipliers. The FPGA does have hardware
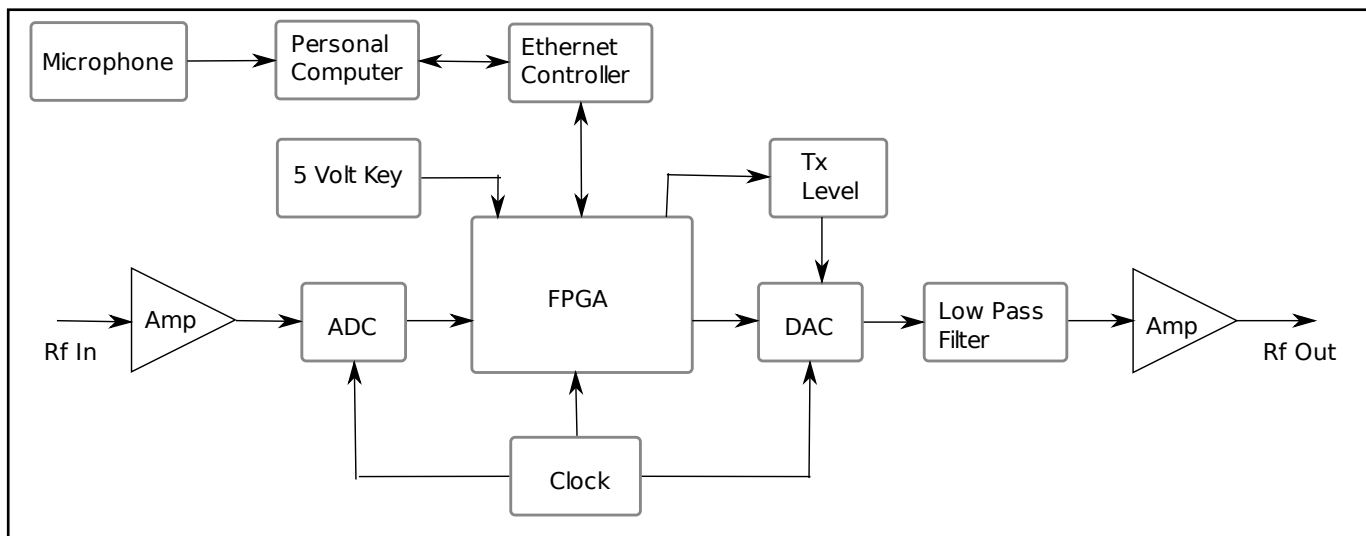
Figure 1 – A block diagram of the transceiver.

multipliers but not enough to make a very big filter. So we use some CIC filters, also. These CIC (cascaded integrator-comb) low-pass filters are only effective near zero hertz. As we move away from zero hertz to perhaps 1% or 10% of their bandwidth, the remainder of the stop-band has poor attenuation. But their advantage is that they require no multiplication at all, are very fast, and are easy to program in an FPGA. So a typical filter plan is to use a few CIC filters and then follow these by a single additional FIR filter to achieve the total filtering required.

Another important digital radio concept is sample rate reduction, or "decimation". Our ADC runs at 122.88 MHz and generates 14-bit samples, so its data rate is 1,720 megabits per second. We will not have much luck sending that through a 100 megabit per second Ethernet connection, so we will need to reduce the sample rate substantially. To do that, we first tune the original spectrum of signals to "baseband"; that is, toward zero hertz. Then, we low-pass filter the signal. We can then reduce the sample rate by eight times, for example, by keeping only every eighth sample. We use decimation on the receive path to reduce the sample rate. A reduction by 512 times gives 240,000 samples per second. On the transmit side we need to increase the sample rate ("interpolation") from 48 kHz to 122.88 MHz, an increase of 2,560 times. To increase the sample rate by eight times, we send a sample and then send seven zero samples. Both decimation and interpolation are usually done in stages with a rate change of two to maybe 40 instead of all at once. They both require low-pass digital filters at every rate change. I still find it amazing that the FPGA can handle all these calcu-
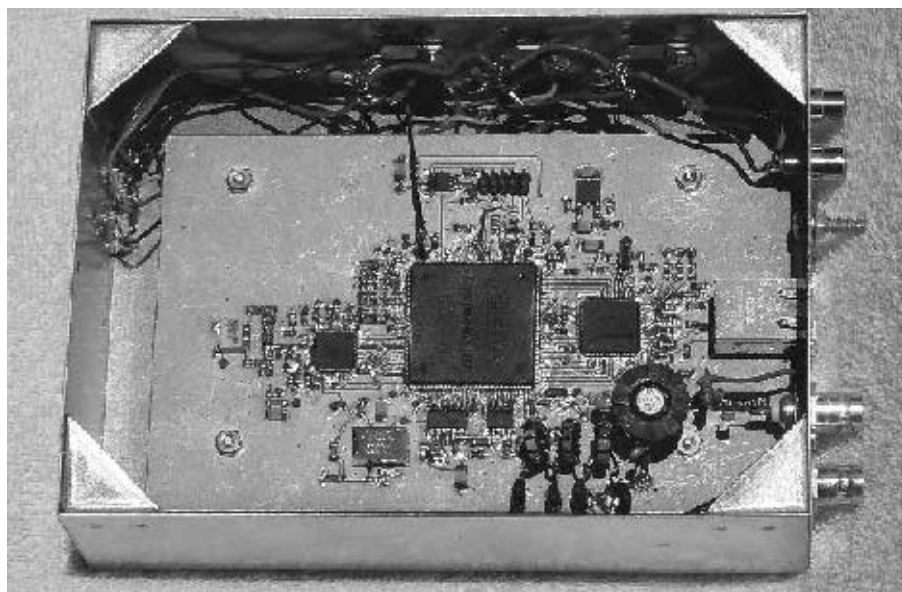
Figure 2 -- The printed circuit board.

lations at the raw data rate of 1,720 megabits per second.

Hardware Design

The hardware design is very simple. Its block diagram is shown in Figure 1, a photograph of the PC board is shown in Figure 2, and a list of the major parts is shown in Table 1. The FPGA is physically and logically the center of the design with the other chips grouped around its edges. The Ethernet controller is on the right in Figure 2 and is connected to the FPGA with a 16-bit data bus and some control signals. The Ethernet controller has its own 25 MHz crystal, some coupling and bypass capacitors, some load

resistors, and an RJ45 jack with built-in magnetics (transformers). The controller is a LAN9115 by SMSC and it operates full or half duplex at 10 or 100 megabits per second. It is a complete Ethernet solution and includes its own PHY (correct physical interface) and its own MAC (media access control) with speed negotiation capability. It has 16k bytes of memory shared between transmit and receive, and it stores received Ethernet packets until the FPGA gets around to processing them.

The transmit DAC is located below the FPGA in Figure 2 together with a smaller 8-bit DAC. The transmit DAC is 14 bits and operates at the 122.88 MHz clock frequency.

The 8-bit DAC generates 0.125 to 1.25 V DC to control the output level of the transmit DAC. I use this to reduce power to 50 watts on 60 meters, and to compensate for lower gain at higher frequencies in my amplifier chain. The output of the transmit DAC goes through a low-pass filter to a 2N5109 feedback amplifier.[7] Note that these are built directly on the board using "ugly" construction. Maybe next time I will make traces for them. The low-pass filter is designed for an input impedance of 180 Ω to reduce the loss from the impedance matching.

Referring to Figure 1, on SSB transmit the microphone audio is processed by the PC into in-phase and quadrature samples (I/Q) at 48 kilohertz, and sent via Ethernet to the FPGA. The FPGA interpolates (increases sample rate) and tunes these to the output frequency as described below, and sends the digital data to the DAC. The analog RF from the DAC is low-pass filtered and amplified, and sent to the power amplifier chain. For CW, note the 5 V key input. This is high for key down and low (0 V) for key up. It is used to directly generate CW in the FPGA to eliminate any pauses or delays that may be introduced by the PC.

The receive circuitry is located to the left of the FPGA in Figure 2. Input RF goes to a Minicircuits 1:4 unbalanced to balanced transformer, and then to an LTC6405 balanced preamp. The preamp output goes through a low-pass filter to the ADC. The low-pass filter is used to reduce noise above the Nyquist frequency of 61.44 MHz. This noise would fold back into the pass-band and degrade our noise figure. The ADC is a 14-bit Texas Instruments ADS5500IPAP operating at the 122.88 MHz clock. If the preamp and filter were omitted, the ADC could be used up to its analog bandwidth limit of 750 MHz, but that would compromise the performance at HF. The ADC connects to the FPGA and sends 14 bits of data plus a clock that is synchronous with the data. Referring again to Figure 1, the ADC samples go to the FPGA where they are tuned to baseband, decimated (sample rate reduced) and sent to the PC via Ethernet.

Below and to the right of the DAC is the Crystek 122.88 MHz low phase noise clock oscillator. Note that this connects directly to the ADC and DAC as well as the FPGA. This creates some problems, but I was worried about degrading the phase noise if I ran the clock through the FPGA to the other chips. The strange clock frequency was chosen because it is an integer multiple of 48 kHz, and the part was available from Digikey. We eventually need to play received audio at a rate that the sound card can handle, and 48 kHz is a good choice. If the original clock were not an integer multiple of the play rate,

Table 1
Major Parts List

Item	Part Number
FPGA	Altera EP3C25Q240C8N, 240 pins
FPGA Flash Memory	EPCS16SI8N in 8-SOIC
Ethernet Controller	LAN9115 by SMSC in 100-TQFP
Clock	Crystek 122.88 MHz CVHD-950-122.880
Receive ADC	125 MHz 14-bit Texas Instruments ADS5500IPAP in 64-TQFP
ADC Preamp	Linear Technology LTC6405 in 8-MSOP
Transmit DAC	Analog Devices 14-bit AD9744 in 28-TSSOP
Transmit Level Control DAC	Analog Devices AD7801 in 20-TSSOP
RJ45 Ethernet Jack	Stewart SI-50170-F
Ferrite Chips	Murata BLM41PG102SN1L in 1806

we would need non-integer decimation, and that is much more complicated.

Directly above the FPGA is its flash memory program storage chip and a 5X2 header to program the flash. Unlike a microcontroller, an FPGA lacks its own flash, so it reads its program out of the memory chip on power up. You may notice a bank of eight small LED's above and to the right of the DAC. These were used for debugging during development, and enabled me to output a byte of data. Now they are used to indicate various error conditions. On the left wall of the box are six LED's. Three connect to the Ethernet controller, and indicate link, activity, speed and full duplex. The other three connect to the FPGA and indicate ADC clip (over range), errors, and on-the-air status. On the top wall of the box are three low dropout voltage regulators for 5 V, 3.3 V and 1.2 V. There is another 2.5 V regulator on the board to the top right. Input power to the box is 6.0 V.

The PC board layout was done with Eagle[8] and all the files including a schematic are available from the authors Web site[9] or from the ARRL's *QEX* binaries site.[10] The parts used were chosen for performance, but also for availability and ease of construction. I chose parts with feet (OK, small feet) because I was not quite ready to try to solder leadless packages.

To me, the most interesting thing about this hardware design is that it doesn't do anything in particular. It has an Ethernet to FPGA link that could be carrying anything, an RF output from a DAC that could generate anything, and an RF input to an ADC that could sample anything. Just looking at the hardware, it could be a transceiver. But it could also be a spectrum analyzer with a tracking generator. To make it a transceiver we need software.

FPGA State Machines

FPGA software is intrinsically parallel, and, left to itself, it will execute every line of code at once. That can be useful, but it is not always what we want. For example, reading from a memory bus requires setting the address on the bus, waiting, asserting the read control line, waiting, reading the data from the bus, waiting, and finally de-asserting the read line. These operations must be sequential and correctly timed. Suppose we receive an Ethernet packet. We must examine the packet headers to decide what block of code should handle the packet, and this also requires sequential operations.

To obtain sequential operation from an FPGA, we use a state machine. A state machine has a state variable that can assume one of a number of values. Based on that value, we perform certain operations. To perform different operations in fixed order, we change the value of the state variable to the next state. For example, to program the memory bus read described above, we would start in the "set address" state. This state would set the address on the bus, set the "next state" variable to "assert read," and set the state to "wait." At the next clock, the state would be "wait." In the "wait" state we increment a counter, wait for a timeout, and then set the state to the value of "next state." In our example, after the timeout, the state would change to "assert read." At the next clock, our state is "assert read," so we assert the read control line, set "next state" to "read data," and set the state to "wait." After another wait interval, the state would change to "read data." At the next clock, our state is "read data," so we read the data from the bus and store it somewhere, set "next state" to "de-assert read," and enter another wait state. After the wait, the "de-assert read" state de-asserts the read control line, and changes the state to an "idle" state. Much of the time the FPGA has nothing to do, and is waiting to receive an Ethernet packet, or to have an ADC data block available to send. The "idle" state typically checks to see if there is any work to do. If there is, it changes the state according to the job to be done.

FPGA Software

There are three major blocks of software running in the FPGA: handle Ethernet reads and writes, write transmit data to the DAC, and read receive data from the ADC. All three blocks run simultaneously. All FPGA code is written in the Verilog language. The FPGA has a fixed IP address of 192.168.2.196, and uses three UDP ports. Port 0xBC77 receives control for the ADC samples and sends sample from that port. Port 0xBC78 receives control data such as transmit frequency and mode. And port 0xBC79 receives the transmit audio.

The simplest block of software reads the 14-bit data from the ADC at the clock rate of 122.88 MHz. It then tunes the data to baseband using CORDIC, and reduces the sample rate by a factor of eight with the first CIC filter. The data then goes to a second CIC filter that reduces the data rate by a factor of 2, 4, 5, 8, 10, 20 or 40 as programmed by a control byte sent from the PC. The data then goes to a final FIR filter that reduces the rate by eight. We now have a final sample rate of 48 to 960 kHz. Samples consist of two 24-bit numbers (I and Q), and are written to a memory buffer for transmission to the PC via Ethernet. The data rate at 960 kHz is 46.08 megabits per second or a little higher if we include packet overhead. That is a significant part of the 100 megabit Ethernet bandwidth and will stress all but the fastest PC's and Ethernet switches. At this rate we can see almost a megahertz of spectrum, and that is not much use on HF. But it could be useful on microwave frequencies if the hardware is used with a transverter. I usually set

my sample rate to 240 kHz as a compromise between seeing a good part of the band, and having the signals be wide enough to tune accurately with the mouse.

The next block of code is used for transmit. For SSB transmit, the PC sends audio samples at a rate of 48 kHz, and these are stored in a memory buffer in the FPGA. When the buffer is half full, the FPGA starts reading the samples; it then interpolates them by factors of 20, 16 and 8 in CIC filters and sends them to the DAC. The buffer provides a store of samples in case the PC samples are delayed. For CW transmit, the PC is not directly involved. Instead, the FPGA reads its key input pin. When the key goes down, a counter counts up to a maximum value, and when the key goes up, the counter counts down to zero. The counter is the input to the CORDIC mixer that tunes the DC counter to the output frequency. The result is a shaped key envelope with no delays from the PC.

The most complicated block of FPGA code is used to manage the Ethernet controller. First we initialize a register in the FPGA to zero. Then on power up, we start counting this register until we reach a timeout value. Until we reach timeout, we assert a reset pin connected to the Ethernet controller. Actually this reset pin is connected to all chips that can be reset, and is monitored by software routines that need initialization. In this way, we insure that power is stable before we start operating.

Next, we program some registers in the Ethernet controller by writing fixed byte values on the bus, and then we change the state of the FPGA state machine to "idle," In the "idle" state, the FPGA first checks

whether there is ADC data ready to be sent to the PC. If there is, it transmits an Ethernet packet with 42 bytes of header, a sequence byte, a status byte that includes the key up/down state, and 1,440 bytes of data consisting of 240 samples. If there is no ADC data, we check to see if an Ethernet packet was received. If no packet was received, we check the Ethernet controller for error conditions, and turn on the on-board LED's for any errors found. Then we enter a short wait and repeat.

If we receive an Ethernet packet, we read the first 80 bytes into a buffer, and examine the buffer to see what to do next. If the packet is an ARP request for our IP address, we send an ARP response. The ARP protocol is used to associate Ethernet addresses with IP addresses, and it makes the transceiver play nicely on your network. If the packet is a ping request to our IP address, we reply with a ping response. Ping isn't completely necessary, but it is useful and should be included in any Ethernet appliance. If the packet is a UDP packet addressed to us we continue to process it. Otherwise it is silently discarded, and we read and discard any remaining bytes in the Ethernet controller to clear it for the next packet.

If the UDP packet is addressed to our ADC sample port and has data 0x7272, we record the return address and use it to send our ADC samples. That way we do not have to program the IP address of the PC into the FPGA. If the data is 0x7373, we stop sending ADC data. If the UDP packet is addressed to our control port, we read the transmit frequency, receive frequency, the transmit level (for the 8-bit DAC), the transmit mode (CW
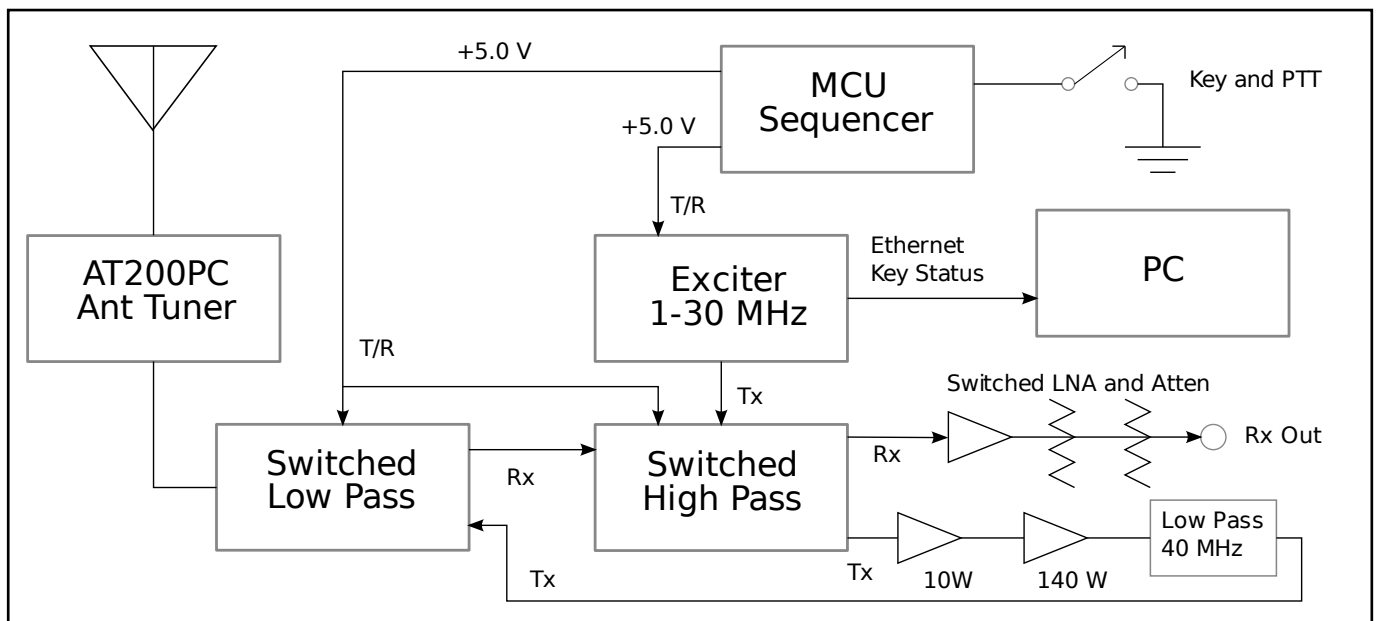

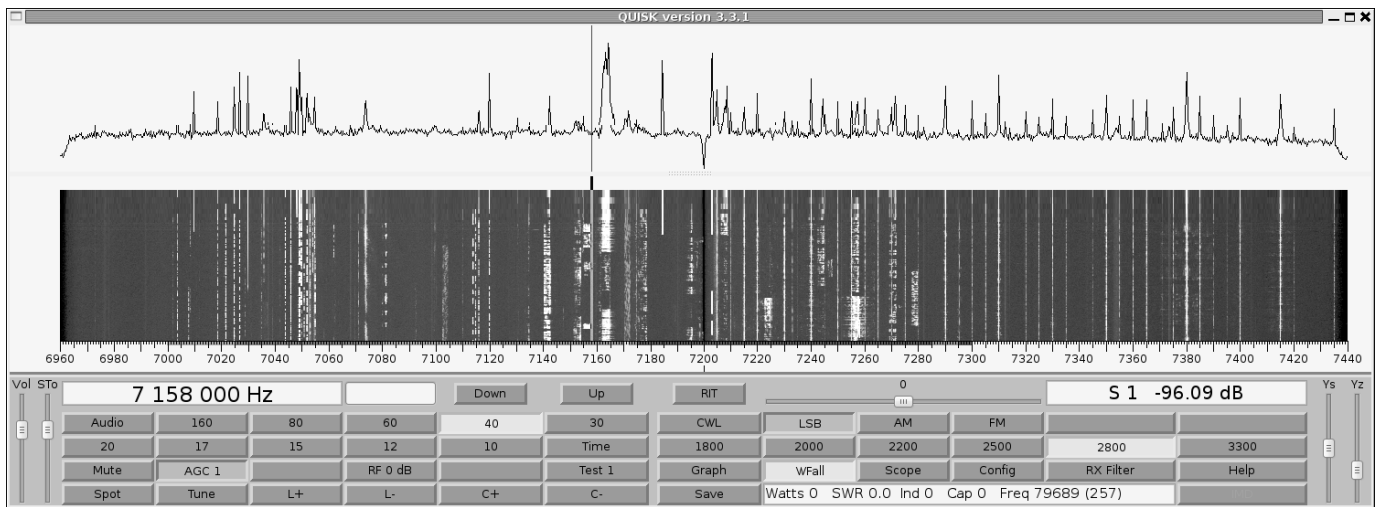**Figure 3 – A block diagram of my station control.**

Figure 4 – A screen shot of Quisk running on my PC.

or SSB) and the requested stage two decimation for the ADC. We then echo the control packet back to the sender as an acknowledgement. If the UDP packet is addressed to our transmit audio port, we skip the two byte sequence number, write the audio from our buffer to memory, and then copy the remainder of the audio from the Ethernet controller to memory. The data is two 16-bit numbers (I and Q) at 48 kHz, or 1.536 megabits per second (plus overhead), a modest rate even on 10 megabit Ethernet.

Station Control

To make a complete station the exciter has to be combined with a power amplifier chain and filters. A block diagram of my station is shown in Figure 3. The antenna tuner is an AT200PC by LDG controlled by the PC through a USB to serial adapter. There are two filter boxes consisting of conventional LC filters switched by relays. The low-pass filters are the usual high power (150 W) transmit filters. The high-pass filters are low power LC filters. The combination results in a band pass response on both receive and transmit.

On receive, the signal goes through the low-pass filter, the high-pass filter, and then possibly through a switched preamp or attenuators. On transmit, the RF output from the exciter goes through the high-pass filter, then to a series of power amplifiers, then through a TVI filter, and finally through the low-pass filter to the antenna. My current antenna is a fan dipole for 60, 40 and 20 meters.

The relays require sequenced switching, so the key is connected to a microcontroller that generates keying signals for the relays and exciter. When the key goes down, the relays are switched first, followed by the exciter, and the reverse on key up. Note that there is no key line to the PC even though

the PC needs to know the key state so it can mute the audio on transmit and substitute a sidetone on CW. The key status is sent to the PC using Ethernet. It is encoded in the second byte of data in an ADC sample block. At one time I used a pin on the parallel port, but newer PC's often don't come with a parallel or serial port.

The station control shown is quite generic, and has been used with several past versions of my station. It worked just as well with my SDR-IQ receiver and 2008 exciter. I am continuing to work on the power amplifier chain, as I am having trouble keeping the gain flat with frequency and the IMD low.

Quisk Software

My Quisk software running on a PC controls my whole station. My PC is an old 3 GHz Pentium 4 with one core. The PC controls the AT200PC antenna tuner through a USB to serial adapter; it controls the filter boxes through Ethernet and an I2C gateway; and it directly controls the transceiver by Ethernet. A screenshot of Quisk is shown in Figure 4. It shows the whole 40 meter band sampled at 480,000 samples per second. Quisk software is available on my Web site.[11]

Quisk was originally designed for CW QSK operation using the sound card as the receiver, and a DDS chip for transmit. Gradually I added the capability to control more hardware: an SDR-IQ receiver by RfSpace, my 2008 exciter, an AOR receiver with an IF output (as a panadapter), and now my transceiver. Quisk uses an external hardware file written in the Python programming language to control its hardware. To use different hardware, you just change this file. You can use one of the hardware files that comes with Quisk, or write your own. There is also a way to add additional controls to Quisk by adding a custom *Python* "wid-

gets" file. *Python* is a powerful language that is very easy to learn. I use it for Quisk and for nearly everything else. Its complex data type is handy for impedance calculations, for example.

The top level of Quisk is written in *Python*, and controls the graphical user interface (GUI), the graph and waterfall screens, and any custom hardware. The rest of Quisk is written in the *C* language, and this code is responsible for the sound cards, the USB interface, the Ethernet IP ports, the FFT (fast Fourier transform) and the digital signal processing. Linking *Python* with *C* in this way is very common. The main program quisk.py is 2300 lines, and the *C* code is 4500 lines. This is a very small program.

Quisk was never intended to be a "product" in the sense that it had every feature imaginable and could run any hardware right out of the box. I always saw it as a starting point for other people's homebrew projects; something that would be simple and understandable, and easy to adapt to different hardware and software designs. That is one reason that Quisk runs on Linux. Linux comes with *Python* and *C* compilers, and it provides a rich software development environment for free. Being simple, small and easy to change is one of the design goals of Quisk. But even though Quisk is simple, and after using other available radio software, I find I prefer Quisk. Some of that is no doubt pride of authorship, but I want my radio to look like a radio, not a Windows program, even though it is an image on a computer screen. And I find that Quisk is easy to tune and has the features I need.

Although I use Quisk to control my transceiver, that is not a requirement. The transceiver is simply sending and receiving I/Q samples just like any other software defined radio. Any software can provide the data the

transceiver requires. All that is needed is an Ethernet IP/UDP interface, and that is simple to supply.

Quisk Internals

Quisk runs in two threads, a GUI thread that runs the user interface, the screens and the FFT, and a second thread that is responsible for reading and writing the sound and performing digital signal processing. When used with my transceiver, Quisk operates as follows. On receive, the sound thread reads a UDP port at timed intervals and collects any ADC samples available. These samples are first copied into an array for the FFT. When enough samples are available, the GUI thread will perform the FFT, average the results, and eventually update the screen. The sound thread then tunes the signal to baseband in a digital mixer, reduces the sample rate, filters the signal and divides it into an I/Q pair and demodulates it. For FM, there are additional filters at this point. There may be more decimation or interpolation depending on the sample rate and source. Finally, AGC is applied and the sound is played on the sound card.

For SSB transmit, the sound thread reads the microphone and boosts the high frequen-

cies in a digital filter. It then filters the audio and divides it into an I/Q pair. This signal is mixed to a higher frequency, clipped, refiltered, and mixed back down to baseband. These audio samples are sent to the UDP port and then to the transceiver.

There are a few other interesting features. The S-meter in Quisk is calculated by squaring and averaging the correct number of FFT bins to equal the indicated filter width. This means that Quisk has a true RMS voltmeter with a known noise width that can be used to make noise measurements. Quisk can generate a two-tone test pattern to make IMD measurements. Quisk generates these tones in the PC software and sends them as an SSB signal. And Quisk has a spot switch that sends a CW carrier in SSB mode. That is not too exciting, except that since everything is digital, the level is guaranteed to be at PEP.

Problems

I am very happy with this transceiver hardware design, but there are still some unresolved problems and directions for future work. I measured the noise figure at 23 dB. Although this is usable, I was expecting a noise figure closer to the preamp noise at 200 Ω of 8 dB. I found this very confusing, but thanks to an email[12] from Jeff Millar, WA1HCO, I now understand the high noise figure. He calculated the noise figure of the ADC alone to be 30 dB. The preamp gain is not sufficient to dominate the noise figure, and using the usual formula $F = F1 + (F2 - 1) / G1$ gives the net noise figure I observe. I do not want more preamp gain, as that would degrade the dynamic range. I have an additional preamp external to the transceiver that I can switch in for 20 meters and up. The purpose of the preamp is to make it easier to drive the receive input.

I am also worried about the clock integrity. I connected the clock oscillator to three IC's, but there is no spec for how many IC's the clock will drive. I could connect the clock to the FPGA and run the clock straight through, but that might degrade the jitter. I could use a clock distribution IC, but the ones I saw had worse specs than my oscillator. I don't have any evidence that the clock is questionable, but I don't have any equipment to measure the clock noise either.

And then there is the power amplifier chain. The IMD spec out of the DAC is exemplary, but it quickly degrades with amplification. On 10 meters, it winds up about 23 dB below one tone, and I am working on improving this. Even good quality commercial ham equipment seems to have IMD specs in the low 30 dB range below PEP. If I can get a cleaner power amplifier chain, then maybe I can bend the response curve in software to improve the IMD.

Conclusion

It has been great fun working with these new digital IC's and getting a lot of hands-on experience with digital signal processing. I am not convinced that an all-digital radio is better than the best analog radio, but I think it is easier to homebrew one, especially since all the software is readily available for free. When I get on the air with it and say that my rig is homebrew, my ham friends take an interest, and that is very gratifying. I would also like to thank the hams that have emailed me and expressed an interest in my work.

I also learned that even in this digital age, some things in radio never change. As pointed out by Wes Hayward et al, building a receiver is a great way to learn humility. If there were a magic bullet formula for a receiver, they would all be built the same way, but instead, receiver design remains a challenge. And filters remain at the heart of radio whether digital or analog. And then there are antennas. It was amazing to me how much better my first digital receiver worked when attached to a good antenna! When I tune across the bands, I hear hams with digital rigs, analog rigs, and some with classic Collins or Drake gear, but one thing I know for sure; they definitely have an antenna.

James Ahlstrom, N2ADR, was first licensed as KN3MXU in 1960. He received a BS in physics from Villanova University in 1967 and a PhD in physics from Cornell University in 1972. He then moved to New York to work in the financial business. He is currently a one-third owner of Interet Corporation, Millburn, New Jersey. Interet publishes software to analyze leveraged equipment leases. His license lapsed while raising his family, and he was re-licensed in 2006. He currently holds an Amateur Extra class license. Besides Amateur Radio, he enjoys bird watching, skiing, music and working out at the gym.

Notes
[1] James C. Ahlstrom, "An All-Digital SSB Exciter for HF", *QEX* May/June 2008, pp 3-10.
[2] J. Bhasker, *Verilog HDL Synthesis, A Practical Primer*, Star Galaxy Publishing, 1998.
[3] J. Bhasker, *A Verilog HDL Primer*, Third Edition, Star Galaxy Publishing, 2005.
[4] Richard G. Lyons, *Understanding Digital Signal Processing*, Second Edition, Prentice Hall, 1996.
[5] **www.arrl.org/software-defined-radio**
[6] Ray Andraka, "A Survey of CORDIC Algorithms for FPGA Based Computers", **www.andraka.com/files/crdcsrvy.pdf**.
[7] Wes Hayward, Rick Campbell and Bob Larkin, *Experimental Methods in RF Design*, ARRL, 2003, ISBN: 0-87259-879-9.
[8] **www.cadsoftusa.com**.
[9] **www.james.ahlstrom.name/transceiver**
[10] **www.arrl.org/qexfiles**
[11] **www.james.ahlstrom.name/quisk**
[12] Jeff Millar, WA1HCO, personal communication

QEX

John Pivnichny, N2DCH

3824 Pembrooke Lane, Vestal, NY 13850; JohnPivn@aol.com

A Two Diode Frequency Doubler

John presents a passive doubler that produces a cleaner output than some active doublers and gives an analysis of the effects of amplitude and phase imbalance.

A schematic representation of the two diode frequency doubler is shown in Figure 1. A toroid core creates two signals which are of equal amplitude and 180 degrees out of phase with each other. Two identical diodes allow only the positive portion of each signal to pass through to the output, producing the ideal waveform of Figure 2. The 50 Ω resistor load may be replaced with an RF choke followed by a series coupling capacitor and a separate 50 Ω load.

The waveform of Figure 2 has more than just the second harmonic present as evidenced by the pointed shape near the bottom of the waveform. These sharp cusps are removed by filtering. Either a low-pass filter with a cutoff above the second harmonic, or a resonant filter tuned to the second harmonic is typically used. Nevertheless, even before filtering, the second harmonic has significantly higher amplitude than the fundamental. This big difference is the main reason for using the circuit instead of a conventional class C amplifier with output tuned to the second harmonic, as we will discuss later.

Any difference in the amplitude of the two signals at the diodes or phase deviation from 180 degrees will of course alter the ideal waveform of Figure 2. Likewise any differences in the two diodes from perfect diodes or from each other will also affect the output waveform. This paper describes the effect of these differences on the operation of the two diode frequency doubler.

Hayward, W7ZOI, and DeMaw, W1FB, describe their measurements of the circuit in *Solid State Design for the Radio Amateur.*[1] They also describe several practical applications elsewhere in the book. In most cases the doubler is followed by a single or double tuned tank circuit tuned to twice the input frequency to filter out any fundamental feed through or higher harmonics which are created by the circuit. Their measurements are
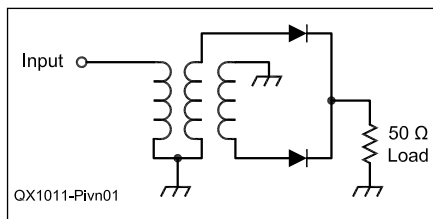
[1]Notes appear on page 11.

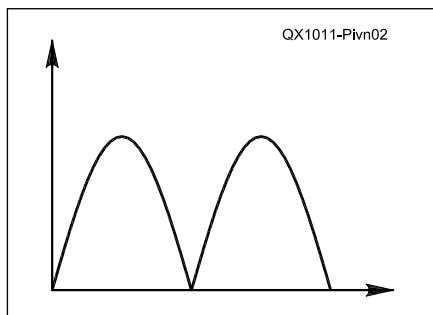**Figure 1 — Schematic of the two diode frequency doubler circuit.**

Figure 2 — Two half sine waves.

taken with a 50 Ω source and a 50 Ω load. In general, they report the desired second harmonic signal is about 10 dB below the input fundamental. However, the fundamental at the output is about 40 dB below its amplitude at the input, undoubtedly due to the inherently balanced nature of the circuit of Figure 1. These values were measured before any improvements provided by a single or double tank circuit filter, but with real diodes and real hand wound toroid cores. This is quite a remarkable result!

Hayward and DeMaw compare this great difference (40 dB) between the desired output and the undesired fundamental feed through, as well as third and higher harmonics, to an ordinary frequency doubler using a weakly biased bipolar transistor with its output tuned to the second harmonic. This ordinary doubler has only about a 13 dB dif-
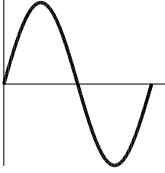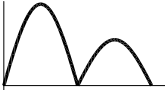
ference between the desired and undesired output signals despite the filtering action of a single tuned circuit.

As noted earlier the two diode frequency doubler depends on a balance in amplitude and a phase difference of 180 degrees, between the two signals provided at the junction of the two diodes. The effects of a departure from this balance in amplitude or phase difference can be theoretically calculated using the Fourier transform. An input of two half sine waves in sequence as shown in Figure 2 is used. This is what would be seen at the junction of the two diodes if there was perfect balance, ideal perfect diodes, no phase error, etc. The amplitude of the second half sine wave is varied to simulate an unbalanced amplitude condition. This unbalance in a practical circuit could be the result of various stray capacitances present in the toroid coil affecting each of the two secondary windings differently. It could also indicate an unintentional difference in the number of turns in each secondary or unmatched diodes or a defective diode being used.

The first four rows of Table 1 show the calculated results. The amplitude of the dc component is shown along with the fundamental feed through, and the desired second harmonic. The input signal is shown in the first row for comparison. If one diode were shorted and the other open, then this input signal would appear at the output. There is no dc component. The amplitude is 16 and the second harmonic zero. This value of 16 can be viewed as a scale factor and represents the peak amplitude of the output sine wave. For example it could represent milliamps through the output termination of 50 Ω or about a 7 dBm signal.

The second row of Table 1 shows the results for the ideal signal of Figure 2. Note there is no fundamental feed through because of the perfect balance. The dc value

Table 1

Signal	dc component	Fundamental component	Second Harmonic
	0	16.0	0
100%	20.31	0	6.85
50%	15.23	4.0	5.14
0%	10.15	8.0	3.42
22.5°	20.11	2.96	6.52
67.5°	18.04	7.96	3.28

QX1011-Pivn04

Figure 3 — A single tuned circuit 910 kHz filter.

Figure 4 — The 455 kHz signal from IF signal generator.

Measured Results

Measurements were taken to help confirm some of the ideas presented above. The circuit of Figure 1 was built using 16 turns trifilar wound #30 gage wire on a FT50-43 ferrite toroid core. Each secondary winding was initially terminated with a 100 Ω resistor, effectively a 50 Ω load on the input.

The input signal is 455 kHz as shown in Figure 4. This signal is a fairly pure sine wave as produced by the IF signal generator and filter described in the QEX article.[2] The filter in the signal generator provides extremely good rejection of the second and third harmonics as well as over 65 dB rejection of higher harmonics. The signal has an amplitude of about +5 dBm as measured on my dBm meter.[3] Each of the secondary outputs, when terminated in 100 Ω (before connecting the diodes), is shown in Figure 5. Note the 180 degree phase difference and excellent amplitude balance. This wound toroid (without the 100 Ω resistors) is then used in the circuit of Figure 1, with two randomly selected 1N4148 silicon diodes.

Figure 6 shows the output on the top trace with the input at the bottom. Notice that this output goes positive at twice the rate of the input because each secondary winding provides a positive going pulse. However, there are gaps between the pulses caused by the diodes not conducting until their forward voltage exceeds about 0.6 volts. Also note the difference in amplitude of alternate pulses, most likely due to differences in the two diodes.

A single tuned circuit filter was constructed as shown in Figure 3. L2 consists of

---

of 20.31 is easily removed by the use of a series coupling capacitor at the output. The second harmonic has a peak value of 6.85 or about 0 dBm. This is more than measured by Hayward and DeMaw, however their measurements included the loss across the diodes. Figure 2 has no such loss (ideal diodes).
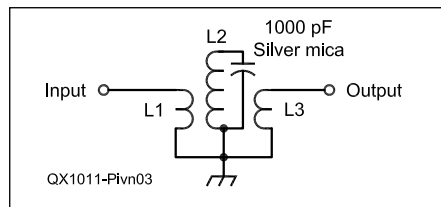
In the third row of Table 1, the second half sine wave has an amplitude 50% of the first half sine wave which is a significant unbalance. Notice the lower dc component and a lower but still fairly strong desired output at the second harmonic. The strong fundamental feed through is nearly as strong as the desired output. Clearly some serious filtering would be needed to remove this component from such a seriously unbalanced frequency doubler.

Proceeding even further, the fourth row of Table 1 shows what happens if one diode opens up. The peak amplitude of the second half sine wave is zero. This also represents the operation of an ordinary frequency doubler without a tuned circuit and no gain. The desired output
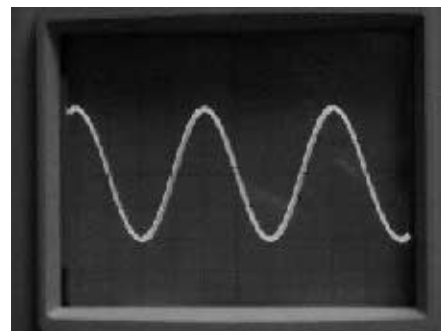
is even weaker but still fairly strong. But note the fundamental is now more than twice the strength of the desired output. Even more filtering is needed to make this work.

In the fifth and sixth rows of Table 1, the second half sine wave has a peak amplitude equal to the first one but is shifted left 22.5 degrees and 67.5 degrees respectively. The sketch shows the output waveform resulting from the diode action assuming equally matched perfect diodes. Such phase shifts could result from stray capacitances, particularly at higher frequencies such as in the lower VHF ranges. The author has observed shifts at 50 MHz and above.

The output with 22.5 degree shift is similar to the 50% amplitude unbalance result but slightly better. With a 67.5 degree shift, the output is nearly the same as with the open diode. These results seem to indicate that it may not be worth trying to build a perfectly balanced wound toroid at VHF and use of a single diode doubler with extra filtering may be a better overall approach.

82 turns of #32 gage wire wound on a T68-2 toroid core, providing an inductance of 33 uH. This resonates with the 1000 pF capacitor at 910 kHz. L1 and L3 are each 7 turns of #32 gage wire. The load resistor was moved to the output of this filter. The top trace of Figure 7 shows the waveform after filtering. Note that the vertical scale of the oscilloscope was made more sensitive to more accurately show the waveform. Some higher order harmonics are evident because this is not a perfect sine wave. There is also still some evidence of fundamental feed through indicating that additional filtering is in order. DeMaw and Hayward use a double tuned filter after doubling to 144 MHz. Based on the author's experience, it may be necessary to use double tuning in the filter in many cases, and especially at VHF, if a reasonably pure sine wave is desired after using the two diode frequency doubler of Figure 1. Nevertheless, the two diode frequency doubler provides significant improvement over ordinary frequency doublers as long as there is a reasonable balance in the two signals provided by the wound toroid core. Table 1 serves as a guide to the results to be expected for various levels of unbalance in either amplitude or phase.
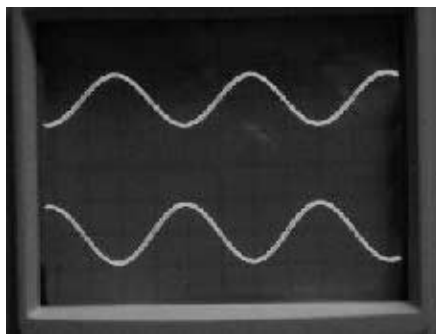
Figure 5 — Two outputs from wound toroid core
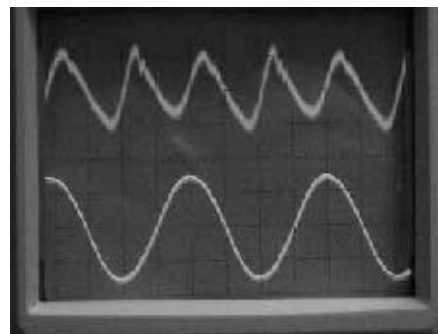
Figure 7 — A 910 kHz signal after filtering with circuit in Figure 3.
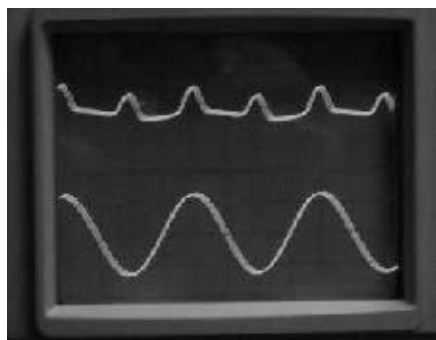
Figure 6 — The output from the circuit in Figure 1.

John Pivnichny, N2DCH, has been licensed since 1956. He has advanced degrees in Electrical Engineering. John has written several articles published in a number of Amateur Radio magazines. He is the author of Ladder Crystal Filters, published by MFJ Enterprises. John is currently interested in encouraging ham radio VHF experiments and operating in the 222 MHz band.

Notes:
[1] *Solid State Design for the Radio Amateur*, ARRL, Newington, CT, 1986 pages 41-42.
[2] "An IF Signal Generator," *QEX*, Sept/Oct 2006, pp 25-27.
[3] "dBm Meter", *Electronics Now*, November 1995, pp 112-113, 158-159.

QEX

Sivan Toledo, 4X6IZ

School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel, stoledo@tau.ac.il

A Driverless Ethernet Sound Card

*Sivan describes a high performance Ethernet interface to a
PC that is portable across operating systems.*

A growing class of radios relies on a laptop or desktop computer to perform signal processing and to implement the radio's user interface. Examples include Flex Radio's transceivers (starting from the SDR-1000[1] and now including the Flex-3000 and Flex-5000), the Softrock kits[2], and others. In all of these radios, which I will refer to as software-defi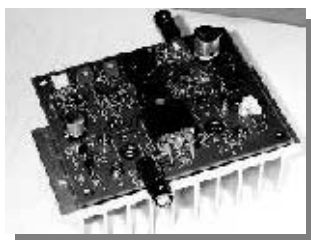ned radios (SDRs[3]), a radio front-end is connected to a generic personal computer (PC). This connection carries a complex baseband signal, radio status information, and commands to the radio.

Many radios in this class transfer analog baseband signals to/from the PC. The PC digitizes the incoming signal and produces the outgoing signal using a sound card, either built-in or external. The status and command communication in early radios used the serial or parallel port of the PC, but recent radios tend to use a USB interface. The overall architecture is shown in Figure 1.

[It is the consensus of the author and our technical reviewers that Hertz is an improper unit to describe sample rate. There is no SI unit for sample rate, so we have to choose something appropriate. [We will use "SPS" to mean samples per second.—Ed]

Some PC sound cards have high dynamic range and moderate sampling rates (up to 192kSPS). Therefore, using a sound card is a reasonable low-cost solution, but it has some disadvantages. One possible disadvantage is the difficulty of finding high-quality reasonably-priced sound cards. The archives of SDR user groups' mailing lists suggest that finding a suitable sound card is challenging for many. Another disadvantage of using a consumer sound card is the possibility of rapid obsolescence. Cards that come with a special driver, which is the case with many high-end cards, only work on operating systems for which the vendor supplies drivers. A sound card that works fine under *Windows XP*, for example, may not be supported by the vendor on *Windows 7*, or it may not be supported on *Linux*. The third disadvantage of relying on sound cards is their limited bandwidth, at most 192kSPS.
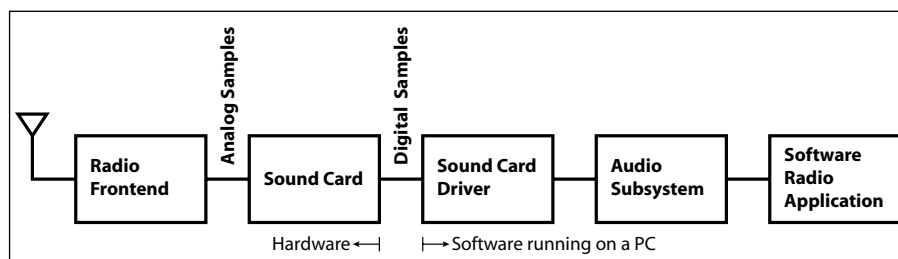
To address these issues, some recent radios include a sound card or non-audio analog-to-digital and digital-to-analog converters (ADCs and DACs). Such radios include some Flex Radio products[4], which include a built-in sound card, the HPSDR project, and the SDR-Widget project (which aims to build a low-cost high-performance sound card for softrock-type radios), and others. Radio front-ends that sample faster than 192kSPS also include ADCs and DACs; these include the USRP radios, the radios built by Pieter-Tjerk de Boer (PA3FWM), and others[5,6].

This article presents a novel way to communicate baseband samples to/from a PC. More specifically, I will describe a sound-card prototype that uses the most ubiquitous Internet communication protocol, TCP/IP, to communicate these samples. This approach has significant advantages over the common approach which uses a USB connection.

From USB to Ethernet

Most external sound cards use a USB connection to transfer audio samples to/from the PC. These cards usually use the standard USB audio protocol[7], but they can also use custom USB protocols. Most operating systems support the USB audio protocol. Therefore, cards that use this protocol do not need a card-specific driver, and they work with all operating systems. The USB audio protocol uses a USB data transfer mode called *isochronous transfer*[8]. The isochronous mode allows the PC to choose a sampling rate for which it can ensure sufficient bandwidth on the USB bus. The PC dedicates a fixed fraction of each USB frame to the audio data transfer. Isochronous transfers are not reliable: missing or corrupt packets are not retransmitted. This is a reasonable and common design decision for real-time data. Sound cards that use a specialized driver can also use isochronous data transfers.

USB sound cards suffer from several problems. One problem is limited support for high sampling-rates and for 24-bit samples. Release 1.0 of the USB Audio standard did not support high-end cards, which forces high-end cards to use card-specific drivers. This problem may disappear over time, since release 2.0 of the standard allows higher sampling rates and 24-bit samples, but operating system support for this version is still patchy. A second problem is

Figure 1 – General SDR architecture when using a PC Sound card.

that the USB driver transfers the samples to/from the operating system's audio sub-system. The software interfaces (APIs) that the audio subsystem presents to applications vary widely between different operating systems. Some operating systems have multiple audio interfaces with different capabilities (e.g., DirectSound and ASIO in *Windows*). This makes it difficult to write portable SDR applications. (There is a good reason for this complexity: multimedia applications like games and video editing need low latency control of audio streams, but this is less important for SDR applications.)

An Ethernet connection to the sound card solves these problems. If the sound card sends and receives samples using UDP or TCP, it communicates directly with the SDR application without passing through the operating system's audio subsystem. The data still passes through device drivers on the PC, but now these are the networking drivers, which are built into all operating systems. Furthermore, the software interfaces to net-working functions are essentially the same across all operating systems, unlike the inter-faces to the audio subsystem. This makes it easy to write portable SDR applications.

The Ethernet frames carrying the base-band samples can be transported on a direct Ethernet cable connecting the sound card and the PC or through an Ethernet switch. They can also be transported though a USB con-nection, using a standard Ethernet-over-USB protocol; *Windows*, *Linux*, and *MacOS* come with built-in support for USB Ethernet cards, so no card-specific driver is needed. The pos-sible interconnect architecture is shown in Figure 2.

The sound card itself contains digital-to-analog (DAC) and analog-to-digital (ADC) chips, or a chip with both (usually referred to as a CODEC) and a microcontroller. Low-end USB sound cards use a single special-purpose chip that serves both as a CODEC and as a USB interface, but in this article the focus is on higher performance designs with a microcontroller. Until recently, micro-controllers with a USB interface were sig-nificantly cheaper than microcontrollers with an Ethernet interface, and there was a wider selection of the former. Today, however, there are reasonably-low cost microcontrollers with a complete 100Mb/s Ethernet inter-face, including the physical signaling. The main external components required are the Ethernet socket and the isolation transformer, which can be built into the socket. The pro-totype described in this article uses such a microcontroller, the Texas Instruments LM3S9B96. This microcontroller also has an interface to audio DACs, ADCs and CODECs, making it suitable for sound cards.
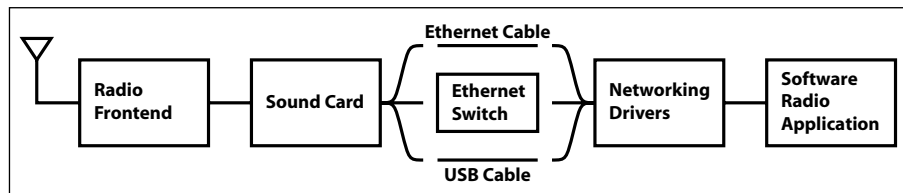

Figure 2 – SDR architecture when using Ethernet connection to the PC.

The Prototype: Hardware and Software

The Ethernet soundcard prototype is built around a simple microcontroller evaluation kit called EK-LM3S9B96 (Figure 4). [The EK-LM3S9B96 was a limited edition kit that is no longer available. The EK-LM3S9B92 uses an MCU with the exact peripherals and capabilities of the LM3S9B96 but does not include SafeRTOS or IEEE1588 in its Flash ROM—Ed.] The board contains the micro-controller and its oscillator crystals, a voltage regulator, Ethernet and USB sockets (I did not use the USB connection), and program-ming and serial-port connections to a small interface board. The interface board con-nects to a PC through a USB cable, allowing the PC to program the microcontroller and allowing the microcontroller to send back diagnostic or debugging information.

The evaluation board makes some of the microcontroller pins available through 0.1" headers, including all the CODEC interface pins. I mounted the evaluation-kit board on a large prototyping board that also had space for the CODEC, a Texas Instruments TLV320AIC23B. The line-in port and head-phones output ports of the CODEC are con-nected to standard ¼" stereo jacks.

The software on the microcontroller is based on example code that came with the microcontroller. The networking software I used is *lwIP* (light weight IP), a small and free implementation of IP, TCP, and UDP. *lwIP* provides several programming inter-faces. I used the raw interface, which is the lowest level interface, since it provides maxi-mum performance and control.

I wrote the PC software in Java. I tested it on *Windows XP*, but it should run unmodified on *Linux*, too. The rendezvous or enumera-tion mechanism that allows the PC applica-tion and the sound card to find each other is described later in this article.

Overcoming Scheduling Jitter

The main challenge that I faced in this project was to drive the sound card's DAC. The common wisdom is that UDP is the appropriate Ethernet protocol for transport-ing real-time data, such as audio[9]. UDP is an IP protocol that is used to send so-called *datagrams*, or discrete limited-length pack-ets of data between applications. UDP is not

reliable; the protocol itself does not acknowl-edge packets and does not retransmit lost or corrupt packets, although applications can implement acknowledgements and retrans-mission for important data.

In spite of the common wisdom, it turns out that UDP does not work well when one side in the communication channel is a microcontroller with a limited amount of RAM. The problem is not with receiving data from the sound card, but with sending data to it; the problem turns out to be more severe under *Windows* than under *Linux*.

To understand the problem, let's consider what happens when the sound card sends and receives baseband samples using UDP pack-ets. We will assume that the PC processes incoming baseband samples into audio that it sends to an internal sound card, and that it sends baseband samples that are derived from its internal sound card's microphone input.

Let's start with the direction that does work well, sending data from the sound card. The microcontroller receives a baseband sample from the ADC or CODEC every few microseconds and puts it automatically in a hardware queue. When the queue reaches a certain level, it generates an interrupt signal that causes the microcontroller to remove samples from the queue and place them in a buffer in RAM. When this buffer has enough samples to be worth sending in an Ethernet frame (usually up to 1500 bytes), the micro-controller sends them to the PC in one UDP packet. A packet every millisecond is a rea-sonable rate; at 24-bits per sample, two sam-pling channels, and 192kSPS, a millisecond of audio is 1152 bytes, small enough to trans-port in one packet. At lower sampling rates or lower resolutions we can either send shorter packets or send less frequently. When the PC receives a packet, its network card generates an interrupt that causes the operating system to process the packet. The operating system stores the packet in a buffer until the SDR application retrieves it. This operating system buffering response happens even if the PC happens to be running some other application when the packet arrives. At some later time the PC's operating system will switch to the SDR application. When the application will reach the point where it reads UDP packets, the packet will be delivered to it.

Table 1

Comparison of Task Latency Across Scheduling Parameters

Time in ms

	Windows XP Normal Priority		Windows XP Realtime Priority		Linux Normal Priority	
	Maximum	Typical	Maximum	Typical	Maximum	Typical
Idle	146	125	126	125	43	43
One Core Busy	155	125	129	125	51	43
Two Cores Busy	--	--	129	125	64	43

The PC's operating system can buffer lots of packets. If many packets are buffered when the SDR application is scheduled to run, it will receive all of these packets, process them, and create audio for the internal sound card. The buffering and the fact that processing is delayed do not prevent the software radio from producing continuous audio; they only mean that the audio that the user hears was received some time ago, say a quarter or half a second.

Now let's consider the other direction, which is more troublesome. The PC's operating system runs the SDR application once in a while. The application reads audio from the internal sound card. The operating system buffers this audio, so it is not lost even if the SDR application is suspended for hundreds of milliseconds. The SDR application processes this audio, produces baseband samples, and sends them to our sound card in UDP packets. If the SDR application is scheduled to run only every 100ms, say, what the sound card sees is a period of almost 100ms with no data at all, and then a burst of UDP packets. A burst after 100ms will contain about eighty 1500-byte packets at 24-bit and 192kSPS.

The large, infrequent bursts cause two problems. One is the need to buffer enough baseband samples to bridge over the periods in which no data is received. At 48kSPS and 16-bit samples, the microcontroller only needs to buffer 9.6KB to be able to feed the DAC for 100ms. But at 192kSPS and 24 bits, it needs to buffer 115KB. This is possible with an external RAM chip, but usually not with the internal RAM of a microcontroller (the LM3S9B96 has 96KB of RAM). The other problem is the high likelihood of lost packets. The PC can send the packets in such a burst much faster than a relatively slow microcontroller can process them, so many of these packets are likely to be lost.

I implemented such a protocol. It worked fine at 48kSPS but not at 96kSPS. At this rate, there were many gaps in the analog baseband signal produced by the sound card.

The severity of this problem depends on how often the PC's operating system schedules the SDR application. To measure this, I wrote a small application that simply reads

(and discards) audio from the internal sound card. The program reads a fixed amount of audio every time. The amount of audio ranged from 1ms worth of audio to 100ms. The program measured the time periods between successive audio-read operations. Ideally, these periods would be close to t milliseconds when the program asks for t milliseconds worth of audio. But this is almost never the case.

Table 1 shows the actual periods that I measured for t=10ms. The experiments were conducted on a Dell D820 laptop connected to ac power. The laptop has a dual-core Intel T7200 processor running at 2GHz and it ran either *Windows XP* SP3 or *Linux* 2.6.31. The test program is written in Java, so exactly the same code runs under *Linux* and *Windows*. The results for other values of t, ranging from 1ms to 40ms, were similar; the results at 60ms, 80ms, and 100ms were similar under *Windows* but showed increased latencies under *Linux*. I ran the experiment when no other application was running (the *idle* row), when another application ran one CPU-intensive thread that used one of the two available cores (CPUs), and when the other application ran two CPU-intensive threads. The maximums are over experiments lasting 100s. The typical values are derived from the number of periods greater than 100ms in *Windows* and greater than 25ms in *Linux*; these numbers were always close to 800 and 2344, respectively.

The results show several interesting facts. The first is that both operating systems schedule the application at roughly constant intervals, even when the application asks for just a small amount of audio every time. *Windows* suspends the program for about 125ms, then runs it until it extracts all the available audio. At this point the program blocks while waiting for more audio. *Windows* suspends it again for 125ms rather than until the amount of audio it asked for is available. *Linux* does the same, but with a shorter scheduling interval of about 43ms.

The other interesting fact is that *Windows* does not run the application often enough when the CPU is busy. When only one core is busy, the maximum scheduling latency for processes with a normal priority rises a

bit, but the program still runs often enough to process all the audio. When both cores are busy, *Windows* does not run the program often enough to collect all the audio even though this application presents a very small CPU load. When we increase the priority of the audio program to real-time priority (using the task manager), the problem disappears. *Windows* schedules the program often enough to process all the audio with the usual latency. *Linux* does not suffer from this problem and schedules the program often enough even when the CPU is heavily loaded.

The conclusion is that to receive data over UDP, the soundcard would need to be able to cope with gaps on the order of 150ms when *Windows* sends data and around 50ms when *Linux* sends the data, and with the packet bursts at the end of such periods. Current microcontrollers cannot do this.

Audio Transport over TCP

To address this problem, I designed a TCP-based[10] transport strategy. TCP is a reliable protocol. The receiver sends acknowledgements to the transmitter. Packets that are not acknowledged are retransmitted until they do. To avoid wasting bandwidth on packets that the receiver cannot accept due to its limited buffer space, the receiver informs the transmitter how much data it is willing to receive. This amount is called the *window*. TCP contains numerous other features, mostly to avoid network congestion. TCP is widely used on the Internet. For example, Web pages are delivered using TCP.

I initially tried a straightforward strategy for receiving data from the PC. The microcontroller allocated a buffer for incoming audio and announced the size of this buffer as the initial window. As incoming packets fill the buffer, the window shrinks. As the microcontroller uses up data from the buffer to send to the CODEC (thereby making space available for more data from the PC), the window grows. The microcontroller program updated the window every millisecond.

This simple strategy worked as long as no packets were lost. When a packet from the PC was lost, the loss triggered a weird behavior in *Windows XP* that caused a significant amount of audio to be lost. A TCP sender can

sense that a single packet was lost, because subsequent packets that are not lost trigger the receiver to send acknowledgments. These acknowledgements do not acknowledge the lost packet, of course, but the packet before. The repeated acknowledgements of the same packet allow the sender to conclude that one packet was lost, and it retransmits it. For some reason, *Windows XP* retransmitted only part of this lost packet (740 bytes out of 1460), not all of it. The microcontroller acknowledged what it received, but *Windows* apparently was waiting for an acknowledgement of the entire packet. At this point *Windows* stopped sending for about 200ms. After that period, it retransmitted the entire missing packet and things got back to normal. But by this time, audio has been lost, producing a gap in the analog output of the sound card. This sequence of events happened every few seconds; it was not a rare event. I discovered why the audio was lost using *wireshark*, a free TCP tracing program.

The moral here is not that *Windows XP* is defective (it eventually did transmit all the data, and there might be an obscure but good reason for the way it behaved), or that TCP is wrong for this application. The moral is that TCP implementations include many complex behaviors that fast computers with a lot of memory can easily cope with but that microcontrollers cannot always cope with.

I decided to use TCP in a simpler way that would not excite any complex TCP behavior. The new strategy also used a large circular buffer to store incoming audio, but it only advertises to the PC a window of one packet (1460 bytes). This allows the PC to send only one packet at a time. When the packet is received, the microcontroller copies its content to its audio buffer. Every millisecond, the microcontroller examines the amount of free space in this buffer. If it is greater than 1460 bytes, it opens the TCP window to one packet, allowing the PC to send another packet. This strategy results in one of two behaviors. When the buffer is fairly empty, say initially or after a period in which the PC did not send data, the microcontroller acknowledges every packet immediately, opening the window back to one packet. This allows the PC to send another packet resulting in a fast packet-acknowledgement ping-pong like behavior. When the buffer is nearly full, which is the way it should be most of the time, the microcontroller receives a packet, acknowledges it, but keeps the window closed. The microcontroller checks the state of the buffer every millisecond; once enough space to receive another packet becomes available, the microcontroller opens the window, allowing the PC to send another packet. At 192kSPS and 24 bits, this happens on average every 1.26ms. At lower rates and resolutions, this happens less frequently. In both behaviors, the microcontroller essentially throttles the PC, allowing it to send one packet at a time. This rules out complex TCP behaviors that occur when there are multiple outstanding unacknowledged packets.

Sending baseband samples from the sound card is easier. The microcontroller reads baseband samples from the CODEC into a circular 2-packet buffer. Every millisecond, the microcontroller checks whether one of these packets is full. If it is, it sends it to the TCP driver. I configured TCP with a send buffer of about 10 packets (16KB), so no data is lost if the PC acknowledges packets rather slowly. Normally the PC acknowledges packets immediately.

Figure 3 shows a simplified diagram of this microcontroller software architecture. TCP packets are received by the Ethernet device driver (an interrupt service routine), which passes them to *lwIP*, which in turn passes them to the sound-card's TCP receive handler function. This function places the data in a large circular buffer (30KB). The data is extracted from this buffer by the CODEC's interrupt service routine. If the buffer is empty, the CODEC's interrupt routine sends zero samples to the CODEC, but this should never happen when the sound card functions normally. The same interrupt service routine also removes samples from the CODEC's input queue and stores them in a separate two-packet circular buffer. Once a millisecond, a periodic function is invoked by *lwIP*. It checks if more than a full packet has been extracted from the large buffer. If so, it updates *lwIP*'s window to allow the PC to send another packet. It also checks if there is a full packet in the small circular buffer. If so, it sends the data to *lwIP*'s TCP send buffer, which is a circular buffer maintained by *lwIP* itself. I configured it to 16KB. If *lwIP* needs to retransmit a packet, it retransmits it from this buffer.

Results

The TCP mechanism works well without dropping audio up to at least 192kSPS. The CODEC only supports 96kSPS and lower, so to test at 192kSPS, I simply replicated each sample and verified that no data is lost.

I tested the sound card by sending audio to it from the PC's microphone and by receiving PC audio that the sound card digitized from the analog output of an iPod music player. I also tested the card with complex baseband samples from a 14MHz radio front end (a *Softrock Lite*). The baseband samples were processed by an SDR application (*Rocky*) that demodulated radio signals.

I tested the system using three different Ethernet interconnects and using the Dell D820 running *Windows*. One was simply an Ethernet cable connecting the sound card and the PC. In this configuration, both the PC and the sound card used a technique called *Auto IP* to obtain IP addresses. This interconnect provides the highest possible performance and lowest possible latency.
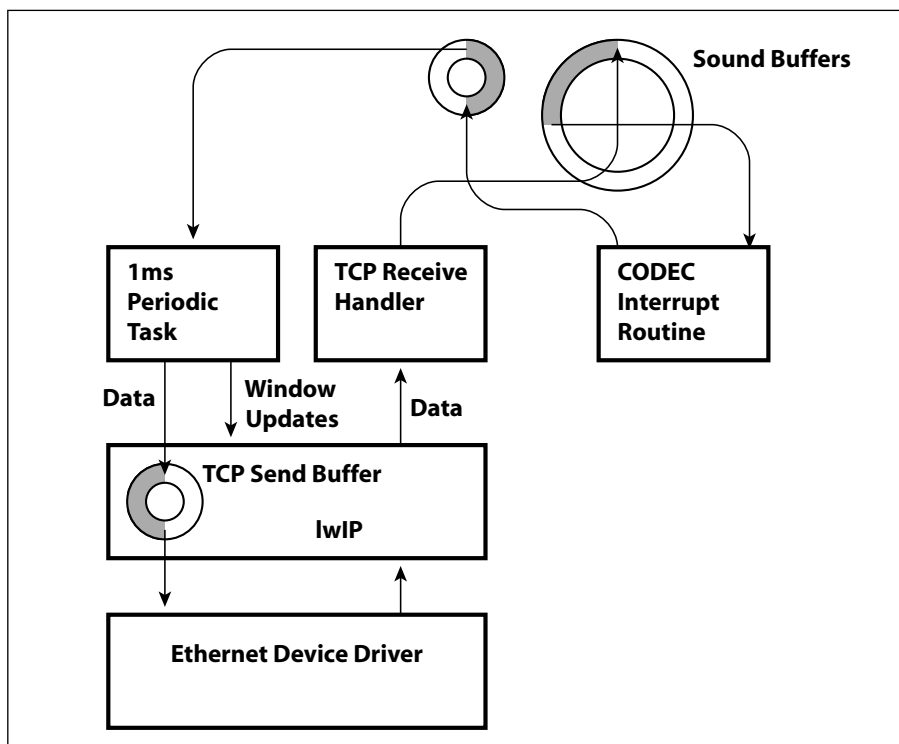

Figure 3 – Software architecture block diagram

In the second setup, the PC and sound card were both connected to an Ethernet switch. I tried both a D-Link switch designed for home use (more accurately, an ADSL modem with a built-in wired and wireless Ethernet switch, a DSL-2650U) and a large Cisco switch. In both of these setups, the sound card performed well without dropping any audio up to 192kSPS.

A third setup did not support high sampling rates. In this setup, the sound card was connected to the home switch through an Ethernet cable, but the PC was connected to the switch through a WiFi connection (802.11g, running at strong signal strength and 54Mb/s). In this setup, performance was good without dropped audio up to 48kSPS, but at 92kSPS and higher the PC-to-soundcard direction dropped a lot of audio. I eventually traced this to round-trip latency of about 2ms associated with the WiFi connection. That is, about 2ms pass between the time the PC sent a packet of data and the time the PC receives an acknowledgement with a window update allowing it to send another packet. This latency restricts the effective bandwidth between the PC and the sound card. Achieving high bandwidth in high-latency channels is exactly the reason that TCP normally uses large windows rather than the single-packet window that is used here; but as explained above, large windows, a small microcontroller, and high data rates do not work well together.

UDP Enumeration and Control

The sound card uses TCP for baseband (audio), but it uses UDP to rendezvous with the PC application and for command and control functions. I designed this UDP protocol earlier as a mechanism for PC programs to control and communicate with various technical gadgets. The protocol works not only with the sound card, but also with another microcontroller board, where the Ethernet frames are transported through a USB interconnect. The Ethernet-over-USB firmware supports both the proprietary *Windows* Ethernet-over-USB protocol, called RNDIS, and with the USB standard protocol, called CDC-ECM, which is supported by *Linux* (and by recent versions of *Windows*). In both cases, the drivers are built into the operating system.

When the sound card boots, it tries to obtain an IP address using the DHCP protocol. This succeeds in virtually any Ethernet environment except a direct cable connection. If the DHCP protocol fails, the card uses an auto-IP address. This behavior is part of *lwIP*. Now the sound card sends a broadcast UDP packet once a second announcing what kind of hardware it is. Since the packet is sent to a broadcast address, the sound card does

not need to know the IP address of the PC that will control it.

The PC application listens for such packets. When it receives one, it sends back a point-to-point UDP packet acknowledging the broadcast. Once the soundcard receives this packet, it knows the IP address of the PC. It starts listening for an incoming TCP connection request from the PC.

The UDP connection remains useful for sending commands to the sound card and for receiving status information from it. In this protocol, commands are sent in both directions. The sender expects an acknowledgement for every command. A command that is not acknowledged is retransmitted until it is.

The UDP channel continues to send heartbeat messages in both directions once a second, unless there are other commands to send. If either side fails to receive these messages for a few seconds, it assumes that the other side has disconnected. In this event, the PC application reflects this on the user interface, and the sound card goes back to sending periodic broadcast messages.

Summary

When sending real-time data over UDP from a general-purpose operating system like *Windows* or *Linux*, scheduling policies cause data to be sent in bursts with fairly long delays between bursts. Microcontrollers with limited amount of buffering and processing power cannot handle this burst property. TCP transport avoids the burst issue because the microcontroller can throttle the incoming data by opening only single-packet windows. In principle, opening larger windows should also work, but it fails in practice, at least when sending from *Windows XP*, because single packet loss triggers complex TCP behaviors that sometimes generate long latencies.

The use of single-packet windows links

the maximum throughput to the round-trip latency. In direct cable connection and connection through a wired Ethernet switch, the prototype supported full-duplex two-channel 192kSPS rate. But when communicating through WiFi, the round trip latency rose and the throughput dropped, allowing only up to 48kSPS. This would also limit throughput when the IP connection goes through multiple switches and routers, which also increases the latency.

From the application viewpoint, real-time data transport over TCP (or UDP when possible) reduces the dependency of SDR applications on both sound-card device drivers and the audio system's API.

Appendix: Additional Technical Details

The schematic of the CODEC circuit is shown in Figure 5. The microphone-input and line-output portions of the circuit are not wired in my prototype. The circuit was not optimized for high dynamic range and low noise; it was intended only as a platform for testing the Ethernet-based protocol.

The microcontroller is configured to generate the 12.28MHz master clock for the CODEC. The same clock frequency is used for all CODEC sampling rates. The CODEC is configured as a clock master for both the DAC and the ADC, and the microcontroller is configured as a slave in both. The CODEC therefore generates the bit clock and the left-right clock signals. Configuring the microcontroller as an audio clock slave gets around a silicon bug in revision B of the microcontroller. I also configured the audio as left-justified (rather than the more standard I2S format) to get around a related bug.

The microcontroller cannot generate a master clock at exactly 12.28MHz; it uses a fractional clock divider to get close, but there is some discrepancy between the CODEC's
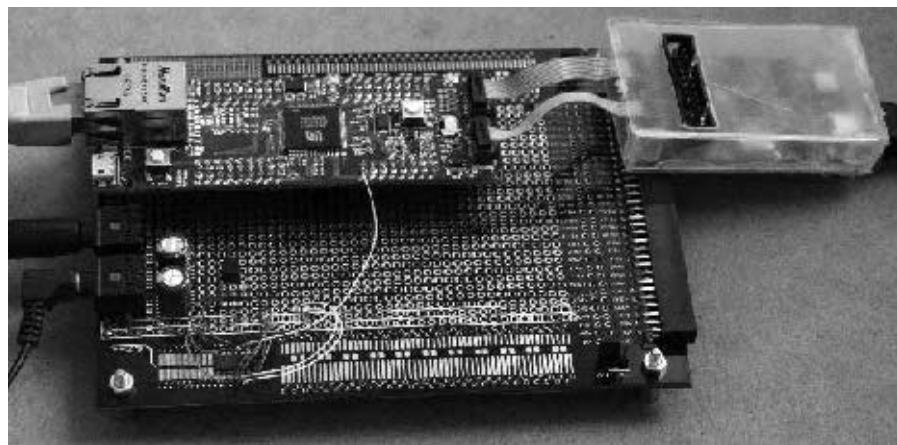

Figure 4 – Photo of the development system. The bottom board contains the CODEC parts. The board on the top left is the TI development system. The board in the plastic box is the debugging interface.

sampling rate and the rate at which the PC sends or consumes audio data. A better but more expensive solution would be to drive the CODEC with a crystal or a canned oscillator at exactly 12.28MHz.

Sivan Toledo is Professor of Computer Science at Tel-Aviv University. He holds BSc and MSc degrees from Tel-Aviv University and a PhD from the Massachusetts Institute of Technology, where he was also Visiting Associate Professor in 2007-2009. He was licensed in 1982. You can contact him at **stoledo@tau.ac.il**.

Notes

[1] Gerald Youngblood, AC5OG, *Software-defined radio for the masses*. QEX Jul/Aug 2002, Sep/Oct 2002, Nov/Dec 2002, and Jan/Feb 2003

[2] See **groups.yahoo.com/group/softrock40**

[3] The term SDR encompasses a wider range of radio architectures, including radios that do not use a personal computer but a special-purpose computer. In this article, however, I will use the term SDR to refer to radios that use a personal computer to do some of the signal processing

[4] **www.flex-radio.com**

[5] These receivers have a Web interface accessible from **wwwhome.cs.utwente.nl/~ptdeboer/ham/sdr**

[6] See, for example, Jim Ahlstrom, N2ADR, An all-digital SSB exciter for HF, QEX May/Jun 2008

[7] USB Implementers Forum. *USB Device Class Definition for Audio Devices*. Release 1.0, 1998, and Release 2.0, 2006

[8] See Jan Axelson, *USB Complete*, 4th edition, Lakeview Research, 2009, or John Hyde, *USB By Example*, 2nd edition, Intel Press, 2001

[9] UDP appears to be used by all or most Ethernet-based SDR platforms, such as the USRP2 (**www.ettus.com**), Jim Ahlstrom's radios, and Pieter-Tjerk de Boer's radios. The PC applications for these radios run on Linux, not Windows, which helps, as we'll see later in the article

[10] For good coverage of TCP, see W. Stevens, *TCP/IP Illustrated*, 3 volumes, Addison-Wesley, 1994
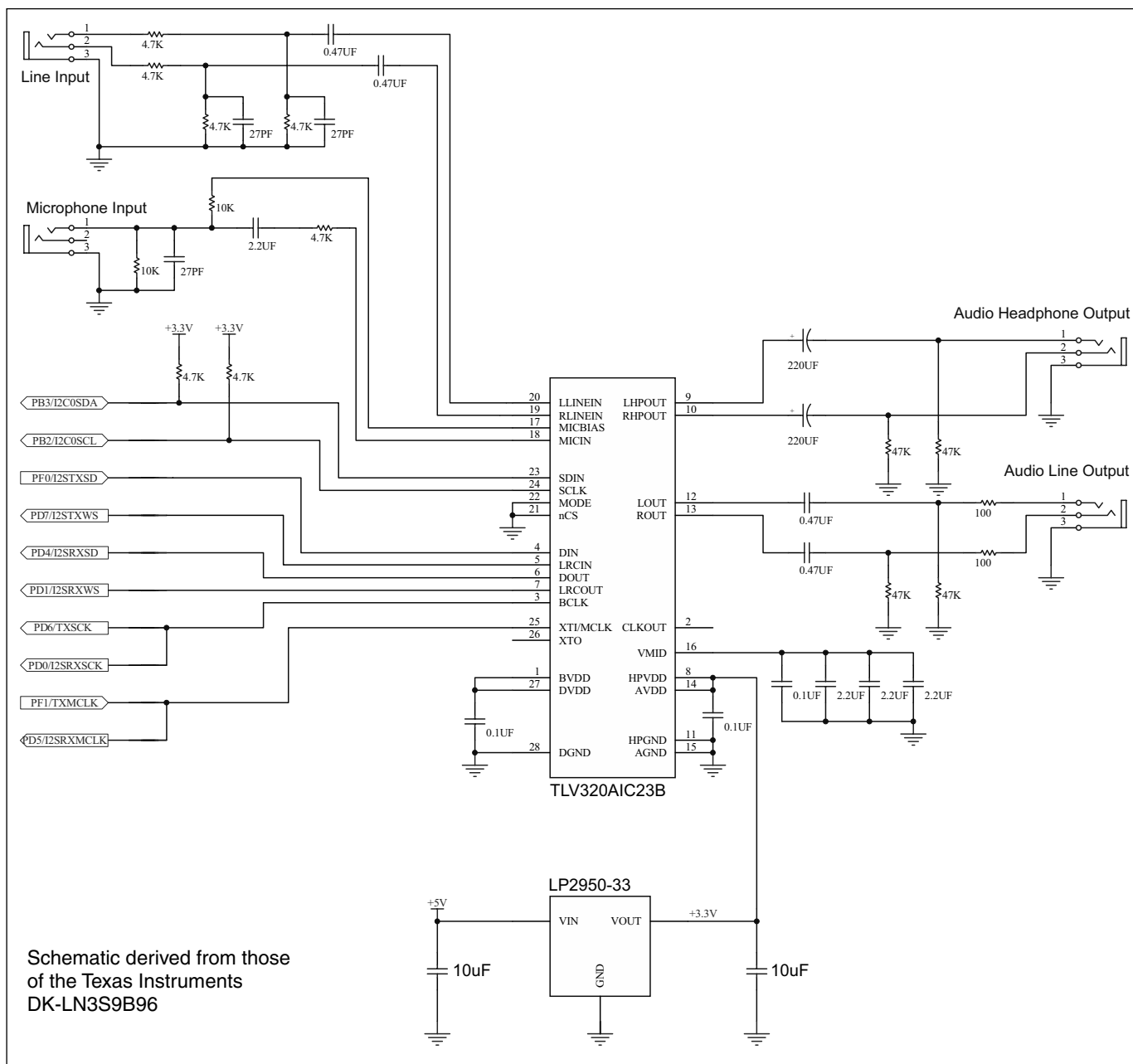
Schematic derived from those of the Texas Instruments DK-LN3S9B96

Figure 5 – Schematic of the development board and CODEC interface.

Oleg Sergin, DL2IPU

Winsen (Luhe), 21423, Germany, dl2ipu@googlemail.com

Simulating Tapped Coupled Inductors

Hands-on, step-by-step example for working with Micro-Cap and FastHenry2: the singe-coil Z-Match coupler.

The Evaluation Process

Some time ago, I decided to build an antenna tuner capable of full legal power. The variety of impedance matching circuits is huge. Each one has its advantages and disadvantages. I liked Z-Match which is an idea that has been used for a very long time.[1] Especially interesting for me were fixed coils – no roller and no taps to move. Finally, I found the description of a single coil Z-Match by Lloyd Butler, VK5BR.[2] Figure 1 shows a very neat one built by John H. Green, ZS1JHG.

First, I wanted to check if I could match my antenna with it. My doublet seems to be rather short for the 80m band. Can this coupler manage it? How sharp will the tuning be? Will it be a nice wide maximum or a hard-to-find peak? Also, what efficiency should I expect? And, will the dissipated power be enough to melt plastic parts holding the coil and solder joints? What is the expected voltage across capacitors? Knowing these parameters, I could decide if surplus parts I have on hand can do the job or if even more spacing between plates is required.

For me, there are at least three ways to get answers on these questions. The first one is to build and try. Of course, there is a certain risk of wasted time if it does not work as expected. The second is to calculate it. It is certainly useful to practice using Kirchhoff's current and voltage laws, but solving a system of six linear equations in complex numbers requires at least math software to do it

[1]Notes appear on page 28.
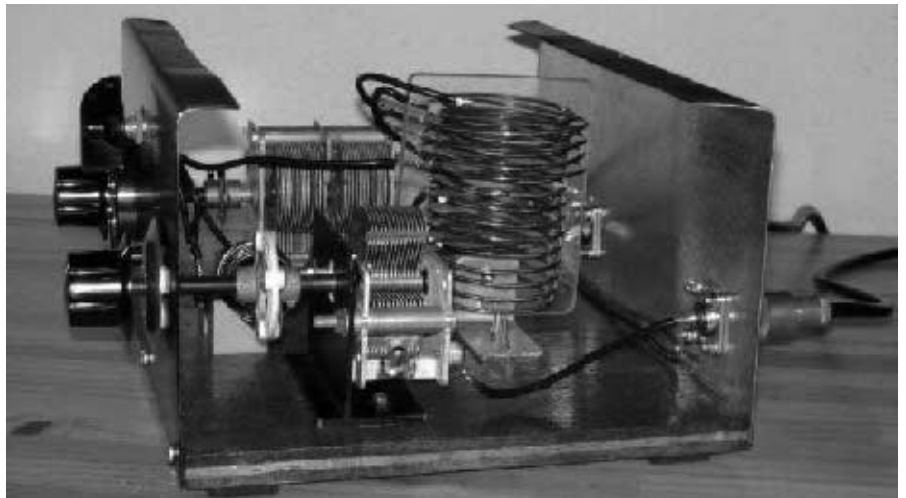
Figure 1 – Single-Coil Z-Match made by John H. Green, ZS1JHG

within a reasonable time. The third way is to get to results using an electrical circuit simulation program. Basically, it gives answers by solving circuit equations numerically. A key advantage is that I can keep myself focused on the circuit instead of bothering with the math. This way seems to be the most practical to me.

The good news is that there is a very sophisticated circuit simulation program available for free. It is the limited demo version of *Micro-Cap*.[3] On the next pages I would like to show you how it works, and hope to encourage you to try it for your needs. [Oleg used *Micro-Cap* version 9 for this article. Spectrum Software has released version 10 since Oleg wrote his

article. Version 10 is described as making only minor incremental changes, so these instructions should be applicable to either version. – Ed.]

We will stick to the example of the single coil Z-Match as shown in Figure 2. If you compare this circuit with the original one[2], you will notice that the optional coil in series with the antenna is missing here. This is done purposely to keep the example even simpler. Most of the concepts here apply to any electrical circuit simulation program and not only to the one described.

First, we'll find representations for transmitter output, antenna, and an air-core transformer with tapped primary winding. Here we will use *FastHenry2* (a neat wrap
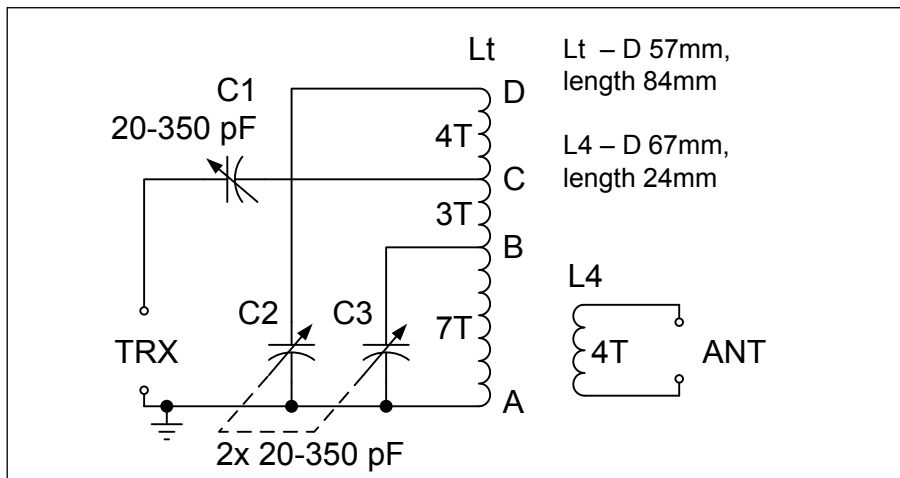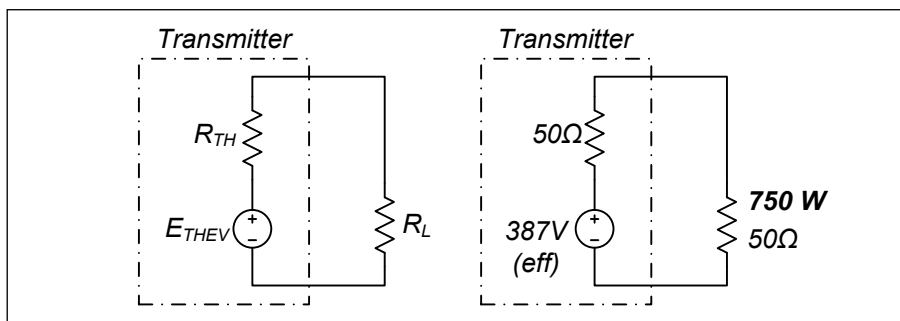
Figure 2 – Single-Coil Z-Match Schematic

Lt

Lt – D 57mm, length 84mm

L4 – D 67mm, length 24mm

C1 20-350 pF

D

4T

C

3T

B

7T

A

C2   C3

2x 20-350 pF

TRX

L4

4T   ANT

Figure 3 – Equivalent Circuit of Transmitter

Transmitter

R_{TH}

E_{THEV} R_L

Transmitter

50Ω

387V (eff)

750 W

50Ω

Figure 4 – Doublet Antenna

Z

Y

X

axis=5m

Figure 5 – Equivalent Circuit of Antenna

Antenna

La 1.75µH

Ra 5.19Ω

for MIT algorithms) to calculate inductances based on geometry. Then, we'll see how to enter the circuit in *Micro-Cap* editor and how to run the simulation. Finally, we will see how to extract and interpret the results.

Using *Micro-Cap*

Micro-Cap is an electronic circuit simulation program, inspired by the *SPICE* engine[4], which provides an interactive sketch and simulate environment for electronics engineers. You can download a demo version able to run on *Windows 2000, XP* or *Vista* with limited functionality. Circuit size is limited to 50 components and it runs anywhere from 0% to 300% slower than the professional version. It has a limited component library, and some of the advanced features are not available. Luckily, it is still more than enough for our task (and many others, too).

So, that's the very first step – fill-in and send the form on the Spectrum Software Web page[5], check your e-mail inbox for the download link, and then download and install the program. The installation procedure contains the usual steps. A lot of examples are delivered together with the program. You may want to have a look at them: simply choose

File → Open after program launch, or take a short features tour on the vendor's home page.

Circuit components

Before we are able to draw a circuit in the editor, we have to know equivalent circuits for transmitter output, antenna and, most interesting, the coil.

Let's start with easy one – transmitter output. See Figure 3. Assume it is a 750 W (legal limit for the extra-class license in Germany), 50Ω output solid state power amplifier. Maximum power P is delivered into the load, if load resistance R_L is the same as the equivalent resistance of voltage source $R_{TH} = 50Ω$. What is the corresponding equivalent voltage?

$$P = \left(\frac{E_{THEV}}{R_{TH} + R_L} \right)^2 \times R_L \qquad \text{[Eq 1]}$$

$$E_{THEV} = \sqrt{\frac{P}{R_L} \times (R_{TH} + R_L)^2} =$$

$$\sqrt{\frac{750W}{50Ω} \times (50Ω + 50Ω)^2} = 387V \text{ (eff.)}$$

$$\text{[Eq 2]}$$
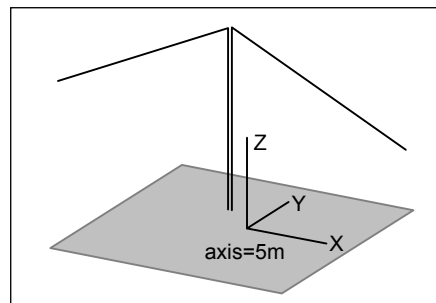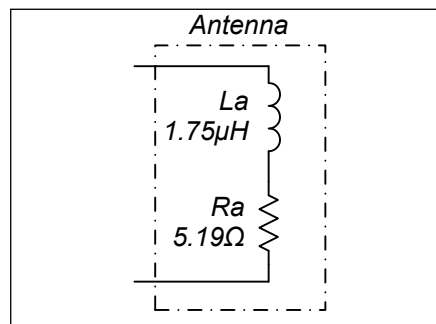
What should the antenna in *Micro-Cap* look like? My doublet's highest point is 11m above the poor ground as shown in Figure 4. Its 11.5m long shoulders are sloped so that low ends are 6.1m high. It is fed in the middle with 13.5m long 600Ω ladder line. As I am concerned that this antenna is a bit too short and too low for 80m and may be challenging for the tuner, let us do the simulation for this band. According to *4nec2* simulation, the impedance at 3.61 MHz is

$$Z_a = 5.19 + 39.6j \; Ω \qquad \text{[Eq 3]}$$

The imaginary part is positive (inductive), and the antenna can be represented as a series connection of a resistor

$$R_a = \text{Re}(Z_a) = 5.19 \; Ω \qquad \text{[Eq 4]}$$

and inductor

$$L_a = \frac{\text{Im}(Z_a)}{2\pi f} = \frac{39.6Ω}{2\pi \times 3.61 \times 10^6 \, Hz} = 1.75\mu H$$

$$\text{[Eq 5]}$$

Of course, this representation (Figure 5) is only correct on this single frequency. Two decimal digits are not an attempt at a very accurate simulation, but copying them as they are from program to program makes troubleshooting easier.

COILS

To represent the air-core tapped-primary

transformer by lumped components in *Micro-Cap*, we need every self and mutual inductance. And, to estimate losses, we need loss resistances. This is shown in Figure 6. Lloyd, VK5BR, used co-axially located coils of 14AWG copper wire (see Table 1). The lower end of *L4* is aligned with the earth end of *Lt*, but *L4* is shifted up half a turn compared to *Lt* to minimize parasitic capacitance between them.

An equivalent circuit for this air-core transformer is shown in Figure 6. Tapped *Lt* consists of *L1*, *L2* and *L3* coupled with each other. *L4* is coupled with all of them. Resistors represent losses in the inductors.

The very next task is to find four self and six mutual inductances from Figure 6. For this purpose, yet another calculation tool can be used. It was developed by the Computational Prototyping Group at MIT[6] and is called *FastHenry*. There is a *Windows* version, *FastHenry2*, available as part of a package called *FastFieldSolvers*.[7] I noticed nothing special about the installation.

Calculating self and mutual inductances with *FastHenry2*

You may want to jump over this part and
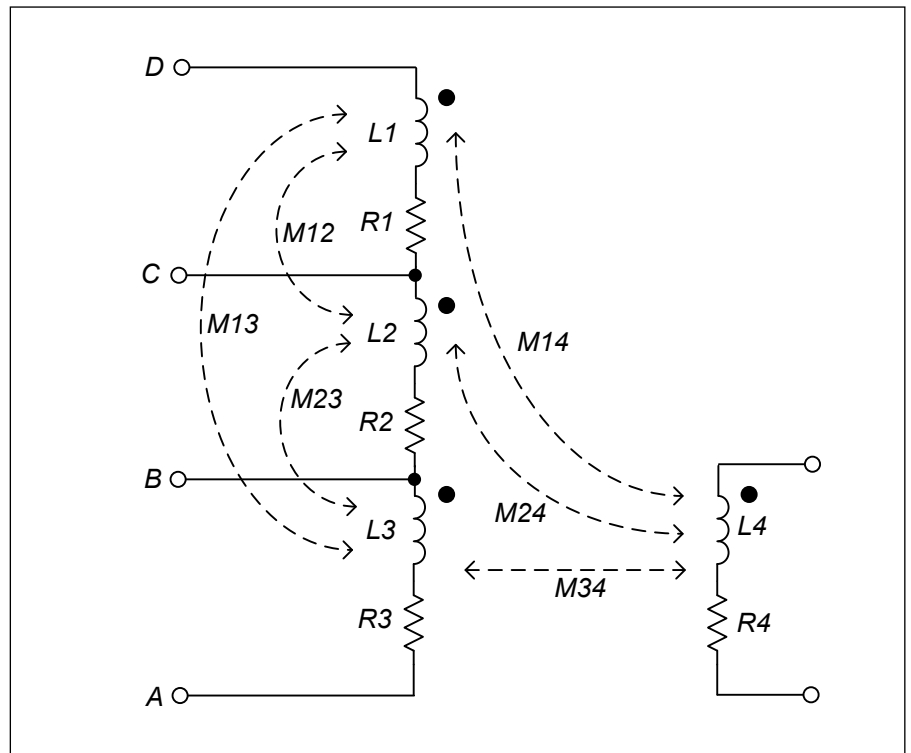
Figure 6 – Equivalent Circuit of Air-Core Tapped Transformer

Figure 7 – Preparing Coils Description in Excel

Table 1

Coil	Diameter, mm	Turns	Step, mm	Taps
Lt – transmitter side	57	14	6	7, 10
L4 – antenna side	67	4	6	--

Table 2

Coil	Start node	End Node	Segments
L3 (A-B)	0	700	E0 – E699
L2 (B-C)	700	1000	E700- E999
L1 (C-D)	1000	1400	E1000-E1399
L4	1401	1801	E1400-E1800

Figure 8 – Verifying Coils Geometry

use the *L* and *M* values provided at the end if you want just to see how *Micro-Cap* works. Personally, I found it very interesting to calculate tapped inductor and coupled inductors numerically and would like to spend a page on it. A script by Claudio, IN3OTD, gave me the idea.[8]

We begin with describing the geometry just as for antenna simulation. Wires are split in segments which are defined by their end nodes. Segments may be split, in turn, into filaments to analyze the impact of skin and proximity effects, but we are not going to over-do it.

Then, ports need to be defined. Those are ends of partial inductors, four total in our case. Given geometry, frequency and copper admittance *FastHenry2* can calculate a matrix of impedances, from which self and mutual inductances can be extracted.

Figure 7 shows a Microsoft *Excel* work-book file used to generate the *FastHenry2* input file. The input sheet contains lengths, diameters, etc, to keep them all in one place. *FastHenry2* can work with straight segments only, so round turns are represented by poly-lines with 100 segments per turn. *Lt*, with 14 turns, is described by 1400 segments and tapped after segment 700 and segment 1000. *L4*, with four turns, is described by 400 segments.

Let's assume that coils are wound along the Z axis in 3D space. Sheets *nodes1* and *nodes2* define nodes of *Lt* and *L4*. Each node is defined by *X*, *Y* and *Z* coordinates. It is nothing but simple trigonometry: *X* and *Y* are defined by cosine and sine of the node's angle in the turn, ranging from 0 to 360°; *Z* is defined by fractional turn number.

Now that we have nodes, we can define segments. For example, segment E0 connects nodes N0 and N1, E1 connects N1 and N2, and so on. *FastHenry2* works with rectangular section segments, requiring width and height for each one. To make it easy, let's assume width = height = wire diameter. This simplification has very little influence on calculated inductance, and we are not using *FastHenry2* to calculate losses.

To check if the geometry was generated correctly, top view (to XY plane, along Z axis) and side view (to ZY plane, along X axis) can be plotted in *Excel*. Figure 8 shows the results. The top view shows two circles: the smaller one is Lt, the larger one is *L4*. The side view looks reasonable, too.

Table 2 shows the logical relationship between inductors and segments/nodes. Using the relationship, we can describe ports for *FastHenry2*. A port is a pair of nodes or inductor ends. There are four inductors (refer to Figure 6, again) and therefore four ports.
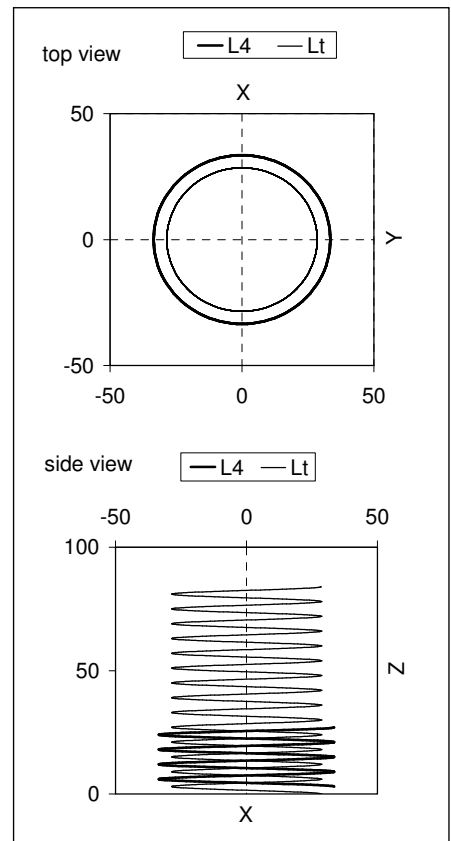
We need to put the *Excel* generated nodes and segments together along with ports in one text file and put in some additional values like frequency. Here is a small trick: if the frequency is $f = 1/2\pi$, the calculated reactance will be numerically equal to the inductance. Figure 9 shows a shortened *FastHenry2* input file. Both *Excel* and text files are available for download.[9]

It's finally time to start *FastHenry2*. Click *File → Open* and then choose the input file, as shown in Figure 10. Then click Run and wait for results. They will be shown in the program window, but a file "Zc.mat" will also be generated. It contains an impedance matrix, 4x4 in our case since we defined 4 ports. Table 3 shows the content of "Zc.mat".

Interpreting results from *FastHenry2*

We need to understand what this matrix contains to use these results. Both lines and columns each represent an inductor. *FastHenry2* is not aware of component names and refers to them by node numbers. Referring to Table 2 to convert node numbers back to *L1* through *L4*, we see that lines 1 to 4 represent *L3*, *L2*, *L1* and *L4* in the order listed.

The four elements on the left to right diagonal of the matrix in Table 3 (elements 11, 22, 33, 44) each represent an inductor (values in bold). Real parts of complex numbers are ohmic resistances. Remember, we did not split segments into filaments; this is why the calculation does not reflect skin and proximity effects. Imaginary parts are posi-

tive (inductive) reactances. For example, the matrix element in fourth row, fourth column has an imaginary part of 1.37×10^{-6}. Since we chose $f = 1/2\pi$, inductance is numerically equal to reactance so $L4 = 1.37 \ \mu H$.

Matrix elements on the right to left diagonals describe mutual effects. The values are approximately equal on both sides (elements 12 and 21, 32 and 23, etc). Imaginary parts reflect mutual inductances between corresponding two inductors. For example, the matrix element in first row (*L3*), second column (*L2*) has an imaginary part of $+3.72 \times 10^{-6}$. This means that mutual inductance between *L3* and *L2* is $M23 = 0.372 \ \mu H$. The mutual values are duplicated above and below the main left to right diagonal through the array.

For *Micro-Cap*, we'll need coupling factors instead of mutual inductances:

$$k_{xy} = M_{xy}\big/\sqrt{L_x L_y} \ ; 0 \le k_{xy} \le 1 \qquad \text{[Eq 6]}$$

Table 4 shows inductances and Table 5 shows coupling factors for all four inductors. As you see, there are six coupling factors all considerably greater than zero since all inductors are located close together. Now we can check the values to see if they are reasonable. The best coupling is between *L3* and *L4* with $k_{34} = 0.687$. Indeed, *L3* and *L4* are located closer to each other than others (see Figure 1). The weakest coupling is between *L1* and *L4*

Table 3
Output data from FastHenry2
Row 4: n1401 to n1801
Row 3: n1000 to n1400
Row 2: n700 to n1000
Row 1: n0 to n700
Impedance matrix for frequency = 0.159155 4 x 4

0.008137 +2.37e-006j	1.64e-018 +3.72e-007j	-7.68e-019 +1.76e-007j	-5.00e-020 +1.24e-006j
7.05e-019 +3.71e-007j	**0.00348 +7.00e-007j**	1.03e-019 +3.00e-007j	-1.14e-018 +1.57e-007j
-4.91e-019 +1.76e-007j	-1.10e-018 +2.99e-007j	**0.00465 +1.07e-006j**	1.33e-018 +8.95e-008j
1.29e-019 +1.24e-006j	-1.84-018 +1.56e-007j	9.9e-019 +8.96e-008j	**0.00546 +1.37e-006j**

with $k_{14} = 0.074$. This sounds reasonable, too.

What are those real parts of non-diagonal matrix elements? A mutual resistance term means that, in response to a variation in the current flowing in one segment, there should be a variation in the voltage across one other segment. This is a direct consequence of the proximity effect, and is not considered here. We also don't consider the distributed capacitance of the coils (see *ARRL Handbook*[10], 5.3.3 "Inductors at Radio Frequencies").

Skin effect – coil resistances

To estimate ohmic resistances, let's start with calculating wire lengths and DC resistances. Given coil diameter *D*, number of turns *n* and coil length *l*, wire length is

$$l_w = \sqrt{(n\pi D)^2 + l^2} \qquad \text{[Eq 7]}$$

Calculated wire lengths are filled into the corresponding line of Table 6.

According to Component Data and References[10], 14 AWG wire has dc resistance of 2.5240 Ω per 1000 ft. Using this value, R_{DC} for every coil is filled into Table 6. To estimate ac resistance on given frequency, we calculate the skin effect cut off frequency where a wire (64 mils diameter for 14AWG) will begin to show appreciable skin effect (5.3.4 Skin Effect[10]):

$$f = \frac{124 \cdot MHz}{(d/mils)^2} = 0.030 MHz \qquad \text{[Eq 8]}$$

Above this frequency, wire resistance increases roughly 3.2 times for every decade. Antenna tuner simulation frequency is 3.61 MHz, so this is 2.1 decades above the skin effect cutoff frequency.

$$R_{AC} = R_{DC} \times 3.2^{2.1} = R_{DC} \times 11.5 \qquad \text{[Eq 9]}$$

These results are in Table 6, too. Finally, we can calculate quality factors $Q = X / R_{AC}$. Q of 300 to 400 is a realistic value for an air-core inductor wound with a thick wire. Now we have all values to begin simulation of the whole antenna tuner.

Entering circuit in *Micro-Cap* editor

Start *Micro-Cap*, close "Tip of the day", and you are in the editor. Let's start with

```
* FastHenry2 input file,
* containing coils definitions

.Units mm
.default sigma=5.8e4

* Nodes Coil1
N0  x=28.5 y=0 z=0
N1  x=28.44 y=1.79 z=0.06
. . .
N700 x=28.5 y=0 z=42
. . .
N1000 x=28.5 y=0 z=60
. . .
N1400 x=28.5 y=0 z=84


* Segments Coil1
E0 N0 N1 w=1.63 h=1.63
E1 N1 N2 w=1.63 h=1.63
. . .
E1399 N1399 N1400 w=1.63 h=1.63


* Nodes Coil2
N1401 x=33.5 y=0 z=3
. . .
N1801 x=33.5 y=0 z=27


* Segments Coil2
E1400 N1400 N1401 w=1.63 h=1.63
. . .
E1800 N1800 N1801 w=1.63 h=1.63


* define ports
.external N0 N700
.external N700 N1000
.external N1000 N1400
.external N1401 N1801

* frequency range
.freq fmin=0.15915 fmax=0.15915 ndec=1

* The end
.end
```

Figure 9 – FastHenry2 Input File

transmitter output which is the voltage source. As shown on Figure 12, select voltage source among the components on the tool bar. The mouse pointer will turn into the corresponding electrical symbol when moved above the circuit sheet. A left mouse click will place the component and open the properties window. A text list in the middle shows the properties. The very first one is part name: "PART=*V1*". To change it, click on it in the list, then the value "*V1*" appears in the entry filed "Value" at the top of the window. Replace "*V1*" with "*Ethev*" or whatever you like. Now, select "Sin" among the tabs at the bottom of the window as it should be a sine voltage source. Enter the frequency by replacing "1meg" with "3.61meg" in the "F0" field. Enter the ac magnitude from Equation 2. Since this is an effective value, all calculated voltages and currents will be effective values, too. This is the value used during ac analysis. Transient mode analysis requires an amplitude value. For a sine source it is magnitude $\times \sqrt{2} = 547$ V. Enter this value into the VA field, just to stay consistent. Click OK.

If you want to rotate a component, press Ctrl-E to switch to select mode. Click on the component to select it and its background will become light-green. Then press Ctrl-R to rotate component 90°. If you find that component names and values are not placed perfectly, you can drag them with the mouse pointer to where you want them to be.

Now it's a good time to put ground into the circuit. Connect it to the "-" pole of the voltage source. This is the zero reference point for all node voltages that *Micro-Cap* calculates during simulation. The simulation cannot proceed without a ground symbol, and you cannot save the file.

Next, put a 50Ω resistor *Rth* in series with voltage source. Select the resistor tool (see Figure 11) from the component tool bar, click on the sheet next to voltage source and enter values in the properties window opened as shown on Figure 13, then click OK:

PART=*Rth*
RESISTANCE=50

You may want to consult Figure 18 during component placing and values assignment. Proceed with *R1* through *R4* and *Ra*. Values for *R1* through *R4* are in the R_{AC} line in Table 6, for R_a see the part "Circuit components – antenna". Dragging and dropping a component while holding Ctrl key creates a copy, like in many other programs.

Then, place capacitors *C1* through *C3*. There is not a special component for a variable capacitor in *Micro-Cap* since all components are variable from the beginning. Constants are normally used for component values. Table 7 shows capacitor values. *Micro-Cap* has a way to handle that *C2* and C3 are two halves of a double variable capaci-

Table 4
Inductors and Mutual Inductances (Henries)

	L3 (A-B)	L2 (B-C)	L1 (C-D)	L4
L3 (A-B)	2.37 e-6	3.72 e-7	1.76 e-7	1.24 e-6
L2 (B-C)	--	7.00 e-7	3.00 e-7	1.57 e-7
L1 (C-D)	--	--	1.07 e-6	8.95 e-8
L4	--	--	--	1.37 e-6

Table 5
Coupling Factors for Inductors

	L3 (A-B)	L2 (B-C)	L1 (C-D)
L2 (B-C)	0.288	--	--
L1 (C-D)	0.110	0.346	--
L4	0.687	0.160	0.074

Table 6
Coil Parameters

Coil	L1	L2	L3	L4
Coil Diameter (mm)	57	57	57	67
Turns	4	3	7	4
Coil Length (mm)	24	18	42	24
Wire Length (mm)	717	538	1254	842
Wire Length (ft)	2.35	1.76	4.11	2.76
R_{dc} (Ω)	0.0059	0.0045	0.0104	0.0070
R_{ac} (Ω)	0.068	0.051	0.119	0.080
Equivalent Resistance	R1	R2	R3	R4
Reactance (Ω)	24.3	15.8	53.7	30.9
Q	356	309	450	385

Table 7
Capacitor Values for 3.61 MHz – 80m Doublet

Part	C1	C2	C3
Capacitance	97.7pF	262.5pF	C(C2)

Table 8
Micro-Cap **Expressions**

| C1 Voltage	U_C1 = [MAG(V(C1))] "**MAG**nitude of **V**olatge across *C1*"
C2 Voltage	U_C2 = [MAG(V(C2))] "**MAG**nitude of **V**olatge across *C2*"
C3 Voltage	U_C3 = [MAG(V(C3))] "**MAG**nitude of **V**olatge across *C3*"
Efficiency	Efficiency% = [100 * MAG(PD(RA)) / (MAG(PD(RA)) + MAG(PD(*R1*)) + MAG(PD(R2))+ MAG(PD(R3)) + MAG(PD(*R4*)))] "100x **MAG**nitude_of_**P**ower**D**issipated_in_*Ra* / (**MAG**nitude_of_**P**ower**D**issipated_in_*Ra* + . . ."
Ant + Tuner Impedance	Z = [V(2) / I(2,4)] "**V**olatge_at_node_**2** / Current_from_node_**2**_to_**4**"
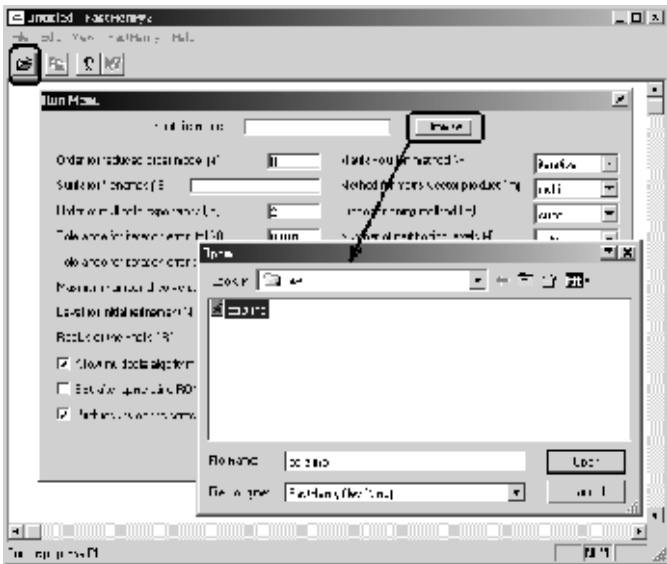
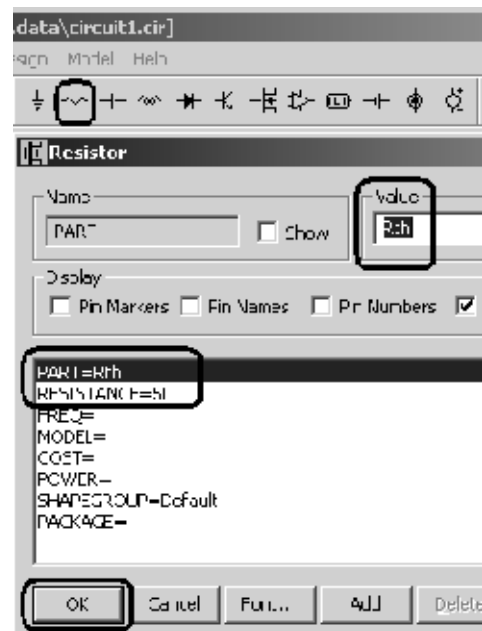Figure 10 – Opening Input File in *FastHenry2*

Figure 13 – *Micro-Cap*: Adding R_{th}
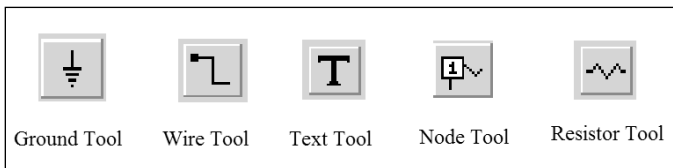

Figure 11 – Tool Icons used in *Micro-Cap*

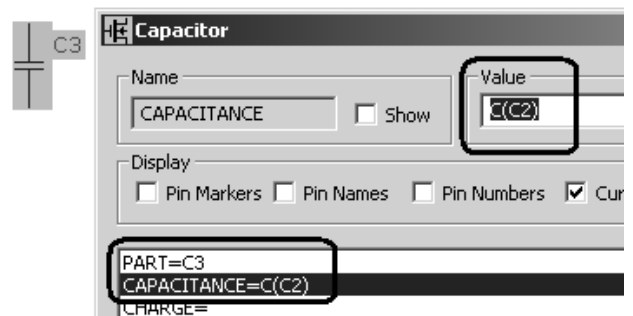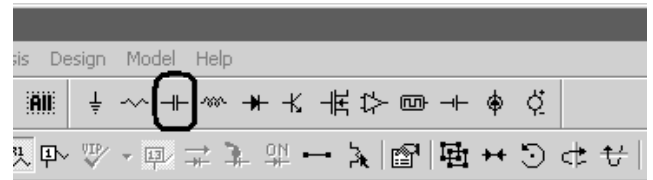Ground Tool   Wire Tool   Text Tool   Node Tool   Resistor Tool
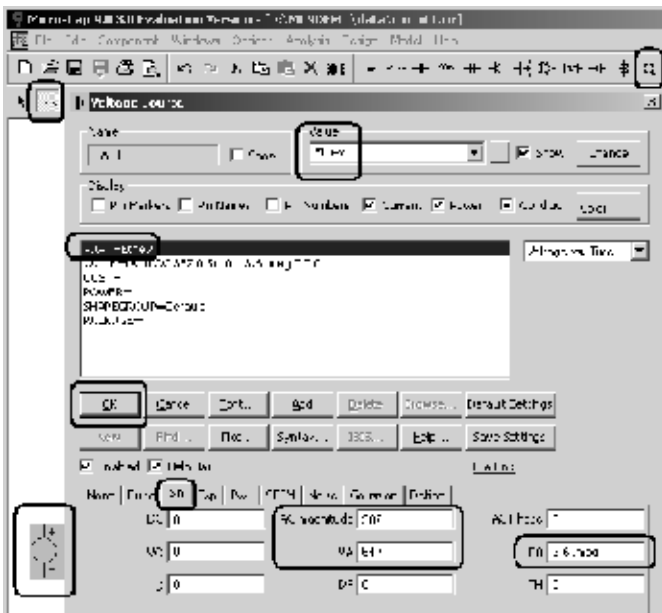

Figure 14 – *Micro-Cap*: Adding *C3*
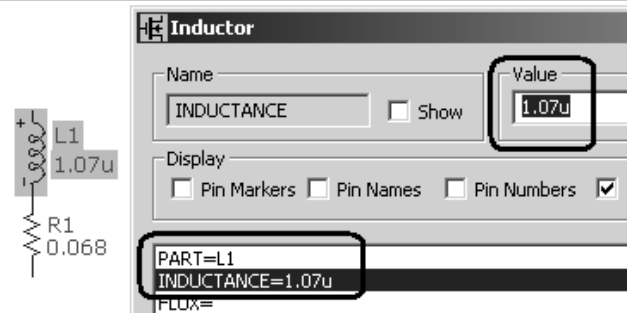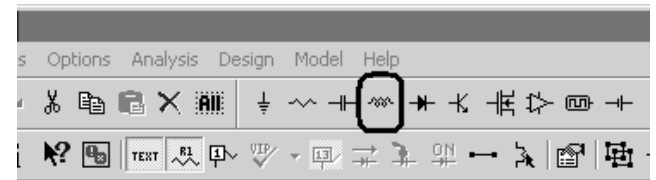

Figure 12 – *Micro-Cap*: Adding Voltage Source

Figure 15 – *Micro-Cap*: Adding *L1*

tor. This is shown on Figure 14. The value for the **CAPACITANCE** property of *C3* is C(*C2*), which means: capacity of *C2*.

It is time to put inductors in the schematic. Pay attention to polarity, indicated by "+" and "-" at inductor ends. Fields of *L1*, *L2* and *L3* must be aiding (not opposing), so let all "+" be on the same side (the top) as in Figure 18. Enter self inductance values of *L1* through *L4* from Table 4 (diagonal), and *La* from Equation 5. Instead of using engineering format (1.07e-06) you may prefer to use "u" for 10^{-6} (see Figure 15).

Next, let's wire components together. Use the wire tool, then, use the text tool (see Figure 11 for the icons) to put node names A to D as on Figure 6, and also "PA OUT" and "ANT" labels. The circuit is almost ready.

We are missing only the most important part, the mutual inductances. Figure 16 shows how to select it from the component list. Let us place one in the circuit and assume it represents the coupling between *L1* and *L2*. From Table 5 you find $k_{12} = 0.346$. Figure 17 shows properties to change:

click on **PART** property, unselect "show"

click on **INDUCTORS** property and enter "*L1 L2*" as value; select "show"

click on **COUPLING**, enter 0.346 as value; select "show"

leave **MODEL** blank, then linear (as air-core is) model is used

Voila, *L1* and *L2* are now coupled with the coupling factor as written above. Both linked inductors and factor are shown next to component for better overview. Repeat it five more times to have all couplings entered according to Table 5. The finished circuit looks like that shown in Figure 18.

Simulation

Among the available analysis modes, we need frequency-domain calculation (although on a single frequency), for which results are updated immediately with user edits. That is called "Dynamic AC Analysis". Select it from Analysis menu, then Dynamic AC Limits dialog will open, as shown on Figure 19. Select node voltages, currents and powers by pushing corresponding buttons down. Input the frequency 3.61E6. It is not a list, but a single one in our case. For complex value display, select magnitude as the first value and none as the second one. Press ok, and the simulation starts.

Results are there immediately, and the display seems to be much more crowded than before. After some drag and drop work, the circuit looks like it does in Figure 20. Now, let's make variable capacitors easily variable by giving them sliders. Proceed as shown on Figure 21, and you will get two nice sliders as in Figure 22. Initial values were not changed. Please leave them so for a while (as in Table 7). These values represent the best tun-
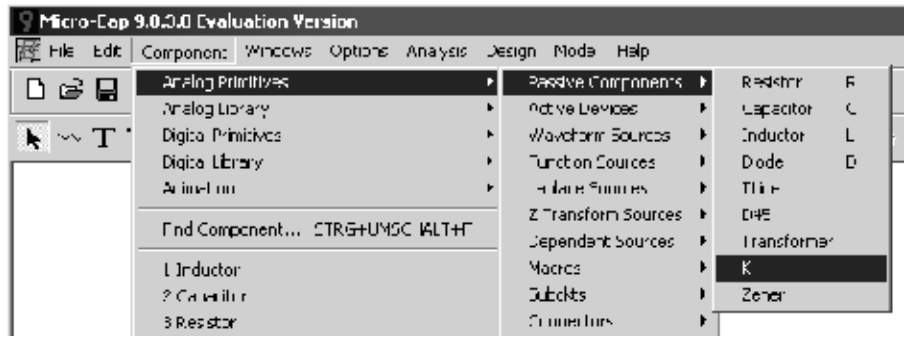
Figure 16 – *Micro-Cap*: **Adding Inductor Coupling Factors**
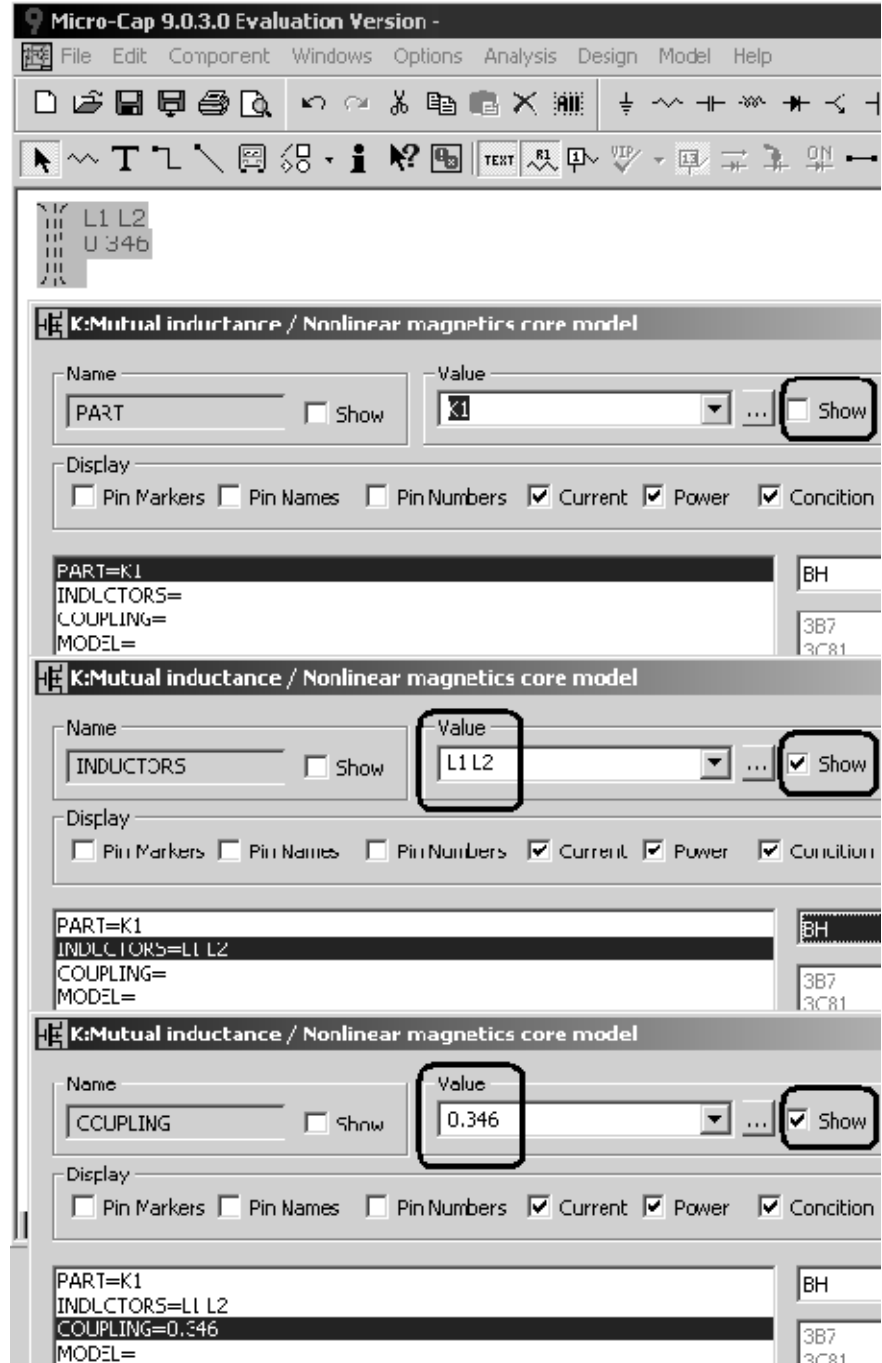
Figure 17 – *Micro-Cap*: **Entering Coupling between *L1* and *L2***
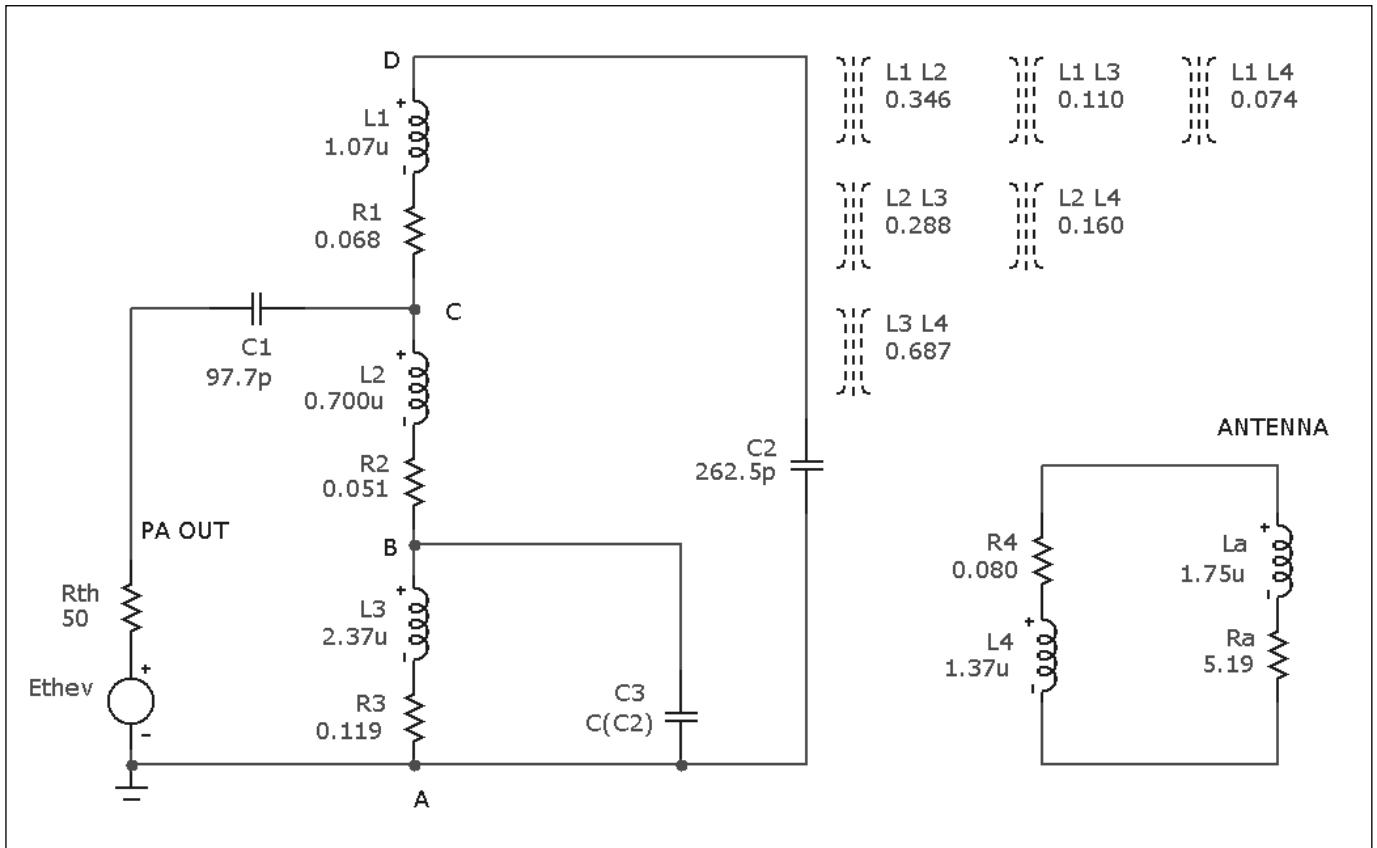
Figure 18 — *Micro-Cap*: **Complete Circuit**
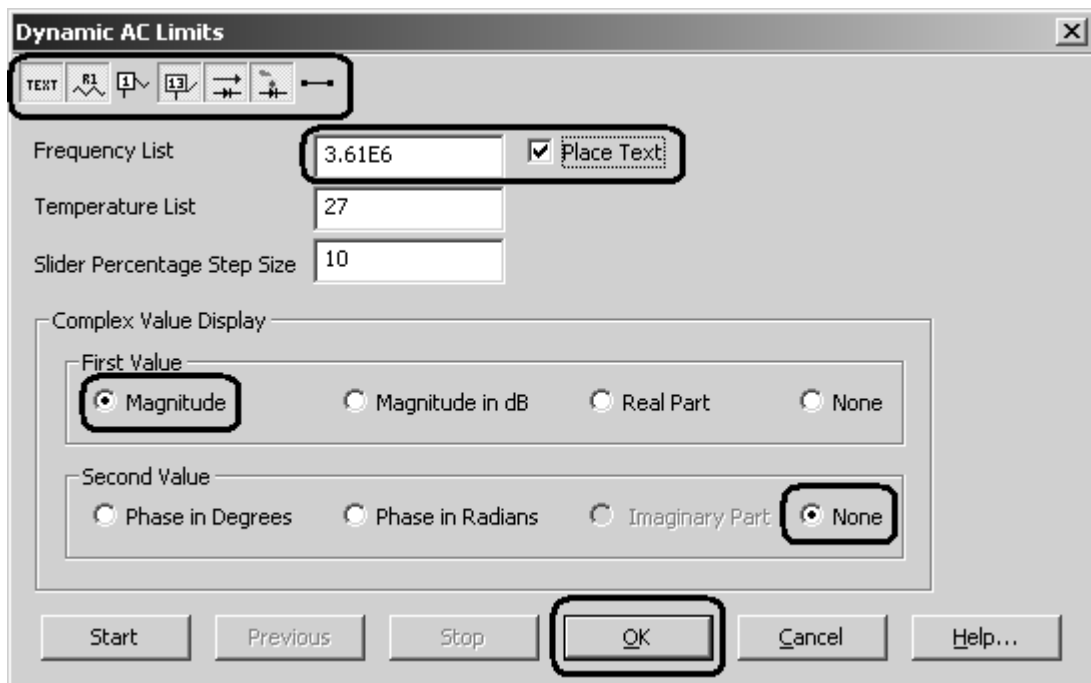
Figure 19 — *Micro-Cap*: **Dynamic AC Limits Dialog**
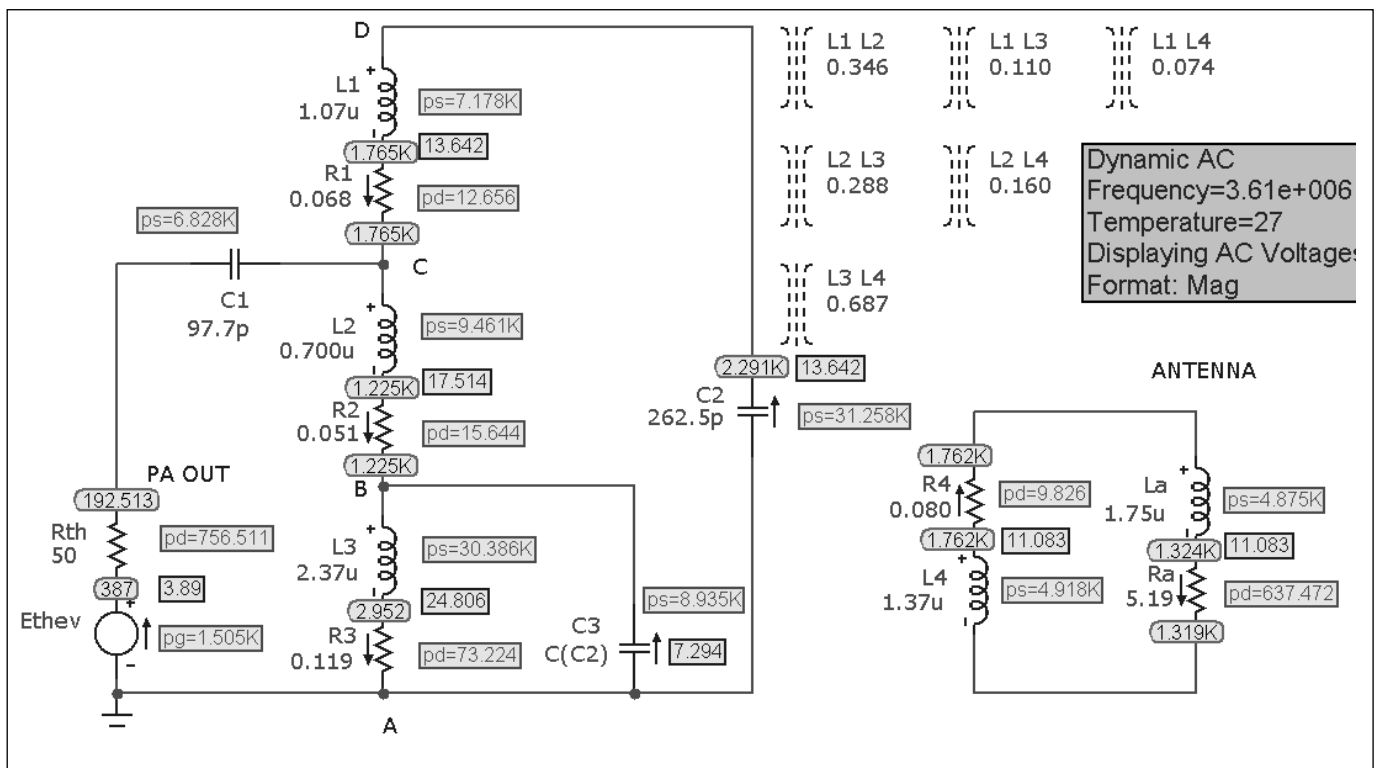
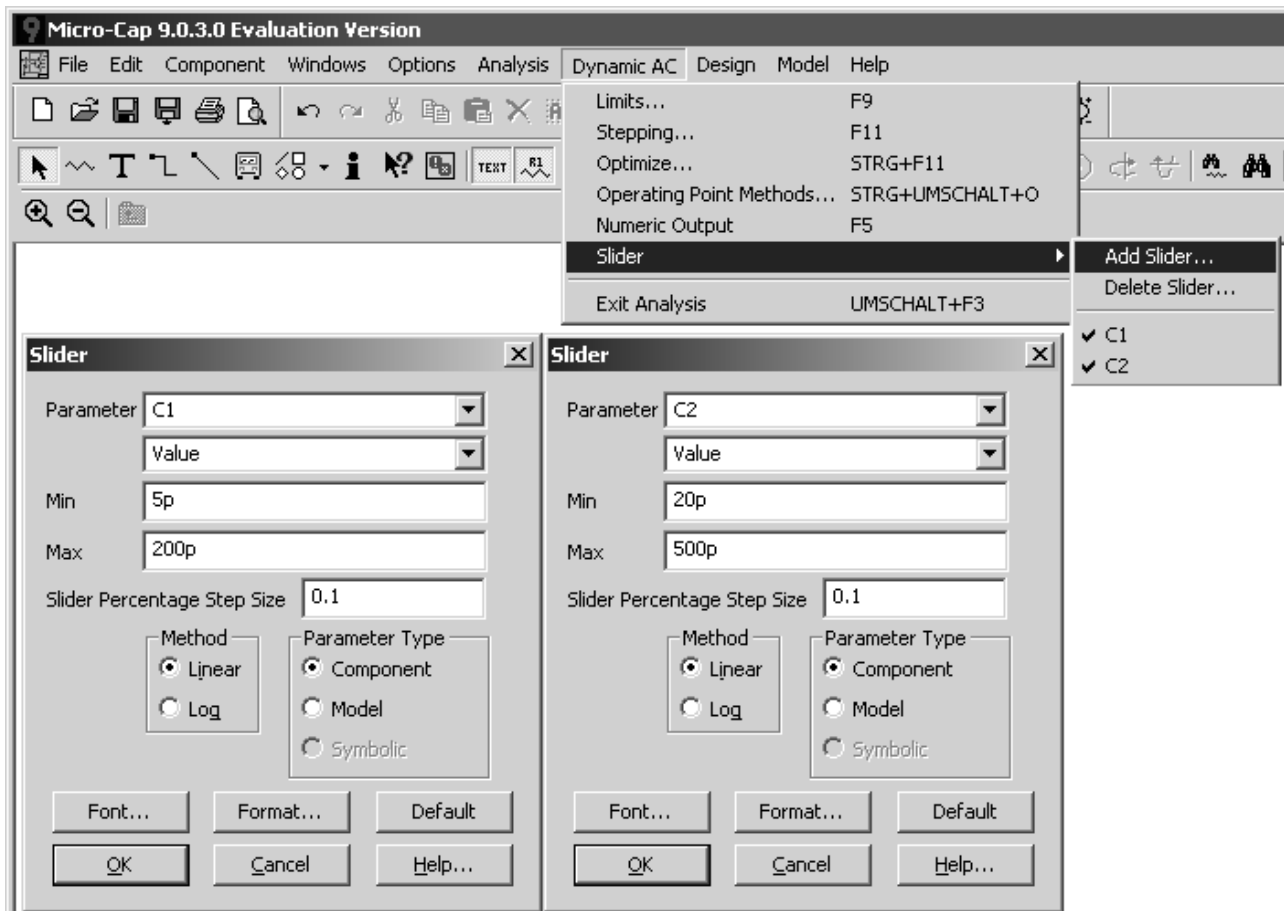**Figure 20 –** *Micro-Cap*: **Results of Analysis**

Figure 21 – *Micro-Cap*: **Adding Sliders for** *C1* **and** *C2*

ing for this particular antenna and frequency. Let us learn how to interpret simulation results before playing with sliders.

So, what makes the *Micro-Cap* display so crowded? Small blue arrows, one in each circuit branch, indicate chosen positive current flow directions. Blue bordered blue numbers are current magnitudes. Note that the current in *L3* (coil part between PA coax shield and first tap, upper part on Figure 1) is 24.8A. This is considerably higher than in any other component, but that is how impedance transformation works.

Pink bordered black numbers are node voltages with ground as common reference. For node "D" (upper end of primary coil, where C2 connects), it is 2.3kV. A bigger gap from other conductors is a good idea.

Red bordered red texts are power terms. Next to the voltage source, it is *pg* – generated power. Next to every resistor, it is *pd* – dissipated power. We want as much of it as possible in *Ra* (antenna) and as little as possible dissipated in inductors. You may want to check the power balance. The sum of all *pd-s* should be *pg*. Then, next to each inductor and capacitor, *ps* – stored power – is displayed. (Stored power balance cannot be calculated, since the circuit contains coupled inductors.)

Let's calculate tuner efficiency (valid for this particular antenna on this particular frequency). If *Pa* is power dissipated in antenna and *Pt* – power dissipated in tuner, then the efficiency is .

$$\eta = 100\% \times P_a/(P_a + P_t) = 100\% \times 637/(637 + 13 + 16 + 73 + 10) = 85\%$$

Well, there is no free lunch; this is still a reasonable tuner performance with such a short antenna.

It is more interesting to let *Micro-Cap* calculate efficiency. Moreover, we let it calculate voltages across *C1, C2* and *C3*. To do so, add five text entries anywhere in the circuit sheet using the text tool. Enter *expressions* as listed in Table 8 into the text field. It is important that you mark the "Formula" checkbox. Parts in brackets are now not static text, but formulas for *Micro-Cap*. A question may arise - why use the magnitude function for dissipated powers? In *Micro-Cap*, dissipated powers are complex numbers, having the same phase as the current flowing through a resistor. They would add as vectors, and we want a simple scalar calculation.

The last expression in the list is for the antenna plus tuner system impedance as "seen" by the PA output. Load impedance is output voltage divided by output current. To refer to them in a *Micro-Cap* expression, we need node numbers. Activate them with the node button. You may need to change node numbers in the impedance expression according to ones you have. Figure 23 shows the circuit with node numbers activated and
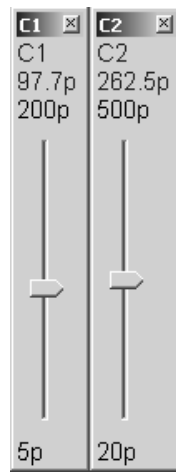
Figure 22 – *Micro-Cap*:
C1 and **C2** Sliders
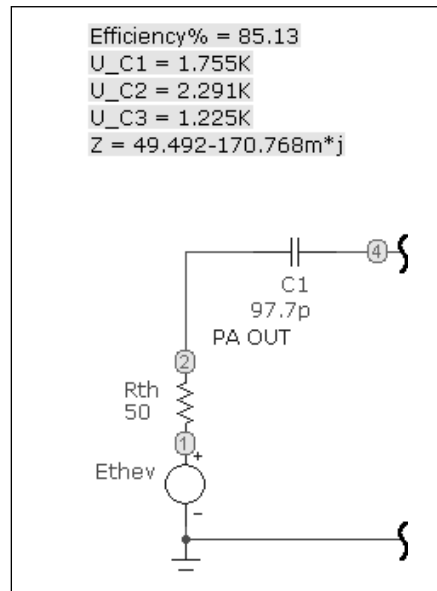
Figure 23 – *Micro-Cap*: **Node Numbers and Expression Results**

expression results.

It looks like the tuning obtained is very good, with $Z = 49.5-0.17j\ \Omega$, at price of 15% PA output power dissipated in tuner. The "hot" half C2 of double variable capacitor has to withstand 2.3 kV, so I would use one with at least 3mm air gaps (or butterfly split with 1.5mm).

Now, let's get back to the 73W power dissipated in *L3*. This is a considerable amount of heat, comparable to a mid-sized soldering iron. Peter E. Chadwick, G3RZP, told me about production Z-Match tuners being quite often returned with the coils unsoldered or even melted. It happened in the 1970's with a different circuit of Z-Match, but this is surely something to avoid. According to Table 22.44 - Copper Wire Specifications[10], 14 AWG current carrying capacity in continuous duty in open air is 32A dc. We saw that skin effect increases wire resistance 11.5 times at 3.6 MHz compared to DC. Assuming

the same allowed heating power, current has to be $\sqrt{11.5} = 3.4$ times lower at 9.4A, or 2.6 times less than expected in *L3* according to simulation. To stay within the rated current, output power has to be reduced $2.6^2 = 6.8$ times, meaning from 750W to about 110W when operating with a 100% duty cycle mode like RTTY. It is bad news but better found now than after tuner assembly!

It is time to see how critical the tuning is. Move *C1* and *C2* sliders, you will see that antenna current max is rather smooth, and load impedance is closest to 50Ω there. If you want to try out other bands, keep in mind that values of components representing the antenna are only valid on one frequency (and you may need to replace *La* with *Ca*). Inductor losses due to skin effect increase with frequency, too.

Summary

My purpose was to show how *Micro-Cap* works, especially how to simulate tapped and coupled inductors using *FastHenry2*. We used a real example – the Z-Match tuner. Of course, there is lot more about *Micro-Cap* to discover (even before hitting the limits of the demo version). I hope I could encourage you to try your own simulation.

Born 1978, Oleg was licensed first in 1996 in Ukraine, and graduated as master for automation and control from Donetsk National Technical University in 2000. Now living in Germany. Oleg is a process automation engineer at work in food industry and a proud owner of Elecraft K2 at home; also programming embedded real-time systems. Some interests are CW, weak signal modes and station remote control, including CW contest operation. He is a member of D.A.R.C. OV E07.

Notes

[1]Charles Lofgren, W6JJZ, "The Z-Match Coupler-Revised and Revisited", *The ARRL Antenna Compendium, Vol. 3*, 1992

[2]Lloyd Butler, VK5BR, The Simple Z Match Tuner Simplified, **users.tpg.com.au/users/ldbutler/SingleCoilZMatch.htm**

[3]Spectrum Software Home Page, **www.spectrum-soft.com/index.shtm**

[4]SPICE (Simulation Program with Integrated Circuit Emphasis), **en.wikipedia.org/wiki/SPICE**

[5]*Micro-Cap* evaluation version request form, **www.spectrum-soft.com/demoform.shtm**

[6]Computational Prototyping Group at MIT, **www.rle.mit.edu/cpg/research_codes.htm**

[7]*FastFieldSolvers*, **www.fastfieldsolvers.com/dwnld2.htm**

[8]Coilgen, a *FastHenry* input file generator for arbitrary coils systems,Claudio Girardi, IN3OTD, **www.qsl.net/in3otd/coilgen.html**

[9]Files used in this article, **www.mydarc.de/dl2ipu/zmatch_article**

[10]Mark J. Wilson, K1RO, Editor, *The ARRL Handbook for Radio Communications 2010*, Newington, CT, American Radio Relay League, ISBN: 0872591468

QEX-

Dave Gordon-Smith, G3UUR

Whitehall Lodge, Salhouse Road, Rackheath, Norwich, NR13 6LB, England; daveg3uur@gmail.com

Seventh-Order Unequal-Ripple SVC Low-Pass Filters with Improved Second Harmonic Attenuation

Enjoy improved filter performance while still using standard-value capacitors.

An unequal-ripple (*acromorphic*) low-pass filter can offer an excellent match over a bandwidth that is more than adequate to accommodate even the widest HF amateur band, including suitable tolerance margins, and still provide the increased harmonic attenuation associated with high-ripple Chebyshev designs. The inherent advantages of unequal-ripple low-pass filters and a method of synthesizing them have already been described in previous *QEX* articles.[1, 2]

In addition to the improved performance they provide, there is another major advantage that stems from their great flexibility and that is the possibility of designing them to use single preferred or *standard value capacitors* (SVCs). This feature is useful in keeping the component count and cost low, particularly for high-order filters.

The standard 7th-order unequal-ripple acromorphic design usually provides adequate attenuation for all harmonics from the third upwards, but the suppression of the second can still be too low for some single-ended RF amplifiers and push-pull designs that have poor inherent balance at certain frequencies. This situation can be remedied, though, by adding a single transmission zero at the second harmonic, as shown in Figure 1, where C4 has been added to tune L4 to the second harmonic. This substantially reduces the output level of the latter and provides a better balance of attenuation for the first few troublesome harmonics. Ed Wetherhold, W3NQN, has
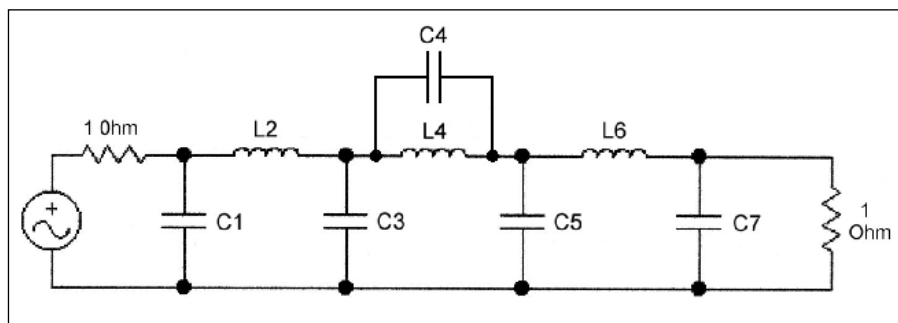
Figure 1 — Seventh-order AWAZ low-pass filter schematic.

already christened a similar equal-ripple form of 7th-order low-pass filter[3] designed by Jim Tonne as the *CWAZ* (Chebyshev with Added Zero), so this version might be called the *AWAZ* (Acromorphic with Added Zero). Whereas Jim, formerly WB6BLD and now W4ENE, had to position each amateur band in the penultimate peak of the 7th-order Chebyshev response to achieve adequate bandwidth, the acromorphic design allows the width of the usable part of the upper passband to be controlled by variations in the pole positions. These are determined by the choice of component values, and can be tailored so that the usable upper passband is broad enough to encompass two adjacent HF amateur bands or restricted to just one with an attendant increase in harmonic attenuation.

Types of Unequal-Ripple Response

There are basically three different types of 7th-order unequal-ripple response that are suitable for transmitter low-pass filter applications, each of which has its own particular set of characteristics. The amplitude plots shown in Figures 2, 3 and 4 illustrate these three useful types of unequal-ripple response. The extent of the usable passband (UPB) is marked by a solid black bar in these plots. The level of the return loss within this UPB region can be set to meet a range of specifications by the appropriate choice of component values. The designs that correspond to these responses are all symmetrical about the center components, C4 and L4 in Figure 1, so what applies to the relationship between C3 and C1 also applies to C5 and C7, because C1 = C7 and C3 = C5. Similarly, L2 = L6, so conditions or statements relating to L2 also apply to L6.

The Type 1 response shown in Figure 2 uses all outer poles, which lie on an ellipse, to produce a usable upper passband with

[1]Notes appear on page 34.

3 peaks closely spaced to give low equal ripple. Their spacing from the pole at ω=0, which lies inside the ellipse, and its real part determine the depth of the lower valley. The UPB can easily be controlled to provide any minimum level of return loss. Often 26.4 dB, which corresponds to a VSWR of about 1.1:1 and a ripple of 0.01 dB, is quoted as a minimum acceptable figure. However, a lower VSWR limit and greater return loss specification than this might be preferable for broadband solid-state RF amplifiers, and possibly 30 dB return loss, or better, should be the aim. For the circuit shown in Figure 1 this requires the ratio of C3 to C1 to be between 1.83 and 2.11, depending on the *lower valley ripple* (LVR). Designs with low LVR can achieve a return loss of 30 dB with ratios between 1.8 and 1.9, but at LVR=1.2 dB a ratio of over 2.1 is required to get the same return loss in the upper part of the passband.

A Type 2 response is illustrated in Figure 3, and here the depth of the middle valley has increased so that the peak lower in frequency is no longer a usable part of the upper passband because the return loss in the valley between it and the rest of the upper passband does not meet the minimum specification. This occurs when the value of L2 drops a certain amount below the value that produces equal ripple in the upper part of the response; the amount depending on the level of equal ripple relative to the maximum tolerable ripple. The ripple in the *middle valley* (MVR) continues to increase if the value of L2 is reduced further below this critical value. There is a see-saw relationship between the valleys, and the upper valley reduces in depth as the middle one increases if the other component values remain constant. Allowing a Type 1 response to degenerate into a Type 2 can sometimes be a useful strategy to get a better return loss over an amateur band in a design that otherwise fits in well with standard values of capacitor, but doesn't have quite a good enough return loss with an equal-ripple response. Normally, a useful improvement can be achieved with around a 1% decrease in inductance. However, if this strategy requires L2 and L6 to be reduced by more than 2%, the ratio C3/C1 may also need to be reduced in order to continue increasing the return loss and improving the match in the UPB of the Type 2 response.

Filters with the Type 3 unequal-ripple response shown in Figure 4 offer slightly better harmonic attenuation than the other two types, but the bandwidth of the usable part of the response is very narrow and only sufficient for amateur applications where the width of the band is less than approximately 2.5% of the center frequency. Typical C3/C1 ratios for the Type 3 unequal-ripple low-pass response are between 1.4 and 1.6. Of the three types of response, Type 1 offers the widest possible usable bandwidth and Type 3 the narrowest. As far as harmonic suppression is concerned, the higher the value of LVR or MVR, the higher the stop-band attenuation in all three types. Also, for the same value of MVR and LVR, Type 3 filters offer more harmonic suppression than Types 1 or 2, and Type 2 can offer more than Type 1 if the middle valley is deep enough. However, Type 1 unequal-ripple filters with modest values of LVR usually offer more than adequate harmonic attenuation for most applications, and the additional attenuation gained by using one of the other types, or a higher value of LVR, is relatively
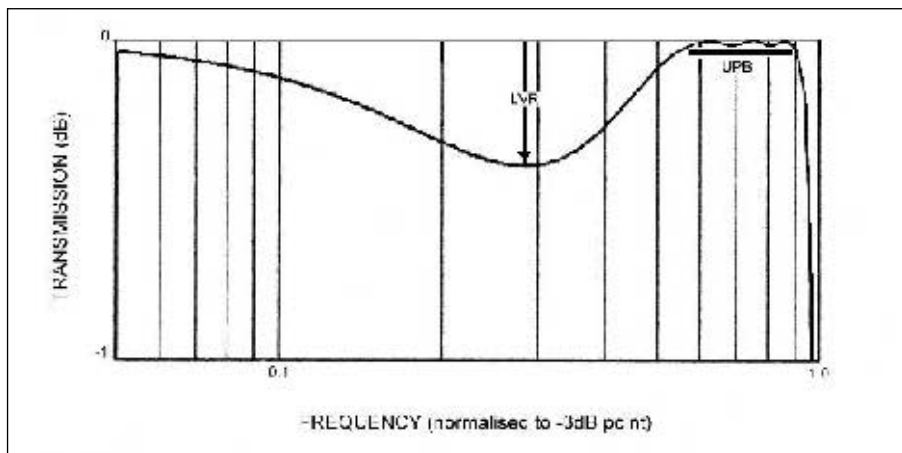
Figure 2 — Type 1 amplitude response for a seventh-order unequal-ripple low-pass filter.
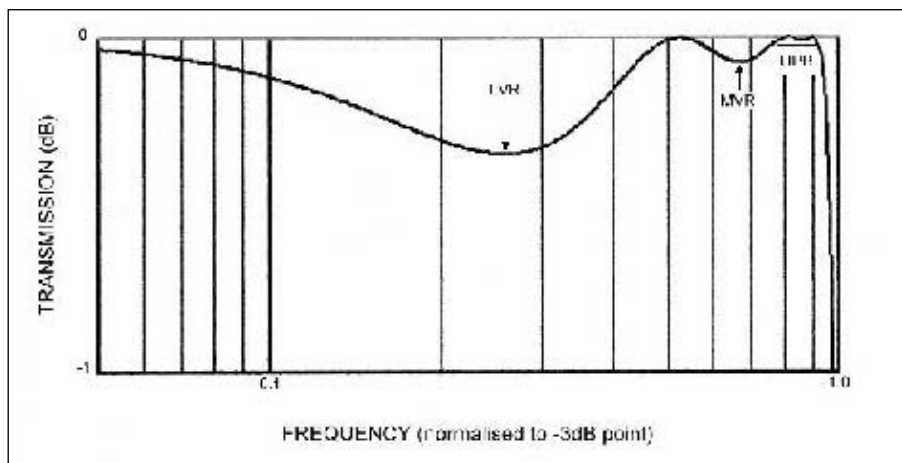
Figure 3 — Type 2 amplitude response for a seventh-order unequal-ripple low-pass filter.
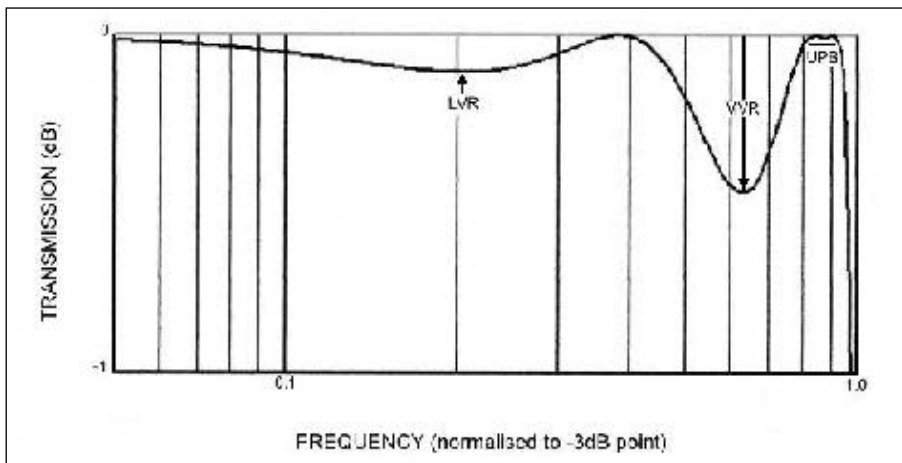
Figure 4 — Type 3 amplitude response for a seventh-order unequal-ripple low-pass filter.

small for the AWAZ version of the filter, and only comes at the cost of greater sensitivity to component values.

Designing SVC Filters with a Type 1 Response

There are a wide range of solutions which give low equal ripple in the upper part of the passband of 7th-order acromorphic low-pass filters, and to present a table of normalized coefficients for a selected few AWAZ designs would be too limiting. Also, the ratio of standard capacitor values varies throughout their range, and depends on whether their capacitance is based on 1%, 2%, 5% or 10% spot values. The availability of different ratios of capacitance for different parts of the spectrum complicates the selection of appropriate normalized designs. So, to leave the range of values as wide open as possible, some suggestions have been put forward in Table 1 as guidance to readers on suitable component choices for 7th-order, Type 1, AWAZ low-pass filters, which give a fair degree of freedom. The values presented in Table 1 are normalized with respect to the -3dB cut-off frequency, apart from the *Min C3/C1* column, which is the suggested minimum ratio of C3/C1 for a return loss of 30 dB. Ratios higher than this minimum produce equal upper pass-band ripple with a return loss greater than 30 dB, and conversely lower ratios produce lower return loss for a Type 1 response. However, making the ratio of C3 to C1 too high results in so much narrowing of the usable upper passband as the three peaks come together to produce ever lower ripple and higher return loss that it becomes self-defeating in the end. The figures for *C1, L2, FL, FU* and *FCR* in columns 2, 3, 6, 7 and 8, respectively, are taken directly from the original normalized table presented in the "Notes" reference (1) for the conventional form of 7th-order unequal-ripple low-pass filter. The other two columns contain suggested starting values and minimum ratios for *C4* and *C3/C1* that are specific to the AWAZ version.

The design procedure is much the same as for other normalized filter tables, but requires the final adjustment of the value of L2 (and L6) to be done using a circuit analysis program, such as *AADE 4.42*,[4] or *in situ* when the constructed filter is undergoing its final check prior to use. Taking the 20-meter band as an example, add about 4% to the upper band limit to allow for component tolerances and then multiply this by the figure in the *FCR* column to get the cut-off frequency. The values of *C1* at the upper and lower limits of Table 1 can be de-normalized to establish the range of C1 for the 20-meter band, if desired. Alternatively, a value for *C1* around the middle of Table 1 can be de-normalized and the nearest SVC chosen. For example, 14.350 MHz multiplied by 1.04

Table 1

Normalized component values and design data for 7th-order, Type 1, AWAZ low-pass filters.

| LVR	C1	L2	Min C3/C1	Median C4	FL	FU	FCR
0.10dB	1.0191	1.4819	1.842	0.321	0.4597	0.8677	1.1525
0.20dB	1.0936	1.4264	1.866	0.334	0.5056	0.8723	1.1464
0.30dB	1.1545	1.3854	1.890	0.348	0.5269	0.8777	1.1393
0.40dB	1.2090	1.3511	1.914	0.361	0.5389	0.8830	1.1325
0.50dB	1.2577	1.3221	1.938	0.375	0.5462	0.8880	1.1261
0.60dB	1.3053	1.2939	1.962	0.388	0.5508	0.8928	1.1201
0.70dB	1.3500	1.2689	1.986	0.403	0.5536	0.8971	1.1147
0.80dB	1.3796	1.2510	2.009	0.419	0.5593	0.8983	1.1132
0.90dB	1.4090	1.2268	2.033	0.434	0.5650	0.8996	1.1116
1.00dB	1.4423	1.2040	2.057	0.450	0.5707	0.9008	1.1108
1.10dB	1.4633	1.1862	2.081	0.466	0.5763	0.9020	1.1086
1.20dB	1.4842	1.1684	2.105	0.481	0.5820	0.9033	1.1070
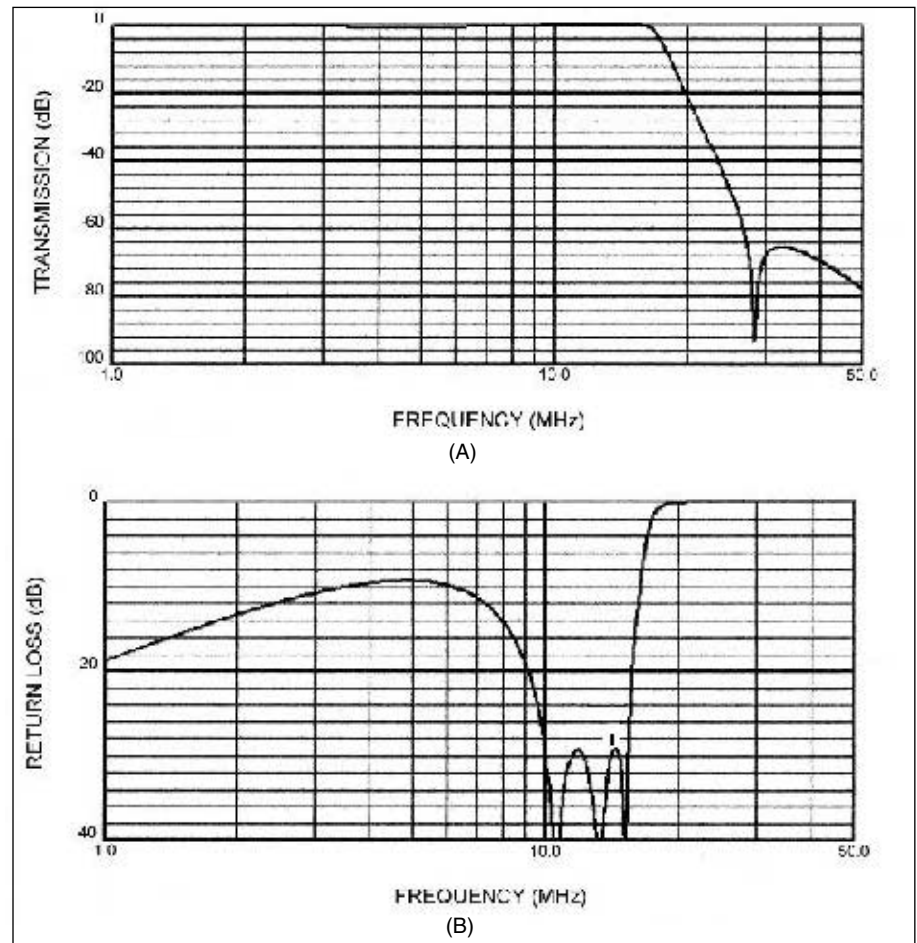
Figure 5 — (A) Amplitude Response and (B) Return Loss plot for the LVR = 0.55 dB example of 7th-order AWAZ low-pass filter design described in the text. The bold black bar marks the 20-meter band.

and then by *FCR* = 1.1261 from the *LVR* = 0.5 dB row gives roughly 16.806 MHz for the -3d B frequency. De-normalizing *C1* = 1.2577 with this cut-off frequency gives C1 = 238.2pF for 50-Ω terminations. The nearest preferred value is 240 pF. According to Table 1, C3 should be at least 1.938 times C1 for a return loss of 30 dB at this value of LVR, so the minimum value for C3 should be at least 465.8 pF. This is lower than the closest

preferred value of 470 pF, so it should work fine. The normalized value of *L2* is 1.3221, and de-normalizing this gives approximately 626 nH. The recommended normalized value for *C4* is 0.375, which corresponds to 71 pF when de-normalized. This is closer to 68 pF than 75 pF, so let's make it 68 pF. In order to get 68 pF to resonate at the mid-band second harmonic of 28.35 MHz requires L4 to be 463.5 nH.

Checking the design with the *AADE 4.42* software shows that the value of L2 (= L6) is a bit too high and needs re-adjusting down to 597 nH to give a perfect equal-ripple UPB response. It's best to use the return loss plot to check the effect of changing L2 and L6 because the variations in amplitude are exceedingly small compared to the changes in return loss. Altering the inductance of both L2 and L6 in a systematic way, while monitoring the return loss level in the UPB, allows the right direction of change to be established and the size of the increments can then be adjusted suitably as the required value is approached.

The LVR of the filter after rounding off the capacitors to the nearest preferred values and adjusting L2 and L6 for the best response is 0.55 dB. The amplitude response and return loss plots for this Type 1 unequal-ripple filter are shown in Figure 5 A and B. The notch in the amplitude response is tuned to 28.35 MHz and the attenuation over the range 28 to 28.7 MHz is greater than 75 dB. The third harmonic, starting at 42 MHz, is more than 71 dB down on the fundamental. In Figure 5B the return loss figures are increasing (improving) downwards, and the broad peak rising to 9 dB down just below 5 MHz is the return loss associated with the lower valley where LVR = 0.55 dB. The dip in the plot above 10 MHz indicates the position of the usable part of the passband. The two small peaks in the dip indicate the return loss associated with the ripple in the upper passband. The bandwidth can either be taken at the level of these peaks in the passband, or at some other convenient level. Since the peak return loss in the passband will vary slightly from design to design, it's more convenient to use the minimum acceptable return loss (26.4 dB) for comparison purposes. In this case, the bandwidth at the 26.4 dB return loss level runs from 9.832 to 15.498 MHz. The worst-case return loss associated with the ripple in this range peaks at 29 dB down. This is slightly less than the design value because component choices have forced LVR higher, but still quite acceptable. The upper frequency limit, FU, is a bit more than 4% high of 14.35MHz and could be positioned better, but the final design gives excellent results despite this deficiency.

Dropping the value of C1 down to 220 pF will reduce the LVR and require L2 and L4 to be larger for roughly the same cut-off frequency. If L4 needs to be larger, this can only be achieved by making C4 smaller, 62 pF for example. The value of L4 will then need to be 508 nH for parallel resonance at 28.35 MHz. C3 can be made 430pF, since this is more than adequate for the lower value of LVR. Comparing 220 pF with 238.2 pF suggests that the value of *LVR* will be between 0.3 and 0.4dB. Using the average of the normalized *L2* values (1.3683) for these two rows predicts that L2 will be around 648 nH. Analysing the circuit and adjusting L2 for best response produces an equal-ripple UPB with LVR = 0.34 dB. The re-adjusted value of L2 needs to be 626 nH.

This achieves a minimum attenuation of 75 dB for 28 to 28.7 MHz and over 70 dB at the third harmonic on 42 MHz and above, as can be seen from the frequency response in Figure 6(A). This is not quite as good as the previous design for suppression of the third harmonic but quite a bit better on return loss within the UPB, which peaks at 31.5 dB down as shown in Figure 6(B), and gives a passband between 9.536 and 15.403 MHz at a return loss of 26.4 dB.

A third possibility is to raise the value of LVR by increasing C1. Comparing 270 pF with 238.2 pF suggests that the next design might have an *LVR* between 0.9 and 1.0 dB (normalized *C1* = 1.4256). Taking the aver-
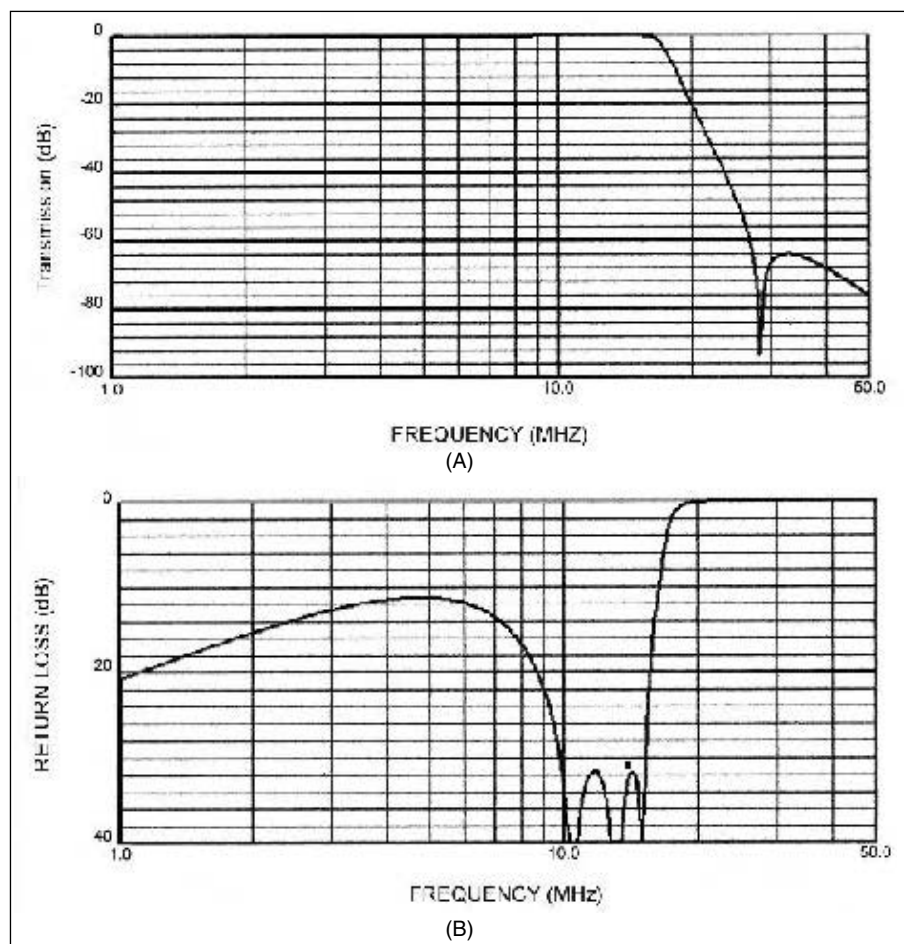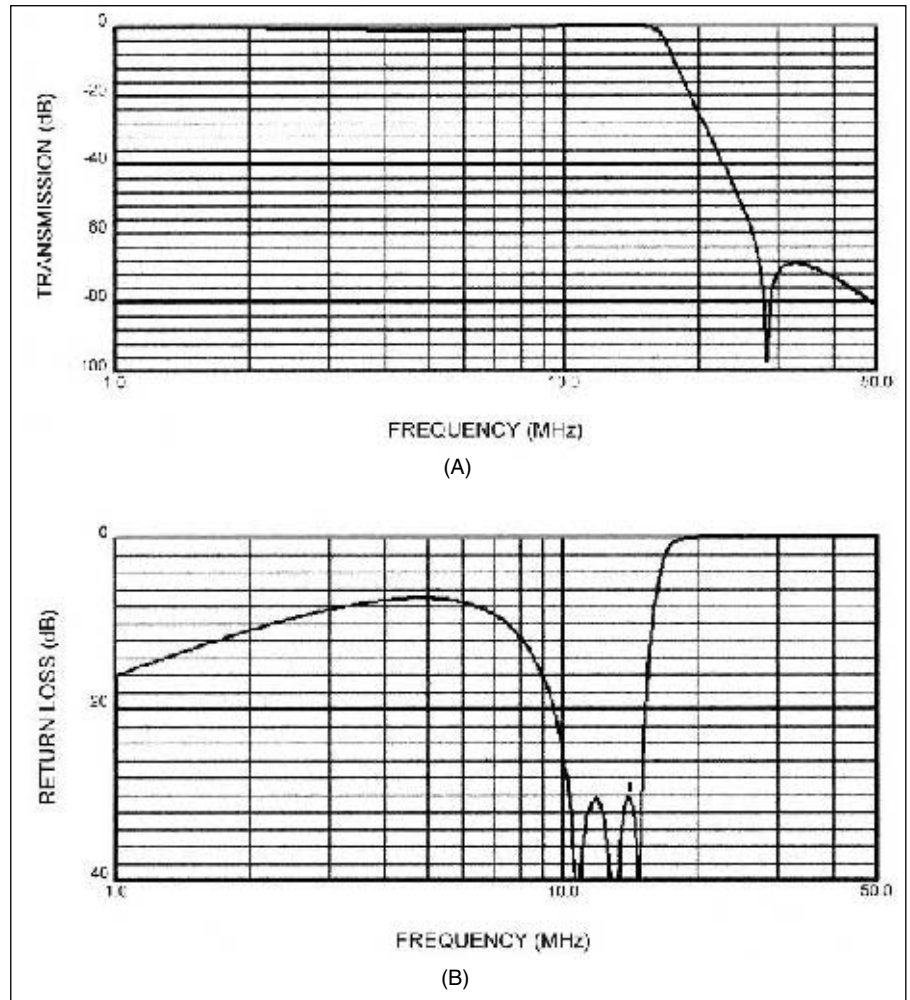

Figure 6 — (A) Amplitude Response and (B) Return Loss plot for the LVR = 0.34 dB example of 7th-order AWAZ low-pass filter design described in the text. The bold black bar marks the 20-meter band.

Table 2

Component values and performance data for 20-meter, 7th-order AWAZ low-pass filters used as design examples.

LVR	C1	L2	C3	L4	C4	C5	L6	C7	FU (26.4dB)	Min 2nd Harmonic Attenuation	Min Return Loss
0.25 dB	220 pF	667 nH	430 pF	563 nH	56 pF	430 pF	667 nH	220 pF	14.842 MHz	79 dB	30.5 dB
0.34 dB	220 pF	626 nH	430 pF	508 nH	62 pF	430 pF	626 nH	220 pF	15.403 MHz	75 dB	32.0 dB
0.45 dB	240 pF	630 nH	470 pF	508 nH	62 pF	470 pF	630 nH	240 pF	14.991 MHz	79 dB	29.0 dB
0.55 dB	240 pF	597 nH	470 pF	464 nH	68 pF	470 pF	597 nH	240 pF	15.498 MHz	75 dB	29.0 dB
0.96 dB	270 pF	570 nH	560 pF	420 nH	75 pF	560 pF	570 nH	270 pF	15.028 MHz	80 dB	30.0 dB

age normalized value of *L2* for these two rows gives 1.2154. Using the original cut-off frequency, which is the quickest but not the most accurate way of doing it – *FCR* should be corrected - gives L2 = 575.5 nH. L4 needs to be lower than the first design, so C4 needs to increase to 75 pF. This makes L4 = 411.4 nH for resonance at 28.35 MHz. Table 1 indicates that C3 must be more than twice C1, so that can be 560 pF. Modelling the filter with these values indicates that the value of L2 needs to be reduced to 570.4 nH to achieve an equal-ripple response in the upper passband and LVR turns out to be 0.96 dB. Its response presented in Figure 7(A) shows an increase of 3 to 5 dB in harmonic attenuation has been achieved over the previous two designs (80 dB down at 28 MHz and 74 dB at 42 MHz). Also, Figure 7(B) indicates that the worst-case return loss in the middle region of the upper passband is 30 dB down. The usable bandwidth at a return loss of 26.4 dB is between 10.094 MHz and 15.028 MHz. The return loss for 14.000 to 14.350 MHz is 30 dB or better.

Variations on a Theme of C4

The normalized figures for *C4* given in Table 1 are just recommended starting values, and can generally be varied by more than ±15% to move the cut-off frequency of the filter around. The value of LVR and the upper pass-band ripple will also change a little as the value of C4 is varied. By dropping the value of C4 in the first example from 68 pF to 62 pF, increasing L4 to 508 nH and L2 to 630 nH, the upper end of the usable passband at *FU* can be brought down from 15.403 MHz to 14.991 MHz. This still leaves a reasonable margin for component tolerances, but offers a slight improvement in harmonic attenuation. It may be possible to use tighter margins in practice. After all, the capacitor values may vary, but the inductor values can always be trimmed *in situ* by compression or expansion of the windings to give a good VSWR over the entire amateur band when the filter is terminated with a 50-Ω load. The value of L4 should always be adjusted *in situ* to set up the transmission zero on the right frequency, so it shouldn't be much of an imposition to set-up the values of L2 and L6 for best input VSWR at the same time. Moving the cut-off frequency down in the first example improves the attenuation at 28MHz and 28.7 MHz by 4dB and the attenuation at 42 MHz by 2 dB over the original design, which had C4 = 68 pF and L2 = L6 = 597 nH.

Table 2 gives the component values of this design and the previous design examples of AWAZ low-pass filters for the 20-meter band. The upper pass-band limit *FU* is specified as the frequency at which the return loss drops to 26.4 dB. The return loss over the 20-meter amateur is better than this, and is

Figure 7 — (A) Amplitude Response and (B) Return Loss plot for the LVR = 0.96 dB example of 7th-order AWAZ low-pass filter design described in the text. The bold black bar marks the 20-meter band.

the figure shown in the *Min Ret Loss* column.

It may appear that the LVR = 0.34 dB design could benefit from an increase in the values of L2, L4 and L6 as well, because FU = 15.403 MHz seems a bit high. That's true, and changing C4 to 56 pF and re-adjusting the values of all three inductors to give equal ripple in the upper passband and a notch at 28.35 MHz produces a filter with FU = 14.842 MHz. This is quite a bit tighter than the first alternative design at the upper end of the passband, but should still leave enough of a margin for component tolerances, variations with temperature and ageing if the inductor values are set carefully at room temperature in the first instance. It's always tempting to go for the ultimate in performance, but there will always be a point where the next step is a step too far or a jump too big with single preferred-value components in the C4 position. Obviously, juggling with the values of two parallel capacitors in this position could bring FU nearer the optimal frequency in certain cases, and perhaps gain a small improvement in performance, but would it be worth the additional effort and the

extra space required in each filter? Possibly not for the 20-meter band, but for the proportionally wider bands, where more could be gained, it might be.

Juggling with the values of parallel capacitors for C4 is really essential for Type 3 AWAZ low-pass filters because their usable bandwidth is often only just wide enough if suitable tolerance margins are included on either side. For example, Figure 8 shows the return loss plot for a 20-meter Type 3 AWAZ filter (LVR = 0.11 dB and MVR = 0.47 dB) with C1 = 270 pF and C3 = 390 pF. In this case the tolerance margins actually need to be wider than the amateur band, which is indicated by the extent of the solid bar above the 30 dB level in this plot. The value of C4 has had to be adjusted carefully to position the amateur band in the middle of the usable part of the passband. Fortunately, the return loss in the UPB of Type 3 unequal-ripple filters can be controlled by varying the inductance of L2 and L6, and a good compromise between bandwidth and return loss can be found for the 20-meter amateur band in this case with L2 = L6 = 702 nH. C4 works out nicely to be
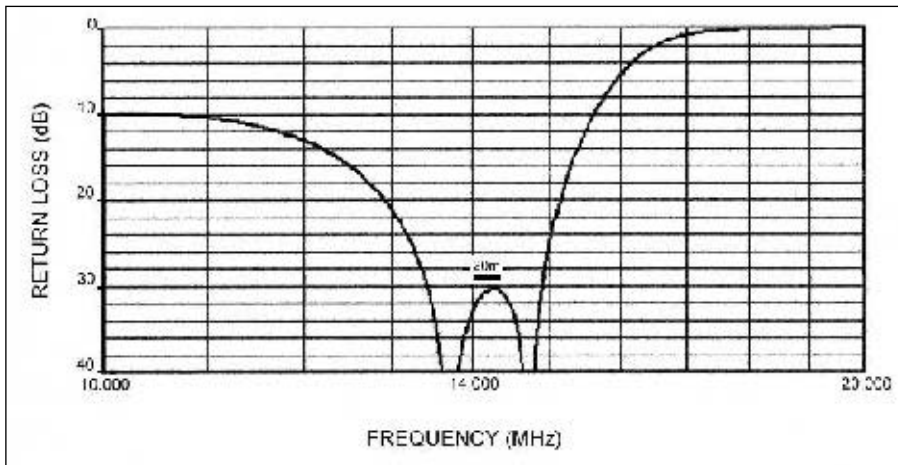
Figure 8 — Return Loss plot between 10 and 20 MHz for a 20-meter Type 3 AWAZ low-pass filter.

two 22 pF capacitors in parallel. However, the minimum second harmonic attenuation for this Type 3 AWAZ low-pass filter is only 2 dB more than the best Type 1 AWAZ design, so this small improvement in performance probably doesn't warrant its use, bearing in mind its very limited bandwidth and the need for more precise values of C4.

Summary and Final Remarks

All the design examples presented in this article offer a very good match over the 20-meter amateur band with ample margins for component tolerances and an excellent balance of attenuation between the second and third harmonics. The Type 1 AWAZ designs can all be constructed with single preferred-value components in the capacitor positions, if required. The design examples presented show there is considerable latitude in the choice of capacitor values for 7th-order AWAZ low-pass filters and usually a good design can be based around standard values without the need to parallel any capacitors to achieve the correct values. Sometimes, it can be convenient to use two capacitors in parallel to form just the right value for C4 to position the upper end of an amateur band at the optimum point in the passband to achieve the greatest possible attenuation, but designs with more-than-adequate performance can be achieved with single capacitors of the nearest preferred value for most amateur bands. However, when space is not a consideration and plenty of capacitors of a particularly suitable type for C1 and C7 are already freely available, there is no good reason not to consider using two of these capacitors in parallel for C3 and C5 if a ratio of 2.0 is suitable for the particular design being considered.

The suggested initial values of L2 for the Type 1 AWAZ filter examples discussed in this article have all had to be reduced in order to achieve an equal-ripple condition in the upper part of the passband, but in practice the

initial value of L2 will sometimes need to be increased. It all depends on the proportional width of the amateur band being considered, and the relative positions of FU and the frequency to which C4 and L4 are tuned. The difference between the suggested starting value and the actual value required to achieve the equal-ripple condition in the upper passband should not amount to much more than ±5%. This is within the range of adjustment of most coils, and if the exact value required to do this is of no interest, the software analysis and adjustment stage can be omitted. The values of L2 and L6 can then be adjusted for best input VSWR over the band when the fully constructed filter is terminated in a 50-Ω load.

Positioning the transmission zero (notch) is a relatively simple matter for the narrower amateur bands, such as the 20-meter band, because the second harmonics of signals at the band edges fall within the region of the notch that's almost symmetrical, and similar harmonic attenuation figures can be obtained at both ends of the band by setting the notch to the second harmonic of the center frequency. However, in order to achieve equal harmonic attenuation at both ends for the proportionally wider bands, C4 and L4 must be tuned to a frequency somewhat lower than the second harmonic of the band center because of the asymmetry of the response well away from the notch. For example, a filter for the 75-meter band requires C4 and L4 to be tuned to 7.25 MHz, rather than 7.5 MHz, to achieve the same harmonic attenuation figures at both 7 and 8 MHz. Obviously, the minimum second harmonic attenuation will be lower at the band edges for the proportionally wider amateur bands, but over 60 dB can still be achieved on these bands with a single transmission zero.

This article has concentrated on demonstrating the range of Type 1 unequal-ripple filter designs that can be produced for a single amateur band using SVCs, but the other two types of unequal-ripple response can also be used to achieve high-performance SVC low-

pass filters with different component values for the same cut-off frequency. Type 3 AWAZ low-pass filters can achieve slightly higher harmonic attenuation than the other two types, but the width of their usable passband is often only just sufficient to accommodate the narrower amateur bands with adequate tolerance margins. Such filters can be made for the 20-meter and similarly narrow bands, but the bandwidth is usually insufficient for low-pass filters on the 75-meter and 160-meter bands. It's evident from the performance figures presented in Table 2, though, that often more can be gained from positioning the upper edge of the amateur band at the correct point in the passband than by changing LVR or the response type. Therefore, changing either can be as much a matter of finding more convenient capacitor values as it can about achieving better performance.

Designs for more 7th-order amateur band AWAZ SVC low-pass filters will be published in due course, but armed with the information given in Table 1 readers will now be able to design their own high-performance Type 1 AWAZ low-pass filters using any standard or non-standard values of capacitor they happen to have lying around already.

Notes

[1]D. Gordon-Smith, G3UUR, "Seventh-Order Unequal-Ripple Low-Pass Filter Design," *QEX*, November/December 2006, pp 31-34.
[2]D. Gordon-Smith, G3UUR, "Fifth-Order Unequal-Ripple Low-Pass Filter Design," *QEX*, November/December 2010, pp 42-47.
[3]E. Wetherhold, W3NQN,"Second-Harmonic-Optimized Low-Pass Filters, "QST, February 1999, pp 44-46.
[4]The *AADE 4.42* filter design and analysis program is available as a free download from **www.aade.com/filter.htm**

Dave Gordon-Smith, G3UUR, was first licensed in 1965 and concentrated mainly on home construction and 160-meter CW DX during his first few years on the air. Constructing his own equipment was a necessity in those days because he was an impoverished schoolboy, but it later became a source of great fun and satisfaction. Filters, antennas, and propagation have always fascinated him, and much of his Amateur Radio construction and experimentation has been driven by a desire to understand them. He tends to use theory to make up for lack of test gear, and enjoys playing with novel and unusual circuits. He served with VSO (British Peace Corps) on Grenada, where he taught math and chemistry, and operated as VP2GBR. He holds a PhD in material science and was a tenured member of the academic staff at the University of Warwick, specializing in the characterization and study of defects in crystalline solids. He's also spent quite a lot of time in the United States over the past 30 years, mainly doing research at Brookhaven National Laboratory and SUNY (Stony Brook) on Long Island. After working part time for several years, he decided to fully retire last year and devote more time to travel, voluntary work and Amateur Radio.

QEX~

Virgil Leenerts, WØINK

1007 W. 30th St, Loveland, CO 80538; w0ink@arrl.net

Loop Antennas—The Factor "N"

When designing loop antennas, most hams focus on maximizing the effective areas of the loops but there are practical limits. An alternative is to modify the number of turns to achieve acceptable performance.

Loop antennas have been in use since the earliest day of radio. When considering the design of a loop antenna, the number of turns (N) is critical because it is one of the factors that determine the voltage and current in the loop, along with the radiation efficiency. While this article won't cover the basics of how a loop antenna works (the physics of how they work have been printed in many widely available textbooks and ARRL publications), it will explore the effects of N by mathematical analysis and experimental results.

A small loop antenna is generally defined as a loop whose circumference is 0.1λ or less. If there are multiple turns, the total length of wire is restricted to be 0.1λ or less. The restrictions put practical limits on the number of turns that small loop antennas can have.

A recent project brought up the question of how many turns should be used for the design of a small loop antenna. The first and most obvious answer as indicated by the equations is that the voltage or current for a loop antenna will increase as the number of turns increase.

The expression for the open circuit output voltage of a small loop antenna is given as follows:

$$\text{Voltage} - \text{open circuit} (V_{OC}) = 2\pi f \mu_0 H_0 NA$$

Where:
 f = Frequency in hertz
 μ_0 = Permeability of Vacuum - $4\pi \times 10^{-7}$ - Henry per meter
 H_0 = Applied magnetic field - amperes per meter
 N = Number of turns
 A = Loop area in meters squared.

[1]Notes appear on page 38.

The closed circuit output current is found by dividing the open circuit voltage with the impedance of the loop. The impedance of the loop is $\sqrt{R_L^2 + X_L^2}$ where R_L and X_L are the resistance and reactance of the loop. If we assume that R_L is much less than X_L then the impedance of the loop is X_L.

Current – closed circuit
$(I_{CC}) = 2\pi f \mu_0 H_0 NA \div X_L$.

From a quick look at the equations, it appears that more turns are better for either the open circuit voltage or the closed circuit current. Before we continue, however, maybe a closer look at the reactance term will provide more insight.

The equation for reactance of an inductor is $X_L = 2\pi f L$ where L is the inductance of the small loop antenna. The inductance of a physical inductor, including small loop antennas, is calculated by using the physical properties of the inductor such as the diameter, size of wire, and shape. Typically, the final equation will multiply the result by N^2 for more than one turn. This assumes that the turns are mutually coupled, which is the normal case. The equation for the reactance of an inductor can now be stated as $X_L = 2\pi f N^2 L_{PT}$ where L_{PT} is the inductance per turn of the inductor or loop. In the case of non-mutual coupled inductors or wide spaced non-mutual coupled loop antennas, the increase in inductance is like inductors in series and thus the per-turn inductance is only multiplied by N. The equation for non-mutual coupled reactance is $X_L = 2\pi f N L_{PT}$. Note that the reactance for the inductor or small loop antenna is affected by how well the turns are coupled.

It is interesting to note that an adjustment of the reactance of an air core inductor can be made by changing the spacing between
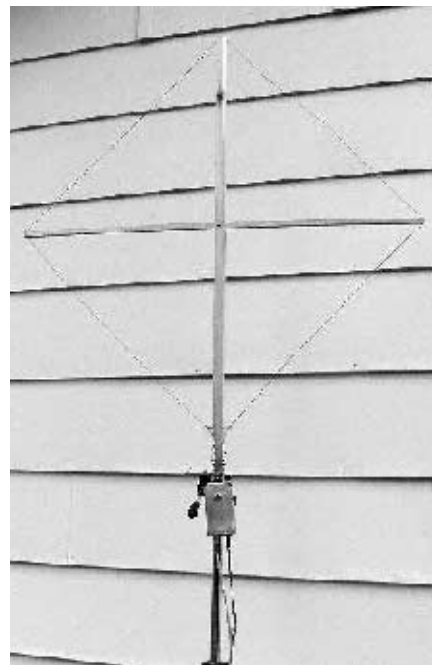
Figure 1 – The one-turn reference loop antenna.

turns (common practice for VHF air core inductors). The maximum change in reactance is going from N^2 to N^1 (if N were 4 then the maximum change would be from 16 to 4 or a factor of 4). This also brings up the possibility of tuning a loop antenna by adjusting the spacing between turns.

Let's go back to the open circuit voltage and closed circuit current equations and replace the reactance term with the number of turns, which is an explicit part of the inductance term. The equation for the open circuit voltage is: $V_{OC} = 2\pi f \mu_0 H_0 NA$. Since loop reactance is not in the open circuit voltage equation, the voltage does increase as the number of turns increases for either

mutual or non-mutual coupled turns. Loop reactance does not have an effect on the open circuit voltage.

The equation for the closed circuit current in a loop antenna with mutual coupled turns is now as follows:

$$I_{MCC} = \frac{2\pi f \mu_0 H_0 NA}{2\pi f N^2 L_{PT}}$$

reduced to

$$I_{MCC} = \frac{\mu_0 H_0 A}{N L_{PT}}$$

We now note that the closed circuit current for a loop antenna with mutual-coupled turns decreases as the number of turns increases by the factor of

$$\frac{1}{N}$$

The equation for the closed circuit current in a loop antenna with non-mutual coupled turns is now as follows:

$$I_{NCC} = \frac{2\pi f \mu_0 H_0 NA}{2\pi f N L_{PT}}$$

reduced to

$$I_{NCC} = \frac{\mu_0 H_0 A}{L_{PT}}$$

You can see how the closed circuit current for a loop antenna with non-mutual coupled turns does not decrease as the number of turns increases.

The open circuit voltage output of a receiving loop is usually connected to a high impedance amplifier, thus more turns gives a higher output voltage. In the case where the loop current is sensed, i.e. a current transformer, adding more turns lowers the loop current or has no effect if the loop turns are not mutually coupled.

Real World Tests

Mathematical equations are the tools to understand the factors for desired results, but practical data to support the equations can bring realism and validate the conclusions.

For my experiments, the test signal was provided by a local 400 kHz NDB (Non Directional Beacon) aircraft navigation station. This station was chosen since it only sent the letters "FN" and had no modulation between each identification. The time between IDs provided a dead carrier interval to make stable measurements. The loop antennas and equipment were constructed to facilitate measurements of voltage and current. The equipment for the experiment included…
• One-turn loop reference antenna
• Loop antenna frame for 2 to 4 turns
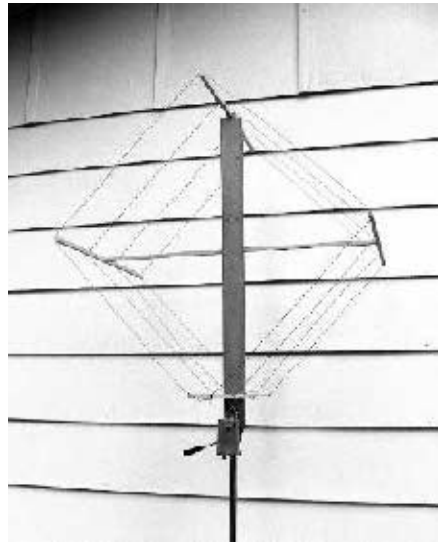
Figure 2 – The four-turn non-mutual coupled loop antenna.
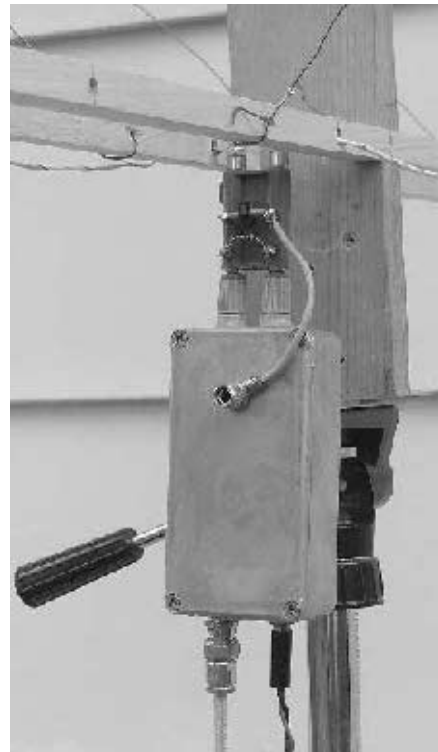
Figure 3 – The amplifier and current transformer.

• Wide space loop antenna frame for 2 to 4 turns
• High input impedance differential amplifier – unity gain into 50 Ω
• Current transformer
• Coaxial cable – 100 feet of RG-55

The frame for the square loop antennas was 635 mm on each side for an area of 0.4 meters squared. The reference antenna shown in Figure 1 was constructed separately from the multi-turn antennas so that the reference measurement could be done often to ensure that the measuring process was as constant as possible. The multi-turn mutual coupled loop antennas were made with a similar frame. The multi-turn non-mutual coupled loop antennas were constructed with 230 mm spacing between turns. The four-turn non-mutual coupled loop antenna is shown in Figure 2.

The loop inductance of each loop antenna was measured as a way of assuring completeness of the experimental data. It was also used to verify that the multi-turn antennas had mutual and non-mutual coupled turns. The inductance was measured with a Boonton inductance bridge Model 63H-S2 at a measurement frequency of 100 kHz.

The high impedance differential amplifier was constructed using two common active antenna amplifiers in a single enclosure. Their outputs were connected to a transformer such that only the difference between the two would be measured and thus any common mode voltages on the loop antenna output would be rejected. Figure 3 shows the differential amplifier with the current transformer. For the voltage measurements, the current transformer was not used and the differential amplifier was connected directly to the loop antenna.

For the current measurement, the current transformer was designed so that the output voltage of the transformer would be proportional to the input or loop current. The load on the transformer with 30 turns is 900 Ω making the insertion resistance to the loop antenna one ohm. The basic design equations and diagram for a current transformer is described in Figure 4. The ferrite core for this current transformer was Fair-Rite
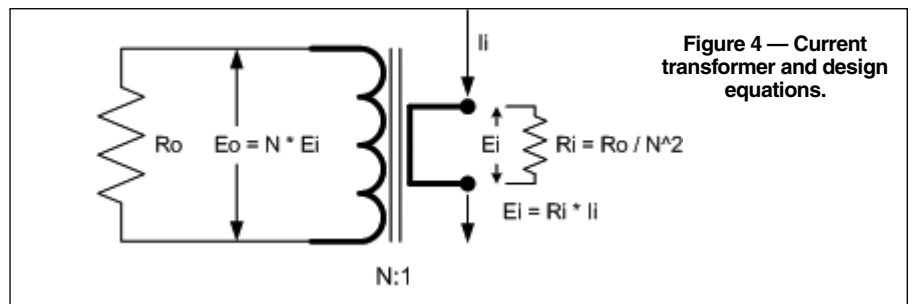
Figure 4 — Current transformer and design equations.

R_o $E_o = N * E_i$ E_i $R_i = R_o / N^2$

I_i

$E_i = R_i * I_i$

N:1

#5975000201 with an Al of 2200 nH/T². I checked the performance of the transformer by placing a known current through the one turn and measuring the output voltage with the spectrum analyzer. For this current transformer design, 1 µA loop current = 30 µV output. As with any current transformer design, careful selection of the core is necessary for proper operation at the frequency and impedance desired.

All voltage measurements were made with an HP E4411B spectrum analyzer.

The cable connecting the loop antenna to the spectrum analyzer is double-shielded RG-55 cable. This was important because common mode currents on a cable with a single braid cable can cause measurement errors.

Measurement of a One-Turn Loop

Using the dimensions of the one turn loop, the calculated impedance at 400 kHz was 8.7 Ω and the calculated impedance at 400 kHz using the inductance measured at 100 kHz with an inductance meter was 9.3 Ω. Using the measured loop voltage of 3.7 µV and current of 0.4 µA from the 400 kHz NDB signal, the calculated impedance is 9.25 Ω. The measured loop voltage and current from the 400 kHz NDB signal is used as the reference values to the multi-turn loops.

Measurements of the open circuit voltage and closed circuit current of the one turn reference loop were then used to calculate the expected voltage and current for the multi-turn loops.

Measurement of the Multi-Turn Loops

The multi-turn loop measurements were made using the same local 400 kHz NDB station. This process took awhile because of the time to rebuild the loop antenna for the increase in number of turns from 2 to 4. Every effort was made to ensure the antenna was placed in the same spot and measurements taken at the same time of day. Also, the reference antenna was re-measured between reconstructions of each multi-turn antenna to assure the system was stable and not varying due to equipment or set-up variation. The expected and measured results are shown in Table 1.

In Table 1, the measured voltage and current followed closely the expected values to illustrate the change in voltage or current as a function of the number of turns. It may be noted that the measured data is not of laboratory quality. However, the measured data does show the expected effect on the voltage and current with increase in the number of turns as predicted by the equations developed earlier in this article. That is, that the open circuit voltage will increase as a function

Table 1
Expected and Measured Results
(MC = Mutual coupled loop turns. NC = Non-mutual coupled loop turns)

	N=1 Turn Reference	N=2 Turns	N=3 Turns	N=4 Turns
$V_{oc}µV \times N$				
Expected	3.7 (ref)	7.4 (ref × 2)	11.1 (ref × 3)	14.8 (ref × 4)
$V_{oc}µV$ (MC)				
Measured		7.5	12	15
$V_{oc}µV$ (NC)				
Measured		7.2	10	14
$I_{oc}µA \div N$				
Expected	0.4 (ref)	0.2 (ref / 2)	0.13 (ref / 3)	0.1 (ref / 4)
$I_{oc}µA$ (MC)				
Measured		0.22	0.17	0.12
$I_{oc}µA \times 1$				
Expected	0.4 (ref)	0.4 (ref × 1)	0.4 (ref × 1)	0.4 (ref × 1)
$I_{oc}µA$ (NC)				
Measured		0.34	0.37	0.33

Table 2
Measurement: Mutually Coupled Loop Antenna

	1 Turn	2 Turns	3 Turns	4 Turns
$X_L \alpha N^2$ Impedance (Calculated from the measured inductance at 100 kHz)	9.3 Ω	32 Ω	65 Ω	113 Ω
Antenna: Volts/Amps	9.25 Ω	34 Ω	70 Ω	125 Ω

Table 3
Measurement: Non-Mutually Coupled Loop Antenna

	1 Turn	2 Turns	3 Turns	4 Turns
$X_L \alpha N$ Impedance (Calculated from the measured inductance at 100 kHz)	9.3 Ω	20 Ω	33 Ω	42 Ω
Antenna: Volts/Amps	9.25 Ω	21 Ω	25 Ω	45 Ω

Table 4
Effect of N (# turns)

	Mutual Coupled Turns	Non-Mutual Coupled Turns
V open circuit	× N	× N
I closed circuit	1 / N	----

of N and that the closed circuit current will decrease as a function of 1/N for a mutual coupled loop turns.

Another item of interest is to compare the calculated impedance of the inductance at 400 kHz with the impedance as calculated from the voltage and current measurements at 400 kHz from the NDB signal for 1 turn to 4 turns. Table 2 shows the comparison for the mutual coupled case and Table 3 shows the comparison for the non-mutual coupled case.

A summary of the effect on N on the voltage and current of a loop antenna is given in Table 4.

For most loop antenna applications, the loop is tuned to resonance with a capacitor, thus involving the Q factor. Increasing the Q of a loop antenna increases both the voltage and current in the loop since the Q of the loop is a multiplier to the open circuit and the short circuit equations. For the case of the open-circuit voltage, increasing both the number of turns and Q will increase the open-circuit voltage ($Q \times N$). For the case of closed-circuit current and mutual coupled turns, increasing Q will increase current, but the current would be diminished by the number of turns (1/N). For the case of closed-circuit current and non-mutual coupled turns, increasing Q will increase current, but the current would not be diminished by the number of turns (N). The aspect of Q is brought up because Q is usually large (typically ≥ 25) and can mask the effect of N. Thus, if the number of turns is being changed to affect Q, then the expected change in current from Q will be modified by the number of turns as well.

Loop antennas are more often used as receiving antennas, which to this point has been the focus. However, loop antennas are also used as transmitting antennas, so the question now arises of how the far field may change according to the number of turns in a transmitting loop. In the receiving loop equation, the magnetic field is the generator for the voltage or current. In a transmitting scenario we want to create a magnetic far field that is as large as practical. The equation to calculate the magnetic far field from a loop antenna is

$$H_o = \frac{\pi I_o N A}{r \lambda^2}$$

The far field equation is from *Antennas* by John D. Kraus.[1] From this equation, it can be noted that the far field increases directly with the increase in the number of turns, assuming that the current and area are held constant. Thus, if N=2, then the expected increase in the magnetic field is 6 dB.

As with the receiving loop, I built a transmitting test loop and measurement system to measure the far field with a one and two turn loop. I made two loops 280 mm square with

one and two turns. The loop current was generated with a current transformer connected to an HP3320B generator. The loop current was measured across a 0.1 ohm resistor connected in series with the loop. The voltage across the 0.1 ohm resistor was measured with an HP3400B voltmeter. By measuring the current in the loop, the generator level could be set to the same value of loop current for the one and two turn loop. The loops were made resonant with a tuning capacitor at the operating frequency of 12.50 MHz. The measuring active antenna was placed approximately 10 meters from the transmitting antenna loop.

The expected result from the transmitter case was a 6 dB increase of the received signal with the one turn loop reference to the two-turn loop. However, the initial measurement unexpectedly showed a much greater increase in signal over the one-turn loop.

The explanation was that the radiation efficiency had increased with the two-turn loop compared to the one-turn loop. A discussion of this topic can be found in the textbook *Antenna Theory Analysis and Design* by Constantine Balanis.[2] On page 171 Constantine states, "To increase radiation efficiency, multi-turn loops are often employed." I performed the calculation outlined in the section.

The basic equation for radiation efficiency is:

$$\eta = \frac{R_r}{R_r + R_L}$$

Where R_r = radiation resistance and R_L = ohmic loss in the radiation efficiency equation. Radiation resistance increases with N^2 and ohmic losses increase with N. Since radiation resistance increases at a faster rate "N^2" than ohmic losses "N", the radiation efficiency increases with an increase of the number of turns for a small loop antenna. The equation for radiation efficiency with the factor N is:

$$\eta = \frac{N^2 R_r}{N^2 R_r + N R_L}$$

The calculation yielded an improvement of 5 dB. Now the expected improvement is 6 dB for the N=2 and 5 dB for efficiency improvement for a total 11 dB increase of the transmitted signal. The experiment measured a received signal increase of 11.2 dB.

Conclusion

The area of a small loop antenna is generally the focus of design for achieving the best possible performance. However, area does have its practical limit and space may be restricted. In this case, modifying the number of turns may be an alternative to achieving acceptable performance. I have presented a perspective as to how the number of turns affects the performance of the antenna by developing the known mathematical equations to include "N" to show its relationship to loop current and radiation efficiency.

Virgil Leenerts, W0INK, is an ARRL member and an ARRL Technical Specialist for Colorado. He was licensed in 1954 and earned his BSEE from the University of Illinois in 1963. Virgil retired as an electrical engineer with Hewlett Packard and Agilent Technologies after 38 years of service, but continues as a part-time engineer for SCOM designing power supply and audio circuits. He has given many presentations to amateurs, including a discussion about switching power supplies at the 2009 ARRL Rocky Mountain Division convention.

Notes

[1] J. D. Kraus, *Antennas*, New York, McGraw-Hill Book Company, 1950, Chapter 6, pp. 155-157.

[2] C. A. Balanis, *Antenna Theory Analysis and Design*, Harper & Row, Publishers, Inc, 1982, Chapter 5, pp.169-173.

QEX–

Roger Paskvan, WA0IUJ

3516 Mill St NE, Bemidji, MN 56601, rogerp@paulbunyan.net
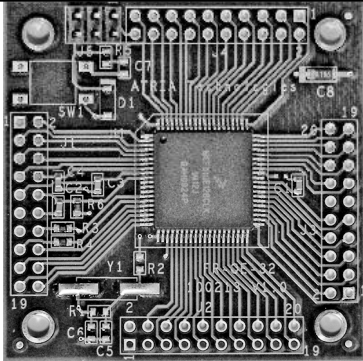
An All Purpose High Gain Antenna for 2400 MHz

Roger shows an easy to fabricate circular horn antenna that can be built using copper plumbing pipe and sheet copper.

For those interested in a relatively easy way to build antenna for the 2400 MHz band, the design described here might be the answer. It features high gain, good capture area, and shielding from strong local signals. The antenna will operate in either vertical or horizontal polarization as a function of mounting. Design data with complete construction and tuning procedures take the guesswork out of building this wonder antenna, out of a simple copper pipe.

This article provides a simple step-by-step procedure for the design and construction of a horn antenna. This horn will work as a stand-alone antenna (boasting nearly 9 db-d of gain) or a feed horn for a parabolic

Figure 1 – Side view of the completed horn antenna.

dish. The possibilities for using this antenna are limited by your imagination. Point to point communication of data and voice has been done over several miles. If you are lucky enough to have a shack in the back yard, two of these provide a great data link with the house. The author has not tried this antenna as a feed horn with a dish but there is no reason why it would not work. Attention should be given to f/d ratio and parabola illu-

mination in that application. With a simple hood addition, this copper wonder will provide 12 db of gain, now that's 17 times!

According to Kraus[1] a horn antenna is regarded as an opened up waveguide. The function of this arrangement is to produce an in-phase wave front thus providing signal gain in a given direction. Signal is injected into the waveguide by means of a small probe that must be critically placed. The type of horn described in this writing is known as a cylindrical horn. It was chosen for simplicity of construction utilizing available copper pipe.

[1]Notes appear on page 46.
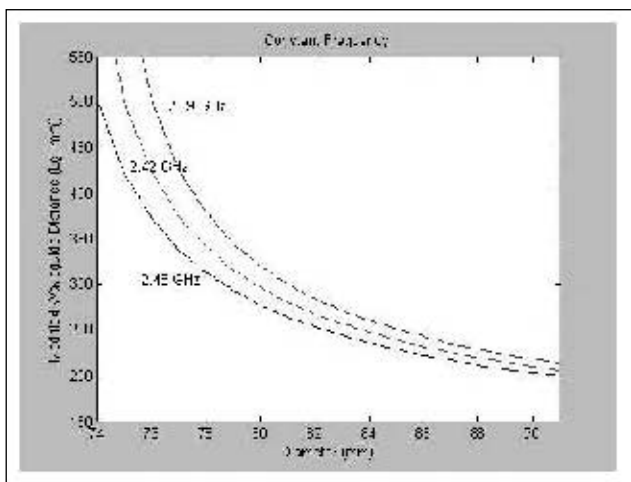
Figure 2 – Graph of Guide Wavelength (Lg) vs Horn diameter for 13 cm Band.
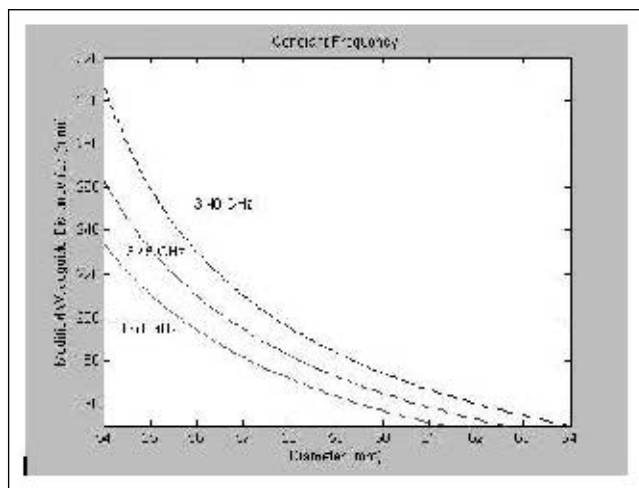
Figure 3 — Graph of Guide Wavelength (Lg) vs Horn diameter for 9 cm Band.
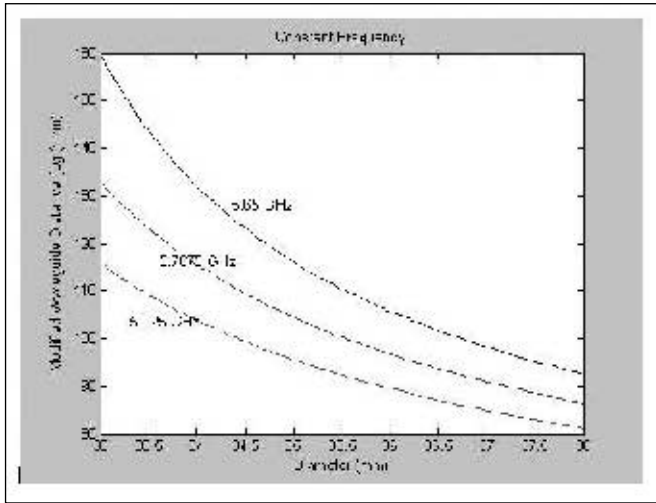
Figure 4 — Graph of Guide Wavelength (Lg) vs Horn diameter for 6 cm Band.

Figure 6 — Raw cylinder ready to be cut.
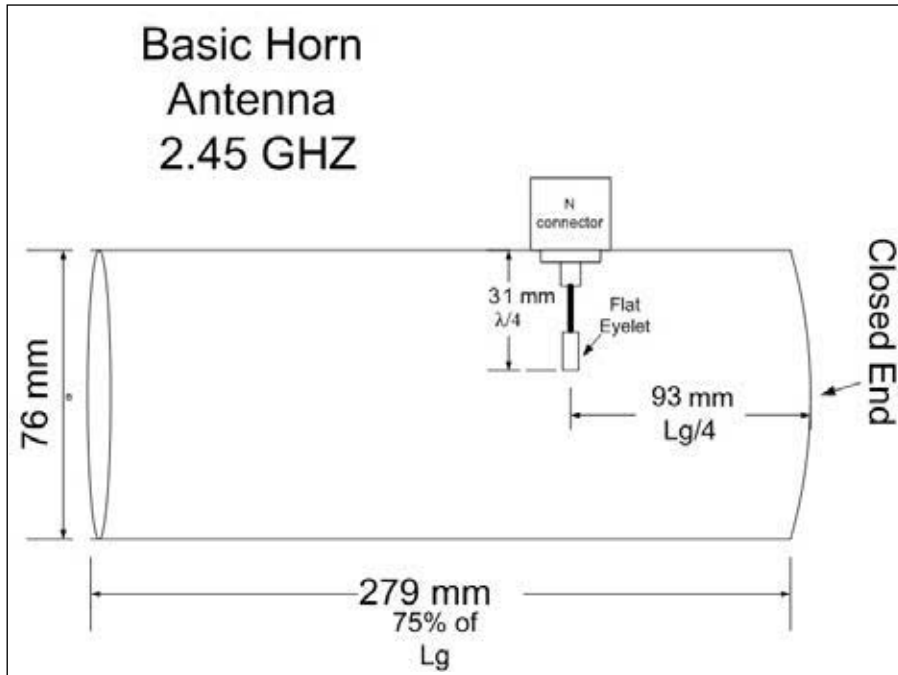
Figure 5 — Basic Horn Antenna Design.

Sometimes relationships that appear to be relatively simple are in-fact very complex. This is certainly the case for the horn antenna. First, only the cylinder diameter determines the lowest and highest frequency that can be used with this arrangement. They are referred to as the cut off frequencies because below and above performance deteriorates rapidly. Secondly, there is a critical optimum location for a probe that excites the waveguide horn cylinder. Finally, the length of the tube, shorted on one end, is a modified version of a wavelength, but not the free space wavelength.

What goes on inside a waveguide is com-

plex to say the least. Instead of propagating in straight lines, the energy bounces off the walls of the cylinder causing constructive interference due to multiple reflections. The wavelength inside the closed cylinder is a compressed version of a free space wavelength. The reasons for the compression are due to in-phase and group velocities within the cylinder. The critical placement of the injection probe reinforces these wave fronts within the pipe waveguide thus providing signal gain out of the open end of the cylinder. This cylindrical waveguide is physically closed on one end and can be thought of similarly to a shorted piece of coaxial cable.

Designing your Antenna

To begin our design, a suitable cylinder must be found. The author used three inch I.D. copper pipe since it was surplus from the local public TV station. Cut pieces of rigid hard line work well and so does copper plumbing pipe. Any type of cylinder will work including coffee cans, but thin walled tin does not seem too stable or surface conductive compared to copper. Larger diameter copper pipe sections can be obtained from plumbing and heating contractors usually working on larger projects. Many commercial boilers utilize 3-4 inch diameter copper pipe. There are always small cut-off pieces (under two feet) that could be available for the asking. No matter what material you want to use, it must pass the frequency cut-off tests. If the diameter is too large or too small, the pipe may not function in the waveguide mode at your frequency of choice. The lowest frequency where this cut-off phenomenon occurs is known as the horn low cut-off diameter.[2] Without getting into propagation modes, let's refer to this number as the low cut-off frequency. There is also a high cut-off frequency, but we will deal with that later. This low cut-off frequency can be found using the equation below.[3]

$$\lambda_C = 3.412 * r$$

λ_C is the low cutoff wavelength, and r is the radius of the cylinder. By dividing the constant in half, this formula (in terms of diameter) becomes:

$$\lambda_C \text{ (mm)} = 1.706 * \text{diameter (mm)} \qquad \text{[Eq 1]}$$

The low cut-off frequency is then found by using the standard wavelength formula.

$$F \text{ (GHz)} = 300/ \lambda_C \qquad \text{[Eq 2]}$$

The author chose to use metric measure-
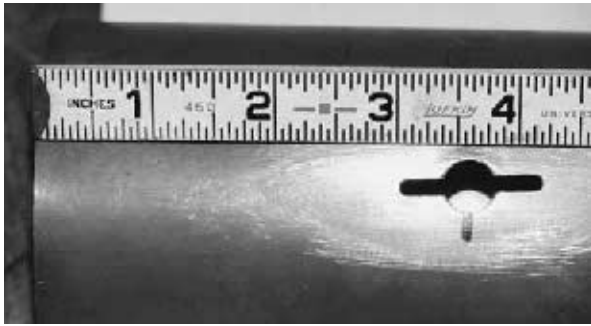
Figure 7 — Drilled opening for type "N" connector probe assembly. The slot was tried to be able to move the probe for best results. No difference was found and the slot was eliminated on later models.

ment for the equations not only for ease of calculations, but also for use in an evaluation program where metric units are used.

The low cut-off frequency is only a function of the diameter of the pipe, therefore, measure each pipe's inside diameter and crank out the above equations (one and two) until the cutoff frequency (λ_C) is slightly lower than the frequency range that you are planning to operate. Step one is to calculate the low frequency cut-off. In the author's particular case, the 76 mm (3 inches) diameter copper pipe from equation one became:

$\lambda_C = 1.706 * 76mm$
$\lambda_C = 130$ mm
$F = 2.31$ GHz $= 300/130$

This is the lowest critical frequency that the 76 mm diameter pipe will operate. Anything below 2.31 GHz and the pipe will stop functioning like a waveguide.

In step two, we must calculate the highest cut off frequency (λ_H) that this diameter pipe will operate as a waveguide antenna.[4]

$\lambda_H = 1.3065 *$ diameter (mm)

Using the author's 76 mm example:

$\lambda_H = 1.3065 * 76$ mm

$\lambda_H = 99.3$ mm is the critical upper wavelength of this 76 mm pipe. Using equation two again, we find that critical upper frequency

Frequency (GHz) $= 300/\lambda_H$
$F = 300/99.3$

Upper critical frequency for λ_H is 3.02 GHz.

You now have the critical lower and upper frequencies where your given pipe will operate as a circular waveguide antenna. In the author's case, this was 2.31 GHz through 3.02 GHz. This pipe will work for the 2.39-2.45 GHz microwave bands. Repeat this process for every diameter pipe you measure until you find one that will operate in the frequency range you desire. The lower cut-off must be below the frequency on which you want to operate. The high cut-off must be above the
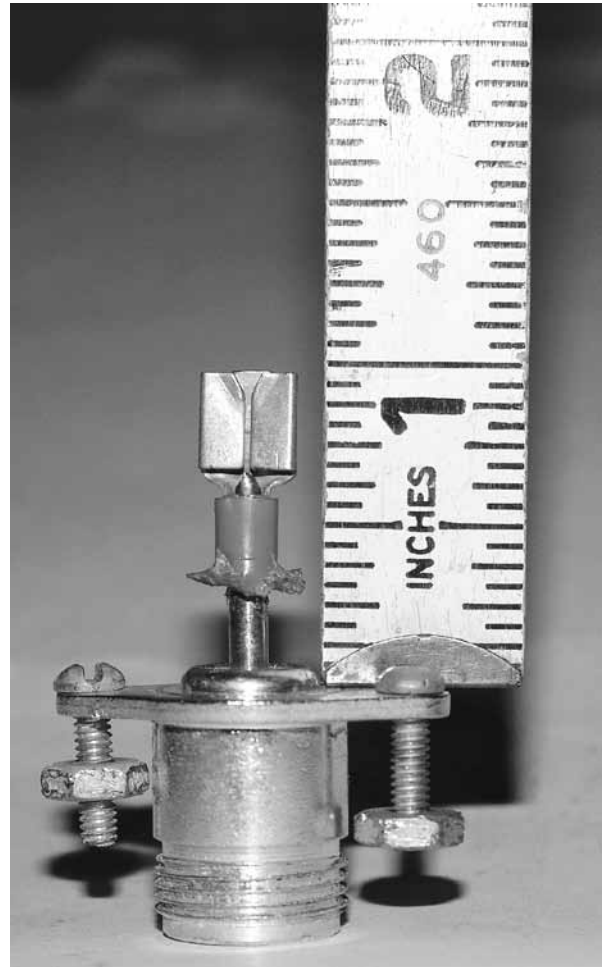
Figure 8 — Probe details including the flat eyelet soldered to the #12 wire on the "N" connector. Carefully measure to the end of the eyelet.

Figure 9 — Materials ready for assembly. The cut cylinder, copper plate for closing the end of the pipe, and the probe assembly.
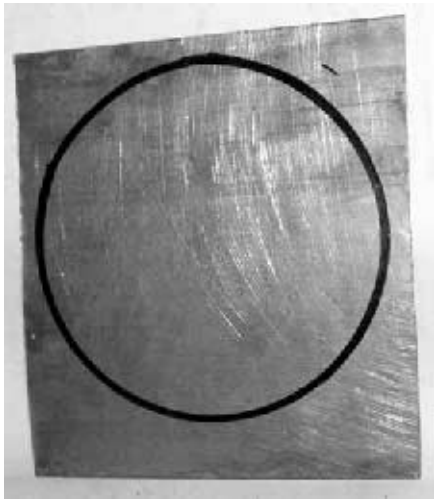
Figure 10 — Using the cylinder end, scribe a circle onto the PC board.

Figure 11 — PC board soldered to the end of the cylinder. The corners need not be trimmed since it will not affect the operation of the antenna.
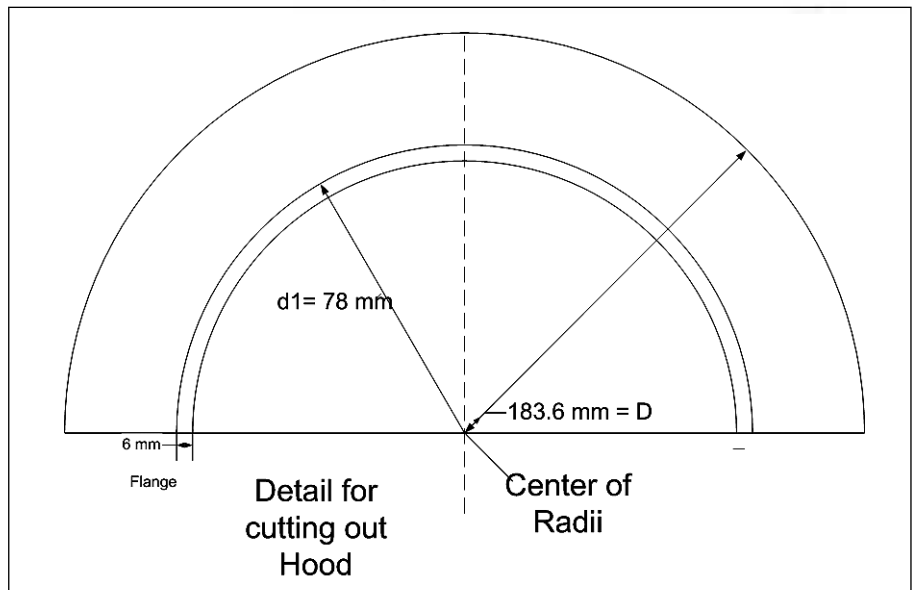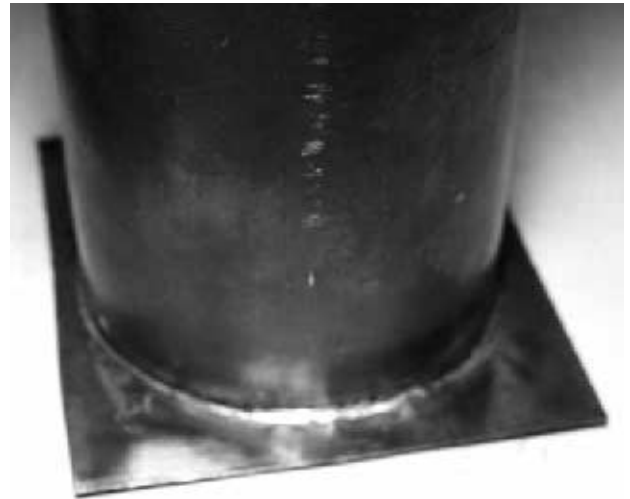
Figure 12 — Layout the hood radius as indicated. Note all semicircles come from the same point. The inner most radius is a flange for soldering.

frequency on which you want to operate.

We have to choose a design frequency within this critical frequency range that you intend to operate this antenna. Let's decide on a design frequency of 2.45 GHz for our example. It is at the top end of the 2.4 GHz band and allows some crossover into the Wi-Fi world. In step three, we calculate free space wavelength for your design frequency:

$$\lambda = 300/\text{Freq (GHz)} \qquad \text{[Eq 3]}$$

$\lambda = 300/2.45 = 122$ mm

Next, in step four, the overall cylinder length of the horn antenna must be calculated. The traveling waves inside the tube travel slower than the speed of light. Therefore, a slightly more complex formula must be used to figure the physical length dimensions of the tube. This modified wavelength distance will be called Lg, the wavelength inside the waveguide. The cylinder length is cut to 75% of Lg.[5]

$$L_g = \frac{1}{\sqrt{\left[\frac{1}{\lambda}\right]^2 - \left[\frac{1}{\lambda c}\right]^2}} \qquad \text{[Eq 4]}$$

$\lambda = 122$ mm
$\lambda_C = 130$ mm
Lg = 372 mm

cylinder length = 0.75 * Lg   [Eq 5]

The cylinder length should be 279 mm which is about 11 inches long and closed at one end. If you don't like the inverse math, Lg can be obtained from the graphs in Figures 2 through 4.

Now that we know the critical dimensions of your ready-made cylindrical waveguide, some means of feeding this antenna is needed. The behavior of microwaves inside the waveguide is similar to that of a shorted

coaxial cable. The excited signal inside the cylinder reflects off the closed end of the tube and sets up standing waves that travel through the cylinder toward the opening and are radiated out. If a means of putting some sort of voltage measuring device (signal probe) inside along the length of the cylinder were set up, we would see points of signal maximums and minimums. Starting at the closed end, the voltage would be zero and maximum at odd quarter wave intervals all along the distance of Lg. If a probe is put into the cylinder at one quarter of the guide wavelength from the closed end, this would be the best place to excite this waveguide antenna. The probe is placed at the first signal maximum from the closed end. Now, we have to find this distance.[6]

Calculating the distance so a suitable "N" connector can be installed on the side of the cylinder is done with the equation below.[4]

Probe Distance Lg/4 = 372/4 = 93 mm

[Eq 6]

An "N" female chassis connector will be mounted on the side of the cylinder at 93 mm from the closed end with a stiff 12 AWG wire soldered to the connector's center conductor. This wire is a quarter wavelength in free space at the operating frequency 2.45 GHz.

Probe Depth = $\lambda/4$ = 122/4 = 31 mm

[Eq 7]

The necessary calculations have now been completed. Our final dimensions are as follows:

Frequency 2.45 GHz (design frequency)
Free space wavelength λ = 122 mm
Wavelength inside tube Lg = 372 mm
Cylinder length   ¾ Lg   = 279 mm
Probe distance   Lg/4   = 93 mm
Probe depth $\lambda/4$ = 31mm
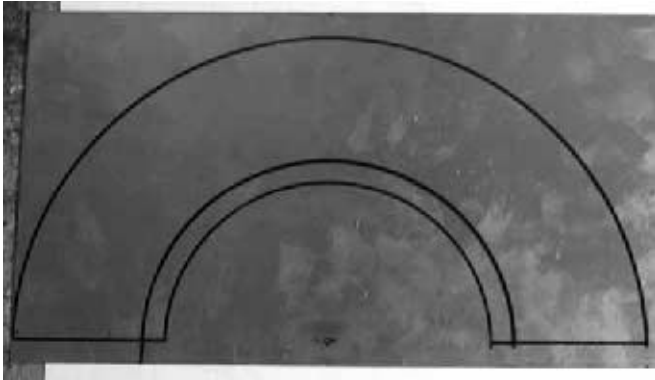
Figure 13 — Scribe the semicircles onto a copper sheet to be cut-out to form the hood.
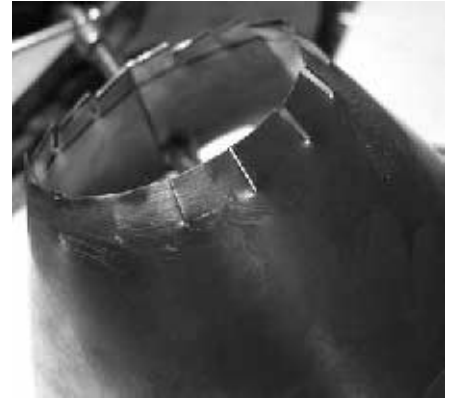
Figure 14 — Cut out segments around the inner most semicircle to aid in soldering to the outer cylinder.

So now that the hard work is done, let's start construction of this new horn antenna. Since the diameter of your cylinder was determined by the cut-off equations in steps 1 and 2, the tube you select is fixed for the frequency of interest. In our example the diameter was 76 mm. The next step in construction is to cut this cylinder to the correct length which is 279 mm. Utilizing a hack saw, cut the cylinder to length. (See Figure 6)

Next, the hole for the probe should be measured and drilled. In the prototype, the end plate was soldered first but it was quickly learned that this only made it very difficult to put the screws on the connector inside the cylinder. First, measure the quarter wave distance from the closed end of the tube to the center of your "N" connector. This would be the probe distance of 93 mm. Mark this point and drill out the hole starting with a pilot drill then widen the opening to fit the inner edge of the "N" connector. In the prototype, a slot was made to move the connector for best output. After testing, there was no improvement in signal within a quarter inch of the exact quarter wave point. (See Figure 7)

At this point, the probe must be carefully fabricated to fit into the cylinder hole with the exact length specified. In our example, this would be 31 mm which is the result of equation 7 (see Figure 8). A small #12 wire will do an excellent job. Note the small crimp connector soldered to the top of this wire. In experimenting with the probe depth and SWR, the flat crimp-on connector improved the match over a plain wire. Please make sure that the overall length is exact to the end of the crimp connector and solder all connections. Everything is ready for assembly. (See Figure 9) Now place this entire "N" connector assembly into the hole and secure with bolts and nuts. Note that the bolt head should be inside the cylinder and the nut on the outside. When things are secure, it is time to solder the end plate.

Scribe the cylinder end onto a flat copper stock or PC board. (See Figure 10) Place the cylinder over the plate and solder neatly. (See Figure 11) The corner excess copper can be

trimmed after the plate has cooled but leaving the plate square will have no negative effect on the operation of the antenna. It may take quite a bit of heat.

The antenna is now ready for testing and can be used in its present form. It will boast a good match and offer about 9 db of gain with a good directional pattern. Why not double your signal output with just one small addition to this antenna, a capture hood. A capture hood is not necessary for operation and the antenna will work quite well as a straight cylinder. But, if you want to increase the gain of your antenna by 3 dB, add this funnel to the front of your cylinder to increase the capture area and double your signal output. You are effectively collecting signal from a larger area to provide additional signal gain. Think of this as a funnel collecting rain water into a barrel. You will always get more water with a funnel.

So the question becomes how large does this funnel need to be to fit over the end of the cylinder? We go back to the drawing board

for a moment and some geometry to find the right size hood for this particular frequency and antenna. What needs to be done is calculating the radius and diameter of a copper flange that will be soldered to the front of your antenna. The actual hood will end up as a curved copper strip soldered to the outside of the antenna's open end.

Calculating the Hood size

Step one is to determine D, the outer diameter of the hood. The outer diameter of the hood should be 1.5 times the free space wavelength λ.[7] The previously calculated value of λ at 2.45 GHz is 122 mm.

$$D = 1.5 * \lambda \text{ or } 1.5 * 122 \text{ mm} = 184 \text{ mm}$$

D1 is the measured outside diameter of the cylinder or 76mm(id) plus the thickness of the copper pipe times two in our case. D1
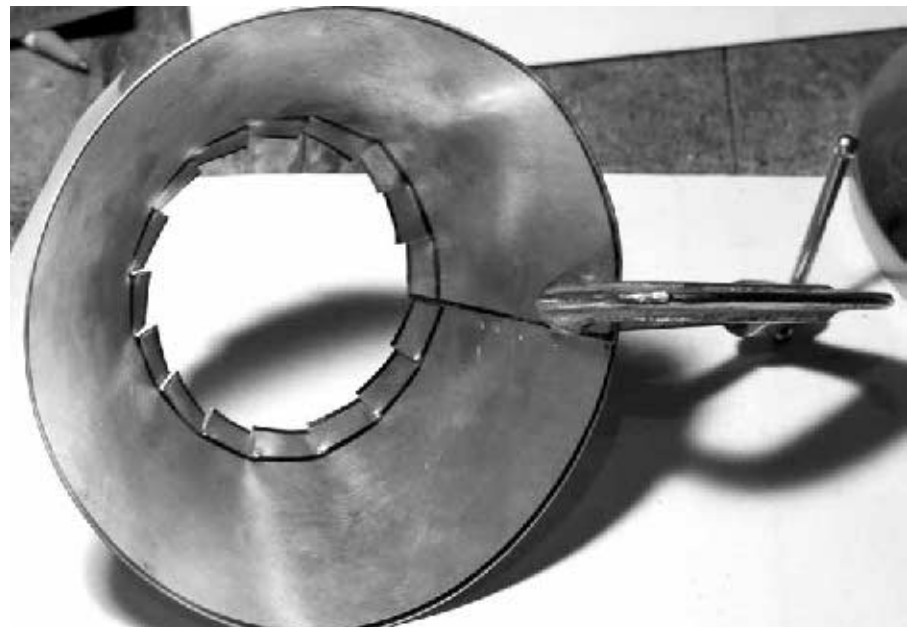
Figure 15 — Bend the flange into a circle, holding with vise grips, solder the seam.

Figure 16 — Cylinder with flange fitted over pipe ready to solder. This may take quite a bit of heat.

is 78 mm. Step two is to draw out two radii on a sheet of copper. Make D the radius of the outer hood semicircle and D1 is the radius of the inner semicircle. Using a compass, measure from a reference line 184 mm and draw the outer arc on the copper sheet. Along the same reference line, measure 78 mm and draw the inner arc. Move down below this inner arc 6 mm and draw a small arc. This will allow about 6 mm (¼") tabs on the shorter radius for soldering to the outer cylinder end. (See Figures 12 and 13). Cut out the outer and inner lines of the copper hood with tin snips. Make small strip cuts into the 6 mm (¼") flange to aid in soldering the hood to the cylinder. (See Figure 14) Roll the strip into a semi-circle. Hold the flange together with vice grips and solder the strip into a circle. (See Figure 15)

When you are ready, mold the flange over the end of your cylinder and carefully solder the tabs onto the cylinder. (See Figure 16) It may take a lot of heat since copper conducts very well. Your flange hood is now part of the antenna and you are almost ready to test the new antenna. The completed horn antenna with hood is diagrammed in Figure 17.

To complete the project, some form of mount is required to attach your high gain horn antenna to a tower or building. After some consideration, bending a one inch width of aluminum stock worked out well. Bend the flat stock over the horn, then around a "form", the same diameter as your tower leg. Drill several holes for mounting hardware and wing nuts. You now have a ready-made custom horn mounting bracket. (See Figure 18) A suitable cover for the mouth of this antenna can be found from any plastic cap that will fit and pass the microwave test. After one minute in the microwave oven, there should be very little heat internal to the cover.
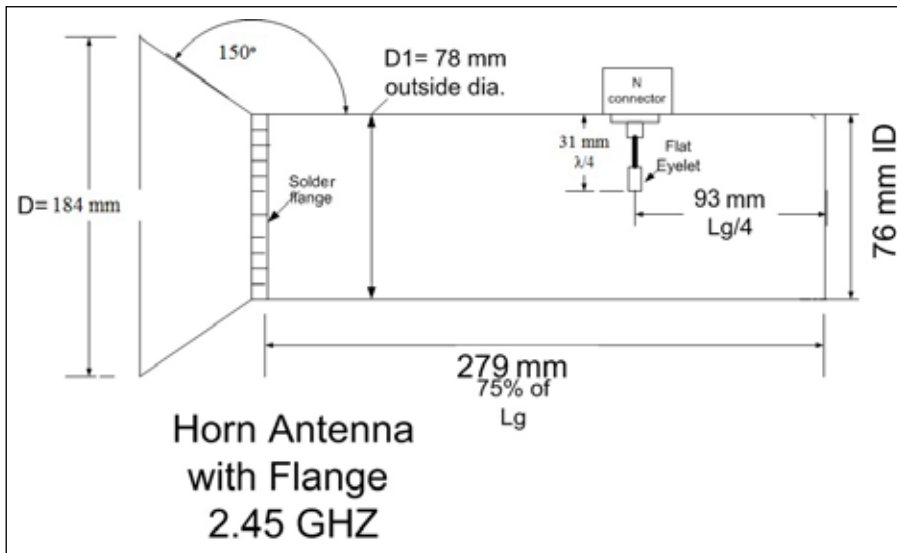
Figure 17 – Diagram of Complete Hooded Horn antenna.

Figure 18 — A practical yet easy to make a mounting bracket for the horn antenna.

Testing and Measurements

Once you have finished your horn antenna, it is time to see it operate in action. The author had access to an HP 8714 Network analyzer that has 3 GHz capability. Stand the horn upright with the opening away from all metal surfaces by at least 4 feet. (See Figure 19) Connecting the horn antenna for return loss measurement provided a really nice curve showing a very good match of 1.08 SWR at 2.45 GHz. The useable SWR bandwidth under 1.6:1 was 2.40 GHz to 2.51 GHz. Putting your hand over the front of your antenna hood will cause the reading to change drastically. It is a crude indication that there is an energy field focused in front of the horn. The author also verified these measurements with a 1W transmitter operating on 2.44 GHz using a Bird 43 wattmeter and a 2.5 GHz slug. (See Figure 20)

To show some of the other parameters of this antenna, the author employed a little known program called *SABOR*. This program is a product of the Universidad Politécnica de Madrid Engineering department and fortunately is offered as freeware on the internet. The program was initially written to evaluate several types of waveguide horn antennas. It is perfect for calculating maximum gain, bandwidth, phase center and generating a plotted pattern of your antenna for evaluating the power gain at some beam width.

To use *SABOR*, go to **www.gr.ssr.upm. es/sabor.htm** and download the freeware version of the program, *SABOR* 1.0. The program can be downloaded as freeware but if you're going to use it a lot, the University asks for a $25 donation. The files will unzip into a folder. Click on the blue icon, SABOR.EXE.

The program opens up with a welcome

page from Madrid, Spain. Hit the OK button at the bottom. You will now see many horn antenna designs. The program covers six different types of horn designs. You will need to choose what type of design you want. The program is written in centimeters so conversion from mm to cm is necessary. Use the upper menu icons along the top of the open-

ing page to navigate the program. The initial page should be a horn design. If not, go under the menu and select HORN. Under horn, you will be viewing three options in a drop down menu. We want to select CIRCULAR. You will now see the screen change to a conical circular diagram similar to the one we just built. Next move over on the top toolbar to

Figure 19 — Test set up utilizing the HP 8714 Network analyzer.
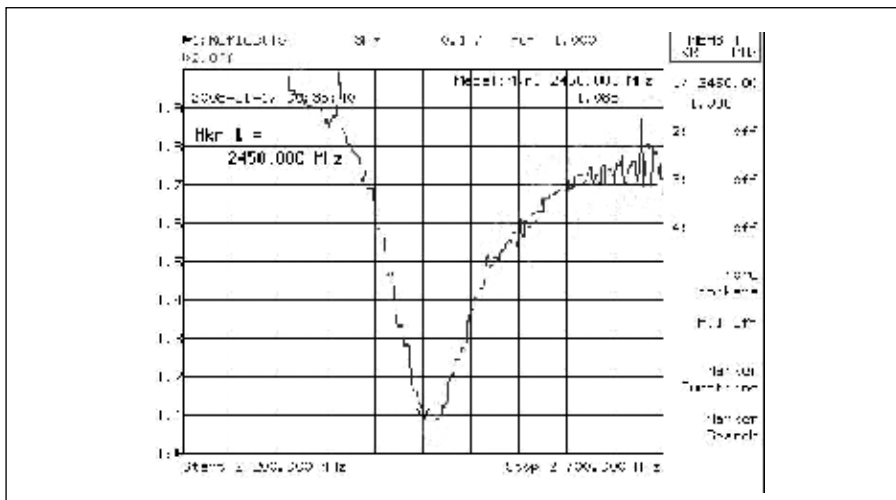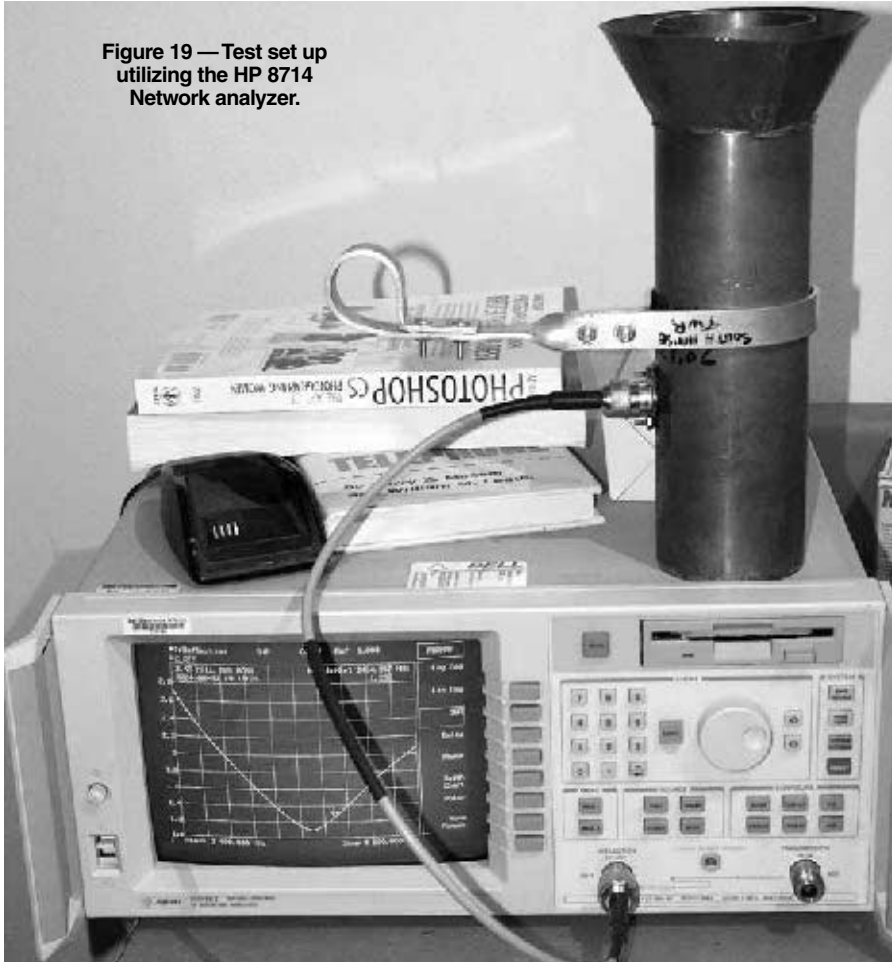
Figure 20 — Actual screen shot of SWR bandwidth boasting 1.086 at 2.45 GHz.
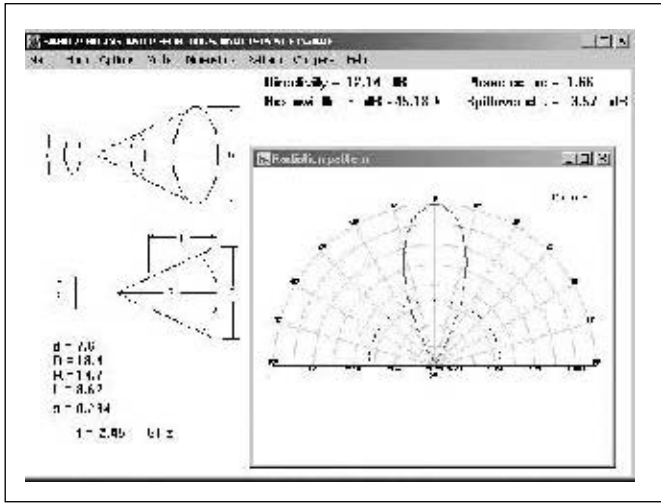
Figure 21 — Polar Plot of the radiation pattern from the *SABOR* program.
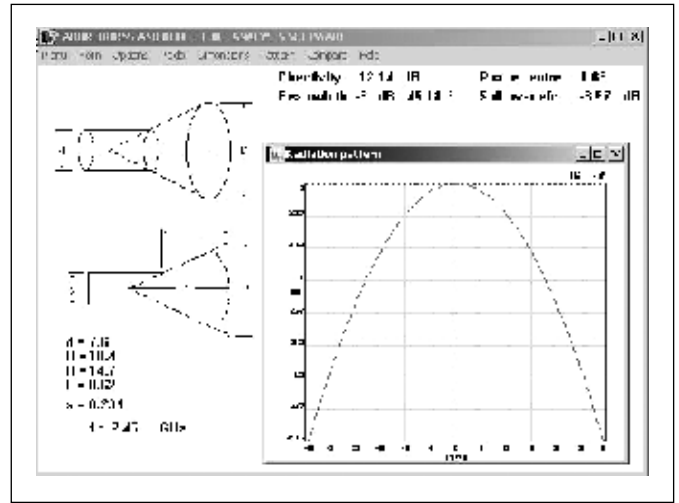
Figure 22 — SABOR X-Y Plot of the pattern. Gain is 12.14 db and the 3 db beam width is 45 degrees.

OPTIONS. The drop down menu will show frequency on top. Click on FREQUENCY and enter the operating frequency. In our example, the author chose 2.45 GHz. Move over on the top menu to DIMENSIONS. Several parameters are requested but we have already determined two of the three. *SABOR* asks for Wave-g, Horn radius and a radius value for R1. Knowing our example hood dimensions, D1 = 76 mm, D = 184 mm, will be all that is necessary to utilize *SABOR*.

Wave-g is half the cylinder diameter (d = 76 mm). In our case, 76 mm/2 = 38 mm or 3.8 cm. Horn Radius is half of the hood diameter (D = 184 mm) and in our case 184/2 = 92 mm or 9.2 cm. R1 is found by D/1.25, so it is 184/1.25 = 147 mm or 14.7 cm. Enter waveg = 3.8 cm. (Refer to the left side of Figure 21 to see these values in *SABOR*.) Given this information, the pattern and gain figures can now be calculated. Move back to the OPTIONS drop down menu and click AUTOMATIC and xy. Now move to the PATTERN icon and click. This is the X-Y plot of your antenna. The gain for our example is over 12 db and the beam width is 45 degrees. To see a polar plot of your antenna, go back to the OPTIONS drop down menu and click POLAR. The display will now change to the familiar directional pattern. (See Figures 21 and 22.)

Applications.

This antenna is currently in use to link the author's radio shack with the home, a distance of 600 feet. A pair of these horn antennas can easily link data services using 10 mW access points. The antenna can be used for any 2.4 GHz point to point communication. A 1W Amateur television signal has been sent over a pair of these horn antennas at a distance of six miles with good A5 results.

Table 1
Standard Type L Copper Pipe Dimensions (Inches)

| Nominal Size | Actual Dimensions | |
Inside Diameter	Outside Diameter	Wall Thickness
¼	0.375	0.035
⅜	0.50	0.049
½	0.625	0.049
⅝	0.750	0.049
¾	0.875	0.065
1	1.125	0.065
1¼	1.375	0.065
1½	1.625	0.072
2	2.125	0.083
2½	2.625	0.095
3	3.125	0.109
3½	3.625	0.120
4	4.125	0.134
5	5.125	0.160
6	6.125	0.192
8	8.125	0.271

This horn antenna could be utilized to illuminate a dish antenna. Polarization is a function of the direction of the probe. Up and down is vertical and 90 degrees to that is horizontal. Your imagination is the limiting factor in applications for this antenna.

Here are dimensions for a 2.40 GHz version of this horn antenna using a 76 mm diameter copper pipe. The cut off frequencies remain the same since the diameter is 76 mm.

Design Frequency is 2.40 GHz
¾ = 125 mm
Lg = 470 mm
Cylinder length = 352 mm
Probe distance from closed end = 118 mm
Probe depth = 31 mm
Hood dimensions D = 188 mm, D1 = 78 mm

There are potential designs for 5 GHz utilizing 38 mm (1.5 inch) diameter copper pipe and 3.4 GHz using 2 inch diameter pipe. Let's experiment. You get a lot of gain for a small amount of effort and the horn antenna is very directional with a clean pattern. To aid in the hunt for copper tubing, Table 1 shows typical ID and OD sizes for commercial plumbing pipe.

The author would like to express his gratitude to Dustin Larson who fabricated prototypes of this antenna during the development and testing phases and Jeremy Vogel for his assistance in the graphing programs.

Notes

[1]John Kraus, *Antennas*, McGraw-Hill, 1988, pp 644-653.
[2]Gershon Wheeler, *Introduction to Microwaves*, Prentice-Hall, 1963, pp 60-63.
[3,4]Jessop, G. R., *VHF-UHF Manual*, 4th edition, Radio Society of Great Britain, 1985, p 9.27.
[5]Barter, Andy, *International Microwave Handbook*, Radio Society of Great Britain, 2002, p 58.
[6]Norm Foot, "Cylindrical Feed Horn," *Ham Radio*, May 1976, p18.
[7]Richard Kolbly, "Low Cost Microwave Antenna," *Ham Radio*, November 1969, p 52.

Photos by the author
Roger Paskvan was first licensed as WA0IUJ in 1961. He holds a Masters Degree and is a 34 year Associate Professor of Communications at Bemidji State University, Bemidji, MN.
*He is a practicing Broadcast Engineer and owns several broadcast stations. You can contact him at **rogerp@paulbunyan.net.***

QEX-

2010 *QEX* Index

Upcoming Conferences

Western Regional Conference 2011 Society of Amateur Radio Astronomers

Call for Papers

The Society of Amateur Radio Astronomers (SARA) hereby solicits papers for presentation at its 2011 Western Regional Conference, to be held April 2-3, 2011, at Embry-Riddle Aeronautical University in Arizona. Papers on radio astronomy hardware, software, education, research strategies, philosophy, observing efforts and methods are welcome.

SARA members or supporters wishing to present a paper should email a letter of intent, including a proposed title and informal abstract or outline (not to exceed 100 words) to the SARA vice president at **westernconference@radio-astronomy.org**, no later than December 15, 2010. Be sure to include your full name, affiliation, postal address, and email address, and indicate your willingness to attend the conference to present your paper. Submitters will receive an email response, typically within one week, along with a request to proceed to the next stage, if the proposal is consistent with the planned program.

A formal Proceedings will be published in conjunction with this meeting. Papers will be peer-reviewed by a panel of SARA members with appropriate professional expertise and academic credentials. First-draft manuscripts must be received no later than January 18, 2011, with feedback, acceptance, or rejection emails to be sent within two weeks thereafter. Upon final editing of accepted papers, camera-ready copy will be due not later than February 21, 2011. Due to printer's deadlines, manuscripts received after that deadline will not make it into the Proceedings. Instructions for preparation and submission of final manuscripts appear in a *Guidelines for Submitting Papers* document on the SARA website.

For more information go to **www.radio-astronomy.org.**

2011 Southeastern VHF Society Conference

Call for Papers

The Southeastern VHF Society is calling for the submission of papers and presentations for the upcoming 15th Annual Southeastern VHF Society Conference to be held in Huntsville, Alabama April 29-30, 2011. Papers and presentations are solicited on both the technical and operational aspects of VHF, UHF and Microwave weak signal amateur radio. Some suggested areas of interest are:

Transmitters; Receivers; Transverters; RF Power Amplifiers; RF Low Noise Pre Amplifiers; Antennas; Construction Projects; Test Equipment And Station Accessories; Station Design And Construction; Contesting; Roving; DXpeditions; EME; Propagation (Sporadic E; Meteor Scatter; Troposphere Ducting; etc.); Digital Modes (WSJT; etc); Digital Signal Processing (DSP);Software Defined Radio (SDR); Amateur Satellites ;Amateur Television

In general papers and presentations on non weak signal related topics such as FM repeaters and packet will not be accepted but exceptions may be made if the topic is related to weak signal. For example, a paper or presentation on the use of APRS to track rovers during contests would be considered.

The deadline for the submission of papers and presentations is March 11, 2011. All submissions for the proceedings should be in Microsoft Word (.doc). Submissions for presentation at the conference should be in PowerPoint (.ppt) format, and delivered on either a USB memory stick or CDROM or posted for download on a web site of your choice.

Pages are 8½ by 11 inches with a 1 inch margin on the bottom and ¾ inch margin on the other three sides. All text, drawings, photos, etc, should be black and white only (no color).

Please indicate when you submit your paper or presentation if you plan to attend the conference and present your paper in person, or if you are submitting solely for publication. The technical program is being handled jointly by Robin Midgett, K4IDC and Steve Kostro, N2CEI. Send all questions and comments to Robin at **k4idc@comcast.net**. Send all presentations to Steve at **SVHFS2011@downeastmicrowave.com**.

2011 Annual Conference, Society of Amateur Radio Astronomers

Call for Papers

The Society of Amateur Radio Astronomers (SARA) hereby solicits papers for presentation at its 2011 Annual Meeting and Technical Conference, to be held June 26 through June 29, 2011, at the National Radio Astronomy Observatory (NRAO), Green Bank WV. Papers on radio astronomy hardware, software, education, research strategies, and philosophy are welcome.

SARA members or supporters wishing to present a paper should email a letter of intent, including a proposed title and informal abstract or outline (not to exceed 100 words) to the SARA vice president at **vicepres@ radio-astronom.org**, no later than March 1, 2011. Be sure to include your full name, affiliation, postal address, and email address, and indicate your willingness to attend the conference to present your paper.

Submitters will receive an email response, typically within one week, along with a request to proceed to the next stage, if the proposal is consistent with the planned program. A formal Proceedings will be published in conjunction with this Meeting. Papers will be peer-reviewed by a panel of SARA members with appropriate professional expertise and academic credentials. First-draft manuscripts must be received no later than April 1, 2011, with feedback, acceptance, or rejection emails to be sent within two weeks thereafter. Upon final editing of accepted papers, camera-ready copy will be due not later than May 1, 2011.

Due to printer's deadlines, manuscripts received after that deadline will not make it into the Proceedings. Instructions for preparation and submission of final manuscripts appear in a Guidelines for Submitting Papers document on the SARA website. The last three year's Proceedings were a landmark accomplishment for our organization. Please help the Society of Amateur Radio Astronomers to make the 2011 edition even better!

For more information go to **www.radio-astronomy.org.**

QEX

Page; Uncompressing Linux Archives; Transferring files to uBoot; Linux and Software Development Computers; DSP Software Execution on the Stamp; Run Time Loading of Drivers in Linux; Simple Hardware Testing; A Bare Metal Test Program; On the Web and Next Installment (Mack): May/Jun, p 41

Tech Notes

Analog Devices 8307 (Kopski): May/Jun, p 44

More on Phase Controlled Differential Drive (or) Outphasing Modulation of High Efficiency Amplifiers (Cripe): Jan/Feb, p 35

More on Phase Controlled Differential Drive (or) Outphasing Modulation of High Efficiency Amplifiers (Hamel): Jan/Feb, p 35

Upcoming Conferences

14th International EME Conference: May/Jun, p 46

2010 Southeastern VHF Society Conference: Jan/Feb, p 38

44th Annual Central States VHF Society: Jan/Feb, p 40; May/Jun, p 46; Jul/Aug, p 45

AMSAT Satellite Space Symposium and Annual Meeting: Jul/Aug, p 46; Sep/Oct, p 47

Microwave Update 2010: Sep/Oct, p 47

Society of Amateur Radio Astronomers: Jan/Feb, p 40; May/Jun, p 46

The 14th International EME Conference: Jul/Aug, p 45

The 29th Annual ARRL and TAPR Digital Communications Conference: May/Jun, p 46; Jul/Aug, p 45; Sep/Oct, p 47

Upcoming Conferences

Western Regional Conference 2011 Society of Amateur Radio Astronomers

Call for Papers

The Society of Amateur Radio Astronomers (SARA) hereby solicits papers for presentation at its 2011 Western Regional Conference, to be held April 2-3, 2011, at Embry-Riddle Aeronautical University in Arizona. Papers on radio astronomy hardware, software, education, research strategies, philosophy, observing efforts and methods are welcome.

SARA members or supporters wishing to present a paper should email a letter of intent, including a proposed title and informal abstract or outline (not to exceed 100 words) to the SARA vice president at **westernconference@radio-astronomy.org**, no later than December 15, 2010. Be sure to include your full name, affiliation, postal address, and email address, and indicate your willingness to attend the conference to present your paper. Submitters will receive an email response, typically within one week, along with a request to proceed to the next stage, if the proposal is consistent with the planned program.

A formal Proceedings will be published in conjunction with this meeting. Papers will be peer-reviewed by a panel of SARA members with appropriate professional expertise and academic credentials. First-draft manuscripts must be received no later than January 18, 2011, with feedback, acceptance, or rejection emails to be sent within two weeks thereafter. Upon final editing of accepted papers, camera-ready copy will be due not later than February 21, 2011. Due to printer's deadlines, manuscripts received after that deadline will not make it into the Proceedings. Instructions for preparation and submission of final manuscripts appear in a *Guidelines for Submitting Papers* document on the SARA website.

For more information go to **www.radio-astronomy.org.**

2011 Southeastern VHF Society Conference

Call for Papers

The Southeastern VHF Society is calling for the submission of papers and presentations for the upcoming 15th Annual Southeastern VHF Society Conference to be held in Huntsville, Alabama April 29-30, 2011. Papers and presentations are solicited on both the technical and operational aspects of VHF, UHF and Microwave weak signal amateur radio. Some suggested areas of interest are:

Transmitters; Receivers; Transverters; RF Power Amplifiers; RF Low Noise Pre Amplifiers; Antennas; Construction Projects; Test Equipment And Station Accessories; Station Design And Construction; Contesting; Roving; DXpeditions; EME; Propagation (Sporadic E; Meteor Scatter; Troposphere Ducting; etc.); Digital Modes (WSJT; etc); Digital Signal Processing (DSP);Software Defined Radio (SDR); Amateur Satellites ;Amateur Television

In general papers and presentations on non weak signal related topics such as FM repeaters and packet will not be accepted but exceptions may be made if the topic is related to weak signal. For example, a paper or presentation on the use of APRS to track rovers during contests would be considered.

The deadline for the submission of papers and presentations is March 11, 2011. All submissions for the proceedings should be in Microsoft Word (.doc). Submissions for presentation at the conference should be in PowerPoint (.ppt) format, and delivered on either a USB memory stick or CDROM or posted for download on a web site of your choice.

Pages are 8½ by 11 inches with a 1 inch margin on the bottom and ¾ inch margin on the other three sides. All text, drawings, photos, etc, should be black and white only (no color).

Please indicate when you submit your paper or presentation if you plan to attend the conference and present your paper in person, or if you are submitting solely for publication. The technical program is being handled jointly by Robin Midgett, K4IDC and Steve Kostro, N2CEI. Send all questions and comments to Robin at **k4idc@comcast.net**. Send all presentations to Steve at **SVHFS2011@downeastmicrowave.com**.

2011 Annual Conference, Society of Amateur Radio Astronomers

Call for Papers

The Society of Amateur Radio Astronomers (SARA) hereby solicits papers for presentation at its 2011 Annual Meeting and Technical Conference, to be held June 26 through June 29, 2011, at the National Radio Astronomy Observatory (NRAO), Green Bank WV. Papers on radio astronomy hardware, software, education, research strategies, and philosophy are welcome.

SARA members or supporters wishing to present a paper should email a letter of intent, including a proposed title and informal abstract or outline (not to exceed 100 words) to the SARA vice president at **vicepres@ radio-astronom.org**, no later than March 1, 2011. Be sure to include your full name, affiliation, postal address, and email address, and indicate your willingness to attend the conference to present your paper.

Submitters will receive an email response, typically within one week, along with a request to proceed to the next stage, if the proposal is consistent with the planned program. A formal Proceedings will be published in conjunction with this Meeting. Papers will be peer-reviewed by a panel of SARA members with appropriate professional expertise and academic credentials. First-draft manuscripts must be received no later than April 1, 2011, with feedback, acceptance, or rejection emails to be sent within two weeks thereafter. Upon final editing of accepted papers, camera-ready copy will be due not later than May 1, 2011.

Due to printer's deadlines, manuscripts received after that deadline will not make it into the Proceedings. Instructions for preparation and submission of final manuscripts appear in a Guidelines for Submitting Papers document on the SARA website. The last three year's Proceedings were a landmark accomplishment for our organization. Please help the Society of Amateur Radio Astronomers to make the 2011 edition even better!

For more information go to **www.radio-astronomy.org.**

QEX