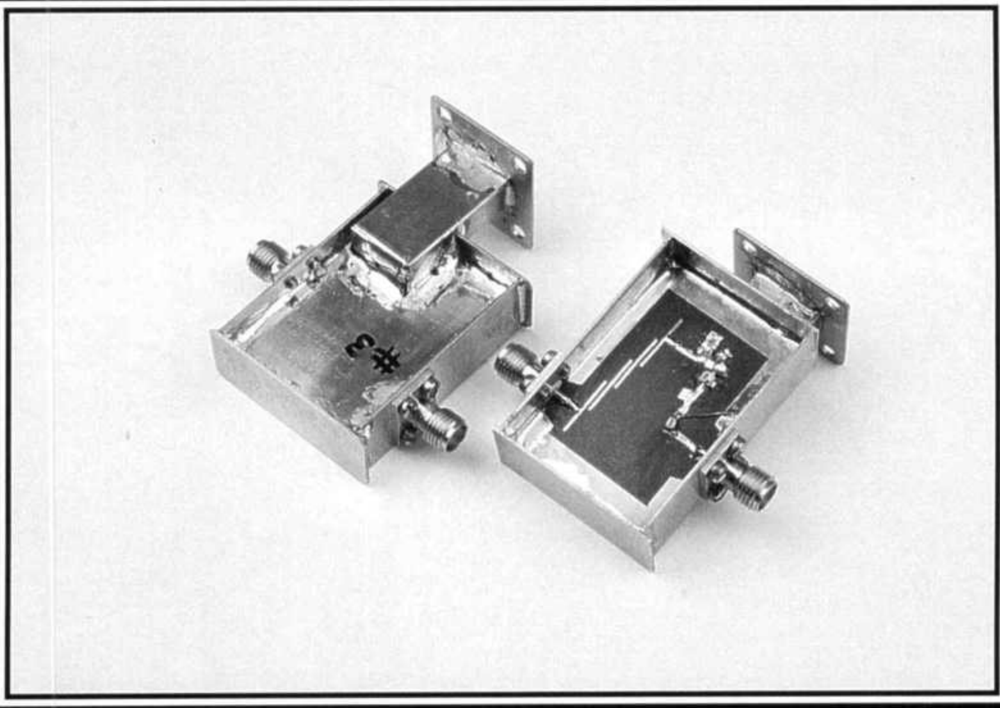
QEX

$1.75

ARRL Experimenter's Exchange

June 1994



Build It — *All* of It!

QEX: The ARRL
Experimenter's Exchange
American Radio Relay League
225 Main Street
Newington, CT USA 06111

QEX (ISSN: 0886-8093) is published monthly by the American Radio Relay League, Newington, CT USA.

Application to mail at second-class postage rates is pending at Hartford, CT.

Offices

225 Main St, Newington, CT 06111-1494 USA
Telephone: 203-666-1541
Telex: 650215-5052 MCI
FAX: 203-665-7531 (24 hour direct line)
Electronic Mail: MCIMAILID: 215-5052
Internet:qex@arrl.org

Subscription rate for 12 issues:

In the US: ARRL Member $12, nonmember $24;

US, Canada and Mexico by First Class Mail:
ARRL Member $25, nonmember $37;

Elsewhere by Airmail:
ARRL Member $48, nonmember $60.

QEX subscription orders, changes of address, and reports of missing or damaged copies may be marked: *QEX* Circulation. Postmaster: Form 3579 requested. Send change of address to: American Radio Relay League, 225 Main St, Newington, CT 06111-1494.

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

About the Cover:

This project from Zack Lau, KH6CP/1, shows how enclosures for microwave circuits can be built.

ISSUE NO. 148

Features

Columns

June 1994 QEX Advertising Index

Empirically Speaking

Let Us Compute

No doubt about it, "personal" computers are looking more like "real" computers all the time. These days, a new PC is likely to sport a 486 processor running at 25 MHz—or more, 4 megabytes of memory—or more, a couple of hundred megabytes of disk space—or more, and accessories such as sound cards, high-resolution video displays and advanced operating systems such as Windows NT, OS/2 or unix. Such machines provide desktop (or operating-table-top) capabilities that only a few years ago weren't found outside of expensive workstations or minicomputers.

What does this mean to amateurs? At present, not enough. For the most part, the increased capabilities of these machines have been applied mostly to user-interface improvements. New programs for use in Amateur Radio typically run in a graphical user interface (GUI) environment and contain all the bells and whistles one might want. They may control equipment, perform database functions such as automated logging, or be utilities for antenna or equipment design. What they do *not* do, except in rare cases, is apply the computer's processing power to signals.

As you'll see in our lead article this month, this is changing. Phil Karn, KA9Q, is using the capabilities of today's desktop computers to process digital data in ways that were impossible only a few years ago with technology readily available to amateurs. Others are beginning to make use of this new technology as well. Modern desktop computers are fast enough to do limited digital signal processing (DSP) without the addition of special-purpose DSP hardware. To date, only a few examples of this kind of application have arisen, but more are sure to follow.

Even better, new hardware such as sound cards that include embedded DSP processors are becoming available at consumer prices. Combine the signal-processing ability of such devices with that of a modern desktop computer and you've got a system that may be limited more by the available radio bandwidth than by the computer's ability to process high-speed signals. In Phil's article, he mentions how a DSP front end could substantially improve the techniques he's working on. The hardware for such a front end exists; we only need to develop the software. When that happens—and it will, we believe—amateurs will once again be at the forefront of developing improved low-cost communications techniques.

Do we need to do these things? Yes, and not only because they serve to improve our own communications, but because that kind of development is a way we can "contribute to the advancement of the radio art," which is one reason Amateur Radio exists.

Amen.

This Month in *QEX*

Moving "Toward New Link-Layer Protocols" is something packeteers desperately need to do. Phil Karn, KA9Q, who is doing just that, describes his work to date.

Analyzing the intermodulation performance of an amplifier isn't always easy. The classic two-tone analysis may not be the end of the story when the amplifier will handle a complex signal, as Dr. Ulrich L. Rohde, KA2WEU, shows in, "Simulation of KH6CP's VHF Driver."

Rus Healy, NJ2L, shares his hard-won experience at "Building Enclosures for Microwave Circuits" that work.

Testing...ya gotta do it, so you'll need a test-signal generator such as "A Simple Versatile UHF Test Source—0.4 to 1.5 GHz," by John Reed, W6IOJ.

Jack Belrose, VE2CV, an ARRL Technical Advisor, presents his analysis "On the Traveling-Wave Linear Dipole." A number of these antennas exist, and Jack has tested and analyzed them to determine their true performance.

In his "Digital Communications" column this month, Harold Price, NK6K, provides some food for thought regarding the level of effort required to produce some of the landmark projects of recent years. And he reports on a new such opportunity scheduled to fly on the upcoming Phase 3D satellite.—*KE3Z, email: jbloom@arrl.org.*

Toward New Link-Layer Protocols

This paper from the Proceedings of the TAPR 1994 Annual Meeting *describes some exciting ways of improving packet radio.*

By Phil Karn, KA9Q

This paper describes an experimental new link-layer protocol for amateur packet radio use. It extends my earlier MACA scheme for dealing with hidden terminals by incorporating a powerful forward error correction (FEC) scheme: convolutional coding with sequential decoding. The resulting hybrid protocol combines the best of the FEC and retransmission (ARQ) techniques. It should perform very well in the presence of hidden terminals, noise and interference, particularly pulsed radar QRM like that often found on the 70-cm amateur band.

Introduction

AX.25 is now 12 years old.[1] Although it has become the universal standard link-level protocol for amateur packet radio, it is widely recognized as being far from optimal. The experience of the past decade plus significant advances in the computer technology now available to the radio amateur suggest that we look at much more sophisticated alternatives. These techniques have existed for many years, but only now have powerful modern PCs brought them within easy reach of the average radio amateur.[2]

This article gives a brief overview of forward error correction techniques and then describes the author's work in progress to develop a new general-purpose link-level protocol suitable for amateur packet radio use at speeds up to about 100 kb/s. This protocol combines the features of my earlier MACA (Multiple Access with Collision Avoidance) access scheme with forward error correction (convolutional encoding with sequential decoding performed in software) in what the literature calls a *Hybrid Type II ARQ/FEC protocol.*[3,4,5] This protocol should perform well over channels with error rates up to 10%, far beyond the capabilities of AX.25. It was originally motivated by, and should be especially well suited for, channels with radar interference. With the right modem, it should also perform very well over analog satellite transponders like those on AO-13.

Computer Technology—1994 Versus 1982

In the late 1970s and early 1980s, the best computer that most radio amateurs could hope to own included a 4- or 6-MHz Z-80 running CP/M, 240-kB 8-inch floppy drives, 64-kB RAM, and either a "dumb terminal" or a small memory-mapped video display. The introduction of the IBM PC in the early 1980s upped these capabilities considerably, particularly in address space, but these machines were quite expensive when first introduced.

Since the original PC was introduced, there have been several more generations of Intel microprocessors: the 80286, first used in the PC/AT; the i386; i486; and now the Pentium. The faster 486 CPUs are nearly 100 times faster than the 8088 used in the original IBM PC.

Other measures of personal computer system performance have increased even more dramatically. It is not at all uncommon to find amateur systems with 16 megabytes of RAM (256 times the 64-kB maximum with CP/M) and hundreds of megabytes, or even gigabytes, of hard disk (thousands of times the capacity, and hundreds of times the speed of the 8-inch floppies used with CP/M). These machines rival the high-end engineering work stations of just a few years ago.

Notes appear on page 9.

7431 Teasdale Ave
San Diego, CA 92122

And best of all, these machines have plummeted in price even as their capabilities have grown. The 8088 has long been obsolete; the 286 more recently so. Motherboards with the 386 chip are now selling for less than a hundred dollars, a sure sign that they, too, will soon be obsolete.

What Modern Computers Make Possible

When AX.25 was developed, it was reasonable to emphasize simplicity over functionality and performance. DARPA had already experimented with much more sophisticated and powerful packet radio protocols, error correction and signal processing techniques, but theirs was a government-funded research program with (by amateur standards) very deep pockets.[6,7,8] Amateurs needed a practical standard they could afford, even if it didn't work very well.[9]

But many of the sophisticated techniques used in the DARPA project are now within our easy reach. Thanks to their wide data paths and high-speed caches, chips like the 386 and 486 are especially well suited to CPU-intensive operations such as forward error correction. Yet amateurs have been slow to exploit these capabilities by updating their protocols.

Amateur packet radio urgently needs FEC. A common rule of thumb is that the packet loss rate should not exceed 1% for good performance with pure ARQ protocols like LAPB or TCP.[10,11,12] The typical amateur packet radio channel far exceeds this. Causes include nonoptimal modem designs and radio interfaces, poor signal-to-noise ratios and interference.

FEC is especially effective against radar. We share our 70-cm amateur band with military radar and, in the coastal cities, the periodic audio whine of a Navy radar is a familiar sound.[13]

Forward Error Correction—Background

Forward Error Correction (FEC) has been around for quite some time. It essentially began when Claude Shannon's revolutionary 1948 paper launched the field of information theory.[14]

Error correction is a very deep and often highly mathematical subject, about which dozens of textbooks and thousands of journal papers have been written. I could not possibly include a complete discussion in this paper. However, the basic principles are easy enough to understand.

Shannon proved that it is possible to send data over even a noisy channel with an arbitrarily low error rate, as long as the data rate is less than the channel capacity, defined by the famous formula.

$$C = B\log_2\left(1+\frac{S}{N}\right)$$

Unfortunately, Shannon did not show *how* one could achieve this capacity, only that it was theoretically possible. Practical systems have only begun to approach the Shannon limit, although some have come remarkably close (the *Voyager* spacecraft downlink operated at about 50% of the Shannon capacity of its channel).

With FEC, a receiver can use all *of the received signal energy.*

The basic idea behind FEC is to add redundancy to the data being sent so that some number of channel errors can be corrected, up to a limit. The encoded data are referred to as *symbols*, to distinguish them from the original user data bits.

Compare this with the more familiar ARQ scheme, where redundancy (eg, a CRC) is added to the data only to *detect* errors; if even one error occurs, the receiver discards the entire block of received data and waits for a retransmission. If this happens often, a significant amount of channel capacity and transmitter energy is wasted.

With FEC, a receiver can use *all* of the received signal energy; it doesn't have to throw any of it away. This actually allows one to reduce the transmitter power required for a given user data throughput. This reduction is called the *coding gain* of the code. Depending on the particular code, its implementation, the modem in use and the amount of redundant information added, coding gains can range from 1 to 8 dB or even more against additive white Gaussian noise (AWGN, eg, the thermal noise of a preamp and antenna system).

White noise is actually a worst case for FEC; certain codes can provide dramatically higher gains against nonwhite noise such as radar pulses, and against short deep fades. Coding gains of 40 to 50 dB or more against radar QRM are easily achievable, assuming that the receiver recovers quickly after each radar pulse.[15] The error rate in a radar-QRMed digital system depends primarily on the user data rate and the radar pulse duration and repetition rate. It is essentially independent of radar signal level over a wide range. For example, radar pulses 10-20 microseconds long are comparable to a bit time at 56 kb/s (17.8 microseconds). If the radar pulse power exceeds the desired signal power minus the required S/N demodulator margin, a data bit will be corrupted. However, if the receiver recovers rapidly after each pulse, the bits sent between the pulses will arrive unharmed no matter how much stronger the radar gets. At a repetition rate of 400 Hz, only every 140th data bit will be clobbered. This is an error rate of 0.7%, which is *easily* handled with just a modest amount of FEC. On the other hand, without coding communication would be almost impossible unless you could increase transmitter power to completely overcome the radar, and this is quite often impractical.

A similar situation exists with fading. If the fades are relatively short, coding can easily handle the situation no matter how deep the fades are.[16] The alternative is to increase transmitter power until there's enough margin even in the deepest fade, which may again be impractical. FEC essentially "smears" individual data bits over time on the channel so it is not necessary to receive an entire message, only enough of it to permit the decoder to fill in the missing spots from what it did get.

There is no free lunch, however; coding gains necessarily come at the expense of increased bandwidth. For example, the code I am currently using in my experiments provides a 4-dB coding gain against AWGN at the expense of doubling the required bandwidth by sending two encoded symbols for each user data bit. That is, if I have a 10-kHz channel capable of carrying 10 kb/s with 100 W of transmitter power, then my code enables me to send the same 10 kb/s with only about 40 W of power, but I need 20 kHz of bandwidth to do it. Or I could stay in my original 10-kHz channel, reduce my data rate to 5 kb/s, and reduce my transmitter power to only 20 W.[17]

Because all FEC schemes add some amount of redundancy to the user data being transmitted over the channel, it is important to distinguish between the user data rate and the encoded symbol rate as sent over the channel. Since it is the user data rate that ultimately counts, the literature uses the parameter E_b/N_0 rather than "signal to noise ratio" to define the performance of a coding system and modem. E_b is the received energy per user data bit; N_0 is the received noise spectral density.

Types of FEC

Block Codes

The earliest error-correcting codes inspired by Shannon's work were Hamming's "block" codes.[18] As the name implies, a block code operates on a fixed amount of data that depends on the particular code. Hamming's code has been used on several amateur spacecraft to correct radiation-induced memory errors; the memory word makes a natural code "block." This code can generally correct one error in each 8-bit byte of memory. Another popular block code, the Golay code, adds 11 redundant bits to 12 data bits to produce 23 bits for transmission. This is referred to as a (23,12) block code. This code can correct any combination of three or fewer errors in its transmitted block of 23 bits.[19]

Another important block code (out of many) are the Reed-Solomon codes, used in the compact disc and elsewhere.[20] Reed and Solomon actually defined a whole family of codes. Reed-Solomon codes can operate on multibit symbols (as opposed to binary symbols) and can be easily constructed with relatively large block sizes. These two characteristics together provide an excellent burst-error-correcting capability, which is especially useful when combined or *concatenated* with other error detecting and correcting codes. The Clover II HF modem, designed by Ray Petit, W7GHM, makes extensive use of Reed-Solomon coding.[21,22,23]

Convolutional Codes

Another class of error-correcting codes operates on an arbitrary stream of bits, rather than fixed-size blocks. These are the convolutional codes, sometimes called "tree" codes. In a more general form they are known as "trellis" codes and are found in modern dial-up telephone modems (eg, V.32 and V.32bis).

Convolutional codes are easily generated in hardware with a shift register, two or more exclusive-OR "parity trees" and a multiplexer. These operations are also easily implemented in software. The length of the shift register defines an important parameter called the "constraint length" K of the code. The larger K is, the better the code will perform (subject to the limitations of the decoding process as discussed below).

Because convolutional codes operate on continuous streams, they are specified by the ratio of the input data and output symbol rates. For example, an encoder that produces two encoded symbols for each user data bit is known as a "rate 1/2" code, often abbreviated to "r=1/2".

Convolutional codes have several interesting properties. They are especially easy to generate and provide excellent performance for a given amount of complexity. They are well suited to varying amounts of data, eg, variable length data packets. They are also readily adapted to *soft decision* decoding.[24] This uses symbol quality indications from the modem to aid the decoding process. For example, instead of slicing the demodulator output to binary 0 or 1 with a comparator, one uses an A/D converter to indicate the relative quality of each 0 and 1.[25]

When properly implemented, soft decision decoding yields a

$$\frac{2}{\pi}(2\text{ dB})$$

performance improvement over 1-bit or *hard decision* decoding in the presence of AWGN. The code I'm using would therefore provide a 6-dB coding gain were I to use full-blown soft decision decoding, but this will have to wait for modems (most likely DSP-based) that provide soft decision samples.

The code would provide a 6-dB coding gain with soft decision encoding.

My decoder currently accepts "unknown" as a received symbol value in addition to "one" and "zero." This 3-level *binary erasure channel* is a good model for a radar-QRMed channel where symbols are erased by radar pulses. Making this erasure information available to a decoder can dramatically improve its performance. For example, without erasure information the rate 1/2 code I'm using can, with considerable effort, barely correct a stream of symbols when the symbol error rate reaches 10%. However, if the modem can tell the decoder *which* symbols have been trashed by radar pulses (perhaps by running the receiver's noise blanker gate line to the decoder), then the decoder will continue to function even when nearly 50% of the received symbols have been erased—ie, when almost all of the redundant information added by the encoder has been removed by the channel.

A 3-level decoder is also useful for *puncturing*, another useful technique easily applied to convolutional codes. This involves simply not transmitting some fraction of the encoded symbols and substituting erasures in their place at the decoder.[26] Puncturing makes it easy to vary the coding rate (the amount of redundant information added to each user data bit) in accordance with changing channel conditions without having to change the encoder or decoder. This is important because the redundant information added by a coder is nearly useless overhead when the channel conditions are good enough to not require the code's full error correcting capability.

Sequential Versus Viterbi Decoding

There are two general classes of decoders for convolutional codes: sequential and minimum-likelihood (Viterbi). The sequential decoder is the older of the two, dating from the late 1950s.[27] The Fano algorithm for sequential decoding dates from the early 1960s and is still in use with few changes.[28]

Viterbi first proposed his algorithm in the late 1960s.[29] It has since become the most popular technique for decoding convolutional codes, primarily because of the availability of high speed VLSI hardware decoders.[30] The *Voyager* spacecraft down-link mentioned earlier uses convolutional encoding with Viterbi decoding, sometimes in conjunction with a Reed-Solomon block coder.

Both the sequential and maximum-likelihood decoders work by regenerating the data stream that, when passed through a local copy of the encoder, most closely matches the received encoded symbol sequence. Because the received symbols usually contain errors, the match is seldom exact; but as long as the error rate is not too high, the decoder will reproduce the original user data without any errors. This is possible because the effect of any particular user data bit is spread over many encoded channel symbols. For example, in the K=32 rate 1/2 code I use, each data bit affects 64 encoded symbols. Even when several symbols are lost, there is usually enough information left in the remaining symbols to reconstruct all of the original data bits. The coding gain of a convolutional code depends on the constraint length, K; the larger the better.

The main difference between the sequential and maximum-likelihood decoders is the way they regenerate the original data sequence that is fed through the decoder's local copy of the encoder for comparison with the received symbol stream. Both use what might be called "intelligent brute force," as each tests and discards many data sequences that don't work out. This requirement for "brute force" CPU crunching is why convolutional decoding is just now becoming practical for amateur use at reasonable speeds.

The Viterbi decoder explores every possible data sequence in parallel, discarding at each step those that can't possibly be correct. These parallel operations are well suited to hardware VLSI implementation. The number of data sequences that a Viterbi decoder explores in parallel is 2^K. Increasing K by 1 doubles the work that the decoder must do to recover each data bit. Viterbi decoders typically work with K=7 or K=9; larger values are very rare.

The sequential decoder, on the other hand, explores only one data sequence at a time, so its work factor is almost independent of K. As long as the sequence being tested produces symbols reasonably close to those being received, it keeps on going. When the decoder "gets into trouble," it backs up and methodically searches increasingly dissimilar data sequences until it eventually finds one that gets it back onto the right track. Much larger values of K are typically used with sequential decoders than with Viterbi decoders. My code uses K=32, a convenient value on modern microprocessors like the 386 and 486 with 32-bit registers.

Since larger values of K give better performance, this seems to tip the scales in favor of sequential decoding. However, the Viterbi decoder always operates at a constant rate, while the speed of the sequential decoder is a random function that depends strongly on the channel error rate; the higher the error rate; the greater and more unpredictable the decoding time becomes. At sufficiently high error rates, the decoding time of a sequential decoder becomes unbounded.[31] Any practical sequential decoder must therefore have a timer to keep it from running "forever" if it is inadvertently fed garbage or an extremely noisy packet. This makes the sequential decoder less attractive than the Viterbi decoder for real-time applications with strict delay limits such as full-duplex digital voice.

On the other hand, the *average* decoding time of a sequential decoder is often less than the fixed decoding time of a software Viterbi decoder running on the same CPU. This is important if (a) you've decided to implement your decoder in software to avoid the cost of a dedicated VLSI Viterbi decoder, and (b) you have a nonreal-time packet application such as file transfer, where it's the average decoding speed that counts.

An interesting property of sequential decoders with large values of K is that the probability of uncorrected errors becomes very small. It is much more likely that the decoder will time out first. This may make it unnecessary to include a CRC or other error-detecting code to ensure that the decoded data is correct; the decoder timer takes the place of a CRC check. We don't actually avoid the overhead of the CRC, though. A convolutional coder requires a tail of zeros at the end of every packet to return the coder to its starting state and to allow every user data bit to influence the full number of channel symbols. The tail length must be at least K–1 bits, or 31 for my code (I use 32 just to keep things in round numbers).

Whether the inherent error-detecting ability of my K = 32 sequential decoder provided by its time-out mechanism is comparable to that of a 32-bit CRC is an interesting question. I don't know yet.

So to recap, Viterbi decoding is usually preferable when the application requires a constant decoding delay and dedicated VLSI hardware decoders are available. Sequential decoding has the edge when a variable decoding delay is tolerable and a software implementation is required (eg, to minimize cost to the radio amateur who already has a PC). For these reasons I have chosen sequential decoding for my experimental protocol.[32]

Interleaving

Convolutional decoders work best when the symbol errors they correct are evenly distributed throughout the transmission. This is the case on channels limited by AWGN (thermal noise). However, errors caused by momentary interference and fading often occur in bursts. These can cause sequential decoders to get into trouble since the work factor goes up exponentially with the length of the burst.

Interleaving is the standard approach to this problem. The symbols from the encoder are rearranged in time before transmission and put back into their original order before decoding. Interleaving doesn't remove any errors, it simply scatters them in case they were adjacent on the channel.

Several classes of interleaving schemes exist. The particular interleaving scheme I've chosen for my protocol uses *address bit reversal.* Each symbol address is written in binary in the usual way with the high order bit on the left. Then the bits are reversed right-to-left, forming a new number. For example, the sequence

0 1 2 3 4 5 6 7

when bit-reversed becomes

0 4 2 6 1 5 3 7

Note that all even numbers are in the first half of the block, and all the odd numbers in the second half. Note also that when these numbers are again bit-reversed, the original sequence reappears.

Since address bit reversal only works when you have 2^N numbers, I need some way to extend this scheme to arbitrary length packets. To facilitate this, I first pad each data packet out to a multiple of 32 bits (64 symbols for my rate 1/2 code). Then I write the symbols as a 2-dimensional matrix with 64 rows, first going vertically down the columns and using as many columns as I need. Here is an example with 193 symbols, numbered 0 through 191:

0	64	128
1	65	129
2	66	130
.	.	.
.	.	.
63	127	191

Then I interchange the rows by bit-reversing the row addresses:

0	64	128
32	96	160
16	80	144
.	.	.
.	.	.
63	127	191

Now I actually transmit the symbols by transmitting horizontally across each row. The transmitted symbol sequence is therefore

0, 64, 128, 32, 96, 160, 16, 80, 144, ..., 63, 127, 191

At the receiver I reverse the process, restoring the symbols to their original order. Note how adjacent symbols from the encoder are always widely separated in time when they go over the channel; in particular, note how all of the even-numbered symbols appear in the first half of the transmission and the odd-numbered symbols in the second half.

Variable Rate Puncturing by Interleaving

I chose this particular interleaving scheme because it makes variable-rate code puncturing especially easy. Suppose I transmit only the first 32 of the 64 rows. This covers all of the even-numbered symbols in the stream; I've sent every other symbol. Since there are twice as many symbols as data bits, the effective code rate is 1/2 divided by 1/2, or 1 (ie, no redundant information, and no ability to correct errors). Now suppose we also send the 33rd row. This gives us 1/2 divided by 33/64, which is 32/33. This "high rate" code cannot correct as many errors as the original rate 1/2 code, but if this is good enough for the channel, we can avoid sending the other 30 rows. On the other hand, if the channel is poor, we simply send enough rows to lower the effective code rate until decoding is possible.[33] Adding the 34th row gives us a code rate of 16/17, and so on up to all 64 rows, which returns us to a rate 1/2 code.

Nothing says we have to send all of the rows in a single transmission. We could start by sending just the first 32 rows (no redundancy) and attempting to decode the packet with a tight timeout. Although we cannot actually correct any errors with only 32 rows, if any do occur the decoder will "get stuck" in the tail and time out, thus indicating that errors exist. If this happens, we can send additional rows until the receiver is finally able to decode the packet. In this way we send only as much redundancy as the channel currently requires. This is the idea behind the Type II Hybrid ARQ/FEC scheme mentioned earlier: use the power of FEC to deal with channel errors, but use the adaptability of ARQ to adjust the FEC overhead to that actually required by the channel.[34,35,36,37]

We send only as much redundancy as the channel requires.

What if the receiver cannot decode the packet even after we send all the rows? One possibility is to send the whole thing again and to use code combining at the receiver to add the two transmissions before decoding.[38] For example, if we send all of the symbols belonging to a rate 1/2 code twice, we have effectively switched to a rate 1/4 code. although this particular rate 1/4 code provides no additional coding gain over the original rate 1/2 code (since the retransmissions are identical), adding the two transmissions increases the total received energy (and E_b/N_0) by 3 dB.[39] Of course, this comes at the expense of halving the user data rate. We could take this scheme even further by adding more than two transmissions but we are eventually limited by the packet synchronization mechanism discussed in the next section.

This scheme can provide some of the benefits of soft decision decoding even when only hard decision samples are available from the modem. For example, the DARPA packet radio could combine two hard decision copies of a packet by erasing those symbols that disagreed between the two transmissions. (See Note 6.) Symbols that agreed were left unchanged. As we've already seen, a sequential decoder has a much easier time dealing with erasures than with errors.

Synchronization

Any packet protocol needs something to reliably flag the beginning of a packet. HDLC uses the 8-bit value 7E (hex) for this purpose, but since we want to operate reliably over very noisy channels this is not acceptable. As anyone who has ever operated a packet station with the squelch open and "PASSALL ON" knows, you don't have to wait very long for a 7E to appear by chance in random noise. And what if a packet is actually present, but one or more bits in the flag is in error? The entire packet would be missed. We *could* accept flags with, say, at most one bit in error but this would make the false alarm problem even worse.

The solution is to use a longer flag, or sync vector. The longer the sync vector, the easier it is to reliably detect real packets with errors while rejecting false alarms (triggering the sync detector on random noise). I have chosen a 64-bit sync vector for my protocol that consists of a 63-bit pseudorandom (PN) sequence (generated by a 5-stage shift register with feedback) augmented by an extra 0 to make 64 bits. The receiver correlates the incoming symbol stream against a local copy of the sync vector, and it declares synchronization whenever they match with 13 or fewer errors. This allows the detector to work reliably up to a channel error probability of about 20%, well above the error correcting capability of the rate 1/2 convolutional code (about 10%).[40] Yet the sync vector is so long that the probability of random noise triggering the detector is quite small.

Scrambling

Several higher-speed packet radio modems, eg, the K9NG/G3RUH 9600 bit/s FSK modems and the WA4DSY 56 kbit/s modem, use scrambling to ensure a sufficiently high bit transition density to allow the demodulator to recover clock regardless of the user's data sequence. However, the self-synchronizing descramblers they use have an unfortunate property: error propagation. Each channel error produces a characteristic pattern of several closely spaced data bit errors that depend on the particular polynomial being used. Without FEC this is of little consequence since even a single bit error is enough to ruin a packet, so extra errors can't make things worse. But with FEC, we don't want the modem to do anything to make the decoder's job harder. So we must disable the scrambling and descrambling functions.

This leaves us with the problem that scrambling was originally intended to solve: how can we ensure a good transition density on the channel no matter what the user sends? It turns out that the solution is relatively simple: we scramble the user's data ourselves, but we don't use a self-synchronizing descrambler. We use a fixed PN sequence that is started from a known point at the beginning of the packet and allowed to "free run" for the length of the packet. This type of scrambling doesn't propagate errors, but it does require independent synchronization—which we already have from the sync vector mechanism just described.

It may turn out that current modem clock recovery mechanisms are inadequate with FEC. FEC can operate with a E_s/N_0 far lower than that required to produce good data without coding, and it's entirely possible that existing modem clock recovery circuits won't work on these weak signals.[41] Other approaches may be necessary.

One possible approach is to perform the sync vector correlation function in a DSP modem on raw A/D input samples at some integer multiple of the incoming data rate. When the correlator output peaks, we can then start blindly counting off the appropriate number of A/D samples between each received symbol. If the transmitter and receiver clocks are closely matched in frequency and the packets aren't too long, this should provide reasonably accurate symbol timing for the entire packet without having to extract clock from those (noisy) symbols. If the clocks are too far apart for this to work, another possibility would be to buffer all of the raw A/D samples in the packet and post-process them looking for the sampling frequency that produces the best eye pattern for the packet as a whole.

Protocol Headers

The discussion of variable rate code puncturing assumed that we have had a reliable way to control it. But the control information can also be corrupted by channel errors. How can we deal with this? By always using full FEC coding for the packet header, regardless of the coding rate in use for the user data portion of the packet. Since the header is (hopefully) small compared to the user data, the overhead incurred by this is (hopefully) also small. To do this, though, we have to be very selective about what goes into the header.

The header in my protocol is currently 16 bytes (128 data bits or 256 encoded symbols) long. It contains the following information: source address, destination address, frame type, transmission length, previous frame error count and coder tail.

This decoder can easily keep up with a 56-ksymbols/s stream.

The coder tail is all zeros. As mentioned earlier, it is required by our use of a convolutional coder. Because we restart the coding process between the header and the data portion of the packet, each portion needs its own coder tail.

The source and destination addresses consist of the station call signs and SSIDs as in AX.25, but the call signs are more efficiently coded. Since there are only 36 legal characters in a call sign (the 26 letters in the English alphabet plus the ten decimal digits) there is no need to spend an entire 8-bit byte on each one. If we add "space" as a 37th character and use radix-37 encoding, we can encode into 32 bits any legal call sign up to 6 characters long.[42] Add 4 bits for an SSID and the complete address fits into 36 bits, as compared with 52 for AX.25.

The frame types are as follows:

- Request to Send (RTS)
- Clear to Send (CTS)
- User Data
- Negative Acknowledgment (NAK)
- Positive Acknowledgment (ACK)

The RTS tells the receiver of the sender's intention to send a certain amount of data. The receiver responds with a CTS that echoes this length. This tells the sender to go ahead with the actual data transmission, and it also has the important side effect of telling anyone else on the channel to remain quiet for the appropriate amount of time.

If the receiver is able to decode the transmission (the sequential decoder completes without timing out), it returns an ACK to the sender that it may proceed to the next block of data. Alternatively, if the timer expires before the sequential decoder finishes, the receiver returns a NAK. The NAK confirms to the sender that its transmission was received, but with too many errors. The sender then sends additional rows from the interleaver output, which the receiver combines with those symbols already received. If the receiver is still unable to decode the packet, the cycle repeats with additional NAKs and data packets containing additional rows of symbols until the receiver finally succeeds and returns an ACK. At this point the sender can continue to the next block of data.

Note that the NAK, like the CTS, also holds off other stations from transmitting so that they do not interfere. In fact, I may eventually merge the NAK and CTS messages into a single type, with the CTS simply being the special case of a NAK sent before any data symbols have been received.

It also goes without saying that should the sender receive neither an ACK nor a NAK in a reasonable time, it must retransmit its last frame. This can only occur if the channel was so poor that the heavily coded header could not be decoded, or perhaps even the sync vector was missed. If this is a temporary condition, then a retransmission will get things moving again.

The previous frame error count field is used in ACK packets to let the sender know how many errors were detected and corrected in the last packet. The sender can use this information to aid in deciding how many of the interleaver symbol rows to send up front in its next transmission, without having to wait for the receiver to ask for them with NAK messages. This helps minimize modem transmit-receive cycles when channel conditions are fairly constant. At the moment I actually have two previous frame error count fields: one for the last header received, and one (in ACK packets only) for the last data field received.

Status and Open Questions

Since sequential decoding is the heart of this protocol and by far the biggest consumer of CPU cycles, I have spent most of my time to date working on my implementation of the Fano algorithm in C. It now runs quite well on the 486-50. At the moment, this decoder can, on average, easily keep up with a 56-kilosymbol/s stream (eg, from a WA4DSY modem) as long as the symbol error rate is less than about 2%. More errors could be tolerated at lower channel speed, with a faster CPU, or with a better optimizing compiler, but this performance is already good enough to be quite useful.

The correlator that searches for the sync vector is another potentially CPU-intensive task. Although correlation is simpler than sequential decoding, the decoder runs only when a packet has actually been received, while the correlator may have to process a continuous symbol stream from the modem if it has no carrier-detect squelch.[43] So it is still desirable to make the correlator run as fast as possible in order to free up the maximum amount of CPU time for other tasks. I have implemented a correlator in assembler that runs at several hundred kilosymbols per second, with most of the time spent in the function that returns the next received symbol.

I have not yet finished the complete protocol, however. When I do, I will have plenty of work left. I need to answer the following questions:

1. Is a rate 1/2 code strong enough, particularly for the packet header? Should I consider using a rate 1/3 or even lower coding for maximum robustness?
2. Should I consider a block code (eg, Golay) for the packet header in order to eliminate the need for a coder tail?
3. What strategy should the sender use to decide how many rows (out of the 64 available) should be sent in any given transmission? How can I make best use of recent history (especially the receiver's observed error rate indication) to send just the required amount of redundancy for each packet in as few transmissions as possible?
4. Is 64 interleaving rows optimum? Is the whole interleaving scheme optimum?
5. What is the optimum packet size for transmission? Should transmissions be large to decrease header and modem turnaround overhead, or should they be small to decrease the chances of a sequential decoding timeout and the resulting need to transmit additional redundancy?
6. Is relying on a decoder timeout sufficient to detect errors when a packet has been punctured back to rate 1 (no redundancy), or is a true CRC still required?
7. Does the MACA algorithm work well in the presence of stations too far away to reliably decode CTS messages, but close enough to cause harmful interference? Does FEC help this problem by improving the capture effect?
8. Can I make a sequential decoder that works well on soft decision samples such as those that might be produced by a PSK modem implemented in DSP, or is the sequential decoder's well-known sensitivity to incorrect soft-decision metrics a serious stumbling block? Can I compute the metrics on the fly according to observed noise levels to mitigate this problem?
9. Will the clock recovery circuits in existing amateur packet radio modems turn out to be the limiting factor instead of the error correcting capability of the code?

And last but not least,

10. Will the average amateur be willing to use this stuff?

Credits

I would like to thank several people for their advice and assistance: Franklin Antonio, N6NKF, and Klein Gilhousen, WT6G, for sharing their considerable expertise in the practical application of convolutional coding; Paul Williamson, KB5MU, and Bob McGwier, N4HY, for their insights and especially their patience in listening to my ravings; Andy Demartini, KC2FF, of Digital Radio Systems Inc, for his donation of two DRSI PCPA Type 1 interface cards; and Skip Hansen, WB6YMH, for his clever trick of turning a Zilog SCC into a simple and dumb serial/parallel converter.

Notes

[1]Fox, Terry, WB4JFI, ed, *AX.25 Amateur Packet-Radio Link-Layer Protocol*, Version 2.0, October 1984, ARRL. (Updates earlier versions dating from 1982.)

[2]Many of the fundamental references cited in this paper date from the 1970s or even earlier.

[3]Karn, Phil, "MACA—A New Channel Access Method for Packet Radio," *Proceedings of the 9th ARRL/CRRL Amateur Radio Computer Networking Conference*," London, Ontario, Canada, September 22, 1990, p 134.

[4]Kallel, Samir and Haccoun, David, "Generalized Type II Hybrid ARQ Scheme Using Punctured Convolutional Coding," *IEEE Transactions on Communications*, Vol 38, No. 11, November 1990, p 1938.

[5]Lin, Shu and Yu, Philip S., "A Hybrid ARQ Scheme with Parity Retransmission for Error Control of Satellite Channels," *IEEE Transactions on Communications*, Vol COM-30, No. 7, July 1982, p 1701.

[6]Kahn, R. E., Gronemeyer, S. A., Burchfiel, J., and Kunzelman, R. C., "Advances in Packet Radio Technology," *Proceedings of the IEEE*, November 1978, p 1468.

[7]Shacham, Nachum, "Performance of ARQ with Sequential Decoding Over One-Hop and Two-Hop Radio Links," *IEEE Transactions on Communications*, Vol COM-31, No. 10, October 1983, p 1172.

[8]Special issue on packet radio networks, *Proceedings of the IEEE*, January 1987.

[9]The military also discovered the same thing, with the ironic result that amateur packet radio technology found its way into several military applications.

[10]Automatic request-repeat: (re)send each packet until an acknowledgment is received.

[11]The connection-oriented part of AX.25, borrowed from X.25 Level 2.

[12]The Internet's connection-oriented transport level protocol.

[13]Much of the early work on FEC was prompted by the Navy's desire for reliable shipboard communication links that could tolerate local radar QRM. My present interest was initially prompted by the heavy radar QRM we often experience to our 70-cm 56-kb/s modems here in San Diego.

[14]Shannon, C. E., "A Mathematical Theory of Communication," *Bell System Technical Journal*, Vol 27, July/October 1948, pp 379-423.

[15]A pulse blanker is important to keep the radar energy out of the receiver AGC and IF filters. It can also directly aid the FEC process, as explained later.

[16]A radar QRMed channel with a pulse blanker at the receiver is really just a channel with infinitely deep fades during each pulse.

[17]The extra 3 dB of power reduction in the latter case is *not* part of the coding gain; it results directly from the 50% reduction in user data throughput.

[18]Hamming, Richard W., "Error Detecting and Error Correcting Codes," *Bell System Technical Journal*, April 1950.

[19]Lin, Shu and Costello, Daniel J., Jr., *Error Control Coding: Fundamentals and Applications*, Prentice Hall, 1983 (ISBN: 0-13-283796-X).

[20]Pohlmann, Ken C., *Principles of Digital Audio*, second edition, Sams, 1990 (ISBN: 0-672-22634-0).

[21]Petit, Raymond C., W7GHM, "The Cloverleaf Performance-Oriented HF Data Communications System," *QEX*, July 1990. Also, *Proceedings of the 9th ARRL/CRRL Amateur Radio Computer Networking Conference*, London, Ontario, Canada, September 22, 1990, p 191.

[22]Petit, Raymond C., W7GHM, "Clover-II: A Technical Overview," *Proceedings of the 10th ARRL Amateur Radio Computer Networking Conference*, San Jose, CA, September 27-29, 1991, p 125.

[23]Henry, Bill, K9GWT, and Petit, Raymond C., W7GHM, "HF Radio Data Communications," *Communications Quarterly*, Vol 2, No. 2, Spring 1992, p 11.

[24]It is possible to "soften" the decoding of block codes, but with considerably more effort. Also see Chase, David, "A Class of Algorithms for Decoding Block Codes With Channel Measurement Information," *IEEE Transactions on Information Theory*, Vol IT-18, No. 1, January 1972.

[25]A simple comparator is really a 1-bit A/D converter, so we are just increasing the resolution of the A/D converter we already have.

[26]Obviously, the receiver has to know which symbols are not sent, otherwise it will get very confused.

[27]Wozencraft, J. M., "Sequential Decoding for Reliable Communications," Research Laboratory of Electronics, MIT, Cambridge, MA, *Technical Report 325*, 1957.

[28]Fano, Robert M., "A Heuristic Discussion of Probabilistic Decoding," *IEEE Transactions on Information Theory*, April 1963, pp 64-74.

[29]Viterbi, Andrew J., "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, Vol IT-13, No. 2, April 1967.

[30]Qualcomm, Inc, data sheets for Q1650, Q0256 and Q1401 Viterbi Decoder ICs.

[31]Jacobs, Irwin Mark, "Sequential Decoding for Efficient Communication from Deep Space," *IEEE Transactions on Communications Technology*, Vol COM-15, No. 4, August 1967, p 492.

[32]The DARPA packet radio project also chose convolutional coding with sequential decoding.

[33]Sending additional rows also increases the total transmitted energy. This increases the E_b/N_0 ratio, which also aids decoding.

[34]Kallel, Samir and Haccoun, David, "Sequential Decoding with ARQ and Code Combining: A Robust Hybrid FEC/ARQ System," *IEEE Transactions on Communications*, Vol 36, No. 7, July 1988, p 773.

[35]Mandelbaum, David M., "An Adaptive-Feedback Coding Scheme Using Incremental Redundancy," *IEEE Transactions on Information Theory*, May 1974, p 388.

[36]Metzner, John J., "Improvements in Block-Retransmission Schemes," *IEEE Transactions on Communications*, Vol COM-27, No. 2, February 1979, p 524.

[37]See Note 4.

[38]Chase, David, "Code Combining—A Maximum-Likelihood Decoding Approach for Combining an Arbitrary Number of Noisy Packets," *IEEE Transactions on Communications*, Vol COM-33, No. 5, May 1985, p 385.

[39]Pactor uses *Memory ARQ*, which is essentially a combining scheme on uncoded data. FEC and interleaving could improve Pactor's performance considerably without any increase in bandwidth.

[40]This provides some margin to allow successful code combining, which could allow us to operate above a 10% symbol error rate.

[41]Symbol energy-to-noise spectral density ratio, as opposed to E_b/N_0. For a rate 1/2 code, E_s/N_0 is 3 dB less than E_b/N_0.

[42]Some might like to see room for longer call signs. However, except for the rare special event station, all Amateur Radio call signs have been 6 characters or less and are likely to remain so for quite some time. In my opinion, the legal requirement for IDs with prefixes during reciprocal operation (eg, "W6/GB1AAA") is better met with a special ID frame every 10 minutes than by requiring everyone to make room for them in every packet header. I'm willing to be persuaded otherwise. Remember that this is still an experimental protocol.

[43]Good carrier detect circuits are already hard to build, and it will probably be almost impossible to make them operate quickly and reliably at the much lower E_s/N_0 ratios usable with FEC. And recall that one of the principles behind MACA is that carrier detect is essentially worthless in a hidden-terminal environment anyway.

Simulation of KH6CP's VHF Driver

An independent simulation of a recent design by Zack Lau, KH6CP/1, shows that two-tone analysis doesn't tell the whole story.

by Dr. Ulrich L. Rohde, KA2WEU

Zack Lau had a good idea by applying both current and voltage feedback to a CATV transistor to obtain a high-performance medium power amplifier that can be used for a number of interesting applications.[1] I decided it might be fun to simulate the amplifier in both linear and nonlinear mode and look at the differences between two- and three-tone nonlinear results.

For the simulation, the schematic has been slightly modified by combining the two emitter-bias circuits into one and eliminating the 50-Ω transmission lines at the input and output and therefore assuming the signal generator drives the transistor directly at the base via C_1.[2]

Based on some educated guesses for the parasitics of the elements, I swept the frequency response of the amplifier as shown in Fig 1. It has a nice frequency response. The low frequency response is determined by the capacitors C_1 and C_2 and the combined values of C_3 and C_4, while the cut-off frequency around 200 MHz is due to the transistor performance itself.

Fig 2 shows the input and output matching in the standard S-parameter form plotted on a Smith Chart. These impedances agree with the published data.

Practical applications for these feedback amplifiers apply not just to two tones but many tones (voice) and, therefore, I decided to do a three-tone test on the amplifier with all three tones being around 200 MHz (195.25 MHz, 200.25 MHz, 205.25 MHz) in accordance with the European three-tone standard with levels of 2×0 dBm and 1×6 dBm. The circuit description can be seen in Table 1. The nonlinear analysis first provides a dc current of about 64 mA at 13.0 V as shown in Table 3 in *QEX* (see Note 1). The main reason for this is the self-heating effect of the chip. The dc current increases slightly and the nonlinear program has to be told the actual

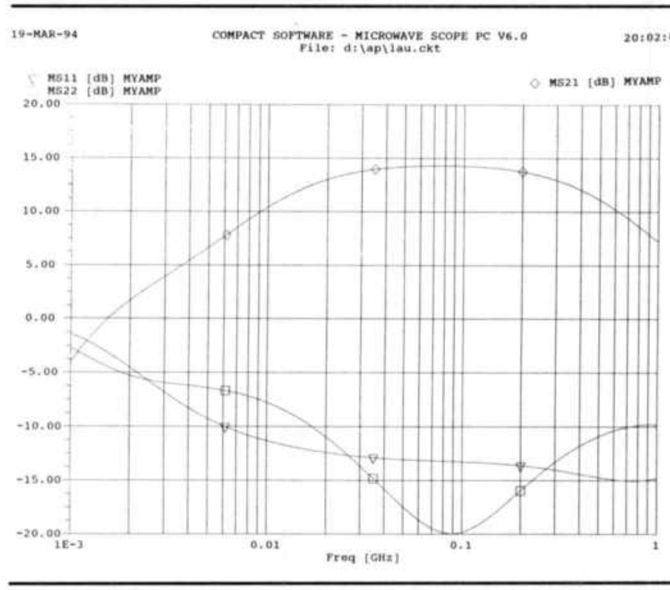


Fig 1

[1]Notes appear on page 14.

52 Hillcrest Drive
Upper Saddle River, NJ 07458

Table 1

```
**********************************************
* CTRL: MICROWAVE HARMONICA PC control block *
**********************************************
CTRL
 SVTOL  1.0E-3
 HBTOL  1.0E-7
 CFNEWT 0.5
 ITER   0
END
*
*****************************************************
* NBLK: Nonlinear block. All the "BIAS", "DIOD",    *
*     "FET" and "BIP" models must be defined in     *
*     this block. Only ONE (1) NBLK is allowed.     *
*****************************************************
* MODEL FOR  TRANSISTOR AT 10V, 100 MA
* using the bip model for noise analysis
* TRANSISTOR EQUIVALENT CIRCUIT
******************************
Ic:64 ; MA COLLECTOR CURRENT
* Ft=5GHZ  EMITTER CUT-OFF FREQUENCY
B:100  ; HFE
********************************
NBLK
* External elements to the intrinsic transistor
  RES  51  52      R=2.38
  RES  52  53      R=2.95
*****************************************************************
******  Preparation for nonlinear second order diode simulation
*****************************************************************
  CAP  51  70      C=0.086PF
  CAP  52  71      C=0.052PF
  CAP  53  72      C=0.052PF
  RES  70  56      R=0.1
  RES  71  56      R=0.1
  RES  72  56      R=0.1
  RES  55  56      R=5
  RES  57  58      R=.17
  CAP  56  57      C=.2PF
* Linear/NONlinear BIP description:
*
 BIP 53 56 58
+   ; LINEAR parameters:
+    RO = 0          RB1 = 0.9974       RC = 1E+030
+    RE = 0.394      RCE = 649.7        CO = 1.703E-013
+    CI = 1.703E-013    CE = 2.381E-011    RC1 = 0
+    RE1 = 0.2       RC2 = 0.1          RB2 = 0
+    LB = 0          LC = 0             LE = 0
+    CBE = 0         CCE = 0            CBC = 0
+    LBT = 0         ZBT = 50           LCT = 0
+    ZCT = 50        LET = 0            ZET = 50
+    CBEP = 0        CBCP = 0           CCEP = 0
+    T = 0           F = 2.205E+010     A = 0.9916
+    TJ = 298
+ {
+   ; NONLINEAR parameters:
+    BF = 100        BR = 5             NF = 1
+    NE = 1.5        NR = 1             NC = 2
+    IS = 8E-016     ISE = 1E-015       ISC = 1E-017
+    VA = 50         VB = 70            IKR = 0.5
+    IKF = 1E+030    RE1 = 0.2          RC2 = 0.1
+    RBM = 0.5       RB = 1             IRB = 0.25
+    TR = 0          TF = 7.2E-012      ITF = 0
+    XTF = 0         VTF = 1E+030       FCC = 0.5
+    VJE = 0.78      MJE = 0.33         CJE = 5E-012
+    XCJC = 0.5      CJC = 1E-012       VJC = 0.4
+    MJC = 0.33      TJ = 298           XTB = 0
+    XTI = 3         TRE1 = 0           TRE2 = 0
+    TRB1 = 0        TRB2 = 0           TRM1 = 0
+    TRM2 = 0        TRC1 = 0           TRC2 = 0
+    TNOM = 298      VCMX = 10          IBMX = 9.973E+011
+    IBMN = 0        NPLT = 6           NAME = BIP_NPN
+    ANA = ON        MODEL = NPN        RB2 = 0
+    LB = 0          LC = 0             LE = 0
+    CBE = 0         CCE = 0            CBC = 0
+    LBT = 0         ZBT = 50           LCT = 0
+    ZCT = 50        LET = 0            ZET = 50
+    CBEP = 0        CBCP = 0           CCEP = 0
+    PTF = 0
+ }
* Intrinsic transistor nodes are  B E C > 51 55 57
*
* UX PACKAGE EQUIVALENT CIRCUIT WITH TRANSISTOR
 IND  41  42    L=.05NH
 TRL  42  43    Z=66  P=60MIL  K=6.6
 IND  43  44    L=.3NH
 CAP  43  48    C=.04PF
 CAP  43  45    C=.03PF
 IND  45  46    L=.2NH
 TRL  45  47    Z=25  P=10MIL  K=6.6
```

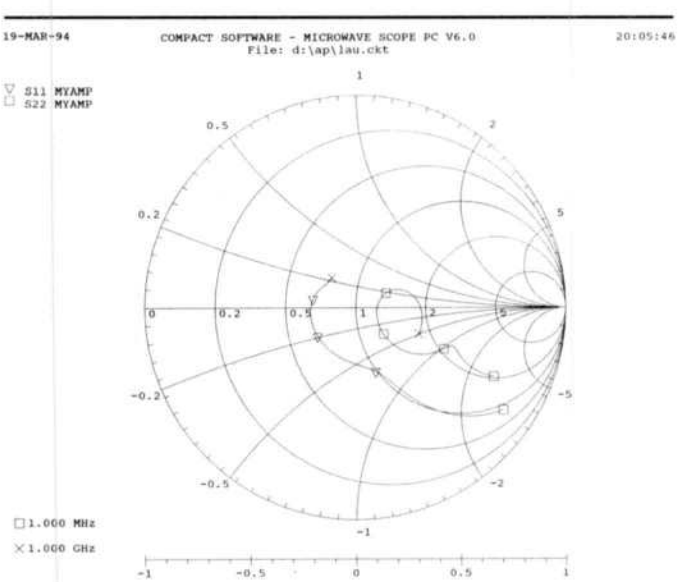


Fig 2

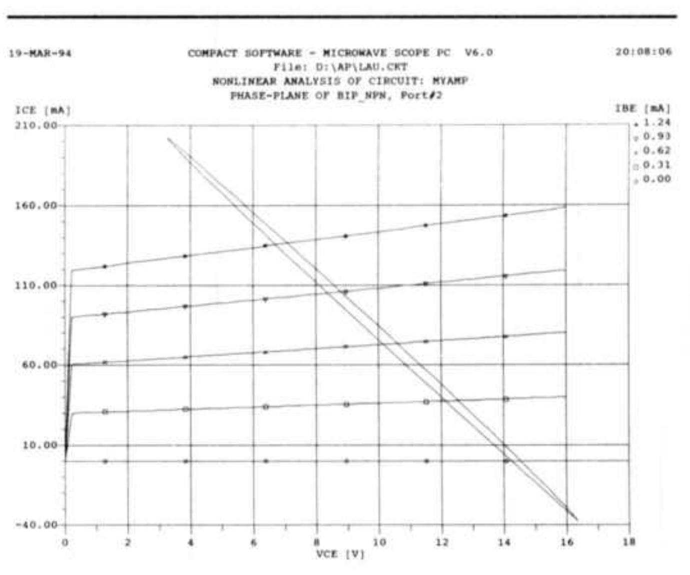


Fig 3

```
IND 47 40    L=.02NH
CAP 45 48    C=.03PF
TRL 48 49    Z=65 P=105MIL K=6.6
IND 49 10    L=.05NH
* connect transistor
RES 44 51    R=0.01 ; BASE      CONNECTION  INTRINSIC
RES 48 55    R=0.01 ; COLLECTOR CONNECTION  INTRINSIC
RES 46 57    R=0.01 ; EMITTER   CONNECTION  INTRINSIC
* BASE 41  COLLECTOR 10   EMITTER 40
********** CIRCUIT ****************
SRX 1   41 R=1 L=1NH C=1NF  ;  INPUT TO BASE
SRL 10  3 R=.1 L=300UH     ;  COLLECTOR VOLTAGE SUPPLY
SRC 10  4 R=1 C=1NF        ;  OUTPUT FROM COLLECTOR
SRL 40 104 R=5 L=2NH       ;  EMITTER FEEDBACK
PRC 104  0 R=22 C=2000PF   ;  EMITTER BIAS
SRL 41  10 R=390  L=20NH   ;  BASE-COLLECTOR FEEDBACK LOOP
SRL 41  0 R=100  L=5NH     ;  BASE TO GROUND
BIAS 3  0 V=13 R=0 {NAME=C_BIAS}
*
MYAMP:2POR 1 {0} 4 {0}
END
*
*************************
* FREQ: Frequency block *
*************************
FREQ
  INTM 3 ; USE 3 IMD products during nonlinear analysis
  TONE 1 195.25MHZ    ; @ 6DBM
  TONE 2 200.25MHZ    ; @ 0DBM
* TONE 3 205.25MHZ    ; @ 0DBM
* 1.300GHZ
 ESTP 1MHZ 1GHZ 100
END
*
**************************
* NEXC: Excitation block *
**************************
NEXC
  P1<H1+H0>=6DBM     ; 1ST TONE
  P1<H0+H1>=0DBM     ; 2ND TONE
  P1<-H1+H2>=0DBM    ; 3RD TONE
END
*
***************************************************
* NVAR: Initial guesses for the state-variables *
***************************************************
*** GENERATE STARTING CONDITION FOR DC FROM SINGLE TONE
ANALYSIS
NVAR
*******************************************
* FINAL SOLUTION: No need for DC Analysis *
*******************************************
DcAnalysis OFF ; Overrides a similar keyword in the CTRL block
BIP_NPN
+ VBE<H0+H0>=? 0.75227599674201?
+ VCE<H0+H0>=? 10.80074299610373?
+ VBE<-H1+H1>=? 7.40722417831421E-002 176.41?
+ VCE<-H1+H1>=? 2.51187652349472E-001 -170.00?
+ VBE<H2-H1>=? 2.68678180873394E-002 -16.84?
+ VCE<H2-H1>=? 1.65844604372978E-001 -14.01?
+ VBE<H1+H0>=? 1.16384156048298E-001 -20.55?
+ VCE<H1+H0>=? 3.59534549713135E+000 158.84?
+ VBE<H0+H1>=? 1.00374892354012E-001 -23.06?
+ VCE<H0+H1>=? 1.54460120201111E+000 158.16?
+ VBE<-H1+H2>=? 4.21312525868416E-002 -24.07?
+ VCE<-H1+H2>=? 1.88302659988403E+000 158.23?
+ VBE<H2+H0>=? 4.02379035949707E-002 139.19?
+ VCE<H2+H0>=? 2.01492458581924E-001 137.04?
+ VBE<H1+H1>=? 6.45341724157333E-002 136.51?
+ VCE<H1+H1>=? 3.35468024015427E-001 136.93?
+ VBE<H0+H2>=? 4.11107093095779E-002 134.79?
+ VCE<H0+H2>=? 1.92502811551094E-001 124.31?
+ VBE<H3+H0>=? 5.60520170256496E-003 -62.06?
+ VCE<H3+H0>=? 3.77730391919613E-002 -46.26?
+ VBE<H2+H1>=? 1.97829566895962E-002 -64.60?
+ VCE<H2+H1>=? 1.11995391547680E-001 -52.15?
+ VBE<H1+H2>=? 2.18342822045088E-002 -66.45?
+ VCE<H1+H2>=? 1.27002149820328E-001 -53.57?
+ VBE<H0+H3>=? 4.71422402188182E-003 -68.19?
+ VCE<H0+H3>=? 4.55842316150665E-002 -37.44?
END
*
***********************************
* OUT: Linear circuit output block *
***********************************
OUT
 PLO MYAMP SK
END
*
***************************************
* NOUT: Nonlinear circuit output block *
***************************************
NOUT
 R1=50 R2=50  ; Use 50 ohms (default) for input and output terminations
END
```

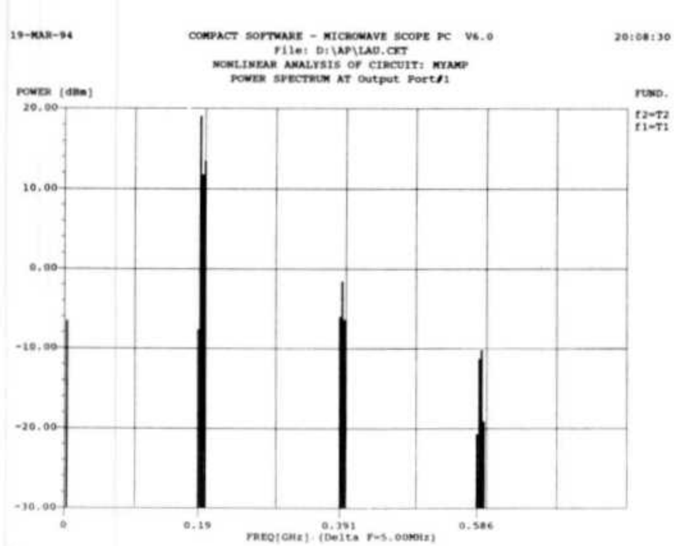


Fig 4

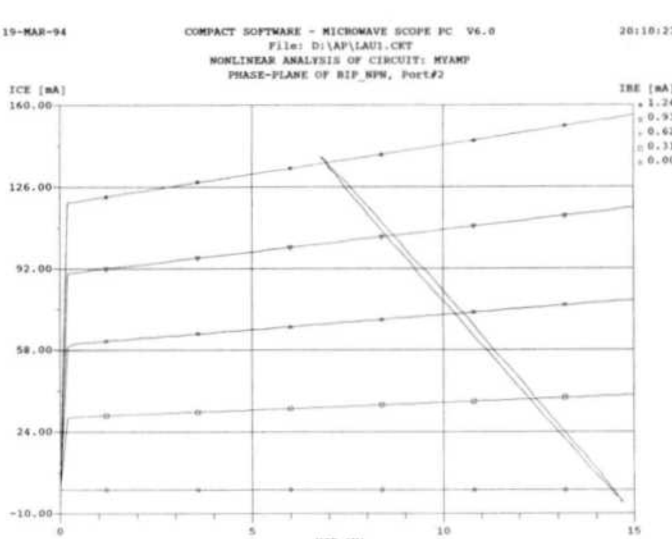


Fig 5

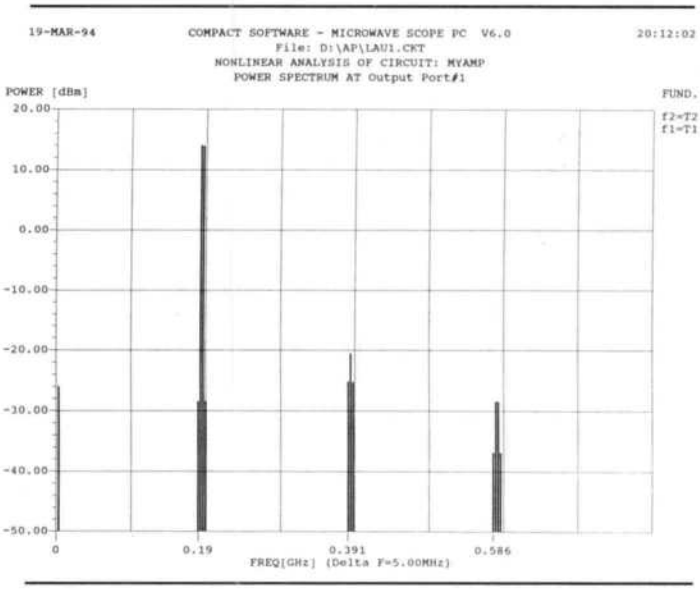


Fig 6

chip temperature. The calculated roughly 65 mA versus 73 mA can be explained by this. Also, we have to look at the tolerances of R_{e1}, R_{e2}, R_2 and R_3 as used by the author.

Fig 3 shows the three-tone load line of the amplifier. The fact that this is not a straight line is due to the fact that there is reactance, mostly capacitance, which is to be expected. However, with the three tones applied to the input, the amplifier is really stretched to its limits. This can be further seen in Fig 4 where the intercept point for three tones is less than 30 dB down and the first harmonics are suppressed only about 22 dB.

By reducing the input signal to two tones, which is frequently used as a benchmark, the output load line in Fig 5 shows much less voltage swing and finally, in Fig 6, the power spectrum at the output now shows an expected clean signal of two tones with the IMD products approximately 45 dB down. This agrees quite well with Lau's measurements. The harmonic content is reduced to about 36 dB. Quite an improvement! This nonlinear analysis is of particular interest because it shows that the three-tone, or multitone, operation is much more nonlinear than the two-tone benchmark indicated.

The 3-tone test shown is the worst case. In practice, the difference in phase between the signals also plays a major role and varies. Therefore, a test set up with the same 3-tone measurement can provide different results depending on the phase of the signals to each other. This can be best observed in a multitone environment like at night listening to frequencies between 6-8 MHz, specifically to a weak station. The background noise from intermodulation distortion products such as cross modulation will vary in intensity due to selective fading (changes in phase). This knowledge is important in order to look into the reproductibility of test results.

Notes

[1]Lau, Z., "RF," *QEX*, March, 1994.

[2]Compact Software, Inc, *Scope Harmonic Balance Simulator Nonlinear Application Manual.*

Building Enclosures for Microwave Circuits

The enclosure is part of the circuit at microwave frequencies. Here's how to make enclosures that work.

By Rus Healy, NJ2L

Many microwave projects are offered as etched circuit boards and board-level kits. Microstrip filters and amplifier circuits, etched on PC-board material, give home builders access to microwave technology that was previously available only to those who were willing to reproduce complex discrete circuits and spend a lot of time tuning them. So now, for reasonable prices, we can build microwave transverters, oscillators, filters, preamplifiers and power amplifiers that work well with few or no tuning adjustments. But one issue remains critical to success with these projects: proper packaging techniques.

The packaging problem is really several individual issues. For one thing, if a board is part of a project that contains many gain stages, the packaging of each board must isolate the stages well enough so that they don't oscillate. It also must provide for low-loss, low-inductance RF input and output connections and appropriate decoupling of power and bias voltages coming from outside the circuit. These aren't difficult matters to address, but they do require some thought and preparation. You just can't mount a microwave circuit into any convenient enclosure, run long leads to it, and expect it to work right.

An effective approach that's much less expensive than use of commercial boxes is to *build* an enclosure to fit the board you're working with. After all, you're building the rest of the circuit, so you may as well build the box!

Unetched fiberglass-epoxy PC-board material and hobby brass make great enclosures. Although PC-board material is widely available, it's a bit less convenient to work with than hobby brass. Brass stock is also widely available (many hobby shops stock K&S brand brass and aluminum stock). It comes in a wide range of widths and thicknesses. A useful size, 0.5-inch wide and 0.025-inch (25 mils) thick, comes in 1-foot lengths that list for 50 cents. Using this material, you can box three or four microwave preamplifiers for a dollar. Now *that's* economical packaging!

Building a Box from Brass

For boxing preamplifiers and other small circuits, half-inch-wide, 25-mil brass strip stock is nearly ideal (other common thicknesses are 20 and 31 mils). Half-inch-wide stock is most convenient because it's the same width as industry standard, four-hole-flange SMA connectors. SMA connectors are best for microwave circuits, because they work well and are widely available in male and female forms for every size of coax commonly used at these frequencies.

Before beginning the brass preparation, clean the brass with a ScotchBrite pad or emery paper. This makes it much easier to solder. Also, drill any necessary holes in the PC board for bias components, transistors, through-hole grounds, and so forth. This is much more difficult after the board is in its case. Look over the circuit and the part-placement diagram carefully so you don't miss anything.

Preparing the sides of the box requires some attention to detail and some common tools. Before you start, collect some brass stock, a drill, the necessary RF and dc connectors, a steel rule, square, file, centerpunch, scribe, utility knife, pencil, and a flat work surface (a piece of plywood is

111 Woodridge Crossing
Henrietta, NY 14467-8923

fine). A drill press and a small vise come in handy, but aren't necessary. You may want to drill and tap the connector holes for no. 2-56 or no. 3-48 screws, but that's not necessary either: gold-plated connectors can be soldered directly to the brass walls. The following procedure assumes you're packaging a typical small microwave project, such as a preamplifier.

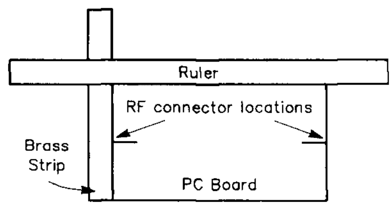


Fig 1—Measuring the first end piece.

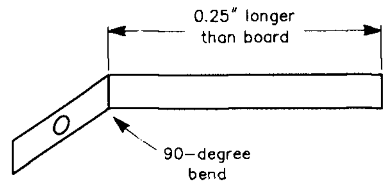


Fig 2—Using a single brass piece to form one end and one side of a box.

Step-by-Step Procedure

1) Start with an end of the circuit board that requires an RF connector. Lay a piece of brass along that edge of the board, with one end of the brass flush with one end of the board. Lay the steel rule along the edge of the board so that it lies flush with the board and crosses the brass. See Fig 1. Mark the brass where the ruler, PC board and brass strip meet. You're marking the brass piece so that it's *exactly* as long as the RF-connector end of the board. Using a square along a long edge of the brass, mark a cut line across the brass with a pencil or scribe.

2) Using a straight edge or square, mark the center position of the RF connector.

3) Measure along the line you just marked, to the center of the brass strip. This is the center point of the connector. Using a center punch, dimple that point for drilling.

4) Repeat steps 2 and 3 for any remaining RF connectors. It's good practice to mount dc connectors on the same sides as RF connectors for ease of mounting the finished assembly inside another case. You may want to standardize on mounting dc connections on the RF-input side, to minimize the possibility of connecting the assembly backwards later. (Stranger things happen every day!) Mark the dc connector hole positions, keeping in mind that the PC board will sit in the center of the finished brass case. Don't locate any feed-through capacitors in places where they'll interfere with the PC board.

5) Lay the steel rule along the cut line you scribed or marked for the edge of the board. Using a utility knife, cut along that line, at first using light-to-moderate pressure. Make sure to follow the scribed line. It's a good idea to cut slightly outside the line, to allow for filing the brass to its final size. Increase the pressure after the first few cuts, once the cut line has been well established. When you've cut about halfway through the strip, grasp the brass between your thumbs and forefingers just on either side of the cut. Flex the brass until it snaps at the cut line, being careful to keep the brass from bending anywhere except the cut line. File the cut edge just enough to clean up the rough edge. Lay the cut piece along the edge of the board to make sure it's the right length and the connector hole centers line up with the appropriate PC-board traces. If the piece is too long, file it to length, keeping in mind that how much material you remove from each end will affect connector positions.

6) Drill or punch the connector-mounting center holes. The holes should be several times the diameter of the connector center pin, and at least slightly wider than the microstrip trace to which the connector's center pin will be soldered. If you're going to solder the connectors in place, drill these holes the same diameter as the connector dielectric. This vastly eases the connector soldering process. If you're drilling the hole, start with a small drill (no. 43 or smaller) and open the hole to its finished size with a larger one. Deburr the hole *by hand* using a drill bit about twice the size of the finished hole. Spin the drill bit between your fingers while applying enough pressure to remove the drilling burrs. This is also a useful technique for clearing the foil away from the ground-plane side of a PC board hole where a wire will pass through the board.

7) If you want to drill and tap the connector-mounting holes (necessary for stainless steel connectors, but optional for gold-plated ones), do the following, otherwise, skip to the next step. Place the connector in the hole and visually center the center pin. While holding the connector in place, mark the flange hole locations on the brass with a pencil or scribe. Remove the connector. Center-punch, drill and tap each hole, then mount the connector using short screws (0.25-inch or less).

8) Wear a heavy cloth or leather work glove for this step. Place the connector in its hole and visually center the center pin. Using a stout soldering implement, heat and solder the connector in place. (Inexpensive soldering pencils sold for stained glass work deliver just the right amount of heat for this job.)

9) Repeat steps 1 through 8 for the opposite end of the board.

10) Measure the sides of the PC board that won't hold any connectors. Add 0.25 inch to the measurement and mark two pieces of brass to this length. Cut them as described in step 5, then file the edges clean and round the corners slightly so that you won't gouge yourself on them later.

11) With the RF connectors mounted to the brass end plates, align their center pins with their respective microstrip traces. Solder the center pins flat to the appropriate traces while holding the board in its final position with respect to the brass. Reheat the center pins and make any small adjustments. *Do not* attempt to adjust the position by twisting the brass or the board without reheating the solder: this can lift the trace from the board, ruining it.

12) Using a stout soldering implement, solder the board to the brass stock along the bottom (ground-plane side) of the board.

13) Repeat steps 11 and 12 for the other connector end of the board.

14) Align the side pieces with the brass end plates that are soldered to the board. This is most easily done by laying the side piece flat on a wooden work surface, then holding the assembly in place above it. Solder the seams formed where the side piece meets the ground-plane side of the board and the brass end plates.

15) Repeat step 14 for the other side plate. Be careful to keep the side plates aligned during this process; the whole assembly should sit flat when you're done.

Tips

Rather than using separate side and end plates, some builders elect to use only two L-shaped pieces of brass, as shown in Fig 2, bending them at 90-degree angles. Each L-shaped piece forms one end and one side of the box. This results in only two seams in the brass enclosure, at opposite corners of the board. This approach makes it easier to make boxes that sit flat and requires making fewer cuts. But it takes extra care to get the bends in exactly the right places so that the connector holes are properly centered.

You should now have an assembly that's relatively inflexible. You can now solder on the board-mounted parts. Although it requires good dexterity and physically small parts, it's possible to build active bias circuits for amplifiers and preamplifiers onto the back side of the board once its enclosure is completed. This isn't recommended for first-time builders, and is unnecessary for boards that use only MMICs, but it is worth the effort in compactness, simplicity and idiot-proofing for complicated projects with lots of separate modules. such as 10-GHz transverters.

Thanks to Zack Lau, KH6CP, whose work has provided me, over the past several years, with a lot of the visual cues I use to make and package microwave circuits.

A Simple Versatile UHF Test Source–0.4 to 1.5 GHz

Generating test signals is a necessity–at UHF, it's a simple task using this circuit.

By John Reed, W6IOJ

We all know the value of a good test source while conducting our UHF experimental projects. While most measurement efforts are best made using sophisticated test gear such as a spectrum analyzer, not many of us have access to such gear so we end up plotting responses the hard way, with a variable frequency source. Actually, it's not as hard as one might expect if the effort is preceded with a little homework to define the minimum number of tests; you may even end up knowing more about the task than if you had used the sophisticated test equipment. If you are an experimenter you may find this simple and reliable UHF signal source a useful tool. The 0.4 to 1.5-GHz frequency range of the source is controlled primarily by a single inductor and capacitor, the total range being separated into bands by the insertion of four different inductors. The inexpensive parts are readily available.

Circuit

The source is essentially a Colpitts oscillator, with the necessary oscillation feedback provided primarily by the transistor characteristics. The key component is the MRF559 transistor, which was designed for use mainly as a 0.5-W, 870-MHz Class-A ampli-

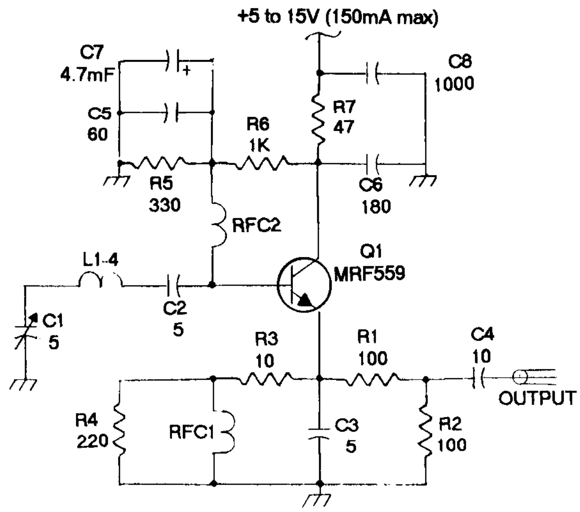
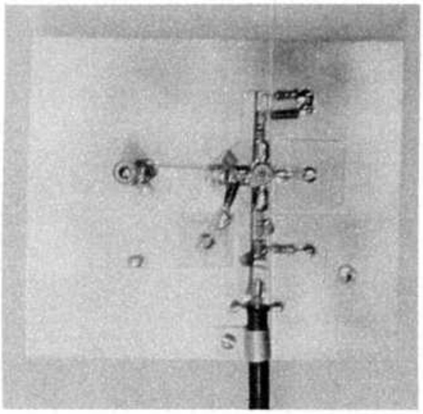


Fig 1—Schematic of the test source. C1 is a 1.6 pF, Sprague GYA5R000. Alternative capacitors are noted in Fig 2. C2 and C3 are 4.7 or 5 pF. Note that C3 is used only in conjunction with L4. C5 and C6 are detailed in Fig 4. L1 through L4 are detailed in Fig 2. R1 can be reduced to a minimum of 10 Ω for increased power output. Results are noted in Figs 6 and 7. RFC1 and RFC2 are 10 turns #24 wire, 1/16 ID, 3/8-inch long. Output is a short piece of RG-58 terminated with a BNC connector. Capacitors are 50-V disc ceramic unless otherwise noted. Resistors are 5%, 1/4-W carbon.

770 La Buena Tierra
Santa Barbara, CA 93111



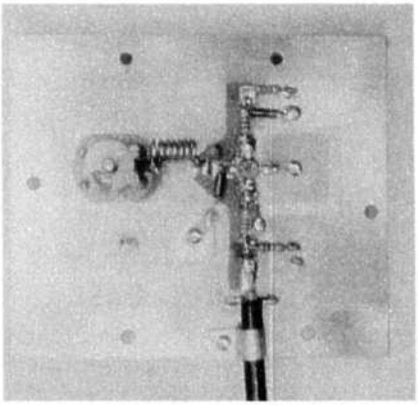
The Radio Shack PC board model.

C1—1.6 to 5-pF Sprague Goodman FILMTRIM, GTA5R000	
L1—$^5/_8 \times ^1/_8 \times$ 0.025-inch brass stripline	0.95 to 1.42 GHz
L2—1 turn #18 wire, $^1/_8$-inch ID (see Fig 5)	0.70 to 1.08 GHz
L3—3 turns #18 wire, $^1/_8$-inch ID	0.55 to 0.80 GHz
L4—7 turns #18 wire, $^1/_8$ inch ID	0.41 to 0.56 GHz

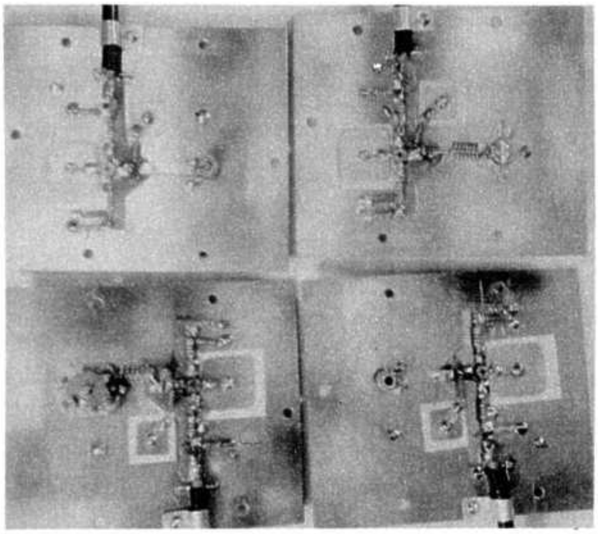
C1—1 to 10-pF piston trimmer (brand not known)	
L1—$^{11}/_{16} \times ^1/_8 \times$ 0.025-inch brass stripline	0.98 to 1.51 GHz
L4—9 turns #18 wire, $^1/_8$-inch ID	0.39 to 0.54 GHz

C1—4 to 10pF miniature air variable (Johnson)	
L1—$^3/_8 \times ^1/_8 \times$ 0.025-inch brass stripline	0.82 to 1.25 GHz
L4—7 turns #18 wire, $^3/_{16}$-inch ID	0.40 to 0.50 GHz

Fig 2—Frequency range of inductor/capacitor combinations.



The variable 4 to 10-pF air capacitor model.



(l-r) (1) Initial reference model, 0.98 to 1.5 GHz; (2) confirmed the FILMTRIM capacitor performance; (3) confirmed the 4 to 10-pF air capacitor performance, and; (4) confirmed the hacksaw-and-file PC board performance.

fier. However, its general specifications, 18-V collector-to-emitter rating, 150-mA continuous collector current, 2-W dissipation and 3-GHz f_t make it a good general-purpose transistor. Also, my experience in other projects has shown it to be a very rugged device.

Referring to the Fig 1 schematic, the base bias voltage is taken from the collector series resistor, R7. This dc feedback tends to stabilize the oscillator for changes in signal feedback, which is minimum at 1.5 GHz and maximum at about 0.42 GHz. With a 10-V supply, the collector current will range from about 70 mA at 1.5 GHz to 90 mA in the 0.4 to 0.5-GHz region. The transistor emitter/collector capacitance is not enough at the lower frequency end to reliably support oscillation. This is corrected by adding the 5-pF capacitor, C3. Note that this is only used on the lower-frequency band. The output is coupled through an emitter resistive divider R1/R2, avoiding frequency-dependent reactance that would likely occur with more efficient inductive coupling. R1 can be reduced to a minimum value of 10 Ω for higher output. This will make the performance strongly dependent upon load reactance and is only recommended in applications where frequency stability is not important. Although 50-V disc ceramic coupling capacitors at C2 and C4 have performed very well in this particular application, chip capacitors may be a more appropriate selection.

The tuning capacitor, C1, is a Sprague FILMTRIM. It was chosen because of its availability and low cost. Although a first-class capacitor, in this application it has two limitations. One, which I consider not particularly important, is the 1.6-pF minimum capacity as compared to 1 pF or less for a piston trimmer. When used in this circuit, the difference results in a maximum frequency of about 1.4 GHz rather than 1.5 GHz. The more significant problem is the capacitor's 180-degree tuning rotation. The tuning range on the high band is over 500-MHz, and this limited tuning rotation makes it very difficult to tune to a particular frequency with reasonable tolerance. I found that an old surplus multiturn piston trimmer was

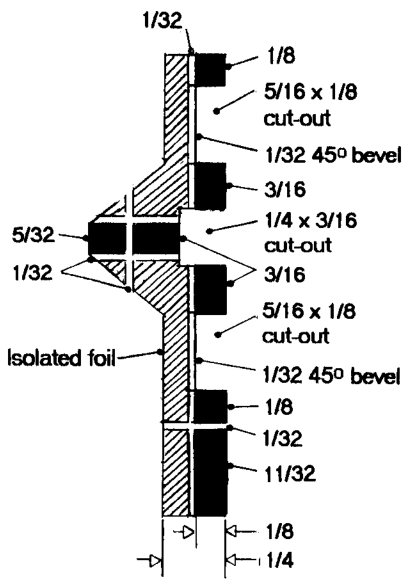


Fig 3—Details of the component PC board. The material is 1/16 glass expoxy, single or double-sided foil (alternative noted in the text and Fig 8). The 1/32 foil gaps can be made using a 24 tooth/inch hack-saw blade. Clamp a straight-edge on the 1/8 pads as a guide. Cut-outs can be made with a standard 8-inch mil-bastard file. Removal of the isolated foils is not necessary. All indicated dimensions are in inches.

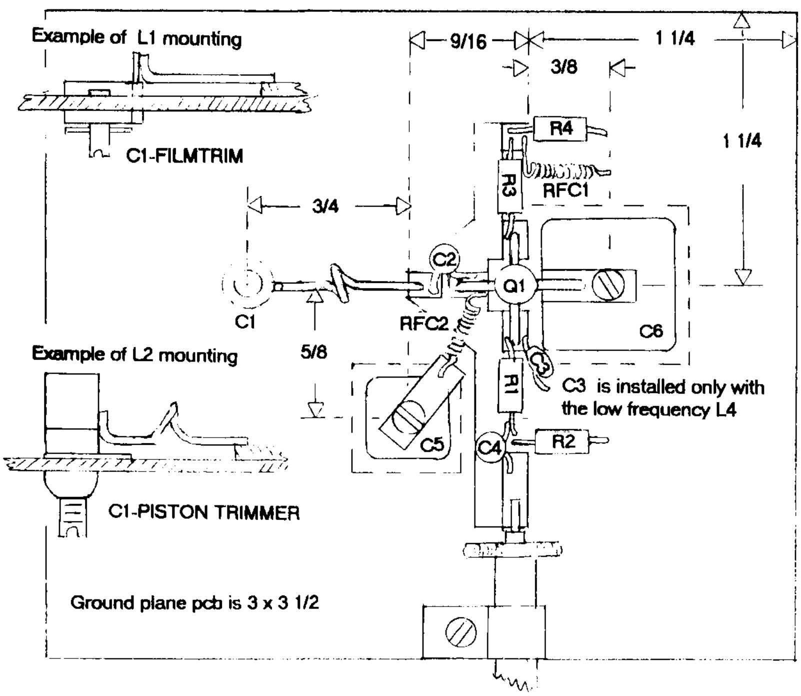


Fig 5—Layout of the components. C7, C8 and R5, R6 and R7 are mounted on the ground-plane PC board bottom.

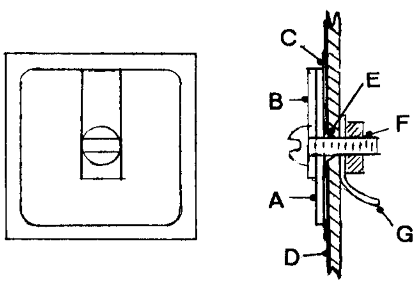


Fig 4—Detail of the feedthrough capacitors.

A—C6 plate, 5/8 × 5/8 × 0.025 brass
B—top solder lug, 7/16 × 5/32 × 0.025 brass
C—paper dielectric, 4-mil
D—ground-plane foil
E—reamed for foil clearance with large drill
F—2-56 screw
G—bottom solder lug

Note: A & B are surface-ground with extra-fine sandpaper to ensure flatness. C5 is identical except the size is smaller, 3/8 × 3/8 × 0.025

much more effective. The capacity range of this particular trimmer measured 1 to 10 pF. A third type that I have used is an old Johnson 4 to 10-pF miniature air trimmer. It has a shaft that can be fitted with a knob or slow-drive to overcome the 180-degree rotation limitation. It has a high frequency range of 0.8 to 1.2 GHz when interfaced to the circuit with a 3/8 × 1/8-inch stripline. This type of capacitor is a particularly good choice for the low-frequency band. Data shown in Fig 2 summarizes the source performance using the three different types of capacitors.

Layout

The layout is unique in that it uses a small glue-down PC board for the mounting of all components except the tuning capacitor, C1, and a few dc operating parts, which are mounted on the ground-plane PC board under-side. The principal advantage of this method is that it offers a solid top-side ground plane. This in turn permits (1) near-zero lead length component mounting, (2) a simple home-brew collector and base circuit by-pass capacitor design and (3) use of single-sided PC boards. The component board can be made by any conventional method. As an example, most of my models use boards made using a Dremel Moto-Tool. My latest model, detailed in Fig 3, is made using only a hacksaw and file. It is simpler, and the performance is equivalent to the others.

My first assemblies of the home-brew feedthrough by-pass capacitors, detailed in Fig 4, used mica insulation. However, I found that 4-mil recycled computer paper worked just as well in this particular application. It is important that both the plate and the top brass soldering lug be surface-ground flat with extra-fine sandpaper. The lug adds minimal additional inductance while bringing the surface close to the transistor lead for soldering.

A detailed parts layout is shown in Fig 5. Following is the recommended assembly schedule:

1) cement the component PC board to the ground plane (I use Elmer's Clear Household Cement).

2) assemble C5-6

3) solder in Q1 after bending the collector lead end so that it clears the C6 mounting screw. In bending any transistor lead make certain there is no bending at the plastic interface. (I always bend the remaining lead ends

slightly to facilitate unplanned removal of the transistor.)

4) mount the remaining components, leaving L and C1 until last.

Performance

Detailed performance in the 0.8 to 1.5-GHz band is shown in Fig 6, and the 0.38 to 0.54-GHz band in Fig 7. Performance levels in the other two bands range between those shown.

As an afterthought, I was curious how an assembly made using Radio Shack PC board (276-1499) would work. The material is 4½×6⅜×1⁄16-inch with double-side foil. The dielectric is a light-colored, non-fibrous plastic with good foil adhesion. A component PC board as detailed in Fig 3, and a 3×3½-inch ground-plane PC board were made from this material and assembled as a test source. Tests indicated no difference in performance on the low-frequency band and, as shown in Fig 8, only small differences occurred on the high-frequency band.

The UHF test source is used with a 2¾×2¼×⅞-inch shield covering the top side. In addition, the assembly is generally placed in an enclosed metal box together with any attenuators required when making low-level receiver measurements.

Parts Suppliers

MRF559—RF Parts Co, 435 South Pacific St, San Marcos, CA 92069, tel 800-737-2787.

FILTRIMs—DigiKey Corp, 701 Brooks Ave S, PO Box 67, Thief River Falls, MN 56702-0677, tel 800-862-5432.

Brass—Hobby shops usually stock small sheets of brass. □□

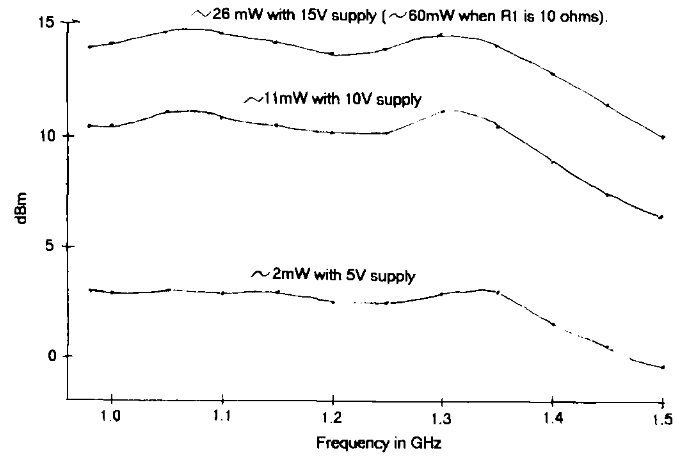


Fig 6—Source performance in the high-frequency band. C1 is a 1 to 10-pF piston trimmer.

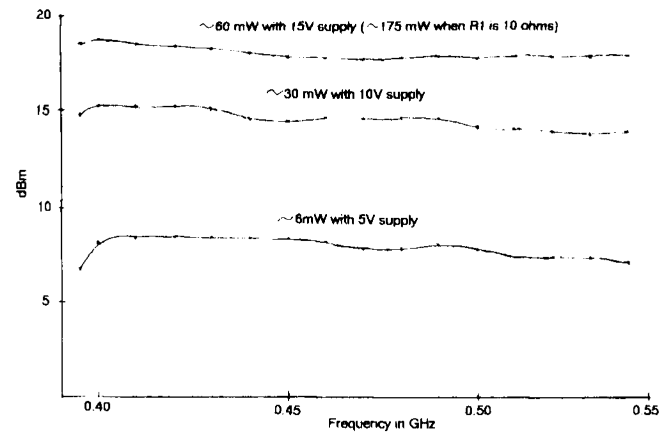


Fig 7—Source performance in the low-frequency band. C1 is a 1 to 10-pF piston trimmer.

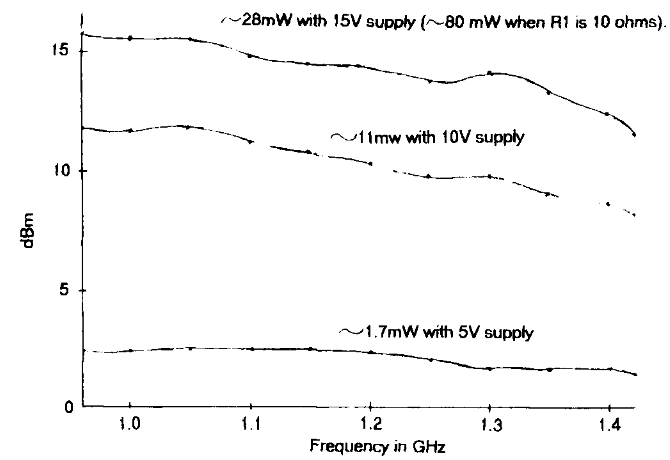


Fig 8—Performance of the RS model in the high-frequency band. All parts (including the PC board) are from Radio Shack, except for the transistor and the 1 to 10-pF C1 piston trimmer. Performance in the low-frequency band is almost identical to that shown in Fig 7.

On the Traveling-Wave Linear Dipole

What's the story with these dipoles with resistors in them? Do they deliver the promised performance?

By John S. (Jack) Belrose, VE2CV

Broadband antennas include antennas designed to present an impedance that remains near-constant over a wide band of frequencies. But low VSWR over a wide band of frequencies does not necessarily imply an effective antenna, since in practice some antennas are rather inefficient on purpose, eg, resistance-loaded antennas, to achieve a wide bandwidth. Wide bandwidth is a requirement for frequency-agile commercial and military radio systems. Besides this, the brochures published by companies manufacturing and marketing antennas often give no information or only misleading information on pattern or gain. There is a danger, therefore, that myths can be created in the absence of numerical analysis and objective measurements.

Introduction

The standard dipole's nearly sinusoidal current distribution is a standing wave, produced by two nearly equal amplitude traveling waves moving in opposite directions along the antenna's wires (the arms of the dipole). We will refer to the above waves as primary and secondary traveling waves where: (a) the primary traveling wave originates at the dipole's feed and moves toward the ends of the wires; and (b) the secondary traveling waves arise as a result of reflection of the above wave from the wire ends and moves from there back toward the feed. Interference between the above waves, at the feed, produces the standard dipole's undesirable large variation of impedance as a function of frequency. Traveling-wave antennas represent an attempt to solve the above problem by the deliberate introduction of attenuation of the traveling waves and hence, at the feed, reducing the amplitude of the secondary traveling wave, leaving essentially only the wave traveling away from the feed. It is possible to achieve this through the use of a long antenna wire and relying on radiation to supply the required attenuation. In the antenna considered here, the above attenuation is obtained by the introduction of resistors into the antenna's structure. But antennas with resistive loading tend to be inefficient because power is absorbed in the resistors.

In an earlier article I considered a terminated folded dipole (TFD) antenna.[1] But linear dipoles are more commonly used. A traveling wave distribution of current can be produced by inserting a suitable resistance one-quarter wavelength from the end of the antenna.[2] Such an antenna will have a traveling-wave distribution of current between the feedpoint and the resistor, and a standing wave of current from the resistor to the end. Such an antenna is popularly known as the Australian dipole, since it was devised by engineers there.[3,4,5] This antenna is a traveling-wave (linear) dipole (TWD). The TWD was commercially developed by Antenna Engineering, Australia (AEA), but versions of it are, or were, available from several manufacturers (Harris RF Communications Group in the US and C&S Antennas Limited (CSA) in the UK).[6]

Notes appear on page 25.

ARRL Technical Advisor
17 Tadoussac Drive
Aylmer, QC Canada J9J IGI

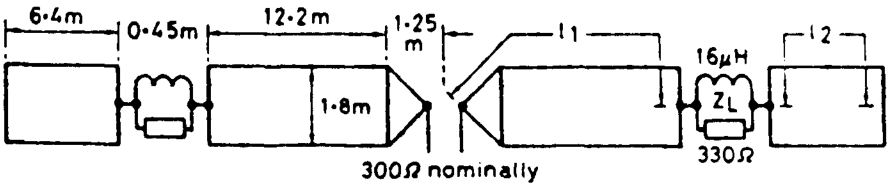


Fig 1—The "Australian dipole," a traveling-wave linear dipole.

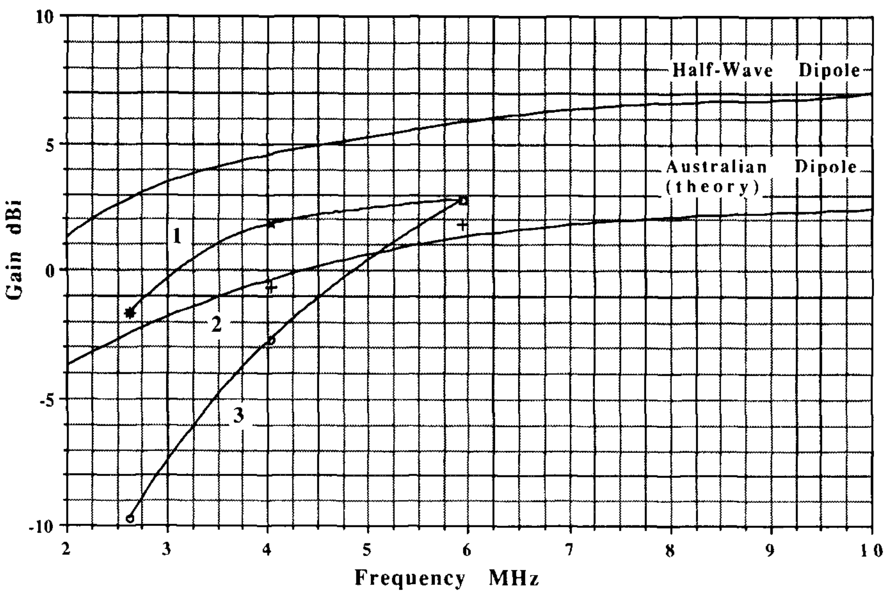


Fig 2—Comparison (gain versus frequency) between the measured NVIS gains of three commercial versions of the TWD antenna, the theoretical gain of Fig 1's, antenna and a half-wave dipole, all antennas at a height of 9 meters (30 feet) over sandy (poor) ground.

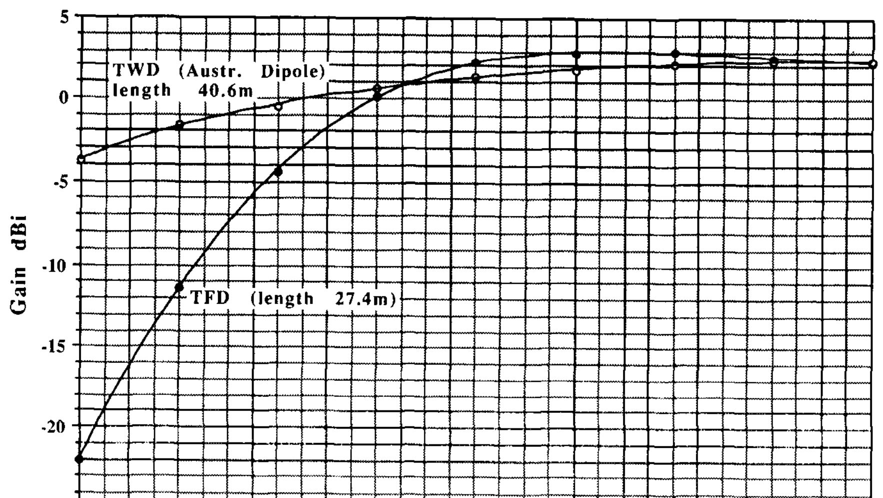


Fig 3—Comparison (theoretical gain versus frequency) between a TWD and a TFD, both dipoles at 9 meters (30 feet) over sandy (poor) ground.

The Australian dipole is a multiwire dipole antenna, multiwire to make it effectively "fatter," and so broaden the bandwidth of the dipole that is resistively loaded by inclusion of a resistor in series with each arm of the dipole (see Fig 1). This version of a TWD antenna has an inductor across the resistor, to "optimize" performance over the band.

Measured and Calculated Gain

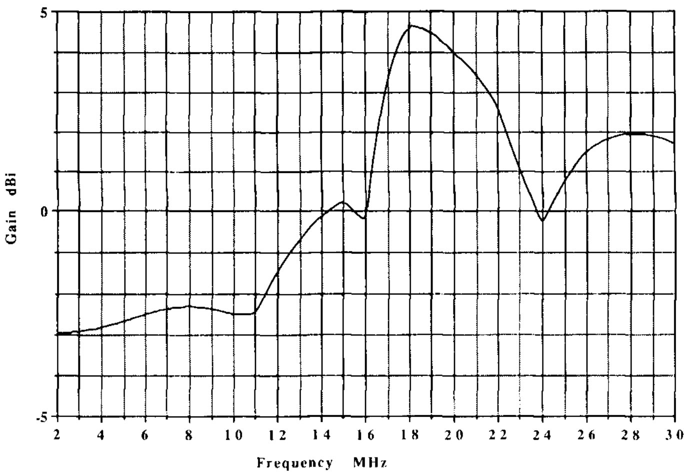
The antennas we tested (AEA Model TWD 215, Harris Model RF 1933 and CSA Model WB DSM) had series resistors in each arm of the dipole of about 300Ω and a nominal input impedance of about 300 Ω. The antenna was fed by coax employing a 300:50-Ω balun.

The dipole end, $\ell 2$, provides a reactance against which the resistance is "terminated," but this termination is only effective when the dipole end length is near to a quarter wavelength. Notwithstanding, the antennas exhibited a good VSWR versus frequency (nominally better than 2:1) over a broad bandwidth. No gain figures are given in the technical information available from these companies, but the TWD's radiation efficiency is claimed (by AEA) to be better than a terminated folded dipole. Austin and Fourie and Royer have modeled traveling-wave dipole antennas and given impedance, VSWR, and gain versus frequency.[7,8] Extended versions of the TWD have been proposed to improve the antenna's radiation efficiency at frequencies below about 4 MHz, but this creates problems associated with achieving broadside directivity over a wide band of frequencies. Both of these researchers have independently suggested using two traveling wave dipoles on the same mast to improve performance. Note that companies that manufacture the antenna, based largely on VSWR, claim operational bandwidths of 1.5 to 30 MHz for this type of antenna.

I have rather rigorously modeled the Australian dipole (Fig 1's antenna) using ELNEC/MaxP option to calculate the free-space gain and pattern. The azimuthal pattern does break down into lobes, beginning with the development of a null in the broadside direction at a frequency of 12 MHz (see below).

In Fig 2, I have plotted the near-vertical incidence (NVIS) gain measured for these three commercial TWD

Fig 4—Free-space gain versus frequency for the Australian dipole (Fig 1).



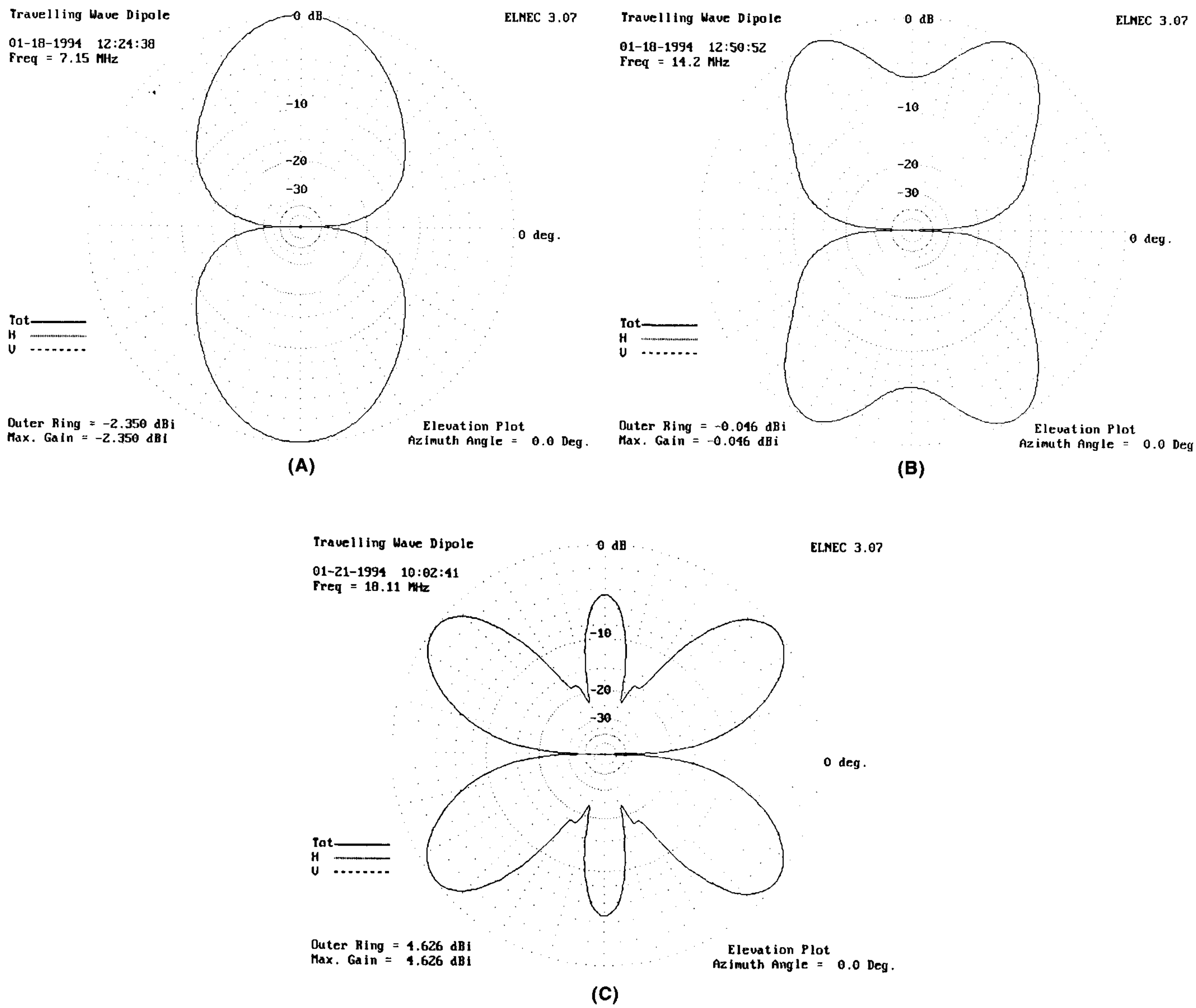


Fig 5—Free-space azimuthal pattern for the Australian dipole (Fig 1), for 7.15, 14.2 and 18.11 MHz.

antennas. (See Note 8.) The curves that I have drawn through the measured gain values for the three TWD dipoles, labeled 1, 2 and 3, can be compared with the theoretical gain for the Australian dipole. As noted above, I calculated the free-space gain for this antenna, gain with respect to a dipole, and I then referenced this gain to the theoretical gain for a half-wave dipole (dBi) over poor ground (conductivity 1 mS/m, dielectric constant 13) calculated by NEC-3. All test antennas were over sandy (poor) ground at a height of 9 meters (30 feet). The gain of a half-wave dipole is also shown, which increases with frequency since the height of the antenna, in wavelengths, increases with frequency.

The curve for TWD dipole 2 is actually our theoretical curve for the Australian dipole, since the data points for this particular antenna lie approximately on the theoretical curve. The data points for TWD dipoles 1 and 3 lie above and below our theoretical curve.

It was noted above that the gain for the Australian dipole was said to be better than the gain for a terminated folded dipole. On Fig 3 I have plotted the theoretical gains for a 40.6-meter (133-foot) TWD compared with the previously calculated gain for a 27.4-meter (90-foot) TFD antenna. Both antennas are at 9 meters over poor ground. The gain difference at frequencies less than 5 MHz is due in part to the difference in the antenna lengths, but if this difference is taken into account, the gain of the TFD decreases more rapidly with decrease in frequency than does that of the TWD.

In Fig 4 I have plotted the free-space gain for the Australian dipole (the antenna of Fig 1) over the entire 2 to 30-MHz band.

Radiation Patterns

The free-space azimuthal patterns at 7.15, 14.2 and 18.11 MHz are shown in Fig. 5. The antenna exhibits broadside (dipole like) directivity between 2 and 10 MHz and VSWR <2:1. As noted above, the antenna starts to develop a null in the broadside direction at about 12 MHz; the null reaches a maximum at 16 MHz, about 15 dB below maximum gain. The gain and VSWR peak at about 18.5 MHz, gain 4.6 dBi, VSWR 3.4:1. At this frequency $\ell 2 = 0.45\ \lambda$ and $\ell 1 = 0.89\ \lambda$. The resistance is poorly "terminated" (ideally $\ell 2$ should be $0.25\ \lambda$). The radiation pattern is multilobed and very frequency dependent between 17 and 30 MHz.

Conclusions

As noted above, Altshuler showed that the insertion of an appropriate resistor a quarter of a wavelength from the end of a dipole produced a virtually constant impedance over about an octave bandwidth, but nothing was said about radiation pattern (see Note 2). I have calculated the free-space gain, and related this gain to measured NVIS gains for several commercial versions of the TWD; and I have calculated the radiation pattern and VSWR over the 2 to 30-MHz band.

For the radio amateur who wishes to work whoever he can hear, the azimuthal pattern of the antenna is unimportant. However, for point-to-point communication the azimuthal pattern is a fundamental parameter of concern. Since more than half of the available power is dissipated in the series resistances, I view this as undesirable. If one runs a kilowatt, one will certainly be heard when one calls, but 500-W noninductive resistances are expensive, and unnecessary, since instantaneous frequency agility is not a requirement for the amateur in radio.

Besides which, broad-band antennas are more prone to receiver-overload problems, since strong adjacent-band signals can contribute to receiver intermodulation problems. Narrow-band, efficient antennas, having a frequency independent azimuthal pattern, are a distinct advantage for the radio amateur, and are a requirement for point-to-point communication links.

It is interesting to note that Altshuler, who devised the TWD, has recently revisited this type of antenna, and has devised a new form of traveling-wave dipole by replacing the resistor by a modified folded dipole.[9] This antenna is not as broad band as the resistance loaded antenna, but now power previously dissipated in the resistor is instead radiated. The peaks and nulls can be controlled by adjusting the height accordingly, and it is possible to achieve near hemispherical coverage.

Acknowledgment

I work for the Communications Research Centre, Ottawa, and have access to instrumentation and facilities that are not available to most radio amateurs.

Notes

[1]Belrose, J. S., "Terminated Folded Dipole," *QST*, Technical Correspondence, May 1994.

[2]Altshuler, E. E., "The Traveling-Wave Linear Antenna," *IEEE Transactions on Antennas and Propagation*, AP-9, 1961, pp 324-329.

[3]Hawker, P., Technical Topics, *Radio Communication*, June 1987, pp 406-407 (also see Technical Topic June 1974 and September 1984).

[4]Guertler, R. J. F., and Collyer, G. E., "Improvements in Traveling Wave Dipoles," *Proceedings IREE Convention*, Melbourne, 1973, pp 70-71.

[5]Treharne, R. F., *Multipurpose Wholeband HF Antenna Architecture*, J. Electrical and Electronic Engineering (Australia), 1983, pp 141-152.

[6]Collins, B. S, and Phillips, B. R., "A Wideband Transportable Antenna for NVIS Links," *IEE Conference Proceedings* No. 284, 4th International Conference on HF Systems and Techniques, London, April 1988, pp 156-158.

[7]Austin, B. A., and Fourie, A. P. C., "Numerical Modelling and Design of Loaded Broadband Wire Antennas," *IEE Conference Proceedings* No. 284, 4th International Conference on HF Systems and Techniques, London, 1988, pp 125-129.

[8]Belrose, J. S., Royer, G. M., and Petrie, L. E., "HF Wire Antennas over Real Ground: Computer Simulation and Measurement," in AGARD Lecture Series No. 165, *Modern Antenna Design Using Computers and Measurement: Application to Antenna Problems of Military Interest*, September 1989, pp 6-1 to 6-30. Available from NTIS, Springfield, VA (Ref NASA Access No. N90-17932).

[9]Altshuler, E. E., "A Monopole Antenna Loaded with a Modified Folded Dipole," *IEEE Transactions on Antennas and Propagation*, AP 41, July 1993, pp 871-876.

Digital Communications

Harold E. Price, NK6K

Are You Being Served?

I recently received a request for comments on various matters in the amateur radio satellite world. Many of the questions had an underlying context of "are the users getting what they want," which of course leads to another question, "are the users getting what they paid for?"

Many users, non*QEX* readers for example, have no idea how much work is donated by implementors (*QEX* readers). I'll use two examples of big projects that I happen to have handy, the Microsat software and the NOS TCP/IP software. Let me say at the outset that there are many other nonsoftware aspects to these projects that I'm not going to mention here, and I don't mean to downplay the significance of that work. The software is readily available for scrutiny, and the numbers derived from the software effort alone are mind boggling enough to prove the point.

Many of the software cost estimation concepts I'll use below are from *Space Mission Analysis and Design*, James R. Wertz and Wiley J. Larson (editors), Kluwer Academic Publishers, 1991, ISBN 0-7923-0971-5. This book is highly recommended for giving you an insight into the problems, from the complex to the mundane, involved in designing a satellite.

One metric used to describe software development effort is the resulting number of lines of source code. While this sounds simple, defining a "line" is not. Nor are all lines equal. For a quick and dirty estimate of C code, however, you can simply count the number of lines containing a ";" character. This is an overestimate, since you also get variable declarations that take one fourth the effort to write. Still, it is a place to start.

Microsat software required 11 person years, NOS 19 person years.

For the Microsat project, I'm including the operating system, I/O drivers, file system, AX.25 protocol, telemetry program, "BBS," test programs and the ground loader. Though some of this is space code and some is ground code, I'm going to use the ground-costing metrics for all of it. For NOS, I'm using all of the source in the JNOS 1.10x3 release.

The Microsat software has 23,500 lines of C code, and 7,000 lines of assembler. NOS has 45,000 lines of C code, and 2,500 lines of assembler. Applying conversion factors of 4:1 for include files to lines of C, and 8:1 for assembler to C, the total equivalent lines of C is 24,800 for Microsat, and 43,000 for NOS.

The Wertz book gives a cost per thousand lines of ground station code written in Ada as $172,000, or $172 for a line of code. The is a "deployed" line, however, after it is specified, written, tested, documented, shipped, installed, and maintained, all to government standards. Wertz supplies a conversion of 0.8 for commercial work, and 1.67 for C. I've further reduced these numbers by 75%, since we in the Amateur Radio field tend to skip several specification and test steps. Applying all the conversions, the development "cost" of the Microsat software is $1.4 million, the cost of NOS is $2.4 million.

A standard industry figure for the cost of a person-year is $130,000. This is not the worker's actual salary, of course, it includes overhead such as taxes and benefits, the office costs, phone, power, heating, etc. Using the above costs and this labor rate, the Microsat software required 11 person years, NOS 19 person years. The people who do this kind of work in the amateur world are self-selected to be at the top of the productivity scale, however. Those people who are elsewhere on the productivity scale usu-

5949 Pudding Stone Lane
Bethel Park, PA 15102
email: nk6k@amsat.org (Internet)
71635,1174 (Compuserve)

ally don't finish a free project of this scope because of the time it takes; or the project never works and doesn't see the light of day. Considering that, and the number of people involved for a number of years, the person-year figures are not that far off.

The people who commit at this level to working for the amateur community seldom stop to think of the effort in these terms. If they did, they probably wouldn't start. This also helps explain why the members of even large organizations can't always get what they want. Assume, for example, that a 5000-member organization charges $20 per year in dues. If *all* of that went to a software development fund, that $100,000 could barely scratch the surface of what some of these projects would cost, if the users were paying for everything. Now add in the hardware, the hardware labor and project management costs. As a past director of both TAPR and AMSAT, I can tell you that very little of the yearly dues goes into projects. Most of it goes into the cost of preparation and mailing of the newsletters and other membership services. At best, with background dues, the users pay some of the minor costs, such as travel and telephone expenses.

Even big-budget and big fund-raising projects, like P3D, only cover the actual cost of the hardware, which is usually heavily discounted by hams and other friends in the industry. For example, the layout work for the eight-layer surface-mount CPU board for the first four Microsats was done at a professional shop. It was bid at near cost and actually ran well over the number of hours estimated for the job. That friend probably lost money, as did the company that did the surface-mount soldering. QualComm donated several days of lab space and the time of a surface-mount rework specialist. Martin Marietta donated several days of thermal vacuum chamber time. A radiation lab in Canada is donating time for chip testing for P3D. You get the idea.

Users are therefore way down on the curve in terms of being able to directly dictate what they get. In Amateur Radio's version of the golden rule, he who codes (or designs hardware) for free, rules. That's life. Still, users do play a very important role. In the satellite world, they buy the parts that can't be obtained by back-channel friends or the junk box. They provide the initial funding that is later used as leverage to get other money or in-kind donations. Without that background level of funding, some projects would not attain critical mass.

P3D will be big, loud and visible for 12 hours every day

This is missing in the packet radio world. We haven't seen much group buying of shared network resources. Yes, individuals do expend much time and effort on local resources, such as a BBS or network node. Still, not much emphasis is placed on the shared network. We aren't much further along today than we were in 1987. Back when I had access to the number of units sold by TAPR licensees, and using other informed sources, I estimated there had been 100,000 TNCs sold. This was by 1989. Lately, one of the TNC manufacturers is claiming 100,000 units sold by itself alone. If we use just 150,000 units as a reasonable estimate of current activity, and an average cost of $140 per unit, we're looking at a $21 million dollar investment. Just image if we had spent an additional 10% on network resources. What could we do with $2 million? There are several large multistate linked-voice repeater systems with five- and six-figure annual budgets. Why has there never been a packet radio network group at this level of effort? My mailbox is open.

There is a brighter side to "developers rule." Given a choice between two interesting alternatives, the developers are likely to choose the one that has the most potential usage. It is important to keep the want list loaded, and to comment on what is desired, my comments above not withstanding. One of the big projects coming up, the so-called "user" RUDAK on P3D, is such an opportunity.

The Next Big Project

The next big digital project will be the international AMSAT Phase 3D satellite. For those who haven't been following P3D development closely, this will be a larger satellite than we've built before, 2.5 meters in diameter, weighing 400 kg. It will have high-gain antennas and high power output, as much as 250 watts on some frequencies. The satellite will be in an inclined, highly elliptical orbit with an apogee of 50,000 km. In short, it will be big, loud and visible for 12 hours every day. Another new feature is the IF matrix. All receivers mix to a 10.7-MHz IF. All transmitters are fed with 10.7 MHz. Rather than a few fixed input and output transponders as in past satellites, P3D can route any of the receivers to any of the transmitters. The uplink frequencies are currently planned to be in the following bands: 146 MHz, 435 MHz, 1260 MHz, 2400 MHz and 10.5 GHz. The downlinks are 146 MHz, 435 MHz, 2400 MHz, 10.5 GHz and 24 GHz.

There are several interesting digital aspects. A total of 250 kHz of uplink bandwidth and 150 kHz of downlink are set aside for digital modes. There are two onboard computers that will make use of this space. The original P3D RUDAK project is run by a team in Germany. This is a follow-on to the hardware design used in RS-14/AO-21. The group is targeting high-speed applications (56 kbit/s and above) as well as DSP-based weak-signal modems. The goal is to push the envelope in terms of new protocols and modulation schemes. As a test bed for new technology, this project plans to be out in front of the user community: support of the current installed base of modems and TNCs will not be a priority.

A proposed second RUDAK *will* have support of the current user base as a priority. Lyle Johnson, WA7GXD, is Project Manager. The project is currently called RUDAK-U, for User. Assisting are the usual suspects, Jeff Ward, GØ/K8KA; Harold Price, NK6K; Peter Guelzow, DB2OS; and Chuck Green, NØADI. This computer will connect to the IF matrix with as many as 10 uplinks and four downlinks. 20 W of output power will be split between the four downlinks, a total of 2000-W EIRP on the higher frequencies.

We plan to use the K9NG/G3RUH-style 9600-baud FSK modulation as one of the standards. Some of the modems will be DSP based, allowing for upgrades over the projected 10-year life of P3D. The hardware will probably be V53 based (80x86 compatible), with 16 DMA I/O channels and up to 16 MB of error-correcting memory. P3D will be in a higher radiation environment than the current digital satellites, so large memories are harder to build and harder to pay for.

Other than that, we're open to suggestions on the best uses for this facility. We have some ideas, of course. The CPU will be compatible with the current Microsat/UoSAT software, so P3D will be PG/PB compatible at launch. These protocols were designed with a LEO store-and-forward mission in mind, however. P3D will be a slower moving target, visible to more people for a longer time. The current LEO protocols are contention-based. We plan on implementing a managed/reservation scheme as well. Several other ideas come to mind. A small number of ground-based servers could offer gigabytes to the network, using RUDAK-U as a buffered bit regenerator. Send in your suggestions, we'll be happy to see them.

GPS

Another P3D project is the on-board GPS receiver. Most of us are used to thinking of the Global Positioning System in terms of ground positioning. It works just as well from orbit. The SSTL-built PoSAT commercial spacecraft has been on the ham bands recently. It also carries a GPS receiver that has worked well. In addition to doing on-board orbit determination, the P3D GPS team hopes to do attitude determination as well, using multiple antennas. Look for this project to keep that team occupied for some time to come. Tom Clark, W3IWI, Lyle Johnson, Chuck Green, and Bdale Garbee, N3EUA, are back for this one, as well as Paul Barrow, VE6ATS, Eric Cottrell, WB1HBU, Doug Varney, WA1UVP, John Conner, WDØFHG, and others. This project may spin off into other useful items; stay tuned.

Meteor Bounce

A group of hams is working on a meteor-bounce experiment. It's been done, you say. Yes, but this group is pointing their antennas down to see the meteors. The UNAMSAT satellite is about ready for launch. Currently planned for late June on a Russian launcher, the Mexican satellite will carry a meteor-bounce experiment along with its amateur radio transponder. Led by David Liberman, XE1TU, and built by students XEØIKQ, XEØMMF, XEØLFD, XEØJMC, XEØRAI, XEØIKZ and XEØJPM, the satellite will attempt to determine the speed, direction and altitude of meteors, based on timing and the Doppler shift of the returning signal. A 50-W, 40-MHz transmitter will be used in short pulses to detect meteor trails. This spacecraft is based on the AMSAT-NA Microsat design, specifically the ITAMSAT version. It will use 1200-baud PSK, PG and PB.

Dayton

Every column in every ham magazine this month will mention Dayton. I wasn't there, but I didn't want to feel left out. I heard it rained. Lyle Johnson, by the way is now a grandfather. It would be impolite to point out that Heather is also a grandmother. Lyle is no longer a Young Turk. □□

Upcoming Technical Conferences

Central States VHF Conference

July 29-30, 1994, Wilson World Hotel, Memphis, Tennessee

Contact: Dave Meier, N4MW, 3205 Covington Pike, Memphis, TN 38128, tel 901-382-4912.

Radio Amateurs of/du Canada—First National Convention

July 29-31, 1994, Marlborough Inn Convention Centre, Calgary, Alberta, Canada.

Contact: Gerry Shand, VE6BLI, 55-51551 Range Road 212A, Sherwood Park, AB, T8B 1B2 Canada, tel 403-922-2099, fax 403-438-4398.

Events: Special event station CI6RAC; exhibitions, dealer displays, seminars, ladies program, children's program, RAC general meeting, QCWA meeting, QSL bureau, banquet with keynote speaker. Contact Gerry Shand, VE6BLI, for more information.

1994 ARRL Digital Communications Conference

August 19-21, 1994, Thunderbird Hotel and Conference Center, Bloomington, Minnesota. Sponsored by the TwinsLAN ARC.

Contact: Paul Ramey, WGØG, 16266 Finland Avenue, Rosemount, MN 55068, tel 612-432-1149 evenings and weekends. Or, Carl Estey, WAØCQG, 276 Walnut Lane, Apple Valley, MN 55124, tel 612-432-0699. Internet: estey@skyler.mavd.honeywell.com; packet: WAØCQG@WAØCQG.#MSP.MN.USA.NA.

Events: Friday afternoon, registration, Hospitality Suite and informal demonstrations. Saturday, presentation of technical papers, buffet luncheon (included), "birds-of-a-feather" forums (digital data methods, DSP developments, HF data methods, TCP/IP developments, ARRL committee updates and high-speed data transfer), Hospitality Suite, informal demonstrations, optional buffet dinner, technical showcase—TAPR special interest groups and ADRS DSP presentation. Sunday, Hospitality Suite, informal demonstration and conference wrap-up. For those not attending the Conference, outings of interest to all family members are being arranged.

Call for papers: Anyone interested in digital communications is invited to submit papers for this conference. Papers can be on any aspect of digital communications. The deadline for receipt of papers is **June 20, 1994**. Papers can be in electronic or camera-ready form, and should be sent to Maty Weinberg, ARRL, 225 Main Street, Newington, CT 06111, Internet: lweinber@arrl.org.

Registration: Conference registration is $45 and includes Saturday's luncheon and a copy of the *Conference Proceedings*. Registration deadline is August 12, 1994. Saturday evening's buffet is an additional $20. Please call Paul Ramey, WGØG, for registration forms.

Hotel/Etc: Best Western Thunderbird Hotel and Convention Center, 2201 East 78th Street, Interstate I-494 at the 24th Avenue exit, Blooming, MN. Free shuttle service is available to Minneapolis/St. Paul international Airport and Mall of America. You can call the DCC Hotline at 800-726-6715 to make reservations and get more information on area events. Be sure to ask for Cathy Thomas.

Microwave Update '94

September 22-24, 1994, The Inn at Estes Park, Estes Park, Colorado.

Contact: William McCaa, KØRZ, PO Box 3214, Boulder, CO 80307-3214, tel 303-441-3069..

Papers: All information on amateur microwave activity on 902 MHz and up is of interest. If you're interested in preparing a paper, please contact Al Ward, WB5LUA, at 214-699-4369.

Registration: Preregistration cost is $35 and will be accepted through August 15, 1994. After August 15, all registrations will be $45. Walk-ins are welcome.

Hotel/Etc: The Inn at Estes, 1701 Big Thompson Avenue (Hwy 34), PO Box 140, Estes Park, CO 80517, tel 303-586-5363 (in Colorado), 800-458-1182 (outside Colorado). They are offering a special rate of $62/night. Rooms are limited and the closing date for reservations is September 1, 1994. Please be sure to mention the Microwave Update 1994 Conference to receive the special rate.

Estes Park is about 85 miles west of Denver International Airport (DIA). Consider using airline shuttle options from DIA to Loveland-Fort Collins airport. "Charles' Tour and Travel" offers transportation from DIA and Loveland-Fort Collins airport to Estes Park. Contact them at 303-586-5151 for details and reservations. Car rental may be more cost effective from DIA or Loveland-Fort Collins, particularly if you can car pool with someone else.

Events: Thursday evening, informal get-together and registration from 7-9 PM. Friday and Saturday morning, technical sessions; Friday evening, swapfest, microwave band measurements of noise figure, scalar transmission and reflection, spectrum and power. Everyone is encouraged to bring along your microwave equipment for testing. Saturday afternoon, free time to enjoy the Colorado mountains. Saturday evening, dinner at the *Barleen Family Country Music Dinner Theatre.* Following dinner, we will return to the Inn at Estes Part for a "Visit My Station" session (bring slides or VHS video tapes for all to see).

Although a spouse program has not been planned, the Estes Parks area offers many shops and interests for spouses. A list of spouses based on preregistration will be available to facilitate potential gatherings.

Pack Rats Conference

A date has not been set at this time, although it's usually the first Saturday in October.

Contact: John Sorter, KB3XG, 5290 Stump Road, Pipersville, PA 18947.

Eastern States VHF Conference

A date has not been set at this time.

Contact: Stan Hilinsky, KA1ZE, 17 Pilgrim Drive, Tolland, CT 06084, tel 203-649-3258 (W), 203- 872-6197 (H).

(*Have an upcoming technical event? Drop us a note with the all the details and we'll include it in* Upcoming Technical Conferences.)

Feedback

• It has come to my attention that there were several omissions in the 8085 schematic in my November 1993 *QEX* article, "An 8085-Based Computer System."

1. For U1, pin 6 should go to ground. Pins 7, 8, 9 and 10 need a 10-kΩ pull down resistor to ground.
2. For U3, pins 26 and 28 go to +5 V.

If you have had trouble getting it to run, these corrections sould solve the problem. Sorry!—*Sam Ulbing, N4UAU*

• In the article, "A CVSD Codec System for Your PC," in the April 1994 issue of *QEX*, the pin labeled OE of U11 in Fig 1 should be numbered pin 1 and connected to ground. Also, I neglected to mention that you can contact MX-COM at 1-800-638-5577.—*Jon Bloom, KE3Z*

Book Review

McGraw-Hill Circuit Encyclopedia and Troubleshooting Guide, Volume II, ISBN 0-07-037610-7. By John D. Lenk, 1994, McGraw-Hill, Inc, 1221 Avenue of the Americas, New York, NY 10020, tel 800-262-4729. Hardbound, 691 pp, $59.50.

Reviewed by Ed Hare, KA1CV
ARRL Laboratory Supervisor

Volume I of this book was reviewed in the December 1993 issue of *QEX*. In that review, I came to the conclusion that it wasn't yet time to retire my 1980 McGraw-Hill *Modern Electronics Circuits Reference Manual* (Markus). Although I find Volume II to be a bit more "hammy" than Volume I (although it *doesn't* have the usual code-practice oscillator—that is found in Volume I), it still doesn't contain the sheer volume of circuits found in my old standby. I think I will keep the old one around for a while longer.

As in the first volume, nearly all of the circuits are derived from manufacturer's data books and applications notes. It would be nice to see more references to the *ARRL Handbook*, *QST* and *QEX*.

The second volume of this book is not a sequel to Volume I. Either book is complete and useful by itself. There are several chapters in common—the cross-reference and IC pinout chapters are virtually identical, although neither chapter is comprehensive. There are several similar-looking chapters in the table of contents of each book, but, for the most part, the similarities stop there.

This volume, like Volume I, promises to be a troubleshooting guide. Unlike Volume I, this book meets that promise. The amplifiers, RF circuits and power supply sections contain a total of 57 pages of testing and troubleshooting text. The digital circuits section has only 4 pages of discussion about troubleshooting and testing, a bit shortchanged, and far from complete.

The circuits chapters all contain circuit descriptions that are more complete than those usually found in circuits books. In many cases, the applicable data sheets or circuit formulas are included. All circuits cite the source, usually a 1990s component-manufacturer catalog.

In addition to the usual op-amp circuits chapter, there are a few chapters that one doesn't usually see in this type of book. For example, there are separate chapters for analog and switching power supplies. The switching-supply chapter is interesting and novel, showing a number of uses of Semtech Corporation ICs. The analog power-supply section is primarily "cookbook" circuits. A generous number of pages are devoted to frequency-to-voltage and voltage-to-frequency circuits, using primarily "cookbook" ICs and components. The chapters on temperature indicators and transducers contain a variety of circuits, primarily using National Semiconductor and Linear Technology devices.

The RF circuits chapter is the most interesting, at least to this ham. There are 108 pages, mostly circuits from GEC Plessey and Motorola data books. Most of the circuits show Motorola bipolar-transistor amplifiers and, just as in the Motorola RF data book, the circuit descriptions include device data and the printed-circuit board patterns for the circuit or test fixture. There are also about a dozen MOSFET amplifier circuits grouped together.

Either Volume I or Volume II would make a useful addition to a personal library that doesn't contain the data books from which the circuits were derived. I rated Volume I as a "5"; I would give this one about a "6." I already have all of the data books used to compile these volumes, so my search for the "perfect" circuits book will have to continue.