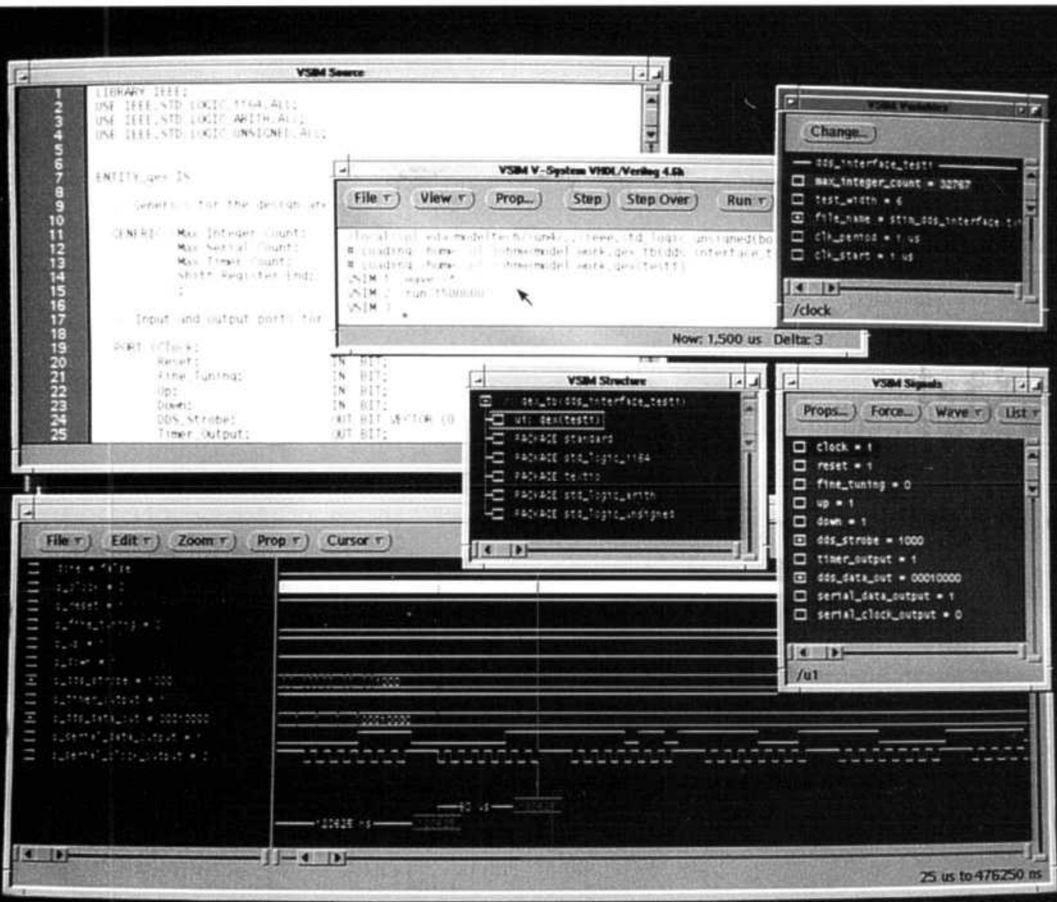
$1.75

QEX

ARRL Experimenter's Exchange

December 1997



```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;


ENTITY qex IS

   -- Generics for the design are

  GENERIC(  Max_Integer_Count:
            Max_Serial_Count:
            Max_Timer_Count:
            Shift_Register_Ind:
            );

   -- Input and output ports for

  PORT (Clock:
            Reset:                 IN  BIT;
            Fine_Tuning:           IN  BIT;
            Up:                    IN  BIT;
            Down:                  IN  BIT;
            DDS_Strobe:            OUT BIT_VECTOR (0
            Timer_Output:          OUT BIT;
```

```
/local/.../eda/modeltech/sunos/../ieee.std_logic_unsigned(bo
# Loading /home/.../model/work.qex_tb(dds_interface_t
# Loading /home/.../model/work.qex(test1)
VSIM 1> wave *
VSIM 2> run 1500000
VSIM 3>
```

Now: 1,500 us Delta: 3

— dds_interface_test1
□ max_integer_count = 32767
□ test_width = 6
⊡ file_name = stim_dds_interface.t
□ clk_period = 1 us
□ clk_start = 1 us

/clock

⊟ qex_tb(dds_interface_test1)
- □ u1: qex(test1)
- □ PACKAGE standard
- □ PACKAGE std_logic_1164
- □ PACKAGE textio
- □ PACKAGE std_logic_arith
- □ PACKAGE std_logic_unsigned

□ clock = 1
□ reset = 1
□ fine_tuning = 0
□ up = 1
□ down = 1
⊡ dds_strobe = 1000
□ timer_output = 1
⊡ dds_data_out = 00010000
□ serial_data_output = 1
□ serial_clock_output = 0

/u1

25 us to 476250 ns

Modern Digital Design Tools for Your Projects

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

QEX

QEX (ISSN: 0886-8093) is published monthly in July 97, August 97, September 97, October 97, November 97 and December 97 by the American Radio Relay League, 225 Main Street, Newington CT 06111-1494. Subscription rate for 6 issues to ARRL members is $18; nonmembers $30. Other rates are listed below. Periodicals postage paid at Hartford CT and at additional mailing offices.

POSTMASTER: Form 3579 requested.
Send address changes to: QEX, 225 Main St, Newington CT, 06111-1494

Issue No. 185

David Sumner, K1ZZ
Publisher

Rudy Severns, N6LF
Editor

Robert Schetgen, KU7G
Managing Editor

Lori Weinberg
Assistant Editor

Zack Lau, W1VT
Contributing Editor

Production Department

Mark J. Wilson, K1RO
Publications Manager

Michelle Bloom, WB1ENT
Production Supervisor

Sue Fagan
Graphic Design Supervisor

David Pingree, N1NAS
Technical Illustrator

Joe Shea
Production Assistant

Advertising Information Contact:

Brad Thomas, KC1EX, *Advertising Manager*
American Radio Relay League
860-594-0207 direct
860-594-0200 ARRL
860-594-0259 fax

Circulation Department

Debra Jahnke, *Manager*
Kathy Capodicasa, N1GZO, *Deputy Manager*
Cathy Stepina, *QEX Circulation*

Offices

225 Main St, Newington, CT 06111-1494 USA
Telephone: 860-594-0200
Telex: 650215-5052 MCI
Fax: 860-594-0259 (24 hour direct line)
Electronic Mail: MCIMAILID: 215-5052
Internet:qex@arrl.org

Subscription rate for 6 issues:

In the US: ARRL Member $18, nonmember $30;

US, Canada and Mexico by First Class Mail: ARRL Member $31, nonmember $43;

Elsewhere by Surface Mail (4-8 week delivery): ARRL Member $23, nonmember $35;

Elsewhere by Airmail: ARRL Member $51, nonmember $63.

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

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

QEX
$1.75
ARRL Experimenter's Exchange
December 1997
Modern Digital Design Tools for Your Projects

About the Cover

A simulation screen from Model Technology Inc's Synopsys VHDL simulator. Learn about the process and other tools in John Wiseman's lead article.

Features

Columns

December 1997 QEX Advertising Index

ByteMark: 30	PC Electronics: 30
Communications Specialists Inc: 21	Tucson Amateur Packet Radio Corp: 31
EM Scientific, Inc: Cov IV	Z Domain Technologies, Inc: 21
HAL Communications Corp: 31	

THE AMERICAN RADIO RELAY LEAGUE

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

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

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

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

Membership inquiries and general correspondence should be addressed to the administrative headquarters at 225 Main Street, Newington, CT 06111 USA.

Telephone: 860-594-0200
Telex: 650215-5052 MCI
MCIMAIL (electronic mail system) ID: 215-5052
FAX: 860-594-0259 (24-hour direct line)

Officers

President: RODNEY STAFFORD, KB6ZV
5155 Shadow Estates, San Jose, CA 95135

Executive Vice President: DAVID SUMNER, K1ZZ

Purpose of *QEX*:

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

2) document advanced technical work in the Amateur Radio field

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

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

Both theoretical and practical technical articles are welcomed. Manuscripts should be typed and doubled spaced. Please use the standard ARRL abbreviations found in recent editions of *The ARRL Handbook*. Photos should be glossy, black and white positive prints of good definition and contrast, and should be the same size or larger than the size that is to appear in *QEX*.

Empirically Speaking

We are all saddened by the passing of Doug DeMaw, W1FB. I had the good fortune to communicate with him a few weeks ago, and he gave me some sage advice for *QEX*. I have taken much of his advice, and *QEX* will be the better for it. We shall miss him.

Many of you may have read the May/June *QST* articles about class-E RF amplifiers. I've spoken with Nat Sokal, WA1HQC, the inventor of this amplifier class. He has promised (remember that Nat!) to provide a *QEX* article sometime soon. In the meantime, he sells a kit to experimenters. He may also provide documentation to those who don't want the kit (for a nominal charge). You can e-mail him at **73507.247@compuserve.com**.

DSP is a hot topic for ham experimenters, and Doug Smith, KF6DX, has written a three-part tutorial article that will appear in the coming months. Besides writing for *QEX*, Doug designed the new Kachina 505 DSP transceiver. It has many interesting features. (You can get more information at **www.kachina-az.com**.)

QEX Goes Bimonthly

We received many comments about changing from monthly to every two months or quarterly for *QEX* publication. Surprisingly, no one we heard from seems to mind changing from monthly, and many feel that quarterly would be okay. The majority, however, are for bimonthly, and we agree with you. Two months is not so long that you forget we are here, and the amount of material is not so great that you can't take the time to read through it. Beginning in January, you will receive your copy of *QEX* every *two* months. The total material you get each year will not decrease! Issues will be either 48 or 64 pages, depending on the available material. Actually, the space for articles increases a bit, because we'll be printing the ads and regular features like Empirically Speaking and the Table of Contents page once instead of twice.

You might wonder how this affects your subscription. The net effect is that current subscribers get a bonus. When you subscribed to *QEX*, you did so for 12 issues, rather than for 12 calendar months. So even though we are doubling up on the material and just sending it less frequently, we will send you the number of issues remaining on your subscription. For example, if you have six issues left after this one, you'll receive *QEX* through the November/December 1998 issue, not through the May/June issue. We'll continue to send renewal notices about four months in advance (when you have two issues left) to help ensure uninterrupted delivery. Any *QEX* renewals or new subscriptions received after December 1, 1997, will be honored for 6 issues, rather than 12.

In addition to the change to bimonthly, we're raising the basic subscription rate from $15 to $18 per year (6 issues) effective January 1. These two changes will help ensure *QEX*'s long-term financial health.

Hopefully, you will get the January/February issue near the beginning of January. If we are a little late on the first one, please accept our apology. We will have in effect moved the delivery of February up one month, so things will be hectic while we prepare two month's worth of material in the time allotted for one. Look at it this way: You'll get six issues on time and the other six a month early! You can't complain about that!

This Month in *QEX*

The past few issues have shown that we like a range of topics in each issue, and this issue is no exception. Antenna tuners are a perennial topic, and you might think there is nothing new to be said. Not so!

John Wiseman, KE3QG, takes a careful look at high-end digital design for hams and encourages your move to a higher level of integration, using field-programmable gate arrays. He gives a good overview, as well as some specific examples.

Bill Sabin, WØIYH, takes a careful look at the T-network as a tuner this month. His analysis shows many settings that may result in a match but not necessarily optimum performance. Bill's analysis shows that this simple and common network has many subtleties.

Just when you think a subject has been exhausted, along comes an author who points out what you missed. Bob Haviland, W4MB, takes us through amplifier tuning and explains off-resonance behavior. This condition is common when running up and down wide bands like 75/80 or 160 m during a contest, but it has received scant attention in the past. We too often think only in terms of a perfectly tuned amplifier. Bob's article brings out some points we may have missed.—*Rudy Severns, N6LF; e-mail* **n6lf@arrl.org**

Modern Digital Design for the Radio Amateur

What is VHDL? It's a software approach to synthesize logic hardware in field-programmable gate arrays (FPGAs)—and it's available to you!

By John Wiseman, KE3QG

Introduction

Within the last few years, three major technological advances have combined to revolutionize the design of digital logic circuits. First, Field Programmable Gate Arrays (FPGA) chips have become large and fast enough to perform significant circuit functions, while also becoming affordable for the typical Amateur Radio experimenter. Second, the usage of VHSIC Hardware Description Language (VHDL) as a circuit descriptor and simulator has made it possible to design large digital circuits quickly and easily. Third, the emergence of digital logic synthesis software tools that can read and operate on VHDL circuit descriptions has considerably simplified the job of translating the design description into a functional logic design. A cursory glance through the electronic engineering job advertisements in newspapers and trade magazines will verify the importance that companies place on these skills; many digital design jobs now require a working knowledge of FPGA design, VHDL and logic synthesis.

In this paper, I will describe an FPGA-based circuit that was designed and implemented with VHDL and digital logic synthesis to control a direct digital synthesizer (DDS) and a vacuum fluorescent front-panel display in a home-brew transceiver. I will describe the basics of FPGAs and how they fit into the digital logic design hierarchy, the details of this particular circuit implementation, some background material on VHDL design techniques, simulation results, logic compilation, programming and circuit prototyping.

Previous designs of circuits such as the one described here have used software programmed microprocessors as the controlling elements.[1,2] This design takes advantage of the inherent modularity of VHDL and synthesized FPGA devices, by allowing powerful high-speed digital logic functions that cannot be implemented in microcontrollers or even DSP chips to be added later, at the designer's leisure, by merely adding I/O wires to the exist-

20 Amaryllis Ln
Newtown, PA 18940
e-mail **johnw@dmsl.hitachi.com**

[1] Notes appear on page 12

ing FPGA chip. As an example of this technique, I have already written and simulated VHDL that describes digital logic that is used to randomly dither the output of a DDS to help with spur reduction.[3] This logic can easily fit within the spare gates and I/O pins of the FPGA that is currently used in the radio for tuning and display. In this way, both low and high-performance digital logic may be incorporated into one single IC with a single design methodology.

Digital Hardware Technology

In the hierarchy of digital circuit design, the most complex level is full custom design. In full custom, the designer has complete control over all the design variables such as speed, layout, voltage, power consumption, etc, by designing at the transistor level and may trade off one or more of these variables for more performance in one of the others. This high level of performance comes at a very high price, however. These circuits are usually the most expensive in terms of engineering-development dollars. Of course they also require very sophisticated design tools as well as highly trained circuit specialists to design them. Last but not least, a state-of-the-art fabrication facility must build the IC and test it. Leading edge Amateur Radio experimenters regularly use complex full custom ICs such as DSP chips that they have purchased from semiconductor companies, but very few have the resources to design their own full custom chip!

Application Specific Integrated Circuits (ASICs) were invented to give digital designers the ability to implement circuits that approach the performance levels of full custom chips, but with lower development costs and less overall design and fabrication time. A particular type of ASIC is designed to be an "array of gates," in which digital logic gates are arranged in a fixed pattern on a substrate, and the fabrication process consists of connecting these gates to perform a particular function by defining custom metalization layers for the chip. Because the circuit designer is not working at the transistor level as in full custom design, the job is considerably easier, but the ultimate performance is somewhat lower as the designer's freedom of choice is more limited. Gate arrays with gate counts of more than 1 million and clock speeds greater than 100 MHz are available from many vendors. Although cheaper to design and develop than full custom, this technology is still way out of the reach of amateur level experimenters. Typical one-time engineering costs are around $100k for complex designs, and expensive design tool packages are required for chip design and simulation.

Field Programmable Gate Array (FPGA) technology is basically a third level in the digital-circuit design hierarchy that holds considerable promise for the advanced amateur experimenter. If carefully designed, an FPGA can perform sophisticated logic functions at clock rates greater than 50 MHz, with densities up to 100,000 gates. An FPGA is an off-the-shelf part that is available from several semiconductor suppliers such as Actel, Altera, AT&T, Cypress, Xilinx and several others. It is designed to be like an ASIC, in that an array of gates or logic elements is placed unconnected on a substrate, and the designer, coupled with the design software, defines the connections. The connections for the FPGA are performed by the user, "in the field," therefore the term Field Programmable Gate Array. This is in contrast to an ASIC, where the connections are done by defining metalization layers at the semiconductor factory. FPGA logic element connections are done in two ways: The first defines a connection that is permanently made. This connection, once made, cannot be changed; it results in a one-time programmable part. This is usually fine for stable designs and generally results in a faster circuit at somewhat lower prices. The second connection type is probably of most interest to amateur experimenters: the many-time programmable style. Here, the connection is accomplished either with EEPROM or SRAM technology. EEPROM based FPGA devices require no external device support, but they must be programmed either by a PROM programmer (with the proper adapter, which can be large and expensive) or in some cases, by a serial link to a PC programming card. The design that I am going to demonstrate uses an Altera 81500 FPGA with SRAM connection technology that is programmed at reset via an inexpensive 8-bit parallel EPROM. These memory chips are very easy to program with standard PROM programmers without the need for expensive footprint adapters or additional PC board programming cards.

A level below FPGAs are the various Programmable Array Logic (PAL) and Programmable Logic Device (PLD) chips.[4] These chips are generally used as "house cleaning" devices on circuit boards by combining relatively simple logic functions together in one chip, instead of several inefficiently utilized SSI or MSI IC packages.

The lowest level of digital design (apart from discrete transistors of course!) is done with Small-Scale Integration (SSI) or Medium-Scale Integration (MSI) ICs, such as 7400 TTL or 4000 CMOS. These chips are typically available in 14 to 28 pin DIPs. Experimenters and hobbyists have been using wire-wrap, point-to-point or PCB interconnection technologies to construct working boards with these chips. Back in the "bad old days," I worked on systems where I designed and debugged prototype wire-wrap boards that consisted of approximately 200 ICs. These projects clearly showed the problems inherent to this style of design, such as technology-specific specifications, package-to-package timing delays, interconnection-wire cross-talk, clock skew, power consumption, etc. Also, one of the most difficult things about designs such as these is that changes can be very difficult to implement. The design example illustrated in this paper would be fairly impractical to implement in discrete ICs for usage in a small radio, but circuit construction is easily done with a modern FPGA device. The overall job is made even easier by using VHDL and logic synthesis as the design description tools for the FPGA.

Functional Design Goals

The digital circuit design described in this paper needs to perform two distinct functions: to communicate frequency information to the DDS chip in a parallel format and to interface with the front panel display unit in a serial format. There are five input signals to the FPGA chip that control the state of the output lines. There are six output signals that are subdivided into three signals that interface with the DDS and three signals that interface with the display unit. These signals are detailed in Table 1 and are shown in block diagram form in Fig 1.

Several of these signals require further explanation. The direct interface to the DDS chip is through a 24-bit parallel bus. In my previous design, I had added a PC parallel-port interface board to the DDS input, resulting in an 8-bit parallel interface to any external device. I decided that it would be more trouble than it is worth to undo this portion of the design, so I left the 8-bit to 24-bit receive/transmit latches in place and did not incorporate them directly into the FPGA design. I can still use my PC parallel-port interface if I choose. Because of the multiplexed 8-bit data interface, the design requires four control signals, DDS Strobe 0-3.

The serial interface to the front panel display unit is fairly standard, with data-input, clock-input and chip-select lines. The clock signal, Serial Clock Output, is gated so that it is activated only when needed (to reduce potential noise from being coupled back into the receiver's analog circuitry). For more detailed explanations of these signals and the required timing, request the Futaba NA16SD03AB data sheet from the factory.

The signal Timer Output is brought out for testing and verification purposes only. I initially used it as a marker to define the end of a frequency update cycle in my simulations, but then thought it would be a good idea to have it available for potential display on an oscilloscope or logic analyzer. With so many available I/O pins on the FPGA, this is generally a good practice. In this case, this signal will toggle to a logic high for two clock pulses after the update cycle is completed. I have programmed this to occur after approximately one quarter of a second in this application.

The user-interface controls were designed for simplicity and to somewhat mimic the function of the PC control design. In the PC controlled DDS interface, the keyboard up-arrow was the coarse frequency increment, while the down-arrow was the coarse frequency decrement. Likewise, the right-arrow was the fine frequency increment and the left-arrow was the fine frequency decrement. The *C* code permitted the user to enter any desired values for the fine and coarse frequency steps, but I found that I used values of 10 Hz and 1000 Hz, respectively, under most conditions. Because of this, I decided to "hardwire" these values directly into the VHDL circuit description as default values. This reasoning led me to single push buttons for **UP** and **DOWN**, with a toggle switch for **FINE TUNING**.

With a coarse tuning resolution of 1000 Hz and a push-button interface, it is not good to tune too quickly. As a matter of fact, some time delay between frequency updates is good, even when the buttons are continually pressed. My own personal preference (based on the changing audio pitch of the scanned signals, as well as the visual update of the front panel display itself) is about one quarter of a second, which is "hardwired" into the VHDL code as a constant value. Of course the trade-off for this delay time is that it takes approximately 13 seconds to scan 50 kHz (**FINE TUNING** off), but that is fine for me.

All of this serves to illustrate part of the beauty of designing hardware with a language such as VHDL, then using synthesis tools to generate the FPGA logic. Preprogrammed constants such as the fine and coarse frequency steps and the tuning time delay may be modified later by simply changing a single value in the VHDL code, recompiling, then reprogramming the FPGA or its EEPROM. What previously would have taken many hours of redesign, most likely resulting in the addition of "dead bug" chips to the board, along with the attending etch cuts and wire substitutes, now only takes a matter of minutes. In this respect, designing hardware is very similar to designing a quality software application for an embedded microprocessor or even a PC application.

FPGA Circuit Description Using VHDL

By deciding to use FPGA technology

Table 1

Signal Name	*# of Bits*	*Input/Output*	*Source/Destination*	*Function*
Clock	1	Input	1.0 MHz. Crystal Oscillator	Master Clock
Reset	1	Input	Reset Circuit	Power-Up Reset to Default Values
Fine Tuning	1	Input	Front Panel Toggle Switch	"On" - 10 Hz. Tuning "Off" - 1 kHz. Tuning
Up	1	Input	Front Panel Push Button	Increase Operating Frequency
Down	1	Input	Front Panel Push Button	Decrease Operating Frequency
DDS Strobe	4	Output	DDS Interface	Data Strobes for DDS Interface
Timer Output	1	Output	DDS Interface	End-of-Cycle for Testing Purposes
DDS Data Output	8	Output	DDS Interface	DDS Frequency Data
Serial Data Output	1	Output	Front Panel Display Unit	Serial Setup and Character Data
Serial Clock Output	1	Output	Front Panel Display Unit	Serial Clock
CS Output	1	Output	Front Panel Display Unit	Chip Select for Display Unit

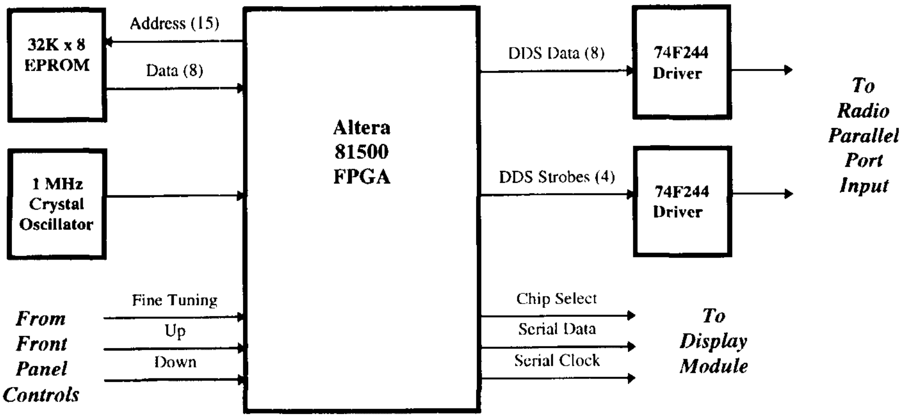


Fig 1—Block diagram of a FPGA based DDS and display control circuit.

for the hardware implementation of this design, we will satisfy several overall design goals such as: relatively low power consumption, small circuit-board area and ease of reprogramming for debugging and enhancements. How then do we define the logic design for the FPGA? As was shown in the hierarchy of The Digital Hardware Technology section, FPGA design at the board level allows us to design the hardware in as high a level of abstraction as we can reasonably afford. The goal for the actual FPGA circuit description design is to also remain as "high level" as possible, removing us from as many of the design variables as possible such that the design is quick and easy. This is where the power of VHDL coupled with logic synthesis is clearly demonstrated.

What is VHDL?

In the early 1980s, the Department of Defense sponsored a program called Very High Speed Integrated Circuit, or VHSIC for short. Several different companies were involved, along with different circuit technologies, so DoD sought a way to consistently describe these various designs independent of the specific technology. From this effort, the VHSIC Hardware Description Language (VHDL) was born. One of the reasons that VHDL has proven popular with designers and Computer Aided Engineering (CAE) software developers alike is that it has been standardized as IEEE Standard 1076 and United States Department of Defense MIL-STD-454L. It is worth mentioning that another hardware-description language called Verilog has also become popular in the last few years. The next few years will decide which language becomes dominant or whether both will peacefully coexist with both logic designers and CAE software vendors.

VHDL is a software language that is used to model and describe the workings of hardware. There are three styles of VHDL coding in active use. These styles are *behavioral, RTL* (sometimes referred to as *dataflow*) and *structural* VHDL, which are hierarchical in nature. The highest level in the VHDL hierarchy, the most abstract in nature, is the behavioral style. Generally, behavioral VHDL is used to describe an algorithm or a system without regard for the clock cycles that actual hardware might use to perform the work. This style is useful for initial or baseline system simulations or even as a system specification. Register Transfer Language (RTL) VHDL is sort of a middle ground description, where functionality is based on a clock cycle. In RTL, the designer thinks more in terms of how digital hardware is usually implemented. In this way, a collection of clocked registers is updated via various transfer functions. This is the level where most current-generation digital-logic-synthesis tools operate, and it is the style I have used to write the VHDL program in this article. The lowest level of abstraction is structural VHDL, where previously defined components are essentially "hardwired" into the circuit. This level of VHDL is somewhat analogous to the drawing of schematics, as all individual connections must be fully specified. This is gate-level design at this point, and function is based on clock cycles as well as real time.

In practice, the three main styles of VHDL are often blended. It is best to work at the highest possible level of abstraction for efficiency. To get the synthesis tools to perform adequately, however, it usually means that the designer must describe hardware at the RTL level. If the resulting synthesized circuit still does not meet critical design criteria such as speed, area or power consumption, the designer may mix in structural VHDL and literally place predefined optimized functions into the logic design. Those familiar with software design will notice that this is somewhat similar to how programs are developed for real-time applications such as DSP. In this case, a high-level language such as *C* is used with a compiler that generates executable code. Only if required by speed or memory limits does the programmer go down a level of abstraction to use assembly language.

Some Basic VHDL Concepts

VHDL is very good in design applications such as the one described in this paper, but I will be the first to admit that it does come at a price. VHDL is a *verbose* language that can be somewhat intimidating to learn. It is certainly not my intention to give a VHDL tutorial in this short presentation (several textbooks I've read haven't done it justice!). Nonetheless, it is important to demonstrate a few quick examples that show the look, feel and power of the language when coupled to modern logic-synthesis tools.

A typical software language such as *C* executes all of its instructions sequentially. In VHDL, instructions are executed *concurrently*. To execute instructions such as conditional tests sequentially, the instructions are placed into design units called *processes*. Processes are executed in a concurrent fashion, therefore ordering within the top-level of the program, the *architecture*, is irrelevant. Because of their importance in the RTL design style, I will give several process examples in the next section.

At the higher level, VHDL code will consist of an *entity* and an *architecture*. The entity in this design is used to declare and define the generic constants that are used throughout, as well as to define the I/O ports for the device. These I/O ports must have an associated name, type and mode. The architecture is really the main body of code for the design, where the actual simulation behavior of the entity is defined. Local signals must be declared here, and the statement region can only consist of concurrent statements. These concurrent statements can also be separate processes containing sequential statements.

At the lower level, VHDL objects can be described as *signals*, *variables* and *constants*. Signals are used to communicate between other concurrent statements or among other processes. They are analogous to wires in a hardware sense. The syntax for updating signals is **new_signal <= old_signal**, where the signal **new_signal** takes on or receives the value of the signal **old_signal**. All signals in the design must be declared, along with an indication of what *type* they are. Being what is known as a *strongly typed* language, VHDL will check to make sure that the designer correctly matches signals within a design. An example of this is that the compiler will report an error if **new_signal** is declared as an 8-bit vector, but **old_signal** is a 12-bit vector. (VHDL names are case insensitive, but they must be contiguous with no spaces. That is why my descriptive signal names contain underscores when more than one word is involved.) Variables are used internally to processes as instantaneous data storage. As in a traditional programming language, variables are updated immediately, unlike signals that are scheduled and updated at the end of the simulation cycle. Constants are used to hold values that do not change; they are generally used to make the VHDL code easier to read and modify.

VHDL Process Examples

The concept of the VHDL process with internal signal assignments is best demonstrated with some examples. In this section I have chosen four separate processes that show some important functions. These are the actual processes that are used in the DDS/Display Controller VHDL code.

Combinatorial and Sequential Logic

The first example process that I am giving (Program Segment A) describes a pulse-generation function. The desired pulse is to be two clock cycles wide and used to trigger the initial default conditions for the display and the DDS chip at reset. The way that this works is that **Reset_Int** (latched Reset signal) is delayed with two *inferred* flip-flops within the **IF Clock...** statement. Note that the term *inferred* refers to the fact that the synthesis tool will interpret what I have written in software to mean a clocked delay, implemented in hardware with a flip-flop. **Reset_Int** is delayed by the first flip-flop, with the output labeled as the *signal* **Reset_Del**. This *signal* is then delayed by the second inferred flip-flop to create **Reset_Del_1**. Notice that these two consecutive statements will be updated *in parallel* during the update phase. A shift register of N bits could be described with this technique by writing N consecutive assignment statements within the **IF Clock...** statement.

This process also contains a combinatorial statement. This statement performs an exclusive-OR function on the original signal **Reset_Int** and the two clock pulse delayed version just created. This will result in a single pulse lasting for two clock cycles that occurs immediately after **Reset** goes high. Note that this XOR statement is *not* contained within the **IF Clock...** statement, as it is not desired to be clocked logic.

It's interesting for beginners to confirm that the synthesis tool indeed puts out the intended hardware from the VHDL description. I ran this process through the Synopsys FPGA Compiler and plotted out a schematic of the resulting gate-level logic. Fig 2 shows that the resulting logic is indeed what was intended when the process was written. Note that the exclusive-OR function was implemented with AND/OR gates, along with inverters, as the Altera library that I was using did not have exclusive-OR primitive functions.

Sequential Logic With Feedback

In Program Segment B, it is desired to produce a clock for the serial section at half of the input clock frequency. A simple method for accomplishing this in discrete logic design uses a D flip-flop by connecting the inverse-Q output back to the D input. That is essentially what is modeled here, with the addition of a clock-enable function (**Serial_Clock_Enable**) and a reset function (**Reset_Int**).

This process is to be synchronous, so a D flip-flop is inferred with the **IF Clock...** statement. Within this statement, another **IF** statement is used to test whether **Serial_Clock_Enable** is low (= 0), OR **Reset_Int** is low. If *either* of these cases is true, then the output is to be a constant high logic level, with no clocking. This is done with the **Serial_Clock_Preout** <= 1 statement. If *both* of these conditions are not satisfied (That is, both are false.—*Ed.*), then the divide-by-two clock is output. This function is contained within the **ELSE** portion of the statement. The statement **Serial_Clock_Preout** <= NOT **Serial_Clock_Preout** assigns an inverted version of this signal to itself on every rising edge of the clock, thus emulating the divide-by-two D flip-flop described above.

Again, as in the previous process example, I have given the gate-level schematic output of the Synopsys FPGA Compiler for this process to show that the gate-level representation is what is desired (see Fig 3). In this case, it is clear that an inverted version of the output is fed back to the D input of a flip-flop and that this input is enabled by both the **Reset_Int** and the **Serial_Clock_Enable** lines.

State-Machine Description With the CASE Statement

Having read through the previous two-process descriptions and having seen the gate-level output schematics, you might be wondering to yourself, "What's all this fuss?" It would appear that I have gone to great lengths to describe in many words what a schematic can describe with relatively few lines and symbols. This may be true for the simplest of circuits, but VHDL becomes far more powerful when the advantages of software-based higher-level languages are coupled with synthesis tools for design of state machines, counters, etc. In these instances, relatively few powerful commands can synthesize down to hundreds if not thousands of gate equivalents, thus saving the designer considerable time and effort.

A powerful method of describing state machines with VHDL is with the CASE statement. As an example Program Segnent C describes a state machine within the DDS and display interface design that generates three outputs. These outputs are Output Strobe 0-3, which go directly to the DDS interface circuitry, Data Mux Enable and Cycle Over, which are both internal signals used for control purposes.

First, take a look at the structure of the process Output State Combinational. As its name implies, this process is going to control the output of the state machine, and it is not sequential or clocked in nature. As such, there is no **IF Clock...** type statement in the

Program segment A

```
Initialization:
PROCESS(Clock, Reset_Int, Reset_Del_1)

- Note: Comments in VHDL are denoted
- with the double dash symbol (--).

- This process produces a single pulse, Init_Signal,
- which is used to trigger the initial
- system outputs following a reset.

BEGIN

  IF Clock = '1' AND Clock'EVENT THEN
    Reset_Del <= Reset_Int;
    Reset_Del_1 <= Reset_Del;
  END IF;
  Init_Signal <= Reset_Int XOR Reset_Del_1;

END PROCESS Initialization;
```

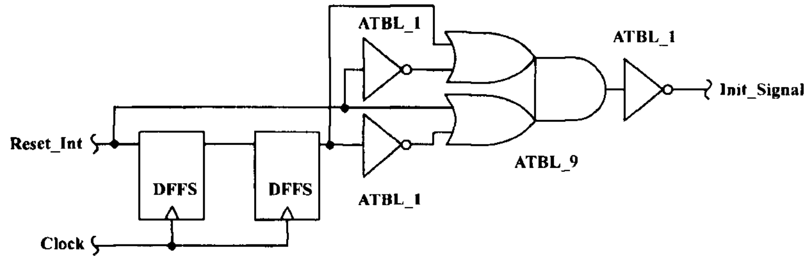


Fig 2—Synthesized output of process "Initialization."

process. For the design of the state machine, start with the number of different states that are desired. In this case, I want 18 discrete output states, so I block off 18 distinct **WHEN** sections as shown below. Each of these blocks is started off with the statement **WHEN State(N) =>** , where N ranges from 0 to 17. Each of these statements within a block is followed by the signal assignments that make up the desired outputs of the state machine at that particular time. A closing statement, **Next_State <= State(N+1)**, is used to tell the state machine what the desired next state is to be. When the 18th and final state is being executed (N=17), the Next State transition will be to go back to State0. What are the signals labeled as State0 - State17 that represent these output states? They are all enumeration literals, each of them elements of an enumeration list. To use these, we must declare a new user-defined data type, and I call that States_Enum_ Type. This declaration must be done before any signal declaration and process listing within the VHDL code. As such, I have given not only the processes of interest here, but the ARCHITECTURE template as well, showing the structure and placement of the TYPE declaration within it.

Now because the enumeration literals State0 - State17 are used to update the signals **Current_State** and **Next_State**, we must declare these signals to be of TYPE States_Enum_ Type. This is also elaborated in the example below for clarity.

Finally, to tie all of this together, the sequential process Output_State_ Sequential is also given as part of this example. The purpose of this process is to synchronize the output from the combinational process Output_State_ Combinational in a fashion that the synthesis tool will recognize as an efficiently organized hardware description.

In the previous process examples, I gave schematic outputs that allowed a beginner to verify that the VHDL description did indeed synthesize into the correct gate-level representation by checking the schematic output, which is probably the more familiar method of circuit description for older-style digital logic designers. This rapidly becomes impractical as the size of the synthesized circuit surpasses more than a handful of gates. It is unnecessary anyway once you become comfortable with the VHDL description. The point to be made here is that the VHDL code *is* the circuit representation. When you become comfortable with it, you will find it easier to think in VHDL terms. The "crutch" of schematic representations will go away. As such, no output schematics for the FPGA logic are shown from this point on in this paper.

VHDL with Arithmetic Operations

This process demonstrates the usage of higher level functionality such as adders and subtractors. Again, the usage of code within the **IF Clock...** statement infers a synchronous nature to the circuit and the use of **IF - ELSIF** statements allows conditional functionality depending on the status of various control signals.

The first condition checked is for the status of the Reset line. If it is at a logic low level, then the default values for the receive and transmit frequency will be loaded into **DDS_Data_ Counter_Rx** and **DDS_Data_ Counter_Tx** respectively. If Reset is not active, the status of the control inputs **UP**, **DOWN** and **FINE TUNING** determine what operation to perform. In the case of **Up_Int** = 1 **AND Fine_Tuning_Int** = 1 (along with **Timer_Preind** = 0, a timing reference), then the value of the *constant* **Fine_Tuning_Offset** will be added to the previous value of **DDS_Data_ Counter_Rx** and **_Tx**. In this design, **Fine_Tuning_Offset** is fixed at the unsigned 24-bit representation of 10, for 10 Hz increments.

Likewise, if **Down_Int** = 1 **AND Fine_Tuning_Int** = 1, then the value of **Fine_Tuning_Offset** (10) is subtracted from the previous value of the data counters. The remaining two conditional tests are for when the **Fine_Tuning_Int** signal is in the 0 state, indicating that fine tuning is off, coarse tuning is desired. If **Up_Int** is = 1, then the value of **Coarse_Tuning _Offset** is added to the data counters. If **Down_Int** is = 1, then it is subtracted from the data counters. Again, **Coarse_Tuning_Offset** is a *constant*, and it is set in this design at the unsigned 24-bit representation of 1000, for 1000 Hz increments. If all of the conditional tests fail, then the previous states of the data counters are used, and no change is made to the DDS frequency.

Now all of this was a real mouthful, wasn't it? The actual VHDL process as shown below is much more concise and to the point than the English language

Program Segment B

```
Serial_Clock_Divider:
PROCESS(Clock)

-- This process produces a divide-by-2 clock for the serial section.

BEGIN

    IF Clock = '1' AND Clock'EVENT THEN
        IF Serial_Clock_Enable = '0' OR Reset_Int = '0' THEN
            Serial_Clock_Preout <= '1';
        ELSE
            Serial_Clock_Preout <= NOT Serial_Clock_Preout;
        END IF;
    END IF;

END PROCESS Serial_Clock_Divider;
```

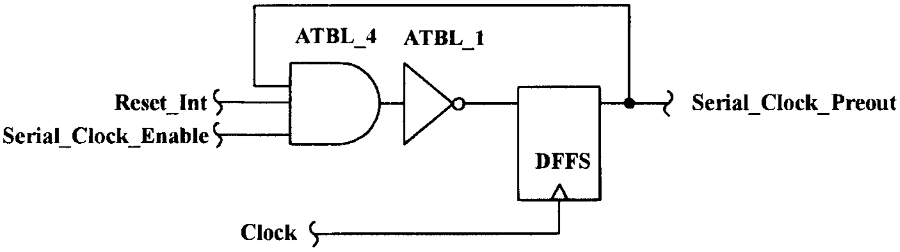


Fig 3—Synthesized output of process "Serial_Clock_Divider."

description, once you are familiar with how to read and understand it. This is one reason why VHDL is often used as a system specification as was mentioned previously.

Processing the Complete VHDL Code

The complete VHDL code describing this design is too long to be included here in its entirety, but it can be downloaded.[5] I would encourage taking a look at the code, as most of the processes used are variations of the four given above and are not hard to understand. I have tried to place comments in the code at critical places where it may be difficult to follow the logic. I have also included a second file that is a VHDL *testbench*. This code is not required directly for compilation of the FPGA design, but it is necessary to perform a simulation. If you download the code for inspection, note that at the end of the testbench code there is a commented section labeled "Test Vectors." This section actually exists as a separate file that is read by the testbench code, then used as input stimulus for the design simulation. In this manner, the designer can quickly and easily test all the desired functions of the circuit.

Now that we have a complete VHDL description of the circuit that we wish to build, we can continue to the simulation stage. After simulating the circuit for the required functionality, we will process the design using the Synopsys logic synthesizer and the Altera FPGA compiler.

Program Segment C

```
ARCHITECTURE test1 OF test_logic_3 IS

-- Types are defined here.

     TYPE States_Enum_Type IS ( State0, State1,
                                State2, State3, State4,
                                State5, State6, State7,
                                State8, State9, State10,
                                State11, State12, State13,
                                State14, State15, State16,
                                State17
                                );

   SIGNAL Current_State:    States_Enum_Type;
   SIGNAL Next_State:       States_Enum_Type;

BEGIN

  Output_State_Sequential:
  PROCESS(Clock)

-- This process synchronizes the output from the
-- process defined by Output_State_Combinational.

  BEGIN

    IF Clock = ‘1’ AND Clock’EVENT THEN
      IF Reset_Int = ‘0’ THEN
        Current_State <= State17;
      ELSIF Reset_Int = ‘1’ AND Cycle_Enable = ‘1’ THEN
        Current_State <= Next_State;
      ELSE
        Current_State <= State17;
      END IF;
    END IF;

  END PROCESS Output_State_Sequential;

  Output_State_Combinational:
  PROCESS(Current_State)

    -- This process defines the state machine that
    -- controls the generation of Output_Strobe,
    -- Data_Mux_En, And Cycle_Over.  Next_State
    -- is latched to Current_State in
    -- PROCESS Output_State_Sequential.

  BEGIN

    CASE Current_State IS

      WHEN State0 =>
        Output_Strobe <= “0000”;
        Cycle_Over <= ‘0’;
        Data_Mux_En <= “000”;
        Next_State <= State1;

      WHEN State1 =>
        Output_Strobe <= “0001”
        Cycle_Over <= ‘0’;
        Data_Mux_En <= “000”;
        Next_State <= State2;

      WHEN State2 =>
        Output_Strobe <= “0000”
        Cycle_Over <= ‘0’;
        Data_Mux_En <= “000”;
        Next_State <= State3;

      WHEN State3 =>
        Output_Strobe <= “0010”
        Cycle_Over <= ‘0’;
        Data_Mux_En <= “001”;
        Next_State <= State4;

          *
          *
          *

      WHEN State17 =>
        Output_Strobe <= “1000”;
        Cycle_Over <= ‘0’;
        Data_Mux_En <= “101”;
        Next_State <= State0;

      WHEN OTHERS =>
        Output_Strobe <= “0000”;
        Cycle_Over <= ‘0’;
        Data_Mux_En <= “000”;
        Next_State <= State0;

    END CASE;

  END PROCESS
Output_State_Combinational;
```

VHDL Functional Simulation

The VHDL simulator that I prefer to use is produced by Model Technology Incorporated. It has a very sophisticated user interface that includes separate windows for most of the desired testing and debugging functions, such as displaying updated variables and signals, a source code display list with single-step and breakpoint capabilities and a waveform input/output trace graphics display. I have included a photograph (Fig 4) showing what a typical testing/debugging session looks like within the Model Technology VHDL simulator. Note that the display computer is a PC running Windows NT, but the simulator is actually running on a Sun workstation under UNIX, via X-Windows software.

The simulator reads the compiled VHDL circuit description as well as the compiled VHDL testbench code and produces an output that may be checked at whatever level the designer deems to be appropriate. Very complex and sophisticated designs might produce large output files that need to be computer checked and compared to "standard" output files that are generated by other means such as *C* code or even behavioral VHDL. In the case of the circuit designed here, a visual check of the output within the waveform window is sufficient for verification purposes. The functionality items that I have chosen to simulate fully before programming the FPGA are:

- Correct default information is output after a power-up reset
- **UP** and **DOWN** controls work properly, both with and without the **FINE TUNING** control,
- Time delay between updates is implemented correctly, and
- No timing violations at the DDS interface or the display-interface module.

In general, simulation will be a two-step process. The first step is as described above where the technology independent VHDL code is simulated for correct performance on a clock- cycle basis, with no inherent timing dependencies. The second step is a back annotation of timing information into the simulator. This allows the designer to view timing information from the actual routed gate-level design and verify that the real design works as planned.

Logic Synthesis

After verifying that the VHDL code does what is expected through simulation, the code can be read by the logic synthesis tool. The dominant force in the digital-logic simulation field today is Synopsys, and that tool was used for the synthesis of the FPGA in this paper. Synopsys markets several tools for the ASIC/FPGA design market, and the one used here is aptly named FPGA Compiler. FPGA Compiler performs several tasks, such as the mapping to flip-flops in I/O modules as well as mapping to complex flip-flops (if available in the library), finite state-machine optimization, technology specific proprietary lookup table mapping transformations and the generation of timing constraints that are used by the place and route tools.[6]

Since the FPGA Compiler transforms the VHDL into technology-specific implementations, it is necessary for it to access vendor supplied FPGA design libraries. This particular design uses an Altera 81500 FPGA, so it is necessary to use the FLEX 8000 family design libraries from Altera. After compilation to the target library, a primitive netlist is generated by FPGA Compiler and stored in Electronic Data Interchange Format (EDIF). This standard text-based file format is commonly used to transfer design files among different vendor's software packages. In this case, it will provide a common format for transferring data from Synopsys' FPGA Compiler to Altera's MAX+ PLUS II design package.

FPGA Postprocessing and Layout

There are several more processing steps that must be done before a programming file may be generated for the EPROM. These final steps are performed within the Altera environment, in a software package called MAX+ PLUS II.[7] First, it reads in the EDIF file that was generated by the Synopsys FPGA Compiler. This data is then further processed to generate place and routing information for FLEX 8000 FPGA family devices. Along the way, design rule checking is done to make sure that good design practices are followed. This step can provide useful information to the designer: It flagged me that I did not initially have synchronizing flip-flops on the asynchronous **UP** and **DOWN** input signal lines. After the logic is fully partitioned into real Altera logic elements, a fitting operation determines what particular part from the FLEX 8000 family will accommodate the design. If you know in advance that a particular part has sufficient gate and pin counts, this stage can be skipped.

When the selected part is placed and routed, final timing information for the FPGA design is generated. This information is then back annotated to the VHDL simulator, through a VHDL netlist output from the Altera software, and the simulation is checked to make sure that the design still works.

Program Segment D

```
DDS_Counter:
PROCESS(Clock)

-- This process defines the count, up or down, procedure for the DDS
-- interface portion of the output.

BEGIN

   IF Clock = '1' AND Clock'EVENT THEN
      IF Reset_Int = '0' THEN
         DDS_Data_Counter_Rx <= Rx_Reset_Freq;
         DDS_Data_Counter_Tx <= Tx_Reset_Freq;
      ELSIF (Up_Int = '1' AND Timer_Preind = '0') AND Fine_Tuning_Int = '1' THEN
         DDS_Data_Counter_Rx <= DDS_Data_Counter_Rx + Fine_Tuning_Offset;
         DDS_Data_Counter_Tx <= DDS_Data_Counter_Tx + Fine_Tuning_Offset;
      ELSIF (Down_Int = '1' AND Timer_Preind = '0') AND Fine_Tuning_Int = '1' THEN
         DDS_Data_Counter_Rx <= DDS_Data_Counter_Rx - Fine_Tuning_Offset;
         DDS_Data_Counter_Tx <= DDS_Data_Counter_Tx - Fine_Tuning_Offset;
      ELSIF (Up_Int = '1' AND Timer_Preind = '0') AND Fine_Tuning_Int = '0' THEN
         DDS_Data_Counter_Rx <= DDS_Data_Counter_Rx + Coarse_Tuning_Offset;
         DDS_Data_Counter_Tx <= DDS_Data_Counter_Tx + Coarse_Tuning_Offset;
      ELSIF (Down_Int = '1' AND Timer_Preind = '0') AND Fine_Tuning_Int = '0' THEN
         DDS_Data_Counter_Rx <= DDS_Data_Counter_Rx - Coarse_Tuning_Offset;
         DDS_Data_Counter_Tx <= DDS_Data_Counter_Tx - Coarse_Tuning_Offset;
      ELSE
         DDS_Data_Counter_Rx <= DDS_Data_Counter_Rx;
         DDS_Data_Counter_Tx <= DDS_Data_Counter_Tx;
      END IF;
   END IF;

END PROCESS DDS_Counter;
```

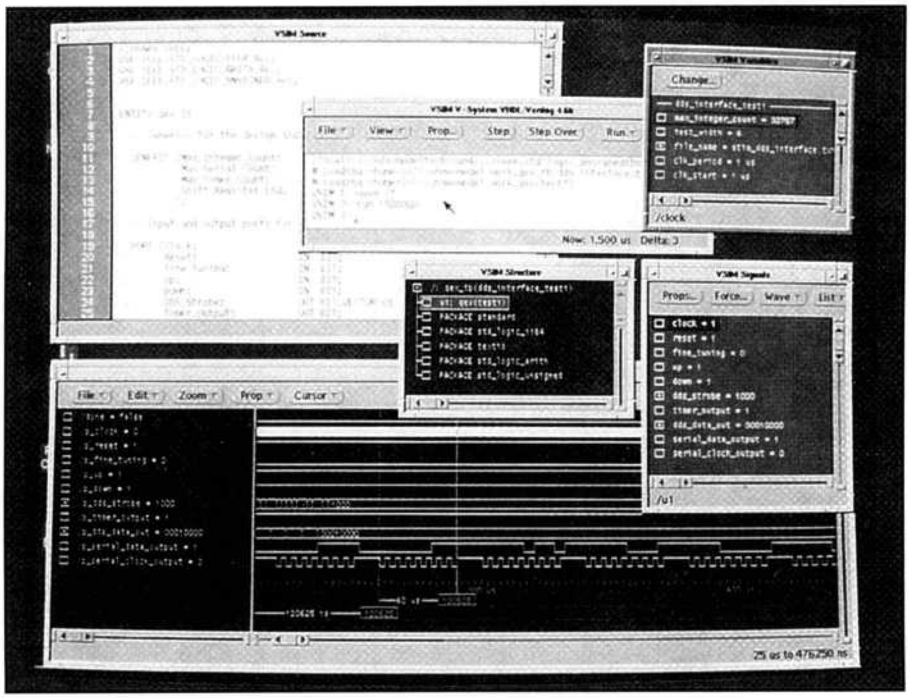


Fig 4—A VHDL simulation using the Model Technology simulator. The windows open in the photo are used to display (clockwise, from upper left-and corner) the VHDL source code, control functions, variables, signals, a waveform-trace window for I/O signals and (center window) overall design structure.

If not, various things can be done, such as rewriting the VHDL source code for more efficiency, optimizing some of the various parameters within the Synopsys environment or even specifying a faster FPGA device. For this particular chip, timing is very loose as the main clock is only 1 MHz and no further processing was needed.

The last stage within the Altera environment is a final assembly. Among other things, an output file in Intel hex format is generated that can be used to program the FPGA EPROM. At this point, the design is now ready to be tested in the system.

FPGA Programming

The Altera FLEX 8000 family of FPGA devices can be programmed in several ways. They can be programmed through an external device such as a PC or a microprocessor, or they can program themselves on power-up, either with a small footprint serial PROM or a standard 8-bit wide parallel EPROM. In the latter cases, self-programming is accomplished via signals on dedicated I/O lines of the FPGA that are automatically generated internally. For simplicity and economics, this design uses a 32 kbyte parallel EPROM for automatic programming. The MAX+PLUS II software generates an Intel hex programming file as part of its final output, and this is loaded directly into a standard PROM programmer.

Circuit Construction and Testing

The circuit board construction was fairly standard, as the board consists of only a handful of parts. After all, consolidation is one of the main goals of FPGA usage. The main problem with this design is that the Altera 81500 chip that I used is most economically priced in a 240 pin quad flat pack package. This particular package has its leads spaced at only 0.5 mm and is virtually impossible to hand wire directly onto a breadboard. Because of this, I purchased a prototyping adapter. This adapter has a surface mount position for the chip on the top, with circuit board traces leading to a more conventional pin grid array arrangement on the bottom. This adapter can then be mounted on a breadboard with wires to the individual pins. This particular adapter was designed for a Motorola MC68360 and is manufactured by Emulation Technology, Inc. With the relatively low clock frequency (1 MHz) in this design and the FPGA outputs programmed for a slow slew rate, conventional point-to-point wiring is acceptable as long as the wires are reasonably short. For higher speed designs that have more I/O pins switching voltage levels simultaneously, this might not be acceptable, and a PC board may have to be constructed to handle the switching transients.

Initial testing of the FPGA with the radio was done by using an FPGA test board that contained a 1.0 MHz clock oscillator, a 32 kbytes EPROM and an Altera 81500 FPGA chip. This is a general purpose board in that only the parallel connections to and from the EPROM, along with power and ground to the FPGA were wired directly. All other chip I/O pins are routed to header pins that may be connected as needed to other chips or peripherals. This is the setup that is shown in Fig 5.

After the vacuum-fluorescent display and the DDS interface were wired to the FPGA test board header pins, power was applied. After some initial testing, it was determined that although the DDS interface was indeed functioning correctly and the receiver was tuning correctly, the characters on the display were not in the correct order and were not being updated correctly. After some telephone consultations with a Futaba engineer, I learned that there was an error in the design specification that came with the display unit. The specification says that the *maximum* time between bytes that are sent to the display unit must be 12 microseconds. What it should say is that the *minimum* time between bytes must be 12 microseconds. My initial design called for a delay period of 5 microseconds, so this was increased to 20 microseconds by providing a divide-by-4 clock to this portion of the circuit. This modification was made to the VHDL in a matter of minutes, and then after recompilation (I didn't bother to resimulate this minor change) a new EPROM was programmed. After replacing the EPROM, the circuit was powered up again and all functions worked correctly. This process is in stark contrast to what would have been necessary if the design were implemented in discrete digital logic chips. The modification would have required a separate chip to provide the two extra D flip-flops. If the design were wire-wrapped, the clock net would have been separated and rewired into the divide-by-4 circuit. If a PC board was used, then etch cutting and IC dead-bugging would have been necessary to make the change. Either case would have been time consuming and messy.

Design Methodology Review

Let's review what the technologies presented in this paper have done for us. First, the usage of an FPGA chip has allowed us to design a fairly complex digital logic circuit that previously would have taken many discrete ICs to build. The circuit design may be modified slightly or even totally redesigned by merely reprogramming a standard EPROM. By treating designs such as this as modules, it is possible to incorporate more than one module into a larger FPGA. As an example, I could save even more circuit board area by incorporating a DSP filter onto the FPGA that holds the tuner/display function demonstrated here. Second, using VHDL as the circuit description gives us a fast, efficient, technology independent, standard way of describing hardware, that may be machine read because of its software nature. An example of this extremely important concept is that anybody can download my VHDL source code from the ARRL site mentioned previously and compile it locally. This compiled VHDL can then be used with a digital logic synthesizer (the third major piece of this methodology) and the appropriate vendor libraries to implement the circuit. Since we are operating in a technology independent mode, it is quite easy to then synthesize to another vendor's chips, for example Xilinx or AT&T. This is much easier than if the design was initially described in schematic form, as considerable modification is generally necessary when transferring a schematic based design between different vendor technologies.

Between the FPGA chip itself, the VHDL code and the logic-synthesis stage of the project, it is probably more important for the designer to have an in-depth knowledge of VHDL and how to use it effectively than it is to understand the absolute low-level details of the FPGA architecture, or even how the synthesis software works. Of course the right chip must be chosen for the right job, and there are various choices within the simulation environment that can ultimately make or break a design, but in general the VHDL will have the largest effect on most designs such as the one described here. That is one of the main reasons why several detailed VHDL process examples have been given here.

More Economical Alternatives

The design environment that I have described in this paper consists of software packages from three separate vendors. The VHDL simulator is from Model Technology, the FPGA Compiler is from Synopsys, and the place and route tools are from Altera, all UNIX versions running on a Sun workstation. This "professional" level approach is great for the designer, but very expensive. Fortunately, there are some alternatives in the market that will allow development of fairly sophisticated FPGA devices without spending a fortune on software tools. To work with Altera FPGA devices exclusively, the MAX+PLUS II software package can be purchased with a VHDL compiler and a simulator. This package is available for PCs running Windows NT, as well as a UNIX version for workstations.

A true bargain is the offering from Cypress Semiconductor that supports their own programmable devices. These chips are smaller than the FPGA described in this paper, but may still be quite useful to amateur experimenters, and the design kit is geared more toward VHDL/programmable-logic beginners. This kit contains a book entitled *VHDL for Programmable Logic*, VHDL design software, a PC programming interface and samples of one of their programmable chip offerings. Cypress has recently advertised this product in several electronics trade magazines for only $175.

Conclusion

The emergence of large and fast FPGA chips that may be reprogrammed many times has created new opportunities for leading-edge amateur experimenters. The example described in this paper was one of low speed tuner and display control, yet these chips are powerful enough to also perform other advanced functions such as high-speed DSP at IF rates beyond those of standard DSP chips or microprocessors. New software methods, such as VHDL and digital logic synthesis, are giving designers the ability to create and modify working circuits with FPGA chips in considerably less time than was possible before. This style of digital design was previously available only to professional engineers. As prices of both chips and design software continue to fall, however, more opportunities for the experimenter and hobbyist will become available.

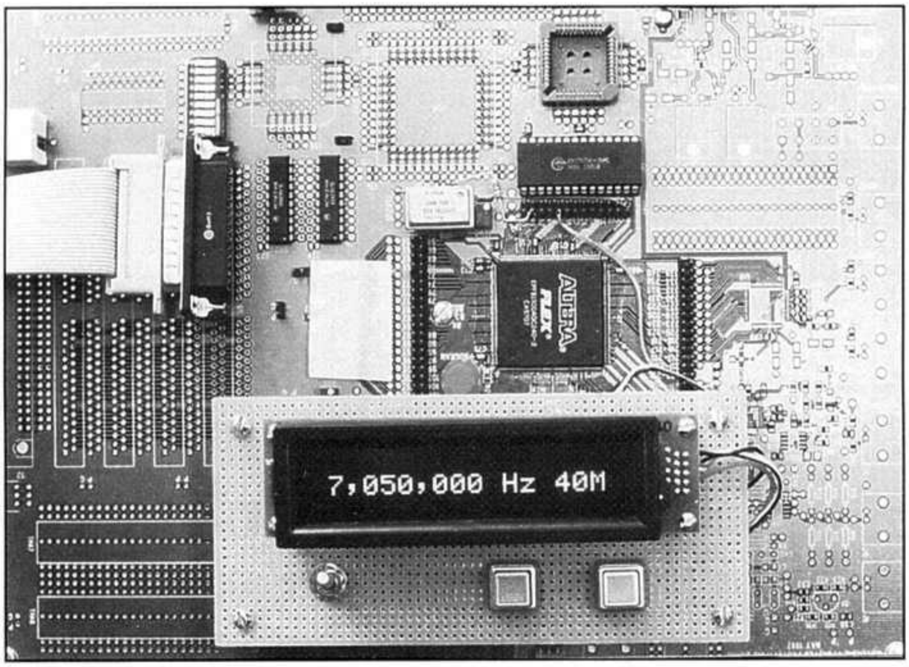


Fig 5—The Altera 81500 FPGA DDS and display circuitry in a prototype board. The toggle switch below the display module is the FINE TUNING control (10 or 1000 Hz tuning steps), and the two push buttons select UP or DOWN tuning.

VHDL Suggested Reading List

I have found the following textbooks to be quite useful for learning VHDL and as references for more advanced applications:

VHDL, by Douglas Perry (McGraw-Hill, 1994) ISBN 0-07-049434-7.

VHDL Techniques, Experiments and Caveats, by Joseph Pick (McGraw-Hill, 1996) ISBN 0-07-049906-3.

VHDL for Logic Synthesis, by Andrew Rushton (McGraw-Hill, 1995) ISBN 0-07-709092-6.

Internet Information Sources

These Internet newsgroups are useful sources of information on VHDL and FPGA issues for all levels of expertise: **comp.lang.vhdl**; **comp.arch.fpga**.

These Web sites contain product information for some of the software design packages and FPGA chips mentioned in this article: **http://www.synopsys.com** (Synopsys logic synthesis software); **http://www.altera.com** (FPGA software compiler and chips); **http://www.cypress.com** (low-end VHDL and programmable logic design starter kit).

Notes

[1] Curtis Preuss, "Building a Direct Digital Synthesis VFO," *QEX*, Jul 1997, p 3.

[2] Don Kirk, "The Ultimate VFO," *QEX*, Apr 1996, p 13.

[3] George Zimmerman and Michael Flanagan, "Reducing Spurious Output of NCOs: Random noise added to either amplitude or phase prior to truncation," *NASA Tech Briefs*, June 1994, reference #NPO-18789.

[4] Advanced Micro Devices, *PAL Device Handbook*, 1988, Section 1.

[5] You can download this package from the ARRL "Hiram" BBS (tel 860-594-0306), or the ARRL Internet ftp site: **oak.oakland.edu** (in the **pub/hamradio/arrl/qex** directory). In either case, look for the file VDHL.ZIP.

[6] Synopsys, *FPGA Synthesis Class Notes*, October 1995.

[7] Altera, *1996 Data Book*, Section 12.

Understanding the T-tuner (C-L-C) Transmatch

Want to build a T-network antenna tuner? Will it be too lossy? Are the components up to the task? Here's how to find out.

By William E. Sabin, WØIYH

The T-tuner circuit as shown in Fig 1 is widely used in transmitter multiband antenna tuners (transmatches). This article will take a close look at the interesting basic principles of this network as it is used in this application. I hope another result will be a better appreciation of its tuning procedures, its advantages and its limitations. These tuners often contain a balun of some kind for feeding a balanced transmission line, but we will not cover that topic in this article. We will also briefly compare the T and the L networks. Other articles are referenced and recommended.[1, 2, 3, 4, 5]

The complex output load impedance is usually the input impedance of a coax line or some kind of wire or rod (mast or whip) that is working against a reference ground plane. This load is shown in Fig 1 as the series combination $R2 \pm jX2$, and it is to be transformed by $C1$, L and $C2$ to $Zin = R1$, a 50 Ω resistive input impedance (SWR = 1.0). The RF power is to be delivered to $R2$, and the reactances $XC2$ and $X2$ are shown lumped together inside a dotted box as a single value of reactance XCo that constitutes the right arm of the T. In other words, the load reactance is "absorbed" into the tuner. To put it yet another way, if $R2$ is fixed and if $C1$ and L have the correct values, a wide range of $\pm X2$ can be "tuned out" by adjusting only $C2$. If $X2$ is capacitive, we increase $C2$ (reduce its reactance) and if $X2$ is inductive we reduce $C2$ (increase its reactance) so that the net reactance $XC2 \pm X2$ inside the dotted box has the value of *capacitive* reactance that the T network requires to make $R1$ = 50 Ω.

Fig 2 is a Smith Chart that traces a T-tuner path ABCDEF from a typical complex load to a 50 Ω pure resistance input. In this particular example the path ABVF is possible for an L-network. Also, two capacitors, series from A to W and shunt from W to F, is a very nice solution. The usefulness of the Smith Chart is apparent from this example.

The use of two capacitors and one coil (C-L-C) has the following advantages. It is economical to use just one variable inductor because of the cost of (a) a roller inductor plus

1400 Harold Dr SE
Cedar Rapids, IA 52403
e-mail **sabinw@crpl.cedar-rapids.lib.ia.us**

its turns counter or (b) a heavy duty RF wafer switch plus a set of fixed inductors. Coils have more loss (lower Q) than good tuning capacitors, so the C-L-C approach reduces (under most conditions) power losses in the tuner. Variable capacitors are less prone to difficult self-resonance problems (aka "suckouts"or energy absorptions) within the desired frequency range.

This C-L-C approach makes the tuner a high-pass circuit that does not provide as much harmonic attenuation as an L-C-L circuit would provide. This attenuation would otherwise be desirable. Fig 3 shows two frequency response plots, one for a capacitive load and one for an inductive load. Each plot assumes that the load is a constant resistance and constant capacitance or inductance that do not change with frequency. We will see later that this is usually unrealistic.

To get the maximum versatility with respect to impedance transformation and efficiency, it is desirable to make all three elements independently and continuously adjustable, if possible.

The Two-Port Loss-Less Network

The T-tuner is a good example of the *two-port network* theory that has been widely taught and used for many years. We will use this circuit as an example in a brief survey that will hopefully be interesting.

Fig 4 shows an idealized pure reactance circuit (no resistances in L, $C1$ and Co) with the terminal voltage polarities (at peak of sine wave) as shown and the input and output currents in the directions shown. XCo is the sum of $XC2$ (in the tuner) and $\pm X2$ (in the load). We are interested in the *Z parameters* of the loss-less two-port network in Fig 4

$$E1 = Z11 \cdot I1 - Z12 \cdot I2 \qquad \text{Eq 1a}$$

$$E2 = Z21 \cdot I1 - Z22 \cdot I2 \qquad \text{Eq 1b}$$

The minus signs associated with $I2$ are correct for the direction of current flow that is shown at the output terminals. $Z11$ is the input impedance that is determined when the output terminals are open-circuited ($I2 = 0$). We apply

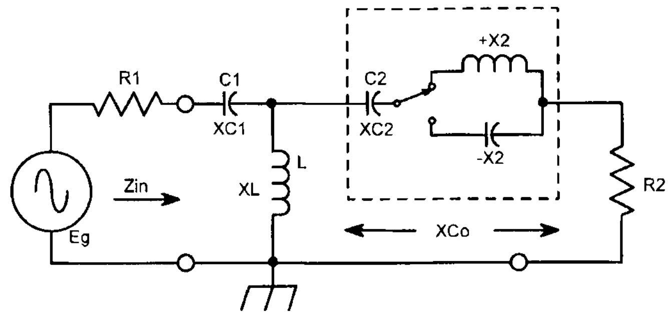


Fig 1—T-tuner connected to complex load impedance. The reactive part of the load (±X2) is absorbed into the tuner as XCo.

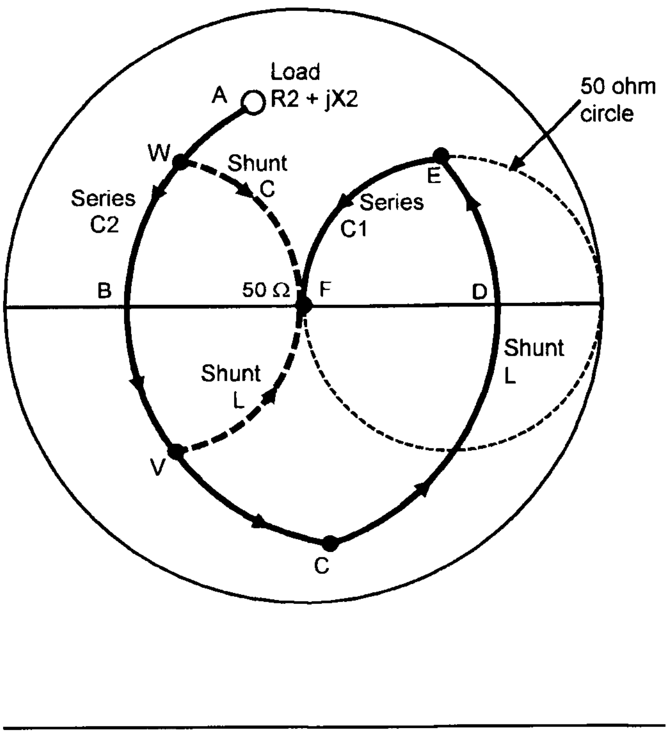


Fig 2—Smith chart for T-Tuner. Reactance of C2 transforms inductive load impedance along path ABC. Coil susceptance transforms along path CDE. C1 transforms path EF to 50 ohms. The dotted path ABVF corresponds to an L network solution. A two-capacitor solution is from A to W (series) and W to F (shunt).

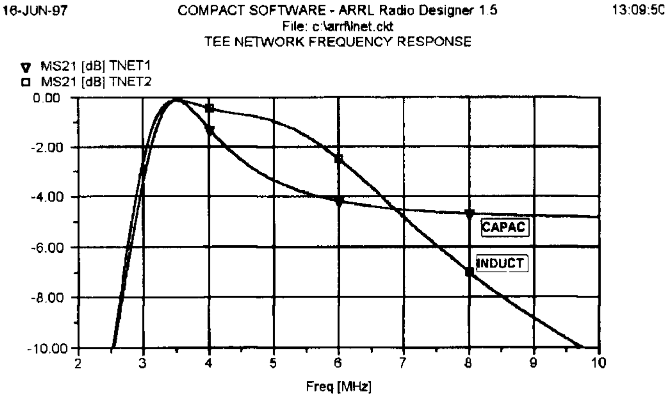


Fig 3—Selectivity of T-tuner for L and C loads (fixed Land C values).

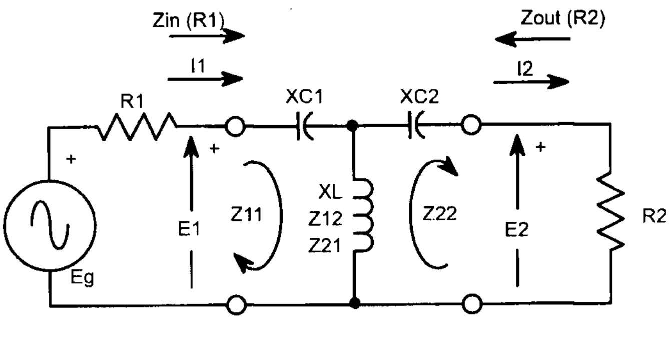


Fig 4—Diagram of two-port T network.

a current generator $I1$ at the input and measure the ratio $E1/I1$. Its value is $-jXC1 + jXL$. Likewise, the open-circuit output impedance is $-jXCo + jXL$. $Z12$ is the ratio of $E1$ to a current generator $I2$, applied to the output, when the input is open-circuited ($I1 = 0$). This value is jXL. $Z21$ also has this value. $Z12$ or $Z21$ is the mutual impedance or "coupling" between the input and output. If jXL goes to zero, the output goes to zero.

At the output, the voltage $E2$ is equal to current $I2$ times the load resistance $R2$. That is, $E2 = I2 \cdot R2$. If we put this value of $E2$ in Eq 1b and solve Eq 1b for $I2$ we get

$$I2 = I1 \cdot \frac{Z21}{R2 + Z22} \qquad \text{Eq 2}$$

and if we substitute this result in Eq 1a, we get an equation that relates $E1$, $I1$, L, $C1$, Co and $R2$. We can then find the ratio $E1/I1$, which is the input impedance Zin (with load $R2$ connected)

$$Zin = \frac{E1}{I1} = Z11 - \frac{Z12 \cdot Z21}{Z22 + R2} = (-jXC1 + jXL) - \frac{(jXL \cdot jXL)}{(-jXCo + jXL) + (R2)} \qquad \text{Eq 3}$$

where XCo is the sum of $XC2$ and $\pm X2$. This sum must be a capacitive reactance. That is, since XCo and $XC2$ are positive numbers, the sum of $XC2$ and $\pm X2$ must also be a positive number. The $-j$ makes the sum a negative (capacitive) reactance.

We then separate Eq 3 into two parts, a real part that is pure resistance and an imaginary part that is pure reactance. The goal then, knowing the values of $\pm X2$ and $R2$, is to find the values of $C1$, $C2$ and L that make the input impedance a pure resistance, 50 Ω, with zero reactance. That is, $Zin = R1 = 50 + j0$ Ω. We will not present the tedious details, but just give the results. First we must note that there are three things to find and only two equations (the real part and the imaginary part of Zin). To do this we must assign a value to one of the components. We will choose to assign a value to XL. The reason for this will become clear. We then set the real part of Eq 3 equal to $R1$ (50 Ω) and solve for the value of XCo. Then we set the imaginary part of Zin equal to zero, eliminate XCo in this equation by replacing it with the first equation, and get the value of $XC1$.

$$XCo = XL + R2 \cdot \sqrt{\frac{XL^2 - R1 \cdot R2}{R1 \cdot R2}} \qquad \text{Eq 4a}$$

$$XC1 = XL + R1 \cdot \sqrt{\frac{XL^2 - R1 \cdot R2}{R1 \cdot R2}} \qquad \text{Eq 4b}$$

Having determined XCo, and knowing $\pm X2$ (which we recall is part of the load impedance), we can find the required reactance of $C2$, which is

$$\begin{aligned} &XC2 = XCo + X2 \quad \text{(if } X2 \text{ is inductive)} \\ &\text{or} \\ &XC2 = XCo - X2 \quad \text{(if } X2 \text{ is capacitive)} \end{aligned} \qquad \text{Eq 5}$$

and L and C are found using the standard formulas

$$L = \frac{XL}{2 \cdot \pi \cdot F}, \; C1 = \frac{1}{2 \cdot \pi \cdot F \cdot XC1}, \; C2 = \frac{1}{2 \cdot \pi \cdot F \cdot XC2} \qquad \text{Eq 6}$$

Choosing XL and Tuning Out Load Reactance

If we rearrange Eq 4a as shown in Eq 7 we can get some insight regarding our initial choice of XL. In this equation XCo is separated into $XC2$ and $\pm X2$. $XC2$ and XL are always positive numbers but $X2$ needs a + sign for inductive reactance and a – sign for capacitive reactance.

$$XC2 = XL + R2 \cdot \sqrt{\frac{XL^2 - R1 \cdot R2}{R1 \cdot R2}} \pm X2 \qquad \text{Eq 7}$$

First, we see that XL^2 must always be greater than $R1 \cdot R2$ so that the quantity under the square-root sign does not go negative. This determines the absolute minimum value of XL. For example, if $R1$ is 50 Ω and $R2$ is 500 Ω (minimum SWR=10) then XL can never be less than 158 Ω. Large values of XL are undesirable, as we discuss later on.

There is a further constraint: $XC2$ must always be a positive number, as mentioned before. So the right hand side of Eq 7 must be equal to the left side and positive. For any variable capacitor $XC2$, its minimum reactance (maximum capacity) is limited, unless we add a fixed capacitor in parallel. So if $X2$ is too negative, making the right side less than the small $XC2$ (that is, if the load has a large capacitive reactance), then XL must be increased beyond the absolute minimum. As XL gets large, Eq 4b says that $XC1$ also gets larger ($C1$ becomes smaller).

On the other hand, if $X2$ is inductive (positive), then Eq 7 says that XL can be small and also, $XC2$ becomes a larger capacitive reactance ($C2$ becomes smaller) in order to "tune out" the inductive $X2$. The maximum reactance (minimum capacitance) of $C2$ is also limited, and a small value of $C2$ also creates a voltage-rating problem for $C2$ (discussed later). In the limit, if we set XL close to its lowest permitted value, then the maximum inductive reactance of $X2$ that can be tuned out is determined by this minimum capacitance of $C2$. $C1$, according to Eq 4b, then becomes large, sometimes quite large. A fixed capacitor across the variable is helpful here also.

The study of these trade-offs is quite complicated, and is in fact one of the main objects of this article, so we will come back to it later in a more systematic and orderly manner. Otherwise, we get into a very messy and tedious discussion.

Selectivity and Operating Q

The T-tuner has selectivity near the resonant frequency (see Fig 3). In Fig 1, if $R2$ is constant and if $X2$ corresponds to a certain value of capacitance or inductance that does not change with frequency, then we have a "textbook" circuit that has a certain frequency response (ratio of output $E2$ to input Eg) as seen in Fig 3. In tuned circuits the 3 dB bandwidth of frequency response is often used to calculate an *operating* Qo:

$$Qo = \frac{F(0\text{ dB})}{F_{\text{HI}}(-3\text{ dB}) - F_{\text{LO}}(-3\text{ dB})} \qquad \text{Eq 8}$$

As we see in Fig 3, the value of Qo can often be difficult to establish and not very accurate, using this 3 dB method. But if we zoom in on Fig 3 at the resonant frequency a definite frequency rolloff pattern is visible. As L becomes larger and $C1$ and $C2$ become smaller, this effect becomes more pronounced.[3]

At this point in the article L, $C1$ and $C2$ are still loss-less (their Qs are very large). We can also calculate a more accurate value for Qo from the circuit of Fig 4 by noting that $C1$ is loaded by $R1$, which defines $Q1$, and Co is loaded by $R2$, which defines $Q2$. Then we see that

$$Qo = \frac{XC1}{R1} + \frac{XCo}{R2} = Q1 + Q2 \qquad \text{Eq 9}$$

This calculated Qo approaches the measured value found in Eq 8 as Qo becomes quite large.

We need to justify Qo as found in Eq 9. In Fig 4 the *energy* that is stored in the reactances travels back and forth between the coil and the capacitors. This means that the *peak* reactive energy stored in the circuit can be calculated by looking at just the peak energy in the capacitors. Qo is, by definition, proportional to the ratio of this peak stored energy to the *stored* energy (set $Eg = 0$) that is dissipated in the circuit over one cycle of the RF. If we repeat this exact process (same value of peak stored energy) in each cycle of RF we get the steady state result at resonance,

$$Qo = \frac{I1^2 \cdot XC1}{I1^2 \cdot R1} + \frac{I2^2 \cdot XCo}{I2^2 \cdot R2} = Q1 + Q2 \qquad \text{Eq 10}$$

This value of Qo accounts for all the energy storage in the filter and all the stored energy that is dissipated in $R1$ and $R2$. The continuous energy (power) supplied by Eg was not used in this derivation. Observe also that if $R1$ is a dynamic (nondissipative) generator resistance, some of the stored energy is not dissipated, but returned to the generator.

The significance of Qo is that it determines the rate of rolloff near resonance. From Eq 9 we see that sharp selectivity requires large values of XC (small values of C) and to achieve resonance, Eq 4 says that large values of L are needed. Since high selectivity is not usually a priority in a tuner, we would prefer large C and small L, to reduce Qo and the need to retune over a frequency range.

There is another observation about selectivity that is interesting. If we think of the T-network as two back-to-back L-networks, one transforming downward in impedance from the coil to $R1$ and the other transforming downward in impedance from the coil to $R2$, the impedance at the coil is always greater than either $R1$ or $R2$, whichever one of those is greater. This means that the T-tuner always has a higher Qo (selectivity) than a single L-network that transforms from $R1$ to $R2$. We will soon see that this higher selectivity is associated with greater tuner losses, when using imperfect components.

There are two things wrong with this selectivity discussion. One is that the tuner output load R and L (or C), which we assumed were constant (as in Fig 3), are frequently not constant but instead vary, sometimes quite rapidly, with just moderate frequency changes. For example, a resonant shortened trap dipole might change very quickly from inductive to capacitive. This would tend to modify the overall selectivity considerably over a moderate frequency range. With some cleverness, it is often possible to model the varying load as an LCR network (fixed values) of some kind within this frequency range.[6] Computer-designed antenna impedance simulators use this method. Another method is to use a list of load impedance values at a set of frequencies, and do a point-by-point frequency response plot. Mathcad or a spreadsheet can do this very nicely. Making the network tune as broadly as possible may help to minimize retuning across a ham band.

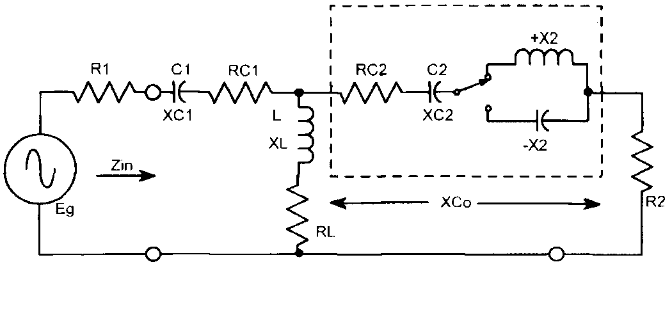


Fig 5—T network with loss resistances added to L, C1 and C2.

Another problem is that the input end of the network may not be properly terminated by the generator. Usually, the generator is a power amplifier (PA) whose dynamic output impedance is unknown and difficult to measure without a special test setup. The analysis that led to Eq 4 assumed a conjugate match at both ends of a *loss-less* tuner. We can easily get 50 Ω looking into the input of the network, but the impedance looking from the input back into the generator may not be 50 Ω. If this is true, we cannot say that the output of the tuner is conjugate matched either. As two realistic examples, suppose the PA is a high-impedance source. The selectivity contributed by $C1$ would be small. If the PA were a very low-impedance source then the selectivity contributed by $C1$ would be large. The textbook situation is that $C1$ is loaded by a 50 Ω physical or dynamic resistance.

The bottom line is that it's important to provide the 50 Ω load that the PA wants to see at the operating frequency. Whether everything is exactly conjugate matched or not is not terribly important. Nonetheless, we would like for the tuner itself to have as little selectivity as possible.

Tuner Losses

The inductor and the capacitors have loss resistances and therefore finite Q values. For most amateur HF transmitting equipment the values $QC = 1000$ and $QL = 200$ to 250 (for variable roller coils) are commonly accepted values. These values of Q cannot generally be considered constant as the capacitors and the coil are adjusted or as the frequency is changed. Accurate Q measurements are not feasible for most amateurs, so we usually use the above values as "best guess" values. Some errors often result from these approximations. The resistances corresponding to these Q values are $R = X / Q$. Fig 5 shows these resistances added to the loss-less tuner. The Z parameters of Eq 1 are modified as follows:

$$Z11 = RC1 - jXC1 + RL + jXL \qquad Z12 = RL + jXL$$
$$Z21 = RL + jXL \qquad Z22 = RC2 - jXCo + RL + jXL$$
$$\text{Eq 11}$$

and Eq 2 and 3 can be used to find $I2$ and Zin using these new values. The complex algebra is very tedious when done by hand, so a Mathcad worksheet will be described later that finds everything of interest very quickly. We are interested in the power loss in each component and the tuner efficiency.

An approximate idea of the losses can be had using the following procedure and example when the input is tuned to $R1 = 50\ \Omega$ (SWR = 1.0) and $XL = j200\ \Omega$, $XC1 = -j239\ \Omega$, $XC2 = -j87\ \Omega$ and the load is $R2 = 500\ \Omega$, $X2 = -j500\ \Omega$:

1. The power input is 1500 W into 50 Ω, so the input current is
$I1 = \sqrt{Pin / R1} = \sqrt{1500 / 50} = 5.48$ A.
2. $XC1 = 239\ \Omega$ and $QC1 = 1000$, so
$RC1 = 239 / 1000 = 0.239\ \Omega$.
3. The power loss in $C1$ is $I1^2 \cdot RC1 = 5.48^2 \cdot 0.239 = 7.18$ W.
4. The voltage input is $E1 = \sqrt{Pin \cdot R1} = \sqrt{1500 \cdot 50} = 274$ V.
5. The voltage at L is

$$VL = \sqrt{(I1 \cdot XC1)^2 + E1^2} = \sqrt{(5.48 \cdot 239)^2 + 274^2} = 1338\ V.$$

(the voltages add RMS-wise).

6. The current in L is $IL = VL\ /\ XL = 1338\ /\ 200 = 6.69$ A.
7. The coil reactance is 200 Ω, so $RL = XL\ /\ QL = 200\ /\ 250 = 0.80\ \Omega$.
8. The coil loss is $IL^2 \cdot RL = 6.69^2 \cdot 0.8 = 35.8$ W.
9. The current in $C2$ is

$$I2 = \frac{VL}{\sqrt{R2^2 + (XC2 + X2)^2}} = \frac{1338}{\sqrt{500^2 + (87 + 500)^2}} = 1.74 \text{ A}$$

In this formula, $X2$ is positive for capacitive, negative for inductive load, and resistance and reactance add as the sum of squares (RMS).

10. The resistance of $C2$ is $XC2\ /\ QC2 = 87\ /\ 1000 = 0.087\ \Omega$.
11. The power loss in $C2$ is $PC2 = I2^2 \cdot RC2 = 1.74^2 \cdot 0.087 = 0.26$ W
12. The total power loss is $PC2 + PL + PC1 = 0.26 + 35.8 + 7.18 = 43.24$ W
13. The efficiency is $(1500 - 43.24)\ /\ 1500 = 0.971 = 97.1\%$
14. The dB loss is $-10 \log (0.971) = 0.13$ dB

In this example the efficiency is very good, but note that the coil may get a bit warm. It also refers to continuous key down conditions that are seldom realistic in SSB/CW/data modes. Quite often this kind of calculation is adequate, in view of the uncertainties in the various other quantities involved. For this approach we need numbers for $C1$, $C2$ and L, obtainable from Eq 4, 5 and 6. It is possible, and not at all difficult, to calibrate the tuning dials for their L and C values using a digital C meter and a grid dipper. We need numbers for the load $R2$ and $X2$; these are either measured or (more often) obtained from antenna simulator programs plus transmission-line calculations.[7]

Mathcad Solutions

Figs 6 and 7 are the Mathcad solution of the T-tuner, using exact equations and giving very accurate data for the tuner performance. A careful inspection of the various segments of these worksheets will show that all of the things mentioned previously in this article, plus some others, are being calculated. A trial value of XL (greater than the minimum permissible value) is chosen by the user, and everything else is automatically calculated in a split second.

When the L and C losses are added in, the input SWR and impedance that are calculated are a little off target. We give it the coil reactance, the Q and load values and it returns the exact results that correspond to those inputs. To get the exactly correct $R1 = 50\ \Omega$ and SWR = 1.0 would require some fine-tuning iterations of the reactance values. These worksheets do not do that. In the many examples that I have worked out, these errors are small compared to inaccuracy of the input data, such as load impedance and Q values. In practice, we would manually fine tune the tuner, which we always do anyway.

Of particular interest are the values of $XC1$ and $XC2$. These must be positive real numbers. If they come out negative or complex, too close to zero or too large, then we must look for a different value of XL that gets everything into real-world range. This also applies to the tuner dissipation values. If efficiency and dissipation become intolerable, try to reduce the value of XL. If the required capacitance values are beyond the range of $C1$ or $C2$ we can add fixed capacitors in parallel. If this doesn't work well enough then we need to think about changing the load impedance or using a different kind of tuner. In particular, a more efficient L-type tuner can sometimes be specially designed for one specific difficult load impedance. Shorting out $C1$ or $C2$ may help to get this done. Also, conventional or transmission-line transformers are often helpful, if used properly.

These Mathcad worksheets can be downloaded from the ARRL/*QEX* Web site.[8] The equations on the Mathcad worksheets can also be implemented, but not nearly as easily, in a spreadsheet (such as Excel) that has complex algebra capability added.

Design Guidelines

We will now try to tie everything together, to see how the T-tuner can be designed and tuned for various values of load impedance, using specific examples. Table 1 is a collection of data obtained from the Mathcad worksheets that will put some perspective into the capabilities and problems of the T-tuner.

The top row shows various load impedances. The coil Q is 250, capacitor Q is 1000, power is 1500 W input, frequency is not specified. In the first column, we choose XL values, starting with the minimum permissible, 160, then 500, then 1000 and we look at the items in that column. The capacitor voltages are peak values. The operating Qo increases from 3.8 to 17 to 35, the power dissipated in the tuner goes up rapidly and the efficiency quickly becomes poor. In particular, coil loss rises to high levels. Coil and capacitor voltages become large.

This example demonstrates clearly that for given values of coil Q, load impedance and input power, the coil dissipation increases rapidly with coil reactance (or equivalently, with operating Qo). This is a key item in understanding the T-tuner and its limitations. It might be thought that less coil would be worse but this is not true at all. Using the numbers in the Mathcad worksheet example, if we double XL from

Table 1. T-tuner Performance Data

R2+ jX2	*500 + j0*	*500 – j500*	*500 + j500*	*5 + j1000*	*5 – j1000*
Choose XL	160;500;1000	200;500;1000	170;500;1000	16	800
XC1	168;650;1312	239;650;1312	190;650;1312	23.7	3329
XC2	237;2000;4122	87;1500;3622	867;2500;4622	1017	53
Qo	3.8;17;35	5.9;17;35	4.5;17;35	3.8	277
PL	22.6;94;176	34.7;94;177	26.6;94;176	19.5	761
PC1	5.0;18;33.6	7.0;18;33.6	5.6;18;34	0.9	46
PC2	0.7;5.5;10.5	0.26;4.2;9.3	2.5;6.9;11.8	250	7.3
VC1	1289;837;9391	1825;4839;9395	1457;4836;9387	202	17449
VC2	576;4703;9326	211;3529;8198	2100;5876;10452	22549	877
VL	1344;4849;9395	1863;4851;9399	1504;4847;9391	395	17445
Eff. %	98.1;92.2;85.3	97.2;92.2;85.4	97.7;92.1;85.2	82	45.7

200 Ω to 400 Ω, the coil voltage approximately doubles, the coil current remains about the same and the coil dissipation approximately doubles. Looking at column one (500+j0) of Table 1 leads to the same conclusion. An exception occurs if the coil Q gets smaller at low or high values of inductance. A good coil design would try to reduce these Q variations.

In the next two columns of Table 1 the load becomes reactive. The minimum XL is still quite low and is very similar to the first column. The reason is that $C2$ is able to tune out the load reactance rather easily. The capacitor voltages show even greater increases than in column one, however.

In the last two columns we have extreme situations, low $R2$ and large $X2$. For the inductive case (+j1000), the tuning inductance XL is quite low, but the power dissipated by $C2$ reaches an astonishing 250 W (assuming that Q is constant as $C2$ is varied, which is most probably not correct and illustrates a possible difficulty with using assumed values of Q) and a voltage of 22,549 V. This capacitor would have to be a special high-Q, high-voltage vacuum type. The $C2$ current is very large also, so lead resistances are critical. For the capacitive load (–j1000), we see that the minimum inductive value is very large, 800 Ω, with a dissipation of 761 W. The coil and $C1$ have extremely high voltage. The operating Qo is 277, an extremely high number, and in fact greater than the coil Q of 250, which illustrates that the accuracy of the calculations may slip somewhat (but not drastically) in situations like this.

This data indicates that the T-tuner is sometimes *not* well suited for highly reactive loads, especially capacitive loads. In these situations, the reader is encouraged to try other approaches that are more in line with the operational needs. Of particular interest is the comparison with other tuner types, in particular the L-circuit, which is sometimes better when custom-designed for one particular difficult value of load. For example, if we refer again to the Smith Chart of Fig 2 for an inductive load, an L-circuit path ABVF uses a smaller series capacitive reactance (larger $C2$) in path ABV and a much smaller shunt inductive susceptance (larger L) in path VF, and this would be inherently more efficient (the operating Qo is less). At the same time, the voltage across L is reduced and the loss in $C1$ is eliminated, for additional efficiency. The two-capacitor solution suggested in Fig 2 (series A to W and shunt W to F) would be better yet, for this particular value of inductive load impedance. Similarly for a highly capacitive load, two inductors or a single tapped inductor would be a good approach, as the Smith Chart would indicate. The Smith Chart feature of the *ARRL Radio Designer* program is especially nice for this kind of work. If the load has a very high capacitive (or inductive) reactance, try to do something to the load to make its reactance much less.

Induced Currents

In some antenna installations, a coax transmission line in the near field of the radiator picks up a stray current that travels down the feed line toward the tuner. This current has a tendency to detune to some extent (usually not a great extent) the feed-point impedance that the transmission line sees. The output impedance of the tuner may have some effect on the flow of this current, and therefore on the antenna impedance itself. This interaction can conceivably cause some complication in the tune-up process. A coax current (choke) balun at the output of the tuner can help to eliminate this problem, if it exists.[9] A good way to check for this current is with a clamp-on RF ammeter around the coax braid at the output of the tuner.

To Summarize

The T-tuner is a good, general-purpose, versatile circuit, but it should never be used beyond its voltage or power limits or if it becomes too inefficient (even at low power levels). If *some* loss of signal (say 1 dB or so, for difficult values of load impedance) is tolerable and if overheating does not occur, then the T-tuner is quite acceptable. At higher power levels, however, 1 dB of loss can cause damage, and we must try to improve efficiency. To reduce losses in the T-tuner, consider the following general rules:

- Make coil Q as high as possible. This is important even at low power levels to improve efficiency.
- Make the $C1$ and $C2$ values large enough that small values of L can be used. This problem is most severe on the lower frequencies where the necessary variable capacitors become physically large and 1000 pF or more may be needed for good efficiency. Fixed padder capacitors in parallel with $C1$ and $C2$ are helpful in meeting this requirement. It is interesting to draw a 5:1 SWR circle on a Smith Chart and see how the $C1$ and $C2$ requirements (capacitance and voltage) vary, say at 3.5 MHz, as we go around the circle in such a way as to maintain high efficiency. In particular, note that if sufficient $C1$ and $C2$ are not available, the tuner losses become quite large.
- Consider the option of shorting out $C1$ or $C2$, if that will help to improve efficiency in certain difficult cases. Small RF knife switches are useful for that.
- When tuning up, bring up the power level gradually, while monitoring the temperature rise of the components. If something gets hot, look for ways to reduce the heating (reduce current or loss—*Ed*). The discussion of basic principles presented in this article should be helpful. As efficiency improves for some particular power level, temperatures drop.
- One good way to improve efficiency is to use an RF ammeter (or clamp-on meter such as the Palomar PCM-1) at the output of the tuner. The idea is to maximize the RF current, for a certain fixed value of power input to the tuner, while maintaining 1:1 SWR at the tuner input. Do this at a reduced power level.
- Keep in mind the realistic duty-cycle factors. 1500 W continuous is seldom real-world. In SSB speech, the average power is never greater that 350 W or so. In CW or an SSB two-tone test, 750 W is worst case. In some data modes, some derating may be needed. Any tuner can be burned up, if we are not careful.
- Look at the load and see if it can be made less reactive. A series or shunt L or C or a transmission-line stub or segment may help. Put the Smith Chart to work on the problem. *Antenna Impedance Matching* is an excellent source of understanding and techniques for this general topic.[10]
- If all this fails using the T-tuner, look for a different approach. For balanced, open-wire lines, consider the various link coupled series or parallel-tuned coupler schemes.[11]

The *TLA*, *AAT* and *Radio Designer* Programs

The ARRL Antenna Book (in particular the 18th edition[5]) software disk contains, among others, the *TLA.EXE* and *AAT.EXE* programs, that are excellent and have been highly refined by Dean Straw at ARRL. Besides the T-tuner, they calculate PI and L tuners. The solutions are iterated to get an exact 50 Ω input. I highly recommend these programs. The Mathcad worksheets are also valuable because they put the actual equations on the screen so that we are, in essence, "doing mathematics" in a "scratchpad" environment. Other equations and experiments can be added by the user, for example: dummy antenna calculations and frequency sweeps. The solutions can also be further analyzed over a frequency range using the *ARRL Radio Designer* program's special features, and the interactive Smith chart feature is especially helpful. The approximate analysis given in this article is also quite good for tuners that have an efficiency of, say 90% or better.

DESIGN LOSSLESS TEE (C-L-C) TRANSMATCH NETWORK

Source and load impedances

Tee input R1	**Load R2**	**Load X2** **XL= +, XC= -**
$R1 := 50$	$R2 := 500$	$X2 := 500$

Choose XL >$\sqrt{R1 \cdot R2}$ $XL := 170$

Minimum XL

$\sqrt{R1 \cdot R2} = 158.114$

XL and XC1, XC2 are positive numbers

Calculate XC2 $XC2 := XL + R2 \cdot \sqrt{\dfrac{XL^2 - R1 \cdot R2}{R1 \cdot R2}} + X2$ $XC2 = 867.484$

Calculate XC1 $XC1 := XL + R1 \cdot \sqrt{\dfrac{XL^2 - R1 \cdot R2}{R1 \cdot R2}}$ $XC1 = 189.748$

Find total operating Q $Q := \dfrac{XC2 - X2}{R2} + \dfrac{XC1}{R1}$ $Q = 4.53$

Specify Frequency

$F := 3.5 \cdot 10^6$

Find C1, C2, L values

$C1 := \dfrac{1}{2 \cdot \pi \cdot F \cdot XC1}$ $C2 := \dfrac{1}{2 \cdot \pi \cdot F \cdot XC2}$ $L := \dfrac{XL}{2 \cdot \pi \cdot F}$

$C1 = 2.396 \cdot 10^{-10}$ $C2 = 5.242 \cdot 10^{-11}$ $L = 7.73 \cdot 10^{-6}$

Include losses in the capacitors and the coil. Add loss resistances to the Z two port network. Get input SWR and peak voltages.

Coil Q **Capacitor Q**

$QL := 250$ $QC := 1000$ $RC1 := \frac{XC1}{QC}$ $RC2 := \frac{XC2}{QC}$ $RL := \frac{XL}{QL}$

Z matrix

Z11 Z12
Z21 Z22

$Z11 := RC1 + RL - XC1 \cdot j + XL \cdot j$ $Z21 := RL + XL \cdot j$

$Z12 := RL + XL \cdot j$ $Z22 = RC2 + RL - (XC2 - X2) \cdot j + XL \cdot j$

Input impedance $Zin := Z11 - \frac{Z12 \cdot Z21}{Z22 + R2}$ $Zin = 50.913 - 0.505j$

Input SWR $Z0 := 50$ $SWR := \frac{|Zin + Z0| + |Zin - Z0|}{|Zin + Z0| - |Zin - Z0|}$ $SWR = 1.021$

Specify Power Input $Pin := 1500$ $Ein := |Zin| \cdot \sqrt{\frac{Pin}{Re(Zin)}}$ $Ein = 276.365$

L and C losses $Ic1 := \frac{Ein}{Zin}$ $|Ic1| = 5.428$ $Prc1 := (|Ic1|)^2 \cdot RC1$ **C1** $Prc1 = 5.59$

$Ic2 := \frac{Ein \cdot Z21}{Z11 \cdot (R2 + Z22) - Z12 \cdot Z21}$ $|Ic2| = 1.712$ $Prc2 := (|Ic2|)^2 \cdot RC2$ **C2** $Prc2 = 2.542$

$Icoil := Ic1 - Ic2$ $|Icoil| = 6.256$ $PL := (|Icoil|)^2 \cdot RL$ **L** $PL = 26.61$

Tuner loss

$Ploss := Prc1 + Prc2 + PL$ $Pout := Pin - Ploss$ **dB Tuner Loss** **Efficiency %**

$Ploss = 34.7$ $Pout = 1465.3$ $-10 \cdot \log\left(\frac{Pout}{Pin}\right) = 0.102$ $\frac{Pout}{Pin} = 97.7 \cdot \%$

Capacitor and coil peak voltages

$Vc1 := |Ic1| \cdot XC1 \cdot \sqrt{2}$ $Vc2 := |Ic2| \cdot XC2 \cdot \sqrt{2}$ $VL := |Icoil| \cdot XL \cdot \sqrt{2}$

$Vc1 = 1457$ $Vc2 = 2100$ $VL = 1504$

Notes

[1] Griffith, A. S., W4ULD, "Getting the Most Out of Your T-Network Antenna Tuner," *QST*, Jan 1995, p 44.

[2] Wingfield, E. A., W5FD, "New and Improved Formulas for the Design of Pi and Pi-L Networks," *QST*, Aug 1983, p 23.

[3] Newkirk, D., W9VES, "Exploring RF," *QST*, Jan 1995, p 80.

[4] "Four High-Power Antenna Tuners," Product Review, *QST*, Mar 1997, p 70.

[5] *The ARRL Antenna Book*, 18th edition, 1997, Chapter 25. ARRL publications are available from your local ARRL dealer or directly from ARRL. Mail orders to Pub Sales Dept, ARRL, 225 Main St, Newington, CT 06111-1494. You can call us toll-free at tel 888-277-5289; fax your order to 860-594-0303; or send e-mail to **pubsales@arrl.org**. Check out the full ARRL publication's line on the World Wide Web at **http://www.arrl.org/catalog**.

[6] Haviland, R. P., W4MB, "Dummy Antennas," *Communications Quarterly*, Summer 1994, p 87.

[7] Sabin, W. E., WØIYH, "Computer Modeling of Coax Cable Circuits," *QEX*, Aug 1996, p 3.

[8] You can download this package from the ARRL "Hiram" BBS (tel 860-594-0306), or the ARRL Internet ftp site: **oak.oakland.edu** (in the **pub/hamradio/arrl/qex** directory). In either case, look for the file TMATCH.ZIP.

[9] Sabin, W. E., WØIYH, "Exploring the 1:1 Current (Choke) Balun," *QEX*, Jul 1997, pp 12-20.

[10] Caron, Wilfred N., *Antenna Impedance Matching* (Newington: ARRL, 1989).

[11] Grammer, G., W1DF, "Impedance Matching with an Antenna Tuner," *QST*, Oct 1946, p 38. (An excellent discussion of link-coupled tuners for balanced lines.)

Suggested Additional Reading

Imamura, B. K., JA6GW, "A T-Network Semiautomatic Antenna Tuner," *QST*, Apr 1980, p 26.

Measures, R., AG6K, "Balanced Antenna Tuner for Balanced Feedlines," *QST*, Feb 1990, p 28.

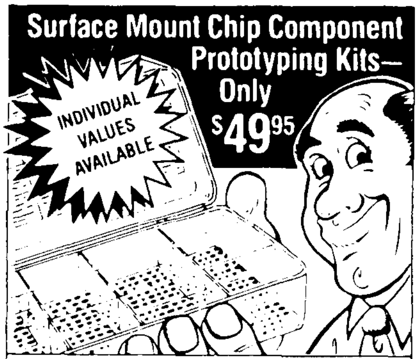
Maxwell, Walter, W2DU, *Reflections*, Chapters 14 to 17. (Newington: ARRL, 1994). (Information on matching networks for antennas.)

Schmidt, K., W9CF, "Estimating T-Network Losses at 80 and 160 Meters," *QEX*, July 1996, p 16.

Call for Papers

The Twelfth Annual Southwest Ohio Digital Symposium will take place Saturday January 17, 1998, from 9 AM to 4 PM at Thesken Hall, Middletown Campus, Miami University, Middletown, Ohio.

The event needs papers about beginning and intermediate packet, AMSAT, APRS, the OHIONET high-speed packet network and any other items relating to digital communications within Amateur Radio. Send an abstract (no more than 100 words) and a brief biographical sketch to Hank Greeb, N8XX, 6580 Dry Ridge Rd, Cincinnati OH 45252-1750, tel 513-385-8363 after 6 PM; packet **n8xx@w8mwo.oh.usa**, e-mail **72277.706@compuserve.com**.



Just Tune to the Dip

How Linear Amplifiers Work

By R. P. Haviland, W4MB

This article stems from trying to understand what really goes on in the design, tune-up and operation of linear amplifiers. In the process of study I found that there was no single source that covered the entire process of linear amplification. There were aspects that no source really addressed. Accordingly, the goal of the paper is to get a complete picture of a linear from conception, through design, first power application and use. At the same time, a purpose has been to keep the subject as simple as possible, by using concepts and graphs rather than mathematics.

[1]Notes appear on page 30.

1035 Green Acres Circle N
Daytona Beach, FL 32119
e-mail **bobh@america.com**

The Design Amplifier

In keeping with extensive use over the years, a single 6146 stage was selected for an example amplifier. Partly, this selection was due to the good availability of data, and of experience. True, the tubes are commonly used in pairs, but this is easily handled by doubling all values of currents and powers, and halving all resistances. The principles used are fully applicable to higher power amplifiers. They are also applicable to solid state amplifiers, although these lead to quite different operating conditions. A few of these will be mentioned, but not explored—this is going to be about vacuum-tube amplifiers.

In keeping with current practice, the design is for single-sideband voice signals, the service we call "linears." The main feature of this is that the duty cycle is low, ie, the average power needed is much less than the peak power. This means that the design can be made on the basis designated by RCA (Reference 1) as ICAS, Intermittent Commercial and Amateur Service. The limiting factors they give are:

Plate dissipation: 25 W
Plate voltage: 750 V

There are ways of reaching the design by numerical values only. The other way is to use graphical solutions. This is the method used here, because it helps visualize what is going on during operation.

Graphical Solution Of Voltage, Current, Resistance Problems

Fig 1 is a schematic representation

of a tube, a load resistance and a power supply. The relations among these elements are, for the quantities shown and in computer notation:

Eb = Ebb – RL × IL
Eb = RS × IP
IL = IP

These are solved graphically in Fig 2 for a particular value of the RS and RL by plotting two lines, starting from the axis IL = 0 with slopes RS and –RL, and finding their intersection.

Note the two cross-hatched areas. Since these represent products of voltage and current, they represent powers. The area between the zero line and the point of crossover (e_b – I_p) is the power in the resistance RS, that between crossover and Ebb the power in RL, with the sum representing the total power in the circuit.

The resistance RS is shown as variable. This follows from the English concept of a vacuum tube being a valve. Its resistance is controllable, by the tube element we call the grid. As the value of RS is being varied, the relative power in the two resistors will vary also. This can be handled by drawing several resistance lines, calculating the powers multiplied by the time spent with each resistance, and dividing by the total time to get the average power. In simple truly linear cases this can be reduced to a relation between the maximum and minimum powers. Where distortion is present, one or more intermediate values must be used.

To really show the various relations resulting from variations in RS we need the more complex plot of Fig 3 for the tube type we are using. Basically, this is a family of plate current versus plate voltage relations, one for each of several values of grid voltage. The lines for other values of grid voltage can be developed by interpolating between two adjacent lines. Each point on any line represents a possible value of RS. If a load line RL passes through this point this gives the voltage and current conditions that will result from the particular grid voltage. Three

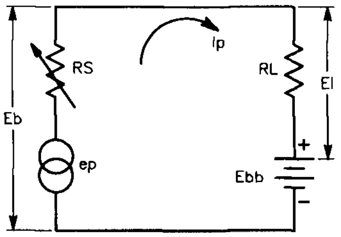


Fig 1—Equivalent circuit of an amplifier. RL is the load resistance for the plate circuit, supplied by Ebb, RS represents the internal resistance of the tube. Eb is the plate to cathode voltage, and Ip the plate current. El is the voltage across the load. Note that RS is shown variable, since it can be controlled by the grid voltage.

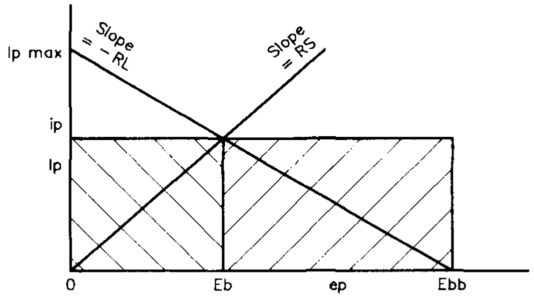


Fig 2—Graphical solution of Fig 1. On a plate voltage-plate current plot, the line Ebb-Ipmax represents the effect of the load resistor RL, and the line starting from zero the effect of RS. The intersection of these two lines gives the plate voltage and current for the particular values of RL and RS. The crosshatched area from 0 to Eb is the power dissipated in the plate, and that from Eb to Ebb the "useful" power in the load resistance. With a varying input signal the slope of RS changes, thus changing the power directed to the load.

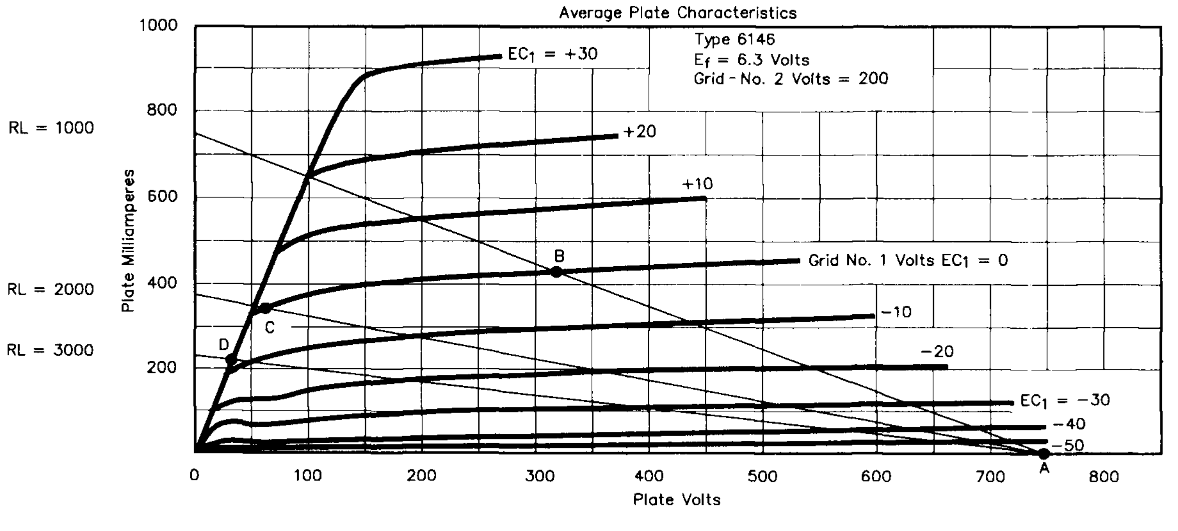


Fig 3—Graphical solution with a tube replacing RS. This extension of Fig 2 includes the tube plate voltage-plate current curves for several values of grid voltage with a 6146 tube with a screen voltage of +200. Three possible load lines are shown, each for Ebb = 750. The factors involved in selecting the best of these are discussed in the text.

possible lines of the resistance RL are shown on the plot. The problem now is to select which of these values of RL to use, and where to operate on the line.

Elements of Amplifier Design

The first factor to consider is the condition with no signal input (dc operating point). Keeping input level low is a goal, which means operating on that part of the load line close to Ebb. We will need to look at this further, but for now assume that the tube current is fully cut off with no signal input, the point A on the three load lines shown in Fig 3.

Second, when the amplifier is operating at full output, we would like the area representing the power in RL to be as large as possible, but we would also need to keep the power dissipated in the tube to be below the rated 25 W. Suppose, as is common practice, we say that the grid voltage should never go positive. The reason for this is that a positive grid will draw current, and therefore require an increase in power from preceding stages. Thus, for the three possible resistance lines shown, the maximum output points are B, C and D. Note that the choice of Point A is also involved in the maximum power condition.

The corresponding voltages, currents and powers for the intersection with the grid voltage = 0 are:

RL	Emin	Imax	Pwr in RS	Pwr in RL
(Ω)	(V)	(mA)	(W)	(W)
3000	35	240	8	172
2000	50	340	17	238
1000	320	420	134	138

We see that there is value of RL that gives maximum output. However, the power lost in RS (representing the tube) increases rapidly as the load resistance decreases. We are led to choose a value of RL based on loss in the tube as well as power output. Some additional load lines will give the best value, but we can see that it is near to 2000 Ω. The RCA tube handbook gives RL = 2000 Ω for one design, and 1850 Ω for another.

The problem of the no signal bias is illustrated in Fig 4A, showing the grid voltage-plate current curve for a particular load resistance. The intersection with the vertical axis is the same point as the intersection of the load line with the 0 V grid line, and the low part of the curve represents the part of the load line close to Ebb. For good linearity the no-signal bias should be set to the point A', rather than at A, the true cutoff value. Some current will flow with no signal, and will reduce the total current swing and the power output, but the improved linearity justifies this.

An RF input signal swings equally above and below the quiescent point. If this point is placed close to cutoff, as shown in Fig 4A, the plate current and therefore the voltage across the load does not reproduce the input signal, but becomes a series of pulses, only the upper half of the input signal, as shown in Fig 4B. While this action is useful in some situations, it is not good here. For one thing, power output at the frequency of the input signal is low. Also, the distorted signal is rich in harmonics, which cannot be radiated legally. We must do something to get the other half of the input signal, thus reducing the distortion.

There are two basic ways we can do this. One is to place another amplifier with inverted connections to handle the second half of the signal. This has come to be called the push-pull amplifier. It gives the maximum output possible with two tubes, and has the very desirable property of being essentially frequency independent, so it is a wide-band amplifier. This is not always good, however. Any stray signals present are also amplified, including those produced by input distortions. Then, too, if component switching is required, the extra connections mean more complexity.

There's another way to handle distortion: We can use the fact that a tuned circuit will respond to an input pulse with a series of voltage cycles, a process called ringing. Because of the resistance loss, each successive cycle becomes smaller. However, with the series of pulses from the amplifier, process is restarted each new pulse.

We can estimate the difference between the two halves of the cycle by considering a quantity called "decrement," which is defined as:

$$\text{Dec} = 2.3 \times \log(E2 / E1)$$

where the Es are the peak-voltage values of two successive cycles. The value of the filled-in half-cycle peak is approximately half this.

It will be recalled that a difference between the upper (positive) and lower (negative) parts of a signal can be evaluated as the sum of a pure sine wave plus a series of harmonics. Thus we can control the amount of harmonic generation in an amplifier by controlling the decrement. Here it is easier to

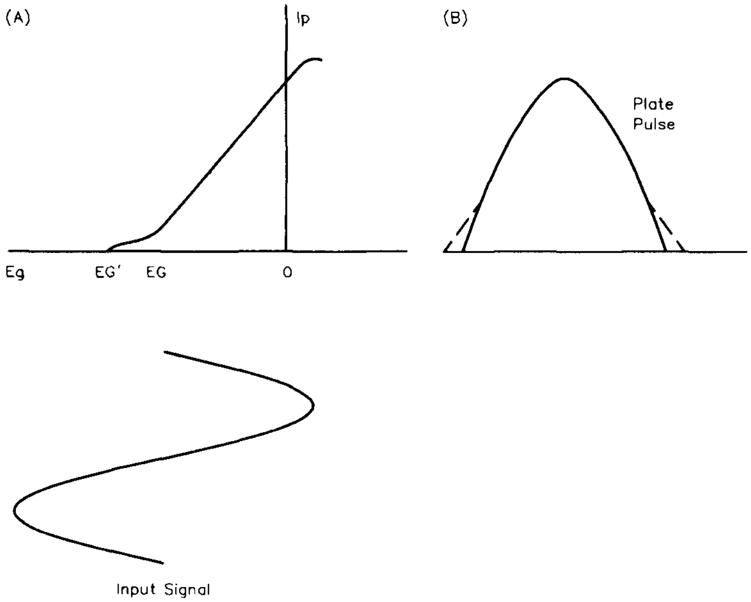


Fig 4—Grid voltage-plate current in a tube circuit. Another plot of tube performance, for a particular value of load resistance, RL. The tube is normally biased to point EG with no input signal for reasons of linearity, rather than to EG', the cutoff bias. The input signal is assumed to swing to Eg = 0, which gives good linearity and a simple method of detecting excessive drive. The plate current is a series of pulses, essentially the more positive half of the input signal. Useful in wave shaping, these pulses would reduce possible power output and greatly increase harmonic distortion if a compensating step were not taken.

use a more modern expression, developed from:

$\text{Decrement} = \pi \times R / (\omega \times L)$
$Q = \omega \times L / R$
$Q = \pi / \text{Dec}$

where $\omega = 2\pi f$, and f = frequency. So we control the amount of harmonic generation by selecting the circuit Q, or the ratio of tuned circuit reactance to load resistance.

While high Q values reduce the harmonic generation, there is a penalty. The inductance must be high: The coil loss increases as inductance increases, so output decreases. Experience has shown that good operation is secured with Qs in the range of 10 to 15 in vacuum-tube amplifiers. Here is one of the main differences between tubes and transistor type solid state devices: the latter are low voltage, high current elements, which leads to accepting lower values of Q, even though the higher harmonic distortion must be corrected by filters.

The addition of the tuned circuit requires a change in the equivalent circuit of Fig 1, to that of Fig 5. A generator representing the tuned circuit has been added, and a switch that automatically disconnects the tube element from the tuned circuit and the load whenever the tube current reaches zero. The graphical solution plot changes to that of Fig 6. The main difference is that the voltage across the tube increases during the nonconductive part of the cycle, very nearly by the difference between the voltage at maximum current and that with no input. There is a change due to the presence of resistance in the tuned circuit, but this is small and can be neglected.

For a load of 2000 Ω and a Q of 12, with the no-signal bias of –50 V, the amplifier performance is, taking the first four items from Fig 5:

Ebmin = 50 V
Ibmax = 340 mA
Ebmax = 730 V
Ibmin = 20 mA

Using the procedure of the RCA handbook,

Ibe = Ibmax / 3 = 113 mA (peak value)
Ibave = Ibe / 1.4 = 81 mA
PEPin = Eb × Ibe = 85 W
PEPout = Ibmax / 4 × (Eb – Ebmin) = 60 W
Average dissipation = (0.5 × PEPin) – PEPout / 2 = 29.5 W

Where the last is calculated for a sine wave signal, and is a little above the rated dissipation. This means that single tone tune-up should not go to full output, although a two-tone tune-up signal is okay. Because of its low average energy, a normal SSB signal stays well within ratings. Commercial amplifiers often take advantage of this fact by increasing the plate voltage above recommended value, as a way to get more output while otherwise being within ratings.

Amplifiers During Tune-up

Before going into the matter of obtaining the 2000 Ω load, let us look at some other aspects of vacuum-tube amplifiers. Suppose we have built the amplifier, and decide to test it with no load connected. This is equivalent to having a very high load impedance presented to the tube, this being the load due to stray losses in the circuit. Such a load line would make a small angle to the zero current line, so the input current and the dissipation should be small. When we set the excitation reasonably high, however, and turn on the amplifier power, the plate current goes to a high value, typically 150 mA or so. If we do nothing, a red spot that gets brighter and brighter

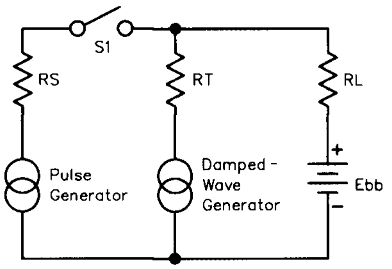


Fig 5—Revised equivalent circuit with plate tuned circuit. RS is now shown with a pulse generator and a disconnect switch that opens when the voltage exceeds Ebb. The added damped wave generator and associated loss RT represents the flywheel effect of the tuned circuit. When the switch is closed, the RS circuit is supplying power to both the tuned circuit and the load, RL. When it is open, only the tuned circuit supplies power to the load.

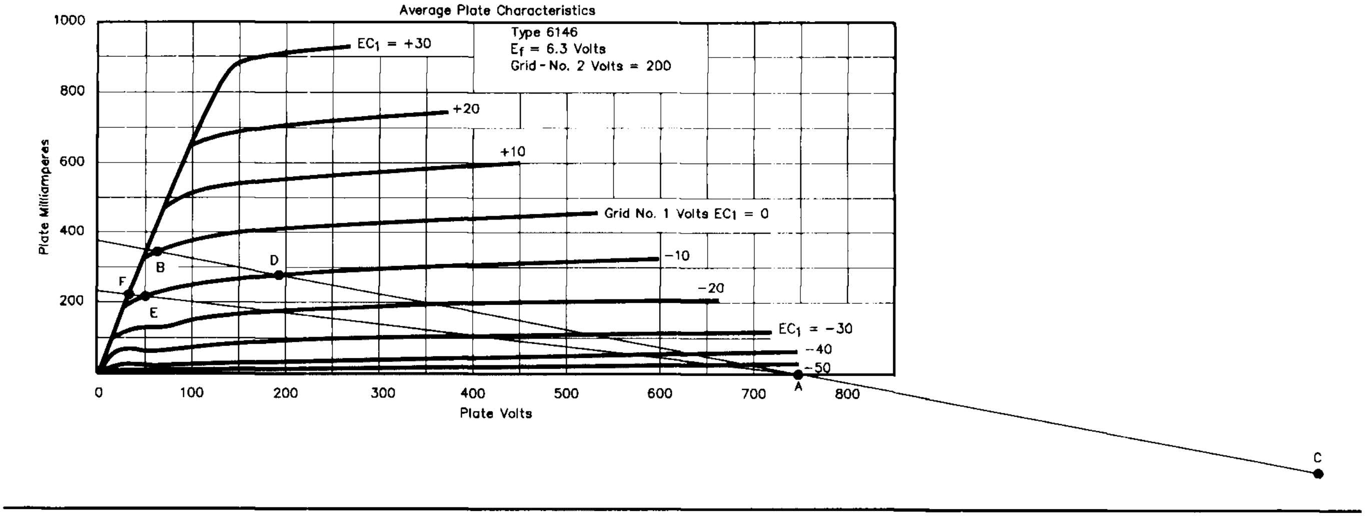


Fig 6—Revised graphical solution. Because of the tuned circuit, the voltage across the tube can be greater than the supply voltage Ebb. Assuming the grid is swinging between cutoff at A and 0 V at B, the plate voltage reaches C ideally, where AC equals AB. Practically, the voltage at C will be a little lower, due to the residual voltage at cutoff, and by the voltage drop in the tuned-circuit loss resistance. See the text for the added load line AEF.

appears on the plate of the tube. If we adjust the tuning control of the amplifier as we should, we will find a point at which the plate current drops markedly. The exact value at the low point depends on the bias setting and on tuned circuit loss, and will typically be 10 to 20 mA for the 6146.

Why this performance? To see this we have to remember that the impedance of a tuned circuit off resonance is a reactance. This is rather small when well away from resonance. The impedance magnitude increases as resonance is approached, and becomes more resistive, finally being a pure resistance at exact resonance. Thus, the load diagram of Fig 6 is not a good representation of conditions during initial tune-up.

To study this we need to look at a matter that is either partly or totally neglected in textbooks. This is the performance of an amplifier with a reactive load, and with a combination of resistance and reactance.

Elements of Graphical Solution of Reactive Loads

In a circuit with only reactance present, the voltage and current are exactly 90° out of phase (Reference 3). A way of saying this is that the point of maximum positive current corresponds to the point of minimum positive voltage. Also, the point of minimum positive current is that of maximum positive voltage. These two points, plus either of the ones relating to negative values define the curve relating voltage and current. This turns out to be an ellipse, as shown at A in Fig 7. The intersections with the vertical axis are the maximum currents, and with the horizontal axis, the voltages. The curve becomes a circle when the voltage and current maxima are numerically equal at the plot scale.

Also shown on the curve is a straight line passing through zero, representing a resistance load line. If both are present in a parallel connection, the total current is the sum of the currents through the two branches at any instant. This has the form of a canted ellipse, as shown in Fig 7B. (These curves are drawn with negative values to the right, as needed for tube load-line usage.)

In either the original or the canted ellipse, the voltage-current relation is given by successive positions around the figure. The direction of movement is determined by the sign of reactance, ie, by whether it is inductive or capacitive, or below or above resonant frequency; as drawn here, the movement is clockwise if the circuit is inductive.

Amplifiers with Reactive Loads

Let us return to the no-load tune-up condition. Since there is no resistive component, the ellipse of Fig 7A is the one to use. For the maximum current, assume that tune-up excitation has been set to −20 V at the peak, and that the reactance is low enough to allow the current given by the tube curve at this bias to flow. For illustration let us assume a lower plate voltage, and that the tube is fully cut off, so another point is at 600 V and 0 mA. For voltage, the ellipse must be centered on the no-signal point and must be scaled to a size that just touches the zero-grid-volt line. Fig 8 is the resulting plot, combining the tube characteristics and the reactive load line.

We can now see why the initial current is high. The tube is drawing current half the time, with a peak current of 450 mA. The dissipation can be determined by selecting equally timed points around the ellipse, reading the voltage and current at each and averaging the results. However, an easier method is possible here, since the product of voltage and current is an area on the chart. The total area of an ellipse is π × A × B, where A and B are the semimajor and semiminor axes, which are also the voltage swing and the maximum current. The dissipation is half of this since there is no current for area below the axis, or π × 600 × 0.450 / 2, or over 170 W. No wonder the tube plate turns red.

This high startup current has such a great damaging effect that most single-package transmitters, or "exciters," have a special position marked **TUNE-UP**. Usually this decreases the screen voltage, perhaps to half of its operating value, or lower. The tube handbook includes a nomogram to calculate the resulting current. For example, if the screen voltage is reduced from the 200 V used for the figures to 100 V for tune-up, the current at any curve point is reduced to 34% of the original value. For this example this is about 60 W, still above rated plate dissipation, but safer. Further, most exciters allow setting the excitation

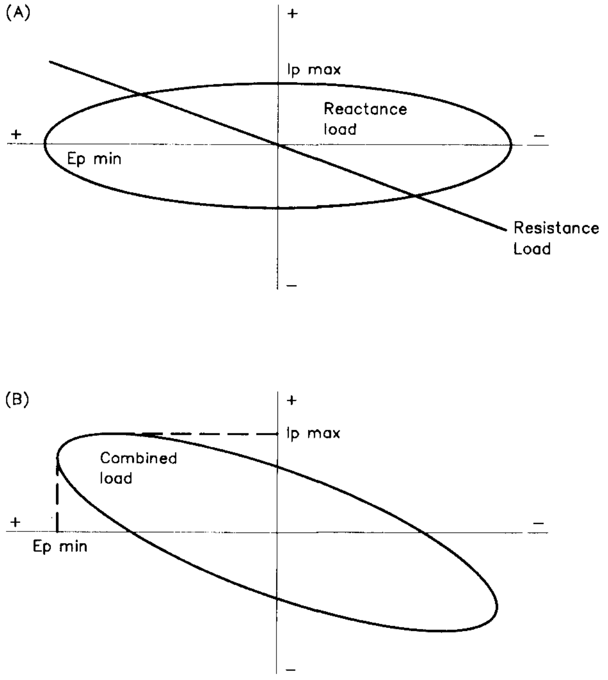


Fig 7—Load lines with reactance. At A, an ellipse representing the load presented by a pure reactance, as described in the text. A straight line representing a resistive load is also shown. When both are present and connected in parallel, the combined load line becomes a tilted ellipse, as shown at B. The major axis is the magnitude of the total impedance. The vertical axis corresponds to Ebb. Note that these curves are drawn with negative voltage to the right, as needed for use with the tube curve.

level by a **DRIVE** control, and starting with this at a lower level will give a safe condition for the tuning process. We will return to this matter later.

The elimination of this overload condition requires bringing the plate tuned circuit to resonance, which should be done quickly. This amounts to increasing the magnitude of the tuned circuit impedance, and to making it more resistive. This also amounts to shrinking the vertical or current axis of the ellipse, which becomes a straight line at resonance. For the artificial conditions used in Fig 8 this line would be the horizontal axis. In a real design, the line would pass through the no-signal point, at a small angle. The angle is determined by the loss of the tuned circuit.

This process can be observed with an oscilloscope. Three probes are useful. One can be a current probe, placed to respond to the tube current, or it can be the voltage across a small resistor in the tube cathode. The second would also be a current probe, to show the current in the tuned circuit. The last would be a voltage probe, for the voltage across the tube. The tube current will show marked distortion, the tuned circuit current a good sine wave, and the voltage a distorted sine wave that becomes a clean sine wave at and near resonance.

The next step is to increase the load on the tube circuit. In most cases this will also introduce some reactance. This means that the load line is no longer a straight line, but that it becomes an ellipse, thin with just a little reactive current, or fat if much reactance is present. Fig 9 shows the load line that would result if the total load impedance is equal to the design load resistance, but with reactance present, and for nearly full excitation (grid swinging to essentially 0 V).

As before, the power can be determined by reading the voltage and current for equal times around the ellipses. However, there is no real reason to do this: inspection of the figure shows that the area between the upper part of the ellipse and the horizontal axis is much greater than the area between the straight line representing a pure resistance of the same magnitude as the ellipse impedance. Since these areas are related to the input power, the ellipse condition results in higher power input. At the same time, since the resistive component of the ellipse condition is less than for a pure resistance equal to the impedance magnitude, the power output of the circuit with reactance is also lower. Both are undesirable. For proper operation, we need to secure a purely resistive load. Fortunately, this is so easy that we don't have to think about it. We automatically get the resistive condition when we tune to a dip in plate current. One caution applies here: A false minimum can occur if the plate tuning capacitor is too small, sometimes a problem on 80 and 160 meters.

Strictly speaking, this "tune to the dip" statement applies only to amplifiers with reasonably high Q tuned circuit operation, Qs of 10 or more. There are a few special designs that use low Q, and need the special tune-up method described in Terman (Reference 4). Transistor circuits are normally low Q, and so have a different tune process. For exciters with tube finals, or for tube linear amplifiers, the injunction of the title, "Just Tune for the Dip" applies.

There is, however, a point to watch. Suppose we have decided to protect the tube by tuning up with reduced excitation, say by setting the drive to reach –10 V rather than 0 V. As shown in Fig 6, this means that our intent is to drop back on the design load line from point B to point D. However, if we tune for maximum output at this excitation,

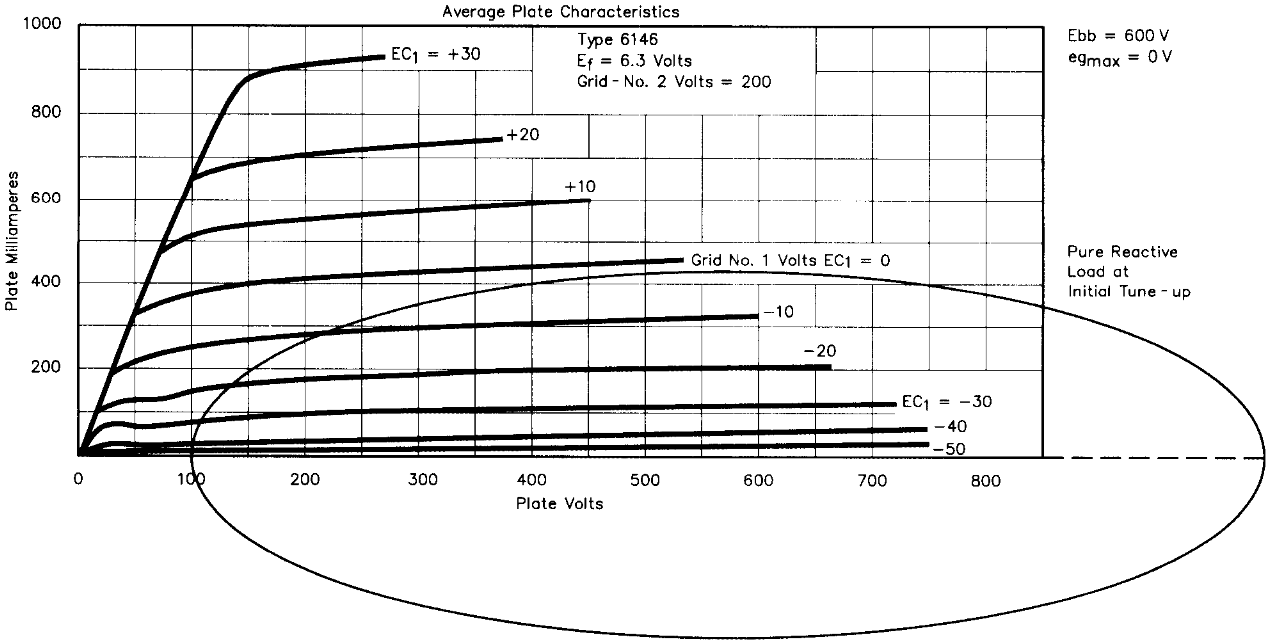


Fig 8—Tube with pure reactance load. The reactive ellipse of Fig 6A imposed on the tube curve representing the condition at start of tune-up, with negligible load and tuned (tank) circuit tuning well away from resonance. Even with no load, plate current flows for half of the time. Its magnitude is set by the grid excitation level, here assumed to cause a peak current of 450 mA. As the tank circuit is tuned to resonance, the vertical axis of the ellipse shrinks toward a straight line.

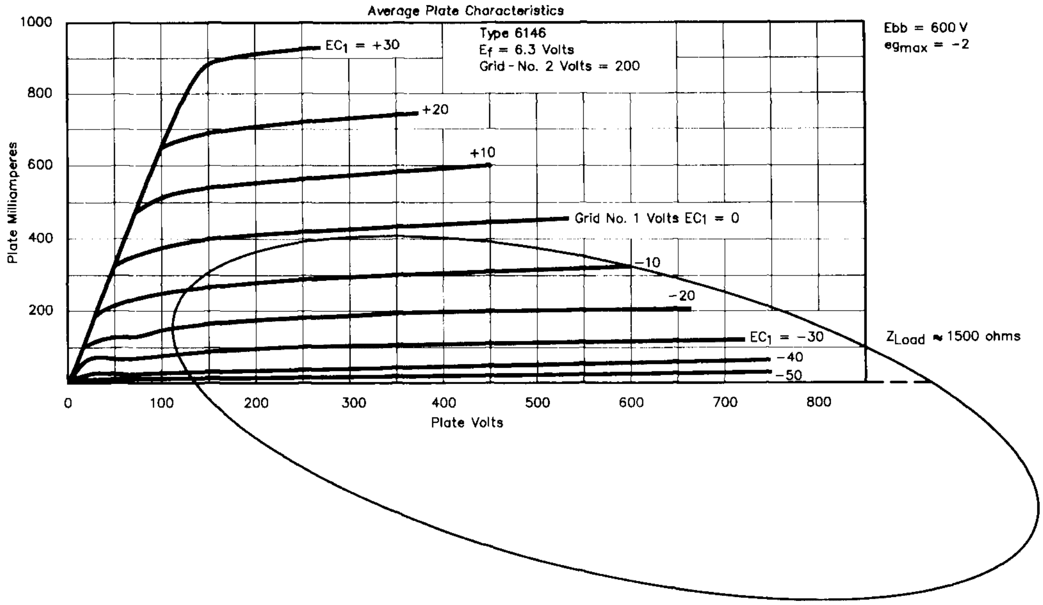


Fig 9—Tube with reactive near-normal load. As the loading is increased, the ellipse becomes canted, moving toward the proper load line BAC of Fig 6. When there is still residual reactance present, the power output is decreased, the plate current and tube heating increases, and signal distortion is present. For proper operation, the reactance must be removed. This automatically occurs when the TUNE control is adjusted for minimum plate current (the dip), provided this is a true minimum and is not caused by insufficient tuning range. However, this does not guarantee that the LOAD control is at the point giving proper operation. See text.

we set conditions to give the lower (or higher resistance) load line, tuning for the point E. If we now increase the drive to reach the 0 V line, the output has only increased by the difference between points D and E. The tuning condition we have chosen reduces power output capability, and introduces distortion if operation to full normal output is attempted. The distortion occurs when the excitation exceeds the value allowing linear operation: the amplifier output no longer increases as the excitation increases, the process is called flat-topping. Again, this condition is easily observed with an oscilloscope, especially if a trapezoidal presentation is used, as described in the handbooks.

Some trials on graphs and tests on amplifiers will show that reducing the screen voltage during tune-up will almost completely avoid the change from tune to operating conditions. If the plate-current change were exactly linear with screen-voltage change, the conditions would remain the same. However, the change is not exactly linear, so the change between tune-up and operate must be watched. Of course, this step is not available when triodes are used as amplifiers. Unfortunately, the high-power amplifiers called linears do not include the special tune-up feature. It is up to the operator to set the proper conditions.

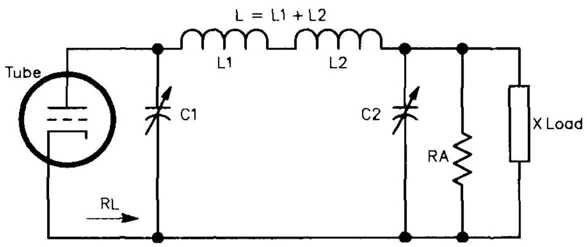


Fig 10—A π network used as a tank circuit for an amplifier. The circuit is drawn as two L networks back-to-back. This approach is used as design steps; see text. In practical amplifiers, L1 and L2 are combined into a single coil. Note that the load may include both resistance and reactance, but that the circuit must present a pure resistance to the tube for proper operation. One result is that practical circuits cannot match all possible values of load. A separate antenna tuner increases the range of allowable loads and has the further advantage of reducing harmonic radiation.

Getting the Load Resistance

We can summarize the above by noting that we have three requirements to meet:

- Eliminate all reactance from the load to the tube.
- Get the proper load resistance at the tube.
- Change the terminal conditions to meet the above.

Collectively, these steps are called "Matching the Load."

At one time, we used a pair of coupled tuned circuits for this. One was an LC combination with relatively great L (ie, high Q), in the plate circuit, to get the resonance or purely resistive condition. The second was a low-Q circuit, series or parallel connected to the load, which served to remove any reactance from the terminals, also making a pure resistance. The coupling between these circuits was adjustable, the method of converting the terminal resistance to the load resistance.

The modern practice is to replace this assembly by two capacitors and an inductance, the π network shown in Fig 10. Very nearly, the left capacitor and part of the inductor can be regarded as the high-Q plate circuit. This capacitor is often marked **TUNE**. The right capacitor and the rest of the inductance form the terminal low-Q tuned circuit. This capacitor is often marked **LOAD**. The resistance changing ratio is set during design, essentially by setting the point at which "part of the inductor" occurs.

References 2 and 5 give equations to calculate the values of L and C. Reference 6 gives a chart for solution. All are based on the left-half, right-half concept of the paragraph above. The ARRL relations, with some change in nomenclature, are;

$$XC1 = \frac{RL}{Q}$$

$$XC2 = \frac{RA}{\sqrt{\frac{RA}{RL}\left(1+Q^2\right) - 1}}$$

$$XL = RA\left(\frac{Q + \left(\frac{RL}{XC2}\right)}{Q^2 + 1}\right)$$

where Q is usually between 10 and 15 for vacuum tubes, and 3 to 5 for transistors. Given the Q, XC1 is determined only by RL, and XC2 by the ratio, RL / RA.

A look at the middle equation shows that RL must be less than the terminal resistance RA multiplied by $1 + Q^2$. For a 50 Ω load and a Q of 10, RL must be less than 5050 Ω. Thus the 2000 Ω load of the 6146 is okay, but a high-voltage, low-current tube of equal power might require an unusually high-Q circuit, or some other matching connection. Similarly, a very low value of RA may run into problems. For the 1500 to 2000 Ω of the 6146, the lower limit of RA is not much less than 20 Ω. Matching will not be possible if the load resistance is very low.

These are not the only limitations. The equations assume that there is no reactance at the terminals, that RA is a pure resistance. If reactance is present, it must be absorbed in C2. If the terminal reactance is capacitive, C2 must be adjusted to decrease its capacitive reactance, the opposite if inductive. In real amplifiers, there is a maximum and a minimum value of C2, so there are both upper and lower limits to the amount of reactance that can be present.

Let us look at some values, using RL = 2000 Ω and design Q = 10, and the chart of Reference 6. For 50 Ω terminal resistance, or design conditions, the value of C1 is –200 Ω, C2 is –25 Ω, L1 is +200 Ω and L2 is +40 Ω, for a total inductive reactance of 240 Ω.

If the load is reactive, say $50 + j50$ Ω, the equivalent load is 100 Ω in parallel with +100 Ω, inductive. Assuming sufficient range in capacitor variation, the +100 reactive ohms can be absorbed in C2, so we need to convert the residual 100 Ω resistive to the needed 2000 Ω for the plate. C1 and L1 remain at –200 and +200 Ω. However, the needed C2 is changed to –50 Ω, and L2 to 40 Ω, for a total inductance of 240 Ω.

If we have made the inductor variable, the value of L is no problem. If we have followed the usual practice and made the inductor a fixed value for each band, we cannot secure the required conditions. We must vary something else, and the usual thing we can vary is the design Q. A few trials show that a Q just less than 11 will transform the 100 Ω resistance to 2000 Ω at the tube. This is not a great change. It amounts to slightly more loss in the circuit and slightly sharper tuning.

There is another approach. If we allow the load resistance RL to vary, we can hold Q constant. For example, If RL is 2100 Ω, XC1 and XL1 are 210 Ω, XC2 is 52 Ω and XL2 is 42 Ω, for a total XL of 218 Ω. Assuming that the design load of 2000 Ω is the one giving maximum output, the change to a 2100 Ω load will reduce the power output. (There may be second-order effects: In particular, the allowable drive may decrease.)

We could try some other larger values, which would show that matching is possible if two conditions are met. One is that the variable capacitor C2 has sufficient range, the other is that the net sign of XC2 is not changed (the terminal reactance in parallel with the reactance of C2 must always be negative). With reasonable deviations from the 50 Ω load condition the change in Q is of little or no importance.

The situation is quite different for very low values of RA. For example, with RA = 20 Ω, a Q of 10 would require XC2 to be infinite, ie, C2 to be zero. Practically, no match at all would be possible until Q is allowed to reach nearly 15. At RA = 5 Ω, the required Q would be 25 or more.

The point of this is that the conventional π tank circuit either requires a 50 Ω resistive load, or there will be a departure from design conditions. For small deviations in load, the departure is not important, but there are limits to keeping good operation. Further, it is necessary to keep the tube load resistive. If it is not, there will be loss in output, increased tube heating and increased distortion. Changing the frequency always introduces a reactance change. If the antenna is close to objects, the SWR may vary as the antenna is turned, introducing both resistance and reactance change. Long transmissions can change component values because of heating. Because of these occurrences, good practice requires monitoring of amplifier operating conditions.

Keeping Things Right

There does not seem to be an automatic way to avoid flat-topping conditions, whether due to improper load, or to improper tune-up. In particular, if an amplifier is tuned for maximum output with reduced drive, it must be retuned for true maximum output, or the drive and output must be set lower than normal to avoid distortion.

The best way to do the monitoring and retuning is with a CRT station monitor. With a trapezoid presentation, adjust the **LOAD** and **DRIVE** controls for straight edges of the trapezoid. The Collins 30L1 did include a special circuit to show correct tuning: This was a pair of voltmeters giving the ratio of excitation to output; a deviation from the design value indicates improper tuning. Lacking one of these, the best method is to tune for maximum output at full excitation using a two-tone audio test signal, a method covered in the handbooks. (Do the tuning into a dummy load to keep from causing unnecessary interference.)

Common practice in both exciter and amplifier design is to provide an auxiliary indicator and feedback circuit to compensate for reduced capability conditions and improper adjustment. This works by measuring the grid current that occurs if the grid goes positive (overdriven) and decreasing the gain of the driver stages if this occurs. We call the process Automatic Level

Control (ALC). For good performance, watch the ALC indication—if this is provided by the amplifier design—and keep it small. Ideally, it should just flicker on voice peaks. If output is below normal at this drive level, the tune-up needs to be redone.

To monitor for an improper load, look at the reflected power at the amplifier output. With an antenna tuner, it can be set to zero, the condition for a load of $50 + j0$. Cross-needle and twin-meter indicators are nice for this. For equivalent performance, connect an external 0 to 100 µA meter (through a variable resistor) to the reverse contact of the station reflectometer-power meter **FORWARD-REVERSE** switch. Even a small meter will do; the goal is to see that it always stays near zero.

Recommendation: Regard ALC as insurance, and always use a CRT station monitor (trapezoid display). Always keep the ALC within the man-ufacturer's recommendations.

"Tune to the Dip" is important in getting proper amplifier operation. Monitoring helps to keep it proper.

References

1. *RCA Transmitting Tubes up to 4 kW Plate Input*, Radio Corporation of America, Harrison, NJ, 1962 (Reprints are often available at hamfests).
2. *ARRL Handbook for the Radio Amateur*, ARRL, Newington CT, 65th edition, or later. ARRL publications are available from your local ARRL dealer or directly from ARRL. Mail orders to Pub Sales Dept, ARRL, 225 Main St, Newington, CT 06111-1494. You can call us toll-free at tel 888-277-5289; fax your order to 860-594-0303; or send e-mail to **pubsales@arrl.org**. Check out the full ARRL publications line on the World Wide Web at **http://www.arrl.org / catalog**.
3. *Radiotron Designers Handbook*, F. Langford Smith, Editor, 4th edition, RCA, Harrison, NJ, 1953 (primarily written for receivers and audio amplifiers, but a useful compendium).
4. *Radio Engineers' Handbook*, F. E. Terman, McGraw Hill, New York, 1943.
5. *Radio Handbook*, W. I. Orr, 23rd edition, Sams, Indianapolis, 1987.
6. *Radio Engineering Handbook*, Keith Henney, Editor, McGraw Hill, NY, 1959.

1997 Index

July 1997 to December 1997 QEX
(issues 180 through 185)
All dates are 1997 unless otherwise indicated. *

*We did not publish issues from January through June 1997.