Contents

6.1 Circuit Simulation Overview

- 6.1.1 Hobby versus Professional Circuit Simulation Tools
- 6.1.2 The Design Cycle
- 6.1.3 Schematic Capture and Simulation Tools

6.2 Simulation Basics

- 6.2.1 *SPICE* — History
- 6.2.2 Conventions
- 6.2.3 Types of Simulations
- 6.2.4 RF-Fluent Simulators

6.3 Limitations of Simulation at RF

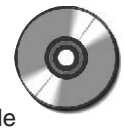
- 6.3.1 *SPICE*-based Simulators
- 6.3.2 Harmonic Balance Simulators
- 6.3.3 Contrasts in Results
- 6.3.4 RF Simulation Tools

6.4 CAD for PCB Design

- 6.4.1 Overview of the PCB Design Process
- 6.4.2 Types of PCB Design Software
- 6.4.3 Schematic Capture
- 6.4.4 PCB Characteristics
- 6.4.5 PCB Design Elements
- 6.4.6 PCB Layout
- 6.4.7 Preparation for Fabrication

6.5 References and Bibliography

Chapter 6 — CD-ROM Content



- "The Dangers of Simple Usage of Microwave Software" by Ulrich Rohde, N1UL and Hans Hartnagel
- "Using Simulation at RF" by Ulrich Rohde, N1UL
- "Mathematical Stability Problems in Modern Non-Linear Simulation Programs" by Ulrich Rohde, N1UL and Rucha Lakhe
- Examples of Circuit Simulation by David Newkirk, W9VES

Chapter 6

Computer-Aided Circuit Design

This chapter provides an overview of computer-aided design (CAD) for electronic design and PCB layout. These tools enable the hobbyist to harness some of the circuit simulation power employed by professional electronic and RF engineers in the product and system design cycle.

Material originally contributed by David Newkirk, W9VES, addresses generic circuit simulation tools. Dr Ulrich Rohde, N1UL, surveys issues associated with linear and nonlinear RF simulation and contributes three extensive papers on the accompanying CD-ROM. Dale Grover, KD8KYZ, presents a comprehensive introduction to the use of PCB design and layout software.

The purpose of this chapter is not to provide detailed instructions for using any particular software package, but to explain the basic operations, limitations and vocabulary for CAD packages commonly used by amateurs. This chapter covers circuit simulators and PCB design tools. Software to aid design and analysis in specialized areas, for example filter design, switchmode power supplies, transmission lines, and RF power amplifiers, is covered in other chapters. Schematic capture is addressed as an element of both simulation and PCB design.

6.1 Circuit Simulation Overview

Mathematics can predict and analyze the action of electromagnetic signals and the radio-electronic circuitry we build to produce and process them. Program an electronic computer — which, at base, is a generic math machine — to do the radio/electronics math in practically applicable ways, and you're ready to do computer-aided design (CAD) of radio and electronic circuits. (Program a computer to do radio-electronics math *in real time*, and you're ready to replace radio-electronics hardware with software, as this book's **DSP and Software Radio Design** chapter describes.)

6.1.1 Hobby versus Professional Circuit Simulation Tools

Professional grade circuit simulation software exists to facilitate the construction of tightly packaged, highly integrated, no-tweaking-required modern electronics/RF products. These products work predictably well even when reproduced by automated processes in large quantities — quantities that may, with sufficient marketing success and buyer uptake, exceed millions of units.

Manufacturers of specialized *electronic design automation* (*EDA*) software serve the engineering needs of this industry. Through comprehensive CAD suites, one may proceed from graphical component level circuit and/or IC design (*schematic capture*), through simulation of circuit and IC behavior (often using a variant of the simulator called *SPICE*, but increasingly with non-*SPICE* simulators more fluent in issues of RF and electromagnetic design), through design of PC board and IC masks suitable for driving validation, testing and production. Comprehensive EDA CAD reduces costs and speeds time to market with the help of features that can automatically modify circuits to achieve specific performance goals (*optimization*); predict effects of component tolerances and temperature on circuit behavior across large populations of copies (*Monte Carlo analysis*); and generate bills of materials (BOMs) suitable for driving purchasing and procurement at every step of the way.

Demonstration or student versions are available for some EDA CAD products at no or low cost (see **Table 6.1**), and a subset of these are especially useful for hobby purposes. Although these *demoware* tools come to us with a large-scale-production pedigree, they are greatly (and strategically) feature-limited. Only a relatively few components, often representing only a subset of available component models, may be used per simulation. Monte Carlo analysis, optimization, BOM generation and similar enhancements are usually unavailable. The licenses for these packages often limits the use of the software to noncommercial applications. Demoware is intended to drive software purchasing decisions and serve as college level learning aids — learning aids in college study toward becoming electronics/RF professionals who will each day work with the unlimited, full versions of the demoware. *Freeware* versions of simulation and layout software with considerable power are also available and may also have some restrictions. In either case, read the licensing agreement to become aware of any obligations on your part.

The radio hobbyist's circuit simulation needs are much simpler than the professional. Most of us will build only one copy of a given design — a copy that may be lovingly tweaked and refined to our hearts' content far beyond "good enough." Many of us may build as much with the intent of learning about and exploring the behavior of circuits as achieving practical results

Table 6.1
Some Sources of Freeware/Demoware Electronic CAD Software

Source	*Address*	*Resource*
Ansoft (now Ansys)	**www.ansys.com**	*Ansoft Designer SV 2* (schematic, linear RF simulator, planar electromagnetic simulator, layout [PCB] design), more. No longer available but older copies of the program may be available
Cadence Design Systems	**www.cadence.com**	*OrCAD 16* (schematic, *SPICE* simulator, layout [PCB] design)
CadSoft	**www.cadsoft.de** **www.cadsoftusa.com**	*EAGLE* schematic and layout design
gEDA	**www.gpleda.org**	GPLed suite of electronic design automation tools
Kicad	**http://iut-tice.ujf-grenoble.fr/kicad**	GPLed full-function schematic and layout design
Linear Technology Corp	**www.linear.com**	*LTSpice* (schematic, *SPICE* simulator enhanced for power-system design)

with them. A demoware circuit simulator can accelerate such self-driven exploration and education in electronics and RF.

6.1.2 The Design Cycle

The components we use to build real circuits always operate to the full extent of their actual properties, regardless of our relative ignorance of what those properties may be and how and why they operate the way they do. *No real-world component operates ideally*. So it is that we may set out to design, build and publish an amplifier circuit only to discover at power-up that we have instead built a persistent oscillator. Or if the prototype is an oscillator, that for 3 out of every 100 subsequent reader-builders the circuit does not oscillate at all!

The electrical and electronic components available in circuit simulators are only mathematical *models* of real world components — because every modeled behavior of a component is only an approximation relative to the behaviors of its real world counterpart. Simulated component characteristics and behaviors approach those of real world components only as closely as science may allow and only as closely as the model's description of the real world behavior.

Were this chapter a textbook, or part of a textbook, on computer-aided circuit design, we might begin exploring simulation by reviewing the basics of what electronic circuits are and do, following this with a discussion of what computerized circuit simulation is and how it works. An excursion into the arcane world of active device modeling — the construction and workings of mathematical electrical equivalents to the transistors, diodes and integrated circuits that await us at our favorite electronics suppliers and in our junk boxes — might follow. Finally, we might systematically proceed through a series of simulation examples from the basic to the more complex, progressively building our store of understood, trustworthy and applicable-to-future-work concepts as we go.

But this is a chapter in a handbook, not a textbook, and following a sequence of abstract basics to concrete practice very likely does not reflect the process most of us have followed, and follow yet, in learning and using what we know about electronics and radio. More realistically, our approach is more like this: We find ourselves in need of a solution to a problem, identify one, and attempt to apply it. If it works, we move on, likely having learned little if we have not had to troubleshoot. If the solution does *not* work, we may merely abandon it and seek another, or — better, if we are open to learning — we may instead seek to understand why, with the happily revised aim of understanding what we need to understand to make the solution work. Even if we must ultimately abandon the solution as unworkable in favor of another, we do not consider our time wasted because we have further accelerated our deepening intuition by taking the initiative to understand why.

There is no smell of burning resistor or overheated transistor in a simulation. The placement of components on the screen has no effect on the behavior of the circuit, so a high gain stage whose input is too close to its output will never break into oscillation. The dc power sources are free of ripple and noise. These effects and many more can only be experienced (and remedies learned) by building real circuits.

Fig 6.1 shows the process by which you really learn circuit design from concept to finished project. The first step is to select a type of circuit and describe what it is supposed to do — these are the *performance requirements*. For example, an amplifier will need to achieve some level of gain over some frequency range. You may need a certain input impedance and output impedance. Armed with that information, choose a circuit and come up with a preliminary set of component values by using pencil and paper or a computer design tool. This is your *design*.

Next, *simulate* the circuit's performance. If the result satisfies your performance requirements, you can move to the next step. If not, change the circuit in some way (or change your requirements) until you are satisfied.

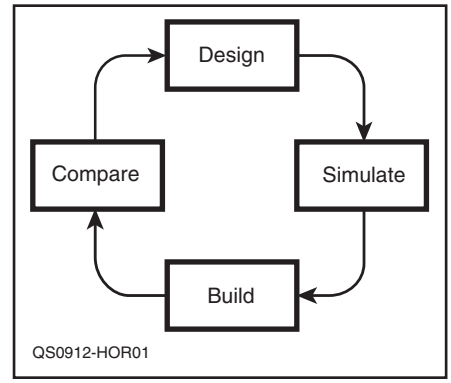


Fig 6.1 — Getting the most out of circuit simulation requires that you compare what the simulator predicts with how the actual circuit behaves.

Now *build* your design as a real world collection of components and verify that the circuit works. This is where the real fun begins as the effects of construction and actual component variation take effect. Are you done? Not yet!

To soak up every bit of design experience and know-how, go back and *compare* your actual measured performance to what the simulator predicted, particularly near the limits of the circuit's function. Look for *design sensitivities* by substituting different parts or values. If the circuit's behavior diverges from the simulator's predictions, now is the time to take a closer look. You may not be able to say exactly why differences are present, but you'll be aware they exist.

This continual *design cycle* simultaneously builds your knowledge of how real circuits and components behave, how your simulation tools work, and — most importantly — the circumstances for which the two are likely to be different.

The CD-ROM that comes with this book includes a number of design examples developed by the original author of this section, David Newkirk, W9VES. In these examples, he creates real world circuits, simulates their

behavior, compares it to actual performance, and explains why there are differences. Later in this chapter, a section by Ulrich Rohde, N1UL, discusses the differences among the different types of simulation tools used at RF — there are even differences among the different tools!

6.1.3 Schematic Capture and Simulation Tools

Almost any circuit-simulation program or electronic design automation (EDA) suite that uses schematic circuit capture can serve as a first rate schematic editor. (See Table 6.1) Although demoware component library limitations usually restrict the types of components you can use — in CAD-speak, *place* — in a design, *part count* limitations usually operate only at simulation time. Restrictions in physical size of the output drawing and layer count will likely apply to whatever layout design facilities may be available. After all, the main purpose of demoware is to let students and potential buyers taste the candy without giving away the store.

Excellent simulation-free schematic capture and layout design products exist, of course. The schematic style long standard in ARRL publications comes from the use of Autodesk *AutoCAD*, a fully professional product with a fully professional price. Long popular with radio amateurs and professionals alike is CadSoft *EAGLE*, a schematic capture and layout design product available in freeware and affordable full version forms. You can even export *EAGLE* schematics to a *SPICE* simulator and back with Beige Bag Software's *B2 Spice* (**www.beigebag.com**). The full function freeware schematic and PCB layout application *Kicad* and the EDA suite *gEDA* come to us from the open source community.

The basic drawing utility included with your computer's operating system, such as *Paint* which comes with *Windows*, can also serve as a limited do-it-yourself schematic capture tool. The ARRL also provides a limited set of schematic symbols that can be used with *PowerPoint* at the Hands-On Radio web page, **www.arrl.org/hands-on-radio**. Cutting, copying, moving and pasting components snipped from favorite graphical schematic files and adding new connections as graphical lines is enough to create a picture of the schematic but without any of the underlying tools or facilities of a true schematic capture tool.

CAD Software and Your Computer's Operating System

The *OrCAD 16.0* and *Ansoft Designer SV 2* demoware packages used in this chapter, and most other CAD products you're likely to use, are compiled to run under Microsoft *Windows*. So what if you want to run RF and electronics CAD software under *Linux* or on the Mac?

You're in luck. *EAGLE* schematic and layout software is available in native versions for *Windows*, *Linux*, and the Macintosh. The GPLed EDA application suite, *gEDA*, are primarily developed on *Linux*, but are intended to run under, or at least be portable to, *POSIX*-compliant systems in general. The GPLed schematic and layout editor *Kicad* runs natively under *Windows* and *Linux*, and has been tested under *FreeBSD* and *Solaris*. Further, the great strides made in the *Wine* translation layer (**www.winehq.com/**) allow many applications written for *Windows* to run well under the operating systems supported by *Wine*, including *Linux* and the Macintosh.

MicroSim *DesignLab 8* (a widely distributed precursor to *OrCAD 16* that can run all of the *SPICE* examples described in this chapter) and Ansoft *Serenade SV 8.5* (a precursor to *Ansoft Designer SV 2*) can be run in *Wine* under *Linux* with few artifacts and their expected schematic capture and simulation capabilities intact. Cursor handling in *OrCAD 16* installs under *Wine* readily enough, but cursor handling artifacts in its schematic editor, at least in the computers tried, seems to preclude its use under *Wine* for now. *Ansoft Designer SV 2* installs but does not properly start.

All things considered, however, especially as *Wine* and CAD applications continue to strengthen and mature, running your favorite *Windows* based applications under *Wine* is well worth a try. You may also consider purchasing an inexpensive used computer that runs one of the later versions of *Windows*, such as XP, and dedicating it to running the simulation

6.2 Simulation Basics

This section is a collection of notes and illustrations that address various important circuit simulation concepts. In this section, conventional *SPICE* notation and vocabulary are used unless specifically noted differently. Not all simulation tools use exactly the same words and phrases to label and explain their features. When in doubt, refer to the software's user manual or HELP system. There are a number of excellent textbooks about using *SPICE*-based simulators Widely used simulation tools almost always have online communities of users, all of whom were beginners once, too. Joining one of these groups is highly recommended.

Simulation tool users groups frequently develop and maintain a library of tutorials, Frequently Asked Questions (FAQ), accessory programs and utilities, even models and complex circuit models. Before you ask questions, consult the available resources, such as searchable message archives, to see if your question has been answered before — it usually has! The other users will appreciate your diligence before asking the entire group.

6.2.1 *SPICE* — History

SPICE — Simulation Program with Integrated-Circuit Emphasis — originates from the Electrical Engineering and Computer Sciences Department of the University of California at Berkeley and first appeared under its current name as *SPICE1* in 1972. "*SPICE,*" write the maintainers of the official *SPICE* homepage at **http://bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE** "is a general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analyses. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent voltage and current sources, four types of dependent sources, lossless and lossy transmission lines (two separate implementations), switches, uniform distributed RC lines, and the five most common semiconductor devices: diodes, BJTs, JFETs, MESFETs, and MOSFETs."

That *SPICE* is "general-purpose" does not mean that its usefulness is unfocused, but rather that it is well established as a circuit simulation mainstay of comprehensive power. A wide, deep *SPICE* community exists as a result of decades of its daily use, maintenance and enhancement by industrial, academic and hobby users. Many excellent commercial versions of *SPICE* exist — versions that may be improved for workhorse use in particular sub-disciplines of electronics, power and RF design.

6.2.2 Conventions

SCALE FACTORS

SPICE's use of unit suffixes — *scale factors* in *SPICE*-speak — differs from what we

are generally accustomed to seeing in electrical schematics, and that we have multiple options for specifying values numerically using integer and decimal floating point numbers. The scale factors available in *SPICE* include:

- F (femto) — 1E–15
- G (giga) — 1E9
- K (kilo) — 1E3
- M (milli) — 1E–3
- MEG (mega) — 1E6
- MIL (0.001 inch) — 25.4E–6 meter
- N (nano) — 1E–9
- P (pico) — 1E–12
- T (tera) — 1E12
- U (micro) — 1E–6

Specifying the value of a resistor as 1M would cause *SPICE* to assign it the value of 1 milliohm (0.001 Ω). This would probably create a wildly different result than expected! Specifying the value of R1 as 1MEG or 1000K would be correct alternatives. *SPICE* scale factors are case insensitive. Until you are thoroughly familiar with using circuit simulation, take the time to double-check component and parameter values. It will save you a lot of time tracking down errors caused by mistaken component values.

Notice that *SPICE* assumes unit *dimensions* — ohms, farads, henrys and so on — from component name context; in specifying resistance, we need not specify ohms. In parsing numbers for scale factors, *SPICE* detects only scale factors it knows and, having found one, ignores any additional letters that follow. This lets us make our schematics more readable by appending additional characters to values — as long as we don't confuse *SPICE* by running afoul of existing scale factors. We may therefore specify "100pF" or "2.2uF" for a capacitance rather than just "100p" or "2.2u" — a plus for schematic readability. (On the reduced readability side, however, *SPICE* requires that there be *no space* between a value and its scale factor — a limitation that stems from programming expediency and is present in many circuit-simulation programs.)

SOURCES

There are two basic types of sources used in simulation models — voltage sources and current sources — as discussed in the **Electrical Fundamentals** and **Analog Basics** chapters. Sources can be *independent* with an assigned value or characteristic that does not change, or *dependent* with a value or characteristic that depends on some other circuit value. An example of an independent source would be a fixed voltage power supply (dc) or a sinusoidal signal source (ac). An example of a dependent source is a bipolar transistor model's collector current source that has a value of βI_B.

COMPONENT MODELS

All *real* inductors, capacitors, and resistors — all real components of any type — are non-ideal in many ways. For starters, as **Fig 6.2** models for a capacitor, every real *L* also exhibits some *C* and some *R*; every real *C*, some *L* and *R*; every real *R*, some *L* and *C*. These unwanted qualities may be termed *parasitic*, like the parasitic oscillations that sometimes occur in circuits that we want to act only as amplifiers. The **RF Techniques** chapter discusses parasitics for various components.

For simulating many ham-buildable circuits that operate below 30 MHz, the effects of component parasitic R, L and C can usually be ignored unless guidance or experience suggests otherwise. In oscillator and filter circuits and modeled active devices, however, and as a circuit's frequency of operation generally increases, neglecting to account for parasitic L, C and R can result in surprising performance shortfalls in real world *and* simulated performance. In active device modeling realistic enough to accurately simulate oscillator phase noise and amplifier phase shift and their effects on modern, phase-error-sensitive data communication modes, device-equivalent models must even include *nonlinear* parasitic inductances and capacitances — Ls and Cs that vary as their associated voltages and currents change.

Figuring out what the circuit of an appropriate model should be is one thing; measuring and/or realistically calculating real world values for R_S, L_S and R_P for application in a circuit simulator is a significant challenge. How and to what degree these parasitic characteristics may cause the electrical behavior of a component to differ from the ideal depends on its role in the circuit that includes it and the frequency at which the circuit operates.

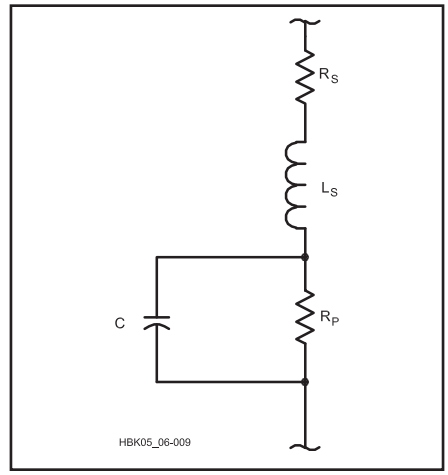


Fig 6.2 — A capacitor model that aims for improved realism at VHF and above. R_S models the net series resistance of the capacitor package; L_S, the net equivalent inductance of the structure. R_P, in parallel with the capacitance, models the effect of leakage that results in self-discharge.

Designers aiming for realism in simulating power circuits that include magnetic core inductors face the additional challenge that all real magnetic cores are nonlinear. Their magnetization versus magnetic field strength (*B-H*) characteristics exhibit hysteresis. They can and will saturate (that is, fail to increase their magnetic field strength commensurately with increasing magnetization) when overdriven. Short of saturation, the permeability of magnetic cores varies, hence changing the inductance of coils that include them, with the flow of dc through their windings. These effects can often be considered negligible in modeling ham-buildable low power circuits.

As active device operation moves from small signal — in which the signals handled by a circuit do not significantly shift the dc bias points of its active devices — to large signal — in which applied signals significantly shift active device dc bias and gain — the reality of *device self-heating* must be included in the device model. Examples: When amplitude stabilization occurs in an oscillator or gain reduction occurs in an amplifier as a result of voltage or current limiting or saturation.

While manufacturers often provide detailed models for their devices, avoid the temptation to assume that models will behave in a simulation just as an actual component will behave in a real circuit. *Absolutely every desired behavior exhibited by a simulated device must be explicitly built into the model*, mathematical element by mathematical element. A simulated device can reliably simulate real world behavior only to the extent that it has been programmed and configured to do so. A 1N4148 diode from your junk box "knows" exactly what to do when ac is applied to it, regardless of the polarity and level of the signal. Mathematically *modeling* the for-ward and reverse biased behavior of the real diode is almost like modeling two different devices. Realistically modeling the smooth transition between those modes, especially with increasing frequency, is yet another challenge.

Mathematical transistor modeling approaches the amazingly complex, especially for devices that must handle significant power at increasingly high frequencies, and especially as such devices are used in digital communication applications where phase relationships among components of the applied signal must be maintained to keep bit error rates low. The effect of nonlinear reactances — for instance, device capacitances that vary with applied signal level — must be taken into account if circuit simulation is to accurately predict oscillator phase noise and

effects of the large signal phenomenon known as *AM-to-PM conversion*, in which changes in signal amplitude cause shifts in signal phase. In effect, different aspects of device behavior require greatly different models — for instance, a dc model, a small signal ac model, and a large signal ac model. Of *SPICE*'s bipolar-junction-transistor (BJT) model, we learn from the *SPICE* web pages that "The bipolar junction transistor model in *SPICE* is an adaptation of the integral charge control model of Gummel and Poon introduced in the **RF Techniques** chapter. This modified Gummel-Poon model extends the original model to include several effects at high bias levels. The model automatically simplifies to the simpler Ebers-Moll model when certain parameters are not specified."

As an illustration of device model complexity, **Fig 6.3** shows as a schematic the BIP linear bipolar junction transistor model from *ARRL Radio Designer*, a linear circuit simulator published by ARRL in the late 1990s. This model worked well at audio and relatively low radio frequencies. The model must be even more complex for accurate results in the upper HF and higher frequency ranges.

Especially in the area of MOSFET and MESFET device modeling, and large signal device modeling in general (of critical importance to designers of RF integrated circuits [RFICs] for use at microwave frequencies), *SPICE* and RF-fluent non-*SPICE* simulators include active device models home experimenters are unlikely to use. Help in deciding on what version of a model is appropriate for your application is another reason to join your simulation tool's users group.

Most of us will go (and need go) no further into the arcanities of device modeling than using *SPICE*'s JFET model for FETs such as the 2N3819, J310, and MPF102, and *SPICE*'s BJT model for bipolar transistors such as the 2N3904. *OrCAD 16* includes preconfigured 1N914, 1N4148, 2N2222, 2N2907A, 2N3819, 2N3904 and 2N3906 devices, among many others, and these will be sufficient for many a ham radio simulation session. Getting the hang of the limitations and quirks of these models may well provide challenge enough for years of modeling exploration.

To get parameters for other devices, especially RF devices and specialty components like transformers, we must search the Internet in general and device manufacturer websites in particular to find the data we need. The manufacturer sites listed in **Table 6.2** will get you started. Because the use of simulation and *SPICE* models is so widespread, manufacturers routinely make those models available at no cost. The usual place to find them is via the device's online data sheet through a hyperlink or in a special model library on the manufacturer's website — enter "models" into the site's search function if you can't find the models.

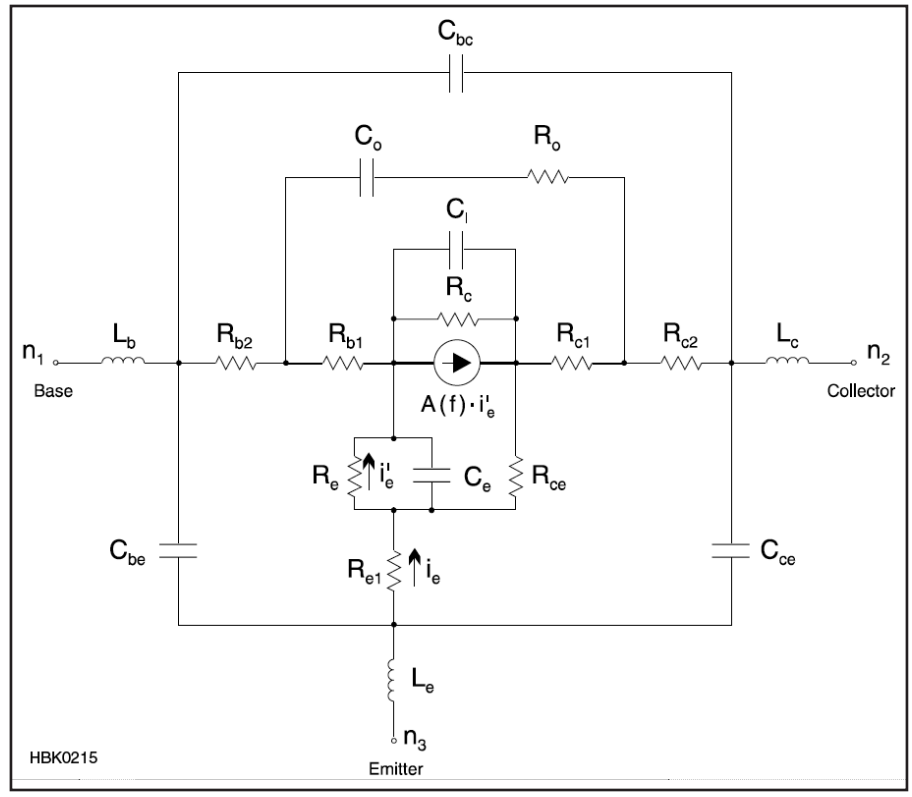


Fig 6.3 — The linear BJT model, BIP, from *ARRL Radio Designer*, a now-discontinued circuit simulation product published by ARRL in the late 1990s.

Table 6.2
Device Parameter Sources

Source	*Address*	*Resource*
Cadence Design Systems	**www.cadence.com/products/orcad/pages/downloads.aspx#cd**	*OrCAD*-ready libraries of manufacturer-supplied device models
California Eastern Laboratories	**www.cel.com**	Data and models NEC RF transistors
Duncan's Amp Pages	**www.duncanamps.com/spice.html**	*SPICE* models for vacuum tubes
Infineon	**www.infineon.com**	Data and models for Siemens devices
Fairchild Semiconductor	**www.fairchildsemi.com**	Device data and *SPICE* models
National Semiconductor and Texas Instruments	**www.ti.com**	Data and *SPICE* models for National and Texas Instruments op amps
NXP	**www.nxp.com/models/index.html**	Data and models for Philips devices
Freescale	**www.freescale.com**	Data and models for Freescale (previously Motorola) devices
On Semiconductor	**www.onsemi.com**	Data and models for On Semiconductor (previously Motorola) devices

Models for any particular device are generally available through a link on the online version of the device's data sheet. Internet searches for the device's part number and "model" usually works, as well.

```
              2N2222
              NPN
          IS   14.340000E-15
          BF  255.9
          NF    1
         VAF   74.03
         IKF     .2847
         ISE   14.340000E-15
          NE    1.307
          BR    6.092
          NR    1
          RB   10
          RC    1
         CJE   22.010000E-12
         MJE     .377
         CJC    7.306000E-12
         MJC     .3416
          TF  411.100000E-12
         XTF    3
         VTF    1.7
         ITF     .6
          TR   46.910000E-09
         XTB    1.5
          CN    2.42
           D     .87

.model Q2N2222 npn (IS=2.48E-13 VAF=73.9 BF=400 IKF=0.1962 NE=1.2069
+ ISE=3.696E-14 IKR=0.02 ISC=5.00E-09 NC=2 NR=1 BR=5 RC=0.3
CJC=7.00E-12
+ FC=0.5 MJC=0.5 VJC=0.5 CJE=1.80E-11 MJE=0.5 VJE=1 TF=4.00E-10
+ ITF=2 VTF=10 XTF=10 RE=0.4 TR=4.00E-08)
```

Fig 6.4 — A list of parameters specified for the Gummel-Poon model of a 2N2222A transistor and the condensed form usually seen in available model files.

A typical *SPICE* model is shown in **Fig 6.4**. This particular model is for the familiar 2N2222A NPN bipolar transistor. Each parameter in the vertical list (IS, BF, NF and so forth) is one element of the Gummel-Poon model for the generic transistor used by *SPICE* simulation programs. Typically, the model is provided as plain ASCII text in the compressed form at the bottom of the figure with several parameters per line. Every parameter between the opening and closing parentheses defines some element of the underlying mode. (See the *SPICE* references listed at the end of this chapter for detailed information about the syntax of *SPICE* models.)

NETLISTS

A *netlist* is a specialized table that names a circuit's components, specifies their electrical characteristics, and maps in text form the electrical interconnections among them. Uniquely numbered nodes or *nets* — in effect, specifications for each of the connections in the simulated circuit — serve as interconnects between components, with each component defined by a statement comprising one or more netlist lines.

The netlist served as the original means of circuit capture for all simulators known to the writer, including *SPICE*; *schematic capture* as we imagine it today, using graphic symbols, came later. Further reflecting *SPICE*'s pre-graphical heritage is the fact that, to this day a *SPICE* netlist may be referred to by long-time *SPICE* users as a *SPICE deck*, as in "deck of Hollerith punch cards." In *SPICE*'s early days, circuit definitions and simulation instructions (netlist statements that begin with a period [.]) were commonly conveyed to the simulation engine in punched-paper-card form. Netlists are still the means of conveying circuit topology and simulation instructions to most simulators, although they are text files today.

In the models and netlists that you will encounter, statements that must span multiple lines include *continuation characters* (+) to tell the netlist parser to join them at line breaks. Asterisk (*) or other non-alphanumeric characters denote comments — informational-to-human lines to be ignored by the simulator.

SUBCIRCUITS

Fig 6.5 illustrates the level of detail involved in more accurate device modeling typical for devices used at VHF and UHF. The device is a California Eastern Labs NE46134, a surface mount BJT intended to serve as a broadband linear amplifier at collector currents up to 100 mA and collector voltages up to 12.5 V.

This manufacturer supplied model for the NE46134 embeds an unpackaged device chip (NE46100, shown as Q1 in the figure) within a *subcircuit*. (Note the .SUBCKT label on the first non-comment line of the model which indicates that the following information defines a subcircuit.) The subcircuit includes Q1 plus the parasitic reactances contributed by the transistor package, such as CCBpkg — a collector-to-base package capacitance. The chip leads themselves are modeled as transmission lines, TB and TE.

Following the lines that define of all of the subcircuit's component values that represent the parasitic reactances, the NE46100 transistor model is provided, beginning with the line .MODEL NE46100 NPN. That information will be used to define Q1 when referenced in the subcircuit model above.

The overall model then appears to the external circuit like a regular three terminal transistor with a base, emitter and collector. In this fashion nested subcircuits can be used to create arbitrarily complex models that can be used as components by the designer.

Figs 6.6A and 6.6B show a schematic of a typical RF circuit — a 7 MHz double-tuned filter. Fig 6.6A is the usual depiction you would see in a magazine or book article about the circuit. It has the usual symbols and variable capacitors of a tunable filter. Fig 6.6B shows the circuit after schematic capture by the *OrCAD Capture CIS* tool. An ac voltage source, V1, is placed at the input with a 50 Ω series resistor, R1, to create a 50 Ω signal source impedance. The transformers TX1 and TX2 actually contain subcircuits consisting of primary and secondary inductances and a coupling element with 0.22 Ω resistors that simulate loss resistance. The paralleled fixed and variable capacitors are combined into single fixed value capacitors. A 50 Ω load, R2, has been attached to the output. Ground symbols now have a 0 nearby to indicate zero voltage.

TIME STEP

Simulators are *discrete time* devices — calculations are performed throughout the circuit and results obtained, then time is changed by a fixed amount (the *time step*) and the calculations run again. The size of the time step can be automatically chosen by the simulator (a usually reasonable value based on a default setting or some evaluation of the circuit component values) or set to specific values by the user.

While using very small time steps makes for a smooth looking output and wide frequency ranges, it makes the simulation run slower (less of an issue as computers get faster) and makes the output data files quite a bit bigger (less of an issue as hard drives get bigger). While today's computers make short

```
* FILENAME:        NE46134.CIR
* from http://www.cel.com/pdf/models/ne46134.cir
* NEC PART NUMBER: NE46134
* LAST MODIFIED:   11/97
* BIAS CONDITIONS: Vce=5V, to 12.5V,  Ic=50mA to 100mA
* FREQ RANGE:      0.1GHz TO 2.5GHz
*
*                           CCBpkg
*                     .-------||--------.
*                     |          CCB     |
*                     |      .--||---.   |          COLLECTOR
* BASE                |      |        | |
*               __Tb_ |      |   /----o-o-------o----o
*  o---------|_____|-o-LB---o--|Q1   |         |
*                     |      B   \   --         |
*                    --          |   -- CCE     --
*                    -- CBEpkg E o----'         --CCEpkg
*                     |          |              |
*                     |         LE              |
*                     `----------o--------------'
*                                |
*                                -
*                               | |  Te
*                               | |
*                                -
*                                |
*                                o EMITTER
*
*
*       CCB   = 0.03 pF       LB  = 1.2nH          Tb/Te:
*       CCE   = 0.5 pF        LE  = 1.2nH          z=60 ohms
*       CCBpkg= 0.18pF                             l=50 mils
*       CCEpkg= 0.18pF                             a=0.0001
*       CBEpkg= 0.01pF                             f=0.9GHz
*
*                    c b e
.SUBCKT NE46134/CEL 2 1 3
Q1 2 6 7 NE46100
CCB 6 2  0.03E-12
CCE 2 7  0.5E-12
LB  4 6  1.2E-9
LE  7 5  1.2E-9
TB 1 0 4 0 Z0=60 TD=9.63E-12
TE 5 0 3 0 Z0=60 TD=9.63E-12
CCBPKG 4 2 0.18E-12
CCEPKG 2 5 0.18E-12
CBEPKG 4 5 0.01E-12
.MODEL NE46100 NPN
+( IS=8.7e-16    BF=185.0         NF=0.959       VAF=30.0         IKF=0.20
+ ISE=5.70e-13   NE=1.80          BR=5.0         NR=1.0           VAR=12.4
+ IKR=0.018      ISC=1.0e-14      NC=1.95        RE=0.630         RB=6.0
+ RBM=4.0        IRB=0.004        RC=3.0         CJE=4.9e-12      VJE=0.60
+ MJE=0.450      CJC=2.50e-12     VJC=0.830      MJC=0.330        XCJC=0.20
+ CJS=0.0        VJS=0.750        MJS=0.0        FC=0.50          TF=12.9e-12
+ XTF=1.60       VTF=19.9         ITF=0.40       PTF=0.0          TR=1.70e-8
+ EG=1.11        XTB=0.0          XTI=3.0        KF=0.0           AF=1.0 )
.ENDS
*$
```

Fig 6.5 — California Eastern Laboratories model of the NE46134 linear broadband transistor. The transistor model is constructed as a subcircuit consisting of several parasitic components connected to the NE46100 transistor (Q1).

work of simulations that would have brought the previous era's mainframes to a halt, there is no need to generate vastly more data than you need for the job at hand.

If you set the time step to at least 10 times smaller than the reciprocal of the highest frequency in which you're interested, that will strike a nice balance. For example, when simulating an audio circuit, you might be interested in signals out to 100 kHz so a good time step size would be 1/10 of 1/100 kHz or about 1 µs. Similarly, if the circuit was supposed to operate up to 10 MHz, the time step should be no smaller than 10 ns. Your simulation software manual will have guidelines for that particular package as well.

6.2.3 Types of Simulations

Once the circuit has been entered into the simulator via schematic capture and all error checks have been completed, simulation can begin in earnest. There are several ways in which the circuit can be simulated, corresponding to the various tests that you would run on a real circuit on the workbench. The following paragraphs are only definitions — for a complete description of these simulations and how to perform them, the reader is directed to the References section of this chapter and reminded that the software provider and associated users groups will also have a lot of information and background material.

DC OPERATING POINT OR BIAS POINT

The first step in simulation is usually to run a *DC Operating Point* or *DC Bias Point*

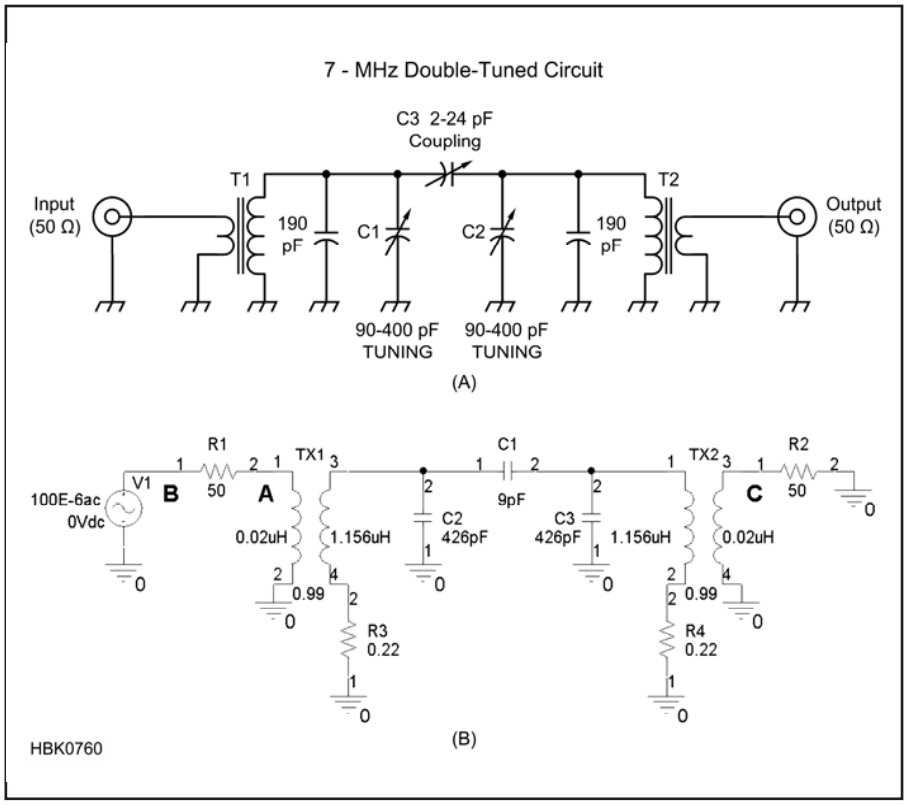


Fig 6.6 — A is a 7 MHz double-tuned filter as drawn for real world builders. B shows the filter schematic as it is typically drawn for construction and service.

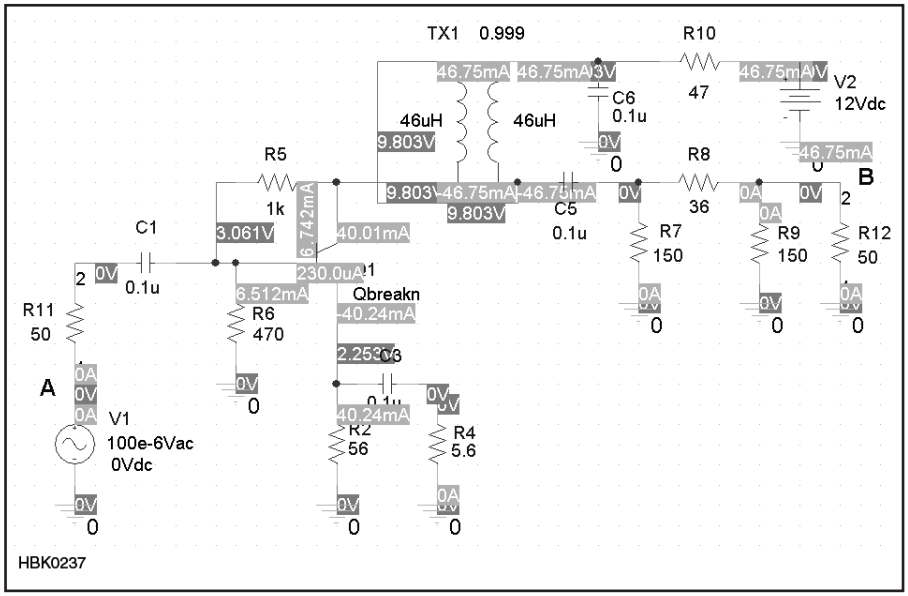


Fig 6.7 — A dc operating point simulation allows the designer to verify that the circuit biasing is realistic for all devices and that all voltages and currents are as expected.

analysis. (Remember that terminology usually changes from program to program.) In this analysis, the simulator "applies power" to the circuit and calculates the dc voltages and currents for all circuit components.

Fig 6.7 shows the results of a typical DC Operating Point analysis for an amplifier circuit. The labels obscure the circuit elements a bit in this print image but you can see a transistor (Qbreakn) in the middle, an input voltage source at the lower left (V1), the power supply voltage source at the upper right (V2), and a variety of components making up the circuit. The darker labels show voltages such as 3.061 V (transistor base voltage) and 46.75 mA (the power supply output current). The information is generally available as a text file, as well. Most simulators also allow you to flag certain points in the circuit for which voltage and current (ac and/or dc) are displayed on screen at all times.

It is good practice, particularly for beginning and occasional modelers, to verify that the dc operating point is as expected for all circuit components before moving ahead to the more interesting ac waveform and spectrum displays. Many hours have been wasted troubleshooting a circuit's simulated ac performance when the problem is really that the dc operating point isn't right!

TIME DOMAIN VS FREQUENCY DOMAIN

Simulators can provide two primary types of ac simulation results. The time domain output displays one or more voltages or currents on an X-Y display as traces with amplitude on the vertical axis and time on the horizontal axis. In other words, it simulates an oscilloscope type display. **Fig 6.8** shows an example time domain display of an R-C oscillator starting to oscillate. The single trace is of the oscillator's ac output voltage, showing that oscillation begins about 30 ms after the simulation begins and quickly stabilizes to a steady sine wave output.

Fig 6.9 shows a frequency domain output that looks very much like a spectrum analyzer display. This particular simulation is of the same amplifier shown in Fig 6.7 with two input signals (the large components at 7 and 10 MHz) and uses a logarithmic vertical scale to show the harmonics and intermodulation products. *SPICE*-type simulators first perform time domain ac analysis to generate a waveform then use Fast Fourier Transform techniques to calculate a frequency domain output. This technique is often insufficient to obtain highly accurate frequency domain simulations, such as noise and intermodulation performance, leading to alternatives such as harmonic balance simulations that are called *RF-fluent* since they give more accurate results at high frequencies. (See the sections on RF-Fluent Simulators and Limitations of Simulation at RF in this chapter.)

TRANSIENT

Transient simulation involves putting the circuit in a stable, known state and then applying a controlled disturbance of some sort. The response to the disturbance is then calculated and displayed over some time period in either time domain or frequency domain form. Typical inputs to transient analysis are steps (a parameter changes very quickly from one value to another), delta pulses (infinitely

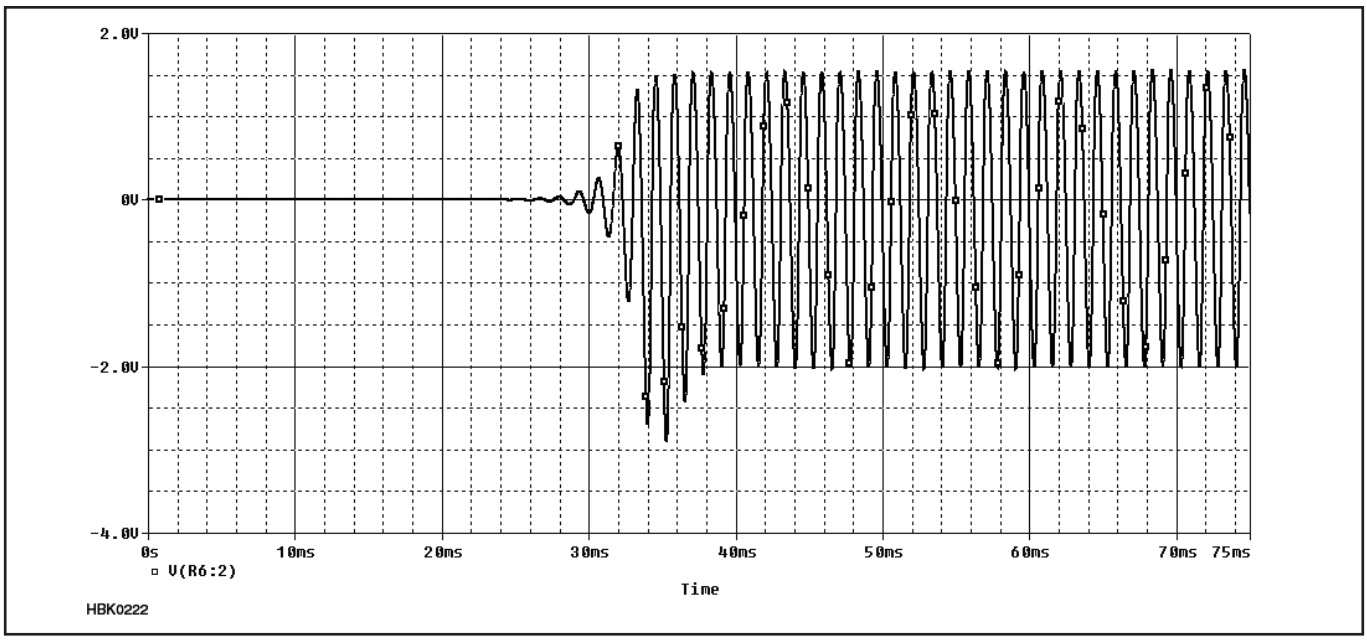


Fig 6.8 — Time domain display of an R-C oscillator startup showing voltage on the vertical axis and time on the horizontal. The display is very much like that of an oscilloscope.

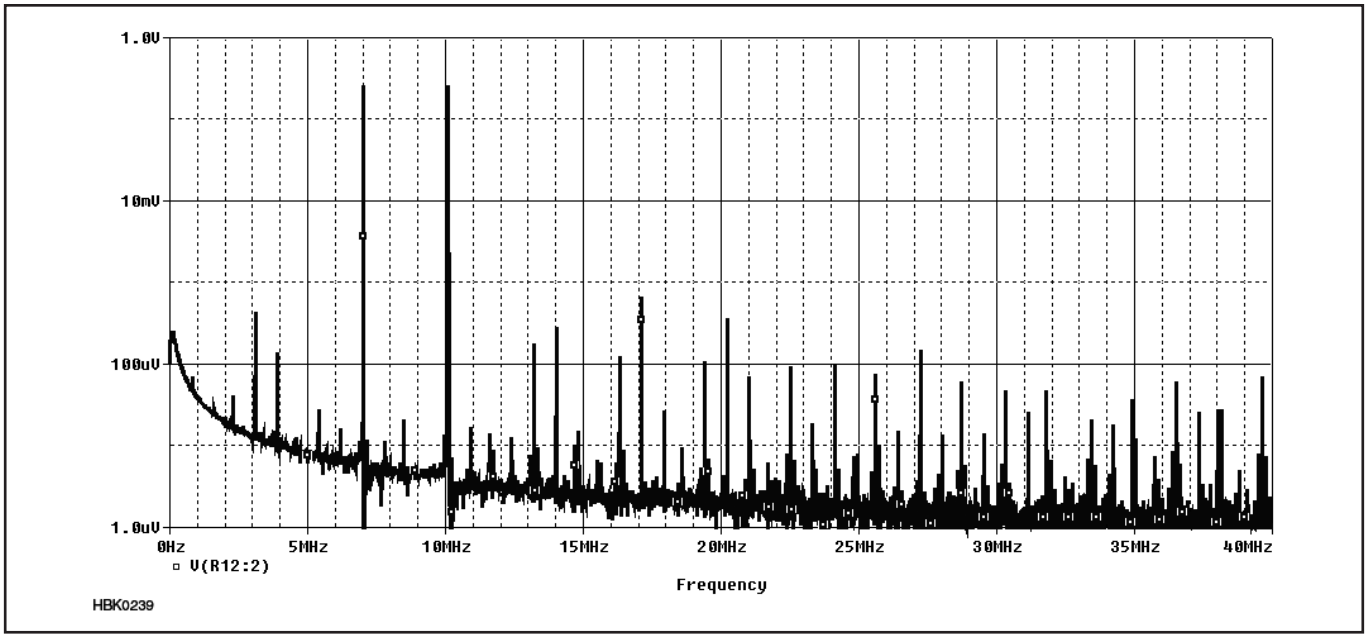


Fig 6.9 — Frequency domain display of an amplifier undergoing a two-tone test. Amplitude in dB is shown on the vertical axis and frequency on the horizontal axis. This display is very much like that of a spectrum analyzer.

narrow pulses with some finite energy), rectangular pulses, ramps, and various other selectable waveforms.

DC AND AC SWEEPS

Once a circuit is working, it is useful to characterize its performance over a range of conditions, called a *sweep*: power supply voltage, input voltage, input frequency and so on. Even temperature can be swept to see how the circuit behaves in different environments. Note that this requires the behavior of components with changing temperature to be accounted for in the component models.

One of the most common swept simulations for ham radio circuits is a frequency sweep to determine the frequency response of an amplifier or filter. **Fig 6.10** shows the insertion loss and return loss (see the **Analog Basics** chapter) for the 7 MHz filter in Fig 6.6 across a range of 6.8 to 7.4 MHz. **Fig 6.11** shows the gain of the amplifier in Fig 6.7 with the input frequency swept across a range of 1 to 100 MHz.

6.2.4 RF-Fluent Simulators

SPICE-based simulators can do wonders in many classes of circuit simulation. For RF use, however, *SPICE* has significant draw-

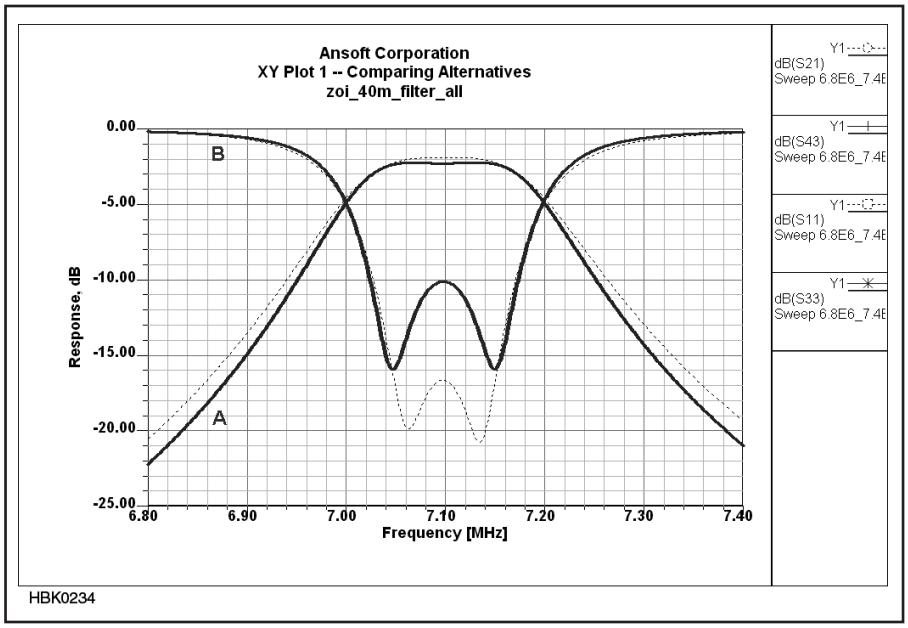


Fig 6.10 — Insertion loss (A) and return loss (B) of the 7 MHz double-tuned filter in Fig 6.6.

backs. For starters, *SPICE* is not RF-fluent in that it does not realistically model *physical* distributed circuit elements — microstrip, stripline and other distributed circuit elements based on transmission lines. It cannot directly work with network parameters (*S*, *Y*, *Z* and more — see the **Analog Basics** chapter), stability factor and group delay. It cannot simulate component *Q* attributable to skin effect. It cannot simulate noise in nonlinear circuits, including oscillator phase noise. It cannot realistically simulate intermodulation and distortion in high-dynamic-range circuits intended to operate linearly. This also means that it cannot simulate RF mixing and intermodulation with critical accuracy.

The feature-unlimited version of *Ansoft Designer* and competing RF-fluent simulation products can do these things and more excellently — but many of these features, especially those related to nonlinear simulation, are unavailable in the student/demoware versions of these packages if such versions exist.

From 2000 to 2005, the free demoware precursor of *Ansoft Designer SV 2*, Ansoft *Serenade SV 8.5*, brought limited use of nonlinear simulation tools to students and experimenters. With *Serenade SV 8.5*, you could simulate mixers, including conversion gain and noise figure of a mixer. Amplifier two-tone IMD could be simulated as in Fig 6.9. Optimization was enabled. Realistic nonlinear libraries were included for several Siemens — now Infineon — parts. You could accurately predict whether or not a circuit you hoped would oscillate would *actually* oscillate, and assuming that it would, you could accurately predict its output power and frequency. **Figs 6.12**, **6.13** and **6.14** give some examples of the power of RF-fluent simulation software, in this case, Ansoft (now ANSYS) *Serenade Designer SV 8.5*.

The harmonic balance techniques used by Ansoft's nonlinear solver — and by the nonlinear solvers at the core of competing RF-fluent CAD products, such as Agilent *Advanced Design System* (ADS) — allowed you to simulate crystal oscillators as rapidly as you can simulate lower-*Q* oscillators based on LC circuits. (In *SPICE*, getting a crystal oscillator to start may be impossible without presetting current and/or voltages in key components to nonzero values.)

Although these features are no longer available as student/demoware/freeware, if you're serious about pushing into RF CAD beyond what *SPICE* can do, see if you can find a used copy of *Serenade SV 8.5* or find a friend working with the professional tools who will let you use it occasionally for hobby applications.

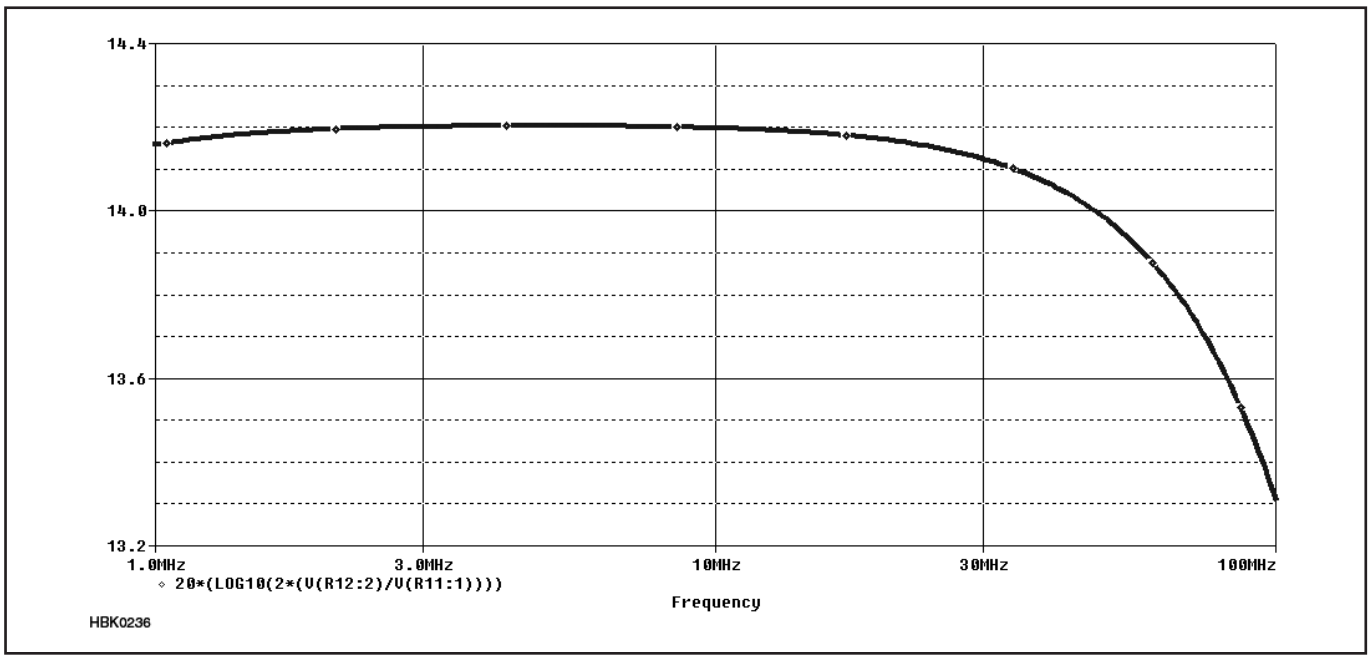


Fig 6.11 — The gain in dB of the amplifier in Fig 6.6 over a range of 1 to 100 MHz.

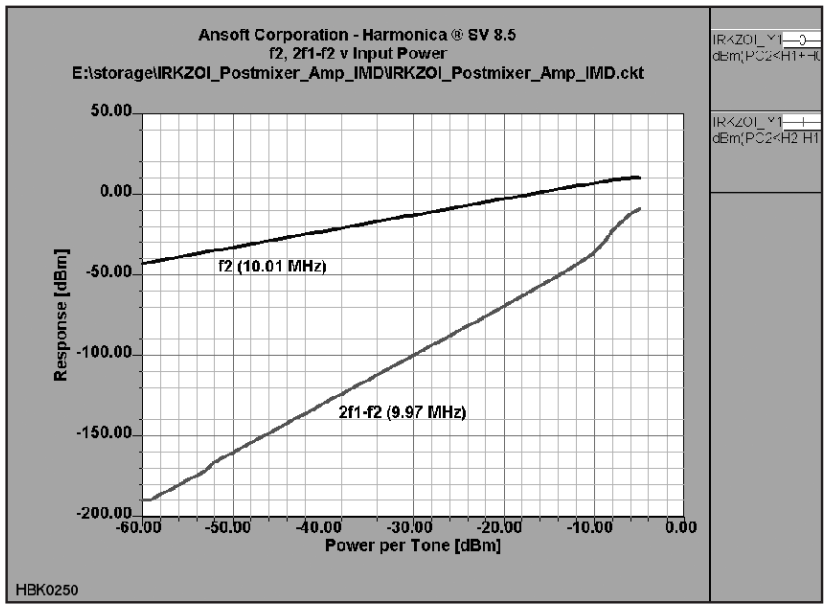


Fig 6.12 — A two-tone nonlinear simulation of third-order IMD performed by Ansoft *Serenade Designer SV 8.5*.

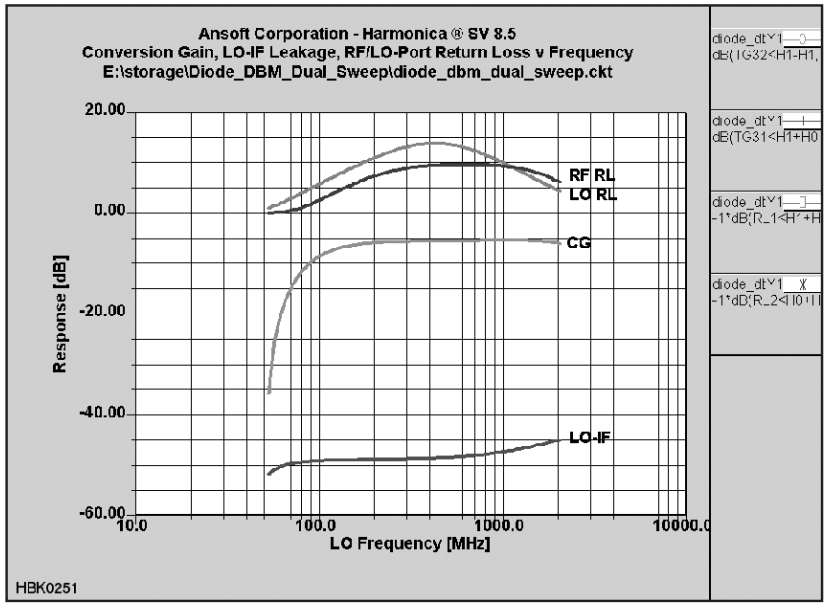


Fig 6.13 — Professional RF circuit simulators can also simulate mixing and the small signal characteristics of mixers, such as port return loss, conversion gain, and port-to-port isolation. (*Serenade SV*

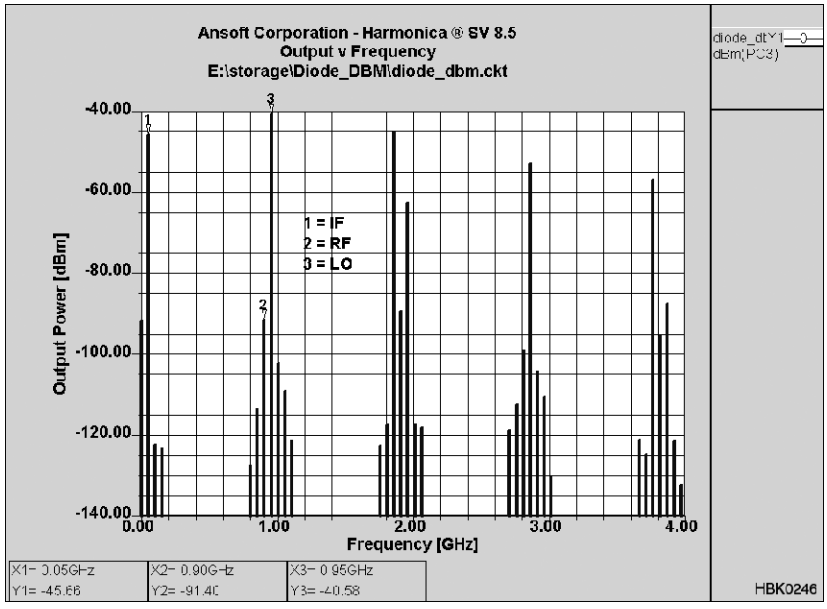


Fig 6.14 — Output spectrum of a diode-ring doubly balanced mixer as simulated by *Serenade SV 8.5*. Note the dynamic range implicit in this graph: In a simulation that includes a local oscillator (LO) signal at 7 dBm, accurate values are calculated for IMD products nearly *140 dB weaker* without encountering mathematical noise — an achievement unapproachable with *SPICE*-based simulators.

6.3 Limitations of Simulation at RF

[Experienced users of circuit simulation software are wary of using any software near or outside the boundaries of circuits and parameters for which it was intended and tested. RF simulation can present just such situations, leading to software failure and unrealistic results. Introduced and summarized in this section, several detailed papers by Dr Ulrich Rohde, N1UL, exploring simulation at RF are provided on the CD-ROM accompanying this *Handbook*. The papers are:

- "Using Simulation at RF" by Rohde, a survey of the issues of RF simulation and the techniques used in current modeling programs.
- "The Dangers of Simple Usage of Microwave Software" by Rohde and Hartnagel, a discussion of inaccuracies introduced by device parameter measurement and model characteristics.
- "Mathematical Stability Problems in Modern Non-Linear Simulations Programs" by Rohde and Lakhe, presenting various approaches to dealing with nonlinear circuit simulation.

In addition, there are many online resources to help you obtain trustworthy simulation results with a simulator designed for RF. For the interested reader with some technical background, the online paper "Introduction to RF Simulation and its Application" by Ken Kundert (**http://icslwebs.ee.ucla.edu/dejan/researchwiki/images/3/30/Rf-sim.pdf**) provides an introduction to RF simulation methods and how they account for the characteristics of RF circuits when generating common RF measurements. The website The Designer's Guide (**www.designers-guide.org**) also provides many tutorials, technical guides, models, and other resources for analog and RF simulation users.

While the precise lower bound of "RF" is ill-defined, RF effects start already at about 100 kHz. This was first noticed as self-resonance of high-Q inductors for receivers. In response, Litz wire was invented in which braided copper wires were covered with cotton and then braided again to reduce self-resonance effects.

As frequencies get higher, passive elements will show the effects of parasitic elements such as lead inductance and stray capacitance. At very high frequencies, the physical dimensions of components and their interconnections reach an appreciable fraction of the signal wavelength and their RF performance can change drastically.

RF simulators fall in the categories of *SPICE*, harmonic balance (HB) programs

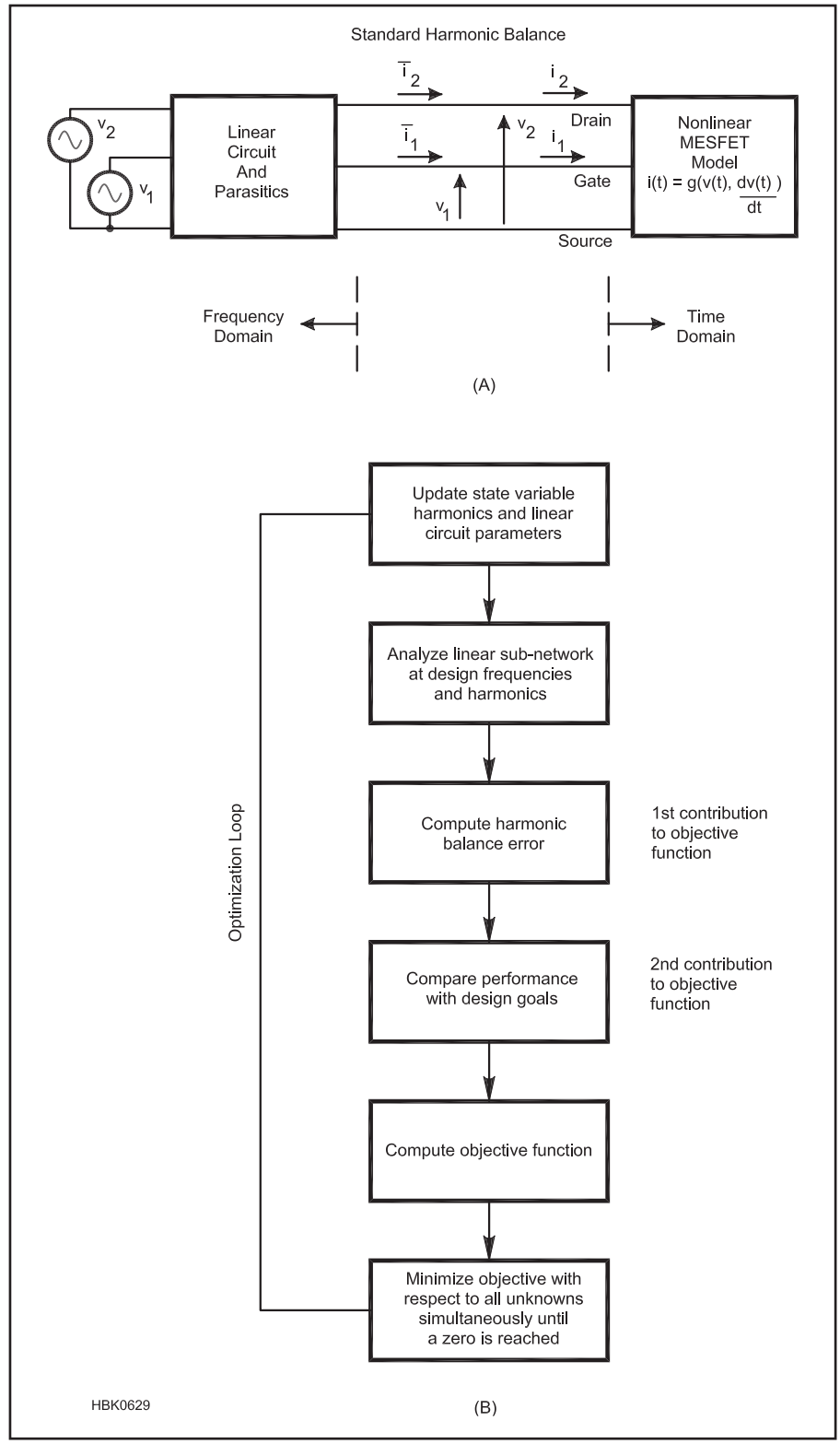


Fig 6.15 — (A) MESFET circuit partitioned into linear and nonlinear sub-circuits for harmonic balance analysis. Applied gate and drain voltages, and relevant terminal voltages and currents, are indicated. (B) Flowchart of a general purpose harmonic balance design algorithm that includes optimization.

and EM (electromagnetic) programs. The EM simulators are more exotic programs. Two types are common, the 2D (2.5) or 2-dimensional and the full 3-dimensional versions. They are used to analyze planar circuits, including vias (connections between layers) and wraparounds (top-to-ground plane connections), and solid-shapes at RF. They go far beyond the *SPICE* concept.

6.3.1 *SPICE*-based Simulators

SPICE was originally developed for low frequency and dc analysis. (Modern *SPICE* programs are based on *SPICE3* from University of California — Berkeley.) While doing dc, frequency, and time domain simulations very well, *SPICE*-based simulation has some problems. The time domain calculation uses the very complex mathematics of the Newton-Raphson solution to nonlinear equations. These methods are not always stable. All kind of adjustments to the program settings may be necessary for the calculations to converge properly. Knowledge of the specifics of different types of electronic circuits can assist the user in finding an accurate solution by specifying appropriate analysis modes, options, tolerances, and suitable model parameters. For example, oscillators require certain initializations not necessary for amplifiers and bipolar transistors may need different convergence tolerances than do MOS circuits. Generally, *SPICE* finds a solution to most circuit problems. However, because of the nonlinearity of the circuit equations and a few imperfections in the analytical device models, a solution is not always guaranteed when the circuit and its specification are otherwise correct.

The next problem at RF is that the basic *SPICE* simulator uses ideal elements and some transmission line models. As we approach higher frequencies where the lumped elements turn into distributed elements and special connecting elements become necessary, the use of the standard elements ends. To complicate matters, active elements such as diodes and transistors force the designer to more complex simulators. Adding the missing component elements leads to highly complex models and problems of convergence in which the simulator gives an error advising of a numerical problem or more likely by failing to generate a solution.

SPICE also has problems with very high-Q circuits and noise analysis. Questions of the noise figure of amplifiers or phase noise of an oscillator cannot be answered by a *SPICE*-based program accurately. Noise analysis, if not based on the noise correlation matrix approach, will not be correct if the feedback capacitance ($Im(Y_{12})$, the imaginary component of Y-parameter Y_{12}) is significant at the frequencies involved. Analysis of oscillators in *SPICE* does not give a reliable output frequency and some of the latest *SPICE* programs resort to some approximation calculations.

6.3.2 Harmonic Balance Simulators

Harmonic balance (HB) analysis is performed using a spectrum of harmonically related frequencies, similar to what you would see by measuring signals on a spectrum analyzer. The fundamental frequencies are the frequencies whose integral combinations form the spectrum of harmonic frequency components used in the analysis. On a spectrum analyzer you may see a large number of signals, even if the input to your circuit is only one or two tones. The harmonic balance analysis must truncate the number of harmonically related signals so it can be analyzed on a computer.

The modern HB programs have found better solutions for handling very large numbers of transistors (>1 million transistors) and their math solutions are much more efficient, lead-

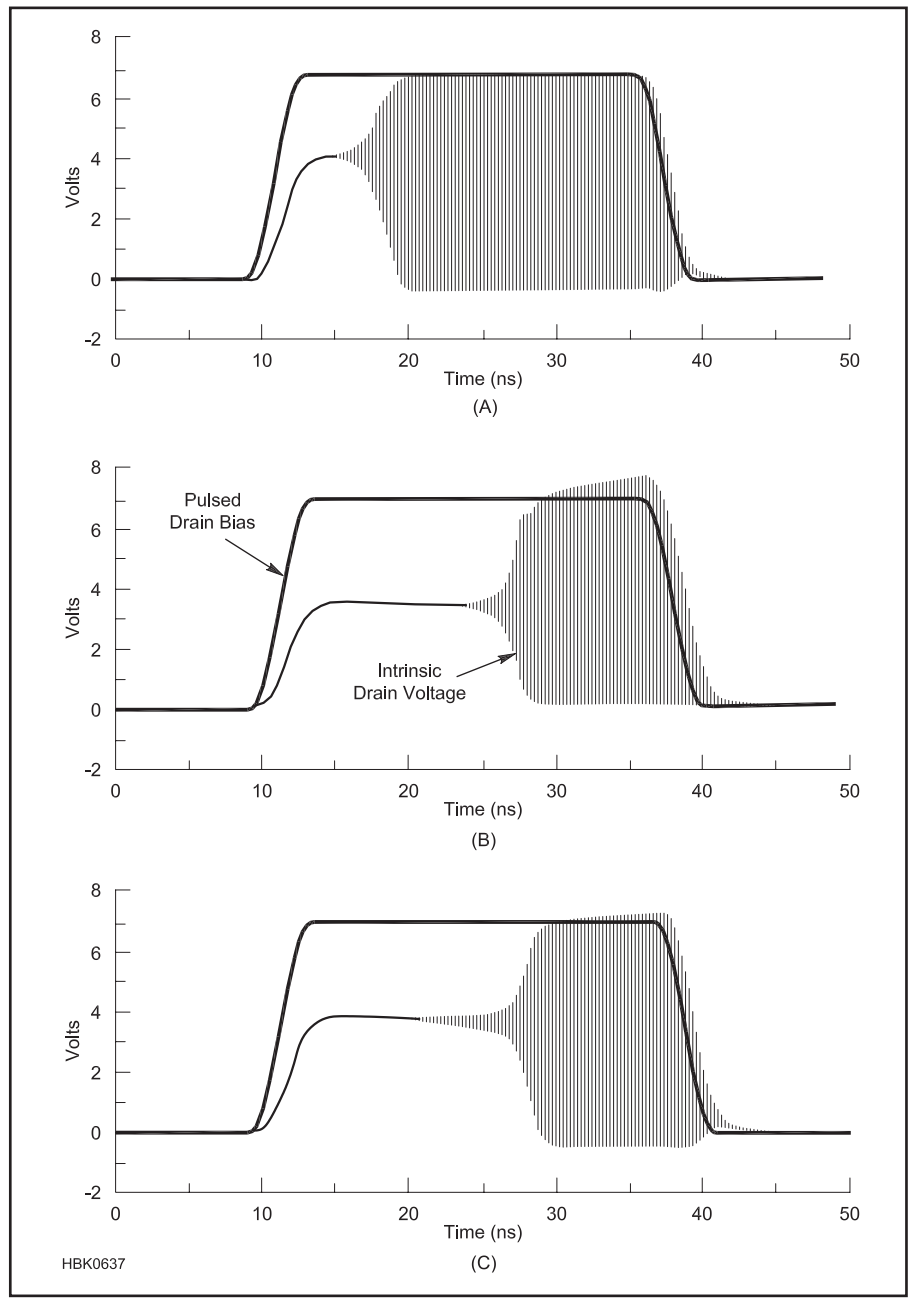


Fig 6.16 — (A) is the initial simulation of a *SPICE*-based simulator. (B) is the correct response of a pulsed microwave oscillator obtained by harmonic balance simulation using the Krylov-subspace solution. (C) is the *SPICE*-based simulation after 80 pulses of the drain voltage.

ing to major speed improvements. Memory management through the use of matrix formulations reduces the number of internal nodes and solving nonlinear equations for transient analysis are some of the key factors to this success.

HB analysis performs steady-state analysis of periodically excited circuits. The circuit to be analyzed is split into linear and nonlinear sub-circuits. The linear sub-circuit is analyzed in the frequency domain by using distributed models. In particular, this enables straightforward intermodulation calculations and mixer analysis. The nonlinear sub-circuit is calculated in the time domain by using nonlinear models derived directly from device physics. This allows a more intuitive and logical circuit representation.

Fig 6.15A diagrams the harmonic balance approach for a MESFET amplifier. Fig 6.15B charts a general purpose nonlinear design algorithm that includes optimization. Modern analysis tools that must provide accurate phase noise calculation should be based on the principle of harmonic balance.

Analysis parameters such as Number of Harmonics specify the truncation and the set of fundamental frequencies used in the analysis. The fundamental frequencies are typically not the lowest frequencies (except in the single-tone case) nor must they be the frequencies of the excitation sources. They simply define the base frequencies upon which the complete analysis spectrum is built.

6.3.3 Contrasts in Results

The following time domain analysis is a good example of differences between *SPICE* and harmonic balance simulation. A microwave oscillator is keyed on and off and a transient analysis is performed. When using the standard *SPICE* based on *SPICE3*, the initial calculation shows an incorrect response after one iteration as seen in **Fig 6.16A**. It takes about 80 pulses (80th period of the pulsed drain voltage) until the simulation attains the correct answer (Fig 6.16C) of the Krylov-subspace-based harmonic balance in Fig 6.16B.

The frequencies involved need not be in the GHz range. Oscillators, in particular, can be very difficult to analyze at any frequency as shown by simulations of a low-MHz phase shift oscillator and a 10 MHz Colpitts oscillator in the referenced papers.

Validating the harmonic balance approach, **Fig 6.17** shows a BJT microwave oscillator entered into the schematic capture module of a commercially available HB simulator (Ansoft *Serenade 8.0*); **Fig 6.18** compares this oscillator's simulated phase noise to measured data. HB analysis gives similarly accurate results for mixers.

6.3.4 RF Simulation Tools

PSPICE: This popular version of *SPICE*, available from OrCAD (now Cadence Design Systems, **www.cadence.com**) runs under the PC and Macintosh platforms. An evaluation version, which can handle small circuits with up to 10 transistors, is freely

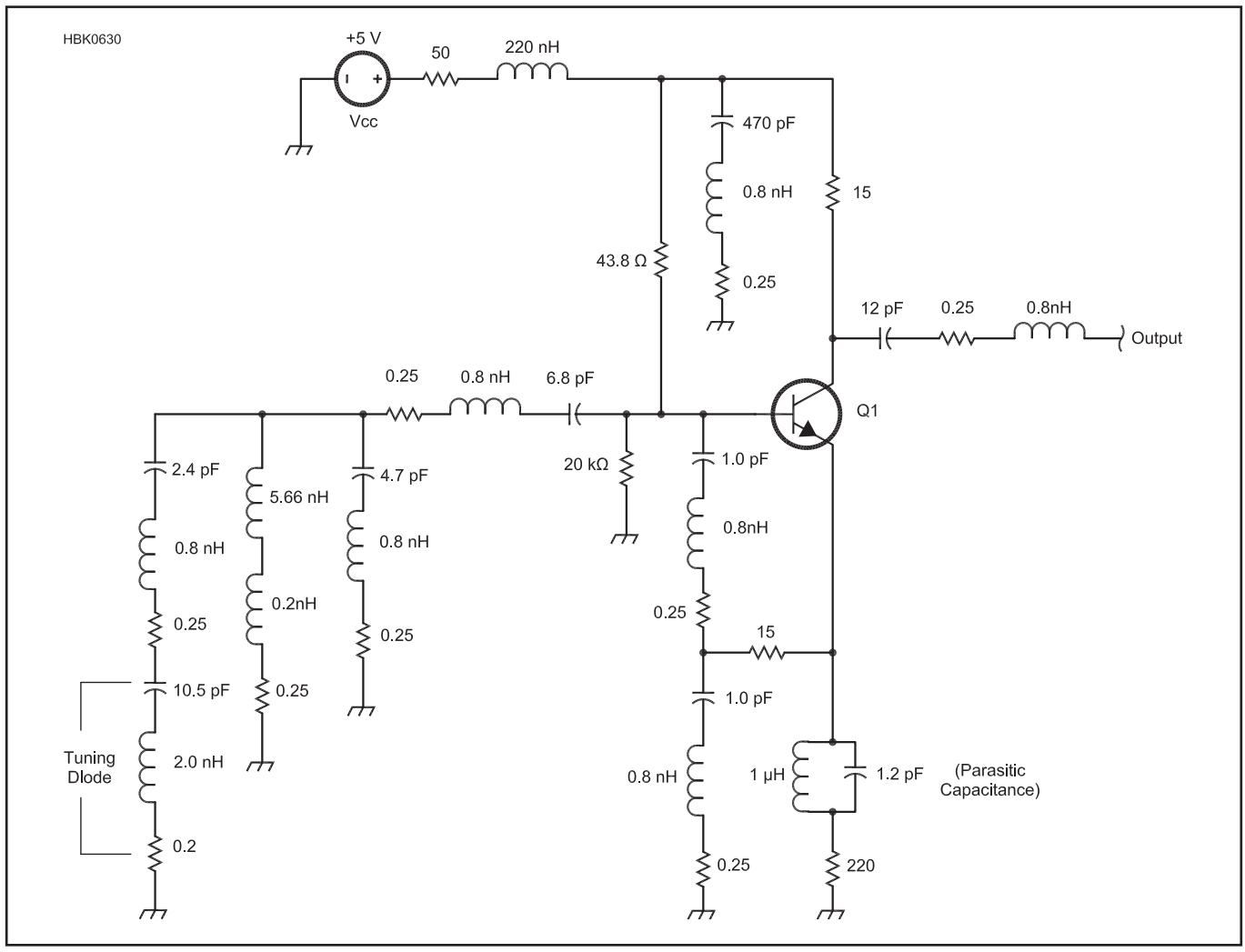


Fig 6.17 — Colpitts oscillator for 800 MHz with lumped elements modeled by their real values.

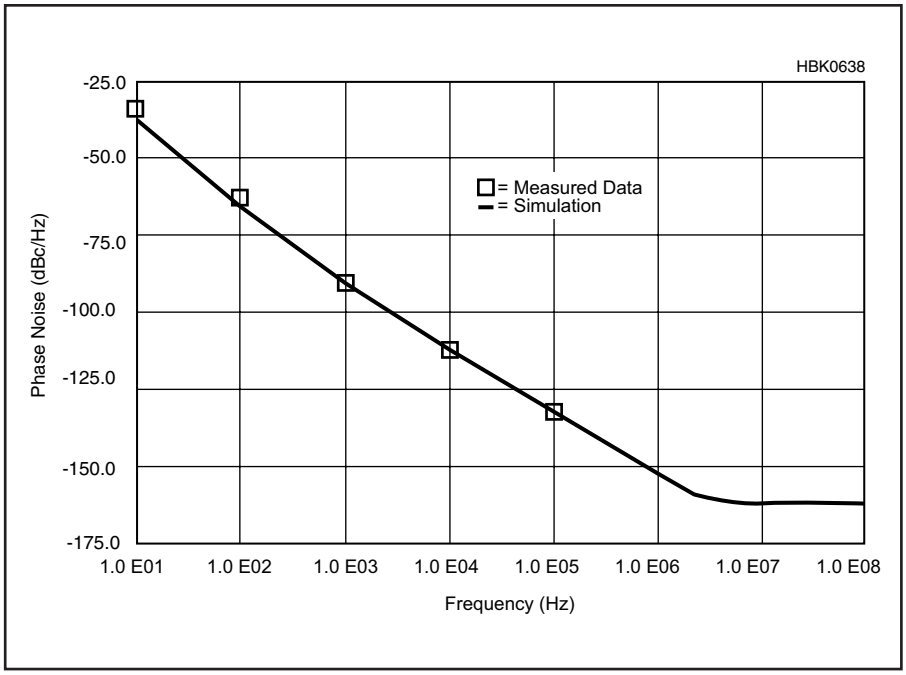


Fig 6.18 — Comparison between predicted and measured phase noise for the oscillator shown in Fig 6.17.

available, such as from **www.electronics-lab.com**. Contact Cadence for a full version or for more information. *AIM-spice* (**www.aimspice.com**) is a PC version of *SPICE* with a revised user interface, simulation control, and with extra models. A student version can be downloaded. Table 6.1 earlier in this chapter shows other free *SPICE* offerings.

There are a number of PC based *SPICE* programs in the $1000 range but they are designed more for switching power supplies and logic circuits optimization than RF. *ICAP4* (**www.intusoft.com/demos.htm**) and *MICROCAP9* (**www.spectrum-soft.com/index.shtm**) both have demonstration/evaluation versions available for download

Agilent, AWR, Ansys, and Synopsis offer very modern mixed-mode CAD tools and they combine the concept of *SPICE* with the advanced technologies. These are professional quality tools, but if one can arrange to make use of them through a friend or associate, the results are worth investing the time to learn their use.

6.4 CAD for PCB Design

[With numerous PCB design software packages available and low quantity, low cost PCB manufacturing services accepting orders electronically, the development of PCBs has never been easier for the amateur. As with any assembly or manufacturing process, it is important to understand the vocabulary and technology in order to achieve the desired result. Thus, this section provides a detailed description of the entire process of PCB design. — *Ed.*]

The primary goal of using software for printed circuit board (PCB) design is the production of so-called PCB artwork — the graphic design used to create the patterns of traces that establish connectivity on the PCB. Historically, PCB artwork was created by hand on clear film using black tape and special decals which were then photographically reduced. However, free and low cost programs specifically for the PCB design process are now widely available. These programs not only allow the creation of artwork efficiently and accurately, but produce the required ancillary files for commercial production, exchange information with schematic capture software, produce Bills of Materials (parts lists), and even include such features as three dimensional visualization of the finished board. While artwork files can be shared with other people for PCB production, the "source" files used by the CAD program can typically only be used by other people who share the same program.

The decision to produce a PCB must take into account the nature of the circuit itself (for example, high frequency, low noise and high current circuits require additional care). Other considerations are time available, expense, available alternatives, quantity required, ability to share and replicate the design, and non-electrical characteristics such as thermal and mechanical, as well as desired robustness.

6.4.1 Overview of the PCB Design Process

The PCB design process begins with establishing the list of components in the circuit, the connections between the components, the physical outline/size of the board, and any other physical, thermal and electrical constraints or design goals. Much of the connectivity and component information is reflected in the schematic for a circuit, so in many cases the PCB layout process begins by entering the schematic in a schematic capture program which may be integrated with the PCB CAD program or standalone. (Schematic capture is not required for PCB layout.) Once the schematic is entered, there may be other options possible such as simulating the circuit as described in the preceding sections. A clean, well organized schematic that is easily modified is an asset regardless of the circuit production and construction methods.

With input from the schematic and other information, the board outline is created, mounting and other holes placed, the components positioned, and the pattern of traces created. Once the layout is complete, in many cases it is possible to run a design rules check — the equivalent of a "spell checker." Design rules include component connections and other information to check for problems related to connectivity and manufacturability. This step can save a great deal of time and expense by catching errors that could be fixed by hand, but would otherwise negate some of the benefits of a PCB.

The final step in the PCB layout program is to produce the collection of up to a dozen or so different files required for PCB production. In brief, the list includes the artwork for the pattern of traces, files for producing the board outline, solder masks, silk screens and holes.

The user then uploads the set of files to a PCB manufacturer. As quickly as two to three days later an envelope will be delivered with the freshly minted boards ready for assembly! Alternatively, the user may create the board "in house" using photomechanical or other processes based on the output files from the software, as is discussed in the **Construction Techniques** chapter along with PCB assembly techniques.

6.4.2 Types of PCB Design Software

PCB software varies in features, function and cost, but for the radio amateur, the most interesting software for introductory use fall into the following categories:

1) Open Source: PCB design software such as *GNU PCB* (**http://pcb.gpleda.org**, for *Linux*, Mac OS X) and *KiCad* (**http://kicad.sourceforge.net/wiki/index.php/Main_Page**, for *Linux*, Mac OS X, and *Windows*, includes schematic capture) are free to use and have no artificial restrictions. Support is through user forums. Source code is available for the user to modify. *gschem* is a schematic capture sister program to *GNU PCB*.

2) Free, restricted use/restricted feature commercial: At least one company makes a version of their PCB and schematic software that is free to use for noncommercial purposes. Though it is restricted in number of layers (two) and maximum board size (4 × 3.2 inches), *Eagle PCB "Light Edition"* (**www.cadsoftusa.com**, for *Linux*, Mac OS X, *Windows*) is very popular among hobbyists. Files can be shared with others; the resulting industry standard files can be sent to nearly any PCB manufacturer. *Eagle* also contains a schematic entry program.

3) Free, restricted output commercial: Several PCB manufacturers offer schematic and PCB software with a proprietary output format tied to their PCB manufacturing service. *PCB123* from Sunstone Circuits (**www.sunstone.com**, for *Windows*) is one such offering, including schematic capture and layout software with up to four layers and board sizes up to 12 × 18 inches (double sided). For an additional fee (per design), industry standard files can be exported. Schematic entry is included. *Express PCB* (**www.expresspcb.com**, for *Windows*) also provides schematic capture and PCB layout capability, tied to the Express PCB board fabrication services, including the fixed size (3.8 × 2.5 inches) Miniboard service. Advanced Circuit's proprietary *PCB Artist* software (**www.4pcb.com**, for *Windows*) includes the ability to import netlists.

4. Low cost commercial: Eagle and many other companies offer PCB and schematic software at a range of prices from $50 to many thousands of dollars. Several versions are typically offered from each company, usually based on limitations on board size, schematic size/complexity and features such as autorouting. Schematic entry may be included in some packages, or be a separate purchase.

PCB design software manuals and tutorials discuss the basic operation but also special keystrokes and other shortcuts that make operations such as routing traces much more efficient.

The first time designing and ordering a PCB can be daunting, so keep the initial job simple and pay attention to details (and read the instructions). When starting to use a specific software package, join a user's support group or forum if one is available. Request sample designs from other users and experiment with them to see how they are constructed and what files are required in the output data set. Once you are comfortable with the tools, you can begin on a design of your own.

6.4.3 Schematic Capture

The first step in PCB design is to create a schematic. It is possible to design a layout directly from a paper schematic, but it is much easier if the schematic is entered (or "captured") in electronic form. Schematic capture software has two outputs — the visual schematic and the component and connectivity data for subsequent PCB layout. These two separate requirements can make some operations during schematic entry more complicated than what would seem at first glance necessary. Bear in mind however, that the user is creating not only a clear graphic representation of the circuit, but of the underlying electrical connectivity.

Schematics are generally entered on a (virtual) page usually corresponding to common paper sizes — for example, 11 × 17 inches. More complicated schematics can span multiple pages, using special labels or components to indicate both visual and electrical connectivity. Often one can group logically related elements into a module that can then be referenced as a "black box" on a higher level schematic. For complex circuits, these features are extremely useful and make the difference between a jumbled diagram that is difficult to use and an organized, compact diagram that efficiently communicates the function and operation of the circuit.

COMPONENTS

The components (resistors, capacitors, etc) on a schematic are either selected from an existing library or created by the user and stored in a custom library. It is also possible to find components and/or additional libraries on the Internet, although each program has its own specific format.

Each component includes a great deal more than shape and pin numbers. A typical component library entry includes:

Symbol — This is the graphic representation shown on the schematic. Many components may have the same symbol (eg, the op amp symbol may be shared by many different types of op amps)

Pins — For each pin or point of electrical connection, the component model may specify the pin number, label (eg, "V_{DD}"), pin type (inverting, noninverting) or pin functions (common).

PCB footprint — A given component may be available in a number of different packages (eg, DIP or surface mount). Many components may have the same physical footprint (eg, op amps, comparators and optoisolators could all map to the same eight-pin DIP footprint). Footprints include the electrical connections (pins) as well as mechanical mounting holes and pad sizes, and the component outline.

Value — Many components such as resistors and capacitors will have identical information except for a difference in value. All ¼ W resistors may be instances of the same component, differing only in value and designator.

Designator — The unique reference to the component, such as R1, C7, D3. This is assigned when the component is used (often automatically and in sequence).

Source information — Part number, vendor, cost, etc. This information is for the Bill of Materials.

Components are typically placed on the schematic by opening a library and searching for the desired component. It may be tempting for the beginner to select a component that looks "about right" when faced by a long list of components in some libraries. However, even at this early stage, the physical PCB often must be taken into account. For example, either "1/8W Resistor, Axial" or "1W Resistor, Upright" will result in the same neatly drawn resistor symbol on the schematic but in the subsequent step of using the component data to create a PCB, the footprints will be dramatically different.

It is not at all uncommon to add new components to the library in the course of creating a schematic. Since many components are closely related to existing devices, the process often consists of selecting an existing schematic symbol, editing the shape and/or component data, creating a new label, and associating the part with an existing footprint. Adding a specific type of op amp is an example. This usually only needs to be done once since symbols can be saved in a personal library (and shared with others). It is usually easier to modify a part that is close to what is desired than to "build" a new part from scratch.

Component symbols can generally be rotated and flipped when placing the component instance on the schematic. Designators (R1, T34, etc) can be assigned and modified by the user although the default designators are usually selected sequentially.

CONNECTIONS

The schematic software will have a mode for making electrical connections, called "nets." For example, one might click on the "draw net" symbol then draw a line using the mouse from one pin to another pin, us-

ing intermediate mouse clicks to route the line neatly with 90° turns on a uniform grid. Internally, the software must not only draw the visual line, but recognize what electrical connectivity that connection represents. So one must click (exactly) on a component pin to start or end a line or when making a connection between two lines that intersect, explicitly indicate a net-to-net connection (often with a special "dot" component). The connections on a schematic can often be assigned additional information, such as the desired width of the trace for this connection on the PCB or a name assigned by the user, such as "input signal."

Not all connections on a schematic are drawn. To make any schematic — electronic or hand drawn — more readable, conventions are often employed such as ground or power symbols or grouping similar connections into busses. Schematic capture software often supports these conventions. In some cases, components may be created with implicit power connections; in these cases the connections may not even be noted on the schematic but will be exported to the PCB software. However, as a general rule, software aimed at beginning PCB designers will not require the use of these advanced features.

Since it is often possible for component pins to be assigned attributes such as "power input", "output," "input," and so on, some schematic entry programs allow one to do an early design check. The program can then flag connections between two outputs, inputs that are missing connections, and so on. This is not nearly as helpful or complete as the Design Rule Check discussed below.

Free text can be placed on the schematic and there will be a text block in a corner for date, designer, version, title and the other information that identifies the schematic.

NETLISTS

Once the components are placed and connections made, the schematic may be printed and any output files for the PCB layout software produced. The connectivity and component information needed for PCB layout is captured in a *netlist* file. The flow from schematic entry to PCB may be tightly integrated, in which case the user may switch between schematic and PCB like two views of the same design (which they are). However, most schematic software will generate a separate netlist to be used by PCB layout software, whether integrated or a separate program. The netlist can also be exported to an external circuit simulation program or be used by an integrated simulator program. (See the first part of this chapter for more information on circuit simulation.)

Netlists are often human readable text files and in most cases it is possible to create a netlist file manually. In the absence of a schematic entry program, this allows the user to take a hand drawn schematic, extract the connectivity information, and create the netlist for the PCB program to perform design rule checks. However, a netlist is generally not required for the PCB layout software; the user will also have the option to create a PCB on-the-fly, adding components and connections as they wish.

ANNOTATION AND BILL OF MATERIALS

The important features of *forward* and *backward annotation* enter at the interface between schematic entry and PCB layout. It is not uncommon during the PCB layout process to either come across some design deficiency or realize that a change to the schematic could produce a design that would be easier to lay out. Likewise, a review of the schematic partway through the PCB layout process could reveal some needed design change. In the case of changes to the PCB (perhaps changing some pins on a connector to make routing easier), back annotation can propagate the changes "backward" to the schematic. The connectivity data will be updated; however the user may need to manually route the connection lines to neaten up the schematic. Likewise, changes to the schematic when the PCB is already (partially) routed are known as forward annotations and like the schematic, while the connectivity is updated the user will likely need to manually route the traces. Neither forward nor back annotation is necessary, but is useful in keeping the schematic and PCB consistent. In their absence, the user is strongly urged to keep the schematic and PCB up to date manually to avoid time consuming problems later on.

Finally, the underlying data in the schematic can be used to produce a *Bill of Materials* (BOM). A BOM lists all the components of the schematic, typically ordered by reference designator(s), and may even be exportable for online ordering.

6.4.4 PCB Characteristics

PCB CONSTRUCTION

It is useful to know a little bit about PCB construction in order to make sense of the PCB design process. **Fig 6.19** shows the basic structure of a PCB and some of its design elements (discussed in later sections). The lami-

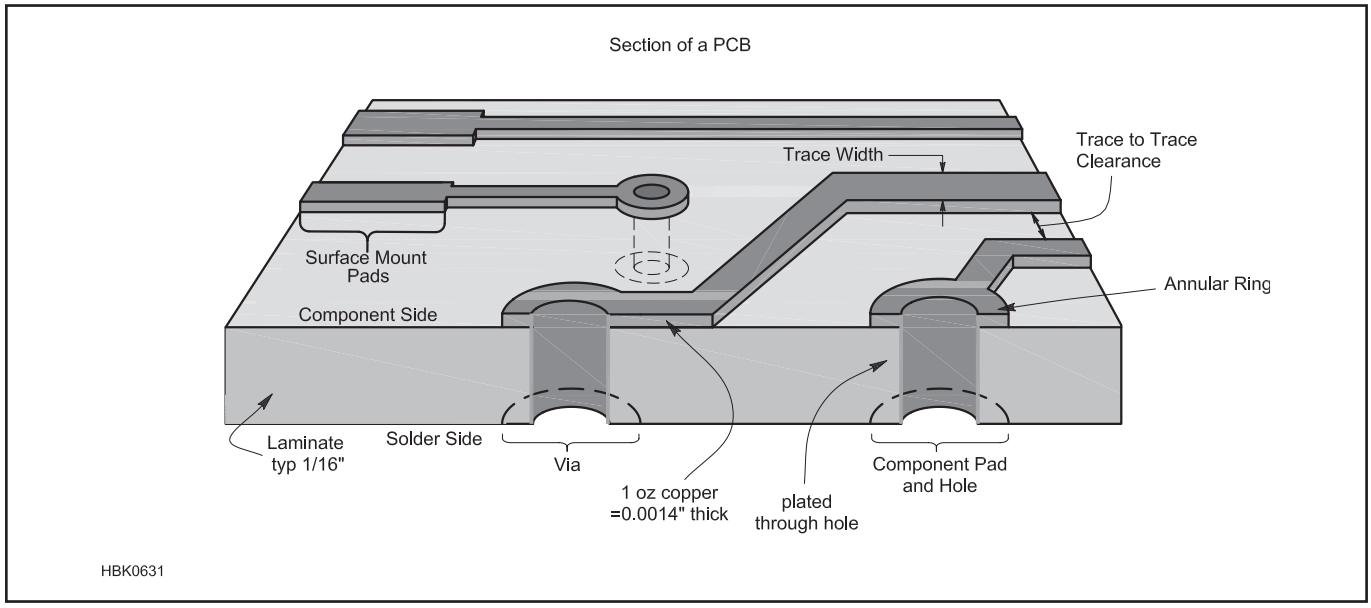


Fig 6.19 — The various elements of PCB construction and specification.

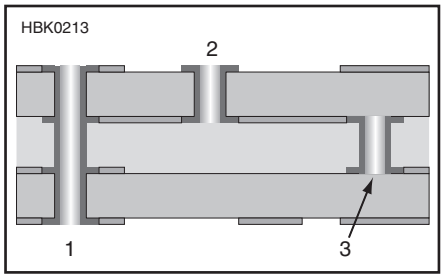


Fig 6.20 — *Via* refers to a plated-through hole that connects one board layer to another. Vias are used for signal, power, or ground connections and even for ventilation. Different via types include through-hole (1); blind (2), and buried (3).

nate material provides a stable, insulating substrate with other known characteristics (thermal, dielectric, etc). Copper is bonded to one or both sides and selectively removed (usually chemically) to leave traces and pads. The pads provide points of connection for components. Though electrical connectivity is crucial, it is important to remember that the solder and pads provide mechanical and thermal connectivity as well. Pads may be drilled for mounting *through-hole* components or left undrilled for *surface-mount* components.

A separate electrochemical process plates the inside surface of *plated-through holes* to provide connectivity between upper and lower pads. Plated-through holes whose sole purpose is to provide electrical connectivity between layers of a PCB are known as *vias*, shown in **Fig 6.20**. Since they do not need to accommodate a component lead, their hole and pad size are smaller.

While two layer boards can mount components on either side, most PCBs will have a primary side called the *component side* upon which most of the components will be placed, and a *solder side* dominated by soldered pins and traces. Where high density is required, surface mount (and sometimes through-hole) devices are mounted on both sides, but this is considerably more complex.

Multi-layer boards are essentially a stack of two or more two-layer boards, with an insulating layer between each board. Plated-through holes make connections possible on every layer, and the laminate material is proportionally thinner so the entire multi-layer board is roughly the same thickness as a regular two-layer board. Vias that join selected, adjacent copper layers without connecting the entire stack of layers are called "buried" or "blind" vias and are typically only needed for very dense designs. Multi-layer boards provide much more flexibility in routing signals and some other benefits such as dedicated layers for power distribution and grounding, but at often substantial additional cost.

PCB MANUFACTURING SPECIFICATIONS

Unless the board is manufactured by the hobbyist, the PCB files are sent out to be manufactured by a *board house*. The most important issue for the amateur may be the pricing policies of the board house. Board size, quantity, delivery time, number of layers, number and/or density of holes, presence of solder masks and silk screens, minimum trace/separation width, type of board material, and thickness of copper will all influence pricing. One cost saving option of the past, a single-layer board, may not be offered with low-cost, low-volume services — two layers may be the simplest option and it results in a more robust board. [Note that most ordering specifications use English units of inches and ounces. Offshore board houses may use both English and metric units, or be metric-only. English units are used here because they are the most common encountered by hobbyists. — *Ed.*]

The second issue to consider is manufacturing capabilities and ordering options. These will vary with pricing and delivery times, but include the following:

Board material and thickness — FR-4 is the most popular board material for low volume PCBs; it consists of flame-resistant woven fiberglass with epoxy binder. Typical thickness is 0.062 inch (1/16 inch), but thinner material is sometimes available. Flexible laminates are also available at greater cost and longer delivery time. Special board laminates for microwave use or high-temperature applications are also available.

Copper thickness — Expressed in ounces per square foot, typical values are 1-2 oz (1 oz corresponds to 0.0014 inch of thickness.) Other values may not be available inexpensively for small volumes. Inner layers on multi-layer boards may be thinner — check if this is important. Most board designs can assume at least 1 oz copper for double-sided boards; trace width is then varied to accommodate any high current requirements.

Layers — Two-layer boards are the most common. Because of the way PCBs are manufactured, the number of copper layers will be multiples of two. For quick-turn board houses, usually only two or four layer boards are available. PCBs with more than two layers will always be more expensive and often take longer to manufacture.

Minimum hole size, number, and density of holes — Minimum hole size will rarely be an issue, but unusual board designs with high hole density or many different hole sizes may incur additional costs. Be sure to include vias when specifying minimum hole size. Some board houses may have a specific list of drill sizes they support. Note that you can often just edit the drill file to reduce the number of different drill sizes.

Minimum trace width and clearance — Often these two numbers are close in value. Most board houses are comfortable with traces at least 0.010 inch in width, but 0.008 and 0.006 inch are often available, sometimes at a higher cost.

Minimum annular ring — A minimum amount of copper is required around each plated-through hole, since the PCB manufacturing process has variations. This may be expressed as the ratio of the pad size to hole size, but more commonly as the width of the ring.

Edge clearances — Holes, pads, and traces may not be too close to the edge of the board.

Board outline and copies — There may be options to route the outline of the board in other shapes than a rectangle, perhaps to accommodate a specific enclosure or optimize space. If multiple copies of a board are ordered, some board houses can *panelize* a PCB, duplicating it multiple times on a single larger PCB (with a reduction in cost per board). These copies may be cut apart at the board house or small tabs left to connect the boards so assembly of multiple boards can be done as a single unit.

Tin plating — Once the traces and pads have been etched and drilled, tin plating is usually applied to the exposed copper surfaces for good soldering.

Solder mask — This is a solder-resistant coating applied after tin plating to both sides of the board covering everything except the component mounting pads. It prevents molten solder from bridging the gaps between pads and traces. Solder mask is offered except by the quickest turn services. Green is the most common color, but other colors may be available.

Silkscreen — This is the ink layer, usually white, on top of the solder mask that lays out component shapes and designators and other symbols or text. A minimum line width may be specified — if not specified, try to avoid thin lines. All but the quickest turn services typically offer silk screening on one or both sides of the PCB.

6.4.5 PCB Design Elements

The schematic may not note the specific package of a part, nor the width or length of a connection. The PCB, being a physical object, is composed of specific instances of components (not just "a resistor," but a "¼ W, axial-lead resistor mounted horizontally," for example) plus traces — connections between pins of components with a specific width and separation from other conductors. Before discussing the process of layout, we briefly discuss the nature of components and connections in a PCB.

COMPONENTS

A component in a PCB design is very similar to its counterpart on the schematic. **Fig 6.21** shows the PCB footprint of an opto-interrupter, including graphics and connectivity information. The footprint of a component needs to specify what the footprint is like on all applicable copper layers, any necessary holes including non-electrical mounting holes or slots, and any additional graphics such a silkscreen layer.

Take a common ¼ W axial-lead resistor as an example. This footprint will have two pins, each associated with a pad, corresponding to the resistor's two leads. This pad will appear on both the top and bottom layers of the board, but will also have a smaller pad associated with inner layers, should there be any. The hole's size will be based on the nominal lead diameter, plus some allowance (typically 0.006 inch). The pad size will be big enough to provide a reasonable annular ring, but is usually much larger so as to allow good quality soldering. The pins will be labeled in a way that corresponds to the pin numbering on the schematic symbol (even though for this component, there is no polarity). A silkscreen layer will be defined, usually a box within which the value or designator will appear. The silkscreen layer is particularly useful for indicating orientation of parts with polarity.

More complicated parts may require additional holes which will not be associated with a schematic pin (mounting hole, for example). These are usually added to the part differently than adding a hole with a pad — in this case, the hole is desired without any annular ring or plating. The silkscreen layer may be used to outline the part above and beyond what is obvious from the pads, for example, a TO-220 power transistor laying on its back, or the plastic packaging around the opto-interrupter in Fig 6.21.

As with schematic entry, it is not uncommon to have to modify or create a new PCB footprint. Good technical drawings are often available for electronic parts; when possible the user should verify these dimensions against a real part with an inexpensive dial caliper. It is also useful to print out the finished circuit board artwork at actual size and do a quick check against any new or unusual parts.

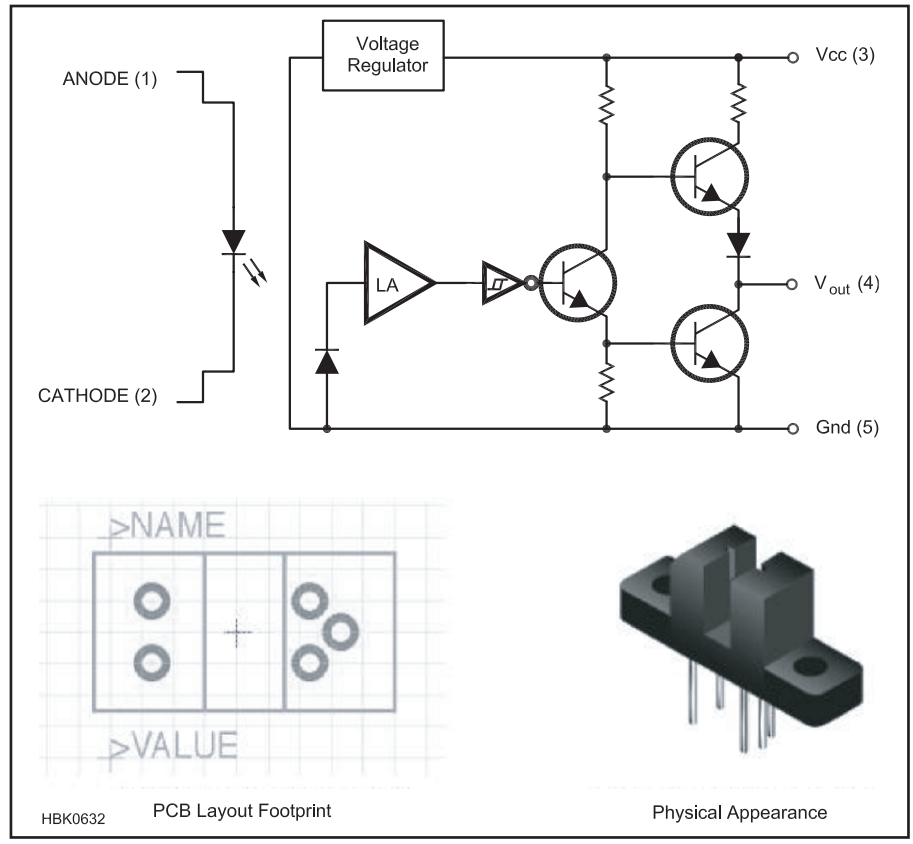


Fig 6.21 — The PCB footprint for a component, such as the opto-interrupter shown here, combines electrical connectivity as defined in the part's schematic and the part's physical attributes.

TRACES

Traces are the other main element of PCB construction — the copper pathways that connect components electrically. PCB traces are merely planar, flat wires — they have no magical properties when compared to an equivalent thickness of copper wire. At VHF/UHF/microwave frequencies and for high-speed digital signals, PCB traces act as transmission lines and these properties need to be accounted for, and can be used to advantage, in the design.

There are few constraints on traces apart from those such as minimum width and clearance imposed by the board house. They are created by chemical etching and can take arbitrary shapes. In fact, text and symbols may be created on copper layers which may be handy if a silkscreen is not included. Traces may be of any length, vary in width, incorporate turns or curves, and so on. However, most traces will be a uniform width their entire length (a width they will likely share with other traces carrying similar signals), make neat 45° or 90° corners, and on two-layer boards have a general preference for either horizontal travel on the component side or vertical travel on the solder side.

The same considerations when building a circuit in other methods applies to PCB design, including current capacity (width of trace, thickness of copper), voltage (clearance to other signals), noise (shielding, guarding, proximity to other signals), impedance of ground and power supplies, and so on.

6.4.6 PCB Layout

With a schematic and netlist ready and all of the PCB characteristics defined, the actual layout of the PCB can begin.

BOARD SIZE AND LAYERS

The first step in PCB layout is to create the board outline to contain not only the circuit itself and any additional features such as mounting holes. For prototype or one-off designs, the board is often best made a bit larger to allow more space between components for ease in testing and debugging. (Some low cost or freeware commercial PCB software imposes limits on board size and number of components.) The board outline may be provided in a default size that the user can modify, or the user may need to enter the outline from scratch.

As discussed above, rectangular board shapes are generally acceptable, but many board houses can accommodate more complex outlines, including curves. These outlines will be routed with reasonable accuracy and may save an assembly step if the PCB needs a cutout or odd shape to fit in a specific location.

While the software may not require deciding at the start how many layers the PCB will use, this is a decision the user should make as early as possible, since the jump from two to four or more will have a big impact on routing the traces as well as cost!

For your initial design, start with a two

layer board for a simple circuit that you have already built and tested. This will reduce the number of decisions you have to make and remove some of the unknowns from the design process.

COMPONENT PLACEMENT

Good component placement is more than half the battle of PCB layout. Poor placement will require complicated routing of traces and make assembly difficult, while good placement can lead to clean, easy-to-assemble designs.

The first elements placed should be mounting holes or other fixed location features. These are often placed using a special option selected from a palette of tools in the software rather than as parts from a library. Holes sufficient for a #4 or #6 screw are usually fine; be sure to leave room around them for the heads of the screws and nut driver or standoff below. These will be non-plated-through holes with no pad (though the board house may plate all the holes in a board, regardless).

Depending on the software and whether schematic capture was performed, the board outline may already contain the footprints of all the circuit components (sometimes stacked in a heap in one corner of the board) and the netlist will already be loaded. In this case, components may be placed by clicking and dragging the components to the desired location on the board. Most PCB programs have a "rat's nest" option that draws a straight line for each netlist connection of a component, and this is a great aid in placement as the connections between components are apparent as the components are moved around. (See **Fig 6.22**) However, connections are shown to the nearest pin sharing that electrical connection; thus, components such as decoupling capacitors (which are often meant to be near a specific component) will show rats nest connections to the nearest power and ground pins and not the pins the designer may have intended. These will have to be manually edited.

The PCB layout software may offer *autoplacement* in which the components are initially arranged automatically. The beginner should certainly feel free to experiment and see how well this tool performs, but it is likely not useful for the majority of designs.

PCBs need not be arranged to precisely mimic the schematic, but it is appropriate to place components in a logical flow when possible so as to minimize the length of traces in the signal path. Sensitive components may need to be isolated or shielded from other components, and grounding and decoupling attended to, just as one would do with a point-to-point soldered version.

If the PCB is being designed "on-the-fly" or using an imported netlist, components may need to be selected and placed on the board manually using the libraries of parts in the PCB software. Not all design software makes this task simple or fast — in particular, the description of component footprints may be confusing. The use of highly condensed industry standard or non-uniform naming conventions often means the user needs to browse through the component library to see the different types of components. Resistors, diodes and capacitors seem particularly prone to a propagation of perplexing options. One solution is to open an example PCB layout and see what library element that designer used for resistors, LEDs and so on. Here, the PCB layout software directed at hobbyists may be superior in that there are fewer options than in professional programs.

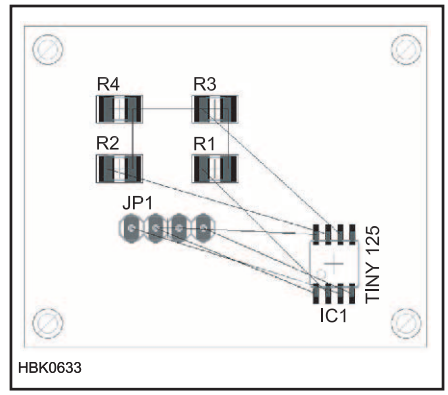


Fig 6.22 — The rat's nest view during PCB layout shows the direct connections between component pins. This helps the designer with component placement and orientation for the most convenient routing.

During placement the user will find that different orientations of components simplify routing (for example, minimizing the number of traces that have to cross over each other or reducing the trace lengths). Components are generally rotated in increments of 90°, although free rotation may be an option. The user is strongly urged to maintain the same orientation on like devices as much as possible. Mixing the position of pin 1 on IC packages, or placing capacitors, diodes, and LEDs with random orientation invites time consuming problems during assembly and testing that can be minimized by consistent, logical layout.

Placement and orientation of components can also affect how easily the final PCB can be assembled. Allow plenty of space for sockets, for example, and for ICs to be inserted and removed. Components with a mechanical interface such as potentiometers and switches should be positioned to allow access for adjustment. Any components such as connectors, switches, or indicators (eg, LEDs) should be positioned carefully, especially if they are to protrude through a panel. Often this will involve having the component overhang the edge of the PCB. (Beware of the required clearance between copper traces and pads to the edge of the PCB.)

Components should include a silkscreen outline that shows the size of the whole component — for example, a transistor in a TO-220 package mounted flat against the PCB should have an outline that shows the mounting hole and the extent of the mounting tab. The user should also consider the clearance required by any additional hardware for mounting a component, such as nuts and bolts or heatsinks — including clearance for nut drivers or other assembly tools.

Take care to minimize the mechanical stress on the PCB, since this can result in cracked traces, separated pads, or other problems. Utilize mounting holes or tabs when possible for components such as connectors, switches, pots. Use two-layer boards with plated-through holes even if the design can be single-layer. Component leads soldered to plated-through holes produce much stronger mechanical connections than single-layer boards in which the soldered pad is held only by the bond between copper and laminate and is easily lifted if too much heat or stress is imposed.

When prototyping a new design, add a few unconnected pads on the circuit board for extra components (eg, a 16 pin DIP, 0.4 inch spaced pads for resistors and other discrete components). Include test points and ground connections. These can be simply pads to which cut off leads can be soldered to provide convenient test points for ground clips or to monitor signals.

Wires or cables can be directly soldered to the PCB, but this is inconvenient when swapping out boards, and is not very robust. Connectors are much preferred when possible

PCB Design and EMC

While amateur projects are rarely subject to electromagnetic compatibility (EMC) standards, using good engineering practices when designing the board still reduces unwanted RF emissions and susceptibility to RF interference. For example, proper layout of a microprocessor circuit's power and ground traces can reduce RF emissions substantially. Proper application of ground planes, bypass capacitors and especially shield connections can have a dramatic effect on RFI performance. (See the **RF Interference** chapter for more on RFI.) A good reference on RFI and PCB design is *Electromagnetic Compatibility Engineering*, by Henry Ott, WA2IRQ.

and often provide strain relief for the wire or cable. However, if a wire is directly soldered to the PCB, the user should consider adding an unplated hole nearby just large enough to pass the wire including insulation. The wire can then be passed from the solder side through the unplated hole, then soldered into the regular plated-through hole. This provides some measure of strain relief which can be augmented with a dollop of glue if desired.

ROUTING TRACES

After placing components, mounting holes and other fixed location features that limit component or trace placement, traces can be *routed*. That is, to complete all the connections between pins without producing short circuits.

Most PCB design programs allow components and traces to be placed on a regular grid, similarly to drawing programs. There may be two grids — a visible coarse pitch grid, and a "snap" fine pitch grid, to which components and other objects will be aligned when placed. It is good practice to use a 0.1 or 0.050 inch grid for component placement and to route traces on a 0.025 inch grid. While the "snap to grid" feature can usually be turned off to allow fine adjustment of placement, a board routed on a grid is likely to look cleaner and be easier to route.

The trace starts at a component pin and wends its way to any other pins to which it should be connected. Traces should start and end at the center of pads, not at the edge of a pad, so that the connection is properly recorded in the program's database. If a netlist has been loaded, most PCB software will display a rat's nest line showing a direct connection between pads. Once the route is completed, the rat's nest line for that connection disappears. The rat's nest line is rarely the desired path for the trace and often not the correct destination. For example, when routing power traces, the user should use good design sense rather than blindly constructing a Byzantine route linking pins together in random order. For this reason, routing the power and ground early is a good practice.

High speed, high frequency, and low noise circuits will require additional care in routing. In general, traces connecting digital circuits such as microprocessors and memories should not cross or be in close proximity to traces carrying analog or RF signals. Please refer to the **RF Techniques** and **Construction Techniques** chapters of this *Handbook*, and the references listed at the end of this section.

Manual routing is a core skill of PCB design, whether or not auto-routing is used. The process is generally made as simple as possible in the software, since routing will take up most of the PCB design time. A trace will be routed on the copper layer currently selected. For a single-sided board, there is only one layer for routing; for a two-layer board the component side and solder side can have traces; and for multi-layer boards additional inner layers can have traces. Often a single keystroke can change the active copper layer (sometimes automatically inserting a via if a route is in progress). The trace is drawn in straight segments and ends at the destination pin. When routing, 90° corners are normally avoided — a pair of 45° angles is the norm. **Fig 6.23** shows some sample traces.

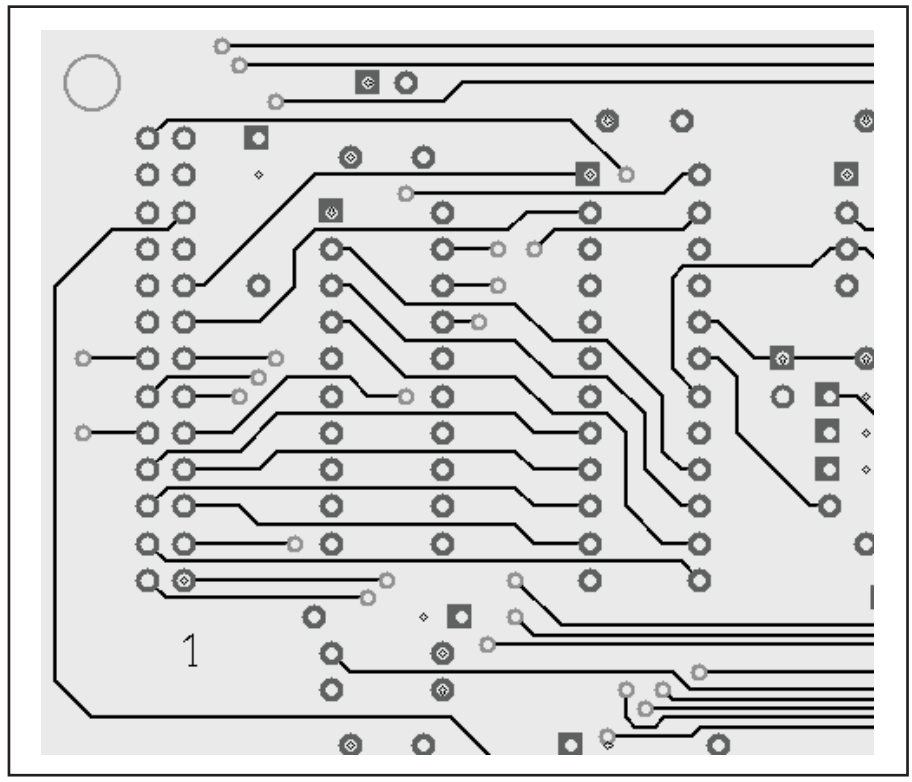


Fig 6.23 — This example shows traces on the side of the PCB for horizontal routing. Traces are routed between pins of ICs. The smaller pads are for vias to a different layer of the PCB.

It is good practice on a double-sided board to have one side of the board laid out with mostly horizontal traces, and the other side laid out with mostly vertical traces. A trace that needs to travel primarily vertically can do so on the side with vertical traces and use a via to move to the other side to complete the horizontal part of the route.

It is easiest during testing and debugging to route most traces on the bottom (solder) side of the board — traces on the component side often run under ICs or other components, making them hard to access or follow. It is often much clearer to connect adjacent IC or connector pins by routing a trace that leaves one pad, moves away from the IC or connector, then heads back in to the adjacent pad to connect. This makes it clear the connection on the assembled board is not a solder bridge, which a direct connection between the two pads would resemble.

It may be the case that no amount of vias or wending paths can complete a route. The one remaining tool for the PCB designer is a jumper — a wire added as a component during assembly just for the purpose of making a connection between two points on the board. Jumpers are most often required for single-sided boards; when the "jump" is rather small, uninsulated wire can be used. Jumpers are usually straight lines, and can be horizontal or vertical. Professional production PCBs use machine-insertable zero-ohm resistors as jumpers. Jumpers on double-sided boards are usually not viewed very favorably, but this is an aesthetic and efficiency issue, not a functional one.

Multi-layer boards clearly offer additional routing options, but again having some dominant routing direction (vertical or horizontal) on each layer is recommended, since mixing directions tends to cause routing problems. However, it is not uncommon to devote one or two inner layers to power and ground, rather than merely be additional layers for routing signals. This allows power and ground to be routed with minimal resistance and exposes the traces carrying interesting signals on the component and solder sides where they are available to be probed or modified. It is very difficult to modify traces on inner layers, needless to say!

Before routing too many traces, it is help-

ful to run the *Design Rule Check (DRC)* on the board. (See the section on Design Rule Checking below.) Applied early and often, DRC can identify areas of concern when it is easiest to correct. For example, a given trace width may provide insufficient clearance when passing between two IC pads.

Some PCB design packages offer *auto-router* capability in which the software uses the component and connectivity data of the netlist and attempts to route the traces automatically. There are some circumstances when they save time, but view these tools with some caution. Auto-routers are good at solving the routing puzzle for a given board, but merely connecting all the pins correctly does not produce a good PCB design. Traces carrying critical signals may take "noisy" routes; components that should have short, low resistance connections to each other may have lengthy traces instead, and so on. More sophisticated auto-routers can be provided with extensive lists of "hints" to minimize these problems. For the beginner, the time spent conveying this design information to the auto-router is likely better spent manually routing the traces.

If an auto-router is used, at a minimum, critical connections should be first routed manually. These include sensitive signals, connections whose length should be minimized, and often power and ground (for both RFI and trace width reasons). Better still is to develop a sense of what a good layout looks like (which will come with practice and analyzing well designed boards), and learn at what stage the auto-router can be "turned loose" to finish the routing puzzle.

TRACE WIDTH AND SPACING

All traces will have some width — the width may be the default width, the last width selected, or a width provided from data in the netlist. It may be tempting to route all but the power traces using the smallest trace width available from the board house (0.008 inch or smaller), since this allows the highest density of traces and eases routing. A better design practice is to use wider traces to avoid hard-to-detect trace cracking and improve board reliability. The more common traces 0.012 inch wide can be run in parallel on a 0.025 inch grid and can pass between many pads on 0.1 inch centers. Even wider traces will make the board easier to produce "in house," though the exact process used (CNC routing, chemical etching, etc) will limit the resolution. Note that it is possible to "neck down" traces where they pass between IC or connector pads — that is, the regular, thick trace is run up close to the narrow gap between the pads, passes between the pads with a narrow width, then expands back to the original width. There is little reason to use traces wider than 0.030 inch or so for most signals (see **Table 6.3**) but power and ground trace widths should be appropriate for the current.

Table 6.3
Maximum Current for 10 °C Rise, 1 oz/ft² Copper
Based on IPC-2221 standards (not an official IPC table)

Trace Width (inches)	*Max. Current (External Trace) (A)*	*Max. Current (Internal Trace) (A)*	*Resistance (ohms/inch)*
0.004	0.46	0.23	0.13
0.008	0.75	0.38	0.063
0.012	1.0	0.51	0.042
0.020	1.5	0.73	0.025
0.040	2.4	1.2	0.013
0.050	2.8	1.4	0.010
0.100	4.7	2.4	0.0051
0.200	7.8	3.9	0.0025
0.400	13	6.4	0.0013

IPC-2221 Generic Standard on Printed Circuit Design,
Institute for Interconnecting and Packaging Electronic Circuits, **www.ipc.org**

All traces have resistance, and this resistance is a function of the cross section of the trace (width times thickness) and the length. This resistance will convert electrical power to heat. If the heat exceeds a relatively high threshold, the trace becomes a fairly expensive and difficult-to-replace fuse. The trace width should be selected such that for the worst case expected current, heat rise is limited to some threshold, often 10 °C. In practice, power traces (especially grounds) are often made as wide as practical to reduce resistance, and they greatly exceed the width required by heat rise limits alone.

Table 6.3 summarizes maximum currents for external (component and solder side) and internal traces for some common trace widths. Internal traces (on inner layers of multi-layer boards) can carry only about half the current of external traces for the same width since the internal layers do not dissipate heat to the ambient air like external traces can. (Note that trace widths are also sometimes expressed in "mils." 1 mil = 0.001 inch; it is shorthand for "milli-inch", not millimeter!)

There is no upper bound on the effective trace width. It is common to have large areas of the board left as solid copper. These *copper fill* areas can serve as grounds, heat sinks, or may just simplify board production (especially homemade boards). It is not a good idea to place a component hole in the middle of a copper fill — the copper is a very efficient heat sink when soldering. Instead, a "wagon wheel" pattern known as a thermal relief is placed (sometimes automatically) around the solder pad, providing good electrical connectivity but reducing the heat sinking. Often, copper fill areas can be specified using a polygon and the fill will automatically flow around pads and traces in that area, but can lead to isolated pads of copper.

In practice, most boards will have only two or perhaps three different trace widths; narrow widths for signals, and a thicker width for power (usually with a healthy margin).

One final note on trace width — vias are typically one size (ie, small), but multiple vias can be used to create low resistance connections between layers. Spacing the vias so their pads do not touch works well; the pads are then shorted on both top and bottom layers.

Voltage also figures into the routing equation, but instead of trace width, higher voltages should be met with an increased clearance between the trace and other copper. The IPC-2221 standard calls for a clearance of 0.024 inch for traces carrying 31-150 V (peak) and 0.050 inch for traces carrying 151-300 V (peak); these are external traces with no coating. (With the appropriate polymer solder mask coating, the clearances are 0.006 inch for 31-100 V and 0.016 inch for 101-300 V. Internal traces also have reduced clearance requirements.) Fully addressing the safety (and regulatory) issues around high voltage wiring is outside the scope of this brief review, however, and the reader is urged to consult UL or IPC standards.

SILKSCREEN AND SOLDER MASK

The silkscreen (or "silk") layer contains the text and graphics that will be silkscreened on the top of the board, shown in **Fig 6.24**. Components will generally have elements on the silkscreen layer that will automatically appear, such as designators and values, but other elements must be created and placed manually. Common silkscreen elements include: Circuit name, date, version, designer (and call sign), company name, power requirements (voltage, current, and fusing), labels for connections (eg, "Mic input"), warnings and cautions, labels for adjustments and switches. A solid white rectangle on the silkscreen layer can provide a good space to write a serial number or test information.

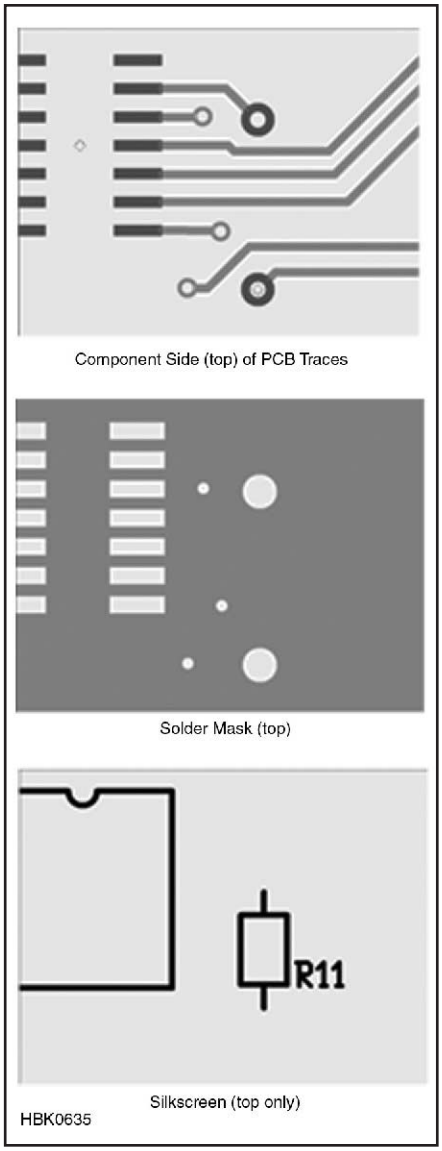


Fig 6.24 — The relationship between the layout's top copper layer with traces and pads, the solder mask that covers the copper (a separate solder mask is required for the top and bottom layers of the PCB) and the silkscreen information that shows component outlines and designators.

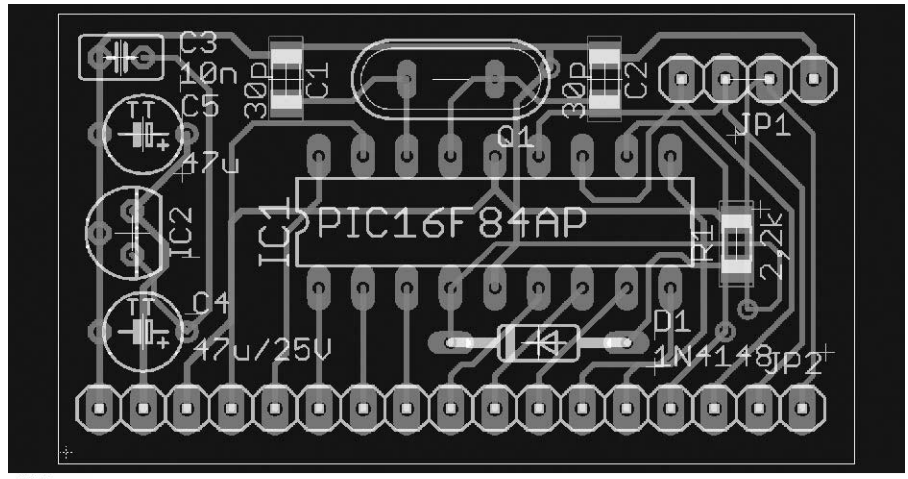


Fig 6.25 — A completed microprocessor board as it is seen in a typical PCB layout editor (*Eagle*). Solder mask layers are omitted for visibility. Traces that appear to cross each other are on different sides of the board and are in different colors in the layout software. The silkscreen layer is shown in white.

The board house will specify the minimum width for silkscreen lines, including the width of text. Text and graphics can be placed anywhere on the solder mask, but not on solder pads and holes.

Many quick-turn board houses omit the silkscreen for prototype boards. As noted earlier, many of the text elements above can be placed on the external copper layers. Component outlines are not possible since the resulting copper would short out traces, but component polarization can be noted with symbols such as a hand-made "+" made from two short traces, or a "1" from a single short trace. (Note that some component footprints follow a practice of marking the pad for pin 1 with a square pad while others are round or oval.)

The solder mask is a polymer coating that is screened onto the board before the silkscreen graphics. As shown in Fig 6.24, it covers the entire surface of the board except for pads and vias. Solder masking prevents solder bridges between pads and from pads to traces during assembly and is particularly important for production processes that use wave soldering or reflow soldering. There is one solder mask layer for the top layer and another for the bottom layer. Internal layers do not need a solder mask. Solder masking may be omitted for a prototype board, but care must be taken to keep solder from creating unwanted bridges or short circuits.

During the PCB layout process, solder mask layers are generally not shown because they do not affect connectivity. **Fig 6.25** shows a typical PCB as it appears when the PCB layout process is complete.

DESIGN RULE CHECK

If a netlist has been provided from the schematic capture program, a *design rule check* (*DRC*) can be made of the board's layout. The PCB software will apply a list of rules to the PCB, verifying that all the connections in the netlist are made, that there is sufficient clearance between all the traces, and so on. These rules can be modified based on the specific board house requirements. As stated above, it is useful to run the DRC even before all the traces have been routed — this can identify clearance or other issues that might require substantial re-routing or a different approach.

If the user has waited until all the routing is done before running the DRC, the list of violations can be daunting. However, it is often the case that many if not all of the violations represent issues that may prevent the board from operating as wished. Whenever possible, all DRC violations should be rectified before fabrication.

6.4.7 Preparation for Fabrication

LAYOUT REVIEW

Once the board has passed DRC, the electrical connectivity and basic requirements for manufacturability have likely been satisfied. However, the design may benefit from an additional review pass. Turn off all the layers but one copper layer and examine the traces — often simplifications in routing will be apparent without the distractions of the other copper layers. For example, a trace can be moved to avoid going between two closely spaced pins. Densely spaced traces could be spaced farther apart. There may be opportunities to reduce vias by routing traces primarily on one layer even if that now means both vertical and horizontal travel. Repeat the exercise for all the copper layers.

Review the mechanical aspects of the board as well, including the proximity of traces to hardware. If your prototype PCB does not have a solder mask, traces that run underneath components such as crystals in a conductive case or too close to mounting hardware can form a short circuit. An insulator must be provided or the trace can be re-routed.

GENERATING OUTPUT FILES

Once the PCB design is complete, the complete set of design description files can be generated for producing the PCB. These are:

Copper layers — One file per copper layer. These are known as *Gerber files* and were

text files of commands originally intended to drive a *photoplotter*. Gerber was the primary manufacturer of photoplotters, machines that moved a light source of variable width (apertures) from one location to another to draw patterns on photographic film. While photoplotters have been replaced by digital technology, the format used by Gerber has been standardized as RS-274X and is universally used except by PCB software tied to a specific manufacturer. RS-274X is related to RS-274D ("G-Code") used by machinists to program CNC machinery but is an *additive* description (essentially saying "put copper here"), rather than describing the movements of a tool to remove material. A program is thus required to translate between Gerber and G-Code if a CNC machine is used to make a PCB by mechanically removing copper.

Drill file — The file containing the coordinates and drill sizes for all the holes, plated or not. Also called the *NC* or *Excellon* file, some board houses may require a specific format for the coordinates, but these are usually available to be set as options in the PCB program. There is only one drill file for a PCB, since the holes are drilled from one side. (Exotic options such as buried vias will require more information.) Like RS-274X apertures, the drill file will generally contain a drill table.

Silkscreen — Also in RS-274 format. Some board houses can provide silkscreen on both sides of the board, which will require two files.

Solder mask — The solder mask file is used by the board house to create the solder mask. One file per side is required.

A Gerber preview program such as *Gerbv* (**gerbv.gpleda.org**, open source, *Linux*, Mac OS X) or *GC-Prevue* (**www.graphicode.com**, *Windows*) can be used to review the trace layout Gerber files. This is a good test — the board house will make the boards from the Gerber files, not the PCB design file. Gerber previewers can import the copper layers, silkscreen and drill files to verify they correspond and make sense.

Any of the layers can usually be printed out within the PCB program (and/or Gerber preview program) for reference and further inspection.

In addition to files for PCB production, PCB layout programs can also generate assembly diagrams, and in some cases can provide 3D views of what the assembled board will look like. These can be useful for documentation as well as verification of mechanical issues such as height clearance.

Sending the files to the PCB manufacturer or board house and ordering PCBs is explained on the manufacturing website or a customer service representative can walk you through the process. Some firms accept sets of files on CD-ROM and may also offer a design review service for first-time customers or on a fee basis.

6.5 References and Bibliography

SIMULATION REFERENCES

Allen, J. Wayde "Gain Characterization of the RF Measurement Path," *NTIA Report TR-04-410* (Washington: US Department of Commerce: 2004). Available as **www.its.bldrdoc.gov/pub/ntia-rpt/04-410/04-410.pdf**.

Hayward, W. W7ZOI; R. Campbell, KK7B; and B. Larkin, W7PUA, *Experimental Methods in RF Design*, 2nd ed (ARRL: Newington, 2009).

P. Horowitz and W. Hill, *The Art of Electronics*, 2nd ed (Cambridge: Cambridge University Press, 1989).

Kundert, K., "Introduction to RF Simulation and its Application," **http://icslwebs.ee.ucla.edu/dejan/researchwiki/images/3/30/Rf-sim.pdf**

NXP Semiconductor: **www.nxp.com/models/bi_models/mextram/**

Newkirk, D. WJ1Z, "Math in a Box, Transistor Modeling, and a New Meeting Place," Exploring RF, *QST*, May 1995, pp 90-92.

Newkirk, D. WJ1Z, "Transistor Modeling with ARRL Radio Designer, Part 2: Optimization Produces Realistic Transistor Simulations," Exploring RF, *QST*, Jul 1995, pp 79-81.

Newkirk, D. WJ1Z, "ARRL Radio Designer as a Learning (and Just Plain Snooping) Tool," Exploring RF, *QST*, Nov 1995, pp 89-91.

Newkirk, D. WJ1Z, "An ARRL Radio Designer Voltage Probe Mystery: The 3-dB Pad that Loses 9 dB," Exploring RF, *QST*, Jan 1996, pp 79-80.

Newkirk, D. WJ1Z, "ARRL Radio Designer versus Oscillators, Part 1," Exploring RF, *QST*, Jul 1996, pp 68-69.

Newkirk, D. WJ1Z, "ARRL Radio Designer Versus Oscillators, Part 2," Exploring RF, *QST*, Sep 1996, pp 79-80.

Newkirk, D. W9VES, "Simulating Circuits and Systems with *Serenade SV*," *QST*, Jan 2001, pp 37-43.

"The Spice Page," (**http://bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/**). The home page for *SPICE*.

Tuinenga, Paul W., *Spice: A Guide to Circuit Simulation and Analysis Using Pspice*, 2nd ed (New York: Prentice-Hall, 1992).

Vladimirescu, Andrei, *The SPICE Book* (New York: John Wiley and Sons, 1994).

Silver, W. NØAX, "Hands-On Radio Experiments 83-85: Circuit Simulation," *QST*, Dec 2009 through Feb 2010.

PCB CAD REFERENCES

Analog Devices, *High Speed Design Techniques*, Analog Devices, 1996.

Johnson, Howard, and Graham, Martin, *High Speed Digital Design: A Handbook of Black Magic*, Prentice Hall, 1993.

Ott, Henry, *Electromagnetic Compatibility Engineering*, Wiley Press, 2009.

Pease, Robert A, *Troubleshooting Analog Circuits*, Butterworth-Heinemann, 1991.

Silver, W. NØAX, "Hands-On Radio Experiments 107-110: PCB Layout," *QST*, Dec 2011 through Mar 2010.