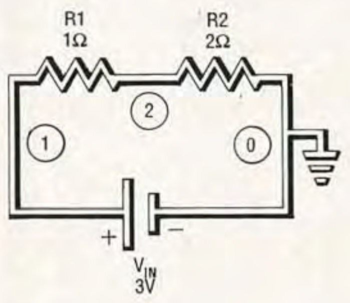
SPICE SPICE SPICE



```
=  *Resistor divider circuit
   VIN 1 03.0volt
   R1  1 21.0ohm
   R2  2 02.0ohm
   .END
```

TJ BYERS

Breadboards are giving way to PC-based circuit simulation.

IF YOU'RE LIKE MOST ELECTRONICS hobbyists, you love to design and build circuits. Unfortunately, one of the hazards of the hobby is that many projects never get beyond the breadboarding stage because of design or debugging problems.

Sure, you can spend countless hours troubleshooting a circuit using a plethora of test instruments, but wouldn't it be nice if you didn't have to touch a component until you were ready for the final assembly? Now you can, using the power of your PC and a growing number of circuit-simulation programs.

Circuit-simulation programs are not new either to the computer or to the PC. What is new is that now these programs are affordable. Simulation programs that sold for $20,000 ten years ago can be obtained today for less than $100, and that low cost makes it cheaper and faster to simulate a circuit than to breadboard it.

Another advantage of circuit-simulation programs is that they give us the ability to do "what-if?" simulations. For example, what if you were to substitute a 0.01 μF capacitor for the 0.015 μF unit specified in a circuit design? Using circuit-simulation software, you could learn the answer in seconds.

This is the first article in a two-part series that will look at circuit-simulation programs for personal computers. This part discusses a venerable analog simulation program called SPICE, and the second part discusses a powerful digital simulation program called SUSIE. Our bias is toward the IBM family, but versions of the programs we'll discuss are available for other platforms; we'll discuss prices and availability later.

The SPICE of life

SPICE (simulation program with integrated circuit emphasis) is far and away the most popular analog circuit-simulation program in use today. SPICE was developed by the University of California at Berkeley in the late 60's, and was released to the public in 1972.

Over the years, SPICE has gone through many upgrades. The most important change came with SPICE2, in which the kernel algorithms were upgraded to support advanced integrated system methods, many of which relate to IC performance. SPICE2 has all but replaced SPICE1 as the SPICE of choice by industry. SPICE2 has been ported to numerous types of computers and operating systems, including the Macintosh and the IBM PC family, and is sold under several different brand names by several software companies.

Recently, SPICE2 was upgraded to SPICE3. In the new version, the program was converted from FORTRAN to C for easier portability, and several devices were added to the program library, including a varactor, semiconductor resistor, and lossy RC transmission lines. However, the kernel algorithms were not changed, and the added components aren't that significant because SPICE2 can simulate all the devices built into SPICE3 using external device modeling. But more on that later.

Creating a SPICE circuit

One reason for SPICE's popularity lies in its programming simplicity. Unlike many circuit-simulation programs, which require special programming tools to create simulation files, all you need to produce a SPICE file is an ASCII text editor. Moreover, because SPICE files are pure ASCII, files created on a Macintosh are identical to those used by a PC and a Sun workstation. That file compatibility allows you to exchange SPICE designs among vastly different computers without file modification.

SPICE files are nothing more than a list, called a *netlist*, of the components used in the circuit; Table 1 summarizes the contents of a netlist. When describing a SPICE component, it's a simple matter of listing the component's value and its physical connections to other components in the circuit. If you wish, you may optionally include qualifying parameters such as temperature coefficient, tolerance, etc., but they're not required.

TABLE 1—NETLIST CONTENTS

Title Line
Analysis Control Statement
Output Control Statement
Circuit Topology
Signal Or Power
.END Statement

To illustrate how a SPICE file is written, refer to the circuit in Fig. 1. That circuit is an RLC bridge-T bandpass filter, often used in audio equalizers. For the sake of argument, let's bias the network at 10-volts DC and specify a 1-volt AC input signal. The design questions are: What is the bandpass frequency, what is the bandwidth, and what are the AC and DC output voltages? Let's see how we could use SPICE to solve the problem.

After drawing a schematic of the circuit, you label each component. SPICE nomenclature is consistent with accepted schematic labels—R represents a resistor, C a capacitor, L a coil, Q a transistor, and so forth.

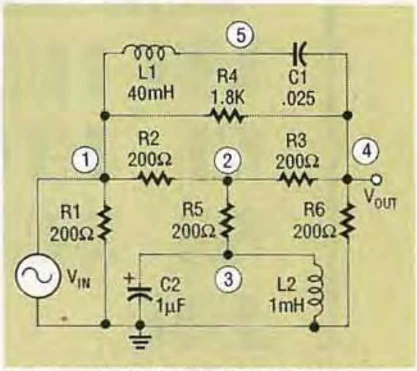


FIG. 1—SAMPLE CIRCUIT 1. This is a Bridge-T bandpass filter; the circled numbers represent the nodes that SPICE will analyze.

Next, you must number the circuit nodes. A node is any point where two or more wires connect. There is no required order to the numbering; you can assign node numbers at random, and even skip numbers. The exception is ground, which has a reserved value of 0. Just be wary that you don't assign separate numbers to opposite ends of a wire because SPICE will think you're talking about two different nodes and not make the connection.

The next step is to create a list of all components by label, node, and value (in that order), and store the list in an ASCII file with a .CIR extension (FILTER.CIR). There are three restrictions on the file format: The file must begin with a file name, end with an .END statement, and it must contain only uppercase letters. SPICE syntax is very inflexible (what programming language isn't?), and unless each component is described explicitly with all the parameters in the right order, the simulation won't work. The SPICE file for the circuit in Fig. 1 is shown in Listing 1.

**LISTING 1
BRIDGE-T BANDPASS
FILTER NETLIST**

```
BRIDGE-T BANDPASS FILTER
.AC DEC 20 1KHZ 10KHZ
.PRINT AC V(4) VDB(4)
R1 1 0 200
R2 1 2 200
R3 2 4 200
R4 1 4 1.8K
R5 2 3 200
R6 4 0 200
C1 5 4 .025UF
C2 3 0 1UF
L1 1 5 40MH
L2 3 0 1MH
VIN 1 0 DC 10V AC 1V
.END
```

Although we listed the resistors in ascending order, there's no set order to a netlist. All voltage sources, resistors, and other components may be listed at random, because SPICE first reads the netlist in its entirety, organizes the contents to its liking, and only then compiles the circuit into a run-time file. The process is quite similar to the way a Pascal or C compiler compiles an ASCII source file into an executable program.

Circuit analysis and simulation

After SPICE compiles a run-time version of the file, it uses its kernel algorithms to analyze the circuit, first for DC parameters, and then for AC performance, if requested. The result is a computer simulation of the circuit under the specified operating conditions. The simulated circuit parameters are then recorded in an ASCII file. The SPICE simulation of our circuit is shown in Listing 2.

SPICE simulation occurs in stages. First SPICE analyzes the netlist to see if there are any errors. It can't check for all errors, simply because it can't read your mind, but it does check the syntax of every entry, and makes sure that every node has two or more connections. If an error occurs, an error message is inserted below the line that caused the error, which greatly simplifies the debugging process.

If no errors are detected, SPICE does a DC analysis of the circuit, calculating the bias voltage for every node, and the total current drain from the power sources. SPICE also measures the total power dissipation of the circuit. From this analysis, we find that the DC output is 2.5 volts.

Next SPICE does an AC analysis of the circuit using simulated signal generators. SPICE is quite adept in this area, in that it's able to emulate several kinds of signal generators, including sine, sweep, pulse, and others.

For our tests we need a sweep generator, which we describe in the second line of the netlist. The .AC label is a special SPICE command called a control statement. (Note that all SPICE control statements begin with a period.) The statement says that we want a sweep generator with a logarithmic sweep of 20 points per decade (DEC 20) beginning at 1 kHz and ending at 10 kHz. We could just as easily have selected a linear (LIN) or octave (OCT) sweep rate, and set the sweep to occur between any two frequencies, ranging from sub-audio to gigahertz.

Next we need to monitor the output signal. This is done using the .PRINT control statement on line three. .PRINT simulates a variety of measuring instruments, including several types of voltage and current meters. For our simulation we need both an RMS AC voltmeter, V(4), and a decibel voltmeter, VDB(4). The numbers in parentheses indicate which node is to be measured. As it turns out, our simulation looks at only one node, but we could have specified any number of nodes.

The .PRINT statement generates a table of values for each point on the generator's sweep. The results of our circuit simulation are shown in the AC ANALYSIS section of Listing 2. Notice that the filter peaks at 5 kHz with a voltage output of 750 mV and an attenuation factor of 12 dB. For the final answer, look at the VDB(4) portion of the table, which shows that the signal is down 6 dB at 4.5 kHz and 5.5 kHz, giving the filter a bandwidth of 1 kHz. Problem solved.

SPICE could just as well have provided more measurements, including transient response time, noise factor, distortion, and phase shift of any or all nodes in the circuit, over a wide range of operating conditions and temperatures. That type of analysis is extremely valuable for seeing how the circuit behaves under extreme environmental conditions.

LISTING 2—SPICE OUTPUT

```
BRIDGE-T BANDPASS FILTER
SMALL SIGNAL BIAS SOLUTION TEMPERATURE = 27.000 DEG C
(NODE) VOLTAGE
(1) 10.00 (2) 4.1667 (3) .00 (4) 2.50 (5) 10.00
  VOLTAGE SOURCE CURRENTS
  NAME      CURRENT
  VIN       -8.333D-02
  TOTAL POWER DISSIPATION 8.33D-01 WATTS
   FREQ                       V(4)                    VDB(4)
1.00000E+03              2.509E-01              -1.201E+01
1.12202E+03              2.511E-01              -1.200E+01
1.25893E+03              2.514E-01              -1.199E+01
1.41254E+03              2.519E-01              -1.198E+01
1.58489E+03              2.524E-01              -1.196E+01
1.77828E+03              2.532E-01              -1.193E+01
1.99526E+03              2.544E-01              -1.189E+01
2.23872E+03              2.561E-01              -1.183E+01
2.51189E+03              2.587E-01              -1.174E+01
2.81838E+03              2.629E-01              -1.160E+01
3.16228E+03              2.706E-01              -1.135E+01
3.54813E+03              2.862E-01              -1.087E+01
3.98107E+03              3.254E-01              -9.752E+00
4.46684E+03              4.662E-01              -6.628E+00
5.01187E+03              9.983E-01              -1.499E-02
5.62341E+03              4.882E-01              -6.228E+00
6.30957E+03              3.303E-01              -9.622E+00
7.07946E+03              2.879E-01              -1.081E+01
7.94328E+03              2.714E-01              -1.133E+01
8.91251E+03              2.634E-01              -1.159E+01
1.00000E+04              2.589E-01              -1.174E+01
```

Device modeling

SPICE can also analyze circuits that use diodes, transistors, and other active devices. Of course, you must provide substantially more information about active components. To avoid redundancy, SPICE lets you describe a device in detail, using a method called device modeling, and then reuse the same device specification at will.

With device modeling, the equations needed to describe the device are in a library within the SPICE program. The only thing you have to do is fill in the blanks. After you define a device, you can use it as many times as you wish in a netlist.

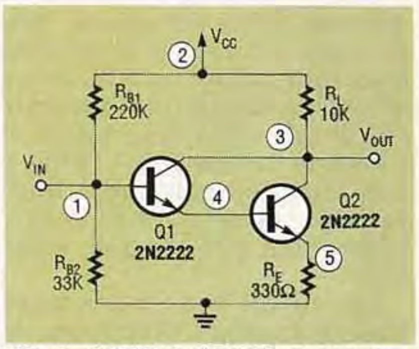


FIG. 2—SAMPLE CIRCUIT 2. This is a Darlington amplifier; like Fig. 1, the circled numbers represent the nodes that SPICE will analyze.

Let's use the circuit in Fig. 2 as an example. In this circuit two 2N2222 transistors are cascaded to create a Darlington amplifier. Resistors R_{B1} and R_{B2} are the

LISTING 3—DARLINGTON AMP DEVICE MODELING

```
DARLINGTON AMPLIFIER
VCC 2 0 DC 10V
RB1 1 2 22OK
RB2 1 0 33K
RL 2 3 10K
RE 5 0 330
Q1 3 1 4 QN2222
Q2 3 4 5 QN2222
.MODEL QN2222 NPN(IS=1.9E-14 BF=150 VAF=100
+IKF=.175 ISE=5E-11 NE=2.5 BR=7.5 VAR=6.38
+IKR=.012 ISC=1.9E-13 NC=1.2 RC=.4 XTB=1.5
+CJE=26PF TF=.5E-9 CJC=11PF TR=30E-9
+KF=3.2E-16 AF=1.0)
.END
```

biasing resistors, R_L is the load resistor, and R_E is the emitter resistor; V_{IN} and V_{OUT} are the input and output voltages, respectively. Examining the circuit, it is apparent that the SPICE netlist must contain four resistors, two transistors, and six nodes (including ground). A complete netlist is shown in Listing 3.

Notice that the transistors are treated exactly like resistors or capacitors, with the node connections listed first, and then the part value. In this case, the part value refers to the QN2222 transistor model at the end of the netlist. The .MODEL label is a control statement that tells SPICE to plug these values into its transistor equations. For each different type of transistor (or other modeling device) used in a circuit you must provide a separate model description.

Subcircuits

To model a complex device, you describe it in a *subcircuit*. Subcircuits can model everything from transformers to IC's. An example of a subcircuit is shown in Fig. 3; the netlist describing it is shown in Listing 4. In Listing 4, note that normally subscripted variables aren't subscripted, to make them into program variables.

The circuit is an active bandpass filter using a 741 op-amp. As in the case of a regular circuit, you begin by drawing the schematic, labeling the components, and assigning numbers to the circuit nodes. Note that the IC is drawn in conventional style as a single element with five external connections.

The description of the 741 begins with the line .SUBCKT UA741 and ends with .ENDS. In the 741 subcircuit netlist you'll find the usual assortment of resistors and capacitors used to specify input and output impedance characteristics. For example, R_L and C_L describe the resistive and capacitive factors of the 741's output pin.

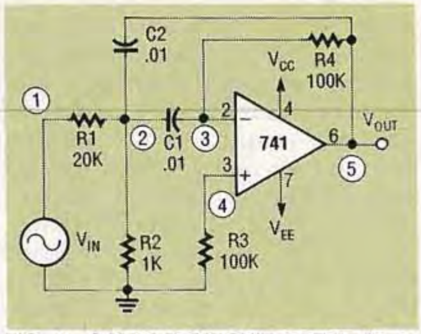


FIG. 3—SAMPLE CIRCUIT 3. This is an active filter built around a 741 op-amp. Again, the circled numbers represent the nodes that SPICE will analyze.

You may find that diodes and transistors are used to describe the inner workings of some subcircuits, but more often than not voltage and current sources are used to emulate the collective actions of these components. In our model, for example, V_{OFST} is a voltage source that sets the IC's input offset voltage at one millivolt, and I_{BN} is a current sink that sets the bias current of the inverting input at 100 nanoamps. Further, notice that the non-inverting input bias current, I_{BP}, is set at 80 nanoamps, giving the op amp an offset bias current of 20 nanoamps, a typical value for that IC.

The node numbering within a subcircuit is independent of the node numbering of the main netlist, and in no way do the two sets of numbers conflict. Furthermore, you can nest subcircuits; that is, you can call another subcircuit into your subcircuit as an element of its design. As with device modeling, you only have to define a subcircuit once, but you must have a separate subcircuit for each different device type. For example, the 741 subcircuit model will not work for an LM3900 quad op-amp.

Digital modeling

SPICE can also perform digital simulations. As a rule, you're better off using a purely digital simulator to measure time sequences and logic levels, but there are times when the two technologies come together. Analog-to-digital (A/D) converters are a prime example; the simulation must run on an analog simulator.

Performing digital simulations with SPICE requires special modeling because of the sheer number of subcircuit elements. The obvious way to model a digital device would be to describe it in a subcircuit using the actual transistor circuit. However, that approach uses large amounts of

USING SCHEMATIC CAPTURE SOFTWARE TO CREATE NETLISTS

If you find creating SPICE netlists for large circuits more than you can handle, or if you just want to make life with SPICE a little easier, there are several schematic-capture programs that can generate SPICE netlists, including the popular programs, Schema and OrCAD.

The procedure is as simple as drawing a schematic using the capture software, then requesting a netlist. Using the points that you identify as nodes, the capture software generates a SPICE netlist with the node connections and device values in the proper order.

You'll probably have to touch up the netlist, adding the opening and closing lines. In addition, most schematic-capture software lacks device and subcircuit modeling, which means that you'll have to go into the netlist and manually insert the missing lines and models. Even so, that process is better than doing the whole thing manually, and it reduces the chance for error.

One exception is SpiceNET, a schematic-capture program from Intusoft ($295) that does generate a functional SPICE netlist. Circuits created by the program can also be used as subcircuits and added to model libraries. **R-E**

LISTING 4
OP-AMP BANDPASS FILTER SUBCIRCUIT

```
OP AMP BANDPASS FILTER
.AC DEC 20 1 100KHZ
.PRINT AC V(5)
X1 3 4 5 7 6 UA741
R1 1 2 20K
R2 0 2 1K
C1 2 3 .01U
C2 2 5 .01U
R4 3 5 100K
R3 0 4 100K
VEE 6 0 -15
VCC 7 0 15
V1 1 0 SIN 0 1 AC 1
-IN
+IN
OUT
VCC
VEE
.SUBCKT UA741 2   3   6   7   4
*PINOUTS    -IN +INOUTVCCVEE
RP 4 7 10K
RXX 4 0 10MEG
IBP 3 0 80NA
RIP 3 0 10MEG
CIP 3 0 1.4PF
IBN 2 0 100NA
RIN 2 0 10MEG
CIN 2 0 1.4PF
VOFST 2 10 1MV
RID 10 3 200K
EA 11 0 10 3 1
R1 11 12 5K
R2 12 13 50K
C1 12 0 13PF
GA 0 14 0 13 2700
C2 13 14 2.7PF
RO 14 0 75
L 14 6 30UHY
RL 14 6 1000
CL 6 0 3PF
.ENDS
.END
```

```
******* 2/19/90 ******* IS SPICE  1.41 12/12/87*******20:30: 6*****

BANDPASS FILTER

****     AC ANALYSIS                        TEMPERATURE =   27.000 DEG C

****************************************************************************



 LEGEND:

*: V(5)
+: VDB(5)

    FREQ      V(5)

(*)-------------- 1.000D-01       3.162D-01      1.000D+00      3.162D+00   1.000D+01
                        - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

(+)------------- -1.000D+01      -5.000D+00      0.000D+00      5.000D+00   1.000D+01
                        - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 1.000D+03   4.773D-01  .          +   .    *         .              .              .
 1.100D+03   5.978D-01  .              .+      *      .              .              .
 1.200D+03   7.661D-01  .              .        +  *  .              .              .
 1.300D+03   1.016D+00  .              .              X              .              .
 1.400D+03   1.406D+00  .              .              .   *   +      .              .
 1.500D+03   1.996D+00  .              .              .       *      .  +           .
 1.600D+03   2.481D+00  .              .              .           *  .       +      .
 1.700D+03   2.182D+00  .              .              .         *    .     +        .
 1.800D+03   1.641D+00  .              .              .     *      + .              .
 1.900D+03   1.260D+00  .              .              .  *  +        .              .
 2.000D+03   1.013D+00  .              .              X              .              .
 2.100D+03   8.475D-01  .              .          + * .              .              .
 2.200D+03   7.296D-01  .              .     +   *    .              .              .
 2.300D+03   6.420D-01  .              .  +      *    .              .              .
 2.400D+03   5.743D-01  .              .+     *       .              .              .
 2.500D+03   5.206D-01  .            + .     *        .              .              .
 2.600D+03   4.768D-01  .          +   .    *         .              .              .
 2.700D+03   4.404D-01  .        +     .   *          .              .              .
 2.800D+03   4.097D-01  .     +        .  *           .              .              .
 2.900D+03   3.834D-01  .    +         . *            .              .              .
 3.000D+03   3.605D-01  .  +           . *            .              .              .
                        - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

FIG. 4—SPICE GRAPHIC. This crude graph uses ASCII characters, so it prints on any printer.

computer memory and requires lengthy run times, even for simple circuits.

A better way is to use threshold logic theory. With this method, the input gates of a device appear as analog voltages that are multiplied by weighting constants, and the resulting analog voltages are compared to a threshold value. If the sum is greater than the threshold, the output of the gate is treated as a logical one; if the sum falls below the threshold, the output is treated as a logical zero. The weighted values and threshold point are usually determined by polynomial algorithms. The problem is that the algorithms aren't easy to work with, and are beyond the expertise of most SPICE users.

Fortunately, however, there's a huge inventory of ready-made SPICE models for many different types of components, including transistors, optoelectric devices, and analog and digital IC's. Most versions of SPICE come with a library of popular components, and you can usually obtain additional libraries at extra cost. Device manufacturers are another source of SPICE models. For not-so-popular or oddball devices, you can buy specialty models from third-party vendors, or roll your own in a subcircuit. The maker of IsSPICE, Intusoft, publishes a free newsletter that contains information and modeling tips for various types of electronic devices.

Plotting SPICE's output

Like most simulation programs, SPICE generates a lot of data, and analyzing that data can be a chore. Like the netlist, all SPICE output files are in ASCII, which means you can transport them from one computer to another without conversion, or make a hardcopy of the file using just about any printer. Data is stored in the output file in the same order that SPICE performs its operations: the netlist, followed by the DC analysis, followed by the AC analysis.

There are several ways to display SPICE output in graphical form. The easiest is to add a .PLOT control statement to your netlist. SPICE then provides you with a tabular listing, similar to that generated by .PRINT, plus a crude graphical representation, as shown in Fig. 4. If you request more than one measurement within the same .PLOT statement on your netlist, you get a multi-line graph, which is handy for comparing two or more sets of measurements.

However, .PLOT graphics leave a lot to be desired. Fortunately, however, SPICE's tabular format is accepted by most database, spreadsheet, and graphics programs, including Lotus 1-2-3. The best way to prepare a SPICE output file for data import is to use a text editor to remove everything but the desired tables, and then import the data into the desired program. Figure 5 shows a 1-2-3 plot of the data in Fig. 4.

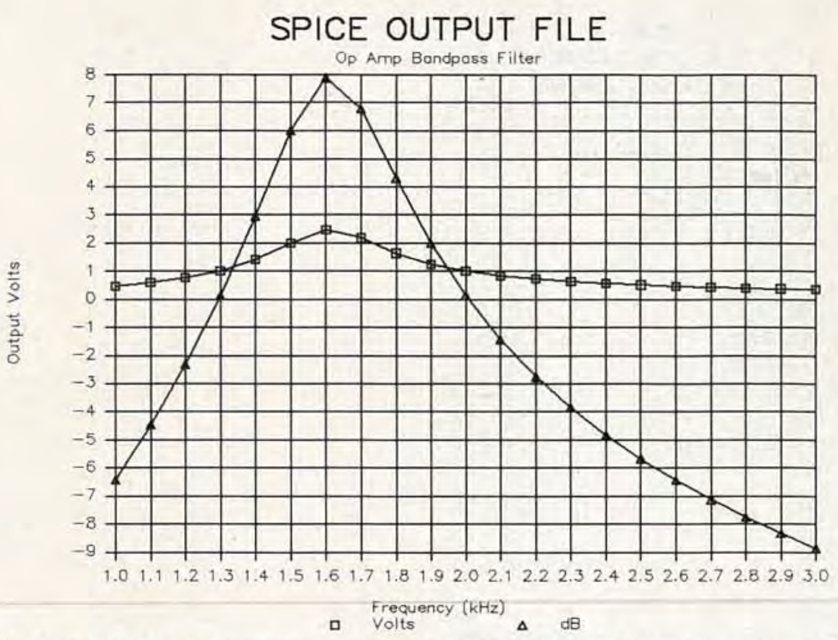


FIG. 5—1-2-3 PLOT. It's simple to import SPICE data into Lotus 1-2-3 and get a decent plot.

Choosing a SPICE program

When choosing a SPICE program, there are several factors to consider, not the least of which is cost. A SPICE package can cost anywhere from $100 to more than $20,000.

The cheapest SPICE simulation package that we know of is IsSpice from Intusoft, which sells for $95. An accessory for IsSpice that lets you draw schematics and create netlists is shown in Fig. 6. By comparison, the popular PSpice program from MicroSim goes for $950. The digital-simulation option for PSpice is shown in Fig. 7. The reason for the large difference in price between the two is due mostly to the number of device models included. IsSpice comes with only a handful of transistor models, whereas PSpice comes with a transistor library containing more than 1500 devices, plus numerous other device models.

Program prices also vary according to the type of computer the program runs on. The Is-

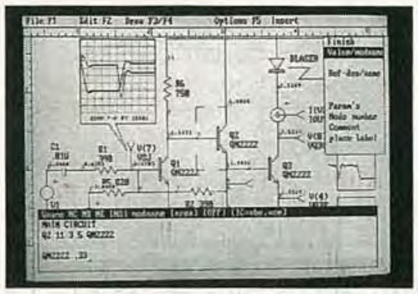


FIG. 6—SPICENET ADD-ON FOR IsSpice.

Spice and PSpice prices mentioned above are for the 8088 version. If you want to use the additional power offered by a 286 or 386 PC, corresponding versions of the program will cost more. PSpice for the Macintosh costs a hefty $1,450.

The type of PC also determines the size of the circuit the SPICE program can support. For example, the PC-only version of SPICE can only simulate circuits of fewer than 200 nodes (about 100 transistors, including subcircuits). For larger simulations, you need to buy a more powerful (and more expensive) version of the program.

Most SPICE packages offer a variety of utility programs that make SPICE easier to use. The most popular utilities are SPICE

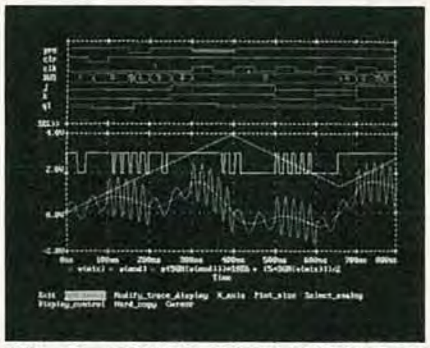


FIG. 7—DIGITAL SIMULATION FOR PSpice.

editors, Monte Carlo simulators, and graphics processors. But get ready to raise the ante again. For example, a fully-loaded IsSpice package goes for $790, and an equivalent PSpice package tips the scales at $1,750. Fortunately, most SPICE makers have demo packages available so you can try before you buy.

Next time we'll look at the world of digital circuit simulation, and examine a digital simulator called SUSIE. **R-E**

ITEMS DISCUSSED

- IsSpice ($95), Intusoft, P.O. Box 6607, San Pedro, CA. 90734. (213) 833-0710.
CIRCLE 41 ON FREE INFORMATION CARD

- PSpice ($950), MicroSim Corp., 20 Fairbanks, Irvine, CA 92718. (714) 770-3022.
CIRCLE 42 ON FREE INFORMATION CARD