

A Beginner's Guide to SPICE3 and PSpice

Hemanshu Roy Pota
School of Electrical Engineering
Australian Defence Force Academy
Canberra ACT 2600
pota@adfa.oz.au

June 12, 2005

Abstract

This short note introduces SPICE3 and PSpice programs. This note is brief and a detailed discussion can be found in various books on SPICE3[6] and PSpice [1, 5, 3].

1 Introduction

SPICE3 is an analog circuit simulation program. Almost all the commercial circuit simulation packages have SPICE3 as their simulation engine. Our department has the original program along with an extension of the analog package to include digital simulation; it's called `xspice`.

In addition to the full blown SPICE3 a port of SPICE3 to microcomputers, called PSpice, is also available. MicroSim, the company which markets PSpice has its own ftp site and also a www site (<http://www.microsim.com>). You can ftp the latest PSpice evaluation version from our site, <ftp://evans.adfa.oz.au/pub/staff/hrp/71dlabe.exe> or the microsim site: <ftp://ftp.netcom.com>, login anonymous, cd /pub/mi/microsim/Eval_Versions, get the files winevals.txt and README.TXT, read them, and then download what you want. Have a look at other directories and you will find many other interesting things.

PSpice with its Schematics capture program is reasonably easy to use. In the following we will mainly discuss about using SPICE3 on our department unix network. Most of the things are identical for PSpice and SPICE3 so reading this note will also benefit all those who are intending to use PSpice.

The first part of this brief tutorial aims at getting you started in using SPICE3. The original SPICE3 package is almost free and consequently there isn't a good documentation accompanying the package. The manual [4] is available in the department but one needs to really struggle with it even to find out where to start. There are some good introductory texts [6] available but overall it is very difficult to get started; once you can start, the

rest is easy. I myself had a great difficulty in starting but now I feel amply rewarded for my efforts in learning SPICE3. I couldn't have understood many interesting things about op-amp's internal working without SPICE3 simulations. My intention in this note is to get you started from scratch.

I assume that you are sitting in front of an xterminal and typing the commands as we progress. So let's begin. And before you begin it's always a good idea to read our `info` pages. Type `info spice`, at the Unix prompt and see what's there. In the following all the text in typewriter type (bold and uniform spacing) is either a command or some technical jargon.

2 A Simple RLC Circuit Analysis

Figure 1 is a schematic of a simple RLC circuit with a voltage source.

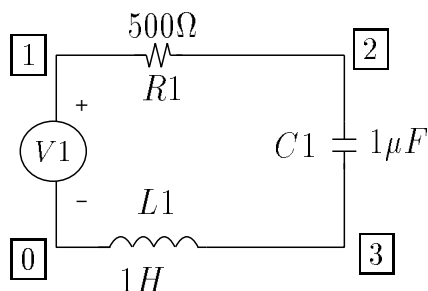


Figure 1: A Simple RLC Circuit

A SPICE3 script to analyse the circuit in Figure 1 follows.

```
Simple example for spice3 tutorial
*Filename: tute-ex1.cir (Modified: 7 August 1996)
*Sources
V1 1 0 DC 10V AC 1V
*Network Elements
R1 1 2 500
C1 2 3 1uF IC=2V
L1 3 0 1H IC=1mA
*Control Statements (suitable for batch processing)
.op
.DC V1 5 10 1
.AC DEC 10 .1 1000
.TF V(2,3) V1
*tf command not available in the interactive spice
.PZ 1 0 3 0 VOL PZ
.TRAN 0.1ms 20ms UIC
```

```

*use only lowercase within the .control .endc construct
.control
op
dc v1 5 10 1
pz 1 0 3 0 vol pz
tran 0.1ms 20ms uic
plot v(2)-v(3) xlabel 'Time'
+ ylabel 'Vc' title 'RLC Circuit'
*first character '+' means a continuation line
ac dec 10 0.1 1000
plot vm(2,3)
set hcopydevtype = postscript
*hardcopy tempfile.ps vm(2)-vm(3)
.endc
.end

```

The first thing to do is create a file `tute-ex1.cir` using your favourite editor, then type the above script in it, and save it; you can leave out lines starting with '*', they are comment lines. In the following discussion this file will be referred to as the script or a spice netlist file. Now get into the directory you have saved the above file `tute-ex1.cir` and invoke SPICE3 by the command `xspice`. SPICE3 can also be invoked by the command `spice3`, but for some reason `xspice` has a better help feature. So for the time being let's say you have typed the command `xspice` at the Unix prompt.

Now we have to read the script file into the spice program. To do so type at the spice prompt (which looks something like `xspice 1-->`) `source tute-ex1.cir`. At this stage the program will complain if there are any syntactical errors in the input script file. For our example script all should go smoothly. The control commands between `.control` & `.endc` will be executed immediately and you will see two plots come up on your screen. As a matter of fact we have performed five different analyses but plotted results from only two of the analyses. Let's now see how we can retrieve the data from other analyses and plot them. Remember that in the interactive mode SPICE3 accepts only lowercase letters. As we progress lots of x-windows with plots will be popping up on your screen. Feel free to click in the `quit` box to get rid of them.

2.0.1 Pole-Zero Analysis

The line starting with `pz` in the script tells the program to get the poles and zeros of the transfer function between the first mentioned pair of nodes (1,0) and the second mentioned pair of nodes (3,0). `VOL` means that the transfer function between the voltages is considered, and the final `pz` tells it to find both poles and zeros.

To see poles and zeros type `setplot`, `xspice` will list all the different plots available for selection. Remember the command `setplot` to find out what all data is stored by `xspice`. Later on you will learn to source more than one file and would want to switch between

the outputs generated by different ‘runs’ and that’s when this command will come in real handy; SPICE3 keeps the data from all the different analysis it performs. Type `pz4` at the ‘?’ prompt. To see the poles and zeros type `print all`. You should see two poles at $-250 \pm j968.2$ and two zeros at the origin. I wonder if we can trust SPICE3 with this difficult job; do a quick calculation and see what is it that SPICE3 has actually done.

2.0.2 Transient Response

The control line `tran 0.1ms 20ms uic` tells SPICE3 to perform a transient analysis with step-size 0.1ms and up to 20ms, using initial conditions. In our problem the capacitor has an initial voltage of 2V and the inductor has a current of 1mA flowing at $t = 0$. The transient analysis is performed with `V1` as a step voltage of 10V. The value of 10V follows ‘DC’ in the script in the line starting with ‘`V1`’. Actually ‘DC’ on that line is telling SPICE3 to use the value following it for transient analysis. The value following ‘AC’ is not used for this analysis. Specifying ‘DC 10V’ is an easy way to set up the transient analysis. Alternatively actual step waveform can be associated with `V1`. Not only a step but other general waveforms like pulse, sin, exp, piecewise-linear, etc. are available. See the manual [4] for an exact syntax.

To look at the transient analysis plots type `setplot` and then select `tran5`. Now whatever plot command you give will plot the data from the transient analysis. Type `plot v(2)-v(3)` to see a plot of the transient voltage across the capacitor and to see the current in the circuit type `plot v1#branch`. You can also plot functions of all the vectors available. To see which all vectors are available type `display all`. To see how functions of vectors can be plotted try the commands: `plot (v(1)-v(2))/0.5K` (current through the resistor), `plot ((v1#branch)^2)*500` (the power dissipated in the resistor). To see what all functions you can use with vectors type `help functions`. To get a hardcopy of the plot first type `set hcopydevtype = postscript` then suppose you want to get a hardcopy of the voltage across the capacitor, type `hardcopy filename.ps v(2)-v(3)`. You can print the resulting .ps file using the `lpr` command. The file will be in the current directory. You can include these commands in the .cir file if you like; see the script `tute-ex1.cir`. For people who want a .eps file there is a program `pstoepsi` which inserts a bounding box in the file; the correct syntax to call it is: `pstoepsi file.ps file.eps` (the first file should have a .ps extension), file.eps has a boundingbox inserted in it.

2.0.3 DC Analysis

The command `dc v1 5 10 1` varies the voltage `V1` from 5V to 10V in the steps of 1V and performs a DC analysis. The results here will not be very surprising. To see what SPICE3 has done type `setplot` and then choose `dc3`. To see the capacitor voltage type `plot v(2)-v(3)`. Is it what you expect? DC analysis looks very uninteresting here but it’s very important in finding the dc operating point for circuits with transistors.

2.0.4 AC Analysis

The command `ac dec 10 .1 1000` tells SPICE3 to vary all the sources which have ‘AC’ appearing in their declaration line. In our script only *V1* has ‘AC 1V’. So an AC analysis is performed with *V1* varying between 0.1 Hz and 1 KHz with 10 points per decade. AC analysis means plain and simple phasor analysis for a range of frequencies. To see what SPICE3 has done for us type `setplot` and select `ac6`. Now type `plot vm(2,3)` and the magnitude of the voltage across nodes (2,3) is plotted. You can try plotting `vr(2,3)`, `vi(2,3)`, `vp(2,3)`, and `vdb(2,3)`. The notation `v(2,3)` to get voltage between nodes (2,3) works only with AC plots. The above plots can be had by typing the functions in longhand—the longhand notation always works—they are: `plot mag(v(2)-v(3))`, `plot real(v(2)-v(3))`, `plot imag(v(2)-v(3))`, `plot ph(v(2)-v(3))`, `plot db(v(2)-v(3))`. Remember to look at `help functions` to see a complete list of available functions.

2.0.5 DC Operating Point Analysis

Before SPICE3 does AC analysis or some other analysis it calculates the dc operating point. This is essential in active circuits. If you want to do the dc analysis yourself the command is `op`. To see what SPICE3 does, type `setplot` and select `op2`. To see the results type `print all`.

2.0.6 Story with ‘.’ Control Commands

The SPICE3 script we have has two types of control commands, one between `.control & .endc` and the second starting with a ‘.’. The control lines starting with a ‘.’ are suited for batch processing¹. With the exception of the `.tf` command every other command has a counterpart in the SPICE3 interactive mode. At times the ‘.’ control commands might prove useful. By trial and error I have found out that all the input between `.control & .endc` has to be in lowercase, the batch mode commands are case insensitive.

To execute the batch mode commands we have to ‘run’ the script. Type `run` and `spice` will execute all the batch mode control commands. Each of these commands generate data which can be printed on the terminal or saved as an ASCII file or plotted. All the data generated by `run` and the original `source` are identical for a particular analysis. See all the available plots by typing `setplot`. To check if the plots are different, type `diff trans5 trans7`. The `diff` command gives the difference between two plot data.

All the ‘.’ commands (with the exception of `.tf`) can be typed in an interactive `xspice` session after removing the dot and making sure that every single letter is in lowercase. Actually all the commands inside the `.control & .endc` construct have the same effect as if you were typing them directly in the interactive session.

¹The ‘.’ commands will work for `PSpice` but the commands between `.control & .endc` are meant for `nutmeg`, an interface to SPICE3. If you are using `PSpice` replace everything between & including `.control & .endc` with `.probe`. The `PSpice` directive `.probe` will save all the simulation data in a `.dat` file which is read by the program `probe` to plot the data. In the windows version of `PSpice`, `probe` is automatically fired after the simulation is complete.

2.0.7 Transfer Function

After you have ‘run’ the script the transfer function can be printed. As usual, type `setplot`, choose `tf7`, and then `print all`. You will see the transfer function between $V1$ and the capacitor voltage is 1. Actually SPICE3 calculates the transfer function at dc. And at dc the capacitor is open-circuited hence the transfer function is 1.

2.0.8 SPICE3 as a Calculator

As mentioned before have a look at all the calculator type of functions available in SPICE3 by typing `help functions`. To do normal calculations type `print sqrt(2*2)` etc. Remember that `print` or `plot` has to proceed the actual calculation for SPICE3 to accept it. You can do all these calculations on all the available data vectors.

2.0.9 Sourcing the Script Again

Now suppose you get more adventurous and want to see what happens when you change $R1 = 1K$. Edit the script `tute-ex1.cir` and replace the last entry (500) in the line starting with `R1` to `1K`. Save the file. Get back to `xspice` and type `source tute-ex1.cir`. Check out the new plots. For example to see the transient voltage across the capacitor, type `setplot`, select `tran13`, and then `plot v(2)-v(3)`. I suppose by now you must have an idea on how to change the script file and explore further.

2.0.10 The Quality of Plots

The nutmeg plots are of a good quality but unfortunately changing labels is a bit difficult. For some reason the x-axis labels are overwritten and the y-axis labels come out horizontal instead of rotated. For the fussy folks there is a command `write` which can be used to write the desired plot data in the ASCII format. Suppose you want to write the capacitor voltage in a file called `capvolt.dat`, the command is `write capvolt.dat v(2)-v(3)`. SPICE3 will automatically write the `time` vector with it. It should be easy to edit the file to make it appropriate for `matlab` or other such plotting program.

There is a `matlab` script `spiceTomatlab.m` to read the spice data from a file ‘`rawspice.txt`’ to a matrix ‘`Var`’. The script which reads in the SPICE3 data for `matlab` can be easily changed to choose the appropriate file names, data names, etc. The file ‘`rawspice.txt`’ should be in the directory from where you call the `matlab` and `spiceTomatlab.m` should be in your search path; on unix it can be in `~/matlab` directory.

2.0.11 Command List

`write rawspice.txt v(1)` to write $v(1)$ in a file `rawspice.txt`
`set hcopydevtype = postscript` to set the output device type to postscript
`hardcopy tempfile.ps v(1)` to get a postscript file `tempfile.ps` which can be printed using `lpr` command

`setplot` to set the appropriate analysis (out of many performed so far) for plotting
`quit` to quit
`help input` to see the syntax of the input script file
`help variables` to see the variables one can set
`show devicename` to see the details of a device, e.g., try `show C1`
`listing` to see the script of the current circuit

3 The Spice Netlist File

The first thing you did before starting this session was to key in a SPICE3 netlist file. I haven't said anything about it till now. What I have described are the interactive features of SPICE3. To learn more on how to create the netlist you really have to have a book or the manual [4]. To get some idea of the type of elements accepted by SPICE3 you can type `help elements` and then explore. Most of the circuit theory books now tell you the elementary concepts about making a SPICE3 script.

In the following we look at three circuits which will give you some idea of how to write script files with active components in it.

3.1 A BJT Common-Emitter Amplifier

Figure 2 shows a BJT common-emitter amplifier. The following PSpice script can be used to simulate the amplifier.

```
Common Emitter Amplifier
* 25 July 1995--for PSpice6.3
V1 1 0 DC 25V
R1 1 2 20k
R2 2 0 5k
R3 1 3 6k
R4 4 0 4.3k
RL 5 0 3k
Rs 7 6 2k
C1 6 2 1u
C2 4 0 50u
C3 3 5 1u
Q2 3 2 4 Qx
.MODEL Qx NPN(BF=200)
VIN 7 0 DC 0 AC 0.5V
*analysis statements
.OP
.ac dec 10 10 10k
.probe
.END
```

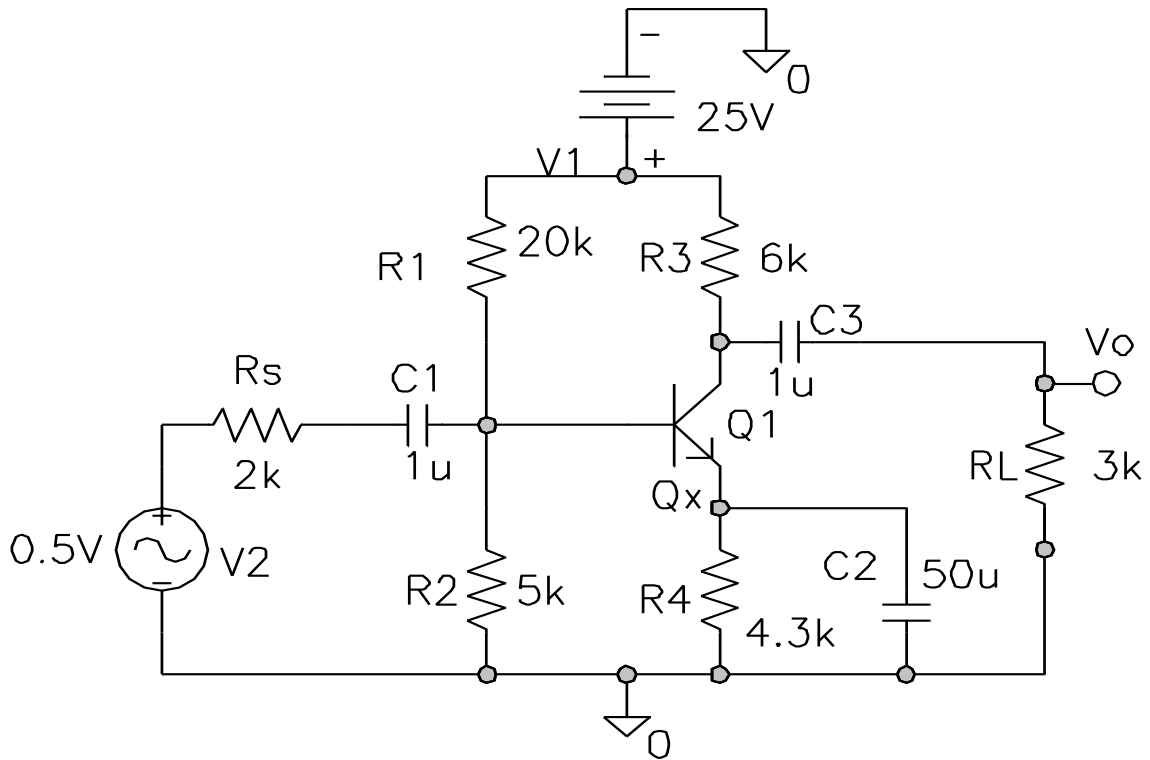


Figure 2: A Common-Emitter Amplifier

You can either directly key in the above script or use the schematic capture [3] feature of PSpice to build the circuit. While building the schematic you will notice that there is no part by the name Qx in the PSpice library. To create Qx choose **Get New Part** (Ctrl-G) from the **Draw** menu from within the Schematics program. Choose part 'QbreakN' and place it in the circuit. Now highlight the part 'QbreakN' and choose **Model** from within the **Edit** menu. Click on the 'Edit instance Model (Text)...' box and edit the '.model' command so that it is what you see in the script above; return to the circuit. This process will save the model for you in a .lib file with the same name as the name you give to the circuit. You can either directly change the model parameters by editing the .lib file or repeat the process given above to change the model parameters.

Remember that the PSpice has provided 'break' parts for you to copy and modify them to your requirements. In the following two examples you can do something similar to what we did here.

3.2 NMOS and JFET Circuits

The two circuits in Figure 3 can be simulated on PSpice using the following two scripts.

Matched MOSFET Circuit

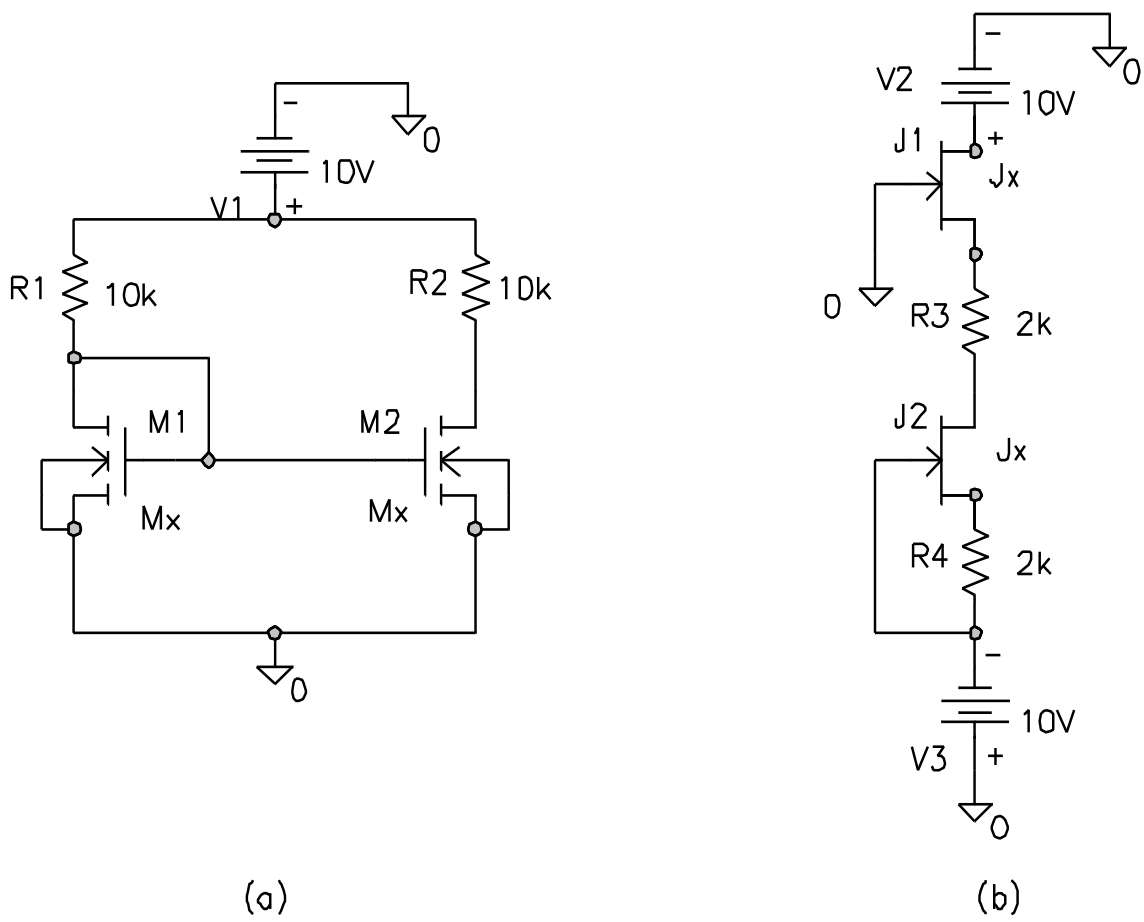


Figure 3: (a) An NMOS Circuit, (b) A JFET Circuit

```

*Problem 4 Assignment 1
*24 July 1996
V1 1 0 DC 10V
R1 1 2 10k
R2 1 4 10k
M1 2 2 0 0 Mx
M2 4 2 0 0 Mx
.MODEL Mx NMOS(VTO=1 KP=2e-3)
.op
.end

```

```

Problem 5.27(f) Horenstein
*25 July 1996
V2 1 0 DC10V
J1 1 0 2 Jx
R3 2 3 2k

```

```

J2 3 4 5 Jx
R4 5 4 2k
V3 4 0 DC-10V
.model Jx NJF(VT0=-2 BETA=1e-3)
.op
.end

```

Here again you can build this circuit using Schematics of PSpice. Remember to use ‘MbreakN’ and ‘JbreakN’ parts to create Mx and Jx respectively.

3.3 PARAM Part – Parametric Analysis

At times you would like to perform analysis for different values of a particular parameter. For example we would like to see how the response of the RLC circuit shown in Figure 1 changes with varying the value of the resistor. You can do this easily using the schematic capture of PSpice and the parametric analysis.

Draw the circuit as seen in Figure 1 using the schematic capture. Double click on the resistance value and a dialog box will pop open. Type in {Rload}. Any thing enclosed in curly brackets will do. Now go to the parts bin (CTRL-G) and pick up the part PARAM and place it somewhere near the schematic. Double click on PARAM and change the attribute of name1 to Rload and of value1 to 0.5k or any value you think is the nominal value of the part. Your schematic should look like it is shown in Figure ?.

Now you go to the analysis setup and setup the values for DC analysis; choose the ‘global parameter’ option. For name fill in Rload; also fill in the start, stop, and increment values that you want. Now perform the analysis and plot the variables using Probe. The plots will be plotted with Rload as the x-axis.

4 The Model Statement

To include semiconductor devices in PSpice or SPICE3 we have to tell the simulator what are the characteristics of the device. Semiconductor devices have different properties unlike say a resistor. To specify a resistor all you have to do is to tell what’s its resistance and the job is done. But semiconductor devices can have varying characteristics. SPICE3 has default values for all the semiconductor devices so if you want to do some simple analysis you can just tell the program that you are happy with the default value and in that case your .model statement can be very simple. Here are a few sample pairs of statements; the first statement gives the node connection of the semiconductor device and the second is the .model statement.

```

D1 1 2 defd ; Diode (node sequence p-side n-side)
.model defd D ; default diode model
Q1 1 2 3 defq ; NPN transistor (node sequence C B E)
.model defq NPN ; default NPN model

```

```

Q2 1 2 3 defq2 ; PNP transistor (node sequence C B E)
.model defq2 PNP ; default PNP model
J1 1 2 3 defj ; N-channel junction FET (node sequence D G S)
.model defj NJF ; default NJF model
J2 5 6 7 defjp ; P-channel junction FET (node sequence D G S)
.model defjp PJF ; default PJF model
M1 1 2 3 4 defm ; N-channel MOSFET (node sequence D G Source Substrate)
.model defm NMOS ; default NMOS model
M2 1 2 3 4 defmp ; P-channel MOSFET (node sequence D G Source Substrate)
.model defmp PMOS ; default PMOS model

```

Note that the in-line comments following ‘;’ are allowed only in PSpice. There are a few important points to remember about the above statements.

- The first letter of every device should be exactly as given, e.g., when you want to include a BJT the first letter should be Q, etc.
- The node sequence also is fixed. For example in the BJT declaration the first node after Qxxx has to be the collector, then the base, and finally the emitter.
- `defq` in the declaration of Q1 above tells the program that the characteristics of the device connected here should be looked up in the `.model` statement for `defq`. There is no restriction on the name you choose for the model.
- The `.model` statement for a particular type of device should use only a particular word. For example, an NPN BJT has to be declared by the letters NPN. The second word after `.model` above is the word reserved for that particular device. Looking at the above statements you can get an idea of which word is used for which device. Many more devices than shown above can be simulated with PSpice, for a complete list see [5] or browse through the PSpice parts library.
- If you want to give the device values other than the default ones then you have to specify them following the reserved word as done in the three scripts we presented before. To get an idea of what the default settings are look in the `.out` file after performing a simulation. To find out which parameters mean what have a look in the text-book [2, pp 185-187].

5 Probe

`Probe` is the plotting package which comes bundled with PSpice. In a PSpice script you can either give command to save the entire simulation data or a few specified voltages and currents. For example

```
.probe
```

will save the entire simulation data and

```
.probe v(2) v(5) I(R1)
```

will save the voltages at nodes 2 and 5 and the current through resistor R1 to be plotted by Probe.

`add trace` is the command to plot data in Probe. You should explore the Probe menu to find out how to label axes, annotate the graphs etc. Probe is a very powerful package. In Probe you can plot the data as it is plus functions of the data. Here are a few modifiers and the functions that you can use with Probe.

Modifier	Effect	Example
M	Magnitude (The default)	VM(2) — Magnitude of V(2)
P	Phase	VP(2) — Phase of V(2)
R	Real Part	VR(2) — Real part of V(2)
I	Imaginary Part	VI(2) — Imaginary part of V(2)
DB	20 times log of value	VDB(2) — $20\log_{10}(V(2))$

Function	Expression	Comment
$sgn(x)$	1 for $x > 0$, -1 for $x < 0$, 0, $x = 0$	
$m(x)$	Magnitude of x	x may be complex
$p(x)$	Phase of x	In degrees
$r(x)$	Real part of x	
$img(x)$	Imaginary part of x	
$g(x)$	Group delay of x	
$d(x)$	Derivative of x	With respect to X-axis
$s(x)$	Integral of x	With respect to X-axis
$avg(x)$	Running average of x	With respect to X-axis
$avg(x, d)$	Running average of x from $x - d$ to x	Over range of X-axis
$rms(x)$	Running RMS average of x	Over range of X-axis
$db(x)$	Magnitude of x in decibels	
$min(x)$	Minimum of real part of x	
$max(x)$	Maximum of real part of x	

You will often repeat the simulate→plot cycle. To plot the same combination of variables you can log the Probe commands using the Log Commands menu item under the File Menu and then replay them.

6 More SPICE3

To simulate a circuit using SPICE3 you need models for various circuit components. Firstly you can copy the models from `eval.lib` which comes with PSpice. If you are using PSpice scripts you can create a library of your own. Let's say you call it `mylib.lib` then you can include the models in the library by using the following line:

```
.LIB "path\mylib.lib"
```

You can also read in a PSpice script into the current script by including:

```
.INC "path\filename"
```

If you are using the schematic capture program of PSpice then the library files etc. can be set from the **Analysis Menu**. The .INC and .LIB commands are not available in SPICE3.

Most of the manufacturers supply model files for the components they manufacture to make it easy for people using SPICE3 to simulate circuits with their parts. You can search the net for manufacturer's site and then download the models. There are sites like the Electronic cookbook archive

at the University of Alberta (<http://www.ee.ualberta.ca/charro/cookbook/>) which have many device models, example scripts, and other useful information. You can easily simulate circuits with literally tens of devices in a matter of hours.

7 End of the Beginning

In this short note I was only trying to get you started and I hope you have got a feel for how SPICE3 works. SPICE3 is a very powerful program and there is much more to it than I have touched here. Please consult the manual [4] and other sources to make effective use of SPICE3.

References

- [1] James G. Gottling. *Hands-On PSpice*. Houghton Mifflin Co., 1995. ISBN 0-395-69916-9.
- [2] Paul R. Gray and Robert G. Meyer. *Analysis and Design of Analog Integrated Circuits*. John Wiley and Sons, Brisbane, 3rd edition, 1993.
- [3] Marc E. Herniter. *Schematic Capture with MicroSim PSpice*. Prentice Hall, NJ, 2nd edition, 1996. ISBN: 0-13-233982-X.
- [4] B. Johnson, T. Quarles, A. R. Newton, D. O. Pederson, and A. Sangiovanni-Vincentelli. *SPICE3 User's Manual*. Department EECS, University of Berkeley, CA 94720 USA, April 1991.
- [5] Paul W. Tuinenga. *SPICE: A Guide to Circuit Simulation & Analysis Using PSpice*. Prentice Hall, NJ, 3rd edition, 1995. ISBN 0-13-433780-8.
- [6] Andrei Vladimirescu. *The SPICE Book*. John Wiley & Sons, Inc., Brisbane, 1994.