

# WinSpice

## A. Introduction

SPICE is short for **S**imulation **P**rogram with **I**ntegrated **C**ircuit **E**mphasis. 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. SPICE originates from the EECS Department of the University of California at Berkeley. It could be said that it is the most well known circuit simulator. In addition, there are at least 30 circuit simulators written, mostly to fulfill other needs, that are derived from SPICE. One of these is WinSpice. WinSpice is a port of Spice3F4 to Win32 systems. WinSpice is ported to run in a window as a native 32-bit application. Spice3F4 is one of the most recent versions of “Berkeley Spice.” The most recent version is Spice3F5.

One advantage of using WinSpice over the numerous other spice programs available is that it has no size limitations on the circuit you may simulate. It is a very powerful, but “bare-bones” circuit simulator. Using WinSpice will force you to understand some simulator details, like netlists for example, that you wouldn’t get by using a more sophisticated program that may use *schematic capture*. Schematic capture allows you to draw the schematic and then simulate this schematic. WinSpice should be installed in most of the computers around the ECE department. You can also download it and install it on your own computer.

In order to either get you familiar, or re-familiar, this short tutorial will cover some of the basics with you. It is by no means complete. This document will also rely heavily on the *WinSpice3 User's Manual* and *A Tutorial for Spice3 / Nutmeg* written by the author of WinSpice, Michael Smith.

## B . Using WinSpice

There are many different ways to do a circuit simulation using WinSpice. This document will discuss one of those.

The steps to performing a circuit simulation with WinSpice are:

1. Draw the circuit and number or label the nodes.
2. Create a WinSpice “netlist”. I usually used notepad for this. The filename should be of the form “\*.cir”, where \* is any filename you give it and “cir” is the extension of the file. The WinSpice file has two components:
  - (1) Circuit description
  - (2) Analysis and output control lines
3. Simulate the circuit (Run WinSpice).
4. Evaluate the results of the output. This is usually done using a combination of the plotting features of WinSpice and the saving results to an output file.

An example of a WinSpice netlist and the schematic of the circuit it is simulating is:

### *Step 1*

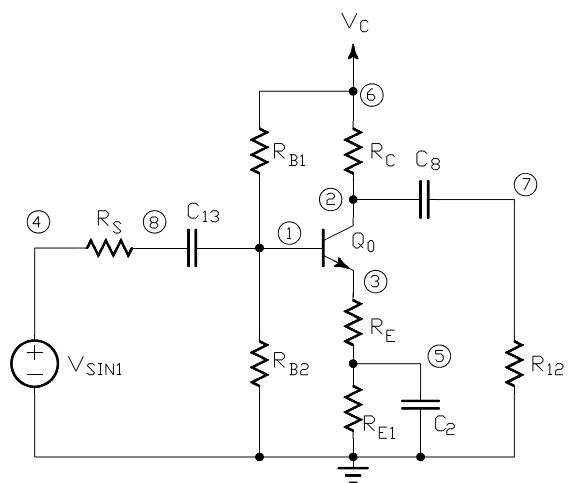


Figure 1

**Step 2**

## EXAMPLE.CIR

```
.control ; start control statements
destroy all ;WinSpice erases all previously stored data and starts fresh.
op ; perform an operating point analysis
tran 1E-6 400E-6 0 1E-6 ; perform a transient (time-domain) analysis
plot v(7) ; plot the results of the transient analysis for the voltage of node 7.
AC DEC 20 1 100K ; perform an ac (small-signal) frequency analysis
plot vdb(7) ; plot the results of the ac analysis for the voltage of node 7.
print vdb(7) > output.txt ; show the calculated values of vdb(7) and write to a file.
.endc ; start control statements
```

\*Begin circuit description

```
VSIN1 4 0 SIN(0 350E-3 10000 0 0) AC 1 DC 0
RS 4 8 600
C2 0 5 10E-6
RC 6 2 5000 ;test
RE 3 5 300
RE1 5 0 2500
RB2 1 0 12000
RB1 6 1 26000
C8 2 7 1E-6
VC 6 0 15
R12 7 0 10000
C13 8 1 1E-6
Q0 2 1 3 MQ0 1
.MODEL MQ0 NPN (BF=100 IS=1E-15) ;BJT "model" for Q0.
*end circuit description
```

```
.END
```

Here are some key points of this netlist:

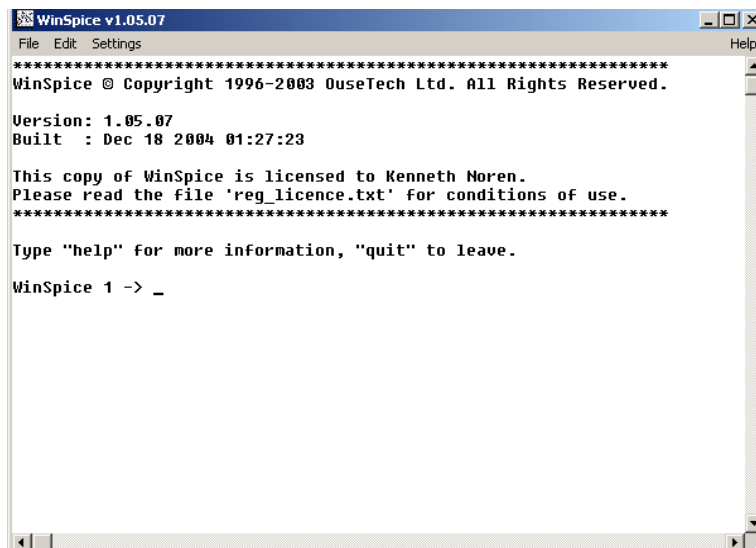
- The first line is a “title line” and must be in the netlist.
  - Winspice will always take the first line to be a “title line”
- The last line is a “.END” line and must be the last line in the netlist
- Comments may be included into the net list in at least two ways:
  - (1) On a single line, using “\*” as the first character
  - (2) After a command line, element line, model line, etc, using “;”.
- The circuit description tells WinSpice what the elements in the circuit are, the values of the elements and their parameters, and how the elements are connected together.
- The “analysis and output control lines” start with “.control” and end with “.endc”
  - This tells WinSpice what types of analysis to perform on the circuit, what type of output is desired, etc. For this particular example, and in this order:
    - (1) All previously stored data (usually in memory) is cleared.
    - (2) An operating point analysis is done.
    - (3) A transient analysis is performed
    - (4) The results of the transient analysis are plotted.
    - (5) A AC analysis is performed
    - (6) The results of the AC analysis are plotted.
- Note that simple elements, like resistors and capacitors show the connection and value.
- More complex elements, like Q0 (and most semiconductor devices) require a “model” statement. The model statement tells WinSpice what model to use and the values of the parameters of that model.

More details will be discussed later.

### Step 3

To run the circuit:

- (1) Run Winspice. The following Winspice “Interactive Interpreter” window should open up:



```
WinSpice v1.05.07
File Edit Settings Help
*****
WinSpice © Copyright 1996-2003 OuseTech Ltd. All Rights Reserved.

Version: 1.05.07
Built : Dec 18 2004 01:27:23

This copy of WinSpice is licensed to Kenneth Noren.
Please read the file 'reg_licence.txt' for conditions of use.
*****

Type "help" for more information, "quit" to leave.

WinSpice 1 -> _
```

**Figure 2** The WinSpice “Interactive Interpreter” window at start up.

- (2) Open the Winspice netlist
  - Go “File-Open” and browse until your file appears.
- (3) Upon opening the netlist, several things have happened:
  - The circuit and control lines have been read into WinSpice
  - The 3 analysis have been performed.
  - The 2 plots that have been specified should open up.

Here are the results:

```

WinSpice v1.05.07
File Edit Settings Help
Version: 1.05.07
Built : Dec 18 2004 01:27:23

This copy of WinSpice is licensed to Kenneth Noren.
Please read the file 'reg_licence.txt' for conditions of use.
*****

Type "help" for more information, "quit" to leave.

WinSpice 1 -> cd
Current directory: C:\Program Files\0useTech\WinSpice
WinSpice 2 -> source \Example.CIR
Reading .\Example.CIR

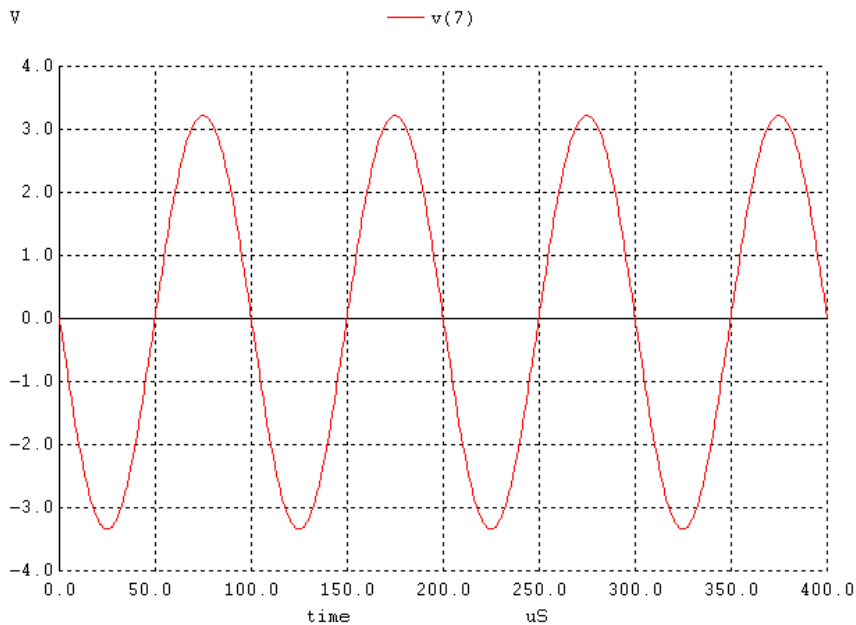
Circuit: EXAMPLE.CIR

TEMP=27 deg C
DC Operating Point ... 100%

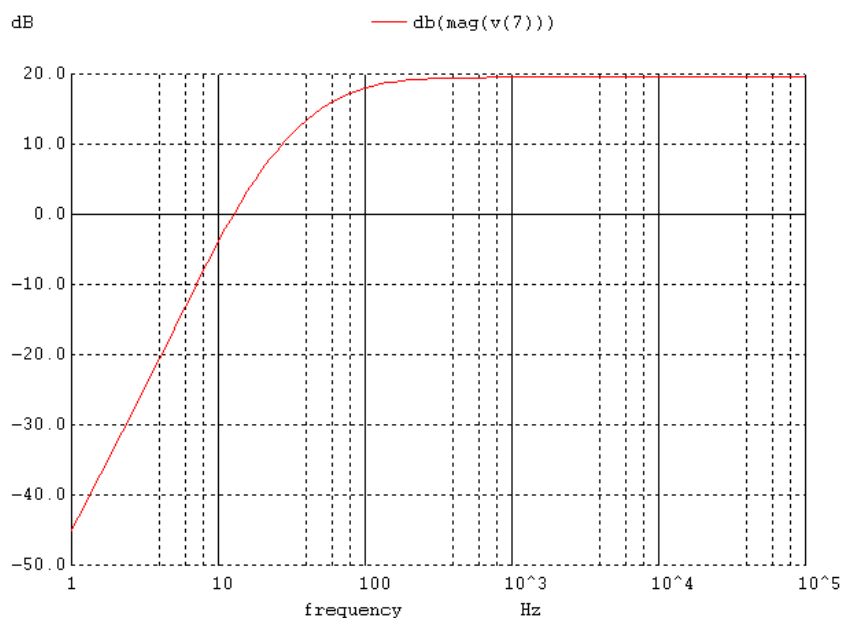
TEMP=27 deg C
Transient analysis ... 100%

TEMP=27 deg C
AC analysis ... 100%
WinSpice 3 -> _
  
```

**Figure 4** WinSpice window after running reading in the circuit and performing the desired analysis.



**Figure 3** WinSpice plot of  $v(7)$  for the transient analysis



**Figure 5** WinSpice plot of the log of the magnitude of  $v(7)$  for the frequency analysis.

---

The partial results (last 10 data points only) of "output.txt"

EXAMPLE.CIR  
AC Analysis Sun Sep 04 11:45:47 2005

Index	frequency		db(mag(v(7)))
91	3.548134e+04,	0.000000e+00	1.954517e+01
92	3.981072e+04,	0.000000e+00	1.954517e+01
93	4.466836e+04,	0.000000e+00	1.954517e+01
94	5.011872e+04,	0.000000e+00	1.954518e+01
95	5.623413e+04,	0.000000e+00	1.954518e+01
96	6.309573e+04,	0.000000e+00	1.954518e+01
97	7.079458e+04,	0.000000e+00	1.954518e+01
98	7.943282e+04,	0.000000e+00	1.954518e+01
99	8.912509e+04,	0.000000e+00	1.954518e+01
100	1.000000e+05,	0.000000e+00	1.954518e+01

**Figure 6** Part of the data that was written to "output.txt"

#### Step 4

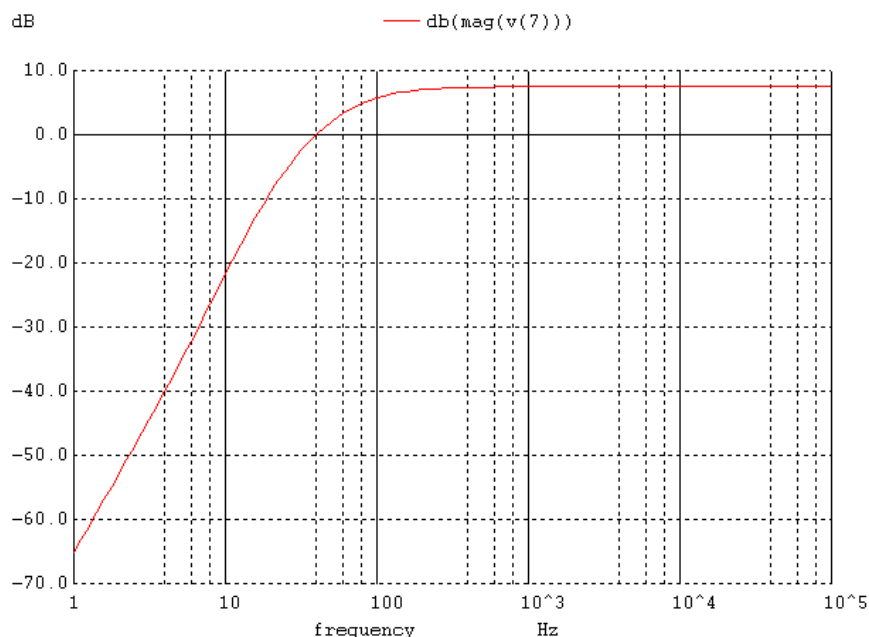
- Interpret the results. Are they correct?

#### Important Final notes

1. Making changes to the original netlist and re-simulating.

- This is easy in WinSpice. Simply make the changes in the file containing the netlist and save the file. The new results are calculated and plotted.
- For example, performing the following operations:
  - type "Edit" in the WinSpice window.
  - Change the value of R12 from 10k $\Omega$  to 1k $\Omega$
  - Save the file

Results in the new plots and output being generated. Only the AC plot will be shown here.



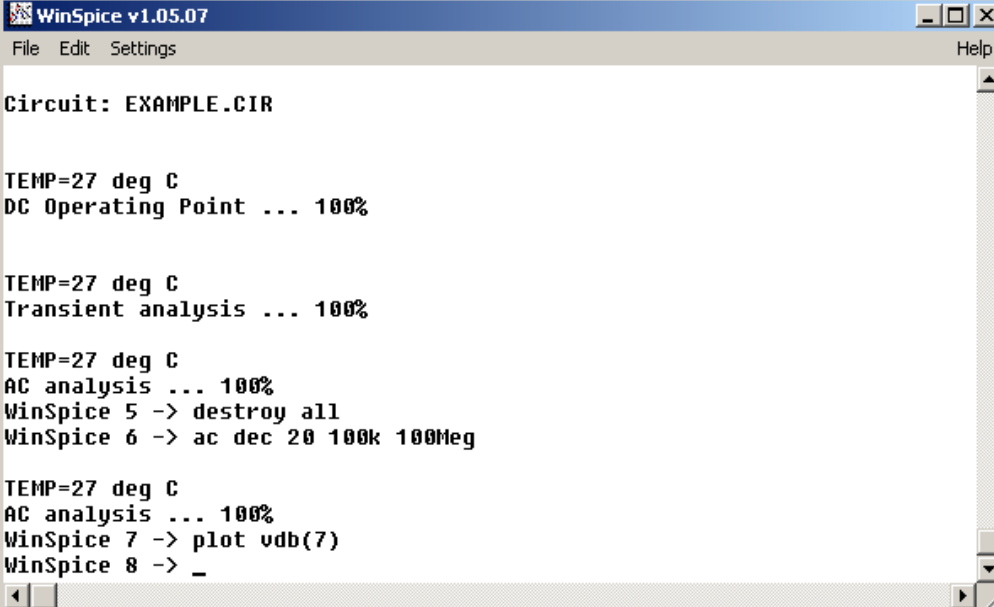
**Figure 7** Plot of 20 times the log of  $v(7)$  vs. frequency after R12 has been changed from 10k $\Omega$  to 1k $\Omega$ .

We note a decrease in gain.

2. Any analysis and output control lines that may be specified in the netlist, may be typed and thus performed directly into the WinSpice interactive interpreter. For example, typing the following in the WinSpice window:

- destroy all
- ac dec 20 100k 100meg
- plot vdb(7)

results in the following:



```

WinSpice v1.05.07
File Edit Settings Help

Circuit: EXAMPLE.CIR

TEMP=27 deg C
DC Operating Point ... 100%

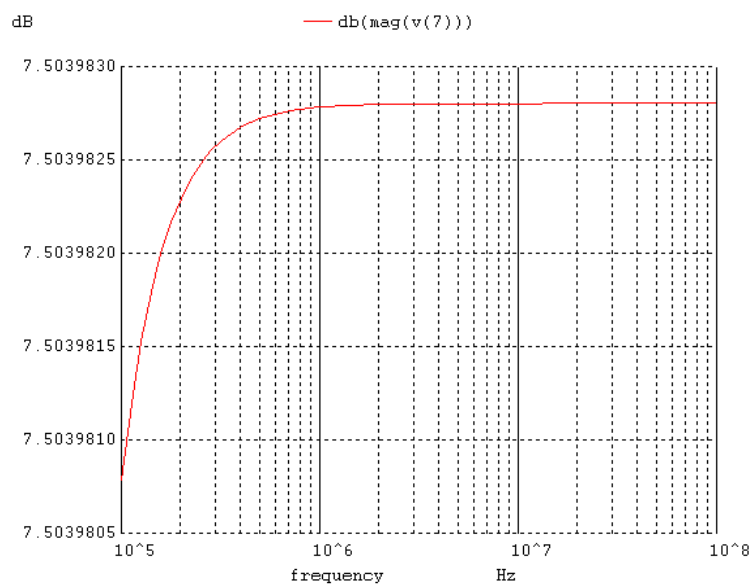
TEMP=27 deg C
Transient analysis ... 100%

TEMP=27 deg C
AC analysis ... 100%
WinSpice 5 -> destroy all
WinSpice 6 -> ac dec 20 100k 100Meg

TEMP=27 deg C
AC analysis ... 100%
WinSpice 7 -> plot vdb(7)
WinSpice 8 -> _

```

**Figure 8** WinSpice window after typing in a sequence of three commands.



**Figure 9** Plot of 20 times the log of the magnitude of  $v(7)$  vs. frequency. This plot and the preceding analysis were a result of typing commands directly in the WinSpice Window.

## C. Creating the Circuit Description

The netlist contains a description of the circuit to be simulated. It shows what types of devices exist, their values, model types and model parameters as well as a description of how those devices are connected. The circuit nodes can be numbered or can be any text string, but **one** of the nodes must be labeled **0**, which is used as the **ground node**. It is somewhat traditional to number the nodes sequentially, however you may want to label the nodes by meaningful names, such as **in** or **out**, for example.

Each line of the circuit description netlist may contain one element. The rules governing the format of these elements are in the manual and we will discuss a few of them. Examples of analog elements which can be used are resistors, capacitors, inductors, independent voltage and current sources, the four types of dependent sources, transmission lines, voltage and current controlled switches, and five common semiconductor devices: diodes, BJT's, JFET's, MOSFET's and GaAsFET's. Two very important statements that may be included in the netlist are the **.MODEL** statement and the **.SUBCKT** statement. A list of the elements that are allowed in WinSpice will follow this discussion. The list is not complete. Also, the elements in the list are often simplified versions of the actual element description. As you gain experience with PSpice, you may want to consult the PSpice manual for more complicated definitions of the elements and some of the other types of elements available.

### Element Statements

#### 1. Resistor

Rxxx <+ node> <- node> <value>

Example:

**R1 1 2 1k**

The xxx represents any other numbers or letters which you desire, and you must use at least one. There is no limit to the length of a component name, however common sense dictates you use names with reasonable lengths which are somewhat meaningful.

#### 2. Capacitor

Cxxx <+ node> <- node> <value>

**C1 3 4 1e-6**

#### 3. Inductor

Lxxx <+ node> <- node> <value>

**LR 5 4 1m**

#### 4. Independent Voltage Source



Independent voltage sources are specified, in general, as:

**V<sub>xxx</sub>** <+ node> <- node> [ [DC] <value> ] [AC <magnitude> [phase]] [transient + specification]

The items surrounded by < > are musts. The items surrounded by [ ] are options. The items surrounded by < > but inside [ ] means that if the option is used, the item must be specified.

Some examples are:

**VBIAS**      **13 0 2.3mV**  
**VAC**        **2 3 AC 0.001**  
**VACPHS**    **2 3 AC 0.001 90**  
**VPULSE**    **1 0 PULSE (-1mV 1mV 2ns 2ns 50ns 100ns)**  
**V3**         **26 27 DC 0.002 AC 1 SIN(0.002 0.002 1.5MEG)**

Note in the first example that a value was specified, however the type of voltage source was not. In this case, PSpice defaults to a DC source. The transient specification must be one of:

PULSE (<parameters>)      for a pulse waveform  
PWL (<parameters>)        for a piecewise linear waveform  
SIN (<parameters>)         for a sinusoidal waveform

The general form of SIN is:

SIN (<voff> <vAMPL> <freq> <td> <df> <phase>)

<voff>      dc offset voltage of the waveform  
<vAMPL>    peak amplitude of voltage waveform  
<freq>      frequency of the sinusoidal  
<td>        the waveform maintains its value at 0s for a specified td  
<df>        the sinusoidal portion of the waveform becomes exponentially damped  
<phase>    phase of the sine wave

The general form of PULSE is:

PULSE (<v1> <v2> <td> <tr> <tf> <pw> <per>)

<v1>        initial voltage  
<v2>        pulsed voltage  
<td>        delay  
<tr>        rise time  
<tf>        fall time  
<pw>        pulse width  
<per>       period

The general form of PWL is:

PWL (<v1> <t1> <v2> <t2>...<vn> <tn>

Which means the voltage of the source at <tn> is <vn>. The points which specify the voltages at given times are then "connected" with a straight line approximation.

## 5. Independent Current Source

Independent current sources are specified, in general, as:

Ixxx <+ node> <- node> [ [DC] <value> ] [AC <magnitude> [phase]] [transient + specification]

Some examples are:

```
IBIAS      13 0 2.3mV
IAC        2 3 AC 0.001
IACPHS     2 3 AC 0.001 90
IPULSE     1 0 PULSE (-1mV 1mV 2ns 2ns 50ns 100ns)
I3         26 27 DC 0.002 AC 1 SIN(0.002 0.002 1.5MEG)
```

Current sources descriptions are nearly identical to voltage sources.

## 6. Diode

Semiconductor devices require a **.model** statement. Consider the diode for example. Its simplified form is:

Dxxx <+ node> <- node> <model name>

*Somewhere in the circuit listing you need a statement of the form*

**.MODEL** <model name> D [model parameters]

There are 25 model parameters that describe the diode behavior and may be specified. You can see from the model line above, the you may choose to specify no model parameters. If you choose this option, PSpice uses default values for these parameters. We will worry about 3 parameters which we are familiar with. They are IS, N, and BV. These are the same parameters which we have taught in EE 316. To make it clear how PSpice interprets these, a table is presented.

Model Parameters	Units	Default Value
IS saturation current	amp	1E-14
N emission coefficient	none	1
BV reverse breakdown "knee" voltage	volt	infinite

An example of a diode listed in the netlist is:

```
D1 2 3 DMOD
.MODEL DMOD D (IS=1e-12 N=0.8 BV=80)
```

Another example which uses the default values is:

```
D1 2 3 DMOD
.MODEL DMOD D
```

## 7. Bipolar Junction Transistor (BJT)

The BJT may be listed in the netlist in a form shown below:

```
Qxxxx <collector node> <base node> <emitter node> <model name>
```

As with the diode, a model statement is required and it has the form of

```
.MODEL <model name> <NPN or PNP> [model parameters]
```

The user must specify whether the BJT is an npn or pnp. The BJT has 55 model parameters that characterize its behavior. The parameters we will give attention to are:

Model Parameters	Units	Default Value
IS saturation current	amp	1E-16
NF emission coefficient	none	1
BF ideal maximum forward beta	none	100
VAF forward early voltage	volts	infinite

In relationship to EE 316 and EE 318, BF is approximately  $\beta$ , and NF is N (often 1 for us).

An examples of a BJT listed in a netlist is:

```
Q1 1 2 3 PNPTYPE
.MODEL PNPTYPE PNP (IS=1e-15 NF=0.8 BF=80 VAF=125)
```

or an example which uses the default values is:

```
Q1 2 3 QMOD
.MODEL QMOD NPN
```

## 8. Junction Field Effect Transistor (JFET)

A simplified form of a JFET is:

```
Jxxxx <drain node> <gate node> <source node> <model name>
```

With a model statement somewhere in the netlist as:

`.MODEL <model name> <NJF or PJF> [model parameters]`

The user must specify whether or not the transistor is an n-channel JFET or a p-channel JFET. The JFET has 21 model parameters that characterize its behavior. The parameters we will give attention to are:

Model Parameters		Units	Default Value
VTO	threshold voltage	volt	-2.0
BETA	transconductance coefficient	amp/volt <sup>2</sup>	1E-4
LAMBDA	channel-length modulation	volt <sup>-1</sup>	0

An examples of a JFET specified in a circuit is:

```
J1 1 2 3 NTYPE
.MODEL NTYPE NJF (VTO=-3 BETA=1E-5 LAMBDA=.1)
```

or an example which uses the default values is:

```
J13 2 13 18 JMOD
.MODEL JMOD PJF
```

## 9. Metal-Oxide-Semiconductor Field Effect Transistor (MOSFET)

A simplified form of a MOSFET is:

```
Mxxxx <drain node> <gate node> <source node> <bulk/substrate node> <model name> +
[L = <value>] [W = <value>]
```

With a model statement somewhere of the form

```
.MODEL <model name> <NMOS or PMOS> [model parameters]
```

The user must specify whether or not the transistor is an n-channel MOSFET or a p-channel MOSFET. The MOSFET has 52 model parameters that characterize its behavior. The parameters we will give attention to are:

Model Parameters		Units	Default Value
W	channel width	meter	100u
L	channel length	meter	100u
VTO	zero-bias threshold voltage	volt	0
KP	transconductance coefficient	amp/volt <sup>2</sup>	2E-5
LAMBDA	channel-length modulation	volt <sup>-1</sup>	0

Additionally there are several *levels* of MOSFET modeling, which give 4 levels of complexity.

An example of a MOSFET specified in a circuit is:

```
M1 1 2 3 3 NTYPE
.MODEL NTYPE NMOS (VTO=-3 KP=1E-5 LAMBDA=.1 W=10U L=50U)
```

This is equivalent to

```
M1 1 2 3 3 NTYPE W=10U L=50U
.MODEL NTYPE NMOS (VTO=-3 KP=1E-5 LAMBDA=.1)
```

Note the different ways that the length and width of the MOSFET can be specified.

An example which uses the default values is:

```
Q13 2 13 18 18 PTYPE
.MODEL PTYPE PMOS
```

## 10. Controlled Sources

### *a. Voltage-Controlled Voltage Source*

A voltage-controlled voltage source has the general form of

```
Exxxx <(+) node> <(-) node> <(+) controlling node> <(-) controlling node> <gain>
```

Example:

```
EOPAMP 1 3 5 6 100000
```

Which means the voltage controlled voltage source is connected between nodes 1 and 3, the positive node at 1, with the controlling voltage being V(5,6) with a gain of 100000.

### *b. Voltage Controlled Current Source*

A voltage-controlled current source has the general form of

```
Gxxxx <(+) node> <(-) node> <(+) controlling node> <(-) controlling node> <gain>
```

Example:

```
G1 4 8 2 1 .1
```

Which means the current flows into node 4 and out of node 8 and with the controlling voltage being V(2,1) with a gain of .1.

### *c. Current-Controlled Voltage Source*

A current-controlled voltage source has the general form of

Hxxxx <(+) *node*> <(-) *node*> <*controlling V device name*> <*gain*>

Example:

HLDR 2 10 VSENSE 10

Which means the current-controlled voltage source is connected between nodes 2 and 10, the positive node at 2, with the controlling current the current through the voltage source named VSENSE with a gain of 10.

#### *d. Current-Controlled Current Source*

A current-controlled current source has the general form of

Fxxxx <(+) *node*> <(-) *node*> <*controlling V device name*> <*gain*>

Example:

F1 4 8 VSOURCE 25

Which means the current flows into node 4 and out of node 8 and with the controlling current the current through the voltage source named VSOURCE with a gain of 25.

Lastly, we need to discuss a .subckt statement. A .subckt statement can be used to define a device or component which will be used repeatedly in the circuit under test or in future circuits.

The form is:

```
.SUBCKT name node [node...]
  components defining subckt
.ENDS [name]
```

Here, in the .ENDS statement, *name* is optional, however it is recommended.

Note that values can be specified in exponential notation as well as with power-of-ten suffix letters. These are

f	femto-	10 <sup>-15</sup>
p	pico-	10 <sup>-12</sup>
n	nano-	10 <sup>-9</sup>
u	micro-	10 <sup>-6</sup>
m	milli-	10 <sup>-3</sup>
k	kilo-	10 <sup>3</sup>
meg	mega-	10 <sup>6</sup>
g	giga-	10 <sup>9</sup>
t	tera-	10 <sup>12</sup>
mil	(0.001")	25.4*10 <sup>-6</sup>

Thus, a capacitor for example can be specified as:

**C1 3 4 1u**

or

**C1 3 4 1e-6**

Once the nodes of the circuit have been labeled and the netlist is created, control statements should be added to tell PSpice what to do. We will use control statements to tell PSpice what type of analysis to do. The first control cards to use are the fundamental types of circuit analysis, an ac analysis, dc analysis, or transient analysis.

In summary, WinSpice supports the following devices:

Letter	Description
R	Resistor
C	Capacitor
L	Inductor
K	Coupled inductor
S	Voltage-controlled switch
W	Current-controlled switch
I	Independent current source
G	Linear voltage-controlled current source
E	Linear and non-linear voltage-controlled voltage source
F	Linear and non-linear current-controlled current source
H	Linear current-controlled voltage source
B	Non-linear dependent voltage or current source
T	Lossless transmission line
O	Lossy transmission line
U	Uniform distributed RC transmission lines
D	Diode
Q	Bipolar Junction Transistor (BJT)
J	Junction Field-Effect Transistor (JFET)
M	MOSFET
X	Subcircuit call
Z	MESFET

*For more details, see Chapter 4 of “WinSpice3 User's Manual”*

## D. Analysis and Output Control Statements

The first line is a “.control”. The last line is a “.endc”. The analysis and output control statement lines are in between.

### 1. For an ac analysis:

*AC* <*sweep type*> <*points value*> <*start frequency value*> <*end frequency value*>

The *sweep type* is either LIN, OCT, or DEC.

**LIN** Linear sweep. The frequency is swept linearly from the starting frequency to the ending frequency. <*points value*> is the **total number of points** in the sweep.

**OCT** Sweep by octaves. The frequency is swept logarithmically by octaves. <*points value*> is the number of **points per octave**.

**DEC** Sweep by decades. The frequency is swept logarithmically by decades. <*points value*> is the number of **points per decade**.

Some examples are:

**AC LIN 101 100 200** The frequency is swept from 100Hz to 200Hz with 101 points, resulting in the sweep being 100, 101, 102, etc.

**AC DEC 20 1MEG 100MEG** The frequency is swept from 1MEG to 100MEG with 20 points per decade. There is 2 decades between 1MEG and 100MEG resulting in a total of 40 points.

**AC OCT 10 1k 16k** The frequency is swept from 1k to 16k with 10 points per octave. There are 4 octaves between 1k to 16k resulting in a total of 40 points.

### 2. For a transient analysis:

*TRAN* <*print step value*> <*final time value*> [*no-print value* [*step ceiling value*]]  
(This is a simplified form)

<*print step value*> is the time interval used for printing or plotting the results of the transient analysis to a specified output file.

<*final time value*> is the ending value of the time interval. Note that the transient analysis always starts off at 0s and ends at the time specified by the final time value.

*no-print value* Although PSpice begins the transient analysis at 0s, a situation may occur where the user is not interested in the results from 0s to a specified value. This specified value is the no-print value and the results from this value are not plotted, printed, or given to probe. Note that PSpice still does the analysis in this interval, but the results are not made available to the user. This may be



particularly helpful if large amounts of output need to be suppressed.

*step ceiling value* When WinSpice calculates/evaluates a time interval, the step size taken is determined by WinSpice. This step size may increase or decrease depending on the rate of change of the voltages and currents in the circuit. The default ceiling is  $\langle \text{final time value} \rangle / 50$ . If the time interval between the points is too large the output may appear to be distorted. Specifying a *step ceiling value* will reduce the maximum size of the time interval and usually will succeed in "smoothing" the displayed waveform. Some examples are:

**TRAN 1U 10U 5U 1U** The transient analysis begins at 0s and ends at 10uS. Any plots or tables printed in the output file will be in 1uS intervals, but will not begin until 5uS. The maximum step actually used to perform the transient analysis is 1uS.

**TRAN 10U 100U** The transient analysis begins at 0s and ends at 100uS. The maximum time step is  $100\text{us}/50 = 2\text{us}$ . Any plots or tables printed in the output file will be in 10us intervals, and begin at 0s.

### 3. Finally, the control card used to perform a dc analysis has the form of:

DC *<sweep variable name>* *<start value>* *<end value>* *<increment value>*

*<sweep variable name>* is the name of the variable which is to be swept.

*<start value>* is the beginning value of the variable

*<end value>* is the ending value of the variable

*<increment value>* is the step in which the variable is to be increased.

The .DC statement causes a DC sweep analysis to be performed on the circuit. The DC sweep analysis calculates the circuit's bias point over a range of values for *<sweep variable name>*.

*<start value>* may be greater or less *<end value>*. *<increment value>* **must be greater than zero**.

Some examples are:

**DC VIN -.25 .25 .05** The dc voltage source, VIN, will be swept from -.25V to .25V in increments of 0.05V. The resulting sweep will be -0.25, -0.20, -0.15, ...

**DC IS 5M -2M 0.1M** The dc current source, IS, will be swept from 5M to 2M in increments of 0.1M. The resulting sweep will be 5M, 4.9M, 4.8M ...

For the circuit we are simulating, we would like to look at a transient response. The period of the input wave form is  $1/60 = 16.67\text{m}$ , and we can then choose to view 3 periods. This gives a *final time value* of 50m. We can choose the *print step* value to be 1u. We will choose the step ceiling value to be 1u. This gives us the following control card:

**TRAN 1U 50U 0 1U**

#### 4. Plot

Is simply:

Plot <node voltage>

**A summary of commands available in WinSpice is provided here. Note that some of these are more appropriate for entering in at the command line.**

Ac: Perform an AC frequency response analysis  
Alias: Create an alias for a command  
Alter: Change a device or model parameter  
Asciiplot: Plot values using old-style character plots  
Bug: Mail a bug  
Cd: Change directory  
Cross: Create a new vector  
Dc: Perform a DC-sweep analysis  
Define: Define a function  
Delete: Remove a trace or breakpoint  
Destroy: Delete a data set (plot)  
Diff: Compare vectors  
Display: List known vectors and types  
Disto: Perform a distortion analysis  
Echo: Print text  
Edit: Edit the current circuit  
Fourier: Perform a fourier transform  
Hardcopy: Save a plot to a file for printing  
Help: Print summaries of WinSpice3 commands  
History: Review previous commands  
Iplot: Incremental plot  
Let: Assign a value to a vector  
Linearize: Interpolate to a linear scale  
Listing: Print a listing of the current circuit  
Load: Load rawfile data  
Noise: Perform a noise analysis  
Op: Perform an operating point analysis  
Plot: Plot values on the display  
Print: Print values  
Pz: Perform a Pole-Zero Analysis  
Quit: Leave WinSpice3  
Rawfile: Send further results directly to a rawfile  
Reset: Reset an analysis  
Reshape: Alter the dimensionality or dimensions of a vector

Resume: Continue a simulation after a stop  
Run: Run analysis from the input file  
Rusage: Resource usage  
Save: Save a set of output vectors  
Sens: Run a sensitivity analysis  
Set: Set the value of a variable  
Setcirc: Change the current circuit  
Setplot: Switch the current set of vectors  
Setscale: Set the scale for a plot  
Settype: Set the type of a vector  
Shell: Call the command interpreter  
Shift: Alter a list variable  
Show: List device state  
Showmod: List model parameter values  
Source: Read a WinSpice3 input file  
Spec: Generate a Fourier transform vector  
Status: Display breakpoint and trace information  
Step: Run a fixed number of time points  
Stop: Set a breakpoint  
Strcmp: Compare strings  
Temp: Define circuit temperature  
Tf: Run a Transfer Function analysis  
Trace: Trace nodes  
Tran: Perform a transient analysis  
Transpose: Swap the elements in a multi-dimensional data set  
Tutorial: Display hypertext help  
Unalias: Retract an alias  
Undefine: Retract a definition  
Unlet: Delete vectors  
Unset: Clear a variable  
Version: Print the version of WinSpice  
Where: Identify troublesome node or device  
Write: Write data to a file

*For more details, see Chapter 6 of “WinSpice3 User's Manual”*

## **E. Device Modeling**

- .MODEL**      The .MODEL statement defines a set of device parameters for a specific device, which can be referenced in the circuit.
- .SUBCKT**      The .SUBCKT statement begins the definition of a subcircuit. Suppose, for example you were simulating a circuit which contained 3 op-amps, each op-amp containing 15 components. Instead of listing the 15 components at each point it was necessary to specify the op-amp, you can list the op-amp once, in a subcircuit, and call that subcircuit up 3 times.
- .ENDS**        Marks the END of a subcircuit.

**References**

[1] <http://www.ousetech.co.uk/winspice2/index.html> - WinSpice homepage

[2] Smith, Michael, *WinSpice3 User's Manual*, 2004

[3] Smith, Michael, *A Tutorial for Spice3 / Nutmeg*, 2004