By Joshua Cantrell
jjc@icsl.ucla.edu

Writing Simple Spice Netlists

Introduction

Spice is used extensively in education and research to simulate analog circuits. This powerful tool can help you avoid assembling circuits which have very little hope of operating in practice through prior computer simulation. The circuits are described using a simple circuit description language which is composed of components with terminals attached to particular nodes. These groups of components attached to nodes are called netlists.

Parts of a Spice Netlist

A Spice netlist is usually organized into different parts. The very first line is ignored by the Spice simulator and becomes the title of the simulation.[1] The rest of the lines can be somewhat scattered assuming the correct conventions are used. For *commands*, each line must start with a '.' (period). For *components*, each line must start with a letter which represents the component type (eg., 'M' for MOSFET). When a command or component description is continued on multiple lines, a '+' (plus) begins each following line so that Spice knows it belongs to whatever is on the previous line. Any line to be ignored is either left blank, or starts with a '*' (asterik).

A simple Spice netlist is shown below:

```
Spice Simulation 1-1

*** MODEL Descriptions ***
.model nm NMOS level=2 VT0=0.7
KP=80e-6 LAMBDA=0.01

*** NETLIST Description ***
M1 vdd ng 0 0 nm W=3u L=3u
R1 in ng 50
Vdd vdd 0 5
Vin in 0 2.5

*** SIMULATION Commands ***
.op

.end
```
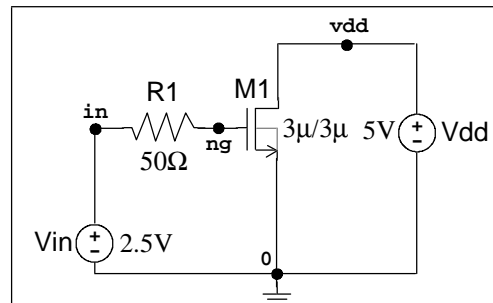

Figure 1: Schematic of Spice Simulation 1-1 Netlist

The first line is the title of the simulation. It's unimportant for the simulation except for identification. The Spice commands under "MODEL Descriptions" are used to define the electrical properties of particular devices. In this example, the MOSFET is defined by the given parameters in the model. In the "NETLIST Description", the components are listed with the nodes they are connected to. Notice that each one starts with a letter. The MOSFET starts with an 'M', the resistor starts with an 'R', and the voltage sources start with 'V's.

---

[1] A common error when simulating is to place a component definition or operation on this first line and the Spice simulator will appear to not function properly. It will either complain about missing components, models, or not do what you think.

Simple Netlist Components

Resistor Component

The register is described by

```
RXXXXXXX N1 N2 <VALUE> <MNAME> <L=LENGTH> <W=WIDTH> <TEMP=T>
```

The parameters are:
   N1 = the first terminal
   N2 = the second terminal
   <VALUE> = resistance in ohms.
   <MNAME> = name of the model used (useful for semiconductor resistors)
   <L=LENGTH> = length of the resistor (useful for semiconductor resistors)
   <W=WIDTH> = width of the resistor (useful for semiconductor resistors)
   <TEMP=T> = temperature of the resistor in Kelvin (useful in noise analysis and
         semiconductor resistors)

Notice that the parameters given with '=' in their definitions must be written with the symbol followed by '=' followed by the value. For example, to set a resistor to 500 Kelvin, you'd write:

```
RHOT n1 n2 10k TEMP=500
```

All of the parameters surrounded by '<' and '>' can be left out and will be replaced by default values. All of them must appear in order, except for the parameters with '=' in their definitions.

Capacitor Component

The capacitor is described by

```
CXXXXXXX N+ N- VALUE <IC=INCOND>
```

The parameters are:
   N+ = the positive terminal
   N- = the negative terminal
   VALUE = capacitance in farads
   <IC=INCOND> = starting voltage in a simulation

Inductor Component

The inductor is described by

```
LYYYYYYY N+ N- VALUE <IC=INCOND>
```

The parameters are:
   N+ = the positive terminal
   N- = the negative terminal
   VALUE = capacitance in farads
   <IC=INCOND> = starting voltage in a simulation

Coupled Inductors Component

Two coupled inductors are described by

```
KXXXXXXX LYYYYYYY LZZZZZZZ VALUE
```

The parameters are:

LYYYYYYY = the name of the first coupled inductor
LZZZZZZZ = the name of the second coupled inductor
VALUE = the coefficient of coupling, K, where $0 < K \leq 1$

The orientation of the inductors is determined by the first node, which is considered to be the *dot*ed node.

Junction Diode Component

A diode is described by

```
DXXXXXXX N+ N- MNAME <AREA> <OFF> <IC=VD> <TEMP=T>
```

The parameters are:

N+ = the name of the positive terminal
N- = the name of the negative terminal
MNAME = name of the model used
<AREA> = the scaling factor of the diode (determines how much current can flow through it)
<OFF> = an optional starting condition for DC analysis
<IC=VD> = starting voltage in a simulation
<TEMP=T> = temperature of the diode in Kelvin

Bipolar Junction Transistor (BJT) Component

A bipolar junction transistor is described by

```
QXXXXXXX NC NB NE <NS> MNAME <AREA> <OFF> <IC=VBE, VCE> <TEMP=T>
```

The parameters are:

NC I= the name of the collector terminal
NB = the name of the base terminal
NE = the name of the emitter terminal
<NS> = the name of the substrate terminal (optional)
MNAME = name of the model used
<AREA> = the scaling factor of the BJT (determines how much current can flow through it)
<OFF> = an optional starting condition for DC analysis
<IC=VBE, VCE> = starting voltage in a simulation
<TEMP=T> = temperature of the transistor in Kelvin

MOSFET Component
    A MOSFET transistor is described by

```
MXXXXXXX ND NG NS NB MNAME <L=VAL> <W=VAL> <AD=VAL> <AS=VAL>
+ <PD=VAL> <PS=VAL> <NRD=VAL> <NRS=VAL> <OFF>
+ <IC=VDS, VGS, VBS> <TEMP=T>
```

The parameters are:
    ND I= the name of the drain terminal
    NG = the name of the gate terminal
    NS = the name of the source terminal
    NB = the name of the bulk (backgate) terminal
    MNAME = name of the model used
    <L=VAL> = length of the gate in meters
    <W=VAL> = width of the gate in meters
    <AD=VAL> = area of the drain contact in sqare meters
    <AS=VAL> = area of the source contact in sqare meters
    <PD=VAL> = perimeter of the drain contact in meters
    <PS=VAL> = perimeter of the source contact in meters
    <NRD=VAL> = equivalent squares that make up the drain to determine the drain resistance
    <NRS=VAL> = equivalent squares that make up the source to determine the source resistance
    <OFF> = an optional starting condition for DC analysis
    <IC=VDS, VGS, VBS> = starting voltage in a simulation
    <TEMP=T> = temperature of the transistor in Kelvin

Voltage Source Component
    A voltage source is described by

```
VXXXXXXX N+ N- <<DC> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>
+      <DISTOF1 <F1MAG <F1PHASE>>> <DISTOF2 <F2MAG <F2PHASE>>>
```

The parameters are:
    N+ I= the name of the positive terminal
    N- = the name of the negative terminal
    <<DC> DC/TRAN VALUE> = the DC offset of the voltage source
    <<AC> ACMAG <ACPHASE>>> = the AC magnitude and phase applied in an AC analysis
    <DISTOF1 <F1MAG <F1PHASE>>> = a distortion factor at frequency *F1*
    <DISTOF2 <F2MAG <F2PHASE>>> = a distortion factor at frequency *F2*

The DC value can be changed in time by using functions such as *pulse()*, *sin()*, *exp()*, and *pwl()*.
The distortion factors only operate with a .disto command.

Current Source Component
A current source is described by

```
IXXXXXXX N+ N- <<DC> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>
+      <DISTOF1 <F1MAG <F1PHASE>>> <DISTOF2 <F2MAG <F2PHASE>>>
```

The parameters are:
    N+ I= the name of the positive terminal
    N- = the name of the negative terminal
    <<DC> DC/TRAN VALUE> = the DC offset of the current source
    <<AC> ACMAG <ACPHASE>>> = the AC magnitude and phase applied in an AC analysis
    <DISTOF1 <F1MAG <F1PHASE>>> = a distortion factor at frequency *F1*
    <DISTOF2 <F2MAG <F2PHASE>>> = a distortion factor at frequency *F2*

The DC value can be changed in time by using functions such as *pulse()*, *sin()*, *exp()*, and *pwl()*.
The distortion factors only operate with a .disto command.

Current and Voltage Source DC Functions

Pulse Function
The pulse function is described by

```
PULSE(V1 V2 <TD> <TR> <TF> <PW> <PER>)
```

The parameters are:
    V1 = the initial value (volts or amps)
    V2 = the pulsed value (volts or amps)
    <TD> = the seconds before the first pulsed value
    <TR> = the seconds it takes the pulse to rise from V1 to V2
    <TF> = the seconds it takes the pulse to fall from V2 to V1
    <PW> = the number of seconds the signal stays at V2
    <PER> = the time between each rising edge of the pulse after the first initial pulse

Sinusoidal Function
The sinusoidal function is described by

```
SIN(V0 VA FREQ <TD> <THETA>)
```

The parameters are:
    V0 = the offset value (volts or amps)
    VA = the peak amplitude value (volts or amps), the peak-to-peak value is twice this
    FREQ = the frequency in Hz of the sinusoid
    <TD> = the seconds before the start of the sinusoid
    <THETA > = the damping factor of the sinusoid in 1/second

Exponential Function

The exponential function is described by

```
EXP(V1 V2 <TD1> <TAU1> <TD> <TAU2>)
```

The parameters are:

V1 = the initial value (volts or amps)
V2 = the pulsed value (volts or amps)
<TD1> = the seconds before the pulsed value
<TAU1> = the rise time constant for the pulse to rise from V1 to V2
<TD> = the seconds before the falling of the pulsed value
<TAU2> = the fall time constant for the pulse to fall from V2 to V1

Piece-Wise Linear Function

The piece-wise linear function is described by

```
PWL(T1 V1 <T2 V2 <T3 V3 <T4 V4 ...>>>)
```

The parameters are:

Tn = the time where the nth voltage is at the desired voltage
Vn = the nth voltage

Model Definition Commands

Generic Model Command

The generic model command is described by

```
.MODEL MNAME TYPE(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

The parameters are:

MNAME = the name to give the model
TYPE = the type of model (eg., D, NPN, PNP, NMOS, PMOS)
PNAMEn = the name of the parameter to be set
PVALn = the parameter's value

Diode Model (D)

The diode model command is described by

```
.model MNAME D(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

name	parameter	units	default	example	area
IS	saturation current	A	1.0e-14	1.0e-14	*
RS	ohmic resistance	Ω	0	10	*
N	emission coefficient	-	1	1.0	

name	parameter	units	default	example	area
TT	transit-time	sec	0	0.1ns	
CJO	zero-bias junction capacitance	F	0	2pF	*
VJ	junction potential	V	1	0.6	
M	grading coefficient	-	0.5	0.5	
EG	activation energy	eV	1.11	1.11 Si 0.69 Sbd 0.67 Ge	
FC	coefficient for forward-bias depletion capacitance formula	-	0.5		
BV	reverse breakdown voltage	V	infinite	40.0	
IBV	current at breakdown voltage	A	1.0e-3		
TNOM	parameter measurement temperature	°C	27	50	

BJT Model (NPN/PNP)

The NPN model command is described by

```
.model MNAME NPN(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

The PNP model command is described by

```
.model MNAME PNP(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

name	parameter	units	default	example	area
IS	transport saturation current	A	1.0e-16	1.0e-15	*
BF	ideal maximum forward beta	-	100	100	
NF	forward current emission coefficient	-	1.0	1	
VAF	forward Early voltage	V	infinite	200	
BR	ideal maximum reverse beta	-	1	0.1	
NR	reverse current emission coefficient	-	1	1	
VAR	reverse Early voltage	V	infinite	200	

name	parameter	units	default	example	area
RB	zero bias base resistance	Ω	0	100	*
RE	emitter resistance	Ω	0	1	*
RC	collector resistance	Ω	0	10	*
CJE	B-E zero-bias depletion capacitance	F	0	2pF	*
VJE	B-E built-in potential	V	0.75	0.6	
MJE	B-E junction exponential factor	-	0.33	0.33	
TF	ideal forward transit time	sec	0	0.1ns	
CJC	B-C zero-bias depletion capacitance	F	0	2pF	*
VJC	B-C built-in potential	V	0.75	0.5	
MJC	B-C junction exponential factor	-	0.33	0.5	
XCJC	fraction of B-C depletion capacitance connected to internal base node	-	1		
TR	ideal reverse transit time	sec	0	10ns	
CJS	zero-bias collector-substrate capacitance	F	0	2pF	*
VJS	substrate junction built-in potential	V	0.75		
MJS	substrate junction exponential factor	-	0	0.5	
EG	energy gap for temperature effect on IS	eV	1.11		
FC	coefficient for forward-bias depletion capacitance formula	-	0.5		
TNOM	Parameter measurement temperature	°C	27	50	

MOSFET Model (NMOS/PMOS)

The NMOS model command is described by

```
.model MNAME NMOS(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

The PMOS model command is described by

```
.model MNAME PMOS(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

name	parameter	units	default	example
LEVEL	model index	-	1	
VTO	zero-bias threshold voltage (VTO)	V	0.0	1.0
KP	transconductance parameter	A/V2	2.0e-5	3.1e-5
GAMMA	bulk threshold parameter (γ)	V1/2	0.0	0.37
PHI	surface potential (ϕ)	V	0.6	0.65
LAMBDA	channel-length modulation (MOS1 and MOS2 only) (λ)	1/V	0.0	0.02
RD	drain ohmic resistance	Ω	0.0	1.0
RS	source ohmic resistance	Ω	0.0	1.0
IS	bulk junction saturation current (IS)	A	1.0e-14	1.0e-15
CGSO	gate-source overlap capacitance per meter channel width	F/m	0.0	4.0e-11
CGDO	gate-drain overlap capacitance per meter channel width	F/m	0.0	4.0e-11
CGBO	gate-bulk overlap capacitance per meter channel length	F/m	0.0	2.0e-10
CJ	zero-bias bulk junction bottom cap per sq.-meter of junction area	F/m2	0.0	2.0e-4
MJ	bulk junction bottom grading coefficient.	-	0.5	0.5
CJSW	zero-bias bulk junction sidewall cap. per meter of junction perimeter	F/m	0.0	1.0e-9
MJSW	bulk junction sidewall grading coefficient.	-	0.50 (level1) 0.33 (level2, 3)	
JS	bulk junction saturation current per sq.-meter of junction area	A/m2		1.0e-8
TOX	oxide thickness	meter	1.0e-7	1.0e-7
NSUB	substrate doping	1/cm3	0.0	4.0e15

name	parameter	units	default	example
LD	lateral diffusion	meter	0.0	0.8μ
UO	surface mobility	cm2/Vs	600	700
VMAX	maximum drift velocity of carriers	m/s	0.0	5.0e4
FC	coefficient for forward-bias depletion capacitance formula	-	0.5	
TNOM	parameter measurement temperature	°C	27	50

Basic Simulation Commands

Set Initial Conditions (.IC)
In circuits with memory or with nodes that need to be set to initial values, .IC is helpful. When there are problems with convergence, .IC can be used to help the simulator find a region of convergence. The command is described by

```
.ic V(NODNUM)=VAL V(NODNUM)=VAL ...
```

The parameters are:
    NODNUM = the name of the node to be set
    VAL = the value of voltage to set it at

Operating Point Analysis (.OP)
This finds the steady state operating conditions of devices and nodes. The command is described by

```
.op
```

Small-Signal Gain Analysis (.TF)
This returns the gain computed using .op steady state values given output and input variables. The command is described by

```
.tf OUTVAR INSRC
```

The parameters are:
    OUTVAR = the output variable (ie, V(NODE1, NODE2) or I(VLOAD))
    INSRC = the voltage or current source at the input (ie, Vin or Iin)

Steady State DC Analysis (.DC)
This calculates the conditions of devices and nodes at a series of input conditions. The command is described by

```
.dc SRCNAM VSTART VSTOP VINCR [SRC2 START2 STOP2 INCR2]
```

The parameters are:

    SRCNAM = the name of the input current or voltage source to vary
    VSTART = the initial voltage of the input source
    VSTOP = the final voltage of the input source
    VINCR = the voltage between each input source voltage tested
    SRCNAM2 = the name of a second input current or voltage source to vary
    VSTART2 = the initial voltage of the second input source
    VSTOP2 = the final voltage of the second input source
    VINCR2 = the voltage between each second input source voltage tested

By specifying two sources, plots can be generated to see how both voltage sources effect each other.

Small-Signal AC Analysis (.AC)

        This calculates how the devices operate in the frequency domain with the small signal characteristics at a given DC operating point. The command is described by

```
.ac STYPE ND FSTART FSTOP
```

The parameters are:

    STYPE = a DECade, OCTave, or LINear step
    ND = the name of the oscillating input current or voltage source
    FSTART = the initial frequency of the input source
    FSTOP = the final frequency of the input source

Transient Analysis (.TRAN)

        This monitors the state of a circuit in time given a particular input. The output resembles what can be seen with oscilloscopes. The command is described by

```
.tran TSTEP TSTOP <TSTART <TMAX>>
```

The parameters are:

    TSTEP = the time between each *sample* in the simulation
    TSTOP = the stop time in the simulation which starts at 0 seconds.
    TSTART = the start time to save data for later analysis (useful when memory is limited)
    TMAX = the maximum step size that WinSpice3 uses

End of Simulation (.END)

        This tells Spice to ignore all commands and lines after this point, and to do the simulation. It usually goes at the end of the file. Make sure there is at least one blank line after it.

```
.end
```

Using WinSpice3

Download and Installation

A free (it still says free at the top of the page) and useful version of Spice is WinSpice, which can be found at "http://www.willingham2.freeserve.co.uk/winspice.html". This spice uses a command interpreter known to UNIX as Nutmeg and can display graphical plots, which can be printed by window capturing in Windows using the ALT-PrintScreen key combination. This is the version of Spice I describe for showing graphical plots because it is absolutely *free*.

At the website, go to *Spice3F4*, and click on the link *here* which should start the download. Once the file is downloaded, create a directory for the program, run the executible and carefully make sure the program is installed in the created directory. You are done!

Creating a Test Spice Netlist

Learning how to display plots in Nutmeg is easy once you know how to navigate it. First, in a text editor like notepad, make a sample circuit as follows:

```
Spice Simulation 1-1

*** MODEL Descriptions ***
.model nm NMOS level=2 VT0=0.7 KP=80e-6 LAMBDA=0.01

*** NETLIST Description ***
M1 vdd ng 0 0 nm W=3u L=3u
R1 in ng 50
Vdd vdd 0 5
Vin in 0 2.5

*** SIMULATION Commands ***
.op
.dc Vin 0 5 0.1 Vdd 0 5 0.5

.end
```
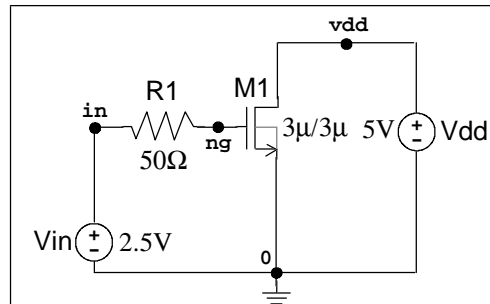

Figure 1: Schematic of Spice Simulation 1-1 Netlist

Using Nutmeg to View and Store Data

First run *wspice3.exe*. Using the menubar option, *file/open*, open the file with the above netlist. (Note that most netlists have the extension *.cir*.) The netlist and simulation commands have now been loaded into *WinSpice3*.

```
WinSpice3 1 -> cd
current directory: D:\Program Files\WinSpice3
WinSpice3 2 -> source test.cir
Reading .\test.cir

Circuit: Spice Simulation 1-1
```

To actually run the simulation, the *run* command must be typed at the command prompt.

```
WinSpice3 3 -> run
DC Operating Point ...
```

Now the simulation data is resident in memory and can be manipulated within Nutmeg.
Mathematical operations can be used on the vector data, and it can be plotted and written to files.
To see what type of data is available, type *display*.

```
WinSpice3 4 -> display
Here are the vectors currently active:

Title: Spice Simulation 1-1
Name: op1 (Operating Point)
Date: Wed Apr 25 08:09:25  2001

    in                  : voltage, real, 1 long
    ng                  : voltage, real, 1 long
    vdd                 : voltage, real, 1 long [default scale]
    vdd#branch          : current, real, 1 long
    vin#branch          : current, real, 1 long
```

Notice that the only data available right now is from the `.op` command line. Nutmeg separates
the different data from different simulations, so they must be selected using *setplot*. Type *setplot*
to view the different data groups available to us.

```
WinSpice3 5 -> setplot

        Type the name of the desired plot:

        new     New plot
Current op1     Spice Simulation 1-1 (Operating Point)
        dc1     Spice Simulation 1-1 (DC transfer characteristic)
        const   Constant values (constants)
```

Notice that both the .op and .dc data are listed. To choose the .dc data by typing *dc1* at the '?'
prompt. Now type *display* again to see the names of the data vectors.

```
WinSpice3 6 -> display
Here are the vectors currently active:

Title: Spice Simulation 1-1
Name: dc1 (DC transfer characteristic)
Date: Wed Apr 25 08:09:25  2001

    in                  : voltage, real, 561 long
    ng                  : voltage, real, 561 long
    sweep               : voltage, real, 561 long [default scale]
    vdd                 : voltage, real, 561 long
    vdd#branch          : current, real, 561 long
    vin#branch          : current, real, 561 long
```

Notice that certain values are current, and others are voltage. To plot IDS of the MOSFET
versus VGS with the plots for all values of VDS, type the command "*plot -vdd#branch vs ng*".

```
WinSpice3 8 -> plot -vdd#branch vs ng
```
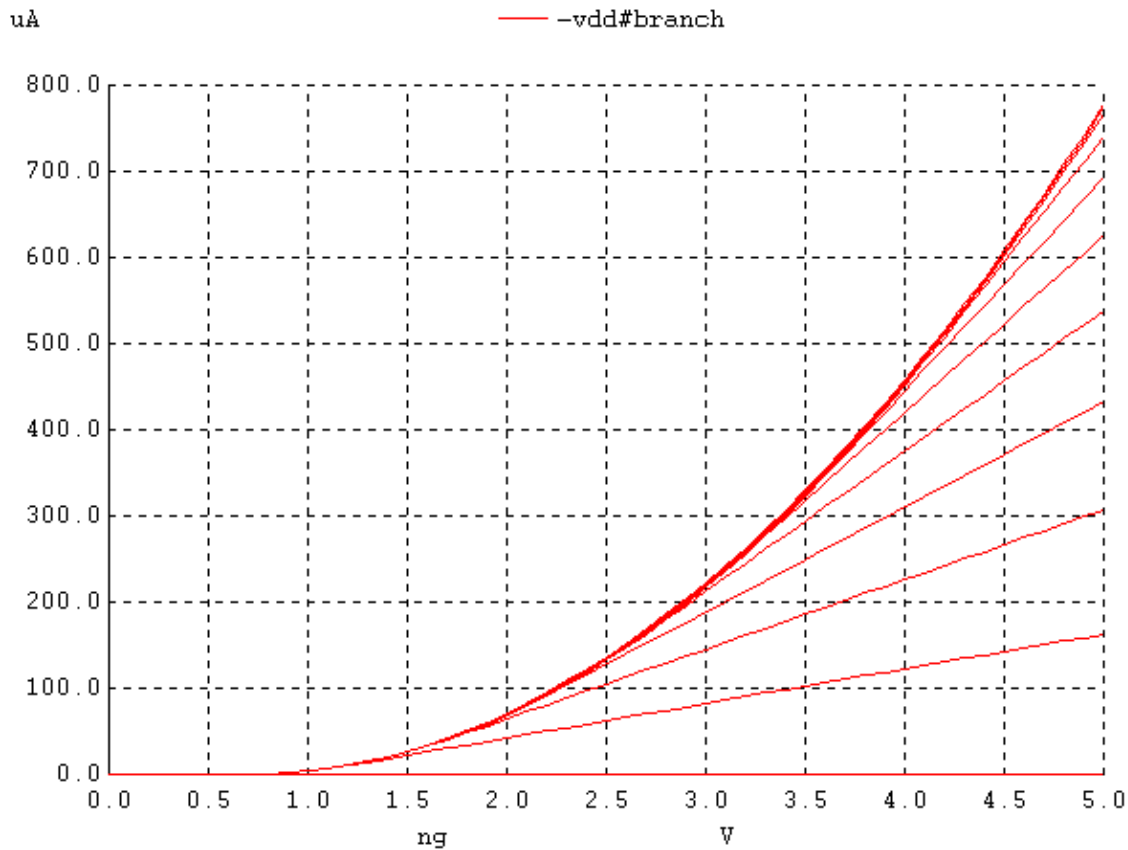
Let's say you want to save this data in case you need to plot it later, you can use *write*. By typing *write* followed by the filename to save the data in, you can save the selected plot data for later use. Type *write testdc.data* to save the current DC plot data.

```
WinSpice3 9 -> write testdc.data
```

Now if you have to load this data, you simply need to use the *load* command followed by the filename the data is saved in. Type *load testdc.data* to load the saved DC plot data.

```
WinSpice3 10 -> load testdc.data
Loading raw data file ("testdc.data") . . . done.
Title:  Spice Simulation 1-1
Name: DC transfer characteristic
Date: Thu Aug 16 12:53:27  2001

Here are the vectors currently active:

Title: Spice Simulation 1-1
)ame: dc2 (DC transfer characteristic
Date: Thu Aug 16 12:53:27  2001

    in                      : voltage, real, 561 long
    ng                      : voltage, real, 561 long
    ng                      : voltage, real, 561 long
```

```
    sweep               : voltage, real, 561 long [default scale]
    vdd                 : voltage, real, 561 long
    vdd#branch          : current, real, 561 long
    vin#branch          : current, real, 561 long
```

The loaded data is now the current plot data as well. Now, let's say you need to get the steady state operating conditions of the devices in the circuit. You can use *show*. If you type *show* by itself, it displays all of the devices with their values. You can also type *show* followed by the device(s) you want to look at if you don't want to see all of them. Type *show* now.

```
WinSpice3 31 -> show
 Mos2: Level 2 MOSfet model with Meyer capacitance model
 device            m1
 model             nm
        id   0.000136
       ibd  -1.93e-12
       ibs          0
        is  -0.000136
        ig          0
        ib  -1.93e-12
       vgs        2.5
       vds          5
       vbs          0
       von        0.7
     vdsat        1.8
        rs          0
        rd          0
        gm   0.000152
       gds   1.44e-06
       gmb          0
       cbd          0
       cbs          0
       cgs   1.04e-15
       cgd          0
       cgb          0

 Vsource: Independent voltage source
 device           vin         vdd
        dc        2.5           5
     acmag          0           0
         i  -1.73e-18 -0.000136
         p   4.34e-18  0.000682
```

        You may want to save this displayed data in a text file instead of seeing it scroll on the text entry display. This can be done using the '>' symbol (the same one used in UN*Xs). To save this data in a file now, type *show > testdv.out*.

```
WinSpice3 31 -> show > testdv.out
```

WinSpice3 Quick Reference

`source <filename>` - loads the given circuit netlist and commands into the simulator.
`run` - runs the simulation as specified in the circuit input file
`show devices ... : parameters ...` - Output the operating point device summary
`dc srcnam vstart vstop vincr [src2 start2 stop2 incr2]` - Like the DC
      simulation command, this runs a dc simulation at the command prompt.
`listing` - shows the circuit listing with line numbers
`display` - shows vector status
`reset` - terminate a simulation after a breakpoint (like a '.end')
`plot expr ... [vs expr] [x1 xlo xhi] [y1 ylo yhi]` - plot simulation results
`setplot [plotname]` - select which plot data to be made current
`write [file [expr ...]]` - write data to a file of the currently active plot data
`let varname = expr` - assign vector variables
`set [option] [option = value] ...` - set a variable
`@name[param]` - pi, e, c, i, kelvin, echarge, boltz, planck

Bibliography

        Most of the Spice information was taken in someway from:
[1] Mike Smith "WinSpice3 User's Manual" 25 October, 1999