GETTING STARTED WITH SPICE

ICAEN UNIX WORKSTATIONS

The version of SPICE on the ICAEN network is xspice. It is on the UNIX (HP) workstations, but is not on the NT workstations. This is essentially the same as OrCAD PSPICE described in Irwin and Wu, but without the graphical interface and schematic capture input. To use this version, you have to type in the input file by hand with a text editor.

To run xspice, enter

    xspice&

at the Unix prompt of an xterm window, and a menu will pop up. If you already have a SPICE input file, enter that file name into the **Working file:** panel or press the **Files...** button under the panel to browse for the file. If you do not have a file, press the **Edit SPICE File** button under ***SPICE 2G Operations...*** and an editor will pop up. When you are done entering your SPICE input, save it to a file and enter its name in the **Working file:** panel. Next, press the **Run SPICE 2G File** button; it will appear that nothing has happened because it does the analysis very quickly. Finally, press **Display Output** to see the output and **Print** to print it. **Graph Spice File** is an interface to ACE/gr (xmgr) which is an Unix graphing package. This interface gives much nicer plots that the native XSPICE plots, but is not as user friendly as it could be. Some help documents are at the bottom of the menu under ***SPICE 2G Help...***. We will not discuss the Spice 3f Operations here.

Some hints on using xspice

- Make sure you press ENTER at the end of the **.end** statement. SPICE will not recognize this as a line unless it has a carriage return/line feed.
- Do not have blank lines after the **.end** statement. SPICE will run, but you'll get an error.
- Do not have blank lines in the beginning of the file. SPICE will interpret the first blank line as the title and the title as a circuit description line.
- If names include a non-alphanumeric character (such as _ & ^ [ ] .), SPICE may accept it, flag it as an error, interpret it in a way you may not expect, or ignore it, depending on the character. Use non-alphanumeric characters at your own risk.
- Watch out where you put spaces, particularly in assigning values.
- Use the command

        .width out 80

    or

        .width out=80

    to keep the output width at a reasonable value.

RUNNING SPICE ON ICAEN FROM A MODEM

If you are running the X Windows System from your home computer, you should be able to run xspice. With only a telnet connection, you can still run SPICE from the UNIX prompt. To do this, add **/usr/ui/class/bin** to your PATH (See the ICAEN consultant if this means nothing to you.). Then you must create or edit your SPICE input file using one of the UNIX screen or line editors such as **vi** or **ed**. Run SPICE with the command

    hpspice

and the output file will be displayed. Or you can run

    hpspice output_filename

to capture the output in output_filename.

SPICE FOR YOUR PC OR MAC

You may want the convenience of running SPICE on your own computer, particularly if you plan to continue with electronics. Here is a list, which is undoubtedly incomplete, of places where you can get SPICE for your PC or Macintosh. Most have student discounts or free demo versions with limited capabilities (but enough for 57:008) that you can download. They typically require a few Mb of disk space. Almost all are part of a much bigger bundle of electronic design programs that you can also get demos for.

This list is merely for your convenience, and does not constitute an endorsement of any product. Absence of a product from the list does not mean that it is not any good; it just means I do not know about it. The version used in Irwin and Wu is OrCAD PSPICE for which you can get a demo CD from the vendor.

B^2SPICE2 has been installed on the ICAEN computers. Click here for instructions on how to use B^2SPICE2.

product name	operating system	Comments
B^2SPICE2	Macintosh, Windows	free student version installed on ICAEN
AIM-SPICE	Windows	demo/student version
ICAP/4Windows	Windows	demo
Interactive SPICE	Windows	demo, student version
Micro-Cap V	Windows	demo, student version
OrCAD PSPICE	Windows	demo CD
PSPICE, old versions	Windows, OS/2, Macintosh	freeware
SPICE	Linux	freeware

WHAT DOES *SPICE* STAND FOR?

- **S**imulation **P**rogram with **I**ntegrated **C**ircuit **E**mphasis (This is what SPICE really stands for.)
- **S**ick **P**ersons **I**dea of **C**erebral **E**ntertainment
- **S**pecial **P**rogram **I**ntended to **C**onfuse **E**ngineers
- **S**ociety for the **P**revention of **I**ntelligence in otherwise **C**ompetent **E**ngineers
- **S**imulation **P**rogram **I**ntended to **C**omplement **E**ngineering

---

Winston Chan
var dateObj = new Date(document.lastModified); document.writeln('This page was updated on ',dateObj.getMonth()+1,'-',dateObj.getDate(),'-',dateObj.getYear(),'.');

GETTING STARTED WITH SPICE

ICAEN UNIX WORKSTATIONS

The version of SPICE on the ICAEN network is xspice. It is on the UNIX (HP) workstations, but is not on the NT workstations. This is essentially the same as OrCAD PSPICE described in Irwin and Wu, but without the graphical interface and schematic capture input. To use this version, you have to type in the input file by hand with a text editor.

To run xspice, enter

    xspice&

at the Unix prompt of an xterm window, and a menu will pop up. If you already have a SPICE input file, enter that file name into the **Working file:** panel or press the **Files...** button under the panel to browse for the file. If you do not have a file, press the **Edit SPICE File** button under ***SPICE 2G Operations...*** and an editor will pop up. When you are done entering your SPICE input, save it to a file and enter its name in the **Working file:** panel. Next, press the **Run SPICE 2G File** button; it will appear that nothing has happened because it does the analysis very quickly. Finally, press **Display Output** to see the output and **Print** to print it. **Graph Spice File** is an interface to ACE/gr (xmgr) which is an Unix graphing package. This interface gives much nicer plots that the native XSPICE plots, but is not as user friendly as it could be. Some help documents are at the bottom of the menu under ***SPICE 2G Help...***. We will not discuss the Spice 3f Operations here.

Some hints on using xspice

- Make sure you press ENTER at the end of the **.end** statement. SPICE will not recognize this as a line unless it has a carriage return/line feed.
- Do not have blank lines after the **.end** statement. SPICE will run, but you'll get an error.
- Do not have blank lines in the beginning of the file. SPICE will interpret the first blank line as the title and the title as a circuit description line.
- If names include a non-alphanumeric character (such as _ & ^ [ ] .), SPICE may accept it, flag it as an error, interpret it in a way you may not expect, or ignore it, depending on the character. Use non-alphanumeric characters at your own risk.
- Watch out where you put spaces, particularly in assigning values.
- Use the command

  .width out 80

  or

  .width out=80

  to keep the output width at a reasonable value.

RUNNING SPICE ON ICAEN FROM A MODEM

If you are running the X Windows System from your home computer, you should be able to run xspice. With only a telnet connection, you can still run SPICE from the UNIX prompt. To do this, add **/usr/ui/class/bin** to your PATH (See the ICAEN consultant if this means nothing to you.). Then you must create or edit your SPICE input file using one of the UNIX screen or line editors such as **vi** or **ed**. Run SPICE with the command

    hpspice

and the output file will be displayed. Or you can run

    hpspice output_filename

to capture the output in output_filename.

SPICE FOR YOUR PC OR MAC

You may want the convenience of running SPICE on your own computer, particularly if you plan to continue with electronics. Here is a list, which is undoubtedly incomplete, of places where you can get SPICE for your PC or Macintosh. Most have student discounts or free demo versions with limited capabilities (but enough for 57:008) that you can download. They typically require a few Mb of disk space. Almost all are part of a much bigger bundle of electronic design programs that you can also get demos for.

This list is merely for your convenience, and does not constitute an endorsement of any product. Absence of a product from the list does not mean that it is not any good; it just means I do not know about it. The version used in Irwin and Wu is OrCAD PSPICE for which you can get a demo CD from the vendor.

B^2SPICE2 has been installed on the ICAEN computers. Click here for instructions on how to use B^2SPICE2.

product name	operating system	Comments
B^2SPICE2	Macintosh, Windows	free student version installed on ICAEN
AIM-SPICE	Windows	demo/student version
ICAP/4Windows	Windows	demo
Interactive SPICE	Windows	demo, student version
Micro-Cap V	Windows	demo, student version
OrCAD PSPICE	Windows	demo CD
PSPICE, old versions	Windows, OS/2, Macintosh	freeware
SPICE	Linux	freeware

WHAT DOES *SPICE* STAND FOR?

- **S**imulation **P**rogram with **I**ntegrated **C**ircuit **E**mphasis (This is what SPICE really stands for.)
- **S**ick **P**ersons **I**dea of **C**erebral **E**ntertainment
- **S**pecial **P**rogram **I**ntended to **C**onfuse **E**ngineers
- **S**ociety for the **P**revention of **I**ntelligence in otherwise **C**ompetent **E**ngineers
- **S**imulation **P**rogram **I**ntended to **C**omplement **E**ngineering

---

Winston Chan
var dateObj = new Date(document.lastModified); document.writeln('This page was updated on ',dateObj.getMonth()+1,'-',dateObj.getDate(),'-',dateObj.getYear(),'.');

THE SPICE INPUT FILE

Introduction

SPICE is a program that analyzes a circuit using nodal analysis. It can do many types of analysis, some of which are discussed in subsequent webpages and some of which are beyond the scope of this class. To get started, we will use SPICE to do what is known as an operating point analysis. This gives the voltage of every node and the current going through every voltage (dependent and independent) source.

From class, we know that the voltage across any element and the current through any element can be determined once all the node voltages have been found. It is simple to get SPICE to give any voltage difference, so getting the voltage across any element is no problem. Getting the current is not so easy with SPICE. If the element is a resistor, for example, you can get the current from the voltage using Ohm's law. SPICE does not let you do simple calculations like that, and it seems silly to have to use a calculator while using a computer. There is a trick to get any current in a circuit. The trick is to put a turned-off voltage source in series with each element for which you need to know the current. Because SPICE calculates the current through every voltage source and this source is in series with the element of interest, we get the current we want. And because the source is turned off (V = 0) it behaves like a short circuit and will not affect the circuit.

Your interaction with SPICE is through the input file, which is also known as the input deck, the SPICE deck (terms stemming from the days of computer cards) or the netlist. It is composed of a mandatory title, statements that describe the circuit using text, control statements for SPICE and comments. We describe the input file for XSPICE; except for some minor differences, the input file is similar for other versions of SPICE.

Title

Every input file begins with a single-line title. This line is mandatory. If you leave it out, SPICE will interpret the first line of the file, even if it is part of the circuit description, as the title. If this happens, you will probably get an error because your circuit description is incomplete. Perhaps a worse thing to happen is that the circuit description will be complete without the first element and SPICE will do the analysis on a different circuit than you think. You will get the wrong answer, but you will have no indication that anything has gone wrong. So, don't forget the title.

Circuit Description

The circuit that SPICE is to analyze is described by a series of statements, with one statement for each circuit element. These statements can be listed in any order (with some exceptions if you use subcircuits). They also can be intermixed with control statements, though we do not recommend doing this because it makes the file harder to understand. Each element statement basically tells SPICE three things: (1) the type and name of the element, (2) how it is connected to the circuit, and (3) its value. The details of how this is done depends on the type of element. We will go over independent and dependent sources and resistors here, and go over capacitors and inductors under transient analysis.

Preliminaries - Each node of the circuit is labeled with a number. Restrictions are

- There must be a reference node. This node must be numbered zero.
- The numbers have to be positive integers or zero. Although SPICE does the analysis for larger node numbers, the output properly prints only node numbers that are <1000. The practical limit for node numbers therefore is 999.
- The numbers do not have to be consecutive. With the exception of the reference node, there does not even have to be any rhyme or reason to how you pick the numbers.
- You cannot break a node up into two nodes. That is, a node must have a single number.
- There are some modifications to these rules when you use subcircuits.

Each element of the circuit is given a name. The name can be up to eight characters long. Its first character tells SPICE the type of element it is, so you have no choice for that character. The others can be any alphanumeric character. SPICE is case insensitive, so it will consider a name with upper case identical to the same name in lower case. The first characters for elements we consider here are

- R - resistor
- V - independent voltage source
- I - independent current source
- G - voltage-controlled current source
- E - voltage-controlled voltage source
- F - current-controlled current source
- H - current-controlled voltage source

Examples of valid resistor names are R, R1, R455, Rinput, Rin23 and RIN23. Note that the last two are the same to SPICE, and you will get an error if you try to name two different resistors with these names.

A numeric value can given by a variety of formats. As an example, consider the value 1000. It may be given by 1000, 1000.0, 1E3, 1.0e3, 10E2, 10000e-1, 1k, 0.001meg. (Note that E3 by itself is not allowed.) The last two use the standard scale prefixes:

T	tera	10^{12}
G	giga	10^{9}
MEG	mega	10^{6}
K	kilo	10^{3}
M	milli	10^{-3}
U	micro	10^{-6}
N	nano	10^{-9}
P	pico	10^{-12}
F	femto	10^{-15}

Notice that SPICE cannot adhere completely to the standard notation because of the case insensitivity and because of the lack of Greek symbols. Mistakes are commonly made with mega, milli and micro.

General rules about numeric values are that they must be written without any spaces (E.g., 1k is interpreted as 1000 whereas 1 k is interpreted as 1), they cannot have commas, and anything after what

SPICE interprets as a valid number is ignored. This lets you give units to the values. For example, you can write 1kohm and SPICE will interpret this as 1000. On other hand, this last rule does not let you mix exponents with scale prefixes. If you try to write 1000000 as 1e3k, SPICE will stop after 1e3 and will ignore the "k" at the end. Finally, SPICE does not accept zero as the value of components such as resistors, inductors and capacitors.

Resistors - The format of a resistor statement is

Rname N1 N2 Value

Rname is the name of the resistor as described above. N1 and N2 are the node numbers to which the resistor is connected. Since a resistor does not have a polarity, it does not matter which of the two nodes is N1 and which is N2. Value is the value of the resistance in ohms.

Independent voltage sources - The format of an independent voltage source statement is

Vname N+ N- <<DC> Value>

Vname is the name of the source. N+ is the node number that the positive side of the source is connected and N- is the node number that the negative side is connected. It is obviously important to specify the nodes in the right order. Parameters is angular brackets <...> are optional. DC is a label that tells SPICE that this is a DC voltage source (there are AC sources, pulse sources, etc. that we will consider later), and Value is its value in volts. The value may be positive, negative or zero. The default source type is a DC source and the default value is zero. Thus, if "DC Value" is left out, SPICE assumes that the voltage source is a DC source with zero volts. As discussed above, this is useful as a current monitor.

Independent current source - The format of an independent current source statement is

Iname N+ N- <<DC> Value>

Iname is the name of the source. N+ is the node number that the positive side of the source is connected and N- is the node number that the negative side is connected. The passive sign convention is followed here, so a positive current flows from N+ to N-. The order in which you list the nodes is obviously important. DC is a label that tells SPICE that this is a DC voltage source (there are AC sources, pulse sources, etc. that we will consider later), and Value is its value in amperes. The value may be positive, negative or zero. Like the voltage source, the default source type is DC and the default value is zero. If "DC Value" is left out, SPICE assumes that the current source is a DC source with zero current.

Voltage-controlled current source - The format of a voltage-controlled current source statement is

Gname N+ N- NC+ NC- Value

Gname is the name of the source. N+ and N- are the nodes to which this source is connected. The passive sign convention is obeyed, so positive current flows from N+ to N-. The controlling voltage is the voltage difference between nodes NC+ and NC-. Value is the proportionality constant between the controlling voltage and the output current, and it has dimensions of siemens.

Voltage-controlled voltage source - The format of a voltage-controlled voltage source statement is

Ename N+ N- NC+ NC- Value

Ename is the name of the source. N+ and N- are the nodes to which this source is connected. The controlling voltage is the voltage difference between nodes NC+ and NC-. Value is the proportionality constant between the controlling voltage and the output voltage, and it is dimensionless.

Current-controlled current source - The format of a current-controlled current source statement is

Fname N+ N- Vmonitor Value

Fname is the name of the source. N+ and N- are the nodes to which this source is connected. The passive sign convention is obeyed, so positive current flows from N+ to N-. The controlling current is the current flowing through the independent voltage source Vmonitor. Vmonitor is usually a dummy source that is inserted just to monitor the current, so it is usually turned off (V = 0). Value is the proportionality constant between the controlling current and the output current, and it is dimensionless.

Current-controlled voltage source - The format of a current-controlled voltage source statement is

Hname N+ N- Vmonitor Value

Hname is the name of the source. N+ and N- are the nodes to which this source is connected. The controlling current is the current flowing through the independent voltage source Vmonitor. Vmonitor is usially a dummy source that is inserted just to monitor the current, so it is usually turned off (V = 0). Value is the proportionality constant between the controlling current and the output voltage, and it has dimensions of ohms.

Control statements

SPICE control statements begin with a period (.) at the beginning of the line (no leading spaces or tabs). They generally tell SPICE what to do and how to format the output. Fortunately, SPICE has reasonable defaults so you do not have to include many control statements. Some of the more common and useful control statements are:

.*END* - The one mandatory control statement is the .END statement that goes at the very end of the netlist. A common error is to forget the carriage return/line feed at the end of the line.

.*WIDTH* - A very handy control statement is

.width out 80

This limits the output file to 80 characters in width, the standard size for printing. Without this, long lines will wrap around and the output can be very confusing.

.*OP* - This tells SPICE to do an operating point analysis. SPICE automatically does this analysis and prints out the results if you tell it to do nothing.

.*DC* - In a DC analysis, SPICE "sweeps" an independent source between two values. The format is

.DC Srcname Start Stop Increment

The source Srcname (either an independent voltage source or an independent current source) will take on values between Start and Stop taking steps of size Increment. The value of the source in the source definition statement is ignored. The default output for the DC analysis is no output, so you have to use either the .PRINT or .PLOT control statements to define the output.

.*PRINT* - This defines the output from a SPICE analysis. The print output is a listing of voltages and currents that you select with this statement. The format is

.PRINT DC Out1 Out2 ...

The DC option tells SPICE that you want the printout of a DC analysis. The selected outputs are given for each value that the source took in the .DC analysis. The parameters (there can be up to eight) Out1, Out2... can be a node voltage for which you enter V(Nodenumber), the difference between two node voltages for which you enter V(N1, N2) or V(N1 N2), or a current going through an independent voltage source for which you enter I(Vname). Thus, a .PRINT statement will look like this:

```
.PRINT DC V(4) V(8,2) I(Vtest1) I(Vtest2)
```

.*PLOT* - This gives an ASCII plot instead of a listing. The format is exactly the same as .PRINT with .PLOT as the first word. To obtain higher quality plots, you will have to use .PRINT to generate a list of data points and then export these data points to a separate graphing program.

Comments

Comments in the SPICE netlist begin with an asterisk in the first position in a line (no leading blanks or tabs).

Winston Chan

B2 Spice

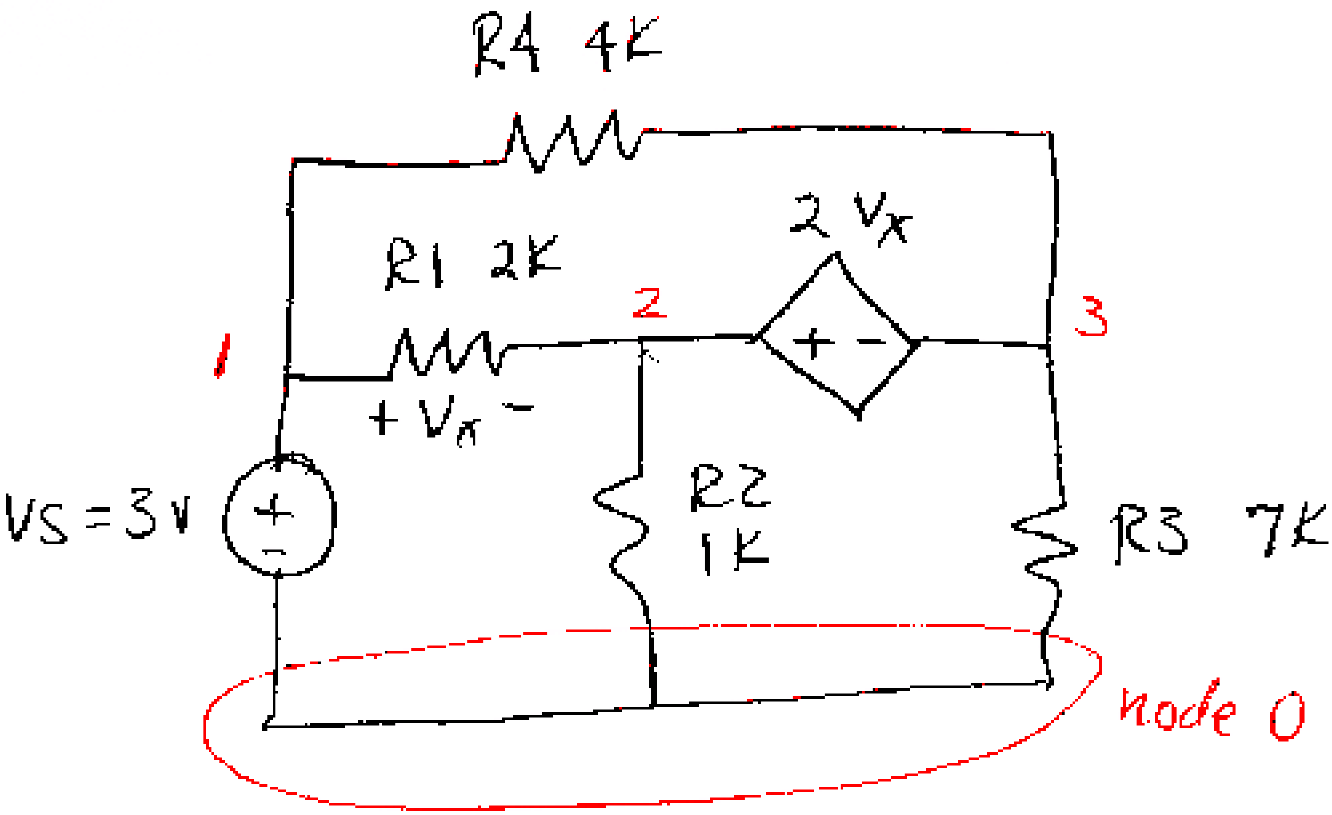
B2 Spice can be found under Start Menu -> Programs -> Engineering Software -> Electrical.

In order to use B2 Spice, the parts database must be copied to a different directory. On the ICAEN computers, this can be done by copying the Parts icon from the Parts Data directory and moving it to either the E drive or the H drive. B2 Spice can then be run by choosing the B2 Spice Workshop icon in the Electrical folder. The first time B2 Spice is run, a folder will pop up for the B2 Spice A_D v4 Lite directory. The path must be changed to the directory in which the Parts icon was moved. Once in this directory, click on the Parts icon to start B2 Spice. In subsequent use, B2 Spice can simply be run by clicking on the B2 Spice Workshop icon.

This program provides several Tutorial examples that can be found in Help Contents under the Help tab.

BASIC DC ANALYSIS

The circuit we will use:



Basic input with default analysis and output

example
.width out 80
vs 1 0 dc 3
r1 1 2 2k
r2 2 0 1k
r3 3 0 7k
r4 1 3 4k
e1 2 3 1 2 2
.end

By default, this does an operating point (.op) analysis and prints out all of the node voltages, all of the independent voltage source currents, and all of the dependent source voltage and current.

Add a current monitor to sense current through R1 (Changes to basic input file shownin red.):

example
.width out 80

```
vs 1 0 dc 3
r1 4 2 2k
r2 2 0 1k
r3 3 0 7k
r4 1 3 4k
e1 2 3 1 2 2
vmeas 1 4 dc 0
.end
```

The monitor is between node 1 and R1.

Scan input *vs* between 0 and 5 volts in 0.5 volt increments (Changes to basic input file shown in red.):

```
example
.width out 80
vs 1 0 dc 3
r1 1 2 2k
r2 2 0 1k
r3 3 0 7k
r4 1 3 4k
e1 2 3 1 2 2
.dc vs 0 5 0.5
.print dc v(3), v(3,1)
.end
```

Because we are specifying an analysis mode (DC), SPICE no longer assumes the default mode (OP). We have to specify what we want SPICE to output. Here, we tell it to print out node voltage 3 and the voltage difference between nodes 3 and 1 (which is the voltage across R4 with node 3 as positive).

Instead of printing columns of numbers, we have SPICE generate an ASCII plot of the V(3) and V(3,1) (Changes to the basic input in shown in red.):

```
example
.width out 80
vs 1 0 dc 3
r1 1 2 2k
r2 2 0 1k
r3 3 0 7k
r4 1 3 4k
e1 2 3 1 2 2
.dc vs 0 5 0.5
.plot dc v(3), v(3,1)
.end
```

When you **Display Output**, you will get an ASCII plot of V(3) and V(3,1). If you press **Graph Spice File** on the XSPICE menu, it will invoke xmgr, a graphics package on Unix. You must be using the .plot command in SPICE; using .print will give you a blank plot. The interface between SPICE and xmgr is not best. If you have a .plot command with more than one variable as we have in this example, xmgr will plot only the first and ignore the rest. You can plot several variables by having a separate .plot command for each variable. Unfortunately, xmgr reads all of the data points into a single array (set) so you will not be able to plot the variables differently (e.g., different colors, different linewidths, or different symbols) and you may get extra lines from the re-trace. To get a plot with several variables, each with its own distinctive symbol, it may be easier to use the SPICE .print command, cut and paste to get the data from the output file into a separate file for each variable, and use xmgr manually to read these files. After entering the data, you still need to do some work, like label the axes, before getting a professional quality plot.

SPICE can also output independent voltage source currents (and only independent voltage source currents) in the DC mode. Here, we add the current for the independent source to the output. (Changes to the basic input are shown in red.):

```
example
.width out 80
vs 1 0 dc 3
r1 1 2 2k
r2 2 0 1k
r3 3 0 7k
r4 1 3 4k
e1 2 3 1 2 2
.dc vs 0 5 0.5
.print dc v(3), v(3,1), i(vs)
.end
```

Winston Chan

MORE DC ANALYSIS

THEVENIN AND NORTON EQUIVALENT CIRCUITS

We can use SPICE to help us obtain a Thevenin or Norton equivalent circuit. Let us consider the circuit (a)

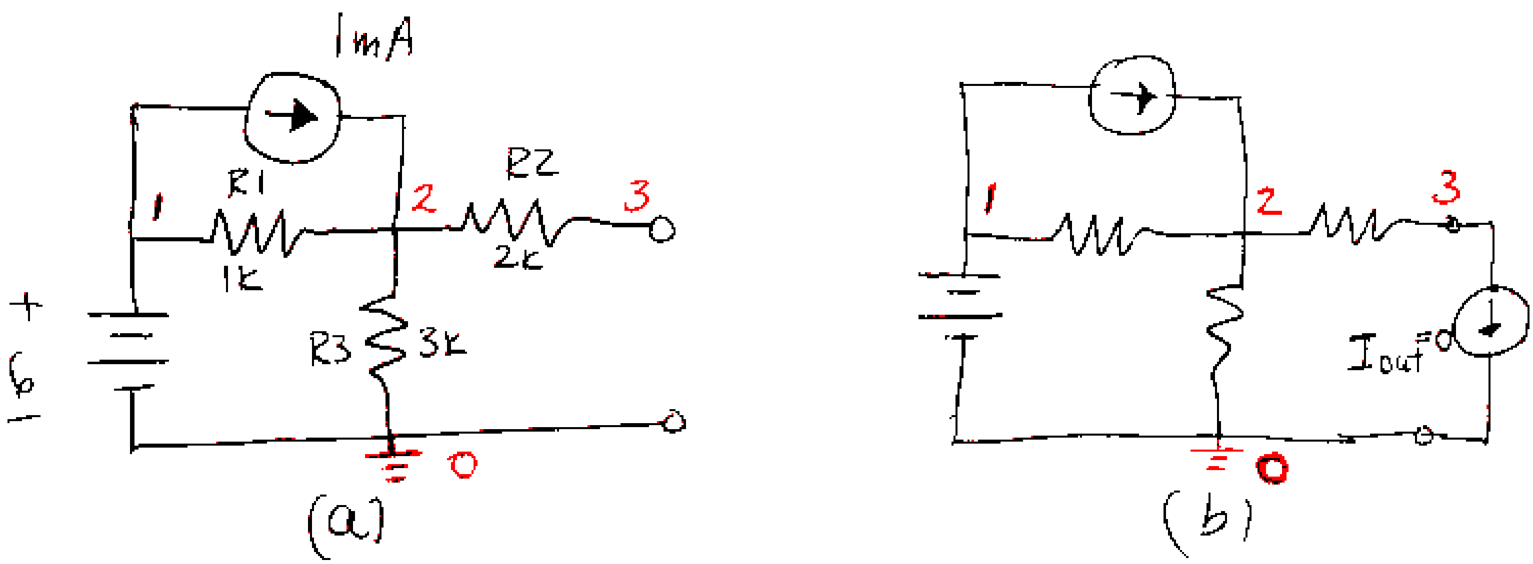


Fig. 1

We want to find a voltage source plus resistor (Thevenin equivalent circuit) or a current source plus resistor (Norton equivalent circuit) that will behave exactly the same way between nodes 3 and 0 as the original circuit. To do this, we need V_{oc} and I_{sc}.

Obtaining the open circuit voltage - The SPICE input for the circuit (a) is

```
WRONG voc determination
.width out 80
vs 1 0 DC 6
r1 1 2 1k
r2 2 3 2k
r3 2 0 3k
is 1 2 1m
.end
```

If we run this, we get an error because there is only one thing attached to node 3. SPICE does not allow a resistor with infinite resistance, so the way around this is to attach a current source with no current between nodes 3 and 0. A turned-off current source is an open circuit. This is a variation on the trick of using a voltage source with no voltage to fool SPICE into giving us a current through a short. The correct file, corresponding to the circuit (b), is

```
CORRECT voc determination
.width out 80
vs 1 0 DC 6
r1 1 2 1k
r2 2 3 2k
r3 2 0 3k
is 1 2 1m
iout 3 0
.end
```

Note that we left out the "DC 0" in the iout statement. SPICE defaults to a DC source with no output for all sources. We look at the output file, and the voltage at node 3 is V_{oc}.

Obtaining the short circuit current - In the same manner, we can determine I_{sc} by using a voltage source with zero voltage at the output.

```
isc determination
.width out 80
vs 1 0 DC 6
r1 1 2 1k
r2 2 3 2k
r3 2 0 3k
is 1 2 1m
vout 3 0
.end
```

We look at the output file, and the current through the voltage source vout is I_{sc}.

With V_{oc} and I_{sc}, we can obtain the Thevenin and Norton equivalent circuits for circuit (a). There is another way of doing it, but we need to go over the transfer function command first.

TRANSFER FUNCTION

A transfer function is the ratio of an output voltage or current to an input voltage or current. A simple example of a transfer function is the voltage divider formula.

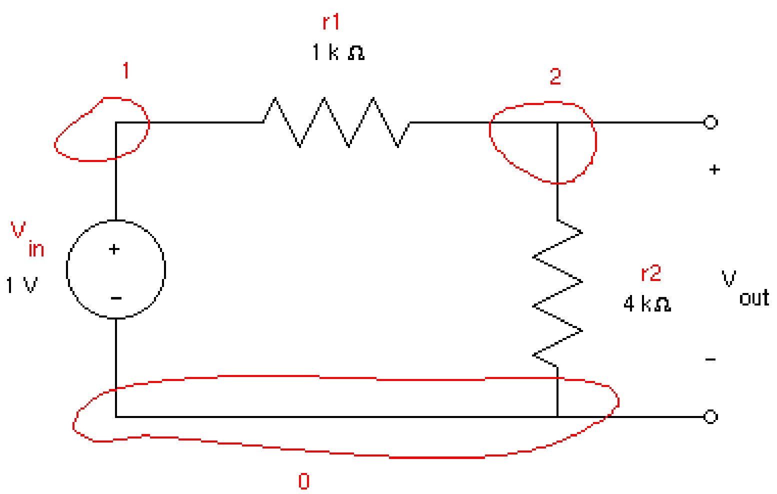


Fig. 2

The transfer function of the circuit in Fig. 2 is V_{out}/V_{in} = R2/(R1 + R2). To get this with SPICE, we can use the transfer function command:

```
voltage divider
.width out 80
vin 1 0 DC 1
r1 1 2 1k
r2 2 0 4k
.tf v(2) vin
.end
```

The line with .tf tells SPICE to do a transfer function analysis with the first variable (v(2)) as the output and the second (vin) as the input. SPICE still does an operating point analysis. When we run this, we get near the end of the output file the lines

```
0 v(2)/vin = 8.000E-01
0 input resistance at vin = 5.000E+03
0 output resistance at v(2) = 8.000E+02
```

The first line gives the value of the transfer function which in this case is r2/(r1 + r2) = 0.8. The second line gives the equivalent (Thevenin or Norton) resistance looking into the circuit where the input is connected. The voltage source vin "sees" r1 in series with r2, so the input resistance is 5k. The last line gives the output resistance which is the Thevenin or Norton resistance looking

back at the circuit between the output nodes 2 and 0 *when the input source, but not any other sources that happen to be in the circuit, is turned off*. Since the input is a voltage source, it becomes a short circuit when it is off. There are no other sources in the circuit so the output resistance is r1 in parallel to r2, or 800 ohms.

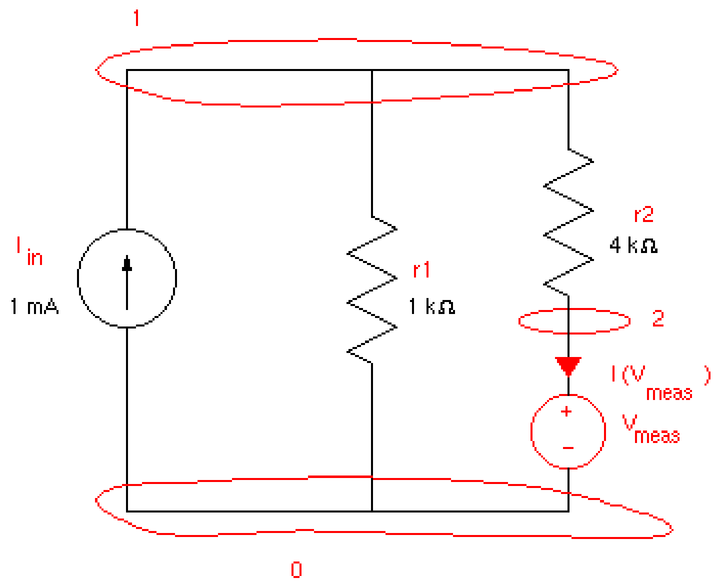


Fig. 3

Let us consider the current divider (Fig. 3). The SPICE input is current divider
.width out 80
iin 0 1 DC 1m
r1 1 0 1k
r2 1 2 4k
vmeas 2 0
.tf i(vmeas) iin
.end

We did some things here that may not be obvious. First, note the way the current source is put in to give us 1 mA going in the direction indicated in Fig. 3. Second, to measure the output current (the current going through r2), we need to use the old trick of putting a turned-off voltage source (vmeas) in series with r2. Finally, we use i(vmeas) to tell SPICE the output that we want is the current going through the voltage source vmeas.

The output gives the lines

0 i(vmeas)/iin = 2.000E-01
0 input resistance at iin = 8.000E+02
0 output resistance at i(vmeas) = 5.000E+03

The transfer function for a current divider is i_{out}/i_{in} = r1/(r1 + r2) = 0.2 as indicated in the first line. The input resistance is r1 in parallel with r2, which is 800 ohms. The output resistance is the resistance look back between the output nodes 2 and 0. We happened to have shorted the output with vmeas to measure the output current so we have to ignore this in determining the output resistance. Looking between nodes 2 and 0 without vmeas and with the input current source turned off (open circuited since it is a current source), we see r1 in series with r2. The output resistance is thus r1 + r2 = 5k.

MORE ON THEVENIN AND NORTON EQUIVALENT CIRCUITS

Knowing about the transfer function, we can obtain the Thevenin or Norton equivalent circuits with one run. To get the Thevenin equivalent, we need the open circuit voltage and equivalent resistance. We do this by placing a turned-off current source across the output nodes to get V_{oc} as we did above. We use .tf to give us the resistance. With the same circuit as in the top of this page, we add the current source iout and the .tf statement with iout as the input. Do not be bothered by the fact that the input for the transfer function is turned off; conceptually, what SPICE does to get the transfer function is to increase the input by a small amount and see how this increase affects the output. The output voltage or current for .tf does not matter here. This is because we are not really interested in a transfer function, but are using .tf to obtain the input resistance. We only need to tell SPICE that the input is iout, and we'll just arbitrarily pick v(3) as the output.

```
thevenin example
.width out 80
vs 1 0 DC 6
r1 1 2 1k
r2 2 3 2k
r3 2 0 3k
is 1 2 1m
iout 0 3
.tf v(3) iout
.end
```

In the output file, the node voltage v(3) is V_{oc} and the Thevenin resistance is given as the input resistance at iout. Do not be confused by the fact SPICE calls this an input resistance - we are adding an input iout at the circuit output nodes to get SPICE to calculate the Thevenin resistance.

The Norton equivalent circuit is obtained by adding a turned-off voltage source (vout) at the output nodes and doing a transfer function analysis

```
norton example
.width out 80
vs 1 0 DC 6
r1 1 2 1k
r2 2 3 2k
r3 2 0 3k
is 1 2 1m
vout 3 0
.tf v(3) vout
.end
```

The current going through the voltage source vout is I_{sc} and the input resistance at vout is the equivalent resistance.

SUBCIRCUITS

A very useful feature of SPICE that is not discussed in Irwin is a subcircuit. For those familiar with computer programming, a subcircuit is analogous to a subroutine or a function in a program. A subcircuit has the same advantages as a subroutine in that

- **it saves typing the same thing over and over,**
- **the file is easier to understand and modify because it has some structure to it,**
- **you can make a change in the subcircuit and the change will be implemented everywhere you use the subcircuit, and**
- **subcircuits files that you can use for your analysis are available from vendors of many electronics parts including integrated circuits, sensors and speakers.**

There are two parts in using subcircuits. First, the subcircuit has to be defined. This is a SPICE listing between the lines

```
.subckt SUBCKT_NAME N1 [N2 N3 ...]
[subcircuit definition goes here]
.ends
```

Here, SUBCKT_NAME is the name of the subcircuit you are defining, and N1, N2, N3 ... are the names of the nodes that the subcircuit connects to the outside world. At least one node is required by SPICE (though it is hard to imagine of a useful subcircuit with just one external node). The only restriction is that a subcircuit cannot use the reference node 0. If a subcircuit node is to be connected to the reference node, it has to be done later. Since each subcircuit has a name, you can have many subcircuits within the SPICE input file. The second part is calling a subcircuit in your main circuit. This is done with

```
Xyyyyyyy M1 [M2 M3 ...] SUBCKT_NAME
```

This is in the same format as the resistor and source definition. The first item is the component name which must begin with X or x. It is followed by the nodes that this subcircuit is attached to in the main circuit. So, M1 is connected to what is known as N1 in the subcircuit definition, and so on. If a node in the subcircuit is grounded, the corresponding M is zero. Finally, SUBCKT_NAME tells SPICE which subcircuit you are using.

An important feature is that the node numbering in the subcircuit definition is independent of the node numbering in the main circuit. The only way they get matched up is in the call to the subcircuit: the first node in the call gets matched to the first node in

the definition, and so on down the line. This feature lets you define your subcircuit independent of the main circuit, and in particular, you do not have to worry about conflicts in node numbering between the subcircuit and the main circuit. Moreover, in the subcircuit definition, the component names apply only within the subcircuit. If you have a R1 in both the subcircuit and the main circuit, or in two subcircuits, they are different entities to SPICE. In this way, you do not have to worry about name conflicts when using subcircuits.

An example using subcircuits is a circuit with two op amps.

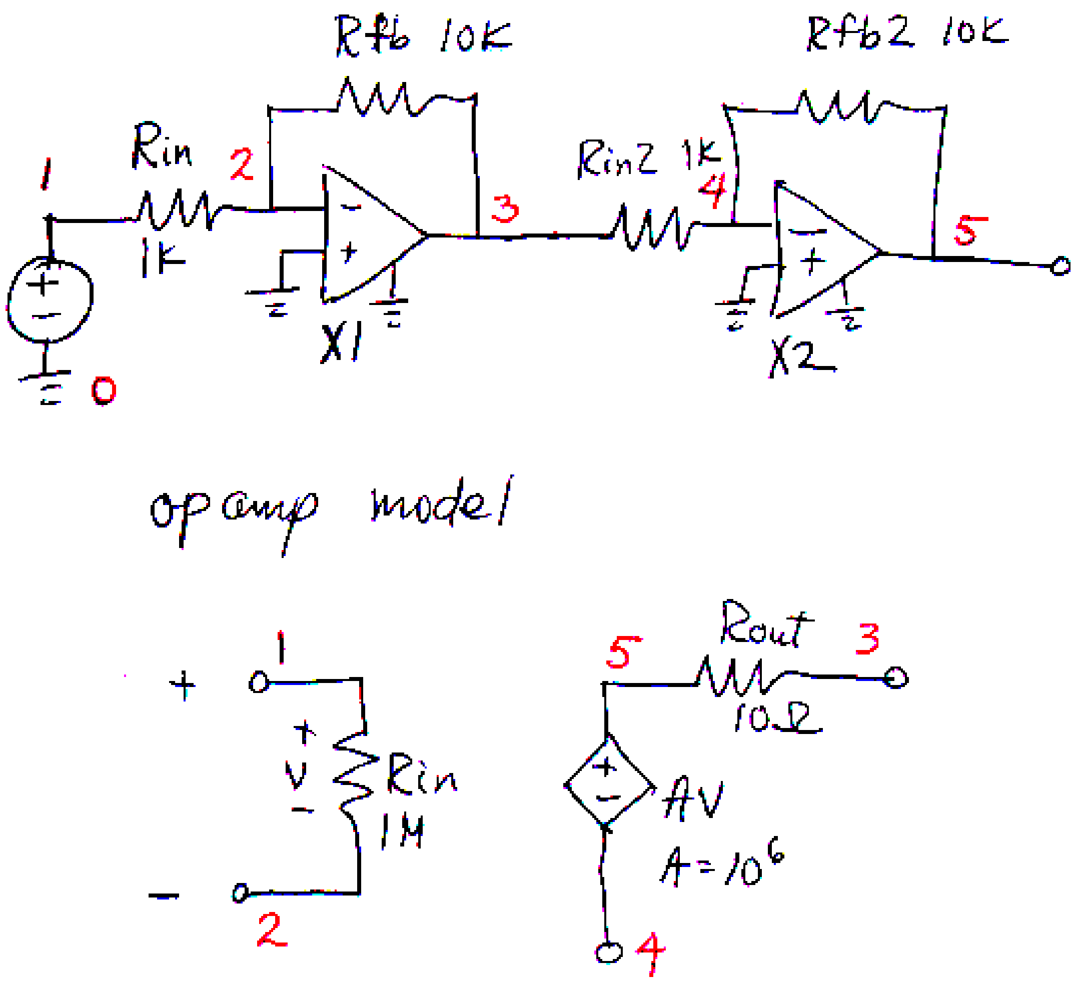


Fig. 4

The subcircuit for the op amp is a simple model (Irwin and Wu, Fig. 3.25c). The SPICE file for this circuit is (the subcircuit statements are highlighted in red):

```
subcircuit example
.width out 80
* subcircuit definition begins here
.subckt opamp 1 2 3 4
rin 1 2 1meg
rout 5 3 10
eamp 5 4 1 2 1e6
.ends
* subcircuit definition ends here
vin 1 0
rin 1 2 1k
rfb 2 3 10k
*op amp x1 is here
x1 0 2 3 0 opamp
rin2 3 4 1k
```

```
rfb2 4 5 10k
*op amp x2 is here
x2 0 4 5 0 opamp
.dc vin -1m 1m 0.1m
.print dc v(2) v(3) v(4) v(5)
.end
```

In this example, the input voltage vin is swept from -1 mV to + 1 mV in 0.1 mV steps, and all of the node voltages (of the main circuit) are printed out. Notice that the subcircuit has the same node numbers that the main circuit has, but they are not necessarily the same nodes. Notice also that two resistors are named rin, one in the op amp and the other in the main circuit, but they are not the same resistor.

The internal node 5 of the subcircuit is available. The convention for labeling these internal nodes is straightforward. To figure out the convention, do an operating point analysis (.OP) and you will see a couple of unexpected nodes in the listing. These are the internal nodes. By varying the number of internal nodes in the subcircuit definition and varying the calls to subcircuits, it is easy to see how SPICE labels these nodes.

Winston Chan

SPICE TRANSIENT ANALYSIS

Capacitors and Inductors

Before we begin to use SPICE for transient and AC analysis, we need to know how to tell SPICE about capacitors and inductors. Like other components, the first letter of the name tells SPICE the type of component it is. A capacitor is labelled with a name beginning with C, and an inductor is labelled with a name beginning with L. The format for a capacitor is

Cname N+ N- Value <IC=V(t=0)>

Here Cname is the name of the capacitor and N+ and N- are the two nodes that it is connected to. The Value is in farads. There is an optional (things between "<" and ">" are optional) term IC=V(t=0) that tells SPICE the value of the capacitor voltage at t = 0. (SPICE ignores spaces on either side of the equal sign.) This option will be used in transient analysis. It is important to keep track of the nodes because the initial voltage is defined positive at node N+ with respect to node N-. If you mistakenly switch N+ and N-, SPICE will analysis a circuit with the negative of the initial voltage that you thought you put in. Similarly, the format for an inductor is

Lname N+ N- value <IC=I(t=0)>

The Value is in henries. The initial condition is the inductor current (in amperes). The passive sign convention applies, so a positive current flows from N+ to N-.

An example of a 5.1 microfarad capacitor between nodes 2 and 3 with an initial voltage of 2 V is

C23 2 3 5.1u IC=2

Switched Circuit Analysis

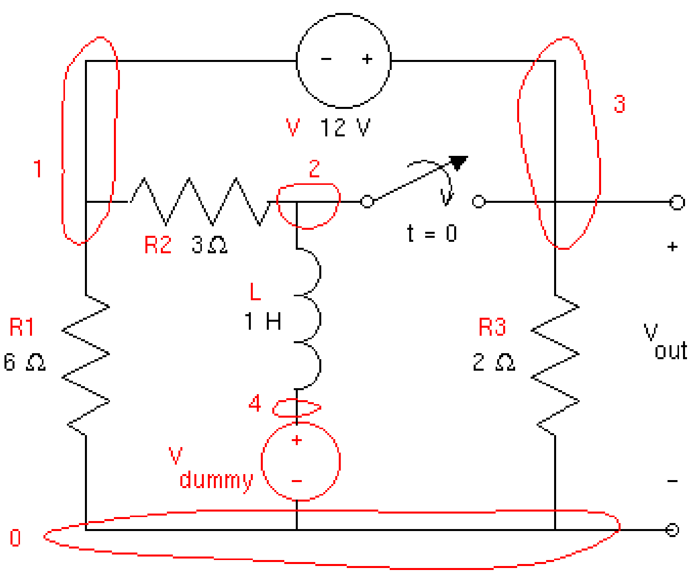


Fig. 1

We will analyze the above circuit as an example. Using the step-by-step approach, we obtain v_{out} = 6 exp(-1.5 t) when t >= 0. Let's find v_{out} using SPICE. The SPICE input file we begin with is

```
transient test - switch opened, t<0
.width out 80
v 3 1 12
r1 1 0 6
r2 2 1 3
r3 3 0 2
L 2 4 1
vdummy 4 0
iswitch 2 3
.end
```

One of the first thing you probably noticed is that we do not yet know how to implement a switch using SPICE. There is no switch in XSPICE so we will have to improvise. (Some more recent versions of SPICE have switches as circuit elements.) The brute force way of implementing a switch is to draw two circuits, one with the switch open and one with the switch closed. We then would create two input files, one for each circuit. There is a complication in doing this: notice that nodes 2 and 3 become the same node when the switch is closed. SPICE does not allow two names for the same node and SPICE does not allow a zero ohm resistor between nodes 2 and 3. So, it appears that we have to rename the nodes if we wish to continue in this way. While renaming nodes is conceptually easy, it is a step where simple mistakes can creep in. A trick to avoid renaming nodes is to insert a turned-off voltage source between nodes 2 and 3. A turned-off voltage source has zero volts dropped across it for any current, so it behaves exactly as a short circuit. Likewise, we can use a turned-off current source for an open switch. We can open and close the switch simply by changing the source between a turned-off current source and a turned-off voltage source. With this trick, the only difference between the circuit with an open switch and the circuit with a closed switch is one source. We can do this with a single input file, and use an editor to change the source back and forth between a voltage source and a current source.

The input file shown is an ordinary DC analysis file except we have an inductor in the file. We include a dummy voltage source to monitor the inductor current and we use a turned-off current source *iswitch 2 3* as an open switch.

SPICE does the default OP analysis when you run this. This is equivalent to finding the voltages and source currents much after all of the sources have been turned on and all the switches have been flipped. Thus, it is equivalent to finding the voltages and source currents at $t = 0^-$. The output file gives a value of 6 for v(3) (which is v_{out}) and -2 for the current going through vdummy (which is i_L).

To find the voltages and source currents after the switch has been flipped, we use this SPICE input file

```
transient test - switch closed, t>0
.width out 80
v 3 1 12
r1 1 0 6
r2 2 1 3
r3 3 0 2
L 2 4 1 ic=-2
vdummy 4 0
vswitch 2 3
.tran 200m 10 uic
.plot tran i(vdummy) v(3)
.end
```

The changes from the first file are highlighted in red. The inductor statement now includes *ic=-2* which tells SPICE that the inductor has an initial value (which to SPICE is at t = 0) of -2 A. The default initial value is 0. Note that, unlike for a resistor, how you label the two nodes is important; the current is defined positive going from node 2 to node 4 in this case.

The next change is changing *iswitch* to *vswitch*. This changes the switch from a turned-off current source (open) to a turned-off voltage source (short).

The next line has the format

    .tran TSTEP TSTOP uic

tells SPICE we wish to do a transient analysis. SPICE essentially integrates the differential equations numerically starting at t = 0 with time increments of TSTEP and stopping at TSTOP. The number TSTEP (equal to 200m in the example) tells SPICE that we would like to step through the transient analysis in 200 msec steps and the number TSTOP (equal to 10 in the example) tells SPICE to stop the analysis at 10 sec. Unfortunately, there is no autoscale. Choosing these TSTEP and TSTOP to get a reasonable analysis requires either using trail and error or independently calculating the time constant. The time constant can be calculated using SPICE to find the Thevenin resistance as we already have learned to do. The characters *uic* at the end of the line mean "use initial conditions." They tell SPICE to use the initial condition given in the inductor statement. The default if *uic* is not included is that SPICE will do a OP analysis for t = 0 to determine the initial conditions before doing a transient analysis. This is appropriate if the only changes in the circuit occur in the sources. It is not appropriate in this example because a switch has flipped at t = 0, changing the circuit.

Finally, since we tell SPICE to do a transient analysis, we have to tell SPICE what output we want. The choices are the same as in a DC analysis - we can get a listing of the voltages and source currents that we are interested in or we can get an ASCII plot. The difference here is that the we use *.plot tran ...* instead of *.plot dc ....* This tells SPICE that the plot has time rather than an input voltage or current as the independent variable. The rest of the statement is exactly the same as for the DC analysis.

Time Varying Sources

In addition to the constant voltage and current sources that we have been using, SPICE has independent sources that vary with time in several ways. We will discuss only the pulse source.

As an example, we consider a series RC circuit with a pulse voltage source. The SPICE input file is

```
step source
```

```
.width out 80
V 1 0 pulse(0  1)
R 1 2 1k
C 2 0 10u
.tran 2m 50m
.plot tran v(2) v(1)
.end
```

We do a transient analysis and plot the capacitor voltage v(2) and the source voltage v(1). Note that since we do not include *uic* in the .tran statement, SPICE does an OP analysis to determine the initial value of the capacitor voltage to be 0 V. The initial capacitor voltage that it determined is printed in the output file.

The voltage source is a SPICE pulse source. Though we use a pulse voltage source in this example, a pulse current source is exactly analogous. The pulse source has seven parameters which are defined in Fig. 2. The full pulse function has the format:

PULSE(V0 V1 TD TR TF PW PER)

where the parameters have the following meaning:

- V0 - initial voltage (or current)
- V1 - pulse voltage (or current)
- TD - time delay (from t = 0)
- TR - rise time
- TF - fall time
- PW - pulse width
- PER - period

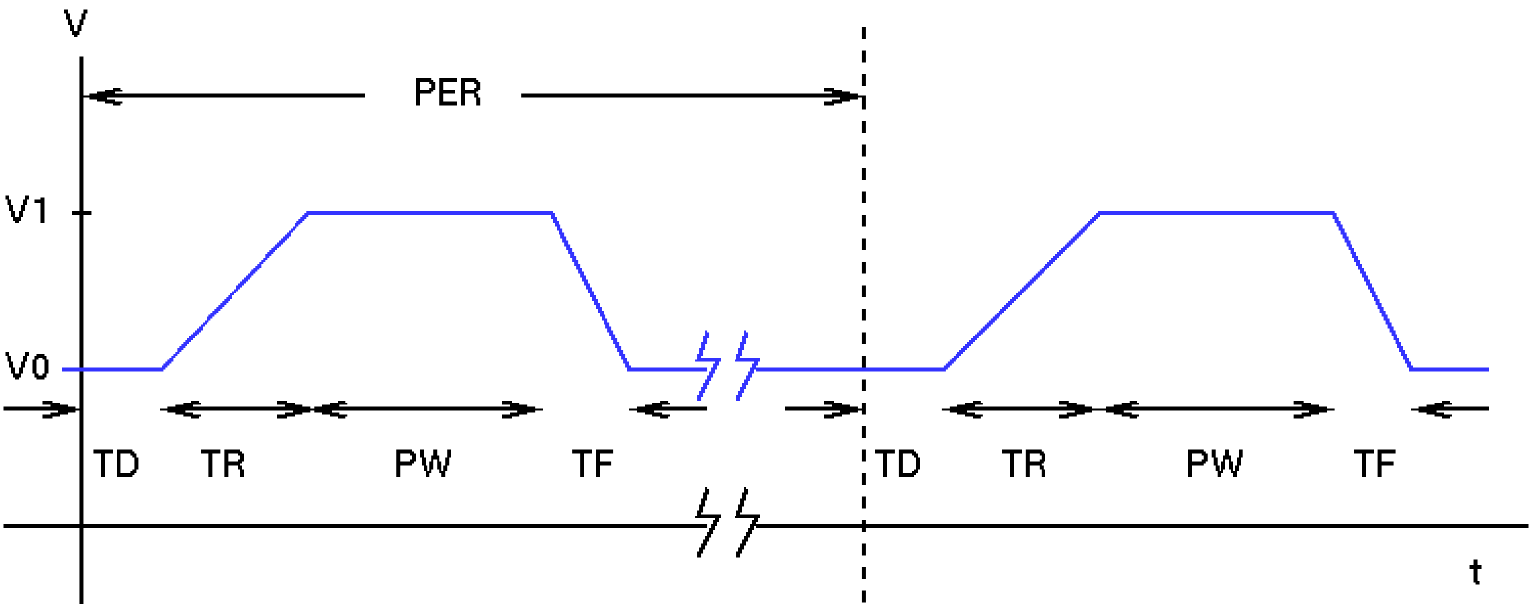


Fig. 2

The units for the first two parameters are volts or amperes, and the units for the others are second. The default values are

Parameter	Default value
V0	input required
V1	input required
TD	0
TR	TSTEP
TF	TSTEP
PW	TSTOP
PER	TSTOP

where TSTEP and TSTOP are obtained from the .tran statement. Fortunately, most of the parameters have reasonable default values so that if we supply only the first two parameters, as we do in the example, the source is a step function going from V0 to V1 at t = 0.

The parameters in the pulse definition are not necessarily in the order in which you would want to change them from their default values. For example, to make the source into a pulse rather than a step, we want to change the pulse width parameter. But in the parameter list, you need to specify the delay, rise and fall times before specifying the pulse width. One way to force the delay, rise and fall times to remain at their default values is to use a minus sign in their places (separated by spaces) in the parameter list. This works with ICAEN's xspice, but may or may not with other versions of SPICE. To be safe, you should specify values for all parameters up to the last one you change unless you verify for yourself that a minus sign will work. The SPICE file for a pulse 8 msec long is

```
pulse source - not all versions of SPICE
.width out 80
V 1 0 pulse(0  1  -  -  -  8ms)
R 1 2 1k
C 2 0 10u
.tran 2m 50m
.plot tran v(2) v(1)
.end
```

The SPICE file that would work more generally would be

```
pulse source
.width out 80
V 1 0 pulse(0  1  0  2m  2m 8ms)
R 1 2 1k
C 2 0 10u
.tran 2m 50m
.plot tran v(2) v(1)
.end
```

Other types of time varying sources are available in SPICE. We will not describe them here but simply list them for reference. They are SIN (sinusoidal), EXP (exponential), PWL (piece-wise linear) and SFFM (single-frequency frequency modulation). Each has its own set of parameters that you can change. These parameters are listed in the **Quick Reference** and **Manual** sections of the XSPICE menu.

Winston Chan

SPICE AC ANALYSIS

AC analysis requires defining the AC source(s), specifying the frequency range and specifying the output. The format is much like that for DC analysis. Keep in mind that SPICE does the AC analysis one frequency at a time, although it can automatically do it for a series of frequencies.

Defining the AC sources - The format for an AC source is

VNAME N+ N- AC <AMPLITUDE> <PHASE>
INAME N+ N- AC <AMPLITUDE> <PHASE>

for an independent voltage and current source, respectively. Here, quantities within the angular brackets <...> are optional. With only the AC label (i.e., no entry for AMPLITUDE and PHASE), SPICE assumes an amplitude of 1 V for a voltage source and 1 A for a current source, and a phase of zero degrees. With the AC label and an amplitude only, SPICE assumes a phase of zero degrees. The phase, if you enter it, is in degrees. Since only phase differences are physically relevant, it is often convenient to define zero phase as one of the sources.

Obviously, there should be at least one AC source for a meaningful AC analysis. **However, not all sources should be AC sources in an AC analysis.** A dummy voltage source that measures a current should *not* be an AC source even though we may want to measure an AC current. The statement

VDUMMY 5 9 AC WRONG

is wrong. It applies a 1 V AC signal between nodes 5 and 9, not something we want the dummy source to do. Some correct ways of defining a dummy source are

VDUMMY 5 9
VDUMMY 5 9 AC 0
VDUMMY 5 9 DC 0 AC 0
VDUMMY 5 9 DC 0
VDUMMY 5 9 DC

A source may be both AC and DC (and transient as well). This simply means that SPICE uses the DC value in the source definition for analysis requiring a DC value (DC, TF, OP and TRAN analysis) and the AC value for AC analysis. This feature is useful for more advanced SPICE analysis (not covered in this class) involving nonlinear elements.

Specifying the frequency range - SPICE does the AC analysis one frequency at a time, but you can get it to do a series of frequencies. This is done with the .AC statement which has the format

.AC OPT N FREQSTART FREQSTOP

The first parameter OPT of the .AC statement is one of three options: LIN, DEC and OCT. The parameter N has a different meaning for the various parameters and will be described below. FREQSTART and FREQSTOP are the first (lowest) and last (highest) frequencies (given in Hertz) in the sequence. Restrictions on the last two parameters are

0 < FREQSTART <= FREQSTOP

Note that you cannot do a zero frequency AC analysis (which is really a DC analysis). You can fool SPICE into doing it by using a very low frequency, but you should be sure that the frequency you picked is low enough. What is low enough is beyond the scope of this class, but operationally you can check it by using a frequency ten times smaller and making sure the results are the same.

The option LIN (for linear) gives a sequence of frequencies that are evenly spaced between FREQSTART and FREQSTOP. The number of frequencies in the sequence is given in the next parameter N in the statement. For example,

```
.AC    LIN    5    10    20
```

will give the frequencies 10, 12.5, 15, 17.5 and 20 Hz.

The option DEC (for decade) gives a sequence of frequencies where the frequencies increase by the same ratio. The range is between FREQSTART and FREQSTOP. The parameter N gives the number of points per decade (a decade is a factor of 10 increase in frequency) so the ratio between frequencies is the Nth root of 10. The statement

```
.AC    DEC    3    1    100
```

will give frequencies 1, 2.154 (= $10^{1/3}$), 4.642 (= $10^{2/3}$), 10, 21.54, 46.42 and 100 Hz.

Finally, the option OCT (for octave) is similar to DEC except N is the number of points per octave (an octave is a factor of 2 increase in frequency). The ratio between frequencies is the Nth root of 2. The statement

```
.AC    OCT    3    10    40
```

will give frequencies 10, 12.60 (= $10*2^{1/3}$), 15.87 (= $10*2^{2/3}$), 20, 25.20, 31.75 and 40 Hz.

Specifying the output - The output statement is either a .PRINT or .PLOT statement. It is similar to the one we used in doing a voltage scan or a transient analysis with a couple of differences. The first is that the statement must call the AC option:

```
.PRINT AC V(4) V(6,7) I(Vdummy)
.PLOT AC V(4) V(6,7) I(Vdummy)
```

This tells SPICE that the independent variable is frequency. The second difference comes from the fact that the voltages and currents are not simply real numbers but are phasors with a magnitude and phase (or equivalently, real and imaginary parts). The format in the above statements gives the magnitude of the voltage at node 4, the magnitude of the voltage difference between nodes 6 and 7, and the magnitude of the current going through the independent voltage source Vdummy. We can append the letters M, P, R or I to V or I to give the magnitude, phase, real part and imaginary part, respectively, of that voltage or current. Thus, VP(4) will give the phase (in degrees) of the voltage at node 4, IR(Vdummy) will give the real part of the current going through the independent voltage source Vdummy, and VI(6,7) will give the imaginary part of the voltage difference between nodes 6 and 7.

Example

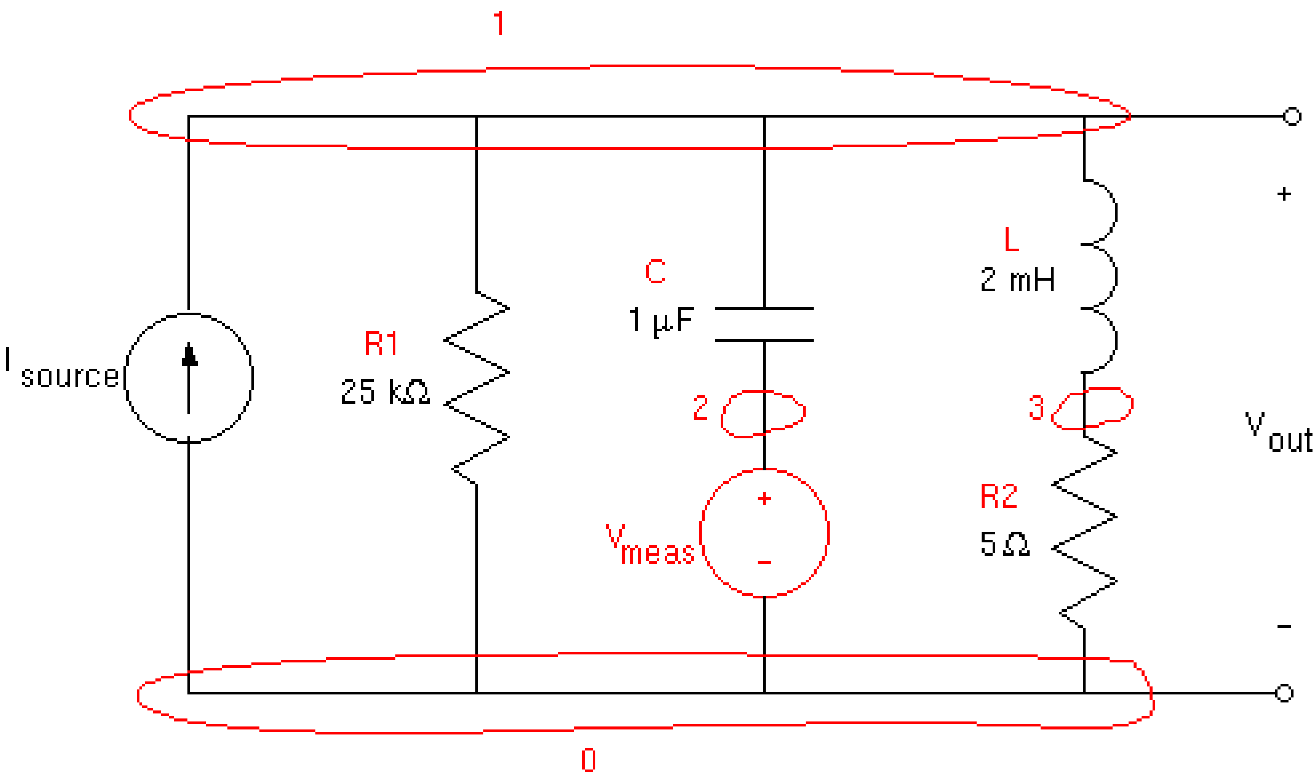


Fig. 1

We wish to find the magnitude and phase of the capacitor current, the magnitude of the output voltage, and the real part of the voltage across the inductor when the current source has an amplitude of 1 A. We have to insert a turned-off voltage source Vmeas in series with the capacitor to measure the capacitor current. The frequency of interest is between 1 kHz and 10 kHz.

```
AC ANALYSIS EXAMPLE
.width out 80
Isource 0 1 ac
C 1 2 1u
Vmeas 2 0
R1 1 0 25k
L 1 3 2m
R2 3 0 5
.ac dec 30 1k 10k
.plot ac im(vmeas) ip(vmeas) vm(1) vr(1,3)
.end
```

Winston Chan