



Ahsanullah University of Science and Technology
Department of Electrical and Electronic Engineering

LABORATORY MANUAL
FOR
ELECTRICAL AND ELECTRONIC SESSIONAL COURSE

Student Name :

Student ID :

Course no : EEE 1210
Course Title : Electrical Circuit Simulation Lab

For the students of
Department of Electrical and Electronic Engineering
1st Year, 2nd Semester

1. INTRODUCTION

SPICE is a powerful general purpose analog and mixed-mode circuit simulator that is used to verify circuit designs and to predict the circuit behavior. This is of particular importance for integrated circuits. It was for this reason that SPICE was originally developed at the Electronics Research Laboratory of the University of California, Berkeley (1975), as its name implies:

Simulation Program for Integrated Circuits Emphasis.

PSpice is a PC version of SPICE (which is currently available from OrCAD Corp. of Cadence Design Systems, Inc.). A student version (with limited capabilities) comes with various textbooks. The OrCAD student edition is called PSpice AD Lite. Information about Pspice AD is available from the OrCAD website: <http://www.orcad.com/pspicead.aspx>

The PSpice Light version has the following limitations: circuits have a maximum of 64 nodes, 10 transistors and 2 operational amplifiers.

SPICE can do several *types of circuit analyses*. Here are the most important ones:

- Non-linear DC analysis: calculates the DC transfer curve.
- Non-linear transient and Fourier analysis: calculates the voltage and current as a function of time when a large signal is applied; Fourier analysis gives the frequency spectrum.
- Linear AC Analysis: calculates the output as a function of frequency. A bode plot is generated.
- Noise analysis
- Parametric analysis
- Monte Carlo Analysis

In addition, PSpice has analog and digital libraries of standard components (such as NAND, NOR, flip-flops, MUXes, FPGA, PLDs and many more digital components,). This makes it a useful tool for a wide range of analog and digital applications.

All analyses can be done at different temperatures. The default temperature is **300K**.

The circuit can contain the *following components*:

- Independent and dependent voltage and current sources
- Resistors

- Capacitors
- Inductors
- Mutual inductors
- Transmission lines
- Operational amplifiers
- Switches
- Diodes
- Bipolar transistors
- MOS transistors
- JFET
- MESFET
- Digital gates
- and other components (see users manual).

2. PSpice with OrCAD Capture (release 9.2 Lite edition)

Before one can simulate a circuit one needs to specify the circuit configuration. This can be done in a variety of ways. One way is to enter the circuit description as a text file in terms of the elements, connections, the models of the elements and the type of analysis. This file is called the SPICE input file or source file and has been described somewhere else (see <http://www.seas.upenn.edu/%7Ejan/spice/spice.overview.html>).

An alternative way is to use a schematic entry program such as OrCAD CAPTURE. OrCAD Capture is bundled with PSpice Lite AD on the same CD that is supplied with the textbook. Capture is a user-friendly program that allows you to capture the schematic of the circuits and to specify the type of simulation. Capture is not only intended to generate the input for PSpice but also for PCD layout design programs.

The following figure summarizes the different steps involved in simulating a circuit with Capture and PSpice. We'll describe each of these briefly through a couple of examples.

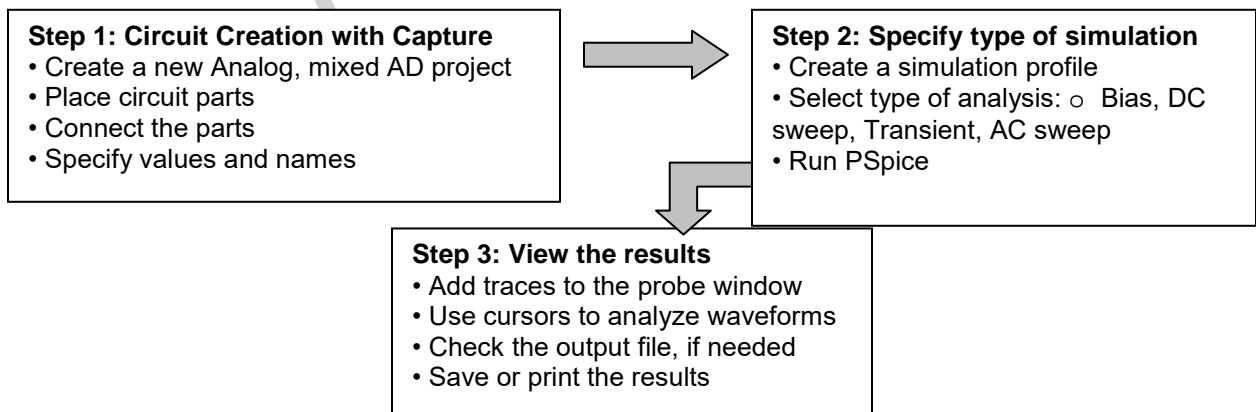


Figure 1: Steps involved in simulating a circuit with PSpice.

The values of elements can be specified using scaling factors (upper or lower case):

T or Tera (= 1E12); G or Giga (= E9); MEG or Mega (= E6); K or Kilo (= E3); M or Milli (= E-3);	U or Micro (= E-6); N or Nano (= E-9); P or Pico (= E-12) F or Femto (= E-15)
---	--

Both upper and lower case letters are allowed in PSpice. As an example, one can specify a capacitor of 225 pifofarad in the following ways:

225P, 225p, 225pF; 225pFarad; 225E-12; 0.225N

Notice that Mega is written as MEG, e.g. a 15 megaOhm resistor can be specified as 15MEG, 15MEGohm, 15meg, or 15E6. Be careful not to use M for Mega! When you write 15Mohm or 15M, Spice will read this as 15 milliOhm!

We'll illustrate the different types of simulations for the following circuit:

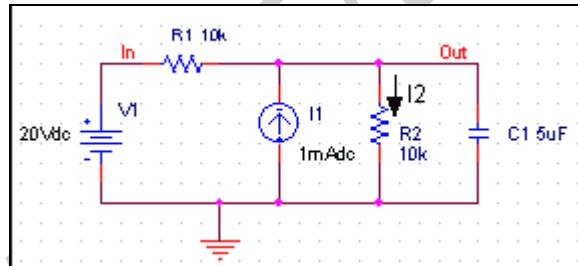
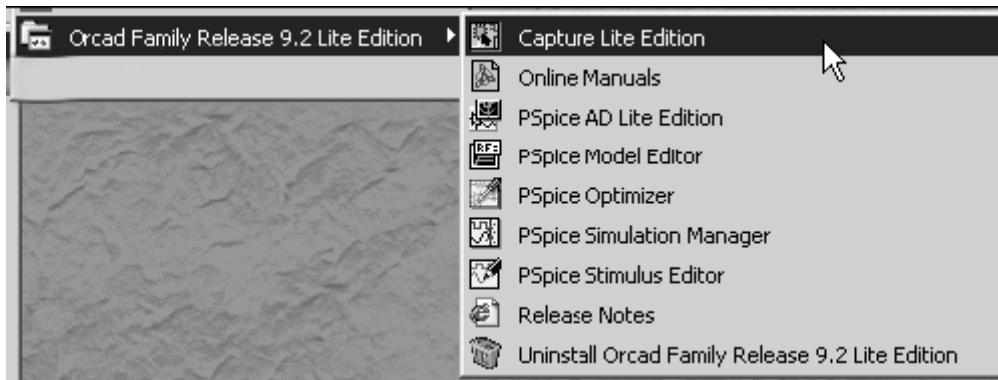


Figure 2: Circuit to be simulated (screen shot from OrCAD Capture).

2.1 Step 1: Creating the circuit in Capture

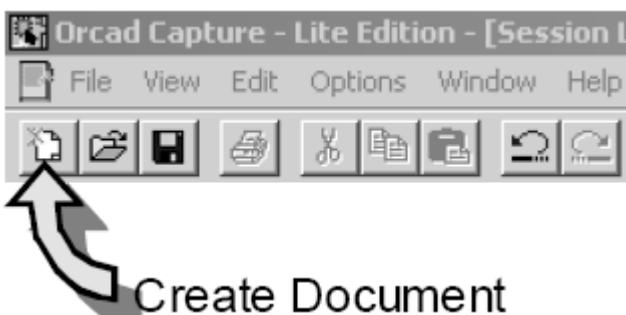
2.1.1 Create new project:

1. Start OrCAD Lite as shown below:

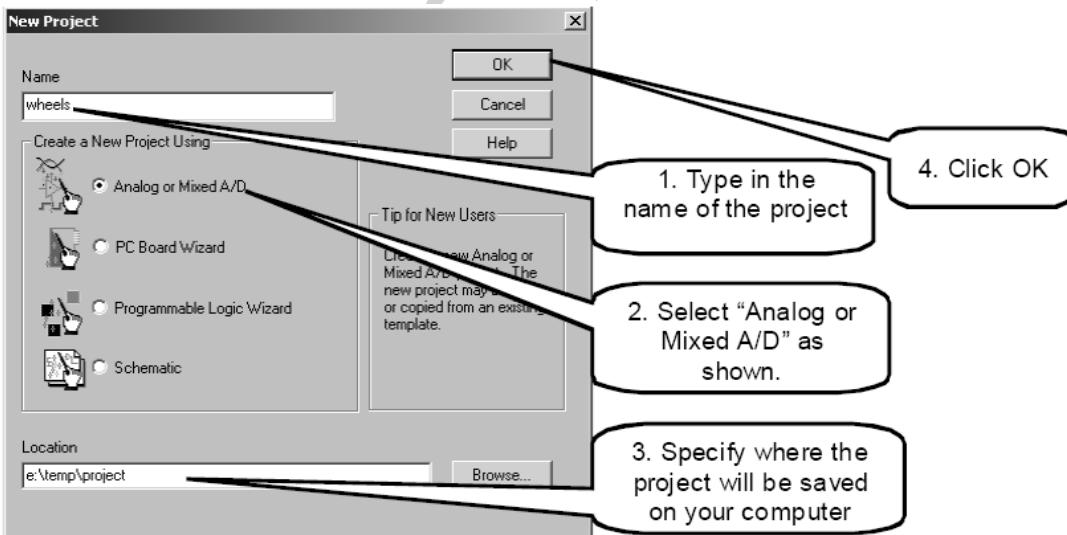


Note that the only program you need to start is “Capture Lite Edition.” The other programs will run automatically as they’re needed.

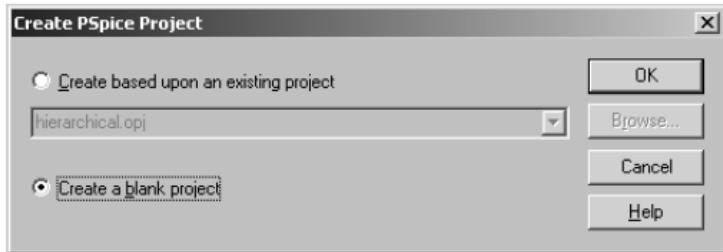
2. Click the “Create Document” button on the toolbar.



3. The following dialog will appear. Fill in the information as shown.



4. The following dialog will appear:



Choose “Create a blank project” and click OK.

5. The schematic editor will open. Enter the following schematic into the program:

A new page will open in the Project Design Manager as shown below.

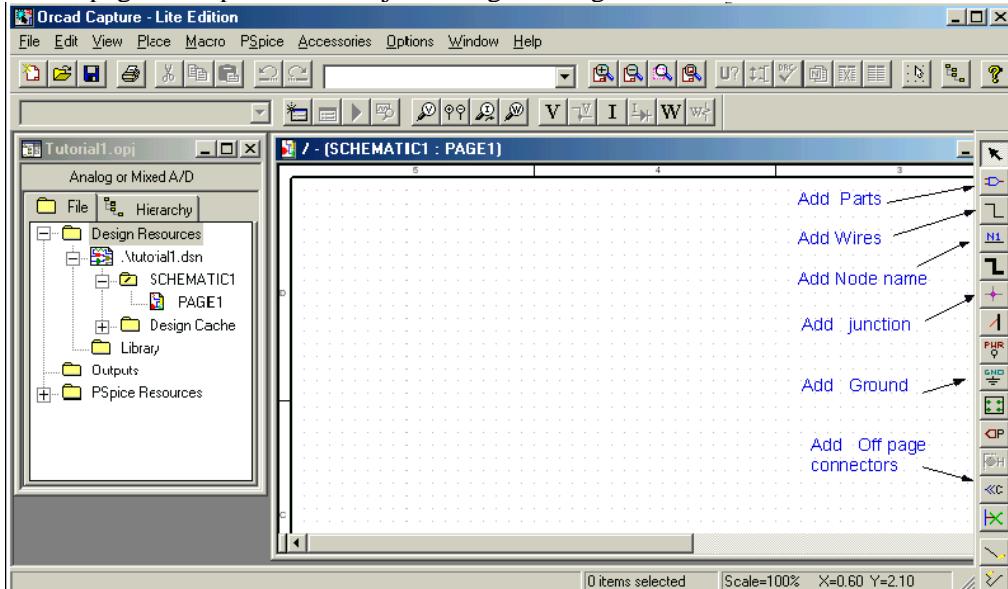


Figure 3: Design manager with schematic window and toolbars (OrCAD screen capture)

2.1.2. Place the components and connect the parts

1. Click on the Schematic window in Capture.
2. To Place a part go to PLACE/PART menu or click on the Place Part Icon. This will open a dialog box shown below.

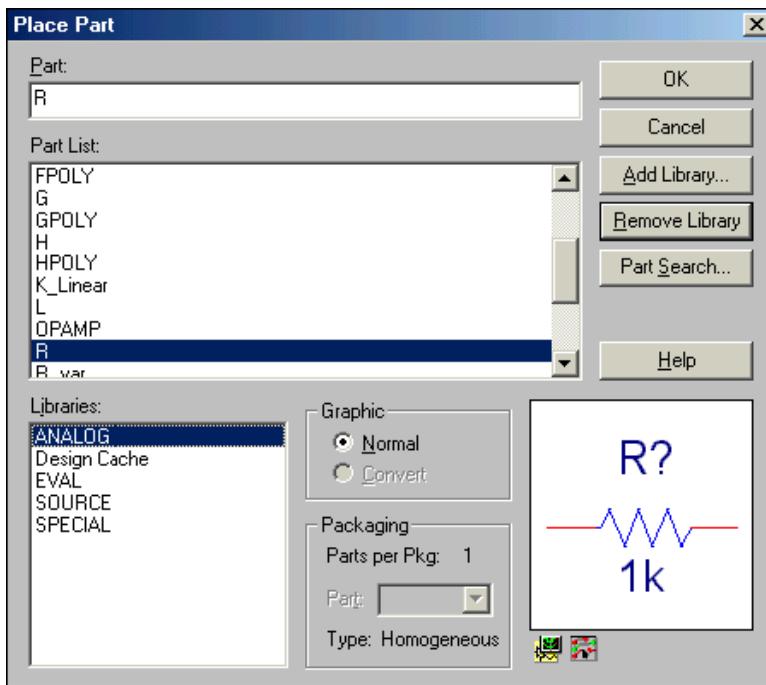


Figure 4: Place Part window

3. Select the library that contains the required components. Type the beginning of the name in the Part box.

The part list will scroll to the components whose name contains the same letters. If the library is not available, you need to add the library, by clicking on the **Add Library button**. This will bring up the Add Library window. Select the desired library. For Spice you should select the libraries from the Capture/Library/PSpice folder.

Analog: contains the passive components (R,L,C), mutual inductance, transmission line, and voltage and current dependent sources (voltage dependent voltage source E, current-dependent current source F, voltage-dependent current source G and current-dependent voltage source H).

Source: give the different type of independent voltage and current sources, such as Vdc, Idc, Vac, Iac, Vsin, Vexp, pulse, piecewise linear, etc. Browse the library to see what is available.

Eval: provides diodes (D...), bipolar transistors (Q...), MOS transistors, JFETs (J...), real opamp such as the u741, switches (SW_tClose, SW_tOpen), various digital gates and components.

Abm: contains a selection of interesting mathematical operators that can be applied to signals, such as multiplication (MULT), summation (SUM), Square Root (SWRT), Laplace (LAPLACE), arctan (ARCTAN), and many more.

Special: contains a variety of other components, such as PARAM, NODESET, etc.

Part	Name	Symbol	Part	Name	Symbol
------	------	--------	------	------	--------

Resistor	R		Sinusoidal Voltage Source	VSIN	
Capacitor	C		Sinusoidal Current Source	ISIN	
Inductor	L		Voltage Pulse	VPULSE	
Ground	AGND		Current Pulse	IPULSE	
DC Voltage Source	VDC		Voltage exponential source	VEXP	
DC Current Source	IDC		Current exponential source	IEXP	
AC Voltage Source	VAC		Voltage piecewise linear source	VPWL	
AC Current Source	VDC		Current piecewise linear source	IPWL	
VCVS Voltage-Controlled Voltage Source	E		Voltage Viewpoint	VIEWPOINT	
CCCS Current-Controlled Current Source	- F		Current Iprobe	IPROBE	
VCCS - Voltage-Controlled Current Source	- G				
CCVS - Current-Controlled Voltage Source	- H				

4. Place the resistors, capacitor (from the Analog library), and the DC voltage and current source.

- You can place the part by the left mouse click.
- You can rotate the components by clicking on the R key.
- To place another instance of the same part, click the left mouse button again.
- Hit the ESC key when done with a particular element.
- You can add initial conditions to the capacitor. Double-click on the part; this will open the Property window that looks like a spreadsheet. Under the column, labeled IC, enter the value of the initial condition, e.g. 2V. For our example we assume that IC was 0V (this is the default value).

5. After placing all parts, you need to place the Ground terminal by clicking on the GND icon (on the right side toolbar – see Fig. 3). When the Place Ground window opens, select GND/CAPSYM and give

it the name 0 (i.e. zero). Do not forget to change the name to 0, otherwise PSpice will give an error or

"Floating Node". The reason is that SPICE needs a ground terminal as the reference node that has the

node number or name 0 (zero).

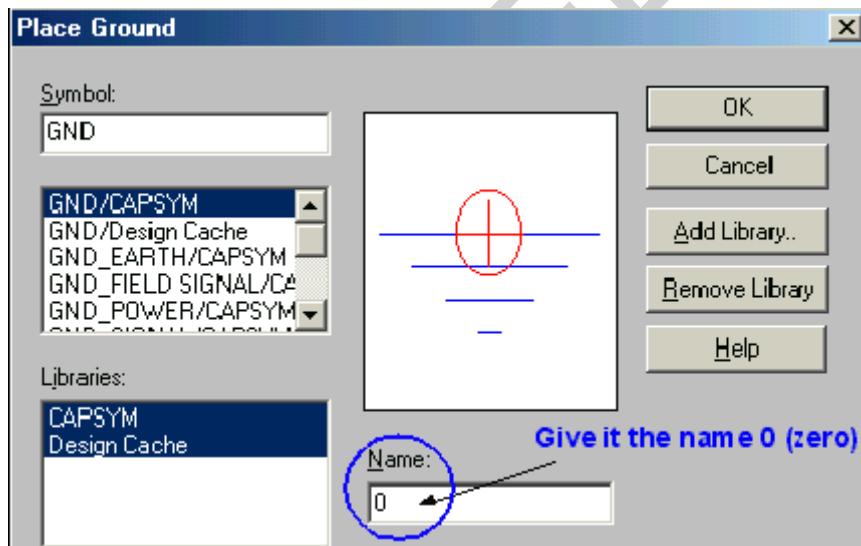


Figure 5: Place the ground terminal box; the ground terminal should have the name 0

6. Now connect the elements using the Place Wire command from the menu (PLACE/WIRE) or by clicking on the Place Wire icon.

7. You can assign names to nets or nodes using the Place Net Alias command (PLACE/NET ALIAS menu).

We will do this for the output node and input node. Name these Out and In, as shown in Figure 2.

2.1.3. Assign Values and Names to the parts

1. Change the values of the resistors by double-clicking on the number next to the resistor. You can also change the name of the resistor. Do the same for the capacitor and voltage and current source.
2. If you haven't done so yet, you can assign names to nodes (e.g. Out and In nodes).
3. Save the project

2.1.4. Netlist

The netlist gives the list of all elements using the simple format:

R_name node1 node2 value

C_name nodex nodey value, etc.

1. You can generate the netlist by going to the PSPICE/CREATE NETLIST menu.
2. Look at the netlist by double clicking on the Output/name.net file in the Project Manager Window
(in the left side File window).

Note on Current Directions in elements:

The positive current direction in an element such as a resistor is from node 1 to node 2. Node 1 is either the

left pin or the top pin for an horizontal or vertical positioned element (.e.g a resistor). By rotating the element

180 degrees one can switch the pin numbers. To verify the node numbers you can look at the netlist:

e.g. R_R2 node1 node2 10k

e.g. R_R2 0 OUT 10k

Since we are interested in the current direction from the OUT node to the ground, we need to rotate the

resistor R2 twice so that the node numbers are interchanged:

R_R2 OUT 0 10k

2.2 Step 2: Specifying the type of analysis and simulation

As mentioned in the introduction, Spice allows you do to a DC bias, DC Sweep, Transient with Fourier analysis, AC analysis, Montecarlo/worst case sweep, Parameter sweep and Temperature sweep. We will first explain how to do the Bias and DC Sweep on the circuit of Figure 2.

2.2.1 BIAS or DC analysis

1. With the schematic open, go to the PSPICE menu and choose NEW SIMULATION PROFILE.
2. In the Name text box, type a descriptive name, e.g. Bias
3. From the Inherit From List: select none and click Create.
4. When the Simulation Setting window opens, for the Analysis Type, choose Bias Point and click OK.
5. Now you are ready to run the simulation: PSPICE/RUN
6. A window will open, letting you know if the simulation was successful. If there are errors, **consult the Simulation Output file**.
7. To see the result of the DC bias point simulation, you can open the Simulation Output file or go back to the schematic and click on the V icon (Enable Bias Voltage Display) and I icon (current display) to show the voltage and currents (see Figure 6).

To check the direction of the current, you need to look at the netlist: the current is positive flowing from node1 to node1 (see note on Current Direction above).

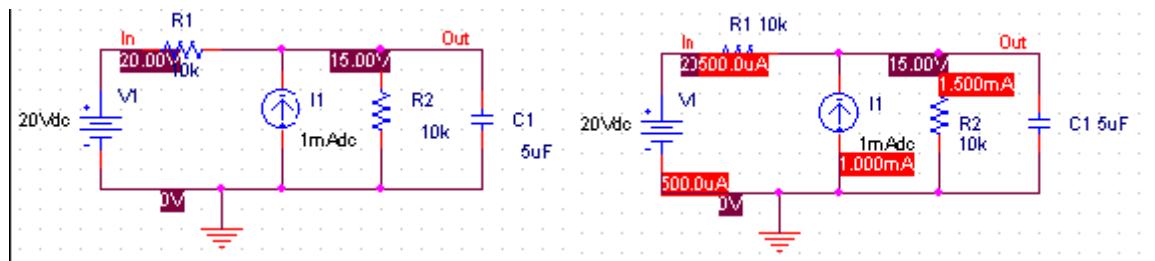
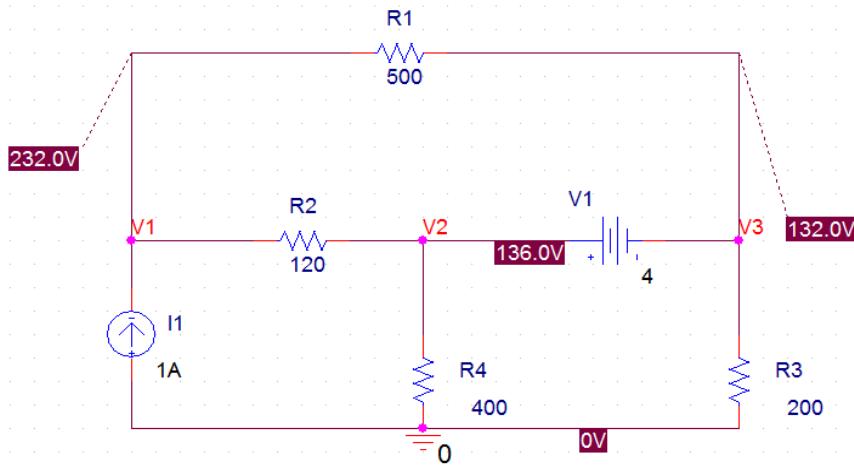


Figure 6: Results of the Bias simulation displayed on the schematic

PRACTICE:

Example 1:



Use the PSpice Schematics to construct the circuit below and obtain the node voltages.

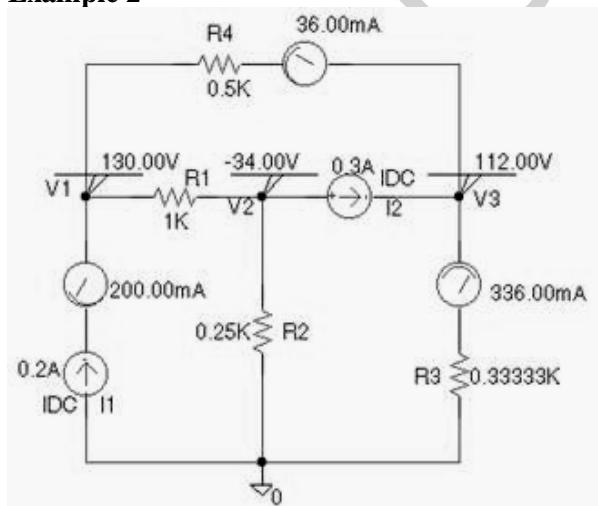
Schematics Netlist

**** INCLUDING "dc analysis-SCHEMATIC1.net" ****

* source DC ANALYSIS

```
R_R1      V1 V3 500
R_R2      V1 V2 120
R_R3      0 V3 200
R_R4      0 V2 400
V_V1      V2 V3 4
I_I1      0 V1 DC 1A
```

Example 2



Use the PSpice Schematics to construct the circuit below and obtain the node voltages and

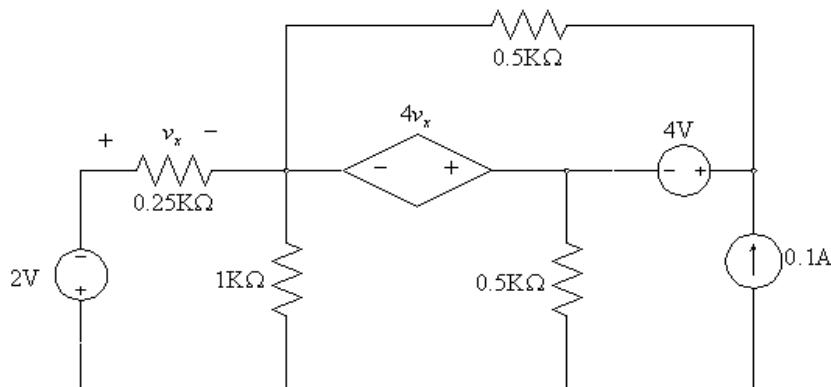
mesh currents. Assume a clockwise direction for the mesh current To obtain the node voltages a VIEWPOINT is placed at each node labeled V1, V2, and V3 (To assign a name or a number to a node double click on that node). To obtain the three mesh currents an IPROBE is placed in each mesh. Note that each IPROBE is rotated (Ctrl R) in such a way as to obtain the mesh current for the assumed clockwise direction.

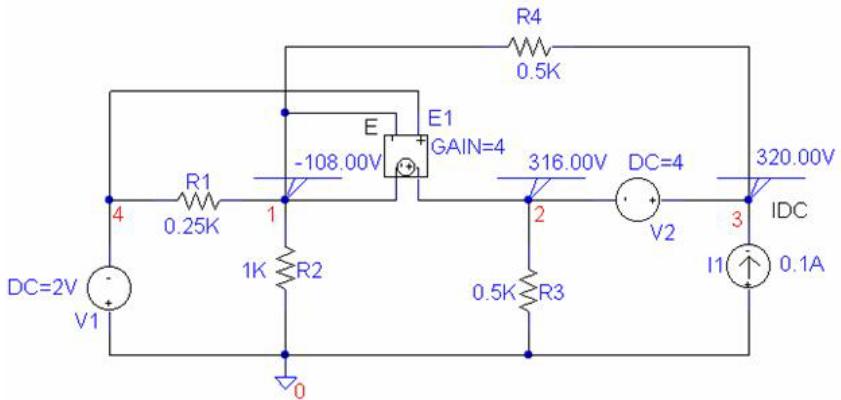
Schematics Netlist

R_R1	V1	V2	1K
R_R4	V1	\$N_0001	0.5K
R_R3	\$N_0002	0	0.33333K
R_R2	V2	0	0.25K
I_I1	0	\$N_0003	DC 0.2A
I_I2	V2	V3	DC 0.3A
v_V2	\$N_0001	V3	0
v_V3	V3	\$N_0002	0
v_V1	\$N_0003	V1	0

Example 3

Determine the node voltages for the circuit containing a Voltage-Controlled Voltage Source as shown.



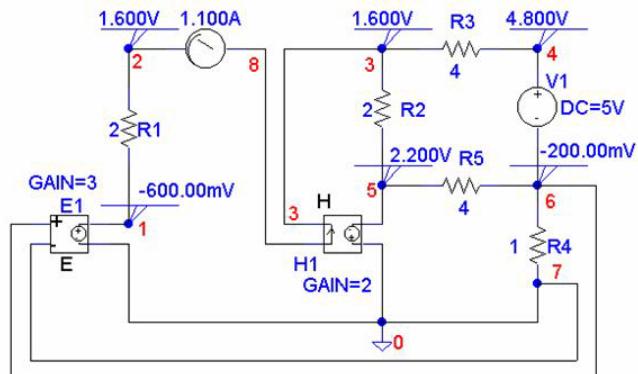
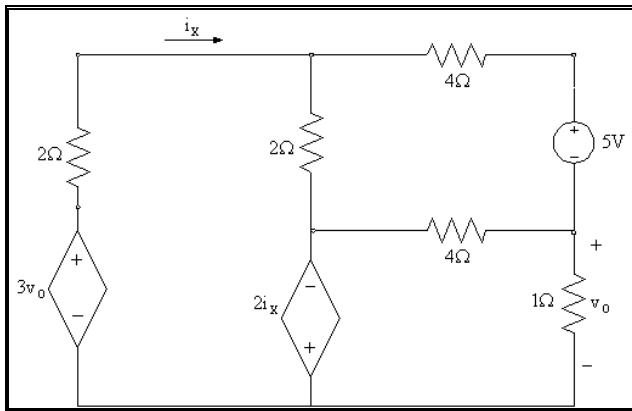


Schematics Netlist

R_R3	2	0	0.5K
R_R1	4	1	0.25K
V_V2	3	2	DC 4
E_E1	2	1	4 1 4
R_R2	1	0	1K
I_I1	0	3	DC 0.1A
V_V1	0	4	DC 2V
R_R4	1	3	0.5K

Example 4

The circuit below contains a Voltage-Controlled Voltage Source and a Current-Controlled Voltage Source. Determine the node voltages and the controlled current of the dependent source.

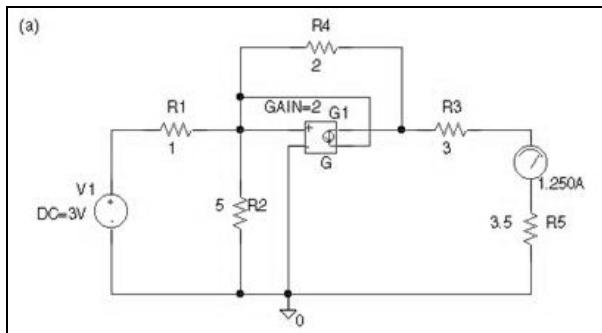


* Schematics Netlist *

R_R5	5	6	4
R_R4	6	0	1
R_R3	3	4	4
R_R2	3	5	2
R_R1	2	1	2
V_V1	4	6	DC 5V
E_E1	1	0	6 0 3
H_H1	0	5	VH_H1 2
VH_H1	8	3	0V
v_V2	2	8	0

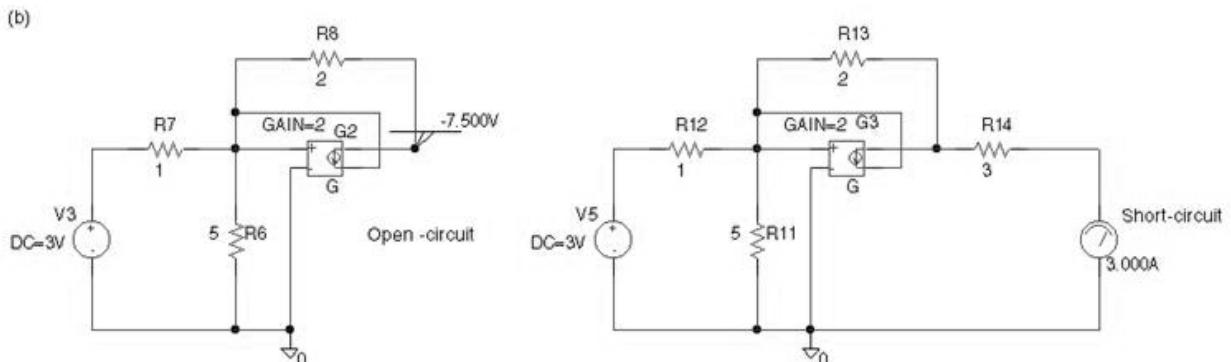
Example 5

- Find the current in the 3.5Ω resistor in the circuit (a).

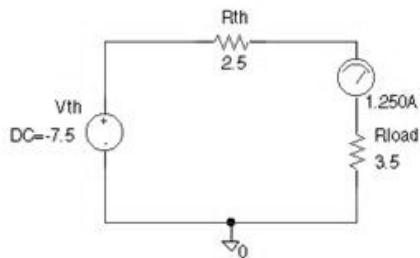


(b) Obtain the Thevenin's equivalent circuit with respect to the terminal of the $3.5\ \Omega$ resistor.

With the $3.5\ \Omega$ resistor removed, we find the open-circuit voltage, and with load resistance shorted we find the short-circuit current. Then, the Thevenin's impedance is



$$R_{th} = (-7.5)/(-3) = 2.5\ \text{Ohms}$$



Schematics Netlist

V_V1	\$N_0001	0	DC 3V
R_R2	\$N_0002	0	5
R_R1	\$N_0001	\$N_0002	1
G_G1	\$N_0003	\$N_0002	\$N_0002
R_R4	\$N_0002	\$N_0003	0
R_R3	\$N_0003	\$N_0004	2

V_V3	\$N_0005	0	DC	3V		
R_R6	\$N_0006	0		5		
R_R7	\$N_0005	\$N_0006	1			
R_R8	\$N_0006	\$N_0007	2			
G_G2	\$N_0007	\$N_0006		\$N_0006	0	2
R_R12	\$N_0009	\$N_0008	1			
R_R13	\$N_0008	\$N_0010	2			
V_V5	\$N_0009	0	DC	3V		
R_R11	\$N_0008	0		5		
R_R14	\$N_0010	\$N_0011	3			
G_G3	\$N_0010	\$N_0008		\$N_0008	0	2
R_Rth	\$N_0013	\$N_0012	2.5			
V_Vth	\$N_0013	0	DC	-7.5		
R_Rload	\$N_0014	0		3.5		
R_R5	\$N_0015	0		3.5		
v_V2	\$N_0004	\$N_0015	0			
v_V7	\$N_0011	0		0		
v_V9	\$N_0012	\$N_0014	0			

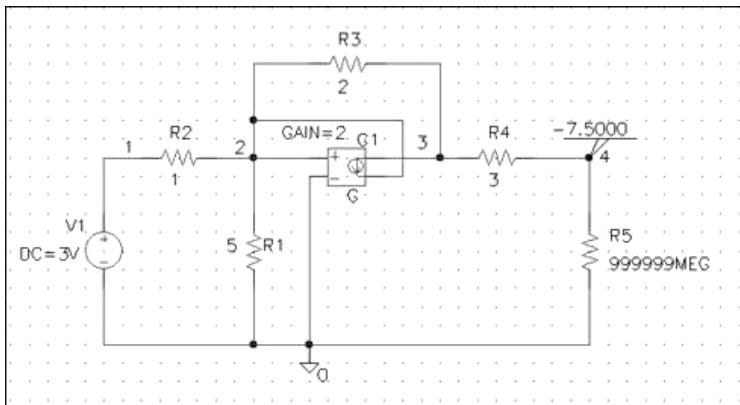
2. Transfer function

Computes the gain and the input and output resistances and automatically sends the results to the output file. To specify the transfer function, choose Setup from the Analysis menu and open the Transfer Function dialog box. In the Output Variable box specify the output voltage, and in the Input Source box type the input source name. The transfer function analysis can be used conveniently to obtain the Thevenin's equivalent circuit. The use of this analysis to obtain the Thevenin's equivalent circuit is demonstrated in the following example

Example 6

Use the transfer function analysis to obtain the Thevenin's equivalent circuit for the circuit of Example 5.

We want to obtain the Thevenin's equivalent circuit with respect to the $3.5\ \Omega$ resistor. The $3.5\ \Omega$ resistor is replaced by an open circuit. Since in PSpice two or more elements must be connected to each node, the $3.5\ \Omega$ resistor is replaced by an infinitely high value resistor as shown. A VIEWPOINT is added at node 4 to display the open-circuit voltage (Thevenin's voltage). To obtain the output or Thevenin's resistance, the transfer Function is enabled. In the Output variable box we type $V(4)$, and in the Input Source box, we type $V1$.



Schematics Netlist

V_V1	1 0 DC 3V
R_R1	2 0 5
R_R2	1 2 1
G_G1	3 2 2 0 2
R_R3	2 3 2
R_R4	3 4 3
R_R5	4 0 999999MEG

In addition to the DC analysis result, the output file contains the Transfer Function analysis results as follows

SMALL-SIGNAL CHARACTERISTICS

V(4,0)/V_V1	= -2.500E+00
INPUT RESISTANCE AT V_V1	= 6.000E+00
OUTPUT RESISTANCE AT V(4,0)	= 2.500E+00

Thus, the Thevenin's voltage is $V_4 = -7.5$ V, and the Thevenin's resistance is the output resistance 2.5 W.

Exercice:

As instructed by the lab teacher.

2.1 DC Sweep simulation

We will be using the same circuit but will evaluate the effect of sweeping the voltage source between 0 and 20V. We'll keep the current source constant at 1mA.

1. Create a new New Simulation Profile (from the PSpice Menu); We'll call it DC Sweep
2. For analysis select DC Sweep; enter the name of the voltage source to be swept: V1. The start and end values and the step need to be specified: 0, 20 and 0.1V, respectively (see Fig. below).

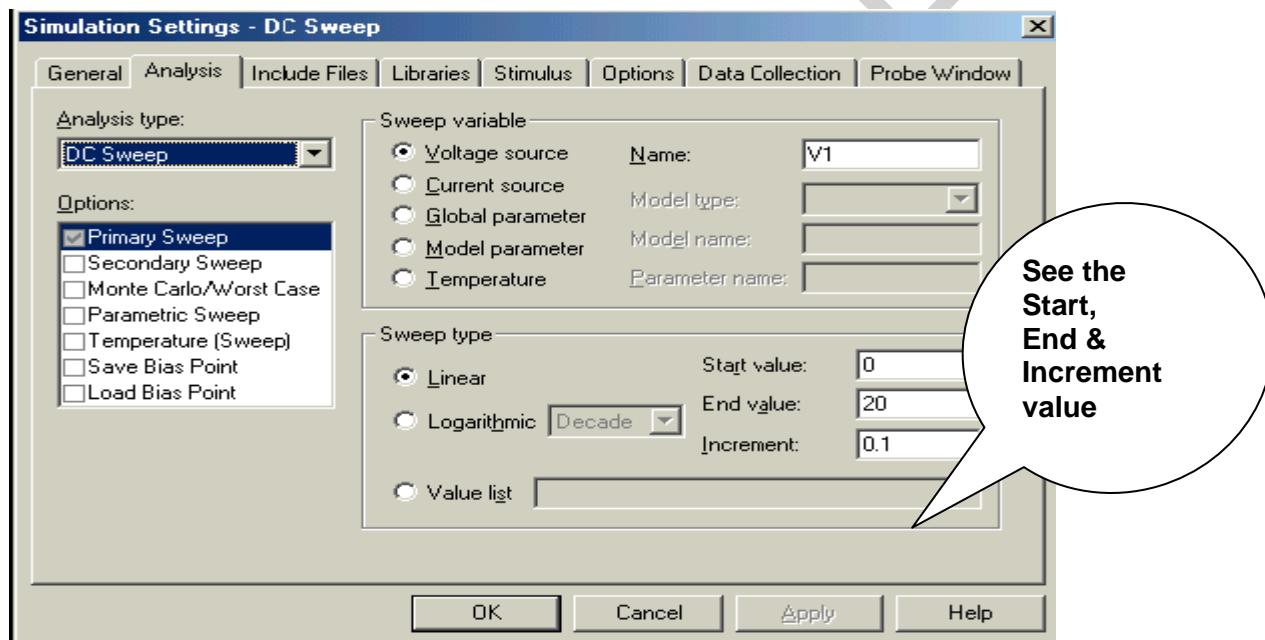


Figure 7: Setting for the DC Sweep simulation.

3. Run the simulation. PSpice will generate an output file that contains the values of all voltages and currents in the circuit.

2.3 Step 3: Displaying the simulation Results

PSpice has a user-friendly interface to show the results of the simulations. Once the simulation is finished a Probe window will open.

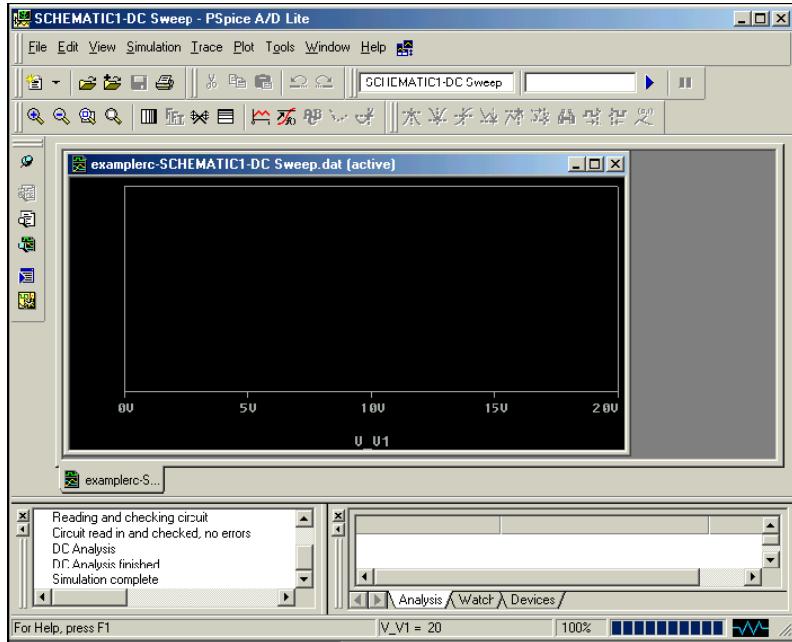


Figure 8: Probe window

From the TRACE menu select ADD TRACE and select the voltages and current you like to display. In our case we'll add V(out) and V(in). Click OK.

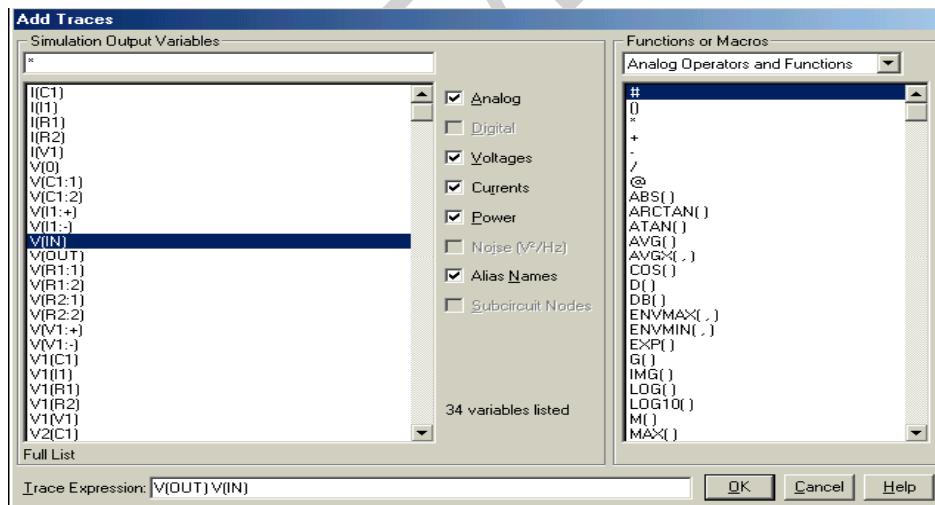


Figure 9: Add Traces window

You can also add traces using the "Voltage Markers" in the schematic. From the PSPICE menu select MARKERS/VOLTAGE LEVELS. Place the makers on the Out and In node. To place a node, first place your mouse on the PLACE NET ALIAS on the right corner of screen. Then write **In** and press OK. Do the same procedure for **Out** node. When done, right click and select End Mode.

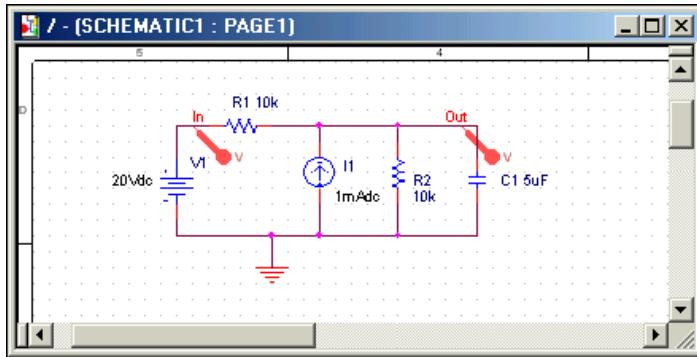


Figure 10: Using Voltage Markers to show the simulation result of $V_{(out)}$ and $V_{(in)}$

Go to back to PSpice. You will notice that the waveforms will appear.

You can add a second Y Axis and use this to display e.g. the current in Resistor R2, as shown below. Go to PLOT/Add Y Axis. Next, add the trace for $I(R2)$. You can also use the cursors on the graphs for V_{out} and V_{in} to display the actual values at certain points. Go to TRACE/CURSORS/DISPLAY The cursors will be associated with the first trace, as indicated by the small rectangle around the legend for $V_{(out)}$ at the bottom of the window. Left click on the first trace. The value of the x and y axes are displayed in the Probe window. When you right click on $V_{(out)}$ the value of the second cursor will be given together with the difference between the first and second cursor.

To place the second cursor on the second trace (for $V_{(in)}$), right click the legend for $V_{(in)}$.

You'll notice the outline around $V_{(in)}$ at the bottom of the window. When you right click the second trace the cursor will snap to it. The values of the first and second cursor will be shown in Probe window. You can chance the X and Y axes by double clicking on them. When adding traces you can perform mathematical calculations on the traces, as indicated in the Add Trace Window to the right of Figure 9.

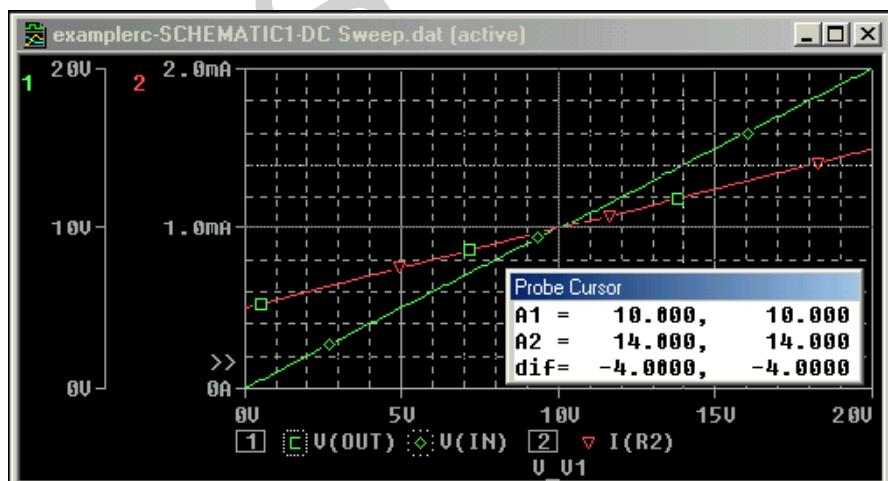


Figure 11: Result of the DC sweep, showing V_{out} , V_{in} and the current through resistor R_2 Cursors were used for $V_{(out)}$ and $V_{(in)}$.

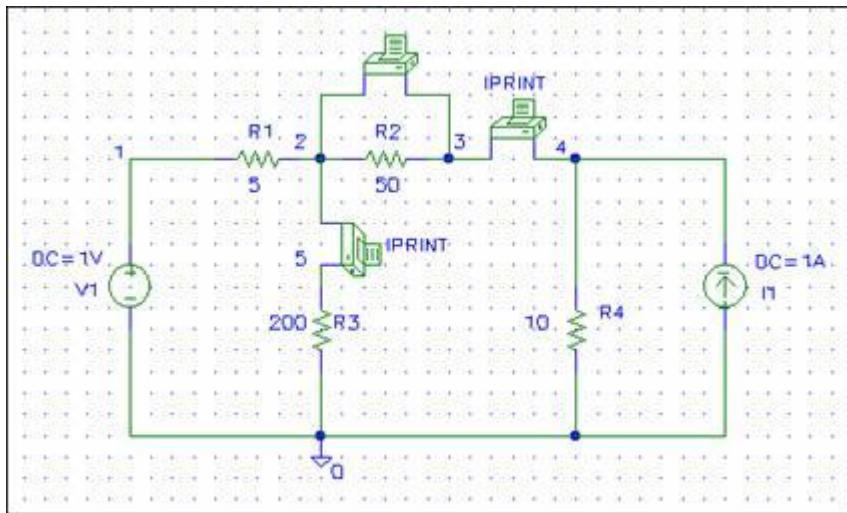
DC Sweep

DC voltage or current sources may be swept over a range through the use of the **DC Sweep Analysis**. The DC Sweep is similar to the node voltage analysis, but adds more flexibility. In the DC sweep, the input variable is varied over a range of values. For each value of the input variable, the DC operating point is

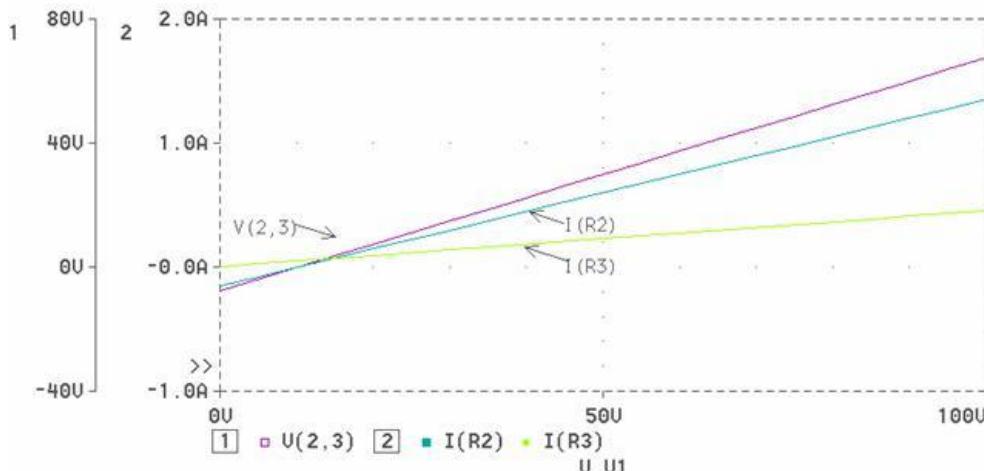
computed. If the transfer function is also enabled the small-signal DC gain and the input/output resistances are also computed.

Example 7

For the circuit shown print and plot the voltage V₂₃, I_{R2}, and I_{R3} as the input voltage is varied from 0 to 100 V in 5 V increment. Select the Setup from the analysis menu, click the DC Sweep dialog button. The DC Sweep dialog box appears. For the Sweep Variable type select the voltage source, and set its name to V1. Using the Sweep type linear, specify the starting value, end value and increment. IPRINT and VPRINT2 are inserted in the appropriate locations to send the desired currents and voltages to the output file. To include the DC values double click on each print icon and set the DC = to yes.



The Probe result which includes traces V(2,3) I(R2), and I(R3) is shown below. The desired currents and voltage copied from the output file are tabulated below.



V_V1 V(2,3)

0.000E+00 -7.707E+00

.....
9.500E+01 6.372E+01
1.000E+02 6.748E+01

V_V1 I(V_PRINT3)

0.000E+00	-1.541E-01
5.000E+00	-7.895E-02
1.000E+01	-3.759E-03
1.500E+01	7.143E-02
.....	
9.500E+01	1.274E+00
1.000E+02	1.350E+00

V_V1 I(V_PRINT1)

0.000E+00	3.759E-03
.....	
9.500E+01	4.323E-01
1.000E+02	4.549E-01

MAXIMUM POWER TRANSFER

Maximum power is transferred from a source to an equal-value load and is proved by differentiating the output power with respect to the load resistance and equating to zero. The potential divider in Fig. 2.1 demonstrates maximum power transfer from the source, V source, and source resistance, R source, to a varying load RL .

Param Part

In this example, select the RL default value of 1 k and replace it by a name of your choice e.g. {Rvar} in the Name box of the Display Parameter menu. *The variable resistance name must be enclosed in curly brackets{ }.* However, the Rvar name has no significance and can be any name you choose. The PARAM part, from the SPECIAL.OLB library, is very useful for investigating circuit behavior for a range of part values. This part, however, is used and accessed in a different manner to that in previous versions of PSpice. What is required is to place the part using the little AND gate symbol, and when selected it turns green. Rclick and select Edit Properties to display the spreadsheet shown in Fig. 2.2. In this spreadsheet, we may add new rows and assign values in the A column. There is now no limit to the number of parameters you may add in version 10.5, unlike the PARAM part in version 8, which was limited to three parameters.

Select New Row and enter Rvar in the Name: box but NO curly brackets, and the nominal value of 300 ohm in the Value box as shown in Fig. 2.3.

Simulation Settings

Press Edit Simulation Settings icon or select the PSpice/New Simulation ProPle menu. Select DC Sweep and enter the parameters shown in Fig. 2.4. Rvar is varied Linearly from 1 to 300 ohm in Increments of 1 ohm

An alternative to the above procedure is to specify a list of values separated by spaces in the Value list box.

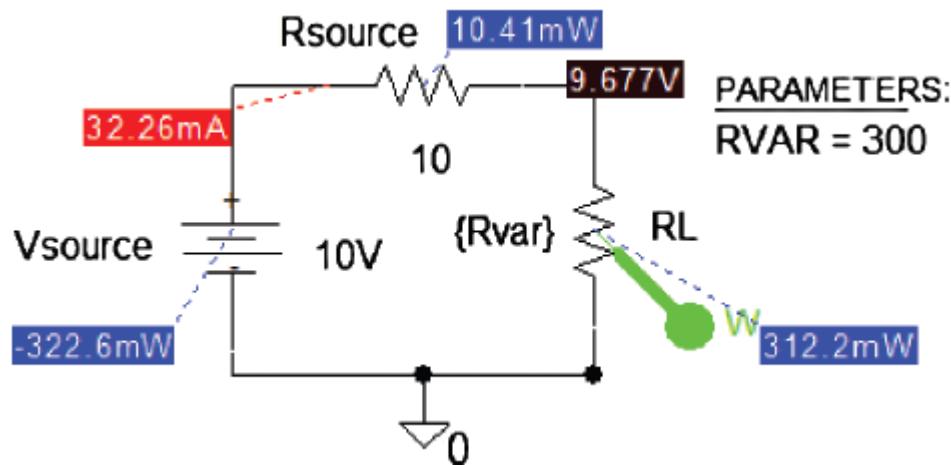


Figure 2.1: Potential divider with variable loads

	New Row...	Apply	Display...	Delete Prop
		A		
	+ FIGURE1-012 : PAGE1 :			
PSpiceOnly		TRUE		
Reference		1		
Value		PARAM		
Rvar		300		
Location X-Coordinate		560		
Location Y-Coordinate		60		
Source Part		PARAM.Normal		

FIGURE 2.2: Edit Properties Param parameters

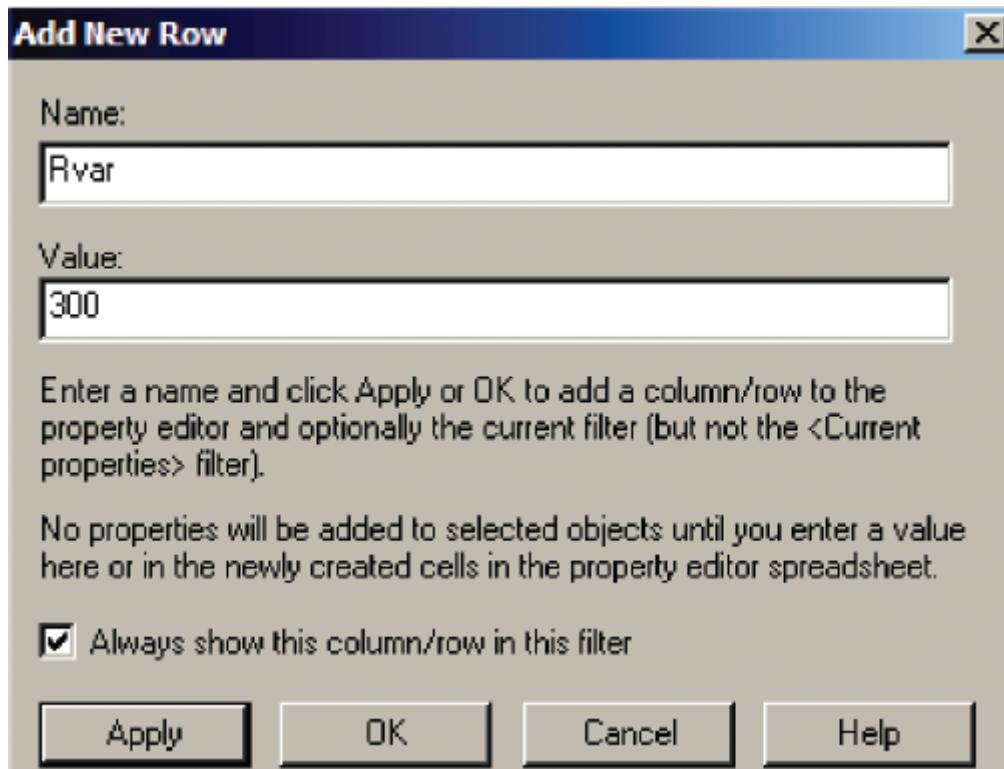


FIGURE 2.3: The Param part parameters

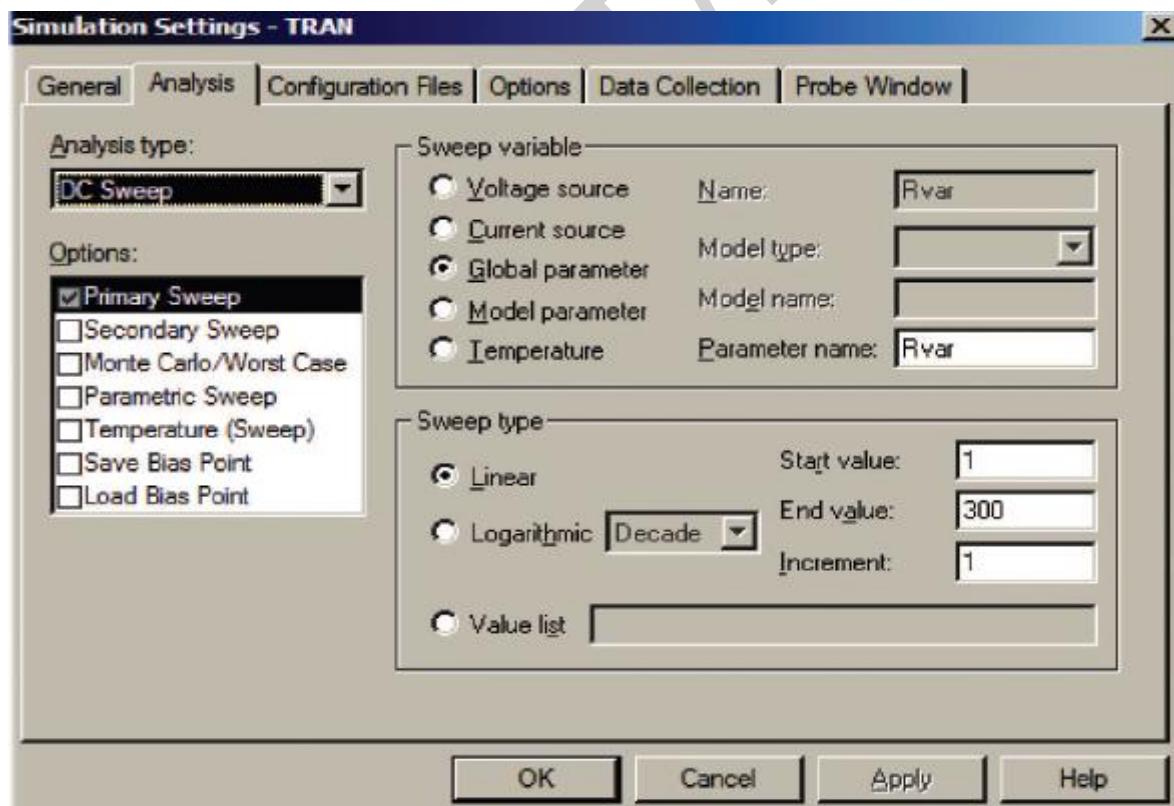


FIGURE 2.4: Analysis and DC sweep

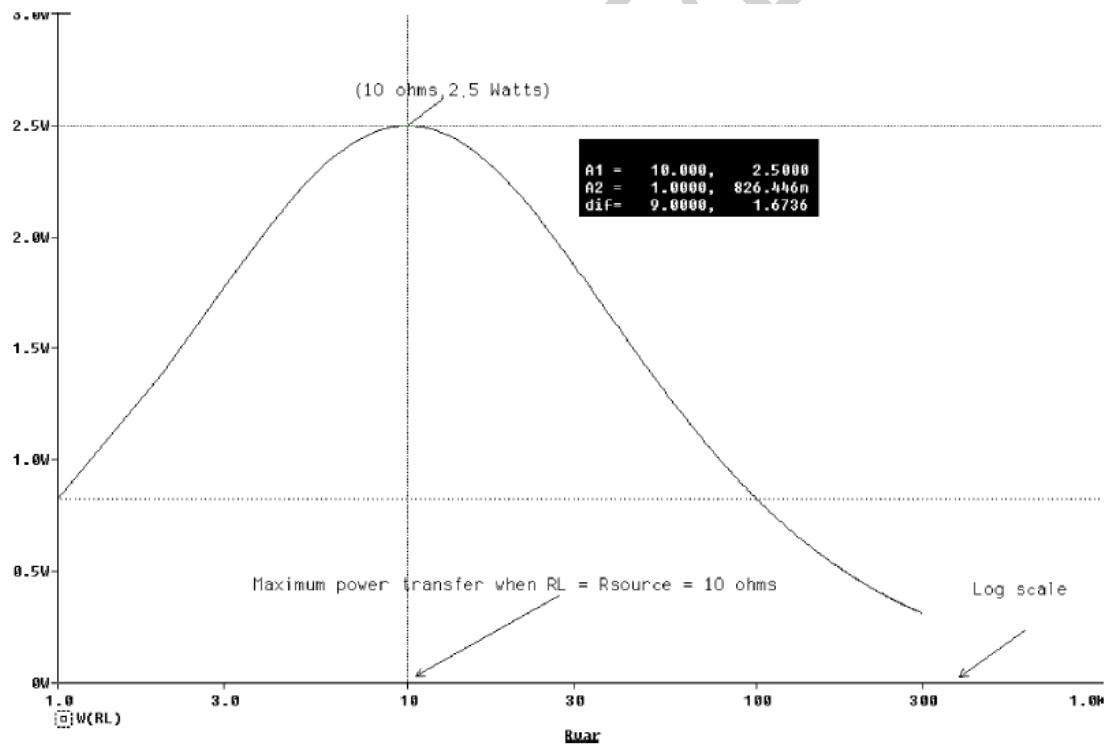
Trace Expression Box

Press F11 and a blank Probe screen appears since no markers were placed on the schematic. To display power dissipated in a device, we may use one of the two techniques. The first technique uses the power marker (W icon) that is located in the middle of the device as shown. The second technique is the manual method accessed from the Probe screen after simulation when you select the Trace add icon (Or press the Insert button on the keyboard). Enter the expression for the load power as $I(R2)*V1(R2)$ in the Trace Expression box shown in Fig. 2.5.



FIGURE 2.5: Adding variables

Note: The resistance orientation has an effect on the final Probe display. If the display comes out upside down, then the resistor is the wrong way around and should be rotated around 180° (the R key), or place a minus sign in front of the current*voltage product. Component orientation is how PSpice handles conventional current direction flow in R , L , and C , so be careful about the way you place them and marker placement. Fig. 2.6 shows the Probe plot when cursors are placed using the left and right mouse buttons. Maximum power is measured



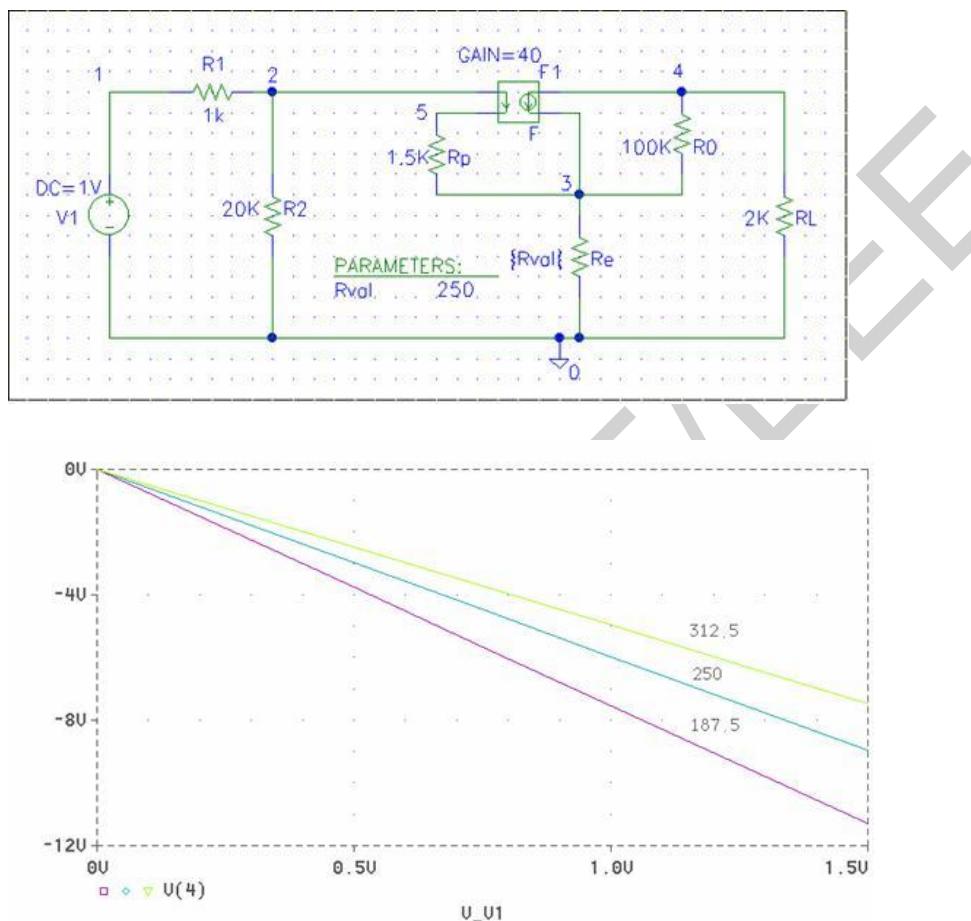
by selecting the maximum Cursor Peak icons to locate the cursor at the maximum power value. Compare the measured maximum power to the value in (2.1).

$$P_{\max} = I^2 R_L = \left[\frac{V_{\text{source}}}{R_{\text{source}} + R_L} \right]^2 R_L = \left[\frac{V_{\text{source}}}{R_L + R_L} \right]^2 R_L = \frac{V_{\text{source}}^2}{4R_L} = \frac{100}{4 \times 10} = 2.5 \text{ W}$$

From Probe, select Plot /Label to access a menu bar containing many useful display graphic tools such as Text, line, Arrows, Box. The plot is copied to clipboard from the Probe/Windows where you have the option of copying with a white or black background.

Example 8

The circuit shown below represents an amplifier circuit modeled by a current-controlled current source. The input voltage is varied from 0 to 1.5 V with an increment of 0.25 V. The resistance R_E changes by $\pm 25\%$. Plot the DC transfer characteristic output voltage V_4 versus the input voltage V_1 for each value of the resistance R_E .



Select the Setup from the analysis menu, click the DC Sweep dialog button. The DC Sweep dialog box appears. For the Sweep Variable type select the voltage source, and set its name to V_1 . Using the Sweep type linear, specify the starting value, end value and increment (0, 0.25, and 1.5).

The Parametric Sweep can be used to change the value of the resistor R_E . To do this, get a part called PARAM from Special.slb and place it in your circuit. Double click on the text PARAMETERS. The attributes allow you to define up to three different variable parameters. For **NAME1**= define a parameter, say R_{val} , and for **VALUE1**= set an initial value of 250. Click the OK button to accept the attribute. We must now change the value of R_E to the name of the parameter. Double click on the value of R_1 and in place of its value type in the text $\{Rval\}$. Next select the **Analysis** and then **Setup** and click the **Parametric** button. Mark **Global Parameter**, set **Name** to R_{val} . Under **Sweep Type**, mark **Value List** and set the Values: to 187.5 250 312.5. Perform the simulation.

LESSON 3: TRANSIENT ANALYSIS

(Lab-3)

Time Domain (Transient) Analysis

The Time Domain, or Transient Analysis has the purpose of predicting how a circuit will respond to a transient event, like a step input. So this would be the method for doing a step response analysis. For the purposes of exploring this method we will build a new circuit, a simple RC network. This is a convenient way to analyze the effects of a single component on the overall behavior of a circuit.

The start with, build the circuit that is illustrated in **Figure 1**. To do this start a new project called RC_Circuit. In so doing, put this project in the subfolder of the temp your desired directory and call it RC_Circuit also. A new factor here is the placing of a capacitor; yet, it works just like the process of placing a resistor. Another new factor is the use of the VPULSE voltage source. How do you place such a part?

- **GO Place→Part** and select the SOURCE library in the dialogue box. Then type in VPULSE as the type of source desired. This will produce a VPULSE symbol, which you can place like other components. Then you can double-click on each parameter listed with this source and fill in the values shown in **Figure: 1 Here it is!** ☺

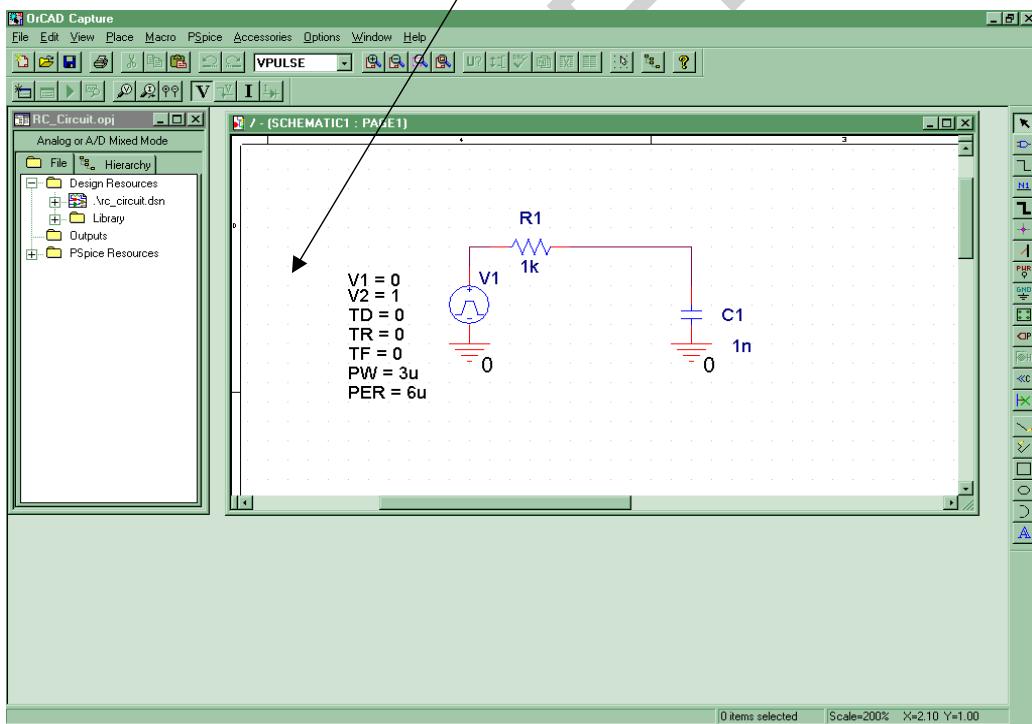


Fig:1

Let's take a moment to look at the parameters shown on the VPULSE source.

V1 = 0	For the pulse, the beginning voltage level
V2 = 1	For the pulse, the ending voltage level
TD = 0	The delay time of the pulse, from the simulation starting point (0 seconds)
TR = 0	The risetime
TF = 0	The falltime
PW = 3u	The pulse width
PER = 6u	The period of the pulse

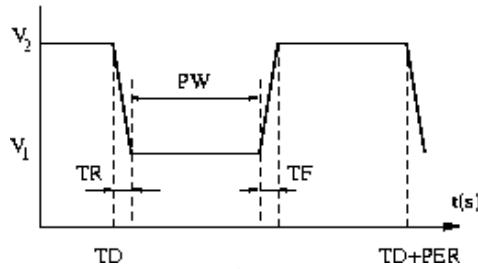


Fig:2

Now go **PSpice** → **New Simulation Profile**. This brings up the Simulation Settings dialogue box. Choose **Time Domain** analysis and set the **Run to time: 6000ns**. Before running the simulation, put a voltage marker on the non-grounded end of the capacitor C1. Set the Data Collection tab setting to **At Markers only**. Now run the analysis and see what happens. The result of this analysis is shown in **Figure 3**. This Probe display is also documented with text and arrows. **Here it is!** ☺

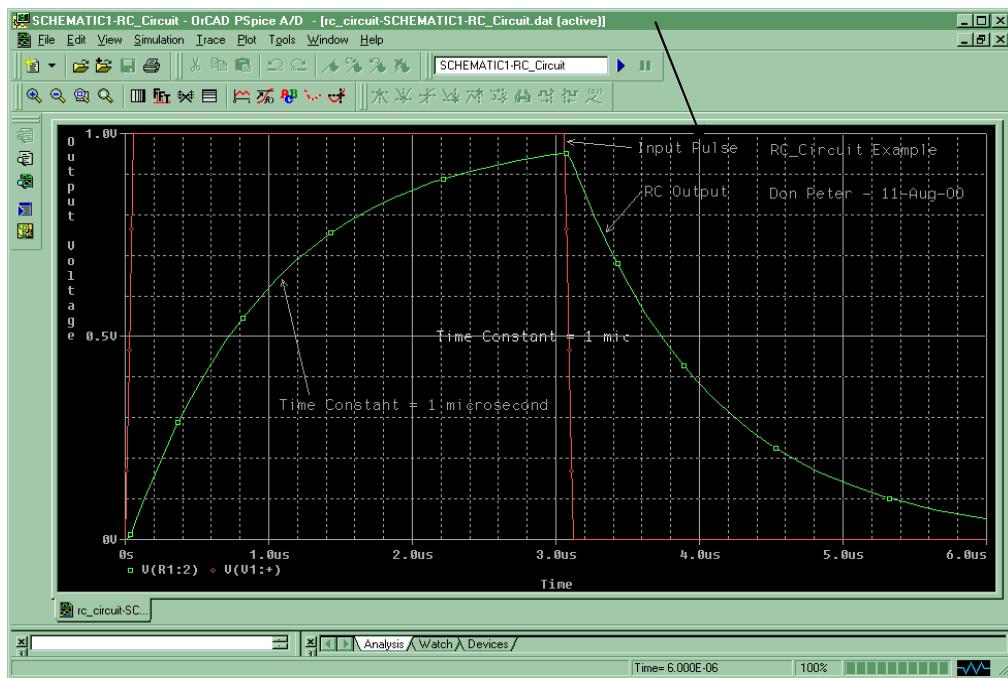


Fig:3

- For a sine wave, use **VSIN** for your voltage source instead of **VAC** (**VOFF** is the DC offset, **VAMPL** is the amplitude, and **FREQ** is the frequency of the sine wave).

a. **Square Wave** is the **VPLUSE** function in the limit of $TR = TF = 0$ and $PW = 0.5 * PER$ (PER is the period of the wave). This limit case, however, causes numerical difficulties in calculations. In any case, we can never make such a square function in practice. In reality, square waves have very small TR and TF . Typically, we use a symmetric function, i.e., we set $TR = TF$ and $PW = 0.5 * PER - 2 * TR$. Thus, for a given frequency we can set up the square function if we choose TR . If we choose TR too large, the function does not look like a square wave. If we choose TR too small, the program will take a long time to simulate the circuit and for TR smaller than a certain value, the simulation will not converge numerically. A good choice for TR is to set it to be 1% of the PER (a period): $TR = TF = 0.01 * PER$, $PW = 0.48 * PER$. This usually results in a nice signal without a huge amount of computational need. Note that TR does not have to be exactly 1% of PER . You can choose nice round numbers for TR , TF , and PW .

b. **Triangular Wave** is the **VPLUSE** function in the limit of $TR = TF = 0.5 * PER$ and $PW = 0$

(convince yourself that this is the case). As before, the limit case of PW = 0 causes numerical difficulties in calculations. So we have to choose PW to be a reasonably small value. A good choice for PW is to be set at 1% of the PER (period): PW = 0.01 * PER, TR = TF = 0.49 * PER (and not TR = TF = 0.495 * PER so that we get a symmetric function). This usually results in a nice signal without a huge amount of computational need. Again, note that PW does not have to be exactly 1% of PER. You can choose nice round numbers for TR, TF, and PW.

Simulation settings:

- Analysis Type: Time Domain (Transient)
- Options: General Settings
- Enter a Run to time so that a few periods will be displayed. Remember that the period (seconds) = 1/frequency (Hz), i.e., if you are using a 1kHz sine wave, it has a 1/1kHz=1ms period, so use a Run to time of 5ms for 5 periods
- Set the Maximum step size to be much smaller than the period.** i.e., for a 1kHz sine wave: It has a 1ms period, so set a maximum step size of approx .01ms. (This works out to 100 data points per period). If you don't set the maximum step size, PSpice may choose one which is too big, making your sine wave look angular and ugly.

Press OK and simulate. The simulation window should now include a place for you to plot your data. See Section A.

Section A: GRAPHING IN PSPICE

1 General Instructions

On the simulation window,

- 1) Go to Trace => Add Trace
- 2) Select the variable you want to plot on the y-axis. Or type in an expression on the Trace Expression prompt at the bottom of the window. Press OK
- 3) To mark points:

a. Click the "Toggle Cursor" button.  (Or go through the menu, Trace => Cursor => Display.) You will now be able to move the cursor along your plot.

b. Click the "Mark Label" button  to label that point. (Or go through the menu, Plot => Label => Mark.)

2 How to Change the x-Axis Variable

You can change the axis variable on your plot from, for example, resistance to voltage like in Lab 1 without adjusting the simulation settings. Here is how:

- 1) Double click the x-axis.
- 2) Select the x-axis tab.
- 3) Click Axis Variable...
- 4) Select new variable and press OK.

Transient simulation with a sinusoidal input.

SWITCH_OPEN (OPEN after given time) :

```
X_U1 start_node end_node Sw_tOpen PARAMS: tOpen=0 ttran=1u Rclosed=0.1m
+ Ropen=100G
```

Where:

tOpen= Time after which it will open

ttran= Transient time while the switch is opening

Rclosed= The resistance of switch when it is closed; generally 0.1m

Ropen= The resistance of switch when it is open; generally 100G

Subckt model of these switches:

```
.SUBCKT Sw_tClose 1 2 PARAMS:
+ tClose=0      ; time at which switch begins to close
+ ttran=1u       ; time required to switch states (must be realistic, not 0)
```

```

+ Rclosed=0.01 ; Closed state resistance
+ Ropen=1Meg ; Open state resistance (Ropen/Rclosed < 1E10)
V1 3 0 pulse(0 1 {tClose} {ttran} 1 10k 11k)
S1 1 2 3 0 Smod
.model Smod Vswitch(Ron={Rclosed} Roff={Ropen})
.ends

*End of Sw_tClose model

```

```

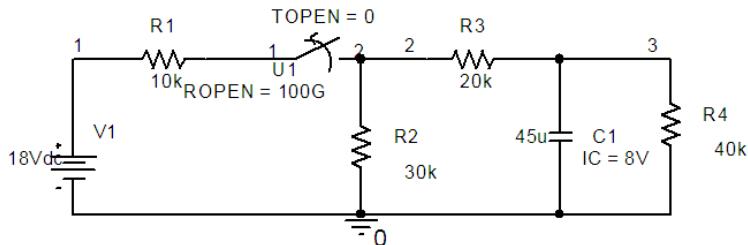
* For DC and AC analyses, the switch will be Closed.
.SUBCKT Sw_tOpen 1 2 PARAMS:
+ tOpen=0 ; time at which switch begins to open
+ ttran=1u ; time required to switch states (must be realistic, not 0)
+ Rclosed=0.01 ; Closed state resistance
+ Ropen=1Meg ; Open state resistance (Ropen/Rclosed < 1E10)
V1 3 0 pulse(1 0 {tOpen} {ttran} 1 10k 11k)
S1 1 2 3 0 Smod
.model Smod Vswitch(Ron={Rclosed} Roff={Ropen})
.ends

```

Example 1

The switch in the circuit below has been closed for a long time. Switch is opened at time $t = 0$. Use PSPICE and PROBE to obtain the natural response $v_c(t)$ and $i_c(t)$. From $v_c(t)$ obtain the circuit time constant. By inspection the initial voltage across the capacitor is given by $v(0) = 8$ volts, and the time constant of the circuit $\tau = RC = 1$ sec. For a complete display of the wavef

orm choose the final time of the transient analysis to be five time constants.



Schematics Netlist

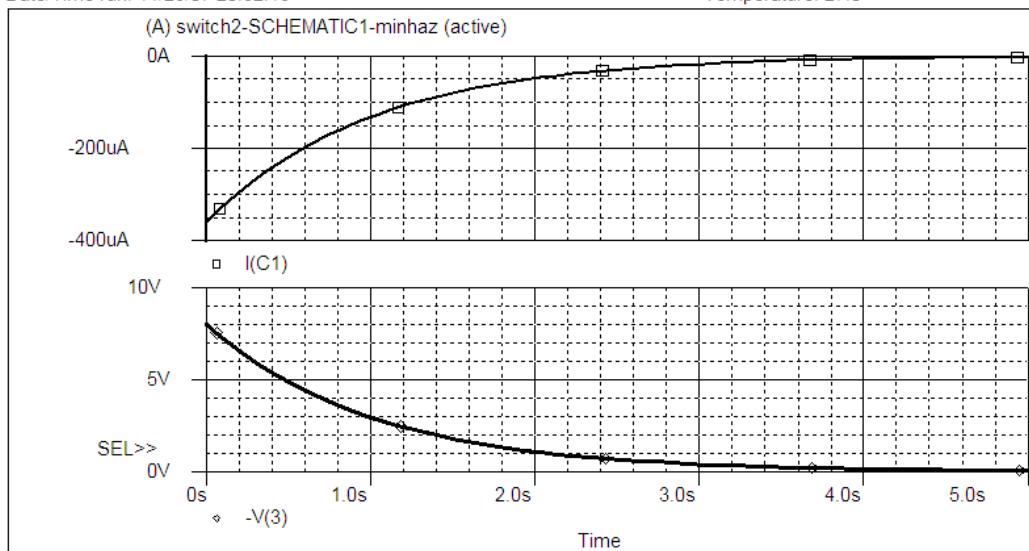
```

R_R1      1 N00103 10k
R_R2      0 2 30k
R_R3      2 3 20k
R_R4      0 3 40k
V_V1      1 0 18Vdc
C_C1      0 3 45u IC=8V
X_U1      N00103 2 Sw_tOpen PARAMS: tOpen=0 ttran=1u Rclosed=0.01 Ropen=1Meg

```

In PROBE, select **plot control** and **add plot** to create two graphs on the screen. Use select graph, up and down key to obtain a plot of voltage V(3) and current I(C1).

** circuit file for profile: minhaz
 Date/Time run: 11/23/07 20:32:13 Temperature: 27.0

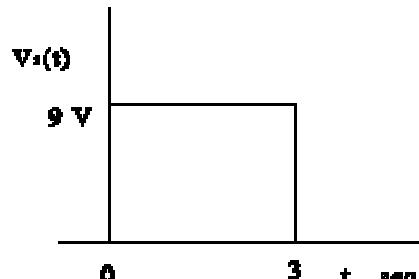
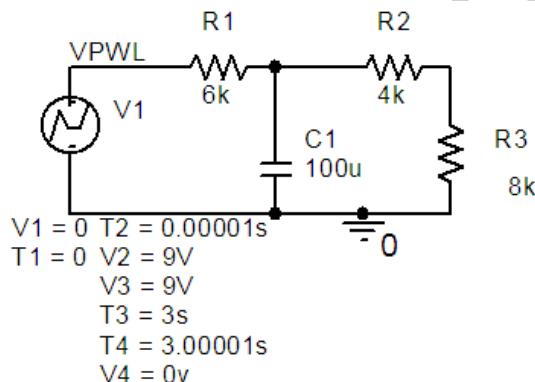


Example 2

Consider the network below in which the voltage input is a square pulse as shown. Use the VPWL function to generate this square pulse. Use PSPICE and PROBE to obtain the transient response of the capacitor voltage and capacitor current separately on one page. In the Transient Analysis use a final value of 6 second. On the voltage graph add a trace at 63.2% of capacitor final voltage and use the cursor command to locate the circuit time constant.

VPWL= voltage for piece wise linear

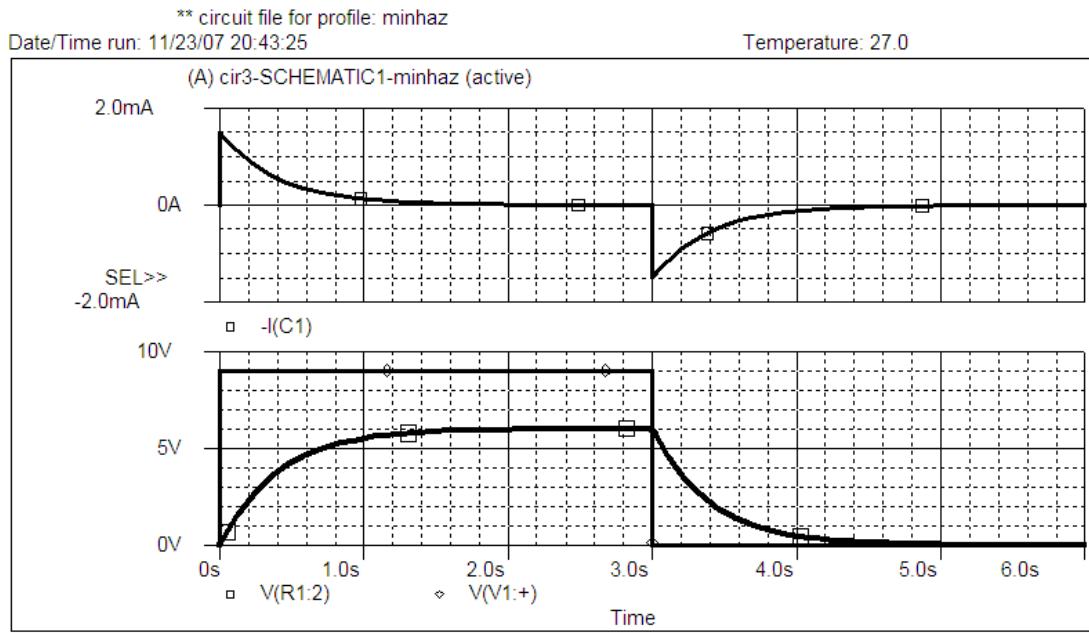
V_name node1 node2 PWL time0 voltage0 time1 voltage1 time2 voltage2



```

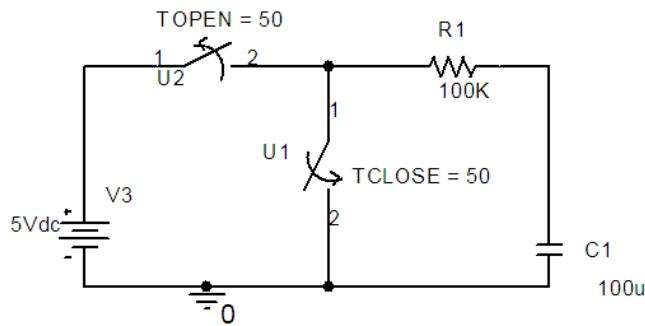
V_V1      1  0
+PWL 0 0 0.000001 9 3 9 3.000001 0
R_R1      1  2  6K
R_R2      2  3  4K
C_C1      2  0  100U IC=0
R_R3      3  0  8K
.TRAN 0 6s 0
.PROBE

```



Example 3

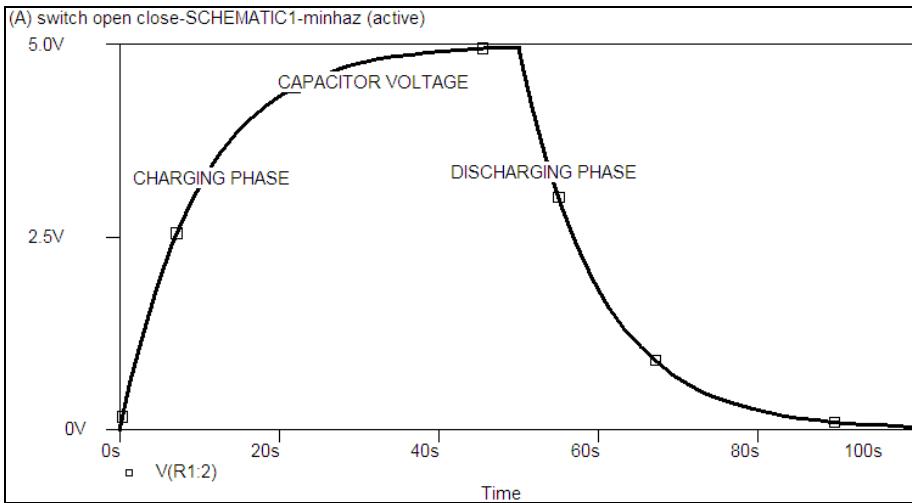
In the circuit shown the uncharged capacitor is charged for a duration of 50 seconds, it is then discharged immediately. Use Spice and Probe to obtain the capacitor voltage during charging and discharging process.



We can use a **SW_tOpen** and a **SW_tClose** switch to simulate the charging and discharging process. Since the switch is opened after 50 seconds we set tOpen = 50, and tClose = 50. The resistance Rclosed is set to a very low value (0.01m), and the resistance Ropen to a very high value 1000G.

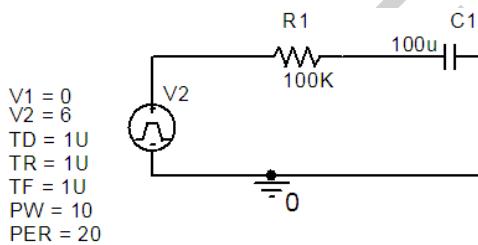
Schematics Netlist

```
C_C1      0 N00187 100u
R_R1      N00322 N00187 100K
V_V3      N00015 0 5Vdc
X_U1      N00322 0 Sw_tClose PARAMS: tClose=50 ttran=1u Rclosed=0.01
+   Ropen=1Meg
X_U2      N00015 N00322 Sw_tOpen PARAMS: tOpen=50 ttran=1u Rclosed=.1m
+   Ropen=1Meg
.TRAN    0 100S 0 SKIPBP
.PROBE
```



Example 4

The capacitor in the RC circuit of Example 4 is being charged for 10 seconds and then discharged for the next 10 seconds repeatedly. Use Spice and Probe to obtain the capacitor voltage for this cyclic charging and discharging process up to 100 seconds.



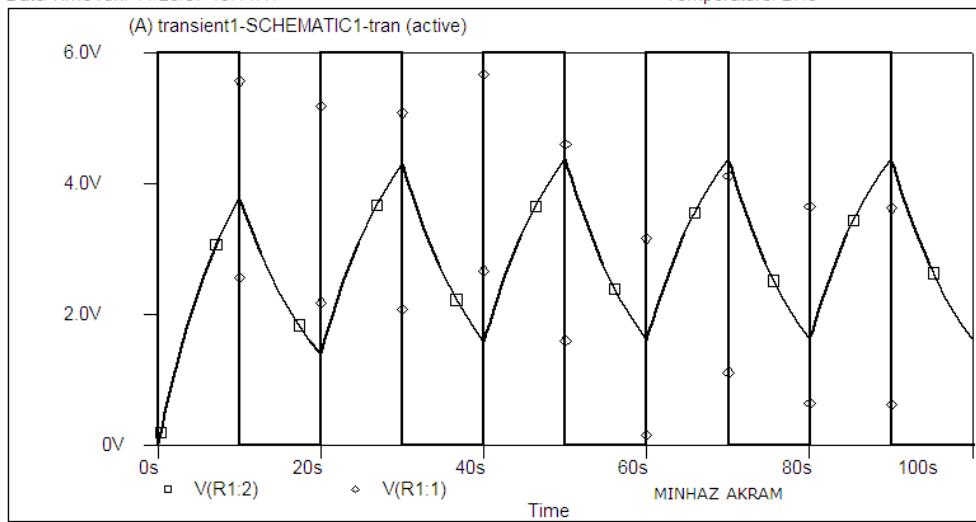
We can use a **VPULSE** source to simulate the on off switching action. The pulse has an initial value of 0 V, a final value of 5 V, a pulse width of 10 seconds and a period of 20 seconds.

Schematics Netlist

```
C_C1      N00187 0 100u
R_R1      N00013 N00187 100K
V_V2      N00013 0
+PULSE 0 6 1U 1U 1U 10 20
.TRAN 0 100S 0 SKIPBP
.PROBE
```

Date/Time run: 11/23/07 19:11:17

Temperature: 27.0



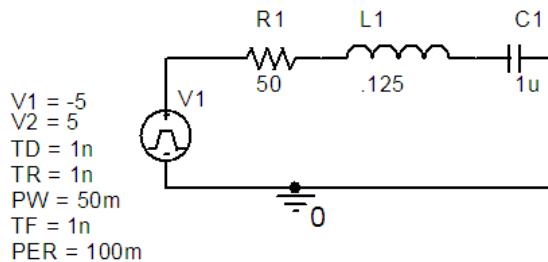
Exercice:

As instructed by the lab teacher.

LESSON 4:**Second Order RC circuit. (Lab- 4)****Time Domain (Transient) Analysis**

Example 1:

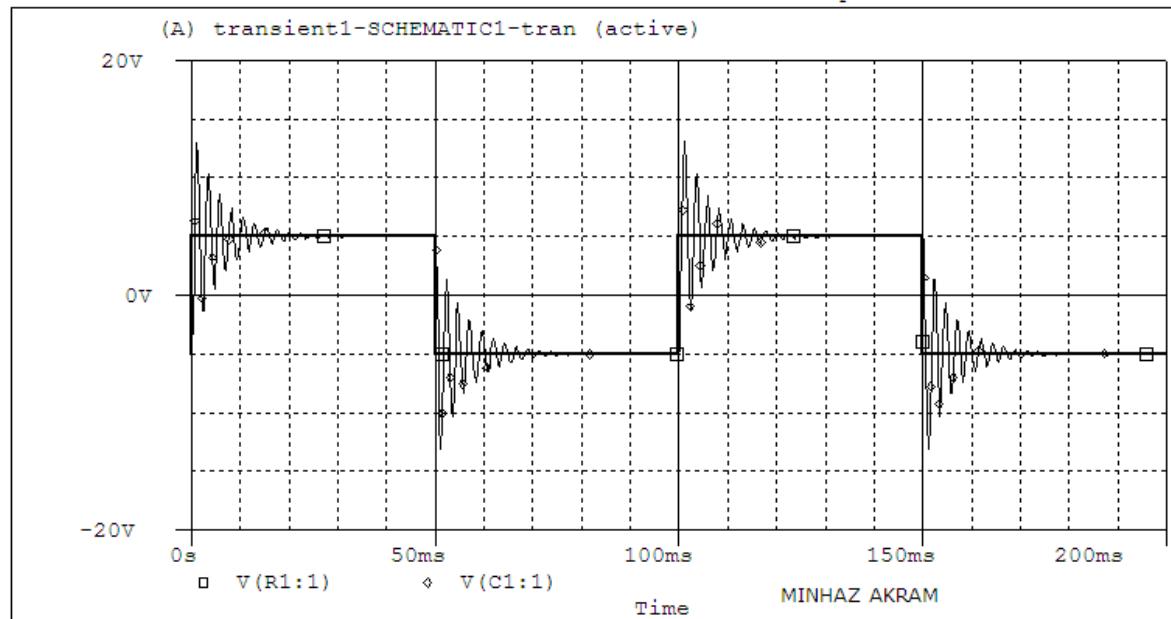
The voltage pulse shown is applied to the series RLC circuit as shown. Obtain the response for the voltage across the capacitor.

**Schematics Netlist**

```
* source TRANSIENT1
C_C1      N000030 0  1u
R_R1      N00013 N000050  50
L_L1      N000050 N000030  .125
V_V1      N00013 0
+PULSE -5 5 1n 1n 1n 50m 100m
.TRAN 0 200m 0
.PROBE
.INC "transient1-SCHEMATIC1.net"
```

Date/Time run: 11/23/07 18:57:20

Temperature: 27.0



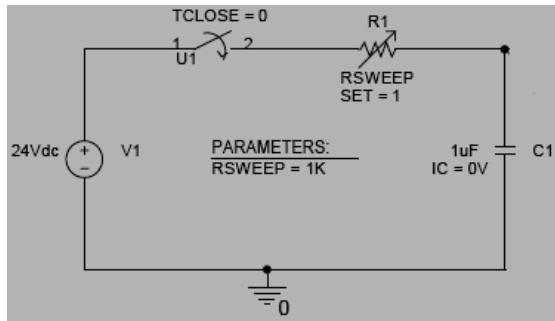
Parametric and Transient Analysis of an RC Circuit

Example 2:

Purpose: Graph the capacitor voltage as the resistance varies from 1k to 10k in steps of 1k.

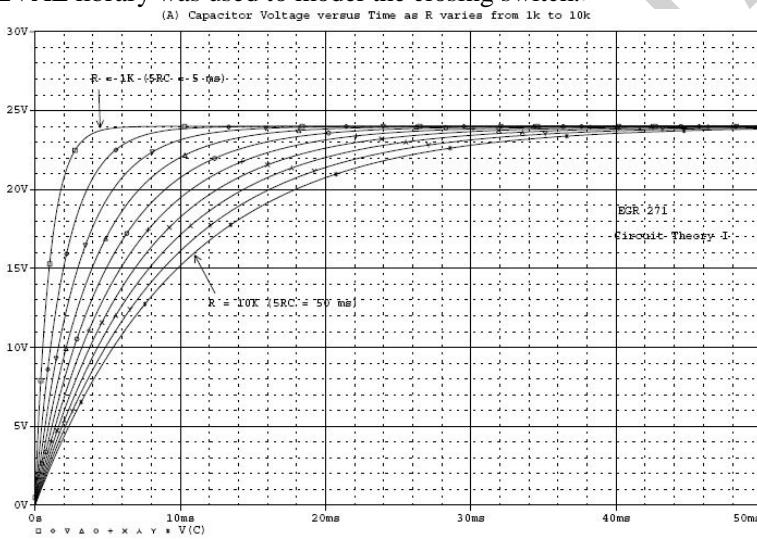
Analysis: A transient analysis from 0 to $5RC$ is performed using the largest value of R (10k). So $5RC = 5(10k)(1\mu F) = 50$ ms.

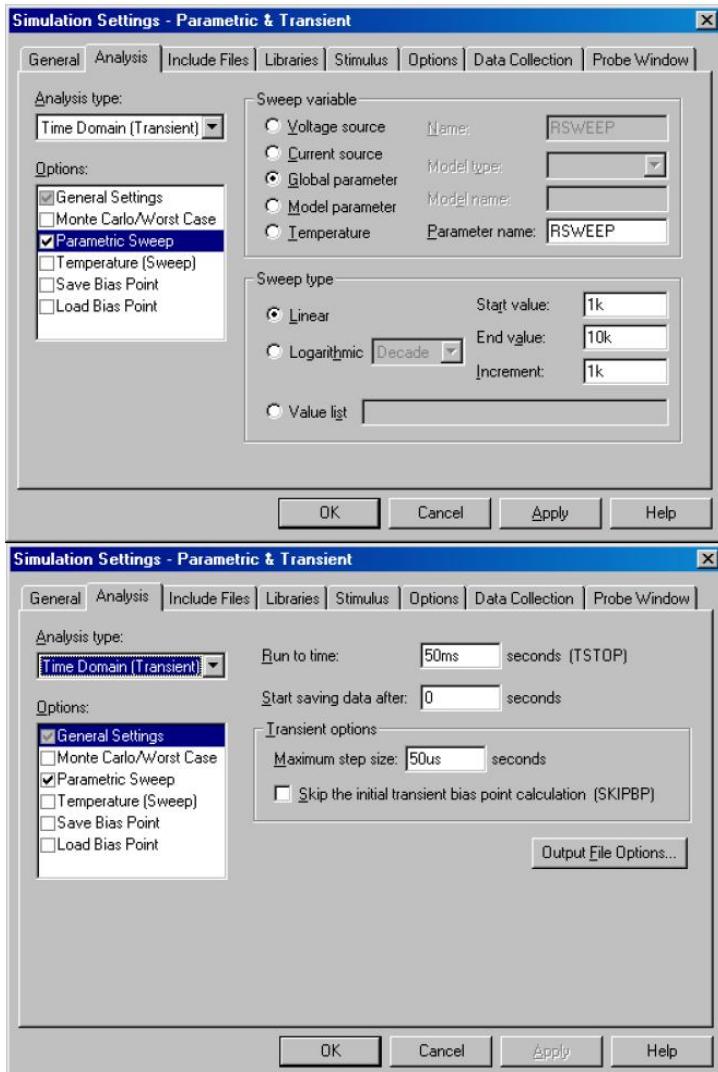
- Place the part PARAM in the schematic from the SPECIAL library.
- Edit the properties of PARAM and ADD a NEW property named RSWEEEP (for example). Right-click on RSWEEEP (while still in the property editor) and change the DISPLAY FORMAT so that both the name and the value will be displayed.
- Go back to the schematic and double-click on RSWEEEP and give it the value 1K (for example).



Note that the part R_var from the ANALOG library was used for the variable resistor. A regular resistor (part R) would work as well. Be sure to change the variable resistor's attribute SET to 1 or else the resistor values will be multiplied by the default SET value of 0.5. An OFFPAGE symbol was used to label the node as C above the capacitor.

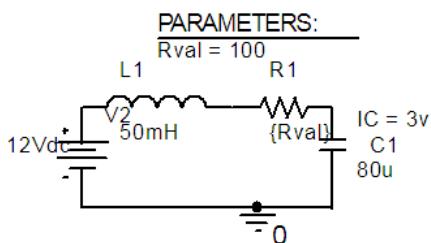
A voltage marker was added so that $V(C)$ will be automatically graphed after analysis. The part Sw_tClose from the EVAL library was used to model the closing switch.





Example 3:

In the series RLC circuit shown the switch is closed at $t = 0$. The initial voltage across the capacitor is 3 volts. The resistance in the circuit is variable. Obtain the transient response for the voltage across the capacitor for $R = 10\Omega$, 50Ω , and 100Ω . In the Transient Analysis use a final value of 50 ms.



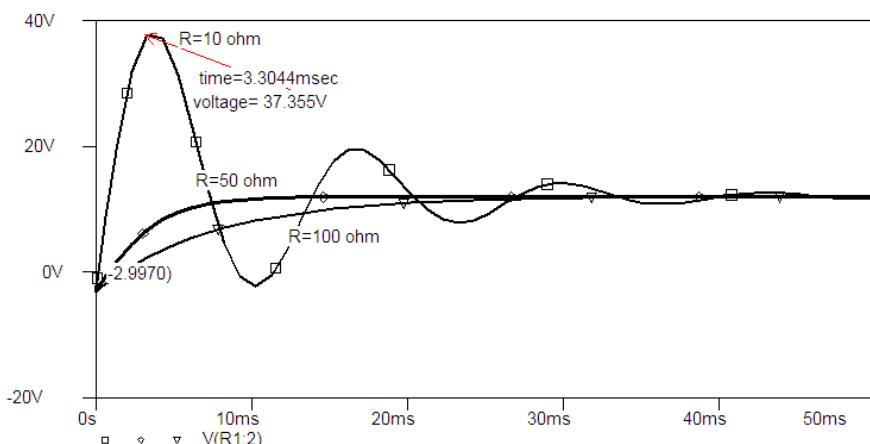
The Parametric Sweep can be used to change the value of the resistor R1. to do this get a part called PARAM from Special.slb and place it in your circuit. Double click on the text PARAMETERS. The attributes allows you to

define up to three different variable parameters. For **NAME1**= define a parameter say **R_x**, and for **VALUE1**= set an initial value of 10. Click the OK button to accept the attribute. We must now change the value of R1 to the name of the parameter. Double click on the value of R1 and in place of its value type in the text **{R_x}**. Next select the **Analysis** and then **Setup** and click the **Parametric** button. Mark **Global Parameter**, set **Name** to **R_x**. Under **Sweep Type**, mark **Value List** and set the Values: to 10 50 100. Perform the simulation.

In the Probe, from **Tools** pull-down menu select **Cursor** and check the **Display** and use **Peak** to find the peak value of the response and the time to reach this peak for the underdamped response. From the **Tools** pull-down menu use **Label** and **Mark** to mark the values for the peak point.

Schematics Netlist

```
V_Vs      1     0      DC 12
L_L1      2     3      0mH IC=0
C_C1      3     0      80U IC=3
R_R1      1     2      {R_x}
```



Example 4:

Transient Analysis of a Second Order Circuit

Purpose: To graph the capacitor voltage until it reaches steady state

Analysis type: Transient. In order to determine the length of the transient analysis, it is necessary to study the response.

Underdamped Series RLC Circuit

$$S1, S2 = -\alpha \pm j\beta$$

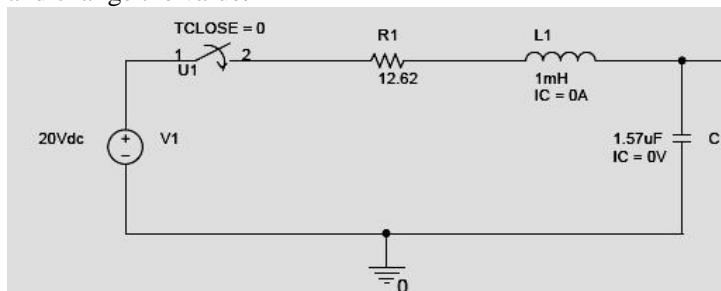
$$\alpha = R/(2L) = -12.62/(2*1e-3) = -6310$$

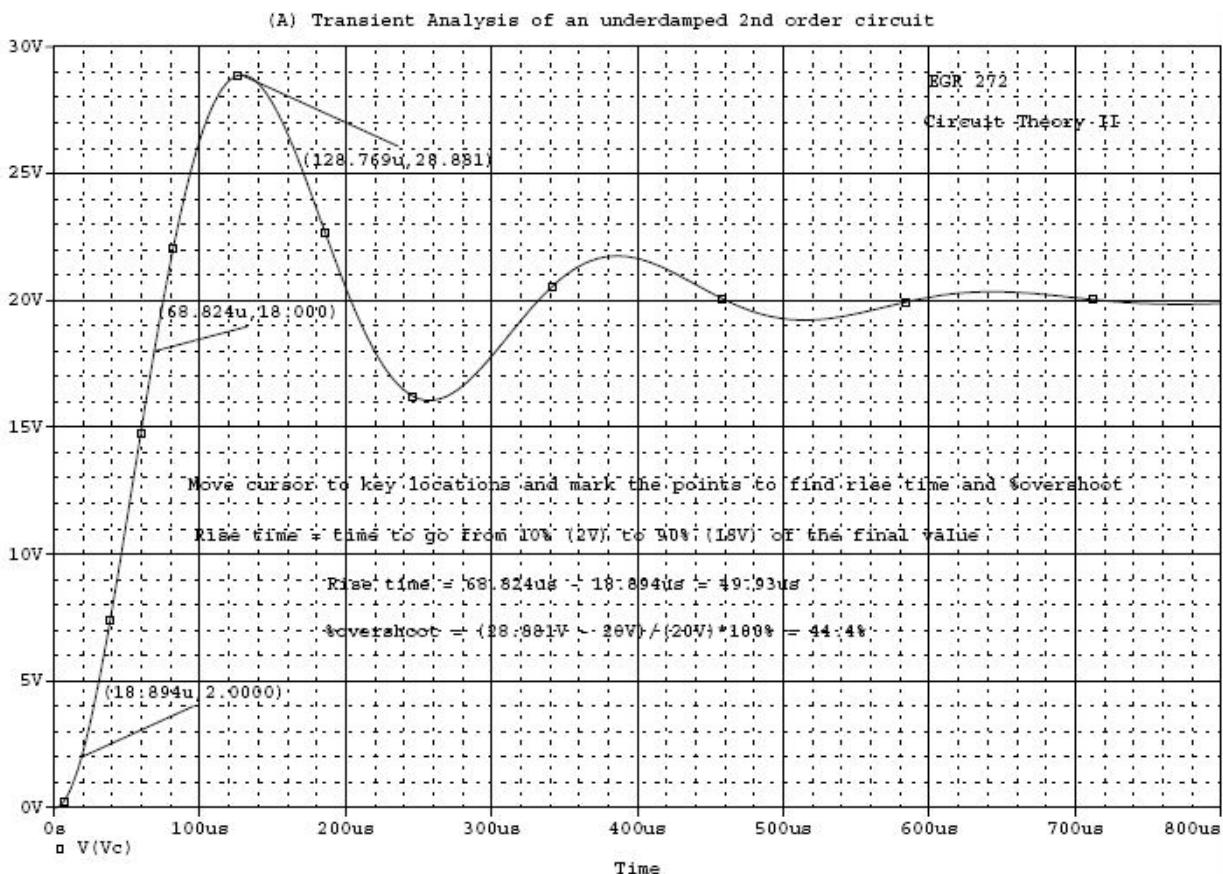
$$5\tau = 5/\alpha = 0.792 \text{ ms}, \text{ so final time for transient analysis} = 0.8 \text{ ms}$$

An opening switch (Sw_tOpen) and a closing switch (Sw_tClose) can be found in the EVAL library.

Note that initial conditions (IC) can be added to a capacitor or an inductor. See instructions below for displaying the IC attributes. Edit attributes of parts as follows:

- 1) If the attribute appears next to the part, double click it and then change its value
- 2) If the attribute does not appear next to the part, double click on the part, find the desired attribute, right click on it and select DISPLAY. Then indicate what Display Format is desired. Once the attribute has been displayed, double-click on it and change the value.





APPENDIX:

THE SINUSOID VOLTAGE:

VSIN is the exponentially damped sinusoidal voltage source.

The VSIN source has the following attributes, which are illustrated in Fig. D.25 and compared with Eq. 1:

PHASE = Phase in degrees, ϕ

DF = Damping factor (dimensionless), α

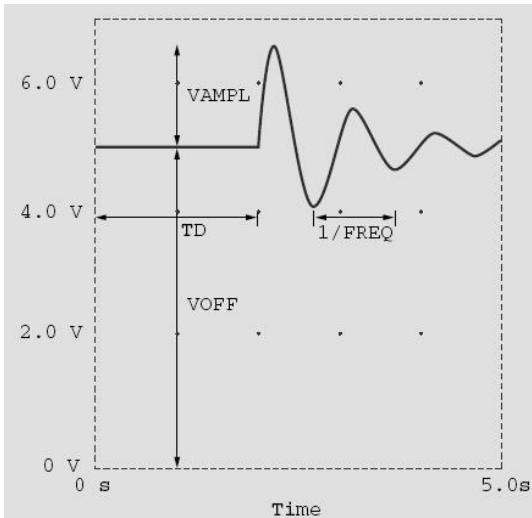
FREQ = Frequency in Hz, f

TD = Time delay in seconds, t_d

VAMPL = Amplitude, V_m

VOFF = Offset voltage, V_o

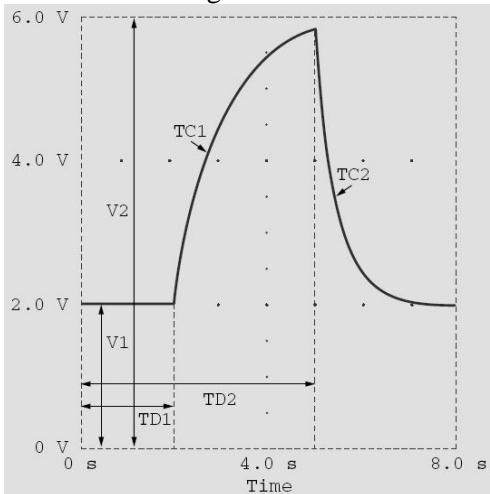
$$v(t) = V_o + V_m e^{-\alpha(t-t_d)} \sin[2\pi f(t - t_d) + \phi] \quad \text{Eq-1}$$



THE EXPONENTIAL VOLTAGE:

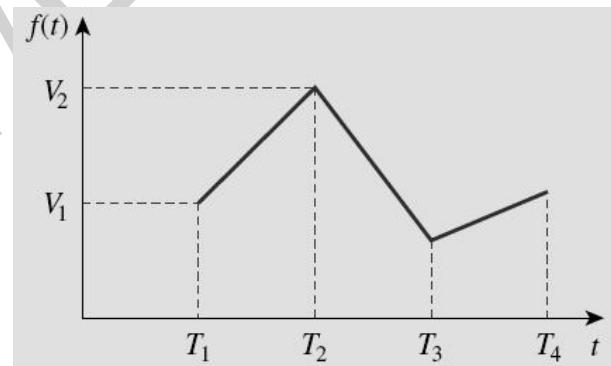
The exponential voltage source VEXP has the following attributes, typically illustrated in Fig

- TC2 = Fall time in seconds
- TD2 = Fall delay in seconds
- TC1 = Rise time constant in seconds
- TD1 = Rise delay in seconds
- V2 = Final voltage
- V1 = Initial voltage



THE PIECE WISE LINEAR VOLTAGE:

The piecewise linear voltage source VPWL, such as shown in Fig. D.28, requires specifying pairs of TN, VN, where VN is the voltage at time TN for N= 1,2,...,10 . For example, for the function in Fig. D.29, we will need to specify the attributes T1 = 0 V1 = 0 T2 = 2 V2 = 4 T3 = 6 V3 = 4 T4 = 8 V4 = 2



Exercice:

As instructed by the lab teacher.

LESSON-5 : AC ANALYSIS part 01

Using AC sweep, *PSpice* can perform AC analysis of a circuit for a single frequency or over a range of frequencies in increments that can vary linearly, by decade, or by octave. In AC sweep, one or more sources are swept over a range of frequencies while the voltages and currents of the circuit are calculated. Thus, we use AC sweep both for phasor analysis and for frequency response analysis: it will output Bode gain and phase plots. (Keep in mind that a phasor is a complex quantity with real and imaginary parts or with magnitude and phase.) While transient analysis is done in the time domain, AC analysis is performed in the frequency domain. For example, if $v_s = 10\cos(377t + 40^\circ)$ transient analysis can be used to display v_s as a function of time, whereas AC sweep will give the magnitude as 10 and phase as 40° . To perform AC sweep requires taking three steps similar to those for transient analysis:

- (1) drawing the circuit,
- (2) providing specifications, and
- (3) simulating the circuit.

1. Drawing the Circuit

We first draw the circuit using *Schematics* and specify the source(s). Sources used in AC sweep are AC sources **VAC** and **IAC**. The sources and attributes are entered into the *Schematics* as stated in the previous section. For each independent source, we must specify its **magnitude and phase**.

2. Providing Specifications

Before simulating the circuit, we need to add some specifications for AC sweep. For example, suppose we want a linear sweep at frequencies 50, 100, and 150 Hz. We enter these parameters as follows:

1. Select **Analysis/Setup/AC Sweep** to open up the dialog box for AC Sweep.
2. **CLICKL Linear** for the X axis to have a linear scale.
3. Type 3 in the *Total Pts* box.
4. Type 50 in the *Start Freq* box.
5. Type 150 in the *End Freq* box.
6. **CLICKL OK/Close** to accept specifications.

A linear sweep implies that simulation points are spread uniformly between the starting and ending frequencies. **Note that the Start Freq cannot be zero because 0 Hz corresponds to DC analysis.** If we want the simulation to be done at a single frequency, we enter 1 in step 3 and the same frequency in steps 4 and 5. If we want the AC sweep to simulate the circuit from 1 Hz to 10 MHz at 10 points per decade, we **CLICKL** on *Decade* in step 2 to make the X axis logarithmic, enter 10 in the *Total Pts* box in step 3, enter 1 in the *Start Freq* box, and enter 10Meg in the *End Freq* box. Keep in mind that a decade is a factor of 10. In this case, a decade is from 1 Hz to 10 Hz, from 10 Hz to 100, from 100 to 1 kHz, and so forth.

3. Simulating the Circuit

After providing the necessary specifications and saving the circuit, we perform the AC sweep by selecting **Analysis/Simulate**. If no errors are encountered, the circuit is simulated. At the end of the simulation, the system displays **AC analysis finished** and creates an output file with extension .out. Also, the *Orcad PSpice* program will automatically run if there are no errors. The frequency axis (or X axis) is drawn but no curves are shown yet. Select **Trace/Add** from the *Orcad Pspice* menu bar and click on the variables to be displayed. **We may also use current or voltage markers to display the traces as explained in the previous section.** To use advanced markers such as *vdb*, *idb*, *vphase*, *iphase*, *vreal*, and *ireal*, select **Markers/Mark Advanced**. In

case the resolution of the trace is not good enough, we may need to check the data points to see if they are enough. To do so, select **Tools/Options/Mark Data Points/OK** in the *Orcad PSpice* menu and the data points will be displayed. If necessary, we can improve the resolution by increasing the value of the entry in the **Total Pts** box in the **Analysis/Setup/AC Sweep and Noise Analysis** dialog box for AC sweep.

Bode plots are separate plots of magnitude and phase versus frequency. To obtain Bode plots, it is common to use an AC source, say V1, with 1 volt magnitude and zero phase. After we have selected **Analysis/Simulate** and have the *Orcad PSpice* program running, we can display the magnitude and phase plots as mentioned above. Suppose we want to display a Bode magnitude plot of V(4). We select **Trace/Add** and type **dB(V(4))** in **Trace Command** box. **dB(V(4))** is equivalent to $20\log(V(4))$, and because the magnitude of V1 or V(R1:1) is unity, **dB(V(4))** actually corresponds to $\text{dB}(V(4)/V(R1:1))$, which is the gain. Adding the trace **dB(V(4))** will give a Bode magnitude/gain plot with the axis in Y dB.

Once a plot is obtained in the *Orcad PSpice* window, we can add labels to it for documentation purposes. To add a title to the plot, select **Edit/Modify Title** in the *Orcad PSpice* menu and type the title in the dialog box. To add a label to the Y axis, select **Plot/Y Axis Settings**, type the label, and **CLICKL OK**. Add a label to the X axis in the same manner. As an alternative approach, we can avoid running the *Orcad PSpice* program by using *pseudocomponents* to send results to the output file. Pseudocomponents are like parts that can be inserted into a schematic as if they were circuit elements, but they do not correspond to circuit elements. We can add them to the circuit for specifying initial conditions or for output control. In fact, we have already used two pseudocomponents, **VIEWPOINT** and **IPROBE**, for DC analysis. Other important pseudocomponents and their usage are shown in Fig. 1 and listed in Table. The pseudo components are added to the schematic. To add a pseudocomponent, select **Draw/Get New Parts** in the *Schematics* window, select the pseudocomponent, place it at the desired location, and add the appropriate attributes as usual. Once the pseudocomponents are added to the schematic, we select **Analysis/Setup/AC Sweep** and enter the specifications for the AC sweep, and finally, select **Analysis/Simulate** to perform AC sweep. If no errors are encountered, the voltages and currents specified in the pseudocomponents will be saved in the output file. We obtain the output file by selecting **Analysis/Examine Output** in the *Schematics* window.



Figure 1: Print and plot pseudocomponents.

TABLE: Print and plot pseudocomponents.

Symbol	Description
IPILOT	Plot showing branch current; symbol must be placed in series
IPRINT	Table showing branch current; symbol must be placed in series
VPLOT1	Plot showing voltages at the node to which the symbol is connected
VPLOT2	Plot showing voltage differentials between two points to which the symbol is connected
VPRINT1	Table showing voltages at the node to which the symbol is connected
VPRINT2	Table showing voltage differentials between two points to which the symbol is connected

EXAMPLE 1: Find current i in the circuit in Fig. 2.

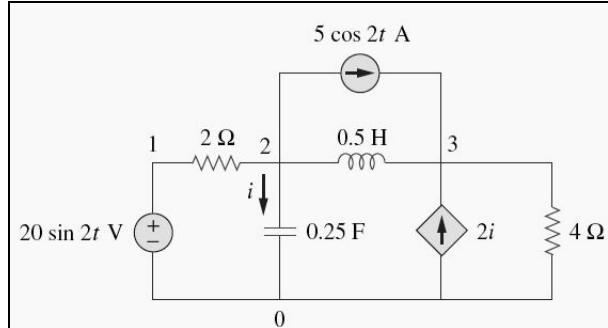


Figure: 2

Solution:

Recall that $20 \sin 2t = 20 \cos(2t - 90^\circ)$ and that $f = \omega/2\pi = 2/2\pi = 0.31831$. The schematic is shown in Fig. 3. The attributes of V1 are set as *ACMAG* = 20, *ACPHASE* = -90° ; while the attributes of IAC are set as *AC* = 5. The current-controlled current source is connected in such a way as to conform with the original circuit in Fig. 2; its gain is set equal to 2. The attributes of the pseudocomponent IPRINT are set as *AC* = yes, *MAG* = yes, *PHASE* = ok, *REAL* =, and *IMAG* =. Since this is a single-frequency ac analysis, we select **Analysis/ Setup/AC Sweep** and enter *Total Pts* = 1, *Start Freq* = 0.31831, and *Final Freq* = 0.31831. We save the circuit and select **Analysis/Simulate** for simulation. The output file includes

FREQ	IM(V_PRINT3)	IP(V_PRINT3)
3.183E-01	7.906E+00	4.349E+01

From the output file, we obtain $I = 7.906/43.49^\circ$ A or $i(t) = 7.906 \cos(2t + 43.49^\circ)$ A. This example is for a single-frequency ac analysis; Example 3 is for AC sweep over a range of frequencies.

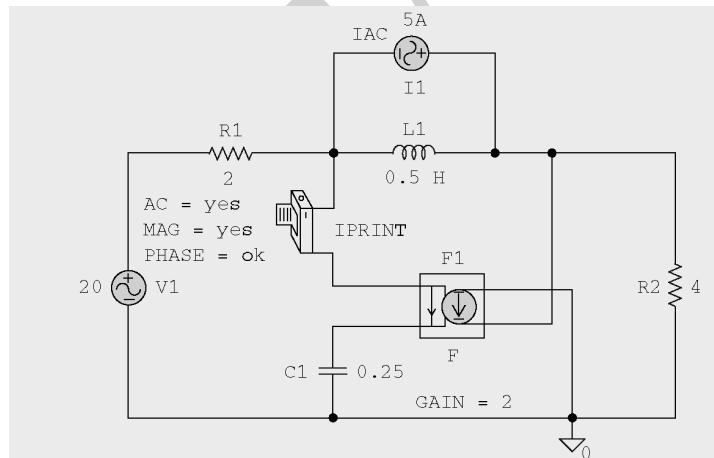


Fig: 3

Example 2: Find $i_x(t)$ in the circuit in Fig.3.

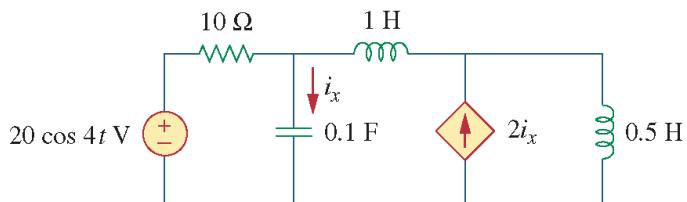


Figure :4

Answer: From the output file, $I_x = 7.59/108.43^\circ$ or $i_x = 7.59 \cos(4t + 108.43^\circ)$ A.

For the RC circuit shown in Fig.4, obtain the magnitude plot of the output voltage v_o for frequencies from 1 Hz to 10 kHz. Let $R = 1 \text{ k}\Omega$ and $C = 4 \text{ }\mu\text{F}$.

Solution:

The schematic is shown in Fig.5. We assume that the magnitude of V1 is 1 and its phase is zero; we enter these as the attributes of V1.

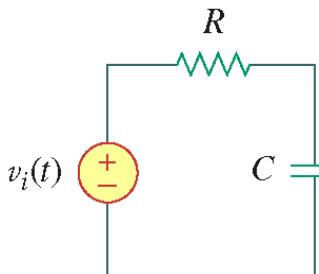


Figure.5

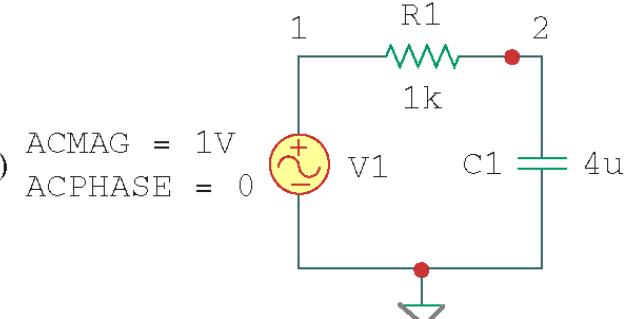


Figure. 6

The schematic of the circuit in Fig.6.

We also assume 10 points per decade. For the AC sweep specifications, we select **Analysis/Setup/AC Sweep** and enter 10 in the *Total Pts* box, 1 in the *Start Freq* box, and 10k in the *Final Freq* box. After saving the circuit, we select **Analysis/Simulate**. From the *Orcad PSpice* menu, we obtain the plot in Fig. 7(a) by selecting **Traces/Add** and clicking $V(2)$. Also, by selecting **Trace/Add** and typing $\text{dB}(V(2))$ in the *Trace Command* box, we obtain the Bode plot in Fig. D.7(b). The two plots in Fig.7 indicate that the circuit is a lowpass filter: low frequencies are passed while high frequencies are blocked by the circuit.

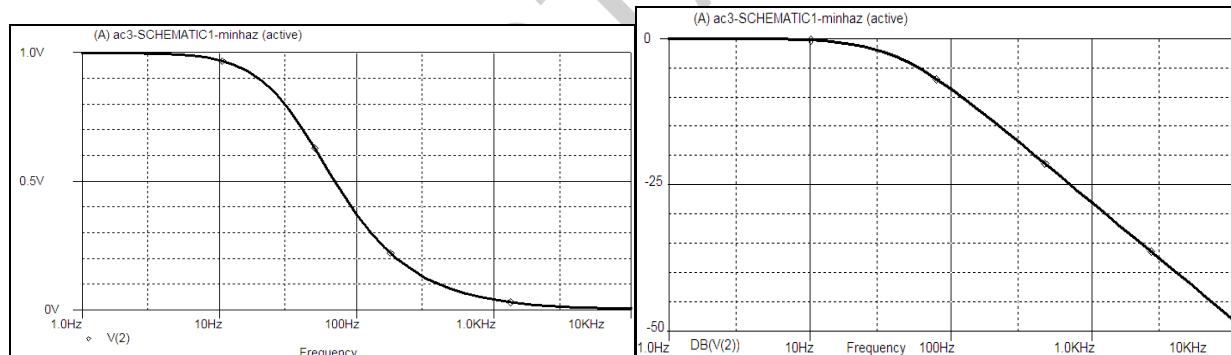


Figure 7

Result of Example 2: (a) linear, (b) Bode plot.

Practice Problem: For the circuit in Fig. 5, replace the capacitor C with an inductor $L = 4 \text{ mH}$ and obtain the magnitude plot (both linear and Bode) for v_o for $10 < f < 100 \text{ MHz}$.

Answer: See the plots in Fig.8.

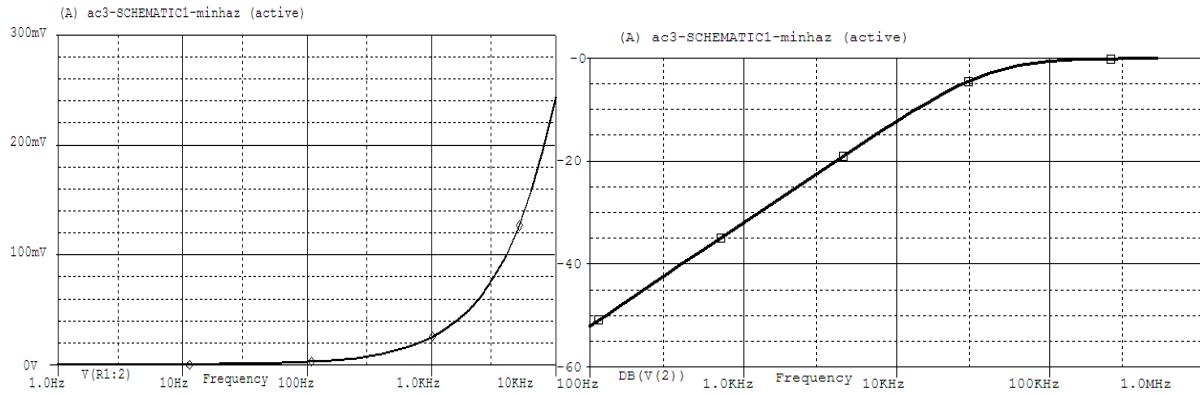
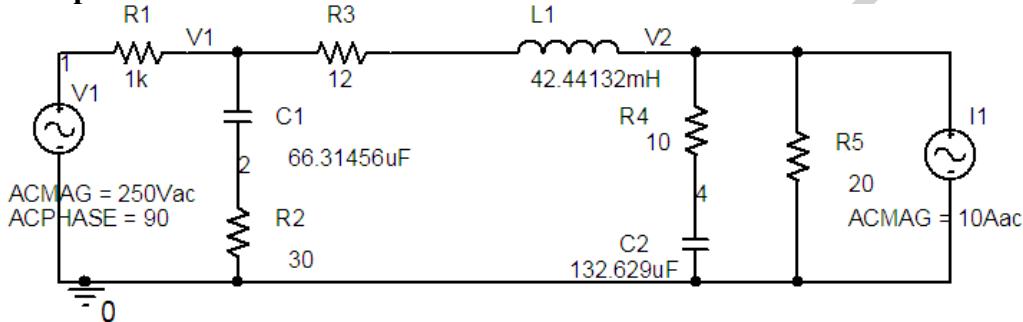


Fig: 8(a)

Fig: 8(b)

Result of Practice Prob.: (a) linear plot, (b) Bode plot.

Example1:



(a) Use node-voltage analysis to find the magnitude and phase angle of the node voltages V_1 and V_2 , then find magnitude and phase angle of current in R_3 .

$$\begin{bmatrix} 0.043 - j0.024 & -0.03 + j0.04 \\ -0.03 + j0.04 & 0.1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} .j0.25 \\ 10 \end{bmatrix}$$

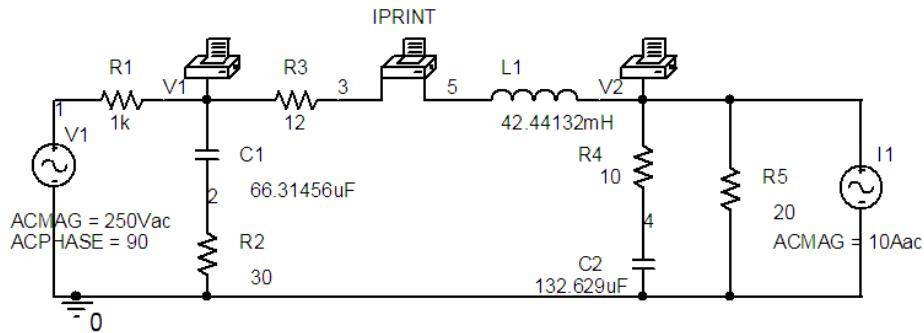
Transforming the voltage source to the current source and writing the node-voltage equation results in

$$\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 20 & 6 - j8 \\ 6 - j8 & 8.6 - j8.4 \end{bmatrix} \begin{bmatrix} j0.25 \\ 10 \end{bmatrix} = \begin{bmatrix} 96.0469 \angle 51.34^\circ \\ 99.5301 \angle -27.85^\circ \end{bmatrix} \text{ Volts}$$

or Current in R_3 is

$$I = \frac{96.0469 \angle -51.34^\circ - 99.5301 \angle -27.85^\circ}{12 + j16} = 1.9977 \angle 172.3769^\circ \text{ A}$$

(b) The voltage source and the current source frequency are 60 Hz. Use PSpice and AC Analysis for a single point 60 Hz to obtain the node voltages V_1 and V_2 , and current in R_3 . Use VPRINT, and IPRINT to send the node voltages and current to the output file. In the VPRINT and IPRINT dialog box set the attributes AC, MAG, and PHASE to yes.



The following results are extracted from the Output File

```

V_V1      1 0 DC 0Vdc AC 250Vac 90
R_R1      1 V1 1k
R_R2      2 0 30
C_C1      V1 2 66.31456uF
R_R3      V1 3 12
L_L1      5 V2 42.44132mH
R_R4      V2 4 10
C_C2      4 0 132.629uF
R_R5      V2 0 20
I_I1      V2 0 DC 0Adc AC 10Aac
V_PRINT1    3 5 0V

.PRINT      AC
+ IM(V_PRINT1)
+ IP(V_PRINT1)

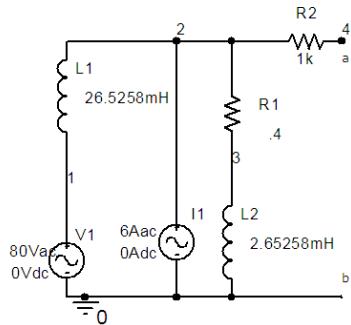
.PRINT      AC
+ VM([V1])
+ VP([V1])

.PRINT      AC
+ VM([V2])
+ VP([V2])
.AC DEC 1 60 60
.PROBE
FREQ      IM(V_PRINT1)      IP(V_PRINT1)
6.000E+01  2.143E+00      2.809E+00
FREQ      VM(V1)           VP(V1)
6.000E+01  1.040E+02      1.252E+02
FREQ      VM(V2)           VP(V2)
6.000E+01  9.750E+01      1.495E+02

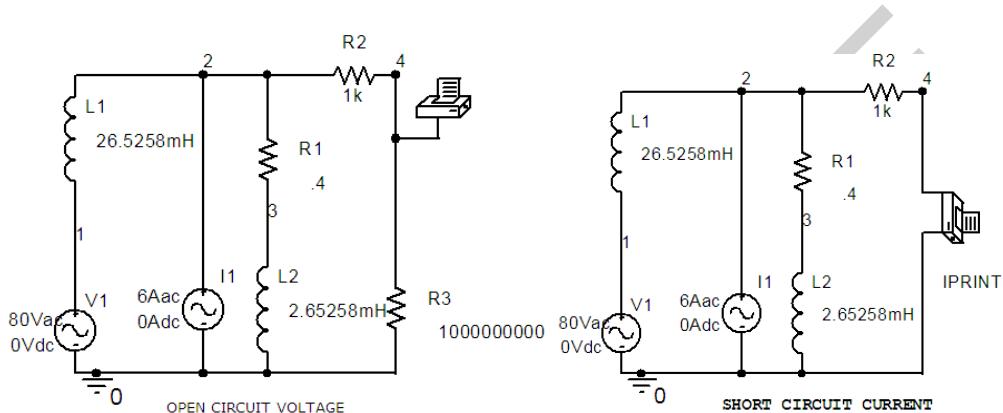
```

Example 2:

In the circuit shown find the Thevenin's equivalent circuit with respect to terminals ab.



In order to obtain the Thevenin's voltage, a high resistance, R_{inf} is connected across terminal ab. VPRINT1 is used to send the voltage magnitude and phase angle to the Output File. The AC, MAG, and PHASE attributes are set to yes. To find the Thevenin's impedance, terminal ab is shorted through an IPRINT device used as an ammeter. The AC, MAG, and PHASE attributes of IPRINT are set to yes.

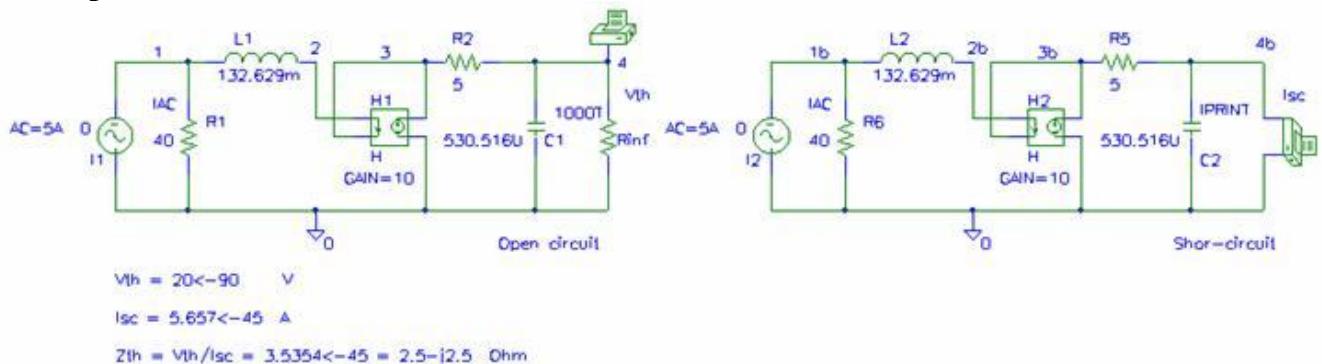


From the Output File,

```
* source AC1
L_L1      1 2 26.5258mH
V_V1      1 0 DC 0Vdc AC 80Vac
I_I1      2 0 DC 0Adc AC 6Aac
R_R1      3 2 .4
L_L2      0 3 2.65258mH
R_R2      2 4 1k
R_R3      0 4 1000000000
.PRINT    AC VM([4]) VP([4])
*Analysis directives:
.AC LIN 1 60 60
.PROBE
FREQ      VM(4) VP(4)
6.000E+01 9.785E+00 -5.659E+01
FREQ      IM(V_PRINT2) IP(V_PRINT2)
6.000E+01 9.782E-03 -5.664E+01
```

$$Z_{th} = \frac{V_{th}}{I_{sh}} = 2.33236 \angle 30.96^\circ = 2 + j1.2 \Omega$$

Example 3:



In the following circuits a single point 60 Hz AC Analysis is set up for an open circuit and a short-circuit conditions to obtain the Thevenin's equivalent circuit. VPRINT1 and an IPRINT are used to include V_{th} and I_{sc} in the output file. The AC, MAG, and PHASE attributes of VPRINT1 and IPRINT are set to yes.

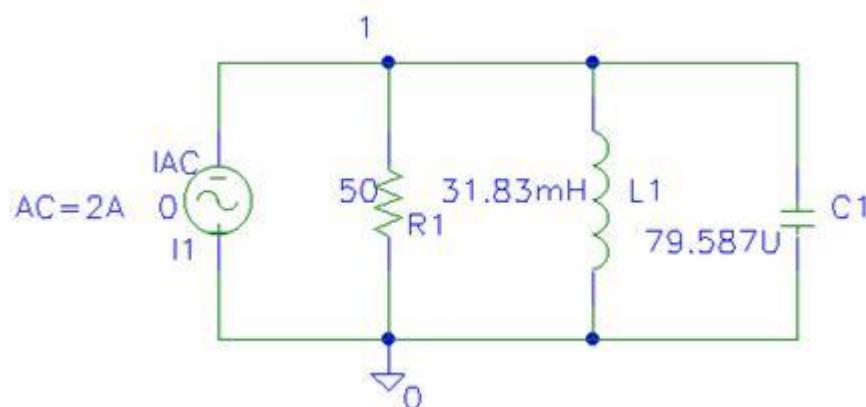
From the Output File,

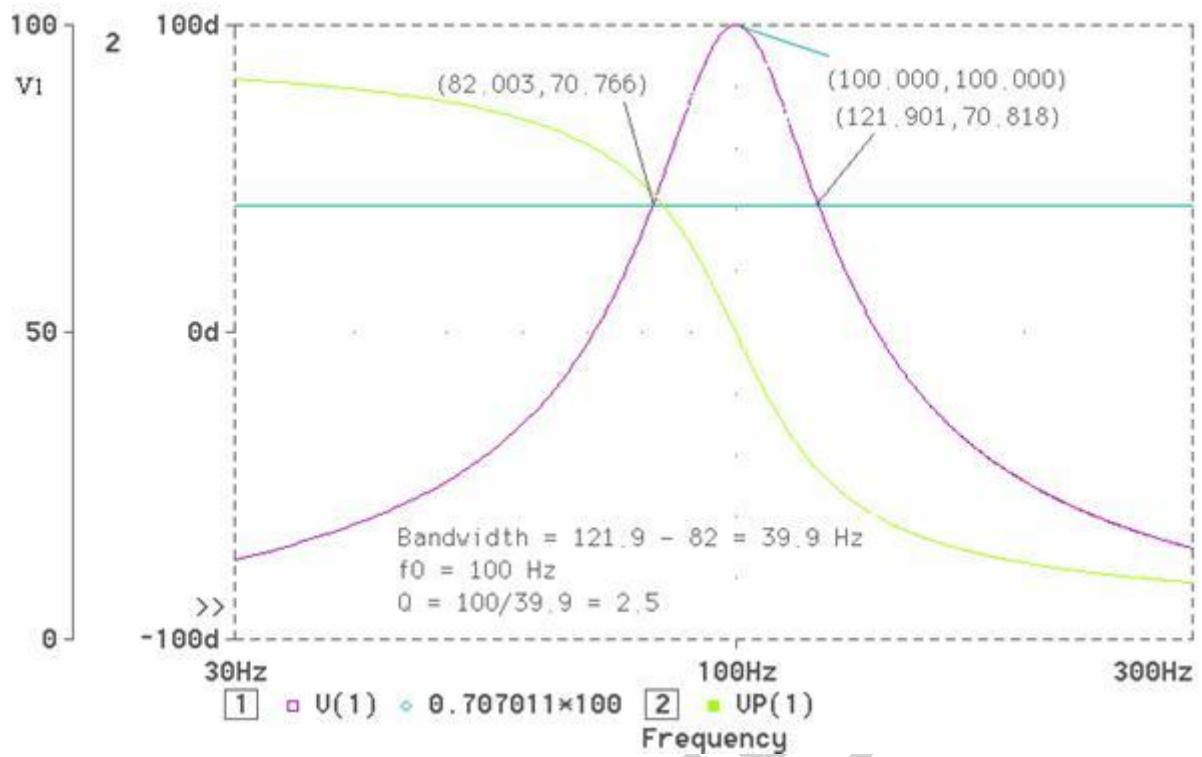
$$V_{th} = 20.57 < 40^\circ \text{ V}$$

$$Z_{th} = \frac{V_{th}}{I_{sc}} = 3.5354 < -45^\circ = 2.5 - j2.5 \Omega$$

Example 4:

For the parallel RLC circuit shown, use PSpice and Probe to plot voltage magnitude and phase angle over a range of 30 to 300 Hz in increment of 1 Hz (Total point =271). Determine the resonant frequency. Also draw the 0.707 V_{0(max)} line and estimate the circuit bandwidth and the Q-factor.





Exercice:

As instructed by the lab teacher.

LESSON-6: AC ANALYSIS part 2

Example 1:

For the series RLC circuit shown, use PSpice, and AC Analysis to sweep the frequency from 10Hz to 100KHz over a decade variation with 200 points per decade.

(a) Use Probe to plot the current magnitude and phase angle. Determine the resonant frequency. Also draw the 0.707(V0(max)) line and estimate the circuit bandwidth b, and the Q-factor.

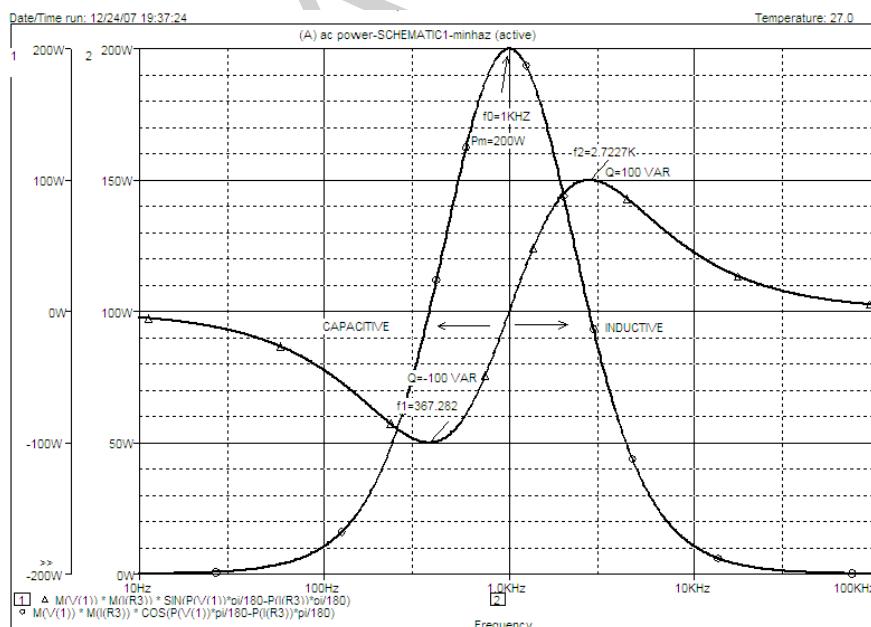
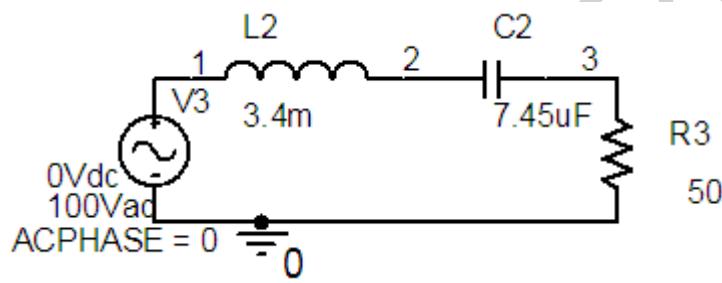
(b) On a separate graph plot the real and reactive power delivered to the circuit as a function of frequency.

$$P = V(1) * I(R1) \cos(VP(1) * pi/180 - IP(R1) * pi/180)$$

$$Q = V(1) * I(R1) \sin(VP(1) * pi/180 - IP(R1) * pi/180)$$

(c) On a separate graph, plot the power factor as a function of frequency.

$$PF = \cos(IP(R1) * pi/180)$$

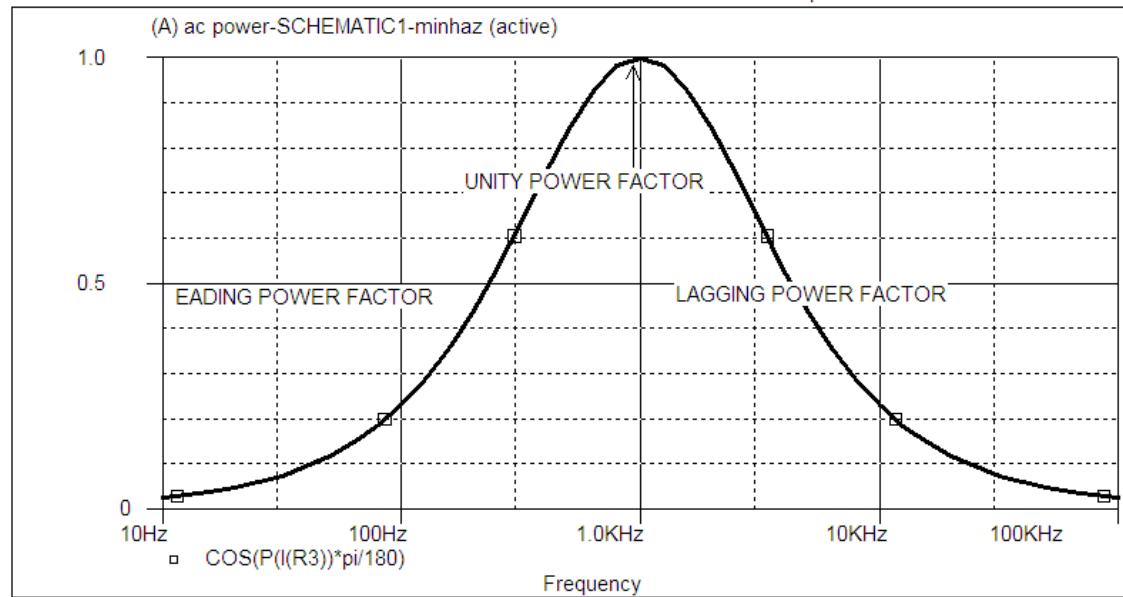


Hints : $M(V(3)) * M(I(V3)) * \sin(P(V(3)) * \pi/180 - P(I(V3)) * \pi/180)$

= Magnitude(voltage) * Magnitude (current) * SIN(Phase Differernce)

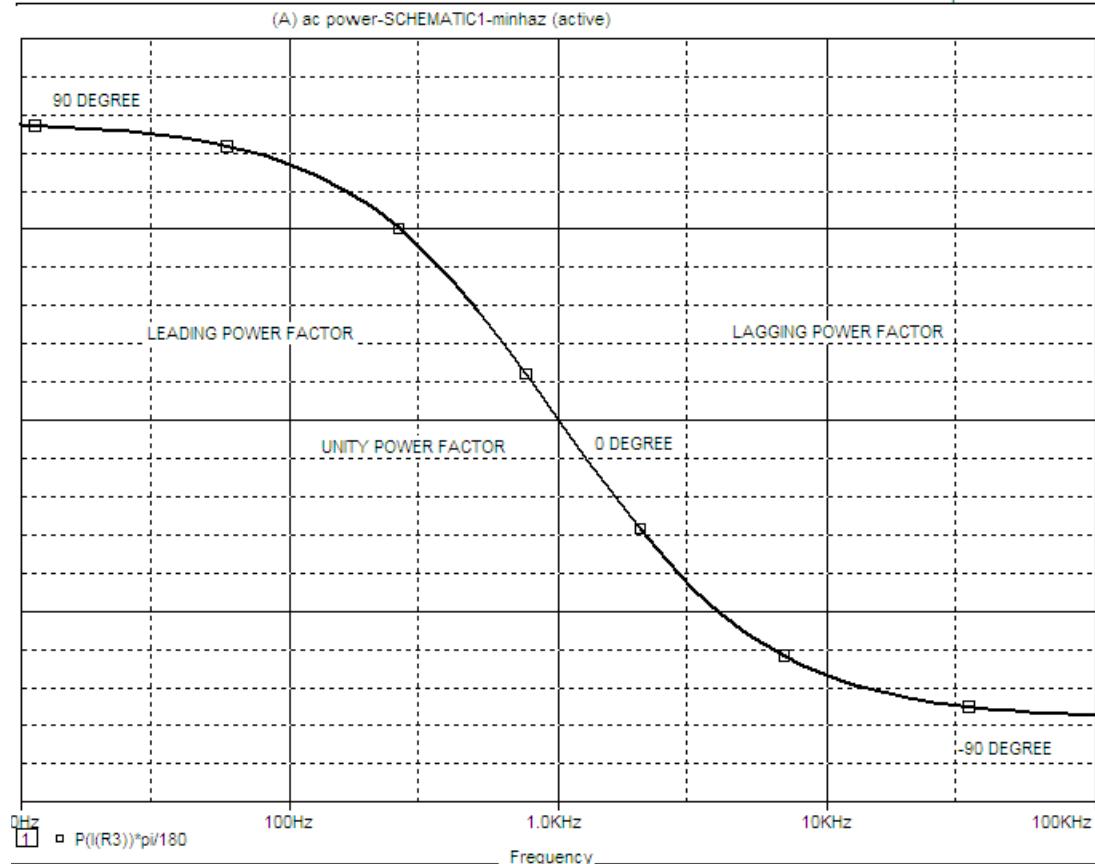
Date/Time run: 12/14/07 00:36:36

Temperature: 27.0



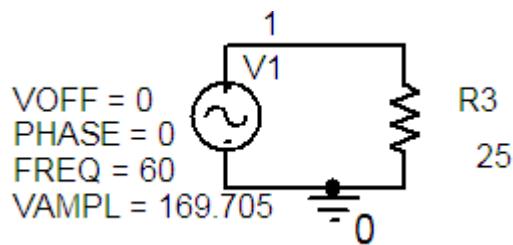
Date/Time run: 12/14/07 00:36:36

Temperature: 27.0



Example 2:

- Determine the real power in the circuit shown.
- Use PSpice and Transient Analysis to obtain a plot of the instantaneous voltage $v_1(t)$, $i(t)$, and the instantaneous power $p(t)$.

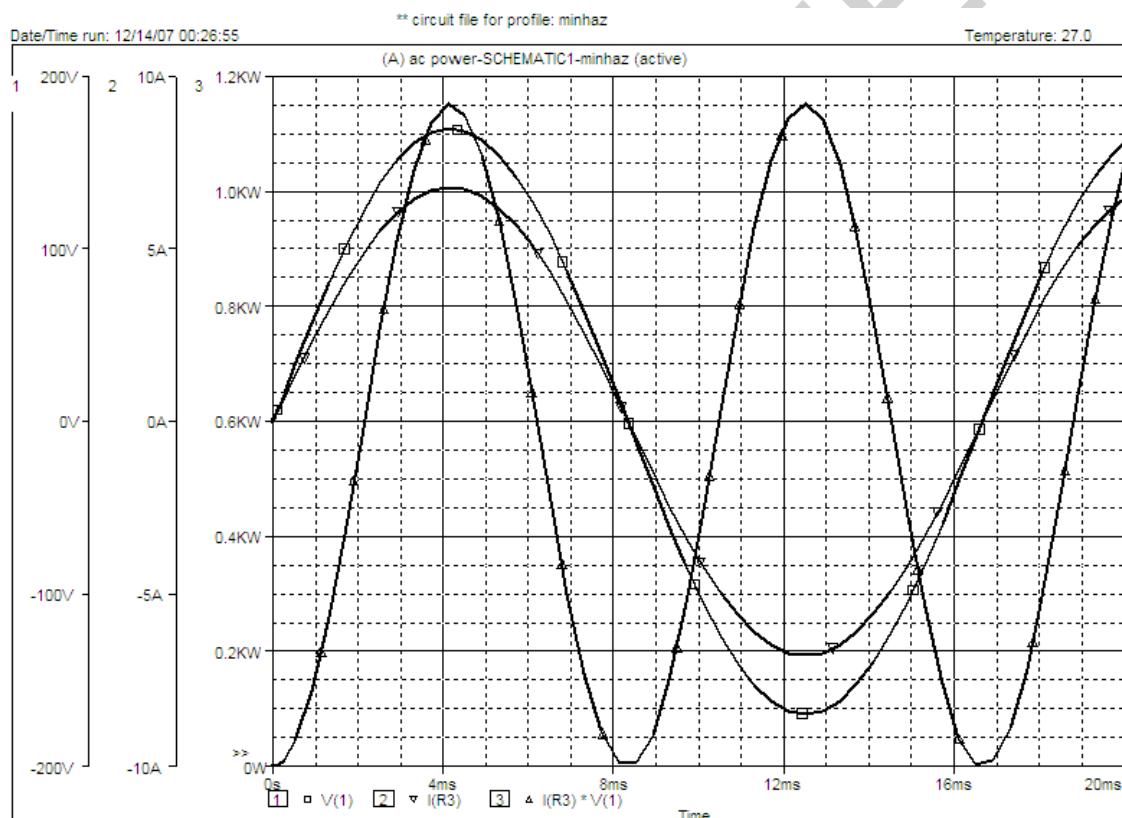


$$V_1 \frac{169.705}{\sqrt{2}} = 120 \text{ V} \quad I = \frac{120 \angle 0}{25} = 4.8 \text{ A}$$

(i)

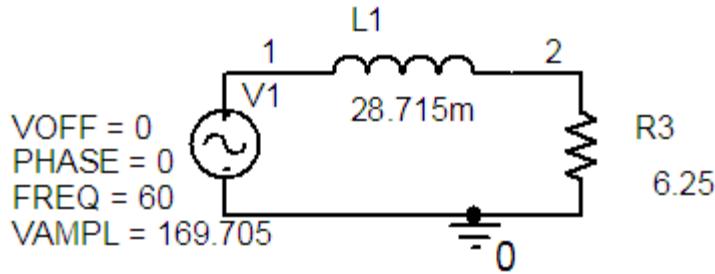
$$P = VI \cos \theta = (120)(4.8)(1) = 576 \text{ W}$$

(ii) The Final Time in the Transient Analysis is set to $T = 1/60 = 16.667\text{ms}$. The following instantaneous quantities and the average power are plotted.



Example 3:

- (i) Determine the real power in the circuit shown.
- (ii) Use PSpice and Transient Analysis to obtain a plot of the instantaneous voltage $v_1(t)$, $i(t)$, and the instantaneous power $p(t)$.



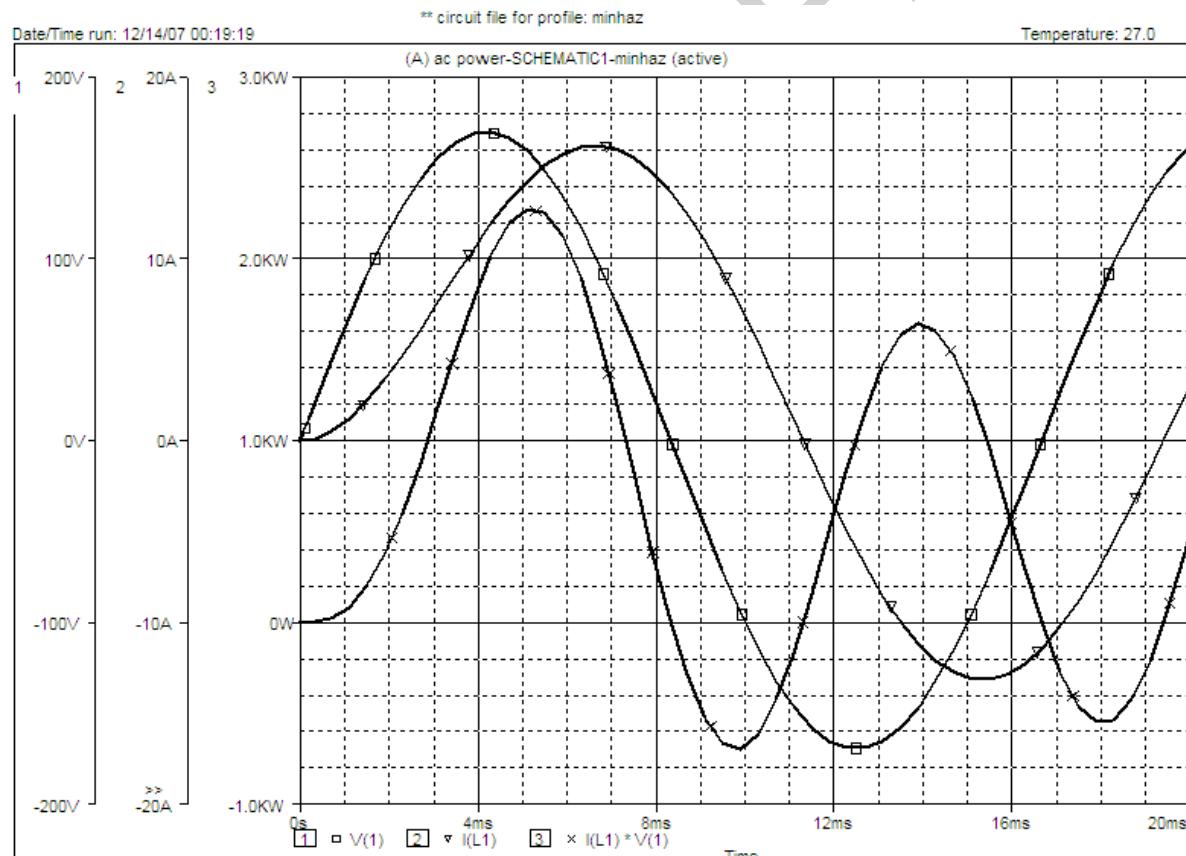
$$v_1(t) = 169.705 \sin(2\pi 60t) \text{ Volt}$$

$$Z = 6.25 + 2\pi 60L = 12.5 \angle 60 \Omega$$

$$I = \frac{120 \angle 0}{12.5 \angle 60} = 9.6 \angle -60 \text{ A}$$

$$P = VI \cos \theta = (120)(9.6)(\cos 60) = 576 \text{ W}$$

(ii) The Final Time in the Transient Analysis is set to $T = 1/60 = 16.667\text{ms}$. The following instantaneous quantities and the average power are plotted



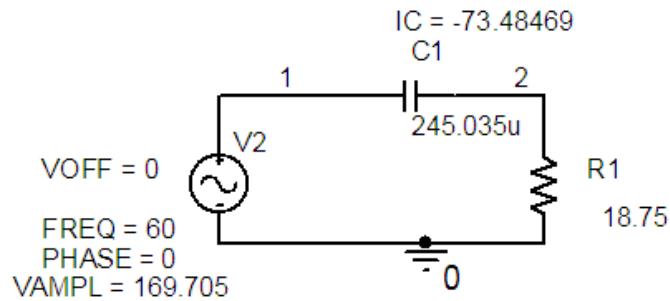
Example 4:

(i) Determine the real power in the circuit shown.

$$v_1(t) = 169.705 \sin(2\pi 60t) \text{ Volt}$$

(ii) Use PSpice and Transient Analysis to obtain a plot of the instantaneous voltage $v_1(t)$, $i(t)$,

and the instantaneous power $p(t)$.



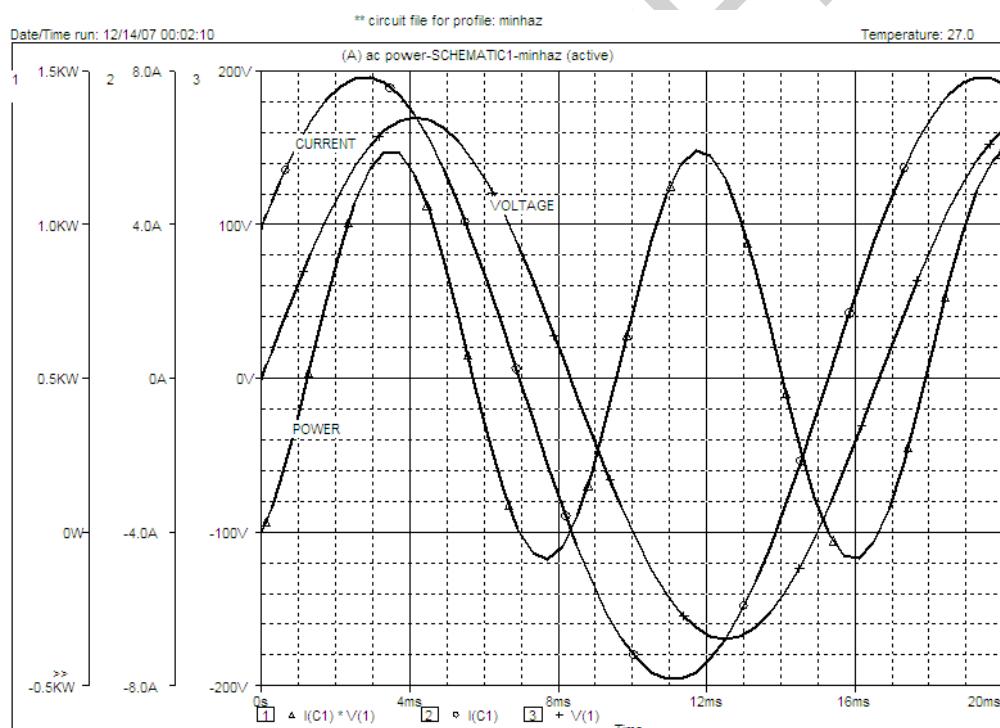
$$I = \frac{120\angle 0}{21.65\angle -30} = 5.543\angle -30 \text{ A}$$

$$Z = 18.75 - j \frac{1}{2\pi 60 C} = 21.65\angle -30 \Omega$$

$$I = \frac{120\angle 0}{21.65\angle -30} = 5.543\angle -30 \text{ A}$$

$$P = VI \cos \theta = (120)(5.543)(\cos 30) = 576 \text{ W}$$

(ii) The Final Time in the Transient Analysis is set to $T = 1/60 = 16.667\text{ms}$. The following instantaneous quantities and the average power are plotted



Exercice:

As instructed by the lab teacher.

LESSON-7+8: PSPICE (simulation using code)

Some Facts and Rules about PSpice

- PSpice is not case sensitive. This means that names such as *Vbus*, *VBUS*, *vbus* and even *vBuS* are equivalent in the program.
- All element names must be unique. Therefore, you can't have two resistors that are both named "Rbias," for example.
- The first line in the data file is used as a title. It is printed at the top of each page of output. You should use this line to store your name, the assignment, the class and any other information appropriate for a title page. PSpice will ignore this line as circuit data. Do not place any actual circuit information in the first line.
- There must be a node designated "0." (Zero) This is the reference node against which all voltages are calculated.
- Each node must have at least two elements attached to it.
- The last line in any data file must be ".END" (a period followed by the word "end.")
- All lines that are not blank (except for the title line) must have a character in column 1, the leftmost position on the line.
 - Use "*" (an asterisk) in column 1 in order to create a comment line.
 - Use "+" (plus sign) in column 1 in order to continue the previous line (for better readability of very long lines).
 - Use "." (period) in column 1 followed by the rest of the "dot command" to pass special instructions to the program.
 - Use the designated letter for a part in column 1 followed by the rest of the name for that part (no spaces in the part name).
- Use "whitespace" (spaces or tabs) to separate data fields on a line.
- Use ";" (semicolon) to terminate data on a line if you wish to add commentary information on that same line.

The above basic information is essential to using PSpice. Learn and understand these issues now to facilitate your use of the program

The Most Basic Parts

Here, we present the simplest circuit elements. Knowing how to model these ideal, linear circuit elements is an essential start to modeling more complex circuits. In each case, we will only present the most fundamental version of the part at this time. Later we will show you more sophisticated uses of the part models.

Ideal Independent Voltage Sources

We begin with the DC version of the ideal independent voltage source. This is the default form of this class of part. The beginning letter of the part name for all versions of the ideal independent voltage source is "V." This is the character that must be placed in column 1 of the line in the text file that is

used to enter this part. The name is followed by the positive node designation, then the negative node designation, then an optional tag: "DC" followed by the value of the voltage. The tag "DC" (or "dc" if you prefer) is optional because it is the default. Later, when we begin modeling AC circuits and voltage sources that produce pulses and other interesting waveforms, we will be required to designate the type of source or it will default back to DC.

One of the interesting uses of ideal independent voltage sources is that of an *ammeter*. We can take advantage of the fact that PSpice saves and reports the value of current entering the positive terminal of an independent voltage source. If we do not actually require a voltage source to be in the branch where we want to measure the current, we simply set the voltage source to a zero value. It still calculates the current in the branch. In fact, we *require* an independent voltage source in a branch where that branch's current is the controlling current for a current-controlled dependent source.

Examples:

```
*name +node -node type value comment
Va      4      2      DC    16.0V; "V" after "16.0" is optional
vs      qe     qc      dc    24m ; "QE" is +node & "qc" is -node
VWX     23     14      18k   ; "dc" not really needed
vwx     14     23      DC    -1.8E4 ; same as above
Vdep    15     27      DC    0V    ; V-source used as ammeter
```

Resistors

Although PSpice allows for sophisticated temperature-dependent resistor models, we will begin with the simple, constant-value resistor. The first letter of the name for a resistor must be "R." The name is followed by the positive node, then the negative node and then the value in ohms or some multiple of ohms. The value of resistance will normally be positive. Negative values are allowed in order to permit an alternative model of an energy source. A value of zero, however, will produce an error. Later, we will introduce special resistor models that will permit additional analysis methods to be used.

The resistor is not an active device, so the polarity of its connection has no effect on the values of the voltages and currents reported in the solution. However, the current through a resistor is reported as that which flows from the node on the left to the node on the right in the source code line in which the resistor is entered. Thus .PRINT statements and PROBE queries that report resistor current may show negative values of current depending on the order in which you list the resistor's two nodes in the *.CIR file. If you want to see the resistor's current as a positive value, reverse the order of the nodes on the resistor's line in the *.CIR file and re-run the analysis. Nothing else will be affected and both solutions can be correct.

Examples:

```
*name +node -node value comment
Rabc   31      0      14k ; reported current from 31 to 0
Rabc   0       31     14k ; reported current changes sign
rshnt  12      15     99m ; 0.099 ohm resistor
Rbig   19      41    10MEG ; 10 meg-ohm resistor
```

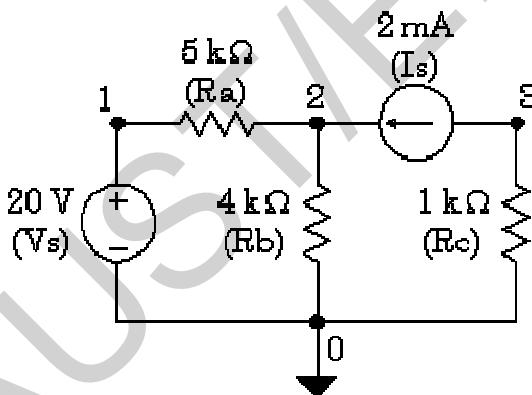
Ideal Independent Current Sources

The name of an ideal independent current source begins with the letter "I" in column 1 of the data file. As with the independent voltage source, we begin by introducing only the DC form of this part, but several other forms exist. Since the current source, is an active element, it matters greatly how it is connected. Designated current flows into the node written on the left, through the current source, out the node written on the right. As with the independent voltage source, the default type is DC. Remember that the so-called +node on a current source may have a negative voltage with respect to the so-called -node. This is due to the fact that the circuit external to the current source determines its voltage.

Examples:

```
*name +node -node type value comment
Icap 11 0 DC 35m ; 35mA flows from node 11 to 0
ix 79 24 1.7 ; "DC" not needed
I12 43 29 DC 1.5E-4 ;
I12 29 43 dc -150uA ; same as above
```

Circuit Example 1



Example_1 EXMPL01.CIR

```
Vs 1 0 DC 20.0V ; note the node placements
Ra 1 2 5.0k
Rb 2 0 4.0k
Rc 3 0 1.0k
Is 3 2 DC 2.0mA ; note the node placements
.END
```

The output file EXMPL01.OUT is below. This has been edited to remove extra lines.

```
Example_1 EXMPL01.CIR                                     <== Title Line
Vs 1 0 20.0V ; note the node placements
Ra 1 2 5.0k
Rb 2 0 4.0k
```

```

Rc 3 0 1.0k
Is 3 2 2.0mA ; note the node placements

Example_1 EXMPL01.CIR           <== Title Line

NODE VOLTAGE NODE VOLTAGE NODE VOLTAGE NODE VOLTAGE
( 1) 20.0000 ( 2) 13.3330 ( 3) -2.0000    <==Results

VOLTAGE SOURCE CURRENTS
NAME CURRENT
Vs -1.333E-03      <== Current entering node 1 of Vs

TOTAL POWER DISSIPATION 2.67E-02 WATTS

JOB CONCLUDED

TOTAL JOB TIME .26

```

This was the bare-bones minimum problem we could ask of PSpice. Note that we obtained the node voltages which is sufficient information to calculate the resistor currents. However, there is another command that we can use to get even that done by PSpice.

Use of the .PRINT Command

One of the many "dot commands" in PSpice is the .PRINT command. It has many uses, but we will concentrate here on using it for printing DC voltages and currents. The .PRINT command can be repeated as often as necessary in an analysis. You can list as many items on a line as you wish.

However, we must keep in mind that the .PRINT command was designed to work with a DC or an AC sweep. This is a method of varying a parameter over a range of values so that we get a batch of cases solved all at once. Often, we do not actually want to run a sweep over many values of a parameter. We can circumvent the sweep by setting its range so that it can only run one value. Usually, a DC sweep is made by changing the values of a source; although we will later learn to sweep over other circuit parameters. For now, let's look at the syntax for a DC sweep command with the default linear type range.

```
.DC Sweep_Variable Starting_Value Stopping_Value Increment
```

For our example problem, we choose the voltage source and set the sweep variable range so that it cannot run more than one value:

```
.DC Vs 20.0 20.0 1.0
```

Since the starting value equals the stopping value, the analysis will only run for one case, i.e., for Vs at 20 volts. Remember that the only reason we are running the DC sweep statement is to *enable* the .PRINT command. The .PRINT command will not work unless there is a sweep going on. Note: What you enter in the .DC statement *overrides* any voltage value you may have placed in the part listing for the source.

Printing DC Voltages

In addition to printing the node voltages in which you type the letter "V" with the node number in parentheses, you can print the voltage between any pair of nodes; ergo, V(m,n) prints the voltage from node "m" to node "n."

```
.PRINT DC V(1) V(2) V(3) ; prints the node voltages  
.PRINT DC V(1,2) ; prints the voltage across Ra  
.PRINT DC V(3,2) ; prints the voltage across Is
```

Printing DC Currents

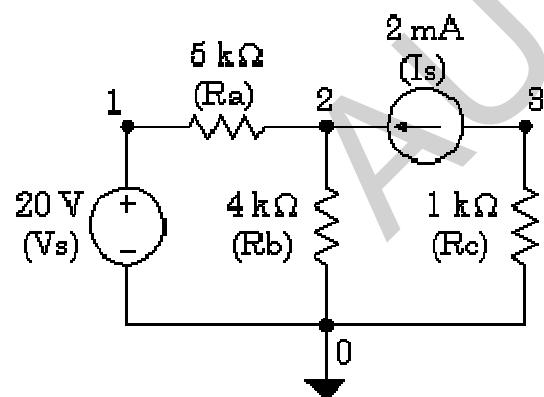
To print currents, you type the letter "I" with the element name in parentheses. Note that the reported current is that which flows into the element from the node listed on the left in the *.CIR file, through the element, and out the node listed on the right in the *.CIR file. If you want to change the sign of the reported current in a resistor, then swap the two nodes for that resistor.

```
.PRINT DC I(Ra) ; prints the currents from + to - of Ra  
.PRINT DC I(Rb) I(Rc) ; prints the currents through Rb and Rc
```

Print Commands can be Combined

```
.PRINT DC V(1,2) I(Ra) ; voltage and current for Ra  
.PRINT DC V(2,0) I(Rb) ; V(2,0) same as V(2)  
.PRINT DC V(3,0) I(Rc) ; V(3,0) same as V(3)
```

Use .PRINT with Previous Example



Example_2 EXMPL02.CIR

```
Vs    1    0    DC    20.0V ; note the node placements  
Ra    1    2    5.0k  
Rb    2    0    4.0k  
Rc    3    0    1.0k  
Is    3    2    DC    2.0mA ; note the node placements  
.DC Vs 20 20 1 ; this enables the .print commands  
.PRINT DC V(1,2) I(Ra)
```

```
.PRINT DC V(2) I(Rb)
.PRINT DC V(3) I(Rc)
.END
```

The output file EXMPL02.OUT is below. This has been edited to remove extra lines.

```
Example_2 EXMPL02.CIR
Vs 1 0 20.0V ; note the node placements
Ra 1 2 5.0k
Rb 2 0 4.0k
Rc 3 0 1.0k
Is 3 2 2.0mA ; note the node placements
.DC Vs 20 20 1 ; this enables the Print commands
.PRINT DC V(1,2) I(Ra)
.PRINT DC V(2) I(Rb)
.PRINT DC V(3) I(Rc)
```

```
Example_2 EXMPL02.CIR
**** DC TRANSFER CURVES TEMPERATURE = 27.000 DEG C
Vs      V(1,2)      I(Ra)
2.000E+01 6.667E+00 1.333E-03 <== data for Ra
Example_2 EXMPL02.CIR
**** DC TRANSFER CURVES TEMPERATURE = 27.000 DEG C
Vs      V(2)      I(Rb)
2.000E+01 1.333E+01 3.333E-03 <== data for Rb
Example_2 EXMPL02.CIR
**** DC TRANSFER CURVES TEMPERATURE = 27.000 DEG C
Vs      V(3)      I(Rc)
<2.000E+01 -2.000E+00 -2.000E-03 <== data for Rc
JOB CONCLUDED
TOTAL JOB TIME .13
```

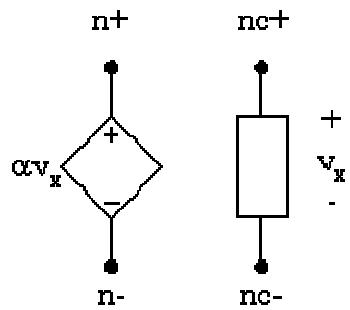
With a little bit of effort, we can get PSpice to do most of the work, most of the time. Note that using .PRINT has suppressed the default printing of all the node voltages. This is not a problem in our case because we printed all three node voltages anyway. Be sure that you include everything you need in the .PRINT statements.

Lesson@ 4.2

Simple Dependent Sources

We now extend our circuit parts list by adding the most basic dependent sources. The four dependent sources we now encounter are simple multiples of the controlling voltage or current. It is possible to model dependent sources that are complex nonlinear functions of several controlling voltages and/or currents. However, we will now concentrate on the basic linear dependent sources.

Voltage Controlled Dependent Voltage Source

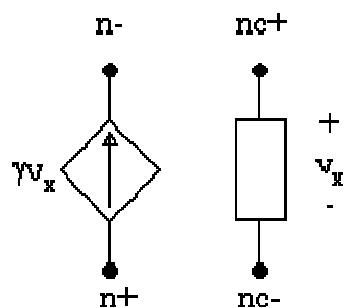


In the above figure, we find the dependent source whose positive terminal is designated as "n+" and whose negative terminal is designated as "n-." The controlling voltage is a branch voltage at some other circuit location. In this case, the positive terminal of the controlling branch is designated as "nc+" while the negative terminal is designated as "nc-." The "gain" of the dependent voltage source is α , a dimensionless quantity. For example, if v_x happened to be 16.0 volts while $\alpha = 4$, then node "n+" would be at 64.0 volts higher potential than node "n-."

The first letter of the part name for the voltage-controlled dependent voltage source is "E." This is the letter that must appear in column 1 of the *.CIR file describing the circuit. Some examples of the voltage-controlled dependent voltage source PSpice entries follow.

*Name	n+	n-	nc+	nc-	gain
Ebar	17	8	42	18	24.0; gain is 24
efix	3	1	11	0	20.0
efix	3	1	0	11	-20.0; same as above
efix	1	3	11	0	-20.0; same as above
efix	1	3	0	11	20.0; same as above
Ellen	12	0	20	41	16.0

Voltage Controlled Dependent Current Source

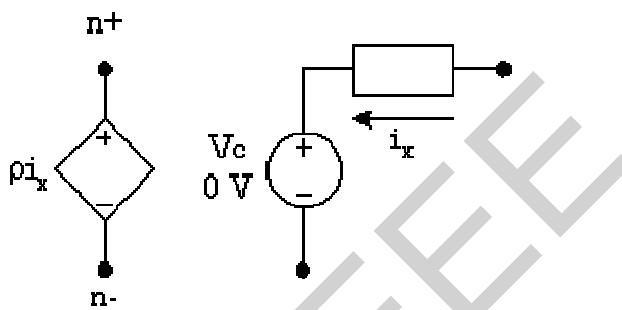


In the above voltage-controlled dependent current source a current equal to γ times v_x flows from node "n+" through the source and out node "n-." γ is called the transconductance and has the dimensions of siemens (inverse ohms). For example, if the controlling branch voltage, v_x , equals 6.0 volts and the transconductance, γ , is 0.25 siemens, the current produced by the dependent source is 1.5 amps.

The first letter of the part name for the voltage-controlled dependent current source is "G." Some examples of how this part is coded into the *.CIR file are shown below.

*Name	n+	n-	NC+	NC	transconductance
Glab	23	17	8	3	2.5
G1	12	9	1	0	4E-2
Grad	19	40	6	99	0.65
Grad	19	40	99	6	-0.65 ; same as above
Grad	40	19	99	6	0.65 ; etc.

Current Controlled Dependent Voltage Source

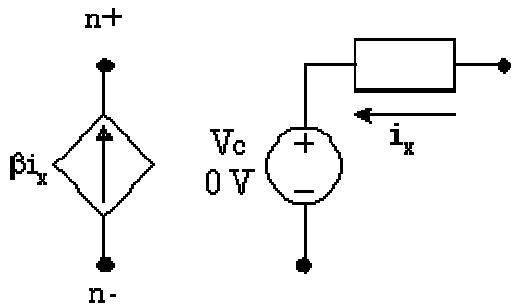


The current-controlled dependent voltage source as shown above, produces a voltage proportional to the current, i_x , in a different branch of the network. The transresistance, ρ , in ohms is multiplied by i_x in amps to produce the dependent source voltage in volts. Unlike the two previous examples, we cannot simply designate the controlling branch by its nodes. Since there could be multiple branches carrying very different currents between any pair of nodes, we must explicitly identify the branch of the controlling current. Eventually, we will be able to do this with any type of element. However, the only reliable method of doing this at present is to use an independent voltage source as an ammeter to report the current of the controlling branch to the dependent source. Usually, this means you must insert a zero-valued independent voltage source in series with the branch containing the controlling current so that the controlling current enters the positive terminal of the independent voltage source. However, if there happens to be an independent voltage source that monitors the controlling current you can use it. If necessary, use a minus sign to get the right polarity.

The first letter of the part name for the current-controlled dependent voltage source is "H." Some examples follow for this device.

*Name	n+	n-	Vmonitor	transresistance
Hvx	20	12	Vhx	50.0
Vhx	80	76	DC	0V ; controls Hvx
Hab	10	0	V20	75.0
V20	15	5	DC	0V ; controls Hab
HAL	20	99	Vuse	10.0
Vuse	3	5	DC	20V ; actual voltage source

Current Controlled Dependent Current Source



The current-controlled dependent current source produces a current proportional to the controlling current, i_x , flowing in a different branch. The current gain, β , is dimensionless. Designating the control scheme is similar to setting up the current-controlled dependent voltage source previously discussed. We must use a voltage source connected in series with the controlling element so that the controlling current enters the positive terminal of the independent voltage source used as an ammeter. If no voltage source is needed for its voltage, we use a zero-valued voltage source as shown in the figure.

The first letter in the part name for this dependent source is "F." The syntax for entering this part in *.CIR files is shown in several examples below.

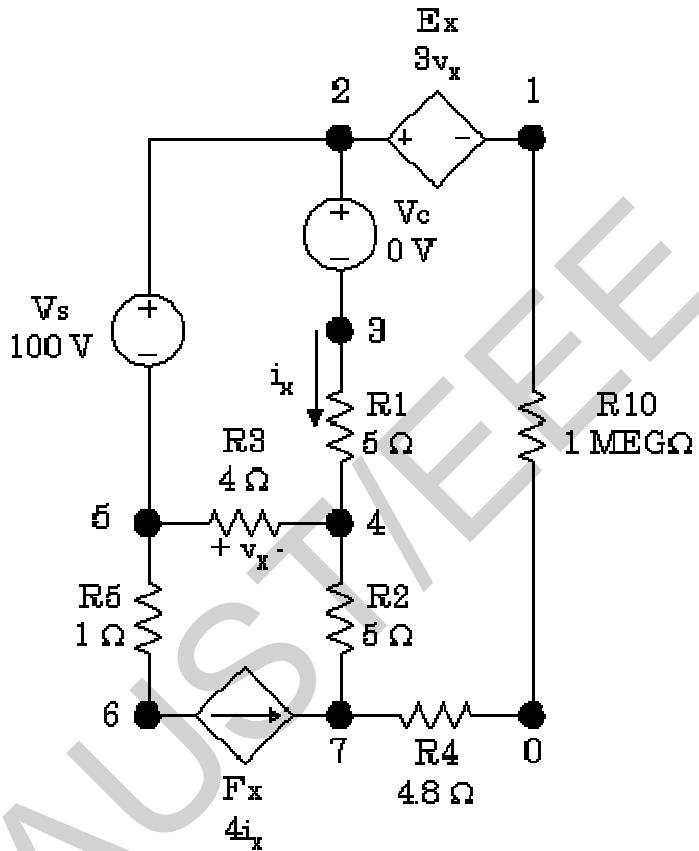
<i>*Name</i>	<i>n-</i>	<i>n+</i>	<i>Vmonitor</i>	<i>Gain</i>
Ftrn	81	19	Vctl	50.0
Vctl	23	12	DC	0V ; controls Ftrn
Fcur	63	48	Vx	20.0
Vx	33	71	DC	0V ; controls Fcur
F3	2	0	V1	15.0
V1	3	1	DC	0V ; controls F3

Using PSpice to find Thévenin Equivalent Circuit

In addition to performing general purpose circuit analysis, PSpice can be used to determine the Thévenin resistance and open circuit voltage of a circuit. This can be of great advantage if the circuit is complex, with several dependent sources, or if the circuit cannot be reduced by successive source transformations. The PSpice "dot command" that makes this easy, is ".TF," where "TF" indicates "transfer function." The transfer function is intended to find the ratio between a source voltage or current, and a resulting voltage difference or branch current. This is useful in characterizing circuits. In addition to reporting the calculated transfer function ratio and input resistance at the source, PSpice reports the *output resistance* at the terminal pair of interest. The voltage across the terminal pair of interest is the Thévenin voltage and the output resistance is the Thévenin resistance. At this point we will ignore the transfer function ratio and the input resistance at the source. In fact, we do not care which source is chosen as long as we only want the Thévenin equivalent circuit parameters. An example of the syntax for the .TF command is shown below.

```
*command output_variable input_source
.TF          V(4)           Vs
```

The above command will report the ratio between source Vs and node voltage V(4). If we wanted the Thévenin circuit from nodes 4 to 0, the output resistance reported would be our Thévenin resistance, and the voltage V(4) would be the Thévenin (open circuit) voltage. The input source can be a voltage or a current source, and the output variable can be a node voltage, branch voltage or a device current. Now we examine a specific example.



In this example, we want the Thévenin equivalent circuit from nodes 1 to 0. The 1 Megohm resistor is placed in the circuit because PSpice requires at least two connections to each node. This resistor is large enough that it will not have an effect on the calculations. Note the use of voltage source Vc which has the purpose of monitoring the control current, i_x , used for the current-controlled dependent current source, Fx. The input lines in the *.CIR file are shown below.

```
Thevenin Example No. 1
Vs    2      5      DC      100V
Vc    2      3      DC      0V; controls Fx
Fx    6      7      Vc      4.0; gain = 4
*    n+    n-    NC+    NC gain
Ex    2      1      5      4      3.0; gain = 3
R1    3      4      5.0
R2    4      7      5.0
R3    5      4      4.0
```

```

R4    7    0    4.8
R5    5    6    1.0
R10   1    0    1MEG; satisfies PSpice
*    out_var  input_source
.TF  V(1,0)  Vs
.END

```

Portions of the output file produced by this case will now be listed.

```

Thevenin Example No. 1
***** CIRCUIT DESCRIPTION
Vs 2 5 DC 100V
Vc 2 3 DC 0V; controls Fx
Fx 6 7 Vc 4.0; gain = 4.0
Ex 2 1 5 4 3.0; gain = 3.0
R1 3 4 5.0
R2 4 7 5.0
R3 5 4 4.0
R4 7 0 4.8
R5 5 6 1.0
Rab 1 0 1MEG
.TF V(1,0) Vs

```

```

Thevenin Example No. 1
***** SMALL SIGNAL BIAS SOLUTION TEMPERATURE = 27.000 DEG C
NODE VOLTAGE     NODE VOLTAGE     NODE VOLTAGE     NODE VOLTAGE
( 1) 180.0000 ( 2) -60.0010 ( 3) -60.0010 ( 4) -80.0010
( 5) -160.0000 ( 6) -176.0000 ( 7)-864.0E-06
VOLTAGE SOURCE CURRENTS
NAME CURRENT
Vs -4.000E+00
Vc 4.000E+00
TOTAL POWER DISSIPATION 4.00E+02 WATTS
***** SMALL-SIGNAL CHARACTERISTICS
V(1,0)/Vs = 1.800E+00 <== Transfer function
INPUT RESISTANCE AT Vs = 2.500E+01
OUTPUT RESISTANCE AT V(1,0) = 5.000E+00 <== Thévenin resistance
JOB CONCLUDED
TOTAL JOB TIME .01

```

We conclude that the Thévenin resistance is 5 ohms and the open circuit voltage is 180 volts. Use of the .TF function allows us to get the answers in a single job. The alternative to using the .TF function would be to run one case with a large resistor across the terminal pair of interest (if necessary) to get the open circuit voltage; and then run a second case with a zero-valued voltage source across the terminal pair to get the short circuit current. Then divide the short circuit current into the open circuit voltage to get the Thévenin resistance. We prefer the ".TF" method for obtaining Thévenin equivalent circuits.

Lesson@ 4.3

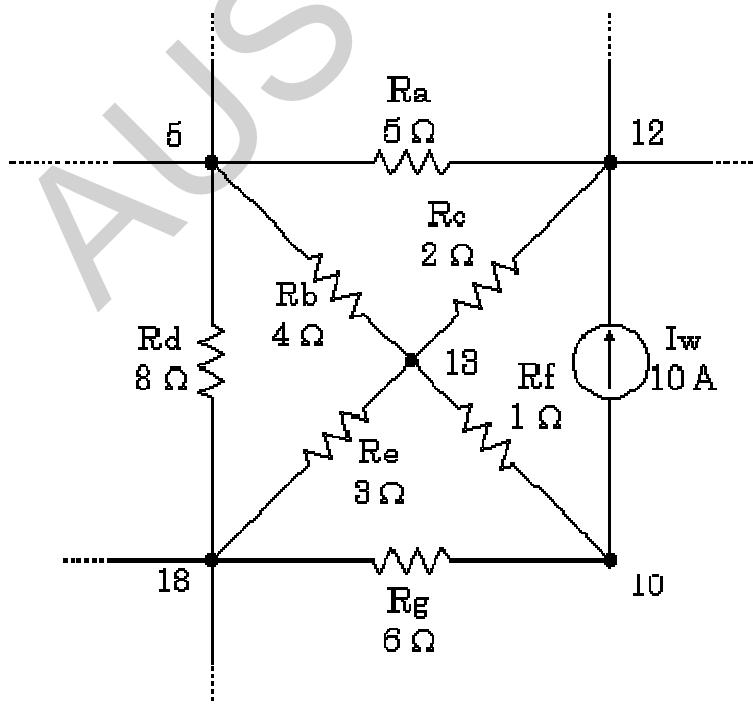
Simple Subcircuits in PSpice

One of the more useful concepts in PSpice is the use of *subcircuits* to group elements into clusters in order to replicate the clusters without having to re-enter all the elements each time. This is very useful for several reasons. First is the labor savings of replacing many lines of circuit data with a single subcircuit call. Second, the use of a subcircuit usually improves clarity by removing confusing clutter. The user can suppress printing unwanted details internal to a subcircuit, thus making the output easier to understand. If desired, the user can place often-used subcircuits into an *include* file so that the main source file for the problem is kept simple. Then the definition of the subcircuit is out of sight entirely.

Coding a Subcircuit

Each subcircuit used in a study must have a unique name. This is true of any other circuit element. Also, there must be a list of at least two nodes that can be connected to elements external to the subcircuit. A subcircuit can have many external node connections, if needed. Later, we will find that parameters can be passed to a subcircuit in order to allow unique behavior and responses from an instance of a subcircuit.

The initial line of a subcircuit section must begin with ".SUBCKT," followed by the name and then the external node list. After that, optional features (not to be discussed yet) can be added. The best method of understanding the use of a subcircuit is by example. Below, we find a cluster of components that can be combined into a subcircuit.



Note that nodes 5, 12 and 18 have external connections. Therefore, they must be included in the node list in the subcircuit definition. Nodes 10 and 13 do not have external connections and need not be

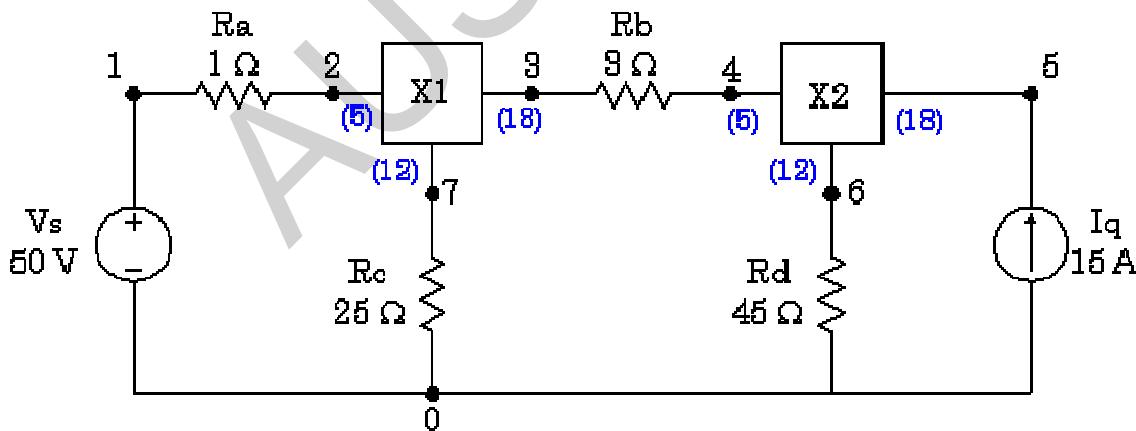
(indeed *should* not be) included in this node list. They are internal nodes and will be used to help define the subcircuit. Now, we can code the above subcircuit as follows. Note that the code could be embedded into the rest of the code for the main circuit or could be placed in a separate *include* file.

```
*      name      nodelist
.SUBCKT Example_1  5  12  18
Iw   10  12  DC  10A
Ra   5   12  5.0
Rb   5   13  4.0
Rc   12  13  2.0
Rd   5   18  8.0
Re   13  18  3.0
Rf   10  13  1.0
Rg   10  18  6.0
.ENDS
```

Note that the subcircuit section must be terminated with a ".ENDS" command.

Invoking a Subcircuit

All subcircuit calls are made by declaring a part with a unique name beginning with "X," followed by the node list and then the subcircuit name. The node list in the calling statement must have the same number of nodes as the node list in the subcircuit definition. To demonstrate the use of the calling statement, we present the following main circuit which contains two instances of the above subcircuit. X1 and X2 are the two instances of the subcircuit "Example_1." For added clarity, the subcircuit's defined external nodes are shown in parentheses. Note that these nodes are mapped into the main circuit by *different names*.



The code for the above circuit with the subcircuit included follows:

```
Subcircuit Example No. 1
*      name      nodelist
.SUBCKT Example_1  5  12  18
Iw   10  12  DC  10A
Ra   5   12  2.0
```

```

Rb      5    13    5.0
Rc     12    13    2.0
Rd      5    18    8.0
Re     13    18    3.0
Rf     10    13    1.0
Rg     10    18    6.0
.ENDS
Vs     1     0   DC   50V
Ra     1     2    1.0 ; different from Ra above
Rb     3     4    3.0 ; different from Rb above
Rc     7     0   25.0 ; different from Rc above
Rd     6     0   45.0 ; different from Rd above
*      nodelist  name
X1     2     7    3   Example_1
X2     4     6    5   Example_1
.END

```

Scope of Element Names and Nodes in a Subcircuit

Scope of names and nodes is local to a subcircuit. In the main circuit of which the above subcircuit is a part, there is a node 5 and there are resistors with the names of "Ra," "Rb," "Rc," and "Rd," and PSpice can keep these apparent duplications separated. If the subcircuit were invoked as "X1," for example, PSpice would consider the subcircuit parts as "X1.Ra," "X1.Rb" and so on. Additionally, the internal node numbers would be treated as "X1.5," "X2.5," "X2.13" and so forth. Thus PSpice maintains uniqueness of element names and node numbers.

Nesting of Subcircuits

Subcircuit *calls* may be nested as long as they are not circular. In other words, you can have a part name starting with "X" within a .SUBCKT/.ENDS block provided that the "X" part definition does not call on that block for its own definition.

However, subcircuit *definitions* may *not* be nested. I.e., you can't have one .SUBCKT/.ENDS block nested within another.

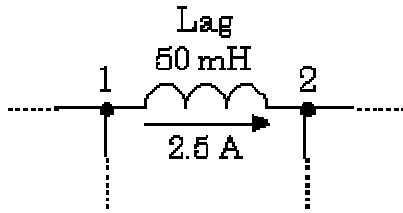
Lesson@ 4.4

Linear Inductors in PSpice

The next passive element we add to our parts list is the linear inductor. This part name begins with the letter, L, in column 1 of the source listing. Be aware that PSpice enables this part to access a nonlinear model description. That will be explained in a later tutorial. For now, our inductor model is a linear device incapable of saturation.

The inductor stores energy in its magnetic field. This makes it necessary to be able to specify its initial current in a simulation. Although we can include inductors in DC circuit simulations, there is usually little advantage in doing so because the inductor behaves as a short circuit under steady-state DC excitation. In steady-state AC simulations the inductor behaves as an imaginary impedance. We do

not specify initial current in an inductor in either of those steady-state conditions. However, when simulating transient operations, we often need to specify this initial current.



The figure shown above shows the circuit symbol for an inductor with node designations of "1" and "2," an initial current of 2.5 A, and a value of 50 mH. An appropriate code listing for entering this element into a PSpice circuit file is:

```
*name nodelist L_val
Lag 1 2 50m IC=2.5
```

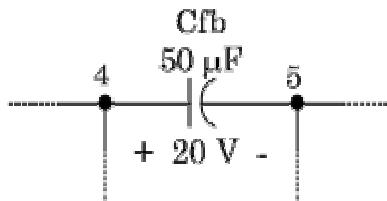
Note that the initial current is assumed to flow from the first node in the node list through the inductor towards the second node in the node list. If there is a need to change the direction of this initial current, either reverse the order of the nodes in the node list or place a minus sign in front of the value of the initial current. For better readability, the above line could be written as:

```
Lag 1 2 50mH IC=2.5A
```

The "H" for henrys and the "A" for amps will be ignored by PSpice.

Linear Capacitors in PSpice

The capacitor is the second energy storing circuit component we add to our parts list. We will assume that the capacitor is ideal in the sense of being linear and lossless. Since it can store energy, PSpice provides a method for specifying the initial voltage across the capacitor. This is useful for simulations of transient behavior of circuits with capacitors. The figure shown below illustrates a capacitor with node designations and an initial voltage of 20 V from node 4 to node 5. The part name for a capacitor must start with the letter, C.



An appropriate code listing to represent this capacitor in a PSpice listing is:

```
*name nodelist C_val
Cfb 4 5 50u IC=20
```

The capacitance of the above element is $50\mu\text{F}$. This can be represented as "50u" in PSpice. Note that the polarity of the initial voltage (as shown) is such that the positive side is the first node in the list

with the negative side on the second node in the list. To reverse the polarity of the initial voltage for the simulation, either reverse the order of the nodes in the node list or place a minus sign in front of the value in the "IC=" phrase. For better clarity, the above capacitor could be coded as:

```
Cfb 4 5 50uF IC=20V
```

PSpice would ignore the "F" for farads and the "V" for volts.

Transient Analysis Using PSpice

One of the most interesting aspects of circuit analysis is the study of natural and step responses of circuits and the responses of circuits to time-varying sources. To perform these analyses we introduce another group of "dot" commands.

Use of the .TRAN command

This is the command that passes the user's parameters for performing the transient analysis on a circuit to the PSpice program. There are four time parameters and an instruction to use the initial conditions rather than calculated bias point values for starting conditions. First, we show a sample .TRAN statement and then we will describe its parameters.

```
*      prt_stp  t_max  prt_dly  max_stp
.TRAN 20us    20ms   8ms     10us   UIC
```

In the above statement, the "20us" value labeled "**prt_stp**" (*print step*) is the frequency with which data is saved. In this case, the system variables are stored each 20 μ s of simulation time. The actual time steps used by PSpice may be different from this. The second parameter, "20ms," labeled as "**t_max**" (*final time*) is the value of time at which the simulation will be ended. Since PSpice starts at $t = 0$, there will be a total of 20ms time span of simulation for the circuit. The third parameter, "8ms," labeled as "**prt_dly**" (*print delay*) is the print delay time. In some cases, we do not want to store the data for the entire time span of the simulation. In our sample statement shown above, we ignore the data from the first 8ms of simulation and then store the data for the last 12ms. Most of the time, this parameter is set to zero or not used. The fourth parameter, "10us," labeled as "**max_stp**" (*max step*) is the maximum time step size PSpice is allowed to take during the simulation. Since PSpice automatically adjusts its time step size during the simulation, it may increase the step size to a value greater than desirable for displaying the data. When the variables are changing rapidly, PSpice shortens the step size, and when the variables change more slowly, it increases the step size. Use of this parameter is optional. The last parameter in our list is "**UIC**." It is an acronym for "**Use Initial Conditions**." Unless you include this parameter, PSpice will ignore the initial conditions you set for your inductors and capacitors and will use its own calculated bias point information instead. Note that the use of the letter "s" after the numbers in the .TRAN statement is optional. PSpice assumes these values are seconds and actually ignores the "s." However, it is recommended that you use units until you are extremely familiar with all of these commands and definitions.

Now, we will examine some more .TRAN examples.

```
.TRAN 10ns 500us
```

In the above example, PSpice will save data at each 10ns interval of the simulation starting at $t = 0$ until the final time of 500 μ s. I.e., there is no print delay and the user has given full control of the calculation step size to PSpice. In addition, PSpice will calculate its own initial conditions for any inductors and capacitors, ignoring any initial conditions set by the user.

```
.TRAN 50m 2.5 0 10m UIC
```

In the above statement, PSpice collects the data at each 50ms time interval starting from zero up to 2.5s. A zero was required as a placeholder for the print delay parameter since the maximum step size of 10ms was specified. PSpice will use the designated initial conditions of capacitor voltage and inductor current. Notice that the units were left off the numbers in this statement. Only the prefixes which size the values are needed.

Use of the .PROBE command

In addition to specifying the time parameters for a transient solution of a circuit problem, we need to specify how the data is to be saved. In most cases, this simply means that we include a line in the *.CIR file consisting of ".PROBE." This instructs PSpice to create a data file and store the data it calculates. If we create a circuit listing named "CIRCUIT1.CIR" containing a ".TRAN" statement and a ".PROBE" statement, PSpice will create a file named "CIRCUIT1.DAT" holding the data as well as the usual "CIRCUIT1.OUT" file with basic information about the circuit. By default, the data file created by PSpice is a binary data file; i.e., you can't read it with a text editor. This is the most efficient way of saving the data. However, there is an optional parameter (/CSDF) for the .PROBE statement that causes PSpice to save the data in a Common Simulation Data Format which is a text format that allows you to look at the raw data with a text editor. However, it will take up more space and PROBE doesn't load it for graphing. You will need to make a second run without the /CSDF parameter if you want to plot the data.

Also by default, .PROBE causes *all* the circuit variables to be saved, including all the variables inside each instance of each subcircuit. In some cases, this can amount to a lot of data. If you simulate a large complex circuit with many parts and need to save data at short time intervals over a very long time span, you can easily create gigabyte-size "DAT" files. To avoid this, you can specify the values you want to save. If the ".PROBE" command is issued without any parameters, everything is saved. If you specify the quantities you want saved, *only* those quantities will be saved. We will now examine some .PROBE statements.

```
.PROBE
```

All the above statement does is enable PSpice to save everything in a binary DAT file.

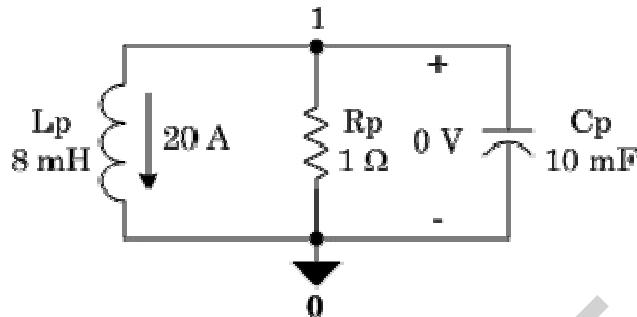
```
.PROBE/CSDF
```

The above statement enables PSpice to save everything in a CSDF file that can be opened (and edited) with a text editor. You can also read and understand the values. Unfortunately, PROBE cannot make plots from the data in this form.

```
.PROBE V(5,23) I(Rx) I(L4)
```

The above statement tells PSpice to save only the voltage drop between nodes 5 and 23, the current through resistor, Rx, and the current through inductor, L4, all in binary format. No other data will be saved.

Example of Transient Circuit Analysis



The complete listing for the "RLCNAT01.CIR" file is as follows:

```
Natural Response of a parallel RLC circuit
Rp    0    1    1.0
Lp    1    0    8mH   IC=20A
Cp    1    0    10mF  IC=0V
.TRAN 500us 100ms 0s 500us UIC
.PROBE
.END
```

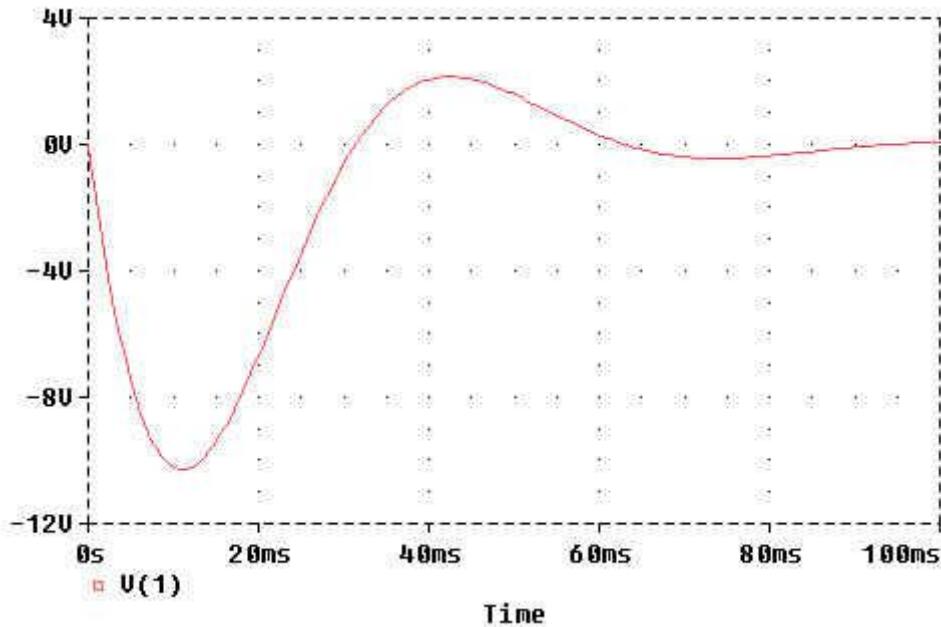
In the above example, the eight millihenry inductor, Lp, has an initial current of 20 amps flowing from node 1 through the inductor to node 0. The 10 millifarad capacitor, Cp, has an initial voltage of 0 volts. Both the print step size and the maximum step size are set to 500 μ s and the final time is 100ms.

There is no print delay, and PSpice is instructed to use the initial conditions provided. The "RLCNAT01.OUT" file is listed below. There is little information in it because the text file can show very little of the transient behavior of the circuit.

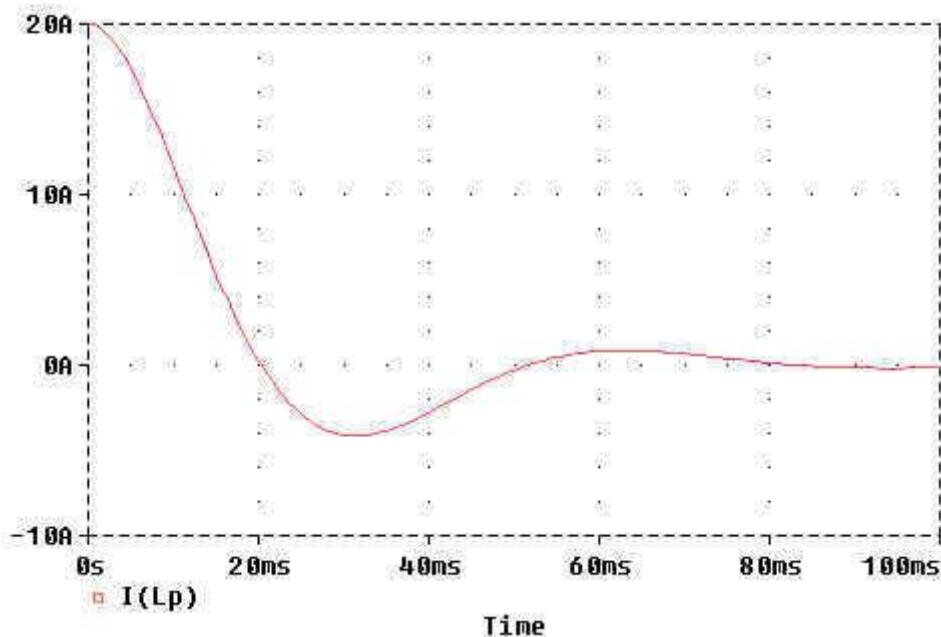
```
***** 07/17/98 18:47:40 ***** NT PSpice 8.0 (July 1997)
Natural response of a parallel RLC circuit
RP 0 1 1.0
LP 1 0 8mH IC=20A
CP 1 0 10mF IC=0V
.TRAN 500us 100ms 0s 500us UIC
.PROBE
JOB CONCLUDED
TOTAL JOB TIME .16
```

For meaningful information about the transient response we need to use another program that is bundled with PSpice. This program is named PROBE. The Probe program graphs the data that was saved in the "RLCNAT01.DAT" file. To invoke this program you left-click on "Run Probe" in the PSpice *File* menu. Probe will automatically open the DAT file you have just created. You can also launch Probe from the Start menu of Windows, but you will then need to go to Probe's File menu and open the DAT file you want to see. After you have Probe running with the proper DAT file open,

choose "Add" in the Probe *Trace* menu. You will see a list of circuit variables that can be displayed. Choose V(1), the voltage at node 1, and then click "OK." You should see the following trace in Probe.



In Probe, click on the V(1) at the lower left corner (not here, you need to be running *Probe*) and then hit the "Delete" key. Then go back to the *Trace* menu in Probe and choose "Add" again. This time choose I(LP) and click "OK." You should see the following trace of the inductor current:



Actually, you will see a negative of the above traces. In order to get the white background as you see above, you will need to modify the "INI" file for PSpice.

LESSON-9+10: PSPICE (Magnetic Circuit)

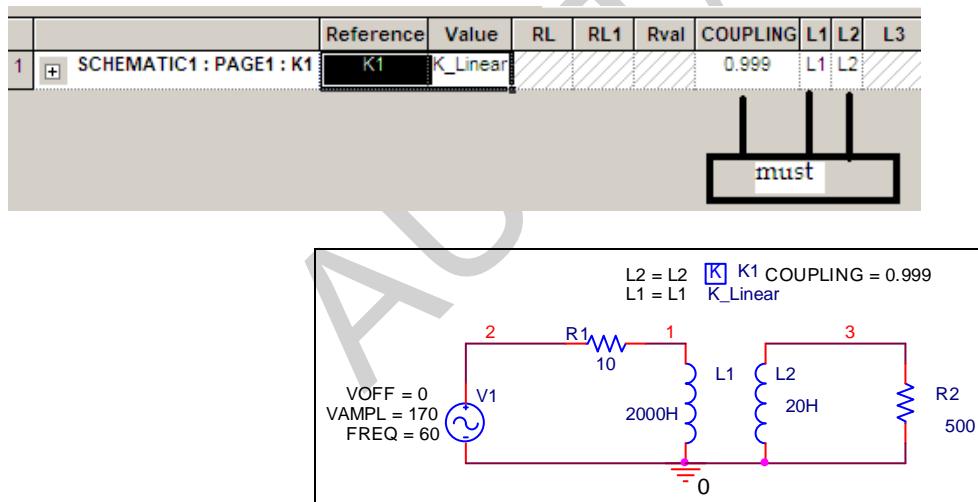
SPICE simulation of an ideal transformers.

An ideal transformer can be simulated using mutually coupled inductors. An ideal transformer has a coupling coefficient $k=1$ and very large inductances. However, Spice does not allow a coupling coefficient of $k=1$. The ideal transformer can be simulated in Spice by making k close to one, and the inductors L_1 and L_2 very large, such that ωL_1 and ωL_2 is much larger than the resistors in series with the inductors. The secondary circuit needs a DC connection to ground. This can be accomplished by adding a large resistor to ground or giving the primary and secondary circuits a common node.

The following example illustrates how to simulate a transformer.

For the above example, lets make $\omega L_2 \gg 500$ Ohm or $L_2 > 500/(60*2\pi)$; lets make L_2 at least 10 times larger, ex. $L_2=20H$. L_1 can than be found from the turn ratio: $L_1/L_2 = (N_1/N_2)^2$. For a turn ratio of 10 this makes $L_1=L_2 \times 100=2000H$. We make K close to 1 lets say 0.99999. A Spice input listing is given below for the following circuit.

For coupling purpose we must use **K_Linear**. The following datas must be provided in the properties table of **K_Linear**.



PSPICE A/D code

Example transformer

VIN 2 0 SIN(0 170 60 0 0); This defines a sinusoid of 170 V amplitude and 60Hz. RS 2 1 10

L1 1 0 2000

L2 3 0 20

K L1 L2 0.99999

RL 3 0 500

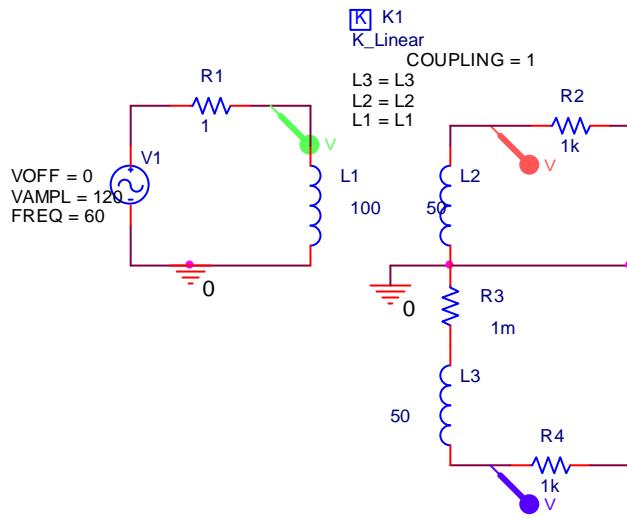
.TRAN 0.2M 25M

.PLOT TRAN V(2)

.PLOT TRAN V(3)

.END

Creating a center tapped transformer to simulate in Pspice:



Creating the schematic

1. Build a simple RL circuit energized by a VSIN. The resistor will represent the parasitic resistance and the inductor will represent the primary windings.
2. Place two more inductors in series separate from the first circuit. The two inductors will represent the secondary windings. Between L2 & L3 connect a series resistance R3 of 1m to remove problem[The problem is created by PSPICE if you not use this R3].
3. Connect two resistors to ground, one from the first secondary winding and the second resistor on the next secondary winding.
4. To complete the secondary side, ground the center tap port.
5. To finish the transformer, couple the windings. Get the part "K_linear" and place it anywhere on the schematic.
6. Double click on the K to edit the attributes. Set L1=L1, L2=L2, L3=L3 and Coupling=1.
7. Double click on the VSIN to edit its attributes. Set VOFF=0, VAMPL=100V and FREQ=60.
8. Set the resistor and inductor values where R1=1, R2=1k, R4=1k, R3=1m; L1=100, L2=50, and L3=50.
9. To view the output and input waveforms in Probe, place voltage markers on the output and input nodes.

Simulating the design

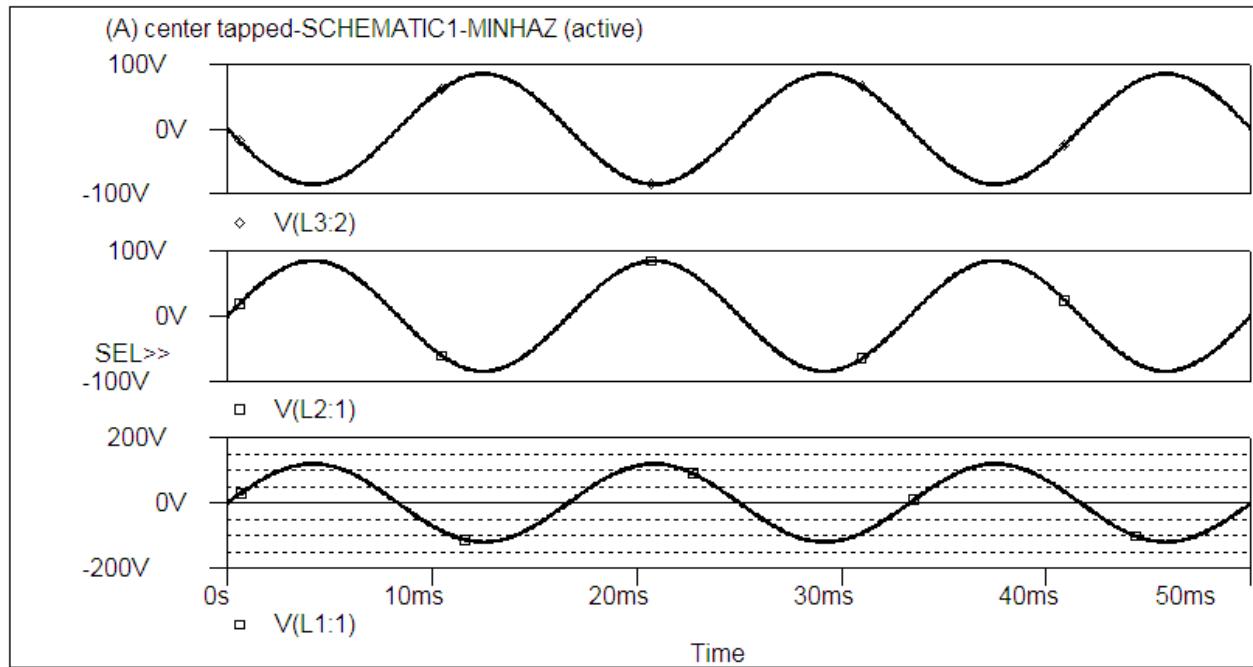
To view the output and input waveforms as a function of time, use transient analysis.

1. From the Analysis menu, choose Setup. Click on the Transient button to set up the parameters.
2. Set Print Step=.1ms, Final Time=50ms and Step Ceiling to .1ms to simulate the circuit for 3 cycles. Once the data is entered, exit by clicking OK.
3. At this point, there should be two boxes checked, Transient and Bias Point Detail. Exit the setup by clicking on Close.
4. Next, simulate by choosing Simulate from the Analysis menu or press F11.

Viewing results in Probe

When PSpice is finished simulating, Probe will automatically open with the input and output waveforms plotted. This is due to the voltage markers placed on the schematic.

- Use the cursor to identify the peak voltage by choosing Cursor then Display from the Tools menu.
- The right and left mouse buttons control the two cursor points. To change the selected plot, right-click or left-click on the plot symbol located in the lower left corner. To move the cursor, hold the right or left mouse button and scroll with the mouse.
- Instead of displaying the voltage levels, the voltage markers can be replaced with current markers to show the current waveforms. Note while voltage markers point to the node, current markers must point to the pin of the device which the current is to be marked.



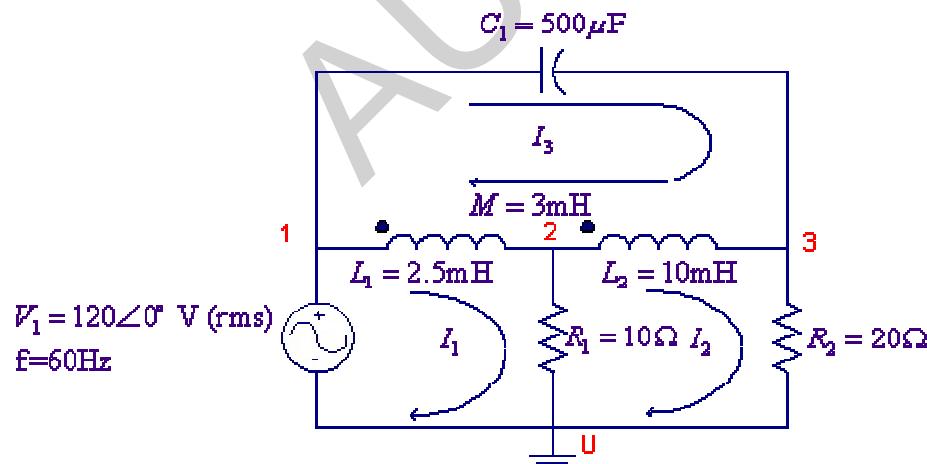
Date: January 18, 2008

Page 1

Time: 13:07:12

Mutually Coupled Circuits

Determine the magnitudes and phase angles of mesh currents in the coupled circuit shown.

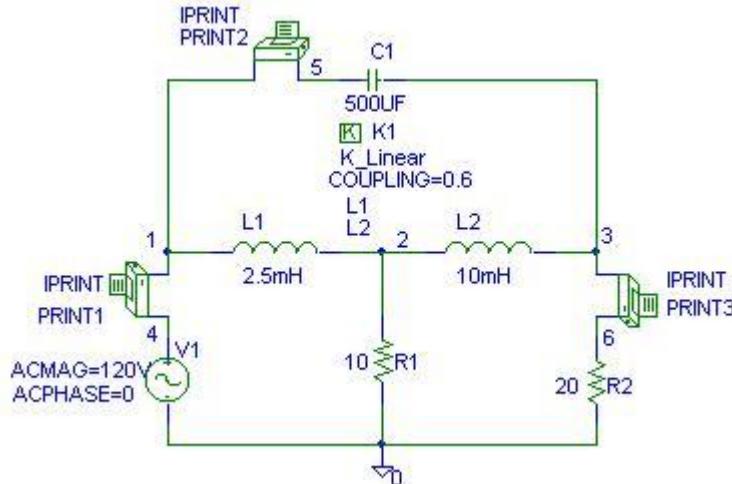


PSpice uses the coupling coefficient to describe the coupled coils, thus we find K from

$$K = \frac{M}{\sqrt{L_1 L_2}} = \frac{3}{\sqrt{(2.5)(10)}} = 0.6$$

The “dot” convention for the coupling is related to the direction in which the inductors are connected. The dot is always next to the first pin to be netlisted. When the inductor symbol, L, is taken from the part library

and is placed without rotation, the “dotted” pin is the left one. Edit/Rotate (<Ctrl R>) rotates the inductor +90deg, which makes this pin the one at the bottom. **The dotted terminal is always referred to the first node of the inductor in the Netlist. So always examine the net list and if the left node is not the dotted side, rotate the inductor in the schematic until the desired dotted node is the first entry in the Netlist.** The part **K_linear** can be used to specify the mutual coupling between two or more inductors. The parameters to be specified are L1, L2, ... up to L6, whose values must be set to the inductors symbols. The coupling value is the coefficient of mutual coupling, which must be specified between zero and 1. The PSpice schematics is as shown.



Three IPRINT symbols are inserted in series in each loop to write the currents in the output file. In the text box for each IPRINT set AC, MAG and PHASE to YES. From the analysis menu select the Probe Setup, and disable the Probe. Enable the AC Analysis, select Linear, and set the Total pts to 1, Start and End Frequencies to 60. Run PSpice (Analysis, Simulate). The Schematics Netlist is as follows

```

L_L1      1      2      2.5mH
L_L2      2      3      10mH
C_C1      5      3      500UF
R_R1      2      0      10
V_PRINT3  3      6      0V
.PRINT    AC
+ IM(V_PRINT3)
+ IP(V_PRINT3)
V_V1      4      0      DC 0V AC 120V 0
R_R2      6      0      20
Kn_K1    L_L1   L_L2   0.6
V_PRINT2  1      5      0V
.PRINT    AC
+ IM(V_PRINT2)
+ IP(V_PRINT2)
V_PRINT1  4      1      0V
.PRINT    AC
+ IM(V_PRINT1)
+ IP(V_PRINT1)

```

The output file contains the following values for the magnitude and angles of the currents

FREQ	IM(V_PRINT1)	IP(V_PRINT1)
6.000E+01	1.164E+01	3.133E+01

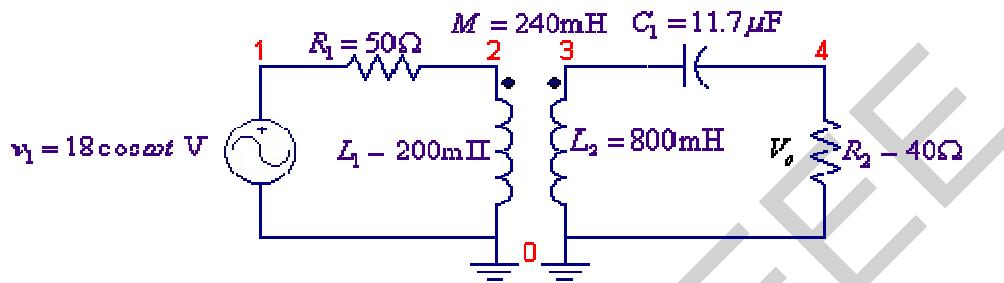
FREQ	IM(V_PRINT2)	IP(V_PRINT2)
6.000E+01	2.438E+01	5.200E+01
FREQ	IM(V_PRINT3)	IP(V_PRINT3)
6.000E+01	4.083E+00	7.719E+01

From the above results, the mesh currents are:

$$I_1 = 11.64 \angle 31.5^\circ A, I_2 = 4.083 \angle 77.19^\circ A, I_3 = 24.38 \angle 52^\circ A$$

Example 5

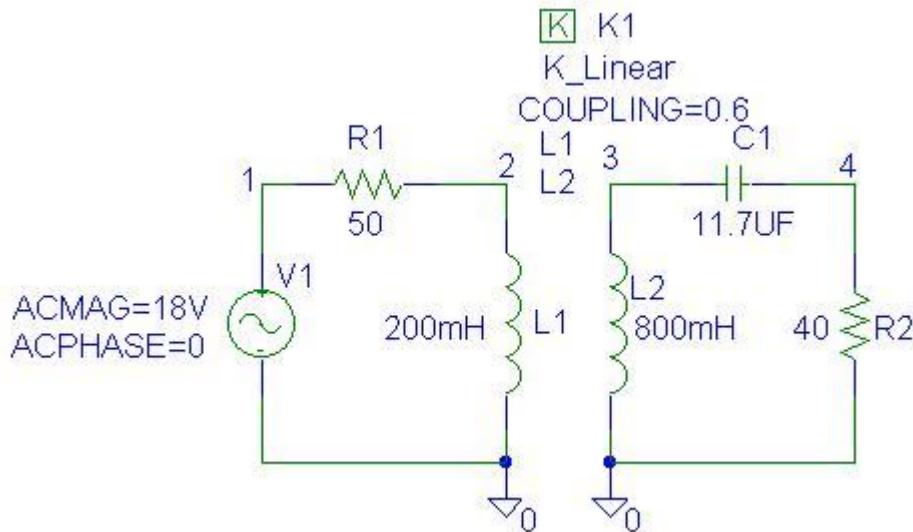
For the circuit shown, use PSpice and Probe to graph the magnitude and phase angle of the output voltage V_o , i.e., $V(4)$ as a function of frequency. Use the AC analysis to sweep the source frequency linearly from 20 HZ to 280HZ in steps of 1HZ. Determine the frequency at which the amplitude of the output voltage V_o is a maximum; find the phase angle at this frequency. Also, find the frequency at which the impedance seen by the source is purely resistive.



First we calculate the coefficient of coupling

$$K = \frac{M}{\sqrt{L_1 L_2}} = \frac{240}{\sqrt{(200)(800)}} = 0.6$$

The PSpice Schematic is as shown.



The Schematics Netlist is as follows:

```

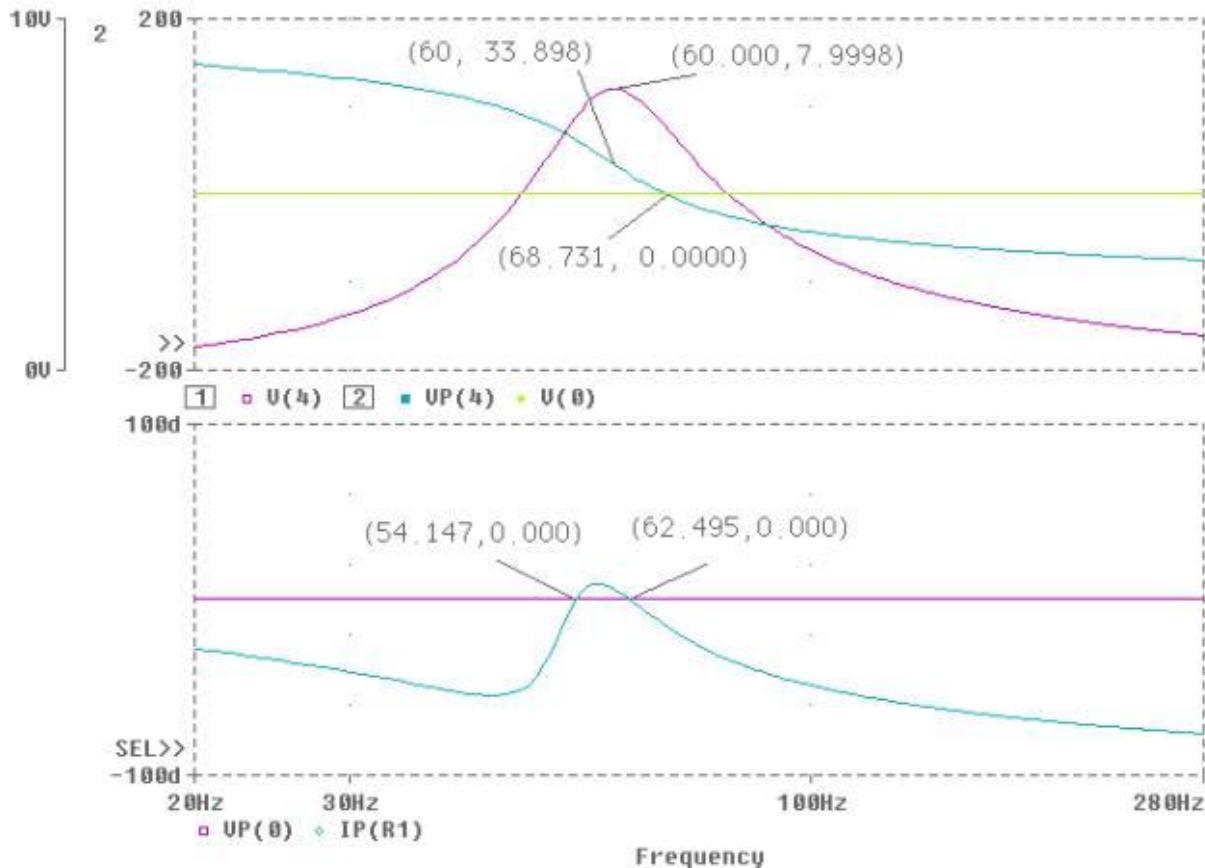
Kn_K1  L_L1  L_L2  0.6
R_R1    1      2      50
R_R2    4      0      40
V_V1    1      0      DC 0V  AC 18V 0 ; DC value 0 and AC value 18V.

```

C_C1	3	4	11.7UF
L_L1	2	0	200mH
L_L2	3	0	800mH

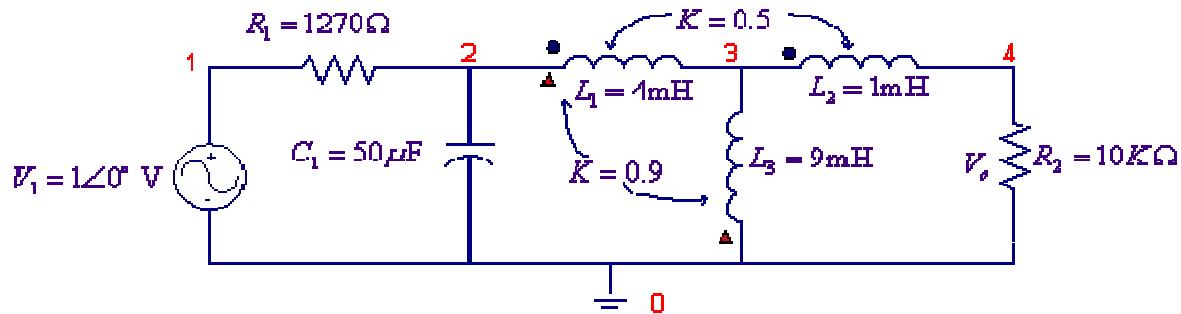
Since the dotted terminal is always the first pin in the Netlist, L1 and L2 are **rotated three times** such that their corresponding nodes are entered as 2 0, and 3 0 respectively.

In probe, Add Plot from the Plot menu to create two graphs on the screen. Using Add from the Trace menu plot V(4). From Plot use Add Y axis to create a new Y-axis, and add the trace for voltage phase angle VP(4). Select Cursor from the Tools menu, select the Display and use Peak to find the peak voltage. Use Label from the Tools menu and Mark the values at the peak position. Switch the Cursor to phase angle plot and Mark the values at the frequency corresponding to the peak value. Switch to the lower graph and use Trace to add the input voltage and the input current phase angles VP(1) and IP(R1). Use Cursor and Mark to get the frequencies at 0. The Probe result is as shown. From the graph the maximum output voltage is $V = 7.9998 < 33.898^\circ V$ at 60 Hz. From the lower graph, the input impedance is purely resistive at frequencies 54.147Hz, and 62.495 Hz.



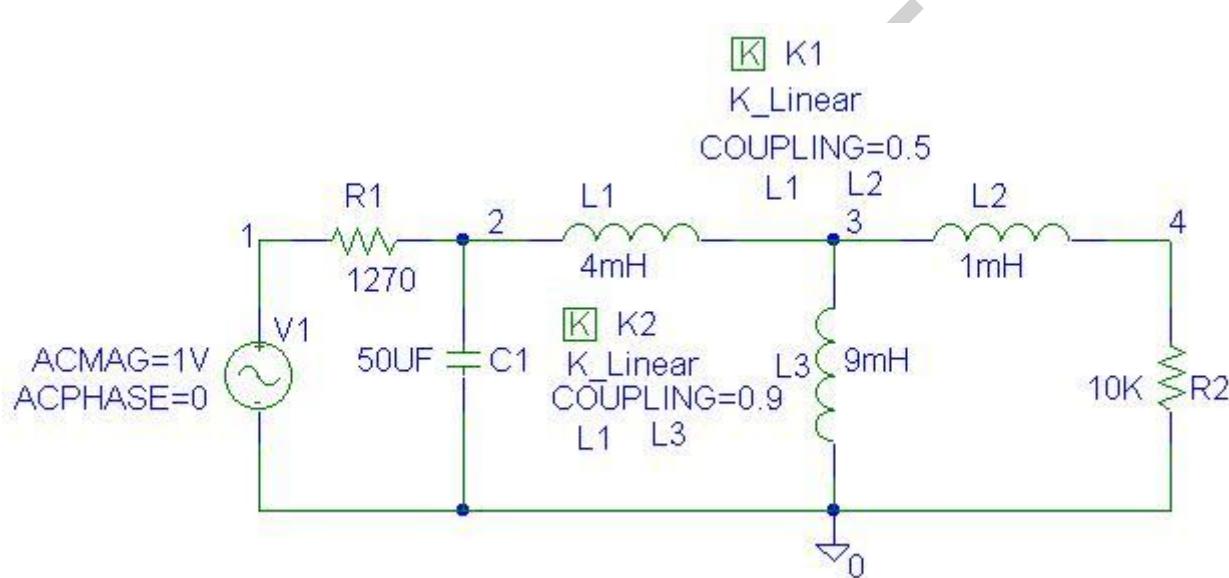
Example 6

For the circuit shown, L_1 and L_2 are mutually coupled with a coupling coefficient of $K = 0.5$. Also, L_1 and L_3 are mutually coupled with a coupling coefficient of $K = 0.9$. Use PSpice and Probe to graph the magnitude of the output voltage V_o as a function of frequency. Use the AC analysis to sweep the source frequency linearly from 450HZ to 500HZ in steps of 0.1HZ. Determine the frequency at which the amplitude of the output voltage V_o is a maximum. If bandwidth is the frequency range within 0.707 of the peak value, find the bandwidth.



Two **K_linear** parts are used to specify the mutual coupling between L_1 , L_2 , and L_1 , L_3 . Since the dotted terminal is always the first pin in the Netlist, L_3 is rotated once such that the corresponding nodes for L_1 and L_3 are entered as 2 3, and 0 3 respectively.

The PSpice Schematic is as shown.



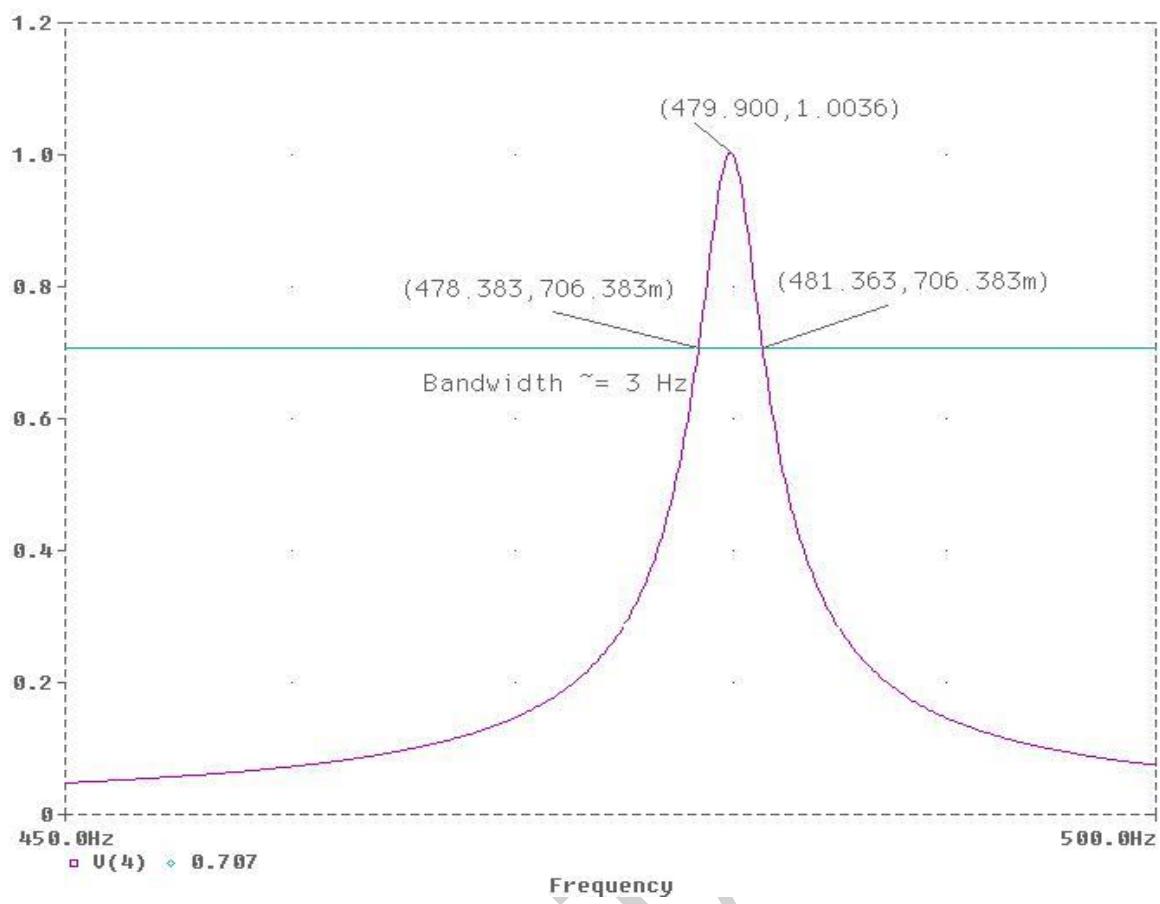
The Schematics Netlist is

```

L_L2      3      4      1mH
V_V1      1      0      DC 0V AC 1V 0
L_L1      2      3      4mH
R_R1      1      2      1270
C_C1      2      0      50UF
Kn_K1    L_L1    L_L2    0.5
R_R2      4      0      10K
Kn_K2    L_L1    L_L3    0.9
L_L3      0      3      9mH

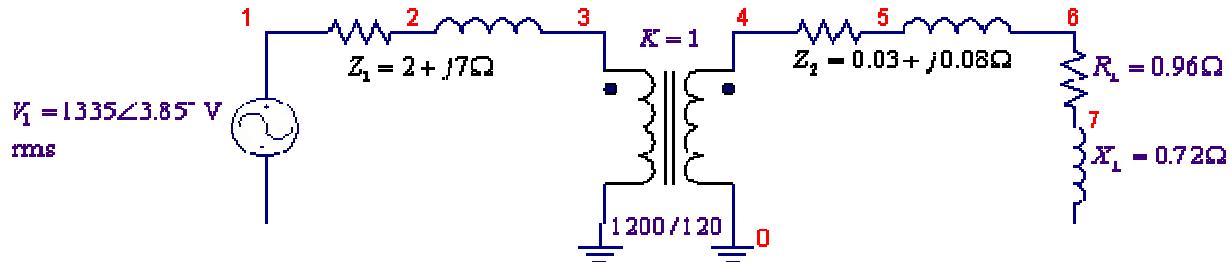
```

Use Add from the Trace menu to plot $V(4)$. From plot use the X_Axis Settings and set the range from 450 Hz to 500 Hz. Select Cursor from the Tools menu, check the Display and use Peak to find the peak voltage. Use Label from the Tools menu and Mark the values at the peak position. Add a trace at 0.707 of the peak value. Use Cursor to Mark the corner frequencies at the intersection with the 0.707 line. Determine the bandwidth and Mark it on the graph. The probe result is shown. From the graph the maximum output voltage is $V_{\text{max}} = 1.0 \text{ V}$ at 479.9Hz. The corner frequencies are $f_1 = 478.383\text{Hz}$, $f_2 = 481.363\text{Hz}$ and the bandwidth is approximately 3.0 Hz.



Example 7

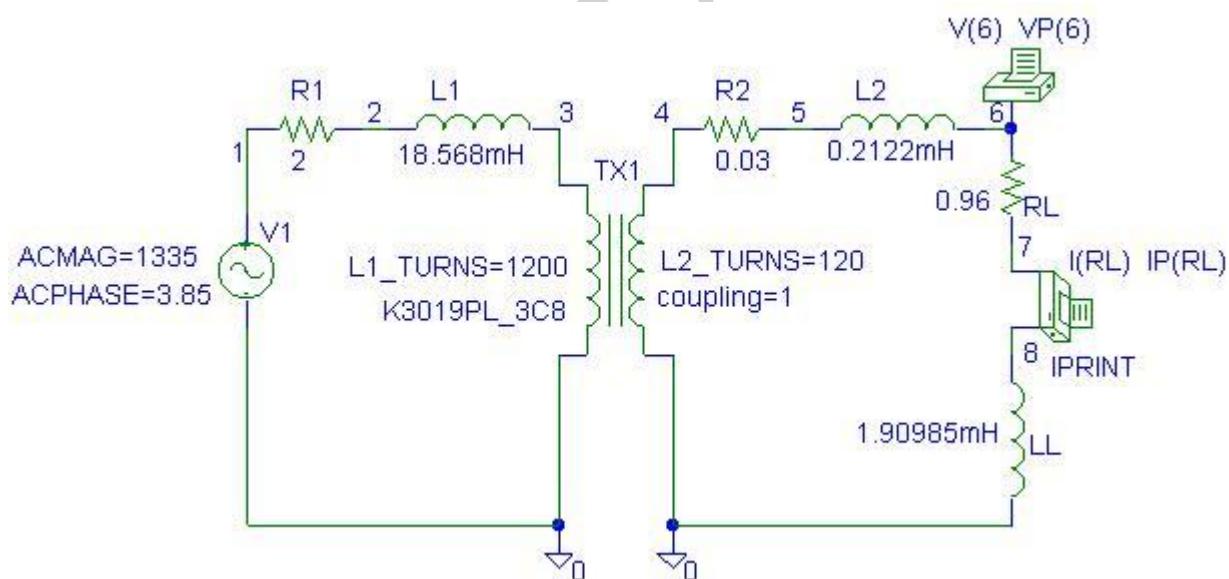
A 1200/120 V single-phase transformer has the following primary and secondary winding impedances, HV winding: $Z_1 = 2 + j7\Omega$, LV winding: $Z_2 = 0.03 + j0.08\Omega$. The voltage at the primary side of the transformer is $1335 \angle 3.85^\circ$ V (rms), 60 HZ. Transformer is supplying a load of $Z_L = 0.96 + j0.72\Omega$ at its low voltage terminal. Determine the load voltage and current.



From the given reactances at 60 HZ, the inductances are given by

$$L_{HP} = \frac{7}{2\pi(60)} = 18.568\text{mH}, L_{IV} = \frac{0.08}{2\pi(60)} = 0.2122\text{mH}, \text{ and } L_I = \frac{0.72}{2\pi(60)} = 1.90986\text{mH}$$

We can use K3019PL non-linear core to model the transformer, to model the ideal transformer the coupling coefficient is set to 1. The L1_TURNS and L2_TURNS values are set to 1200 and 120 respectively. The PSpice Schematic is as shown.



The Schematics Netlist is

```

R_R2      4      5      0.03
L_L2      5      6      0.2122mH
R_R1      1      2      2
R_RL      6      7      0.96
L1_TX1    3      0      1200
L2_TX1    4      0      120
K_TX1    L1_TX1   L2_TX1 1 K3019PL_3C8

```

```

L_L1      2      3      18.568mH
L_LL     8      0      1.90985mH
V_V1      1      0      DC 0V AC 1335 3.85
.PRINT    AC
+ VM([6])
+ VP([6])
V_PRINT2   7 8 0V
.PRINT    AC
+ IM(V_PRINT2)
+ IP(V_PRINT2)

```

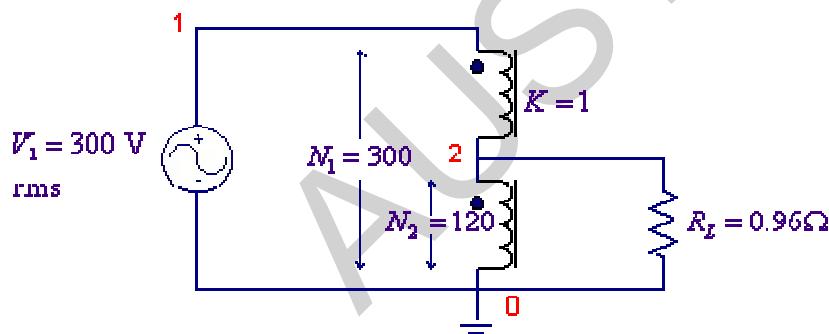
Double-click on the VPRINT1 symbol. Select 'SIMULATIONONLY=' and for value type V(6) VP(6). In the text box for VPRINT1 set AC, MAG and PHASE to YES. Also in the text box for IPRINT set AC, MAG and PHASE to YES. From the analysis menu select the Probe Setup and disable the Probe. Enable the AC Analysis, select Linear, and set the Total pts to 1, Start and End Frequencies to 60. Run PSpice (Analysis, Simulate). The output file contains the following values for the magnitude and phase angle of currents.

FREQ	VM(6)	VP(6)
6.000E+01	1.200E+02	3.730E-04
FREQ	IM(V_PRINT2)	IP(V_PRINT2)
6.000E+01	1.000E+02	-3.687E+01

That is, $V_x = 120 \angle -0^\circ$ V, and $I_x = 100 \angle -36.87^\circ$ A

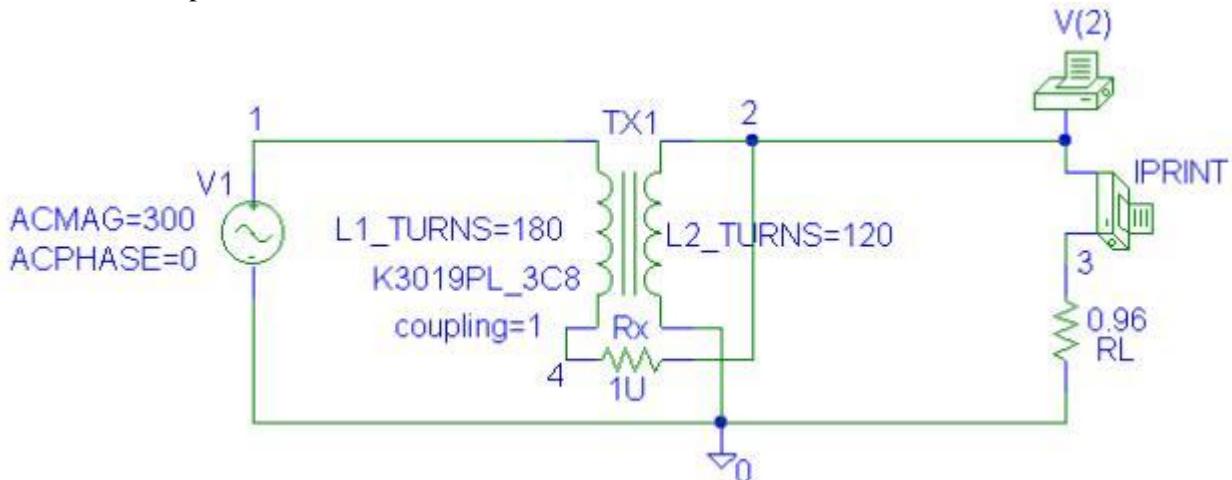
Example 8

A 300/120V ideal autotransformer is supplying a load $R_L = 0.96\Omega$ from a 300 V source. Find the secondary load current.



We can use K3019PL non-linear core to model the transformer, to model the ideal transformer the coupling coefficient is set to 1. For L1_Turns, we use $300 - 120 = 180$, and L2_Turns 120. PSpice will not allow a loop of all inductor and voltage source. To avoid this in the primary loop a negligible resistance ($R_x = 1\mu\Omega$)

is added. The PSpice Schematic is as shown.



The Schematics Netlist is

```

L1_TX1      1 4      180
L2_TX1      2 0      120
K_TX1       L1_TX1 L2_TX1 1 K3019PL_3C8
V_V1        1         0      DC 0V AC 300 0
R_RL        3 0      0.96
.PRINT      AC
+ VM([2])
V_PRINT2   2 3      0V
.PRINT      AC
+ IM(V_PRINT2)
R_RX        4         2      1U

```

Double-click on the VPRINT1 symbol. Select 'SIMULATIONONLY=' and for value type V(2). In the text box for VPRINT1 set AC and MAG to YES. Also in the text box for IPRINT set AC and MAG to YES.

From the analysis menu select the Probe Setup and disable the Probe. Enable the AC Analysis, select Linear, and set the Total pts to 1, Start and End Frequencies to 60. Run PSpice (Analysis, Simulate). The output file contains the following values:

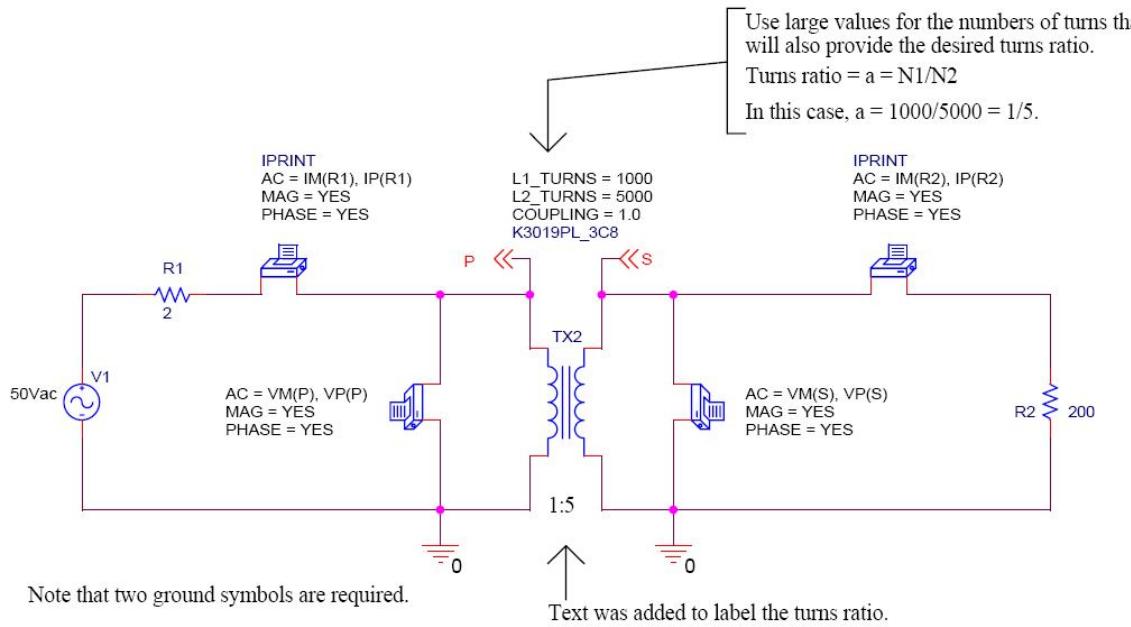
FREQ	VM(2)
6.000E+01	1.200E+02
FREQ	IM(V_PRINT2)
6.000E+01	1.250E+02

That is, $V_L = 120 \text{ V}$, and $I_L = 125 \text{ A}$.

Non-Ideal Transformer

Purpose: Determine the voltage and current for the primary and secondary of a transformer circuit using an ideal transformer.

Analysis: The source voltage is $50\cos(1000t)$, so us an AC Sweep with a single frequency of $1000/(2\pi) = 159.15 \text{ Hz}$



The non-ideal transformer (part K3019PL_3CB) is available in the EVAL library
 To change the orientation of a symbol, right-click on the symbol and then select ROTATE(or use ctrl-R)
MIRROR HORIZONTALLY, or MIRROR VERTICALLY.

For convenience, OFFPAGE symbols were used to label the primary (P) and the secondary (S).

Edit attributes of parts as follows:

- 1) If the attribute appears next to the part, double click it and then change its value
- 2) If the attribute does not appear next to the part, double click on the part, find the desired attribute, right click on it and select DISPLAY. Then indicate what Display Format is desired. Once the attribute has been displayed, double-click on it and change the value.

PROFILE:

```
***** 01/26/00 18:25:16 **** Evaluation PSpice (Mar 1999) ****
** circuit file for profile: AC Sweep
**** CIRCUIT DESCRIPTION
*****
** WARNING: THIS AUTOMATICALLY GENERATED FILE MAY BE OVERWRITTEN BY
SUBSEQUENT
PROFILES
*Libraries:
* Local Libraries :
* From [PSPICE NETLIST] section of pspiceev.ini file:
.lib nom.lib
*Analysis directives:
.AC LIN 1 159.15Hz 159.15Hz
.PROBE
.INC "transformer - non-ideal-SCHEMATIC1.net"
**** INCLUDING "transformer - non-ideal-SCHEMATIC1.net" ****
```

```

* source TRANSFORMER - NON-IDEAL
V_V1 N00023 0 DC 0Vdc AC 50Vac
R_R1 N00023 N00029 2
R_R2 0 N00065 200
V_PRINT1 N00029 P 0V
.PRINT AC
+ IM(V_PRINT1)
+ IP(V_PRINT1)
.PRINT AC
+ VM([S],[0])
+ VP([S],[0])
.PRINT AC
+ VM([P],[0])
+ VP([P],[0])
V_PRINT4 S N00065 0V
.PRINT AC
+ IM(V_PRINT4)
+ IP(V_PRINT4)
L1_TX2 P 0 1000
L2_TX2 S 0 5000
K_TX2 L1_TX2 L2_TX2 1.0 K3019PL_3C8
**** RESUMING "transformer - non-ideal-SCHEMATIC1-AC Sweep.sim.cir" ****
.INC "transformer - non-ideal-SCHEMATIC1.als"
**** INCLUDING "transformer - non-ideal-SCHEMATIC1.als" ****
.ALIASES
V_V1 V1(+=N00023 -=0 )
R_R1 R1(1=N00023 2=N00029 )
R_R2 R2(1=0 2=N00065 )
V_PRINT1 PRINT1(1=N00029 2=P )
V_PRINT4 PRINT4(1=S 2=N00065 )
L1_TX2 TX2(1=P 2=0 )
L2_TX2 TX2(3=S 4=0 )
K_TX2 TX2()
___(P=P)
___(S=S)
.ENDALIASES
**** RESUMING "transformer - non-ideal-SCHEMATIC1-AC Sweep.sim.cir" ****
.END
**** 01/26/00 18:25:16 ***** Evaluation PSpice (Mar 1999) *****
** circuit file for profile: AC Sweep
**** Ferromagnetic Core MODEL PARAMETERS
*****
K3019PL_3C8
LEVEL 2
AREA 1.38
PATH 4.52
MS 415.200000E+03
A 44.82
C .4112
K 25.74
**** 01/26/00 18:25:16 ***** Evaluation PSpice (Mar 1999) *****
** circuit file for profile: AC Sweep

```

```

***** SMALL SIGNAL BIAS SOLUTION TEMPERATURE = 27.000 DEG C
*****
NODE VOLTAGE NODE VOLTAGE NODE VOLTAGE NODE VOLTAGE
( P) 0.0000 ( S) 0.0000 (N00023) 0.0000 (N00029) 0.0000
(N00065) 0.0000
VOLTAGE SOURCE CURRENTS
NAME CURRENT
V_V1 0.000E+00
V_PRINT1 0.000E+00
V_PRINT4 0.000E+00
TOTAL POWER DISSIPATION 0.00E+00 WATTS
***** 01/26/00 18:25:16 ***** Evaluation PSpice (Mar 1999) *****
** circuit file for profile: AC Sweep
**** AC ANALYSIS TEMPERATURE = 27.000 DEG C
*****
FREQ IM(V_PRINT1)IP(V_PRINT1)
1.592E+02 5.000E+00 -3.540E-02 So IP = 5.00/-0.035□□A or IP = 5.00/0□□A
***** 01/26/00 18:25:16 ***** Evaluation PSpice (Mar 1999) *****
** circuit file for profile: AC Sweep
**** AC ANALYSIS TEMPERATURE = 27.000 DEG C
*****
FREQ VM(S,0) VP(S,0)
1.592E+02 2.000E+02 8.849E-03 So VS = 200.0/-0.0088□□V or VS = 200.0/0□□V
***** 01/26/00 18:25:16 ***** Evaluation PSpice (Mar 1999) *****
** circuit file for profile: AC Sweep
**** AC ANALYSIS TEMPERATURE = 27.000 DEG C
*****
FREQ VM(P,0) VP(P,0)
1.592E+02 4.000E+01 8.849E-03 So VP = 40.0/-0.0088□□V or VP = 40.0/0□□V

```