

VALENCIA COLLEGE

Department of Electrical & Computer Engineering Technology

ORCAD PSPICE

A Tutorial By

MASOOD EJAZ

Note: This tutorial is written specially for *CET 3464 – Software Programming in Engineering Technology*, a course offered as part of BSECET program at Valencia College.

SPICE in PSpice stands for *Simulated Program with Integrated Circuit Emphasis* and *P* shows that it can run on *personal* computers.

LITE version has some limitations, as given below, but it will work for this course.

Limitations of the Lite Version of OrCAD Products:

The lite versions of OrCAD Capture and PSpice A/D products have the following limitations with design size and complexity. If your design exceeds these limits, you will not be able to save your work or take your design through the flow.

OrCAD Capture CIS Lite

- You cannot save designs that have more than 75 nets, including the hierarchical blocks in the design. You can still view or create larger designs.
- You cannot save a design with more than 60 parts, including the hierarchical blocks in the design. You can still view or create larger designs.
- You cannot have more than 1000 parts in the Capture CIS database.
- The Internet Component Assistant (ICA) tab in the CIS Explorer window opens the About Active Parts page (www.activeparts.com) and not the component search page.
- You cannot create parts with more than 100 pins.
- The Capture FPGA flow is not available.

PSpice A/D Lite

- Circuit simulation limited to circuits with up to 75 nodes, 20 transistors, no sub-circuit limits but 65 digital primitive devices, and 10 transmission lines (ideal or non-ideal) with not more than four pairwise coupled lines.
- Device characterization and parameterized part creation using the PSpice Model Editor limited to diodes.
- No limit to stimulus generation using Stimulus Editor.
- Sample model library named eval.lib (containing analog and digital parts) and evalp.lib (containing parameterized parts) are provided.
- The library nomd.lib is configured for simulations. The nomd.lib file references the set of libraries that can be used with the lite version.
- You cannot simulate parameterized parts that are not from the evalp.lib library. This library consists of parameterized resistor, source, and diode.
- You cannot use Level 3 of Core model (Tabrizi), MOSFET BSIM 3.2, or MOSFET BSIM 4 models.
- Displays only simulation data created using the lite version of the simulator.
- Magnetic Parts Editor allows you to design power transformers only. The database shipped with Magnetic Parts Editor cannot be edited and contains a single core.

- The Model Import Wizard supports parts and simulation models that have a maximum of two pins or two terminals, respectively.
- The maximum nodes in a digital circuit can be equal to or less than 250.
- The non-ideal *Tline* is limited to 4.

ELECTRICAL CIRCUITS

STARTUP

To simulate a circuit in PSPICE, first we will have to design it in the circuit editor. To start circuit editor, run *ORCAD CAPTURE/Capture CIS Lite* file from *Cadence* folder; do not run PSpice file. You will see the following ORCAD Capture opening screen:

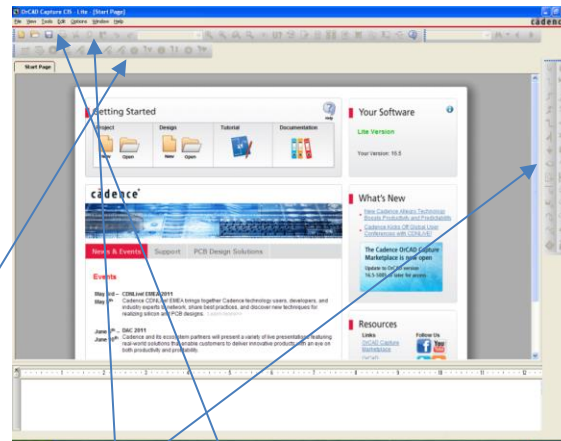


Figure 1: ORCAD Capture Opening Screen (Version 16.5)

Screen layout is very basic with *toolbar* and *file menu* on top of the screen. Under toolbar there is *PSPICE simulation menu* that will be active once you will have a circuit design. On the right hand side there is *tool column* that you will need to design a circuit. There are three steps to simulate a circuit,

- (i) Draw or design the circuit,
- (ii) Simulate it,
- (iii) Plot or print the results.

CIRCUIT DRAWING

Either from the main screen or from the file menu, create a *New Project*. Choose **PSpice Analog or Mixed A/D**, give a name to your project and select the location where you want to save it.

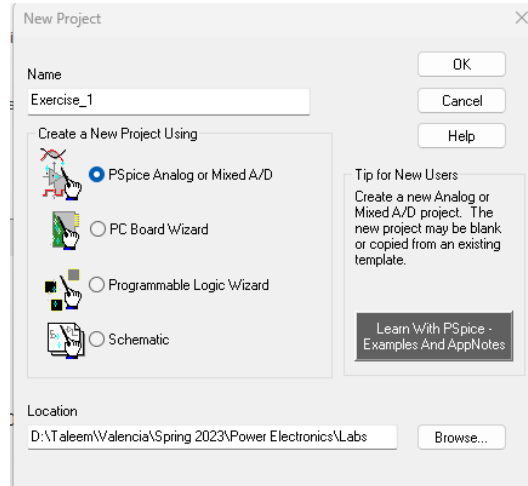


Figure 2: New Project Dialog Box

Notes:

(i) By default, under the *New Project* window, *Schematic* option will be selected. Make sure to choose *PSPice Analog or Mixed A/D*, as shown in figure 2. If you leave the *Schematic* option selected, you will only be able to create a circuit but will not be able to simulate it.

(ii) When you assign a name to your file, do not start with a number or a special character. Always start with an alphabet. Additionally, do not use any special character except underscore (_) in the filename. Filename must not have any spaces either.

After selecting *PSPice A/D*, assigning a proper filename, and file location in the *New Project* window, click *OK*. The following window will open up:

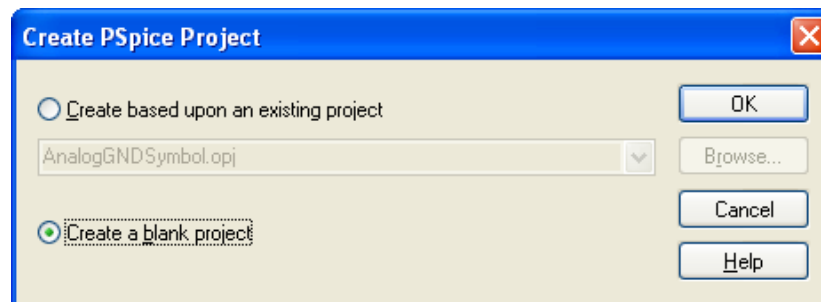


Figure 3: Options to Create PSPICE Project

Choose **Create a blank project** and click **OK**. This will create a project file (.opj) and a design file (.DSN) under the desired folder with the name that you chose. Once a blank project window is opened, you will see your schematic page to draw a circuit and some more choices will be added to the file menu. *PSICE simulation menu toolbar* and *design toolbar* will also be active now. The items located under *Design toolbar* on the right-hand side can also be found and selected under the *PLACE* menu.

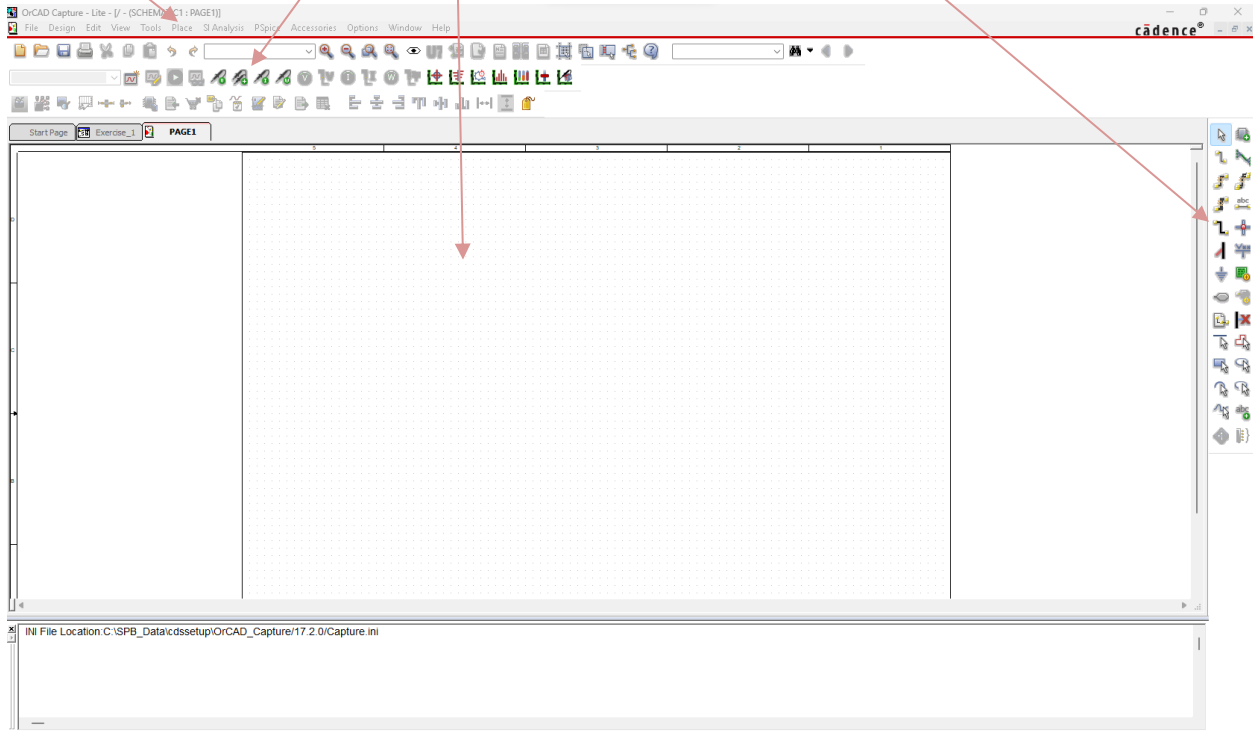


Figure 4: New PSPICE Project Window

Let's start with a circuit drawing. Suppose we want to draw the circuit shown in *figure 5*.

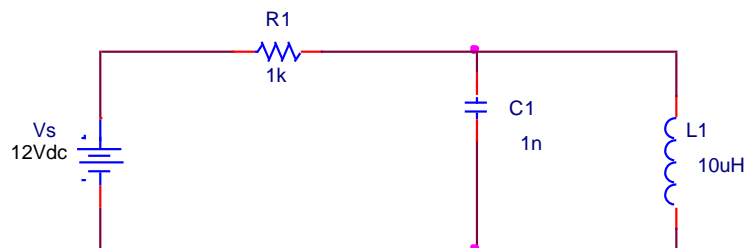


Figure 5: A Basic RLC Circuit

To start drawing this circuit, go to the file menu, *PLACE* → *PART*, you will see *Place Part* window will appear on the right side of the screen. To place any part, first you have to include appropriate libraries where those parts are located. To draw the circuit from *figure 5*, you will need *Source* library (*source.olb*) and *Analog* library (*analog.olb*). Add these two libraries by clicking *Add Library* button located next to 'X' under *Libraries*.

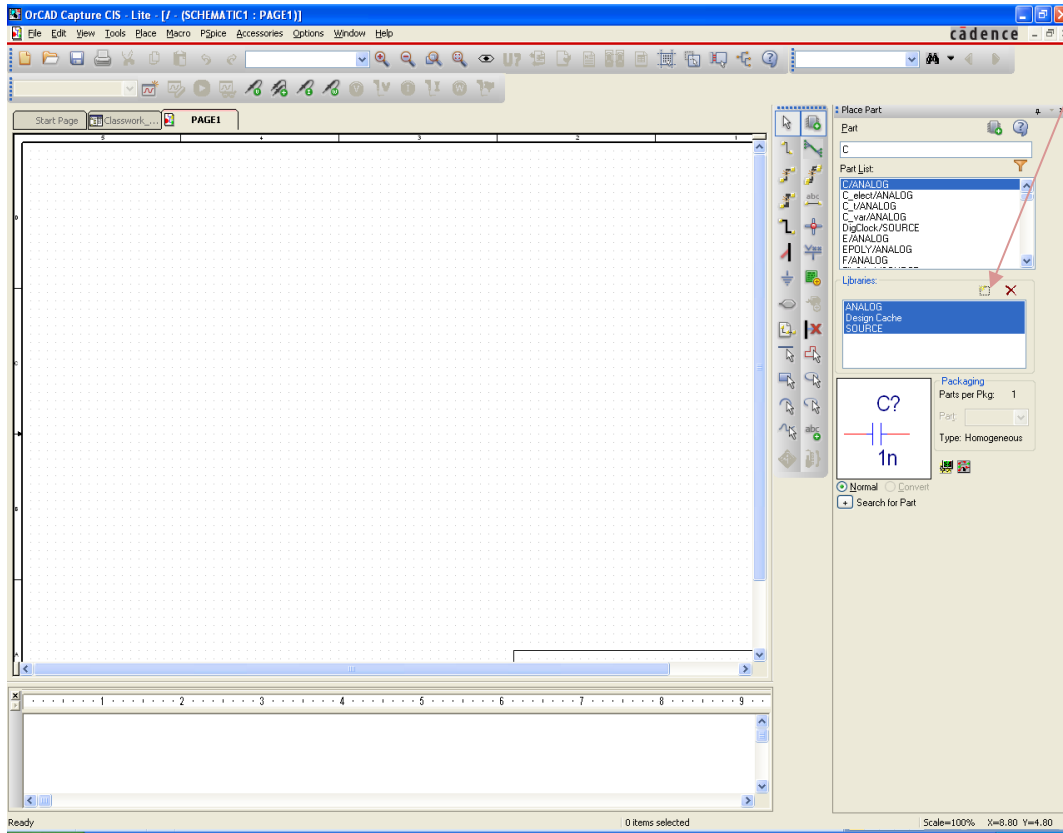


Figure 6: Project Window with Place Part Window

Analog library contains analog components including *resistor*, *inductor*, and *capacitor* that you need for your design from *figure 5*. Choose *R* and place it on the design schematic. To proceed with the other parts, right click your mouse and choose *End Mode* to stop putting any more resistors or simply press *Escape* (*Esc*) on the keyboard. Next, choose *C* and *L* and put them on the schematic too. You can select any component by left click of your mouse and then perform several operations by right clicking it. Right click on *C* and *L* and rotate them so they will be vertical as shown in *figure 5*. Next, from *Source* library choose *V_{dc}*. Now you have all the components that you require for *figure 5* on the design schematic.

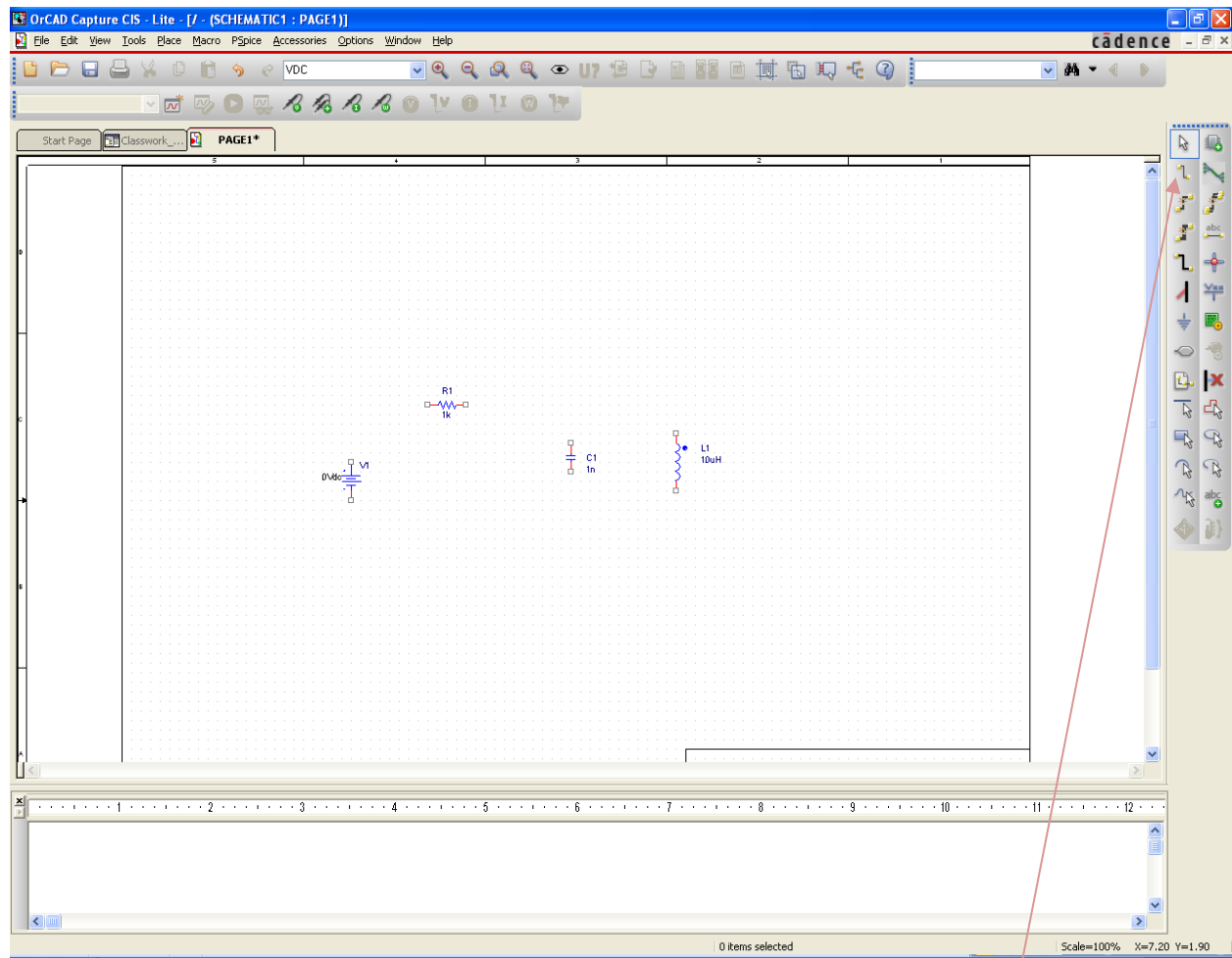


Figure 7: Full Schematic Window

To connect all the components, choose *wire* from *Place* menu or choose *wire* button from the right panel (second one from the top). Click left mouse button to select one of the terminals of a component to start wiring and move your mouse to one of the terminals of any other component and left click it again. Once you are done wiring all the components, right click your mouse and choose *End Wire* (or simply press *Esc*). Your circuit should look like the one shown in figure 8.

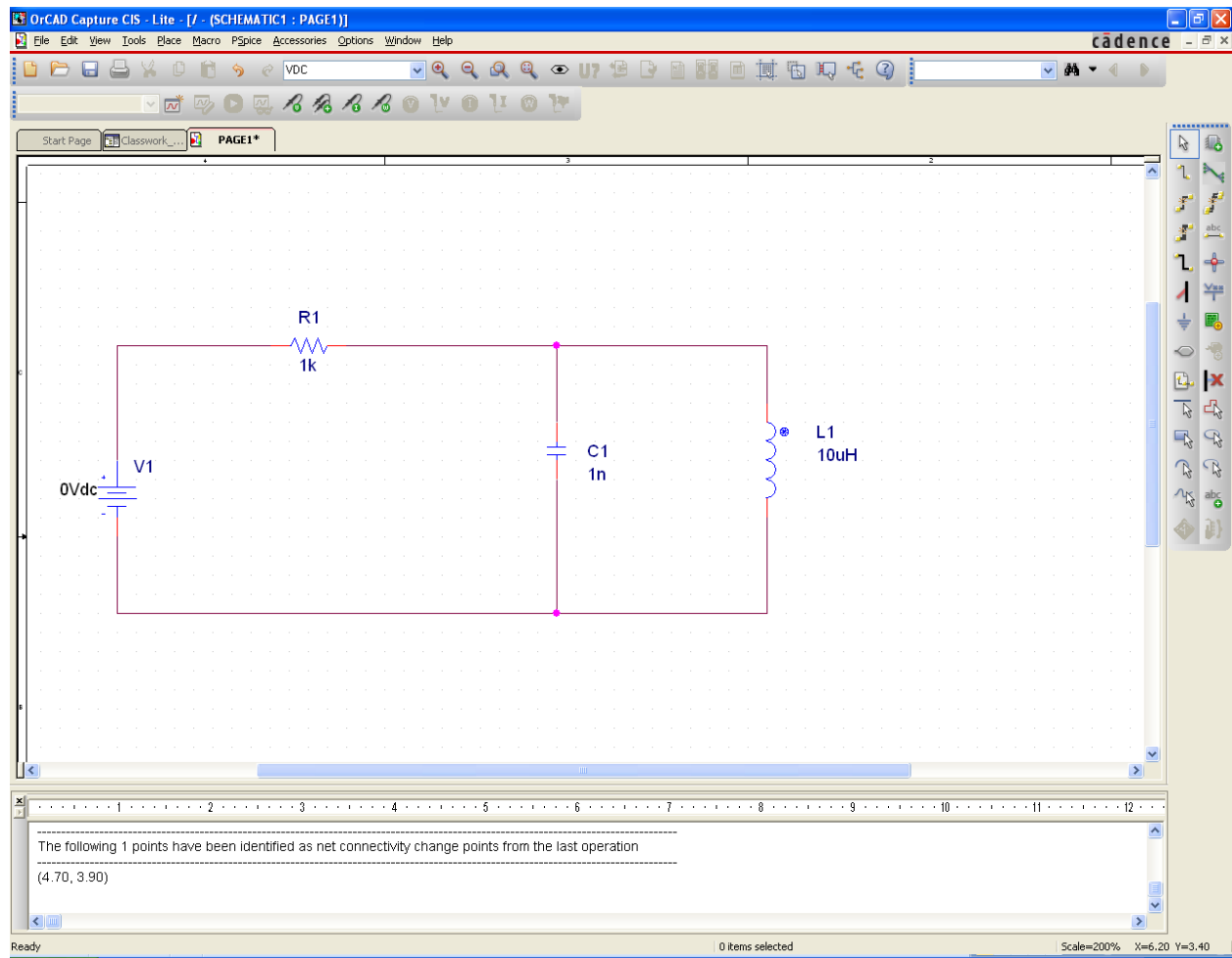


Figure 8: Connected Circuit from Figure 5

You can move around label and value of any component on your schematic by holding it with the left mouse button and dragging it. To change the value of any component, double click on the value and change it from the dialog box.

Note: Make sure not to have any spaces in any of the variable's name. Likewise, do not put any spaces between a component's value, its prefix, and its unit.

To change the value of the *DC voltage source* shown in figure 8 to 12V, double click on '0Vdc' to open the following dialog box:

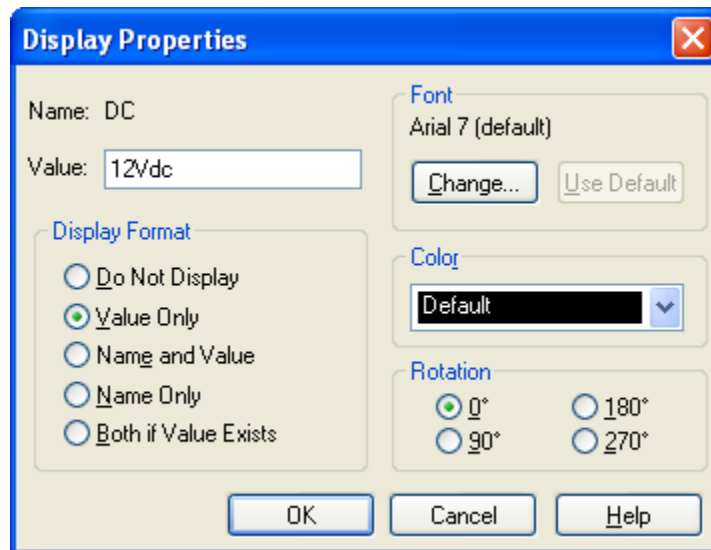


Figure 9: Display Properties' Dialog Box for the Voltage Source

You can see different options in this dialog box. Change the value to '12Vdc' (You can also just write 12 or 12V, both will work). The final thing that you have to do is to add a **ground** to your circuit in order to simulate it properly. *Ground* represents the reference voltage point in the circuit with respect to which any other *node* voltage is measured. To select ground, either go to the *Place* menu and select *Ground* or use *Ground* button from the right panel. Choose the default ground that appears (0/CAPSYM) and make sure that you have '0' under *Name*.

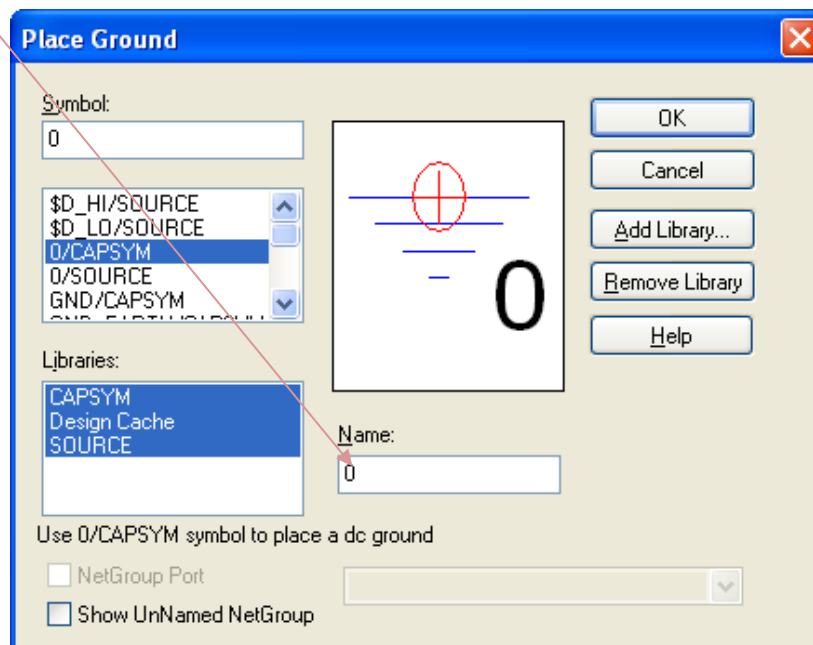


Figure 10: Properties of Ground

Connect ground to your circuit and you have your first complete circuit that is ready to be simulated.

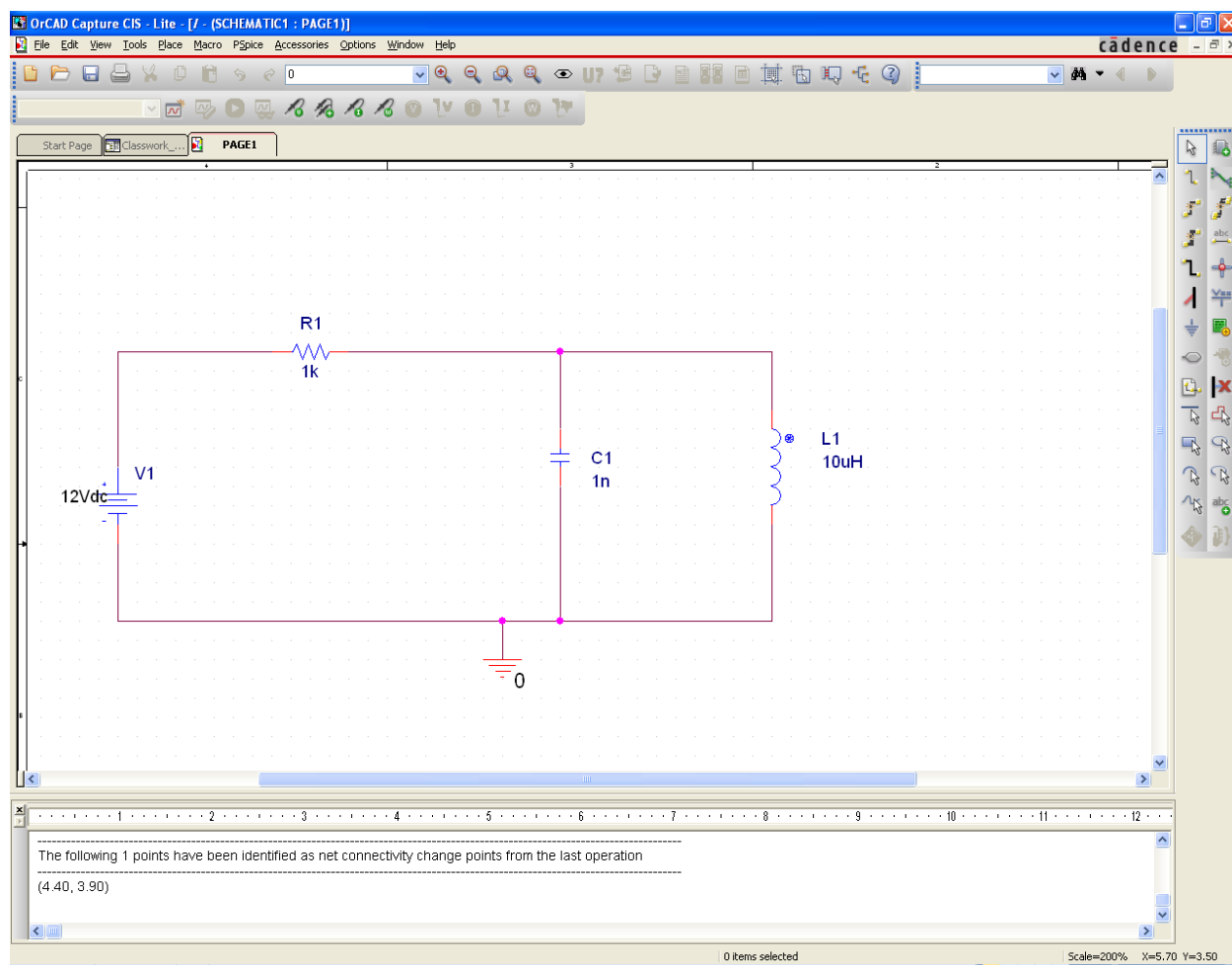


Figure 11: Complete Circuit

NOTE: You can add more design schematics to the same project by right clicking on the design name and choosing *New Schematic*. This will be useful if you are working on your homework or exam and you have two or three different circuits; you can create all different circuits on different schematics under the same project. Make sure that if you have multiple schematics under the same project, the one that you are simulating has to be the *Root Schematic*. To change any schematic to root, right click on its name and select *Make Root*. Also, note that you have to open a *new page* under any new simulation that you create to design your circuit. Do not create multiple pages under the same simulation as it will end up running all pages under that simulation once you run that simulation, and there will be more chances of encountering different errors.

EXERCISE- 1

Create the resistive circuit shown in *figure 12*. Change component values as shown in the circuit. Save your circuit after you are done designing it.

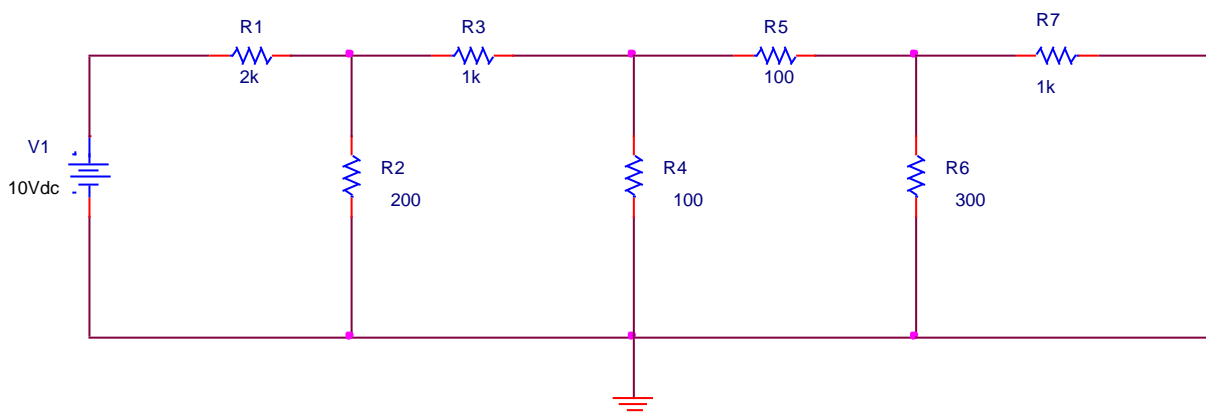


Figure 12: Resistive Circuit for Exercise – 1

In engineering, most of the values are represented in engineering units and corresponding prefixes. Likewise, in circuit analysis, components have their values expressed in engineering units using prefixes. *Table 1* shows the prefixes that you can use for different engineering notations.

Table 1: Prefixes for Electrical Components and Quantities used in PSPICE

Prefix Letter	Metric Prefix	Multiplying Factor
T or t	Tera	10^{12}
G or g	Giga	10^9
Meg, Mega, meg, or mega	Mega	10^6
K or k	Kilo	10^3
M or m	Milli	10^{-3}
U or u	Micro	10^{-6}
N or n	Nano	10^{-9}
P or p	Pico	10^{-12}
F or f	Femto	10^{-15}

NOTE: Be very careful with *mega* and *milli*. Do not use lowercase ‘m’ for *milli* and uppercase ‘M’ for *mega*. Both uppercase and lowercase M are used for *milli*. For *mega*, use *Mega/mega* or *Meg/meg*.

Circuit Simulation

Before we start simulating a circuit, let's revisit a common mistake that many students make when they are creating projects, which is mentioned on *page 4*. Since *Schematic* is the default option under *New Project* window, many students do not pay attention and create a new project with *Schematic* instead of *PSpice Analog* or *Mixed A/D*. Under *Schematic*, you can only create schematic or design of a circuit, but you cannot simulate it. **Your simulation buttons will not be highlighted if you create a project using *Schematic* option.** If your simulation buttons are not highlighted, copy your circuit, open a new project and choose *Analog* or *Mixed A/D*, and paste your circuit in the schematic window. You should have the simulation buttons highlighted now.

The circuit that you created in *Exercise 1* can be simulated to get *DC Bias Points* of the circuit, i.e., *DC (Direct Current)* voltage at each node, current flowing through each branch, and power dissipated in each component. This is one of the four different types of simulations that will be done in PSpice. From the PSpice simulation menu, choose *New Simulation Profile* (first button on the left under the simulation menu). A *New simulation* window will pop up, as shown in the following figure:

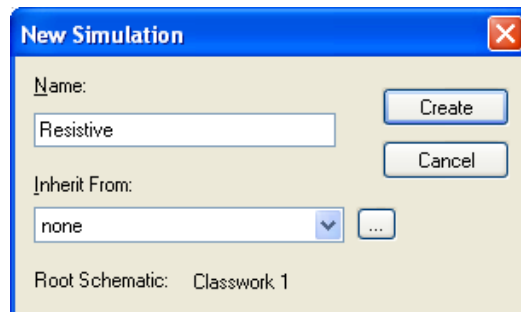


Figure 13: New Simulation Dialog Box

Choose a name for your simulation (no spaces in the name, and no special characters except underscore) and press *Create*. *Simulation Settings* dialog box will pop up. Under *Analysis Type* choose *Bias Point* and click OK.

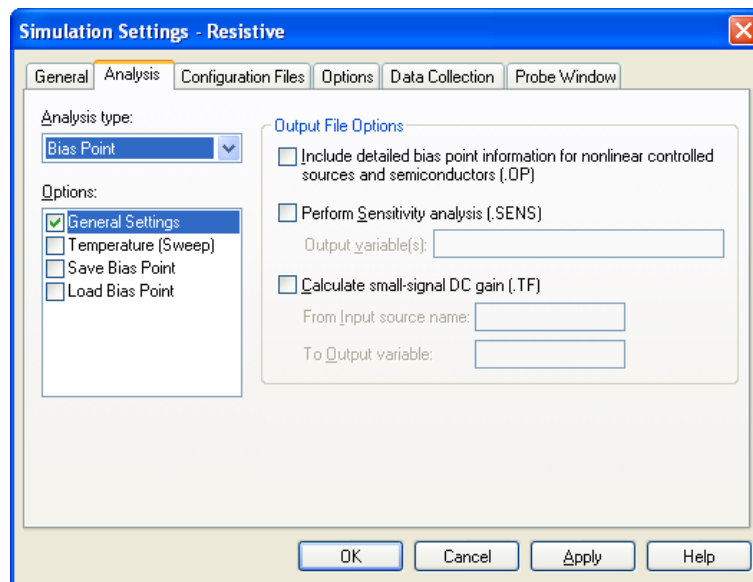


Figure 14: Simulation Setting Dialog Box

Now run your simulation from *PSPICE* menu using *RUN* (▶). PSpice A/D window will open up and will show you different messages about the simulation (errors or not, completed etc.) as shown in figure 15.

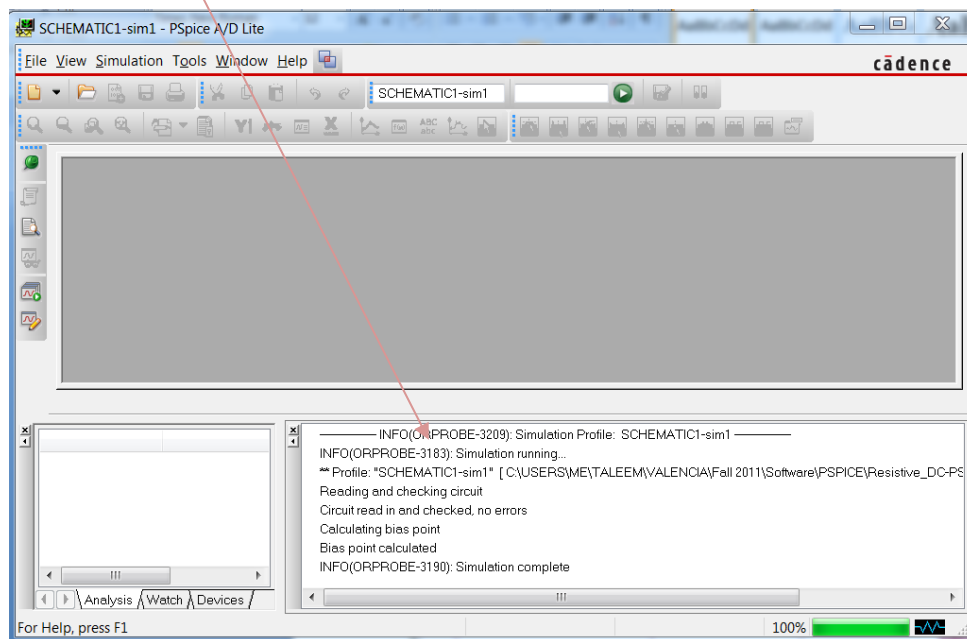


Figure 15: Simulation Window

When you are performing *Bias Point* simulation, simulation data do not appear in the simulation window, rather, bias point values appear right on the circuit. You can go back to your schematic to see the node voltages, branch currents, and power in different components by pressing *voltmeter* (**V**), *ammeter* (**I**), and *wattmeter* (**W**) buttons. You can also open simulation output file (.out) in the PSpice A/D window to check if there are any errors and other details. Your circuit should have node voltages and branch currents (if you select **V** and **I**, respectively) as shown in the following figure:

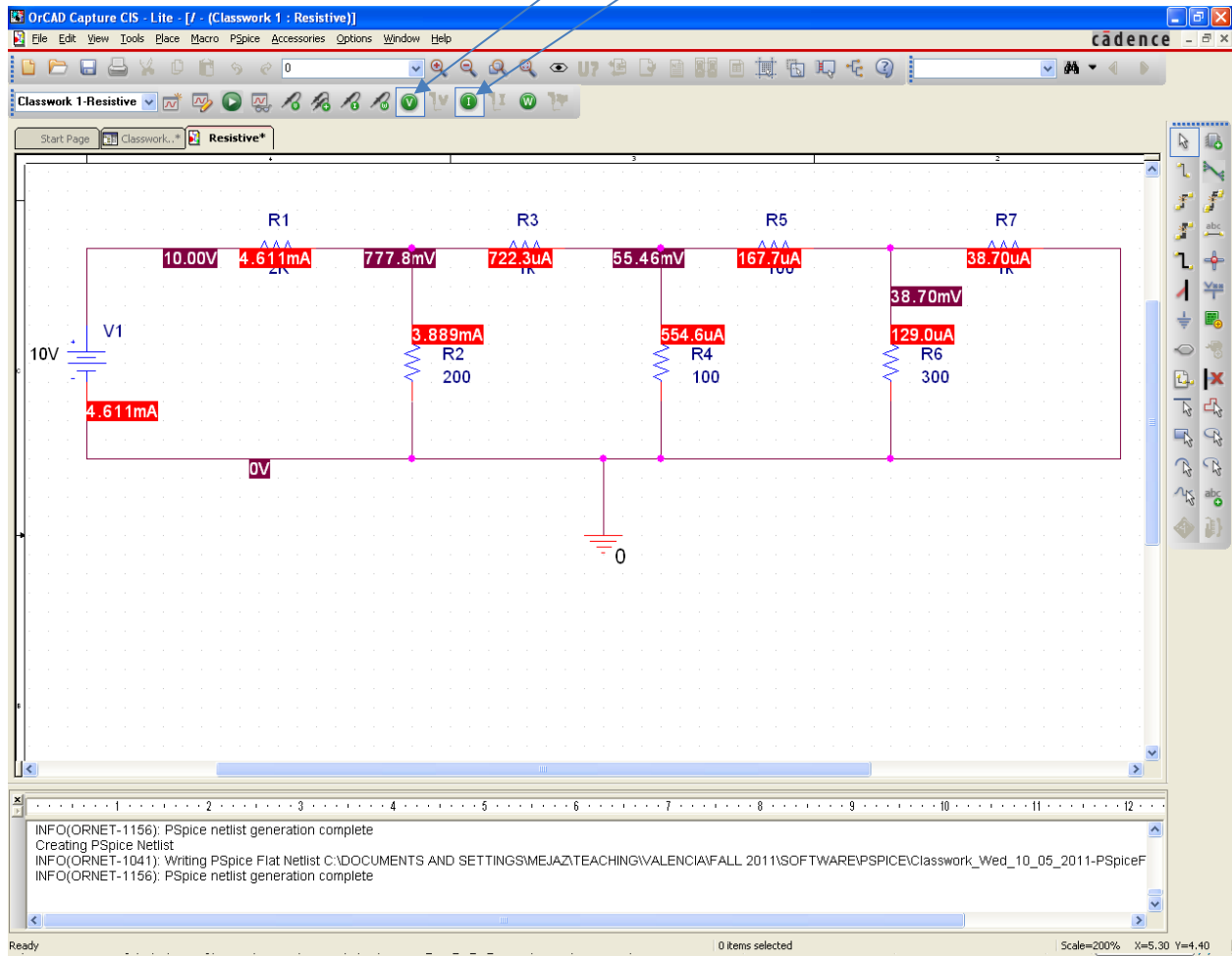


Figure 16: Node Voltages and Branch Currents for Exercise – 1 Circuit

EXERCISE – 2

Create the *RLC* circuit shown below and carry out the *DC Bias Point* simulation for different voltages, currents, and power dissipations.

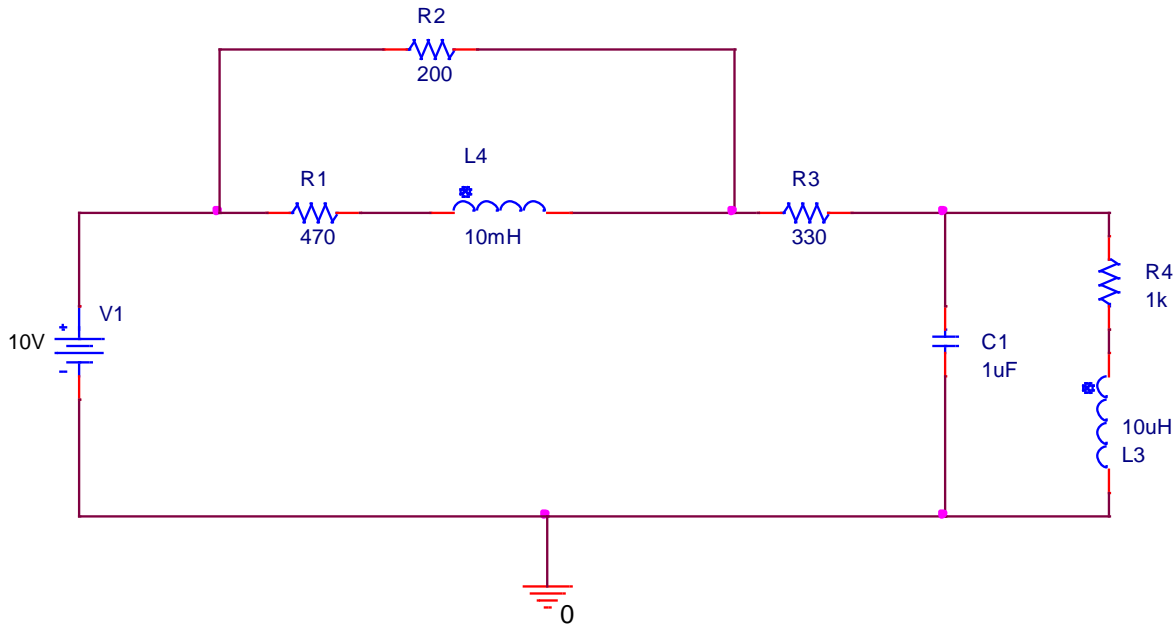


Figure 17: RLC Circuit for Exercise 2

NOTE: You can check the description of your simulated results from the *output* file created during the simulation, as mentioned earlier. To open the output file, go to the PSpice A/D window and press the *View Simulation Output File* button from the left vertical menu bar (third icon from the top) or from the *view* menu. The output file also helps with the description of errors in the simulation, if there are some.

DC Sweep Analysis

The second type of analysis, *DC sweep*, refers to the analysis of different voltages and currents when you are changing the value of your power supply over a range of values. Suppose we want to evaluate voltages at different nodes and current through the circuit shown in *figure 18* when we increase the input voltage from 5V to 10V gradually in steps of 0.5V

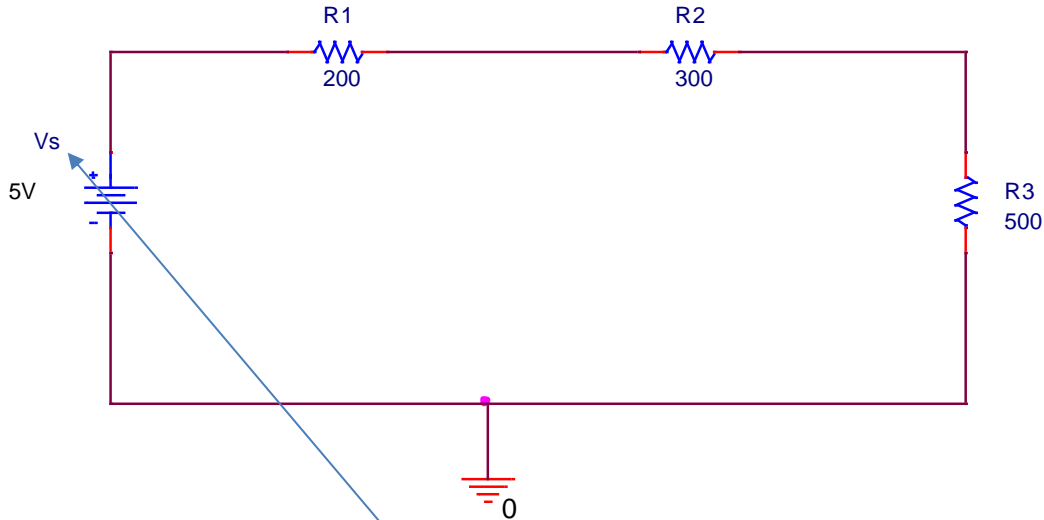


Figure 18: Series resistive circuit to test DC sweep

Once you put together the circuit, start a new simulation profile and give your simulation a name. In the *Simulation Settings* window under *Analysis Type* choose *DC Sweep*. Under *Sweep Variable* select voltage source and enter name of your source (*Vs* in our case). Choose *Sweep Type* to be *Linear* and enter *start value*, *end value*, and *increment* that you need (5V, 10V, and 0.5V in our case).

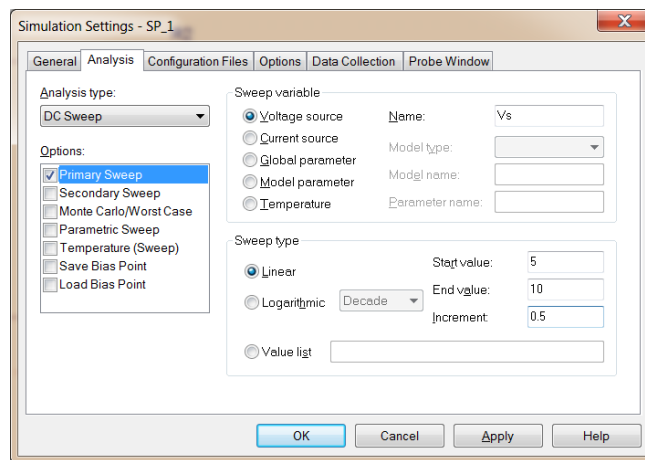


Figure 19: Simulation Setting Window for DC Sweep

While doing *DC sweep* analysis, first you have to determine the node at which you want to check the voltage change (*with respect to ground*) or component across which you want to check voltage

change (*voltage drop*) when input voltage is changed. To select node or a component, we use *probes*.

There are four probes in the simulation menu as shown in *figure 20*. From left to right, first probe is to plot *node voltage (with respect to ground)*, second is to plot *voltage drop or potential drop* across a component, third is to plot *current flow* through a branch, and the last one is to plot *power delivered or dissipated* in a component



Figure 20: Available Probes in Simulation Menu.

Suppose you want to see how voltage at the junction of $R2$ and $R3$ will change when you change the input voltage. Place voltage probe (first one from the left in *figure 20*) at the junction of $R2$ and $R3$, as shown in *figure 21*.

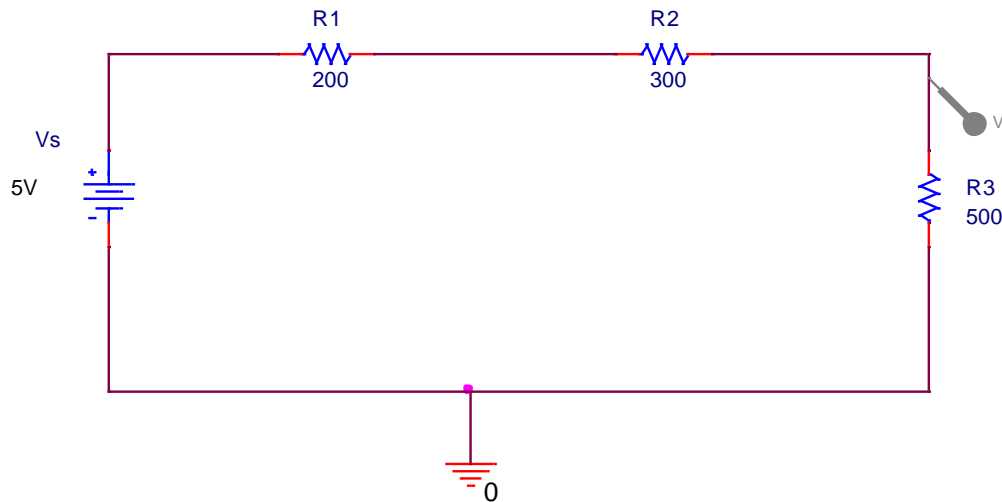


Figure 21: Placement of Voltage Probe to Measure Node Voltage

When you run your simulation, PSpice A/D window will open and display the simulation results.

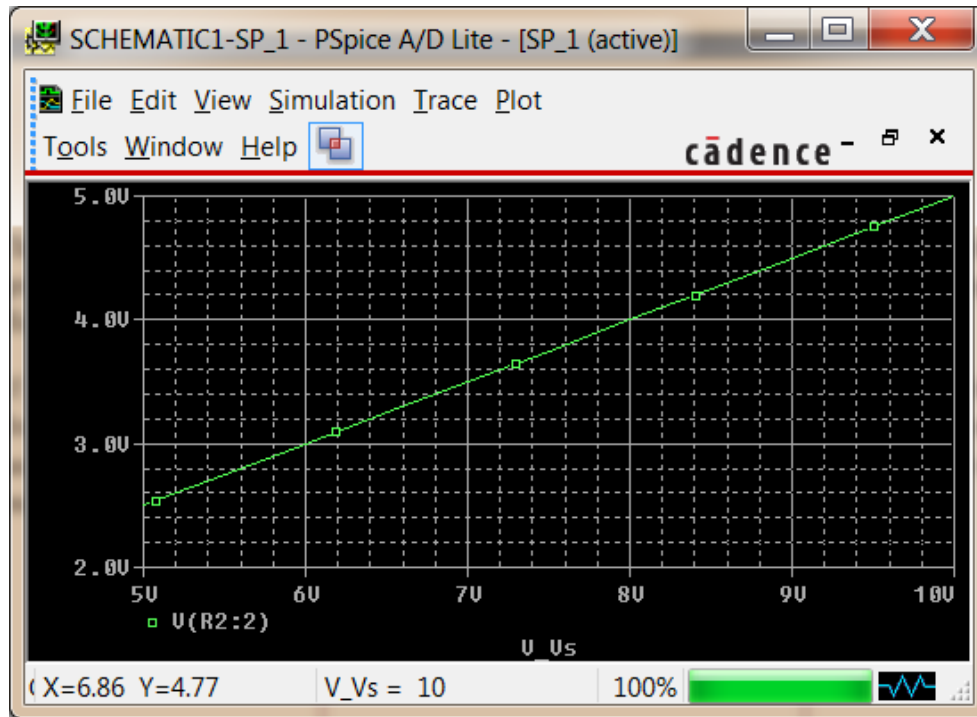


Figure 22: Simulation Result for DC Sweep between V_s and R_3

This graph represents how node voltage at the junction of R_2 and R_3 is changing with respect to the ground (or in other words, voltage across R_3) when you change voltage of the input source. Horizontal axis is input voltage source and vertical axis is the voltage across R_3 .

If you want to measure voltage *across* any component that is not connected to ground, you have to use the probe with two ends (*potential difference probe*) from the list. Suppose you want to measure the voltage drop across R_2 ; choose double-ended probe and place one end (positive end) at the beginning of R_2 and the other end (negative end) at the end of R_2 , as shown in the following figure. Here you cannot use single-ended probe as none of the terminals of R_2 is grounded.

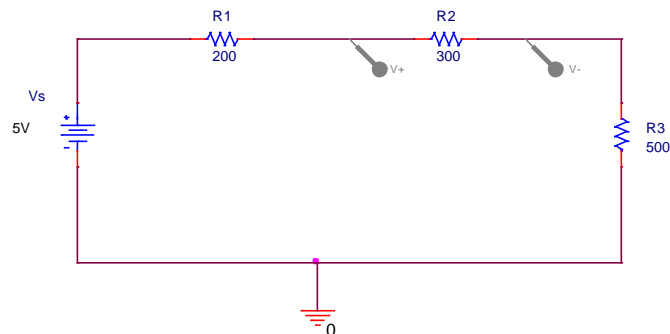


Figure 23: Measurement of DC Sweep Voltage across a Component

When you run your simulation, you will see a plot of voltage across $R2$ versus input voltage V_s . Similarly, you can create a plot for the current flowing through a component (choose *current probe* and put it at the terminal where current is entering into the component to see positive values of current) and power dissipating through a component (choose *power probe* and put it on the component). You can also plot voltage drop across multiple components, current flowing through multiple components and power dissipated through multiple components versus change in the input voltage. For example, if we want to check how power dissipation is changing through all three resistors versus the input voltage, here's how we are going to place power probes:

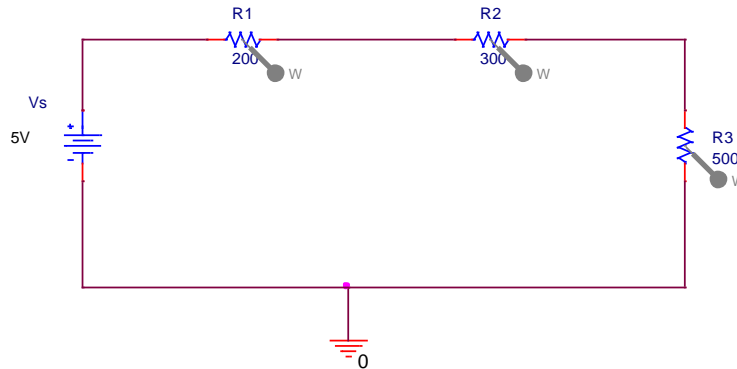


Figure 24: Measurement of Power Change through Multiple Components

When you run your simulation, you will see the following results.

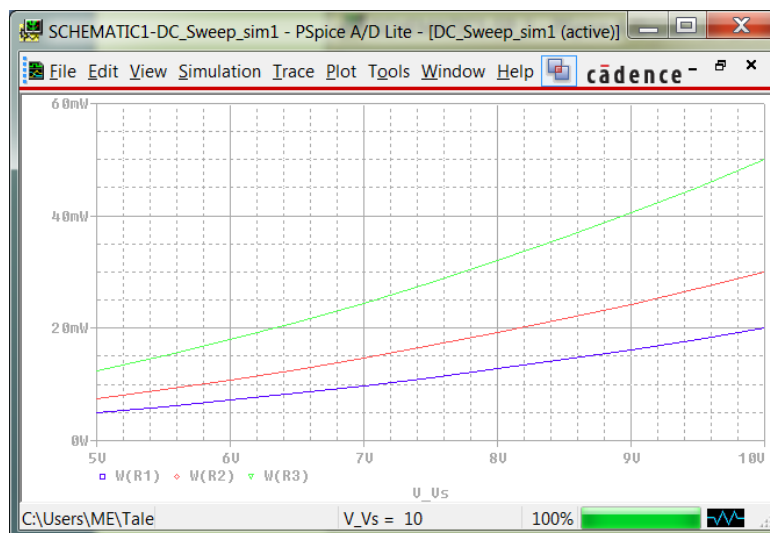


Figure 25: Simulation Results using Multiple Power Probes

NOTE: In *figure 25*, background color is white. The default background color for PSpice simulation is black. To change background, foreground, and probes colors, go to *Tools* → *Options* → *Color Settings* in your PSpice A/D window. Background and foreground color can change as soon as you choose any new color, however, color of probes will only change when you remove probes, close your design and then re-open it. There is another option to copy your simulation to clipboard except print screen option; go to *Window* → *Copy to Clipboard*. Make sure to explore this option as well.

EXERCISE – 3

For the *RLC* circuit that you created in *Exercise 2*, find the change in current values through *R1*, *R2*, and *R4* when you change the input voltage from 0 to 10 V with interval of 0.1V

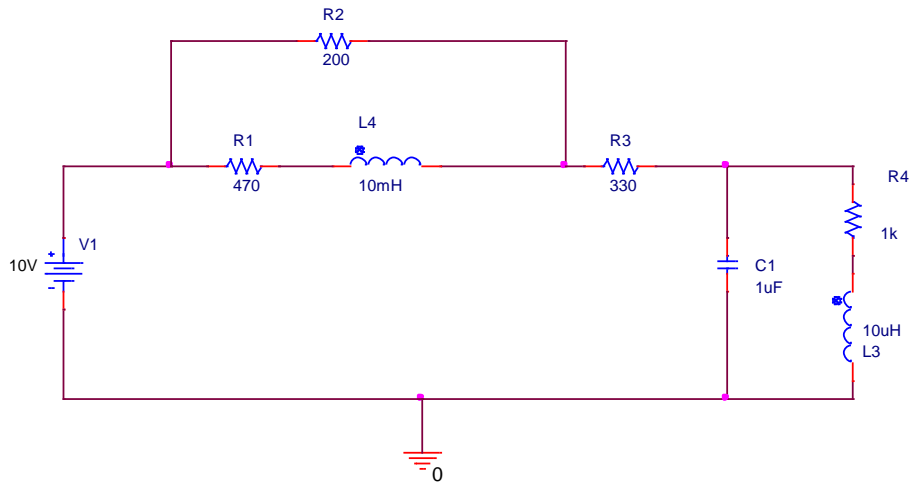


Figure 26: Circuit for Exercise - 3

There is another way to check DC sweep across different components. In the PSpice simulation window, go to the *Trace* menu at the top and select *Add Trace*. For example, if you want to perform DC sweep analysis on the circuit from *figure 18* and do not put any probe on the circuit, your Pspice A/D window will open with a blank plot. From this window select *Trace* and *Add Trace*, you will see the following dialog box,

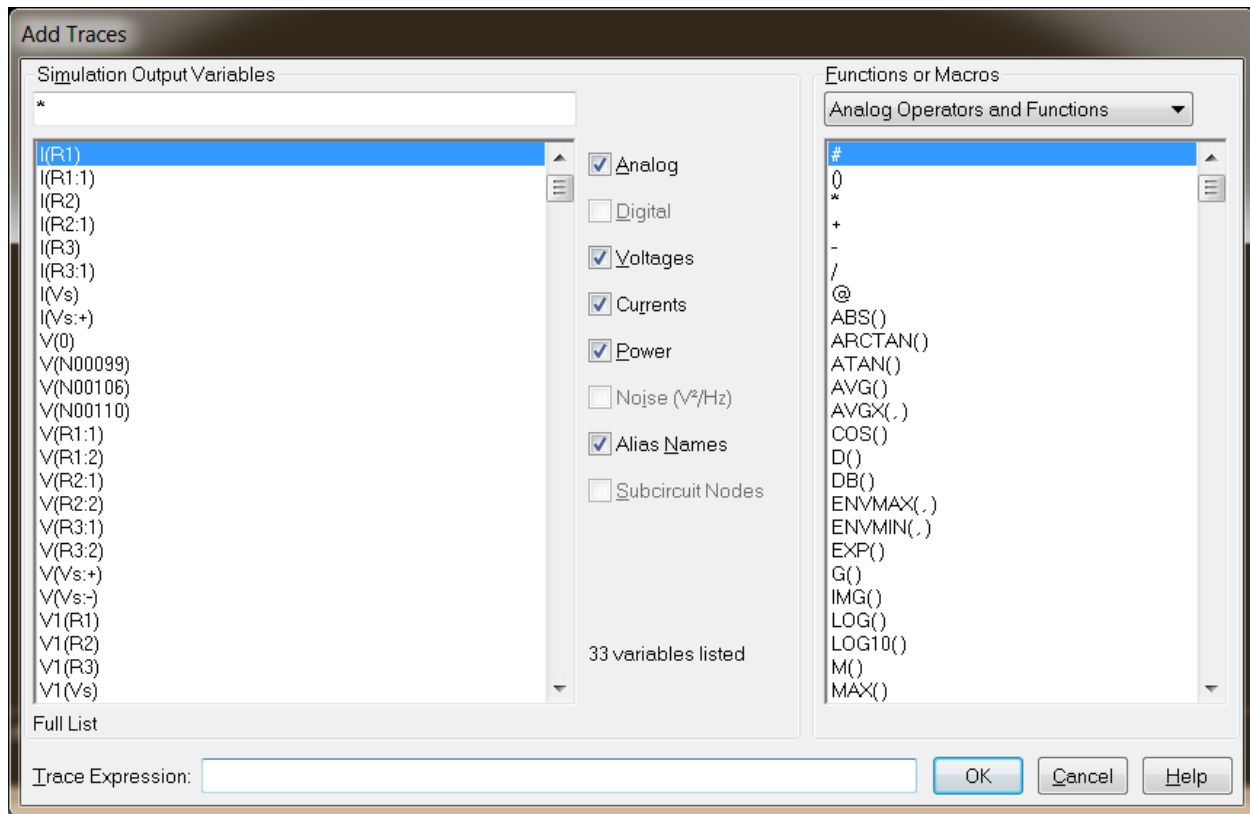


Figure 27: DC Sweep through Trace

In this window you can see a list of all the node voltages, currents, and power dissipations for which your sweep plot is available. Select whichever you want and press OK to check their DC sweep plot. If you want to see the voltage drop across some component then write down its expression in the *Trace Expression* field and press OK. For example, if you want to plot voltage drop across *R1*, write down $V(R1:1) - V(R1:2)$ in the *Trace Expression* field and press OK. You will see the plot of voltage drop across *R1* vs. input voltage in the trace window of PSpice A/D.

NOTE: To delete any trace, select its label and hit *delete* from your keyboard. To delete all traces, go to *Trace* → *Delete All traces* from menu bar.

Transient Analysis

The third type of analysis is *transient analysis*. Transient analysis refers to the change in electrical quantities with respect to time.

Transient Analysis with Constant DC Sources

Let's start with the transient analysis of a simple *RL* circuit, as shown in *figure 28*. Transient analysis with constant DC sources requires a switch that can either open or close at a specific time. These switches can be found in the *eval* library (*eval.olb*). Switch that is used in the following circuit will close at time $t = 0$ sec.

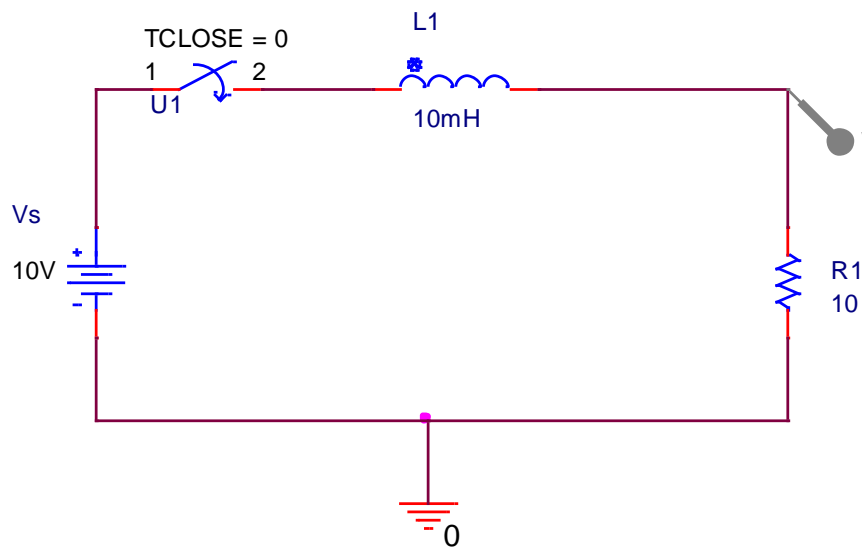


Figure 28: Simple Series RL Circuit to Analyze DC Transients

To analyze transient analysis, create a new simulation profile for the circuit and from *Simulation Settings* dialog box, choose *Analysis Type* to be *Time Domain (Transient)*. Enter the values for the total run time and the maximum step size in the dialog box and run the simulation. You should see the voltage change across the resistor *R1* from zero to 6ms, as shown in *figure 30*.

Note: The *maximum step size* dictates the number of points that will be calculated and plotted when the simulation will be executed. If your simulation is not smooth and it has sharp edges, your maximum step size is large, and you are plotting very few points. You should go back and reduce the maximum step size.

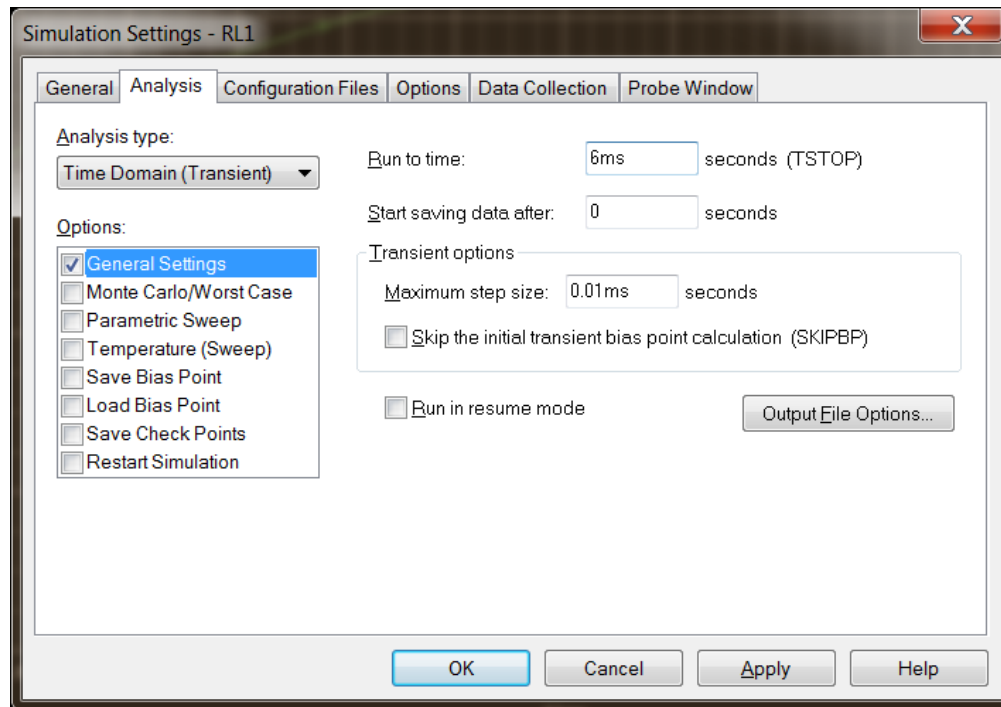


Figure 29: Dialog Box for Transient Analysis

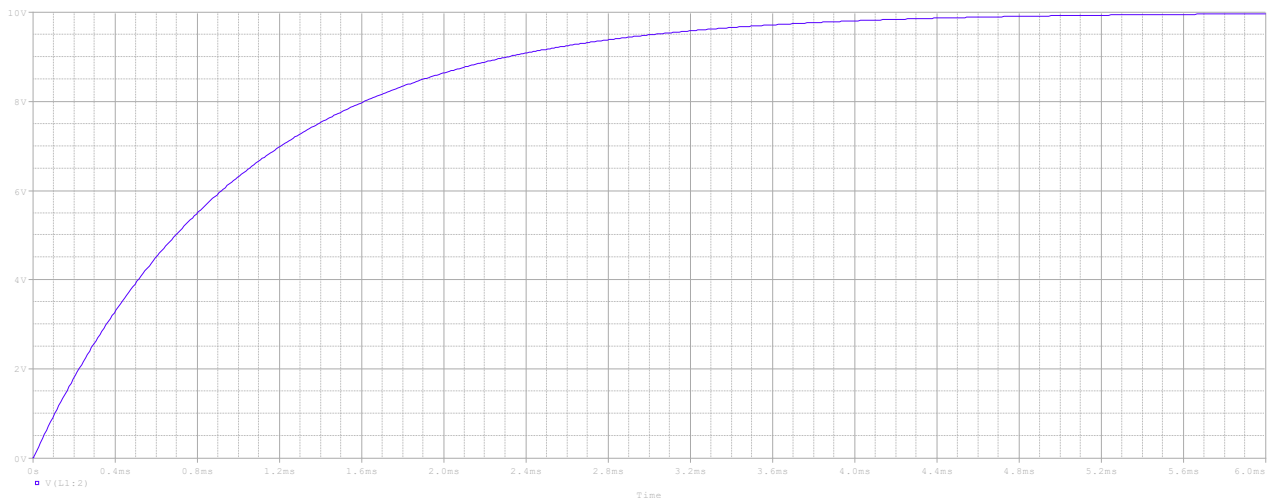


Figure 30: Transient Voltage across R1

NOTE: Figure 30 is printed using Window → Copy to Clipboard option

EXERCISE – 4

For the circuit shown in *figure 31*, find the transient voltage across $R3$ when the switch will open at 6ms. Plot the transient voltage for 20ms (*Run to time*) with the maximum step size to be 0.01ms. Start saving data after 0s (the first point it will display the data value).

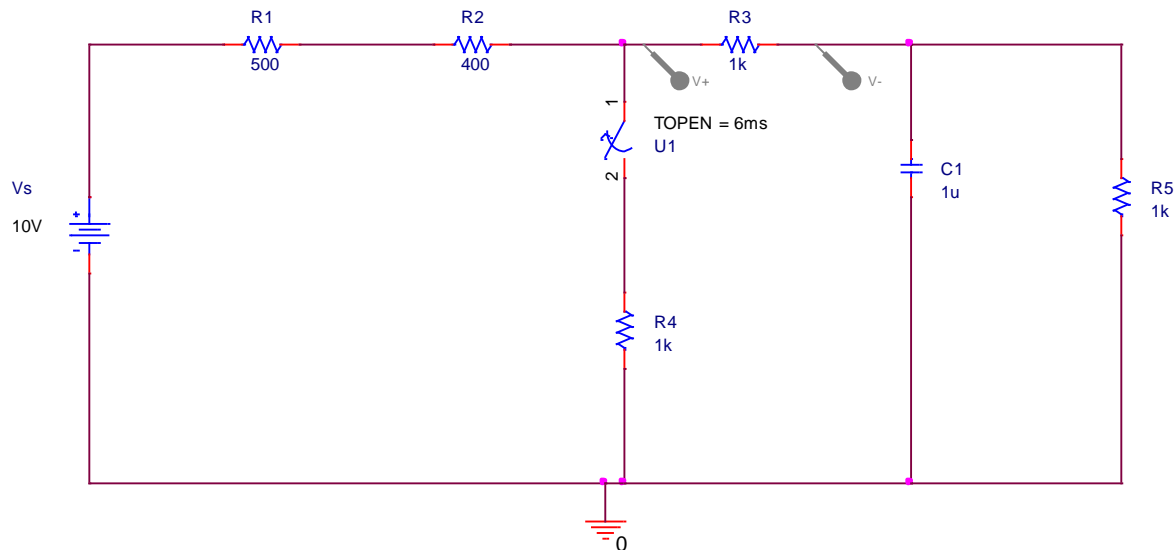


Figure 31: RC Circuit for Exercise 4

Transient Analysis with AC Sources

AC (Alternating Current) sources are the ones that change their polarity (positive and negative voltage sides) with respect to time. These sources can be comprised of square waves, sinusoidal waves, triangular waves or any other type of periodic functions. Remember that not every alternating source is AC. If source's value is changing but not its polarity then it is not an AC source, it is an *alternating DC source*.

Let's use a sinusoidal AC source and plot its transient response for some circuit component. Consider circuit given in *figure 32*. Voltage source used is *VSIN* under *source* library.

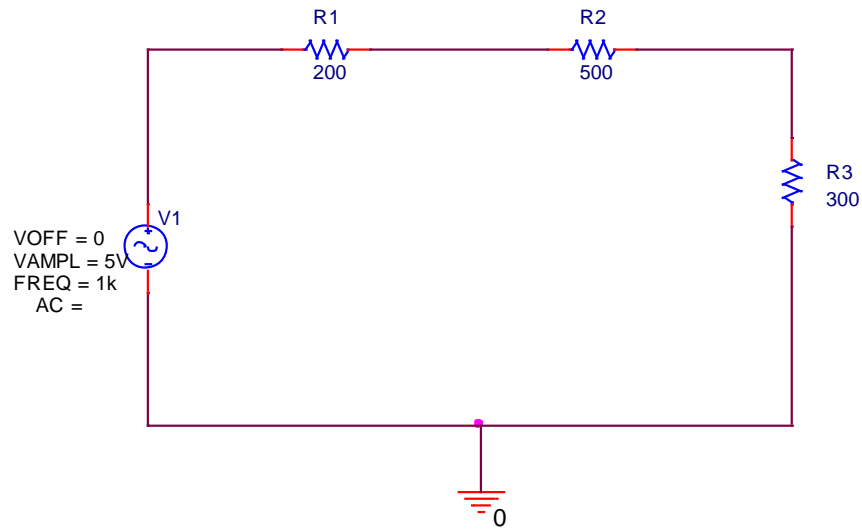


Figure 32: AC Series Resistive Circuit

When you simulate it and compare the voltage drop of *R2* against the source, you will see the following transient output:

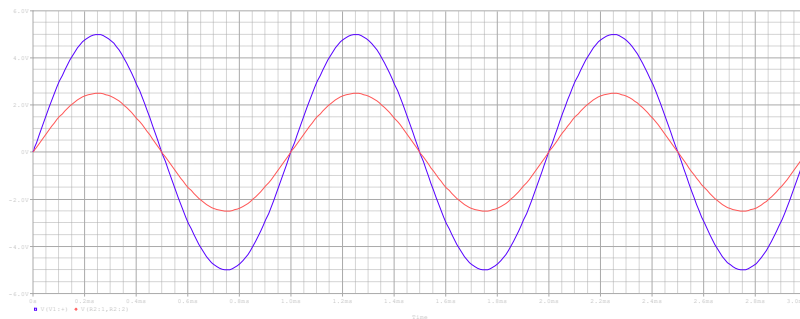


Figure 33: Transient Voltage across *R2* versus the Input Source

To find different values of waveforms, you can use *cursors*. For example, *maximum* or *peak* value of output voltages can be found out by pressing cursor button and choosing *cursor peak* (first button from left in the cursor menu). According to cursor values, peak value of the output voltage is 2.5V, as shown in *figure 34*.

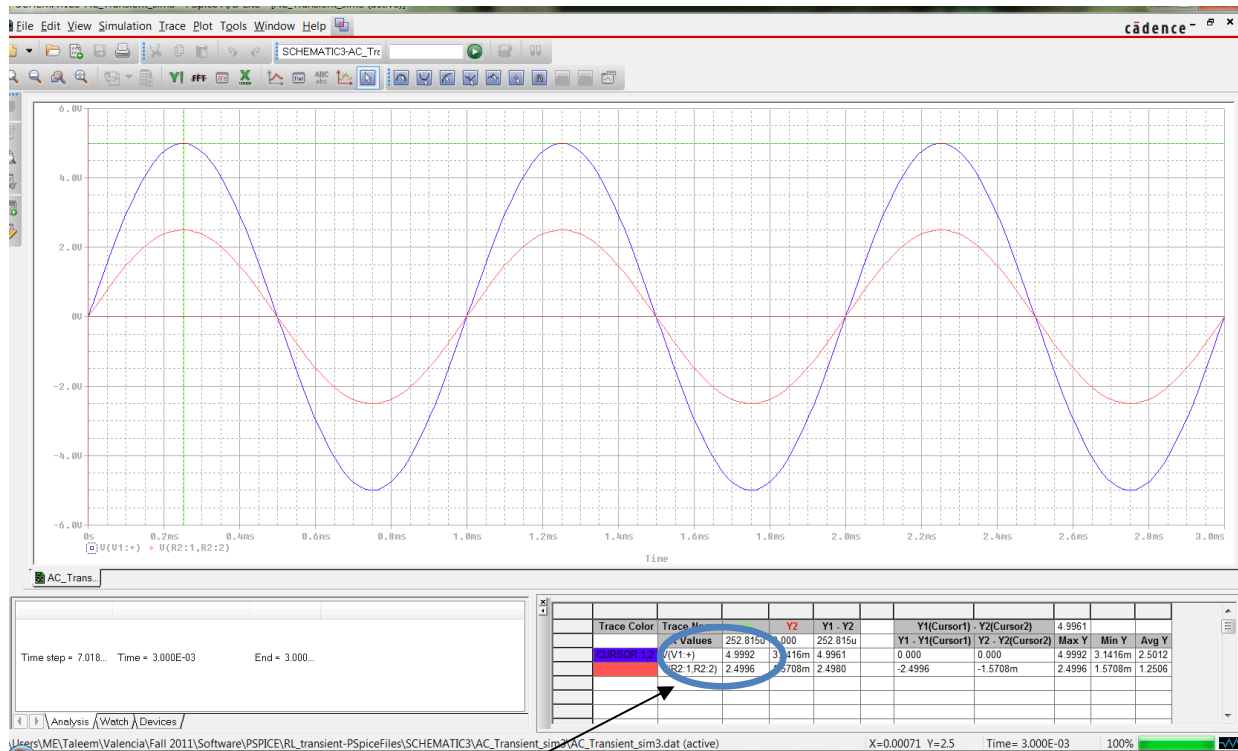


Figure 34: Cursor Peak showing Peak values of Input and R2 Voltages

Likewise, you can grab cursor lines and move them around to find voltage at any corresponding time. There are two cursors, **Cursor 1 (red)** and **Cursor 2 (green)**. These are default colors of the cursors. Cursor 1 is operated by the *left* mouse button whereas cursor 2 is operated by the *right* mouse button. The color of a cursor can be changed from **Tools** → **options** → **Cursor Settings**

NOTE: If you want to change *phase shift* of the AC sinusoidal source and its field is not given to you, go to the properties by double clicking on the source and add *New Column*. Name this new column as *Phase* and choose any value (in degrees) to change the phase shift. For example, if you want a cosine source instead of sinusoidal, use phase shift to be 90.

EXERCISE – 5

Create the following circuit in *PSpice* and carry out the transient analysis. Trace the following voltages for 10ms with the step-size of 0.01ms: (i) Input voltage, V_I (ii) Voltage across $L1$, v_{L1} (iii) Voltage across $L2$, v_{L2} .

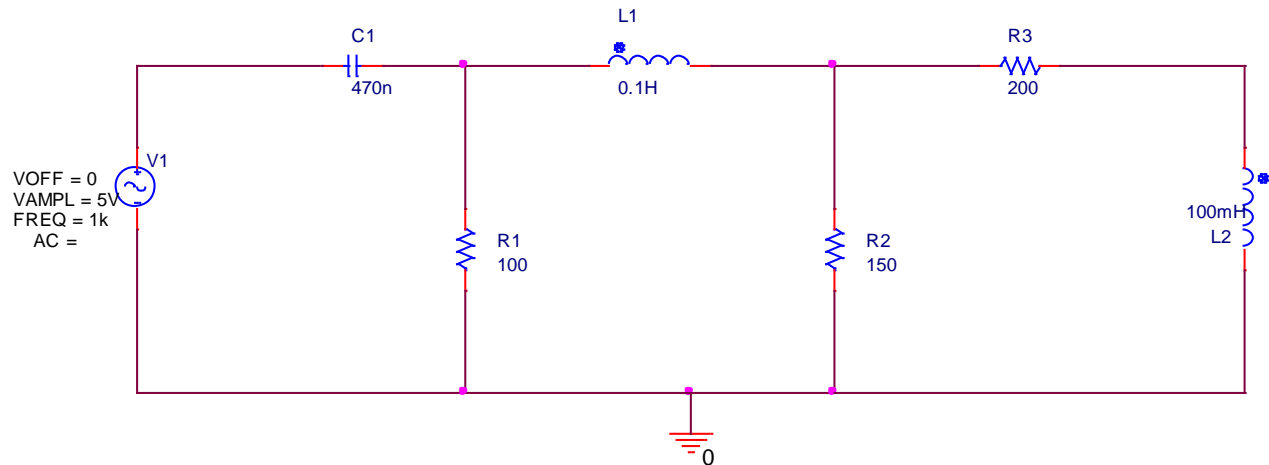


Figure 35: RLC Circuit for Exercise 5

Observe voltage values for all the three voltages at the following time instances:

- (i) $t = 2.3\text{ms}$
- (ii) $t = 5.2\text{ms}$
- (iii) $t = 9.0\text{ms}$

EXERCISE – 6

Perform transient analysis on the circuit shown in *figure 36* and find the voltage across C_1 and L_1 at 2.5ms and 7.7ms. Note that the sinusoidal current sources ($ISIN$) instead of voltage sources are used in this circuit. Run the simulation for 8ms with a step-size of 0.01ms.

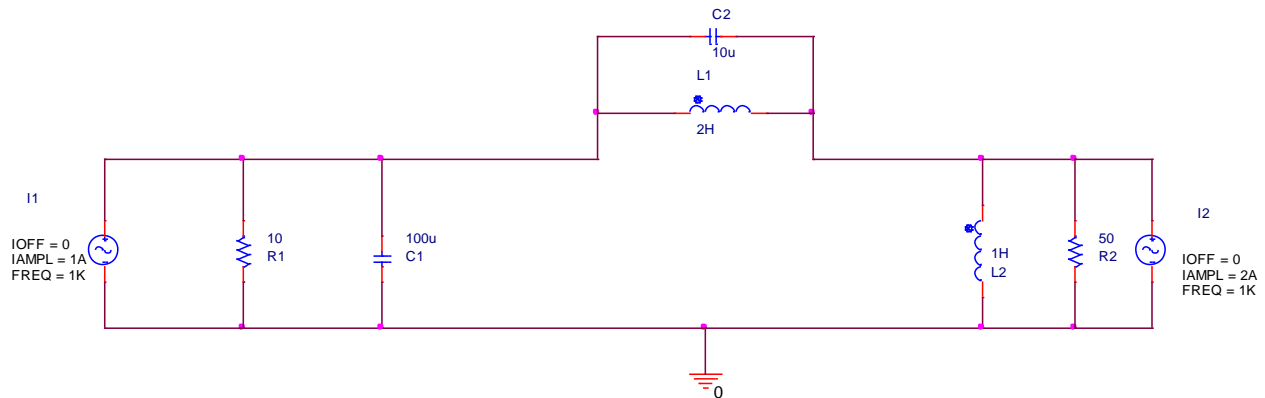


Figure 36: Circuit with Sinusoidal Current Sources

Transient Analysis with Non-Sinusoidal Sources

A number of circuits are required to be excited by non-sinusoidal AC sources, especially *square* or *rectangular* sources. Generally, these sources are termed as *periodic pulse sources* or *periodic pulse waveforms*, most common of which is a square wave. In PSpice, *VPULSE* is used for pulse waveforms (located in the *source* library). Different parameters associated with *VPULSE* are explained in the following figure.


<p>VPULSE</p> <p>V1 = V2 = TD = TR = TF = PW = PER =</p> 	<table border="1"> <tr> <td>V1</td><td>Low Level Voltage</td></tr> <tr> <td>V2</td><td>High Level Voltage</td></tr> <tr> <td>TD</td><td>Time delay before the first transition</td></tr> <tr> <td>TR</td><td>Time it takes to go from low level to high level; <i>Rise Time</i></td></tr> <tr> <td>TF</td><td>Time it takes to go from high level to low level; <i>Fall Time</i></td></tr> <tr> <td>PW</td><td>Time span for which waveform is high during one period; <i>Pulse Width</i></td></tr> <tr> <td>PER</td><td>Time period of waveform</td></tr> </table>	V1	Low Level Voltage	V2	High Level Voltage	TD	Time delay before the first transition	TR	Time it takes to go from low level to high level; <i>Rise Time</i>	TF	Time it takes to go from high level to low level; <i>Fall Time</i>	PW	Time span for which waveform is high during one period; <i>Pulse Width</i>	PER	Time period of waveform
V1	Low Level Voltage														
V2	High Level Voltage														
TD	Time delay before the first transition														
TR	Time it takes to go from low level to high level; <i>Rise Time</i>														
TF	Time it takes to go from high level to low level; <i>Fall Time</i>														
PW	Time span for which waveform is high during one period; <i>Pulse Width</i>														
PER	Time period of waveform														

Figure 37: Explanation of different parameters associated with the Pulse Waveform

EXERCISE - 7

Design a series RLC circuit with $R = 1\text{K}\Omega$, $L = 100\mu\text{H}$, and $C = 1\text{nF}$. Use a square waveform as your input with zero to 5V. Keep time delay, rise time and fall time, all to be zero. Pulse width (PW) is $6.75\mu\text{s}$, and to make a square wave, time period (PER) of the waveform should be double the size of the pulse width.

Simulate your circuit for transient analysis. Keep your run time to be twice of the time period and step-size to be $0.01\mu\text{s}$. Trace input voltage (square wave) and voltage across the capacitor on the same plot.

AC Sweep Analysis

The final analysis type is AC Sweep. As it was discussed earlier, DC sweep analysis corresponds to the change of voltage, current, or power across a component in a DC circuit when input voltage is changed from one value to another. AC Sweep Analysis corresponds to the change of voltage, current, or power when *frequency* of an AC sinusoidal voltage is changed from one value to another. Hence, AC sweep analysis is a plot between voltage, current, or power versus frequency.

Let's analyze the following first order RC Low-pass passive filter for its *frequency response*. Make sure to use VAC instead of VSIN for the AC sweep analysis. Set the AC voltage to be 10V.

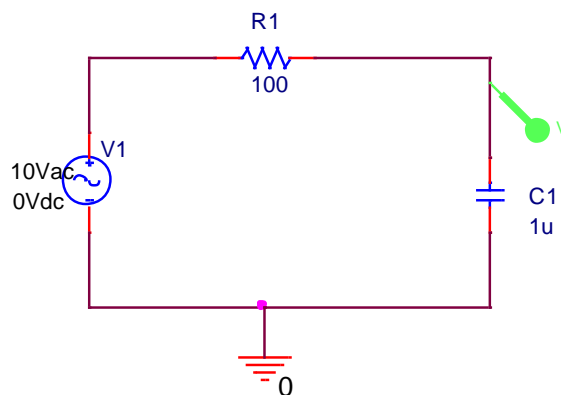


Figure 38: RC Passive LPF

Create a new simulation profile and choose AC Sweep/Noise in the *Simulation Settings* window under *Analysis Type*. You can use frequency scale to be either linear or logarithmic. Choose logarithmic scale and give values for the *start frequency*, *end frequency* and *points per decade*.

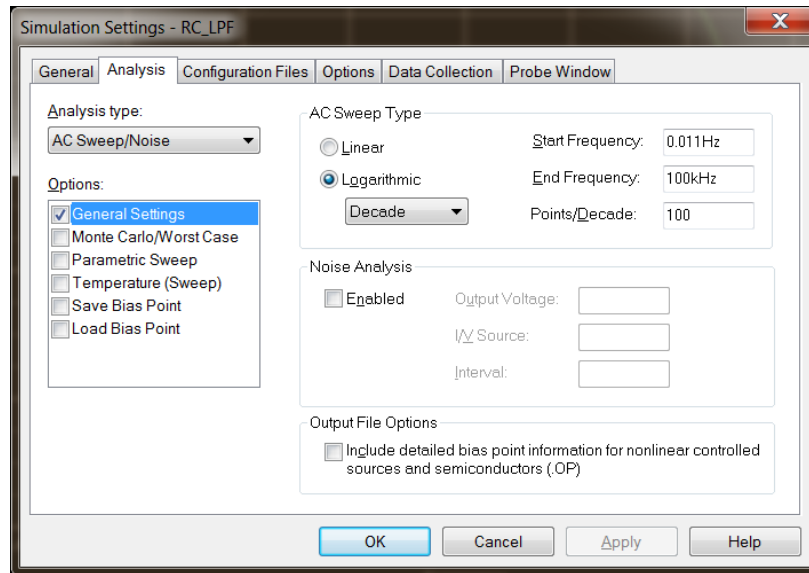


Figure 39: Simulation Settings for AC Sweep Analysis

Circuit simulation should give you the following plot:

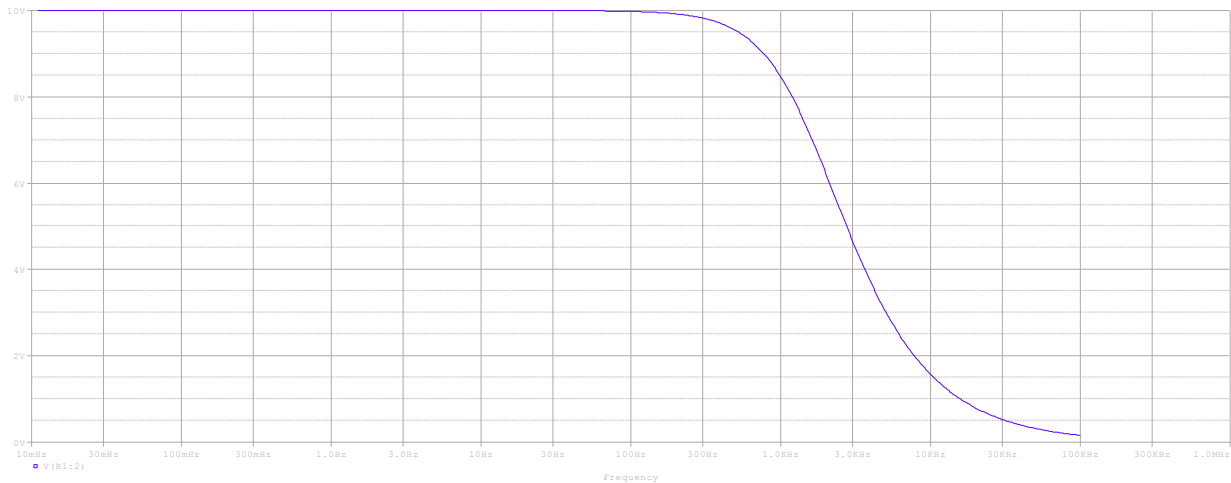


Figure 40: Frequency Response of RC LPF

EXERCISE – 8

For the following *RC band-pass filter*, perform AC sweep analysis to trace the frequency response across the load resistor R_L . Start at 1Hz and trace up to 1MHz (be careful with the Mega prefix, do not use ‘M’). Use logarithmic sweep with 100 points per decade.

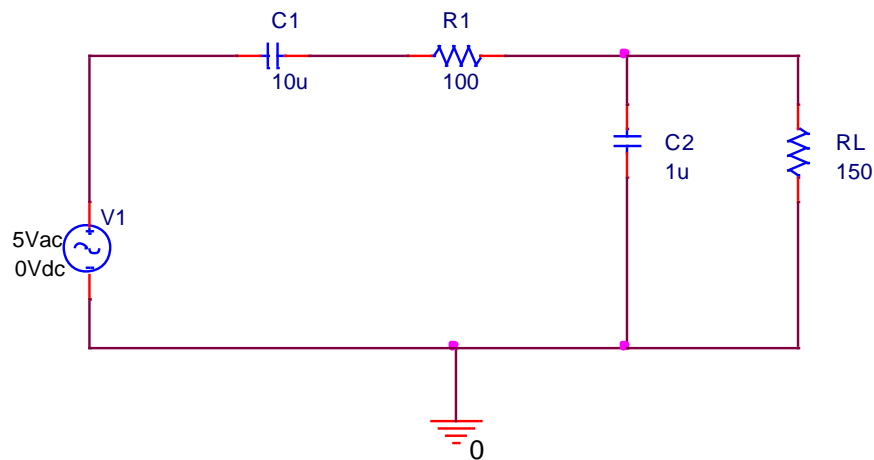


Figure 41: RC Band-pass filter

ELECTRONIC CIRCUITS

This part of the tutorial deals with circuits containing basic semiconductor components including *diodes* and *transistors*. Let's start with diodes. There are different types of diodes including, but not limited to, *rectifier diodes*, *zener diodes*, *photo diodes*, *light emitting diodes (LED)*, and *Schmitt trigger diodes*. Diodes are two terminal devices like resistors, capacitors or inductors. However, unlike resistors, inductors, and some types of capacitors, two terminals of diodes play an important and a very different role in the conduction of current. *Figure 42* shows circuit symbol of a PN junction rectifier diode with its positive (anode) and negative (cathode) sides.

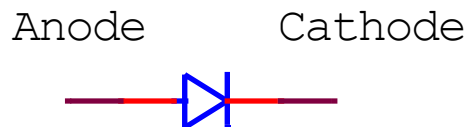


Figure 42: PN Junction Rectifier Diode

Diode acts as an electronic switch. When the anode of a diode is more positive than its cathode, it is said to be in the *forward biased* region. In the forward biased region when voltage difference between the anode and cathode exceeds a certain voltage, called junction voltage (0.7V for silicon diodes and 0.3V for germanium diodes), diode starts conducting. Voltage across the diode becomes almost constant (either 0.7 or 0.3, based on which diode is being used) and current keeps on increasing exponentially if input voltage keeps on increasing. Ideally, it is assumed that when diodes are conducting, their junction voltage is zero, to keep analysis simple. However, in PSpice, you will see a voltage drop of around 0.7V across a Silicon diode when it is conducting.

When a diode's cathode is more positive than the anode, it is said to be in the *reverse biased* region. In this region, diode does not conduct any current (except a very small reverse saturation current) until voltage becomes very large. If it becomes very large and exceeds what is called *breakdown* voltage, a large amount of current starts flowing through the diode. Generally, rectifier diodes are not operated in the breakdown region as they get damaged. However, a special type of diode called Zener diode is made to operate in the breakdown region without any problem.

Let's analyze diode characteristics from a simple circuit. Create the circuit shown in *figure 43*. Diode is located in the *eval* or *diode* library.

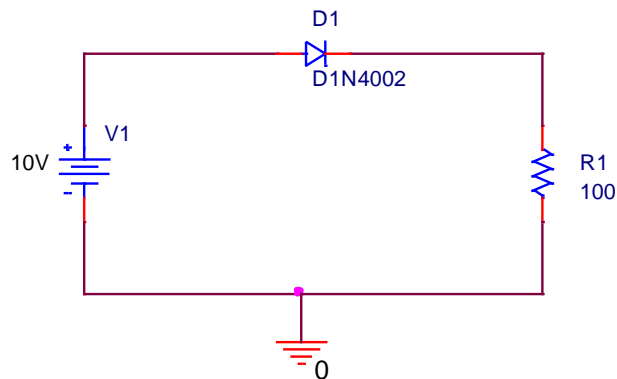


Figure 43: Simple Diode Circuit

Create a new simulation profile and choose *DC Sweep* simulation type. Sweep the input voltage from -10V to 10V and observe the voltage across diode. You will see the simulation shown in *figure 44*.

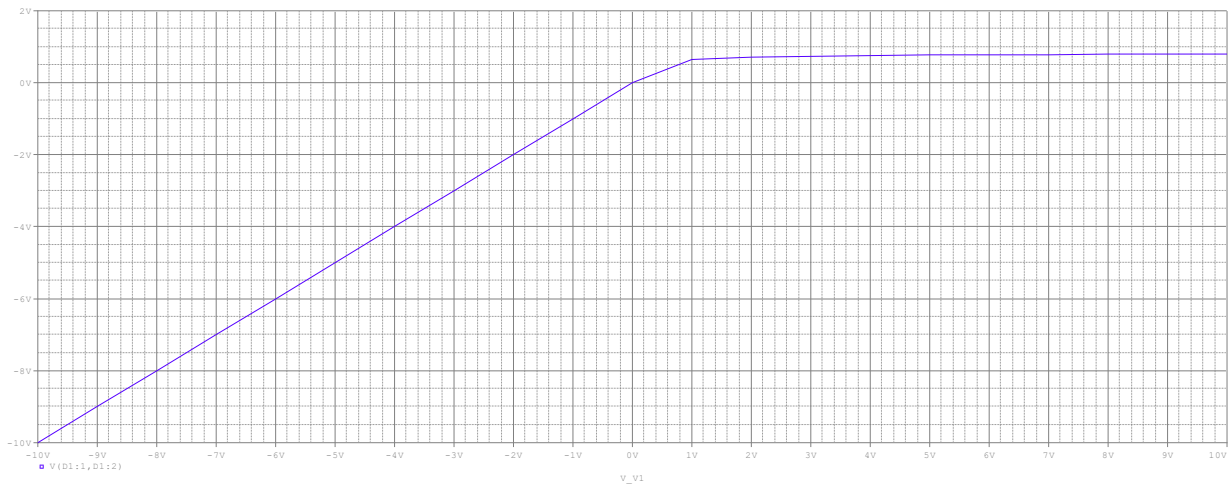


Figure 44: Voltage Drop across Diode

Voltage Characteristics of diode, as shown in *figure 44*, depicts that it does not conduct until approximately 0.6-0.7V, at which point it starts conducting and voltage across it almost becomes constant.

Let's plot both voltage and current characteristics of a diode in two sub-plots. From top menu, go to *PLOT → ADD PLOT TO WINDOW*. This will add another trace window on the same plot. Put a current probe on any component and it will show you how current is changing in the circuit as shown in the following figure.

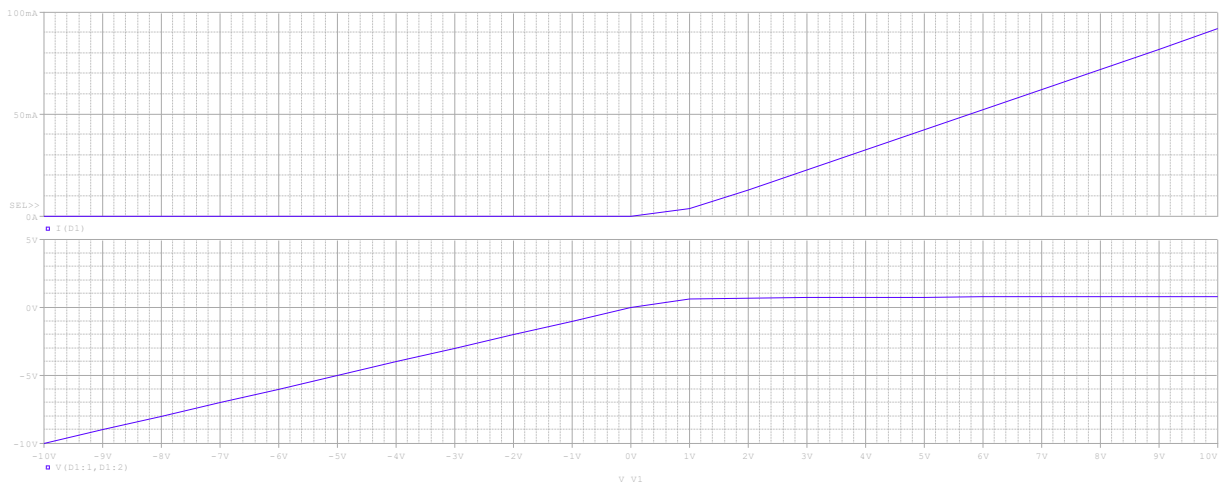


Figure 45: Diode Current (top) and Voltage (bottom)

Figure 45 shows that as long as diode is not conducting (off), there is no current in the circuit. Once diode starts conducting around 0.7V, current starts increasing in the circuit with the increase of the input voltage.

Let's perform *transient analysis* of the same circuit with a sinusoidal AC source. Circuit is shown in figure 46 and simulation is shown in figure 47.

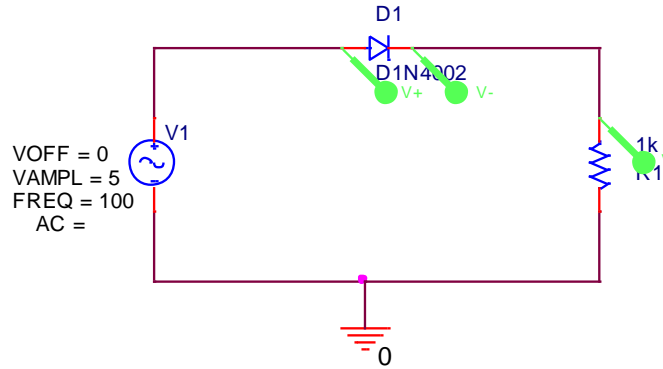


Figure 46: Half-wave Rectifier Circuit

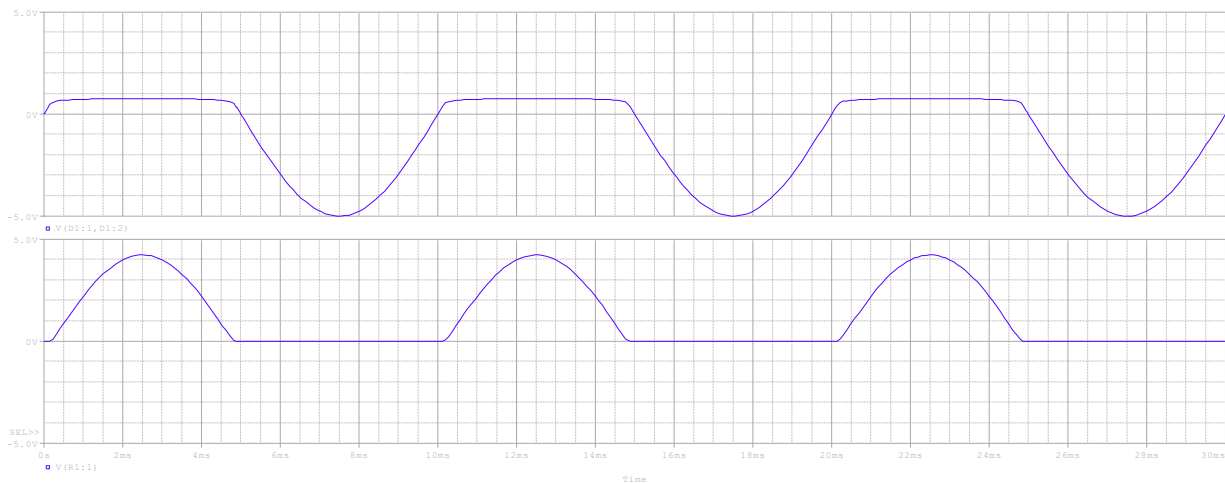


Figure 47: Diode Voltage Drop (Top) and Voltage Drop across R1(Bottom)

It can be observed from figure 47 that as long as diode is OFF, i.e., when the voltage from anode to cathode is less than 0.7 V approximately, there is no voltage drop across the output resistor as there is no current flow in the circuit. All the voltage is dropped across diode. When diode is ON and current is flowing in the circuit, there is a voltage drop across the output resistor and the voltage across the diode is constant, which is equal to the junction voltage ($\approx 0.7V$). Observe that there is no negative voltage cycle across the output resistor at any time, hence we say that the output voltage has been *rectified* from AC to *pulsating DC*. Since diode is conducting (ON) for only half

of the input voltage cycle, this circuit is called *Half-wave Rectifier*. Output of a half-wave rectifier can be made smoother using some filters.

There are rectifier circuits where diode conducts during the whole input voltage cycle. Those circuits are called *Full-wave Rectifiers* and their performance is better than the half-wave rectifier circuits. Some of the commonly used full-wave rectifier circuits are *center-tapped transformer full-wave rectifiers* and *full-wave bridge rectifiers*. Bridge rectifier is the subject of *exercises 9 and 10*.

EXERCISE – 9

Create a *Full-wave Bridge Rectifier* circuit as shown in *figure 48* and perform transient analysis to trace voltage across *RL* versus the input voltage for 5ms.

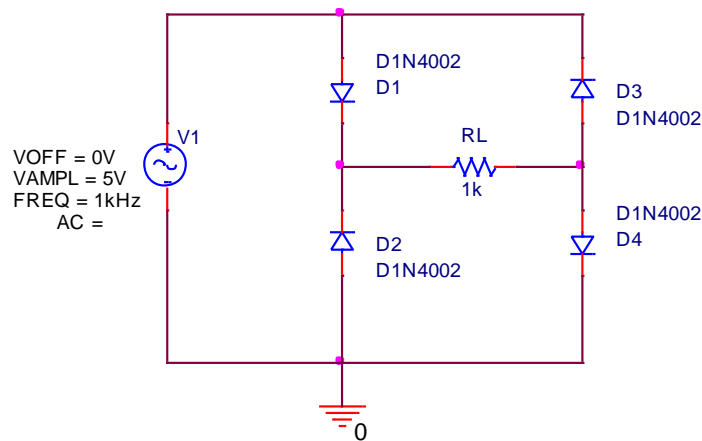


Figure 48: Full-wave Bridge Rectifier

EXERCISE – 10

For the full-wave bridge rectifier that you designed in *exercise 9*, add a capacitor in parallel to *RL*. Choose its values as follows: $1\mu\text{F}$, $10\mu\text{F}$, and $100\mu\text{F}$, one by one, and trace the voltage across *RL* vs. the input voltage source. Observe how the *pulsating DC* voltage is converted into almost constant DC voltage across *RL*.

Transistor Circuits

Transistor is another very commonly used semiconductor component. Unlike any other component that has been discussed so far, transistor has *three* terminals. There are two major types of transistors: *Bipolar Junction Transistor (BJT)* and *Field Effect Transistor (FET)*. FET is further improved into another type of transistor called *Metal-Oxide Semiconductor Field Effect Transistor (MOSFET)*.

Let's start our discussion with BJT circuits. Three terminals of BJT are named as *Base (B)*, *Collector (C)*, and *Emitter (E)*. BJT is a current-controlled current device where base current controls the flow of current from emitter to collector. There are three regions of operation of BJT: *Active Region (BE is forward biased and BC is reverse biased)*, *Saturation Region (both BE and BC are forward biased)*, and *Cut-off Region (both BE and BC are reverse biased)*. Two of the most common applications of transistors are *amplification* and *switching*. There are two types of BJTs: *NPN* and *PNP*. Figure 49 shows circuit symbols of both types of transistor with their terminal names.

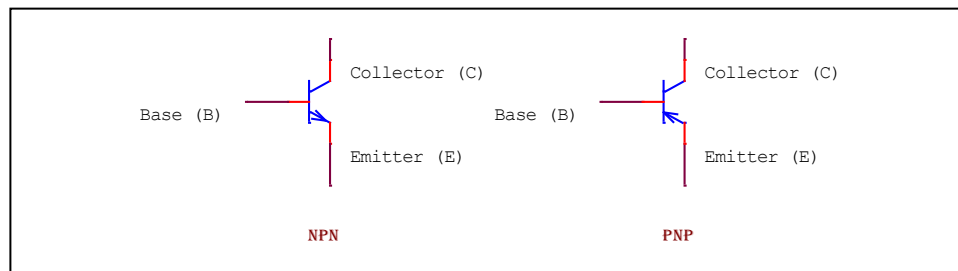


Figure 49: Types of BJTs

Let's create BJT circuit shown in figure 50 and perform DC Bias analysis. BJT can be found in the *eval* or *bipolar* library.

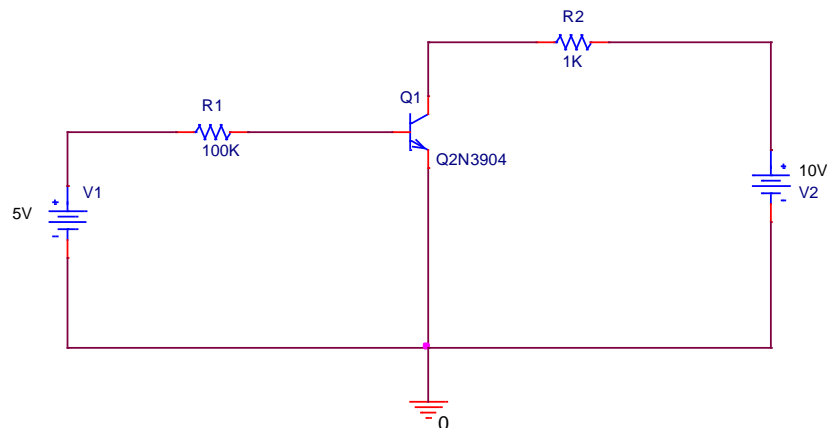


Figure 50: BJT Common Emitter Amplifier

As it can be seen from the DC bias point values (*figure 51*), since voltage $V_{CE} = 3.00\text{V}$, BJT is operating in the *active* region. For the transistor to be in the *saturation* region, V_{CE} is approximately equal to 0.2V (for the sake of simplicity, we take it as 0V) and in the *cut-off* region, when no current is flowing through the BJT, V_{CE} is approximately equal to the value of voltage source V_2 and current is very close to zero.

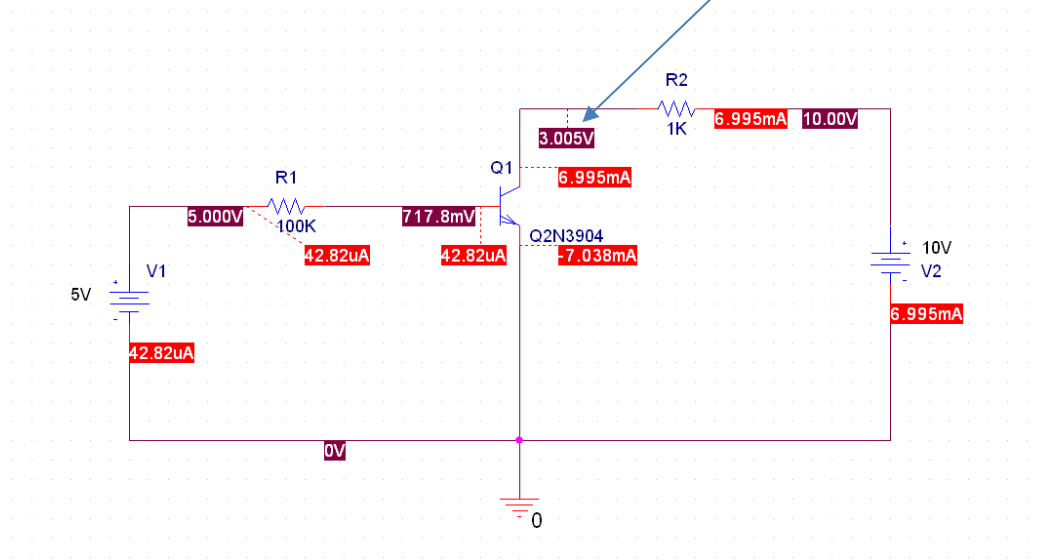


Figure 51: Bias Points of Common Emitter Amplifier

To check different regions of BJT operation, let's perform DC sweep analysis with value of the source V_1 changing from -1 to 10 volt and trace the voltage V_{CE} . Result is shown in *figure 52*.



Figure 52: Different Regions of BJT Operation

Transient Analysis of Transistor Circuits

Transient analysis of circuits containing transistors shows their switching or amplification characteristics. Let's perform transient analysis for the BJT common emitter amplifier circuit from *figure 50*. Modify the circuit as shown in *figure 53*:

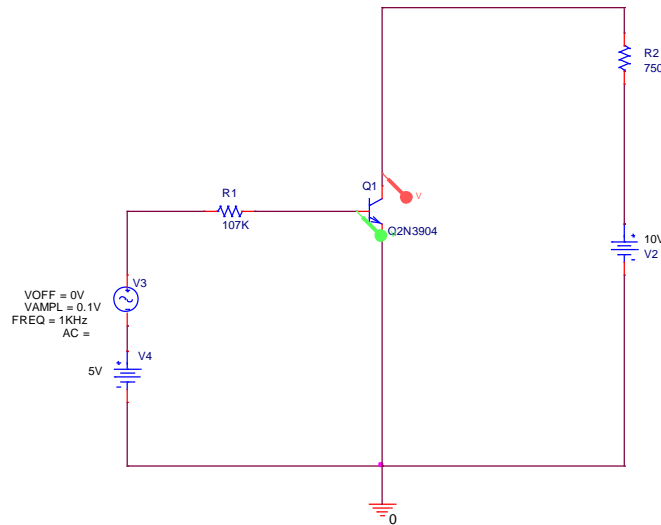


Figure 53: BJT Common Emitter Amplifier with AC Source

When you perform the transient analysis and observe the voltage gain between *base* and *collector* of the circuit, you will see the following traces:

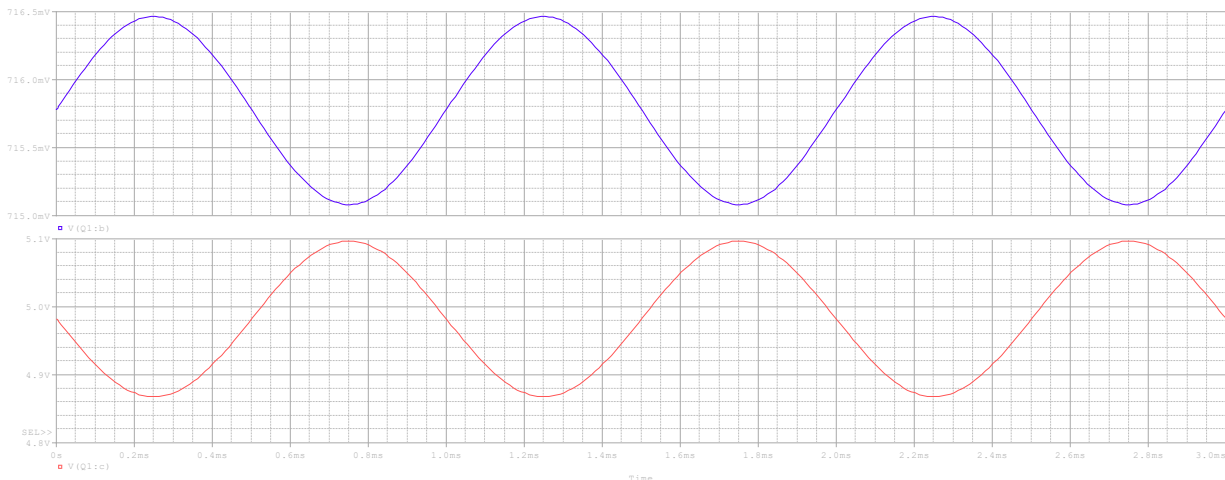


Figure 54: Voltage Amplification from Base to Collector; Blue Waveform is the Base Voltage & Red is the Collector Voltage

Observe how voltage is amplified from base to collector ($Voltage\ Gain = v_c(p-p)/v_b(p-p) \approx 228mV/1.38mV = 165$). Also, observe that the voltage at collector is inverted compared to the base voltage. This is why it is also called an *inverting voltage amplifier*.

EXERCISE – 11: Design the common emitter amplifier shown in *figure 55* and calculate the voltage gain $\frac{v_{out}(pp)}{v_{in}(pp)}$ from the transient analysis of the circuit. Run the simulation for four time periods and choose the step-size to be one-hundredth of the time period.

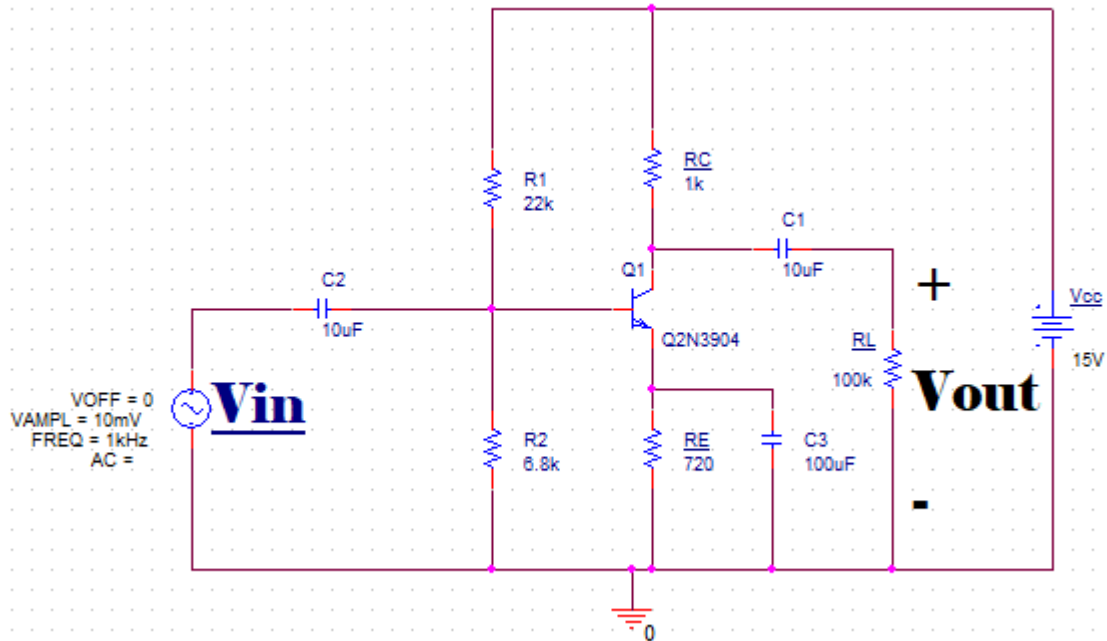


Figure 55: BJT Common Emitter Amplifier

Metal-Oxide Semiconductor Field-Effect Transistors (MOSFET)

MOSFET is the class of field-effect transistors that are *voltage-controlled current* devices. MOSFET has three terminals: *Gate (G)*, *Source(S)*, and *Drain(D)*. There are two types of MOSFETs: *n-channel* (NMOS) and *p-channel* (PMOS). For field-effect transistors, voltage applied between the gate and source controls the flow of current between the drain and source. There is also a fourth connection in MOSFET which is called *body* or *substrate*. For an *n-channel* MOSFET, body is held at the most negative voltage in the circuit and for a *p-channel* MOSFET, it is held at the most positive voltage in the circuit. In order for the current to flow through the MOSFET channel, voltage applied between the gate and source should exceed a specific value called *threshold voltage* ($V_{GS} > V_{TR}$)

Based on the polarity of the threshold voltage, MOSFET can further be divided into four categories,

- Depletion Mode NMOS: $V_{TR} < 0$

- Enhancement Mode NMOS: $V_{TR} > 0$
- Depletion Mode PMOS: $V_{TR} > 0$
- Enhancement Mode PMOS: $V_{TR} < 0$

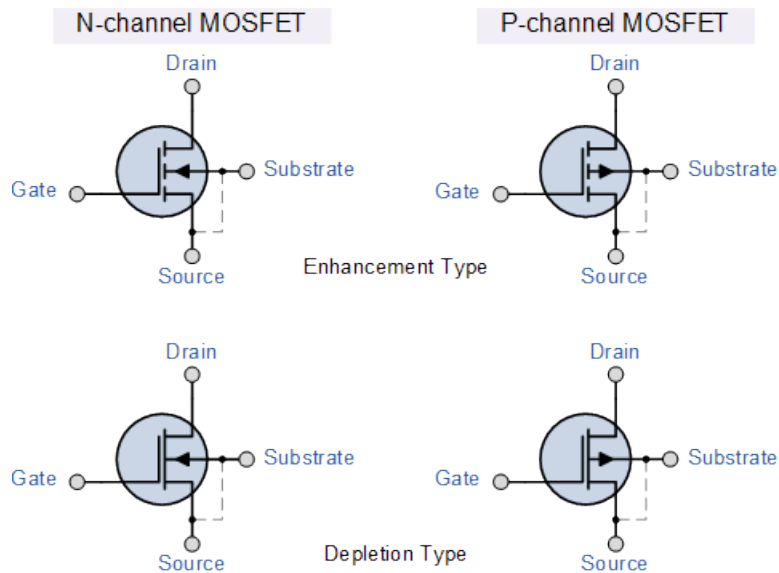


Figure 56: MOSFET Circuit Symbols

Like BJT, there are three regions of operation for a MOSFET. These regions are characterized as follows,

1. *Triode region*: $V_{DS} < V_{GS} - V_{TR}$ (current and voltage has square relationship)
2. *Constant Current or Saturation region*: $V_{DS} > V_{GS} - V_{TR}$ (current is constant in this region)
3. *Cut-off region*: $V_{GS} \leq V_{TR}$ (no current flows in this region)

Let's design a circuit with 2N7000 *enhancement mode n-channel MOSFET*. This MOSFET can be located in *PHIL_FET*, *FAIRCHILD* and *PWRMOS*.

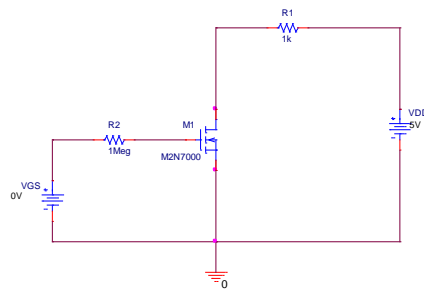
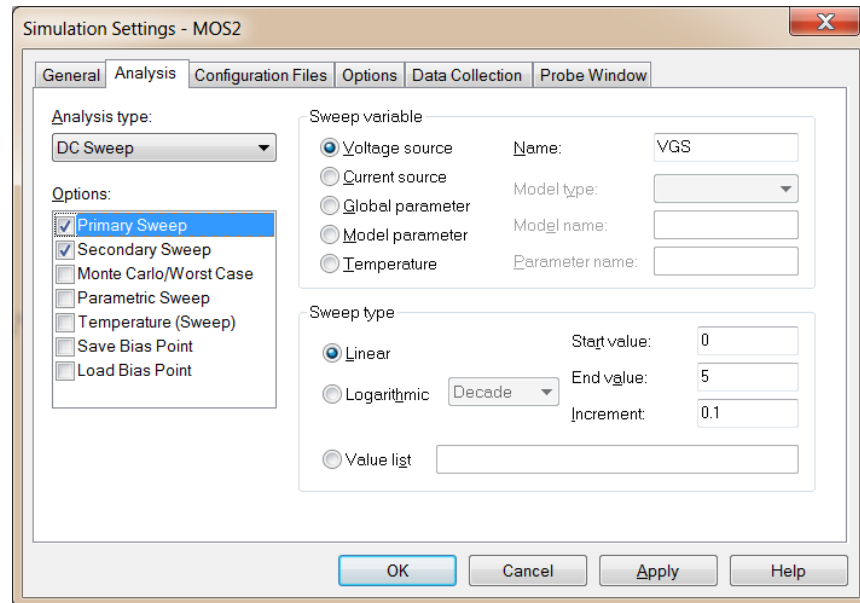
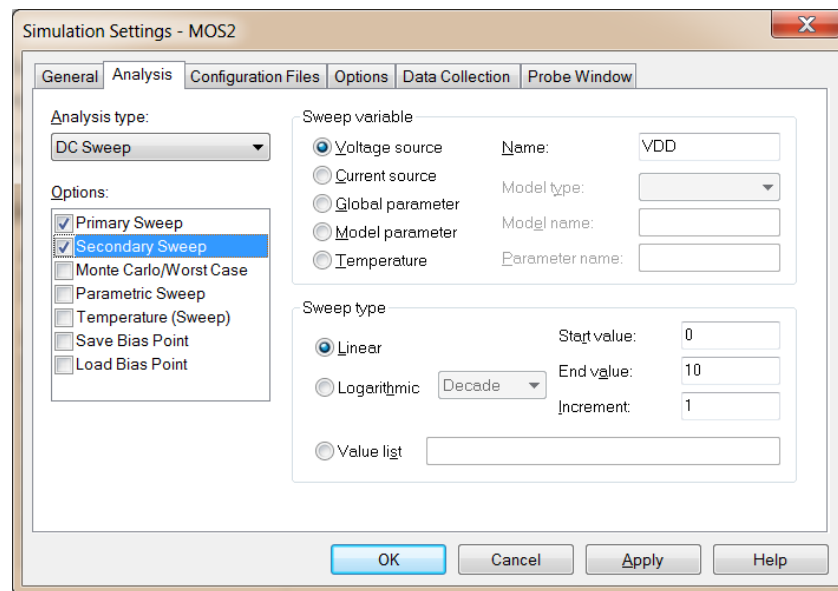


Figure 57: NMOS Circuit

Perform *DC Sweep* analysis and choose both *Primary Sweep* and *Secondary Sweep*. For *Primary Sweep* choose source *VGS* and for *Secondary sweep* choose *VDD*.



(a)



(b)

Figure 58: Primary Sweep and Secondary Sweep Parameters

Simulate your circuit to observe the following traces:

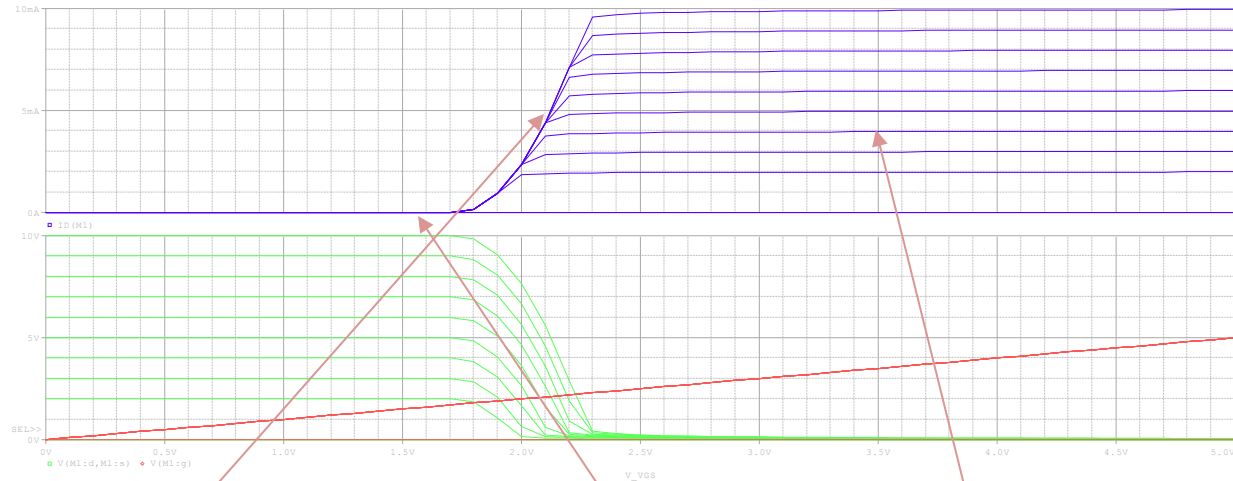


Figure 59: Current and Voltages in the Circuit

In figure 59, top set of the characteristic curves is current flowing from *drain* to *source*. When there is no current, that region corresponds to the *cutoff region*, curved region corresponds to the *ohmic or triode region*, and when current becomes constant, this is the *saturation region*. Bottom set of characteristic curves (green) shows how values of V_{DS} are changing for each value of V_{GS} (red line)

NOTE: If your simulation gives error regarding *MOSFET in the simulation output file (component is undefined)*, go to the simulation editor, select *configuration files* tab and add the library that you are using for the *MOSFET* (figure 60). This library will be located in "...\\Cadence\\SPB_16.5\\tools\\PSpice\\Library". For example, if you are using MOSFET M2N7000 from *pwrmos* library, select this library under *Filename* and press *Add to Design*.

Note that this screenshot is from PSpice version 16.5. For version 17.2, use the appropriate library path.

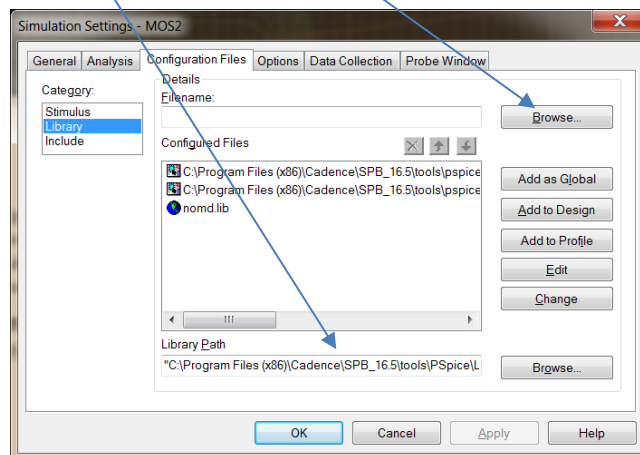


Figure 60: Addition of Libraries to the Simulation Editor

EXERCISE– 12:

A very basic electronic circuit is an inverter. An inverter is a circuit that reverses the input at its output, i.e., if input is high, output is low and vice-versa. Inverter basically comes under the category of digital circuits that are discussed in the next section. Most of the practical inverters are made out of *complementary MOSFET (CMOS)* circuits. *CMOS* circuits are the ones that employ both NMOS and PMOS in a specific configuration.

Design the following *CMOS* inverter circuit and perform DC sweep analysis at output (v_{out}). Sweep the input voltage source (only primary sweep) from 0V to 5V and trace the output voltage. *Be very careful to connect the drain of PMOS to the drain of NMOS, as shown by arrows. You will have to flip the PMOS vertically to connect both drains.*

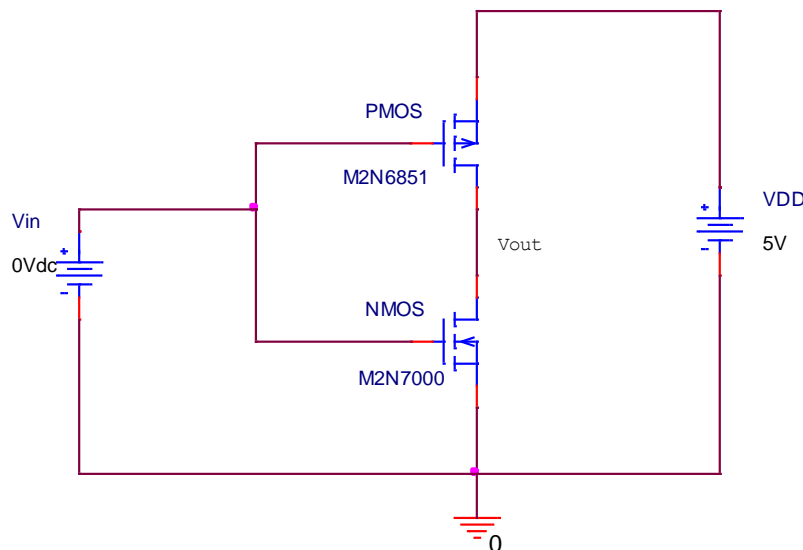


Figure 61: CMOS Inverter

DIGITAL CIRCUITS

In this part of the tutorial, we will concentrate on digital gates and digital circuits. Digital circuits can take *two* types of inputs, either *high (1)* or *low (0)*, and can generate three types of outputs, *high*, *low*, or *floating*. *Floating* means that the output logic level is neither high nor low. For every gate or IC, high and low levels correspond to certain voltages. For example, it could mean that any voltage between 3.5V to 5V will be considered as *high* level or 1, and any voltage from 0.2V to 1V will be considered as *low* level or 0. Any other voltage will be considered as *floating* also known as *high-Z* or *high-impedance* state.

For most of the gates and ICs, input high level is supplied through a 5V DC supply and low level is ground or zero voltage. Let's start our analysis with basic digital gates. There are *three* basic gates including *NOT* or *INVERTER*, *AND*, and *OR*. From these basic gates, four more gates can be derived: *NOR*, *NAND*, *EXOR* or *XOR*, and *EXNOR* or *XNOR*. Circuit symbols for all gates are shown in *figure 62*

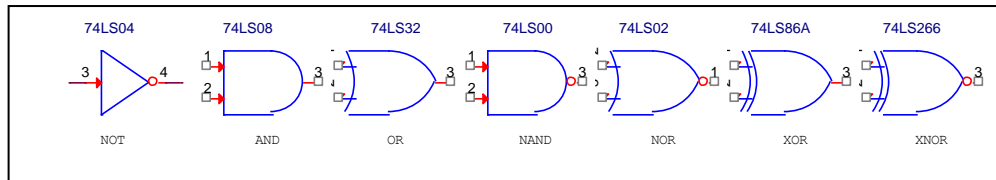


Figure 62: Logic Gates' Circuit Symbols

Following is the description of how these gates work,

- *NOT*: Output is opposite of the input
- *AND*: Output is high only if all the inputs are high
- *OR*: Output is low only if all the inputs are low
- *NAND*: Opposite of *AND*. Output is low only if all the inputs are high
- *NOR*: Opposite of *OR*. Output is high only if all the inputs are low
- *XOR*: Output is high if both the inputs are opposite
- *XNOR*: Output is high if both inputs are same

Simulation of digital gates and circuits is quite different from analog circuits. All gates are present in a number of libraries. For this tutorial, *74ls* library has been used. There are three types of signal sources we are going to use to give high or low inputs: *HI*, *LOW*, and *DigStim1*. Two sources are present in the *Place Ground* library that we use to place ground in every circuit. If your input is a constant high signal, you can use *\$D_HI/SOURCE* from the *Ground* library (*Place* → *Ground*) and if it is a constant low then you can use *\$D_LO/SOURCE*. If *HI* and *LOW* signals are not present in the *Ground* library, you may be missing *Source* library there. Click on *Add Library* button and from the *pspice* folder, add the *Source* library. Now you should have both *HI* and *LOW* signals available. If you want to use a square waveform (clock) signal or want to define your own signal, you will have to use *DigStim1* from *SOURCESTM* library.

Let's create a simple digital circuit to analyze a *NOT* gate. Get *74LS04* from *74LS* library and *DigStim1* from *SOURCESTM* library and put them together as shown in *figure 63*.



Figure 63: NOT Gate Analysis

DSTM1 stands for *Digital Stimulus for single input*. Implementation is the name of signal to be defined and used by the stimulus. Let's define two signals that can be used by *DSTM1*. Select *DSTM1* by left mouse click and then right click your mouse to select *Edit PSpice Stimulus*. Stimulus Editor window will appear and *New Stimulus* dialog box will pop up. Choose a name for your stimulus. This name will be used under *Implementation* if you want to use this stimulus. First, let's define a clock signal by choosing *CLOCK* in the dialog box and pressing *OK*.

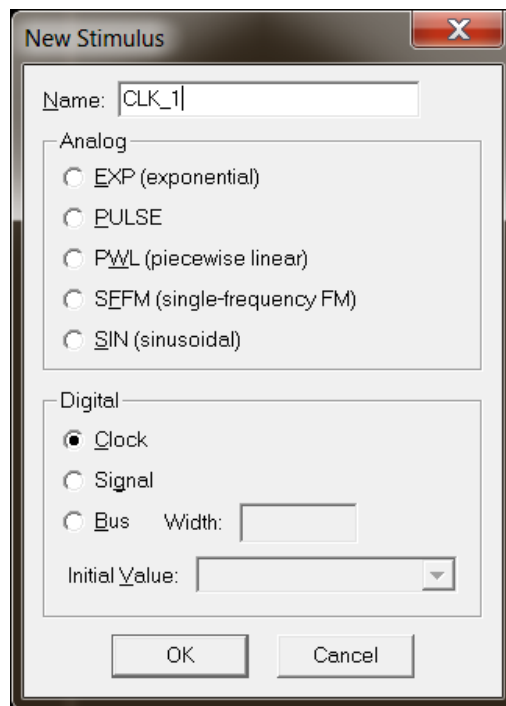


Figure 64: New Stimulus Dialog Box

When you press *OK*, *clock attributes* dialog box will pop up. Choose frequency of the clock. Keep *duty cycle* to be 0.5 to have a symmetric clock, i.e., half of the time low and half of the time high.

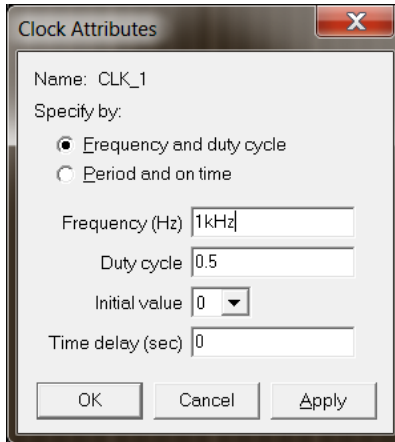


Figure 65: Clock Attributes Dialog Box

You can set the length of your signal by changing *Axis Setting* under *Plot* menu. The default length is 4ms

Now let's define one of our own signals that can go high and low at any time that we want. This is a random or asynchronous input signal. From *Stimulus* menu choose *New*, *New Stimulus* dialog box will pop up. Give a name to your stimulus and choose *Signal* under *Digital* (instead of Clock) and press *OK*. You will see a new line will appear in the stimulus editor window. The initial value of this signal will either be zero or one based on your choice. From *toolbar* menu, choose the last button from left that shows *high-low* signal. Once you press this button, your mouse will change into a *pencil*. Put this pencil on different parts of your signal and click it to change level of your signal (*Observe SIG_1 in figure 66*).

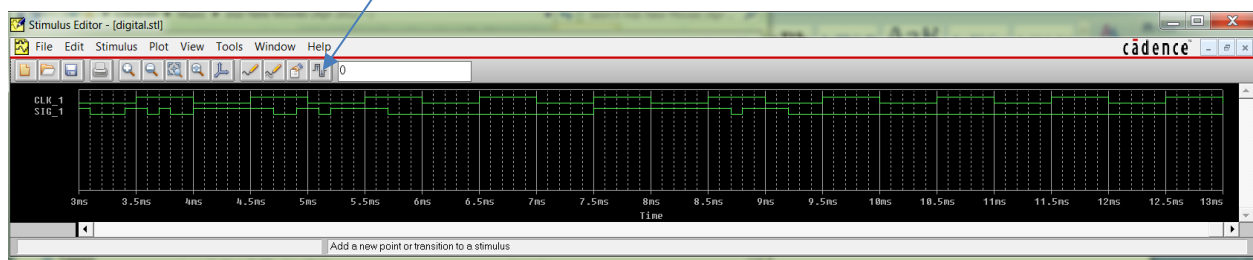


Figure 66: Defined Signals for DSTM1

Once you are done with defining your signal, press *Esc* to get out of the signal drawing mode.

Let's simulate our circuit with both of our stimuli one by one. Let's choose *CLK_1* as our input signal first. Name your *Implementation* field *CLK_1*. Create a new simulation profile and choose transient analysis. Run transient analysis for the length of time that you have define the signal. Place voltage probe at the output of stimulus and the output of gate.

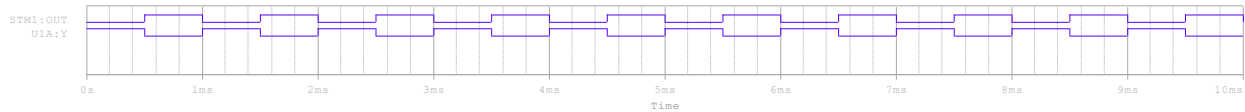


Figure 67: Input and Output of NOT gate for CLK_1 Signal

Now let's use signal SIG_1 as our input to the NOT gate. Change name of your *Implementation* field to SIG_1 and perform transient analysis again.

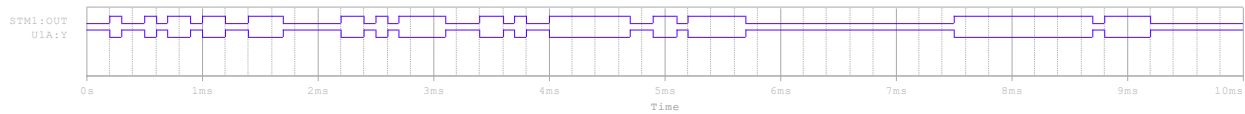


Figure 68: Input and Output of NOT gate for SIG_1 Signal

NOTE:

- (i) If your simulation does not run and you get an error “Subcircuit 74LS04 used by X_U1A is undefined” in the output file, make sure to add 74ls library under the *Configuration* tab of the *Simulation Settings* window. If you still get simulation error and in your output file you see, “Model IO_LS used by X_U1A.U1 is undefined”, add dig_io library under the *configuration* tab of the *Simulation Settings* window. Addition of libraries is explained on page 42.
- (ii) If you see red lines in your simulation window instead of green signal lines, check if your stimulus file is present under the *configuration* tab (*configuration* → *category* → *stimulus*). If it is not there, add it from your current schematic folder.

EXERCISE – 13

Simulate an XOR gate, as shown in figure 69, with one of its inputs having clock with frequency 1KHz and other with a constant *high* signal.

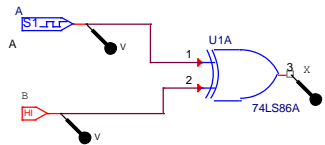


Figure 69: Testing an XOR Gate

EXERCISE – 14

Design the logic circuit shown in *figure 71*. Choose *A* to be a clock signal with frequency 1KHz, and *B* and *C* as your own random signals (some random 1s and 0s). Simulate it for transient analysis for 4ms. Keep maximum step size to be 0.01ms.

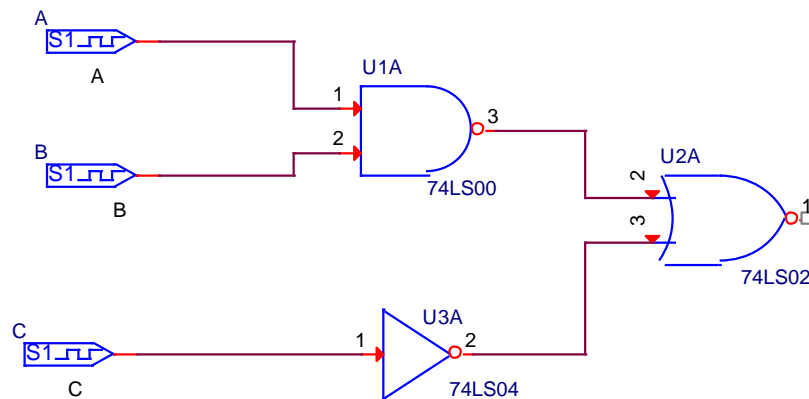


Figure 70: Logic Circuit for Exercise – 14