Contents

Chapter 1

Introduction

Electronic systems are an integral part of human life. They have simplified our lives to a great extent. Starting from small systems made of a few discrete components to the present day integrated circuits (ICs) with millions of logic gates, electronic systems have undergone a sea change. As a result, design of electronic systems too have become extremely difficult and time consuming. Thanks to a host of computer aided design tools, we have been able to come up with quick and efficient designs. These tools are called `Electronic Design Automation` or `EDA` tools.

1.1 EDA design flow

Let us see the steps involved in EDA. In the first stage, the specifications of the system are laid out. These specifications are then converted to a design. The design could in the form of a circuit schematic, logical description using an HDL language etc. The design is then simulated and re-designed, if needed, to achieve the desired results. Once simulation achieves the specifications, the design is either converted to a PCB, a chip layout, or ported to an FPGA etc. The final product is again tested for specifications. The whole cycle is repeated until desired results are obtained.

1.2 EDA Tools

If you are building an electronic system, you would first design your circuit, make its schematic diagram, simulate it and finally convert it into a Printed

Circuit Board (PCB) . There are various tools available that would help you do this. Some of the popular EDA tools are those of `Cadence`, `Synopys`, `Mentor Graphics`, `Xilinx` etc. These are proprietary tools. There are some open source EDA tools like `gEDA`, `KiCad`, `Ngspice` etc.

1.3 What is Oscad?

Oscad is a free and open source EDA tool. It is an acronym for **O**pen **s**ource **c**omputer **a**ided **d**esign. Oscad is created using several open source software packages namely KiCad, Ngspice, Scilab and Python. Using Oscad, one can create circuit schematics, perform simulation and design PCB layouts. It can create or edit new device models, and create or edit subcircuits for simulation. It also has a Scilab based Mini Circuit Simulator (SMCSim) which is capable of giving the circuit equations for each simulation step. This feature is unique to Oscad.

1.4 Why Oscad?

Proprietary EDA tools are fairly comprehensive and high end. But their licences are very expensive. The main drawback of the open source tools is that they are not comprehensive. Some of them are capable of PCB design (e.g., `KiCad`) while some of them are capable of performing simulations (e.g., `gEDA`). There is no well known open source software that can perform circuit design, layout design and circuit simulation together. Oscad is capable of doing all of the above. This is why Oscad is very important to students, teachers and other professionals who would want to study and/or design electronic systems. Oscad is also very useful for entrepreneurs and small scale enterprises who do not have the capability to invest on heavily priced proprietary tools.

1.5 Structure of the book

This book introduces Oscad to the reader and illustrates all the features of Oscad with examples. Chapter **??** gives step by step instructions to install Oscad on your computer. The software architecture of Oscad is presented in Chapter **??**. Chapter **??** gets you started with Oscad. It takes you through

a tour of Oscad with the help of a simple RC circuit example. Chapter 5 explains how to create circuit schematics using Oscad, in detail using examples. Chapter 6 illustrates how to simulate circuits using Oscad. Chapter **??** explains PCB design using Oscad, in detail. The advanced feature of Oscad like Model builder, subcircuit builder and Scilab based simulations are covered in the Apter **??**. Chapter **??** describes the spoken tutorials on Oscad and contains instructions to use them. Appendix A presents examples from the book `Microelectronic Circuits` by Sedra and Smith that have been worked out using Oscad.

Chapter 2

Instructions for Installing and Setting Up Oscad

The step by step instruction to install Oscad is given below.
Before starting the installation, first you need to meet system requirements and installation requirement. Installation script for Oscad is written in Bash. This makes the installation efficient and user friendly. It is expected to have basic knowledge on Linux commands

System Requirements:

- Ubuntu 12.04 OS(64bit/32bit)
- Oscad
- Scilab 5.4.1

Installation Requirements:

- Working Internet connection
- Require to be an admin user to do the installation

Prerequisites:

- Basic knowledge of analog and digital electronics

- Basic knowledge of Linux shell commands
- Requires Synaptic Package Manager

2.1 Procedure for installing Oscad

2.1.1 Step 1: Download Oscad and examples

Go to **http://www.oscad.net/downloads**
Figure 2.1 is shown below.

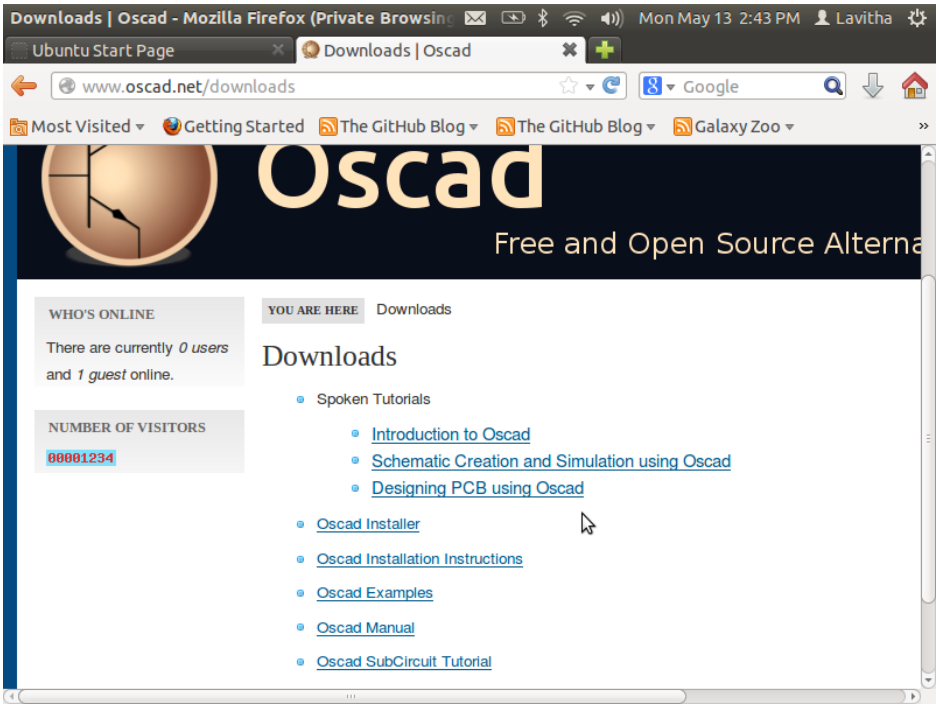

Figure 2.1: Oscad Website

- (a)Click on **Oscad Installer** and save the file in a folder

- (b)Again click on **Oscad Examples** and save the file in same folder

2.1.2 Step 2: Check whether machine is 32bits or 64bits

Go to Terminal and type
uname -m
and press enter
If the output is **i686** then machine is 32 bits and if output is **x86_64** then the machine is 64 bits.
Screen shot 2.2 shows that machine is 32 bits.

Figure 2.2: Terminal

2.1.3 Step 3: Download Scilab

Go to: **http://www.scilab.org/download/5.4.1** and download the scilab 5.4.1 under GNU/Linux section in the same folder where Oscad and Examples were saved
Download scilab for 32 bits or 64 bits depending on the OS.
Figure 2.3 shows the options available under GNU/Linux

Figure 2.3: scilab

2.1.4 Step 4: Extracting

Go to folder where all the three folders are saved and do **extract here**

2.1.5 Step 5: Navigate to Folder

Go to folder where all the three folders are saved
For this go to terminal and type:
Note: Terminal commands are shown in box
cd folder-where-downloaded-files-are-saved
Press Enter
Now check whether we have navigted to desired folder. To check please type:

ls
Press Enter
Screen shot 2.4 is shown below:

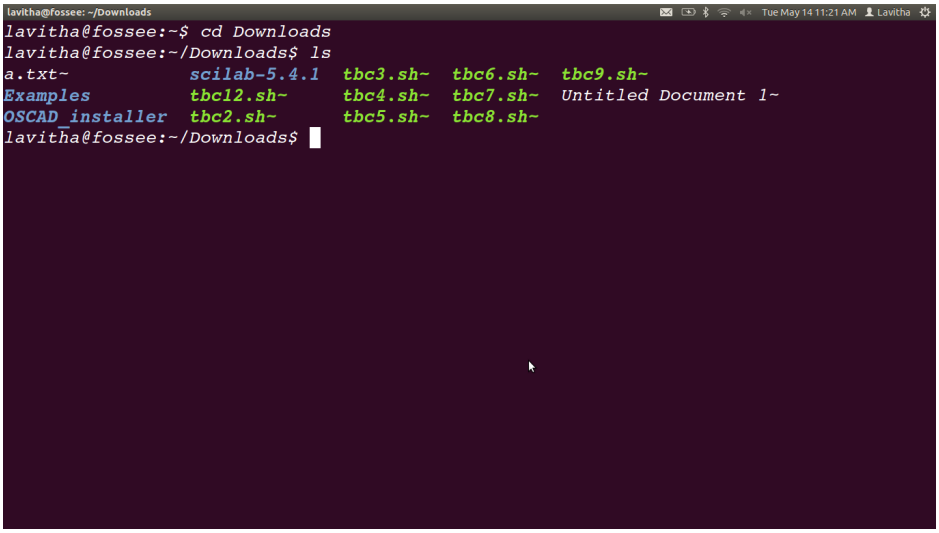

Figure 2.4: Terminal

2.1.6 Step 6: Navigate to OSCAD_Installer

Now navigate to OSCAD_installer, To do this type:
cd OSCAD_installer
Press Enter

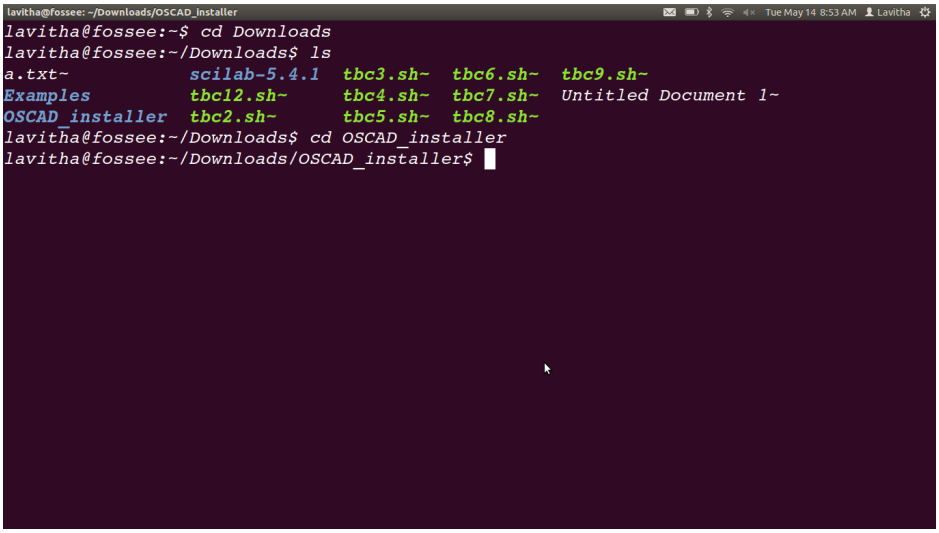

Figure 2.5: Terminal

2.1.7 Step 7: Make the installOSCAD and installModule files executable

To do this go to terminal and Type:
sudo chmod 755 installOSCAD.sh installModule.sh
Press Enter
Type the sudo(root) password
The Terminal should look like below as shown in figure 2.6

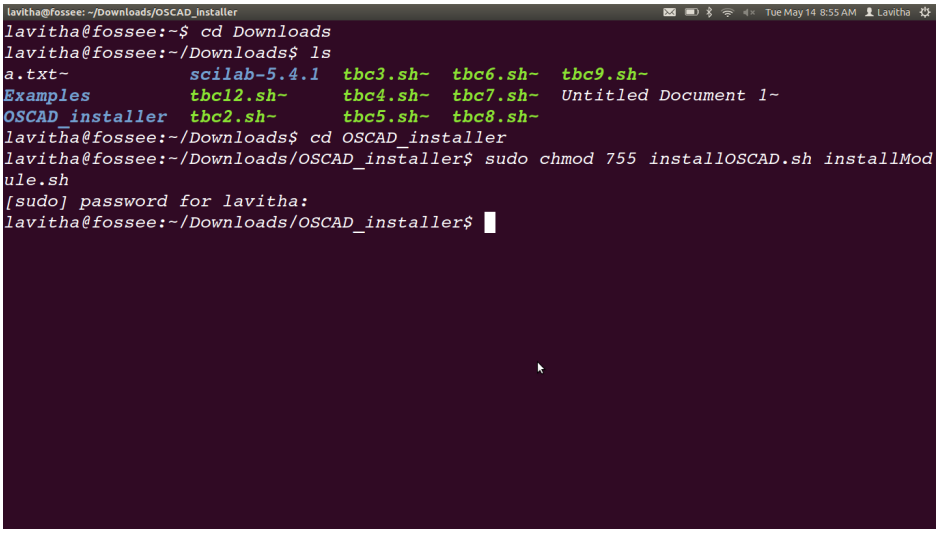

Figure 2.6: Terminal: To make executable

2.1.8 Step 8: Begin installation

To begin installation, on the Terminal type
./installOscad.sh
Press Enter
The Terminal prompts for proxy setting as shown in 2.7

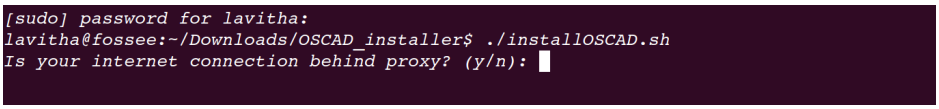

Figure 2.7: Terminal: Installation

Enter the related proxy details as shown in figure 2.8
Now the prompt displays message **Do you want to continue [Y/n]:**
Type 'y' and press Enter. Kicad, ngspice and necessary python modules will be installed.
Note: While installing the python modules, you may get some error messages. In that case, install the missing packages using Synaptic Package Manager. Once this is done, re-run the installOscad.sh script.

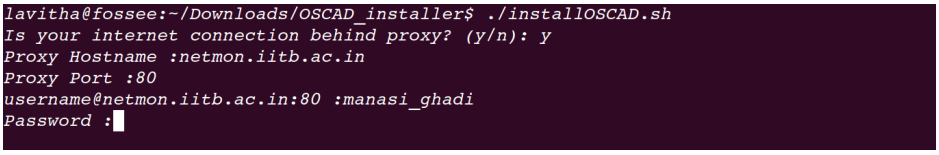

Figure 2.8: Terminal: Proxy

2.1.9 Step 9: Linking of Scilab

The prompt displays the message:
Do you have scilab5.4 or above? (y/n)

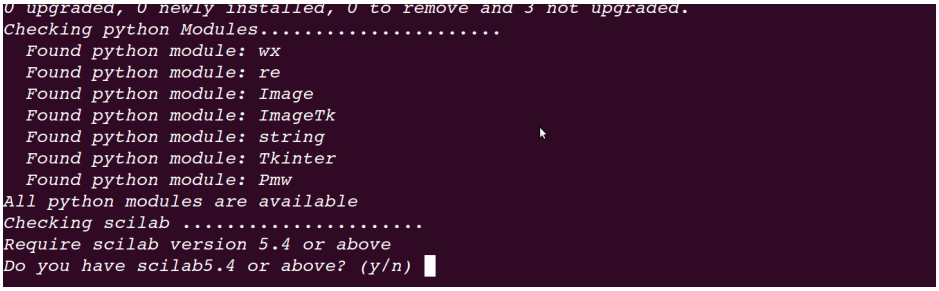

Figure 2.9: Terminal: To make executable

Type: 'y' and press Enter
Now give the complete path: on Terminal:
Example: **/home/lavitha/downloads/scilab-5.4.1**
The screen shot is shown below:
and Press enter

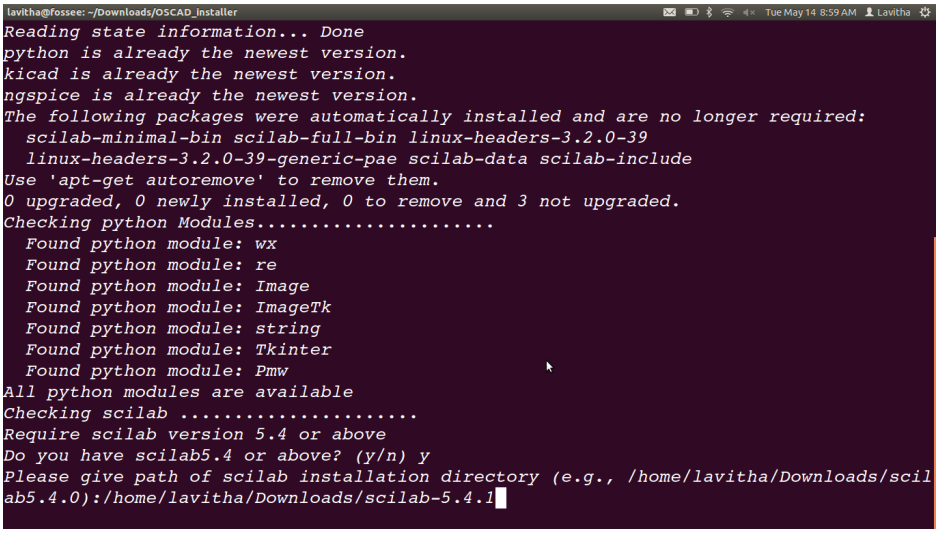

Figure 2.10: Terminal: Scilab instllation

The **metanet toolbox** is installed and this take couple of minutes

2.1.10 Step 10: Final installation

The prompt displays the message
Please select installation directory
Select the desired location and press Enter
For example:
/home/lavitha
Press Enter
If everything is installed then prompt displays message
installation completed
This creates textbfOscad shortcut on Desktop.

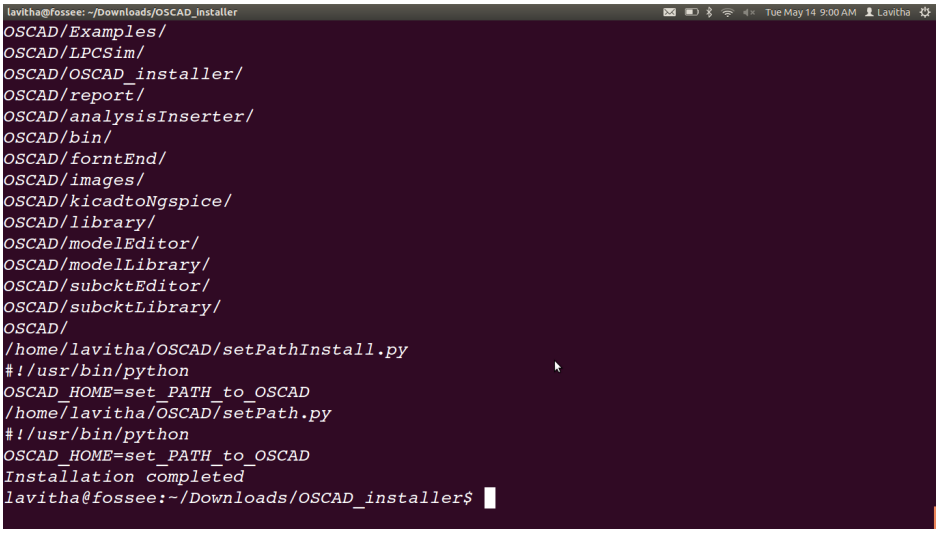

Figure 2.11: Terminal: Installation Completed

2.1.11 Step 11: Testing

- Double click on Oscad shortcut created on Desktop as shown below:

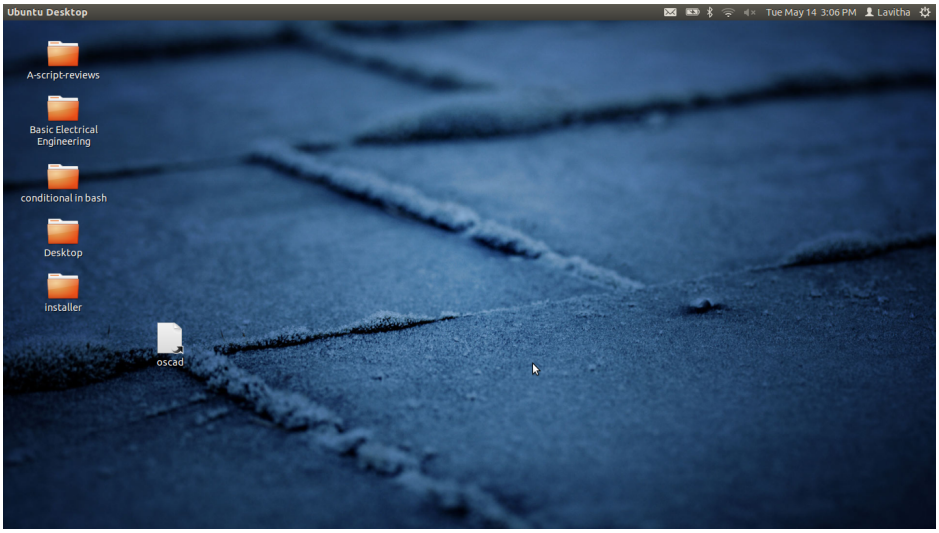

Figure 2.12: Desktop: Oscad shortcut

- A tab is displayed as shown below. select **Run option**

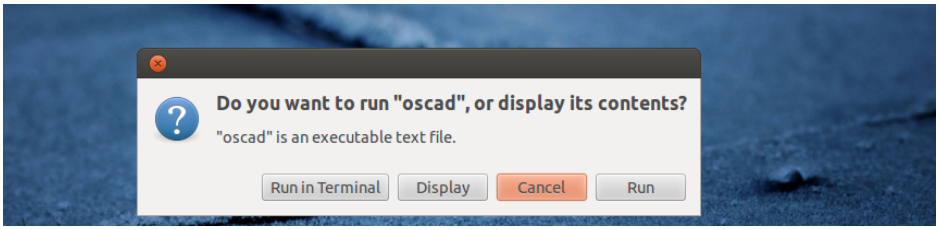

Figure 2.13: Desktop: Oscad

- It opens the Oscad window as show below:

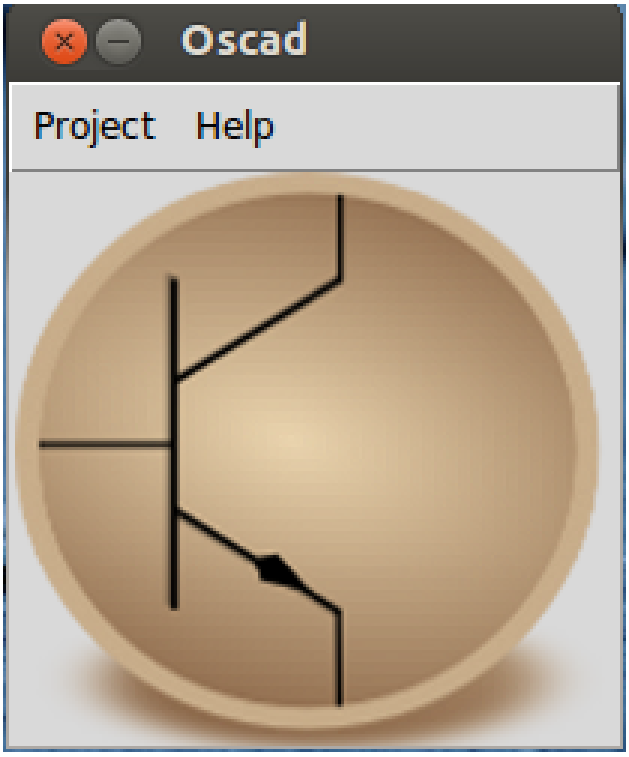

Figure 2.14: Desktop: Oscad

- select **project** and then **open** Browse the folder where examples is saved
- Double click on Examples
- select **RC** from the examples by double clicking on RC

Chapter 3

Architecture Of Oscad

Oscad is a CAD tool which provides the ease of testing circuits for electronic system designers. But the important feature of this tool is that it is Open Source and hence the user can check,modify the source as per their need. The software provides a generic, modular and extensible platform for experiment with electronic circuits. This software runs on all UNIX platforms and also with python, Kicad, Ngspice and Scilab 5.4 or above.

The objective behind the development of Oscad is to provide an open source solution for electronics and electrical engineers. The software should be capable of performing Circuit Schematic, PCB Design and Circuit Simulation (Analog, Digital and Mix-signal). It should provide facilities to create new models and components. In addition to this, it should have the capability to explain the circuit by giving symbolic equations and numerical values. Keeping these objectives in mind, we have designed the architecture of Oscad.

Keeping the above features in mind, the plan was to

- Integrate existing open source tools for Circuit Design and drawing, PCB Layout, Circuit Simulation.
- Create additional modules if required.
- Create user friendly graphical interface.

3.1 Modules used in Oscad:

We have used different open-source tools for the underlying build-up of Oscad. In this section we will give a brief idea about all packages.

3.1.1 Eeschema – Schematic Editor

Eeschema is an integrated software where all functions of circuit drawing, control, layout, library management and access to the PCB design software are carried out within itself. Eeschema is intended to work with printed circuit software such as Pcbnew. It can provide the netlist file, which describes the electrical connections of the PCB. Eeschema also integrates a component editor which allows the creation, editing and visualization of components. It also allows the user to effectively handle of the symbol libraries i.e; import, export, addition and deletion of library components).

Eeschema also integrates the following additional but essential functions needed for a modern schematic capture software:

- Design rules check (DRC) for the automatic control of incorrect connections and inputs of components left unconnected.
- Generation of layout files in POSTSCRIPT or HPGL format.
- Generation of layout files printable via printer.
- Bill of Material generation.
- Netlist generation for PCB layout or for simulation.

3.1.2 CvPCB – Component-Footprint mapper

CvPcb is a tool that allows user to associate components in the schematic to component footprints used when laying out the printed circuit board. Typically the netlist file generated by Eeschema does not specify which printed circuit board footprint is associated with each component in the schematic. Although this is not always the case as component footprints can be associated during schematic capture by setting the component's footprint field. CvPcb provides a convenient method of associating footprints to components. It provides footprint list filtering, footprint viewing, and 3D component model viewing to help ensure the correct footprint is associated to each

component. Components can be assigned to their corresponding footprints manually or automatically by creating equivalence files. Equivalence files are look up tables associating each component with it's footprint. This interactive approach is simpler and less error prone than directly associating the footprints in the schematic editor because as well as allowing for automatic association, CvPcb allows you to see the list of available footprints available and display them on the screen to ensure you are associating the correct footprint.

3.1.3 Pcbnew – PCB Layout Editor

Pcbnew is a powerful printed circuit board software tool. It is used in association with the schematic capture software program Eeschema, which provides the netlist file - this describes the electrical connections of the PCB to design. CvPcb is used to assign each component in the Netlist produced by Eeschema, to a module that is used by Pcbnew.

- It manages libraries of modules. Each module is a drawing of the physical component including its footprint - the layout of pads providing connections to the component. The required modules are automatically loaded during the reading of the netlist produced by CvPcb.
- Pcbnew integrates automatically and immediately any circuit modification, by removal of any erroneous tracks, addition of the new components, or by modifying any value (and under certain conditions any reference) of the old or new modules, according to the electrical connections appearing in the scheme.
- This tool provides a rats nest display, a hairline connecting the pads of modules which are connected on the schematic. These connections move dynamically as track and module movements are made.
- It has an active Design Rules Check (DRC) which automatically indicates any error of track layout in real time.
- It can automatically generates a copper plane, with or without thermal breaks on the pads.
- It has a simple but effective autorouter to assist in the production of the circuit. An Export/Import in `SPECCTRA` dsn format allows to use more advanced auto-routers.

- This provides options specifically for the production of ultra high frequency circuits (such as pads of trapezoidal and complex form, automatic layout of coils on the printed circuit...).
- Pcbnew displays the elements (tracks, pads, texts, drawings and more) as actual size and according to personal preferences:
  - display in full or outline.
  - display of the track/pad clearance.

3.1.4 Kicad to Ngspice netlist converter

It converts Kicad generated netlists to ngspice compatible format.It has the capability to

- Insert parameters for fictitious components
- Convert IC into discrete blocks
- Insert D-A and A-D converter at appropriate place
- Insert plotting and printing statement in netlist
- Find current through all components

3.1.5 Analysis Inserter

This feature helps the user to perform different type of analysis such as Operating point analysis, DC analysis, AC analysis, transient analysis etc.It has the facility to

- Insert type of analysis
- Option of analysis
- Option of simulator

3.1.6 Component Model Builder

- This tool provides a facility to define a new model for devices such as
  - Diode
  - Bipolar Junction Transistor (BJT)
  - Metal Oxide Semiconductor (MOS)
  - Junction Field Effect Transistor (JFET)
  - IGBT
  - Magnetic core
- Provides facility to edit existing model.
- Provides help related to model parameter.

3.1.7 Component Sub-circuit Builder

This feature allows the user to create a sub-circuit for a component. Once the sub-circuit for a component is created, the user can use it for different circuits. It has the facility of

- Provides facility to define new components such as
  - Op-amp
  - IC-555
- Provides facility to edit existing sub-circuit
- Provides help related to components parameters

3.1.8 Circuit Simulator – Ngspice

Ngspice is a general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analyses. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent voltage and current sources, four types of dependent sources, lossless and lossy transmission lines (two separate implementations), switches, uniform distributed RC lines, and the five most common semiconductor devices: diodes, BJTs, JFETs, MESFETs, and MOSFETs.

3.1.9 Scilab based circuit simulator – SMCSim

The feature of SMCSim is that it gives the system of equations for the circuit under test. The SMCSim works in three modes: normal, symbolic and numerical mode. Details about this will be expalined in later chapters.

3.2 Workflow of Oscad:

Fig. 1 shows a block diagram of Oscad. The block diagram consists mainly three parts:

- Schematic Editor
- PCB Layout Editor
- Circuit Simulators

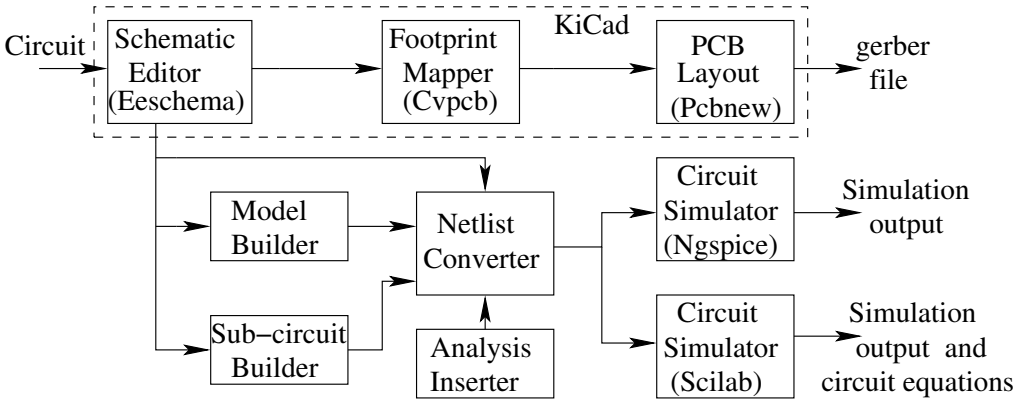

Figure 3.1: Block Diagram of Oscad

Here, we explain the functionality of each block to design electronic systems.

The circuit design is the first step of design of an electronic circuit. Generally a circuit diagram is drawn on a paper, and then entered into a computer using a schematic editor. We have used Eeschema as a schematic editor for Oscad. Thus all the functionalities of Eeschema are naturally available in Oscad.

Using the functionalities of Eeschema, we have created separate libraries for components supported by ngspice ex-plicitly or implicitly. As Eeschema is originally intended for PCB Design, there are no fictitious components5 such as voltage or current sources. Thus, we have added a new library for different types of voltage and current sources such as sine, pulse, square wave, etc. We have also built a library which gives a functionality of printing and plotting solution.

The schematic editor provides a netlist file, which describes the electrical connections of the PCB to a design. In order to create a PCB layout, physical components are required to map into their footprints. To create component to footprint mapping, we have used Cvpcb software which is a part of KiCad . For additional components in newly defined library, we have assigned or created footprints. We have used Pcbnew software to draw a PCB layout.

After designing a circuit, it is essential to check the integrity of the circuit design. In case of large size electronic circuits, breadboard testing is impractical. Therefore electronic system designers rely heavily on simulation.

The accuracy of the simulation results can be increased by accurate modeling of the circuit elements. Model Builder, provides a facility to define a new model for devices and edit existing models. Complex circuit elements can be created by hierarchical modeling. Sub-circuit Builder provides an easy way to create a sub-circuit. The schematic editor provides a netlist file, which describes the electrical connections between circuit components. But it cannot be directly used for simulation due to compatibility issues. Netlist Converter, denoted converts the netlist into a Ngspice compatible netlist. The type of simulation to be performed using the netlist and the corresponding options are provided through a graphical user interface (GUI). We call this as the Analysis Inserter.

We have used Ngspice for mixed-level/mixed-signal circuit simulation. Ngspice is based on three open source software packages[1]:

- Spice3f5 (analog circuit simulator)
- Cider1b1 (couples Spice3f5 circuit simulator to DSIM device simulator)
- Xspice (code modeling support and simulation of digital components through an event driven algorithm)

It is a part of gEDA project. Ngspice is capable of simulating devices with BSIM, EKV, HICUM, HiSim, PSP, PTM models. It is widely used due to its accuracy even for latest technology devices.

In order to provide an explanation capability, we have also developed a Scilab based circuit simulation capability within OSCAD. It generates equations from the netlist and gets them solved by Scilab, which has many state of the art numerical methods built in. We have referred to this as Scilab based Mini Circuit Simulator (SMCSim).

Chapter 4

Getting Started

In this chapter we will get started with Oscad. We will run through the various options available on the with an example circuit. Refering to this chapter will make you familiarize with Oscad thereby helping you plan your project before actually designing a circuit. Lets get started.

4.1 Oscad GUI

After you finish installing Oscad, a shortcut icon will be created on your desktop. You will click on the link to launch oscad. Oscad main window will open up. It is shown in figure4.1. On the menu bar there are two options, Project and Help. To create a new project or open an existing project, use the project option.

Let us open an existing project. Cick on Project and select Open. A window will open asking where is the folder containing the file to be opend. This window is shown in figure4.2. Navigate it to the corresponding directory

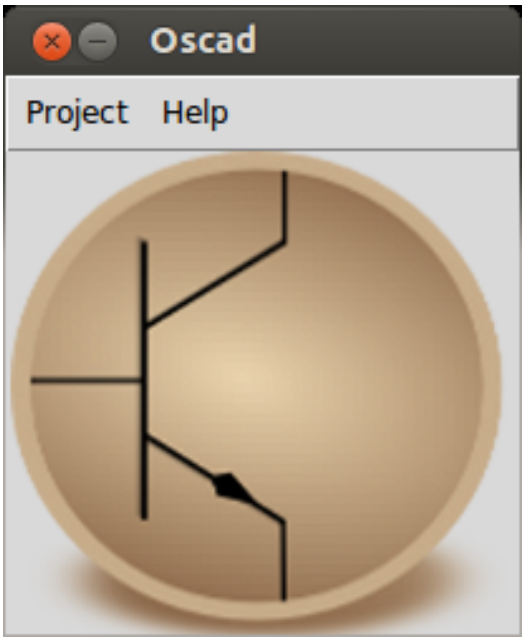

Figure 4.1: Oscad main window

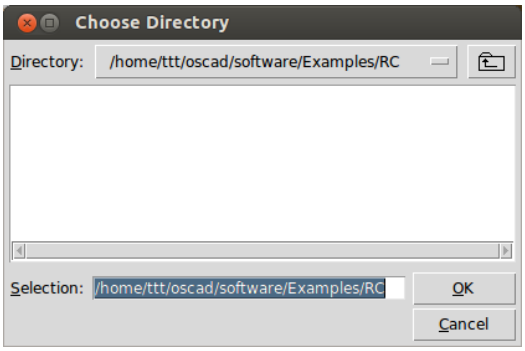

Figure 4.2: Open project directory

Chapter 5

Schematic Creation

The first step in the design of an electronic system is the design of its circuit. This circuit is usually created using a `Schematic Editor` and is called a `Schematic`. Oscad uses `EEschema` as its schematic editor. EEschema is the schematic editor of KiCad. It is a powerful schematic editor software. It allows the creation and modification of components and symbol libraries and supports multiple hierarchical layers of printed circuit design.

5.1 Familiarising the schematic editor interface

Figure 5.1 shows the schematic editor and the various menu and tool bars.

5.1.1 Top Menu Bar

Some of the important menu options in the top menu bar are:

1. File - The file menu items are given below:
   (a) New - Clear current schematic and start a new one
   (b) Open - Open a schematic
   (c) Open Recent - A list of recently opened files for loading
   (d) Save Whole Schematic project - Save current sheet and all its hierarchy.

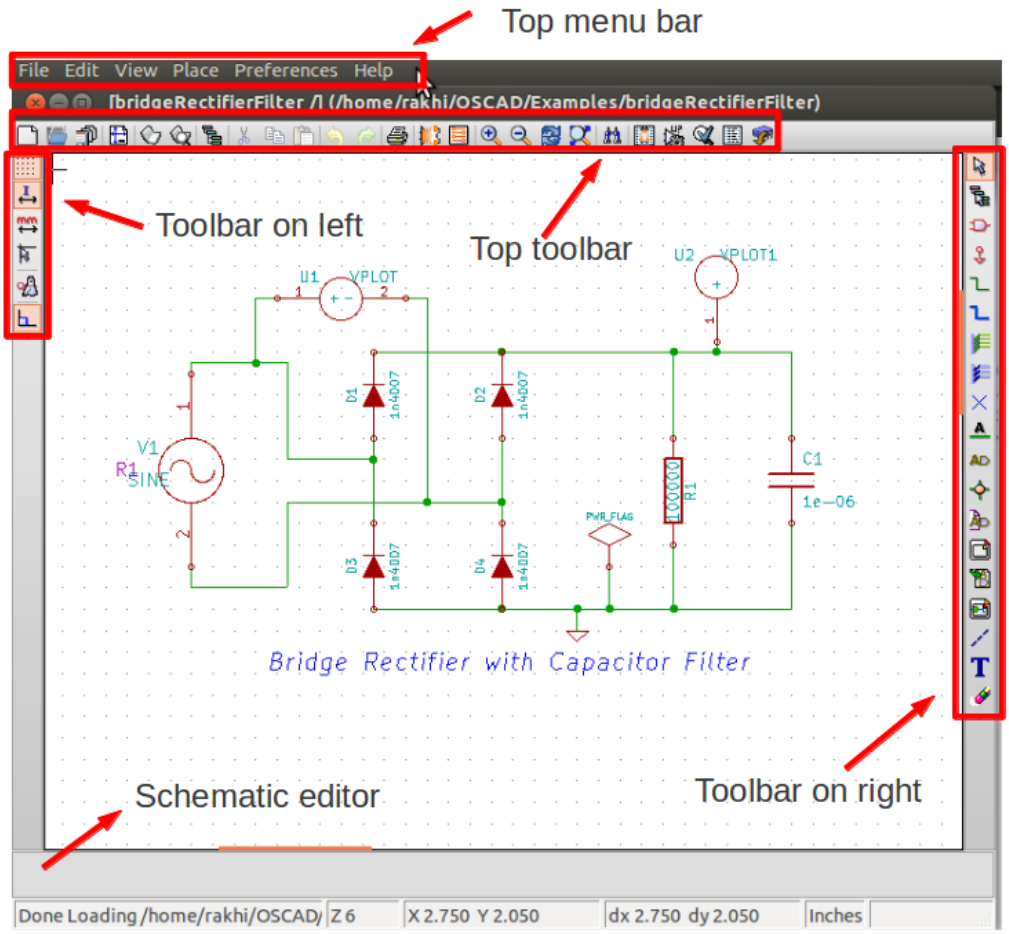

Figure 5.1: Schematic editor with the menu and tool bars shown

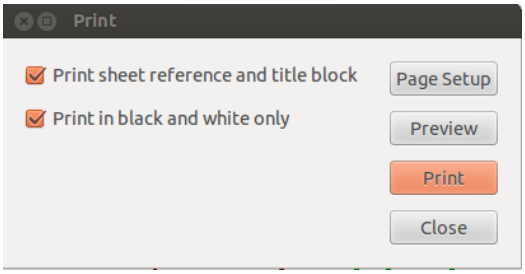

Figure 5.2: Print options

(e) Save Current Sheet Only - Save current sheet, but not others in a hierarchy.

(f) Save Current sheet as - Save current sheet with a new name.

(g) Print - Access to print menu (See Figure 5.2).

(h) Plot - Plot the schematic in Postscript, HPGL, SVF or DXF format

(i) Quit - Quit the schematic editor.

2. Place - The place menu is a short cut to placing various items like components, wire, junction etc. onto the schematic editor. See the section 5.2 to know more about various short cut keys (hotkeys).

3. Preferences - The preferences menu has the following options:

   (a) Library - Select libraries and library paths

   (b) Colors - Select colors for various items.

   (c) Options - Display schematic editor options (Units, Grid size).

   (d) Language - Shows the current list of translations. Use default.

   (e) Hotkeys - Access to the hot keys menu. See the section 5.2 about hotkeys.

   (f) Read preferences - Read configuration file.

   (g) Save preferences - Save configuration file.

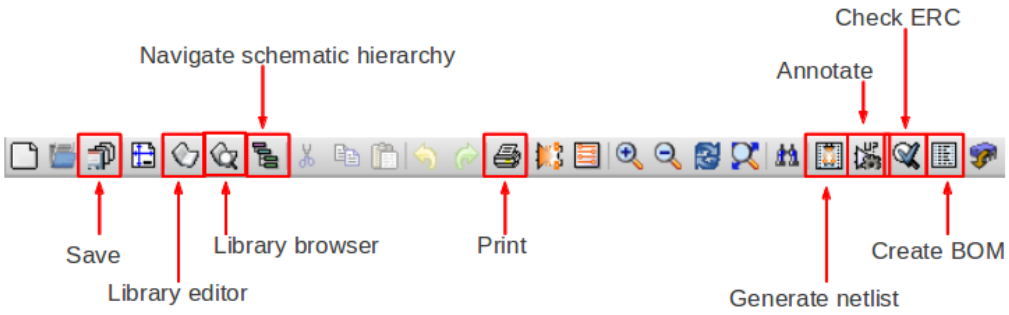

Figure 5.3: Toolbar on top - important tools

5.1.2 Top toolbar

Some of the important tools in the top toolbar are discussed below. They are marked in Figure 5.3

1. Save - Save your current schematic
2. Library Editor - Create or edit components. See section 5.3 for more details.
3. Library Browser - Browse through the various component libraries available
4. Navigate schematic hierarchy - Navigate between the root and sub-sheets in the hierarchy
5. Print - Print your schematic
6. Generate netlist - Generate a netlist for PCB design or for simulation.
7. Annotate - Annotate your schematic
8. Check ERC - Do Electric Rules Check for your schematic
9. Create BOM - Create a Bill of Materials of your schematic

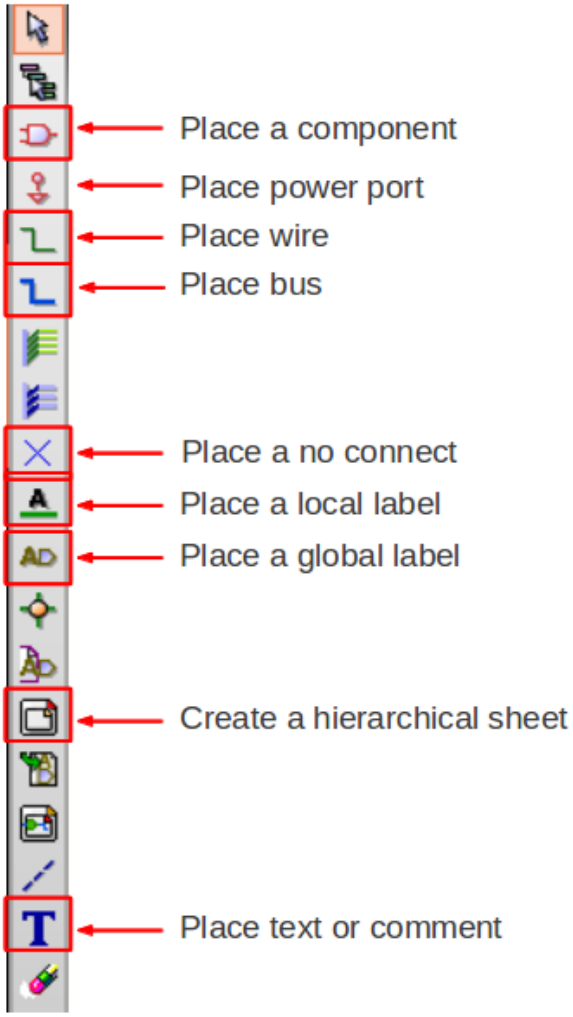

Figure 5.4: Toolbar on right - important tools

5.1.3 Toolbar on the right

The toolbar on the right side of the schematic editor has many important tools. Some of them are marked in Figure 5.4.

Let us now see these tools one by one.

1. Place a component - Load a component to your schematic. See section 5.3 for more details.
2. Place a power port - Load a power port (Vcc, ground) to your schematic
3. Place wire - Draw wires to connect components in schematic
4. Place bus - Place a bus on your schematic
5. Place a no connect - Place a no connect flag, particularly useful in ICs
6. Place a local label - Place a label or node name which is local to the schematic
7. Place a global label - Place a global label (these are connected across all schematic diagrams in the hierarchy)
8. Create a hierarchical sheet - Create a sub-sheet with in the root sheet in the hierarchy. Hierarchical schematics are a good solution for big projects
9. Place a text or comment - Place a text or comment in your schematic

5.1.4 Toolbar on the left

Some of the important tools in the toolbar on the left are discussed below. They are marked in Figure 5.5

1. Show/Hide grid - Show or Hide the grid in the schematic editor. Pressing the tool again hides (shows) the grid if it was shown (hidden) earlier.
2. Show hidden pins - Show hidden pins of certain components, for example, power pins of certain ICs.

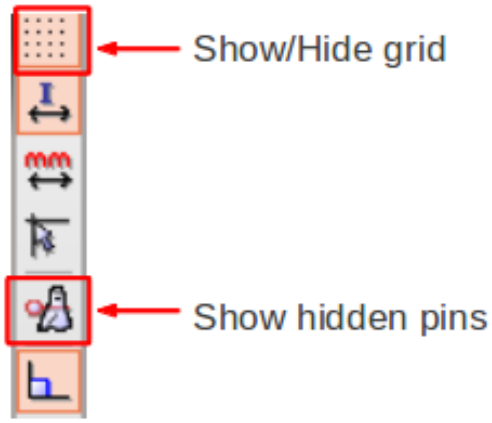

Figure 5.5: Toolbar on right - important tools

5.2 Hotkeys

A set of keyboard keys are associated with various operations in the schematic editor. These keys saves time and make it easy to move from one operation to another. The list of hotkeys can be viewed by going to Preferences in the top menu bar. Choose *Hotkeys* and select *List current keys*. You can also edit the hotkeys by selecting the option *Edit Hotkeys*. Some of the useful hotkeys are listed below:

- F1 - Zoom in
- F2 - Zoom out
- Ctrl + Z - Undo
- Delete - Delete item
- M - Move item
- C - Copy item
- A - Add/place component
- P - Place power component
- R - Rotate item

- X - Mirror component about X axis
- Y - Mirror component about Y axis
- E - Edit schematic component
- W - Place wire
- T - Add text
- S - Add sheet

Note that both lower and upper-case keys will work as hotkeys.

5.3 Components and Component libraries

Oscad schematic editor has a huge collection of components. All the component libraries in EEschema, on which Oscad schematic editor is based, are available. As EEschema is meant to be a schematic editor to create circuits for PCB, EEschema lacks some components that are necessary for simulation (e.g., plots, current sources, etc.). A set of component libraries has been created with such components. If you are using Oscad only for designing a PCB, then you might not need these libraries. However, these libraries are essential if you need to simulate your circuit. Hereafter, we will refer to these libraries as *Oscad libraries* to distinguish them from libraries already present in EEschema (EEschema libraries).

5.3.1 Oscad libraries

The Oscad libraries (created for simulation) are given below:

1. analogSpice - Discrete components like capacitor, resistor, BJT etc.
2. analogXSpice - Analog Xspice library
3. convergenceAidSpice - To set initial conditions
4. converterSpice - A/D and D/A converters
5. digitalSpice - ICs for digital circuits e.g., the 74 series

6. digitalXSpice - Flip-flops, logic gates etc.
7. linearSpice - 555 timer IC, op amp 741 etc
8. measurementSpice - Plot and print components
9. portSpice - Port
10. sourcesSpice - Current and voltage sources for simulation

Note that the names of all Oscad libraries end with the word *Spice*. Consider the Oscad library *linearSpice*. If you want to simulate a circuit that has a 555 timer IC in it, you should use the 555 timer from this library. This is because only then it will be mapped to the subcircuit of 555 which is required for simulation. Similarly if you use a Flip flop from digitalXSpice, the Xspice description of the Flip flop will be mapped to it and enables us to simulate the Flip flop behaviour.

5.3.2 Adding Oscad component libraries to project

Let us see how you can add the Oscad libraries to your project. Go to *Preferences* from the top menu bar. Choose *Library*. You will get the window shown in Figure 5.6. Click on *Add* (marked in red). Browse to the folder where Oscad is installed. Go to the folder *Library*. Select all the `*.lib` files as shown in Figure 5.7. Click on Open. Now click on OK on the window shown in Figure 5.6.

Note: You will have add these to your project each time you create or edit your schematic.

5.3.3 Library browser

You can browse through EEschema and Oscad libraries using the library browser tool from the top menu bar. The components in the *sourcesSpice* library is shown in Figure 5.8 to illustrate this.

Note that you will be able to view the Oscad libraries in the library browser ONLY IF you have added them to your project as described in Section 5.3.2

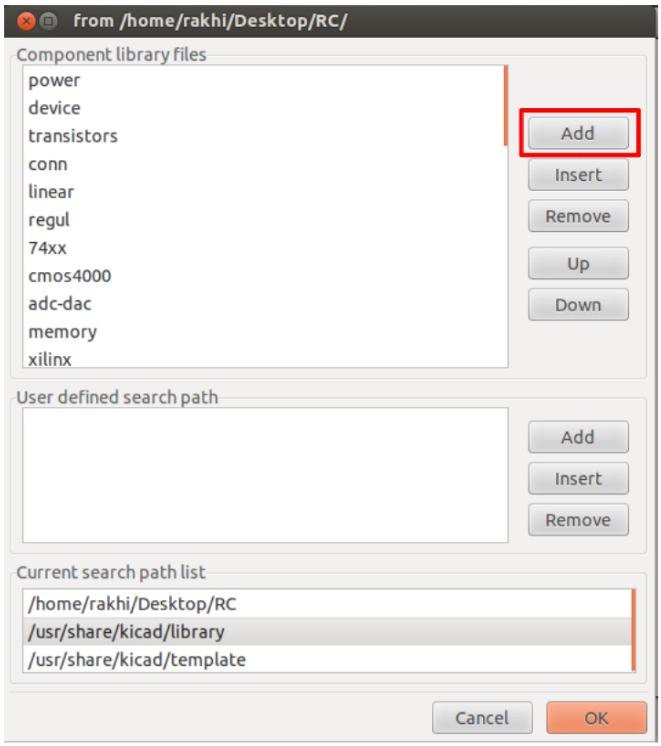

Figure 5.6: Add component library

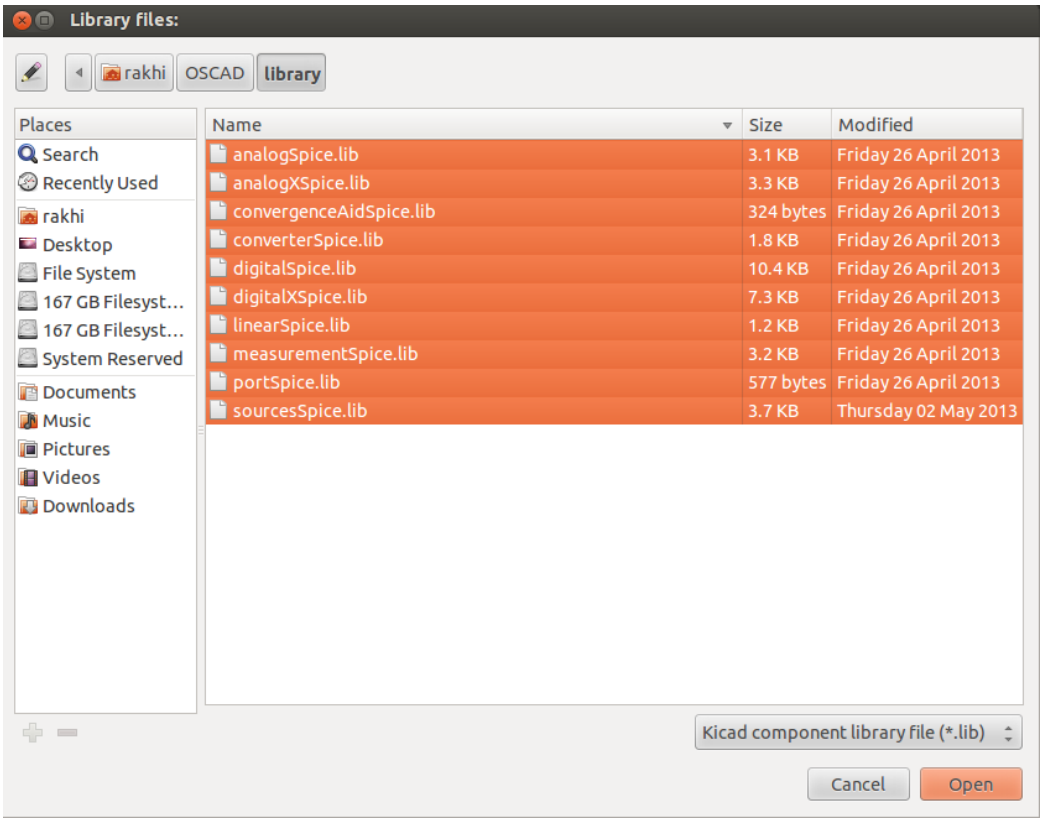

Figure 5.7: Select all the Oscad library (*.lib) files as shown

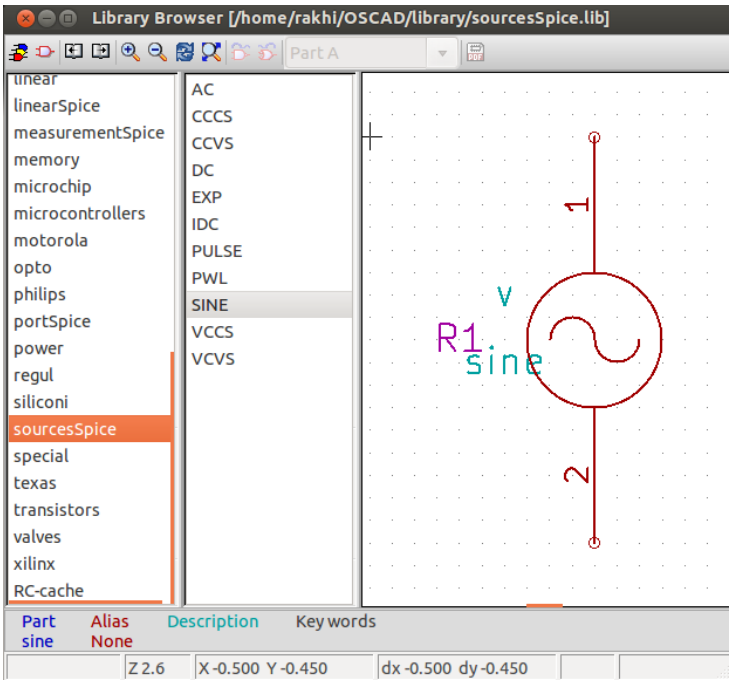

Figure 5.8: Library browser - an example

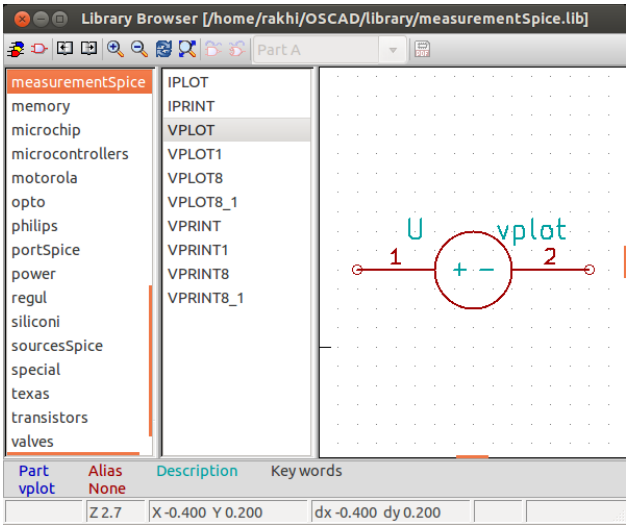

Figure 5.9: The Oscad *measurementSpice* library

5.3.4 Plot component library

Plot components are required to view the results of simulation. These are available in the Oscad library *measurementSpice* shown in Figure 5.9. These are used only for simulations. Some of the plots available in this library are:

- IPLOT - Plot the current through a component.
- VPLOT1 - Plot the voltages at nodes in separate graph windows.
- VPLOT8_1 - Plot the voltages at nodes in the same graph window.
- VPLOT - Plot the voltage difference between the two nodes where it is placed.

5.3.5 Power component library

Power components (Vcc and ground) are essential parts of a schematic - both for PCB design and simulation. Power components are available in the EEschema library *power* as shown in Figure 5.10. Another important component in this library is the Power Flag, *PWR_FLAG*. It is a dummy component placed in schematic to tell the schematic editor that the pin/node is driven by a power source and hence prevent ERC errors.

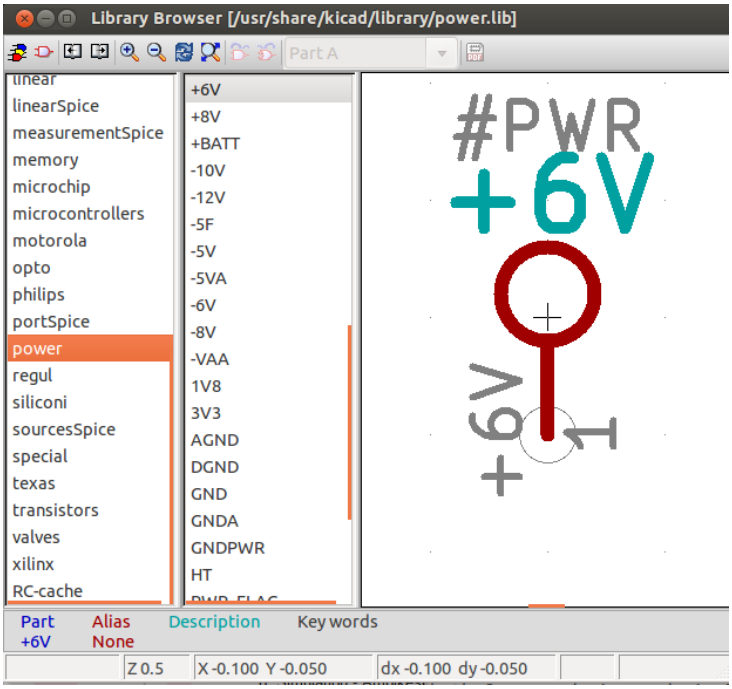

Figure 5.10: The EEschema *power* library

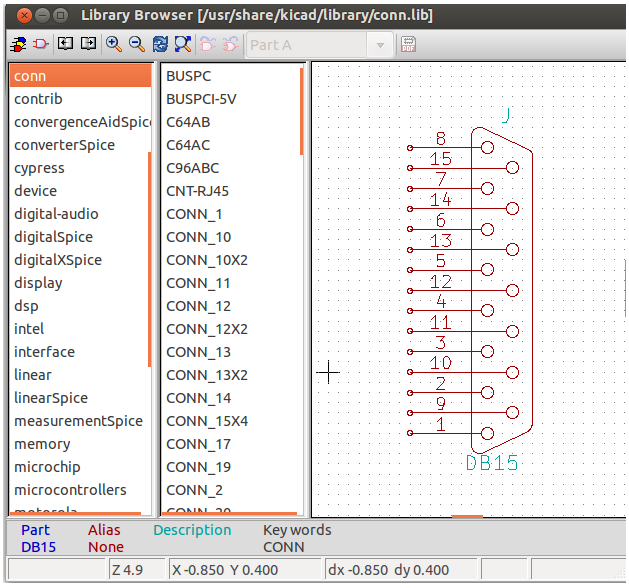

Figure 5.11: The EEschema *conn* library

5.3.6 Connector library

You would want to place connectors in your PCB to take signals in and out of it. These connectors are available in the EEschema library *conn* as shown in Figure 5.11.

5.3.7 Component references

Every component has a unique reference. For e.g., resistor has a reference `R`, BJTs have a reference `Q`, MOSFETs have a reference `M`, ICs have a reference `U` etc. When a component is placed in the schematic editor, the reference will be shown with a question mark. This indicates that the component is not annotated. See section 5.5.4 for more information about annotation.

5.4 Schematic creation for Simulation and PCB design - Differences

There are certain differences between the schematic created for simulation and that created for PCB design. We need certain components like plots,

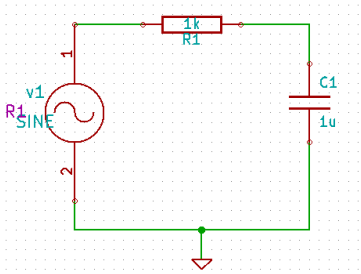

Figure 5.12: RC circuit

current sources etc. for simulation whereas these are not needed for PCB design. For PCB design, we would require connectors (e.g., DB15, 2 pin connector etc) for taking signals in and out of the PCB whereas these have no meaning in simulation.

5.5 Schematic creation for simulation

The first step in the creation of circuit schematic is the selection and placement of required components. Let us see this using an example. Let us create the circuit schematic of an RC filter given in Figure 5.12 and do a transient simulation.

5.5.1 Selection and Placing of components

We would need a resistor, a capacitor, a voltage source, ground terminal and some plot components.

Add the Oscad libraries to your project as described in section 5.3.2.

To place a resistor on your schematic editor, select the *Place a component* tool from the toolbar on the right side and click anywhere on the schematic editor. This opens up the component selection window. The above action can also be performed by pressing the key A. Type `R` in the field *Name* of the `component selection` window as shown in Figure 5.13. Click on OK. A resistor will be tied to the cursor. Place the resistor on the schematic editor by a single click.

To place the next component, i.e., capacitor, click again on the schematic editor. Type `C` in the Name field of component selection window. Click on OK. Place the capacitor on the schematic editor by a single click.

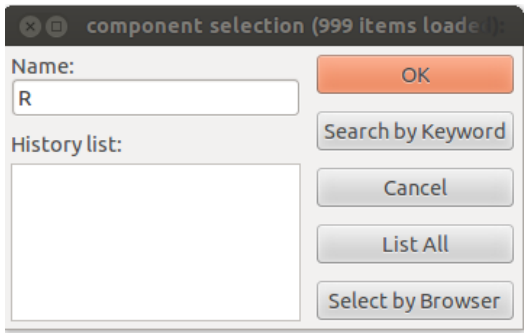

Figure 5.13: Placing a resistor using the Place a Component tool

Let us now place a sinusoidal voltage source. This is required for performing transient analysis. To place it, click again on the schematic editor. On the component selection window, click on `List all`. Choose the library `sourcesSpice` by double-clicking on it. Select the component `SINE` and click on OK. Place the sine source on the schematic editor by a single click.

We need to place two plot components. Let us place `vplot8_1` as we need to view input and output waveforms in the same graph window. To do so, choose and place vplot8_1 from the `measurementSpice` library. To place one more vplot8_1, place the cursor on top of vplot8_1 and press the key `C` to copy it. Place the component by clicking on the schematic editor.

Similarly place a ground terminal `gnd` from the library *power*. It can also be placed using the *Place a power port* tool from the toolbar on the right. Click anywhere on the editor after selecting place a power port tool. Click *List all* and choose `gnd`.

Once all the components are placed, the schematic editor would like the Figure 5.14.

Let us rotate the resistor to complete the circuit as shown in Figure 5.12. To rotate the resistor, place the cursor on the resistor and press the key `R`. Note that if you place the cursor above the letter `R` (not `R?`) on the resistor, you may be asked to clarify selection. Choose the option *Component R*. You can avoid this by placing the cursor slightly away from the letter R as shown in Figure 5.15. This applies to all components.

If you want to move a component, place the cursor on top of the component and press the key `M`. The component will be tied to the cursor and can be moved in any direction.

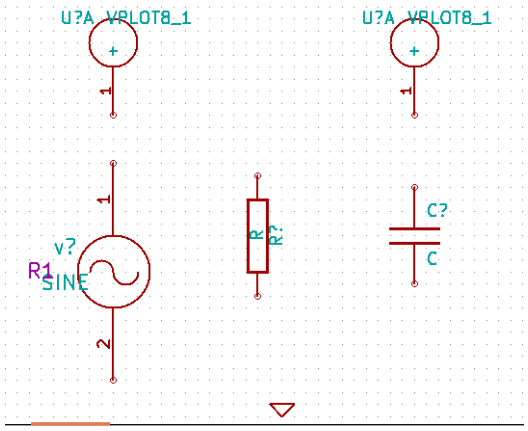

Figure 5.14: All components for RC circuit simulation are placed

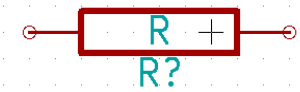

Figure 5.15: Place the cursor (shown by the cross mark) slightly away from the letter R

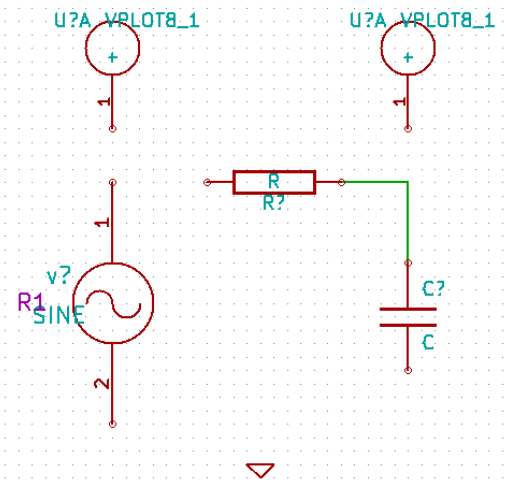

Figure 5.16: Wiring example

5.5.2 Wiring the circuit

The next step is to wire the connections. Let us connect the resistor to the capacitor. To do so, point the cursor to the terminal of resistor you want to connect and press the key `W`. It has now changed to the wiring mode. Move the cursor towards the terminal of the capacitor and click on it. A wire is formed as shown in Figure 5.16. Similarly connect the wires between all terminals and the final schematic would look like Figure 5.17.

5.5.3 Assigning values to components

We need to assign values to the components in our circuit i.e., resistor and capacitor. Note that the sine voltage source has been placed for simulation. The specifications of sine source will be given during simulation.

To assign value to resistor, place the cursor above the letter `R` (not `R?`) and press the key `E`. Choose *Field value*. Type `1k` in the *Edit value field* box as shown in Figure 5.18. 1k means $1k\Omega$. Similarly give the value `1u` for the capacitor. 1u means $1\mu F$.

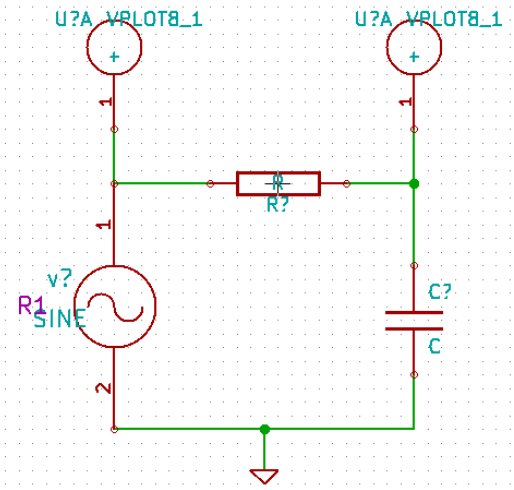

Figure 5.17: Wiring done

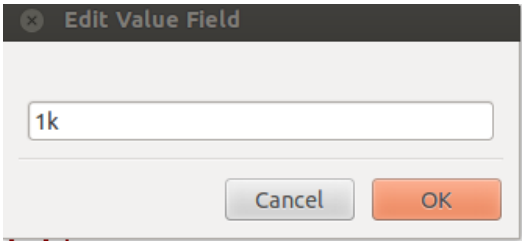

Figure 5.18: Editing value of resistor

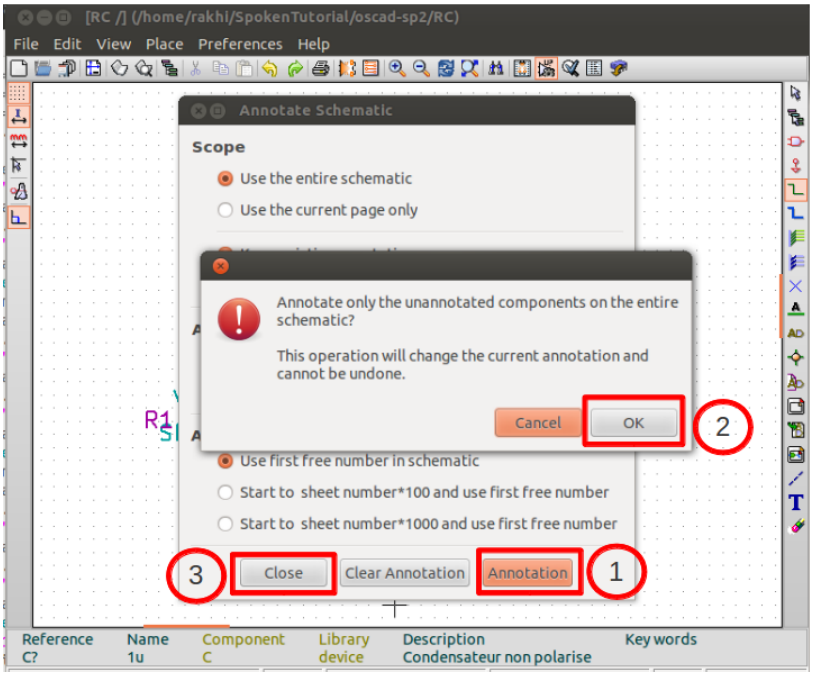

Figure 5.19: 1. First click on Annotate then 2. Click on Ok then 3. Click on close

5.5.4 Annotation and ERC

The next step is to annotate the schematic. Annotation gives unique references to the components. to annotate the schematic, click on *Annotate schematic* tool from the top toolbar. Click on annotate, then click on OK and finally click on close as shown in Figure 5.19. The schematic is now annotated. The question marks next to component references have been replaced by unique numbers. If there are more than one instance of a component (say resistor), the annotation will be done as R1, R2 etc.

Let us do ERC or Electric Rules Check next. To do so, click on *Perform electric rules check* tool from the top toolbar. Click on *Test Erc* button. You may get an error as shown in Figure 5.20. Click on close in the test erc window. There will be a green arrow pointing to the source of error, here it points to the ground terminal. This is shown in Figure 5.21

To correct this error, place a `PWR_FLAG` from the EEschema library `power`. Connect the power flag to the ground terminal as shown in Figure 5.22. More information about PWR_FLAG is given in section 5.3.5. Repeat the

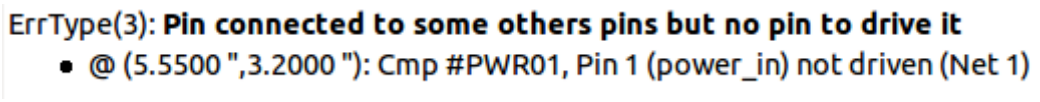

Figure 5.20: ERC error

Figure 5.21: Green arrow pointing to Ground terminal indicating an ERC error

ERC. Now there are no errors. With this we have created the schematic for simulation.

5.5.5 Netlist generation

To simulate the circuit that you created in the previous section, we need to generate its netlist. `Netlist` is a list of components in the schematic along with their connection information. To do so, click on the *Generate netlist* tool from the top toolbar. Click on spice from the window that appears now. Uncheck the option `Prefix references 'U' and 'IC' with 'X'`. Then click on *Netlist*. This is shown in figure 7.8. Save the netlist. This will be a `.cir` file. Do not change the directory while saving.

Now you are ready with the netlist to be simulated. The next chapter will guide you to perform simulation.

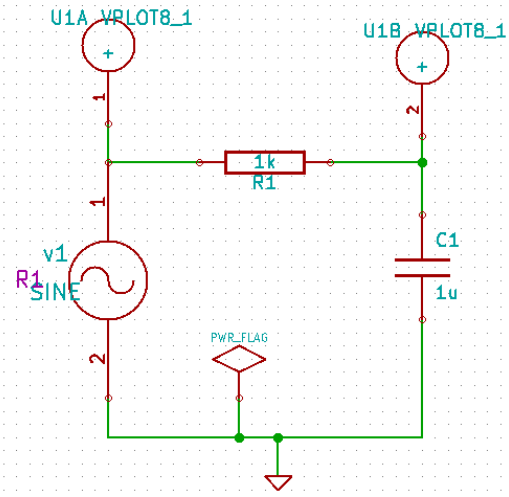

Figure 5.22: Final schematic with PWR_FLAG

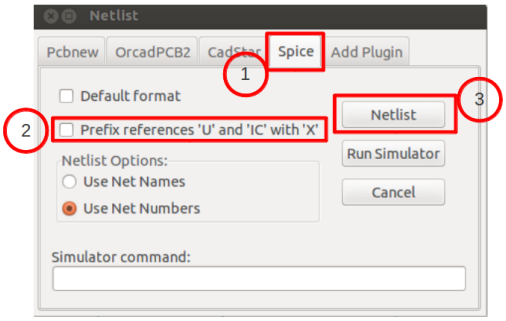

Figure 5.23: 1. Click on Spice then 2. Uncheck the option `Prefix references 'U' and 'IC' with 'X'` then 3. Click on Netlist

Chapter 6

Simulation

Circuit simulation uses mathematical models to replicate the behaviour of an actual device or circuit. Simulation software allows for modeling of circuit operation. Simulating a circuit's behaviour before actually building it can greatly improve design efficiency by making faulty designs known as such, and providing insight into the behavior of electronics circuit designs. Oscad uses Ngspice for mixed-level/mixed-signal circuit simulation.

6.1 Analysis

Oscad supports three types of analyses which are depicted below:

1. DC Analysis (Operating Point and DC Sweep)
2. AC Small-Signal Analysis
3. Transient Analysis

In order to understand the relationship between the different analyses and the two underlying simulation algorithms of ngspice, it is important to understand what is meant by each analysis type. This is detailed below.

6.1.1 DC Analysis

The dc analysis portion of ngspice determines the dc operating point of the circuit with inductors shorted and capacitors opened. The dc analysis options are specified on the .DC, .TF, and .OP control lines. There is assumed

to be no time dependence on any of the sources within the system description. The simulator algorithm subdivides the circuit into those portions which require the analog simulator algorithm and those which require the event-driven algorithm. Each subsystem block is then iterated to solution, with the interfaces between analog nodes and event-driven nodes iterated for consistency across the entire system.

Once stable values are obtained for all nodes in the system, the analysis halts and the results may be displayed or printed out as you request them.

A dc analysis is automatically performed prior to a transient analysis to determine the transient initial conditions, and prior to an ac small-signal analysis to determine the linearized, small- signal models for nonlinear devices. If requested, the dc small-signal value of a transfer function (ratio of output variable to input source), input resistance, and output resistance is also computed as a part of the dc solution. The dc analysis can also be used to generate dc transfer curves: a specified independent voltage, current source, resistor or temperature1 is stepped over a user- specified range and the dc output variables are stored for each sequential source value.

6.1.2 AC Small-signal Analysis

AC analysis is limited to analog nodes and represents the small signal, sinusoidal solution of the analog system described at a particular frequency or set of frequencies. This analysis is similar to the DC analysis in that it represents the steady-state behavior of the described system with a single input node at a given set of stimulus frequencies.

The program first computes the dc operating point of the circuit and determines linearized, small-signal models for all of the nonlinear devices in the circuit. The resultant linear circuit is then analyzed over a user-specified range of frequencies. The desired output of an ac small-signal analysis is usually a transfer function (voltage gain, transimpedance, etc). If the circuit has only one ac input, it is convenient to set that input to unity and zero phase, so that output variables have the same value as the transfer function of the output variable with respect to the input.

6.1.3 Transient Analysis

Transient analysis is an extension of DC analysis to the time domain. A transient analysis be-gins by obtaining a DC solution to provide a point of

departure for simulating time-varying behavior. Once the DC solution is obtained, the time-dependent aspects of the system are rein- troduced, and the two simulator algorithms incrementally solve for the time varying behavior of the entire system. Inconsistencies in node values are resolved by the two simulation algorithms such that the time-dependent waveforms created by the analysis are consistent across the entire simulated time interval. Resulting time-varying descriptions of node behavior for the specified time interval are accessible to you. All sources which are not time dependent (for example, power supplies) are set to their dc value. The transient time interval is specified on a .TRAN control line.

6.2 Analysis Inserter

In order to simulate a circuit, an user must define the type of simulation to be run. The type of analysis includes Operating point analysis, DC analysis, AC analysis, transient analysis etc. Also, user needs to specify the option corresponding to the analysis. Analysis inserter generate the command for ngspice. When you click on analysis inserter the window shown in Figure 6.1 will appear. It consists of type of analysis in top, and details which are needed to sweep the input value.

6.2.1 DC Analysis

By default DC analysis window will open when you click on analysis inserter, where we need to give the details of input source name, start point of input, increment and stop point. When you click on Add Simulation Data the window shown in Figure 6.2 will appear. In this example we consider 'v1' as a input voltage source which starts from '0Volt' and incremented by '1Volt' and stopped at '10Volt'. The syntax generated by this process is of the form

. dc sourcename vstart vstop vincr

The .dc line defines the dc transfer curve source and sweep limits (again with capacitors open and inductors shorted). srcnam is the name of an independent voltage or current source, a resistor or the circuit temperature. vstart, vstop, and vincr are the starting, final, and incrementing values respectively.

When we select .op option then the window given in Figure 6.3 will appear. `.op` - The inclusion of this line in an input file directs ngspice to

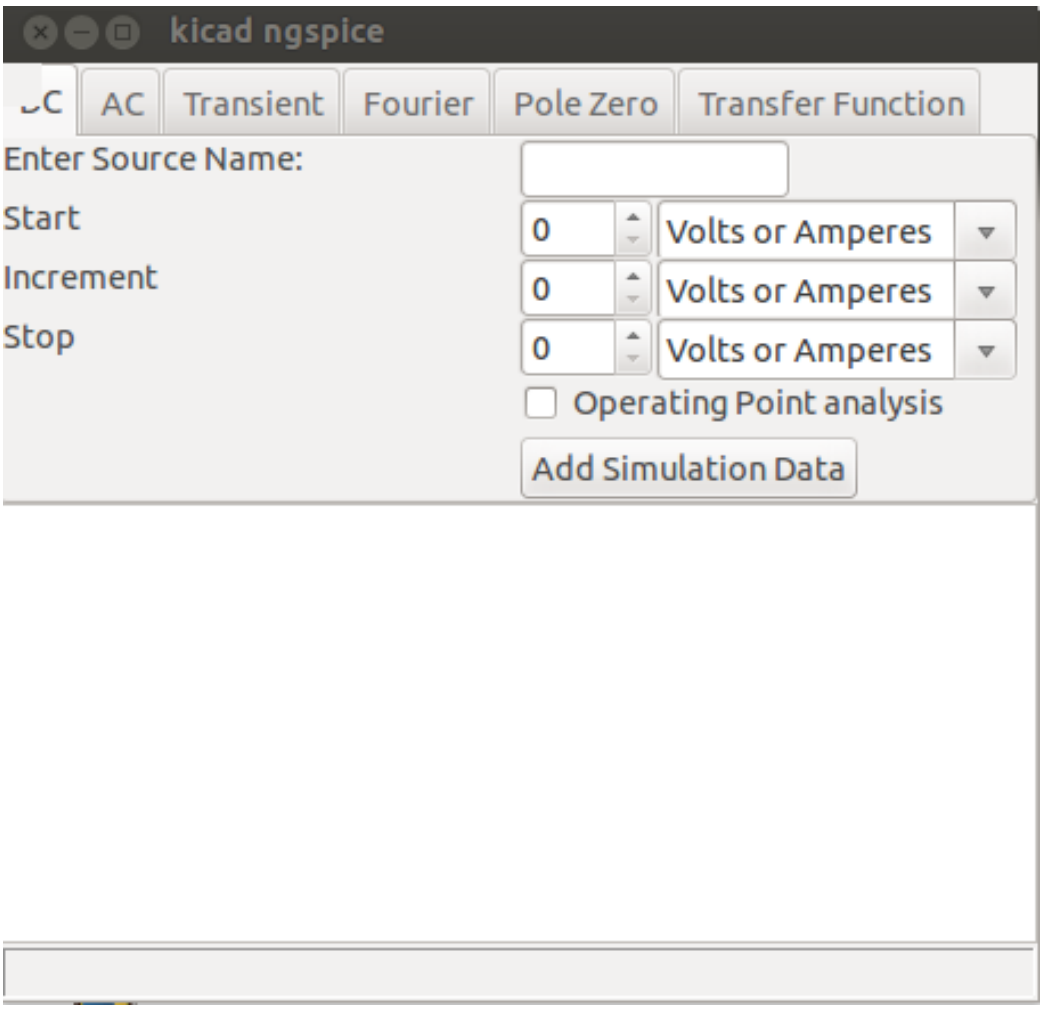

Figure 6.1: Analysis inserter GUI

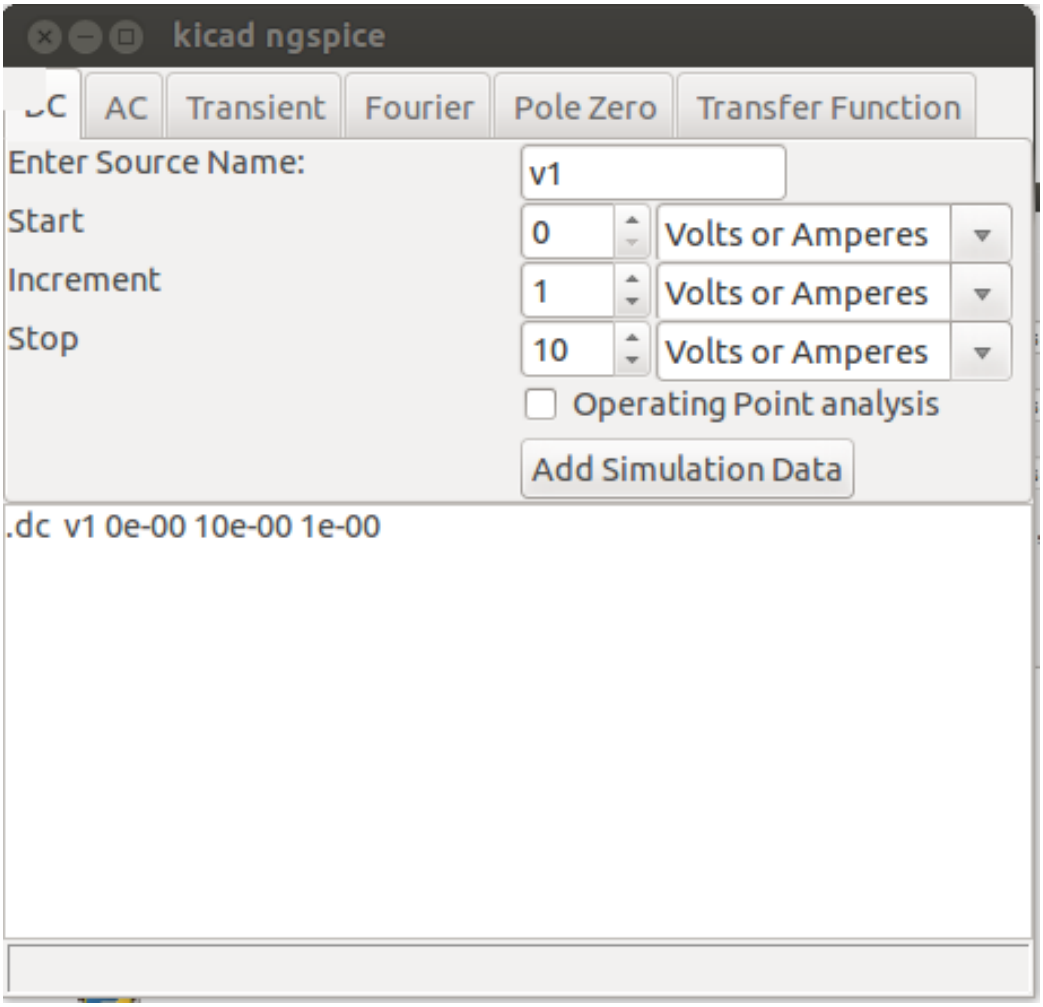

Figure 6.2: DC Analysis

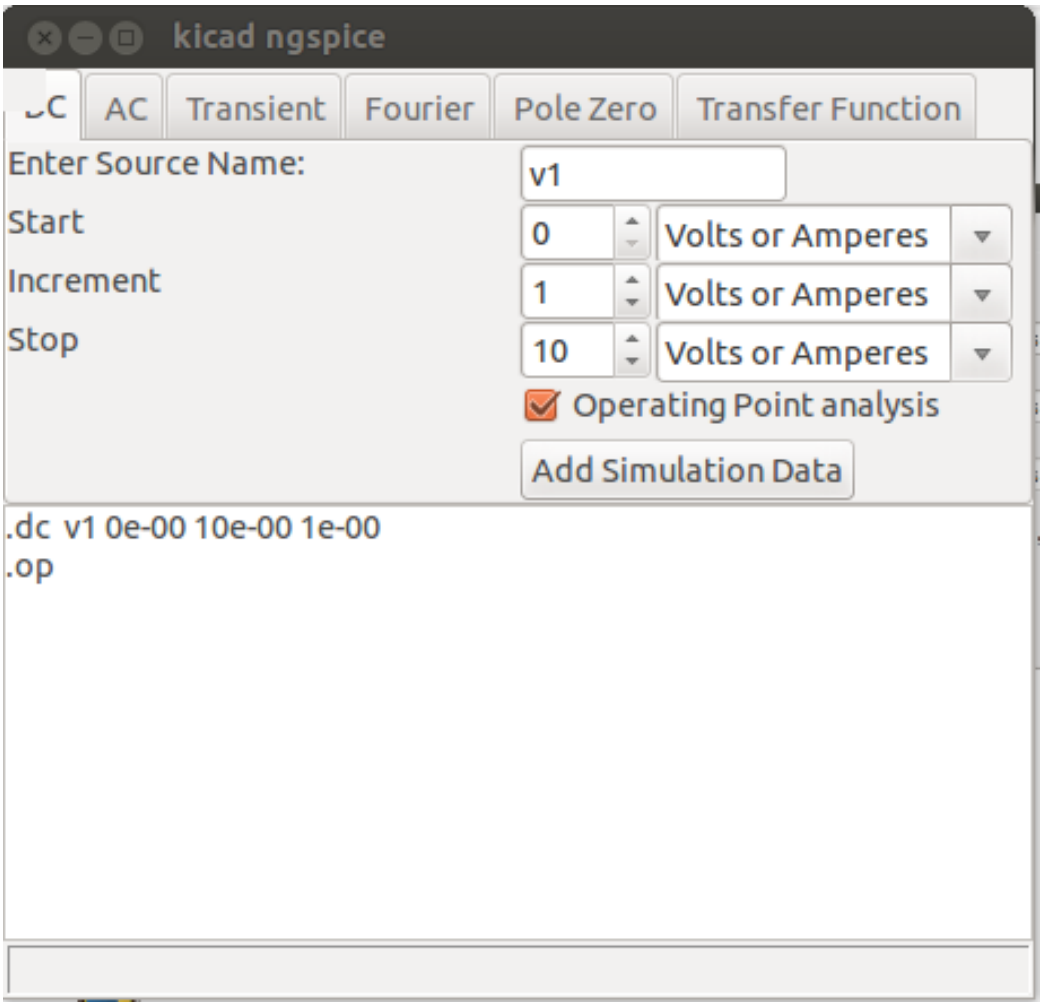

Figure 6.3: OP Analysis

determine the dc operating point of the circuit with inductors shorted and capacitors opened.

6.2.2 AC Analysis

When you click on AC the window given in Figure 6.4 will appear. This window will ask you to enter the detail of scale, start frequency, stop frequency, numbers of point.

After entering all these values and clicking on Add Simulation Data you will get the window as shown in Figure 6.5. This window shows that start frequency is 1 Hz, stop frequency is 10 MegHz and numbers of point 10. The syntax generated by this is of the form:

```
.ac dec nd fstart fstop
.ac oct no fstart fstop
.ac lin np fstart fstop
```

`dec` stands for decade variation, and `nd` is the number of points per decade. `oct` stands for octave variation, and `no` is the number of points per octave. `lin` stands for linear variation, and np is the number of points. `fstart` is the starting frequency, and `fstop` is the final frequency. If this line is included in the input file, ngspice performs an AC analysis of the circuit over the specified frequency range. Note that in order for this analysis to be meaningful, at least one independent source must have been specified with an ac value.

6.2.3 Transient Analysis

If you click on the option 'Transient', the window given in Figure 6.6 will appear. Above window will ask to enter the details like start time, step time, stop time After entering these values when you click on Add Simulation Data you will get the window as shown in Figure 6.7. Above window shows that start time is 0 sec, step time is 1 sec and stop time is 10 sec. The syntax generated by this process is of the form:

```
.tran tstep tstop <tstart <tmax >> <uic>
```

tstep is the printing or plotting increment for line-printer output. For use with the postprocessor, tstep is the suggested computing increment. tstop is the final time, and tstart is the initial time. If tstart is omitted, it is assumed to be zero. The transient analysis always begins at time zero. In the interval ¡zero, tstart¿, the circuit is analyzed (to reach a steady state), but no outputs

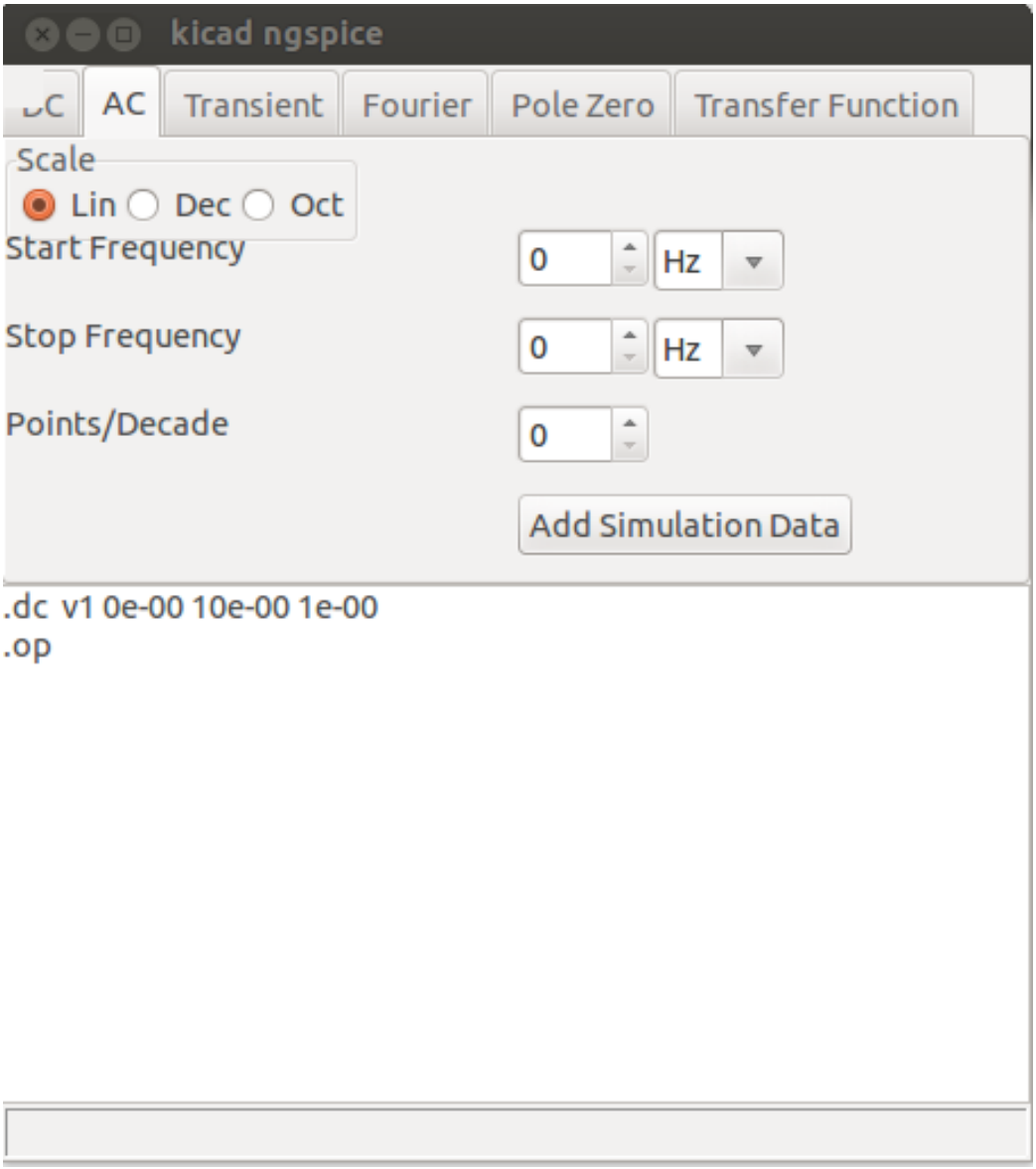

Figure 6.4: AC Analysis GUI

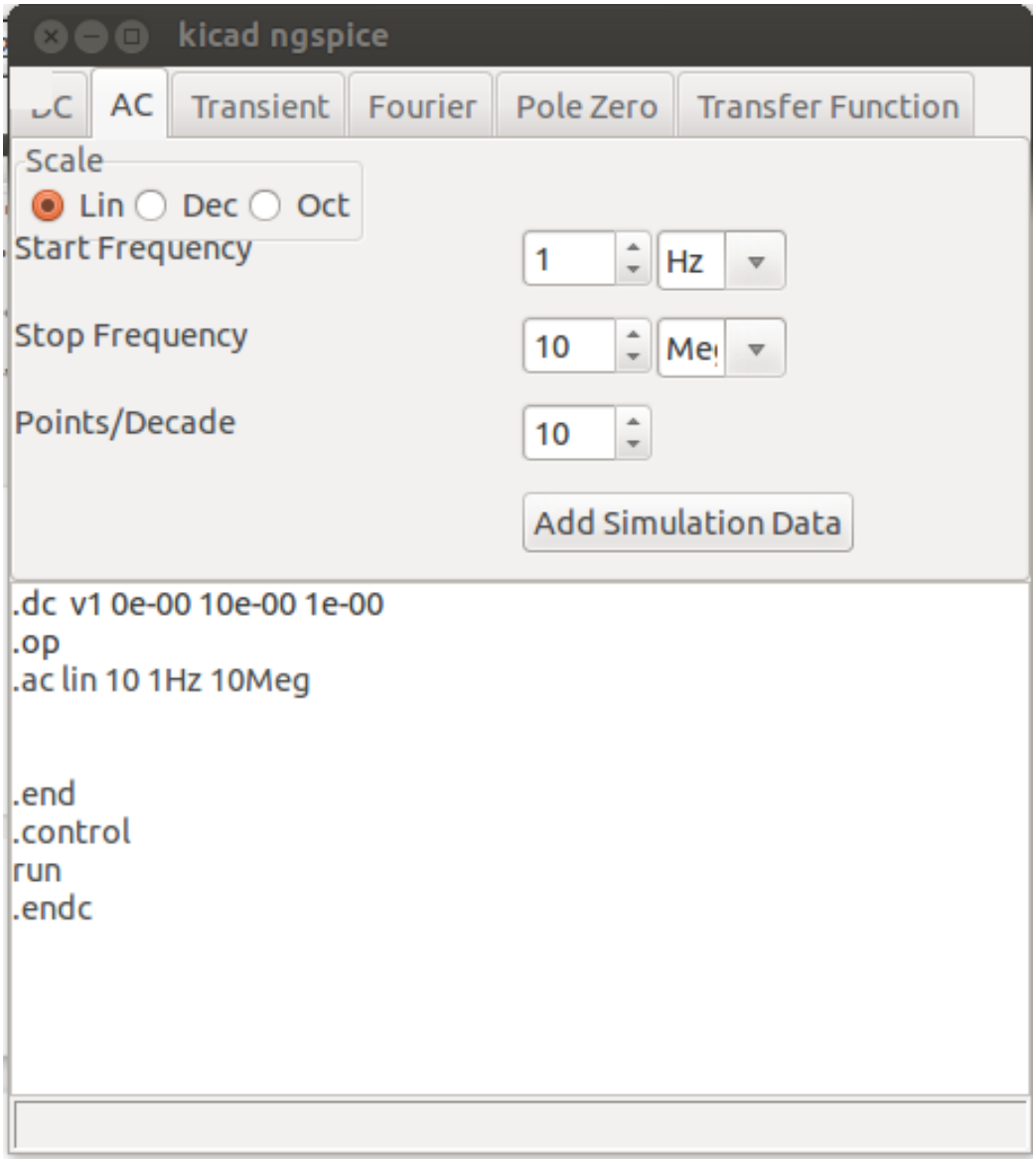

Figure 6.5: AC Analysis example

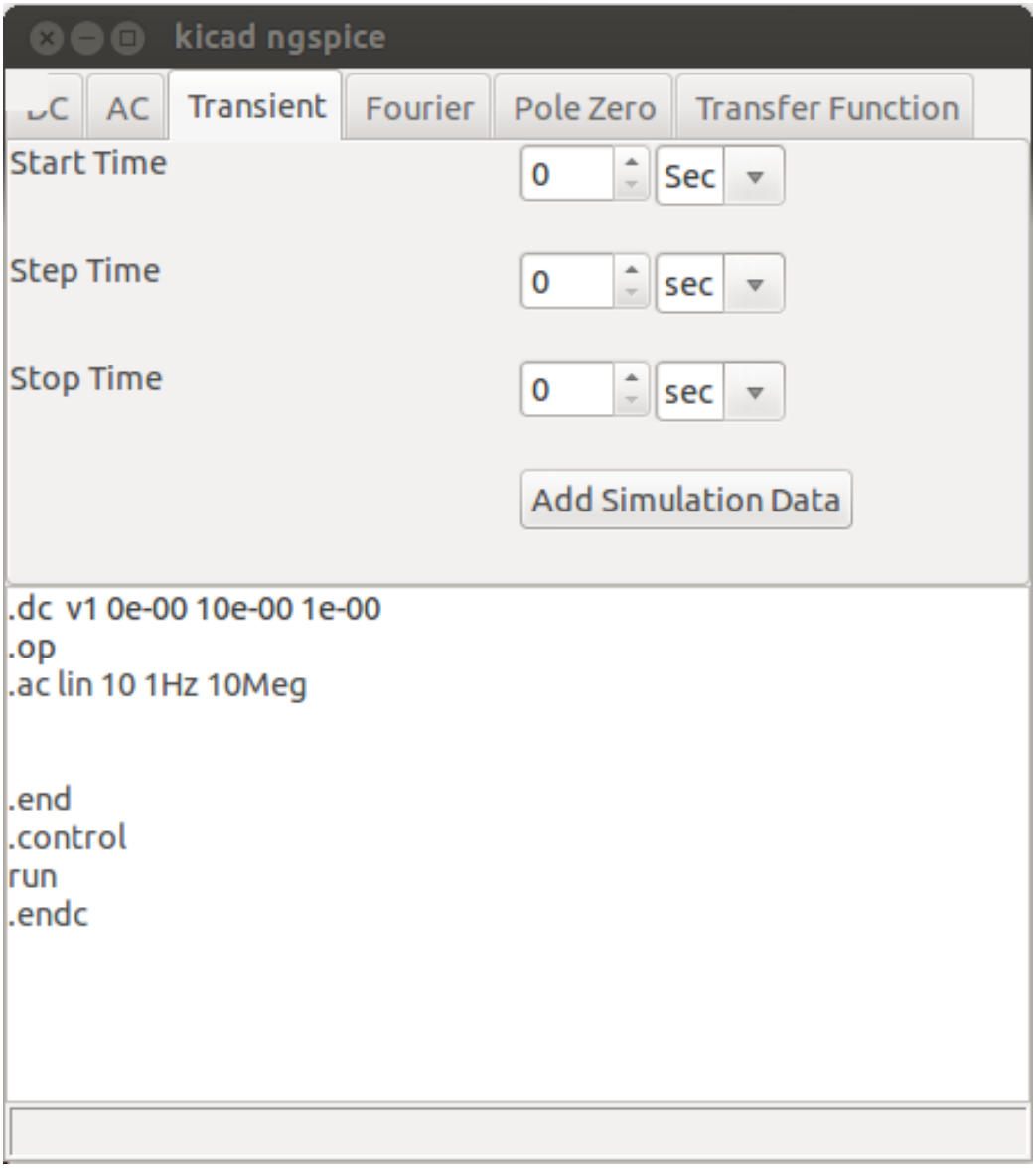

Figure 6.6: Transient Analysis - GUI

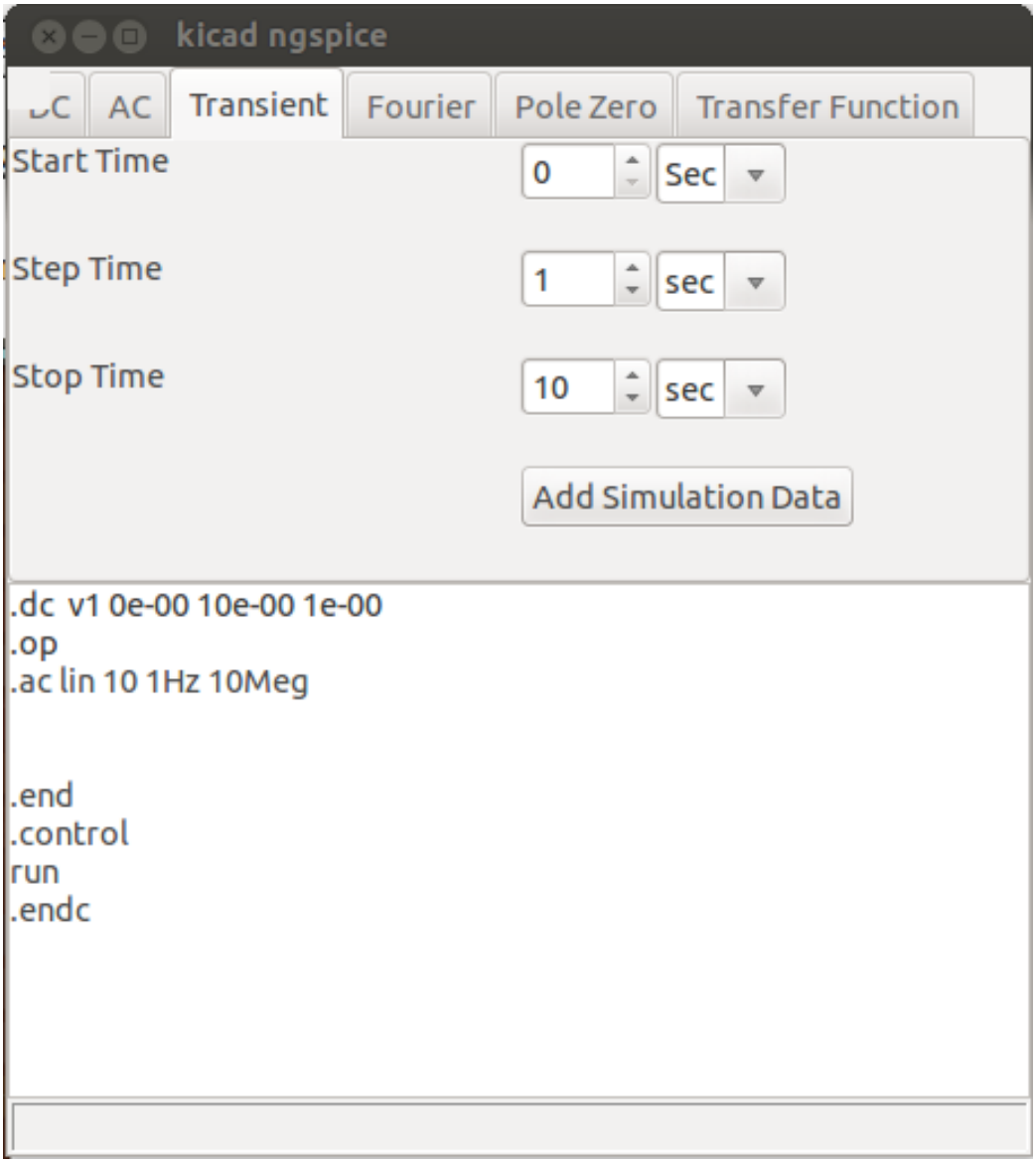

Figure 6.7: Transient Analysis - example

are stored. In the interval ¡tstart, tstop¿, the circuit is analyzed and outputs are stored. tmax is the maximum stepsize that ngspice uses; for default, the program chooses either tstep or (tstop-tstart)/50.0, whichever is smaller. tmax is useful when one wishes to guarantee a computing interval which is smaller than the printer increment, tstep. An initial transient operating point at time zero is calculated according to the following procedure: all independent voltages and currents are applied with their time zero values, all capacitances are opened, inductances are shorted, the non linear device equations are solved iteratively. uic (use initial conditions) is an optional keyword which indicates that the user does not want ngspice to solve for the quiescent operating point before beginning the transient analysis.

After entering the details of analysis we need to save the analysis file. These details will be store in analysis file. Use following process to save the analysis file. Move your cursor to file as shown in figure 6.8 then click on it. save option will come after clicking on file. Click on save. The file will be saved in analysis file.

6.3 KiCad to Ngspice Conversion

A schematic editor provides a netlist file, which describes the electrical connections between circuit components. Eeschema provides a SPICE netlist which can not be directly used for simulation due to compatibility issues. After Analysis insertion click on Kicad to Ngspice tool from the Oscad toolbar. Oscad netlist converter performs following operation to generate Ngspice compatible netlist.

6.3.1 Insert parameters for fictitious components

For simulation of the circuit, the value of sources must be specified, e.g., for sine wave voltage source, parameters (frequency, amplitude, offset etc.) are required to be entered. Our netlist converter scan the netlist file and ask for parameter values for the sources wherever required. When we click on kicad to ngspice converter, a terminal window appears. It asks for various parameter values. This varies depending upon the voltage/current sources added in the schematic for simulation.

1. When sinusoidal source is available in schematic - in this terminal we

Figure 6.8: Saving the analysis information to netlist

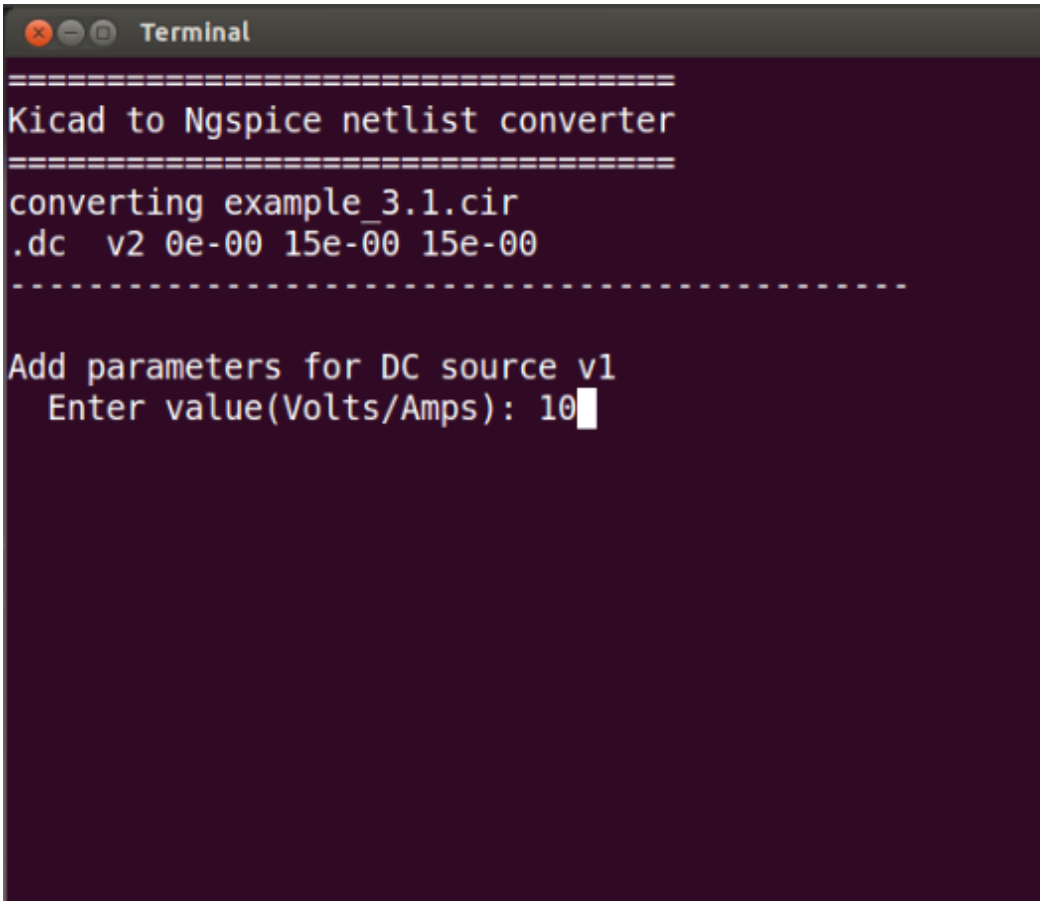

Figure 6.9: Parameters to be added for sinusoidal source

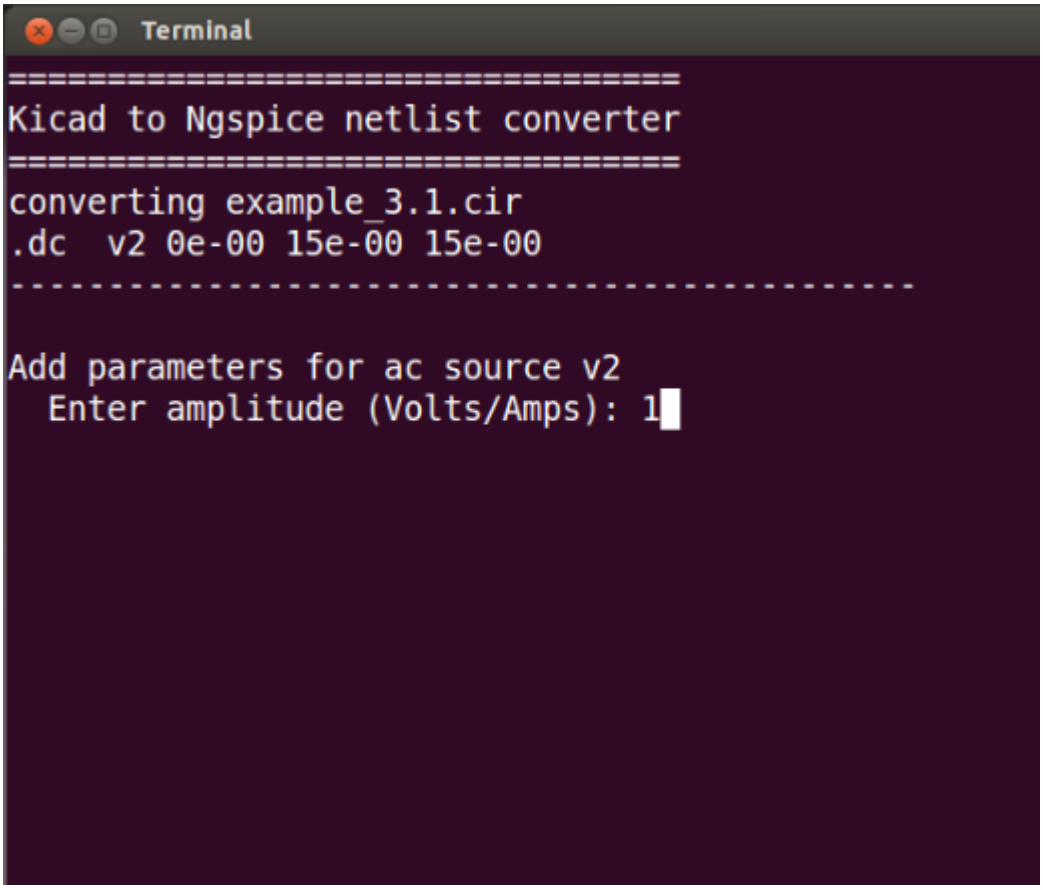

Figure 6.10: Parameters to be added for dc source

will add all the details of sinusoidal source like offset value, amplitude, frequency, delay time, damping factor.

2. When DC source of unknown value is available in schematic - It will ask for the value of DC source, in our case value of DC is 10V.

3. When AC source is available in schematic - It will ask for amplitude of AC, in our case value is 1V.

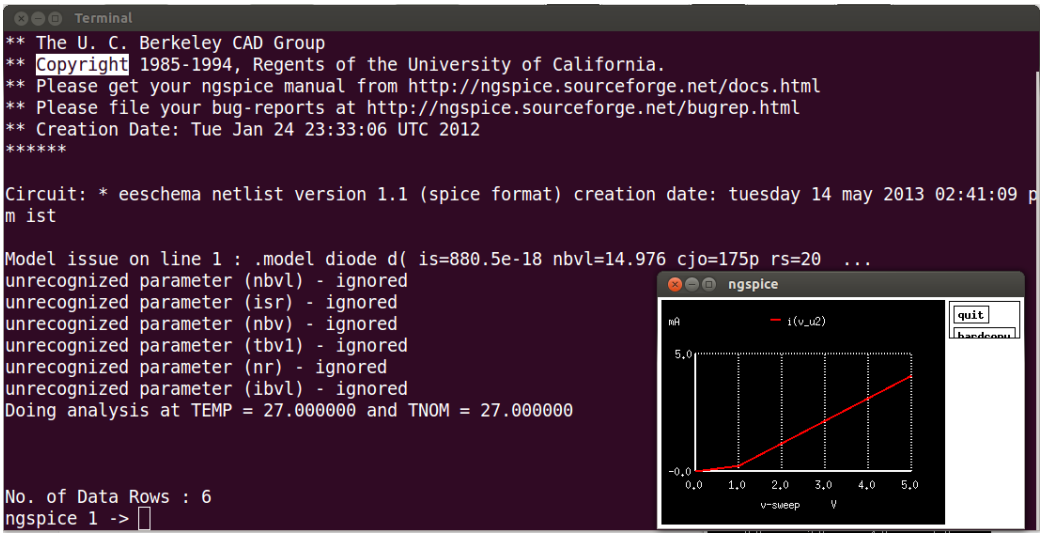

Figure 6.11: Parameters to be added for ac source

6.3.2 Convert IC into discrete blocks

As Eeschema is intended for PCB Designing, it gives netlist in terms of IC not components, e.g., if design contains a two-input Nand gate, then in the netlist, IC 7400 appears instead of the Nand gate. OSCAD netlist converter converts the IC into discrete blocks with considering proper input and output connections and IC specifications (voltage levels, speed of the operation etc.).

6.3.3 Insert Digital-to-Analog (D-to-A) and Analog-to-Digital (A-to-D) converter at appropriate places

Oscad provides capability to perform mix mode simulation. Thus circuits with analog and digital components can be analyzed. In order to simulate such kind of circuits, D-to-A and A-to-D converter are inserted at appropriate places. We assume the netlist generated from Eeschema has analog connections and for digital components, we have added A-to-D converter for inputs and D-to-A converter for outputs.

6.3.4 Insert plotting and printing statements

There is a library for plotting and printing the circuit solution (voltages and currents). The netlist converter adds appropriate printing and plotting commands (current or voltage plot, or single or differential plot) in the netlist depending on the print/plot components. Ngspice is able to find the current through voltage sources only. Netlist converter inserts a zero volt voltage source in series with the component through which current needs to be computed.

6.3.5 Insert analysis and option

Oscad netlist converter inserts analysis information created in Section 6.2 into the netlist.

6.3.6 Insert models and sub-circuits

Oscad netlist converter inserts models or sub-circuits for required components into the netlist.

6.4 Simulation

After Kicad to Ngspice we get `.cir.out` file which is compatible to Ngpice simulation software.

6.4.1 Ngspice

Ngspice is a general-purpose circuit simulation program for nonlinear and linear analyses. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent or dependent voltage and current sources, loss-less and lossy transmission lines, switches, uniform distributed RC lines, and the five most common semiconductor devices: diodes, BJTs, JFETs, MESFETs, and MOSFETs.

Ngspice is an update of Spice3f5, the last Berkeleys release of Spice3 simulator family. Ngspice is being developed to include new features to existing Spice3f5 and to fix its bugs. Improving a complex software like a circuit simulator is a very hard task and, while some improvements have been made, most of the work has been done on bug fixing and code refactoring.

Ngspice supports mixed-signal simulation through the integration of XSPICE code into it. XSPICE software, developed as an extension to Spice3C1 from GeorgiaTech, has been ported to ngspice to provide board level and mixed-signal simulation.

6.4.2 Ngspice simulation in Oscad

After clicking on Ngspice from the Oscad toolbar, we will get Ngspice terminal and waveform window as shown in Figure 6.13.

6.5 Simulation Examples

6.5.1 DC simulation example

Consider the nodal analysis example giving in the Examples available in the OScad webpage `www.oscad.net`. we open Kicad and generate spice netlist as given in section (schematic chapter), we click on analysis inserter,Here we decide which type of analysis we want to do.

`DC Analysis` - We click on DC and Enter the details source name=i1, start=0A, Increment=1A, stop=10A and then click on Add Simulation data as shown in Figure **??**. Now we click on Kicad to Ngpice which will show the terminal given in Figure 6.14. Press 'enter' key. After this we click on Ngspice button which will show Ngspice terminal with waveform window as shown in the Figure 6.15.

6.5.2 Example of AC and Transient Analysis

Consider the RC_ac example folder from the Example folder available in Oscad website and follow the steps given in section 6.5.1

`AC Analysis` - We click on AC and add details like scale, start frequency, stop frequency, number of points and then click on Add simulation data. This is shown in Figure 6.16. Now we click on Kicad to Ngpice which will show the following terminal where we add the amplitude of AC and press 'enter' key.

After this we click on Ngspice button which will show Ngspice terminal with waveform window as shown in following figure:

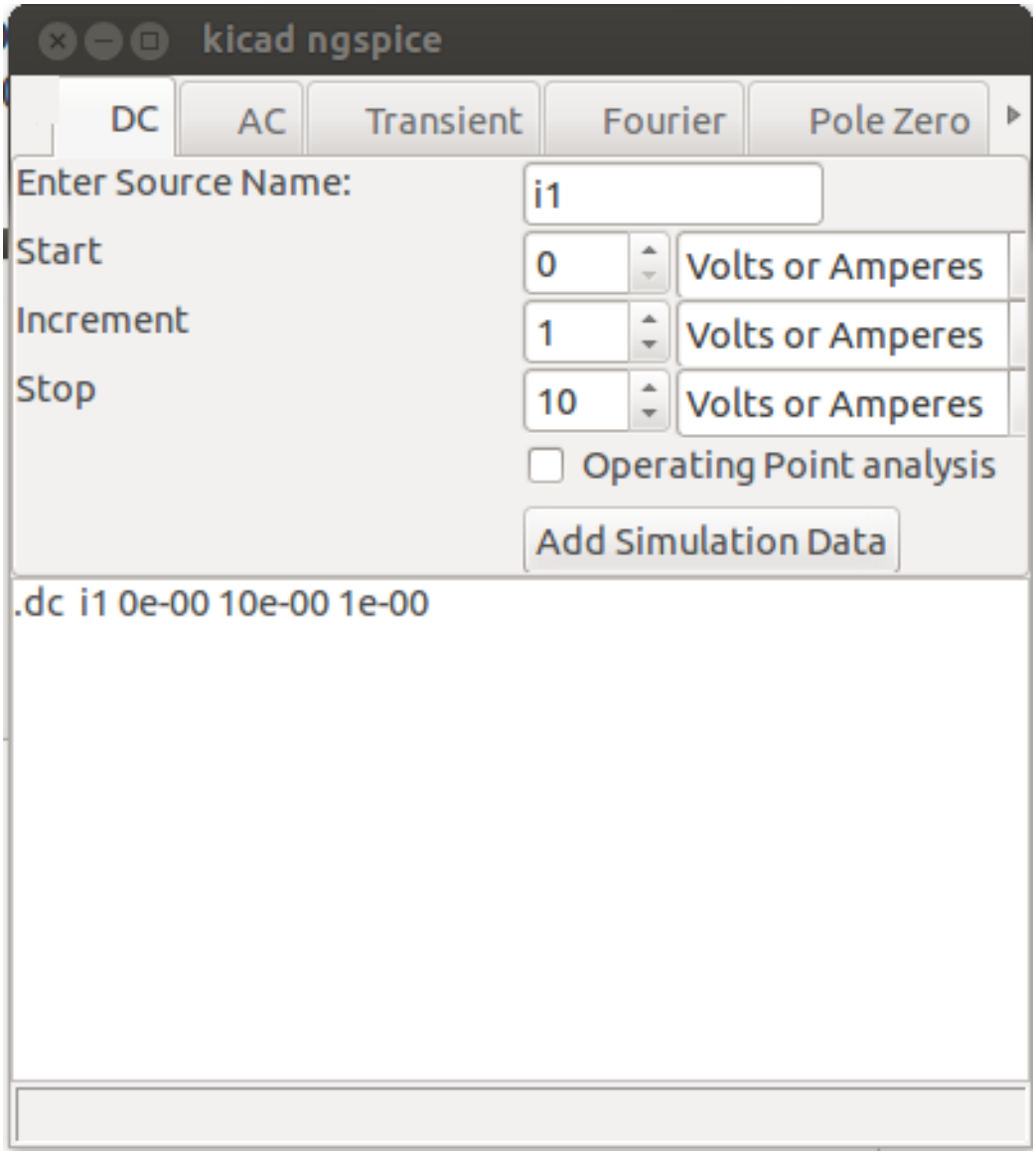

Figure 6.12: Ngspice simulation of `.cir.out` file

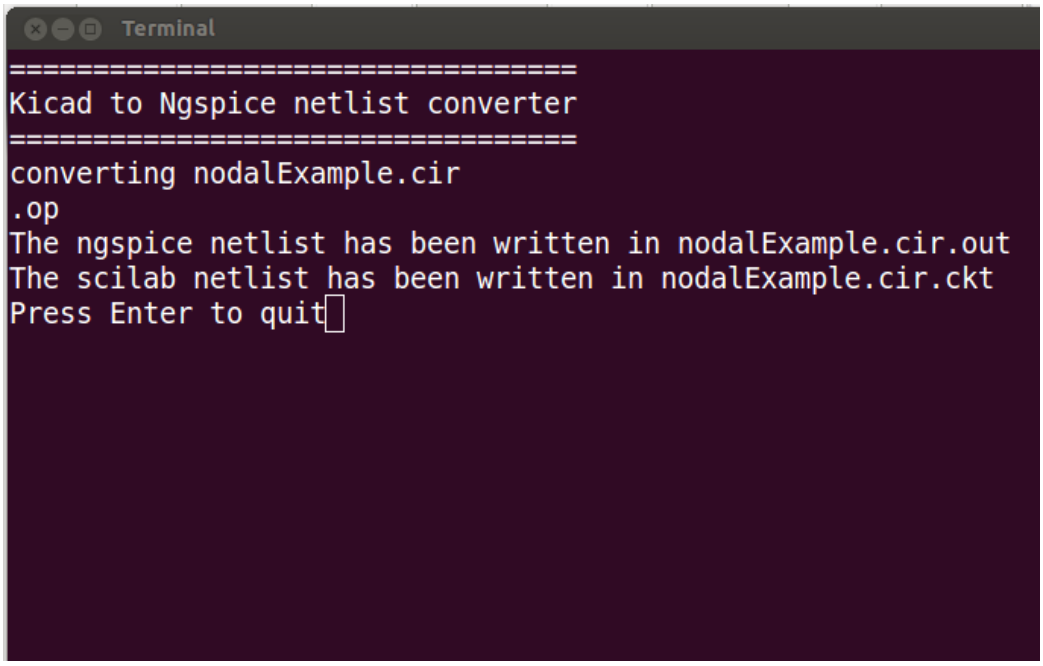

Figure 6.13: DC analysis - adding simulation data

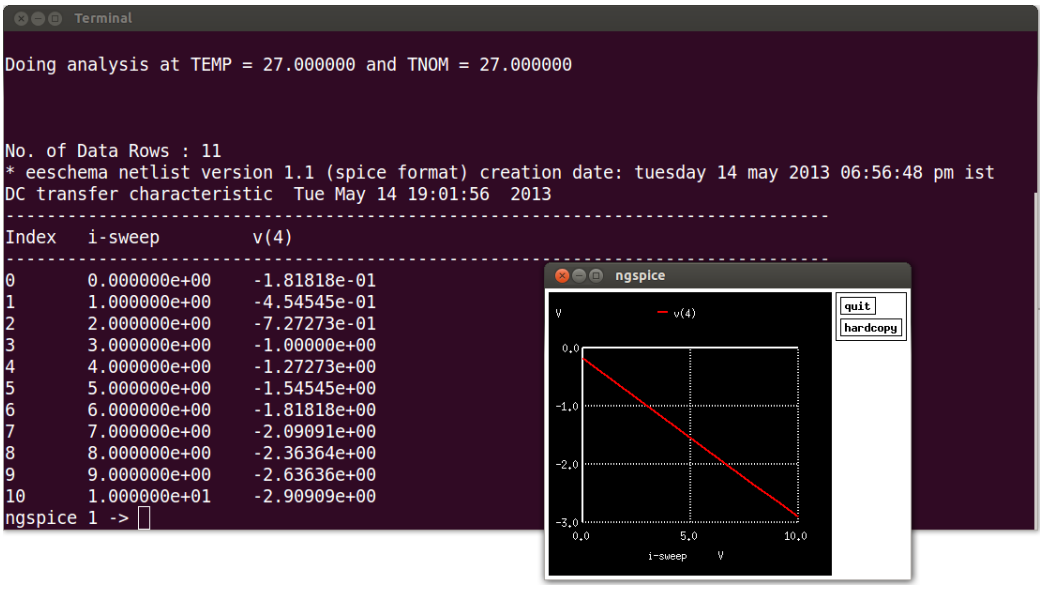

Figure 6.14: Terminal - KiCad to Ngspice netlist converter

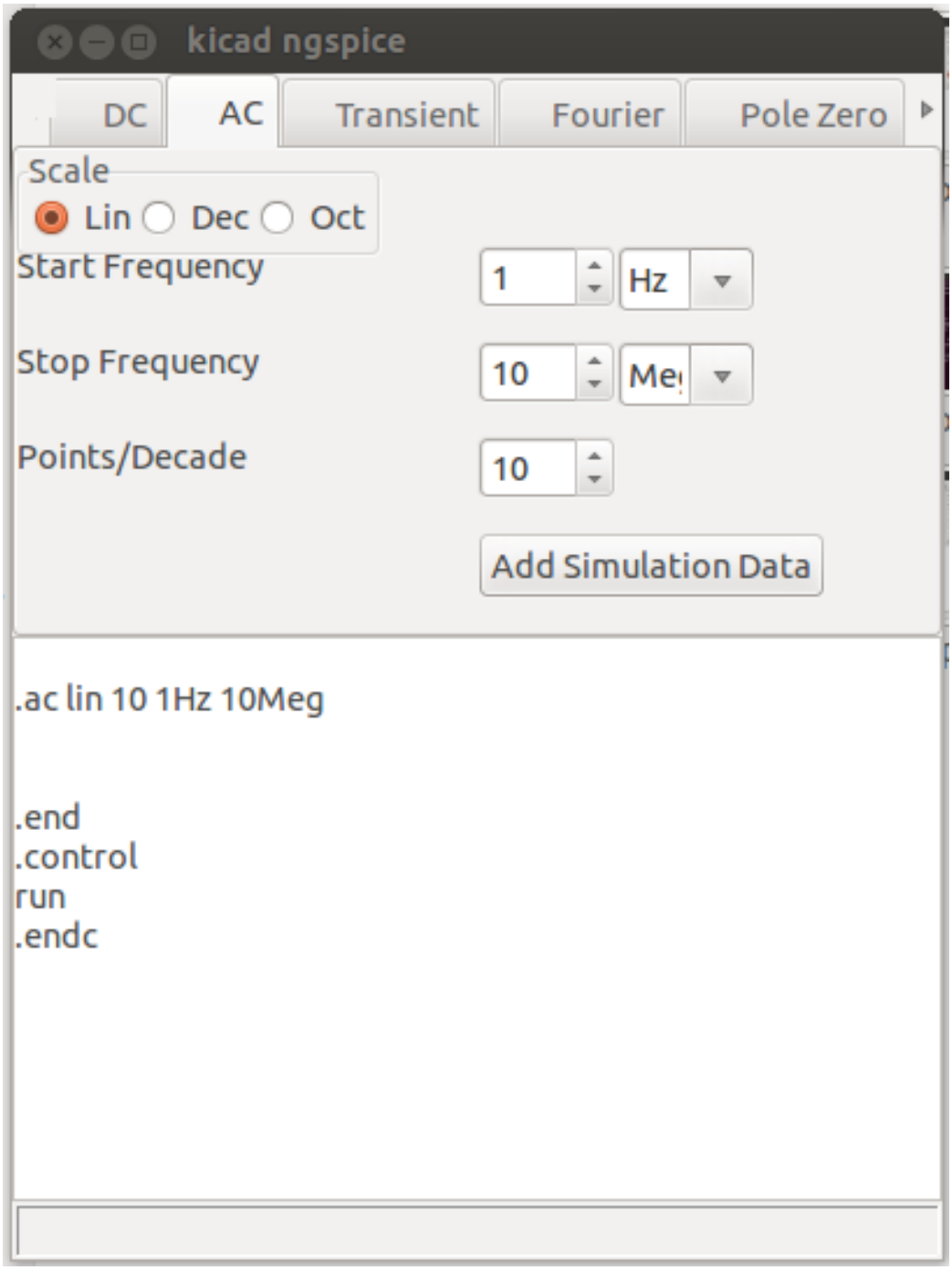

Figure 6.15: Ngspice simulation result

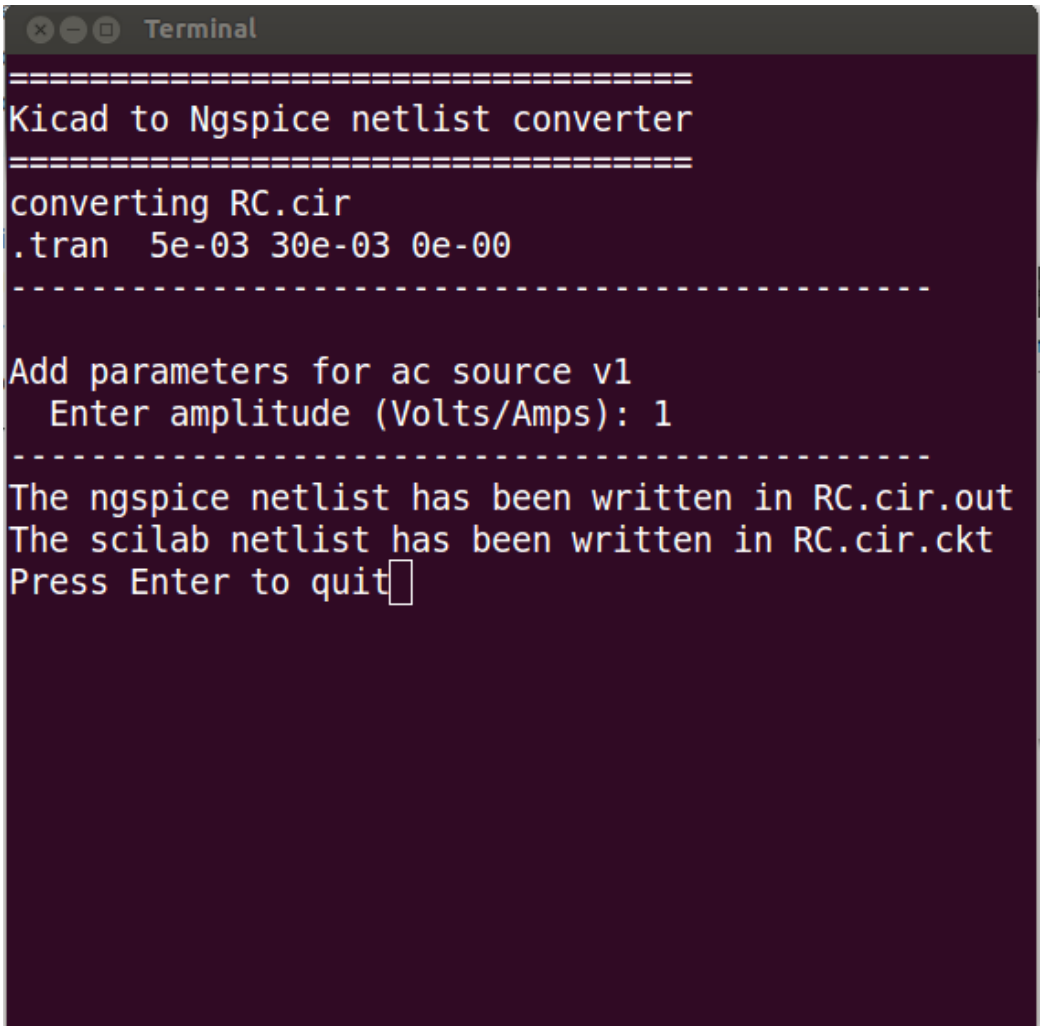

Figure 6.16: AC analysis example

We click on analysis inserter and then select type of analysis as Transient following window will appear:

now we add the details like start time, step time, stop time

Chapter 7

Model builder and Subcircuit builder

7.1 Model builder

Spice based simulators include a feature which allow accurate modelling of semiconductor devices such as diodes, transistors etc. OSCAD Model Builder provides a facility to define a new model for devices such as diodes, MOS, BJT, JFET, IGBT, Magnetic core etc. Model Builder in OSCAD asks for a value of parameters depending on the type of the device for which a model is required. The parameter values can be obtained from a datasheet of the device. A newly created model can be exported to the model library and whenever required one can import it for different projects. Model Builder also provides a facility to edit existing models.

Now let us take an example of Bridge rectifier circuit containing diode model 1N4007 and see how Model Builder works in Oscad.

In figure 7.1 you can see the diode model 1N4007 in the Bridge rectifier circuit.

Now to build a new model for the diode 1N4007 you will click on the Model builder of the toolbar of Oscad which will opens up the window shown in Figure 7.2 where it asks for the value of parameters for diode 1N4008. In this window you can change the values of the parameters for 1N4007 diode model as it is given in the datasheet and then save it.

Once a new diode model of 1n4007 is created it can be exported to the model library and whenever required one can import it for different projects

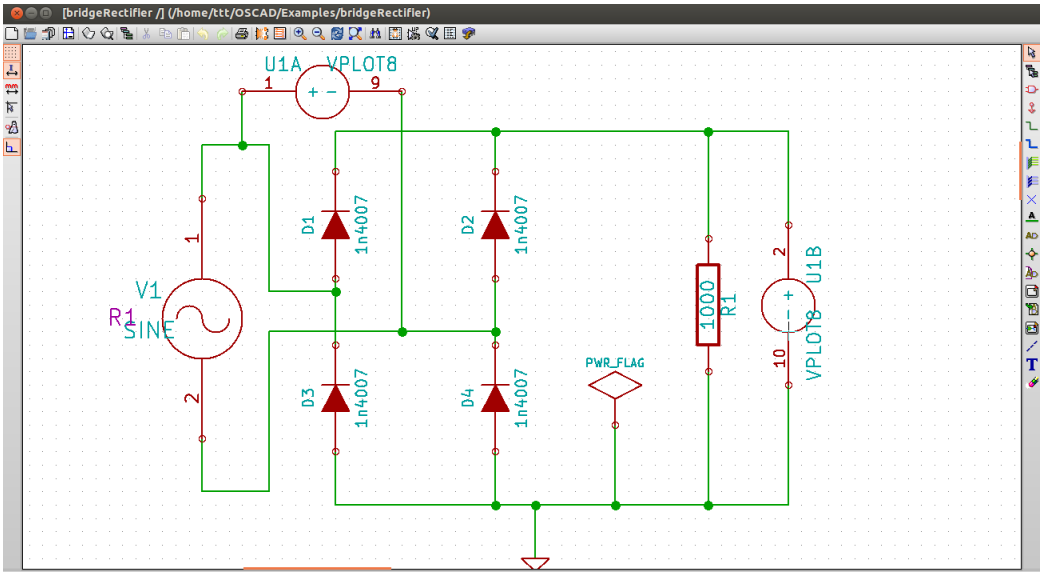

Figure 7.1: Diode model in bridge rectifier circuit

which shown in the Figure 7.3.

7.2 Sub-circuit Builder:

Sub-circuit is the way to implement hierarchical modeling. Once a sub-circuit for a component is created, user can use it for different circuits.

Oscad provides an easy way to create a sub-circuit in steps:

Firstly open Oscad sub-circuit builder. It will open the schematic editor (Eeschema). Draw a sub-circuit. Then connect external pin of the sub-circuit to the component called port (provided by Oscad). Last step is to save it.

Once saved, Sub-circuit builder finds out the pin connected to port and creates .Subckt line. In this line, the name of the component and the circuit nodes that connect sub-circuit to the main circuit are specified. The Ngspice compatible netlist is generated using the Netlist Converter explained in this section. The last line .ends is inserted in the sub-circuit definition. Now, the definition will be available for the project in which it is created.

Figure 7.4 shows a sub-circuit for voltage divider where nodes 1, 2 and 3 are external nodes and 4 is an internal node. The sub-circuit definition

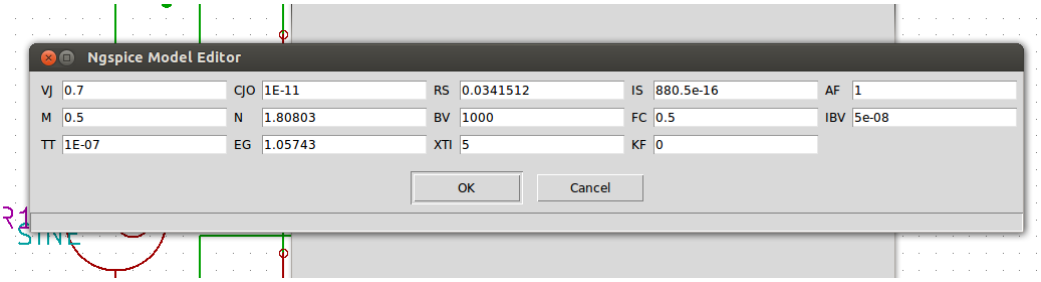

Figure 7.2: Edit model paramters

is given in the same figure. Once sub-circuit is created, it can be exported to the sub-circuit library and whenever required one can import it for different projects. The Oscad sub-circuit library already has some sub-circuit definitions for complex components such IC555, OpAmp 741 etc. The Oscad provides a library with 41 basic building blocks such as multiplier, limiter, basic gates, flip-flops, latches, D to A converter etc. Using these blocks, one can realize the most of the electronic circuits.

Now let us take an example of building a subcircuit of IC 555 timer which is a part of Astable Multiovibrator circuit and see how Subcircuit Builder works in Oscad.

Once you click on the Subcircuit Builder of the Oscad it asks you to choose the subcircuit you wanted to create out of the list of all subcircuits present in the given circuit. It is shown in Figure 7.5.

Once you select the subcircuit it opens up the Schematic Editor where you can create the subcircuit of IC 555 timer shown in Figure 7.6.

Once you completed the creation of the subcircuit save it in respective project folder and close the Schematic editor. While you close the Shematic Editor a terminal pop-up of Kicad to Ngspice netlist converter will ask you to enter the value for the different parameters of subcircuit as shown in the Figure 7.7.

At the end of the terminal you see that it tells you The ngspice netlist has been written in lm555n.cir.out The scilab netlist has been written in lm555n.cir.ckt Press Enter to quit which says that netlist for the subcircuit is created. It is shown in the Figure 7.8.

Finally it says that Created sub-circuit lm555n.sub as shown in Figure 7.9 and then press ok to complete the subcircuit creation.

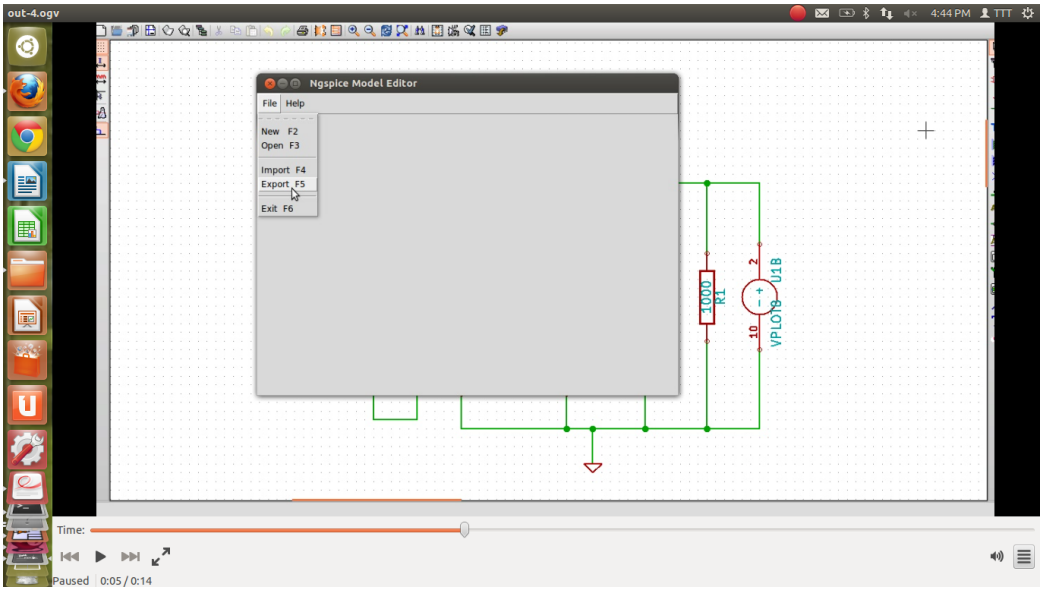

Figure 7.3: Export model

Once sub-circuit is created, it can be exported to the sub-circuit library as shown in Figure 7.10 and whenever required one can import it for different projects.

Finally close the subcircuit editor window.

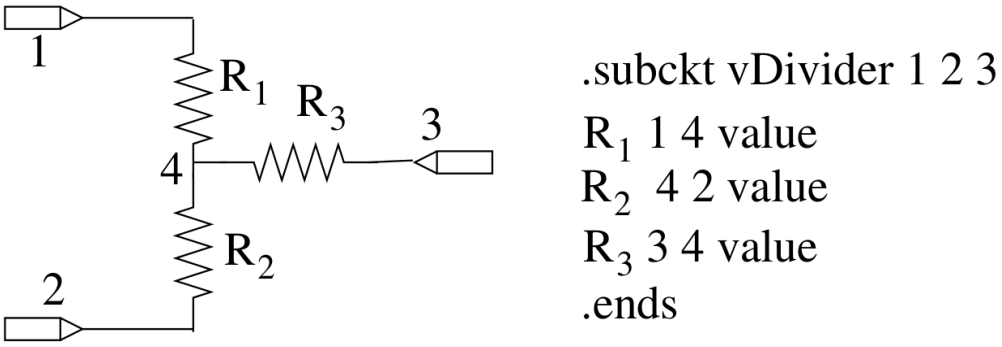

Figure 7.4: Voltage divider subcircuit and definition

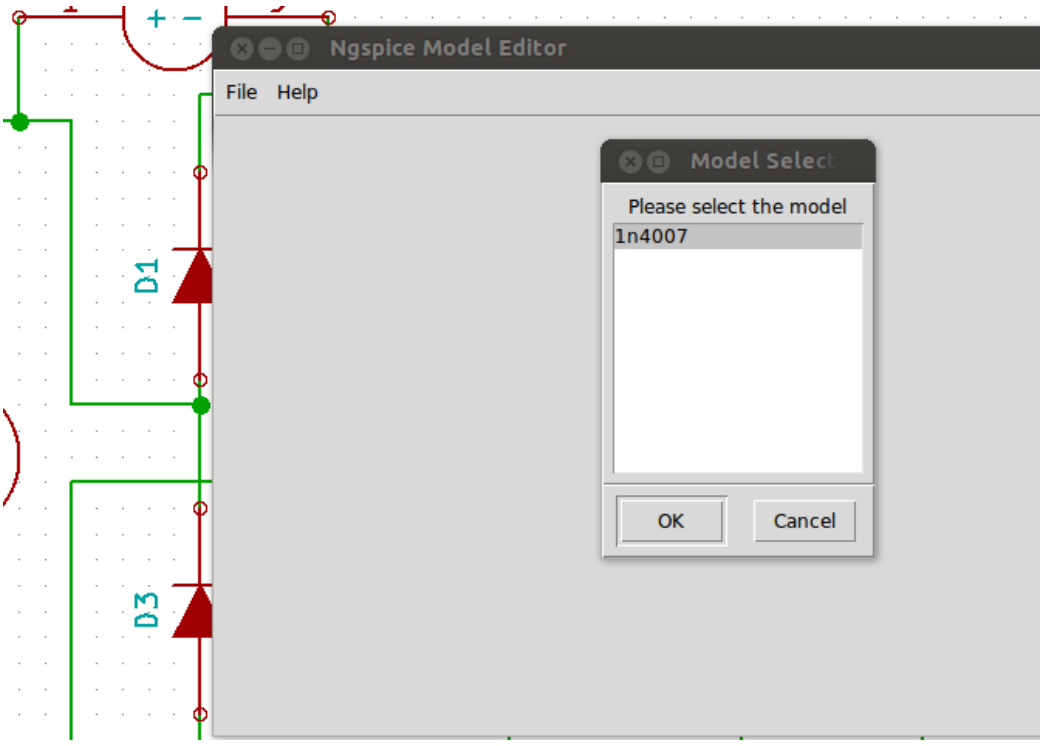

Figure 7.5: Select Model

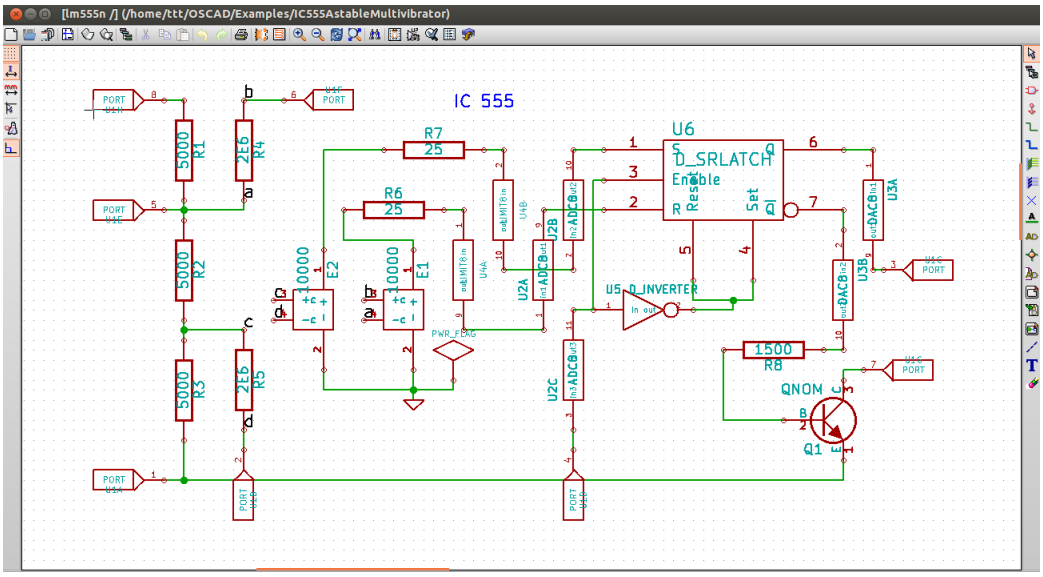

Figure 7.6: IC 555 timer subcircuit

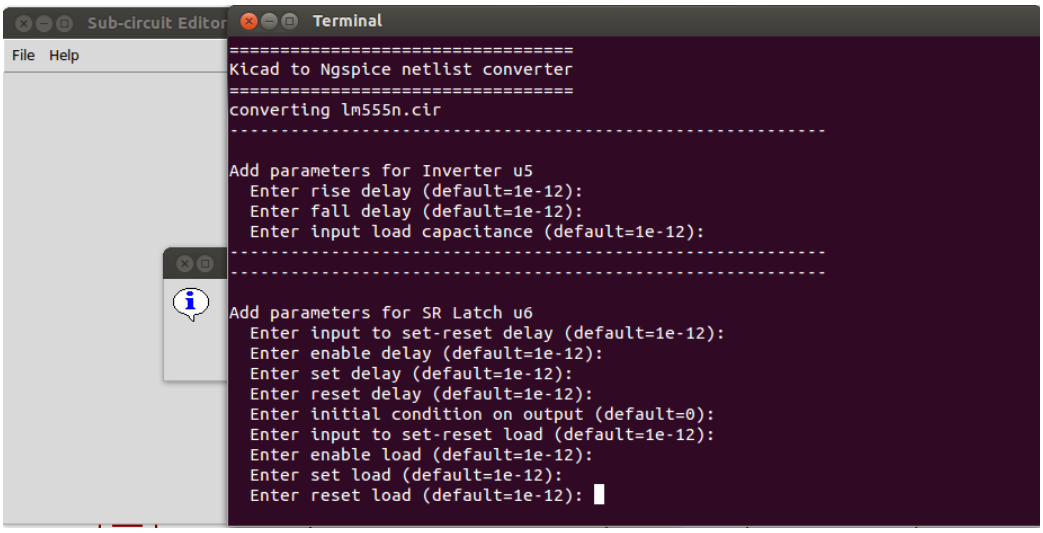

Figure 7.7: IC 555 timer subcircuit - parameters

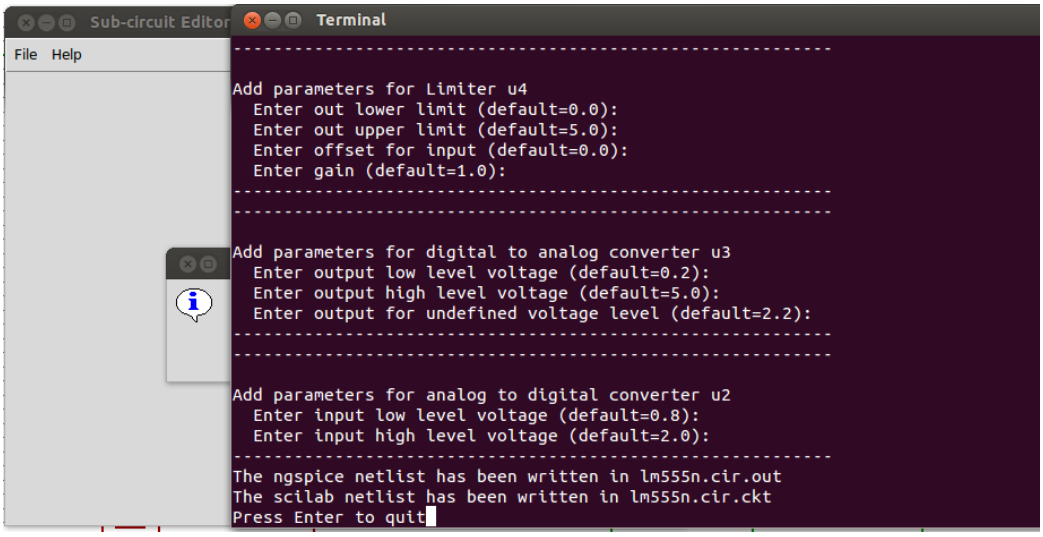

Figure 7.8: Netlist conversion

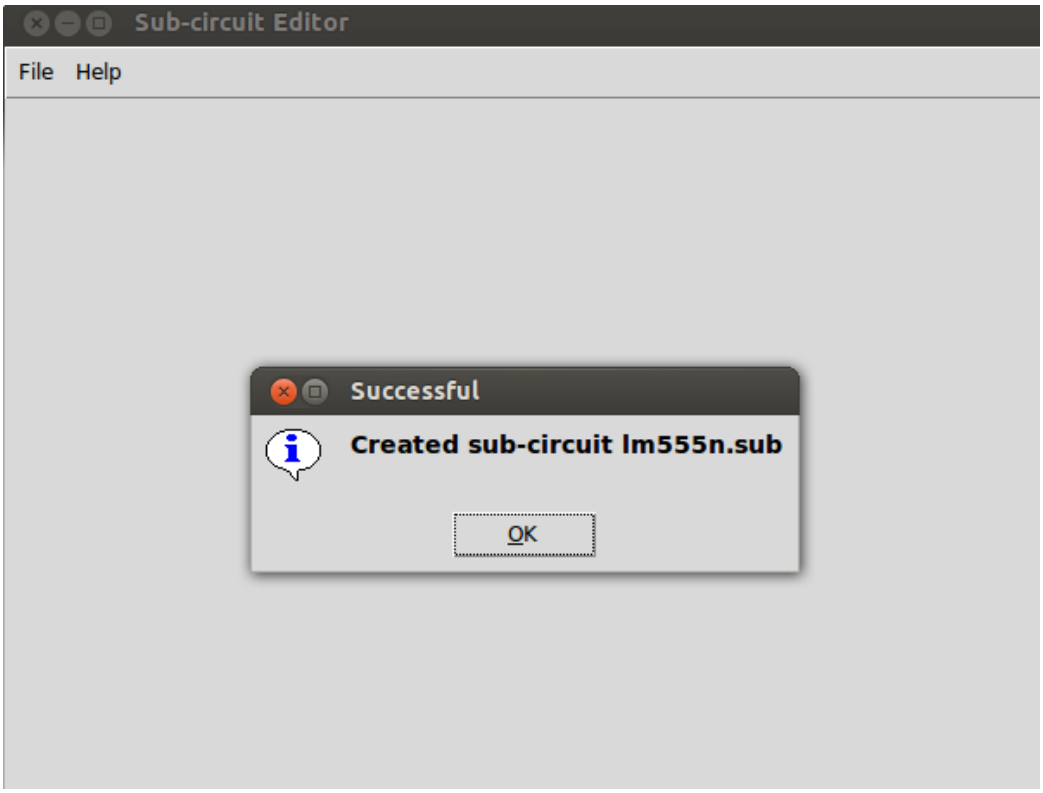

Figure 7.9: Subcircuit creation successfull

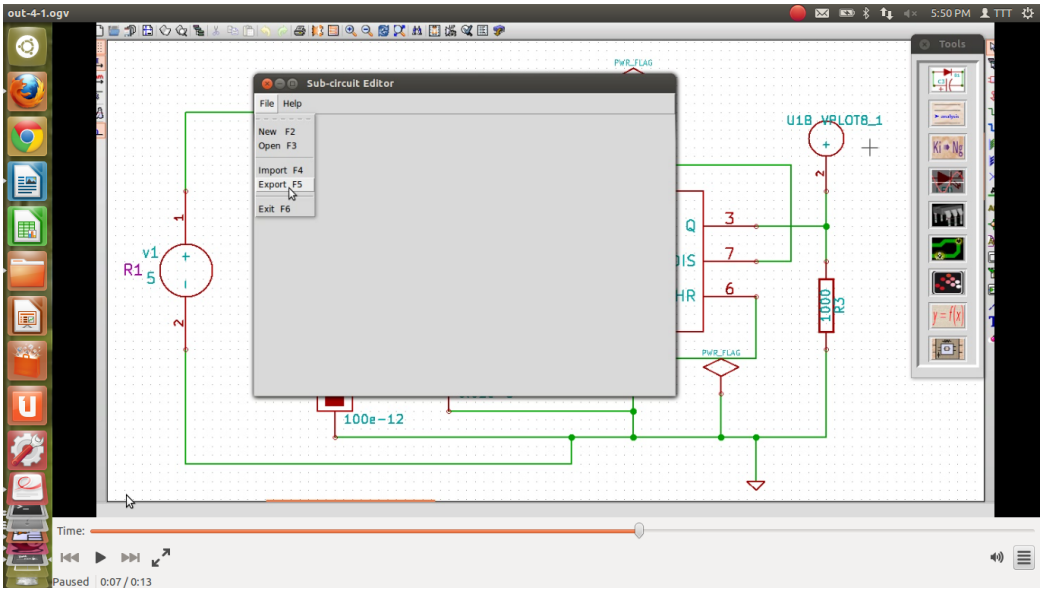

Figure 7.10: Export Subcircuit

Chapter 8

Scilab Based Circuit Simulation

Electronic circuit simulation uses mathematical models to replicate the behavior of an electronic circuit. Unfortunately, no simulator gives the system of equations it solves, in order to get the solution. In Oscad there is an option to simulate the circuit using SMCSim(Scilab Based Mini Circuit Simulator)An important feature of SMCSim is that it gives the system of equations for the circuit under test. The SMCSim works in three modes: normal, symbolic and numerical mode. In normal mode, SMCSim solves the circuit and gives the final output. In symbolic mode, it gives symbolic equations along with the result. In numerical mode, it gives symbolic equations, intermediate numerical values of the components and entries in system matrices, and the final output. Here, we present the working and implementation of SMCsim with an example.

Consider Half-Wave Rectifier with Filter shown in Figure 8.1.The circuit is drawn using Eeschema integrated with Oscad and spice compatible netlist is generated using Oscad circuit simulation tools. The generated netlist is given below. Note that this netlist is generated for SMCSim which has a simple model implementation for a diode. Thus, users need to specify only Is and Vt values.

```
Half-Wave Rectifier
V1 1 0 sine (5 50)
D1 1 2 mymodel (1e-8 0.026)
R1 2 0 10000
C1 2 0 10e-3
```

```
.tran 0 100 0.5
.plot v(1) v(2)
.end
```

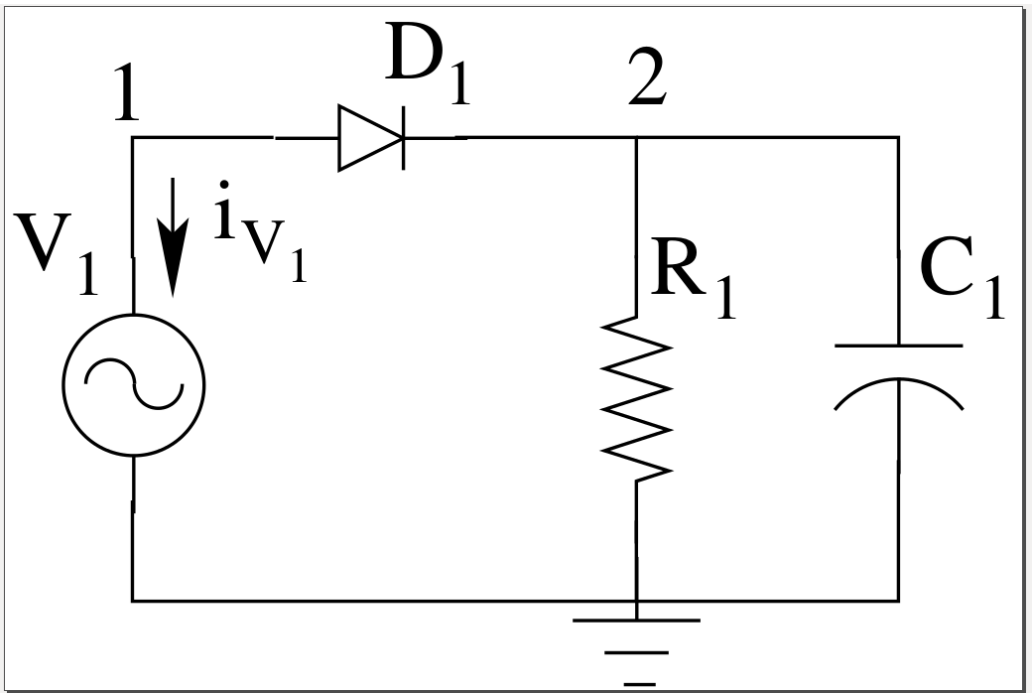

Figure 8.1: Bridge rectifier circuit

SMCSim firstly read the netlist and create a graph corresponding to it using Scilab based graph library metanet. Then circuit is translated into circuit equations using a circuit equations formulation method. All the methods of formulating circuit equations use Kirchhoffs current and voltage equations (KCE and KVE), and device characteristics constraints but differ in the manner in which these constraints are imposed [11]. We have used Modified Nodal Analysis (MNA) [12] as it is applicable to all kinds of electrical circuits. Using MNA m ethod and graph operations, we have efficiently built the circuit equations. The system of Equations representing the electrical circuit shown in Figure 4 is given below. SMCSim has capability to display these equations.

———————————————————————-

iV1 + D1f (v1 , v2) = 0
(R1)v2 + (C1) (dv2/dt)+ - D1(v1, v2)=0

v1=V1

+ D1f (v1 , v2) = 0
dt
v1 = V1

Dnf (va , vb) = Isn (1 e(va vb)/vtn)
where Isn =reverse saturation current and vtn =threshold voltage of diode n

———————————————————————

Now, we explain how SMCSim performs Operating point (DC) analysis and Transient analysis. (DCI)

8.1 Operating Point (DC) Analysis

: A circuit can reach an equilibrium point only when stimulus is constant. So, first step of operating point analysis is to configure the independent sources such that they are constant[13]. Since all waveforms are constant-valued at equilibrium points, dv/dt = 0 and di/dt = 0 and so capacitors act as open circuits and inductors act as short circuits. Thus, for operating point analysis, SMCSim removes the time-dependent components properly. The equations that describe the resulting system are nonlinear and algebraic and solution gives equilibrium point. The equations for the circuit are given below:

———————————————————————-

iV1 + D1f (v1 , v2) = 0 (4)
(R1)v2 + D1f (v1 , v2) = 0 (5)
v1 = V1 (6)
Dnf (va , vb) = Isn (1 e(va vb)/vtn)

where Isn =reverse saturation current and vtn =threshold voltage of diode n

———————————————————————

Generally for nonlinear devices, a linear model is con- structed that is valid only locally around a point [14]. We have used Newton-Raphson method to construct a linear model for a nonlinear devices. In the example, diode D1 is a nonlinear device. SMCSim constructs the linear model for diode D1 as shown in the Figure 8.2. Note that the value of resistor and current source changes at every iteration and SMCSim allows user to observe the value. This is very useful for debugging the circuit when the simulation is not converging. The system of equations representing the linearized electrical circuit is given below:

(RD1)v1 + (RD1)v2 + iV1 = iD1 (7)
(RD1)v1 + (RD1 + R1)v2 = iD1 (8)
v1 = V1 (9)

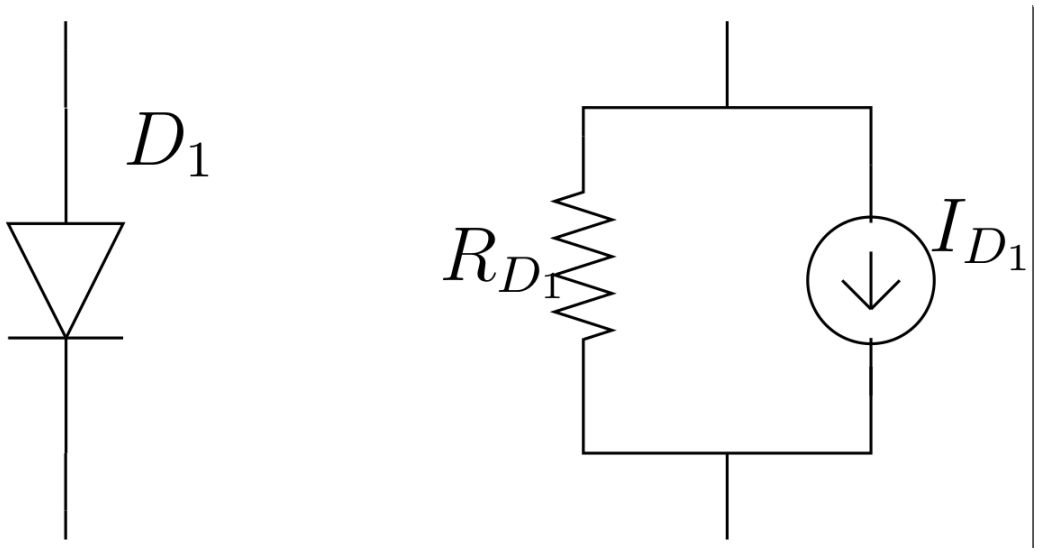

Figure 8.2: Diode model linearisation

8.2 Transient Analysis:

In transient analysis, time dependent components are discretized, i.e., for dynamic devices, a static model is constructed, using a numerical integration method, that is valid for a particular time point. The circuit contains capacitor C1 , as a dynamic device. To solve the circuit, SMCSim constructs a static model for the capacitor C1 using Backward Euler method and performs operating point analysis for a time instant t. The operating point solution gives the solution at time instant t. Note that for each time instant, the values of the static model and the voltage source change.

Now let us take an example of Bridge Rectifier circuit shown in the schematic given in Figure 8.3 and see how to do Scilab based circuit simulation

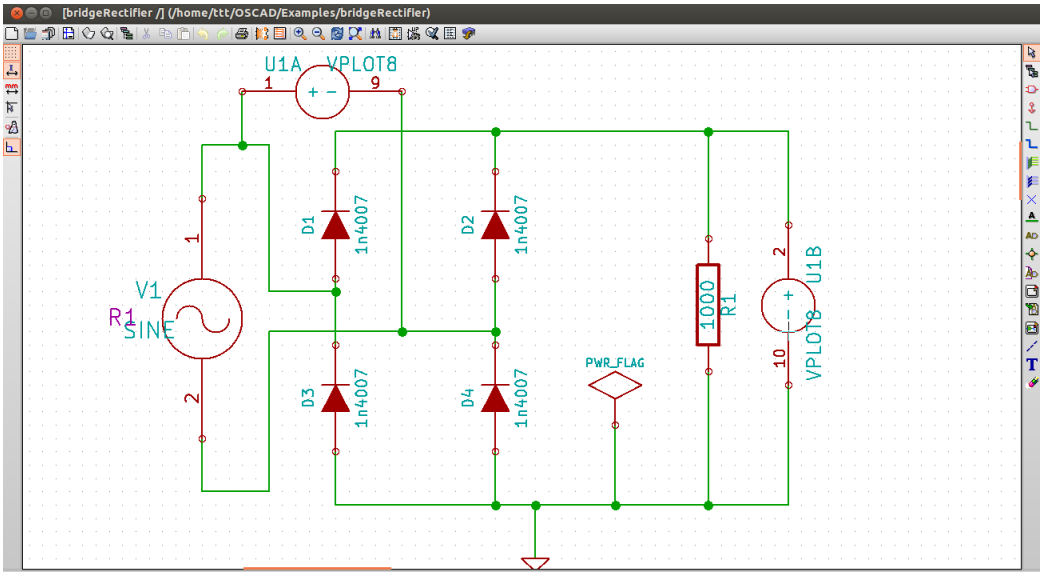

Figure 8.3: Bridge rectifier circuit schematic

To go for Scilab based simulation you will have to follow all the steps of simulations explained in the chapter ? on simulations and finally click on the button of SMCSim on the Oscad Tool Bar as shown in Figure 8.4.

And then you will see the the output of the Scilab based simulations as shown in Figure 8.5.

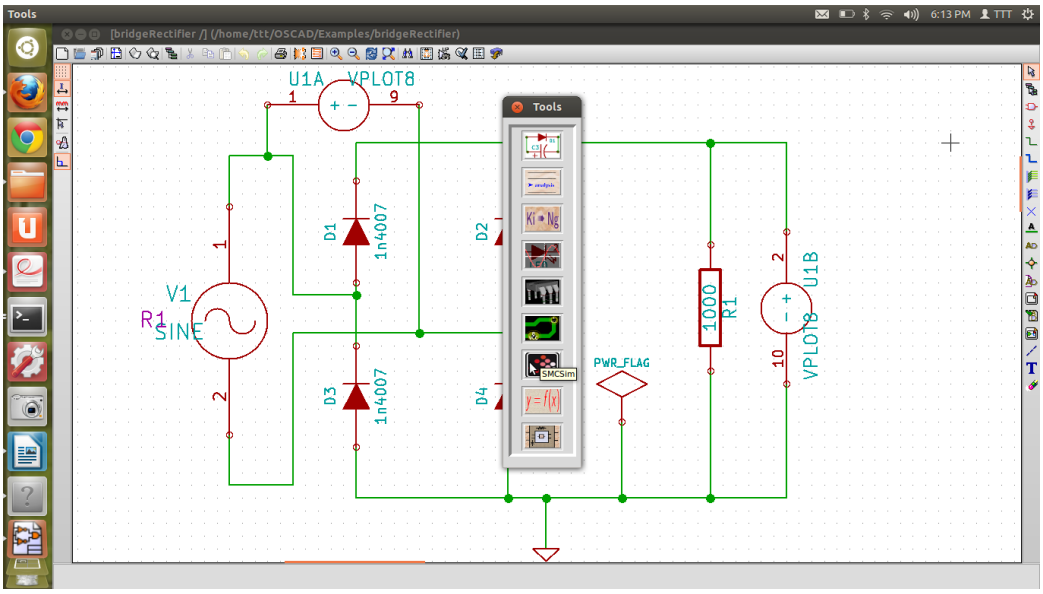

Figure 8.4: Select SMCSim from Oscad toolbar

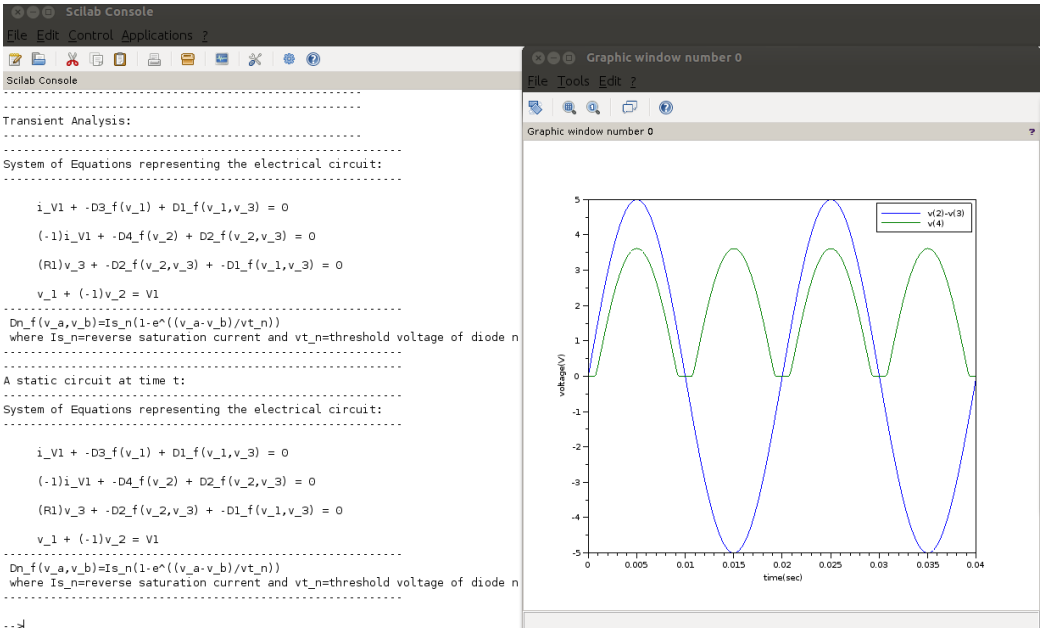

Figure 8.5: Scilab based simulation output

Chapter 9

Oscad Spoken Tutorials

Apart from learning how to use Oscad from this book, one can also refer to spoken tutorials (audio-video tutorials) created for Oscad. These tutorials are basicaly made for self learning and are very clear and detailed. Spoken Tutorial based SELF workshops on OSCAD comprises the following tutorials in the order mentioned below. They are categorized in to two levels, Beginner Level and Advanced Level.

9.1 Beginner Level

The beginner level has a set of tutorials on OSCAD and KiCad. As OSCAD uses KiCad for schematic creation and PCB layout design, the supporting tutorials on KiCad enhances the user's understanding of OSCAD.

9.1.1 Introduction to Oscad

This tutorial aims at making the users get started with OSCAD. It covers the following:

- In this tutorial installation of OSCAD through a shell script is shown.
- This script installs all the requisite software like Ngspice, KiCad, Scilab and Python.
- After installation of all software, an already created schematic of an RC filter is opened.

- A test run of Oscad is done using this circuit.

9.1.2 Schematic creation and simulation using Oscad

This tutorial teaches how to create circuit schematic and simulate it using Oscad. A simple RC filter circuit is used as an example. The following sequence is adopted in the tutorial.

- Required components are choosen from their corresponding libraries and placed in the schematic editor.
- Components are connected together, annotated and values are assigned to them.
- Electric Rules Check is done and erroneous connections are corrected, if any.
- Spice netist is generated and is converted to NGSpice format.
- Circuit simuation is doen using NGSpice.

9.1.3 Designing Circuit Schematic in KiCad

- This tutorial introduces KiCad.
- It teaches how to create a circuit schematic using EESchema and annotate various components in the schematic.
- Astable multivibrator circuit is used as an example.
- An assignment is given in the end for practice.

9.1.4 Designing Printed Circuit Board using Oscad

- Create netlist for PCB from schematic.
- Map footprints to components.
- Generate PCB layout.
- PCB layout of RC filter is created in this tutorial.

9.1.5 Electric rule checking and Netlist Generation in KiCad

This tutorial teaches the following

- To assign values to components in the astable multivibrator circuit schematic created in the previous tutorial
- To perform electric rule check.
- To generate netlist for designing PCB layout

9.1.6 Mapping components in KiCad

- This tutorial explains how to map components in a schematic with corresponding footprints.
- Cvpcb, the footprint editor in Kicad, is used to explain the same.
- Every component in the astable multivibrator circuit schematic is assigned a footprint

9.1.7 Designing PCB in KiCad

- In this tutorial, printed circuit board layout of the astable multivibrator circuit is created.
- It also explains how to lay the tracks, modify the width of the tracks etc.
- Layer selection and track routing are also covered.

9.2 Advanced Level

Advanced level has tutorials on Ngspice, model buildng using Oscad and subcircuit creation using Oscad. As Oscad uses Ngspice for simulation, the set of tutorials on Ngspice helps the user to know more about how simulations are done in Oscad.

9.2.1 Operating point analysis in Ngspice

- This tutorial explains how to perform operating point analysis.
- It shows how to verify Kirchoff's voltage law using ngspice in, interactive mode using commandline interface & using command script included in netlist.

9.2.2 DC sweep analysis in Ngspice

This tutorial covers the following

- How to perform DC sweep analysis.
- How to perform nested DC sweep analysis using two sweep variables.

9.2.3 Model building using Oscad

In this tutorial, we show how to build a model for a diode. This includes

- Opening an already created circuit schematic of bridge rectifier
- Building/editing the 1N4007 diode model present in the bridge rectifier circuit using model builder tool.
- This is explained using a bridge rectifier circuit which contains 1N4007 diode.

9.2.4 Subcircuit creation using Oscad

We show how to create and edit a subcircuit. This is explained using astable multivibrator circuit that has 555 timer IC as a subcircuit. The tutorial covers the following:

- An already created astable multivibrator schematic is opened to show the component 555 timer in it.
- As 555 timer will be modelled as a subcircuit, the subcircuit schematic of 555 timer is shown next.
- The tutorial then shows how to edit the 555 timer subcircuit schematic.

9.3 Instruction Sheet

This section should be used as set of instructions provided you have the Oscad spoken tutorials CD/DVD with you.

9.3.1 The procedure to practice

- You have been given a set of spoken tutorials and files.
- You will typically do one tutorial at a time.
- You may listen to a spoken tutorial and reproduce all the commands shown in the video.
- If you find it difficult to do the above, you may consider listening to the whole tutorial once and then practice during the second hearing.

9.3.2 Please ensure

- You have Linux Ubuntu 12.04 OS or above installed on your computer.
- You have a working internet connection on your computer.
- You have basic `knowledge about Linux` to follow these tutorials.

9.3.3 Basic Module

- Right-click on the file named `index.html`, and choose `Open with Firefox` to open this file in the Firefox web browser.
- Read the instructions given.
- In the left hand side panel you will see `Basic Level`.
- Please click on the module `Basic Level`.
- In this module, there are a few tutorials.
- `Introduction to Oscad` teaches how to install Oscad and test run Oscad using an example.
- Click on it. You will see the video in the centre.

- Click on the play button on the player to play the tutorial.
- To view the tutorial in a bigger player, right-click on the video and choose `View Video` option.
- Adjust the size of the player in such a way that you are able to practice in parallel.
- Follow the tutorial and reproduce all the activities as shown in the tutorial.
- Now you will have Oscad installed and working on your computer.

9.3.4 Schematic creation and Simulation

- Locate the next topic `Schematic creation and Simulation`.
- Click on it. Follow the tutorial and reproduce all the activities as shown in the tutorial.
- Please save your project files that you will create while you practice this tutorial.
- Guidelines for saving your work are as follows-

  Instructions for Practice

  - Create a folder on the `Desktop` with your Name-RollNo-Component `(Eg.vin-04-Oscad)`.
  - Give a unique name to the files you save, so as to recognize it next time. `(Eg. Practice-1-Oscad)`.
  - Remember to save all your work in your folder.
  - This will ensure that your files don't get over-written by someone else.
  - Remember to save your work from time to time instead of saving it at the end of the task.

Instructions for Assignments

  - Attempt all the given assignments.
  - Save your work by creating a folder called `Oscad-Assignment` in your main folder.

- At 09:37 the video says that you have to watch KiCad tutorial - `Designing Circuit schematic in KiCad`.
- Locate this tutorial on the left hand panel and watch it.
- Reproduce the astable multivibrator circuit schematic shown in it using Oscad.
- After you finish this tutorial, locate the next tutorial `Designing Printed Circuit Board`.

9.3.5 Designing Printed Circuit Board

- Click on the next topic `Designing Printed Circuit Board`.
- You will need to use the practice files created in the previous tutorial.
- Follow this tutorial and reproduce all the activities as shown.
- At 08:50 the video says that you have to watch KiCad tutorials -
  1. Electric rule checking and netlist generation.
  2. Mapping components in KiCad.
  3. Designing printed circuit board in KiCad.
- Locate these tutorials on the left hand panel and watch it.
- Reproduce the layout of astable multivibrator shown in it using Oscad.

Bibliography

[1] Steven M. Sandler and Charls Hymowitz. *SPICE Circuit Handbook*. 2006.

Index