Oscad
An open source EDA tool for circuit design, simulation, analysis and PCB design

Yogesh Save, Rakhi R, Shambhulingayya N. D.
Rupak M. Rokade, Ambikeshwar Srivastava
Manas Ranjan Das, Lavitha Pereira
Sachin Patil, Srikant Patnaik
Kannan M. Moudgalya

Indian Institute of Technology Bombay

May 2013

To

Mr. Narendra Kumar Sinha, IAS
An Electronics Engineer and a Bureaucrat,
Who dreamt of educating all Indians through NMEICT and
Who envisioned and made possible the Aakash Tablet

Contents

Preface

Seeds for Oscad were sown when the National Mission on Education through ICT (NME-ICT) was launched: the mission document identified *Adaption & deployment of open source simulation packages equivalent to Matlab, OrCAD, etc.*, as one of the areas NME-ICT would concentrate on. The FOSSEE (free and open source software in science and engineering education) group at IIT Bombay, of which we are a part of, initially started working on Python and Scilab. The Standing Committee of NMEICT encouraged us to contribute to other open source software as well. This push helped us develop Oscad, an open source alternative to OrCAD.

Oscad is an electronic design automation (EDA) tool, developed using KiCad, Ngspice and Scilab. We have made the netlist files generated by KiCad suitable for simulation through Ngspice. In order to provide an explanation facility, we have developed a method to automatically generate differential equations that describe a given analog circuit and to solve them using Scilab. Once satisfied with simulation results, the user can create a Gerber file for PCB fabrication.

While working on Scilab and Python, the FOSSEE group, jointly with the Spoken Tutorial team, created a large number of Spoken Tutorials [2]. Spoken Tutorials are audio-video tutorials in the IT and simulation areas, created for self learning using screencast technology. This instructional material has been used to train more than 20,000 college students on Scilab and Python in the past two years.

We have created seven spoken tutorials of ten minutes each, using which, a beginner level SELF workshop can be conducted on Oscad. We plan to conduct these workshops in about 100 colleges in the next one year, free of cost.

The FOSSEE team has also created more than 160 Scilab Textbook Companions, each of which contains Scilab code for worked out examples of standard textbooks, mostly in engineering and science. These have been created by the students and professors from various educational institutions in India. These textbooks can be downloaded free of cost from [3]. They can also be executed remotely on GARUDA cloud [4].

We are embarking on a similar methodology for Oscad as well: we have solved most of the worked out examples of [1] and given the solution in Appendix A. We hope to create Oscad Textbook Companions for all other relevant standard textbooks as well in the near future, once again through students and other volunteers.

Solving the worked out examples of [1] was a good exercise, as it helped identify and include some missing features. The yet to be created Oscad Textbook Companions are expected to help in this regard, while simultaneously increasing the available documentation.

Lab migration is another important activity that the FOSSEE team is involved in. It provides equivalent Scilab code for Matlab based labs. This is also carried out through students and volunteers. We are starting this activity for Oscad as well: we will try to provide equivalent Oscad based solution to all circuit design labs that currently use proprietary software.

We have successfully ported Oscad on Aakash, the world's lowest cost computing tablet. As Ubuntu 12.10 runs on native mode on Aakash, we could port Oscad to it. Chapter 11 explains this activity, along with a few screenshots. As the Aakash tablet costs Rs. 2,263, and hence, for less than Rs. 2,500 (including a keyboard and a mouse), one can get access to a powerful EDA system. This is expected to help the students who are enthusiastic about circuit design, but cannot afford expensive hardware and software.

Porting of Oscad demonstrates the power of the concept of Aakash: an unlimited number of open source educational software systems can be made available even in a low cost device. Aakash can serve the dual purpose of a tablet and a computing device. This is the only way to address the aspirations of the millions of poor students who cannot afford even a computer system or an expensive tablet, let alone both.

The FOSSEE team is currently working on the promotion/development of the following open source software systems as well: 1. OpenFOAM, a CFD solver and an open source alternative to Fluent and StarCD. 2. COIN-OR, an open source software suite for optimisation problems. 3. OpenFormal for formal verification of computer software. About ten professors and 25 full time staff members and students are working on FOSSEE projects at IIT Bombay. Many more are expected to join in the near future.

Another important project supported by NMEICT is the Teach 10,000 Teachers (T10KT) programme. This methodology, pioneered at IIT Bombay [5, 6] has demonstrated that it is possible for the best people in the field to provide extremely high quality training to a large number of learners simultaneously. Oscad is expected to be used in the forthcoming T10KT course on Analog Electronics, organised by IIT Kharagpur [7].

We invite all EDA enthusiasts to work with us through the following resources: 1. URL for all FOSSEE activities: http://fossee.in 2. URL for all Oscad resources: http://oscad.in 3. Textbook companion: textbook-companion@oscad.in 4. Lab migration: lab-migration@oscad.in 5. SELF workshops: SELF-workshop@oscad.in 6. Oscad development and enhancing its capabilities: Oscad-dev@oscad.in 7. Feedback on this book: Oscad-textbook@oscad.in. We also hope to establish forum based discussion services for Oscad.

Finally, an electronic version of this book is available for noncommercial purposes at http://oscad.in.

Acknowledgements

We would first like to thank Mr. N. K. Sinha, IAS, for without him, there would have been no National Mission on Education through ICT (NMEICT), without which, there would have been no FOSSEE, without which, there would have been no Oscad. The idealistic guiding principles of NMEICT, namely, reliance on open source software, providing free access to e-content, Internet connectivity for all educational institutions and providing a low cost access device to every student through Aakash, egged us to contribute our best and one of the outcomes is Oscad.

We would like to thank the former Human Resource Development Minister (HRM) Mr. Arjun Singh for getting NMEICT started. We would like to acknowledge the former HRM Mr. Kapil Sibal for his unstinting support and the faith he had in the NMEICT administration team. We would like to thank the current HRM Dr. Pallam Raju for extending the tenure of NMEICT by five more years.

We want to thank the Members of the Standing Committee of NMEICT who met once in two weeks for almost two years to review project proposals and to recommend them for funding or giving suggestions for improvement. We also want to thank them for urging us to work on more FOSS systems than what we were prepared for. Without this kind of active support, the ecosystem required for projects like Oscad to flourish, established at IIT Bombay through the many projects funded through NMEICT, would not have materialised.

We want to thank the FOSSEE faculty members Profs. Prabhu Ramachandran, Madhu Belur, Mani Bhushan, Shiva Gopalakrishnan, Jayendran Venkateswaran, Ashutosh Mahajan and Supratik Chakraborty for establishing a vibrant FOSSEE group at IIT Bombay. We want to thank Prof. D. B. Phatak for being a constant source of inspiration and encouragement and for supporting our activities directly and indirectly through the Teach 10,000 Teacher Programme [5] and the Aakash [8] Project. We want to thank other faculty members with NMEICT projects at IIT Bombay, namely, Profs. Kavi Arya, Ravi Poovaiah, Santosh Noronha, Anil Kulkarni, Sridhar Iyer, Sahana Murthy and Shishir Jha for sharing their dreams, processes and facilities. We want to thank the staff members of all NMEICT projects at IIT Bombay in general and of FOSSEE and Spoken Tutorial projects in particular, for providing a wonderful work environment.

We want to thank the IIT Bombay administration in general and R&D office in particular for providing us with an excellent environment to make us work efficiently. We want to thank the researchers and faculty members in our departments for providing us with necessary space and for putting up with our tantrums.

We would like to thank the professors, staff and students affiliated with the Wadhwani Electronics lab at IIT Bombay for trying out Oscad in lab courses and for the useful suggestions. We would like to thank Abhishek Pawar for creating Spoken Tutorials on KiCad. We would like to thank Saket Choudhary for making the netlist files

generated by KiCad compatible with Ngspice. We want to thank Hardik for his help in implementing the current GUI of Oscad. We want to thank Kiran for designing the logo of Oscad. We want to thank Bella for helping with the coordination of FOSSEE in general and Oscad in particular. We want to thank Mr. Sunil Shastri of Shroff Publishers for bringing out this book in a short time.

Finally, we want to thank our family members for allowing us to work extended hours and for bearing with us.

Yogesh Save
Rakhi R
Shambhulingayya N. D.
Rupak M. Rokade
Ambikeshwar Srivastava
Manas Ranjan Das
Lavitha Pereira
Sachin Patil
Srikant Patnaik
Kannan M. Moudgalya

IIT Bombay
22 May 2013

List of Acronyms

ADC	Analog to Digital Converter
BJT	Bipolar Junction Transistor
BV	Breakdown Voltage
CCCS	Current Controlled Current Source
CCVS	Current Controlled Voltage Source
CPU	Central Processing Unit
DAC	Digital to Analog Converter
DRC	Design Rules Check
DXF	Drawing Interchange Format or Drawing Exchange Format
EDA	Electronic Design Automation
ERC	Electric Rules Check
FOSS	Free and Open Source Software
FPGA	Field Programmable Gate Array
gEDA	Electronic Design Automation released under GPL
GUI	Graphical User Interface
HDL	Hardware Descrition Language
HPGL	Hewlett-Packard Graphics Language
IC	Integrated Circuit
ICT	Information and Communication Technology
IGBT	Insulated Gate Bipolar Transistor
JFET	Junction Field Effect Transistor
KCE	Kirchoff's Current Law
KVE	Kirchoff's Voltage Law
LXDE	Lightweight X11 Desktop Environment
MNA	Modified Nodal Analysis
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
NMEICT	National Mission on Education through ICT
Op-amp	Operational Amplifier
OTC	Oscad Textbook Companion
PCB	Printed Circuit Board
RS	Ohmic Resistance
SELF	Spoken Tutorial based Education and Learning through Free FOSS study
SMCSim	Scilab based Mini Circuit Simulator
SVF	Serial Vector Format
T10KT	Teach 10,000 Teachers
VCCS	Voltage Controlled Current Source
VCVS	Voltage Controlled Voltage source

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 are called `Electronic Design Automation` or `EDA` tools.

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 be 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. The final product is again tested for specifications. The whole cycle is repeated until desired results are obtained [9].

A person who builds an electronic system has to first design the circuit, produce a virtual representation of it through a schematic for easy comprehension, simulate it and finally convert it into a Printed Circuit Board (PCB). There are various tools available that help do this. Some of the popular EDA tools are those of `Cadence`, `Synopys`, `Mentor Graphics` and `Xilinx`. Although these are fairly comprehensive and high end, their licences are expensive, being proprietary.

There are some free and open source EDA tools like `gEDA`, `KiCad` and `Ngspice`. The main drawback of these 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`). To the best of our knowledge, there is no open source software that can perform circuit design, simulation and layout design together. Oscad is capable of doing all of the above.

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 open source software packages, such as

KiCad, Ngspice, Scilab and Python. Using Oscad, one can create circuit schematics, perform simulations 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. Because of these reasons, Oscad is expected to be useful to students, teachers and other professionals who would want to study and/or design electronic systems. Oscad is also useful for entrepreneurs and small scale enterprises who do not have the capability to invest in heavily priced proprietary tools.

This book introduces Oscad to the reader and illustrates all the features of Oscad with examples. Chapter 2 gives step by step instructions to install Oscad on a typical computer system and to validate the installation. The software architecture of Oscad is presented in Chapter 3. Chapter 4 gets the user started with Oscad. It takes them 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 7 explains PCB design using Oscad, in detail. The advanced features of Oscad such as Model Builder and Subcircuit Builder are covered in Chapter 8. Scilab based circuit simulator is covered in Chapter 9. Chapter 10 describes the spoken tutorials on Oscad and contains instructions to use them. Oscad is a light weight software and it has been ported to Aakash tablet. Chapter 11 explains about Oscad on Aakash. Appendix A presents examples, that have been worked out using Oscad, from the book `Microelectronic Circuits` by Sedra and Smith [1]. Appendix B explains the resources available for the use and promotion of Oscad.

The following convention has been adopted throughout this book. All the menu names, options under each menu item, tool names, certain points to be noted, etc., are given in *italics*. Some keywords, names of certain windows/dialog boxes, names of some files/projects/folders, messages displayed during an activity, names of websites, component references, etc., are given in `typewriter` font. Some key presses, e.g. `Enter` key, `F1` key, `y` for yes, etc., are also mentioned in `typewriter` font.

Chapter 2

Installing and Setting up Oscad

The step by step instructions to install Oscad are given below. Installation script for Oscad is written in Bash. This makes the installation efficient and user friendly. Before starting the installation, one should ensure that all the system and installation requirements, and the prerequisites given below, are met. This chapter is presented in conversational style.

System requirements: The following are required.

- Ubuntu 12.04 OS (64-bit/32-bit)
- Scilab 5.4.0 or above (Optional)

Installation requirements: The following are required.

- working Internet connection
- admin (or root) privileges
- working knowledge and availability of Synaptic Package Manager [10]

Prerequisites: The following are required.

- Basic knowledge of analog and digital electronics
- Basic knowledge of Linux shell commands

Linux commands typed on terminal during installation are given in boxes with round corners

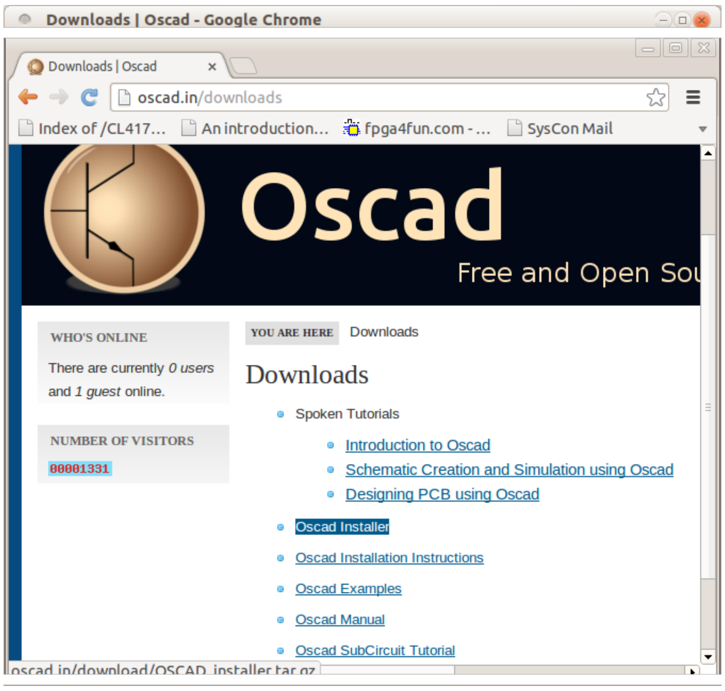
2.1 Procedure for installing Oscad

Step 1: Download Oscad and examples: Go to `http://www.oscad.in/downloads`. Fig. 2.1 shows the downloads page of Oscad website. Do the following:

1. Click on *Oscad Installer* and save the file, `OSCAD_installer.tar.gz`.
2. Click on *Oscad Examples* and save the file, `Examples.tar.gz`, in the same folder as above.

Step 2: Download Scilab:

Note: Installation of Oscad can be done without Scilab



WHO'S ONLINE

There are currently *0 users* and *1 guest* online.

NUMBER OF VISITORS

00001331

YOU ARE HERE Downloads

Downloads

- Spoken Tutorials
  - Introduction to Oscad
  - Schematic Creation and Simulation using Oscad
  - Designing PCB using Oscad
- Oscad Installer
- Oscad Installation Instructions
- Oscad Examples
- Oscad Manual
- Oscad SubCircuit Tutorial

Figure 2.1: Oscad website

Scilab is used for simulating the circuit and giving symbolic equations

If one does not need Scilab, skip this step and proceed to step 3

Go to `http://www.scilab.org/`. Click on the *Download Scilab* option on the home page. Save the file in the same folder where `OSCAD_installer` and `Examples` are saved. Fig. 2.2 shows the Download Scilab option on the Scilab web page.

One can skip this step, if scilab 5.4.0 or above is already available in the system

Step 3: Extract the downloaded files: Go to the folder where `OSCAD_installer`, `Examples` and Scilab (Optional) files are saved. Select all, right click and choose *Extract Here* as shown in Fig. 2.3.

Step 4: Navigate to folder: Go to the directory where all the three folders are saved and extracted. To do this, open a terminal window and type:
`cd folder-where-downloaded-files-are-saved`. In this command, replace folder-where-downloaded-files-are-saved with the complete path of the folder where the downloaded files are saved and extracted. Press `Enter`. Now check whether you have navigated to desired folder by typing `ls` and pressing `Enter`. Fig. 2.4 shows navigation to the folder having all files.

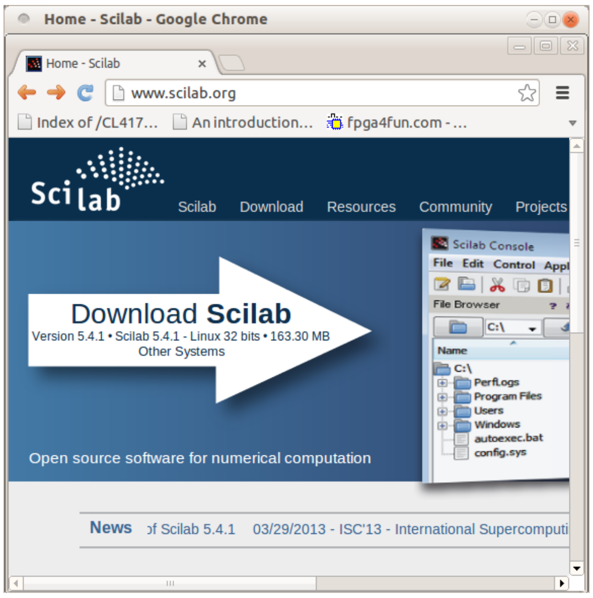


Figure 2.2: Scilab website

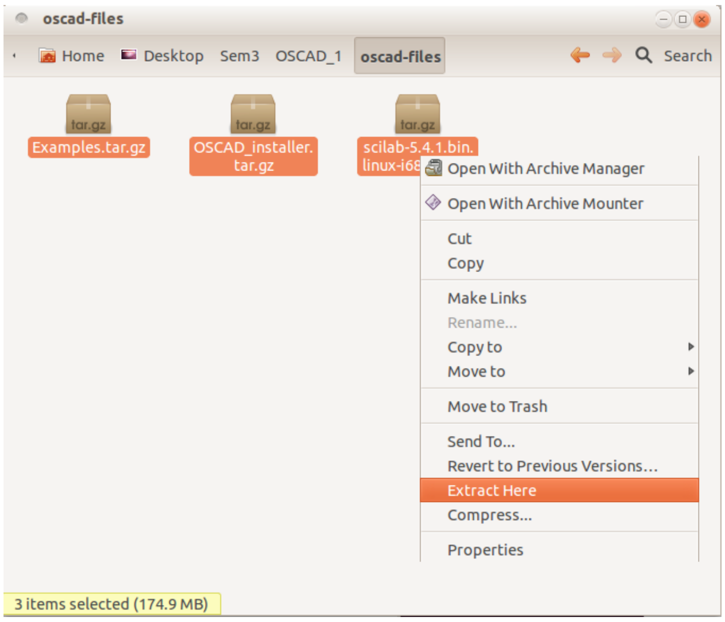
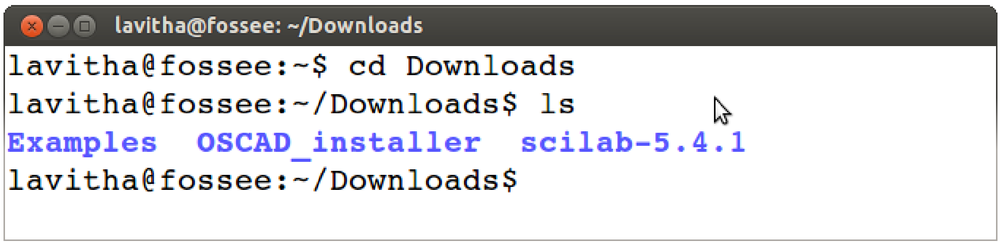


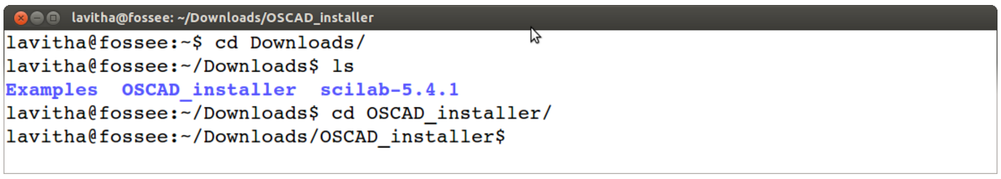
Figure 2.3: Extracting all files

Step 5: Navigate to `OSCAD_installer`: To do this type $\boxed{\texttt{cd OSCAD\_installer}}$ and press `Enter`. Fig. 2.5 shows that we have navigated to OSCAD_installer folder.



```
lavitha@fossee:~$ cd Downloads
lavitha@fossee:~/Downloads$ ls
Examples   OSCAD_installer   scilab-5.4.1
lavitha@fossee:~/Downloads$
```

Figure 2.4: `ls` command to verify navigation to the required folder



```
lavitha@fossee:~$ cd Downloads/
lavitha@fossee:~/Downloads$ ls
Examples   OSCAD_installer   scilab-5.4.1
lavitha@fossee:~/Downloads$ cd OSCAD_installer/
lavitha@fossee:~/Downloads/OSCAD_installer$
```

Figure 2.5: Changing directory to `OSCAD_installer`

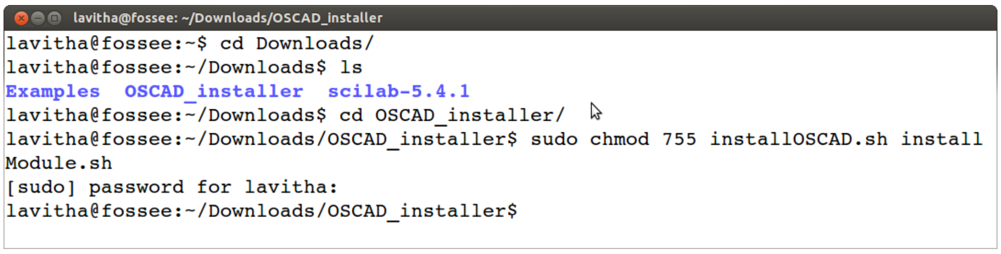
Step 6: Make the `installOSCAD` and `installModule` files executable: To do this, type in the terminal `sudo chmod 755 installOSCAD.sh installModule.sh` and press `Enter`. Type the root password. The terminal should look like as shown in Fig. 2.6.

Step 7: Begin installation: To begin the installation, type `./installOscad.sh` and press `Enter`. The terminal prompts for proxy settings as shown in Fig. 2.7.

Note: If one is not behind proxy, type `n` and press `Enter`.

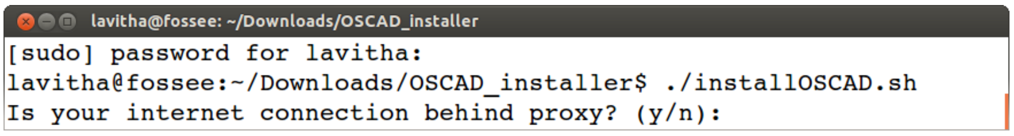
The option to enter proxy parameters will not be displayed.

If `y` is typed, then enter the proxy details as shown in Fig. 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 automatically.



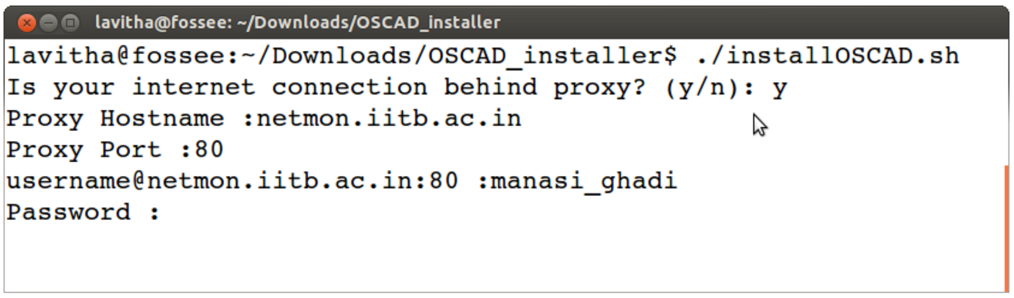
```
lavitha@fossee:~$ cd Downloads/
lavitha@fossee:~/Downloads$ ls
Examples   OSCAD_installer   scilab-5.4.1
lavitha@fossee:~/Downloads$ cd OSCAD_installer/
lavitha@fossee:~/Downloads/OSCAD_installer$ sudo chmod 755 installOSCAD.sh install
Module.sh
[sudo] password for lavitha:
lavitha@fossee:~/Downloads/OSCAD_installer$
```

Figure 2.6: Making files executable using `chmod` command



```
[sudo] password for lavitha:
lavitha@fossee:~/Downloads/OSCAD_installer$ ./installOSCAD.sh
Is your internet connection behind proxy? (y/n):
```

Figure 2.7: Terminal prompt for proxy settings during the installation of Oscad



```
lavitha@fossee:~/Downloads/OSCAD_installer$ ./installOSCAD.sh
Is your internet connection behind proxy? (y/n): y
Proxy Hostname :netmon.iitb.ac.in
Proxy Port :80
username@netmon.iitb.ac.in:80 :manasi_ghadi
Password :
```

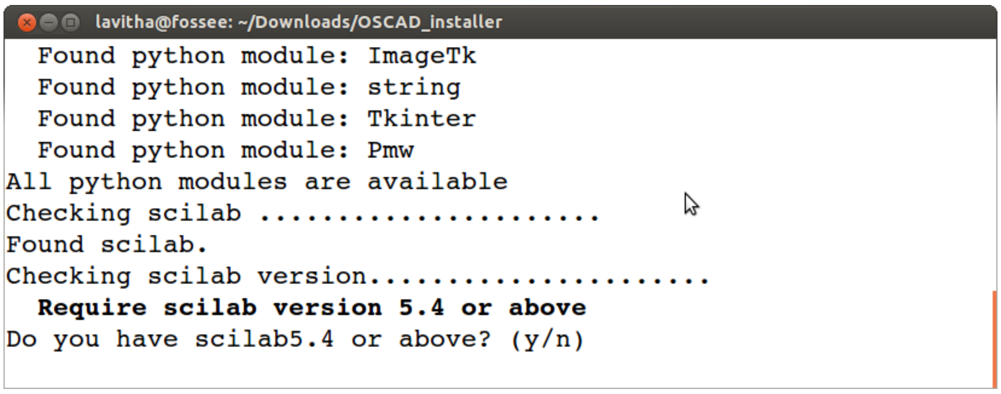
Figure 2.8: Entering proxy details

Note: While installing python modules, one may get error messages

If so, install the missing packages using Synaptic Package Manager [10]

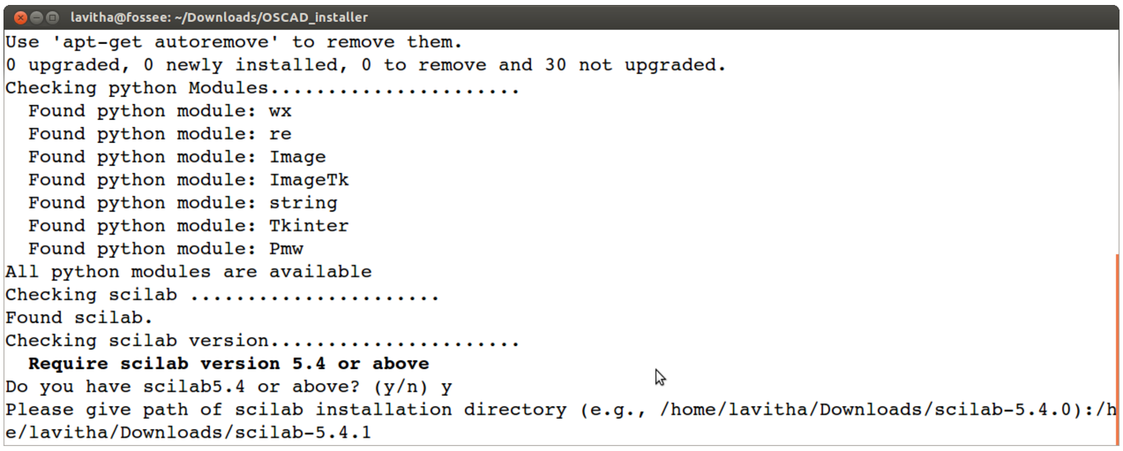
Once this is done, re-run the installOscad.sh script.

Step 8: Linking of Scilab: The prompt displays the message: `Do you have scilab 5.4 or above? (y/n)` as shown in Fig. 2.9. If one types `n` and presses the `Enter` key, it skips the linking of Scilab and goes to step 9. If instead, one types `y` and presses the `Enter` key, one has to type the complete path where Scilab (5.4.0 or



```
  Found python module: ImageTk
  Found python module: string
  Found python module: Tkinter
  Found python module: Pmw
All python modules are available
Checking scilab .....................
Found scilab.
Checking scilab version.....................
  Require scilab version 5.4 or above
Do you have scilab5.4 or above? (y/n)
```

Figure 2.9: Linking Scilab



```
lavitha@fossee: ~/Downloads/OSCAD_installer
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 30 not upgraded.
Checking python Modules.....................
  Found python module: wx
  Found python module: re
  Found python module: Image
  Found python module: ImageTk
  Found python module: string
  Found python module: Tkinter
  Found python module: Pmw
All python modules are available
Checking scilab .....................
Found scilab.
Checking scilab version.....................
  Require scilab version 5.4 or above
Do you have scilab5.4 or above? (y/n) y
Please give path of scilab installation directory (e.g., /home/lavitha/Downloads/scilab-5.4.0):/h
e/lavitha/Downloads/scilab-5.4.1
```

Figure 2.10: Giving directory information to link Scilab

above) has been saved. Example: `/home/lavitha/downloads/scilab-5.4.1.` Press `Enter`. Refer to Fig. 2.10.

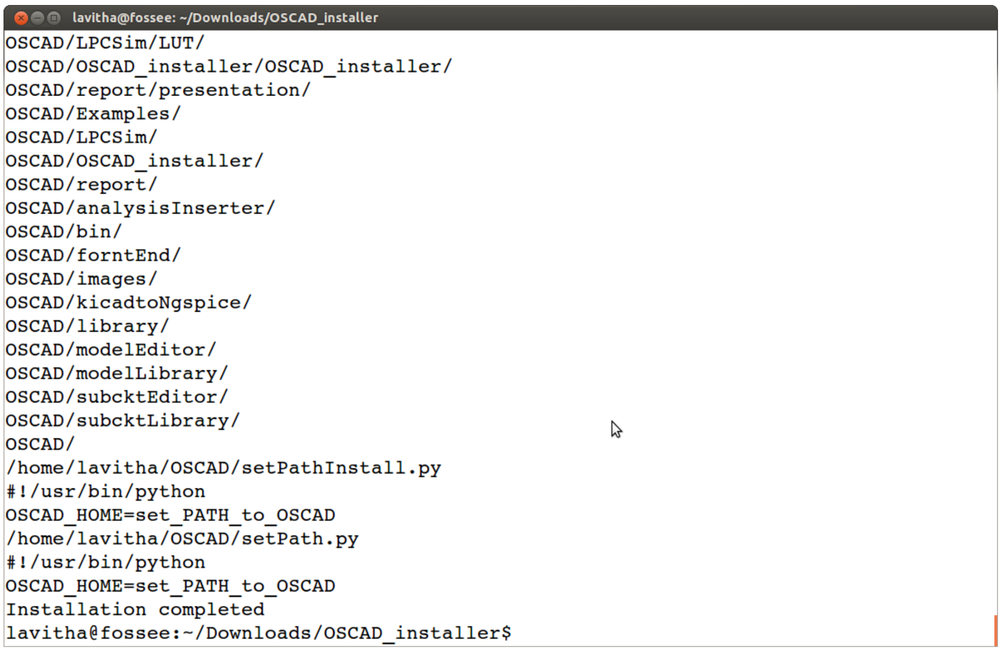
`metanet` library will be installed after this. It takes a couple of minutes.

Step 9: Final installation: The message `Please select installation directory` is displayed. Type the desired location where Oscad should be installed. For example: `/home/lavitha`. Press `Enter`. Once the installation is completed, the message `Installation completed` is displayed as shown in Fig. 2.11. This creates Oscad shortcut on Desktop.

2.2 Testing

In order to check whether Oscad is correctly installed, we propose the following procedure:

- Double click on Oscad shortcut created on Desktop as shown in Fig. 2.12.
- A window is displayed as shown in Fig. 2.13a. Select the option *Run*. It opens the Oscad window as shown in Fig. 2.13b.
- Select *Project* tab at the top left hand corner of Oscad window and then click on *Open*. Browse to the folder where Examples are saved as shown in Fig. 2.14.
- Select `RC` from the examples by double clicking on RC and then click on *OK*.
- `Enter Project name` dialog box opens up as shown in Fig. 2.15. Click on *OK*.
- A toolbar appears as in Fig. 2.16a. Select *Schematic Editor* from the toolbar as shown in Fig. 2.16b.
- An error message as shown in Fig. 2.17a will pop up. Click on *Close*.
- Schematic Editor window appears. Press `F1` to zoom in and press `F2` to zoom out.



```
OSCAD/LPCSim/LUT/
OSCAD/OSCAD_installer/OSCAD_installer/
OSCAD/report/presentation/
OSCAD/Examples/
OSCAD/LPCSim/
OSCAD/OSCAD_installer/
OSCAD/report/
OSCAD/analysisInserter/
OSCAD/bin/
OSCAD/forntEnd/
OSCAD/images/
OSCAD/kicadtoNgspice/
OSCAD/library/
OSCAD/modelEditor/
OSCAD/modelLibrary/
OSCAD/subcktEditor/
OSCAD/subcktLibrary/
OSCAD/
/home/lavitha/OSCAD/setPathInstall.py
#!/usr/bin/python
OSCAD_HOME=set_PATH_to_OSCAD
/home/lavitha/OSCAD/setPath.py
#!/usr/bin/python
OSCAD_HOME=set_PATH_to_OSCAD
Installation completed
lavitha@fossee:~/Downloads/OSCAD_installer$
```

Figure 2.11: Terminal status when Oscad installation is completed

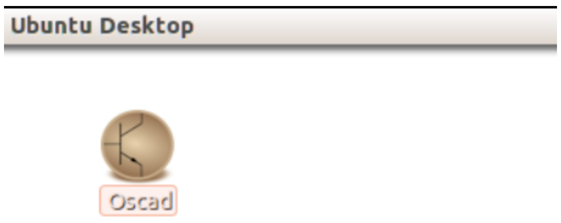
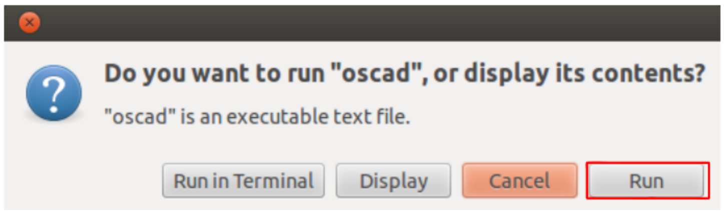


Figure 2.12: Oscad shortcut on the Desktop

RC filter circuit seen on schematic editor window is as shown in Fig. 2.17b.

- Close Schematic Editor.
- Click on *Ngspice* from Oscad Toolbar as shown in Fig. 2.16c. This will simulate the netlist using Ngspice.
- The graph and terminal window appears as shown in Fig. 2.18. This shows the result of transient analysis of RC circuit and verifies installation.[1]

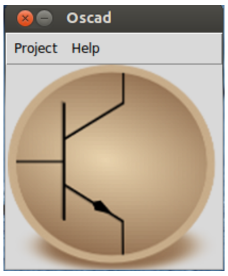
[1]Note: Fig. 2.18 has Ngspice graph with white background. One would normally get Ngspice graphs with black background.



Do you want to run "oscad", or display its contents?

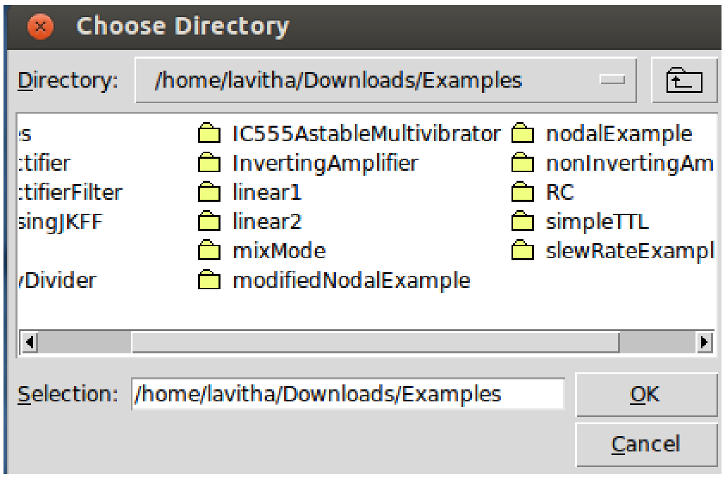
"oscad" is an executable text file.

(a) Choosing the option, *run* to launch Oscad



(b) Oscad window

Figure 2.13: Final stages in launching Oscad

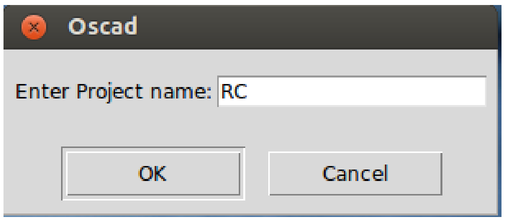


Directory: /home/lavitha/Downloads/Examples

```
s                IC555AstableMultivibrator   nodalExample
tifier           InvertingAmplifier          nonInvertingAm
tifierFilter     linear1                     RC
singJKFF         linear2                     simpleTTL
                 mixMode                     slewRateExampl
Divider          modifiedNodalExample
```

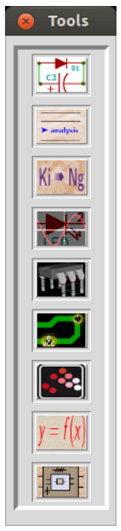
Selection: /home/lavitha/Downloads/Examples

Figure 2.14: Browsing to the folder where Oscad examples are saved

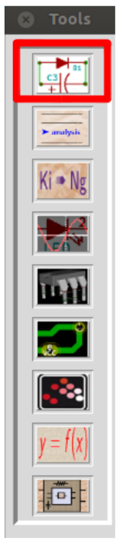


Enter Project name: RC

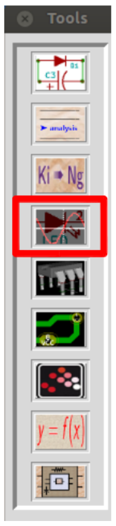
Figure 2.15: Dialog box to enter the project name



(a) Default appearance

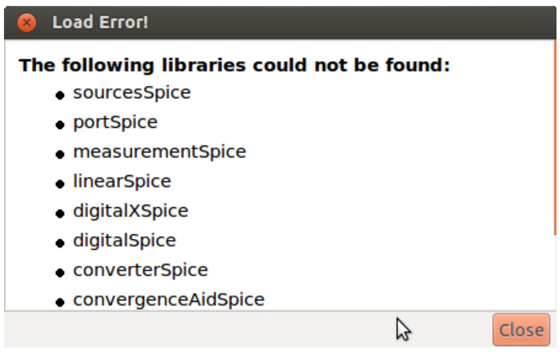


(b) Schematic editor selected



(c) Ngspice selected

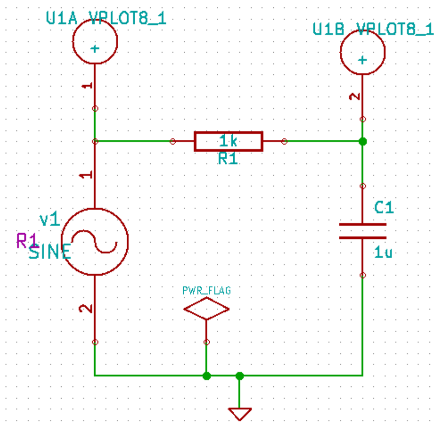
Figure 2.16: Oscad toolbar


Load Error!

The following libraries could not be found:

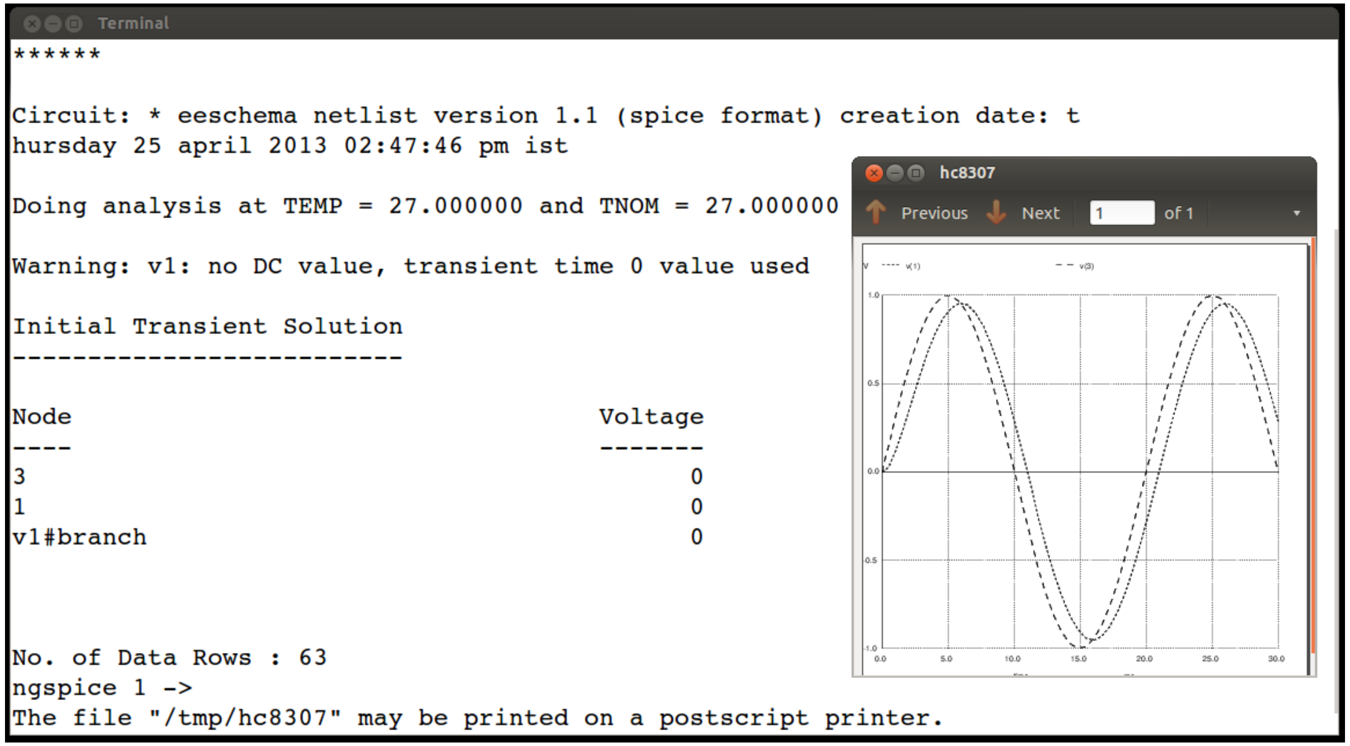
- sourcesSpice
- portSpice
- measurementSpice
- linearSpice
- digitalXSpice
- digitalSpice
- converterSpice
- convergenceAidSpice

(a) Schematic editor Load error



(b) Schematic of RC circuit

Figure 2.17: Stages in opening the schematic of RC circuit



```
******

Circuit: * eeschema netlist version 1.1 (spice format) creation date: t
hursday 25 april 2013 02:47:46 pm ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.000000

Warning: v1: no DC value, transient time 0 value used

Initial Transient Solution
--------------------------

Node                                   Voltage
----                                   -------
3                                            0
1                                            0
v1#branch                                    0



No. of Data Rows : 63
ngspice 1 ->
The file "/tmp/hc8307" may be printed on a postscript printer.
```

Figure 2.18: Transient analysis of RC circuit using Ngspice. Plots of v(3) and v(1) are shown, see Footnote 1 on Page 9.

Chapter 3

Architecture of Oscad

Oscad is a CAD tool that helps electronic system designers to design, test and analyse their circuits. But the important feature of this tool is that it is open source and hence the user can modify the source as per his/her need. The software provides a generic, modular and extensible platform for experiment with electronic circuits. This software runs on all Ubuntu Linux distributions. It uses `Python`, `KiCad`, `Ngspice` and `Scilab` (5.4.0 or above).

The objective behind the development of Oscad is to provide an open source EDA solution for electronics and electrical engineers. The software should be capable of performing schematic creation, PCB design and circuit simulation (analog, digital and mixed 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. The architecture of Oscad has been designed by keeping these objectives in mind.

3.1 Modules used in Oscad

Various open-source tools have been used for the underlying build-up of Oscad. In this section we will give a brief idea about all the modules used in Oscad.

3.1.1 EEschema

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. It is the schematic editor tool used in KiCad [11]. EEschema is intended to work with PCB layout software such as Pcbnew. It provides netlist that describes the electrical connections of the PCB. EEschema also integrates a component editor which allows the creation, editing and visualisation of components. It also allows the user to effectively handle 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: 1. Design rules check (`DRC`) for the automatic control of incorrect connections and inputs of components left unconnected. 2. Generation of layout files in `POSTSCRIPT` or `HPGL` format. 3. Generation of layout files printable via printer. 4. Bill of material generation. 5. Netlist generation for PCB layout or for simulation. This module is indicated by the label 1 in Fig. 3.1.

As Eeschema is originally intended for PCB Design, there are no fictitious components[2] 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 and square wave. We have also built a library which gives printing and plotting solutions. This extension, developed by us for Oscad, is indicated by the label 2 in Fig. 3.1.

3.1.2 CvPcb

CvPcb is a tool that allows the user to associate components in the schematic to component footprints when designing the printed circuit board. CvPcb is the footprint editor tool in KiCad [11]. Typically the netlist file generated by EEschema does not specify which printed circuit board footprint is associated with each component in the schematic. However, 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 that the correct footprint is associated with 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 its footprint. This interactive approach is simpler and less error prone than directly associating footprints in the schematic editor. This is because CvPcb not only allows automatic association, but also allows to see the list of available footprints and displays them on the screen to ensure the correct footprint is being associated. This module is indicated by the label 3 in Fig. 3.1.

3.1.3 Pcbnew

Pcbnew is a powerful printed circuit board software tool. It is the layout editor tool used in KiCad [11]. It is used in association with the schematic capture software EEschema, which provides the netlist. Netlist describes the electrical connections of the circuit. CvPcb is used to assign each component, in the netlist produced by EEschema, to a module that is used by Pcbnew. The features of Pcbnew are given below:

[2]Signal generator or power supply is not a single component but in circuit simulation, we consider them as a component. While working with actual circuit, signal generator or power supply gives input to the circuit externally thus, doesn't require for PCB design.

- 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 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 schematic.
- This tool provides a rats nest display, a hairline connecting the pads of modules 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 automatically generates a copper plane, with or without thermal breaks on the pads.
- It has a simple but effective auto router to assist in the production of the circuit. An export/import in `SPECCTRA` dsn format allows to use more advanced autorouters.
- It 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 such as:
  - display in full or outline.
  - display the track/pad clearance.

This module is indicated by the label 4 in Fig. 3.1.

3.1.4 Model Builder

This tool provides the facility to define a new model for devices such as, 1. Diode 2. Bipolar Junction Transistor (BJT) 3. Metal Oxide Semiconductor Field Effect Transistor (MOSFET) 4. Junction Field Effect Transistor (JFET) 5. IGBT and 6. Magnetic core. This module also helps edit existing models. It is developed by us for Oscad and it is indicated by the label 5 in Fig. 3.1. The use of this module is explained in detail in Sec. 8.1.

3.1.5 Subcircuit Builder

This module allows the user to create a subcircuit for a component. Once the subcircuit for a component is created, the user can use it in other circuits. It has the facility to define new components such as, Op-amps and IC-555. This component also helps edit existing subcircuits. This module is developed by us for Oscad and it is indicated by the label 6 in Fig. 3.1. The use of this module is explained in detail in Sec. 8.2.

3.1.6 KiCad to Ngspice netlist converter

It converts KiCad generated netlists to Ngspice (see Sec. 3.1.8) compatible format. It has the capability to 1. Insert parameters for fictitious components 2. Convert IC into discrete blocks 3. Insert D-A and A-D converter at appropriate places 4. Insert plotting and printing statements in netlist and 5. Find current through all components.

This module is developed by us for Oscad and it is indicated by the label 7 in Fig. 3.1. The use of this module is explained in detail in Sec. 6.2.

3.1.7 Analysis Inserter

This feature helps the user to perform different types of analysis such as Operating point analysis, DC analysis, AC analysis and transient analysis. It has the facility to 1. insert type of analysis and 2. insert options for analysis. This module is developed by us for Oscad and it is indicated by the label 8 in Fig. 3.1. The use of this module is explained in Sec. 6.1.

3.1.8 Ngspice

Ngspice is a general purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analyses [12]. 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 MOSFET. This module is indicated by the label 9 in Fig. 3.1.

3.1.9 Scilab based Mini Circuit Simulator (SMCSim)

We felt that an explanation capability would help students appreciate simulation results. This would also help students improve their circuits more easily. To this end, we have developed this module for Oscad, indicated by the label 10 in Fig. 3.1. SMCSim automatically generates the system of equations for the circuit under study and solves them using the state of the art open source simulator Scilab [3, 13]. SMCSim works in three modes: normal, symbolic and numerical. SMCSim is explained in more detail in

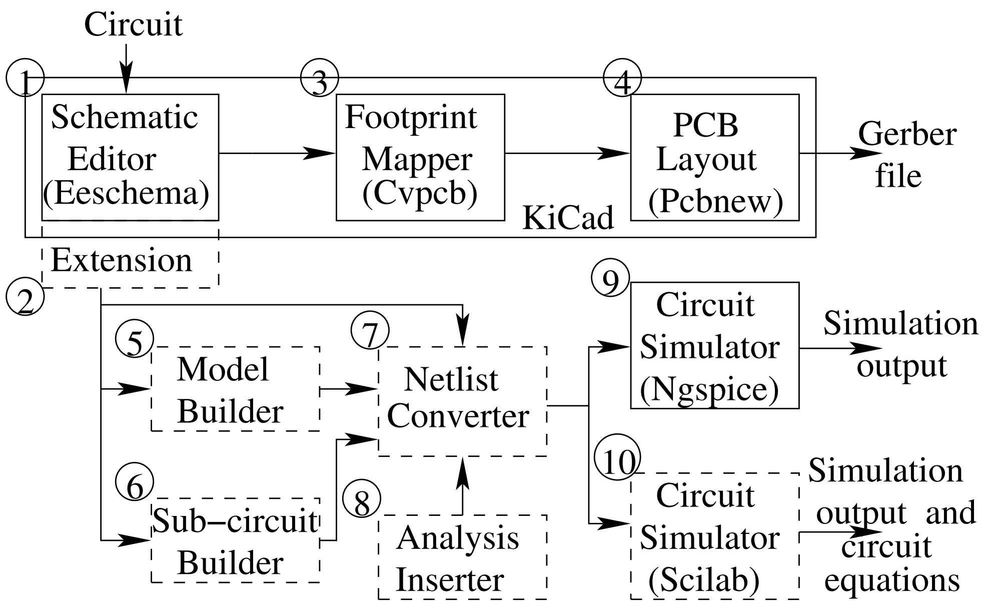


Figure 3.1: Work flow in Oscad. Boxes with dotted lines denote the modules developed in this work.

Chapter 9. To the best of our knowledge, this facility is not available in any commercial simulator.

3.2 Work flow of Oscad

Fig. 3.1 shows the work flow in Oscad. The block diagram consists of mainly three parts:

- Schematic Editor
- PCB Layout Editor
- Circuit Simulators

Here we explain the role of each block in designing electronic systems. Circuit design is the first step in the design of an electronic circuit. Generally a circuit diagram is drawn on a paper, and then entered into a computer using a schematic editor. EEschema is the schematic editor for Oscad. Thus all the functionalities of EEschema are naturally available in Oscad.

Libraries for components, explicitly or implicitly supported by Ngspice, have been created using the features of EEschema. As EEschema is originally intended for PCB design, there are no fictitious components such as voltage or current sources. Thus, a new library for different types of voltage and current sources such as sine, pulse and square wave, has been added in Oscad. A library which gives the functionality of printing and plotting has also been created.

The schematic editor provides a netlist file, which describes the electrical connections of the design. In order to create a PCB layout, physical components are required to be mapped into their footprints. To perform component to footprint mapping, CvPcb is used. Footprints have been created for the components in the newly created libraries. Pcbnew is used to draw a PCB layout.

After designing a circuit, it is essential to check the integrity of the circuit design. In the case of large electronic circuits, breadboard testing is impractical. In such cases, 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 the facility to define a new model for devices and edit existing models. Complex circuit elements can be created by hierarchical modeling. Subcircuit Builder provides an easy way to create a subcircuit.

The netlist generated by Schematic Editor cannot be directly used for simulation due to compatibility issues. Netlist Converter converts it into Ngspice compatible format. The type of simulation to be performed and the corresponding options are provided through a graphical user interface (GUI). This is called Analysis Inserter in Oscad.

Oscad uses Ngspice for analog, digital, mixed-level/mixed-signal circuit simulation. Ngspice is based on three open source software packages [14]:

- 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, and PTM models. It is widely used due to its accuracy even for the latest technology devices.

Oscad also has a Scilab based circuit simulator. It generates equations from the netlist and gets them solved by Scilab, which has many built in state of the art numerical methods. This tool is called Scilab based Mini Circuit Simulator (SMCSim) in Oscad.

Chapter 4

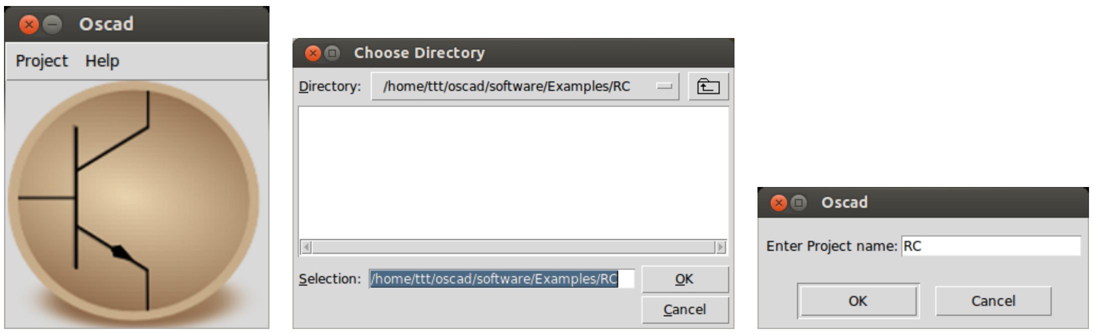
Getting Started

In this chapter we will get started with Oscad. We will run through the various options available with an example circuit. Referring to this chapter will make one familiar with Oscad and will help plan the project before actually designing a circuit. Lets get started.

After one finishes installing Oscad, a shortcut link for Oscad will be created on the desktop. Double click on this link to launch Oscad. Oscad main window will open up. It is shown in Fig. 4.1a. On the menu bar there are two menus, *Project* and *Help*. To create a new project or open an existing project, use the *Project* menu.

Let us create a new project. Click on *Project* and select *New*. The `Choose Directory` window opens up. This window is shown in Fig. 4.1b. Browse to the desired directory. Click on OK. A new window opens up with `Enter Project name` field. Type the name of the new project here. Click on OK. A folder will be created in the specified directory. The name of this folder will be the same as that of the project created.

A vertical toolbar will appear. This toolbar is shown in Fig. 2.16a on Page 11. This is the `Oscad toolbar`.



Directory: /home/ttt/oscad/software/Examples/RC

Selection: /home/ttt/oscad/software/Examples/RC

Enter Project name: RC

(a) Oscad main window (b) Open project directory (c) Enter Project Name window

Figure 4.1: Getting started with Oscad

Now let us see how to open an existing project. Click on *Project* and select *Open*. The `Choose Directory` window, as shown in Fig. 4.1b, opens up. Browse to the directory of the project. Choose `RC` from the Examples folder that has been downloaded from the Oscad web page (`www.oscad.in`). Click on OK. The `Enter Project name` window opens up. This window is illustrated in Fig. 4.1c. Since we are opening an already existing project, the project name will appear in the text box automatically. Click on OK.

The Oscad toolbar appears. It contains 9 tools. The tool icons have images depicting their purpose. If one places the mouse pointer on these tools, the name of the tool appears at the bottom of the mouse pointer. Following is the list of tools, from top to bottom as they appear, in the Oscad toolbar.

1. Schematic Editor 2. Analysis Inserter 3. Netlist Converter 4. Ngspice 5. Footprint Editor 6. Layout Editor 7. SMCSim 8. Model Builder 9. Subcircuit Builder.

4.1 Schematic Editor

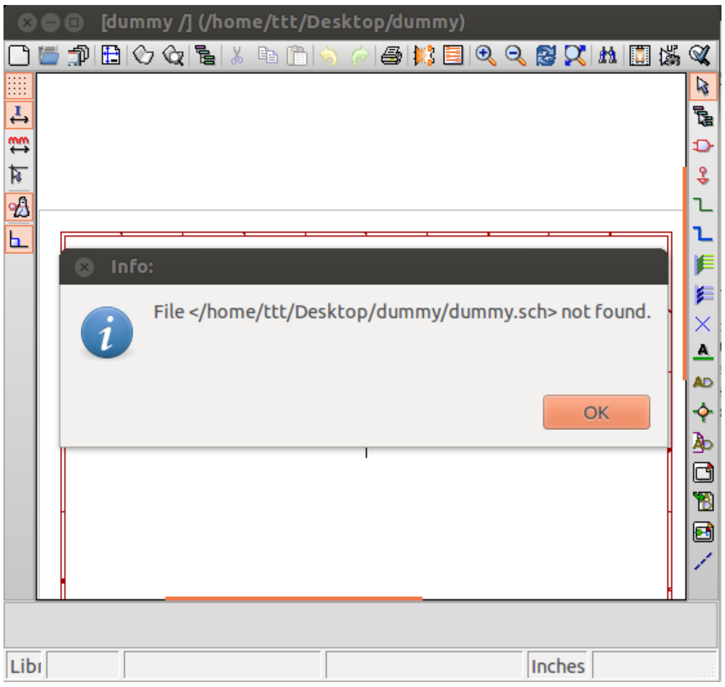
Click on the first tool on the toolbar i.e. *Schematic Editor*. Doing so will open EEschema, the schematic editor used in Oscad. If a new project is being created, one will get the schematic editor window with an info dialog box. This is illustrated in Fig. 4.2. This warning can be safely ignored by clicking on `OK`.

However, if an already existing project is opened, one would get the schematic editor window along with a Load error. This is illustrated in Fig. 4.3. This error occurs because the schematic that is opened has not been loaded with the libraries mentioned in the Load Error message. Sec. 5.2.2 explains how to load these libraries to the project. Close the Load Error message by clicking on the `Close` button. The RC circuit diagram opens up as shown in Fig. 4.4. Now the circuit schematic can be created/edited. To know how to use the schematic editor to create circuit schematics, refer to Chapter 5.

4.2 Analysis Inserter

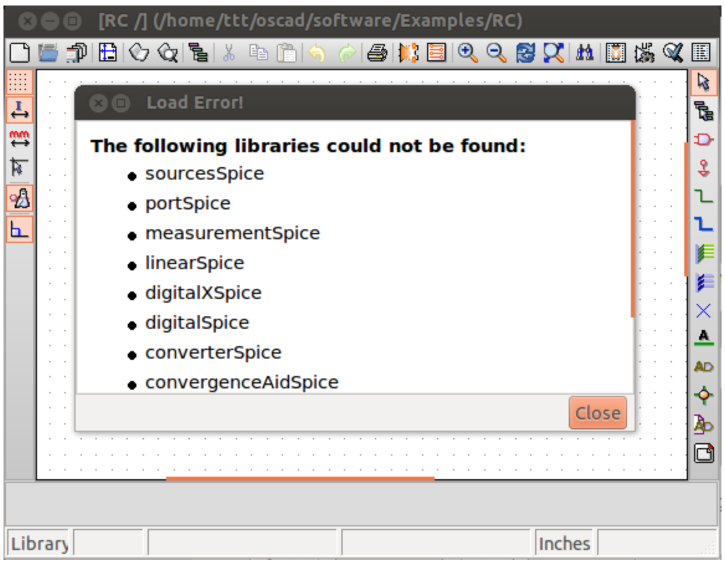
The second tool on the toolbar is the *Analysis Inserter*. Note that the spice netlist file (.cir) should be generated before using this tool. This is because this tool is used to insert analysis commands to the spice netlist file. To know how to generate the spice netlist file, refer to Sec. 5.3.5.

When one clicks on this tool, a window titled `KiCad Ngspice` will open. This is the Analysis Inserter GUI. It is shown in Fig. 4.5. This window helps the user to insert the analysis commands into the spice netlist file. The use of analysis inserter is explained in detail in Sec. 6.1.



File </home/ttt/Desktop/dummy/dummy.sch> not found.

Figure 4.2: Schematic Editor (opening a new project)



The following libraries could not be found:

- sourcesSpice
- portSpice
- measurementSpice
- linearSpice
- digitalXSpice
- digitalSpice
- converterSpice
- convergenceAidSpice

Figure 4.3: Schematic Editor (opening an already existing project)

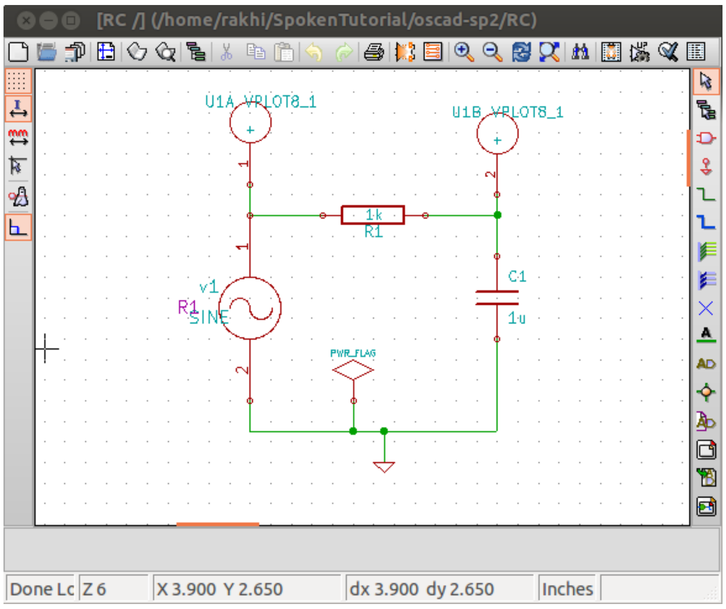
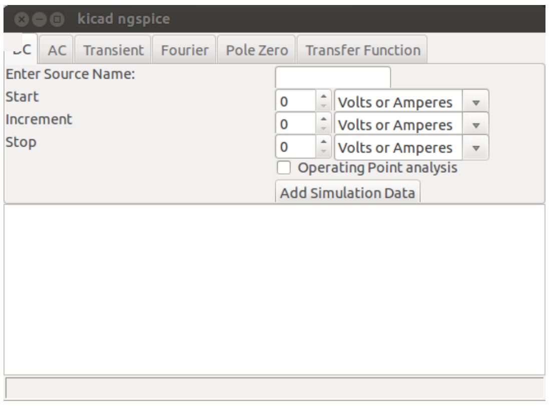


Figure 4.4: Schematic Editor (with RC circuit)



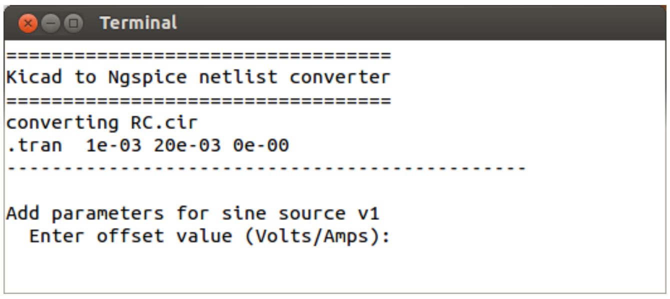
Enter Source Name:
Start 0 Volts or Amperes
Increment 0 Volts or Amperes
Stop 0 Volts or Amperes
Operating Point analysis

Figure 4.5: Analysis Inserter

4.3 Netlist Converter

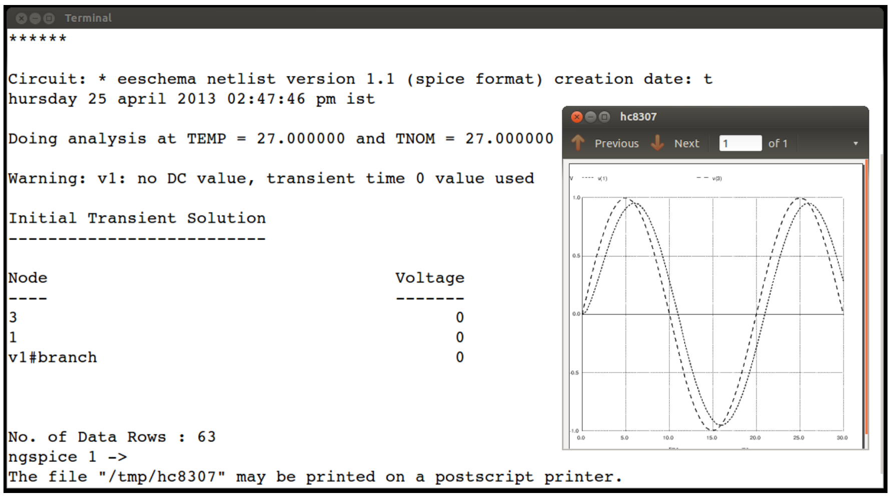
The third tool on the toolbar is the *Netlist Converter*. Before one uses this tool, one should have already created the spice netlist file and used analysis inserter to generate analysis commands. To know how to generate spice netlist, refer to Sec. 5.3.5. To know how to insert analysis commands to the netlist, refer to Chapter 6. This file is not directly usable for simulation. In other words, it is not compatible with Ngspice.

The spice netlist file contains only the component placement information and it



```
==================================
Kicad to Ngspice netlist converter
==================================
converting RC.cir
.tran  1e-03 20e-03 0e-00
----------------------------------------------

Add parameters for sine source v1
  Enter offset value (Volts/Amps):
```

Figure 4.6: Netlist Converter



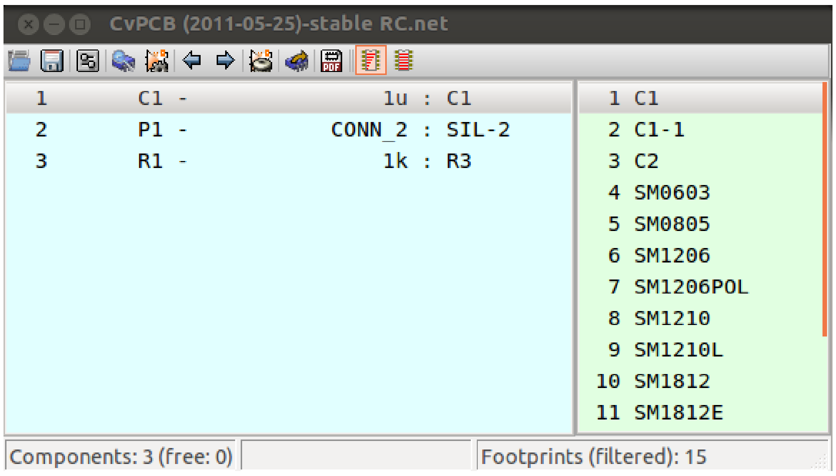
```
******

Circuit: * eeschema netlist version 1.1 (spice format) creation date: t
hursday 25 april 2013 02:47:46 pm ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.000000

Warning: v1: no DC value, transient time 0 value used

Initial Transient Solution
--------------------------

Node                                  Voltage
----                                  -------
3                                           0
1                                           0
v1#branch                                   0


No. of Data Rows : 63
ngspice 1 ->
The file "/tmp/hc8307" may be printed on a postscript printer.
```

Figure 4.7: Ngspice simulation result showing v(3) and v(1), see Footnote 1 on Page 9.

says nothing about the magnitude and other parameters (if applicable) of the source components like voltage source and current source. When one clicks on the *Netlist Converter* tool, a terminal window will open up as shown in Fig. 4.6. The terminal window displays the name of the .cir file it is referring to. It also asks for the source value. It may ask for other parameters depending upon the type of sources used. Once the values have been entered, press the `Enter` key. It will generate `.cir.out` and `.cir.ckt` files in the same project directory.

4.4 Ngspice

Sections 4.2 and 4.3 help generate a netlist suitable to be simulated using Ngspice. Clicking on the tool *Ngspice* will open a terminal window and a graph window as shown in Fig. 4.7. One should have the converted netlist file `.cir.out` before using this tool.


CvPCB (2011-05-25)-stable RC.net

```
1        C1 -             1u : C1
2        P1 -         CONN_2 : SIL-2
3        R1 -             1k : R3
```

```
 1 C1
 2 C1-1
 3 C2
 4 SM0603
 5 SM0805
 6 SM1206
 7 SM1206POL
 8 SM1210
 9 SM1210L
10 SM1812
11 SM1812E
```

Components: 3 (free: 0)

Footprints (filtered): 15

Figure 4.8: CvPcb window

4.5 Footprint Editor

Clicking on the *Footprint Editor* tool will open the `CvPcb` window. This window will ideally open the .net file for the current project. So, before using this tool, one should have the netlist for PCB design (a .net file). To know more about how to generate netlist for PCB, refer to Sec. 7.1.1.

Open the project `RC_pcb` available in the `Examples` folder downloaded from the Oscad website. On clicking the *Footprint Editor* tool, we see the corresponding RC_pcb.net file for RC circuit. This window is shown in Fig. 4.8. The main purpose of this window is to let one choose the footprints for the various components in the circuit. Let us view the footprint `C1` for capacitor C1. Click on `C1` from the right hand side of CvPcb window. Click on *View Selected Footprint* tool from the tool bar of CvPcb window. This will show the footprint corresponding to C1. This is illustrated in Fig. 4.9. To know more about how to assign footprints to components, see Chapter 7.

4.6 Layout Editor

Open the RC_pcb project available in `Examples`. Clicking on the *Layout Editor* tool will open `Pcbnew`, the layout editor used in Oscad. This shows the PCB design for RC circuit. In this window, one will create the PCB. It involves laying tracks and vias, performing optimum routing of tracks, creating one or more copper layers for PCB, etc. The PCB design for RC circuit is shown in Fig. 4.10. This is how the PCB will look like when one actually prints it on a copper-clad board. It will be saved as a `.brd` file in the same directory. Chapter 7 explains how to use the *Layout Editor* to design a PCB.

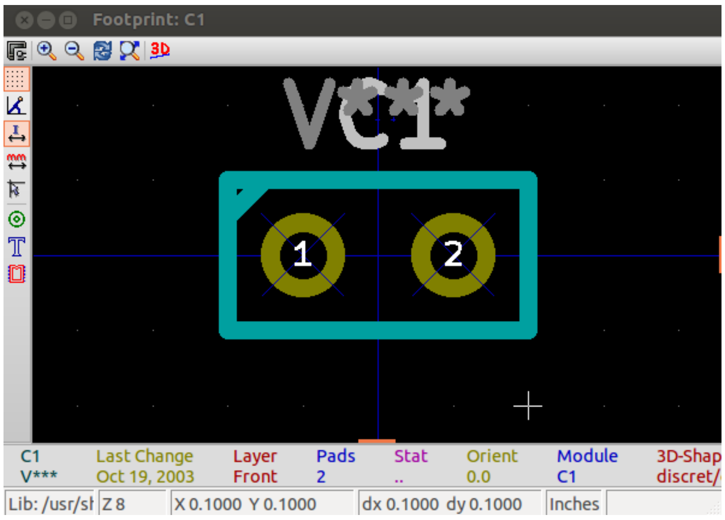


Figure 4.9: Footprint for C1

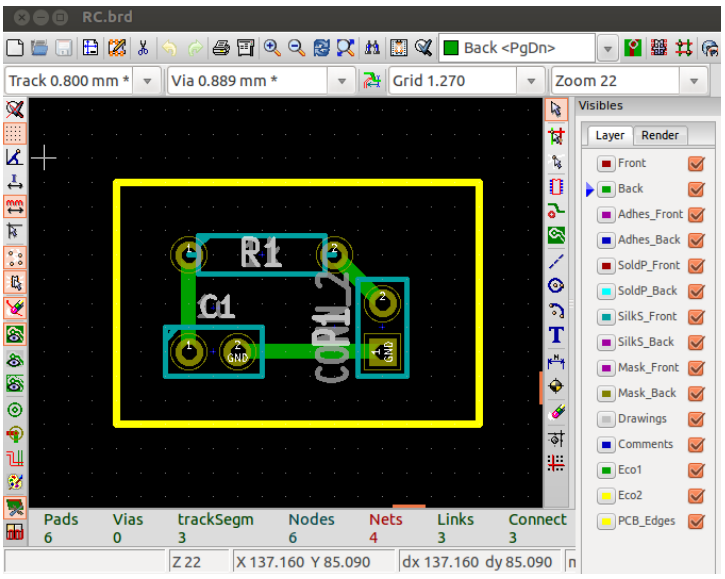
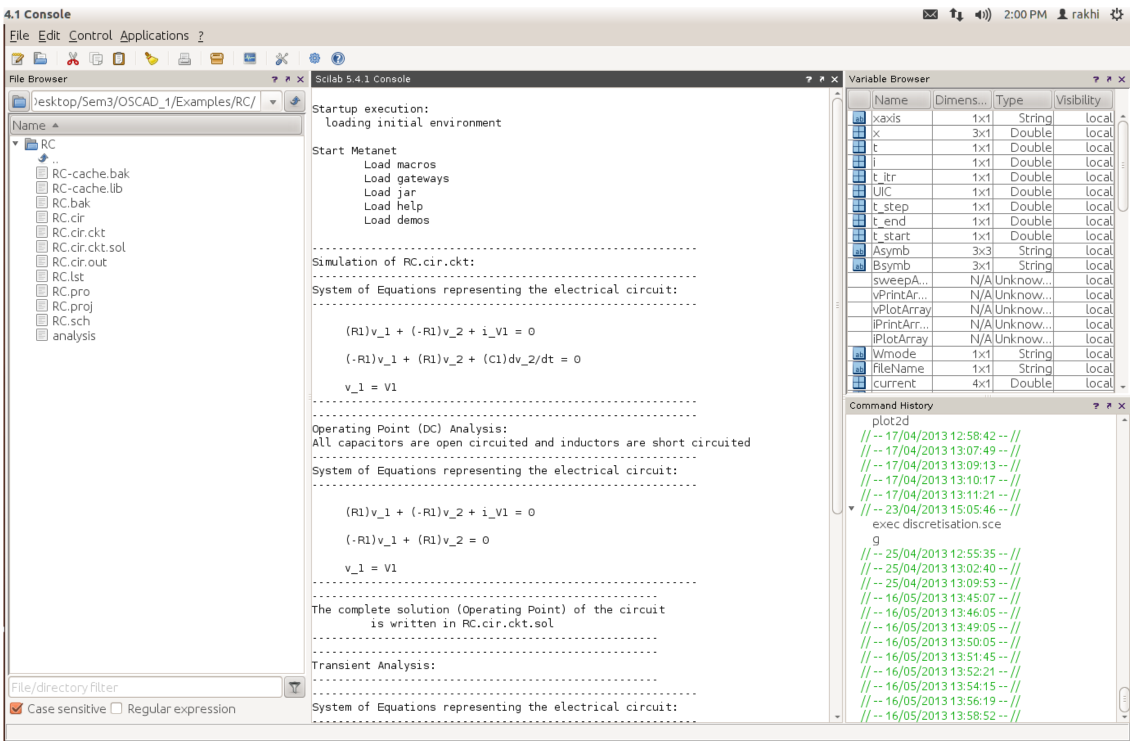


Figure 4.10: PCB design for RC circuit

4.7 SMCSim

SMCSim stands for Scilab based Mini Circuit Simulator. This tool generates mathematical equations for the circuit, thereby giving a better understanding of the circuit. Oscad uses Scilab [3] for this purpose. Clicking on this tool will open a small window



```
)esktop/Sem3/OSCAD_1/Examples/RC/
Name
RC
  ..
  RC-cache.bak
  RC-cache.lib
  RC.bak
  RC.cir
  RC.cir.ckt
  RC.cir.ckt.sol
  RC.cir.out
  RC.lst
  RC.pro
  RC.proj
  RC.sch
  analysis
```

```
Startup execution:
  loading initial environment

Start Metanet
        Load macros
        Load gateways
        Load jar
        Load help
        Load demos

-----------------------------------------------------------
Simulation of RC.cir.ckt:
-----------------------------------------------------------
System of Equations representing the electrical circuit:
-----------------------------------------------------------


     (R1)v_1 + (-R1)v_2 + i_V1 = 0

     (-R1)v_1 + (R1)v_2 + (C1)dv_2/dt = 0

     v_1 = V1
-----------------------------------------------------------
-----------------------------------------------------------
Operating Point (DC) Analysis:
All capacitors are open circuited and inductors are short circuited
-----------------------------------------------------------
System of Equations representing the electrical circuit:
-----------------------------------------------------------


     (R1)v_1 + (-R1)v_2 + i_V1 = 0

     (-R1)v_1 + (R1)v_2 = 0

     v_1 = V1
-----------------------------------------------------------
-----------------------------------------------------
The complete solution (Operating Point) of the circuit
         is written in RC.cir.ckt.sol
-----------------------------------------------------
-----------------------------------------------------
Transient Analysis:
-----------------------------------------------------
-----------------------------------------------------------
System of Equations representing the electrical circuit:
-----------------------------------------------------------
```

	Name	Dimens...	Type	Visibility
ab	xaxis	1x1	String	local
▦	x	3x1	Double	local
▦	t	1x1	Double	local
▦	i	1x1	Double	local
▦	t_itr	1x1	Double	local
▦	UIC	1x1	Double	local
▦	t_step	1x1	Double	local
▦	t_end	1x1	Double	local
▦	t_start	1x1	Double	local
ab	Asymb	3x3	String	local
ab	Bsymb	3x1	String	local
	sweepA...	N/A	Unknow...	local
	vPrintAr...	N/A	Unknow...	local
	vPlotArray	N/A	Unknow...	local
	iPrintArr...	N/A	Unknow...	local
	iPlotArray	N/A	Unknow...	local
ab	Wmode	1x1	String	local
ab	fileName	1x1	String	local
▦	current	4x1	Double	local

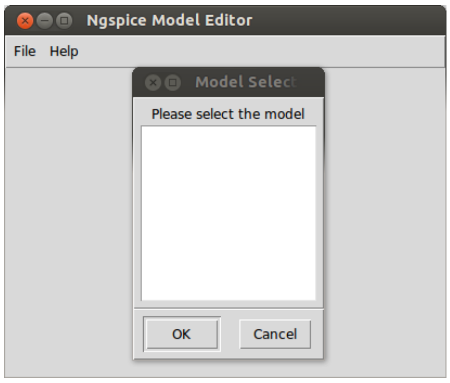
```
    plot2d
// -- 17/04/2013 12:58:42 -- //
// -- 17/04/2013 13:07:49 -- //
// -- 17/04/2013 13:09:13 -- //
// -- 17/04/2013 13:10:17 -- //
// -- 17/04/2013 13:11:21 -- //
// -- 23/04/2013 15:05:46 -- //
    exec discretisation.sce
    g
// -- 25/04/2013 12:55:35 -- //
// -- 25/04/2013 13:02:40 -- //
// -- 25/04/2013 13:09:53 -- //
// -- 16/05/2013 13:45:07 -- //
// -- 16/05/2013 13:46:05 -- //
// -- 16/05/2013 13:49:05 -- //
// -- 16/05/2013 13:50:05 -- //
// -- 16/05/2013 13:51:45 -- //
// -- 16/05/2013 13:52:21 -- //
// -- 16/05/2013 13:54:15 -- //
// -- 16/05/2013 13:56:19 -- //
// -- 16/05/2013 13:58:52 -- //
```

Figure 4.11: System of equations representing the electrical circuit in Scilab

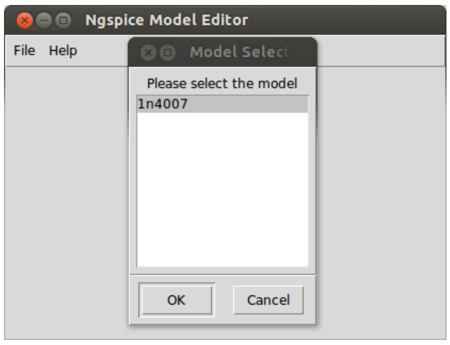
where one has to choose between three options: 1. Normal 2. Symbolic and 3. Matrix. The last option has been referred to as the numerical mode in this book. These are the modes used for circuit simulation. After one of them is selected and ok button is clicked, Scilab will be launched automatically. Let us choose the option `Symbolic`. The scilab console will show the set of equations for the circuit as shown in Fig. 4.11. A plot window opens up showing the plot of variables in the circuit. The placement of `plot` components in the circuit schematic determines which variables to be plotted. To know more about how to use this feature, refer to Chapter 9.

4.8 Model Builder

Oscad also gives an option to re-configure the model of a component. It facilitates the user to change models of components such as diode, transistor, MOSFET, etc. When one clicks on the *Model Builder* tool, the window as shown in Fig. 4.12a will appear. One sees a blank window because the RC circuit which has been opened does not have any component whose model can be edited. Let us open from the `Examples` folder, `bridgeRectifier` project which has a component with an editable model. Click on

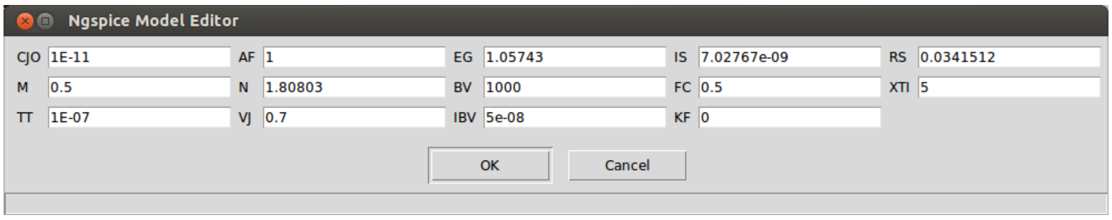


(a) Default appearance



(b) Model Builder window of circuit having a diode

Figure 4.12: Model Builder window



CJO	1E-11	AF	1	EG	1.05743	IS	7.02767e-09	RS	0.0341512
M	0.5	N	1.80803	BV	1000	FC	0.5	XTI	5
TT	1E-07	VJ	0.7	IBV	5e-08	KF	0		

Figure 4.13: Model Builder window showing model of diode

the *Model Builder* tool from the Oscad toolbar. The window shown in Fig. 4.12b will appear. We can see that it shows `1n4007` in the window. After choosing `1n4007` and clicking on *OK*, it will respond saying `Do you want to edit?` After clicking on *OK*, a window will open. This window will contain the various parameters governing the model of diode, for example reverse breakdown voltage (BV) and ohmic resistance (RS). This is illustrated in Fig. 4.13. To know more about how to use this feature, refer to Chapter 8.

4.9 Subcircuit Builder

Oscad has an option to build subcircuits. The subcircuits can again have components having subcircuits and so on. This enables users to build commonly used circuits as subcircuits and then use it across circuits. For example, one can build a 12 Volt power supply as a subcircuit and then use it as just a single component across circuits without having the need to recreate it. Clicking on *Subcircuit Builder* tool will allow one to edit or create a subcircuit. To know how to make a subcircuit, refer to Chapter 8. Fig. 4.14 shows the subcircuit of 555 timer IC.

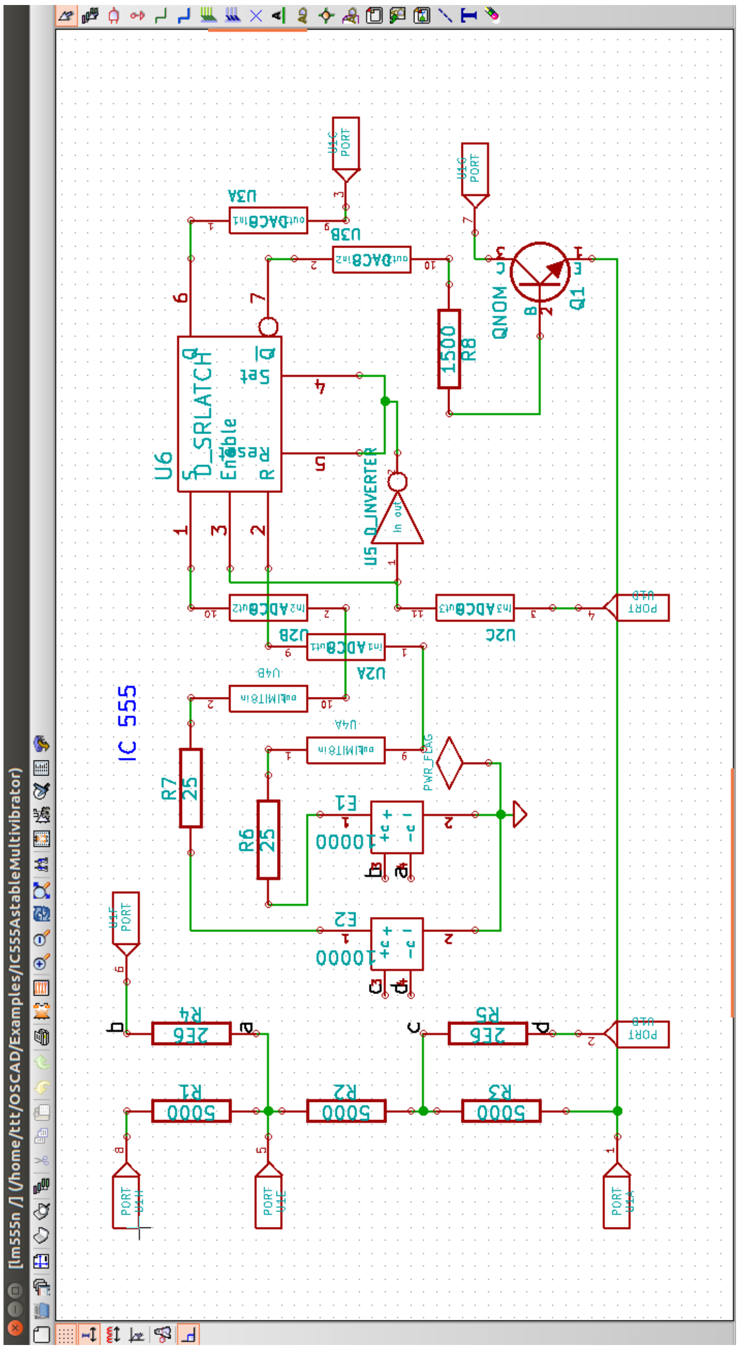


Figure 4.14: Subcircuit of 555 timer IC

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

Fig. 5.1 shows the schematic editor and the various menu and toolbars. We will explain them briefly in this section.

5.1.1 Top menu bar

The top menu bar will be available at the top left corner. 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.
    (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 Fig. 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 has shortcuts for placing various items like components, wire and junction, on to the schematic editor window. See Sec. 5.1.5 to know

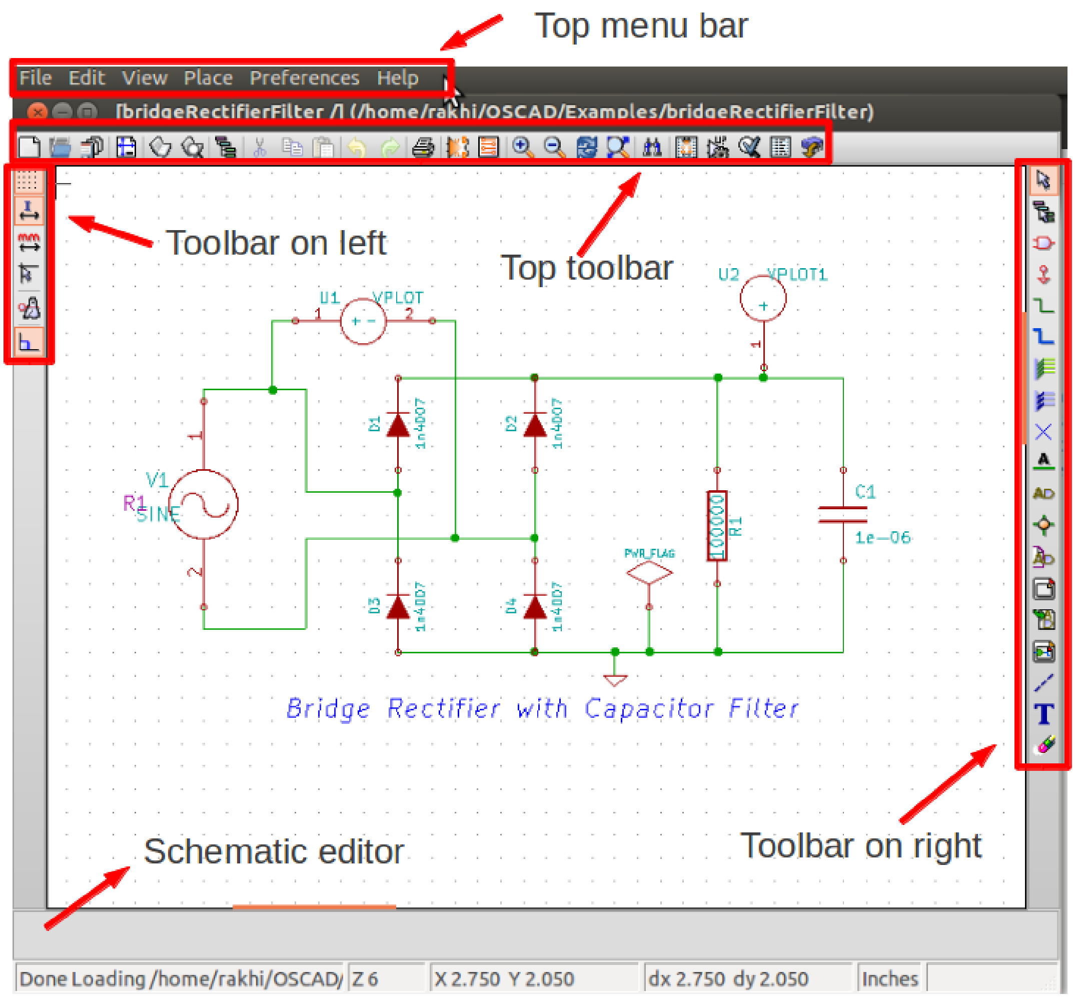
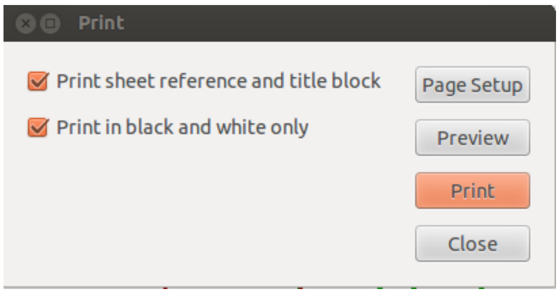


Figure 5.1: Schematic editor with the menu bar and toolbars marked

more about various shortcut 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 Sec. 5.1.5 about hotkeys.
   (f) Read preferences - Read configuration file.



Print sheet reference and title block [✓]

Print in black and white only [✓]

Figure 5.2: Print options

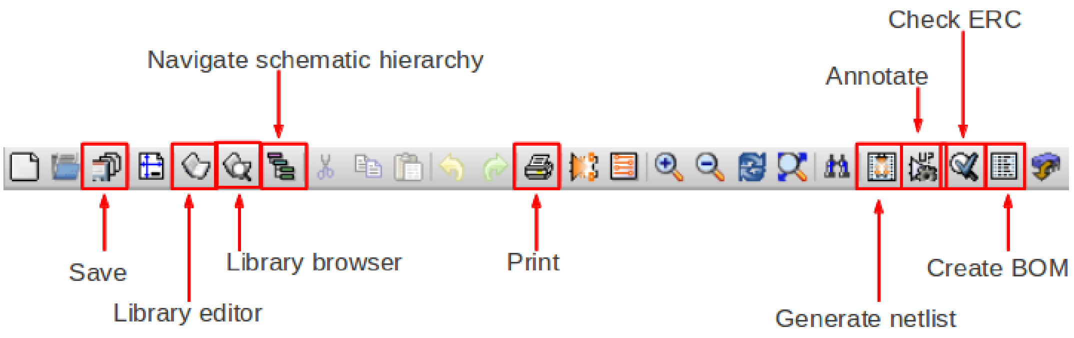


Figure 5.3: Toolbar on top with important tools marked

(g) Save preferences - Save configuration file.

5.1.2 Top toolbar

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

1. Save - Save the current schematic
2. Library Editor - Create or edit components. See Sec. 5.2 for more details.
3. Library Browser - Browse through the various component libraries available
4. Navigate schematic hierarchy - Navigate among the root and sub-sheets in the hierarchy
5. Print - Print the schematic
6. Generate netlist - Generate a netlist for PCB design or for simulation.
7. Annotate - Annotate the schematic
8. Check ERC - Do Electric Rules Check for the schematic
9. Create BOM - Create a Bill of Materials of the schematic

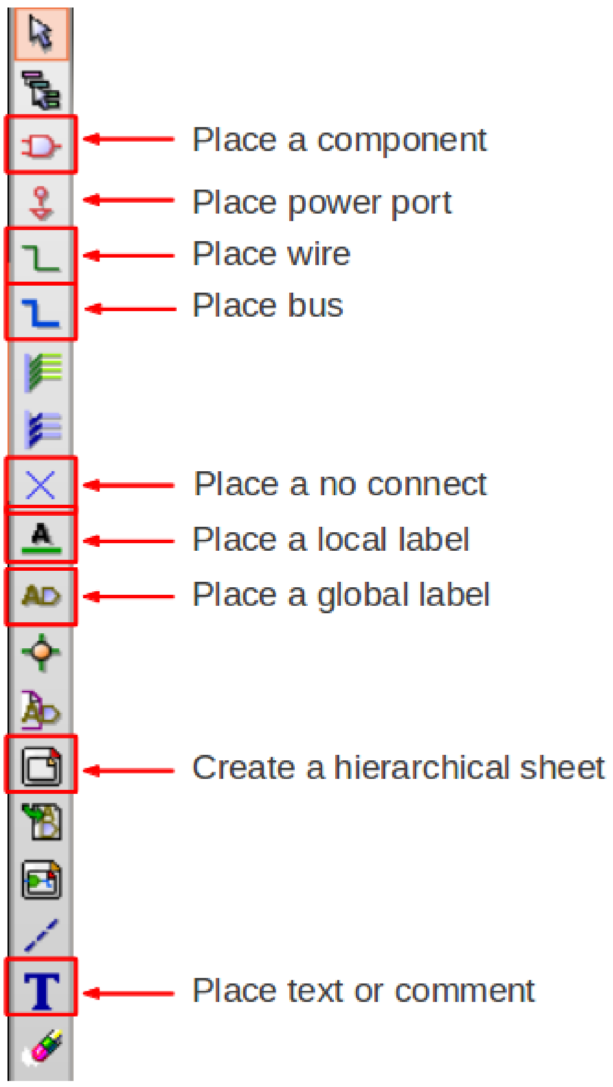


Figure 5.4: Toolbar on right with important tools marked

5.1.3 Toolbar on the right

The toolbar on the right side of the schematic editor window has many important tools. Some of them are marked in Fig. 5.4. Let us now look at each of these tools and their uses.

1. Place a component - Load a component to the schematic. See Sec. 5.3.1 for more details.

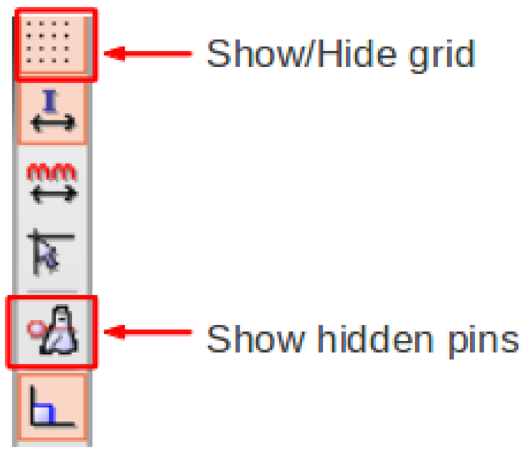


Figure 5.5: Toolbar on left with important tools marked

2. Place a power port - Load a power port (Vcc, ground) to the schematic
3. Place wire - Draw wires to connect components in schematic
4. Place bus - Place a bus on the 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 within 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 the 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 Fig. 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.

5.1.5 Hotkeys

A set of keyboard keys are associated with various operations in the schematic editor. These keys save time and make it easy to switch 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.* The hotkeys can also be edited by selecting the option *Edit Hotkeys.* Some frequently used hotkeys, along with their functions, are given 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: Both lower and upper-case keys will work as hotkeys.

5.2 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 and current sources). A set of component libraries has been created with such components. If one is using Oscad only for designing a PCB, then one might not need these libraries. However, these libraries are essential if one needs to simulate one's circuit. Hereafter, we will refer to these libraries as *Oscad libraries* to distinguish them from libraries already present in EEschema (EEschema libraries).

5.2.1 Oscad libraries

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

1. analogSpice - Discrete components like capacitor, resistor and BJT
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 one wants to simulate a circuit that has a 555 timer IC in it, one 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 one uses 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.

5.2.2 Adding Oscad component libraries to project

Let us see how to add the Oscad libraries to a project. Go to *Preferences* from the top menu bar. Choose *Library*. The window shown in Fig. 5.6 will be obtained. Click on *Add* (marked in red). Browse to the folder where Oscad is installed. Double click on the folder *OSCAD*. Go to the folder *Library*. Select all the `*.lib` files as shown in Fig. 5.7. Click on Open. Now click on OK on the window shown in Fig. 5.6.

Note: One should add these to one's project each time the schematic is created/edited.

5.2.3 Library browser

One 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 Fig. 5.8 to illustrate this.

Note: One can view the Oscad libraries in the library browser ONLY IF they have been added to the project as described in Sec. 5.2.2

5.2.4 Plot component library

Plot components are required to view the results of simulation. These are available in the Oscad library *measurementSpice* shown in Fig. 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.

5.2.5 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 Fig. 5.10. Another important component in this library is the Power Flag

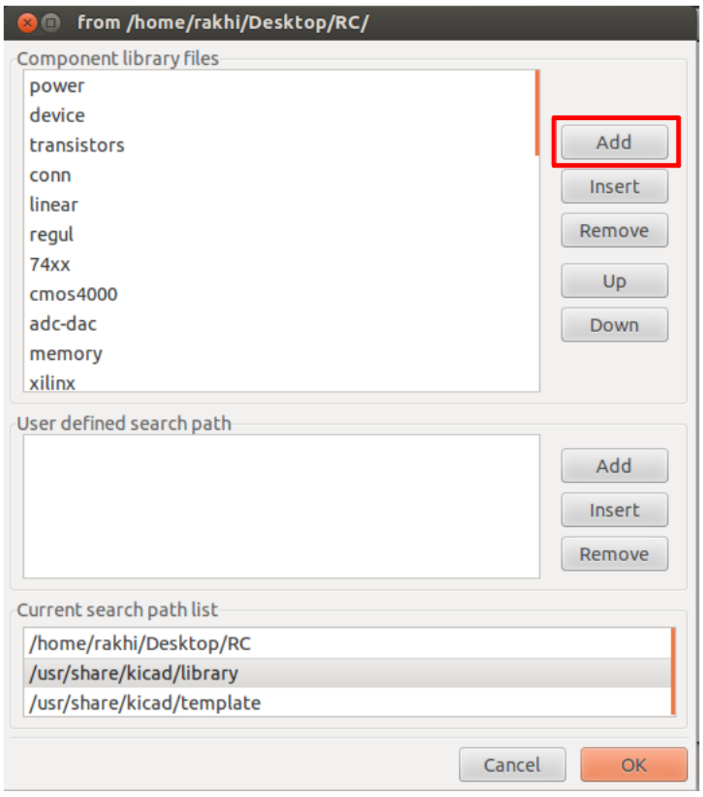


Figure 5.6: Adding component library

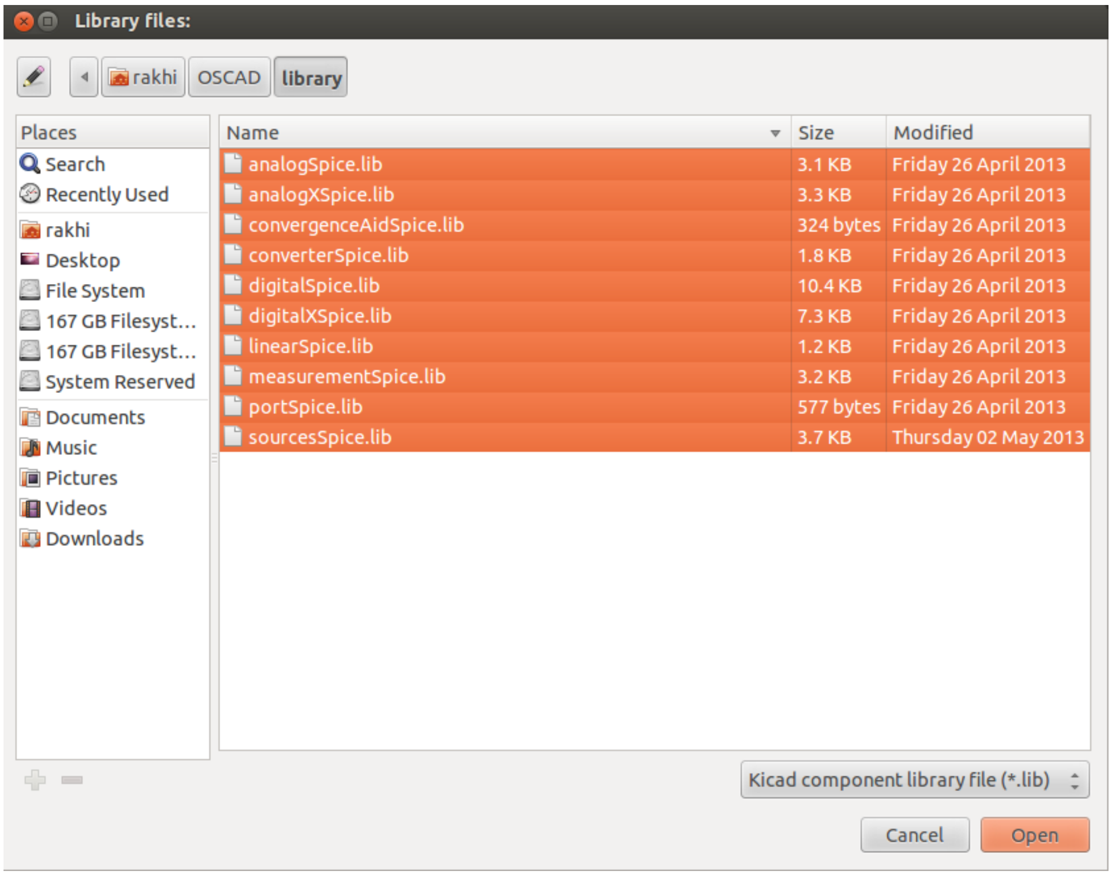
(*PWR_FLAG*). It is a dummy component placed in schematic to tell the schematic editor that the pin/node is driven by a power source and hence prevents ERC errors.

5.2.6 Connector library

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

5.2.7 Component references

Every component has a unique reference. For example, resistor has a reference `R`, BJTs have a reference `Q`, MOSFETs have a reference `M` and ICs have a reference `U`. If a component is to be made a subcircuit, then its reference should be ‘X’. When a component is placed in the schematic editor window, the reference will be shown with a question mark. This indicates that the component is not annotated. See Sec. 5.3.4 for more information about annotation.



Name	Size	Modified
analogSpice.lib	3.1 KB	Friday 26 April 2013
analogXSpice.lib	3.3 KB	Friday 26 April 2013
convergenceAidSpice.lib	324 bytes	Friday 26 April 2013
converterSpice.lib	1.8 KB	Friday 26 April 2013
digitalSpice.lib	10.4 KB	Friday 26 April 2013
digitalXSpice.lib	7.3 KB	Friday 26 April 2013
linearSpice.lib	1.2 KB	Friday 26 April 2013
measurementSpice.lib	3.2 KB	Friday 26 April 2013
portSpice.lib	577 bytes	Friday 26 April 2013
sourcesSpice.lib	3.7 KB	Thursday 02 May 2013

Kicad component library file (*.lib)

Figure 5.7: Selecting all the Oscad library (*.lib) files

5.3 Schematic creation for simulation

There are certain differences between the schematic created for simulation and that created for PCB design. We need certain components like plots and current sources. for simulation whereas these are not needed for PCB design. For PCB design, we would require connectors (e.g. DB15 and 2 pin connector) for taking signals in and out of the PCB whereas these have no meaning in simulation. This section covers schematic creation for simulation. Refer to Chapter 7 to know how to create schematic for PCB design.

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 Fig. 5.12 and do a transient simulation.

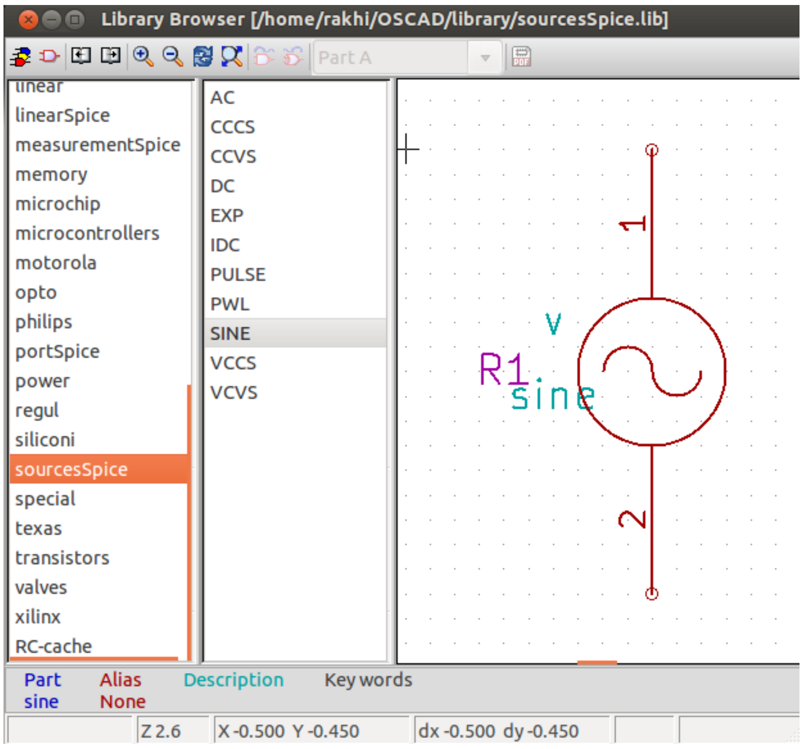


Figure 5.8: Library browser showing *sourceSpice* library

5.3.1 Selection and placement of components

We would need a resistor, a capacitor, a voltage source, ground terminal and some plot components. Add the Oscad libraries to the project as described in Sec. 5.2.2. To place a resistor on the schematic editor window, 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 Fig. 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. 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

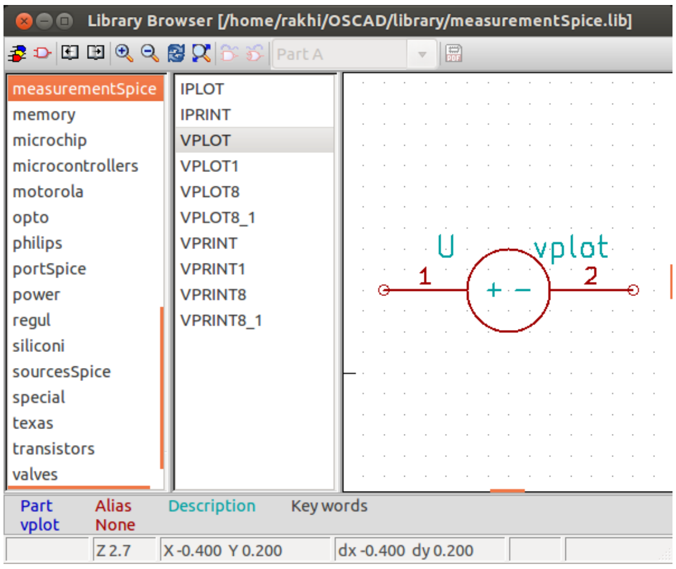


Figure 5.9: The Oscad *measurementSpice* library

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 look like the Fig. 5.14.

Let us rotate the resistor to complete the circuit as shown in Fig. 5.12. To rotate the resistor, place the cursor on the resistor and press the key `R`. Note that if the cursor is placed above the letter `R` (not `R?`) on the resistor, it asks to clarify selection. Choose the option *Component R*. This can be avoided by placing the cursor slightly away from the letter R as shown in Fig. 5.15. This applies to all components. If one wants 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.

5.3.2 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 to be connected 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 Fig. 5.16a. Similarly connect the wires between all terminals and the final schematic would look like Fig. 5.16b.

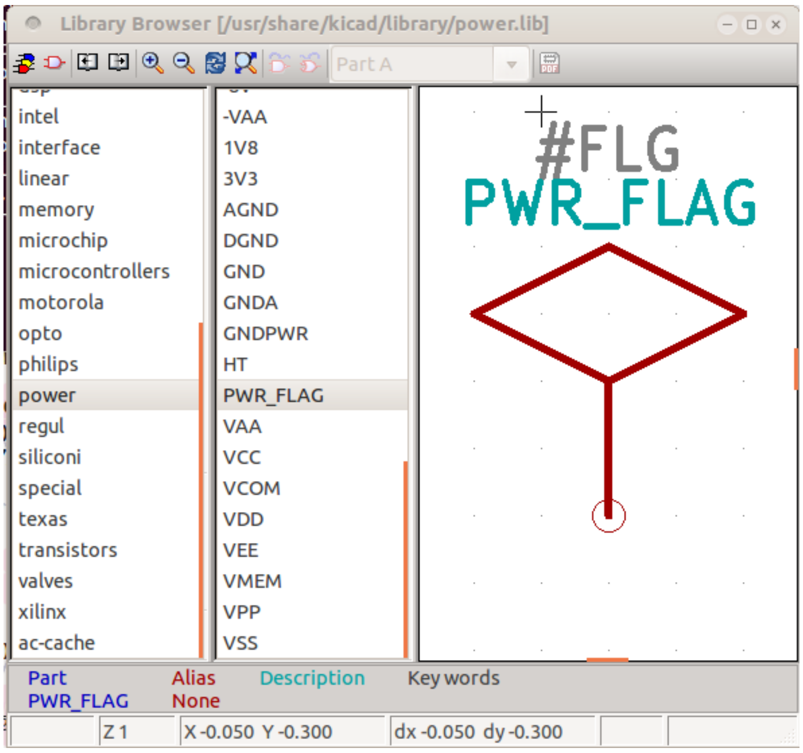


Figure 5.10: The EEschema *power* library

5.3.3 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 the 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 Fig. 5.17. 1k means $1k\Omega$. Similarly give the value `1u` for the capacitor. 1u means $1\mu F$.

5.3.4 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 annotation, then click on OK and finally click on close as shown in Fig. 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 now do `ERC` or `Electric Rules Check`. To do so, click on *Perform electric*

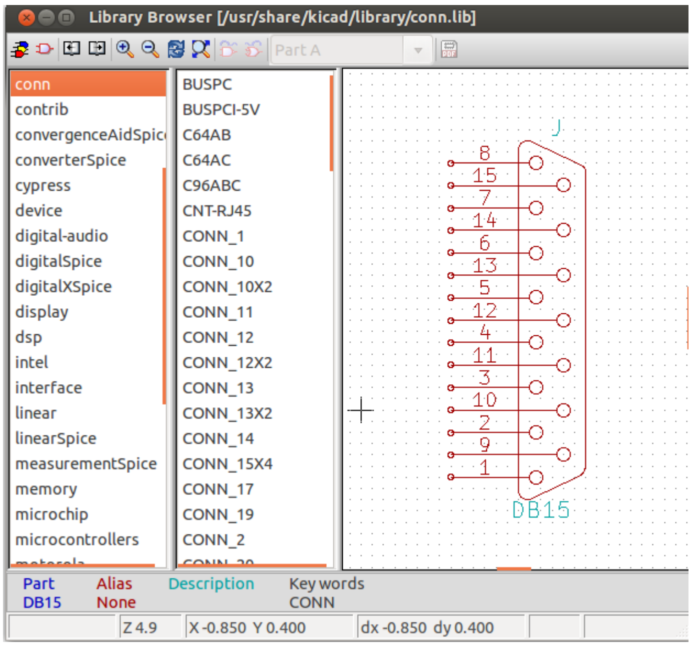


Figure 5.11: The EEschema *conn* library

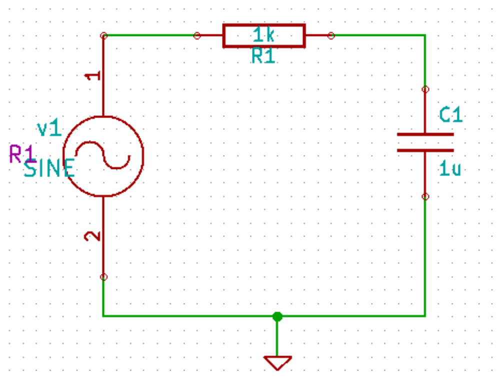


Figure 5.12: RC circuit

rules check tool from the top toolbar. Click on *Test Erc* button. The error as shown in Fig. 5.18 may be displayed. Click on close in the test erc window. There will be a green arrow pointing to the source of error in the schematic. Here it points to the ground terminal. This is shown in Fig. 5.20. To correct this error, place a `PWR_FLAG` from the EEschema library *power*. Connect the power flag to the ground terminal as shown in Fig. 5.16c. More information about PWR_FLAG is given in Sec. 5.2.5. One needs to place `PWR_FLAG` wherever the error shown in Fig. 5.18 is obtained. Repeat the

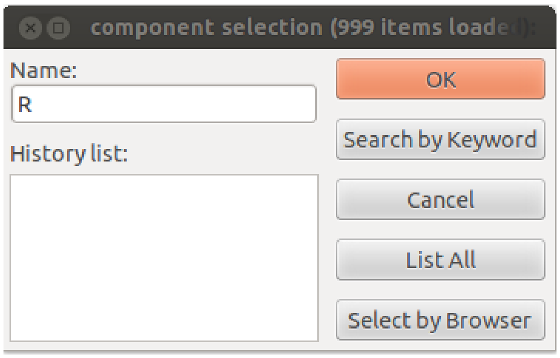


Figure 5.13: Placing a resistor using the Place a Component tool

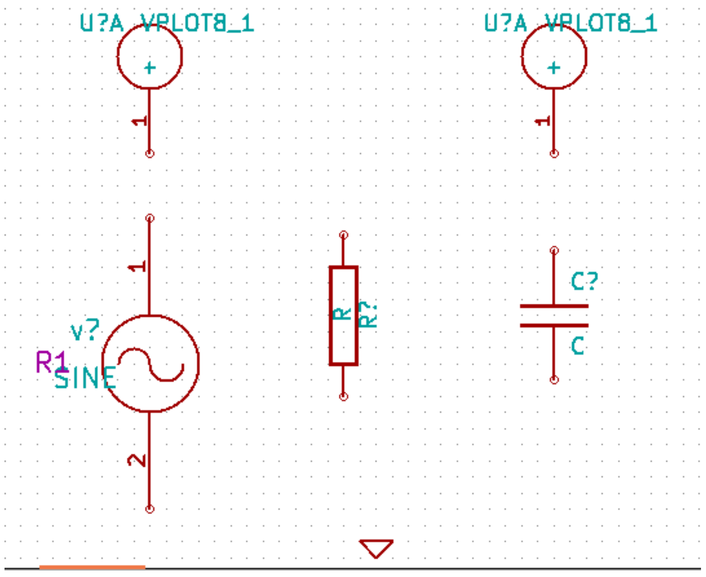


Figure 5.14: All RC circuit components placed

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

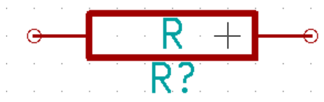
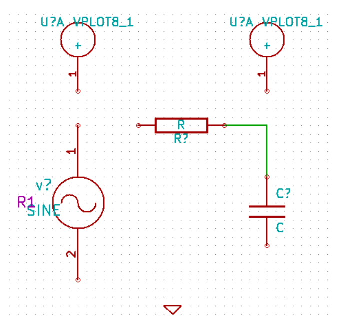
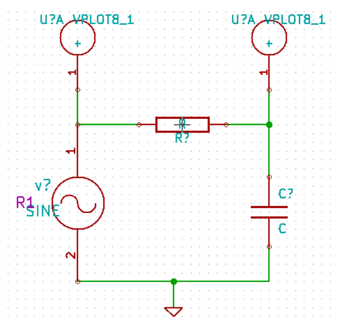


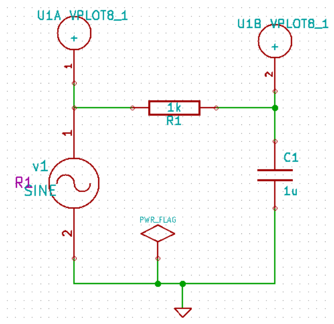
Figure 5.15: Placing the cursor (cross mark) slightly away from the letter R



(a) Initial stages



(b) Wiring done



(c) Final schematic with PWR_FLAG

Figure 5.16: Various stages of wiring

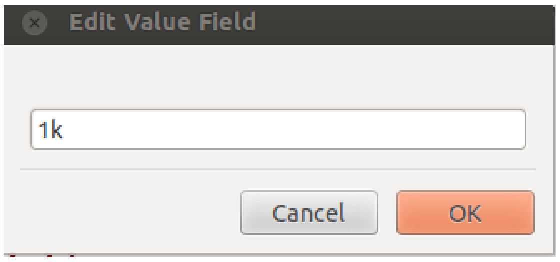
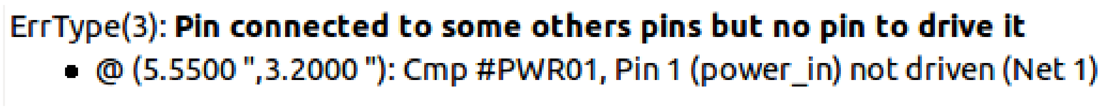


Figure 5.17: Editing value of resistor

5.3.5 Netlist generation

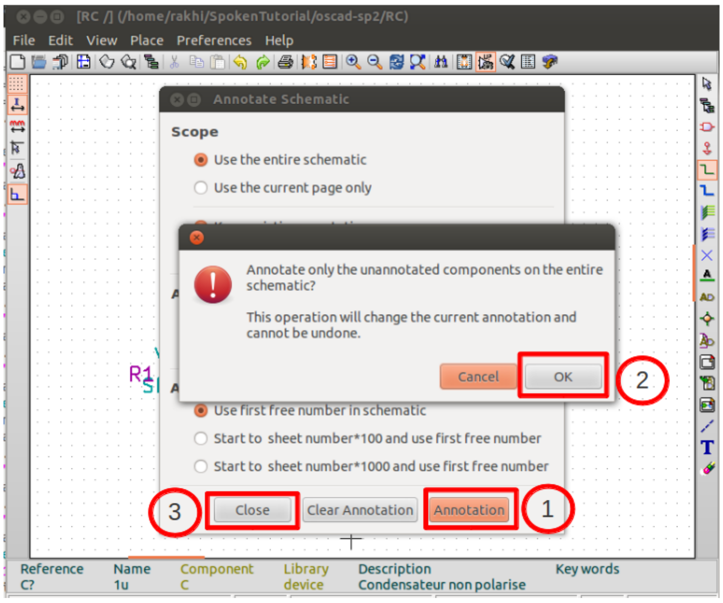
To simulate the circuit that has been created in the previous section, we need to generate its netlist. `Netlist` is a list of components in the schematic along with their connection information. To do so, click on the *Generate netlist* tool from the top toolbar. Click on spice from the window that opens up. Uncheck the option `Prefix references 'U' and 'IC' with 'X'`. Then click on *Netlist*. This is shown in Fig. 5.21. Save the netlist. This will be a `.cir` file. Do not change the directory while saving. Now the netlist is ready to be simulated. Chapter 6 explains how to perform simulations. Refer to [15] or [16] to know more about EEschema.



ErrType(3): **Pin connected to some others pins but no pin to drive it**

- @ (5.5500 ",3.2000 "): Cmp #PWR01, Pin 1 (power_in) not driven (Net 1)

Figure 5.18: ERC error



Scope

(●) Use the entire schematic
( ) Use the current page only

Annotate only the unannotated components on the entire schematic?

This operation will change the current annotation and cannot be undone.

(●) Use first free number in schematic
( ) Start to sheet number*100 and use first free number
( ) Start to sheet number*1000 and use first free number

Reference	Name	Component	Library	Description	Key words
C?	1u	C	device	Condensateur non polarise	

Figure 5.19: Steps in annotating a schematic: 1. First click on Annotation then 2. Click on Ok then 3. Click on close

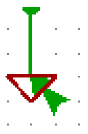
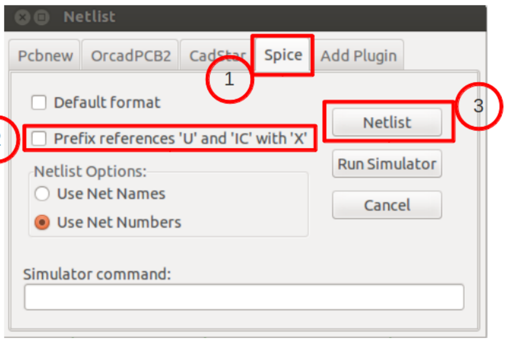


Figure 5.20: Green arrow pointing to Ground terminal indicating an ERC error



[ ] Default format
[ ] Prefix references 'U' and 'IC' with 'X'

Netlist Options:
( ) Use Net Names
(●) Use Net Numbers

Simulator command:

Figure 5.21: Steps in generating a Netlist for simulation: 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 to model circuit operations. Simulating a circuit's behaviour before actually building it can greatly improve design efficiency. Oscad uses `Ngspice` for analog, digital and mixed-level/mixed-signal circuit simulation. The various steps involved in simulating a circuit schematic in Oscad are given below:

1. Analysis insertion - This tool is used to insert the type of analysis to the netlist. It is done by the *Analysis Inserter* tool in the Oscad toolbar.

2. Netlist conversion - The netlist created in the *Schematic Editor* is converted to Ngspice format and analysis commands is appended to it. It is done by the *Netlist Converter* tool in the Oscad toolbar.

3. Ngspice simulation - Ngspice simulation of the netlist is performed. It is done by clicking on the *Ngspice* tool in the Oscad toolbar.

In the following sections, we shall describe each of the above steps.

6.1 Analysis Inserter

In order to simulate a circuit, the user must define the type of analysis to be done on the circuit. The types of analysis include `Operating point analysis`, `DC analysis`, `AC analysis`, `transient analysis`, etc. The user should also specify the options corresponding to each analysis. This is facilitated by the *Analysis Inserter* tool in Oscad.

Analysis Inserter generates the commands for Ngspice. When one clicks on *Analysis Inserter* from the Oscad toolbar, one gets the Analysis Inserter GUI as shown in Fig. 6.1. The various tabs in this GUI correspond to the various types of analysis. The user can enter the details, needed to perform simulation, in the corresponding fields under these tabs.

6.1.1 Types of analysis

Oscad supports three types of analyses: 1. DC Analysis (Operating Point and DC Sweep) 2. AC Small-signal Analysis 3. Transient Analysis. Other analysis in the *Analysis Inserter* are currently under progress. The different types of analyses supported in Oscad are explained below [17].

DC analysis

The `DC analysis` determines the dc operating point of the circuit with inductors shorted and capacitors opened. The DC analysis options are specified on the *.dc* 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 could be displayed or printed out.

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 linearised, small-signal models for nonlinear devices. The `DC analysis` can also be used to generate dc transfer curves: a specified independent voltage or current source is stepped over a user-specified range and the dc output variables are stored for each sequential source value.

AC small-signal analysis

`AC analysis` is limited to analog nodes. It 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 behaviour 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 linearised, 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, trans impedance, 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.

Transient analysis

`Transient analysis` is an extension of `DC analysis` to the time domain. A `transient`

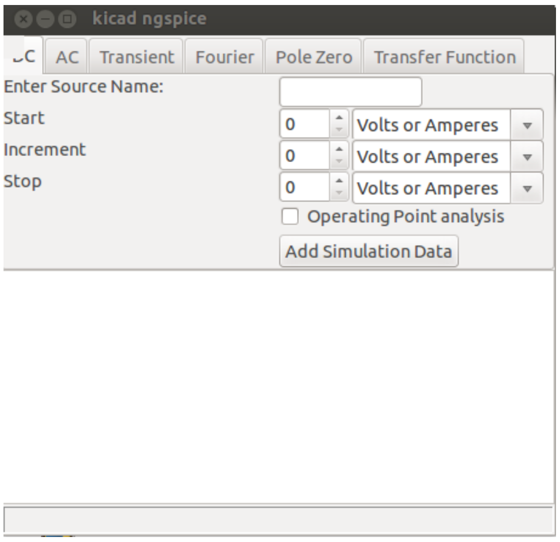


Figure 6.1: Analysis Inserter GUI

`analysis` begins by obtaining a DC solution to provide a point of departure for simulating time-varying behaviour. Once the DC solution is obtained, the time-dependent aspects of the system are reintroduced and the simulator algorithms incrementally solve for the time varying behaviour of the entire system. Inconsistencies in node values are resolved by the 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 behaviour for the specified time interval are accessible. 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.

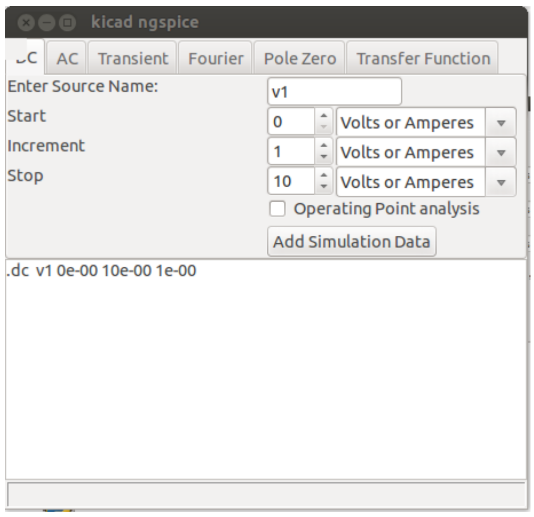
6.1.2 DC analysis inserter

By default `DC analysis` option appears when one clicks on *Analysis Inserter*. Here we need to give the details of input *source name*, *start value* of input, *increment* and *stop* value. Once this is done, click on *Add Simulation Data*.

Fig. 6.2a gives an example of `DC analysis` inserter. In this example, `v1` is the input voltage source which *starts* at `0 Volt`, *increments* by `1 Volt` and *stops* at `10 Volt`. On clicking *Add Simulation Data*, the analysis command is generated and is of the form:

```
.dc sourcename vstart vstop vincr
```

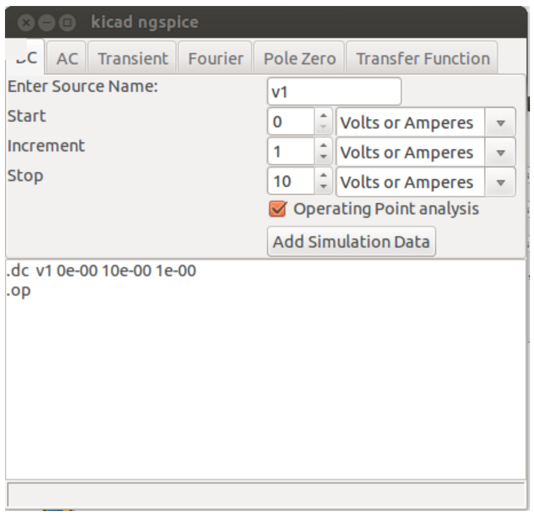
The `.dc` line defines the dc transfer curve source and sweep limits (with capacitors open and inductors shorted). `srcnam` is the name of an independent voltage or current source. `vstart`, `vstop`, and `vincr` are the starting, final, and incrementing values respectively, of the source.



Enter Source Name: v1
Start: 0 Volts or Amperes
Increment: 1 Volts or Amperes
Stop: 10 Volts or Amperes
Operating Point analysis: ☐

```
.dc v1 0e-00 10e-00 1e-00
```

(a) DC Analysis command added



Enter Source Name: v1
Start: 0 Volts or Amperes
Increment: 1 Volts or Amperes
Stop: 10 Volts or Amperes
Operating Point analysis: ✓

```
.dc v1 0e-00 10e-00 1e-00
.op
```

(b) OP Analysis command added

Figure 6.2: Different analysis options

When we check the option *Operating Point analysis* on the DC analysis window, `.op` gets appended to the analysis statement. This is shown in Fig. 6.2b. The inclusion of the line `.op` in the analysis file directs Ngspice to determine the dc operating point of the circuit with inductors shorted and capacitors opened.

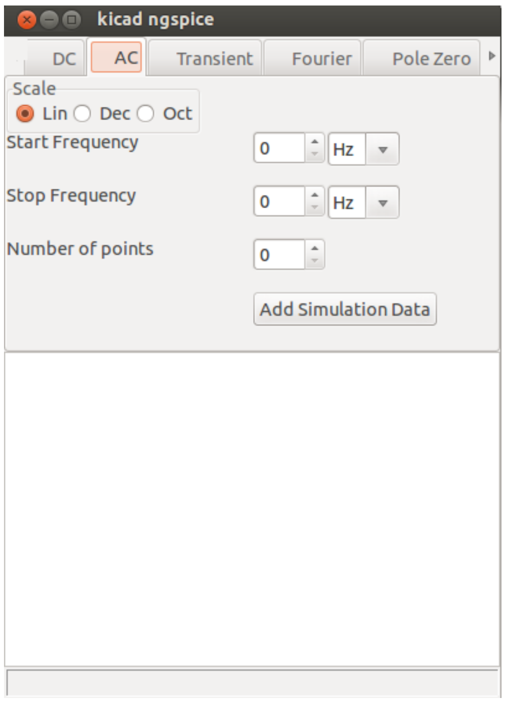
6.1.3 AC analysis inserter

When one clicks on the option *AC* in the *Analysis Inserter* GUI, the window given in Fig. 6.3a appears. Here one needs to enter the details of *scale*, *start frequency*, *stop frequency* and *Number of points*.

After entering these values, click on *Add Simulation Data*. The analysis statement is generated. This is in one of the three forms listed below, depending on the type of *scale* that one chooses. The types of *scale* available are *dec*, *oct*, and *lin*, the usage of which is explained below:

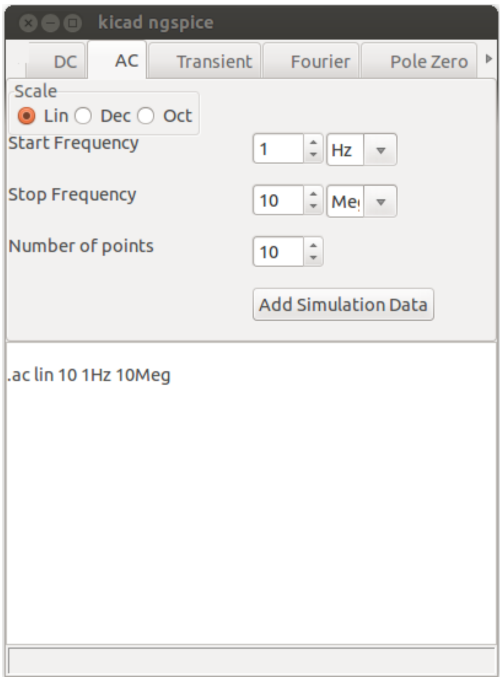
```
.ac dec nd fstart fstop
.ac oct no fstart fstop
.ac lin np fstart fstop
```

Here, `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. An example of *lin* scale is given in Fig. 6.3b. Here the *start frequency* is `1 Hz`, *stop frequency* is `10 Meg` and *number of points* is 10.



Start Frequency 0 Hz
Stop Frequency 0 Hz
Number of points 0

(a) AC Analysis GUI



Start Frequency 1 Hz
Stop Frequency 10 Meg
Number of points 10

```
.ac lin 10 1Hz 10Meg
```

(b) AC Analysis command added

Figure 6.3: AC Analysis GUI

If the `.ac` analysis is included in the analysis 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. While creating the schematic for performing ac analysis, add the component `AC` from the *sourcesSpice* library.

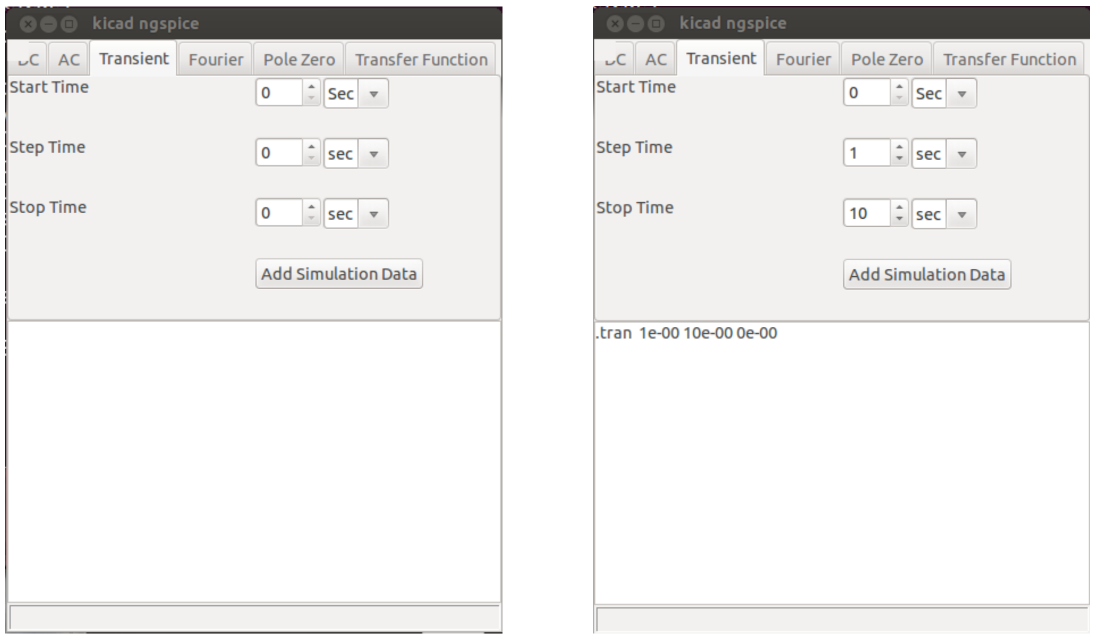
6.1.4 Transient analysis inserter

When one clicks on the option *Transient* in the *Analysis Inserter* GUI, the window given in Fig. 6.4a appears. Here one needs to enter the details of *start time*, *step time*, and *stop time*. After entering these values, click on *Add Simulation Data*. The analysis statement is generated. It is of the form:

```
.tran tstep tstop tstart
```

Here, `tstep` is the printing or plotting increment for line-printer output. For use with the post-processor, `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 are stored. In the



Start Time 0 Sec
Step Time 0 sec
Stop Time 0 sec

(a) Initial stage

Start Time 0 Sec
Step Time 1 sec
Stop Time 10 sec

```
.tran 1e-00 10e-00 0e-00
```

(b) Transient Analysis command added

Figure 6.4: Transient Analysis GUI

interval <`tstart, tstop`>, the circuit is analyzed and outputs are stored. An example of transient analysis inserter is given in Fig. 6.4b. Here *start time* is `0 sec`, *step time* is `1 sec` and *stop time* is `10 sec`.

6.1.5 Saving the analysis file

After entering the details of analysis we need to save the `analysis` file, which contains the options we added. Save the `analysis` file as explained below:

Click on *File* from the top menu bar. Click on *Save*. This is shown in Fig. 6.5. Click on *Save* in the *Choose a File* dialog box as shown in Fig. 6.6. The options added in the *Analysis Inserter* GUI are saved in a file named `analysis`.

6.2 Modifying KiCad netlist for Ngspice simulation

The *Schematic Editor* generates a `netlist file`, which describes the electrical connections between circuit components. This is a `Spice` netlist which cannot be directly used for Ngspice simulation due to compatibility issues. This needs to be converted to Ngspice format. For this, click on *Netlist Converter* tool from the Oscad toolbar after

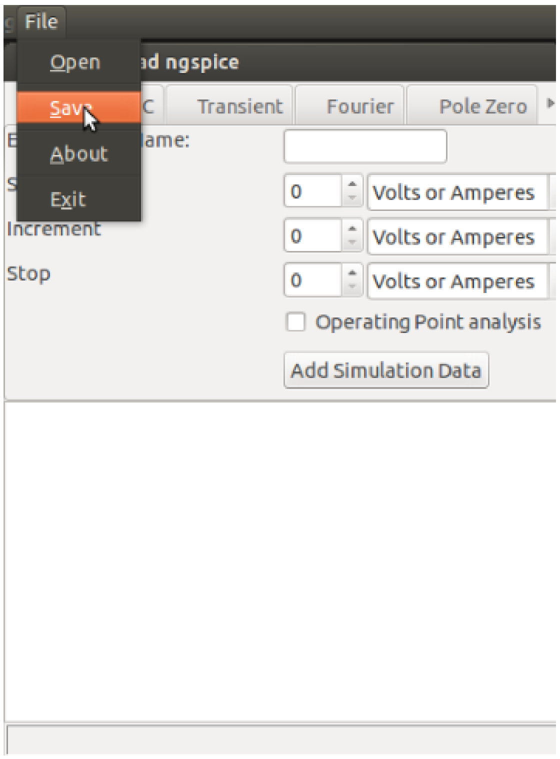


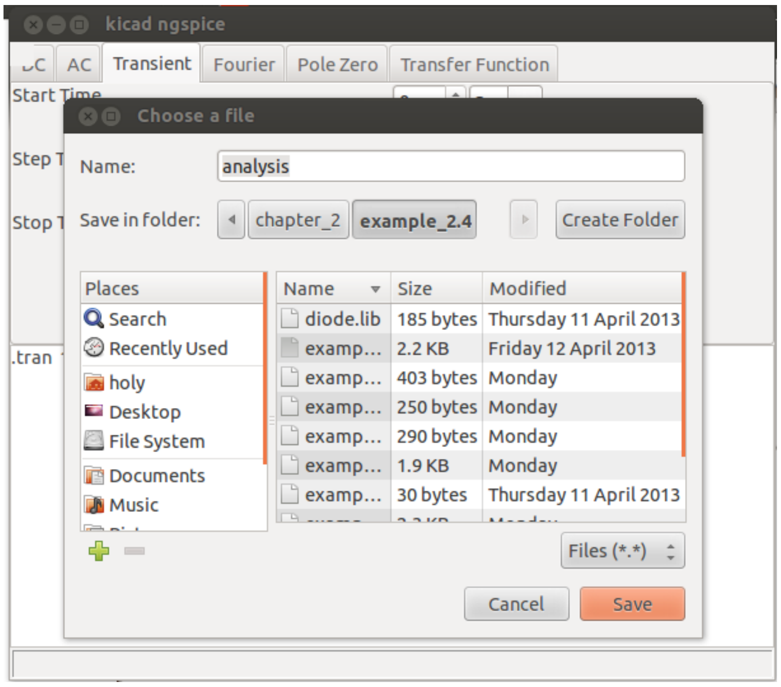
Figure 6.5: Choosing the Save option from File menu

performing analysis insertion. Oscad netlist converter performs the following operations to generate Ngspice compatible netlist.

Insert parameters for fictitious components: For simulation of the circuit, the values of sources must be specified. For example, for a `sine wave` voltage source, parameters such as *frequency*, *amplitude* and *offset* are required to be entered.

The *Netlist Converter* scans the netlist file and asks for parameter values for the sources wherever required. When we click on *Netlist Converter*, a terminal window appears. It asks for various parameter values. The parameter values that one needs to enter, varies with the voltage/current sources added in the schematic for simulation. For example:

1. When `sinusoidal source` is available in schematic - in the terminal the details of sinusoidal source like *offset value*, *amplitude*, *frequency*, *delay time*, *damping factor* are asked. This is shown in Fig. 6.7.
2. When `DC source` of unknown value is available in schematic, it asks for the value of DC source. In the example shown in Fig. 6.8a, the value of DC source is 10V.
3. When `AC source` is available in schematic, it asks for amplitude of AC source,



Name: analysis

Save in folder: chapter_2 / example_2.4

Name	Size	Modified
diode.lib	185 bytes	Thursday 11 April 2013
examp…	2.2 KB	Friday 12 April 2013
examp…	403 bytes	Monday
examp…	250 bytes	Monday
examp…	290 bytes	Monday
examp…	1.9 KB	Monday
examp…	30 bytes	Thursday 11 April 2013

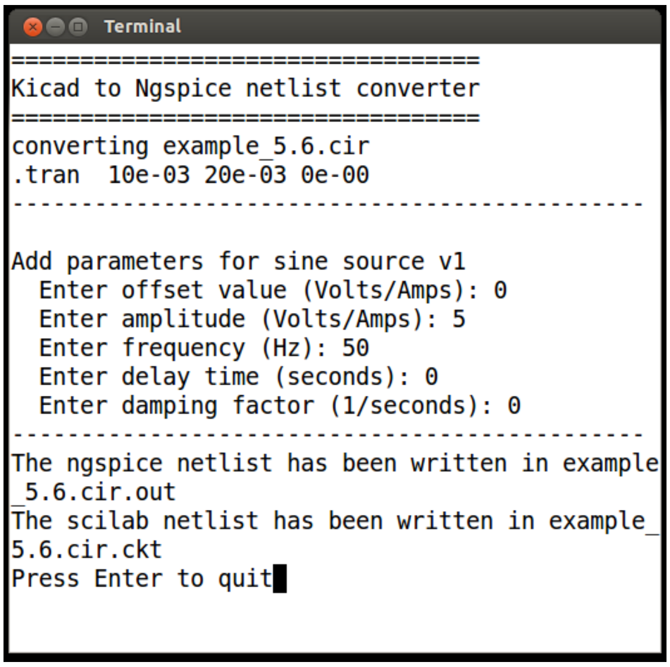
Figure 6.6: Saving the analysis information to netlist

which in this case is `1V`. This is shown in Fig. 6.8b.

Convert IC into discrete blocks: As `EEschema` (schematic editor used in Oscad) is intended for `PCB` Designing, it creates netlist in terms of ICs and not components, e.g. if the circuit 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 by considering proper input and output connections and IC specifications (voltage levels, speed of the operation, etc.).

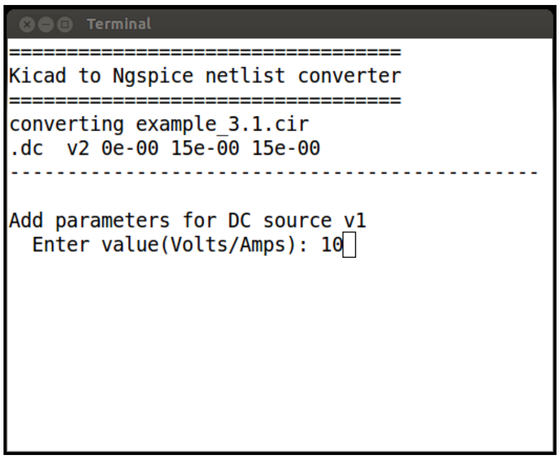
Insert Digital-to-Analog (D-to-A) and Analog-to-Digital (A-to-D) converter at appropriate places: Oscad provides capability to perform mixed mode simulation. Thus circuits with analog and digital components can be analyzed. In order to simulate such kind of circuits, D-to-A and A-to-D converters are inserted at appropriate places. The netlist generated from EEschema is assumed to have analog connections and for digital components, A-to-D converter for inputs and D-to-A converter for outputs are added.

Insert plotting and printing statements: There is a library for plotting and printing the voltages and currents in the circuit. The *Netlist Converter* adds appropriate



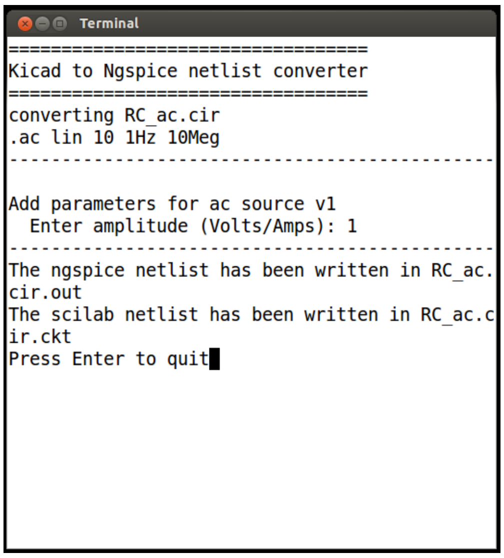
```
==================================
Kicad to Ngspice netlist converter
==================================
converting example_5.6.cir
.tran  10e-03 20e-03 0e-00
----------------------------------------------

Add parameters for sine source v1
  Enter offset value (Volts/Amps): 0
  Enter amplitude (Volts/Amps): 5
  Enter frequency (Hz): 50
  Enter delay time (seconds): 0
  Enter damping factor (1/seconds): 0
----------------------------------------------
The ngspice netlist has been written in example
_5.6.cir.out
The scilab netlist has been written in example_
5.6.cir.ckt
Press Enter to quit
```

Figure 6.7: Parameters to be added for sinusoidal source



```
==================================
Kicad to Ngspice netlist converter
==================================
converting example_3.1.cir
.dc  v2 0e-00 15e-00 15e-00
----------------------------------------------

Add parameters for DC source v1
  Enter value(Volts/Amps): 10
```

(a) Parameters to be added for dc source



```
==================================
Kicad to Ngspice netlist converter
==================================
converting RC_ac.cir
.ac lin 10 1Hz 10Meg
----------------------------------------------

Add parameters for ac source v1
  Enter amplitude (Volts/Amps): 1
----------------------------------------------
The ngspice netlist has been written in RC_ac.
cir.out
The scilab netlist has been written in RC_ac.c
ir.ckt
Press Enter to quit
```

(b) Parameters to be added for ac source

Figure 6.8: Parameters for dc and ac sources for Netlist Converter

printing and plotting commands (current or voltage plot, or single or differential plot) in the netlist depending on the print/plot components. Ngspice can only plot currents through (independent) voltage sources. *Netlist Converter* inserts a zero volt voltage source in series with the component through which current needs to be computed. Thus current through any component can be obtained.

Insert analysis and option: *Netlist Converter* inserts analysis commands, created using the *Analysis Inserter* GUI, into the netlist.

Insert models and subcircuits: Oscad netlist converter inserts models or subcircuits for required components into the netlist. To know more about model building and subcircuit creation, refer to Chapter 8.

Simulation: After netlist conversion, a `.cir.out` file is generated which is compatible with Ngspice simulation software. Let us see how to simulate this file and obtain the results.

Ngspice simulation in Oscad: Click on *Ngspice* from the Oscad toolbar. The Ngspice terminal and waveform windows appear. An example is shown in Fig. 6.9.[3]

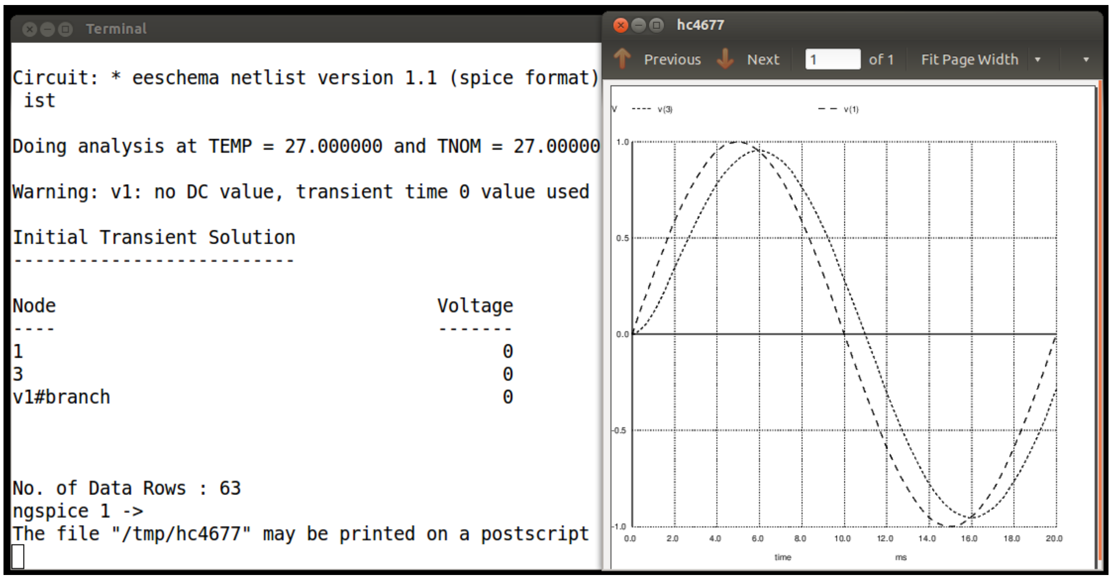
6.3 Examples

Let us see a few simulation examples which describes the use of *Analysis Inserter*, *Netlist Converter* and *Ngspice*.

6.3.1 DC analysis

Consider the `nodalExample_plot` (nodal analysis) given in the `Examples` folder available on the Oscad web page `www.oscad.in`. Open *Schematic Editor* and generate spice netlist as explained in Chapter 5. Click on *Analysis Inserter*. Here we decide which type of analysis we want to do. Let us do DC Analysis.

[3]Note: Ngspice graph window has a black background, by default. The graph window also has a button named `hardcopy`. When one clicks on this button, a message "The file `/tmp/hcxxxx` may be printed on a postscript printer" will be displayed on the Ngspice terminal window (`xxxx` is a number). This means that a copy of the graph has been stored in the location `/tmp/` with the name `hcxxxx`. This copy can be viewed by opening the above file. All the Ngspice graphs in this chapter and a few graphs in the Appendix are obtained using the `hardcopy` feature. Hence, these Ngspice simulation graphs have a white background. These graphs show v(y), where y is the node in the schematic where the plot component has been placed. To know about these nodes, see the `.cir` netlist file in the project directory. This file may have a line that is similar to `U1 3 2 VPLOT8_1`. This means that VPLOT8_1 is inserted between the nodes 3 and 2 in the schematic. These node numbers are randomly assigned by Ngspice. Ground is always assigned 0.



```
Circuit: * eeschema netlist version 1.1 (spice format)
 ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.00000

Warning: v1: no DC value, transient time 0 value used

Initial Transient Solution
--------------------------


Node                                   Voltage
----                                   -------
1                                            0
3                                            0
v1#branch                                    0



No. of Data Rows : 63
ngspice 1 ->
The file "/tmp/hc4677" may be printed on a postscript
```

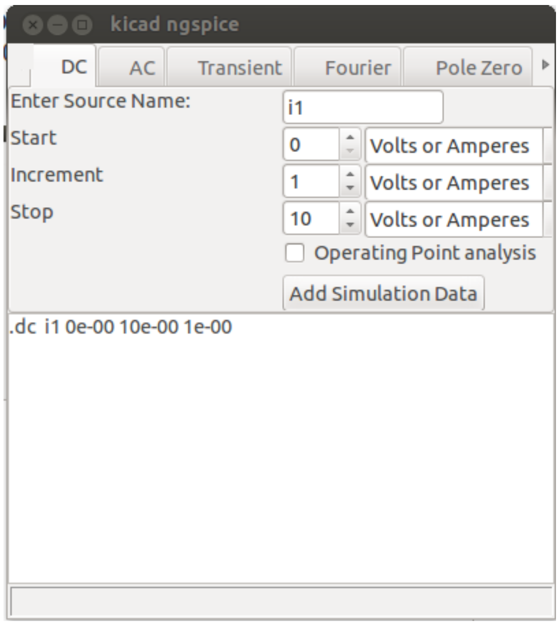
Figure 6.9: Ngspice simulation result showing v(3) and v(1), see Footnote 3 on Page 54. (v(1) plot is shown with long dashes)

Click on *DC* and enter the following details: *source name* = i1, *start* = 0A, *Increment* = 1A, *stop* = 10A and then click on *Add Simulation Data*. The window as shown in Fig. 6.10 is displayed. Save and close the analysis. Now click on *Netlist Converter* which shows the terminal given in Fig. 6.11. Press `Enter` key. After this click on *Ngspice*, the Ngspice terminal with waveform window as shown in Fig. 6.12, appears.

6.3.2 AC small-signal analysis

Consider the `BJT_amplifier` example from the `Example` folder available in Oscad website. Open *Schematic Editor* and generate spice Netlist as explained in Chapter 5. Click on *Model Builder* tool. A terminal window opens up with the message `Please enter type of BJT (NPN/PNP):` Type `npn` and press enter. This is shown in Fig. 6.13a. A new window will open asking to select the model. This is shown in Fig. 6.13b. Select `npn` and press *OK*. A new window to edit the BJT parameters opens up. This is shown in Fig. 6.14. These are nothing but spice parameters for BJT [14]. Some of them are explained in Table A.2 on Page 106. After editing the parameter list, click on *OK*. An `Info` dialog box opens up. Click on *OK*. Close the *Model Builder* windows. To know more about the *Model Builder* tool, refer to Chapter 8.

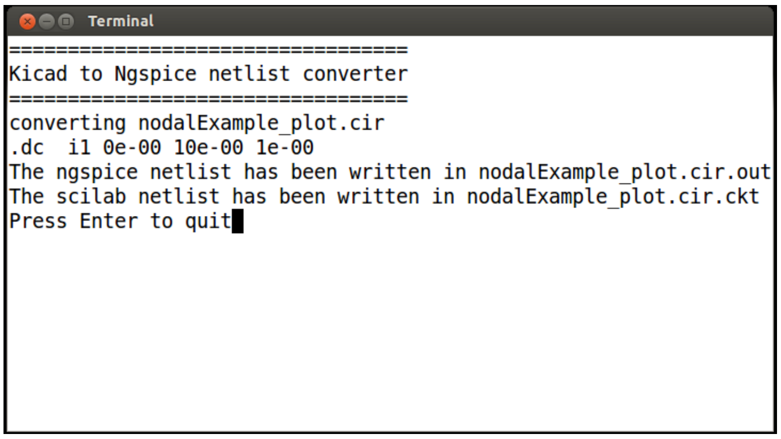
Let us now do AC analysis. Click on *Analysis Inserter*. Click on *AC* and add the following details: *scale* = `dec`, *start frequency* = 100 Hz, *stop frequency* = 10 KHz, *number of points* = 100 and then click on *Add Simulation Data*. This is shown in



Enter Source Name: i1
Start 0 Volts or Amperes
Increment 1 Volts or Amperes
Stop 10 Volts or Amperes
☐ Operating Point analysis

```
.dc i1 0e-00 10e-00 1e-00
```

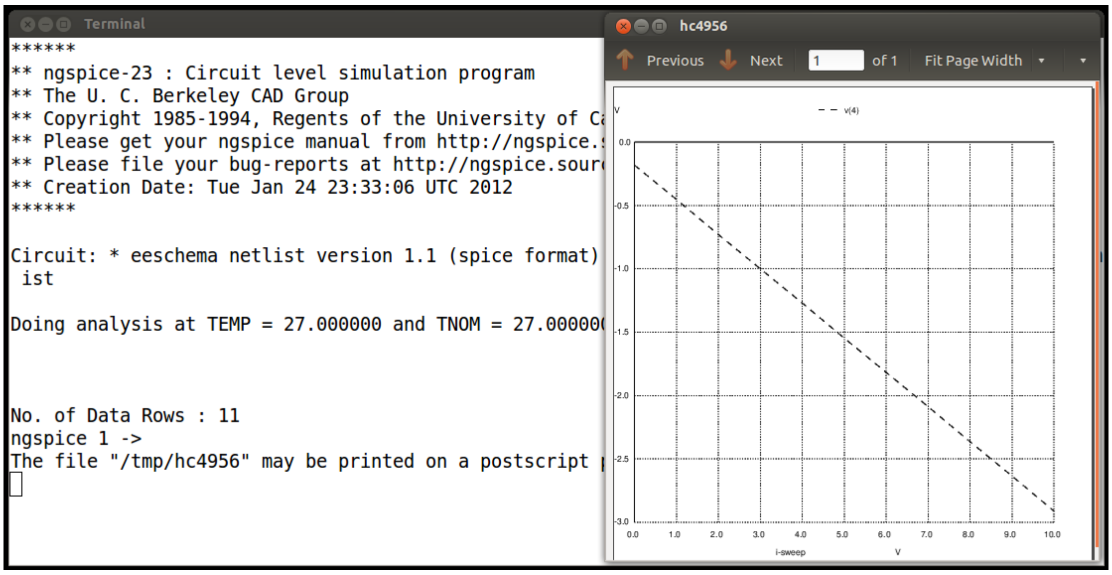
Figure 6.10: Adding simulation data for DC analysis



```
==================================
Kicad to Ngspice netlist converter
==================================
converting nodalExample_plot.cir
.dc  i1 0e-00 10e-00 1e-00
The ngspice netlist has been written in nodalExample_plot.cir.out
The scilab netlist has been written in nodalExample_plot.cir.ckt
Press Enter to quit
```

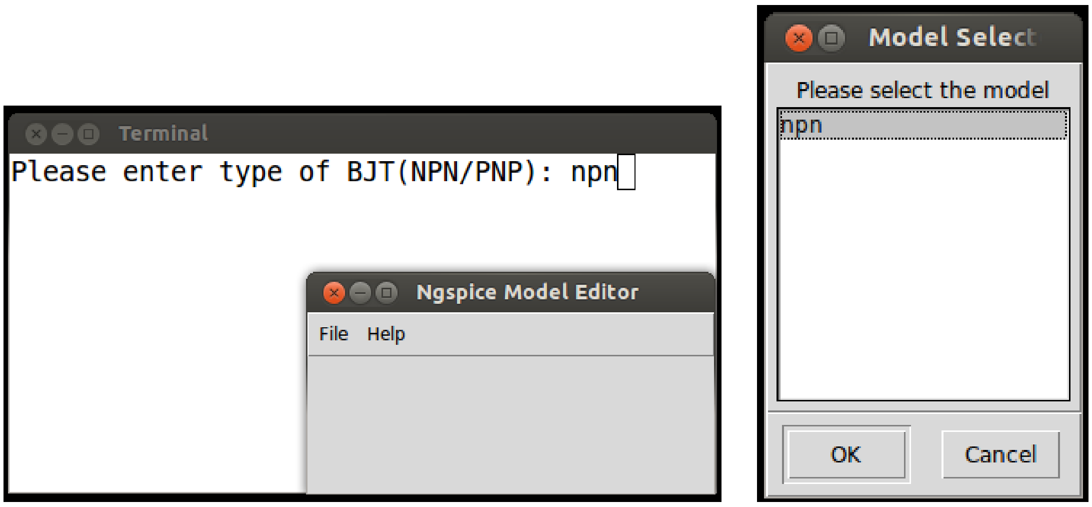
Figure 6.11: Netlist conversion terminal window for DC analysis

Fig. 6.15a. Now click on *Netlist Converter* which opens up a terminal. It asks for the amplitudes of `AC` and `DC` sources. Type `20` for DC and `0.025` for AC and press the `Enter` key. A `.cir.out` netlist file is generated. This is shown in Fig. 6.15b. After this click on *Ngspice*, the Ngspice terminal with waveform window as shown in Fig. 6.16, appears.



```
******
** ngspice-23 : Circuit level simulation program
** The U. C. Berkeley CAD Group
** Copyright 1985-1994, Regents of the University of Ca
** Please get your ngspice manual from http://ngspice.s
** Please file your bug-reports at http://ngspice.sour
** Creation Date: Tue Jan 24 23:33:06 UTC 2012
******

Circuit: * eeschema netlist version 1.1 (spice format)
 ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.000000


No. of Data Rows : 11
ngspice 1 ->
The file "/tmp/hc4956" may be printed on a postscript p
```

Figure 6.12: DC analysis: Ngspice simulation result showing v(4), see Footnote 3 on Page 54.



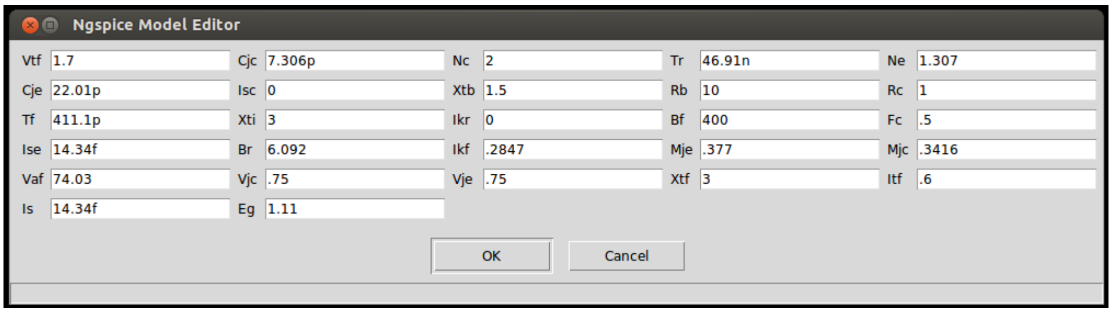
```
Please enter type of BJT(NPN/PNP): npn
```

(a) Selecting BJT type (b) Selecting model

Figure 6.13: Steps in selecting model for BJT

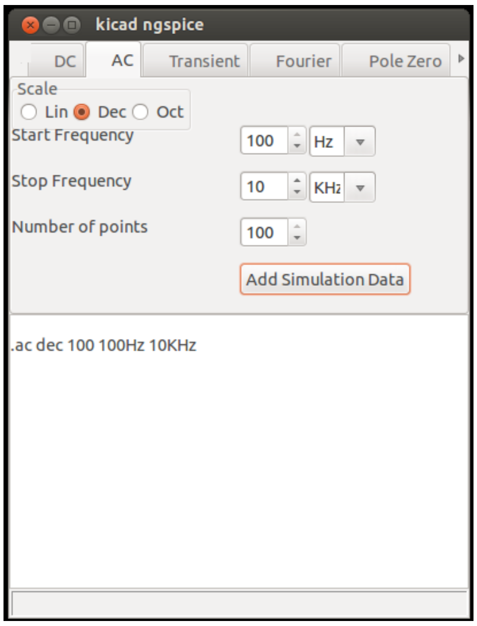
6.3.3 Transient analysis

Let us use the `RC circuit` project created in Chapter 5 to do `transient analysis`. Open the *Analysis Inserter* tool. Click on *Transient* and then add the following details:



Vtf	1.7	Cjc	7.306p	Nc	2	Tr	46.91n	Ne	1.307
Cje	22.01p	Isc	0	Xtb	1.5	Rb	10	Rc	1
Tf	411.1p	Xti	3	Ikr	0	Bf	400	Fc	.5
Ise	14.34f	Br	6.092	Ikf	.2847	Mje	.377	Mjc	.3416
Vaf	74.03	Vjc	.75	Vje	.75	Xtf	3	Itf	.6
Is	14.34f	Eg	1.11						

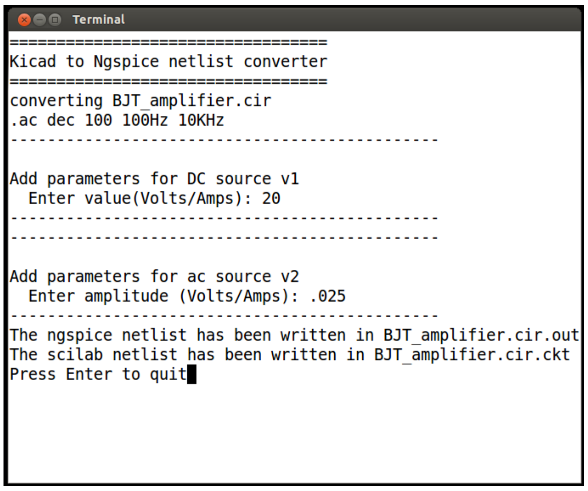
Figure 6.14: GUI for modifying BJT model parameters



Scale: Lin [✓ Dec] Oct
Start Frequency 100 Hz
Stop Frequency 10 KHz
Number of points 100

.ac dec 100 100Hz 10KHz

(a) AC analysis inserter



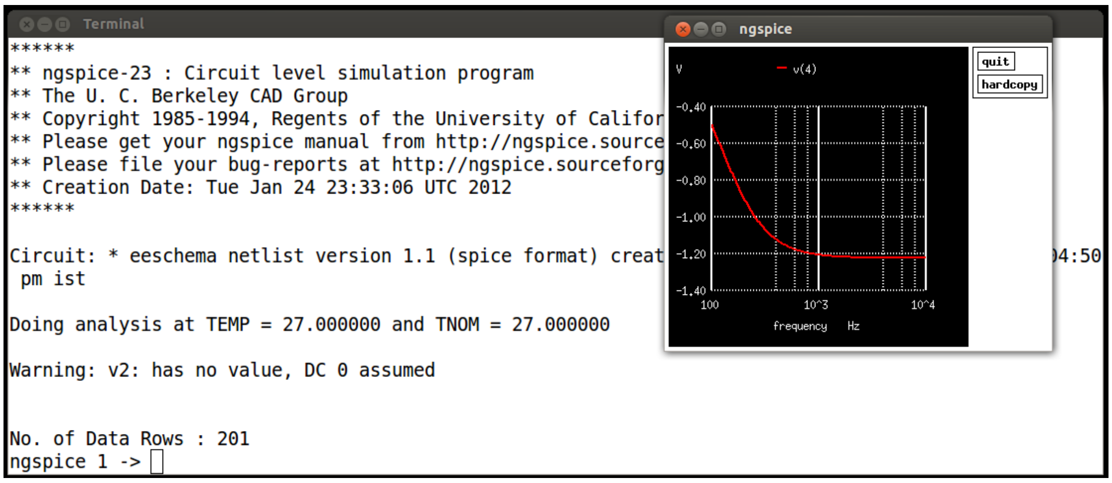
```
==================================
Kicad to Ngspice netlist converter
==================================
converting BJT_amplifier.cir
.ac dec 100 100Hz 10KHz
----------------------------------------------

Add parameters for DC source v1
  Enter value(Volts/Amps): 20
----------------------------------------------
----------------------------------------------

Add parameters for ac source v2
  Enter amplitude (Volts/Amps): .025
----------------------------------------------
The ngspice netlist has been written in BJT_amplifier.cir.out
The scilab netlist has been written in BJT_amplifier.cir.ckt
Press Enter to quit
```

(b) AC analysis - Netlist conversion

Figure 6.15: AC analysis insertion and Netlist conversion

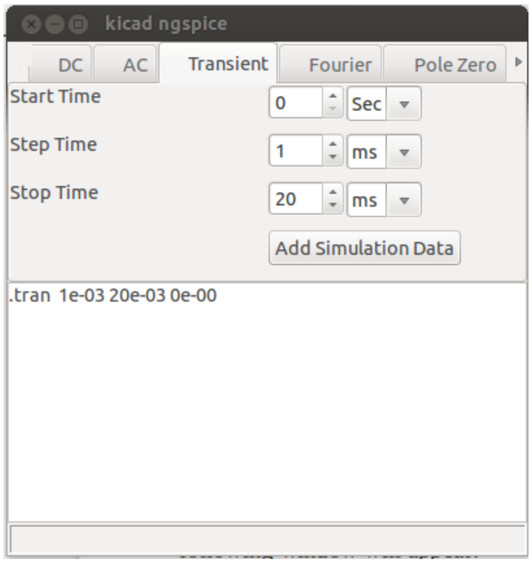
start time = 0 sec, *step time* = 1 ms, *stop time* = 20 ms and then click on *Add Simulation Data*. This is shown in Fig. 6.17a. Now click on *Netlist Converter* which opens up a terminal. It asks for various parameters for the sine source. Enter the details as given below: *Offset value* = 0, *Amplitude* = 1, *frequency* = 50, *delay time* = 0, *damping factor* = 0 and press `Enter key`. This is shown in Fig. 6.17b.

Now click on *Ngspice*. The Ngspice terminal with waveform window as shown in Fig. 6.18, appears.



```
******
** ngspice-23 : Circuit level simulation program
** The U. C. Berkeley CAD Group
** Copyright 1985-1994, Regents of the University of Califor
** Please get your ngspice manual from http://ngspice.source
** Please file your bug-reports at http://ngspice.sourceforg
** Creation Date: Tue Jan 24 23:33:06 UTC 2012
******

Circuit: * eeschema netlist version 1.1 (spice format) creat                                )4:50
 pm ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.000000

Warning: v2: has no value, DC 0 assumed


No. of Data Rows : 201
ngspice 1 ->
```

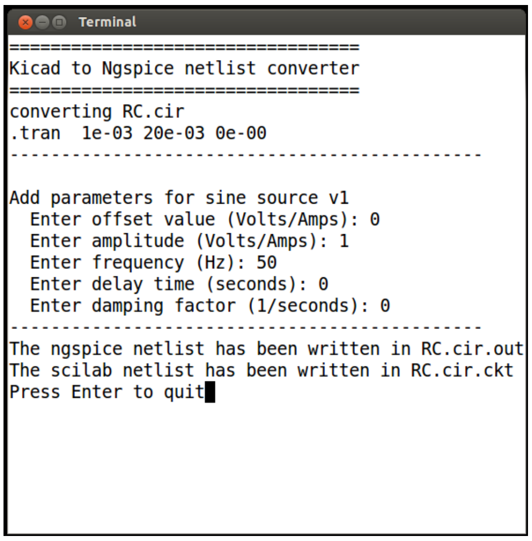
Figure 6.16: AC analysis: Simulation results showing v(4) versus frequency, see Footnote 3 on Page 54



Start Time 0 Sec
Step Time 1 ms
Stop Time 20 ms

```
.tran 1e-03 20e-03 0e-00
```

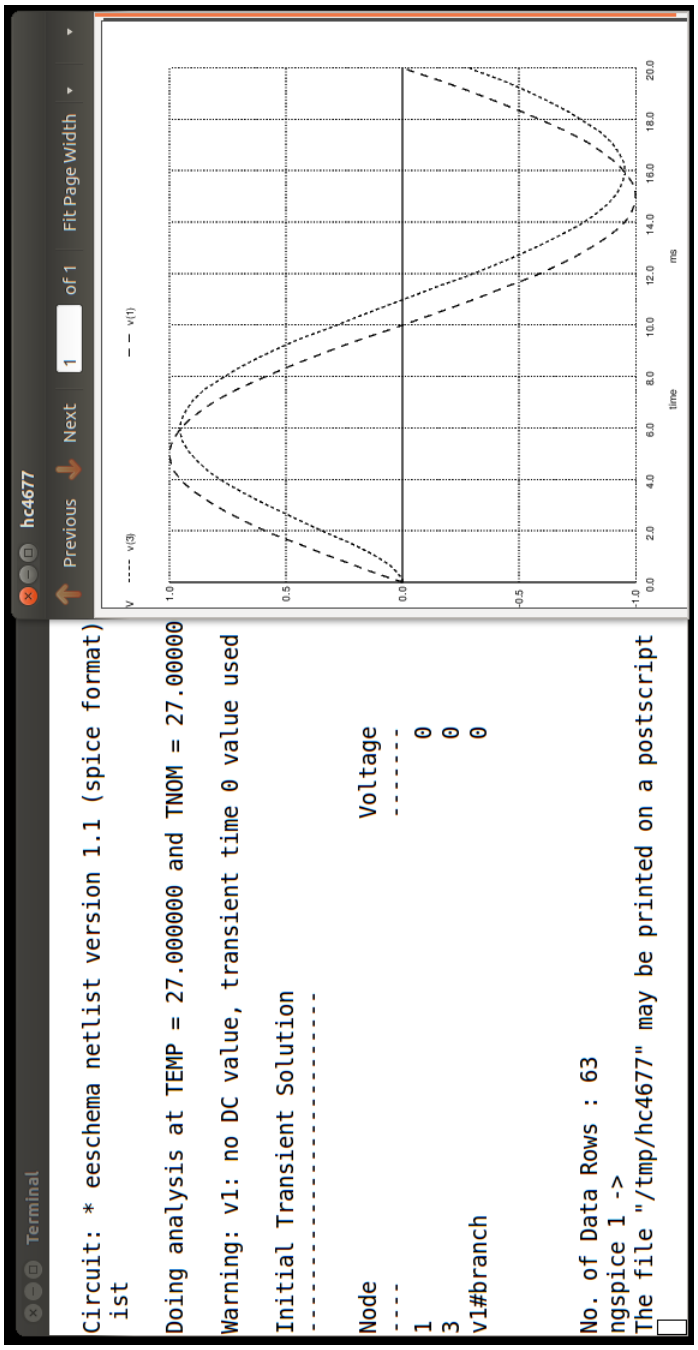
(a) Transient analysis inserter



```
==================================
Kicad to Ngspice netlist converter
==================================
converting RC.cir
.tran  1e-03 20e-03 0e-00
----------------------------------------------

Add parameters for sine source v1
  Enter offset value (Volts/Amps): 0
  Enter amplitude (Volts/Amps): 1
  Enter frequency (Hz): 50
  Enter delay time (seconds): 0
  Enter damping factor (1/seconds): 0
----------------------------------------------
The ngspice netlist has been written in RC.cir.out
The scilab netlist has been written in RC.cir.ckt
Press Enter to quit
```

(b) Netlist conversion in transient analysis

Figure 6.17: Transient analysis insertion and Netlist conversion



```
Circuit: * eeschema netlist version 1.1 (spice format)
 ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.00000

Warning: v1: no DC value, transient time 0 value used

Initial Transient Solution
--------------------------

Node                                   Voltage
----                                   -------
1                                            0
3                                            0
v1#branch                                    0


No. of Data Rows : 63
ngspice 1 ->
The file "/tmp/hc4677" may be printed on a postscript
```

Figure 6.18: Transient analysis results showing v(3) and v(1), see Footnote 3 on Page 54. v(1) plot is shown with long dashes.

Chapter 7

PCB Design

Printed Circuit Board (PCB) design is an important step in electronic system design. Every component of the circuit needs to be placed and connections routed to minimise delay and area. Each component has an associated footprint. Footprint refers to the physical layout of a component that is required to mount it on the PCB. PCB design involves associating footprints to all components, placing them appropriately to minimise wire length and area, connecting the footprints using tracks/vias and finally extracting the required files needed for printing the PCB. Let us see the steps to design PCB using Oscad.

7.1 Schematic creation for PCB design

In Chapter 5, we have seen the differences between schematic for simulation and schematic for PCB design. Let us design the PCB for an RC circuit. A resistor, capacitor, ground, power flag and a connector are required. Connectors are used to take signals in and out of the PCB.

Create the circuit schematic as shown in Fig. 7.1. The two pin connector (*CONN_2*) can be placed from the EEschema library *conn*. See Sec. 5.2.6 to know more about EEschema library *conn*. Do the annotation and test for ERC. Refer to Chapter 5 to know more about basic steps in schematic creation.

7.1.1 Netlist generation for PCB

The netlist for PCB is different from that for simulation. To generate netlist for PCB, click on the *Generate netlist* tool from the top toolbar in Schematic editor. In the Netlist window, under the tab *Pcbnew*, click on the button *Netlist*. This is shown in Fig. 7.2. Click on *Save* in the Save netlist file dialog box that opens up. Do not change the directory or the name of the netlist file. Save the schematic and close the schematic editor. *Note that the netlist for PCB has an extension* `.net`. *The netlist created for*

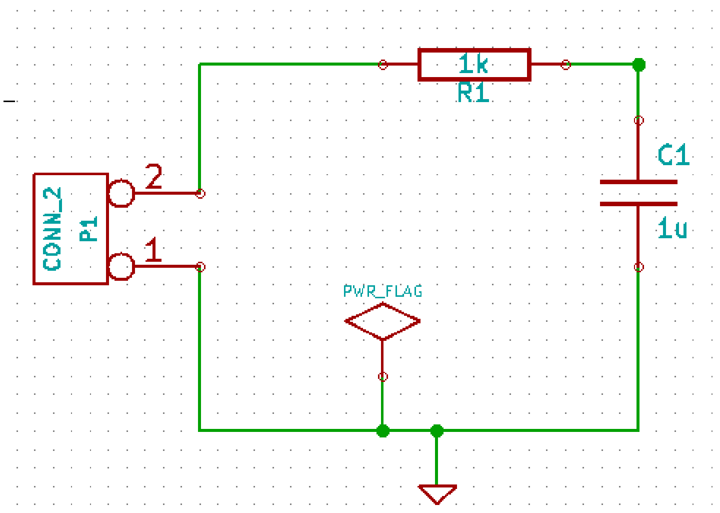


Figure 7.1: Final circuit schematic for RC low pass circuit

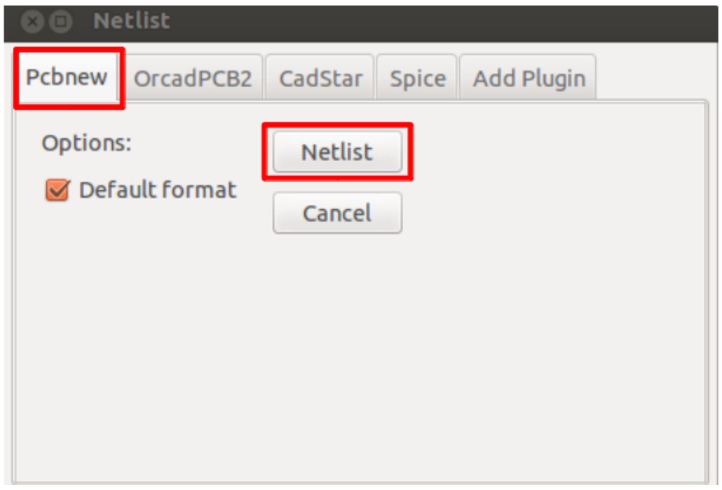
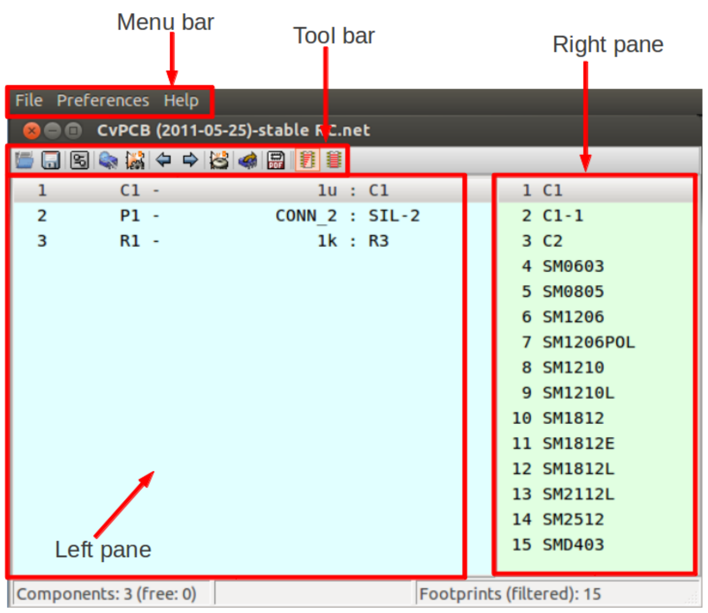


Figure 7.2: Netlist generation for PCB

simulation has an extension `.cir`.

7.1.2 Mapping of components using Footprint Editor

Once the netlist for PCB is created, one needs to map each component in the netlist to a footprint. The tool *Footprint Editor* is used for this. Oscad uses `CvPcb` as its footprint editor. `CvPcb` is the footprint editor tool in KiCad.



```
1    C1 -        1u : C1
2    P1 -    CONN_2 : SIL-2
3    R1 -        1k : R3
```

```
 1 C1
 2 C1-1
 3 C2
 4 SM0603
 5 SM0805
 6 SM1206
 7 SM1206POL
 8 SM1210
 9 SM1210L
10 SM1812
11 SM1812E
12 SM1812L
13 SM2112L
14 SM2512
15 SMD403
```

Figure 7.3: Footprint editor with the menu bar, toolbar, left pane and right pane marked

7.1.3 Familiarising the Footprint Editor tool

If one opens the *Footprint Editor* after creating the `.net` netlist file, the Footprint editor as shown in Fig. 7.3 will be obtained. The menu bar and toolbars and the panes are marked in this figure. The menu bar will be available in the top left corner. The left pane has a list of components in the netlist file and the right pane has a list of available footprints for each component. *Note that if the Footprint Editor is opened before creating a '.net' file, then the left and right panes will be empty.*

Toolbar

Some of the important tools in the toolbar are shown in Fig. 7.4. They are explained below:

1. Save netlist and footprint files - Save the netlist and the footprints that are associated with it.
2. View selected footprint - View the selected footprint in 2D. See Sec. 7.1.4 for more details.
3. Automatic footprint association - Perform footprint association for each component automatically. Footprints will be selected from the list of footprints available.
4. Delete all associations - Delete all the footprint associations made
5. Display filtered footprint list - Display a filtered list of footprints suitable to the

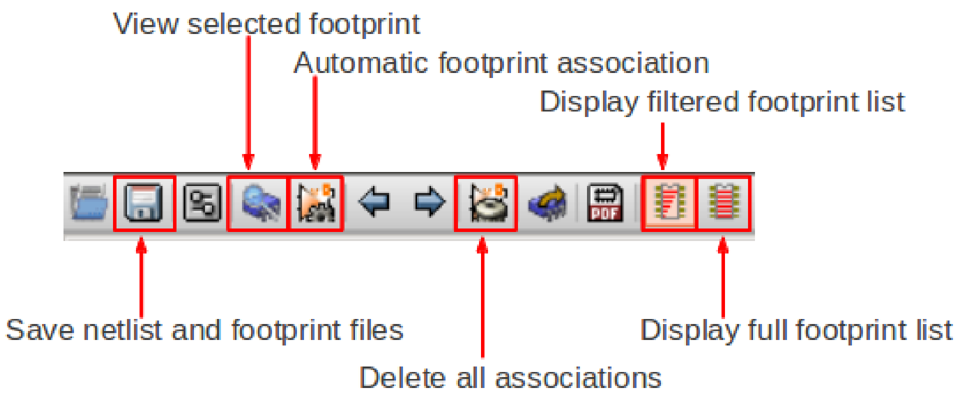

View selected footprint
Automatic footprint association
Display filtered footprint list
Save netlist and footprint files
Delete all associations
Display full footprint list

Figure 7.4: Some important tools in the toolbar

selected component

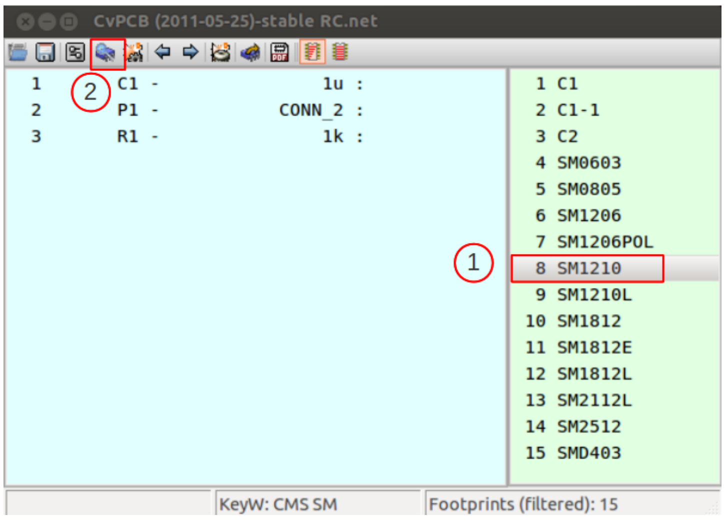
6. Display full footprint list - Display the list of all footprints available (without filtering)

7.1.4 Viewing footprints in 2D and 3D

To view a footprint in 2D, select it from the right pane and click on *View selected footprint* from the menu bar. Let us view the footprint for `SM1210`. Choose SM1210 from the right pane as shown in Fig. 7.5. On clicking the *View selected footprint* tool, the `Footprint` window with the view in 2D will be displayed. Click on the *3D* tool in the `Footprint` window, as shown in Fig. 7.6. A top view of the selected footprint in 3D is obtained. Click on the footprint and rotate it using mouse to get 3D views from various angles. One such side view of the footprint in 3D is shown in Fig. 7.7.

7.1.5 Mapping of components in the RC circuit

Click on `C1` from the left pane. Choose the footprint *C1* from the right pane by double clicking on it. Click on connector `P1` from the left pane. Choose the footprint *SIL-2* from the right pane by double clicking on it. Similarly choose the footprint *R3* for the resistor `R1`. The footprint mapping is shown in Fig. 7.8. Save the footprint association by clicking on the *Save netlist and footprint files* tool from the `CvPcb` toolbar. The `Save Net and component List` window appears. Browse to the directory where the schematic file for this project is saved and click on *Save*. The netlist gets saved and the *Footprint Editor* window closes automatically. *Note that one needs to browse to the directory where the schematic file is saved and save the '.net' file in the same directory.*



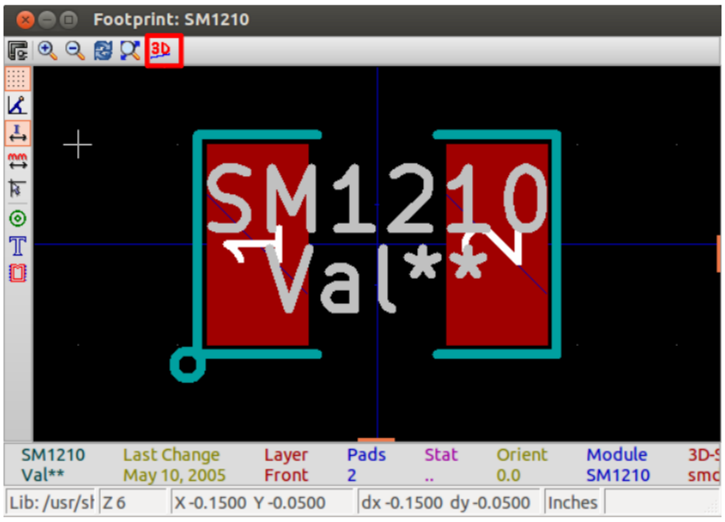
CvPCB (2011-05-25)-stable RC.net

```
1    C1 -          1u :
2    P1 -      CONN_2 :
3    R1 -          1k :
```

```
 1 C1
 2 C1-1
 3 C2
 4 SM0603
 5 SM0805
 6 SM1206
 7 SM1206POL
 8 SM1210
 9 SM1210L
10 SM1812
11 SM1812E
12 SM1812L
13 SM2112L
14 SM2512
15 SMD403
```

KeyW: CMS SM | Footprints (filtered): 15

Figure 7.5: Viewing footprint for SM1210: 1. Choose the footprint SM1210 from the right pane, 2. Click on *View selected footprint*



Footprint: SM1210

SM1210	Last Change	Layer	Pads	Stat	Orient	Module	3D-S
Val**	May 10, 2005	Front	2	..	0.0	SM1210	smc

Lib: /usr/sl | Z 6 | X -0.1500 Y -0.0500 | dx -0.1500 dy -0.0500 | Inches

Figure 7.6: Footprint view in 2D. Click on *3D* to get 3D view

7.2 Creation of PCB layout

The next step is to place the footprints and lay tracks between them to get the layout. This is done using the *Layout Editor* tool. Oscad uses `Pcbnew`, the layout creation tool in KiCad, as its layout editor.

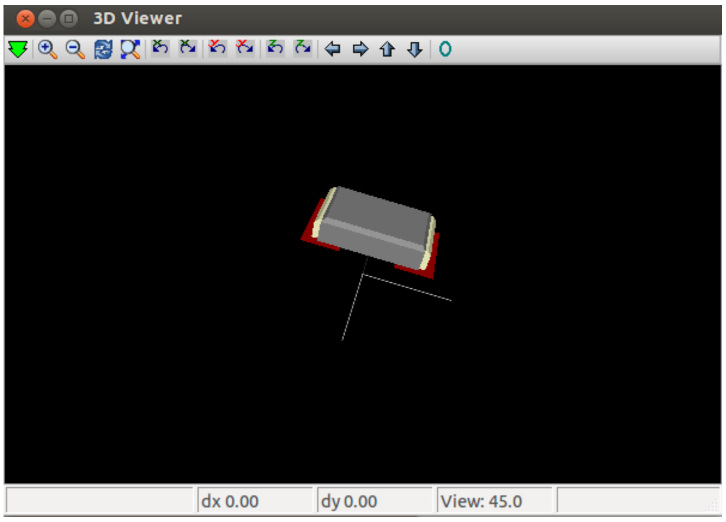
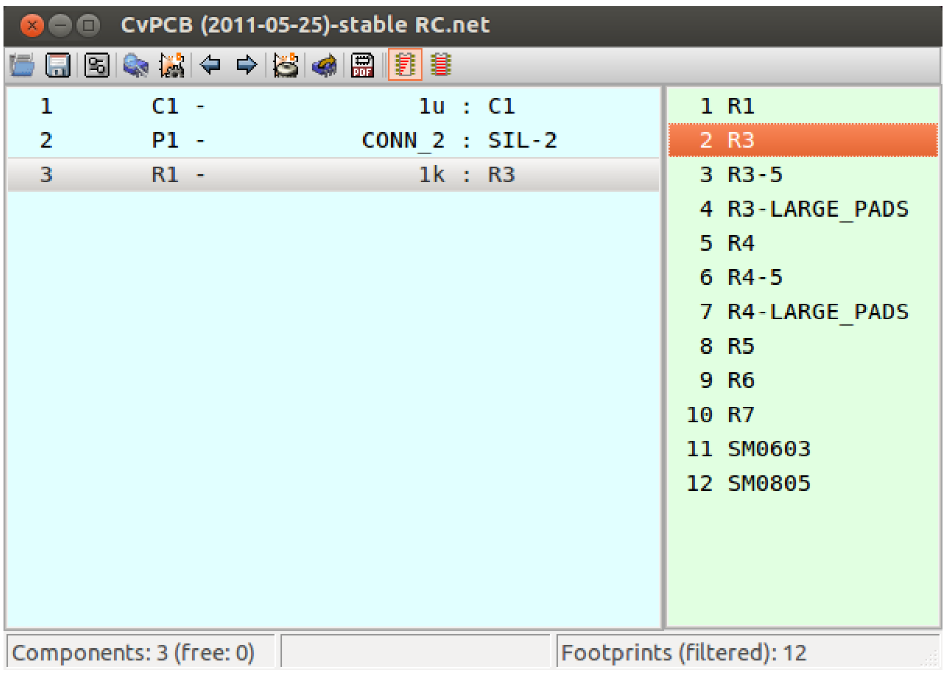


Figure 7.7: Side view of the footprint in 3D



```
1        C1  -          1u  : C1
2        P1  -      CONN_2  : SIL-2
3        R1  -          1k  : R3
```

```
 1 R1
 2 R3
 3 R3-5
 4 R3-LARGE_PADS
 5 R4
 6 R4-5
 7 R4-LARGE_PADS
 8 R5
 9 R6
10 R7
11 SM0603
12 SM0805
```

Components: 3 (free: 0)   Footprints (filtered): 12

Figure 7.8: Footprint mapping done

7.2.1 Familiarising the Layout Editor tool

The layout editor with the various menu bar and toolbars is shown in Fig. 7.9.

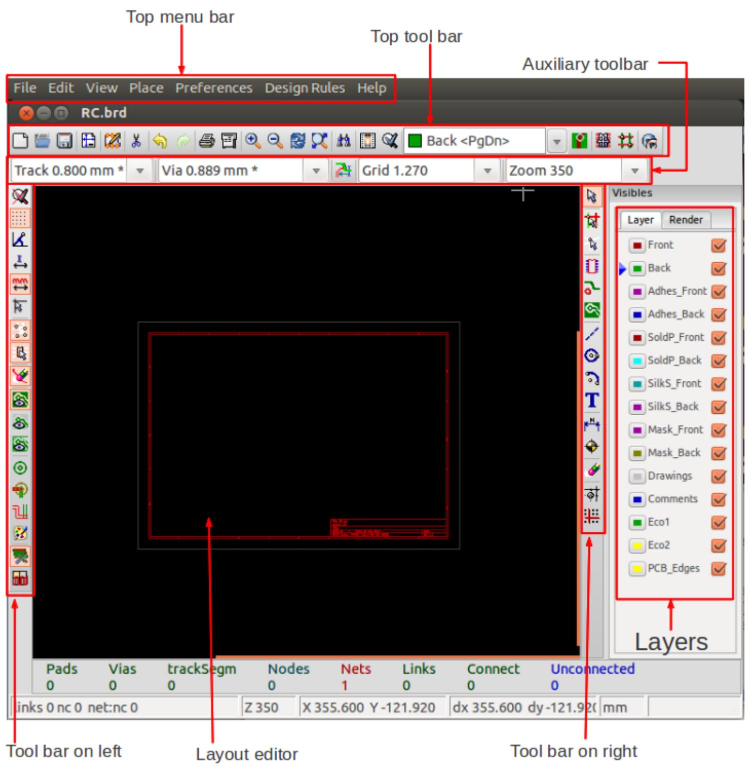


Figure 7.9: Layout editor with menu bar, toolbars and layer options marked

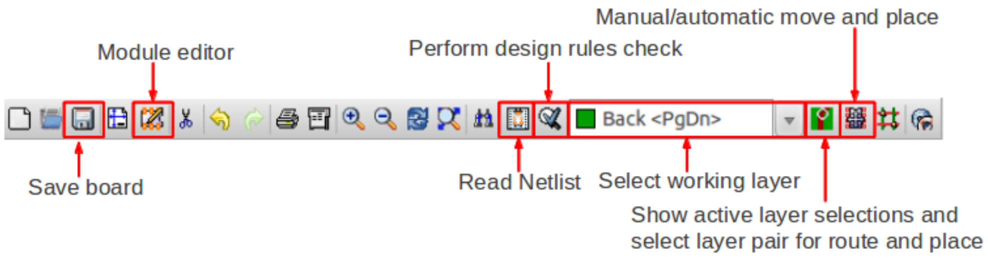

Manual/automatic move and place
Module editor
Perform design rules check
Back <PgDn>
Save board
Read Netlist
Select working layer
Show active layer selections and select layer pair for route and place

Figure 7.10: Top toolbar with important tools marked

Top toolbar

Some of the important menu options in the top menu bar are shown in Fig. 7.10. They are explained below:

1. Save board - Save the printed circuit board
2. Module editor - Open module editor to edit footprint modules or libraries
3. Read netlist - Import the netlist whose layout needs to be created.

4. Perform design rules check - Check for design rules, unconnected nets, etc., in the layout.
5. Select working layer - Selection of working layer
6. Show active layer selections and select layer pair for route and place - Select layer in top and bottom layers. It also shows the currently active layer selections.
7. Mode footprint: Manual/automatic move and place - Move and place modules

7.2.2 Hotkeys

A list of hotkeys are given below:

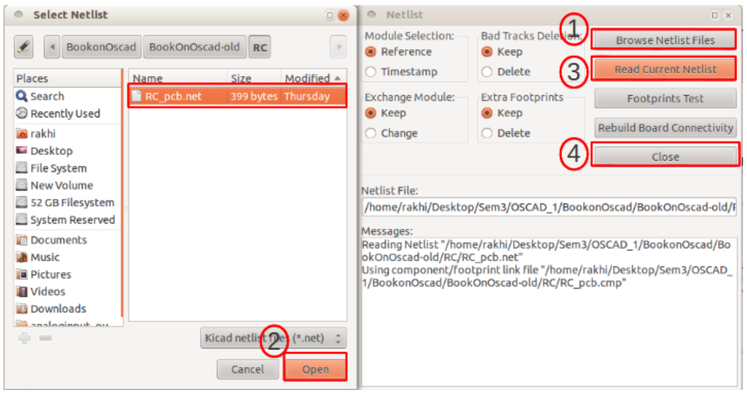
1. F1 - Zoom in
2. F2 - Zoom out
3. Delete - Delete Track or Footprint
4. X - Add new track
5. V - Add Via
6. M - Move Item
7. F - Flip Footprint
8. R - Rotate Item
9. G - Drag Footprint
10. Ctrl+Z - Undo
11. E - Edit Item

The list can be viewed by selecting *Preferences* from the top menu bar and choosing *List Current Keys* from the option *Hotkeys.*

7.2.3 PCB design example using RC circuit

Click on *Layout Editor* from the Oscad toolbar. Click on *Read Netlist* tool from the top toolbar. Click on *Browse Netlist files* on the Netlist window that opens up. Select the `.net` file that was modified after assigning footprints. Click on *Open.* Now Click on *Read Current Netlist* on the Netlist window. The message area in the Netlist window says that the RC_pcb.net has been read. The sequence of operations is shown in Fig. 7.11. The footprint modules will now be imported to the top left hand corner of the layout editor window. This is shown in Fig. 7.12. Zoom in to the top left corner by pressing the key `F1` or using the scroll button of the mouse. The zoomed in version of the imported netlist is shown in Fig. 7.13.

Let us now place this in the center of the layout editor window. Click on *Mode footprint: Manual/automatic move and place* tool from the top toolbar. Place the cursor near the center of the layout editor window. Right click and choose *Glob move and place.* Choose *move all modules.* The sequence of operations is shown in Fig. 7.14. Click on *Yes* on the confirmation window to move the modules. Zoom in using the F1 key. The current placement of components after zooming in is shown in Fig. 7.15a. We need to arrange the modules properly to lay tracks. Rotate the connector P1 by placing



Name	Size	Modified
RC_pcb.net	399 bytes	Thursday

Module Selection: [✓ Reference] Timestamp

Bad Tracks Deletion: [✓ Keep] Delete

Exchange Module: [✓ Keep] Change

Extra Footprints: [✓ Keep] Delete

Netlist File:
/home/rakhi/Desktop/Sem3/OSCAD_1/BookonOscad/BookOnOscad-old/I

Messages:

```
Reading Netlist "/home/rakhi/Desktop/Sem3/OSCAD_1/BookonOscad/Bo
okOnOscad-old/RC/RC_pcb.net"
Using component/footprint link file "/home/rakhi/Desktop/Sem3/OSCAD_
1/BookonOscad/BookOnOscad-old/RC/RC_pcb.cmp"
```

Figure 7.11: Importing netlist file to layout editor: 1. Browse netlist Files, 2. Choose the RC_pcb.net file, 3. Read Netlist file, 4. Close

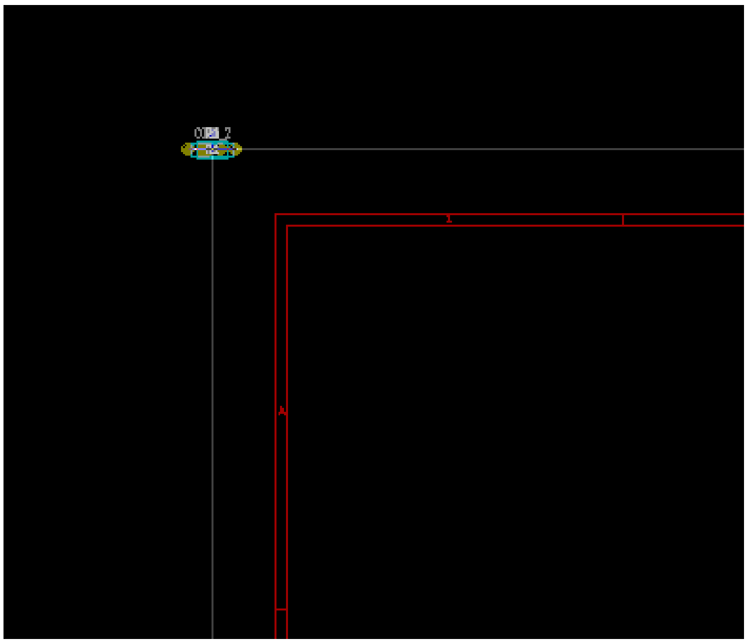


Figure 7.12: Footprint modules imported to top left corner of layout editor window

the cursor on top of P1 and pressing R. Move it by placing the cursor on top of it and pressing M. The final placement is shown in Fig. 7.15b.

Let us now lay the tracks. Let us first change the track width. Click on *Design rules* from the top menu bar. Click on *Design rules*. This is shown in Fig. 7.16. The *Design Rules Editor* window opens up. Here one can edit the various design rules. Double click on the track width field to edit it. Type 0.8 and press `Enter`. Click on OK. Fig. 7.17

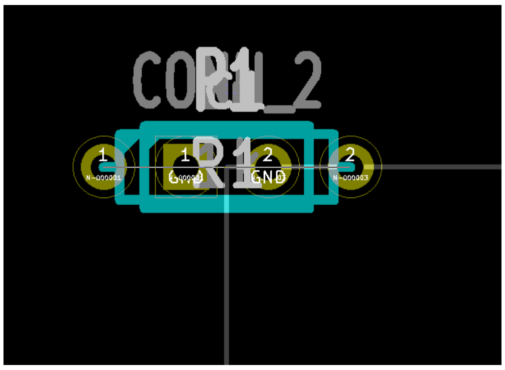


Figure 7.13: Zoomed in version of the imported netlist

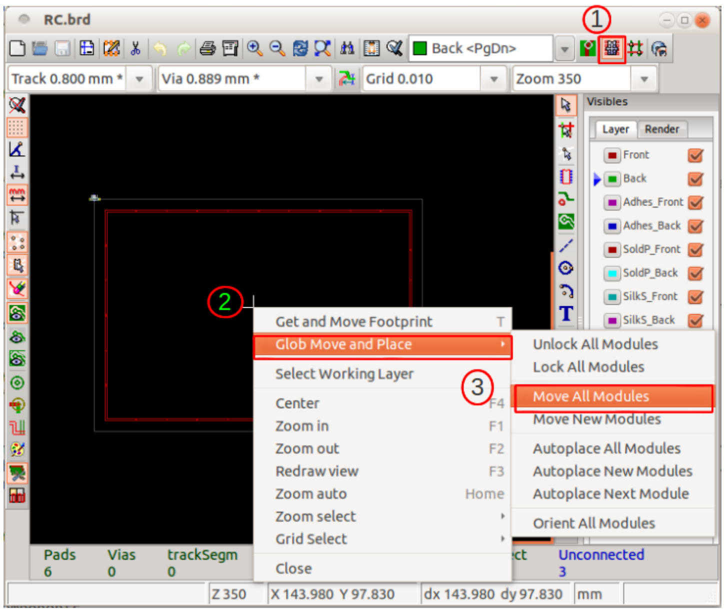
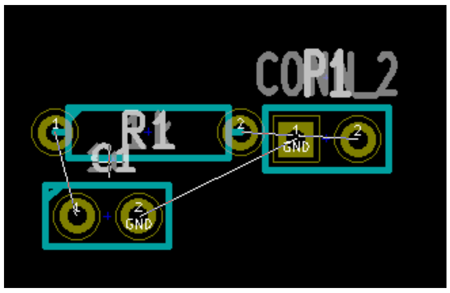


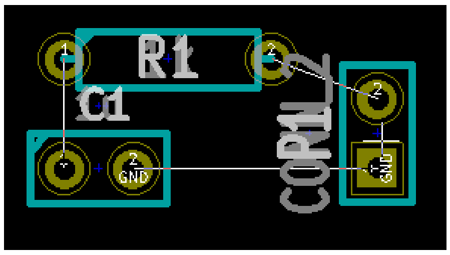
Figure 7.14: Moving and placing modules to the center of layout editor. 1. Click on *Mode footprint: Manual/automatic move and place*, 2. Place cursor at center of layout editor and right click on it 3. Choose *Glob Move and Place* and then choose *Move All Modules*.

shows the sequence of operations.

Click on *Back* from the *Layer* options as shown in Fig. 7.18. Let us now start laying the tracks. Place the cursor above the left terminal of R1 in the layout editor window. Press the key x. Move the cursor down and double click on the left terminal of C1. A track is formed. This is shown in Fig. 7.19a. Similarly lay the track between

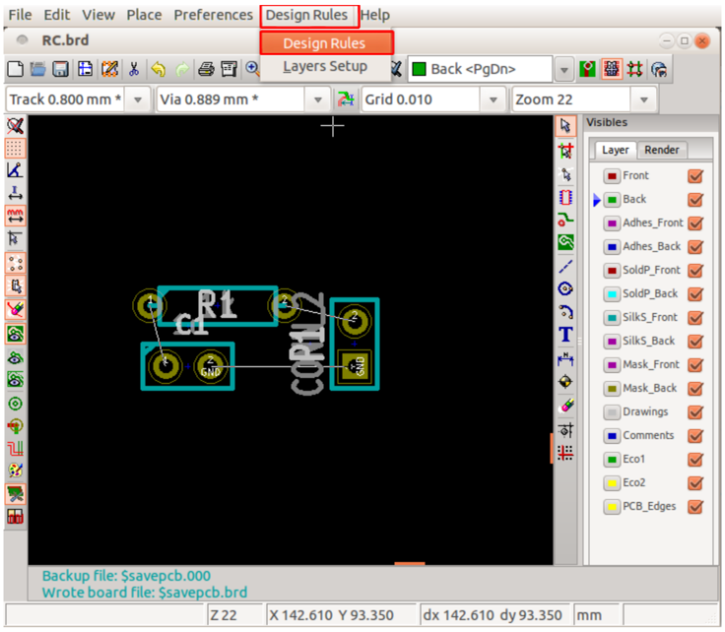


(a) Zoomed in version of the current placement after moving modules to the center of the layout editor



(b) Final placement of footprints after rotating and moving P1

Figure 7.15: Different stages of placement of modules on PCB

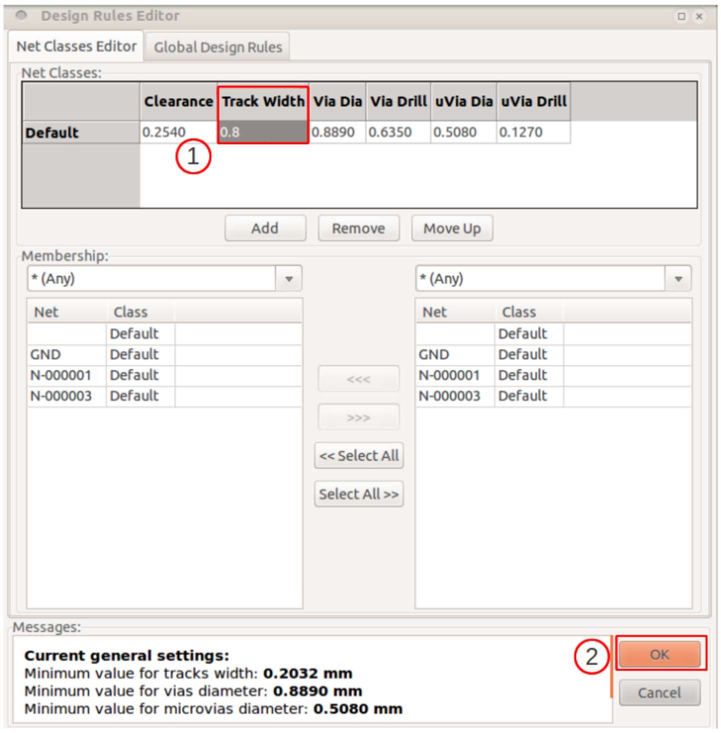


```
Backup file: $savepcb.000
Wrote board file: $savepcb.brd
```

Figure 7.16: Choose *Design Rules* from the top menu bar and *Design Rules* again

capacitor C1 and connector P1 as shown in Fig. 7.19b. The last track needs to be laid at an angle. To do so, place the cursor above the second terminal of R1. Press the key x and move the cursor diagonally down. Double click on the other terminal of the connector. The track will be laid as shown in Fig. 7.19c. All tracks are now laid. The next step is to create PCB edges.

Choose *PCB_edges* from the *Layer* options to add edges. Click on *Add graphic line or polygon* from the toolbar on the left. Fig. 7.20 shows the sequence of operations.



	Clearance	Track Width	Via Dia	Via Drill	uVia Dia	uVia Drill
Default	0.2540	0.8	0.8890	0.6350	0.5080	0.1270

Net	Class
	Default
GND	Default
N-000001	Default
N-000003	Default

Net	Class
	Default
GND	Default
N-000001	Default
N-000003	Default

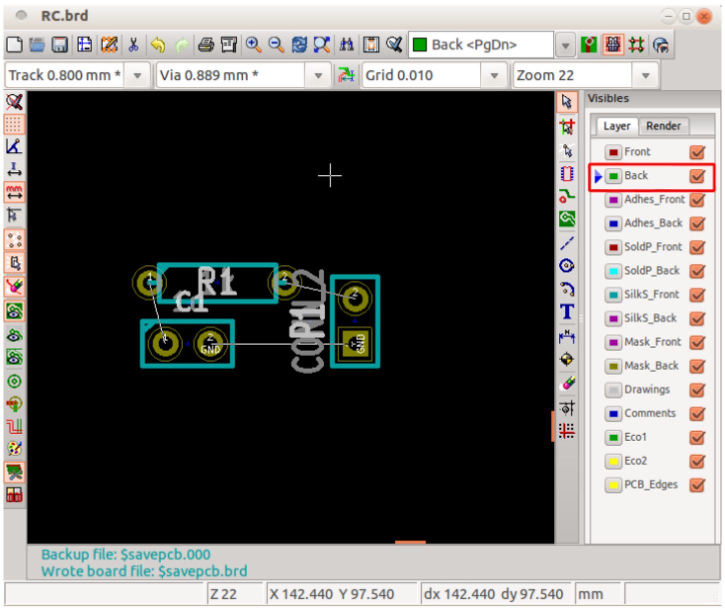
Current general settings:
Minimum value for tracks width: **0.2032 mm**
Minimum value for vias diameter: **0.8890 mm**
Minimum value for microvias diameter: **0.5080 mm**

Figure 7.17: Changing the track width: 1. Double click on *Track Width* field and type 0.8, 2. Click on *OK*

Let us now start drawing edges for PCB. Click to the left of the layout. Move cursor horizontally to the right. Click once to change orientation. Move cursor vertically down. Draw the edges as shown in Fig. 7.21. Double click to finish drawing the edges.

Click on *Perform design rules check* from the top toolbar to check for design rules. The *DRC Control* window opens up. Click on *Start DRC*. There are no errors under the `Error messages` tab. Click on *OK* to close DRC control window. Fig. 7.22 shows the sequence of operations. Click on *Save board* on the top toolbar.

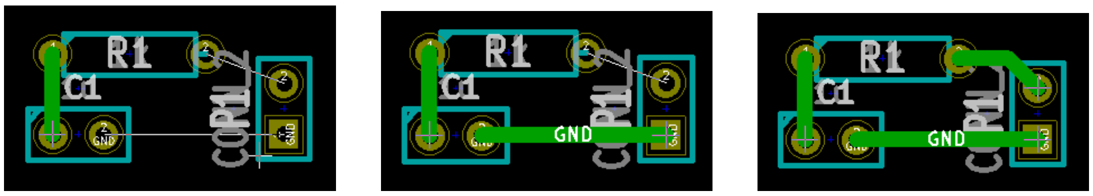
To generate Gerber files, click on *File* from the top menu bar. Click on *Plot*. This is shown in Fig. 7.23. The plot window opens up. One can choose which layers to plot by selecting/deselecting them from the `Layers` pane on the left side. One can also choose the format used to plot them. Choose *Gerber*. The output directory of the plots created can also be chosen. By default, it is the project directory. Some more options can be chosen in this window. Click on *Plot*. The message window shows the location in which the Gerber files are created. Click on *Close*. This is shown in Fig. 7.24. The PCB design of RC circuit is now complete. To know more about Pcbnew, refer to [15] or [16].



Backup file: $savepcb.000
Wrote board file: $savepcb.brd

Z 22 X 142.440 Y 97.540 dx 142.440 dy 97.540 mm

Figure 7.18: Choosing the copper layer *Back*

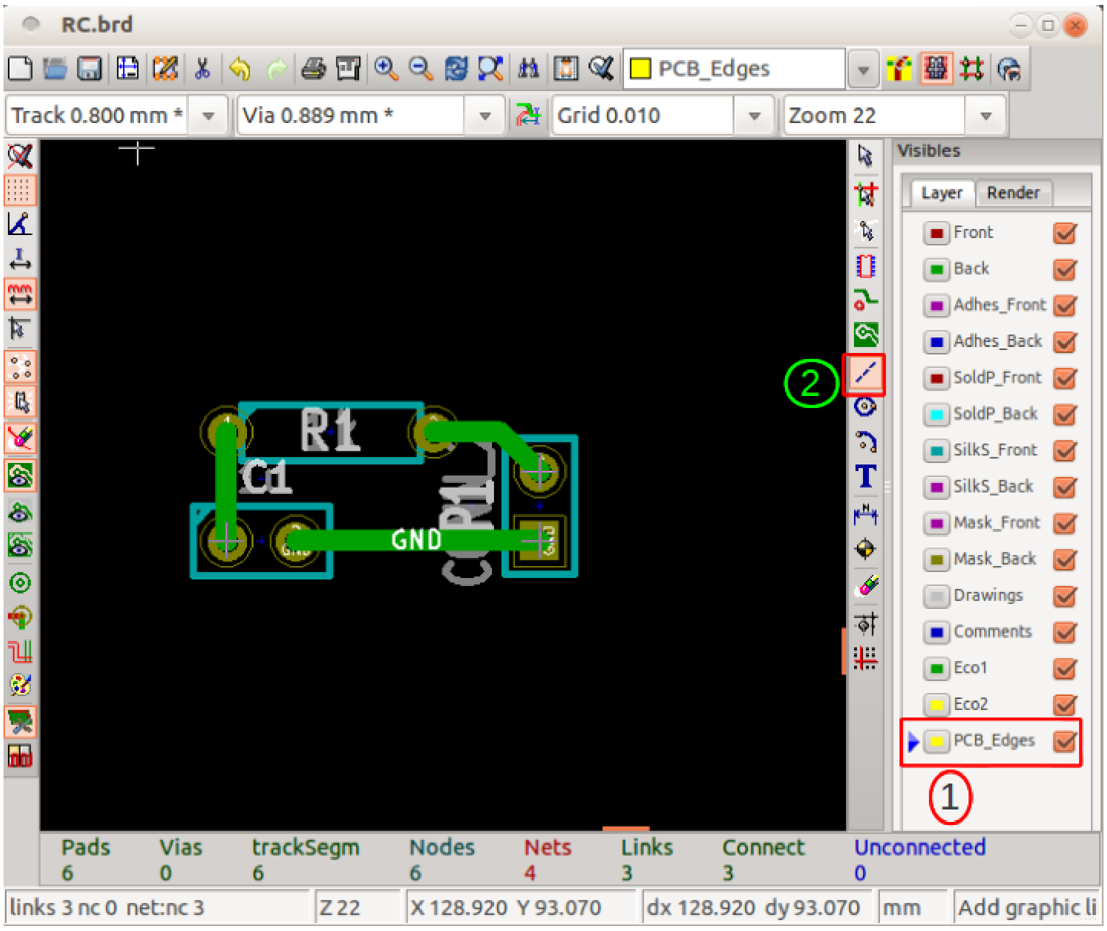


(a) A track formed between resistor and capacitor

(b) A track formed between capacitor and connector

(c) A track formed between connector and resistor

Figure 7.19: Different stages of laying tracks during PCB design



Pads 6 Vias 0 trackSegm 6 Nodes 6 Nets 4 Links 3 Connect 3 Unconnected 0

links 3 nc 0 net:nc 3 Z 22 X 128.920 Y 93.070 dx 128.920 dy 93.070 mm Add graphic li

Figure 7.20: Creating PCB edges: 1. Choose *PCB_Edges* from *Layer* options 2. Choose *Add graphic line or polygon* from left toolbar

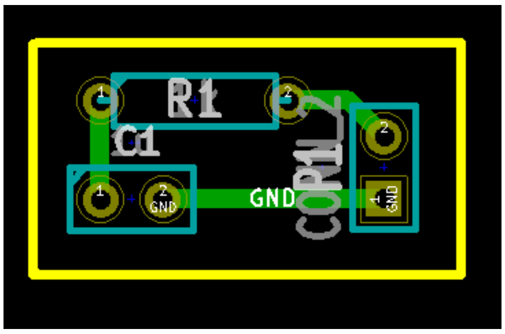
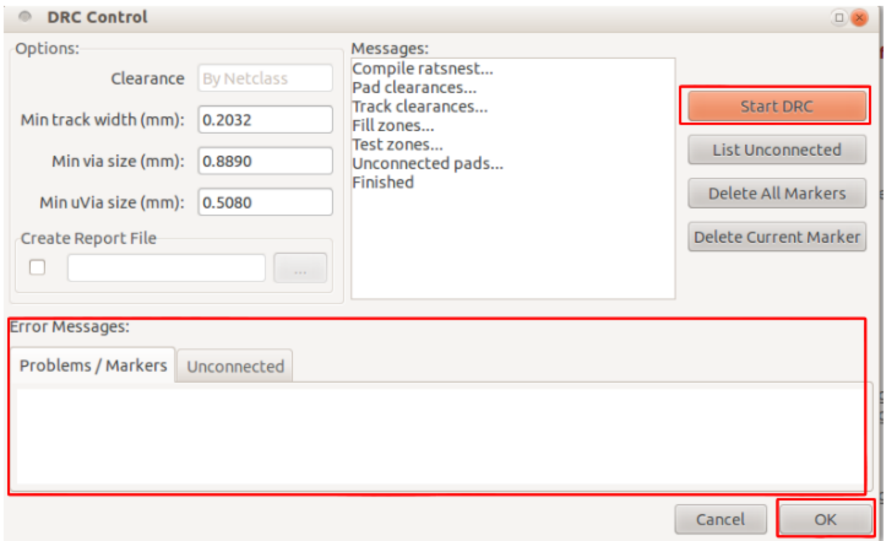


Figure 7.21: PCB edges drawn



Options:

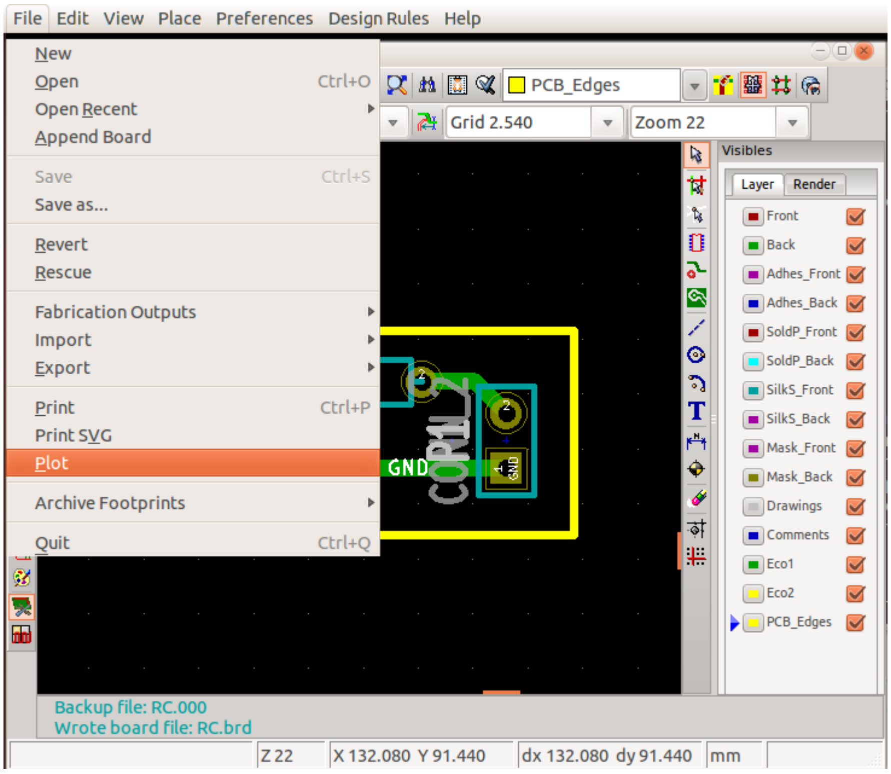
Clearance: By Netclass
Min track width (mm): 0.2032
Min via size (mm): 0.8890
Min uVia size (mm): 0.5080
Create Report File

Messages:

Compile ratsnest...
Pad clearances...
Track clearances...
Fill zones...
Test zones...
Unconnected pads...
Finished

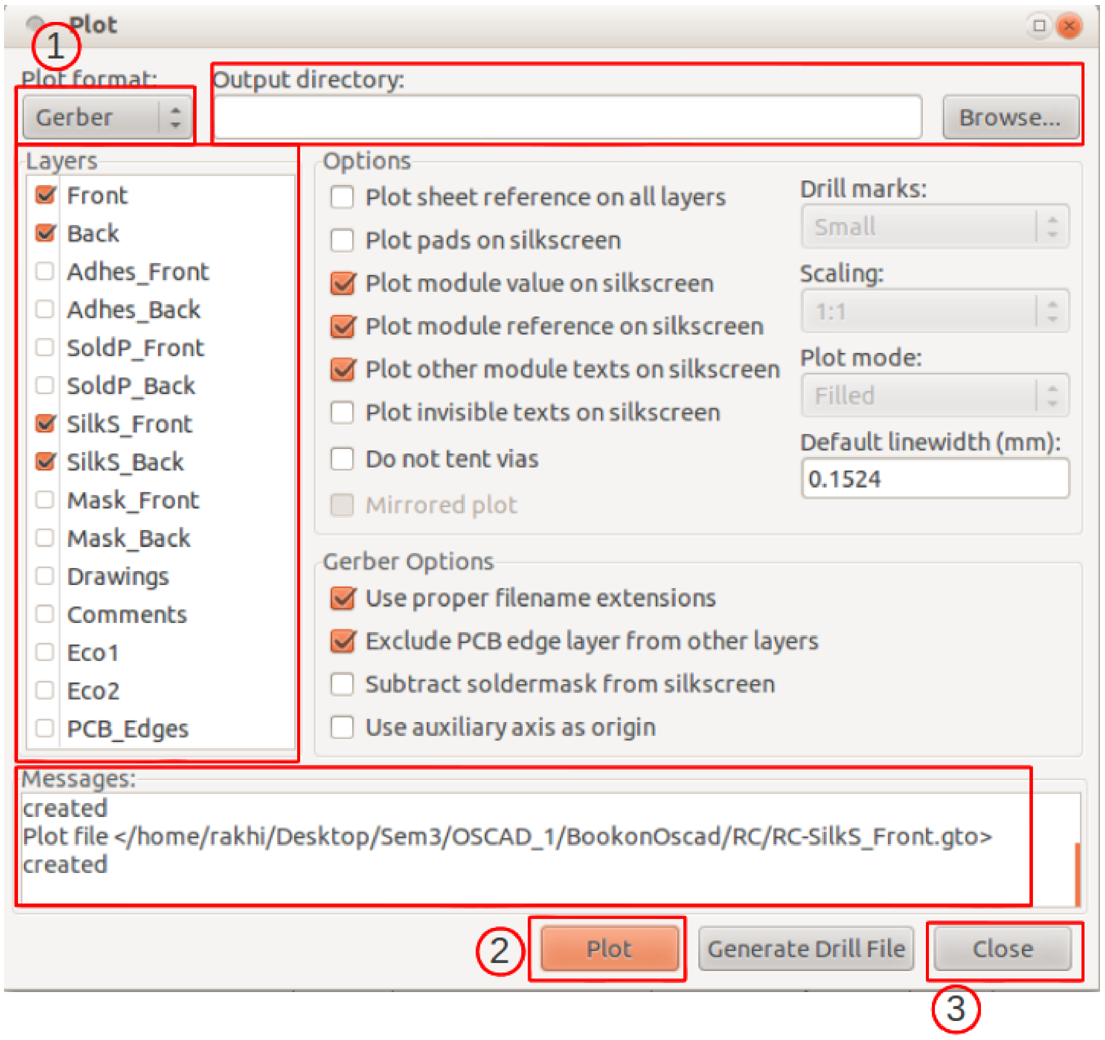
Error Messages:

Figure 7.22: Performing design rules check: 1. Click on *Start DRC*, 2. Click on *Ok*



Backup file: RC.000
Wrote board file: RC.brd

Figure 7.23: Choosing *Plot* from the *File* menu



Plot format: Gerber

Output directory:

Layers
- [✓] Front
- [✓] Back
- [ ] Adhes_Front
- [ ] Adhes_Back
- [ ] SoldP_Front
- [ ] SoldP_Back
- [✓] SilkS_Front
- [✓] SilkS_Back
- [ ] Mask_Front
- [ ] Mask_Back
- [ ] Drawings
- [ ] Comments
- [ ] Eco1
- [ ] Eco2
- [ ] PCB_Edges

Options
- [ ] Plot sheet reference on all layers
- [ ] Plot pads on silkscreen
- [✓] Plot module value on silkscreen
- [✓] Plot module reference on silkscreen
- [✓] Plot other module texts on silkscreen
- [ ] Plot invisible texts on silkscreen
- [ ] Do not tent vias
- [ ] Mirrored plot

Drill marks: Small

Scaling: 1:1

Plot mode: Filled

Default linewidth (mm): 0.1524

Gerber Options
- [✓] Use proper filename extensions
- [✓] Exclude PCB edge layer from other layers
- [ ] Subtract soldermask from silkscreen
- [ ] Use auxiliary axis as origin

Messages:
```
created
Plot file </home/rakhi/Desktop/Sem3/OSCAD_1/BookonOscad/RC/RC-SilkS_Front.gto>
created
```

Figure 7.24: Creating Gerber files: 1. Choose *Gerber* as the plot format, 2. Click on *Plot*. Message window shows location in which Gerber files are created, 3. Click on *Close*

Chapter 8

Model Builder and Subcircuit Builder

Model Builder and Subcircuit Builder have been created to expand the capabilities of Oscad. We will describe them in this chapter.

8.1 Model Builder

Spice based simulators include a feature which allows accurate modeling of semiconductor devices such as diodes and transistors. Oscad Model Builder provides a facility to define a new model for devices such as diodes, MOSFET, BJT, JFET, IGBT and Magnetic core. Model Builder in Oscad lets the user enter the values of parameters depending on the type of device for which a model is required. The parameter values can be obtained from the data sheet of the device. A newly created model can be exported to the model library and one can import it for different projects, whenever required. Model Builder also provides a facility to edit existing models.

8.1.1 Example

Let us take the example of the Bridge Rectifier circuit containing diode model `1N4007` and see how the model builder works in Oscad. First create the circuit schematic of the Bridge Rectifier shown in Fig. 8.1 using the *Schematic Editor* tool. Choose the component `diode` from the *device* library while creating the schematic. The `value` field of the diode components is `DIODE` by default. Now edit the `value` field of the diodes to the desired model name. Fig. 8.2 and Fig. 8.3 show how to do this. In Fig. 8.1, it has been changed to `1n4007`. Generate the spice netlist.

Now to build a new model for the diode 1n4007, click on *Model Builder* from the Oscad toolbar. It opens up *Model Select* window which shows 1n4007. Since we are going to create a new model, we will click on *cancel* button as shown in Fig. 8.4a. Then

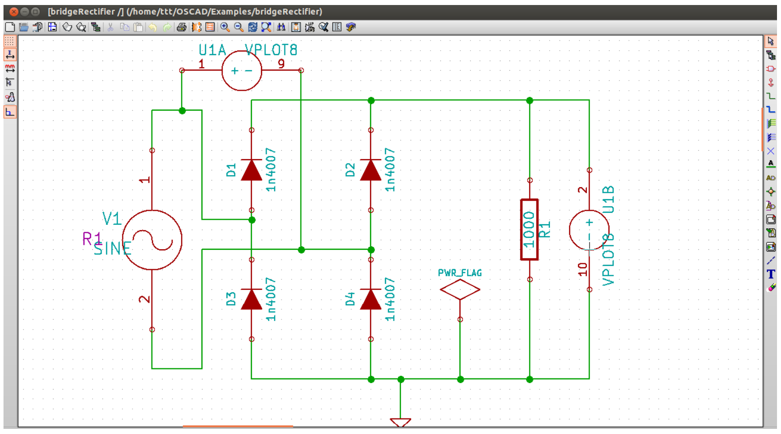
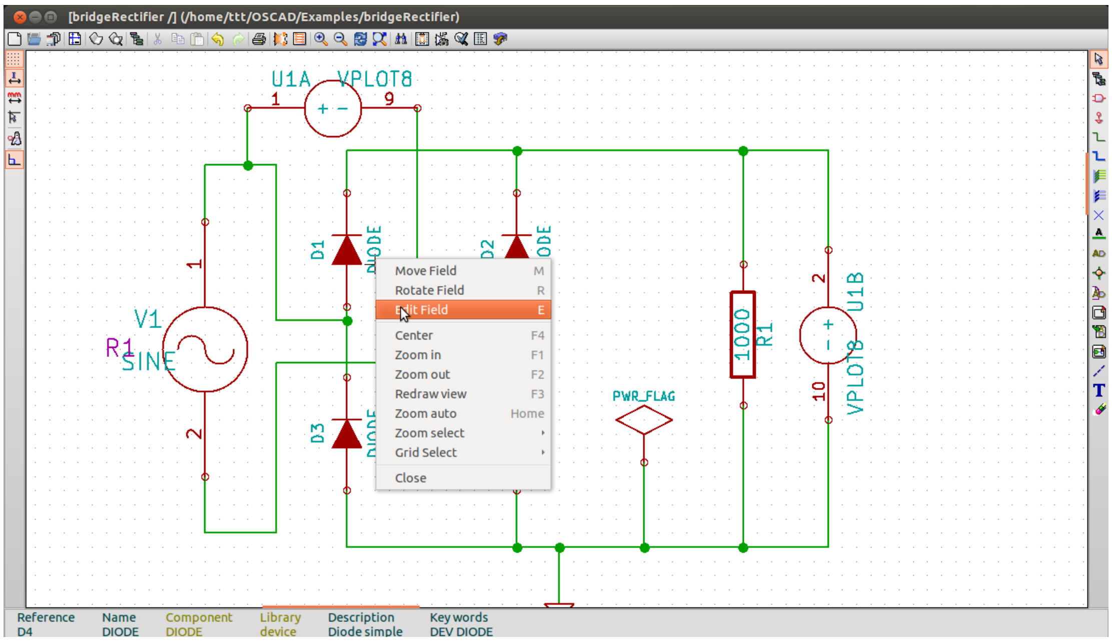


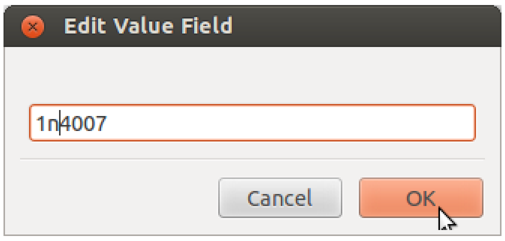
Figure 8.1: Bridge Rectifier circuit schematic



Reference	Name	Component	Library	Description	Key words
D4	DIODE	DIODE	device	Diode simple	DEV DIODE

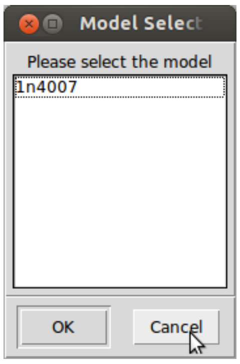
Figure 8.2: Right click and choose *Edit Field*

click on *New* from the *File* menu as shown in Fig. 8.4b. An *Ngspice Model Editor* window opens up. In the *Enter Component name* field, type `1N4007`. In the *Enter type of Component* option, select `Diode` and click on *OK*. This is illustrated in Fig. 8.4c. One needs to enter the values of model parameters in the window that appears now. This is shown in Fig. 8.5. These are nothing but spice parameters [14]. Some of them are explained in Table A.1 on Page 100. Add these parameters as given in the data sheet of



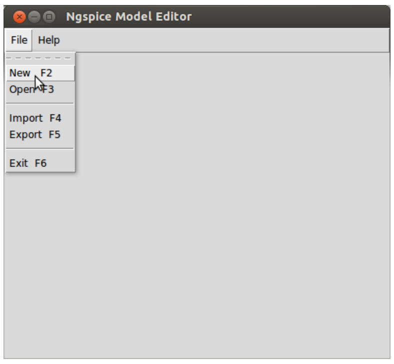
1n4007

Figure 8.3: Type 1n4007 in the `Edit Value Field` window

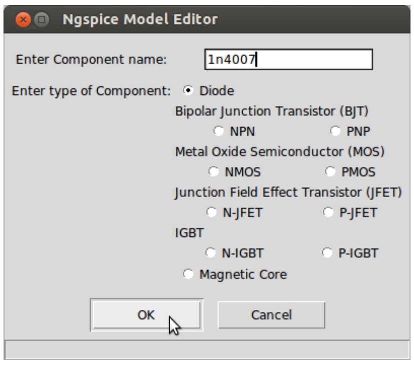


1n4007

(a) Cancel selecting existing model



(b) Select new model


Ngspice Model Editor
Bipolar Junction Transistor (BJT)
NPN
PNP
Metal Oxide Semiconductor (MOS)
NMOS
PMOS
Junction Field Effect Transistor (JFET)
N-JFET
P-JFET
IGBT
N-IGBT
P-IGBT
Magnetic Core
OK
Cancel

Enter Component name: 1n4007

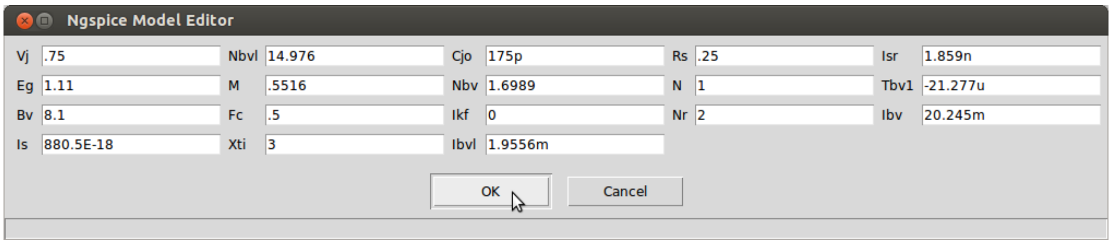
Enter type of Component: [✓ Diode]

(c) Select type of model

Figure 8.4: The procedure to select a model in Model Builder

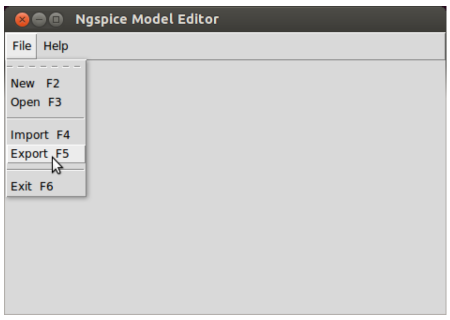
1N4007. Click on *OK* to save it. Click on *OK* on the Info dialog box that says *Model file 1N4007 is created*. Similarly, the model parameters for a BJT model are given in Table A.2 on Page 106.

Once a new diode model of 1N4007 is created it can be exported to the model library as shown in Fig. 8.6a. Whenever required, one can also import it for different projects as shown in Figures 8.6b and 8.6c.

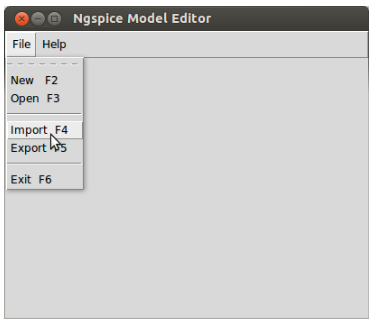


Vj	.75	Nbvl	14.976	Cjo	175p	Rs	.25	Isr	1.859n
Eg	1.11	M	.5516	Nbv	1.6989	N	1	Tbv1	-21.277u
Bv	8.1	Fc	.5	Ikf	0	Nr	2	Ibv	20.245m
Is	880.5E-18	Xti	3	Ibvl	1.9556m				

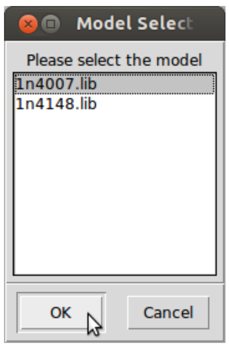
Figure 8.5: Editing model parameters



(a) Export model



(b) Import model-1



(c) Import model-2

Figure 8.6: Exporting and importing models in *Model Builder*

8.2 Subcircuit Builder

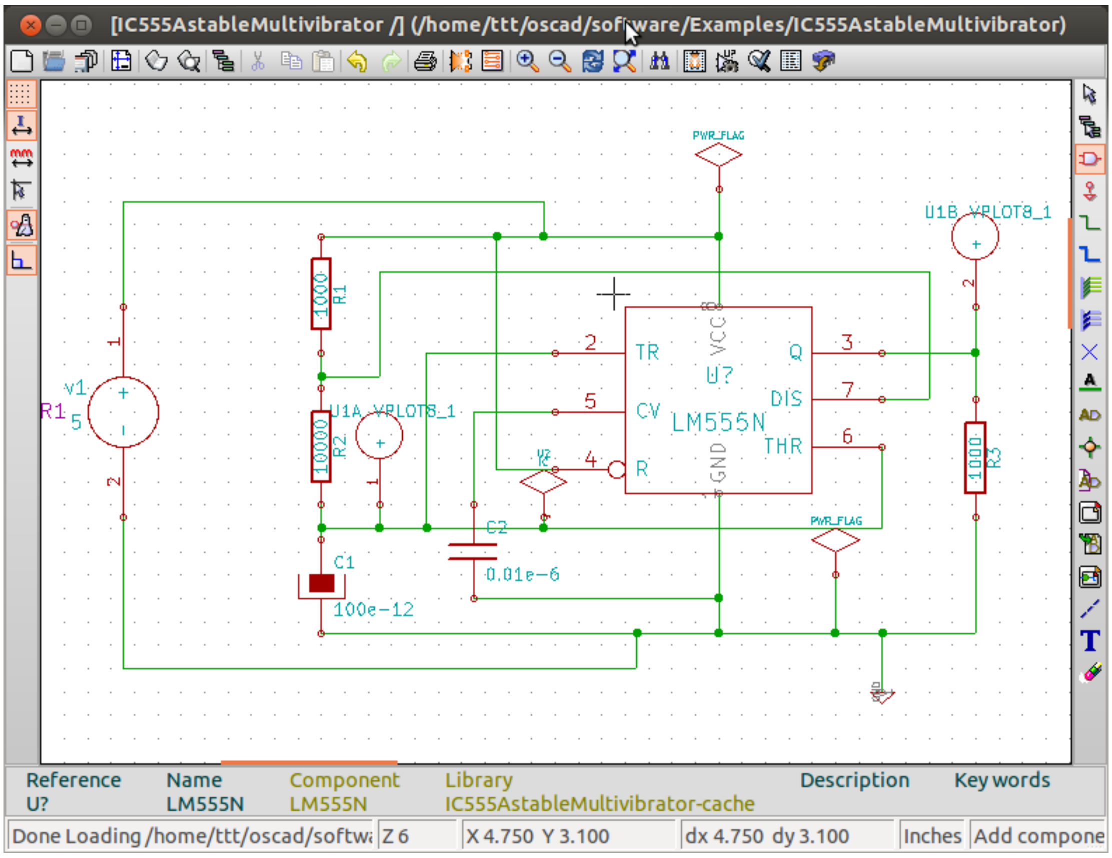
Subcircuit is a way to implement hierarchical modeling. Once a subcircuit for a component is created, it can be used in other circuits. Oscad provides an easy way to create a subcircuit.

8.2.1 Example

Let us take an example of building a subcircuit of the IC 555 timer which is a part of a Astable Multivibrator circuit. Create the schematic of the Astable Multivibrator as shown in Fig. 8.7. Before making the subcircuit for components, make sure that the *Reference Field* of these components should always be `X`. Let us see how to change *Reference Field* of a component. Right click on *U?* of the LM555N IC and choose *Field Reference*. This is shown in Fig. 8.8a. Then choose *Edit field* as shown in Fig. 8.8b. Change the reference field of LM555N from *U?* to `X`. Here the reference field `X` denotes that LM555N can have a subcircuit. This is shown in Fig. 8.9a.

Now to create subcircuit, click on the *Subcircuit Builder* tool from the Oscad toolbar. A window named *subcircuit select*, displaying the list of components whose *Reference Field* value is *X*, will open up. It is shown in Fig. 8.9b. Once the subcircuit is selected, it opens up the schematic editor window where one can draw the schematic of the subcircuit. After the subcircuit is created, connect a port to the external pins of the subcircuit. `Port` component can be added using the *Place a component* tool. The subcircuit of LM555N is shown in Fig. 8.10. Once the creation of subcircuit is complete, save it in the respective project folder and close the schematic editor window.

When the schematic editor window is closed, a terminal window will pop-up. It asks for the values of different parameters of subcircuit as shown in the Fig. 8.11. After entering all the parameters, the following message will be seen on the terminal:



[IC555AstableMultivibrator /] (/home/ttt/oscad/software/Examples/IC555AstableMultivibrator)

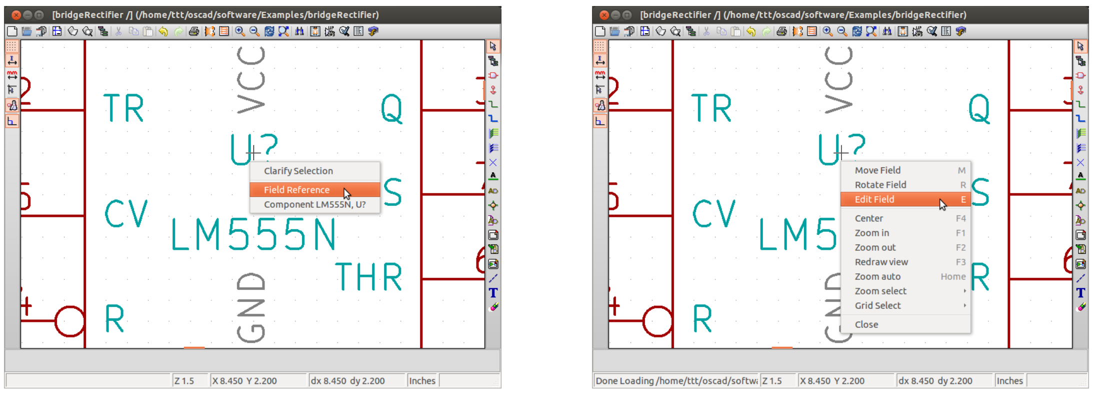
Reference	Name	Component	Library	Description	Key words
U?	LM555N	LM555N	IC555AstableMultivibrator-cache		

Done Loading /home/ttt/oscad/softwa | Z 6 | X 4.750 Y 3.100 | dx 4.750 dy 3.100 | Inches | Add compone

Figure 8.7: Astable Multivibrator circuit schematic

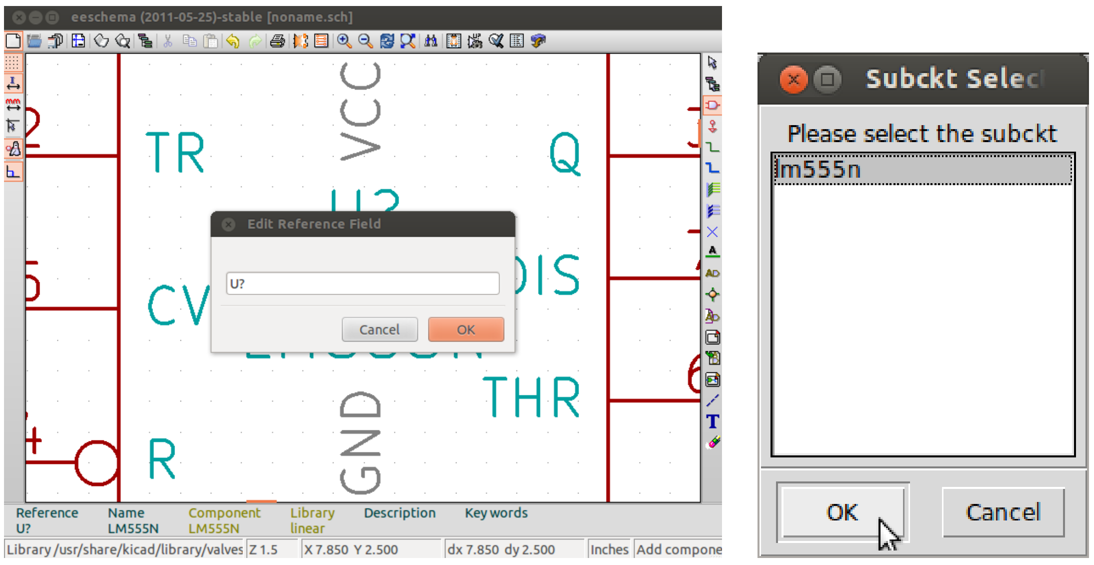
The ngspice netlist has been written in lm555n.cir.out
The scilab netlist has been written in lm555n.cir.ckt.
Press Enter to quit
This message means that netlist for the subcircuit is created. It is shown in the Fig. 8.12. Finally it opens up a window that says *Created subcircuit lm555n.sub*. Click on *OK*. This completes the subcircuit creation.

Once subcircuit is created, it can be exported to the subcircuit library as shown in Fig. 8.13a and whenever required one can import it to different projects as shown in Fig. 8.13b. Finally close the subcircuit editor window.



(a) Choosing *Field Reference* of LM555N (b) Choosing *Edit Field* of LM555N

Figure 8.8: Changing reference field of LM555N from U? to X



(a) Changing reference field of LM555N from U? to X (b) Selecting subcircuit

Figure 8.9: Different stages in subcircuit building

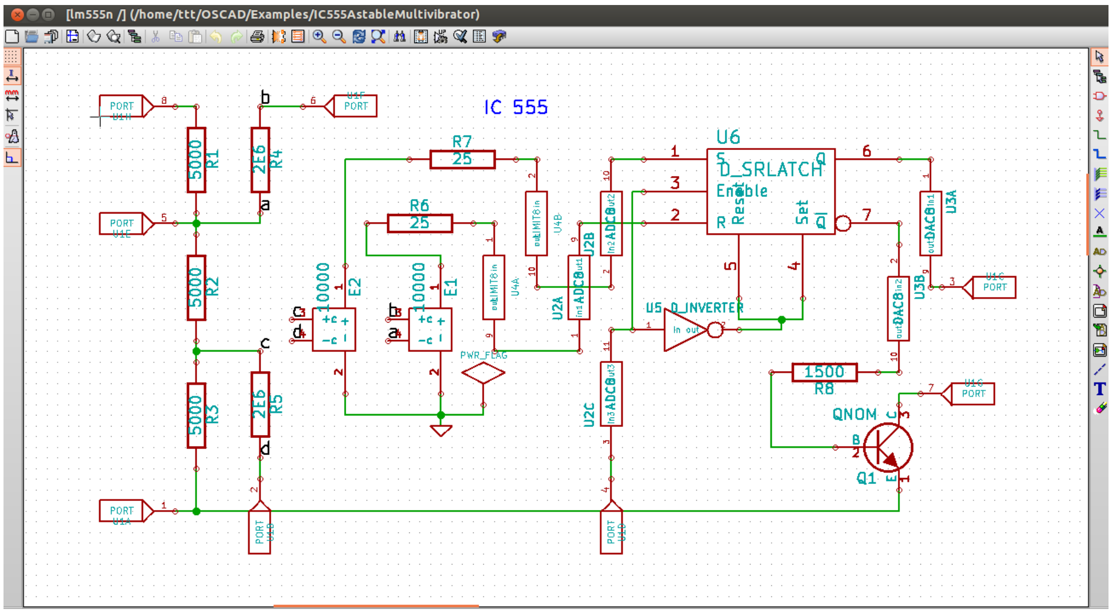
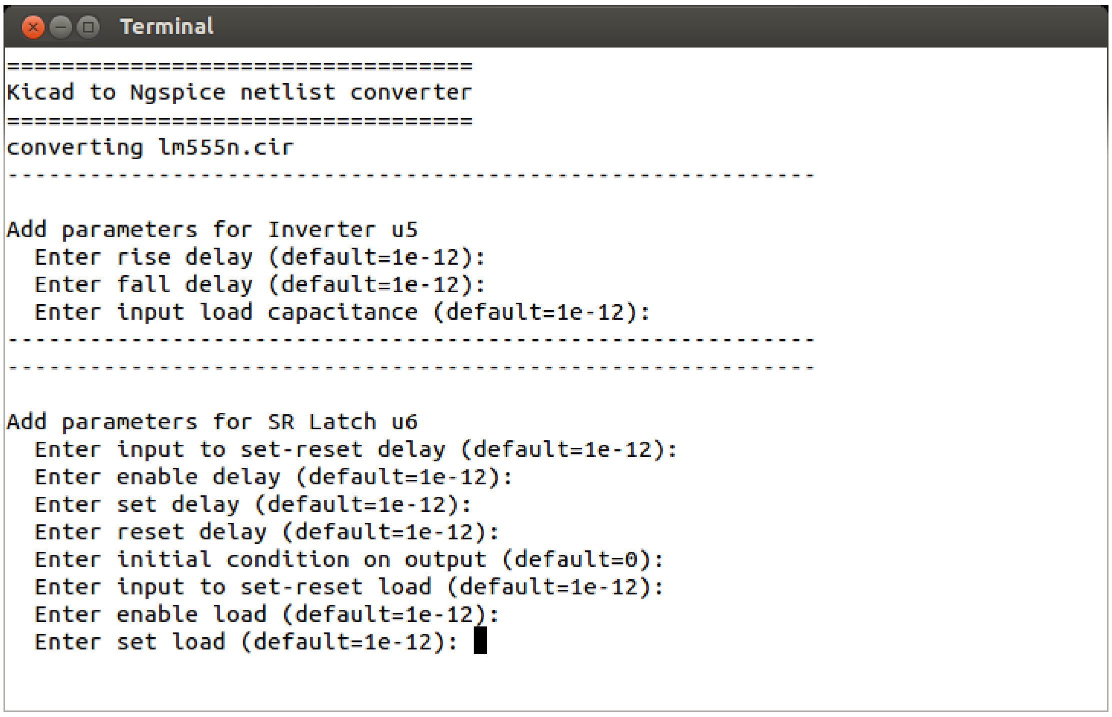
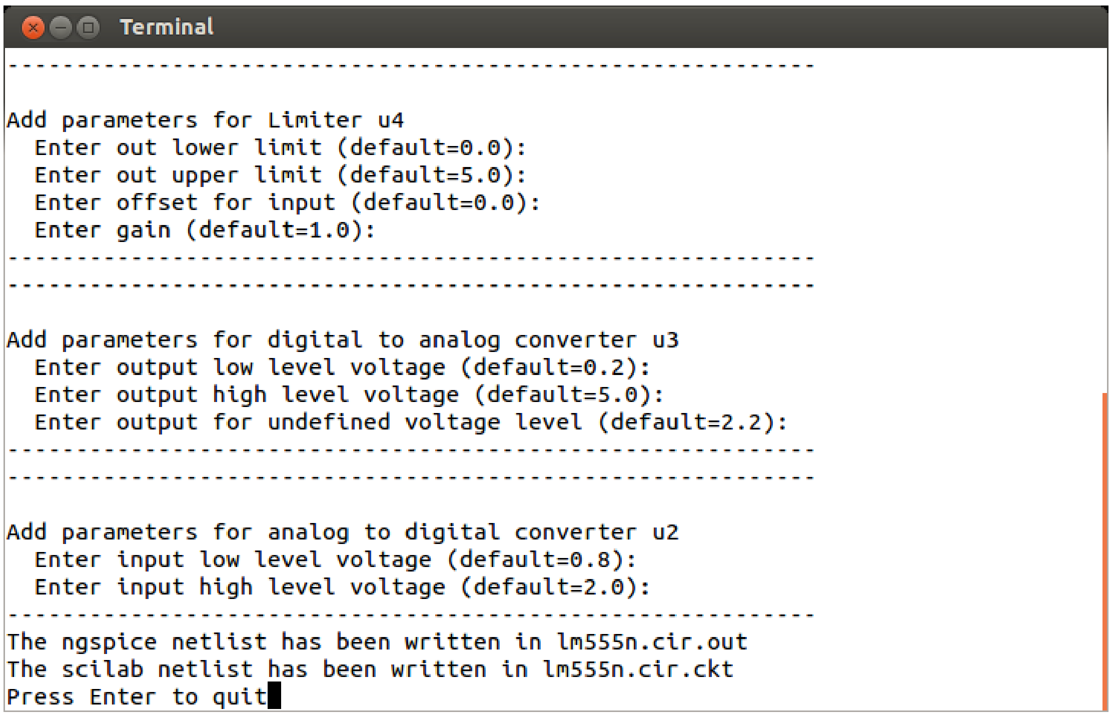


Figure 8.10: IC 555 timer subcircuit



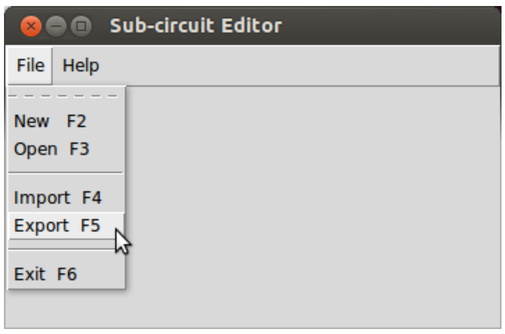
```
==================================
Kicad to Ngspice netlist converter
==================================
converting lm555n.cir
-----------------------------------------------------------

Add parameters for Inverter u5
  Enter rise delay (default=1e-12):
  Enter fall delay (default=1e-12):
  Enter input load capacitance (default=1e-12):
-----------------------------------------------------------
-----------------------------------------------------------

Add parameters for SR Latch u6
  Enter input to set-reset delay (default=1e-12):
  Enter enable delay (default=1e-12):
  Enter set delay (default=1e-12):
  Enter reset delay (default=1e-12):
  Enter initial condition on output (default=0):
  Enter input to set-reset load (default=1e-12):
  Enter enable load (default=1e-12):
  Enter set load (default=1e-12):
```

Figure 8.11: IC 555 timer subcircuit parameters

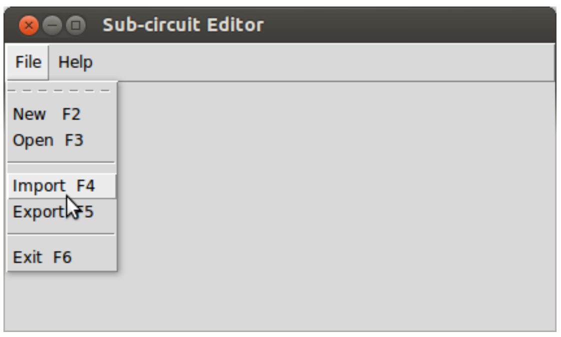


```
------------------------------------------------------------

Add parameters for Limiter u4
  Enter out lower limit (default=0.0):
  Enter out upper limit (default=5.0):
  Enter offset for input (default=0.0):
  Enter gain (default=1.0):
------------------------------------------------------------
------------------------------------------------------------

Add parameters for digital to analog converter u3
  Enter output low level voltage (default=0.2):
  Enter output high level voltage (default=5.0):
  Enter output for undefined voltage level (default=2.2):
------------------------------------------------------------
------------------------------------------------------------

Add parameters for analog to digital converter u2
  Enter input low level voltage (default=0.8):
  Enter input high level voltage (default=2.0):
------------------------------------------------------------
The ngspice netlist has been written in lm555n.cir.out
The scilab netlist has been written in lm555n.cir.ckt
Press Enter to quit
```

Figure 8.12: Netlist conversion for the 555 timer



(a) Export Subcircuit



(b) Import Subcircuit

Figure 8.13: Exporting and importing subcircuits

Chapter 9

Scilab Based Circuit Simulation

Electronic circuit simulation uses mathematical models to replicate the behaviour of an electronic circuit. Unfortunately, no simulator gives the system of equations it solves, in order to understand the simulation. 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. Currently this feature is available only for analog circuits.

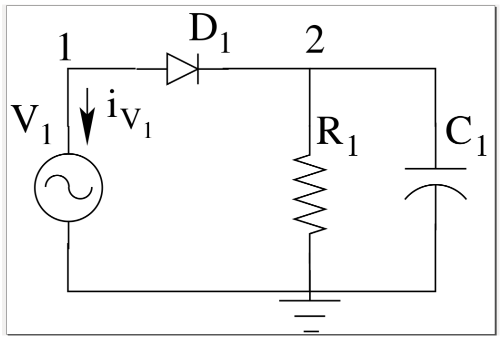
SMCSim works in three modes: normal, symbolic and numerical. In normal mode, SMCSim solves the circuit and gives the final output. In symbolic mode, it gives symbolic equations along with the result. In numerical mode, it gives symbolic equations, intermediate numerical values of the components and elements in system matrices, and the final output. Here, we present the working and implementation of SMCSim with an example.

Consider a Half-Wave Rectifier with filter shown in Fig. 9.2a. The circuit is drawn using the Oscad *Schematic Editor* and scilab compatible netlist (.cir.ckt) is generated using Oscad tools. The generated netlist is given in Fig. 9.1. Note that this netlist is generated for SMCSim which has a simple model implementation for a diode. Thus, users need to specify only I_s and V_t values.

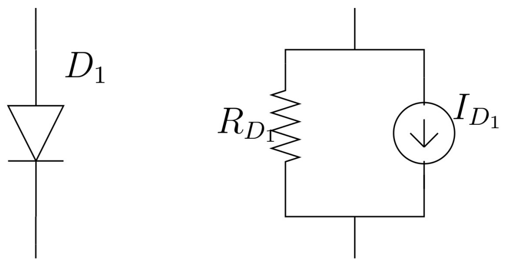
SMCSim first reads the netlist and creates a graph corresponding to it using Scilab based graph library `metanet`. Then circuit is translated into circuit equations using a circuit equation formulation method. All the methods of formulating circuit equations use Kirchhoff's current and voltage equations (KCE and KVE) and device characteristics constraints but differ in the manner in which these constraints are imposed. We have used Modified Nodal Analysis (MNA) as it is applicable to all kinds of electrical circuits. Using MNA method and graph operations, we have efficiently built the circuit equations. The system of equations representing the electrical circuit shown in Fig. 9.2a is given below. SMCSim has capability to display these equations.

```
* 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 9.1: Netlist of Half-Wave Rectifier with filter, used for Scilab simulations



(a) Half-Wave Rectifier with filter



(b) Diode model linearisation

Figure 9.2: Solving a Half-Wave Rectifier circuit with filter

$$i_{V_1} + D_{1f}(v_1, v_2) = 0 \tag{9.1}$$

$$(R_1)v_2 + (C_1)\frac{dv_2}{dt} + -D_{1f}(v_1, v_2) = 0 \tag{9.2}$$

$$v_1 = V_1 \tag{9.3}$$

$$D_{nf}(v_a, v_b) = Is_n(1 - e^{(v_a - v_b)/vt_n})$$

where Is_n=reverse saturation current and vt_n=threshold voltage of diode n. Now, we explain how SMCSim performs Operating point (DC) analysis and Transient analysis.

9.1 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. Since all waveforms are constant-valued at equilibrium points, $dv/dt = 0$ and $di/dt = 0$, capacitors act as open circuits and inductors act as short circuits. Thus, for operating point analysis, SMCSim removes the time-dependent components. The equations that describe the resulting system are nonlinear and algebraic and solution gives equilibrium point. The equations for the circuit are given below:

$$i_{V_1} + D_{1f}(v_1, v_2) = 0 \tag{9.4}$$
$$(R_1)v_2 + -D_{1f}(v_1, v_2) = 0 \tag{9.5}$$
$$v_1 = V_1 \tag{9.6}$$
$$D_{nf}(v_a, v_b) = Is_n(1 - e^{(v_a - v_b)/vt_n})$$

where Is_n=reverse saturation current and vt_n=threshold voltage of diode n.

Generally for nonlinear devices, a linear model is constructed that is valid only locally around a point. We have used Newton-Raphson method to construct linear models for nonlinear devices. In the example, diode D1 is a nonlinear device. SMCSim constructs the linear model for diode D1 as shown in the Fig. 9.2b. Note that the value of resistor and current source changes at every iteration and SMCSim allows user to observe the value. This is useful for debugging the circuit when the simulation is not converging. The system of equations representing the linearised electrical circuit is given below:

$$(R_{D_1})v_1 + (-R_{D_1})v_2 + i_{V_1} = -i_{D_1} \tag{9.7}$$
$$(R_{D_1})v_1 + (R_{D_1} + R_1)v_2 = i_{D_1} \tag{9.8}$$
$$v_1 = V_1 \tag{9.9}$$

9.2 Transient analysis

In transient analysis, time dependent components are discretised, 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 the example of Bridge Rectifier circuit shown in Fig. 9.3 and see how to do Scilab based circuit simulation. The steps to do Scilab based simulation are same as that of Ngspice simulation, except for the last step. Here, instead of clicking on the *Ngspice*, click on the *SMCSim* tool from the Oscad toolbar, as shown on Fig. 2.16a on page 11. The output of Scilab based simulations will be as shown in Fig. 9.4. Here one can see that Scilab based simulations not only gives the plot for the output but also the system of equations for the circuit. This is extremely helpful for students in improving their knowledge of circuit simulation.

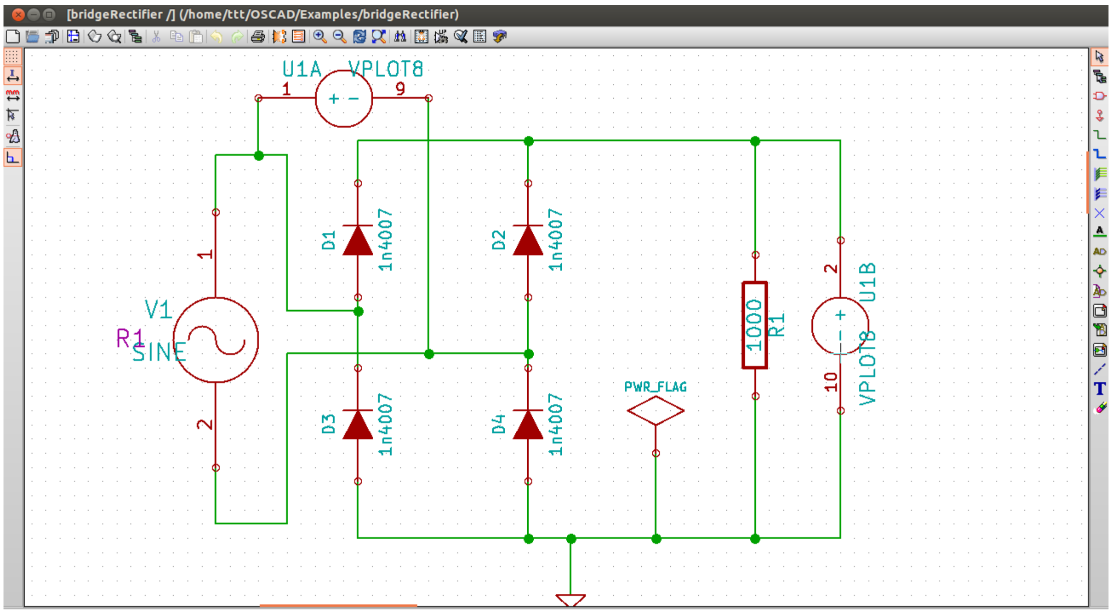
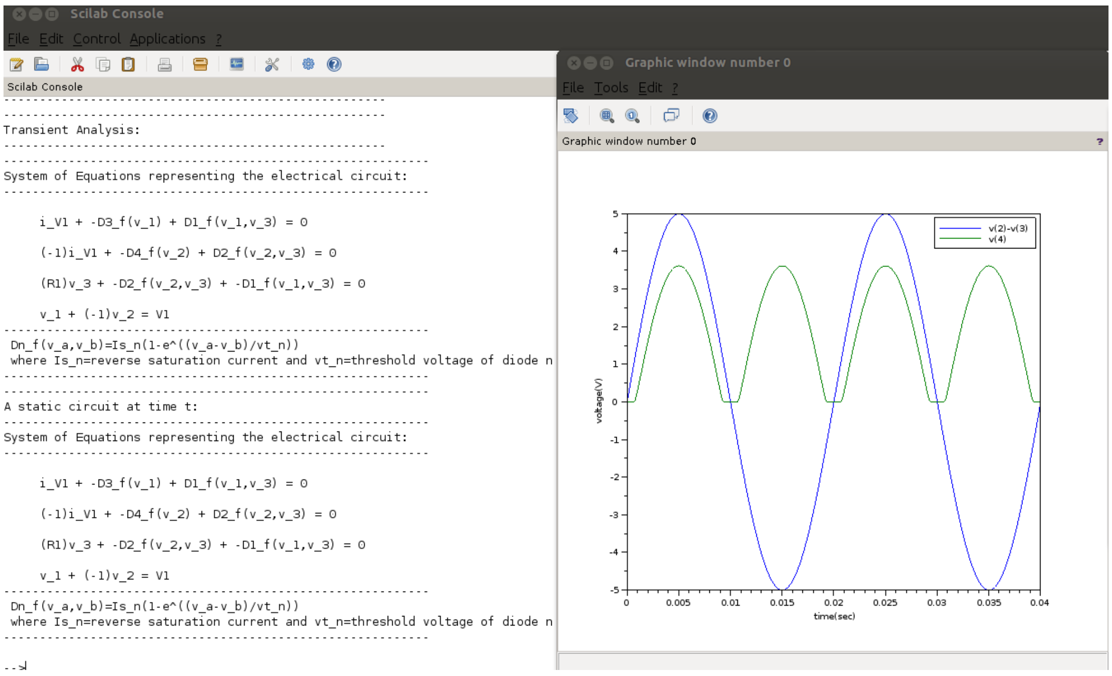

[bridgeRectifier /] (/home/ttt/OSCAD/Examples/bridgeRectifier)
U1A
VPLOT8
V1
R1
SINE
D1
D2
D3
D4
1n4007
1000
R1
PWR_FLAG
U1B
VPLOT8

Figure 9.3: Bridge rectifier circuit schematic



```
Scilab Console
-----------------------------------------------------
-----------------------------------------------------
Transient Analysis:
-----------------------------------------------------
-----------------------------------------------------------
System of Equations representing the electrical circuit:
-----------------------------------------------------------

     i_V1 + -D3_f(v_1) + D1_f(v_1,v_3) = 0

     (-1)i_V1 + -D4_f(v_2) + D2_f(v_2,v_3) = 0

     (R1)v_3 + -D2_f(v_2,v_3) + -D1_f(v_1,v_3) = 0

     v_1 + (-1)v_2 = V1
-----------------------------------------------------------
 Dn_f(v_a,v_b)=Is_n(1-e^((v_a-v_b)/vt_n))
 where Is_n=reverse saturation current and vt_n=threshold voltage of diode n
-----------------------------------------------------------
-----------------------------------------------------------
A static circuit at time t:
-----------------------------------------------------------
System of Equations representing the electrical circuit:
-----------------------------------------------------------

     i_V1 + -D3_f(v_1) + D1_f(v_1,v_3) = 0

     (-1)i_V1 + -D4_f(v_2) + D2_f(v_2,v_3) = 0

     (R1)v_3 + -D2_f(v_2,v_3) + -D1_f(v_1,v_3) = 0

     v_1 + (-1)v_2 = V1
-----------------------------------------------------------
 Dn_f(v_a,v_b)=Is_n(1-e^((v_a-v_b)/vt_n))
 where Is_n=reverse saturation current and vt_n=threshold voltage of diode n
-----------------------------------------------------------

-->
```

Figure 9.4: Scilab based simulation output

Chapter 10

Oscad Spoken Tutorials

Spoken tutorial is an instructional methodology based on screen-cast technology, meant to teach and learn IT systems [2], created for self learning. Volunteers are sufficient to conduct workshops using Spoken Tutorials [18] - domain experts are not required. A Spoken Tutorial is typically ten minutes long.

Skill based topics can be taught easily through Spoken Tutorials. IT and simulation topics are especially suited to be taught through Spoken Tutorials. This approach is particularly conducive to provide training on Oscad. We conduct Spoken Tutorial based SELF workshops, explained in detail in Sec. B.1, on many FOSS families. We currently offer SELF workshops on C, C++, Java, PHP/MySQL, Python, Scilab, LaTeX, Oscad, OpenFOAM, Blender and GIMP and GeoGebra [19]. We also conduct these workshops on topics of IT literacy, namely LibreOffice, Firefox, GNU/Linux and Ubuntu. As the spoken part of these tutorials are dubbed into all Indian languages, these are useful also to the students who are weak in English. We also conduct on-line tests on most of these topics and offer certificates.

In this chapter, we explain the spoken tutorials available on Oscad, classified into beginner and advanced levels. These tutorials are available at [19] and [20]. Appendix B explains a procedure to conduct Oscad SELF workshops using Spoken Tutorials.

10.1 Beginner level tutorials

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.

10.1.1 Introduction to Oscad

This tutorial aims to help the users get started with Oscad. It covers the following:

- This tutorial teaches how to install and verify Oscad

- The process of downloading Oscad, Oscad Examples and Scilab is shown
- The installation of Oscad is done through a shell script
- The script installs all the requisite software, Ngspice, KiCad, Scilab and Python
- As software is to be downloaded during installation, proxy settings are explained
- Installation of Oscad is verified by opening a pre-existing RC circuit schematic
- Ngspice simulation is carried out for this RC circuit and the resulting plots are displayed

10.1.2 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.

- Required components are chosen from their corresponding libraries
- Required components are placed in the schematic editor
- Components are connected together to form an RC circuit
- Components are then annotated and appropriate values are assigned to them
- Electric Rules Check is done on the created RC circuit
- Erroneous connections are identified, the method to overcome them explained and verified through Electric Rule Check
- Spice netist is generated
- Transient simulation parameters are added to the netlist using Analysis Inserter
- The netlist is converted to Ngspice format using Netlist Converter tool
- Circuit simulation is done using Ngspice and the results displayed

10.1.3 Designing circuit schematic in KiCad

- This tutorial introduces KiCad.
- It teaches how to create a circuit schematic using EEschema
- It explains this using an Astable Multivibrator circuit
- Required components are placed on the schematic
- Components are connected together to form an Astable Multivibrator.
- Components are then annotated and appropriate values are assigned to them.
- Electric Rules Check is done on the created circuit
- Erroneous connections are identified.
- A solution to correct the error is illustrated
- Electric Rules Check is again done on the corrected RC circuit to depict that the error has been corrected

10.1.4 Designing printed circuit board using Oscad

- This tutorial uses the RC circuit created in the tutorial `Schematic creation and simulation using Oscad`

- Sine source and plot components are deleted.
- A two pin connector is added.
- Rewiring of the circuit is done.
- Annotation is carried out on the circuit.
- Netlist for PCB is generated.
- Mapping of footprints to components is done using Footprint Editor.
- 3D view of the footprint is shown.
- Netlist is imported in the Layout Editor.
- PCB layout of RC is created. It explains how to choose design rules, lay tracks, choose layers, etc.
- Gerber files are generated for the PCB.

10.1.5 Electric rules checking and netlist generation in KiCad

- This tutorial uses an Astable Multivibrator circuit schematic created in the tutorial `Designing Circuit Schematic in KiCad`
- Values are assigned to components
- Electric Rules Check is done on the created circuit
- Erroneous connections are identified.
- A solution to correct the error is illustrated
- Electric Rules Check is again done on the corrected RC circuit to depict that the error has been corrected
- Netlist for designing PCB layout is generated

10.1.6 Mapping components in KiCad

- This tutorial explains how to map components in a schematic with corresponding footprints.
- It uses an Astable Multivibrator circuit schematic created in the tutorial `Designing Circuit Schematic in KiCad`
- CvPcb, the footprint editor in KiCad, is used.
- Every component in the Astable Multivibrator circuit schematic is assigned a footprint
- Netlist, along with the footprint mapping for PCB, is generated.
- This netlist is saved as a .net file.

10.1.7 Designing PCB in KiCad

- In this tutorial, printed circuit board layout of the Astable Multivibrator circuit is created.
- It uses the netlist created in the tutorial `Mapping components in KiCad`
- Netlist is imported in the Pcbnew window

- It explains how to lay the tracks and interconnect the footprints.
- It explains how to modify the width of the tracks, etc.
- It explains how to adjust the grid size.
- It also explains how to draw the PCB edges.
- Layer selection and track routing is also explained.
- The file is saved as a .brd file

10.2 Advanced level tutorials

Advanced level has tutorials on Ngspice, model building 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.

1. Operating point analysis in Ngspice. This tutorial explains:
    - How to perform operating point analysis.
    - How to verify Kirchoff's voltage law using Ngspice in, interactive mode using command line interface & using command script included in netlist.

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.

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 a Bridge Rectifier
    - Building/editing the `1N4007` diode model present in the Bridge Rectifier circuit using Model Builder tool in Oscad.
    - This is explained using a Bridge Rectifier circuit which contains 1N4007 diode.

4. Subcircuit creation using Oscad. We show how to create and edit a subcircuit. This is explained using the Astable Multivibrator circuit that has a 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 the 555 timer is 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.

10.3 Instruction sheet

As they are created for self learning, one can use a set of spoken tutorials to conduct workshops, even without domain experts. The workshops are conducted by volunteers using a clearly articulated and a regimented process [18]. These are known as SELF workshops. The instruction sheet to be used in an Oscad SELF workshop is given below. One needs the Spoken Tutorials explained in the previous section to practise with this instruction sheet.

10.3.1 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.

10.3.2 Please ensure that

- 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.

10.3.3 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.

10.3.4 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 (E.g. `vin-04-Oscad`).
  - Give a unique name to the files you save, so as to recognise it next time (E.g. `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`.

10.3.5 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 the Astable Multivibrator shown in it, using Oscad.

Chapter 11

Oscad on Aakash

Aakash provides a great platform for learning and education. We have ported GNU/Linux on to Aakash. As a result, most GNU/Linux applications run on Aakash. As KiCad, Ngspice, Scilab and Python run on GNU/Linux, we have not had any difficulty in porting Oscad also on to Aakash, as the latter is based on the former four.

Installation of Oscad in Aakash is similar to its GNU/Linux desktop version. One has to Follow the same instructions mentioned in Chapter 2. Oscad installer runs the `Synaptic package manager` underneath. This package manager is smart enough to detect the CPU architecture. It downloads and installs packages specific to the machine architecture. The installer is being further enhanced so that it could download the required version of Scilab for specific machine architecture from the Internet with minimum user intervention.

The Aakash tablet is connected to the Internet through *WiFi* before starting the installation procedure. Although Oscad is currently in development stage on Aakash, it is usable, nevertheless. Oscad will soon come pre-installed on Aakash. The design framework of Oscad has made it extremely simple to run it on Aakash with minimum modifications. Oscad was initially developed on an Ubuntu distribution which has GNOME-terminal primarily. This GNOME-terminal was called whenever there was a need for user input using terminal. On Aakash, LX terminal is used.

With Oscad running on Aakash, it proves Aakash's capability to run electronic design automation tools. It also shows Oscad's portability from desktops to hand held devices. The benefit of using Oscad on Aakash lies in its portability and performance. Although a capacitive touch screen is available on Aakash, one may attach an external keyboard and a mouse. One would find the same interface of Oscad on Aakash as in the Desktop version. Some times keyboard shortcuts are easier to use than the touch controls when running design and simulation tools. Figures 11.1 and Fig. 11.2a show a few screenshots of Oscad running on Aakash.

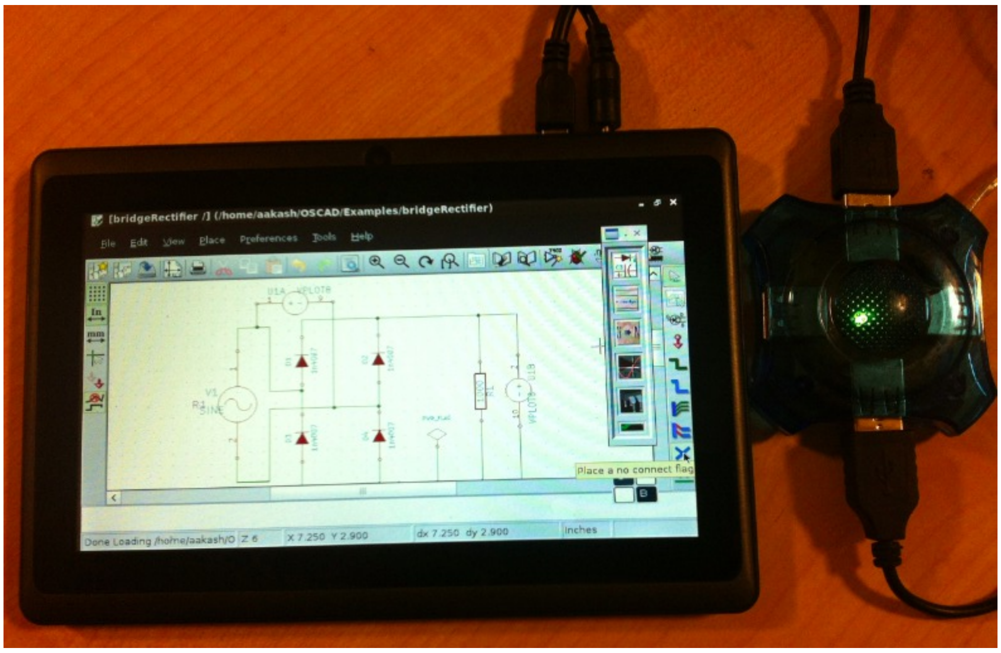
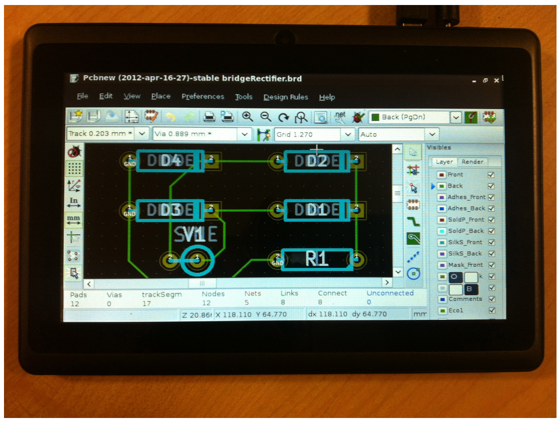
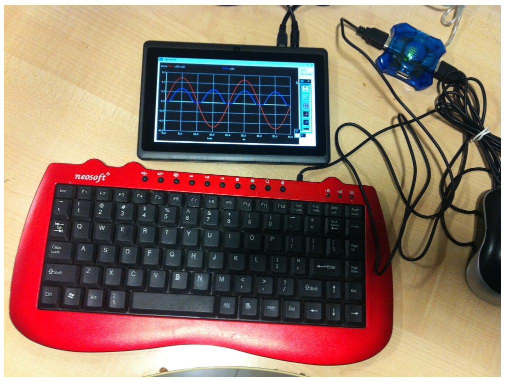


Figure 11.1: Bridge Rectifier schematic on Aakash. A USB hub is used with Aakash to connect a mouse and a keyboard.



(a) PCB layout using Oscad on Aakash



(b) Oscad running on Aakash with external keyboard and mouse attached. Output from Ngspice can be seen on Aakash.

Figure 11.2: Photographs of Oscad running on Aakash

Appendix A

Solved Examples from [1]

A few solved examples from [1] are presented in this section with identical example numbers. Oscad solutions for the other solved examples are available at [20]. The reader should understand the concepts explained in [1] before attempting this part. We invite the readers to solve using Oscad the worked out examples of the other circuit design textbooks as well.

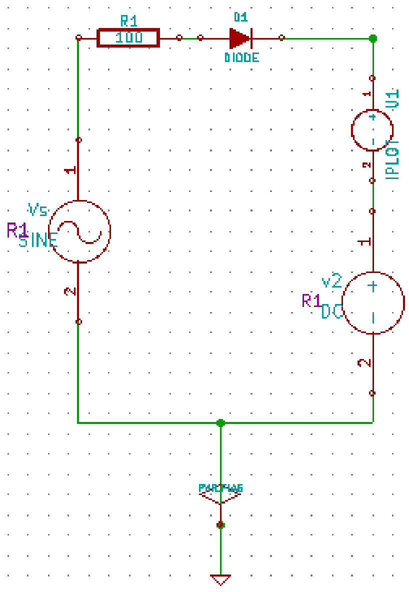
A.1 Diode

A.1.1 Example 2.1 from [1]

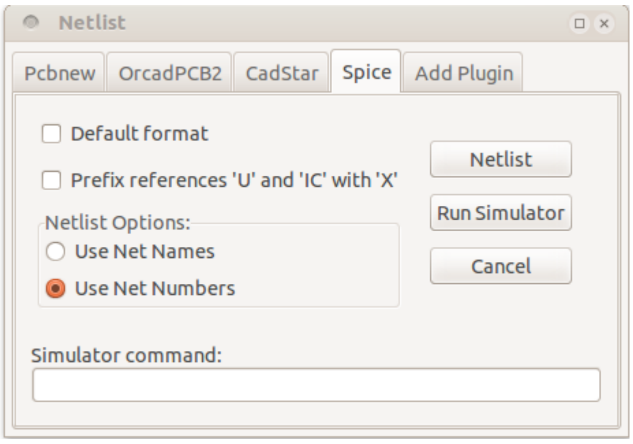
Problem Statement: Fig. A.1a shows a circuit for charging a 12-V battery. If v_s is a sinusoid with 24-V peak amplitude, find the fraction of each cycle during which diode conducts. Also, find the peak value of the diode current and the maximum reverse-bias voltage that appears across diode.

Solution: Draw the schematic shown in Fig. A.1a using the schematic editor. Annotate the schematic using the *Annotate* tool from the top toolbar in Schematic editor. Perform Electric Rules check using the *Perform electric rules check* tool from the top toolbar. Ensure that there are no errors in the circuit schematic. Now generate Spice netlist for simulation using the *Generate Netlist* tool from the top toolbar. This is shown in Fig. A.1b. The next step is to invoke Analysis inserter from the Oscad tool bar. Click on *Analysis Inserter* and select the option *Transient* as given in Fig. A.1c. Enter `start time` = 0, `step time` = 1 ms, `stop time` = 100 ms as in Fig. A.1d. Click on *Add Simulation Data* and save the analysis file as in Fig. A.2.

Now click on *Netlist Converter* from Oscad tool bar and enter the values of DC source and sine source as shown in Fig. A.3 and then press `Enter` key. This will generate the Ngspice netlist (.cir.out). Now click on *Ngspice* from the Oscad toolbar. This will open up the Ngspice terminal with waveform window as shown in Fig. A.4. Although not



(a) Schematic



Pcbnew | OrcadPCB2 | CadStar | Spice | Add Plugin

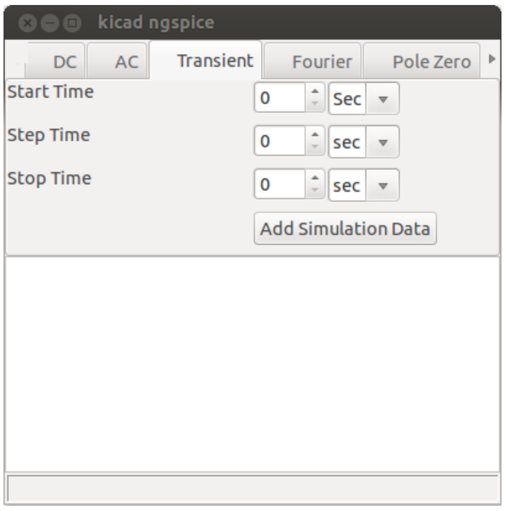
☐ Default format

☐ Prefix references 'U' and 'IC' with 'X'

Netlist Options:
Use Net Names [✓ Use Net Numbers]

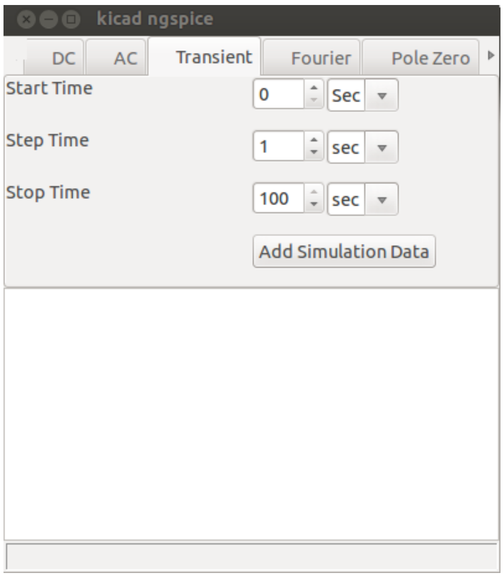
Simulator command:

(b) Netlist generation



Start Time	0	Sec
Step Time	0	sec
Stop Time	0	sec

(c) Analysis Inserter

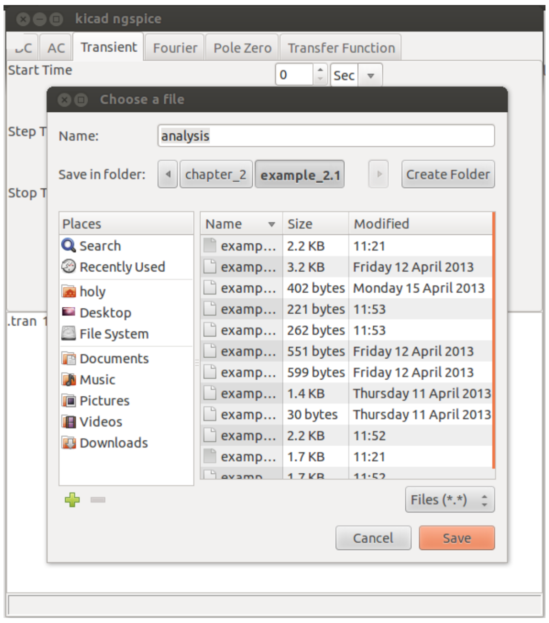


Start Time	0	Sec
Step Time	1	sec
Stop Time	100	sec

(d) Transient analysis options

Figure A.1: Initial steps in solving the diode problem in Sec. A.1.1

required for this example, the explanation given in Table A.1 will be useful if one wants to use a model builder, as explained in Fig. 8.5 on Page 79.

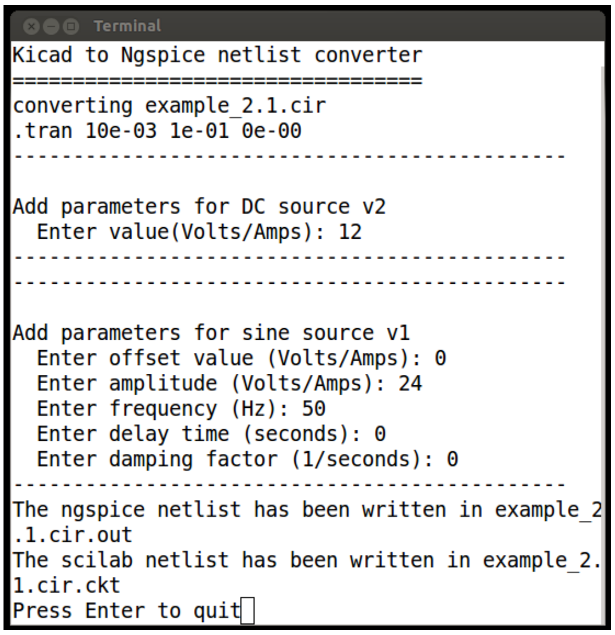


Name: analysis

Save in folder: chapter_2 example_2.1

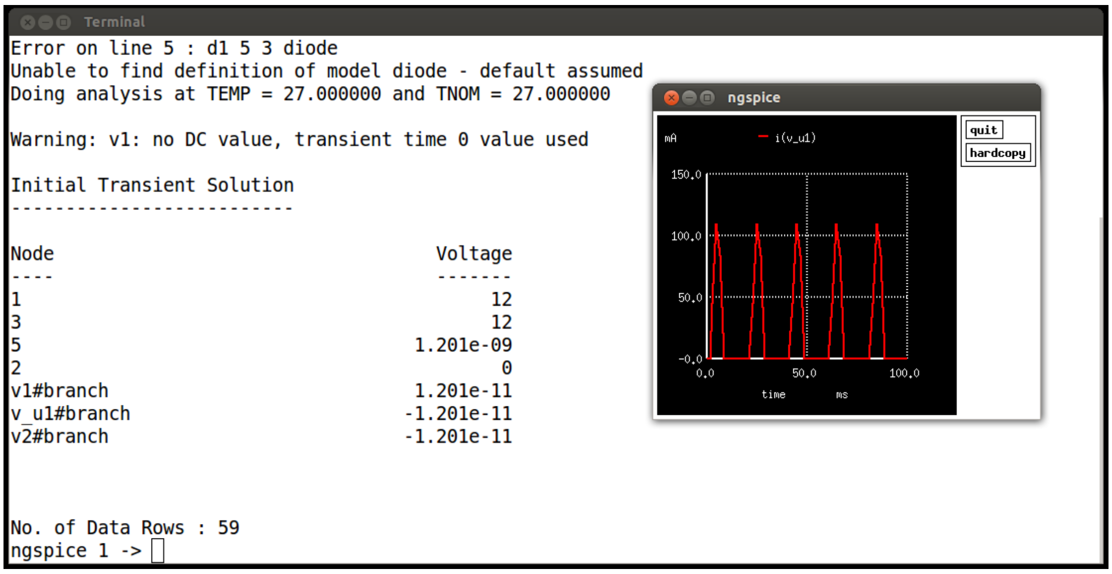
Name	Size	Modified
examp...	2.2 KB	11:21
examp...	3.2 KB	Friday 12 April 2013
examp...	402 bytes	Monday 15 April 2013
examp...	221 bytes	11:53
examp...	262 bytes	11:53
examp...	551 bytes	Friday 12 April 2013
examp...	599 bytes	Friday 12 April 2013
examp...	1.4 KB	Thursday 11 April 2013
examp...	30 bytes	Thursday 11 April 2013
examp...	2.2 KB	11:52
examp...	1.7 KB	11:21
examp...	1.7 KB	11:52

Figure A.2: Analysis Inserter: Saving the analysis file



```
Kicad to Ngspice netlist converter
==================================
converting example_2.1.cir
.tran 10e-03 1e-01 0e-00
----------------------------------------------

Add parameters for DC source v2
  Enter value(Volts/Amps): 12
----------------------------------------------
----------------------------------------------

Add parameters for sine source v1
  Enter offset value (Volts/Amps): 0
  Enter amplitude (Volts/Amps): 24
  Enter frequency (Hz): 50
  Enter delay time (seconds): 0
  Enter damping factor (1/seconds): 0
----------------------------------------------
The ngspice netlist has been written in example_2
.1.cir.out
The scilab netlist has been written in example_2.
1.cir.ckt
Press Enter to quit
```

Figure A.3: Entering the values of DC and sine sources during netlist conversion



```
Error on line 5 : d1 5 3 diode
Unable to find definition of model diode - default assumed
Doing analysis at TEMP = 27.000000 and TNOM = 27.000000

Warning: v1: no DC value, transient time 0 value used

Initial Transient Solution
--------------------------

Node                                       Voltage
----                                       -------
1                                               12
3                                               12
5                                        1.201e-09
2                                                0
v1#branch                                1.201e-11
v_u1#branch                             -1.201e-11
v2#branch                               -1.201e-11


No. of Data Rows : 59
ngspice 1 ->
```

Figure A.4: Ngspice simulation results for diode example in Sec. A.1.1, see Footnote 3 on Page 54.

Table A.1: Diode model parameters

Parameter	meaning
Bv	Reverse breakdown voltage
Cjo	Zero-bias junction capacitance
Eg	Band-gap energy
Fc	CJ forward-bias coefficient
Ibv	Current at breakdown voltage
Is	Saturation current
Kf	Flicker noise coefficient
M	Junction capacitance grading exponent
N	Emission coefficient
Rs	Ohmic resistance
Vj	Junction potential
Xti	IS temperature exponent

A.1.2 Example 2.5 from [1]

Problem Statement: Find the voltage V_D and current I_D in the circuit shown in Fig. A.5, where $V_{DO} = 0.65$, $r_D = 20\text{m}\Omega$.

Solution: Create schematic and generate netlist in the same way as given in Sec. A.1.1. Click on *Analysis Inserter* from Oscad tool bar. Select *DC* and then enter the following details: `enter source name` = v1, `start` = 0, `increment` = 1V and `stop` = 5V as given in Fig. A.6. Click on *Add Simulation Data* and save the analysis file. Now click on *Netlist Converter* tool and enter the value of DC source as shown in Fig. A.7 and then press `Enter` key. Now click on *Ngspice* from the Oscad tool bar. The results of Ngspice simulation is shown in Fig. A.8. This shows the current and voltage waveforms and Ngspice terminal.

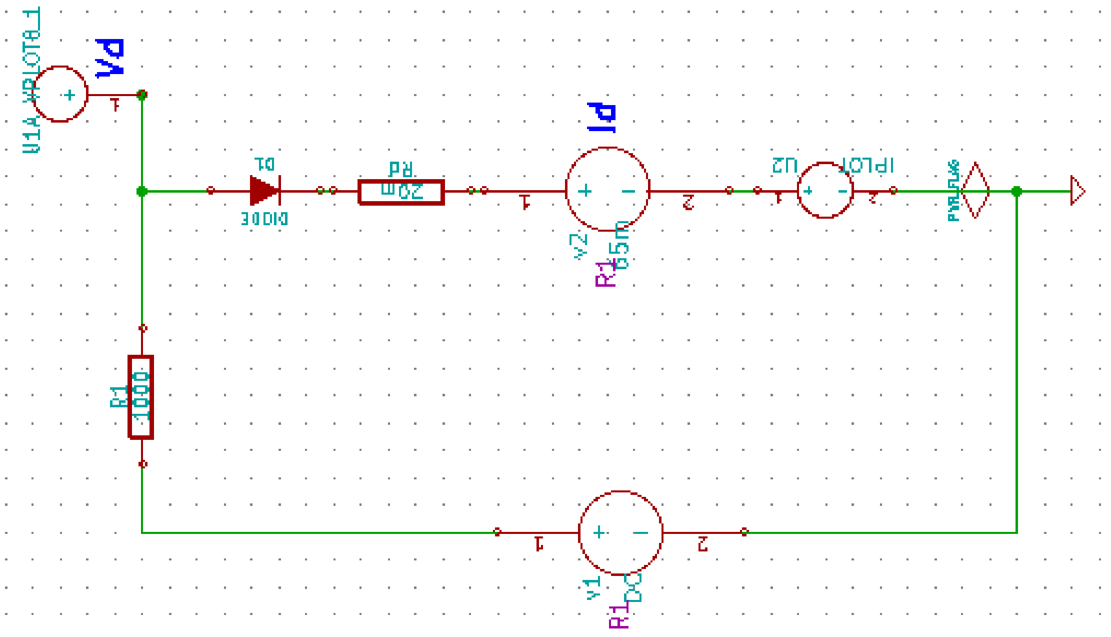
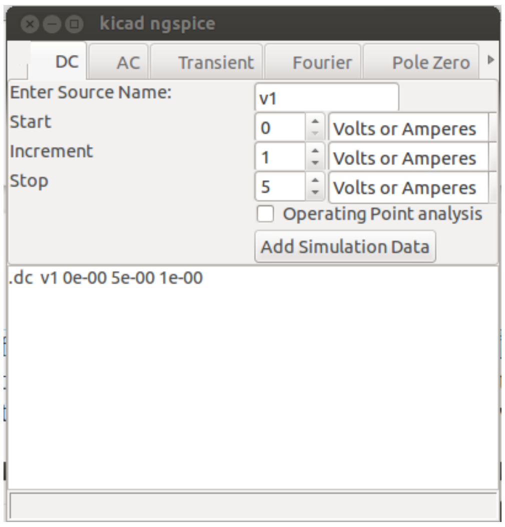


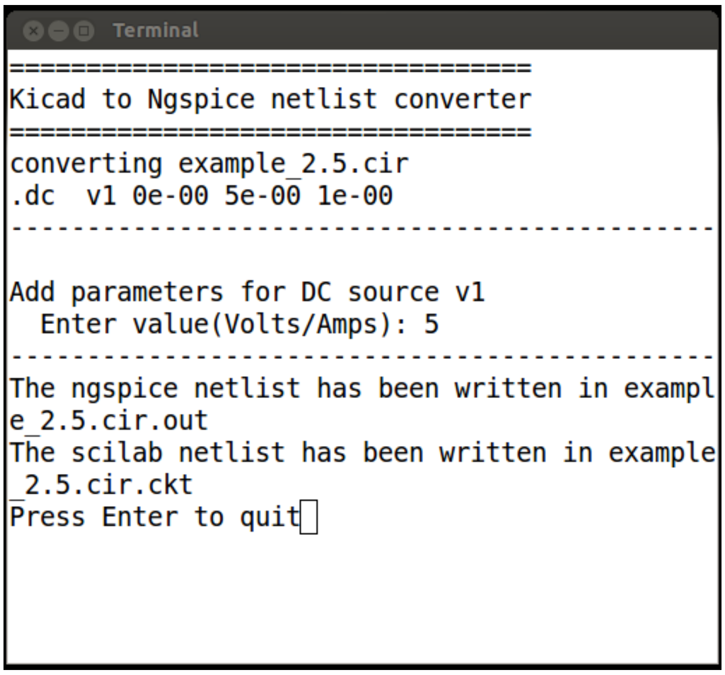
Figure A.5: Circuit for Example 2.5



Enter Source Name: v1
Start 0 Volts or Amperes
Increment 1 Volts or Amperes
Stop 5 Volts or Amperes
Operating Point analysis

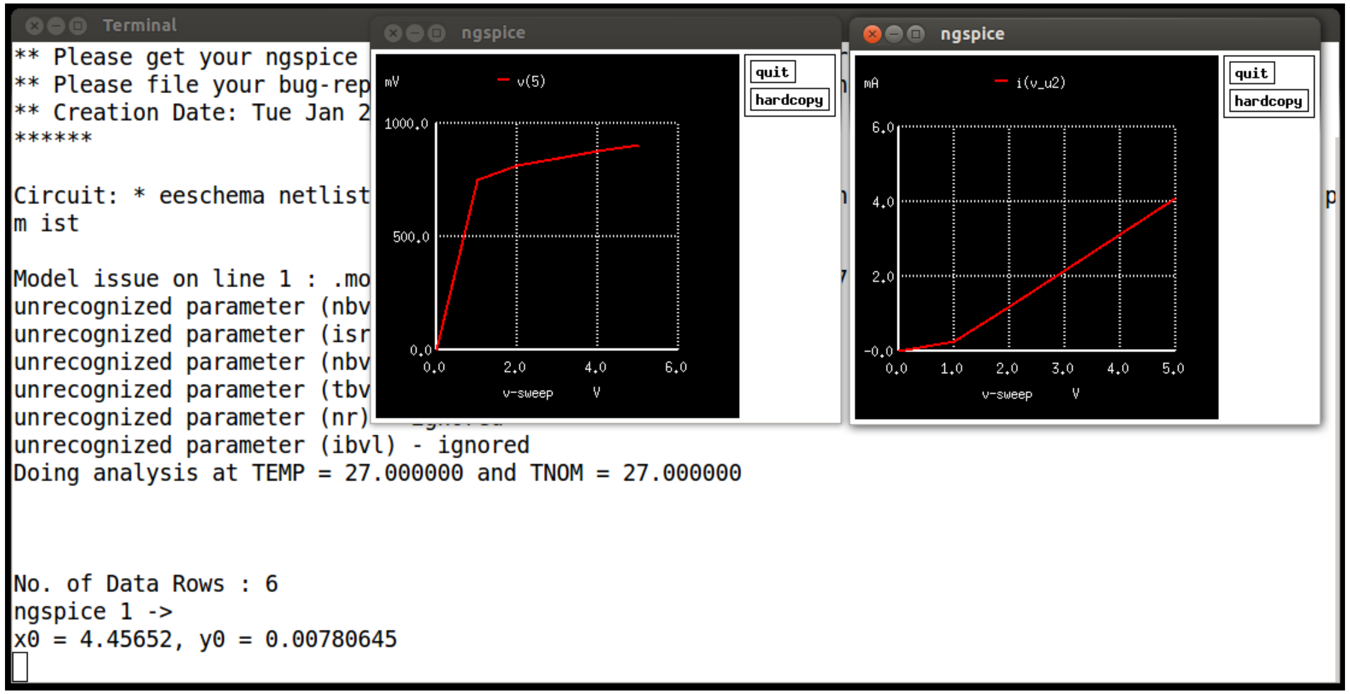
```
.dc v1 0e-00 5e-00 1e-00
```

Figure A.6: Entering DC analysis options for Example 2.5



```
==================================
Kicad to Ngspice netlist converter
==================================
converting example_2.5.cir
.dc  v1 0e-00 5e-00 1e-00
----------------------------------------------

Add parameters for DC source v1
  Enter value(Volts/Amps): 5
----------------------------------------------
The ngspice netlist has been written in exampl
e_2.5.cir.out
The scilab netlist has been written in example
_2.5.cir.ckt
Press Enter to quit
```

Figure A.7: Entering the value of DC source during netlist conversion for Example 2.5



```
** Please get your ngspice
** Please file your bug-rep
** Creation Date: Tue Jan 2
******

Circuit: * eeschema netlist
m ist

Model issue on line 1 : .mo
unrecognized parameter (nbv
unrecognized parameter (isr
unrecognized parameter (nbv
unrecognized parameter (tbv
unrecognized parameter (nr)
unrecognized parameter (ibvl) - ignored
Doing analysis at TEMP = 27.000000 and TNOM = 27.000000



No. of Data Rows : 6
ngspice 1 ->
x0 = 4.45652, y0 = 0.00780645
```

Figure A.8: Ngspice simulation results for diode Example 2.5, see Footnote 3 on Page 54.

A.2 BJT: Example 3.1 from [1]

Problem Statement: Find the voltage at all nodes in the transistors shown in Fig. A.9. We will assume that β is specified to be 100.

Solution: Create schematic and generate netlist in the same way as given in Sec. A.1.1. Click on *Analysis Inserter* tool from Oscad tool bar. Select *DC* and then enter the following details: `enter source name` = v1, `start` = 0, `increment` = 1V and `stop` = 4V as given in Fig. A.10. Click on *Add Simulation Data* and save the analysis file. Now click on *Netlist Converter*. A terminal window will open as shown in Fig. A.11. Press `Enter` key. Now click on `Ngspice` from the Oscad tool bar. The results of Ngspice simulation is shown in Fig. A.12.

Although not required in this example, if model building is required as explained through Fig. 8.5 on Page 79, the explanation in Table A.2 will be useful.

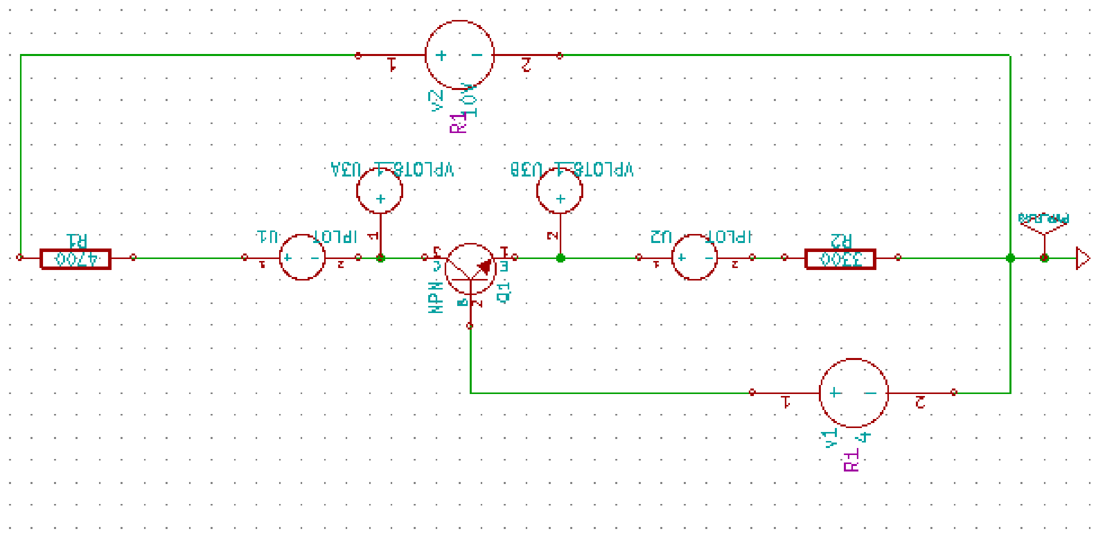
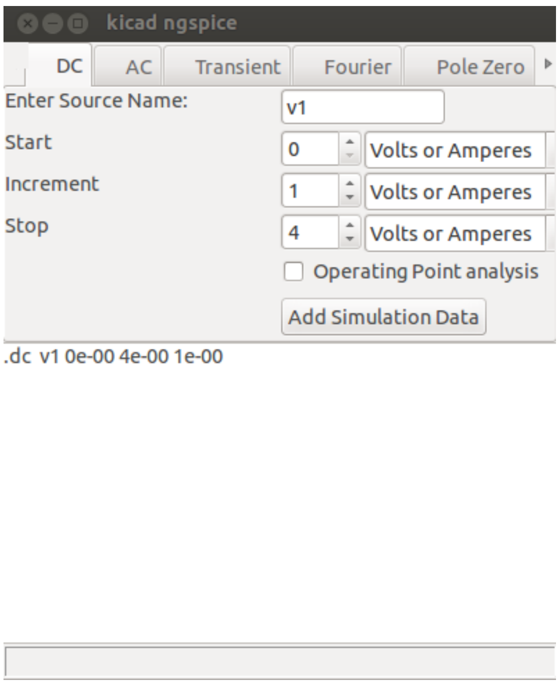


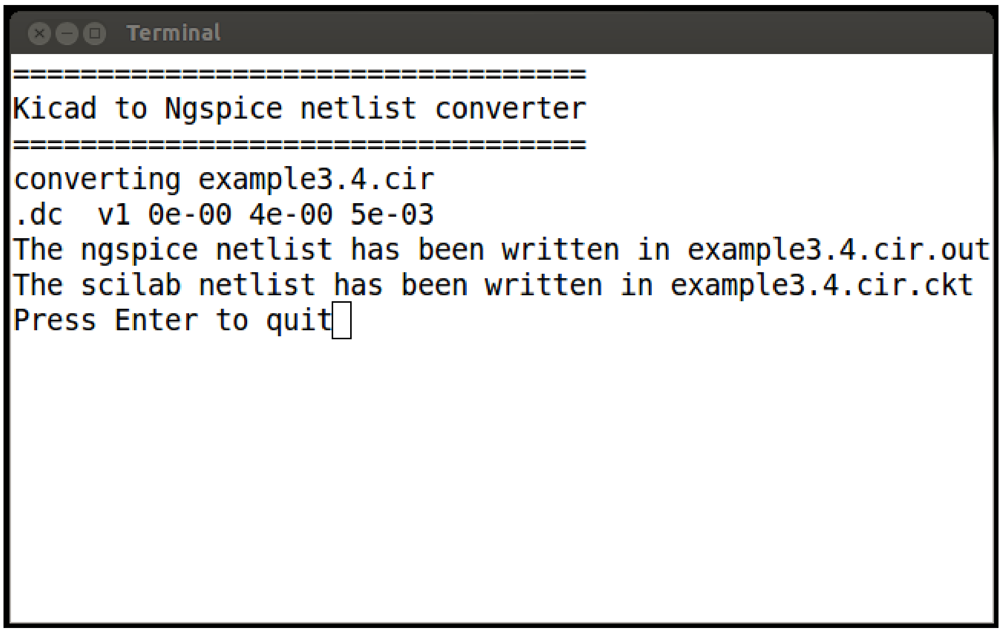
Figure A.9: Circuit for Example 3.1



Enter Source Name: v1
Start 0 Volts or Amperes
Increment 1 Volts or Amperes
Stop 4 Volts or Amperes
Operating Point analysis

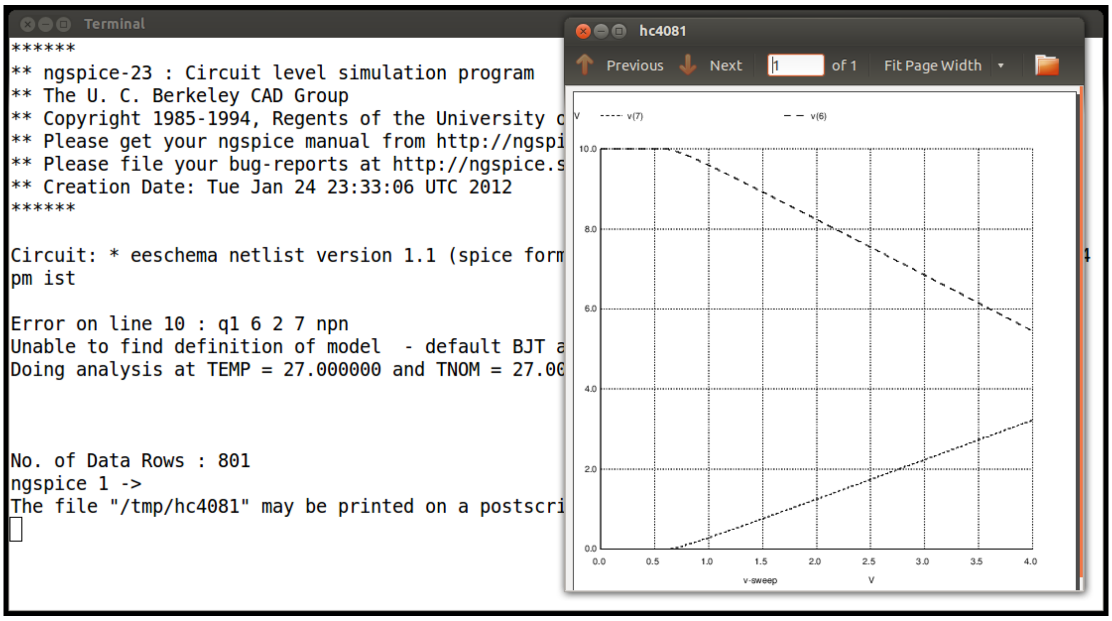
```
.dc v1 0e-00 4e-00 1e-00
```

Figure A.10: Entering DC analysis options for Example 3.1



```
==================================
Kicad to Ngspice netlist converter
==================================
converting example3.4.cir
.dc  v1 0e-00 4e-00 5e-03
The ngspice netlist has been written in example3.4.cir.out
The scilab netlist has been written in example3.4.cir.ckt
Press Enter to quit
```

Figure A.11: Netlist converter terminal window for Example 3.1



```
******
** ngspice-23 : Circuit level simulation program
** The U. C. Berkeley CAD Group
** Copyright 1985-1994, Regents of the University c
** Please get your ngspice manual from http://ngspi
** Please file your bug-reports at http://ngspice.s
** Creation Date: Tue Jan 24 23:33:06 UTC 2012
******

Circuit: * eeschema netlist version 1.1 (spice form
pm ist

Error on line 10 : q1 6 2 7 npn
Unable to find definition of model  - default BJT a
Doing analysis at TEMP = 27.000000 and TNOM = 27.00


No. of Data Rows : 801
ngspice 1 ->
The file "/tmp/hc4081" may be printed on a postscri
```

Figure A.12: Ngspice simulation results for BJT Example 3.1, see Footnote 3 on Page 54. The plot shows v(7) and v(6). v(6) is shown with long dashes.

Table A.2: BJT model parameters

Parameter	Meaning
Bf	Forward active current gain
Br	Reverse active current gain
Cjc	Base-collector zero-bias Junction capacitance
Cje	Base-emitter zero-bias Junction capacitance
Eg	energy gap for temperature effect on IS
Is	Transport saturation current
Mjc	base-collector capacitance exponent
Mje	base-emitter capacitance exponent
Rb	zero bias base resistance
Rc	collector resistance
Vaf	Forward mode Early voltage
Vjc	Base-collector built-in potential
Vje	Base-emitter built-in potential

A.3 MOSFET: Example 4.5 from [1]

Problem statement: Analyse the circuit shown in Fig. A.13 to determine the voltages at all nodes and the current through all branches.

Solution: Create schematic and generate netlist in the same way as given in Sec. A.1.1. Click on *Analysis inserter* from Oscad tool bar. Select *DC* and then enter the following details: `enter source name` = v1, `start` = 0, `increment` = 1V and `stop` = 10V as given in Fig. A.14. Click on *Add Simulation Data* and save the analysis file. Now click on *Netlist Converter*. A terminal window will open as shown in Fig. A.15. Press `Enter` key. Now click on *Ngspice* from the Oscad tool bar. The results of Ngspice simulation is shown in Fig. A.16.

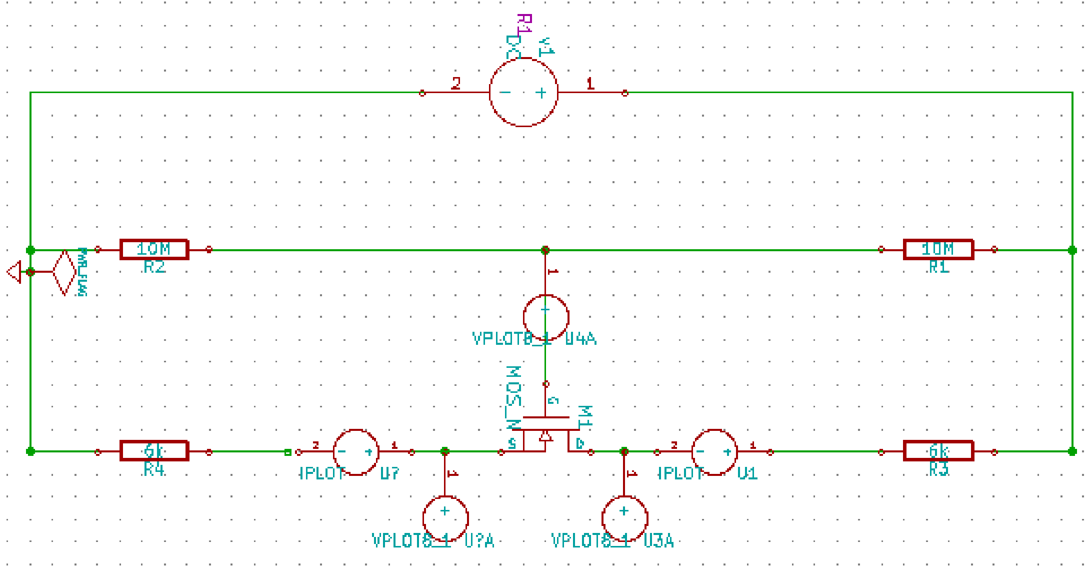
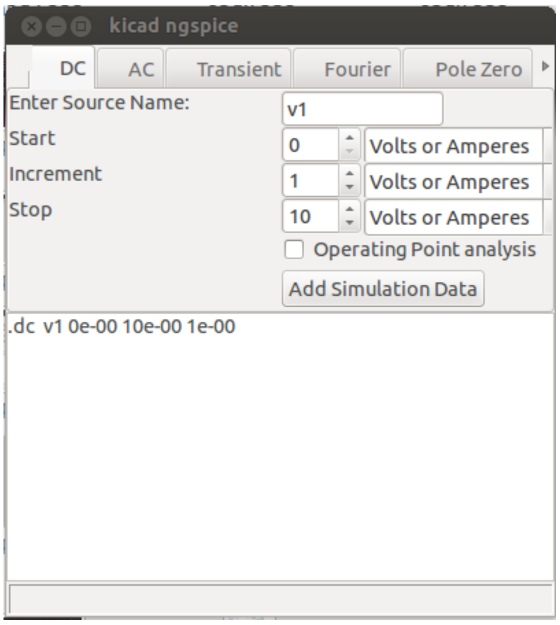


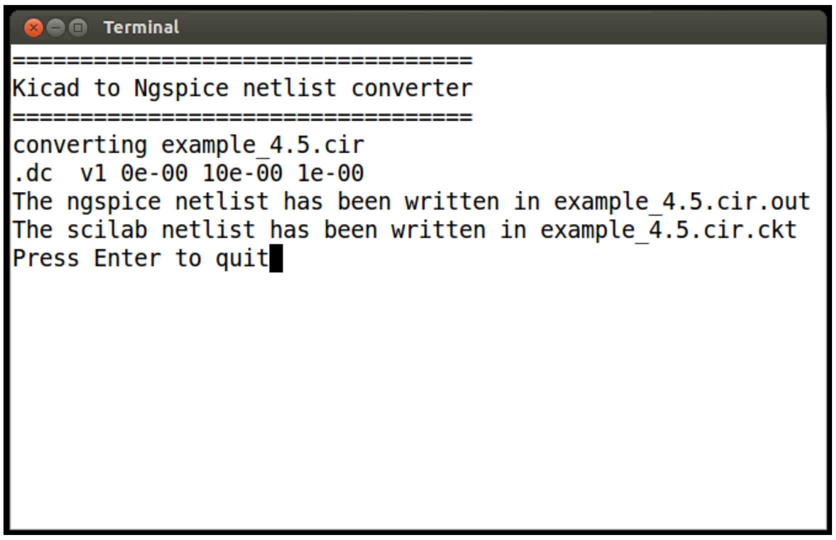
Figure A.13: Circuit for Example 4.5



Enter Source Name: v1
Start 0 Volts or Amperes
Increment 1 Volts or Amperes
Stop 10 Volts or Amperes
Operating Point analysis

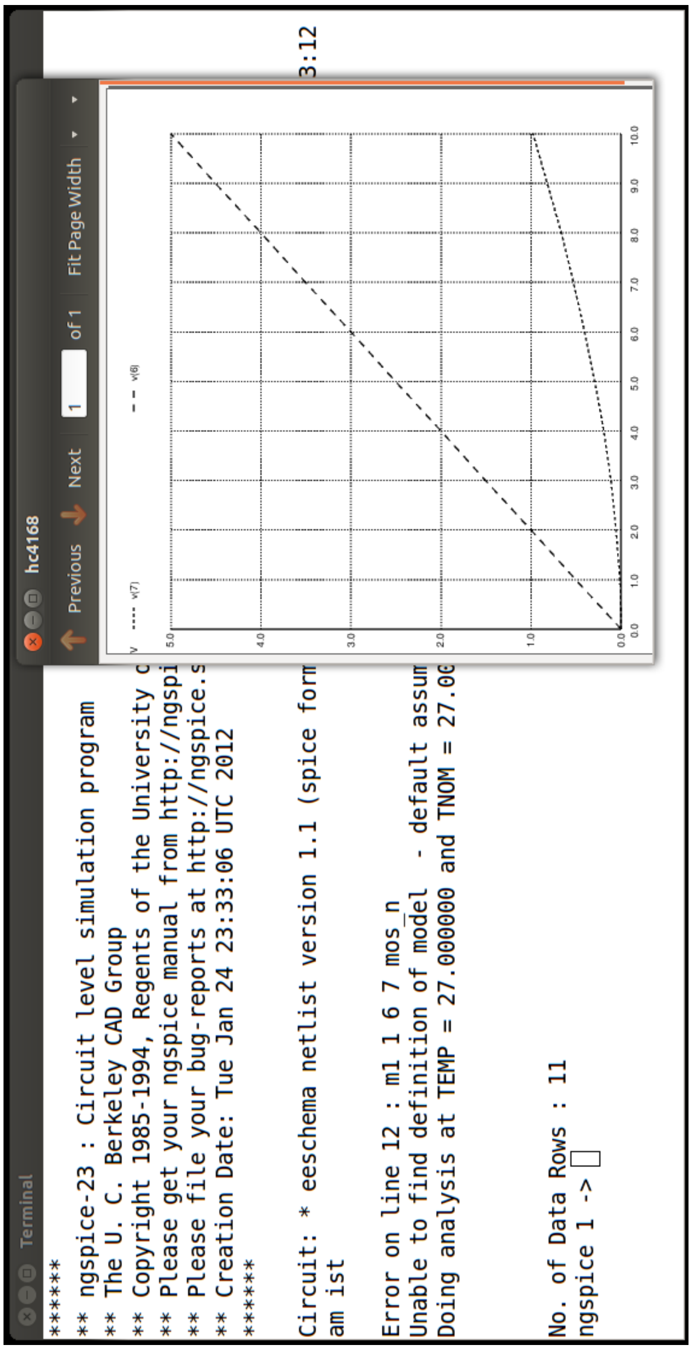
```
.dc v1 0e-00 10e-00 1e-00
```

Figure A.14: Entering DC analysis options for Example 4.5



```
==================================
Kicad to Ngspice netlist converter
==================================
converting example_4.5.cir
.dc  v1 0e-00 10e-00 1e-00
The ngspice netlist has been written in example_4.5.cir.out
The scilab netlist has been written in example_4.5.cir.ckt
Press Enter to quit
```

Figure A.15: Netlist converter terminal window for Example 4.5



```
******
** ngspice-23 : Circuit level simulation program
** The U. C. Berkeley CAD Group
** Copyright 1985-1994, Regents of the University c
** Please get your ngspice manual from http://ngspi
** Please file your bug-reports at http://ngspice.s
** Creation Date: Tue Jan 24 23:33:06 UTC 2012
******

Circuit: * eeschema netlist version 1.1 (spice form
am ist

Error on line 12 : m1 1 6 7 mos_n
Unable to find definition of model  - default assum
Doing analysis at TEMP = 27.000000 and TNOM = 27.00

No. of Data Rows : 11
ngspice 1 ->
```

Figure A.16: Ngspice simulation results for MOSFET Example 4.5, see Footnote 3 on Page 54. Figure shows v(7) and v(6). v(6) is shown with long dashes.

A.4 OP-AMP

A.4.1 Example 5.3 from [1]

Problem statement: Fig. A.17 shows an op-amp circuit. Find the output voltage where $R_1 = 10\ \Omega$, $R_f = 1\text{k}\Omega$, $R_L = 100\text{k}\Omega$ and input voltage $v_I = 10\text{V}$.

Solution: Create schematic and generate netlist in the same way as given in Example 2.1. While generating netlist, DO NOT uncheck the option `Prefix references 'U' and 'IC' with 'X'`.

Click on *Analysis inserter* from Oscad tool bar. Select *DC* and then enter the following details: `enter source name` = v1, `start` = 0, `increment` = 1V and `stop` = 10V as given in Fig. A.14. Click on *Add Simulation Data* and save the analysis file. Click on *Subcircuit Builder* from the Oscad toolbar and click on *Cancel* in the subcircuit selector window. Now, *import* the existing subcircuits. Fig. A.18a will appear. Choose `ua741` and click on *ok*. Fig. A.18b will now appear. Click on ok in the `Successfully imported` window.

Let us open the subcircuit schematic of `ua741`. This can be done as follows: Click on the *File* menu. Choose *open* and then type `ua741` in the *Enter Component name* field as shown in Fig. A.19. Click on ok. Add the Oscad libraries to the subcircuit schematic to prevent the Load Error. Refer to Sec. 5.2.2 to know how to add Oscad libraries to the schematic. The schematic of the subcircuit is as shown in Fig. A.20.

Now click on *Netlist Converter*. A terminal window will open as shown in Fig. A.21. Press the`Enter` key. Now click on *Ngspice* from the Oscad toolbar. Fig. A.22 will appear.

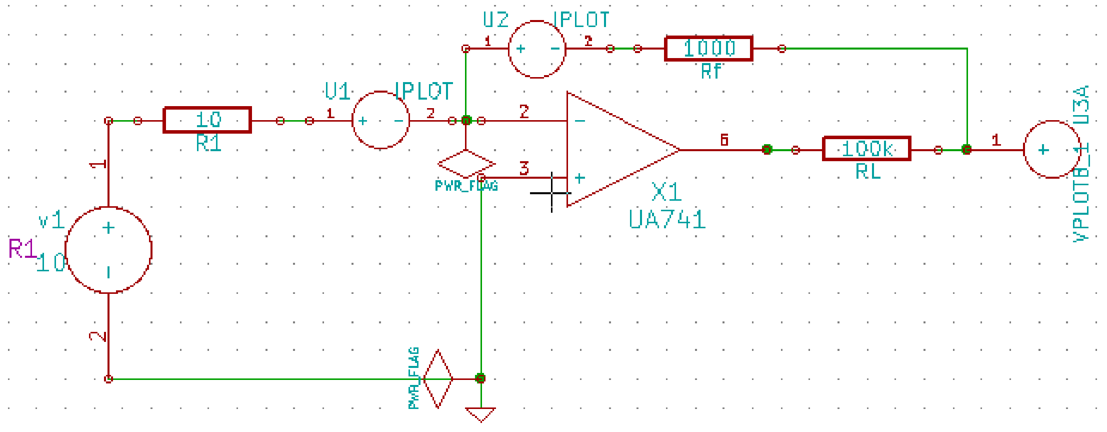
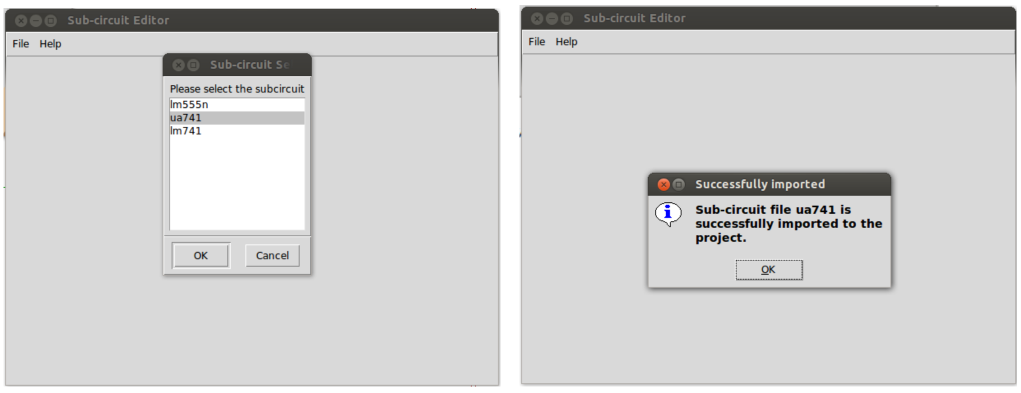


Figure A.17: Circuit for Example 5.3

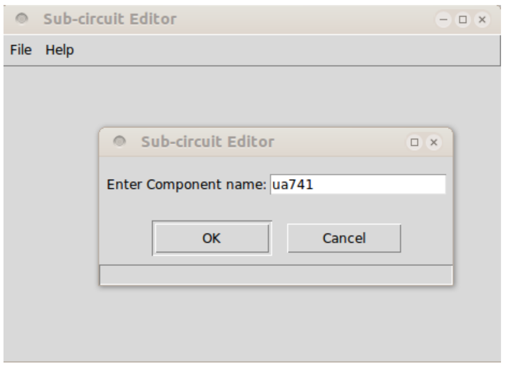


Please select the subcircuit
lm555n
ua741
lm741

Sub-circuit file ua741 is successfully imported to the project.

(a) Selecting subcircuit (b) Successful importing of subcircuit

Figure A.18: Steps in importing existing subcircuits



Enter Component name: ua741

Figure A.19: Open component subcircuit

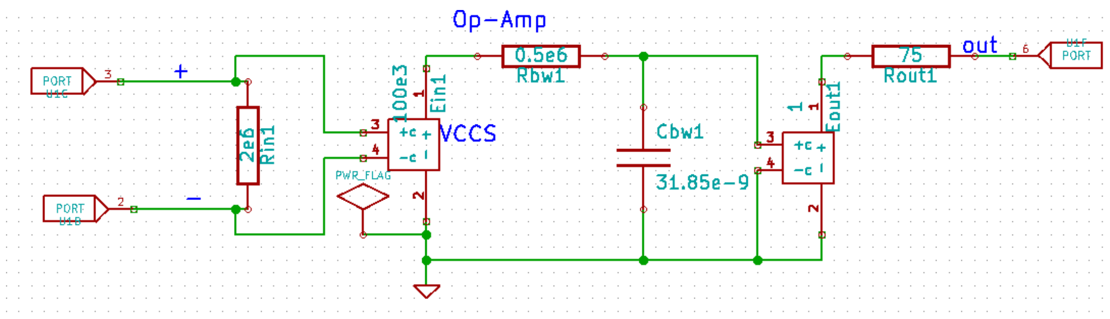
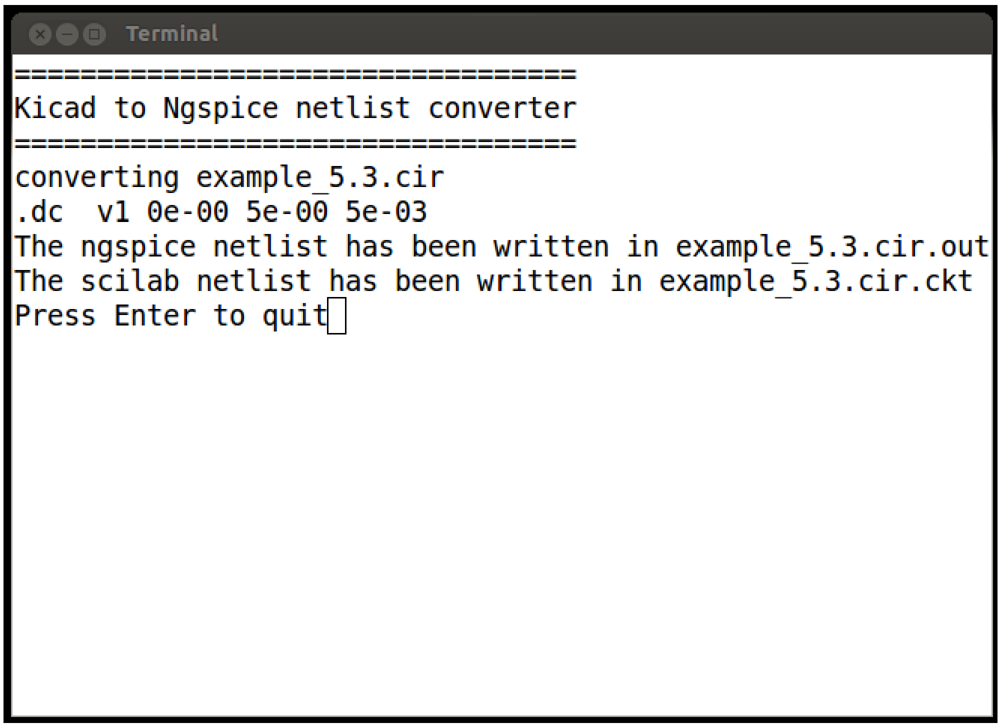
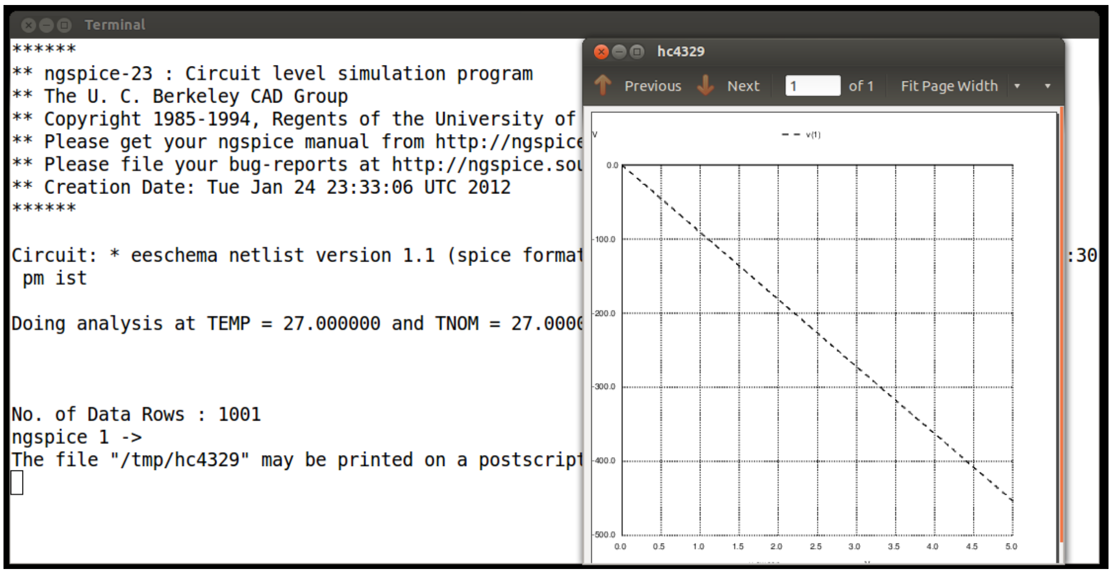


Figure A.20: Subcircuit schematic



```
==================================
Kicad to Ngspice netlist converter
==================================
converting example_5.3.cir
.dc  v1 0e-00 5e-00 5e-03
The ngspice netlist has been written in example_5.3.cir.out
The scilab netlist has been written in example_5.3.cir.ckt
Press Enter to quit
```

Figure A.21: Netlist converter terminal window for Example 5.3



```
******
** ngspice-23 : Circuit level simulation program
** The U. C. Berkeley CAD Group
** Copyright 1985-1994, Regents of the University of
** Please get your ngspice manual from http://ngspice
** Please file your bug-reports at http://ngspice.sou
** Creation Date: Tue Jan 24 23:33:06 UTC 2012
******

Circuit: * eeschema netlist version 1.1 (spice format                              :30
 pm ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.0000


No. of Data Rows : 1001
ngspice 1 ->
The file "/tmp/hc4329" may be printed on a postscript
```

Figure A.22: Ngspice simulation results for Op-amp Example 5.3, see Footnote 3 on Page 54. Figure shows v(1).

A.4.2 Example 5.6 from [1]

Problem statement: Consider the non inverting amplifier circuit shown in Fig. A.23. It is fed with a lower frequency sinusoidal signal of peak voltage $V_i = 1$V, and is connected to load resistance $R_L = 1$kΩ.

Solution: Create schematic and generate netlist in the same way as given in Sec. A.1.1. Click on *Analysis Inserter* from Oscad tool bar. Select *Transient* and then enter the following details: `Start time` = 0 sec, `Step time` = 1 ms, `stop time` = 20 ms as given in Fig. A.24. Click on *Add Simulation Data* and save the analysis file. For subcircuit builder, follow the steps given in Sec. A.4.1. Now click on *Netlist Converter* and enter the value of sine source as shown in Fig. A.25 and then press `Enter` key. Now click on *Ngspice* tool from Oscad toolbar. Fig. A.26 will appear.

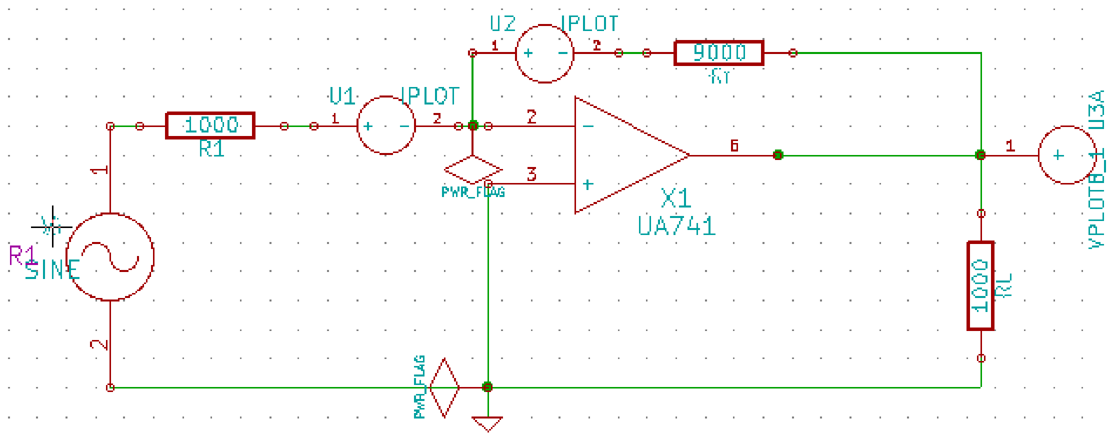
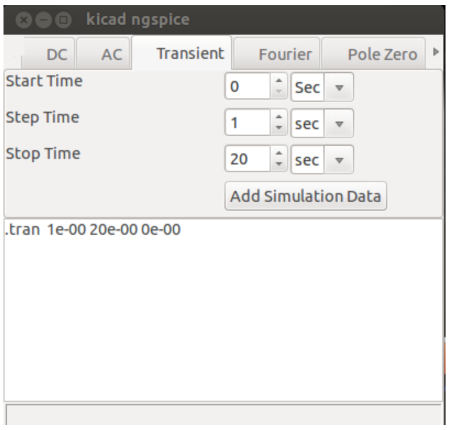


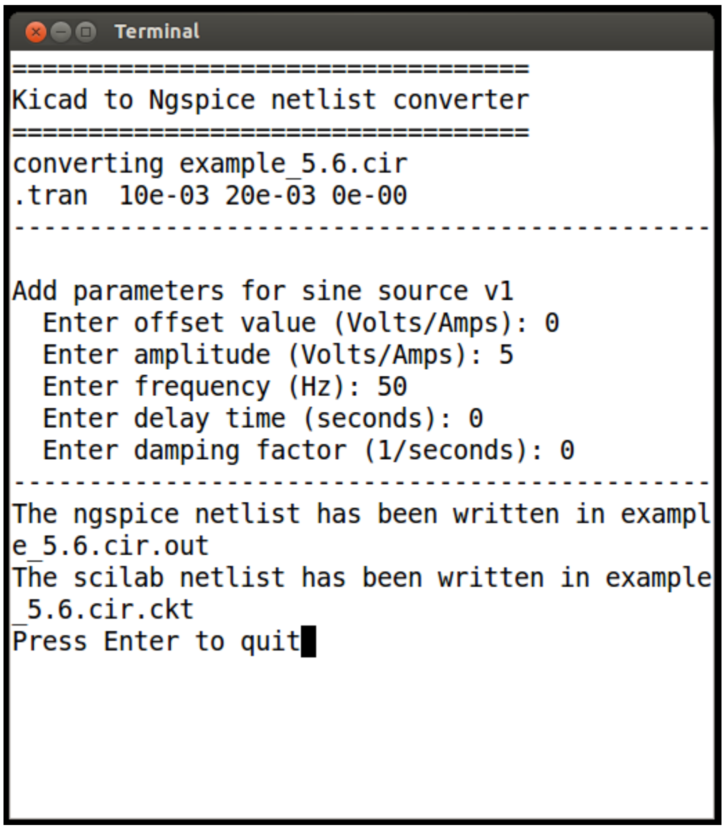
Figure A.23: Circuit for Example 5.6



Start Time 0 Sec
Step Time 1 sec
Stop Time 20 sec

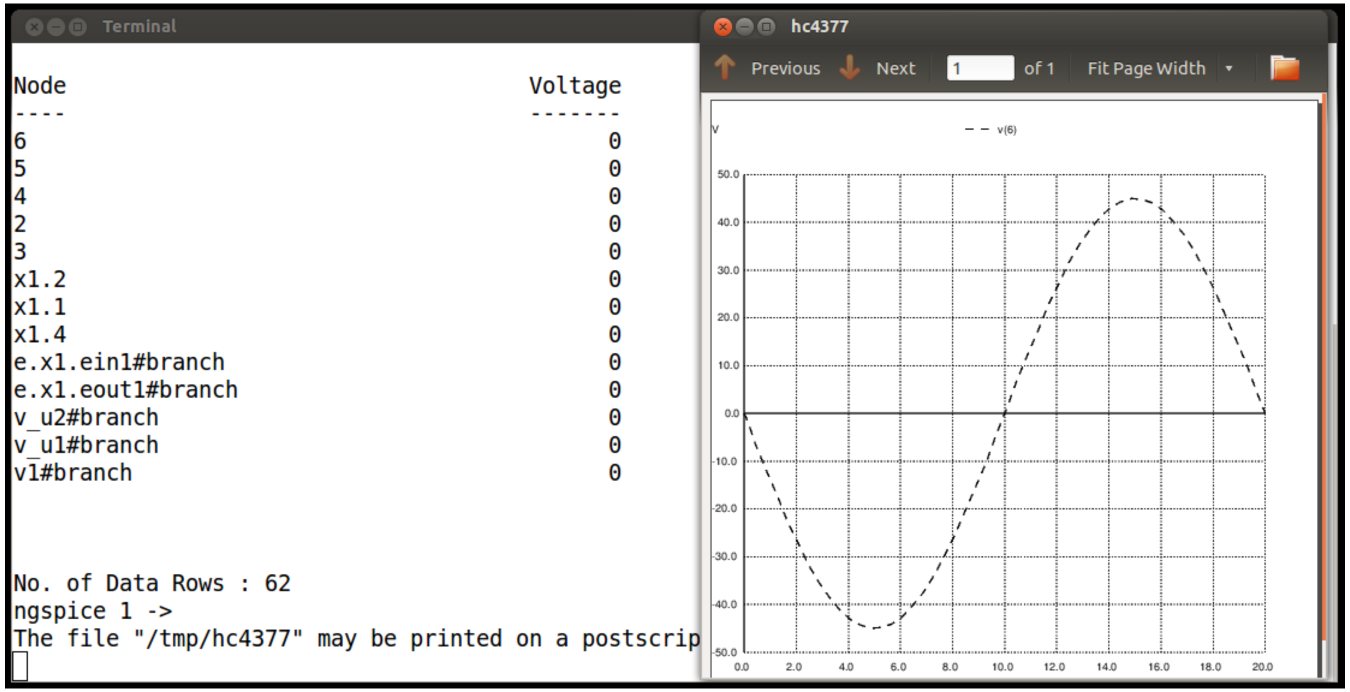
```
.tran 1e-00 20e-00 0e-00
```

Figure A.24: Entering Transient analysis options for Example 5.6



```
Terminal
==================================
Kicad to Ngspice netlist converter
==================================
converting example_5.6.cir
.tran  10e-03 20e-03 0e-00
----------------------------------------------

Add parameters for sine source v1
  Enter offset value (Volts/Amps): 0
  Enter amplitude (Volts/Amps): 5
  Enter frequency (Hz): 50
  Enter delay time (seconds): 0
  Enter damping factor (1/seconds): 0
----------------------------------------------
The ngspice netlist has been written in exampl
e_5.6.cir.out
The scilab netlist has been written in example
_5.6.cir.ckt
Press Enter to quit
```

Figure A.25: Entering sine source parameters during netlist conversion for Example 5.6



```
Node                                       Voltage
----                                       -------
6                                                0
5                                                0
4                                                0
2                                                0
3                                                0
x1.2                                             0
x1.1                                             0
x1.4                                             0
e.x1.ein1#branch                                 0
e.x1.eout1#branch                                0
v_u2#branch                                      0
v_u1#branch                                      0
v1#branch                                        0


No. of Data Rows : 62
ngspice 1 ->
The file "/tmp/hc4377" may be printed on a postscrip
```

Figure A.26: Ngspice simulation results for op-amp Example 5.6, see Footnote 3 on Page 54. Figure shows v(6).

A.4.3 Example 5.7 from [1]

Problem statement: For the circuit given in Fig. A.27, perform AC analysis where $R_1 = 1\text{k}\Omega$, $R_f = 100\text{k}\Omega$, $C_f = 1.59\text{nF}$ and input voltage $V_i = 1\text{V}$ A.C.

Solution: Create schematic and generate netlist in the same way as given in Sec. A.1.1. Click on *Analysis Inserter* from Oscad tool bar. Select *AC* and then enter the following details: `scale` = Lin, `start frequency` = 1, `stop frequency` = 1 Meg, `No. of points` = 10 as given in Fig. A.28. Click on *Add Simulation Data* and save the analysis file. For subcircuit builder, follow the steps given in Sec. A.4.1. Now click on *Netlist Converter* and enter the value of AC source as shown in Fig. A.29 and then press `Enter` key. Now click on *Ngspice* tool in Toolbar. Fig. A.30 will appear.

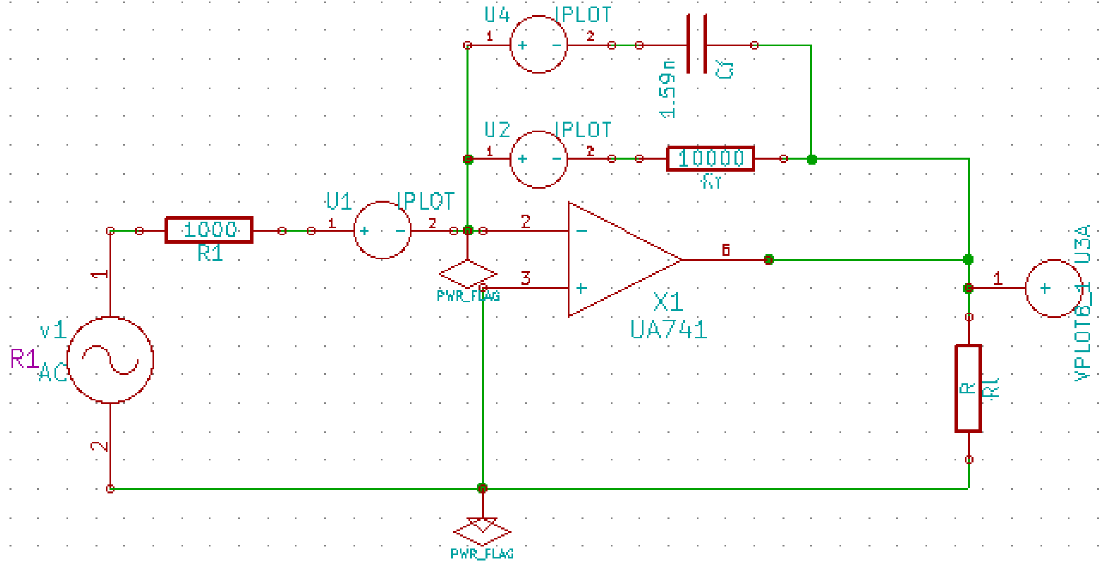
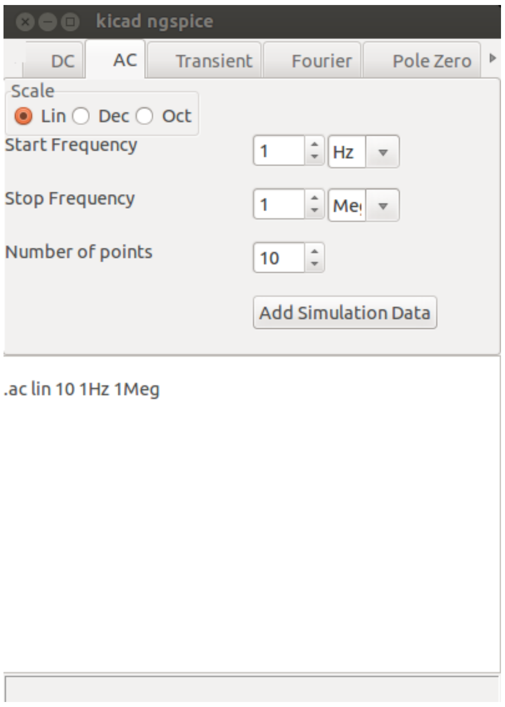


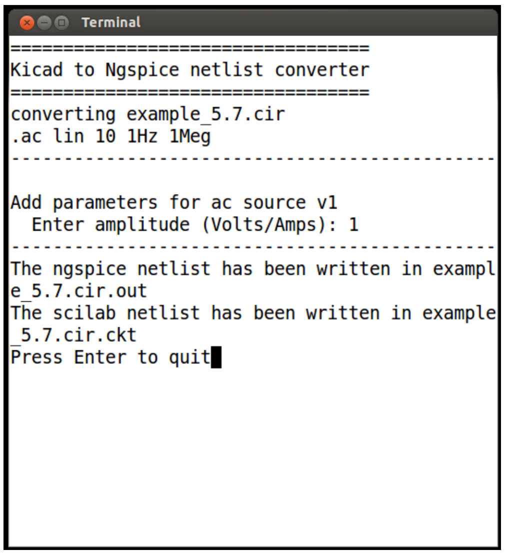
Figure A.27: Circuit for Example 5.7



Scale: [✓ Lin] Dec Oct
Start Frequency 1 Hz
Stop Frequency 1 Me
Number of points 10

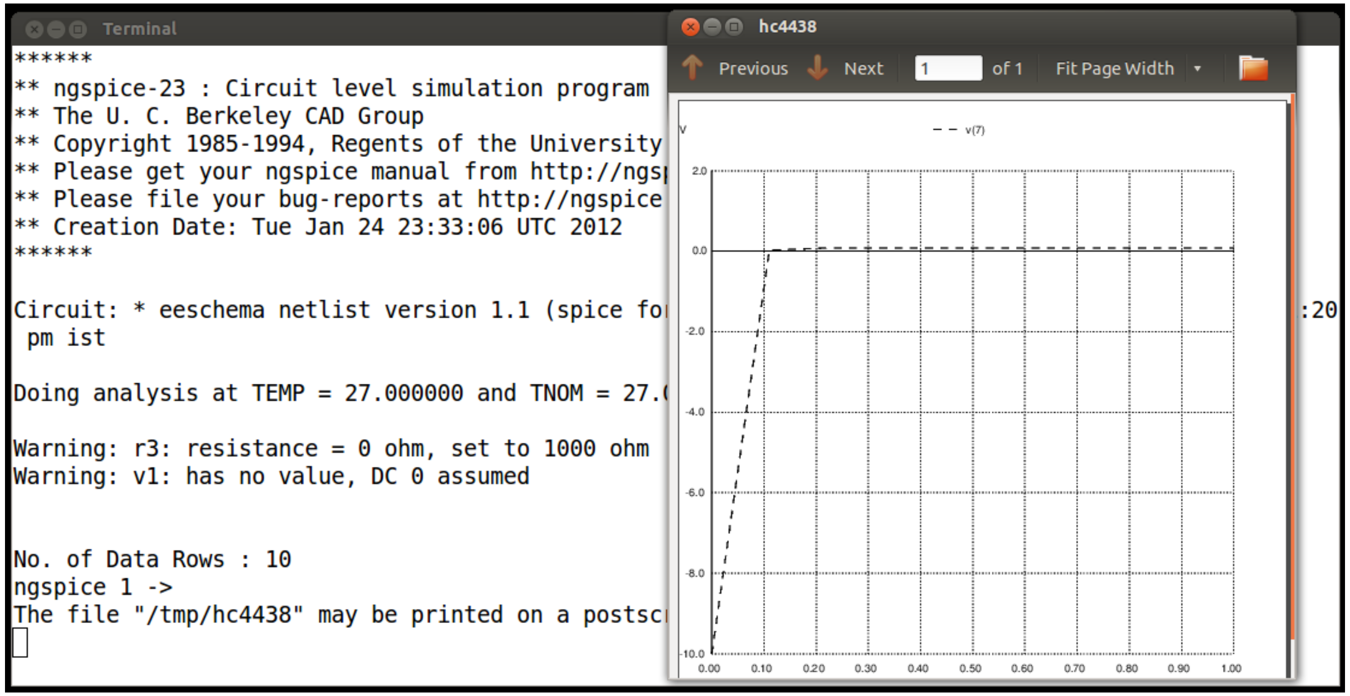
```
.ac lin 10 1Hz 1Meg
```

Figure A.28: Entering AC analysis options for Example 5.7



```
==================================
Kicad to Ngspice netlist converter
==================================
converting example_5.7.cir
.ac lin 10 1Hz 1Meg
----------------------------------------------

Add parameters for ac source v1
  Enter amplitude (Volts/Amps): 1
----------------------------------------------
The ngspice netlist has been written in exampl
e_5.7.cir.out
The scilab netlist has been written in example
_5.7.cir.ckt
Press Enter to quit
```

Figure A.29: Entering AC source amplitude during netlist conversion for Example 5.7



```
******
** ngspice-23 : Circuit level simulation program
** The U. C. Berkeley CAD Group
** Copyright 1985-1994, Regents of the University
** Please get your ngspice manual from http://ngsp
** Please file your bug-reports at http://ngspice
** Creation Date: Tue Jan 24 23:33:06 UTC 2012
******

Circuit: * eeschema netlist version 1.1 (spice fo
 pm ist

Doing analysis at TEMP = 27.000000 and TNOM = 27.(

Warning: r3: resistance = 0 ohm, set to 1000 ohm
Warning: v1: has no value, DC 0 assumed


No. of Data Rows : 10
ngspice 1 ->
The file "/tmp/hc4438" may be printed on a postsc
```

Figure A.30: Ngspice simulation results for op-amp Example 5.7, see Footnote 3 on Page 54. Figure shows v(7).

A.5 CMOS Inverter: Example 9.4 from [1]

Solution: Create schematic and generate netlist in the same way as given in Sec. A.1.1. Click on *Analysis Inserter* from Oscad tool bar. Select *DC* and then enter the following details: `enter source name` = v1, `start` = 0, `increment` = 1 V, `stop` = 15V as given in Fig. A.32a. Click on *Add Simulation Data* and save the analysis file. Now click on *Netlist Converter* and enter the value of DC source as shown in Fig. A.32b and then press `Enter` key. Now click on *Ngspice* tool from Oscad toolbar. The Fig. A.33 will appear.

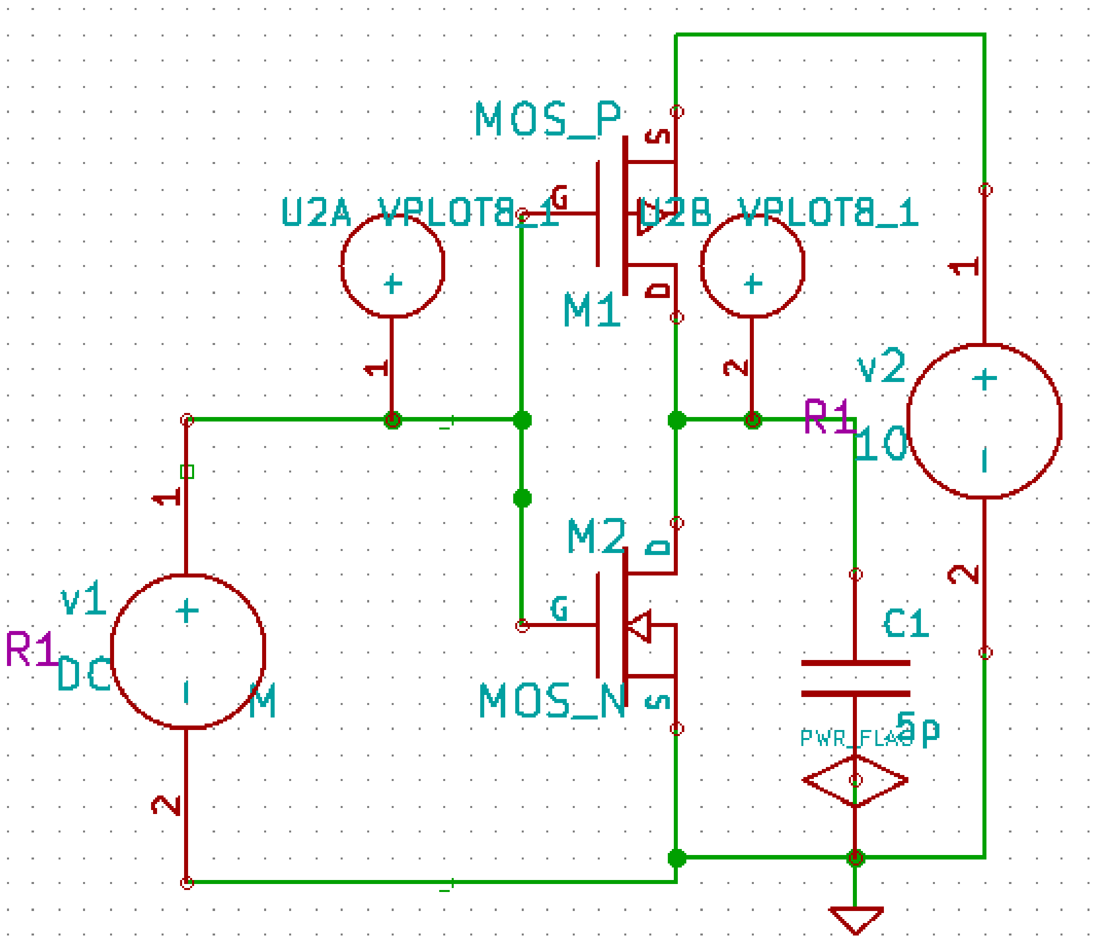
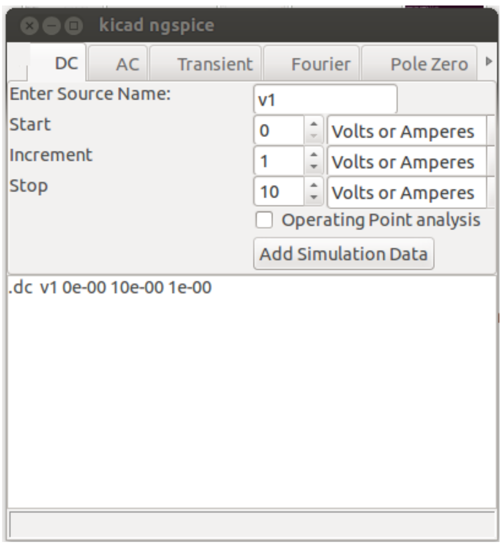


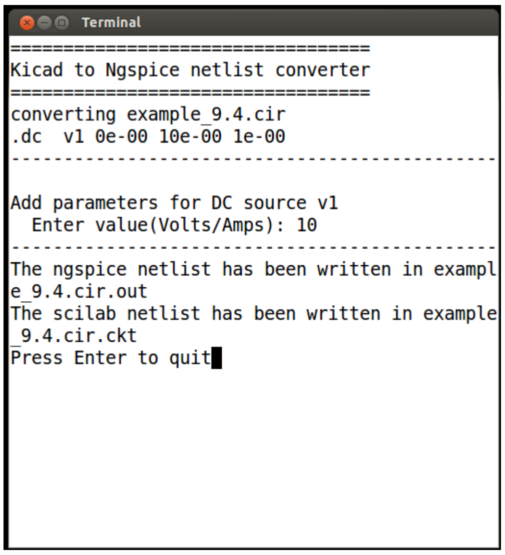
Figure A.31: Circuit for Example 9.4



Enter Source Name: v1
Start 0 Volts or Amperes
Increment 1 Volts or Amperes
Stop 10 Volts or Amperes
Operating Point analysis

```
.dc v1 0e-00 10e-00 1e-00
```

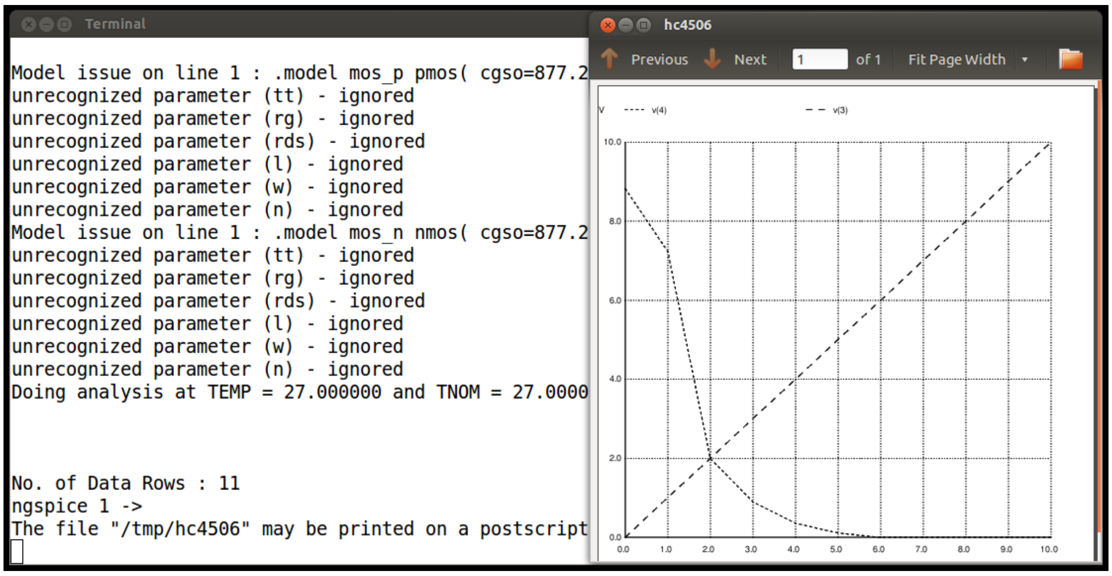
(a) Entering DC analysis options for Example 9.4



```
==================================
Kicad to Ngspice netlist converter
==================================
converting example_9.4.cir
.dc  v1 0e-00 10e-00 1e-00
---------------------------------------------

Add parameters for DC source v1
  Enter value(Volts/Amps): 10
---------------------------------------------
The ngspice netlist has been written in exampl
e_9.4.cir.out
The scilab netlist has been written in example
_9.4.cir.ckt
Press Enter to quit
```

(b) Entering DC source voltage during netlist conversion for Example 9.4

Figure A.32: Various stages in solving Example 9.4 from [1]



```
Model issue on line 1 : .model mos_p pmos( cgso=877.2
unrecognized parameter (tt) - ignored
unrecognized parameter (rg) - ignored
unrecognized parameter (rds) - ignored
unrecognized parameter (l) - ignored
unrecognized parameter (w) - ignored
unrecognized parameter (n) - ignored
Model issue on line 1 : .model mos_n nmos( cgso=877.2
unrecognized parameter (tt) - ignored
unrecognized parameter (rg) - ignored
unrecognized parameter (rds) - ignored
unrecognized parameter (l) - ignored
unrecognized parameter (w) - ignored
unrecognized parameter (n) - ignored
Doing analysis at TEMP = 27.000000 and TNOM = 27.0000


No. of Data Rows : 11
ngspice 1 ->
The file "/tmp/hc4506" may be printed on a postscript
```

Figure A.33: Ngspice simulation results for CMOS inverter Example 9.4, see Footnote 3 on Page 54. Figure shows v(4) and v(3). v(3) is shown with long dashes.

Appendix B

Oscad Resources

In this chapter, we explain the different resources that help promote the use of Oscad amongst the public in general and the academic community in particular.

B.1 SELF Workshops through Spoken Tutorials

As mentioned in Chapter 10, Spoken Tutorials are instructional material created for self learning. We now explain the modalities of Oscad SELF workshops.

- We offer Ubuntu Linux live CD/DVD with Oscad software and Spoken Tutorials on Oscad, KiCad and Ngspice. A description of these Spoken Tutorials and an associated instruction sheet is available in Chapter 10.
- Using the material mentioned above, one can get effective training through SELF workshops of two hour duration.
- A college that wants to organise a SELF workshop should provide the following: (1) A volunteer, who can be a faculty member or a student (2) One computer (desktop or laptop) per participant with audio output capability (3) One head phone per participant - a low cost head phone will do.
- Although not compulsory, one (only one) Skype connection will make the SELF workshops effective as it provides a mechanism for interaction between the participants and the Oscad Team at IIT Bombay.
- There is no lower or upper limit on the number of participants. This just depends on the number of computers available.
- We hope to offer on on-line test and provide Certificates from the Spoken Tutorial Project, IIT Bombay, for participants who pass the test. The on-line test is typically held two weeks after SELF workshops. Every computer should have Internet for On-line tests.
- The current mandate of the funding agency NMEICT is that SELF workshops and on-line certificates are offered free of cost.

- Useful links for conducting SELF workshops are [19], [20] and `SELF-workshop@oscad.in` or `contact@spoken-tutorial.org`.

B.2 Textbook Companion on Oscad

An Oscad Textbook Companion (OTC) has the Oscad code for the solved problems of standard textbooks on circuit design and simulation. We have already created an OTC for [1], which can be downloaded free of cost from [20]. All OTCs have only code and no documentation. For example, to understand the code given in OTC of [1], one needs a copy of [1] and to understand the theory explained therein. We invite students, teachers and professionals to create OTCs for other standard textbooks as well.

The latest list of completed and in-progress OTCs will be available at [20]. An enthusiast who is interested in creating an OTC should code the solved examples of a book that is not completed nor in-progress and upload them on to [20]. If done correctly, the work will be assigned to the proponent.

As per the recommendations of NMEICT, the person who creates an OTC will be given an honorarium of Rs. 10,000 and their teacher shall be given an honorarium of Rs. 5,000. The teacher's honorarium is for reviewing the OTC and for certifying its correctness. Teachers can also create OTCs.

B.3 Other Oscad Activities

Lab Migration: This service is provided for all labs that are interested in shifting their proprietary EDA tool based circuit design labs to Oscad. The interested college could contact us expressing their intent and a statement of their lab experiments. The Oscad Team at IIT Bombay will provide the equivalent Oscad code to help conduct the experiments. The required coding will be done by our team and student/teacher volunteers from colleges. Thanks to NMEICT funds, we are in a position to give honorarium for those who do this coding. This code will be available free of cost from [20] for everyone.

Circuit Creation: We will provide a mechanism for students who do their projects on circuit design using Oscad to upload on to [20]. We will provide forum based help service to clear conceptual questions. We will also provide a ranking and a feedback mechanism for the uploaded circuits.

Expanding Oscad's Capabilities: We invite participation from the public to expand Oscad's capabilities. We are also establishing processes to support student projects in this area. To participate in this activity, one needs IT skills and a knowledge of electronic design. These services will also be coordinated through [20].

Appendix C

Oscad on Windows XP

In this chapter, we explain the procedure to install and use Oscad on Windows XP. At present, because of the difficulties in installing its constituent components, Oscad is not available in other flavours of Windows.

C.1 Installing Oscad on Windows XP

The procedure to install Oscad on Windows is given below in conversational style:

1. Insert the Oscad installer CD for Win XP in the CD/DVD drive of the computer.
2. Browse to the CD contents. Double click on `OSCAD_Setup.exe` file.
3. The window as shown in Fig. C.1a appears. Click on `Next`.
4. Click on `Next` again when the window as shown in Fig. C.1b appears. DO NOT change the `Destination Folder`.
5. Click on `Install` when the window shown in Fig. C.1c appears.
6. Wait for a few seconds. The window shown in Fig. C.1d appears. Click on `OK` for installing KiCad. DO NOT change the installation path. Click `Next/Install` when prompted.
7. Python installation dialog box appears. Click on `OK`. This starts the installation of Python.
8. Select the option `Install for all users` and click on `Next` as shown in Fig. C.1e.
9. DO NOT change the directory for Python 2.7.3 files and click on `Next` as shown in Fig. C.1f.

10. Click `Next` as shown in Fig. C.2a. Wait for a few seconds.
11. Click `Finish` as shown in Fig. C.2b.
12. Now wxPython2.8 will be installed. Click on `OK` as shown in Fig. C.2c.
13. Click `Next` as shown in Fig. C.2d.
14. Choose `I accept the agreement` and click on `Next` as shown in Fig. C.2e.
15. Click `Next` and `Next` again as shown in Figures C.2f and C.3a. DO NOT change the destination location.
16. Uncheck the `View README.win32.txt` and click on `Finish` as shown in Fig. C.3b.
17. A terminal window appears as shown in Fig. C.3c. Nothing needs to be done here. Wait for a few seconds.
18. PIL package for Python will be installed next. Click on `OK` as shown in Fig. C.3d.
19. A new wizard appears for PIL installation. Click `Next/Install` whenever prompted and DO NOT change the installation path.
20. With this the installation of Oscad is complete. Click on `Close` as shown in Fig. C.3e.
21. To launch oscad, go to Start menu. Click on `All programs`, click on `OSCAD` and choose `OSCAD`.

C.2 Procedure to set paths in the environment variable PATH for Windows XP

It is important NOT to change any installation paths during installation. One should keep the default paths.

The user needs to manually set the paths of KiCad, Ngspice and Python in the environment variable *PATH*, if an error of the type `Failed to set the path` is seen towards the end of installation. The paths for the above are given in Table C.1.

The procedure to set the environmental variable *PATH* for various software packages is given below.

1. Right click on `My Computer` from the Desktop and click on `Properties`.
2. In the System Properties window, click on the tab `Advanced`.
3. In the Advanced section, click on the `Environment Variables` button.

Table C.1: Paths for software in Windows XP

Software	Path
Python	`C:\Python27`
KiCad	`C:\Program Files\KiCad\bin`
Ngspice	`C:\OSCAD\spice\bin`

4. Select the variable `Path` under System variables and click on `Edit`.
5. Type the path of the software and end it with a semicolon. Do this for all the software packages (KiCad, Ngspice and Python).

C.3 Special instructions to use Oscad on Windows XP

The following instructions have to be kept in mind while using Oscad. Details of the use of Oscad are explained throughout the rest of the book.

- To load Oscad libraries to the schematic, browse to

  `C:\OSCAD\OSCAD\library.`

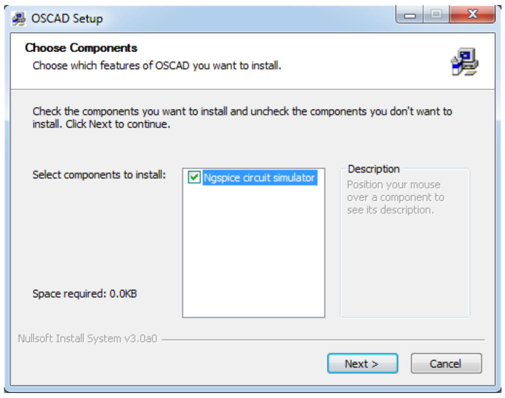
- Oscad Examples can be found in

  `C:\OSCAD\OSCAD\Examples.`

  New and updated examples can be downloaded from [20].

- The Schematic Editor, Layout Editor and Footprint Editor windows might appear minimised, when one opens them. One may maximise them for better view. This wil be helpful to view all the tools in the toolbars.
- To load a MOSFET (N or P) to the schematic, type MOS_N (or MOS_P) after choosing the Place a component tool from the toolbar on the right. The component reference may be 'Q'. Before simulating the circuit, ensure that the component reference has been changed to 'M' (M without quotes). One need not change the reference for PCB design. Refer to the Oscad book to know more about changing component references.
- Check command prompt window for parameter entry during Netlist conversion, subcircuit creation and model building.

Updated information on Oscad is available at [20].


OSCAD Setup

Choose Components

Choose which features of OSCAD you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install: [✓ Ngspice circuit simulator]

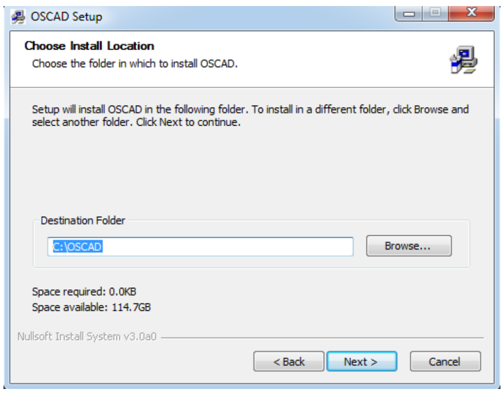
Description

Position your mouse over a component to see its description.

Space required: 0.0KB

Nullsoft Install System v3.0a0

(a) Ngspice Installation. Click on Next


OSCAD Setup

Choose Install Location

Choose the folder in which to install OSCAD.

Setup will install OSCAD in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder

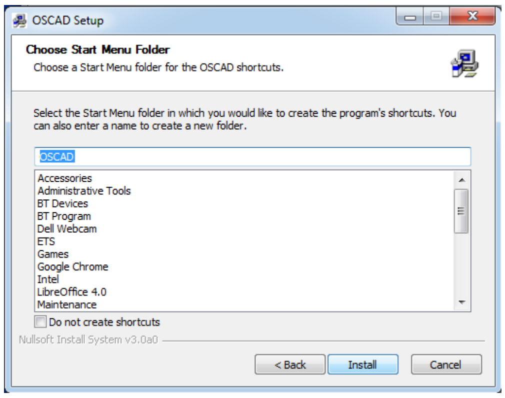
C:\OSCAD

Space required: 0.0KB

Space available: 114.7GB

Nullsoft Install System v3.0a0

(b) Oscad Setup: Click on Next


OSCAD Setup

Choose Start Menu Folder

Choose a Start Menu folder for the OSCAD shortcuts.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

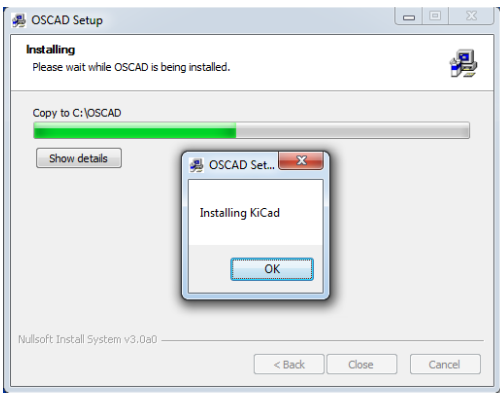
OSCAD

Accessories
Administrative Tools
BT Devices
BT Program
Dell Webcam
ETS
Games
Google Chrome
Intel
LibreOffice 4.0
Maintenance

☐ Do not create shortcuts

Nullsoft Install System v3.0a0

(c) Oscad Setup: Click on Next


OSCAD Setup

Installing

Please wait while OSCAD is being installed.

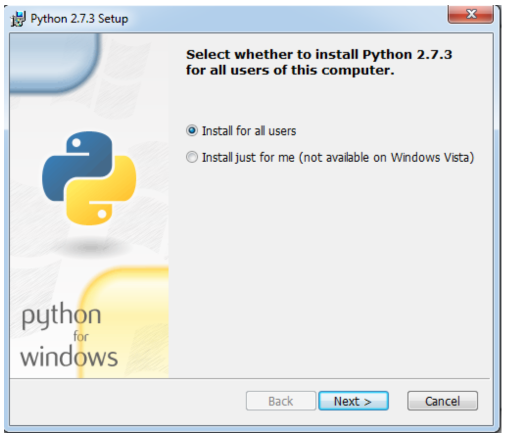
Copy to C:\OSCAD

OSCAD Set...

Installing KiCad

Nullsoft Install System v3.0a0

(d) Installing KiCad. Click on OK

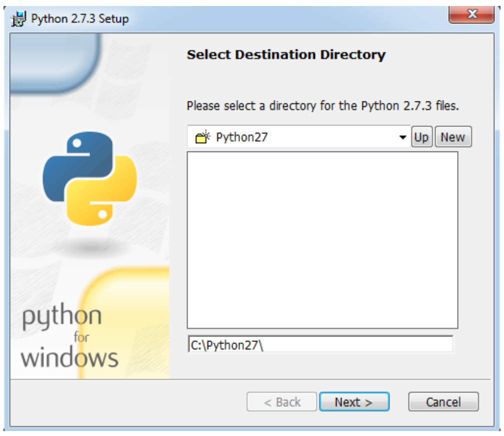

Python 2.7.3 Setup

Select whether to install Python 2.7.3 for all users of this computer.

[✓ Install for all users]

Install just for me (not available on Windows Vista)

(e) Python Setup: Choose Install for all users. Click on Next


Python 2.7.3 Setup

Select Destination Directory

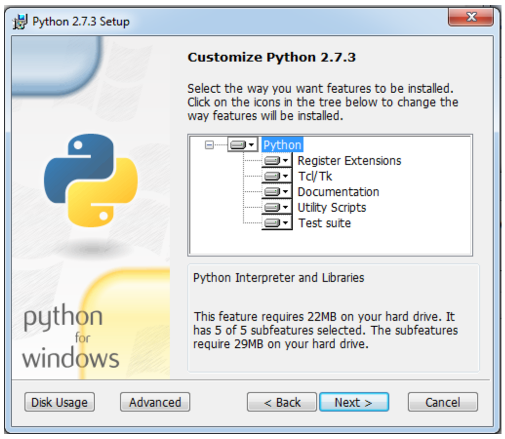
Please select a directory for the Python 2.7.3 files.

Python27

C:\Python27\

(f) Python Setup: Do not change the directory. Click on Next.

Figure C.1: Steps in installing Oscad on Windows


Python 2.7.3 Setup

Customize Python 2.7.3

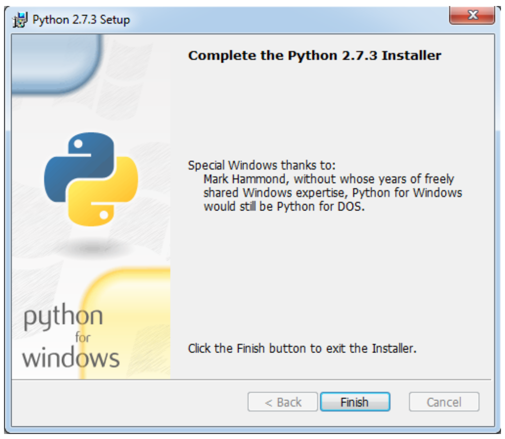
Select the way you want features to be installed.
Click on the icons in the tree below to change the way features will be installed.

Python
- Register Extensions
- Tcl/Tk
- Documentation
- Utility Scripts
- Test suite

Python Interpreter and Libraries

This feature requires 22MB on your hard drive. It has 5 of 5 subfeatures selected. The subfeatures require 29MB on your hard drive.

(a) Python Setup: Click on Next

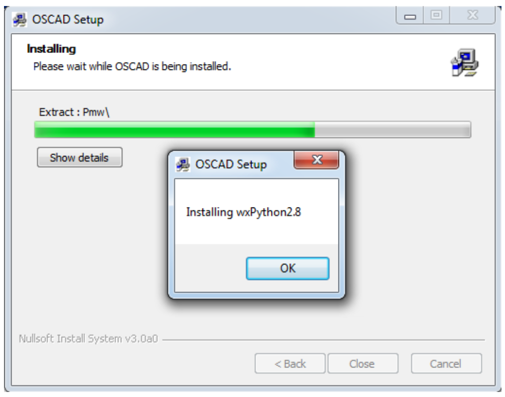

Python 2.7.3 Setup

Complete the Python 2.7.3 Installer

Special Windows thanks to:
Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

Click the Finish button to exit the Installer.

(b) Python Setup: Click on Finish


OSCAD Setup

Installing
Please wait while OSCAD is being installed.

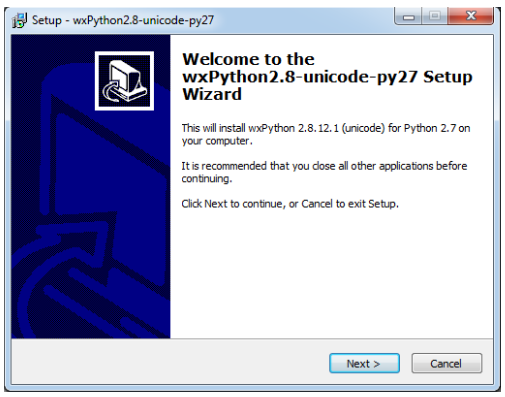
Extract : Pmw\

OSCAD Setup

Installing wxPython2.8

Nullsoft Install System v3.0a0

(c) Installing wxPython2.8. Click on OK


Setup - wxPython2.8-unicode-py27

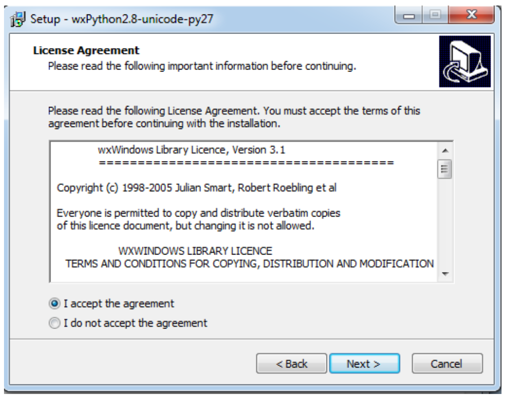
Welcome to the wxPython2.8-unicode-py27 Setup Wizard

This will install wxPython 2.8.12.1 (unicode) for Python 2.7 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

(d) wxPython2.8 Setup: Click on Next


Setup - wxPython2.8-unicode-py27

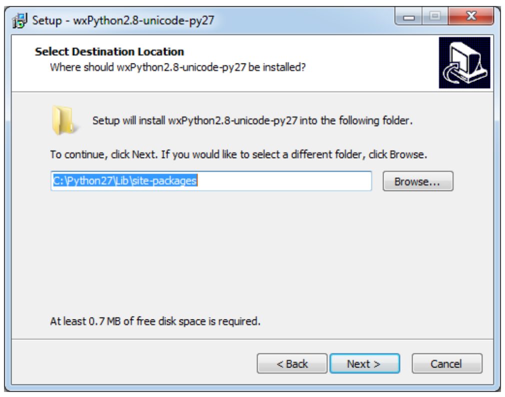
License Agreement
Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

```
          wxWindows Library Licence, Version 3.1
          ======================================

Copyright (c) 1998-2005 Julian Smart, Robert Roebling et al

Everyone is permitted to copy and distribute verbatim copies
of this licence document, but changing it is not allowed.

              WXWINDOWS LIBRARY LICENCE
  TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
```

[✓ I accept the agreement]
I do not accept the agreement

(e) wxPython2.8 Setup: Choose `I accept the agreement` and click on Next


Setup - wxPython2.8-unicode-py27

Select Destination Location
Where should wxPython2.8-unicode-py27 be installed?

Setup will install wxPython2.8-unicode-py27 into the following folder.

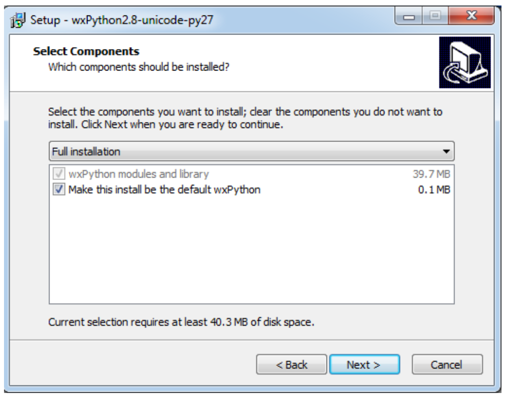
To continue, click Next. If you would like to select a different folder, click Browse.

C:\Python27\Lib\site-packages

At least 0.7 MB of free disk space is required.

(f) wxPython2.8 Setup: Do not change Destination Location. Click on Next

Figure C.2: Steps in installing Oscad in Windows



Setup - wxPython2.8-unicode-py27

Select Components
Which components should be installed?

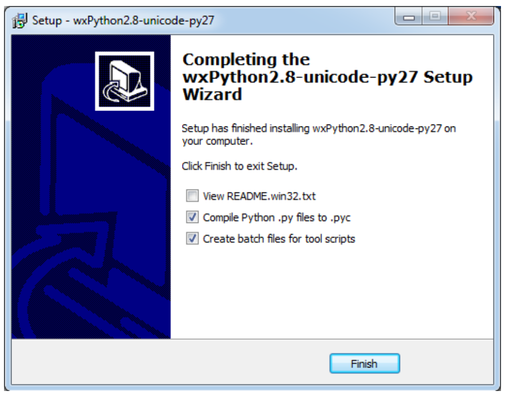
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

Full installation

☑ wxPython modules and library	39.7 MB
☑ Make this install be the default wxPython	0.1 MB

Current selection requires at least 40.3 MB of disk space.

(a) wxPython2.8 Setup: Click on Next



Setup - wxPython2.8-unicode-py27

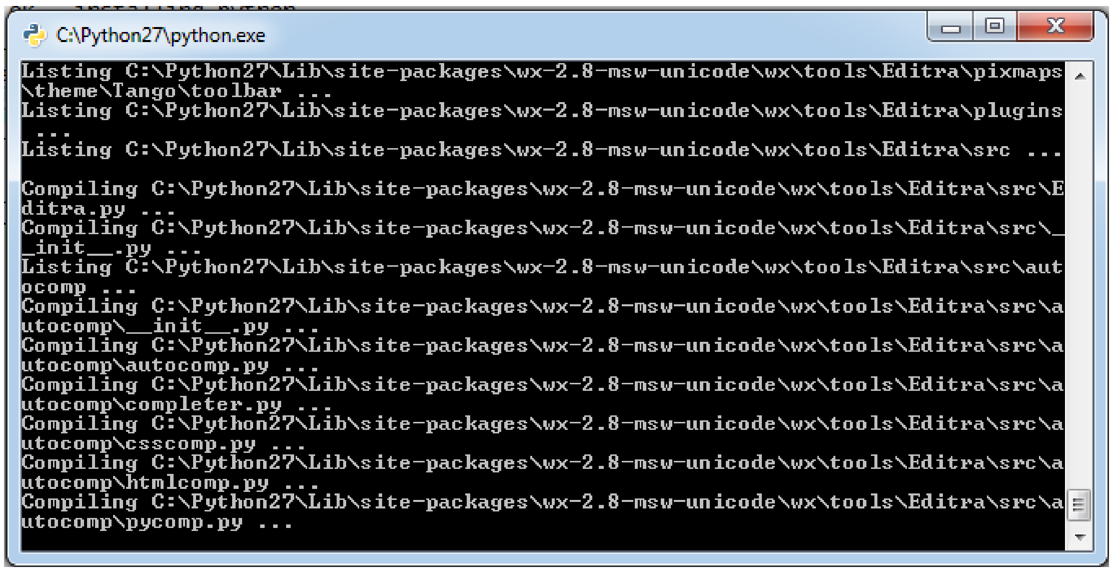
Completing the wxPython2.8-unicode-py27 Setup Wizard

Setup has finished installing wxPython2.8-unicode-py27 on your computer.

Click Finish to exit Setup.

☐ View README.win32.txt
☑ Compile Python .py files to .pyc
☑ Create batch files for tool scripts

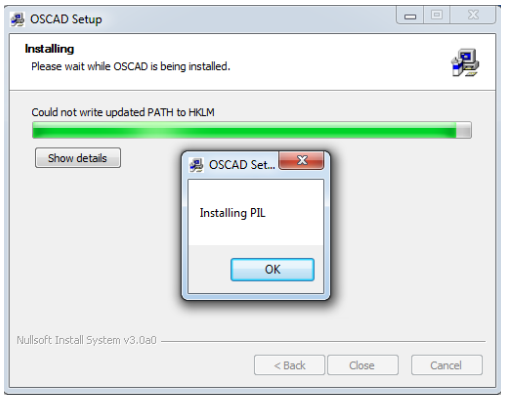
(b) wxPython2.8 Setup: Uncheck the View README option and click on Finish



C:\Python27\python.exe

```
Listing C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\pixmaps
\theme\Tango\toolbar ...
Listing C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\plugins
 ...
Listing C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src ...

Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\E
ditra.py ...
Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\_
_init__.py ...
Listing C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\aut
ocomp ...
Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\a
utocomp\__init__.py ...
Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\a
utocomp\autocomp.py ...
Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\a
utocomp\completer.py ...
Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\a
utocomp\csscomp.py ...
Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\a
utocomp\htmlcomp.py ...
Compiling C:\Python27\Lib\site-packages\wx-2.8-msw-unicode\wx\tools\Editra\src\a
utocomp\pycomp.py ...
```

(c) Terminal window appearing during installation



OSCAD Setup

Installing
Please wait while OSCAD is being installed.

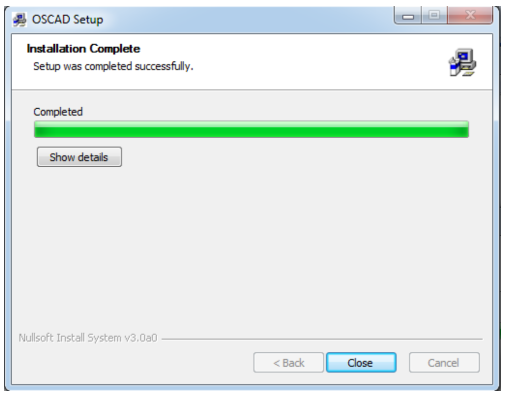
Could not write updated PATH to HKLM

OSCAD Set...

Installing PIL

Nullsoft Install System v3.0a0

(d) Installing PIL: Click on OK



OSCAD Setup

Installation Complete
Setup was completed successfully.

Completed

Nullsoft Install System v3.0a0

(e) Installation Complete: Click on Close

Figure C.3: Steps in installing Oscad

References

[1] A. S. Sedra and K. C. Smith, *Microelectronic Circuits - Theory and Applications*. Oxford University Press, 2009.

[2] K. M. Moudgalya, "Spoken Tutorial: A Collaborative and Scalable Education Technology," *CSI Communications*, vol. 35, no. 6, pp. 10–12, September 2011, available at http://spoken-tutorial.org/CSI.pdf.

[3] (2013, May). [Online]. Available: http://www.scilab.org/

[4] (2013, May). [Online]. Available: http://scilab-test.garudaindia.in/scilab_in/,http://scilab-test.garudaindia.in/cloud

[5] D. B. Phatak. (2013, May) Teach 10,000 teacher programme. [Online]. Available: http://www.it.iitb.ac.in/nmeict/MegaWorkshop.do

[6] K. Kannan and K. Narayanan, "Ict-enabled scalable workshops for engineering college teachers in india," in *Post-Secondary Education and Technology: A Global Perspective on Opportunities and Obstacles to Development (International and Development Education)*, R. Clohey, S. Austin-Li, and J. C. Weldman, Eds. Palgrave Macmillan, 2012.

[7] (2013, May) Teach 10,000 teacher programme on analog electronics. [Online]. Available: http://www.nmeict.iitkgp.ernet.in/Analogmain.htm

[8] (2013, May). [Online]. Available: http://www.aakashlabs.org/

[9] (2013, May). [Online]. Available: http://en.wikipedia.org/wiki/Electronic_design_automation

[10] (2013, May) Synaptic Package Manager Spoken Tutorial. [Online]. Available: http://www.spoken-tutorial.org/list_videos?view=1&foss=Linux&language=English

[11] (2013, May). [Online]. Available: http://www.kicad-pcb.org/display/KICAD/KiCad+EDA+Software+Suite

[12] (2013, May). [Online]. Available: http://ngspice.sourceforge.net/

[13] (2013, May). [Online]. Available: http://scilab.in/

[14] S. M. Sandler and C. Hymowitz, *SPICE Circuit Handbook*. New York: McGraw-Hill Professional, 2006.

[15] J.-P. Charras and F. Tappero. (2013, May). [Online]. Available: http://www.kicad-pcb.org/display/KICAD/KiCad+Documentation

[16] D. Jahshan and P. Hutchinson. (2013, May). [Online]. Available: http://bazaar.launchpad.net/~kicad-developers/kicad/doc/files/head:/doc/tutorials/

[17] P. Nenzi and H. Vogt. (2013) Ngspice users manual version 25plus. [Online]. Available: http://ngspice.sourceforge.net/docs/ngspice-manual.pdf

[18] K. M. Moudgalya, "LaTeX Training through Spoken Tutorials," *TUGboat*, vol. 32, no. 3, pp. 251–257, 2011.

[19] (2013, May). [Online]. Available: http://www.spoken-tutorial.org/

[20] (2013, May). [Online]. Available: http://oscad.in/

Index

Author Information

Yogesh Save is a research member at IBM, SRDC, Bengaluru, India. He received his M.Tech and Ph.D in Electrical Engineering from Indian Institute of Technology Bombay, Mumbai, India.

Rakhi R is an M.Tech student in Systems and Control Engineering at Indian Institute of Technology Bombay, Mumbai, India. She is also a research associate in the Talk to a Teacher project at IIT Bombay. Before joining IIT Bombay, she was a Deputy Engineer at Bharat Electronics Ltd., Chennai. Her research interests are Digital Control, VLSI and Embedded Systems.

Shambhulingayya N.D. is working as a Research Assistant in FOSSEE project, IIT Bombay.

Rupak M. Rokade is a Research Assistant in the Talk to a Teacher project at IIT Bombay. He received his Bachelor's degree in Instrumentation Engineering from Mumbai University. He is one of the developers of Virtual Laboratory for SBHS at IIT Bombay.

Ambikeshwar Srivastava is a Research Assistant in the FOSSEE Project at IIT Bombay. He has recieved Post Graduate Diploma in VLSI from CDAC-ACTS Pune and Bachelor of Technology in Electronics and Communication Engineering from Uttar Pradesh Technical University.

Manas Ranjan Das is an open source developer. He holds a Bachelor of Engineering degree in Electronics and Commmunication Engineering. He is currently working as a Research Associate in the FOSSEE Project at Indian Institute of Technology Bombay. His main interests are open-source FPGA and CAD tools.

Lavitha Pereira is currently working as a Research Associate in FOSSEE Project at Indian Institute of Technology Bombay. Lavitha holds a B.E. degree in Electrical and Electronics from VTU University, India. She is passionate about Free and open source software.

Sachin Patil is currently working as a Linux System Administrator in Indian Institute of Technology Bombay. Apart from System Administration, he has also gained experience in Android and embedded systems. He has ported Scilab, a software for Numerical Computations, on 'Aakash'. He is also interested in customising GNU/linux distributions. Besides Ubuntu, his other favourite GNU/linux distribution is Slackware due to its simplicity and robustness.

Srikant Patnaik is a developer, teacher and motivator. He served as a Lecturer at Loyola academy, Hyderabad and later joined IIT Bombay as a Research Assistant in FOSSEE project. He contributed to the porting of GNU/Linux on Aakash and is also associated with the Android app to run Scilab and other programming languages. His interests include blogging, designing circuits, bridging software and hardware.

Kannan M. Moudgalya is a Professor in Chemical Engineering, Systems and Control and Education Technology at IIT Bombay. He has B.Tech, Ph.D degrees for his work in Chemical Engineering and a Master of Electrical Engineering degree at IIT Madras and Rice University. He is the Principal Investigator (PI) of Spoken Tutorial project and a Co-PI of T10KT and Aakash projects at IIT Bombay, funded by NMEICT, MHRD, Govt. of India. He is a member of the Standing Committee of NMEICT.