IOSCOPY

A viewer for plotting electrical simulation results
User Manual

Arnaud Gardelein

April 25, 2013

Abstract

Oscopy is an interactive oscilloscope written in python designed to simplify the electrical design workflow. It allow to read, view and post-process signals with support for automatic dependency tracking. File re-reading (updates) can be triggered by external applications like gEDA suite through D-Bus messaging system, and then Oscopy can call netlist generator and electrical simulator programs automatically. As oscopy is built on top of IPython, post-processing include as well as simple arithmetics operation as complex functions like FFT. Oscopy can be easily extended to a multi-purpose viewer, as adding new data file formats and new types of plots is really easy.

This document covers the user interface and command description.

Chapter 1

Introduction

In the electrical system design workflow, viewing results from analog simulation or experiment is not a trivial task: there exist numerous different program with even more different file formats, the user interface has to be friendly and functional, and the program should be memory efficient due to the number of data points per file that can quickly grow.

1.1 Rationale

The gEDA suite contains mainly all tools required to design electrical boards, from scheme drawings to PCB routing. There already exist several programs to view analog simulation results: gwave, GSpiceUI, dataplot.

Gwave is designed as a waveform viewer, and can read text files as well as binary files generated by Spice2, Spice3, ngspice, CAzM or gnucap. The user interface supports drag and drop signals onto graphs, vertical bar cursors, multiple files and multiple panes.

GSpiceUI is more focused on the interaction between the user and the simulation program: it imports the schematic from gschem, allows the user to build the file to be used by the simulation and plot the results, eventually using GWave.

Dataplot has support for formats like gnucap, ngspice, hdf5 and touchstone. The user interface has tabs for multiple plots and presents the data in a hierarchical manner.

Another way of viewing results is to use Octave (and generally gnuplot). This approach enables one to post-process the results with operations such as FFT, diff. Support for multiple figures is present. Octave support HDF5 file format and tab-separated text-based files such as gnucap output. The user interaction is essentially based on the command line interface.

The idea behind Oscopy is to combine the best of these approaches into a single, easily extendable program. In this purpose, it has features such as multiple plots, multiple windows, different plot types (linear, log) and allows the user to do math with data, including basic operations, trigonometry, fft, diff. It supports the gnucap file format for input and output, and has an update mechanism to reread data from files. New file formats and new graph types can be added by following the guidelines presented in this document.

1.2 Oscopy and IOscopy: the core and the app

To enable reuse and make it flexible, Oscopy has two main components:

Oscopy the core of it which gather all the framework API about file operations, signals management, figures and graph handling

IOscopy the application based on top of IPython which provide the command line interface

This means that IOscopy is just an application of using Oscopy core. As of today the term *Oscopy* means either Oscopy or IOscopy and when differentiation is needed *oscopy-core* for the framework and either *oscopy-ipython* or *IOscopy* for the IPython application.

The framework API oscopy-core is described in oscopy-api.pdf that is usually installed at the same place of the this manual; typically `/usr/share/doc/oscopy`. The present document is only about the application IOscopy.

1.3 Supported input data formats

Reading data files is performed using Oscopy. Supported formats are:

Gnucap Text format from Gnucap

Cazm Text format from CaZM

Nsout NanoSim format from Synopsis; *Independent variable assumed to be `Time`*

Spice2raw Berkeley Spice2G6 raw format; *Only one dataset per file, `Date` and `Time` fields not processed*

Spice3raw Berkeley Spice3 format; **ascii** *and* **binary** *formats supported, real and complex numbers supported. Only one simulation per file.*

Hspice Hspice format; **ascii** *and* **binary** *formats supported. Only one sweep per file. Auto signals are note returned, only probe signals are. Endianness is not managed in binary mode.*

Touchstone Touchstone s2p and co. from IBIS Open Forum; *Version 1 and 2.0 supported. for version 1 uses the file extension to determine the number of ports ('.snp' where n is 1-4). Noise parameter data is read and stored in self.info['noise_param']. Mixed mode parameters of version 2.0 not supported.*

1.4 Supported output format

Writing data files is performed using Oscopy. Supported formats are:

gnucap Text format from Gnucap

1.5 Supported plots

Available Graphs are:

linear Standard 2D plot with X and Y axis being either linear or logarithmic

Chapter 2

IOscopy: Oscopy on top of IPython

2.1 First steps

2.1.1 Install dependencies

Requested dependencies for building and downloading:

- autoconf
- gettext
- intltool $\geq$ 0.41.0
- git

Requested dependencies for execution:

- python
- python–gtk
- ipython $\geq$ 0.13
- python–numpy
- python–matplotlib
- python–dbus
- python–xdg

Optional dependencies to run demo/demo.oscopy

- geda–gaf and more specifically gschem and gnetlist
- gnucap

Debian

Tested on stable (squeeze) and unstable (sid). Will not install on stable due to default release ipython 0.10.

Install git, autotools and related packages:

```
# apt-get git autoconf gettext intltool python-gtk2-dev
```

Install oscopy dependencies:

```
# apt-get install ipython python-matplotlib
```

Optional: to execute demo.oscopy script install geda and gnucap:

```
# apt-get install geda gnucap
```

Ubuntu

Tested on Ubuntu 12.10 Quantal Quetzal.

Install git, autotools and related packages:

```
$ sudo apt-get install git autoconf gettext intltool python-gtk2-dev
```

Install oscopy dependencies:

```
$ sudo apt-get install ipython python-matplotlib
```

Optional: to execute demo.oscopy script install geda and gnucap:

```
$ sudo apt-get install ipython python-matplotlib
```

Fedora

Tested on Fedora 18.

Install git, autotools and related packages (maybe needed to run the command twice ?):

```
# yum install git autoconf gettext intltool pygtk2-devel gcc
```

Install oscopy dependencies:

```
# yum install ipython dbus-python
```

Optional: to execute demo.oscopy script install geda and gnucap:

```
# yum install geda-gaf gnucap
```

Note that in Fedora 18 default geda-gaf and gnucap appear to be very old version and might not run demo/demo.oscopy correctly.

2.1.2 Download the source

The source can be downloaded from repo.or.cz: http://repo.or.cz/w/oscopy.git

To download it:

```
$ git clone git://repo.or.cz/oscopy.git
```

Optional: to use the experimental branch:

```
$ cd oscopy
$ git pull git://repo.or.cz/oscopy.git experimental
```

2.1.3 Install oscopy

IOscopy shall be compiled and installed on the system:

```
$ cd oscopy
$ ./autogen.sh && ./configure && make install
```

Once installed, PYTHONPATH might need some update, for example if you install it in ${HOME}/geda, to launch oscopy you might need the following line:

```
$ export PYTHONPATH=${HOME}/geda/lib/python2.X/site-packages:$PYTHONPATH
$ export PATH=$PATH:${HOME}/geda/bin
```

To use gschem integration, you will need to use the same `--prefix` that you used to install gschem.

2.1.4 Getting Started

To run the program, just do

```
ioscopy
```

This assume that ioscopy is in your PATH.

A sample circuit is provided for demonstration purposes.

```
$ cd demo
$ ioscopy
```

Just select "File>Run Netlister and simulate..." and fill in the window with the commands provided below:

- For netlister: `gnetlist -g spice-sdb -s -o demo.net demo.sch`
- For simulator: `gnucap -b demo.net`

then in the terminal window:

```
oscopy> oexec demo.oscopy
```

2.1.5 Integration with gschem

To use gschem integration, add the following line to your gschemrc:

```
(load-from-path "oscopy.scm")
```

On the next start of gschem, you should see the oscopy menu. Note that you should have done `./configure --prefix=''same prefix as gschem''` Then assuming you are in the oscopy directory:

- launch gschem
- open demo/demo.sch
- oscopy>>Launch oscopy
- `oscopy> oexec demo.oscopy`

- Once the script finished to execute, go back to gschem and change the value of a component e.g. the capacitor C.
- oscopy>>Update oscopy
- once netlister and simulator ran, figures should have changed, e.g. figure 3 `vout` should have moved and `vo` not.

2.2 IOscopy interfaces

IOscopy has three main interfaces:

The command line which is used to enter commands

The Main Window includes general file I/O, windows list and signal tree

The figures when instantiated

The two former are specific to IOscopy while the latter is a specialisation of Oscopy Figure. The command line is an IPython application with customisation for Signal management. All IOscopy commands are implemented as magic functions.

The Main Window and the figures form the IOscopy GUI and are implemented using GTK libraries.

2.3 Command line options

Command line options are reported in Tab. 2.1.

-b	Execute an oscopy command file
-h	Show this help message
-i	Resume to command line executing after batch file
-q	Disable banner printing

Table 2.1: Command line options

2.4 IOscopy Commands Reference

This section describes the IOscopy commands. Unless otherwise noticed, examples assume that `demo/demo.oscopy` has been run.

oadd SIG [, SIG [, SIG]...]

Add a graph to the current figure. Figure and graph are instanciated if not present.

```
oscopy> oselect 1-1
oscopy> oadd vgs
```

ocreate [SIG [, SIG [, SIG]...]]

Create a new figure, set it as current, add the signals in a first graph.

```
oscopy> ocreate vgs,vds
```

ocontext

Return the Context object used within ioscopy. Use it only if you want to have direct access to internal ioscopy objects.

odelete GRAPH#

Delete a graph from the current figure.

```
oscopy> odelete 1
```

odestroy FIG#

Destroy a figure

```
oscopy> odestroy 3
```

oexec FILENAME

Execute commands from file.

This following example assumes that demo/demo.oscopy has **not** been run.

```
oscopy> oexec demo/demo.oscopy
```

ofactors X, Y

Set the scaling factor of the graph (in powers of ten). Use `auto` for automatic scaling factor. The following example sets the scale factor at 1e-3 for X axis and 10e6 for Y axis

```
oscopy> oselect 1-1
oscopy> ofactor -3, 3
```

ofiglist

Print the list of figures. The layout of the figure is indicated, and the graph mode as well as the Signals are shown. The current figure and graph are marked with a star.

```
oscopy>ofiglist
  Figure 1: horiz
   Graph 1 : (linear) vgs
   Graph 2 : (linear) vsqu
  Figure 2: quad
   Graph 1 : (linear) iRD
   Graph 2 : (linear) vgs
   Graph 3 : (linear) vds vgs
```

```
     Graph 4 : (linear) vds
    Figure 3: horiz
     Graph 1 : (linear) vout vo
    Figure 4: horiz
     Graph 1 : (linear) vsqu
     Graph 2 : (linear) vsqufft
     Graph 3 : (linear) v1
  * Figure 5: horiz
      * Graph 1 : (linear) vs
```

ofreeze SIG [, SIG [, SIG]...]

Do not consider signal for subsequent updates. See also **ounfreeze**.

```
oscopy> ofreeze vout,vds
```

ogui

Show the GUI window if it was closed.

oimport SIG [, SIG [, SIG]...]

Import a list of signals into oscopy to handle dependencies during updates

```
oscopy> pwr=iRD*vds
oscopy> oimport pwr
oscopy> ocreate pwr
oscopy> oupdate  #if iRD or vds changed, pwr will be automatically updated
```

oinsert SIG [, SIG [, SIG]...]

Insert a list of signals into the current graph

```
oscopy> oselect 1-1
oscopy> oinsert vds
```

olayout horiz|vert|quad

Define the layout of the current figure

olayout horiz Graphs are stacked from top to bottom

olayout vert Graphs are side by side from left to right

olayout quad One graph per figure corner

```
oscopy> oselect 2-1
oscopy> olayout horiz
oscopy> olayout vert
oscopy> olayout quad
```

omode MODE

Set the type of the current graph of the current figure
Available modes:

omode lin Linear graph

orange [x|y min max]|[xmin xmax ymin ymax]|[reset]

Set the axis range of the current graph of the current figure

orange x xmin xmax set x axis range

orange y ymin ymax set y axis range

orange xmin xmax ymin ymax set both axis range

```
oscopy> oselect 1-1
oscopy> orange x 0 1
oscopy> orange y -4 12
oscopy> orange 0.5 0.6 -2 2
```

oread DATAFILE

Read signal file

```
oscopy> oread demo/tran.dat
```

orefresh FIG#|current|all|on|off

Force/toggle autorefresh of current/#/all figures on update

orefresh FIG# refresh figure #

orefresh current refresh current figure

orefresh all refresh all figures

orefresh on turn on autorefresh on Signal updates

orefresh off turn off autorefresh on Signal updates

```
oscopy> orefresh 3
oscopy> orefresh on
```

oremove SIG [, SIG [, SIG]...]

Delete a list of signals into from current graph

```
oscopy> oselect oselect 2-3
oscopy> oremove vds
```

oscale [lin|logx|logy|loglog]

Set the axis scale

oscale lin Set linear scale on both axis

oscale logx Set log scale on x axis and linear scale on y axis

oscale logy Set linear scale on x axis and log scale on y axis

oscale loglog Set log scale on both axis

```
oscopy> oselect 3-1
oscopy> oscale logx
oscopy> oscale loglog
```

oselect FIG#-GRAPH#

Select the current figure and the current graph

```
oscopy> oselect 2-1
```

osiglist

List loaded signals

```
oscopy> siglist
Name Unit    Ref     Reader  Last updated (sec)
vds  V       Time    demo/irf540.dat 128
iRD  A       Time    demo/irf540.dat 128
vgs  V       Time    demo/irf540.dat 128
vsqu V       Time    demo/tran.dat   124
v1   None    Time    v1=((vsqu * 3) + 10)    123
vout V       Freq    demo/ac.dat     126
vs   None    Time    vs=sin((Time * 1000000.0))      121
vs2  V       Time    vs2=sin((Time * 1000000.0))     121
vo   V       Freq    vo=vout 121
vsqufft              Frequency       vsqufft=fft(vsqu)       124
```

ounfreeze SIG [, SIG [, SIG]...]

Consider signal for subsequent updates. See also **ofreeze**

```
oscopy> ounfreeze vout,vds
```

ounit [XUNIT,] YUNIT

Set the unit to be displayed on graph axis

ounit XUNIT, YUNIT Set both axis unit

ounit YUNIT Set Y axis unit

```
oscopy> oselect 3-1
oscopy> ounit W      # Set Y axis unit
oscopy> ounit /s,W   # Set both axis unit
oscopy> ounit Hz, V
```

oupdate

Reread data files.

```
oscopy> oupdate
```

owrite format [(OPTIONS)] FILE SIG [, SIG [, SIG]...]

Write signals to file. Options common to all formats are:

ow If `True` or 1 overwrite existing file

This example writes signals v1 and vsqu to the file `demo/res.dat` using format gnucap format, overwriting the file if it already exists.

```
oscopy> owrite gnucap (ow:1) demo/res.dat v1,vsqu
```

Chapter 3

Oscopy GUI

The Graphical User Interface of oscopy is composed of several windows (Figure 3.1):

- The command line terminal
- The main window
- The (many) figures windows

The next paragraphs describe the two last types.

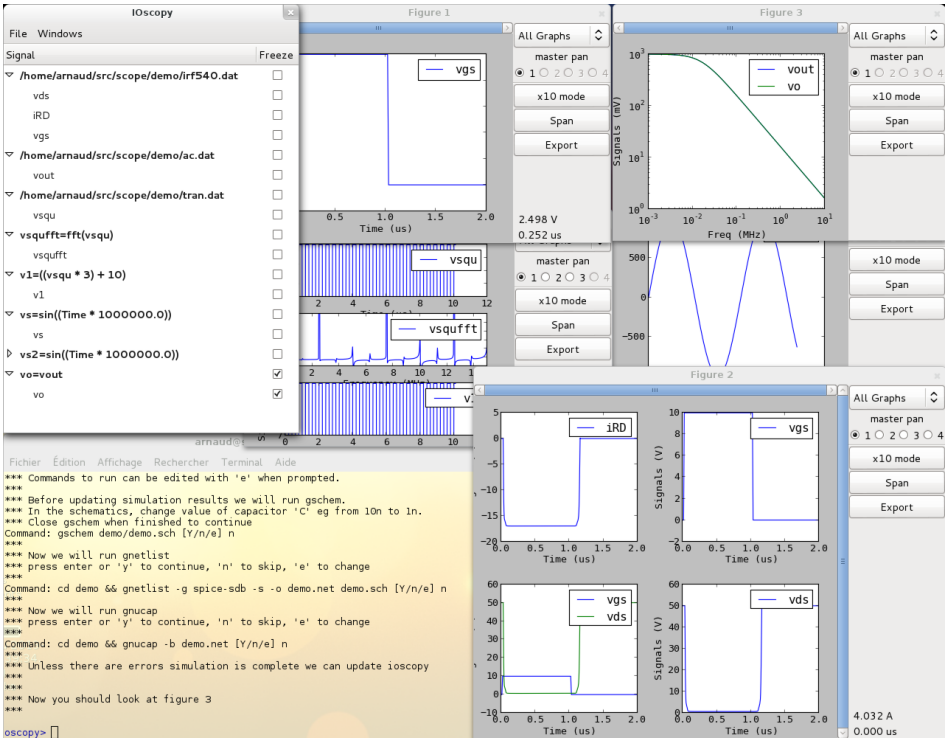

Figure 3.1: ioscopy after running `demo/demo.oscopy`

3.1 The main window

This is the first window that you will see when starting ioscopy. At anytime, it can be shown again by calling the command `ogui` from the terminal if closed.

It contains the list of Readers and their Signals currently handled by ioscopy. Double–clicking on a Signal inserts it in a new Figure.

Each Signal 'freeze' status can be toggled using the checkbox located in the right column. Toggling the checkbox for a Reader set the status for all the signals contained in the Reader.

The 'File' menu:

Add file(s)... To read Signals from file(s)

Update To read Signals from file(s) again

Execute script... To read ioscopy commands from file

New Math Signal To compute a new Signal from existing ones

Run netlister and simulate To generate the netlist, run the simulator and eventually update the Signals (Figure 3.5)

Quit Exit ioscopy

The 'Windows' menu contains the list of the windows, and select one to show it.
Available mouse operations:

left Select signal

double left Insert signal in new Figure

right Show 'Insert signal' menu

3.2 The Figure windows

Each Figure window is composed of two parts:

- On the top part, a zone containing up to 4 graphs
- On the bottom part, the Matplotlib toolbar
- On the right part, the Operation bar

A contextual menu is available for each graph, raised by a right-click on the mouse button. Access to most of the ioscopy functionality is possible through this menu:

- Add/delete graph
- Layout (Figure 3.7)
- Range settings (Figure 3.11)
- Unit settings (Figure 3.12)

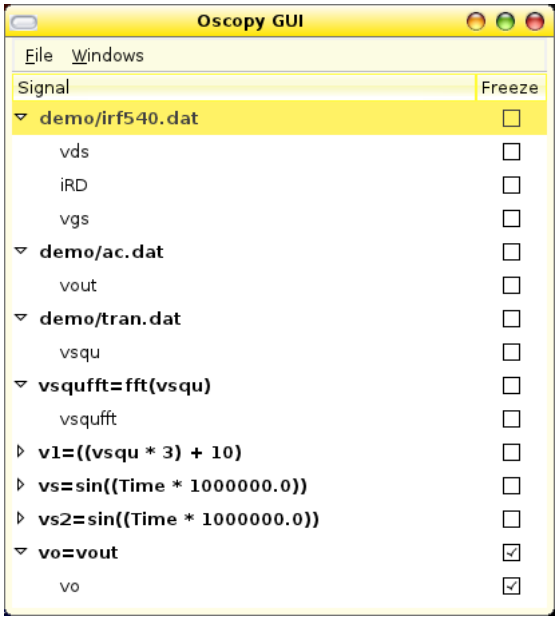

Figure 3.2: The main window

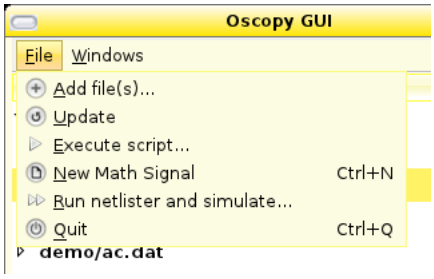

Figure 3.3: The 'File' menu

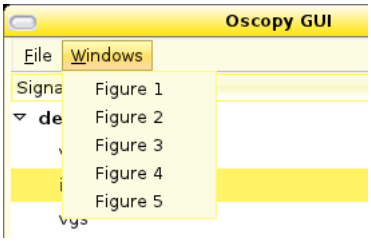

Figure 3.4: The 'Window' menu

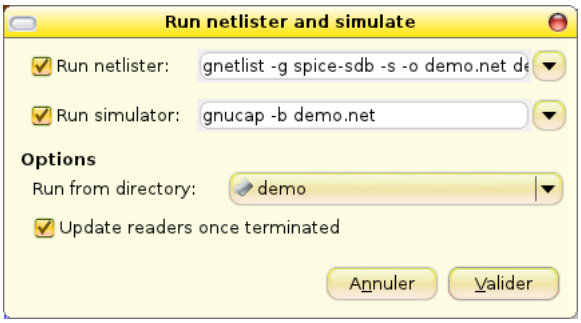

Figure 3.5: The 'Run netlister and simulate' window. Call third party programs to generate oscopy input files. Can be toggled with the checkboxes. Setting are saved in .config/oscopy/gui file

- Scale (Figure 3.8)
- Remove Signal (Figure 3.10)

For each Graph, cursors are available through keys (Figure 3.13):

- '1' for first vertical cursor
- '2' for second vertical cursor
- '3' for first horizontal cursor
- '4' for second horizontal cursor

The value of each cursor is displayed on the bottom part of the graph, and the difference when both cursors are activated.
Available mouse operations:

left In a graph and when 'span' is selected, define region for zooming

roll In a graph, Zoom in / Zoom out

right Show 'Contextual signal' menu. Sensitive to position, e.g. Graph-related items are only available when mouse is over a Graph.

3.2.1 The Operation Bar

This bar is used to set options for all the Graphs at the same time or for a particular Graph. The target Graph is selected through the top-most combo-box.

Master pan When scroll bars are moved, move also all Graphs with same X unit than selected.

Zoom x10 When enabled, the graph range is expanded at a x10 scale, keeping the center of the view. When disabled, return to full range. E.g. if the initial range is [6 .. 11] and the data range is [0 .. 20], the x10 range will be [7.5 .. 9.5] and will be [0 .. 20] on disabling.

Span When enabled, use mouse–left to zoom in a region of a Graph (Figure 3.14). Region selection is made upon the Figure layout configuration:

horizontal Region is selected on X axis

vertical Region is selected on Y axis

quad Region is selected on both axis

Export Save figure as image

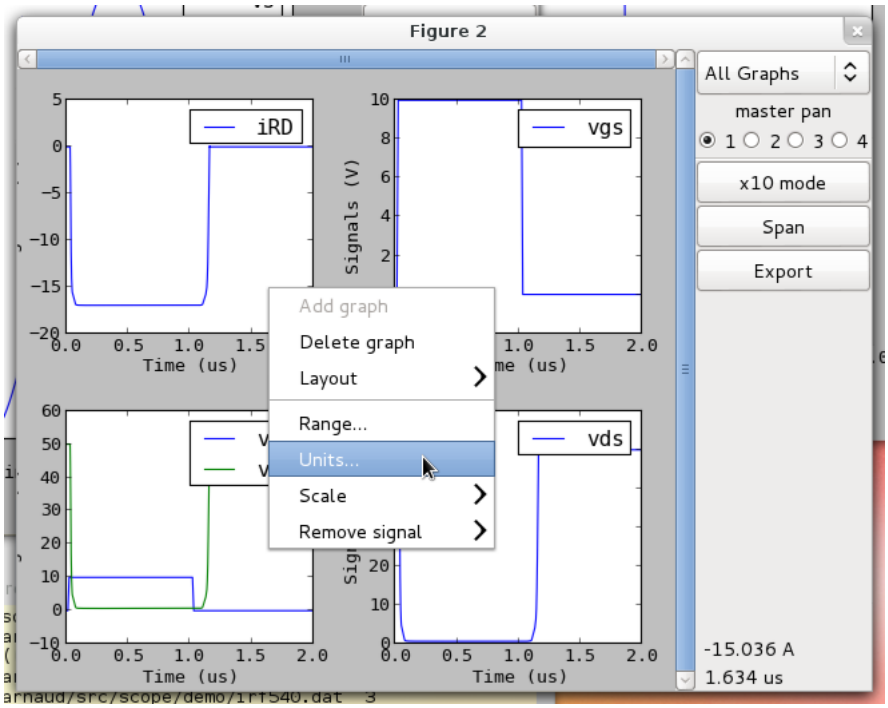

Figure 3.6: A Figure window with the contextual menu.

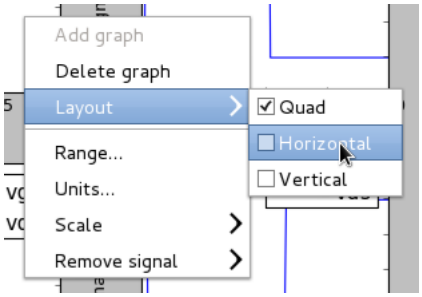

Figure 3.7: Layout configuration menu

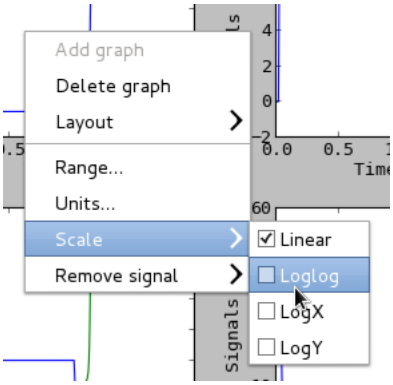

Figure 3.8: Scale configuration menu

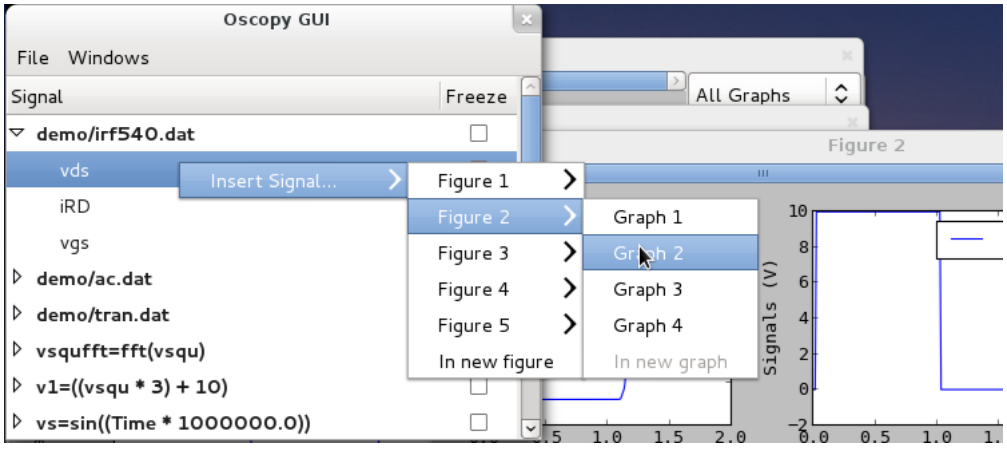

Figure 3.9: Insert signal menu

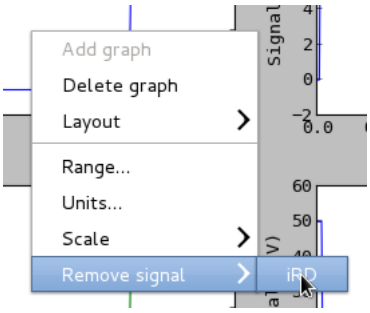

Figure 3.10: Remove signal menu

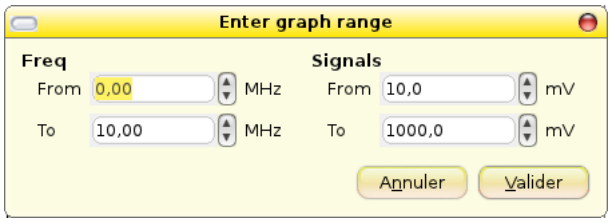

Figure 3.11: Dialogue to set graph range

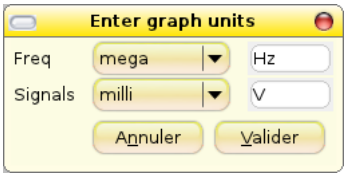

Figure 3.12: Dialogue to set graph units

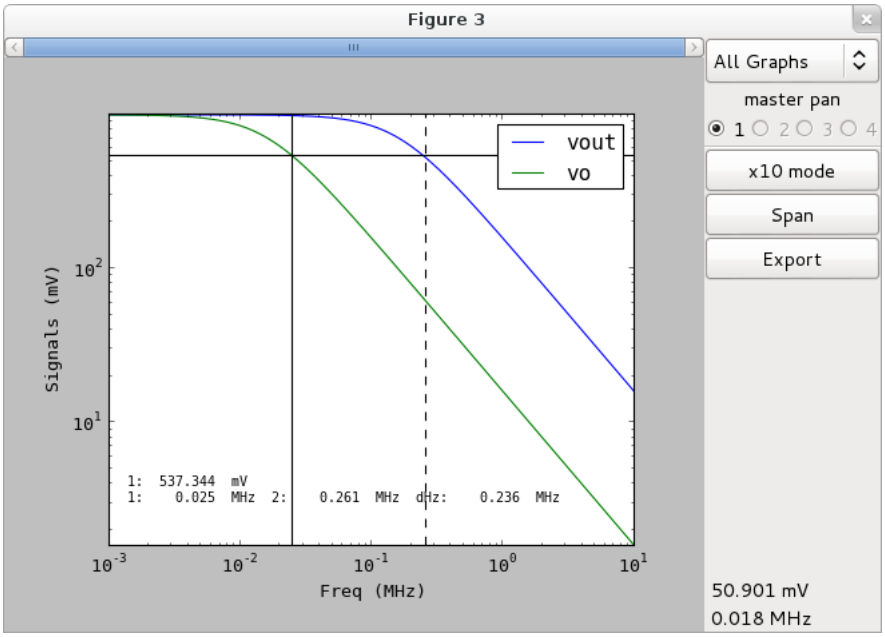

Figure 3.13: Figure with cursors set (after changing capacitance value of `demo.sch` and running netlister and simulator

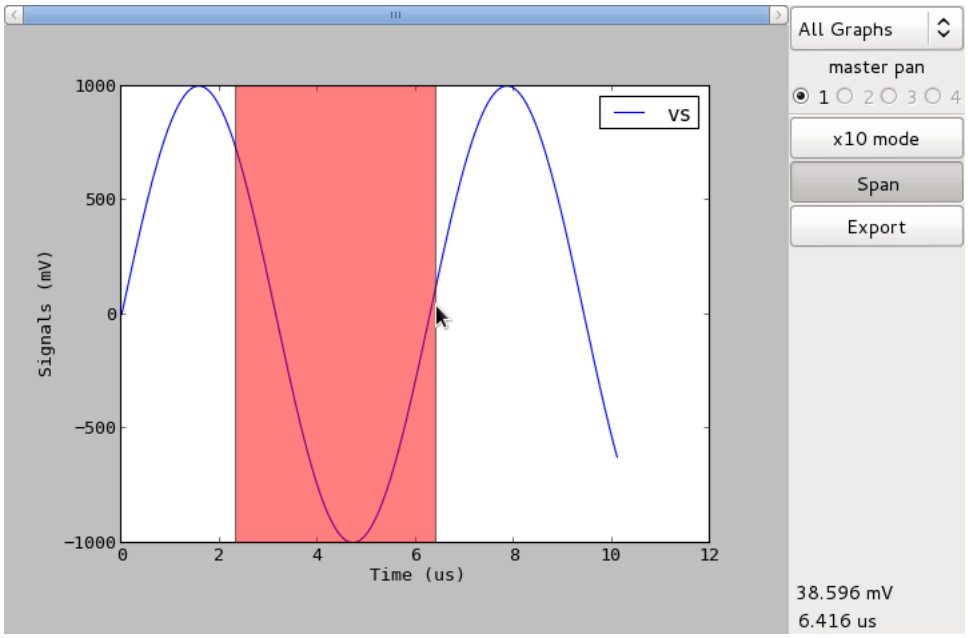

Figure 3.14: Select region for zooming. Here with horizontal layout.

Appendix A

IOscopy commands quick reference guide

Command	Arguments	Description
oadd	SIG [, SIG [, SIG]...]	Add a graph to the current figure
ocontext		Return the Context object used within ioscopy. Use it only if you want to have direct access to internal ioscopy objects.
ocreate	[SIG [, SIG [, SIG]...]]	Create a new figure, set it as current, add the signals
odelete	GRAPH#	Delete a graph from the current figure
odestroy	FIG#	Destroy a figure
oexec	FILENAME	execute commands from file
ofactors	X, Y	set the scaling factor of the graph (in power of ten) use 'auto' for automatic scaling factor e.g. factor -3, 6 set the scale factor at 1e-3 and 10e6
ofiglist		Print the list of figures
ofreeze	SIG [, SIG [, SIG]...]	Do not consider signal for subsequent updates
ogui		Show the GUI
oimport	SIG [, SIG [, SIG]...]	Import a list of signals into oscopy to handle dependencies during updates
oinsert	SIG [, SIG [, SIG]...]	Insert a list of signals into the current graph
olayout	horiz — vert — quad	Define the layout of the current figure
omode	MODE	Set the type of the current graph of the current figure Available modes : lin Linear graph
orange	[x — y min max] — [xmin xmax ymin ymax]	Set the axis range of the current graph of the current figure
oread	DATAFILE	read signal file
orefresh	FIG# — on — off — current — all	Force/toggle autorefresh of current/#/all figures on update
oremove	SIG [, SIG [, SIG]...]	Delete a list of signals into from current graph
oscale	[lin — logx — logy — loglog]	Set the axis scale
oselect	FIG#-GRAPH#	Select the current figure and the current graph
osiglist		List loaded signals
ounfreeze	SIG [, SIG [, SIG]...]	Consider signal for subsequent updates
ounit	[XUNIT,] YUNIT	Set the unit to be displayed on graph axis
oupdate		Reread data files
owrite	format [(OPTIONS)] FILE SIG [, SIG [, SIG]...]	Write signals to file

Table A.1: Summary of IOscopy commands