IOSCOPY
An interactive program for viewing electrical simulation results User Manual

Arnaud Gardelein

June 11, 2011

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.

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.

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, an can read text file as well as binary file generated by Spice2, Spice3, ngspice, CAzM or gnucap. The user interface present features such as drag and drop signal into the graphs, vertical bar cursors, support for multiple files and multiples panels.

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

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

Another way of viewing results is to use Octave (and generally gnuplot). This approach permit to post-process the results with operation 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 command line interface.

The idea behind Oscopy is to combine the better of those approaches into a single program easily extendable. In this purpose, it present features like multiple plots, multiple windows, different plot types (linear, log) and allow the user to do math with data, including basic operations, trigonometry, fft, diff. It support 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.

2 IOscopy: Oscopy on top of IPython

2.1 Commands

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 assume that demo/demo.oscopy has **not** been run.

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

ofactors X, Y
Set the scaling factor of the graph (in power of ten). Use auto for automatic scaling factor. The following example set 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 Example:

```
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.

This example write Signals v1 and vsqu in file `demo/res.dat` using format gnucap format, overwrite file if already existing.

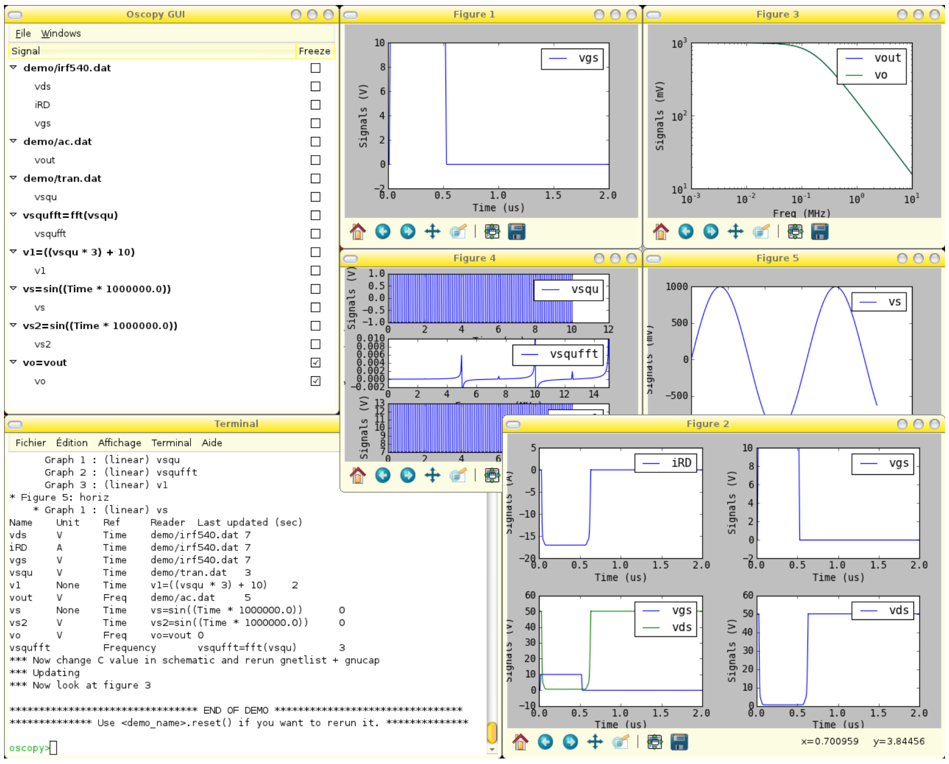
```
oscopy> owrite gnucap (ow:1) demo/res.dat v1,vsqu
```

3 Oscopy GUI

The Graphical User Interface of oscopy is composed of several windows (Figure 1):

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

The next paragraphs describe the two last types.



Signal	Freeze
demo/irf540.dat	☐
vds	☐
iRD	☐
vgs	☐
demo/ac.dat	☐
vout	☐
demo/tran.dat	☐
vsqu	☐
vsqufft=fft(vsqu)	☐
vsqufft	☐
v1=((vsqu * 3) + 10)	☐
v1	☐
vs=sin((Time * 1000000.0))	☐
vs	☐
vs2=sin((Time * 1000000.0))	☐
vs2	☐
vo=vout	☑
vo	☑

```
      Graph 1 : (linear) vsqu
      Graph 2 : (linear) vsqufft
      Graph 3 : (linear) v1
* Figure 5: horiz
    * Graph 1 : (linear) vs
Name    Unit    Ref     Reader  Last updated (sec)
vds     V       Time    demo/irf540.dat 7
iRD     A       Time    demo/irf540.dat 7
vgs     V       Time    demo/irf540.dat 7
vsqu    V       Time    demo/tran.dat   3
v1      None    Time    v1=((vsqu * 3) + 10)    2
vout    V       Freq    demo/ac.dat     5
vs      None    Time    vs=sin((Time * 1000000.0))      0
vs2     V       Time    vs2=sin((Time * 1000000.0))     0
vo      V       Freq    vo=vout 0
vsqufft         Frequency       vsqufft=fft(vsqu)       3
*** Now change C value in schematic and rerun gnetlist + gnucap
*** Updating
*** Now look at figure 3

******************************** END OF DEMO ********************************
************** Use <demo_name>.reset() if you want to rerun it. **************

oscopy>
```

Figure 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 insert 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 5)

Quit Exit ioscopy

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

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

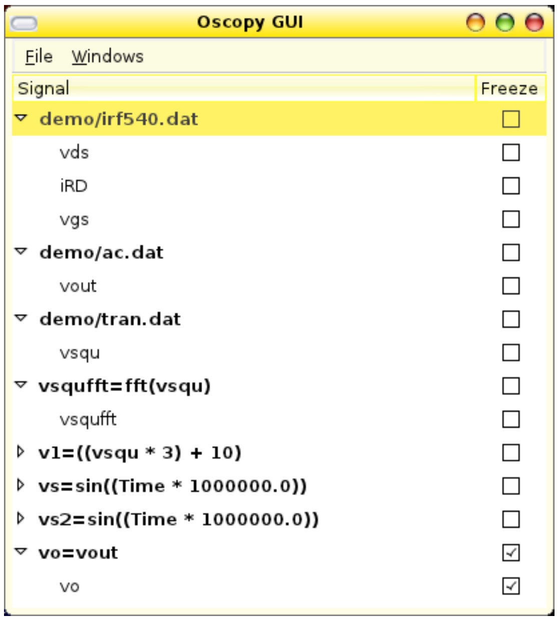
A contextual menu is available for each graph, raised by a right-click on the mouse button. Access to most of the ioscopy functionality is possible through this menu:

- Add/delete graph
- Layout (Figure 7)
- Range settings (Figure 11)
- Unit settings (Figure 12)
- Scale (Figure 8)
- Insert Signal (Figure 9)
- Remove Signal (Figure 10)

For each Graph, cursors are available through keys (Figure 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.



Signal	Freeze
demo/irf540.dat	☐
vds	☐
iRD	☐
vgs	☐
demo/ac.dat	☐
vout	☐
demo/tran.dat	☐
vsqu	☐
vsqufft=fft(vsqu)	☐
vsqufft	☐
v1=((vsqu * 3) + 10)	☐
vs=sin((Time * 1000000.0))	☐
vs2=sin((Time * 1000000.0))	☐
vo=vout	✓
vo	✓

Figure 2: The main window

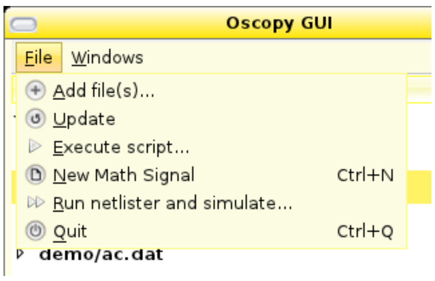


Figure 3: The 'File' menu

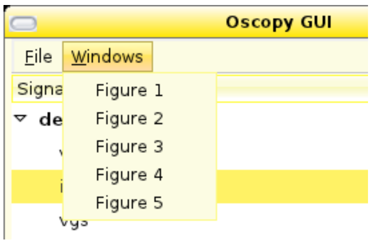
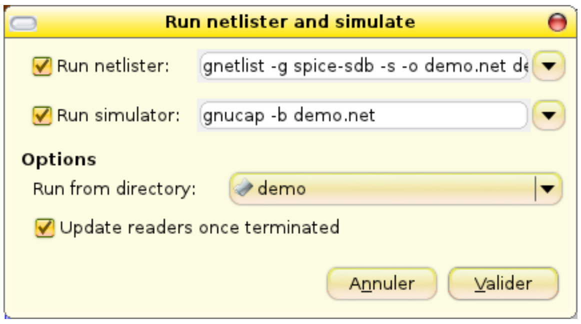


Figure 4: The 'Window' menu



✓ Run netlister: gnetlist -g spice-sdb -s -o demo.net de

✓ Run simulator: gnucap -b demo.net

Options

Run from directory: demo

✓ Update readers once terminated

Figure 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

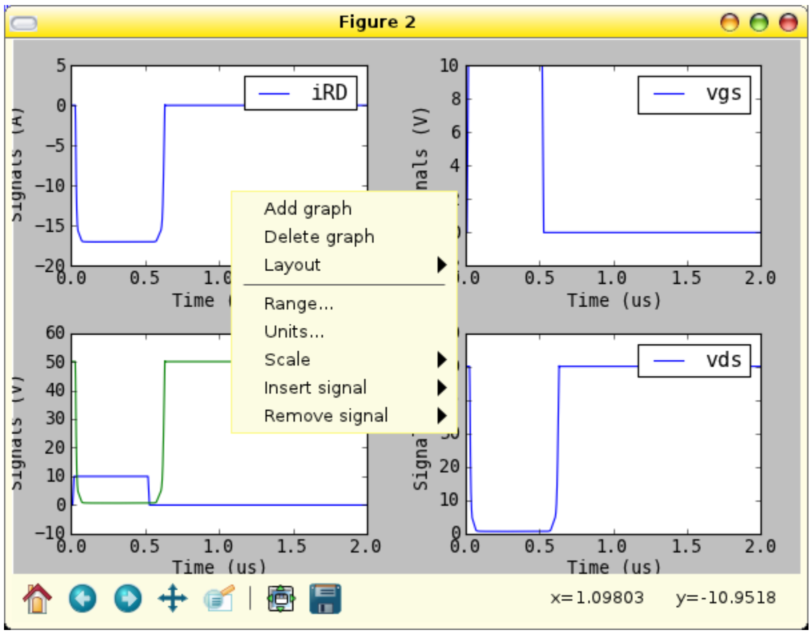


Figure 6: A Figure window with the contextual menu.

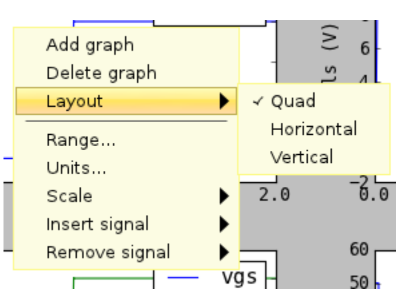


Figure 7: Layout configuration menu

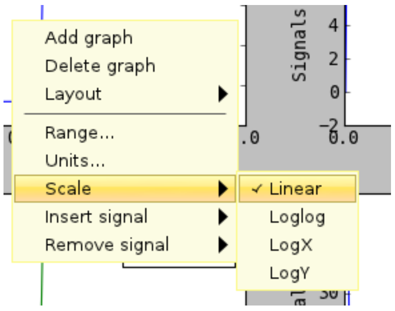


Figure 8: Scale configuration menu

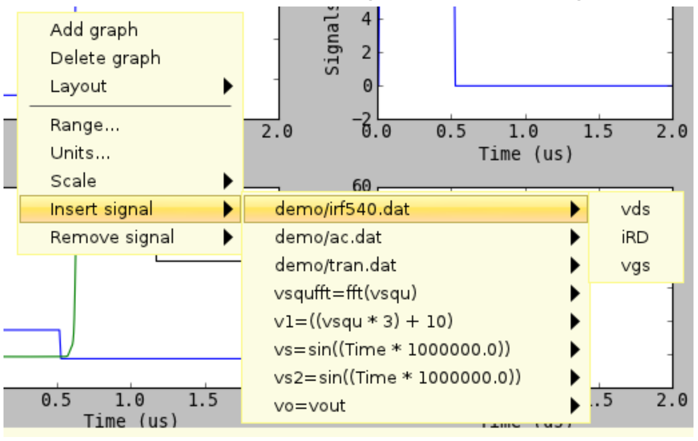


Figure 9: Insert signal menu

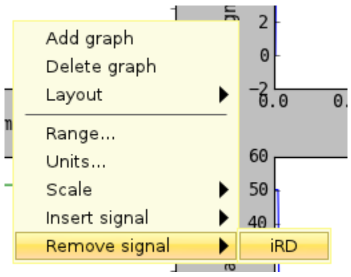
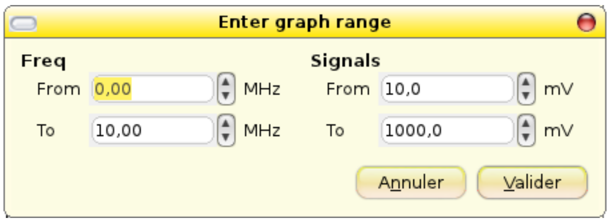


Figure 10: Remove signal menu

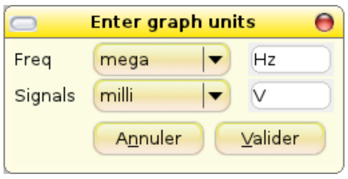


Enter graph range

Freq			**Signals**		
From	0,00	MHz	From	10,0	mV
To	10,00	MHz	To	1000,0	mV

Annuler Valider

Figure 11: Dialogue to set graph range

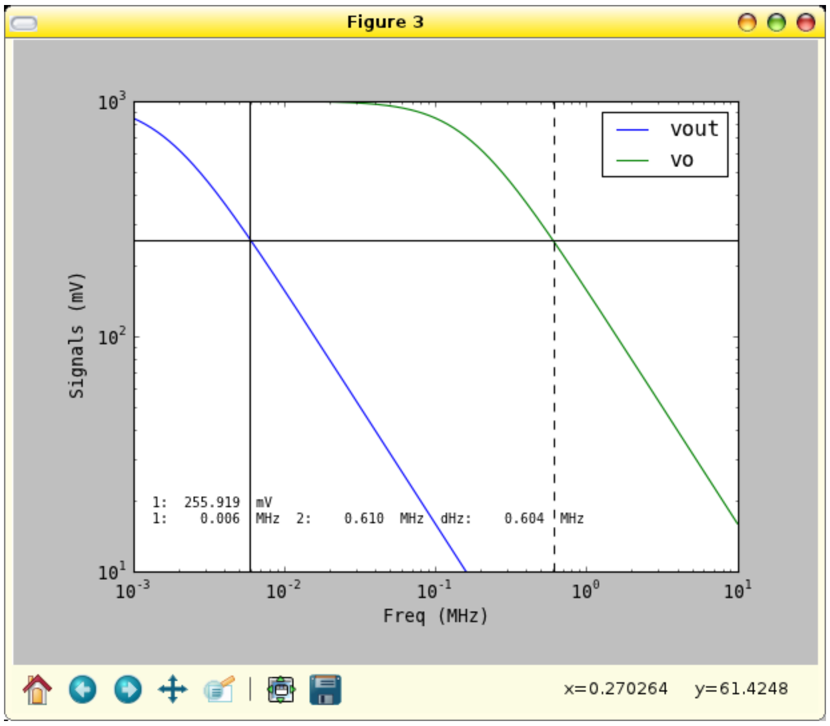


Enter graph units

Freq	mega	Hz
Signals	milli	V

Annuler Valider

Figure 12: Dialogue to set graph units



```
1:  255.919  mV
1:    0.006  MHz  2:    0.610  MHz  dHz:    0.604  MHz
```

Figure 13: Figure with cursors set (after changing capacitance value of `demo.sch` and running netlister and simulator