sphinxcontrib-gtkwave Documentation

Release 0.0.3

ponty

February 07, 2012

CONTENTS

1 About **2**
- 1.1 Features . . . 2
- 1.2 Known problems . . . 2
- 1.3 Basic usage . . . 2
- 1.4 How it works . . . 2
- 1.5 Installation . . . 3

2 Usage **4**
- 2.1 Configuration . . . 4
- 2.2 Basic . . . 4
- 2.3 waiting . . . 4
- 2.4 Options . . . 4
- 2.5 Image options . . . 5

3 Development **7**
- 3.1 Tools . . . 7
- 3.2 Install on ubuntu . . . 7
- 3.3 Tasks . . . 7

sphinxcontrib-gtkwave

Date February 07, 2012

PDF sphinxcontrib-gtkwave.pdf

CHAPTER

ONE

ABOUT

This Sphinx 1.0 extension executes GTKWave during the build step and includes its screenshot into the documentation. GTKWave can display wave files like VCD (value change dump).

Links:

- home: https://github.com/ponty/sphinxcontrib-gtkwave
- documentation: http://ponty.github.com/sphinxcontrib-gtkwave

1.1 Features

- development on linux

1.2 Known problems

- Python 3 is not supported
- PDF output is not perfect
- no unittests

1.3 Basic usage

```
.. gtkwave:: docs/gtkwave_output.vcd
```

1.4 How it works

This is a workaround, there is no image export in GTKWave

1. start Xvfb headless X server using pyvirtualdisplay
2. redirect GTKWave display to Xvfb server by setting $DISPLAY variable.
3. start GTKWave with VCD file. Options are set on command-line and in temporary rc file
4. temporary tcl script will set time interval and select all signals
5. wait until GTKWave is displayed
6. take screenshot by pyscreenshot which needs scrot.
7. image is processed: toolbar, scrollbar and empty space are removed
8. use `.. image::` directive to display image

1.5 Installation

1.5.1 General

- install GTKWave
- install Xvfb and Xephyr
- install PIL
- install scrot
- install pip
- install the program:

```
# as root
pip install sphinxcontrib-gtkwave
```

1.5.2 Ubuntu

```
sudo apt-get install gtkwave
sudo apt-get install python-pip
sudo apt-get install scrot
sudo apt-get install xvfb
sudo apt-get install xserver-xephyr
sudo apt-get install python-imaging
sudo pip install sphinxcontrib-gtkwave
```

1.5.3 Uninstall

```
# as root
pip uninstall sphinxcontrib-gtkwave
```

CHAPTER

TWO

USAGE

2.1 Configuration

Add `sphinxcontrib.gtkwave` to `extensions` list in `conf.py`:

```
extensions = [
              'sphinxcontrib.gtkwave',
              ]
```

2.2 Basic

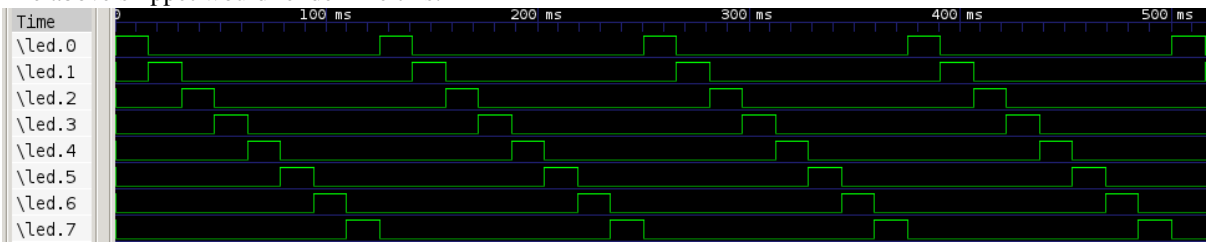
The main directive is *gtkwave*:

```
.. directive:: gtkwave
```

This directive accepts a single string as argument, which is file path to input file:

```
.. gtkwave:: docs/gtkwave_output.vcd
```

The above snippet would render like this:



2.3 waiting

The program is waiting until something is displayed. If nothing happens (e.g. missing gtkwave), after timeout (:timeout:) assertion is raised.

2.4 Options

2.4.1 timeout

If nothing happens, after timeout (default 12 sec) exception is raised, you can change it with this option:

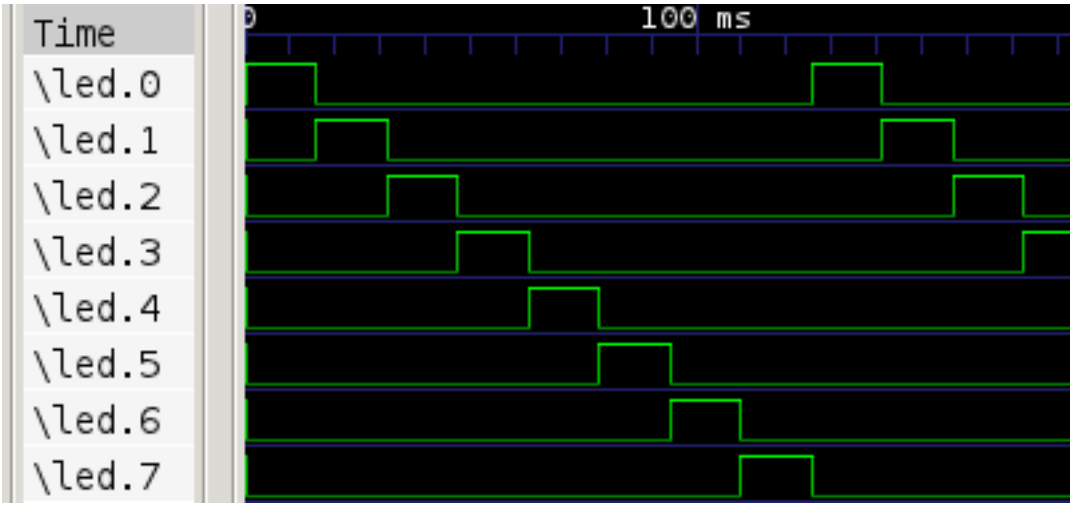
```
.. gtkwave:: docs/gtkwave_output.vcd
      :timeout:  120
```

2.4.2 screen

Using the option `screen` you can set the screen size, default is 1024x768, scrollbar and toolbar is removed from image:

```
.. gtkwave:: docs/gtkwave_output.vcd
      :screen:   400x400
```

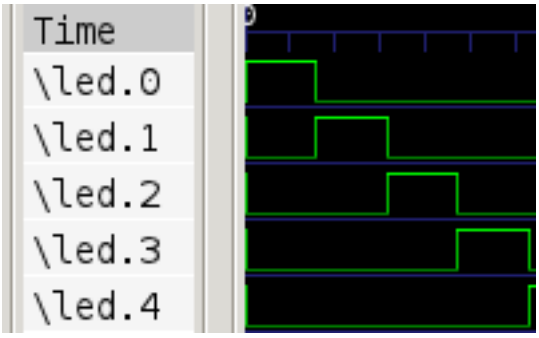
The above snippet would render like this:



Other resolution:

```
.. gtkwave:: docs/gtkwave_output.vcd
      :screen:   100x100
```

The above snippet would render like this:



2.5 Image options

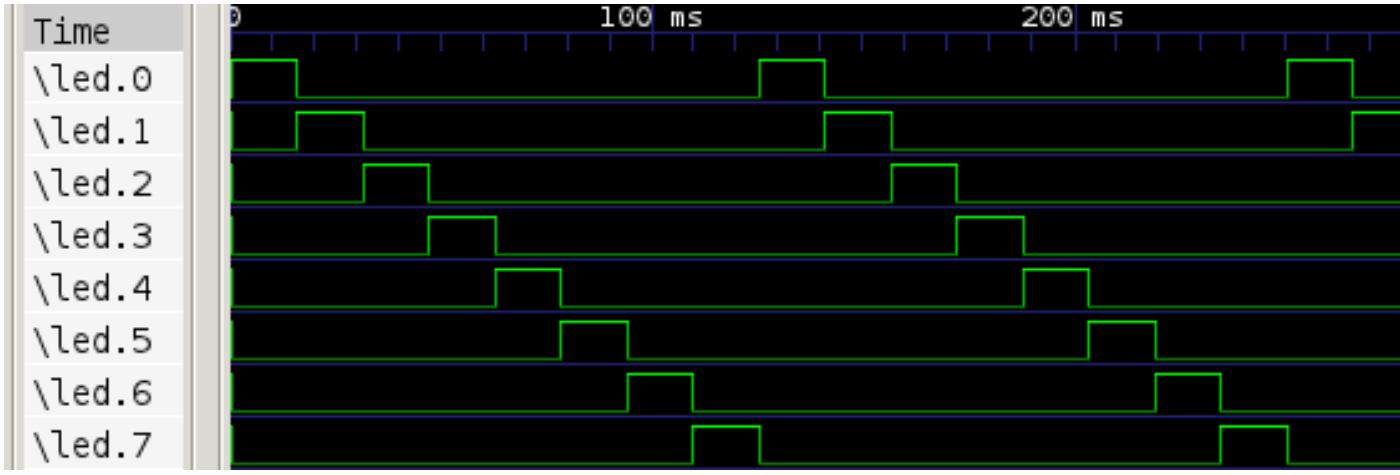
Same as in http://docutils.sourceforge.net/docs/ref/rst/directives.html#image

2.5.1 scale, alt

Example:

```
.. gtkwave:: docs/gtkwave_output.vcd
     :scale: 200 %
     :alt: alternate text
```

The above snippet would render like this:

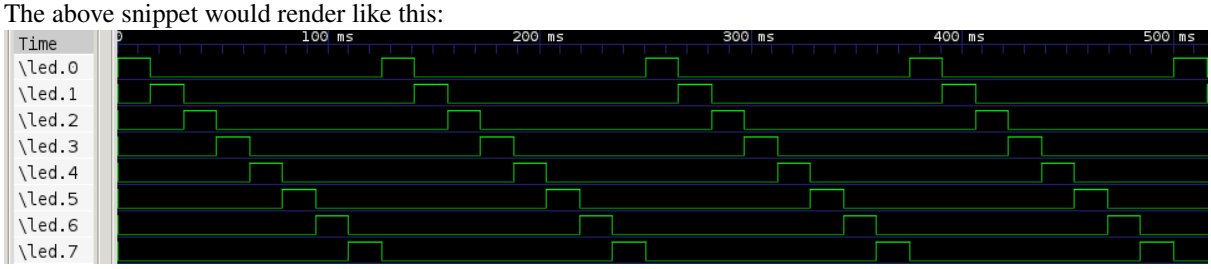


2.5.2 height, width

Example:

```
.. gtkwave:: docs/gtkwave_output.vcd
     :height: 100px
     :width:  100 px
```

The above snippet would render like this:

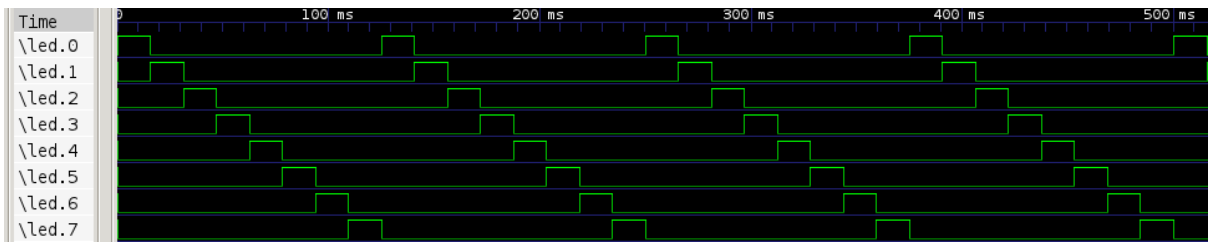


2.5.3 align

Example:

```
.. gtkwave:: docs/gtkwave_output.vcd
     :align: right
```

The above snippet would render like this:



CHAPTER

THREE

DEVELOPMENT

3.1 Tools

1. setuptools
2. Paver
3. nose
4. ghp-import
5. pyflakes
6. pychecker
7. paved fork
8. Sphinx
9. sphinxcontrib-programscreenshot
10. sphinxcontrib-paverutils
11. `autorun` from sphinx-contrib (there is no simple method, you have to download/unpack/setup)

3.2 Install on ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install python-paver
sudo apt-get install python-nose
sudo easy_install ghp-import
sudo apt-get install pyflakes
sudo apt-get install pychecker
sudo easy_install https://github.com/ponty/paved/zipball/master
sudo apt-get install scrot
sudo apt-get install xvfb
sudo apt-get install xserver-xephyr
sudo apt-get install python-imaging
sudo apt-get install python-sphinx
sudo easy_install sphinxcontrib-programscreenshot
sudo easy_install sphinxcontrib-programoutput
sudo easy_install sphinxcontrib-paverutils
```

3.3 Tasks

Paver is used for task management, settings are saved in pavement.py. Sphinx is used to generate documentation.

print paver settings:

```
paver printoptions
```

clean generated files:

```
paver clean
```

generate documentation under *docs/_build/html*:

```
paver cog pdf html
```

upload documentation to github:

```
paver ghpages
```

run unit tests:

```
paver nose
#or
nosetests --verbose
```

check python code:

```
paver pyflakes
paver pychecker
```

generate python distribution:

```
paver sdist
```

upload python distribution to PyPI:

```
paver upload
```