



User's Manual

March 4, 2024

This document contains information that is proprietary to Analog Flavor. The software and documentation are furnished under a license agreement. Use and distribution of the proprietary information is only authorized in accordance with the terms of the license agreement . This document may be copied in whole or in part for internal business purposes only, provided that this entire notice appears in all copies.

This document is for information and instruction purposes. Analog Flavor reserves the right to make changes in specifications and other information contained in this publication without prior notice.

ANALOG FLAVOR AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

ANALOG FLAVOR SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF ANALOG FLAVOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Analog Flavor EURL au capital de 5000 Euro
SIREN 814 075 727

3 bis, rue des Engenières, 38360 Sassenage, France.
Telephone: +33 6 28 34 91 82
support@analogflavor.com
www.analogflavor.com

Table of Contents

Chapter 1	BeSpice Wave	8
Chapter 2	Installation	9
2.1	System Requirements	9
2.2	Installing the Software	9
2.3	Installing the License	9
Chapter 3	Running BeSpice Wave	11
Chapter 4	The Graphical User Interface	12
4.1	The Menu Bar	13
4.2	The Tool Bar	14
4.3	The Plot Window	14
4.3.1	The Individual Plots	14
4.3.2	The Analog Plot Page	14
4.3.3	The Spreadsheet Page	15
4.3.4	The Analog Bus Plot Page	15
4.3.5	The Calculator Page	15
4.4	The Sidebar Window	15
4.4.1	The Browser Window	15
4.4.2	The Measurements Window	16
4.4.3	The Page Parameters	17
4.4.4	The Page Layout	17
Chapter 5	Using BeSpice Wave	18
5.1	Reading Files	18
5.2	Plotting Curves	18
5.3	Analyzing Curves	19
5.3.1	Zoom	19
5.3.2	Scroll	20
5.3.3	Auto-Scroll	20
5.3.4	Pan	20
5.3.5	View ports	21
5.3.6	Maximize / Minimize Plot	21
5.3.7	Plotting Complex Curves	21
5.3.8	Using a Logarithmic x-Axis Scale	21
5.3.9	The Legend	21
5.3.10	Selecting Curves	22
5.3.11	Highlighting Curves	22
5.3.12	Editing Curve Parameters	22
5.3.13	Connecting Plots	22
5.3.14	Moving Plots	22
5.4	Performing Measurements	23
5.4.1	Cursors	23
5.4.2	The Measurement Window	23

5.4.3	Annotations	23
5.5	Adding Annotations	24
5.5.1	Creating an Annotation	24
5.5.2	Editing an Annotation	24
5.5.3	Moving an Annotation	24
5.5.4	Creating an Annotation from the Cursor	24
5.5.5	Creating an Annotation from the Measurement Window	25
5.6	Exporting Plot Pages as Image	25
5.7	Exporting Plots as Data	25
Chapter 6	The Eye Diagram	26
Chapter 7	The Waveform Calculator	27
7.1	Trigonometric Functions	27
7.1.1	sin	27
7.1.2	sinh	28
7.1.3	asinh	28
7.1.4	acosh	28
7.1.5	atanh	28
7.1.6	tan	28
7.1.7	tanh	28
7.1.8	acos	28
7.1.9	asin	29
7.1.10	atan	29
7.1.11	cos	29
7.1.12	cosh	29
7.1.13	sind	29
7.1.14	sinhd	29
7.1.15	asinhd	29
7.1.16	acoshd	30
7.1.17	atanhd	30
7.1.18	tand	30
7.1.19	tanhd	30
7.1.20	acosd	30
7.1.21	asind	30
7.1.22	atand	30
7.1.23	cosd	31
7.1.24	coshd	31
7.2	Other Functions	31
7.2.1	abs	31
7.2.2	ceil	31
7.2.3	db	31
7.2.4	exp	31
7.2.5	floor	31
7.2.6	trunc	32
7.2.7	ln	32

7.2.8	log10	32
7.2.9	round	32
7.2.10	sqrt	32
7.3	Manipulating Complex Numbers	32
7.3.1	real	32
7.3.2	imag	33
7.3.3	angle	33
7.3.4	magnitude	33
7.3.5	conj	33
7.3.6	phase	33
7.3.7	argument	33
7.4	Vector Manipulation	34
7.4.1	vector	34
7.4.2	simplify	34
7.4.3	size	34
7.5	Polynomials and Interpolation	34
7.5.1	poly	34
7.5.2	piecewise_poly	34
7.5.3	interpolate_poly	34
7.5.4	interpolate_linear	35
7.5.5	interpolate_spline	35
7.5.6	extrapolate_poly	35
7.5.7	extrapolate_linear	35
7.5.8	extrapolate_spline	35
7.6	Manipulating Waveforms	36
7.6.1	plot	36
7.6.2	waveform	36
7.6.3	create_waveform	36
7.6.4	reduce_waveform	36
Chapter 8	The Spreadsheet Page	37
8.1	Working with 1-dimensional Data	37
8.2	Working with n-dimensional Data	37
8.3	Exporting Data	38
8.4	Configuring the Spreadsheet Page	38
Chapter 9	Mouse and Keyboard Controls	39
Chapter 10	Configuring BeSpice Wave	40
Chapter 11	Interactive Mode	41
11.1	Commands Controlling BeSpice Wave	41
11.1.1	help	41
11.1.2	generate_documentation_from_help	41
11.1.3	close	41
11.1.4	get_status	41
11.1.5	wait	42
11.1.6	execute_command_file	42

11.1.7	<u>read_configuration_file</u>	42
11.1.8	<u>save_configuration_file</u>	42
11.1.9	<u>get_configuration_file_name</u>	42
11.1.10	<u>set_option</u>	42
11.2	<u>Commands for Setting Options</u>	43
11.2.1	<u>set_option browser_colorize_curves background</u>	43
11.3	<u>Commands for File Operations</u>	43
11.3.1	<u>open_file</u>	43
11.3.2	<u>reload_file</u>	43
11.3.3	<u>close_all</u>	43
11.3.4	<u>close_file</u>	44
11.4	<u>Commands for Retrieving Curve Information</u>	44
11.4.1	<u>get_number_of_sections</u>	44
11.4.2	<u>get_section_name</u>	44
11.4.3	<u>get_section_file_name</u>	44
11.4.4	<u>get_section_number_of_curves</u>	44
11.4.5	<u>get_curve_name</u>	44
11.4.6	<u>get_curve_x_unit</u>	44
11.4.7	<u>get_curve_y_unit</u>	45
11.5	<u>Commands for Handling Plots</u>	45
11.5.1	<u>add_plot</u>	45
11.5.2	<u>make_plot_visible</u>	45
11.5.3	<u>get_number_of_plots</u>	45
11.5.4	<u>get_plot_name</u>	45
11.5.5	<u>get_current_plot_name</u>	45
11.5.6	<u>close_plot</u>	46
11.5.7	<u>clear_plot</u>	46
11.5.8	<u>maximize_plot</u>	46
11.5.9	<u>minimize_plot</u>	46
11.6	<u>Commands for Showing Curves</u>	46
11.6.1	<u>highlight_curve</u>	46
11.6.2	<u>add_curve_to_plot_by_name</u>	46
11.6.3	<u>add_curve_to_plot_by_index</u>	46
11.6.4	<u>colorize_curve_by_index</u>	47
11.6.5	<u>colorize_curve_by_name</u>	47
11.6.6	<u>colorize_curve_by_id_and_index</u>	47
11.6.7	<u>decOLORize_all_curves</u>	47
11.6.8	<u>zoom</u>	47
11.6.9	<u>set_cursor</u>	47
11.6.10	<u>set_curve_style</u>	48
11.7	<u>Commands for Adding Curve Data</u>	48
11.7.1	<u>add_section</u>	48
11.7.2	<u>add_curve</u>	48
11.7.3	<u>add_points</u>	48

<u>11.8</u>	<u>Registering Callback Functions</u>	<u>48</u>
<u>11.8.1</u>	<u>register_callback_function_close_button</u>	<u>49</u>
<u>11.8.2</u>	<u>register_callback_function_highlight_curve</u>	<u>49</u>
<u>11.8.3</u>	<u>register_callback_function_unhighlight_curve</u>	<u>49</u>
<u>11.8.4</u>	<u>register_callback_function_activate_curve</u>	<u>49</u>
<u>11.9</u>	<u>Example 1: Command File</u>	<u>49</u>
<u>11.10</u>	<u>Example 2: Use in a Python Script</u>	<u>49</u>
<u>Chapter 12</u>	<u>Troubleshooting</u>	<u>51</u>
<u>12.1</u>	<u>The Log Window</u>	<u>51</u>
<u>12.2</u>	<u>License Problems</u>	<u>51</u>
<u>Chapter 13</u>	<u>Limitations</u>	<u>52</u>
<u>Chapter 14</u>	<u>Credits</u>	<u>53</u>
<u>Chapter 15</u>	<u>Comma-separated value format</u>	<u>55</u>

CHAPTER 1 BESPICE WAVE

BeSpice Wave is a waveform viewer enhanced and specialized for visualizing and analyzing Spice simulation results. Many different file formats are supported.

BeSpice Wave can be integrated into other applications. This feature makes use of the interactive commands and is handled in detail in a separate user manual.

The underlying waveform parser is also available as a C shared library. This allows you to use it in your own application. We also provide script interfaces.

CHAPTER 2 INSTALLATION

Analog Flavor software is installed in two steps. First the software package itself is installed. In a second step a license file is generated and installed to allow the software to run.

2.1 System Requirements

Currently Analog Flavor software is available for Linux 32 and 64 bits and Windows XP or later. The executables are build for Red Hat Linux 5 and later. They require GTK+ which is installed by default on all supported Linux distributions. Executables for other platforms can be built on demand.

2.2 Installing the Software

After purchasing the software it can be downloaded from our ftp site. Connect to the ftp site and download the compressed Analog Flavor archive. The file name is “analog_flavor.tar.gz” and might have an extension indicating the version or build date. Copy this archive to an appropriate location and decompress it using the shell command “tar xzf analog_flavor.tar.gz”.

This will generate a directory “analog_flavor” with the following directory structure:

- **analog_flavor**, the main directory.
 - **bin**, containing the wrappers for the main executables (Linux) or the binary executables (Windows).
 - **documentation**, containing the user manuals and documentation.
 - **examples**, containing some basic examples for the use of Analog Flavor software.
 - **license**, the location of the license file and executables to start and stop the license server.
 - **platform**, containing the binary executables for different Linux platforms.

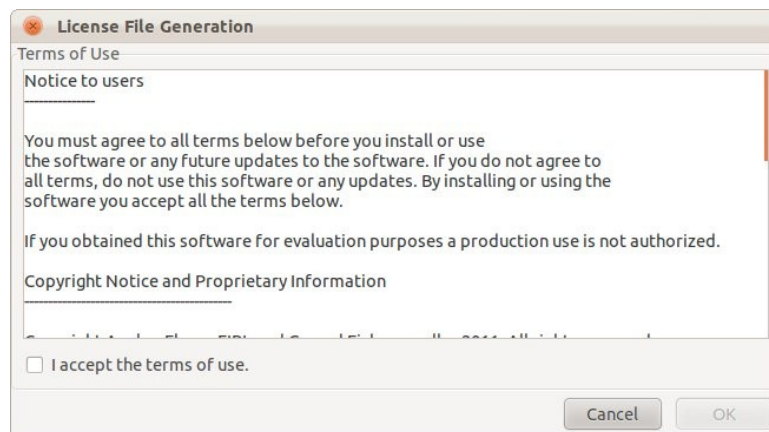
The Analog Flavor executables can be launched directly by calling the wrapper scripts in the “bin” directory:

```
> ./analog_flavor/bin/bspwave
```

To simplify launching Analog Flavor executables you can add the directory “./analog_flavor/bin” to your PATH variable or define aliases in your “.bashrc” file or equivalent.

2.3 Installing the License

A license file might be required for the software. In this case a window will open when you first launch the software:



After having accepted the terms of use, a file required for a license request can be generated and saved for example as “license-request.txt”. Send this file to your Analog Flavor support. If a license server is required for your installation, also send the IP and an available range of ports for the license server. You will receive a valid license file “license.txt” in return. This file must be copied to the directory “analog_flavor/license/” in your install tree. For a local installation the file can also be copied to the configuration directory “~/.AnalogFlavor”.

If a license server is required for your installation, you can start it using by running the command:

```
> ./analog_flavor/license/af_license_server_start.
```

If another license server from a previous Analog Flavor software installation is already running, you must first stop the old server by using the command:

```
> ./analog_flavor/license/af_license_server_stop.
```

The status of the license server can be checked with:

```
> ./analog_flavor/license/af_license_server_status.
```

Now your software installation is complete. Verify it by launching an Analog Flavor executable such as

```
> ./analog_flavor/bin/bspwave.
```

If a license request file has to be re-generated or if the “License File Generation” window does not show up automatically it can be shown manually by selecting the appropriate menu item in all Analog Flavor applications.

CHAPTER 3 RUNNING BESPICE WAVE

BeSpice Wave is launched using the command

```
> ./analog_flavor/bin/bspwave <file_names>.
```

The specified files will be parsed and opened in the viewer if the file extension is recognized. Any number of files can be specified. To simplify launching Analog Flavor executables you can add the directory “./analog_flavor/bin” to your PATH variable or define aliases in your “.bashrc” file or equivalent. Some command line options are available:

- “--help” prints a a help message and the available options and exits.
- “--log <log_file_name>”. All error messages and warnings are printed to the specified file. This might be useful for generating a debug output for your Analog Flavor support. This option is also available from the log window.
- “--socket <connection_ip> <connection_port>” launches BeSpice Wave in interactive mode. The communication is handled over a socket connection which must exist when the tool is launched.
- “--command_file <file_name>” launches BeSpice Wave in interactive mode and reads commands from the specified file. See the corresponding section of this user's guide for details.

Any number of file names can be specified in the command. All files will be opened in BeSpice Wave. The different options for executing interactive commands aren't compatible and can't be used at the same time.

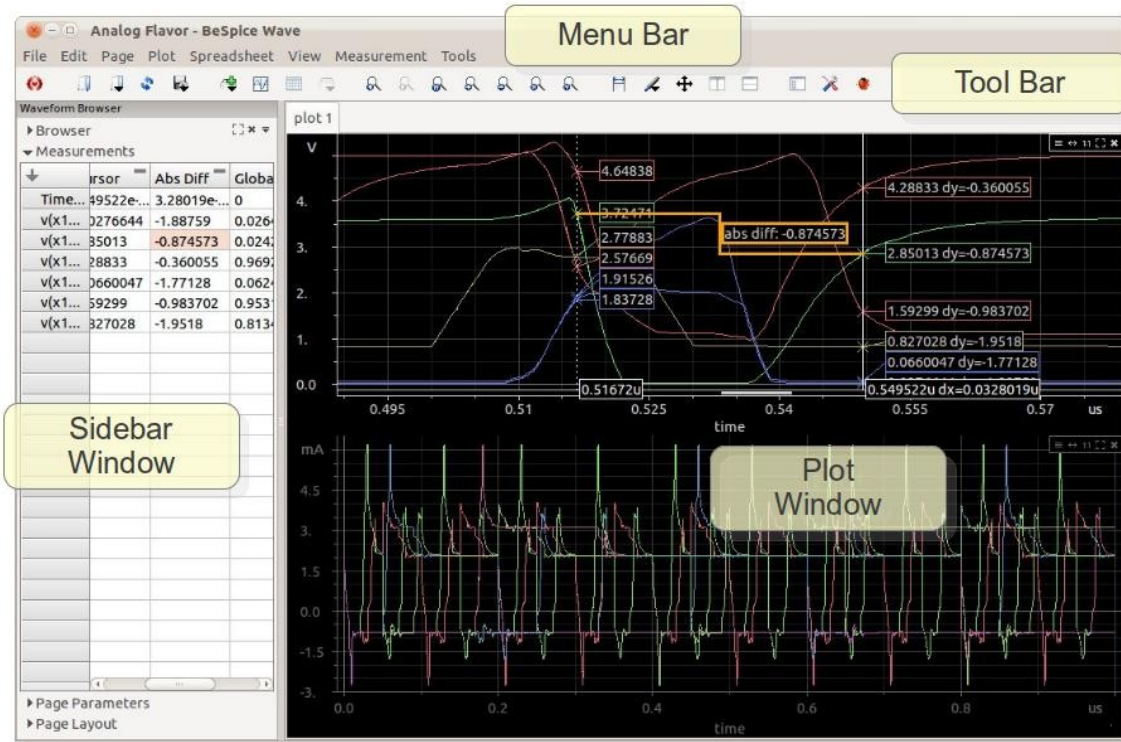
CHAPTER 4 THE GRAPHICAL USER INTERFACE

The BeSpice Wave Window is composed of several sub-windows. Their appearance and number can vary with the version and extensions of your software. The main components are:

- the menu bar: all operations can be performed from a menu in the menu bar.
- the tool bar. It defines shortcuts to the most often used menu items.
- The sidebar window. It contains several sub-windows such as the browser, measurements, page parameters and layout.
- the browser window. All files that were opened in BeSpice Wave are shown here. The read curves can be shown by dragging them to a compatible plot window.
- the plot window. All curves that have been opened are shown in corresponding tabs.
- the measurements window. This tool allows to perform measurements on the curves in the currently active plot window.

The arrangement of the different window can be changed. All windows except the plot window can be closed and re-opened by clicking on the corresponding tool icons in the tool bar or by selecting the appropriate items in the menu bar. The windows' size can be changed by dragging the window borders with the mouse. The sub-windows can be detached from the parent window by dragging the sub-window's title bar.

All components of BeSpice Wave's graphical user interface and their functionality are detailed in the corresponding sections of this user's guide.



4.1 The Menu Bar

The menu bar contains several menus allowing to access BeSPICE Wave's functions. The menu items are grouped by functionality:

- The “File” menu allows to open or close files, export plots as images or as data and to exit the tool.
- The “Edit” menu allows to select, cut, copy, paste and delete curves in the current plot. An undo and redo functionality for these actions is also available. All these actions apply to the currently selected plot.
- The “Page” menu allows to open or close pages in the plot window or to split the page that is currently shown.
- The items in the “Plot” menu allow to switch between linear and logarithmic x-axis scale and to modify the way the grid is shown in the current plot. The way complex curves are plotted can also be adapted.

- The menu items in “Spreadsheet” allow to change aspects of the current page if the currently shown page is a spreadsheet page.
- Using the items in the “View” menu the zoom magnification can be changed and the plot can be scrolled into any direction.
- The “Measurement” menu regroups functions allowing to analyze the displayed curves. The cursor mode can be enabled or disabled. The measurement window can be activated. Annotations in the plot window can be added.
- The “Tools” menu allows to show or hide the browser and the properties window. The log file or the configure dialog can be shown.

4.2 The Tool Bar

The tool bar allows faster access to the most often used items in the menu bar.

4.3 The Plot Window

All curves can be shown in pages in the plot window. Several different page types with specific functionalities are available. Some of them can be split horizontally or vertically to hold several individual plots.

2 types of plot pages are currently available:

- The “Analog Plot Page” can be split horizontally or vertically to hold up to 16 individual plots for. Plots with identical x-axis type might be connected.
- The “Analog Bus Plot Page” can only be split horizontally and hold up to 100 individual plots. All plots must have identical x-axis type. All of them are connected.

Further a spreadsheet page allows to view curve values in text form. Several pages allow to perform post-processing tasks such as fft computations and waveform calculations.

4.3.1 The Individual Plots

Curves are plotted by dragging them from the browser to an appropriate plot. They can also be dragged to another plot or copied and pasted. Only curves with identical x-axis units such as time or frequency can be shown in the same plot. Two different y-axis units such as current and voltage can be displayed. Incompatible curves will not be added to an existing plot.

Each individual plot has a legend listing the names of the plotted curves. Several functions for controlling the way the curves are shown on the screen are available.

4.3.2 The Analog Plot Page

This page can be split horizontally or vertically to hold up to 16 individual plots for. Plots with identical x-axis type might be connected by checking the corresponding menu item in the menu “Page”. When plots are connected zooming the x-axis of one plot automatically zooms all connected plots. In addition BeSpice Wave synchronizes cursors in the connected plots.

4.3.3 The Spreadsheet Page

This page allows to display data in a spreadsheet, organized by rows and columns. The data can be sorted and reorganized and exported to external tools. More details on all available features are given in the corresponding chapter of this documentation.

4.3.4 The Analog Bus Plot Page

This page can only be split horizontally and hold up to 100 individual plots. All plots must have identical x-axis type. All of them are connected. That means that the visible x-axis ranges are identical and all cursors are synchronized.

4.3.5 The Calculator Page

The Waveform calculator allows to create or modify waveforms. It also allows to perform computations on real and complex vectors and scalar values. More details are given in the corresponding chapter of this documentation.

4.4 The Sidebar Window

This page holds several sub-windows that can be shown individually.

4.4.1 The Browser Window

When files are opened in BeSpice Wave a corresponding entry appears in the browser window. It can be expanded to display all defined sub-sections and curves. Clicking on the right mouse button allows to modify the color or line style used for drawing curves or to plot curves or entire sections. Curves or sections can also be displayed by dragging them to an appropriate plot window.

The way the curves and data sections are displayed in the browser can be configured. The configuration dialog box can be reached either by selecting the corresponding menu item from the “Tools” menu in the menu bar or by selecting the appropriate item in the context menu.

The following aspects can be changed:

- When parsing curve data, BeSpice Wave creates a curve hierarchy. That means that the curve V(X1.X1.node1) will be shown in an expandable tree item “X1” which has another expandable tree item “X1”. If “Use full hierarchical curve names” is checked, the full hierarchical curve names such as V(X1.X1.node1) are used instead of the simplified curve names such as V(node1) inside these expandable tree items.
- Several data sections might have been generated for different simulation runs and different sets of parameters such as supply voltages etc. “Parameters for multiple runs” allows to define how these parameters and the run indices appear in the tree hierarchy of the browser.
- If “Highlight selected curves” is checked all curves selected in the browser will be highlighted in every plot or spreadsheet tab they are shown in.
- If “Select curves highlighted in plot” is active, all curves highlighted in a plot or spreadsheet tab are selected in the browser window. This can be limited to visible curves.

In addition to the browser configuration some additional setup options are available when selecting the “Show Browser Setup” icon in the browser window.

- In “Item order” the order in which the tree is constructed can be changed. In general the “File name” will appear at the highest tree level. However if several files with identical curve names are opened it can be advantageous to rise “Curve name” above “File name”. Therefore select an item and move it with the shown arrow icons.
- “Run parameter order” allows to change the order for the parameters used for different simulations in multi-run files.
- The curves shown by BeSpice Wave might be N-dimensional. That means that they have been generated with respect to several sweep variables and form the entries in a N-dimensional matrix. When plotted the curves are always plotted versus the fastest varying sweep variable. The other sweep variables are used as if they were parameters in a multi-run simulation. This tab allows to change the sweep variables order. That means that other variables are used as parameters and the curves can be plotted with respect to another sweep variable.

Selecting “Show Browser Filters” allows to define browser entries with specified values that will be filtered out. The filters apply to run parameters values or the values of sweep variables for n-dimensional data. When large sets of different parameters or seep variables are used, the filter tab allows to hide a subset of all shown parameters and sweep variables.

4.4.2 The Measurements Window

The measurement window is a table that shows values computed for the current plot window. All plotted curves are processed depending on the items selected in the “Configure” dialog and the “Measurement Preferences”. Globally defined values refer to the entire curve, locally defined values refer to the interval between the 2 cursors.

In some cases clicking on table entries generates annotations for values, intervals or peak-to-peak computations in the corresponding plot window.

The following values can be computed:

- “Reference Cursor Value”: the value under the reference cursor is displayed.
- “Cursor Value”: the value under the main cursor is shown.
- “Absolute Difference”: the absolute difference between cursor values of the same curve is computed.
- “Relative Difference”: the relative difference between cursor values of the same curve is computed if the value under the reference cursor is different from 0.
- “Global Minimum”: the overall minimum of the curve is shown.
- “Global Maximum”: the overall maximum of the curve is shown.
- “Global Peak to Peak”: the overall peak-to-peak value of the curve is computed.
- ”Local Minimum”: the minimum of all values between the two cursors is shown.

- "Local Maximum": the maximum of all values between the two cursors is shown.
- "Local Peak to Peak": the peak-to-peak value of the curve is computed between the two cursors.

Clicking on values in the table generates annotations in the corresponding plot window. These annotations might be useful when exporting the plots for further use.

4.4.3 The Page Parameters

The plot currently active in the plot window can be parameterized. This window shows the current window's parameters and allows to modify them.

4.4.4 The Page Layout

This window allows to split the current plot window into several plots. The resulting individual plots can be re-arranged, maximized or closed.

CHAPTER 5 USING BESPICE WAVE

The first step when using BeSpice Wave is to open the files containing the waveform data to be visualized. Depending on the version of your software several proprietary and non-proprietary formats can be read by BeSpice Wave.

Now the read curve data can be plotted in an appropriate plot. Each plot can hold many curves of identical x-axis type and of one of two available y-axis types. Plots can be copied or moved to other plots if the current plot page is split into several individual plots.

The shown curves can be analyzed graphically by using the various possibilities to zoom and scroll a plot. The way a curve is shown can be adjusted and it can be highlighted.

The cursors allow to perform measurements on the plotted curves. The “Measurement Window” allows to supervise basic curve properties.

Adding and editing annotations also allows to perform basic measurements on the curves. Further it allows to add text to the curves and plots. This is useful if the plot will be used in a report.

To use a plot in a report it can be exported as image. Several formats such as *.jpg and *.png are available.

5.1 Reading Files

Select “Open File” from the “File” menu to read a file containing waveform data. Alternatively the corresponding keyboard shortcut or tool bar button can be used. A dialog box will allow you to choose the file to open. Click “OK” to parse the waveform file. If it has been read successfully it will be listed in the browser window. Otherwise a message indicating a parser error will be shown. The error log might guide you to find out what went wrong.

Some file types can be reloaded if they change on disk. This can be done using the from the browser by clicking on the right mouse button over the file name and selecting the corresponding menu item. It can also be done automatically at regular time intervals by defining an “Auto-reload time interval” in the configuration dialog.

Files can be removed from the browser and all plots by closing them. This can be done by clicking on the right mouse button over the file name in the browser and selecting the corresponding menu item. All files can be closed at once by selecting “Close all Files” from the “Files” menu.

5.2 Plotting Curves

One or several curves are plotted by selecting them in the browser window and dragging to the appropriate location. They will only be plotted if their x- and y-axis types are compatible with the concerned plot. Otherwise an warning message will be printed to the error log. If no plot page is shown a new plot page will be generated automatically.

Curves or entire sections can also be plotted from the browser's context menu or by activating (double-clicking) a browser entry. In this BeSpice Wave will plot all curves in the current page and create new plots if necessary.

Curves can be moved among the individual plots of the same plot page by dragging them to the the

new plot. When dragging the curves from the legend all selected curves are moved. When dragging starts directly from the plot all highlighted curves will be moved to the new location. If curves should be copied rather than moved the “CTRL” key should be held down while starting to drag the curves.

Instead of moving curves they can also be copied to a new plot by using the functionality from the “Edit” menu or the corresponding keyboard shortcuts. Curves are only cut or copied from the currently active plot. Curves from several plots can't be cut or copied at once. Curves can't be cut or copied from the browser.

5.3 Analyzing Curves

Many user interface functions such as scrolling and zooming allow to analyze the plotted curves graphically.

Important: When used through the menu bar or the tool bar the functions apply to the active plot. If several plots are shown in the current plot page the axes of all non-active plot are drawn in gray. As the active window changes when the mouse cursor moves over another plot it might be preferable to use keyboard shortcuts instead of menu bar or tool bar buttons for all functionality concerning on an individual plot.

Important: The menus “Edit”, “Plot”, “View” and “Measurement” are in BeSpice Wave's menu bar but some of their entries have been duplicated to the temporary context menu that is shown when actioning the right mouse button in an individual plot. Using these context menus also allows to avoid ambiguities if several plots are shown on the current page.

5.3.1 Zoom

For viewing a curve in details or as a whole it might be important to zoom to a specific region of a curve or to un-zoom such that the whole curve is shown. Several mechanisms can be zooming a plot:

- Mouse wheel zoom: actioning the mouse wheel while over a plot changes the zoom perspective while the position of the mouse cursor in the plot is fixed. If the cursor is over an axis only the corresponding axis is scrolled. This allows to scroll each y-axis independently.
- Rectangular zoom: When clicking on an empty space in a plot and holding the mouse button down a zoom box can be drawn. When the button is released only the region of the plot that is inside the zoom box is shown. When clicking close to a curve BeSpice Wave will attempt to drag this curve instead. To draw a zoom box in this case the “shift” key on the keyboard must be pressed in addition.

Depending on how the mouse is moved after having clicked on the plot the zoom box might extend to the entire plot in x- or y-direction.

By pressing the “space” key of the keyboard once or several times BeSpice Wave rotates through different available types of zoom boxes. This includes un-zoom boxes that are drawn using dashed lines.

The rectangular zoom functionality can also be triggered by clicking on the right mouse button on an empty space of the plot and by selecting the appropriate function in the “Edit” menu.

When starting to draw a zoom box over the axes, the zoom boxes will extend over the entire plot in the other direction.

The rectangular zoom mode can be exited pressing the “Escape” key.

- Menu “View”: the menu “View” defines functions allowing to zoom in or out or to zoom into x- or y-direction. Alternatively the indicated keyboard shortcuts can be used. If the mouse cursor is over the active plot, the position of the cursor is chosen as center of the zoom operation.
- The un-zoom icon: each plot has some icons in it's upper right corner. One of these icons allows to un-zoom the concerned plot such that the curves exactly fit into the plot.

The precision for the axis tick labels might not be sufficient when the plot is zoomed to a very small part of the curve. In this case only one axis tick label will show the truncated accurate value. All other labels are blue. They display the distance to the accurate tick label.

5.3.2 Scroll

If a plot is zoomed only a part of the curves might be shown. Scrolling the curve allows to shift the plot such that a neighboring region of the plot is shown at the same zoom scale. Several functions allow to scroll a plot:

- Scroll bars: if scroll bars are shown for a plot they can be moved by actioning the left mouse button and moving the mouse while holding the mouse button down.
- Mouse wheel scroll: if scroll bars are shown for a plot and the mouse wheel is actioned while the mouse cursor is positioned on them, the plot is scrolled into the corresponding direction. This allows to scroll one of the two y-axes independently from the other.
- Menu “View” or tool bar: by selecting the corresponding functions in the menu “View” or actioning the corresponding tool bar buttons the active plot can be scrolled into any direction. Alternatively the keyboard arrows can be used. If the mouse cursor is over one of two y-axes only the corresponding axis is scrolled up or down.

The scroll direction can be inversed by configuring BeSpice Wave accordingly.

5.3.3 Auto-Scroll

If the auto-scroll function is activated, an additional icon is drawn on the active plot. Depending on the mouse cursor position relatively to this icon the active plot is scrolled more or less fast into the predominant direction. The auto-scroll function can be activated in several ways.

- Clicking on the middle mouse button.
- Selecting the corresponding item in the menu “View”.
- Using a keyboard shortcut that might have to be defined previously in the “Configure” dialog.

The auto-scroll mode can be exited pressing the “Escape” key.

5.3.4 Pan

The pan function can be used by clicking the right mouse button while holding the “Ctrl” key down.

Now the concerned plot can be moved by moving the mouse.

5.3.5 View ports

When zooming or scrolling a plot, the currently shown region of a plot is stored as a view port. Several buttons and menus allow access to this functionality:

- Menu “View”: the items “Previous View Port” and “Next View Port” allows to navigate through the view ports. Keyboard shortcuts can be defined in BeSpice Wave's configure dialog.
- Tool bar buttons: two tool bar buttons allow access to the view port navigation.

5.3.6 Maximize / Minimize Plot

When viewing several curves it can be useful to split a page to show them into distinct plots. Each individual plot can be maximized. This will show only the concerned plot until it is minimized again. This functionality is available over several menus and buttons:

- Menu “Plot”: select “Maximize Current Plot” to maximize the active plot or to minimize it if it has already been maximized.
- The maximize / minimize icon: each plot has some icons in it's upper right corner. One of these icons allows to minimize / maximize the concerned plot.

5.3.7 Plotting Complex Curves

Complex curves can be shown in several different ways (real and imaginary part, magnitude and phase, etc.). The way complex curves are shown for the active plot can be changed in “Complex Curve Representation” from “Plot” menu in the menu bar or from the individual plots context menu.

The default settings can be changed in BeSpice Wave's configuration.

5.3.8 Using a Logarithmic x-Axis Scale

In some cases a logarithmic x-axis scale might be more appropriate than a linear axis. BeSpice Wave will automatically use a logarithmic scale if the x-axis data corresponds. However users can choose between a logarithmic or a linear scale by actioning the menu item “Logarithmic X Axis” in the menu “Plot”.

5.3.9 The Legend

The legend allows to have an overview over the plotted curves. It also allows to select some of them to prepare a copy and paste operation. Besides the curve parameters can be changed here.

The legend can be shown inside the plot window, on it's left or on it's right. It can also be configured to be hidden automatically when it is no longer used. If shown inside the plot window the legend can be shown or hidden in several ways:

- Menu “Plot”: select “Show Legend” to show the legend when it is hidden or hide it when it is shown.
- The legend icon: each plot has some icons in it's upper right corner. One of these icons allows to

show the legend of the concerned plot.

- The close button on the legend: the legend has a close button in its upper right corner that can be used to close it.

A legend shown inside a plot button that is not hidden automatically can be moved. Therefore click on its header and drag it to its new position by holding the mouse button pressed.

5.3.10 Selecting Curves

The Legend can be used to select curves for example for a copy and paste operation. The names of the selected curves are highlighted. The menu item “Select All Curves” in the “Edit” menu allows to select all curves displayed in the active plot at once.

5.3.11 Highlighting Curves

Clicking onto a curve or close to a curve highlights the concerned curve and removes highlighting for all curves that might have been highlighted before. What exactly highlighting means can be defined in BeSpice Wave's configure dialog. It is also possible to highlight curves from the legend. All highlighted curves are also selected.

To highlight a curve plotted as histogram a point close to the top of the histogram must be clicked.

5.3.12 Editing Curve Parameters

The parameters used for plotting an individual curve can be changed. The parameters contain the line style, the used symbol, the line width and the line color. They can be changed in the browser window or for each plot. The changes made in the browser window will apply when the curve is dragged to a new plot. The changes made for an individual plot will only apply to the currently shown curve.

In the browser the curve parameters from the context menu shown when actioning the right mouse button over a curve name. For each plot they can be changed when showing the context menu over a legend entry or close to the concerned curve.

5.3.13 Connecting Plots

Connecting plots might be very useful when analyzing waveform data. The visible x-axis ranges of connected plots are identical and all cursors are synchronized. All plots of analog bus plot pages are automatically connected. Matching plots of analog plot pages can be connected and disconnected by selecting the menu item “Connect Plots” from the menu “Page”.

5.3.14 Moving Plots

The position of several individual plots in a single plot page can be changed. Therefore select the menu item “Move Plots” from the “Page” menu or use the corresponding keyboard shortcut. A dialog box will allow to change the position of the shown plots. When the dialog is closed the new positions are transferred to the individual plots.

5.4 Performing Measurements

BeSpice Wave integrates several possibilities to perform measurements on the shown curves. These possibilities include annotations and cursors especially when the measurement window is shown.

5.4.1 Cursors

Up to two cursors can be defined for each individual plot. If more cursors are required, equivalent annotations can be used to complete cursors. The cursor mode can be activated in several ways:

- Menu “Measurement”: select the item “Activate Cursor Mode” from the menu “Measurement”.
- The cursor icon: each plot has some icons in it's upper right corner. One of these icons allows to activate or deactivate the cursor mode for the concerned plot.

After the cursor mode has been activated, moving the mouse cursor over the active plot will move the first cursor. It can be positioned by clicking the left mouse button. Now the second cursor can be positioned like the first one.

Both cursors can be re-positioned by dragging them to a new location.

Some aspects of cursor appearance and behavior can be configured in BeSpice Wave's configure dialog:

- Snap to existing x-values: if enabled, the allowed x-axis position of the cursors are limited to the values for which a point has been defined in the underlying data. Otherwise any x-axis position is allowed. Missing y-axis values are interpolated.
- Measure differences: if enabled, the differences in x- and y-direction are computed and shown on the cursor.
- Cursor style: this affects how the values on the cursor are shown. In order to obtain a more readable plot showing values on the cursors can also be deactivated. The cursor values might just as well be read in the measurement window.
- Normalize values: if activated the value $1.0\text{e-}9$ will be shown as 1.0n.

Cursors can be removed from the active plot by selecting the item “Remove Cursors” from the “Measurements” menu.

5.4.2 The Measurement Window

The measurement window can be activated from the “Measurement” menu. It automatically shows the cursor values and computes a number of values for the curves of the active plot. See the corresponding chapter of this documentation for more detailed information.

5.4.3 Annotations

Annotations can also be used for performing simple measurements. They are handled in detail in a dedicated section of this user guide.

5.5 Adding Annotations

Annotations can be used to add complementary information to a plot. They can be generated automatically and hold curve names, curve values or differences between values. They can also be generated using the cursors or from the measurements.

5.5.1 Creating an Annotation

Annotations can be created from the menu item “Add Annotation” in the “Measurement” menu. The following types of annotations can be created:

- Text annotations: they are associated to the plot and not to a particular curve.
- Curve value annotations: BeSpice Wave will find the curve closest to the mouse cursor and the closest point on this curve. An annotation containing the y-value will be added at this location. Clicking the left mouse button will fix this annotation. If the mouse cursor is too far from the closest curve the annotation can't be set.
- Curve name annotation: the mechanism is identical to the curve value annotation except that the curve's name is printed instead of the curve's value.
- Compare values on identical curve annotations: This annotation will contain the difference between two values of the identical curve. The first point is defined like for curve value annotations. Now a second point needs to be defined. BeSpice Wave only accepts points on the same curve.

To create a several curve name or value annotations without having to creating each of them individually the menu items “Several Curve Values” and “Several Curve Names” can be selected.

Creating annotations can be interrupted pressing the “Escape” key.

5.5.2 Editing an Annotation

Annotations can be edited after having created them. Therefore move the mouse cursor over the annotation and press the right mouse button. The shown context menu contains the item “Edit Annotation”.

The shown dialog allows to edit the annotations text. A certain number or predefined curve properties can be inserted through the “Insert Text” drop down menu.

A temporary annotation can be made permanent by checking “Make Permanent” in the dialog.

5.5.3 Moving an Annotation

Annotations can moved from their initial position for example if they hide another annotation. Therefore the annotation simply has to be dragged to it's new location.

5.5.4 Creating an Annotation from the Cursor

If configured accordingly the cursor values are shown in annotations. These annotations can be made permanent. That means that they will remain even if the cursor is moved or deleted. This can be done through the annotations context menu or by selecting “Make Annotations Permanent” in the

“Measurement” menu.

5.5.5 Creating an Annotation from the Measurement Window

Annotations can also be generated from the measurement window. When clicking on a value computed for example for “Global Maximum”, a temporary annotation highlighting this value is generated in the active plot at the corresponding location.

This annotation can be made permanent just like a cursor annotation.

5.6 Exporting Plot Pages as Image

Plot Pages can be exported as images for further use in reports or by other tools. This can be achieved by selecting the corresponding item in the “File→Export” menu.

5.7 Exporting Plots as Data

The data in the current plot can be exported to a data file. Currently comma-separated values or as html files can be generated. These files can then be read by other tools such as text editors or table calculation software.

To export a spreadsheet select the item “Export As Data File” from the “File” menu. A dialog box for selecting the file name, the file type and some export options will open.

This procedure can be simplified by selecting the curves to be exported and by dragging them to the external tool. The “copy and paste” mechanism can also be used. In this case the options defined for the last successful “Export As Data File” action are re-used. This concerns for example the used separator for comma-separated values.

CHAPTER 6 THE EYE DIAGRAM

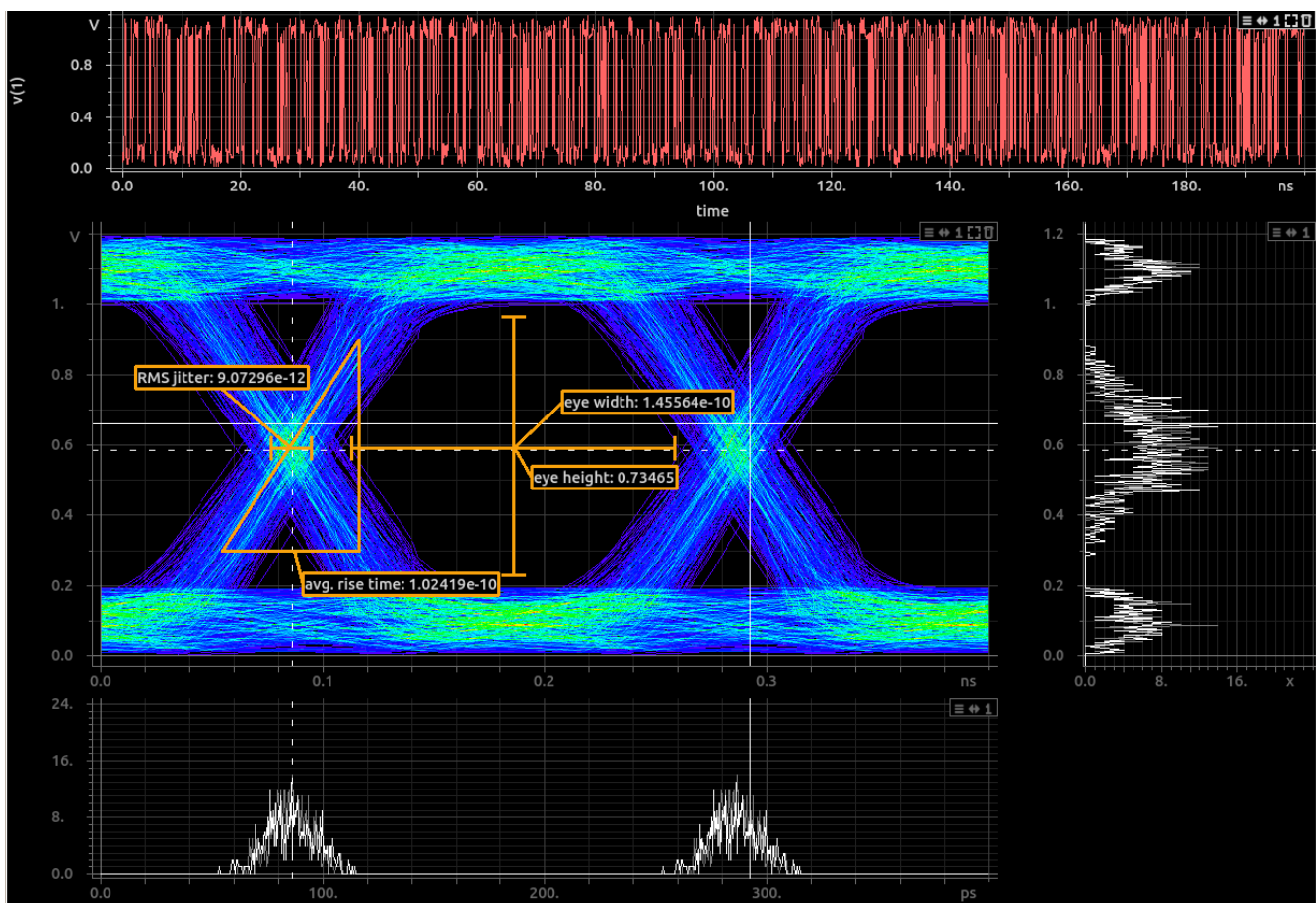
Eye diagrams are used to quickly and accurately measure the quality of a digital signal. They are built by folding a waveform into a single graph. The resulting graph will resemble to an eye.

Measuring the eye diagram's characteristics allows to determine the electrical quality of the signal.

To use BeSpice Wave's eye diagram, create a new eye diagram page by choosing “Page → New Post Processing Page → Eye Diagram” from the menu. Drag and drop one or several waveforms from the browser to the new page. The eye diagram is computed immediately.

Depending on how it has been configured the page shows the folded and unfolded curves. The page can also show vertical and horizontal histograms at the current cursor positions.

The parameters used to compute and display the eye diagram can be adjusted in the page parameters window. The measurement window shows the characteristics extracted from the diagram. Clicking on the measurements creates annotations in the diagram.



CHAPTER 7 THE WAVEFORM CALCULATOR

The waveform calculator allows to perform calculations on waveforms, vectors and scalar values. Complex values can be handled as well. They must be written as $a+j*b$. All computations are performed over the terminal-like interface but a calculator keyboard can be shown from the waveform calculators context menu.

Typing “help” or “?” shows all topics for which a help information is available. Typing “help <topic>” yields an overview over the concerned function and some hints on it's usage.

Waveforms defined in the browser can simply be dragged to the waveform calculator. This will yield a “waveform(<section_name>, <curve_index|curve_name>)” statement. This statement can be used for computations such as:

```
> curve_1 = 2 * waveform(<section_name>, <curve_name>)
```

This will yield a new waveform where all values have been multiplied by 2. The resulting curve is available in the “Post Processing” section of the browser and can be plotted in any other plot window. The index of the curve in the section can be used if the name is not unique. The first curve has the index 1.

It is also possible to create waveforms from vectors or algebraic expressions. The command

```
> curve_2 = create_waveform(“s”, “v”, sin(s))
```

will create a curve “sin(s)”. “s” which is equivalent for seconds is the x-axis unit. “v” which is an equivalent for voltage is the y-axis unit. The commands

```
> x_vec = [0.0:0.5:7.0]
```

```
> curve_3 = create_waveform(“s”, “v”, x_vec, sin(x_vec))
```

will also generate a sine curve, but here it's values are given explicitly by the vectors “x_vec” and the vector returned by “sin(x_vec)”. The values of curve_2 are not given explicitly. They will be computed when the curve is plotted.

The waveform calculator can allows to use the operators +, -, *, /, ^ for computations with all available data types. The operator = defines an assignment. The operators == and != can be used for comparisons. The operators >, <, >=, <= are not defined for complex arguments. In addition many functions are available for manipulating curves, vectors and scalars.

7.1 Trigonometric Functions

7.1.1 sin

Usage: sin(x), x <scalar|vector|waveform>

Returns the sine of the function argument which is given in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.2 *sinh*

Usage: $\sinh(x)$, x <scalar|vector|waveform>

Returns the hyperbolic sine of the function argument which is given in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.3 *asinh*

Usage: $\operatorname{asinh}(x)$, x <scalar|vector|waveform>

Returns the inverse hyperbolic sine of the function argument in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.4 *acosh*

Usage: $\operatorname{acosh}(x)$, x <scalar|vector|waveform>

Returns the inverse hyperbolic cosine of the function argument in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.5 *atanh*

Usage: $\operatorname{atanh}(x)$, x <scalar|vector|waveform>

Returns the inverse hyperbolic tangent of the function argument in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.6 *tan*

Usage: $\tan(x)$, x <scalar|vector|waveform>

Returns the tangent of the function argument which is given in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.7 *tanh*

Usage: $\tanh(x)$, x <scalar|vector|waveform>

Returns the hyperbolic tangent of the function argument which is given in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.8 *acos*

Usage: $\operatorname{acos}(x)$, x <scalar|vector|waveform>

Returns the arc cosine of the function argument in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.9 asin

Usage: `asin(x)`, x <scalar|vector|waveform>

Returns the arc sine of the function argument in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.10 atan

Usage: `atan(x)`, x <scalar|vector|waveform>

Returns the arc tangent of the function argument in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.11 cos

Usage: `cos(x)`, x <scalar|vector|waveform>

Returns the cosine of the function argument which is given in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.12 cosh

Usage: `cosh(x)`, x <scalar|vector|waveform>

Returns the hyperbolic cosine of the function argument which is given in radians.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.13 sind

Usage: `sind(x)`, x <scalar|vector|waveform>

Returns the sine of the function argument which is given in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.14 sinh

Usage: `sinh(x)`, x <scalar|vector|waveform>

Returns the hyperbolic sine of the function argument which is given in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.15 asinh

Usage: `asinh(x)`, x <scalar|vector|waveform>

Returns the inverse hyperbolic sine of the function argument in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.16 acoshd

Usage: `acoshd(x)`, `x <scalar|vector|waveform>`

Returns the inverse hyperbolic cosine of the function argument in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.17 atanhd

Usage: `atanhd(x)`, `x <scalar|vector|waveform>`

Returns the inverse hyperbolic tangent of the function argument in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.18 tand

Usage: `tand(x)`, `x <scalar|vector|waveform>`

Returns the tangent of the function argument which is given in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.19 tanhd

Usage: `tanhd(x)`, `x <scalar|vector|waveform>`

Returns the hyperbolic tangent of the function argument which is given in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.20 acosd

Usage: `acosd(x)`, `x <scalar|vector|waveform>`

Returns the arc cosine of the function argument in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.21 asind

Usage: `asind(x)`, `x <scalar|vector|waveform>`

Returns the arc sine of the function argument in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.22 atand

Usage: `atand(x)`, `x <scalar|vector|waveform>`

Returns the arc tangent of the function argument in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.23 cosd

Usage: `cosd(x)`, x <scalar|vector|waveform>

Returns the cosine of the function argument which is given in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.1.24 coshd

Usage: `coshd(x)`, x <scalar|vector|waveform>

Returns the hyperbolic cosine of the function argument which is given in degrees.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2 Other Functions

7.2.1 abs

Usage: `abs(x)`, x <scalar|vector|waveform>

Returns the absolute value or magnitude of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.2 ceil

Usage: `ceil(x)`, x <scalar|vector|waveform>

Returns the smallest integral value not less than the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.3 db

Usage: `db(x)`, x <scalar|vector|waveform>

Converts the absolute value of function argument to db.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.4 exp

Usage: `exp(x)`, x <scalar|vector|waveform>

Raises e to the power of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.5 floor

Usage: `floor(x)`, x <scalar|vector|waveform>

Returns the largest integral value not greater than the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.6 trunc

Usage: `trunc(x)`, x <scalar|vector|waveform>

Rounds the function argument to an integer value toward zero.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.7 ln

Usage: `ln(x)`, x <scalar|vector|waveform>

Returns the natural logarithm of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.8 log10

Usage: `log10(x)`, x <scalar|vector|waveform>

Returns the base 10 logarithm of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.9 round

Usage: `round(x)`, x <scalar|vector|waveform>

Rounds the function argument to an integer value away from zero.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.2.10 sqrt

Usage: `sqrt(x)`, x <scalar|vector|waveform>

Returns the square root of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.3 Manipulating Complex Numbers

The functions manipulating complex numbers, curves and vectors also return complex numbers. That means that “`real(2+3*j)`” returns “`2+0*j`”. The function “`simplify`” converts complex arguments to real ones if the imaginary part is 0. That means “`simplify(real(2+3*j))`” yields “`2`”.

7.3.1 real

Usage: `real(x)`, x <scalar|vector|waveform>

Returns the real part of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.3.2 *imag*

Usage: `imag(x)`, `x <scalar|vector|waveform>`

Returns the imaginary part of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.3.3 *angle*

Usage: `angle(x)`, `x <scalar|vector|waveform>`

Returns the complex angle of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.3.4 *magnitude*

Usage: `magnitude(x)`, `x <scalar|vector|waveform>`

Returns the absolute value or magnitude of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.3.5 *conj*

Usage: `conj(x)`, `x <scalar|vector|waveform>`

Returns the complex conjugate of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.3.6 *phase*

Usage: `phase(x)`, `x <scalar|vector|waveform>`

Returns the complex argument or phase of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.3.7 *argument*

Usage: `argument(x)`, `x <scalar|vector|waveform>`

Returns the complex argument or phase of the function argument.

The returned type can be a real or complex. It can be a scalar, a vector or a waveform and is identical to the argument type. For vectors and waveforms the function is applied to each value.

7.4 Vector Manipulation

7.4.1 vector

Usage: `vector(x, ..)`, `x <scalar|expression>`

or `vector(start:step:stop)`, `start`, `step`, `stop <scalar>`

This function defines vector. The vector can be defined from discrete values or as a range of real or complex values.

The operator `[]` can be used instead of the function `vector()`.

7.4.2 simplify

Usage: `simplify(x)`, `x <expression|scalar|vector|waveform>`

Simplifies the function argument.

A complex function argument is converted to a real one if possible.

7.4.3 size

Usage: `size(x)`, `x <vector|waveform>`

Returns the size of a vector and the number of points in a waveform.

7.5 Polynomials and Interpolation

7.5.1 poly

Usage: `poly(x, [coeff])`, `x <scalar|vector>`, `[coeff] <vector>`

This function defines a polynomial by it's coefficients `[coeff]`.

The coefficients are given in decreasing order (`a_n, ... , a_1, a_0`).

If a value is given for `x`, the polynomial will be evaluated in `x`.

7.5.2 piecewise_poly

Usage: `piecewise_poly(x, [x], [[coeff_1], [coeff_2], ..])`, `x <scalar|vector>`, `[x] <vector>`, `[[coeff_1], [coeff_2], ..] <vector of vectors>`

This function defines a piecewise polynomial on the intervals defined in the vector `[x]`.

For each interval the coefficients of the polynomials are defined in decreasing order in the coefficient matrix `[[coeff_1], [coeff_2], ..]`.

If a value is given for `x`, the piecewise polynomial will be evaluated in `x`.

7.5.3 interpolate_poly

Usage: `interpolate_poly(x, [x], [y])`, `x <scalar|vector>`, `[x] <vector>`, `[y] <vector>`

This function computes a polynomial interpolating the points defined by the vector `[y]` at the points defined by the vector `[x]`.

If the value for `x` is a variable, the function returns a piecewise polynomial.

If a scalar or vector value is given for `x`, the resulting polynomial will be evaluated at `x`.

The value for `x` must lie within the points given in `[x]`.

7.5.4 interpolate_linear

Usage: `interpolate_linear(x, [x], [y])`, x <scalar|vector>, $[x]$ <vector>, $[y]$ <vector>

This function computes a piecewise linear function interpolating the points defined by the vector $[y]$ at the points defined by the vector $[x]$.

If the value for x is a variable, the function returns a piecewise polynomial.

If a scalar or vector value is given for x , the resulting polynomial will be evaluated at x .

The value for x must lie within the points given in $[x]$.

7.5.5 interpolate_spline

Usage: `interpolate_spline(x, [x], [y])`, x <scalar|vector>, $[x]$ <vector>, $[y]$ <vector>

This function computes a natural cubic spline interpolating the points defined by the vector $[y]$ at the points defined by the vector $[x]$.

If the value for x is a variable, the function returns a piecewise polynomial.

If a scalar or vector value is given for x , the resulting polynomial will be evaluated at x .

The value for x must lie within the points given in $[x]$.

7.5.6 extrapolate_poly

Usage: `extrapolate_poly(x, [x], [y])`, x <scalar|vector>, $[x]$ <vector>, $[y]$ <vector>

This function computes a polynomial interpolating the points defined by the vector $[y]$ at the points defined by the vector $[x]$.

If the value for x is a variable, the function returns a piecewise polynomial.

If a scalar or vector value is given for x , the resulting polynomial will be evaluated at x .

The value for x must lie outside of the interval defined by the points in $[x]$.

7.5.7 extrapolate_linear

Usage: `extrapolate_linear(x, [x], [y])`, x <scalar|vector>, $[x]$ <vector>, $[y]$ <vector>

This function computes a piecewise linear function interpolating the points defined by the vector $[y]$ at the points defined by the vector $[x]$.

If the value for x is a variable, the function returns a piecewise polynomial.

If a scalar or vector value is given for x , the resulting polynomial will be evaluated at x .

The value for x must lie outside of the interval defined by the points in $[x]$.

7.5.8 extrapolate_spline

Usage: `extrapolate_spline(x, [x], [y])`, x <scalar|vector>, $[x]$ <vector>, $[y]$ <vector>

This function computes a natural cubic spline interpolating the points defined by the vector $[y]$ at the points defined by the vector $[x]$.

If the value for x is a variable, the function returns a piecewise polynomial.

If a scalar or vector value is given for x , the resulting polynomial will be evaluated at x .

The value for x must lie outside of the interval defined by the points in $[x]$.

7.6 Manipulating Wavefrms

7.6.1 plot

Usage: `plot<expression>`

This function allows to plot an algebraic expression without having to create a waveform. However the x- and y-axis units of the resulting waveform are undefined.

Example: `plot new_curve = x^2`

7.6.2 waveform

Usage: `waveform(<section name>, <curve name>)`

This function allows to reuse any loaded waveform in the calculator.

When dragging a curve from the browser, a `waveform()` expression is created automatically.

Example: `new_curve = waveform("fourbitadder.spi3", "i(vin1b)")`

7.6.3 create_waveform

Usage: `create_waveform(<x axis type>, <y axis type>, <x vector>, <y vector>)`
or `create_waveform(<x axis type>, <y axis type>, <expression>)`

This function creates a waveform that can be plotted. It accepts either two vectors or an algebraic expression as arguments.

Example: `new_curve = create_waveform("x", "y", x^2)`

7.6.4 reduce_waveform

Usage: `reduce_waveform(<waveform>, <eps>)`

This function reduces the number of points in a waveform. The parameter `eps` controls the precision.

`Eps` defines the acceptable absolute error and depends on the waveform's values.

A value of $0.01 * (\text{curve_maximum} - \text{curve_minimum})$ should yield reasonable precision.

CHAPTER 8 THE SPREADSHEET PAGE

A new spreadsheet page can be added to the plot window by activating the corresponding tool bar button or by selecting the appropriate menu from the menu bar. It allows to display data in a spreadsheet, organized by rows and columns.

The data is shown in a table. It can be sorted and reorganized. The visible data can then be sent to external tools such as text editors or table calculation software. The spreadsheet page handles 1-dimensional data such as ordinary waveforms but it can also handle multi-dimensional data with more than one swept variables. The spreadsheet page can also display non-numerical data such as text strings.

8.1 Working with 1-dimensional Data

Output files generated by simulators mostly contain curves represented by 2 vectors. The first vector contains the values for the swept x-variable. The second vector represents the curve values plotted on the y-axis. This data is considered as 1-dimensional data. When a 1-dimensional curve is shown in a spreadsheet page, the x-axis data is shown in the first column and the y-axis data is shown in the second.

When another curve is added to the spreadsheet, the x-axis data in the first column is merged and the y-axis values are shown in a new column. Additional rows might be added if the x-axis values of the curves do not correspond. In this case, if the corresponding option is enabled, the missing values in the curve data are interpolated.

The column header menu allows to re-organize the data:

- The x-axis data must remain in the first column but the other columns can be permuted. The columns to be moved can also be dragged to their new position.
- The x-axis can be filtered to show just a part of the available rows.
- The first column can be sorted in ascending or descending order.
- Rows and columns can be inverted such that each new curve appears as a new row.

8.2 Working with n-dimensional Data

BeSpice Wave supports n-dimensional data. In this case n different variables are swept. The x-axis vectors are replaced by n-dimensional matrices defined by n vectors. Each curve holds m values where m is the product of the sizes of all n vectors forming the n-dimensional matrix.

Data of this form might issue from lab measurements.

A spreadsheet allows to work with n-dimensional data if the option “Accept several sweep variables” is checked in BeSpice Wave's configuration.

When displaying n-dimensional data, the first n columns in the spreadsheet hold the values of the n sweep variables. These columns are automatically sorted. Identical values are grouped and an expander symbol allows to collapse or expand a group of identical values.

The order of the sweep variables can be changed by dragging a column representing a sweep variable to its new location. In this case the spreadsheet is entirely re-organized. The columns are re-sorted and

the expander symbols are moved to the new appropriate positions.

8.3 Exporting Data

The data in a spreadsheet can be exported to a data file. Currently comma-separated values or as html files can be generated. These files can then be read by other tools such as text editors or table calculation software.

To export a spreadsheet select the item “Export As Data File” from the “File” menu. A dialog box for selecting the file name, the file type and some export options will open.

This procedure can be simplified by selecting the columns to be exported and by dragging them to the external tool. The “copy and paste” mechanism can also be used. In this case the options defined for the last successful “Export As Data File” action are re-used. This concerns for example the used separator for comma-separated values.

8.4 Configuring the Spreadsheet Page

The appearance of the spreadsheet page can be configured in BeSpice Wave's configuration menu. Therefore select “Tools→Configure” from the menu bar. The section “Spreadsheet Preferences” allows to define the used fonts, the precision and other aspects of the spreadsheet. Options that affect the structure of the spreadsheet such as “Split Complex Curves” only apply to newly opened spreadsheet pages.

In addition to this the current spreadsheet page can be modified from the “Spreadsheet” menu in the menu bar. Modifications made here only apply to the currently active spreadsheet page.

CHAPTER 9 MOUSE AND KEYBOARD CONTROLS

The following table gives an overview over the most important mouse actions in a plot window:

Auto-scroll	middle mouse button, menu “View” or keyboard shortcut.
Scroll	Mouse wheel when over scroll bars, menu “View”, tool bar buttons or keyboard arrows. The plot's scroll bars can be dragged.
Pan	Right mouse button while pressing the “CTRL” key.
Zoom X	Mouse wheel when over x-axis, menu “View” or keyboard shortcut.
Zoom Y	Mouse wheel when over one of the y-axes, menu “View” or keyboard shortcut.
Zoom XY	Mouse wheel, menu “View” or keyboard shortcut.
Rectangular zoom X	Draw zoom box when over x-axis, draw zoom box with height = 0, draw zoom box and press “SPACE” several times.
Rectangular zoom Y	Draw zoom box when over one of the y-axes, draw zoom box with width = 0, draw zoom box and press “SPACE” several times.
Rectangular zoom	Draw zoom box, press “SHIFT” if the mouse cursor is over a curve.
Rectangular un-zoom	Draw zoom box while pressing the “CTRL” key. Draw zoom box and press “SPACE” several times.
Fit curves to window (un-zoom)	Menu “View”, keyboard shortcut or un-zoom icon in plot window.
Show or hide legend	Menu “Plot”, keyboard shortcut or legend icon in plot window.
Activate cursors	Menu “Measurement”, keyboard shortcut or cursor icon in plot window.
Context menu	Right mouse click. Close to a curve or a legend item the curve context menu is shown. Over an annotation the annotation context menu is shown.

CHAPTER 10 CONFIGURING BESPICE WAVE

Many features and BeSpice Wave's appearance can be configured by selecting the menu “Tools→Configure”. Among others the following aspects can be changed:

- Key bindings
- Used fonts
- Colors used for drawing axes and grids
- Default values for curve styles
- Behavior for curve highlighting
- Symbol sizes
- Cursor preferences
- Grid preferences

CHAPTER 11 INTERACTIVE MODE

The interactive mode for the BeSpice Wave allows to integrate the viewer into your own applications and tools. To activate the interactive mode the option “--socket <connection_ip> <connection_port>” or “--command_file <file_name>” has to be added when launching BeSpice Wave from the command line.

When using the interactive mode the user has still full control of the tool. That means that curves or data sections can be added or removed, plot windows can be closed or BeSpice Wave might be closed by the user. As a consequence some interactive commands might fail.

Script languages offer appropriate wrapper functions to handle this mechanism efficiently.

All parameters containing white space characters must be enclosed by double quotes.

The commands always return a code indicating the success or the failure of the operation. The code “0” is returned on success and “1” on failure. For some operations such as parsing a file the interactive command interface does not wait until the operation has been completed. Instead the command is appended to the execution queue. In this case the code “2” for “queued” is returned.

This chapter describes the most important available interactive commands. The interactive command “help” will list additional commands.

11.1 Commands Controlling BeSpice Wave

The following commands allow to control BeSpice Wave.

11.1.1 help

Syntax: help <command>

If the argument <command> is empty, this command returns the list of all available commands. Otherwise it returns a help string for the command specified by <command>.

11.1.2 generate_documentation_from_help

Syntax: generate_documentation_from_help <file_name>

This command dumps all commands and their help strings to a text file.

11.1.3 close

Syntax: close

The viewer is closed. All open files are closed. A message box is opened if some files need to be saved. If BeSpice Wave is used as a widget the “close_button_activated” callback function is called after the unsaved files have been handled.

11.1.4 get_status

Syntax: get_status

The purpose of this command is to verify the current status of the command execution queue. The

returned value is “af_code_queued”, when BeSpice Wave is currently executing a command, “af_code_error” if the previous command execution failed and “af_code_ok” otherwise.

11.1.5 wait

Syntax: wait <time_in_seconds>

This command instructs BeSpice Wave to wait for “time_in_seconds” seconds before processing the next command. The time is given as fixed point real value.

11.1.6 execute_command_file

Syntax: execute_command_file <file_name>

This command instructs BeSpice Wave to execute the commands listed in the text file <file_name>. The commands in this file must be separated by newline characters. A command can't be split into several lines.

11.1.7 read_configuration_file

Syntax: read_configuration_file <file_name>

This command reads the file <file_name>. This file contains user settings such as keyboard shortcut definitions and the list of previously opened files. If BeSpice Wave is used as a widget the configuration file is not loaded and saved automatically. If the argument <file_name> is omitted, the default file location is used.

11.1.8 save_configuration_file

Syntax: save_configuration_file <file_name>

This command saves the user's configuration to <file_name>. If <file_name> is omitted, the default file location is used. This operation is performed automatically if BeSpice Wave is not used as a widget.

11.1.9 get_configuration_file_name.

Syntax: get_configuration_file_name

This command returns the name of the currently used configuration file.

11.1.10 set_option.

Syntax: set_option <option_name> <argument_1> <argument_2> ...

This command allows to modify the appearance or the behavior of the tool. The following options can be set:

- “enable_status_bar [1|0]”. This option switches the status bar on or off. The status bar is the small temporarily shown window showing the tool's progress or error messages.
- “tool_bar_style [permanent_left|permanent_right|permanent_top|in_sidebar|temporary|none]”. This option alters how the tool bar is shown. It can be shown temporarily only if the sidebar style is either temporary_left or temporary_right.

- “use_notebook_with_tabs [1|0]”. This options activates or deactivates tabs on the plot pages. If no tabs are shown, The pages can be switched using a drop-down menu in the tool bar.
- “use_curve_names_for_internal_dnd [1|0]”. If this action is enabled, curve names are passed as a string for drags from a BeSpice Wave widget to another window of the using application. See the BeSpice Wave Widget user guide for more details.
- “side_bar_style [permanent_left|permanent_right|temporary_left|temporary_right|none]“. This options alters the way the sidebar is shown. It can be shown permanently on the left or on the right of the main window. Further it can be shown dynamically when requested by the user or it can be hidden.

11.2 Commands for Setting Options

The following commands allow to define options and customize the appearance of BeSpice Wave:

11.2.1 set_option browser_colorize_curves background

Syntax: set_option browser_colorize_curves background <none|text|background>

This command allow to define how colorized curves appear in the browser. If “text” is set, the curve names are also colorized. If “background” is set, the curve names are printed on a colorized background.

11.3 Commands for File Operations

The following commands allow to open and close files in the viewer :

11.3.1 open_file

Syntax: open_file <file_name>

This command opens the file <file_name> and parses it.

11.3.2 reload_file

Syntax: reload_file <file_name>

This command reloads the file "file_name" and parses it. All shown curves are updated automatically.

11.3.3 close_all

Syntax: close_all

This command closes all open files and sections.

11.3.4 close_file

Syntax: close_file <file_name>

This command closes the file <file_name> and all associated sections and curves.

11.4 Commands for Retrieving Curve Information

The following commands allow to retrieve information on the parsed files sections and curves. These commands return two strings separated by a whitespace character. The first string is “0” on success and “1” on failure. The second string represents the returned value. These value are enclosed in quotes as they might contain whitespace characters.

11.4.1 get_number_of_sections

This commands returns the number of open sections. Each opened file might generate more than one section.

11.4.2 get_section_name

Syntax: get_section_name <section_index>

This command returns the name of a section for a given section index. The section index can take the values 0 .. N-1 where N is the value returned by get_number_of_sections. The section index is not reliable to identify a section as the user can close any section at any time from the graphical user interface.

11.4.3 get_section_file_name

Syntax: get_section_file_name <section_name>

This command allows to retrieve the name of the file that generated the section “section_name” where “section_name” must have been obtained using get_section_name.

11.4.4 get_section_number_of_curves

Syntax: get_section_number_of_curves <section_name>

This command allows to obtain the number of curves defined in the section “section_name”.

11.4.5 get_curve_name

Syntax: get_curve_name <section_name> <curve_index>

This command allows to obtain the name of the curve defined in the section “section_name” and identified by its index. “curve_index” can take all values 0 .. N-1 where n is the values returned by get_section_number_of_curves.

11.4.6 get_curve_x_unit

Syntax: get_curve_x_unit <section_name> <curve_name>

This command allows to obtain the x-axis unit for the curve “curve_name” defined in section

“section_name”.

11.4.7 get_curve_y_unit

Syntax: get_curve_y_unit <section_name> <curve_name>

This command allows to obtain the y-axis unit for the curve “curve_name” defined in section “section_name”.

11.5 Commands for Handling Plots

The following commands allow to add and remove plots and to control which page and which plot is currently shown.

11.5.1 add_plot

Syntax: add_plot <plot_name> <plot_type> <flag_new_page>

This command adds a new plot to the plot window. <plot_name> will be used to identify it in other commands. The values allowed for <plot_type> are “analog” and “spreadsheet”. <flag_new_page> can take the value 1 if a new page should be used or the value 0 if the current page should be split to generate a new plot. The names of all generated plots are returned.

11.5.2 make_plot_visible

Syntax: make_plot_visible <plot_name>

This command makes sure the plot “plot_name” is visible. Therefore the page shown in the plot window might be changed and scrolled accordingly.

11.5.3 get_number_of_plots

Syntax: get_number_of_plots

This command returns the number of plots shown in the current and in all hidden plot pages.

11.5.4 get_plot_name

Syntax: get_plot_name <plot_index>

This command returns the name of the individual plot identified by its index “plot_index”. Plot index ranges from 0 to N-1 where N is the integer value returned by get_number_of_plots.

11.5.5 get_current_plot_name

Syntax: get_current_plot_name

This command returns the name of the currently active plot.

11.5.6 close_plot

Syntax: close_plot <plot_name>

This command closes the plot “plot_name”. If a page with multiple plots is concerned, the plot is cleared. Then all empty plots in the same row and column are removed.

11.5.7 clear_plot

Syntax: clear_plot <plot_name>

This command removes all curves from the plot “plot_name”.

11.5.8 maximize_plot

Syntax: maximize_plot <plot_name>

This command maximizes the plot “plot_name” if it is shown in a page with multiple plots. All other plots in this page are temporarily hidden.

11.5.9 minimize_plot

Syntax: minimize_plot <plot_name>

This command shows all plots in the multi-plot page containing “plot_name”.

11.6 Commands for Showing Curves

The following commands allow to show curves in a plot window and to control the plot window. They return “0” on success and “1” on failure.

11.6.1 highlight_curve

Syntax: highlight_curve <section_name> <curve_name> <flag_highlight>

This command highlights the curve “curve_name” from section “section_name” in all plots it is currently shown. If “flag_highlight” is “0” the curve will be un-highlighted.

11.6.2 add_curve_to_plot_by_name

Syntax: add_curve_to_plot_by_name <plot_name> <section_name> <curve_name> <flag_clear>

This command adds the curve “curve_name” from section “section_name” to the plot “plot_name”. If “flag_clear” is set to 1 all other curves in the plot will be removed.

11.6.3 add_curve_to_plot_by_index

Syntax: add_curve_to_plot_by_index <plot_name> <section_name> <curve_index> <flag_clear>

This command adds the curve with the index <curve_index> from section <section_name> to the plot <plot_name>. This command should be used instead of add_curve_to_plot_by_name if the curve names in the concerned section are not unique. The first curve indices are reserved to the sweep

variables. So in general 1 will be the index associated to the first curve in the section.

If <flag_clear> is set to 1 all other curves in the plot will be removed..

11.6.4 colorize_curve_by_index

Syntax: colorize_curve_by_index <section_name> <curve_index> <color> <min_x> <max_x>

This command allows to colorize a curve. The curve is identified by the name of it's section and by it's index. A colorized curve is drawn in the specified color in all plot windows. If min_x and max_x are specified, the curve is only colorized in the x-axis interval [min_x, max_x].

11.6.5 colorize_curve_by_name

Syntax: colorize_curve_by_name <section_name> <curve_name> <color> <min_x> <max_x>

This command allows to colorize a curve. The curve is identified by the name of it's section and by it's name. A colorized curve is drawn in the specified color in all plot windows. If min_x and max_x are specified, the curve is only colorized in the x-axis interval [min_x, max_x].

11.6.6 colorize_curve_by_id_and_index

Syntax: colorize_curve_by_id_and_index <section_id> <curve_index> <color> <min_x> <max_x>

This command allows to colorize a curve. The curve is identified by the identifier of it's section and by it's index. A colorized curve is drawn in the specified color in all plot windows. If min_x and max_x are specified, the curve is only colorized in the x-axis interval [min_x, max_x].

11.6.7 decolorize_all_curves

Syntax: decolorize_all_curves

This command removes colorizing for all curves.

11.6.8 zoom

Syntax: zoom <plot_name> x0 x1 y00 y01 y10 y11

This command defines the visible range of the plot <plot_name>. The x-axis will be visible from x0 to x1. The first y axis will be visible from y00 to y01. The values y10 and y11 act on the second y-axis. They can be omitted if only one axis is used. By default all values are considered being absolute values. However if at least one value is followed by a “%” they are considered being percentages of the maximum range of the current values. Values such as -10% or 200% are allowed.

11.6.9 set_cursor

Syntax: set_cursor <plot_name> x0

This command adds a cursor to the plot <plot_name> the x-value can be given as absolute value or as percentage.

11.6.10 set_curve_style

Syntax: set_curve_style <section_name> <curve_name> <line_style_real> <symbol_real>
<width_real> <red_real> <green_real> <blue_real> <line_style_imag> <symbol_imag>
<width_imag> <red_imag> <green_imag> <blue_imag>

This command allows to define a line color, a line style and a symbol to be used for displaying the curve “curve_name” in “section_name”. The parameters “line_style_real” and “line_style_imag” can take the values “no line”, “solid line”, “dotted line”, “dashed line”, “short dashed line”, “data points” or “histogram”. The parameters “symbol_real” and “symbol_imag” can take the values “no symbol”, “crosses”, “circles”, “rectangles” or “histogram”. The line colors are defined by the parameters “red_real”, “green_real”, “blue_real”, “red_imag”, “green_imag” and “blue_imag”. They can take any integer value between 0 and 255.

Calling this function doesn't alter the curve style and color for curves already plotted.

11.7 Commands for Adding Curve Data

The following commands allow to define sections and curve values. These curves are added to the waveform browser. These curves can be increased by adding points. Once the curves have been generated they can be plotted using the appropriate interactive commands. The following commands return “0” on success and “1” on failure.

11.7.1 add_section

Syntax: add_section <section_name>

A new section is generated and added to the waveform browser.

11.7.2 add_curve

Syntax: add_curve <section_name> <curve_name> <flag_complex> <x_unit> <y_unit> <values>

This command adds a new curve to the section “section_name” and updates the waveform browser window. The section must have been created using the command “add_section”. The value for “flag_complex” can be 1 for a complex curve or 0 for a real curve. Any number of values can be given. For real curves they have to be organized in pairs (x0, y0), (x1, y1), ... and for complex curves they have to be organized in triplets (x0, real(y0), imag(y0)), (x1, real(y1), imag(y1)),

11.7.3 add_points

Syntax: add_points <section_name> <curve_name> <values>

This command adds values to a curve “curve_name” in section “section_name”. The curve must have been created using the command “add_curve”. Any number of values can be given. For real curves they have to be organized in pairs (x0, y0), (x1, y1), ... and for complex curves they have to be organized in triplets (x0, real(y0), imag(y0)), (x1, real(y1), imag(y1)),

11.8 Registering Callback Functions

The following commands are limited to the use of BeSpice Wave in a script environment such as

Tcl/Tk. They allow to define functions for an increased interaction with the calling software.

For a more detailed description, see the BeSpice Wave Widget user guide. The arguments required by the callback functions are also detailed there.

11.8.1 register_callback_function_close_button

Syntax: register_callback_function_close_button <function_name>

This command will register the script function “function_name” as callback function. When the BeSpice Wave's exit button is hit, the corresponding function will be called.

11.8.2 register_callback_function_highlight_curve

Syntax: register_callback_function_highlight_curve <function_name>

This command will register the script function “function_name” as callback function. When a curve is highlighted in BeSpice Wave, the corresponding function will be called

11.8.3 register_callback_function_unhighlight_curve

Syntax: register_callback_function_unhighlight_curve <function_name>

This command will register the script function “function_name” as callback function. When a curve is un-highlighted in BeSpice Wave, the corresponding function will be called

11.8.4 register_callback_function_activate_curve

Syntax: register_callback_function_activate_curve <function_name>

This command will register the script function “function_name” as callback function. When a curve is activated by double-clicking on it, the corresponding function will be called.

11.9 Example 1: Command File

An example of an interactive command file is located in your software distribution. To execute it change to the corresponding directory

```
> cd analog_flavor/examples/bspwave/example/
```

Run BeSpice Wave with the `--command_file` option

```
> ../../bin/bspwave --command_file bspwave_interactive.txt
```

The interactive commands from the file will be executed in BeSpice Wave.

11.10 Example 2: Use in a Python Script

Two examples of the use of BeSpice Wave's interactive mode from a Python script are located in your software distribution. To execute it change to the corresponding directory

```
> cd analog_flavor/examples/bspwave/python_test
```

Run the script:

```
> ./bspwave_socket.py
```

Now you can type commands that will be executed in BeSpice Wave :

```
> open_file ../example/fourbitadder.spi3
> add_plot "plot 2" analog 0
> add_curve_to_plot_by_name "plot 2" fourbitadder.spi3 v(1)
> add_curve_to_plot_by_name "plot 2" fourbitadder.spi3 i(vcc)
> zoom "plot 2" 0% 150% -10% 110% -10% 110%
> set_cursor "plot 2" 10%
> set_cursor "plot 2" 70%
> hide
> show
> x
```

The example “bspwave_socket.py” establishes a socket connection for communication.

CHAPTER 12 TROUBLESHOOTING

12.1 The Log Window

Selecting the item “Log” from the menu or the tool bar shows all errors and warnings emitted by the waveform parser and BeSpice Wave. If things don't look as you expected you should check the log. Here you can find information on lines ignored by the parser or included files that were not found etc.

12.2 License Problems

If BeSpice Wave does not run properly, this might be due to a problem with your license. Information on the currently used license file and its expiry date can be displayed by selecting “Tools→License→Information” from the menu bar. The dialog also shows the location of BeSpice Wave's configuration file. Your license might be linked to particular hardware or a license server. If you have no valid license to run BeSpice Wave please generate a hardware file by selecting “Tools→License→Generate Hardware File” from the menu bar and contact your support.

CHAPTER 13 LIMITATIONS

The current version of BeSpice Wave has some limitations:

CHAPTER 14 CREDITS

Your Analog Flavor Software distribution is based on several software libraries and tools:

- The wxWidgets Toolkit is published under the wxWindows Library Licence. For details see <http://http://www.wxwidgets.org/>.
- The QT cross platform UI framework is available under the GNU Lesser General Public License (LGPL). A copy of the LGPL license is included in your software package. For more details see <http://qt-project.org>. Please contact our support to receive the sources that were used to compile the Qt version used for your software package.
- Information on Scintilla (Copyright by Neil Hodgson) is available at <http://www.scintilla.org>.
- Boost Math is published under the Boost Software License. For more details see <http://www.boost.org>.
- The Python programming language is published by the Python Software Foundation under an open source license. See <http://www.python.org/> for details.
- The Tcl/Tk programming language is published under a BSD-style open source license. See <http://www.tcl.tk> for more details.
- The used icons were created by Everaldo Coelho and published under the GNU Lesser General Public License (LGPL). A copy of the LGPL license is included in your software package. See <http://www.veraldo.com> for more details. Please contact our support to receive a copy of the icon library.
- The SQL database engine Sqlite has been dedicated to the public domain by the authors. For more details see <http://sqlite.org>.
- Cmake is used for generating makefiles for different platforms and compilers. For details see <http://www.cmake.org>.
- A part of the Analog Flavor documentation is created using doxygen. For more details see <http://www.stack.nl/~dimitri/doxygen.html>.
- The xml parser library TinyXML is used for parsing all xml files. For more details see <http://www.grinninglizard.com/tinyxml/index.html>.
- The fft is computed using the Kiss FFT library. More information about this library is available from <https://github.com/mborgerding/kissfft>.
- We use the zlib to compress data. For more information see the [zlib website](#).
- The PCRE library is used to parse regular expressions. See <http://www.pcre.org/> for more information about the library.

Some BeSpice Wave packages contain extensions implemented as plug-ins. These plug-in libraries are based on the following software libraries.

- The BeSpice Waveform Parser can be extended to read seismic recordings in SEED format. Therefore it makes use of the miniSEED library which is published by the Incorporated Research Institutions for Seismology (IRIS) under the GNU Lesser General Public License

(LGPL). A copy of the LGPL license is included in your software package. See <http://seiscode.iris.washington.edu/projects/libmseed> for more details. Please contact our support to receive a copy of the icon library.

- The BeSpice Waveform Parser can be extended to read FST and LXT waveform formats. These formats have been developed for GTKwave. The extension uses the GTKwave's waveform parsers. Visit <http://gtkwave.sourceforge.net> for more information.
- The BeSpice Waveform Parser can be extended to read HDF5 files. HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 2006 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by The Board of Trustees of the University of Illinois. See <https://www.hdfgroup.org/> for more details.

Additional information and copies of the license files are available from our website in the downloads section.

CHAPTER 15 COMMA-SEPARATED VALUE FORMAT

BeSpice Wave supports reading of comma separated values from *.csv files. The supported formats allow to define files with several data sections or sweep variables. Some example files have been included to your Analog Flavor software package. When generating csv data for BeSpice Wave a similar format should be used.

The file “analog_flavor/examples/bspwave/example/example_1.csv” defines 3 sweep variables (vbat, rload and vctrl). The slowest varying sweep variable is the first column, the fastest varying sweep variable is in the 3rd column. The values for faster varying sweep variables must exactly repeat for each set of slower varying sweep variables. The file parser determines the number of sweep variables by analyzing the values in the parsed columns.

The file “analog_flavor/examples/bspwave/example/example_2.csv” defines 3 data sections with varying values for the run parameter vbat and one sweep variable per data section.

The file “analog_flavor/examples/bspwave/example/example_3.csv” is an ordinary csv file defining a single data section with a single sweep variable.

Alphabetical Index

annotations.....	24
auto-scroll.....	20
comma separated values.....	52
command file.....	11, 47
complex curves.....	21
configuration file.....	49
connecting plots.....	22
csv.....	52
editing an annotation.....	24
export plot as file.....	25
export spreadsheet as file.....	37
highlighting.....	22
interactive mode.....	40
legend.....	21
license.....	9
license problems.....	49
log.....	49
log window.....	49
logarithmic scale.....	21
name annotation.....	24
pan.....	21
Python.....	47
socket.....	11
text annotation.....	24
value annotation.....	24
view ports.....	21
zoom.....	19