Manual Simeye

30th September 2019

1 Synopsys

simeye [X options] [-IRLeh] [-l|-m|-c] [-a|-b] [name]

2 Options

Beside the standard X Toolkit options (e.g. to set the display, the geometry, and the foreground color of the program), the following other options may be specified:

-freestateColor *color* **:** Specifies the color for signal values that are in **free state**.

-high_lowColor *color* **:** Specifies the color for signal values that are **logic O** or **logic I**.

-undefinedColor *color* **:** Specifies the color for signal values that are **logic X**.

-voltsColor *color* **:** Specifies the color for **analog** signal values.

-pointerColor *color* **:** Specifies another color for the pointer (default: canvas.foreground).

-std_bg *color* **:** Specifies another color for the standard background (default: grey).

-I : Start *simeye* in input (editing) mode.

-R : Start *simeye* in input (editing) mode and set circuit name and command file name in order to prepare a simulation run.

-L : Load window settings each time when new signals are read.

-e : Do not add or use extension of specified *name*.

-h : Display only help information.

-l : Use **sls-logic** as simulation type (default).

-m : Use **sls-timing** as simulation type.

-c : Use **spice** as simulation type.

-a : Use **analog** representation for **sls** signals.

-b : Use **digital** representation for **sls** signals (default).

3 Description

Simeye is an X-window (OSF/Motif) based simulation user interface for viewing the results of the *sls(1ICD)*,and *spice(1ICD)* simulators. The program can be used to inspect the graphical representation of the output signals of the simulators and it can be used to edit input signals that are described in the SLS command file format (see: User's Manual). Moreover, from *simeye* simulations can be run by starting either the *sls* or *spice* simulator (Note: *spice* via the script *nspice(1ICD)*).

The following program argument may be specified:

name **:** This name specifies the name of the cell or file for which simulation signals are displayed and/or edited.

The *name* argument should be a file name. If the *name* contains no '.' in it, the name is assumed to be a cell name. In that case a standard extension is added, based on the specified option (precedence order: **-I**, **-c**, **-a**). Give option **-e**, if no extension must be added. If a file name is specified with a standard extension (from first '.' position), this extension type is used (has precedence of options). Give option **-e**, if you do not want this feature.

The standard extensions are:
".res" for **sls digital** (output) signals (option **-b**).
".plt" for **sls analog** or waveform (output) signals (option **-a**.
".ana" for **spice** analog (output) signals (option **-c**).
".cmd" for simulator **commands** and input signals (option **-I**).

Note that the spice output file should contain the signals of a transient analysis in tabular format. When running the program *spice* directly this is achieved by using the card ".print tran ..." in the spice input file. When running *nspice*, this is achieved by using sls "plot ..." commands in the command file.

4 Commands

Commands are organized in six PullDown Menu's (i.e. *File*, *Edit*, *Simulate*, *View*, *Options* and *Help*). This MenuBar buttons can also be activated by typing the <**Meta**> key and the mnemonic character (the mnemonic is the character that is underlined). Commands in the PullDown Menu's can also be activated by only typing a mnemonic character (Note: This mnemonic can be turned off, if you are using accelerator keys).
Under the MenuBar there is a StatusArea, which shows the name of the "editing" command file or "viewing" simulation results file, and the simulation setup, and most right the active command (if any). You can also turn off an active command by clicking on that field.

The commands in the *File* PullDown Menu are:

New : Start editing of a completely new command file or cell "cmd" stream after an OK-confirmation has been given.

Open : Read *cell* command file or stream for editing. If this *cell* or this command file does not exist, *simeye* will try to read a default command file called "simeye.def.d". First, it tries to read this file from the current directory and second it tries to read this file from the process directory.

Save : Save (update) the edited command file or stream under the same file or cell name after an OK-confirmation has been given.

Save As : Save the edited command file *cell* under a different file or cell name after an OK-confirmation has been given.

Read : Read a new *cell* signal (simulation results) file or stream. If the 'read' **Digital** or **Analog** setting is activated an *sls* "res" or "plt" output file/stream is read. If the 'read' **Spice** setting is activated the *spice* "ana" output file is read.

Read Append : Append the *cell* signal (simulation results) file/stream to the current signals. Only Digital signals can be appended to Digital signals and Analog signals to Analog signals (see *Read*).

Hardcopy : Generate a hardcopy of the current screen and send it to the printer by making an X Window pixel dump (fast, but of low resolution).

Print : Make a PostScript-file of the current screen and send it to the printer after an OK-confirmation has been given (slow, but of high resolution).

Exit : Exit the program after an OK-confirmation has been given.

The commands in the *Edit* PullDown Menu are:

Add : Create a new signal. The user has to enter the name of the new signal, and the signal will be placed at the bottom of the window.

Change : Change (edit) a particular signal by inserting a new logical level for a particular time interval (t1, t2). The signal description may eventually already be defined for this interval. The insertion of the new logical level is done in two steps: During the first click of the mouse the signal, the new logical level and t1 are selected. During the second click t2 is selected and the new signal description is drawn. To insert an interval that has a free state one should hold the <Shift> key pressed down while making the first selection.

Delete : Delete one or more signals by selecting them with the cursor.

EraseAll : Erase all signals after an OK-confirmation has been given.

Move : With this command the order of the signals can be changed or one signal can be placed over another signal in order to compare them. First the signal that is moved or that is placed over another signal is selected, and then the new position of the signal is selected. To place the selected signal over another signal one should hold the <Shift> key down while selecting the new position. To remove a signal that is placed over another signal, initially select the signal with the <Shift> key held down.

Copy : Copy the signal description from one signal to another signal.

ReName : Change or show the (full) name of the selected signal.

Yank : Store a (part of a) signal description in the buffer.

Put : Insert a copy of the signal description that is in the buffer onto a particular position. The user has to select the signal to which the contents of the buffer is added and the time from which on the new signal description is valid. The new signal description may (partly) override the existing signal description for the selected signal. Furthermore, the user is asked to type how may periods of the stored signal part are added. If a value -1 is specified, the selected signal will become a periodical signal and (seemingly) an infinite number of periods are added. In that case, when the simulation end time is enlarged, the signal description will automatically be extended according to the periodicity of the signal description. A periodical signal is indicated by means of a tilde ($\sim$) immediately to the right of the signal description.

Grid : Specify the smallest unit for the x-axis (= time axis).

Speed : Speed up the signals by some factor. If a value < 1 is specified, the signals will be slowed down.

T_end : Update the end time of the input signals (= end time of simulation).

The commands in the *Simulate* PullDown Menu are:

Run : Start the simulation run as set in Prepare menu. A simulation may be aborted by typing <Control>C in the *simeye* window (place the cursor in the *simeye* window and type the character C while holding down the <Control> key).

Edit : Edit the current command file set in the Prepare menu.

Prepare : Prepare a simulation by using either the *sls* or the *spice* simulator after an OK-confirmation has been given.
When 'type' **spice** is set a *spice* simulation is performed. The *spice* simulator is run by calling the program *nspice*. After simulation the 'ana' *spice* output file is read.
When 'type' **sls-logic** (level 1 or 2) or **sls-timing** (level 3) is set, an *sls* simulation at that level is performed. When the *sls* simulator is run, also intermediate simulation results are displayed. At the end the final simulation results are displayed. Which results are displayed for *sls* ('res' or 'plt') depends on the 'read' **Digital** or **Analog** setting.

Command : Display the current simulation command.

The commands in the *View* PullDown Menu are:

Measure : Move a vertical scanline over the display window, according to the position of the pointer, and show the value of the corresponding x position (and y position). When clicking the middle (or right) button of the mouse, one may toggle between displaying in the right margin (if the number of signals is not too large) the y values of the signals at the selected x position. To store the value of a particular position, click the left button of the mouse. This position will then be subtracted from the next positions of the scanline, so as to measure delay times and/or voltage differences.

Redraw : Redraw the current window.

Values : Display the value points (for analog simulation results).

Full : Draw all present signals from the beginning of the simulation time till the end of the simulation time.

ZoomIn : Zoom in on the current window. When the <Shift> key is held down and the window contains only one analog signal, also a zoom-in on the y-axis of the signal is performed. The ScrollBars can be used to move the current window left-, right-, up- or down-wards respectively.

ZoomOut : Zoom out on the current window. Use zoom out if you want to add new signal states (*Change* command) to signals on the right side of the screen.

Load Window : Load the current window settings (i.e. start-time, stop-time and the names of the signals that are displayed) from a file (default: "simeye.set"). The name of this file can be specified in the configuration file.

Save Window : Save the current window settings (i.e. start-time, stop-time and the names of the signals that are displayed) in a file (default: "simeye.set"). The name of this file can be specified in the configuration file.

The commands in the *Options* PullDown Menu are:

DetailZoomON : When the DetailZoomON option is set (see also DETAIL_ZOOM_ON) the zoom-in function is also defined for the y-axis of a (single analog) signal.

AutoLoadWin : When the AutoLoadWin option is set (or by the command line option **-L**), the program *simeye* loads the saved window settings each time a *Read* or *Simulate* command is performed. See also the *Load Window* and *Save Window* commands.

AutoSaveEdit : When the AutoSaveEdit option is set the program *simeye* saves automatically the command file (if changed) when a *Run* command is given.

The working of the *Help* MenuBar facility:

Manual : When the Manual command is clicked, the program *icdman* with the manual page of *simeye* is popped up in a window. Type 'q' to quit this program.

5 Commandfile

For simulation with both *sls* and *spice*, *simeye* uses a command file called '*cell*.cmd' This file should contain a description of the input signals, values for the simulation control variables, and a listing of the terminals for which output should be generated. This is explained in more detail in the users manuals of the *sls* simulator and the *spice* simulator.
The 'set' commands (signals) in this file/stream may be edited by enabling the *Edit* PullDown Menu of *simeye*. The new signal descriptions will then be written back to the command file when updating the command file. However, when the file contains a 'set' command that is followed by the keyword 'no_edit' between comment signs, the signal description of this 'set' command can not be modified. Additionally, when the command file contains on a separate line between comment signs the keyword 'auto_set', *simeye* will automatically turn-on or turn-off these 'non-editable' 'set' commands according to whether or not the node the command refers to is part of the terminal list of the cell that is simulated (by *Simulate* command). This feature may for example be useful to specify default signals for one or more input terminals.
When the file contains on a separate line between comment signs the keyword 'auto_print' ('auto_plot'), *simeye* will automatically add a 'print' ('plot) statement to the command file for each terminal of the cell that is simulated. The new 'print' ('plot') commands will have a keyword 'auto' between comment signs following the keyword 'print' ('plot').
An example of a default command file that can be used for *sls* and *spice* simulations is:

```
/* auto_set */
set /* no_edit */ vdd = h*~
set /* no_edit */ vss = l*~
set /* no_edit */ phi1 = (l*110 h*80 l*10)*~
set /* no_edit */ phi2 = (l*10 h*80 l*110)*~
option sigunit = 1n
option outacc = 10p
option simperiod = 4000
option level = 3
/*
*%
tstep 0.1n
trise 0.5n
tfall 0.5n
*%
*%
*/
/* auto_print */
/* auto_plot */
```

6 Configurationfile

At start-up of the program, *simeye* will read some information from a configuration file called '.simeyerc'. First, it tries to find this file in the current directory. Second, it tries to find this file

in the home directory of the user. Thirdly, it looks for this file in process directory. The configuration file may contain the following keywords, followed by a specification on the same line if the keyword ends with ':'.

SLS: Specifies the command for running the *sls* simulator.

SLS_LOGIC_LEVEL: Specifies the level of simulation when '**sls-logic**' is selected (use 1 or 2).

SLS_LOGIC_SIGNAL: Specifies the default signal representation for **sls-logic** simulations (use A for **Analog** or D for **Digital**).

SLS_TIMING_SIGNAL: Specifies the default signal representation for **sls-timing** simulations (use A for **Analog** or D for **Digital**).

SPICE: Specifies the command for running the *spice* simulator (use *nspice* or a derivative of it).

XDUMP_FILE: Specifies the name of the X Window dump file that is generated when the *Hardcopy* command is clicked.

PRINT: Specifies the command that is executed when the *Hardcopy* command is clicked in order to process the X Window dump file (e.g. to convert the window dump to PostScript and to send the output to a laser-printer).

PRINT_LABEL: Specifies an optional label that is placed in upper left corner of the display window when an X Window dump is generated.

SETTINGS_FILE: Specifies the name of the file in (from) which the window settings are stored (loaded) when using the *Save Window* command (*Load Window* command or **-L** option).

DETAIL_ZOOM_ON If this keyword is specified in the configuration file, the zoom-in function is also defined for the y-axis of a (single analog) signal. In this case it is not necessary to hold the <Shift> key down (see command *ZoomIn*).

TRY_NON_CAPITAL_ON If this keyword is specified in the configuration file and *simeye* fails to open a command file that starts with a capital letter, the program will try to open the same command file but now starting with a non-capital letter. If this succeeds and when performing a simulation, *simeye* will first run the program *arrexp* on a copy of the command file to expand one dimensional array node names into single node names (e.g. a[1..3] is converted into a_1_ a_2_ a_3_).

In the above specifications, the strings '$circuit' and '$stimuli' must be used to refer to the selected cell or file name for simulation commands, '$file' may be used to refer to the current file on screen (use '$cell' for this name without extension (maximum = 22 chars)), '$date' and '$time' may be used to refer to the current date and time, and '$user' may be used to refer to your login name.

Example of a configuration file (default values are shown):

```
SLS: sls $circuit $stimuli
SLS_LOGIC_LEVEL: 2
```

```
SPICE: nspice $circuit $stimuli
PSTAR: npstar $circuit $stimuli
XDUMP_FILE: simeye.wd
PRINT: xpr -device ps -output $cell.ps simeye.wd; lp $cell.ps;
       rm simeye.wd $cell.ps
PRINT_LABEL: $user  $file  $date  $time
SETTINGS_FILE: simeye.set
```

7 Files

.simeyerc	(default) configuration file
HOME/.simeyerc	(altern.) configuration file
ICDPATH/share/lib/process/*process*/.simeyerc	(altern.) configuration file}
ICDPATH/share/lib/process/*process*/.simeyerc	(altern.) configuration file}
cell.res	(default) input file (SLS logic results)
cell.plt	(opt.) input file (SLS analog results)
cell.ana	(opt.) input file (SPICE analog results)
cell.cmd	(opt.) command file
.cxx	temporary files
simeye.def.d	(default) template for command file
ICDPATH/share/lib/process/*process*/simeye.def.d	(altern.) template for command file
ICDPATH/share/lib/process/*process*/simeye.def.d	(altern.) template for command file
ICDPATH/share/lib/.X2PSfontmapfile	for PostScript output
simeye.eps	(default) print output file
simeye.set	(default) window-settings file
sim.diag	simulation diagnostics
pr.diag	print diagnostics

8 X-defaults

Some examples of ".Xdefaults" values are:

```
simeye*.borderColor: white
simeye*StatusArea*.foreground: yellow
simeye*cellDialog*.foreground: cyan
simeye*iDialog*.foreground: red
simeye*pDialog*.foreground: red
simeye*qDialog*.foreground: red
simeye*commands*.foreground: white
simeye*commands*.background: black
simeye*canvas.background: black
simeye*canvas.foreground: white
simeye*canvas.height: 400
simeye*canvas.width: 800
```

```
simeye.freestateColor: red
simeye.high_lowColor: cyan
simeye.undefinedColor: blue
simeye.valuesColor: red
simeye.voltsColor: green
simeye.pointerColor: yellow
simeye.std_bg: grey80

Use the following set of accelerator key bindings:

simeye.keymnemonic: False
simeye*Add.accelerator: <Key>a
simeye*Add.acceleratorText: a
simeye*Change.accelerator: <Key>c
simeye*Change.acceleratorText: c
simeye*Command.accelerator: Shift<Key>c
simeye*Command.acceleratorText: C
simeye*Copy.accelerator: <Key>o
simeye*Copy.acceleratorText: o
simeye*Delete.accelerator: <Key>d
simeye*Delete.acceleratorText: d
simeye*Edit.accelerator: <Key>e
simeye*Edit.acceleratorText: e
simeye*EraseAll.accelerator: Ctrl<Key>e
simeye*EraseAll.acceleratorText: ^e
simeye*Exit.accelerator: <Key>x
simeye*Exit.acceleratorText: x
simeye*Full.accelerator: <Key>f
simeye*Full.acceleratorText: f
simeye*Grid.accelerator: <Key>g
simeye*Grid.acceleratorText: g
simeye*Hardcopy.accelerator: Ctrl<Key>h
simeye*Hardcopy.acceleratorText: ^h
simeye*Load Window.accelerator: Shift<Key>l
simeye*Load Window.acceleratorText: L
simeye*Manual.accelerator: <Key>F1
simeye*Manual.acceleratorText: F1
simeye*Measure.accelerator: Shift<Key>m
simeye*Measure.acceleratorText: M
simeye*Move.accelerator: <Key>m
simeye*Move.acceleratorText: m
simeye*New.accelerator: Ctrl<Key>n
simeye*New.acceleratorText: ^n
simeye*Open.accelerator: Ctrl<Key>o
simeye*Open.acceleratorText: ^o
simeye*Prepare.accelerator: <Key>p
simeye*Prepare.acceleratorText: p
```

```
simeye*Print.accelerator: Ctrl<Key>p
simeye*Print.acceleratorText: ^p
simeye*Put.accelerator: <Key>u
simeye*Put.acceleratorText: u
simeye*ReName.accelerator: <Key>n
simeye*ReName.acceleratorText: n
simeye*Read Append.accelerator: Ctrl<Key>q
simeye*Read Append.acceleratorText: ^q
simeye*Read.accelerator: Ctrl<Key>r
simeye*Read.acceleratorText: ^r
simeye*Redraw.accelerator: Ctrl<Key>f
simeye*Redraw.acceleratorText: ^f
simeye*Run.accelerator: <Key>r
simeye*Run.acceleratorText: r
simeye*Save As.accelerator: Ctrl<Key>a
simeye*Save As.acceleratorText: ^a
simeye*Save Window.accelerator: Shift<Key>s
simeye*Save Window.acceleratorText: S
simeye*Save.accelerator: Ctrl<Key>s
simeye*Save.acceleratorText: ^s
simeye*Speed.accelerator: <Key>s
simeye*Speed.acceleratorText: s
simeye*T_end.accelerator: <Key>t
simeye*T_end.acceleratorText: t
simeye*Values.accelerator: <Key>v
simeye*Values.acceleratorText: v
simeye*Yank.accelerator: <Key>y
simeye*Yank.acceleratorText: y
simeye*ZoomIn.accelerator: <Key>z
simeye*ZoomIn.acceleratorText: z
simeye*ZoomOut.accelerator: Shift<Key>z
simeye*ZoomOut.acceleratorText: Z
```

9 See also

sls(1ICD), spice(1ICD), nspice(1ICD),
SLS: Switch-Level Simulator User's Manual,
User's Guide for SPICE, Univ. of California at Berkeley,
X Window System Documentation.