

Synthesis Tools Installation Notes

Version C-2009.06

June 8, 2009

These installation notes present information about installing Synopsys synthesis tools in the following sections:

- [Specific Tools in the Synthesis Group](#)
- [Media Availability and Supported Platforms](#)
- [Memory Requirements](#)
- [Installing the Software](#)
- [Setting Up the User Environment](#)
- [Configuring the Browser for Design Vision Online Help](#)
- [Installing Optional Tools](#)
- [Verifying the Synthesis Tools Installation](#)
- [Limitations](#)

Note:

The installation instructions in this chapter are the most up-to-date available at the time of production. However, changes might have occurred. For the latest installation information, see the product release notes or documentation.

See also <http://www.synopsys.com/Support/Licensing/Installation/Pages/default.aspx> for additional installation and licensing information.

Specific Tools in the Synthesis Group

This installation note covers the following Synopsys synthesis tools:

- Automated Chip Synthesis
- BSD Compiler
- Design Compiler
- Design Vision
- DesignWare
- DFT Compiler
- HDL Compiler (Presto Verilog)
- HDL Compiler (Presto VHDL)
- Library Compiler
- Module Compiler
- Power Compiler
- VHDL Compiler

If you have purchased any of these tools, you must install the synthesis tools suite. These tools are also referred to collectively as the synthesis tool set.

Media Availability and Supported Platforms

The synthesis tools are available by EST download upon initial software release, and at a later date on DVD (or CD depending on image size).

Obtain the appropriate binary executable files based on the operating system you need. [Table 1](#) shows the supported compute platforms, operating systems, Synopsys platform keywords, and windowing environments for this release.

Table 1 Supported Platforms, Operating Systems, and Keywords

Compute Platform	Operating System	Synopsys Platform Keyword	Windowing Environment
x86_64	Red Hat Enterprise Linux v4, 5 ¹	amd64 (64-bit mode) linux (32-bit mode) ²	GNOME
x86_64	SUSE Enterprise Linux v9, 10 ¹	suse64 (64-bit mode) suse32 (32-bit mode)	KDE
x86_64	Solaris 10	x86sol64 (64-bit mode) x86sol32 (32-bit mode)	CDE
IBM RS6000	AIX 5.3	rs6000 (32-bit mode) aix64 (64-bit mode)	CDE
x86	Red Hat Enterprise Linux v4, 5 ¹	linux (32-bit mode) ²	GNOME
x86	SUSE Enterprise Linux v9, 10 ¹	suse32 (32-bit mode)	KDE
Sun SPARC	Solaris 9, 10 ¹	sparc64 (64-bit mode) sparcOS5 (32-bit mode)	CDE

1. Binary-compatible hardware platform or operating system. Note, however, that binary compatibility is not guaranteed. See <http://www.synopsys.com/Support/Licensing/Pages/default.aspx> for latest information.

2. The 32-bit (x86) and 64-bit (x86_64) Linux software is binary compatible with the Intel EM64T or AMD Opteron running Red Hat Enterprise Linux. See <http://www.synopsys.com/Support/Licensing/Pages/default.aspx> for latest information.

Memory Requirements

This section describes tool disk and memory requirements in the following subsections:

- [Minimum Memory Requirements](#)
- [Swap Space Requirements](#)
- [Accessing Memory Beyond 2 GB With 32-Bit Synthesis Tools](#)

Minimum Memory Requirements

The synthesis tools have the following minimum memory requirements:

- Physical Memory – 150 MB (1 GB recommended)
- Swap space – 256 MB (2 GB recommended)

For large designs, the expected amount of required memory is approximately 1 million bytes per 2,000 gates.

Swap Space Requirements

The amount of swap space required by the synthesis tools depends on the size and type of each circuit design.

Use the following formula to help you determine the minimum amount of available swap space required for HDL designs:

$16.3 \text{ MB} + (5.9 \times (\text{size of the design in K gates}))$.

For example, a 5K-gate design requires $16.3 + (5.9 \times 5) = 45.8$ MB of available swap space.

Accessing Memory Beyond 2 GB With 32-Bit Synthesis Tools

Synthesis tools can extend memory beyond 2 GB. Note that available memory is calculated as space not used by the OS, the windowing system, or other applications. The following synthesis tools can extend memory beyond 2 GB:

- Design Compiler
- DFT Compiler
- HDL Compiler (Presto Verilog)
- HDL Compiler (Presto VHDL)

- Power Compiler

Note:

Available memory is space not used by the OS, the windowing system, or other applications.

To access memory beyond 2 GB,

1. Make sure your server has at least 4 GB of memory (physical and swap space) available.

Note:

Physical memory equals data size plus stack size, and stack size is used before data size. Therefore setting stack size to a large value causes problems for designs that need to go over 2 GB. If you set the stack size too high, you cannot get enough memory for your data. To check the settings, use the `limit` command at the system prompt.

2. Make sure the system you are using does not have restrictions that prevent you from using more than 2 GB of memory.

3. Create unlimited data size in the shell that you are using: C, Bourne, Korn, or Bash. If there are system-wide limits on the data size you can create, you can remove them or override them. You can do this in one of two ways:

- Enter one of the following commands based on the shell you are using:

For the C shell,

```
% limit datasize 3800000
```

For the Bourne, Korn, or Bash shell,

```
# ulimit -s -d 3800000
```

- Modify the kernel of your server. This approach allows everyone using your server to extend memory beyond 2 GB.

Installing the Software

The synthesis tools use the Synopsys Installer tool, which allows you to use a text script or a graphical user interface (GUI). For information about downloading the Synopsys Installer and synthesis, see the document *Installing Synopsys Tools* at <http://www.synopsys.com/Support/Licensing/Installation/Pages/default.aspx>.

To install the synthesis tool set, follow the procedures described in *Installing Synopsys Tools*. This document provides a Synopsys media installation script. The synthesis tool set is installed in a similar manner.

The synthesis tool set is a stand-alone product and cannot be installed over an existing Synopsys product, including prior versions of the tool set. You must create a new directory for the synthesis tool set.

Setting Up the User Environment

Configuring the synthesis tools is described in the following sections:

- [Specifying the Executable File Location](#)
- [Setting the SNPSLMD_LICENSE_FILE Environment Variable](#)

Specifying the Executable File Location

A platform-independent wrapper script is provided for the synthesis tools. This script automatically determines the operating system platform at runtime, which simplifies the setup required to use the synthesis tools.

The platform-independent wrappers are located at *install_dir/bin* and include the following options:

-32bit | -64bit

Note:

If you select a platform executable file that is not available, you will automatically be switched to an available platform based on your current environment. A warning message will not be issued.

To set up the environment by using the platform-independent wrapper script, add the synthesis tool set bin directory to the `PATH` environment variable.

- If you are using the C shell, add the following line to the `.cshrc` file:

```
set path=(install_dir/bin $path)
```

- If you are using the Bourne, Korn, or Bash shell, add the following line to the `.profile`, `.kshrc`, or `.bashrc` file:

```
PATH=install_dir/bin:$PATH
export PATH
```

Replace *install_dir* with the synthesis tool set installation directory.

The `.synopsys_dc.setup` file contains the system defaults for the synthesis tools.

Setting the SNPSLMD_LICENSE_FILE Environment Variable

You must install the SCL software and define the `SNPSLMD_LICENSE_FILE` variable before you can verify the synthesis installation.

For information about downloading SCL, installing SCL, or setting the license variable, see *Installing Synopsys Tools* at <http://www.synopsys.com/Support/Licensing/Installation/Pages/default.aspx>

Configuring the Browser for Design Vision Online Help

The online Help system is a browser-based HTML Help system designed for viewing in the Firefox and Mozilla Web browsers. Supported versions include Firefox 2.0, Firefox 1.5, and Mozilla 1.7.

If you prefer to use other browsers, note the following limitations:

- Online Help does not work in Netscape version 6.0. (Netscape 6.x browsers are not supported.)
- Online Help might work in Netscape versions 4.7x, 4.8, and 7.0, or in other browsers such as Internet Explorer or Opera, but it is not tested or supported in these browsers.

When you use online Help from within the GUI, the directory containing the browser executable file must be in the search path specified by your UNIX or Linux `$PATH` variable.

The default browser is Firefox. To use a different browser, change the `gui_online_browser` variable in your `.synopsys_dv_gui.tcl` setup file to the browser of your choice. You can change the browser for the current session by setting the `gui_online_browser` variable from within the GUI. For example, to change your current session browser to Mozilla, execute the following:

```
set gui_online_browser "mozilla"
```

Installing Optional Tools

The synthesis media installation script automatically installs most of the synthesis tools. Considerations for tools that require additional installation setup are described in the following sections:

- [Linking Power Compiler VPOWER to a Verilog-XL Simulator](#)
- [Linking Power Compiler VPOWER to the VCS Simulator](#)
- [Using SoCBIST With DFT Compiler and TetraMAX](#)
- [Using the Model Adaptation System and the Library Quality Assurance System](#)

Linking Power Compiler VPOWER to a Verilog-XL Simulator

VPOWER is the Power Compiler interface to VCS, the Cadence Verilog-XL and NC-Verilog simulators, and the MTI Verilog simulator. VPOWER contains user tasks that allow you to monitor toggle activity during simulation and to output the information in a form readable by Power Compiler. To use VPOWER, link the user tasks to the executable file of your simulator.

The following sections describe the steps for static-linking VPOWER with a version of the Verilog-XL simulator that contains the standard features you normally use at your site and includes the toggle count utilities needed for Power Compiler. For information about linking VPOWER with other simulators, see the *Power Compiler User Guide*.

Note:

You must perform this installation on a machine that has access to your Verilog-XL simulator vendor distribution.

Consult your Verilog system administrator to obtain the following information before beginning the VPOWER installation:

- The directory path to your Verilog .o, .a, and .h files
- The directory location of your central Verilog distribution, for obtaining a current site copy of the veriusers.c file

This installation requires modification of your veriusers.c file. By obtaining a current site copy of the veriusers.c file, you can be sure to include any current site modifications when you modify this file.

To install VPOWER,

1. Change to the Synopsys vpower directory.

2. Modify a copy of your site veriusers.c file.
3. Link the VPOWER user tasks to the simulation executable file.
4. Copy the linked executable file.

The following sections describe these steps.

Changing to the Synopsys Power Directory

All directories listed are relative to the root of the vpower directory: `$$SYNOPSIS/auxx/syn/power/vpower`, where `$$SYNOPSIS` is the path to the synthesis tool set installation directory.

To change to the Synopsys vpower directory,

1. Make sure the `$$SYNOPSIS` environment variable is set.

```
% echo $$SYNOPSIS
```

If it is not set, set it to the correct value.

```
% setenv SYNOPSIS root_directory
```

2. Change to the Synopsys vpower directory.

```
% cd $$SYNOPSIS/auxx/syn/power/vpower
```

Modifying the veriusers.c File

To modify the veriusers.c file to define the new toggle count utilities,

1. Change to the `vx1/vx1.sample` directory, and review the sample veriusers.c file, which shows the edits you will have to make.

```
% cd vx1/vx1.sample
```

2. Copy your current site version of veriusers.c into the sample directory. To copy veriusers.c, you must know the directory location of your central Verilog distribution.

```
% cp site_location_dir_path/veriusers.c .
```

By using a current site copy of veriusers.c, you ensure that any existing customizations are included in the VPOWER installation.

3. As shown in the sample veriusers.c file, make the following changes in your current site copy of veriusers.c:

- Add the following line:

```
# include "tc_extern.h"
```

- Add the following user tasks:

```
{usertask, 0, 0, 0, tc_set, tc_set_sync, "$toggle_set", 1},
{usertask, 0, 0, 0, tc_start, 0, "$toggle_start", 1},
{usertask, 0, 0, 0, tc_stop, 0, "$toggle_stop", 1},
{usertask, 0, 0, 0, tc_reset, 0, "$toggle_reset", 1},
{usertask, 0, 0, 0, tc_compatibility, 0, "$toggle_count", 1},
{usertask, 0, toggle_report_check, 0, toggle_report, 0, "$toggle_report", 0},
{usertask, 0, 0, 0, read_lib_saif, tc_lib_sync, "$read_lib_saif", 1},
{usertask, 0, 0, 0, read_rtl_saif, tc_set_sync, "$read_rtl_saif", 1},
```

- Comment out the following line:

```
char *veriuser_version_str = "";
```

4. Save your modified veriuser.c file.
5. Exit your text editor and remain in the sample directory to link the executable file.

Linking User Tasks to the Simulation Executable File

VPOWER provides two ways to link the user tasks to your simulator executable file: by using the vconfig utility or by using a UNIX makefile. Each method links your simulator to the VPOWER user tasks. Choose the method that you find familiar or comfortable.

Using vconfig to Link the Executable File

The vconfig utility creates a script called cr_vlog. The cr_vlog script links your Verilog-XL simulator's executable file to the VPOWER user tasks. You must define the name of the executable file created by cr_vlog, for example, verilog_toggle.

To use the vconfig method to link your executable file,

1. Use your vconfig utility or an equivalent utility to generate the cr_vlog script or an equivalent script.
2. In the script, set an environment variable pointing to the directory of the generated library archive. For example (if you are using Solaris 7 or later),

```
setenv PPLILIB "../..../lib-sparcOS5/libvpower.a"
```

3. In cr_vlog, look for the line that includes the math libraries:

```
-lm \
```

4. Add a line above this line to include the libvpower.a library. For example,

```
$PPLILIB \
-lm \
```

5. Run `cr_vlog`.

```
% cr_vlog
```

This script links your executable file to the VPOWER user tasks and creates the customized executable file called `verilog_toggle`. For details about linking the programmable language interface (PLI) by using the `vconfig` utility, see the *Power Compiler User Guide*.

Proceed to [“Copying the Linked Executable File.”](#)

Using a Makefile to Link the Executable File

Using the UNIX `make` command, you can use a makefile to link your Verilog-XL executable file to the VPOWER user tasks. The makefile creates a modified executable file called `verilog_toggle`.

Two makefiles exist: `Makefile.sol` and `Makefile.hp`.

To use the makefile method to link your executable file,

1. Using a text editor such as `vi`, edit the appropriate makefile to set variable values for `VERILOG_LIB` and `VERILOG_INC`.

Modify the lines in the makefile to read according to your data. For example, enter

```
VERILOG_LIB = path1  
VERILOG_INC = path2
```

where *path1* is the path to your Verilog distribution `.o` and `.a` files, and *path2* is the path to your Verilog distribution `.h` files.

The `VERILOG_LIB` variable must point to the directory path of the `vlog.o` and `omnitasks.o` files. The `VERILOG_INC` variable must point to the directory path of the `acc_user.h` and `veriusers.h` files.

2. Save the modified makefile and exit your text editor.
3. Use the `make` utility to link the executable file.

```
% make -f Makefile.platform
```

The *platform* extension is `sol` or `hp`.

The `make` command uses the modified makefile to link your executable file, creating a customized executable file called `verilog_toggle`.

Copying the Linked Executable File

After you create your customized executable file, change the permissions so that the file is not writable, and copy it to a directory suitable for group access.

Enter the following commands at the UNIX prompt:

```
% chmod ogu-w verilog_toggle
```

This removes write access to other, group, and user.

```
% cp verilog_toggle site_verilog_bin_location
```

This copies the file to the site_verilog_bin_location directory for group access.

Linking Power Compiler VPOWER to the VCS Simulator

VPOWER is the Power Compiler interface to VCS, the Cadence Verilog-XL and NC-Verilog simulators, and the MTI Verilog simulator. VPOWER contains user tasks that allow you to monitor toggle activity during simulation and to output the information in a form readable by Power Compiler. To use VPOWER, link the user tasks to the executable file of your simulator.

The following sections describe the steps for static-linking VPOWER with to a version of VCS that contains the standard features you normally use at your site and includes the toggle count utilities needed for Power Compiler. For information about linking VPOWER with other simulators, see the *Power Compiler User Guide*.

Note:

The PLI library has been tested with VCS version 3.0 and later versions.

To install VPOWER,

1. Change to the Synopsys vpower directory.
2. Modify a copy of the PLI table file.
3. Compile the simulation executable file.

The following sections describe these steps.

Changing to the Synopsys VPOWER Directory

All directories listed are relative to the root of the VPOWER directory: `$(SYNOPSIS)/auxx/syn/power/vpower`, where `$(SYNOPSIS)` is the path to the synthesis tool set installation directory

1. Make sure the `$(SYNOPSIS)` environment variable is set.

```
% echo $(SYNOPSIS)
```

If it is not set, set it to the correct value.

```
% setenv SYNOPSIS synthesis_root_directory
```

2. Change to the Synopsys VPOWER directory.

```
% cd $(SYNOPSIS)/auxx/syn/power/vpower
```

Modifying the PLI Table File

To modify the PLI table file (`vpower.tab`) to define the new toggle count utilities,

1. Change to the `vcs/vcs.sample` directory, and review the sample `vpower.tab` file, which shows the edits you will have to make.

```
% cd vcs/vcs.sample
```

2. Make the necessary changes to the `vpower.tab` file.

Compiling the Simulation Executable File

VCS is a compiled simulator, so you must compile your designs along with VCS libraries to make a simulation executable file. To add PLI functionality to the simulation executable file, you need to link an extra PLI library when you compile your designs.

For Solaris the appropriate PLI library is

```
../../lib-sparcOS5/libvpower.a
```

You normally get a VCS simulation executable file by entering the following command at the UNIX prompt:

```
% vcs -Mupdate your_verilog_design_files compiler_options
```

To link with the PLI library, enter

```
% vcs -Mupdate -P $(SYNOPSIS)/auxx/syn/power/vpower/vcs/vcs.sample/vpower.tab \  
your_verilog_design_files compiler_options \  
$(SYNOPSIS)/auxx/syn/power/vpower/lib-sparcOS5/libvpower.a
```

This generates an executable file called `simv` that includes PLI functionality.

Note:

You can copy `vpower.tab` and `libvpower.a` into any file locations that are convenient for you.

Using SoCBIST With DFT Compiler and TetraMAX

To insert SoCBIST into your design, you need DFT Compiler, which installs with the synthesis tools. You also need TetraMAX if you want to use the SoCBIST pattern generation functionality. You can install TetraMAX as an overlay on the synthesis tools or as a stand-alone installation. For required SoCBIST environment variables, see [“Specifying the Executable File Location” on page 1-6.](#)

Using the Model Adaptation System and the Library Quality Assurance System

Library Compiler can perform various types of library formatting operations. Each operation converts or merges one or more existing library files in Liberty (.lib) format to create a new library, which is written out in Liberty format. The set of formatting capabilities is called the Model Adaptation System. The Library Quality Assurance System checks libraries in .lib format for errors in consistency and accuracy. It can perform library validation and library correlation.

To run the Model Adaptation System operations and the Library Quality Assurance System utility, you must set the path for the `SYNOPSYS_NCX_ROOT` variable in your `.cshrc` file and point it to the `NCX_Install_Directory` location. If you are using the C shell, add the following line to the `.cshrc` file:

```
setenv SYNOPSYS_NCX_ROOT path_to_NCX_Install_Directory
```

If you are using the Bourne, Bash, or Korn shell, add these lines to the `.profile`, `.bashrc`, or `.kshrc` file:

```
SYNOPSYS_NCX_ROOT = path_to_NCX_Install_Directory  
export SYNOPSYS_NCX_ROOT
```

Verifying the Synthesis Tools Installation

To verify installation,

1. Make sure you are in a directory, such as your home directory, where you have read/write privileges. For example, to change to your home directory, you might execute

```
% cd $HOME
```

2. Invoke the synthesis tools on a licensed machine. For example, invoke Design Compiler, Library Compiler, or Design Vision by entering one of the following commands:

```
% install_dir/platform/syn/bin/dc_shell-topographical
```

```
% install_dir/platform/syn/bin/lc_shell
```

```
% install_dir/platform/syn/bin/design_vision
```

where *install_dir* is the synthesis tool set installation directory and *platform* is the appropriate platform (see [Table 1](#)).

Note:

You can verify other synthesis tools by using the preceding command. Simply replace the executable file name with the name of another synthesis tool.

If you see information about the product version, production date, and copyright, the installation was successful.

Limitations

Milkyway Support

When using Milkyway, Design Compiler does not support the IBM RS/6000 AIX (32) or IBM RS/6000 AIX (64) platforms.

