SYNOPSYS®

SmartModel Library Administrator's Manual

To search the entire manual set, press this toolbar button. For help, refer to intro.pdf.

February 2001

Contents

Figures

Tables

Preface

About This Manual

This manual contains reference information about how to administer the SmartModel Library. It does not contain information about the different kinds of SmartModels or how to use them—that information is presented in the *SmartModel Library User's Manual*.

Related Documents

For general information about SmartModel Library documentation, or to navigate to a different online document, refer to the *Guide to SmartModel Documentation*. For the latest information on supported platforms and simulators, refer to *SmartModel Library Supported Simulators and Platforms*.

For detailed information about specific models in the SmartModel Library, use the Browser tool ($LMC_HOME/bin/sl_browser) to access the online model datasheets.

Manual Overview

This manual contains the following chapters and appendixes:

Preface	Describes the manual and lists the typographical conventions and symbols used in it; explains how to get technical assistance.
Chapter 1: Admin Tool	How to use the Admin tool to install and purge models, and otherwise manage the SmartModel Library installation.
Chapter 2: SmartModel License Agreement	The SmartModel Library license agreement.

Chapter 3: GNU General Public License	The GNU General Public License.
Chapter 4: Third-Party Tools	Information about third-party tools that work in conjunction with SmartModels.
Chapter 5: FLEXlm Command Reference	The FLEXlm licensing software command man pages.
Appendix A: Glossary	Definitions for terms that have special meaning in the context of this manual.

Typographical and Symbol Conventions

- **Default UNIX prompt**

  Represented by a percent sign (%).

- **User input** (text entered by the user)

  Shown in **bold** type, as in the following command line example:

  ```
  % cd $LMC_HOME/hdl
  ```

- **System-generated text** (prompts, messages, files, reports)

  Shown as in the following system message:

  ```
  No Mismatches: 66 Vectors processed: 66 Possible
  ```

- **Variables** for which you supply a specific value

  Shown in italic type, as in the following command line example:

  ```
  % setenv LMC_HOME prod_dir
  ```

  In this example, you substitute a specific name for *prod_dir* when you enter the command.

- Command syntax

  Choice among alternatives is shown with a vertical bar (|) as in the following syntax example:

  ```
  -effort_level  low | medium | high
  ```

  In this example, you must choose one of the three possibilities: low, medium, or high.

Optional parameters are enclosed in square brackets ([ ]) as in the following syntax example:

pin1 [*pin2 ... pinN*]

In this example, you must enter at least one pin name (*pin1*), but others are optional ([*pin2 ... pinN*]).

Getting Help

If you have a question while using Synopsys products, use the following resources:

1. Start with the available product documentation installed on your network or located at the root level of your Synopsys CD-ROM. Every documentation set contains overview information in the intro.pdf file.

   Additional Synopsys documentation is available at this URL:

   http://www.synopsys.com/products/lm/doc

   Datasheets for models are available using the Model Directory:

   http://www.synopsys.com/products/lm/modelDir.html

2. Visit the online Support Center at this URL:

   http://www.synopsys.com/support/lm/support.html

   This site gives you access to the following resources:

   - SOLV-IT!, the Synopsys automated problem resolution system
   - product-specific FAQs (frequently asked questions)
   - lists of supported simulators and platforms
   - the ability to open a support help call
   - the ability to submit a delivery request for some product lines

3. If you still have questions, you can call the Support Center:

   North American customers:
   Call the Synopsys Eagle*i* and Logic Modeling Products Support Center hotline at 1-800-445-1888 (or 1-503-748-6920) from 6:30 AM to 5 PM Pacific Time, Monday through Friday.

   International customers:
   Call your local sales office.

The Synopsys Website

General information about Synopsys and its products is available at this URL:

http://www.synopsys.com

Synopsys Common Licensing (SCL) Document Set

Synopsys common licensing (SCL) software is delivered on a CD that is separate from the tools that use this software to authorize their use. The SCL documentation set includes the following publications, which are located in (root)/docs/scl on the SCL CD and also available on the Synopsys FTP server (ftp://ftp.synopsys.com):

- *Licensing QuickStart*—(142K PDF file)
  This booklet provides instructions for obtaining an electronic copy of your license key file and for installing and configuring SCL on UNIX and Windows NT.
- *Licensing Installation and Administration Guide*—(2.08M PDF file)
  This guide provides information about installation and configuration, key concepts, examples of license key files, migration to SCL, maintenance, and troubleshooting.

You can find general SCL information on the Web at:

http://www.synopsys.com/keys

Comments?

To report errors or make suggestions, please send e-mail to:

doc@synopsys.com

To report an error that occurs on a specific page, select the entire page (including headers and footers), and copy to the buffer. Then paste the buffer to the body of your e-mail message. This will provide us with information to identify the source of the problem.

1 Admin Tool

Introduction

The Admin tool (sl_admin) provides a graphical user interface to the SmartModel Library. This tool is of use primarily to system administrators tasked with installing and managing the SmartModel Library. Use the Admin tool if you want to:

- Install models into a new or existing SmartModel Library tree; for detailed information on installing and configuring the SmartModel Library product, refer to the *SmartModel Library Installation Guide*.
- Purge older versions of models from an installed SmartModel Library.
- Check library integrity.

The SmartModel Library accommodates multiple versions of the same model. You do not need to create a new directory when you receive new models to install. Instead, install the new models in the same tree you have been using. Individual members of your design team can create custom configurations to specify a particular version of a model, without affecting the designs of others that use the library.

Using the Admin Tool

Set your environment variables and search paths according to the setup instructions in "Step 2: Configure each User's Environment" in the *SmartModel Library Installation Guide*. You are now ready to use the Admin tool. At the command line, enter the following command to start the Admin tool:

```
% $LMC_HOME/bin/sl_admin
```

Model Versions

Models are the basic units in the library. Each versioned model supports multiple timing versions. Each timing version, in turn, can support multiple devices, or physical integrated circuits that you can order from a manufacturer. When you install a particular model version, you get all of the timing versions for that model. Similarly, when you purge a particular version of a model from the library, you purge all of its timing versions as well. For information about all of the timing versions and components supported by any model in the library, review the model's datasheet. For more information about model versioning, refer to the *SmartModel Library User's Manual*.

Configuration Files

A configuration is a list of SmartModel Library models, each with a specific version defined. Normally, a configuration is stored in one or more files, called configuration files.

A configuration file is also called an ".lmc" file or "LMC" file; "LMC" stands for "List of Model Configurations." There are two kinds of configuration (LMC) files:

- Default configuration (LMC) files
- Custom configuration (LMC) files

All configuration (LMC) files must have the extension .lmc.

The entire process of selecting a model version for a particular simulation is carried out using configuration files and the LMC_HOME and LMC_CONFIG environment variables.

Configuration File Example

Configuration (LMC) files contain a list of model names and user-versioned tool names, with a version number specified for each model and tool. Following is an example configuration file:

```
%PLT hp700
# Models added by Sl_Admin: Fri Feb 23 15:56:24 1996
%EXE swiftcheck 01009
%EXE mi_trans 04059
%EXE ptm_make 01006
%MOD atv2500 01000
%MOD bt458 01000
%MOD c5c_c8c_2 01000
%MOD dm74s188 01000
%MOD ecl01 01000
```

```
%MOD gal18v10 01000
%MOD hm658128 01000
%MOD ifc161 01000
...
%MOD ttl0 01000 7400 74LS00
...
%MOD windows 01000
%MOD z8536 01000
```

Configuration File Syntax

There are three commands that can appear in an LMC file:

%PLT *platform_name*

This command specifies the platform. Examples of allowed values are hp700, pcnt, x86_linux, solaris, and ibmrs. The first line in the example:

```
%PLT hp700
```

indicates the hp700 platform.

If PLT is absent, no platform checking is done. Therefore, if you want a single LMC file to be shared among several platforms, omit the PLT command.

%EXE *tool_name version*

This command specifies the versioned name of the executable for a tool that can be run on the user's configuration. Examples of such tools include:

- swiftcheck—a tool that checks the integrity of your SmartModel installation.
- mi_trans—a tool that translates memory image files for memory models.
- ptm_make—a tool that creates simulator-specific PortMap files, which map a model's port names to the physical pins of the device.

In the example, the lines:

```
%EXE swiftcheck 01009
%EXE mi_trans 04059
%EXE ptm_make 01006
```

indicate that, when the tools are called, the versions to be used are 01009 for swiftcheck, 04059 for mi_trans, and 01006 for ptm_make.

%MOD *model_name model_version* [*alias*[*alias*]...[]

This command specifies the model name, model version, and optionally any alias names that might apply to the model. In the example, the line:

```
%MOD ttl00 01000 7400 74LS00
```

indicates model ttl00, version 01000, with aliases that include 7400 and 74LS00.

The Default Configuration File

The default configuration (LMC) file that comes with the SmartModel Library lists all of the installed SmartModel Library models, user-versioned tools, and their most recently-installed versions. This file is platform-specific, and is named *platform*.lmc. For example, the default configuration file could be named hp700.lmc, ibmrs.lmc, pcnt.lmc, x86_linux, or solaris.lmc, depending on your installation platform. Normally, when a model version is installed in the library, the $LMC_HOME/data/*platform*.lmc file is updated with the most recently added model version. For example, if Version 01002 is installed after Version 01004, then Version 01002 is the default version used even though Version 01004 has a higher version number.

Before using the Admin tool, you must specify the default configuration (LMC) file by setting your local LMC_HOME environment variable to the install directory. The model versions specified in the default configuration (LMC) file will be used by the simulator unless you have defined other model versions in one or more custom configuration (LMC) files.

Custom Configuration Files

If users at your site want to use model versions that are different from those listed in the default configuration file, they can create one or more custom configuration (LMC) files by copying the default configuration file and editing it. After creating the custom configuration (LMC) files, these users must set the local LMC_CONFIG environment variable with the paths to the files.

Note

For NT, the list of paths specified for LMC_CONFIG must be separated by semicolons, not colons as in UNIX.

There are several reasons why users at your site might want to create custom configuration files:

- They want to freeze a design, so that it always references the same model versions.
- They need a specialized model version that no one else at your site should use.
- They want to check new models before releasing them to others.
- They want to archive a design, along with the model versions used.
- They need an updated model version to use a new or revised function, but other design teams do not want to disturb their designs by using the updated model.

The model versions specified in custom configuration files are used by the simulator, overriding the versions of those same models that are specified in the default configuration file. To determine what versions of a specific model are installed in your library, and thus available to the simulator as specified in a configuration file, refer to "Determining What Versions You Have" on page 28.

Model and Tool Version Numbers

Model versions have the five-digit format xxyyy (for example, 01001), where xx designates a major revision and yyy designates a minor revision. This five-digit version number appears on all model datasheets.

SmartModel Library tools are also versioned and use the same numbering scheme as models.

How Model Versions are Selected

Multiple versions of the same model can exist in the same installed library. You can select which model versions are used in a particular design simulation using a configuration (LMC) file.

By default, the model version selected is the most recently-added version, as determined by the default configuration (LMC) file, which comes with the SmartModel Library. To override the default model versions for individual designs, you can create and use any number of custom configuration (LMC) files, which contain user-specified versions of individual installed library models. The software locates the default and custom configuration files through the LMC_HOME and LMC_CONFIG environment variables. For instructions on setting these variables, refer to the *SmartModel Library User's Manual*.

When you invoke the SmartModel Library Browser, the selection pane displays a list of timing versions corresponding to models that will be used by the simulator for the current design according to the current environment settings. Although multiple model versions exist, you see only one version at a time. To find out what model versions are available at your site, refer to "Determining What Models You Have" on page 28.

How Tool Versions are Selected

For some SmartModel Library tools, called model-versioned tools, the version of the tool to use is determined by the model. Users cannot select versions of model-versioned tools. Model-versioned tools include compile_timing, ccn_report, smartbrowse, and smartccn.

For other SmartModel Library tools, called user-versioned tools, the version of the tool is selected through default and custom configuration (LMC) files, in the same way previously described for models. User-versioned tools include ptm_make, vsb, mi_trans, and swiftcheck.

Environment Settings and the Simulator

To determine each model version to be used in simulation, the software uses the following environment variables:

- LMC_CONFIG, which contains path names to any custom configuration files
- LMC_HOME, which points to the default configuration file

If there are no custom configuration files, then the simulator uses the model versions specified in the default configuration file.

For each model called by the design, the software searches first in the files referenced by the LMC_CONFIG variable, in the order in which they are listed. If a model is not found, the software next searches for it in the default configuration file ($LMC_HOME/data/*platform*.lmc). The software uses the version of the first model it encounters.

For example, if

```
LMC_HOME=/d/lsl/latest
```

and

```
LMC_CONFIG=/user/me/my_platform.lmc:/user/joe/joes_platform.lmc
```

for each model, the model version is obtained as follows:

1. The software first searches for the model in /user/me/my_platform.lmc. If the model is found, that model version is used and the software stops searching for that model even though it might be present in subsequent files.
2. If the software did not find the model in the first file, it searches /user/joe/joes_platform.lmc. If the model is found, that model version is used and the software stops searching for that model.
3. If the first model encountered is a version that is not in the library, the model will not be used in simulation, even though there might be other versions of that model in the library. This means that you must be sure to only specify model versions in custom configuration files that are actually installed at your site.
4. If the software did not find the model in the last custom configuration file by the LMC_CONFIG variable, it searches the default configuration file (*platform*.lmc) specified by the LMC_HOME variable (/d/lsl/latest/data). If the model is found, that model version is used and the search ends.
5. If the model is not found in any file, the SWIFT interface indicates to the simulator that the model is invalid. Depending on the simulator, an error message may or may not be generated. You can investigate error messages about bad integration or a possible missing model by setting the LMC_COMMAND environment variable to "verbose on" and resimulating. If the messages produced are not sufficient to diagnose the problem, the next step is to run swiftcheck to verify the integrity of your SmartModel Library installation. For information about the swiftcheck tool, refer to the *SmartModel Library User's Manual*.

   Other reasons for a model not being found include a possible typing error in a configuration file or a reference to a model version not installed in the library at your site.

Installing Models

To install a list of models, follow these steps:

1. Look in the Admin tool status area where the path to the library (LMC_HOME) is displayed. If this is the path to the library you want to install or update, skip to Step 4; otherwise proceed to Step 2.
2. From the File menu, choose Open Library. The Set Library Directory dialog box opens.
3. Enter the directory of the library (LMC_HOME) you want to install or update and click OK. The Set Library Directory dialog box closes and the status area displays the directory you just entered.

4. From the Actions menu or the Tool Bar, choose Install. The Install From dialog box opens.
5. If you are installing from a CD-ROM, enter the CD-ROM directory in the Install From text field and click on Open. If you are installing from an FTP shipment, complete the FTP transfer and unzip the image as explained in the e-mail that you received with your order. The unzip process produces a CD image that you install by using the Browse button on the Install From dialog box to navigate to the unzipped CD image.
6. Select the file you want and click on Open. The Select Models to Install dialog box appears.
7. Select the models you want to install. For more information, refer to "Selecting Models to Install or Purge" on page 22.
8. Click on Continue. The Select Models to Install dialog box closes and the Select Platforms dialog box appears.
9. In the Platforms checklist, select all platforms where you want the models installed. (By default, the platform that is selected is the one on which you are running the Admin tool.)
10. The Configuration Files text box shows the path name to a configuration file the Admin tool found by appending /data/*platform*.lmc to your install directory. If this is the configuration file you want to update, proceed to the next step. If not, type in the correct file name. (You can use the Browse button to open the Locate Media dialog box to assist you in finding the correct file.)
11. In the EDAV Packages field, select one or more simulators. You must select at least one EDAV package. If you do not use any of the named simulators, click on the "other" button.
12. Click on the Install button. The Select Platforms dialog box closes. The transcript window displays status messages as it reads the media, checks the user selections, and loads the models. The transcript window also displays each model name, version number, and platform as it is installed.

Purging Models

To purge a model from the library means to remove one or more of its versions, To remove model versions, use the Purge function. Do not attempt to remove versions by manually deleting files. Each model version references multiple files, some of which are shared with other models and model versions. By manually deleting files, you might remove shared files that are needed by model versions that remain in the library. The Purge function is designed to remove only those files that are unique to the model version currently being purged.

You do not need to purge old model versions in order to install new model versions. Instead, use the Browser and Admin tools to locate the model versions that you want to use.

You can have lots of versions of the same model installed at your site. However, there are some good reasons to purge old model versions from the library, as follows:

- You want to recover needed disk space.
- You want to force users to upgrade to newer model versions.

Before purging a list of models, if possible, consult with all designers that are using the library at your site. Some existing designs might still need the older model versions.

To purge a list of models, follow these steps:

1. Look in the status area where the path to the library is displayed. If this is the path to the library you want to update, skip to Step 4; otherwise proceed to Step 2.
2. From the File menu, choose Open Library. The Set Library Directory dialog box opens.
3. Enter the directory of the library you want to update, and click OK. The Set Library Directory dialog box closes. The status area now displays the directory you just entered.
4. From the Actions menu or the Tool Bar, choose Purge.

Caution

At this point a message appears to let you know that purging models from your $LMC_HOME may interfere with the work of other users at your site. For example, if someone is using the LMC_PATH and LMC_CONFIG environment variables to point to models not installed in $LMC_HOME or versions of models not listed in the default LMC file, some files needed by those model versions could be deleted. Before purging models it is best to check with all SmartModel users at your site to see if they still need specific model versions.

5. Next, the Select Platform for Purge dialog box opens. In the Platforms list, select the platforms from which you want to purge models.
6. The Configuration Files text box shows the path names to one or more configuration files the Admin tool found by appending /data/*platform*.lmc to your install directory. If these are the configuration files you want to purge, proceed to the next step. If not, type in the correct file names. (You can use the Browse button to open the Locate File for Purge Update dialog box to assist you in finding the correct file.)
7. In the "Number of versions to keep" text field, enter the number of installed model versions that you want to protect from the purge operation. The default for this is 3 and legal values range from 2 through 99. For example, if you leave this value at 3, the purge operation deletes all but the 3 most recent model versions for each model that you select.
8. Click on the Continue button. The Select Models to Purge dialog box opens.
9. Select the desired models to purge.
10. Click on Continue. The Select Models to Purge dialog box closes. The transcript window displays status messages as it reads the media, checks the user selections, and purges the models. The transcript window displays each model name, version number, and platform as it is purged.

Selecting Models to Install or Purge

When the Select Models to Install dialog box opens, the Items Available list either displays timing-versions (UNIX platforms) or model names (NT platforms). Select a model or timing-version name to add to the Models to Install list. The Admin tool adds the corresponding model or timing-version name. If you are on UNIX and want to display the Items Available list by model name, click on the Model Name button.

Attention

The Select Models to Purge dialog box always displays model names, not timing-version names.

For simplicity, the procedure that follows assumes that you are selecting from the Items Available list box by model name.

1. From the Items Available list, select the items you want to install or purge.

To select all items for installing or purging, click on the Add All button. All model names from the Items Available list box now appear also in the Models to Install/Purge list box.

To select individual items for installing or purging, click on the names one at a time. As you click on each name, the model name appears in the Models to Install/Purge list.

Note

Clicking on the model's icon displays the model's timing-versions and selects the model, but does not add it to the Models to Install/Purge list. Use the Add button to add selected models to the Models to Install/Purge list.

To select a set of consecutive model names, click on the top name, then hold down the shift key while clicking on the bottom name. The top and bottom names, plus all names in between, are selected and added to the Models to Install/Purge list. On NT platforms, press Ctrl and click to select multiple entries in the model list.

To remove all models from the Models to Install/Purge list, click on the Remove All button. The Models to Install/Purge list is cleared.

To remove some models from the Models to Install/Purge list, select them as previously described, then click on the Remove button. The selected models are removed from the list.

2. When you are satisfied with the Models to Install/Purge list, click on the Continue button (for Install) or the Purge button (for Purge).

If you are purging, the Select Models to Purge dialog box closes, and you are finished. The transcript window displays status messages, displays each model as it is purged, and finally displays this message:

```
Updating Library file
Purge complete
```

If you are installing, the Select Platforms dialog box opens.

3. Select a platform and a simulator.

4. Click on the Install or Purge button.

The Select Platforms dialog box closes. The transcript window displays status messages as it initializes and reads the media and checks user selections. Then it displays each model as it is loaded or purged.

If you purge some models and then discover that you got rid of some model versions that you still need, you can reinstall those model versions. The purge function automatically updates your *platform*.lmc file with a list of models purged and the purge date. It also comments out the model names by replacing %MOD with #MOD. For example, following is a typical entry in a *platform*.lmc file after a purge:

```
%PLT hp700

# Models removed by Sl_Admin: Thu May 23 14:48:55 1996
#      purged: i27960c2, 01001
#      purged: i27960k1, 01001
#      purged: i80386sx_hv, 01001
#      purged: pal20ra10, 01001
#      purged: seeq_36c16, 01001
#      purged: ti16244, 01001

# Models added by Sl_Admin: Thu May 23 14:44:38 1996

%MOD smartlib 01000
%EXE mi_trans 04059
%EXE ptm_make 01006
#MOD i27960c2 01001
#MOD i27960k1 01001
#MOD i80386sx_hv 01001
#MOD pal20ra10 01001
#MOD seeq_36c16 01001
#MOD ti16244 01001
```

With this record of models that have been purged, you can easily select the appropriate ones for reinstallation.

Saving a Copy of the Installed/Purged Model List

If you want to save a copy of the Models to Install/Purge list, follow these steps:

1. In the Select Models dialog box, click on the Write List button. The Write List dialog box opens.
2. Specify the name of the output file for the list of models.
3. Click OK. The Models to Install/Purge list is written to the specified file.

Detecting and Fixing a Corrupted Library

If you are having problems with more than one model in the library, the library database could be corrupted. To detect and fix a corrupted library, follow these steps:

1. From the Actions menu or the Tool Bar, choose Consistency Check/Report. The Select Platform for Check dialog box opens.
2. Select the appropriate platform by clicking on its radio button. Enter the appropriate configuration file in the Configuration File text field or click on Browse to open the Locate File to Check dialog box—this will help you locate the configuration file.
3. Click on Continue. The Library Report dialog box opens.
4. Choose Run consistency check on entire library by clicking on its radio button.
5. Click on OK. The Library Report dialog box closes. The Admin tool displays the Report of Consistency Check in the window.

   For more information about consistency checks, refer to "Report 3. Consistency Check" on page 26.

Generating a Library Report or Executing a Command

To generate a library report or execute a command, follow these steps:

1. From the Actions menu, choose Consistency Check/Report. The Select Platform for Check dialog box opens.
2. Open the Library Report dialog box by clicking on the Continue button.
3. Select one of the library report options by clicking on its radio button. By default, the first option is already selected.
4. Click on OK. The dialog box closes and the library report or consistency check result appears in the transcript window. You can save the library report by using the Save Transcript dialog box.

   For more information about library reports, refer to "Library Report Dialog Box" on page 44.

Repairing Errors from a Library Report

Of the seven possible library report options, only the three described below report errors.

Report 1. Configuration (LMC) File Errors

When you select this report, the Admin tool lists models found in the configuration file that you specified to use for the check, but not found in the specified library. The Admin tool reports these models as errors.

To repair errors in Report 1, follow these steps:

1. For each model, find out whether it is being used by anyone at your site. If not, then the model's absence from the library is not a problem. You can missing models from the configuration file to prevent the Admin tool from reporting them as errors in the future
2. For each model used by someone at your site, use the Install command to install the model or model version in the library.

Report 2. Models Found in Library but not in Configuration (LMC) Files

When you select this report, the Admin tool lists models that were found in the specified library but not in the configuration file that you specified to use for the check. The Admin tool reports these models as errors.

To repair errors in Report 2, follow these steps:

1. For each model, find out whether it is being used by anyone at your site. If not, then the model's absence from the configuration file is not a problem. You can models to the configuration file to prevent the Admin tool from reporting them as errors in the future.
2. For each model used by someone at your site, check the configuration file for an incorrectly entered model name.
3. Edit the configuration file to correct invalid model names.

Report 3. Consistency Check

When you select this report, the Admin tool performs a consistency check, similar to a checksum, and reports any errors that it finds.

To repair errors in Report 3, follow these steps:

1. If there are errors for just one or two models, use the Install command to reinstall those models.
2. If there are errors for more than two models, use the Install command to reinstall all models.

3. If reinstalling the models does not correct the errors, contact Technical Support. For contact information, refer to "Getting Help" on page 11.

Regenerating the Library Files

Normally, you do not need to regenerate the library files because the data is updated automatically when you install or purge models from the library. However, you may need to regenerate the library files if you suspect that:

- The library cache is corrupted (for example, if you are not able to access all of the models you believe should be available)
- You are not accessing the correct version of SWIFT

Running this function will not harm your library installation. However, before using this command, you might want to first run the Consistency Check and reload any models for which the Consistency Check reports errors. For more information, refer to "Repairing Errors from a Library Report" on page 26.

Selecting the Regenerate command from the Actions menu opens the Regenerate Library Files dialog box. Executing this command produces the following effects:

- Regenerates the default configuration (LMC) file—All values are set equal to the latest version of each model found. Any update information about models previously added or purged is lost.
- Rebuilds the library cache to provide faster access to the models
- Resets all library-versioned links to point to the latest version of SWIFT

You must run the Regenerate command once for each platform, as follows:

1. To execute the command, click Continue. A Yes/No query box opens, informing you that the regenerate function may take several minutes to run, and asking whether you want to continue.
2. To continue, click on Yes. As the Regenerate function executes, it displays these messages in the transcript:

```
Generating new Library Cache...
SmartModel Note
       Copyright (c) 1984-96 Synopsys Inc. ALL RIGHTS RESERVED
 Info: Generating: a1010_100
...
(lists the models)
...
Info: Generating: z8530
Generate complete
Generating new default Configuration File...done
Writing new Library Cache...done
```

```
Regenerate complete
Building new Library Versioned links...done
Rebuild links complete
```

Determining What Models You Have

To find out what models you have installed, follow these steps:

1. From the Actions menu, choose Consistency Check/Report. The Select Platform for Check dialog box opens.
2. Select the appropriate platform, and make sure that the correct configuration file is in the Configuration Files text field.
3. Click on Continue. The Library Report dialog box opens.
4. Select Report on entire Library Content by clicking on its radio button.
5. Click on OK. The Library Report dialog box closes. The transcript window displays the report of all versions of all models in the library.

Determining What Versions You Have

To find out what versions you have installed for any model, follow these steps:

1. From the Actions menu, choose Consistency Check/Report. The Select Platform for Check dialog box opens.
2. Select the platform on which the model is installed, and make sure that the correct configuration file is in the Configuration Files text field.
3. Click on Continue. The Library Report dialog box opens.
4. Select Report on entire Library Content by clicking on its radio button.
5. Click on OK. The Library Report dialog box closes and the transcript window displays all model versions in the library.
6. Locate the model you are interested in and note its version number.
7. Repeat the entire procedure for each platform on which you have that model installed. Note that models are updated only on the requested platform, so it is possible to have different model versions installed on different platforms.

Determining if a Model is Out of Date

To determine if a model is out of date, find out the MDL version number for the model installed in the library at your site. Then check the latest datasheet for the model using the Model Directory on the Web at:

http://www.synopsys.com/products/lm/modelDir.html

Enter the model name and start the search. From the resulting list, select the required timing version. When the product information appears, click on the product code to show the datasheet. Each SmartModel datasheet contains a field in the banner section on the first page that shows the MDL version number. That version number reflects the most recent version of the model that is available.

If the latest model version is different than the one you have, read the datasheet's history section to find out what kind of change was made when the new model version was created.

If the change was functional or a bug fix, contact Synopsys to request the newer version for all platforms on which you have that model installed. If the change was purely administrative and did not affect model functionality, you may not need the newer version.

Saving the Transcript from an Admin Tool Session

As you use the Admin tool, the transcript window displays messages and records models you have installed or purged. You can save the contents of the window as a log of your Admin tool session, using the Save Transcript function.

To save a transcript, follow these steps:

- From the File menu, choose Save Transcript. The Save Transcript dialog box opens.
- Specify the file name you want to use to save the transcript and click on the OK button.

Printing the Transcript on NT

To print a transcript on an NT platform, follow these steps:

- From the File menu, select Print Setup. The Print Setup dialog box opens.
- Specify the printer for your output file.
- From the File menu or tool bar, select Print and click the OK button.

Admin Tool Graphical User Interface

The SmartModel Library Administration tool has standard graphical user interface features. Following are brief descriptions of the windows, menus, icons, and dialog boxes you can use to work with the library.

Admin Tool Window

The Admin tool window (see Figures 1 and 2) is the command center that you use to control the tool.

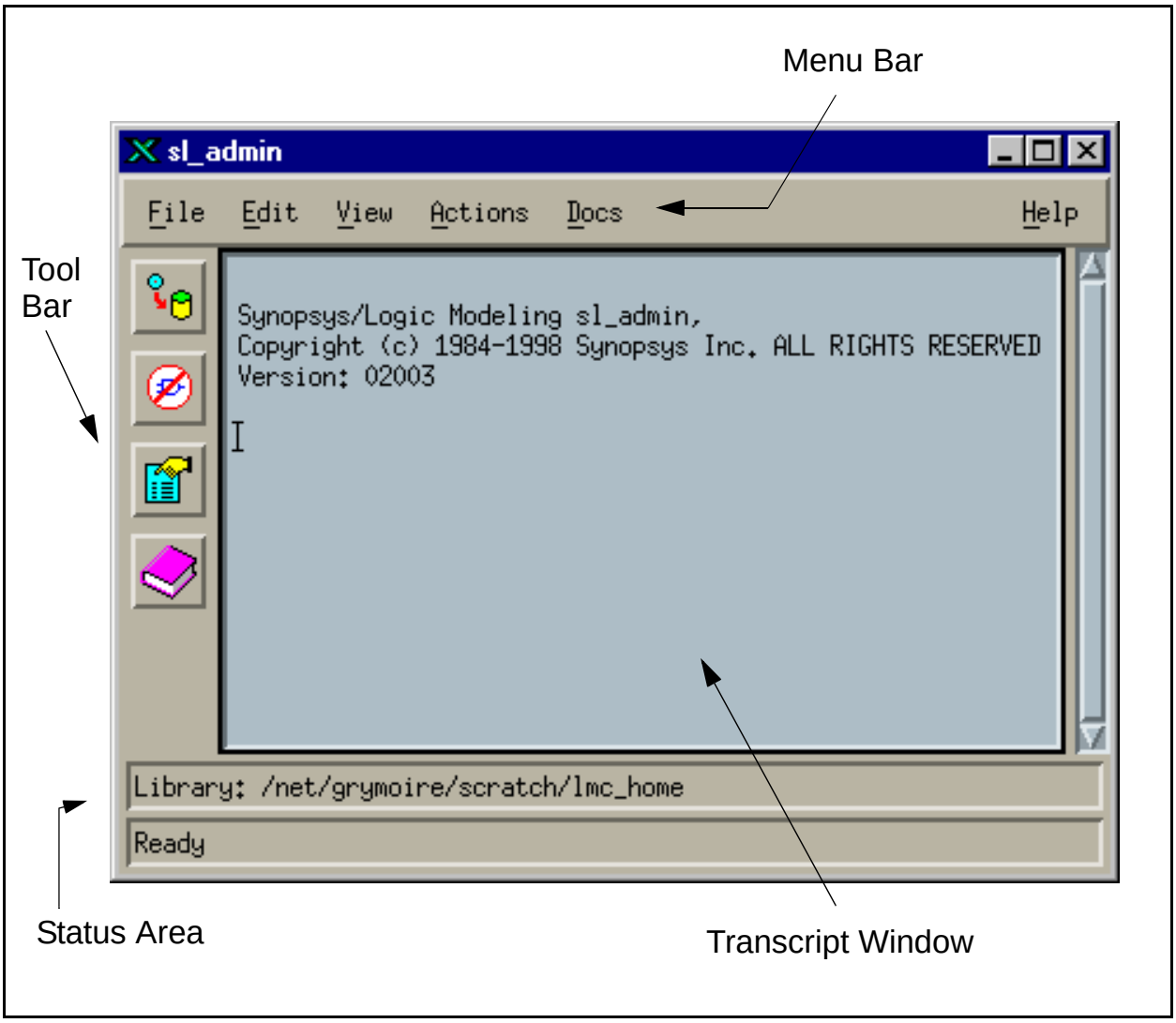

Figure 1: UNIX Admin Tool Window

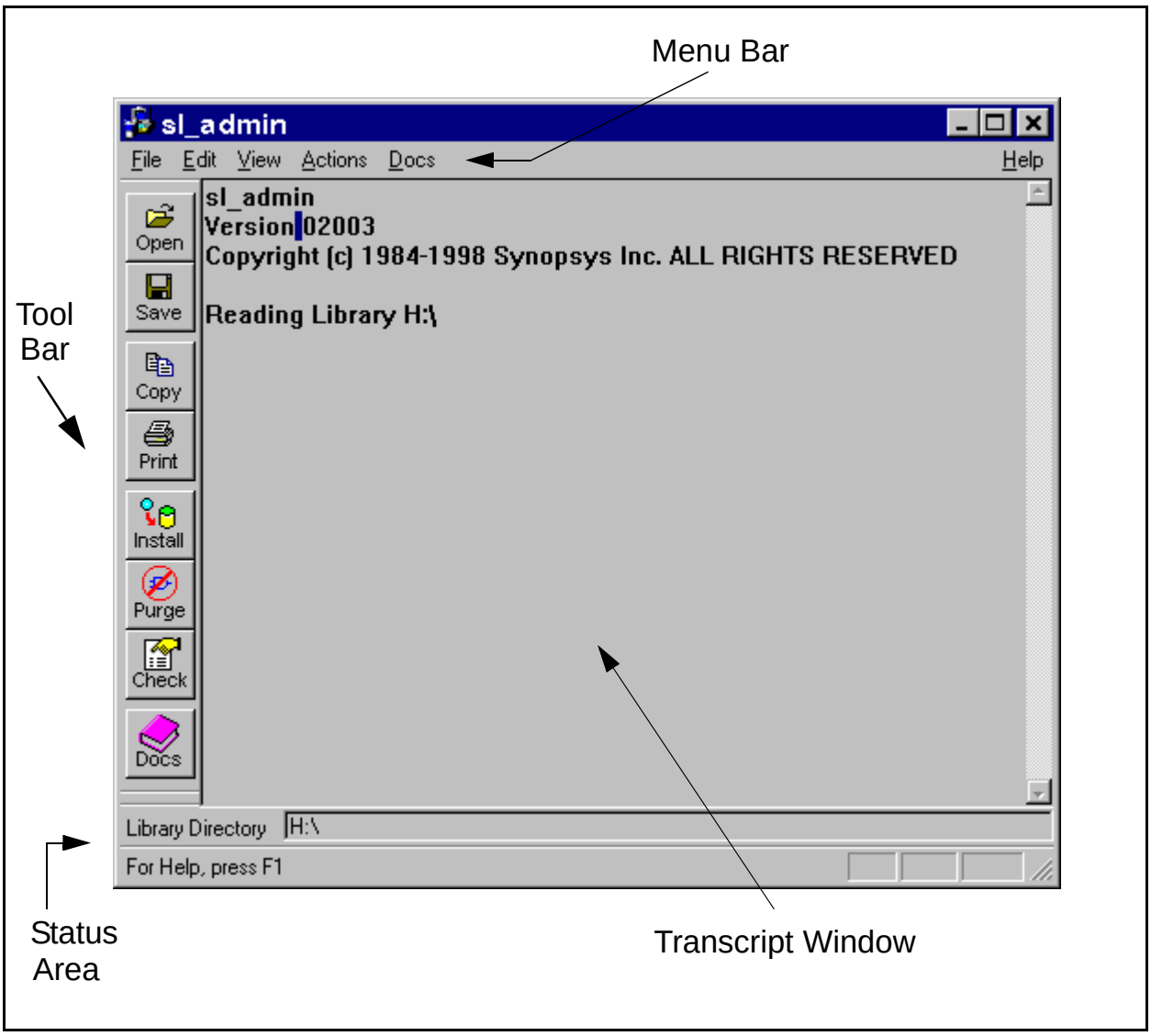

Figure 2: NT Admin Tool Window

The Admin tool window elements are as follows:

- **Menu Bar**—At the top of the Admin tool window, the menu bar contains the File, Edit, View, Actions, Docs, and Help pull-down menus. You can use these menus to initiate all Admin tool functions.
- **Tool Bar**—Arranged vertically at the left side of the Admin tool window, the tool bar buttons offer another way to access functions available through the pull-down menus.

- **Transcript Window**—The transcript window displays lists of models as they are installed or purged. The transcript window also displays library reports and status messages.
- **Status Area**—The status area shows the path name to the SmartModel Library being operated on, and gives various status messages as appropriate.

Menu Bar

The menu bar (see Figure 3) features several pull-down menus that you can use to perform the tasks described below.

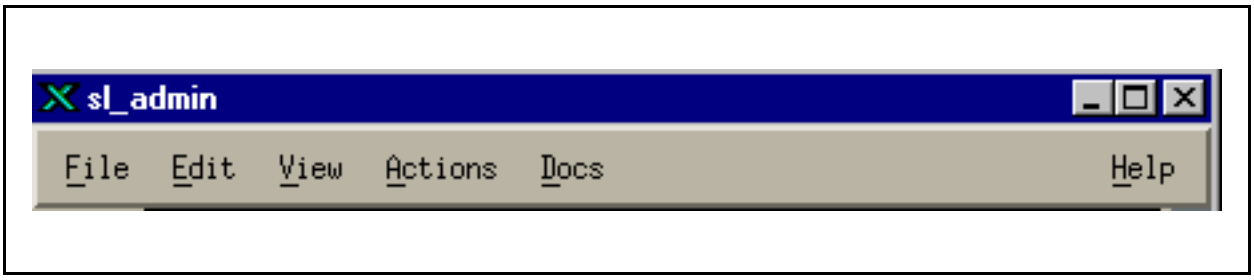

Figure 3: Admin Tool Menu Bar

File Menu

- Open Library—Opens the Set Library Directory dialog box, where you can specify the path to the library (LMC_HOME) you want to open.
- Save Transcript—Opens the Save Transcript dialog box, where you can specify the name of the file where you want to save the transcript.
- User Options—Opens the User Options dialog box, where you can specify the message level verbosity for the tool transcript window. Options include standard, verbose, and debug message levels.
- Print—Opens the Print dialog box, where you can specify a printer for your print job. (This menu item is only available on NT platforms.)
- Print Preview—Opens the Print Preview window, where you can view the print output before actually routing the job to the printer. (This menu item is only available on NT platforms.)
- Print Setup—Opens the Print Setup dialog box, where you can select a default or user-defined printer. (This menu item is only available on NT platforms.)
- Exit—Ends the active Admin tool session.

Edit Menu

- Copy—Copies the lines selected in the transcript window to the clipboard.
- Select All—Selects all lines in the transcript window.

View Menu

- Display by Timing-Version Name—Displays an alphabetical model list based on timing-version names.
- Display by Model Name—Displays an alphabetical model list based on model names.
- Show Model Names—Displays the model name associated with each timing-version name. Note that Display by Timing-Version Name must be set also.
- Expand Model—Expands the model by displaying all of its timing-versions in the form of a hierarchical tree. You get the same result by clicking on the model icon.
- Expand All—Expands all of the models by displaying the timing-versions for all models in the form of hierarchical trees.
- Collapse Model—Hides the timing-versions for the selected model. You can get the same result by clicking on the icon of the expanded model.
- Collapse All—Hides the timing-versions for all models.

The Expand/Collapse menu items are only available in the UNIX version of the Admin tool. For NT, the same functions are provided with standard Windows widgets such as the little box with a plus sign inside.

The NT version of the Admin tool also contains menu items for Toolbar, Library Directory, and Status Bar. You can select or de-select these menu items to change the appearance of your Admin tool window.

Actions Menu

- Install—Opens the Install From dialog box, where you can select CD-ROM or FTP shipment as the source for installing models. (Same as the Install tool bar button.)
- Purge—Opens the Select Platform for Purge dialog box, where you can select model purge options. (Same as the Purge tool bar button.)

- Consistency Check/Report—Opens the Select Platform for Check dialog box, where you can select a platform and a configuration file that you want to generate a library report for. (Same as the Consistency Check/Report tool bar button.)
- Regenerate—Opens the Regenerate Library Files dialog box, where you can regenerate the default configuration file, the library cache, and the library versioned links.

Docs Menu

- Provides links to the SmartModel Library online documentation. When you click on the manual you want, the Admin tool fires up the Acrobat Reader third-party software and displays the online PDF manual on your screen. Click on *Guide to SmartModel Library Documentation* for descriptions of all the manuals in the set.

Help Menu

- This menu contains an "About" entry that displays copyright and version information.

Tool Bar

You can invoke the major Admin tool functions from the tool bar. The tool bar buttons provide another way to access functions available from the Actions menu. To display the function of each tool bar button, place the pointer on the button. The following table describes the different tool bar buttons and what you can do with them.

Table 1: UNIX Tool Bar Button Descriptions

Button	Use To ...
	**Install Button**— Opens the Install From dialog box, where you can select a CD-ROM or unzipped CD-ROM image from an FTP transfer to use as the source for installing models. (Same as the Install command on the Actions menu.)
	**Purge Button**— Opens the Select Platform for Purge dialog box, where you can select the platforms from which you want to purge models. (Same as the Purge command on the Actions menu.)

Table 1: UNIX Tool Bar Button Descriptions (Continued)

Button	Use To ...
	**Consistency Check/Report Button**— Opens the Select Platform for Check dialog box, where you can select a platform and a configuration file that you want to generate a library report for. (Same as the Consistency/Check Report command on the Actions menu.)
	**Docs Button**— Opens the *SmartModel Library Administrator's Manual* (this manual) in PDF format using the Acrobat Reader.

Table 2: NT Tool Bar Button Descriptions

Button	Use To ...
	**Open Button**— Opens the Set Library Directory dialog box (same as Open Library from the File pull-down menu).
	**Save Button**— Saves the transcript to a user-designated file (same as Save Transcript from the File pull-down menu).
	**Copy Button**— Copies the selected text to the Windows clipboard.

Table 2: NT Tool Bar Button Descriptions (Continued)

Button	Use To ...
	**Print Button**— Opens the Print dialog box (same as Print from the File pull-down menu).
	**Install Button**— Opens the Install From dialog box, where you can select a CD-ROM or unzipped CD-ROM image from an FTP transfer to use as the source for installing models. (Same as the Install command on the Actions menu.)
	**Purge Button**— Opens the Select Platform for Purge dialog box, where you can select the platforms from which you want to purge models. (Same as the Purge command on the Actions menu.)
	**Check Button**— Opens the Select Platform for Check dialog box, where you can select a platform and a configuration file that you want to generate a library report for. (Same as the Consistency Check/Report command on the Actions menu).
	**Docs Button**— Opens the *SmartModel Library Administrator's Manual* (this manual) in PDF format using the Acrobat Reader.

Transcript Window

When you invoke the Admin tool, the transcript window is blank except for the copyright statement. When you add models to or purge models from the library, the transcript window lists them by name and version, and displays various status messages. You can save the transcript as a record of your Admin tool session.

If you are having problems during installation, select User Options from the File menu and set messages to Debug before rerunning the installation. Save the transcript to transmit to Technical Support. For contact information, refer to "Getting Help" on page 11.

Status Area

The status area displays the path to the model library whose contents appear in the transcript window. The status area also shows the number of models and timing-versions found, and delivers various status messages.

In addition, the status area displays brief descriptions of the tool bar buttons as you move the pointer over them.

Dialog Boxes

The Admin tool has a set of dialog boxes that pop up to get more information from you when you use to the tool to perform library administration tasks. Following are brief descriptions for how to work with these dialog boxes.

Set Library Directory Dialog Box

The Open Library function opens the Set Library Directory dialog box. Use it to specify the library on which you want to operate. The default is the value of the $LMC_HOME environment variable.

To open a library, follow these steps:

1. In the Name text field, type the name of your library install directory.
2. Click on the OK button. The dialog box closes and the directory name appears in the status area.

Save Transcript Dialog Box

The Save Transcript function opens the Save Transcript dialog box. Use it to specify a file name in which to save the transcript of the Admin tool session.

File Locator Dialog Boxes

The Admin tool has several File Locator dialog boxes that you can use to locate a file:

- Save Transcript
- Locate Media
- Locate File for Install Update
- Locate File for Purge Update
- Locate File to Check
- Read List
- Write List

File Locator dialog boxes have the following fields:

- Filter text field—Contains a file name or file extension to search for in the current working directory using a default file extension or the wildcard (*). You execute the filter function using the Filter button at the bottom of the dialog box.
- Directories list box—Contains a list of directories, initially those within the hierarchy of the home directory. Select a directory from this list where you want the tool to search for files specified in the Filter field.
- Files list box—Contains a list of files that meet the filtering criteria. This field initially contains files in the top-level directory that have the default extension. From this list, you select a file.
- Selection field—Contains the path name to the file that will be opened when you click on the OK button. This field initially contains only the top-level directory, with no file designated. The file you select from the Files list box appears in the Selection field. You can also type a file name directly into the Selection field.

Note

For NT, file locator dialog boxes are implemented using standard Windows functions such as the Save As and Choose Directory dialog boxes.

To specify an existing file

To specify an existing file, follow these steps:

1. Select the file you want from the Files list box. The file name is now appended to the directory path name in the Selection text field.
2. Click on the OK button. The dialog box closes.

To traverse the directory structure

If the file you want is not in the selected directory, you can search for it by following these steps:

1. Select a different directory from the Directories list box.
2. Click on the Filter button. The selected directory now appears as the top-level directory in the Directories list box. The Files list box contains the names of any files in the selected directory that meet the filtering criteria. Notice that the file names are not displayed until you click on the Filter button.

To specify a new file name

To specify a new file name, follow these steps:

1. Edit the Selection text field so that it contains the new file name.
2. Click on the OK button. The dialog box closes.

Options Dialog Box

The User Options function on the File menu opens the Options dialog box. Use it to choose the style of messages sent by the Admin tool for subsequent commands that you enter. You select a message style by clicking on its radio button.

These are the three possible message styles:

- Standard messages—Provides the minimum necessary information.
- Verbose messages—Additionally provides information about unpacking of archives.
- Debug messages—Provides information needed for correcting problems.

To select a message style, follow these steps:

1. Select one of the message style options by clicking on its radio button. By default, the first option is already selected.
2. Click OK. The dialog box closes. The selected message style is enabled.

Install From Dialog Box

The Install command, chosen from the tool bar or the Actions menu, opens the Install From dialog box. Use it to specify the source (CD-ROM or FTP shipment) of library models you want to install.

Locate Media Dialog Box

The Browse button on the Install From dialog box opens the Locate Media dialog box. Use it to locate and specify the CD-ROM or FTP shipment from which to install models.

Note

For NT, file locator dialog boxes are implemented using standard Windows functions such as the Save As and Choose Directory dialog boxes.

Select Models to Install Dialog Box & Select Models to Purge Dialog Box

The Select Models to Install and Select Models to Purge dialog boxes have these fields:

- Items Available list box—Lists the names and version numbers of models or timing-versions available for installing from the CD-ROM or other source; or names and version numbers of models available for purging from your library. For purging, the list box displays only models, but for installing you can toggle the display between models and timing-versions.
- Models to Install/Purge list box—Lists names and version numbers of models you have selected for installing or purging. This field is blank when the dialog box first opens. As you select from the Items Available list, models are added to the Models to Install/Purge list. If you select a timing-version to add to this list, the corresponding model is added, so that the list always contains just model names.
- Add button—Adds to the Models to Install/Purge list those models selected from the Items Available list.
- Add All button—Adds to the Models to Install/Purge list all models on the Items Available list.
- Filter button—Opens the Model Filters dialog box to assist you in specifying a subset of models to select for installing or purging.
- Remove button—Removes selected models from the Models to Install/Purge list.
- Remove All button—Removes all models from the Models to Install/Purge list.
- Read List button—Opens the Read List dialog box, so that you can select a file from which to read in a list of models.

- Write List button—Opens the Write List dialog box, so that you can write to a file the list of models you have selected for installing or purging.
- Model Name/TimVer Name button (Install only)—On the Select Models to Install dialog box only, toggles the display of the Items Available list between model names and timing-version names.
- Purge button (Purge only)—Starts the purge operation. Click on this button only when all models that you want to purge appear in the Models to Purge list box.

Read List and Write List Dialog Boxes

The Read List or Write List dialog box appears when you select Read List or Write List from the Select Models to Install or Select Models to Purge dialog boxes. Use the Read List dialog box to specify a file that contains a list of models to install or purge. Use the Write List dialog box to specify a file that contains a list of models you have installed or purged. Model lists you save can be read in using the Read List function.

Note

For NT, the Read List dialog box is named Open and the Write List dialog box is named Save As.

Read File Example

The following example shows the required format for files that can be read in using the Read List option on the Select Models dialog box.

```
am25s557:01001
cy7c263:01001
gal18v10:01001
inv:01001
l4c381:01001
ncr53c820:01001
plx464:01001
r65c52:01001
s10149:01001
ttl7623:01001
z8536:01001
```

Select Platforms Dialog Box

The Select Platforms dialog box has these fields:

- Header information—Lists the source and destination directories for installed models, and the number of models to be installed.
- Platforms checklist—Lists platform options, each with a check box. Select a platform by clicking on its check box. You can select as many platforms as you want for installation but only one at a time for purge.
- Configuration File text field—For each platform, contains the path to the configuration file to be updated when the models are installed.
- Browse button—Opens the Locate Media dialog box to help you locate the configuration file you want to update.
- EDAV Packages checklist—Lists simulator options, each with a check box. Select a package by clicking on its check box. You can select as many packages as you want, but you must select at least one.
- Install button—Installs the models in the specified library. For NT, use the OK button.

Locate File for Install Update Dialog Box & Locate File for Purge Update Dialog Box

Clicking on Browse in the Select Platforms dialog box or in the Select Platform for Purge dialog box opens the Locate File for Install Update or Locate File for Purge Update dialog box. You can use these dialog boxes to locate a configuration file to be updated when new models are installed in or purged from the library. By default, the Admin tool updates the configuration file in your $LMC_HOME/data directory.

Note

For NT, file locator dialog boxes are implemented using standard Windows functions such as the Save As and Choose Directory dialog boxes.

Model Filters Dialog Box

The Filter function opens the Model Filters dialog box. You can use it to specify filter options for displaying a subset of the model library. Select one or more of the four filter options by clicking on the appropriate check box. The dialog box fields are as follows:

- String Match field—Contains a string that specifies a model name or timing version name to search for; or part of a name plus the wildcard character (*). This field is initially set to display all models or timing versions. By default, this field is selected when the dialog box opens.
- Vendors list box—Contains a list of vendors to select as a filter option. Use this field if you want to narrow the display to models of parts from specific vendors.
- Function/Subfunction list box—Contains a list of functions and subfunctions to select as a filter option. Use this field if you want to narrow the display to models that have specific functions.
- Licensed Packages list box—Contains a list of licensed packages to select as a filter option. Use this field if you want to narrow the display to models contained in one or more specific licensed packages, or if you want to know whether a specific model is contained in a particular licensed package.
- Summary of Filter Options field—Displays the filter options currently selected.

To filter a model list, follow these steps:

1. Click on the check box for each desired option.
2. For each option you checked, select one or more items from its list box. Use the scroll bar to traverse the list. To select more than one item, hold down the Ctrl key as you click on each item. To specify a model or timing-version to search for, type the complete name of the model or timing-version, or a partial name with the wildcard character (*) in the String Match text field in each position where you have omitted characters. The name you typed appears in the Summary of Filter Options list box. If you select multiple items, they appear ORed together.
3. When you finish selecting filter options, click on the Filter or OK button.
4. Click on the Close button to dismiss the dialog box. The list box in the transcript window displays a list of models and timing versions that meet the filtering criteria. The status pane displays the number of models and timing-versions found.

Select Platform for Purge Dialog Box & Select Platform for Check Dialog Box

Choosing the Purge or Consistency Check/Report commands from the Actions menu opens the Select Platform for Purge or the Select Platform for Check dialog box. The two dialog boxes are similar, and have these fields:

- Platform list—Lists platform options, each with a radio button. The platform is selected when its radio button is selected. You can select only one platform.
- Configuration File text field—Contains the path to the configuration file to be updated when the models are purged; or the configuration file to be used for comparison when the models are checked.
- Browse button—Opens the Locate File for Purge Update or Locate File to Check dialog box to assist you in locating the configuration file.
- Continue button—Opens the Select Models to Purge dialog box or the Library Report dialog box so that you can choose models to purge or select one of the Library Report options.
- Number of versions to keep text field—Use this field to specify the number of model versions to protect from the purge operation. Legal values are 2 through 99 and the default is 3, which protects the 3 most recent versions of the model from the purge.

Locate File to Check Dialog Box

The Browse button on the Select Platform for Check dialog box opens this dialog box. You can use it to specify a configuration file to be checked against your installed library when generating a library report. By default, the Admin tool uses the configuration file in your $LMC_HOME/data directory, based on your platform.

Library Report Dialog Box

Clicking on the Continue button in the Select Platform for Check dialog box opens the Library Report dialog box. You can use this dialog box to generate a library report or execute library maintenance commands. To select a report or command, click on its radio button.

The dialog box fields are as follows:

- Environment variable field—Contains paths referenced by the environment variables used by the Admin tool. You can use this information to verify that the models are referenced as you intended.

- Report/command selection field—Contains options you can choose for generating a report, and two library maintenance commands, as described below:
  - Report Configuration file errors—Generates a report of models listed either in the default configuration or custom configuration files but not found in the library installed at your site.
  - Report models found in library but not in any configuration files—Generates a report of models in the library installed at your site but not listed either in the default configuration or in your custom configuration files.
  - List all models with their source configuration files—Generates a report that shows each installed model and the configuration file where it is listed.
  - Run consistency check on entire Library—Generates a report on the integrity of the library database.
  - Report on entire Library Content—Generates a report showing all versions of all models in the library. For more information about library reports, refer to "Repairing Errors from a Library Report" on page 26.

Regenerate Library Files Dialog Box

Selecting the Regenerate command from the Actions menu opens the Regenerate Library Files dialog box. Executing this command produces the following effects:

- Regenerates the default configuration (LMC) file—All values are set equal to the latest version of each model found. Any update information about models previously added or purged is lost.
- Rebuilds the library cache to provide faster access to the models.
- Resets all library-versioned links to point to the latest version of SWIFT.

The Regenerate command works on one platform at a time; you must run it a separate time for each platform.

2
SmartModel License Agreement

Introduction

Following is the text of the SmartModel license agreement:

DO NOT USE THE LICENSED SOFTWARE UNTIL YOU HAVE READ THE LICENSE AGREEMENT. BELOW. YOUR USE OF THE LICENSED SOFTWARE CONSTITUTES ACCEPTANCE OF THE TERMS AND CONDITIONS OF THIS LICENSE AGREEMENT. IF YOU HAVE ANY QUESTIONS, PLEASE CONTACT THE LOGIC MODELING GROUP OF SYNOPSYS, INC. ("SYNOPSYS").

1. LICENSE. Synopsys hereby grants Licensee a non-exclusive, nontransferable, limited license, without right of sublicense to (i) use the Licensed Software in machine-readable form at the Use Area during the term and in accordance with the related Documentation; (ii) use and copy the related documentation; and (iii) make copies of Licensed Software in machine-readable form only for archival and backup purposes. Synopsys and/or its licensors retain all right, title and interest in and to Licensed Software. No other rights or license are granted or implied to use Licensed Software or to license or authorize others to use Licensed Software beyond those set forth in this Agreement.

"Documentation" is any user manuals, reference manuals, release, application and methodology notes, written utility programs and other materials in any form provided for use with the Licensed Software.

"Use Area" is the: Key Server(s) [the one or three computational nodes, which control access to and enable the Licensed Software], Client(s) [an enabled instance of Licensed Software running on a single processor computational node], and End User(s) [the authorized person(s) who access and use the Client] all located within the same five (5) miles radius, unless such Use Area is defined in a separate quotation and agreed upon by the parties.

2. COPIES. Licensed Software and Documentation are copyrighted. Licensee may not copy or transfer to a third-party any Licensed Software or Documentation except as provided in this Agreement. Licensee must reproduce and include the copyright notice and any other notices that appear on the original copy of Licensed Software and Documentation on any copies made thereof by Licensee.

3. OWNERSHIP. Synopsys and/or its licensors own and shall retain all right, title and interest in and to Licensed Software and Documentation, including all patents, patent rights, copyrights, trade secrets, service marks, mask works and trademarks, and any applications for any of the foregoing, in all countries in the world ("Intellectual Property Rights") embodied therein, and Licensee shall have no rights with respect thereto other than the rights expressly set forth in this Agreement. Title, in the media only, passes upon Licensee's receipt of Licensed Software and Documentation.

4. LICENSE RESTRICTIONS. For software shipped in source code format, Licensee may modify Licensed Software provided that Licensee is solely responsible for supporting such modified software. Synopsys retains all proprietary rights to such modified Licensed Software, which shall be deemed to be licensed to Licensee under the terms of this Agreement. Licensee shall not attempt to convert or compile Licensed Software from one form to another, or attempt to reverse-engineer, reverse-compile or reverse-assemble Licensed Software, except as necessary for the authorized use of Licensed Software as contemplated in this Agreement.

Third-party proprietary information may have been used in the development of certain portions of Licensed Software. Licensee will not use or attempt to use Licensed Software to reverse-engineer any third-party components modeled or described by Licensed Software. This Agreement does not grant Licensee any rights or license, express or implied, to any third-party Intellectual Property Rights. In cases where third-party proprietary information was used, Synopsys' third party licensors may enforce their rights under this Section 4. Third-parties are listed in the Documentation.

Licensee agrees not to use the Licensed Software to provide modeling or system verification services to third parties. Licensee further agrees not to provide, lease, lend, use for timesharing or service bureau purposes, or otherwise use or allow others to use the Licensed Software and Documentation for the benefit of third parties.

5. PAYMENT AND DELIVERY TERMS

5.1 Purchase Order. Licensee's receipt and use of Licensed Software and Documentation shall be governed by the terms and conditions of this Agreement. Nothing contained in any purchase order, purchase order acknowledgment, or invoice shall in any way modify such terms or add any additional terms or conditions.

5.2 Delivery. Upon acceptance of Licensee s purchase order, Synopsys shall deliver to Licensee, at Synopsys' expense, the Licensed Software. Any lost or damaged media will be replaced by Synopsys at minimal charge.

5.3 Payment. Licensee agrees to pay Synopsys upon receipt of invoice from Synopsys, the license fee for Licensed Software. Payment terms are net thirty (30) days from date of invoice.

6. LIMITED WARRANTY. The extent of warranty provided by Synopsys will depend upon the type of license granted to Licensee in the authorization code. (a) Warranty for Non-perpetual License: If the term of this license is non-perpetual, then Synopsys warrants that Licensed Software will substantially conform to the specifications in the Documentation for a period equal to the term of such license as determined by the authorization code; (b) Warranty for Perpetual License: If the term of this license is perpetual, then Synopsys warrants for a period of ninety (90) days from delivery of Licensed Software to Licensee that such Licensed Software, as delivered, will substantially conform to the specifications in the Documentation. SUPPORT FOR THE PERPETUAL LICENSE IS NOT COVERED BY THIS AGREEMENT AND MAY BE PURCHASED BY LICENSEE PURSUANT TO A SEPARATE AGREEMENT AS SET FORTH IN SECTION 8 BELOW.

Under the warranty set forth above, Synopsys accepts no responsibility and explicitly disclaims any warranty for all or any part of Licensed Software which has been misused, or if the Licensed Software has been modified by Licensee pursuant to Section 4.0. There are no warranties with respect to third-party information used to develop Licensed Software.

Synopsys does not warrant that: (i) operation of any of the Licensed Software shall be uninterrupted or error free, or (ii) functions contained in the Licensed Software shall operate in the combinations which may be selected for use by Licensee or meet Licensee requirements. Synopsys' sole obligation for breach of warranty will be, at its option, to provide a copy of conforming Licensed Software to Licensee or refund the license fee paid, less a reasonable allowance for the period of successful use. THIS LIMITED

WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON INFRINGEMENT, OR ARISING FROM A COURSE OF DEALING OR USAGE OF TRADE.

7. LIMITATION OF LIABILITY. SYNOPSYS' TOTAL LIABILITY FOR DIRECT DAMAGES UNDER THIS AGREEMENT SHALL NOT EXCEED THE TOTAL FEE RECEIVED BY SYNOPSYS FROM LICENSEE. UNDER NO CIRCUMSTANCES, SHALL SYNOPSYS OR ITS LICENSORS BE LIABLE FOR ANY SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING IN ANY WAY OUT OF THIS AGREEMENT OR THE USE OF LICENSED SOFTWARE AND DOCUMENTATION, HOWEVER CAUSED, (WHETHER ARISING UNDER A THEORY OF CONTRACT, TORT (INCLUDING NEGLIGENCE); OR OTHERWISE), INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOST PROFITS, LOSS OF DATA, INTERRUPTION OF SERVICE, OR COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES. THE LIMITATIONS ON SYNOPSYS' AND ITS LICENSORS' LIABILITY SET FORTH IN THIS SECTION 7 SHALL APPLY NOTWITHSTANDING THE FAILURE OF ESSENTIAL PURPOSE OF ANY OF THE LIMITED REMEDIES SET FORTH IN SECTION 6 ABOVE. Some states and provinces do not allow the exclusion or limitation of implied warranty or the limitation of incidental and consequential damages, so the above limitation may not apply to Licensee.

8. ANNUAL SUPPORT AND MAINTENANCE OPTION. Licensee may purchase Annual Site Services and/or Updates services, if offered by Synopsys, to commence upon delivery of the Licensed Software, by paying the applicable fees then in effect. If Licensee allows site services and/or updates to lapse, then Licensee may resume site services and/or updates, respectively by paying the applicable then-current annual fee(s) in addition to Synopsys' resumption fee.

9. TERM AND TERMINATION

9.1 Term. The term of this Agreement will be equal to the term of the license granted in the authorization code, unless terminated earlier in accordance with the terms of this Agreement.

9.2 Termination. This Agreement shall apply to any Licensed Software in the possession of, or used by, Licensee. Synopsys may terminate this Agreement if Licensee breaches any material provision of this Agreement. The termination of any license, or rejection of any particular Licensed Software program shall not affect the application of this Agreement to other Licensed Software. Upon expiration or termination of this Agreement, Licensee shall destroy or render ineffective any authorization codes for such Licensed Software. Sections 3, 4, 6, 7, 10 and 11 shall survive any termination or expiration of this Agreement.

10. GOVERNMENT MATTERS

10.1 Export Controls. Licensee agrees and certifies that neither Licensed Software or Documentation, nor any other technical data received from Synopsys, nor the direct product thereof, will be exported or re-exported outside the United States except as authorized and as permitted by the laws and regulations of the United States.

10.2 Government Use. Licensee represents that it is not a government agency nor is it obtaining any Synopsys material under this Agreement pursuant to a government contract or with government funds. If Licensee is acquiring any Synopsys material on behalf of any unit or agency of the United States Government, then Licensee will notify Synopsys in writing prior to delivery of any such materials and will obtain the Government's agreement as follows:

(i) if Licensed Software and Documentation are being supplied to the Department of Defense ("DOD"), Licensed Software is classified as "Commercial Computer Software" and the Government is acquiring only "restricted rights", and as to the Documentation the Government is acquiring only "limited rights" (as those terms are defined in Clause 252.227-7013 of the DFARS); and

(ii) if Licensed Software and Documentation are being supplied to any unit or agency of the United States Government other than DOD, Licensed Software is classified as "Commercial Computer Software" and the Government is acquiring only "restricted rights" and as to the Documentation the Government is acquiring only "limited rights" (as those terms are defined in Clause 52.227-19 of the FAR or, in the case of NASA, in Clause 18-52.227-86 of the NASA Supplement to the FAR).

11. GENERAL PROVISIONS. (a) This Agreement will be governed by and construed in accordance with the laws of the United States and the State of California; (b) This Agreement may not be assigned by Licensee; (c) Failure by either party to enforce any provision of this Agreement will not be deemed a waiver of future enforcement of that or any other provision; (d) If for any reason a court of competent jurisdiction finds any provision of this Agreement, or portion thereof, to be unenforceable, that provision of the Agreement will be enforced to the maximum extent permissible so as to effect the intent of the parties, and the remainder of this Agreement will continue in full force and effect; (e) The prevailing party in any action to enforce the Agreement shall be entitled to recover costs and expenses including, without limitation, reasonable attorneys' fees; (f) This Agreement may be amended only by a written agreement signed by authorized representatives of Synopsys and Licensee; (g) This Agreement, including the Equipment Lease Agreement, the Source Models Software License Agreement, the Non-disclosure Agreement and the attached quotes (if applicable), constitutes the entire agreement between the parties with respect to the subject matter hereof, and supersedes all prior agreements or representations, oral or written, regarding such subject matter.

3 GNU General Public License

Introduction

This chapter provides information about the GNU General Public License. This information is presented in the following major sections:

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions:

"Package"
: Refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version"
: Refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder"
: Is whoever is named in the copyright or copyrights for the package.

"You"
Is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee"
Is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available"
Means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.
3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:
    1. place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
    2. rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.
    3. make other distribution arrangements with the Copyright Holder.
4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:
    1. distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.
    2. accompany the distribution with the machine-readable source of the Package with your modifications.

3. give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.
4. make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.
6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whomever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package via the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.
7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.
8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.
9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.
10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you

cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free

Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
This program is free software; you can redistribute it and/or modify it
under the terms of the GNU General Public License as published by the
Free Software Foundation; either version 2 of the License, or (at your
option) any later version.
This program is distributed in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General
Public License for more details.
You should have received a copy of the GNU General Public License along
with this program; if not, write to the Free Software Foundation, Inc.,
675 Mass Ave, Cambridge, MA 02139, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author Gnomovision
comes with ABSOLUTELY NO WARRANTY; for details type ‘show w'. This is
free software, and you are welcome to redistribute it under certain
conditions; type ‘show c' for details.
```

The hypothetical commands ‘show w' and ‘show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ‘show w' and ‘show c'; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
‘Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

4
Third-Party Tools

Introduction

In the course of creating and distributing its software, Synopsys uses various third-party tools. This chapter provides general information about these tools, presented in the following sections:

- Perl
- Info-ZIP
- WISH

Perl

Perl is an interpreted programming language optimized for scanning arbitrary text files, extracting information from those text files, and printing reports based on that information. Perl is also used for many system management tasks. SmartModels use Perl for various scripts and utilities.

For detailed information, refer to the Perl 5 Web page at:

http://www.metrnet.com/perlinfo/perl5.html

You can order Perl books from O'Reilly & Associates, 1-800-998-9938 (domestic) or +1 707 829 0515 (overseas). If you have an O'Reilly order form, you can FAX it to +1 707 829 0104.

- *Programming Perl* is a reference work that covers nearly all of Perl (version 4).

  Programming Perl (the Camel Book):

  ISBN 0-937175-64-1 (English)
  ISBN 4-89052-384-7 (Japanese)

- *Learning Perl* is a tutorial that covers the most frequently used subset of the language.

  Learning Perl (the Llama Book):

  ISBN 1-56592-042-2 (English)
  ISBN 4-89502-678-1 (Japanese)
  ISBN 2-84177-005-2 (French)
  ISBN 3-930673-08-8 (German)

Perl is covered by the Artistic License. We have also included a copy of the GNU General Public License.

How to Get Perl

Perl source code is widely available from any comp.sources.unix archive. The following machines have Perl available via anonymous FTP:

```
ftp.uu.net 137.39.1.9
archive.cis.ohio-state.edu 128.146.8.52
jpl-devvax.jpl.nasa.gov 128.149.1.143
```

If you do not have Internet access, Perl is available via anonymous uucp from both uunet and osu-cis.

If you need Perl source code and are unable to access any of these sources, contact the Synopsys Technical Support Center for assistance.

Info-ZIP

The SmartModel Library is packaged on this CD using Info-ZIP's compression utility. The installation program uses UnZip to read zip files from the CD. There are no extra charges or costs due to the use of this code.

How to get Zip/UnZip

Info-ZIP's software (Zip, UnZip, and related utilities) is free and can be obtained as source code or executables from various docBulletin board services and anonymous-ftp sites, including CompuServe's IBMPRO forum and ftp.uu.net:/pub/archiving/zip/*.

WISH

5
FLEXlm Command Reference

Introduction

SmartModel Library products are authorized using FLEXlm software from GLOBEtrotter. The complete set of FLEXlm documentation is available in $LMC_HOME/doc/flexlm/TOC.htm after installation. However, for your convenience we have included the command reference information here in case you need access to this information prior to installation. Command reference information is available in this chapter for the following FLEXlm commands:

- license.dat
- license.opt
- lmcksum
- lmdown
- lmgrd
- lmhostid
- lmremove
- lmreread
- lmstat
- lmutil
- lmver

For more information about authorizing models, refer to "Step 1: Authorize the SmartModel Products" in the *SmartModel Library Installation Guide*.

license.dat

NAME

license.dat

License configuration file for FLEXlm licensed applications

SYNOPSIS

/usr/local/flexlm/licenses/license.dat

DESCRIPTION

A license file consists of the following sections:

- Optional license server section, with information about node where the SERVER (or redundant SERVERs) are running, along with a list of all vendor-specific DAEMON(s) that the SERVER needs to run. This section is required if any features are counted.
- Features section, consisting of any combination of FEATURE, INCREMENT, UPGRADE or PACKAGE lines. This section is required.
- Optional FEATURESET section, consisting of at most one FEATURESET line per DAEMON line in the file.
- Comments. The convention is to begin comments with a '#' character. However, in practice all lines not beginning with a FLEXlm reserved keyword are considered comments.

Vendors and end-users will read the license file to understand how the licensing will behave: what features are licensed, for how many users, whether these features are node-locked, if the features are demo or regular, etc. The options are very broad for what can be specified in a license file.

End-users often need to edit this file. Nearly all of the file is encrypted, and if these portions are edited by the end-user, a LM_BADCODE error will result. However, end-users often must edit the license file in order to change the SERVER nodename, the SERVER TCP/IP port address, or the path to the vendor daemon.

SERVER Line

A SERVER line specifies the node on which a license server process can run. If a license file has three SERVER lines, then two of the three servers must be running in order for the licensing system to operate. The SERVER node name in the license file can be any network alias for the node (prior to FLEXlm v2.4 it was restricted to the return of the gethostname() call).

SERVER *nodename id optional-port-number*

nodename
The string returned by the Unix "hostname" command. This can be edited by the end-user

id
The string returned by the lmhostid command. (Case insensitive). In addition, a hostid of ANY can be specified, which will allow the license to be run on any node.

optional-port-number
The TCP port number to use. This can be edited by the end user.

Note
The SERVER line must apply to all lines in the license file. It is permitted and encouraged to combine license files from different vendors, but this only works if the SERVER hostids are identical in all files that are to be combined.

DAEMON Line

The DAEMON line specifies the name and location of a vendor daemon, as well as the location of the end-user options file.

DAEMON *daemon-name path options*

daemon-name
The name of the daemon used to serve some feature(s) in the file.

path
The pathname to the executable code for this daemon.

options
The pathname of the end-user specified options file for this daemon.

FEATURE or INCREMENT Line

A FEATURE line describes the license to use a product. An INCREMENT line can be used in place of a FEATURE line, as well as to add licenses to a prior FEATURE or INCREMENT line in the license file.

Note

If the vendor daemon has ls_use_all_feature_lines set, then FEATURE lines function as INCREMENT lines, and the behavior of a FEATURE line is unavailable to that application. If ls_use_all_feature_lines is set, read INCREMENT for FEATURE.

Only one FEATURE line for a given feature will be processed by the vendor daemon. If you want to have additional copies of the same feature (for example, to have multiple node-locked counted features), then you must use multiple INCREMENT lines. INCREMENT lines form license groups based on the feature name, version, and node-lock hostid. If the feature name, version, and node-lock hostid (and optionally, the vendor string, if ls_compare_vendor_on_increment is non-zero) match a prior INCREMENT or FEATURE line, the new number of licenses is added to the old number. If any of the three do not match, a new group of licenses is created in the vendor daemon, and this group is counted independently from others with the same feature name. INCREMENT is not available for pre-v2.61 FLEXlm clients or servers. A FEATURE line does not give an additional number of licenses, whereas an INCREMENT line ALWAYS gives an additional number of licenses. There are two formats for FEATURE, pre-v3.0 and current. The old format is still understood and correct with new clients and servers, but the new format is more flexible, and therefore recommended where possible.

FEATURE|INCREMENT *name daemon version exp_date #lic code* "*vendor_string*" [*hostid*] (Pre-v3.0 format)

Or:

FEATURE|INCREMENT *name daemon version exp_date #lic code*
[HOSTID=*hostid***][VENDOR_STRING="vendor-string"]**
[vendor_info="..."] [dist_info="..."] [user_info="..."] [asset_info="..."]
[ISSUER="..."] [NOTICE="..."] [*ck=nnn***] [OVERDRAFT=***nnn***]**
[DUP_GROUP=NONE|SITE|[UHDV]] (Current format)

name

The name given to the feature. Legal feature names in FLEXlm must start with either a letter, a number, or the underscore character.

daemon
: The daemon-name from a DAEMON line. The specified daemon serves this feature.

version
: The latest (highest-numbered) version of this feature that is supported. (3 decimal places).

exp_date
: The expiration date in the format: dd-mmm-yyyy, (for example, 22-mar-1988). If you do not want the feature to expire, select an expiration date with the year as 0 (any day- any month-0). (Case insensitive).

#lic
: The number of licenses for this feature.

code
: The encryption code for this feature line. The code is produced by lc_crypt() in create_license, or lc_crypstr() in lmcrypter. (Case insensitive). The start date is encoded into the code; thus identical codes created with different start dates will be different. The encryption code field can be made case sensitive by calling lc_set_attr(LM_A_CRYPT_CASE_SENSITIVE, 1). Remember to select the "case sensitive" option when building your daemon, if you use a case sensitive encryption code comparison.

The following fields are all optional (except for vendor-string in the old format). For optional fields of the "name=value" syntax, if the name is lowercase, it can be modified and the license will remain valid.

vendor-string
: The vendor-defined string, enclosed in double quotes. This string can contain any characters except a quote (white space will be ignored in the vendor-defined string). The application can retrieve this string by calling lc_auth_data(). [Pre-v3.0 format]

hostid
: The string returned by lmhostid. Use if this feature is to be bound to a particular host, whether its use is counted or not. (Case insensitive). If the number of licenses is 0, then this field is required. If hostid is DEMO, the hostid is any system. If DEMO, the application can determine this is a demo license by calling lc_auth_data() and noting the hostid type. [Pre-v3.0 format]

HOSTID=*hostid*
: The node-locked hostid (same as "hostid" in the old format, above)

VENDOR_STRING="..."
: The vendor-defined license data (same as "vendor_string" in the old format, above)

vendor_info="..."
: Additional information provided by the software vendor. Not encrypted into the feature's "code".

dist_info="..."
: Additional information provided by the software distributor. Not encrypted into the feature's "code".

user_info="..."
: Additional information provided by the software end-user's system administrator. Not encrypted into the feature's "code".

asset_info="..."
: Additional information provided by the software enduser's system administrator for asset management. Not encrypted into the feature's "code".

ISSUER="..."
: Issuer of the license.

NOTICE="..."
: A field for intellectual property notices.

ck=*nnn*
: A checksum, useful with the lmcksum utility, which will verify that the license has been entered correctly by the enduser. Not encrypted.

OVERDRAFT=*nnn*
: The OVERDRAFT policy allows you to specify a number of additional licenses which your end-user will be allowed to use, in addition to the licenses they have purchased. This is useful if you want to allow your users to not be denied service when in a "temporary overdraft" state. Usage above the licensed limit will be reported by the FLEXadmin reporting tool. In addition, you can determine if you are currently in an overdraft condition by calling lc_get_attr(job, LM_A_VD_FEATURE_INFO, members of interest: lic_in_use, lic_avail, and overdraft.

  If lic_in_use is greater than lic_avail - overdraft, then you are in an "overdraft state".

DUP_GROUP=...
: You can also specify the Duplicate Grouping parameter in the license in FLEXlm v3.2. If DUP_GROUP is specified in the license, this parameter overrides the

dup_group parameter in the lc_checkout() call. If not specified in the license, the dup_group parameter from lc_checkout() will be used, as in prior versions of FLEXlm. The syntax is:

DUP_GROUP=NONE|SITE|[UHDV] U = DUP_USER
H = DUP_HOST
D =DUP_DISPLAY V = DUP_VENDOR_DEF

Any combination of UHDV is allowed, and the DUP_MASK is the OR of the combination. For example "DUP_GROUP=UHD" means the duplicate grouping is (DUP_USER|DUP_HOST|DUP_DISPLAY), so a user on the same host and display will have additional uses of a feature not count.

Examples

To illustrate INCREMENT, the two feature lines:

```
FEATURE f1 demo 1.000 1-jan-0 4... FEATURE f1 demo 2.000 1-jan-0 5...
```

would only result in 4 licenses for v1 or 5 licenses for v2, depending on their order in the file, whereas:

```
INCREMENT f1 demo 1.000 1-jan-0 4... INCREMENT f1 demo 2.000 1-jan-0 5
...
```

would result in 4 licenses for v1 and 5 licenses for v2 being available, giving a total of 9 licenses for f1.

To illustrate counted vs. uncounted licenses, the following FEATURE line:

```
FEATURE f1 demo 1.000 1-jan-95 0 12345678901234567890 HOSTID=DEMO
```

has unlimited usage on any hostid, requires no lmgrd server or SERVER or DAEMON lines (and is therefore a complete license file by itself), and expires on 1-jan-95. In contrast:

```
FEATURE f1 demo 1.000 1-jan-0 5 12345678901234567890 \
HOSTID=INTERNET=195.186.*.*
```

This FEATURE requires the demo vendor daemon to be running, is limited to 6 users on any host with an internet IP address matching 195.186.*.*, and never expires.

Note

Assuming that ls_use_all_feature_lines is 0, then only one FEATURE line for a given feature will be processed by the vendor daemon. If you want to have additional copies of the same feature (for example, to have multiple node-locked counted features), then you must use multiple INCREMENT lines. INCREMENT lines form license groups based on the feature name, version, and node-lock hostid. In other words, if the feature name, version, and node-lock hostid (and optionally, the vendor string, if ls_compare_vendor_on_increment is non-zero) match a prior INCREMENT or FEATURE line, the new number of licenses is added to the old number. If any of the three do not match, a new group of licenses is created. INCREMENT is not available for pre-v2.61 FLEXlm clients or servers.

UPGRADE Line

UPGRADE *name daemon fromversion version exp_date #lic code* "*string*" [*hostid*] **ck=nnn**

All the data is the same as for a FEATURE or INCREMENT line, with the addition of the fromversion field. An UPGRADE line removes up to the number of licenses specified from any old version (>= fromversion) and creates a new version (version) with that same number of licenses.

For example, the two lines:

```
FEATURE f1 demo 1.000 1-jan-94 5 9BFAC03164EDB7BC0462
UPGRADE f1 demo 1.000 2.000 1-jan-94 2 1B9A30316207EC8CC0F7 ""
```

would result in 3 licenses of FLEXlm v1.0 of f1 and 2 licenses of v2.0 of f1. UPGRADE will operate on the most recent FEATURE or INCREMENT line (i.e., closest preceding FEATURE or INCREMENT line) with a version number that is >= fromversion, and < version.

Note that UPGRADE does not work for node-locked, uncounted licenses. A new FEATURE line should be issued in this case, since the license count is irrelevant.

If a user has UPGRADE lines for his license file and performs the following sequence:

1. Start daemons WITHOUT UPGRADE lines in license file
2. Check out old versions of software
3. Add UPGRADE lines to license file
4. Run lmreread

The user will have more checked-out licenses of the old version of the software than the FEATURE/UPGRADE or INCREMENT/UPGRADE lines would normally allow. This license overdraft (of the old versions) will only exist for the life of the old processes.

PACKAGE Line

The purpose of the PACKAGE line is to support two different vendor needs: to license a product SUITE, or to provide a more efficient way of distributing a license file that has a large number of features, which largely share the same FEATURE line arguments. A PACKAGE line, by itself, does not license anything—it requires a matching FEATURE/INCREMENT line to license the whole PACKAGE. A PACKAGE line can be shipped with a product, independent of any user information. Later, the user can purchase one or more corresponding FEATURE/INCREMENT licenses that will turn on the PACKAGE.

PACKAGE *pgn_name vendor pkg_version pkg_key* **COMPONENTS**=*pkg_list* [**OPTIONS**=*pkg_options*]

pkg_name
: Name of the PACKAGE. The corresponding FEATURE/INCREMENT line must have the same name.

vendor
: Name of the vendor daemon that supports this PACKAGE (VENDOR_NAME in lm_code.h).

pkg_version
: Version of the PACKAGE. The corresponding FEATURE/INCREMENT line must have the same version.

pkg_key
: 20-character encryption code.

pkg_list
: The list of components. Format is:

feature[:*version*[:*count*]]

The PACKAGE must consist of at least one COMPONENT version and count are optional, and if left out, their values come from the corresponding FEATURE/INCREMENT line. count is only legal if OPTIONS=SUITE is not set—in this case the resulting number of licenses will be the count on the COMPONENTS line multiplied by the number of licenses in the FEATURE/INCREMENT line. Examples:

```
COMPONENTS="comp1 comp2 comp3 comp4"
COMPONENTS="comp1:1.5 comp1 comp1:2.0:4"
```

OPTIONS=*pkg_options*

Currently the only legal option is SUITE. This is what distinguishes a suite PACKAGE from a PACKAGE used to ease distribution. With OPTIONS=SUITE, the corresponding FEATURE is checked out in addition to the component/feature being checked out. If OPTIONS=SUITE is not set, then the corresponding FEATURE is removed once the PACKAGE is turned on, and it also is not checked out when a component/feature is checked out.

Examples

```
PACKAGE suite demo 1.0 20CHARCODEXXXXXXXXXX
COMPONENTS="comp1 comp2" OPTIONS=SUITE FEATURE
suite demo 1.0 1-jan-0 5 20CHARCODEXXXXXXXXXX
```

This is a typical, simple, OPTIONS=SUITE example. The user will have 2 features available: comp1 and comp2, which are each version 1.0, 5 uses available, unexpiring.

When comp1 or comp2 are checked out, "suite" will also be checked out. The vendor will most likely want to turn on DUP_GROUP (either through the FEATURE line, or lc_checkout) so that the same user can use comp1 and comp2 while using only one use of FEATURE suite.

```
PACKAGE suite demo 1.0 20CHARCODEXXXXXXXXXX
COMPONENTS="comp1 comp2 comp3 comp4 comp5"
INCREMENT suite demo 1.0 1-jan-0 1
20CHARCODEXXXXXXXXXX HOSTID=12345678
INCREMENT suite demo 1.0 1-jan-0 1
20CHARCODEXXXXXXXXXX HOSTID=12345678
```

This is a good way to distribute multiple node-locked, counted licenses. Rather than requiring 5 INCREMENT lines per node, only one INCREMENT line is required per node, and the features are indicated in the PACKAGE line.

```
PACKAGE suite demo 1.0
COMPONENTS="comp1:1.5:2 comp2:3.0:4 comp3"
FEATURE suite demo 1.0 1-jan-95 3
20CHARCODEXXXXXXXXXX ISSUER=distrib1
```

The component versions override the FEATURE versions, and the uses available is the product of the 3 uses for suite and the components count. The result is equivalent to:

```
FEATURE comp1 demo 1.5 1-jan-95 6
20CHARCODEXXXXXXXXXX ISSUER=distrib1
FEATURE comp2 demo 3.0 1-jan-95 12
20CHARCODEXXXXXXXXXX ISSUER=distrib1
FEATURE comp3 demo 1.0 1-jan-95 3
20CHARCODEXXXXXXXXXX ISSUER=distrib1
```

FEATURESET Line

FEATURESET *daemon-name code*

daemon-name
The name of the daemon used to serve some feature(s) in the file.

code
The encryption code for this FEATURESET line. This code encrypts the codes of all FEATUREs that this daemon supports, so that no FEATURE lines can be removed or added to this license file.

The FEATURESET line allows the vendor to bind together the entire list of FEATURE lines supported by one daemon. If a FEATURESET line is used, then all the FEATURE lines must be present in the same order in the customer's license file. This is used, for example, to insure that a customer uses a complete update as supplied, without adding in old FEATURE lines from the vendor.

Any amount of white space of any type can separate the components of license file lines, and the data can be entered via any text editor. Vendors can therefore distribute license data via FAX or telephone.

Only four data items in the license file are editable by the end-user:

```
hostnames on SERVER lines
port-numbers on SERVER lines
pathnames on DAEMON lines
options file pathnames on DAEMON lines
```

The SERVER hostid(s) and everything on the FEATURE line (except the daemon name) is input to the encryption algorithm to generate the code for that FEATURE.

Comments

Comment lines can begin with a '#'.

Continued Lines

Lines can be continued with a "

License file name limits

The limits on names for the major parameters employed in the FLEXlm license file are:

Host Names
32 characters

Feature Names
30 characters

Date String
11 characters (dd-mmm-yyyy)

License file lines (total)
2048 characters

User Names
20 characters

DAEMON Names
10 characters

Version
10 characters, in floating point format, i.e., nnn.nnn

All other parameters
Limited only by the total length of the license file line.

When using TCP, the maximum number of spawned daemons is 255, which effectively limits the maximum number of end-users to 256*25 -(2* # servers -1) or approximately 6,400 per vendor daemon. When using UDP, there is no limit to the number of end-users. Note that multiple daemons can be run on a single network, making the number of even TCP end-users effectively unlimited.

Example License File

```
SERVER pat 17003456 1700
SERVER lee 17004355 1700
SERVER terry 17007ea8 1700
DAEMON demo /etc/mydaemon
FEATURE f1 demo 1.000 0-jan-96 10 1AEEFC8F90030EABF324
FEATURE f2 demo 1.000 01-jan-96 10 0A7E8C4F561FE98BA073
```

This example illustrates the license file for single vendor with two features, and a set of three server nodes, any two of which must be running for the system to function.

Locating the License File

The license file default is:

/usr/local/flexlm/licenses/license.dat

Client applications get the license file location in the following manner (in order of precedence; lowest to highest):

1. Default location - /usr/local/flexlm/licenses/license.dat.
2. lc_set_attr(LM_A_LICENSE_FILE_PTR, path).

3. End-user or application sets LM_LICENSE_FILE environment variable to path.
4. LM_A_LICENSE_FILE_PTR set and lc_set_attr(LM_A_DISABLE_ENV,1), which makes the application override the LM_LICENSE_FILE environment variable.

Most FLEXlm utilities will accept a "-c license_file_path" option, in order to use a different license file.

The LM_LICENSE_FILE environment variable can be used to establish a new default location for the license file (note that a "-c" option will override the setting of LM_LICENSE_FILE). In addition, client programs can process a series of license files by setting LM_LICENSE_FILE to a path, as in:

```
% setenv LM_LICENSE_FILE file1:file2:file3:....:filen
```

or, on VMS:

```
$ assign file1/file2/file3/.../filen LM_LICENSE_FILE
```

Client programs will then try using file1; if it fails, file2 will be tried, and so on.

Note that the FLEXlm daemons do NOT use the license file path; they will only process the first file in a license file path.

Starting in FLEXlm v2.4, the license file no longer needs to be accessible on each client node; it can be read by the client from lmgrd. In order to do this, specify the license file as "port@host", either in the LM_LICENSE_FILE environment variable or in your call to lc_set_attr(LM_A_LICENSE_FILE_PTR,...);

The new syntax is:

port@host[,*port2@host2*[,*port3@host3*]]

port
: The (integer) TCP/IP port number from the license file

host
: The name of the server host from the license file.

A non-redundant server would have only a single port@host, whereas redundant servers could be specified as 1, 2, or 3 sets of "port@host", separated by commas. For example, if you have a single server node named "serverhost", and you are running FLEXlm on port 1700, you could specify your "license file" as:

```
1700@serverhost
```

You could have a license file path which looked like the following:

```
1700@serverhost:/usr/local/license.dat:1700@shost2
```

or, if the second server was a set of 3 redundant servers, the path might look like this:

```
1700@serverhost:1700@host1,1700@host2,1700@host3:1700@shost2
```

In this last example, the "1700@host1,1700@host2,1700@host3" part specifies a set of 3 redundant servers.

Prior to FLEXlm v2.61, client and server could read different license files, however, after v2.61, both the client and server need to be reading the SAME license file, since the client passes the encryption code from the FEATURE line to the vendor daemon.

license.opt

NAME

license.opt

Site administrator options file for FLEXlm licensed applications

SYNOPSIS

DAEMON *name path* *path/license.opt*

DESCRIPTION

The license.opt file contains optional information supplied by the system administrator at the end-user site. This information can be used to tailor the behavior of the license daemons. The options file can contain the following information:

- Reserved license information
- Logfile control options
- License timeout control
- License access control

Lines beginning with an pound sign (#) are ignored, and can be used as comments. In version 4.0, the maximum length for a line is 2048 characters with backslash '\'line continuation character supported.

There is no default location or name for the options file; it is only active if it has been specified in the license.dat file as the fourth argument on the DAEMON line.

Note that if there are multiple DAEMON lines in the license.dat file, then there can be multiple options files, one for each DAEMON line. Not all of the lines in an options file refer to a feature, so the site administrator MUST set up separate options files in order to use the NOLOG feature.

Each line in the options file starts with a keyword which identifies the information on that line. The keyword is one of EXCLUDE, EXCLUDEALL, GROUP, INCLUDE, INCLUDEALL, NOLOG, RESERVE, or TIMEOUT.

The RESERVE line is of the form:

RESERVE *numlic feature* {USER|HOST|DISPLAY|GROUP} n*ame*

The RESERVE command reserves the specified number of licenses for the specified user, host, display, or group. Note that reserving a license decreases the number of generally available licenses.

The NOLOG line is of the form:

NOLOG {IN|OUT|DENIED|QUEUED}

NOLOG causes messages of the specified type to be filtered out of the daemon's log file. Specifying a NOLOG option will reduce the amount of output to the log file, which can be useful in those cases where the log file grows too quickly.

The GROUP line is of the form:

GROUP *group-name member-list*

The GROUP command is used to define collections of users, which can then be used in RESERVE, INCLUDE, or EXCLUDE commands.

The INCLUDE and EXCLUDE commands are of the form:

{**INCLUDE**|**EXCLUDE**} *feature* {USER|HOST|DISPLAY|GROUP|INTERNET} *name*

INCLUDE and EXCLUDE are used to specify which users (or hosts, displays, groups, or internet addresses) are allowed to use a particular feature. Any user who is EXCLUDEd from a feature will not be able to use that feature. Specifying an INCLUDE line has the effect of excluding everyone else from that feature; thus, only those users specifically INCLUDEd will be able to use that feature.

The INCLUDEALL and EXCLUDEALL commands are of the form:

{**INCLUDEALL**|**EXCLUDEALL**}
{USER|HOST|DISPLAY|GROUP|INTERNET}
name

INCLUDEALL and EXCLUDEALL are used to specify which users (or hosts, displays, groups, or internet addresses) are allowed to use ANY features that this daemon supports. Any user who is EXCLUDEd will not be able to use any of this daemon's features. Specifying an INCLUDE line has the effect of excluding everyone else from all features; thus, only those users specifically INCLUDEd will be able to use the daemon's features.

The INTERNET address is specified in the standard IP address notation, and parts of the address can be wild-carded with a "*", such as:

```
192.9.200.1
```

Or:

```
192.9.200.*
```

For example, the following line would allow only users from the 192.9.200 network to use the features of this daemon:

```
INCLUDEALL INT ERNET 192.9.200.*
```

Any users from machines on another network would not have access to these features. NOTE: INCLUEALL, EXCLUDEALL, and INTERNET are only available in FLEXlm v2.4 or later.

The TIMEOUT line is of the form:

TIMEOUT *feature idletime*

The TIMEOUT command is used to set up a minimum idle time after which a user will lose his license if he is not using it. This allows the site administrator to prevent users from wasting a license (by keeping it checked out when they are not using it) when someone else wants a license.

EXAMPLE

Here is an example of an options file:

```
RESERVE compile USER pat
RESERVE compile USER less
RESERVE compile HOST terry
NOLOG QUEUED
```

FILES

/usr/local/flexlm/options/license.opt

SEE ALSO

lmgrd(1), license.dat(5)

lmcksum

NAME

lmcksum

Checksum the license file.

SYNOPSIS

lmcksum [-c *license_file*] [-k]

DESCRIPTION

lmcksum computes the checksum of the portions of the license file that are not changeable by the end-user. lmcksum computes a checksum for each line in the license file, as well as an overall checksum of the license file. The fields that participate in the checksum are:

- hostid on the SERVER lines.
- daemon name on the DAEMON lines.
- feature name, version, daemon name, expiration date, # of licenses encryption code, vendor string, and hostid on FEATURE lines.
- daemon name and encryption code on FEATURESET lines.

Note

lmcksum is only available in FLEXlm v2.4 and later. lmcksum is either a link or a copy of the lmutil program.

OPTIONS

-c *license_file*

Use the specified license_file. If this switch is not specified, lmcksum looks for the environment variable LM_LICENSE_FILE. If that environment variable is not set, lmcksum looks for the file /usr/local/flexlm/licenses/license.dat. If no -c option is specified, lmcksum looks for the environment variable LM_LICENSE_FILE in order to find the license file to use. If that environment variable is not set, lmcksum looks for the file /usr/local/flexlm/licenses/license.dat.

-k

Case-sensitive checksum. If this switch is specified, lmcksum will compute the checksum using the EXACT case of the FEATURE's and FEATURESET's encryption codes.

SEE ALSO

lmutil(1), license.dat(5).

lmdown

NAME

lmdown

Graceful shutdown of all license daemons

SYNOPSIS

lmdown [-c *license_file*] [-q]

DESCRIPTION

lmdown sends a message to every license daemon asking it to shut down. The license daemons write out their last messages to the log file, close the file, and exit. All licenses which have been given out by those daemons will rescinded, so that the next time a client program goes to verify his license, it will not be valid. NOTE: In FLEXlm v2.4 and later, lmdown is either a link or a copy of the lmutil program. (On VMS, lmdown is still a separate program).

OPTIONS

-c *license_file*

Use the specified license_file. If this switch is not specified, lmdown looks for the environment variable LM_LICENSE_FILE. If that environment variable is not set, lmdown looks for the file /usr/local/flexlm/licenses/license.dat. If no -c option is specified, lmdown looks for the environment variable LM_LICENSE_FILE in order to find the license file to use. If that environment variable is not set, lmdown looks for the file /usr/local/flexlm/licenses/license.dat.

-q

Quiet mode. If this switch is not specified, lmdown asks for confirmation before asking the license daemons to shut down. If this switch is specified, lmdown will not ask for confirmation.

SEE ALSO

lmgrd(1), lmstat(1), lmreread(1), lmutil(1).

lmgrd

NAME

lmgrd

Flexible license manager daemon

SYNOPSIS

lmgrd [-2] [-b] [-c *license_file*] [-d] [-l *logfile*] [-p] [-s *interval*] [-t *timeout*]

DESCRIPTION

lmgrd is the main daemon program for the FLEXlm distributed license management system. When invoked, it looks for a license file containing all required information about vendors and features.

OPTIONS

-2

Specifies V2 startup arguments. This is the opposite of the -b switch. Note that -2 switch is REQUIRED if you intend to use the -p switch. (Available in lmgrd v2.4 and later.)

-b

Specifies backward compatibility mode. Use this switch if you are running a v2.1 or later lmgrd with a v1.5 or earlier vendor daemon. Note that in FLEXlm v2.4 or later the -b switch is the default.

-c license_file

Use the specified license_file. If this switch is not specified, lmgrd looks for the environment variable LM_LICENSE_FILE. If that environment variable is not set, lmgrd looks for the file /usr/local/flexlm/licenses/license.dat.

-d

Specifies that hostnames which are read from the license file should have the local domain name appended to them before sending to a client. This is useful when clients are accessing licenses from another domain. (Available in lmgrd v2.4 and later.)

-l logfile

Specifies the output log file to use.

-p
Specifies that the lmdown and lmremove utilities can only be run by a "license administrator". A "license administrator" is a member of the lmadmin group, or, if the lmadmin group does not exist, then a member of group 0. (Available in lmgrd v2.4 and later.)

-s interval
Specifies the logfile timestamp interval, in minutes. The default is 360 minutes.

-t timeout
Specifies the timeout interval, in seconds, during which daemons must complete their connections to each other. The default value is 10 seconds. A larger value may be desirable if the daemons are being run on busy systems or a very heavily loaded network.

ENVIRONMENT

If no -c option is specified, lmgrd looks for the environment variable LM_LICENSE_FILE in order to find the license file to use. If that environment variable is not set, lmgrd looks for the file /usr/local/flexlm/licenses/license.dat.

SEE ALSO

lmdown(1), lmstat(1)

lmhostid

NAME

lmhostid
Report the hostid of a system

SYNOPSIS

lmhostid [*ether*]

DESCRIPTION

lmhostid calls the FLEXlm version of gethostid and displays the results.


Note
In FLEXlm v2.4 and later, lmhostid is either a link or a copy of the lmutil program.

The output of lmhostid looks like this:

```
lmhostid - Copyright (C) 1989, 1990 Highland Software, Inc. The FLEXlm
host ID of this machine is "1200abcd"
```

OPTIONS

ether (only on HP and NT)
: Specifies that the ethernet hostid should be used rather than the external ID module.

long (on HP and SCO machines)
: On HP, specifies that the HP ID-Module should be used for a hostid. This is a 10 character decimal number, and is used by FLEXlm as a 32-bit long int. On SCO, this specifies that the serial number should have leading characters stripped, and converted to a 32-bit long int.

uname (on HP systems only)
: Returns the machine id from uname -i, and stores the id as a 32-bit long int.

string (on SCO systems only)
: Uses the 'Serial = ' field from uname -X, and stores this as a string, and is the default for SCO.

HOSTID

Here's how to determine the hostid on many support systems.

Architecture Host ID How to Determine

HP
: 32-bit dec hostid uname -i; for example, 2005771344

id module
: read ID typed on module, remove 'A', and convert remainder to hex; for example, 0c127532

ethernet lanscan
: (use station address without leading 0x); such as, 0000F0050185

RS/6000
: 32-bit hostid uname -m, remove last 2 digits, use remaining last 8 digits; for example, 12345678

32-bit hostid
: same, but remove leading characters; for example, 3514

SUN
: 32-bit hostid *hostid*; for example, 170a3472

SEE ALSO

lmutil(1).

lmremove

NAME

lmremove

Remove specific licenses and return them to license pool

SYNOPSIS

lmremove [-c *license_file*] *feature user host display*

DESCRIPTION

lmremove allows the system administrator to remove a single user's license for a specified feature. This could be required in the case where the licensed user was running the software on a node that subsequently crashed. This situation will sometimes cause the license to remain unusable. lmremove will allow the license to return to the pool of available licenses. NOTE: In FLEXlm v2.4 and later, lmremove is either a link or a copy of the lmutil program.

PARAMETERS

feature, user, host, and display describe the user's license which is to be removed. These parameters should be entered exactly as they are displayed by lmstat.

OPTIONS

-c license_file

Use the specified license_file. If this switch is not specified, lmremove looks for the environment variable LM_LICENSE_FILE. If that environment variable is not set, lmremove looks for the file /usr/local/flexlm/licenses/license.dat. If no -c option is specified, lmremove looks for the environment variable LM_LICENSE_FILE in order to find the license file to use. If that environment variable is not set, lmremove looks for the file /usr/local/flexlm/licenses/license.dat.

SEE ALSO

lmstat(1), lmutil(1).

lmreread

NAME

lmreread

Tells the license daemon to reread the license file

SYNOPSIS

lmreread [-c *license_file*]

DESCRIPTION

lmreread allows the system administrator to tell the license daemon to reread the license file. This can be useful if the data in the license file has changed; the new data can be loaded into the license daemon without shutting down and restarting it.

lmreread uses the license file from the command line (or the default file, if none specified) only to find the license daemon to send it the command to reread the license file. The license daemon will always reread the original file that it loaded. If you need to change the path to the license file read by the license daemon, then you must shut down the daemon and restart it with that new license file path.

You can not use lmreread if the SERVER node names or port numbers have been changed in the license file. In this case, you must shut down the daemon and restart it in order for those changes to take effect.

lmreread does not change any option information specified in an options file. If the new license file specifies a different options file, that information is ignored. If you need to reread the options file, you must shut down the daemon and restart it.

Note

In FLEXlm v2.4 and later, lmreread is either a link or a copy of the lmutil program.

OPTIONS

-c license_file

Use the specified license_file. If this switch is not specified, lmreread looks for the environment variable LM_LICENSE_FILE. If that environment variable is not set, lmreread looks for the file /usr/local/flexlm/licenses/license.dat. If no -c option is specified, lmreread looks for the environment variable LM_LICENSE_FILE in order to find the license file to use. If that environment variable is not set, lmreread looks for the file /usr/local/flexlm/licenses/license.dat.

SEE ALSO

lmdown(1), lmutil(1).

lmstat

NAME

lmstat

Report status on license manager daemons and feature usage

SYNOPSIS

lmstat [-a] [-A] [-c *license_file*] [-f [*feature*]] [-l [*regular_expression*]] [-s [*server*]] [-S [*DAEMON*]] [-t *timeout*]

DESCRIPTION

lmstat provides information about the status of the server nodes, vendor daemons, vendor features, and users of each feature. Information can be optionally be qualified by specific server nodes, vendor daemons, or features.

Note

In FLEXlm v2.4 and later, lmstat is either a link or a copy of the lmutil program.

OPTIONS

-a

Display everything.

-A

List all active licenses.

-c *license_file*

Use the specified license_file. If this switch is not specified, lmstat looks for the environment variable LM_LICENSE_FILE. If that environment variable is not set, lmstat looks for the file /usr/local/flexlm/licenses/license.dat.

-f [*feature*]

List all users of the specified feature(s).

-l [*regular_expression*]

List all users of the features matching the given regular expression.

-s [*server*]
Display the status of the specified server node(s).

-S [*daemon*]
List all users of the specified daemon's features.

-t *timeout*
Specifies the timeout interval, in seconds, during which daemons must complete their connections to each other. The default value is 10 seconds. A larger value may be desirable if the daemons are being run on busy systems or a very heavily loaded network.

ENVIRONMENT

If no -c option is specified, lmstat looks for the environment variable LM_LICENSE_FILE in order to find the license file to use. If that environment variable is not set, lmstat looks for the file /usr/local/flexlm/licenses/license.dat.

NOTES

lmstat -a is a potentially expensive command. With lots of active users, this call can generate a lot of network activity, and therefore shouldn't be used too often.

SEE ALSO

lmgrd(1), lmutil(1).

lmutil

NAME

lmutil
Generic FLEXlm utility program.

SYNOPSIS

lmutil [-c *license_file*] *utility-program*

DESCRIPTION

lmutil is the "general-purpose" FLEXlm utility program. Normally, end users would not use lmutil directly, they would use the individual utility programs which are either a copy or a link to lmutil.

PARAMETERS

utility-program is one of lmcksum, lmdown, lmhostid, lmremove, lmreread, lmstat, or lmver.

OPTIONS

-c license_file

Use the specified license_file. If this switch is not specified, lmutil looks for the environment variable LM_LICENSE_FILE. If that environment variable is not set, lmutil looks for the file /usr/local/flexlm/licenses/license.dat. If no -c option is specified, lmutil looks for the environment variable LM_LICENSE_FILE in order to find the license file to use. If that environment variable is not set, lmutil looks for the file /usr/local/flexlm/licenses/license.dat.

SEE ALSO

lmcksum(1), lmdown(1), lmhostid(1), lmremove(1), lmreread(1), lmstat(1), lmver(1).

lmver

NAME

lmver

Report the FLEXlm version of a library or binary file

SYNOPSIS

lmver [filename]

DESCRIPTION

lmver scans the contents of a binary or library file for the FLEXlm version string and displays it. If no argument is given, lmver assumes the filename is "lmgr.a" and attempts to find and display the version from that file.

Note

lmver depends on the strings utility program. Some platforms do not have this utility; thus lmver will not work. In FLEXlm v2.4 and later, lmver is either a link or a copy of the lmutil program.

OPTIONS

lmver has no command line options.

SEE ALSO

lmutil(1).

A
Glossary

Introduction

Following are definitions of some terms that have special meaning in the context of using the SmartModel Library.

Configuration. A platform-specific set of SmartModel Library models and user-versioned tools, with one version number specified for each.

Configuration (LMC) File. A file that contains a configuration—that is, a platform-specific set of SmartModel Library models and user-versioned tools, with one version number specified for each.

Custom Configuration. A user-specified, platform-specific set of SmartModel Library models and user-versioned tools, with one version number specified for each. Overrides model and tool versions specified in the Default Configuration.

Custom Configuration (LMC) File. A file that contains a custom configuration—that is, a user-specified, platform-specific set of SmartModel Library models and user-versioned tools, with one version number specified for each.

Datasheet. A document that describes a model in the SmartModel Library, its sources, supported hardware components and devices, programming, use, timing parameters, and any differences between the model and the corresponding hardware part.

Default Configuration. A system-specified, platform-specific set of SmartModel Library models and user-versioned tools, with one version number specified for each; used if no other configuration exists.

Default Configuration (LMC) File. The file that contains the default configuration—that is, a system-specified, platform-specific set of SmartModel Library models and user-versioned tools, with one version number specified for each. This file is supplied with the SmartModel Library.

Dialog Box. A window used for exchanging information between the user and the application.

Filter. A function that outputs a limited set of data from a larger database, based on specified criteria.

Install Directory. The directory in which the model library is installed at your site; usually referenced by the LMC_HOME environment variable. See your system administrator for the path name to that directory.

LD_LIBRARY_PATH Environment Variable. For Sun operating systems only. An environment variable that contains the path to Sun libraries that are to be executed.

LMC_COMMAND Environment Variable. An environment variable that contains a semicolon-separated list of session commands to be used when simulating.

LMC_CONFIG Environment Variable. An environment variable that contains a colon-separated list of paths to user-specified configuration (LMC) files. For NT this list of paths must be separated by semicolons.

LMC_HOME Environment Variable. An environment variable that contains the path to the SmartModel installation tree, which contains the system-specified default configuration (LMC) file provided with the software. You must set your local LMC_HOME variable with the appropriate path before using the Browser tool.

LMC_PATH Environment Variable. An environment variable that contains a colon-separated list of paths to user-specified timing version files. For NT this list of paths must be separated by semicolons.

LM_LICENSE_FILE Environment Variable. An environment variable that contains the path to a FLEXlm license file. For NT this list of paths must be separated by semicolons.

LMC or .lmc file. A configuration file. "LMC" stands for "List of Model Configurations".

Model. A software representation of a standard integrated circuit.

Model Name. A string of alphanumeric characters that identifies a specific model in the SmartModel Library (for example, am2168, dflipflop, or i80c31).

Model Report. One of the outputs of the Browser tool. Three different reports can be generated.

Model Version. A string of numbers that identifies a specific version of a model in the SmartModel Library (for example, 01000, 01003, or 01012).

Model-versioned Tool. A tool whose version number is specified by the model and cannot be changed by the user. Examples of model-versioned tools include compile_pcl and compile_timing.

Platform. The workstation on which the SmartModel Library is to be installed (for example, hp700, sunSunOS, decalpha, or pcnt).

Predefined Window Element. A window element created by Synopsys and supplied with a specific model.

SmartModel Library. A collection of behavioral simulation models of standard integrated circuits designed to be used in EDA simulation environments that use the SWIFT interface.

.td file. A source timing version file.

.tf file. A timing version file that has been compiled and is ready for simulation.

Timing File. A file that contains timing parameters for a SmartModel.

Timing Version. A SmartModel representation that specifies model timing parameters. Each timing version has a unique name.

Tool Bar. Part of a graphical user interface (GUI), usually either across the top or down one side of an application window, that contains a set of icons, each representing a function of the application.

User-defined Window Element. A window element created by a user (currently available only for FPGA models).

User-versioned Tool. A tool whose version number can be specified by the user, usually by placing it and its version number in a configuration file. Examples of user-versioned tools include ptm_make, mi_trans, and swiftcheck.

Window. A view through which you can access one or more of a model's internal registers.

Window Element. A window with a specific name, created to monitor a specific register or memory array.

Index

E

F

G

I

L

M

N

O

P

R

S

T

U

V

W

Z