

# MaxCompiler Install Guide

Version 2014.1



# Contents

<b>Contents</b>	<b>1</b>
<b>1 Introduction</b>	<b>2</b>
<b>2 Third-party Software</b>	<b>3</b>
<b>3 Installation</b>	<b>5</b>
3.1 Install Development Tools	5
3.2 Install the Oracle Java Development Kit (JDK)	5
3.3 Install Apache Ant	5
3.4 Install Xilinx ISE	5
3.4.1 Xilinx ISE Post-Installation	7
3.5 Install Altera Quartus	7
3.5.1 Altera Quartus Post-Installation	9
3.6 Install Quartus and ISE licenses	9
3.6.1 Floating Licences	9
3.6.2 Node-locked Licences	10
3.7 Install MaxCompiler	10
3.8 Install MaxCompiler License File	11
3.9 Install MaxelerOS	11
3.10 Install GraphViz (optional)	11
3.11 Install ModelSim (optional)	12
3.11.1 Install Ncurses	12
3.11.2 Acquire ModelSim Files	12
3.11.3 Compile ModelSim Xilinx Libraries (MAX2 and Vectis users)	13
3.11.4 Compile ModelSim Altera Libraries (Coria, Maia and Isca users)	20
<b>4 Testing a MaxCompiler Installation</b>	<b>21</b>
4.1 Changing Board Models	21
4.2 Building Tutorial Examples	21
4.3 Testing ModelSim	21
<b>5 Optional Setup</b>	<b>22</b>
5.1 Build Configuration File	22
5.2 Build Paths	22
5.3 Locating the Build Directory	22
5.4 Core Cache	22
<b>6 Uninstalling</b>	<b>24</b>

## 1 Introduction

In this document we provide a brief overview of installing MaxCompiler and software it relies upon.

---

## 2 Third-party Software

During compilation, MaxCompiler uses a number of third-party programs. Below is a list of these programs including versions which are known to work with this release. The presence of these components is verified during the MaxCompiler installation process.

### **CentOS 5/6 or Red Hat Enterprise Linux 5/6**

Linux distributions. CentOS is available for free download at <http://www.centos.org/>.

### **GCC 4.1+ & GNU Make 3.80+**

C language compiler and development tools. Packages for these tools are included with CentOS and Red Hat.

### **Xilinx ISE 13.3 (for MAX2 and Vectis DFEs)**

Xilinx's low-level FPGA compilation tools.

### **Altera Quartus 13.1 (for Coria, Maia and Isca DFEs)**

Altera's low-level FPGA compilation tools.

### **Oracle Java SE Development Kit 1.6+**

Java language compiler, development tools and run-time environment.

### **Apache Ant 1.7+**

Java-based build tool (similar to `make`).

### **GraphViz 2.14+ (optional)**

Graph visualization tools. A copy of GraphViz is supplied with MaxCompiler but not automatically installed or checked for during installation.

### **ModelSim DE 10.0a (optional)**

Logic simulation tool relevant only to designs containing custom Hardware Description Language (HDL) blocks, licensed separately by Mentor Graphics.

### **Ncurses (optional)**

Console oriented user interface library needed only by ModelSim.

### **WebKitGTK 1.2.x (recommended)**

HTML rendering library used by MaxIDE.

### **XULRunner 1.9.x (optional)**

Alternative HTML rendering library used by MaxIDE.

### **Python 2.6 and Python header files (optional)**

For MaxCompiler Skins support.

### **Matlab and Mex 2012b (optional)**

For MaxCompiler Skins support.

### **R 2.11 (optional)**

For MaxCompiler Skins support.

## 2 Third-party Software

---

XULRunner is a standard library in RedHat and CentOS distributions and probably does not need to be installed. MaxIDE searches for WebKitGTK and tries to use XULRunner if WebKitGTK is not found or is the wrong version. Older or newer versions of either library than those listed above are incompatible with MaxIDE. The absence of both libraries disables MaxIDE tool tips and causes some differences from the screen shots shown in the tutorial documents.



MaxCompiler is only compatible with 64 bit systems.

## 3 Installation

To set-up an environment for using MaxCompiler, we recommend the following steps, including the installation of necessary third-party software. Each step includes OS environment variables which should be set for MaxCompiler to function properly.

### 3.1 Install Development Tools

First install standard developer tools onto the system by running the following command as root when connected to the Internet.

```
[root@machine ~]# yum groupinstall "Development Tools"
```

### 3.2 Install the Oracle Java Development Kit (JDK)

Download the Java SE development kit from the Oracle web site (<http://goo.gl/EaWKhe>). It is recommended that the file `jdk-6u45-linux-x64-rpm.bin` be downloaded. This can be installed by running these commands as root.

```
[root@machine ~]# chmod +x jdk-6u45-linux-x64-rpm.bin
[root@machine ~]# ./jdk-6u45-linux-x64-rpm.bin
```

JDK Environment Variables	
PATH	Should include The bin/ sub-directory of the Java install

### 3.3 Install Apache Ant

This can be installed by downloading the packages and following the instructions at <http://ant.apache.org>, or more easily if possible by running this command as root.

```
[root@machine ~]# yum install ant.x86_64
```

Apache Ant Environment Variables	
PATH	Should include the bin/ sub-directory of the Apache Ant install

Shared library files for `libgmp` are downloaded and relevant paths are set automatically.

### 3.4 Install Xilinx ISE

Xilinx ISE is required when targeting MAX2 and Vectis DFEs. Be aware that any particular MaxCompiler version is typically only compatible with exactly one version of ISE as indicated in [section 2](#).

ISE is installed using a single installer available from the Xilinx website (<http://www.xilinx.com/support/download/index.htm>) or may be provided by Maxeler.

ISE can use either a node-locked license, which is restricted by MAC address, or a floating license via a FlexLM license server. ISE looks for the license using this environment variable:

ISE Environment Variables	
XILINXD_LICENSE_FILE	license file or license server paths

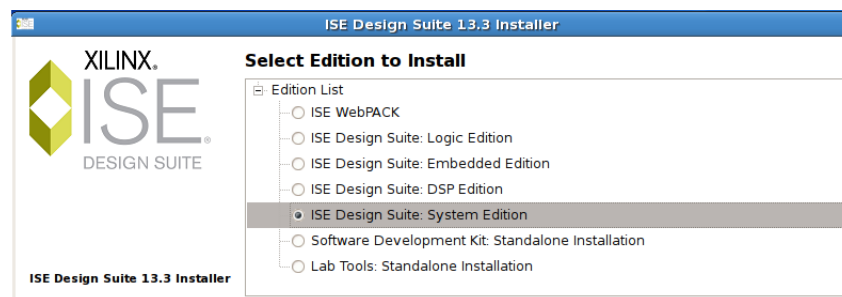
This variable may contain a colon separated list of file paths or network addresses of license servers. Please consult the document *ISE Design Suite 13: Installation and Licensing Guide* for more information, at [http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx13\\_1/iil.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_1/iil.pdf). You should set this environment variable before proceeding so that the Xilinx graphical installation utility detects it.

The Xilinx tools are shipped in a file named `Xilinx_ISE_DS_Lin_13.3_0.76xd.1.0.tar`. After obtaining a copy of this file from the Xilinx web site or installation media, run these commands as root:

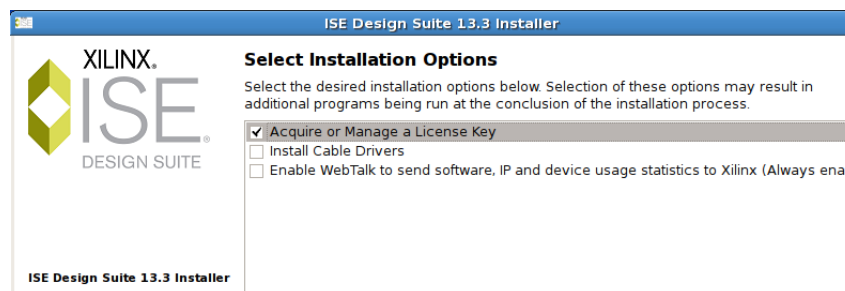
```
[root@machine ~]# tar -xf Xilinx_ISE_DS_Lin_13.3_0.76xd.1.0.tar
[root@machine ~]# cd Xilinx_ISE_DS_Lin_13.3_0.76xd.1.0
[root@machine ~]# ./xsetup
```

`xsetup` launches a graphical interface which goes through several stages. The significant stages are:

**Select Edition to Install:** Select “ISE Design Suite: System Edition”.



**Select Installation Options:** **Acquire or Manage a License Key** should be set on this page, which enables a subsequent dialog confirming that your licenses are detected based on the environment variables you have set, or letting you make any necessary adjustments.



**Select Destination Directory:** Here you can set the directory in which to install ISE. The default is `/opt/Xilinx/13.3`.



Refer to the *Xilinx ISE Design Suite 13: Release Notes* document ([http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx13\\_3/irn.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_3/irn.pdf)) for full details on installation of ISE.

### 3.4.1 Xilinx ISE Post-Installation

Now run the Xilinx wrapper script install script that was distributed with MaxCompiler. This creates wrapper scripts for every Xilinx binary, which set the appropriate environment variables that the Xilinx tool requires.


```
[root@machine ~]# bash ./xilinx_wrapper_install.sh
```

The install script assumes that you installed Xilinx ISE to the default location. If you installed Xilinx ISE to an alternative location, you need to pass this as a parameter:

```
[root@machine ~]# bash ./xilinx_wrapper_install.sh /path/to/xilinx/ISE_DS
```

This script adds a file in `/etc/profile.d` which adds the directory containing the wrapper scripts to the default search path. You can manually add the directory to the path by running

```
[root@machine ~]# bash ./xilinx_wrapper_install.sh /path/to/xilinx/ISE_DS
```

 If you source `/opt/Xilinx/13.3/ISE_DS/settings64.sh` manually or in `.bashrc`, then the changes it makes to the `LD_LIBRARY_PATH` environment variable break many other programs under some Linux distributions.

## 3.5 Install Altera Quartus

Altera Quartus is required when targeting Coria, Maia and Isca DFEs. Be aware that any particular MaxCompiler version is typically only compatible with exactly one version of Quartus as indicated in [section 2](#).

Altera Quartus is installed using an installer available from the Altera website (<https://www.altera.com/download/software/quartus-ii-se>). Ensure the drop-down box shows version '13.1' and select *Linux* under *Operating System*. Download and unpack *Quartus-13.1.0.162-linux-complete.tar*.

You will also require a license file to use the Quartus tools. Quartus looks for the license using this environment variable:

Quartus Environment Variables	
LM_LICENSE_FILE	license file or license server paths

This variable may contain a colon (:) separated list of file paths or network addresses of license servers.

Before running the Quartus installation tool on CentOS 6 it is necessary to install 32bit versions of some software by running the command below.

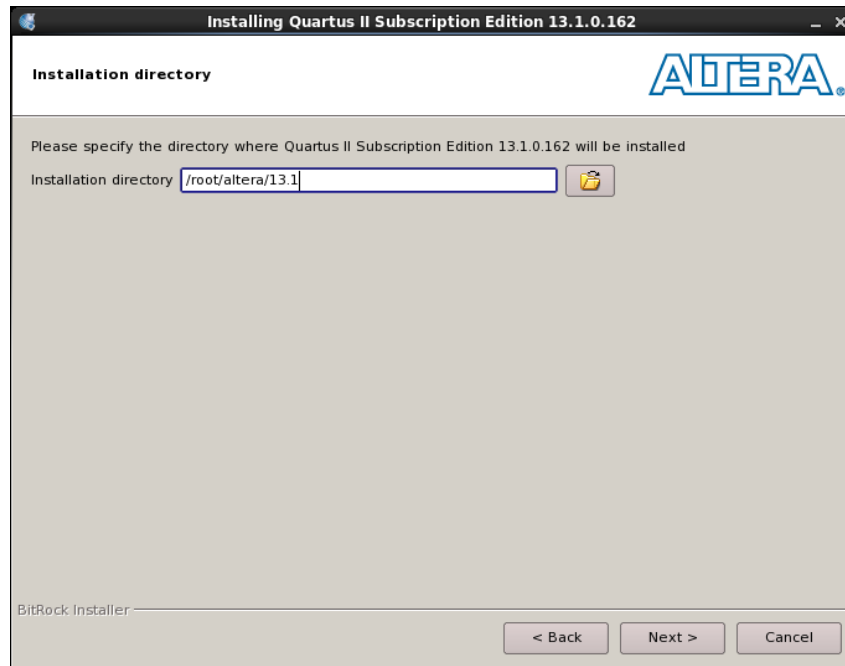
```
[root@machine ~]# yum install compat-libstdc++-33.i686 expat.i686 fontconfig.i686 freetype.i686 glibc.i686 gtk2.i686 libcanberra-gtk2.i686 gtk2-engines-2.18.4-5.el6.centos.i686 libpng.i686 libICE.i686 libSM.i686 libuuid.i686 ncurses-devel.i686 ncurses-libs.i686 PackageKit-gtk-module.i686 tcldevel.i686 tcl.i686 zlib.i686 libX11.i686 libXau.i686 libXdmcp.i686 libXext.i686 libXft-devel.i686 libXft.i686 libXrender.i686 libXt.i686 libXtst.i686
```



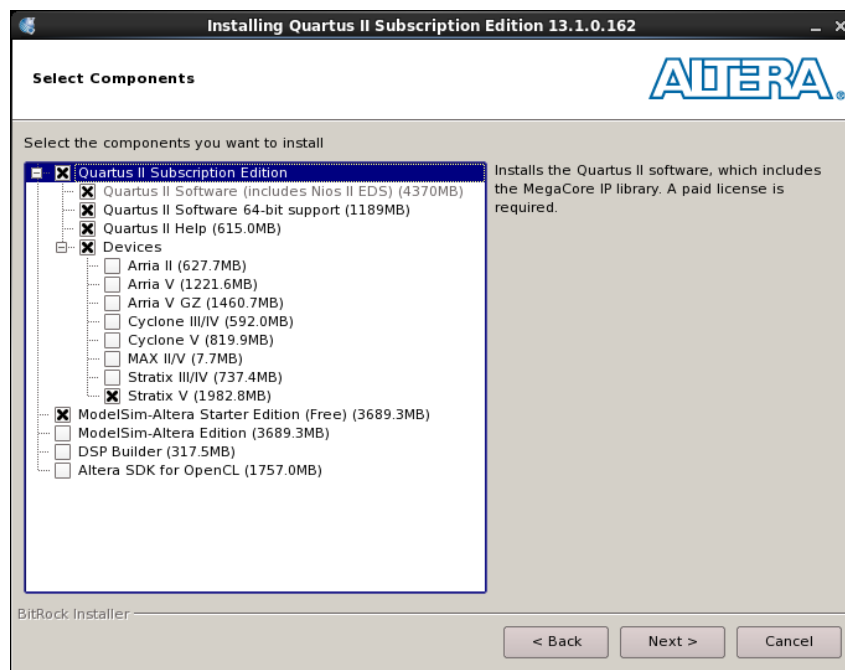
### 3 Installation

When running the Altera installation tool there are several stages to the GUI installation. The significant stages are:

**Select Destination Directory:** Here you can specify the directory where Quartus will be installed. You must remember this location when configuring environment variables after installation.



**Select Products:** Here you can select which Quartus components will be installed. Ensure that Stratix V device support is included.



### 3.5.1 Altera Quartus Post-Installation

After installation of the Quartus tools you must additionally configure the PATH environment variable to begin using the tools.

Quartus Environment Variables	
PATH	Should include the quartus/bin directory inside the Quartus installation directory

Quartus does not work correctly when SELinux is enabled and gives the error  
 error while loading shared libraries: <path to quartus/bin>/libcd\_err.so:  
 cannot restore segment prot after reloc: Permission denied  
 To avoid these errors, disable SELinux enforcement by editing /etc/selinux/config and changing the line  
 SELINUX=enforcing  
 to  
 SELINUX=permissive  
 and then restart the machine.

### 3.6 Install Quartus and ISE licenses

Licenses are required to use Altera and Xilinx software and can be obtained directly from the vendors. These licenses can be floating licenses or node-locked licenses.

#### 3.6.1 Floating Licences

The license should be tied to a specific hostname and MAC address. To install a floating license server and connect to it follow these steps :-

1. Download and install the license server from <https://www.altera.com/download/sw/dnl-sw-index.jsp> or <http://www.xilinx.com/support/download/index.htm>.
2. Get the hostname and MAC address and port, if given, from the line in the license file starting *SERVER*.
3. Modify any lines starting *VENDOR alterad* or *VENDOR xilinxd* so that any paths following them point at the following files in your installation.

Line	Location
VENDOR alterad	QUARTUS_INSTALLATION/linux64/alterad
VENDOR xilinxd	ISE_INSTALLATION/ISE_DS/ISE/bin/linux64/xilinxd

4. Create a directory on the license server to contain the license files and move them there.
5. Specify the license to the license manager by running `lmgrd -c <LICENSE_FILE_PATH>`. This spawns a daemon.
6. Verify that the server is running by running `lmutil lmstat -a -c <port@hostname>` where port and hostname correspond to the service on the license server.
7. To use the license server from a machine with MaxCompiler installed set the environment variable `LM_LICENSE_FILE` to `port@hostname` for the port and hostname of the license server.

### 3.6.2 Node-locked Licences

When given a node-locked license file the environment variable `LM_LICENSE_FILE` can simply contain the absolute path of the license file.



Further information can be found on the Xilinx and Altera websites.

## 3.7 Install MaxCompiler

An installation of MaxCompiler requires both the MaxCompiler software itself and a valid MaxCompiler license file.

First, unpack the MaxCompiler install archive file:

```
[root@machine ~]# tar xzvf maxcompiler-2014.1-full-installer.tar.gz
```

The MaxCompiler software install is automated by the `install` program provided in the MaxCompiler package. This should be launched with the desired installation directory specified on the command line. It is recommended that you do this as the root user on your system.

Additionally, you must specify which edition you are using to build DFE designs with the `--edition` option. This must be one of the following:

- `s`, supporting simulation only,
- `v`, supporting Vectis, Vectis-Lite and MAX2 DFEs,
- `a`, supporting Coria, Maia and Isca DFEs,
- `c`, supporting all DFEs (complete).

The installation script verifies the presence of the required software components mentioned above and fails if this software is not found. However, if you wish to ignore missing software components, use the `--ignore-missing` option when running the script.

For example, use the following command to install MaxCompiler to the directory `/opt/maxcompiler` for Vectis, Coria and Maia DFEs:

```
[root@machine maxcompiler-2014.1-installer]# ./install --edition c /opt/maxcompiler
```

```
This program is about to install MaxCompiler 2014.1
```

```
To proceed you must agree to the terms and conditions of the licenses which cover this software. These licenses can be found in:
```

```
/home/user/maxcompiler-2014.1-installer/licensing
```

```
An overview of these licenses is provided in licenses_overview.txt which is also in this directory.
```

```
Enter 'yes' to confirm you agree to these licenses or 'no' to abort: yes
```

```
Checking for required applications...
```

```
Installing to /opt/maxcompiler...
```

### 3 Installation

```
-- INSTALLATION COMPLETE! --
Please set the following environment variables:
MAXCOMPILERDIR=/opt/maxcompiler
PATH=/opt/maxcompiler/bin:$PATH
```

Alternatively, you can use the scripts in MAXCOMPILERDIR.  
For BASH:

```
source /opt/maxcompiler/settings.sh
```

For CSH / TCSH:

```
source /opt/maxcompiler/settings.csh
```

The console output from the installer gives the PATH and MAXCOMPILERDIR environment variables that must be set for MaxCompiler to function correctly.

These can either be set manually or using the `settings.sh` (for BASH) or `settings.csh` (for csh) scripts located in the MaxCompiler install directory:

```
[root@machine ~]# source /opt/maxcompiler/settings.sh
```

#### 3.8 Install MaxCompiler License File

In addition to the MaxCompiler software, a license file is also required to use MaxCompiler. Valid licenses are provided by Maxeler and by default these should be placed in the MaxCompiler install directory in the `licenses` sub-directory. Multiple license files may be placed in this directory, all of which are scanned for license authorization provided they are named with a `.lic` extension.

If necessary, MaxCompiler licenses may be placed in an alternative directory specified with the `MAXLICENSEDIR` environment variable.

MaxCompiler Environment Variables	
PATH	Should include the <code>bin/</code> sub-directory of the MaxCompiler install root
MAXCOMPILERDIR	Set to point to the root of the MaxCompiler install directory
MAXLICENSEDIR	Optionally specifies an alternative license file search-directory

#### 3.9 Install MaxelerOS

Documentation for installing MaxelerOS is provided with the MaxelerOS install package.

The `MAXELEROSDIR` environment variable must be set for CPU code compilation to function:

MaxelerOS Environment Variables	
MAXELEROSDIR	Set to point to the root of the MaxelerOS directory

#### 3.10 Install GraphViz (optional)

During compilation MaxCompiler outputs several data files detailing parts of compilation process. These files can be visualized using the GraphViz tool set and can be useful when debugging designs.

RPMs for GraphViz are distributed with MaxCompiler in the `extras/` directory of the installation tar-ball. These can be installed as follows.

```
[root@machine maxcompiler-2014.1-installer]# rpm -ivh extras/graphviz-2.24.0-1.e15.x86_64.rpm
[root@machine maxcompiler-2014.1-installer]# rpm -ivh extras/graphviz-gd-2.24.0-1.e15.x86_64.rpm
```

ModelSim Environment Variables	
LM_LICENSE_FILE	license servers or files
MTI_VCO_MODE	always set to 32
MODELSIMDIR	the installation directory, /opt/modelsimde-10.0a
MODELSIM	the configuration file, \$MODELSIMDIR/modelsim_dlx/modelsim.ini
PATH	\$MODELSIMDIR/modelsim_dlx/linuxpe:\$PATH

Table 1: The path setting shows how the ModelSim executable files can be added to an existing path.

Note that you must have root permissions to perform this install. Also, as installation is via RPM, installation cannot be made to a network share. As such, GraphViz should be installed on all individual computers on which users may wish to run the GraphViz tools.

### 3.11 Install ModelSim (optional)

ModelSim is a third party hardware simulation tool published by Mentor Graphics, which can be used in conjunction with MaxCompiler. It is beneficial mainly to users whose applications incorporate custom Hardware Description Language (HDL) blocks. Custom HDL blocks are needed only in unusual circumstances. Most MaxCompiler users do not need them and may safely ignore the remainder of this section.

If you intend to use ModelSim with MaxCompiler, obtain a license from Mentor Graphics and set the environment variables shown in [Table 1](#). The license environment variable should contain a colon separated list of license servers or license file paths, while the mode variable enables a 32 bit version of the tool to run on a 64 bit system.

#### 3.11.1 Install Ncurses

ModelSim needs the 32 bit version of `ncurses-libs`, which is a free library for console interfaces. To install it, run the following command as root.

```
[root@machine ~]# yum install ncurses-libs.i686
```

#### 3.11.2 Acquire ModelSim Files

MaxCompiler supports Version 10.0a of ModelSimDE. You may obtain a copy of the relevant executable files either from the installation media or from <http://model.com/content/modelsim-downloads>, taking care to navigate to the archive for the required version, which should contain these files:

```
install.linux
INSTALL_NOTES
INSTALL_NOTES.txt
mgc_licen.pdf
modelsim_dlx_10.0_install.pdf
modelsim_dlx-base.mis
modelsim_dlx-docs.mis
modelsim_dlx-gcc-4.2.1-mingw32vc9.zip
modelsim_dlx-gcc-linux.mis
modelsim_dlx-linuxpe.mis
```

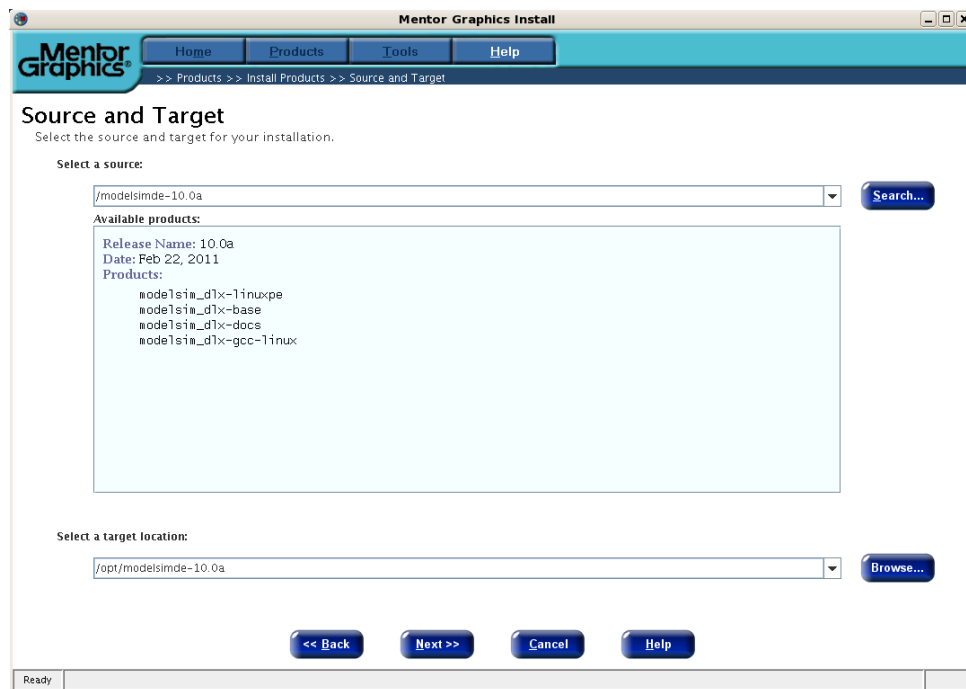


Figure 1: ModelSim graphical installer screen shot, showing target location `/opt/modelsimde-10.0a`; source location may differ

```
modelsim_dlx-win32-10.0a-de.exe
RELEASE_NOTES
RELEASE_NOTES.html
RELEASE_NOTES.txt
```

To install ModelSim, create a new temporary directory named `modelsimde-10.0a`, copy these files into it, change to that directory, and execute `install.linux`, which launches a graphical installer. It may be necessary to enable execute permission for this file as shown.

```
[root@machine modelsimde-10.0a]# chmod ugo+x install.linux
[root@machine modelsimde-10.0a]# ./install.linux
```

Select `/opt/modelsimde-10.0a` for a target location on the screen shown in [Figure 1](#), and select all packages on the screen shown in [Figure 2](#).

### 3.11.3 Compile ModelSim Xilinx Libraries (MAX2 and Vectis users)

When using MAX2 and Vectis DFEs, the next step is to run the Xilinx tool `compplib`, which enables Xilinx and ModelSim software to work together. Before running `compplib`, be sure to set the environment variables shown in [Table 1](#), so that ModelSim is detected.

Normally, `compplib` allows ModelSim DE as a simulation target only with the 32 bit version of ISE. However, MaxCompiler requires a 64 bit version of ISE. Our Xilinx wrapper scripts deal with this automatically, by sourcing `settings32.sh`, found under `/opt/Xilinx/13.3/ISE_DS/` when `compplib` is called.

Running `compplib` as shown with no parameters should launch a graphical configuration utility.

```
[root@machine modelsimde-10.0a]# /opt/Xilinx/13.3/ISE_DS/ISE/bin/lin/compplib
```

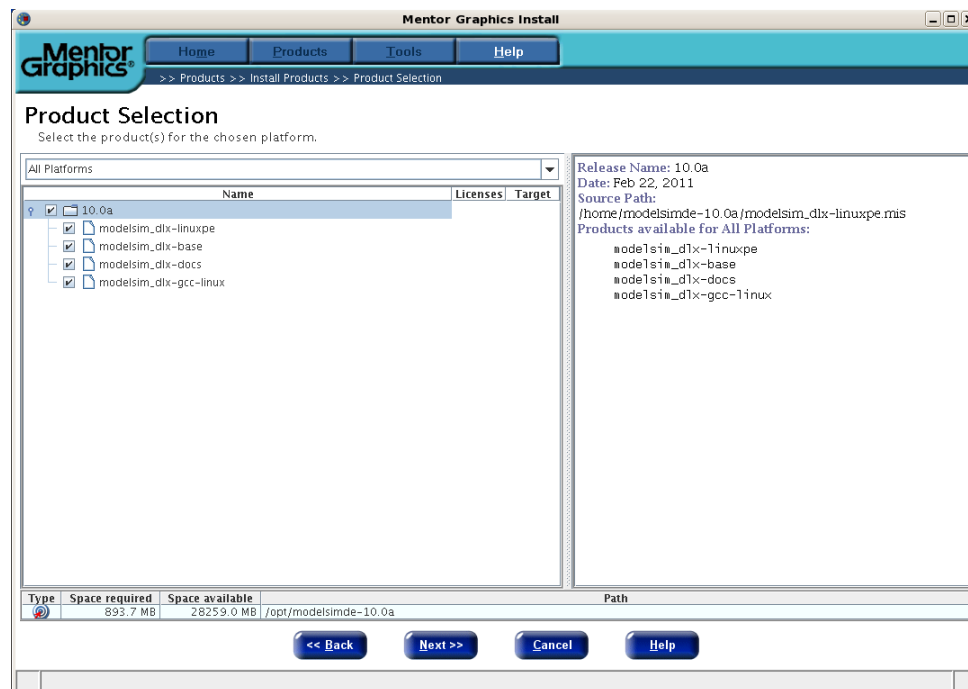


Figure 2: ModelSim graphical installer screen shot, showing all packages selected for installation; source path may differ

If the utility does not launch, it may be due to your SELinux settings, which you can change using this command.

```
[root@machine modelsimde-10.0a]# find /opt/Xilinx/13.3/ISE_DS/ISE/lib/lin -name *.so | xargs chcon -t textrel_shlib_t
```

The `compxlib` utility presents a sequence of dialogs allowing you to select various settings. The selections shown in [Figure 3](#) through [Figure 7](#) are needed for compatibility with MaxCompiler.

- In the simulator configuration dialog shown in [Figure 3](#), select the “ModelSim DE” radio button, and let other fields be filled by default. If the ModelSim DE radio button is disabled, check that you have temporarily enabled 32 bit settings as explained above.
- In the HDL configuration dialog shown in [Figure 4](#) select the radio button for “Both VHDL and Verilog”.
- In the device families configuration dialog shown in [Figure 5](#), select only “Virtex 5” and “Virtex 6” for use with MaxCompiler. (There is no harm in selecting the full set of device families for other uses but their installation may take several days to complete.)
- In the simulation library configuration dialog shown in [Figure 6](#), select UNISIM, CORE, and SIM-PRIM.
- In the output directory configuration dialog shown in [Figure 7](#) keep the default settings and click on the “Launch Compile Process” button.

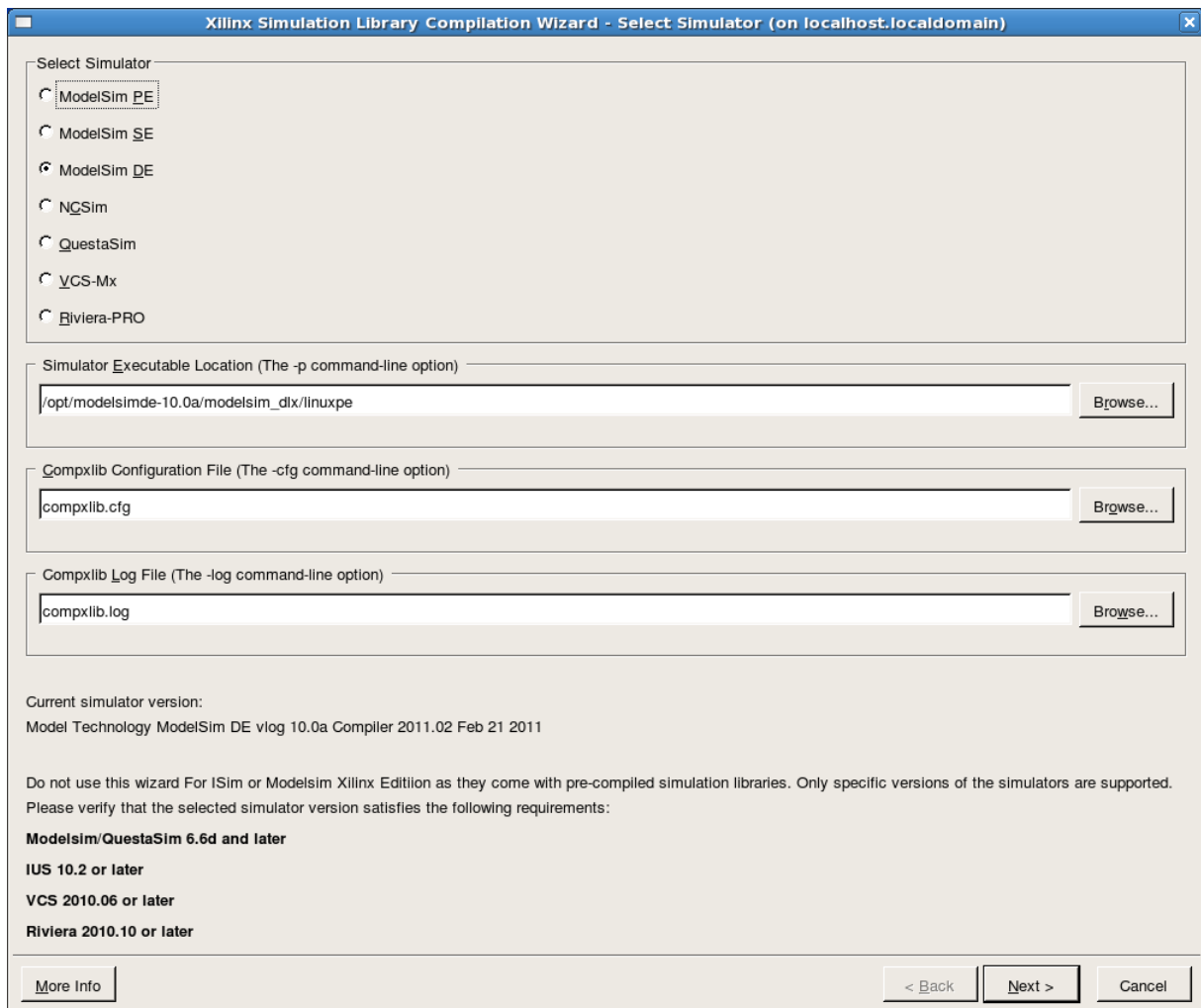


Figure 3: `complib` simulator configuration dialog showing “ModelSim DE” selected, with other fields filled by default



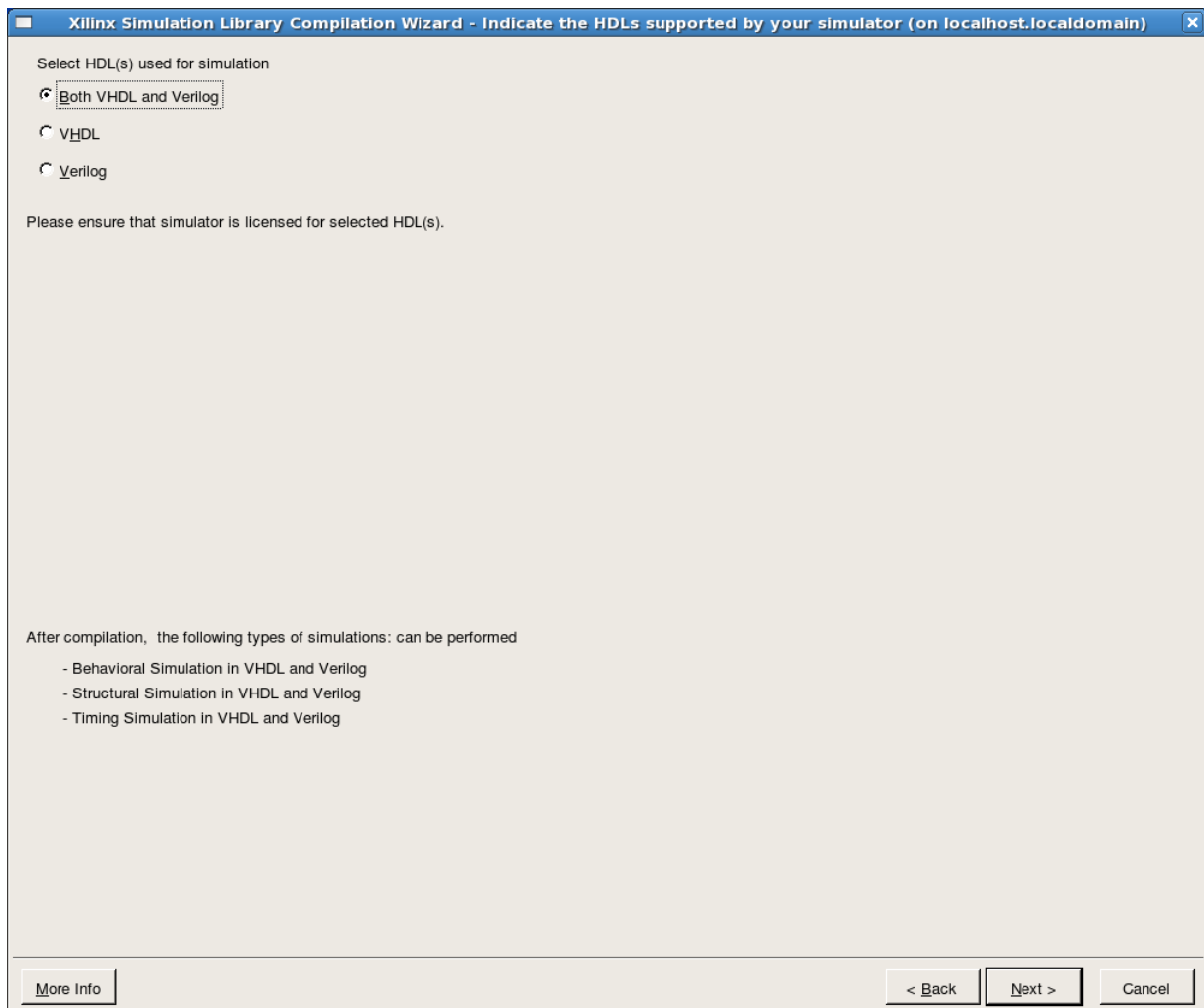


Figure 4: compxlib HDL configuration dialog showing “Both VHDL and Verilog” selected

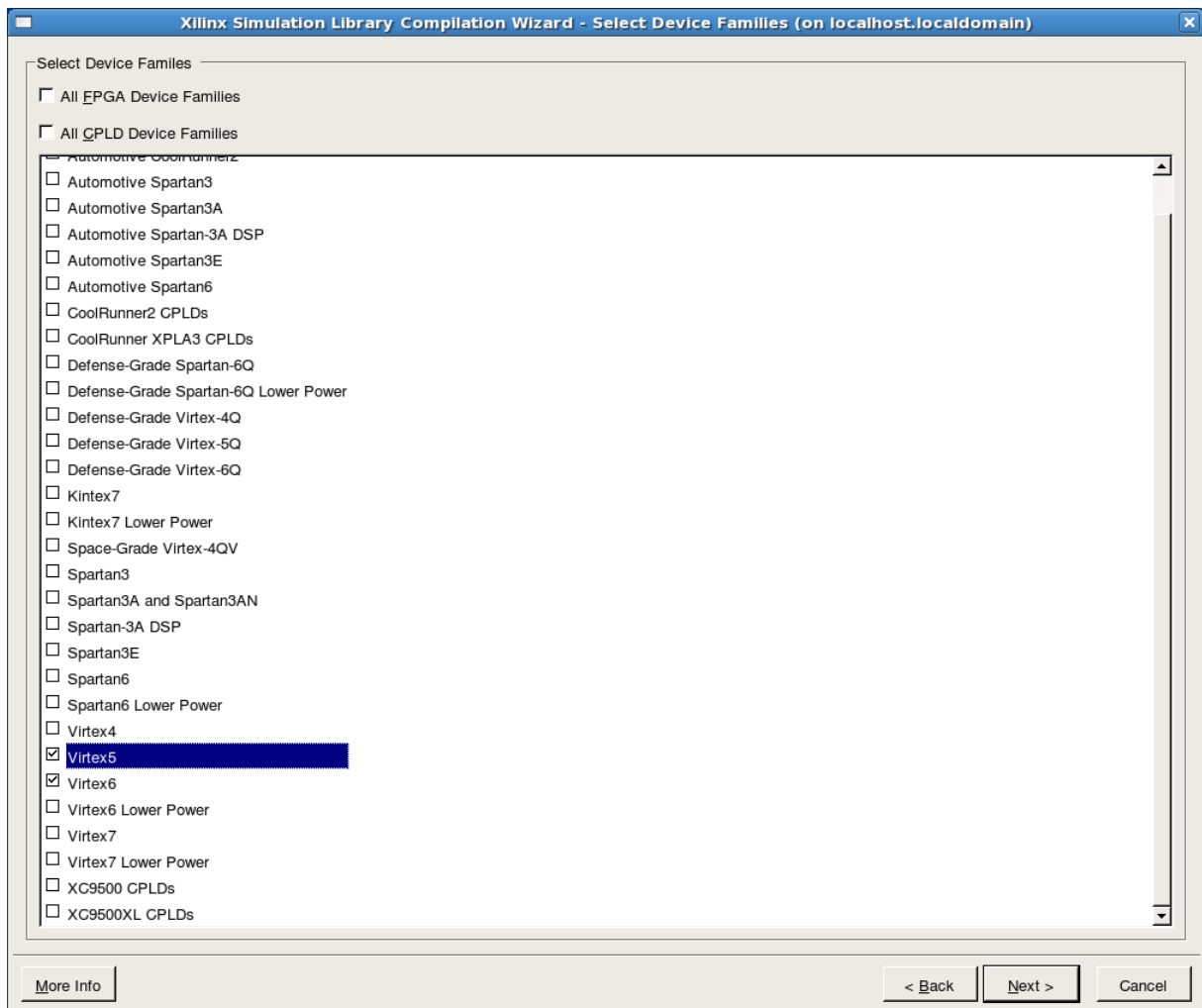


Figure 5: comp1ib device families configuration dialog, showing “Virtex 5” and “Virtex 6” selected

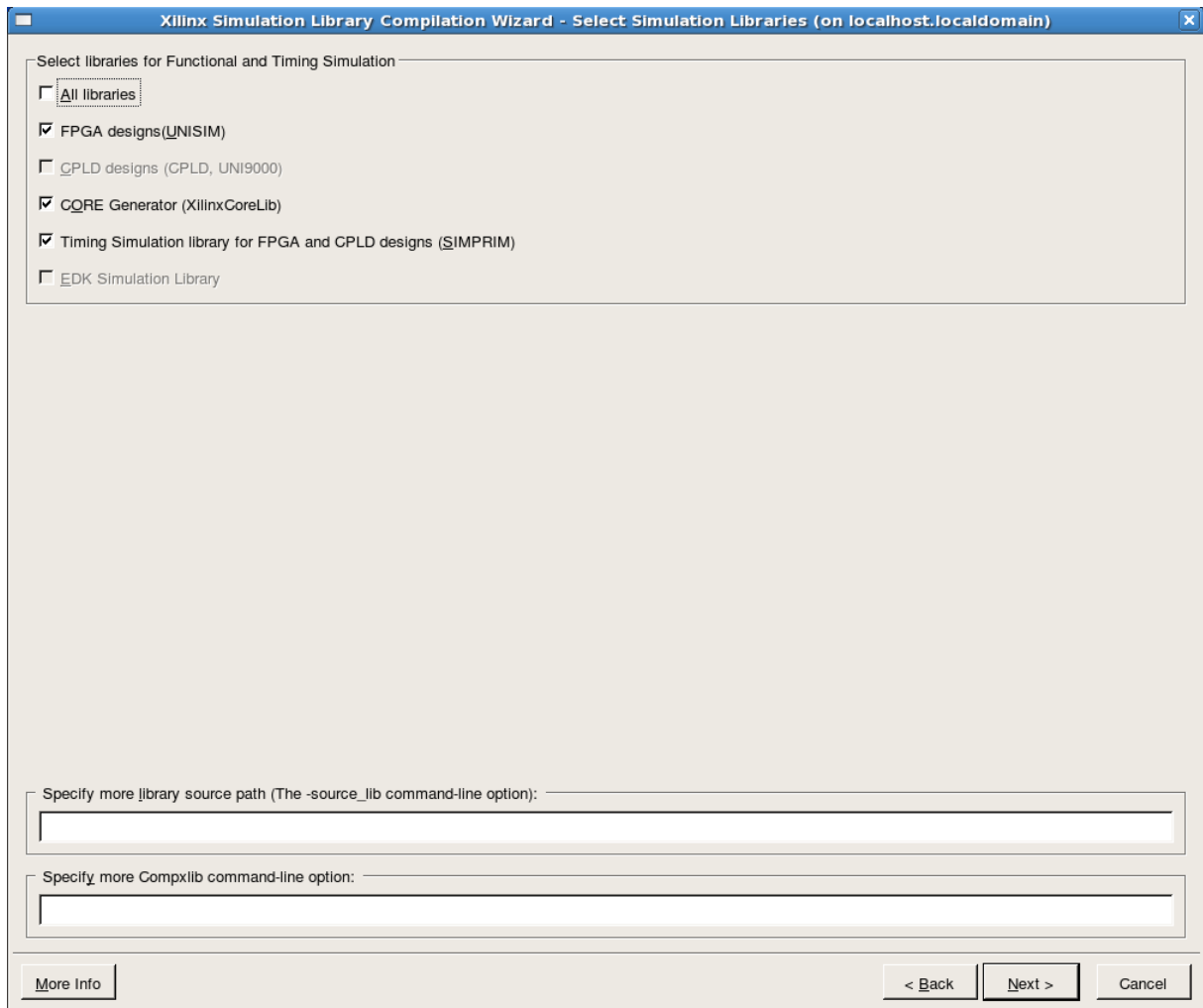


Figure 6: compxlib simulation library configuration dialog, showing UNISIM, CORE, and SIMPRIM selected

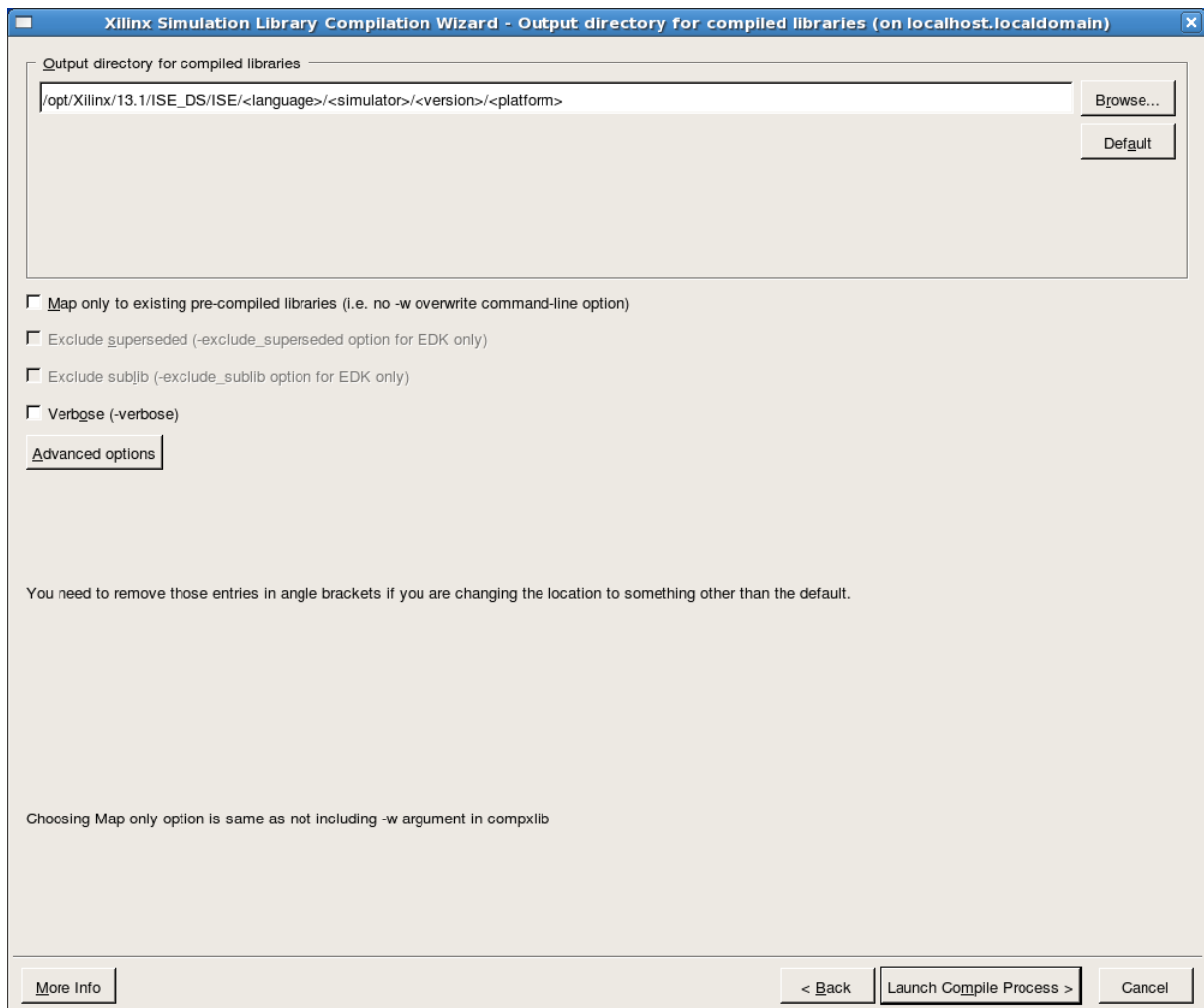


Figure 7: compxlib output directory configuration dialog with default settings

---

#### 3.11.4 Compile ModelSim Altera Libraries (Coria, Maia and Isca users)

When using Coria, Maia and Isca DFEs, the next step is to run a tool shipped with Quartus tool to enable ModelSim simulations of Altera devices. Before running this, Quartus must already be fully installed and configured as well as the environment variables shown in [Table 1](#). Once this is done, execute the following command in a Bash shell:

```
quartus_sh --simlib_comp -tool modelsim -language vhdl -tool_path "$(dirname $(which vsim))" -directory "$MAXCOMPILERDIR/quartus_vhdl/" -rtl_only
```

This will compile Quartus simulation models into the MaxCompiler directory and update the `modelsim.ini` file pointed to by the `MODELSIM` environment variable so ModelSim can find them during operation.

## 4 Testing a MaxCompiler Installation

After MaxCompiler is fully installed, it can be tested by building one of the tutorial examples provided.

**X** If MaxCompiler is installed by the root user, non-root users can not build or modify the tutorial examples in the \$MAXCOMPILERDIR directory. As a non-root user, copy \$MAXCOMPILERDIR/examples to a location writable by you (e.g., your \$HOME directory).

### 4.1 Changing Board Models

The tutorial examples are configured to build hardware and simulation models for a Vectis DFE. If a different Maxeler DFE model is installed, execute the following commands from your \$HOME/examples directory, substituting your DFE model for *nnnnn* using capital letters.

```
find . -name RunRules.settings | xargs sed -i s/VECTIS/nnnnn/
find . -name Makefile.include | xargs sed -i s/VECTIS/nnnnn/
```

These commands update the model for all examples, exercises, and solutions in all tutorial documents. Editing these files manually is not recommended, but you may change the model for each tutorial example individually using MaxIDE.

### 4.2 Building Tutorial Examples

To build and run a simulation example, perform the following steps (assuming the tutorial examples have been copied from \$MAXCOMPILERDIR to your \$HOME directory):

```
[user@host ~]$ cd $HOME/examples/maxcompiler-tutorial/examples
[user@host examples]$ cd tutorial-chap03-example1-movingaveragesimple/RunRules/
Simulation
[user@host Simulation]$ make runsim
```

To build and run a DFE example:

```
[user@host ~]$ cd $HOME/examples/maxcompiler-tutorial/examples
[user@host examples]$ cd tutorial-chap03-example1-movingaveragesimple/RunRules/
DFE
[user@host DFE]$ make run
```

### 4.3 Testing ModelSim

If you have installed ModelSim ([subsection 3.11](#)), you can test it by simulating one of the custom HDL examples from Chapter 7 of the MaxCompiler Manager tutorial.

```
[user@host ~]$ cd $HOME/examples/maxcompiler-manager-tutorial/examples
[user@host examples]$ cd manager-chap07-example01-simplehdl/EngineCode
[user@host EngineCode]$ ant -f SimpleHDLsimRunner_run.xml
```

The simulation run by the ant utility automatically starts a non-interactive session of ModelSim if it has been installed correctly. This whole process may take about twenty minutes.

## 5 Optional Setup

### 5.1 Build Configuration File

To configure the behavior of MaxCompiler for an individual user, MaxCompiler looks for a configuration file called `.MaxCompiler_build_user.conf` in the user's home directory. An example configuration file `example_maxcompiler_build.conf` is given in the `$MAXCOMPILERDIR/docs/` directory. This file shows common configuration options with descriptions as comments.

### 5.2 Build Paths

You can specify the root directory for all build output by setting `build.root_dir` in `.MaxCompiler_build_user.conf`. You can refer to environment variables in the configuration file using `${ENV_VAR}`, for example `build.root_dir = ${MAXCOMPILER_BUILD_DIR}`.

You can enable date-stamping of build paths to keep past builds by setting `build.datestamp_builds = false` and setting `build.datestamp_builds_format` to a date format, for example `build.datestamp_builds_format = dd-MM-yy`. The date format used is the `SimpleDateFormat` class in the JDK; refer to <http://java.sun.com/javase/6/docs/api/java/text/SimpleDateFormat.html> for more details on `SimpleDateFormat`.

The build output path for a Manager is made up of three components:

`/build.root_dir/datestamp/build_name`

`build_name` is the same as the name given to the Manager.

Therefore, if the build root directory is set to `/home/user/builds`, the date stamp is set to `dd-MM-yy`, today's date is the first of September 2010 and the Manager is called `MyManager`, MaxCompiler uses an output directory of:

`/home/user/builds/01-09-10/MyManager`.

### 5.3 Locating the Build Directory

MaxCompiler provides a tool called `maxGuessBuildDir`, which locates the build directory for an application. If date stamping is enabled, `maxGuessBuildDir` looks for a build from the current day (default) or from the last two days (optionally with `-s`). `maxGuessBuildDir` is used in the makefiles for all of the examples and exercises provided with MaxCompiler.

**X** Dates should be specified with a granularity no finer than a day otherwise `maxGuessBuildDir` will not work correctly.

### 5.4 Core Cache

MaxCompiler can keep a repository of IP blocks that can be shared between projects. Once each block is built and stored in this cache, it does not need to be rebuilt for each subsequent build of the same project or another project that uses the same block with the same configuration. This can dramatically reduce hardware build times.

To set the directory for the core cache, you can set `build.arbitrated_core_cache` to point to a directory for the cache in `.MaxCompiler_build_user.conf`.

## 5 Optional Setup

---

You need to put a text file called `cache.conf` in the root of the cache directory with the line `use_arbiter=false`.

`cache.conf` can also have a `permissions` option which can be set to `user`, `group`, `other` or `all`, which have their Unix meanings. For example: `permissions=group`.



You cannot run multiple builds simultaneously using the core cache.



## 6 Uninstalling

To remove MaxCompiler, simply remove the MaxCompiler installation directory. You may wish to remove the environment variables that you set during installation.