Viewlogic Interface Guide

Printed in U.S.A.

The Xilinx logo shown above is a registered trademark of Xilinx, Inc.

FPGA Architect, FPGA Foundry, NeoCAD, NeoCAD EPIC, NeoCAD PRISM, NeoROUTE, Timing Wizard, TRACE, XACT, XILINX, XC2064, XC3090, XC4005, XC5210, and XC-DS501 are registered trademarks of Xilinx, Inc.

The shadow X shown above is a trademark of Xilinx, Inc.

All XC-prefix product designations, A.K.A. Speed, Alliance Series, AllianceCORE, BITA, CLC, Configurable Logic Cell, CORE Generator, CoreGenerator, CoreLINX, Dual Block, EZTag, FastCLK, FastCONNECT, FastFLASH, FastMap, Foundation, HardWire, LCA, LogiBLOX, Logic Cell, LogiCORE, LogicProfessor, MicroVia, PLUSASM, PowerGuide, PowerMaze, QPro, RealPCI, RealPCI 64/66, SelectI/O, Select-RAM, Select-RAM+, Smartguide, Smart-IP, SmartSearch, Smartspec, SMARTSwitch, Spartan, TrueMap, UIM, VectorMaze, VersaBlock, VersaRing, Virtex, WebLINX, XABEL, XACT*step*, XACT*step* Advanced, XACT*step* Foundry, XACT-Floorplanner, XACT-Performance, XAM, XAPP, X-BLOX, X-BLOX plus, XChecker, XDM, XDS, XEPLD, Xilinx Foundation Series, XPP, XSI, and ZERO+ are trademarks of Xilinx, Inc. The Programmable Logic Company and The Programmable Gate Array Company are service marks of Xilinx, Inc.

All other trademarks are the property of their respective owners.

Xilinx, Inc. does not assume any liability arising out of the application or use of any product described or shown herein; nor does it convey any license under its patents, copyrights, or maskwork rights or any rights of others. Xilinx, Inc. reserves the right to make changes, at any time, in order to improve reliability, function or design and to supply the best product possible. Xilinx, Inc. will not assume responsibility for the use of any circuitry described herein other than circuitry entirely embodied in its products. Xilinx, Inc. devices and products are protected under one or more of the following U.S. Patents: 4,642,487; 4,695,740; 4,706,216; 4,713,557; 4,746,822; 4,750,155; 4,758,985; 4,820,937; 4,821,233; 4,835,418; 4,855,619; 4,855,669; 4,902,910; 4,940,909; 4,967,107; 5,012,135; 5,023,606; 5,028,821; 5,047,710; 5,068,603; 5,140,193; 5,148,390; 5,155,432; 5,166,858; 5,224,056; 5,243,238; 5,245,277; 5,267,187; 5,291,079; 5,295,090; 5,302,866; 5,319,252; 5,319,254; 5,321,704; 5,329,174; 5,329,181; 5,331,220; 5,331,226; 5,332,929; 5,337,255; 5,343,406; 5,349,248; 5,349,249; 5,349,250; 5,349,691; 5,357,153; 5,360,747; 5,361,229; 5,362,999; 5,365,125; 5,367,207; 5,386,154; 5,394,104; 5,399,924; 5,399,925; 5,410,189; 5,410,194; 5,414,377; 5,422,833; 5,426,378; 5,426,379; 5,430,687; 5,432,719; 5,448,181; 5,448,493; 5,450,021; 5,450,022; 5,453,706; 5,455,525; 5,466,117; 5,469,003; 5,475,253; 5,477,414; 5,481,206; 5,483,478; 5,486,707; 5,486,776; 5,488,316; 5,489,858; 5,489,866; 5,491,353; 5,495,196; 5,498,979; 5,498,989; 5,499,192; 5,500,608; 5,500,609; 5,502,000; 5,502,440; 5,504,439; 5,506,518; 5,506,523; 5,506,878; 5,513,124; 5,517,135; 5,521,835; 5,521,837; 5,523,963; 5,523,971; 5,524,097; 5,526,322; 5,528,169; 5,528,176; 5,530,378; 5,530,384; 5,546,018; 5,550,839; 5,550,843; 5,552,722; 5,553,001; 5,559,751; 5,561,367; 5,561,629; 5,561,631; 5,563,527; 5,563,528; 5,563,529; 5,563,827; 5,565,792; 5,566,123; 5,570,051; 5,574,634; 5,574,655; 5,578,946; 5,581,198; 5,581,199; 5,581,738; 5,583,450; 5,583,452; 5,592,105; 5,594,367; 5,598,424; 5,600,263; 5,600,264; 5,600,271; 5,600,597; 5,608,342; 5,610,536; 5,610,790; 5,610,829; 5,612,633; 5,617,021; 5,617,041; 5,617,327; 5,617,573; 5,623,387; 5,627,480; 5,629,637; 5,629,886; 5,631,577; 5,631,583; 5,635,851; 5,636,368; 5,640,106; 5,642,058; 5,646,545; 5,646,547; 5,646,564; 5,646,903; 5,648,732; 5,648,913; 5,650,672; 5,650,946; 5,652,904; 5,654,631; 5,656,950; 5,657,290; 5,659,484; 5,661,660; 5,661,685; 5,670,896; 5,670,897; 5,672,966; 5,673,198; 5,675,262; 5,675,270; 5,675,589; 5,677,638; 5,682,107; 5,689,133; 5,689,516; 5,691,907; 5,691,912; 5,694,047; 5,694,056; 5,724,276; 5,694,399; 5,696,454; 5,701,091; 5,701,441; 5,703,759; 5,705,932; 5,705,938; 5,708,597; 5,712,579; 5,715,197; 5,717,340; 5,719,506; 5,719,507; 5,724,276; 5,726,484; 5,726,584; 5,734,866; 5,734,868; 5,737,234; 5,737,235; 5,737,631; 5,742,178; 5,742,531; 5,744,974; 5,744,979; 5,744,995; 5,748,942; 5,748,979; 5,752,006; 5,752,035; 5,754,459; 5,758,192; 5,760,603; 5,760,604; 5,760,607; 5,761,483; 5,764,076; 5,764,534; 5,764,564; 5,768,179; 5,770,951; 5,773,993; 5,778,439; 5,781,756; 5,784,313; 5,784,577; 5,786,240; 5,787,007; 5,789,938; 5,790,479;

5,790,882; 5,795,068; 5,796,269; 5,798,656; 5,801,546; 5,801,547; 5,801,548; 5,811,985; 5,815,004; 5,815,016; 5,815,404; 5,815,405; 5,818,255; 5,818,730; 5,821,772; 5,821,774; 5,825,202; 5,825,662; 5,825,787; 5,828,230; 5,828,231; 5,828,236; 5,828,608; 5,831,448; 5,831,460; 5,831,845; 5,831,907; 5,835,402; 5,838,167; 5,838,901; 5,838,954; 5,841,296; 5,841,867; 5,844,422; 5,844,424; 5,844,829; 5,844,844; 5,847,577; 5,847,579; 5,847,580; 5,847,993; 5,852,323; Re. 34,363, Re. 34,444, and Re. 34,808. Other U.S. and foreign patents pending. Xilinx, Inc. does not represent that devices shown or products described herein are free from patent infringement or from any other third party right. Xilinx, Inc. assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction if such be made. Xilinx, Inc. will not assume any liability for the accuracy or correctness of any engineering or software support or assistance provided to a user.

Xilinx products are not intended for use in life support appliances, devices, or systems. Use of a Xilinx product in such applications without the written consent of the appropriate Xilinx officer is prohibited.

Copyright 1991-1999 Xilinx, Inc. All Rights Reserved.

About This Manual

This manual describes the Xilinx Viewlogic® Interface program, a tool used for interfacing Viewlogic's Workview Office® or Powerview® with the Xilinx tools to create FPGAs or CPLDs.

Before using this manual, you should be familiar with the operations that are common to all Xilinx software tools: how to bring up the system, select a tool for use, specify operations, and manage design data. These topics are covered in the *Development System Reference Guide*. Other publications you can consult for related information are the *Libraries Guide*, the *LogiBLOX Reference/User Guide*, the *Design Manager/Flow Engine Guide*, the *EPIC Design Editor Reference/User Guide*, and *The CPLD Schematic Design Flow.*

Additional Resources

For additional information, go to http://support.xilinx.com. The following table lists some of the resources you can access from this page. You can also directly access some of these resources using the provided URLs.

Resource	Description/URL
Tutorial	Tutorials covering Xilinx design flows, from design entry to verification and debugging http://support.xilinx.com/support/techsup/tutorials/index.htm
Answers Database	Current listing of solution records for the Xilinx software tools Search this database using the search function at http://support.xilinx.com/support/searchtd.htm
Application Notes	Descriptions of device-specific design techniques and approaches http://support.xilinx.com/apps/appsweb.htm

Resource	Description/URL
Data Book	Pages from *The Programmable Logic Data Book*, which describe device-specific information on Xilinx device characteristics, including read-back, boundary scan, configuration, length count, and debugging http://support.xilinx.com/partinfo/databook.htm
Xcell Journals	Quarterly journals for Xilinx programmable logic users http://support.xilinx.com/xcell/xcell.htm
Tech Tips	Latest news, design tips, and patch information on the Xilinx design environment http://support.xilinx.com/support/techsup/journals/index.htm

Manual Contents

This manual covers the following topics.

- The "Introduction" chapter describes the programs that comprise the Viewlogic interface and the Xilinx-Viewlogic design flow for FPGAs and CPLDs.
- The "Getting Started" chapter discusses how to configure your system to use Workview Office on a PC. It also discusses how to use the Xilinx interface with Powerview on workstations.
- The "Design Entry" chapter describes how to use ViewDraw to enter a schematic design. Not intending to document all the features of Workview Office, it discusses in detail the features needed for Xilinx designs.
- The "Functional Simulation" chapter describes how to prepare a simulation network for functional simulation within the Viewlogic simulation environment. It also describes how to load ViewTrace to view the simulation signals in a waveform format.
- The "Implementing a Design" chapter describes how to use the Design Manager and the Flow Engine to translate and implement your design.
- The "Timing Simulation" chapter describes how to prepare a simulation network for timing simulation within the Viewlogic simulation environment. It also describes how to load ViewTrace to view the simulation signals in a waveform format.

- The “Design and Simulation Techniques” chapter discusses aspects of schematic entry and simulation you need to know to use Workview Office and Powerview effectively.

Conventions

This manual uses the following typographical and online document conventions. An example illustrates each typographical convention.

Typographical

The following conventions are used for all documents.

- `Courier font` indicates messages, prompts, and program files that the system displays.

  ```
  speed grade: -100
  ```

- **`Courier bold`** indicates literal commands that you enter in a syntactical statement. However, braces "{ }" in Courier bold are not literal and square brackets "[ ]" in Courier bold are literal only in the case of bus specifications, such as bus [7:0].

  `rpt_del_net=`

  `Courier bold` also indicates commands that you select from a menu.

  `File` → **`Open`**

- *Italic font* denotes the following items.
  - Variables in a syntax statement for which you must supply values

    `edif2ngd` *design_name*

  - References to other manuals

    See the *Development System Reference Guide* for more information.

- Emphasis in text

  If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected.

- Square brackets "[ ]" indicate an optional entry or parameter. However, in bus specifications, such as bus [7:0], they are required.

  `edif2ngd` [*option_name*] *design_name*

- Braces "{ }" enclose a list of items from which you must choose one or more.

  `lowpwr =`{**`on`**|**`off`**}

- A vertical bar " | " separates items in a list of choices.

  `lowpwr =`{**`on`**|**`off`**}

- A vertical ellipsis indicates repetitive material that has been omitted.

  ```
  IOB #1: Name = QOUT'
  IOB #2: Name = CLKIN'
  .
  .
  .
  ```

- A horizontal ellipsis "…" indicates that an item can be repeated one or more times.

  `allow block` *block_name loc1 loc2* `...` *locn;*

Online Document

The following conventions are used for online documents.

- Red-underlined text indicates an interbook link, which is a cross-reference to another book. Click the red-underlined text to open the specified cross-reference.
- Blue-underlined text indicates an intrabook link, which is a cross-reference within a book. Click the blue-underlined text to open the specified cross-reference.

Chapter 1

Introduction

This chapter describes the programs that comprise the Viewlogic interface and the Xilinx-Viewlogic design flow for FPGAs and CPLDs. This chapter contains these sections.

- "Viewlogic Toolsets"
- "Viewlogic Programs"
- "Xilinx Device Support"
- "Following the Design Flow"

Viewlogic Toolsets

The Viewlogic interface programs support both of Viewlogic's current systems, Workview Office and Powerview. However, this manual primarily discusses only Workview Office, the only Viewlogic software distributed by Xilinx.

Workview Office

The Workview Office software runs on a Windows-based system.

This release supports Workview Office version 7.5 and later on Windows 95 and Windows NT 4.0.

Powerview

The Powerview software runs on a workstation system with a graphic user interface.

This release supports Powerview V6.1 and later on SunOS™ V4.1.4, Solaris® V2.5, HP-UX® V10.2, and AIX® V4.1.5.

Viewlogic Programs

This section briefly describes the Viewlogic programs used with Workview Office and Powerview.

ViewDraw®

Viewlogic’s schematic entry tool.

Digital Fusion

Viewlogic’s Integrated Simulation Environment tool that calls ViewSim® or Speedwave™ to run functional or timing simulations.

ViewSim

Viewlogic’s gate-level simulator.

Speedwave

Viewlogic’s VHDL simulator for Workview Office.

VWaves™

Viewlogic’s waveform viewer and editor.

Xilinx Device Support

The Viewlogic interface supports XC3000A / L, XC3100A, XC4000E / L / EX / XL / XLA / XLT / XV, XC5200, Spartan / XL, Virtex, and XC9500 / XL / XV devices. XC9500 / XL / XV are CPLD devices and the other families are FPGA devices.

Following the Design Flow

Creating FPGA and CPLD designs with Viewlogic tools involves the following steps.

1. Enter your design with the ViewDraw schematic editor, making sure that you observe the Xilinx design requirements noted in this manual.
2. Test the functionality of your design by creating a functional simulation network (VSM file) and loading it into Digital Fusion to simulate the design. You can use VWaves to view the waveforms generated by the simulation.
3. Implement your FPGA or CPLD design using Xilinx Design Manager.
4. Verify the timing of your design by creating a timing simulation network (VSM file) and loading it into Digital Fusion to simulate the design. You can use VWaves to view the waveforms generated by the simulation.
5. Download the design to the Xilinx device and verify the board.

The following figure shows the Xilinx-Viewlogic design flow.

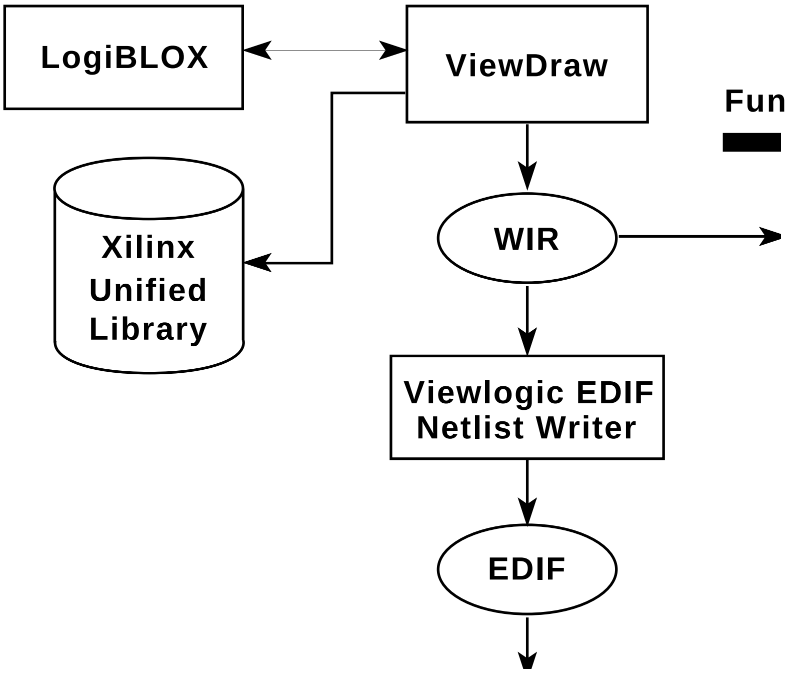


Figure 1-1 Xilinx-VIEWlogic Flow

Chapter 2

Getting Started

This chapter discusses how to configure your system to use Workview Office on a PC. It also discusses how to use the Xilinx interface with Powerview on workstations. This chapter contains these sections.

- "Configuring for Workview Office"
- "Configuring for Powerview"

Configuring for Workview Office

This manual provides instructions for Workview Office 7.5, including illustrations of that environment. However, you can use the Xilinx/Viewlogic interface programs with any current Viewlogic software, including Powerview 6.1.

Required Software

To run Workview Office, you need the following versions of the development software.

- Workview Office release 7.5 or later
- Xilinx/Viewlogic Interface and Libraries version M1 or later
- Xilinx Development System Software version M1 or later

Installing the Software

Before you can use the Workview Office software, you must set up your PC to use the Viewlogic and Xilinx Development System software.

1. Verify proper configuration of your system. Consult the Xilinx *Alliance Series 2.1i Installation Guide and Release Notes* for instructions about installing this software and setting up your machine to run the software.
2. Verify that your autoexec.bat file contains the following variables, assuming that you loaded the software noted in the previous step to the c:\wvoffice and c:\xilinx directories on your PC. If the software resides in different areas, modify the following Set statements according to the appropriate location. See the Xilinx *Alliance Series 2.1i Installation Guide and Release Notes* for additional information on system environment setup.

   - The PATH variable sets the overall executable search path. This variable must include the directories where the Workview Office and Xilinx Development System software reside, as in the following example.

   PATH=C:\XILINX\BIN\NT;C:\WVOFFICE;*other_paths*

     The PATH variable cannot include any previous version of either the Xilinx or Viewlogic software. Remove all paths to older software.

   - Xilinx and Workview Office software use the XILINX variable to locate data files. This variable must specify the directory where the Xilinx Development System resides, as in the following example.

   SET XILINX=C:\XILINX

   - Set the WDIR variable for the data file search path for the Workview Office software as follows.

   SET WDIR=C:\WVOFFICE\STANDARD

   - The LM_LICENSE_FILE variable directs the software to the license files. The location of these files do not matter as long as this variable points to the license files themselves, not just the directory where they reside. The first license file authorizes the Workview Office tools; the other authorizes the

Xilinx tools(M1.4 and older versions). Xilinx M1.5 and later versions does not require a license.

The following syntax applies to Workview Office 7.31 and older versions only.

```
SET LM_LICENSE_FILE=C:\WVOFFICE\STAN-
DARD\LICENSE.DAT,;C:\FLEXLM\LICENSE.DAT
```

Because the Workview Office 7.5 (and older versions) and the Xilinx M1 software use different versions of the FLEXlm software, you must specify the Workview Office license file first. Separate this path from the Xilinx license file by a comma followed by a semicolon (,;). For Workview Office 7.4 and newer versions, use only the semicolon as the delimiter.

- Speedwave uses the VANTAGE_VSS and VANTAGE_CC variables for functional VHDL simulation. Only systems with extended licenses require these variables as shown in the following examples.

```
SET VANTAGE_VSS=C:\WVOFFICE\V
```

```
SET VANTAGE_CC=C:\WVOFFICE\MSVCNT\BIN\CL
```

3. After you install the Xilinx and Viewlogic software tools, you must copy two files from the Xilinx install directory to the Workview Office tree. You can find these two files, LIBS.LST and POWERVIE.INI, in the C:\XILINX_PATH\VIEWLOG\DATA directory. In Windows Explorer, copy these to files to the STANDARD directory of the Workview Office tree (for example, C:\WVOFFICE\STANDARD), replacing the two files of the same names that currently exist there.

   You must install these two files for a Xilinx-specific installation of Workview Office.

   - *libs.lst*, a library definition that contains the updated listing for the M1 libraries.
   - *powervie.ini*, a file that contains a string that tells Workview Office to use a Xilinx-restricted license.

Verifying Your Software Setup

Before beginning a project, use the following steps to verify the correct software setup.

1. Verify the proper loading of Viewlogic and Xilinx software according to the instructions provided, and that your paths are set correctly.
2. Customize the Workview Office Toolbar according to your preference. Click on the Workview Office Icon (farthest to the left) to see your customizing options. The Moveable Window Toolbar appears in the following figure.

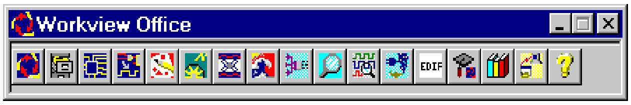
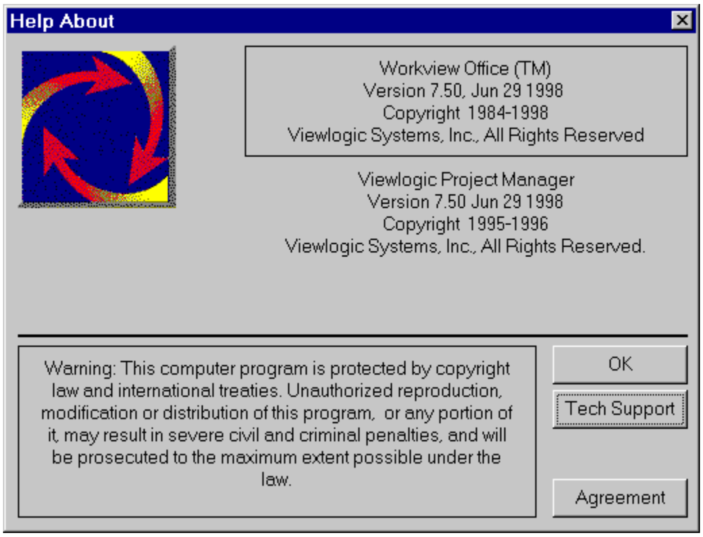


Figure 2-1 Workview Office Toolbar

3. Click on the **`Workview Office`** icon located at the far left on the Workview Office Toolbar. A pull-down menu appears.
4. Click on **`About Workview Office...`**

   The Workview Office “Help About” screen appears, as shown in the following figure.



Workview Office (TM)
Version 7.50, Jun 29 1998
Copyright 1984-1998
Viewlogic Systems, Inc., All Rights Reserved

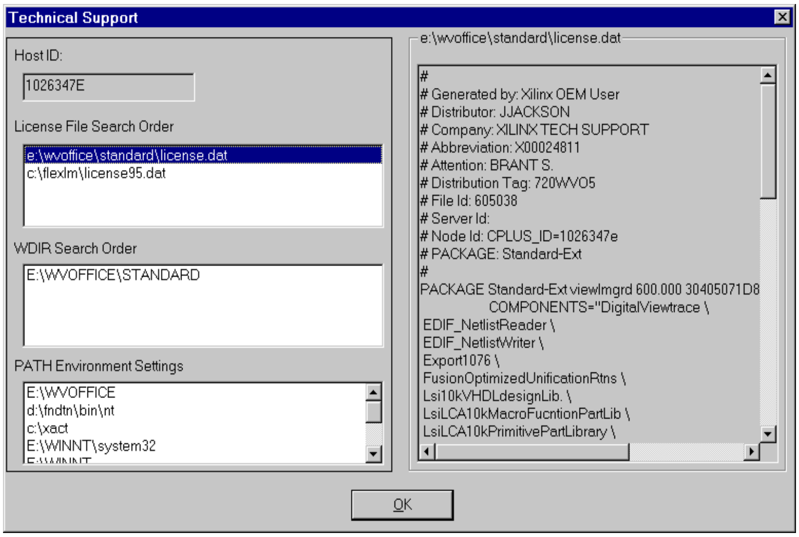
Viewlogic Project Manager
Version 7.50 Jun 29 1998
Copyright 1995-1996
Viewlogic Systems, Inc., All Rights Reserved.

Warning: This computer program is protected by copyright law and international treaties. Unauthorized reproduction, modification or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Figure 2-2 Workview Office Help About Window

5. Click on the **`Tech Support`** button.

   The Technical Support Box appears as shown in the next figure.



Host ID: 1026347E

License File Search Order:

```
e:\wvoffice\standard\license.dat
c:\flexlm\license95.dat
```

WDIR Search Order:

```
E:\WVOFFICE\STANDARD
```

PATH Environment Settings:

```
E:\WVOFFICE
d:\fndtn\bin\nt
c:\xact
E:\WINNT\system32
```

e:\wvoffice\standard\license.dat

```
#
# Generated by: Xilinx OEM User
# Distributor: JJACKSON
# Company: XILINX TECH SUPPORT
# Abbreviation: X00024811
# Attention: BRANT S.
# Distribution Tag: 720WVO5
# File Id: 605038
# Server Id:
# Node Id: CPLUS_ID=1026347e
# PACKAGE: Standard-Ext
#
PACKAGE Standard-Ext viewlmgrd 600.000 30405071D8
        COMPONENTS="DigitalViewtrace \
 EDIF_NetlistReader \
 EDIF_NetlistWriter \
 Export1076 \
 FusionOptimizedUnificationRtns \
 Lsi10kVHDLdesignLib. \
 LsiLCA10kMacroFucntionPartLib \
 LsiLCA10kPrimitivePartLibrary \
```

Figure 2-3 Technical Support Box

6. Verify that your Host ID number displays correctly. If you use a node-lock license, this number reads from the security key. If the key is missing, “unknown” appears in the box designated for the Host ID. Windows NT 4.0 sometimes requires the installation of a sentinel driver in order to display the security key. If your key is attached to the system and your Host ID appears as “unknown,” please look for the solution located at http://support.xilinx.com/support/searchtd.htm

7. Directly below your Host ID number is the path you designate for the License File Search Order. Click on the following line.

   `C:\WVOFFICE\STANDARD\LICENSE.DAT`

   The license.dat file appears in the display window to the right.

8. Scroll down to the bottom of this window and verify that the Host ID number that appears near the lower-right corner matches the one in the upper-left corner.

9. Verify that the WDIR Search Order displays C:\WVOFFICE\STANDARD, and that the PATH Environment Settings display properly. If either of these two windows is empty, modify the autoexec.bat file accordingly.

Note: As previously stated, these environment settings assume that you installed the Workview Office software to C:\WVOFFICE. If this is not the case, you must change these settings.

10. After verifying all the settings, Click **OK** and the Technical Support window closes. Click **OK** in the Help About window and it closes.

You can now open an existing project or start a new project.

Creating a Project

A project consists of a working directory that contains the sub-directories and data files for a given design. To create a project, select the libraries you need, arrange them in a particular search order, and save that scheme under the project directory. Saving this directory results in the creation of the following two files.

- *design*.vpj (Viewlogic project directory)
- viewdraw.ini (the initialization file)

The *design*.vpj file includes working directory and library information for your project. After you define your project libraries and set them up in the correct search order, saving the setup creates the file. This file can reside outside the project directory.

Project Manager creates the viewdraw.ini file automatically whenever you define or make changes to a project. This file contains library information as well as project settings. Do not modify the viewdraw.ini file directly. This file must reside in the project directory.

Setting Up Your Project Libraries

This section provides information about specifying the library search order. Use the following table to determine which library to use for particular Xilinx family members.

Table 2-1 Xilinx Families and Libraries

Family	Library
XC3000A, XC3000L, XC3100A	XC3000
XC4000E, XC4000L	XC4000E

Table 2-1 Xilinx Families and Libraries

Family	Library
XC4000EX, XC4000XL, XC4000XLA, XC4000XLT, XC4000XV	XC4000X
XC5200	XC5200
Spartan	Spartan
SpartanXL	SpartanXL
Spartan2	Spartan2
Virtex	Virtex
XC9500, XC9500XL, and XC9500XV	XC9500

The XC4000EX library changed in M1.4. Any design using the XC4000EX / XL / XV family should now use the XC4000X library.

Upgrading to M1.x for Existing M1.3 Designs

For existing M1.3 designs that use the XC4000EX library, you must make the following modifications when upgrading to M1.*x*. Taking these steps enables both the old (XC4000EX) and the new (XC4000X) aliases to refer to the new library.

1. With M1.x software installed, select the XC4000EX / XL / XV library listing.
2. In the library search order window, select the XC4000X library.
3. Modify the alias to read **`XC4000EX`**.
4. Click on **Add**, (not Change).
5. Use the **`Move Up`** button to position this new library reference below the existing XC4000X library.

Creating the Library Search Order

This section describes the steps for creating the library search order.

1. On the Workview Office Task Bar, click on the **`Project Manager`** icon.

   The Viewlogic Project Manager window appears.

2. Define the project directory.

A prompt asks you to enter your project directory when you create a new project. To change your project directory, select **`Edit`**→**`Project Directory`**, then browse to the location.

3. Select **`Edit`**→**`Libraries`** from the pulldown menus.

   The Library Search Order dialog box appears.

4. Click on the **`FPGA Lib...`** button in the Library Search Order dialog box.

   The FPGA Libraries menu appears as shown in the following figure.

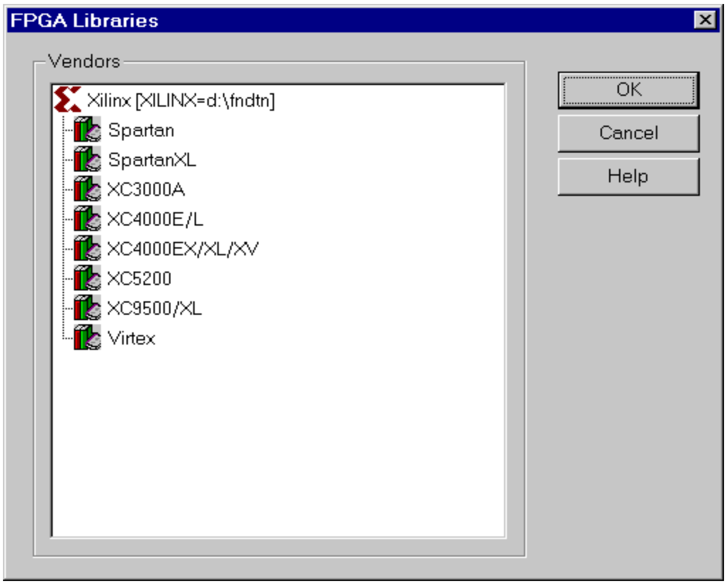


Figure 2-4 FPGA Libraries Menu

5. Choose a Xilinx family and click **`OK`**.

6. Add a Primary library for storing user-created symbols and schematics, and so you can apply the primary alias to all user-created symbols taken from this library. In the Library Information section of the Library Search Order window, create a new library with the following criteria.

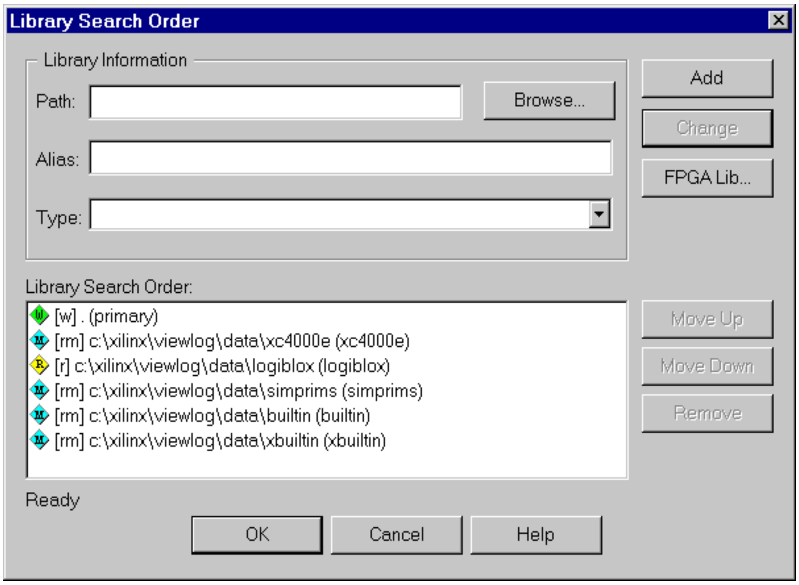
   - Path: .
   - Alias: primary

- Type: writable

Click on **Add** after creating the library.

7. Make this new directory first in the search order, so select this directory and click on the **Move Up** button until the Primary library appears at the top. Click **OK**.

   The Library Search Order window should look similar to the one in the next figure. This example shows the library search order for an XC4000E design.



```
[w] . (primary)
[rm] c:\xilinx\viewlog\data\xc4000e (xc4000e)
[r] c:\xilinx\viewlog\data\logiblox (logiblox)
[rm] c:\xilinx\viewlog\data\simprims (simprims)
[rm] c:\xilinx\viewlog\data\builtin (builtin)
[rm] c:\xilinx\viewlog\data\xbuiltin (xbuiltin)
```

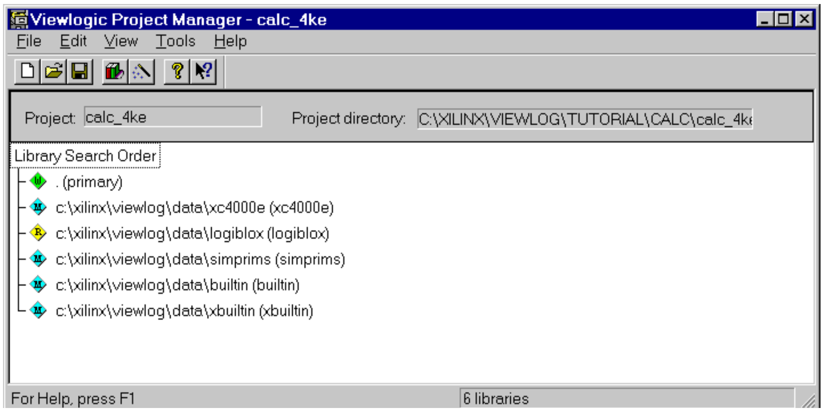
Figure 2-5 Final Library Search Order

8. Add any other user-created libraries.

   For each library, fill out the three fields in the Library Information section. Define the full path to the library, a unique alias, and the type, then click **Add**. Use the **Move Up** button to place these libraries between the primary library and the Xilinx family library.

9. Click **OK**.

   The Viewlogic Project Manager window appears with the libraries listed the same way you arranged them in the Library Search Order window, as shown in the following figure.


Viewlogic Project Manager - calc_4ke
File Edit View Tools Help

Project: calc_4ke   Project directory: C:\XILINX\VIEWLOG\TUTORIAL\CALC\calc_4ke

Library Search Order
- . (primary)
- c:\xilinx\viewlog\data\xc4000e (xc4000e)
- c:\xilinx\viewlog\data\logiblox (logiblox)
- c:\xilinx\viewlog\data\simprims (simprims)
- c:\xilinx\viewlog\data\builtin (builtin)
- c:\xilinx\viewlog\data\xbuiltin (xbuiltin)

For Help, press F1   6 libraries

Figure 2-6 Viewlogic Project Manager

10. Select **`File`**→**`Save As`** and enter *filename*.**`vpj`**.

    This saves all of the library and project information in the file-name.vpj file and automatically places (or updates) the view-draw.ini file in the project directory.

In summary, a valid Xilinx project requires a library search order that contains the following items (shown in the proper order).

1. Primary
2. User-created libraries (optional)
3. Family library (for example, XC4000E)
4. LogiBLOX (optional)
5. SimPrims
6. Builtin
7. Xbuiltin

To designate this as your current working project and begin working on this project in ViewDraw, open this file in the Viewlogic Project Manager. Close all other Viewlogic tools (such as ViewDraw and ViewSim) before changing projects.

Setting Up To Use LogiBLOX

Using the LogiBLOX Module Selector can require a few additional setup steps, depending on which type of simulation model you want LogiBLOX to create. This section outlines these steps.

1. Ensure that the Add LogiBLOX and Change LogiBLOX commands appear on the ViewDraw tools menu. If they do not, see the "Adding LogiBLOX Custom Commands to ViewDraw" section.
2. Ensure that your viewdraw.ini search order includes both the LogiBLOX library and the SimPrims library, each with the appropriate alias. See the "Creating the Library Search Order" section for details about this.
3. Decide which type of simulation model you want LogiBLOX to create, behavioral VHDL or gate-level EDIF. See the "Choosing Between VHDL and EDIF Models" section.
4. If you want LogiBLOX to create VHDL models, you need to complete the following steps.
   a) Locate the Speedwave library that contains the LogiBLOX VHDL source provided by Xilinx. If your site has not analyzed this library, you need to run the Speedwave analyzer. See the "Analyzing the LogiBLOX VHDL Library" section.
   b) Create a Speedwave working library in your current project directory for analyzed LogiBLOX models. Define the Speedwave VHDL library search order to include this working library, the LogiBLOX VHDL library (described above) and the standard IEEE library. See the "Initializing the Project for Speedwave" section.

Adding LogiBLOX Custom Commands to ViewDraw

You can add the following two custom Xilinx commands to the Tools menu in ViewDraw.

- Add LogiBLOX
- Change LogiBLOX

If these commands do not already appear on your ViewDraw **`Tools`** menu, you need to customize the Tools menu accordingly.

To customize the Tools menu, run the following from a command (MS-DOS) prompt.

`custmenu` *xilinx_path*`\viewlog\data\viewblox.txt`

The custmenu program installs with Workview Office and uses the information in the viewblox.txt file (provided by Xilinx) to customize the ViewDraw Tools menu. ViewDraw cannot run when you execute the custmenu command.

Alternatively, you can customize the Tools menu directly in View-Draw, by following these steps.

1. Start the ViewDraw application.
2. Select **Tools**→**Customize**.

   The Customize Tools Menu dialog appears.
3. Choose **User Menu** or **Common Menu.**

   Adding the LogiBLOX commands to the Common menu allows all users of ViewDraw on that machine to see the commands. Adding the commands to the User menu makes them visible only when you run ViewDraw.
4. To add the **Add LogiBLOX** command, complete the fields as shown in the following table and click the **Add** button.

Table 2-2 Adding LogiBLOX

In Field	Enter Text	Comments
Menu Text	**`&Add LogiBLOX`**	The & indicates that A is the keyboard equivalent for this menu item. You can choose a different mnemonic if you want.
Command	**`viewblox.exe`**	This executable resides in the same directory as all other Xilinx executables, found via your system path.

Table 2-2 Adding LogiBLOX

In Field	Enter Text	Comments
Arguments	**`$GUID`**	This tells ViewDraw to pass a unique code to LogiBLOX, which uses this code to identify the invoking instance of ViewDraw.
Initial Directory		Leave the directory field blank. This indicates that the initial directory is the project directory.

5. To add the **`Change LogiBLOX`** command, change the fields as shown in the next table and click the **Add** button.

Table 2-3 Changing LogiBLOX

In Field	Enter Text	Comments
Menu Text	**`Change L&ogiBLOX`**	The & indicates that O is the keyboard equivalent for this menu item. You can choose a different mnemonic if you want.
Command	**`viewblox.exe`**	This executable resides in the same directory as all other Xilinx executa-bles, found via your system path.
Arguments	**`$GUID /modify`**	This tells ViewDraw to pass a unique code to LogiBLOX, which uses this code to identify the invoking instance of ViewDraw. The modify option tells LogiBLOX that a Change command is being invoked.
Initial Directory		Leave the directory field blank. This indicates that the initial directory is the project directory.

6. Click the **`OK`** button to close the Customize dialog box.

Choosing Between VHDL and EDIF Models

For functional simulation purposes, LogiBLOX can create either behavioral VHDL models or gate-level EDIF models. Which type you choose depends on your available VIEWlogic simulator, as described in this section.

To use VHDL models, you must have a license for the Speedwave simulation engine (also known as Vantage). This simulation flow takes advantage of the mixed gate-VHDL capability of Speedwave, modelling LogiBLOX modules in VHDL, the remainder of the schematic design in gates.

If you do not have Speedwave available, use LogiBLOX to create EDIF models. These gate-level models translate into VIEWlogic WIR files, usable by the standard Viewsim gate simulator.

VHDL model creation is faster than EDIF model creation in LogiBLOX. Use VHDL models if your VIEWlogic environment supports it.

Specify the simulation model type in the LogiBLOX Setup dialog, as described in the "Adding LogiBLOX Components" section of the "Design Entry" chapter.

Analyzing the LogiBLOX VHDL Library

The VHDL models created by LogiBLOX use some standard functions defined in the LogiBLOX VHDL library. Xilinx provides the VHDL source files for this library and you can find these in the standard Xilinx installation area. You need to analyze this library for Speedwave, making sure that analyzed library is available at the time you create individual LogiBLOX modules.

You need to analyze the LogiBLOX VHDL library only once, unless you upgrade your version of Speedwave or Xilinx software. Multiple projects can reference a single copy of the analyzed LogiBLOX library.

To analyze the LogiBLOX VHDL library for Speedwave, run the following Xilinx-provided batch file from a command (MS-DOS) prompt.

`vaninit` *wvoffice_path*\v\pgm\libs

This creates the logiblox.lib Speedwave library directory under the specified *wvoffice_path*\v\pgm\libs directory.

Initializing the Project for Speedwave

LogiBLOX analyzes a VHDL simulation model into the current working Speedwave library after it creates the model. For this analysis to succeed, define the Speedwave VHDL library search order for the current project. The project requires a valid working library.

To create the working library and define the VHDL library search order, follow these steps.

1. Start the Digital Fusion or Speedwave application.
2. Choose **File**→**Analyze VHDL Design**.
3. After the HDL manager jumpstart wizard displays, click **Next**.
4. To use an existing VHDL library as your working library, click **Yes, please use my existing library** and browse for the library. Otherwise, click **Next** and your Library Path is set to your current Project Directory. Click **Next** again (if you used an existing library, skip to step 6).
5. You can now add your source files to your newly created VHDL library. Browse for your VHDL files and add them to your source files. After making the addition(s), click **Next**.
6. The VHDL System Libraries displays. Select it by clicking on the checkbox to the left of each Library. Select IEEE.LIB, and LOGIBLOX.LIB. If SYNOPSYS.LIB appears in the Available Libraries list, select it. (If you installed the Vantage Speedwave libraries with Workview Office, SYNOPSYS.LIB does not appear; you do not need it.) After you have made the selections, click **Finish** and the HDL Manager appears.
7. The VHDL View should now have four libraries with check marks (three libraries if you use the Vantage Speedwave libraries). The VHDL User Library in this list is the working library.
8. Choose **File**→**Save As** to save the VHDL library search order for the current project.
9. Choose **File**→**Exit** to leave the HDL Manager; you can also exit from Digital Fusion at this point.

Setting Up Other Custom Menus

There are additional custom Xilinx commands you can add to the Tools menu in ViewDraw.

- Write Xilinx EDIF
- Xilinx Functional Simulation
- Read Xilinx Timing EDIF

To customize the Tools menu with these commands, run the following from a command (MS-DOS) prompt.

- Windows 95

  `custmenu` ***`xilinx_path`*****`\viewlog\data\xvdraw95.exe`**

- Windows NT

  `custmenu` ***`xilinx_path`*****`\viewlog\data\xvdrawnt.exe`**

Alternatively, you can customize the Tools as explained in the "Adding LogiBLOX Custom Commands to ViewDraw" section. Complete the fields in Customize Tools Menu window as shown in the following tables.

Table 2-4 Adding Tools

In Field	Enter Text	Comments
Menu Text	**`&Write Xilinx EDIF`**	The & indicates that W is the keyboard equivalent for this menu item. You can choose a different mnemonic if you want.
Command	**`cmd.exe`**	This executable resides in the WINNT/system32 path of your Windows NT system. For Windows 95 system, change this command to command.exe.

Table 2-4 Adding Tools

In Field	Enter Text	Comments
Arguments	**`edifneto -l xilinx $BLOCK-NAMES`**	If the current schematic is called DESIGN.1, this command creates a DESIGN.EDN file readable by Xilinx M1 Design Manager. The –l xilinx option is required for Xilinx designs.
Initial Directory		Leave the directory field blank. This indicates that the initial directory is the project directory.

Table 2-5 Changing Tools

In Field	Enter Text	Comments
Menu Text	**`Read Xilinx &Timing EDIF`**	The & indicates that T is the keyboard equivalent for this menu item. You can choose a different mnemonic if you want.
Command	**`cmd.exe`**	This executable resides in the WINNT/system32 path of your Windows NT system. For Windows 95 system, change this command to command.exe.
Arguments	**`edifneti time_sim.edn`**	This command reads in the Timing Simulation Data produced by the Xilinx M1 Core Technology tools. TIME_SIM.1 writes into the project WIR directory, along with a number of XBA#.1 files. Follow this step with **`Tools`**→ **`Create Digital Netlist`**, filling in the *Design Name* field with TIME_SIM.1, to run a Viewlogic Timing Simulation.
Initial Directory		Leave the directory field blank. This indicates that the initial directory is the project directory.

Table 2-6 Functional Simulation Tools

In Field	Enter Text	Comments
Menu Text	**`Xilinx &Functional Simulation`**	The & indicates that F is the keyboard equivalent for this menu item. You can choose a different mnemonic if you want.
Command	**`vfuncsim.exe`**	This executable resides in the same directory as all other Xilinx executables, found via your system path.
Arguments	**`$BLOCKNAME`**	
Initial Directory		Leave the directory field blank. This indicates that the initial directory is the project directory.

VFUNCSIM.EXE runs five programs, EDIFNETO, NGDBUILD, NGD2EDIF, EDIFNETI, and VSM. Refer to the "Using Powerview to Create the Functional Simulation Netlist for Category A Designs" section of the "Functional Simulation" chapter for more information about the command lines used.

Configuring for Powerview

This section contains information about the required Powerview software, installing the software, setting up your project libraries, and setting up to use LogiBLOX.

Required Software

To run Powerview, you need the following versions of the development software.

- Powerview release 6.1 or later
- Xilinx/Viewlogic Interface and Libraries version M1 or later
- Xilinx Development System Software version M1 or later

Installing the Software

The following instructions assume that you already installed and configured Powerview software. Please consult the Viewlogic documentation for details about installing and configuring Powerview.

1. Install Xilinx's M1 Development System and the Xilinx-Viewlogic Interface Tools and Libraries. Ensure that your .cshrc or .login file sets the XILINX environment variable to point to the root of the Xilinx software tree and that your path contains the appropriate platform bin directory. Consult the Xilinx *Alliance Release Document* for instructions about setting up your machine to run the Xilinx software.

2. Define the WDIR environment variable for Powerview in your .cshrc or .login file. To integrate Xilinx LogiBLOX with Powerview, you must add the following directory to the beginning of your WDIR variable (separated from the existing path by a colon).

   ```
   $XILINX/viewlog/data/logiblox/standard
   ```

   For example, if Powerview is in /tools/powerview, enter the following.

   ```
   setenv WDIR $XILINX/viewlog/data/logiblox/stan-
   dard:/tools/powerview/standard
   ```

   This command adds this Xilinx directory to the WDIR variable so that ViewDraw can locate the ViewScript customizations for LogiBLOX. You can place this directory anywhere in your WDIR search path, provided that the Xilinx directory is the first (or only) one that contains a vdrawus.vs file (this file contains customizations to ViewDraw). If you have other customizations in a vdrawus.vs file elsewhere in your WDIR search path, you can merge the Xilinx customizations into your existing vdrawus.vs file. Add the following line to the end of that file.

   ```
   load( "logiblox.vs" );
   ```

Setting Up Your Project Libraries

Powerview uses the viewdraw.ini file to set up the ViewDraw environment. You modify the viewdraw.ini file manually.

The viewdraw.ini file contains the path and directory search order for your libraries. You must keep a copy of the viewdraw.ini file in each project directory so you can customize the library search order for each project.

The library search order format allows you to specify an unlimited number of directories, with only one primary directory, as well as the library directories search order. The order of the list of directories at the end of the viewdraw.ini file determines the search order.

Viewdraw.ini File Syntax

Use the following syntax when adding libraries to the viewdraw.ini file.

```
DIR [type] path (alias)
```

- *Path* is the full path specification of the library. Specifying a period (.) as the path name for the primary ([p]) directory causes ViewDraw to use the directory where the file resides as the base directory for all new schematics, user-generated symbols, netlists, and related simulation files.
- *Alias* is the library name associated with each component that you place in your schematic.
- *Type* indicates one of the following library directory formats.
  - [p] denotes the primary or project directory. All symbols and schematics that you create are saved here.
  - [w] indicates a read-write directory.
  - [r] indicates a read-only directory.
  - [m] indicates a library compressed into a megafile format. Megafiles are read-only by default.

Adding Xilinx Libraries

Use the information in the "Xilinx Families and Libraries" table to determine which library to use for each particular Xilinx family member.

Upgrading to M1.x for Existing M1.3 Designs

For existing M1.3 designs that use the XC4000X library, you must make the following modifications when upgrading to M1.*x*. Taking these steps enables both the old (XC4000EX) and the new (XC4000X) aliases to refer to the new library.

- Follow the library setup instructions required for the XC4000X library.
- Duplicate the XC4000 library, modifying the alias to **XC4000EX**. Ensure both of the following lines exist in your viewdraw.ini.

  DIR [m] /*xilinx_path***/viewlog/data/xc4000x (xc4000x)**

  DIR [m] /*xilinx_path***/viewlog/data/xc4000x(xc4000ex)**

You cannot mix different family libraries in the same project directory. Specify libraries from exactly one Xilinx family for each project directory.

Adding SimPrims, Builtin, and Xbuiltin Libraries

The Xilinx Viewlogic library package includes the builtin library, a simplified version of the complete builtin library sold by Viewlogic. Use only the Xilinx-supplied builtin library (/*xilinx_path*/viewlog/data/builtin) in any Xilinx project directory.

You must add the SimPrims, builtin, and xbuiltin libraries (the Viewlogic simulation model libraries) to the viewdraw.ini file.

You cannot use the SimPrims, builtin, or xbuiltin libraries to capture your design.

Adding Library Aliases

When specifying the library search order, you must also add a library alias to each library directory. A library alias is a name that identifies a specific library directory along with the parts that it contains. The alias distinguishes identically named components from different libraries. You must specify, in parenthesis, in the viewdraw.ini file the aliases for each Xilinx library for proper netlist translation.

Specifying the Library Search Order

Library directories should conform to the search order shown in the following table.

Table 2-7 Library Search Order

Type	Path	Alias
p	/*project_directory*	primary
m	/*xilinx_path*/viewlog/data/xc3000	xc3000
m	/*xilinx_path*/viewlog/data/xc4000e	xc4000e
m	/*xilinx_path*/viewlog/data/xc4000x	xc4000x
m	/*xilinx_path*/viewlog/data/xc5200	xc5200
m	/*xilinx_path*/viewlog/data/spartan	spartan
m	/*xilinx_path*/viewlog/data/spartanxl	spartanxl
m	/*xilinx_path*/viewlog/data/spartan2	spartan2
m	/*xilinx_path*/viewlog/data/virtex	virtex
m	/*xilinx_path*/viewlog/data/xc9000	xc9000
r	/*xilinx_path*/viewlog/data/logiblox	logiblox
m	/*xilinx_path*/viewlog/data/simprims	simprims
m	/*xilinx_path*/viewlog/data/builtin	builtin
m	/*xilinx_path*/viewlog/data/xbuiltin	xbuiltin

Include the library alias names under the Library column. You must enter these aliases exactly as shown.

You can add user-defined libraries, but you must add them after the primary directory.

XC4000XL/XV Library Search Order Example

For a new XC4000XL/XV design on a workstation, ensure the library definition section looks like this.

```
DIR [p] .(primary)
DIR [m] /xilinx_path/viewlog/data/xc4000x (xc4000x)
DIR [r] /xilinx_path/viewlog/data/logiblox (logiblox)
DIR [m] /xilinx_path/viewlog/data/simprims (simprims)
DIR [m] /xilinx_path/viewlog/data/builtin  (builtin)
DIR [m] /xilinx_path/viewlog/data/xbuiltin (xbuiltin)
```

Setting Up To Use LogiBLOX

Using the LogiBLOX Module Selector requires a few additional setup steps, depending on which type of simulation model you want LogiBLOX to create. This section outlines these steps.

Note: When you use LogiBLOX for the first time in a new project, it checks your setup, allowing you to correct any setup problems at that time. If you want LogiBLOX to do the setup for you, you need only review the "Choosing Between VHDL and EDIF Models" section.

1. Ensure that your viewdraw.ini search order includes both the LogiBLOX library and the SimPrims library, each with the appropriate alias. See the"Specifying the Library Search Order" section for details about this.

2. Decide which type of simulation model you want LogiBLOX to create, either behavioral VHDL or gate-level EDIF. See the "Choosing Between VHDL and EDIF Models" section.

3. If you want LogiBLOX to create VHDL models, you need to complete the following steps.

   a) Locate the Vantage library that contains the LogiBLOX VHDL source provided by Xilinx. If this library has not already been analyzed at your site, you need to run the Vantage analyzer. Ensure your vsslib.ini file contains the path to the analyzed library. See the "Analyzing the LogiBLOX VHDL Library" section.

   b) Create a Vantage working library in your current project directory to contain analyzed LogiBLOX models. Your vsslib.ini file designates this as the working library. See the "Creating a Vantage Library for the Project" section.

   c) Create a vsslib.ini initialization file, required by LogiBLOX and by the Fusion simulator. See the "Creating the vsslib.ini Library List File" section.

Choosing Between VHDL and EDIF Models

For functional simulation purposes, LogiBLOX can create either behavioral VHDL models or gate-level EDIF models. Which type you choose depends on which VIEWlogic simulator you have available, as described in this section.

To use VHDL models, you must have a license for the FusionHDL or Fusion/Speedwave simulation engine (also known as Vantage). This simulation flow takes advantage of the mixed gate-VHDL capability of Fusion, modelling LogiBLOX modules in VHDL, the remainder of the schematic design modelled in gates.

If you do not have a Fusion/Speedwave license, use LogiBLOX to create EDIF models. These gate-level models translate into VIEW-logic WIR files, simulated by the standard Viewsim gate simulator.

Because VHDL model creation is faster that EDIF model creation in LogiBLOX, use VHDL models if your VIEWlogic environment supports it.

Specify the simulation model type in the LogiBLOX Setup dialog, as described in "Adding LogiBLOX Components" section of the "Design Entry" chapter.

Analyzing the LogiBLOX VHDL Library

The VHDL models created by LogiBLOX use some standard functions defined in the LogiBLOX VHDL library. Xilinx provides the VHDL source files for this library. Find these VHDL source files in the standard Xilinx installation area. You must analyze this library for Vantage. Make the analyzed library available at the time you create individual LogiBLOX modules.

You do not need to re-analyze the LogiBLOX library for every new project. However, a Vantage library is specific to the platform and operating system under which it was analyzed, so you must ensure that the library you use is correct for your environment.

To analyze the LogiBLOX VHDL library for Vantage, run the following command from the UNIX prompt.

vaninit *parent_directory*

This Xilinx-provided script creates a new logiblox.lib Vantage library directory under the *parent_directory* that you specify.

If you want to analyze the LogiBLOX library manually, follow these steps.

1. Ensure that the Vantage analysis tools are properly configured. Define the VANTAGE_VSS environment variable and place the Vantage analysis tools in the system path. Consult the VIEWlogic documentation for details about this.

2. Change to the directory where you want to create the Vantage library for LogiBLOX. The Vantage tools create a directory called “logiblox.lib” below the current one. Run the remaining steps from this directory.
3. Run the following command to create a new Vantage library for the LogiBLOX VHDL source.

   ```
   vanlibcreate logiblox.lib LOGIBLOX
   ```

   The symbolic name of the new library is the last argument to this command, **LOGIBLOX**. Do not substitute any other name, as the LogiBLOX-generated VHDL models reference the library by this name.
4. Run the following commands to analyze the LogiBLOX VHDL source files.

   ```
   analyze -src $XILINX/vhdl/src/logiblox/
   mvlutil.vhd -lib logiblox.lib -libieee
   ```

   ```
   analyze -src $XILINX/vhdl/src/logiblox/
   mvlarith.vhd -lib logiblox.lib -libieee
   ```

   ```
   analyze -src $XILINX/vhdl/src/logiblox/logi-
   blox.vhd -lib logiblox.lib -libieee
   ```

   Analyze the three VHDL source files in the order shown above.

Your vsslib.ini file for each project using LogiBlox specifies the Vantage library created here. You can find more information about the vsslib.ini file in the “Creating the vsslib.ini Library List File” section.

Creating a Vantage Library for the Project

LogiBLOX creates and then analyzes a VHDL simulation model into the current working Vantage library. Typically, you want this working library to reside under the VIEWlogic project directory. This section describes how to create a new Vantage library for this purpose. The following section describes how to designate this new library as the working library in vsslib.ini.

To create a new Vantage library under the project directory, move to the project directory and run the following command.

```
vanlibcreate logiview.lib LOGIVIEW
```

You can choose a different directory name and symbolic name for this library.

Creating the vsslib.ini Library List File

The vsslib.ini file tells the Vantage analysis tools (run from LogiBLOX) where to find the current working library, the analyzed LogiBLOX VHDL library, and the standard IEEE library.

The vsslib.ini file, a simple text file, lists the path to each Vantage library directory on a separate line. The first directory listed is considered the working library during analysis of LogiBLOX models. The order of directories is otherwise irrelevant.

The vsslib.ini file specifies the following directories.

1. The Vantage library into which LogiBLOX models analyze. Make this the first directory listed in vsslib.ini because it is considered the working directory.
2. The analyzed version of the LogiBLOX VHDL library.
3. The standard IEEE library provided with the Vantage tools.

The following example shows a complete vsslib.ini file.

```
/proj/designs/memmap/logiview.lib
/proj/vanlibs/xilinx/logiblox.lib
/tools/powerview/standard/van_vss/pgm/libs/ieee.lib
```

In this example, *logiview.lib* is the working library.

Chapter 3

Design Entry

This chapter describes how to use ViewDraw to enter a schematic design. This chapter does not document all the features of Workview Office, but instead discusses in detail the tools you need for specific Xilinx features.

For more information about commands and tools in Workview Office, consult the online help files described in the "Obtaining Help" section.

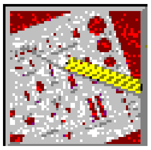
This chapter contains these sections.

- "Saving Schematics"
- "Closing Schematics"
- "Converting a Design"

Invoking ViewDraw

To start ViewDraw, follow this procedure.

1. Verify that a valid project exists in the Workview Office Project Manager.
2. From the Workview Office Toolbar, click on the ViewDraw button, shown in the next figure. ViewDraw opens.



Opening an Existing Schematic

Use the following steps to open an existing schematic.

1. To open an existing schematic, select **`File`**→**`Open`** in ViewDraw.

   Viewlogic allows a schematic to contain multiple sheets, saved as separate files to the project directory's sch directory. The extension of the file is the actual sheet number. For example, in a top-level schematic with two sheets, the sch directory contain *schematic*.1 and *schematic*.2 files.

2. To select the top-level schematic, select the primary library (or any other user-defined library) on the right side of the Open dialog box, shown in the following figure. You can either type the name directly in the Schematic field or click on the file in the design list box.

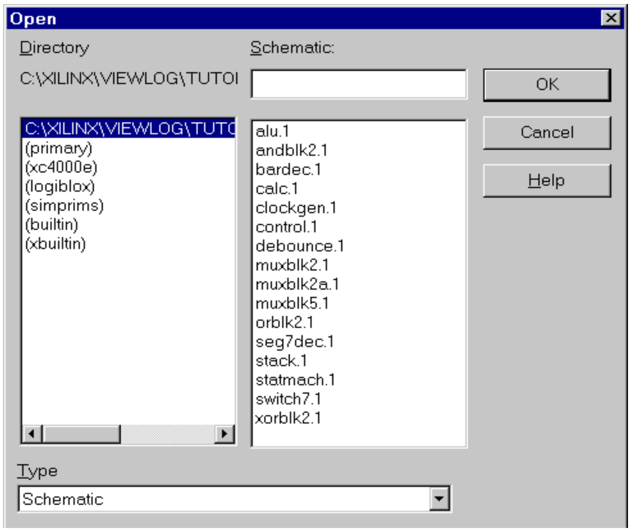


Figure 3-1 File Open Dialog Box

3. Click on **OK**. The selected schematic now appears.

Creating a New Schematic

The following procedure demonstrates how to open and create schematics in ViewDraw.

1. Open a blank schematic window by clicking on **File**→**New**.
2. In the **Name** field, type in the name of the new schematic. Make sure that Schematic highlights in the **Type** field.
3. Click on **OK**. The blank schematic sheet now opens with the name of the new schematic at the top of the window, followed by a .1 extension.

Creating Schematics with Multiple Sheets

When you create multiple-sheet designs, each sheet must have the same file name with extensions .1, .2, .3, and so forth. This convention applies to both top-level and lower-level schematics. The default sheet number is 1. To open a different sheet, type the schematic name and sheet number separated by a period in the Name field, or click on the file in the Designs list box. To view each sheet, use the **Goto**

Page icon from the View Toolbar or right mouse click and select **GoTo Page**.

Changing the ViewDraw Window Colors

You can change the ViewDraw color settings to make it easier to view a schematic, choosing from a variety of available color schemes. In addition to these, Xilinx provides a special color scheme that provides optimal viewing of the Xilinx libraries in ViewDraw. If you want to change to this color palette, proceed with these steps.

1. Select **Project→Settings**.
2. Click on the **Color Palette** tab.
3. Click the **Import Scheme** button.
4. In the Load Color Scheme dialog box that appears, navigate to the directory containing the Xilinx Development System, and then go to the viewlog\data subdirectory. Select the **xilinx.scm** file, and click **Open**.
5. Back in the Project Settings dialog box, the color scheme appears as Xilinx Library. Click on **Apply** to activate this new color palette. Click on **OK** to save the settings and close the Project Settings dialog box.

Changing ViewDraw Settings

The Project Settings dialog box allows you to change various other aspects of the ViewDraw display and save these settings to the project's viewdraw.ini file.

1. Select **Project→Settings** (if you closed it from the previous step).
2. Select any tab and make desired global changes.
3. To activate the new settings, click on **Apply** or **OK**. Apply keeps the Project Settings box open, **OK** closes it.

To find out the current settings of the key parameters without bringing up the dialog box, select **Project→Status**. ViewDraw opens a window displaying the settings.

Tools→Check Project gives an error stating that LogiBLOX symbols do not have an underlying schematic. You can ignore this error safely, as an NGC netlist describing schematic functionality merges in later.

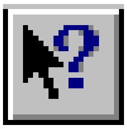
Obtaining Help

You can obtain help about ViewDraw's commands and procedures by selecting commands on the Help menu or selecting the Help icon in the toolbar. In addition, the dialog boxes associated with some commands offer a Help button that you can click on to obtain context-sensitive help.

Help from the Toolbar

To obtain help from the toolbar, follow these steps.

1. Click on the **`Help`** toolbar icon, shown in the following figure. Notice that your cursor now includes a question mark.



2. Select any icon or pull-down menu to get the information you need.

The ViewDraw Help screen appears showing the command you have selected. To go to the main ViewDraw Help window, select the **`Help Topics`** button.

Help from the Menu

To obtain help from the pull-down menu, select **`Help`**→**`ViewDraw Help Topics`**.

The main ViewDraw Help screen appears.

Help Topics

From this main help window, you can search for help in three ways.

- Under the Contents tab, follow the books to chapters dealing with such topics as Getting Started, or Problem Solving.
- Under the Index tab, select a keyword from the list.

- Under the Find tab, enter a keyword from a complete list of keywords. Similar to the Index, this method takes longer to find information due to the greater number of keywords available.

Working in ViewDraw

This section briefly discusses using mouse buttons, menus, keyboard commands, function keys, toolbar icons, and dialog boxes in View-Draw.

Mouse Buttons

Mouse buttons perform the following functions in ViewDraw.

- The left mouse button selects objects in ViewDraw. Use the Control key with the left mouse button to select multiple items.
- The right mouse button brings up a menu of common commands, like Properties, Add Component, Add Net, Delete, and Zoom. These commands refer to the selected items, or the schematic sheet itself if you selected nothing. The menu choices vary depending on the items selected.

Menus

ViewDraw offers eight menus. You can select menu commands with the mouse or the keyboard. With the mouse, click the left mouse button on the desired command. With the keyboard, press the Alt key and type the letter underlined in the command.

This document uses only the menu command in most situations. You can access most commands in a number of other ways, including keyboard hot keys or function keys, the command line, toolbar icons, and the right mouse button.

Keyboard Commands

You can use a set of “hot keys” in place of the pull-down menu or the toolbar icon. The keyboard commands appear next to their equivalent command in the pull-down menus. For example, selecting **File→Open** displays the keyboard command Ctrl+O on the right side of the pull-down menu.

Function Keys

You can use the function keys labeled F1, F2, F3, and so forth on your keyboard to invoke particular commands in the Workview Office tools. The next figure shows their assigned functions.

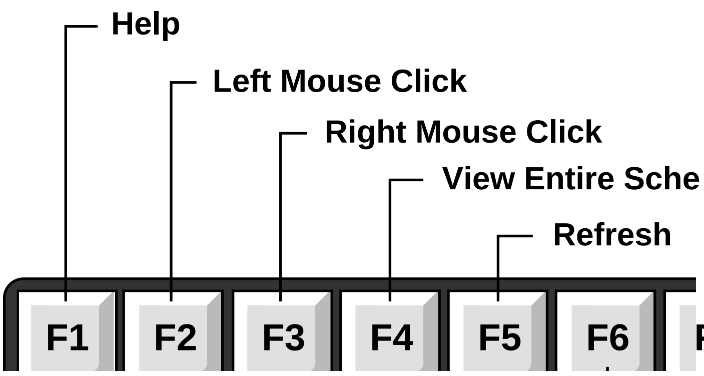


Figure 3-2 Function Keys

Command Line

You can enter certain commands via the command line, shown in the following figure. Open this toolbar from **View**→**Command Line**, and you can dock or float it like the other toolbars. To obtain the complete list of valid command line commands, select **Help**→**ViewDraw Help Topics**. Under the Index tab, select "Command Line Commands" and click on the **Display** button.

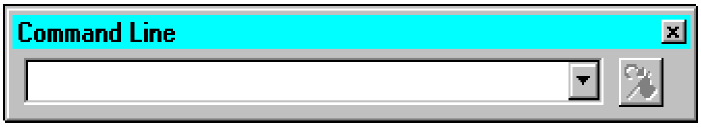


Figure 3-3 Command Line Toolbar

Toolbar Icons

Toolbar icons appear around the edges of ViewDraw. When you move the cursor over each icon, a description of its function appears in the status bar at the bottom of the ViewDraw screen. You can place these toolbars anywhere around the ViewDraw screen, or you can pull them off and leave them "floating." See the "Toolbar" figure, "View Toolbar" figure, "Object Toolbar" figure, and "Transform Toolbar" figure. You can turn them on or off from the View pull-down menu.

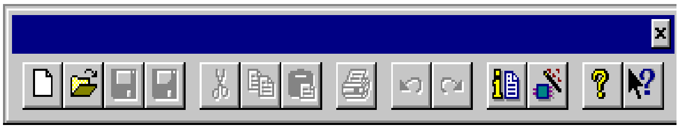


Figure 3-4 Toolbar

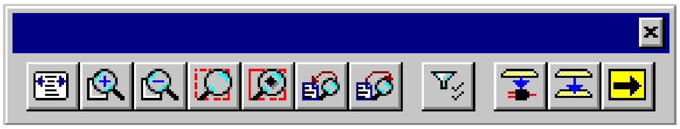


Figure 3-5 View Toolbar

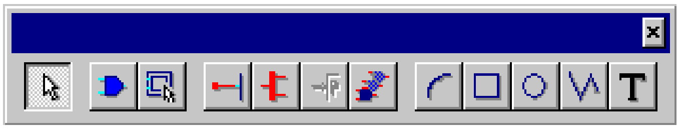


Figure 3-6 Object Toolbar

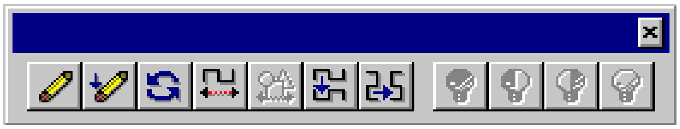


Figure 3-7 Transform Toolbar

Dialog Boxes

An ellipsis (...) following a ViewDraw menu command indicates that the command brings up a dialog box in which you can enter information and set options.

You can find the following common fields are common in many of the ViewDraw dialog boxes.

- OK closes the dialog box and implements the intended action according to the settings in the dialog box
- Cancel closes the dialog box without effecting any action

- Help gives you information about how to use the dialog box

Adding Components

Follow these steps to place library components on your schematic.

1. To place a component on the schematic, select the **`Add`**→**`Component`** command or click on the Add Component toolbar icon, shown in the next illustration.

   The Add Component dialog box appears, shown in the "Add Component Dialog Box" figure.

   When you initially bring up the Add Component dialog box, ViewDraw displays the components found in the first library in the Directory field.

2. To view the available components in a library, select the library in the **`Directory`** list box.

   The previous action updates the Components list box in the dialog box.

3. Using the down arrow in the scroll bar of the components list box, scroll down until you find the desired component.

Note: After selecting the components list box, you can jump to a component by typing the component's name.

4. Select the component.

   This highlights the component, updates the Symbol field, and an image of the component appears on the right.

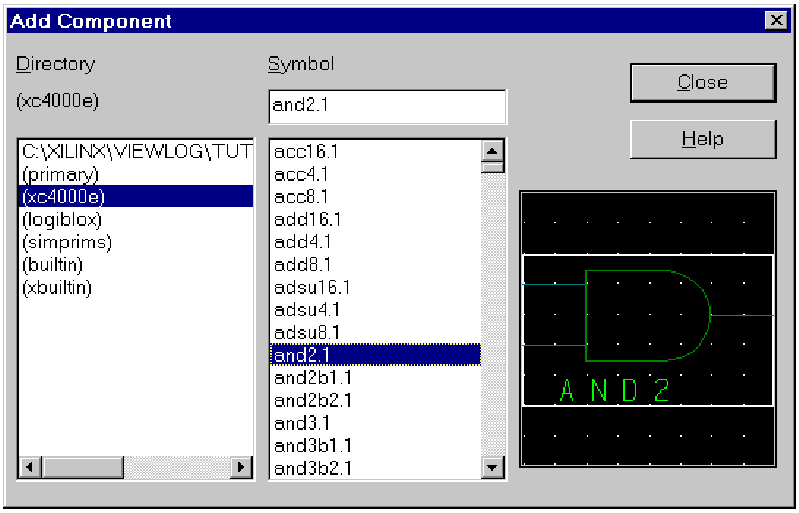


Figure 3-8 Add Component Dialog Box

5. To place a component on the schematic, left click on the image of the component. Move the cursor into the schematic window and left click again to place the component.

   Make sure all user-defined blocks that you add to your design come from the Primary directory, not from the discrete path that describes your project directory. This ensures that each symbol in the design has a proper alias, reducing the risk of errors for programs like Altran that refer to these aliases.

6. Place as many components as desired, then click on the **Close** button to close the Add Component dialog box.

   Do not use **Add→Component** to select LogiBLOX components from the LogiBLOX library directly. The "Adding LogiBLOX Components" section describes how to add LogiBLOX modules to your design using the interactive LogiBLOX Module Selector.

Adding LogiBLOX Components

This section describes how to add LogiBLOX components to your schematic design in ViewDraw. Refer to the *LogiBLOX Users Guide* for detailed information on creating modules with LogiBLOX.

Follow these steps to place a LogiBLOX component on your schematic.

1. Select the **`Tools`**→**`Add LogiBLOX`** command. If this command does not appear on the Tools menu, refer to the "Adding LogiBLOX Custom Commands to ViewDraw" section of the "Getting Started" chapter.

Note: Powerview users choose **`Add`**→**`LogiBLOX`** to start LogiBLOX. If this command does not appear on the Add menu, refer to the "Configuring for Workview Office" section of the "Getting Started" chapter.

2. When adding the first LogiBLOX component in a project, the LogiBLOX Setup dialog appears. Pay particular attention to the following two important settings for a ViewDraw schematic design.

   a) The **`Device Family`** setting must match the architecture library specified for this project (in the Project Manager or viewdraw.ini file).

   b) The **`Simulation Model`** setting (under the **`Options`** tab) must indicate the type of functional simulation model you want LogiBLOX to create, either behavioral VHDL or gate-level EDIF. Refer to the "Choosing Between VHDL and EDIF Models" section of the "Getting Started" chapter for details.

   Refer to the *LogiBLOX Users Guide* for information about the remaining LogiBLOX Setup options.

   Click on **`OK`** to close the Setup dialog.

3. The LogiBLOX Module Selector dialog appears. Define the module type and parameters, and give the module a name.

Note: Because the corresponding ViewDraw symbol, also uses the module name, choose a name that conforms to the standard VIEWlogic naming restrictions.

4. Click on **`OK`** in the Module Selector dialog. This verifies module definition and generates a simulation model before the Module Selector disappears.

5. LogiBLOX prepares the simulation model for use in the VIEWlogic environment (analyzing the VHDL model, or translating the EDIF model to WIR files).

6. LogiBLOX generates a ViewDraw symbol and places the symbol on the current schematic sheet in move mode. You can then drop the LogiBLOX symbol in the proper location, as you would do with the **`Add`**→**`Component`** command.

The component created by LogiBLOX is represented by a symbol in your primary design directory. You can place this symbol on your schematic as many times as you like, either by copying the first instance, or by choosing the new symbol with the **`Add`**→**`Component`** command. If you want to change the definition of the LogiBLOX component, see the "Changing LogiBLOX Components" section.

Changing Components

Rather than placing a new component, you can convert a placed component to another component. Use this method if you placed the wrong component and want to replace it with the correct component.

1. Select the component(s) to change. If you select multiple items at on time, each of the instances changes to the one target component.
2. Select the **`Edit`**→**`Replace`** command. The Find dialog box appears.
3. In the Object Type box, select Component. The Expression window now shows "<Selected Components>."
4. Fill in the Replace With box with the component type you want. If you do not know the exact name of the component, click on **`Browse...`** to obtain the listing of all available components. This Replace With Component dialog box looks similar to the Add Component box, with one difference; after you select the component, click on **`OK`**. The Find Dialog Box appears as shown in the following figure.

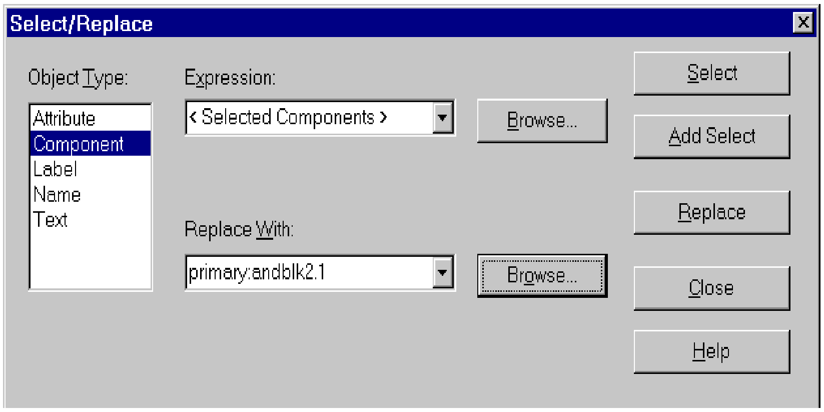


Figure 3-9 Find Dialog Box

5. Click on **Replace**, then **Close**. This closes the Find dialog box and updates the schematic.

Changing LogiBLOX Components

If you want to change the definition of a LogiBLOX component, follow these steps.

1. On your ViewDraw schematic, select the LogiBLOX component you want to modify. If more than one instance of this LogiBLOX component exists and you want to change them all, you can select any one of the instances.
2. Select the **Tools→Change LogiBLOX** command.

   Powerview Users choose **Change→LogiBLOX**.
3. The LogiBLOX Module Selector dialog appears with the selected module as the active one. Make the desired changes to the module definition.
4. If you want to overwrite the original module definition, you can simply click **OK**, without changing the module name. This action causes all instances of this module to change (including instances other than the selected one).

   Or, if you want to keep the original module definition, changing only the selected instance, enter a new module name before clicking **OK**. After the command completes, a new symbol replaces the selected symbol.

Adding Nets

Nets and buses establish connectivity between pins on the same hierarchical level of a design. However, you do not need to physically connect nets on the schematic. If two dangling nets or buses in a single schematic share the same name, they are considered electrically connected. Labeling unconnected nets with the same name can make schematics easier to read, especially when dealing with common signals like clocks or resets. However, you must keep track of all signal names and make sure they match exactly.

Note: Net name association also applies to nets on different schematic sheets at the same level of hierarchy. For example, a net on design.1 named DATA0 connects electrically to another net on design.2 named DATA0.

Use the following procedure to add a net.

1. Select the **`Add`**→**`Net`** command. See the associated icon shown in the next figure.

2. Point the cursor at the point where the net starts, a point on a net, bus, or pin.
3. Click and hold the left mouse button.
4. Click the right mouse button without releasing the left mouse button each time that you want the net to pivot.
5. Move the mouse to the desired end point of the net and release the left mouse button. Unlike the start point, you can place the end point somewhere other than on a net, bus, or pin.

Adding Buses

You can draw a set of signals as a bus rather than as several separate wires. You do not have to connect a bus physically with the nets that make up the bus.

To add a bus, follow these steps.

1. Select the **Add→Bus** command. See the associated icon shown as follows.
2. Point the cursor at the point where the bus starts.
3. Click and hold the left mouse button to begin the bus.
4. Click the right mouse button without releasing the left mouse button each time that you want the bus to pivot.
5. Move the mouse to the desired end point of the bus and click the left mouse button. Unlike the start point, you can place the end point somewhere other than on a net, bus, or pin.

Creating Custom Macros

A macro is any symbol defined by an underlying Viewlogic schematic. The Xilinx libraries contain several symbols that are macros. Macro schematics contain primitives and other macro symbols. When the software reads a schematic, it expands (flattens) the macro symbols into their underlying schematics. The components actually processed and reported by the software are the underlying primitives, referenced by their hierarchical instance names, after macro expansion.

You can create your own custom macro symbols to use in your designs. The procedure for creating a custom macro is the same for CPLDs and FPGAs. You can create and store your custom macro symbols and their underlying schematics in your local project directory, or you can create a custom library directory to store your custom macros for use in multiple projects. See the “Design and Simulation Techniques” chapter for instructions on creating a custom library.

Never add custom symbols or macros to the Xilinx library directories or modify any of the library symbols or macros. You can, however, copy any of the Xilinx-supplied macros from the library into your project or custom library directory and modify them to suit your design needs, saving the modified component to your local primary directory.

Note: When using a Xilinx library macro as a template for a new component, specify a new name when saving the component to avoid confusion.

When you create a custom macro symbol, you must set the Viewlogic block type to Composite, not Module. The Xilinx software for macro symbols does not require symbol attributes. However, if you copy a Xilinx-supplied primitive symbol from the library to use as the basis of your custom macro symbol, make sure you delete the LEVEL=XILINX and LIBVER=2.0.0 symbol attributes. You delete these symbol attributes because they mark the symbol as a primitive.

Creating Symbols (Macros)

A symbol is a graphic representation of a level of hierarchy. Symbols can represent user-defined macros or design files from other sources. This section describes how to create a symbol.

1. Open a blank symbol window by clicking on **`File`**→**`Open`**, which displays the Open dialog box.
2. Change the **`Directory`** to primary (or any writable library directory) and change the **`Type`** to Symbol.
3. In the **`Symbol`** field, type in the symbol name.
4. Click on **`OK`**.

   The File Open dialog box closes and a symbol window opens.

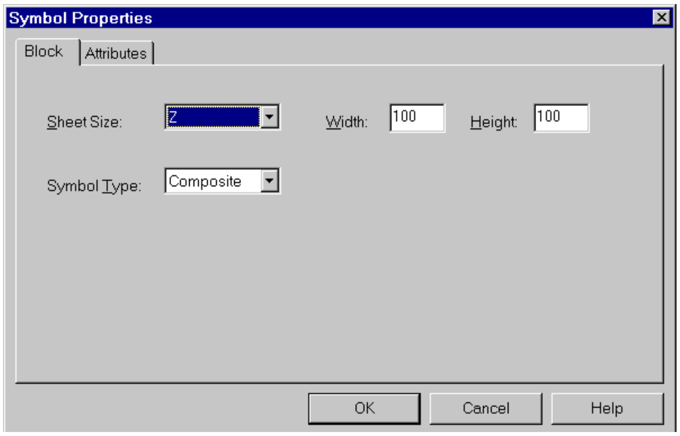
The symbol window contains a box called a block sheet that defines the perimeter of the symbol. The block sheet does not show up on the screen when the symbol is placed in a design schematic. You can see only the elements that you add to the symbol.

The initial size of the symbol, shown as the area defined by the block sheet, is 1 inch by 1 inch or 100 x 100 grid units. For most symbols, you must enlarge or reduce this default size.

Changing Symbol Size

To change the size of the symbol, follow these instructions.

1. Click the right mouse button and select Properties, or double click in the work area. The Symbol Properties dialog box appears as represented in the next figure.



Sheet Size: Z
Width: 100
Height: 100
Symbol Type: Composite

Figure 3-10 Symbol Properties Dialog Box

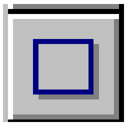
2. The default Width and Height appear as 100. Change these to your new values and click on **`OK`**.

3. Press the F4 key to resize the screen relative to the new block size.

Creating Symbol Box

Most symbols have a visible frame or symbol body to which pins attached. To add a box for a symbol, follow these steps.

1. Select the **`Add`**→**`Box`** command. See the associated icon represented next.



2. Place the cursor on the grid pip two down and two to the right of the upper left hand corner of the working area. Turn the grid pips on, if necessary, from the **`Project`**→**`Settings`** menu. Under the Project tab, check the box in the top row labeled Grid. Click on **`OK`**.

3. With the left mouse button, click and drag to the grid pip two up and two to the left of the lower right hand corner of the work area. Release the mouse button to complete the box.

The Box toolbar icon remains depressed after you define the box because ViewDraw remains in Add Box mode.

4. Press Escape or click on the Cursor icon to terminate this command.

   You can move the mouse in any direction, as long as you go from corner to opposite corner. The symbol appears similar to the one represented in the following figure.

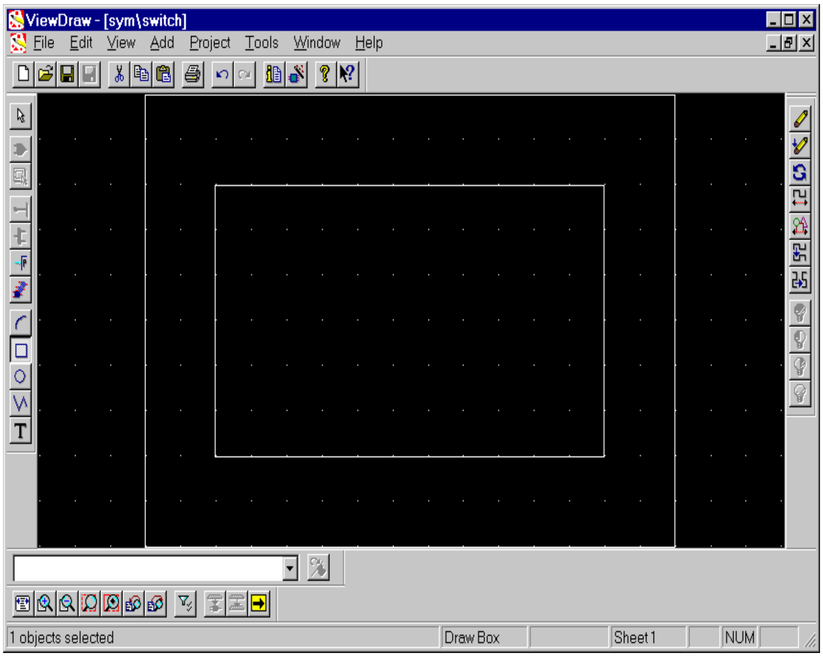


Figure 3-11 Symbol Box

A two-pip distance exists from the symbol box to the edge of the work area. While not required, this distance provides a constant symbol pin length. The Xilinx Unified Libraries follow this convention.

Adding Pins

To add pins to the symbol, follow these steps.

1. Select the **`Add`**→**`Pin`** command. See the associated icon represented in the following figure.

2. On the symbol box just drawn, single click the left mouse button at the spot where you want the pin to start. This draws a pin from this point to the edge of the work area.

   The Pin toolbar icon remains depressed after the box has been defined because ViewDraw remains in Add Pin mode. You can add multiple pins in succession without re-invoking the **Add→Pin** command.

3. Press Escape or click on the Cursor icon to terminate this command.

Adding Pin Labels

Pin labels must exactly match the labels used for the same signals in the corresponding schematic. For example, for a pin labeled “clock” on the symbol, there must exist a net labeled “clock” in the symbol’s underlying schematic. To add pin labels, follow the procedure given in the “Adding Labels” section in this chapter.

Adding Pin Attributes

You can attach attributes to pins as well as to symbols. The most common attribute applied to a pin is the PINTYPE attribute. The valid optional values for the PINTYPE attribute are IN, OUT, and BI. You can add attributes to pins using the procedure given in the “Adding Attributes” section in this chapter.

Merging Design Files from Other Sources

You can enter part of your design in some form other than schematics, such as state machine entry. You can also bring in netlist files produced by interface software from a Xilinx Alliance partner. Whatever the form of entry, you must use as the starting point for inclusion into a Viewlogic schematic design a Xilinx Netlist Format (XNF) file or an EDIF 2 0 0 netlist file. *This file must reside in the project directory.* Without an XNF file, you cannot include this portion of the design; with it, the origin of the logic becomes irrelevant.

See the "Design and Simulation Techniques" chapter for instructions about creating custom symbols for non-schematic-based modules.

Adding Labels

Labeling identifies a net, bus, component, or pin by assigning a text string to it. Bus labels are required, and labeling all nets on the schematic makes debugging easier. Label all user-created macros.

Follow the conventions described in this section when you add labels.

Naming Conventions

FPGA names for nets, buses, components, and pins must follow these conventions.

- Use only A–Z, a–z, 0–9, "_," and "-" in user-defined names. Do not include other characters in names.
- Use the Invert Sense convention. ViewDraw places a tilde (~) as the leading character of any signal name using this convention.
- Give names at least one non-numeric character.
- Restrict name length to no more than 256 characters.

Reserved Names

You cannot apply the physical names associated with every resource on every part to signals and symbols. These reserved names include CLBs, IOBs, clock buffers, BUFTs, oscillators, package pin names, CCLK, DP, GND, VCC, RT, PWRDN, and RST. Other examples include CLB names such as AA and AB, pin names such as P1 and P2, pad names such as PAD1 and PAD2, and primitive names such as TDO, BSCAN, M0, M1, M2, or STARTUP.

Net Names

ViewDraw and ViewSim fully specify hierarchical signal names; some examples follow.

- Unlabeled signals receive internal names generated automatically by ViewDraw. These names consist of a dollar sign, sheet number, "N" for net or bus, and a unique number assigned by ViewDraw for each net.

- ABC represents a labeled signal named ABC in the top-level drawing.
- $1I22\ABC represents a labeled signal named ABC underneath an unlabeled component called $1I22, where $1I22 is the symbol reference designator named with a dollar sign. This symbol reference designator derives from the sheet number (sheet 1), "I" for instance, and a unique instance number assigned by ViewDraw after each instance, in this case, "22."
- $1N118 represents net $1N118, located on sheet 1 of the root-level drawing.
- $1I5\$1N118 represents net $1N118, a net in the top-level symbol $1I5. If the net exists in a schematic represented by a symbol on the design's top level, the default signal name reflects this hierarchy.

As these examples clearly show, putting more labels in your design makes it easier to locate signals for simulation.

Component Names

To give components more meaningful names than those issued by ViewDraw, use the Label field to name symbols, just as you would nets. The following lists some examples of symbol names.

- MYSYM, a component located at the top level of the drawing
- TOP\MYSYM, a component located one level below TOP

Components with or without user-assigned labels receive names in the following manner.

top-level_instance\instance

For example, $1I3\$1I5 represents a component (instance I5) located one level below symbol $1I3.

Bus Names

To ensure that bus signals process correctly, use the following naming conventions.

- Label all buses and all nets going into buses. For example, a bus labeled A[0:2] requires nets labeled A0, A1, and A2.

- Use EDIFNETO to expand bus notation. EDIFNETO expands bus and symbol pin names into individual signal or pin names. For example, a bus labeled DATA[0:3] converts into four nets labeled DATA0, DATA1, DATA2, and DATA3.

You must consistently apply the order of bus indices for a single bus. For example, do not connect busa[0:3] to busb[3:0] at another level of your schematic unless you are deliberately reversing the bus order.

See the following table for examples of legal bus names.

Table 3-1 Legal Bus Names

Bus Name	Description
Q[0:7]	8-bit bus, signals Q0 (MSB) through Q7 (LSB)
Q[7:0]	8-bit bus, signals Q7 (MSB) through Q0 (LSB)
Q[7:0],SET,CLK	10-bit bus, signals Q7 through Q0; also signals SET and CLK
A[7:0],B[7:0]	16-bit bus, signals A7 through A0 and signals B7 through B0
DATA[0:7:2]	4-bit bus, signals DATA0, DATA2, DATA4, and DATA6
DATA[0:F / H]	16-bit bus, specified in hexadecimal (You can also specify a bus in decimal, octal, or binary.)

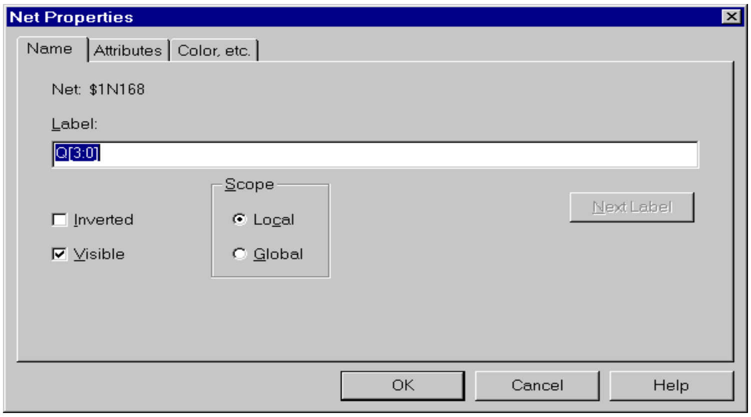
Adding a Label

Follow these steps to add a label to a net, bus, component, or pin.

1. Using the left mouse button, double click on the object you want to label. Alternatively, click the right mouse button on the object and select **`Properties`**.

   The Net Properties dialog box appears as shown in the "Net Properties Dialog Box" figure.

2. Type the name of the object in the **`Label`** field of the dialog box.



Net: $1N168

Label:

Q[3:0]

☐ Inverted

☑ Visible

Scope: [✓ Local] Global

Figure 3-12 Net Properties Dialog Box

3. Click on **OK**.

   The Net Properties dialog box closes, placing the label on the selected object.

4. To reposition the label, select the label only (not the object) with the left mouse button, then click and drag with the left mouse button to place the label.

You must label all buses and nets going into a bus. See the "Net Names" section, "Component Names" section, and "Bus Names" section earlier in this chapter for information on how to label these entities correctly.

Changing Net and Label Properties

To invoke the Net Properties dialog box, double-click on the net or bus. Use the Net Properties dialog to change some of the local properties of nets or buses, including the following.

- Net/Bus Label

  Under the Name tab, add or modify the label of net or bus in the Label field.

- Label Visibility and Sense

  Under the Name tab, you can use check boxes to define the label's visibility and sense. De-selecting the Visible box renders the label invisible. Selecting the Inverted box causes an overscore to appear over the signal name. Xilinx does not support the use of the Inverted signal option.

- Color and Style of the Net

  You can find these options under the Color, Etc. tab.

- Net Attributes and Visibility

  Under the Attributes tab, you can add or modify net and bus attributes. To change the visibility of an attribute, select the attribute, then change the Visibility field. Click **OK** to save the change. See the "Adding Attributes" section of this chapter for more information.

Access other properties through the Label Properties dialog box. To invoke this dialog box, double click on the label (make sure the net or bus itself is not selected). In addition to the Label, Visibility, Color, and Sense parameters, you can modify the following.

- Text Size

  Under the Name tab, change the size of the text in the Size field.

- Text Origin

  Under the Color, Etc. tab, choose the location of text origin of this label.

- Label Color and Font

  Under the Color, Etc. tab, choose the color and Font for this label.

To apply any changes made in the Net Properties or Label Properties dialog boxes, click on **OK**. To exit the dialog box without making schematic changes, click on Cancel.

You can set the default value of many of these properties from the **Project**→**Settings** menu selection.

Adding Attributes

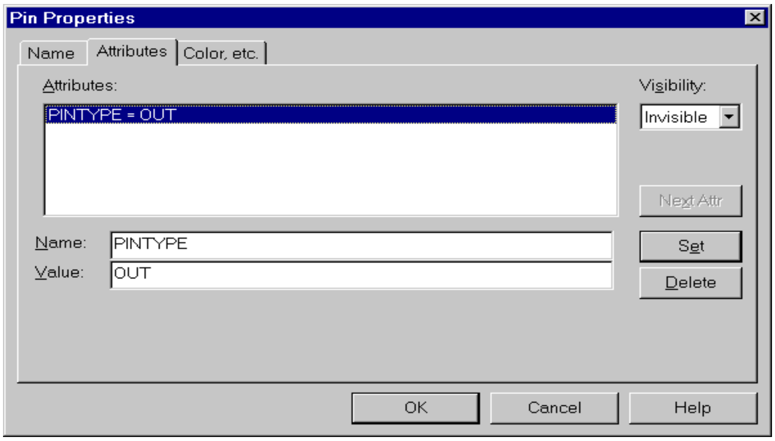
Attributes are instructions placed on symbols or nets in an FPGA or CPLD schematic to indicate their placement, implementation, naming, directionality, and so forth. You can find the list of attributes that you can place on the components in your Viewlogic schematic in the *Libraries Guide*.

To assign attributes to nets, buses, components, or pins, complete the following steps. You must open the Net Properties, Component Properties, or Pin Properties dialog box, represented in the "Adding Attributes to a Pin" figure.

1. Open the Properties dialog boxes by either double-clicking the left mouse button on the item or single-clicking the right mouse button on the item and selecting Properties.
2. Under the Attributes tab, fill in the Name and Value fields. For example, an attribute on a symbol pin would be as follows.

   Name: PINTYPE

   Type: IN
3. To make this attribute invisible, change the Visibility field to Invisible. Click on Set to accept the attribute.
4. After you have added all the needed attributes, click on **OK**.



Attributes:
PINTYPE = OUT

Visibility: Invisible

Name: PINTYPE
Value: OUT

Figure 3-13 Adding Attributes to a Pin

5. To reposition visible attributes, select the attribute only with a single left mouse button click. Click and drag with the left mouse button to the desired location.

Saving Schematics

To save the schematic, select the **File**→**Save + Check** command.

This command checks the schematic for any errors and then saves the schematic. If the message tracker dialog box appears, correct the schematic accordingly and re-save when finished. The following message displays on the status bar at the bottom of the schematic after you make all corrections.

```
Check complete. 0 error(s) and 0 warning(s) in project
primary:schematic_name.
```

To save the schematic to a file of a different name, click on **File→Save Copy As**.

ViewDraw prompts you to specify the schematic name and the sheet number. If the file already exists, ViewDraw prompts you to verify that you want to overwrite it.

Use the Save Copy As command to rename a schematic or symbol. The Viewlogic licensing scheme saves each schematic sheet and symbol with a specific license number. This number refers to the key number and the name of the schematic or symbol. If you rename or copy the file outside of ViewDraw, the license number no longer matches, and licensing errors occur when ViewDraw opens the newly copied schematic or symbol.

Closing Schematics

To close a design, click on **File→Close**.

Converting a Design

The Unified Libraries provide a means of effortless migration between Xilinx architectures. Because the shared components in the various libraries have the same names and symbol definitions, you need only run Altran to convert a design from one family to another. Altran changes the current library alias of each primitive used in the schematics of the current project to the desired target library alias.

As an example, designing a circuit that targets an XC3000 part requires each primitive in your schematic reference the XC3000 library using the (XC3000) alias. Later in the design cycle, to take advantage of the features offered by the XC4000E family of devices, you do not have modify the schematic manually to target this new family. Instead, you can use Altran to convert the primitive aliases from (XC3000) to (XC4000E). Now all the primitives reference the XC4000E library.

To convert a design in the Viewlogic environment, use the Altran program. After you use Altran to convert the design schematics, you must update the Workview Office Project Manager, adding the target library to the library list, as the following describes.

1. Close all Workview Office tools except the Toolbar.
2. Open an MS-DOS session and change the current directory to the project directory.
3. Run the ALTRAN command with the following syntax.

   `altran –l library` *old_alias*`=`*new_alias*

   For example, to change a project from the XC4000E family to the XC4000X, type the following.

   ```
   altran –l primary xc4000e=xc4000x
   ```

   Altran changes the library aliases in all of the schematic sheets in the specified library directory. Altran also modifies the alias of the targeted library in the Viewdraw.ini file.

4. After running ALTRAN, open the Workview Office Project Manager. Note that the libraries do not change. Select **`Project→Libraries`** to modify the Library Search Order.
5. Select the Xilinx family library. In the Path field, change the path so it points to the XC9000 library directory. In the Alias field, change the value to XC9000. Click on **Change**, then **OK**.
6. Save the changes in the Project Manager.

Now you can re-open ViewDraw and open the desired schematic. Only components in the current technology that have equivalents in the target technology translate. Those components that do not have an equivalent in the target technology do not appear in the converted schematics. For example, an XC4000E RAM component has no equivalent in an XC5200 device. Therefore, if you convert an XC4000E design with RAM components to an XC5200 design, the RAM components appear as white boxes on the converted schematic. Modify these portions of your design manually.

Chapter 4

Functional Simulation

Functional simulation provides an effective means of identifying logic errors in designs not yet implemented into a Xilinx device. The simulator tests the logic in the design using unit delays instead of timing information, not available because placing and routing has yet to occur on the design. Simulate the functionality after entering your design to verify correct circuit logic before mapping, placing, and routing take place. Finding errors before routing your design saves debugging time later in the design process.

This chapter describes how to prepare a simulation network for a functional simulation in the Viewlogic simulation environment. This chapter also describes how to load VWaves to view the simulation signals in a waveform format.

However, this chapter does not document specific Viewlogic commands for ViewSim, Speedwave, or VWaves. For information regarding the use of these tools, consult the online Viewlogic help files accessible from all of these tools. This chapter contains these sections.

Understanding the Simulation Procedure

Perform the initial simulation by issuing the simulation commands manually instead of using a command file. Then, after defining the sequence of commands, save the log of the session in a command file and use this file to re-simulate the design whenever you make a design change. The sequence of steps in this chapter reflects this methodology.

The following shows a typical procedure for performing a functional simulation.

1. Create the simulation network (VSM file).
2. Start ViewSim.
3. Load the VSM file into ViewSim.
4. Simulate the device's startup sequence.
5. Manually enter the simulation commands.
6. Run a command file (optional).
7. Start VWaves.
8. View the waveforms produced by the simulation.
9. Repeat steps 5, 6, and 8 until the design is verified.

The rest of the chapter discusses these steps in detail.

Preparing the Design for Functional Simulation

As a first step in the functional simulation process, create the simulation network (VSM file) and load it into ViewSim to simulate the design. The procedure for preparing for functional simulation on a Xilinx design depends on which components you use to enter the design.

You cannot simulate designs that target the Virtex family after NGDBuild in Viewsim.

Using Category A and B Designs

This section describes the two categories of designs.

Category A Designs

This category includes designs that contain modules with non-schematic models (such as ABEL modules) and components with parameterized functionality (for example, the behavior of a ROM component depends on the assigned INIT attribute).

A complete list of such modules follows.

- ABEL modules
- Modules defined only by an EDIF or XNF netlist, without a corresponding source schematic
- XC3000 CLB and IOB primitives (defined by attributes)
- ROM primitives (contents defined by INIT attribute)
- XC4000 series RAM primitives (initial contents defined by INIT attribute or by default for architecture); also LogiBLOX RAM modules for these families (if using EDIF models)

Category B Designs

This category includes all designs not included in Category A. Category B designs do not contain any non-schematic modules or parameterized components. The presence of LogiBLOX modules does not exclude a design from this category, except for RAM modules.

You can simulate Virtex designs that fall under Category B except for the LUT, Block Ram, Shift Register, and CLKDLL library components.

Functionally Simulating Category A Designs

In order to functionally simulate a design that contains modules with non-schematic models and components with parameterized functionality, compile the design to a single netlist file (NGD). You must run two programs to compile the design to this netlist, then run two to bring the design back into the Viewlogic environment for the simulation.

Running the Xilinx Functional Simulation Interface

Part of the installation and setup process includes running the Custmenu command. This command sets up five menu choices within ViewDraw, including one entitled “Xilinx Functional Simulation.”

With the top-level schematic open in ViewDraw, select this menu item. When the interface opens, select the family for this design and click on **`OK`**.

The Functional Simulation GUI runs five programs.

- EDIFNETO writes an EDIF file for the design
- NGDBuild combines the design and uncompiled modules
- NGD2EDIF creates a complete EDIF file for the design
- EDIFNETI reads the new EDIF file back into Viewlogic
- VSM creates a ViewSim netlist, ready for simulation

The "Using Powerview to Create the Functional Simulation Netlist for Category A Designs" section details the specific command lines that run within the Functional Simulation GUI.

The Functional Simulation GUI produces a func_sim.vsm file. The name of this file differs from the name of the original schematic, so you need to tell Viewsim which schematic to annotate. See the "Annotating Values to Original Schematic" section for details.

After creating the VSM file you can move on to the "Loading the Design into the Viewlogic Simulator" section.

Using Powerview to Create the Functional Simulation Netlist for Category A Designs

To create a VSM file for Category A designs, first write out an EDIF netlist from Powerview using either the Netlist Out button in the Powerview Cockpit or from a UNIX prompt. The EDIFNETO tool translates a Viewlogic design into an EDIF 2.0.0 netlist.

Double-click on the Netlist Out button from the Powerview Cockpit, then fill out the following two fields.

1. Enter the name of your top-level design in the Wire file name field.
2. Set the Level field to *xilinx* so the EDIF Netlist Writer knows how far to descend into the schematic hierarchy. Setting the level to *xilinx* tells EDIFNETO to stop at Xilinx primitives.

Click on **`OK`** or **`Apply`** to write the *design*.edn file.

This command line syntax follows.

edifneto –l xilinx *design*

Use the name of the top-level design for the *design* parameter.

Creating a Flattened Netlist

After writing the .edn file, compile the design to one complete and flattened NGD file. The syntax for NGDBUILD follows.

ngdbuild –p part *design***.edn**

This Xilinx program reads the EDIF netlist, expands any LogiBLOX modules, and generates gate models for non-schematic modules. It produces a *design*.ngd file.

Use the –p option only if there a part type is not specified in the schematic. You need specify only the architecture family if you do not yet know the exact device and package information. For example, you can specify “xc4000x” rather than “XC4028XPG299-3.”

Creating a New EDIF File

After creating an NGD file to represent your entire flattened design, you must create a new EDIF file to send back to the Viewlogic tools for the functional simulation. Use NGD2EDIF with the following syntax.

ngd2edif –v viewlog *design***.ngd func_sim.edn**

This Xilinx program generates a gate-level EDIF netlist for the complete expanded design. All design logic is expressed in terms of the Xilinx simprims library components.

The –v option tells NGD2EDIF to perform some Viewlogic-specific processing when generating the EDIF netlist.

The EDIF file is named func_sim.edn, not the original design name to avoid overwriting the original EDIF netlist and the original WIR file(s) in the next step. The name “func_sim.edn” is not required, but it should be a name different from the design name.

If the design contains RAM elements, and if the target family supports initial values on RAM, NGD2EDIF also generates a func_sim.xmm file. This file contains ViewSim LOADM commands to model the initial contents, discussed in the “Loading XMM Files” section.

Reading in the EDIF Netlist

Read this new EDIF file back into the Powerview environment. Double-click on the Netlist In button from the Powerview Cockpit.

Fill in the EDIF netlist file field if the EDIF file resides in the project directory. Otherwise, ensure the Pathname for output files field points to the project directory.

Click on **OK** or **Apply** to read in the func_sim.edn file.

The command line syntax follows.

```
edifneti func_sim.edn
```

EDIFNETI reads the EDIF netlist generated by NGD2EDIF and produces WIR files to represent the design. Because the EDIF netlist is hierarchical, EDIFNETI generates one WIR file for each level of hierarchy. Lower-level WIR files are named xba1.1, xba2.1 and so on, to avoid conflicts with the original design WIR files.

Creating the VSM Netlist

After reading in the EDIF netlist, in the Powerview Cockpit, double-click the VSM icon. This opens the ViewSim Wirelister. Use the following steps to create a functional simulation netlist from the func_sim.edn file.

1. In the Design Name field, enter the func_sim file that you just read in using EDIFNETI.

Note: Do not select the *design*.1 or *design*.edn file as the input, as this uses the uncompiled schematic design instead of the compiled version that you just created.

2. Click on **OK** or **Apply** to create the func_sim.vsm file.

This Viewlogic program reads the WIR files generated by EDIFNETI and produces a single VSM file for use by Viewsim.

Enter the following on a UNIX command line to run this command.

```
vsm func_sim
```

Functionally Simulating Category B Designs

Preparing a Category B design for functional simulation requires running the VIEWlogic VSM program. You call this program from

ViewDraw or from the Workview Office Toolbar. How you do this depends on whether or not LogiBLOX modules were used, and what type of models it created.

In ViewDraw, select **`Tools`**→**`Create Digital Netlist`**, or from the Workview Office Toolbar, select the VSM icon, shown in the following figure.

This opens the ViewSim Wirelister. Follow the following steps to create a functional simulation netlist from your schematic.

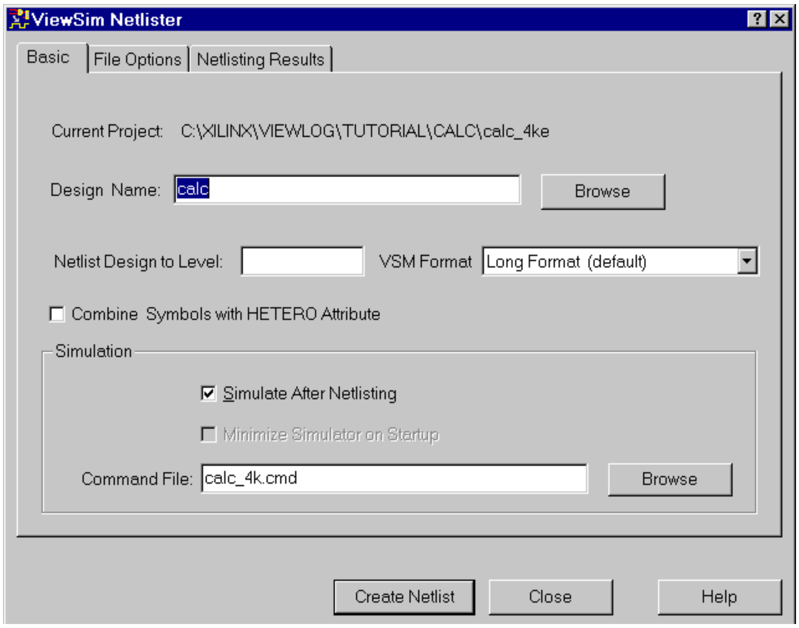
1. Under the Basic tab, fill in the Design Name field. Select the *design*.1 file found in the sch directory of your project. Use the **`Browse`** button if necessary.

   After you complete the Design Name field, go to the File Options tab and fill in the **`VSM File Name`** field with *design*.vsm. Optionally change this to func_sim.vsm.

2. Check the **`Invoke Simulator when Finished`** checkbox. This opens the Viewlogic simulator after you create the VSM file.

3. Fill in the **`Command File`** field if you use a command file. You can use the **`Browse`** button if necessary.

At this point, the ViewSim Wirelister appears similar to the following figure.



Current Project: C:\XILINX\VIEWLOG\TUTORIAL\CALC\calc_4ke

Design Name: calc

Netlist Design to Level:

VSM Format: Long Format (default)

☐ Combine Symbols with HETERO Attribute

☑ Simulate After Netlisting

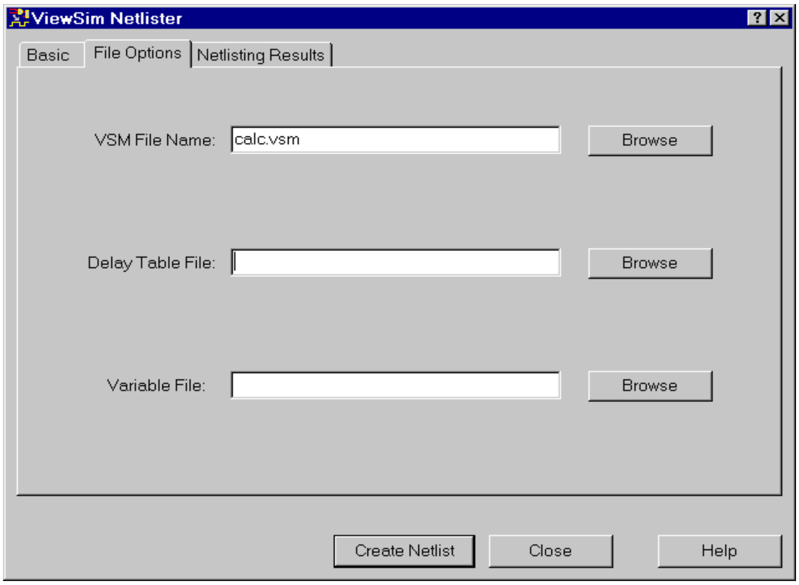
☐ Minimize Simulator on Startup

Command File: calc_4k.cmd

Figure 4-1 Completed ViewSim Wirelister for Category B

If the design contains no LogiBLOX modules, or if LogiBLOX created EDIF models for its modules, click **OK** to start the ViewSim Wirelisting flow.

If LogiBLOX created VHDL models for its modules, you need to fill in one more field. Under the Advanced tab, fill in the Wirelist Design to Level field with *vhdl*, as shown in the next figure.


ViewSim Netlister

VSM File Name: calc.vsm

Delay Table File:

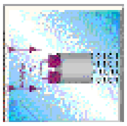
Variable File:

Figure 4-2 Advanced Tab for VHDL Models

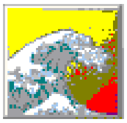
The vhdl argument tells VSM to netlist down to LogiBLOX modules. The actual VHDL modules will be incorporated by the simulator.

Loading the Design into the Viewlogic Simulator

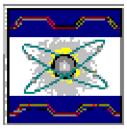
Checking the **`Invoke Simulator when Finished`** checkbox when you create the VSM file causes the Viewlogic simulator to automatically open. Otherwise, to open the Viewlogic simulation environment, click on the ViewSim icon, shown in the following figure.



If your design contains LogiBLOX elements with VHDL models, you must use the Speedwave simulator. Click on the Speedwave icon shown in the next figure. This simulator requires an extended license to use.



Another option for designs containing VHDL models involves using Digital Fusion. Click on the Digital Fusion icon shown in the following figure. This also requires an extended license.



Digital Fusion is the top-level wrapper that incorporates the Viewlogic Digital Simulation Tools. The tools applicable to Xilinx designs are ViewSim, a gate-level simulation tool, and Speedwave, a gate-level and VHDL simulation tool. These tools, along with VCS, a Verilog simulation tool, and the analog simulation tools, comprise the Viewlogic Integrated Simulation Environment (ISE). These programs provide a common user interface and set of commands for any simulation flow.

To load a VSM file into any of the Viewlogic simulation tools, select **`File`→`Load ViewSim Netlist`**. Navigate to your project directory and select func_sim.vsm.

Loading XMM Files

If the design contains RAM elements, and if the target family supports initial values on RAM, the flow described in the "Functionally Simulating Category A Designs"section generates a command file with the .xmm extension. Use this command file in the Viewlogic simulator. The file contains LOADM commands for each Viewsim RAM built-in primitive in the design.

To initialize the RAMs, execute the .xmm file from the Viewlogic simulator prompt by entering the following.

```
execute func_sim.xmm
```

You can include this statement in your design command file so that the initial contents load automatically. The Viewlogic simulator's restart command clears the RAM contents, so if you issue a restart in your command file, execute the .xmm file after that.

Executing Global Reset

After loading the design into the simulation environment and initializing the RAM / ROM components and external signals, you need to execute a global set / reset (GSR) command. This mimics the startup sequence of the FPGA or CPLD that you are simulating.

For example, for an XC4000 series device, run the following commands.

```
h GSR

sim 1000

l GSR
```

The names and polarities of the GSR signals appear in the following table.

Table 4-1 GSR Signal Names and Polarities

Family	Global Set/Reset Signal	Polarity
XC3x00A / L	GR	Active-Low
XC4000E / L XC400EX / XL / XLA / XLT / XV	GSR	Active-High
Spartan	GSR	Active-High
SpartanXL	GSR	Active-High
Virtex	GSR	Active-High
XC5200	GR	Active-High
XC9500 / XL / XV	PRLD	Active-High

Annotating Values to the Original Schematic

Normally, node values calculated in the simulator annotate automatically to the Viewdraw schematic of the same name.

However, because you gave the name func_sim to your functional simulation netlist, the VSM file created has a different name than the original schematic (to avoid conflicts with the original EDIF and WIR files). You must explicitly tell the simulator to annotate values to the original schematic, using the following command from the Viewlogic simulator.

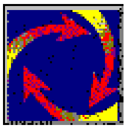
```
schemnam <design>
```

The parameter <*design*> is the top-level schematic name. Values should then appear in the Viewdraw window.

Invoking VWaves

To view a large number of signals simultaneously, open a waveform window in Viewsim. Use the **Signals→Display in VWaves** command in the Viewlogic simulator to create a .vcd file that contains the signals and vectors you want to view. The wave command also opens VWaves. You can also follow these steps.

1. From the Workview Office Toolbar, click on the VWaves icon, represented in the following figure.

2. The Open Waveforms File dialog box opens, pointing to your project directory to select the waveform display file.

   The name of the waveform file is the name you used when you added the signals to the waveform. The default name is *design*.vcd.

3. Click on **OK**.

   The Open dialog box closes, and the specified waveform display file opens.

Chapter 5

Implementing a Design

After completing functional simulation, you can implement your design using Design Manager. Design Manager first reads in the design netlist in EDIF format, then interfaces with the Flow Engine to implement the design. The Flow Engine optimizes, places, and routes the design, creates timing simulation data, and creates physical design data for downloading. This chapter describes how to use Design Manager and the Flow Engine to translate and implement your design.

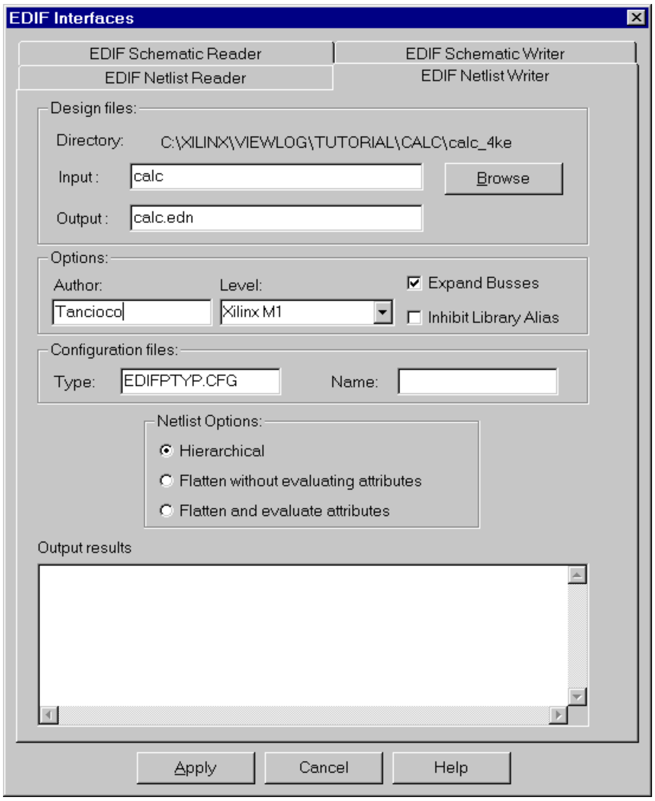
In a project, you can use Design Manager to manage implementation versions and revisions. Each time that you change a schematic, you create a new implementation version. For each implementation, you can create multiple design revisions, one for each pass through the implementation software. For more information about Design Manager and how to manage versions and revisions, see the *Design Manager/Flow Engine Guide*.

This chapter contains these sections.

Writing Out an EDIF Netlist

Before you can start Design Manager, you must write out an EDIF netlist from Workview Office. If you have a current *design*.edn file from functional simulation, you do not have to repeat this step. With the top-level schematic open in ViewDraw, select **`Tools`**→**`Write Xilinx EDIF`** to create the *design*.edn file.

As an alternate method of creating the EDIF file for the design, use the EDIF Interfaces GUI. Click on the EDIF button on the Workview Office Toolbar. A dialog box similar to the one shown in the following figure appears. One of the programs under this Toolbar program is EDIFNETO, which translates a Viewlogic schematic into an EDIF 2.0.0 netlist. Select the EDIF Netlist Writer tab.



Design files:

Directory: C:\XILINX\VIEWLOG\TUTORIAL\CALC\calc_4ke

Input : calc

Output : calc.edn

Options:

Author: Tancioco

Level: Xilinx M1

[✓] Expand Busses

[ ] Inhibit Library Alias

Configuration files:

Type: EDIFPTYP.CFG

Name:

Netlist Options:

(●) Hierarchical

( ) Flatten without evaluating attributes

( ) Flatten and evaluate attributes

Figure 5-1 EDIF Netlist Writer

At a minimum, you must fill in the following three fields.

- The Input field, where you enter your top level design. Use the **`Browse`** button to select the discrete path to the *design*.1 file.
- The Output field, where you enter *design*.edn as soon as you fill in the Input field. You only need to change this if you do not want *design*.edn as the output of EDIFNETO.

  Write the EDIF file into the Viewlogic project directory (the default), because the Xilinx Design Manager assumes that the directory containing the top-level EDIF is the source design directory.
- The Level field, found in the Options section. You must set this to *xilinx* so the EDIF Netlist Writer knows how far to descend into the hierarchy. Setting the level set to *xilinx* tells EDIFNETO to stop at Xilinx primitives.

  Click on **`Apply`** to write the *design*.edn file.

Invoking Design Manager

Invoke Design Manager from the Xilinx program group using the following steps.

1. Open Design Manager from the Xilinx M1 program group. Select **`Start`**→**`Programs`**→**`Xilinx M1`** from the Windows 95/NT toolbar.
2. Click on the Design Manager Icon.

The Design Manager window appears, as shown in the following figure.

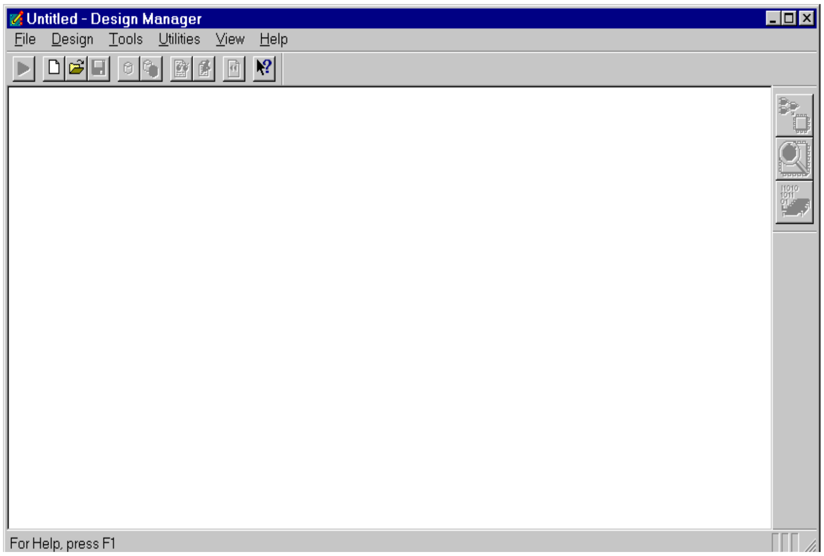

Untitled - Design Manager
File Design Tools Utilities View Help
For Help, press F1

Figure 5-2 Design Manager Window

If you named your project, the project name appears on the title bar of Design Manager. The full set of available menus and icons does not become active until you open or create a project. The toolbar display icons that perform the same functions as the most commonly used menu commands. Move the cursor onto an icon to determine the functionality of the icon.

The Project View section of the window contains a graphic representation of the versions and revisions of the design. The status bar at the bottom of the window displays the family, part number, version number, and revision number of the current version.

Creating the Xilinx Project

Before you can begin implementing a design, you must create its Xilinx project, a different process than creating the design's Viewlogic project. The Xilinx project directories contain version and revision information for multiple runs of your design through the Xilinx implementation tools.

Follow these steps to create the Xilinx project.

1. Select the **`File`**→**`New Project`** command.

The New Project dialog box appears, as shown in the next figure.

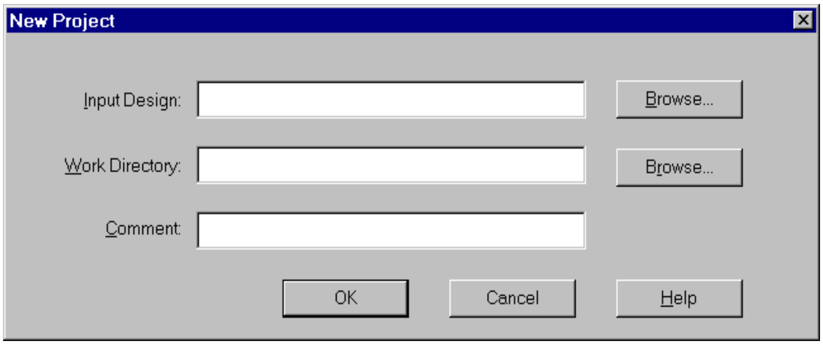


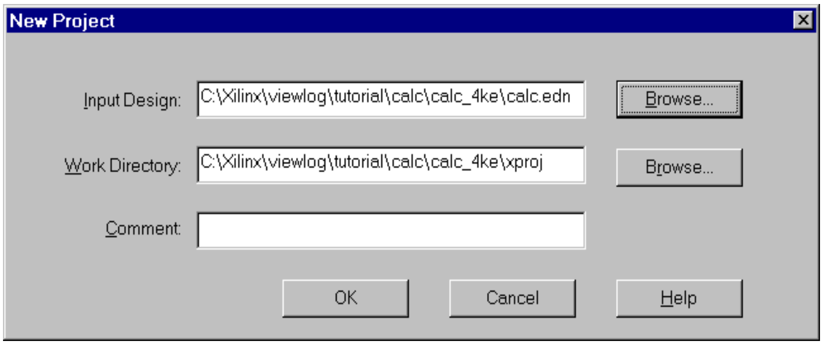
Figure 5-3 New Project Dialog Box

2. Specify the EDIF file for the top-level design name in the Input Design field, or click on the **`Browse`** button.

   The Browse dialog box appears, allowing you to select EDIF files with varying acceptable extension formats. The Viewlogic EDIF Netlist Writer uses the EDN extension.

3. Select the input file, then click on **`OK`**.

   The Browse dialog box now closes, and the New Project dialog box is updated with the selected file, illustrated in the next figure.



Input Design: C:\Xilinx\viewlog\tutorial\calc\calc_4ke\calc.edn

Work Directory: C:\Xilinx\viewlog\tutorial\calc\calc_4ke\xproj

Figure 5-4 Updated New Project Dialog Box

The Work Directory field now displays the default value of the project directory. The project directory, called xproject, goes in the working directory and contains the files created by Design Manager after design versions and revisions compile.

4. To add comments for this project, fill in the optional Comment field.

5. Click on **OK**.

   The design loads into Design Manager. The design name and any comments list as shown in the following figure.

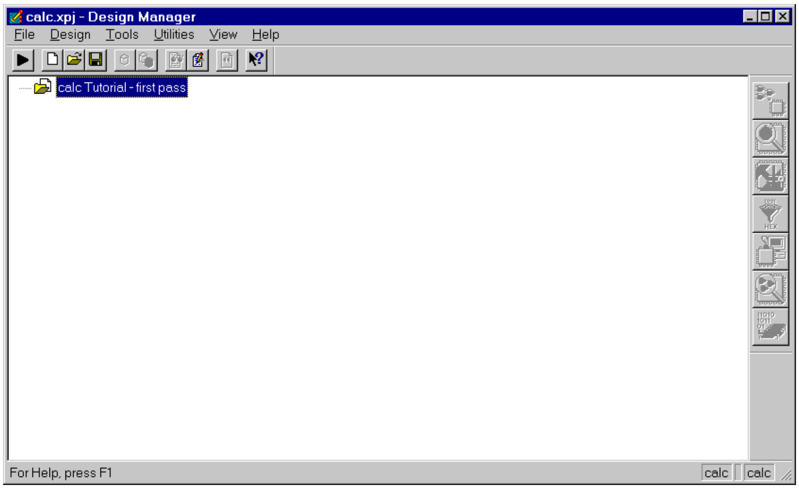
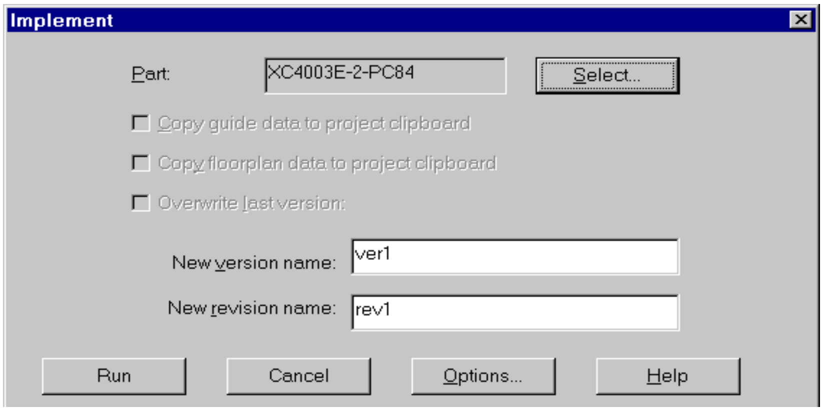

calc.xpj - Design Manager
File Design Tools Utilities View Help
calc Tutorial - first pass
For Help, press F1
calc
calc

Figure 5-5 Design Manager with Initial Project

Implementing a Design

The following steps show you how to implement your design.

1. In the Design Manager window, select **`Design`**→**`Implement`**. The Implement Dialog Box appears, as shown in the following figure.


Implement
Part:
Select...
Copy guide data to project clipboard
Copy floorplan data to project clipboard
Overwrite last version:
Run
Cancel
Options...
Help

Part: XC4003E-2-PC84

New version name: ver1

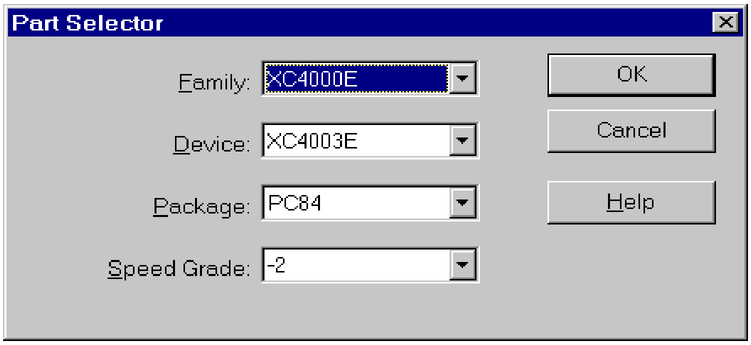
New revision name: rev1

Figure 5-6 Implement Dialog Box

The New Version Name field displays the version number of the design. For the first translation, “ver1” appears by default. With subsequent translations of the schematic design, the version number automatically increments.

The New Revisions Name field displays the revision number of the design. For the first revision, “rev1” appears by default. With subsequent revisions of the same implementation of the design, the revision number automatically increments.

2. If the input design netlist does not specify a part, fill in the Part field. Click on the Select button to choose the target device. The Part Selector Dialog Box appears, as shown in the following figure.



Family: XC4000E

Device: XC4003E

Package: PC84

Speed Grade: -2

Figure 5-7 Part Selector Dialog Box

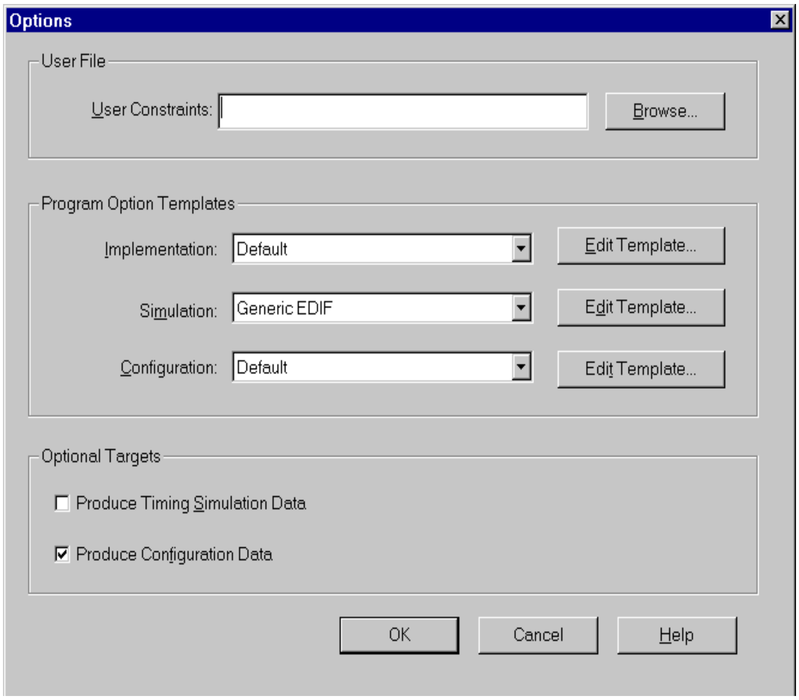
In this dialog box, the Device field reflects only the parts for the family selected in the Family field. Similarly, the Package field displays only those packages suitable for the device selected. The Speed Grade field displays only those speed grades for the part and package selected.

3. Select the family, device, package, and speed grade appropriate for your device. Click on **OK**.

Setting up the Implementation Options

Design Manager uses the Flow Engine to implement a design. Before invoking the Flow Engine, however, set up any options you want applied to this design.

To modify the implementation options used by the Flow Engine, click on the **Options** button. The Options dialog box opens, as shown in the next figure.



Options

User File

User Constraints:

Browse...

Program Option Templates

Implementation: Default

Edit Template...

Simulation: Generic EDIF

Edit Template...

Configuration: Default

Edit Template...

Optional Targets

☐ Produce Timing Simulation Data

☑ Produce Configuration Data

OK

Cancel

Help

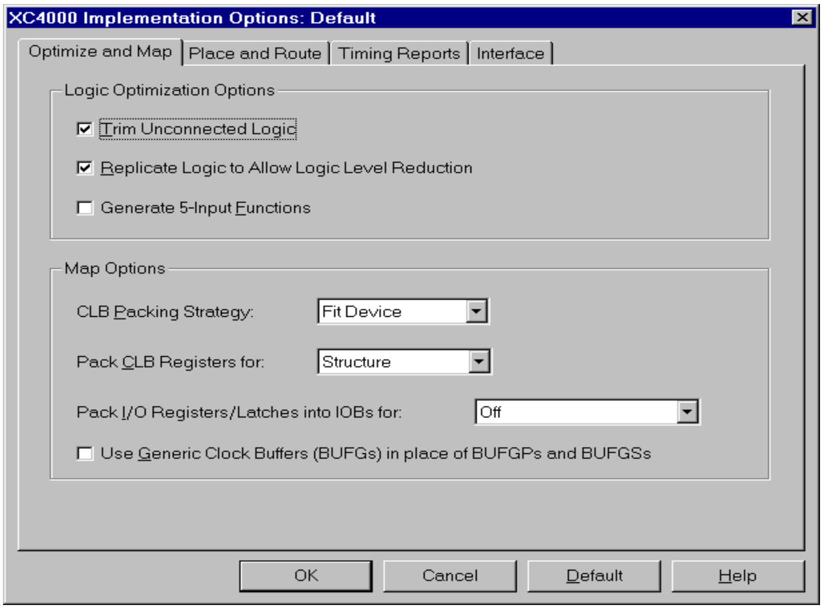
Figure 5-8 Options Dialog Box

From this dialog box, you have access to a number of Flow Engine options.

- The Constraints option allows you to specify a .ucf file. This User Constraints File contains placement and performance requirements for the design.
- The Optional Targets section gives you a choice of output targets to produce. The “Produce Configuration Data” option creates the bitstream used to configure the Xilinx device. Timing simulation requires the “Produce Timing Simulation Data” option; see the “Timing Simulation” chapter for more information about the timing simulation flow.
- In the Program Option Templates section, you can access three sets of options covering implementation, simulation, and configuration. Click on the **`Edit Template...`** button to access these dialog boxes or select from the pull-down menu.

Setting Implementation Options

After clicking on the **`Edit Template...`** button next to Implementation, the Implementation Options dialog box opens. This dialog box contains implementation options specific to the specified family of Xilinx device. The following figure shows this dialog box for the XC4000 family.



Logic Optimization Options

- [✓] Trim Unconnected Logic
- [✓] Replicate Logic to Allow Logic Level Reduction
- [ ] Generate 5-Input Functions

Map Options

CLB Packing Strategy: Fit Device

Pack CLB Registers for: Structure

Pack I/O Registers/Latches into IOBs for: Off

- [ ] Use Generic Clock Buffers (BUFGs) in place of BUFGPs and BUFGSs

Figure 5-9 Implementation Options Dialog Box

This dialog box allows access to the more common options used in the Flow Engine. The tabs at the top of this window separate the options into four subdivisions.

- The Optimize & Map tab contains options for NGDBuild and MAP.
- The Place & Route tab contains options for PAR.
- The Time tab contains options for TRACE.
- The Interface tab contains options for Macro Search Path, Rules File, and Create I/O Pads from Ports.

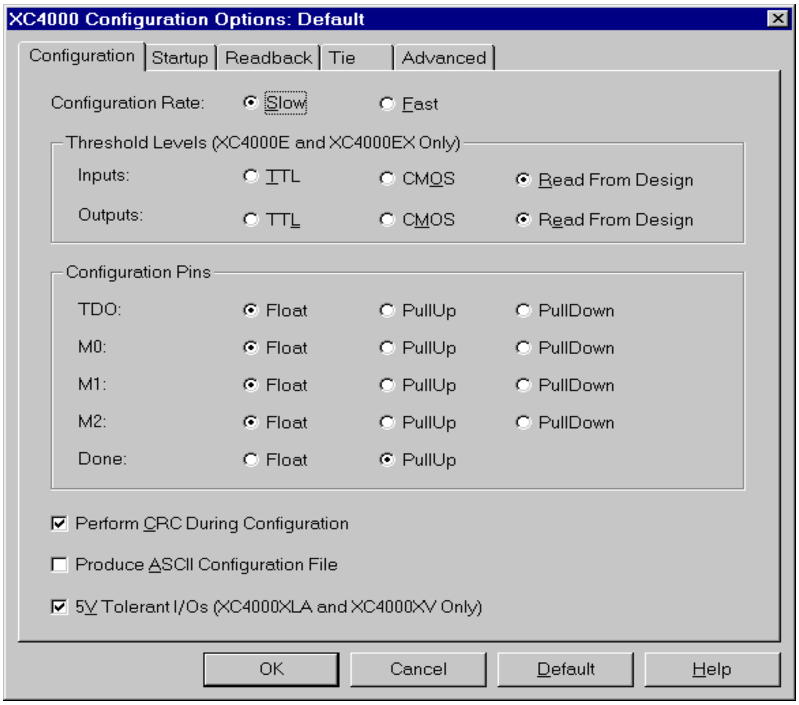
To the right of the Simulation Program Option Template is a pull-down menu. Click and select Viewsim EDIF.

Consult the *Design Manager/Flow Engine Guide* for more detailed information about the options available in the Implementation Options dialog box.

Click on **`OK`** to save these settings or **`Cancel`** to discard any changes. Either button closes the Implementation Options dialog box.

Setting Configuration Options

After clicking on the **`Edit Template...`** button next to Configuration, the Configuration Options dialog box opens. This dialog box contains configuration options specific to the specified family of Xilinx devices. The next figure shows this dialog box for the XC4000 family.



Configuration Rate: [✓ Slow] Fast

Threshold Levels (XC4000E and XC4000EX Only)

Inputs:	TTL	CMOS	[✓ Read From Design]
Outputs:	TTL	CMOS	[✓ Read From Design]

Configuration Pins

TDO:	[✓ Float]	PullUp	PullDown
M0:	[✓ Float]	PullUp	PullDown
M1:	[✓ Float]	PullUp	PullDown
M2:	[✓ Float]	PullUp	PullDown
Done:	Float	[✓ PullUp]	

[✓] Perform CRC During Configuration

[ ] Produce ASCII Configuration File

[✓] 5V Tolerant I/Os (XC4000XLA and XC4000XV Only)

Figure 5-10 Configuration Options Dialog Box

This dialog box allows access to the more common options used when producing the configuration bitstream. The tabs at the top of this window separate the options into four subdivisions.

- The Configuration tab contains options for threshold levels and the configuration pins.
- The Startup tab contains options that define the startup sequence of the device.
- The Readback tab contains options to enable the readback feature.
- The Tie tab contains an option to enable the tie feature.
- The Advanced tab contains advanced options per device.

Consult the *Design Manager/Flow Engine Guide* for more detailed information about the options available in the Configuration Options dialog box.

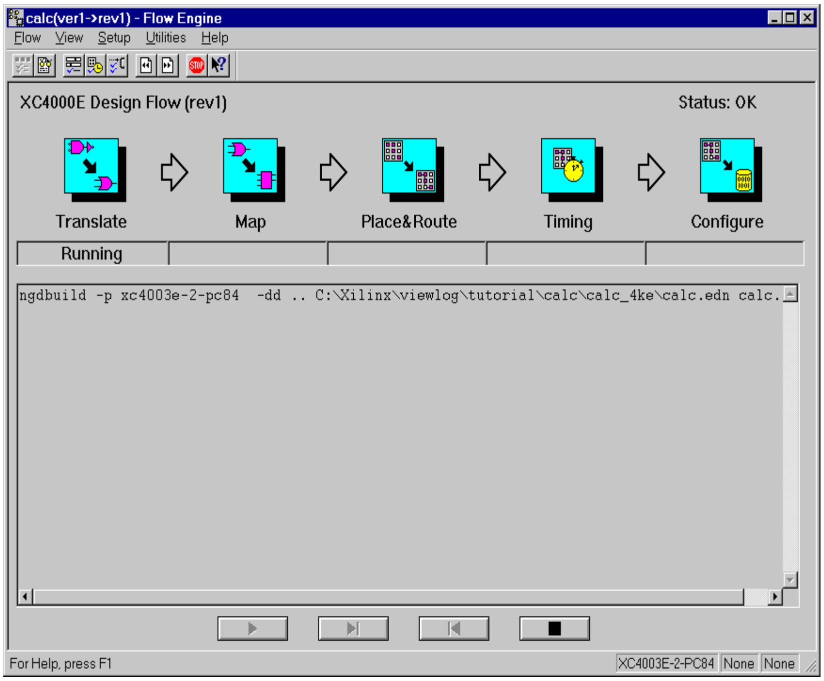
For both of these dialog boxes, the **`Default`** button resets all of the options in that dialog box to their default settings. Use the **`Help`** button to access the Design Manager online help.

Click on **`OK`** to save these settings or **`Cancel`** to discard any changes. Either button closes the Configuration Options dialog box.

In the Options dialog box, click on **`OK`** to return to the Implement dialog box.

Running the Flow Engine

Now that all the options have been set, click on **`Run`** in the Implement dialog box. The Flow Engine window opens and begins running. Refer to the following figure.



```
ngdbuild -p xc4003e-2-pc84  -dd .. C:\Xilinx\viewlog\tutorial\calc\calc_4ke\calc.edn calc.
```

Figure 5-11 Flow Engine Window

The Flow Engine creates a placed and routed NCD file, a bitstream used to configure the Xilinx device, and an EDIF file used to perform a timing simulation. For more information about the functionality and use of the Flow Engine, consult the *Design Manager/Flow Engine Reference/User Guide*.

Using the Report Browser

After you complete the steps in the Flow Engine, you can generate reports. You can access these reports via the Report Browser. To start Report Browser, from either the Design Manager or the Flow Engine select **`Utilities`**→**`Report Browser`**. The next figure illustrates the Report Browser after the Flow Engine completes processing.

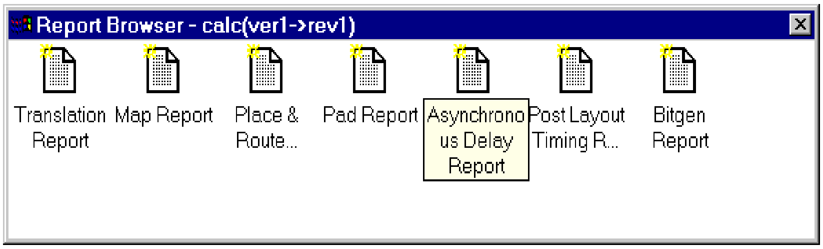


Figure 5-12 Report Browser

The Report Browser keeps track of available reports but also indicates if those reports are new. Unread reports get a yellow star in the upper left corner of the report icon.

1. To review a report, double-click on its icon with the left mouse button.

   The report opens in a text editor so you can review it. You can also edit the report and save it in a new area. After you view a report, the yellow star in the upper left corner disappears.

2. To close the Report Browser, click on the X box in the upper right-hand corner.

To change the program used to view the reports, select **`File`**→**`Preferences`** from the Design Manager. You can use wordpad.exe, found in the Program Files \ Accessories directory.

Exporting Revision Data

As you re-implement designs and create new revisions, only the most recent revision data gets placed in the Viewlogic project directory. To place data from previous revisions in this directory, export the data using the Xilinx Design Manager. Otherwise, you have to push into the design version and revision directories to access the simulation and configuration files.

The following figure displays a design version containing four revisions, including two configured revisions. Rev2 data, the most recently produced, resides in the Viewlogic project directory. If you want to use the data in Rev1, export it to the Viewlogic project directory.

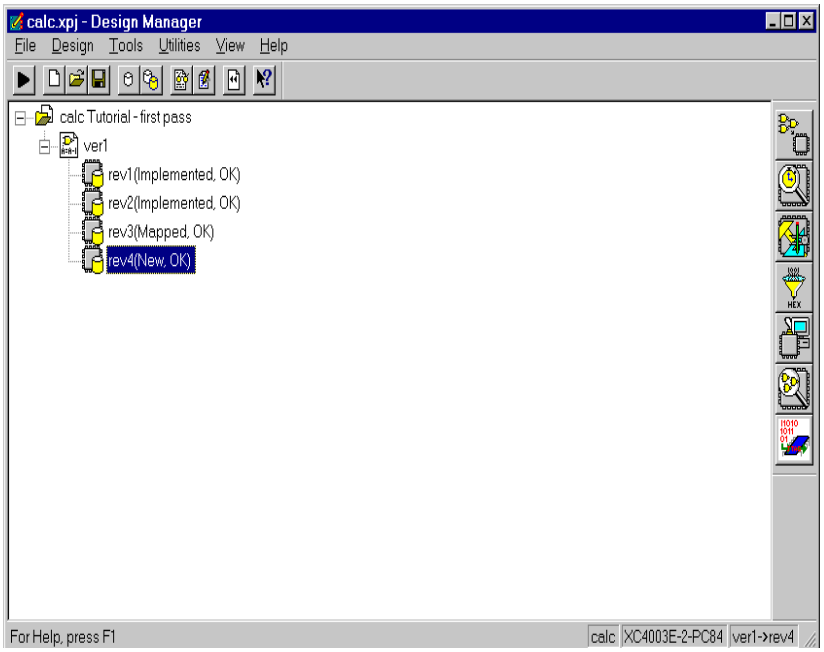


Figure 5-13 Four Revisions of a Version

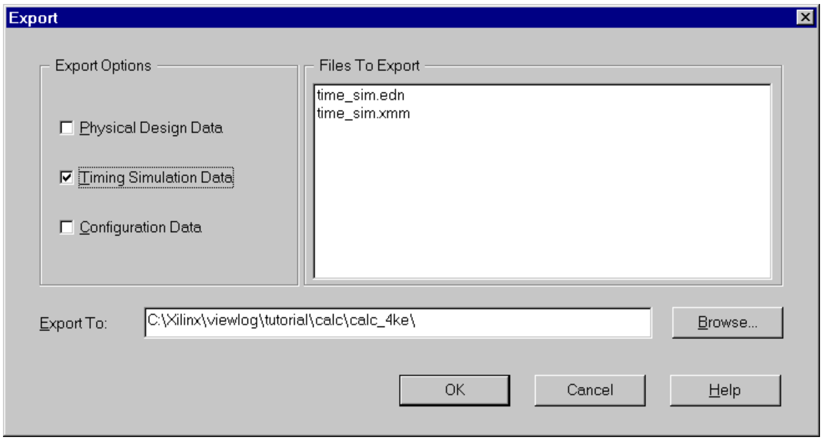
To export data, complete the following steps.

1. With the left mouse button, select the revision whose data you want to export. The selected revision highlights.

2. Select **`Design`**→**`Export...`**

   The Design Export dialog box allows you to select the types of data to place in the Viewlogic project directory. This tool allows access to physical design data, timing simulation data, and configuration data. When you select one or more of these types, the Files to Export list box updates to show the actual names of the exported files. In the Export To box, you can also select the destination directory, set by default to the Viewlogic project directory, if that is where the input EDIF file resides.

3. Click on the Timing Simulation Data checkbox, as shown in the figure that follows.



☐ Physical Design Data
☑ Timing Simulation Data
☐ Configuration Data

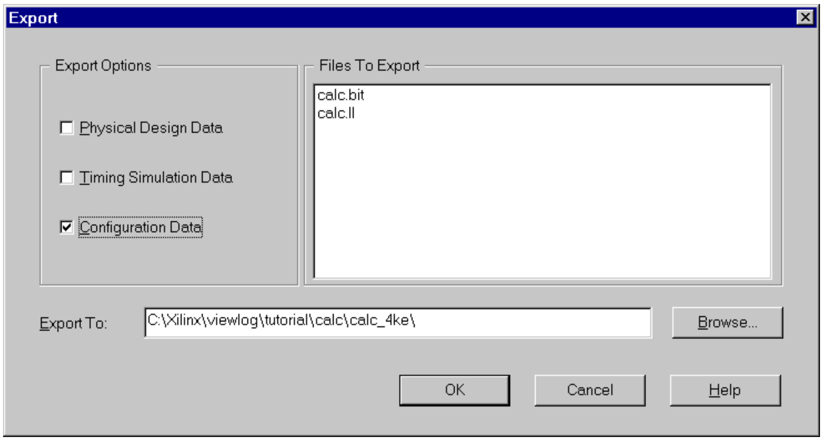
Files To Export:
time_sim.edn
time_sim.xmm

Export To: C:\Xilinx\viewlog\tutorial\calc\calc_4ke\

Figure 5-14 Selecting Timing Simulation Data Option

When you select this option, the Design Manager places the time_sim.edn file in the selected destination directory.

4. De-select the Timing Simulation Data checkbox and click on the Configuration Data checkbox, as shown in the next figure.



☐ Physical Design Data
☐ Timing Simulation Data
☑ Configuration Data

Files To Export:
calc.bit
calc.ll

Export To: C:\Xilinx\viewlog\tutorial\calc\calc_4ke\

Figure 5-15 Selecting Configuration Data Option

When you select this option, the Design Manager places the *design*.bit and *design*.ll files in the selected destination directory.

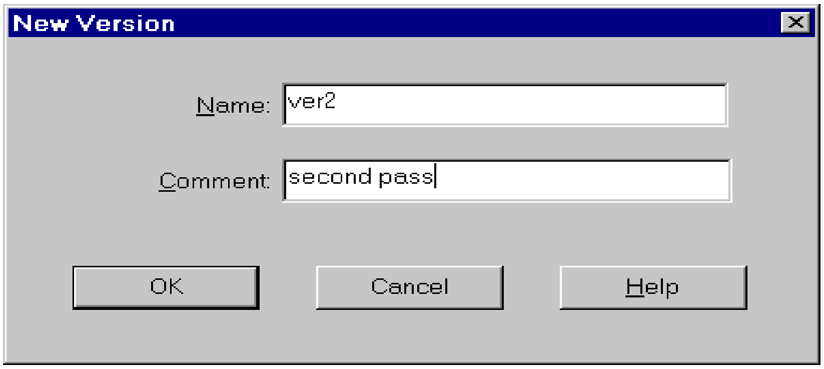
5. To export data, select any or all of the checkboxes in the Export Options section of this dialog box. Change the destination directory in the **`Export To`** field if necessary, and click on **`OK`**.

Translating the Design After Schematic Changes

Each time you change the input design, you must write a new EDIF file from ViewDraw for Design Manager to use. To keep this version of the schematic separate from previous versions, create a new version in Design Manager before implementing the design.

Note: When writing the EDIF file from ViewDraw for subsequent implementation in an existing Xilinx project, you must keep the same *design*.edn file name and project directory.

1. In the Design Manager window, select **`Design`**→**`New Version.`** This brings up the New Version dialog box, shown in the "New Version Dialog Box" figure. The version name increments by default.
2. Change the name of this version, if desired, and add any comments for this version. Click **`OK`** to create this new version.



Name: ver2

Comment: second pass

Figure 5-16 New Version Dialog Box

3. Select this new version and re-implement the design by following the instructions in the "Implementing a Design" section.

Chapter 6

Timing Simulation

Timing simulation verifies a placed and routed design by using worst-case routing and block delay information. The delay information extracts from the routed design and passes to the back-annotated simulation netlist for use during timing simulation. Timing simulation reduces the need for hardware debugging by determining whether or not the design works under worst-case conditions.

You can also use timing simulation to determine the device speed grade required for a particular application.

This chapter describes how to prepare a simulation network for a timing simulation in the Viewlogic simulation environment. It also describes how to load VWaves to view the simulation signals in a waveform format.

However, this chapter does not document specific Viewlogic commands for ViewSim, Speedwave, or VWaves. For information regarding the use of these tools, consult the online Viewlogic help files accessible from all of these tools.

This chapter contains these sections.

Understanding Simulation Procedures

In most cases, you can use the command file generated during functional simulation to perform the timing simulation. If necessary, you can make minor adjustments to the command file used to functionally simulate the design before performing a timing simulation.

The typical procedure for performing a timing simulation follows.

1. Export an EDIF file with timing information.
2. Create the timing simulation network (VSM file).
3. Start ViewSim.
4. Load the VSM file into ViewSim.
5. Simulate the device's startup sequence.
6. Manually enter the simulation commands, which can be the same as those used during functional simulation. Otherwise, execute the command file generated during functional simulation.
7. Run a new command file (optional).
8. Start VWaves.
9. View the waveforms produced by the simulation.
10. Repeat steps 5, 6, and 8 as necessary to verify the timing information.

The rest of the chapter discusses these steps in detail.

Preparing for Timing Simulation

This section describes how to prepare for timing simulation by exporting the timing simulation netlist from Design Manager and creating the VSM netlist.

Exporting the Timing Simulation Netlist from Design Manager

To export the timing simulation netlist from Design Manager, create an annotated NGD file, create a new EDIF file, and then read in the EDIF netlist. This section describes these steps.

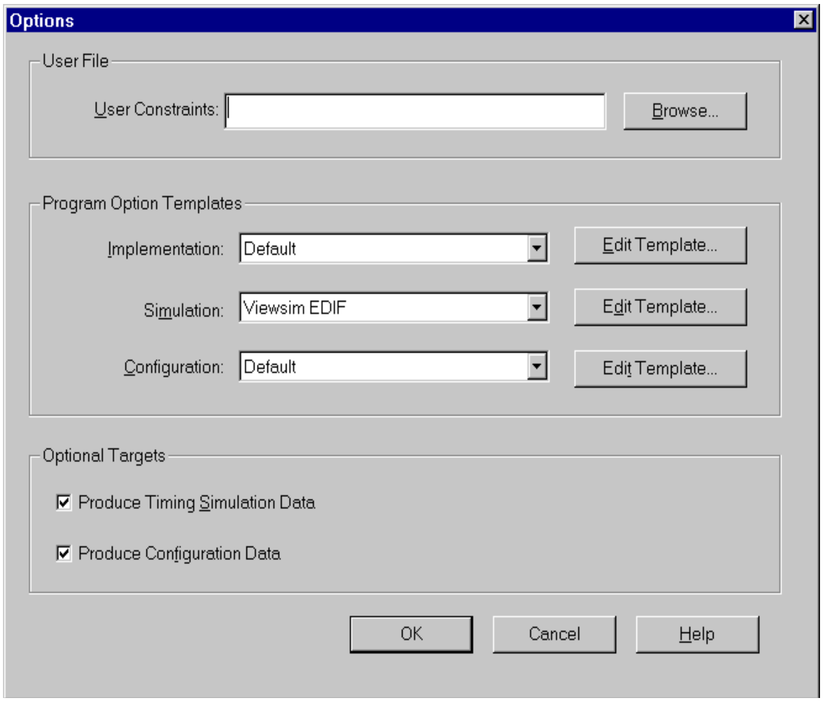
Creating an Annotated NGD File

To prepare for timing simulation of a Xilinx design, you must first use the Xilinx Design Manager and Flow Engine to produce and export timing simulation data for the design version and revision you want to simulate. The Flow Engine calls two programs you can run by selecting two options.

First, create the timing simulation data with NGDAnno, accessed in the Options dialog box.

Note: NGDAnno does not apply to CPLD devices, so do not use it with those families.

To open the Options dialog box from the Flow Engine, select **`Setup`**→**`Options`**. You can also choose this template from Design Manager when you select **`Design`**→**`Implement`**, then click on the Options button. This dialog box appears in next figure.


Options

User File

User Constraints:

Program Option Templates

Implementation: Default

Simulation: Viewsim EDIF

Configuration: Default

Optional Targets

☑ Produce Timing Simulation Data

☑ Produce Configuration Data

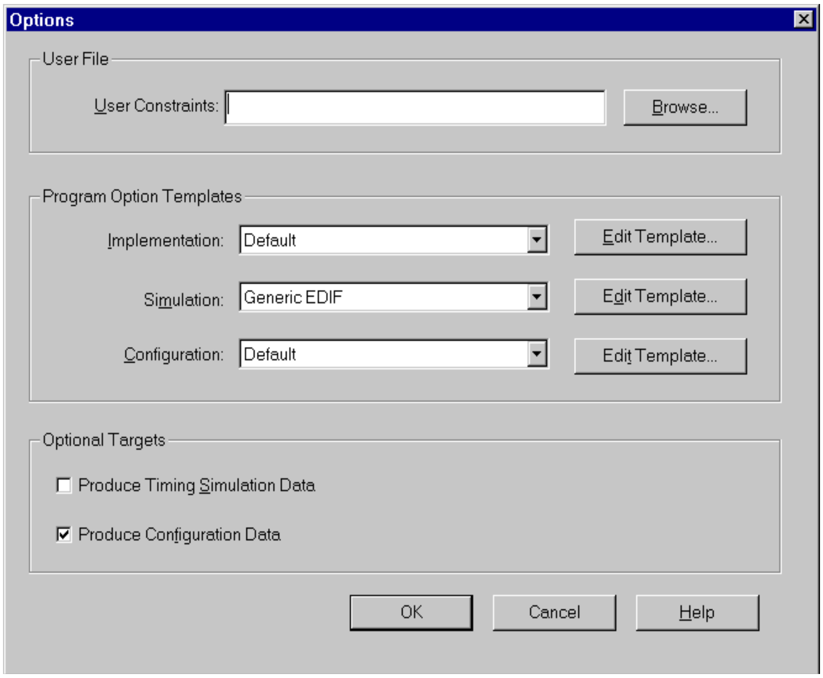
Figure 6-1 Options Dialog Box

Click on the “Produce Timing Simulation Data” checkbox to add a new step called “Timing” to the Flow Engine when it runs.

Creating a New EDIF File

Run NGD2EDIF to create an EDIF file. You can also access NGD2EDIF from the pull-down menu to the right of the Simulation title and then selecting Viewsim EDIF.

The following figure illustrates this dialog box.



Options

User File

User Constraints:

Program Option Templates

Implementation: Default

Simulation: Generic EDIF

Configuration: Default

Optional Targets

☐ Produce Timing Simulation Data

☑ Produce Configuration Data

Figure 6-2 Implementation Options Dialog Box

Click on **OK** to accept these implementation options close the Options dialog box.

These settings run NGD2EDIF, which takes the NGD file with timing information created by NGDAnno and exports it to a Viewlogic-compatible EDIF netlist.

The EDIF gets the name time_sim.edn, not the original design name, to avoid overwriting the original EDIF netlist and the original WIR file(s) produced in the next step.

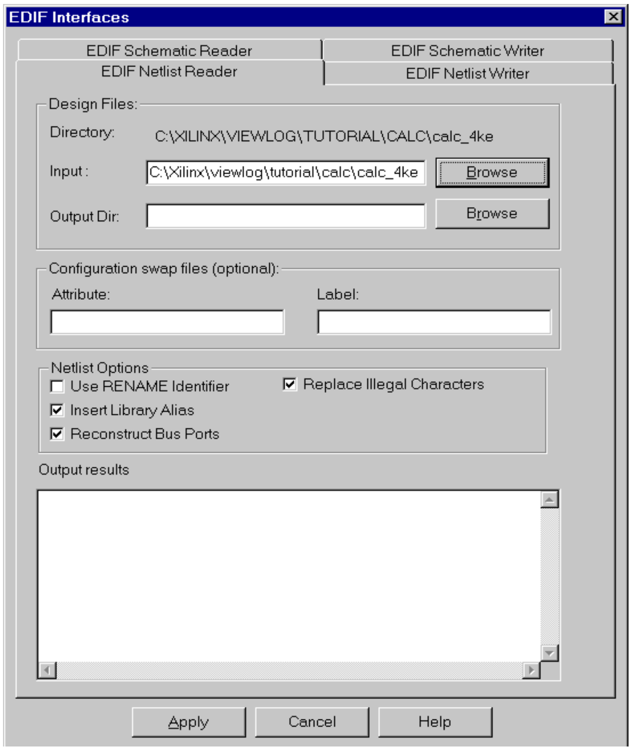
Reading in the EDIF Netlist

After the design has been placed and routed, read the new EDIF file back into the Workview Office environment. using Viewlogic's EDIFNETI. In ViewDraw, select **`Tools`**→**`Read Xilinx Timing EDIF`** to read in the time_sim.edn file.

As an alternate method of reading in the EDIF file with the timing information, use the EDIF Interfaces GUI. Click on the EDIF icon from the Workview Office Toolbar and select the EDIF Netlist Reader tab.

Fill in only the **`Input`** field if the EDIF file resides in the project directory. Use the **`Browse`** button if necessary. If the EDIF file resides in another directory, fill in the **`Output`** field with the path to your project directory.

The EDIF Netlist Reader looks similar to the next figure at this point.



Directory: C:\XILINX\VIEWLOG\TUTORIAL\CALC\calc_4ke

Input : C:\Xilinx\viewlog\tutorial\calc\calc_4ke

Output Dir:

☐ Use RENAME Identifier

☑ Replace Illegal Characters

☑ Insert Library Alias

☑ Reconstruct Bus Ports

Figure 6-3 EDIF Netlist Reader

Click on **`Apply`** to read in the time_sim.edn file.

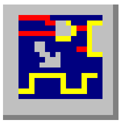
You can also enter the following on the command line.

```
edifneti time_sim.edn
```

This program reads the EDIF netlist exported by the Xilinx Design Manager, and produces WIR files to represent the design. Because the EDIF netlist is hierarchical, EDIFNETI generates one WIR file for each level of hierarchy. Lower-level WIR files are named xba1.1, xba2.1 and so on, to avoid conflicts with the original design WIR files.

Creating the VSM Netlist

Next, in ViewDraw, select **`Tools`**→**`Create Digital Netlist`**, or from the Workview Office Toolbar, select the VSM icon, illustrated in the following figure.



Either of the previous options opens the ViewSim Wirelister. Follow the following steps to create a timing simulation netlist from the time_sim.edn file.

1. Under the Basic tab, fill in the Design Name field. Select the time_sim.1 file that you just created using EDIFNETI. Use the **`Browse`** button if necessary.

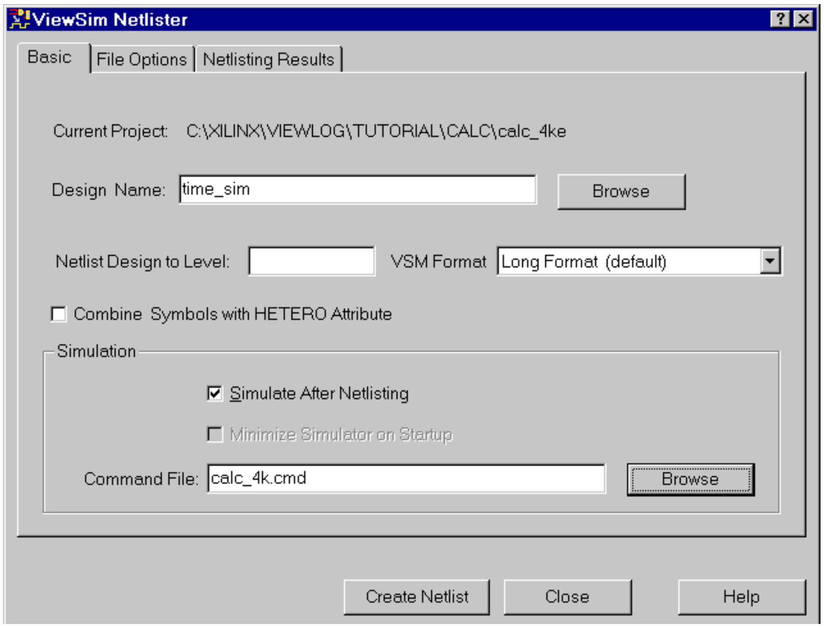
Note: Do not select the *design*.1 or *design*.edn file as the input, as this uses the uncompiled schematic design instead of the compiled version just created.

After filling in the Design Name field, fill in the **`VSM File Name`** field with time_sim.vsm.

2. Check the **`Invoke Simulator when Finished`** checkbox. This opens the Viewlogic simulator after creating the VSM file.

3. If you have a command file for this simulation, fill in the **`Command File`** field, using the Browse button if necessary.

At this point, the ViewSim Wirelister should appear similar to the next figure.


ViewSim Netlister

Basic | File Options | Netlisting Results

Current Project: C:\XILINX\VIEWLOG\TUTORIAL\CALC\calc_4ke

Design Name: time_sim

Netlist Design to Level:

VSM Format: Long Format (default)

☐ Combine Symbols with HETERO Attribute

Simulation

☑ Simulate After Netlisting

☐ Minimize Simulator on Startup

Command File: calc_4k.cmd

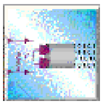
Figure 6-4 Completed ViewSim Wirelister

This Viewlogic program reads the WIR files generated by EDIFNETI and produces a single VSM file for use by Viewsim.

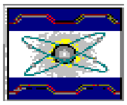
This produces a time_sim.vsm file. Because the name of this file differs from the name of the original schematic, you need to tell Viewsim which schematic to annotate. See the "Annotating Values to Original Schematic" section for details.

Loading the Design into the Viewlogic Simulator

If you checked the **`Invoke Simulator when Finished`** checkbox when creating the VSM file, the Viewlogic simulator opened automatically. Otherwise, to open the Viewlogic simulation environment, click on the ViewSim icon shown in the figure that follows.



You can also launch the Viewlogic simulation tools by opening Digital Fusion. Click on the Digital Fusion icon shown in the next figure. This also requires an extended license.



To load a VSM file into ViewSim, select **File**→**Load ViewSim Netlist**.Navigate to your project directory and select time_sim.vsm.

Loading XMM Files

If the design contains RAM elements, and if the target family supports initial values on RAM, the Flow Engine generates a file with the .xmm extension when you run NGD2EDIF. This command file, for use in the Viewlogic Integrated Simulation Environment (Viewlogic simulator), contains LOADM commands for each Viewsim RAM builtin primitive in the design.

To initialize the RAMs, execute the .xmm file from the Viewlogic simulator prompt as follows.

```
execute time_sim.xmm
```

You can include this statement in your design command file to automatically load the initial contents. The Viewlogic simulator restart command clears the RAM contents. If you issue a restart in your command file, execute the .xmm file after that.

Executing Global Reset

After loading the design into the simulation environment and initializing the RAM/ROM components and external signals, execute a global set/reset (GSR) command. This mimics the startup sequence of the FPGA or CPLD that you are simulating.

For example, for an XC4000 series device, run the following commands.

```
h GSR
sim 1000
l GSR
```

The names and polarities of the GSR signals appear in the following table.

Table 6-1 GSR Signals and Polarities

Family	Global Set/Reset Signal	Polarity
XC3x00A/L	GR	Active-Low
XC4000E/L XC4000EX/XL/ XLA/XLT/XV	GSR	Active-High
Spartan	GSR	Active-High
SpartanXL	GSR	Active-High
Virtex	GSR	Active-High
XC5200	GR	Active-High
XC9500/XL/XV	PRLD	Active-High

Annotating Values to Original Schematic

Normally, node values calculated in the simulator annotate automatically to the Viewdraw schematic of the same name.

However, because you gave the name time_sim to your timing simulation netlist, the VSM file created has a different name than the original schematic (to avoid conflicts with the original EDIF or WIR files). You must explicitly tell the simulator to annotate values to the original schematic, using the following command from the Viewlogic simulator.

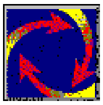
```
schemnam <design>
```

The <design> is the top-level schematic name. Values then appear in the Viewdraw window.

Invoking VWaves

To view a large number of signals simultaneously, open a waveform window in Viewsim. Use the wave command in the Viewlogic simulator to create a .vcd file that contains the signals and vectors you want to view. The **`Signals`**→**`Display in VWaves`** command also opens VWaves, or you can use these steps.

1. Click on the VWaves icon from the Workview Office Toolbar, as shown in the next figure.



2. The Open Waveforms File dialog box appears, pointing to your project directory to select the waveform display file.

   The name of the waveform file is the name you used when you added the signals to the waveform. The default name is *design*.vcd.

Note: As noted in the “Functional Simulation” chapter, the default waveform file created during both functional and timing simulation is called *design*.vcd. To avoid overwriting the functional waveform file, specify a new name when adding signals to the timing waveform file. You can also do this by editing the functional command file, changing the waveform file name after adding the signals. You can then save the modified command file to a new file name so you can use both a functional and a timing command file.

3. Click on **`OK`**.

   The Open dialog box closes and the specified waveform display file opens.

Chapter 7

Design and Simulation Techniques

This chapter discusses aspects of schematic entry and simulation with Workview Office and Powerview that you need to know to use the tools effectively. This chapter contains these sections.

Creating Your Own Libraries

You can create your own libraries of frequently used components, especially if you want to share parts of your designs with other people on your system.

You can make new library components either by editing the soft macros provided by Xilinx and saving them under another name, or simply by saving schematic blocks of your own design.

To create your own library, follow these steps.

1. Create a directory for the library. You can place it anywhere on the system. First, navigate to the directory where you want to create the library directory. Then use the mkdir command to create the library directory.

   ```
   mkdir userlib
   ```

2. Using the Project Manager for Workview Office, add a new library in the Library Search Order. Add the following information.

```
Path: c:\xilinx\userlib
```

```
Alias: userlib
```

```
Type: Writable
```

On workstations, use a text editor to modify your viewdraw.ini file to include the new library on your search path. Add the following line in the viewdraw.ini file.

```
DIR [w] /tools/xilinx/userlib (userlib)
```

In either case, place this user library after the primary library, but before the Xilinx libraries. Define the explicit path to the new library directory, unless the library resides below the project directory.

3. Open ViewDraw and enter a schematic for the component.
4. Make a symbol for the component. See the “Design Entry” chapter for instructions on creating symbols.
5. When you save the schematic and symbol, specify the location as the new user library. After saving, ViewDraw creates the sch, sym, and wir directories in the user library directory.
6. You can optionally use the Mega program to compress the library into megafile format. However, to make changes to your library, you must skip this step until all changes are complete. The megafile format is read-only.

   If you choose to compress your library into megafile format, you must specify the Type as Megafile in Project Manager, or use [m] instead of [w] in the viewdraw.ini file.

You can now specify any component from your user library and place it in your schematic as you would place a Xilinx component.

Merging Non-Schematic-Based Modules

This section describes how to incorporate a symbol representing XNF or EDIF files into your schematic.

Suppose that you choose to include an XNF or EDIF file that you generated using software from one of Xilinx’s Alliance partners. You have an XNF or EDIF file representing a portion of your design. To incorporate this netlist into your schematic, create a symbol for the XNF or EDIF file and place it on your schematic as you would any other component. However, you must observe two extra requirements.

- You must use a symbol of type Module. The default block type for a symbol is Composite, which means that a schematic exists for the symbol. When editing the symbol in Workview Office, you can change the Symbol Properties by double-clicking on the workspace (anywhere but on the symbol itself). Under the Block tab, change the Symbol Type to **`Module`**. When editing the symbol in Powerview, select **`Change`**→**`Block`**→**`Type`** and choose Module.
- You must specify a FILE=*filename* attribute to define the name of the XNF or EDIF file referenced by the symbol. For example, if the name of the XNF or EDIF file referenced by the symbol is myfile.xnf or myfile.edn, specify the FILE=myfile attribute. This file must reside in the project directory. You can add the attribute either to the symbol or to each instance of the component after placing it on the schematic. As a general rule, add the attribute to the symbol itself in the symbol editor, unless you plan to use the symbol to represent several different XNF or EDIF files with identical pins. In either case, make sure nothing is selected, then double-click on the symbol (Workview Office) or select **`Change`**→**`Object Attributes`**→**`Dialog`** (Powerview). Add the attribute and value to the dialog box.

Using Pintype Attributes

When creating a symbol, you may want to specify the directionality of the pins by using the PINTYPE attributes listed in the "PINTYPE Attributes" table.

The syntax of the PINTYPE attribute follows.

PINTYPE=*attribute*

Table 7-1 PINTYPE Attributes

IN	Input
OUT	Output
BI	Bidirectional
TRI	3-state
OCL	Open collector
OEM	Open emitter

The *attribute* can be one of the values listed in the left-hand column.

Using Power and Ground Signals

Do not leave unused inputs on symbols unconnected. You should never assume a default value for any unconnected symbol input except basic logic gates such as AND or OR. In some cases, an unconnected control input to a library symbol causes resulting behavior different from that of an input tied either High or Low. For example, with an unconnected CE input of an FDRE component, the CE logic that selects between the D-input and Q-feedback is removed. The flip-flop loads the value of the D-input ORed with its Q-feedback, clearly not the intended functionality. Timing simulation exhibits this resulting incorrect behavior; functional simulation propagates unknown signal values (represented with one or more X).

Tie unused inputs to a constant High or Low logic level in the schematic. Use the VCC or GND symbol from the Xilinx family library (not the builtin library) to tie a net to a constant logic High or Low. As an alternative, you can specify a constant High or Low value by connecting a net on the component input pin and then labeling the net as VDD or GND, global names recognized by both Viewlogic and Xilinx software.

Simulating Oscillators

The OSC symbol represents the oscillator driver for the XC3000 family. This oscillator driver, a special dedicated circuit on the FPGA, can drive an off-chip crystal oscillator to generate a high-speed clock signal within the FPGA.

The XC4000 and XC5200 families contain multiple-frequency clock signal generators, represented by the OSC4 and OSC5 symbols respectively.

To simulate these clock sources in ViewSim, apply a clock signal to the net connected to the output of the oscillator component.