
Affirma™ NC VHDL Simulator Tutorial

Product Version 3.1
June 2000

© 1995-2000 Cadence Design Systems, Inc. All rights reserved.
Printed in the United States of America.

Cadence Design Systems, Inc., 555 River Oaks Parkway, San Jose, CA 95134, USA

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. (Cadence) contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 1-800-862-4522.

All other trademarks are the property of their respective holders.

Restricted Print Permission: This publication is protected by copyright and any unauthorized use of this publication may violate copyright, trademark, and other laws. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. This statement grants you permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used solely for personal, informational, and noncommercial purposes;
2. The publication may not be modified in any way;
3. Any copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement; and
4. Cadence reserves the right to revoke this authorization at any time, and any such use shall be discontinued immediately upon written notice from Cadence.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. The information contained herein is the proprietary and confidential information of Cadence or its licensors, and is supplied subject to, and may be used only by Cadence's customer in accordance with, a written agreement between Cadence and its customer. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Contents

1

<u>Affirma™ NC VHDL Simulator Tutorial</u>	3
<u>Copying the Example</u>	4
<u>Preparing the VHDL Source Files for Simulation</u>	4
<u>Using the Command Line Interface</u>	5
<u>Using NCLaunch/NCDesktop</u>	8
<u>The Affirma SimVision Analysis Environment</u>	15
<u>The SimControl Window</u>	16
<u>Simulating the Design</u>	21
<u>Signalscan Waves</u>	22
<u>Conclusion</u>	24

Affirma™ NC VHDL Simulator Tutorial

This tutorial is a brief introduction to the Affirma NC VHDL simulator. After finishing the tutorial, you will have a basic working knowledge of the main features of the simulator.

This document assumes that you have knowledge of the Operating System that you are using and that you are familiar with the VHDL hardware description language.

The design used as the example in the tutorial is a DTMF (dual-tone multi-frequency) receiver written in VHDL RTL. The simulation testbench loads binary-encoded pulse tones into a ROM, and then decodes the digits and displays the numbers.

The tutorial contains the following sections:

- Copying the Example
- Preparing the VHDL Source Files for Simulation
 - Using the Command Line Interface
 - Using NCLaunch/NCDesktop
- The Affirma SimVision Analysis Environment
- Simulating the Design
- Signalscan Waves

Copying the Example

The source files for the example used in this tutorial are available in the following location:

UNIX:

```
install_directory/tools/inca/tutorials/Intro_To_NCVHDL
```

Linux:

```
install_directory/tools/inca/tutorials/Intro_To_NCVHDL
```

Windows:

```
install_directory\tools\inca\tutorials\intro_to_ncvhd1
```

Create a local directory and copy the tutorial files to this location. This ensures that you have write access to your project directory while preserving the original contents of the tutorial files.

Preparing the VHDL Source Files for Simulation

To prepare your source files, you must:

1. Compile the VHDL source files with the *ncvhd1* compiler.
2. Elaborate the design with the *ncelab* elaborator.

The elaborator generates a simulation snapshot that you can then load into the *ncsim* simulator.

You can prepare your source files in one of two ways:

- By opening a shell window on UNIX or Linux or a command window on Windows and then entering commands to run the compiler and the elaborator. See [“Using the Command Line Interface”](#) on page 5 .
- By using a convenient graphical interface to the tools, which is called NCLaunch on UNIX or Linux and NCDesktop on Windows. See [“Using NCLaunch/NCDesktop”](#) on page 8 .

Using the Command Line Interface

If you are using the command-line interface, you must create two setup files:

- The `cds.lib` file.

The `cds.lib` file is an ASCII text file that defines which libraries are accessible and where they are located. The file contains statements that map logical library names to their physical directory paths.

For this tutorial, you will define a library called `worklib`. The `cds.lib` file must also contain an `INCLUDE` statement to include the system `cds.lib` file provided in the installation. Use any text editor to create the following `cds.lib` file:

```
INCLUDE your_install_directory/tools/inca/files/cds.lib
DEFINE worklib ./worklib
```

See “[The cds.lib File](#)” in the *Affirma NC VHDL Simulator Help* for details on the `cds.lib` file.

Now create a directory or folder called `worklib`. This is the physical location of the library called `worklib`.

- The `hdl.var` file.

The `hdl.var` file is an ASCII text file that contains a definition of the `WORK` variable. This variable specifies the work library where the compiler stores compiled objects and other derived data. The `hdl.var` file must also contain an `INCLUDE` statement to include the system `hdl.var` file provided in the installation. Use any text editor to create the following `hdl.var` file:

```
INCLUDE your_install_directory/tools/inca/files/hdl.var
DEFINE WORK worklib
```

See “[The hdl.var File](#)” in the *Affirma NC VHDL Simulator Help* for details on the `hdl.var` file.

Now open a window so that you can run programs from the command line.

Compiling VHDL Source Files

The first step in preparing your files for simulation is to compile your VHDL source code. The program you use to compile the source files is called *ncvhdI*. This tool performs syntactic and semantic checking on the source files and VHDL design units.

Invoke *ncvhdI* with options and VHDL source file name(s). These arguments can appear in any order. However, the VHDL source files must be compiled based on dependency order. In

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

this tutorial, the VHDL source must be compiled in the following order, and, therefore, they must be ordered this way on the command line:

- `packages.vhd`
- `dtmf_recvr_core.vhd`
- `testbench.vhd`

Parameters to command-line options must immediately follow the option they modify.

If you are running the NC VHDL simulator, compile the source files with the following command:

```
% ncvhdl packages.vhd dtmf_recvr_core.vhd testbench.vhd -messages -linedebug
```

The `-messages` option displays informative messages during compilation.

The `-linedebug` option enables support for setting line breakpoints and for single-stepping through source code. This option is required for the tutorial. Because the `-linedebug` option impacts performance, it should only be used when you want to debug the source code.

If you are running the VHDL Desktop simulator, do not include the `-linedebug` option. In the VHDL Desktop simulator, all debug functionality is always on.

As `ncvhd` compiles your source files, observe the messages that it displays. Notice the following line, which appears near the end of the compilation output:

```
WORKLIB.DTMF_RECVR_CORE_TEST:BEHAVIOR (architecture)
```

All compiled VHDL design units are represented in this Library.Cell:View format. You will use this format to specify your design to downstream programs.

See [“Compiling VHDL Source Files With ncvhdl”](#) in the *Affirma NC VHDL Simulator Help* for details on `ncvhd`.

Elaborating the Design

After compiling the VHDL source code, you must elaborate the design using a program called `ncelab`.

The elaboration process constructs a design hierarchy based on the instantiation and configuration information in the design, establishes signal connectivity, and computes initial values for all objects in the design. This design hierarchy is stored in a simulation snapshot. The snapshot is the representation of your design that the simulator uses to run the simulation.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

Invoke *ncelab* with command-line options and the Library.Cell:View name of the compiled top-level HDL design unit. In the tutorial example, the Library.Cell:View name of the top-level unit is `WORKLIB.DTMF_RECVR_CORE_TEST:BEHAVIOR`.

- `worklib` is the library. This is the logical name of a library defined in the `cds.lib` file and the library that is defined as the work library in the `hdl.var` file.
- `dtmf_recvr_core_test` is the cell. This is the entity name of the design.
- `behavior` is the view. This is the architecture associated with the entity `dtmf_recvr_core_test`.

To elaborate the design, enter the following command:

```
% ncelab -messages worklib.dtmf_recvr_core_test:behavior
```

The `-messages` option displays informative messages during elaboration.

At the end of elaboration, *ncelab* writes out a simulation snapshot called `WORKLIB.DTMF_RECVR_CORE_TEST:BEHAVIOR`. Notice that the default snapshot name matches the syntax of the top-level design unit. You can use the `-snapshot` option to specify a different name.

See [“Elaborating the Design With *ncelab*”](#) in the *Affirma NC VHDL Simulator Help* for details on *ncelab*.

Loading the Design into the Simulator

After you have compiled and elaborated your design, you can invoke the simulator, which is called *ncsim*.

Invoke *ncsim* with command-line options and the simulation snapshot name as follows:

```
% ncsim -gui WORKLIB.DTMF_RECVR_CORE_TEST:BEHAVIOR
```

`WORKLIB.DTMF_RECVR_CORE_TEST:BEHAVIOR` is the name of the snapshot created in the previous step.

The `-gui` option invokes the simulator with the Affirma SimVision analysis environment and stops the simulation at simulation time 0.

See [“Simulating Your Design With *ncsim*”](#) in the *Affirma NC VHDL Simulator Help* for details on *ncsim*.

You can now simulate the design. Go to the section called [“The Affirma SimVision Analysis Environment”](#) on page 15 .

Using NCLaunch/NCDesktop

NCLaunch (called NCDesktop on Windows) is a graphical user interface that helps you manage large design projects. The launch tool gives you a unified view of the files and libraries in your design and provides you with an easy and consistent way to configure and launch your Cadence simulation tools. NCLaunch/NCDesktop can help you get your designs into simulation faster, so that you can find the maximum number of problems in the least amount of time.

See the *NCLaunch User Guide* for details on NCLaunch/NCDesktop.

1. Invoke NCLaunch or NCDesktop.

On UNIX or Linux, invoke NCLaunch from the directory into which you copied the tutorial files using the following command:

```
% nclaunch -new &
```

If you are running the VHDL Desktop simulator, invoke NCDesktop with the `ncdesktop` command or from the *Start* menu (*Start—Programs—Cadence Design Systems—Design & Verification—NCDesktop*).

The `-new` option on the command line specifies that you want to start a new design.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

This command opens the Open Design Directory form, which lets you choose a design directory from which to start. On the form, the *Select the Design Directory* field is seeded with the path to the directory that contains the tutorial source files.

The image shows a dialog box titled "Open Design Directory". It has three main sections for user input:

- Select the Design Directory:** A text field containing the path `/hm/belanger/inca/vhdl/ncvhdltut` and a folder icon button.
- Select the cds.lib file:** A text field containing the path `/hm/belanger/inca/vhdl/ncvhdltut` and a "New..." button.
- Select the work library:** A dropdown menu and a "New..." button.

At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

Note: In this tutorial, the forms that are shown are the NCLaunch forms as they appear on UNIX or Linux. Some of the NCDesktop forms that appear on Windows will be slightly different. For example, on Windows, the banner on some forms will show `NCDesktop` instead of `NCLaunch`, and the command prompt in the main window will be `ncdesktop>` instead of `nclaunch>`.

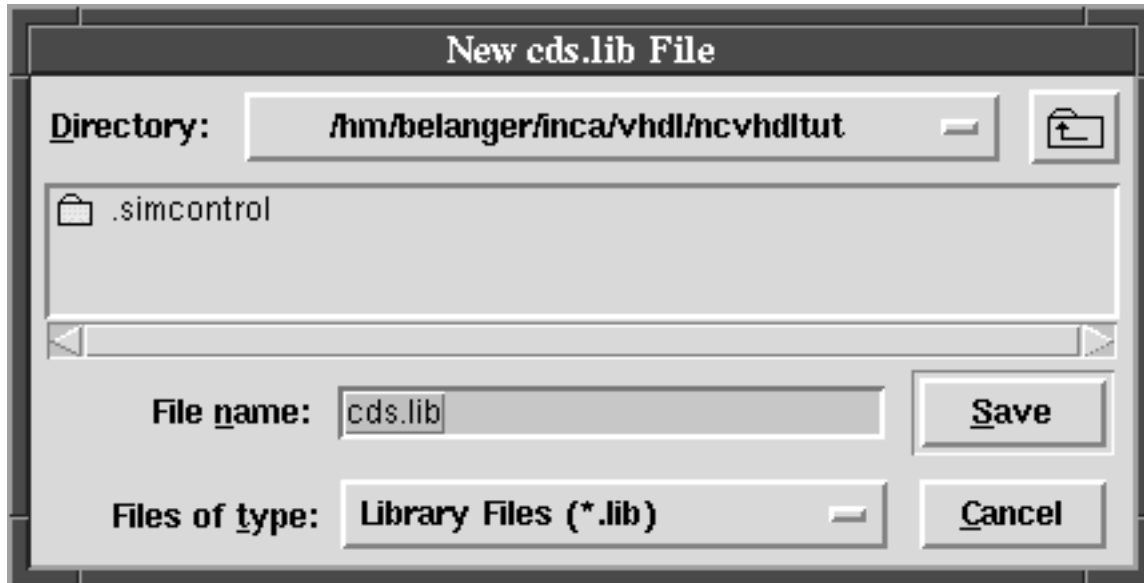
2. Create a `cds.lib` file. The `cds.lib` file is an ASCII text file that defines which libraries are accessible and where they are located. The file contains statements that map logical library names to their physical directory paths.

To create the file, click the *New* button next to the *Select the cds.lib file* field. This opens the New `cds.lib` File form.

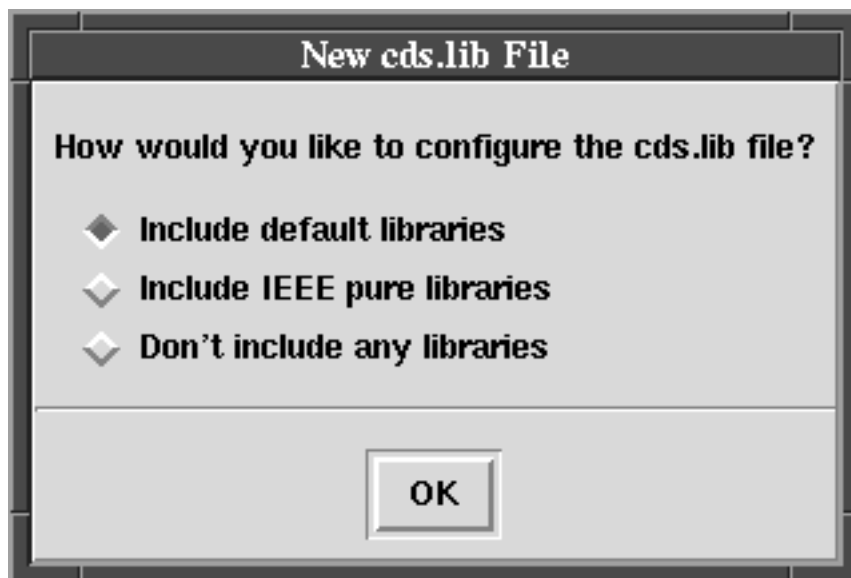
Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

Enter `cds.lib` in the *File name* field on the form and then click the *Save* button.



This opens a form on which you can select which IEEE libraries to use. The default is to use the Synopsys IEEE libraries.



Click the *OK* button.

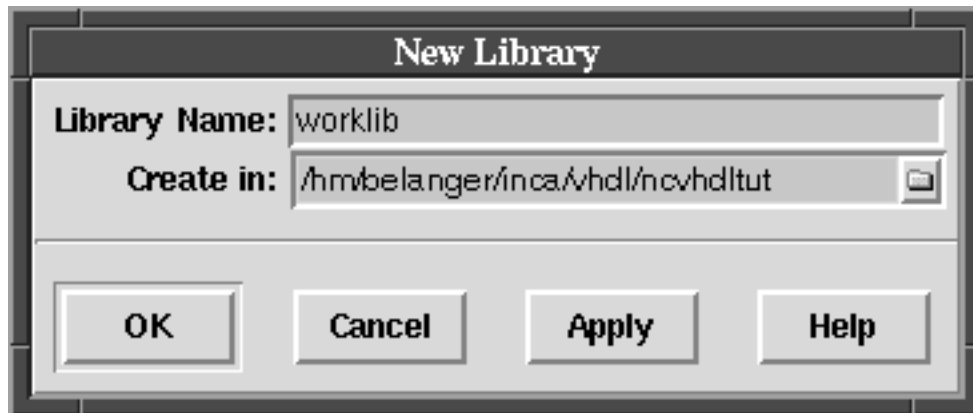
See "[IEEE Libraries](#)" in the *Affirma NC VHDL Simulator Help* for more information on IEEE libraries.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

3. Select the work library. To do this, click the *New* button next to the *Select the work library* field on the Open Design Directory form.

The New Library form appears. Enter `worklib` in the *Library Name* field.



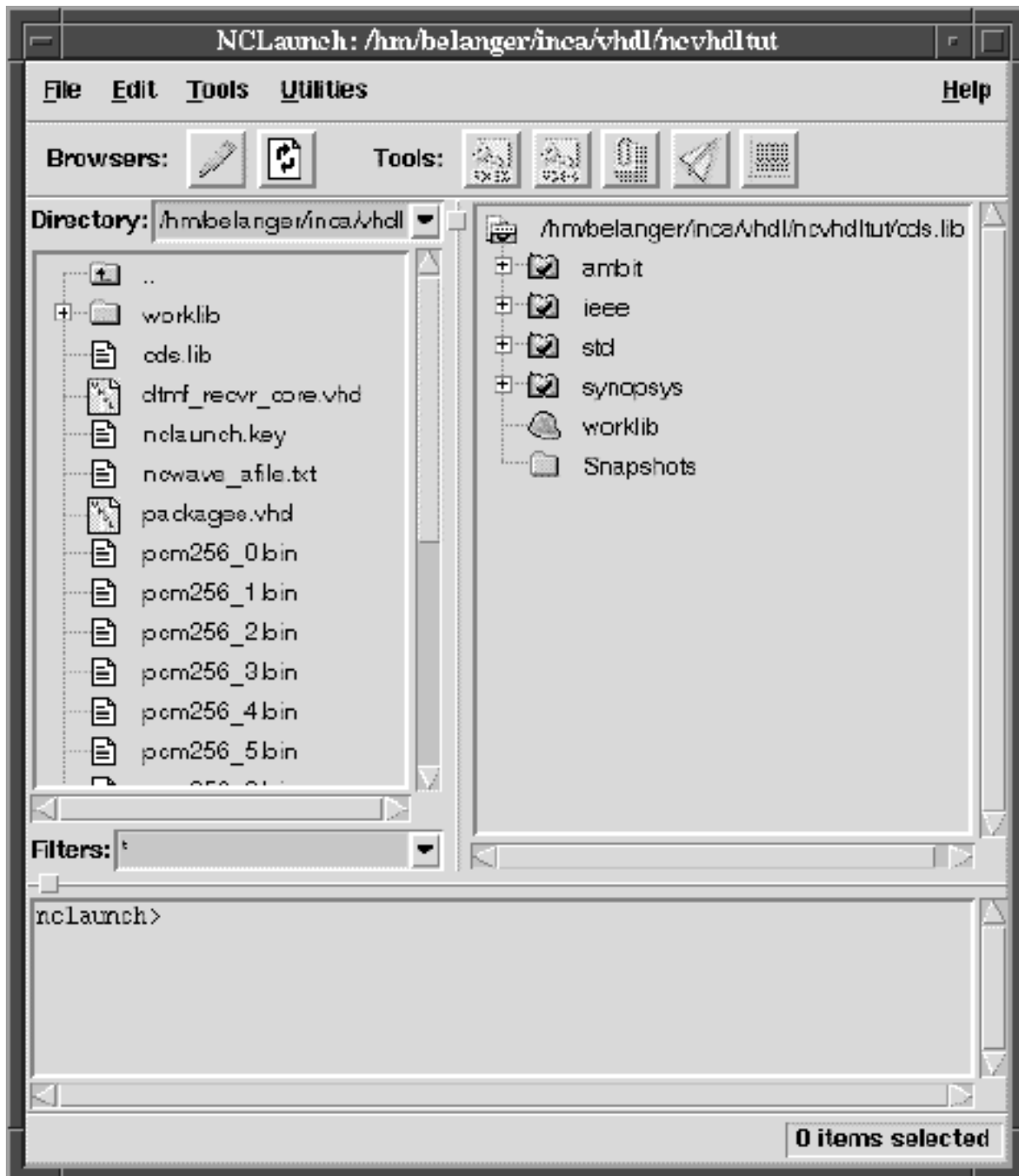
The image shows a dialog box titled "New Library". It has two text input fields. The first is labeled "Library Name:" and contains the text "worklib". The second is labeled "Create in:" and contains the path "/hmvbelanger/inca/vhdl/novhdlut". To the right of the "Create in:" field is a small folder icon. At the bottom of the dialog box, there are four buttons: "OK", "Cancel", "Apply", and "Help".

Click the *OK* button on the New Library form, and then click the *OK* button on the Open Design Directory form.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

The tutorial directory now contains a `cds.lib` file, and the main NCLaunch window appears. At this point, the window looks as follows:



Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

The NCLaunch/NCDesktop window is divided into the following components:

- Menu Bar and Tool Bar - Provides the commands and fast action buttons that let you manipulate design elements and start the various tools.
- File Browser - The pane on the left side of the window displays the files in the file system.
- Library Browser - The pane on the right side of the window displays objects in the libraries.
- Console Window - Displays output from tools, and takes user input to generate Tcl scripts.

You are now ready to compile the VHDL source files.

Compiling VHDL Source Files

To compile the VHDL files:

1. Select your VHDL source files.

The VHDL source must be compiled based on dependency order. In the tutorial, the VHDL source must be compiled in the following order:

- `packages.vhd`
- `dtmf_recvr_core.vhd`
- `testbench.vhd`

Hold down the `CTRL` key while selecting each file with the left mouse button.

2. If you are running the NC VHDL simulator on UNIX or Linux, set the `-linedebug` option to enable single-stepping and the setting of line breakpoints. This is required in order to complete the tutorial. Because this option impacts performance, you must set it explicitly. To set this option:

- a. Select *Tools—VHDL Compiler*.

The Compile VHDL form appears.

- b. Select the *Enable line debug* option and click the *OK* button.

This sets the option and invokes the VHDL compiler. The results of the compilation appear in the Console Window of the NCLaunch window.

NCLaunch saves your selected options and will use them in subsequent runs.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

If you are running the VHDL Desktop simulator, all debug features are always on by default, so this option is not available. Just click the *Launch VHDL Compiler* button on the Tool Bar to compile the files.

After the source files have been compiled, you can click the plus sign to the left of `worklib` in the Library Browser. This expands the `worklib` library and lists the VHDL design units that have been compiled.

Elaborating the Design

After compiling the VHDL source code, you must elaborate the design.

The elaboration process constructs a design hierarchy based on the instantiation and configuration information in the design, establishes signal connectivity, and computes initial values for all objects in the design. This design hierarchy is stored in a simulation snapshot. The snapshot is the representation of your design that the simulator uses to run the simulation.

To elaborate the design:

1. Expand the `worklib` library in the Library Browser.
2. Expand the `dtmf_recvr_core_test` design unit.
3. Select the top-level design unit.

The top-level in the tutorial is `WORKLIB.DTMF_RECVR_CORE_TEST:BEHAVIOR`.

4. Click the *Launch Elaborator* button on the Tool Bar.

For the tutorial, no special elaborator options need to be set. Clicking on the button on the Tool Bar runs the elaboration using the default options.

This process populates the `Snapshots` folder with the name of the simulation snapshot: `worklib.dtmf_recvr_core_test:behavior`.

You are now prepared to start a simulation.

Loading the Design into the Simulator

To load the snapshot into the simulator:

1. Select the snapshot `worklib.dtmf_recvr_core_test:behavior` (in the `Snapshots` folder).
2. Click the *Launch Simulator* button on the Tool Bar.

Because `-gui` is set as a default option for the simulator, the simulator is automatically invoked in graphical mode when you click the *Launch Simulator* button, and the Affirma SimVision analysis environment main window (the SimControl window) appears.

See [“The Affirma SimVision Analysis Environment”](#) on page 15 .

The Affirma SimVision Analysis Environment

The Affirma SimVision analysis environment is a unified graphical debug environment for Cadence simulators. The SimVision environment features advanced debug and analysis tools and innovative high-level design and visualization capabilities. These tools include:

- The SimControl window, which allows you to directly interact with the simulator. You can single step, trace signals, set breakpoints, and observe signals to verify your designs. SimControl also provides access to the following debug tools:
 - The Watch Objects Window, which lets you observe the value of selected signals.
 - The Navigator, which displays the design hierarchy and shows you signal values at any level of the hierarchy.
 - The Signal Flow Browser, which lets you trace backwards through a design from a signal that has a questionable value to where a signal first diverges from expected behavior.
- Signalscan waves—Lets you display waveforms.
- Comparescan—Lets you compare SHM and VCD waveform databases.

See the [SimVision Analysis Environment User Guide](#) for full details on the SimVision analysis environment.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

The SimControl Window

SimControl is the main SimVision analysis environment window that appears when you invoke the simulator with the `-gui` option. The following figure shows the SimControl window as it appears when the simulator is invoked using the tutorial example:



The different parts of the SimControl window are:

- The Menu Bar, which contains the pulldown menus that let you execute simulator commands.
- The Tool Bar, which contains buttons that give you fast access to commonly-used commands and to the other SimVision tools. You can define your own buttons for Tcl commands and add them to the Tool Bar.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

- The Source Browser, which displays your source code. You can select scopes, signals, or ports in the Source Browser and operate on them. Because you have compiled the VHDL source code for the example with line debugging enabled, you will be able to set line breakpoints and single-step through the source.
- The Scope Region, which displays the current scope and allows you to quickly set the scope to another level in the hierarchy.
- The Input/Output Region, which displays simulator output and allows command-line input to the simulator.
- The Message Region, which displays information about the menu item or button where the mouse pointer is pointing.

At this point, you could just run the simulation by clicking the *Run Simulation* button.



However, before we simulate, let's look at a few important features that are available.

Suppressing 1164 Package Warnings

The tutorial example generates many 1164 assertion warning messages. To suppress the display of these messages:

1. Select *Set—Tcl Variable*.
2. On the Set Variable form, click the down arrow next to the *Name* field to display a list of Tcl variables. Select `assert_1164_warnings`.
3. In the *Value* field enter `no`.
4. Click *OK*.

Notice how the Tcl command is echoed in the I/O region.

Setting a Line Breakpoint

Now set a line breakpoint on the following source code line:

```
stim_end <= not stim_end;
```

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

To find this line in the source code:

1. With the Scope field set to : (colon alone indicates the top level of a VHDL design), select *File—Find—Text*.
2. On the Find Text form, enter `stim_end` in the *Text* field and click the *case insensitive* button to do a case insensitive search for this text string in the top-level testbench file.
3. Click the *Forward* button.
This takes you to the first occurrence of `stim_end`, which is its declaration.
4. Click the *Forward* button again to go to the next occurrence, which is the source code line that you want.
5. Click the *Cancel* button to close the Find Text form.

To set the line breakpoint, double-click on the line number of the statement. This sets a red breakpoint icon next to the line number.

Now that you have the breakpoint set, let's navigate the design hierarchy.

Navigating the Design and Selecting Signals to Monitor

In the SimControl window, invoke the Navigator by selecting *Tools—Navigator* or by clicking on the Navigator button on the Tool Bar.



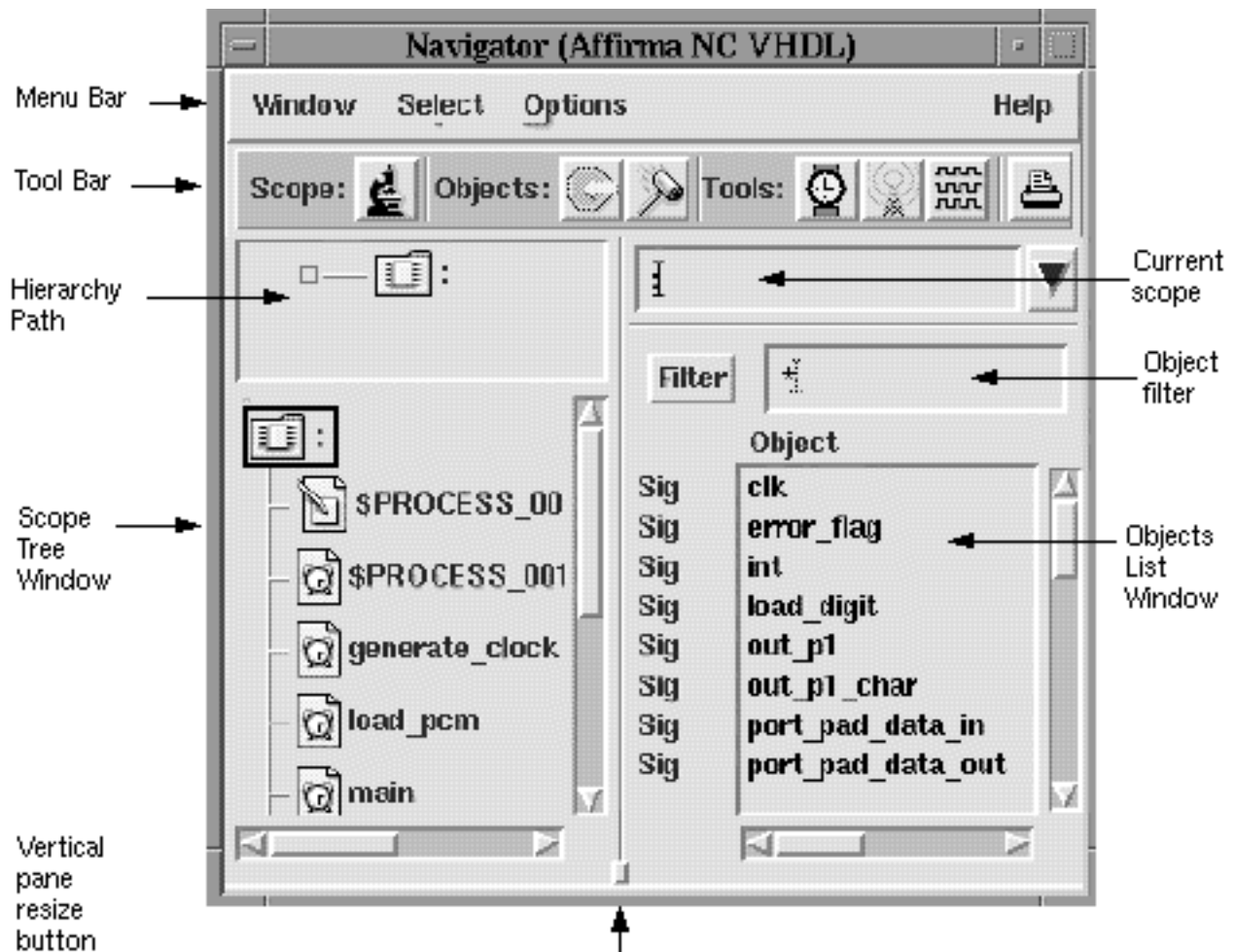
The Navigator consists of two sides, or panes. On the left is the Scope Tree Window, in which you can view your current design hierarchy in a graphical tree representation. To browse the hierarchy, double-click the left mouse button on a node to expand to the lower level. On the right side of the Navigator is the Objects List Window, which displays a list of objects with their current simulation values and declarations. A filter controls the signals that are displayed in this window.

1. Using the left mouse button select the component icon in the hierarchy tree.

Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

This displays all top-level objects with their values in the Objects List Window, as shown in the following figure:



2. Select the signals `load_digit` and `stim_end`.

Select the first with the left mouse button, then hold down the `CTRL` key while selecting the second.

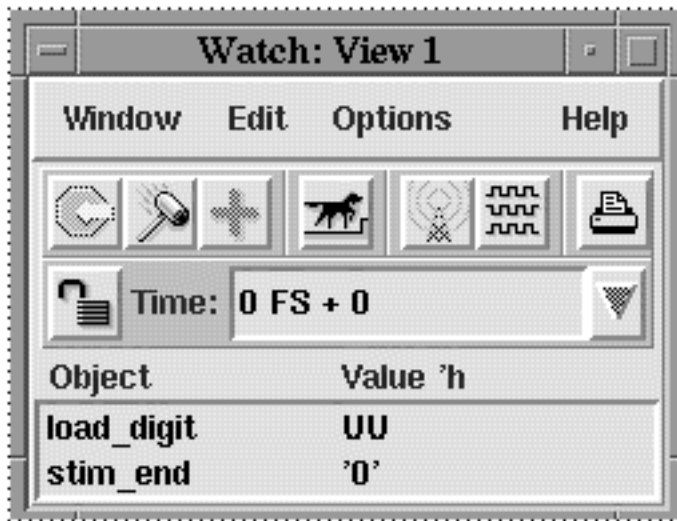
3. Click the Watch Window button on the Navigator tool bar.



Affirma NC VHDL Simulator Tutorial

Affirma™ NC VHDL Simulator Tutorial

This invokes the Watch Window. The two signals that you want to monitor during the simulation are displayed in the window.



4. Traverse the hierarchy using the Navigator. To do this, double-click the component icon that you selected previously.

This expands the tree and displays the components and processes contained within this block.

5. Double-click on the instance called `TOP` to go into this component. Proceed through the hierarchy through `TDSP_CORE_INST` down to `TDSP_CORE_MACH_INST`.

Note: If the instance names referred to above are not visible, use the Scope Tree Options form to turn on the display of instance names. On UNIX, select *Options—Scope Tree* and then click the *instance name* button. On NT, select *Options—Scope Tree*, select the *Formatting* tab, and then click the *instance names* button.

Notice that the SimControl source browser automatically changes to the part of the design displayed in the Navigator.

The Objects List Window of the Navigator now displays all of the objects for the hierarchical scope `:TOP:TDSP_CORE_INST:TDSP_MACH_INST`. The objects consist of ports represented by `In` and `Out`, internal signals represented by `Sig`, and constants represented by `Const`.

Viewing Signals in Signalscan Waves

1. Disable the display of constants in the Navigator Objects List Window. To do this:
 - a. Select *Options—Objects List* in the Navigator.
 - b. On the Object List Options form, under *Show VHDL*, click on the *constants* button to deselect it and click *OK*. This prevents the constants from being displayed in the Objects list.
2. Select all of the objects in the Objects List. To do this, hold down the left mouse button and drag the cursor across the names of the objects.
3. Click the *Waveform View* button on the Navigator tool bar.



This opens an SHM database called `waves`, probes the selected signals to the database, and invokes Signalscan waves with all of the selected signals displayed.

Look at the SimControl window. Each Tcl command that you have executed using the SimVision windows is echoed in the simulator I/O region. The line breakpoint was created with the `stop` command. A `scope` command was executed whenever you changed scope. The waveform database was created with the `database` command, and the signals were added to the waveform database with the `probe` command.

You are now ready to begin simulating.

Simulating the Design

To simulate, click the *Run Simulation* button on the SimControl window.



The *Run Simulation* button changes to the *Stop Simulation* button when simulation starts.



Notice how the simulation time updates in the SimControl window just under the tool bar. Also notice that the waveforms update dynamically in Signalscan waves as the simulation runs.

The simulation runs to the line breakpoint that you have set. The breakpoint should trigger around 3.16 ms. The SimControl Source Browser jumps to the breakpoint line with an arrow pointing to the line.

The Watch Window is updated when the simulation is interrupted at the breakpoint. The window shows that the value of the signal `load_digit` has changed value by highlighting the value of this signal.

1. Disable the line breakpoint. To do this, double-click on the line number with the red breakpoint icon.

The breakpoint icon turns green. The breakpoint is now disabled, but a marker is left so that you can easily find this point again.

2. Set an object breakpoint on the signal `stim_end`.

The Watch Window indicates that the value of this signal is 0. To set a breakpoint that will trigger when this signal transitions to a 1:

- a. Click on the signal `stim_end` in the Source Browser to select it. The signal is now highlighted.
- b. Select *Set—Breakpoint—Object*. This seeds the Set Break form with the name of the selected signal.
- c. Click the `if` condition button and then enter the value 1 in the value field.
- d. Click *OK*.

Click the *Run Simulation* button again. The simulation stops when the value of `stim_end` changes value. Notice in the Watch Window that the signal value of `stim_end` is highlighted to indicate a change in this value.

Signalscan Waves

Signalscan waves is a powerful tool for analyzing your simulation. It plays an important role in the verification cycle, allowing you to analyze large amounts of complex simulation data quickly and accurately.

At this point, the waveform viewer is zoomed in to the end of the simulation.

1. Click the *ZmOutXFull* button on the waveform tool bar.
2. Click the *ZoomInX* button to zoom in until you can view the transitions on the `phi_#` signals.

This will take several clicks with the left mouse button as you zoom in more each time.

Creating a Bus

1. Create a bus out of the `phi_#` signals. To do this:
 - a. Select all six of the `phi_#` signals by pressing the left mouse button and dragging the cursor over the signal names.
 - b. Select *Edit—Create—Bus*.

On the Make Bus From Selected Variable(s) form, the name `phi_` is chosen as the default name for the bus, and the *Replace Original* button is selected.

- c. Click *OK*.

Signalscan waves creates a bus called `phi_%[1:6]`, and the individual bits disappear from the display.

Use the *ZoomInX* button to better view the values in the waveform of the newly created bus.

To break the bus into individual bits, select the bus and then select *Edit—Expand*.

Zooming In on a Time Region

To zoom in more closely on the waveforms:

1. Select a signal transition with the left mouse button. This inserts one cursor.
2. Insert a second cursor by clicking the middle mouse button.
3. Click the right mouse button to zoom in between the cursors.

Reordering Signals

You can also reorder the signals very easily. To do this:

1. Select a signal name with the left mouse button.
2. Hold down the middle mouse button and drag the signal to the desired position.
3. Release the middle mouse button.

Saving Settings

To save the changes you have made:

1. Select *File—Save Do File*.
2. On the form, enter the name `vhdl_tutorial.do` in the *Save Do-File* field and click *OK*.

A form appears that prompts you for the settings that you want to save. By default, all settings are saved.

3. Click *OK*.

Signalscan waves saves your settings in a file called `vhdl_tutorial.do`.

See the [*Signalscan Waves User Guide*](#) for details on Signalscan waves.

Conclusion

You have now completed the NC VHDL simulator tutorial.

You can continue to experiment with the tools introduced in this tutorial. When you are finished, select *File—Exit* to close the SimControl, Signalscan waves, and launch tool windows.