

CADENCE NCLAUNCH TUTORIAL

1	Directory Structure.....	2
2	Invoking NCLaunch.....	3
3	Components of NCLaunch.....	3
4	Copying the Example.....	4
5	Preparing the VHDL Source Files for Simulation.....	4
5.1	Compile the VHDL files.....	4
5.2	Elaborating the Design.....	5
5.3	Loading the Snapshot into the Simulator.....	6
6	Using the SimVision Analysis Environment.....	7
6.1	The SimControl Window.....	8
6.2	Navigating the Design and Selecting Signals to Monitor.....	10
6.3	Viewing Signals in SignalScan Waveform Viewer.....	13
7	Simulating the Design.....	14
8	Using SignalScan Waveform Viewer.....	15
9	Reference.....	16

NCLaunch is a graphical user interface that helps you manage large design projects and lets you configure and launch your Cadence simulation tools. This NCLaunch tutorial is intended for students to help them simulate Verilog, VHDL, or mixed-language designs using the NCLaunch tool. This tutorial explains the functionality of the tool and gives examples of simulating a VHDL module with NCLaunch. After finishing this tutorial, you will have a basic working knowledge of the main features of the simulator.

(Note: Most of the material in this tutorial has been borrowed from the Cadence Documentation.)

1 Directory Structure

Before beginning, create a new directory under your home directory to store all your vhdl/verilog designs.

```
class3: /home/username/ $mkdir vhdl
```

You need to have two important files in this new directory before you begin working. These are the **cds.lib** file and the **hdl.var** 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 that is provided in your installation. Use any text editor to create the following *cds.lib* file:

```
include $CDS_INST_DIR/tools/inca/files/cds.lib  
DEFINE workdir /home/username/vhdl
```

Save this *cds.lib* file under the *vhdl* directory that you just created.

The **hdl.var** file is an optional ASCII text file that contains definitions for system variables. One of these variables is the WORK variable. This variable specifies the work library in which the compiler stores compiled objects and other derived data. You can define the WORK variable in the *hdl.var* file, or you can specify the work library on the command line with the -work option. You can define several other variables in an *hdl.var* file. For example, you can define variables to specify commonly used command-line options for the NC tools. Use any text editor and create an *hdl.var* file that contains the following lines:

```
#Define the work library  
DEFINE WORK /home/username/vhdl  
#Define valid Verilog file extensions  
DEFINE VERILOG_SUFFIX (.v, .vr, .vb, .vg)  
#Define valid VHDL file extensions  
DEFINE VHDL_SUFFIX (.vhd, .vhdl)  
# Specify command line options for the ncvhdl compiler  
DEFINE NCVHDL_OPTS -messages
```

```

# Specify command line options for the ncvhdl compiler
DEFINE NCVLOGOPTS -messages
# Specify command line options for the elaborator
DEFINE NCELABOPTS -messages
## Specify the simulation start-up command file
DEFINE NCSIMRC /usr/design/simrc.cmd

```

2 Invoking NCLaunch

Once you have your cds.lib and hdl.var files set up in your working directory, you are ready to invoke NCLaunch. This is done with the following command in a shell window:

```

class3: /home/username/vhdl $. cadasic
class3: /home/username/vhdl $ nclaunch &

```

3 Components of NCLaunch

The NCLaunch command opens up the NCLaunch graphical user interface (GUI) main window. The main window is composed of a menu bar, toolbar icon strip, file browser, design area and console window as shown in the figure below.

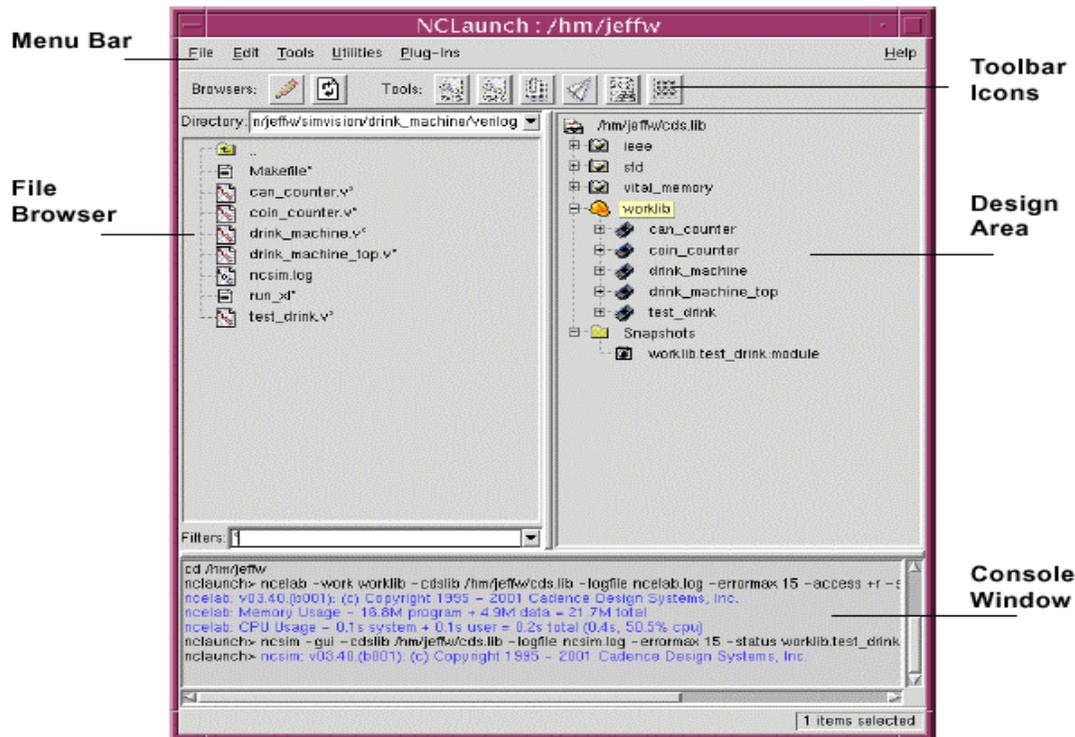


Figure 1: NCLAUNCH Main window

The main NCLaunch window is divided into the following components:

- Menu Bar and Tool Bar--Provide 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 allows you to input commands.

The rest of the tutorial is organized into the following sections:

- Copying the Example
- Preparing the VHDL Source Files for Simulation
- Using NCLaunch
- Using the SimVision Analysis Environment
- Simulating the Design
- Using SignalScan Waveform Viewer

4 Copying the Example

The source files for the example used in this tutorial are located in the following directory: `/net/cadence2001/LDV31/tools.sun4/inca/tutorials/Intro_To_NCVHDL`. Copy the contents of the `Intro_To_NCVHDL` directory into your `vhdl` directory. The design used as the example in the tutorial is a dual-tone multi-frequency (DTMF) 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.

5 Preparing the VHDL Source Files for Simulation

5.1 Compile the VHDL files

1. Select your VHDL source files: The VHDL source must be compiled based on dependency order. In this tutorial, you must compile the VHDL source files 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.

Note: Beginning with the LDV 3.4 release, you can compile design files in an order-independent manner.

2. If you are running the NC-VHDL simulator, 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:

- Select Tools → VHDL Compiler.
The Compile VHDL form appears.
- Select the *Enable line debug* option.
- Click the *OK* button.

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

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 you have compiled.

5.2 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.

To elaborate the design:

- Click the plus sign to the left of the worklib library in the Library Browser to expand it.
- Expand the *dtmf_recvr_core_test* design unit.
- Select the top-level design unit.
- The top-level in the tutorial is
WORKLIB.DTMF_RECVR_CORE_TEST:BEHAVIOR.
- Click the ***Launch Elaborator*** button on the Tool Bar.



Figure 1. NCLaunch Elaborator

For this tutorial, you do not need to set any special elaborator options. Clicking 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.

5.3 Loading the Snapshot into the Simulator

To load the snapshot into the simulator:

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

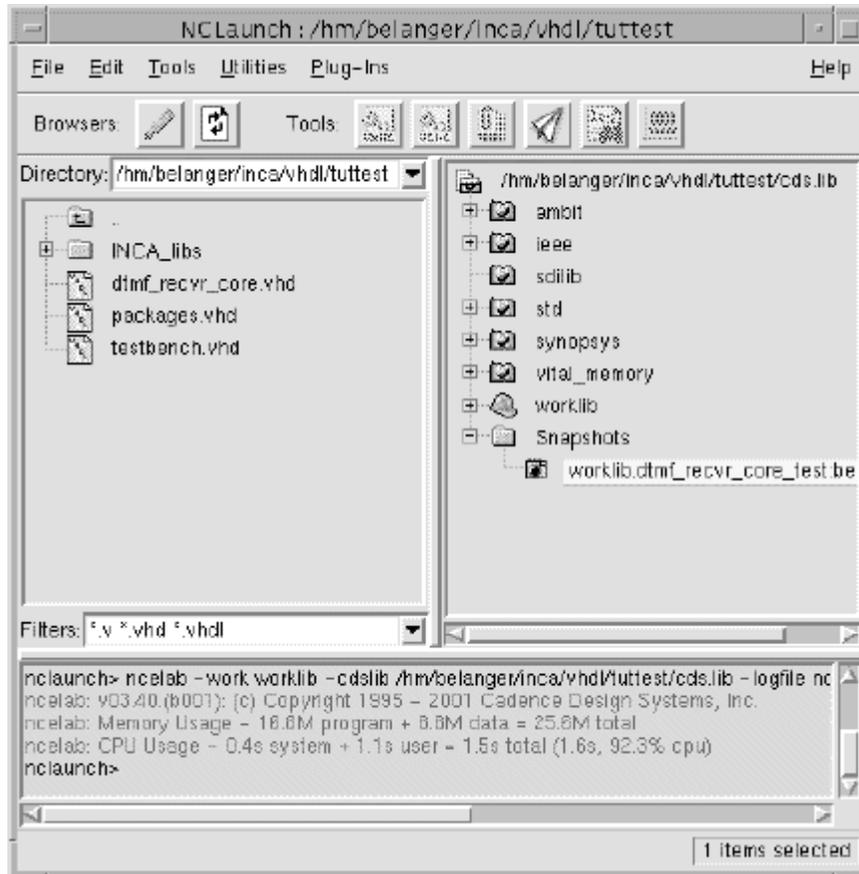


Figure 2. NCLaunch Loading SnapShot into Simulator

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

6 Using the SimVision Analysis Environment

The 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 lets you 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 values 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.
- The Cycle View, which lets you step through the simulation cycle to debug delta cycle bugs in the design.
- SimVision Waveform Viewer--Lets you display waveforms.
- Comparescan--Lets you compare SHM and VCD waveform databases.

6.1 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 you invoke the simulator using the tutorial example:

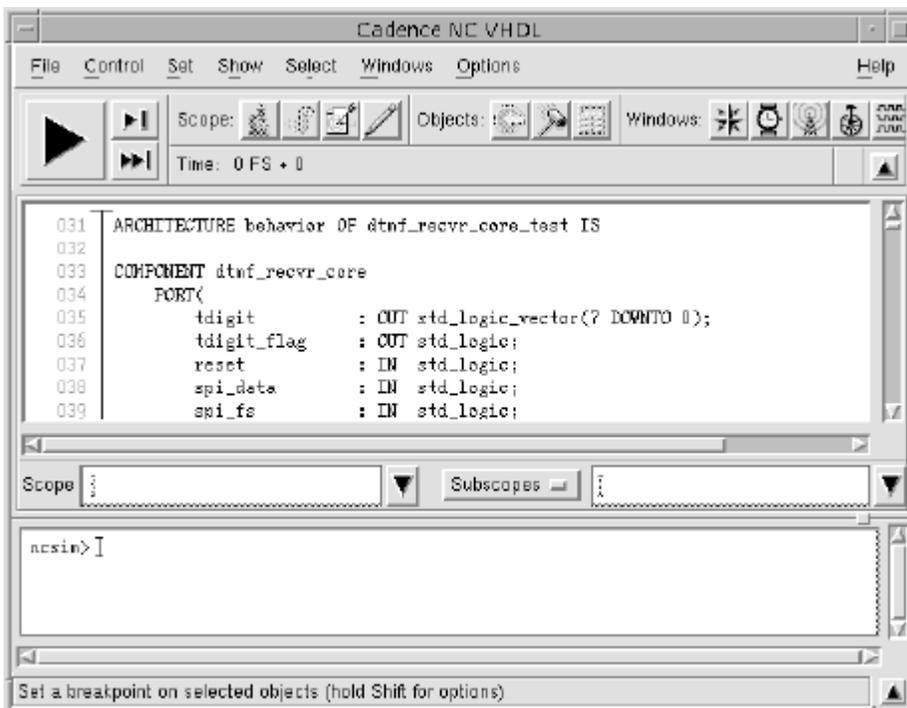


Figure 3. SimControl Window

The SimControl window is divided into the following sections:

- 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.

- The Source Browser, which displays your source code. You can select scopes, signals, or ports in the Source Browser and operate on them.
- The Scope Region, which displays the current scope and lets you quickly set the scope to another level in the hierarchy.
- The Input/Output Region, which displays simulator output and lets you give 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 shown below.



However, before you 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:

- Select *Set->Tcl Variable*.
This opens the Set Variable form.
- 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*.
- In the *Value* field enter *no*.
- Click *OK*.

Notice how the TCL command is echoed in the I/O region of the SimControl window.

Setting a Line Breakpoint

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

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

To find this line in the source code:

- With the Scope field on the SimControl window set to : (colon alone indicates the top level of a VHDL design), select **File → Find → Text**.
The Find Text form appears.

- 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.
- Click the *Forward* button.
This takes you to the first occurrence of `stim_end`, which is its declaration.
- Click the *Forward* button again to go to the next occurrence, which is the source code line that you want.
- 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 set the breakpoint, let's navigate the design hierarchy and select the signals that you want to monitor during simulation.

6.2 Navigating the Design and Selecting Signals to Monitor

In the SimControl window, invoke the Navigator by selecting *Tools* → *Navigator* or by clicking 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 it 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.

To display and then select the signals that you are going to monitor:

1. Using the left mouse button, select the component icon in the hierarchy tree. This displays all top-level objects with their values in the Objects List Window, as shown in the following figure:

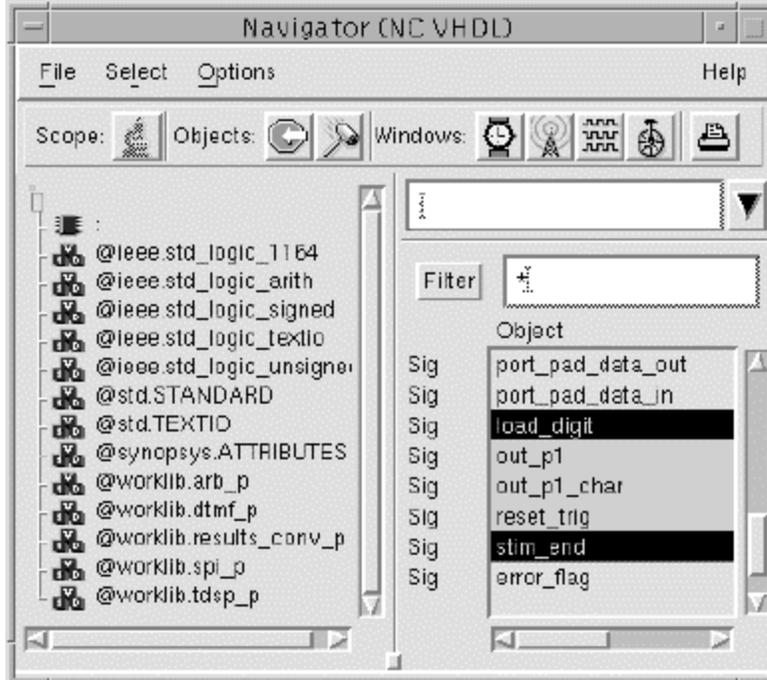


Figure 4. Navigator Window

2. In the Objects List window, select the signals `load_digit` and `stim_end`. Select the first signal with the left mouse button, and then hold down the CTRL key while selecting the second signal
3. Click the *Watch Window* button on the Navigator tool bar.



This invokes the Watch Window. The two signals that you selected from the Navigator are displayed in the Watch Window. These are the signals that you will monitor during the simulation.

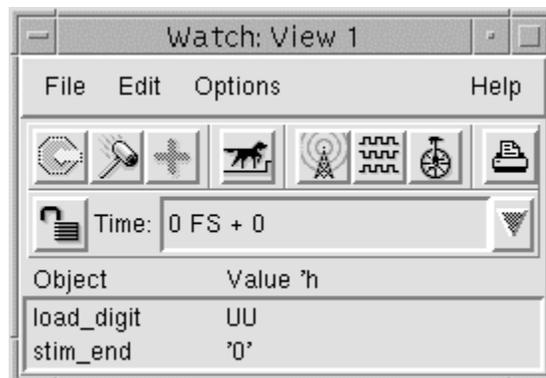


Figure 5. Watch Window

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 that are 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. Select *Options* → *Scope Tree* and then click the *instance name* 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. These objects consist of ports represented by In and Out, internal signals represented by Sig, and constants represented by Const.

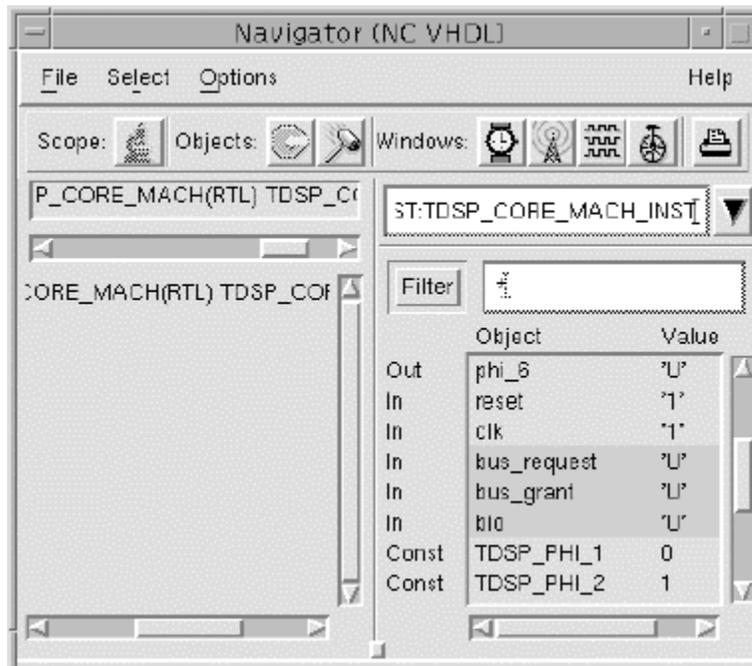


Figure 6. Objects List

6.3 Viewing Signals in SignalScan Waveform Viewer

You will now select signals that you want to probe to a database and display in the SignalScan waves waveform viewer.

1. Disable the display of constants in the Navigator Objects List window. To do this:
 - Select Options--Objects List in the Navigator.
 - 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 window.
2. Select all of the objects in the Objects List window. 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 Waveform Viewer 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.

7 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 Waveform Viewer 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 and displays 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:

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

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

8 Using SignalScan Waveform Viewer

SignalScan Waveform Viewer is a powerful tool for analyzing your simulation. It plays an important role in the verification cycle, letting you 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. To view other parts of the simulation:

1. Click the *ZmOutXFull* button on the waveform window 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.

The following part of this section describes some important features of SignalScan Waveform Viewer that you can try.

Creating a Bus

Create a bus out of the `phi_#` signals. To do this:

1. Select all six of the `phi_#` signals by pressing the left mouse button and dragging the cursor over the signal names.
2. Select *Edit* → *Create* → *Bus*

This brings up a pop up window, in which you can specify the ordering of the signals in the bus and also assign a new name to the bus.

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 in the display:

1. Select a signal transition with the left mouse button. This places the primary cursor, `Cursor1`, at that transition.

2. Place a baseline cursor, Cursor2, by clicking the middle mouse button on a second transition.
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.

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 Waveform Viewer, and NCLaunch windows.

9 Reference

1. **Affirma NC VHDL Simulator Tutorial** located at </net/cadence2001/LDV31/doc/ncvhdtut/ncvhdtut.pdf>.
2. **NCLaunch User Guide** located at </net/cadence2001/LDV31/doc/nclaunch/nclaunch.pdf>.
3. **Affirma NC VHDL Simulator Help** located at </net/cadence2001/LDV31/doc/ncvhdl/ncvhdl.pdf>.
4. **Affirma NC Verilog Simulator Help** located at </net/cadence2001/LDV31/doc/ncvlog/ncvlog.pdf>.
5. **SignalScan Waves User Guide** located at </net/cadence2001/LDV31/doc/signalscanwaves/signalscanwaves.pdf>.