Chapter 12 Integration with Third-party Tools

Introduction

The Undertow-DFII Integration kit integrates the Undertow *waveform* viewer into Cadence dfII. Cadence users can use this integration kit, to probe schematics and text in the Undertow *waveform* viewer; this integration helps you to debug your designs within the Cadence Composer environment. Before you can use this integration, you must define a successful OSS netlist directory, within the integration environment, for a target digital or analog simulator.

Veritools has successfully tested, and fully supports, integration with the following target simulators:

- Cadence Verilog-XL, NC-Verilog
- Antrim

Undertow-DFII Integration Kit

The integration kit includes the following two files:

Undertow.Init:

```
unless( boundp('Undertow_loaded)
        Undertow_loaded = t
        cond(
             ( (simUndertowSeDir = getShellEnvVar("simUndertowSeDir"))
               && stringp(simUndertowSeDir)
               printf("Undertow: simUndertowSeDir=%s\n" simUndertowSeDir)
              )
              ( t
                  simUndertowSeDir = prependInstallPath( "local/undertow"
```

```
)                       )
        ) ;cond

      if( isDir(simUndertowSeDir)
        then
            load( strcat( simUndertowSeDir "/undertow.ile"))
            Undertow_loaded = simUTConfig()
        else
           println("ERROR - environment variable 'simUndertowSeDir' is not
                        set properly, not able to load
                        undertow.ile from %s" simUndertowSeDir)
       Undertow_loaded = nil
        ) ; endif
) ;unless
```

Append this file into your Cadence `.cdsinit` file, so that dfII downloads the Undertow-dfII Skill automatically.

undertow.ile:

This is the encrypted Skill for the Undertow-dfII UI integration environment.

Installation

1. Create a new directory:

   <cds_installed_hierarchy>`/local/undertow`

2. Place the undertow.ile file under this new directory.

Environment Setup

1. Set the following for Undertow, in your `.cshrc` file:

   setenv UT_ROOT_DIR undertow_install_directory_path

   setenv UT_WORK_DIR undertow_working_directory_path

setenv XENV $UT_ROOT_DIR/XENV

2. Append the `Undertow.init` file into your `.cdsinit` file, to make sure that the Undertow encrypted Skill file loads automatically, whenever you start dfII.

3. Add the following line into your `verilogI.ini` file for your Undertow-Verilog Integration environment.

   Note: Add this line to the bottom of a Skill procedure named `vlogifCreatePW()`.

   Using this additional Skill call, the Verilog-XL Integration Encapsulation window automatically loads the **Undertow** menu into the banner menus.

```
The “verilogI.ini” should be located under your “<cds_install_hierarchy>/tools/dfII/etc/context”. Please ask your CAD engineer for help if you don’t have permission to modify the file.
“verilogI.ini”
........
procedure( vlogifCreatePW( @optional (winCoord nil) )
...........
veritoolVlogAddUT()
return( t)
......
)
.......
```

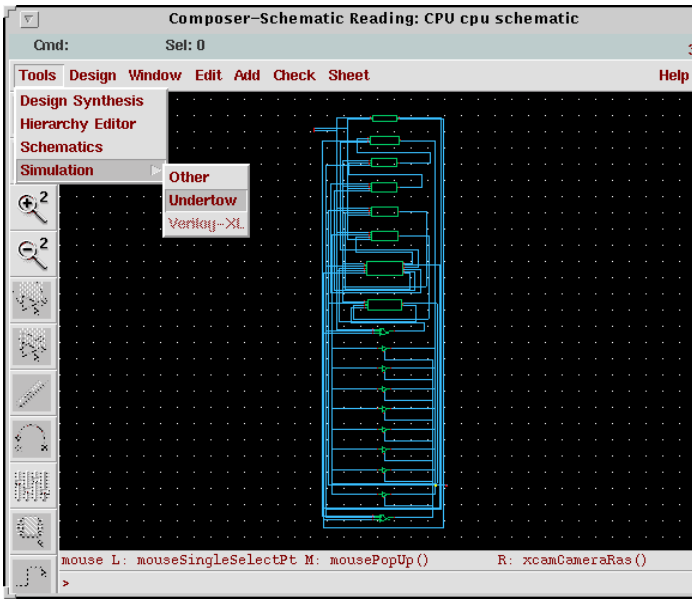
Standalone Usage Of Undertow-Cadence DFII Integration

This section describes how to use Undertow to display signals in a stand-alone Cadence Composer environment.

Usage

1. Open a schematic or Verilog text cell, from either the Library Manager or the ***CIW*** => ***File*** => ***Open*** menu item.

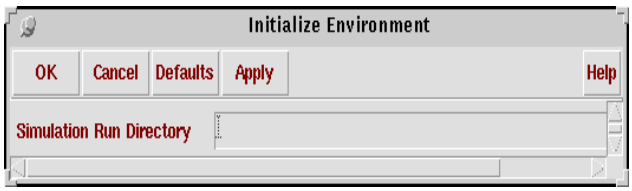
2. Select the ***Tools*** => ***Simulation*** => ***Undertow*** menu item.



A form opens, and asks you to specify the name of a pre-existing **OSS Simulation Run** directory.

3. If you have not already done so, create the OSS run directory.

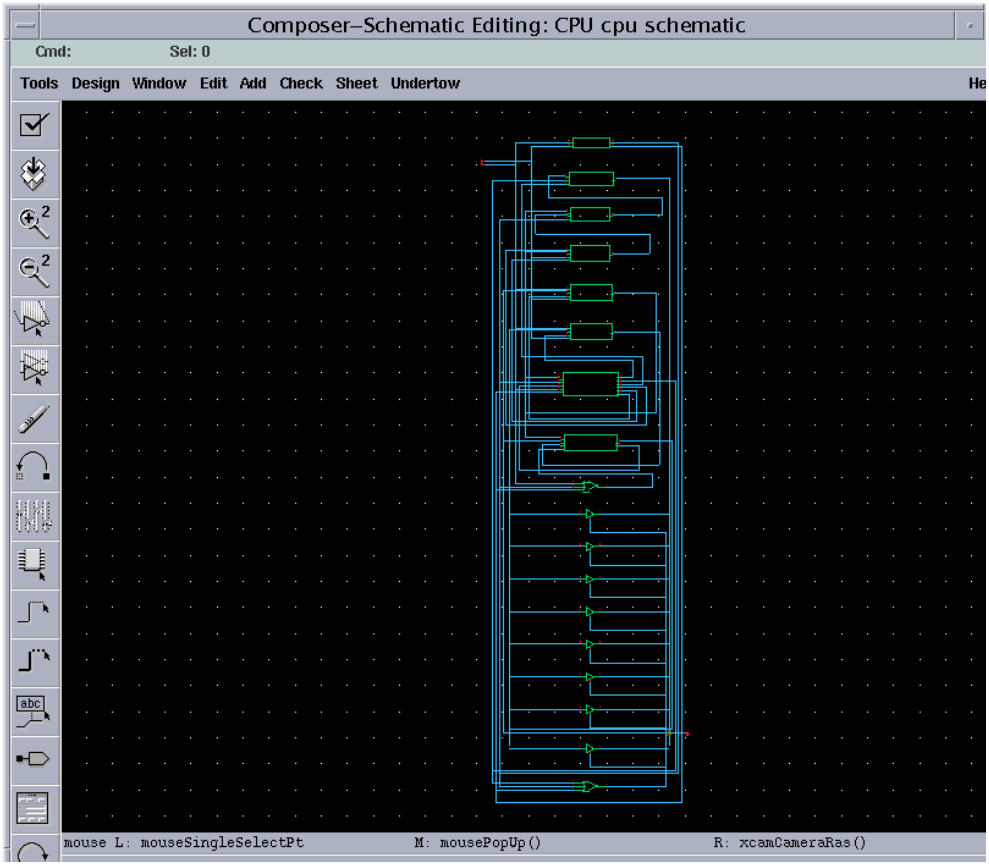
4. Make sure that both the results, and the netlist, that the netlist Integration environment generated for the target simulator, are in this directory.



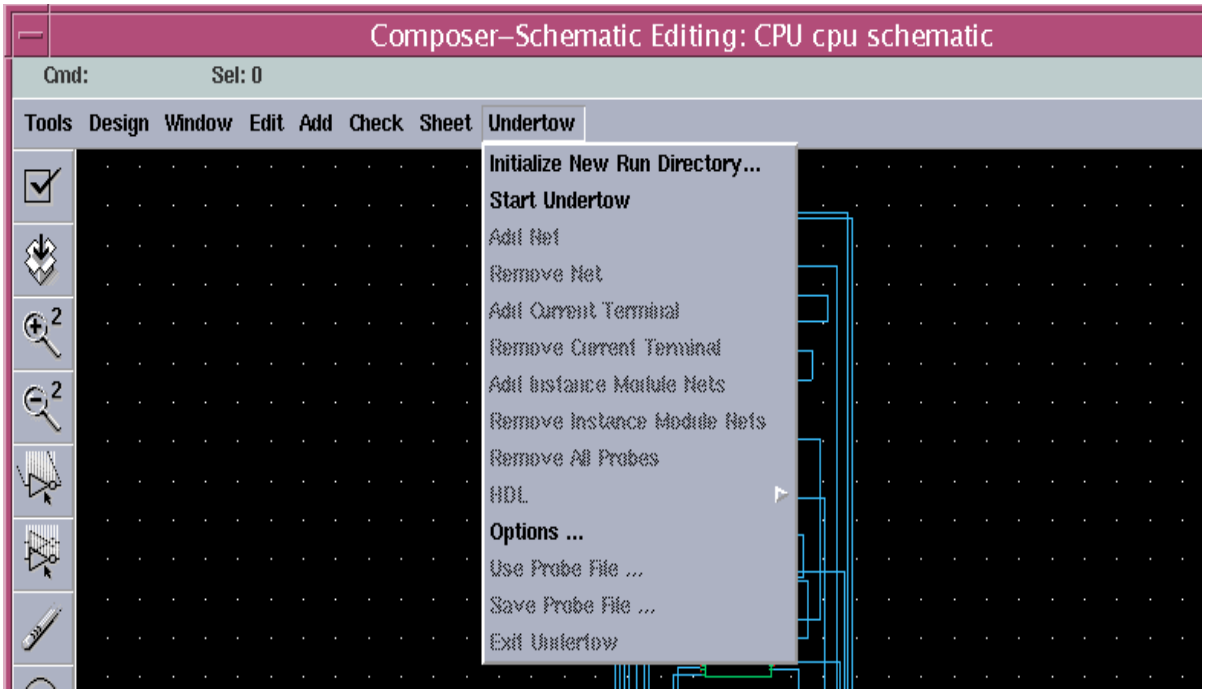
This release supports both **Buses** netlist and **Bit** netlist modes for the Verilog Integration environment.

Two new menus appear in the banner menus of either the Composer schematic window, or the Verilog-HDL window:

- **Run directory**
- **Undertow**



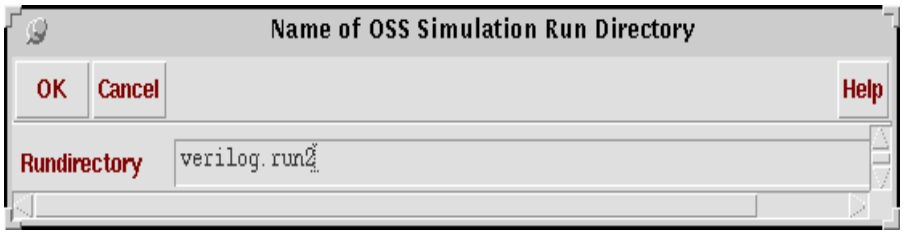
The **Undertow** menu contains the following options:



- **Initialize New Run Directory ...**

This menu item pops up the **OSS Simulation Run Directory** form.

Use this form to switch to a new pre-existing **OSS Simulation Run** directory.

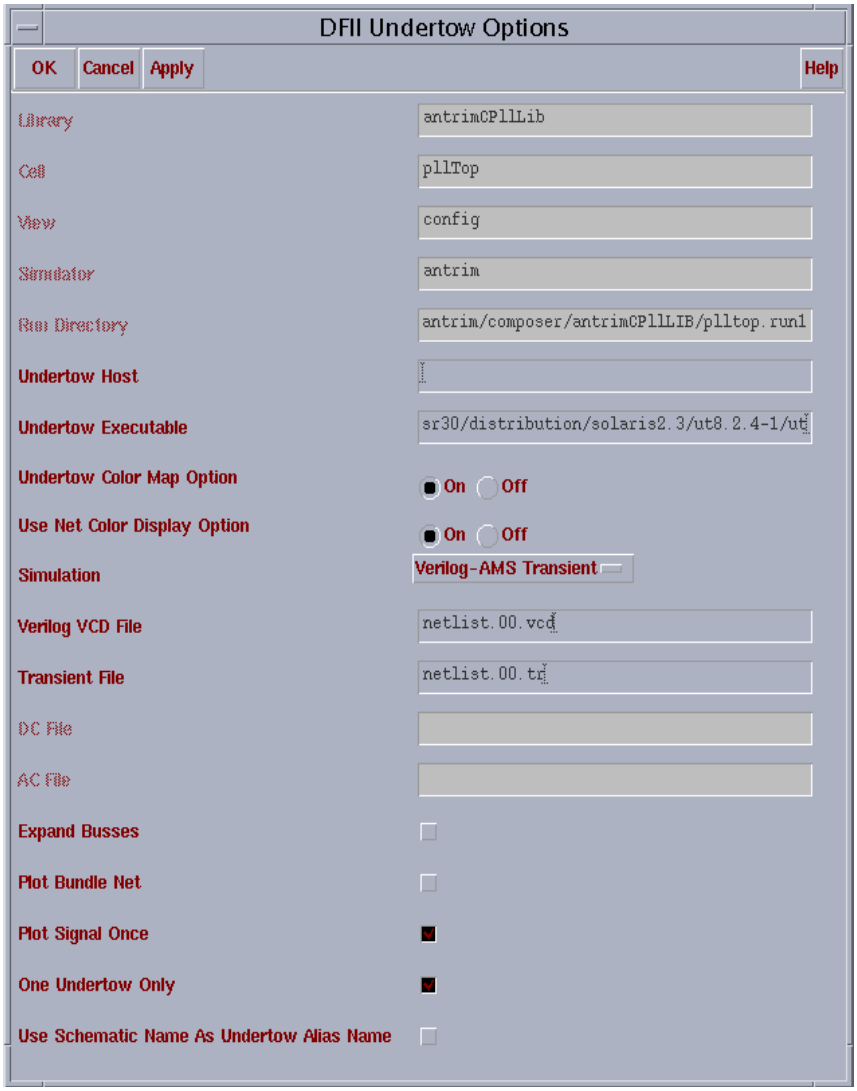


- **Start Undertow**

  This menu item invokes Undertow through a Skill IPC process. It also enables those probing fields.

- **Options...**

  This menu item pops up a form, in which you set options to customize the Undertow-dfII integration environment:



- **Library**

  This field displays the library name, captured from the specified **OSS** netlist directory.

  Note: This is a grey-out (inactive) field.

- **Cell**

  This field displays the cell name, captured from the specified **OSS** netlist directory.

  Note: This is a grey-out (inactive) field.

- **View**

  This field displays the view name, captured from the specified **OSS** netlist directory.

  Note: This is a grey-out (inactive) field.

- **Run Directory**

This field displays the run directory name specified from the initialization.

Note: This is a grey-out (inactive) field.

- **Undertow Host**

  This field indicates the name of the host on which Undertow will run. By default, Undertow runs on your local host.

- **Undertow Executable**

  This field indicates the full path to the Undertow executable. To set this field, use the **UT_ROOT_DIR** UNIX environment variable.

- **Undertow Color Map Option**

  When you toggle this option to **On**, Undertow allocates resources to the color display.

- **Use Net Color Display Option**

  By default, this option is **On**, so it always uses the color of the probed net, as the displayed color of the waveform. If you set this option to **Off**, Undertow uses the signal's defined color to plot each signal.

- **Simulation**

  Use this pull-down menu to specify the types of simulation to run. Supported simulation types are:

  - Analog Transient
  - Analog AC
  - Analog DC
  - Verilog-AMS Transient
  - Verilog

  If you are running Antrim Analog DC simulation, select the **Analog DC** simulation mode.

  If you are running Antrim Analog AC simulation, select the **Analog AC** simulation mode.

  If you are running Antrim Analog Transient simulation, select the **Analog Transient** simulation mode.

- **Verilog VCD File**

This field specifies the name of a Verilog digital simulation file, such as a **vcd** file or a Veritools supported dump file. You must specify this field if you selected **Verilog-AMS Transient** or **Verilog** in the simulation field.

*Note: This field is greyed-out (inactive) if the simulation field is neither **Verilog-AMS Transient** nor **Verilog**.*

- **Transient File**

This field specifies the name of the analog Transient file. You must specify this field if you selected **Verilog-AMS Transient** or **Analog Transient** in the simulation field.

*Note: This field is greyed-out (inactive) if the simulation field is neither **Verilog-AMS Transient** nor **Analog Transient**.*

- **DC File**

This field specifies the name of the analog DC file. You must specify this field if you selected **Analog DC** in the simulation field.

*Note: This field is greyed-out (inactive) if you did not select **Analog DC** in the simulation field.*

- **AC File**

This field specifies the name of the analog AC file. You must specify this field if you selected **Analog AC** in the simulation field.

*Note: This field is greyed-out (inactive) if you did not select **Analog AC** in the simulation field.*

- **Expand Buses**

If this button is **ON**, and if you generated the netlist in a **Buses** netlist mode, then Undertow expands all bus signals that you selected from the schematic probing window, into individual bits, for display in Undertow.

- For example, the **data[2:3]** signal displays as data[2] and **data[3]** in Undertow, if the button is turned *On*.
- By default, the button is turned “Off”.

If you used the **Bit** netlist mode in the Verilog integration environment, to generate the netlist within the specified **OSS Simulation Run** directory, then this button has no effect.

- **Plot Bundle Net**

Use this button to plot a *bundle* signal as *bundle* waveform. By default, this option is set to **false**. If you turn this option **Off**, Undertow breaks the selected bundle net into separate pieces, such as buses or a scalar net, for display.

  - For example, if the option is turned **On**, then the **a<2:44>,b,c** signal displays as **a<2:4>,b,c**.
  - Otherwise, Undertow plots three signals, **a<2:4>**, **b**, and **c**.

- **Plot Signal Once**

  When you toggle this button **On**, Undertow does not plot any specified signal if the signal has already been plotted. By default, this option is set to **true**.

- **One Undertow Only**

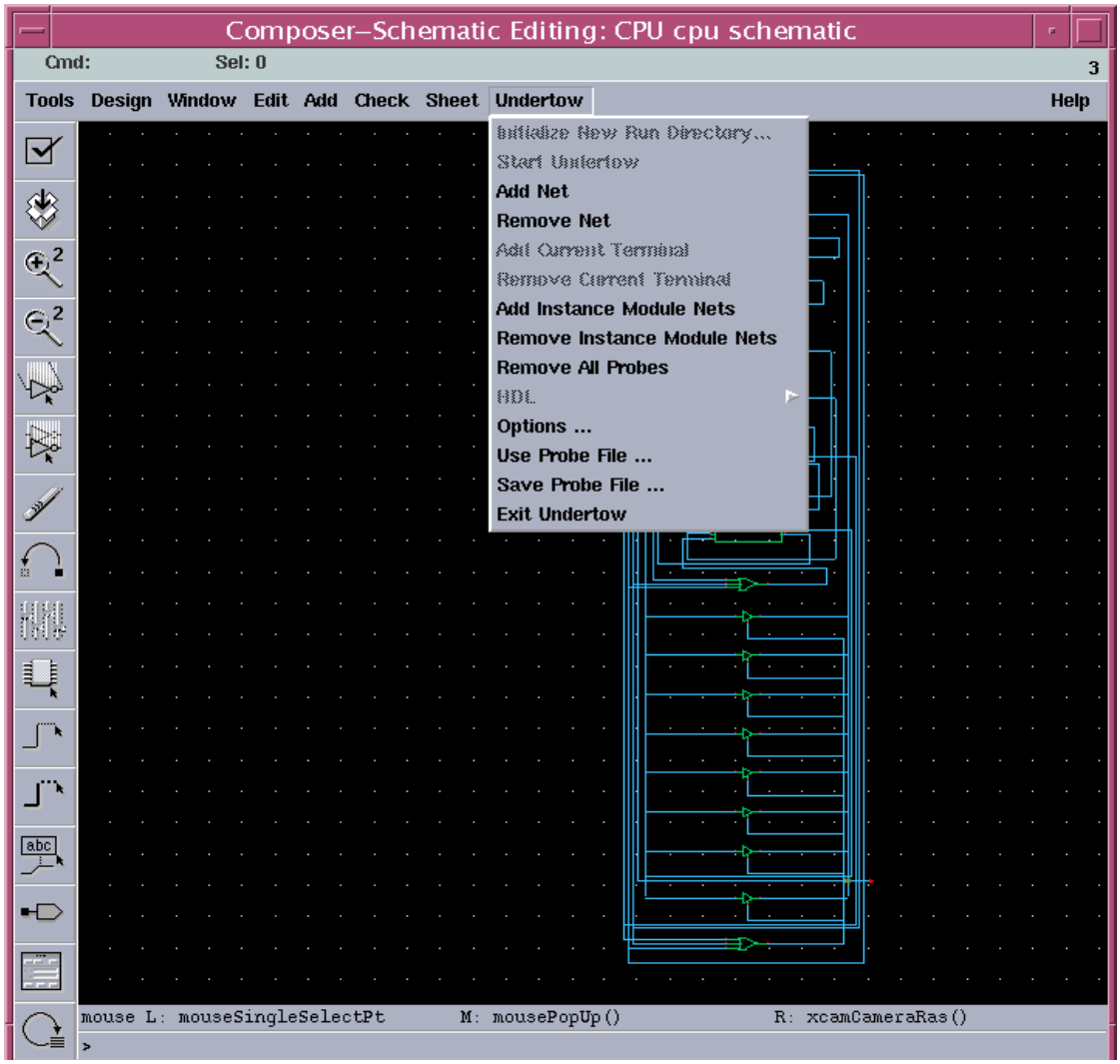
  This button limits the number of Undertow executables that can run at any one time, to one. If you start a new Undertow session while another Undertow session is still running, this option kills the previous Undertow process.

  The default is that you can run more than one Undertow process at one time, but all commands are always sent to the newest Undertow process.

- **Use Schematic Name As Undertow Alias Name**

  If this button is turned **On**, Undertow uses the schematic name space for display.

  For example, a schematic signal named **COUNTER/q<2>** maps to **test.top.COUNTER.q[2]** in a Verilog netlist. If the button is turned ON, then **COUNTER/q<2>** becomes an alias signal name for **test.top.COUNTER.q[2]**. The plot window in Undertow shows **COUNTER/q<2>**.



After you select **Start Undertow**, the integration kit starts an Undertow **waveform** viewer through a Skill IPC process. When the Undertow **waveform** opens, some pull-down fields are enabled, so that you can start to probe signal waveforms from either a schematic or a text view.

- **Add Net**

  When you probe a net, this menu item lets you add the net to the plot in Undertow.

  After you select **Add Net**, you can use the schematic window to probe a net. If a net is properly probed, an **Add** signal message is sent, so that Undertow displays the probed net in the *waveform* window.

  This menu item is disabled if you use an HDL text viewing window.

- **Remove Net**

When you probe a net, this menu item lets you remove the net from the plot in Undertow.

After you select **Delete Net**, you can use the schematic window to probe a net. If a net is properly probed, a **Delete** signal message is sent, so that Undertow deletes the probed net from the *waveform* window.

This menu item is disabled if you use an HDL text viewing window.

- **Add Instance Module Nets**

  When you probe a net, this menu item lets you add referenced nets to the plot in Undertow.

  After you select **Add Instance Module Nets**, you can use the schematic window to probe a referenced net. If a net is properly probed, the Undertow *waveform* window displays all nets in the same object instance, that are referenced by the probed instance net.

  - This menu item is enabled if the simulator is **Verilog**.
  - This menu item is disabled if you use a Verilog HDL text viewing window.

- **Remove Instance Module Nets**

  When you probe a net, this menu item lets you remove referenced nets from the plot in Undertow.

  After you select **Remove Instance Module Nets**, you can use the schematic window to probe a referenced net. If a net is properly probed, the Undertow *waveform* window no longer displays any nets in the same object instance, that are referenced by the probed instance net.

  - This menu item is enabled if the simulator is **Verilog**.
  - This menu item is disabled if you use a Verilog HDL text viewing window.

- **Remove All Probes**

  Remove All Probes sends a message to Undertow, to remove all plotted signals from the schematic view. This options clears all probed nets and instances within the schematic.

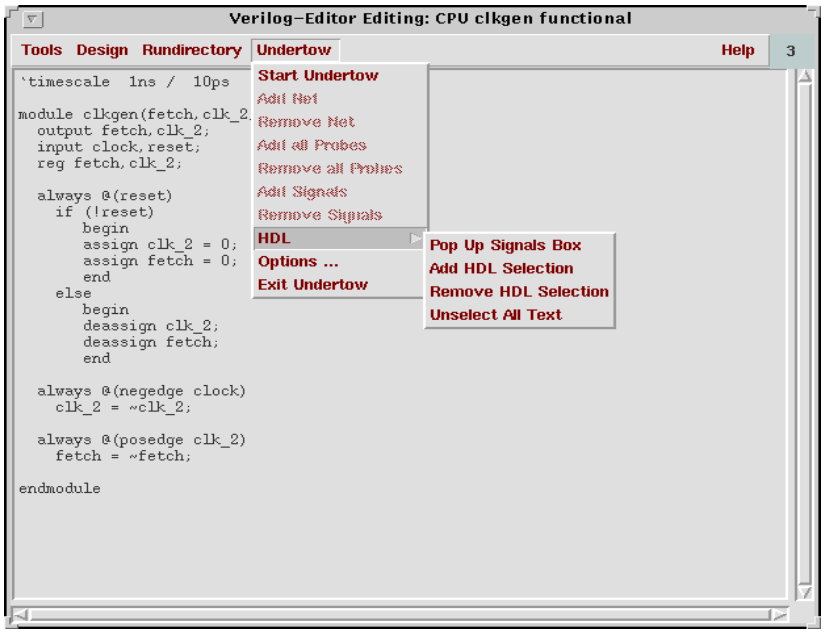
  This menu item is disabled if you use a Verilog HDL text viewing window.

- **HDL->**

  This submenu contains commands to select signals in a HDL text viewing window, such as **verilog** or **antrim**.

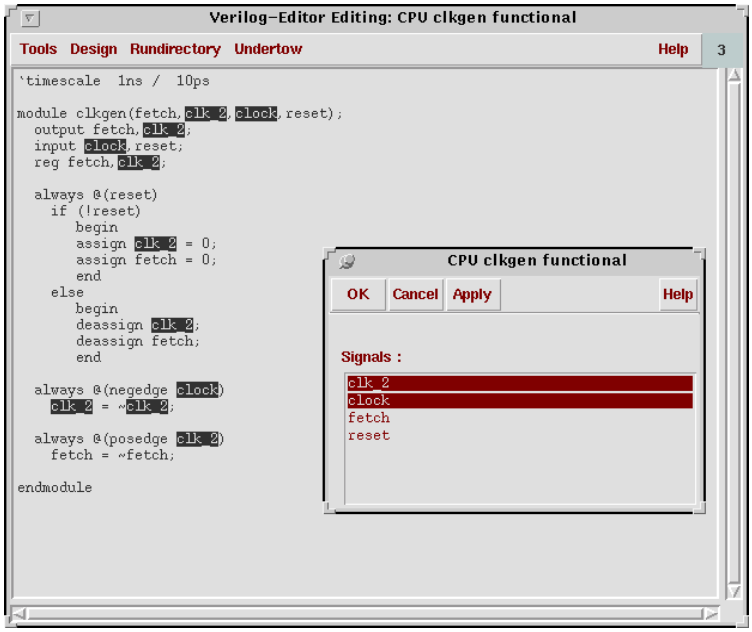
  - This menu item is disabled if the opening window is a schematic window.

- This menu item is enabled if you use a HDL text viewing window.



- **Pop Up Signals Box...**

  This menu item pops up a signal box, to display all signals for the HDL *text viewing* window. If you select multiple signals from the signal box, the form callback automatically searches and highlights all matched signals, within the HDL *text viewing* window.



- **Add HDL Selection**

  1. Either manually highlight a signal, or use the Text Signal Box option.
  2. Select **Add HDL Selection**.

     Undertow displays all selected text(s).

- **Remove HDL Selection**

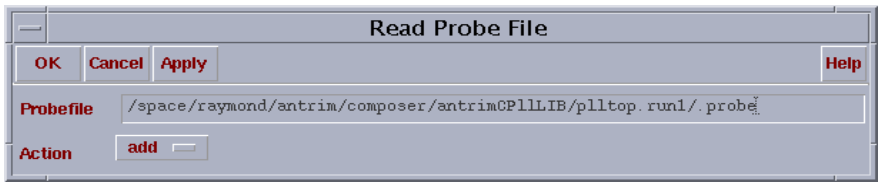
  1. Either manually highlight a signal, or use the Text Signal Box option.
  2. Select **Remove HDL Selection**.

     Undertow removes all selected text(s).

- **Unselect all Text**

  Clears all text signal selections.

- **Use Probe File...**



  You can specify a probe file to add or delete from the schematic view.

  If you select the **add** operation, the integration skill reads the probe file, to probe any missing net/instance to the schematic view. It also sends a message to Undertow, to plot the waveform.

  This menu item is disabled if you use an HDL text viewing window.

- **Save Probe File...**

  You can save the current probed instance and net, from the schematic view.

  This menu item is disabled if you use an HDL text viewing window.

- **Exit Undertow**

  This menu item exits Undertow.

Instance Hierarchy Traversal

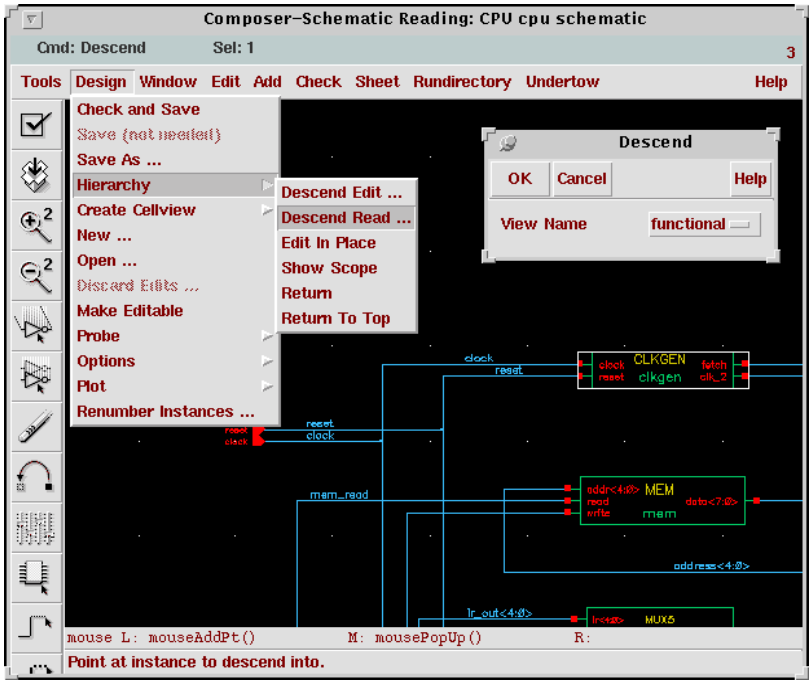
1. Open a schematic or Verilog-HDL text viewing window, from the topmost cell of your OSS simulation run directory.

   This step ensures that the OSS environment successfully determines the instance hierarchy.

2. To traverse the design hierarchy, select **Composer => Design => Hierarchy => descend Read**.

3. Specify the right switching view for a selecting instance.

   This step ensures that the OSS environment determines the instance hierarchy for OSS name mapping.



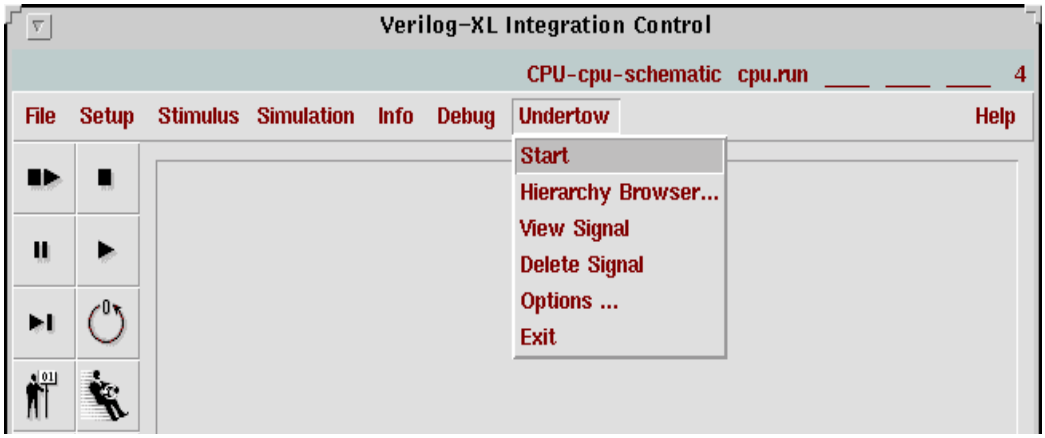
Using Undertow in the Cadence DFII Verilog Integration Environment

This section describes how to use Undertow to display signals within the Verilog Integration environment.

Usages

1. If you have not already done so:
   - Run the Verilog netlister, to generate the netlist, and
   - Run the simulation, to generate the Verilog VCD file.
2. Open a Verilog-XL **Integration Control Encapsulation** window.
3. Specify an **OSS Simulation Run** directory.
   - The **Encapsulation** window opens.
4. Make sure that:
   - The **Encapsulation** window shows simulation results, in Verilog VCD format, and
   - The netlists have been generated.

   *Note: This release supports post-simulation probing **only** for Undertow. If you run Verilog-XL simulation in interactive mode, within the Verilog Integration environment, Undertow is not supported.*
5. Verify that a new menu, named *Undertow,* appears in the banner menus of the Verilog-XL **Integration Control Encapsulation** window.



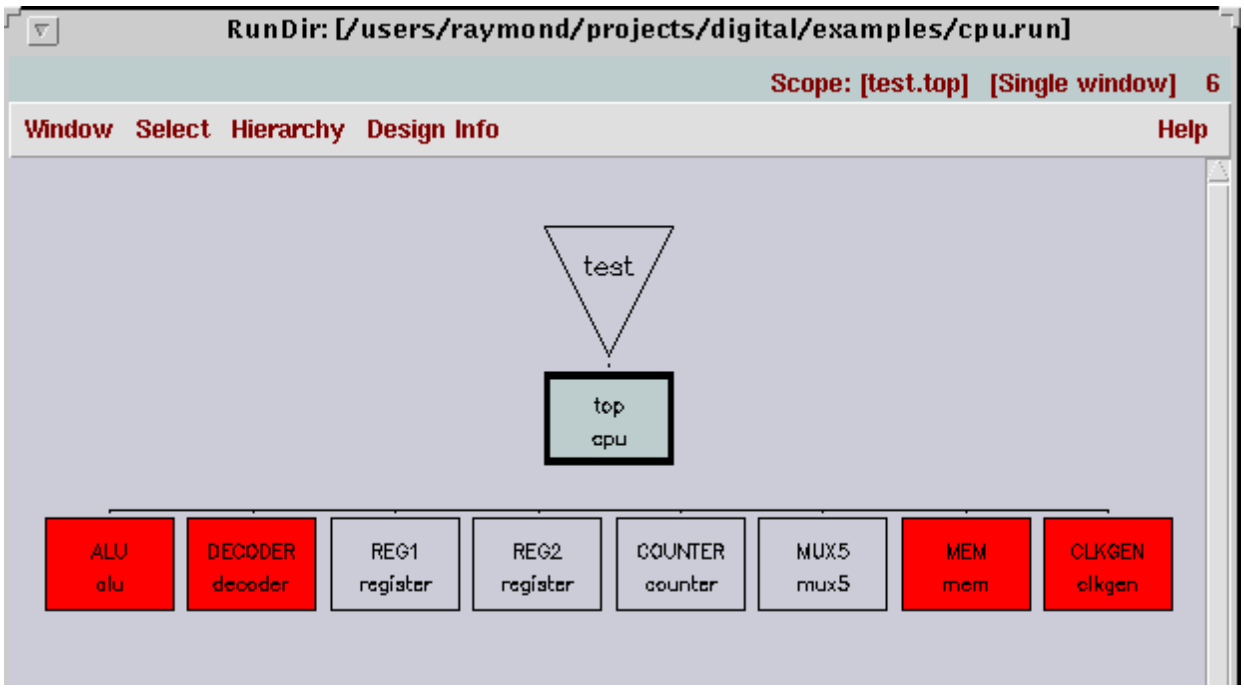
The Undertow menu contains the following menu items:

- **Start Undertow**

  This menu item invokes **Undertow**, through a Skill IPC process.

- **Hierarchy Browser...**

  This menu item pops up the **Hierarchy Browser** of the Verilog Integration.



- **View Signals**

  1. From the **Hierarchy Browser**, open either the *schematic viewing* window, or the Verilog-XL-HDL *text viewing* window.

2. Select signal(s), from either a schematic viewing window, or the Verilog-XL-HDL text viewing window.

3. Select **View Signal**.

   The Undertow window displays the selected probed signals.

- **Delete Signals**

  1. From the **Hierarchy Browser**, open either the *schematic viewing* window, or the Verilog-XL-HDL *text viewing* window.

  2. Select signal(s), from either a schematic viewing window, or the Verilog-XL-HDL text viewing window.

  3. Select **Delete Signal**.

     The Undertow window no longer displays the selected probed signals.

- **Options...**

  This menu item pops up the same options form described in **Standalone Usage** mode. Refer to Section , “Standalone Usage Of Undertow-Cadence DFII Integration”.

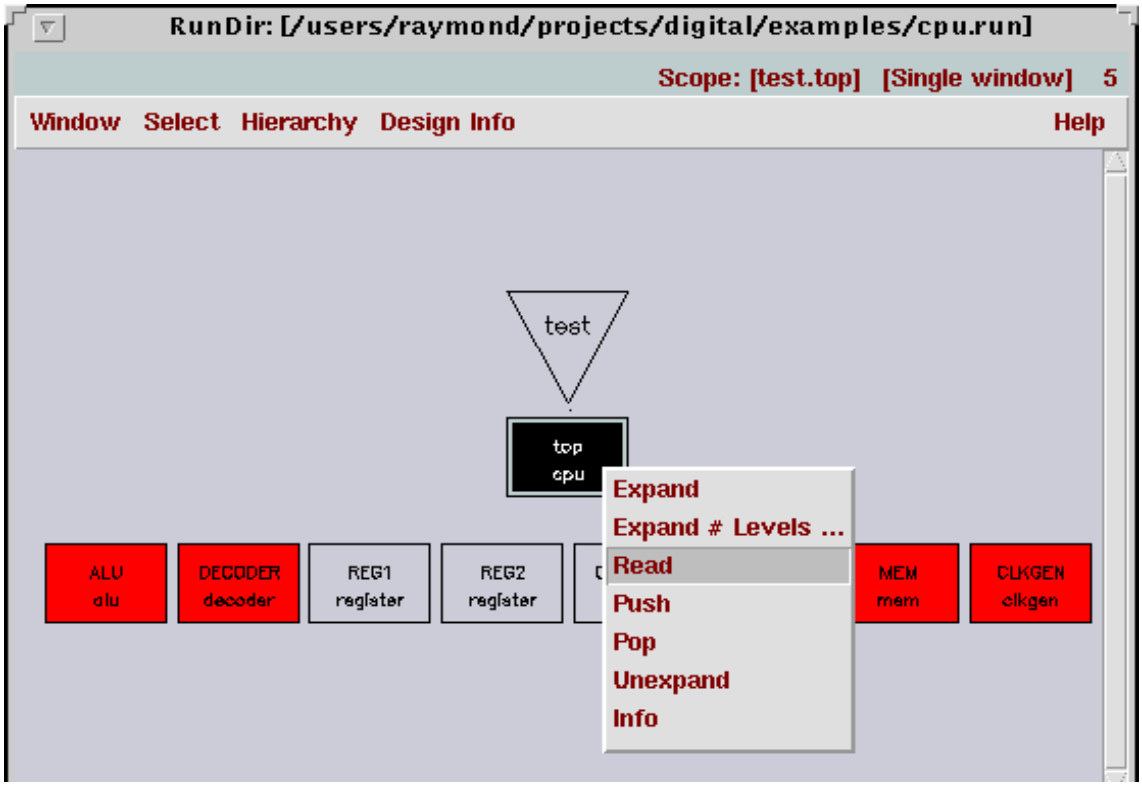
- **Exit**

  This menu item exits Undertow.

Signal Probing In Verilog Integration Environment

1. From the **Hierarchy Browser**, open either the *schematic viewing* window, or the Verilog-XL-HDL *text viewing* window.

2. Select signal(s) to probe, from either a schematic viewing window, or the Verilog-XL-HDL text viewing window.



Preparing a Verilog VCD File For Undertow

Add the following lines into your Verilog stimulus file:

```
initial
  begin
    ....
    $dumpfile("verilog.vcd");
    $dumpvars;
    ......
  end
```

These lines enable Verilog-XL to generate simulation results in Verilog VCD format.

After the Verilog-XL simulation finishes, a simulation Verilog VCD file, named *verilog.vcd*, appears in the *run* directory.

Preparing Veritools dump file for Verilog-XL

Add the following lines into your Verilog stimulus file:


```
initial
  begin
    ....
    $vtDumpvars;
    ......
  end
```

These lines enable Verilog-XL to generate simulation results in Veritools *dumpfile* format.

Note: Refer to , for more details about how to use the Veritools PLIs.

Preparation Of Verilog Netlists In Verilog Integration Environment

- ***OSS "si" mode***

  1. Set up an OSS initialized file named `si.env`
  2. Copy the Cadence `cds.lib` file into the simulation `run` directory.
  3. Do either of the following:
     - Set up a OSS standalone **SI** batch call, or
     - Invoke the **si** binary, then type **netlist()**.

- ***Verilog Integration Environment Interactive Mode***

  1. Open the Verilog-XL Integration **Encapsulation Control** window.
  2. To trigger the Verilog HNL netlister, either:
     - Press the **Start Interactive** icon from the **Encapsulation** window, or
     - Type **netlist()** in CIW, to generate a Verilog netlist for the design.

Preparing Netlists In Antrim Integration Environment

Refer to Antrim's manual for how to prepare netlists for the Antrim simulator.

Interface With Undertow From Other Applications

This section describes how to interface other applications with Undertow.

Invoking Undertow From Other C Applications:

Start Undertow with command line arguments:

1. Use the following command line argument to start Undertow:

   "ut -v filename.vcd -P 0/1" for stdin/stdout

2. Type commands into the same shell terminal that you used to start Undertow.

   Undertow reads commands from the standard input:

3. In the same terminal, type:

   ADD test.top.reset

   ADD test.top.COUNTER.q COUNTER/q

   Undertow displays **test.top.reset** and **COUNTER/q** in the plot window.

4. Use the same mechanism, *cat* a file into **ut** as input:

   "cat commands.txt | ut -v filename.vcd -P 0/1"

   where the `commands.txt` file stores all of the Undertow-recognized commands, line by line.

Set up a link between Undertow and Application by a C

routine:

A parent program (your **C** application) can set-up pipes, for input/output with Undertow. It can then *fork/exec*, with the command line arguments, for the *fileno()* of the Unix pipes:

fork() => exec()

with

"ut -v filename.vcd -P fileno(in_pipe)/fileno(out_pipe)”

0.0.0.1 C Example:

```
void InvokeUndertow()
{
 char *argv[4];
 char *command;
 int in_fds[2]; /* incoming file descriptor */
 int out_fds[2]; /* outgoing file descriptor */
 int pid;
 strcpy(buf0, "ut");
 argv[0] = buf0;
 strcpy(buf1, "-p");
 argv[1] = buf1;
 sprintf(buf2, "%d/%d", out_fds[0], in_fds[1]);
 argv[2] = buf2;
 argv[3] = NULL;
 envp[0] = NULL;
 if (!(pid = fork())) {
 close(in_fds[0]);
 close(out_fds[1]);
 /* command - full path for ut executable */
 execve(command, argv, environ);
 perror(argv[0]);
 exit(-1);
 }
 close(in_fds[1]);
 close(out_fds[0]);
 /* in_fds[0], out_fds[1], pid where
 pid - will be the process id for future communication
 in_fsd - file descriptor the UT will write it to
 out_fds - file descriptor the application will write the
command to UT
 */
 /* Example to write command to UT :
 write( out_fds[1], &string[index], len);
 where :
out_fds[1] - write file descriptor to UT
string - command to send it to UT such as "ADD test.top.reset"
```



```
len - length of the String command
*/
}
```

Current Supported Undertow Commands

- Each command terminates with a newline (`\n`).
- Any other white space separates the arguments for a command.
- Currently, Undertow reads *only* the `in_file_desc`.
- Undertow does not write anything to the `out_file_desc`.

You can use the `in_file_desc` to send the following commands to Undertow:

ADD <options> <signal_name> [<alias_signal_name>

This command displays the specified signal, and assigns the specified name alias to the signal.

For example:

"ADD test.top.reset"

"ADD test.top.COUNTER.clear"

If you specify an <*alias_signal_name*>, the signal displays as the <*alias_signal_name*>.

For example:

"ADD test.top.reset reset"

"ADD test.top.COUNTER.clear COUNTER/clear"

ADD-ALL <options> module_name

This command displays all signals in the *module_name* module.

For example:

"ADD test.top.CLKGEN"

where *CLKGEN* is a module.

ADD-TO-OVERLAY <-color color_name> signal_name

This command uses the same arguments as the **ADD** command, but adds the signal to the selected overlay, instead of displaying it separately.

For example, adding a new overlay:

ADD -select -color Red top.a1

ADD-TO-OVERLAY -color Green top.b2

ADD-TO-OVERLAY -color Blue top.c3

For example, adding to a selected overlay:

ADD-TO-OVERLAY -color Red top.a1

ADD-TO-OVERLAY -color Green top.b2

ADD-TO-OVERLAY -color Blue top.c3

BUNDLE <options> bundle_name <signal_name> <signal_name> ...

This command creates and displays the *bundle_name* bundle, composed of the specified signals. If you do not specify any signal names, the bundle is deleted.

For example:

“BUNDLE my_bundle1 test.top.ir[5:7] test.top.dec[2:1]"

DELETE <signal_name>

This command removes all signals with the specified name, from the display Undertow window.

For example:

"DELETE test.top.reset"

"DELETE test.top.COUNTER.clear"

GET-COLOR <signal_name>

This command reports the color of the *signal_name* signal. The return format is:

"COLOR *signal_name color_name*"

For example:

"GET-COLOR test.top.reset"

Undertow returns:

"COLOR test.top.reset RED"

<options>

-color <color_name>

This option performs the selected operation, using the specified color.

*Note: Undertow currently does not support different colors for logic waveforms. You can assign different colors **only** to analog waveforms.*

-select

This option toggles the signal name on (selected), when you add the signal. Use this command option with the **ADD-TO-OVERLAY** command.

