

## Contents

### Contents

1. Introduction .....	2
2. What should we learn from this exercise? .....	2
3. Design Browser .....	2
3.1. Traversing the Design Hierarchy – How to look at more than just the top level signals.....	2
3.2. Closing the hierarchy .....	6
3.3. Alternative Selection routes .....	6
4. Source Browser Tool .....	6
4.1. Moving around the Design Hierarchy.....	7
4.2. Setting a Break point.....	8
4.2.1. Set Breakpoint on line or instance.....	8
4.2.2. Simulating with a breakpoint set.....	10
4.2.3. Debugging operations .....	11
4.2.3.1. Step and run Options .....	11
4.2.4. Create force / Release Force.....	12
4.3. Finding and highlighting the current execution point.....	13
5. Making use of the schematic trace tool .....	14
5.1. How to use it: .....	14
6. Experimentation and using the tools. ....	15

# 1. Introduction

This document is intended to cover some of the more advanced options available within the SimVision environment to help you debug your design.

All the browsers, the design browser, source code browser the schematic browser supports the same range of operations in terms of force, release, transfer to waveform viewer, trace etc. Indeed all the “browsers” support a similar range of facilities.

Given their nature there are unique functions they support however the main differences is in how they present and interpret the information you are requesting.

It is recommended that you try this out using a design with a hierarchical structure such as the D Type shift register or counter, either the behavioural or the structural versions.

The examples in this document all use a behavioural counter test bench.

*It is strongly recommended that students utilise the “Help” on the Cadence tools and in particular look at the tutorial guides for the relevant tools.*

*You should be aware that these are commercial tools and a number of the functions and features should regard as expert or advanced level.*

*Hint:*

*Remember that hovering the mouse over the icon will give you a short description of the icons function.*

*Remember also that all icons have corresponding menu items.*

## 2. What should we learn from this exercise?

We should gain the following understanding to:

- Use the source code browser
- Traverse a design hierarchy using the Design Browser
- Traverse a design hierarchy using the Source Browser
- Set break points in VHDL code to halt execution using the Source browser
- Enable/Disable break points in the source code browser.
- Single Step the VHDL code / Run for a fixed time
- Force signals to a fixed value to aid in debugging
- Release a forcing signal
- Find the current execution line in the VHDL code in the Source browser

## 3. Design Browser

### 3.1. **Traversing the Design Hierarchy – How to look at more than just the top level signals**

Normally using the SimVision Design Browser we select the top level entity and this displays the top level signals. We then transfer these into the waveform viewer.

However we can in fact traverse the entire design hierarchy and selectively display signals, variables and constants from any level of the design hierarchy. This is extremely useful as it allows us to debug the function of the VHDL by examining signals or variables level we wish, rather than only by looking at the top level nets. We can even examine individual instances of the same component.

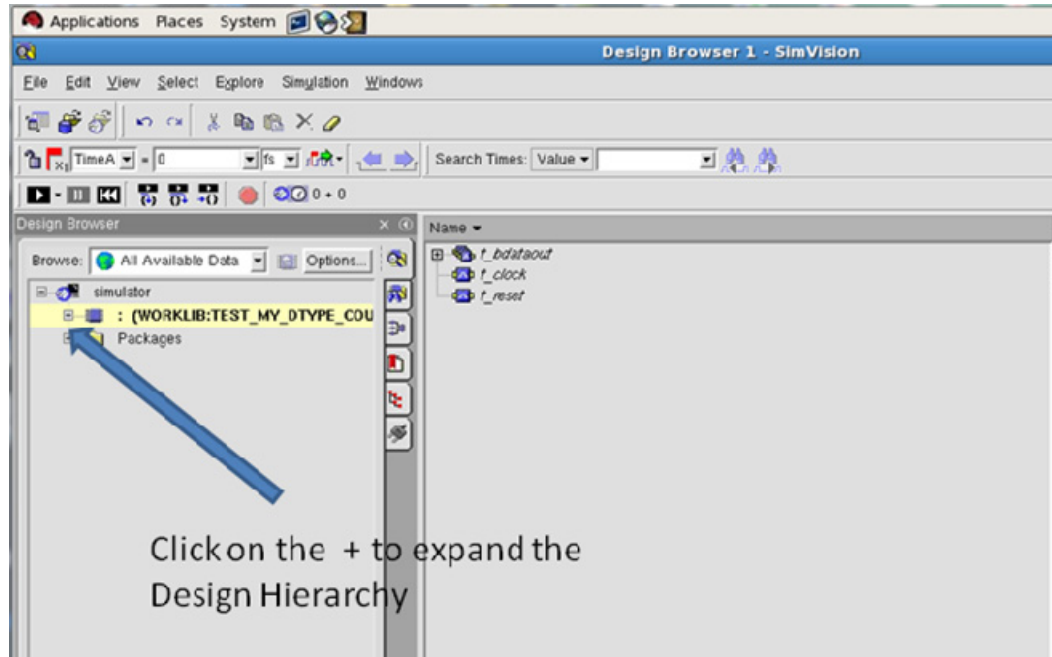


Figure 1: Design Browser Showing Top level selected and top level signals displayed

If you click the + it will expand the relevant design hierarchy. You can then select the relevant instance or process to display the signal, constants or variables associated with the selected item.

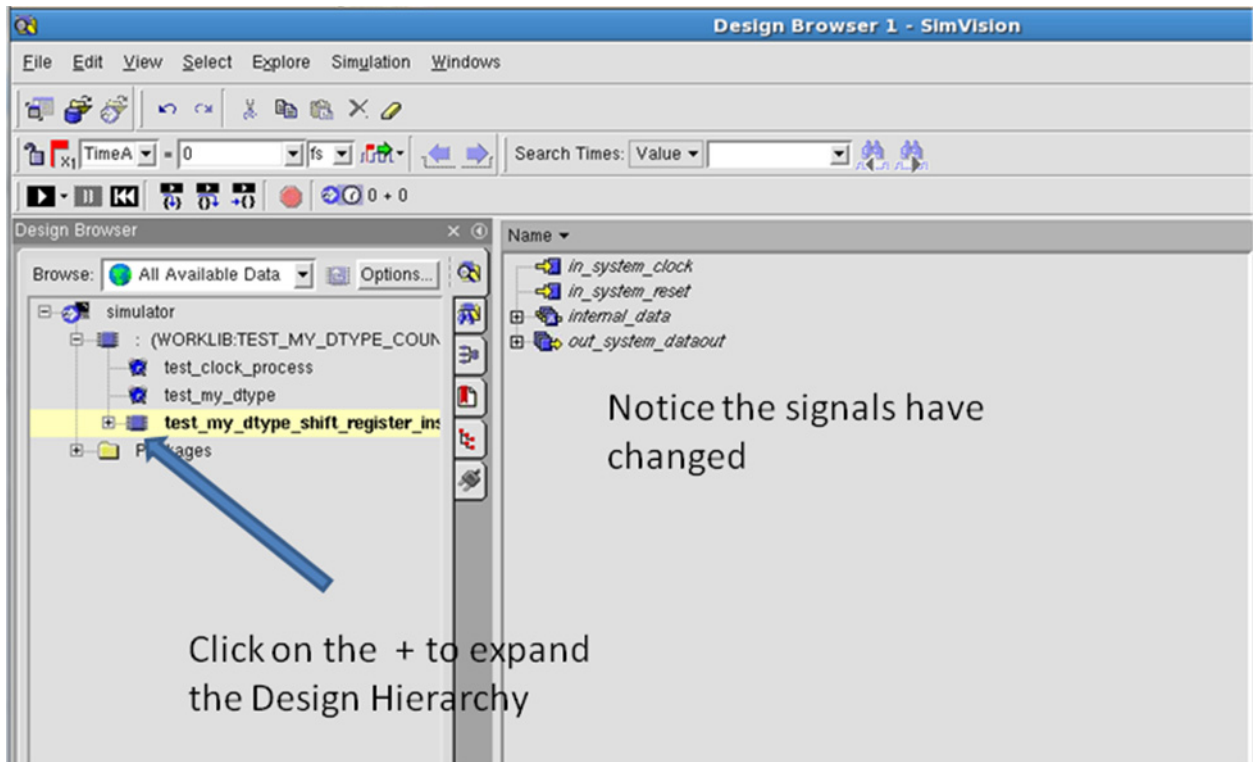


Figure 2: Showing Initial Expansion of top level

Notice that the “test\_my\_dtype\_shift\_register\_instance” has underlying hierarchy as indicated by the “+” on the RHS. If we select this level by selecting “test\_my\_dtype\_shift\_register\_instance” in the left hand pane then right hand pane will display the signals and variables displayed for this instance. Notice they are different from those on the top level.

We can repeat this operation until we reach the bottom of the design hierarchy. In this case counter is a process. By selecting the “counter” we can display the variables which are local to this process. In this case “int\_count”.

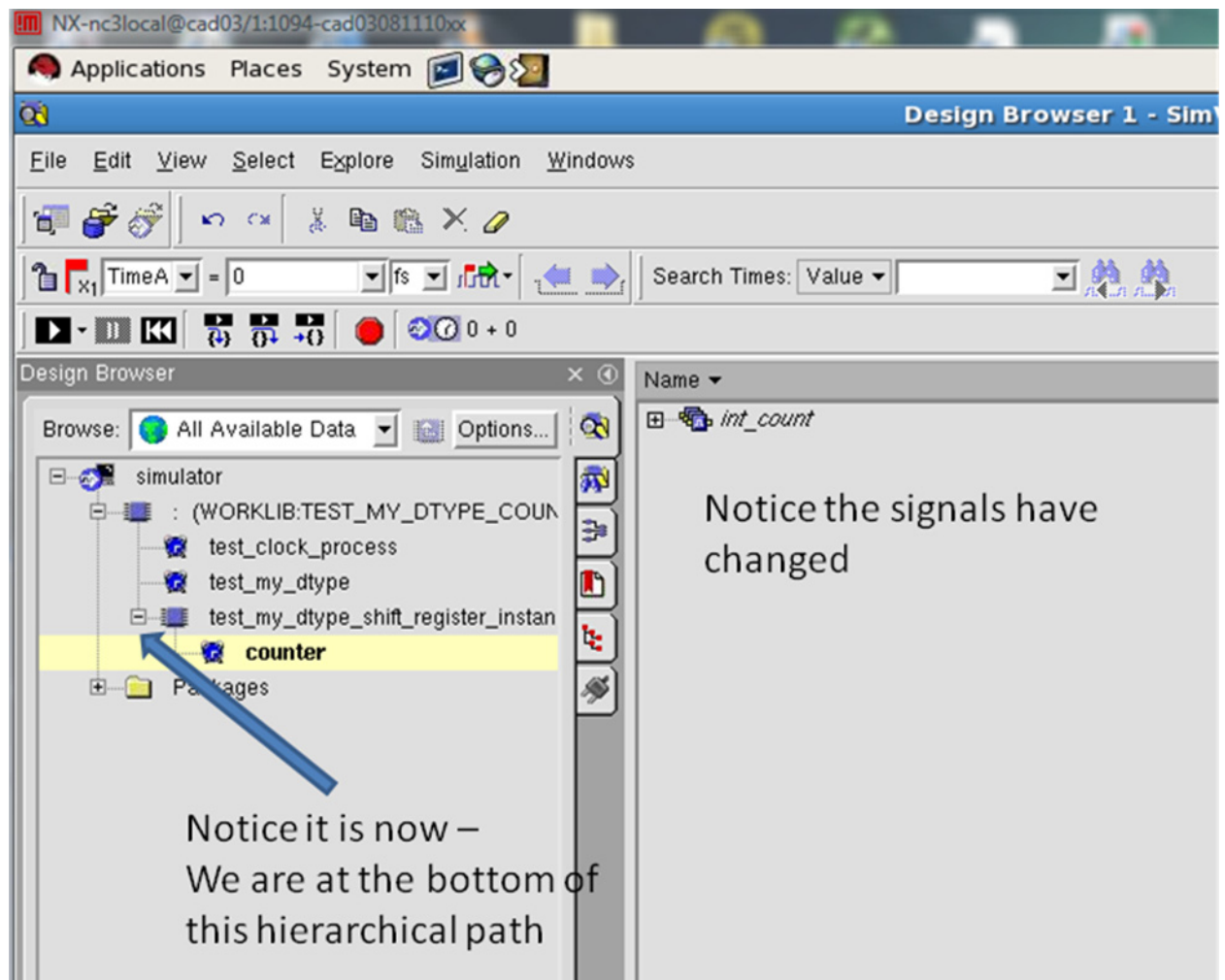



Figure 3: Showing Full Level Expansion

We can select the signals, constants or variables at any level and add them to the waveform display using the context sensitive menus, by right clicking, the menu item or the  icon.

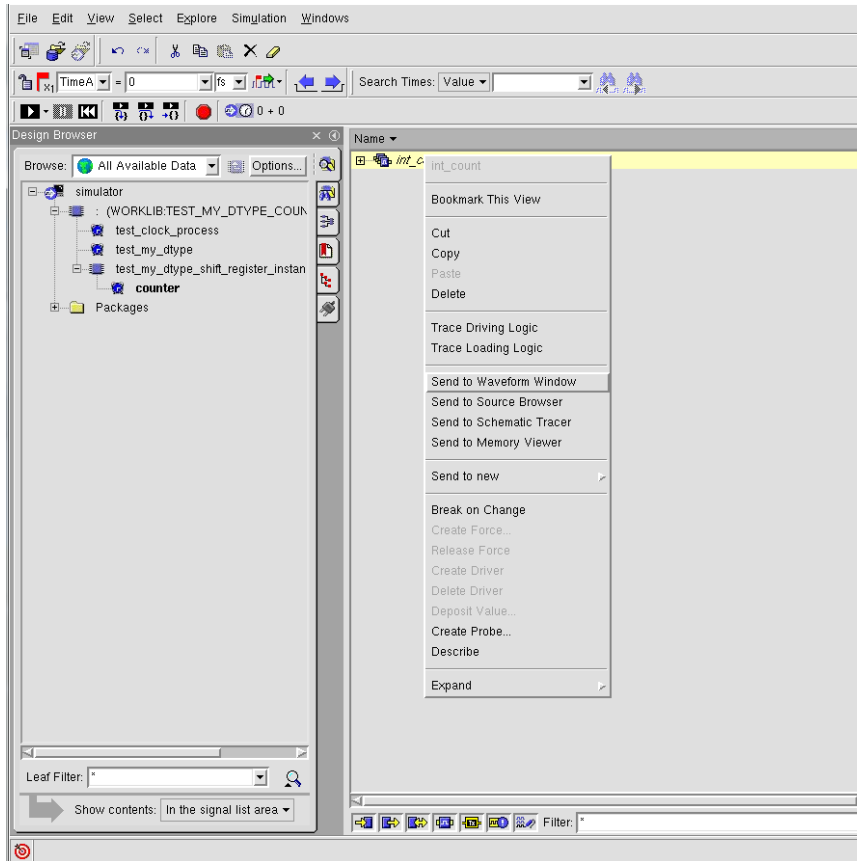


Figure 4: Selecting Signals and sending them to the waveform viewer

### 3.2. Closing the hierarchy

You can use the “-“ entry to “shut” the hierarchy.

### 3.3. Alternative Selection routes

You can select waveforms and signals using the select menu on the design browser. Select Signals Will select all signals on the currently selected level. In a similar manner the other options will limit what elements are selected. Icons on the lower left hand side of the central pane perform the same functions.

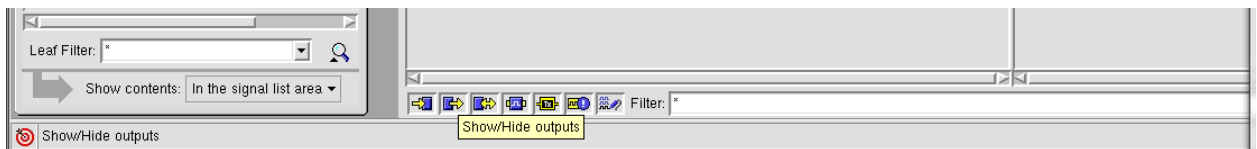


Figure 5: Showing the selection filter options.

And this can be extended by changing the filter value from “\*” to one that contains the requisite search strings. If you leave the mouse over an option the system will display the function. Both over the icon and in the panel on the lower left hand side.

## 4. Source Browser Tool

Perhaps the most useful of the other tools like the tools we use to analyse computer programmes, of which the VHDL simulation can be considered one there is also the option of stepping through the program using the “Source Code Browser”. This contains many of the features we are used to in a standard code debugger. All of which can be useful when debugging a complex structure.

## 4.1. Moving around the Design Hierarchy

The level of the hierarchy the source code debugger will open up on depends on the selected level when you invoke it. So which ever element is highlighted in the left hand pane will be the level at which the source code is opened.

You can however move up and down the hierarchy as you wish.

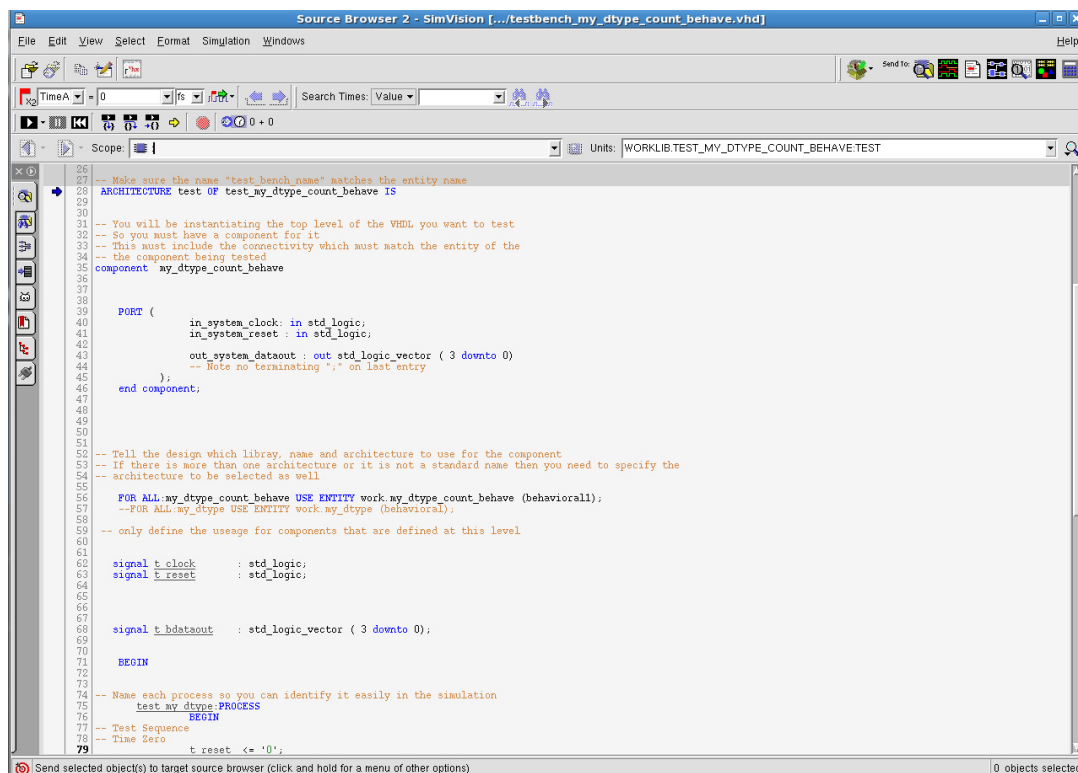


Figure 6: Source Code Debugger opened with the TEST\_MY\_DETYPE\_COUNT\_BEHAVE:TEST architecture selected.

There are a number of fields and items that need to be identified.

1. Code Execution point, identified by the blue line  
This shows at what point the execution of the code was stopped. When using break on change and other features this indicates which “break” was activated.
2. Current Scope field  
This can be used to select at where in the design hierarchy the user wishes the source browser to move to:  
For example for the current example the selections are:

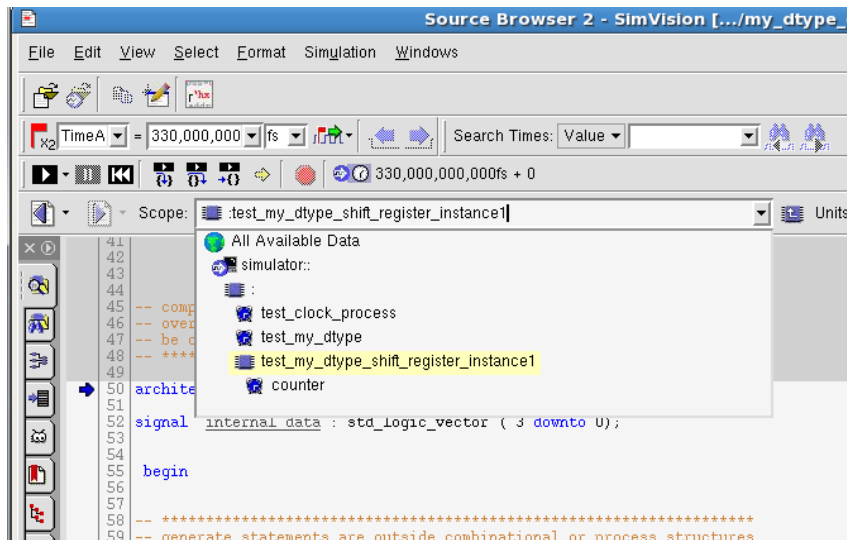


Figure 7: Scope cyclic field on Source Browser

Allowing you to traverse the design hierarchy with ease. In the example above I am viewing the source code for instance test\_my\_dtype\_shift\_register\_instance1.

## 4.2. Setting a Break point

There are a number of break operations that can be used with this tool. These are:

- Set Breakpoint on Time
- Set Breakpoint on Signal
- Set Breakpoint on Condition
- Set Breakpoint on Line
- Set Breakpoint on Process
- Set Breakpoint on Subprogram.

We will consider the following breakpoint functions

- Set Breakpoint on Line
- Set Breakpoint on Process
- Set Breakpoint on Condition

### 4.2.1. Set Breakpoint on line or instance

Select a valid line from the numbered list on the LHS of the display, valid lines are highlighted in bold. Use the right mouse button to bring up the context sensitive menu and select the break on line option. The menu will toggle to let you set or delete a breakpoint.

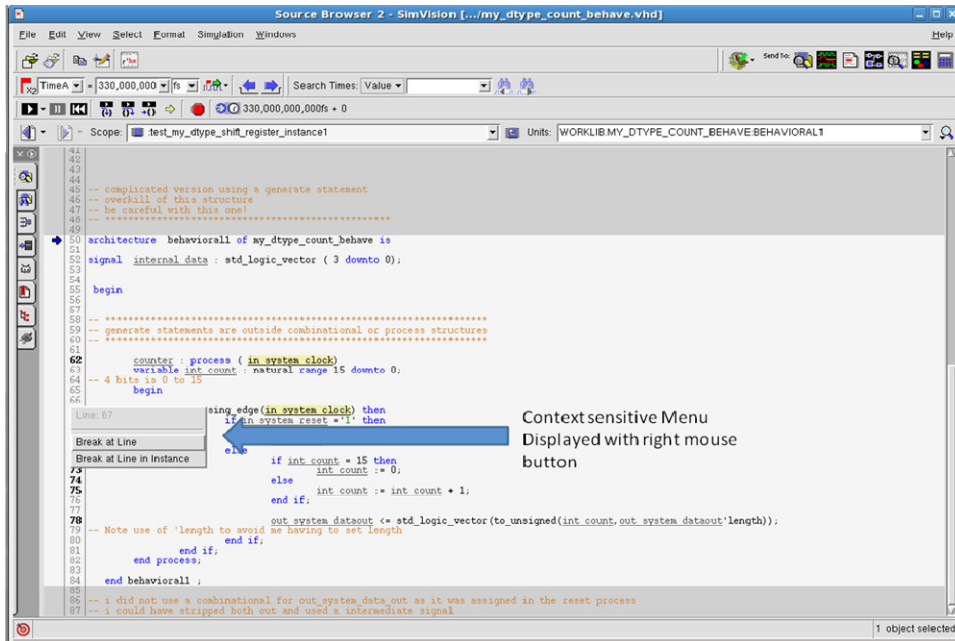


Figure 8: Setting a break point

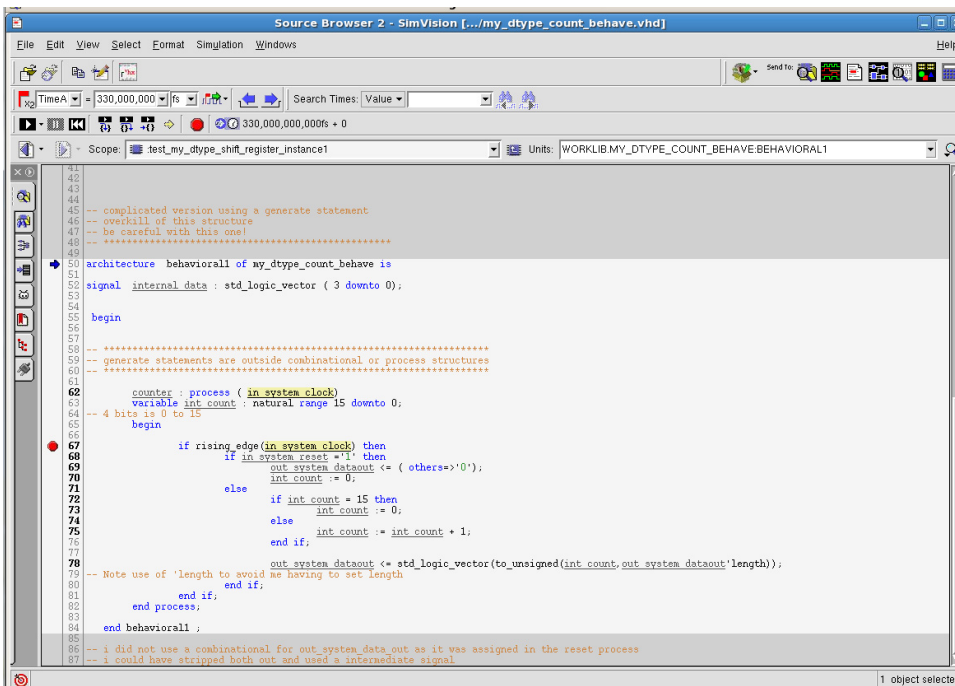


Figure 9: Red Circle indicates set breakpoint

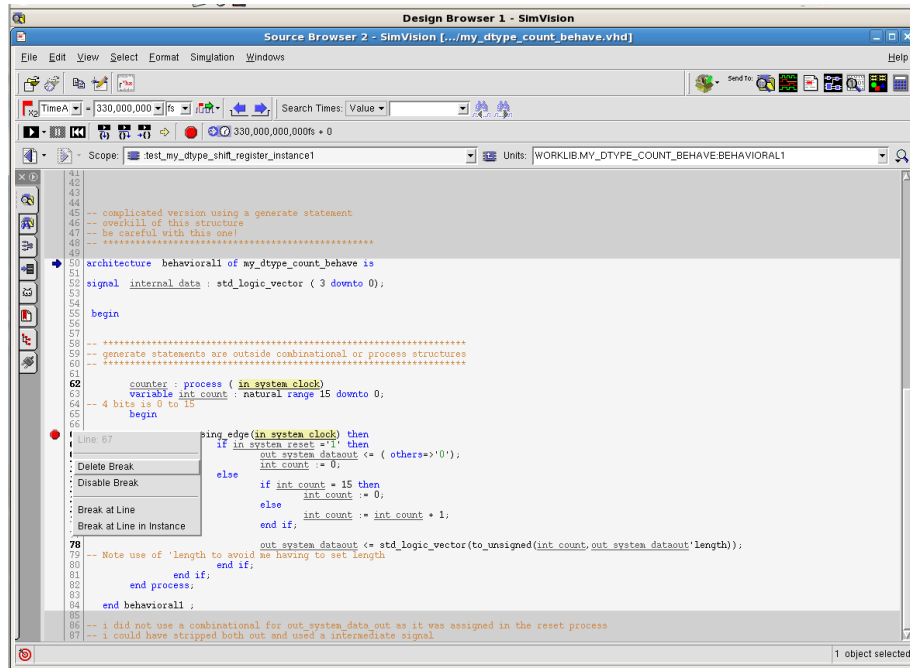


Figure 10: Removing a breakpoint

## 4.2.2. Simulating with a breakpoint set

When we run the simulation. In this case by setting a run time in the console window the simulation will run until this line is executed at which point the simulation will halt. The console window will display the location, run time, file and line number at which it has halted.

The Source browser will move so the active break point is visible and an arrow will indicate which break point caused execution to halt.

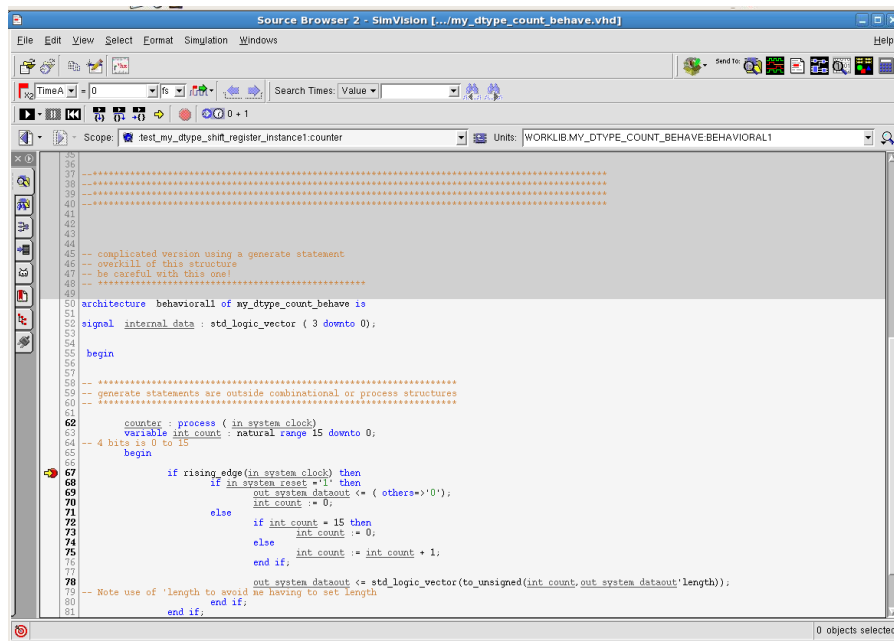


Figure 11: Source Browser Stopped at break point

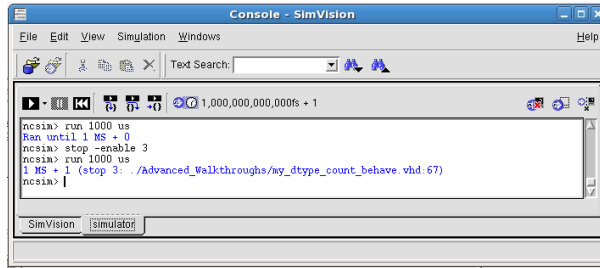


Figure 12: Console Halted by breakpoint

## 4.2.3. Debugging operations

The majority of these operations are available in the Design and the Waveform Browser. However the source code browser allow us to track execution through the impact on the code execution.

We have the usual range of options to single step and control the execution of the code available from the menu.

### 4.2.3.1. Step and run Options

There are three main step options:

- The first steps a single execution sequence.
- The second will step a single execution sequence forward but will not step into any subroutines. It maintains focus in the selected VHDL architecture.
- The third will step into any subroutines encountered and will not retain focus in the selected VHDL architecture.

There are also a number of run options

- The first will run until the next breakpoint, or end of simulation time period is encountered.
- The second will pause a running simulation.
- The last one will reset the simulation back to time 0.

They can be identified by the icons. As usual each of these has a corresponding menu option and can be found on all the tools.

Notice in figure 13 that the execution point, indicated by the yellow arrow is not at the breakpoint!

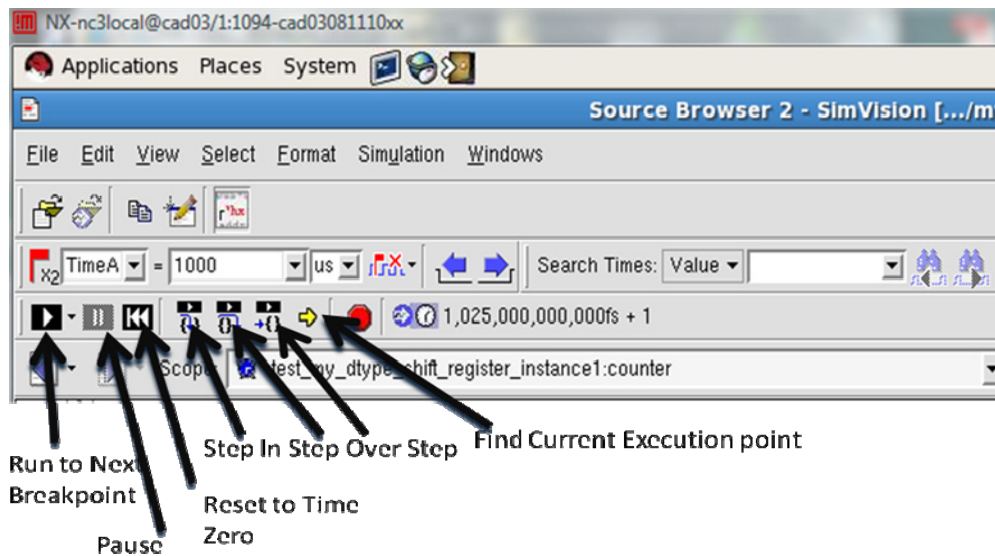


Figure 12A: Annotated Run and Step Icons

The example in figure 13 has been stepped from the breakpoint. Notice that the execution point, indicated by the yellow arrow is not at the breakpoint!

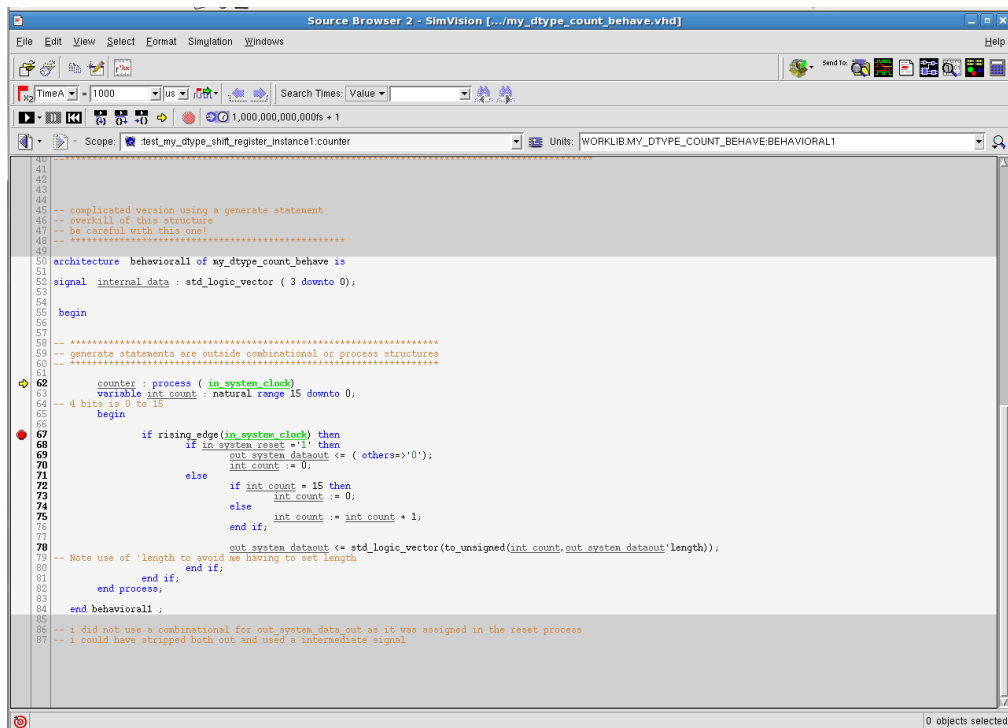


Figure 13: Breakpoint and Step Operation

#### 4.2.4. Create force / Release Force

The Create force option allows us to drive a node to a fixed value and to hold it at this until we “release”. We can drive a node to a 0/1/X or Z. The normal procedure is to select the node in the Releasing a node allows the normal drive for that node to resume.

We can do this, as we can in the design and waveform browser by selecting the relevant signal and invoking the menu command to either create or release a force.

- Select Signal
- Create/Release Force

*Hint:*

*Remember that the same signal can present at different levels in the hierarchy. So if the signal is passed down the hierarchy only the level you created the force at and below will see the changed value!*

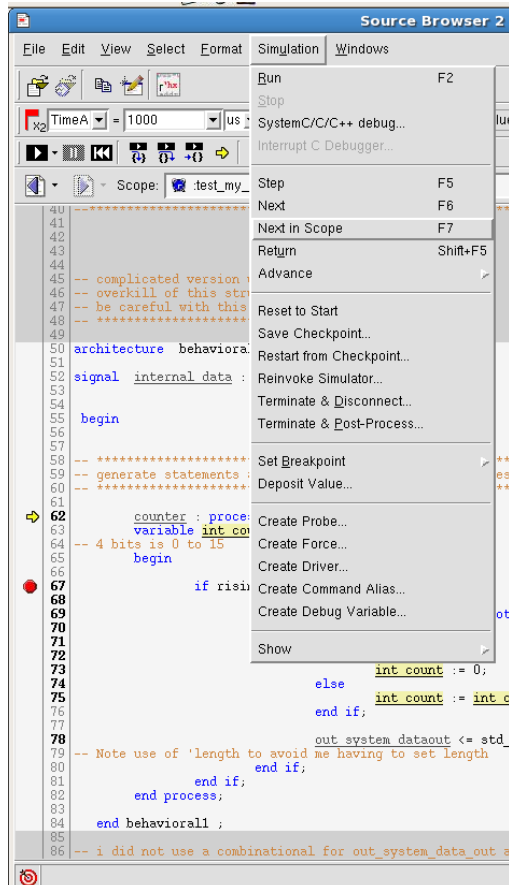


Figure 14: Simulation Debug operations

We have a large number of options at this point in addition to the usual Hint:

*Double clicking on a line number will set/disable a breakpoint*

### 4.3. Finding and highlighting the current execution point

If you select the yellow arrow on the icons Top middle in figure 16, then this will indicate the current execution point for the VHDL code. The source code browser will scroll to show the current execution point at the base of the browser.

## 5. Making use of the schematic trace tool

Again one of those tools most useful on larger designs but which it would be good to have a basic understanding off.

Quite often of the most use when working on code where you are unfamiliar with as the structure will not then be known to you in great detail. For example when integrating modules into an existing design infrastructure. It will automatically build a hierarchical schematic of the VHDL you are working. In particular it allows you to trace the path of signals across levels in the same way you would on a normal schematic. It can be used to make clear the overall structure and interconnectivity of a VHDL design.

### 5.1. How to use it:

Select a single or instance and invoke the schematic browser using either the menu of the icon. It has an icon in the same set at the design browser, which looks

You can move up and down the hierarchy

Trace logic that is driven by a node

Trace logic that is driving a node,

Trace signals up or down the design hierarchy.

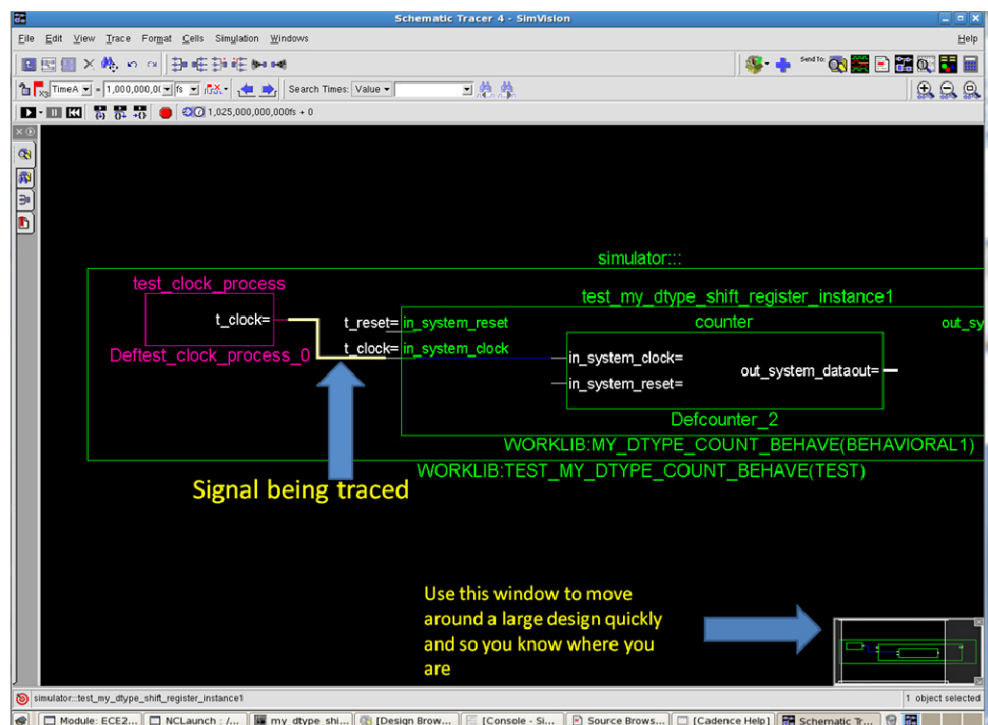


Figure 15: Schematic Tracer : Showing traced signal and large scale selection frame

Notice that the display includes the instances, the signals between them and the names, and widths, of the ports between them. This allows you to get a visual feel for the overall structure of the design.

## **6. Experimentation and using the tools.**

Experiment with these options on an existing design. Even when not required an understanding of the functions will be useful. Many of these facilities are standard techniques used in programming and HDL. You may not know the exact command but you should not that the facility will exist so you can try and determine how to use on the tool you are using i.e. modelsim. I would suggest you make notes. As usual if you find something you consider needs covering let me know. Try and find out what the other icons do and if they would be useful!