HUNT ENGINEERING
Chestnut Court, Burton Row,
Brent Knoll, Somerset, TA9 4BP, UK
Tel: (+44) (0)1278 760188,
Fax: (+44) (0)1278 760199,
Email: sales@hunteng.co.uk
www.hunteng.co.uk
www.hunt-dsp.com

Simulating your HERON FPGA Design using ModelSim

v 1.0 R.Williams 13-12-02

For modules in the HERON-FPGA and HERON-IO families, HUNT ENGINEERING provide a comprehensive VHDL support package. The VHDL package consists of a "top level", with corresponding user constraints file, VHDL sources, a Hardware Interface Layer, and User VHDL files as part of many examples.

In addition to the VHDL source and Hardware Interface Layer, HUNT ENGINEERING also provides files that can be used for simulating these example projects. This simulation support is written to use the ModelSim VHDL simulator, and can be used with any version of this simulator, whether it is ModelSim Xilinx Edition, Personal Edition or Special Edition. The Modelsim tool is not included in the standard Xilinx ISE package, but is purchased separately. Of course if you have not purchased Modelsim you will not be able to simulate your designs.

The simulation files provided with each example perform two functions. Firstly, for each example the simulation that is performed can be used to demonstrate the correct functioning of that particular example.

For example, for each FPGA module type there is an Example1 provided as part of the 'Getting Started' tutorial that reads data from an input HERON FIFO and transfers the data, unaltered, to an output HERON FIFO. When the simulation for that example is run, the simulation will model the behaviour of the example by transferring the data contained in one text file, through the example1 design, to an output text file that can be compared against the original.

The second function of the simulation files is to create a starting point for your own simulation. By using one of the example projects as a template a project can be quickly created that will enable simulation using the ModelSim VHDL simulator.

This document demonstrates how to run the supplied simulation for the example projects, and discusses the key points to consider when developing a new simulation using the examples as a template. This document is written for use with projects that are based around Version2.0 of the Hardware Interface Layer. If you wish to convert an existing HIL v1.x project to use the latest version of the Hardware Interface Layer, please read the document 'Converting HIL v1.x Projects to v2.0'

History

Rev 1.0 First written

Simulating the Supplied Example Projects

For all HERON-FPGA and HERON-IO module types HUNT ENGINEERING provide an Example1 project. This project is supplied as part of the 'Getting Started' tutorial and it is expected that users will work through this example in the process of learning how to use their new hardware. For all module types, the Example1 project includes a set of simulation files that both demonstrate the functionality of the example and provide a starting point for creating a new design, and along with it a new simulation model.

In addition to Example1 most board types are also supplied with several other example projects, each of which can be found on the CD. For example, for FPGA modules such as the HERON-FPGA3 there are Digital-I/O examples, and for IO modules such as the HERON-IO4 there are A/D and D/A examples. For each of these examples there is a simulation that can be run.

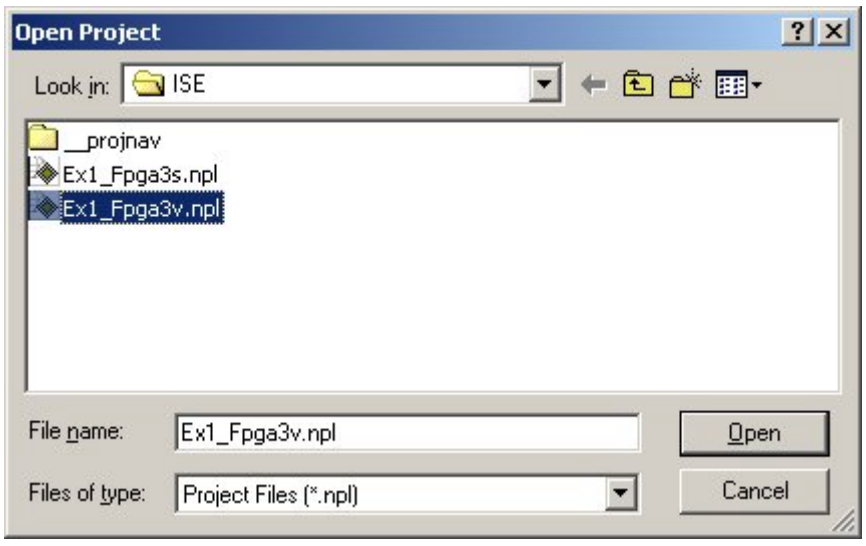
The following description covers running a functional simulation for any of the example projects supplied on the HUNT ENGINEERING CD. The following steps assume that there has been no functional change made to the example that is being simulated. If changes have been made, then it is entirely possible that the results of the simulation will be different to that which is expected, and this could be interpreted as a simulation failure.

If you wish to work on simulating a modified example, please work through this section by simulating Example1 for the board type you are using and then move on to the following section in this document.

Note, the following steps will run a functional simulation, and as such no timing issues will be taken into account. For advanced users who wish to run timing simulation this can be done by invoking the 'Simulate Post-Place & Route VHDL Model' process in place of the 'Simulate Behavioral VHDL Model' process.

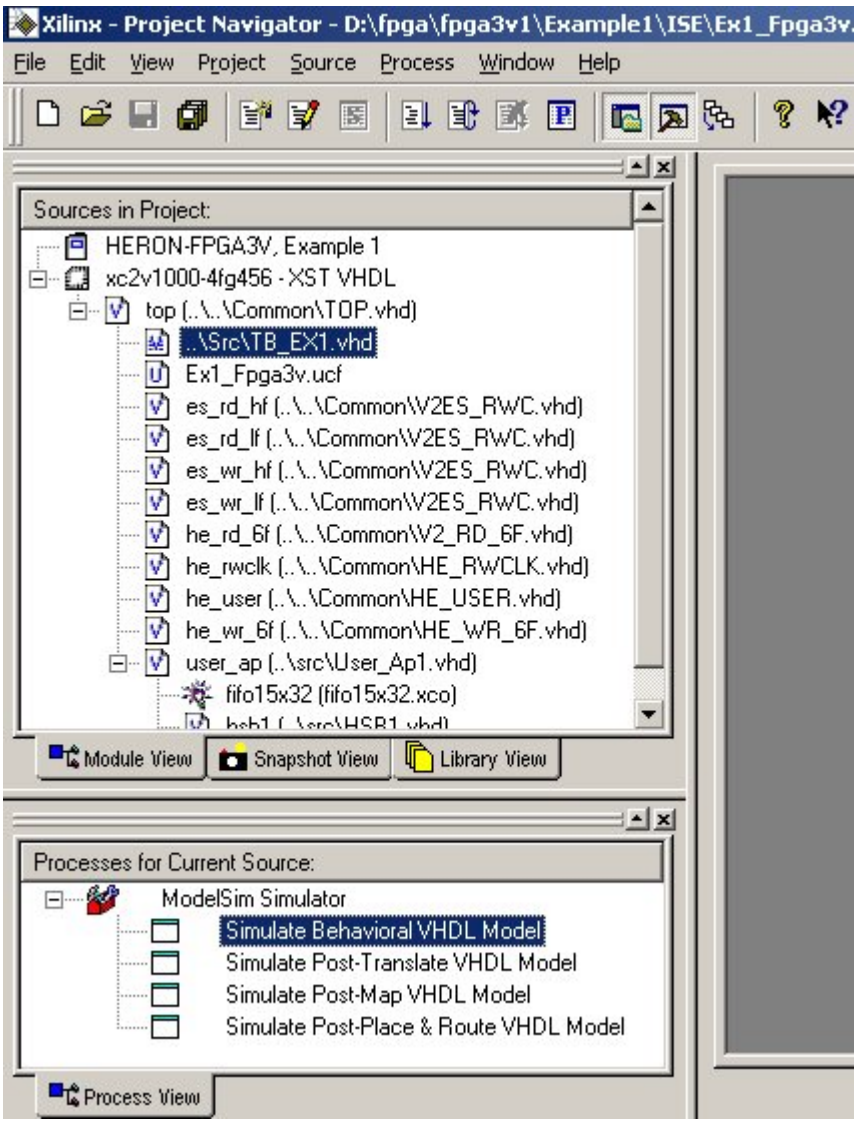
Users that want to do timing simulation of their design should not be concerned with simulating the timing of the HERON FIFO interfaces as these timings are guaranteed by the Hardware Interface layer. The timing simulation is really only offered for simulating your own VHDL code.

1. Open the ISE Project Navigator.
2. Open the example you wish to simulate. For the purpose of this document we will work through simulating Example1 for the HERON-FPGA3V. However if you are simulating a different example, simply replace the references to HERON-FPGA3V with the module type you are using and replace Example1 with the name of the example you using.



Please remember, the example you choose to simulate at this stage should not be modified from what appears on the HUNT ENGINEERING CD.

3. To start the simulation tool, ModelSim, first select the simulation Test Bench in the 'Sources in Project' view. For the standard examples on the HUNT ENGINEERING CD a test bench is automatically included in the project and should be listed just below the top design file 'top.vhd'. When the test-bench has been correctly selected it will become highlighted and the ModelSim processes will become available in the 'Process View', as shown below.



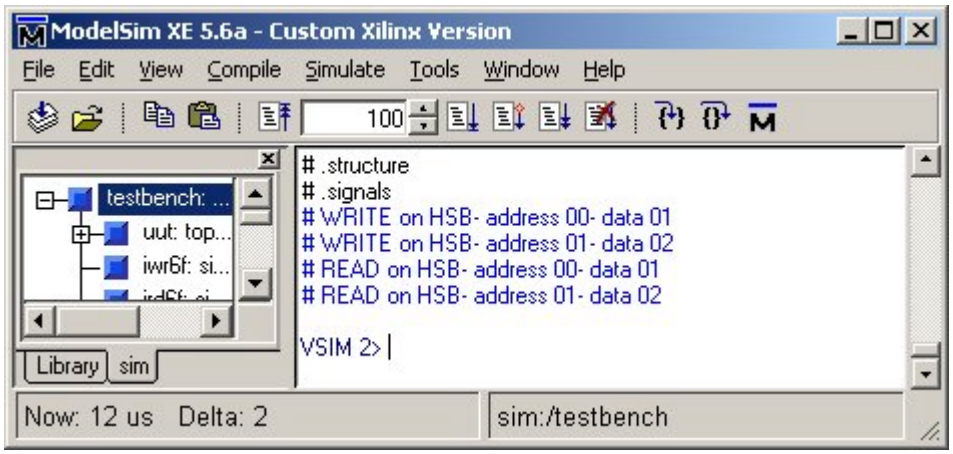
With the test-bench selected, double-click on 'Simulate Behavioral VHDL Model'. At this point ModelSim should start.

4. The first operation ModelSim performs is to compile all of the design units. If there are any syntax mistakes in the VHDL at this point, ModelSim will stop with errors displayed in the main window. Once all design units are compiled the test bench is loaded and the ModelSim waveform, structure and signals window are opened. At this point, all signals listed at the end of the simulation script file are displayed in the waveform window. The simulation script file used for Example1 is 'tb_ex1.do', and is located in the project's **sim** directory.

5. At this point, the simulation begins and simulation messages are displayed in the transcript. During the simulation any supplied text input files are read and output text files are created. In the case of Example1, the files Fifo_In0.txt, Fifo_In1.txt, Fifo_In2.txt, Fifo_In3.txt, Fifo_In4.txt and Fifo_In5.txt are all used as input data for the six Heron Input FIFO

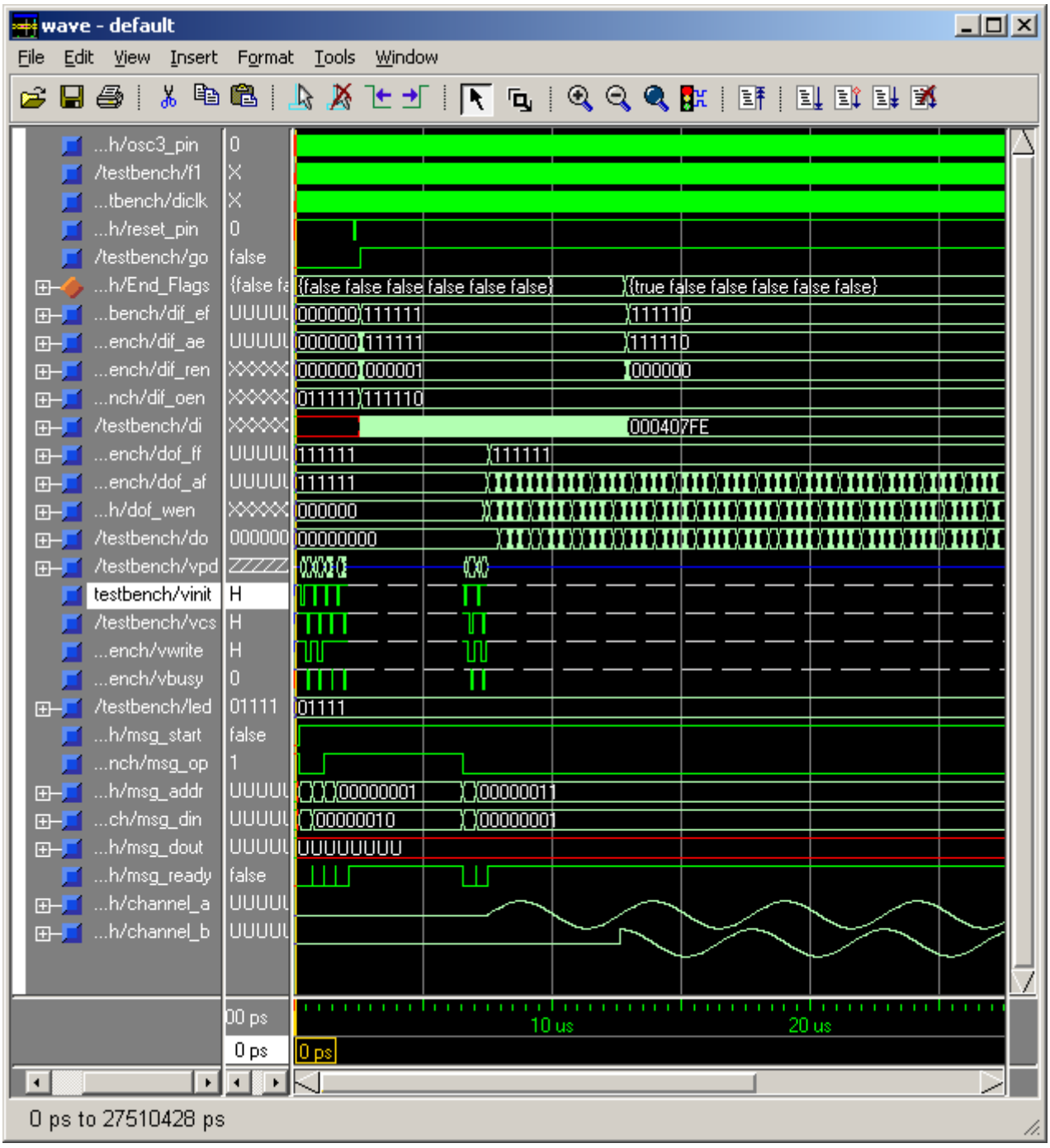
connections. Six new files are created as output (Fifo_Outn.txt). One of these files will contain the data written during simulation, depending on which FIFO was defined in the test-bench (normally FIFO 1).

6. When the simulation has completed, ModelSim should return to the command prompt in the main ModelSim window.



As we can see above, in this simulation two write accesses were made over the HSB message interface. The first access was to address 0 (to set the Input FIFO number used) and the second access was to address 1 (to set the Output FIFO number). These accesses were then followed by two read accesses to check the values that were written.

7. When simulation completes it is then necessary to check any output text files that were created, as well as looking at the transcript for possible error or warning messages and the waveform display for potential errors in the waveforms.



A typical waveform display from a simulation.

Note that you can see activity on the FIFO signals. Generally you should not see any red signals, but the FIFO data in signals are red until they are driven. Note also that the msg_dout is red because it is bi-directional and is not driven unless a cycle is in progress on the user interface.

Note also that the screenshot is taken from a simulation of a board with A/Ds and Modelsim is used to display the sinewave signals coming from the text file.

For more information about the simulation results please refer to the Modelsim documentation.

For the standard examples on the HUNT ENGINEERING CD it is normal for at least one output text file to be written during simulation. Typically this file will need to be compared to another file to check the contents against what was expected after simulation.

In the case of Example1, data is read from the selected HERON Input FIFO and passed, unaltered, to the selected HERON Output FIFO. The result of this is that the output file written during simulation should contain the same data that was supplied through one of the input files. To check this compare the input file 'Fifo_In0.txt' that was read for simulation, with the output file 'Fifo_Out1.txt' that was created during simulation. This can be done either opening a text editor to view the files and compare them by eye, or by using a utility such as the DOS command 'fc' (e.g. `fc Fifo_In0.txt Fifo_Out1.txt`).

For other examples such as the A/D and D/A examples of the HERON-IO modules, and the Digital I/O examples of the HERON-FPGA modules, there typically exists a file called 'ExpResult.txt'. This file will contain the expected simulation results for those projects, and should be used to compare against any file written during simulation.

Simulating Your Own Modified Projects

As you begin to modify the VHDL code contained in the projects from the standard examples on the HUNT ENGINEERING CD, you will soon reach a point where you wish to also modify and expand the simulation model.

There are several ways that a project may change, and depending on these changes you will need to add certain changes to the simulation test bench, simulation script, input text files and expected-results.

Adding New VHDL Source Modules

Firstly, for each new module added to the project you will need to reference the new module in the simulation script file 'tb_ex1.do' contained in the project **sim** directory.

Each VHDL source module added to the simulation script must appear in a certain compile order so that it is compiled before any files that reference it. To ensure that this is achieved it must be placed just above the existing entries for the source modules in the project **src** directory, and below the entries for source modules in the **../common** directory.

For example, if we were to add a new source module to the HERON-FPGA3 Example1 named 'new.vhd', we would add the following lines shown in bold below:

```
vcom -just e -93 -explicit   ../../common/HE_WR_6F.vhd
vcom -skip e -93 -explicit   ../../common/HE_WR_6F.vhd
vcom -just e -93 -explicit   ../src/new.vhd
vcom -skip e -93 -explicit   ../src/new.vhd
vcom -just e -93 -explicit   ../src/HSB1.vhd
vcom -skip e -93 -explicit   ../src/HSB1.vhd
```

Changing the Simulation

If you need to develop the simulation in order to test new functionality, or need to look at different signals in more detail then there are several things that you may wish to change. The simulation supplied with each project works with a simulation script that is usually found in the project **sim** directory. This simulation script performs three main tasks. Firstly it lists all of the source modules that must be compiled for simulation. Secondly it provides an initial set-up of the simulation tool, including which ModelSim windows are opened and more importantly, the set of signals are first displayed in the waveform window. Thirdly, it gives an initial run time for simulation.

If you need to change the signals that are displayed in the waveform window, you can either edit the simulation script file in the **sim** directory, or you can add or modify the originally displayed signals when the first simulation has completed, and then restart the simulation and re-run for a new time period.

If you want to change the stimuli applied to the signals in the simulation you will need to alter the test-bench. The test bench is included in the project source tree, and is typically the first module listed under the top design source ('top.vhd'). It is located in the project **src** directory. Any changes should be made to the 'STIMULI' section at the end of this file.

Modifying the VHDL Source Code

Each time you modify the design VHDL there may be a change to the results of simulation. This change to the simulation will be due to one of two factors.

Firstly, if your simulation has been correctly set up such that the input stimuli are correct and the results you expect are correct, then if the simulation results do not match what is expected there is a mistake in your design. In this situation, changes will be necessary to the design so that the simulation becomes successful.

Secondly, as your design develops, the original simulation stimuli may no longer be valid. In this case you will to modify the input stimuli to match the new design expectations.

For the standard examples on the HUNT ENGINEERING CD the simulation can be changed in several ways. These changes can be done by modifying the input text files used by the simulation, or by editing the simulation processes at the end of the test-bench source file.

When changing the test-bench to modify simulation, please remember the following points:

1. Each simulation should include a system reset at the start. To do this simply set the designs reset input (typically named '`RESET_PIN`') low for 5 cycles and then set it high again.
2. Many of the examples require control information to be sent over the HSB message interface. For example, in the Example1 simulation, the input FIFO and output FIFO numbers are first programmed over the HSB interface before any FIFO data transfer is performed. If your design requires that one or more HSB messages be sent to the module, please ensure these messages are implemented in the simulation stimuli. If you are unsure how to do this, simply refer to Example1 for an example.
3. The simulation component for the Hardware Interface Layer Six FIFO Read component (SIM_RD_6F), has an input 'GO'. This input must be set to false at the start of simulation. As soon as the input 'GO' is set true, the simulation component will start reading data from the input text file and data will be passed through the design. Please ensure the 'GO' is first set to true when the simulation is ready for FIFO data transfer to begin.
4. When simulating designs that use external clock inputs, such as with several of the A/D and D/A examples, please check that the simulation stimuli include a clock input that represents the clock you are applying to the system.