



Tutorial 2: VHDL Design and Simulation

1 Introduction

The aim of this tutorial is to provide you with experience in VHDL design and simulation by implementing various building blocks which are used for microprocessor interfacing. These include a decoder and a latch. All of these designs will first be simulated using Altium Designer before downloading to the NanoBoard for implementation. The DIP switches and TEST/RESET button will be used as inputs and the LEDs will be used as outputs. As an additional task, the bi-directional octal register described in lectures can be simulated.

Each design should be implemented as a separate FPGA project. Parts of the design from one project can be copied to the next. The detailed procedure for the decoder, which should also be followed for the latch is:

- Follow example from the previous tutorial to start a new FPGA project in a new directory and create blank schematic.
- This time a sheet symbol will be created first which will include the VHDL code for the decoder. Create a sheet symbol by using **Place** → **Sheet Symbol** on the schematic view.
- Edit the *Filename* parameter of the symbol to be the name of your choice (for example *Decoder3to8.Vhd*). The template VHDL file will be this name and the root name will be used as the entity name.
- You also need to add the parameter *VHDLENTITY* to the symbol to match the entity name in the VHDL code by selecting the symbol and using the pull-down menu to select the **Properties...** option.
- To add inputs and outputs to the sheet (ports), use **Place** → **Add Sheet Entry** and place them as required. Edit the names and types by selecting the port and using the **Properties...** menu option. The decoder requires a bus with three signals as an input and a bus with eight signals as an output.
- Place the DIPSWITCH part to be used as the input and the LED part to be used as the output. Connect these to the sheet symbol that was created above. The input will require a bus joiner to connect only three of the DIPSWITCH signals to the decoder.
- Create a template VHDL file by selecting the sheet symbol created above and using **Sheet Symbol Actions** → **Create VHDL File from Symbol**. Rename the VHDL file to an appropriate name by using the **Save as ...** menu option.

Note: Alternatively, the VHDL file can be written first and a sheet symbol created from it. Doing it this way, the *Filename* and the *VHDLENTITY* parameters are set automatically. The method you use is your choice.

- Write the VHDL code for the decoder in the template VHDL file. You will also want to include the following two lines for the libraries:

```
use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;
```

The use of the IEEE library has to be enabled for the project. Select **Project** → **Project Options...** and click on the **Synthesis** tab. Select the **Include IEEE Numeric STD Library** check box and close the window. However, this should not be required if using the Altium synthesiser.

- To simulate the design a VHDL test bench is required. Normally, an option to create a VHDL test bench from the top-level schematic should work, however it is proving problematic.

- This is the alternative method for creating a VHDL test bench. First select **Simulator** → **HDL Compile** and fix any possible compilation errors. Once this is done, find the generated VHDL file which represents the top level schematic in the **Projects** view under **Generated / VHDL Files**. Open this VHDL file and select **Design** → **Create VHDL Testbench**. This will create a template file for stimulus code to be inserted. You may want to include extra libraries as required at the top of the file.
- The top level of the simulation is the test bench. Select **Project** → **Project Options...** again and click on the **Simulation** tab. Fill in the details in the **Design** area to match the test bench for the design and close the window.
- Start the simulation by selecting **Simulator** → **Simulate**. Click **Done** on the pop up window and start the simulation using **Simulator** menu with the appropriate **Run** command.
- Follow the example from the previous tutorial for downloading the design to the NanoBoard. Remember that the project has to be configured to target the FPGA on the NanoBoard.

Be careful not to name projects, sheets and VHDL files and entities with the same name as this can cause problems with the software. As always, refer to the in-line documentation. A convenient method is to use the search function to locate specific information.

One more thing. When a symbol has a bus input or output (such as the decoder input and output) don't use individual wires but rather a bus. The notation for a bus on a schematic is *name[from..to]* where *name* is the name of the bus, *from* is the number of the first signal and *last* is the name of the last signal (for example, the decoder input could be labelled *a[2..0]* which refers to three signals).

2 Decoder

- a) Create a new FPGA project and design a three-to-eight decoder, which each of the outputs high for a particular combination of the input. Connect three of the DIP switches to the input and connect the outputs to the LEDs.
- b) Simulate the design by creating a VHDL test bench and using the Altium Designer Simulator. Test all input combinations and verify correct operation. If the simulation does not provide the expected results, modify the design and repeat the simulation.
- c) Download the design to the NanoBoard and verify that the design operates as predicted in the simulation.
- d) If the decoder is used to decode the top three bits of an eight-bit address space, what are the addresses corresponding to each of the decoder outputs?

3 Latch

- a) Create a new FPGA project and design a positive edge-triggered three-input latch. Connect three of the DIP switches to the input and connect the outputs to three of the LEDs. Use the TEST/REST button as the clock input. Do not worry about debouncing the clock input.
- b) Simulate the design by creating a VHDL test bench and using the Altium Designer Simulator. Verify correct operation and fix if necessary.
- c) Download the design to the NanoBoard and verify that the design operates as predicted in the simulation.
- d) Create a new FPGA project to include the latch connected to the decoder. Connect three of the DIP switches to the input of the latch and connect the outputs of the decoder to the LEDs. Simulate to verify correct operation and then download the design to the NanoBoard.

4 Bi-Directional Octal Register

This task is optional. Create a new FPGA project and implement the bi-directional octal register as described in lectures. Simulate the design for all possible combinations of direction and output-enable signals. Take care when changing the direction signal as the new output has to be driven to high impedance.

February 18, 2009