Class project
Architecture of Digital Systems 1

December 4, 2014

Part I
Introduction

Imagine you are a new employee of the company IP-Design GmbH. This company develops IP-Cores and you are working in a branch in Kaiserslautern. As every new employee you have to learn the ropes. To ease this process you will be integrated directly in the development process of a project, working on less critical tasks.

1 Situation

One active project of the company is the development of a frequency selective digital filter. A C-Program of the filter has been already developed. It was tested and verified thoroughly and has been found error free. Thereafter a first version has been described in VHDL. However the outputs of the hardware module are different from the outputs of the C-Program.

To find the Bug, the components of the filter have to be verified individually. The filter and all its components are working on floating point numbers.

2 Task

Your task is to write a test bench for the components in VHDL. The test bench should be able to convert decimal numbers to floating point numbers and vice versa.

The test bench should be able to convert numbers from a decimal representation to floating point representation and to send them to a filter component according to the communication protocol described in the specification section. The component will then perfom some computation and afterwards output the results. The test bench should now be able to read these values and convert them back to a decimal representation.

To reduce simulation time and the number of signals in the design, a dummy module will be given to you. It has a simplified interface and I/O behaviour and can therefore be used for every component. Internally the dummy module stores the values which have been send in an internal memory and then writes the received values in the same order to the output, skipping any computation. You can use this module to verify that your test bench is working correctly.

3 Specifications

The filter modules are a clocked process and sensitive on either falling or rising clock edge. They use a single precision floating point number system which is IEEE 754 compliant.

3.1 Communication Protocol

The dummy module uses a simple serial protocol to communicate with the environment. To send an input sequence, the *enable* signal has to be set high. At the time the *enable* signal is high, the module expects a value on the *operand* signal at every rising or falling clock edge, depending on the value of the *edge* signal. If the *edge* signal is high, the module will be sensitive on rising edges and if it is low, the module will be sensitive on falling edges. The *edge* signal should be set on initialisation and shall never be changed afterwards. The number of rising or falling clock edges while the enable signal is high determines the number of values detected by the dummy module.

When the module starts to output the result, it sets the *valid* signal high and writes the first value to the *result* signal. For every rising or falling edge of the clock a new value will be written to the output and the *valid* signal will be kept high. After the last value has been written, the *valid* signal will be finally set to low.

3.2 Interface

The VHDL code in the code fragment 1 shows the interface of the dummy module. Every input signal (clk, enable, edge and operand) and output signal (result and valid) of the module as well as their bit widths are specified there.

```vhdl
ENTITY Dummy_Module IS
  PORT(
     CLK      : IN  std_logic;
     enable   : IN  std_logic;
     edge     : IN  std_logic;
     operand  : IN  std_logic_vector(31 downto 0);

     result   : OUT std_logic_vector(31 downto 0);
     valid    : OUT std_logic
  );
END Dummy_Module;
```

Code Fragment 1: Interface Dummy Module

4 Work Stations

In building 12 room 524 are work stations available, which can be used to solve this and the following tasks of the class project. Every group gets an windows account where the required software is already installed. You get the login data in the introductory lecture of the class project.

You can use the work stations on the following Days:

- Monday: 8 - 14 o'clock

- Wednesday: 8 - 18 o'clock
- Thursday: 8 - 15 o'clock
- Friday: 8 - 18 o'clock

4.1 Using private Computers

You can also use your own computer to solve the tasks. All you need is ModelSim and a text editor of your choice.

You can get a free student version of ModelSim from the website of Mentor Graphics[1]. After the installation process you have to fill out a request form to get the required licence file. This file will be send to you by email. Therefore declaring a correct email address in the request form is advisable.

4.2 Documents

To help you solving the tasks some documents are stored on the work stations. These documents are tutorials for ModelSim, test benches and the IEEE standards for floating point numbers (one from 1985 and one from 2008).

5 Additional Advices

Read the task descriptions carefully and write down any information which might be important to solve the task. As this is not a classical lab, some requirements for the solution can be hidden. Working in a structured way helps to find possible requirements! If you don't know any structured apporach: figure one out and try to develop one.

Get information about the IEEE 754 standard, using the internet or the documents given to you. Figure out the different cases you have to consider when decimal numbers have to be transformed to floating point numbers (and vice versa).

Use the Tutorials to write a running skeleton test bench, which does not have any functionality but gets you used to ModelSim and VHDL.

Before implementing functionality to the test bench, write down what and how you want to implement it.

[1] `http://www.mentor.com/company/higher_ed/modelsim-student-edition`