Royal Institute of Technology (KTH)
School of Electrical Engineering and
Computer Science (EECS)
Department of Electronics (ELE)

Version 1.1
Last revision 2019-11-04

SystemVerilog - Laboration 1

IL2203 Digital Design and Validation using HDLs

Testing an unknown function

Student Name:…………………………………………………………..

Personal Number:…………………………………………………….

Date of Approval:…………………………………………………….

Assistant:……………………………………………………………….

SystemVerilog - Laboration 1

Purpose

The purpose of this homework is to practice using user defined types to test a configuration register design. A configuration register is a bank of registers that is used to store setup information for the system (i.e. the configuration). They can be quite complex, with read only bits, write 1 to clear bits, and varying bit-widths. The configuration register design you are to test is very simple. All registers can be read and written and are 16-bits wide. The design contains 8 registers as defined in Table 1 along with their reset values. All bits are writable. The reset is active high.

Register	Width	Address	Reset Value
adc0_reg	[15:0]	0	16'hFFFF
adc1_reg	[15:0]	1	16'h0
temp_sensor0_reg	[15:0]	2	16'h0
temp_sensor1_reg	[15:0]	3	16'h0
analog_test	[15:0]	4	16'hABCD
digital_test	[15:0]	5	16'h0
amp_gain	[15:0]	6	16'h0
digital_config	[15:0]	7	16'h1

Table 1: Config Register Defines

Tasks

Write a testplan describing how you are going to test the design. Then write a self-checking testbench to perform the testing you described in your testplan. Use a user defined enumerated type to enumerate the registers. There is 1 bug per register. An encrypted version of the buggy configuration register code is provided. Compile this module just like an unencrypted module is compiled.

Identify the 8 bugs in config_reg_buggy. For each bug, report:

a) Design Input for Bug to Appear:

E.g. Write FFFF to register x

b) Expected Behavior:

E.g. Bit 15 of register x is writable to a 1

c) Observed Behavior

E.g. Bit 15 of register x cannot be written to a 1

Remember, your bug report should contain enough detail for the designer of the configuration register to debug the problem.

Deliverables:

1. Test plan
2. Testbench code
3. Complete bug report for each register

Appendix A – Setting Up QuestaSim

You can (1) work from a computer in room 209 or (2) work from your own laptop by connecting remotely to a KTH server.

In case you choose option (2), open a Unix terminal and type:

ssh –Y YourKTHUsername@malavita.sys.ict.kth.se

If you have a Windows computer and want to work from home, it is recommended that you install Linux near Windows. The Ubuntu linux distribution can very easily be installed along with Windows without requiring you to partition your filesystem, as long as you have some free Gigabytes. You can then delete Ubuntu at any time through the Windows control panel. In alternative you can use Xming and Putty to connect remotely to the KTH server, but be aware that this solution is sometimes buggy.

From this time on, everything you type in the terminal happens remotely on the server. The remainder of the instructions are identical for both scenarios (1) and (2), i.e. both if you work in the lab and if you work at home. If you work in the lab, open a terminal using the graphical interface and type all commands there.

First we have to make questasim work. Give the command: *vsim*

• if questasim opens, you are fine.
• if nothing happens or modelsim opens then follow the next instructions:

gedit ~/.bashrc

An editor will pop out, search for all lines that contain the string "modelsim" and comment them out with a hash or delete them. Save the .bashrc file, then close the editor, log out, log in, try again giving the command *vsim*. This time you should get the message: "command not found".

• if you get the message "command not found" or similar, then type on the console the two following commands:

```
export LM_LICENSE_FILE=1727@lic02.ug.kth.se
export PATH=$PATH:/afs/kth.se/pkg/mentor/questa/20170221/questa_core_prime_10.5c_4/questasim/bin/
```

Now try giving the command *vsim*, questasim 10.5 should open. Note: every time you open a new terminal, you should retype the two *export* commands.

QuestaSim is an advanced version of ModelSim that allows mixed language simulations.