Norwegian University of Science
and Technology
Department of Electronics and
Telecommunication

TFE4171 Design of Digital Systems 2 Spring 2019

Exercise set 1

Delivery time: Friday 15th February, 23:59

About the exercises:

There will be 5 exercises in the course, and each will count for 4 points, all in all 20 points. These points count for 20% of the final grade.

Exercises should be executed in groups of two students, as registered in the beginning of the course. Both students will get the same points. The points will be awarded based on either: (1) show your work and conclusions to one of the TA's in the lab and answer any questions; or (2) a short report delivered for each exercise containing answers to questions, explanations of the solutions, and additional analysis if needed.

All the exercises will be run on the `venus.iet.ntnu.no` server (later in the semester you might also use the `jupiter.iet.ntnu.no` server), and it can be accessed from the computers on the DAKLAB (B-322) or you can ssh into it if you are on the ntnu network. You will be assigned user accounts for your group when you register it.

I will registerer user accounts for the groups that are ready, and send you passwords through email. If you are going to change the password, then you need to log in to the server `aurora.iet.ntnu.no` and change it with the `passwd` command. If you change it on the other servers, it will bounce back to the old password after some time, so don't do that!

The student computers in the DAKLAB B322 have recently been replaced by Windows machines. That means that for communication with the servers (linux machines) you can use `putty` for simple text interface (which will be enough in this exercise at least). It is also possible to connect with XWin32 or the CygWin tools for a graphical interface with the linux servers. However, I recommend you wait with that until it is strictly needed! Good luck!

Learning outcome

Testing out binding of assertions to modules, simple assertions with and without implication, how overlap in SVA operators work. Then you will test your basic knowledge through instrumenting two simple examples with a set of assertions for given conditions, one FIFO and one counter and comparing with expected results. You will also be using concurrent assertions to check requirements for protocol verification and also functional verification of state machine based controllers.

Reference literature

Cerny et. al., "SVA: The Power of assertions in SystemVerilog", chapters 4, 5, 6.4.3 and 10.6.

1 Lab 1

At first, to setup your user account for using Mentor Questa tools, you need to copy a .bashrc file which defines necessary environment variables:

```
cp -r /home/courses/desdigsys2/2019s/dd2master/.bashrc .
```

Execute the commands in this file by:

```
source .bashrc
```

You only need to do this once, next time you log in this file will be automatically executed and the environment setup as required.

Now create a directory for this exercise in you home directory. After logging in, execute the

```
mkdir ex-1
```

command and then enter this directory by

```
cd ex-1
```

Now you can copy the code for Lab-1 by

```
cp -r /home/courses/desdigsys2/2019s/dd2master/ex-1/lab1 .
```

and enter the directory for the Lab 1 examples:

```
cd lab1
```

For the rest of Lab-1 you will follow the description in the `SVALAB_LINUX_LAB1.pdf` file in the lab1 directory. The directory names may differ slightly. Also, the instructions favour the `vi` editor, you will of course use your own favourite such as `vi, vim, gedit, nano, emacs` or whatever you prefer.

a) Perform task 2 in the instructions. Report how you solved it.

b) Perform task 3 (no_implication) in the instructions. Answer the two questions for the no_implication case:
Q: WHY IS THERE A FAIL -AND- A PASS AT TIME (70) ??
Q: WHY ARE THERE 2 FAILs AT TIME (130) ??

c) Perform task 4 (implication) in the instructions. Answer the two questions for the implication case:
Q: WHY ARE THERE 2 PASSES AT TIME 70 ?
Q: WHY IS THERE A PASS -and- A FAIL AT TIME 130 ?

d) Perform task 5 (implication_novac) in the instructions. Do you get the expected result?

2 Lab 2

Go back to your `ex-1` directory in your home directory. Copy the code and instructions for lab-2 by:

```
cp -r /home/courses/desdigsys2/2019s/dd2master/ex-1/lab2 .
```

and enter the lab2 directory. Follow the instructions in the `SVALAB_LINUX_LAB2.pdf` file in the lab2 directory.

a) Perform task 2 (overlap) in the instructions. Answer the following questions for the overlap case:
Q: WHY DOES THE PROPERTY FAIL at 30?
Q: WHY DOES THE PROPERTY PASS at 90?
Q: WHY DOES THE PROPERTY PASS at 150?
Q: WHY DOES THE PROPERTY FAIL at 170?
Q: WHY DOES THE PROPERTY FAIL at 190?

b) Perform task 3 (non_overlap) in the instructions. Answer the following questions for the non_overlap case:
Q: WHY DOES THE PROPERTY FAIL at 70?
Q: WHY DOES THE PROPERTY PASS at 90?
Q: WHY DOES THE PROPERTY FAIL at 170?
Q: WHY DOES THE PROPERTY FAIL at 190?
Q: WHY DOES THE PROPERTY PASS at 210?

3 Lab 3

Go back to your `ex-1` directory in your home directory. Copy the code and instructions for lab-3 by:

```
cp -r /home/courses/desdigsys2/2019s/dd2master/ex-1/lab3 .
```

and enter the lab3 directory. Carefully read the description of the FIFO and follow the instructions in the `SVALAB_LINUX_LAB3.pdf` file in the lab3 directory.

a) Perform task 2 by running the nobugs case. Study carefully the log to understand the function of the FIFO. Remember that no assertions are active yet.

b) Perform tasks 3 through 9 by adding one by one the given properties (that are already asserted) by removing the DUMMY code and adding your property code as specified. Run the check for each property and compare the log with the solution log. Report the result, change your property code if necessary, and report your solution.

4 Lab 4

Go back to your `ex-1` directory in your home directory. Copy the code and instructions for lab-4 by:

```
cp -r /home/courses/desdigsys2/2019s/dd2master/ex-1/lab4 .
```

and enter the lab4 directory. Carefully read the description of the Counter and follow the instructions in the `SVALAB_LINUX_LAB4.pdf` file in the lab4 directory.

a) Perform task 2 by running the nobugs case. Study carefully the log to understand the function of the Counter. Remember that no assertions are active yet.

b) Perform tasks 3 through 5 by adding one by one the given properties (that are already asserted) by removing the DUMMY code and adding your property code as specified. Run the check for each property and compare the log with the solution log. Report the result, change your property code if necessary, and report how you solved it.

5 Lab 5

Go back to your `ex-1` directory in your home directory. Copy the code and instructions for lab-5 by:

```
cp -r /home/courses/desdigsys2/2019s/dd2master/ex-1/lab5 .
```

and enter the directory for the Lab 5 examples:

```
cd lab5
```

For Lab-5 you will follow the description in the `SVALAB_LINUX_LAB5.pdf` file in the lab5 directory. The directory names may differ slightly. Also, the instructions favour the `vi` editor, you will of course use your own favourite such as `vi, vim, gedit, nano, emacs` or whatever you prefer.

a) Perform task 2 in the instructions, this should familiarize you with the design.

b) Perform task 3 (check1) and create the check1 assertion according to the instructions. Compare your results with the check1 log in the .solution directory. Report how it works, and fix the assertion if it does not work. Explain how and why.

Then repeat the same for task 4 and 5 on the check2 and check3 cases, respectively. Report in the same way how results compare to the .solution log, fix it if necessary, and report the how and why.

c) Perform task 6 (checkall) in the instructions. Report how it works wrt. several bugs exposed by several assertions.

6 Lab 6

Go back to your `ex-1` directory in your home directory. Copy the code and instructions for lab-6 by:

```
cp -r /home/courses/desdigsys2/2019s/dd2master/ex-1/lab6 .
```

and enter the directory for the Lab 6 examples:

```
cd lab6
```

For Lab-6 you will follow the description in the `SVALAB_LINUX_LAB6.pdf` file in the lab5 directory. The directory names may differ slightly. Also, the instructions favour the `vi` editor, you will of course use your own favourite such as `vi, vim, gedit, nano, emacs` or whatever you prefer.

a) Perform task 2 (check1) and create the check1 assertion according to the instructions. Compare your results with the check1 log in the .solution directory. Report how it works, and fix the assertion if it does not work. Explain how and why.

Then repeat the same for task 3 to 6 on the check2 to check5 cases, respectively. Report in the same way how results compare to the .solution log, fix it if necessary, and report the how and why.