Lab Exercise 1
Design Flow and VHDL

INF3430/4431 Autumn 2017

Version 1.4/29.08.2017

Note! Before you start, read this: "Mandatory assignments and other hand-ins at the Department of Informatics" at http://www.mn.uio.no/ifi/english/studies/admin/mandatory-assignments/index.html.

The goal of our first lab exercise is to get some practice with VHDL and become familiar with the design flow and development tools Questa and Vivado.

Note! Use your private area on the network (M disk) to save source and result files. If the network response is poor, you can create a temporary area on a local hard drive. Remember to delete this area afterwards.

All the submitted VHDL files must follow the indenting guidelines as described in the cookbook.

Table 1. Attached files. The file `comp_first.do` must be modified to use your own path.

Filename	Description
`first.vhd`	VHDL source file exercise 1
`tb_first.vhd`	VHDL test bench exercise 1
`comp_first.do`	Modelsim command file for compilation
`sim_first.do`	Modelsim command file for simulation
`delay.vhd`	VHDL source file exercise 3
`tb_delay.vhd`	VHDL test bench exercise 3
`variables_vs_signals.vhd`	VHDL source file exercise 3
`tb_variables_vs_signals.vhd`	VHDL test bench exercise 3
`dff.vhd`	VHDL source file exercise 4

Exercise 1

a)

Follow the description in the cookbook *Design Flow. The Development Tools Questa and Vivado*. All of the necessary files are obtained from the website for INF3430. Simulate the chip at the RTL level. Afterwards, the design will be downloaded to the test board.

Before you program the chip, take a look at the *IO report* to check that the pin numbers are placed correctly. This means checking the pin assignment stated in the *ZedBoard Hardware User's Guide*:
http://zedboard.org/sites/default/files/ZedBoard_HW_UG_v1_1.pdf

Check the *Timing summary report* to see whether the timing constraints have been satisfied.

Approval:

No special requirements.

b)

Change the VHDL code so that the counter becomes an up/down counter by adding an `UP` signal. The `UP` signal should be assigned to SW6 (pin H17). If `UP='1'` the counter should count up, and if `UP='0'` the counter should count down. In addition, we will add a `MIN_COUNT` signal, which will give a pulse with a duration of 1 clock period when the counter goes to 0 on the way down. `MIN_COUNT` shall be assigned to LD6. `MAX_COUNT` will be modified so that it will give a pulse at the maximum value on the way up. Add the necessary modification to the XDC file to Vivado so that the `UP` and `MIN_COUNT` signals are assigned to the correct pin numbers. Modify the test bench so that we can verify that the counter is functioning correctly. A counting sequence from zero to 15, and then reversal of the direction and counting down to zero, for example, is fine. Perform the simulation, verify the pin placement and program the chip when everything appears to be correct.

Approval:

Modified VHDL file, test bench, modified XDC file and IO report.

Exercise 2

In this part of the exercise, you will design a 2-to-4 bit decoder. In the VHDL textbook, concurrent statements are used in the example in Chapter 4, but you will be using a case statement in a process instead. The outputs on the decoder should be active low, see truth table below. The decoder must be implemented on the test board. Use SW1 and SW2 as input, and LD1, LD2, LD3 and LD4 as output.

SW2	SW1	LD4	LD3	LD2	LD1
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	1

Approval:

VHDL design file, test bench, do file for compilation and display of waveform, as well as XDC file.

The exercise must be demonstrated to the laboratory supervisor for approval.

Exercise 3

In this part of the exercise, we will look at the difference between signals and variables in VHDL.

a)

Simulate the attached code in `delay.vhd` and `tb_delay.vhd`. When does the output data signal change, and what is the cause of this delay?

b)

Modify `delay.vhd` so that all of the variables are replaced by signals (TIP: signals cannot be declared within a process). In addition, `tb_delay.vhd` is to be modified so that the chip is only in reset from the time 100 ns to 200 ns (TIP: change, for example, to the value `'1'` from time zero) and the input data should now change from `"00000000"` at 0 ns (i.e. start) to `"11110000"` at time 300 ns and to `"00001111"` at time 400 ns. When does the output data signal change now? Why is the output data equal to `"UUUUUUUU"` at time 50 ns?

c)

In this part of the exercise, the attached code in `variables_vs_signals.vhd` and `tb_variables_vs_signals.vhd` is to be simulated. Why is `output(7 downto 6)` always equal to `output(3 downto 2)`? Why is `output(5 downto 4)` different from `output(1 downto 0)`?

d)

Now the signals `sig1` and `sig2` are to be removed from the sensitivity list in `variables_vs_signals.vhd`. Why does `output(7 downto 6)` and `output(3 downto 2)` have different values than in exercise c?

Exercise 4

In this part of the exercise we will look at the instantiation of components. The attached code in `dff.vhd` is a register with an asynchronous reset, where the register is given the value `'0'` when the signal `rst_n` is active low (i.e. has the value `'0'`).

a)

Create an 8-bit shift register by instantiating the component `dff` 8 times, and call this component `shift8`. Use named association in `port map`. Create a test bench that simulates the shift register.

b)

Create a 32-bit shift register by means of the component `dff` and the VHDL generate statement. Name this component `shift32`. Use named association in `port map`. Create a test bench that simulates the shift register.

c)

Create an n-bit shift register by means of a generic statement. Name this component `shiftn`. Create a test bench that simulates an instantiation of `shiftn` with the length set at 64 bit.

Approval:

VHDL source file and test bench for the individual questions.