D0011E Lab 1 (Design and Simulation of BCD checker)

The goal of this lab is to get acquainted with basic VHDL syntax and to start using Xilinx Design tools, which will be used in the course for generation of RTL (register-transfer level) schematic and simulation of VHDL code. You should be able to complete the lab during the first lab session.

All the VHDL code together with a README file will be uploaded to GitLab. You can commit the whole Vivado project to facilitate reviewers in their task. The README file will include the table of the preparation and the answers to the questions in this document, include the text and the index of the question for it to be traced.

Preparation (to be completed BEFORE the lab session):

Start by reviewing introduction to VHDL (slides from a lecture available in Canvas).

Download *lab1.zip* from Canvas under Labs >Resources. Study the source files *bcdcheck.vhd* and *bcdcheck2.vhd* with two implementations of 4-bit BCD checker (BCD stands for binary-coded decimal, see http://en.wikipedia.org/wiki/Binary-coded_decimal). Note that the inputs *x0*, *x1*, *x2*, *x3* represent bits 0 to 3 of a 4-bit encoding. Based on your understanding of the code, complete the following table and add it to the README file that will be uploaded together with the code written during this lab:

Input	max	min	even	lo3	noBCD
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
10 (A)					
11 (B)					
12 (C)					
13 (D)					
14 (E)					
15 (F)					

In the README file include the Boolean algebra equations for each variable of the truth table, with the same name as the columns and as a function of the inputs *x0*, *x1*, *x2*, *x3*.

Also study the source files *bcdcheck_tb.vhd* and *bcdcheck2_tb.vhd*. These will be used to verify the implementations of BCD checker in simulation.

Part 1. Starting up Xilinx

Start “Project Navigator” from the map “Xilinx Design Tools > Vivado 2016.4 > Vivado 2016.4” in the Start menu. Create a new project and choose an appropriate name and location in your home directory on the server. Leave the default settings under “Project Type” and go for next.

In ”Default Part” window, select the FPGA chip we are going to configure for our implementations. This chip corresponds to the one which is used in the development board we are deploying for this course. This can simply done by typing **xc7a35tcsg324-1** in the search filed. Once selected just finish for project creation.

Use the menu **Project Manager> Add Sources> Add or Create Design Sources>Add Files**... to add the 2 VHDL design files from *lab1.zip* to your project.

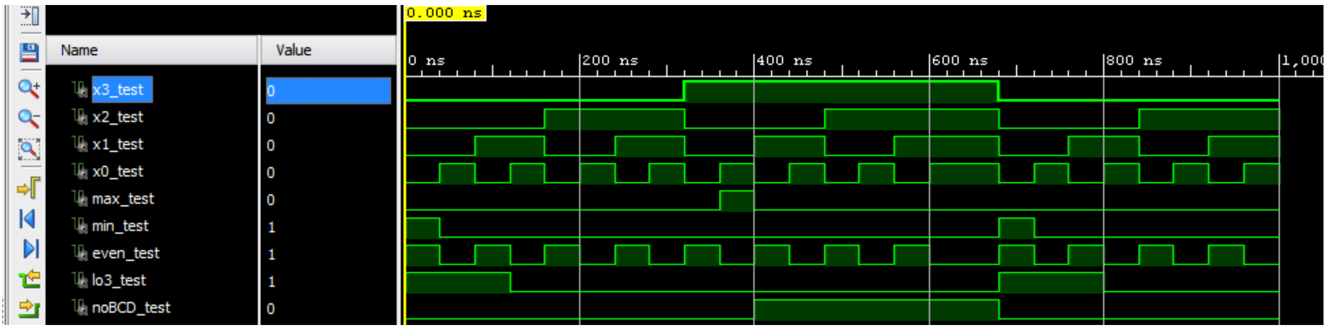
Use the menu **Project Manager> Add Sources> Add or Create Simulation Sources>Add Files** to add the 2 VHDL simulation files from *lab1.zip* to your project. (Simulation files are the test bench files which are identified by **_tb**, in their names).

The program will automatically associate test bench files (*bcdcheck_tb.vhd* and *bcdcheck2_tb.vhd*) with Simulation and the other two files (*bcdcheck.vhd* and *bcdcheck2.vhd*) with both Simulation and Design ("All").

To proceed with simulation and implementation of *BCDcheck desing, make sure that BCDcheck is set as top module name in the project settings.*

Part 2. Running a simulation

To verify *BCDcheck*, we will run a simulation of *bcdcheck_tb*. In the Simulation panel, go to simulation settings and make sure "*bcdcheck_tb" as simulation top module name*. Default simulation time will suffice to verify this design. Run the simulation by clicking on "Run Simulation" and "Simulate Behavioral Simulation". The result of simulation will open in a new window (ISim). Use the menu View > Zoom Fit to see the full simulation on the screen:



Name	Value
x3_test	0
x2_test	0
x1_test	0
x0_test	0
max_test	0
min_test	1
even_test	1
lo3_test	1
noBCD_test	0

We will use the "Objects" panel (showing signals for the selected instance or process), and the wave window. In the wave window, the changes in each signal over time are shown; in this lab, all signals have values 0 or 1. Note that the "Value" column on the left displays the values of the signals at the point in time marked by a yellow cursor (vertical line).

The default behavior is to run the simulation for 1 us (1000 ns), this can be changed before starting the simulation under "Simulation Settings". For this lab, we only need to run the simulation for 640 ns.

You can also set a breakpoint in the code. For example, double-click *bcdcheck_tb* in the Instances and Processes panel and toggle a breakpoint at the last "wait for PERIOD" instruction by highlighting the bullet of the corresponding line. Restart the simulation (Simulation > Restart followed by Simulation > Run All). Note that "Run All" will run a simulation until paused if no breakpoint is set.

By default, only signals declared in the top-level entity (in our case, *bcdcheck_tb*) are shown in the wave window. However, you can add internal signals of components (e.g. *bcdcheck_instance*) by selecting a component in the Instances and Processes panel and dragging signals from the Objects panel to the wave window.

Part 3. Verifying the BCD checker

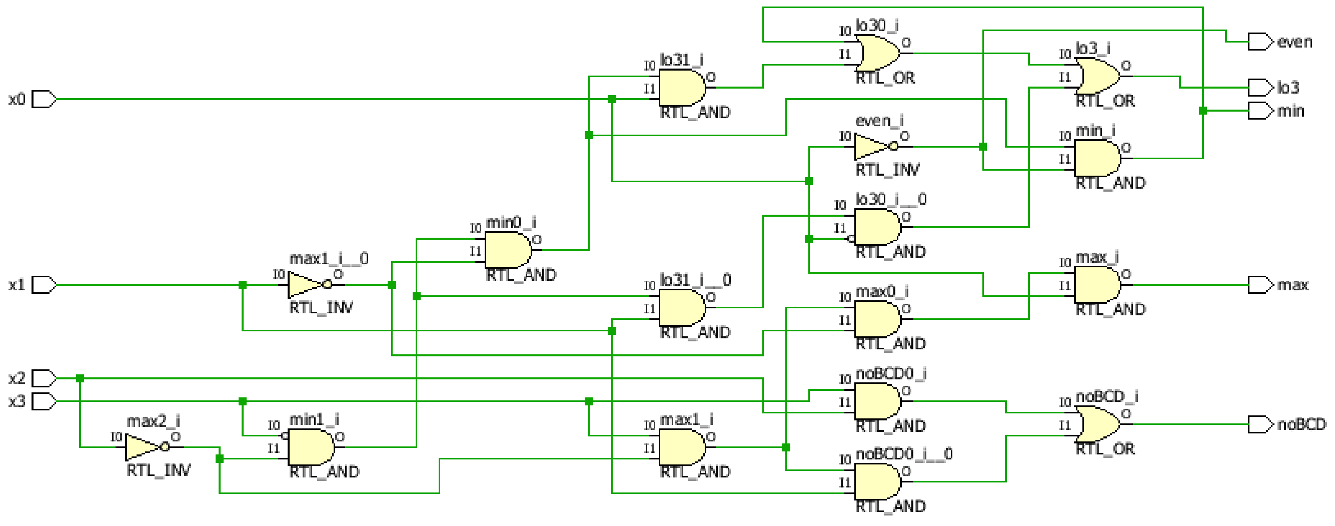
Use the table you prepared earlier and verify that for each tested input, all the output signals of *bcdcheck* are correct. *Bcdcheck* is verified by simulating *bcdcheck_tb.*

- Check that the simulation produces a correct result, by placing the yellow cursor at an appropriate point in the wave window and taking a screen shot. Make sure that the simulation is executed for 640 ns and that the whole simulation is visible on the screen.

- Explain, based on the VHDL, why these values of the output signals are correct (it is not sufficient to refer to the truth table, we are checking that you understand the VHDL code here).

Part 4. Generating RTL schematic

Leave the default elaboration settings in the RTL analysis and generate the schematic by clicking on “Open Elaborated Design”. Once finished, the schematic of *bcdcheck* design is displayed.



Part 5 Checking second design of BCD checker

Repeat Part 2 to part 4 to verify and implement BCDcheck2. This time, BCDcheck2 is set as top module name in the project settings.

Part 6 Upload the Vivado project and the README file created during the lab, remember:

- **Present the truth table you compiled at the beginning of the lab and the Boolean equations**
- **The questions in part 3**
- **An elaborated explanation of the main differences between the two designs.**