⚠ You have previously submitted this assignment with David Rama Jimeno. Group can only change between different assignments.

This course has already ended.

« 4. Exercise 2 (/comp.ce.240/spring-2024/... 5. Exercise 3 » (/comp.ce.240/spring-2024/...

COMP.CE.240 (/comp.ce.240/spring-2024/) / 4. Exercise 2 (/comp.ce.240/spring-2024/E02/) / 4.1 Exercise 02: Ripple carry adder (Gate level description)

Assignment description (/comp.ce.240/spring-2024/E02/ripple_carry_adder/)

My submissions (8/1000)

Exercise 02: Ripple carry adder (Gate level description)

General guide about directory structure and simulation process

Read the general instructions (https://plus.tuni.fi/comp.ce.240/spring-2024/material/misc/) page, at least the directory structure section. It's a good idea to get the structure down now so that you don't need to run vlib/vmap commands anymore after this exercise.

After you've got the structure down, you can follow these instructions:

First of all, write your own design under vhd/ directory and copy the given testbench under tb/ directory. When simulating follow this:

1. Start Modelsim
2. cd <your project directory (root of it)>
3. vlib <your_library_name> (e.g. vlib work, vlib my_lib, vlib synth_lib etc. whatever)
4. cd sim/
5. vmap work ../<your_directory_name> (e.g. vmap work ../work, vmap work ../my_lib, vmap work ../synth_lib etc.)
6. (if you haven't written your own code until this point, do it now)
7. vcom -check_synthesis ../vhd/ripple_carry_adder.vhd
8. vcom ../tb/tb_ripple_carry_adder.vhd

9. vsim tb_ripple_carry_adder
10. add signals to wave form and run simulation

From now on, you shouldn't need to run vlib / vmap again.
You should be fine if you (after starting Modelsim) first change your directory to sim/ and after that continue from step 6 onwards. Just make sure that you are in sim/ directory when compiling and simulating - the (v)mapping is directory specific. You can always check that your library is mapped into right working library by running vmap without any parameters.

(Atleast later on when working with multiple files at the same time, using macros is a good way to speed up the compilation/simulation process.)

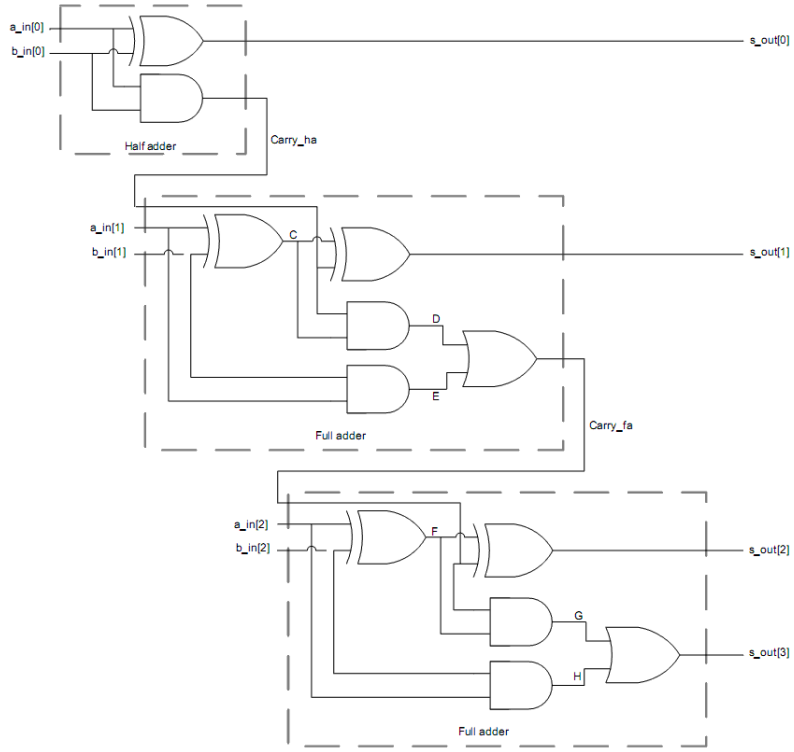
More tips on Modelsim usage can be found in modelsim tutorial (https://plus.tuni.fi/comp.ce.240/spring-2024/material/modelsim_tutorial/) that you used in last exercise and in general instuctions (https://plus.tuni.fi/comp.ce.240/spring-2024/material/misc/) page.

The actual exercise guide

- The purpose of this exercise is to introduce basic terms
  - *entity* which defines the interface of a block (and parameters)
  - *architecture* which defines the internal implementation of a block
  - gate level representation
  - basic logic gates
  - defining *signal*s
  - datatypes *std_logic* and *std_logic_vector*

- Your task is to represent a 3-bit ripple-carry-adder which has a 4-bit output (s = a + b)

- Use this circuit diagram



(https://plus.tuni.fi/graderA/static/compce240-f2021/E02/rca.gif)

- and use VHDL-file ripple_carry_adder.vhd (https://plus.tuni.fi/graderA/static/compce240-f2021/E02/ripple_carry_adder.vhd) as a base
- **Use only one entity (at this stage) and name the signals the same way as in the diagram**
- There are some further details and hints in the VHDL file
- There are also hints (https://plus.tuni.fi/comp.ce.240/spring-2024/material/misc/) for the directory structure
- When you have written the code, compile it together with the given test bench tb_ripple_carry_adder.vhd (https://plus.tuni.fi/graderA/static/compce240-f2021/E02/tb_ripple_carry_adder.vhd) using command **vcom -check_synthesis**
- Actually, it is advisable to compile the code constantly during the writing process. This way, finding the possible syntax errors is a lot easier.
- Verify the functionality using ModelSim (on linux: start using command vsim, on windows: start menu search modelsim)
  - Compile the entities and start the test bench to simulator
  - Run the simulation for 10 us
  - There should not be any error messages
  - Finally, the simulation should end stating # ** Failure: Simulation ended! (Failure is only a handy way to stop the simulation and in this it does not indicate error)
- Remember to comment your code and add header comments. Comments begin with "--" and they continue to the end of the row.

- Hints
  - Miscellaneous instructions (https://plus.tuni.fi/comp.ce.240/spring-2024/material/misc/) contain amongst other things an example of directory structure

- See the lecture slides for hints on VHDL syntax
- VHDL coding rules (https://plus.tuni.fi/graderA/static/compce240-s2024/lgs_vhdl_coding_rules_sl_v5_1.pdf) are ***enforced*** in *all* exercises. Failure to comply can and will result in a ***boomerang***!
- Quick reference of naming conventions (https://plus.tuni.fi/comp.ce.240/spring-2024/material/naming_rules/)
- Make sure that your code is indented and aligned to make readability better (e.g., automagically in Emacs: VHDL -> Beautify -> Beautify buffer)

Return:

- Put your returned files under `E02` folder in your Git repository
  - Return your own `ripple_carry_adder.vhd`
- Check that the file's header comments are valid, made according to instructions, and you have followed the coding rules
- Push the changes to your repository and submit (with your partner if you have a group!)
  - Use the **ssh variant** of the repository url in the submission. Otherwise the tests will fail 100% even with working design.
  - The url looks like *git@course-gitlab.tuni.fi:compce240-spring2024/<your_group_number>.git*.

Enter your Git repository address for grading

Did you remember git add - git commit - git push?

Submit with David Rama Jimeno Submit

Earned points

4 / 4

Exercise info

Assignment category
VHDL exercises

Your submissions

8 / 1000

Points required to pass
4

Deadline
Sunday, 28 January 2024, 23:59

Late submission deadline
Friday, 31 May 2024, 23:59

Group size
1-2

Total number of submitters
64

« 4. Exercise 2 (/comp.ce.240/spring-2024/... 5. Exercise 3 » (/comp.ce.240/spring-2024/...