M.EEC041 - Digital Systems Design

2022/2023

Laboratory 1 - V1.0
Sep 20, 2022

1. Introduction

In this exercise the students will be introduced to the digital simulation environment Questasim and review basic concepts of the hardware description language Verilog. After concluding this exercise the students should be able to:

1. setup a simulation project in Questasim and execute a basic simulation flow;
2. distinguish between the structure of Verilog models for supporting a simulation (*testbenches*) and the models that represent the digital circuit being verified or the *system under verification*;
3. identify the structure of basic Verilog design entities (**`module(…)`**) and the implementation of the interconnections among them.
4. understand the structure of an elementary Verilog *testbench* module.

2. Project setup

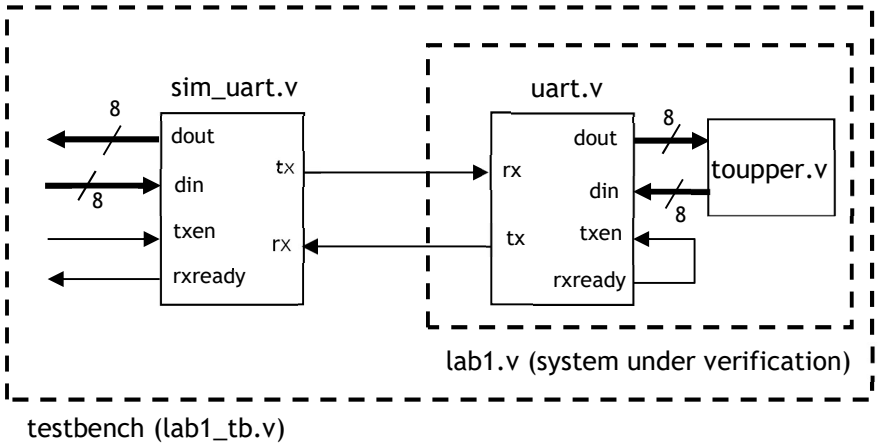
For this lab you should download the archive **PSD2223-T1.zip** available in the contents section in the course web page. This archive contains 4 Verilog files placed in different folders, as referred below. Unpack the archive to a convenient location under your home directory (for example **`<myfolder>/psdi/labs/lab1/`**…)[1], keeping the same folder hierarchy as in the original archive. Although this organization of files and folders is not mandatory, it is always a good practice to maintain the various files created during the development of a project organized in a convenient directory tree. For now, we will use the following tree structure:

```
<root project dir>
  |--src (all source files)
  |    |--verilog (all Verilog files)
  |    |   |--testbench (Verilog models for simulation only)
  |    |   |--rtl       (Verilog models for implementation)
  |    |--data (any additional source data files, not HDL)
  |--sim (simulation directory: create here the Questa project)
```

3. The system to simulate

The digital system to simulate in this lab consists in a UART transceiver (*Universal Asynchronous Receiver/Transmitter*) connected in loopback mode through a hardware implementation of the `toupper()` C function (convert lowercase letters to uppercase while any other characters are kept unchanged). The testbench model (`lab1_tb.v`) instantiates another UART (a simplified model, designed for simulation only) to simplify the generation of the simulation stimuli. Figure 1 represents a block diagram of the complete circuit that will be simulated in this exercise.

[1] Do not use a directory tree that contains spaces or special characters in the directory names. Some design tools do not process correctly such names, as for example:
`c:\ambiente de trabalho\utilizadores\...`)



1. Open the two Verilog files `lab1_tb.v` and `lab1.v` (use the text editor **notepad++**) and annotate the block diagram above with the names of the signals represented and add other signals not shown in the figure.

2. Execute QuestaSim and create a new project under folder `./sim`. Add to the project the five Verilog files found under directories `./src/verilog/testbench` (all sources related to testbenches) and `./src/verilog/rtl` (the models that will be later implemented into a physical digital circuit – the reason for the designation "rtl" will be explained later)

3. Compile all Verilog files (accept all the default compilation options) and confirm that there are no compilation errors. The compilation process translates the Verilog files into a proprietary format and places the compiled models in the working user library (the default user library name is `work` and it is created as a folder under the project directory).

4. Start the simulation by selecting the model `lab1_tb` from the `work` library and executing the command "*Simulate without optimization*" (for now we will execute a non-optimized simulation to keep the names of the signals in the simulation environment, as this is necessary for exporting them to the waveform viewer). Open the waveform viewer by executing the command `do wave_lab1.do` in the command shell and run the simulation with the command *Simulate->Run->Run -All* (or executing in the shell `run -all`).

5. The testbench `lab1_tb.v` includes a Verilog *task* (similar to a C function) to automate the verification process. During the simulation some errors are wrongly reported. Analyze and interpret the simulation results observed in the waveform window and in the text window and explain why the error reported is not a real error and what is wrong with the verification procedure.

Using Icarus Verilog and the waveform viewer GTKWave (free tools, open source)

Download and install Icarus Verilog from http://iverilog.icarus.com/home
Download and install GTKWave from http://gtkwave.sourceforge.net/

(installation packages for Windows64 also available in Moodle)

Icarus Verilog does not have a graphical interface and is executed from the command line with a text only interface. To run a simulation execute at the shell prompt:

```
iverilog <list of verilog files>
```

or

```
iverilog –c text_file_with_list_of_verilog_files   -o output_file_name
```

This compiles the Verilog sources to an intermediate format in file a.out (default filename) or to the file name specified by option -o. The simulation is executed with the command:

```
vvp  a.out
```

To dump the signal transitions to a file for later analysis, include in the Verilog top-level testbench the statement:

```
initial
begin
  $dumpfile("mysimdata.vcd");// The filename with the waveform data
  $dumpvars(0, lab1_tb );    // The root node to dump
end
```

To analyze the waveforms of the signals created during the simulation, run GTKWave and open the file specified in the Verilog function $dumpfile( ), in this example the file mysimdata.vcd