Visible Image Watermarking

Digital Design and Logical Synthesis for Electric Computer Engineering (36113611)
Course Project

Verification

Version 1.0

Revision History

Rev	Description	Done By	Date
1.0	Initial document	Yusef Ektilat, Roy Shor	17-Nov-2020
2.0			
3.0			

Contents

LIST OF TABLES

1. VERIFICATION PLAN

At the beginning of each phase, there is need to set the goals, limits, requirements, etc. In other words, we need to draw a plan for the whole phase. In this part of the project, **you are requested to design your own plan for the verification phase needed to be applied for the Visible_Watermarking system you designed in part I.**

Write your verification plan as explained\shown in Lab2, describe and explain your plan, why you chose such tests, how are you going to imply them. Make sure to cover both following scenarios:

1) Standard Scenarios – Regular Input.

2) Extreme Scenarios.

1.1 General Description

Define and Explain the full system design and its parts (your test bench units), describe each one of them (importance and role). In addition, describe the interface between these units (TB) and the device under test (DUT) – connections, and ports. Do not forget to describe the stimulus operations appropriately.

1.2 Full System Architecture

Show the Full system diagram, including all units (DUT and TBs) and the interface.

1.3 Functional Coverage

Build and Fill the Functional Coverage Table (As in Lab2) with the proper cases, the table contain the following sections:

- Function: Name of the function you want to cover.
- Event: At what event (when) the function coverage will be checked.
- Coverage Point: What points/ports is about to be checked.
- Bins: Which values must be covered.
- Scenario {New Section}: In which scenario the coverage is done (Standard or Extreme).

1.4 Functional Checker

Build and Fill the Functional Checker Table (As in Lab2) with the proper cases, the table contain the following sections:

- Condition: At what condition the function will be checked.
- Event: At what event (when) the condition will be checked.
- Expected Result: What the result supposed to be.
- Scenario {New Section}: In which scenario the coverage is done (Standard or Extreme).

2. VERIFICATION IMPLEMENTATION

Implement the Stimulus, Coverage, and Checker using SystemVerilog language. You can combine graphical designs (Verilog 2005) with SystemVerilog TB by creating a Verilog 2005 text files around the SystemVerilog Modules. Build an Overall TB that represent the full system (As in Lab2) using your own interface. Implement and Run the tests according to following parts in this section.

2.1 Functional Coverage

Implement the Functional Coverage TB using the Functional Coverage Table you have built (in section 1.3) in SystemVerilog (As in Lab2).

2.2 Functional Checker

Implement the Functional Checker TB using the Functional Checker Table you have built (in section 1.4) in SystemVerilog (As in Lab2).

2.3 Stimulus

Implement the proper Stimulus TB according to your verification plan in SystemVerilog (As in Lab2). You can use the example given in Lab2 to build your own generator. Be careful, you need to implement the APB BUS Protocol for Data Transmission (Storing Data in the DUT's Register-Bank).

2.4 Golden-Model

Build a general TB in SystemVerilog for comparing the DUT results to the Golden Model (which is implemented using MATLAB).

Tip: Make this TB appropriate to work with the Stimulus TB.

2.5 Interface & Overall TB

Build a proper Interface and overall TB that connect the Stimulus, Checker, Coverage, Golden-Model, and DUT using the interface.

3. VERIFICATION RESULTS

In this part you should show the result of your implementation using QuestaSim analyzer and menu:

$$"Tools \rightarrow Coverage\ Report \rightarrow HTML"\ /or/\ "Tools \rightarrow Coverage\ Report \rightarrow Text"$$

In addition, you can show the Covergroups results, assertions results, code coverage (from sim window) using the proper window is QuestaSim.

Then you need to explain the reports as follows:

3.1 Functional Coverage Report

Show and Explain the Functional Coverage report for each scenario (What Result you got and Why?).

Is there any improvement needed? If yes, then How to improve it?

3.2 Functional Checker Report

Show and Explain the Functional Checker coverage report for each scenario (What Result you got and Why?).

Is there any Error? If yes, then Why? and How to fix it?

3.3 Code Coverage Report

Show and Explain the Code Coverage report for each scenario (What Result you got and Why?).

Is there any improvement needed? If yes, then How to improve it?

Note: You need to analyze all code coverage techniques.

Now using the QuestaSim simulation reports/display and waveform you need to do the following test:

3.4 Golden Model Comparison

Compare and Explain the results of your DUT vs the given Golden-Model for same inputs (How much the design is good comparing to an ideal model? Is it Enough? Why there is such difference? Where is the critical part that defines this difference gap?)

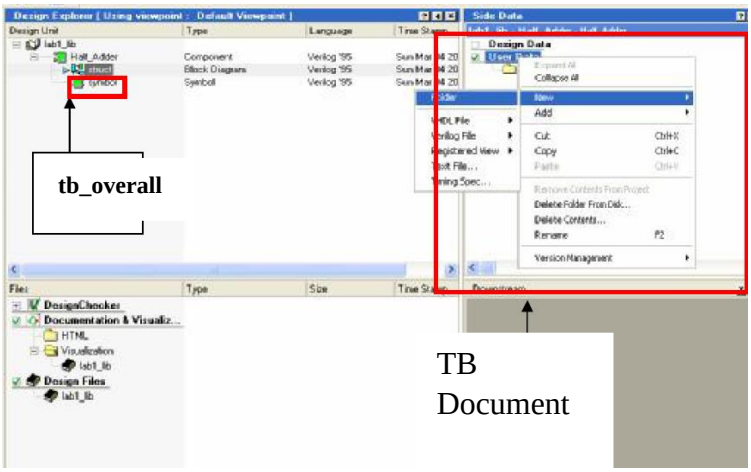
3.5 Formal Checker

You must have ‘0’ Warning after compiling each Verilog module.

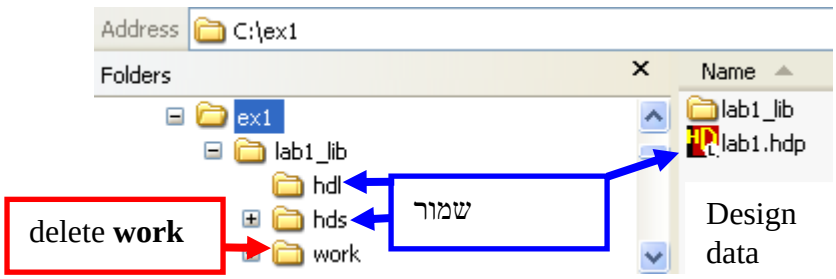
4. SUBMISSION REQUIREMENTS

For the second (Verification) part of the project you must submit the following:

1) **Test Bench Files** that containing the following units:
   - Coverage – File name: Coverage.sv
   - Checker – File name: Checker.sv
   - Stimulus – File name: Stimulus.sv
   - Interface – File name: Interface.sv
   - Golden-Model – File name: GoldModel.sv
   - Overall TB – File name: tb_overall.sv
2) **HDL Designer library including project files**, hdl and hds libraries. Top level module should be named **Visible_Watermarking**, and the file should be named **Visible_Watermarking.v**. Verilog 2005 syntax
3) **Short Verification Guide** (no more than 10 pages) containing the requirement described and shown in previous sections (in pdf format – you can use the template provided in the model under final project label: "Digital Block_02_General_Test_Plan_Template.doc").
4) **TB Document** should be attached in the **tb_overall** design in HDL designer using side data library



5) **Delete** the **work** library
6) Compress **Project file**, **hds** and **hdl** libraries into a ZIP file named ***<ID1>_<ID2>.zip***, where ID1 and ID2 are the ID (*teudat zeut*) numbers of the submitters.



7) All the library files should be zipped to a file named **id1_id2.zip** and submitted to Moodle.

8) Both Students Should Submit Their work to Moodle.

SUBMISSION DATE: 17/12/2020 AT 23:55

5. PROJECT GRADE EVALUATION

This part is **30 points** **from the project grade.** Preliminary checks must pass before your work is checked. **Failure in the preliminary checks means grade of 0 for this part of the project.**

Task	Description
Project opens HDL Designer and TBs compiles	Code compiles without compilation error in QuestaSim
APB BUS Protocol	APB can read/write data from/to all the registers

Table 1: Preliminary tests

After passing the preliminary tests the system's performance will be checked with additional tests.

grade	Task	Description
30%	TB Implementation	**Simulations compile and run. Using interfaces, assertion, cover groups, proper stimulus**.
30%	Document	Documents contain all requirement and reports, **easy to read and proper diagram for your Full System**, right explanations, and answers.
40%	Functional Tests	High Functional and Code Coverage, working tests, fully covered Functional Checkers.

Table 2: Grading policy

Late submission will degrade the project score by 5*days