Digital Design & Logic Synthesis
Course Project: Matrix Multiplication
2-Verification

Mike Pines

02/2024

Contents

1 Revision Notes **3**

2 Verification Plan **4**
- 2.1 General Description . . . 4
- 2.2 Full System Architecture . . . 4
- 2.3 Functional Coverage . . . 4
- 2.4 Functional Checker . . . 5

3 Verification Implementation **6**
- 3.1 Functional Coverage . . . 6
- 3.2 Functional Checker . . . 6
- 3.3 Stimulus . . . 6
- 3.4 Golden-Model . . . 6
- 3.5 Interface & Top-Level TB . . . 6

4 Verification Results **7**
- 4.1 Functional Coverage Report . . . 7
- 4.2 Functional Checker Report . . . 7
- 4.3 Code Coverage Report . . . 7
- 4.4 Golden Model Comparison . . . 7
- 4.5 Formal Checker . . . 7

5 Submission Requirements **8**
- 5.1 Base Project . . . 8
- 5.2 Test-Bench Files . . . 8
- 5.3 Short Verification Guide . . . 8

6 Submission **8**

7 Submission Evaluation **10**
- 7.1 Preliminary Test . . . 10
- 7.2 Checks & Grading . . . 10

1 Revision Notes

2 Verification Plan

In this part of the project you are required to design your own verification plan for the system you designed in part-1.
Your verification plan should follow the methods shown in lab-2, you should explain every part and your plan and how you chose to implement it.
Make sure you cover both

- Standard scenarios for regular input
- Extrememem scenraios for edge cases and unplanned input

2.1 General Description

Define and explain the full system design and its parts (test-bench modules), describe the role and importance of each module as well as the interface between them and the device under test (DUT). Describe the stimuli in detail.

2.2 Full System Architecture

Show the full system diagram including all modules (TB, DUT) and the interface.

2.3 Functional Coverage

Create and fill the functiona coverage table with the appropriate cases, the table should contain the following sections:

- Function: Name of the function you want to cover
- Event: When is the function going to be covered (checked)
- Cover Point: What points/ports/signals are checked
- Bins: What values are covered
- Scenario: Standard/Extreme

2.4 Functional Checker

Create and fill the functional checker table with the appropriate cases, the table should contain the following sections:

- Condition: Condition for checking
- Event: When is the condition going to be checked
- Expected Result: What the outcome is expected to be
- Scenario: Standard/Extreme

3 Verification Implementation

Implement the various test-bench modules (Stimulus, Checker, Coverage) using SystemVerilog.
Create a top-level TB that represents the full system using the appropriate interface.
Implement and run the tests according to the following sub-sections

3.1 Functional Coverage

Implement functional coverage using the functional coverage table in Verification Plan - Functional Coverage.

3.2 Functional Checker

Implement functional checker using the functional checker table in Verification Plan - Functional Checker.

3.3 Stimulus

Implement the stimulus in SystemVerilog according to your verification plan. Samples are provided in lab-2 but be careful, you must implement your own APB-master stimulus and make the proper adjustments to the samples to make them work with this project.

3.4 Golden-Model

Implement your own golden model in any high-level programming language (Python, MatLab).
Implement a TB for comparing the results from the DUT to the golden model.

3.5 Interface & Top-Level TB

Implement an interface and top-level TB to connect the various modules (Stimulus, Checker, Coverage, Golden-Model, DUT).

4 Verification Results

Show the results of your implementation using QuestaSim analyzer:
Tools → Coverage Report → HTML (Or 'Text') Show the covergroups, assertion results, code coverage (from sim window).
Explain the reports according to the following sub-sections:

4.1 Functional Coverage Report

Show and explain the functional coverage report for each scenario (what is the result and why was is received)
Are there any improvements possible? If yes, explain how to achieve them.

4.2 Functional Checker Report

Show and explain the functional checker report for each scenario (what is the result and why was is received)
Are there any errors? If yes, explain why and how to fix them.

4.3 Code Coverage Report

Show and explain the code coverage report for each scenario (what is the result and why was is received)
Are there any improvements possible? If yes, explain how to achieve them.
Note: you should utilize all code coverage techniques.

4.4 Golden Model Comparison

Compare and explain the results from your DUT to the given golden-model for similar inputs.

4.5 Formal Checker

There must be **0** warnings for all verilog modules.

5 Submission Requirements

5.1 Base Project

- HDL-Designer library including the **project file**, **hdl** & **hds** folders.
- The top level Verilog module must be called **matmul** and the file **matmul.v**, written in Verilog-2005.

5.2 Test-Bench Files

- Coverage : matmul_coverage.sv
- Checker : matmul_checker.sv
- Stimulus : matmul_stimulus.sv
- Interface: matmul_intf.sv
- Top-TB : matmul_tb.sv
- Golden-Model: matmul_golden.sv

5.3 Short Verification Guide

No more than 10 pages containing the requirements described in the previous sections, in PDF format. Use the template 'Digital_Block_02_General_Test_Plan_Template' TB document should be attached to the matmul_tb design in HDL-Designer.

6 Submission

- Attach the document to the top module in HDL-Designer using the 'Side Data' panel as required in the previous section.
- Delete the 'work' library.
- Compress the project file, hds and hdl directories into a zip file named <ID1>_<ID2>.zip where ID1 & ID2 are the teudat-zehut numbers of the pair.
- Submitted to Moodle by **both** students.

Submission Date : 24/02/24 at 23:59

Late submissions incur a penalty of 5 points per day.

7 Submission Evaluation

This part (verification) is 50% of the **total projects grade**.

7.1 Preliminary Test

Your code (Verilog & SystemVerilog) must **compile** without errors in HDL-Designer and QuestaSim.
APB can read/write data from/to the design.
Failure in the above means a 0 grade on this part of the project.

7.2 Checks & Grading

Grade-%	Task	Description
30%	TB Implementation	Simulations compile and run, using interfaces, assertions, cover-groups and proper stimuli.
30%	Documentation	Block diagrams, all the required reports, easy to read, correct explanations and answers.
40%	Functional Tests	High functional and code coverage, working tests, fully covered functional checkers.