

Digital Design & Computer Arch.

Lab 6 Supplement: Testing the ALU

Prof. Onur Mutlu

ETH Zurich

Spring 2021

20 April 2021

What Will We Learn?

- In Lab 6, you **learn** how to:
 - Verify the functionality of your designs using testbenches.
 - Find and resolve bugs in your design.

- You will:
 - Write a testbench that verifies the correctness of your ALU from Lab 5.
 - Use the same testbench to find and fix bugs in a buggy ALU that we provide.

Preparation

- **You are expected to finish Lab 5 before continuing,** because we will be testing the ALU from Lab 5.

- Download the material for Lab 6, which includes:
 - A template **testbench file**;
 - A template for the **test-vectors**;
 - A Verilog description of an ALU, which **contains some bugs**.

Part 1: Expected Results

- Before writing our testbench, we need to prepare a set of inputs for which the **expected results** are known.
- You will be given **a set of inputs** for the ALU you designed in Lab 5.
- Determine the **correct result** for each set. Then, specify them in the file **testvectors_hex.txt** that we provide.
- **For output 'zero'**: directly set its expected value within the testbench

Part 2: Preparing the Testbench

- Create a project with your ALU from Lab 5 and the testbench template we provided you with.
- Make the necessary modifications to the testbench.
- After this, you will have a testbench that will
 - Apply the vectors in the **testvectors_hex.txt** file;
 - Check the **actual outputs** of our ALU against **what we expect**.

Part 3: Simulating the ALU

- Run behavioral simulation using Vivado's built-in simulator.

The screenshot displays the Vivado 2019.2.1 interface during a behavioral simulation of an ALU. The left sidebar shows the Project Manager with the SIMULATION section active. The main workspace is divided into several panes: Scope, Objects, Tcl Console, and a waveform viewer. The Scope pane shows the ALU test design unit containing uut and glbl modules. The Objects pane lists variables such as a[31:0], b[31:0], result[31:0], and clk. The Tcl Console shows simulation completion messages, including 'XSim completed. Design snapshot 'ALU_test_behav' loaded.' and 'XSim simulation ran for 1000ns'. The waveform viewer displays signals over time, with a yellow box highlighting the '1,00' ns mark. A yellow callout box on the right contains the text '1. Errors' and '2. Display messages', with an arrow pointing to the Tcl Console.

1. Errors
2. Display messages

Part 4: Debugging the Problem

- Using a simulator can help you locate the problems in your circuits.
- You can not only observe the outputs but the state of all internal variables as well.

The screenshot displays the Vivado 2019.2.1 interface during a behavioral simulation of an ALU circuit. The main window is titled "SIMULATION - Behavioral Simulation - Functional - sim_1 - ALU_test".

Project Manager: Shows the project structure with folders for PROJECT MANAGER, IP INTEGRATOR, SYNTHESIS, IMPLEMENTATION, and PROGRAM AND DEBUG.

Simulation Window: Displays the simulation progress and a table of objects. The table lists various signals and their current values:

Name	Value	Data Type
a[31:0]	00001100	Array
b[31:0]	00001000	Array
aluop[3:0]	a	Array
result[31:0]	00000001	Array
zero	0	Logic
st[31:0]	00000001	Array
logicse[31:0]	00001100	Array
alu_val[31:0]	00000001	Array
diff[31:0]	ffff1100	Array
ss0	0	Logic
ss1	1	Logic
ss2	0	Logic
ss3		

Waveform Viewer: Shows a time-based view of the signals. The signals listed include a[31:0], b[31:0], aluop[3:0], result[31:0], zero, clk, exp_result[31:0], exp_zero, vec_cnt[10:0], err_cnt[10:0], and testvec[100:0][99:0]. The time axis is labeled "1.00 ns".

Tcl Console: Shows the simulation log with the following output:

```
xsim: Time (s): cpu = 00:00:08 ; elapsed = 00:00:05 .  
INFO: [USF-XSim-56] XSim completed. Design snapshot 'ALU_test' saved.  
INFO: [USF-XSim-97] XSim simulation ran for 1000ns  
launch_simulation: Time (s): cpu = 00:00:10 ; elapsed = 00:00:05 .
```

Last Words

- In Lab 6, you learn how to
 - write testbenches in Verilog to verify the functionality of the design.
 - Find and resolve bugs in your design
- Write a testbench that verifies the correctness of your ALU from Lab 5.
- Use the same testbench to find and solve bugs in a buggy ALU that we provide.
- In the report, you will design a testbench for your FSM from Lab 4.

Report Deadline

23:59, 14 May 2021

Digital Design & Computer Arch.

Lab 6 Supplement: Testing the ALU

Prof. Onur Mutlu

ETH Zurich

Spring 2021

20 April 2021