

# Digital Design & Computer Arch.

## Lab 6 Supplement: Testing the ALU

Prof. Onur Mutlu

ETH Zurich

Spring 2024

8 April 2024

# 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 2: Why use a Testbench?

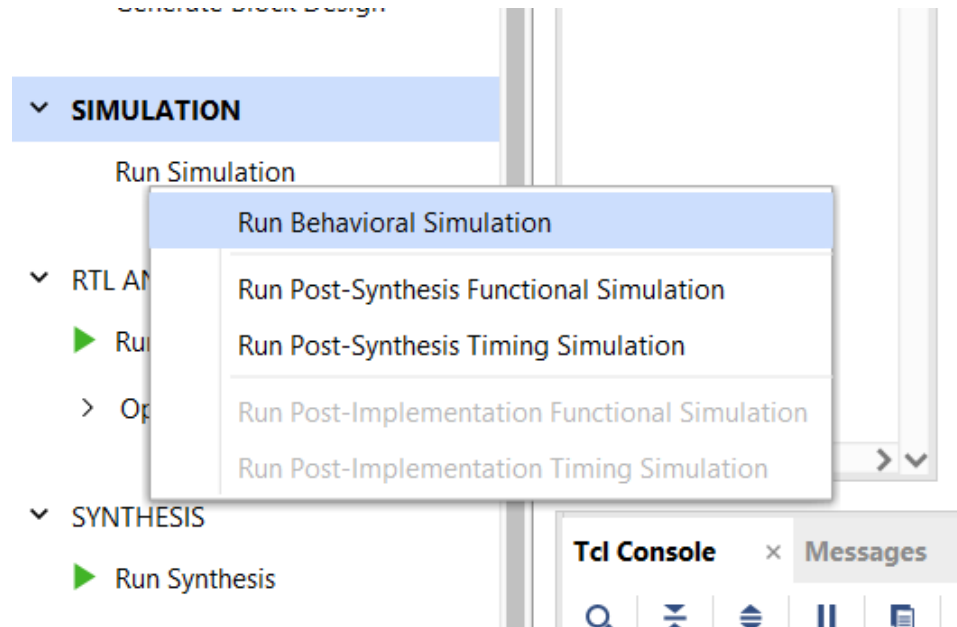
---

- In Lab 5 we have seen that **exhaustive search** is not a feasible option anymore.
- Instead we test our ALU on a set of **representative values** using the testbench.
  - If the ALU can add 1 and 2 it can probably also add 1 and 3.
  - For **each operation** test if it works for **some examples**.

# Part 3: Simulating the ALU

---

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



# Part 3: Simulating the ALU: Caching

- It can happen that Vivado fails to **recognise file changes**
  - It then uses cached (and **outdated**) information for the behavioral simulation.
- This can be fixed by **deleting all cached files** and then doing the behavioral simulation again.

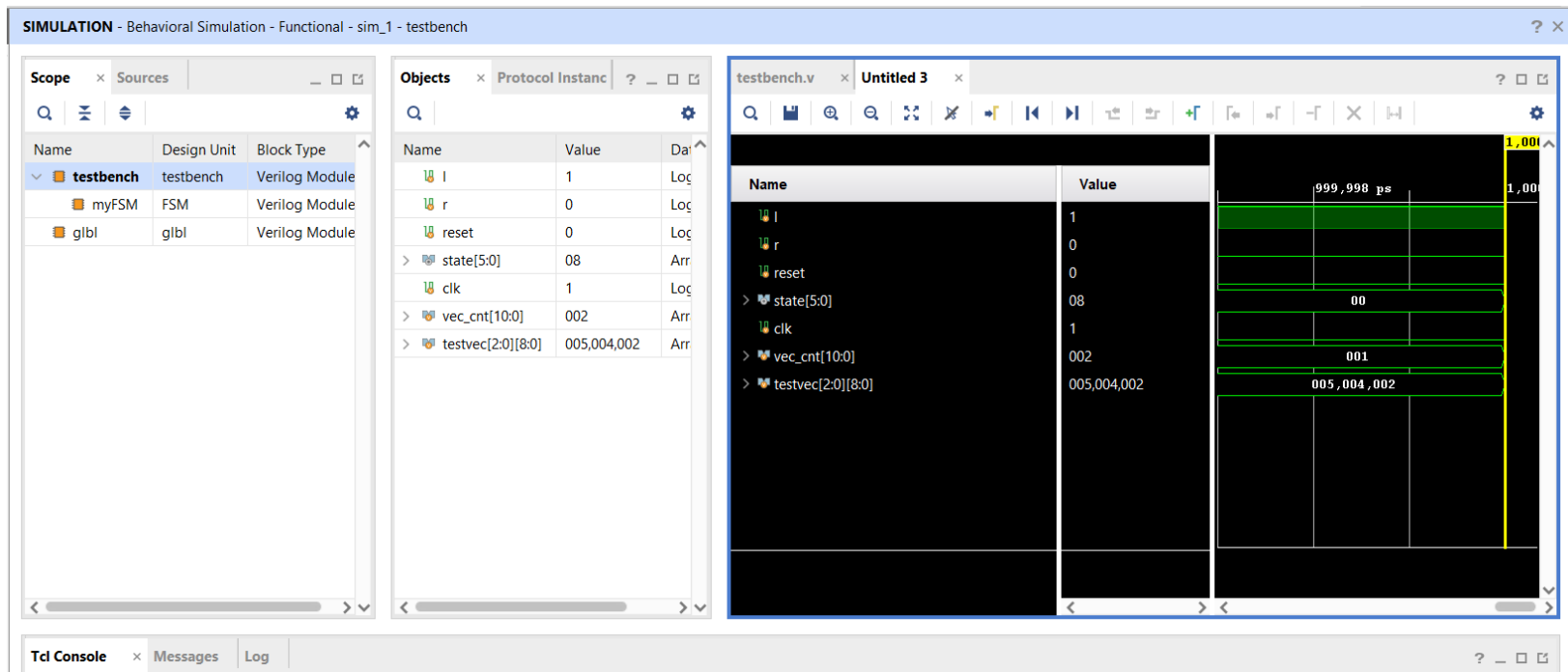
Name	Date modified	Type	Size	
.Xil	01/02/2024 09:48	File folder		
Lab4.cache	01/02/2024 09:49	File folder		can be deleted
Lab4.hw	01/02/2024 09:45	File folder		
Lab4.ip_user_files	01/02/2024 09:48	File folder		
Lab4.runs	01/02/2024 09:48	File folder		
Lab4.sim	01/02/2024 09:48	File folder		
Lab4.srcs	01/02/2024 09:48	File folder		
Lab4	01/02/2024 10:09	Vivado Project File	13 KB	
upgrade_project_migration_report	01/02/2024 09:48	Text Document	3 KB	
vivado.jou	01/02/2024 09:46	JOU File	1 KB	can be deleted
vivado	01/02/2024 09:46	Text Document	1 KB	
vivado_pid6000.str	01/02/2024 09:46	STR File	0 KB	

do not delete



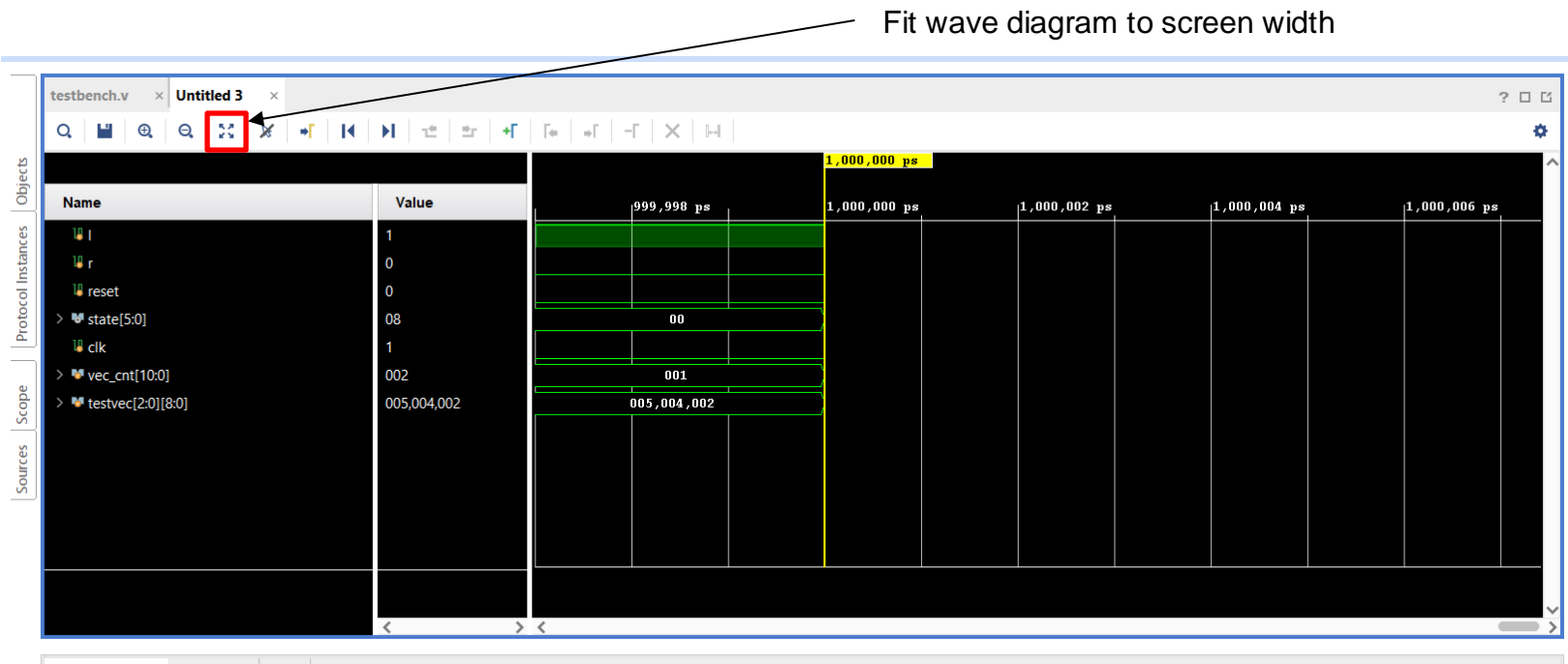
# Part 4: Debugging the Problem

- 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.



# Part 4: Wave Diagrams Introduction

- At first the **wave diagrams** are not very useful.
  - They are **unformatted**
  - Data is displayed in **hexadecimal**



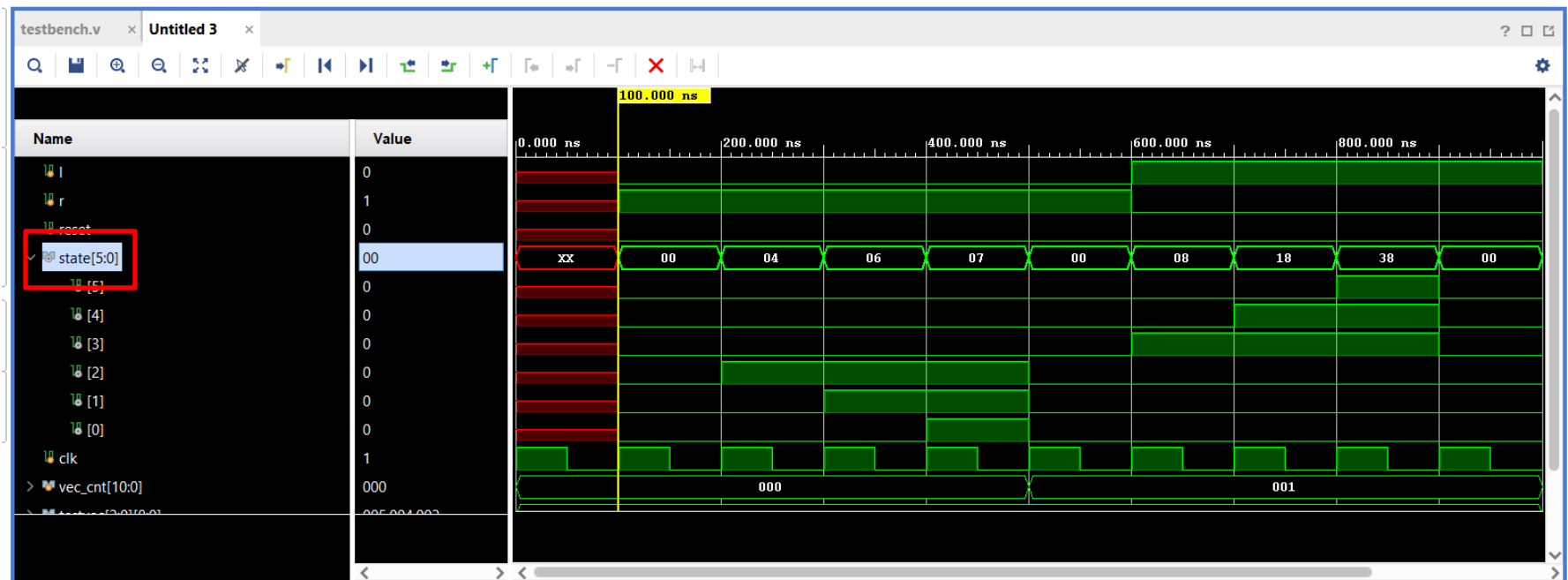
# Part 4: Wave Diagrams Introduction

- At first the wave diagrams are not very useful.
  - They are **unformatted**
  - Data is displayed in **hexadecimal**



# Part 4: Wave Diagrams Introduction

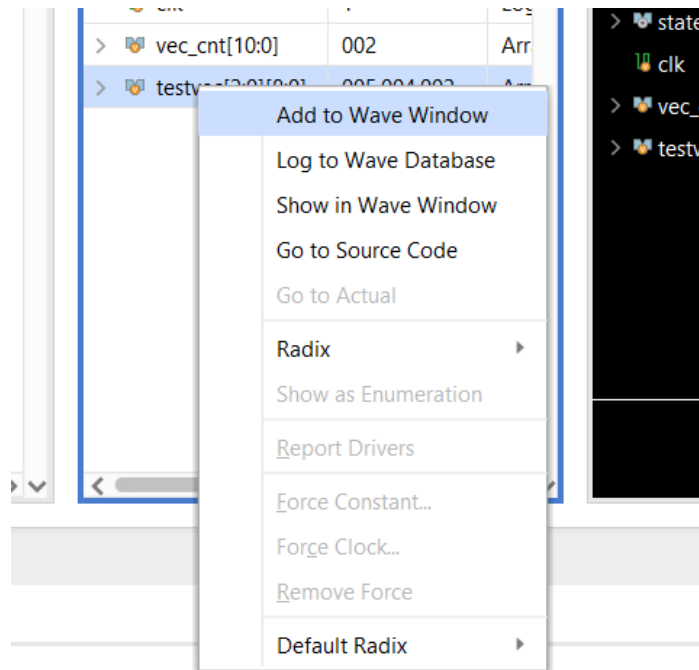
- **Selecting an Object** in the wave diagram allows you to navigate the wave diagram using the < and > keys.
  - Allows you to **jump between wave fronts**.



# Part 4: Wave Diagrams Introduction

---

- You can **add other signals** to the wave diagram.
- This is useful for **debugging** the *Unit-Under-Test* (UUT).



```
.. Design snapshot 'testbench_behav' loaded.  
n ran for 1000ns
```

# 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

---

**[09 May 2025 23:59]**

# Digital Design & Computer Arch.

## Lab 6 Supplement: Testing the ALU

Prof. Onur Mutlu

ETH Zurich

Spring 2024

8 April 2024