ALU PROJECT

AMRUTA.H

EMP ID:6067

INTRODUCTION

Arithmetic logic unit (ALU) is a combinational digital circuit that performs arithmetic and bitwise operations on integer binary numbers. It is responsible for carrying out arithmetic operations such as addition and subtraction, as well as logical operations like AND, OR, and NOT. This project presents the implementation of a parameterized ALU designed using Verilog HDL. The ALU supports a wide range of operations including arithmetic, logical, shift, and rotate instructions. The goal is to create a modular, synthesizable, and simulation-verified ALU that can serve as a reliable component in larger digital design projects. The ALU design also includes comprehensive flag generation to support conditional operations in a processor pipeline.

OBJECTIVES

- The main objective is to implement a fully functional ALU in Verilog that supports both arithmetic and logical operations.
- The design aims to handle multiple instructions such as addition, subtraction, bitwise logic, shifting, and rotation.
- Another key objective is to generate appropriate status flags such as carry, overflow, equal, greater than, less than, and error, based on the results of operations.
- The design must be verified thoroughly using a Verilog testbench and waveform simulations to ensure its correctness.
- The project also aims to maintain modular and reusable code for easy maintenance and scalability.

ARCHITECTURE

The architecture of the ALU consists of clearly defined input and output ports, which include two operands (OPA and OPB), control signals, a command selector, and result and flag outputs. The operands are designed with parameterized widths, providing the flexibility to adapt the ALU to different bit-length requirements. Control signals such as mode, clock, reset, and clock enable help in managing the operation flow, while the command input determines the specific operation to be executed. The mode signal determines if the operation to be performed is arithmetic or logical. Internally, combinational logic circuits interpret these inputs to perform the corresponding function. The ALU outputs the result of the operation along with status indicators like carry-out (COUT), overflow (OFLOW), greater (G), less (L), equal (E), and error (ERR), which provide critical information about the outcome of each instruction.

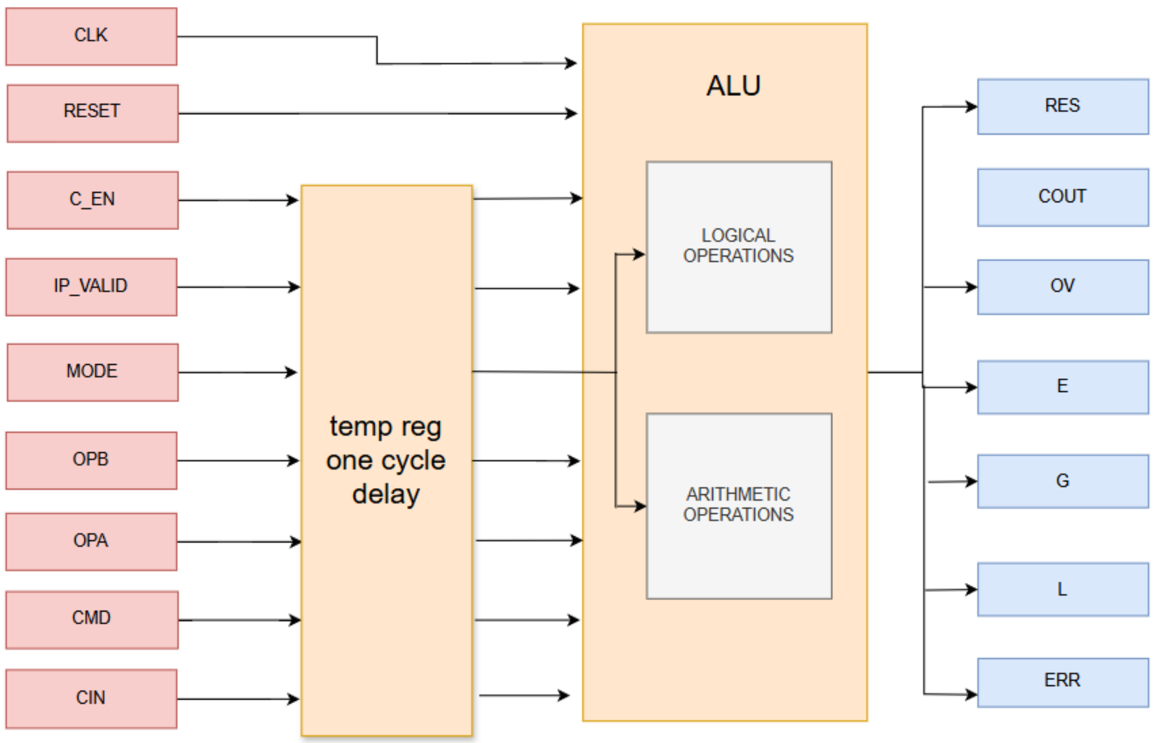


Figure 1: Design Architecture

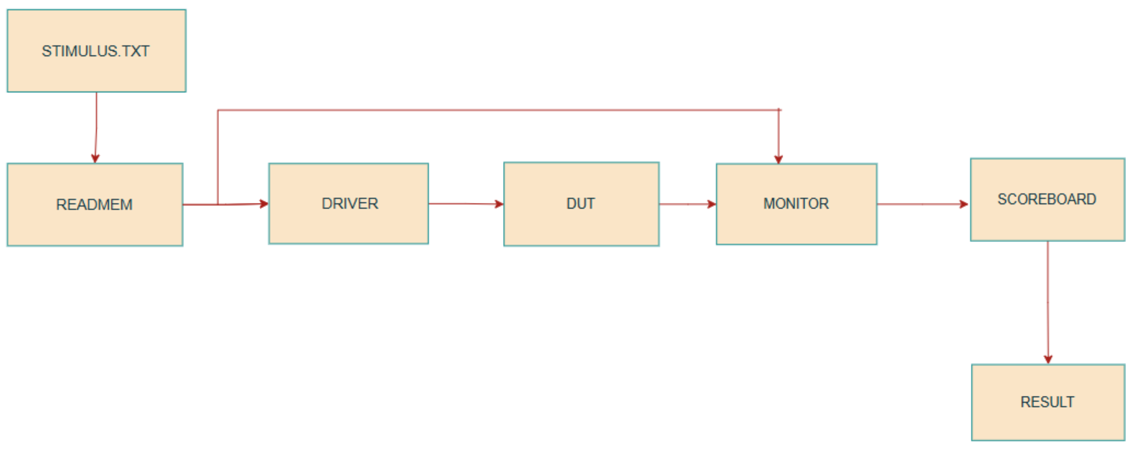


Figure 2: Testbench Architecture

WORKING

The Arithmetic Logic Unit (ALU) is a combinational digital circuit that performs a wide range of arithmetic and logical operations, such as addition, subtraction, AND, OR, XOR, and various shift or rotate instructions. Controlled by a command signal (CMD), it operates on two input operands of configurable width. Internally, it decodes the command and routes inputs through corresponding logic blocks—adders/subtractors for arithmetic, and logic gates for bitwise operations. It produces an output result (RES) and sets status flags like carry-out (COUT), overflow (OFLOW), equal (E), greater (G), less (L), and an error flag (ERR) for invalid commands. These flags are essential for decision-making in processors. While the ALU core is combinational, it may also include clock, reset, and enable signals for integration with synchronous systems.

The ALU module is a parameterized and versatile design supporting both signed and unsigned operations based on a mode selector. With a validity signal (ip_valid) and a control enable (ce), the ALU can operate on one or both operands. In arithmetic mode, it performs signed/unsigned additions, subtractions, comparisons, and multiplication with a 2-cycle delay using a control counter and a flag. In logical mode, it handles bitwise operations and both standard and circular shifts/rotates, with error detection for invalid shifts. Outputs include the result, carry, overflow, and comparison flags. Internally, temporary registers hold synchronized inputs for stable computation. This design supports both basic and advanced processing needs efficiently.

RESULTS

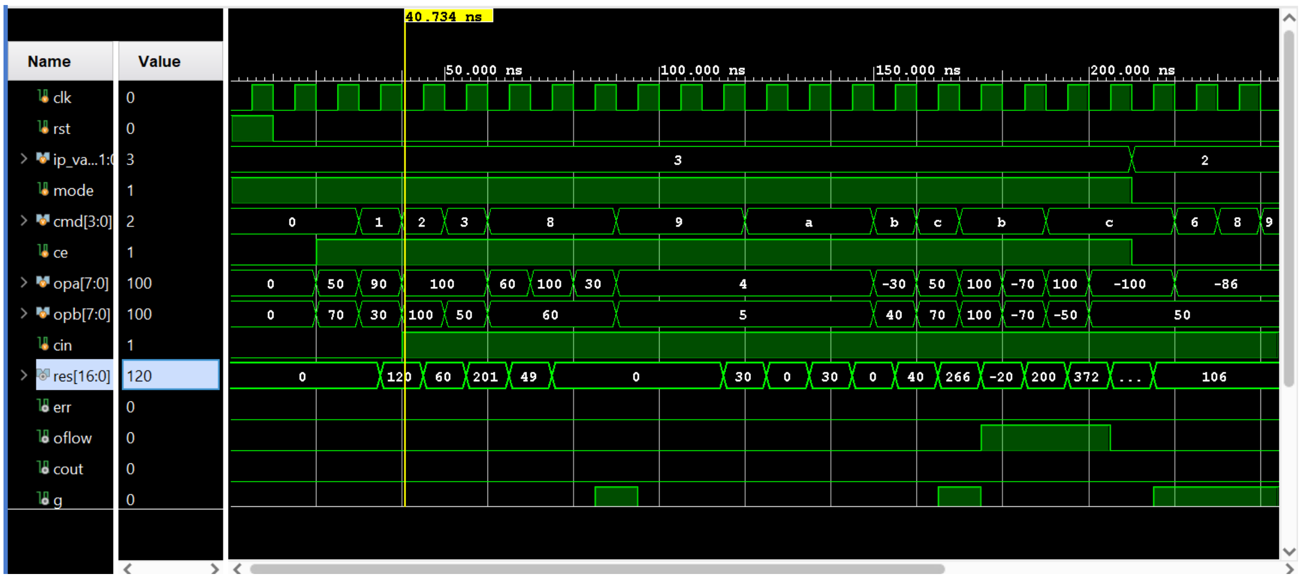


Figure 3: waveform of operations

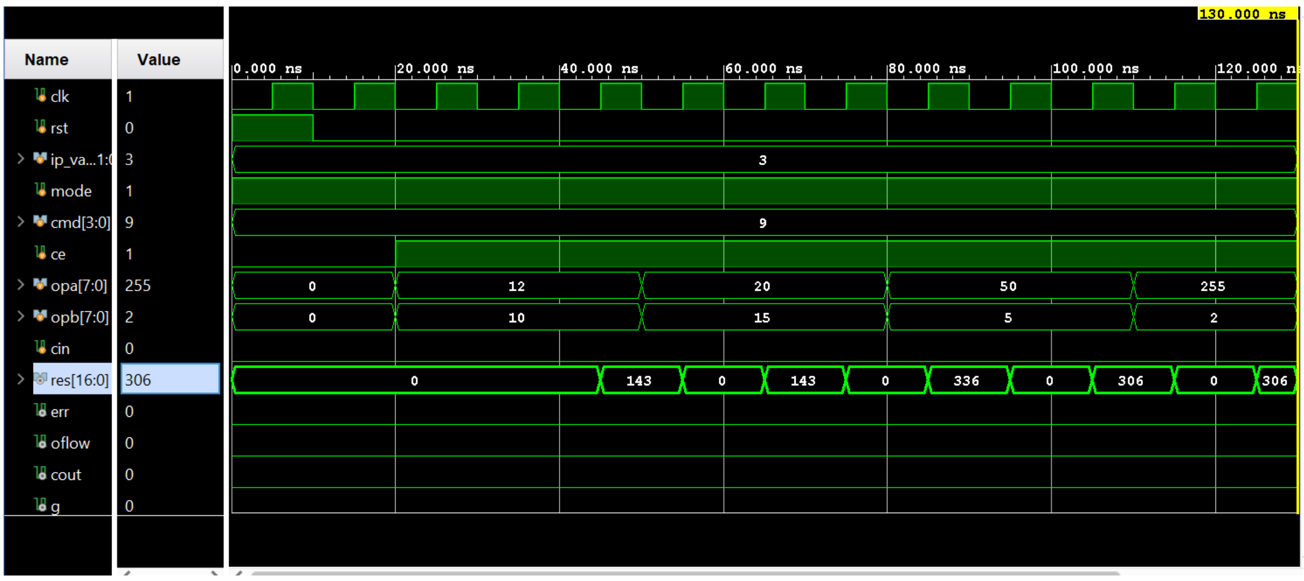
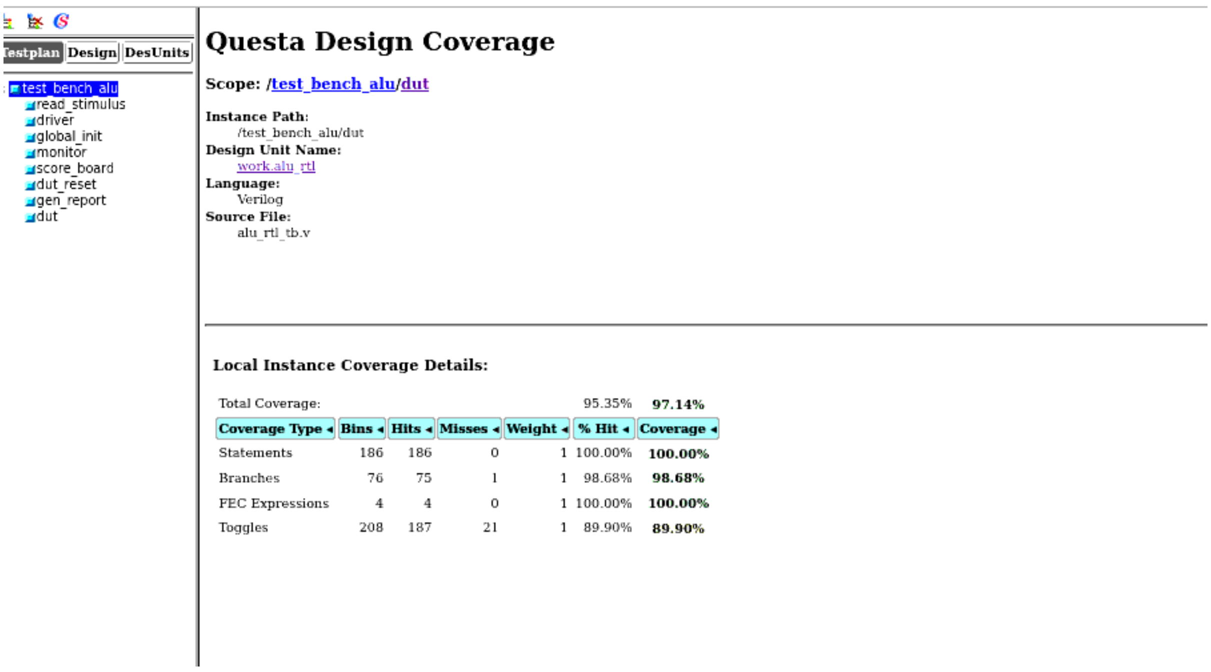


Figure 4: waveform of multiplication operation



Questa Design Coverage

Scope: /test_bench_alu/dut

Instance Path:
/test_bench_alu/dut
Design Unit Name:
work.alu_rtl
Language:
Verilog
Source File:
alu_rtl_tb.v

Local Instance Coverage Details:

Total Coverage: 95.35% **97.14%**

Coverage Type	Bins	Hits	Misses	Weight	% Hit	Coverage
Statements	186	186	0	1	100.00%	**100.00%**
Branches	76	75	1	1	98.68%	**98.68%**
FEC Expressions	4	4	0	1	100.00%	**100.00%**
Toggles	208	187	21	1	89.90%	**89.90%**

Figure 5: code coverage

CONCLUSIONS

In conclusion, the ALU was successfully implemented in Verilog and verified through simulation. It supports a comprehensive range of operations, from simple arithmetic and logical instructions to more advanced shift, rotate, and comparison functions. The design includes detailed flag generation and error detection, making it suitable for integration into more complex systems such as processors or controllers. The code is modular and parameterized, ensuring reusability and ease of expansion. Simulation results confirm the functional correctness of the design and validate that all requirements were met.

FUTURE IMPROVEMENTS

Introduce pipelining to break the ALU into multiple stages (fetch,decode, execute, write-back), enabling higher throughput and making it suitable for use in CPUs and high-performance systems. Functionality can be improved by adding floating point, complex numbers.