Jonathan Honrada
CPE233

OTTER ALU Info

1. Black Box Diagram

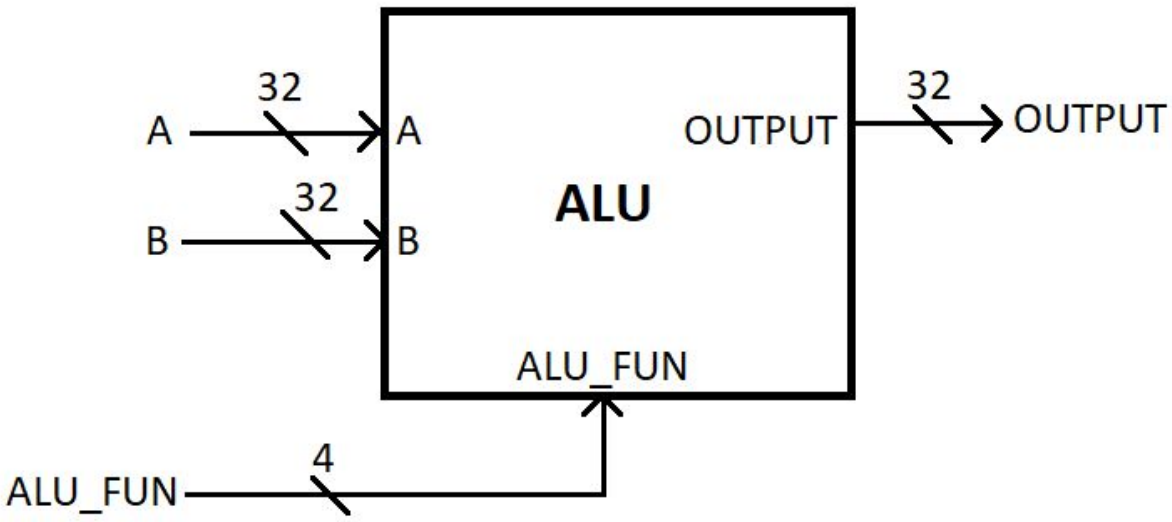

Figure 1. Black Box Diagram of ALU

2. Behavior Description

The ALU module takes two 32-bit inputs and performs some operation involving those input signals that is dependent on the 4-bit ALU_FUN signal. The result of this operation is output on the 32-bit OUTPUT signal. All logic in this circuit is combinatorial, but was designed using SystemVerilog behavioral modeling.The following table displays the operations performed depending on the ALU_FUN signal.

{func7[5],func3}

alu_fun	operation
0000	add
1000	sub
0110	or
0111	and
0100	xor
0101	srl
0001	sll
1101	sra
0010	slt
0011	sltu
1001	copy (lui)

Table 1. ALU Operation Codes

3. Structural Design

This design was generated using Vivado's Elaborated Design feature. A link to a higher resolution version is given.

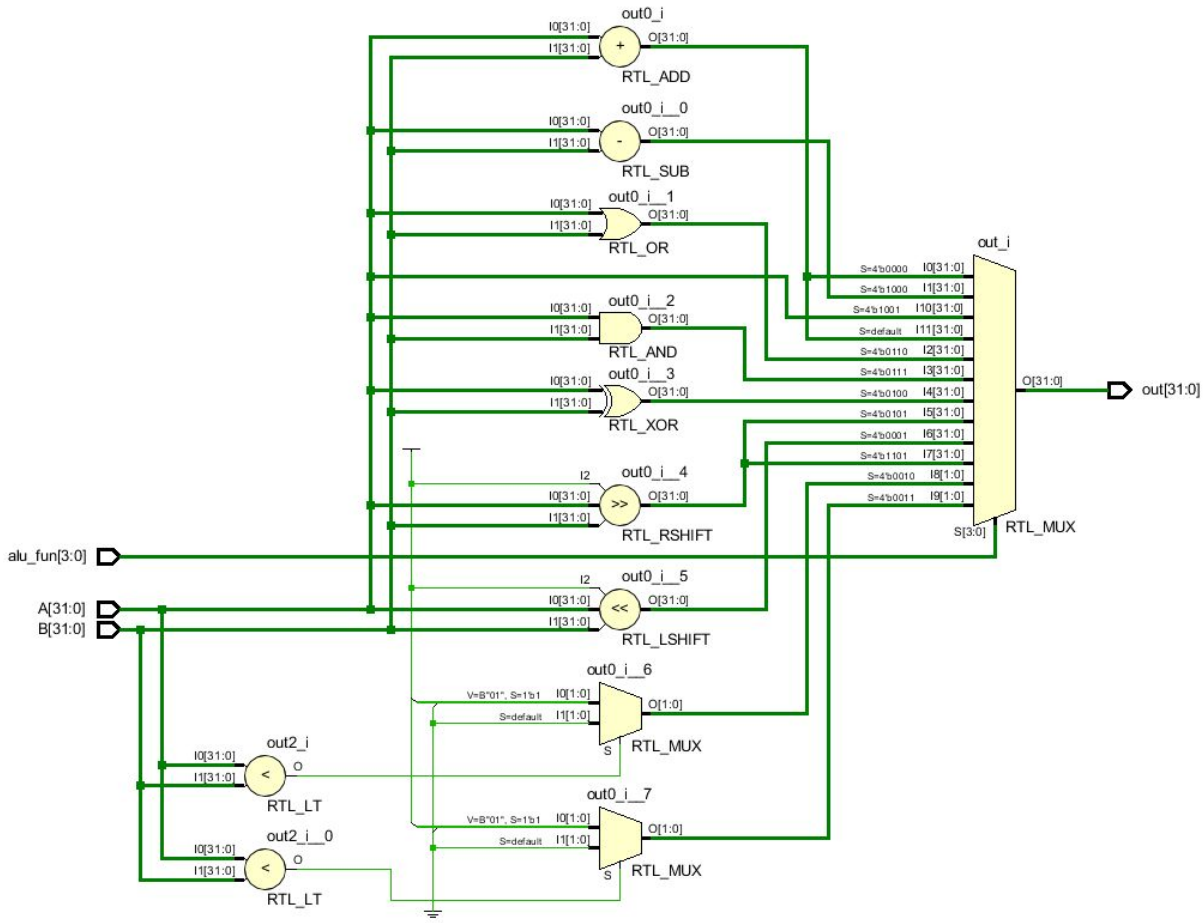

Figure 2. Structural Design of ALU (Higher res: https://i.imgur.com/uHaw0iz.png)

4. Verification

A screenshot of our simulation results is provided. The top row represents the A input, the second row represents the B input, the third row represents the alu_fun input, and the last row represents the output. A link to a higher resolution image is provided as well as our verification test cases table.

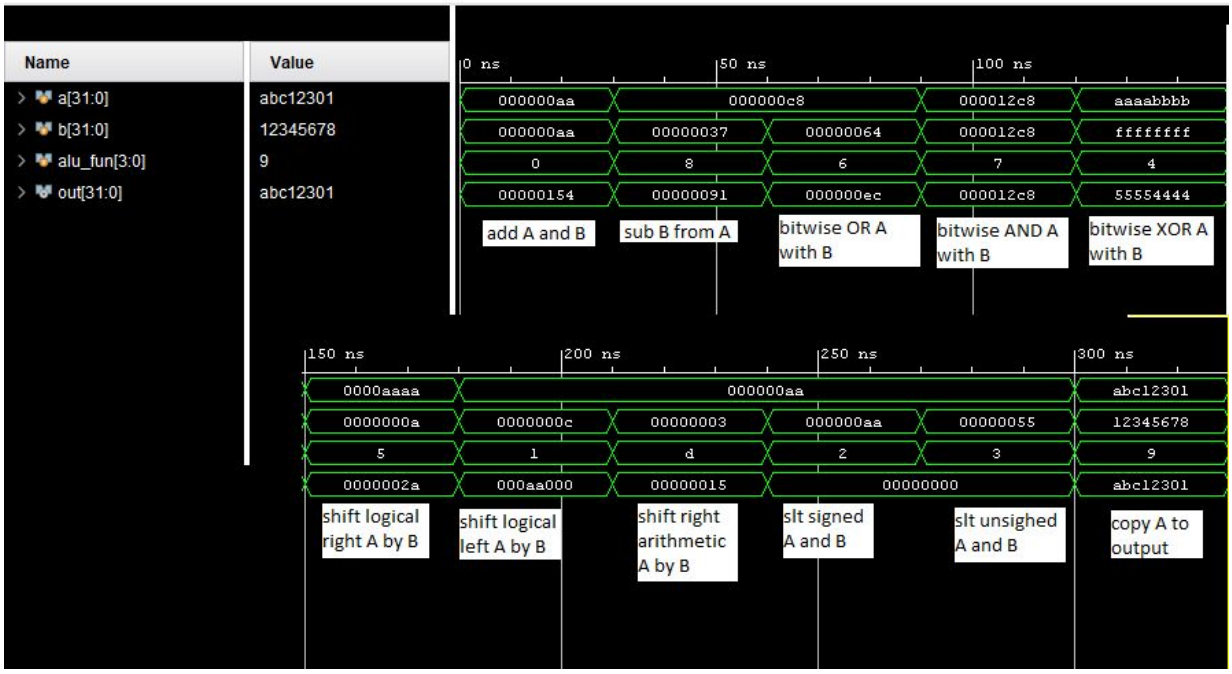

Figure 3. Annotated Simulation Results for ALU(Higher res: https://i.imgur.com/zaUb8nv.png)

Instr.	Inputs (A, B)	Expected Results (hex)	Simulated Value	Correct?
add	0xAA, 0xAA	0x154	0x154	✔
sub	0xC8, 0x37	0xEC	0xEC	✔
or	0xC8, 0x64	0x12C8	0x12C8	✔
xor	0xAAAABBBB, 0xFFFFFFFF	0x55554444	0x55554444	✔
slr	0xAAAA, 0x0A	0x2A	0x2A	✔
sll	0xAA, 0x0C	0xAA000	0xAA000	✔
sra	0xAA, 0x03	0x15	0x15	✔
slt	0xAA, 0xAA	0x0	0x0	✔
sltu	0xAA, 0x55	0x0	0x0	✔
copy	0xABC12301, 0x12345678	0xABC12301	0xABC12301	✔

Table 2. Summary of ALU Test Cases