Processor Components Implementation I
ALU DESIGN

Huy Duong Eiman Nour Uyen Le Wilson Pinales

April 6th ,2020

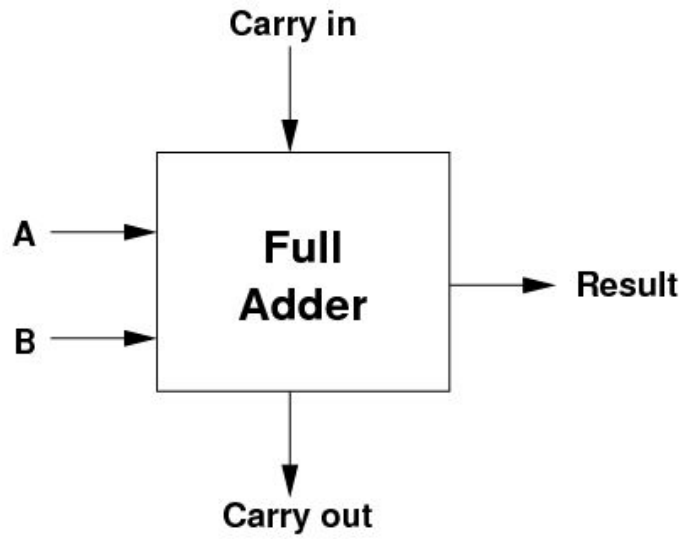
Objective

The objective of this project is to design a 32-bit ALU to perform all the arithmetic, logic and shift operations required by data path, and model the designed 32-bit ALU in Logisim, then test the correct functionality of all operations implemented by the ALU.

Design and Implementation

The designed ALU has four main units as arithmetic unit, comparator unit, logic operation unit and shifter unit to perform arithmetic, comparison, and logic and shift operations, respectively. In order to select the output from either the arithmetic unit, the comparator unit, the logical operation unit for the shifter unit, a 32-bit 4 x 1 multiplexer is used with 2-bit ALU selection signals.

1. Arithmetic Unit:
    a. ADD operation:
    - We build a full adder that will perform 1-bit ADD, take carry in, give 1-bit output (result) and a carry-out.



- The below is the truth table for the full adder:

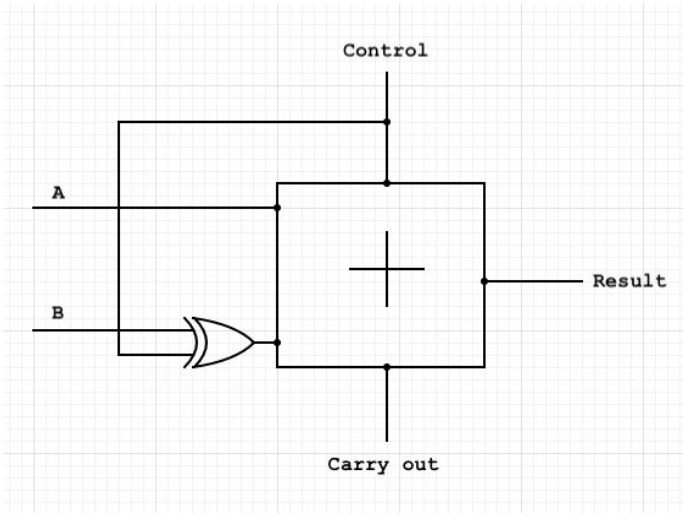
Cin	A	B	Result	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

b. SUB operation:

- Instead of building a completely new one for SUB, we use the full adder which was built for the ADD operation and an XOR gate.

- We know that if we have:

  A - B = A + (-B) = A + ~B + 1
- This is how we build the SUB operation from the full adder



  - If we set the control = 0, bit of input B will remain unchanged, then the Cin = 0. Thus we do ADD
  - If we set the control = 1, bit of input B will be inverted when going into the XOR gate, then the Cin = 1. Thus we do SUB.

2. Comparator Unit:

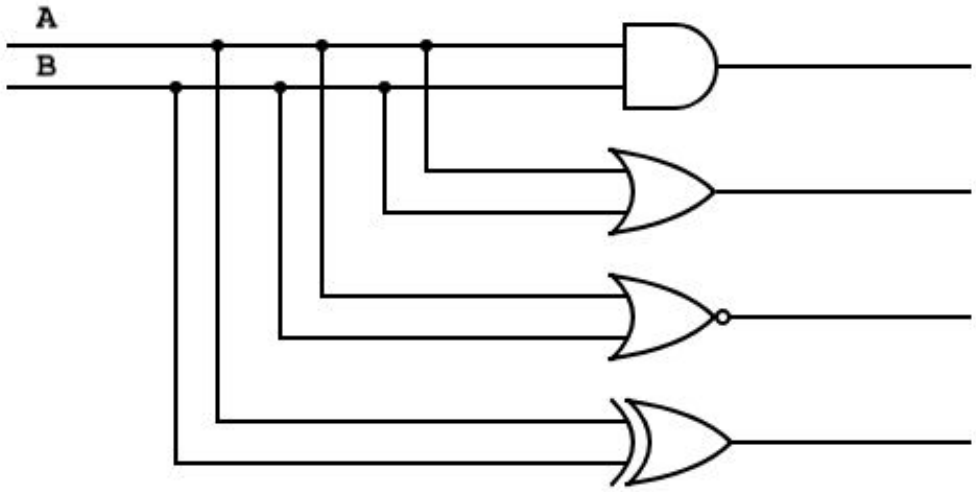
The comparator unit is mainly used for comparing signed and unsigned numbers and it shares the 32-bit full adder with the Arithmetic Unit. For the MIPS ISA implementation, we need to compare if a number is less than another number for implementing the set on less than instructions: SLT, SLTU, and SLTI

- To compare signed numbers A and B, we perform the subtraction operation A – B. This can be done by XORing the Sign and the Overflow signals. If the result is 1, this means that A < B, otherwise, means A ≥ B.
- To compare unsigned of two numbers A and B, we also need to perform a subtraction operation, A - B, and then check whether we have a Carry-Out or not. We use a signed number full-adder to perform unsigned number subtraction. If Carry-Out=0, this means that there was a borrow, thus A < B. However, if Carry Out=1 then this implies that there was no borrow and hence A ≥ B.

3. Logical Unit:

The designed logical operation unit performs AND, OR, NOR, and XOR logical operations for MIPS logic instruction. AND, OR, NOR and XOR logical gates outputs will be forwarded to the inputs to a 32-bit 4 x 1 multiplexer. The

logical operation unit has two control bits to determine the logical operation is AND, OR, NOR, or XOR, which will be connected to the input selection bits of the multiplexer.



4. Shift Unit:

The shifter block is used to implement SLL (shift left logic), SRL (shift right logic), SRA (shift right arithmetic) and ROR (rotate right) MIPS shift instructions.

- shift op = SRL, then ext_data[62:0] = 031 || data[31:0]
- shift op = SRA, then ext_data[62:0] = data[31]31 || data[31:0]
- shift op = ROR, then ext_data[62:0] = data[30:0] || data[31:0]
- shift op = SLL, then ext_data[62:0] = data[31:0] || 0

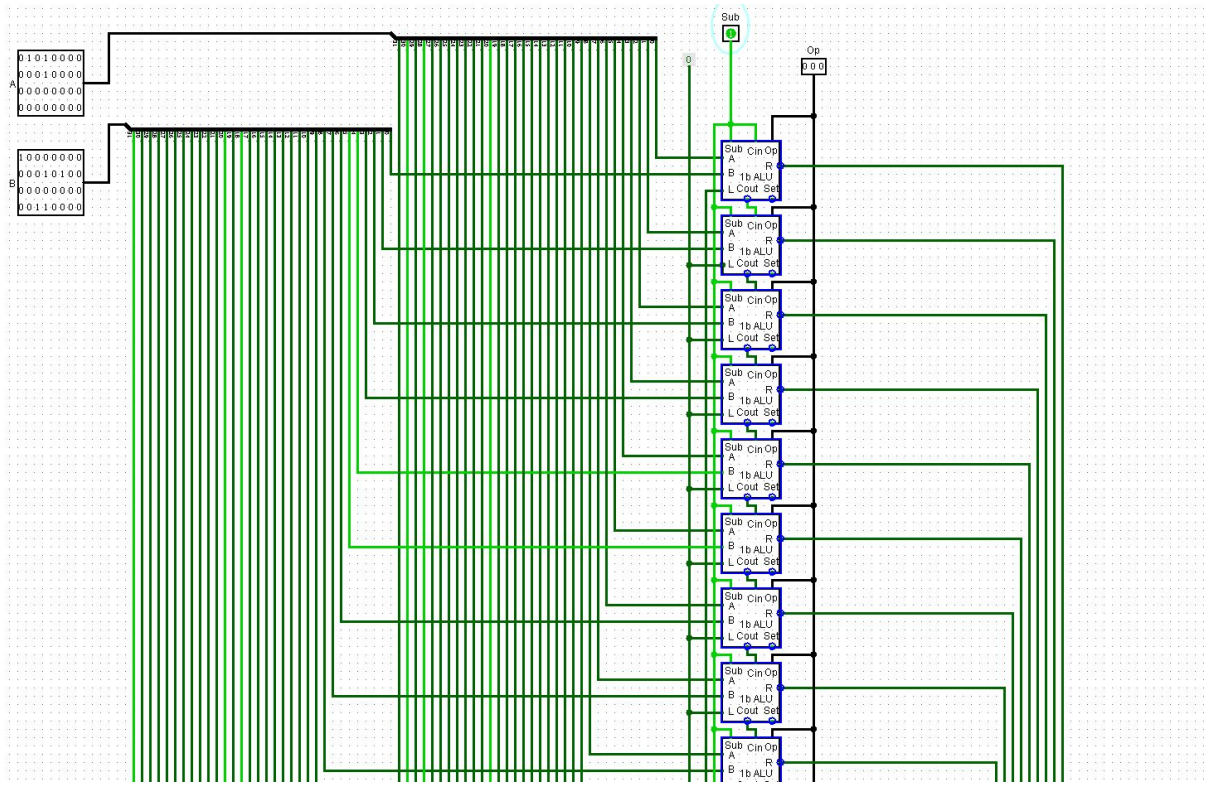
Simulation and Testing

Below is the testing table:

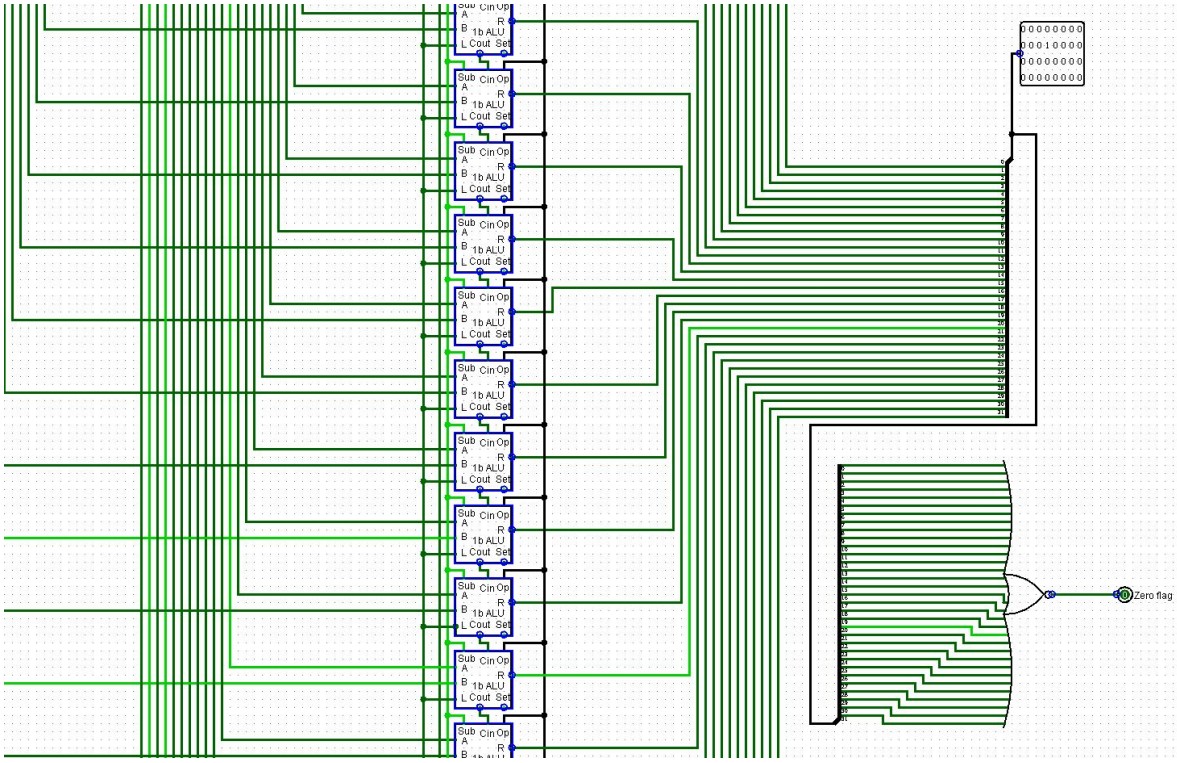
Function	ALU Op Code	Sub Flag (fed in from ALU Control)
and	000	0
or	001	0
xor	010	0
add	100	0
sub	100	1
slt	101	1

Below is a screenshot for some of the testing sample:

1 bit ALU test



32 bit ALU testing



32 bit ALU testing (the rest of the previous picture)