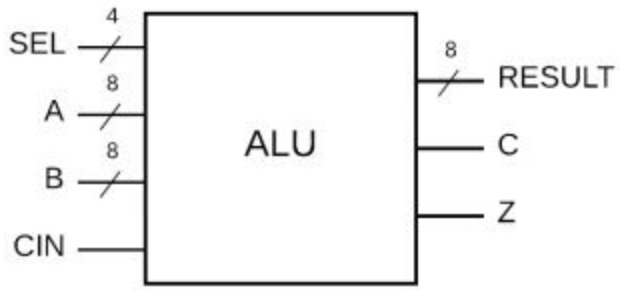
Michael Hegglin & Hunter Borlik

Black Box Block Diagram

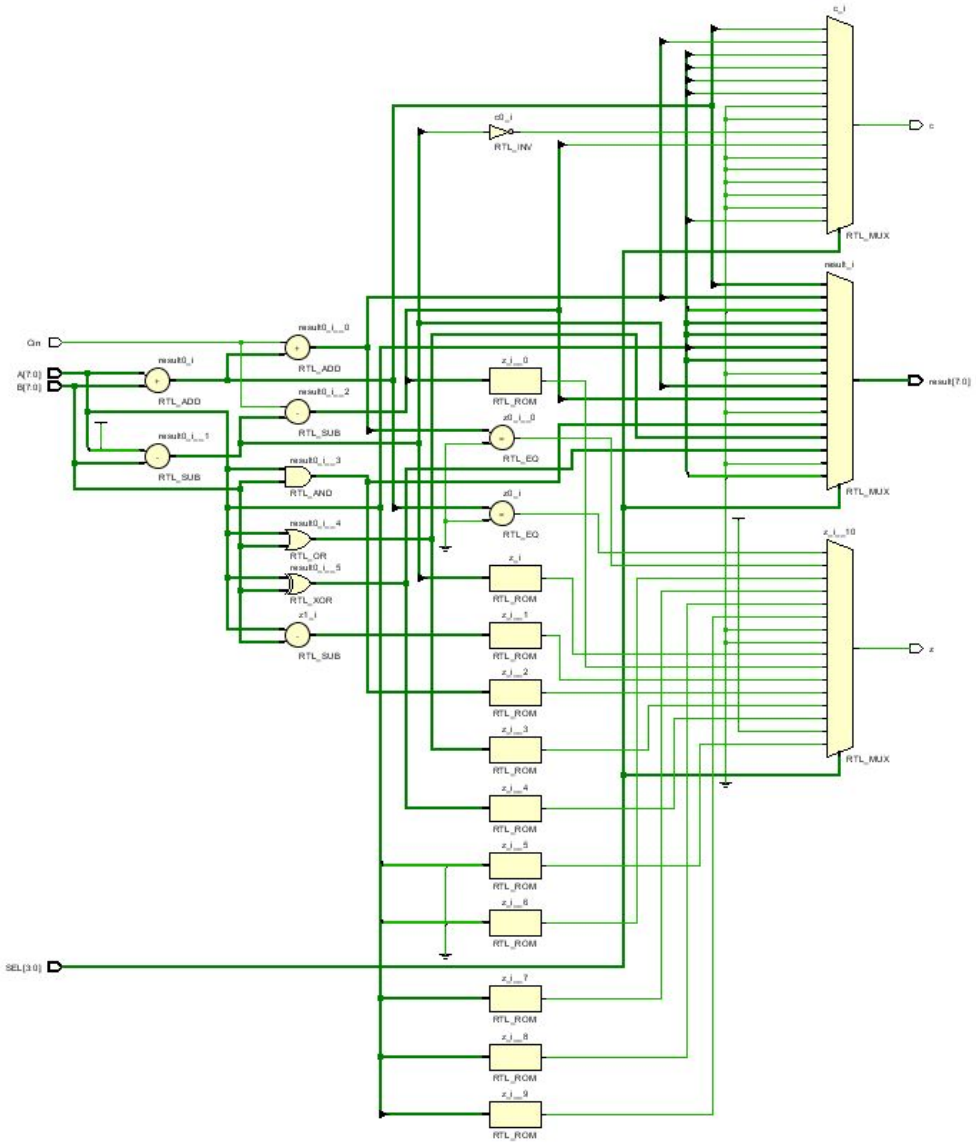


Black Body Diagram

Behavior Description

In this lab assignment our team created an ALU. To accomplish this goal, the 15 RAT arithmetic and logic functions were implemented in system verilog.

Structural Design



Verification

Operation	**A**	**B**	**Cin**	**Result**	**z**	**c**
Add	0x00	0x01	0	0x01	0	0
Addc	0x01	0x03	1	0x05	0	0
Sub	0x02	0x05	0	0xfd	0	1
Subc	0x03	0x07	1	0xfb	0	0
cmp	0x04	0x09	0	0x00	0	0
and	0x05	0x0b	1	0x01	0	0
or	0x06	0x0d	0	0x0f	0	0
xor	0x07	0x0f	1	0x08	0	0
test	0x08	0x11	0	0x00	1	0
lsl	0x09	0x13	1	0x12	0	0
lsr	0x0a	0x15	0	0x05	0	0
ROL	0x0b	0x17	1	0x16	0	0
ROR	0x0c	0x19	0	0x06	0	0
ASR	0x0d	0x1b	1	0x06	0	1
Mov	0x0e	0x1d	0	0x0e	0	0

The test cases above it can be seen that that the zero flag, carry flag, Cin, and result were all used at some point. By thoroughly testing our code with lots of test cases, it ensures that the flags are being set when needed, and the results turn out correct. This ensures the use of test cases. The test cases were done first by hand, then the simulation was run so that they may match.

SystemVerilog Source Code

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 02/05/2019 01:08:59 PM
// Design Name:
// Module Name: realMain
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module realMain(A, B, Cin, SEL, result, c, z);

    input [7:0] A, B;
    input [3:0] SEL;
    input Cin;

    output logic [7:0] result;
    output logic c, z;

    logic [8:0] temp_result;

    always_comb begin

        result = 8'b00000000;
        temp_result = 8'b00000000;
```

```
c = 0;
z = 0;

case(SEL)

    'b0: begin //Add
        temp_result = A + B;
        result = temp_result[7:0];
        c = temp_result[8];

        if (result == 'b0)
            z = 'b1;
    end

    'b0001: begin //Addc
        temp_result = A + B + Cin;
        result = temp_result[7:0];
        c = temp_result[8];

        if (result == 'b0)
            z = 'b1;
    end

    'b0010: begin //Sub
        temp_result = {1'b1, A} - B;
        result = temp_result[7:0];
        c = ~temp_result[8];

        if (result == 'b0)
            z = 'b1;
    end

    'b0011: begin //Subc
        temp_result = {1'b1, A} - B - Cin;
        result = temp_result[7:0];
        c = temp_result[8];

        if (result == 'b0)
            z = 'b1;
    end

    'b0100: begin //compare
        if((A - B) == 'b0)
```

```
        z = 'b1;
    else
        z = 'b0;

end

'b0101: begin //And
    result = A & B;

    if (result == 'b0)
        z = 'b1;
end

'b0110: begin //or
    result = A | B;

    if (result == 'b0)
        z = 'b1;
end

'b0111: begin //exor
    result = A ^ B;

    if (result == 'b0)
        z = 'b1;
end

'b1000: begin //test
    temp_result = A & B;

    if (result == 'b0)
        z = 'b1;
end

'b1001: begin //lsl
    result = {A[6:0], c};
    c = A[7];

    if (result == 'b0)
        z = 'b1;
end

'b1010: begin //lsr
```

```
            result  = {c, A[7:1]};
            c = A[0];

            if (result == 'b0)
                z = 'b1;
        end

        'b1011: begin //rol
            c = A[7];
            result = {A[6:0], A[7]};

            if (result == 'b0)
                z = 'b1;
        end

        'b1100: begin //ror
            c = A[0];
            result = {A[0], A[7:1]};

            if (result == 'b0)
                z = 'b1;
        end

        'b1101: begin //asr
            c = A[0];
            result = {A[7], A[7:1]};

            if (result == 'b0)
                z = 'b1;
        end

        'b1110: begin //mov
            result = A;
        end

        'b1111: begin //unused

        end


    endcase
```

```
    end

endmodule
```