

CSE 4207 CT 4 Assignment

Roll No: 1903178

Assignment Problem: 7-bit ALU operation with NOT and Shift Right operations.

Category: C

Word Size = 7

ALU Operations = NOT and Shift Right

Solution:

Video:

Have you uploaded the video?	YES
Check List 1: Have you explained the design using testbenches to prove that your circuit is working correctly and giving correct results?	YES
Check List 2: Have you shown full synthesis results showing all the required info including RTL Synthesis, RTL Floorplan, RTL Power Analysis, GDS and Heatmap (for details, see assignment template doc)?	YES
NB: Failing to upload video will cause heavy point penalty (5-6 Marks)	
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

HDL Code:

Check List: Have you added all the modules written in verilog including ALU, ALU_OP1, ALU_OP2, CONTROLLER, TOP, TOP_TESTBENCH, ALU_TESTBENCH, CONTROLLER_TESTBENCH?	YES
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

ALU_7bit.v

```
module ALU_7bit (
    input wire [6:0] A, B,
    input wire OP,          // 0: NOT, 1: SHR
    output reg [6:0] result,
    output wire ZF          // Zero flag
);
    wire [6:0] R_NOT, R_SHR;

    // Instantiate modules
    ALU_NOT_7bit NOT1 (.A(A), .result(R_NOT));
    ALU_SHR_7bit SHR1 (.in(A), .shift(B[2:0]), .out(R_SHR));

    always @(*) begin
        case (OP)
            1'b0: result = R_NOT; // NOT operation
            1'b1: result = R_SHR; // SHR operation
            default: result = 7'b0000000;
        endcase
    end

    // Zero flag (1 when result is all zeros)
    assign ZF = (result == 7'b0000000);
endmodule
```

ALU_7bit_tb.v

```
`timescale 1ns/1ns
module ALU_7bit_tb;

    reg [6:0] A, B;
    reg OP;
    wire [6:0] result;
    wire ZF;
    ALU_7bit uut (
        .A(A),
        .B(B),
        .OP(OP),
        .result(result),
        .ZF(ZF)
    );
    initial begin
        $dumpfile("alu_7bit_test.vcd");
        $dumpvars(0, ALU_7bit_tb);

        // Test NOT operation
        A = 7'b1010101; B = 7'b0000000; OP = 1'b0;
        #10;

        // Test SHR operation with different shift amounts
        A = 7'b1100110; B = 7'b0000001; OP = 1'b1; // Shift
by 1
        #10;
        A = 7'b1100110; B = 7'b0000011; OP = 1'b1; // Shift
by 3
        #10;
        A = 7'b1100110; B = 7'b0000111; OP = 1'b1; // Shift
by 7 (all bits out)
        #10;

        // Test zero flag
        A = 7'b0000000; B = 7'b0000000; OP = 1'b0; // NOT of
0 should be all 1s
        #10;
        A = 7'b0000001; B = 7'b0000001; OP = 1'b1; // Shift
1 right by 1 (should be 0)
        #10;

        $finish;
    end
    initial begin
```

```

        $monitor("Time=%3t A=%b B=%b OP=%b -> Result=%b
ZF=%b",
                $time, A, B, OP, result, ZF);
    end
endmodule

```

ALU_NOT_7bit.v

```

module ALU_NOT_7bit (
    input wire [6:0] A,
    output wire [6:0] result
);
    assign result = ~A;
endmodule

```

ALU_SHR_7bit.v

```

                                module ALU_SHR_7bit (
    input wire [6:0] in,
    input wire [2:0] shift, // Shift amount (0-7)
    output reg [6:0] out
);
    always @(*) begin
        case(shift)
            3'b000: out = in;
            3'b001: out = {1'b0, in[6:1]};
            3'b010: out = {2'b00, in[6:2]};
            3'b011: out = {3'b000, in[6:3]};
            3'b100: out = {4'b0000, in[6:4]};
            3'b101: out = {5'b00000, in[6:5]};
            3'b110: out = {6'b000000, in[6]};
            3'b111: out = 7'b0000000; // Shift all bits out
        endcase
    end
endmodule

```

top.v

```
module top (
    input wire clk, reset,
    output wire [6:0] result,
    output wire flag_gt_zero
);
    wire [6:0] A, B;
    wire OP;
    wire [6:0] alu_result;
    wire ZF;

    controller fsm (
        .clk(clk),
        .reset(reset),
        .A(A),
        .B(B),
        .OP(OP)
    );

    ALU_7bit datapath (
        .A(A),
        .B(B),
        .OP(OP),
        .result(alu_result),
        .ZF(ZF)
    );

    assign result = alu_result;
    assign flag_gt_zero = (alu_result != 7'b0); // Direct
comparison
endmodule
```

top_tb.v

```
`timescale 1ns/1ns
module top_tb;

    reg clk, reset;
    wire [6:0] result;
    wire flag_gt_zero;

    top uut (
        .clk(clk),
        .reset(reset),
        .result(result),
        .flag_gt_zero(flag_gt_zero)
    );

    initial begin
        clk = 0;
        forever #5 clk = ~clk;
    end

    initial begin
        $dumpfile("top_test.vcd");
        $dumpvars(0, top_tb);

        reset = 1;
        #10;
        reset = 0;

        #150; // Cover all states
        $finish;
    end

    initial begin
        $monitor("Time=%0t Result=%b Flag=%b",
            $time, result, flag_gt_zero);
    end
endmodule
```

controller.v

```
module controller (  
    input wire clk, reset,  
    output reg [6:0] A, B,  
    output reg OP  
);  
    reg [2:0] pstate, nstate;  
  
    parameter START = 3'b000,  
               ONE  = 3'b001,  
               TWO  = 3'b010,  
               THREE = 3'b011,  
               FINISH = 3'b100;  
  
    // State register  
    always @(posedge clk or posedge reset) begin  
        if (reset) pstate <= START;  
        else pstate <= nstate;  
    end  
  
    // Next state and output logic  
    always @(*) begin  
        A = 7'b0;  
        B = 7'b0;  
        OP = 1'b0;  
        nstate = pstate;  
  
        case (pstate)  
            START: nstate = ONE;  
  
            ONE: begin          // Test NOT operation (result  
non-zero)  
                A = 7'b1010101;  
                OP = 1'b0;    // NOT opcode  
                nstate = TWO;  
            end  
  
            TWO: begin          // Test SHR operation (result  
non-zero)  
                A = 7'b1100110;  
                B = 7'b0000001; // Shift by 1  
                OP = 1'b1;    // SHR opcode  
                nstate = THREE;  
            end  
  
            THREE: begin        // Test SHR operation (force  
result=0)
```

```
        A = 7'b0000001;
        B = 7'b0000111; // Shift by 7 (all bits out)
        OP = 1'b1;
        nstate = FINISH;
    end

    FINISH: nstate = START;

    default: nstate = START;
endcase
end
endmodule
```


controller_tb.v

```
`timescale 1ns/1ns
module controller_tb;

    reg clk, reset;
    wire [6:0] A, B;
    wire OP;

    controller uut (
        .clk(clk),
        .reset(reset),
        .A(A),
        .B(B),
        .OP(OP)
    );

    initial begin
        clk = 0;
        forever #5 clk = ~clk;
    end

    initial begin
        $dumpfile("controller_test.vcd");
        $dumpvars(0, controller_tb);

        reset = 1;
        #10;
        reset = 0;

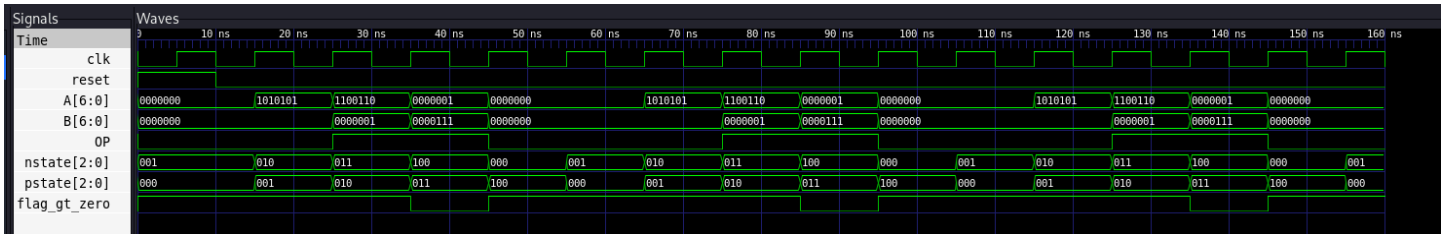
        #100;
        $finish;
    end

    initial begin
        $monitor("Time=%3t State=%b A=%b B=%b OP=%b",
            $time, uut.pstate, A, B, OP);
    end
endmodule
```

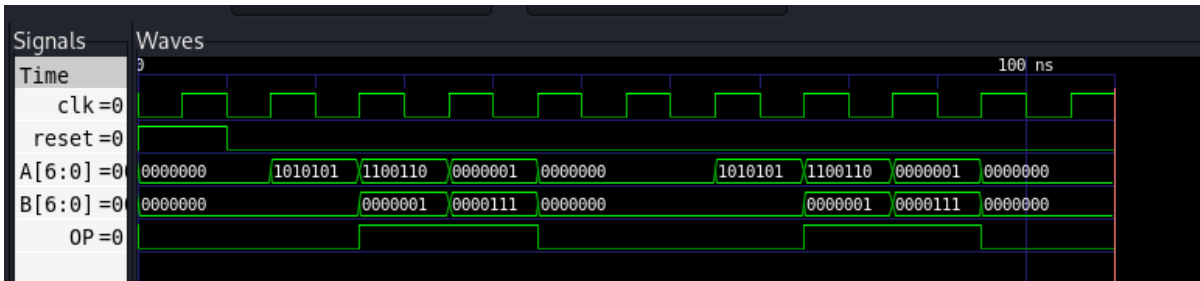
RTL Timing Diagram:

Check List: Have you added all the timing diagrams of ALU_TESTBENCH, CONTROLLER_TESTBENCH, TOP_TESTBENCH?	YES
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

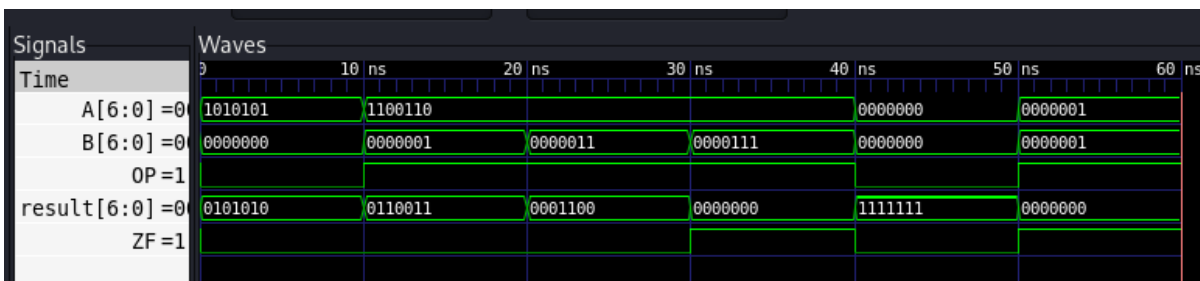
top_tb.v



controller_tb.v



ALU_7bit_tb.v



RTL Synthesis (130nm Skywater PDK with OpenLane toolchain):

Check List: Have you added <i>RTL synthesis summary</i> , <i>RTL synthesized design figure</i> and <i>Standard cell usage in synthesized design</i> ?	YES
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

RTL synthesis summary

RTL synthesis summary (TOP)

```
=== top ===
```

```
Number of wires:           19
Number of wire bits:       25
Number of public wires:    10
Number of public wire bits: 16
Number of ports:           4
Number of port bits:       10
Number of memories:        0
Number of memory bits:     0
Number of processes:       0
Number of cells:           23
```

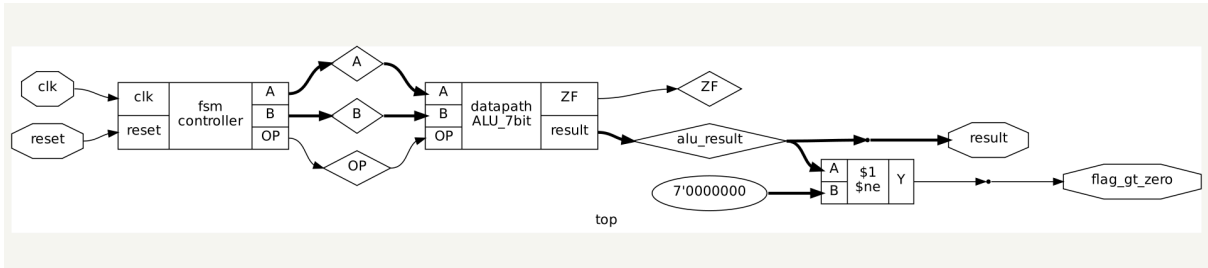
RTL synthesis summary (ALU)

```
=== ALU_7bit ===
```

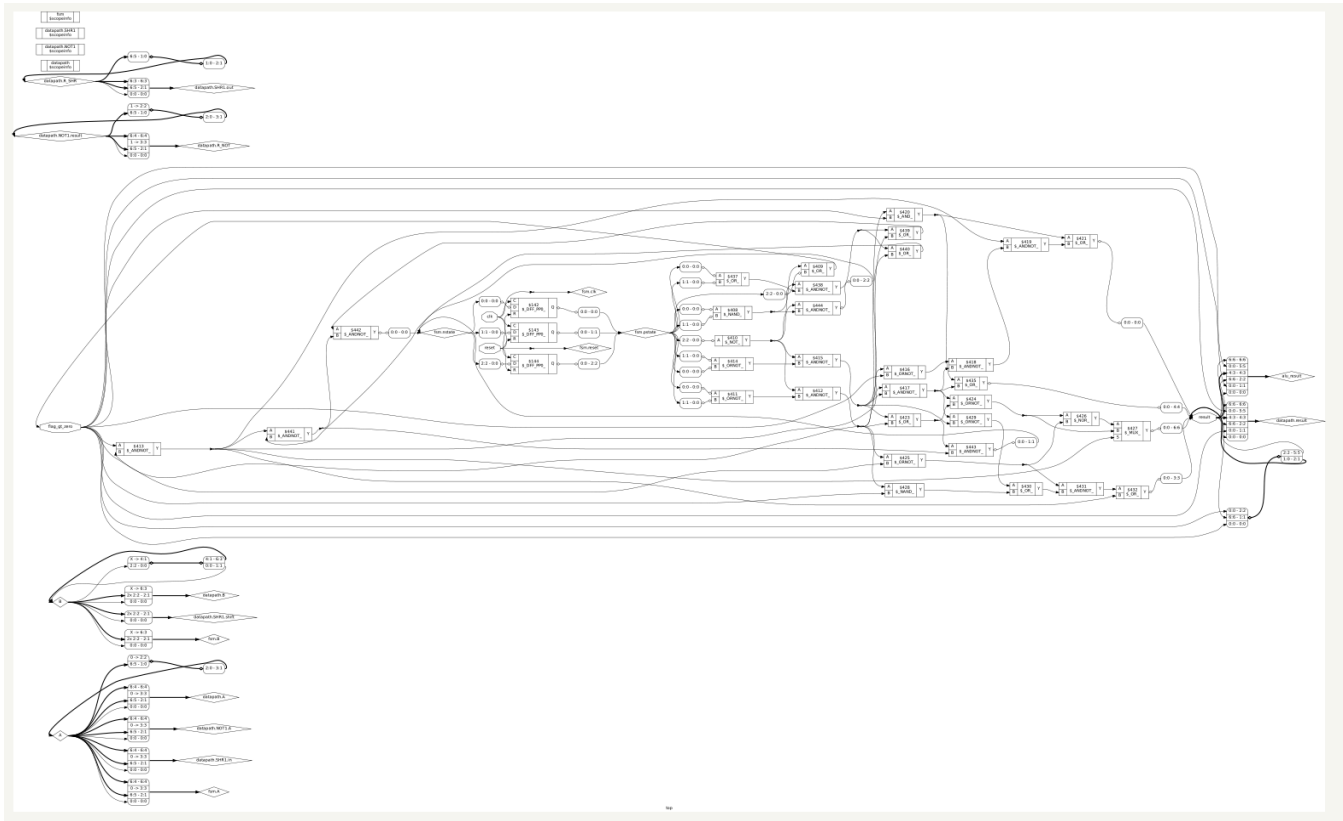
```
Number of wires:           43
Number of wire bits:       61
Number of public wires:     5
Number of public wire bits: 23
Number of ports:           5
Number of port bits:       23
Number of memories:        0
Number of memory bits:     0
Number of processes:       0
Number of cells:           46
```

RTL synthesized design figure:

Hierarchy (top)



Primitive techmap (top)



Standard cell usage in synthesized design

Standard Cells (Top)

sky130_fd_sc_hd__buf_1	2
sky130_fd_sc_hd__buf_2	3
sky130_fd_sc_hd__buf_6	2
sky130_fd_sc_hd__dfirtp_2	3
sky130_fd_sc_hd__inv_2	4
sky130_fd_sc_hd__nand2b_4	1
sky130_fd_sc_hd__nor2_2	3
sky130_fd_sc_hd__o21bai_2	1
sky130_fd_sc_hd__or2_2	1
sky130_fd_sc_hd__or2_4	1
sky130_fd_sc_hd__or2b_2	1
sky130_fd_sc_hd__xnor2_2	1

Chip area for module '\top': 222.713600

of which used for sequential elements: 78.825600 (35.39%)

Standard Cells (ALU)

sky130_fd_sc_hd__a211o_2	3
sky130_fd_sc_hd__a21o_2	1
sky130_fd_sc_hd__a21oi_2	1
sky130_fd_sc_hd__a2bb2o_4	2
sky130_fd_sc_hd__a311o_2	1
sky130_fd_sc_hd__a31o_2	2
sky130_fd_sc_hd__and2_2	2
sky130_fd_sc_hd__and2b_2	4
sky130_fd_sc_hd__and4b_2	1
sky130_fd_sc_hd__inv_2	6
sky130_fd_sc_hd__mux2_2	5
sky130_fd_sc_hd__mux4_2	1
sky130_fd_sc_hd__nor2_2	2
sky130_fd_sc_hd__nor4_4	1
sky130_fd_sc_hd__o2111a_2	1
sky130_fd_sc_hd__o211a_2	3
sky130_fd_sc_hd__o211a_4	1
sky130_fd_sc_hd__o21a_2	2
sky130_fd_sc_hd__o21a_4	1
sky130_fd_sc_hd__o22a_2	1
sky130_fd_sc_hd__or3_2	1
sky130_fd_sc_hd__or3_4	1
sky130_fd_sc_hd__or3b_2	2
sky130_fd_sc_hd__or4_4	1

Chip area for module '\ALU_7bit': 462.944000

of which used for sequential elements: 0.000000 (0.00%)

RTL Floorplan (130nm Skywater PDK with OpenLane toolchain)

Check List: Have you added RTL Floorplan info?	YES
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

RTL Floorplan (ALU): or_metrics_out.json file info is given here

```
{ } or_metrics_out.json ×
reports > alu > { } or_metrics_out.json > ...
1 {
2   "design__die__bbox": "0.0 0.0 50.0 50.0",
3   "design__core__bbox": "5.52 10.88 44.16 38.08",
4   "design__io": 23,
5   "design__die__area": 2500,
6   "design__core__area": 1051.01,
7   "design__instance__count": 46,
8   "design__instance__area": 462.944,
9   "design__instance__count__stdcell": 46,
10  "design__instance__area__stdcell": 462.944,
11  "design__instance__count__macros": 0,
12  "design__instance__area__macros": 0,
13  "design__instance__utilization": 0.440476,
14  "design__instance__utilization__stdcell": 0.440476,
15  "design__instance__count__class:inverter": 6,
16  "design__instance__count__class:multi_input_combinational_cell": 40,
17  "flow__warnings__count": 6,
18  "flow__errors__count": 0
19 }
20
```

RTL Floorplan (top): or_metrics_out.json file info is given here

```
reports > top > { } or_metrics_out.json > ...
1 {
2   "design__die__bbox": "0.0 0.0 50.0 50.0",
3   "design__core__bbox": "5.52 10.88 44.16 38.08",
4   "design__io": 10,
5   "design__die__area": 2500,
6   "design__core__area": 1051.01,
7   "design__instance__count": 23,
8   "design__instance__area": 222.714,
9   "design__instance__count__stdcell": 23,
10  "design__instance__area__stdcell": 222.714,
11  "design__instance__count__macros": 0,
12  "design__instance__area__macros": 0,
13  "design__instance__utilization": 0.211905,
14  "design__instance__utilization__stdcell": 0.211905,
15  "design__instance__count__class:buffer": 7,
16  "design__instance__count__class:inverter": 4,
17  "design__instance__count__class:sequential_cell": 3,
18  "design__instance__count__class:multi_input_combinational_cell": 9,
19  "flow__warnings__count": 2,
20  "flow__errors__count": 0
21 }
22
```


RTL Power Analysis (130nm Skywater PDK with OpenLane toolchain)

Check List: Have you added RTL Power Analysis info?	YES
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

RTL Power Analysis (ALU)

reports > alu > power.rpt

```
1
2 =====
3 | report_power
4 =====
5 ===== nom_tt_025C_1v80 Corner =====
6
7 Group          Internal   Switching   Leakage     Total
8 |          |          |          |          |
9 |          |          |          |          |
10 Sequential    0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0%
11 Combinational 4.102680e-06 6.307530e-06 2.711749e-10 1.041048e-05 100.0%
12 Clock         0.000000e+00 0.000000e+00 1.328418e-10 1.328418e-10 0.0%
13 Macro         0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0%
14 Pad           0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0%
15 -----
16 Total         4.102680e-06 6.307530e-06 4.040168e-10 1.041061e-05 100.0%
17 |          |          |          |          |
18 |          |          |          |          |
```

RTL Power Analysis (top)

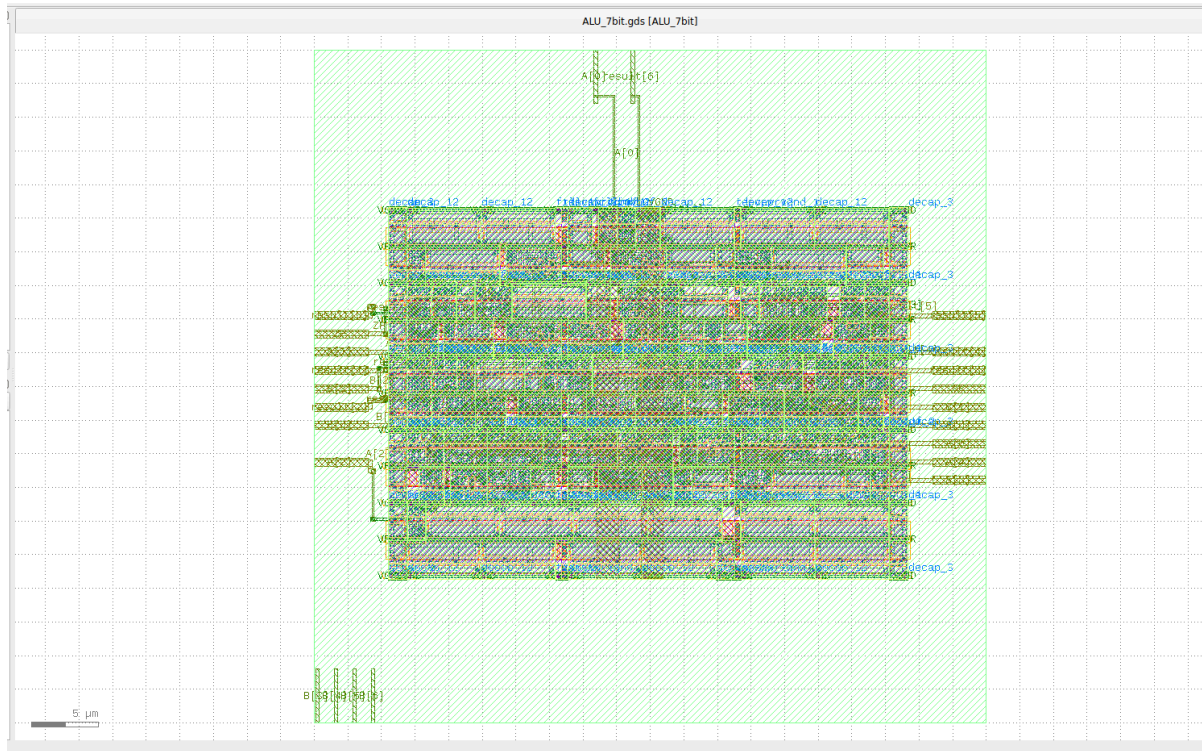
reports > top > power.rpt

```
1
2 =====
3 | report_power
4 =====
5 ===== nom_tt_025C_1v80 Corner =====
6
7 Group          Internal   Switching   Leakage     Total
8 |          |          |          |          |
9 |          |          |          |          |
10 Sequential    4.893699e-06 0.000000e+00 3.315248e-11 4.893733e-06 15.8%
11 Combinational 1.633014e-07 1.893003e-07 1.335659e-10 3.527353e-07 1.1%
12 Clock         2.200258e-05 3.779069e-06 1.841320e-10 2.578183e-05 83.1%
13 Macro         0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0%
14 Pad           0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0%
15 -----
16 Total         2.705958e-05 3.968370e-06 3.508503e-10 3.102830e-05 100.0%
17 |          |          |          |          |
18 |          |          |          |          |
```

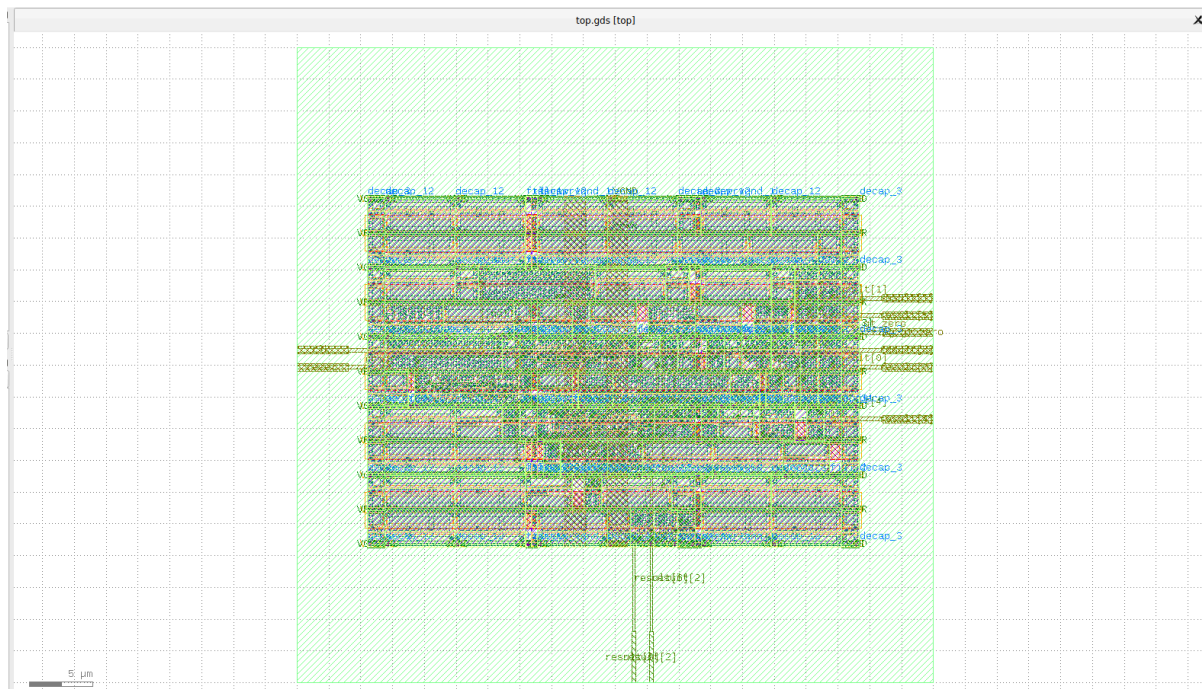
GDS Layout (130nm Skywater PDK with OpenLane toolchain)

Check List: Have you added the GDS Layout figure?	YES
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

GDS Layout (ALU)



GDS Layout (top)

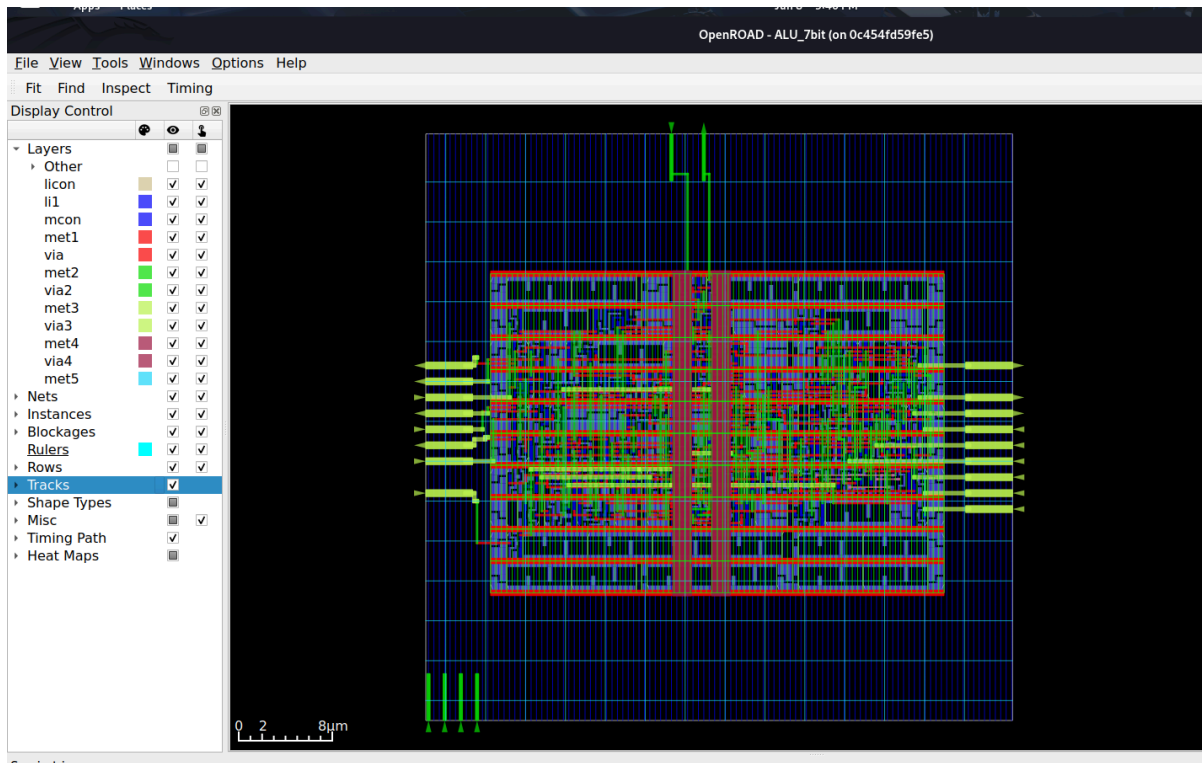


Heatmap (130nm Skywater PDK with OpenLane toolchain)

Check List: Have you added the heatmap?	YES/NO
NB: Failing to add any required info will cause point penalty (1-2 Marks)	

(Following figure is showing what info need to be shown /Just copy paste these figures)

Heatmap (ALU)



Heatmap (top)

