**CSE 4207 CT 4 Assignment**

**Roll No: 1903178**

**Assignment Problem:** 7-bit ALU operation with NOT and Shift Rightoperations.

**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** |
| --- |

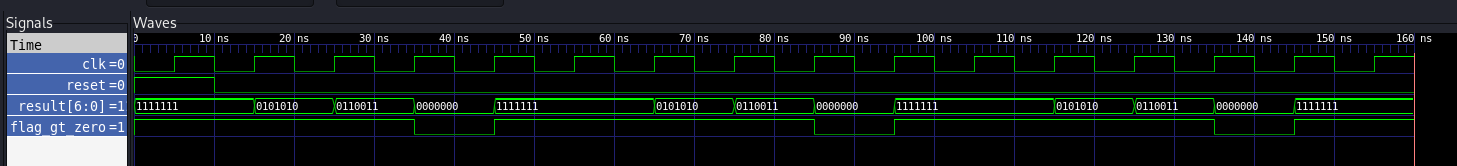
**controller\_tb.v**

| **`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** |
| --- |

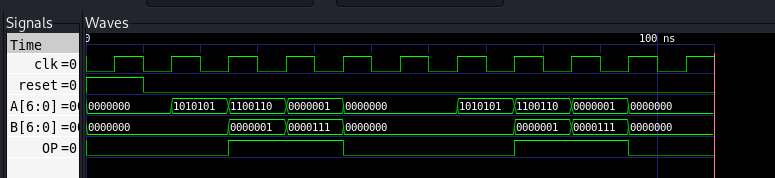
**RTL Timing Diagram:**

| **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)** | |

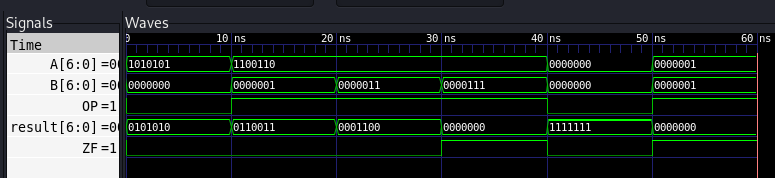
**top\_tb.v**

****

**controller\_tb.v**

****

**ALU\_7bit\_tb.v**

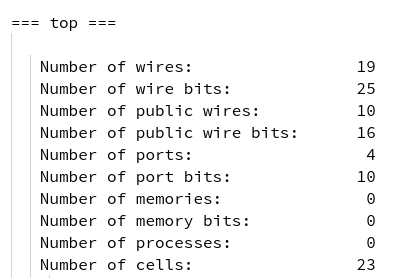


**RTL Synthesis (130nm Skywater PDK with OpenLane toolchain):**

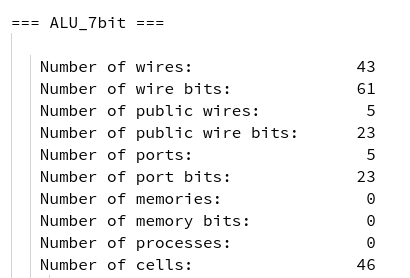
| **Check List:** Have you added *RTL synthesis summary*, *RTL synthesized design figure* and *Standard cell usage in synthesized design*? | YES |
| --- | --- |
| **NB: Failing to add any required info will cause point penalty (1-2 Marks)** | |

RTL synthesis summary

**RTL synthesis summary (TOP)**

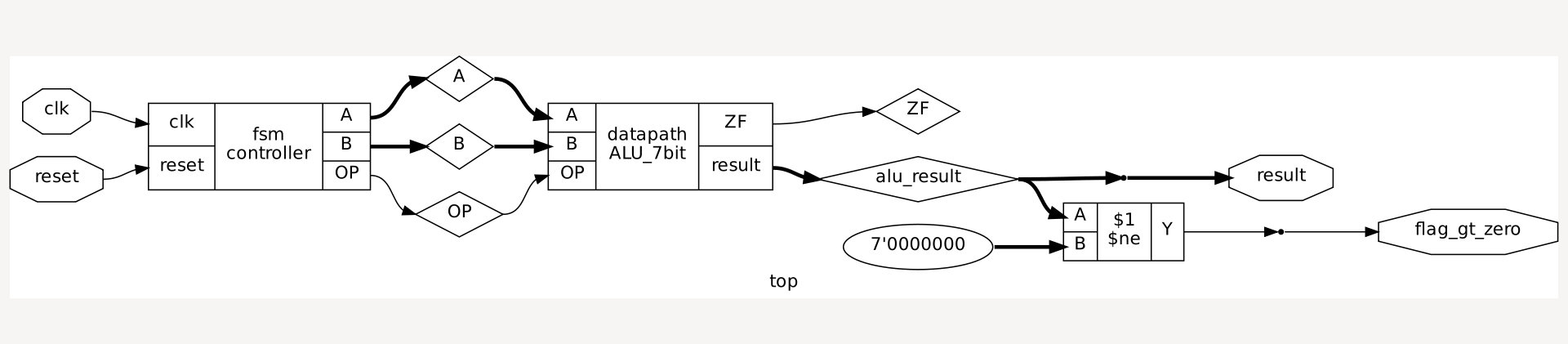


**RTL synthesis summary (ALU)**

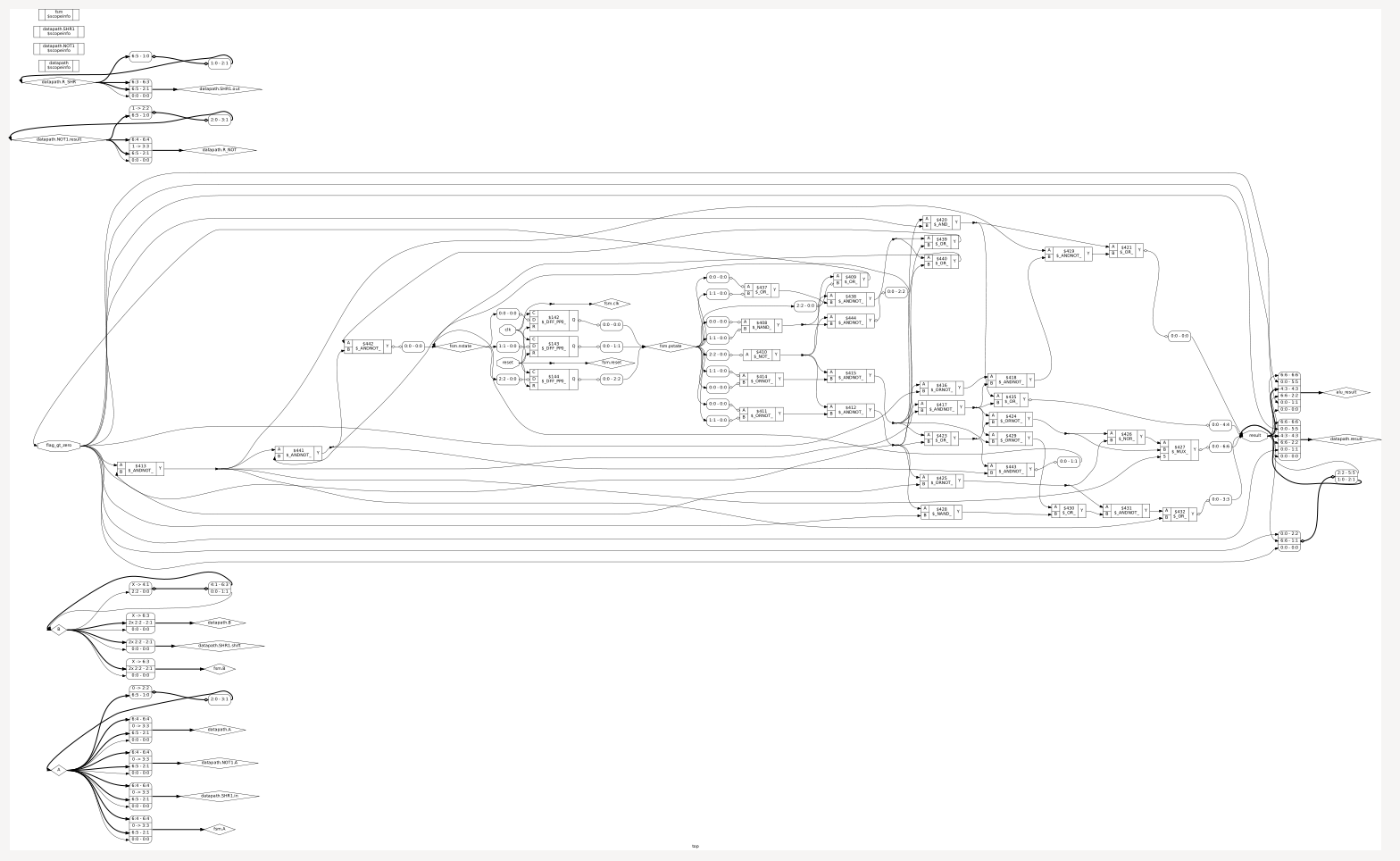


RTL synthesized design figure:

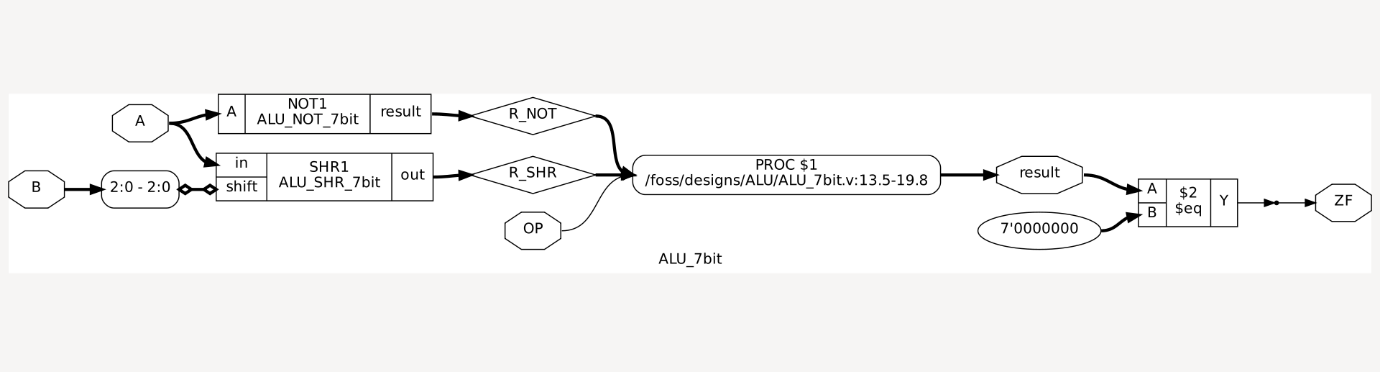
**Hierarchy (top)**



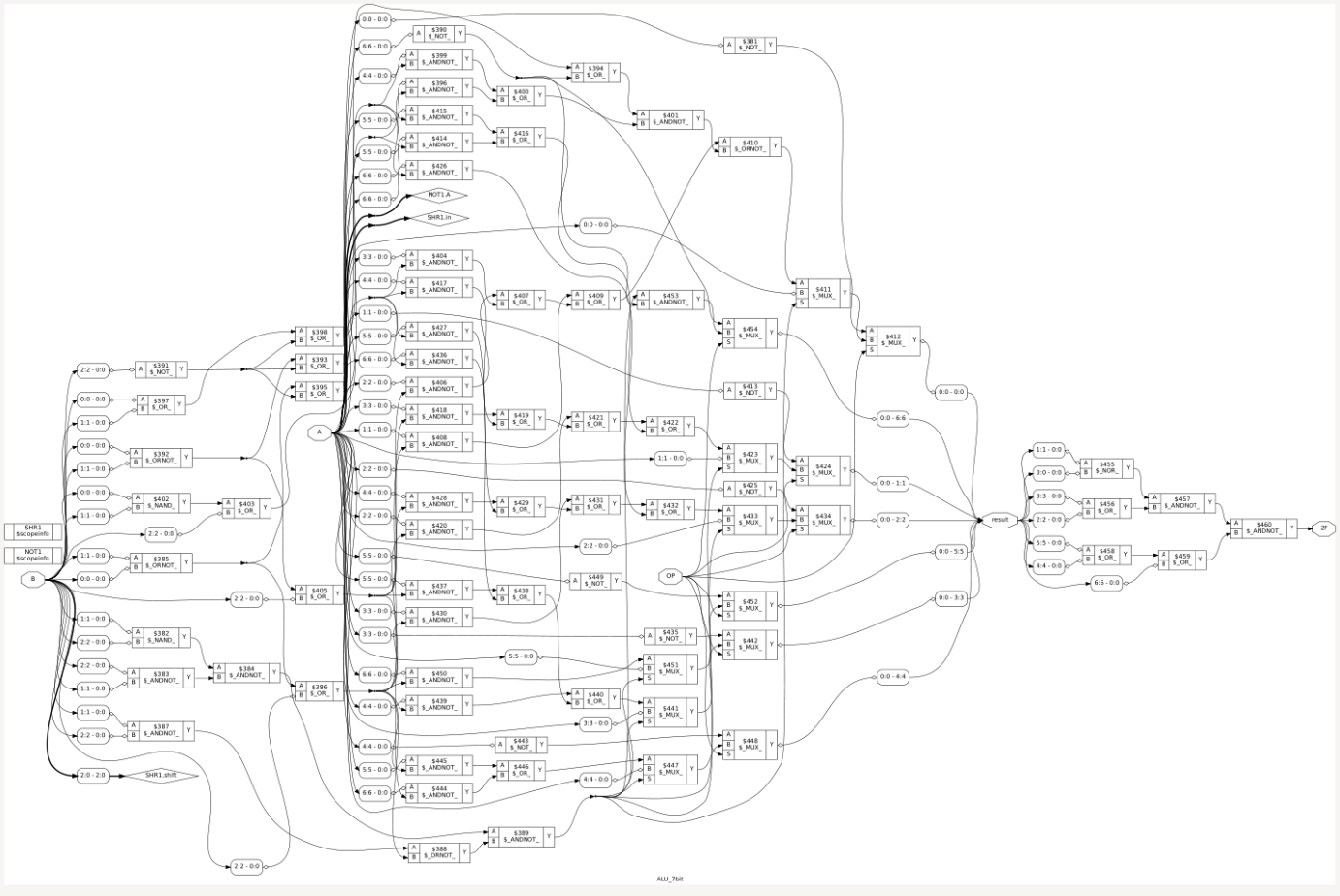
**Primitive techmap (top)**



**Hierarchy (ALU)**

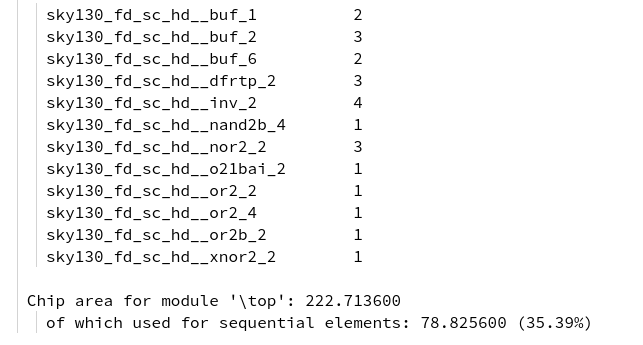


**Primitive techmap (ALU)**

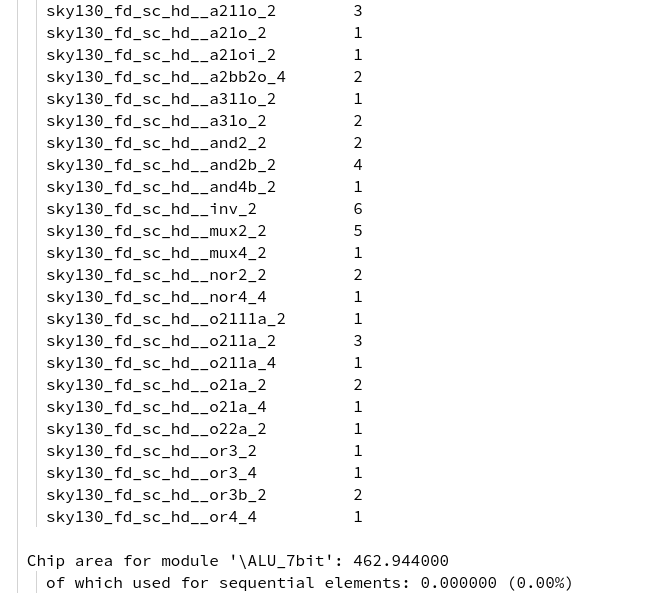


Standard cell usage in synthesized design

**Standard Cells(Top)**



**Standard Cells(ALU)**



**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



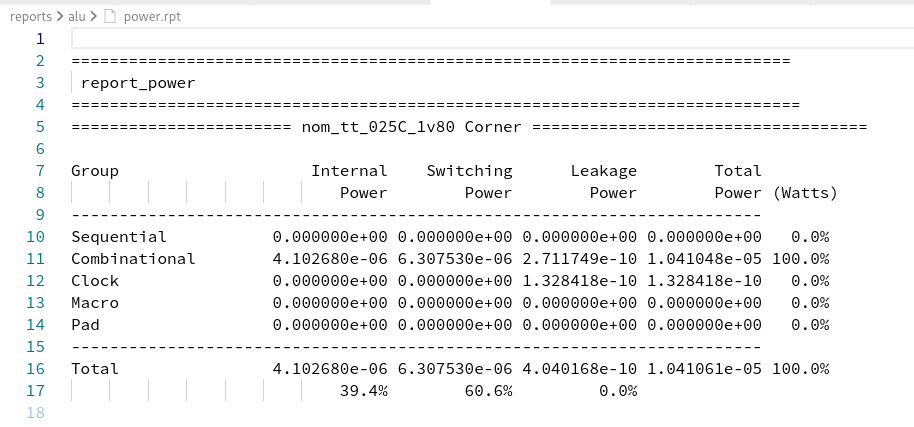
**RTL Floorplan (top):** or\_metrics\_out.json file info is given here

****

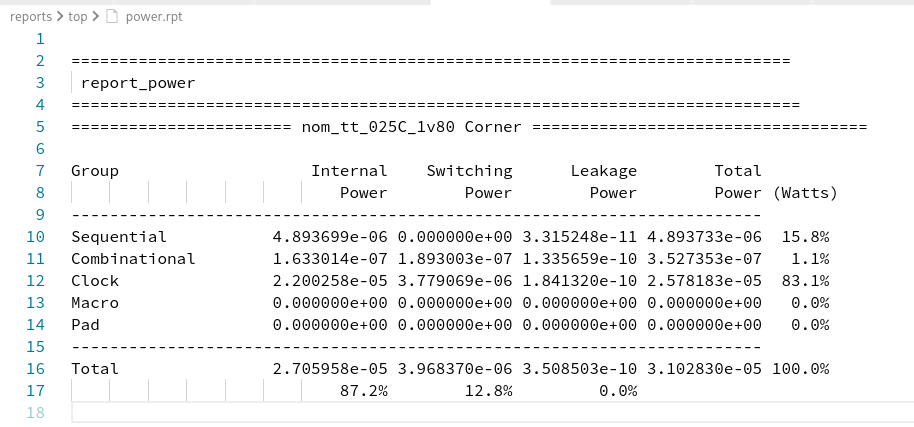
**RTL Power Analysis (130nm Skywater PDK with OpenLane toolchain)**

| **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)**



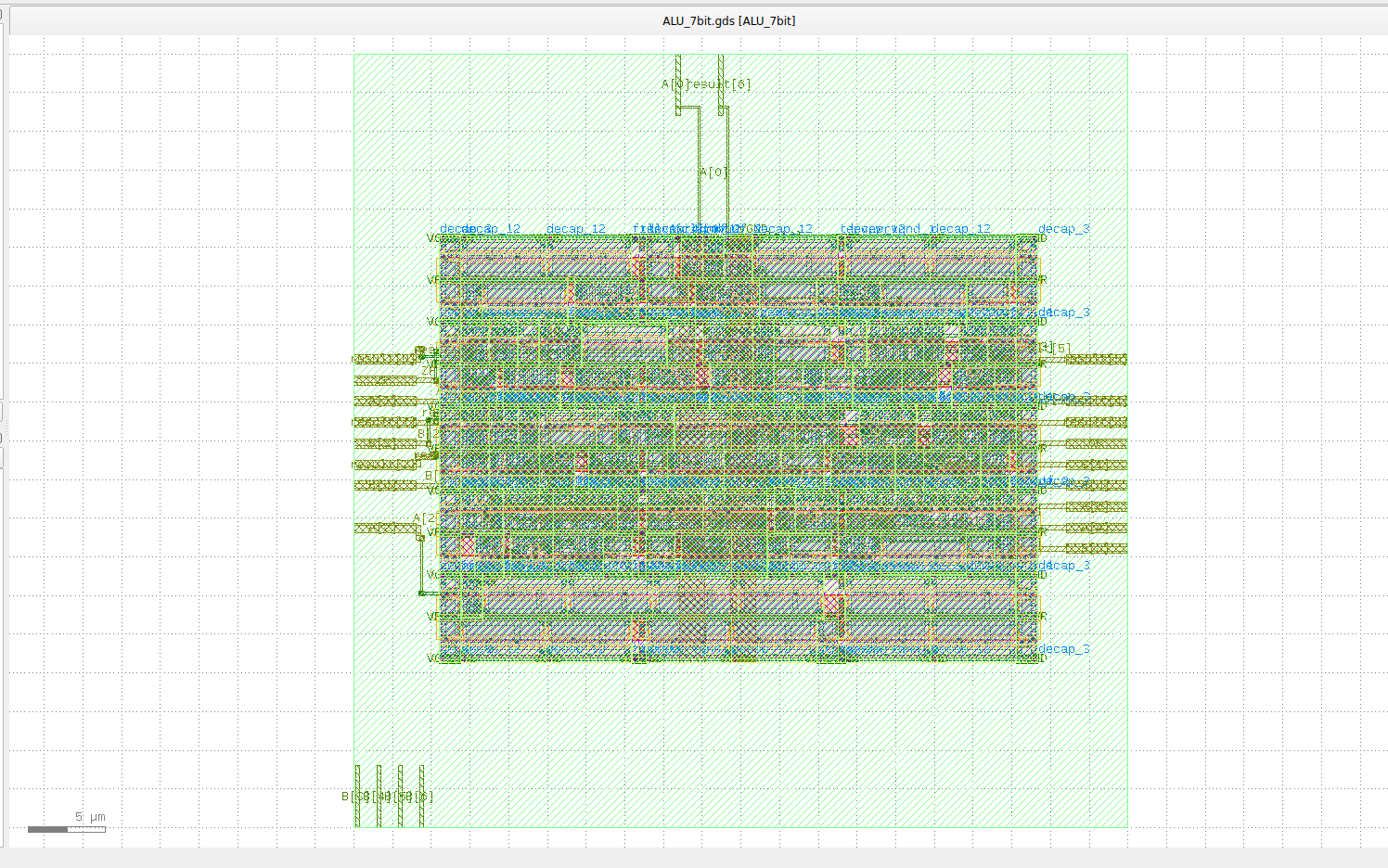
**RTL Power Analysis (top)**



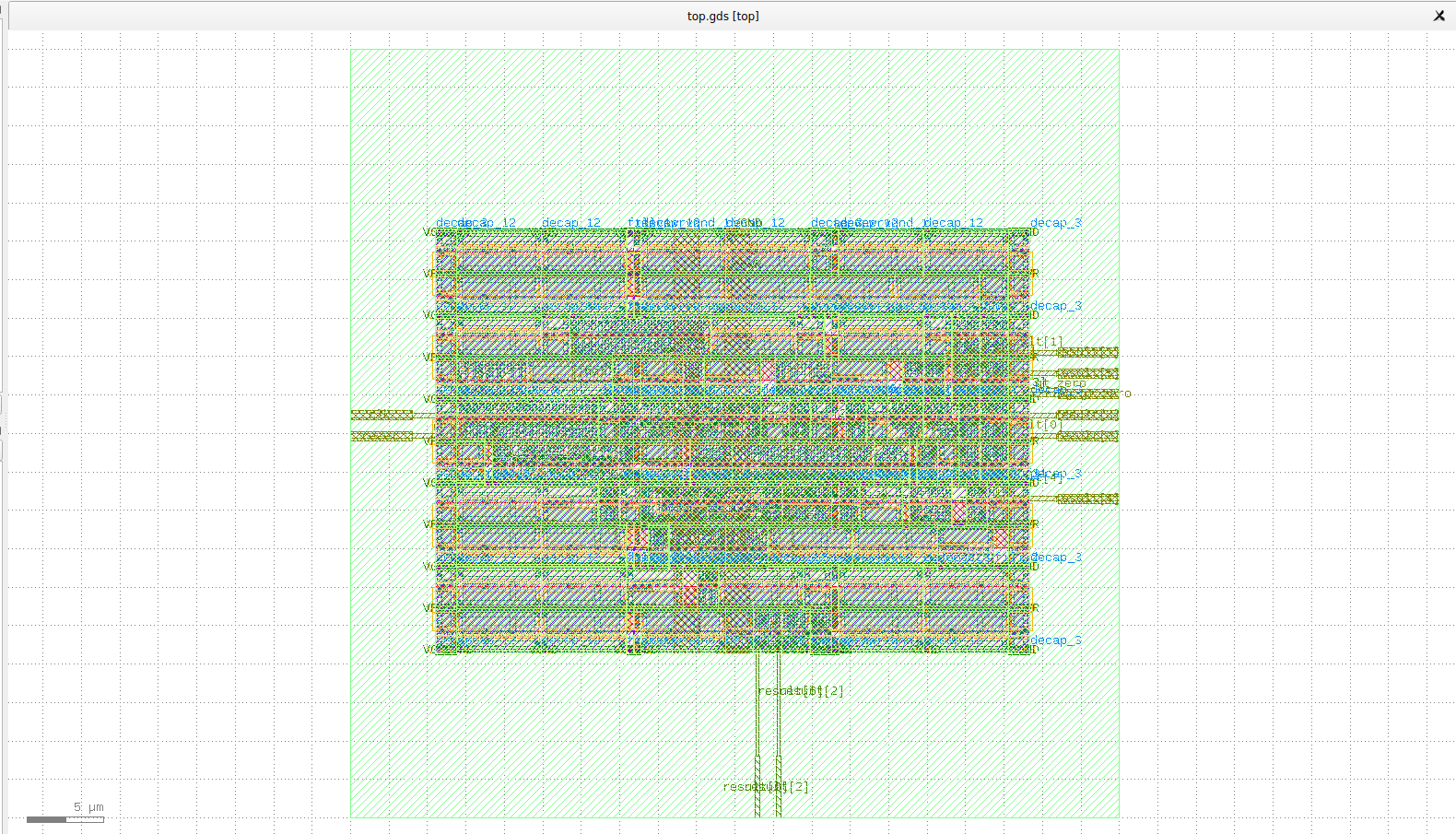
**GDS Layout (130nm Skywater PDK with OpenLane toolchain)**

| **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)**

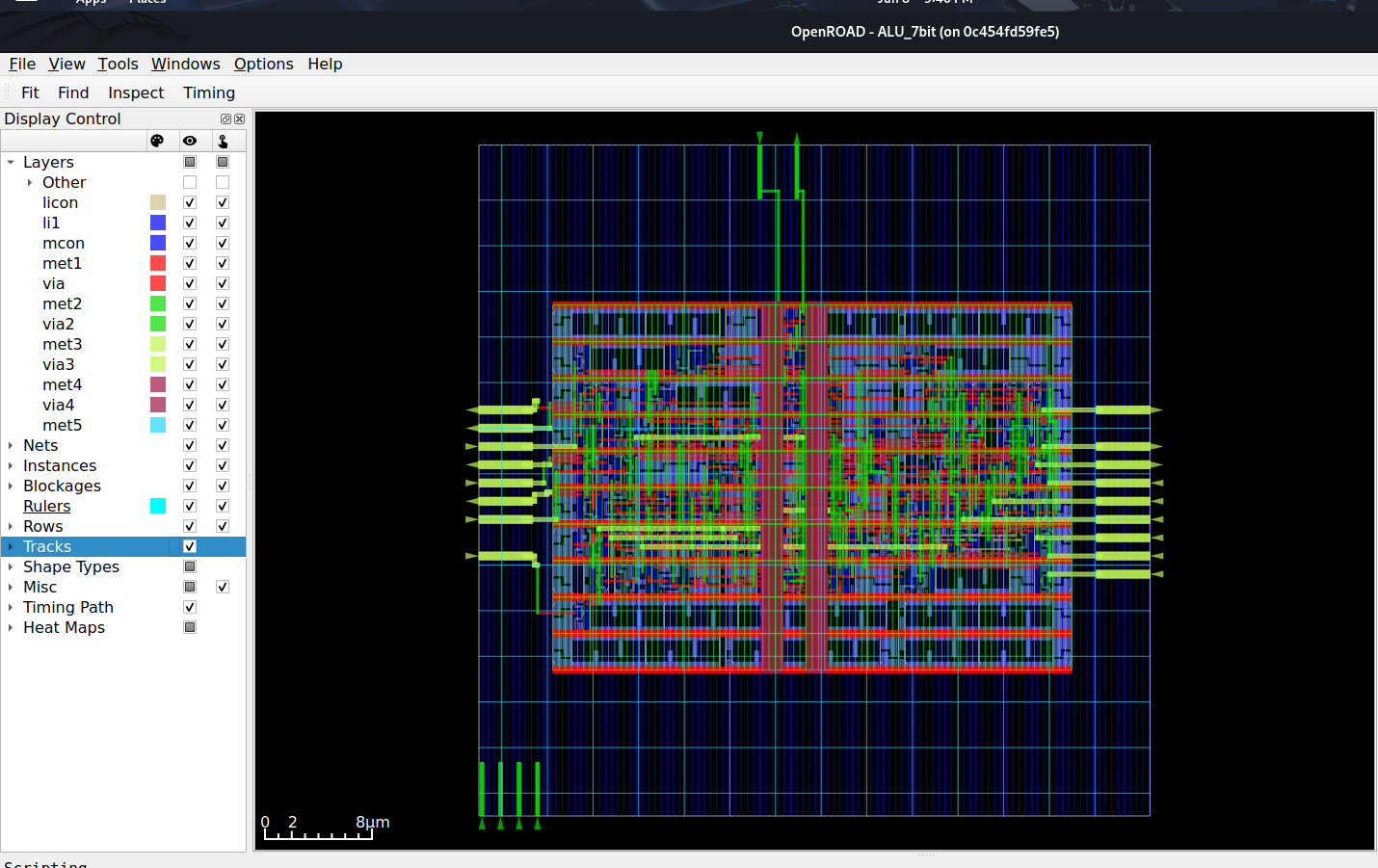


**Heatmap (130nm Skywater PDK with OpenLane toolchain)**

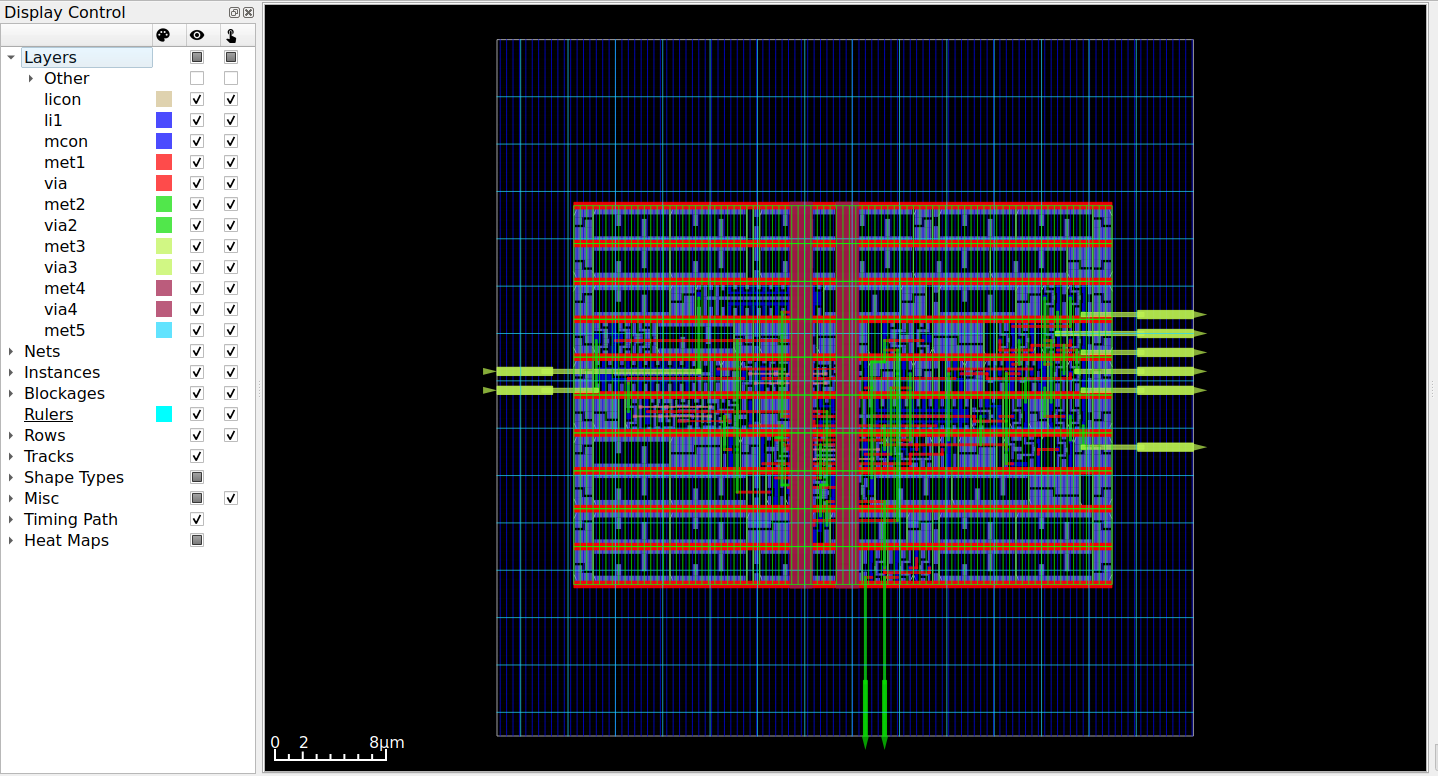
| **Check List:** Have you added the heatmap? | YES/NO |
| --- | --- |
| **NB: Failing to add any required info will cause point penalty (1-2 Marks)** | |

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

**Heatmap (ALU)**



**Heatmap (top)**

****