Assignment Problem: Combinational (ALU OP)circuit using Verilog
Category: A,B,C
Word Size = 6
ALU Operations = XOR AND OR ((A xor B) and (A or B)),SHR

Solution:
[Github repo]

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

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

ALU

```
module ALU (
    input wire [5:0] A, B,          // 6-bit inputs A and B
    input wire OP,                  // 1-bit operation code
    output reg [5:0] R,             // 6-bit result
    output wire CF,                 // Carry flag
    output wire SF,                 // Sign flag
    output wire ZF                  // Zero flag
);

    wire [5:0] R_XANDOR;            // Result from XANDOR module
    wire [5:0] R_SHR;               // Result from SHR module

    // Instantiate modules
    ALU_XOR_AND_OR_6bit XANDOR1 (
        .A(A),
        .B(B),
        .result(R_XANDOR)
```

```
    );

    ALU_SHR_6bit SHR1 (
        .in(A),
        .shift(B[2:0]),
        .out(R_SHR)
    );

    // Operation selection
    always @(*) begin
        case (OP)
            1'b0: begin
                R = R_XANDOR; // XANDOR
            end
            1'b1: begin
                R = R_SHR;    // SHR
            end
            default: begin
                R = 6'b000000; // Default case
            end
        endcase
    end

    // Flag assignments
    assign CF = 1'b0;                    // No carry for XANDOR or SHR
    assign SF = R[5];                    // Sign flag: MSB of result
    assign ZF = ~(R[5] | R[4] | R[3] | R[2] | R[1] | R[0]); // Zero
flag: NOR of result bits

endmodule
```

ALU_OP1

```
module ALU_XOR_AND_OR_6bit(
    input wire [5:0]A,    //6 bit input A
    input wire [5:0]B,    //6 bit input B
    output wire [5:0]result//6 bit result
);
    assign result= (A^B) & (A|B);
endmodule
```

ALU_OP2

```
module ALU_SHR_6bit (
  input wire [5:0] in,      // 6-bit input
  input wire [2:0] shift,   // 3-bit input shift
  output reg [5:0] out // 6-bit output
);

  always @(*) begin
     case (shift)
        3'b000: out = in;                    // No shift
        3'b001: out = {1'b0, in[5:1]};      // Right shift by 1 bit
        3'b010: out = {2'b00, in[5:2]};     // Right shift by 2 bits
        3'b011: out = {3'b000, in[5:3]};    // Right shift by 3 bits
        3'b100: out = {4'b0000, in[5:4]};   // Right shift by 4 bits
        3'b101: out = {5'b00000, in[5]};    // Right shift by 5 bits
        3'b110: out = 6'b000000;            // Right shift by 6 bits
(all zeros)
        3'b111: out = 6'b000000;            // Right shift by 7 bits
(all zeros)
     endcase
  end

endmodule
```

TOP

```
module top
(
   input wire clk,reset,
   output wire [5:0]result,
   output wire flag_zero
);

wire [5:0]A,B;
wire OP;
wire [5:0] R_result;

controller controller1(
   .clk(clk),
   .reset(reset),
   .A(A),
   .B(B),
   .OP(OP)
);

ALU datapath1(
   .A(A),
   .B(B),
   .OP(OP),
   .result(R_result),
   .ZF(R_ZF)
);
assign result=R_result;
assign flag_zero=~R_ZF;

endmodule
```

CONTROLLER

```
module controller(
      input wire clk,reset,
      output reg [5:0]A,B,
      output reg OP
);
reg [2:0]pstate,nstate;
parameter[2:0]START= 3'b000, //state
                  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 Outputs
always @(*) begin
      A=6'b000000;
      B=6'b000000;
      OP=1'b0;
      nstate=pstate;

      case (pstate)
      START:
      begin
            nstate = ONE;
            end
      ONE:
      begin


            end
      TWO:
      begin

            end
```

```
      THREE:
      begin
            nstate = FINISH;
            end
      FINISH:
      begin
            nstate = FINISH;
            end
      default
      : begin
            nstate = START;
            end
      endcase
end
// Output logic
      always @(*) begin
      case (pstate)
            ONE: begin
                  A = 6'b101010; // Example values
                  B = 6'b010101;
                  OP = 1'b0;  //XOR AND OR
                  nstate = TWO;
            end
            TWO: begin
                  A = 6'b111100;
                  B = 6'b000011;
                  OP = 1'b0;   //XOR AND OR
                  nstate = THREE;
            end
            THREE: begin
                  A = 6'b100001; // SHR
                  B = 6'b000000;
                  OP = 1'b1;  //
                  nstate=FINISH;
            end
            FINISH: begin
                  nstate=START;
            end
            default:nstate=START;
      endcase
      end
endmodule
```

Top Testbench

```
`timescale 1ns/1ns
module top_tb;
reg clk,reset;
wire [5:0]result;
wire flag_zero;
top uut(
   .clk(clk),
   .reset(reset),
   .result(result),
   .flag_zero(flag_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;    #15
   reset=1;    #20
   reset=0;    #100
   $finish
end
endmodule
```

Controller_Testbench

```
`timescale 1ns/1ns
module controller_tb;
reg clk, reset;
wire [5:0] 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

endmodule
```

ALU_Testbench

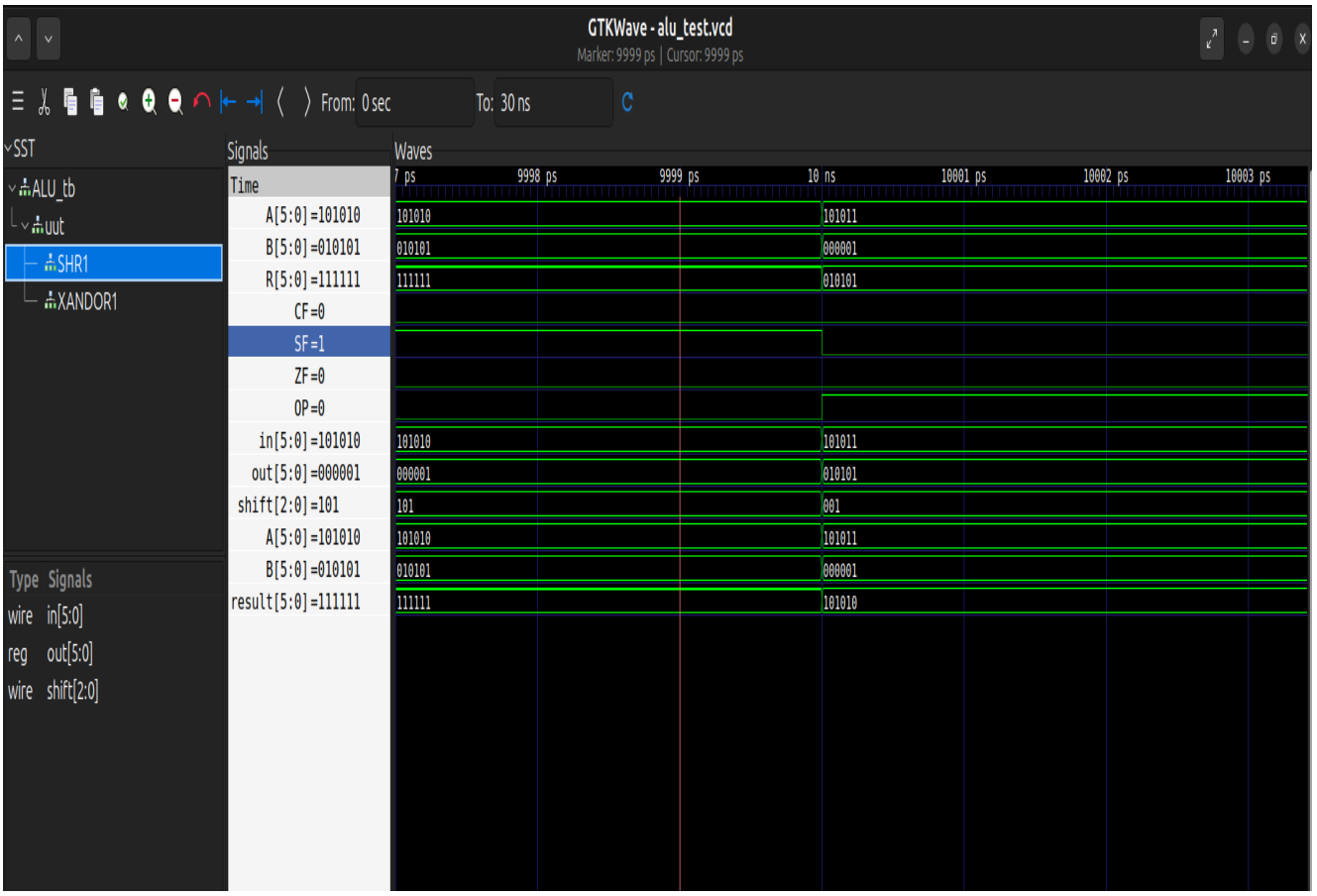
```
`timescale 1ns / 1ps

module ALU_tb;

  // Signals for ALU
  reg [5:0] A, B;             // ALU inputs
  reg OP;                     // ALU operation code (1-bit)
  wire [5:0] R;               // ALU output
  wire CF;                    // Carry flag
  wire SF;                    // Sign flag
  wire ZF;                    // Zero flag
```

```
  // Instantiate ALU module
  ALU uut (
      .A(A),
      .B(B),
      .OP(OP),
      .R(R),
      .CF(CF),
      .SF(SF),
      .ZF(ZF)
  );

  // Dump file setup and test procedure
  initial begin
      $dumpfile("alu_test.vcd"); // Output VCD file
      $dumpvars(0, ALU_tb);    // Dump all variables in the testbench

      // Initialize inputs
      A = 6'b000000;
      B = 6'b000000;
      OP = 1'b0;

      // Test XANDOR operation (OP = 0)
      $display("\nTesting XOR AND OR Operation ((A XOR B) AND (A OR
B)):");
      A = 6'b101010; B = 6'b010101; OP = 1'b0; // Test case 1
      #10 $display("XOR AND OR: A=%b, B=%b, R=%b, SF=%b, ZF=%b", A, B,
R, SF, ZF);

      // Test SHR operation (OP = 1)
      $display("\nTesting SHR Operation (Right Shift):");
      A = 6'b101011; B = 3'b001; OP = 1'b1; // Shift by 1
      #10 $display("SHR by 1: A=%b, shift=%d, R=%b, SF=%b, ZF=%b", A,
B[2:0], R, SF, ZF);

      // Finish simulation
      #10 $finish;
  end

endmodule
```

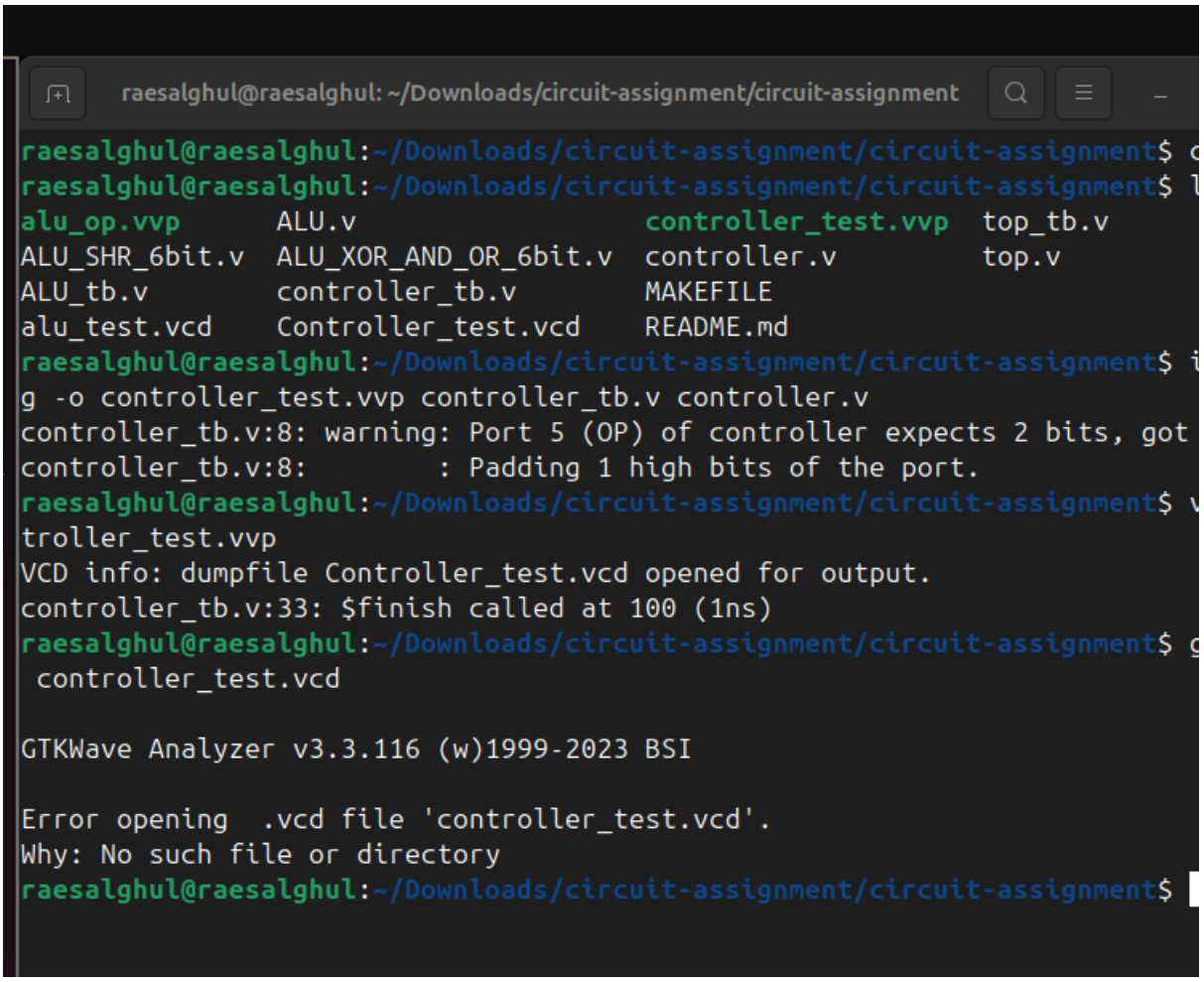
RTL Timing Diagram:

Check List: Have you added all the timing diagrams of ALU_TESTBENCH, CONTROLLER_TESTBENCH, TOP_TESTBENCH?	NO Only ALU_TEST BENCH

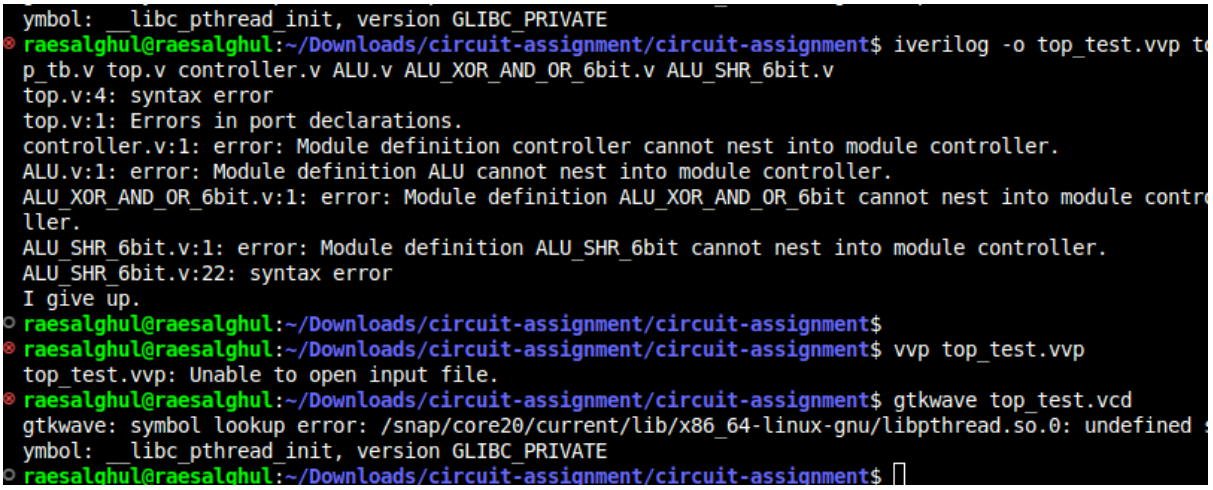
Alu_tb.v



Controller



Top



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

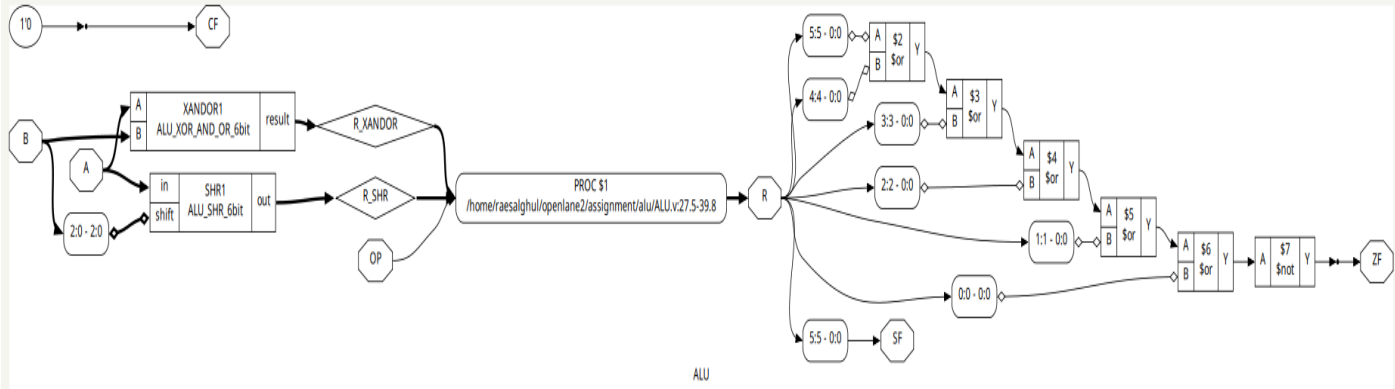
Check List: Have you added *RTL synthesis summary*, *RTL synthesized design figure* and *Standard cell usage in synthesized design*?	YES

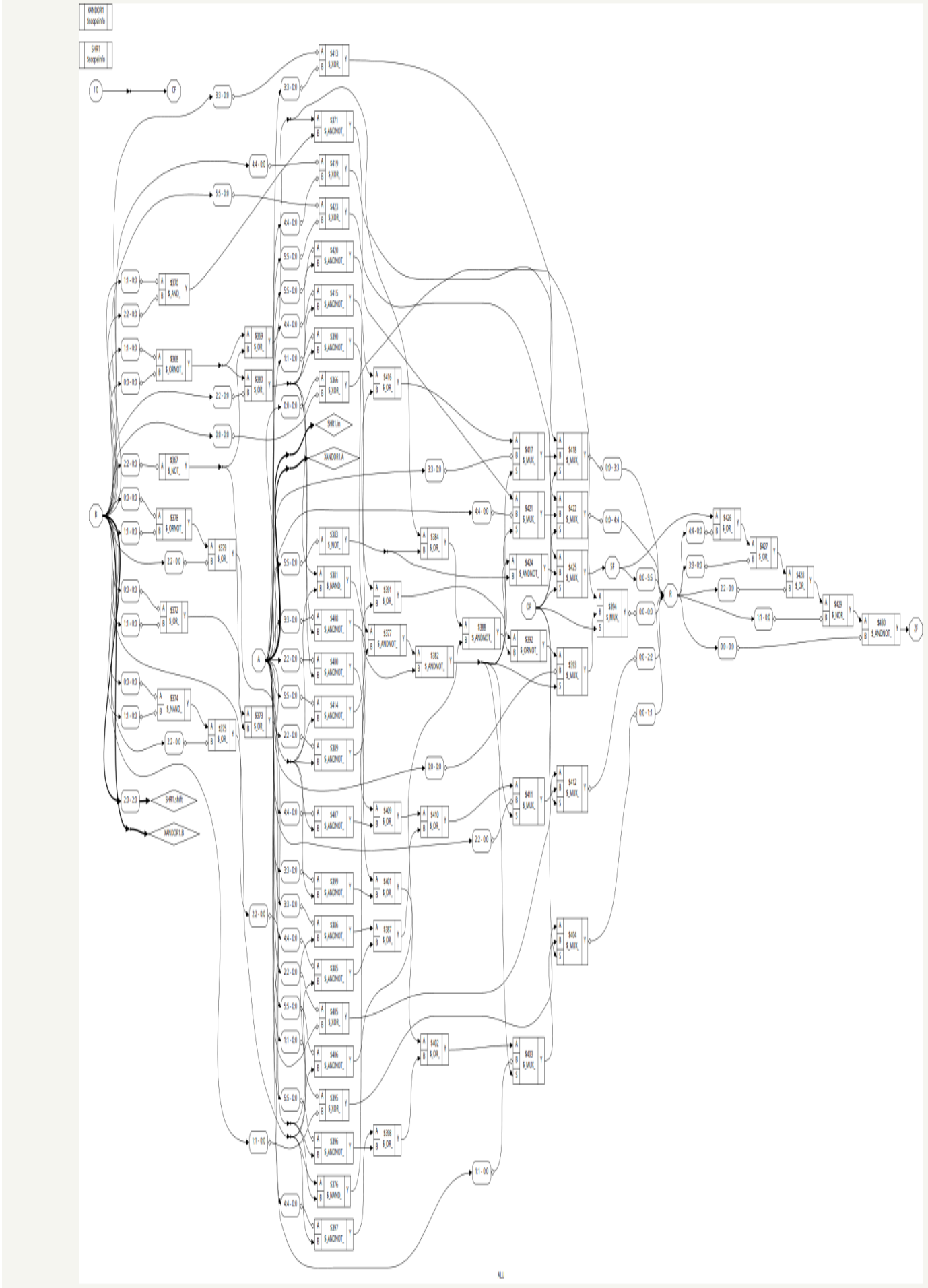
RTL synthesis summary

```
110. Printing statistics.

=== ALU ===

   Number of wires:                 48
   Number of wire bits:             63
   Number of public wires:           7
   Number of public wire bits:      22
   Number of ports:                  7
   Number of port bits:             22
   Number of memories:               0
   Number of memory bits:            0
   Number of processes:              0
   Number of cells:                 50
```

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

There will be 2 figures in 2 .dot files in the synthesis folder. Show them both.




XANDOR1
$xcopointo
SHR1
$xcopointo
YO
CF
B
A
SHR1.shift
XANDOR1.B
SHR1.in
XANDOR1.A
SF
OF
ALU

Standard cell usage in synthesized design **(Following table is showing what info need to be shown /Just copy paste these info from terminal here)**

```
15      sky130_fd_sc_hd__a21bo_2         2
16      sky130_fd_sc_hd__a22oi_2         1
17      sky130_fd_sc_hd__a31o_2          2
18      sky130_fd_sc_hd__a32o_2          3
19      sky130_fd_sc_hd__a41o_2          1
20      sky130_fd_sc_hd__and2b_2         2
21      sky130_fd_sc_hd__and3_2          2
22      sky130_fd_sc_hd__and4_2          1
23      sky130_fd_sc_hd__and4b_2         1
24      sky130_fd_sc_hd__and4bb_2        2
25      sky130_fd_sc_hd__buf_2           1
26      sky130_fd_sc_hd__conb_1          1
27      sky130_fd_sc_hd__inv_2           4
28      sky130_fd_sc_hd__mux2_1          1
29      sky130_fd_sc_hd__nand2_2         2
30      sky130_fd_sc_hd__nor2_2          1
31      sky130_fd_sc_hd__nor3_2          2
32      sky130_fd_sc_hd__nor4_2          1
33      sky130_fd_sc_hd__o21ai_2         2
34      sky130_fd_sc_hd__o21ba_2         3
35      sky130_fd_sc_hd__o22a_2          3
36      sky130_fd_sc_hd__o31a_2          1
37      sky130_fd_sc_hd__or2_2           4
38      sky130_fd_sc_hd__or3_2           4
39      sky130_fd_sc_hd__xor2_2          3
40
```

RTL Floorplan (130nm Skywater PDK with OpenLane toolchain)

Check List: Have you added RTL Floorplan info?	YES

(Following table is showing what info need to be shown /Just copy paste these info from terminal here)

```
[INFO] Extracting DIE_AREA and CORE_AREA from the floorplan
[INFO] Floorplanned on a die area of 0.0 0.0 50.0 50.0 (µm).
[INFO] Floorplanned on a core area of 5.52 10.88 44.16 38.08 (µm).
Writing metric design__die__bbox: 0.0 0.0 50.0 50.0
Writing metric design__core__bbox: 5.52 10.88 44.16 38.08
Setting global connections for newly added cells…
[INFO] Setting global connections...
Updating metrics…
```

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

Check List: Have you added RTL Power Analysis info?	YES

(Following table is showing what info need to be shown /Just copy paste these info from terminal here)

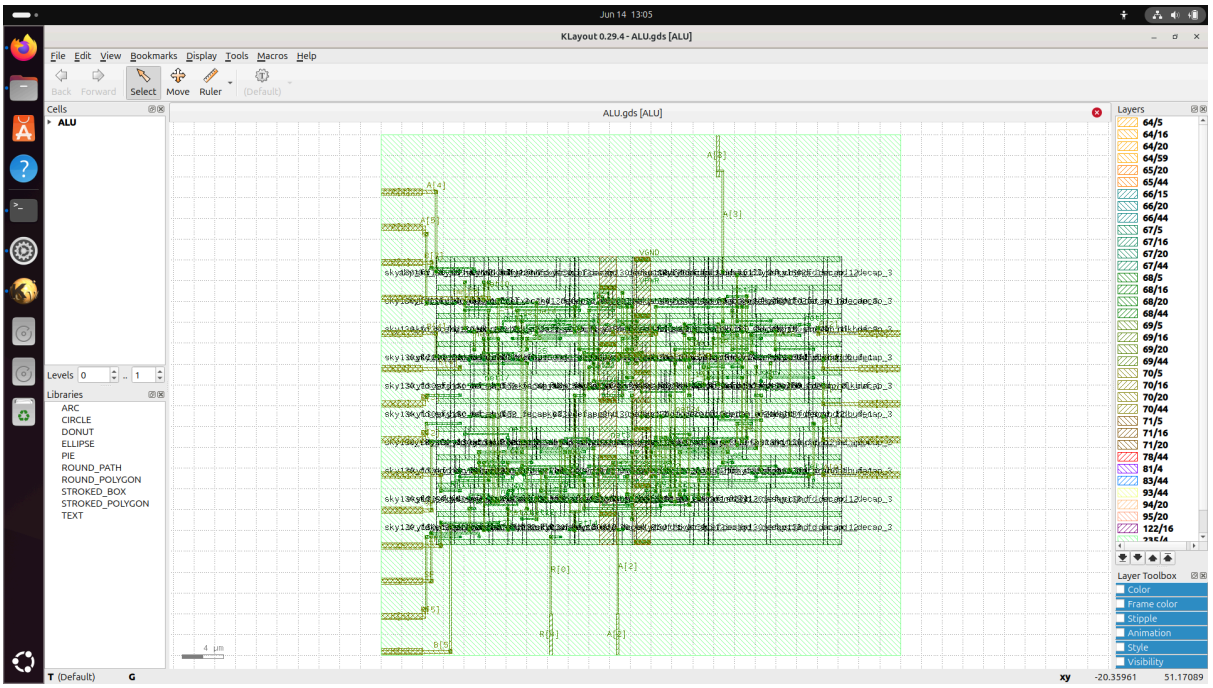
Open power.rpt ~/openlane2/assignment/alu/runs/RUN_2025-06-1...47-11/54-openroad-stapostpnr... Save

```
1
2 ===========================================================================
3  report_power
4 ===========================================================================
5 ======================== nom_tt_025C_1v80 Corner ===================================
6
7 Group                    Internal    Switching      Leakage        Total
8                             Power        Power        Power        Power (Watts)
9 ----------------------------------------------------------------------------
10 Sequential           0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00   0.0%
11 Combinational        1.080277e-05 1.700216e-05 2.737968e-10 2.780520e-05 100.0%
12 Clock                0.000000e+00 0.000000e+00 1.522820e-10 1.522820e-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                1.080277e-05 1.700216e-05 4.260789e-10 2.780535e-05 100.0%
17                             38.9%        61.1%         0.0%
18
```

GDS Layout (130nm Skywater PDK with OpenLane toolchain)

Check List: Have you added the GDS Layout figure?	YES/NO

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



Heatmap (130nm Skywater PDK with OpenLane toolchain)

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