**CSE 4207 CT 4 Assignment**

**Roll No: 1903048**

**Assignment Problem:**

**Category:** A

**Word Size = 7bits**

**ALU Operations = NOT ,ROR**

**HDL Code:**

|  |  |
| --- | --- |
| **Check List:** Have you added all the modules written in verilog including ALU, ALU\_OP1, ALU\_OP2, ALU\_TESTBENCH? | YES |

**ALU\_OP1.v**

|  |
| --- |
| **module ALU\_OP1**  **(**  **input wire [6:0] A,**  **output wire [6:0] R**  **);**  **assign R = ~A;**  **endmodule** |

**ALU\_OP2.v**

|  |
| --- |
| **module ALU\_OP2**  **(**  **input wire [6:0] A,**  **input wire [2:0] B, // Number of bits to rotate (0–7)**  **output reg [6:0] R**  **);**  **always @(\*) begin**  **case (B)**  **3'd0: R = A; // No rotation**  **3'd1: R = {A[0], A[6:1]};**  **3'd2: R = {A[1:0], A[6:2]};**  **3'd3: R = {A[2:0], A[6:3]};**  **3'd4: R = {A[3:0], A[6:4]};**  **3'd5: R = {A[4:0], A[6:5]};**  **3'd6: R = {A[5:0], A[6]};**  **3'd7: R = A; // Full rotate results in original**  **default: R = A;**  **endcase**  **end**  **endmodule** |

**ALU.v**

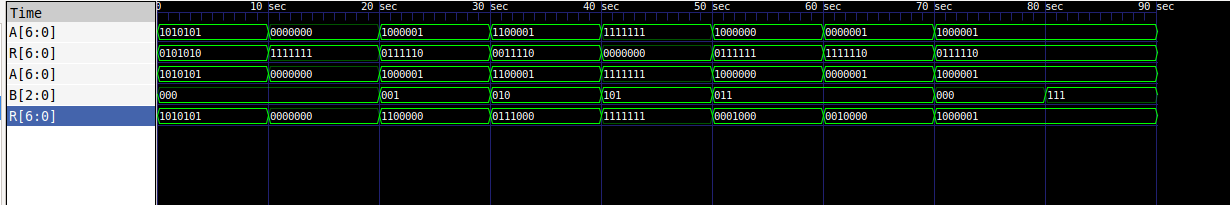
|  |
| --- |
| **module ALU**  **(**  **input wire [6:0] A, B,**  **input wire [1:0] OP, // 00 = NOT, 01 = ROR**  **output reg [6:0] R,**  **output wire CF,**  **output wire SF,**  **output wire ZF**  **);**  **wire [6:0] R\_NOT, R\_ROR;**  **// Submodules**  **ALU\_OP1 NOT1 (.A(A), .R(R\_NOT));**  **ALU\_OP2 ROR1 (.A(A), .B(B[2:0]), .R(R\_ROR));**  **always @(\*) begin**  **case (OP)**  **2'b00: R = R\_NOT;**  **2'b01: R = R\_ROR;**  **default: R = 7'b0000000;**  **endcase**  **end**  **assign CF = (OP == 2'b01) ? A[B % 7] : 1'b0; // CF is last bit rotated out**  **assign SF = R[6]; // Sign flag (MSB)**  **assign ZF = ~(|R); // Zero flag**  **endmodule** |

**ALU\_TESTBENCH.v**

|  |
| --- |
| **module ALU\_TESTBENCH;**  **reg [6:0] A, B;**  **reg [1:0] OP;**  **wire [6:0] R;**  **wire CF, SF, ZF;**  **ALU uut (**  **.A(A),**  **.B(B),**  **.OP(OP),**  **.R(R),**  **.CF(CF),**  **.SF(SF),**  **.ZF(ZF)**  **);**  **initial begin**  **$dumpfile("test7.vcd");**  **$dumpvars(0, ALU\_TESTBENCH);**  **// Test NOT operation**  **A = 7'b1010101; B = 7'b0000000; OP = 2'b00; #10;**  **A = 7'b0000000; B = 7'b0000000; OP = 2'b00; #10;**  **// Test Rotate Right by various amounts**  **A = 7'b1000001; B = 7'b0000001; OP = 2'b01; #10; // Rotate by 1**  **A = 7'b1100001; B = 7'b0000010; OP = 2'b01; #10; // Rotate by 2**  **A = 7'b1111111; B = 7'b0000101; OP = 2'b01; #10; // Rotate by 5**  **A = 7'b1000000; B = 7'b0000011; OP = 2'b01; #10; // Rotate by 3**  **A = 7'b0000001; B = 7'b0000011; OP = 2'b01; #10; // Rotate by 3**  **A = 7'b1000001; B = 7'b0000000; OP = 2'b01; #10; // Rotate by 0**  **A = 7'b1000001; B = 7'b0000111; OP = 2'b01; #10; // Rotate by 7 (same as original)**  **$finish;**  **end**  **initial begin**  **$monitor("Time=%0t A=%b B=%b OP=%b -> R=%b CF=%b ZF=%b SF=%b", $time, A, B, OP, R, CF, ZF, SF);**  **end**  **endmodule** |

**RTL Timing Diagram:**

|  |  |
| --- | --- |
| **Check List:** Have you added all the timing diagrams of ALU\_TESTBENCH? | YES |



**Category: B**

**Word Size = 7bits**

**ALU Operations = NOT ,ROR**

**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**

=== ALU ===

METRIC COUNT

Number of wires: 134

Number of wire bits: 153

Number of public wires: 7

Number of public wire bits: 26

Number of ports: 7

Number of port bits: 26

Number of memories: 0

Number of memory bits: 0

Number of processes: 0

Number of cells: 137

**Standard cell usage in synthesized design**

sky130\_fd\_sc\_hd\_\_a2111oi\_2 1

sky130\_fd\_sc\_hd\_\_a211o\_2 4

sky130\_fd\_sc\_hd\_\_a21bo\_2 2

sky130\_fd\_sc\_hd\_\_a21o\_2 5

sky130\_fd\_sc\_hd\_\_a21oi\_2 4

sky130\_fd\_sc\_hd\_\_a221o\_2 3

sky130\_fd\_sc\_hd\_\_a221oi\_2 3

sky130\_fd\_sc\_hd\_\_a22o\_2 1

sky130\_fd\_sc\_hd\_\_a311o\_2 1

sky130\_fd\_sc\_hd\_\_a31o\_2 4

sky130\_fd\_sc\_hd\_\_a31oi\_2 1

sky130\_fd\_sc\_hd\_\_a32o\_2 2

sky130\_fd\_sc\_hd\_\_and2\_2 4

sky130\_fd\_sc\_hd\_\_and2\_4 1

sky130\_fd\_sc\_hd\_\_and3\_2 4

sky130\_fd\_sc\_hd\_\_and4b\_2 1

sky130\_fd\_sc\_hd\_\_buf\_2 1

sky130\_fd\_sc\_hd\_\_inv\_2 12

sky130\_fd\_sc\_hd\_\_mux2\_1 3

sky130\_fd\_sc\_hd\_\_mux2\_2 4

sky130\_fd\_sc\_hd\_\_mux4\_2 8

sky130\_fd\_sc\_hd\_\_nand2\_2 9

sky130\_fd\_sc\_hd\_\_nand2b\_2 2

sky130\_fd\_sc\_hd\_\_nand3\_2 2

sky130\_fd\_sc\_hd\_\_nand3b\_2 1

sky130\_fd\_sc\_hd\_\_nor2\_2 7

sky130\_fd\_sc\_hd\_\_nor2b\_2 1

sky130\_fd\_sc\_hd\_\_nor3\_2 1

sky130\_fd\_sc\_hd\_\_nor3b\_2 1

sky130\_fd\_sc\_hd\_\_nor4\_4 1

sky130\_fd\_sc\_hd\_\_nor4b\_2 1

sky130\_fd\_sc\_hd\_\_o211a\_2 8

sky130\_fd\_sc\_hd\_\_o211a\_4 2

sky130\_fd\_sc\_hd\_\_o211ai\_2 1

sky130\_fd\_sc\_hd\_\_o21a\_2 5

sky130\_fd\_sc\_hd\_\_o21bai\_2 1

sky130\_fd\_sc\_hd\_\_o22a\_2 1

sky130\_fd\_sc\_hd\_\_o2bb2a\_2 1

sky130\_fd\_sc\_hd\_\_o311a\_2 1

sky130\_fd\_sc\_hd\_\_o31a\_2 3

sky130\_fd\_sc\_hd\_\_o31ai\_2 1

sky130\_fd\_sc\_hd\_\_o32ai\_2 1

sky130\_fd\_sc\_hd\_\_or2\_2 6

sky130\_fd\_sc\_hd\_\_or2\_4 2

sky130\_fd\_sc\_hd\_\_or3\_2 1

sky130\_fd\_sc\_hd\_\_or3\_4 2

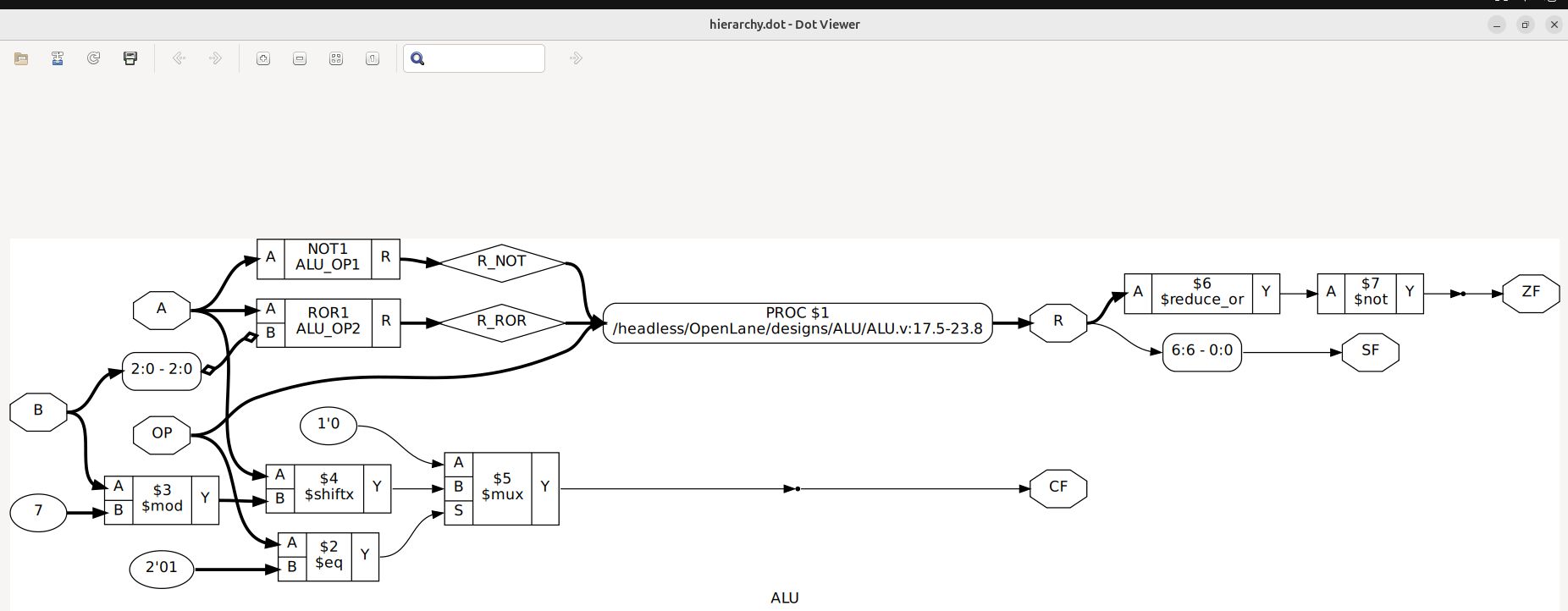
sky130\_fd\_sc\_hd\_\_or3b\_2 1

sky130\_fd\_sc\_hd\_\_or4\_4 1

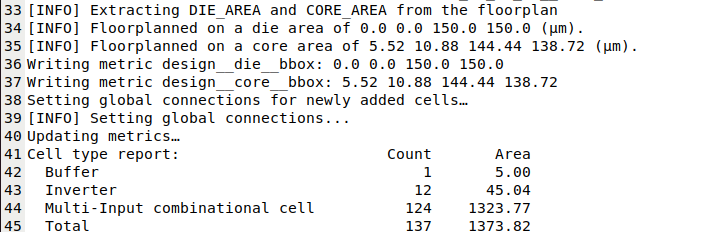
sky130\_fd\_sc\_hd\_\_xnor2\_2 2

sky130\_fd\_sc\_hd\_\_xor2\_2 2

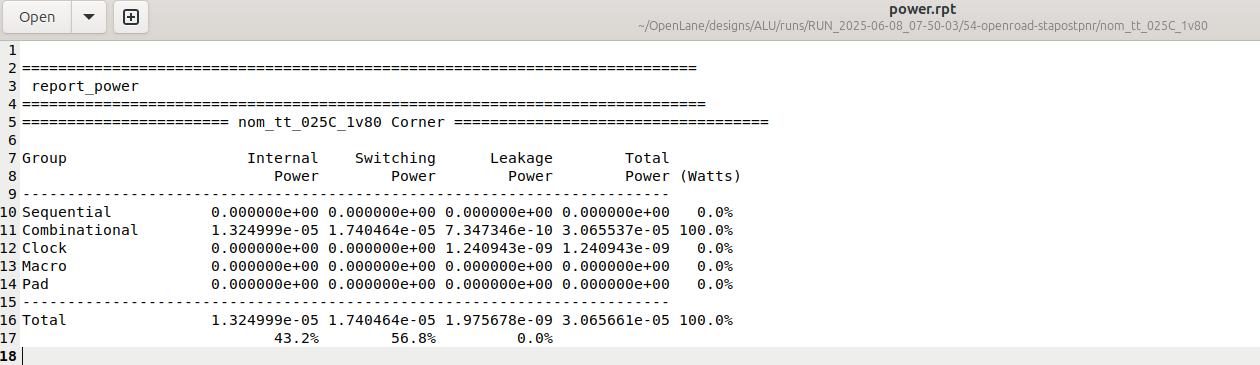
**RTL synthesized design figure**



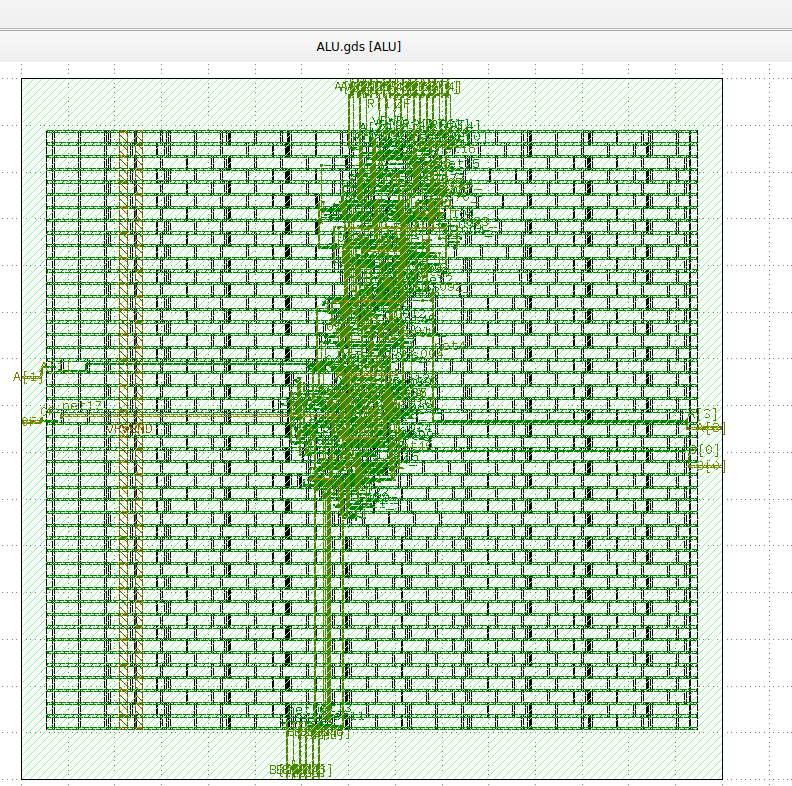
**RTL Floorplan (130nm Skywater PDK with OpenLane toolchain)**



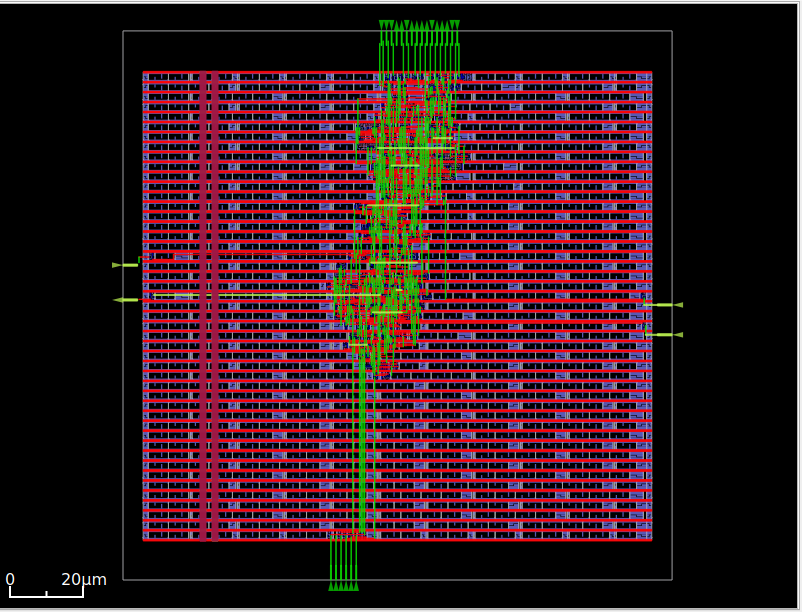
**RTL Power Analysis (130nm Skywater PDK with OpenLane toolchain)**



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



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



**Category: C**

**Word Size = 7bits**

**ALU Operations = NOT ,ROR**

**HDL Code:**

|  |  |
| --- | --- |
| **Check List:** Have you added all the modules written in verilog including CONTROLLER,TOP, TOP\_TESTBENCH,CONTROLLER\_TESTBENCH? | YES |

**TOP.v**

|  |
| --- |
| **module TOP(**  **input wire clk, reset,**  **output wire [6:0] result,**  **output wire flag**  **);**  **wire [6:0] a, b;**  **wire [2:0] op;**  **wire [6:0] res;**  **wire f;**  **CONTROLLER controller(**  **.clk(clk), .reset(reset),**  **.a(a), .b(b), .op(op),**  **.result(res),**  **.flag(f)**  **);**  **CONTROLLER\_TESTBENCH datapath(**  **.a(a), .b(b), .op(op),**  **.result(res), .flag(f)**  **);**  **assign result = res;**  **assign flag = f;**  **endmodule** |

**TOP\_CONTROLLER.v**

|  |
| --- |
| **`timescale 1ns/1ns**  **module TOP\_TESTBENCH;**  **reg clk, reset;**  **wire [6:0] result;**  **wire flag;**  **TOP top (**  **.clk(clk), .reset(reset),**  **.result(result), .flag(flag)**  **);**  **initial begin**  **clk = 0;**  **forever #5 clk = ~clk;**  **end**  **initial begin**  **$dumpfile("alu\_fsm.vcd");**  **$dumpvars(0,TOP\_TESTBENCH);**  **reset = 1;**  **#10;**  **reset = 0;**  **#100; // Run for a few clock cycles**  **$finish;**  **end**  **endmodule** |

**CONTROLLER.v**

|  |
| --- |
| **module CONTROLLER(**  **input wire clk, reset,**  **output reg [6:0] a, b,**  **output reg [2:0] op,**  **input wire [6:0] result,**  **input wire flag**  **);**  **reg [2:0] state, next\_state;**  **parameter START = 3'b000,**  **ONE = 3'b001,**  **TWO = 3'b010,**  **THREE = 3'b011,**  **FINISH = 3'b100;**  **always @(posedge clk or posedge reset) begin**  **if (reset) state <= START;**  **else state <= next\_state;**  **end**  **always @(\*) begin**  **a = 7'd0; b = 7'd0; op = 3'd0;**  **case (state)**  **START: next\_state = ONE;**  **ONE: begin**  **a = 7'b0101010; op = 3'd0; // NOT**  **next\_state = TWO;**  **end**  **TWO: begin**  **a = 7'b0001101; op = 3'd1; // ROTATE RIGHT**  **next\_state = THREE;**  **end**  **THREE: begin**  **a = 7'd0; op = 3'd2; // NOP or final test case**  **next\_state = FINISH;**  **end**  **FINISH: next\_state = FINISH;**  **default: next\_state = START;**  **endcase**  **end**  **endmodule** |

**CONTROLLER\_TESTBENCH.v**

|  |
| --- |
| **module TOP(**  **input wire clk, reset,**  **output wire [6:0] result,**  **output wire flag**  **);**  **wire [6:0] a, b;**  **wire [2:0] op;**  **wire [6:0] res;**  **wire f;**  **CONTROLLER controller(**  **.clk(clk), .reset(reset),**  **.a(a), .b(b), .op(op),**  **.result(res),**  **.flag(f)**  **);**  **CONTROLLER\_TESTBENCH datapath(**  **.a(a), .b(b), .op(op),**  **.result(res), .flag(f)**  **);**  **assign result = res;**  **assign flag = f;**  **endmodule** |

**RTL Timing Diagram:**

|  |  |
| --- | --- |
| **Check List:** Have you added all the timing diagrams of CONTROLLER\_TESTBENCH, TOP\_TESTBENCH? | YES |

