Minimalistic Processor based on Microprocessor 8085

VDF Project Part-I

Group Number-17

Submitted By:

Lakshya Karwasara(2020216)

Ahmad Farhan(MT23209)

Farogh Alam(MT23213)

Abdullah Ansari(MT23150)

Submitted To:

Dr. Sneh Saurabh

Operations and Tools

S. No	Process	Tool
1	Simulation	ncverilog/simvision
2	Coverage	Incisive Metrics
3	Synthesis	Genus
4	Equivalence Checking	Conformal
5	Static Timing Analysis	Tempus
6	Design ForTesting (DFT)	Genus

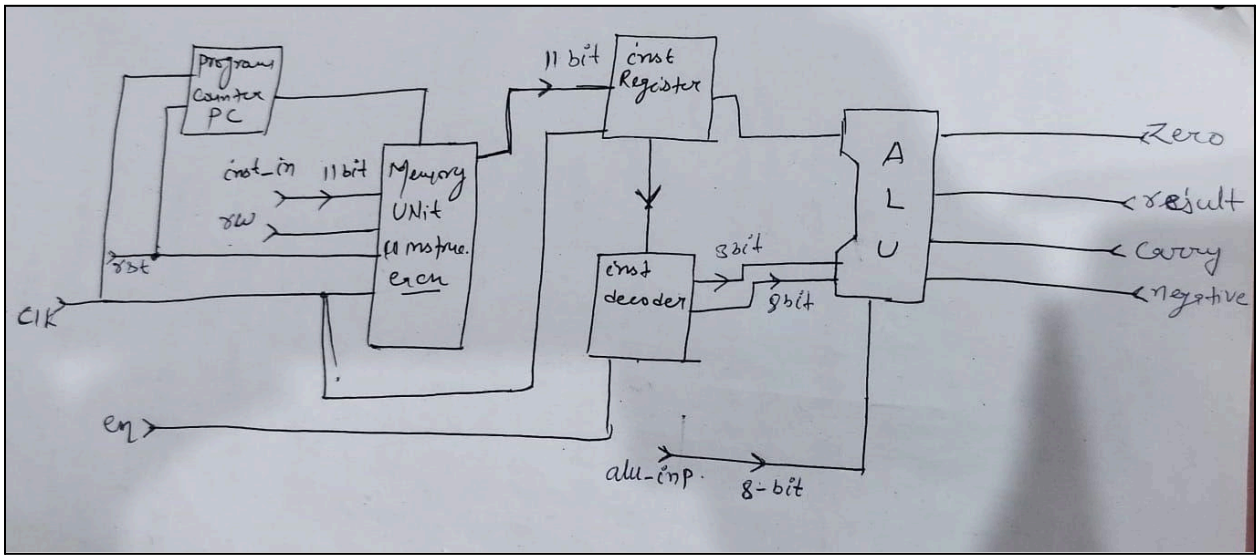
Library used: Cadence 90nm: Fast.lib

Project Specification

Minimalistic microprocessor 8085 Implementation:

Problem Statement:

The task involves designing and implementing a microprocessor based on the Intel 8085 architecture, capable of executing instructions and generating the desired output accurately. The microprocessor 8085 is a classic example of a Complex Instruction Set Computing (CISC) architecture, characterized by its ability to perform a wide range of operations using comprehensive instruction sets.



Overview:

Central to the microprocessor 8085 architecture are components such as the Program Counter (PC), Instruction Memory, Arithmetic and Logic Unit (ALU), Registers, and Control Unit. The PC tracks the memory address of the next instruction to be fetched, incrementing with each clock cycle or upon receiving a reset signal. This address points to the Instruction Memory, where instructions, typically 8-bit or 16-bit in size, are stored. Instructions within the microprocessor 8085 are encoded with opcodes and additional segments specifying operands, memory addresses, or data. The opcode dictates the

operation to be executed by the microprocessor. The opcode is decoded to determine the appropriate action after fetching an instruction.

The ALU performs arithmetic and logical operations, utilizing data and operands stored in various registers, including the Accumulator, General Purpose Registers, and Special Purpose Registers. Memory operations involve reading from or writing to data memory based on specified addresses.

The Control Unit plays a pivotal role in coordinating the execution of instructions by generating control signals for memory read, memory write, and register write operations. It ensures seamless interaction between different components of the microprocessor.

The microprocessor 8085 architecture offers a robust framework for executing instructions efficiently. By adhering to the architectural principles outlined above, the microprocessor can accurately execute instructions and perform various arithmetic, logical, and memory operations essential for computing tasks.

Verilog Code:

Module for Program Counter:

```
module pc (
    input clk,                // Clock input
    input rst,                // Reset signal
    output reg [5:0] pc_out_addr // Address output for instruction memory
);
    reg [5:0] next_addr;       // next address register

    always @(posedge clk or posedge rst) begin
        if (rst) begin
            next_addr <= 5'b00000; // Reset the next_addr to 0
            pc_out_addr <= 5'b00000;    // Reset the address output
        end
        else begin
            // Increment the counter on each clock cycle
            next_addr <= next_addr + 1;

            // Output the incremented value as the address
            pc_out_addr <= next_addr;
        end
    end
endmodule
```

The Program Counter (PC) module tracks memory addresses for instruction fetching. It operates synchronously with the clock signal (`clk`) and resets (`rst`) to establish a known starting point. On each clock cycle, it increments its internal register, indicating the next instruction's address. This address is then outputted, guiding the microprocessor's instruction retrieval. Crucial for program flow, the PC module ensures sequential instruction execution and forms the backbone of microprocessor programming, facilitating efficient program execution.

Module for Instruction Memory:

```
module instr_memory (
    input [5:0] mem_addr,
    input [10:0] instr_in,    // Address for writing instruction. i.e 8 words memory
    output reg [10:0] mem_instr_out, // Instruction output
    input rw, // read/write control
    input clk, // clock input
    input rst // reset signal
   );
    reg [10:0] mem[63:0]; // Instruction memory array
    integer i;
    // Memory initialization
    always @(*) begin
        if (rst) begin

        for (i = 0; i < 64; i = i + 8) begin
            mem[i] = 11'b00010010010;   // Initialize each memory location to 0
            mem[i+1] = 11'b00110011010;
            mem[i+2] = 11'b01011010010;
            mem[i+3] = 11'b01110010010;
            mem[i+4] = 11'b10010110010;
            mem[i+5] = 11'b10110011110;
            mem[i+6] = 11'b11010011010;
            mem[i+7] = 11'b11111110010;

        end

        end
    end
```

```
    // Memory read and write operations
    always @(posedge clk) begin
        if (!rst) begin
            if (!rw)
                mem[mem_addr] <= instr_in;
            else
                mem_instr_out <= mem[mem_addr]; // Read operation
        end
    end
endmodule
```

The Instruction Memory module stores and retrieves instructions based on memory addresses. It comprises an array `mem` initialized to zeros upon reset (`rst`). Each memory location stores an 11-bit instruction. On reset, memory is pre-loaded with instruction patterns. Upon receiving a clock signal (`clk`) during operation and when not in reset, the module performs read/write operations based on the control signal `rw`. When writing (`rw=0`), the input instruction is written to the specified memory address (`mem_addr`). When reading (`rw=1`), the module outputs the instruction stored at the

specified address. This module is crucial for program execution, providing instructions to the microprocessor as needed.

Module for Instruction register:

```
module instr_reg(ir_inp_instr,en,ir_op_instr);
  input [10:0]ir_inp_instr;
  input en;
  output reg [10:0]ir_op_instr;
  always @(*)
  begin
    if(en==1)begin
  ir_op_instr <= ir_inp_instr;
end
end
endmodule
```

The Instruction Register (IR) module stores and outputs instructions for decoding and execution. It takes as input `ir_inp_instr,` an 11-bit instruction, and an enable signal `en`. When the enable signal is asserted (`en=1`), the module updates its internal register `ir_op_instr` with the input instruction. This register holds the instruction until the next update. The module operates asynchronously, updating the output based on changes in the input and enabling signal. The Instruction Register is a critical component in the microprocessor, facilitating the retrieval and processing of instructions during program execution.

Module for Instruction Decoder:

```
module decoder(in_decoder,en,opcode,out_decoder);
input [10:0]in_decoder;
input en;
output reg [2:0]opcode;
output reg[7:0]out_decoder;
  always @(*)
  begin
    if(en==1)
      begin
  opcode = in_decoder[10:8];
  out_decoder = in_decoder[7:0];
   end
end
endmodule
```

The Decoder module interprets instruction codes into control signals for the microprocessor. It takes an 11-bit instruction `in_decoder` and an enable signal `en`. When enabled (`en=1`), the module extracts the opcode from bits 10 to 8 of the input instruction and assigns it to the `opcode` output. It extracts the remaining bits (7 to 0) and assigns them to the `out_decoder` output. This module operates asynchronously, updating the outputs based on changes in the input instruction and enabling signal. The Decoder is vital for translating instruction codes into signals that govern the microprocessor's behavior during execution.

Module for ALU:

```
module alu(
    input   [7:0] alu_in1,
    input   [7:0] alu_in2,
    input   [2:0] alu_opcode,
    output  zero,
    output  carry,
    output  negative,
    output  [7:0] alu_result
);

    reg [8:0] result_with_carry;

    // Outputs
    reg zero_reg, carry_reg, negative_reg;

    // Intermediate signals
    reg [7:0] result_reg;

    always @* begin
        case(alu_opcode)
            3'b000: begin  // Bitwise OR
                result_reg = alu_in1 | alu_in2;
        zero_reg = result_reg == 8'b0;
        carry_reg = 1'b0;
        negative_reg = 1'b0;
            end

            3'b001: begin  // Bitwise XOR
                result_reg = alu_in1 ^ alu_in2;
        zero_reg = result_reg == 8'b0;
        carry_reg = 1'b0;
        negative_reg = 1'b0;
            end
```

```
        3'b010: begin  // Addition
            result_with_carry = alu_in1 + alu_in2;
            carry_reg = result_with_carry[8];
            result_reg = result_with_carry[7:0];
            zero_reg = result_reg == 8'b0;
    negative_reg = 1'b0;
        end

        3'b011: begin  // Subtraction
            result_with_carry = alu_in1 - alu_in2;
            negative_reg = alu_in1 < alu_in2;
            if (alu_in1 >= alu_in2) begin
                carry_reg = 1'b0;
                zero_reg = result_with_carry[7:0] == 8'b0;
        negative_reg = 1'b0;
            end else begin
                result_reg = ~result_with_carry[7:0] + 1'b1;
                carry_reg = 1'b0;
                zero_reg = 1'b0;
                negative_reg = 1'b1;
            end
        end

        3'b100: begin  // Compare
            negative_reg = alu_in1 < alu_in2;
            carry_reg = alu_in1 >= alu_in2;  // Carry flag indicates A >= B
            zero_reg = alu_in1 == alu_in2;
        end
```

```
        3'b101: begin  // Bitwise AND
            result_reg = alu_in1 & alu_in2;
            zero_reg = result_reg == 8'b0;
            carry_reg = 1'b0;
            negative_reg = 1'b0;
        end

        3'b110: begin  // Shift Left (SLL)
            result_reg = alu_in1 << alu_in2[2:0];
            zero_reg = result_reg == 8'b0;
    carry_reg = 1'b0;
            negative_reg = 1'b0;
        end

        3'b111: begin  // Shift Right (SRL)
            result_reg = alu_in1 >> alu_in2[2:0];
            zero_reg = result_reg == 8'b0;
    carry_reg = 1'b0;
            negative_reg = 1'b0;
        end
```

```
        default: begin
    // Default to grounding the floating pins
    result_reg[0] = 1'b0; // Set bit 0 to 0
    result_reg[1] = 1'b0; // Set bit 1 to 0
    result_reg[2] = 1'b0; // Set bit 2 to 0
    result_reg[3] = 1'b0; // Set bit 3 to 0
    result_reg[4] = 1'b0; // Set bit 4 to 0
    result_reg[5] = 1'b0; // Set bit 5 to 0
    result_reg[6] = 1'b0; // Set bit 6 to 0
    result_reg[7] = 1'b0; // Set bit 7 to 0
    zero_reg = 1'b1;     // Ground zero output
    carry_reg = 1'b0;    // Ground carry output
    negative_reg = 1'b0; // Ground negative output
end

endcase
    end

    assign zero = zero_reg;
    assign carry = carry_reg;
    assign negative = negative_reg;
    assign alu_result = result_reg;

endmodule
```

The Arithmetic Logic Unit (ALU) module performs arithmetic and logical operations on two 8-bit inputs (`alu_in1` and `alu_in2`) based on the specified operation (`alu_opcode`). It produces the result (`alu_result`) along with status flags: zero (`zero`), carry (`carry`), and negative (`negative`). Within the ALU, a case statement interprets the operation code (`alu_opcode`) to determine the appropriate operation. It handles operations like bitwise OR, XOR, addition, subtraction, comparison, bitwise AND, shift left (SLL), and shift right (SRL). The ALU operates synchronously, updating its outputs based on changes to the input signals and the operation code. The ALU's functionality is crucial for executing arithmetic and logical operations within the microprocessor, making it a core component of computational tasks. A separate flip-flop module (`ff`) also stores and updates data asynchronously based on clock cycles, ensuring stable output. This facilitates reliable data handling within the microprocessor architecture.

Module for flip flop:

```
module ff (
    input wire clk,       // Clock input
    input d,        // 8-bit Data input
    output reg q    // 8-bit Output
);

    always @(posedge clk) begin
        q <= d;           // Update output based on data input on positive clock edge
    end

endmodule
```

```
module ff_8bit (
    input wire clk,      // Clock input
    input [7:0] d,       // 8-bit Data input
    output reg [7:0] q   // 8-bit Output
);

    always @(posedge clk) begin
        q <= d;  // Update output based on data input on positive clock edge
    end

endmodule
```

Module for Instantiating different modules:

```
module top8_module (
    input clk,                  // Clock input
    input rst,                  // Reset signal
    input [7:0] alu_in1_inp,

    input [10:0] inst_in,       // Instruction input
    input rw,                   // Read/Write control
    input en,
    output wire [7:0] alu_result,
    output wire ff_zero,
    output wire ff_carry,
    output wire ff_negative
);
    wire ff_pc_rst;
    wire [7:0] alu_in1_ff; // Declare 8-bit wire to connect flip-flop output to ALU input
    wire [5:0] pc_out_addr;      // Address output for instruction memory
    wire [10:0] mem_instr_out;   // Output from instruction memory
    wire [10:0] ir_op_instr;     // Output from instruction register
    wire [2:0] opcode;           // Output opcode from decoder
    wire [7:0] out_decoder;      // Output from decoder
    wire carry;                  // ALU carry output
    wire negative;               // ALU negative output
    wire zero;                   // ALU zero output

    // Instantiate 8-bit flip-flop for alu_in1
    ff_8bit ff_alu_in1 (
        .clk(clk),
        .d(alu_in1_inp),
        .q(alu_in1_ff)
    );
```

```
    // Instantiate Program Counter
    pc pc_inst (
        .clk(clk),
        .rst(ff_pc_rst),
        .pc_out_addr(pc_out_addr)
    );

    // Instantiate Instruction Memory
    instr_memory mem_inst (
        .rw(rw),
        .instr_in(inst_in),
        .mem_addr(pc_out_addr),
        .mem_instr_out(mem_instr_out),
        .clk(clk),
        .rst(rst)
    );

    // Instantiate instruction register
    instr_reg ir_inst (
        .ir_inp_instr(mem_instr_out),
        .en(en),
        .ir_op_instr(ir_op_instr)
    );

    // Instantiate decoder
    decoder decoder_inst (
        .in_decoder(ir_op_instr),
        .en(en),
        .opcode(opcode),
        .out_decoder(out_decoder)
    );
```

```
    // Instantiate flip-flops
    ff ff_pc_rst_inst (
        .clk(clk),
        .d(rst),
        .q(ff_pc_rst)
    );

    ff ff_carry_inst (
        .clk(clk),
        .d(carry),
        .q(ff_carry)
    );

    ff ff_zero_inst (
        .clk(clk),
        .d(zero),
        .q(ff_zero)
    );

    ff ff_negative_inst (
        .clk(clk),
        .d(negative),
        .q(ff_negative)
    );

    alu alu_inst (
        .alu_in1(alu_in1_ff),  // Connect to the output of 8-bit flip-flop
        .alu_in2(out_decoder),
        .alu_opcode(opcode),
        .zero(zero),
        .carry(carry),
        .negative(negative),
        .alu_result(alu_result)
    );
endmodule
```

The Top Module is the central control hub in the microprocessor design, coordinating the interaction between its core components. At its heart lies the Program Counter (PC), responsible for tracking memory addresses, while the Instruction Memory stores and retrieves instructions based on these addresses. Fetched instructions are stored temporarily in the Instruction Register (IR) before decoding by the Decoder module, translating them into the microprocessor's control signals. Additionally, flip-flops are utilized to store and update status flags asynchronously. The Arithmetic Logic Unit (ALU) executes arithmetic and logical operations based on decoded instructions. Through meticulous orchestration of these components and their interactions, the Top Module ensures efficient and accurate execution of instructions within the microprocessor architecture.

Testbench and Simulation:

We developed the testbench that simulates the above design and compares the result with the specifications. Three different testbenches are written, and a code-coverage analysis is performed. The results are attached below.

The commands used for code coverage in ncverilog.

To check the code coverage:

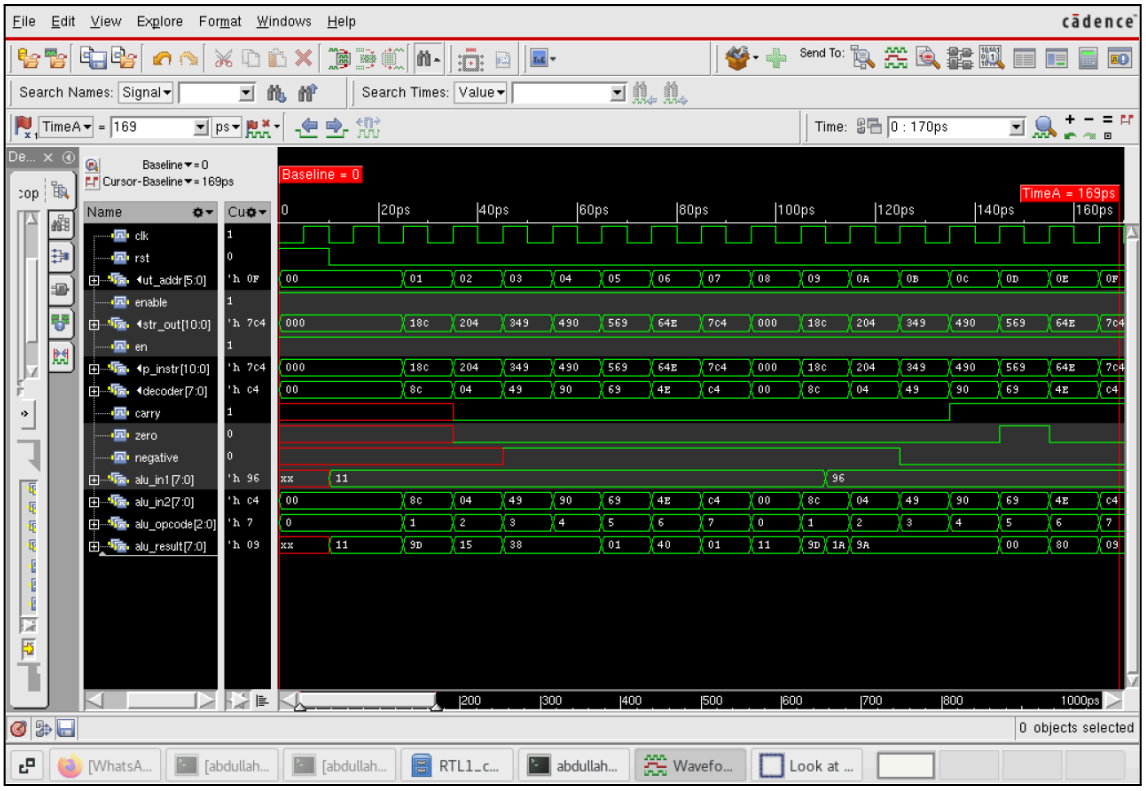
```
tcsh
source /cadence/cshrc
ncverilog -access rwc top5.v top5_tb.v -coverage all -gui
```

Testbench 1:

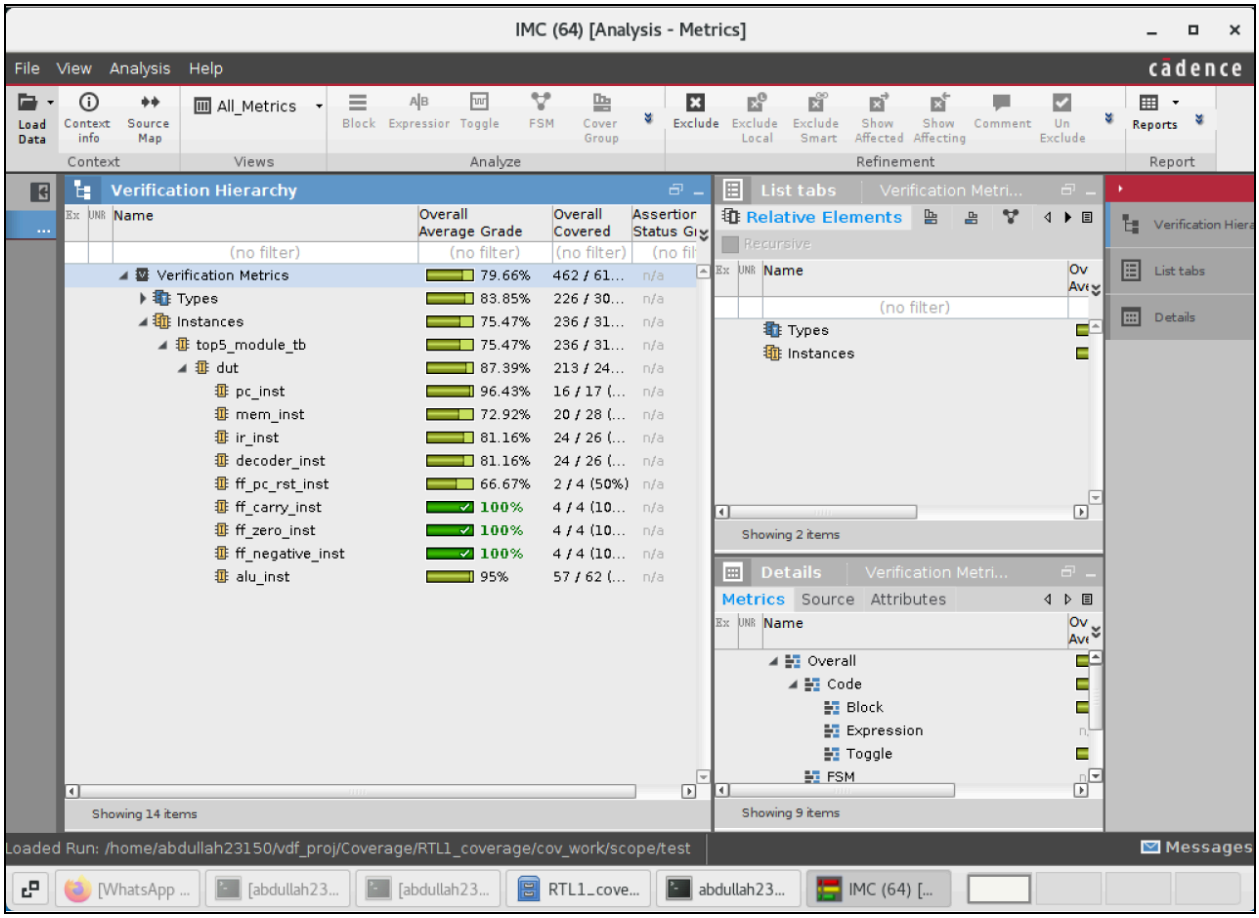
```
`timescale 1ps/1ps
module top5_module_tb();

    reg clk;                    // Clock input
    reg rst;                    // Reset signal
    reg enable;                 // Enable signal for memory
    reg [5:0] mem_addr;         // Memory address
    reg [10:0] in_decoder;      // Input to the decoder
    reg en;                     // Enable signal for decoder
    reg [7:0] alu_in1, alu_in2; // ALU inputs
    reg [7:0] out_decoder;
    reg [2:0] alu_opcode;       // ALU opcode
    wire ff_zero, ff_carry, ff_negative;
    wire [7:0] alu_result;

    // Instantiate the top module
    top5_module dut (
        .clk(clk),
        .rst(rst),
        .alu_in1(alu_in1),
        .ff_zero(ff_zero),
        .ff_carry(ff_carry),
        .ff_negative(ff_negative),
        .alu_result(alu_result)
    );

    // Clock generation
    always #5 clk = ~clk;
```

```
    // Stimulus Generation
    initial begin
        clk = 0;
        rst = 1;
        #10 rst = 0;
        alu_in1 = 8'h11;
        #100;
        alu_in1 = 8'h96;
        #100;
        alu_in1 = 8'h11;
    end

    // Dump VCD file for waveform generation
    initial begin
        $dumpfile("top5_module_tb.vcd");
        $dumpvars(0,top5_module_tb);
        #1000; // Simulate for 1000 time units
        $finish; // Finish simulation
    end

endmodule
```

Simulation:


File Edit View Explore Format Windows Help
cadence
Search Names: Signal
Search Times: Value
Send To:
TimeA = 169 ps
Time: 0 : 170ps
Baseline = 0
Cursor-Baseline = 169ps
Baseline = 0
TimeA = 169ps
Name
clk
rst
out_addr[5:0]
enable
instr_out[10:0]
en
ip_instr[10:0]
decoder[7:0]
carry
zero
negative
alu_in1[7:0]
alu_in2[7:0]
alu_opcode[2:0]
alu_result[7:0]
0 20ps 40ps 60ps 80ps 100ps 120ps 140ps 160ps
200 300 400 500 600 700 800 1000ps
0 objects selected

Code-Coverage Report 1:



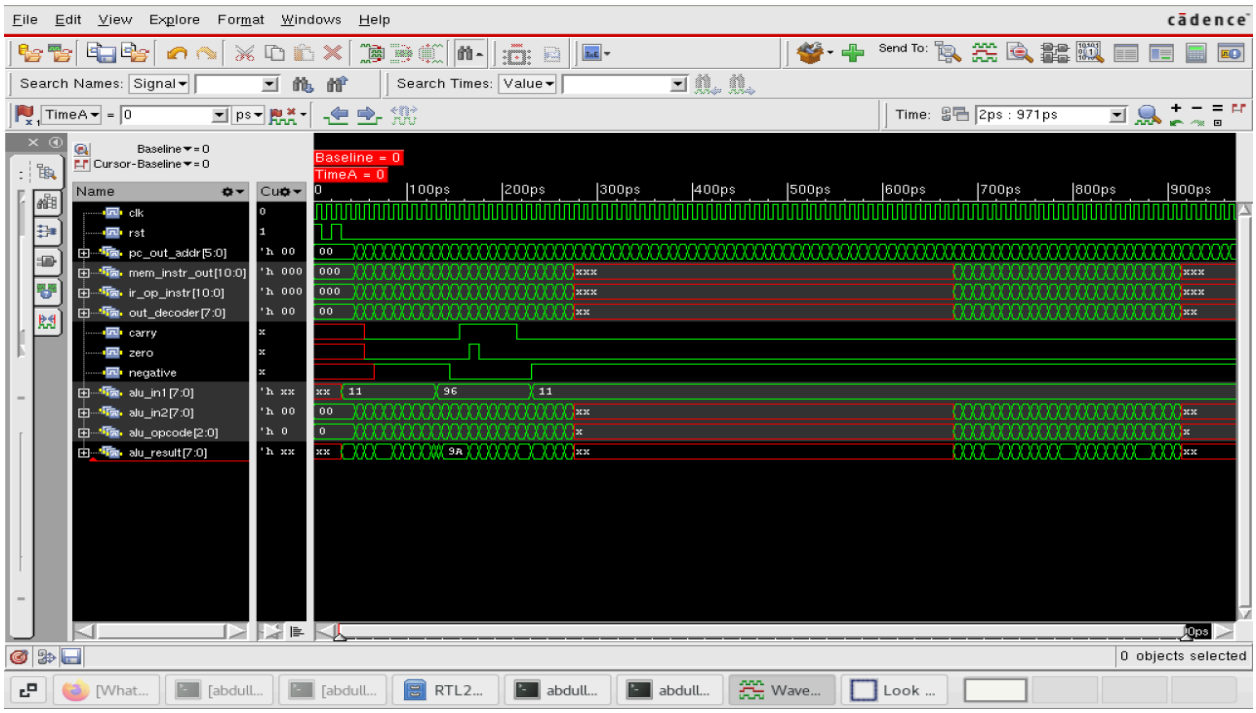
Analysis for 1st case: The simulation results are shown for the given test bench. Only two inputs were provided for 100 ps each for the ALU. As a result, alu showed only 95% coverage. ff_pc_rst_inst, decoder, and program counter modules were not getting toggled in some cases. Hence, their overall coverage is stuck at **79.66%**.

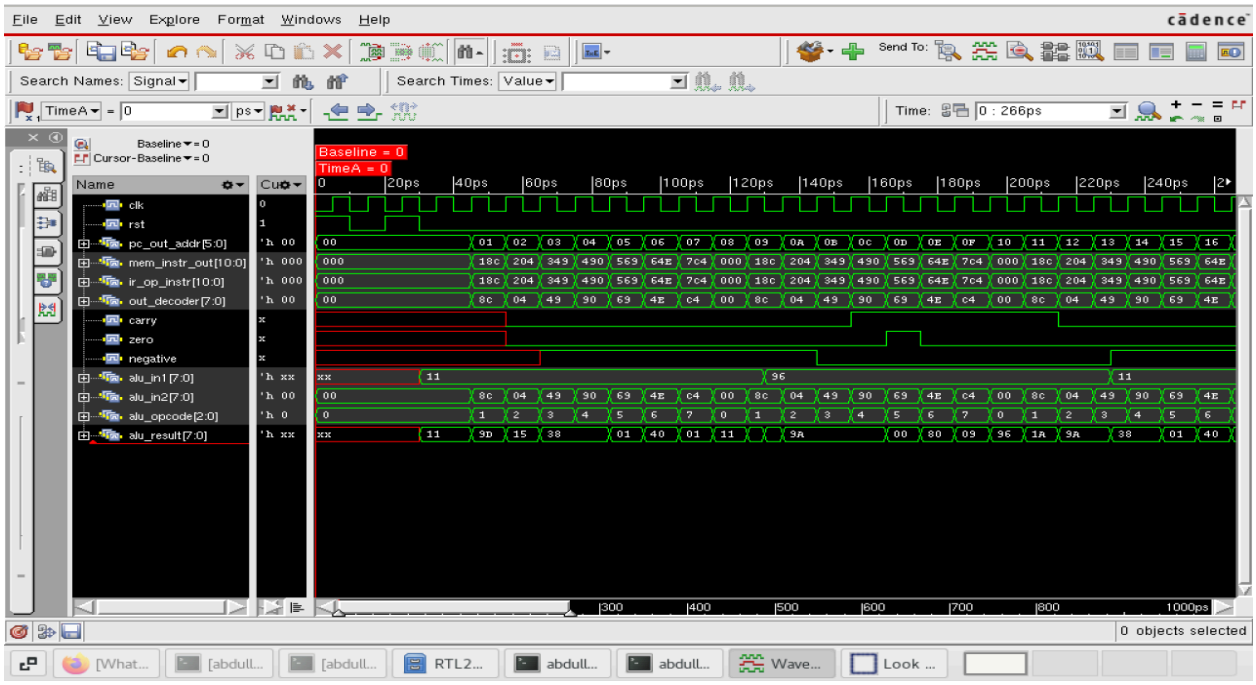
Testbench 2:

```
`timescale 1ps/1ps
module top5_module_tb();

    reg clk;                  // Clock input
    reg rst;                  // Reset signal
    reg [7:0] alu_in1; // ALU input
    wire ff_zero, ff_carry, ff_negative;
    wire [7:0] alu_result;

    // Instantiate the top module
    top5_module dut (
        .clk(clk),
        .rst(rst),
        .alu_in1(alu_in1),
        .ff_zero(ff_zero),
        .ff_carry(ff_carry),
        .ff_negative(ff_negative),
        .alu_result(alu_result)
    );

    // Clock generation
    always #5 clk = ~clk;
```

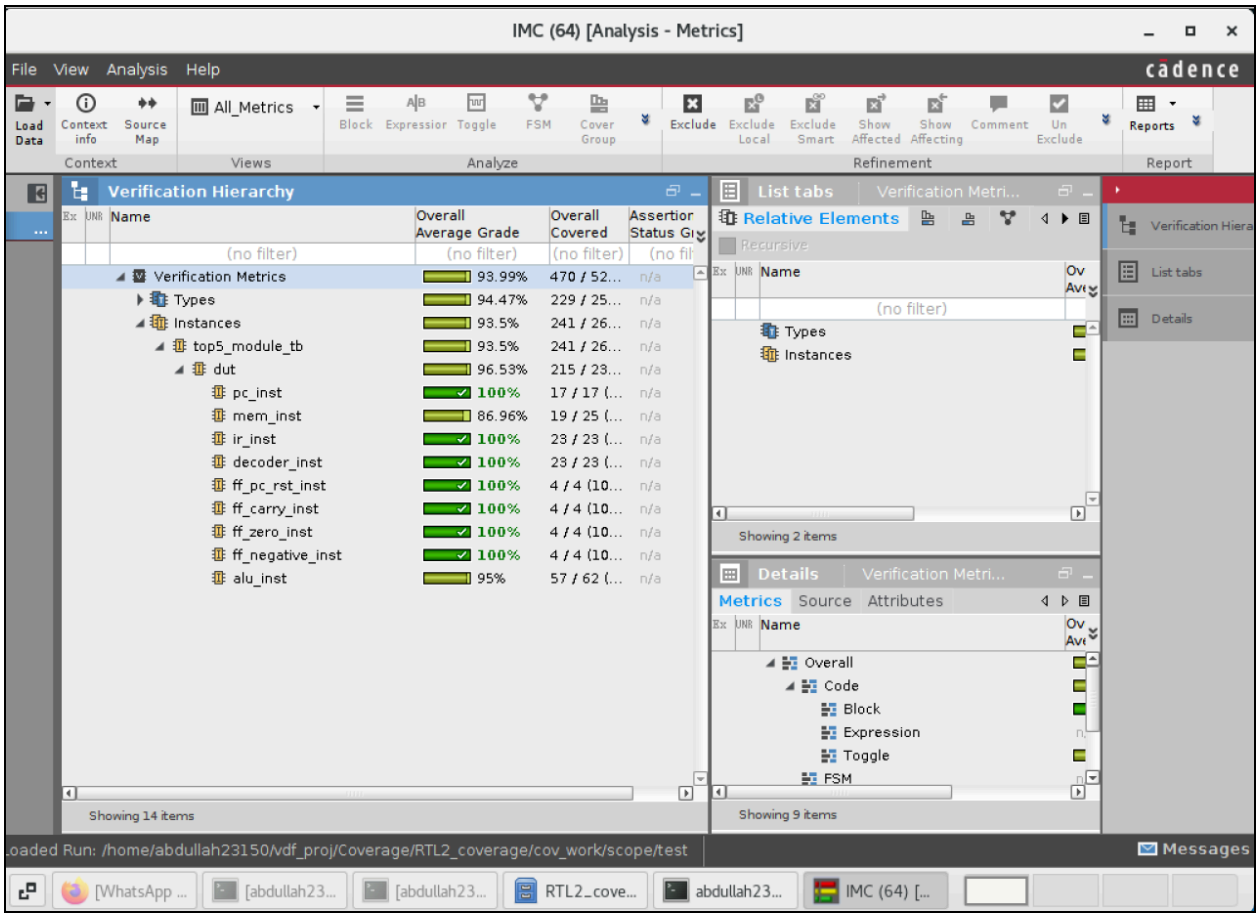
```
    // Stimulus Generation
    initial begin
        clk = 0;
        rst = 1;
        #10 rst = 0;
        #10 rst = 1;
        #10 rst = 0;
        alu_in1 = 8'h11;
        #100;
        alu_in1 = 8'h96;
        #100;
        alu_in1 = 8'h11;
    end

    // Dump VCD file for waveform generation
    initial begin
        $dumpfile("top5_module_tb.vcd");
        $dumpvars(0,top5_module_tb);
        #1000; // Simulate for 1000 time units
        $finish; // Finish simulation
    end

endmodule
```

Simulation:


File Edit View Explore Format Windows Help
cadence
Search Names: Signal
Search Times: Value
Time: 2ps : 971ps
Baseline = 0
TimeA = 0
Name
clk
rst
pc_out_addr[5:0]
mem_instr_out[10:0]
ir_op_instr[10:0]
out_decoder[7:0]
carry
zero
negative
alu_in1[7:0]
alu_in2[7:0]
alu_opcode[2:0]
alu_result[7:0]
0 objects selected


File Edit View Explore Format Windows Help
cadence
Search Names: Signal
Search Times: Value
Time: 0 : 266ps
Baseline = 0
TimeA = 0
Name
clk
rst
pc_out_addr[5:0]
mem_instr_out[10:0]
ir_op_instr[10:0]
out_decoder[7:0]
carry
zero
negative
alu_in1[7:0]
alu_in2[7:0]
alu_opcode[2:0]
alu_result[7:0]
0 objects selected

Code-Coverage Report 2:



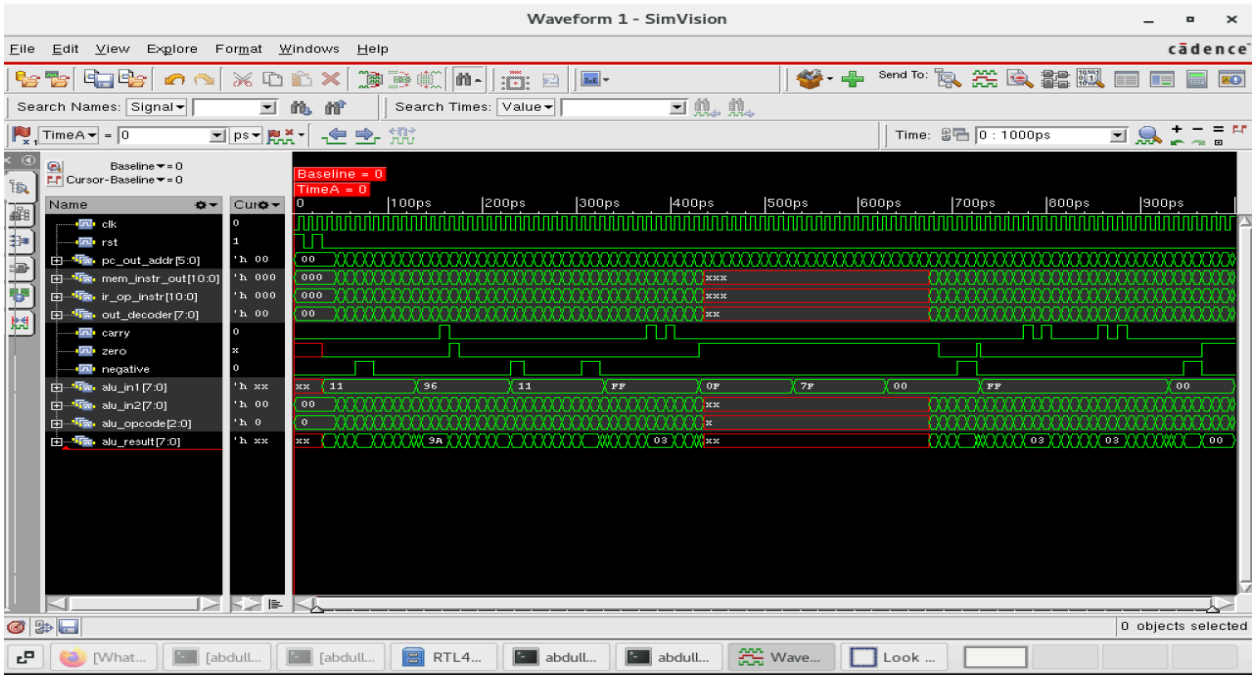
Analysis for 2nd case: After enabling signal carry, zero, and negative flag registers, and rst getting toggled enough without making extra changes to ALU input or in testbench, the code coverage went to 100% for those modules, and the overall coverage updated to **93.99%.**

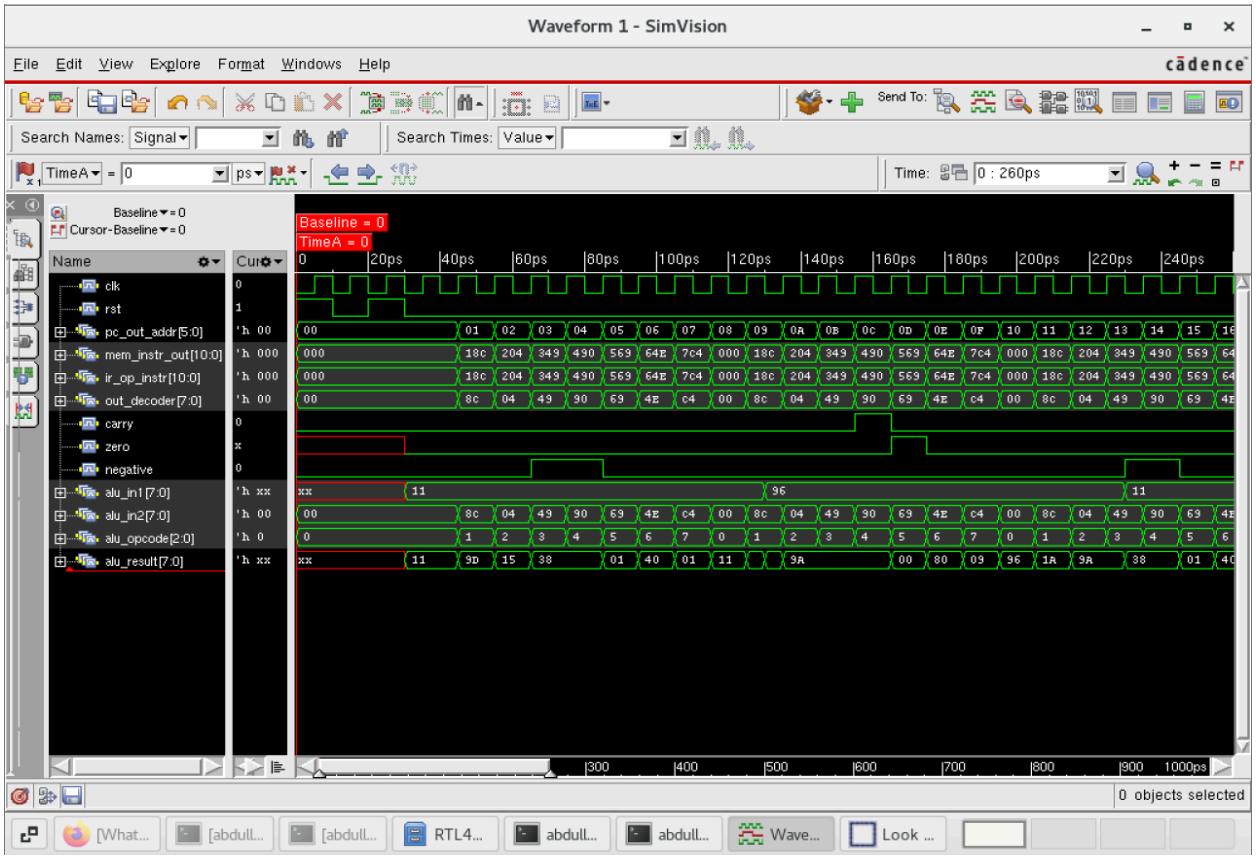
Testbench 3:

```
`timescale 1ps/1ps
module top5_module_tb();

    reg clk;                    // Clock input
    reg rst;                    // Reset signal
    reg [7:0] alu_in1;          // ALU input
    wire ff_zero, ff_carry, ff_negative;
    wire [7:0] alu_result;

    // Instantiate the top module
    top5_module dut (
        .clk(clk),
        .rst(rst),
        .alu_in1(alu_in1),
        .ff_zero(ff_zero),
        .ff_carry(ff_carry),
        .ff_negative(ff_negative),
        .alu_result(alu_result)
    );

    // Clock generation
    always #5 clk = ~clk;

    // Stimulus Generation
    initial begin
        clk = 0;
        rst = 1;
        #10 rst = 0;
        #10 rst = 1;
        #10 rst = 0;

        // Test case 1: Test addition operation
        alu_in1 = 8'h11;
        #100;
```

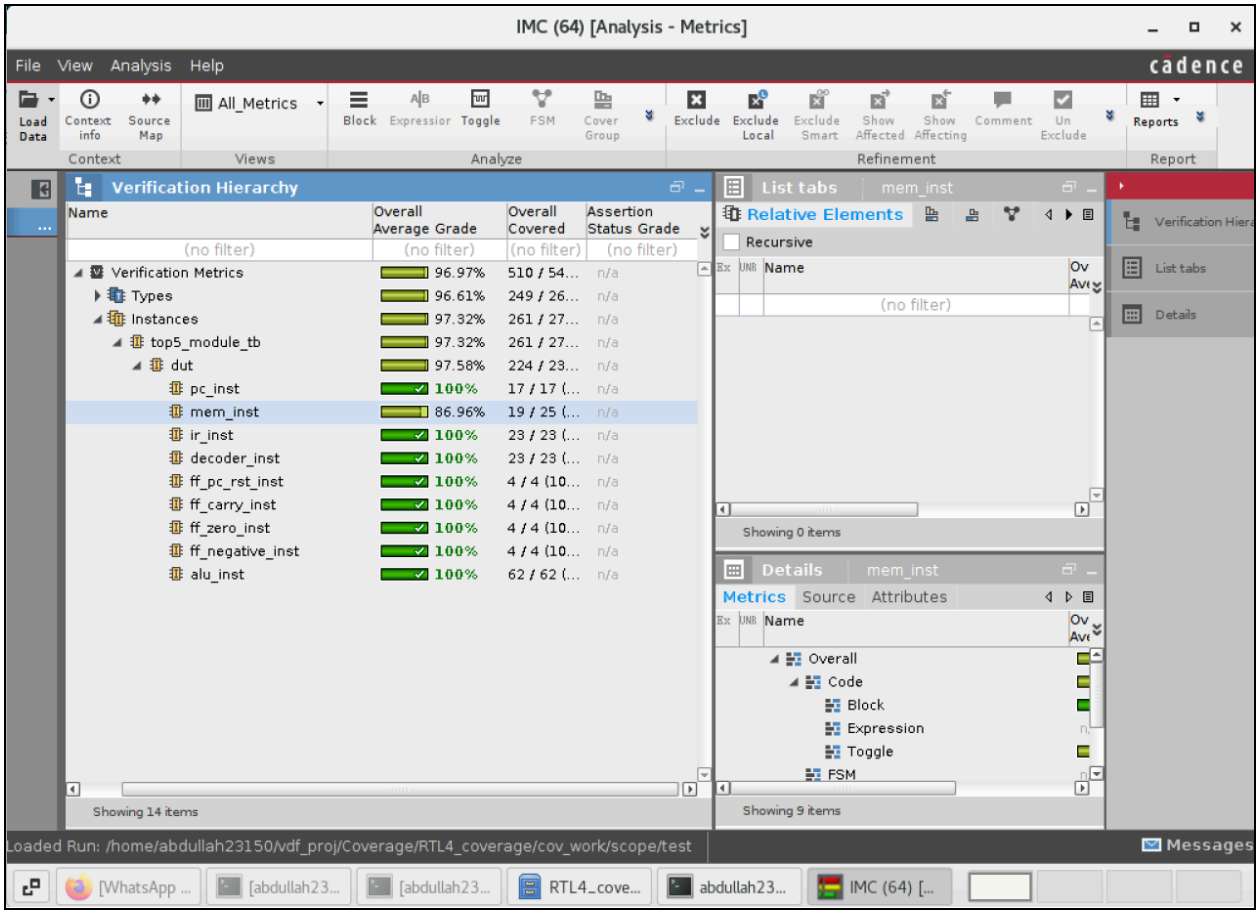
```
        // Test case 2: Test subtraction operation
        alu_in1 = 8'h96;
        #100;
        // Test case 3: Test bitwise AND operation
        alu_in1 = 8'h11;
        #100;
        // Test case 4: Test shift left operation
        alu_in1 = 8'hFF;
        #100;
        // Test case 5: Test shift right operation
        alu_in1 = 8'h0F;
        #100;
        // Test case 6: Test set less than operation
        alu_in1 = 8'h7F;
        #100;
        // Test case 7: Test bitwise OR operation
        alu_in1 = 8'h00;
        #100;
        // Test case 8: Test bitwise XOR operation
        alu_in1 = 8'hFF;
        #100;
        // Test case 9: Test all 1's input
        alu_in1 = 8'hFF;
        #100;
        // Test case 10: Test all 0's input
        alu_in1 = 8'h00;
        #100;
    end
    initial begin
        $dumpfile("top5_module_tb.vcd");
        $dumpvars(0,top5_module_tb);
        #1000; // Simulate for 1000 time units
        $finish; // Finish simulation
    end
endmodule
```

Simulation:


Waveform 1 - SimVision
File Edit View Explore Format Windows Help
cadence
Search Names: Signal
Search Times: Value
Time: 0 : 1000ps
Baseline = 0
TimeA = 0
Name
clk
rst
pc_out_addr[5:0]
mem_instr_out[10:0]
ir_op_instr[10:0]
out_decoder[7:0]
carry
zero
negative
alu_in1[7:0]
alu_in2[7:0]
alu_opcode[2:0]
alu_result[7:0]
0 objects selected


Waveform 1 - SimVision
File Edit View Explore Format Windows Help
cadence
Search Names: Signal
Search Times: Value
Time: 0 : 260ps
Baseline = 0
TimeA = 0
Name
clk
rst
pc_out_addr[5:0]
mem_instr_out[10:0]
ir_op_instr[10:0]
out_decoder[7:0]
carry
zero
negative
alu_in1[7:0]
alu_in2[7:0]
alu_opcode[2:0]
alu_result[7:0]
0 objects selected

Code-Coverage Report 3:



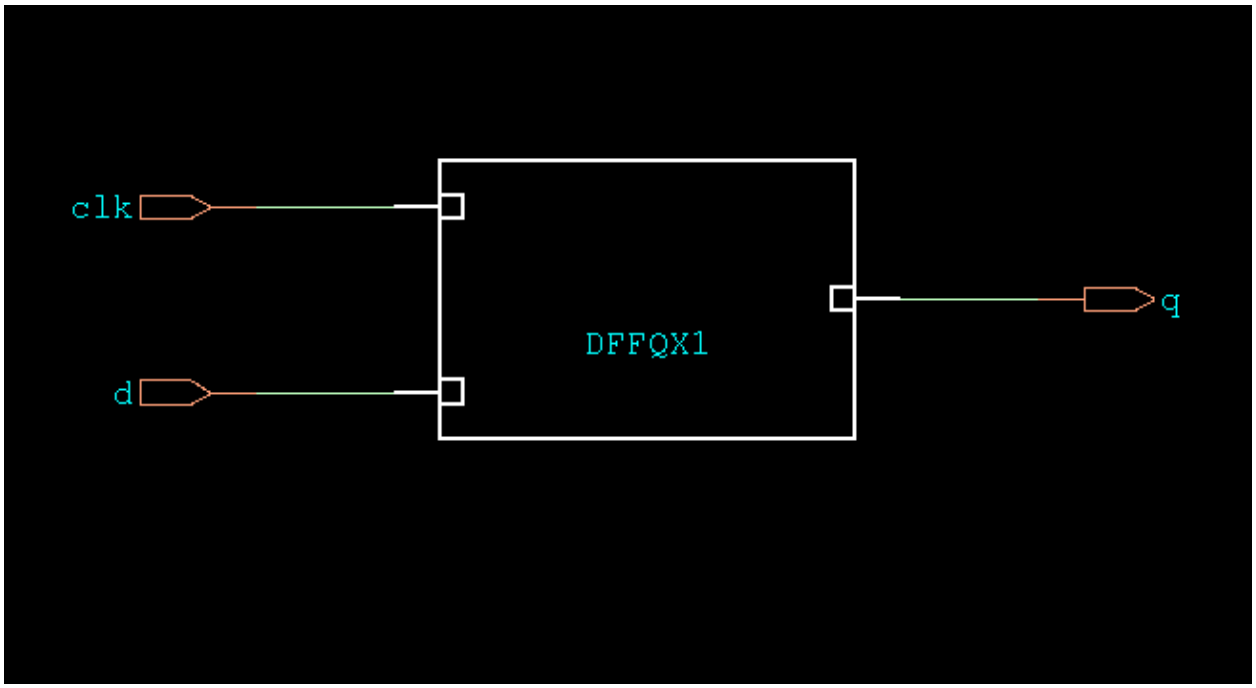
Analysis for 3rd case: In this scenario, the range of direct inputs to the ALU (alu_in1) has been significantly diversified to encompass all cases of ALU functionality. The ALU now receives 10 inputs to ensure comprehensive coverage of its various operations, particularly when paired with a wider range of memory inputs (alu_in2). This comprehensive approach has resulted in achieving 100% coverage for ALU operations. However, full coverage was not obtained for the memory elements across all modules. As a result of these improvements, the overall code coverage has been elevated to **96.97%**.

TESTBENCH	OVERALL COVERAGE (%)
tb1	79.66%
tb2	93.99%
tb3	96.97%

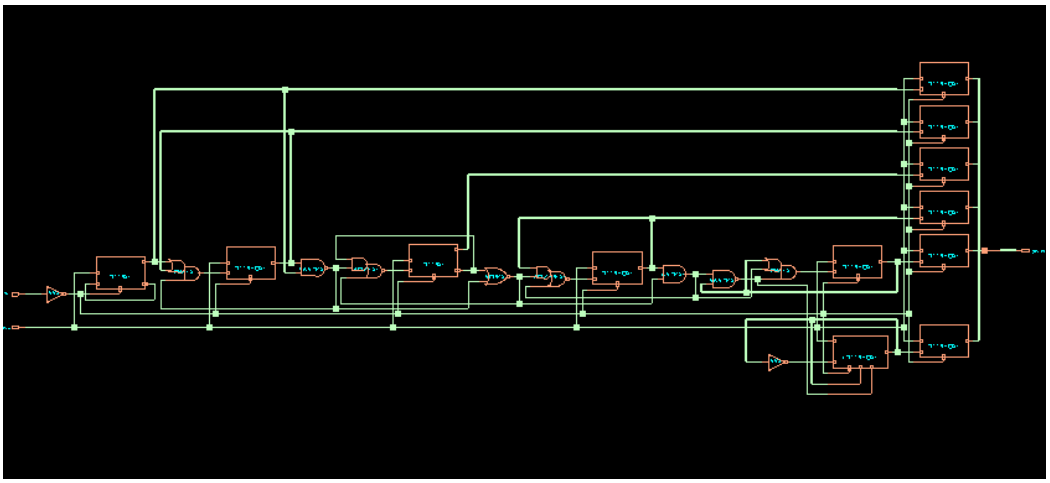
Logical Synthesis

Detailed schematic generated after Synthesis:

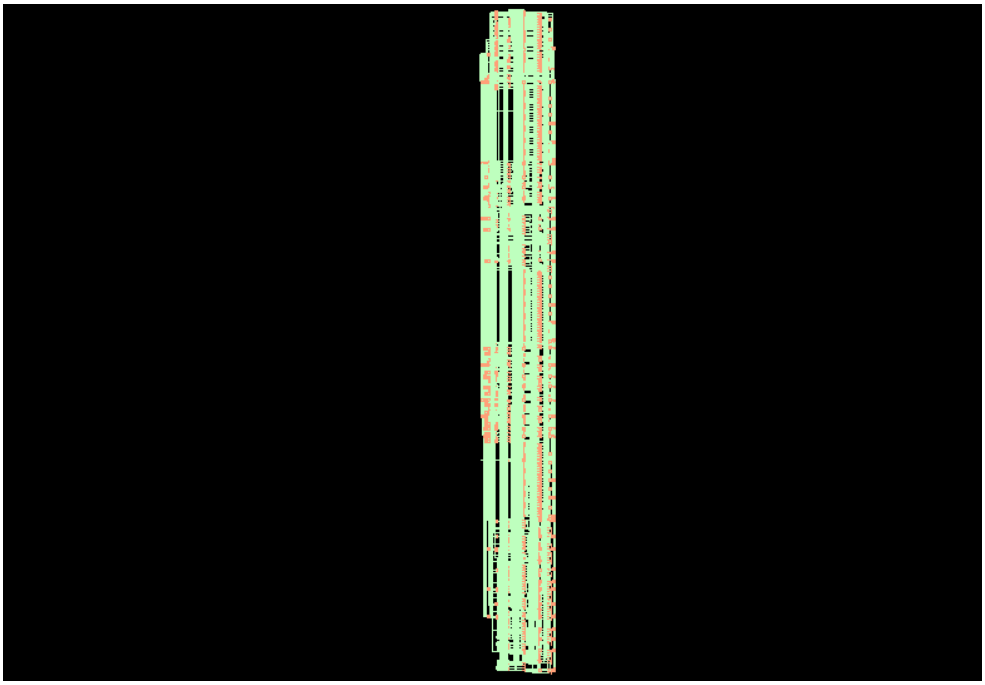
Input Flip Flop:



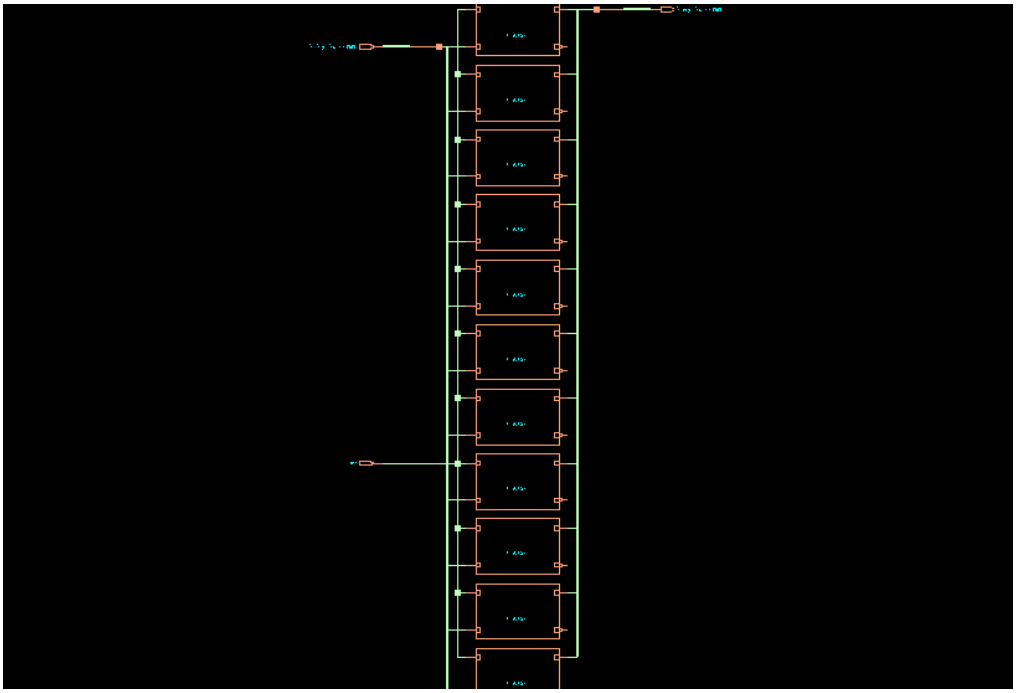
Program Counter:



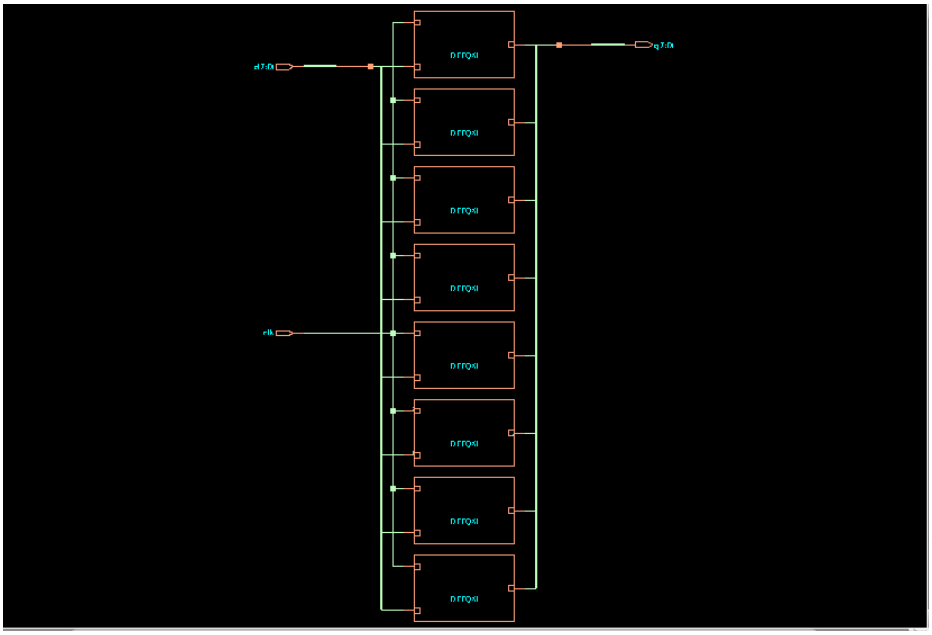
Instruction Memory:



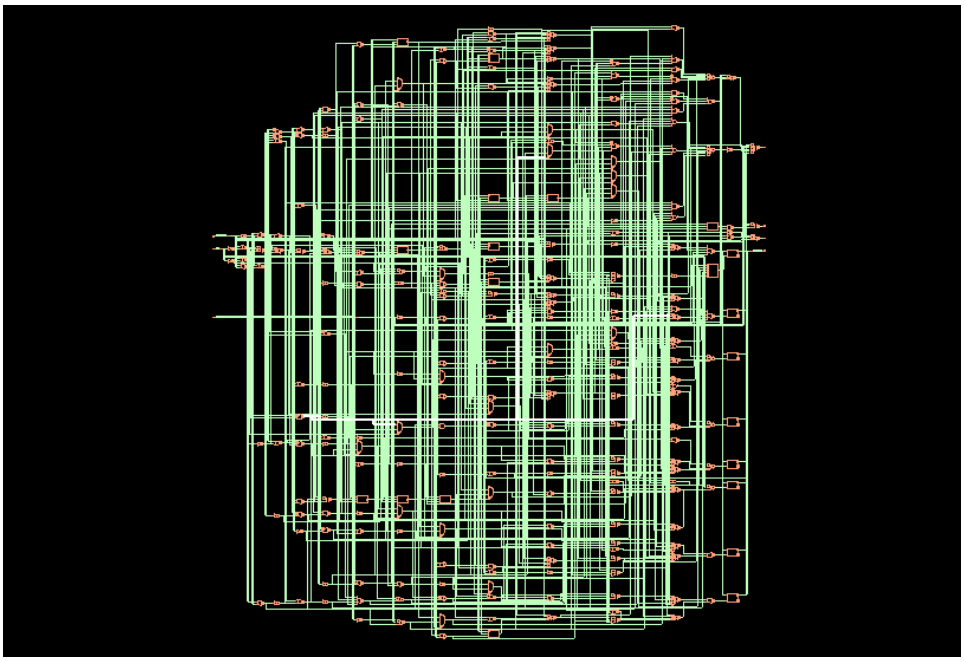
Instruction Register:



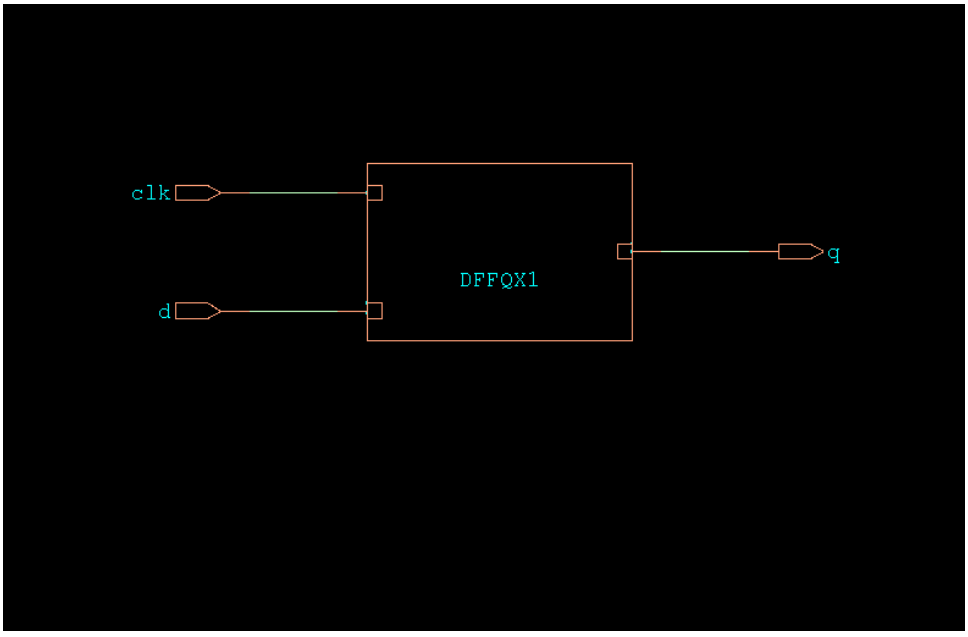
Input 8-bit Flip Flop to ALU:



Arithmetic Logical Unit (ALU):



Output Flip Flops:



(a) Synthesis for minimum area:

How did we arrive at the minimum area?

The below describes the variation in the area obtained by varying the clock period. Timing constraints were kept highly relaxed during the synthesis.

Clock Period (ns)	Area (µm2)	Slack (ps)
1	19898	12
1.5	19902	484
3	19914	1967

The trend of the clock period with that of the area is inversely proportional. The more we increase the clock's time, i.e., the more relaxed the timing, the less area there will be. So, for the constraint, a clock period of 3ns is used for the minimum area because Slack is longer, and it will help us analyze. Also, since timing is relaxed, i.e., time is traded off for area, the slacks increase and become more positive.

TCL file used for synthesis:

```
set_attribute lib_search_path "/home/ahmad23209/Desktop/VDF/"
set_attribute hdl_search_path "/home/ahmad23209/Desktop/VDF/"
set_attr library "/home/ahmad23209/Desktop/VDF/fast.lib"

read_hdl vdf.v
elaborate

read_sdc Min_area.sdc
synthesize -to_mapped -effort medium
write_hdl > HDL_min_Netlist.v
write_sdc > constraints.sdc
write_script > synthesis.g
report_timing > synthesis_timing_report.rep
report_power > synthesis_power_report.rep
report_gates > synthesis_cell_report.rep
report_area > synthesis_area_report.rep
gui_show
```

Constraint (.sdc) file:

```
create_clock -name CLK -period 3 [get_ports clk]
set_clock_transition  0.4 [get_clocks CLK]
#set_clock_transition  0.4 [get_ports clk ]
set_input_delay -clock CLK 0.4 [get_ports rst]
set_input_delay -clock CLK 0.4 [get_ports alu_in1]
set_input_delay -clock CLK 0.4 [get_ports ff_pc_rst]
set_output_delay -clock CLK 0.2 [get_ports ff_zero]
set_output_delay -clock CLK 0.2 [get_ports ff_carry]
set_output_delay -clock CLK 0.2 [get_ports ff_negative]
set_output_delay -clock CLK 0.2 [get_ports alu_result]
set_load 0.1 [get_ports ff_zero]
set_load 0.1 [get_ports ff_carry]
set_load 0.1 [get_ports ff_negative]
set_load 0.1 [get_ports alu_result]

# Specify that the memory should be implemented using flip-flops
set_attribute -library SEQUENTIAL -design {mem[*]} ramstyle registers

# Optimize memory initialization process
set_attribute -name INIT -value "11'b10110010010" [get_cells -hierarchical -filter {name =~
"mem[*]"}]

# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_1_seq -value false [current_design]
# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_0_seq -value false [current_design]

# Set 'preserve' instance attribute to 'true'
set_attribute -instance * -name preserve true
```

Area report:

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 19.13-s073_1
  Generated on:           Mar 23 2024  03:59:11 pm
  Module:                 top8_module
  Technology library:     fast
  Operating conditions:   fast (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================

     Instance         Module     Cell Count  Cell Area  Net Area   Total Area  Wireload
-------------------------------------------------------------------------------------------
top8_module                          1664  19914.796     0.000    19914.796    <none> (D)
  mem_inst         instr_memory      1361  17952.154     0.000    17952.154    <none> (D)
  alu_inst         alu                247   1160.328     0.000     1160.328    <none> (D)
  pc_inst          pc                  22    295.191     0.000      295.191    <none> (D)
  ir_inst          instr_reg           11    158.192     0.000      158.192    <none> (D)
  decoder_inst     decoder             11    158.192     0.000      158.192    <none> (D)
  ff_alu_in1       ff_8bit              8    127.159     0.000      127.159    <none> (D)
  ff_zero_inst     ff_153               1     15.895     0.000       15.895    <none> (D)
  ff_pc_rst_inst   ff                    1     15.895     0.000       15.895    <none> (D)
  ff_negative_inst ff_152               1     15.895     0.000       15.895    <none> (D)
  ff_carry_inst    ff_154               1     15.895     0.000       15.895    <none> (D)

 (D) = wireload is default in technology library
```

Area Optimization in Processor Design:

We aimed to minimize the area (physical size) of the TOP module in our microprocessor design. To achieve this, we employed a strategy of relaxing the timing constraints. This essentially means we allowed the clock to run slower (increased the clock period) until the area of the circuit reached its minimum size.

Analysis of Results:

By examining the Standard Delay File (SDF) used for timing analysis, we identified a clock period of 3 nanoseconds (ns) as the point where the area of the TOP module became minimized. However, there's a trade-off here: a slower clock reduces processing speed.

Cell Count: The synthesized design currently has a cell count of 1664, which surpasses the target of utilizing only 500 logic gates. This indicates a need for further optimization to meet the specified gate count.

Minimum Area: At the 3ns clock period, the design achieved a minimal area of 19914 square micrometers (μm²). This is the smallest area across all three netlists we considered, and it leverages cells from a 90nm technology library.

Timing report:

```
          Pin                    Type      Fanout Load Slew Delay Arrival
                                             (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------
(clock CLK)                     launch                              0 R
(Min_area.sdc_line_4)           ext delay                        +400     400 F
rst                             in port         3  6.5    0    +0     400 F
mem_inst/rst
  g17947/B                                                       +0     400
  g17947/Y                      NOR2BX1         5 12.3   66   +49     449 R
  g17936/A                                                       +0     449
  g17936/Y                      NAND2XL        16 28.8  152  +117     566 F
  g17854/B                                                       +0     566
  g17854/Y                      NOR2XL         11 18.7  174  +163     729 R
  g17572/A0                                                      +0     729
  g17572/Y                      AOI22XL         1  1.6   57   +23     752 F
  g16904/A                                                       +0     752
  g16904/Y                      NAND4XL         1  3.0   36   +38     790 R
  g16656/C0                                                      +0     790
  g16656/Y                      AOI211X1        1  1.6   82    +9     800 F
  g16641/A                                                       +0     800
  g16641/Y                      NAND4XL         1  1.7   38   +38     838 R
  g16630/D                                                       +0     838
  g16630/Y                      OR4XL           1  2.9   20   +31     870 R
  g16620/C0                                                      +0     870
  g16620/Y                      AOI221X1        1  1.6   71   +10     880 F
  g16608/A                                                       +0     880
  g16608/Y                      NAND4XL         1  1.7   35   +36     916 R
  mem_instr_out_reg[7]/D  <<<   DFFQX1                         +0     916
  mem_instr_out_reg[7]/CK       setup                   400  +117    1033 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock CLK)                     capture                            3000 R
-------------------------------------------------------------------------
Cost Group   : 'CLK' (path_group 'CLK')
Timing slack :    1967ps
Start-point  : rst
End-point    : mem_inst/mem_instr_out_reg[7]/D
```

The timing report shows that the slack equals 1967 ps because the timing constraints are relaxed. This would be made tighter in the next step (synthesizing for minimum delay) by appropriately editing the constraint file.

Power report:

```
Instance: /top8_module
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     1.81669e-04  1.00792e-02  1.29208e-05  1.02738e-02  92.36%
        latch     6.49584e-06  4.08148e-05  2.70962e-05  7.44068e-05   0.67%
        logic     3.35935e-05  1.23467e-04  8.53939e-05  2.42454e-04   2.18%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  5.33449e-04  5.33449e-04   4.80%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     2.21758e-04  1.02435e-02  6.58860e-04  1.11241e-02 100.01%
  Percentage           1.99%       92.08%        5.92%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Total Power is 11.124 mW

Cell report:

```
  Module:                  top8_module
  Technology library:      fast
  Operating conditions:    fast (balanced_tree)
  Wireload mode:           enclosed
  Area mode:               timing library
==============================================================


   Gate    Instances    Area    Library
----------------------------------------
AND2X1             4    18.166    fast
AO22X1            10    75.690    fast
AO22XL            13    98.397    fast
AOI211X1           8    42.386    fast
AOI211XL           6    31.790    fast
AOI21X1            9    40.873    fast
AOI21XL            3    13.624    fast
AOI221X1          28   211.932    fast
AOI221XL          11    83.259    fast
AOI222XL           7    58.281    fast
AOI22X1            5    30.276    fast
AOI22XL          294  1780.229    fast
AOI2BB1X1          1     6.055    fast
AOI31X1            1     6.055    fast
AOI31XL            3    18.166    fast
AOI32X1            2    13.624    fast
CLKINVX1          37    84.016    fast
DFFQX1            23   365.583    fast
DFFRHQX1           9   183.927    fast
DFFRX1             2    43.900    fast
INVX1             18    40.873    fast
INVXL              3     6.812    fast
MX2X1              1     6.812    fast
MXI2XL            15    90.828    fast
NAND2BX1          13    59.038    fast
NAND2BXL           2     9.083    fast
NAND2XL           92   278.539    fast
NAND3X1            3    13.624    fast
NAND3XL            2     9.083    fast
NAND4BXL          13    88.557    fast
NAND4XL           81   429.162    fast
```

```
NOR2BX1              3     13.624    fast
NOR2BXL              3     13.624    fast
NOR2XL             139    420.836    fast
NOR3BXL              1      6.055    fast
NOR3XL               1      4.541    fast
NOR4BX1              2     13.624    fast
NOR4X1               2     12.110    fast
OA21X1               1      6.812    fast
OA21XL               2     13.624    fast
OAI21X1             10     45.414    fast
OAI21XL              5     22.707    fast
OAI221X1             1      7.569    fast
OAI221XL             1      7.569    fast
OAI222XL             1      8.326    fast
OAI22X1              3     18.166    fast
OAI22XL             10     60.552    fast
OAI2BB1X1            6     31.790    fast
OAI2BB1XL            1      5.298    fast
OAI31X1              2     12.110    fast
OR2XL                1      4.541    fast
OR4X1                1      6.812    fast
OR4XL               13     88.557    fast
SDFFQX1            143   2922.391    fast
SDFFQXL            561  11464.764    fast
SDFFRHQX1            1     24.978    fast
TLATX1              30    431.433    fast
XNOR2X1              1      8.326    fast
-----------------------------------------
total             1664  19914.796


     Type       Instances    Area   Area %
-----------------------------------------
sequential           769 15436.975   77.5
inverter              58   131.701    0.7
logic                837  4346.120   21.8
physical_cells         0     0.000    0.0
-----------------------------------------
total                1664 19914.796  100.0
```

The total minimum Number of cells is 1664

Analysis and impact of altering the constraints on Quality of Results (QoR):

We prioritized minimizing the area of the TOP module while relaxing timing constraints. This means we allowed the clock to run slower (increased the clock period) until the area stabilized. Through multiple trials, we found a clock period of [insert specific period] ns offered the best balance between area and timing.

Impact of Clock Period:

Area: Increasing the clock period had minimal impact on the area within a specific range. This suggests focusing on other factors for further area reduction.

Power Consumption: A slower clock (higher period) reduces the processor's frequency. Since dynamic power consumption is directly proportional to frequency, this translates to lower power usage.

Constraint Variations: Adjusting constraints like clock latency, input delay, and output delay didn't significantly affect the area. This means optimizing these may not be the most effective approach for area reduction in this case.

(b) Synthesis for best timing:

TCL file used for synthesis:

syn.tcl - Notepad

File Edit Format View Help

```
set_attribute lib_search_path "/home/abdullah23150/Synthesis/vdf_proj"
set_attribute hdl_search_path "/home/abdullah23150/Synthesis/vdf_proj"
set_attr library "/home/abdullah23150/vdf_proj/Synthesis/fast.lib"
read_hdl top8.v
#set_top_module PROCESSOR
elaborate
read_sdc Min_area.sdc
synthesize -to_mapped -effort medium
#write_sdf -timescale ns -nonegchecks -recrem split -edges check_edge>report_Mid/tight_area.sdf
set_attribute hdl_preserve_unused_register true
set_attribute delete_unloaded_seqs false
set_attribute optimize_constant_0_flops false
set_attribute optimize_constant_1_flops false
set_attribute optimize_constant_latches false
set_attribute optimize_constant_feedback_seqs false
#set_attribute prune_unsued_logic false
write_hdl -lec > report_Mid/HDL_tight_Netlist.v
write_do_lec -revised_design report_Mid/HDL_tight_Netlist.v -logfile report_Mid/dofile1.lec.log >./dofile1.lec.do
write_sdc > report_Mid/constraints.sdc
write_script > report_Mid/synthesis.g
report_timing > report_Mid/synthesis_timing_report.rep
report_power > report_Mid/synthesis_power_report.rep
report_gates > report_Mid/synthesis_cell_report.rep
report_area > report_Mid/synthesis_area_report.rep

#gui_show
```

Constraint (.sdc) file:

```
create_clock -name CLK -period 0.6 [get_ports clk]
set_clock_transition  0.4 [get_clocks CLK]
#set_clock_transition  0.4 [get_ports clk ]
set_input_delay -clock CLK 0.4 [get_ports rst]
set_input_delay -clock CLK 0.4 [get_ports alu_in1]
set_input_delay -clock CLK 0.4 [get_ports ff_pc_rst]
set_output_delay -clock CLK 0.2 [get_ports ff_zero]
set_output_delay -clock CLK 0.2 [get_ports ff_carry]
set_output_delay -clock CLK 0.2 [get_ports ff_negative]
set_output_delay -clock CLK 0.2 [get_ports alu_result]
set_load 0.1 [get_ports ff_zero]
set_load 0.1 [get_ports ff_carry]
set_load 0.1 [get_ports ff_negative]
set_load 0.1 [get_ports alu_result]

# Specify that the memory should be implemented using flip-flops
set_attribute -library SEQUENTIAL -design {mem[*]} ramstyle registers

# Optimize memory initialization process
set_attribute -name INIT -value "11'b10110010010" [get_cells -hierarchical -filter {name =~ "mem[*]"}]

# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_1_seq -value false [current_design]
# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_0_seq -value false [current_design]

# Set 'preserve' instance attribute to 'true'
set_attribute -instance * -name preserve true
```

Area report:

```
=============================================================
  Generated by:           Genus(TM) Synthesis Solution 19.13-s073_1
  Generated on:           Mar 23 2024  03:03:18 am
  Module:                 top8_module
  Technology library:     fast
  Operating conditions:   fast (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
=============================================================

     Instance         Module     Cell Count  Cell Area  Net Area   Total Area  Wireload
-------------------------------------------------------------------------------------------
top8_module                          2008  21218.178     0.000    21218.178    <none> (D)
  mem_inst         instr_memory      1704  19257.050     0.000    19257.050    <none> (D)
  alu_inst         alu                248   1158.057     0.000     1158.057    <none> (D)
  pc_inst          pc                  22    295.191     0.000      295.191    <none> (D)
  ir_inst          instr_reg           11    158.192     0.000      158.192    <none> (D)
  decoder_inst     decoder             11    158.192     0.000      158.192    <none> (D)
  ff_alu_in1       ff_8bit              8    127.159     0.000      127.159    <none> (D)
  ff_zero_inst     ff_153               1     16.652     0.000       16.652    <none> (D)
  ff_pc_rst_inst   ff                    1     15.895     0.000       15.895    <none> (D)
  ff_negative_inst ff_152               1     15.895     0.000       15.895    <none> (D)
  ff_carry_inst    ff_154               1     15.895     0.000       15.895    <none> (D)

 (D) = wireload is default in technology library
```

The analysis reveals that using cells from the "fast.lib" library resulted in a larger area of 21218.178 µm². This exceeds the acceptable area for the netlist with stricter timing constraints. The optimization tool automatically selects larger cells from a different library to achieve the desired timing performance, but this comes at the cost of increased area. This confirms that load variations have minimal impact on the area in this context.

Timing report:

```
=============================================================
  Generated by:           Genus(TM) Synthesis Solution 19.13-s073_1
  Generated on:           Mar 23 2024  03:03:17 am
  Module:                 top8_module
  Technology library:     fast
  Operating conditions:   fast (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
=============================================================

          Pin                    Type      Fanout  Load Slew Delay Arrival
                                                  (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------------
(clock CLK)                   launch                                    0 R
(Min_area.sdc_line_4)         ext delay                        +400     400 F
rst                           in port         76 310.2    0    +0     400 F
mem_inst/rst
  g22514/A                                                      +0     400
  g22514/Y                    CLKINVX20        9  26.6    5    +8     408 R
  g22439/B                                                      +0     408
  g22439/Y                    CLKAND2X12      16 204.9   52   +59     466 R
  g22315/B                                                      +0     466
  g22315/Y                    NAND2X4         11  43.6   63   +36     502 F
  mem_reg[49][10]1354/SE <<<  SDFFQX1                           +0     502
  mem_reg[49][10]1354/CK      setup                    400  +154     656 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock CLK)                   capture                                 651 R
-------------------------------------------------------------------------------
Cost Group   : 'CLK' (path_group 'CLK')
Timing slack :      -5ps (TIMING VIOLATION)
Start-point  : rst
End-point    : mem_inst/mem_reg[49][10]1354/SE
```

The strict timing constraints result in a slight negative slack of -5 picoseconds (ps). This indicates the design is pushing the limits of its timing capabilities with a clock period of 0.651 ns. In simpler terms, the data propagation times are very close to the clock cycle. To prioritize timing performance, we've chosen the clock period where the first negative slack appears. This ensures the design meets the timing requirements with minimal

margin for error. The previous synthesis runs focused on minimizing area, but in this case, timing is paramount.

Power report:

```
Instance: /top8_module
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     1.73005e-04  4.81384e-02  3.95346e-04  4.87068e-02  92.35%
        latch     6.49584e-06  2.09008e-04  1.28808e-04  3.44312e-04   0.65%
        logic     5.86301e-05  6.95746e-04  4.73330e-04  1.22771e-03   2.33%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  2.46535e-03  2.46535e-03   4.67%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     2.38131e-04  4.90432e-02  3.46283e-03  5.27441e-02 100.00%
  Percentage           0.45%       92.98%        6.57%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Total Power is 52.7441 mW

Cell report:

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 19.13-s073_1
  Generated on:           Mar 23 2024  03:03:18 am
  Module:                 top8_module
  Technology library:     fast
  Operating conditions:   fast (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


   Gate     Instances    Area    Library
-----------------------------------------
AND2X1             34    154.408    fast
AND4XL              1      6.812    fast
AO22X1              1      7.569    fast
AO22XL              1      7.569    fast
AOI211X1            4     21.193    fast
AOI211XL            7     37.088    fast
AOI21X1             8     36.331    fast
AOI21XL             5     22.707    fast
AOI221X1           11     83.259    fast
AOI221XL           37    280.053    fast
AOI222X1            1      9.083    fast
AOI222XL           18    149.866    fast
AOI22X1            52    314.870    fast
AOI22XL           233   1410.862    fast
AOI2BB1X1           1      6.055    fast
AOI2BB2X1           1      9.083    fast
AOI32X1             1      6.812    fast
CLKAND2X12          1     23.464    fast
CLKAND2X2          12     63.580    fast
CLKINVX1           61    138.513    fast
CLKINVX2           30    113.535    fast
CLKINVX20           1     19.679    fast
DFFHQX1             1     16.652    fast
DFFQX1            198   3147.190    fast
DFFRHQX1            9    183.927    fast
DFFRX1              2     43.900    fast
```

DFFRX1	2	43.900	fast
INVX1	19	43.143	fast
INVXL	6	13.624	fast
MX2X1	177	1205.742	fast
MXI2X1	1	6.812	fast
MXI2XL	16	96.883	fast
NAND2BX1	11	49.955	fast
NAND2BXL	2	9.083	fast
NAND2X2	7	42.386	fast
NAND2X4	13	127.916	fast
NAND2X6	3	43.143	fast
NAND2XL	121	366.340	fast
NAND3BX1	1	6.055	fast
NAND3X1	3	13.624	fast
NAND3XL	8	36.331	fast
NAND4BXL	4	27.248	fast
NAND4XL	79	418.566	fast
NOR2BX1	44	199.822	fast
NOR2BXL	3	13.624	fast
NOR2X1	16	60.552	fast
NOR2X2	2	12.110	fast
NOR2X4	1	9.840	fast
NOR2X6	1	12.867	fast
NOR2X8	1	16.652	fast
NOR2XL	103	311.843	fast
NOR3X1	1	4.541	fast
NOR4BBX1	1	8.326	fast
NOR4X1	4	24.221	fast
OA21XL	2	13.624	fast
OA22X1	1	7.569	fast
OAI211XL	4	21.193	fast
OAI21X1	10	45.414	fast
OAI21XL	7	31.790	fast
OAI221X1	1	7.569	fast
OAI221XL	1	7.569	fast
OAI222XL	2	16.652	fast
OAI22X1	1	6.055	fast
OAI22XL	5	30.276	fast
OAI2BB1X1	11	58.281	fast
OAI2BB1XL	14	74.176	fast

```
OAI2BB1XL          14     74.176    fast
OAI31XL             1      6.055    fast
OAI32X1             1      6.812    fast
OR2X1               2      9.083    fast
OR4X1               1      6.812    fast
OR4XL               5     34.060    fast
SDFFQX1           122   2493.229    fast
SDFFQX2            56   1229.206    fast
SDFFQXL           350   7152.705    fast
SDFFRHQX1           1     24.978    fast
TLATX1             30    431.433    fast
XNOR2X1             1      8.326    fast
----------------------------------------
total            2008  21218.178


     Type      Instances    Area   Area %
----------------------------------------
sequential           769 14723.219   69.4
inverter             117   328.495    1.5
logic               1122  6166.464   29.1
physical_cells         0     0.000    0.0
----------------------------------------
total                2008 21218.178  100.0
```

The total number of cells is 2008

Analysis and Impact of Altering Constraints on Quality of Results (QoR)

Trade-offs Between Timing, Area, and Power:

Negative Slack and Area Impact: When timing constraints become very strict, "negative slack" can occur. This means data propagation times are too close to the clock cycle, potentially causing errors. The synthesis tool automatically selects larger, stronger cells from the library to address this. However, these larger cells come at the cost of increased area, exceeding the minimum achievable size.

Impact of Clock Period on Slack: Reducing the clock period (increasing frequency) squeezes the time available for data to settle before the next clock tick. This directly leads to worsening slack (potentially becoming negative) because the data has less time to stabilize.

Power Consumption and Frequency: Dynamic power consumption in a processor is directly proportional to its operating frequency. As you decrease the clock period (increasing frequency), the power consumption also increases. It is important to consider this relationship when balancing timing and power efficiency.

(c) Synthesize for timing constraints that are between (a) and (b):

TCL file used for synthesis:

syn.tcl - Notepad

File Edit Format View Help

```
set_attribute lib_search_path "/home/abdullah23150/Synthesis/vdf_proj"
set_attribute hdl_search_path "/home/abdullah23150/Synthesis/vdf_proj"
set_attr library "/home/abdullah23150/vdf_proj/Synthesis/fast.lib"
read_hdl top8.v
#set_top_module PROCESSOR
elaborate
read_sdc Min_area.sdc
synthesize -to_mapped -effort medium
#write_sdf -timescale ns -nonegchecks -recrem split -edges check_edge>report_Mid/tight_area.sdf
set_attribute hdl_preserve_unused_register true
set_attribute delete_unloaded_seqs false
set_attribute optimize_constant_0_flops false
set_attribute optimize_constant_1_flops false
set_attribute optimize_constant_latches false
set_attribute optimize_constant_feedback_seqs false
#set_attribute prune_unsued_logic false
write_hdl -lec > report_Mid/HDL_tight_Netlist.v
write_do_lec -revised_design report_Mid/HDL_tight_Netlist.v -logfile report_Mid/dofile1.lec.log >./dofile1.lec.do
write_sdc > report_Mid/constraints.sdc
write_script > report_Mid/synthesis.g
report_timing > report_Mid/synthesis_timing_report.rep
report_power > report_Mid/synthesis_power_report.rep
report_gates > report_Mid/synthesis_cell_report.rep
report_area > report_Mid/synthesis_area_report.rep

#gui_show
```

Constraint (.sdc) file:

```
create_clock -name CLK -period 1.5 [get_ports clk]
set_clock_transition  0.4 [get_clocks CLK]
#set_clock_transition  0.4 [get_ports clk ]
set_input_delay -clock CLK 0.4 [get_ports rst]
set_input_delay -clock CLK 0.4 [get_ports alu_in1]
set_input_delay -clock CLK 0.4 [get_ports ff_pc_rst]
set_output_delay -clock CLK 0.2 [get_ports ff_zero]
set_output_delay -clock CLK 0.2 [get_ports ff_carry]
set_output_delay -clock CLK 0.2 [get_ports ff_negative]
set_output_delay -clock CLK 0.2 [get_ports alu_result]
set_load 0.1 [get_ports ff_zero]
set_load 0.1 [get_ports ff_carry]
set_load 0.1 [get_ports ff_negative]
set_load 0.1 [get_ports alu_result]

# Specify that the memory should be implemented using flip-flops
set_attribute -library SEQUENTIAL -design {mem[*]} ramstyle registers

# Optimize memory initialization process
set_attribute -name INIT -value "11'b10110010010" [get_cells -hierarchical -filter {name =~ "mem[*]"}]

# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_1_seq -value false [current_design]
# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_0_seq -value false [current_design]

# Set 'preserve' instance attribute to 'true'
set_attribute -instance * -name preserve true
```

Area report:

```
====================================================
  Generated by:           Genus(TM) Synthesis Solution 19.13-s073_1
  Generated on:           Mar 23 2024  04:21:35 pm
  Module:                 top8_module
  Technology library:     fast
  Operating conditions:   fast (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================

    Instance         Module     Cell Count  Cell Area  Net Area   Total Area  Wireload
-------------------------------------------------------------------------------------
top8_module                         1674  19902.685     0.000    19902.685    <none> (D)
  mem_inst        instr_memory      1370  17946.099     0.000    17946.099    <none> (D)
  alu_inst        alu                248   1154.273     0.000     1154.273    <none> (D)
  pc_inst         pc                  22    295.191     0.000      295.191    <none> (D)
  ir_inst         instr_reg           11    158.192     0.000      158.192    <none> (D)
  decoder_inst    decoder             11    158.192     0.000      158.192    <none> (D)
  ff_alu_in1      ff_8bit              8    127.159     0.000      127.159    <none> (D)
  ff_zero_inst    ff_153               1     15.895     0.000       15.895    <none> (D)
  ff_pc_rst_inst  ff                   1     15.895     0.000       15.895    <none> (D)
  ff_negative_inst ff_152              1     15.895     0.000       15.895    <none> (D)
  ff_carry_inst   ff_154               1     15.895     0.000       15.895    <none> (D)

 (D) = wireload is default in technology library
```

The total intermediate area is 19902.685 µm²

Timing report:

```
  Module:                 top8_module
  Technology library:     fast
  Operating conditions:   fast (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
 ====================================================

          Pin                   Type      Fanout Load Slew Delay Arrival
                                             (fF) (ps)  (ps)   (ps)
 -------------------------------------------------------------------------
 (clock CLK)                   launch                                 0 R
 (Min_area.sdc_line_4)         ext delay                   +400     400 F
 rst                           in port         3  5.4    0    +0     400 F
 mem_inst/rst
   g21241/AN                                                 +0     400
   g21241/Y                    NAND2BX1       13 26.7   82   +76     476 F
   g21234/A                                                  +0     476
   g21234/Y                    CLKINVX1        3  5.1   32   +35     511 R
   g21231/A                                                  +0     511
   g21231/Y                    NAND2XL        16 28.8  152  +113     624 F
   g21158/B                                                  +0     624
   g21158/Y                    NOR2XL         11 22.0  200  +182     806 R
   g19897/B1                                                 +0     806
   g19897/Y                    AOI221X1        1  2.9   71   +19     824 F
   g19892/B0                                                 +0     824
   g19892/Y                    OAI2BB1X1       1  2.9   28   +29     854 R
   g19888/C0                                                 +0     854
   g19888/Y                    AOI221X1        1  1.7   71   +11     865 F
   g19882/C                                                  +0     865
   g19882/Y                    NAND4BXL        1  1.7   32   +32     897 R
   mem_instr_out_reg[6]/D  <<<  DFFQX1                       +0     897
   mem_instr_out_reg[6]/CK      setup                   400  +120    1016 R
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 (clock CLK)                   capture                             1500 R
 -------------------------------------------------------------------------
 Cost Group   : 'CLK' (path_group 'CLK')
 Timing slack :     484ps
 Start-point  : rst
 End-point    : mem_inst/mem_instr_out_reg[6]/D
```

Intermediate timing slack is 484ps

Power report:

```
Instance: /top8_module
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     1.80121e-04  2.01530e-02  2.60794e-05  2.03592e-02  92.54%
       latch     6.49584e-06  8.20699e-05  5.42372e-05  1.42803e-04   0.65%
       logic     3.21886e-05  2.28431e-04  1.64920e-04  4.25540e-04   1.93%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  1.07262e-03  1.07262e-03   4.88%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     2.18805e-04  2.04635e-02  1.31786e-03  2.20002e-02 100.00%
  Percentage           0.99%       93.02%        5.99%      100.00% 100.00%
  ------------------------------------------------------------------------
```

The total intermediate power is 22.0002 mW

Cell report:

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 19.13-s073_1
  Generated on:           Mar 23 2024  04:21:35 pm
  Module:                 top8_module
  Technology library:     fast
  Operating conditions:   fast (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


   Gate    Instances    Area    Library
----------------------------------------
AND2X1             2     9.083    fast
AO22X1            10    75.690    fast
AOI211X1           1     5.298    fast
AOI211XL          13    68.878    fast
AOI21X1            5    22.707    fast
AOI21XL            1     4.541    fast
AOI221X1          10    75.690    fast
AOI221XL          13    98.397    fast
AOI222XL          11    91.585    fast
AOI22X1            8    48.442    fast
AOI22XL          316  1913.443    fast
AOI2BB1X1          1     6.055    fast
CLKINVX1          29    65.850    fast
DFFQX1            23   365.583    fast
DFFRHQX1           9   183.927    fast
DFFRX1             2    43.900    fast
INVX1             20    45.414    fast
INVXL              2     4.541    fast
MXI2XL            18   108.994    fast
NAND2BX1          14    63.580    fast
NAND2BXL           2     9.083    fast
NAND2XL           86   260.374    fast
NAND3BX1           2    12.110    fast
NAND3X1            2     9.083    fast
NAND3XL            4    18.166    fast
NAND4BXL           7    47.685    fast
NAND4XL           89   471.549    fast
```

```
NAND4XL             89    471.549    fast
NOR2BX1             10     45.414    fast
NOR2BXL              4     18.166    fast
NOR2XL             152    460.195    fast
NOR3BX1              1      6.055    fast
NOR3X1               4     18.166    fast
NOR4BXL              1      6.812    fast
NOR4X1               3     18.166    fast
NOR4XL               3     18.166    fast
OA21XL               2     13.624    fast
OAI211XL             4     21.193    fast
OAI21X1             10     45.414    fast
OAI21XL              7     31.790    fast
OAI221XL             1      7.569    fast
OAI222XL             3     24.978    fast
OAI22X1              1      6.055    fast
OAI22XL              5     30.276    fast
OAI2BB1X1           13     68.878    fast
OAI2BB1XL            4     21.193    fast
OAI33X1              1      8.326    fast
OR2X1                4     18.166    fast
OR4XL                6     40.873    fast
SDFFQXL            704  14387.155    fast
SDFFRHQX1            1     24.978    fast
TLATX1              30    431.433    fast
----------------------------------------
total             1674  19902.685


     Type      Instances    Area   Area %
------------------------------------------
sequential            769 15436.975   77.6
inverter               51   115.806    0.6
logic                 854  4349.904   21.9
physical_cells          0     0.000    0.0
------------------------------------------
total                1674 19902.685  100.0
```

The total intermediate Number of cells are 1674

Impact of Constraint Files on Synthesis:

The table summarizes the results of using different constraint files (RTL and TCL) for synthesis. As expected, varying constraints lead to different outcomes regarding area (size), timing slack, power consumption, and cell count. This variation occurs because the synthesis tool selects cells from the library based on the specified constraints. It prioritizes optimizing the parameters you define (e.g., area or timing) while balancing other factors. For example, as we discussed earlier, a shorter clock period (faster design) leads to reduced slack (increased risk of errors) and higher power consumption.

Key Findings

Uncertainty: Increasing uncertainty (variation in signal arrival times) leads to a stricter Required Time (RT) for data paths, ultimately reducing slack. However, it doesn't affect area or power since no cell changes occur, and the uncertainty variation is relative to the clock edge, not affecting power levels.

Input Delay: Higher input delay on the critical path (longest data path) tightens slack, prompting the tool to utilize larger cells with improved timing characteristics. This expands the design area, but power consumption remains constant. Prolonged input delay increases the overall data path delay, leaving less time for logic operations or delaying data arrival, reducing slack.

Network Latency: Network latency (delay within a logic block) doesn't affect slack, area, or power when uniformly added to both Required Time (RT) and Arrival Time (AT). This is because the latency is accounted for in both values, canceling out its impact on the slack calculation. Similarly, area and power remain unchanged due to the balanced addition of latency.

Clock Transition: Increased clock transition time (time taken for the clock signal to rise or fall) degrades slack. With a longer transition, the setup time requirement becomes more stringent, reducing the time for data to settle before the clock triggers a new operation. While the area remains constant, power consumption increases. This is because longer transition times lead to increased periods where both NMOS and PMOS transistors are partially ON, resulting in higher short-circuit power dissipation.

Input Transition: Similar to clock transition, increased input transition (time taken for an input signal to rise or fall) decreases slack while maintaining a constant area. Extended transition delays the signal settling at the flip-flop's data pin, demanding more time to meet the setup requirement. This reduces slack. Power consumption increases due to longer NMOS and PMOS ON states, leading to higher short-circuit power dissipation.

Output Delay: Increasing output delay on a path worsens slack. Careful management is needed to prevent setup and hold time violations. Longer output delay extends the overall data path delay or arrival time, reducing the available time for subsequent operations. However, area and power are not directly impacted by output delay.

Logical Equivalence Checking

Tools and Libraries for Logical Equivalence Checking:

The logical equivalence tool utilized is Conformal, while the library employed is the 90 nm slow.lib.

Invocation Command:

To activate the tool, the following command sequence is employed:

tcsh

Source /cadence/cshrc

lec -lpgxl -dofile dofile1.lec..do

The equv.do file is crucial for initiating the Conformal tool.

Design Verification and Formal Verification:

Simulation is a common method in the design verification process, employing various test vectors. However, this approach presents a higher likelihood of errors and requires a significant increase in test cases with rising input numbers. Additionally, overlooking certain test cases is plausible, leading to potential errors. To mitigate such risks, formal verification offers an alternative method by covering all test cases through mathematical reasoning and equations.

Formal Verification Methods:

Two primary formal verification methods are employed:

1. Model checking

2. Equivalence checking

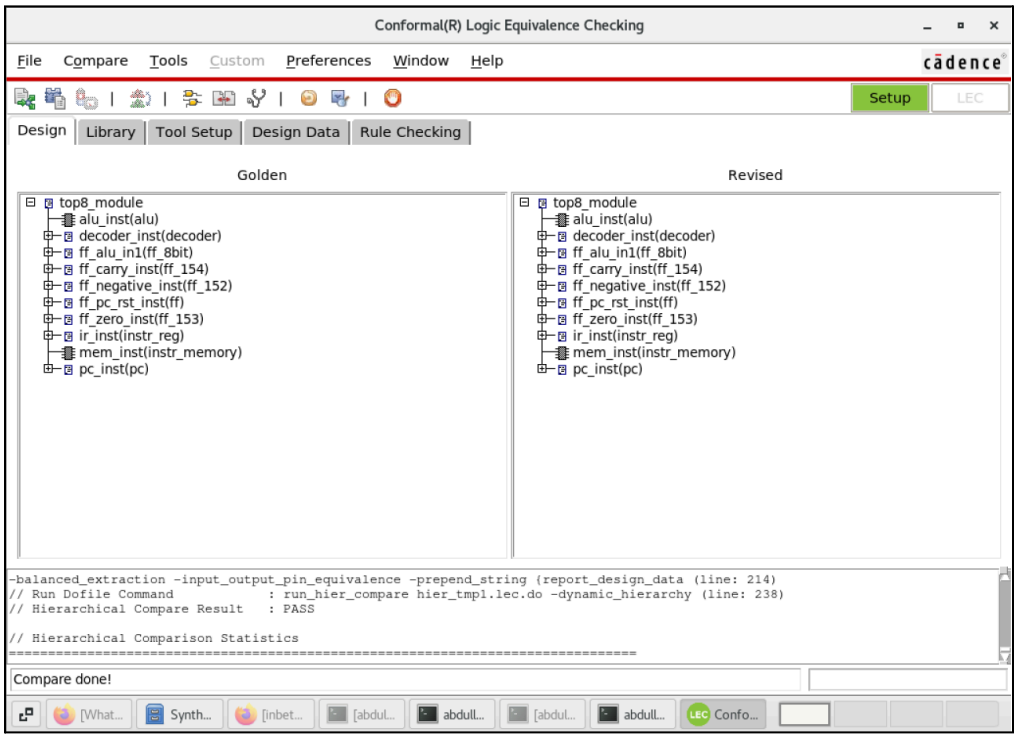
Equivalence Checking:

Equivalence checking ensures that two representations of the same design demonstrate identical behavior. If discrepancies are identified, debugging is conducted before proceeding to the subsequent stage. Equivalence checking comprises two categories:

1. Sequential Equivalence Checking
2. Combinational Equivalence Checking

Sequential equivalence checking offers a broad approach for comparing two models. Conversely, in combinational equivalence checking, a direct correspondence is established between memory elements or flip-flops and the ports of the two models. Any errors detected during this process halt the operation and indicate non-equivalence.

Case 1: Equivalence checking for minimum area with highly relaxed timing constraint




```
--------------------------------------------------------------------------------
                        Verification Report
--------------------------------------------------------------------------------
Category                                                                  Count
--------------------------------------------------------------------------------
1. Non-standard modeling options used:                                        0
     Tri-stated output:                                   checked
     Revised X signals set to E:                          yes
     Floating signals tied to Z:                          yes
     Command "add clock" for clock-gating:                not used
--------------------------------------------------------------------------------
2. Incomplete verification:                                                   0
     All primary outputs are mapped:                      yes
     Not-mapped DFF/DLAT is detected:                     no
     All mapped points are added as compare points:       yes
     All compared points are compared:                    yes
     User added black box:                                no
     Black box mapped with different module name:         no
     Empty module is not black boxed:                     no
     Command "add ignore outputs" used:                   no
     Always false constraints detected:                   no
     Verified pin-equivalent outputs are unmapped:        no
--------------------------------------------------------------------------------
3. User modification to design:                                               0
     Change gate type:                                    no
     Change wire:                                         no
     Primary input added by user:                         no
--------------------------------------------------------------------------------
4. Conformal Constraint Designer clock domain crossing checks recommended:   1
     Multiple clocks in the design:                       yes *
--------------------------------------------------------------------------------
```

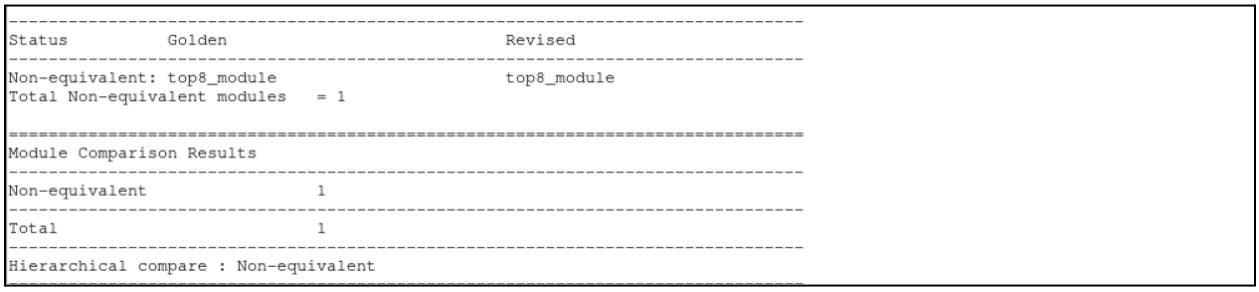
```
--------------------------------------------------------------------------------
5. Design ambiguity:                                                         0
     Duplicate module definition:                          no
     Black box due to undefined cells:                     no
     Golden design has abnormal ratio of unreachable gates:  no
     Ratio of golden unreachable gates:                    24%
     Revised design has abnormal ratio of unreachable gates:  no
     Ratio of revised unreachable gates:                   19%
     All primary input bus ordering is consistent:         yes
     All primary output bus ordering is consistent:        yes
     DFF/DLAT not compared due to disabled clock port(s):    0
--------------------------------------------------------------------------------
6. Compare Results:                                                       PASS
     Total Equivalent modules  = 3
================================================================================
pass
```

● The Verilog code compares with the synthesized netlist, with relaxed timing constraints and minimized area. The comparison results indicate a PASS status.

A part of bad Netlist:

```
AOI31XL g5031(.A0 (n_68), .A1 (n_3), .A2 (n_119), .B0 (n_201), .Y (n_208));
AOI31XL g5032(.A0 (n_17), .A1 (n_3), .A2 (n_117), .B0 (n_202), .Y (n_207));
MXI2XL g5033(.A (n_98), .B (n_99), .S0 (n_195), .Y (n_210));
MXI2XL g5034(.A (n_99), .B (n_98), .S0 (n_200), .Y (n_209));
NAND2XL g5035(.A (n_197), .B (n_156), .Y (n_206));
NAND4XL g5036(.A (n_183), .B (alu_opcode[0]), .C (n_81), .D (n_118), .Y (n_205));
OAI2BB1X1 g5037(.A0N (alu_in2[7]), .A1N (n_5), .B0 (n_195), .Y (n_204));
AOI21XL g5038(.A0 (n_181), .A1 (n_176), .B0 (n_198), .Y (n_203));
OAI22XL g5039(.A0 (n_189), .A1 (n_71), .B0 (n_42), .B1 (n_60), .Y (n_202));
OAI22XL g5040(.A0 (n_179), .A1 (n_71), .B0 (n_42), .B1 (n_55), .Y (n_201));
NOR2XL g5041(.A (n_181), .B (n_176), .Y (n_198));
AOI222XL g5042(.A0 (n_134), .A1 (n_3), .B0 (n_70), .B1 (n_164), .C0 (n_43), .C1 (n_57), .Y (n_197));
OAI2BB1X1 g5043(.A0N (n_47), .A1N (n_186), .B0 (n_49), .Y (n_200));
NAND2BXL g5044(.AN (n_181), .B (n_176), .Y (n_199));
OAI21XL g5045(.A0 (n_18), .A1 (n_132), .B0 (n_192), .Y (n_193));
MXI2XL g5046(.A (n_97), .B (n_96), .S0 (n_186), .Y (n_196));
OAI21X1 g5047(.A0 (alu_in2[6]), .A1 (n_13), .B0 (n_190), .Y (n_195));
MXI2XL g5048(.A (n_97), .B (n_96), .S0 (n_187), .Y (n_194));
NAND2XL g5049(.A (n_182), .B (n_72), .Y (n_192));
NAND4BXL g5050(.AN (n_117), .B (n_120), .C (n_167), .D (n_112), .Y (n_191));
OAI2BB1X1 g5051(.A0N (alu_in2[6]), .A1N (n_13), .B0 (n_187), .Y (n_190));
AOI21XL g5052(.A0 (n_148), .A1 (n_158), .B0 (n_185), .Y (n_189));
AOI221XL g5053(.A0 (n_72), .A1 (n_166), .B0 (n_45), .B1 (n_78), .C0 (n_144), .Y (n_188));
NOR2XL g5054(.A (n_148), .B (n_158), .Y (n_185));
//NAND2XL g5055(.A (n_173), .B (n_174), .Y (n_184));
NOR4BX1 g5056(.AN (n_113), .B (n_119), .C (n_160), .D (n_157), .Y (n_183));
OAI22X1 g5057(.A0 (n_41), .A1 (n_168), .B0 (alu_in2[5]), .B1 (n_6), .Y (n_187));
OAI21X1 g5058(.A0 (n_48), .A1 (n_169), .B0 (n_53), .Y (n_186));
OAI221XL g5059(.A0 (n_112), .A1 (n_18), .B0 (alu_opcode[2]), .B1 (n_94), .C0 (n_171), .Y (n_180));
AOI21XL g5060(.A0 (n_126), .A1 (n_165), .B0 (n_175), .Y (n_179));
OAI22X1 g5061(.A0 (n_11), .A1 (n_170), .B0 (n_47), .B1 (n_151), .Y (n_178));
AOI211XL g5062(.A0 (n_17), .A1 (n_161), .B0 (n_115), .C0 (n_121), .Y (n_177));
MXI2XL g5063(.A (n_92), .B (n_93), .S0 (n_169), .Y (n_182));
MXI2XL g5064(.A (n_93), .B (n_92), .S0 (n_168), .Y (n_181));
```

```
--------------------------------------------------------------------------------
Status          Golden                          Revised
--------------------------------------------------------------------------------
Non-equivalent: top8_module                     top8_module
Total Non-equivalent modules   = 1

================================================================================
Module Comparison Results
--------------------------------------------------------------------------------
Non-equivalent                 1
--------------------------------------------------------------------------------
Total                          1
--------------------------------------------------------------------------------
Hierarchical compare : Non-equivalent
--------------------------------------------------------------------------------
```



A part of bad Netlist:

ALU part of netlist was made bad, one instance was commented out and the LEC failed.

```
 NAND4BXL g16671(.AN (n_558), .B (n_429), .C (n_505), .D (n_430), .Y (n_579));
 NAND4BXL g16672(.AN (n_563), .B (n_462), .C (n_504), .D (n_461), .Y (n_578));
 NAND4BXL g16673(.AN (n_568), .B (n_493), .C (n_503), .D (n_494), .Y (n_577));
 NAND4BXL g16674(.AN (n_553), .B (n_405), .C (n_506), .D (n_407), .Y (n_576));
 NAND4BXL g16675(.AN (n_548), .B (n_237), .C (n_507), .D (n_378), .Y (n_575));
 NAND4BXL g16676(.AN (n_543), .B (n_352), .C (n_508), .D (n_353), .Y (n_574));
 NAND4BXL g16677(.AN (n_538), .B (n_335), .C (n_509), .D (n_336), .Y (n_573));
 NAND4BXL g16678(.AN (n_533), .B (n_305), .C (n_510), .D (n_306), .Y (n_572));
 NAND4BXL g16679(.AN (n_528), .B (n_271), .C (n_511), .D (n_272), .Y (n_571));
 //NAND4BXL g16680(.AN (n_523), .B (n_242), .C (n_512), .D (n_243), .Y (n_570));
 AND4BXL g16680(.AN (n_523), .B (n_242), .C (n_512), .D (n_243), .Y (n_570));
 NAND4BXL g16681(.AN (n_518), .B (n_210), .C (n_513), .D (n_211), .Y (n_569));
 NAND4XL g16885(.A (n_492), .B (n_489), .C (n_490), .D (n_491), .Y (n_568));
 NAND4XL g16886(.A (n_488), .B (n_486), .C (n_484), .D (n_485), .Y (n_567));
 NAND4XL g16887(.A (n_483), .B (n_480), .C (n_481), .D (n_482), .Y (n_566));
 NAND4XL g16888(.A (n_477), .B (n_476), .C (n_478), .D (n_479), .Y (n_565));
 NAND4XL g16889(.A (n_471), .B (n_469), .C (n_468), .D (n_470), .Y (n_564));
 NAND4XL g16890(.A (n_460), .B (n_457), .C (n_458), .D (n_459), .Y (n_563));
 NAND4XL g16891(.A (n_454), .B (n_452), .C (n_456), .D (n_453), .Y (n_562));
 NAND4XL g16892(.A (n_329), .B (n_449), .C (n_450), .D (n_451), .Y (n_561));
 NAND4XL g16893(.A (n_398), .B (n_411), .C (n_448), .D (n_447), .Y (n_560));
 NAND4XL g16894(.A (n_443), .B (n_415), .C (n_440), .D (n_442), .Y (n_559));
 NAND4XL g16895(.A (n_437), .B (n_433), .C (n_436), .D (n_432), .Y (n_558));
```

The change was made in memory; NAND4BXL was replaced with AND4BXL

Equivalence check, hence failed and the result was failed equivalence.

Case 2: Equivalence checking for tight timing and having a slight negative slack.

```
                          Verification Report
--------------------------------------------------------------------------------
Category                                                                  Count
--------------------------------------------------------------------------------
1. Non-standard modeling options used:                                      0
     Tri-stated output:                                  checked
     Revised X signals set to E:                         yes
     Floating signals tied to Z:                         yes
     Command "add clock" for clock-gating:               not used
--------------------------------------------------------------------------------
2. Incomplete verification:                                                 0
     All primary outputs are mapped:                     yes
     Not-mapped DFF/DLAT is detected:                    no
     All mapped points are added as compare points:      yes
     All compared points are compared:                   yes
     User added black box:                               no
     Black box mapped with different module name:        no
     Empty module is not black boxed:                    no
     Command "add ignore outputs" used:                  no
     Always false constraints detected:                  no
     Verified pin-equivalent outputs are unmapped:       no
--------------------------------------------------------------------------------
3. User modification to design:                                             0
     Change gate type:                                   no
     Change wire:                                        no
     Primary input added by user:                        no
--------------------------------------------------------------------------------
4. Conformal Constraint Designer clock domain crossing checks recommended:  1
     Multiple clocks in the design:                      yes *
--------------------------------------------------------------------------------
5. Design ambiguity:                                                        0
     Duplicate module definition:                        no
```

```
--------------------------------------------------------------------------------
5. Design ambiguity:                                                        0
     Duplicate module definition:                        no
     Black box due to undefined cells:                   no
     Golden design has abnormal ratio of unreachable gates:    no
     Ratio of golden unreachable gates:                  24%
     Revised design has abnormal ratio of unreachable gates:   no
     Ratio of revised unreachable gates:                 19%
     All primary input bus ordering is consistent:       yes
     All primary output bus ordering is consistent:      yes
     DFF/DLAT not compared due to disabled clock port(s):       0
--------------------------------------------------------------------------------
6. Compare Results:                                                      PASS
     Total Equivalent modules  = 3
================================================================================
pass
```

● The Verilog code compares with the synthesized netlist, with relaxed timing constraints and minimized area. The comparison results indicate a PASS status.

Case 3: Equivalence checking for timing constraints between the above two.

```
                          Verification Report
--------------------------------------------------------------------------------
Category                                                                  Count
--------------------------------------------------------------------------------
1. Non-standard modeling options used:                                        0
     Tri-stated output:                                  checked
     Revised X signals set to E:                         yes
     Floating signals tied to Z:                         yes
     Command "add clock" for clock-gating:               not used
--------------------------------------------------------------------------------
2. Incomplete verification:                                                   0
     All primary outputs are mapped:                     yes
     Not-mapped DFF/DLAT is detected:                    no
     All mapped points are added as compare points:      yes
     All compared points are compared:                   yes
     User added black box:                               no
     Black box mapped with different module name:        no
     Empty module is not black boxed:                    no
     Command "add ignore outputs" used:                  no
     Always false constraints detected:                  no
     Verified pin-equivalent outputs are unmapped:       no
--------------------------------------------------------------------------------
3. User modification to design:                                               0
     Change gate type:                                   no
     Change wire:                                        no
     Primary input added by user:                        no
--------------------------------------------------------------------------------
4. Conformal Constraint Designer clock domain crossing checks recommended:    1
     Multiple clocks in the design:                      yes *
--------------------------------------------------------------------------------
5. Design ambiguity:                                                          0
     Duplicate module definition:                        no
```

```
--------------------------------------------------------------------------------
5. Design ambiguity:                                                          0
     Duplicate module definition:                        no
     Black box due to undefined cells:                   no
     Golden design has abnormal ratio of unreachable gates:    no
     Ratio of golden unreachable gates:                  24%
     Revised design has abnormal ratio of unreachable gates:   no
     Ratio of revised unreachable gates:                 19%
     All primary input bus ordering is consistent:       yes
     All primary output bus ordering is consistent:      yes
     DFF/DLAT not compared due to disabled clock port(s):    0
--------------------------------------------------------------------------------
6. Compare Results:                                                        PASS
     Total Equivalent modules  = 3
================================================================================
pass
```

● The Verilog code compares with the synthesized netlist, with relaxed timing constraints and minimized area. The comparison results indicate a PASS status.

Static Timing Analysis

Tool Used: Tempus

Command Used: Tempus -nowin

1. Arrival Time: It represents the duration required for the input signal to reach the final sequential element. In this context, Arrival Time (AT) is calculated as the sum of d1 and d2.

2. Required Time: This signifies the timing necessity for the input signal to be accessible at the final sequential element. In this scenario, Required Time (RT) is calculated as the difference between d3 and d4, added to the period of the clock (T).

3. Slack: Slack denotes the difference between RT and AT.

- Slack (setup) = RT - AT
- Slack (hold) = AT-RT

4. Cell Delay: It signifies the delay encountered at the input and the load at the output. The delay at the input could be either a rise or fall delay, while the output load pertains to the transition from rise to fall. These values are specified in the liberty file for each cell under respective indexes.

5. Timing Arcs: These represent the time taken for various input pins to propagate signals to the output port of the same cell. For instance, in an AND cell where A and B are input pins, there will be one timing arc between A and the output and another between B and the output.

6. GBA & PBA: GBA (Graph-Based Analysis) and PBA (Path-Based Analysis) are used for delay calculation. GBA considers worst-case values for delay computation, whereas PBA employs actual values. PBA can be referred to as a more precise method for delay calculation.

1. Arrival Time: It represents the duration required for the input signal to reach the final sequential element. In this context, Arrival Time (AT) is calculated as the sum of d1 and d2.

2. Required Time: This signifies the timing necessity for the input signal to be accessible at the final sequential element. In this scenario, Required Time (RT) is calculated as the difference between d3 and d4, added to the period of the clock (T)**.**

3. Slack: Slack denotes the difference between RT and AT.

- Slack (setup) = RT - AT
- Slack (hold) = AT-RT

4. Cell Delay: It signifies the delay encountered at the input and the load at the output. The delay at the input could be either a rise or fall delay, while the output load pertains to the transition from rise to fall. These values are specified in the liberty file for each cell under respective indexes.

5. Timing Arcs: These represent the time taken for various input pins to propagate signals to the output port of the same cell. For instance, in an AND cell where A and B are input pins, there will be one timing arc between A and the output and another between B and the output.

6. GBA & PBA: GBA (Graph-Based Analysis) and PBA (Path-Based Analysis) are used for delay calculation. GBA considers worst-case values for delay computation,

whereas PBA employs actual values. PBA can be referred to as a more precise method for delay calculation.

Case 1: Synthesis for the minimum area with highly relaxed timing constraint

TCL File:

```
file mkdir reports/sta_after_synthesis/min_delay

read_lib /home/abdullah23150/vdf_proj/STA/min_area_STA/fast.lib
read_verilog /home/abdullah23150/vdf_proj/STA/min_area_STA/HDL_tight_Netlist.v

set_top_module top8_module

read_sdc /home/abdullah23150/vdf_proj/STA/min_area_STA/c.sdc

# Uncomment the following line if you want to specify the wire load model
# set_wire_load_model -name tsnc18wl10

check_timing > reports/sta_after_synthesis/min_delay/check_timing_mid.rpt

report_timing -late -max_paths 5 -retime path_slew_propagation -format {cell pin arc delay slew arrival required} > reports/sta_after_synthesis/min_delay/timing_report_min_delay.rpt

report_analysis_coverage > reports/sta_after_synthesis/min_delay/analysis_coverage_min_delay.rpt
report_analysis_summary > reports/sta_after_synthesis/min_delay/analysis_summary_min_delay.rpt

report_clocks > reports/sta_after_synthesis/min_delay/clocks_min_delay.rpt

report_case_analysis > reports/sta_after_synthesis/min_delay/case_analysis_min_delay.rpt

report_constraints -all_violators > reports/sta_after_synthesis/min_delay/allviolations_min_delay.rpt
```

SDC file:

```
create_clock -name CLK -period 1.5 [get_ports clk]
set_clock_transition  0.4 [get_clocks CLK]
#set_clock_transition  0.4 [get_ports clk ]
set_input_delay -clock CLK 0.4 [get_ports rst]
set_output_delay -clock CLK 0.2 [get_ports ff_zero]
set_output_delay -clock CLK 0.2 [get_ports ff_carry]
set_output_delay -clock CLK 0.2 [get_ports ff_negative]
set_load 0.1 [get_ports ff_zero]
set_load 0.1 [get_ports ff_carry]
set_load 0.1 [get_ports ff_negative]
set_load 0.1 [get_ports alu_result]

# Specify that the memory should be implemented using flip-flops
set_attribute -library SEQUENTIAL -design {mem[*]} ramstyle registers

# Optimize memory initialization process
#set_attribute -name INIT -value "11'b10110010010" [get_cells -hierarchical -filter {name =~ #"mem[*]"}]

# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_1_seq -value false [current_design]
# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_0_seq -value false [current_design]

# Set 'preserve' instance attribute to 'true'
set_attribute -instance * -name preserve true
```

Timing Report:

```
Path 5: MET Setup Check with Pin mem_inst/mem_reg[40][0]1146/CK
Endpoint:   mem_inst/mem_reg[40][0]1146/SE (^) checked with  leading edge of 'CLK'
Beginpoint: rst                            (v) triggered by  leading edge of 'CLK'
Path Groups: {CLK}
Retime Analysis { Data Path-Slew }
Other End Arrival Time          0.000
- Setup                         0.113
+ Phase Shift                   1.500
= Required Time                 1.387
- Arrival Time                  0.919
= Slack Time                    0.468
= Slack Time(original)          0.467
     Clock Rise Edge                      0.000
     + Input Delay                        0.400
     = Beginpoint Arrival Time            0.400
      ------------------------------------------------------------
      Cell      Pin  Arc         Delay  Slew   Arrival  Required
                                                Time     Time
      ------------------------------------------------------------
      -         rst  rst v       -      0.003  0.400    0.868
      NOR2XL    B    -           0.000  0.003  0.400    0.868
      NOR2XL    Y    B v -> Y ^  0.042  0.051  0.441    0.909
      NAND2XL   A    -           0.000  0.051  0.441    0.909
      NAND2XL   Y    A ^ -> Y v  0.025  0.030  0.466    0.934
      NOR2XL    B    -           0.000  0.030  0.466    0.934
      NOR2XL    Y    B v -> Y ^  0.144  0.190  0.610    1.077
      CLKINVX1  A    -           0.000  0.190  0.610    1.077
      CLKINVX1  Y    A ^ -> Y v  0.021  0.063  0.631    1.099
      NOR2XL    B    -           0.000  0.063  0.631    1.099
      NOR2XL    Y    B v -> Y ^  0.288  0.384  0.919    1.387
      SDFFQX1   SE   -           0.000  0.384  0.919    1.387
      ------------------------------------------------------------
```

Setup Slack Time = 0.468 nS

Case 2: Synthesizing for tight timing and having a slight negative slack.

Timing Report (GBA Report)

```
      ------------------------------------------------------------
Path 3: MET Setup Check with Pin mem_inst/mem_reg[1][3]291/CK
Endpoint:   mem_inst/mem_reg[1][3]291/SE (v) checked with  leading edge of 'CLK'
Beginpoint: rst                          (v) triggered by  leading edge of 'CLK'
Path Groups: {CLK}
Retime Analysis { Data Path-Slew }
Other End Arrival Time          0.000
- Setup                         0.142
+ Phase Shift                   1.500
= Required Time                 1.358
- Arrival Time                  0.506
= Slack Time                    0.852
= Slack Time(original)          0.840
     Clock Rise Edge                     0.000
     + Input Delay                       0.400
     = Beginpoint Arrival Time           0.400
      ------------------------------------------------------------
      Cell        Pin  Arc         Delay  Slew   Arrival  Required
                                             Time     Time
      ------------------------------------------------------------
      -           rst  rst v       -      0.003  0.400    1.252
      CLKINVX20   A    -           0.000  0.003  0.400    1.252
      CLKINVX20   Y    A v -> Y ^  0.010  0.008  0.410    1.262
      CLKAND2X12  B    -           0.000  0.008  0.410    1.262
      CLKAND2X12  Y    B ^ -> Y ^  0.060  0.052  0.470    1.322
      NAND2X4     B    -           0.000  0.052  0.470    1.322
      NAND2X4     Y    B ^ -> Y v  0.036  0.063  0.506    1.358
      SDFFQX1     SE   -           0.000  0.063  0.506    1.358
```

Setup Slack Time = 0.852 nS

Case 3: Synthesizing for in-between timing constraints of the above two cases, Intermediate slack and intermediate area.

Timing Report

```
Path 5: MET Setup Check with Pin mem_inst/mem_instr_out_reg[7]/CK
Endpoint:   mem_inst/mem_instr_out_reg[7]/D (^) checked with  leading edge of 'CLK'
Beginpoint: rst                             (v) triggered by  leading edge of 'CLK'
Path Groups: {CLK}
Retime Analysis { Data Path-Slew }
Other End Arrival Time          0.000
- Setup                         0.083
+ Phase Shift                   1.500
= Required Time                 1.417
- Arrival Time                  0.903
= Slack Time                    0.514
= Slack Time(original)          0.513
     Clock Rise Edge                      0.000
     + Input Delay                        0.400
     = Beginpoint Arrival Time            0.400
      ------------------------------------------------------------
      Cell      Pin  Arc          Delay  Slew   Arrival  Required
                                             Time     Time
      ------------------------------------------------------------
      -         rst  rst v        -      0.003  0.400    0.914
      NAND2BX1  AN   -            0.000  0.003  0.400    0.914
      NAND2BX1  Y    AN v -> Y v  0.080  0.082  0.480    0.994
      CLKINVX1  A    -            0.000  0.082  0.480    0.994
      CLKINVX1  Y    A v -> Y ^   0.034  0.032  0.515    1.028
      NAND2XL   A    -            0.000  0.032  0.515    1.028
      NAND2XL   Y    A ^ -> Y v   0.114  0.152  0.628    1.142
      NOR2XL    B    -            0.000  0.152  0.628    1.142
      NOR2XL    Y    B v -> Y ^   0.175  0.190  0.803    1.317
      AOI222XL  A0   -            0.000  0.190  0.803    1.317
      AOI222XL  Y    A0 ^ -> Y v  0.032  0.073  0.835    1.349
      NAND4XL   A    -            0.000  0.073  0.835    1.349
      NAND4XL   Y    A v -> Y ^   0.037  0.036  0.873    1.386
      NOR2XL    B    -            0.000  0.036  0.873    1.386
      NOR2XL    Y    B ^ -> Y v   0.011  0.016  0.883    1.397
      NAND3XL   A    -            0.000  0.016  0.883    1.397
      NAND3XL   Y    A v -> Y ^   0.020  0.019  0.903    1.417
      DFFQX1    D    -            0.000  0.019  0.903    1.417
      ------------------------------------------------------------
```

Setup Slack Time = 0.514 nS

STEP 6: Design for Test

Tool Used: Genus

Commands Used:

tcsh

Source /cadence/cshrc

genus -legacy_ui

Source a.tcl

About:

Design for Testability (DFT) is a process that introduces additional circuits into the design to facilitate testing of the device post-manufacturing. DFT addresses issues of controllability and observability within the circuit, simplifying the testing of deeply embedded devices.

Functioning of a Scan Chain:

The operation of a scan chain involves three stages:

Step 1: Scan In Objective - Test patterns are loaded while the design is in test timing mode. The scan enable (SE) is set to 1, allowing all flip-flops to accept Scan In (SI) as inputs. The scan chain becomes active, and test patterns generated by Automatic Test Pattern Generation (ATPG) can be shifted to the Q pin of the flip-flops over N clock cycles.

Step 2: Capture Objective - The design transitions to functional timing mode, and the response to the test pattern is captured. After N clock cycles, SE transitions to 0, and the D pin inputs of all flip-flops are latched to the Q pin. Primary inputs are also utilized, and any faults result in corruption of the value at the Q pin.

Step 3: Scan Out Objective - The design is returned to test timing mode, and the response to the test pattern is unloaded. At times, this operation also initiates injecting the next test pattern via Scan In. With the output observed at the SO pin, the design proceeds through another N-clock cycle, simultaneously allowing a new pattern to enter the chain.

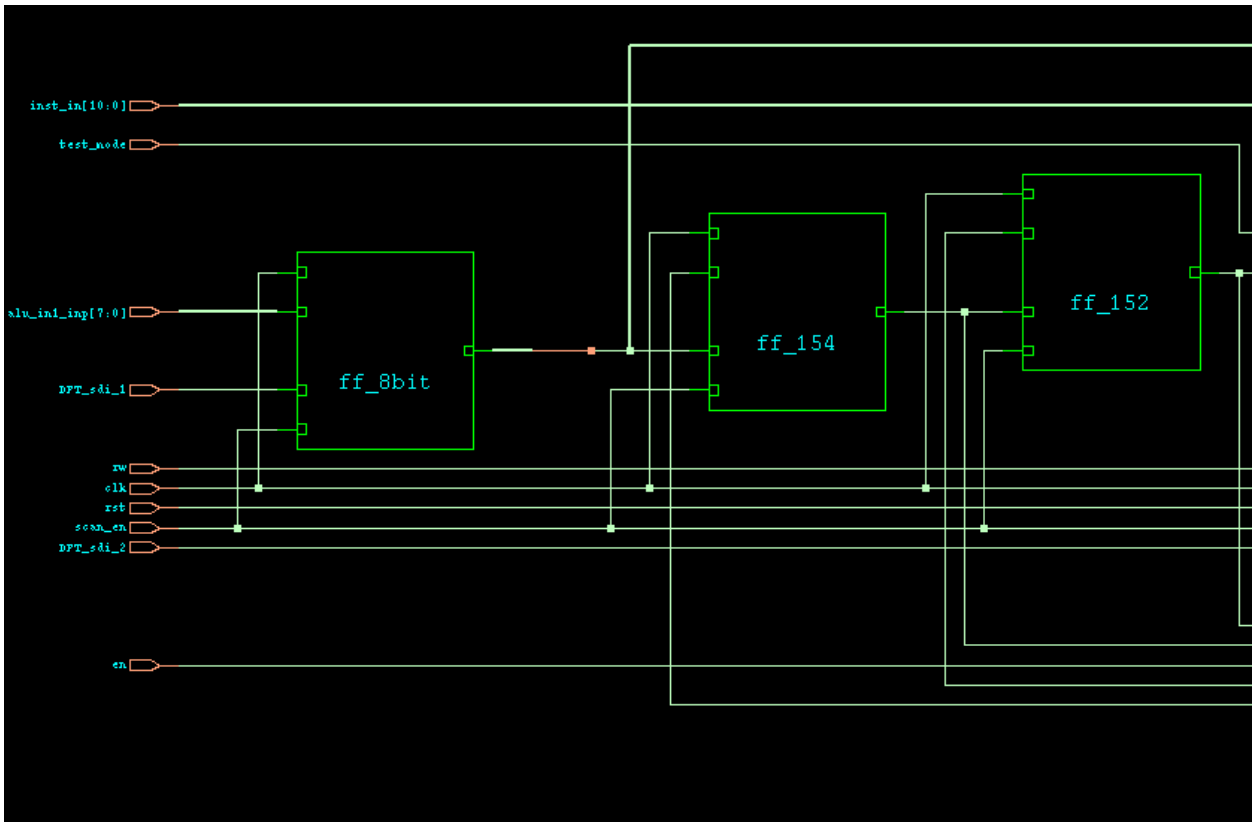
Tcl file:

```
set_attribute lib_search_path "/home/ahmad23209/Downloads/DFT/minareareports/"
set_attribute hdl_search_path "/home/ahmad23209/Downloads/DFT/minareareports/"
set_attr library "/home/ahmad23209/Downloads/DFT/minareareports/fast.lib"
read_hdl top8.v
#set_top_module top6_module
elaborate
read_sdc Min_area.sdc
synthesize -to_mapped -effort medium
report timing -lint
set_attribute dft_scan_style muxed_scan
define_dft shift_enable -active high -create_port scan_en
define_dft test_mode -active high -create_port test_mode
define_dft test_clock clk
report dft_setup
check_dft_rules >../reports/dft_report/minar/dft_rules_report_minar
fix_dft_violations -test_control test_mode -async_set -async_reset -clock
synthesize -to_mapped
set_attr dft_min_number_of_scan_chains 2 top6_module
set_attr dft_mix_clock_edges_in_scan_chains true top6_module
replace_scan
connect_scan_chains -auto_create_chains -preview
connect_scan_chains -auto_create_chains
report qor
write_atpg -cadence > ../reports/dft_report/minar/rtl_module_minar.atpg
write_atpg -stil > ../reports/dft_report/minar/rtl_module_still_minar.atpg
write_scandef > ../reports/dft_report/minar/rtl_module_minar.def
write_sdf -timescale ps -nonegchecks -recrem split -edges check_edge > dft_report/minar/delays_minar.sdf
write_hdl -lec > report_Mid/HDL_tight_Netlist.v
write_sdc > report_Mid/constraints.sdc
write_script > report_Mid/synthesis.g
report_timing > report_Mid/synthesis_timing_report.rep
report_power > report_Mid/synthesis_power_report.rep
report_gates > report_Mid/synthesis_cell_report.rep
report_area > report_Mid/synthesis_area_report.rep

gui_show
```

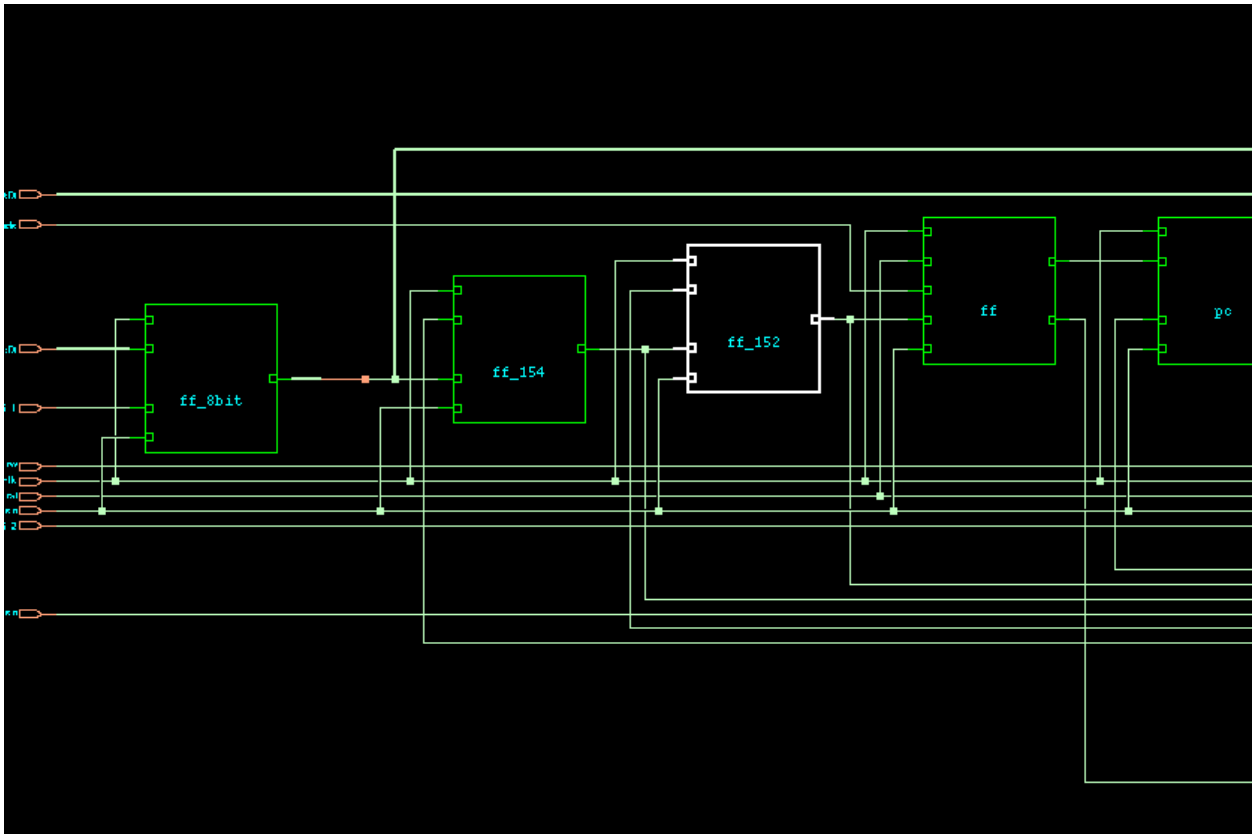
SDC file:

```
create_clock -name CLK -period 1.5 [get_ports clk]
set_clock_transition  0.4 [get_clocks CLK]
#set_clock_transition  0.4 [get_ports clk ]
set_input_delay -clock CLK 0.4 [get_ports rst]
set_input_delay -clock CLK 0.4 [get_ports alu_in1]
set_input_delay -clock CLK 0.4 [get_ports ff_pc_rst]
set_output_delay -clock CLK 0.2 [get_ports ff_zero]
set_output_delay -clock CLK 0.2 [get_ports ff_carry]
set_output_delay -clock CLK 0.2 [get_ports ff_negative]
set_output_delay -clock CLK 0.2 [get_ports alu_result]
set_load 0.1 [get_ports ff_zero]
set_load 0.1 [get_ports ff_carry]
set_load 0.1 [get_ports ff_negative]
set_load 0.1 [get_ports alu_result]

# Specify that the memory should be implemented using flip-flops
set_attribute -library SEQUENTIAL -design {mem[*]} ramstyle registers

# Optimize memory initialization process
set_attribute -name INIT -value "11'b10110010010" [get_cells -hierarchical -filter {name =~ "mem[*]"}]

# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_1_seq -value false [current_design]
# Set optimize_constant_1_flops to false
set_attribute -name optimize_constant_0_seq -value false [current_design]

# Set 'preserve' instance attribute to 'true'
set_attribute -instance * -name preserve true
```

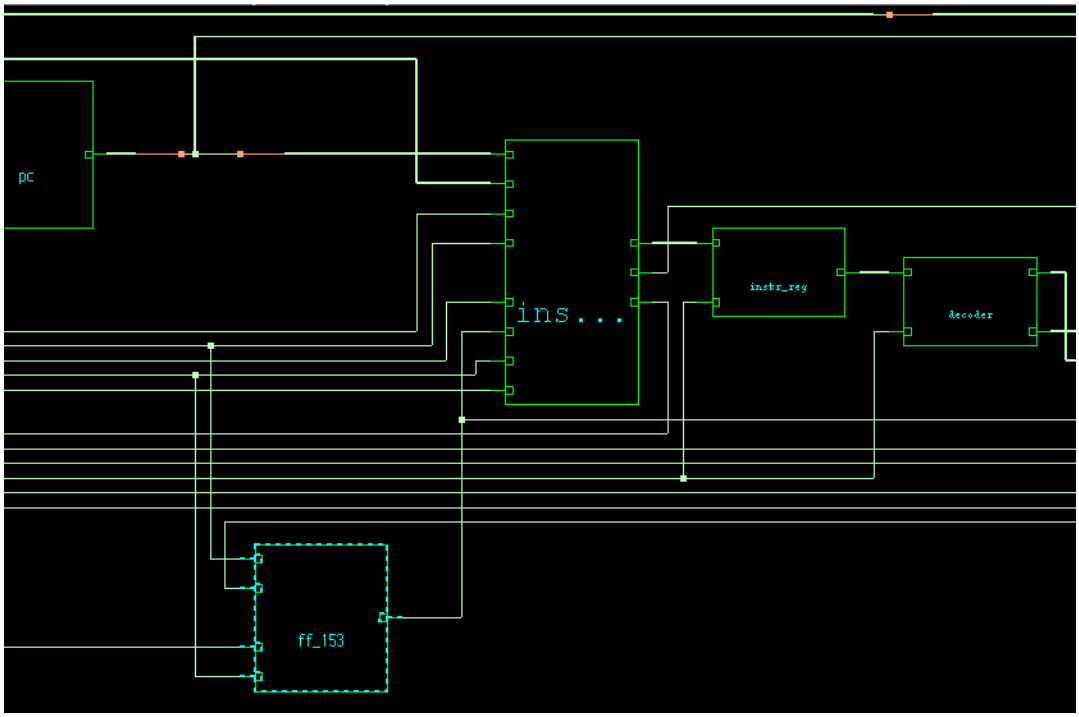
New Schematic after DFT:



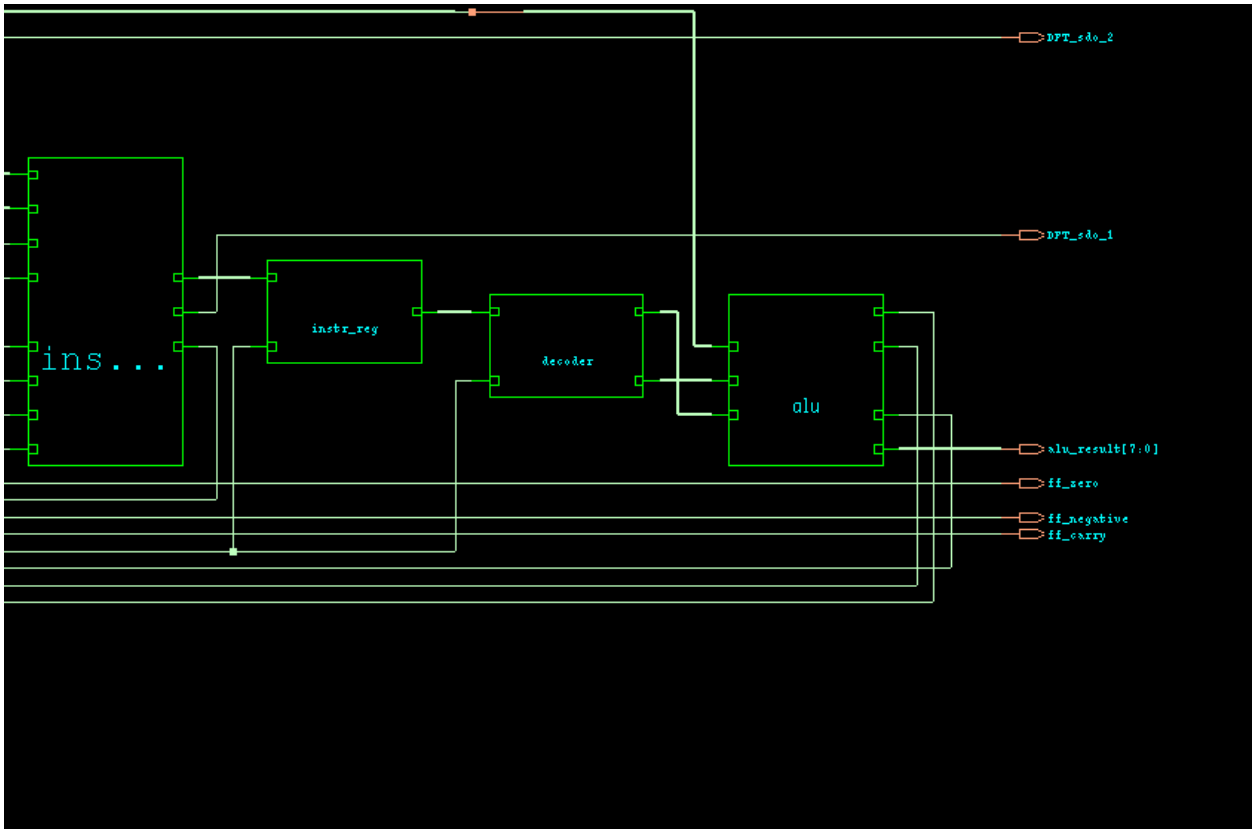
DFT_sdi_1, DFT_sdi_2 and scan_en inputs added.



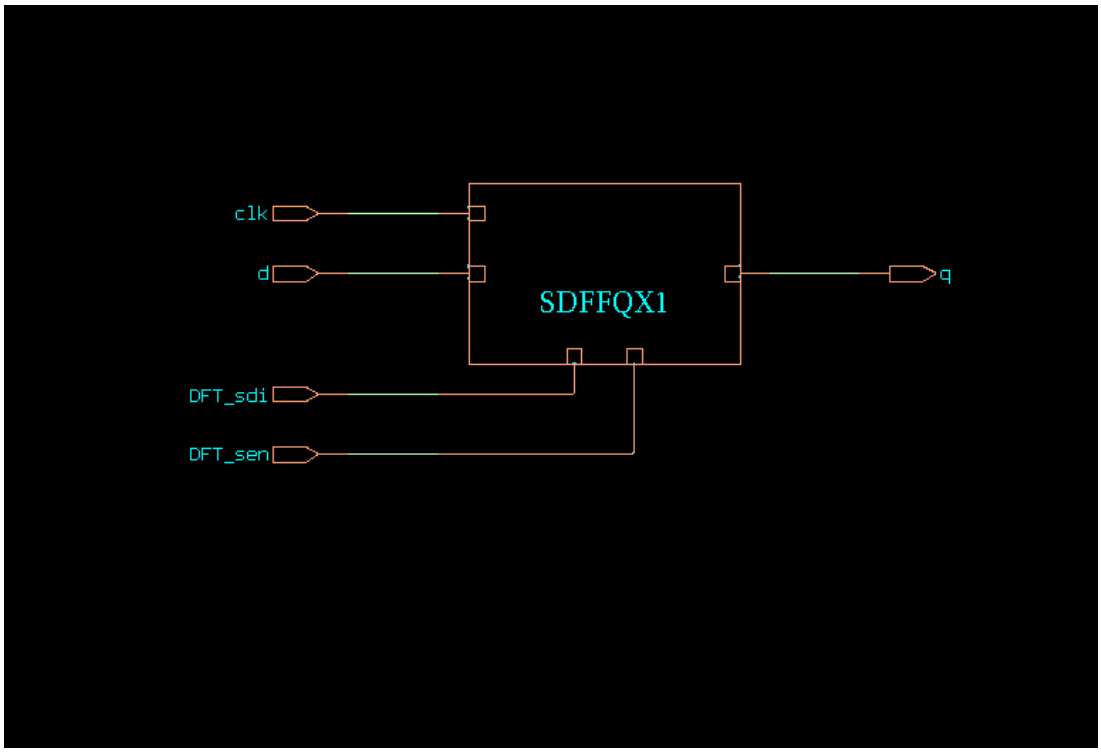
A Flip flop is added before instruction memory



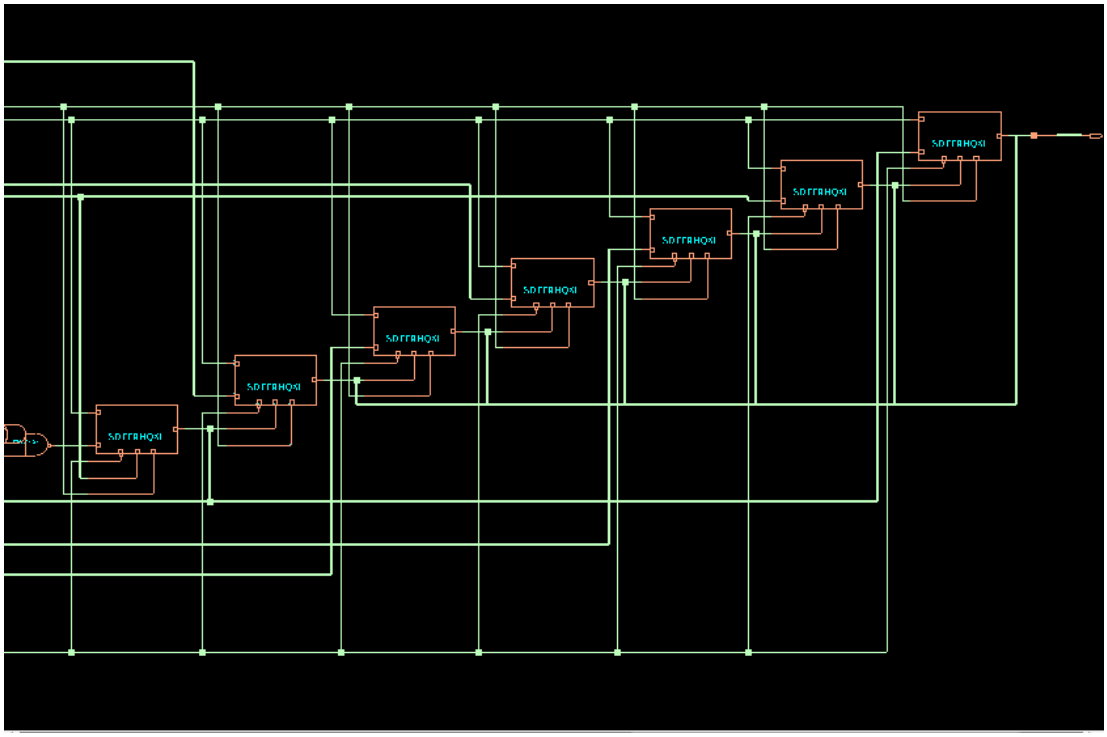
Outputs DFT_sdo_1 and DFT_sdo_2 outputs added.



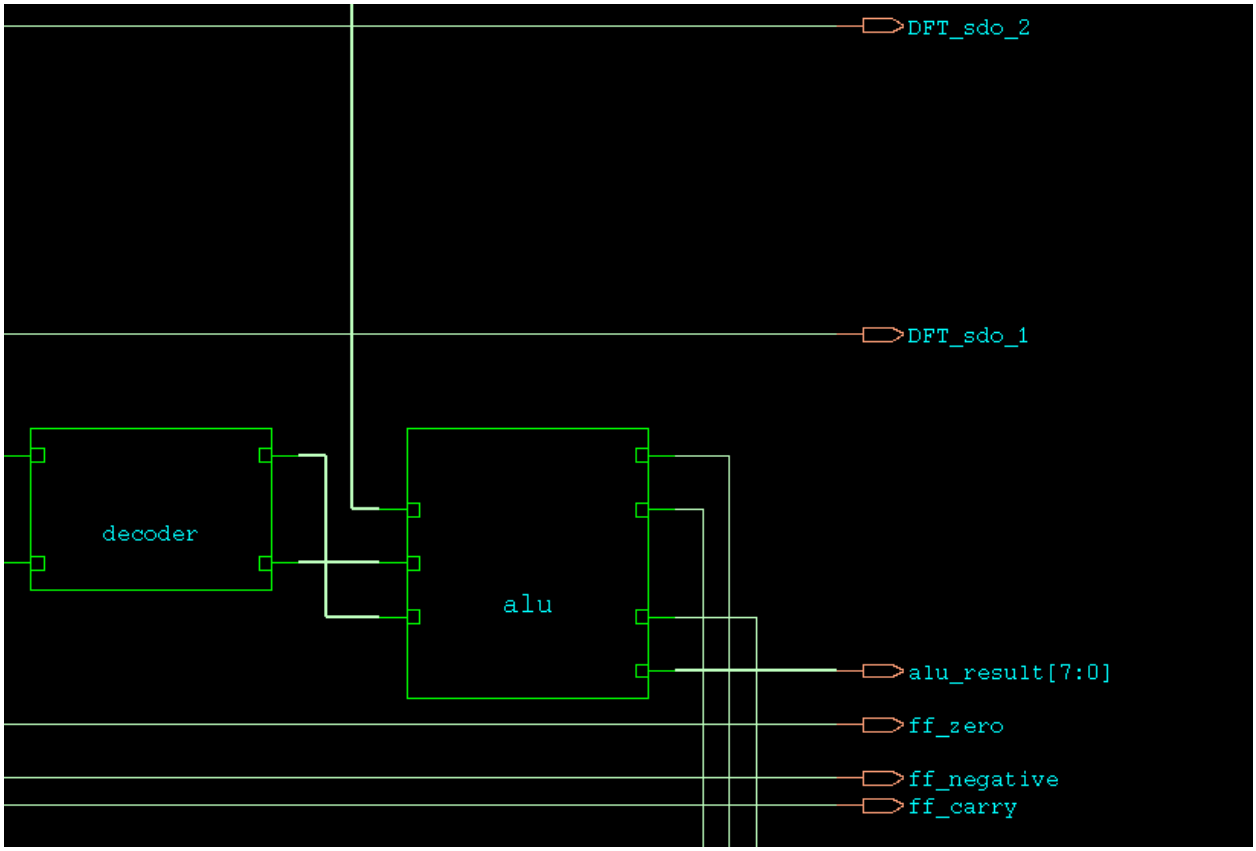
Flip flop:



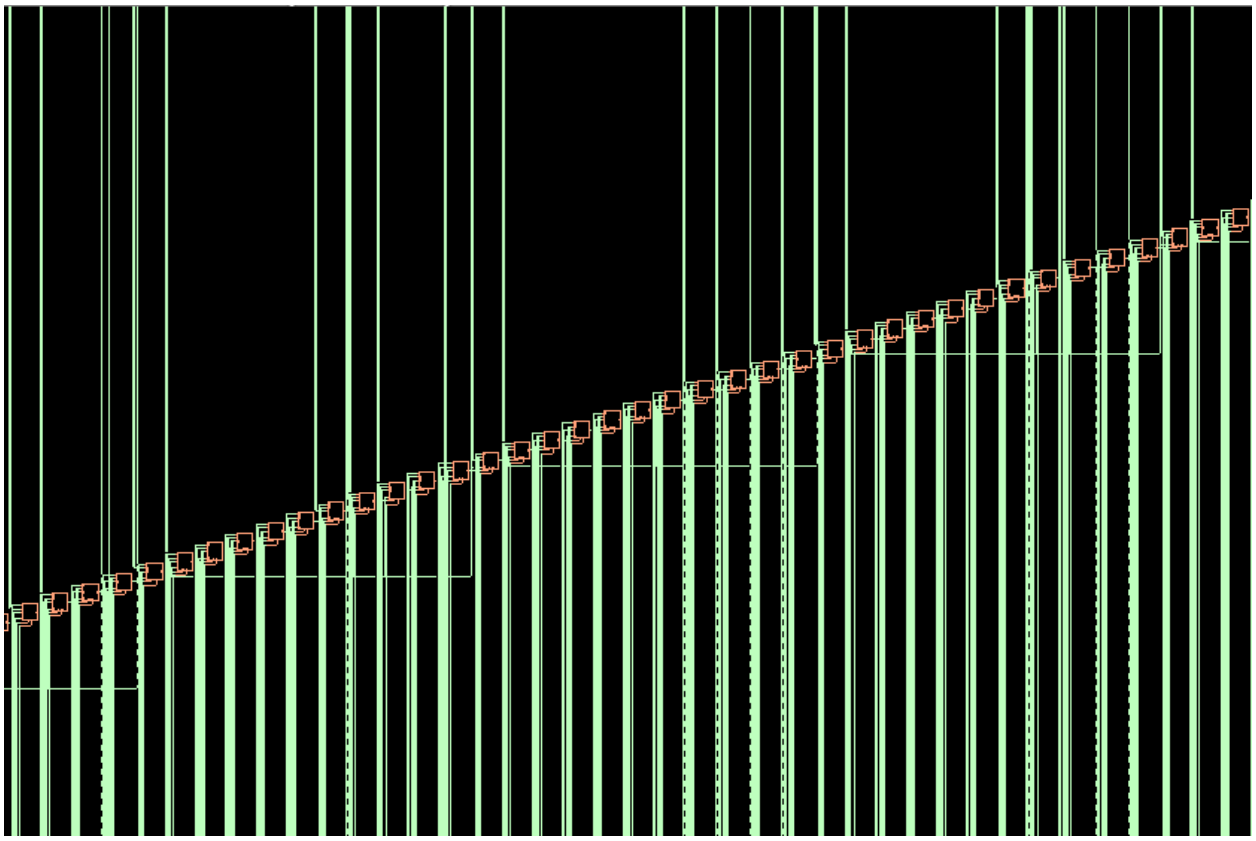
A Number of SDFFRHQXs were added to the Program Counter



Flip flops at the end were removed during DFT



Most of the scan cells were added inside the Instruction Memory module.



Added SMDFFHQX1s in instruction memory

Netlist:

```
module pc(clk, rst, pc_out_addr);
  input clk, rst;
  output [5:0] pc_out_addr;
  wire clk, rst;
  wire [5:0] pc_out_addr;
  wire [5:0] next_addr;
  wire n_0, n_1, n_2, n_3, n_4, n_5, n_6, n_7;
  wire n_8, n_9, n_10, n_11;
  DFFRHQX1 \pc_out_addr_reg[5] (.RN (n_2), .CK (clk), .D (next_addr[5]), .Q (pc_out_addr[5]));
  DFFRHQX1 \pc_out_addr_reg[4] (.RN (n_2), .CK (clk), .D (next_addr[4]), .Q (pc_out_addr[4]));
  DFFRHQX1 \pc_out_addr_reg[2] (.RN (n_2), .CK (clk), .D (next_addr[2]), .Q (pc_out_addr[2]));
  DFFRHQX1 \pc_out_addr_reg[1] (.RN (n_2), .CK (clk), .D (next_addr[1]), .Q (pc_out_addr[1]));
  DFFRHQX1 \pc_out_addr_reg[3] (.RN (n_2), .CK (clk), .D (next_addr[3]), .Q (pc_out_addr[3]));
  DFFRHQX1 \pc_out_addr_reg[0] (.RN (n_2), .CK (clk), .D (next_addr[0]), .Q (pc_out_addr[0]));
  INVXL g35(.A (rst), .Y (n_2));
  SDFFRHQX1 \next_addr_reg[5] (.RN (n_2), .CK (clk), .D (n_3), .SI (next_addr[5]), .SE (n_10), .Q (next_addr[5]));
  DFFRHQX1 \next_addr_reg[4] (.RN (n_2), .CK (clk), .D (n_11), .Q (next_addr[4]));
  OA21XL g108(.A0 (next_addr[4]), .A1 (n_8), .B0 (n_10), .Y (n_11));
  DFFRHQX1 \next_addr_reg[3] (.RN (n_2), .CK (clk), .D (n_9), .Q (next_addr[3]));
  NAND2XL g110(.A (n_8), .B (next_addr[4]), .Y (n_10));
  AOI2BB1X1 g111(.A0N (next_addr[3]), .A1N (n_6), .B0 (n_8), .Y (n_9));
  AND2X1 g113(.A (next_addr[3]), .B (n_6), .Y (n_8));
  AOI21X1 g114(.A0 (n_1), .A1 (n_4), .B0 (n_6), .Y (n_7));
  DFFRHQX1 \next_addr_reg[1] (.RN (n_2), .CK (clk), .D (n_5), .Q (next_addr[1]));
  NOR2XL g116(.A (n_1), .B (n_4), .Y (n_6));
  OA21XL g117(.A0 (next_addr[0]), .A1 (next_addr[1]), .B0 (n_4), .Y (n_5));
  NAND2XL g119(.A (next_addr[1]), .B (next_addr[0]), .Y (n_4));
  INVX1 g121(.A (next_addr[5]), .Y (n_3));
  DFFRX1 \next_addr_reg[2] (.RN (n_2), .CK (clk), .D (n_7), .Q (next_addr[2]), .QN (n_1));
  DFFRX1 \next_addr_reg[0] (.RN (n_2), .CK (clk), .D (n_0), .Q (next_addr[0]), .QN (n_0));
endmodule
```

```
  assign \mem[0] [10] = 1'b0;
  assign \mem[0] [9] = 1'b0;
  assign \mem[0] [8] = 1'b0;
  assign \mem[0] [7] = 1'b1;
  assign \mem[0] [6] = 1'b0;
  assign \mem[0] [5] = 1'b0;
  assign \mem[0] [4] = 1'b1;
  assign \mem[0] [3] = 1'b0;
  assign \mem[0] [2] = 1'b0;
  assign \mem[0] [1] = 1'b1;
  assign \mem[0] [0] = 1'b0;
  DFFQX1 \mem_instr_out_reg[0] (.CK (clk), .D (n_615), .Q (mem_instr_out[0]));
  DFFQX1 \mem_instr_out_reg[1] (.CK (clk), .D (n_617), .Q (mem_instr_out[1]));
  DFFQX1 \mem_instr_out_reg[2] (.CK (clk), .D (n_634), .Q (mem_instr_out[2]));
  DFFQX1 \mem_instr_out_reg[3] (.CK (clk), .D (n_652), .Q (mem_instr_out[3]));
  DFFQX1 \mem_instr_out_reg[4] (.CK (clk), .D (n_619), .Q (mem_instr_out[4]));
  DFFQX1 \mem_instr_out_reg[5] (.CK (clk), .D (n_654), .Q (mem_instr_out[5]));
  DFFQX1 \mem_instr_out_reg[6] (.CK (clk), .D (n_653), .Q (mem_instr_out[6]));
  DFFQX1 \mem_instr_out_reg[7] (.CK (clk), .D (n_618), .Q (mem_instr_out[7]));
  DFFQX1 \mem_instr_out_reg[8] (.CK (clk), .D (n_645), .Q (mem_instr_out[8]));
  DFFQX1 \mem_instr_out_reg[9] (.CK (clk), .D (n_644), .Q (mem_instr_out[9]));
  DFFQX1 \mem_instr_out_reg[10] (.CK (clk), .D (n_643), .Q (mem_instr_out[10]));
  SDFFQXL \mem_reg[0][0]266 (.CK (clk), .D (\mem[0] [0]), .SI (instr_in[0]), .SE (n_122), .Q (\mem[0] [0]));
  SDFFQXL \mem_reg[0][1]267 (.CK (clk), .D (\mem[0] [1]), .SI (instr_in[1]), .SE (n_122), .Q (\mem[0] [1]));
  SDFFQXL \mem_reg[0][2]268 (.CK (clk), .D (\mem[0] [2]), .SI (instr_in[2]), .SE (n_122), .Q (\mem[0] [2]));
  SDFFQXL \mem_reg[0][3]269 (.CK (clk), .D (\mem[0] [3]), .SI (instr_in[3]), .SE (n_122), .Q (\mem[0] [3]));
  SDFFQXL \mem_reg[0][4]270 (.CK (clk), .D (\mem[0] [4]), .SI (instr_in[4]), .SE (n_122), .Q (\mem[0] [4]));
  SDFFQXL \mem_reg[0][5]271 (.CK (clk), .D (\mem[0] [5]), .SI (instr_in[5]), .SE (n_122), .Q (\mem[0] [5]));
  SDFFQXL \mem_reg[0][6]272 (.CK (clk), .D (\mem[0] [6]), .SI (instr_in[6]), .SE (n_122), .Q (\mem[0] [6]));
  SDFFQXL \mem_reg[0][7]273 (.CK (clk), .D (\mem[0] [7]), .SI (instr_in[7]), .SE (n_122), .Q (\mem[0] [7]));
  SDFFQXL \mem_reg[0][8]274 (.CK (clk), .D (\mem[0] [8]), .SI (instr_in[8]), .SE (n_122), .Q (\mem[0] [8]));
  SDFFQXL \mem_reg[0][9]275 (.CK (clk), .D (\mem[0] [9]), .SI (instr_in[9]), .SE (n_122), .Q (\mem[0] [9]));
  SDFFQXL \mem_reg[0][10]276 (.CK (clk), .D (\mem[0] [10]), .SI (instr_in[10]), .SE (n_122), .Q (\mem[0] [10]));
```

The need for new entities in the scan-chain inserted netlist:

1. The circuit resulting from DFT introduces four new ports: scan_en, DFT_sdi_i, DFT_sdo_1, and test_mode, catering to three distinct modes: Normal mode, shift mode, and capture mode. During DFT, the system operates in the Shift mode, where test_mode is set to 1. scan_en facilitates scan mode, and DFT_sdi_i transfers the desired test vector within the circuit. The resultant output is observed at DFT_sdo_1, allowing comparison with expected results to verify device correctness.
2. During Normal mode (select signal = 0), input D is routed to output Q. Upon activation of test mode (select signal = 1), test vectors propagate within the design, enabling testing of different die portions.

3. The design implements a single scan chain. Multiple scan chains can be configured to form a shift register known as scan chains. Test vectors can be shifted in from scan-in ports to scan cells over N clock cycles, and test responses can be shifted out from scan cells to check ports in N clock cycles.

Detection of Failures with the Scan-chain Inserted Netlist:
Circuit defects include Stuck at 0, Stuck at 1, and faults in gates or flip-flops.
The scan operation encompasses three primary modes:
1. Shift mode: Test vectors are applied through the DFT_sdi pin to set internal gate outputs to a known state, enhancing controllability and observability.
2. Capture mode: Logic functionality is captured in a single clock cycle.
3. Repeat: The system re-enters shift mode, propagating the captured output to the DFT_sdo pin for comparison against expected values. A mismatch indicates a fault in the internal gate under test. This cycle is reiterated for every potential fault node. Test vectors tailored to specific defects aid in diagnosing gate problems.

1. For min area case:

CELL AND AREA REPORT:

Gate	Instances	Area	Library
ADDHX1	1	12.110	fast
AND2X1	1	4.541	fast
AO22X1	9	68.121	fast
AO22XL	6	45.414	fast
AOI211X1	8	42.386	fast
AOI211XL	3	15.895	fast
AOI21X1	9	40.873	fast
AOI21XL	1	4.541	fast
AOI221X1	34	257.346	fast
AOI221XL	12	90.828	fast
AOI222XL	18	149.866	fast
AOI22X1	32	193.766	fast
AOI22XL	253	1531.966	fast
AOI31XL	1	6.055	fast
AOI32X1	1	6.812	fast
AOI33XL	1	7.569	fast
CLKINVX1	45	102.182	fast
INVX1	14	31.790	fast
INVXL	9	20.436	fast
MX2X1	1	6.812	fast
MXI2XL	18	108.994	fast
NAND2BX1	21	95.369	fast
NAND2BXL	1	4.541	fast
NAND2XL	100	302.760	fast
NAND3X1	11	49.955	fast
NAND4BBXL	3	24.978	fast
NAND4BXL	2	13.624	fast
NAND4XL	71	376.179	fast
NOR2BX1	11	49.955	fast
NOR2BXL	3	13.624	fast
NOR2XL	143	432.947	fast
NOR3X1	8	36.331	fast

NOR3X1	8	36.331	fast
NOR3XL	1	4.541	fast
NOR4X1	4	24.221	fast
OA21X1	1	6.812	fast
OA21XL	2	13.624	fast
OA22X1	2	15.138	fast
OAI211X1	11	58.281	fast
OAI211XL	5	26.492	fast
OAI21X1	8	36.331	fast
OAI21XL	7	31.790	fast
OAI221X1	1	7.569	fast
OAI221XL	1	7.569	fast
OAI222XL	3	24.978	fast
OAI22X1	2	12.110	fast
OAI22XL	4	24.221	fast
OAI2BB1X1	6	31.790	fast
OAI33X1	1	8.326	fast
OR2X1	2	9.083	fast
OR4XL	2	13.624	fast
SDFFQX1	23	470.035	fast
SDFFRHQX1	12	299.732	fast
SMDFFHQX1	704	18650.017	fast
TLATX1	30	431.433	fast
total	1683	24356.286	

Type	Instances	Area	Area %
sequential	769	19851.217	81.5
inverter	68	154.408	0.6
logic	846	4350.661	17.9
physical_cells	0	0.000	0.0
total	1683	24356.286	100.0

Total Area = 24356.286 µm²

Total Cells = 1683

Timing report:

```
TIMING REPORT:
          Pin                      Type      Fanout Load Slew Delay Arrival
                                                    (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------
(clock CLK)                        launch                                0 R
(Min_area.sdc_line_4)              ext delay                  +400     400 R
rst                                in port         3  6.7    0    +0     400 R
mem_inst/rst
  g67143/B                                                    +0     400
  g67143/Y                         NOR2BX1        5 11.0   25   +17     417 F
  g67131/A                                                    +0     417
  g67131/Y                         NAND2XL       16 28.8  150  +116     533 R
  g67049/B                                                    +0     533
  g67049/Y                         NOR2XL        11 21.1   99   +76     609 F
  g66697/A1                                                   +0     609
  g66697/Y                         AO22XL         1  2.9   20   +63     672 F
  g66631/C0                                                   +0     672
  g66631/Y                         AOI221X1       1  1.7   43   +33     705 R
  g65893/B                                                    +0     705
  g65893/Y                         NAND4XL        1  2.9   49   +38     743 F
  g65882/C0                                                   +0     743
  g65882/Y                         AOI221X1       1  3.0   50   +48     791 R
  g65871/C                                                    +0     791
  g65871/Y                         NAND3X1        1  2.9   29   +23     814 F
```

```
  g65860/C0                                                   +0     814
  g65860/Y                         AOI221X1       1  1.6   46   +35     849 R
  g65849/A                                                    +0     849
  g65849/Y                         NAND4XL        1  1.8   36   +30     879 F
  g65838/B                                                    +0     879
  g65838/Y                         NOR2XL         1  1.6   26   +28     907 R
  g65827/A                                                    +0     907
  g65827/Y                         NAND4XL        1  2.9   45   +37     944 F
  g65816/C                                                    +0     944
  g65816/Y                         NOR3X1         1  1.6   30   +31     975 R
  g65805/A                                                    +0     975
  g65805/Y                         NAND4XL        1  3.0   46   +37    1012 F
  g65794/C0                                                   +0    1012
  g65794/Y                         AOI211X1       1  3.0   43   +43    1055 R
  g65788/C0                                                   +0    1055
  g65788/Y                         OAI211X1       1  1.9   28   +24    1079 F
  mem_instr_out_reg[4]/D  <<<      SDFFQX1                     +0    1079
  mem_instr_out_reg[4]/CK          setup                  400  +153    1233 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock CLK)                        capture                            3000 R
-------------------------------------------------------------------------
Cost Group   : 'CLK' (path_group 'CLK')
Timing slack :    1767ps
Start-point  : rst
End-point    : mem_inst/mem_instr_out_reg[4]/D
```

Timing Slack = 1767ps

Power Report:

```
POWER REPORT:

Instance: /top8_module
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     2.59204e-04  1.18325e-02  1.12528e-04  1.22042e-02  93.33%
       latch     6.49584e-06  3.93495e-05  2.76477e-05  7.34930e-05   0.56%
       logic     3.39466e-05  1.48511e-04  9.35712e-05  2.76029e-04   2.11%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  5.22075e-04  5.22075e-04   3.99%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     2.99646e-04  1.20204e-02  7.55822e-04  1.30758e-02  99.99%
  Percentage           2.29%       91.93%        5.78%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Total Power = 13.0758mW

Analysis:

Introducing a scan chain increases area, primarily due to SDFFQX1, which occupies more space than DFFQX1. This is attributed to SDFFQX1 having more pins and connections. Normal flip-flops in the design are substituted with scan-type flip-flops, keeping the area occupied by other elements unchanged.

Following the implementation of the scan chain, there is a reduction in timing slack compared to the scenario without the scan chain. This decrease in slack is a consequence of the additional delay incurred while routing the scan cells.

Case 2: For Best timingArea report:

```
CELL AND AREA REPORT:

   Gate     Instances    Area    Library
-----------------------------------------
ADDHX1               1     12.110    fast
AND2X1              10     45.414    fast
AND4X1               1      6.812    fast
AND4XL               3     20.436    fast
AO22X1              13     98.397    fast
AOI211X1             9     47.685    fast
AOI211XL            10     52.983    fast
AOI21X1             40    181.656    fast
AOI21XL             16     72.662    fast
AOI221X1             7     52.983    fast
AOI221XL            30    227.070    fast
AOI222XL            20    166.518    fast
AOI22X1             36    217.987    fast
AOI22XL            194   1174.709    fast
AOI31X1              2     12.110    fast
AOI31XL              1      6.055    fast
AOI32X1              1      6.812    fast
AOI33XL              1      7.569    fast
CLKAND2X12           4     93.856    fast
CLKBUFX2             8     36.331    fast
CLKINVX1            65    147.596    fast
```

NAND2X4	62	610.061	fast
NAND2X6	2	28.762	fast
NAND2XL	615	1861.974	fast
NAND3X1	5	22.707	fast
NAND3XL	1	4.541	fast
NAND4BBXL	2	16.652	fast
NAND4BXL	4	27.248	fast
NAND4XL	76	402.671	fast
NOR2BX1	52	236.153	fast
NOR2BXL	1	4.541	fast
NOR2X1	16	60.552	fast
NOR2X2	1	6.055	fast
NOR2XL	121	366.340	fast
NOR3BX1	34	205.877	fast
NOR3BXL	1	6.055	fast
NOR3X1	3	13.624	fast
NOR3XL	1	4.541	fast
NOR4BX1	1	6.812	fast
NOR4X1	4	24.221	fast
OA21X1	1	6.812	fast
OA21XL	2	13.624	fast
OA22X1	1	7.569	fast
OAI211X1	5	26.492	fast
OAI211XL	6	31.790	fast
OAI21X1	217	985.484	fast
OAI21X2	1	8.326	fast
OAI21XL	375	1703.025	fast
OAI221X1	5	37.845	fast
OAI221XL	1	7.569	fast
OAI222XL	6	49.955	fast
OAI22X1	3	18.166	fast
OAI22XL	2	12.110	fast
OAI2BB1X1	3	15.895	fast
OAI2BB1XL	14	74.176	fast
OAI32X1	1	6.812	fast
OR2X1	3	13.624	fast
OR4XL	1	6.812	fast
SDFFHQX1	365	8011.786	fast
SDFFQX1	18	367.853	fast
SDFFRHQX1	352	8792.150	fast

```
SDFFQX1              18    367.853    fast
SDFFRHQX1           352   8792.150    fast
SDFFRX1               1     28.005    fast
SDFFRX2               3     86.287    fast
TLATX1               30    431.433    fast
XNOR2XL               1      8.326    fast
-----------------------------------------
total              3180  28889.359


     Type       Instances    Area   Area %
-----------------------------------------
sequential             769 17717.515   61.3
inverter               133   407.969    1.4
buffer                   8    36.331    0.1
logic                 2270 10727.544   37.1
physical_cells           0     0.000    0.0
-----------------------------------------
total                  3180 28889.359  100.0
```

Total Area = 28889.359 µm²

Total Cells = 3180

Timing report:

```
TIMING REPORT:
============================================================

         Pin                    Type      Fanout  Load Slew Delay Arrival
                                                 (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------
(clock CLK)                  launch                                   0 R
(Min_area.sdc_line_4)        ext delay                  +400       400 F
rst                          in port        113 510.5    0    +0     400 F
mem_inst/rst
  fopt72121/A                                             +0     400
  fopt72121/Y                CLKINVX20        1  12.5    4    +6     406 R
  g71888/B                                                +0     406
  g71888/Y                   CLKAND2X12      16 193.1   50   +57     463 R
  g71661/B                                                +0     463
  g71661/Y                   NAND2X4         19  36.3   72   +32     494 F
  g70909/A                                                +0     494
  g70909/Y                   NAND2X1          1   1.8   25   +26     521 R
  g70438/B0                                               +0     521
  g70438/Y                   OAI21XL          1   3.0   39   +24     544 F
  mem_reg[29][8]912/D  <<<   SDFFHQX1                     +0     544
  mem_reg[29][8]912/CK       setup                   400  +123     668 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock CLK)                  capture                                651 R
-------------------------------------------------------------------------
Cost Group   : 'CLK' (path_group 'CLK')
Timing slack :     -17ps (TIMING VIOLATION)
Start-point  : rst
End-point    : mem_inst/mem_reg[29][8]912/D
```

Timing Slack = -17ps

Power Report:

```
POWER REPORT:
Instance: /top8_module
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     2.08823e-04  5.17238e-02  2.25586e-04  5.21582e-02  91.58%
        latch     6.49584e-06  2.08976e-04  1.34641e-04  3.50113e-04   0.61%
        logic     9.05538e-05  9.12836e-04  9.04732e-04  1.90812e-03   3.35%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  2.53710e-03  2.53710e-03   4.45%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     3.05873e-04  5.28456e-02  3.80206e-03  5.69535e-02  99.99%
  Percentage           0.54%       92.79%        6.68%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Total Area = 56.9535mW

Case 3: Between (a) and (b) Timing

Area Report:

Gate	Instances	Area	Library
ADDHX1	1	12.110	fast
AO22X1	10	75.690	fast
AOI211X1	1	5.298	fast
AOI211XL	15	79.475	fast
AOI21X1	11	49.955	fast
AOI221X1	2	15.138	fast
AOI221XL	22	166.518	fast
AOI222XL	10	83.259	fast
AOI22X1	6	36.331	fast
AOI22XL	309	1871.057	fast
AOI2BB1X1	1	6.055	fast
AOI2BB1XL	1	6.055	fast
AOI31XL	1	6.055	fast
AOI32X1	1	6.812	fast
CLKINVX1	43	97.640	fast
INVX1	11	24.978	fast
INVXL	2	4.541	fast
MXI2XL	16	96.883	fast
NAND2BX1	15	68.121	fast
NAND2BXL	1	4.541	fast
NAND2XL	78	236.153	fast
NAND3BX1	1	6.055	fast
NAND3X1	4	18.166	fast
NAND3XL	5	22.707	fast
NAND4BXL	5	34.060	fast
NAND4XL	86	455.654	fast
NOR2BX1	11	49.955	fast
NOR2BXL	1	4.541	fast
NOR2XL	150	454.140	fast
NOR3BX1	1	6.055	fast
NOR3X1	6	27.248	fast
NOR3XL	2	9.083	fast
NOR4BXL	1	6.812	fast
NOR4X1	4	24.221	fast
NOR4XL	3	18.166	fast
OA21XL	2	13.624	fast
OA22X1	1	7.569	fast
OAI211XL	6	31.790	fast

```
OAI211XL              6      31.790    fast
OAI21X1              11      49.955    fast
OAI21XL               5      22.707    fast
OAI221X1              4      30.276    fast
OAI222XL              7      58.281    fast
OAI22X1               6      36.331    fast
OAI22XL               7      42.386    fast
OAI2BB1X1             3      15.895    fast
OAI2BB1XL             2      10.597    fast
OAI32X1               2      13.624    fast
OR2X1                 3      13.624    fast
OR3XL                 1       6.055    fast
OR4XL                 4      27.248    fast
SDFFQX1              23     470.035    fast
SDFFRHQX1            12     299.732    fast
SMDFFHQX1           704   18650.017    fast
TLATNX1               8     115.049    fast
TLATX1               22     316.384    fast
----------------------------------------
total              1670   24320.711


     Type       Instances    Area   Area %
------------------------------------------
sequential             769 19851.217   81.6
inverter                56   127.159    0.5
logic                  845  4342.335   17.9
physical_cells           0     0.000    0.0
------------------------------------------
total                 1670 24320.711  100.0
```

Total Area = 24320.711µm²

Total Cells = 1670

Timing report:

```
TIMING REPORT:

============================================================

          Pin                    Type      Fanout Load Slew Delay Arrival
                                                 (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------
(clock CLK)                    launch                                  0 R
(Min_area.sdc_line_4)          ext delay                    +400     400 F
rst                            in port          3  5.6    0    +0     400 F
mem_inst/rst
  g67717/B                                                    +0     400
  g67717/Y                     NOR2XL           4  8.1   78   +57     457 R
  g67691/A                                                    +0     457
  g67691/Y                     NAND2XL         14 26.3  140  +110     567 F
  g67578/B                                                    +0     567
  g67578/Y                     NOR2XL          11 56.1  486  +379     946 R
  mem_reg[61][10]1618/S0 <<<   SMDFFHQX1                      +0     946
  mem_reg[61][10]1618/CK       setup                     400  +137    1083 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock CLK)                    capture                                1500 R
-------------------------------------------------------------------------
Cost Group   : 'CLK' (path_group 'CLK')
Timing slack :     417ps
Start-point  : rst
End-point    : mem_inst/mem_reg[61][10]1618/S0
```

Timing Slack = 417pS

Power Report:

```
POWER REPORT:

Instance: /top8_module
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total     Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     2.59204e-04  2.36673e-02  2.25410e-04  2.41519e-02  93.70%
        latch     6.49287e-06  8.30722e-05  5.59309e-05  1.45496e-04   0.56%
        logic     3.22419e-05  2.32305e-04  1.70486e-04  4.35033e-04   1.69%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  1.04415e-03  1.04415e-03   4.05%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     2.97939e-04  2.39827e-02  1.49598e-03  2.57766e-02 100.00%
  Percentage           1.16%       93.04%        5.80%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Total Power = 25.7766mW

Comparison in Power, Area, and Timing slack pre-DFT and post-DFT for all 3 cases (tabular form and analysis)

	Area without scan Chain Cells(um^2)	Area with Scan Chain Cells(um^2)	Slack without Scan Chain Cells (pS)	Slack with Scan Chain Cells (pS)
Case 1	19914.796	24356.286	1967	1767
Case 2	21218.178	28889.359	-5	-17
Case 3	19902.685	24320.711	484	417

• In all three cases, it's evident that the area increases, primarily attributed to the additional pins in scan cells and the increased routing resources necessary for constructing scan chains.

• Furthermore, the timing slack diminishes after the implementation of the scan chain when compared to scenarios without it. This decline in slack can be attributed to the supplementary delay introduced by routing the scan cells.

Power comparison pre and post DFT:

	Power without scan Chain Cells(uW)	**Power with scan Chain Cells(uW)**
1) Min Area	**11.24**	**24356.286**
2) Best Timing	**52.74**	**28889.359**
3) Intermediate	**22**	**24320.711**

Slack comparison pre and post DFT:

	Slack without scan Chain Cells(ps)	**Slack with scan Chain Cells(ps)**
1) Min Area	**468**	**411**
2) Best Timing	**852**	**828**
3) Intermediate	**514**	**414**