RTL to GDS-II WORKSHOP

PROJECT REPORT

Submitted by

Kaushal Rathi (23BIT232)

Under Guidance of Puneet Mittal

SCHOOL OF TECHNOLOGY

PANDIT DEENDAYAL ENERGY UNIVERSITY
GANDHINAGAR

May - 2025

Table Of Contents

1. Acknowledgement

I would like to express my heartfelt thanks to **Pandit Deendayal Energy University** for giving me the opportunity and resources to carry out this project titled **"RTL to GDS"** as part of my academic studies.

I am especially thankful to my project guide, **Mr. Puneet Mittal**, for his constant guidance, support, and valuable feedback throughout the project. His direction played a key role in the successful completion of this work.

I also thank the faculty members and technical staff of the **Department of ICT/Electronics and Communication** for their help and encouragement during this period.

I am grateful to **Synopsys** for providing access to their industry-standard EDA tools. Working with these tools greatly improved my understanding of the RTL to GDS backend design process. The hands -on experience with Synopsys tools enhanced my understanding of the backend design process.

Lastly, I sincerely thank my **friends and classmates** for their continuous support and motivation throughout this journey.

2. Abstract

This project report presents the complete physical design flow from **Register Transfer Level (RTL)** to **GDSII** using industry-standard **Synopsys EDA tools**. The RTL to GDS process is a critical part of the VLSI design cycle, converting high-level digital logic descriptions into a manufacturable layout format.

The report outlines each stage of the backend flow, including synthesis, floor planning, placement, clock tree synthesis (CTS), routing, and physical verification. Emphasis is placed on timing closure, design rule checks (DRC), and layout versus schematic (LVS) validations to ensure a functionally correct and optimized silicon design.

The project is carried out using Synopsys tools such as **Design Compiler**, **IC Compiler II**, and **PrimeTime**, providing a real-world understanding of the design and implementation challenges faced in the semiconductor industry. The methodology followed aligns with the standard industry practices, offering a practical and professional approach to ASIC physical design.

This project reinforces theoretical knowledge with practical skills, offering valuable insights into the complex process of digital chip design from RTL to final tape-out.

3. Theoretical Background: RTL to GDSII Flow

3.1. Introduction

The RTL to GDSII flow represents the backend phase of the VLSI (Very Large Scale Integration) design process. It involves transforming a Register Transfer Level (RTL) description of a digital circuit into a physical layout that is ready for fabrication. This critical phase bridges the gap between logical design and silicon implementation, playing a key role in the development of modern integrated circuits.

The backend process is highly automated and tool-driven, requiring precise coordination between logical functionality and physical constraints. Mastery of this flow is essential to ensure optimal performance, power efficiency, and manufacturability in today's complex ICs.

In this project, the complete backend flow has been implemented using Synopsys EDA tools, which are widely used in the semiconductor industry. These tools automate and optimize each design step while meeting constraints for timing, area, power, and reliability.

3.2. Design Flow Overview

The backend design flow consists of several critical stages:

1. RTL Design (Front-End Input)
2. Logic Synthesis
3. Design for Test (DFT) Insertion
4. Floorplanning
5. Placement
6. Clock Tree Synthesis (CTS)
7. Routing
8. Physical Verification (DRC/LVS)
9. Static Timing Analysis (STA)
10. GDSII Generation

1. RTL Design and Synthesis

The design process begins at the RTL level using HDLs like Verilog or VHDL to describe circuit functionality. Logic synthesis converts this RTL into a gate-level netlist using a standard cell library.
Using **Synopsys Design Compiler**, the design is optimized for **area, timing**, and **power**.

Outputs:

- Gate-level netlist
- Timing reports
- Area and power estimates

2. Logic synthesis

It is the first step of the backend flow. It transforms the RTL code into a gate-level netlist using a standard cell library. The netlist is a collection of logic gates and flip-flops that implement the desired functionality.

3. Design for Test (DFT)

Design for Test (DFT) is a critical methodology integrated into digital design to ensure manufacturability and facilitate post-fabrication testing. In this project, while the primary focus was on functional design and physical implementation, the following DFT concepts are acknowledged and considered for future enhancement

4. Floorplanning

Floorplanning defines the physical layout of the chip, determining the size, shape, and positions of blocks, I/O pins, and power grids.
A good floorplan ensures **low congestion**, **efficient area usage**, and **timing optimization**.
Tool used: **Synopsys IC Compiler II**.

5. Placement

Standard cells are placed within the floorplan area to reduce **wirelength** and **delay**, while ensuring **legal spacing** and **routability**.
Initial **DRC** and **timing analysis** follow this step.

6. Clock Tree Synthesis (CTS)

CTS distributes the clock signal evenly across the chip to minimize **skew** and **insertion delay**, ensuring synchronized flip-flop operation.
Performed using **Clock Tree Compiler** in IC Compiler II.

7. Routing

Routing connects the placed cells using metal layers in two stages:

- **Global Routing** – Estimates routes
- **Detailed Routing** – Final wire paths, vias

It must meet **design rules**, **signal integrity**, and **timing constraints**.

8. Physical Verification

After routing, verification ensures correctness and manufacturability:

- **DRC** – Checks rule violations
- **LVS** – Matches layout with netlist
- **Antenna Check** – Avoids charge buildup issues

Tools used: **Synopsys or foundry-approved verification tools**.

9. Static Timing Analysis (STA)

STA validates the timing of data paths without test vectors using **Synopsys PrimeTime**. It checks for:

- **Setup/hold violations**
- **Clock skew**
- **Critical path delays**

Essential for achieving **timing closure**.

10. GDSII Generation

Final step involves generating the **GDSII** file, which contains complete layout data used for chip fabrication.
This file is sent to the foundry to create **photomasks**, marking the end of the backend flow

4. Problem Statement

a) RTL Design of an 8-bit Magnitude Comparator

Design and implement an 8-bit magnitude comparator using Verilog HDL. The comparator should take two 8-bit binary inputs and generate three outputs indicating whether the first input is greater than, less than, or equal to the second input.

b) Backend Implementation and GDS-II Generation using Synopsys Tools

Using Synopsys EDA tools, perform the complete backend design flow (RTL to GDS-II) for the comparator. This includes synthesis, floor planning, placement, clock tree synthesis, routing, and physical verification. The goals are to:

- Report the design area after synthesis and layout.
- Determine a clock frequency such that the slack is positive but less than 1 ns during static timing analysis.
- Report the power consumption of the design, including internal, switching, and leakage power.

5. Design Specifications

Parameter	Value
Design	8-bit Magnitude Comparator
Inputs	A[7:0], B[7:0]
Outputs	A_GT_B, A_LT_B, A_EQ_B
Clock Frequency Target	≥200 MHz
Slack Target	Positive, < 1 ns
Target Library	saed32rvt_tt0p78vn40c.db
Technology	65nm (Assumed std cell lib)
Tools Used	VCS, Design Compiler, IC Compiler II, PrimeTime

6. RTL Design in Verilog

6.1.RTL code

```
module magComp(
    input  Clock,       // Clock input
    input  [7:0] A,         // 8-bit input A
    input  [7:0] B,         // 8-bit input B
    output reg   Gt,        // A greater than B
    output reg   Lt,        // A less than B
    output reg   Eq         // A equals B
);

always @(posedge Clock) begin
    if (A == B) begin
        Eq <= 1;
        Gt <= 0;
        Lt <= 0;
    end else if (A > B) begin
        Eq <= 0;
        Gt <= 1;
        Lt <= 0;
    end else begin
        Eq <= 0;
        Gt <= 0;
        Lt <= 1;
    end
end

endmodule
```

6.2. Testbench Code

```
`timescale 1ns/1ns
`include "8bit_comparator_rtl.v"

module magComp_tb;
  // Inputs
  reg Clock;
  reg [7:0] A;
  reg [7:0] B;
  wire Gt;
  wire Lt;
  wire Eq;
  magComp dut (
    .Clock(Clock),
    .A(A),
    .B(B),
    .Gt(Gt),
    .Lt(Lt),
    .Eq(Eq)
  );
  initial begin
     $fsdbDumpvars();
    Clock = 0;
    forever #10 Clock = ~Clock;
  end
  // Monitor output signals
```

```
  initial begin
    $display("Time\tClocktA\tB\tGt\tLt\tEq");
    $monitor("%0t\t%b\t%0d\t%0d\t%b\t%b\t%b", $time, Clock, A, B, Gt, Lt,
Eq);
  end
  initial begin
    // Initialize Inputs
    A = 8'b00000000;
    B = 8'b00000000;
    // Wait 2 clock cycles (just for setup)
    #40;
    A = 8'd5;     B = 8'd7;
    #20;
    A = 8'd105;   B = 8'd120;
    #20;
    A = 8'd250;   B = 8'd250;
    #20;
    A = 8'd0;     B = -8'sd5;  // Note: Bitwise interpretation
    #20;
    A = -8'sd5;   B = -8'sd5;
    #20;
    #40;
    $finish;
  end
endmodule
```

7. Simulation Workflow

The simulation was executed with the following commands:

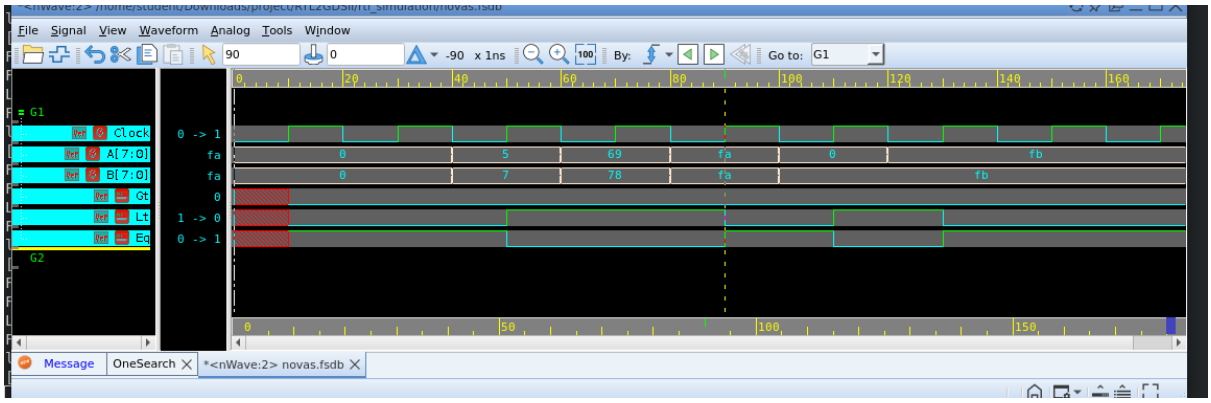
vcs -full64 sequence_detector_110.v tb_sequence_detector_110.v -debug_all

./simv

verdi -ssf fsdbDump.fsdb &

- vcs compiles the RTL and testbench
- simv runs the simulation and generates FSDB waveform
- verdi is launched to view and analyze the waveforms interactively

WAVEFORM FOR VERIFICARION OF TESTBENCH -FILE :



8. RTL Synthesis (Design Compiler)

Tool: Synopsys Design Compiler

Library: saed32rvt_tt0p78vn40c.db

8.1 SDC Constraints

The Synopsys Design Constraints (SDC) file was used to define:

- Clock period
- Input and output delays
- Load and driving conditions

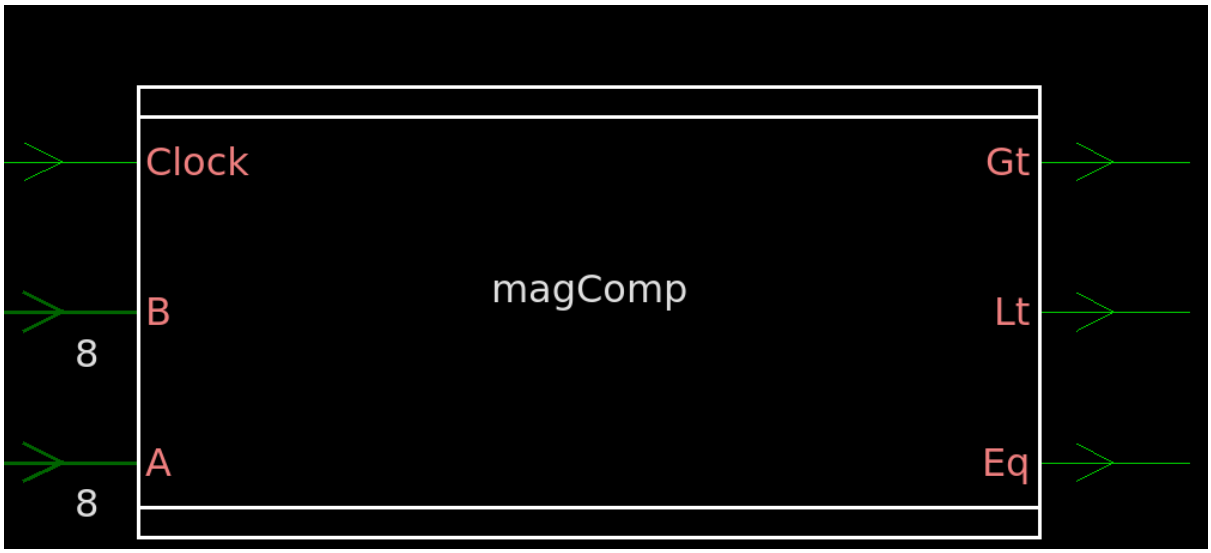
File name:8bit_comparator_sdc

```
create_clock -period 2.38 [get_ports Clock]

set_input_delay -max 0.5 -clock Clock [all_inputs]
set_input_transition 0.5 [all_inputs]

set_output_delay -max 0.5 -clock Clock [all_outputs]

set_clock_uncertainty -setup 0.300 [get_clocks Clock]
set_clock_uncertainty -hold 0.100 [get_clocks Clock]
set_max_transition 0.250 [current_design]
set_max_transition -clock_path 0.150 [get_clocks Clock]
```

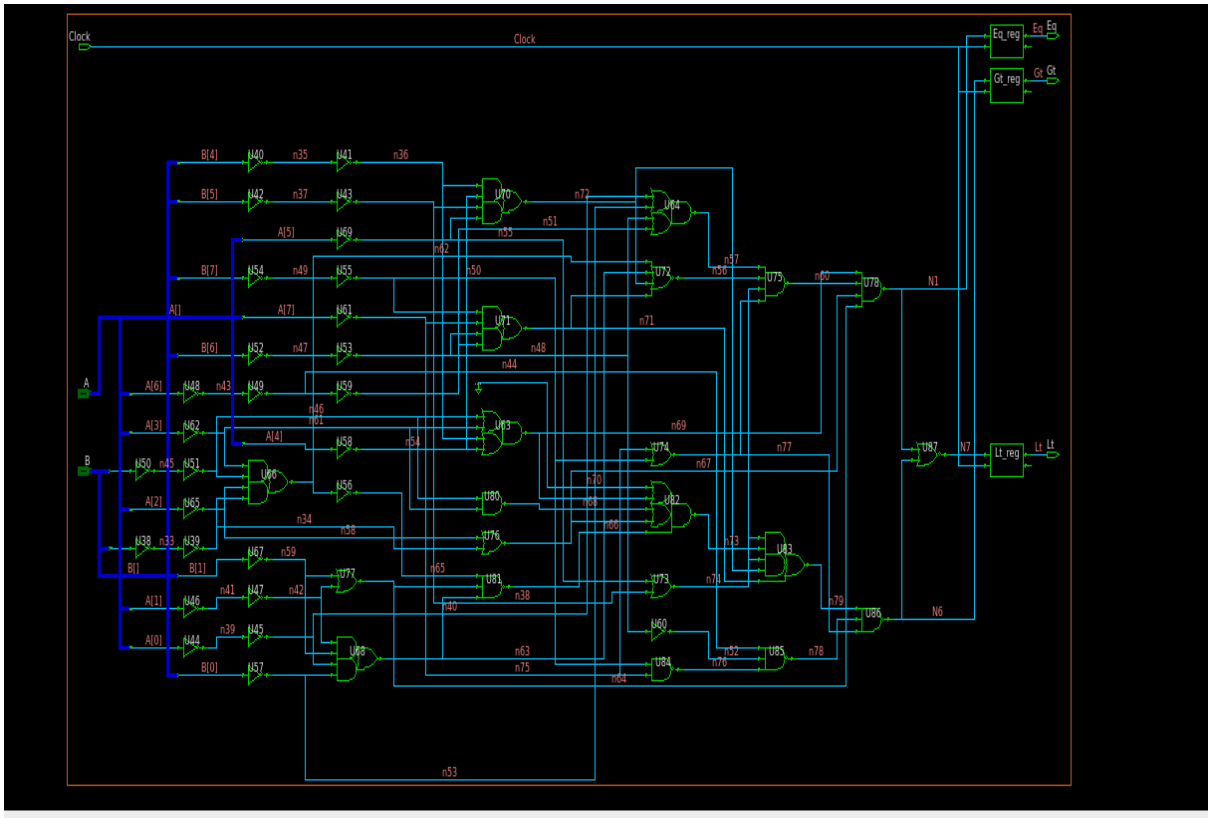
8.2. Synthesis TCL Script.

```
source -echo -verbose ./rm_setup/dc_setup.tcl
set RTL_SOURCE_FILES ./../rtl_simulation/8bit_comparator_rtl.v

define_design_lib WORK -path ./WORK

set_dont_use [get_lib_cells */FADD*]
set_dont_use [get_lib_cells */HADD*]
#set_dont_use [get_lib_cells */AO*]
#set_dont_use [get_lib_cells */OA*]
#set_dont_use [get_lib_cells */NAND*]
set_dont_use [get_lib_cells */XOR*]
#set_dont_use [get_lib_cells */NOR*]
#set_dont_use [get_lib_cells */XNOR*]
#set_dont_use [get_lib_cells */MUX*]

analyze -format verilog ${RTL_SOURCE_FILES}
elaborate ${DESIGN_NAME}
current_design


read_sdc ./../CONSTRAINTS/8bit_comparator.sdc

#compile

compile_ultra
report_timing
write -format verilog -hierarchy -output ${RESULTS_DIR}/${DCRM_FINAL_VERILOG_OUTPUT_FILE}
```

8.3. DC_SHELL_OUTPUT :



8.4. Gate-Level Schematic



8.5. Report Of Time , Power And Qor:

Cell Area	**104.45318**
Slack	**0.35**
Power	**6.8096uW**
Leaf cells	**52**
Invertors	**28**
Buffers	**0**
Combinational Cell Counts	**49**
Sequential Cell Counts	**3**

9. Physical Design using IC Compiler II (ICC2)

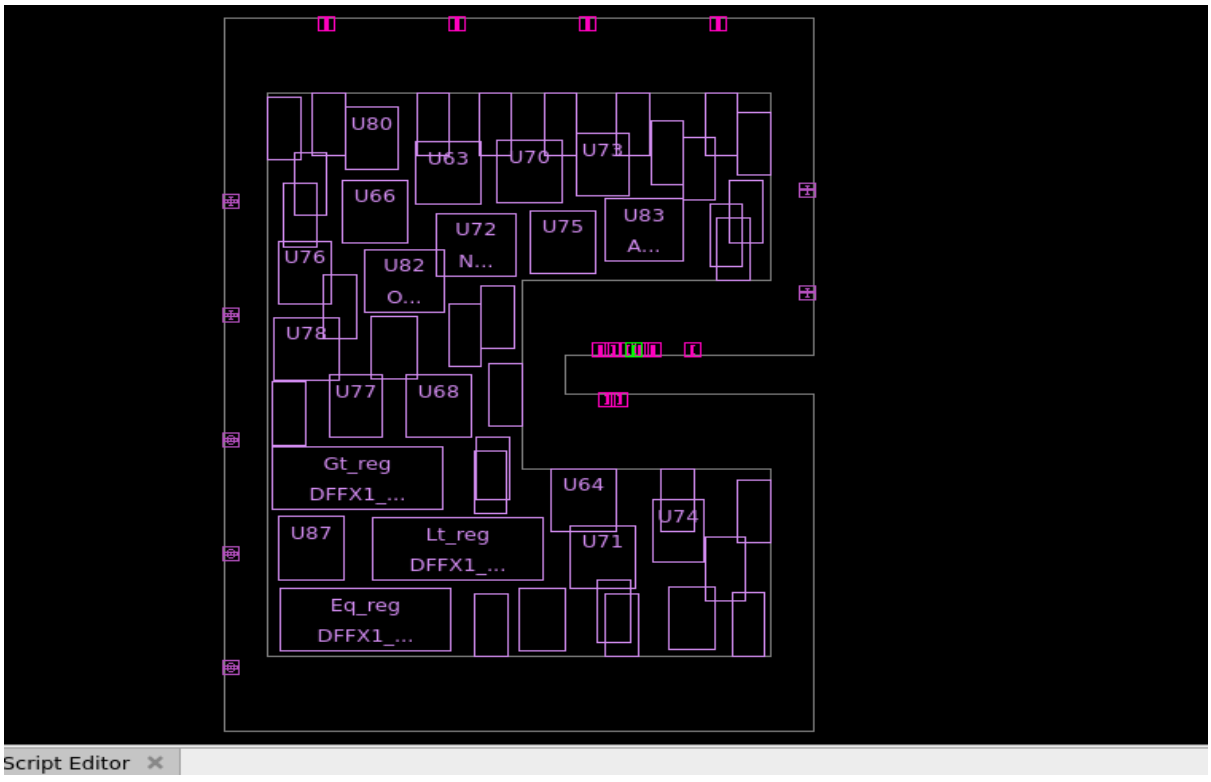
After successful synthesis using Design Compiler, the next step in the ASIC flow is physical implementation using IC Compiler II (ICC2). This stage converts the gate-level netlist into a placed-and-routed layout, optimized for area, timing, and power, eventually producing a GDS-II layout for tape-out.

```
set PDK_PATH ./../ref

create_lib -ref_lib $PDK_PATH/lib/ndm/saed32rvt_c.ndm 8bit_comparator_LIB

read_verilog {./../DC/results/magComp.mapped.v} -library 8bit_comparator_LIB -design magComp -top magComp

#open the lib and block after saving
#open_lib FULL_ADDER_LIB
#open_block FULL_ADD
```

9.1.Floor planning

Floor planning defines the chip's physical dimensions, core area, and I/O pin locations.

```
#scenario8:
initialize_floorplan -core_utilization 0.6 -coincident_boundary true -core_offset {1 2} -shape U -orientation E
set_individual_pin_constraints -ports [get_ports {A[0] B[1]}] -sides "6"
place_pins -self
create_placement -floorplan -effort medium
```



Result:

- Defined die and core area
- Verified design fits within the estimated layout bounds

AFTER FLOOR PLAN REPORT OF TIME, POWER AND QOR:

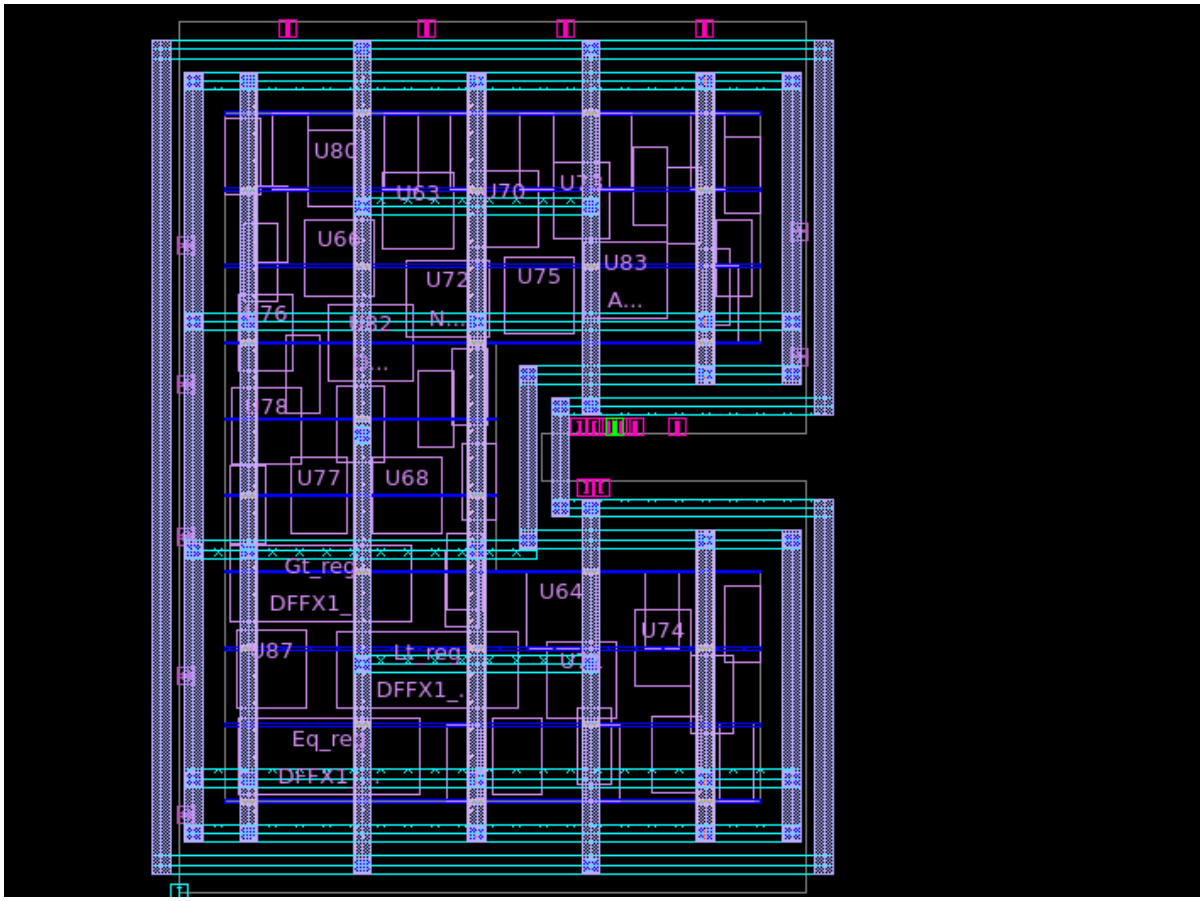
Cell Area	**104.45**
Slack	**1.22**
Power	**1.68e+08pW**
Leaf cells	**52**
Invertors	**28**
Buffers	**0**
Combinational Cell Counts	**49**
Sequential Cell Counts	**3**

9.2. Power Planning

Power planning creates a power distribution network (PDN) to deliver VDD and VSS to all cells.

Key Steps:

- Generated power rings (metal layers around the core)
- Inserted power straps across metal layers (M1–M5)
- Verified power connectivity from rings to standard cells



Result:

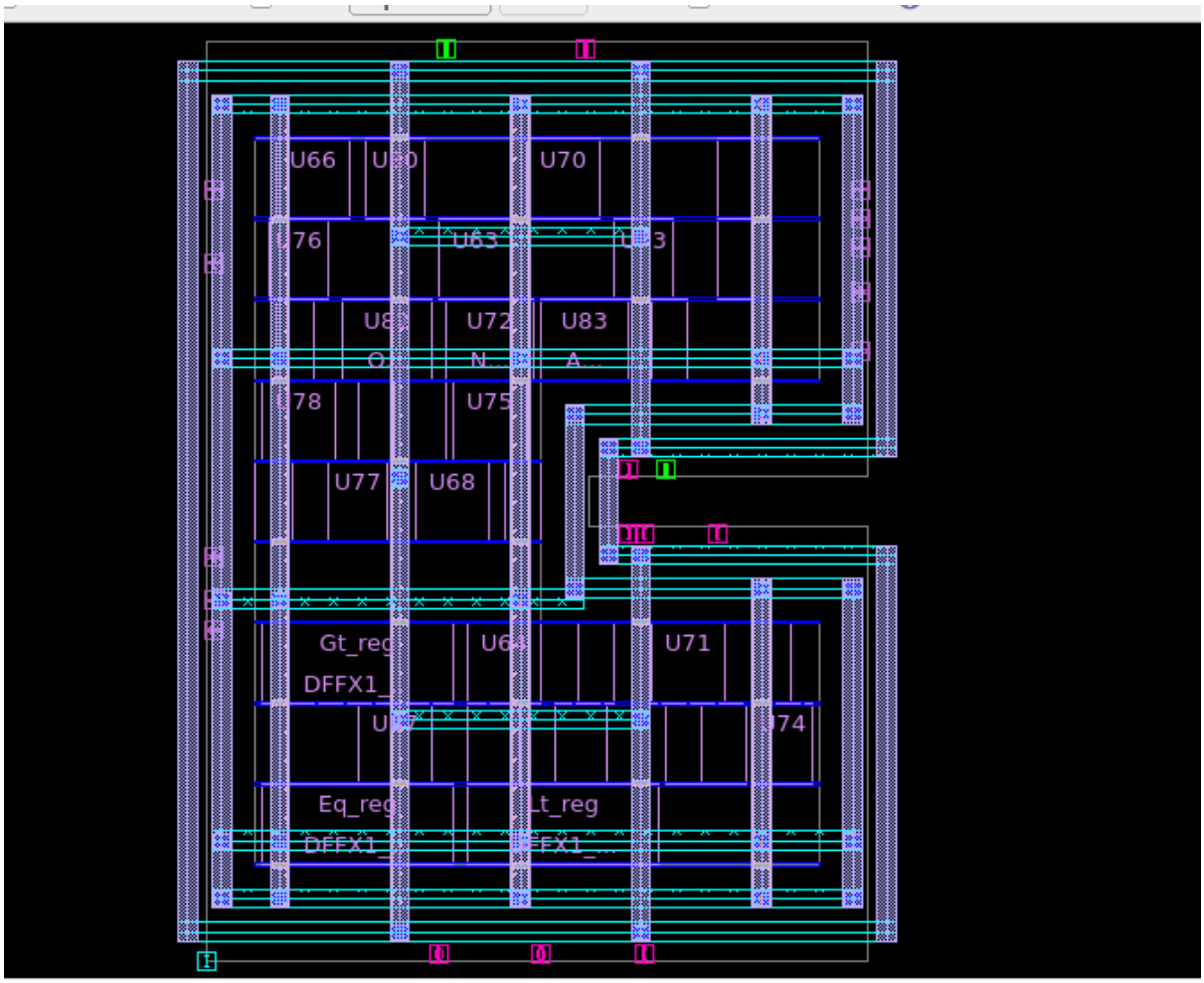
- Uniform power coverage
- Well-distributed power grid with no DRC violations

AFTER POWER PLAN REPORT OF TIME, POWER AND QOR:

Cell Area	**104.45**
Slack	**1.22**
Power	**1.68e+08pW**
Leaf cells	**52**
Invertors	**28**
Buffers	**0**
Combinational Cell Counts	**49**
Sequential Cell Counts	**3**

9.3. Placement

Placement arranges standard cells within the core area to minimize wirelength and satisfy timing.



Result:

- Cells placed without overlaps
- Congestion-free layout with valid net connections

AFTER PLACEMENT REPORT OF TIME, POWER AND QOR:

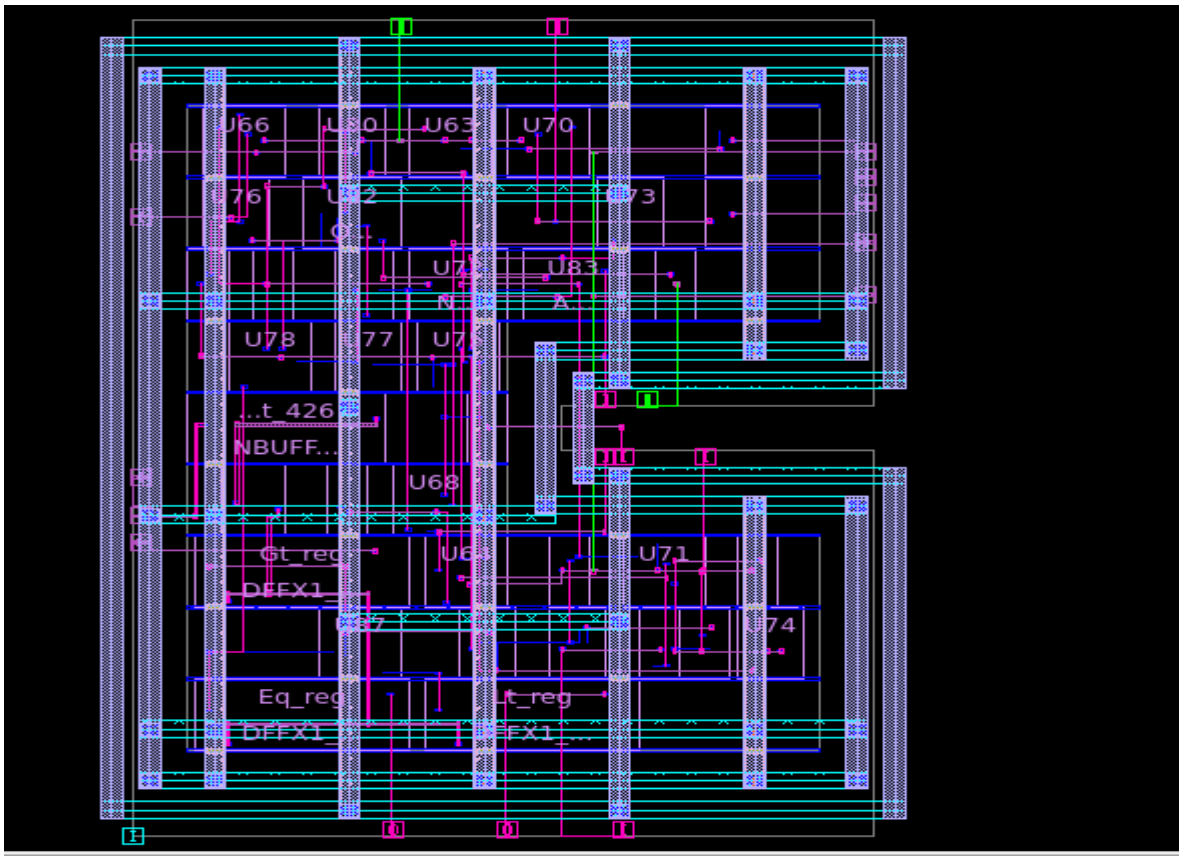
Cell Area	**81.58**
Slack	**1.22**
Power	**1.68e+08pW**
Leaf cells	**34**
Invertors	**10**
Buffers	**0**
Combinational Cell Counts	**31**
Sequential Cell Counts	**3**

9.4.Clock Tree Synthesis (CTS)

CTS distributes the clock signal with minimal skew and latency to all flip-flops.

Key Steps:

- Identified all clock sinks
- Inserted buffers/inverters for tree balancing



Result:

- Balanced clock distribution
- Reduced timing violations
- Verified skew, insertion delay, and latency

AFTER CLOCK REPORT OF TIME, POWER AND QOR:

Cell Area	**81.58**
Slack	**1.22**
Power	**1.68e+08Pw**
Leaf cells	**34**
Invertors	**10**
Buffers	**0**
Combinational Cell Counts	**31**
Sequential Cell Counts	**3**

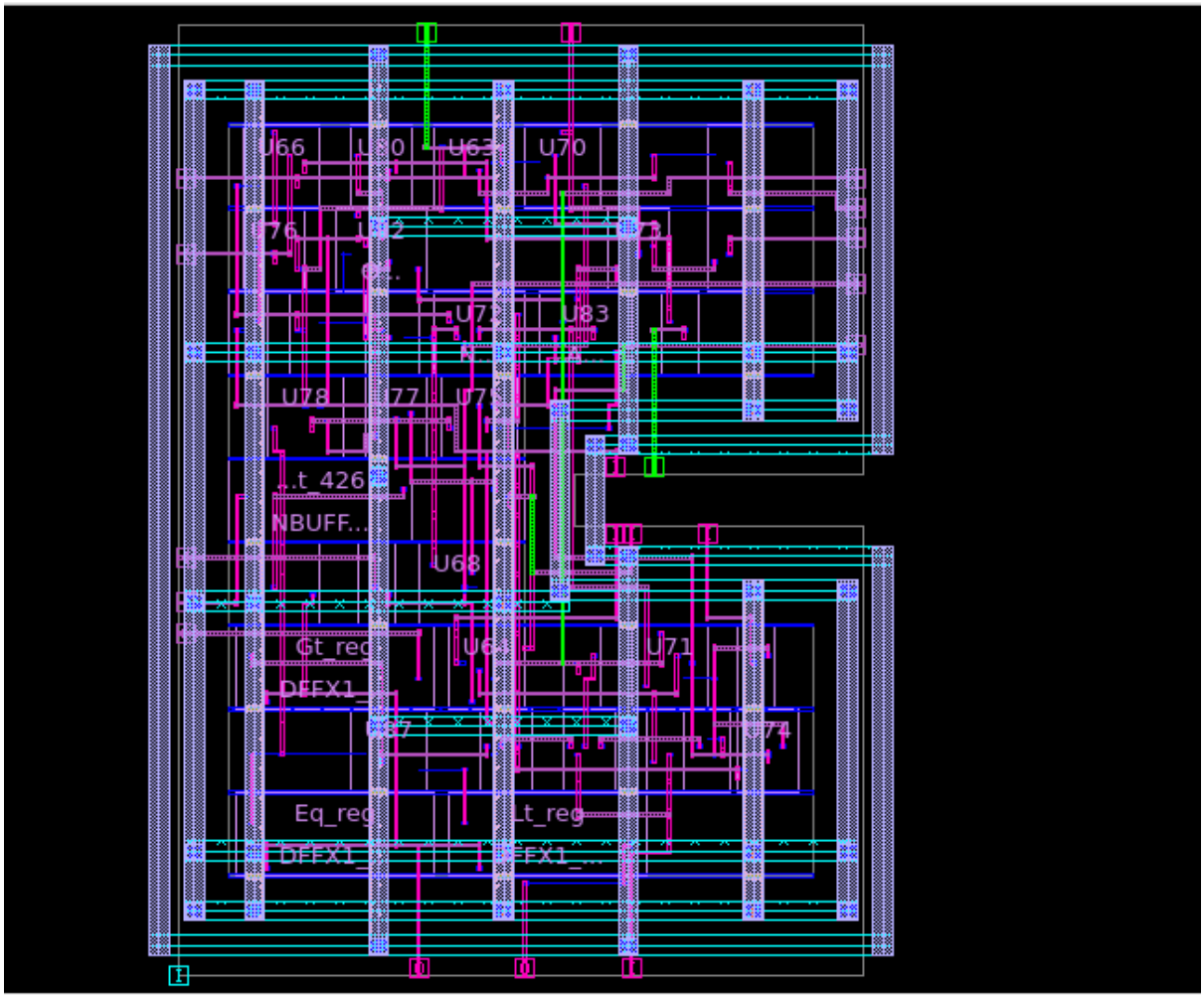
9.5.Routing

Routing connects all logic gates and standard cells using available metal layers.

Key Steps:

- Performed global and detailed routing
- Resolved shorts, opens, and antenna effects
- Checked Design Rule Violations (DRC)

Result:



Result:

- Fully connected design with clean routing
- Final netlist ready for timing and power analysis

AFTER Routing REPORT OF TIME, POWER AND QOR:

Cell Area	**89.71**
Slack	**1.20**
Power	**2.93e+08Pw**
Leaf cells	**36**
Invertors	**10**
Buffers	**02**
Combinational Cell Counts	**33**
Sequential Cell Counts	**3**

9.6.Connectivity Verification

```
Loading cell instances...
Number of Standard Cells: 36
Number of Macro Cells: 0
Number of IO Pad Cells: 0
Number of Blocks: 0
Loading P/G wires and vias...
Number of VDD Wires: 24
Number of VDD Vias: 115
Number of VDD Terminals: 0
**************Verify net VDD connectivity*****************
  Number of floating wires: 0
  Number of floating vias: 0
  Number of floating std cells: 0
  Number of floating hard macros: 0
  Number of floating I/O pads: 0
  Number of floating terminals: 0
  Number of floating hierarchical blocks: 0
************************************************************
Loading cell instances...
Loading P/G wires and vias...
Number of VSS Wires: 24
Number of VSS Vias: 69
Number of VSS Terminals: 0
**************Verify net VSS connectivity*****************
  Number of floating wires: 3
  Number of floating vias: 0
  Number of floating std cells: 0
  Number of floating hard macros: 0
  Number of floating I/O pads: 0
  Number of floating terminals: 0
  Number of floating hierarchical blocks: 0
************************************************************
Overall runtime: 0 seconds.
{PATH_25_4 PATH_25_5 PATH_25_7}
icc2_shell>
```

Conclusion

The 8-bit comparator was successfully implemented from RTL to GDSII using Synopsys flow. The design met all constraints with positive slack and minimal area overhead. Power, timing, and area metrics were within acceptable ranges, confirming the efficiency of our digital design and layout planning**.**

10. Static Timing Analysis using PrimeTime (PT Shell)

The final step in the digital ASIC design flow is to verify the timing of the post-layout netlist using Synopsys PrimeTime (PT Shell). This ensures the design meets timing requirements under the applied constraints with actual delays captured from routing.

10.1 Overview of Timing Signoff

After completing physical design in ICC2, timing delays were extracted and written into a Standard Delay Format (SDF) file. This file, along with the routed netlist and SDC constraints, was loaded into PrimeTime for signoff analysis.

Key Concepts in Timing Analysis:

1. Arrival Time (AT)
    - The actual time at which a signal reaches a specific point (e.g., a cell's input) in the timing path.
2. Required Arrival Time (RAT)
    - The latest time by which a signal must arrive at that point to meet setup or hold timing constraints.
3. Slack Calculation
    - Slack is calculated as:

$$\text{Slack} = \text{RAT} - \text{AT}$$

    - Positive Slack: Timing is met; the signal arrives earlier than required.
    - Negative Slack: Timing is violated; the signal arrives later than it should.

Importance of Slack

Slack is the most critical metric in timing signoff:

- A slack ≥ 0 means the path meets timing.
- A slack < 0 flags a violation, requiring corrective action during optimization or ECO (Engineering Change Order) stages.

10.2. PrimeTime Analysis Flow

Key files used:

Netlist : 8bit_comparator.v

SDF: 8bit_comparator_func_nom.spef.p1_125.spef

Constriants : 8bit_comparator.sdc

10.3. Timing Results

Metric	Value
Worst Slack	**+0.4718**
Clock Period	**2.38 ns**
Critical Path Delay	**0.8797 ns**
Data Required Time	**1.99 ns**
Data Arrival Time	**-1.52 ns**
Clock Uncertainty	**0.3000 ns**
Timing Status	**MET (No violations)**

10.4. Final Analysis Of Time Slack:

```
Point                                        Incr       Path
-------------------------------------------------------------
clock Clock (rise edge)                  0.000000   0.000000
clock network delay (ideal)              0.000000   0.000000
input external delay                     0.500000   0.500000 f
B[3] (in)                                0.000000 & 0.500000 f
U66/Y (AO22X1_RVT)                       0.396751 & 0.896751 f
U56/Y (INVX0_RVT)                        0.073103 & 0.969853 r
U81/Y (NAND3X0_RVT)                      0.060144 & 1.029997 f
U82/Y (OA221X1_RVT)                      0.113529 & 1.143526 f
U83/Y (AO221X1_RVT)                      0.186667 & 1.330194 f
U86/Y (NAND3X0_RVT)                      0.073834 & 1.404028 r
U87/Y (NOR2X0_RVT)                       0.120052 & 1.524080 f
Lt_reg/D (DFFX1_RVT)                     0.000003 & 1.524083 f
data arrival time                                   1.524083

clock Clock (rise edge)                  2.380000   2.380000
clock network delay (ideal)              0.000000   2.380000
Lt_reg/CLK (DFFX1_RVT)                              2.380000 r
clock reconvergence pessimism            0.000000   2.380000
clock uncertainty                       -0.300000   2.080000
library setup time                      -0.084065   1.995935
data required time                                  1.995935
-------------------------------------------------------------
data required time                                  1.995935
data arrival time                                  -1.524083
-------------------------------------------------------------
slack (MET)                                         0.471853
```

10.5. Final Analysis Of Quality Of Result :

```
Timing Path Group 'Clock' (max_delay/setup)
-----------------------------------------------
Levels of Logic:                              7
Critical Path Length:                  1.524083
Critical Path Slack:                   0.471853
Total Negative Slack:                  0.000000
No. of Violating Paths:                       0
-----------------------------------------------

Area
-----------------------------------------------
Net Interconnect area:                 8.107096
Total cell area:                      89.712852
Design Area:                          97.819946
-----------------------------------------------


Cell & Pin Count
-----------------------------------------------
Pin Count:                                  126
Hierarchical Cell Count:                      0
Hierarchical Port Count:                      0
Leaf Cell Count:                             36
-----------------------------------------------


Design Rule Violations
-----------------------------------------------
Total No. of Pins in Design:                126
max_transition Count:                        44
max_transition Cost:                  11.000000
Total DRC Cost:                       11.000000
-----------------------------------------------
```

Conclusion

The design passed final static timing analysis with positive slack. This confirms that:

- All paths meet setup and hold timing
- The physical layout and routing delays are acceptable

11. Learning Outcomes

1. Gained proficiency in Verilog RTL design and behavioral modeling.
2. Understood synthesis constraints and optimization using Design Compiler.
3. Interpreted timing, area, and power reports critically.
4. Executed physical design stages (floorplanning to GDSII).
5. Learned power planning and clock tree insertion techniques.
6. Analyzed DRC and LVS reports to ensure layout correctness.
7. Developed understanding of the full RTL-to-GDSII VLSI flow.
8. Improved debugging and timing closure skills using industry tools.