RTL to GDS-II WORKSHOP

PROJECT REPORT

Submitted by

Chetankumar Suhagiya (23BIT183)

Under Guidance of

Puneet Mittal

SCHOOL OF TECHNOLOGY

PANDIT DEENDAYAL ENERGY UNIVERSITY
GANDHINAGAR

May - 2025

Acknowledgement

I would like to express my sincere gratitude **to Pandit Deendayal Energy University** for providing me with the opportunity and resources to undertake this project titled **"RTL to GDS"** as a part of my academic curriculum.

I am deeply thankful to my project guide **Puneet Mittal** for their valuable guidance, continuous support, and constructive feedback throughout the course of this project. Their insights have been instrumental in shaping the direction and outcome of this work.

I also extend my appreciation to the faculty members and technical staff of the **Information and Communication Technology / Electronics and Communication** for their encouragement and assistance during the project period.

I am grateful to **Synopsys** for the use of their industry-standard EDA tools, which played a critical role in implementing and validating the RTL to GDS flow. The hands-on experience with Synopsys tools significantly enhanced my understanding of the backend design process.

Lastly, I wish to thank my family and peers for their unwavering support and motivation throughout this endeavor.

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, floorplanning, 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.

INDEX

Theoretical Background: RTL to GDSII Flow

1. Introduction

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

It is a highly automated, tool-driven process that demands precise coordination between logical functionality and physical constraints. Mastery of this flow is essential for ensuring performance, power efficiency, and manufacturability in complex integrated circuits.

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 various design steps while ensuring timing, area, power, and reliability constraints are met.

2. Design Flow Overview

The backend design flow consists of several key stages, as illustrated below:

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**

Each of these stages is discussed in the following sections.

3. RTL Design and Synthesis

At the RTL stage, the digital design is described in a Hardware Description Language (HDL) such as Verilog or VHDL. This high-level code represents the functionality and behavior of the digital system without any physical details.

Logic synthesis 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.

Synopsys Design Compiler is used for this task. The tool reads the RTL and maps the design to a target technology library while optimizing for area, timing, and power.

Key outcomes of synthesis include:

- Technology-mapped netlist
- Timing reports
- Area and power estimation

4. Floorplanning

Floorplanning is the process of defining the physical layout of the chip. It involves determining the size and shape of the chip core, placing major functional blocks (macros/IPs), and assigning locations for power/ground pads and I/O pins.

A well-optimized floorplan ensures:

- Minimal congestion
- Effective utilization of area
- Reduced wirelength
- Better timing performance

Synopsys IC Compiler II is used to perform floorplanning and subsequent steps in the flow.

5. Placement

In this stage, standard cells from the synthesized netlist are placed within the defined floorplan. The goal is to position the cells to minimize wirelength and ensure routability, without violating any design rules.

Placement algorithms aim to:

- Minimize total interconnect delay
- Avoid congestion
- Maintain legal spacing between cells

After placement, Design Rule Checks (DRC) and initial timing analysis are performed.

6. Clock Tree Synthesis (CTS)

CTS is responsible for distributing the clock signal evenly across the chip with minimal skew and latency. An efficient clock tree ensures that all sequential elements (flip-flops) receive the clock signal at nearly the same time.

Key goals of CTS:

- Minimize clock skew and insertion delay
- Balance the clock tree
- Ensure low power consumption

The **Clock Tree Compiler** module in Synopsys IC Compiler II is used for this purpose.

7. Routing

Routing connects the placed cells using metal layers defined by the technology library. It is divided into:

- **Global Routing:** Estimates paths between blocks.
- **Detailed Routing:** Finalizes wire tracks, vias, and metal layers.

Routing must obey:

- Design rules for spacing and width
- Layer constraints
- Timing and signal integrity requirements

Routing is a critical stage that greatly affects performance and manufacturability.

8. Physical Verification

After routing, the design undergoes verification to ensure manufacturability and correctness. This includes:

- **Design Rule Check (DRC):** Ensures that the layout adheres to the manufacturing rules provided by the foundry.
- **Layout Versus Schematic (LVS):** Confirms that the physical layout matches the logical netlist.
- **Antenna Check:** Ensures that long metal wires do not accumulate excessive charge during fabrication.

These checks are typically performed using Synopsys or other foundry-certified tools.

9. Static Timing Analysis (STA)

STA is a method of verifying the timing performance of a digital circuit without requiring test vectors. It ensures that the data paths meet the required setup and hold times for all clock domains.

Synopsys PrimeTime is the industry-standard tool for STA. It identifies critical paths, timing violations, and helps guide fixes during optimization.

STA helps validate:

- Setup/hold violations
- Clock skew

- Path delays
- Timing closure

10. GDSII Generation

Once the layout is verified, the final step is to generate the **GDSII** (Graphic Data System II) file. This is the standard format used for transferring the physical layout to the foundry for fabrication.

The GDSII file contains all geometrical and layer information required to create masks for silicon production.

Problem Statement

The objective of this project is to design, synthesize, and implement a digital circuit using Verilog HDL and carry out a complete backend flow to generate its GDSII layout using Synopsys EDA tools. The design must be verified for area, performance, and power efficiency under specified timing constraints.

(a) RTL Design Objective

Design an 8-bit shifter in Verilog HDL capable of shifting its input data either to the left or to the right by a variable number of bit positions (N-bits), where the direction and shift amount are controlled through input signals. The shifter must support the following:

- Logical left shift and logical right shift operations.
- Parametrizable N-bit shift (based on input value).
- Synchronous design suitable for synthesis and physical implementation.

(b) Physical Design and GDSII Generation

Use the Synopsys backend flow to perform RTL-to-GDSII implementation of the 8-bit shifter. The flow should include synthesis, floorplanning, placement, clock tree synthesis (CTS), routing, physical verification, and final layout generation.

The following outcomes must be reported:

1. Design Area:
    - Report the final cell area after place and route.
    - Include both standard cell area and total utilized area.
2. Clock Frequency:
    - Determine the maximum operating frequency such that the timing slack is positive but less than 1 ns.
    - Ensure that the design meets setup and hold timing requirements.
3. Power Consumption:
    - Report total power consumption of the design post-layout.
    - Include dynamic power, leakage power, and clock power.

RTL VERILOG SCRIPT:

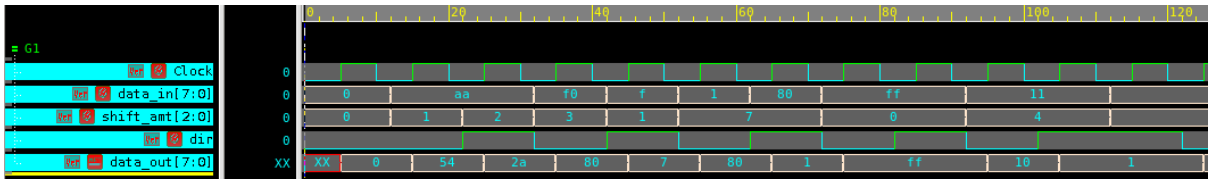
```
module shifter (
    input Clock,
    input [7:0] data_in,
    input [2:0] shift_amt,
    input dir, // 0 = left, 1 = right
    output reg [7:0] data_out
);

reg [7:0] shifted;

always @(*) begin
    if (dir == 1'b0)
        shifted = data_in << shift_amt;
    else
        shifted = data_in >> shift_amt;
end

// Register output at positive clock edge
always @(posedge Clock) begin
    data_out <= shifted;
end

endmodule
```

RTL TEST -BENCH VERILOG SCRIPT :

```
`timescale 1ns/1ns
`include "shifter_rtl.v"

module testbench;
    reg Clock;
    reg [7:0] data_in;
    reg [2:0] shift_amt;
    reg dir;
    wire [7:0] data_out;

    shifter dut (
        .Clock(Clock),
        .data_in(data_in),
        .shift_amt(shift_amt),
        .dir(dir),
        .data_out(data_out)
    );

    always #5 Clock = ~Clock;

    initial begin
        $fsdbDumpvars();

        Clock <= 0;
        data_in <= 0;
        shift_amt <= 0;
        dir <= 0;

        #12 data_in <= 8'b10101010; shift_amt <= 3'd1; dir <= 0;
        #10 data_in <= 8'b10101010; shift_amt <= 3'd2; dir <= 1;
        #10 data_in <= 8'b11110000; shift_amt <= 3'd3; dir <= 0;
        #10 data_in <= 8'b00001111; shift_amt <= 3'd1; dir <= 1;

        #10 data_in <= 8'b00000001; shift_amt <= 3'd7; dir <= 0;
        #10 data_in <= 8'b10000000; shift_amt <= 3'd7; dir <= 1;
        #10 data_in <= 8'b11111111; shift_amt <= 3'd0; dir <= 0;
        #10 data_in <= 8'b11111111; shift_amt <= 3'd0; dir <= 1;
```

```
        #10 data_in <= 8'b00010001; shift_amt <= 3'd4; dir <= 0;
        #10 data_in <= 8'b00010001; shift_amt <= 3'd4; dir <= 1;
        #10 data_in <= 8'b01010101; shift_amt <= 3'd6; dir <= 1;
        #10 data_in <= 8'b01010101; shift_amt <= 3'd6; dir <= 0;

        #50 $finish;
    end
endmodule
```

WAVEFORM FOR VERIFICARION OF TESTBENCH -FILE :



CONSTRAINT-FILE :

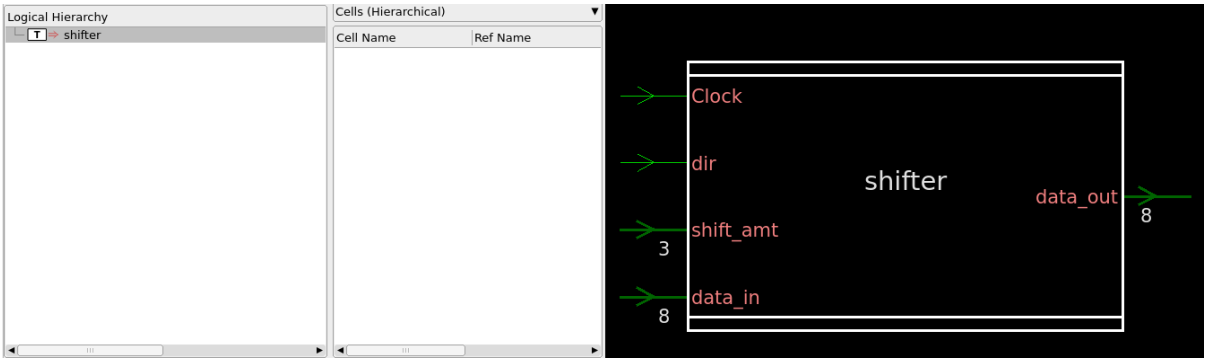
```
create_clock -period 2.5 [get_ports Clock]

set_input_delay -max 0.5 -clock Clock [get_ports data_in]
set_input_delay -max 0.5 -clock Clock [get_ports shift_amt]
set_input_delay -max 0.5 -clock Clock [get_ports dir]

set_input_transition 0.5 [get_ports data_in]
set_input_transition 0.5 [get_ports shift_amt]
set_input_transition 0.5 [get_ports dir]

set_output_delay -max 0.5 -clock Clock [get_ports data_out]

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]
```

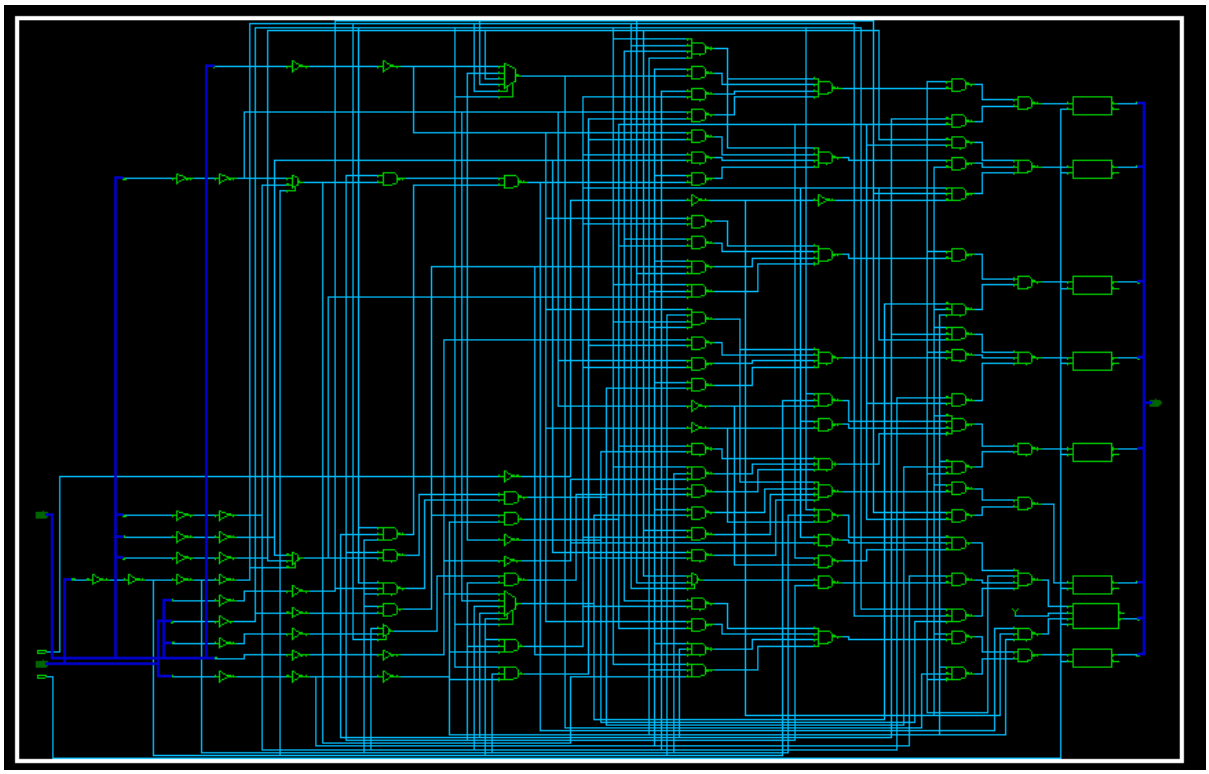
I changed the name of design in dc/rm_setup/common_setup.tcl .
I used lib/stdcell_rvt/saed32rvt_tt0p78vn40c.db.

DC_DESIGN_COMPILE-FILE :

```
source -echo -verbose ./rm_setup/dc_setup.tcl
set RTL_SOURCE_FILES /home/student/RTL2GDSII/rtl/shifter_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 /home/student/RTL2GDSII/CONSTRAINTS/shifter.sdc

#compile

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

DC_SHELL_OUTPUT :





REPORT OF TIME , POWER AND QOR:

SLACK	NO.LEAF-CELL	NO.BUF-CELL	TOTAL-POWER
0.25		32	15.27 uW

ICC2_SHELL_SCRPITS AND OUTPUT :

FLOORPLAN_SHAPE:

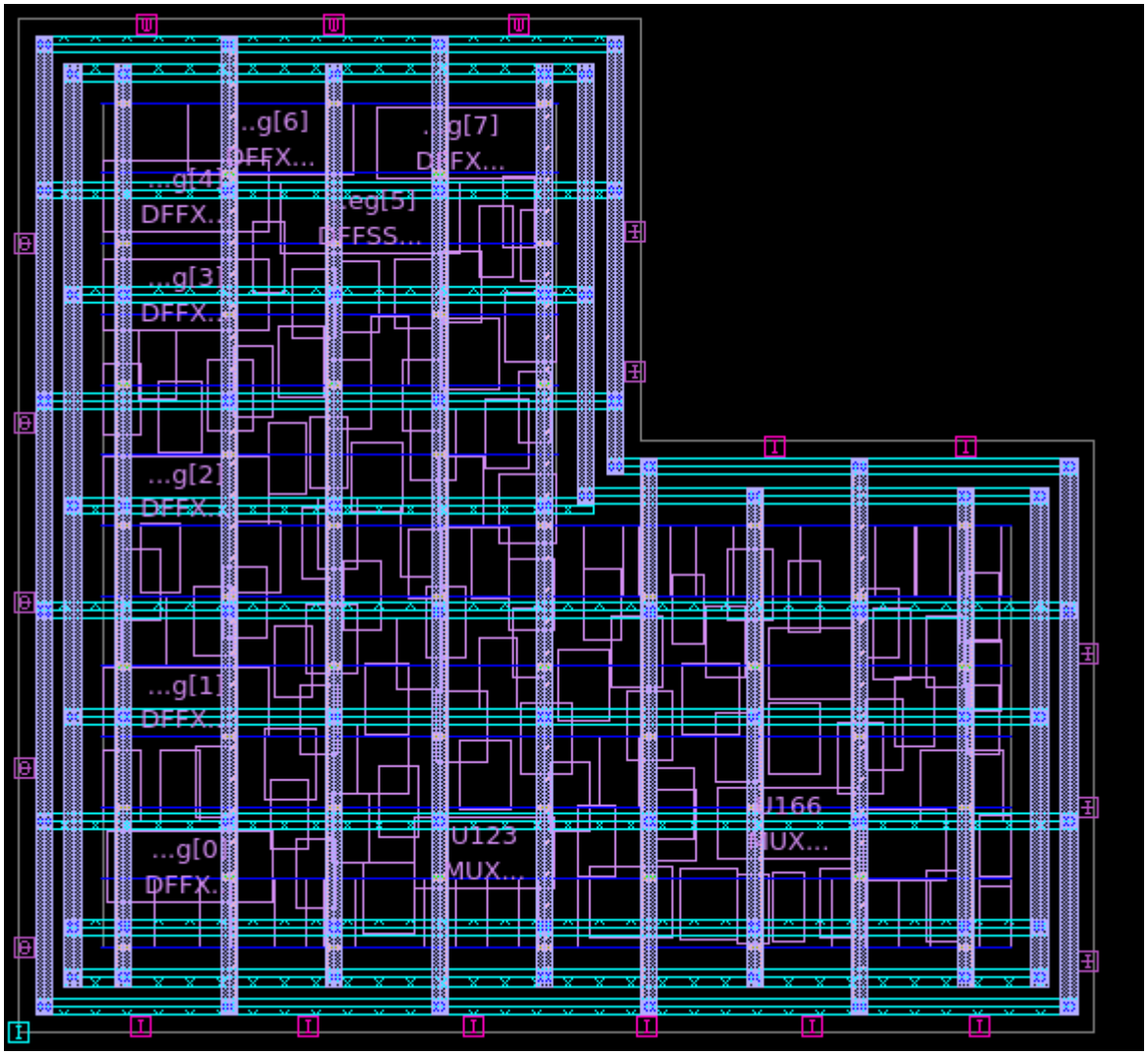
initialize_floorplan -shape L -core_offset 2 -coincident_boundary true
set_individual_pin_constraints -ports [get_ports {A B}] -sides 6
place_pins -self
create_placement -floorplan -effort medium

REPORT OF TIME , POWER AND QOR:

SLACK	NO.LEAF-CELL	NO.BUF-CELL	TOTAL-POWER
1.21	123	32	3.17e+08 PW

POWER_PLANNING:

There is no any change in power_planning script.

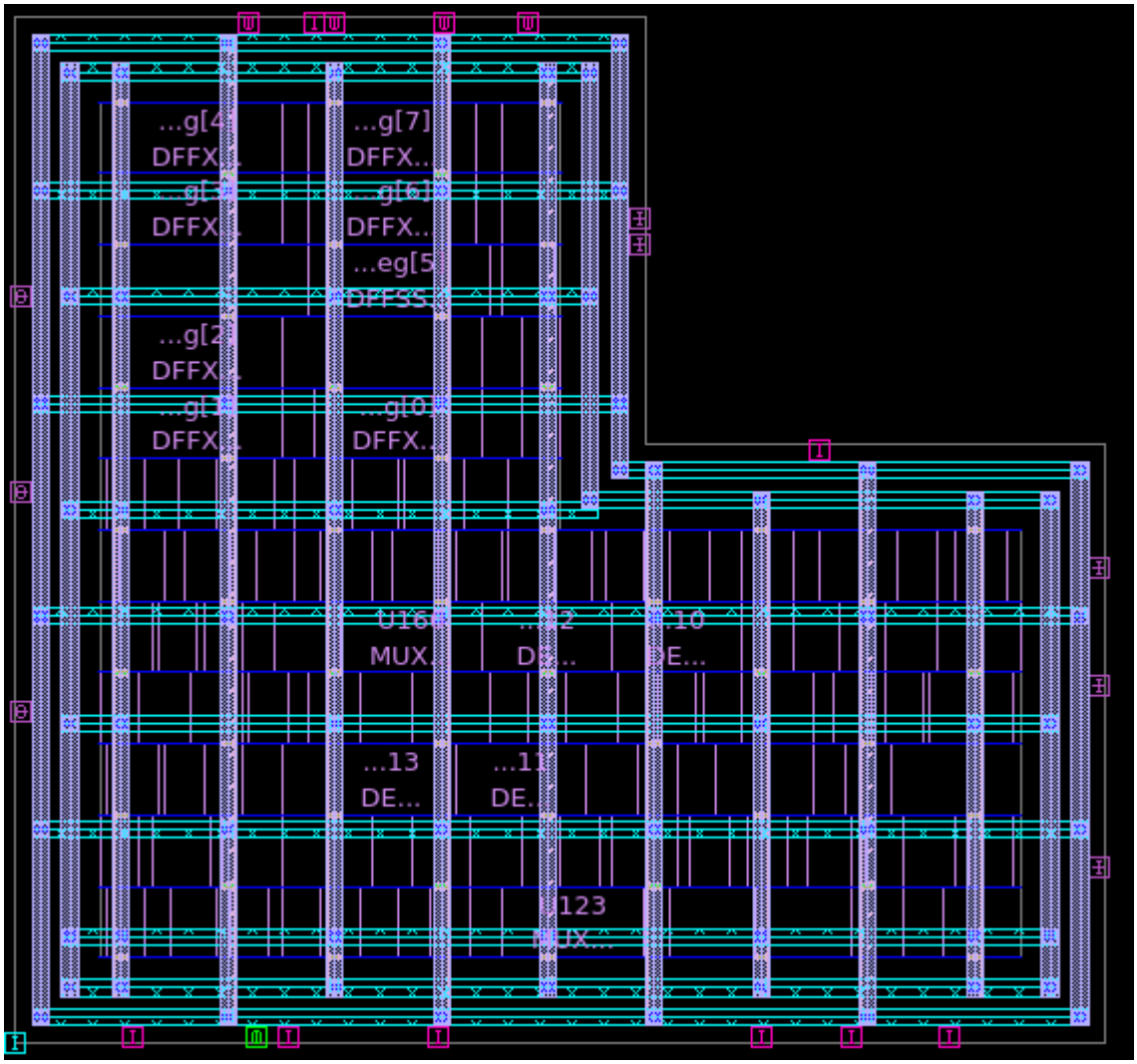


REPORT OF TIME , POWER AND QOR:

SLACK	NO.LEAF-CELL	NO.BUF-CELL	TOTAL-POWER
1.21	123	32	3.17e+08 PW

PLACEMENT :

I saved block as shifter_placement.
I set the path of constraint file.

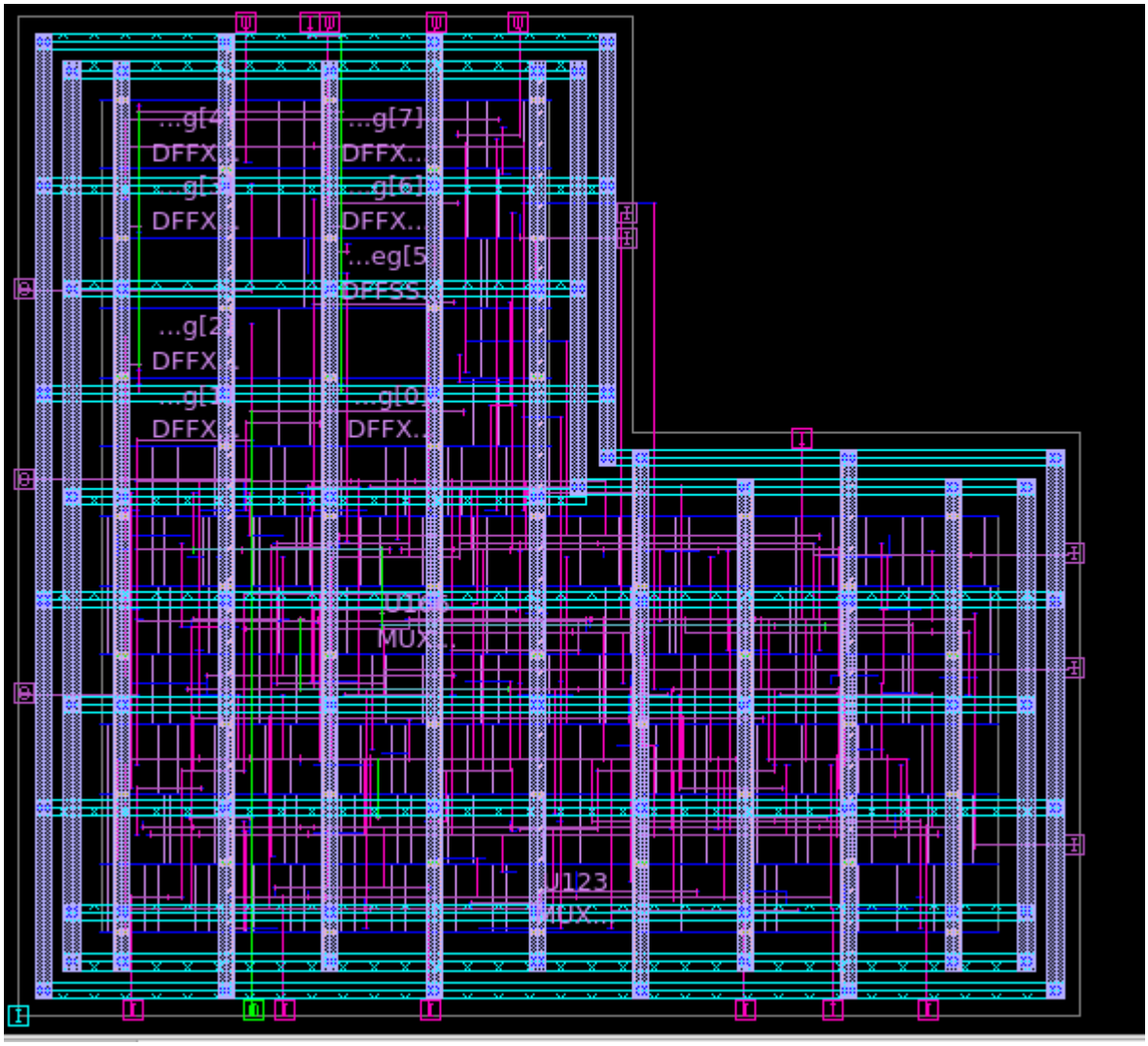


REPORT OF TIME , POWER AND QOR:

SLACK	NO.LEAF-CELL	NO.BUF-CELL	TOTAL-POWER
1.23	123	32	3.28e+08 PW

CLOCK:

I saved block as shifter_cts_CCD.

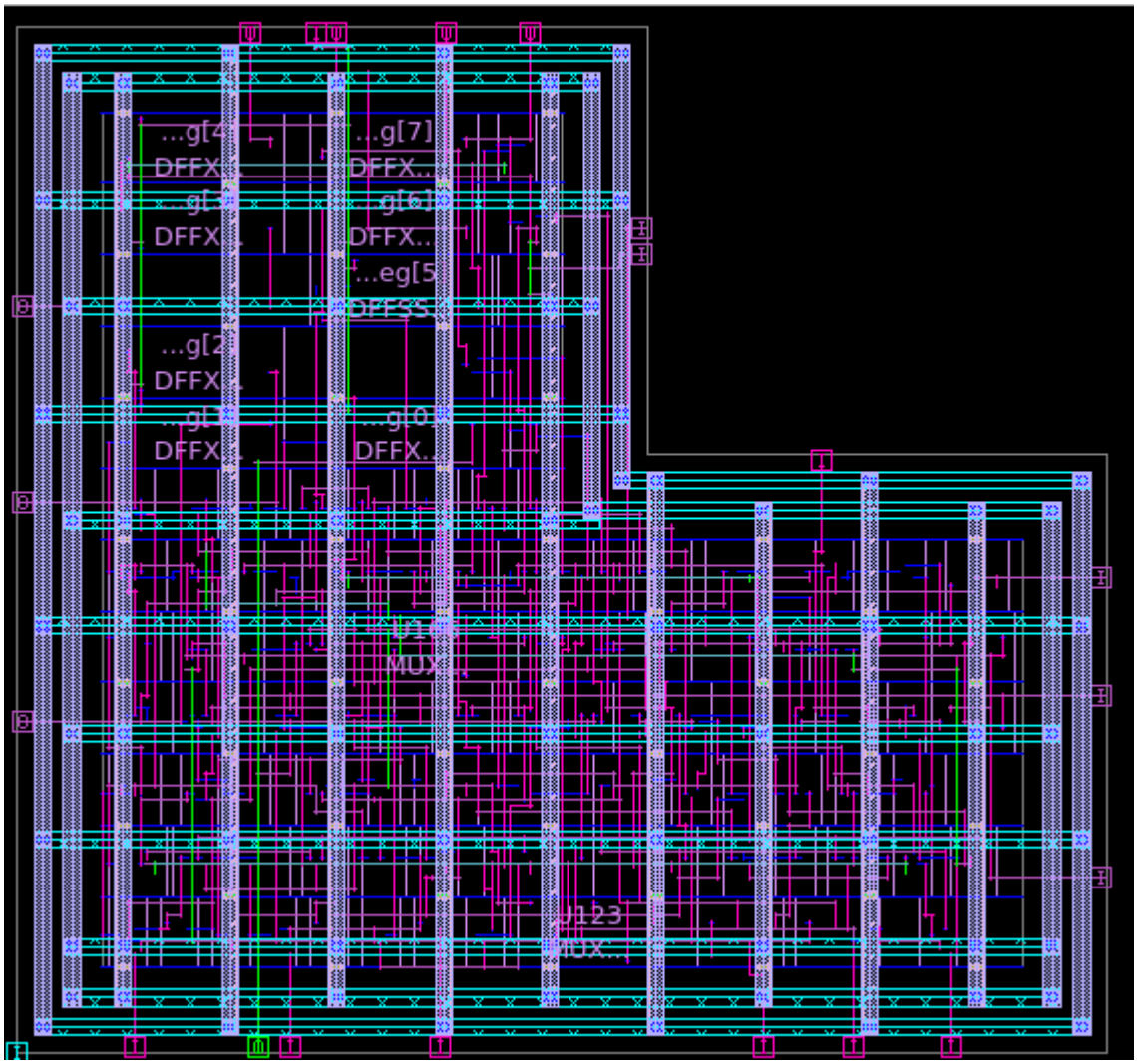


REPORT OF TIME , POWER AND QOR:

SLACK	NO.LEAF-CELL	NO.BUF-CELL	TOTAL-POWER
1.23	123	32	3.51e+08 PW

ROUTE:

I changed the name .v file as shifter.routed.v
I changed the name .sdc file as shifter.routed.sdc



REPORT OF TIME , POWER AND QOR:

SLACK	NO.LEAF-CELL	NO.BUF-CELL	TOTAL-POWER
1.23	123	32	3.49e+08 PW

PT_SHELL :

In Synopsys PrimeTime, pt_cell refers to an instance (cell) within the timing path being analyzed. PrimeTime calculates slack by analyzing the arrival time and required time at each cell in the path. It performs the following steps:

1. Arrival Time (AT): The actual time at which a signal arrives at a cell's input.
2. Required Arrival Time (RAT): The latest time by which a signal must arrive without violating setup or hold constraints.
3. Slack = RAT - AT:
    - Positive slack: The design meets timing.
    - Negative slack: There is a timing violation.

PrimeTime traces the signal through each pt_cell along the path, calculating cumulative delays, taking into account cell delay, net delay, clock skew, and setup/hold margins. This enables identification of the critical path and aids in optimizing cells that are contributing the most delay.

FINAL ANALYSIS OF TIME SLACK:

```
Path Type: max

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 r
shift_amt[0] (in)                                    0.000000   0.500000 r
U120/Y (MUX21X1_RVT)                                 0.483553   0.983553 f
U139/Y (NAND2X0_RVT)                                 0.117420   1.100973 r
U141/Y (NAND2X0_RVT)                                 0.070104   1.171077 f
U142/Y (NAND2X0_RVT)                                 0.100542   1.271619 r
U143/Y (NAND4X0_RVT)                                 0.144881   1.416500 f
U144/Y (NAND2X0_RVT)                                 0.107471   1.523971 r
U146/Y (NAND3X0_RVT)                                 0.115039   1.639010 f
data_out_reg[1]/D (DFFX1_RVT)                        0.012099   1.651109 f
data arrival time                                               1.651109

clock Clock (rise edge)                              2.500000   2.500000
clock network delay (ideal)                          0.000000   2.500000
data_out_reg[1]/CLK (DFFX1_RVT)                                 2.500000 r
clock reconvergence pessimism                        0.000000   2.500000
clock uncertainty                                   -0.300000   2.200000
library setup time                                  -0.140574   2.059426
data required time                                              2.059426
---------------------------------------------------------------------------
data required time                                              2.059426
data arrival time                                               -1.651109
---------------------------------------------------------------------------
slack (MET)                                                     0.408317
```

FINAL ANALYSIS OF QUALITY OF RESULT :

```
  ------------------------------------------------
  Levels of Logic:                               7
  Critical Path Length:                   1.651109
  Critical Path Slack:                    0.408317
  Total Negative Slack:                   0.000000
  No. of Violating Paths:                        0
  ------------------------------------------------

  Area
  ------------------------------------------------
  Net Interconnect area:                 51.405251
  Total cell area:                      231.016769
  Design Area:                          282.422028
  ------------------------------------------------


  Cell & Pin Count
  ------------------------------------------------
  Pin Count:                                   359
  Hierarchical Cell Count:                       0
  Hierarchical Port Count:                       0
  Leaf Cell Count:                             102
  ------------------------------------------------


  Design Rule Violations
  ------------------------------------------------
  Total No. of Pins in Design:                 359
  max_transition Count:                         99
  max_transition Cost:                   24.750000
  Total DRC Cost:                        24.750000
  ------------------------------------------------

1
pt_shell>
```

FINAL ANALYSIS OF POWER :

```
 Attributes
 ----------
    u  -  User defined power group
    i  -  Includes clock pin internal power

Power Group        Internal Power       Switching Power         Leakage Power           Total Power    (   %  )    Attrs
--------------------------------------------------------------------------------------------------------------------
io_pad                  0.00e+00             0.00e+00             0.00e+00             0.00e+00    (  0.0%)
memory                  0.00e+00             0.00e+00             0.00e+00             0.00e+00    (  0.0%)
black_box               0.00e+00             0.00e+00             0.00e+00             0.00e+00    (  0.0%)
clock_network           1.28e+07             0.00e+00             0.00e+00             1.28e+07    (  3.7%)       i
register               -1.50e+07             1.28e+05             1.60e+08             1.45e+08    ( 41.5%)
sequential              0.00e+00             0.00e+00             0.00e+00             0.00e+00    (  0.0%)
combinational           1.87e+07             2.27e+06             1.70e+08             1.91e+08    ( 54.9%)
--------------------------------------------------------------------------------------------------------------------
Total                   1.65e+07 pW          2.40e+06 pW          3.30e+08 pW          3.49e+08 pW
1
icc2_shell>
```

FINAL ANALYSIS OF CLOCK :

```
pt_shell> report_clock
****************************************
Report : clock
Design : shifter
Version: W-2024.09
Date   : Mon Jun  2 07:40:50 2025
****************************************

Attributes:
    p - Propagated clock
    G - Generated  clock
    I - Inactive   clock

Clock             Period          Waveform            Attrs     Sources
------------------------------------------------------------------------
Clock           2.500000          {0.000000 1.250000}           {Clock}

1
pt_shell>
```

FREQUENCY OF SHIFTER :

f = 1/period

f = 1 / 2.5 GHZ

f = 400 MHZ

Changes which are done :

1. I created new constraint file in CONSTRAINT folder.

2. I changed the design name in common_setup.tcl and set the name of library to saed32rvt_tt0p78vn40c.db .

3. I changed the constraint path and also set the use cell into run_dc.tcl file.

4. I changed the name of library and block and also choosed the a suitable scenario and these changes are done in floorplan.tcl file.

5. There is no any changes in power_planning.tcl file.

6. I set the right constraint path in placement.tcl and saved the block as shifter_placement.

7. Same changes are done in clock.tcl and saved the block as shifter_cts_CCD.

8. Same changes are done in route.tcl and I changed the name of output Verilog , sdc and parasitics file name in route.tcl file.

9. I set the right path of my library and changed the current_design name to shifter and set the right path of constraint path in run_p1.tcl file.

10. I wrote the Verilog code and test bench code of 8 bit shifter and saved those code in rtl and rtl_simulation folder as shifter_rtl.v and shifter_tb.v

Learning and Outcomes

Learnings:

1. End-to-End Physical Design Flow:

- Gained hands-on experience in implementing the complete VLSI backend design flow—from RTL to GDSII.
- Understood each phase including synthesis, floorplanning, placement, CTS, routing, and sign-off.

2. Tool Proficiency:

- Developed working knowledge of Synopsys EDA tools such as Design Compiler, IC Compiler II, and PrimeTime.
- Learned how to write and debug TCL scripts for design automation.

3. Constraint Management:

- Learned how to define and apply SDC (Synopsys Design Constraints) for realistic timing, area, and power targets.
- Applied proper input/output delays, transitions, and clock uncertainties to reflect practical operating environments.

4. Timing and Power Optimization:

- Understood the relationship between design decisions and timing slack, and how to interpret timing reports.
- Performed timing analysis and power estimation, and validated the design met required frequency (400 MHz) with positive slack.

5. Design Verification and Sign-Off:

- Conducted physical verification through DRC and LVS to ensure layout correctness.
- Carried out Static Timing Analysis (STA) using PrimeTime to confirm setup and hold time requirements were met.

6. **Design Quality Metrics:**

    - Interpreted QoR (Quality of Results) metrics such as slack, cell count, buffer count, and total power consumption.
    - Improved understanding of how area, power, and performance trade-offs are managed during implementation.

7. **Project Execution and Debugging:**

    - Learned to systematically debug design issues, set proper file paths, modify constraints, and interpret tool feedback.
    - Successfully navigated complex tool setups, file structures, and error messages to drive the design to completion**.**

Outcomes:

- **Functional RTL Design**: Developed and verified a synthesizable 8-bit bidirectional shifter with parameterizable shift amount using Verilog HDL.

- **Positive Timing Closure**: Achieved a final slack of +0.4083 ns, validating that timing constraints were met across all paths.

- **Efficient Power Usage:** Final post-route power consumption was ~349 µW, demonstrating good power optimization.

- **High Performance:** Design supported a frequency of 400 MHz, suitable for high-speed digital applications.

- **Tape-Out Ready GDSII**: Successfully generated a GDSII layout file ready for silicon fabrication.

This project provided a deep and practical insight into the semiconductor industry's backend design practices and enhanced the understanding of physical design constraints, tools, and workflows, preparing for advanced roles in ASIC and SoC design.

Conclusion :

This project successfully demonstrated the complete physical design flow from RTL to GDSII using Synopsys industry-standard EDA tools. Starting from Verilog-based RTL design of an 8-bit shifter, the design was synthesized, floorplanned, placed, routed, and verified through a systematic and automated flow.

Throughout the stages—synthesis, placement, clock tree synthesis, and routing—timing, area, and power parameters were monitored and optimized to meet design constraints. The final post-layout results indicate:

- **Slack** of **+0.4083 ns**, confirming that the design is timing clean.
- **Total power consumption** of approximately **349 µW**, ensuring power-efficient operation.
- **Design frequency** achieved is **400 MHz**, based on a clock period of 2.5 ns.
- **Logical and physical correctness** were validated via DRC/LVS and STA using PrimeTime.

This hands-on project not only reinforced theoretical concepts of VLSI backend flow but also provided valuable experience in dealing with real-world physical design challenges. The entire process—from RTL to tape-out-ready GDSII—reflects industry practice and has laid a strong foundation for future work in ASIC design and digital chip implementation.