Semi Custom Design

Non Restoring Division Algorithm

Project report submitted for

Vth Semester Course Project

in

Department of Electronics and Communication Engineering

By,

Bhavesh Balendra (211010221)

Soumya Rayast (211010248)

Guided By: Dr. Rohit Chaurasiya

Department of Electronics and Communication Engineering

Dr. Shyama Prasad Mukherjee

International Institute of Information Technology, Naya Raipur

(A Joint Initiative of Govt. of Chhattisgarh and NTPC)

Email: iiitnr@iiitnr.ac.in, Tel: (0771) 2474040, Web: www.iiitnr.ac.in

Non Restoring Division Algorithm

Introduction

The **Non-restoring division algorithm** is a binary division method employed in computer arithmetic. Unlike the restoring division algorithm, it takes a non-restoring approach by avoiding the restoration of the partial remainder to its original value after each subtraction. The primary objective is to efficiently divide a binary number (the dividend) by another (the divisor), yielding both the quotient and remainder. The algorithm initializes the partial remainder to the dividend and the quotient to zero. Through an iterative process, it performs a series of subtract and adjust operations based on the divisor, updating the quotient bit by bit. The final quotient represents the division result, and the remaining partial remainder serves as the remainder. Non-restoring division finds application in hardware implementations, providing a balance between simplicity and speed. While it simplifies hardware design compared to restoring division, it may necessitate additional steps for correction when the partial remainder is negative. Overall, this algorithm's efficiency and suitability for parallel processing make it a valuable tool in binary division within computational systems.

The non-restoring division algorithm offers several advantages, making it a preferred choice in certain applications:

Simplified Hardware Implementation:

Non-restoring division simplifies hardware design compared to restoring division. The avoidance of the restoration step reduces the complexity of the hardware circuitry required for the division process, making it more efficient in terms of resources.

Parallel Processing Potential:

The algorithm's structure lends itself well to parallel processing and pipelining. This means that multiple steps of the division process can be carried out simultaneously, enhancing the overall speed and efficiency of the computation.

Efficiency in Digital Systems:

In digital systems and hardware implementations, efficiency is a critical factor. Non-restoring division strikes a balance between simplicity and performance, making it particularly suitable for use in digital circuits and processors where speed and resource optimization are essential.

Reduced Latency:

Due to its non-restoring nature, the algorithm tends to have lower latency in comparison to some other division methods. This characteristic makes it advantageous in applications where minimizing processing time is crucial.

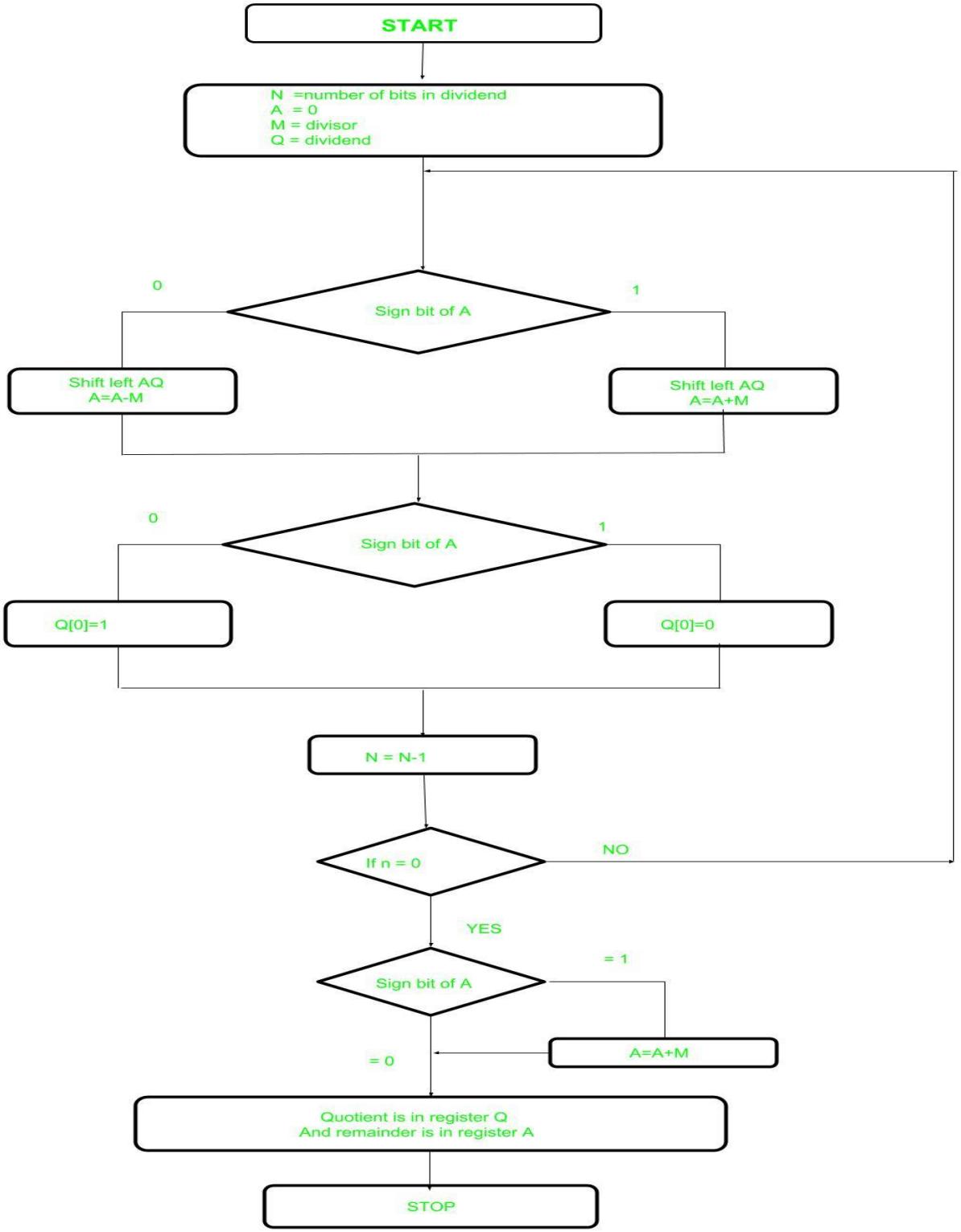
Parallelization of Correction Steps:

While non-restoring division may require additional correction steps when the partial remainder becomes negative, these steps can also be parallelized. This parallelization helps mitigate the impact on overall processing time, maintaining the algorithm's efficiency.

Balance Between Complexity and Performance:

The algorithm strikes a balance between the simplicity of hardware implementation and computational performance. This balance is especially advantageous in scenarios where hardware resources are limited, and a streamlined design is desired.

Flowchart


START
N =number of bits in dividend
A = 0
M = divisor
Q = dividend
Sign bit of A
0
1
Shift left AQ
A=A-M
Shift left AQ
A=A+M
Sign bit of A
0
1
Q[0]=1
Q[0]=0
N = N-1
If n = 0
NO
YES
Sign bit of A
= 1
= 0
A=A+M
Quotient is in register Q
And remainder is in register A
STOP

Code

div_nonrestoring.v

```
`timescale 10ps / 1ps
module div_nonrestoring (a, b, start, clk, clrn, q, r, busy, ready, count);
  input [31:0] a; // dividend
  input [15:0] b;
  input start; // start
  input clk, clrn; // clk,reset
  output [31:0] q; // quotient
  output [15:0] r; // remainder
  output reg busy; // busy
  output reg ready; // ready
  output [4:0] count; // count
  reg [31:0] reg_q;
  reg [15:0] reg_r;
  reg [15:0] reg_b;
  reg [4:0] count;
  wire [16:0] sub_add = reg_r[15] ? (reg_r + {1'b0, ~reg_b}) : (reg_r - {1'b0, reg_b}); //
Corrected subtraction and addition
  assign q = reg_q;
  assign r = reg_r[15] ? (reg_r + reg_b) : reg_r; // adjust r

  always @(posedge clk or negedge clrn) begin
    if (!clrn) begin
      busy <= 0;
      ready <= 0;
    end else begin
```

```
      if (start) begin
        reg_q <= a; // load a
        reg_b <= b; // load b
        reg_r <= 0;
        busy <= 1;
        ready <= 0;
        count <= 0;
      end else if (busy) begin
        reg_q <= {reg_q[30:0], ~sub_add[16]}; // << 1, Corrected concatenation
        reg_r <= sub_add[15:0];
        count <= count + 5'b1; // count++
        if (count == 5'h1f) begin // finish
          busy <= 0;
          ready <= 1; // q,r ready
        end
      end
    end
  end
endmodule
```

Testbench : div_nonrestoring.tb

```
`timescale 10ps / 1ps
module div_nonrestoring_tb;
  reg [31:0] a;
  reg [15:0] b;
  reg start;
  reg clk, clrn;
  wire [31:0] q;
```

```
  wire [15:0] r;
  wire busy, ready;
  wire [4:0] count;
div_nonrestoring dut(a, b, start, clk, clrn, q, r, busy, ready, count);
initial begin
a = 0;
forever #20 a = 0;
end
initial begin
b = 0;
forever #15 b = ~b;
end
initial begin
clk = 0;
forever #30 clk = ~clk;
end
initial begin
clrn = 0;
#10 clrn = 1;
end
initial begin
start = 0;
forever #40 start = ~start;
end
endmodule
```

Semi Custom

In VLSI (Very Large Scale Integration) fabrication, "semi custom" refers to an approach that lies between full custom design and fully standardized, off-the-shelf solutions. Semi Custom design involves using predefined, partially customized components or blocks, typically through standard cell libraries or gate arrays. Designers have flexibility in configuring and connecting these blocks, allowing for a balance between customization and design efficiency. This approach strikes a middle ground, offering some level of customization while benefiting from the standardization of certain components, leading to a more cost-effective and time-efficient VLSI design process.

UMC 40nm Library

UMC's 40nm library is a collection of semiconductor design components tailored for the 40nm process node in VLSI fabrication. This library includes standard cells, I/O cells, and other essential elements that enable chip designers to create integrated circuits efficiently. UMC, a semiconductor foundry, provides these resources to support the development of advanced and cost-effective electronic devices using their 40nm process technology. For the latest and most accurate details, it is recommended to refer to UMC's official documentation or contact UMC directly.

Cadence

Cadence Design Systems is a leading provider of electronic design automation (EDA) software, offering a suite of tools to facilitate the design and verification of integrated circuits. Here's a brief introduction to the Cadence tools that we used in our semi custom design : NClaunch, Innovus, and Genus.

NClaunch:

Role: NClaunch is a command-line tool within the Cadence environment, serving as a frontend for managing the execution of various Cadence tools.

Functionality: It provides a convenient interface for launching and managing simulations, analyses, and other design tasks. Designers can use it to streamline and automate the execution of different EDA processes in a Cadence environment.

Innovus:

Role: Innovus is Cadence's place and route solution, a critical step in the physical design of integrated circuits.

Functionality: Innovus optimizes the placement and routing of standard cells, ensuring efficient use of chip area while meeting performance, power, and timing constraints. It incorporates advanced algorithms to enhance the overall quality of the design and is particularly valuable in achieving high-performance and low-power IC implementations.

Genus:

Role: Genus is Cadence's logic synthesis tool, a key component in the early stages of the digital design process.

Functionality: Genus transforms a high-level RTL (Register-Transfer Level) description of a design into an optimized netlist, mapping the functionality into a gate-level representation. It focuses on improving design quality, reducing power consumption, and optimizing for performance. Genus plays a crucial role in creating a solid foundation for subsequent steps in the IC design flow.

RTL Design

RTL, or Register-Transfer Level, is a pivotal abstraction in digital design. It acts as a bridge between high-level circuit behavior and low-level gate-level implementation. At this level, designers describe the circuit using registers, data transfers, and control logic. It captures the essential flow of data and control, detailing how information moves through the system. This abstraction is crucial for synchronous digital design, aligning operations with a clock signal. RTL serves as a foundation for synthesis, translating designs into gate-level netlists, and for rigorous

verification through simulation. In essence, RTL simplifies the complexity of digital functionality, enabling an organized and modular approach in the circuit design process.

GDS File

A GDS (Graphic Database System) file, commonly known as GDSII, is a binary file format crucial in the semiconductor industry for representing the geometric layout of integrated circuits. It encapsulates information about different layers, shapes, and their spatial relationships, organizing data into records that define elements like polygons, paths, and text labels. GDS files serve as a standard means for exchanging design information between Electronic Design Automation (EDA) tools, enabling interoperability in the IC design flow. They play a vital role in semiconductor fabrication, aiding in the generation of photomasks used during manufacturing. Widely supported by various EDA tools, GDS files simplify the visualization and manipulation of layout data throughout the design process, contributing to the efficiency and standardization of integrated circuit design.

RTL to GDS

Using Cadence tools for the RTL-to-GDS design flow, the steps and tools involved can vary depending on the specific tools within the Cadence suite and the semiconductor process technology. Below is a generalized overview of the process using popular Cadence tools:

1. RTL Design:
    a. Using a hardware description language (HDL) such as Verilog or VHDL to describe the digital circuit at the register-transfer level.
2. Functional Verification:
    a. Simulate the RTL code using tools like Cadence Incisive or Nclaunch for functional verification.
3. Synthesis:
    a. Use Cadence Genus for logic synthesis to convert RTL to a gate-level netlist.
4. Netlist Optimization:
    a. Optimize the synthesized netlist using Genus or other optimization tools.

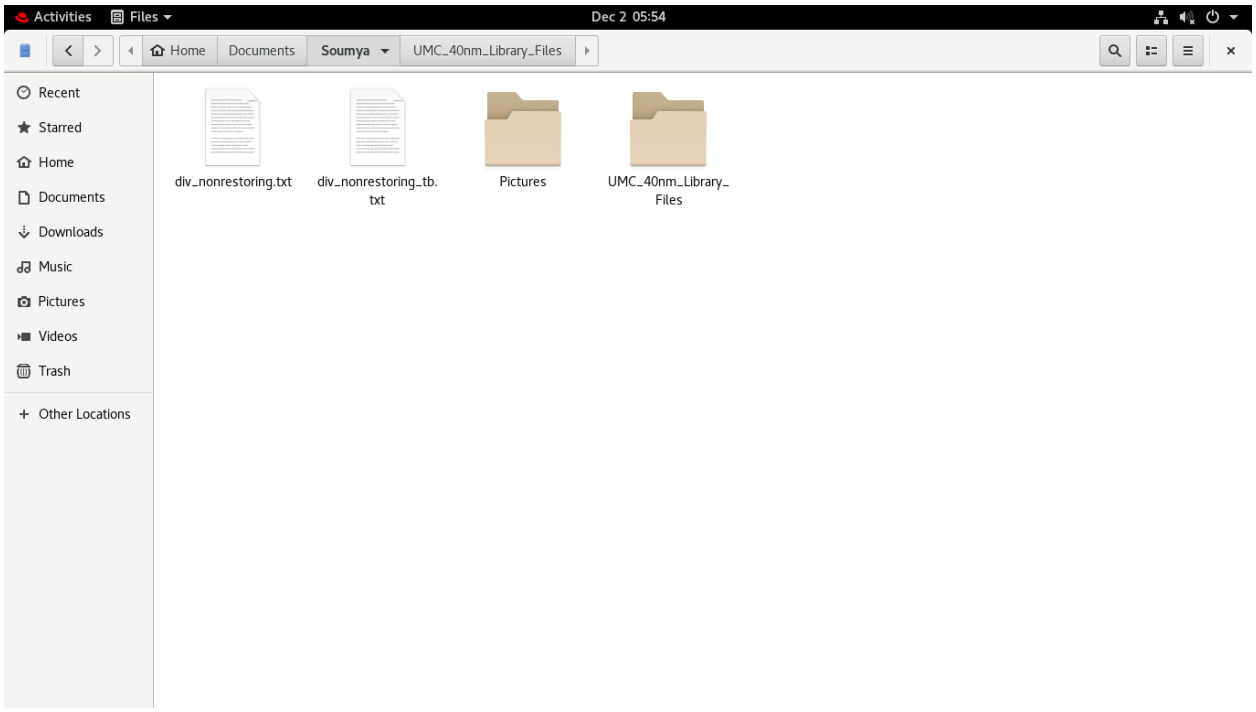
5. Static Timing Analysis (STA):
    a. Perform STA using tools like Cadence Tempus to ensure timing requirements are met.
6. Floorplanning:
    a. Use the Cadence Innovus tool for floorplanning, defining the placement of blocks and regions on the chip.
7. Placement:
    a. Place standard cells and other components on the chip using Innovus.
8. Clock Tree Synthesis (CTS):
    a. Perform CTS using Innovus to implement an optimized clock distribution network.

9. Routing:
    a. Utilize the Innovus tool for detailed routing to connect the placed components.
10. Design Rule Check (DRC):
    a. Perform DRC using Cadence PVS (Physical Verification System) to ensure the layout adheres to manufacturing rules.
11. Layout vs. Schematic (LVS) Check:
    a. Perform LVS checks using Cadence Assura to verify that the layout matches the schematic.
12. Extraction:
    a. Use Cadence Quantus QRC for parasitic extraction to include information about resistances and capacitances.
13. Physical Verification:
    a. Perform physical verification checks using Cadence tools, including checks for reliability and manufacturability.
14. GDS Generation:
    a. Generate the GDSII file using the Cadence Virtuoso layout editor.

15. Tape-Out:
    a. Prepare the GDSII file for submission to the semiconductor foundry for manufacturing.

Semi Custom process :

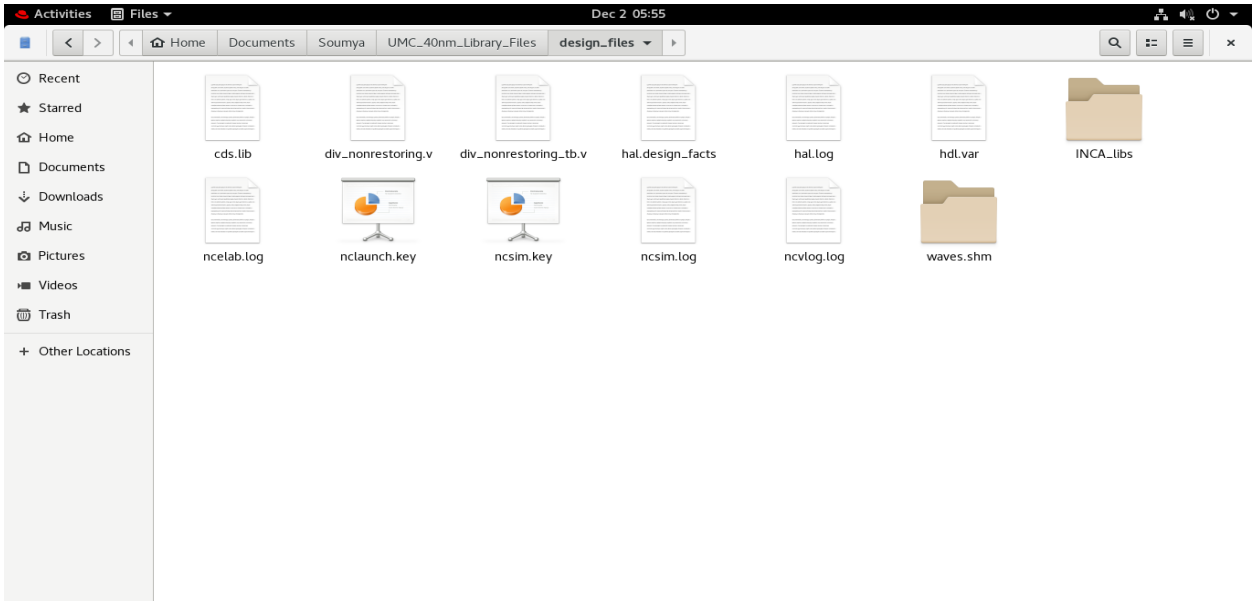
File

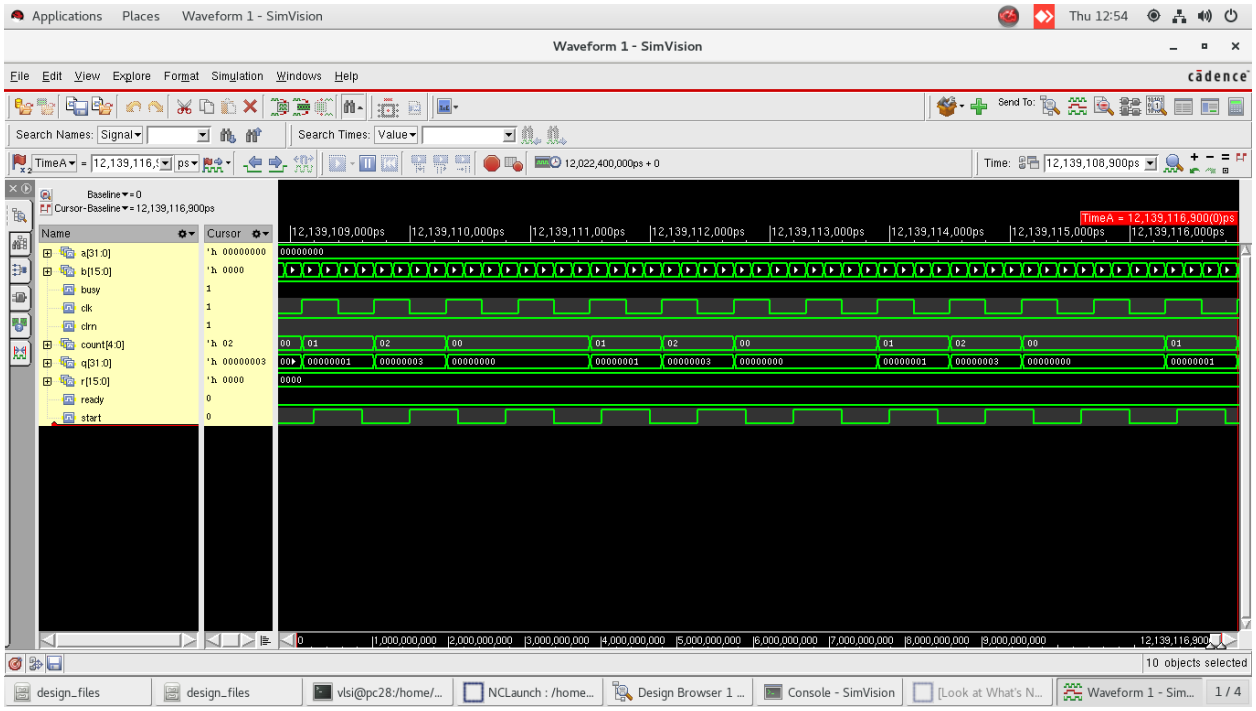


Nclaunch

After uploading the netlist files

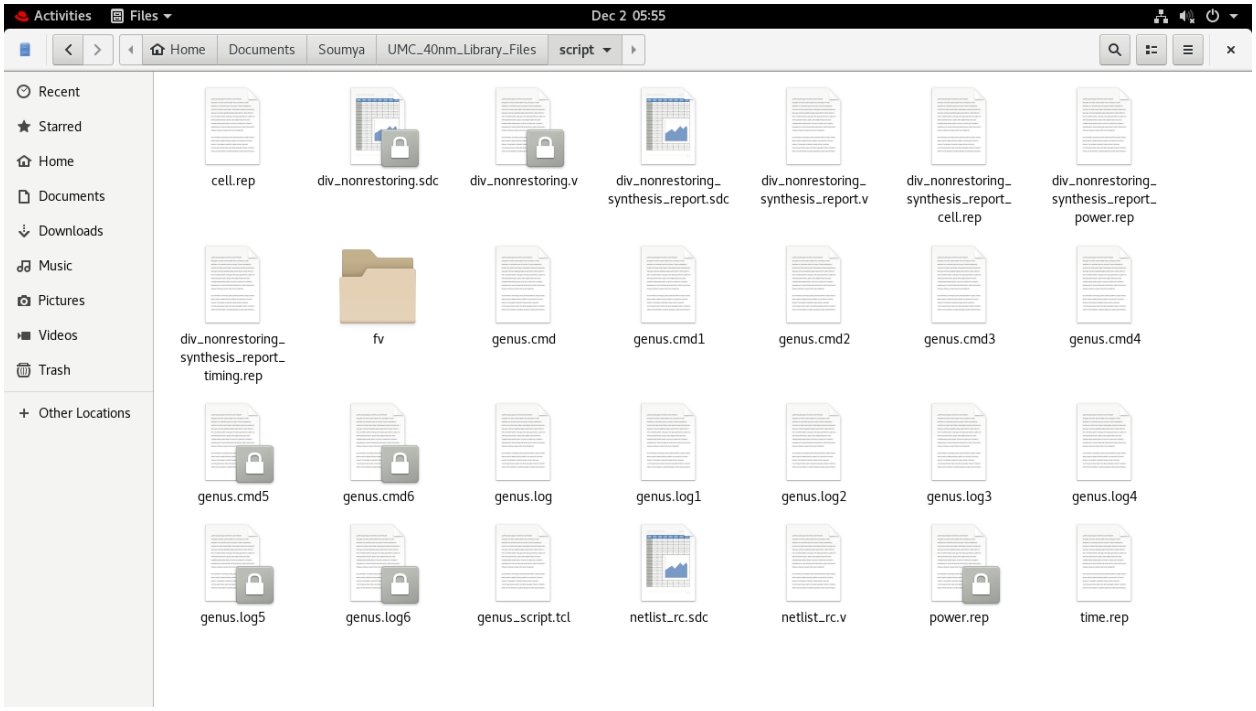
div_nonrestoring.v Div_nonrestoring_tb.v

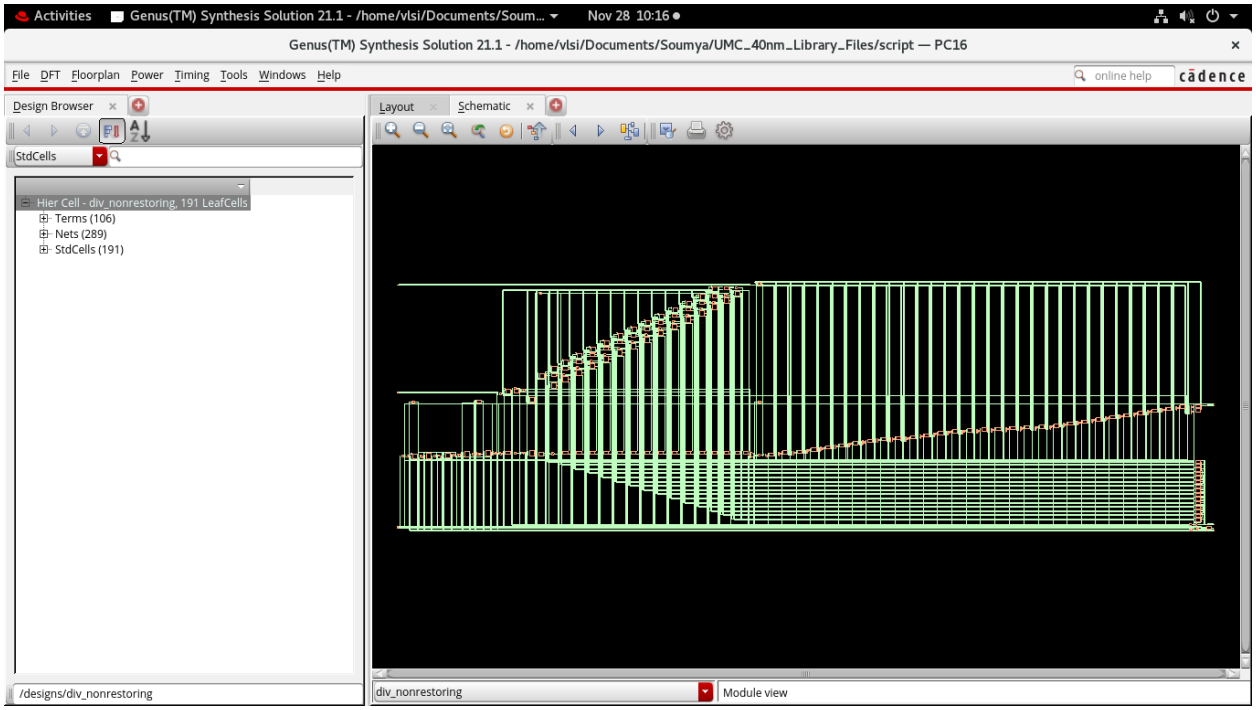


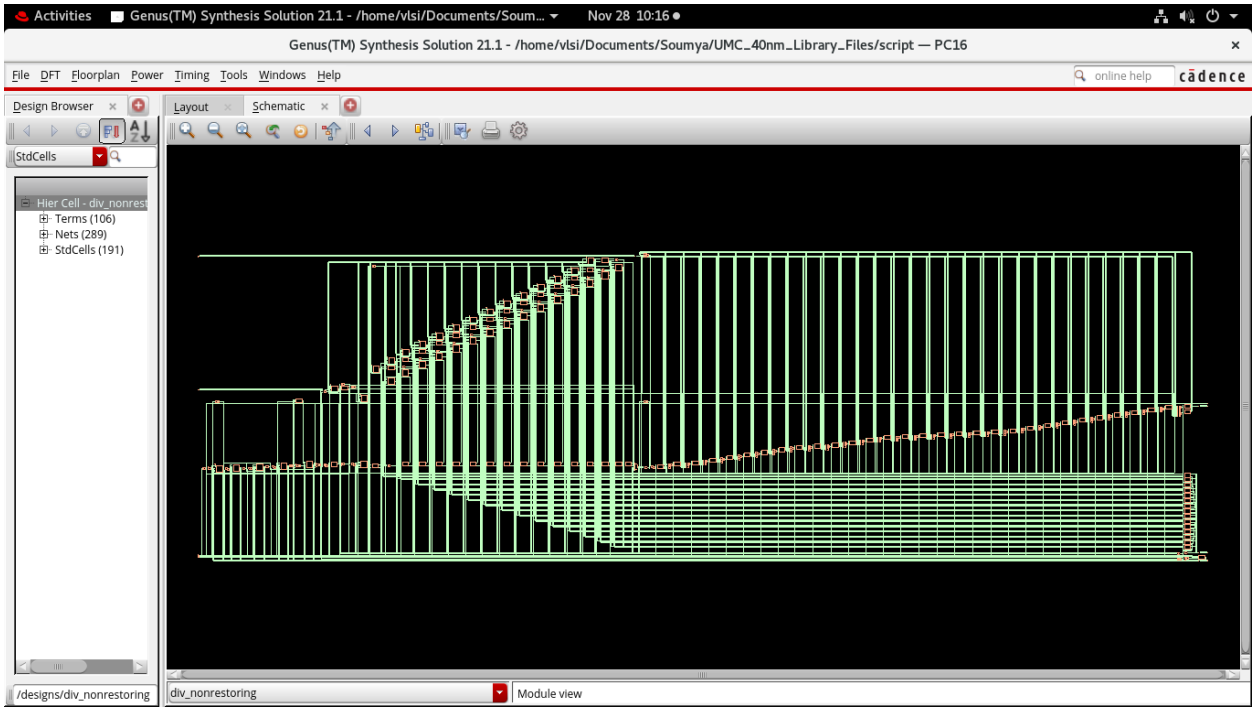


Genus

Genus Files

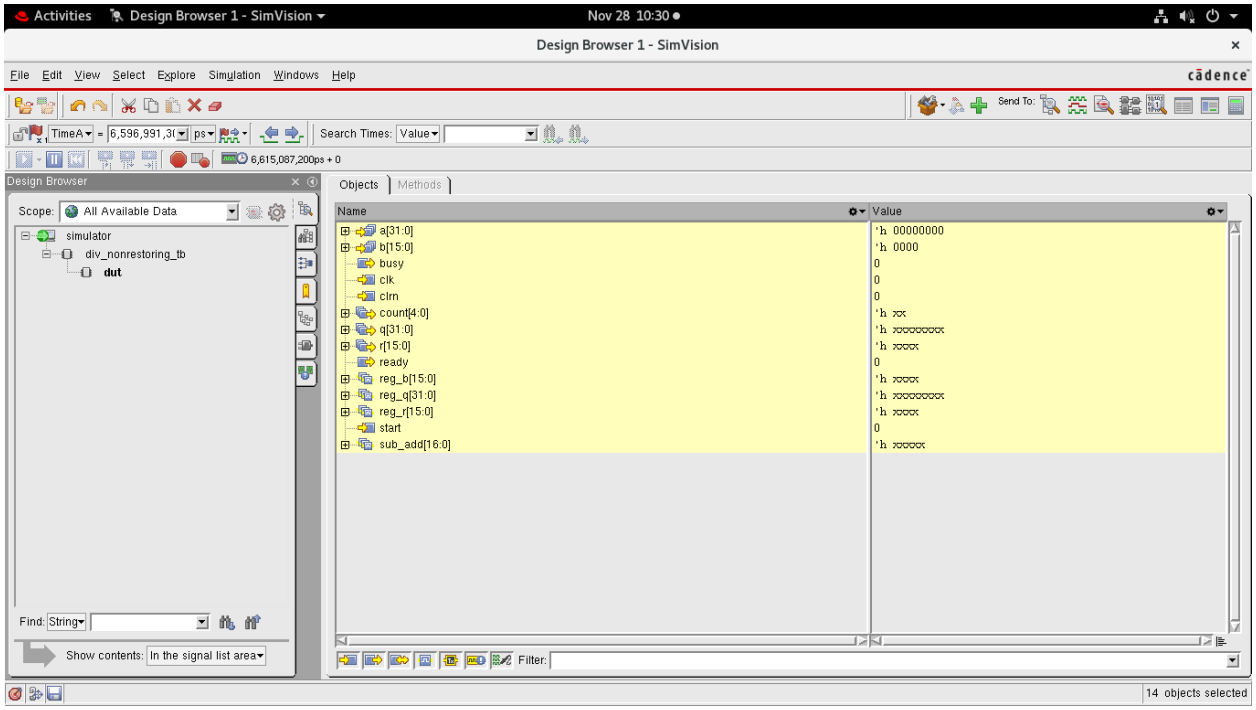




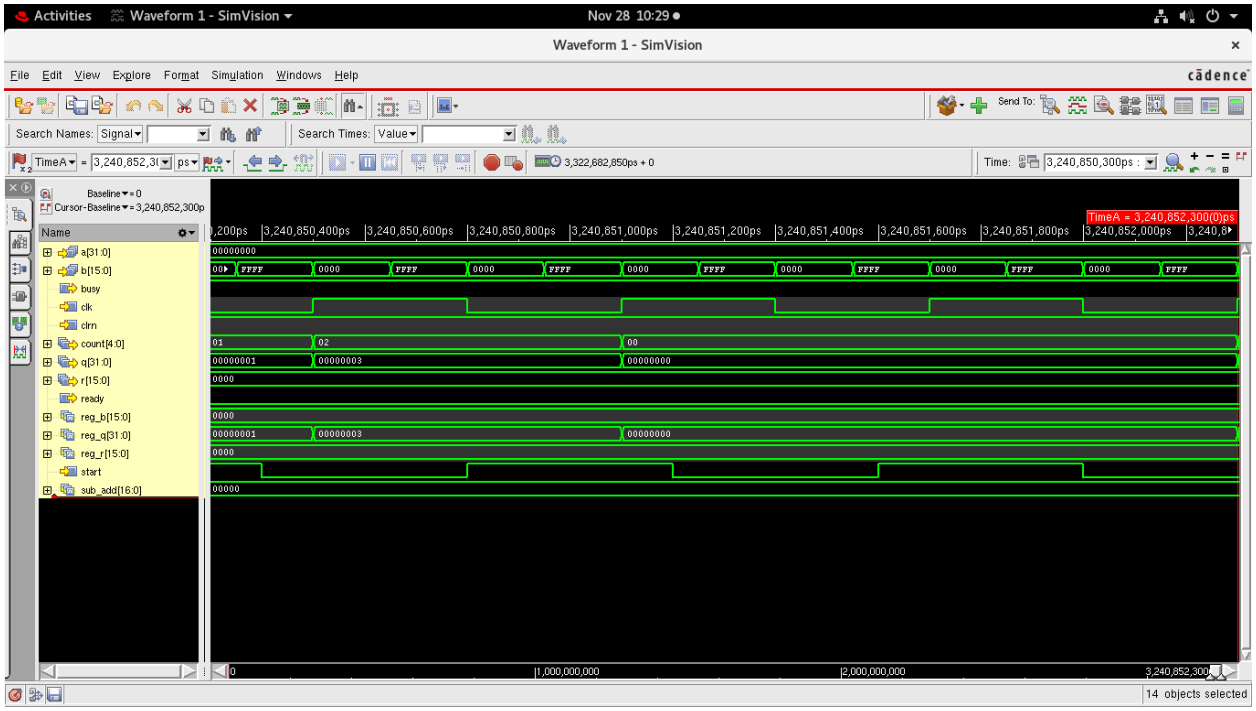


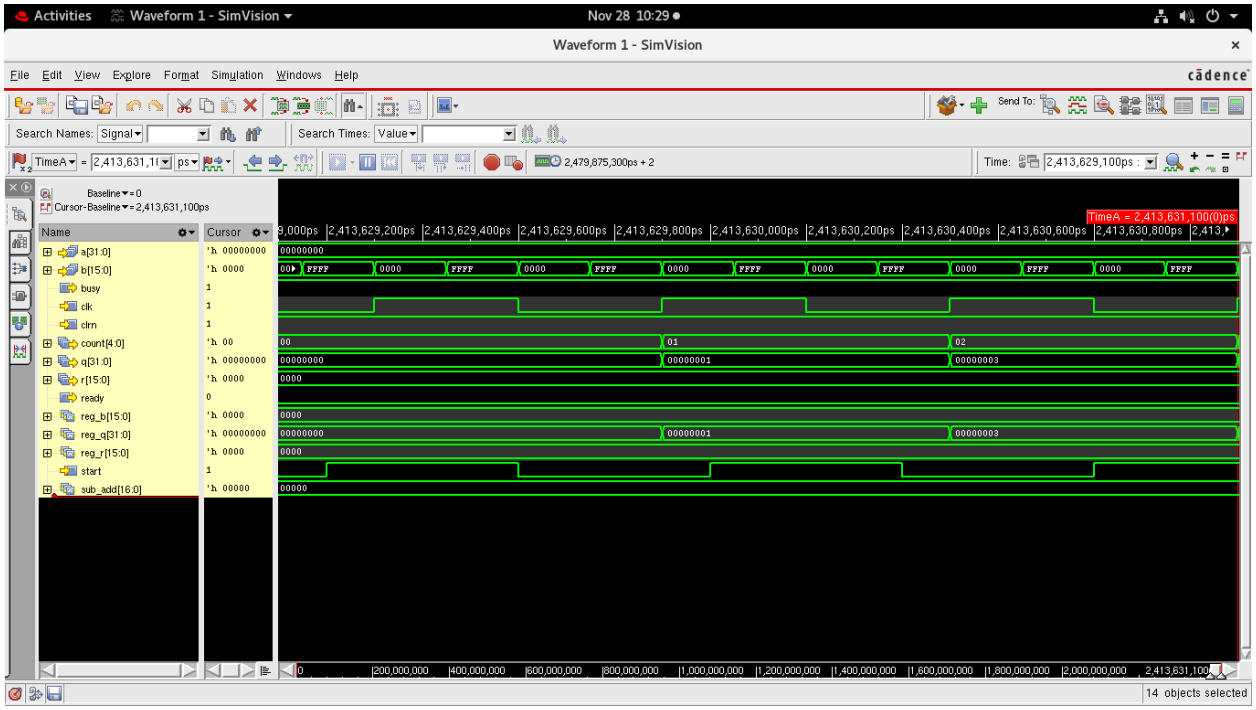
Nclaunch Verification

Design Browser

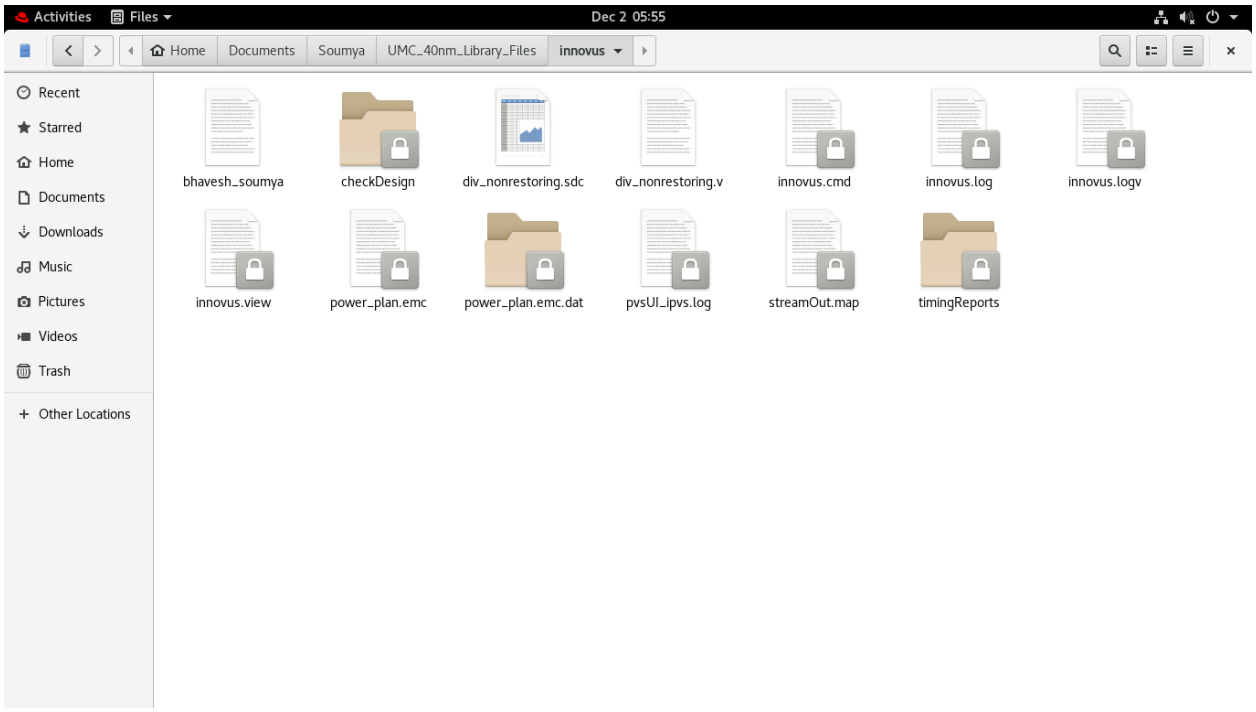

Activities
Design Browser 1 - SimVision
Nov 28 10:30
Design Browser 1 - SimVision
File Edit View Select Explore Simulation Windows Help
cadence
Search Times: Value
Design Browser
Scope: All Available Data
simulator
div_nonrestoring_tb
dut
Objects
Methods
Name
Value
a[31:0]
b[15:0]
busy
clk
clrn
count[4:0]
q[31:0]
r[15:0]
ready
reg_b[15:0]
reg_q[31:0]
reg_r[15:0]
start
sub_add[16:0]
Find: String
Show contents: In the signal list area
Filter:
14 objects selected

Waveform

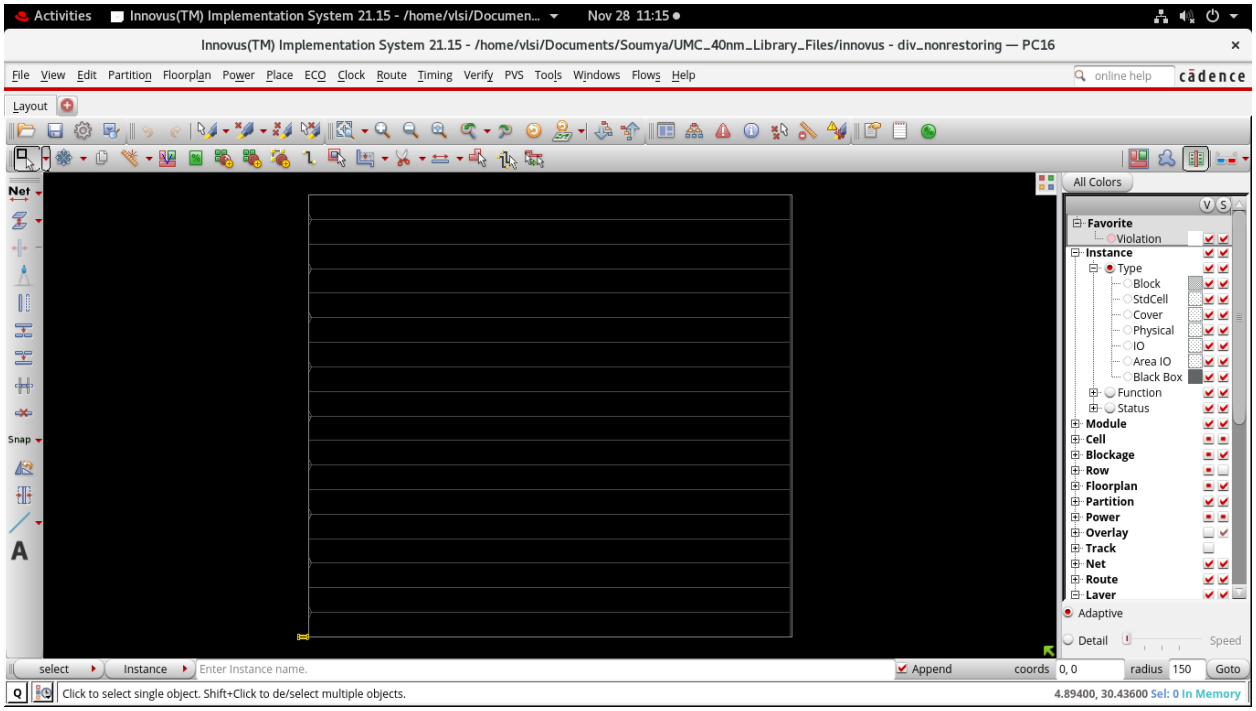

Activities
Waveform 1 - SimVision
Nov 28 10:29
Waveform 1 - SimVision
File Edit View Explore Format Simulation Windows Help
cadence
Search Names: Signal
Search Times: Value
Time: 3,240,850,300ps
Baseline = 0
Cursor-Baseline = 3,240,852,300p
TimeA = 3,240,852,300(0)ps
Name
3,240,850,400ps
3,240,850,600ps
3,240,850,800ps
3,240,851,000ps
3,240,851,200ps
3,240,851,400ps
3,240,851,600ps
3,240,851,800ps
3,240,852,000ps
a[31:0]
b[15:0]
busy
clk
clrn
count[4:0]
q[31:0]
r[15:0]
ready
reg_b[15:0]
reg_q[31:0]
reg_r[15:0]
start
sub_add[16:0]
0
1,000,000,000
2,000,000,000
14 objects selected


Activities
Waveform 1 - SimVision
Nov 28 10:29
Waveform 1 - SimVision
File Edit View Explore Format Simulation Windows Help
cadence
Search Names: Signal
Search Times: Value
Send To:
Time: 2,413,629,100ps
Name
Cursor
TimeA = 2,413,631,100(0)ps
14 objects selected

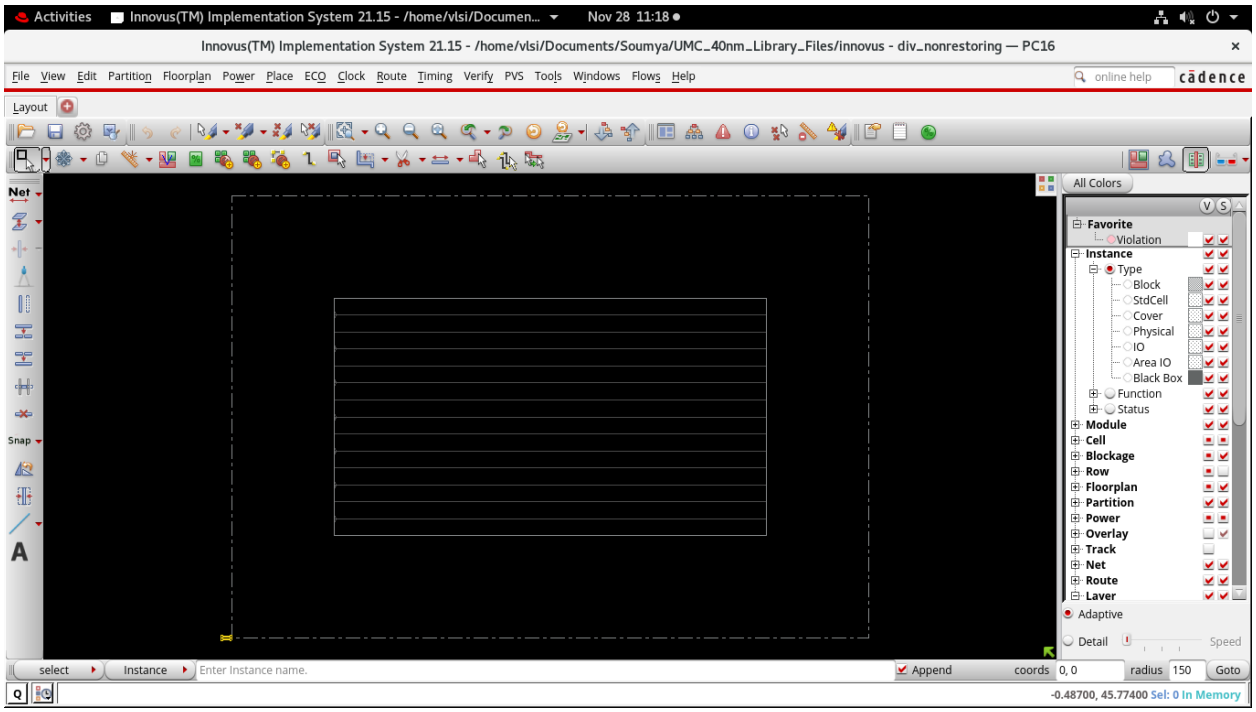
Innovus


Activities
Files
Dec 2 05:55
Home
Documents
Soumya
UMC_40nm_Library_Files
innovus
Recent
Starred
Home
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
bhavesh_soumya
checkDesign
div_nonrestoring.sdc
div_nonrestoring.v
innovus.cmd
innovus.log
innovus.logv
innovus.view
power_plan.emc
power_plan.emc.dat
pvsUI_ipvs.log
streamOut.map
timingReports

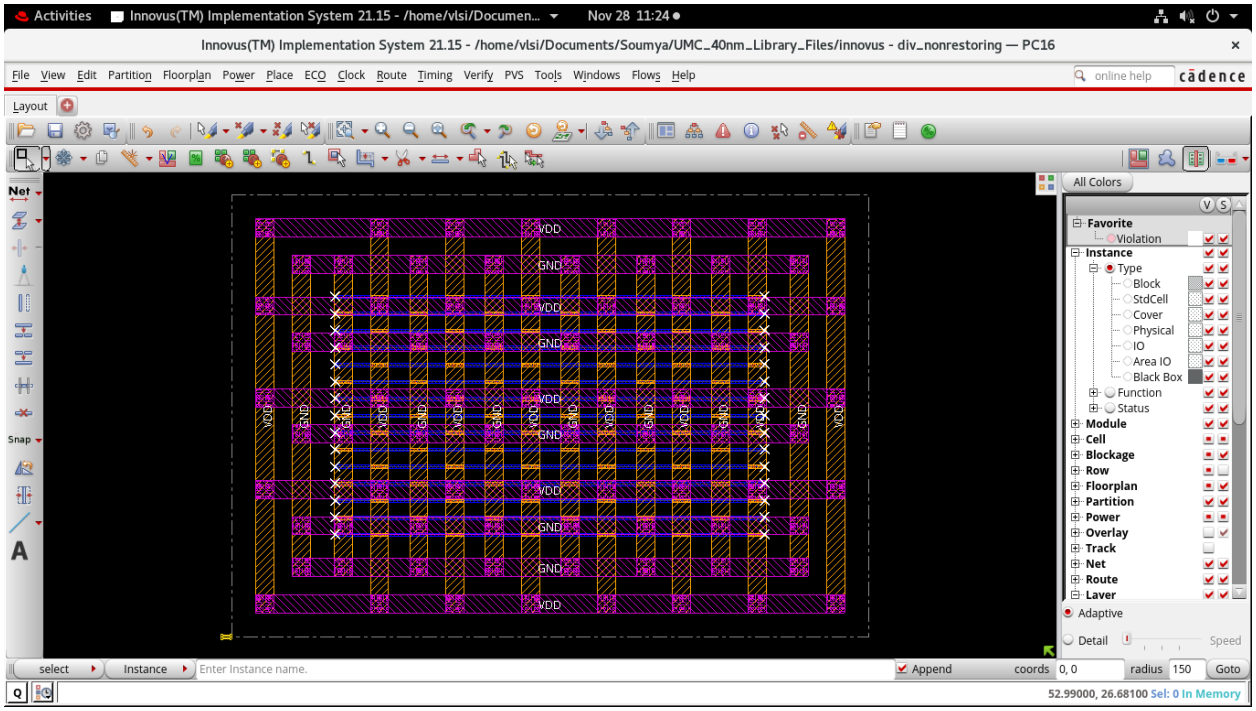
Floor Plan



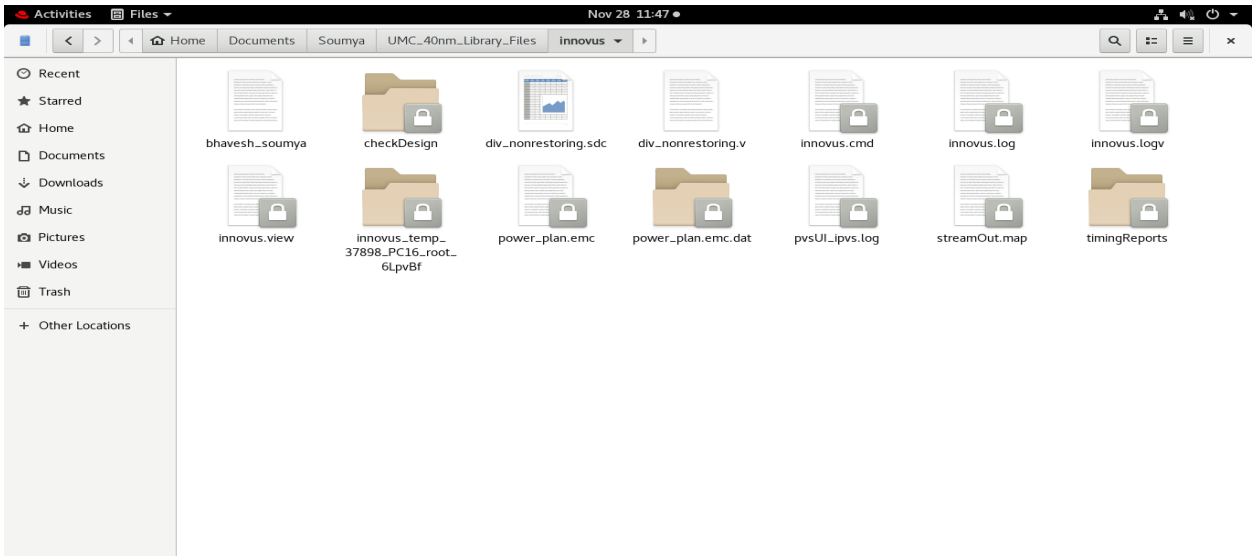
Power Planning



Routing (VDD & GND)



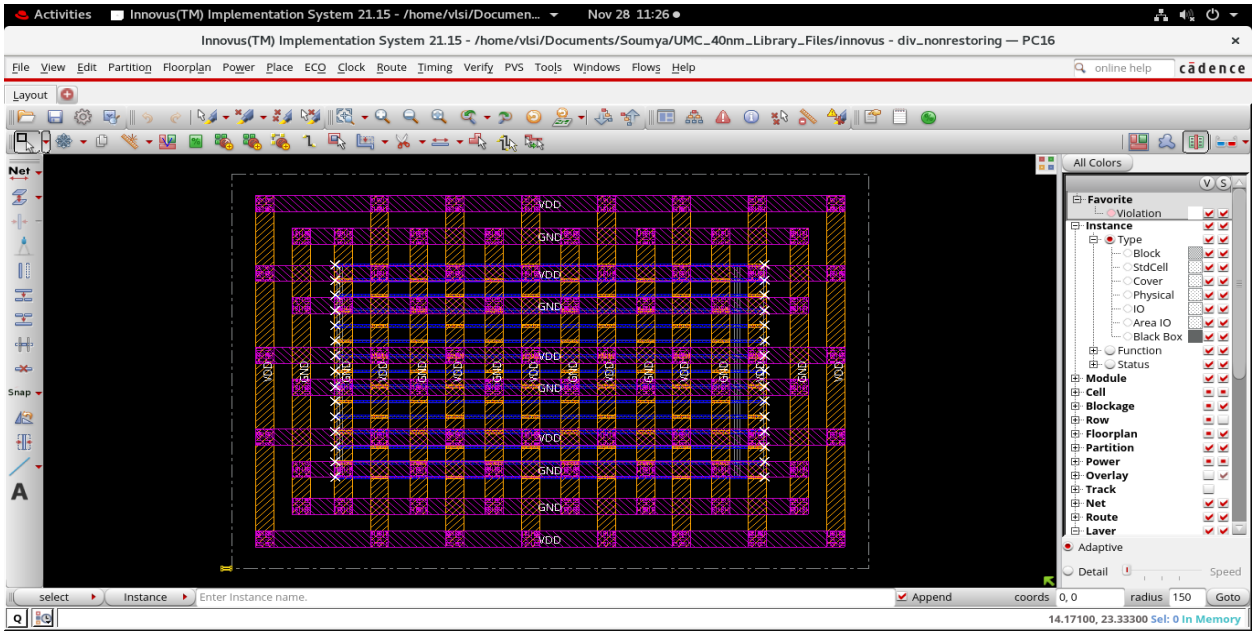
power_plan.emc Folder will created



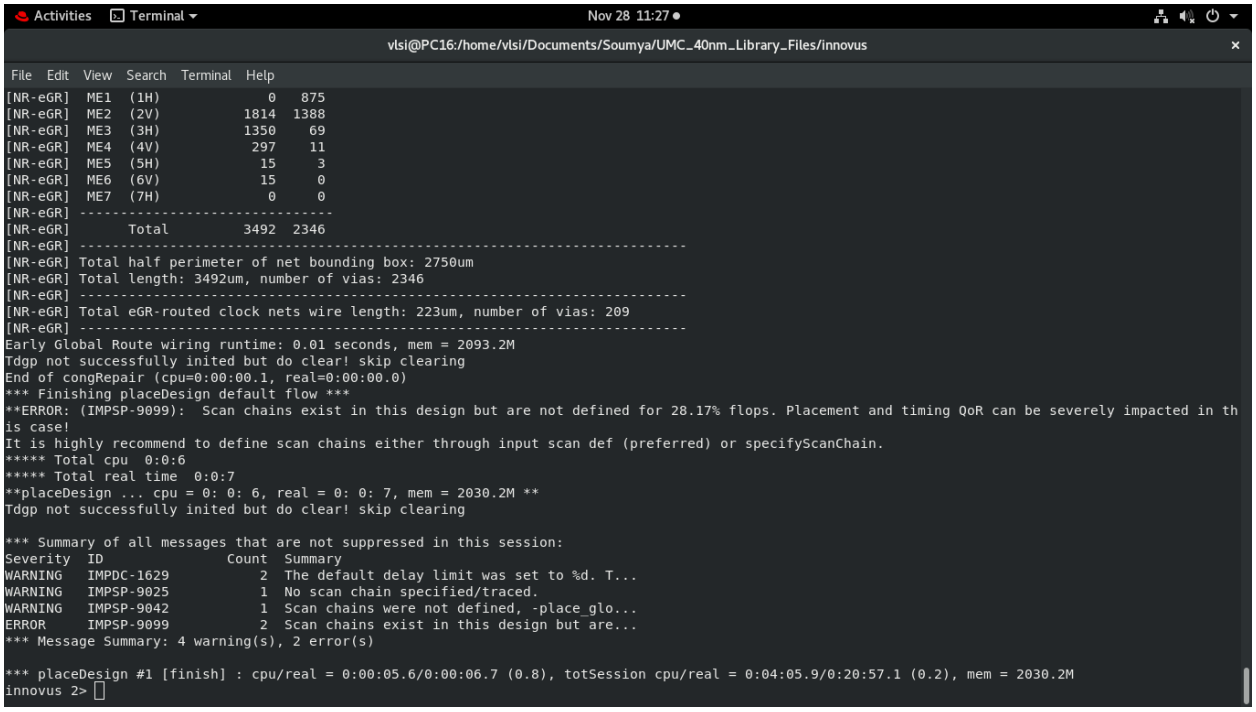
For Physical Cell

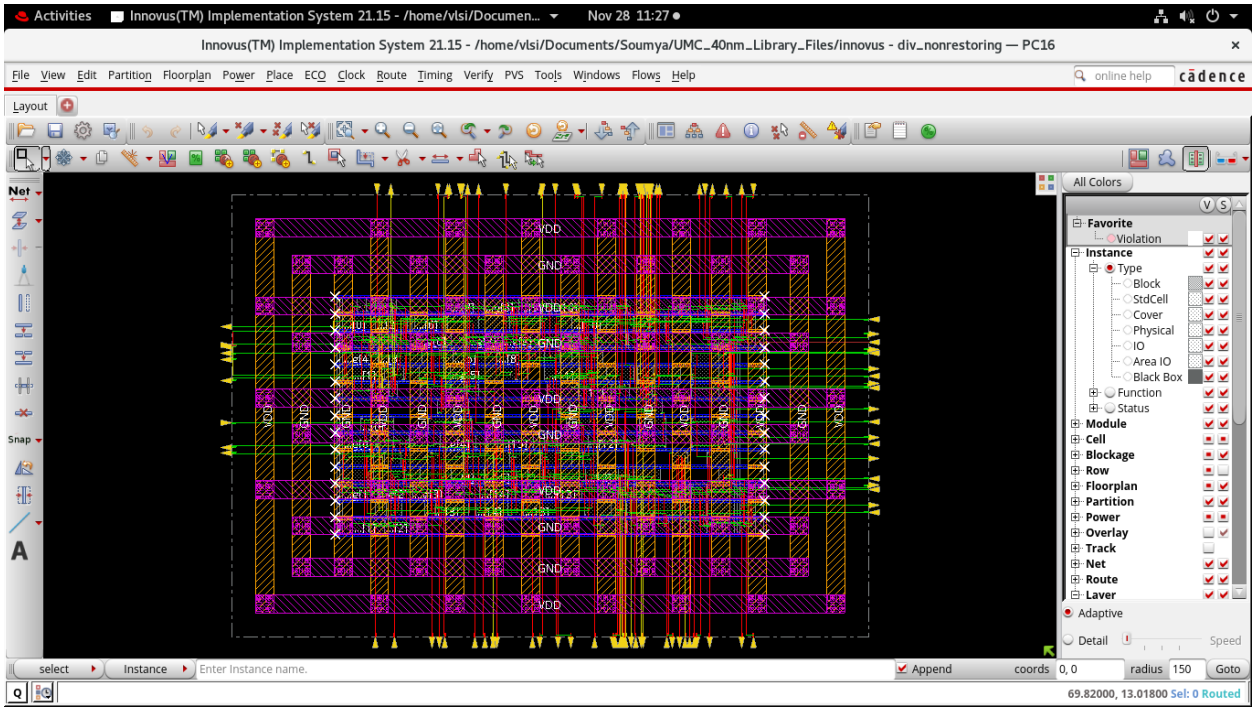
Added end cap and well tap

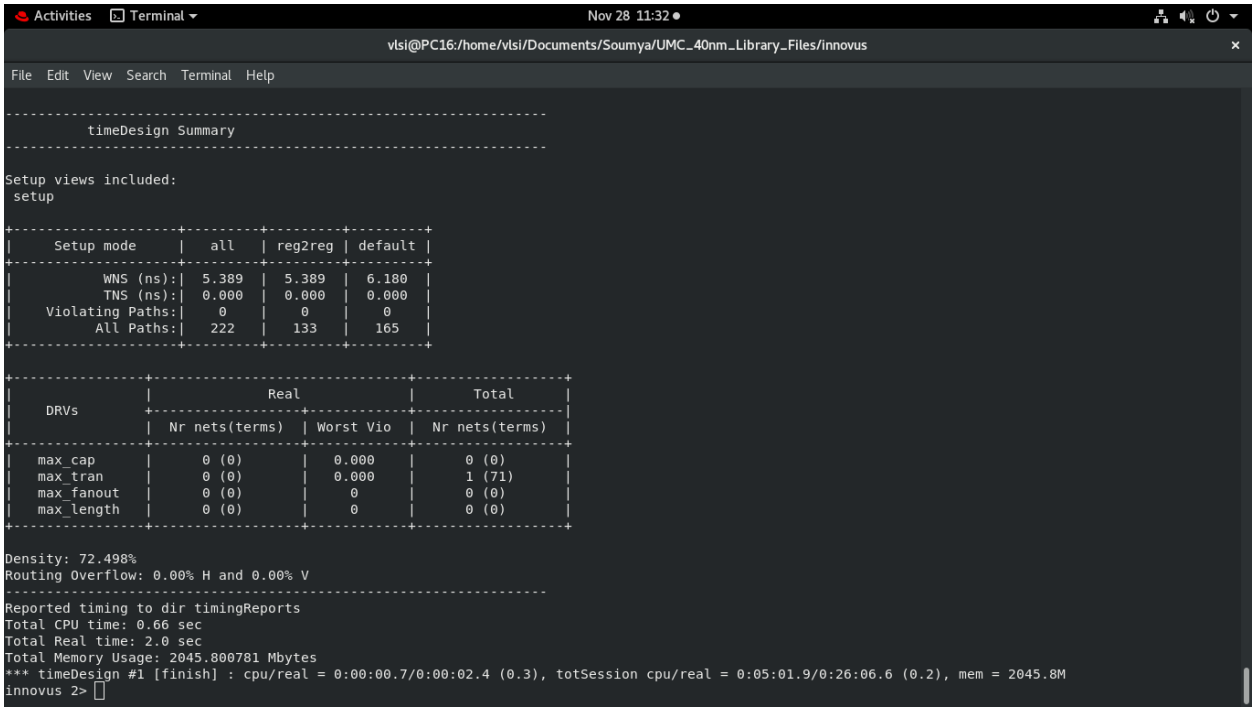
Choose FILLER D2L HHM and FILLER D2L HHM ,40



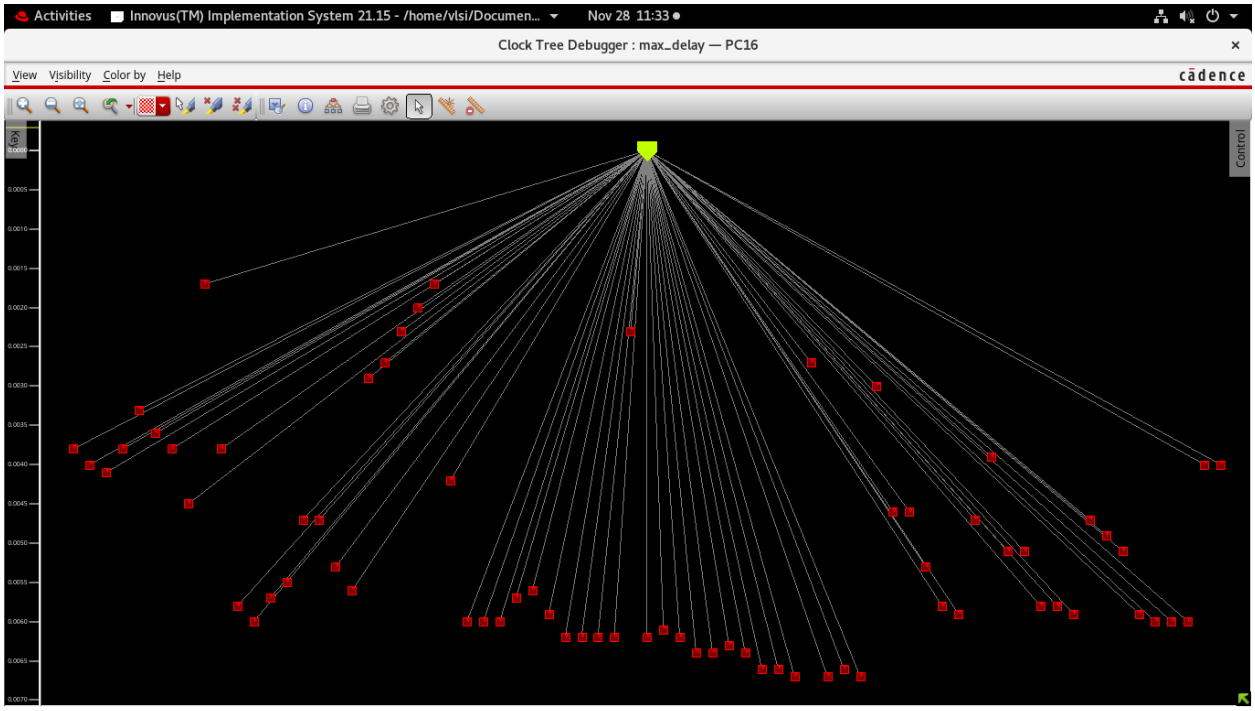
Memory Store in Terminal



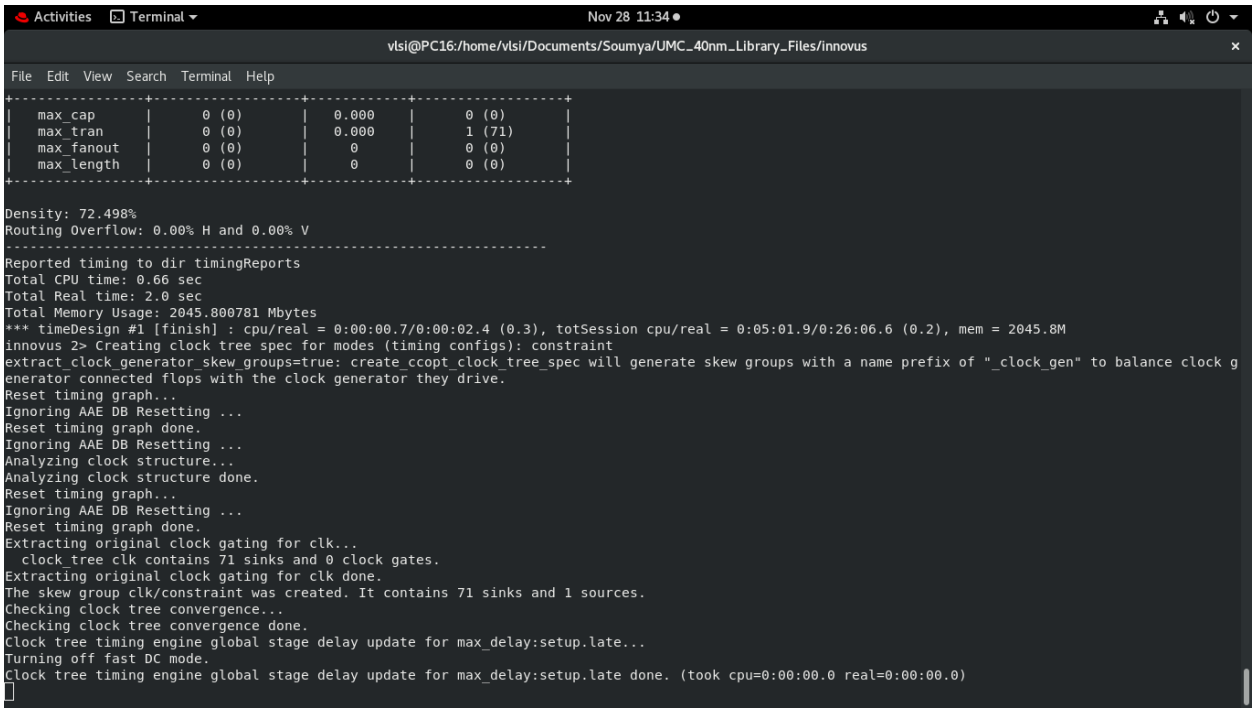

Activities
Innovus(TM) Implementation System 21.15 - /home/vlsi/Documen...
Nov 28 11:27
Innovus(TM) Implementation System 21.15 - /home/vlsi/Documents/Soumya/UMC_40nm_Library_Files/innovus - div_nonrestoring — PC16
File View Edit Partition Floorplan Power Place ECO Clock Route Timing Verify PVS Tools Windows Flows Help
online help
cadence
Layout
All Colors
Favorite
Violation
Instance
Type
Block
StdCell
Cover
Physical
IO
Area IO
Black Box
Function
Status
Module
Cell
Blockage
Row
Floorplan
Partition
Power
Overlay
Track
Net
Route
Laver
Adaptive
Detail
Speed
select
Instance
Enter Instance name.
Append
coords
0, 0
radius
150
Goto
69.82000, 13.01800 Sel: 0 Routed


Activities
Terminal
Nov 28 11:32
vlsi@PC16:/home/vlsi/Documents/Soumya/UMC_40nm_Library_Files/innovus
File Edit View Search Terminal Help
timeDesign Summary
Setup views included:
setup
Setup mode	all	reg2reg	default
WNS (ns):	5.389	5.389	6.180
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	222	133	165
DRVs	Real		Total
	Nr nets(terms)	Worst Vio	Nr nets(terms)
max_cap	0 (0)	0.000	0 (0)
max_tran	0 (0)	0.000	1 (71)
max_fanout	0 (0)	0	0 (0)
max_length	0 (0)	0	0 (0)
Density: 72.498%
Routing Overflow: 0.00% H and 0.00% V
Reported timing to dir timingReports
Total CPU time: 0.66 sec
Total Real time: 2.0 sec
Total Memory Usage: 2045.800781 Mbytes
*** timeDesign #1 [finish] : cpu/real = 0:00:00.7/0:00:02.4 (0.3), totSession cpu/real = 0:05:01.9/0:26:06.6 (0.2), mem = 2045.8M
innovus 2>

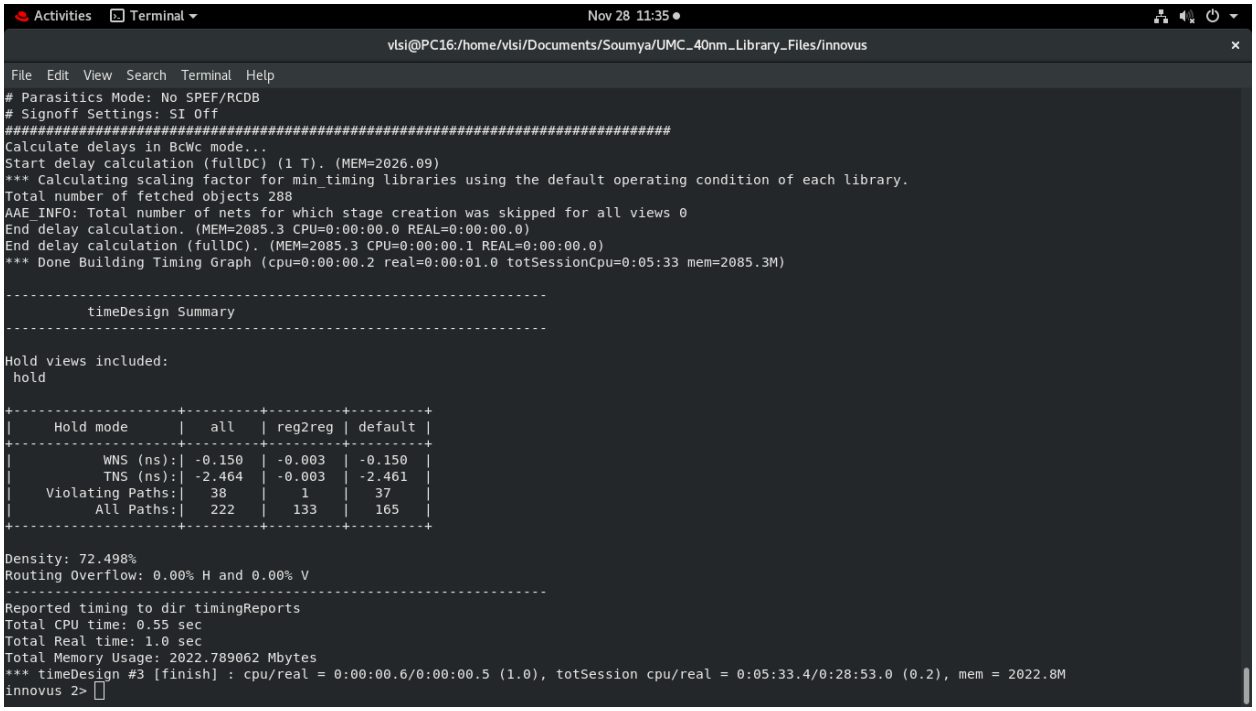
Clock Tree



Clock Tree Timing Report Terminal

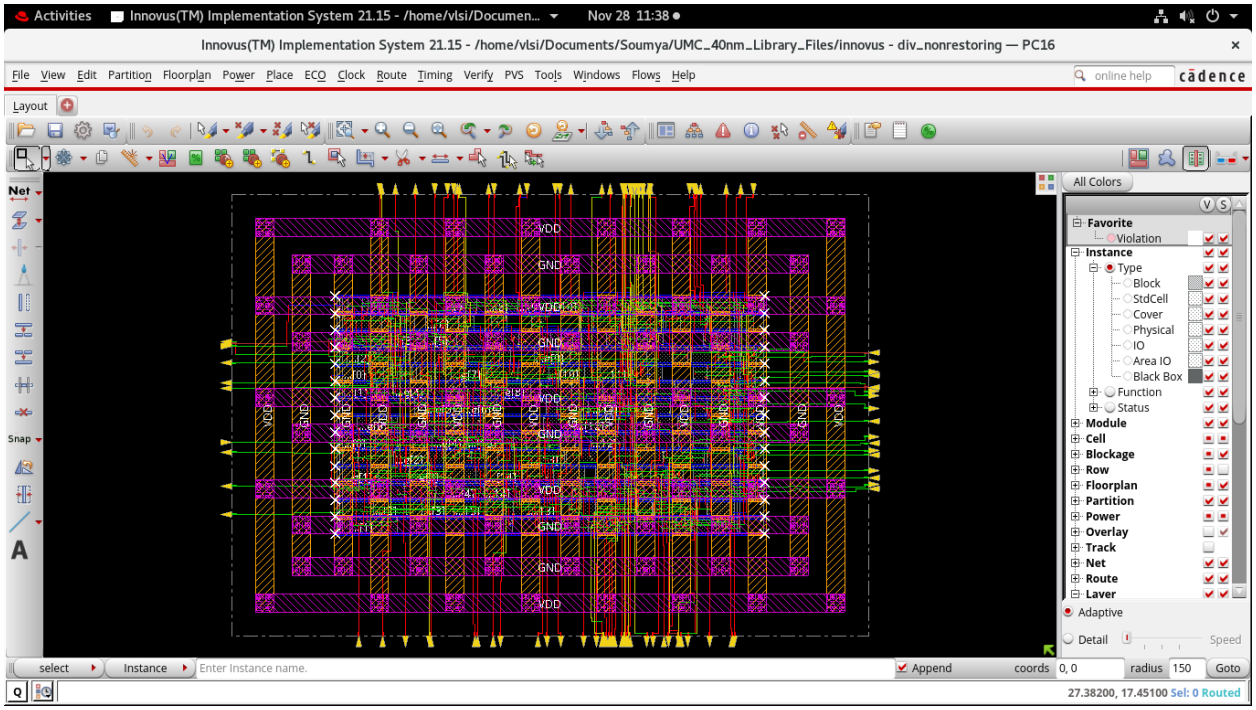


Timing Design Hold

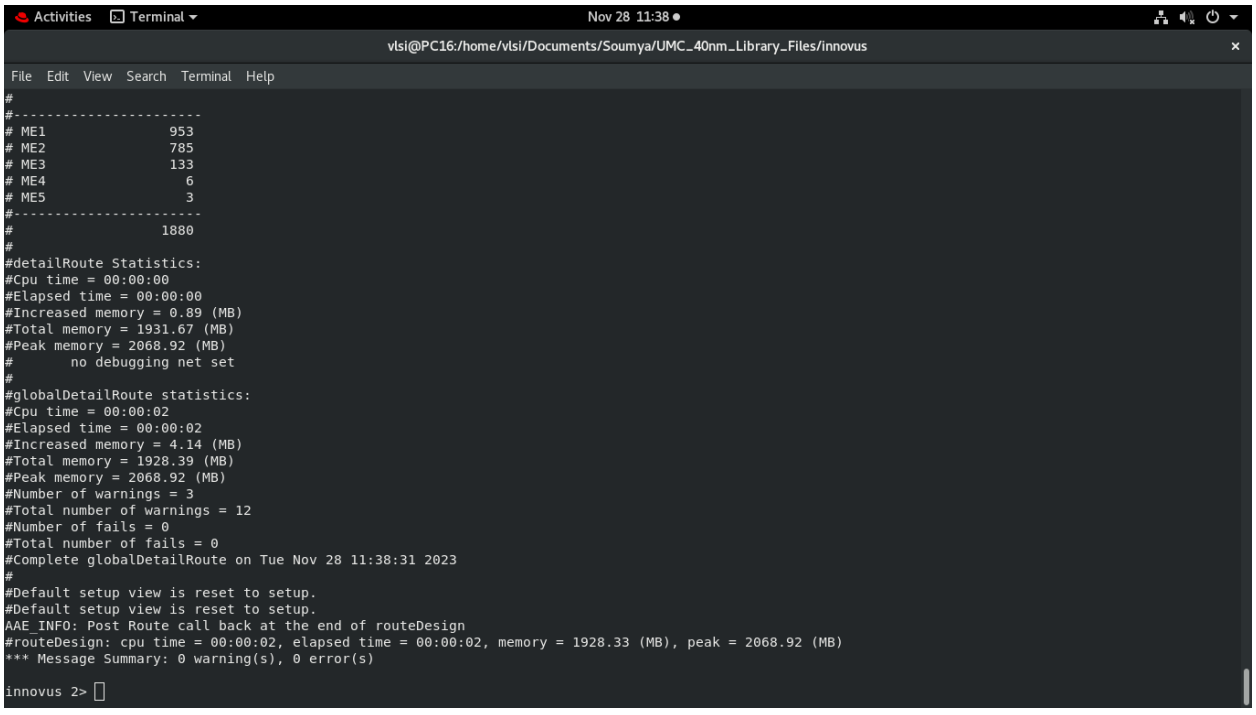


```
Activities  Terminal  Nov 28 11:35
vlsi@PC16:/home/vlsi/Documents/Soumya/UMC_40nm_Library_Files/innovus
File  Edit  View  Search  Terminal  Help
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
################################################################################
Calculate delays in BcWc mode...
Start delay calculation (fullDC) (1 T). (MEM=2026.09)
*** Calculating scaling factor for min_timing libraries using the default operating condition of each library.
Total number of fetched objects 288
AAE_INFO: Total number of nets for which stage creation was skipped for all views 0
End delay calculation. (MEM=2085.3 CPU=0:00:00.0 REAL=0:00:00.0)
End delay calculation (fullDC). (MEM=2085.3 CPU=0:00:00.1 REAL=0:00:00.0)
*** Done Building Timing Graph (cpu=0:00:00.2 real=0:00:01.0 totSessionCpu=0:05:33 mem=2085.3M)

------------------------------------------------------------------
         timeDesign Summary
------------------------------------------------------------------

Hold views included:
 hold

+--------------------+---------+---------+---------+
|     Hold mode      |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):| -0.150  | -0.003  | -0.150  |
|           TNS (ns):| -2.464  | -0.003  | -2.461  |
|    Violating Paths:|   38    |    1    |   37    |
|          All Paths:|   222   |   133   |   165   |
+--------------------+---------+---------+---------+

Density: 72.498%
Routing Overflow: 0.00% H and 0.00% V
------------------------------------------------------------------
Reported timing to dir timingReports
Total CPU time: 0.55 sec
Total Real time: 1.0 sec
Total Memory Usage: 2022.789062 Mbytes
*** timeDesign #3 [finish] : cpu/real = 0:00:00.6/0:00:00.5 (1.0), totSession cpu/real = 0:05:33.4/0:28:53.0 (0.2), mem = 2022.8M
innovus 2>
```

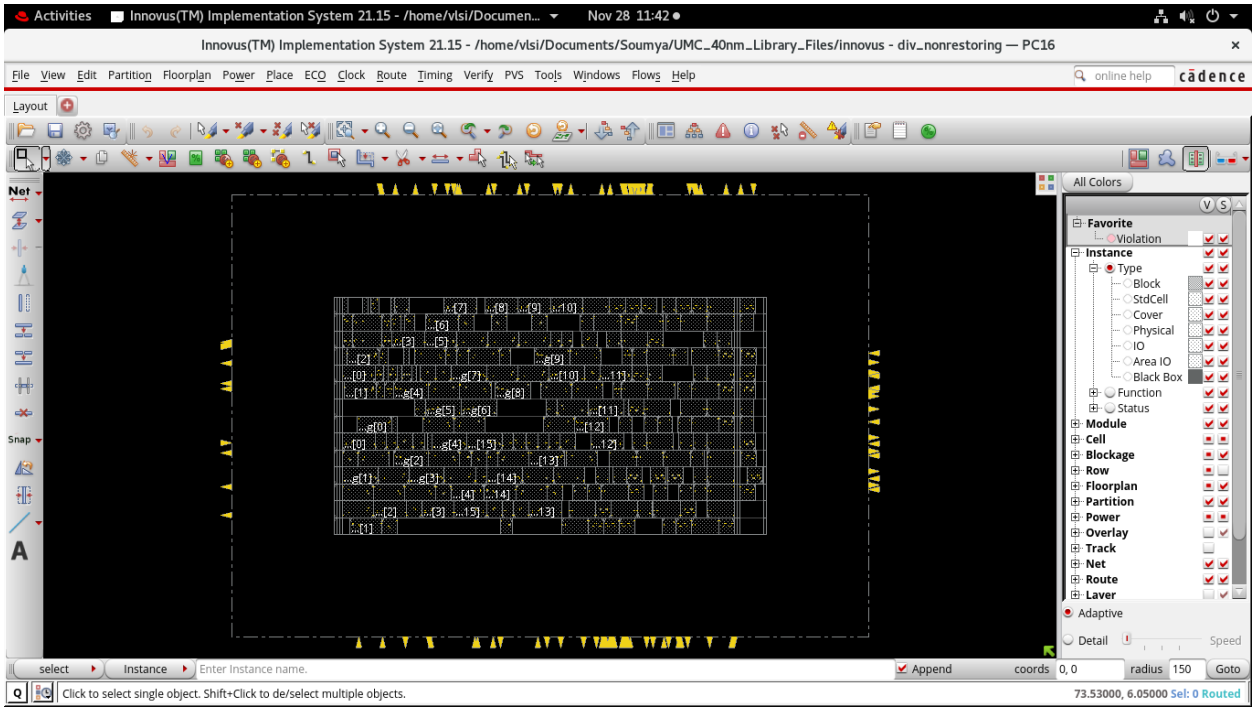
```
Activities  Terminal  Nov 28 11:37
vlsi@PC16:/home/vlsi/Documents/Soumya/UMC_40nm_Library_Files/innovus
File  Edit  View  Search  Terminal  Help
+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  5.397  |  5.397  |  6.180  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|   222   |   133   |   165   |
+--------------------+---------+---------+---------+

+--------------------+---------+---------+---------+
|     Hold mode      |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.004  |  0.007  |  0.004  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|   222   |   133   |   165   |
+--------------------+---------+---------+---------+

+----------------+-------------------------------+------------------+
|                |             Real              |       Total      |
|    DRVs        +------------------+------------+------------------|
|                |  Nr nets(terms)  | Worst Vio  |  Nr nets(terms)  |
+----------------+------------------+------------+------------------+
|   max_cap      |      0 (0)       |   0.000    |      0 (0)       |
|   max_tran     |      0 (0)       |   0.000    |      1 (71)      |
|   max_fanout   |      0 (0)       |     0      |      0 (0)       |
|   max_length   |      0 (0)       |     0      |      0 (0)       |
+----------------+------------------+------------+------------------+

Density: 80.904%
Routing Overflow: 0.00% H and 0.00% V
------------------------------------------------------------------
**optDesign ... cpu = 0:00:10, real = 0:00:12, mem = 2000.9M, totSessionCpu=0:06:09 **
*** Finished optDesign ***
Info: Destroy the CCOpt slew target map.
*** optDesign #2 [finish] : cpu/real = 0:00:10.4/0:00:12.0 (0.9), totSession cpu/real = 0:06:09.3/0:30:55.6 (0.2), mem = 2137.5M
innovus 2>
```

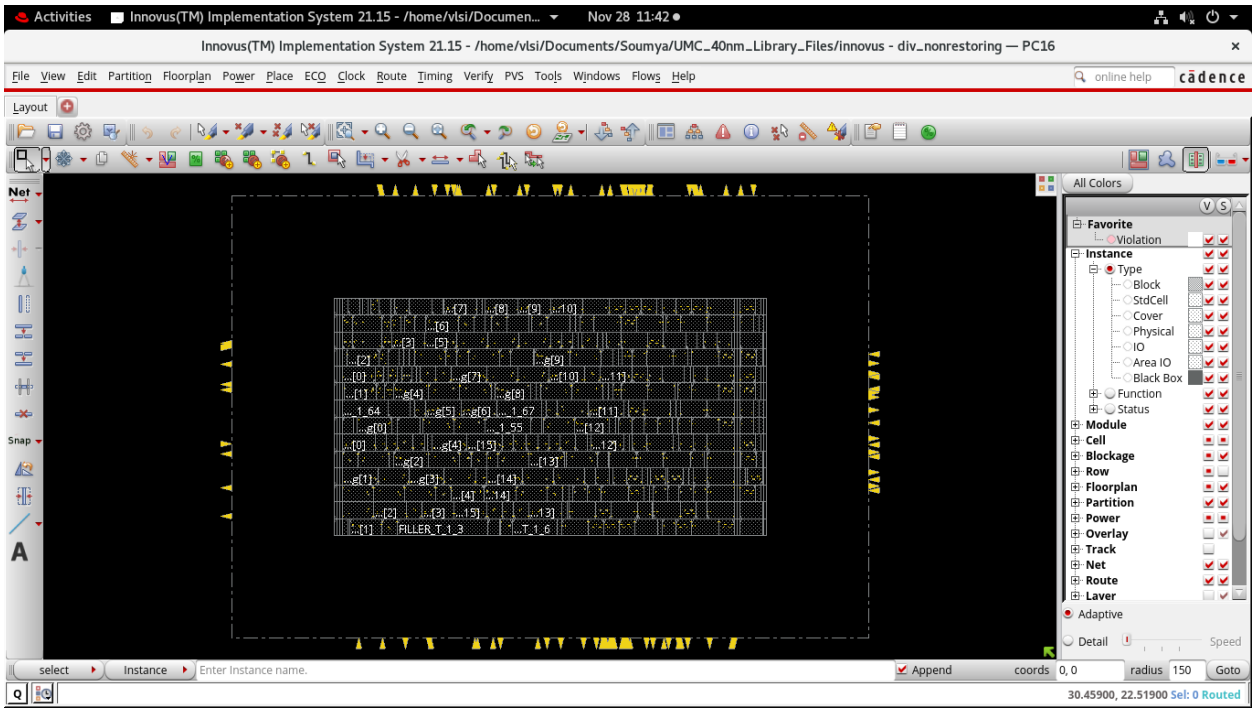


Route Statistics



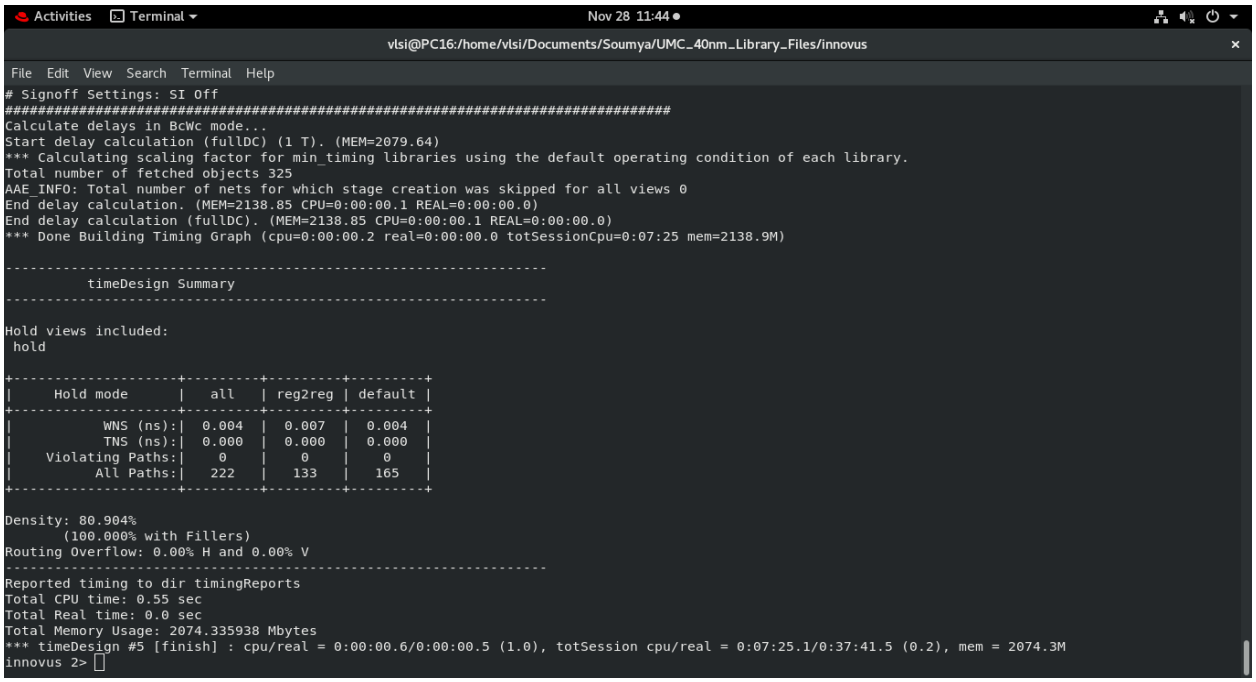
Without Layer



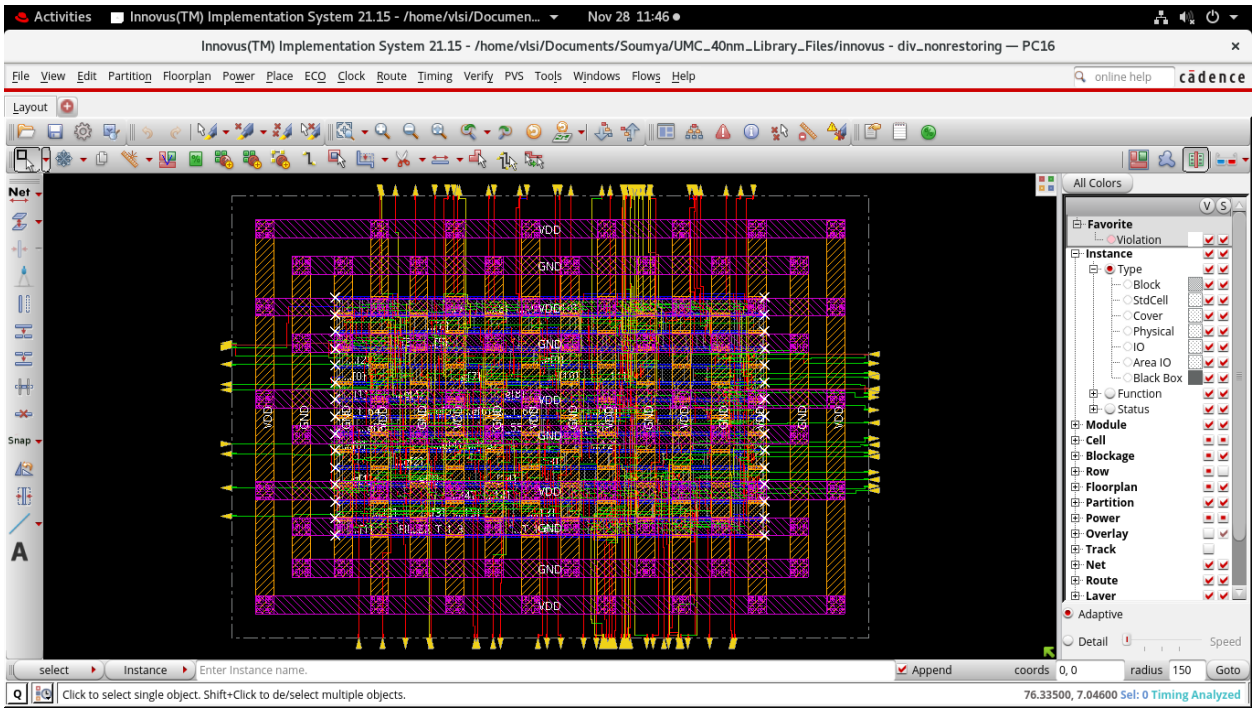


Report Timing

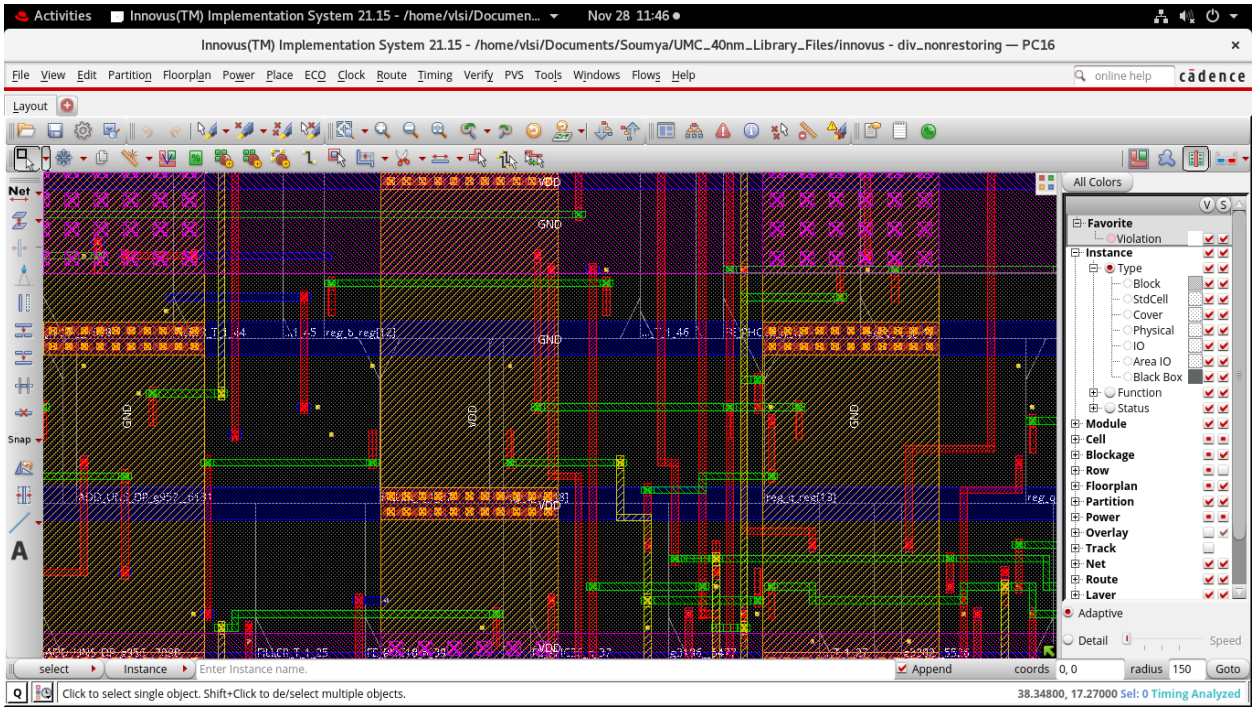
Total CPU Time, Total Real Time, Total Memory Usage



Final Custom Design



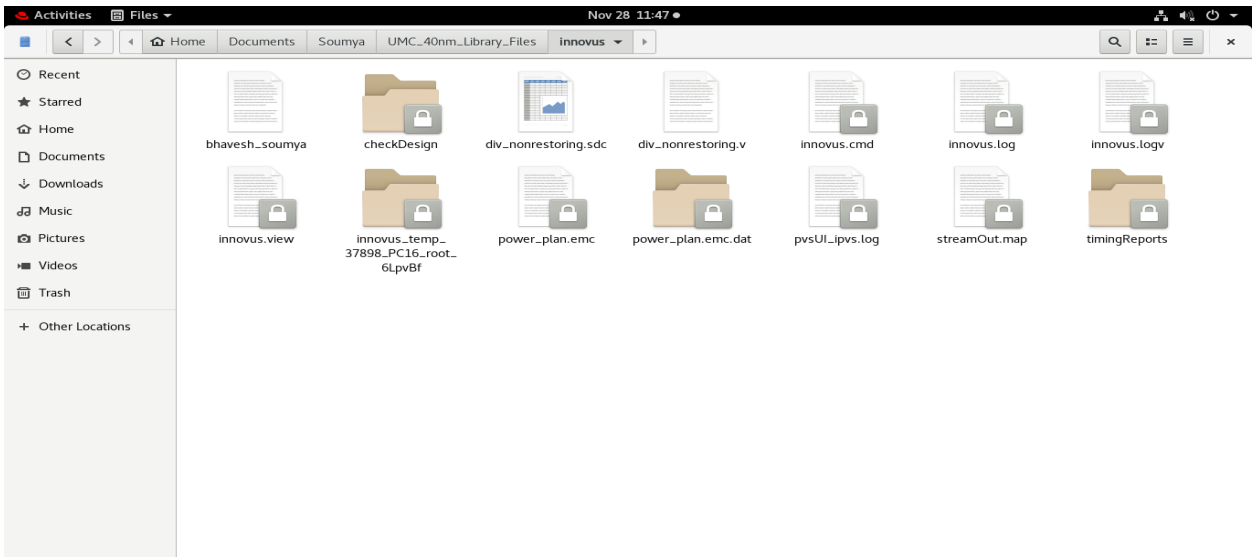
Zoomed Custom Design



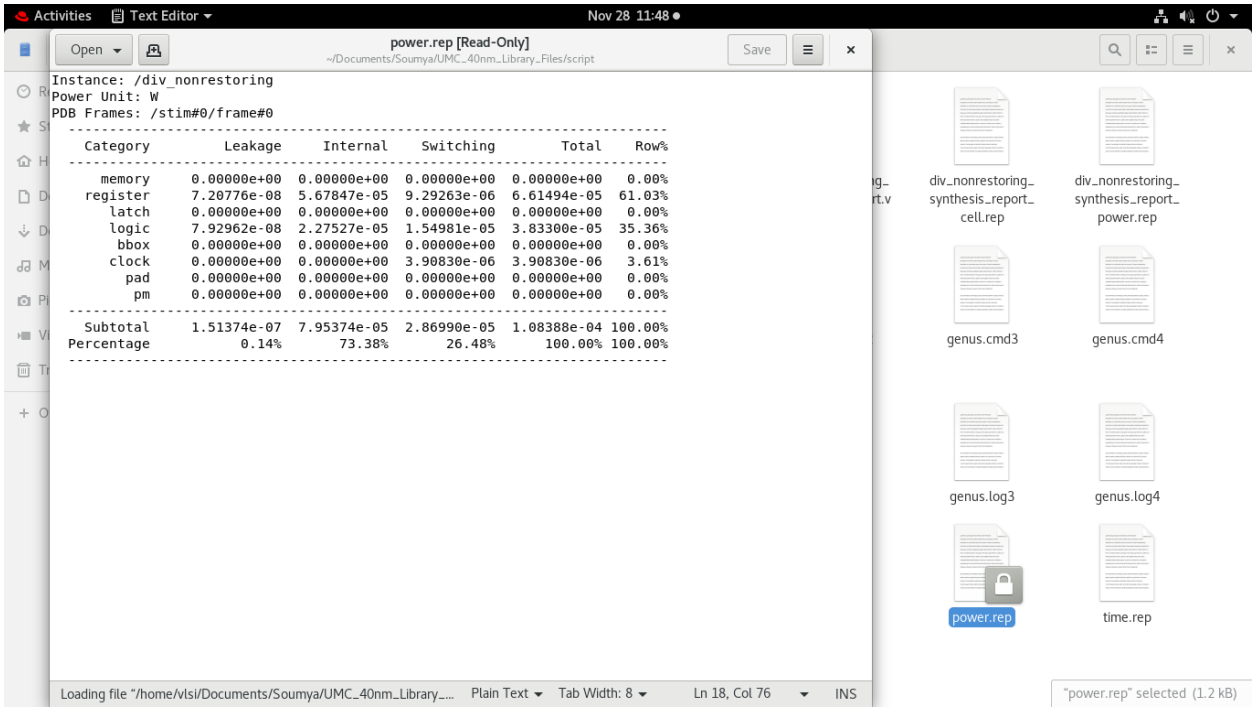
GDS File

Created at Innovus file after the semi custom design of Non Restoring Division Algorithm

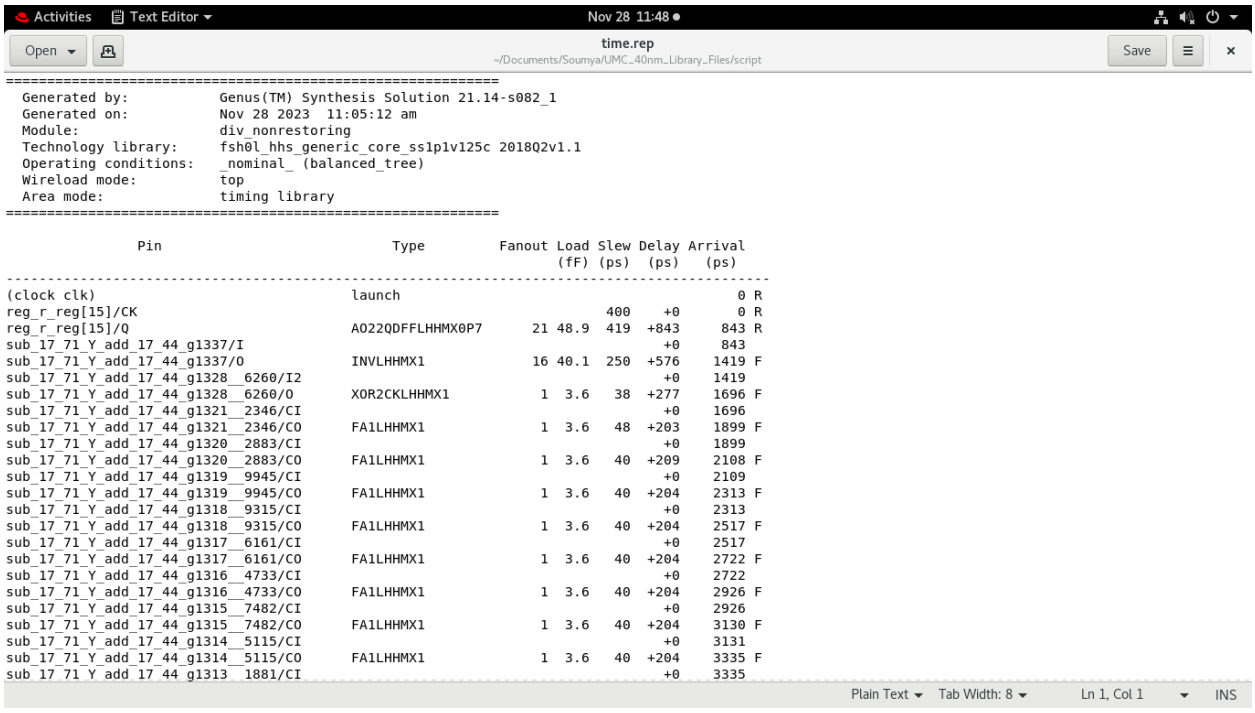
File Name : bhavesh_soumya.gds

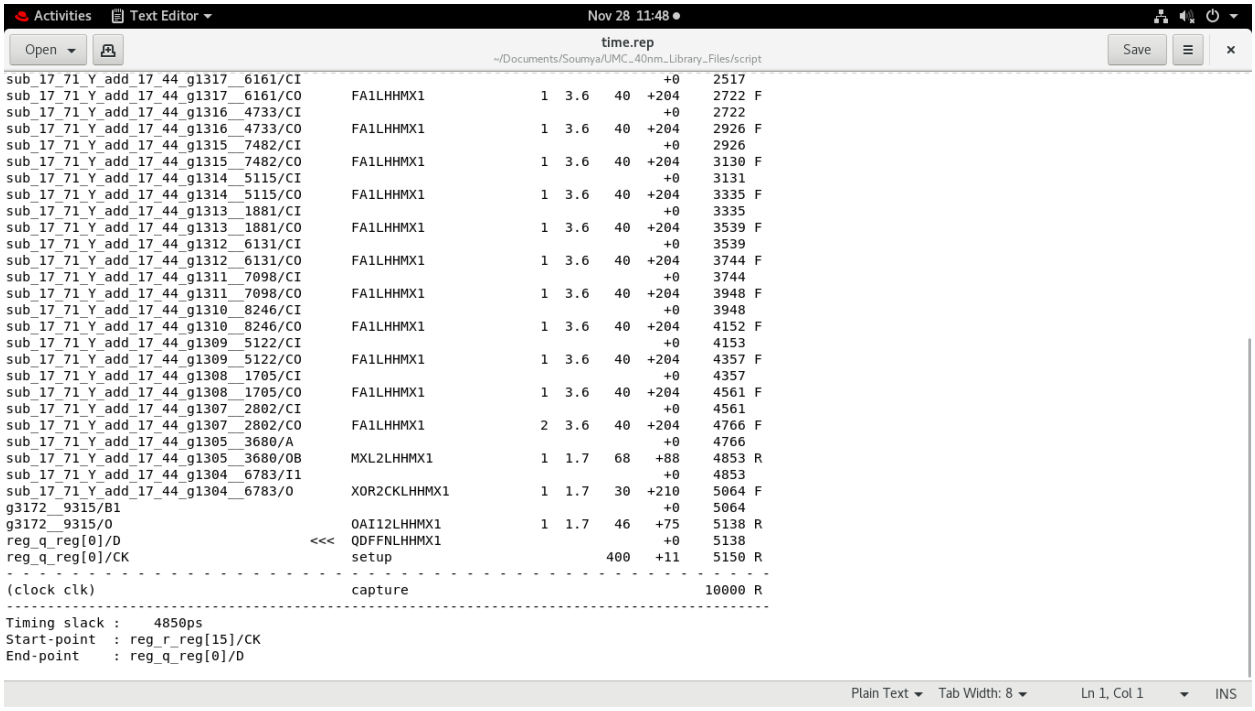


Power Report



Time Report




Activities
Text Editor
Nov 28 11:48
Open
time.rep
~/Documents/Soumya/UMC_40nm_Library_Files/script
Save
sub_17_71_Y_add_17_44_g1317__6161/CI +0 2517
sub_17_71_Y_add_17_44_g1317__6161/CO FA1LHHMX1 1 3.6 40 +204 2722 F
sub_17_71_Y_add_17_44_g1316__4733/CI +0 2722
sub_17_71_Y_add_17_44_g1316__4733/CO FA1LHHMX1 1 3.6 40 +204 2926 F
sub_17_71_Y_add_17_44_g1315__7482/CI +0 2926
sub_17_71_Y_add_17_44_g1315__7482/CO FA1LHHMX1 1 3.6 40 +204 3130 F
sub_17_71_Y_add_17_44_g1314__5115/CI +0 3131
sub_17_71_Y_add_17_44_g1314__5115/CO FA1LHHMX1 1 3.6 40 +204 3335 F
sub_17_71_Y_add_17_44_g1313__1881/CI +0 3335
sub_17_71_Y_add_17_44_g1313__1881/CO FA1LHHMX1 1 3.6 40 +204 3539 F
sub_17_71_Y_add_17_44_g1312__6131/CI +0 3539
sub_17_71_Y_add_17_44_g1312__6131/CO FA1LHHMX1 1 3.6 40 +204 3744 F
sub_17_71_Y_add_17_44_g1311__7098/CI +0 3744
sub_17_71_Y_add_17_44_g1311__7098/CO FA1LHHMX1 1 3.6 40 +204 3948 F
sub_17_71_Y_add_17_44_g1310__8246/CI +0 3948
sub_17_71_Y_add_17_44_g1310__8246/CO FA1LHHMX1 1 3.6 40 +204 4152 F
sub_17_71_Y_add_17_44_g1309__5122/CI +0 4153
sub_17_71_Y_add_17_44_g1309__5122/CO FA1LHHMX1 1 3.6 40 +204 4357 F
sub_17_71_Y_add_17_44_g1308__1705/CI +0 4357
sub_17_71_Y_add_17_44_g1308__1705/CO FA1LHHMX1 1 3.6 40 +204 4561 F
sub_17_71_Y_add_17_44_g1307__2802/CI +0 4561
sub_17_71_Y_add_17_44_g1307__2802/CO FA1LHHMX1 2 3.6 40 +204 4766 F
sub_17_71_Y_add_17_44_g1305__3680/A +0 4766
sub_17_71_Y_add_17_44_g1305__3680/OB MXL2LHHMX1 1 1.7 68 +88 4853 R
sub_17_71_Y_add_17_44_g1304__6783/I1 +0 4853
sub_17_71_Y_add_17_44_g1304__6783/O XOR2CKLHHMX1 1 1.7 30 +210 5064 F
g3172__9315/B1 +0 5064
g3172__9315/O OAI12LHHMX1 1 1.7 46 +75 5138 R
reg_q_reg[0]/D <<< QDFFNLHHMX1 +0 5138
reg_q_reg[0]/CK setup 400 +11 5150 R
(clock clk) capture 10000 R
Timing slack : 4850ps
Start-point : reg_r_reg[15]/CK
End-point : reg_q_reg[0]/D
Plain Text
Tab Width: 8
Ln 1, Col 1
INS