# ASIC Design Flow Manual
# Using Cadence Tool Suite

**Tools used for ASIC Flow:**

1. **INCISIVE** - Used for Functional Simulation
2. **GENUS** - Used for Synthesis and pre-Layout Timing Analysis
3. **INNOVUS** - Used for Physical Design

**Getting Started :**

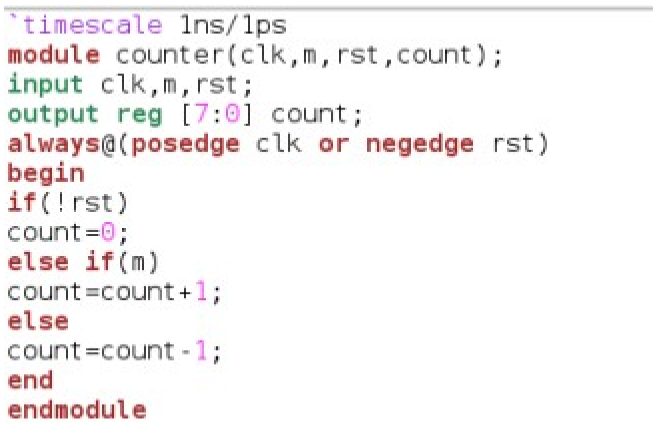
1. Make sure the Licensing Server is switched ON and the client is connected to server."
2. Open the "**counter**" directory and make a right click to "**Open in Terminal**"**.**
3. To open the tools to be used, type in the command "**csh**" (Press Enter) followed by "**source /home/install/cshrc**" <Or the path of tools whichever is applicable>.
4. A welcome string "**Welcome to Cadence Tool Suite**" appears indicating terminal ready to invoke Cadence Tools available for you.

**Module 1:** Creating an RTL Code

In order to create an RTL Code, you can open a text editor and type in your Verilog code or VHDL Code.

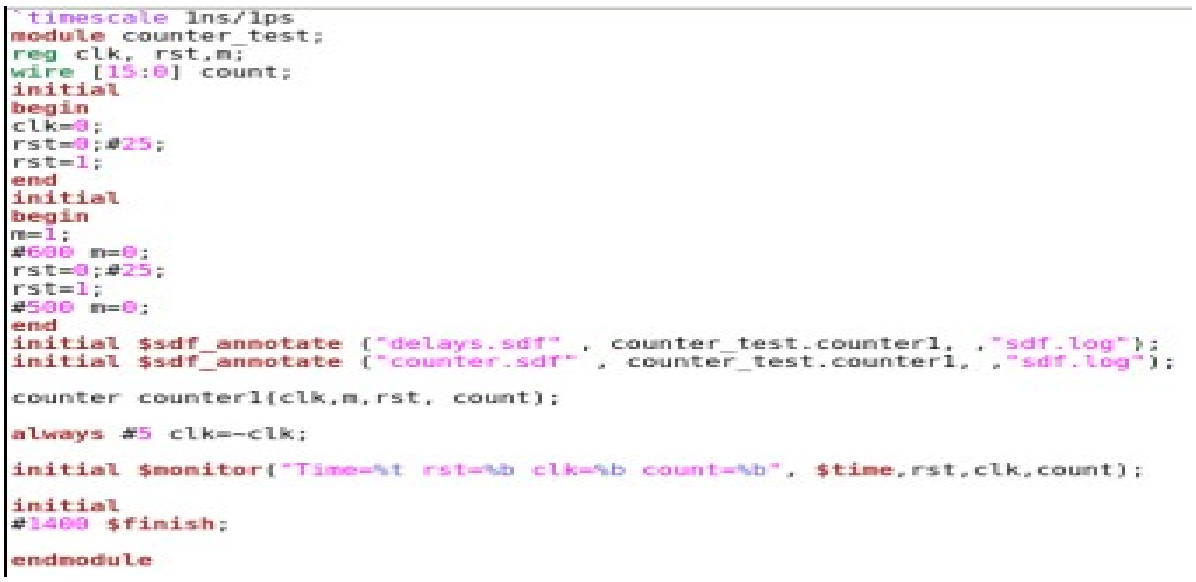
1. In the terminal, type in "**gedit <filename>.v [OR] <filename>.vhdl**"**.** The file extensions depends on the type of RTL Code you write as shown.

2. Similarly, using same command, Test Bench also could be written as shown below.



```
`timescale 1ns/1ps
module counter(clk,m,rst,count);
input clk,m,rst;
output reg [7:0] count;
always@(posedge clk or negedge rst)
begin
if(!rst)
count=0;
else if(m)
count=count+1;
else
count=count-1;
end
endmodule
```

RTL Code for a 16-bit Synchronous Up-Down Counter

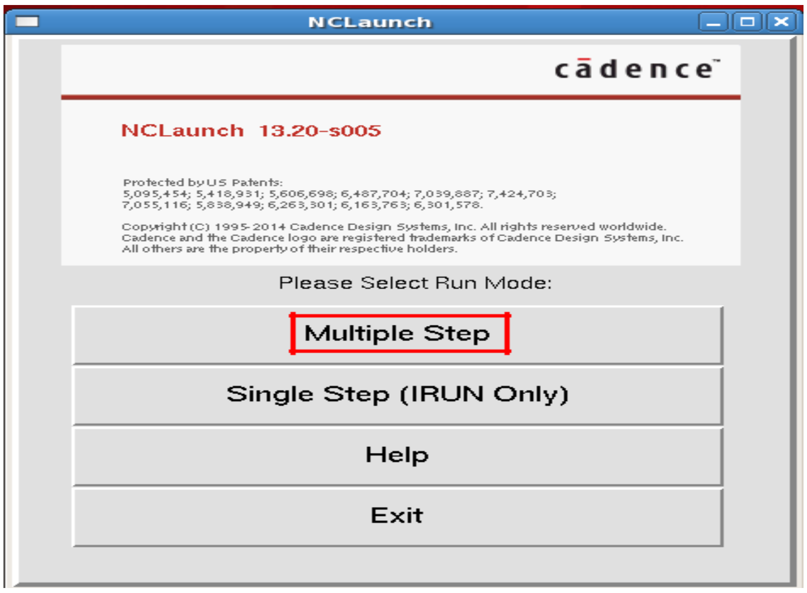


```
`timescale 1ns/1ps
module counter_test;
reg clk, rst,m;
wire [15:0] count;
initial
begin
clk=0;
rst=0;#25;
rst=1;
end
initial
begin
m=1;
#600 m=0;
rst=0;#25;
rst=1;
#500 m=0;
end
initial $sdf_annotate ("delays.sdf" , counter_test.counter1, ,"sdf.log");
initial $sdf_annotate ("counter.sdf" , counter_test.counter1, ,"sdf.log");

counter counter1(clk,m,rst, count);

always #5 clk=~clk;

initial $monitor("Time=%t rst=%b clk=%b count=%b", $time,rst,clk,count);

initial
#1400 $finish;

endmodule
```

Test Bench for the Up-Down Counter

**Module 2:** Functional Simulation

1. To perform Functional Simulation, "**Incisive**" tool is to be used.
2. In your terminal, type the command "**nclaunch  -new**" to open the tool.

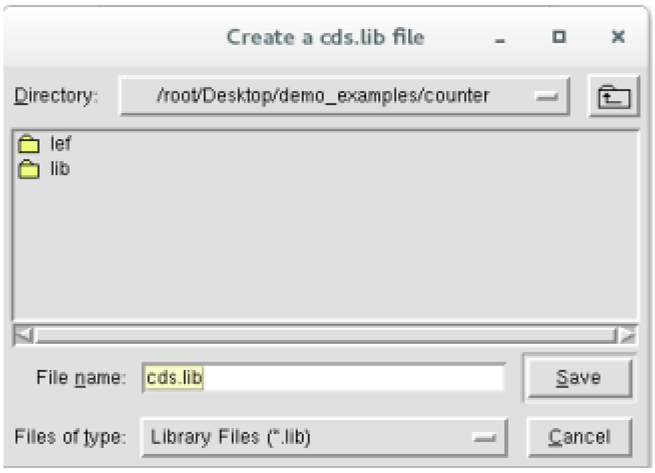


Protected by US Patents:
5,095,454; 5,418,931; 5,606,698; 6,487,704; 7,039,887; 7,424,703;
7,055,116; 5,838,949; 6,263,301; 6,163,763; 6,301,578.

Copyright (C) 1995-2014 Cadence Design Systems, Inc. All rights reserved worldwide.
Cadence and the Cadence logo are registered trademarks of Cadence Design Systems, Inc.
All others are the property of their respective holders.

3. Select "**Multiple Step**". And then select "**Create cds.lib**"

**Note :** The '-new' switch is used only for the first time the design is being run. For the next time on wards, the command to be used could be 'nclaunch' only.

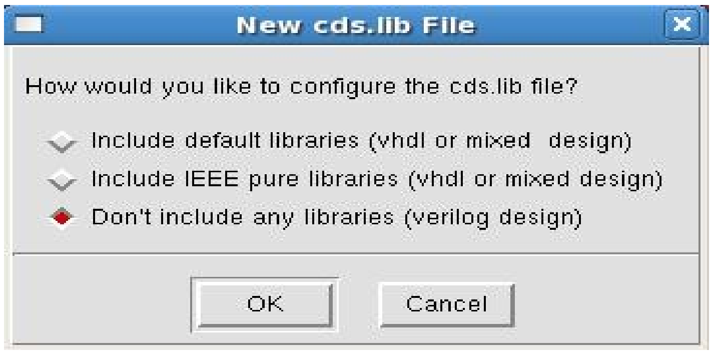


Directory: /root/Desktop/demo_examples/counter

File name: cds.lib

Files of type: Library Files (*.lib)

4. Save the cds.lib file. It is a tool file that holds the design location information for easy access by the tool.

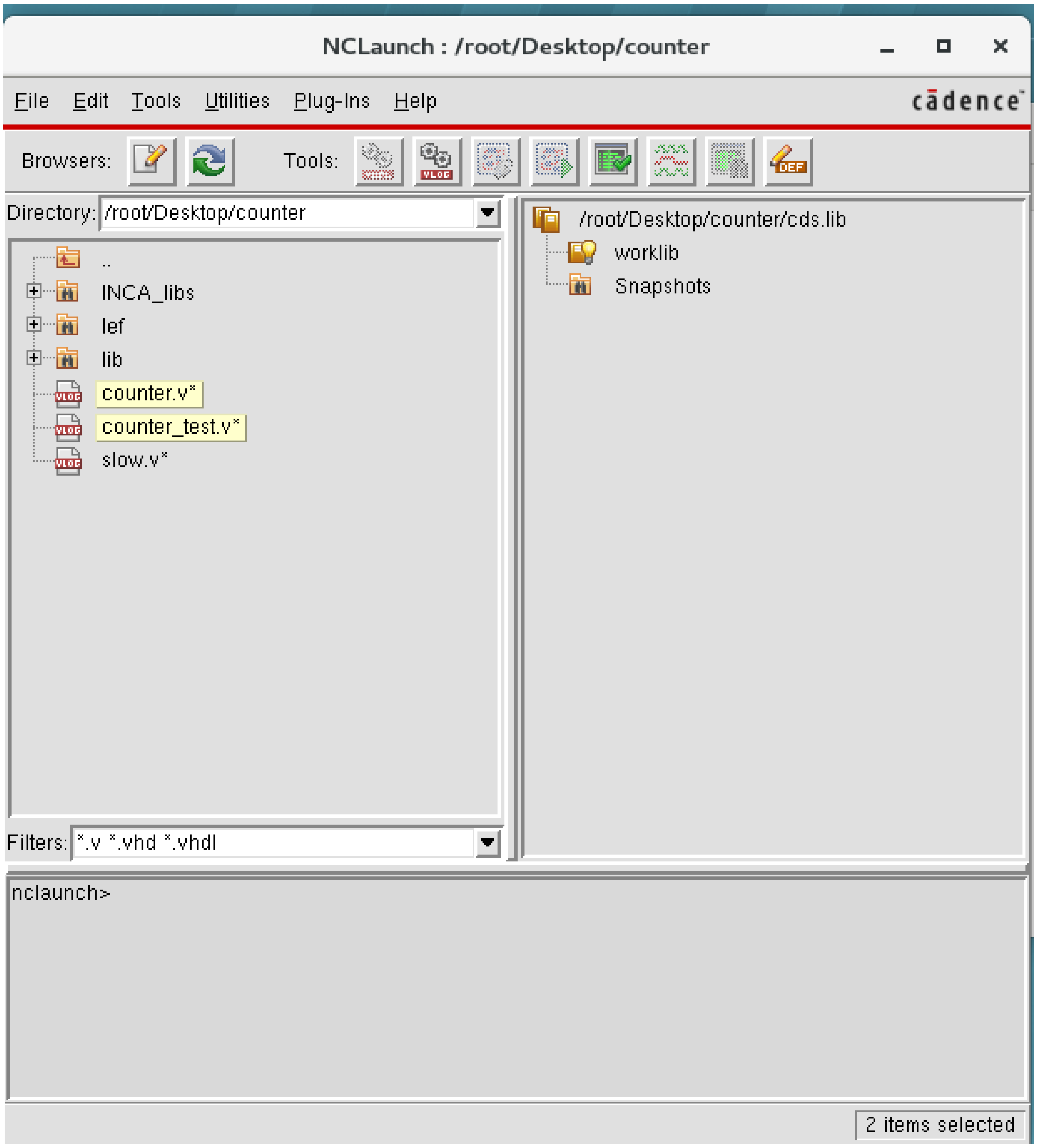

New cds.lib File

How would you like to configure the cds.lib file?

- Include default libraries (vhdl or mixed design)
- Include IEEE pure libraries (vhdl or mixed design)
- [✓ Don't include any libraries (verilog design)]

5. Based on the Libraries available and the type of RTL Code written, one of the three shown above is to be selected. Cadence tool suite provides default gpdk libraries. Here, counter RTL is of Verilog Format and hence third option is selected.
6. A new pop-up "nclaunch" opens which will contain all the .v and .vhdl files as per the cds.lib file created.
7. **Functional Simulation using Cadence runs in 3 stages:**
   **→ Compilation of Verilog/VHDL Code and/or Test Bench**
   **→ Elaboration of the Code & Test Bench Compiled**
   **→ Simulating the Test Bench or Top Module[in absence of Test Bench]**

8. A set of tools are shown in the nclaunch window which refer to **VHDL Compiler, Verilog Compiler, Elaborator, Simulator** corresponding from Left To Right.
9. Select the .v or .vhdl files to be compiled and launch Compiler. On successful completion of compilation, on the Right hand Side, the modules appear under "**Worklib**" .
10. Select the Module under Worklib and "**Launch Elaborator**" . On successful completion of Elaboration, "**Snapshots**" are generated.
11. Select the Test Bench under snapshots and "**Launch Simulator**" .
12. The above steps are depicted under following snapshosts.


NCLaunch : /root/Desktop/counter

Directory: /root/Desktop/counter

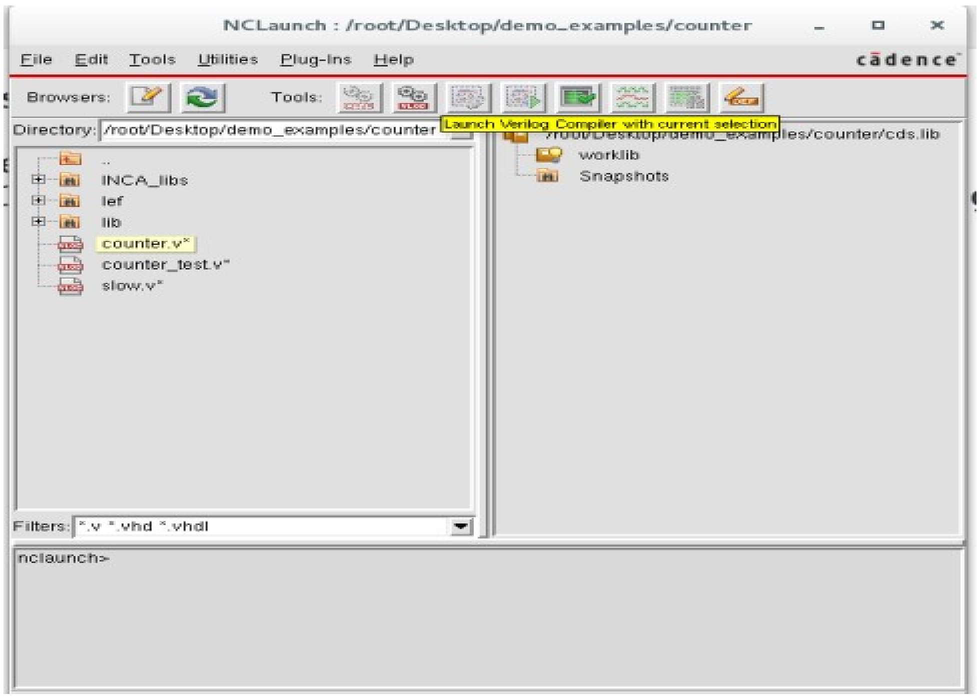
```
..
INCA_libs
lef
lib
counter.v*
counter_test.v*
slow.v*
```

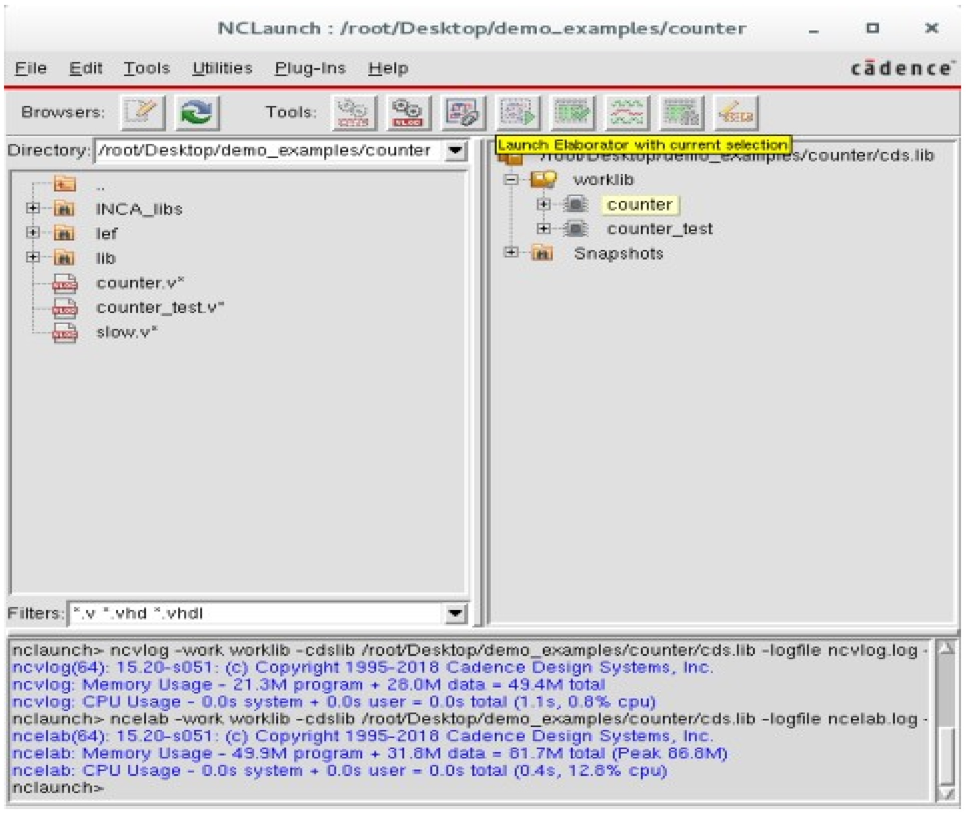
/root/Desktop/counter/cds.lib
- worklib
- Snapshots

Filters: *.v *.vhd *.vhdl

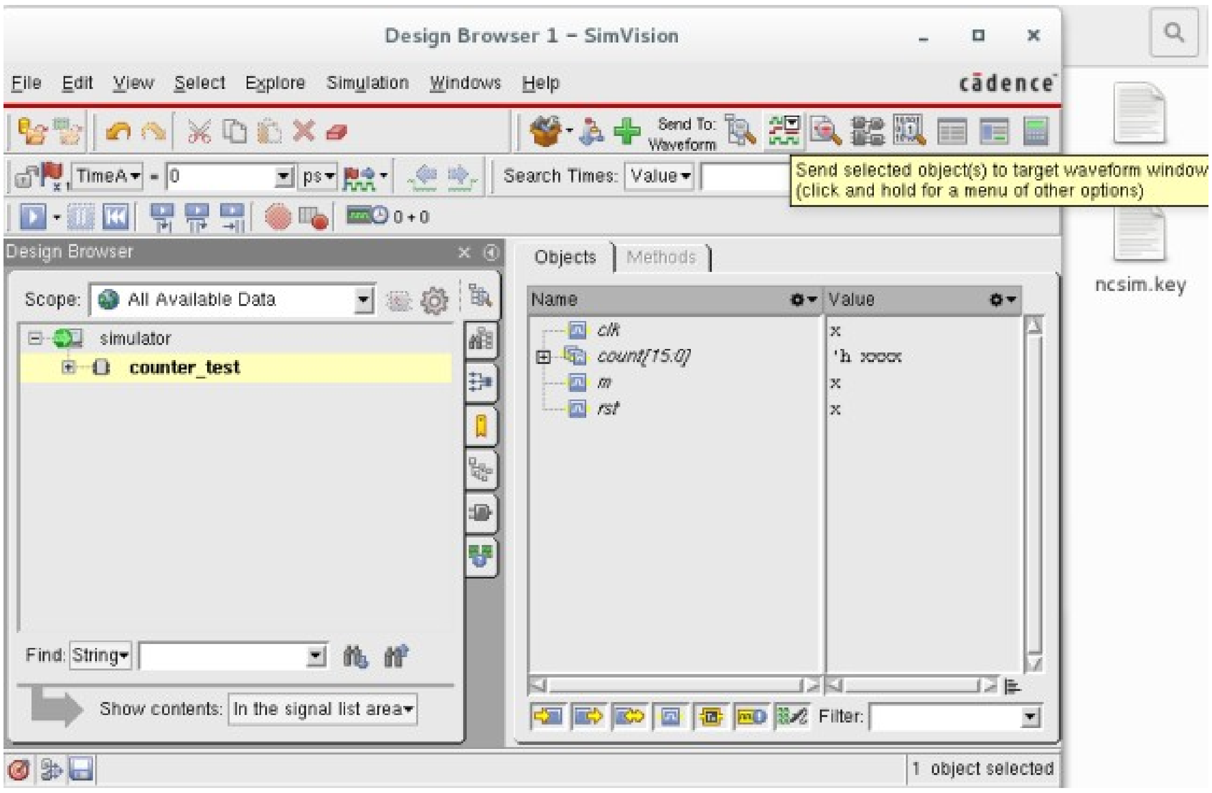
```
nclaunch>
```

2 items selected


NCLaunch : /root/Desktop/demo_examples/counter
File Edit Tools Utilities Plug-Ins Help
cādence
Browsers:
Tools:
Launch Verilog Compiler with current selection
Directory: /root/Desktop/demo_examples/counter
/root/Desktop/demo_examples/counter/cds.lib
..
INCA_libs
lef
lib
counter.v*
counter_test.v*
slow.v*
worklib
Snapshots
Filters: *.v *.vhd *.vhdl
nclaunch>


NCLaunch : /root/Desktop/demo_examples/counter
File Edit Tools Utilities Plug-Ins Help
cādence
Browsers:
Tools:
Launch Elaborator with current selection
Directory: /root/Desktop/demo_examples/counter
/root/Desktop/demo_examples/counter/cds.lib
..
INCA_libs
lef
lib
counter.v*
counter_test.v*
slow.v*
worklib
counter
counter_test
Snapshots
Filters: *.v *.vhd *.vhdl

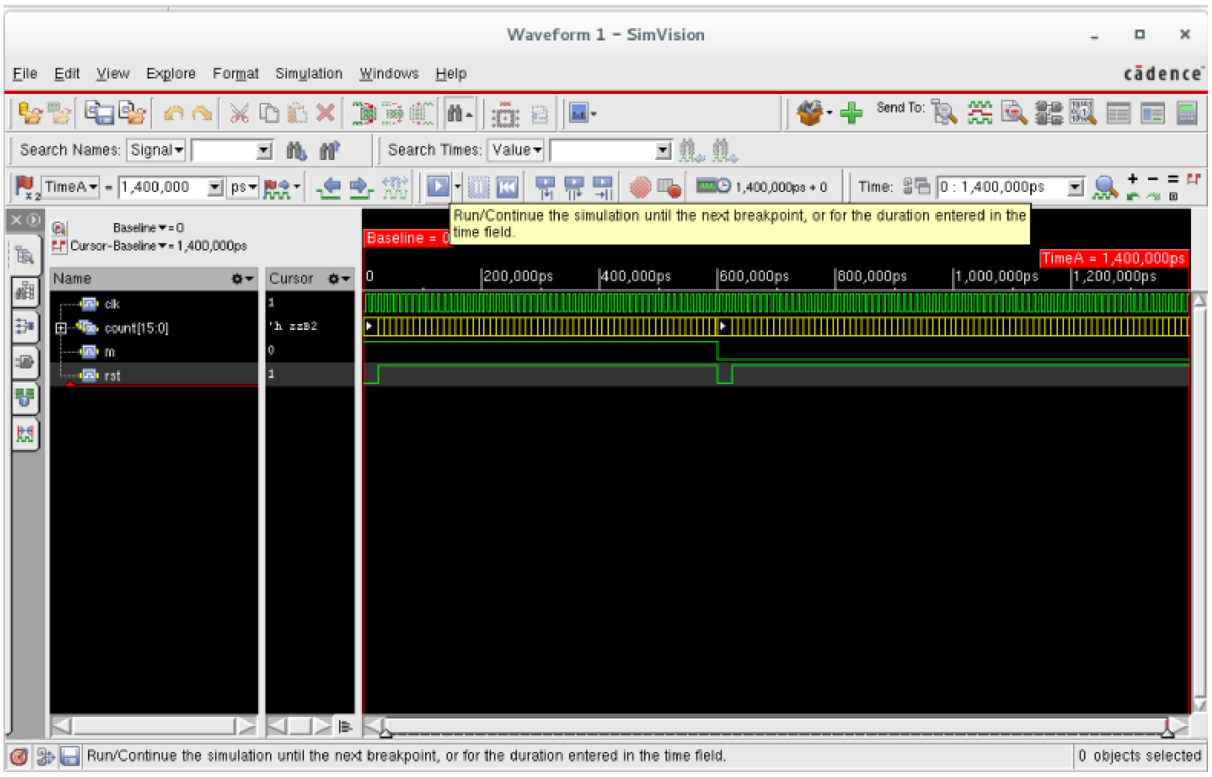
```
nclaunch> ncvlog -work worklib -cdslib /root/Desktop/demo_examples/counter/cds.lib -logfile ncvlog.log -
ncvlog(64): 15.20-s051: (c) Copyright 1995-2018 Cadence Design Systems, Inc.
ncvlog: Memory Usage - 21.3M program + 28.0M data = 49.4M total
ncvlog: CPU Usage - 0.0s system + 0.0s user = 0.0s total (1.1s, 0.8% cpu)
nclaunch> ncelab -work worklib -cdslib /root/Desktop/demo_examples/counter/cds.lib -logfile ncelab.log -
ncelab(64): 15.20-s051: (c) Copyright 1995-2018 Cadence Design Systems, Inc.
ncelab: Memory Usage - 49.9M program + 31.8M data = 81.7M total (Peak 86.8M)
ncelab: CPU Usage - 0.0s system + 0.0s user = 0.0s total (0.4s, 12.8% cpu)
nclaunch>
```


Design Browser 1 – SimVision
File Edit View Select Explore Simulation Windows Help
cadence
Send To: Waveform
Send selected object(s) to target waveform window
(click and hold for a menu of other options)
Search Times: Value
Design Browser
Scope: All Available Data
simulator
counter_test
Find: String
Show contents: In the signal list area
Objects Methods
Name Value
clk x
count[15:0] 'h xxxx
m x
rst x
Filter:
1 object selected
ncsim.key

The Design Browser pops-up and The Test Bench Module name can be seen on the left and the Pin list on the right when selected. For Simulation, The number of Pins / Ports to be simulated can be selected.

Make a right click on the selected and Select “Send to Waveform Window”.

In the waveform window, we can see different ports in the design. Now click on the Run simulation key to start the simulation. Use the ‘pause’ key to interrupt or stop the simulation. Use different options like zoom in, zoom out etc to analyze the plot.


Waveform 1 - SimVision
File Edit View Explore Format Simulation Windows Help
cādence
Search Names: Signal
Search Times: Value
TimeA = 1,400,000 ps
1,400,000ps + 0
Time: 0 : 1,400,000ps
Run/Continue the simulation until the next breakpoint, or for the duration entered in the time field.
Baseline = 0
Cursor-Baseline = 1,400,000ps
Baseline = 0
TimeA = 1,400,000ps
Name
Cursor
0
200,000ps
400,000ps
600,000ps
800,000ps
1,000,000ps
1,200,000ps
clk
count[15:0]
m
rst
1
'h zzB2
0
1
Run/Continue the simulation until the next breakpoint, or for the duration entered in the time field.
0 objects selected

**Module 3: Synthesis**

**Inputs for Synthesis :**

1. **RTL Code (.v or .vhdl)**
2. **Chip Level SDC (System Design Constraints)**
3. **Liberty Files (.lib)**

**Expected Outputs of Synthesis :**

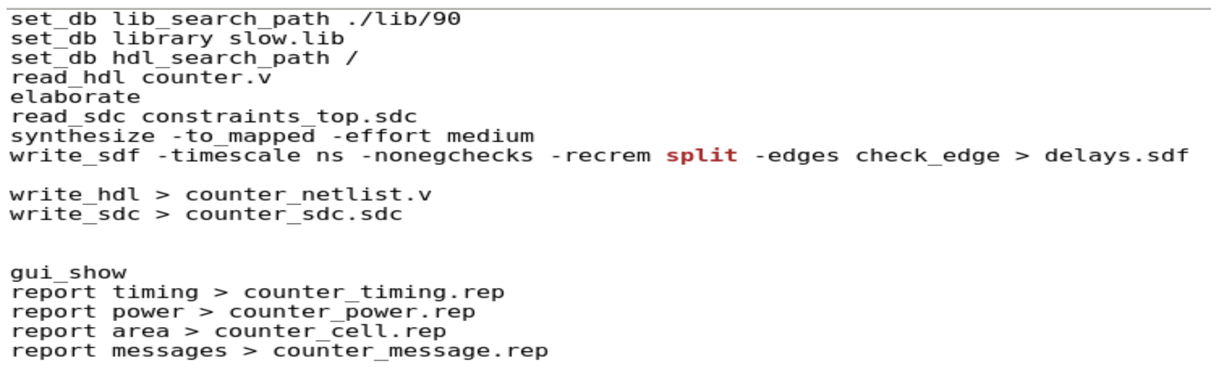
1. **Gate Level Netlist**
2. **Block Level Netlist**
3. **Timing, Area, Power reports**

**Synthesis** is a 3-stage process which converts Virtual RTL Logic into Physical Gates in order to give a Physical Shape to the design through Physical Design.

Synthesis runs in following stages :

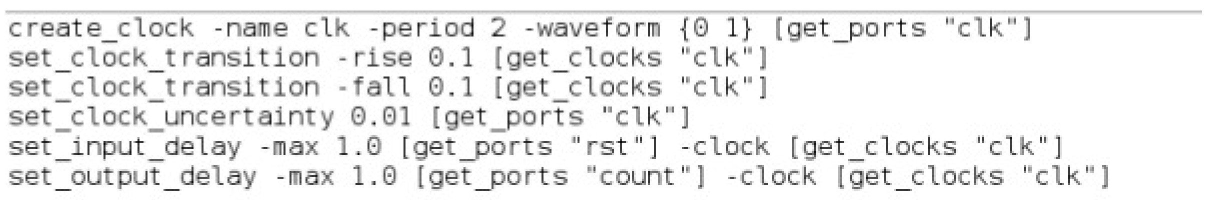
→ Translation - RTL Codes are compiled
→ Elaboration / Mapping - Pieces of Logic are replaced with corresponding Gates from Libraries with same Functionality
→ Optimization - Tool tries to reduce cell count without affecting the functionality

To run the synthesis, the following script can be used.



```
set_db lib_search_path ./lib/90
set_db library slow.lib
set_db hdl_search_path /
read_hdl counter.v
elaborate
read_sdc constraints_top.sdc
synthesize -to_mapped -effort medium
write_sdf -timescale ns -nonegchecks -recrem split -edges check_edge > delays.sdf

write_hdl > counter_netlist.v
write_sdc > counter_sdc.sdc


gui_show
report timing > counter_timing.rep
report power > counter_power.rep
report area > counter_cell.rep
report messages > counter_message.rep
```

Chip Level SDC is as follows :



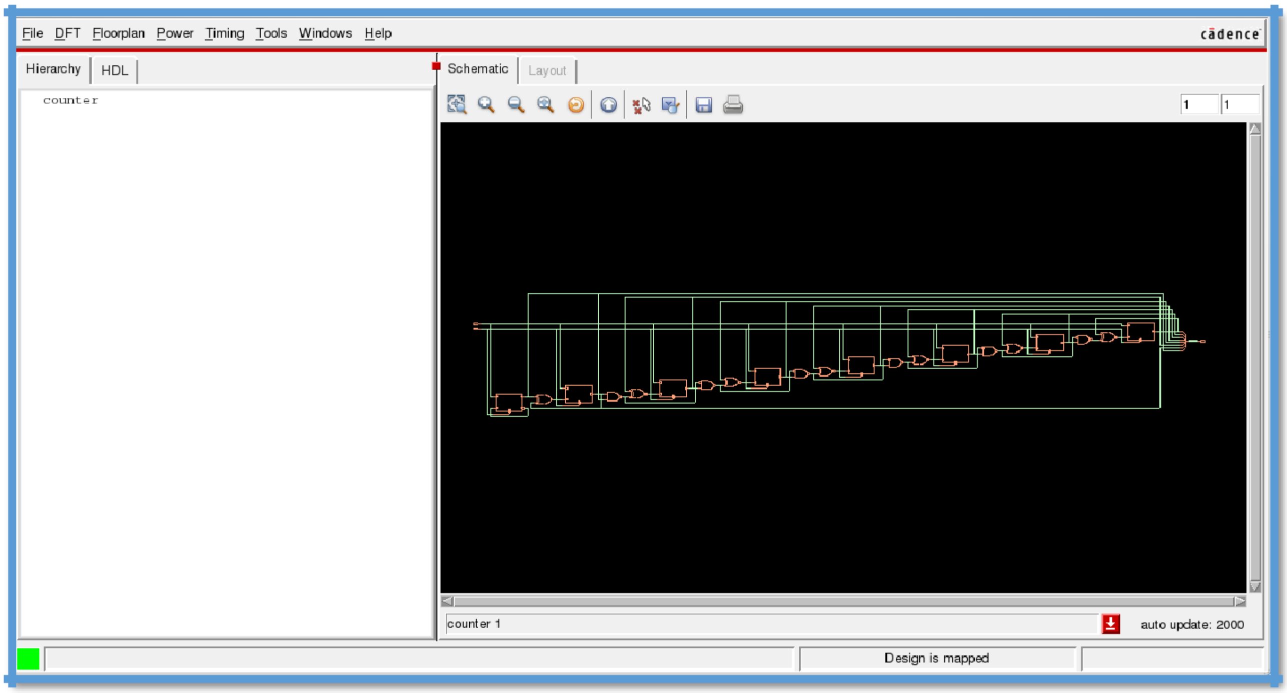
```
create_clock -name clk -period 2 -waveform {0 1} [get_ports "clk"]
set_clock_transition -rise 0.1 [get_clocks "clk"]
set_clock_transition -fall 0.1 [get_clocks "clk"]
set_clock_uncertainty 0.01 [get_ports "clk"]
set_input_delay -max 1.0 [get_ports "rst"] -clock [get_clocks "clk"]
set_output_delay -max 1.0 [get_ports "count"] -clock [get_clocks "clk"]
```

Close out all INCISIVE windows and in the same terminal type in the following command to run Synthesis.

**genus -f rc_script.tcl**

The tcl [Tool Command Language] script runs executing each command one after the other.

A window of Genus GUI pops – up with the top hier cell on the left top. Make a right click and select Schematic Viewer → In Main.



The Gate level Circuit that implements the RTL Logic can be seen and analysed.

As from the script, Block Level SDC, Gate Level Netlist, Timing, Power and Area reports are generated which are readable.

The Timing report gives the path with Worst timing.

The area report gives Cell count and Total area occupied by them.

The total power consumed by those cells are given in Power report.

**Module 4 : Physical Design**

**Mandatory Inputs for PD :**

1. **Gate Level Netlist [Output of Synthesis]**
2. **Block Level SDC [Output of Synthesis]**
3. **Liberty Files (.lib)**
4. **LEF Files (Layer Exchange Format)**

**Expected Outputs from PD :**

1. **GDS II File (Graphical Data Stream for Information Interchange – Feed In for Fabrication Unit).**

Close out all windows relating to Genus and in the terminal, type the command

**innovus** (Press Enter)

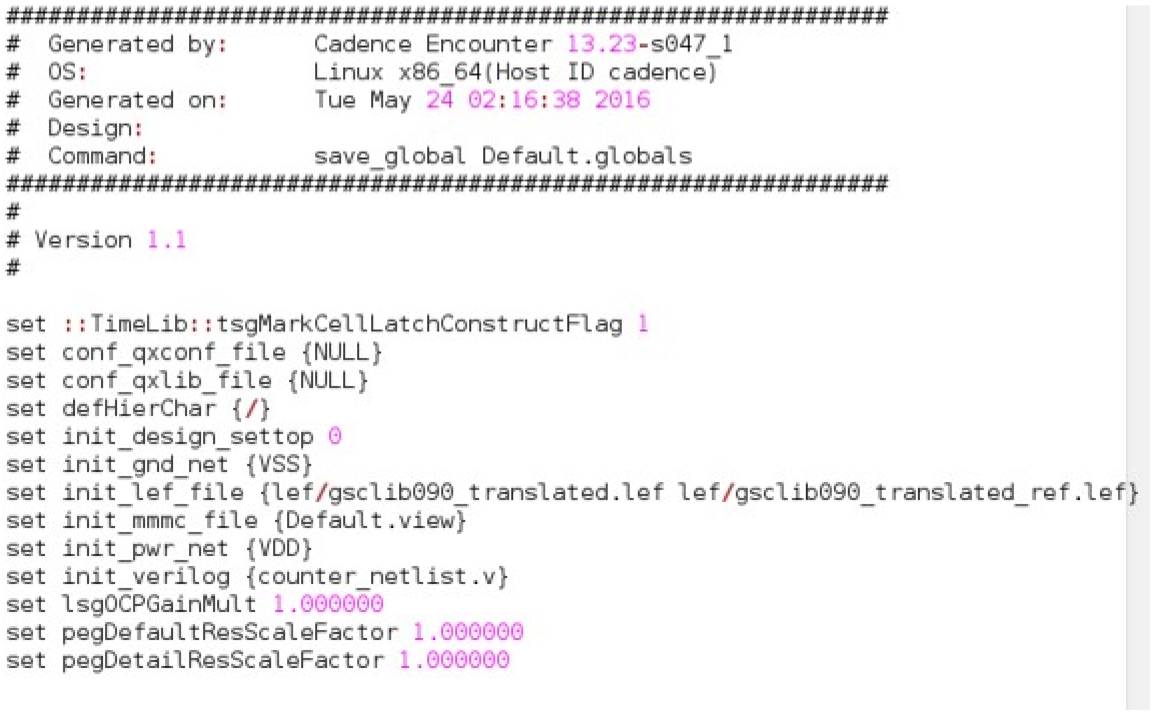
For Innovus tool, a GUI opens and also the terminal enters into innovus command prompt where in the tool commands can be entered.

Physical Design involves 5 stages as following :

After Importing Design,
→ Floor Planning
→ Power Planning
→ Placement
→ CTS (Clock Tree Synthesis)
→ Routing

## Module 4.1 : Importing Design

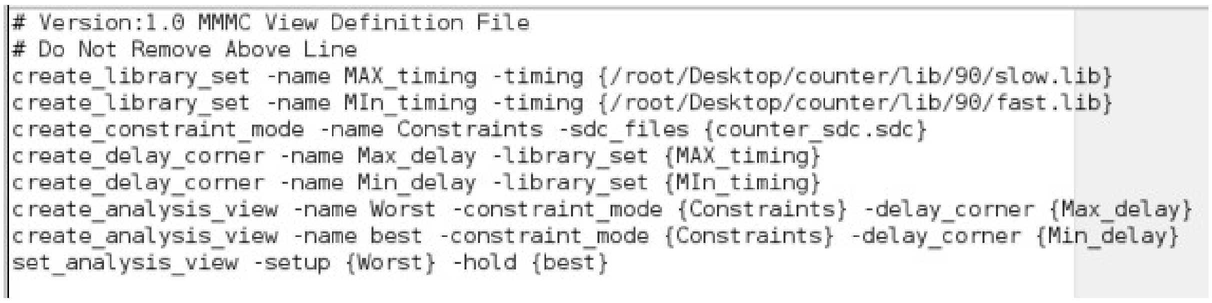
To Import Design, all the Mandatory Inputs are to be loaded and this can be done using script files named with .globals and .view/.tcl



```
#################################################################
#  Generated by:      Cadence Encounter 13.23-s047_1
#  OS:                Linux x86_64(Host ID cadence)
#  Generated on:      Tue May 24 02:16:38 2016
#  Design:
#  Command:           save_global Default.globals
#################################################################
#
# Version 1.1
#

set ::TimeLib::tsgMarkCellLatchConstructFlag 1
set conf_qxconf_file {NULL}
set conf_qxlib_file {NULL}
set defHierChar {/}
set init_design_settop 0
set init_gnd_net {VSS}
set init_lef_file {lef/gsclib090_translated.lef lef/gsclib090_translated_ref.lef}
set init_mmmc_file {Default.view}
set init_pwr_net {VDD}
set init_verilog {counter_netlist.v}
set lsgOCPGainMult 1.000000
set pegDefaultResScaleFactor 1.000000
set pegDetailResScaleFactor 1.000000
```

Globals File to import design using Mandatory Inputs

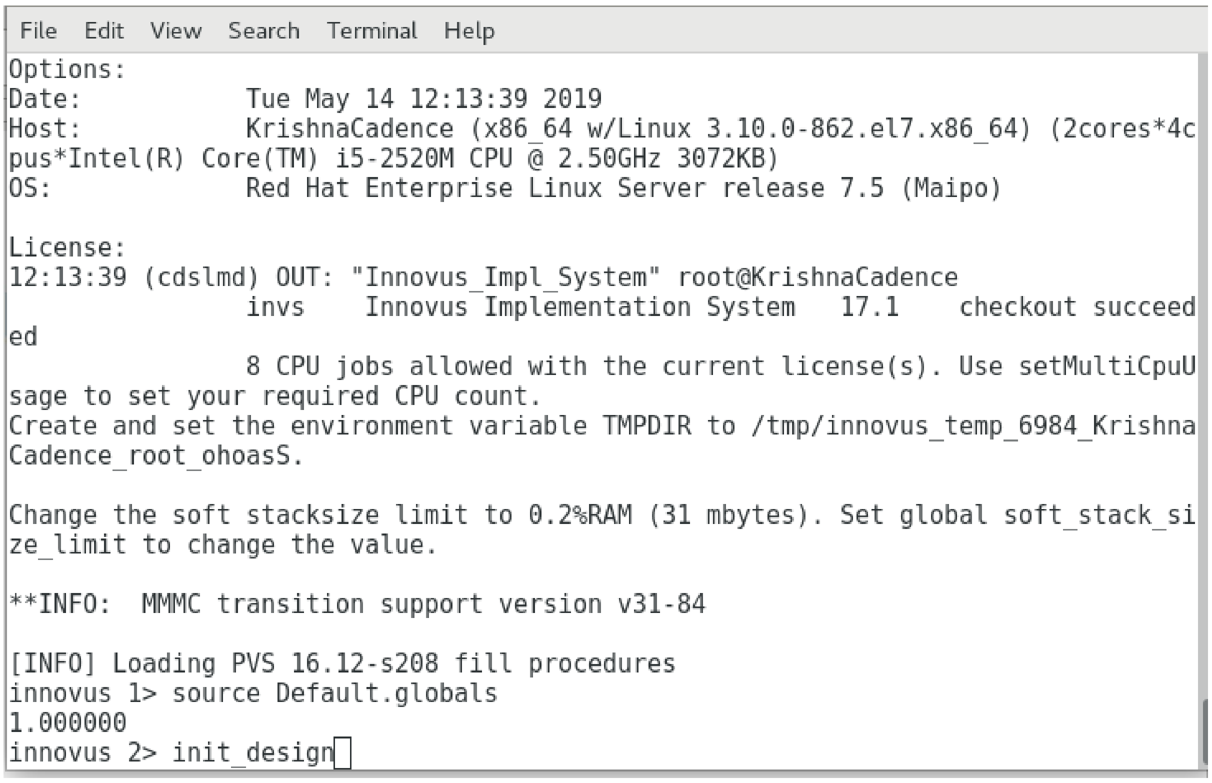
The Globals file reads in the LEF's and Gate Level Netlist and .view file implicitly.



```
# Version:1.0 MMMC View Definition File
# Do Not Remove Above Line
create_library_set -name MAX_timing -timing {/root/Desktop/counter/lib/90/slow.lib}
create_library_set -name MIn_timing -timing {/root/Desktop/counter/lib/90/fast.lib}
create_constraint_mode -name Constraints -sdc_files {counter_sdc.sdc}
create_delay_corner -name Max_delay -library_set {MAX_timing}
create_delay_corner -name Min_delay -library_set {MIn_timing}
create_analysis_view -name Worst -constraint_mode {Constraints} -delay_corner {Max_delay}
create_analysis_view -name best -constraint_mode {Constraints} -delay_corner {Min_delay}
set_analysis_view -setup {Worst} -hold {best}
```

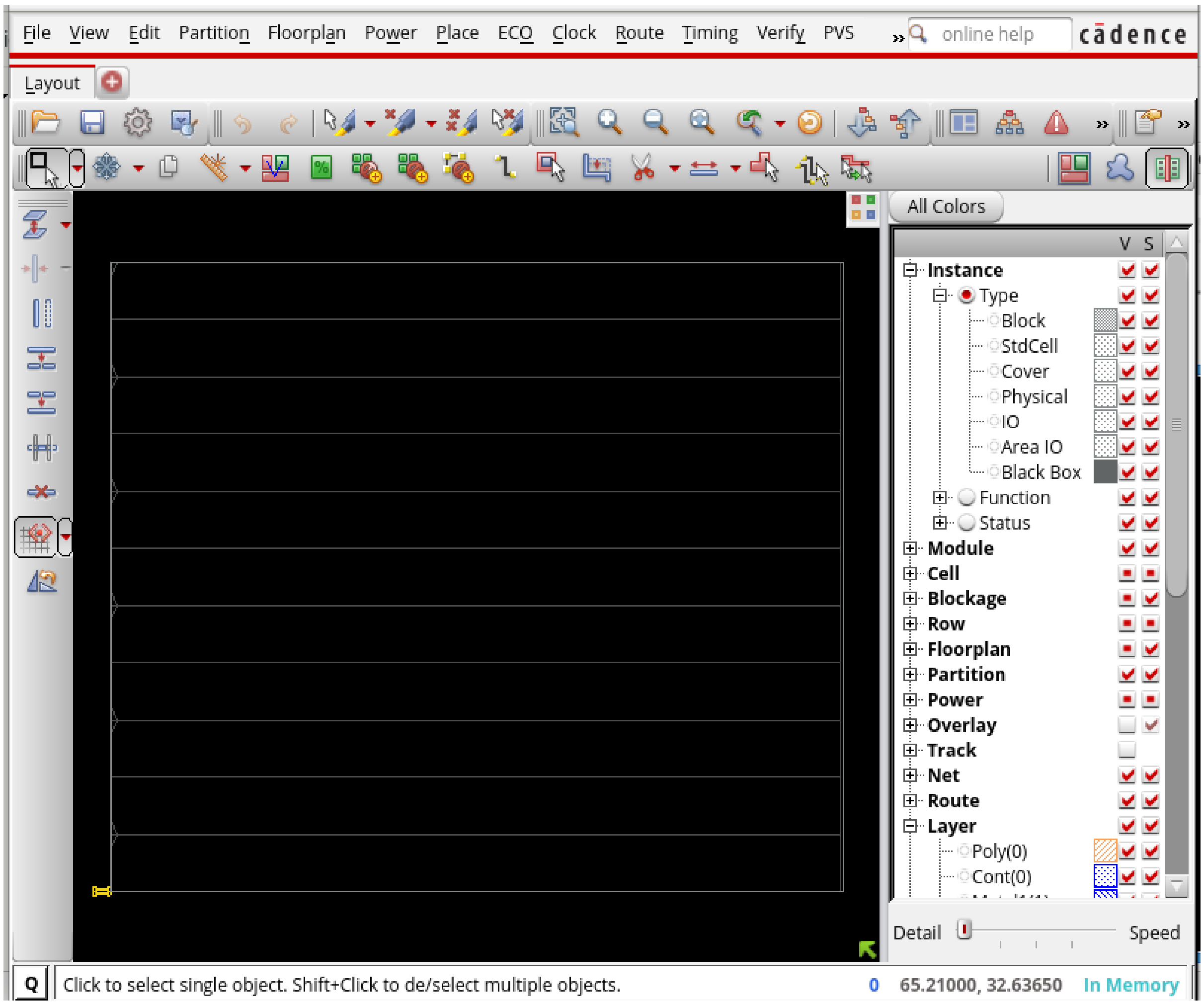
The .view file reads Liberty Files and Block Level SDC to create various PVT Corners for analysis.

In the terminal command prompt, type the commands as shown. The design is imported and “Core Area” is calculated by tool and shown on GUI.



```
Options:
Date:           Tue May 14 12:13:39 2019
Host:           KrishnaCadence (x86_64 w/Linux 3.10.0-862.el7.x86_64) (2cores*4c
pus*Intel(R) Core(TM) i5-2520M CPU @ 2.50GHz 3072KB)
OS:             Red Hat Enterprise Linux Server release 7.5 (Maipo)

License:
12:13:39 (cdslmd) OUT: "Innovus_Impl_System" root@KrishnaCadence
                invs    Innovus Implementation System   17.1    checkout succeed
ed
                8 CPU jobs allowed with the current license(s). Use setMultiCpuU
sage to set your required CPU count.
Create and set the environment variable TMPDIR to /tmp/innovus_temp_6984_Krishna
Cadence_root_ohoasS.

Change the soft stacksize limit to 0.2%RAM (31 mbytes). Set global soft_stack_si
ze_limit to change the value.

**INFO:  MMMC transition support version v31-84

[INFO] Loading PVS 16.12-s208 fill procedures
innovus 1> source Default.globals
1.000000
innovus 2> init_design
```

The Horizontal Lines on the GUI across the Core Area are alternative VDD and VSS tracks and Standard Cell Placement Rows.



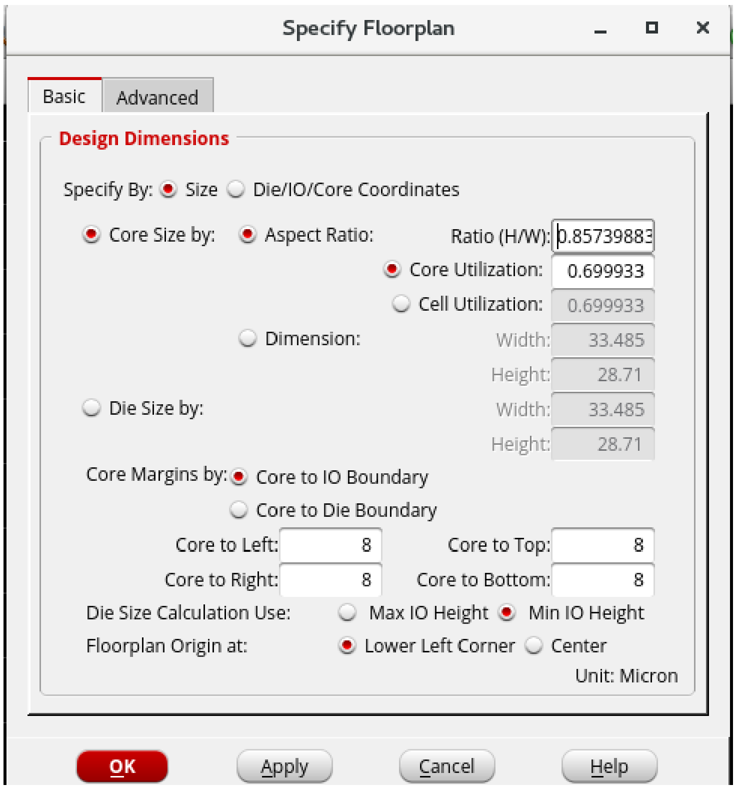
## Module 4.2 : Floorplan

**Steps under Floorplan :**

1. **Aspect Ratio** [Ratio of Vertical Height to Horizontal Width of Core]
2. **Core Utilisation** [The total Core Area % to be used for Floor Planning]
3. **Channel Spacing between Core Boundary to IO Boundary**

Select **Floorplan → Specify Floorplan** to modify/add concerned values to the above Factors.

On adding/modifying the concerned values, the core area is also modified.



Specify Floorplan

Basic | Advanced

Design Dimensions

Specify By: [✓ Size] Die/IO/Core Coordinates

[✓ Core Size by:] [✓ Aspect Ratio:] Ratio (H/W): 0.85739883

[✓ Core Utilization:] 0.699933

Cell Utilization: 0.699933

Dimension: Width: 33.485 Height: 28.71

Die Size by: Width: 33.485 Height: 28.71

Core Margins by: [✓ Core to IO Boundary] Core to Die Boundary

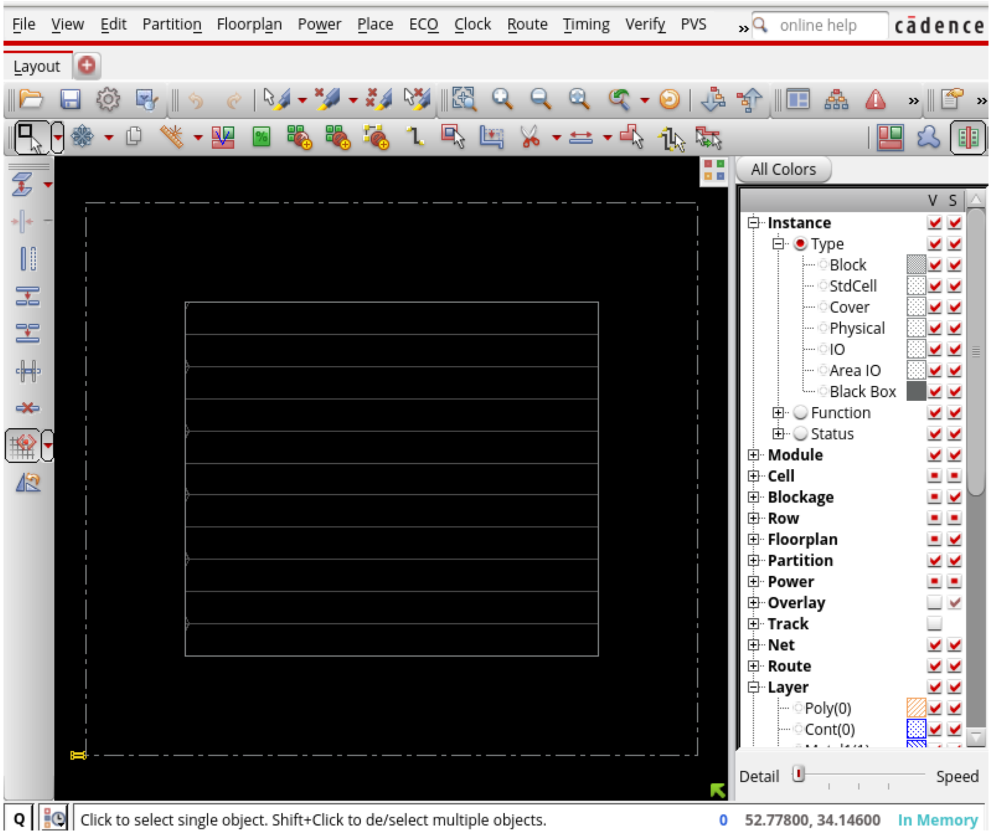
Core to Left: 8 Core to Top: 8

Core to Right: 8 Core to Bottom: 8

Die Size Calculation Use: Max IO Height [✓ Min IO Height]

Floorplan Origin at: [✓ Lower Left Corner] Center

Unit: Micron



Click to select single object. Shift+Click to de/select multiple objects. 0 52.77800, 34.14600 In Memory

The Yellow patch on the Left Bottom are the group of “Unassigned pins” which are to be placed along the IO Boundary along with the Standard Cells [Gates].

**Module 4.3 : Power Planning**

**Steps under Power Planning :**

1. **Connect Global Net Connects**
2. **Adding Power Rings**
3. **Adding Power Strings**
4. **Special Route**

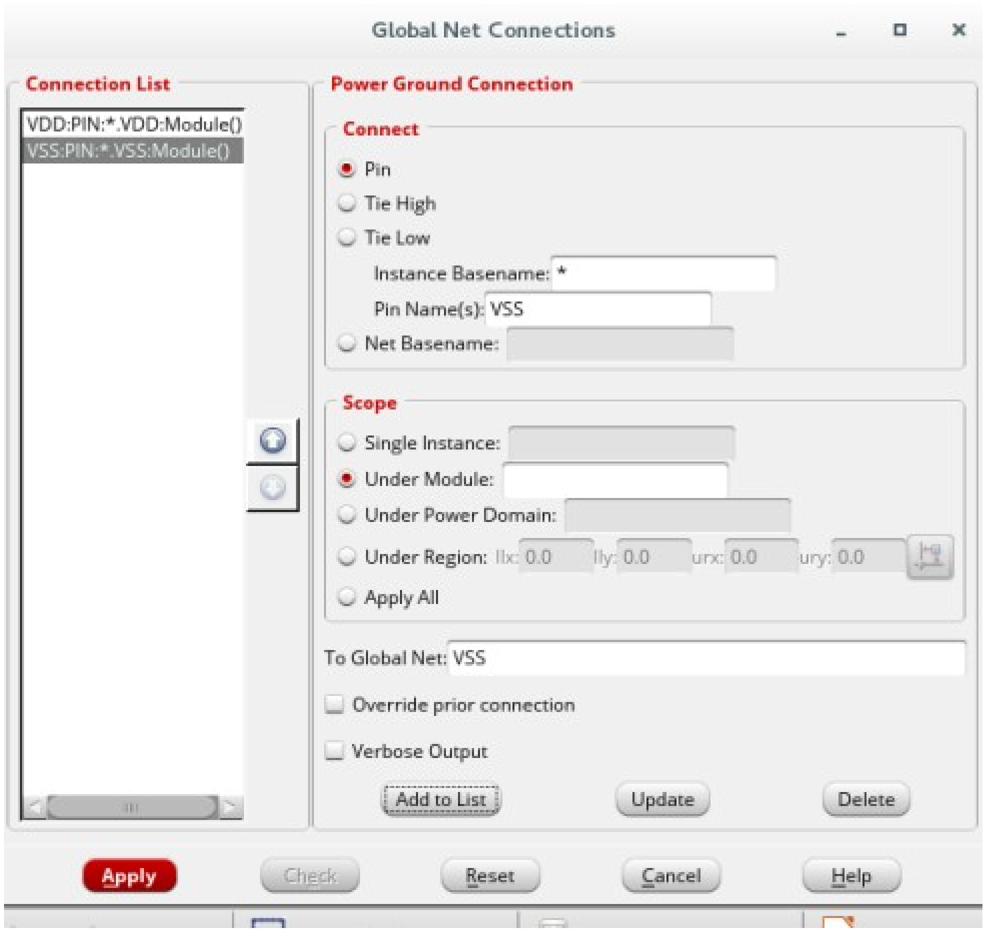
During the stage of Importing Design, under the Globals file, Two command lines state the names of Power and Ground Nets.

However, in order to Current flow through these Power nets, they are to converge at a point, preferably a common net connected to a Pin.

Under **Connect Global Net Connects**, we create two pins, one for VDD and one for VSS connecting them to corresponding Global Nets as mentioned in Globals file.

Select **Power → Connect Global Nets..** to create “Pin” and “Connect to Global Net” as shown and use “Add to list”.

Click on “Apply” to direct the tool in enforcing the Pins and Net connects to Design and then Close the window.



**Connection List**

VDD:PIN:*.VDD:Module()
VSS:PIN:*.VSS:Module()

**Power Ground Connection**

**Connect**

- [✓ Pin]
- Tie High
- Tie Low
- Instance Basename: *
- Pin Name(s): VSS
- Net Basename:

**Scope**

- Single Instance:
- [✓ Under Module:]
- Under Power Domain:
- Under Region: llx: 0.0 lly: 0.0 urx: 0.0 ury: 0.0
- Apply All

To Global Net: VSS

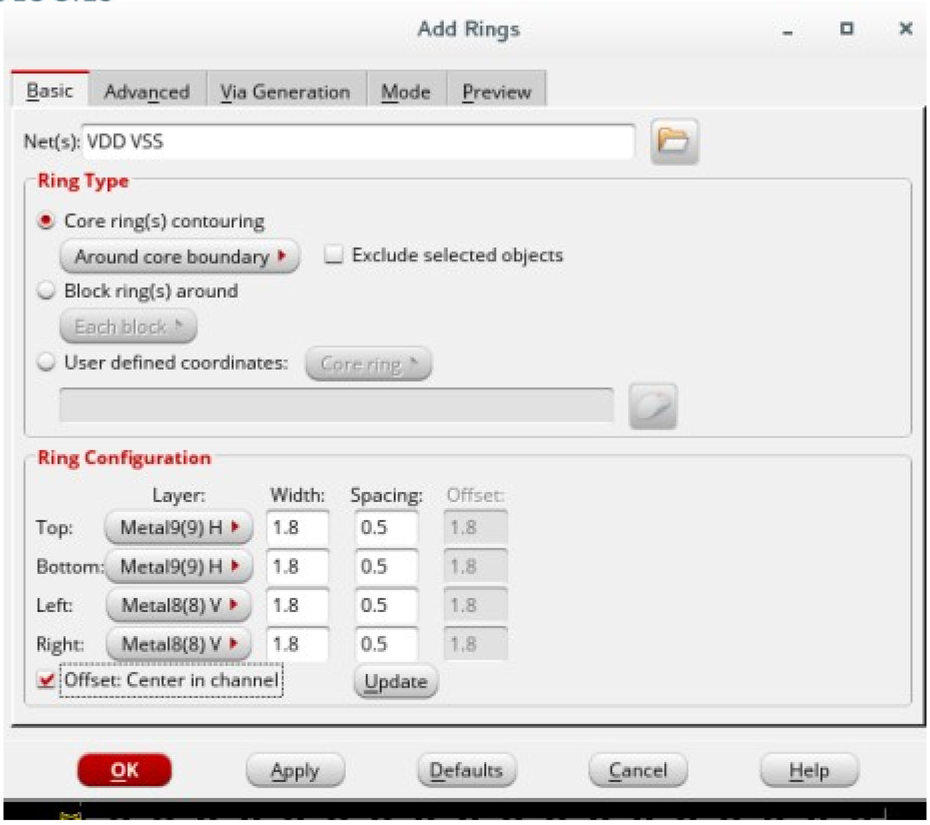
☐ Override prior connection

☐ Verbose Output

In order to Tap in Power from a distant Power supply, Wider Nets and Parallel connections improve efficiency. Moreover, the cells that would be placed inside the core area are expected to have shorter Nets for lower resistance.

Hence Power Rings [Around Core Boundary] and Power Stripes [Across Core Boundary] are added which satisfies the above conditions.

Select **Power → Power Planning → Add Rings** to add Power rings 'around Core Boundary'.


Add Rings

Basic | Advanced | Via Generation | Mode | Preview

Net(s): VDD VSS

**Ring Type**

(●) Core ring(s) contouring — Around core boundary; ☐ Exclude selected objects

( ) Block ring(s) around — Each block

( ) User defined coordinates: Core ring

**Ring Configuration**

| | Layer: | Width: | Spacing: | Offset: |
|---|---|---|---|---|
| Top: | Metal9(9) H | 1.8 | 0.5 | 1.8 |
| Bottom: | Metal9(9) H | 1.8 | 0.5 | 1.8 |
| Left: | Metal8(8) V | 1.8 | 0.5 | 1.8 |
| Right: | Metal8(8) V | 1.8 | 0.5 | 1.8 |

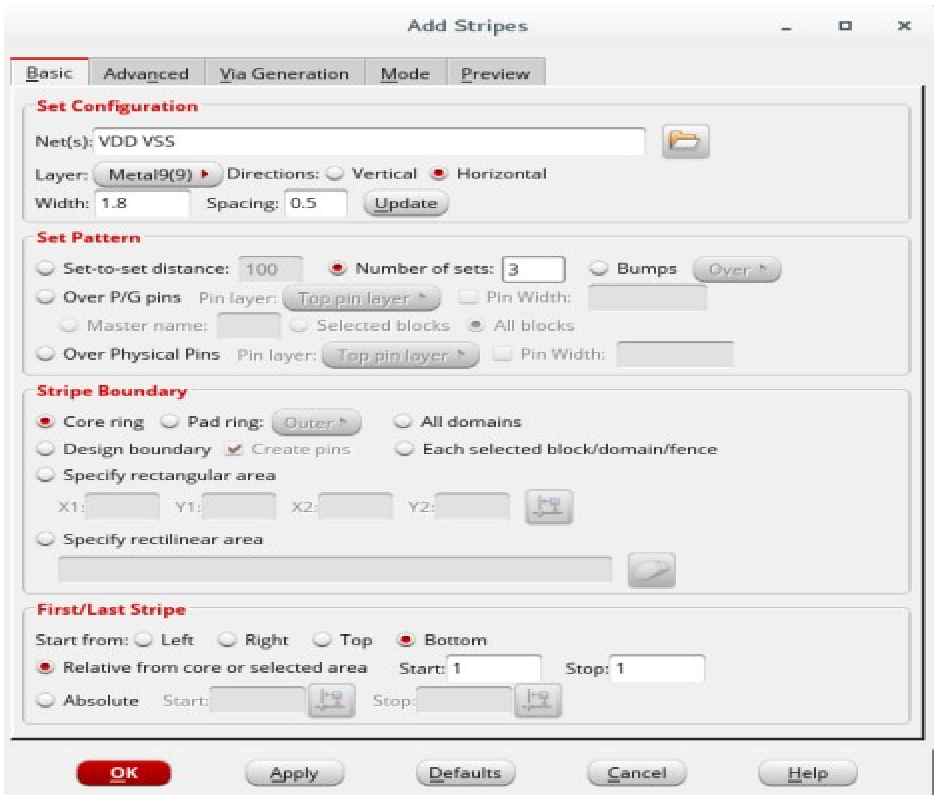
☑ Offset: Center in channel

→ Select the Nets from Browse option OR Directly type in the Global Net Names separated by a space being Case and Spelling Senstive.

→ Select the Highest Metals marked 'H' [Horizontal] for Top and Bottom and Metals marked 'V' [Vertical] for Right and Bottom. This is because Highest metals have Highest Widths and thus Lowest Resistance.

→ Click on Update after the selection and "Set Offset : Center in Channel" in order to get the Minimum Width and Minimum Spacing of the corresponding Metals and then Click "OK".

→ Similarly, Power Stripes are added using similar content to that of Power Rings.


Add Stripes

Basic | Advanced | Via Generation | Mode | Preview

**Set Configuration**

Net(s): VDD VSS

Layer: Metal9(9) Directions: Vertical [✓ Horizontal]

Width: 1.8 Spacing: 0.5 Update

**Set Pattern**

Set-to-set distance: 100 [✓ Number of sets: 3] Bumps Over

Over P/G pins Pin layer: Top pin layer Pin Width:

Master name: Selected blocks [✓ All blocks]

Over Physical Pins Pin layer: Top pin layer Pin Width:

**Stripe Boundary**

[✓ Core ring] Pad ring: Outer All domains

Design boundary [✓ Create pins] Each selected block/domain/fence

Specify rectangular area

X1: Y1: X2: Y2:

Specify rectilinear area

**First/Last Stripe**

Start from: Left Right Top [✓ Bottom]

[✓ Relative from core or selected area] Start: 1 Stop: 1

Absolute Start: Stop:

OK | Apply | Defaults | Cancel | Help

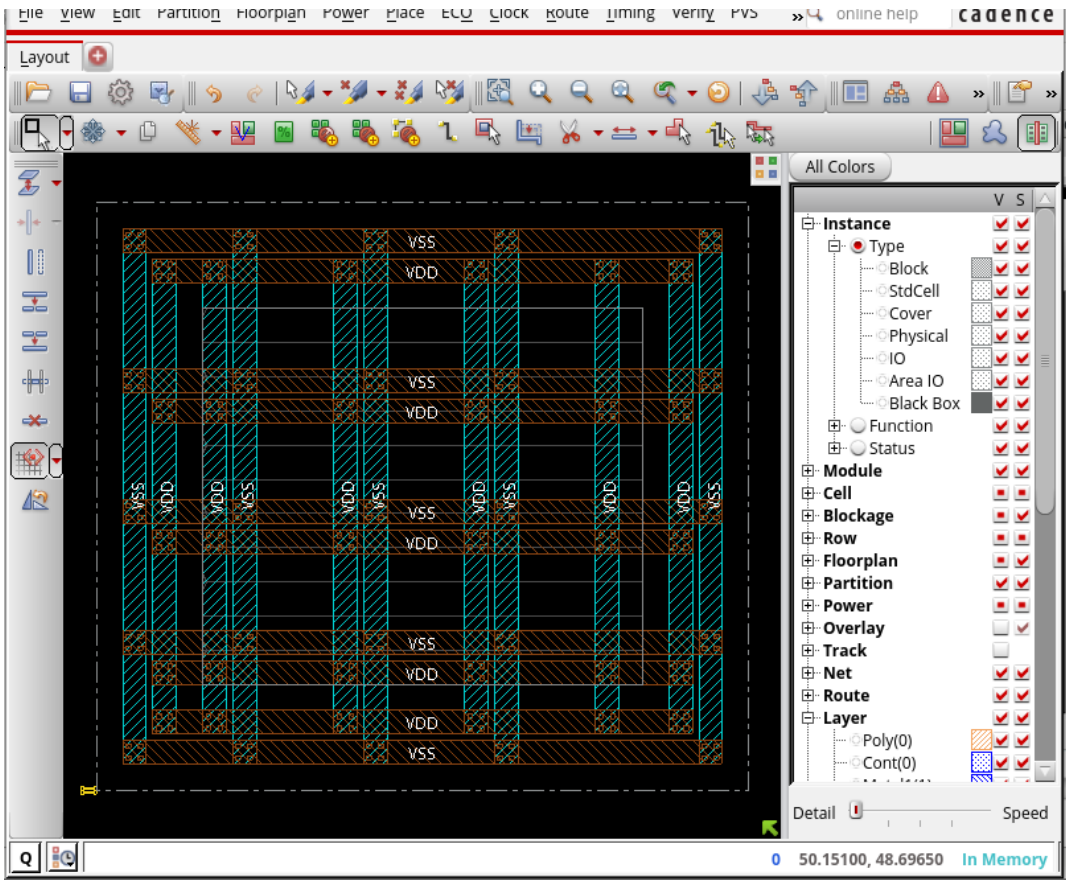
Factors to be considered under Power Stripes :

→ Nets

→ Metal and It’s Direction

→ Width and Spacing [Updated]

→ Set to Set Distance = ( Minimum Width of Metal + Min. Spacing ) x 2

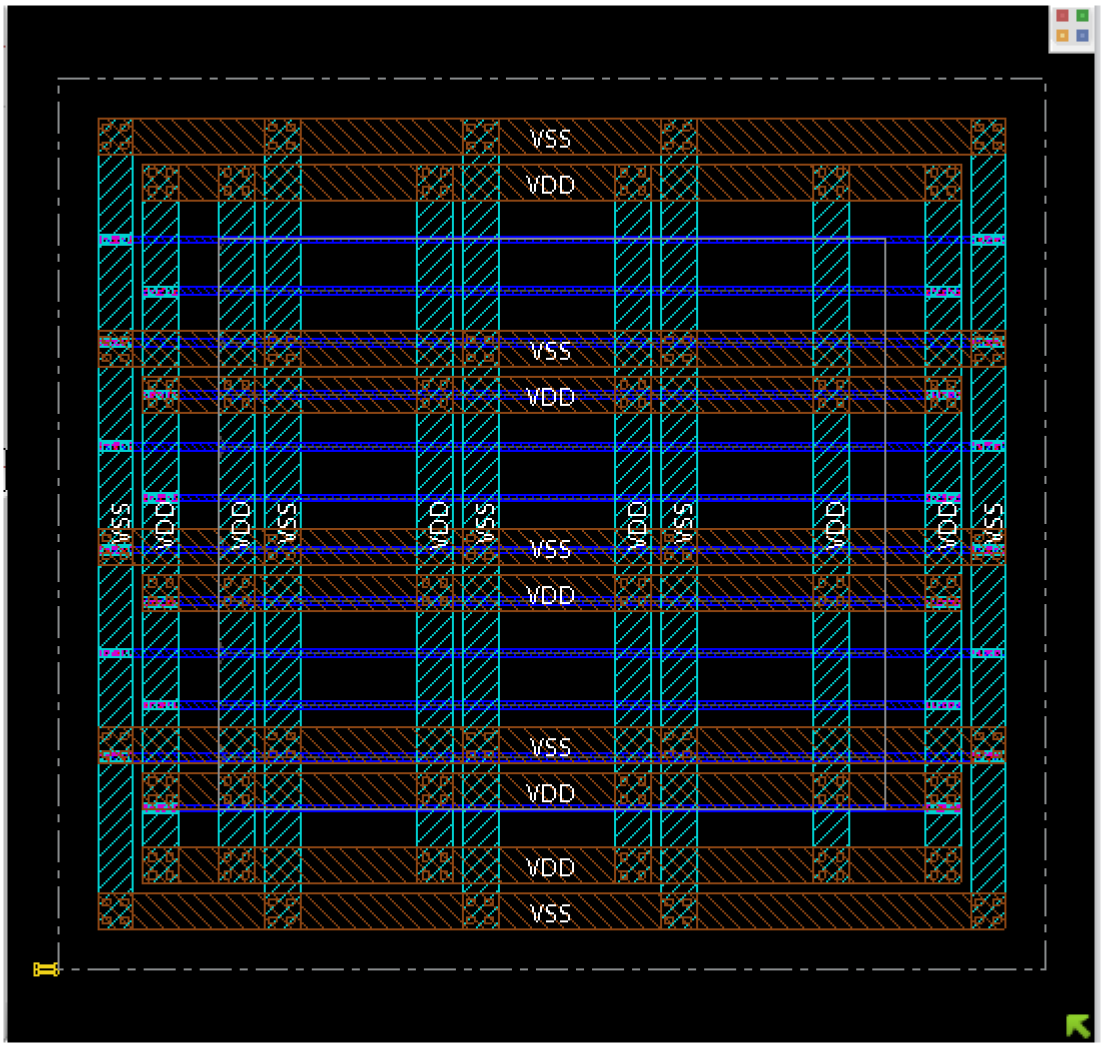


On adding Power Stripes, The Power mesh setup is complete as shown. However, There are no Vias that could connect Metal 9 or Metal 8 directly with Metal 1 [VDD or VSS of Standard Cells are generally made up of Metal 1].

The connection between the Highest and Lowest Metals is done through Stacking of Vias done using “Special Route”.

To perform Special Route, **Select Route → Special Route → Add Nets → OK.**

After the Special Route is complete, all the Standard Cell Rows turn to the Color coded for Metal 1 as shown below.

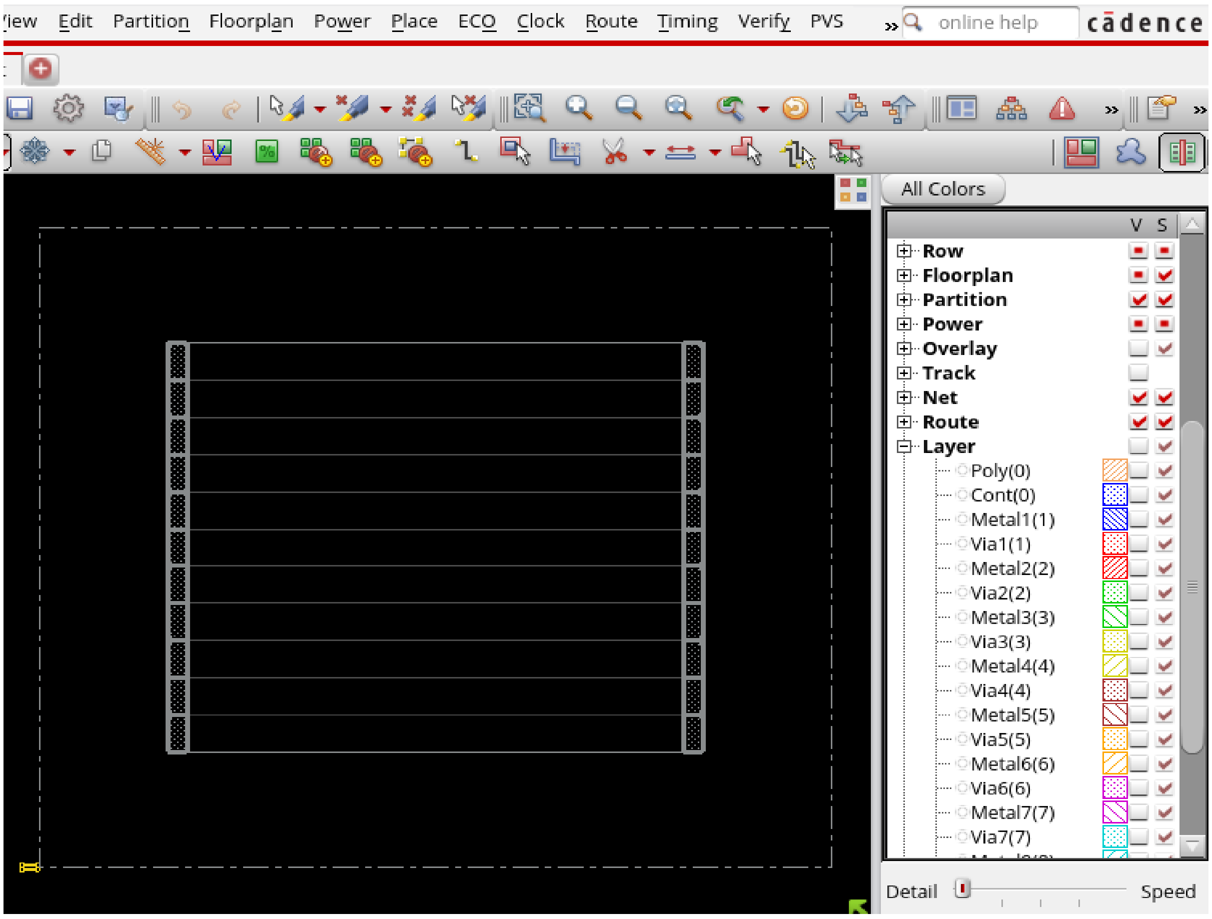


The complete Power Planning process makes sure Every Standard Cell receives enough power to operate smoothly.

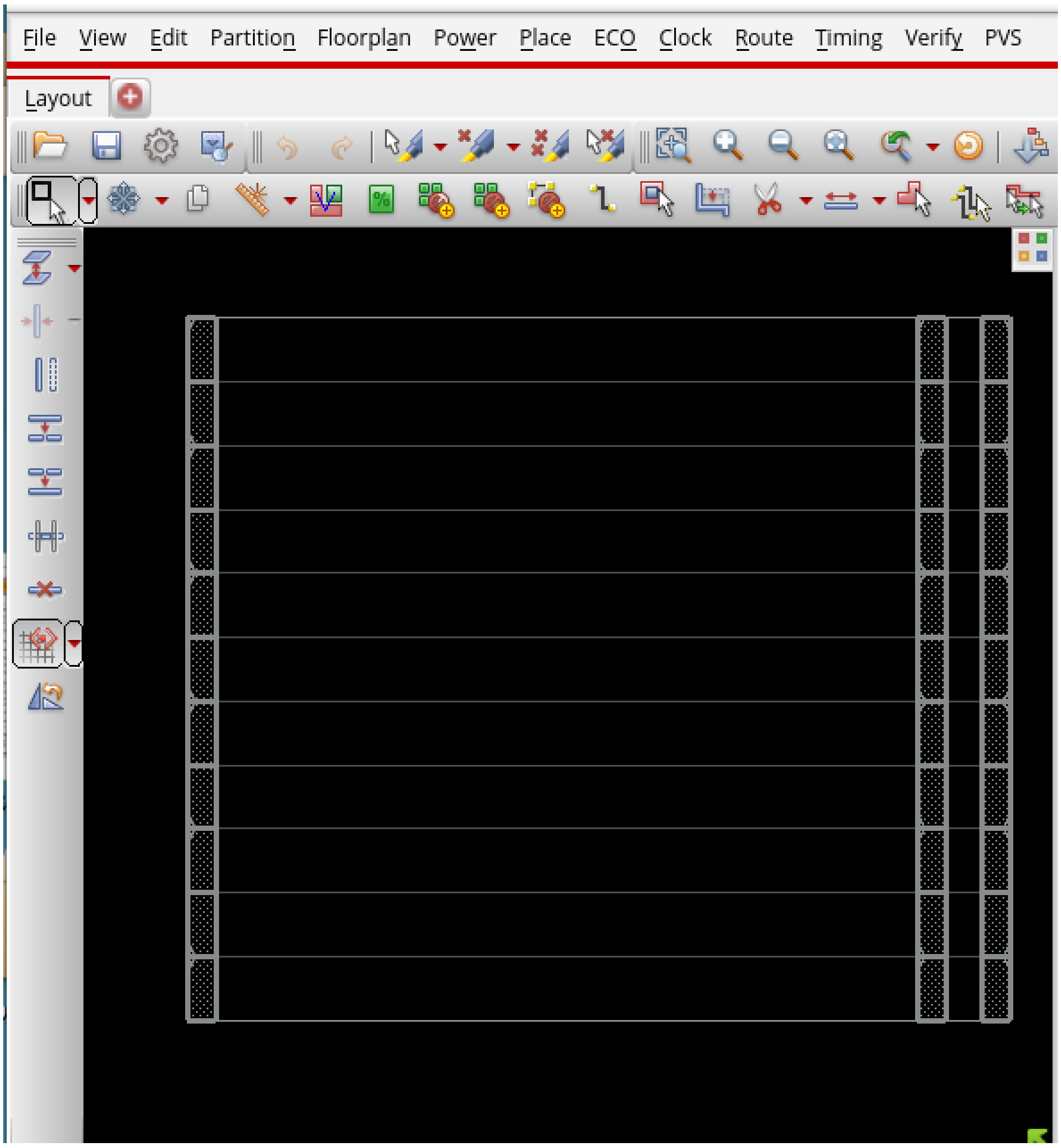
## Module 4.4.1 : Pre - Placement

→ After Power Planning, a few Physical Cells are added namely, **End Caps and Well Taps**.
→ **End Caps :** They are Physical Cells which are added to the Left and Right Core Boundaries acting as blockages to avoid Standard Cells from moving out of boundary.
→ **Well Taps :** They act like Shunt Resistance to avoid Latch Up effects.

1. To add End Caps, Select **Place → Physical Cell → Add End Caps** and "Select" the FILL's from the available list. Higher Fills have Higher Widths. As shown Below, The End Caps are added below your Power Mesh.

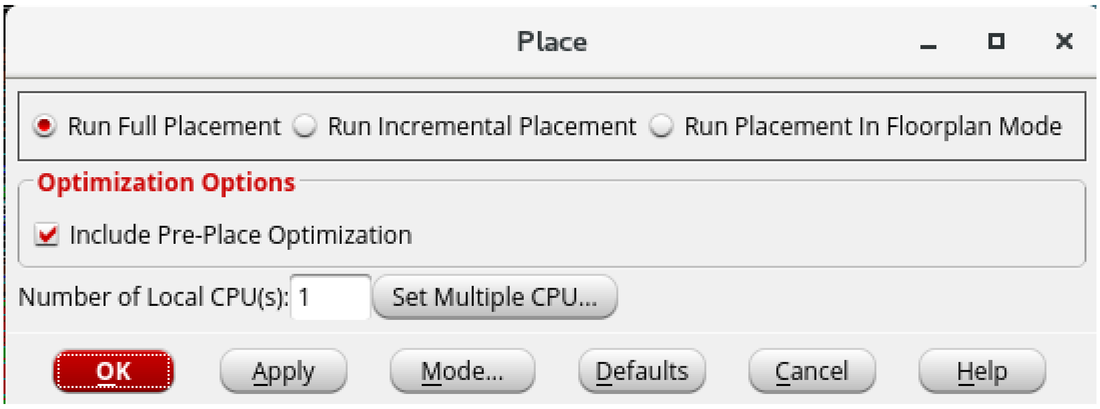


2. To add Well Taps, Select **Place → Physical Cell → Add Well Tap → Select → FillX** [X → Strength of Fill = 1,2,4 etc] **→ Distance Interval** [Could be given in range of 30-45u] **→ OK**



## Module 4.5 : Placement

1. The Placement stage deals with Placing of Standard Cells as well as Pins.
2. Select **Place → Place Standard Cell → Run Full Placement → Mode → Enable 'Place I/O Pins' → OK → OK .**

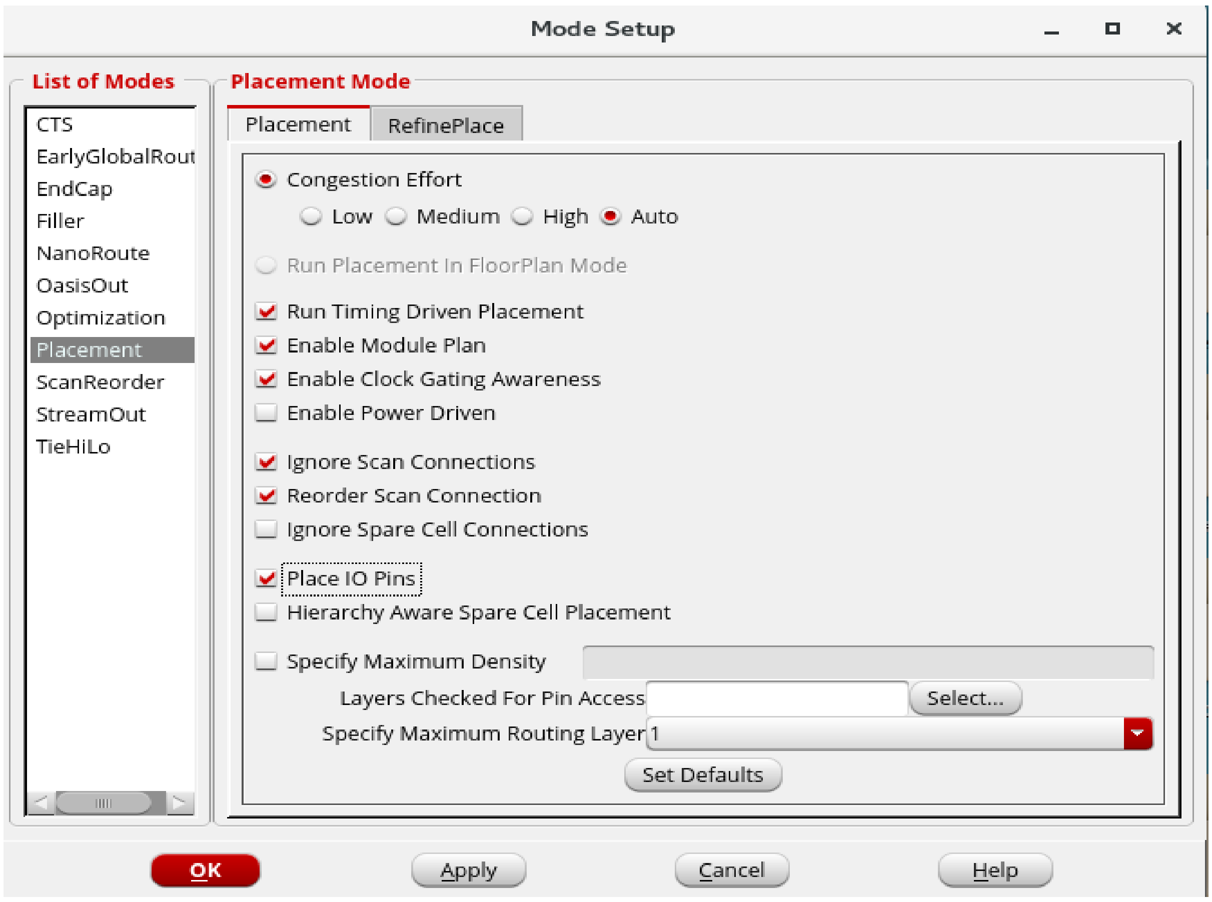

Place

[✓ Run Full Placement] Run Incremental Placement Run Placement In Floorplan Mode

Optimization Options

[✓ Include Pre-Place Optimization]

Number of Local CPU(s): 1 Set Multiple CPU...


Mode Setup

List of Modes

- CTS
- EarlyGlobalRout
- EndCap
- Filler
- NanoRoute
- OasisOut
- Optimization
- Placement
- ScanReorder
- StreamOut
- TieHiLo

Placement Mode

Placement | RefinePlace

[✓ Congestion Effort]

Low Medium High [✓ Auto]

Run Placement In FloorPlan Mode

[✓ Run Timing Driven Placement]
[✓ Enable Module Plan]
[✓ Enable Clock Gating Awareness]
Enable Power Driven

[✓ Ignore Scan Connections]
[✓ Reorder Scan Connection]
Ignore Spare Cell Connections

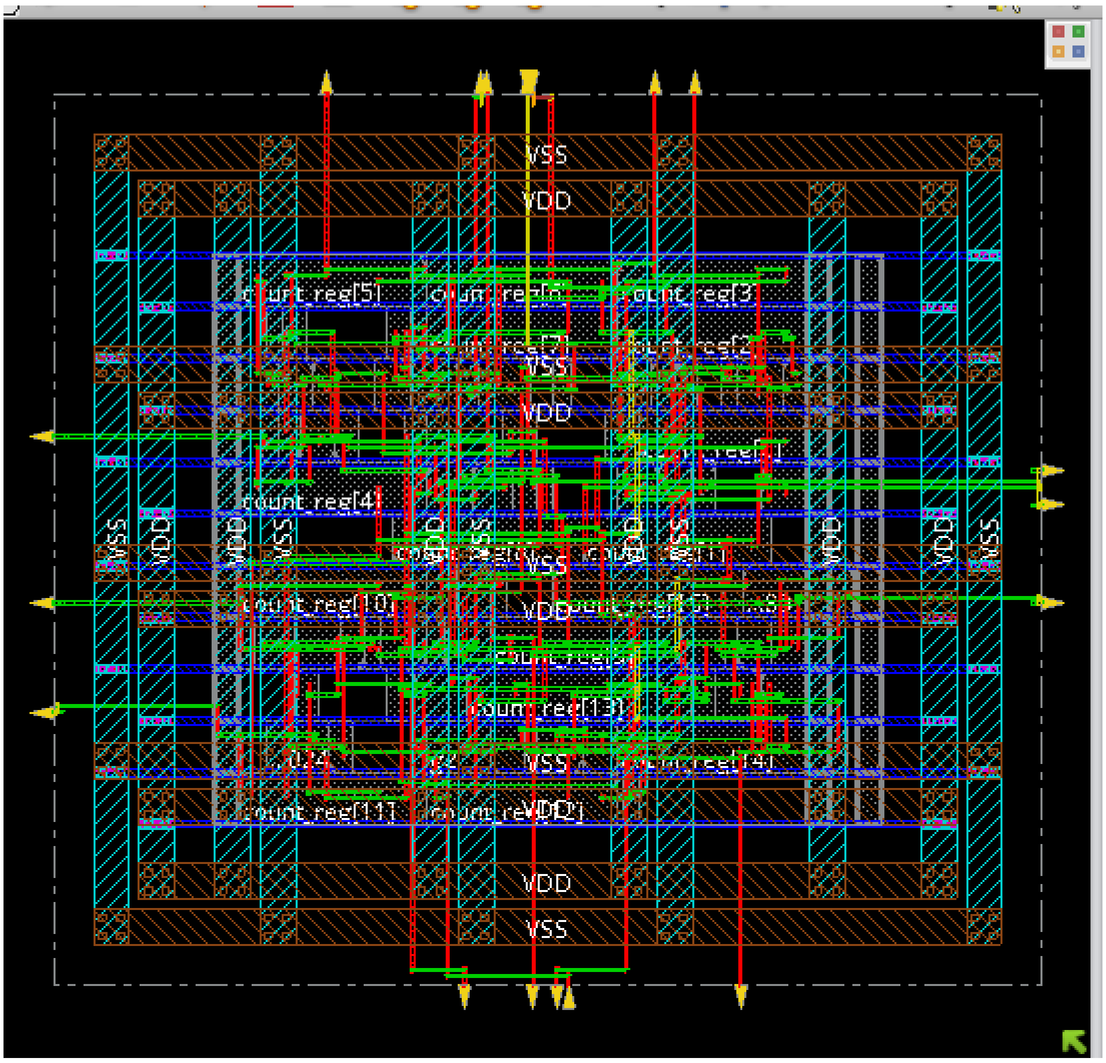
[✓ Place IO Pins]
Hierarchy Aware Spare Cell Placement

Specify Maximum Density

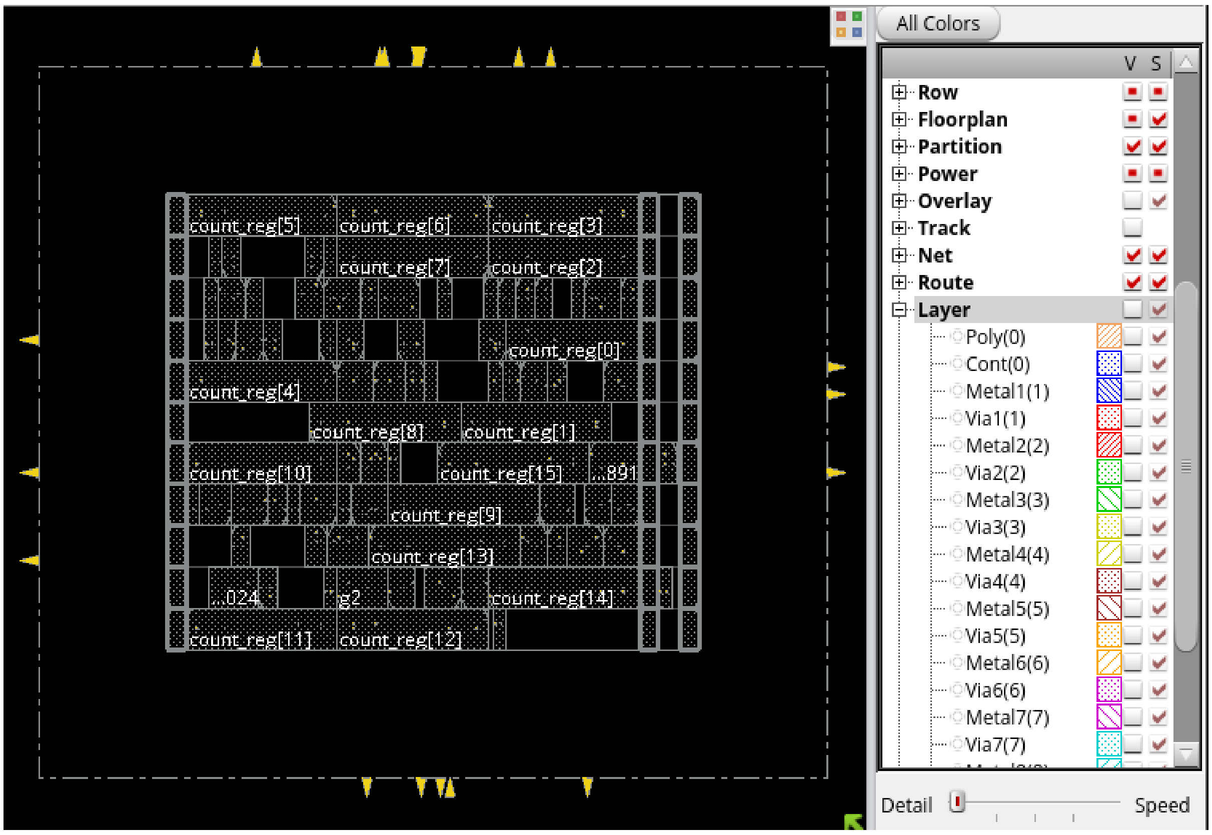
Layers Checked For Pin Access

Specify Maximum Routing Layer 1

All the Standard Cells and Pins are placed as per the communication between them, i.e., Two communicating Cells are placed as close as possible so that shorter Net lengths can be used for connections as Shorter Net Lengths enable Better Timing Results.



**Placed Design**



**Standard Cells Placed**

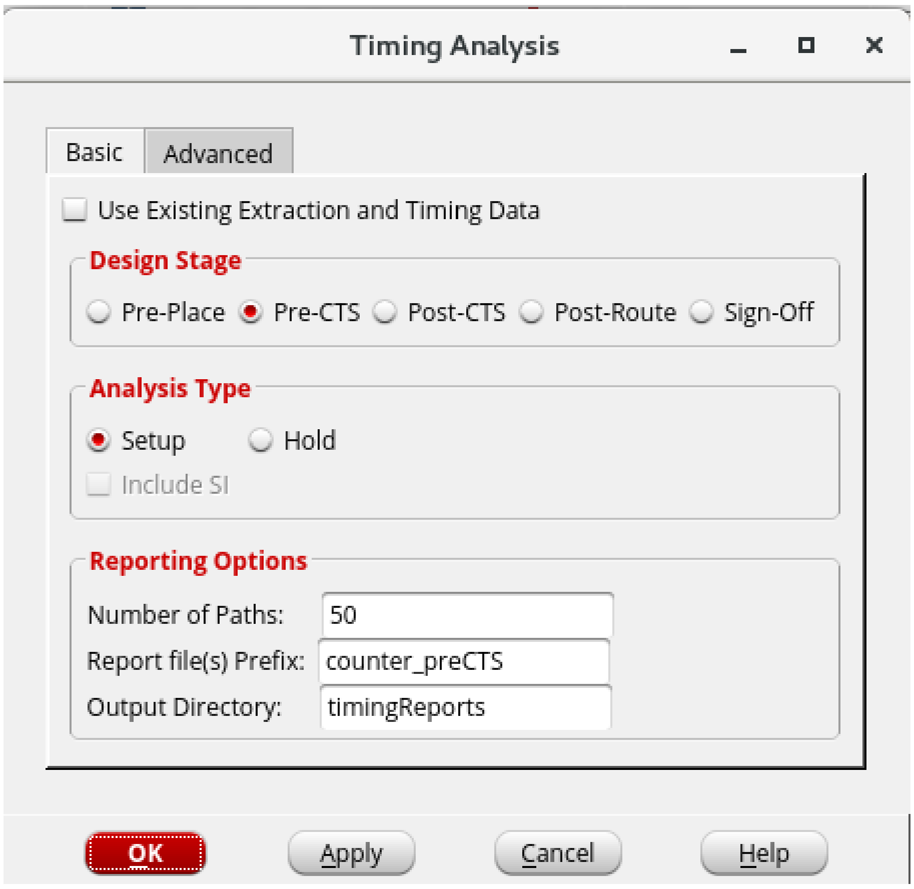
You can toggle the Layer Visibility from the list on the Right.

**Report Generation and Optimization :**

**→ Timing Report :**

To generate Timing Report, **Timing → Report Timing → Design Stage – PreCTS → Analysis Type – Setup → OK**

The Timing report Summary can be seen on the Terminal.



[ ] Use Existing Extraction and Timing Data

Design Stage: Pre-Place [✓ Pre-CTS] Post-CTS Post-Route Sign-Off

Analysis Type: [✓ Setup] Hold

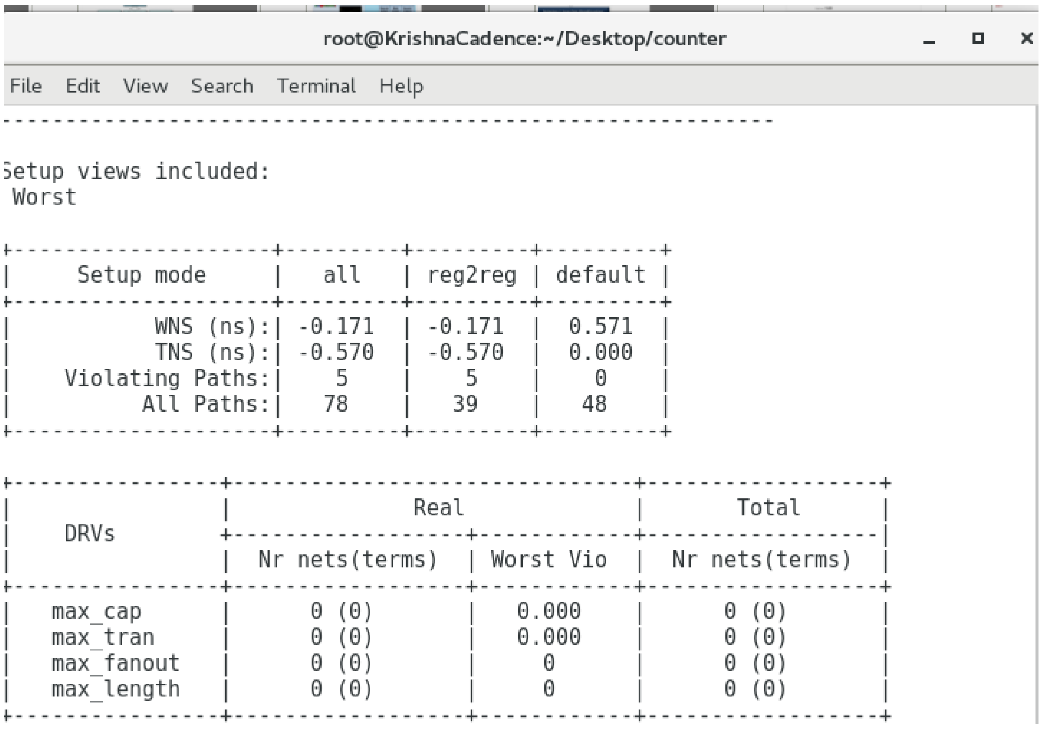
[ ] Include SI

Reporting Options

Number of Paths: 50

Report file(s) Prefix: counter_preCTS

Output Directory: timingReports

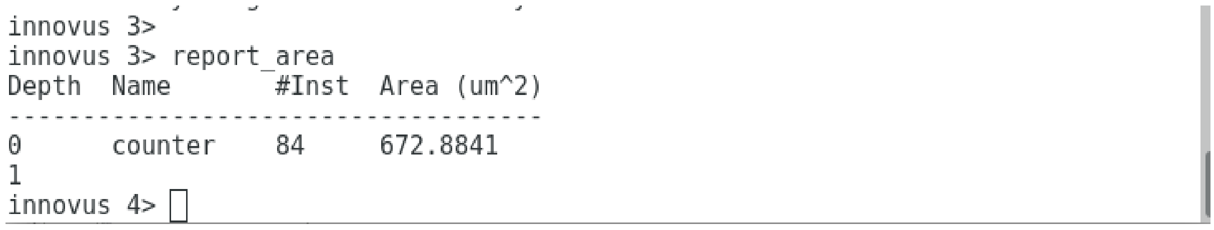

root@KrishnaCadence:~/Desktop/counter
File Edit View Search Terminal Help

```
------------------------------------------------------------

Setup views included:
 Worst

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):| -0.171  | -0.171  |  0.571  |
|           TNS (ns):| -0.570  | -0.570  |  0.000  |
|    Violating Paths:|    5    |    5    |    0    |
|          All Paths:|   78    |   39    |   48    |
+--------------------+---------+---------+---------+

+----------------+-------------------------------+------------------+
|                |              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    |     0 (0)        |
|   max_fanout   |    0 (0)         |     0      |     0 (0)        |
|   max_length   |    0 (0)         |     0      |     0 (0)        |
+----------------+------------------+------------+------------------+
```

**→ Area Report :**

To generate Area Report, Switch to the Terminal and type the command ,

**report_area** to see the Cell Count and Area Occupied.

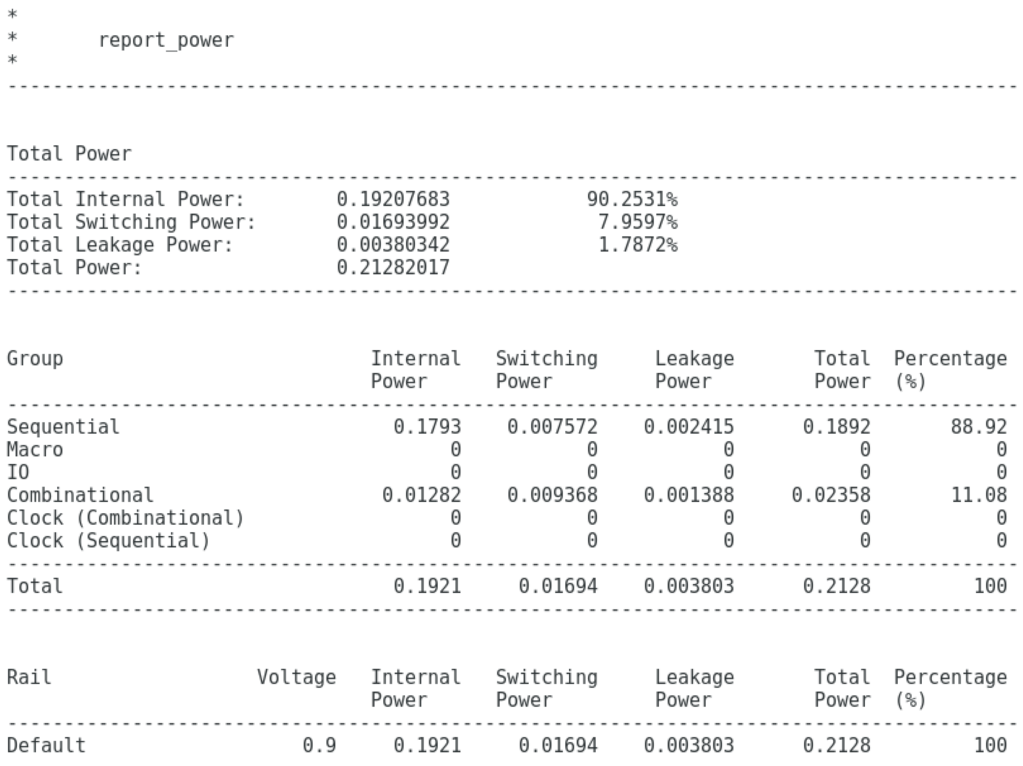


```
innovus 3>
innovus 3> report_area
Depth  Name       #Inst  Area (um^2)
------------------------------------
0      counter    84     672.8841
1
innovus 4>
```

**→ Power report :**

To generate Power Report, In the Terminal type the command

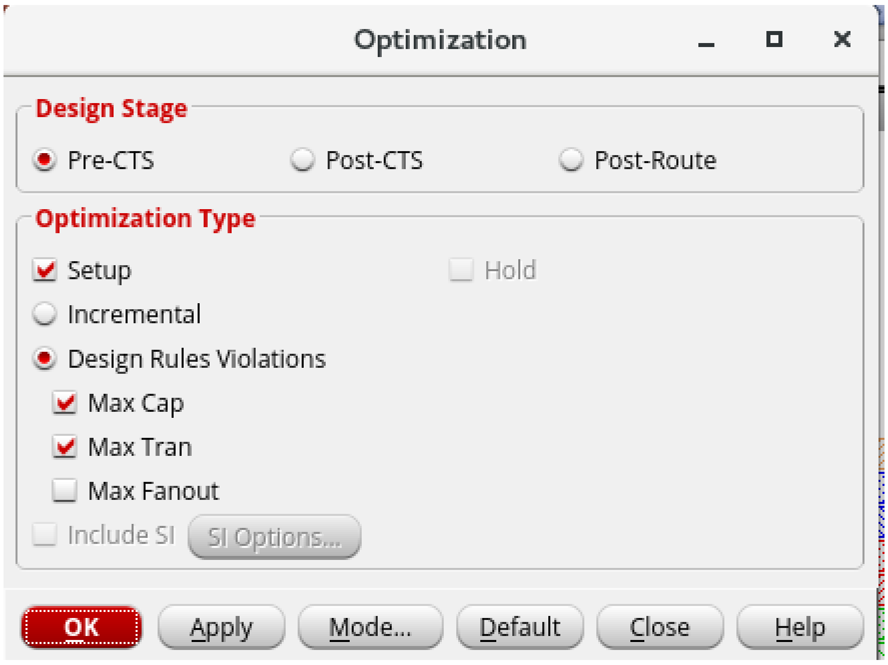
**report_power** to see the Power Consumption numbers.



```
*
*       report_power
*
-----------------------------------------------------------------------------------------

Total Power
-----------------------------------------------------------------------------------------
Total Internal Power:        0.19207683           90.2531%
Total Switching Power:       0.01693992            7.9597%
Total Leakage Power:         0.00380342            1.7872%
Total Power:                 0.21282017
-----------------------------------------------------------------------------------------

Group                           Internal   Switching     Leakage      Total  Percentage
                                Power      Power         Power        Power  (%)
-----------------------------------------------------------------------------------------
Sequential                        0.1793    0.007572    0.002415     0.1892       88.92
Macro                                  0           0           0          0           0
IO                                     0           0           0          0           0
Combinational                    0.01282    0.009368    0.001388    0.02358       11.08
Clock (Combinational)                  0           0           0          0           0
Clock (Sequential)                     0           0           0          0           0
-----------------------------------------------------------------------------------------
Total                             0.1921     0.01694    0.003803     0.2128         100
-----------------------------------------------------------------------------------------

Rail                  Voltage   Internal   Switching     Leakage      Total  Percentage
                                Power      Power         Power        Power  (%)
-----------------------------------------------------------------------------------------
Default                   0.9     0.1921     0.01694    0.003803     0.2128         100
```

## Design Optimization :

To optimize the Design, Select **ECO → Optimize Design → Design Stage [PreCTS] → Optimization Type – Setup → OK**



**Design Stage**

[✓ Pre-CTS] Post-CTS Post-Route

**Optimization Type**

[✓ Setup] Hold

Incremental

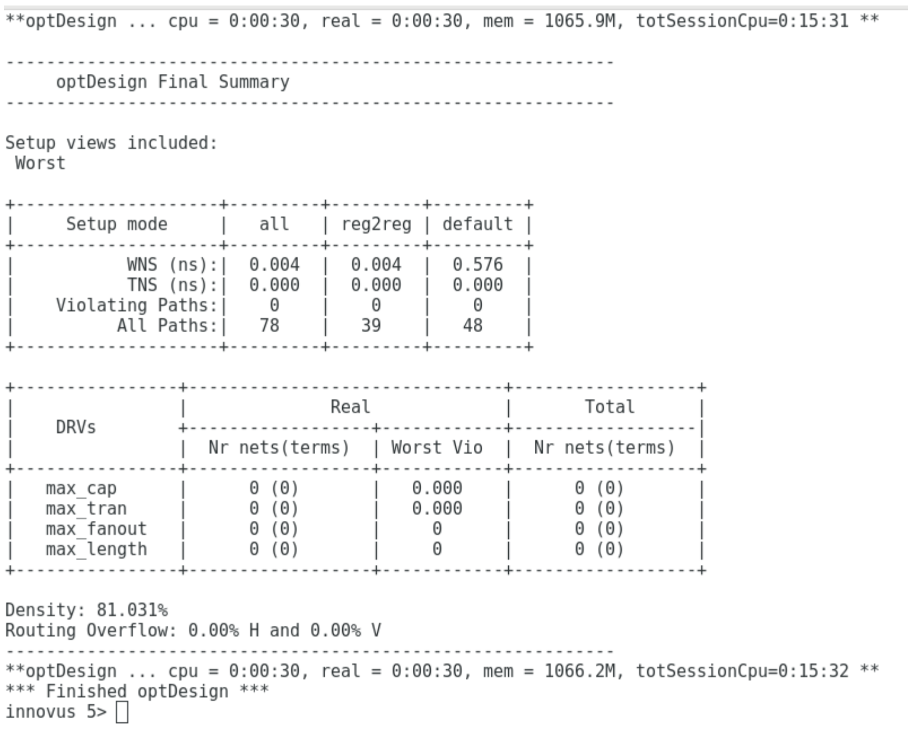
[✓ Design Rules Violations]

[✓ Max Cap]

[✓ Max Tran]

Max Fanout

Include SI



```
**optDesign ... cpu = 0:00:30, real = 0:00:30, mem = 1065.9M, totSessionCpu=0:15:31 **

------------------------------------------------------------
     optDesign Final Summary
------------------------------------------------------------

Setup views included:
 Worst

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.004  |  0.004  |  0.576  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|   78    |   39    |   48    |
+--------------------+---------+---------+---------+

+----------------+-------------------------------+------------------+
|                |              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    |     0 (0)        |
|   max_fanout   |     0 (0)        |     0      |     0 (0)        |
|   max_length   |     0 (0)        |     0      |     0 (0)        |
+----------------+------------------+------------+------------------+

Density: 81.031%
Routing Overflow: 0.00% H and 0.00% V
------------------------------------------------------------
**optDesign ... cpu = 0:00:30, real = 0:00:30, mem = 1066.2M, totSessionCpu=0:15:32 **
*** Finished optDesign ***
innovus 5> 
```

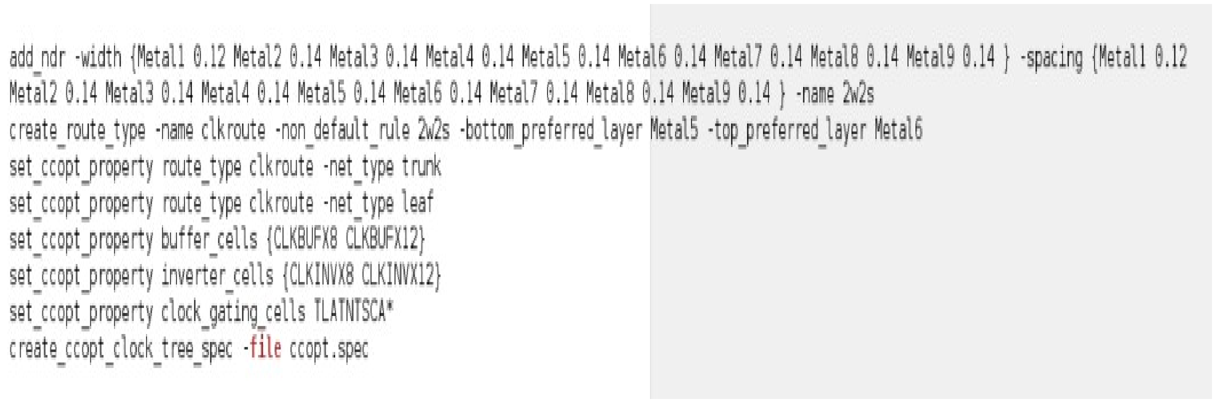
This step Optimizes your design in terms of Timing, Area and Power.

You can Generate Timing, Area, Power in similar way as above report Post – Optimization to compare the Reports.

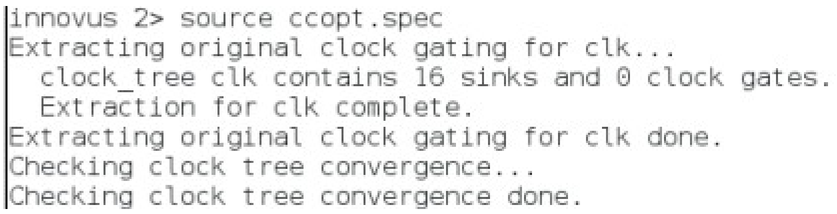
**Module 4.6 : Clock Tree Synthesis**

The CTS Stage is meant to build a Clock Distribution Network such that every Register (Flip Flop) acquires Clock at the same time (Atleast Approximately) to keep them in proper communication.

A Script can be used to Build the Clock Tree as follows :

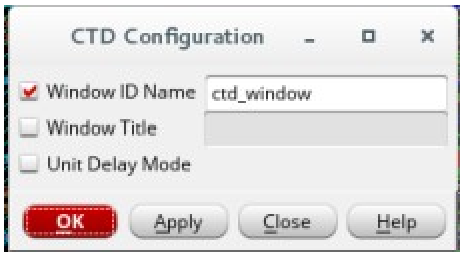


```
add_ndr -width {Metal1 0.12 Metal2 0.14 Metal3 0.14 Metal4 0.14 Metal5 0.14 Metal6 0.14 Metal7 0.14 Metal8 0.14 Metal9 0.14 } -spacing {Metal1 0.12
Metal2 0.14 Metal3 0.14 Metal4 0.14 Metal5 0.14 Metal6 0.14 Metal7 0.14 Metal8 0.14 Metal9 0.14 } -name 2w2s
create_route_type -name clkroute -non_default_rule 2w2s -bottom_preferred_layer Metal5 -top_preferred_layer Metal6
set_ccopt_property route_type clkroute -net_type trunk
set_ccopt_property route_type clkroute -net_type leaf
set_ccopt_property buffer_cells {CLKBUFX8 CLKBUFX12}
set_ccopt_property inverter_cells {CLKINVX8 CLKINVX12}
set_ccopt_property clock_gating_cells TLATNTSCA*
create_ccopt_clock_tree_spec -file ccopt.spec
```



```
innovus 2> source ccopt.spec
Extracting original clock gating for clk...
  clock_tree clk contains 16 sinks and 0 clock gates.
  Extraction for clk complete.
Extracting original clock gating for clk done.
Checking clock tree convergence...
Checking clock tree convergence done.
```

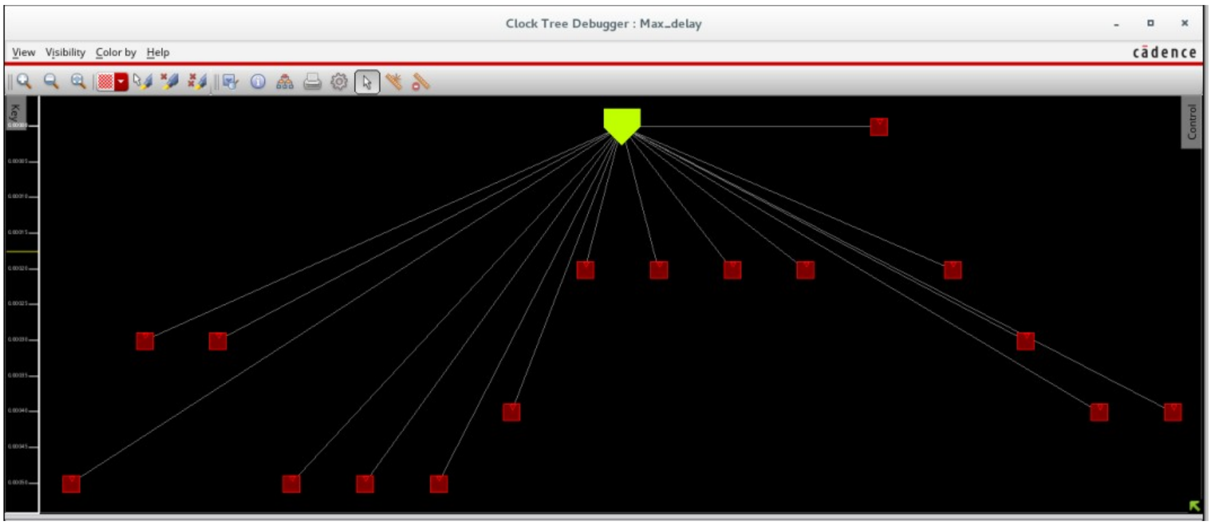
Source the Script as shown in the above snapshot through the Terminal and then Select **Clock → CCOpt Clock Tree Debugger → OK** to build and view clock tree.


CTD Configuration

[✓] Window ID Name ctd_window

Window Title

Unit Delay Mode



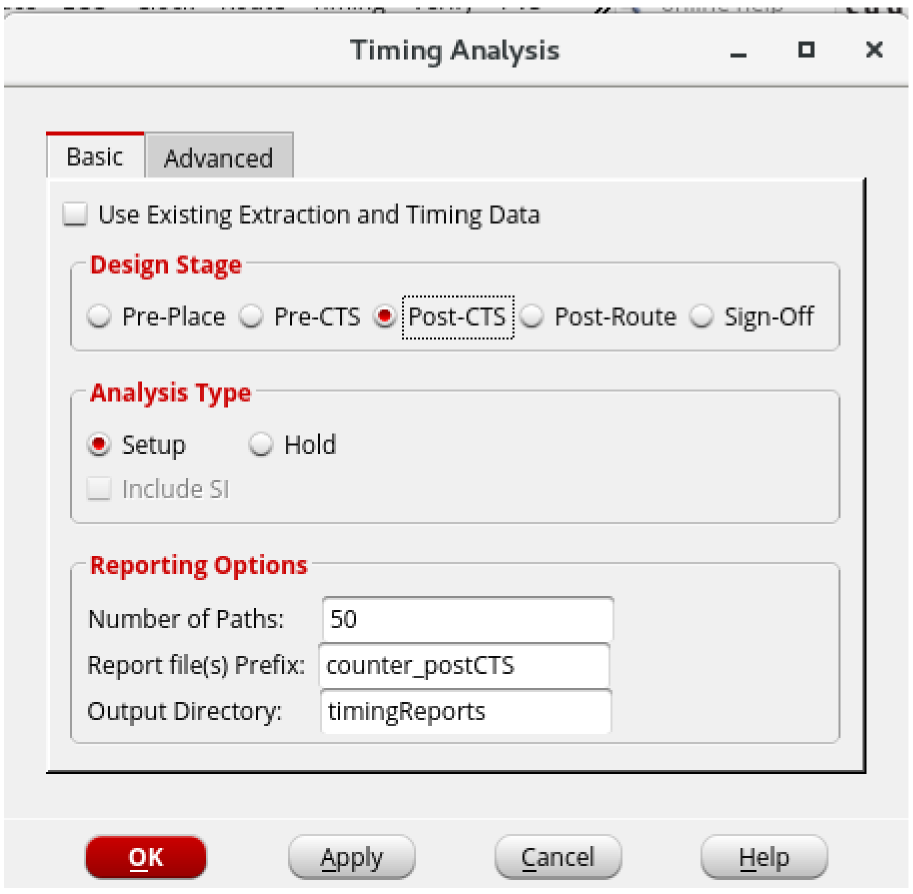
The Red Boxes are the Clock Pins of various Flip Flops in the Design while Yellow Pentagon on the top represents Clock Source.

The Clock Tree is built with Clock Buffers and Clock Inverters added to boost up the Clock Signal.

**Report Generation and Design Optimization :**

CTS Stage adds real clock into the Design and hence "Hold" Analysis also becomes prominent. Hence, **Optimizations can be done for both Setup & Hold, Timing Reports are to be Generated for Setup and Hold Individually**.

**Setup Timing Analysis :**


Timing Analysis

Basic | Advanced

Use Existing Extraction and Timing Data [ ]

Design Stage: Pre-Place [✓ Post-CTS] Pre-CTS Post-Route Sign-Off

Analysis Type: [✓ Setup] Hold

Include SI [ ]

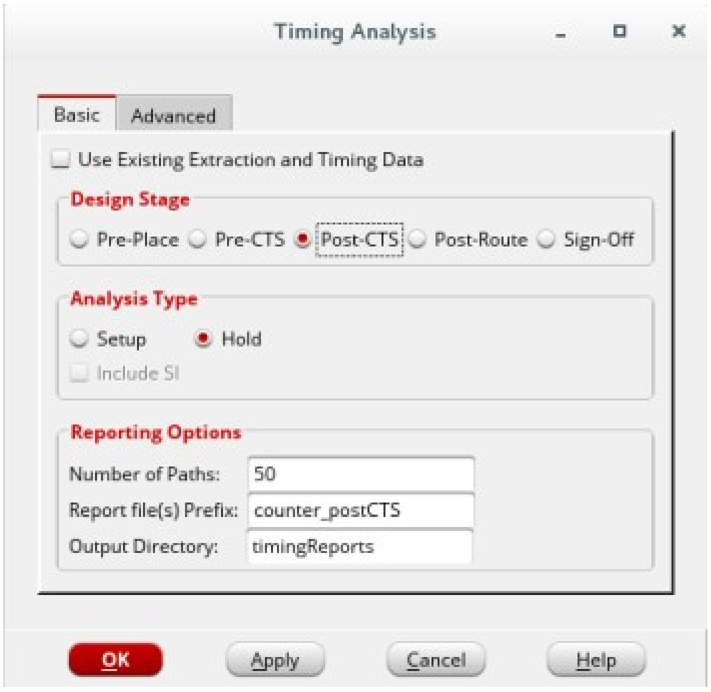
Reporting Options

Number of Paths: 50

Report file(s) Prefix: counter_postCTS

Output Directory: timingReports

**Hold Timing Analysis :**



Use Existing Extraction and Timing Data [ ]

Design Stage: Pre-Place Pre-CTS [✓ Post-CTS] Post-Route Sign-Off

Analysis Type: Setup [✓ Hold]

Include SI [ ]

Reporting Options

Number of Paths: 50

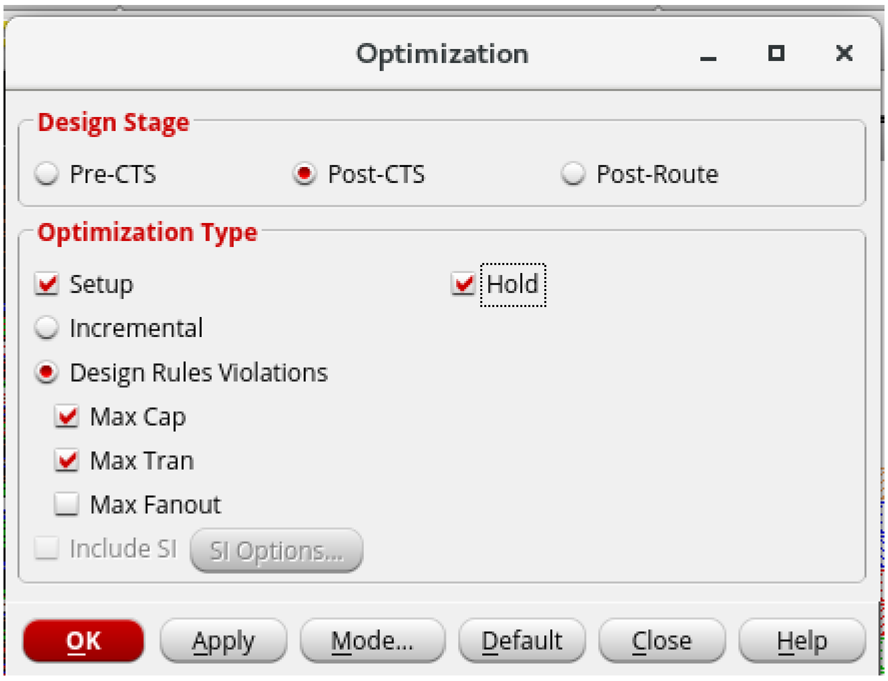
Report file(s) Prefix: counter_postCTS

Output Directory: timingReports

For Area and Power Report Generation,

**report_area & report_power** commands can be used.

**Design Optimizations :**



Design Stage: Pre-CTS [✓ Post-CTS] Post-Route

Optimization Type

[✓] Setup [✓] Hold

Incremental [✓ Design Rules Violations]

[✓] Max Cap

[✓] Max Tran

[ ] Max Fanout

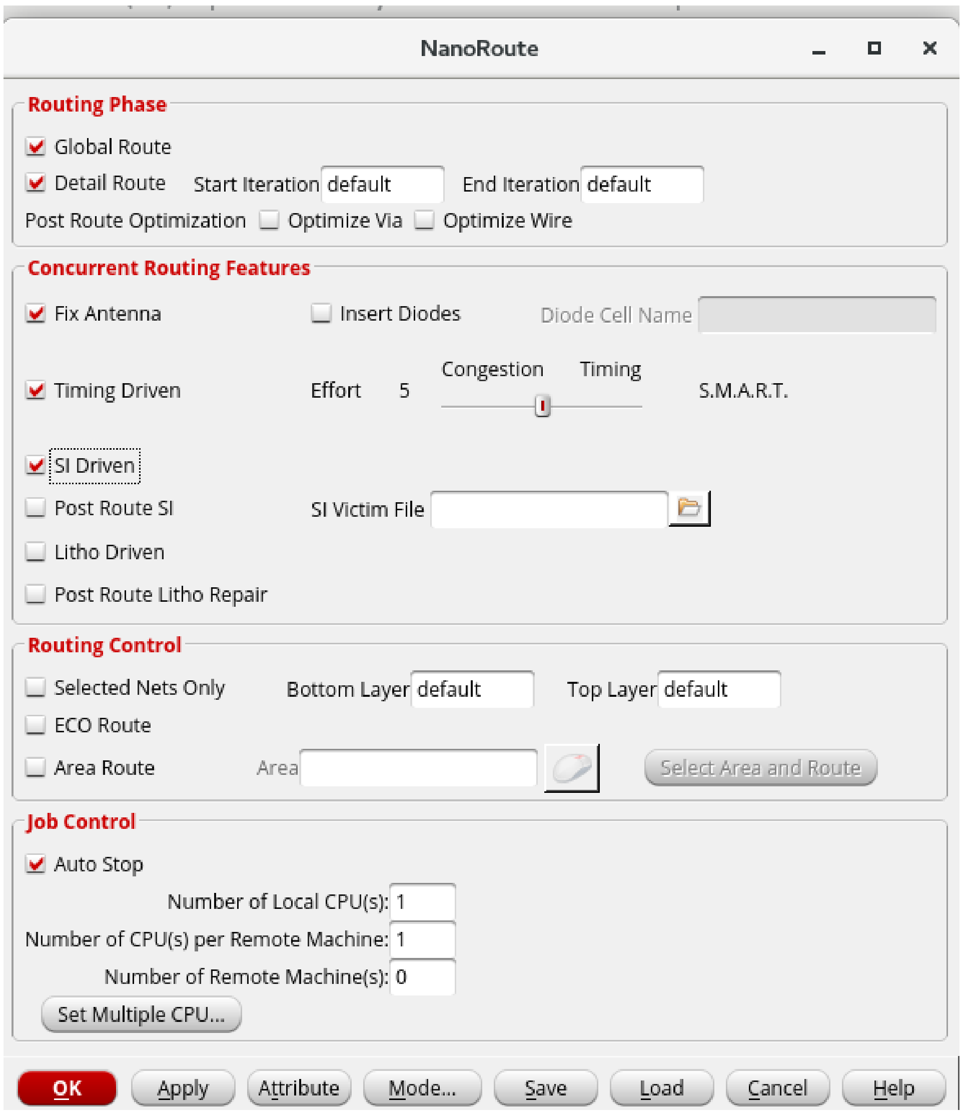
[ ] Include SI

**Module 4.7 : Routing**

→ All the net connections shown in the GUI till CTS are only based on the Logical connectivity.

→ These connections are to be replaced with real Metals avoiding Opens, Shorts, Signal Integrity [Cross Talks], Antenna Violations etc.

To run Routing, Select **Route → Nano Route → Route** and enable **Timing Driven** and **SI Driven** for Design Physical Efficiency and Reliability.



NanoRoute

**Routing Phase**

- ✓ Global Route
- ✓ Detail Route — Start Iteration: default — End Iteration: default
- Post Route Optimization: ☐ Optimize Via ☐ Optimize Wire

**Concurrent Routing Features**

- ✓ Fix Antenna — ☐ Insert Diodes — Diode Cell Name:
- ✓ Timing Driven — Effort 5 (Congestion — Timing) — S.M.A.R.T.
- ✓ SI Driven
- ☐ Post Route SI — SI Victim File:
- ☐ Litho Driven
- ☐ Post Route Litho Repair

**Routing Control**

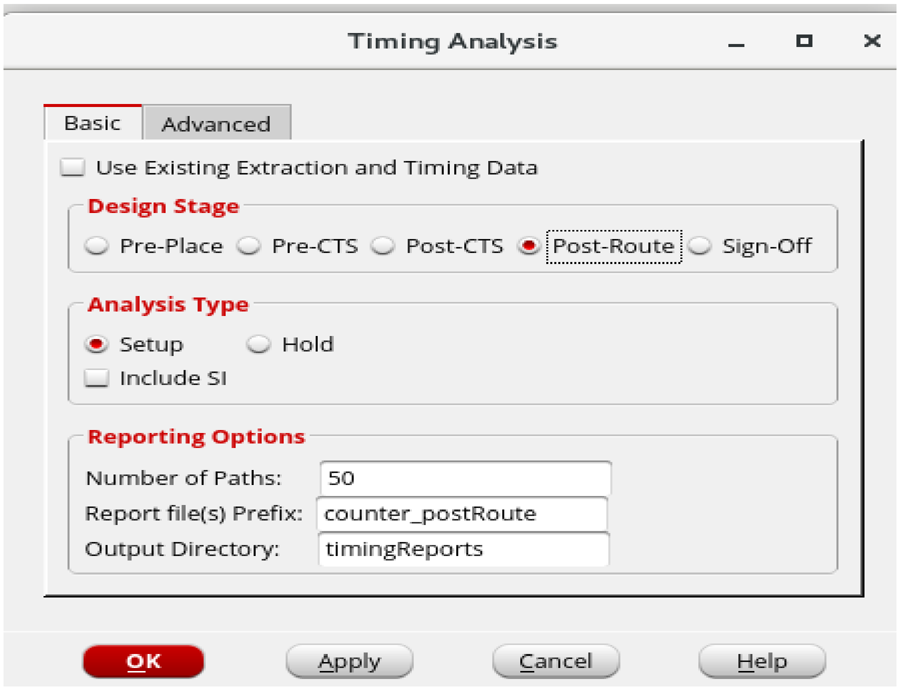
- ☐ Selected Nets Only — Bottom Layer: default — Top Layer: default
- ☐ ECO Route
- ☐ Area Route — Area:

**Job Control**

- ✓ Auto Stop
- Number of Local CPU(s): 1
- Number of CPU(s) per Remote Machine: 1
- Number of Remote Machine(s): 0

## Report Generation and Design Optimization :

### Setup Report :


Timing Analysis

Basic | Advanced

Use Existing Extraction and Timing Data [ ]

Design Stage: Pre-Place Pre-CTS Post-CTS [✓ Post-Route] Sign-Off

Analysis Type: [✓ Setup] Hold

Include SI [ ]

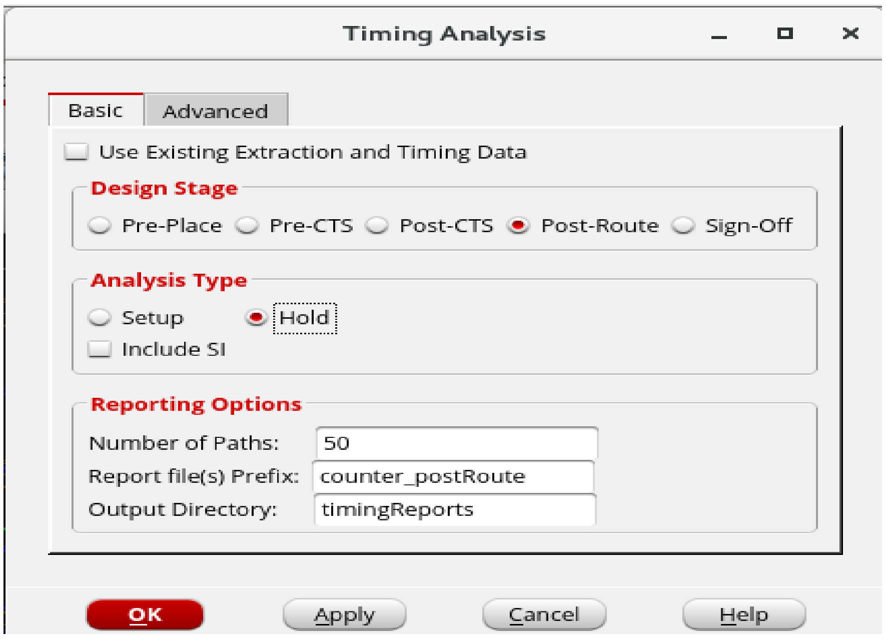
Reporting Options

Number of Paths: 50

Report file(s) Prefix: counter_postRoute

Output Directory: timingReports

### Hold Report :


Timing Analysis

Basic | Advanced

Use Existing Extraction and Timing Data [ ]

Design Stage: Pre-Place Pre-CTS Post-CTS [✓ Post-Route] Sign-Off

Analysis Type: Setup [✓ Hold]

Include SI [ ]

Reporting Options

Number of Paths: 50

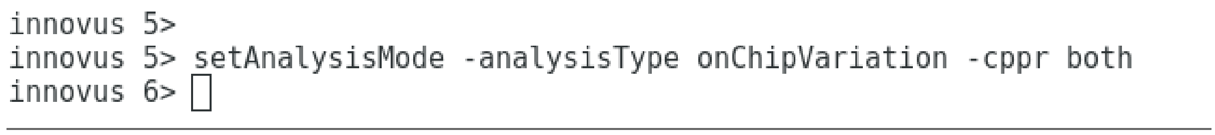
Report file(s) Prefix: counter_postRoute

Output Directory: timingReports

**Area and Power Reports :**

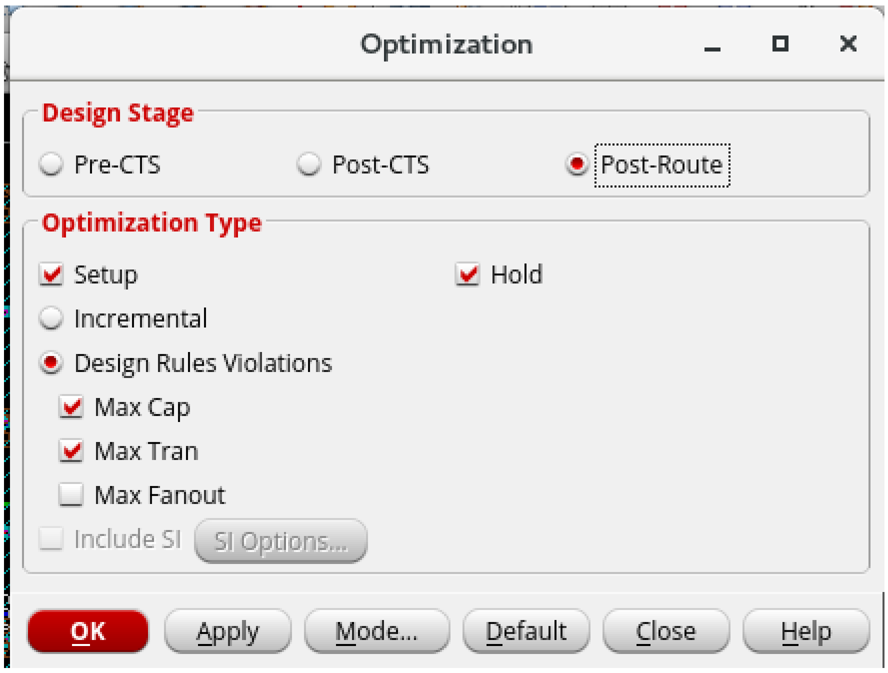
Use the commands **report_area** and **report_power** for Area and Power Reports respectively.

**Design Optimization :**



```
innovus 5>
innovus 5> setAnalysisMode -analysisType onChipVariation -cppr both
innovus 6> 
```

Enter the above shown command in the Terminal in order to run the Design Optimization first Post-Route.



Optimization

Design Stage

Pre-CTS  Post-CTS  [✓ Post-Route]

Optimization Type

[✓ Setup]  [✓ Hold]

Incremental

[✓ Design Rules Violations]

[✓ Max Cap]

[✓ Max Tran]

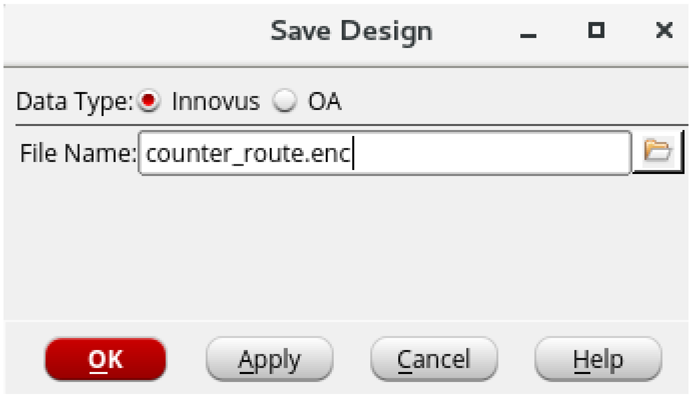
Max Fanout

Include SI

The Report generation is same as shown prior to Design Optimization.

**Saving Database :**

1. **Saving Design => File → Save Design → Data Type : Innovus → <DesignName>.enc → OK**

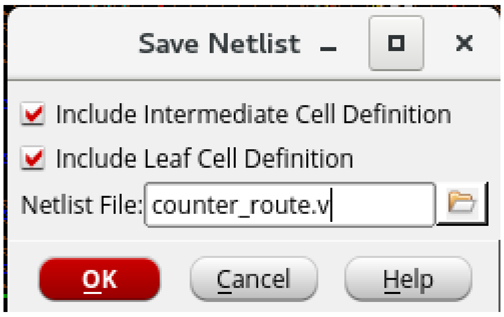


Save Design

Data Type: [✓ Innovus] OA

File Name: counter_route.enc

2. **Saving Netlist => File → Save → Netlist → <NetlistName>.v → OK**



Save Netlist

[✓] Include Intermediate Cell Definition

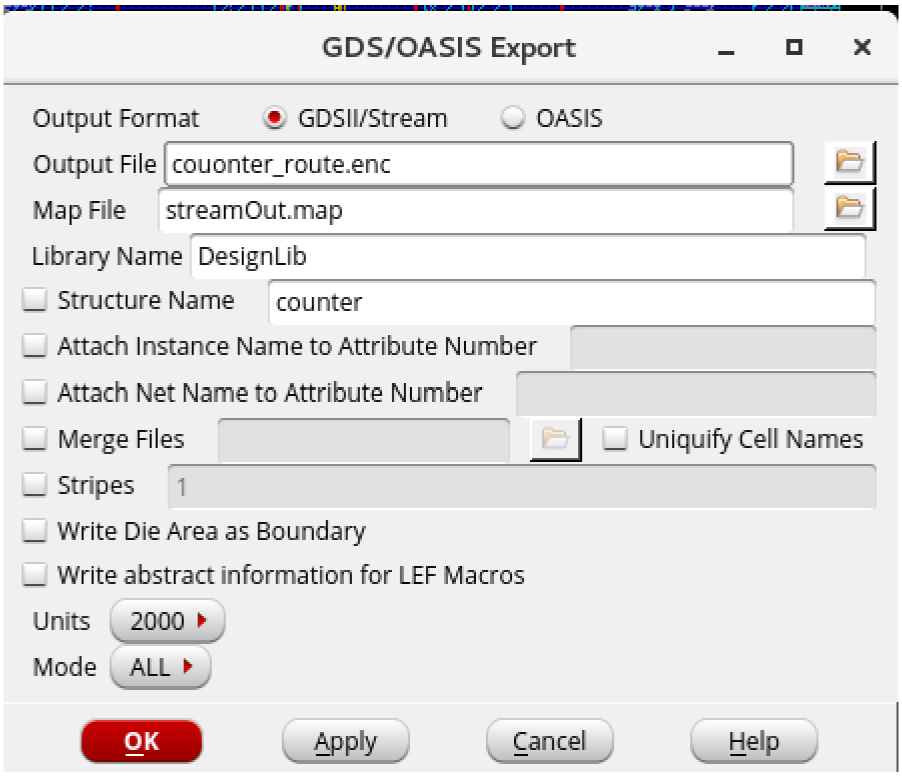
[✓] Include Leaf Cell Definition

Netlist File: counter_route.v

It is recommended to save Netlist and Design at every stage.

To restore a Design Data Base, type **source <DesignName>.enc** in the terminal.

3. **Saving GDS => File → Save → GDS/OASIS → <FileName>.gds → OK**



GDS/OASIS Export

Output Format [✓ GDSII/Stream] OASIS

Output File couonter_route.enc

Map File streamOut.map

Library Name DesignLib

Structure Name counter

Attach Instance Name to Attribute Number

Attach Net Name to Attribute Number

Merge Files

Uniquify Cell Names

Stripes 1

Write Die Area as Boundary

Write abstract information for LEF Macros

Units 2000

Mode ALL

The GDS File is a Binary Format File which is not Readable and is fed to the Fabrication unit with data of various layers used depicted in terms of Geometrical Shapes.