

ABSTRACT:

The 8-bit microprocessor design demonstrates the fundamental principles of computer architecture and provides a platform for understanding the intricacies of processor operation. By striking a balance between simplicity and functionality, it serves as a valuable educational tool for studying microprocessors and computer systems. Furthermore, it offers potential for integration into various applications where cost-effective computing solutions are required.

The microprocessor is implemented using a hardware description language and simulated using appropriate tools. The design is optimized for area efficiency and low power consumption, making it suitable for embedded systems and low-cost applications. Performance evaluation includes measurements of clock frequency, execution time for various instructions, and overall system throughput.

INTRODUCTION

In the realm of integrated circuit design, physical design plays a vital role in transforming the conceptual architecture of a microprocessor into a tangible and functional electronic device. This introduction provides an overview of the physical design considerations specific to 8-bit microprocessors, including transistor-level implementation, layout design, and the optimization techniques employed to achieve performance, power efficiency, and manufacturability.

The physical design of an 8-bit microprocessor involves translating the logical and architectural specifications of the microprocessor into a physical representation that can be fabricated on a silicon wafer. At the heart of this process lies the implementation of individual transistors, which serve as the building blocks for the various functional units within the microprocessor.

Transistor-level implementation requires careful consideration of the fabrication process, transistor types (such as NMOS or CMOS), and sizing. The design must take into account parameters such as speed, power consumption, and area utilization. While the fundamental operations of the microprocessor—such as arithmetic and logical computations—are defined by its architecture, the physical design influences the actual performance and characteristics of these operations.

Layout design is another crucial aspect of physical design. It involves determining the placement and interconnection of various components, including transistors, registers, memory cells, and other functional units, within the microprocessor. Efficient layout design is essential to minimize delays in signal propagation, reduce power consumption, and ensure reliable operation.

System Model

A system model for an 8-bit microprocessor in physical design encompasses the various components and interactions involved in the physical implementation of the microprocessor. It provides a holistic view of the system, including the microarchitecture, interconnects, memory subsystem, and peripheral interfaces. Here is a high-level overview of the key components typically considered in the system model:

Microarchitecture: The microarchitecture defines the internal organization and functional units of the microprocessor. It includes the arithmetic logic unit (ALU) responsible for performing arithmetic and logical operations, the control unit for instruction decoding and execution control, and registers for temporary storage of data and instructions. The microarchitecture also encompasses the data path, control signals, and the interface to the memory subsystem.

Memory Subsystem: The memory subsystem includes both data memory and instruction memory. The data memory is

used for storing operands, intermediate results, and variables during program execution. The instruction memory holds the program instructions that are fetched and executed by the microprocessor. The system model specifies the organization, size, and access mechanisms of these memory components.

Interconnects: Interconnects facilitate communication between various components of the microprocessor, including the ALU, control unit, registers, and memory subsystem. This includes data buses for transferring data between the components, control signals for coordinating the execution of instructions, and address buses for accessing memory locations.

Peripheral Interfaces: In addition to the core microprocessor components, the system model may include peripheral interfaces for communication with external devices. These interfaces may include input/output (I/O) ports, serial communication interfaces, timers, interrupt controllers, and other peripheral modules required for interacting with the external environment.

Power and Clock Distribution: The system model considers the power distribution network (PDN) and clock distribution network (CDN) required to ensure proper functioning of the microprocessor. This includes the placement of power supply pads, power and ground rails, decoupling capacitors, and clock tree synthesis for distributing clock signals throughout the design.

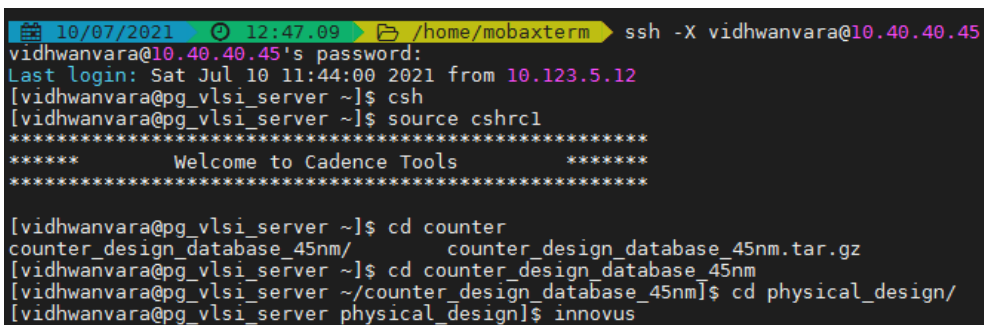
Design Constraints: The system model incorporates various design constraints and specifications, such as area constraints, power consumption targets, timing requirements, and technology-specific limitations. These constraints guide the physical design process to ensure that the resulting microprocessor meets the desired performance, power, and manufacturability goals.

The Procedures to create physical layout:

- The first step was to open the MobaXterm terminal and enter the appropriate username and password for authentication. Once logged in, the user needed to type specific commands to initiate the Cadence Innovus tool. These commands would typically include accessing the tool's installation directory, launching the software, and setting up the necessary environment variables and configurations. Following these instructions ensured that the user gained access to Cadence Innovus and set up the required environment for subsequent design activities.

1. To begin the physical design process, we opened the MobaXterm terminal and logged in using the respective username and password.

- Open the MobaXterm terminal:
 - Enter respective user name and password.
 - Type the following commands.

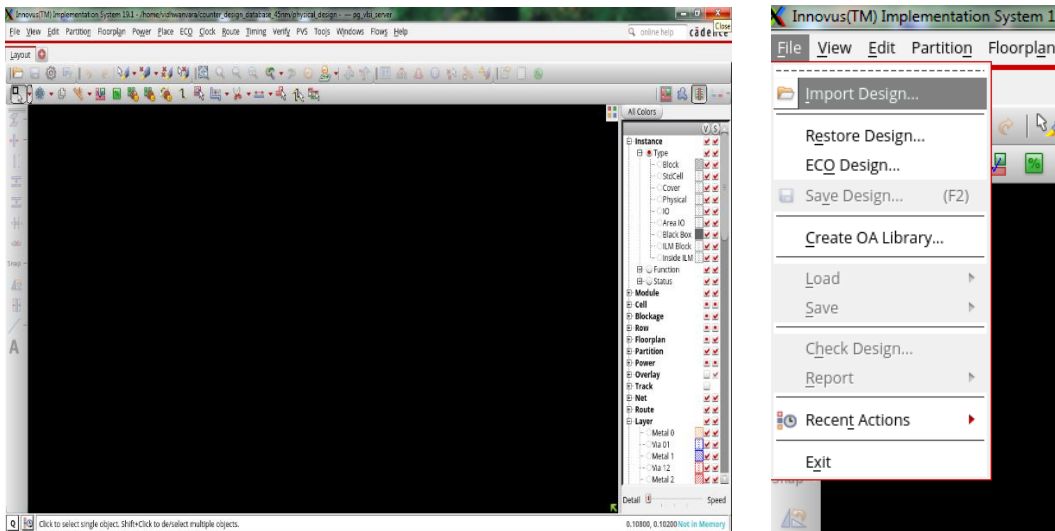


```
10/07/2021 12:47:09 /home/mobaxterm ssh -X vidhwanvara@10.40.40.45
vidhwanvara@10.40.40.45's password:
Last login: Sat Jul 10 11:44:00 2021 from 10.123.5.12
[vidhwanvara@pg_vlsi_server ~]$ csh
[vidhwanvara@pg_vlsi_server ~]$ source cshrc1
*****
***** Welcome to Cadence Tools *****
*****
[vidhwanvara@pg_vlsi_server ~]$ cd counter
counter_design_database_45nm/ counter_design_database_45nm.tar.gz
[vidhwanvara@pg_vlsi_server ~]$ cd counter_design_database_45nm
[vidhwanvara@pg_vlsi_server ~/counter_design_database_45nm]$ cd physical_design/
[vidhwanvara@pg_vlsi_server physical_design]$ innovus
```

2. After executing the necessary commands, the Innovus window appeared, providing access to the physical design environment.

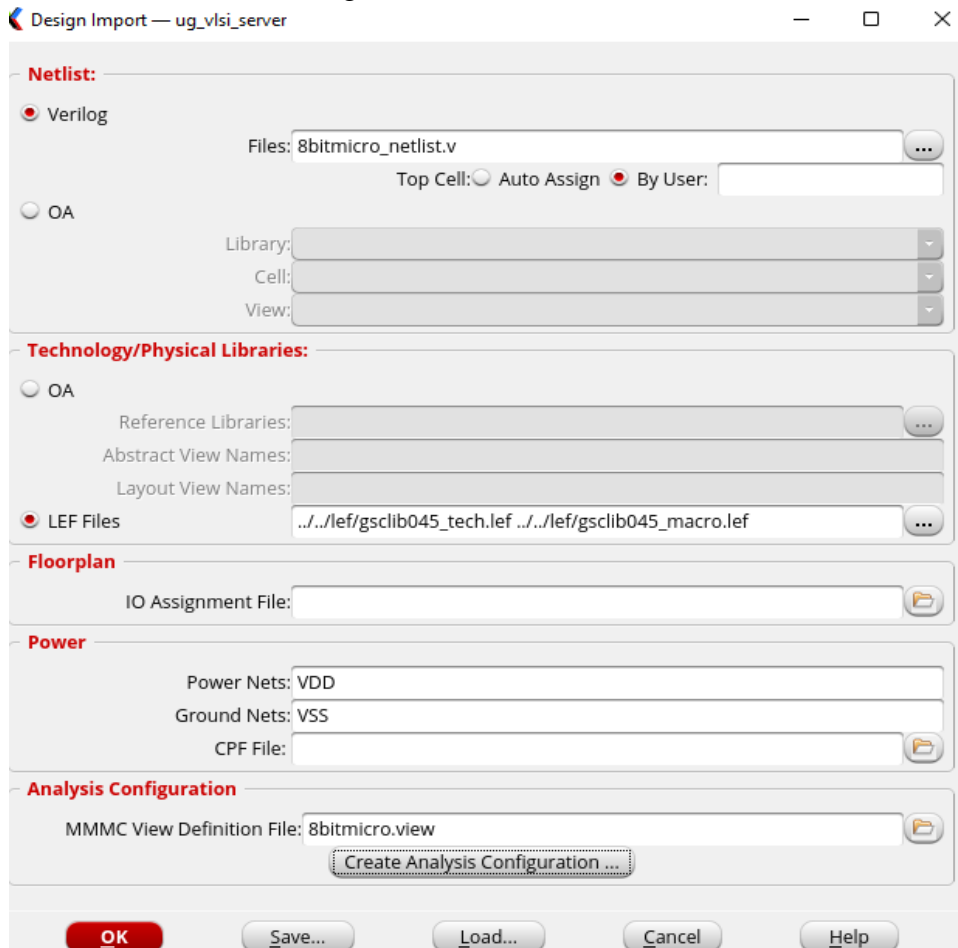
3. To import the design, we navigated to the "File" menu in Innovus and selected "Import design," allowing us to bring our design files into the workspace.

- **Innovus window** will appear.
 - Go to File>Import design

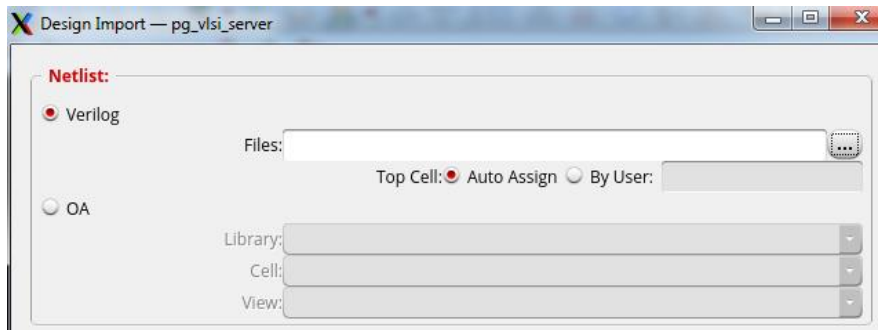


4. In the design import window, we enabled auto design and browsed for the netlist files that describe the circuit connections.

- **Design import window>Netlist**
 - Files>enable auto design>click on browse



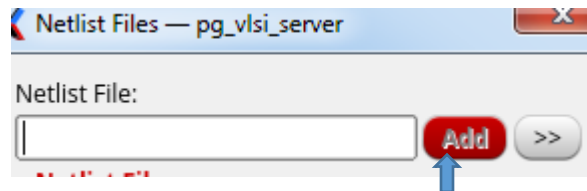
clock tree timing engine global stage delay update for slow:setup.late done. (took cpu=0:00:00.1 real=0:00:00.1)



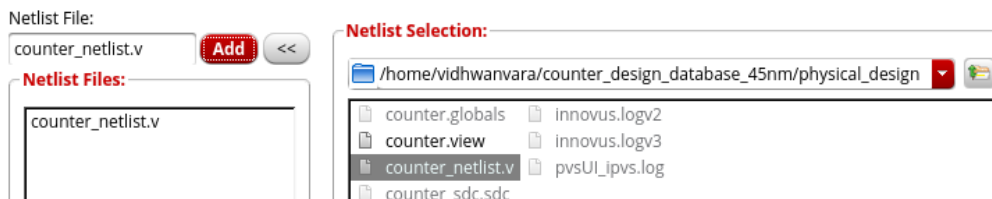
5. Within the netlist files window, we clicked on ">>" to browse for and add the specific "counter_netlist.v" file from the designated folder. The added file should be visible in the Netlist Files list.

- **Netlist Files window**

- Click on >>



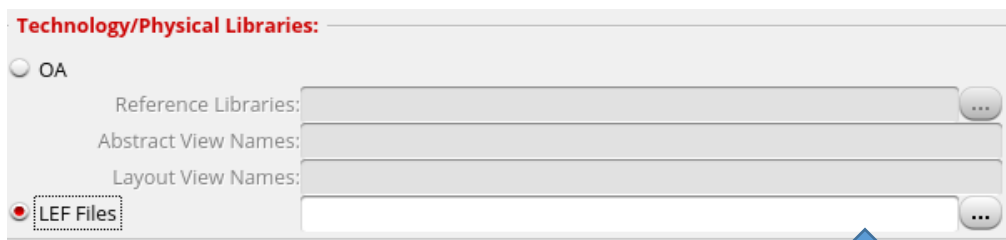
- Select counter_netlist.v from counter_design_database_45nm/design folder.
- Click on Add>close (counter_netlist.v file should be visible under Netlist Files)



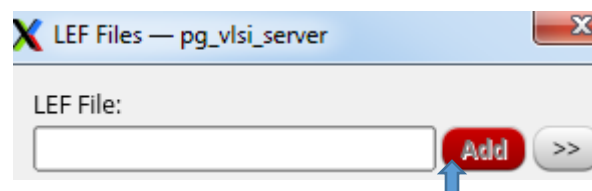
6. For proper library integration, we selected the required LEF (Library Exchange Format) files by clicking on "Select LEF Files" and navigating to the appropriate folder.

- **Technology/Physical Libraries**

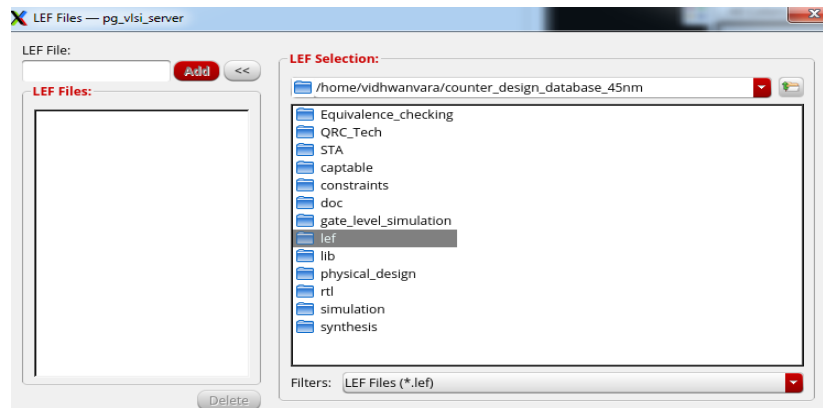
- Select LEF Files>click on browse



- Click on>>

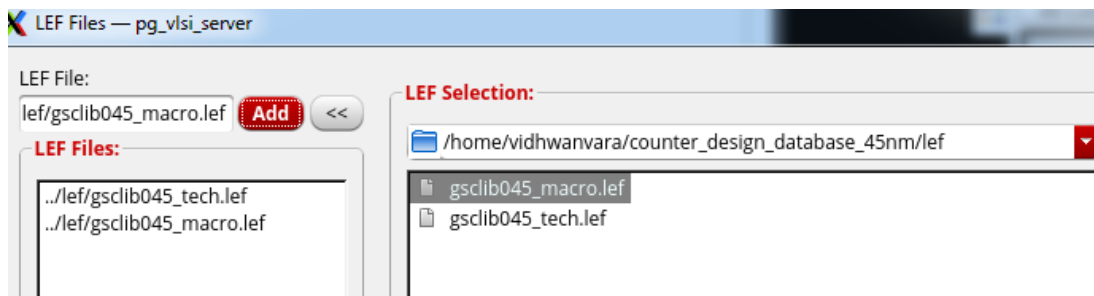


- Double click on lef folder from counter_design_database_45nm/design folder.

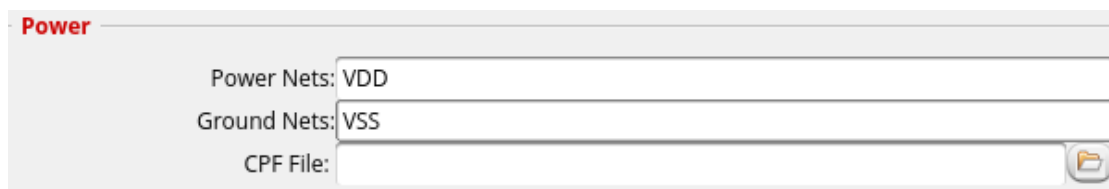


7. After selecting the necessary LEF files, we proceeded to specify the power and ground nets by entering "VDD" for power nets and "VSS" for ground nets.

- **LEF Selection**>first select tech.lef file>add>macro.lef file>add>close

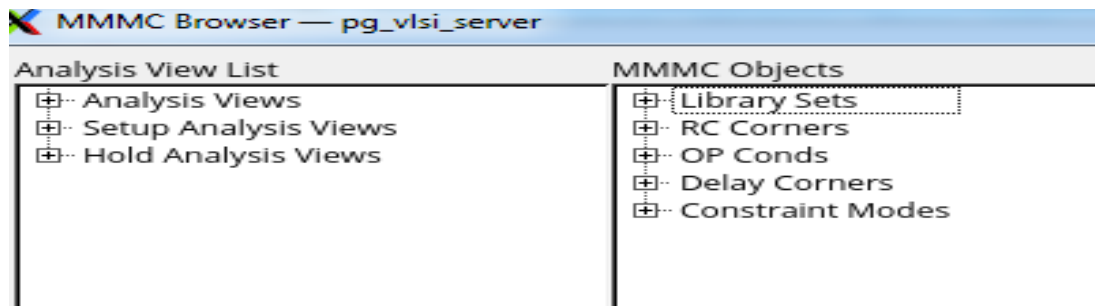
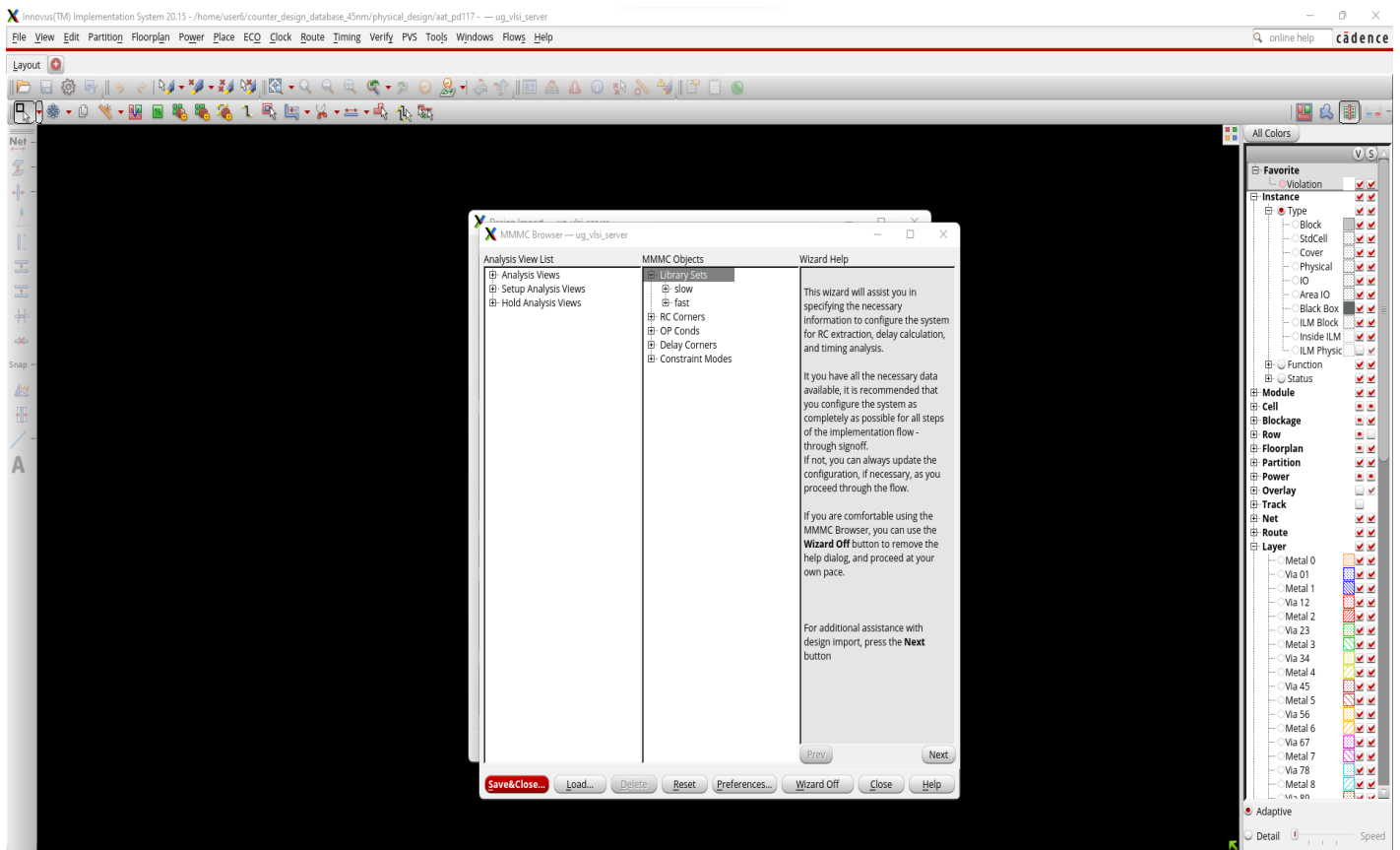


- **Power**>Power Nets: type VDD>Ground Nets: VSS



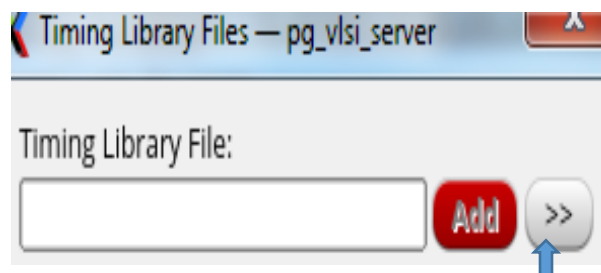
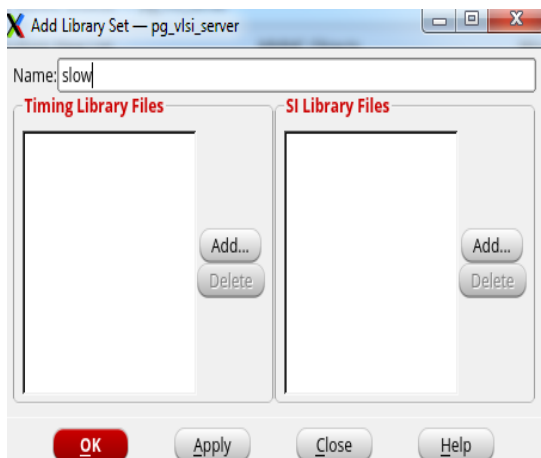
8. In the analysis configuration step, we created an analysis configuration and opened the MMMC (Multi-Mode Multi-Corner) Browser window.

- **Analysis Configuration**>create analysis configuration>MMMC Browser window will appear.

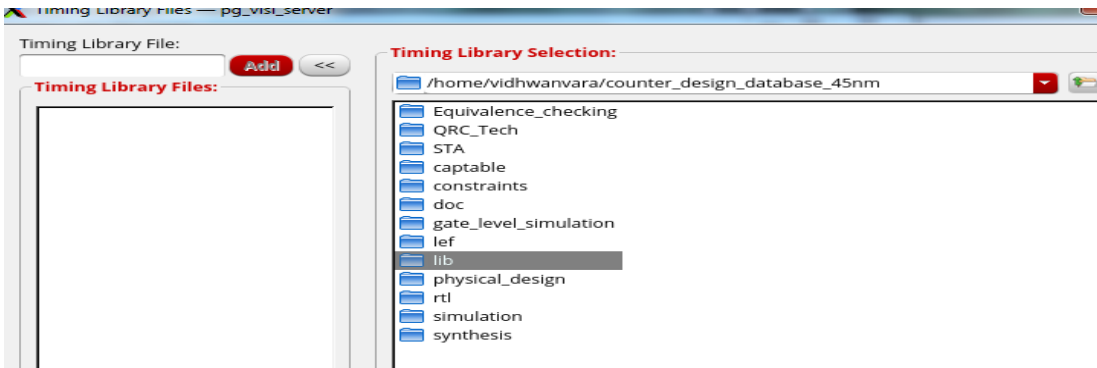


➤ Double click on Library Sets>Name: slow>Add>Timing Library Files window> click on>>

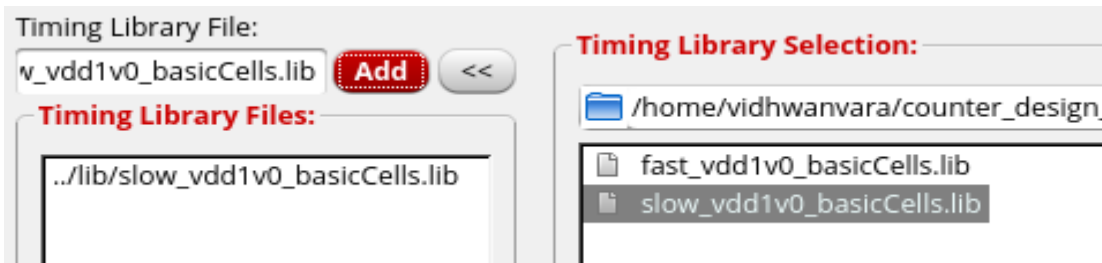
9. Inside the MMMC Browser window, we added the timing library files for the "slow" library set by double-clicking on "Library Sets," providing a name (e.g., "slow"), and selecting the appropriate library files.



- Double click on lib files

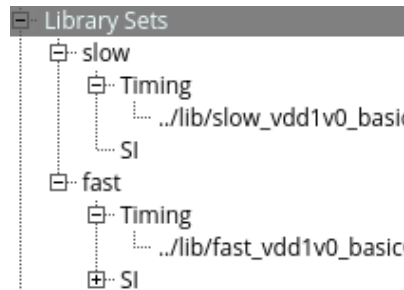


- Select slow_vdd>Add>close>ok



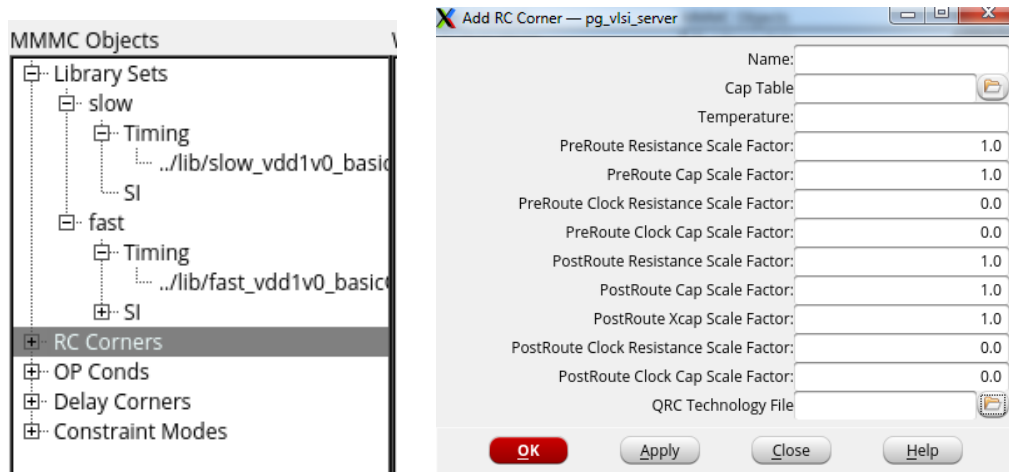
10. Similarly, we repeated the procedure to add the timing library files for the "fast" library set, ensuring that both the slow and fast library files were visible under "Library Sets."

- Double click on Library Sets>Name: fast>Add>Timing Library Files window> click on>>
- Double click on lib files
- Select fast_vdd>Add>close>ok
- Under Library Sets both the slow and fast lib files should be visible.



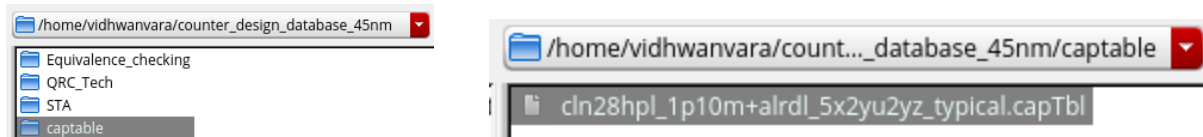
11. To account for process variations, we added RC corners by double-clicking on "RC Corners" and specifying a name (e.g., "rc_corner").

- Double click on RC Corners>Add RC corner window will appear>Name: rc_corner



12. We then configured the Cap Table by clicking on the folder icon and selecting the captable folder from the designated location.

- Cap Table>click on folder icon>Under counter_design_database_45nm>double click on captable>click on the .capTbl>open

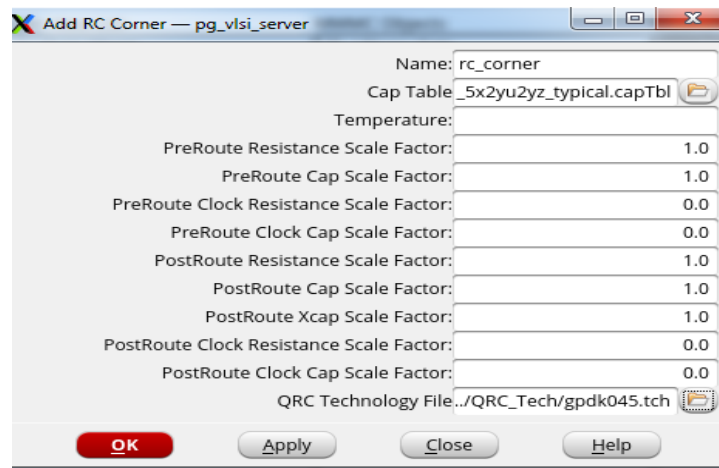


13. For accurate technology file selection, we clicked on the folder icon under "QRC Technology File" and navigated to the QRC_Tech folder, where we selected the "gpdK045.tch" file.

- QRC Technology File> click on folder icon>Under counter_design_database_45nm>double click on QRC_Tech>select gpdK045.tch>open

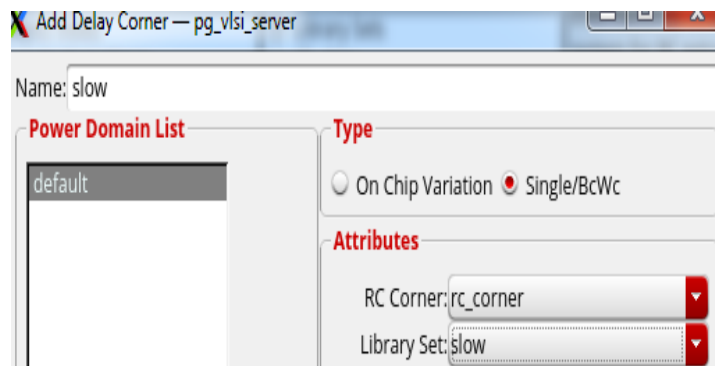
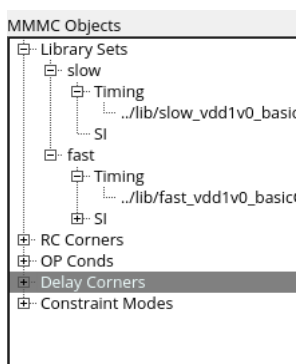


- Once the required contents are filled as shown in the below figure click ok.



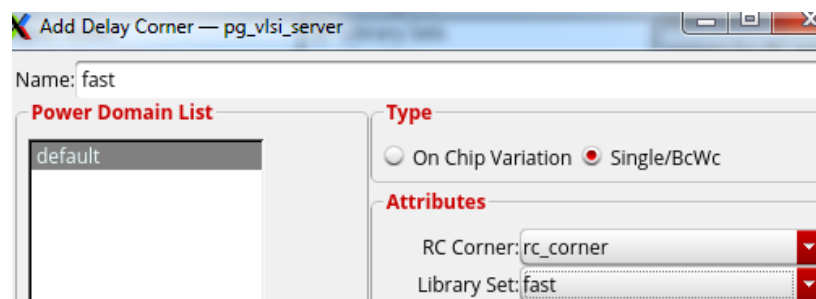
14. After filling in the required contents, we clicked "OK" to proceed with the analysis configuration setup.

- Double click on Delay Corners>Add Delay Corner window will appear>Name: slow>Under Attributes>RC Corner: rc_corner>Library Set: select slow from drop down>ok

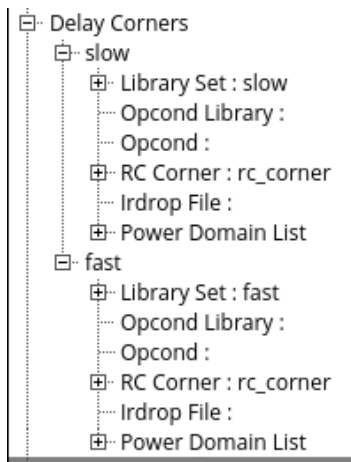


15. Next, we added delay corners for both the "slow" and "fast" libraries, ensuring that the specified library files were visible.

- Repeat the same step for fast.

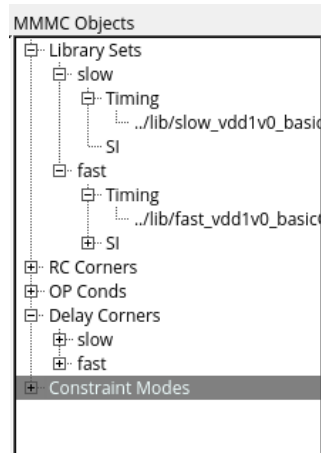


- Once the slow and fast lib files are added to the delay corners make sure the files shown below are visible.



16. To define the constraints, we added constraint modes by double-clicking on "Constraint Modes" and providing a name (e.g., "counter_sdc").

- Double click on Constraint Modes> Add constraint mode window will appear>Name: type counter_sdc>click on Add>



17. Under "SDC constraint file," we clicked on ">>>" and selected the "counter_sdc.sdc" file from the counter_design_database_45nm/physical design folder.

- Under SDC constraint file click on>>>select 8bitmicro_sdc.sdc from counter_design_database_45nm/physical design>click on Add>Sdc files will be visible under Constraint modes>ok

18. In the Analysis View List, we double-clicked on "Analysis Views" and added analysis views for both "slow" and "fast" modes, specifying the appropriate constraint mode and delay corner.

- **Under Analysis View List**>Double click on Analysis Views>Add Analysis View window will pop up>Name: type slow>Constraint Mode: counter_sdc>Delay Corner: select slow from drop down>ok
- Repeat the above steps for fast analysis view.

19. Continuing with the setup, we added a setup analysis view by selecting "Setup Analysis Views" and choosing the "slow" analysis view from the drop-down menu.

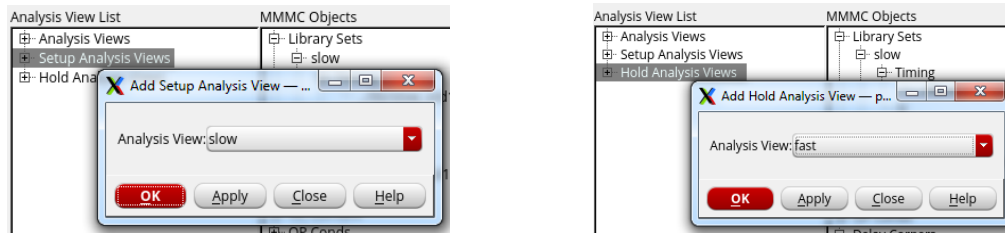
- Setup Analysis Views>Add Setup Analysis View>Analysis View: Select slow from drop down>ok

20. Likewise, we added a hold analysis view by selecting "Hold Analysis Views" and choosing the "fast" analysis view

- Hold Analysis Views>Add Hold Analysis View>Analysis View: Select fast from drop down>ok

21. Once the analysis views were added, we saved and closed the configuration.

- Save&close



22. Upon saving the MMMC View Definition File, we were prompted to provide a file name and save it accordingly.

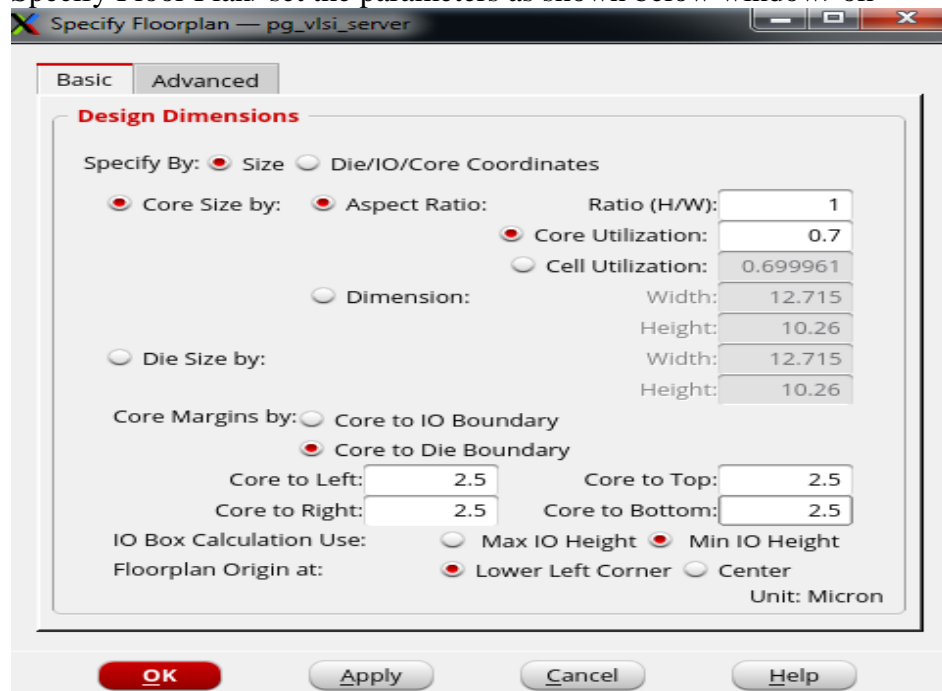
- Save MMMC View Definition File window will pop up>Give the file 8bitmicro.view>save>

23. In the Global Variables window, we saved the file with a .globals extension and clicked "OK" on the Design Import window to proceed.

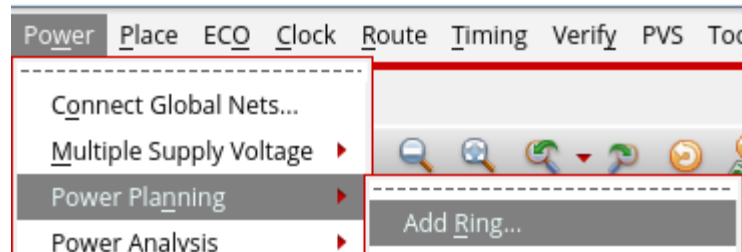
Once saved Global Variables window will pop up>enter file 8bitmicro.globals>save>click ok on Design Import window.

23. In the Global Variables window, we saved the file with a .globals extension and clicked "OK" on the Design Import window to proceed.

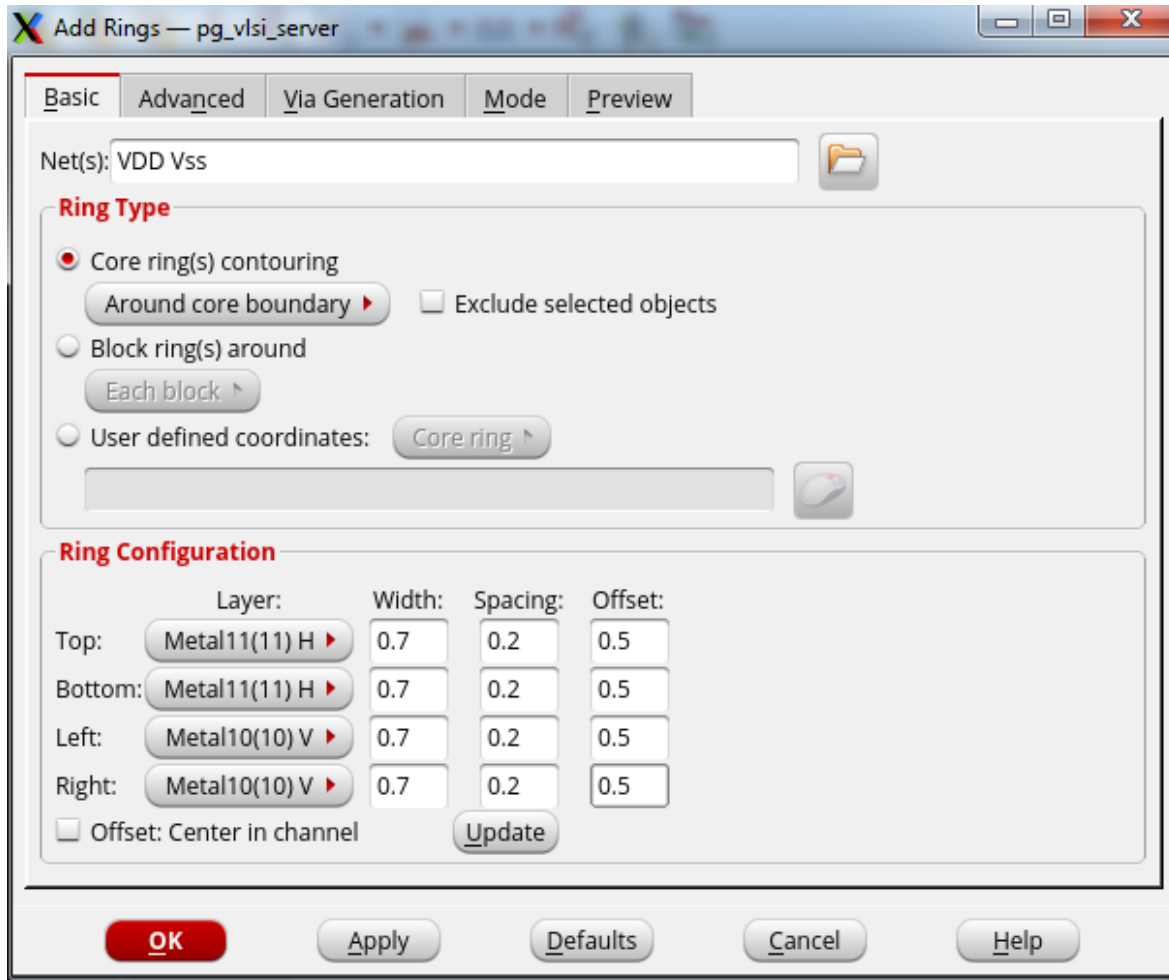
- Floor Plan>Specify Floor Plan>set the parameters as shown below window>ok



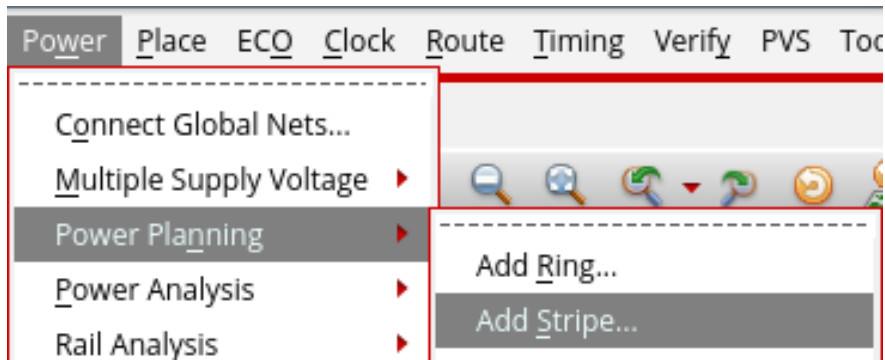
- **Power**>Power Planning>Add Ring> Net(s)>select VDD and VSS from folder(Vdd and Vss not case sensitive)



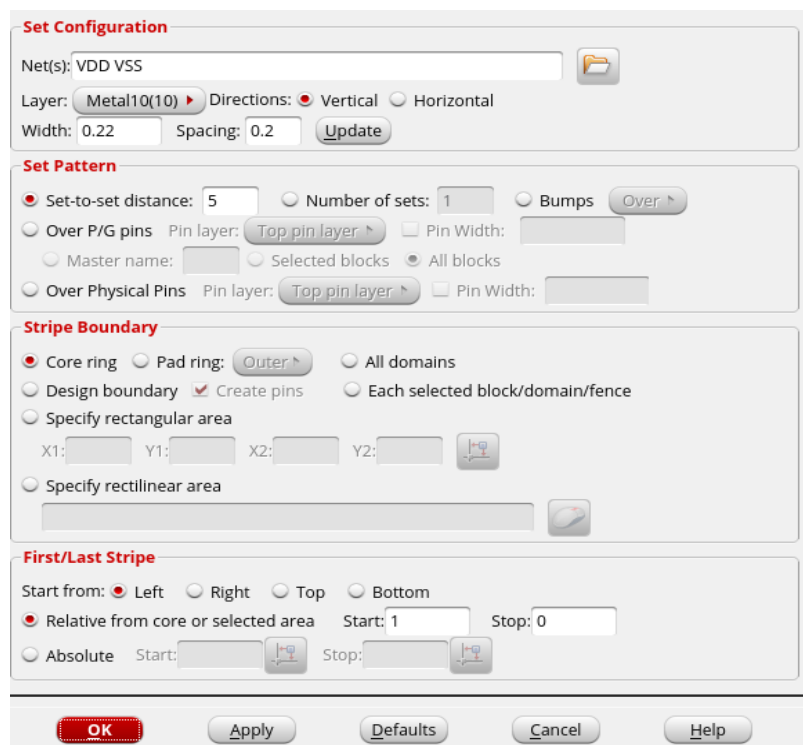
- **Ring Type**>enable Core ring(s)contouring
- **Ring Configuration**>select the metal layers>Top & Bottom: metal 11>Left & Right: metal 10>width:0.7>Spacing:0.2>Offset:0.5>ok



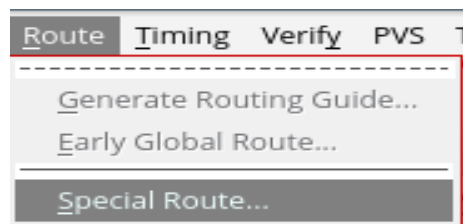
- Once the power planning parameters are set the window should appear as shown below
- Power>Power Planning>Add Stripe



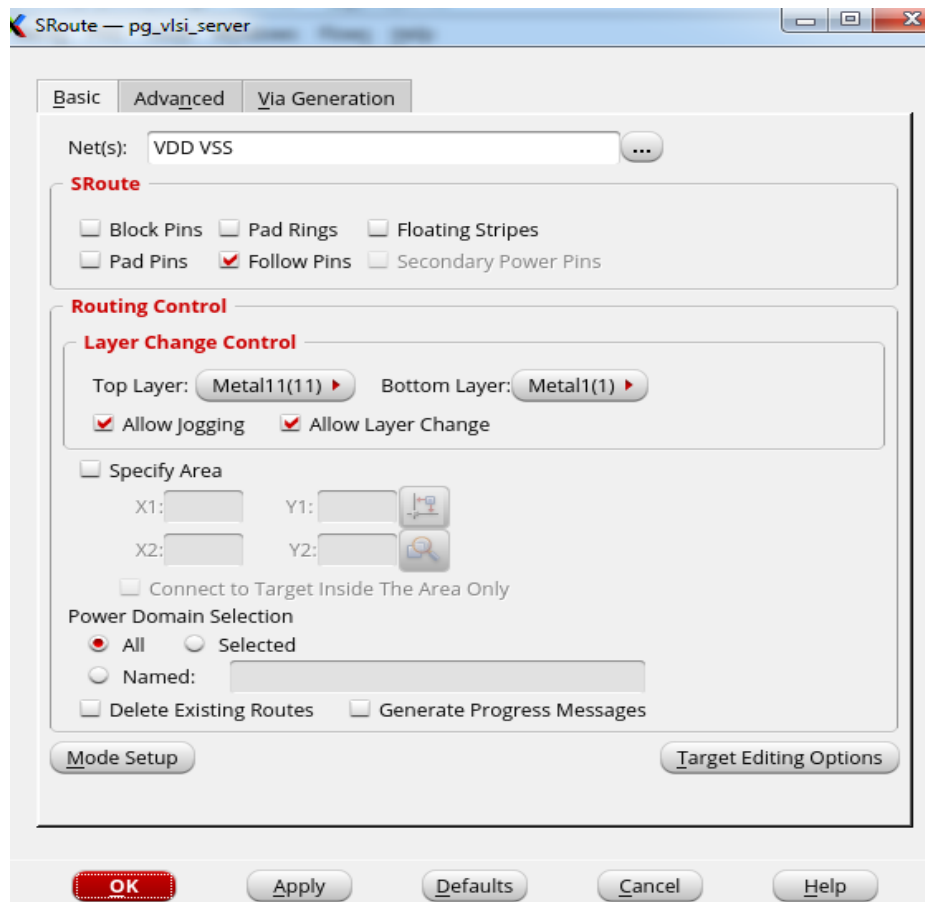
- Set Configuration>Net(s):select VDD and VSS from folder>Set pattern>set to set distance:5>First/Last stripe>Start:1>stop:0>ok



- Route>Special Route



- SRoute>Net(s): select VDD and VSS from folder>Sroute>enable Follow Pins>ok



The 8bitmicro design consists of scan chains.

What are scan chains?

Scan chains are elements in scan base designs that are used to shift-in and shift-out test data. A scan chain is formed by number of flops connected back to back in a chain with the output of one flop connected to another. Purpose: Scan chains are inserted into designs to shift the test data into the chip and out of the chip in order to make every point in the chip controllable and observable.

Steps to extract scandef files:

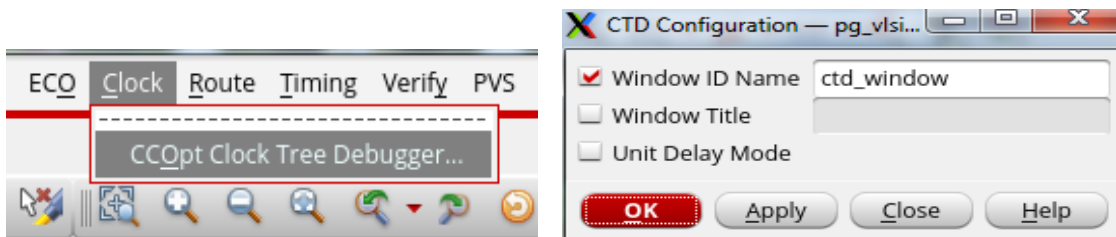
- Open new terminal>enter the login credentials>counter_design_database_45nm>synthesis>genus – legacy_ui
- Legacy_genus:/>source rc_dft_script.tcl>write_scandef > counter.scandef

➤ `innovus 3> setPlaceMode -place_global_place_io_pins true`

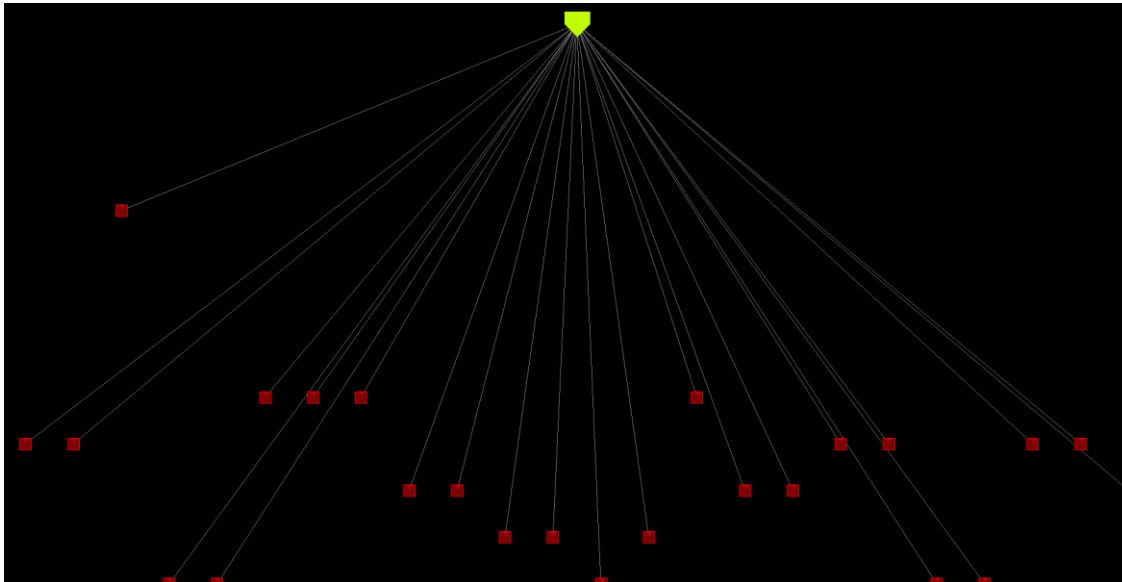
➤ `innovus 4> place_opt_design`

➤ `innovus 5> innovus 5> create_ccopt_clock_tree_spec`

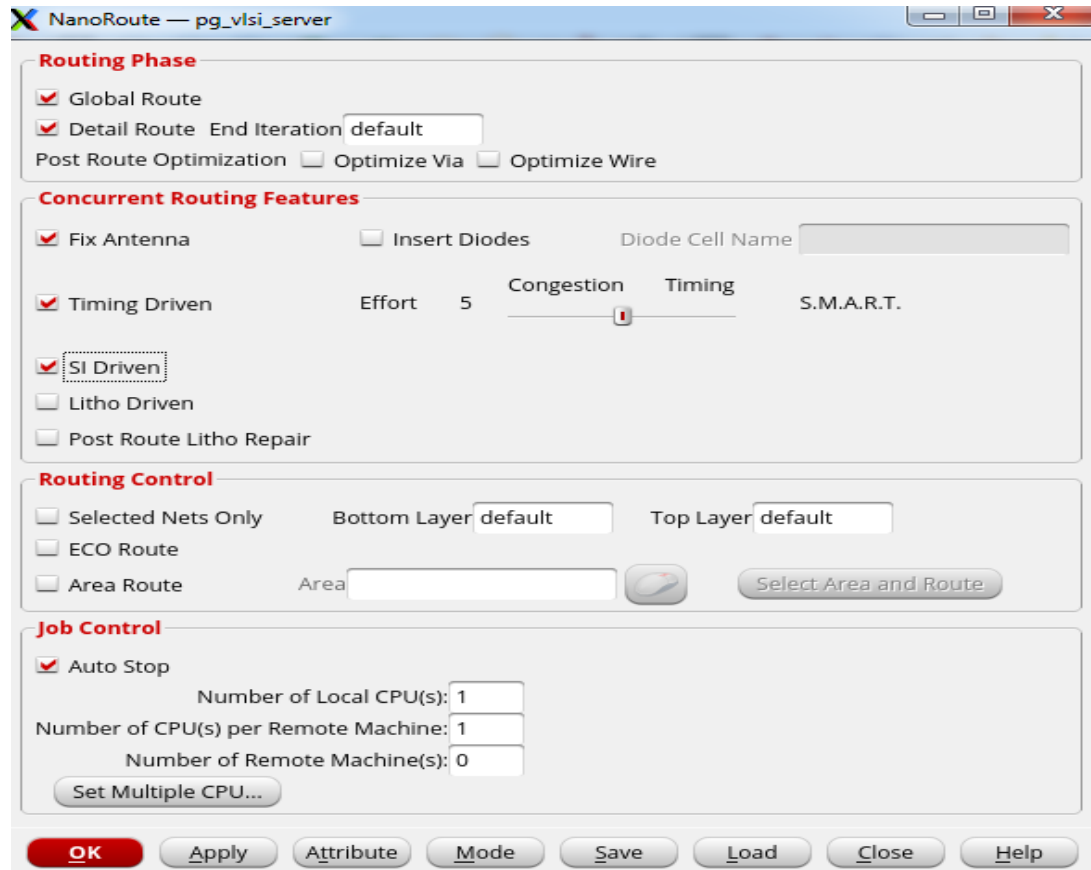
➤ `innovus 6> ccopt design`



- It shows the clock tree built.



- Route>Nanoroute>enable SI Driven>ok



- Type report_power in the terminal for power report

Total Power					

Total Internal Power:	0.01403873		79.9200%		
Total Switching Power:	0.00350609		19.9596%		
Total Leakage Power:	0.00002116		0.1204%		
Total Power:	0.01756597				

Group	Internal Power	Switching Power	Leakage Power	Total Power	Percentage (%)

Sequential	0.01096	0.0008455	9.372e-06	0.01182	67.27
Macro	0	0	0	0	0
I/O	0	0	0	0	0
Combinational	0.003077	0.002661	1.178e-05	0.00575	32.73
Clock (Combinational)	0	0	0	0	0
Clock (Sequential)	0	0	0	0	0

Total	0.01404	0.003506	2.116e-05	0.01757	100

Rail	Voltage	Internal Power	Switching Power	Leakage Power	Total Power Percentage (%)

Default	1.1	0.01404	0.003506	2.116e-05	0.01757 100

* Power Distribution Summary:					
* Highest Average Power: accumulator_reg[1] (SDFFRHQX1): 0.0006887					
* Highest Leakage Power: accumulator_reg[6] (SDFFRHQX1): 5.073e-07					
* Total Cap: 2.67015e-13 F					
* Total instances in design: 125					
* Total instances in design with no power: 0					
* Total instances in design with no activity: 0					
* Total Fillers and Decap: 0					

➤ Type report_timing in the terminal for timing report

```

Path 1: MET Setup Check with Pin temp_reg[6]/CK
Endpoint: temp_reg[6]/D (^) checked with leading edge of 'clk'
Beginpoint: opcode[1] (v) triggered by leading edge of '@'
Path Groups: {clk}
Analysis View: slow
Other End Arrival Time      0.000
- Setup                    0.014
+ Phase Shift              10.000
- Uncertainty              0.010
= Required Time            9.976
- Arrival Time             0.595
= Slack Time               9.381
  Clock Rise Edge          0.000
  + Input Delay            0.000
  = Beginpoint Arrival Time 0.000

```

Instance	Arc	Cell	Delay	Arrival Time	Required Time

	opcode[1] v			0.000	9.381
g3775	A v -> Y ^	INVXL	0.095	0.095	9.477
g3711	A1 ^ -> Y v	OAI21X1	0.067	0.162	9.544
g3685	B v -> C0 v	ADDFX1	0.056	0.218	9.599
g3680	B v -> C0 v	ADDFX1	0.051	0.268	9.650
g3675	B v -> C0 v	ADDFX1	0.050	0.318	9.699
g3670	B v -> C0 v	ADDFX1	0.050	0.368	9.749
g3665	B v -> C0 v	ADDFX1	0.052	0.420	9.801
g3660	B v -> S ^	ADDFX1	0.071	0.491	9.872
g3656	A1 ^ -> Y v	AOI22X1	0.026	0.517	9.898
g3654	A1 v -> Y ^	OAI22X1	0.034	0.551	9.932
g3651	A1 ^ -> Y ^	A022XL	0.044	0.595	9.976
temp_reg[6]	D ^	DFFQXL	0.000	0.595	9.976

➤ Give the following command to save the design

innovus 9> saveDesign postRoute

- View for the message summary for warnings and errors.

```
innovus 9> saveDesign postRoute
#% Begin save design ... (date=06/26 11:35:30, mem=1450.9M)
% Begin Save ccopt configuration ... (date=06/26 11:35:30, mem=1453.9M)
% End Save ccopt configuration ... (date=06/26 11:35:30, total cpu=0:00:00.1, real=0:00:00.0, peak res=1454.8M, current mem=1454.8M)
% Begin Save netlist data ... (date=06/26 11:35:30, mem=1454.8M)
Writing Binary DB to postRoute.dat/Microprocessor8Bit.v.bin in single-threaded mode...
% End Save netlist data ... (date=06/26 11:35:30, total cpu=0:00:00.0, real=0:00:00.0, peak res=1455.0M, current mem=1455.0M)
Saving symbol-table file ...
Saving congestion map file postRoute.dat/Microprocessor8Bit.route.congmap.gz ...
% Begin Save AAE data ... (date=06/26 11:35:30, mem=1455.6M)
Saving AAE Data ...
% End Save AAE data ... (date=06/26 11:35:30, total cpu=0:00:00.0, real=0:00:00.0, peak res=1455.6M, current mem=1455.6M)
Saving preference file postRoute.dat/gui.pref.tcl ...
Saving mode setting ...
Saving global file ...
% Begin Save floorplan data ... (date=06/26 11:35:31, mem=1456.3M)
Saving floorplan file ...
% End Save floorplan data ... (date=06/26 11:35:31, total cpu=0:00:00.1, real=0:00:00.0, peak res=1456.4M, current mem=1456.4M)
Saving Drc markers ...
... 1 markers are saved ...
... 0 geometry drc markers are saved ...
... 0 antenna drc markers are saved ...
% Begin Save placement data ... (date=06/26 11:35:31, mem=1456.4M)
** Saving stdCellPlacement_binary (version# 2) ...
Save Adaptive View Pruning View Names to Binary file
% End Save placement data ... (date=06/26 11:35:31, total cpu=0:00:00.1, real=0:00:00.0, peak res=1456.4M, current mem=1456.4M)
% Begin Save routing data ... (date=06/26 11:35:31, mem=1456.4M)
Saving route file ...
*** Completed saveRoute (cpu=0:00:00.0 real=0:00:01.0 mem=1977.2M) ***
% End Save routing data ... (date=06/26 11:35:32, total cpu=0:00:00.1, real=0:00:01.0, peak res=1456.6M, current mem=1456.6M)
Saving property file postRoute.dat/Microprocessor8Bit.prop
*** Completed saveProperty (cpu=0:00:00.0 real=0:00:00.0 mem=1980.2M) ***
#Saving pin access data to file postRoute.dat/Microprocessor8Bit.apa ...
#
% Begin Save power constraints data ... (date=06/26 11:35:32, mem=1457.2M)
% End Save power constraints data ... (date=06/26 11:35:32, total cpu=0:00:00.1, real=0:00:00.0, peak res=1457.3M, current mem=1457.3M)
Generated self-contained design postRoute.dat
#% End save design ... (date=06/26 11:35:32, total cpu=0:00:01.2, real=0:00:02.0, peak res=1459.9M, current mem=1459.9M)
*** Message Summary: 0 warning(s), 0 error(s)
```

- Type ls in the terminal window and view the saved file.

```
innovus 10> ls
8bitmicro.globals 8bitmicro.view 8bitmicro_sdc.sdc innovus.cmd1 innovus.log1 innovus.logv1 postRoute.dat pvsUI ipvs.log timingReports
8bitmicro.v 8bitmicro_netlist.v innovus.cmd innovus.log innovus.logv postRoute power.rpt rc_model.bin
innovus 11>
```

```
innovus 10> ls
8bitmicro.globals 8bitmicro.view 8bitmicro_sdc.sdc innovus.cmd1 innovus.log1 innovus.logv1 postRoute.dat pvsUI ipvs.log timingReports
8bitmicro.v 8bitmicro_netlist.v innovus.cmd innovus.log innovus.logv postRoute power.rpt rc_model.bin
innovus 11>
```

