SECTION OF ELECTRICAL ENGINEERING

EPFL STI – SEL
ELG 030
Station nº 11
CH-1015 Lausanne

Téléphone : +4121 693 13 46
Fax :
E-mail : alexandre.levisse@epfl.ch
Site web : esl.epfl.ch

Fundamentals of VLSI

September 2024

FVLSI – Digital Design Flow
Back End – From Synthesis to GDS

1. SETUP THE LAB ENVIRONMENT

For this part of the lab, we will use three new tools, namely Innovus, Embedit and Virtuoso configured for the 65nm globalfoundries design kit.

You will find them in your working environment.

2. TUNING THE SIZE OF AN SRAM MEMORY

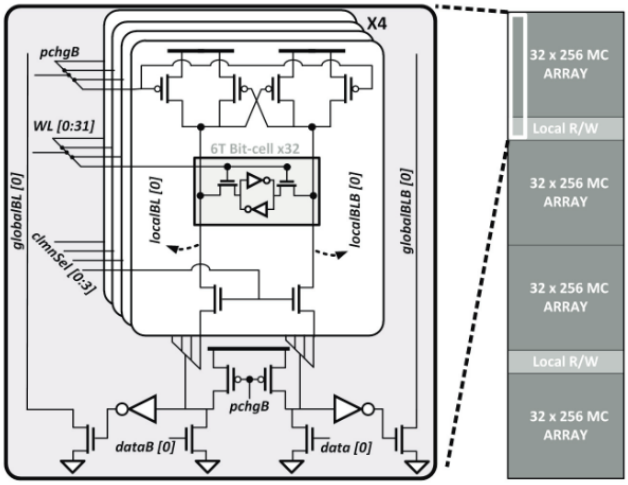
As you did identify previously, the 64x64bit memory appeared as a bottleneck in the design, from an area standpoint. In this section, we propose to explore how to generate a SRAM memory using a tool called a "memory compiler". And generate smaller memories which could improve the density of the design we are exploring.

2.1. GENERALITIES ABOUT SRAM MEMORIES

SRAM memories are complex and cannot be automatically generated by a synthesis tool. For e.g. instantiating in a RTL language, an array of memorization points will infer an array of flipflops. This is inefficient from an (i) area, (ii) power and (iii) speed standpoint from a certain size. Thereby, it exists some kind of trade-off between the memory size, and it's implementation as an array of register files and the need for the utilization of an SRAM memory[1].

An SRAM bitcell is built as shown in the following picture. It is practically composed of two inverters and two NMOS transistors called access transistors. The SRAM bitcells are then organized in columns and rows. The following screenshot shows an example of memory organization from a publication in the ISSCC conference[2]. In that case, several small 32x256bicells arrays are built, each with their associated peripheral circuit. These blocks are then replicated and connected together to build larger memory arrays.

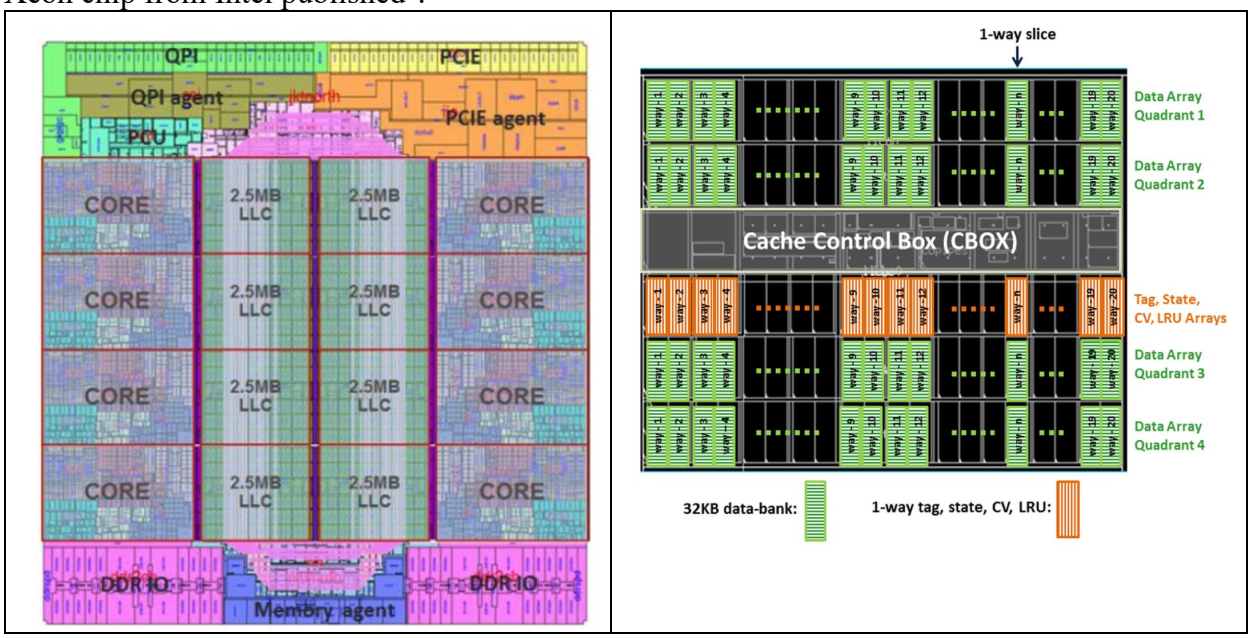
[1] Some works actually explored the trade-off https://ieeexplore.ieee.org/abstract/document/5976987
[2] https://ieeexplore.ieee.org/document/5746310



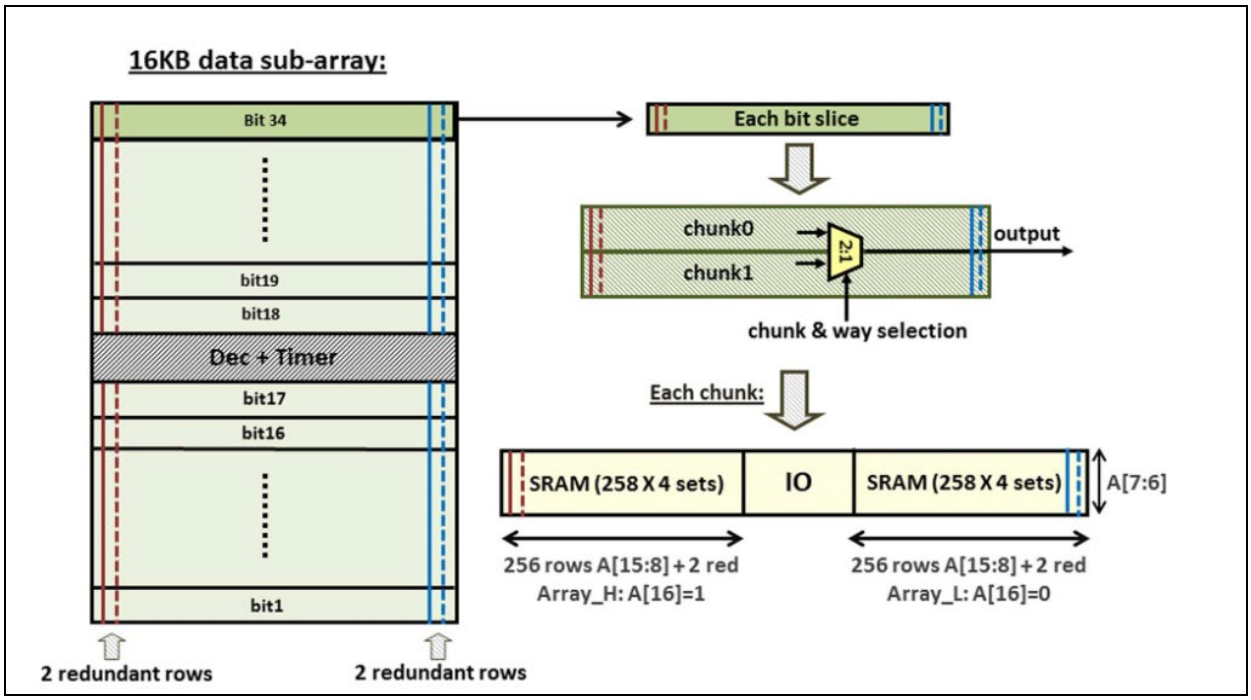
Each block composing a SRAM array has to be designed with the full custom flow as you did learn during phase 1 of these labs. As today memory compose more than 50% of the area of advanced chips, being able to automate the design of such blocks is mandatory.

Let's start with an example of an actual memory, to understand the complexity of real memory systems. Note that here we only focus on the physical organization of a memory from a 2013 processor built in 32nm CMOS technology. We do not detail process associated with cache management etc.

The associated screenshots is a detailed Last Level Cache memory hierarchy example of a 2013 Xeon chip from Intel published[3].



[3] https://ieeexplore.ieee.org/abstract/document/6515193



On the top left, one can see the 8 processors (CORE), and the 8x2.5MB (Mega Byte) LLC (Last Level Cache) in the middle, which overall makes 20MB (note that each LLC block is not tied to a core but that it is overall controlled by the memory agent on the bottom, each block being controlled by its Cache Control Box – cf top right picture). The right picture is a zoom in on a 2.5MB block. It is composed of 20 "way slices" (vertical) of 128KB each composed of 4 32KB data banks (green boxes). Each 32KB data-bank is composed of 2 16KB data sub-array shown on the bottom picture.

The 16KByte array is then cut into 34 "bit" slices. Each "bit" slice is composed of two "chunks" (chunk 0 and chunk 1). Each chunk is composed of two 258 x 4 bitcell SRAM arrays.

Let's make the calculation :

1 chunk is 2* (258*4 bits) = 2.064bits

1 bit slice is 2 chunks = 4.128bits

1 data sub array = 34 bit slices = 34 * 4.128bit = 140.352bits = 17.544Bytes

And so on…

- Note how the number of bits does not match the actual memory size, and how things do not match between the physical quantity of bits being integrated.
- The 16KB actual memory size is only obtained when removing physical redundancy bitcells (in prevision of failure for e.g.), reserved ECC (Error Correction Codes) bits etc.

Bottomline being that the physical organization of a memory is not homogeneous, extremely hierarchical. Ultimately, unitary memory blocks are relatively small, here around 16KB. Picture that each of these 16KB blocks is replicated 1280 times to make the complete 20MB LLC cache.

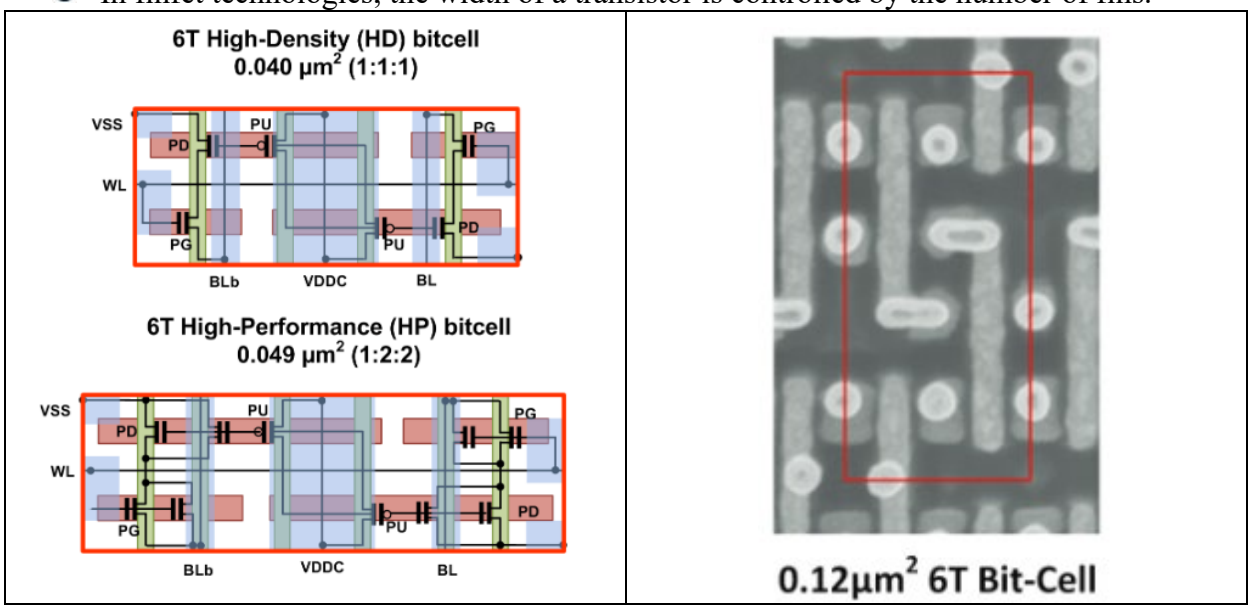
When diving down the CORE block, besides the LLC, in these Xeon cores, there is also a L2 and a L1 cache, private for each processor. In this architecture, the L2 has a 256KB hierarchical

organization while the L1 is 32KB[4] with the same kind of oversizing as we discussed before. Note that the L1, L2 and LLC are built with different constraints in terms of speed. Closer cache levels are expected to go faster, while LLC is expected to be denser and slower.

Various parameters can be tuned to change the performances of a memory in terms of speed or power. One of them is by tuning the size of the memory. Indeed, smaller arrays mean less metal, in line meaning faster access (remember the layout of your 8bit adder ?).

Another approach consists in tuning the parameter of the access transistors. The following left figure is an example of two different bitcells in 10nm FinFet technology from Samsung[5]. Both being what we call 6T (6 transistor) SRAM , the top one High Density (HD), the bottom one High Speed (HS). Here, increasing the width of the access transistors (called PG) and of the pulldowns (called PD) makes the bitcell go faster for read operations.

- In finfet technologies, the width of a transistor is controlled by the number of fins.



One particularity of SRAM (and memories in general), that they have an extremely regular layout pattern as opposed to logic circuits (as you did in your 8bit ALU). Thereby, SRAM designers can discuss with the foundry and negotiate on breaking DRC rules. The main reason being that DRC rules are built for logic circuits and can be twisted in some case, if agreed with the foundry. The right part of the graph above shows a SEM picture of a 6T SRAM cell (Poly, Active and Contacts). Note the rectangular contacts, this is generally forbidden in logic DRC.

Then, what does a memory compiler do ?

There is no general rules. Each provider does things in their ways. Though, the general approach is as follows. The process of designing a SRAM memory starts by designing and evaluating by hand a memory. Then evaluating the trends and characterizing the different blocks. Build them in a way that they can be automatically and efficiently abutted. Build a performance/power/area

[4] https://www.intel.com/content/www/us/en/support/articles/000027820/processors/intel-xeon-processors.html
[5] https://ieeexplore.ieee.org/abstract/document/7725555

model. Finally, build a tool that builds a memory that meets the specifications required by the user based on these models. Bottomline being that a memory compiler is mainly playing lego with already existing/configurable blocks.

One these basics are understood, let's now play with a memory compiler and build different memories.

2.2. USING THE MEMORY COMPILER EMBEDIT FROM SYNOPSYS

The Synopsys company does develop EDA tools (such as DesignCompiler or PrimeTime) though they are also a renown high quality silicon IPs which companies and universities can license and use their designs[6].

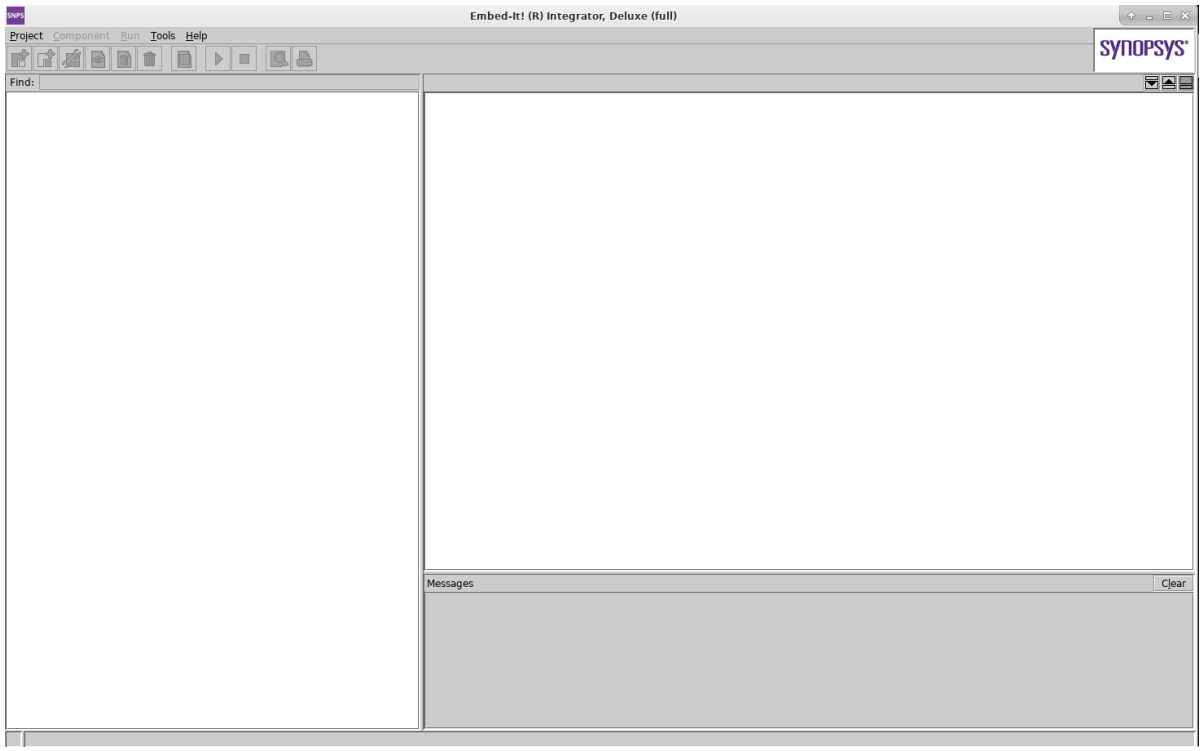
In this lab, we will use the Synopsys memory compilers and their tool called Embedit. Note that there exist hundreds of memory compilers, each having their own specificities. We use embedit here as it provides a comfortable-to-use Graphical User Interface (GUI), though, most of the memory compilers do not provide GUI (surprised ? you should not be).

1. Go in the EMBEDIT folder

edalabs_phase2_2024>cd EMBEDIT

2. Run the integrator tool (don't forget the space between snps and integrator)

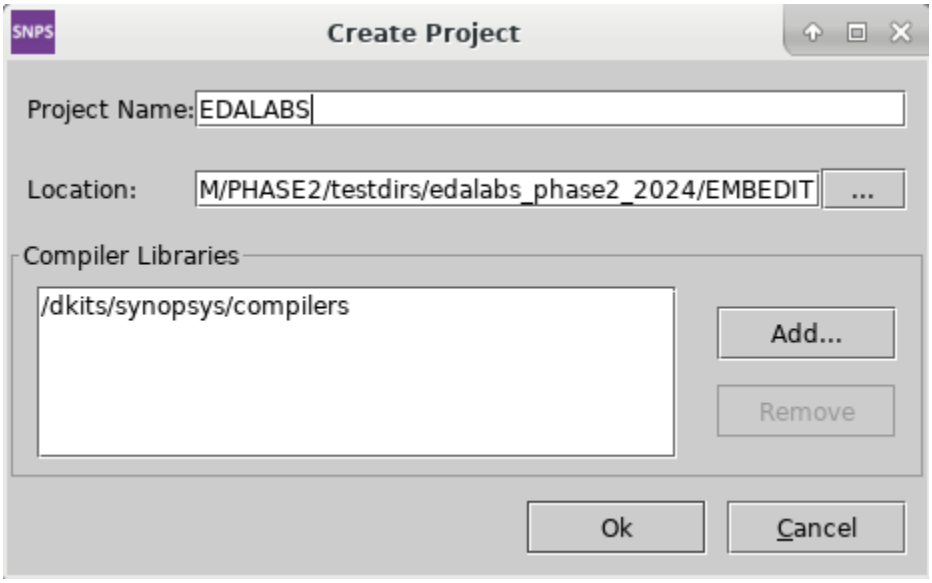
EMBEDIT> snps integrator &



3. Click on projet>New
4. Call the project EDALABS

[6] https://www.synopsys.com/designware-ip.html

5. Keep the location in the EMBEDIT folder
6. Click OK → *in the screenshot, the path may not be the same as yours*

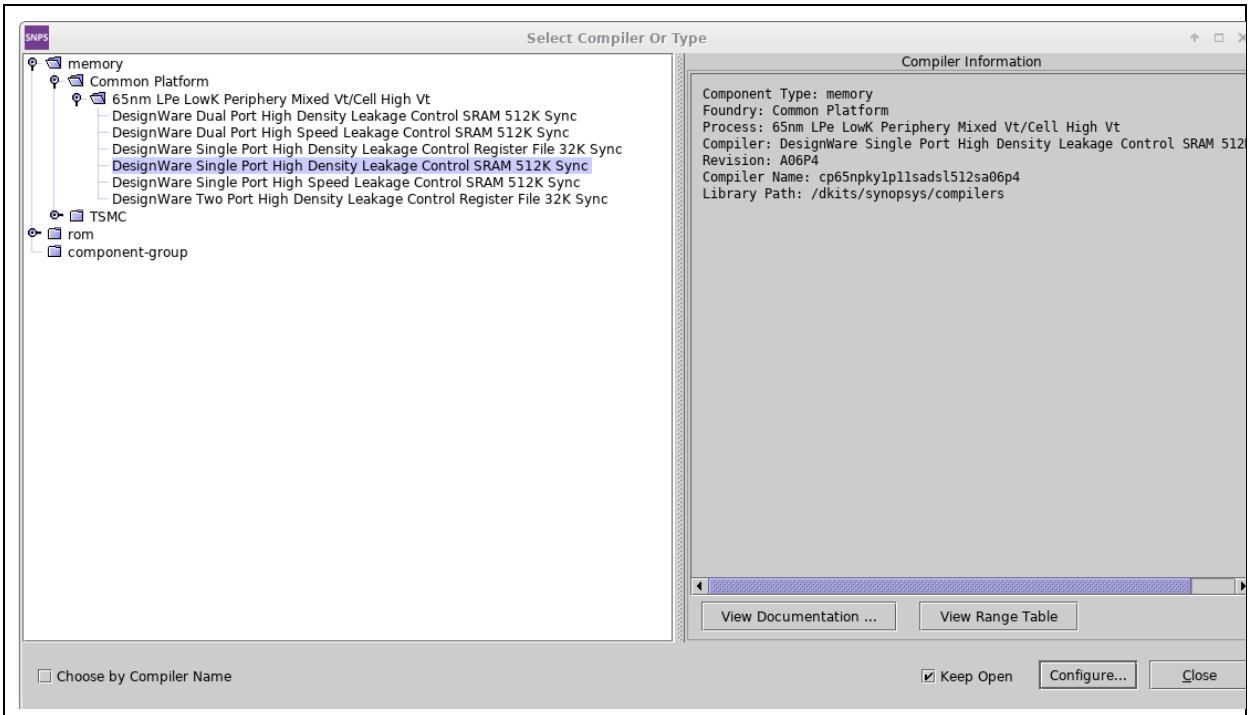


7. Check the message panel of the GUI. It should print a list of INFO about glb files in the /dkits/synopsys/compilers/ folder being read.

```
INFO              [Integrator,       CFG-018]:       Reading       file
'/dkits/synopsys/compilers/cp65npks1p10aspv201msa05p1/cp65npks1p10a
spv201msa05p1.glb' ...
INFO              [Integrator,       CFG-018]:       Reading       file
'/dkits/synopsys/compilers/cp65npky1p11asdrl32ksa04p3/cp65npky1p11a
sdrl32ksa04p3.glb' ...
INFO              [Integrator,       CFG-018]:       Reading       file
'/dkits/synopsys/compilers/cp65npky1p11sadsl512sa06p4/cp65npky1p11s
adsl512sa06p4.glb' ...
INFO              [Integrator,       CFG-018]:       Reading       file
'/dkits/synopsys/compilers/cp65npky1p11sassl512sa05p2/cp65npky1p11s
assl512sa05p2.glb' ...
INFO              [Integrator,       CFG-018]:       Reading       file
'/dkits/synopsys/compilers/cp65npky2p11asdrl32ksa05p2/cp65npky2p11a
sdrl32ksa05p2.glb' ...
INFO              [Integrator,       CFG-018]:       Reading       file
'/dkits/synopsys/compilers/cp65npky2p22sadsl512sa04p4/cp65npky2p22s
adsl512sa04p4.glb' ...
INFO              [Integrator,       CFG-018]:       Reading       file
'/dkits/synopsys/compilers/cp65npky2p22sassl512sa04p1/cp65npky2p22s
assl512sa04p1.glb' ...
```

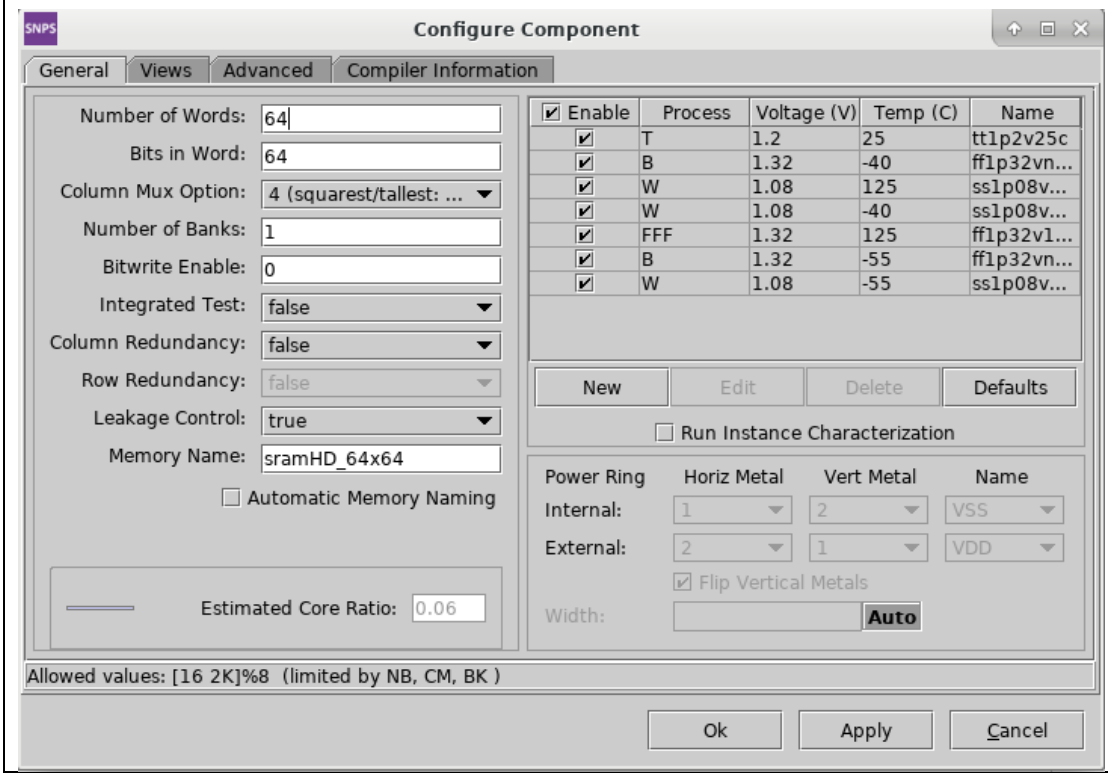
8. Click on  and expend the part related to Common Platform[7] and 65nm LPe LowK Periphery Mixed Vt/Cell High Vt.
9. Select the Single Port High Density Leakage Control SRAM 512K Sync (make sure you select the compiler name cp65npky1p11sadsl512sa06p4)
10. Click Configure

7

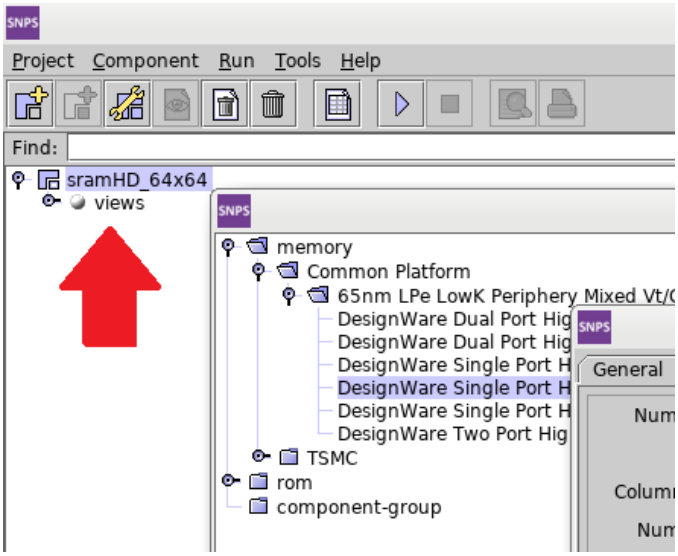


11. Fill the Configure Component window as follows. Make sure you properly untick “automatic memory naming” and call it sramHD_64x64. Make it 64 words of 64 bits with a squarest aspect ratio.

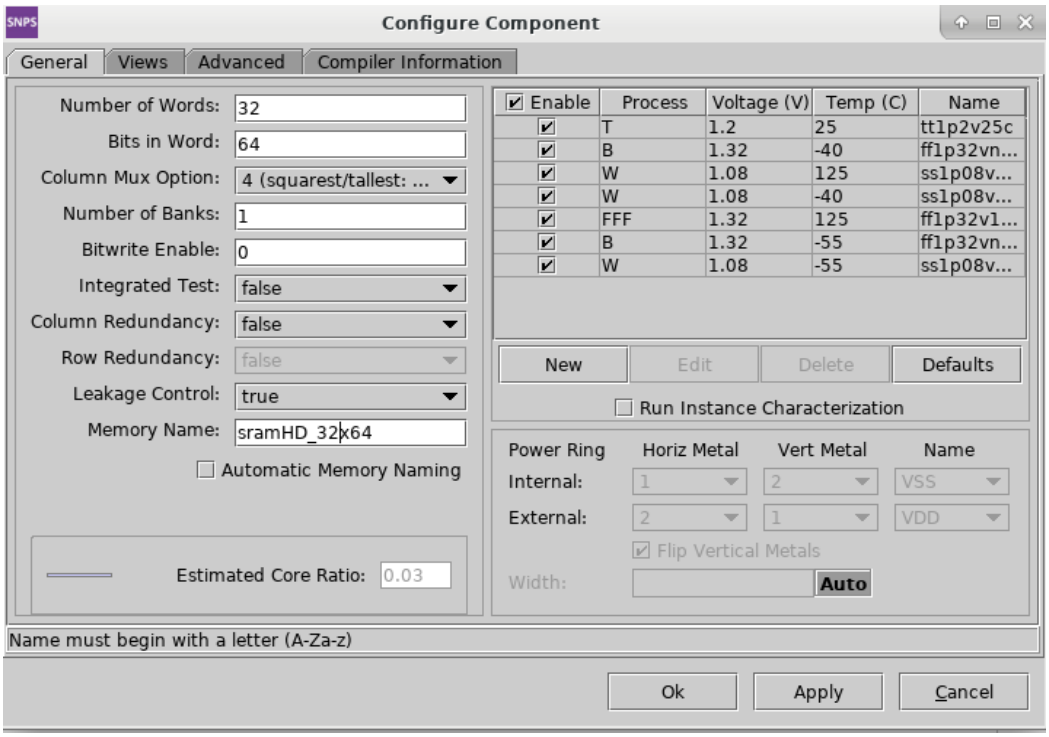
Note all the corners. You could potentially create new corners if you wanted to use the memory in different conditions. Note that generally, memory compilers do not provide arbitrary parameters, but only parameters in a given range.



12. Press Apply. And see the sramHD_64x64 appear in the main window of embedit.

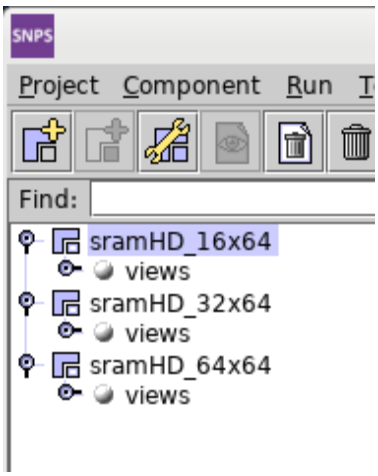


13. Change the numbers of word to 32, change the name to sramHD_32x64
14. Press Apply



Note that you can change more advanced parameters in the panel "views" and "advanced". Have a look there, but keep everything as set by default.

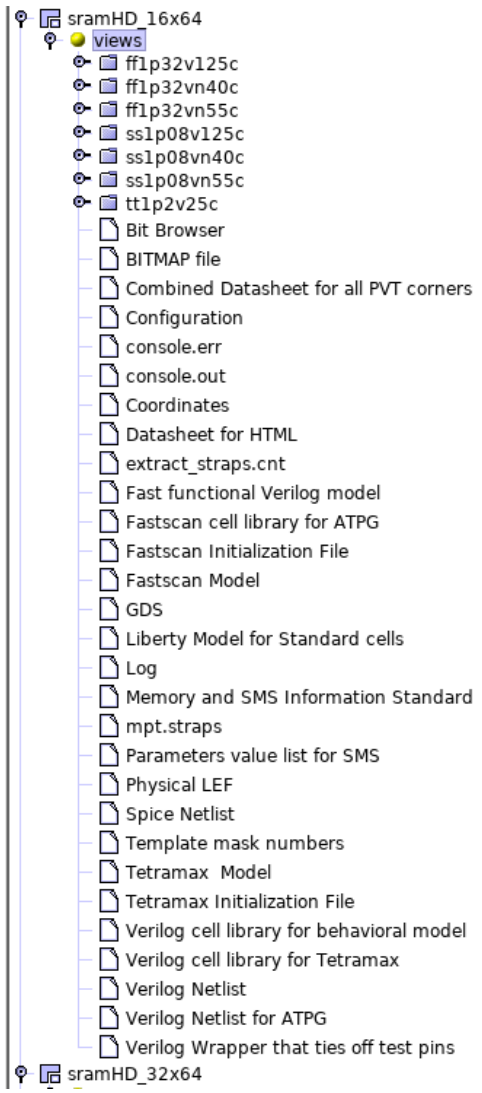
15. Do the same for 16words and call it sramHD_16x64
16. Press OK to close the window.
17. Close the select type of compiler window by clicking "close"
18. You should now see the following :



19. Press  or Run>Generate All

```
INFO [Integrator, IAPI-011]: Checking sramHD_16x64
INFO [Integrator, IAPI-011]: Checking sramHD_32x64
INFO [Integrator, IAPI-011]: Checking sramHD_64x64
INFO [Integrator, IAPI-001]: Running "GENERATE" on component "sramHD_16x64".
INFO [Integrator, IAPI-001]: Running "GENERATE" on component "sramHD_32x64".
INFO [Integrator, IAPI-001]: Running "GENERATE" on component "sramHD_64x64".
INFO [Integrator, IAPI-010]: Finished "GENERATE" on "sramHD_16x64". Status: Successful
INFO [Integrator, IAPI-010]: Finished "GENERATE" on "sramHD_32x64". Status: Successful
INFO [Integrator, IAPI-010]: Finished "GENERATE" on "sramHD_64x64". Status: Successful
```

The tool will run for a little while and generate the three memories as asked. While it runs, start reading the rest of the document and section 1.3. And come back here once finished.



20. Expend the views in one of the generated memories. Note all the files and folders that have been created by the compiler.

QUESTION 2-1 : what's the difference between a single port and a dual port memory ?

QUESTION 2-2: What's the difference between the SRAM and a register file ? From a physical standpoint. Check the available parameters in the compiler for a "High Density Leakage Control" SRAM and Register file. Then rephrase it with your own words the comment below.

- It is important at this point, if you come from a computer science background, or have had experience with architectural design, to "align your vocabulary". The vocabulary difference between circuit-level memory designers and architects is deep. The motivation behind these naming resides in the fact that SRAM are generally larger then Register Files. So, register files are generally adapted for small and fast memories and will become suboptimal for larger ones. And conversely for SRAM. The documentation of the cp65npky1p11asdrl32ksa04p3 is available on moodle. Check it out (specifically section

2.3.6) compare it to the same section in the cp65npky2p22sadsl512s documentation – alternatively you could click on “View Range Table” in the compiler GUI.

2.3. EXPLORING THE OUTPUT FILES

A memory compiler generate the files which are needed in the design process. As a designer, in order to include a memory, you need :

- A Verilog (.v) or Vital (.vhd) file to simulate the behavioural of the memory in your design.
- Note that vital is getting more and more deprecated.
- A liberty (.lib) file which contains the timing information from the memory, for the timing and power analysis.
- A library exchange format (.lef) which contains the physical specifications of the memory and its pins positions.
- You could also get other files such as test models (remember EE530?).

These are call the Front end views. Most of the agreements you may get with an IP provider do only involve front end views as these do not contain the details of the design and allow a company to share IPs without taking too many risks on their design knowhow.

The following views are generally more sensitive and not gracefully shared by IP providers without strong NDAs and associated liabilities.

- GDS views. The detailed layout of the IP that could be imported in a layout editor (such as virtuoso).
- Spice netlists. Containing the detailed spice netlist of the design with all the transistor sizes.

Let’s first explore the spice netlist :

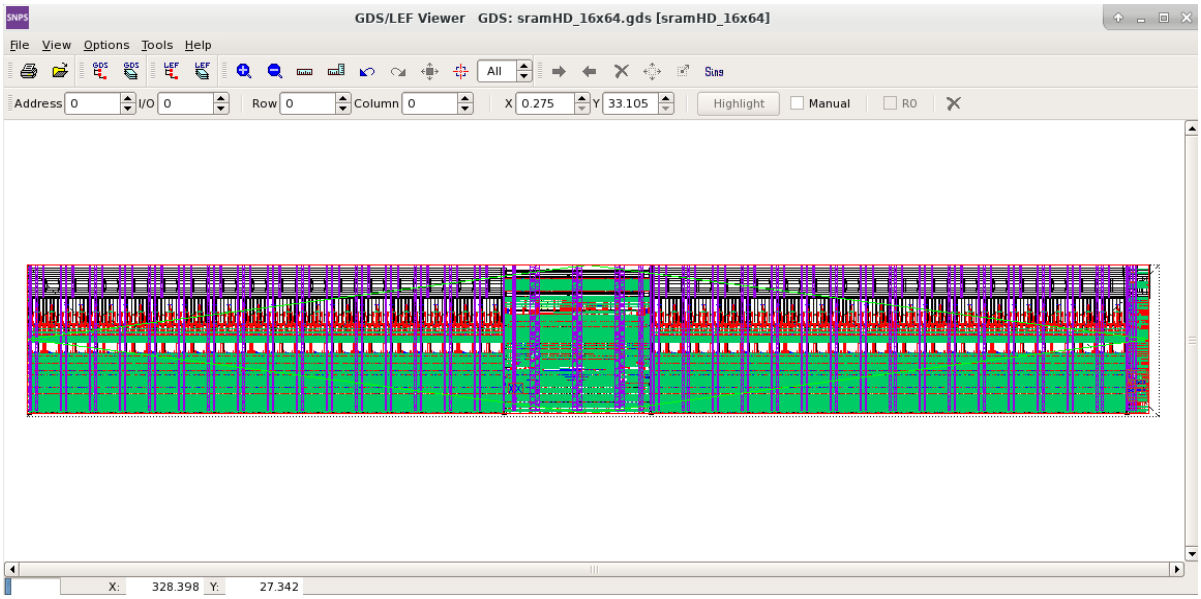
1. Double click on the Spice Netlist and scroll through it. It should remind you something from phase 1.

QUESTION 2-3: scroll through the netlist and find devices starting with MP and MN. What do these correspond to ?

- Mos transistors in a netlist are generally syntaxed as follows : InstanceName Drain Gate Source Bulk PDKdeviceName Parameters(W,L etc.)

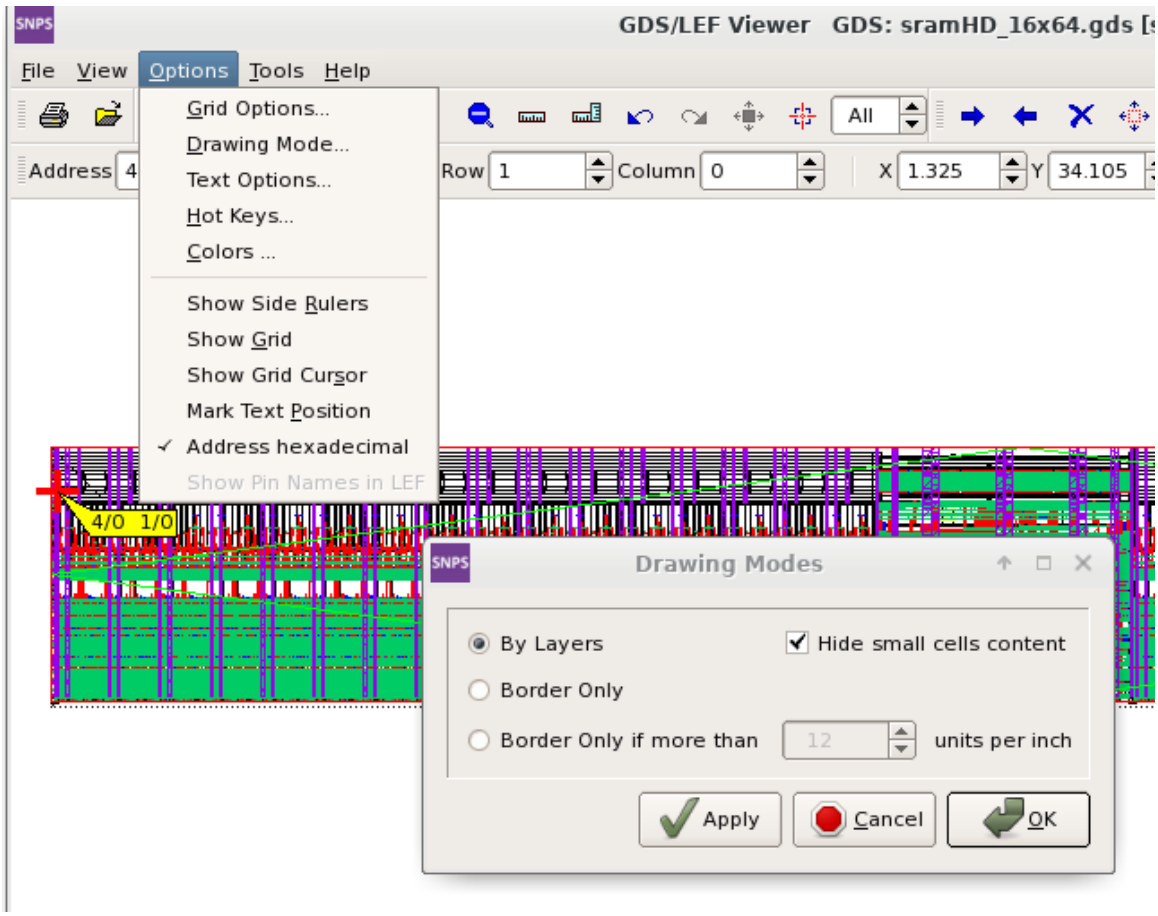
Let’s then explore a the GDS :

1. Double click on the GDS in the list
2. A new window opens, resize to make it easy to read. This is a GDS viewer.



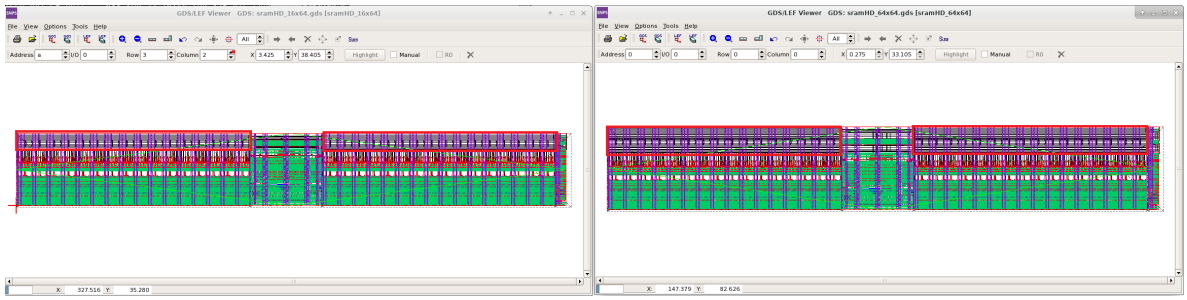
Use the  to fit the view to the screen.

3. Click on Options>Drawing Mode and make sure that the drawing mode is “By Layer”
4. Press OK



5. You can zoom in by left-clicking and maintaining to draw the zone to zoom in.

6. Explore the design, and zoom down to the transistor-level layout. To zoom out, use the fit button.
7. Note that we did build a quite small SRAM memory. Compare the layout of the 16x64 , 32x64 and 64x64 SRAM memories.

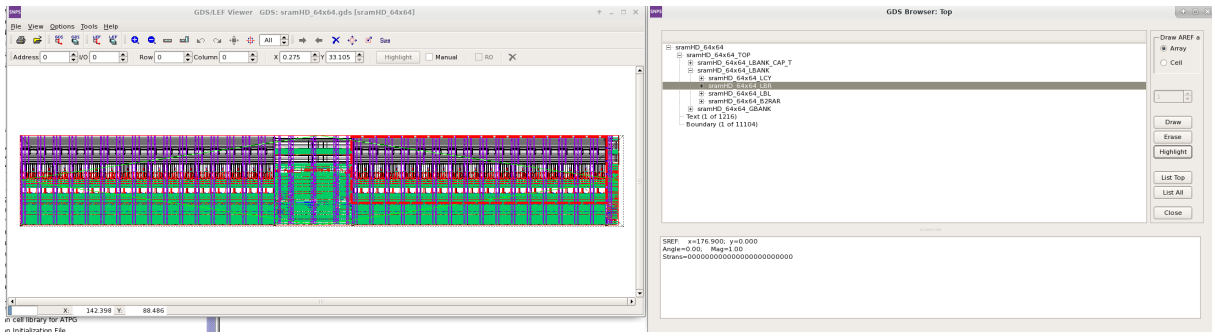


In the screenshots, the highlighted area in red is the actual subarray.

8. Keep the 64x64 GDS opened, and click in the 

button to open the hierarchy of the GDS.
9. Put side-to-side the gds viewer and the gds browser. And dive in the hierarchy.
10. Click on + on sramHD_64x64 then on sramHD_64x64_TOP
11. Alternatively select the three blocks inside sramHD_64x64_TOP and click on Highlight.

ℹ If you click on Erase by mistake, you select back the top cell and click on Draw to print it back.

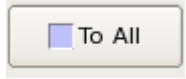


12. Find the actual memory array.
13. Find a bitcell and zoom in it.

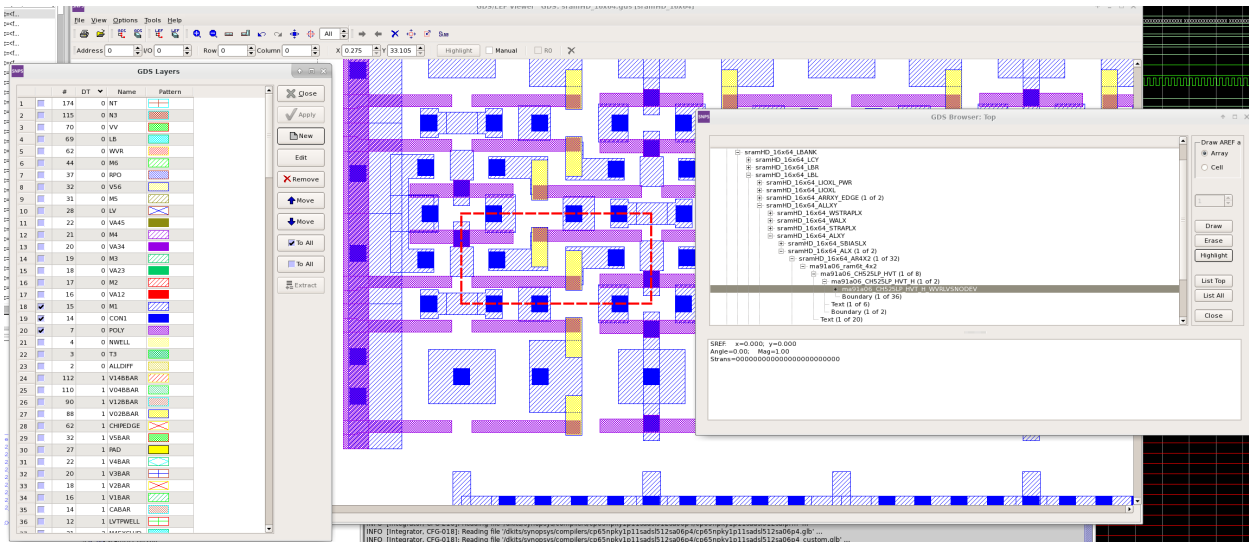
QUESTION 2-4: measure approximatively the size of the bitcell using the ruler. Units for the ruler are in um.

QUESTION 2-5 : The view of the bitcell is layout, as we did in phase 1. From the layout of one bitcell, by only representing the layers as described in the next comment, infer the schematic of the bitcell and draw it in your report. Are there things you see in this layout which seem to break the DRC rules you had to respect in the layout of phase 1 ?

ℹ *You can use the GDS layer button to enable/disable layers in order to ease the view. You could for e.g. order the layer by DT and only keep CON1, POLY, M1 NWELL and ALLDIFF visible. Note the layer called CAREC which is the special layer used for rectangular contacts in this technology. Add the layer CAREC as well.*

- *You can add and remove layers by ticking them in the "GDS Layers" panel accessible through the*  *button. To remove all the layers press*  *then*  *. Then tick the layers you want to add, and press Apply again.*
- *The NWELL layer sometimes stays over the rest, so try adding it and removing it to identify where the PMOS are.*
- *Orders of magnitude for the size of an SRAM cell in this technology node can be found in https://en.wikichip.org/wiki/65_nm_lithography_process*

Here is a screenshot of the bitcell, with one way to access one bitcell in the hierarchy, and one example of GDS layers panel with M1 Con1 Poly and CAREC activated. Here you would like to add ALLDIFF to find the transistors.



Once you are done, you can for now close integrator.

QUESTION 2-6 : Now that you have played with a memory compiler. Comment on the memory size you have been using. Considering the area ratio inside the memories you generated. Does it make sense to use this family of SRAM memories for such a small memory array?

3. MAKE YOUR DESIGN READY FOR A SRAMHD_16X64 MEMORY

Let's now solve the memory area bottleneck problem by replacing the sramHD_64x64 from the top_32b block, by a sramHD_16x64.

3.1. MODIFY THE TOP_32B

Let's modify the top_32b to have a version of it ready for future explorations.

1. With your terminal move to the HDL/RTL/
2. Make a copy of the file top_32b.vhd

```
> cp top_32b.vhd top_32b_16x64bitMEM.vhd
```

3. Open the newly created `top_32b_16x64bitMEM.vhd` with gedit or any text editor of your choice. Modify lines 159 to 191 in the following way :

```
MEM0 : component sramHD_16x64
```

```
    port map (
        Q  => data_out_mem0,
        ADR   => addr_mem0(3 downto 0),
        D  => data_in_mem0,
        WE => we_mem0,
        ME => en_mem0,
        CLK  => clk,
        TEST1 => '0',
        RME  => '0',
        RM => (others => '0'));
MEM1 : component sramHD_16x64
    port map(
        Q  => data_out_mem1,
        ADR   => addr_mem1(3 downto 0),
        D  => data_in_mem1,
        WE => we_mem1,
        ME => en_mem1,
        CLK  => clk,
        TEST1 => '0',
        RME  => '0',
        RM => (others => '0'));
MEM2 : component sramHD_16x64
    port map(
        Q  => data_out_mem2,
        ADR   => addr_mem2(3 downto 0),
        D  => data_in_mem2,
        WE => we_mem2,
        ME => en_mem2,
        CLK  => clk,
        TEST1 => '0',
        RME  => '0',
        RM => (others => '0'));
```

4. Save the file and close it.

You should also update the alu32_pkg.vhd file as it does contain definitions of memories for the simulation.

1. Open the file alu32_pkg.vhd
2. Add the following lines before line 85 as already done in the file for the sramHD_32x64.

```
component sramHD_16x64
port(
        Q       :   out std_logic_vector (63 downto 0);
        ADR     :   in std_logic_vector (3 downto 0);
        D       :   in std_logic_vector (63 downto 0);
        WE      :   in std_logic;
        ME      :   in std_logic;
        CLK     :   in std_logic;
        TEST1   :   in std_logic;
        RME     :   in std_logic;
        RM      :   in std_logic_vector (3 downto 0));
  end component sramHD_32x64;
```

3. Save and close the file

3.2. COMPILE THE LIB INTO A DB FILE

Before proceeding with logic synthesis, you must generate the db for your memory. This step is needed as designCompiler does not understand liberty files, and requires the lib file to be compiled

in a db file. Also, Embedit does not generate db files for you, so you need to generate them using a tool called LibraryCompiler and its command line : lc_shell

1. Go in the LibraryCompiler folder
2. Run >lc_shell
3. Run the following commands :

```
lc_shell> read_lib ../EMBEDIT/EDALABS/compout/views/sramHD_16x64/ss1p08v125c/sramHD_16x64.lib
```

- *The tool will read the lib file and check its syntax. As the lib file has been generated by a tool, you may want to read the warnings it does issue, though, at this point there is not much you can do. Identifying errors, info and warnings generally gives you insights on the quality and age of the model you deal with.*
- *It should finish by* `Technology library 'sramHD_16x64_lib' read successfully`

```
lc_shell> write_lib sramHD_16x64_lib -format db -output ../EMBEDIT/EDALABS/compout/views/sramHD_16x64/ss1p08v125c/sramHD_16x64.db
Wrote the 'sramHD_16x64_lib' library to '.../EMBEDIT/EDALABS/compout/views/sramHD_16x64/ss1p08v125c/sramHD_16x64.db' successfully
```

Once you are done,

```
lc_shell> exit
Maximum memory usage for this session: 127.71 MB
CPU usage for this session:      2 seconds (  0.00 hours)
Elapsed time for this session:    571 seconds (  0.16 hours)
Thank you for using Library Compiler.

Thank you...
```

Now, you have your db file ready for when you want to run a synthesis with this new memory size. At this point, when this is needed, it will be your duty to update the tcl script with the path to this new IP.

You could, for e.g., run the following command, to copy this new sramHD_16x64 folder in your IP/Memories folder (in this example, we run it from the LibraryCompiler folder):

```
>cp -r ../EMBEDIT/EDALABS/compout/views/sramHD_16x64 ../IPS/MEMORIES/
```

4. Place and Route the Mapped Design with Innovus

In this step, you will learn how to place the synthesized design from DesignCompiler. Here, you should take a design that you generated in the previous step. The Verilog and SDC files should be taken from the SDC and RTL/GATE/ folders.

IMPORTANT DISCLAIMER : from 2020, Cadence released a new syntax for their tools, called Stylus (or Common UI). This User Interface (UI) replaces what's called Legacy UI. This disclaimer is important for two reasons :

1- If you had experience in the past with Innovus, you did most likely learn legacy UI. Switching to Common UI will require a bit of gymnastics, but is an investment for the future, as from now on, only Common UI will be formally supported by Cadence.
2- If you go in a company, any flow developed before 2020 will most likely still use Legacy UI. It is important to be aware that these two UI will continue to co-exist.

In practice :

1- Some commands are discontinued between legacy and common
2- Some new commands have been introduced
3- Syntax has been homogenized between all cadence tools
    a. Example of Legacy command :

```
globalNetConnect VDD -type pgpin -pin VDD -inst *
```

    b. example of Common UI command :

```
connect_global_net VDD -type pgpin -pin VDD -inst *
```

4- Even in the 2022 version of Innovus , the GUI does not fully support yet Common UI. Which means… we will not use the GUI commands in this lab ! (it's a good thing, trust me)

For the curious you, on moodle you will find a documentation called "Translation of Innovus Legacy UI Commands into Common UI (CUI) Commands" and which proposed examples of how to translate a flow from Legacy to Common UI.

4.1. BEFORE STARTING

Innovus, as DesignCompiler, is an EXTREMELY COMPLEX tool. It does take commands through the terminal, and has, as expected, a long documentation which you would be expected, as a designer, to dive through.

To cope with that, besides the translation guide, we also provide the following documentation on moodle :

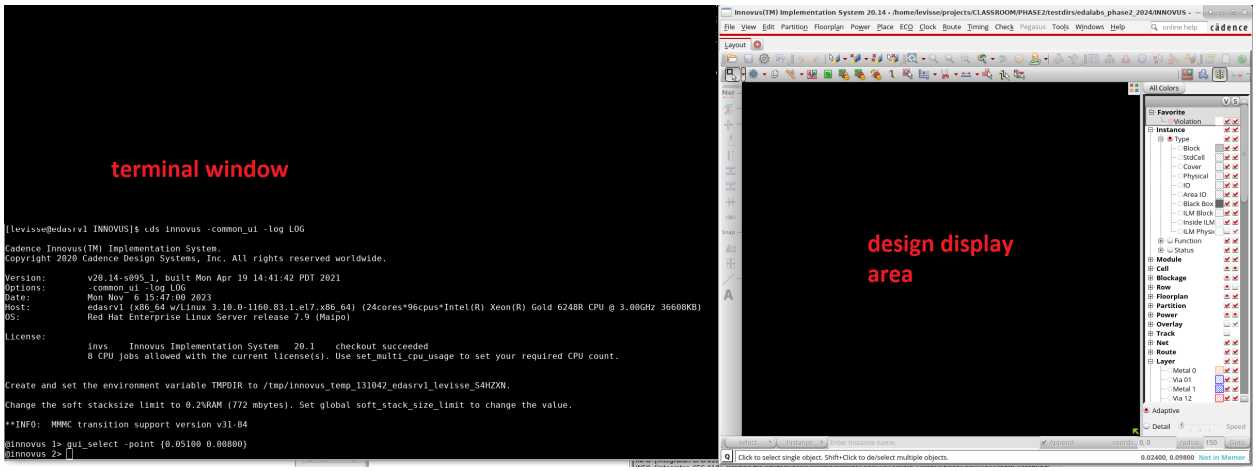
- The "*Innovus Stylus Common UI Text Command Reference"* which contains the details of all the commands used in innovus
- The "*Innovus Stylus Common UI User Guide"* which helps you selecting the good commands, and gives insights on how to use innovus and the UI.

For reference, the user guide is 2000+ pages and the command reference is 5000+ pages. Not something you would read before sleeping, though it can be extremely helpful during your designs.

4.2. STARTING INNOVUS

To start the Innovus tool, go to the `INNOVUS` directory and run the `innovus` command:

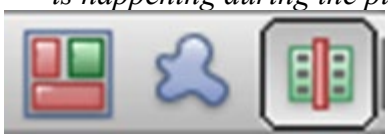
```
> cd INNOVUS
> innovus -common_ui -log LOG
```



The `-log` option redirects all log files to the `LOG` directory. Three log files are created: the file `innovus.cmd` records all entered commands and the files `innovus.log` and `innovus.logv` contain the respectively compact and verbose session logs.

Two things happen. The terminal becomes a @innovus> command line as it did with dc_shell. A window called Innovus Implementation System appears. You'll notice that this window looks a lot like virtuoso.

Make the Innovus console terminal large and visible enough, so you can better monitor what is happening during the place and route steps.



The design display area includes three different *design views* that you can toggle during a session (from left to right): the **Floorplan** view, the **Amoeba** view, and the **Physical** view:

- The Floorplan view displays the hierarchical module and block guides, connection flight lines, and floorplan objects, including block placement, and power/ground nets.
- The Amoeba view displays the outline of the modules and submodules after placement, showing physical locality of the module.
- The Physical view displays the detailed placements of the module's blocks, standard cells, nets, and interconnects.

When the AutoQuery box is enabled, the properties of the object below the cursor are automatically displayed.

There are a number of *binding keys* available (hit the key when the Innovus GUI is active). Some useful binding keys:

Binding key	Action
b	display the list of binding keys
d	(de)select or delete objects
f	zoom the display to fit the core area
k	create a ruler
K	remove last ruler displayed
q	display the object attribute editor form for the selected object click the left-button mouse to select an object shift-click to select or deselect an object
u	undo the last command
U	redo the last command
z	zoom-in 2x
Z	zoom-out 2x
Arrows	pan the display
CTRL+R	refresh the display

4.3. REQUIRED DATA

A number of specific data must be available or defined before using the Innovus tool.

- **Physical libraries**

  This includes information on the technology process to be used (routing layers (kind (metal, poly), name, pitch, and spacing), via/cut cells, core and pad site definitions, legal placement orientations) and abstract[8] information for every standard macro cell (type of site (core, pad, etc.), bounding box size, pins (name, kind, size, and position), blockages (routing obstructions)).
  Physical libraries are provided by a foundry or an IP provider as LEF (Library Exchange Format) files[9]. LEF files have the `.lef` extension. There could be two LEF files, one for the technology process and one for the macro cells, or a single one including both aspects.

- **Timing libraries**

  This includes the timing information for all macro cells (path delays, setup/hold times, etc.) and for various PVT (process/voltage/temperature) corners. Timing libraries will be used in the so-called *BC-WC (best case - worst case) timing analysis mode*: max/worst-case (resp., min/best-case) timing libraries are used for verifying setup (resp., hold) time

[8] The place and route does not need full geometrical layouts. i.e., you can work with LEF files and not gds as inputs.
[9] The Innovus help provides a LEF language reference.

constraints.
These libraries are the same as the ones used by DesignCompiler synthesizer, but here in the *Liberty* text (`.lib`) format. As opposed to designCompiler, innovus does not need the lib files to be compiled in db.

- **Parasitics extraction data**

  This includes technological data required to extract actual parasitics (RC) values for wires in the design. There could either be capTables (.capTbl) or directly extracted from the technology files (qrcTechFile). Generally nodes older than 65 would rely on captables while more advanced nodes will consider the more precise qrc files.

- **Gate-level netlist**

  This is the netlist to be placed and routed defined as a Verilog (`.v`) file. The file has been generated during logic synthesis and is placed in the HDL/GATE folder.

- **Timing constraints**

  This relates to the constraints used in logic synthesis. A `.sdc` constraint file has been generated at that step that can be used, although only timing commands (clock specification, timing delays) are supported.

  The SDC file generated in synthesis defines typical conditions. Place and route allows you more optimization through buffer insertion and clock tree synthesis, thereby, you could overconstrain the required timings by, say, 10%, to exploit possible further timing optimizations during place and route. We propose you one way to do it at the end of section.

4.4. DEFINING YOUR DESIGN ENVIRONMENT

The first step to do, as for designCompiler, is to define the working environment.

In the cadence environment, this is called the view definition, and it does define what's called the MMMC, i.e., Multi Mode Multi Corner.

1. Go in the INNOVUS/BIN/ folder and find the "default_mmmc.view" file.
2. You will find also a copy of this file identified with the features of your design : svt, mem64, 4.5ns top32_mmmc_svt_mem64_4p5ns.view
3. Open this file with a text editor

```
>gedit top32_mmmc_svt_mem64_4p5ns.view
```

We define this file for you already as it can be a bit complex to setup. Though, you must understand it as you will need to update it in order to use it.

Disclaimer. As usual, there is no absolute rule on how to deal with scripts in IC design. Here, we propose to use a MMMC template which have been used in EPFL for circuit design projects.

The approach described in this section is quite hierarchical and allows for some flexibility when adding or removing corners. Indeed, it becomes easy to add a new standard cell library, a new corner, or a new constraint file.

4.4.1. DEFINTION OF THE TECHNOLOGY CORNERS

first check lines 12 to 15

```
set     cap_tbl(rcbest)         /dkits/gf/65nm/65LPe/pdk/65LPe/PEX/QRC/EDA-CAD-65N-
EX038/Rev9/cmos10lpe_6_00_01_00_LB/cmos10lpe_FuncCmin_6_00_01_00_LB_effective/qrc
TechFile


set      cap_tbl(rcworst)       /dkits/gf/65nm/65LPe/pdk/65LPe/PEX/QRC/EDA-CAD-65N-
EX038/Rev9/cmos10lpe_6_00_01_00_LB/cmos10lpe_FuncCmax_6_00_01_00_LB_effective/qrc
TechFile


set      cap_tbl(typical)       /dkits/gf/65nm/65LPe/pdk/65LPe/PEX/QRC/EDA-CAD-65N-
EX038/Rev9/cmos10lpe_6_00_01_00_LB/cmos10lpe_nominal_6_00_01_00_LB_effective/qrcT
echFile


set cap_tables {rcworst rcbest typical}

foreach rc $cap_tables {
   create_rc_corner -name corner_${rc} \
            -qrc_tech $cap_tbl($rc) \
            -T {25}
}
```

Here we first define the technology files for the metal levels. Note that relative paths are not supported by the view files.

Note that we use the qrcTechFile which is a compiled version of the parasitic. This file can also be used for parasitic extraction in virtuoso.

QUESTION 4-1 : what do these 3 corners correspond to ?

4.4.2. DEFINITION OF THE IPS CORNERS

check lines 30 to 32

```
set lib_std(wc)
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/s
vt/5.00a/liberty/ccs/cp65npksdst_ss1p08v125c.lib


set lib_std(bc)
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/s
vt/5.00a/liberty/ccs/cp65npksdst_ff1p32vn40c.lib

set lib_std(tc)
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/s
vt/5.00a/liberty/ccs/cp65npksdst_tt1p2v25c.lib
```

Here we define the three corners for the standard cell library with the svt flavour as used during synthesis. If you intend to use lvt and hvt, you would need to add them there.

Check lines 37 to 39

```
set lib_mem(wc)  ../IPS/sramHD_64x64/ss1p08v125c/sramHD_64x64.lib
set lib_mem(bc)  ../IPS/sramHD_64x64/ff1p32vn40c/sramHD_64x64.lib
set lib_mem(tc)  ../IPS/sramHD_64x64/tt1p2v25c/sramHD_64x64.lib
```

Same as for the standard cells, we define here the lib file corresponding to the sramHD_64x64 memory.

Then, lines 42 to 48 the lib files are being clustered together per corner in a library_set.

```
set libsets {wc bc tc}
foreach libset $libsets {
    create_library_set -name libset_${libset} -timing [list \
    $lib_std($libset) \
    $lib_mem($libset)
    ]
}
```

4.4.3. DEFINITION OF THE CONSTRAINTS

Here we import the SDC file containing the design constraints. This is where the clock is being defined for e.g. The design constraints are being matched to all the libsets defined before. i.e., here we define a single constraint_mode for all the corners.

Lind 54 to 56

```
foreach libset $libsets {
    create_constraint_mode -name constraint_${libset} -sdc_files
{../DesignCompiler/SDC/DATE/TIME/postSynthesis_svt_clk_4.5.sdc}
}
```

Here you are expected to update the XXXX and DATE/TIME to match with the one of your design. Also make sure that the SDC file name matches.

4.4.4. DEFINE THE DELAY CORNERS

Then we match the capacitance corners defined before, with the timing conditions from the standard cell libraries.

Line 58 to71

```
foreach libset $libsets {
    create_timing_condition -name timing_condition_${libset} -
library_sets [list libset_${libset}]
}

set cap_map {rcworst rcbest typical}

# association of the capacitance corners (called cap_map) with the
timing conditions and constraints in the delay corner
foreach libset $libsets rc $cap_map {
    create_delay_corner -name delay_corner_${libset} \
        -early_timing_condition [list timing_condition_${libset}] \
        -late_timing_condition [list timing_condition_${libset}] \
        -early_rc_corner corner_${rc} \
        -late_rc_corner corner_${rc}
}
```

4.4.5. DEFINITION OF THE HOLD AND SETUP VIEWS

First we define the name of the variables being used. Note that here, we associate the hold corner to bc only and the setup corner to wc only.

Here, one may want to define much more than only 1 corner per check. Various voltages, temperatures could make sense to be added. In some conditions temperature inversion effects force the designers to add more than one corner.

Line 77 to 81

```
set hold_lib_corners {bc}
set setup_lib_corners {wc}

set setup_views {}
set hold_views {}
```

QUESTION 4-2: why is the BC corner (ff, 1.32V, -40C) being used in the hold view ?

QUESTION 4-3 why is the WC corner (ss, 1.08V, -125C) being used in the setup view ?

From line 83 to 102

```
# for each hold_lib_corner we define an analysis view called
analysis_hold_$(corner)
foreach corner $hold_lib_corners {
    create_analysis_view -name analysis_hold_${corner} \
                         -constraint_mode constraint_${corner} \
                         -delay_corner delay_corner_${corner}
    lappend hold_views analysis_hold_${corner}
}
# for each setup_lib_corner we define an analysis view called
analysis_setup_$(corner)
foreach corner $setup_lib_corners {
    create_analysis_view -name analysis_setup_${corner} \
                         -constraint_mode constraint_${corner} \
                         -delay_corner delay_corner_${corner}
    lappend setup_views analysis_setup_${corner}
}

# we define the set_analysis_view and associate setup_views to -setup
and hold_views to -hold
# this approach is extremely scalable and allows for the definition
of a large range of corners used during hold and setup.
# for the example given here, it could seem like a bit of overdoing
though.
set_analysis_view -setup $setup_views \
                  -hold $hold_views
```

We now define the views to be considered during hold and setup checks.

4.5. IMPORT THE MMMC AND INITIALIZE THE DESIGN ENVIRONMENT

4.5.1. IMPORT THE MMMC

First we want to import the MMMC file which we just read. Let's import it and check the corresponding logs.

```
Innovus > read_mmmc BIN/top32_mmmc_svt_mem64_4p5ns.view
Note that innovus does read the commands you import starting them with a @

#@ Begin verbose source (pre):
@file 1: # Version:1.0 MMMC View Definition File
@file 2: # Do Not Remove Above Line
@file 3: #this script has been written and prepared by Alexandre Levisse in oct2023 -
alexandre.levisse@epfl.ch
@file 4: #this script is designed as an example for the labs on EDA for innovus.
@file 5: #feel free to use it as a basis for your designs in EPFL
```

```
@file 6:
@file 7: # mmmc files do not accept relative paths, thereby, in the rest of the file, the
paths are using EPFL's edadk file organization. feel free to update them with your environment.
@file 8: # definition of the technology corners
@file 9: # here the qrcTechFile is being used. One could use a capTable in 65nm if available
- not available for this technology
@file 10: # definition of the different metal corners through variables
@file 11: # 3 corners are being selected rcbest, rcworst and typical.
...
}
@file 97:
@file 98: # we define the set_analysis_view and associate setup_views to -setup and hold_views
to -hold
@file 99: # this approach is extremely scalable and allows for the definition of a large range
of corners used during hold and setup.
@file 100: # for the example given here, it could seem like a bit of overdoing though.
@file 101: set_analysis_view -setup $setup_views \
                  -hold $hold_views
#@ End verbose source: BIN/top32_mmmc_svt_mem64_4p5ns.view
```

Until this point innovus was simply reading and printing the file

```
Reading                    libset_wc                    timing                    library
'/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp
65npksdst_ss1p08v125c.lib' ...
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_TIEDIN_1'. The cell will only
be                 used                 for                 analysis.                (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib)
...
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_DCAP32'. The cell will only
be                 used                 for                 analysis.                (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib)
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_DCAP16'. The cell will only
be                 used                 for                 analysis.                (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib)
```

Here innovus complains about some cells not having a defined logic functionality. Think about what TIE cells and DCAP cells are. Does that make sense ?

```
**WARN: (TECHLIB-1435):      For dc_current table defined in cell 'SEN_FDNRBSBQ_2' and pin
'Q', the current values are not monotonically decreasing in the range '-0.00339' to '-
0.003388'            for            'input_voltage'            '0.702000'.            (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib, Line 4281393)
**WARN: (TECHLIB-1435):      For dc_current table defined in cell 'SEN_FDNRBSBQ_2' and pin
'Q', the current values are not monotonically decreasing in the range '-0.005477' to '-
0.005426'            for            'input_voltage'            '0.756000'.            (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib, Line 4281393)
**WARN: (TECHLIB-1435):      For dc_current table defined in cell 'SEN_FDNRBSBQ_2' and pin
'Q', the current values are not monotonically decreasing in the range '-0.007974' to '-
0.007907'            for            'input_voltage'            '0.810000'.            (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib, Line 4281393)
...
**WARN: (TECHLIB-1435):      For dc_current table defined in cell 'SEN_EO3_6' and pin 'X',
the current values are not monotonically decreasing in the range '-0.231' to '0.1281' for
'output_voltage'                          '0.486000'.                          (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib, Line 4022900)
**WARN: (TECHLIB-1435):      For dc_current table defined in cell 'SEN_EO3_6' and pin 'X',
the current values are not monotonically decreasing in the range '-0.2372' to '0.1214' for
'output_voltage'                          '0.540000'.                          (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ss1p08v125c.lib, Line 4022900)
Message <TECHLIB-1435> has exceeded the message display limit of '20'. Use 'set_message -
no_limit -id list_of_msgIDs' to reset the message limit.
```

Some more complains about some device definition in the lib file. At this point you can ignore these. Though remember that in real life, you must

```
make sure that you understand what all the warnings mean. Because otherwise
you could be in a situation where your results are incorrect because of
some wrong assumption taken by the tool


Read 1128 cells in library 'cp65npksdst_ss1p08v125c'
Reading                        libset_wc                        timing                        library
'/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/IPS/sramHD_64x64/ss1p
08v125c/sramHD_64x64.lib' ...
Read 1 cells in library 'sramHD_64x64_lib'

Here Innovus acknowledges the read of the stdcells and memories
defined in the mmmc file for the libset_wc case.
Then it does the same for libset_bc


Reading                        libset_bc                        timing                        library
'/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp
65npksdst_ff1p32vn40c.lib' ...
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_TIEDIN_1'. The cell will only
be                    used                    for                    analysis.                    (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ff1p32vn40c.lib)
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_DCAP8'. The cell will only be
used                  for                  analysis.                  (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ff1p32vn40c.lib)
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_DCAP64'. The cell will only
be                    used                    for                    analysis.                    (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ff1p32vn40c.lib)
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_DCAP4'. The cell will only be
used                  for                  analysis.                  (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ff1p32vn40c.lib)
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_DCAP32'. The cell will only
be                    used                    for                    analysis.                    (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ff1p32vn40c.lib)
**WARN: (TECHLIB-302):       No function defined for cell 'SEN_DCAP16'. The cell will only
be                    used                    for                    analysis.                    (File
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt/5.00a/liberty/ccs/cp6
5npksdst_ff1p32vn40c.lib)
Read 1128 cells in library 'cp65npksdst_ff1p32vn40c'
Reading                        libset_bc                        timing                        library
'/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/IPS/sramHD_64x64/ff1p
32vn40c/sramHD_64x64.lib' ...
Read 1 cells in library 'sramHD_64x64_lib'
timing_initialized
```

QUESTION 4-4: Why do TIE and DCAP cells not have logic behaviour ? what are these cells doing ?

4.5.2. IMPORT THE PHYSICAL FOOTPRINT OF THE IPS BEING USED

Before starting the design we need to tell innovus the physical shape of all the cells.

In other words :

- The physical footprint of the standard cells and memory : area and shape
- Where are the pins located : inputs, outputs, vdd, grounds (cf what you did in the layout)
- How large are the pins

Though, for this, we do not use the GDS file, but a simplified version of it, inside a file called LEF.

- One of the reasons for using a LEF file instead of the GDS is that it allows to only show the interface pins. This is of value for companies which do not want to share the details of their design to their users.

- The second reason is that lef files are much lighter and more efficient to handle.

Note that besides the lef file for cp65nkpstdt you also import a file with a reference to m06f0f1. This is called the tech LEF. It does contain a simplified version of the DRC for a given metal stack (here we use m06f0f1), so that innovus knows how to connect wires.

- For some reason, the lef file of the sramHD is called plef for physical LEF, but this does not have any impact in your design.

```
Innovus                >                read_physical                -lef
{../IPS/STDCELLS/hd/base/svt/latest/lef/5.6/cp65npksdst_m06f0f1.lef
../IPS/STDCELLS/hd/base/svt/latest/lef/5.6/cp65npksdst.lef
../IPS/MEMORIES/sramHD_64x64/sramHD_64x64.plef}



Loading LEF file ../IPS/hd/base/svt/latest/lef/5.6/cp65npksdst_m06f0f1.lef ...
**WARN: (IMPLF-122):      The direction of the layer 'M2' is the same as
the previous routing layer. Make sure this is on purpose or correct
the direction of the layer. In most cases, the routing layers
alternate in direction between HORIZONTAL and VERTICAL.
```

Note how innovus reads about the routing directions.

```
Loading LEF file ../IPS/hd/base/svt/latest/lef/5.6/cp65npksdst.lef ...
Set DBUPerIGU to M2 pitch 400.

Loading LEF file ../IPS/sramHD_64x64/sramHD_64x64.plef ...
WARNING (LEFPARS-2007): NAMESCASESENSITIVE statement is obsolete in version 5.6 and
later.
The LEF parser will ignore this statement.
To avoid this warning in the future, remove this statement from the LEF file with
version 5.6 or later. See file ../IPS/sramHD_64x64/sramHD_64x64.plef at line 39.
```

This warning comes from the memory compiler and can be ignored

```
**WARN: (IMPLF-200):      Pin 'X' in macro 'SEN_TIEDIN_1' has no ANTENNAGATEAREA
value defined. The library data is incomplete and some process antenna rules will
not be checked correctly.
Type 'man IMPLF-200' for more detail.
```

This warning means that in the standard cell library, the cell SEN_TIEDIN_1 does not have some antenna protection on one of its outputs. This could lead to errors in DRC later on. At this point there is nothing we can do about it as this was designed by the IP provider, but it's a good example of something to be aware of.

```
viaInitial starts at Tue Nov  7 11:54:22 2023
viaInitial ends at Tue Nov  7 11:54:22 2023

##  Check design process and node:
##  Both design process and tech node are not set.
```

4.5.3. READ THE POST SYNTHESIS NETLIST

```
Innovus>              read_netlist              -top              top_32b
../HDL/GATE/XXXXX/postSynthesis_svt_clk_4.5.v
```

Update the XXXX with the path to your netlist

```
#% Begin Load netlist data ... (date=11/07 13:21:19, mem=646.5M)
```

```
*** Begin netlist parsing (mem=838.2M) ***
Created 1129 new cells from 4 timing libraries.
Reading netlist ...
Backslashed names will retain backslash and a trailing blank
character.
Reading verilog netlist
'../HDL/GATE/11072023/10h50m41s/postSynthesis_svt_clk_4.5.v'

*** Memory Usage v#1 (Current mem = 838.184M, initial mem = 292.316M)
***
*** End netlist parsing (cpu=0:00:00.1, real=0:00:00.0, mem=838.2M)
***
#% End Load netlist data ... (date=11/07 13:21:19, total
cpu=0:00:00.1, real=0:00:00.0, peak res=679.2M, current mem=679.2M)
Set top cell to top_32b.
Hooked 2258 DB cells to tlib cells.
Starting recursive module instantiation check.
No recursion found.
Building hierarchical netlist for Cell top_32b ...
*** Netlist is unique.
** info: there are 2317 modules.
** info: there are 3675 stdCell insts.
** info: there are 3 macros.

*** Memory Usage v#1 (Current mem = 907.109M, initial mem = 292.316M)
***
0
```

4.5.4. INITIALIZE THE DESIGN ENVIRONMENT

Let's initialize some variables being useful later on in the design

```
@innovus 6> set_db init_ground_nets VSS
1 VSS
@innovus 7> set_db init_power_nets VDD
1 VDD
```

Here we defined the names of the power nets being used in the design

```
@innovus 8> set_db timing_analysis_type ocv
1 ocv
@innovus 9> set_db timing_analysis_cppr both
1 both
```

Here we define the type of timing analysis being performed. OCV for on chip variations, and remove pessimism in the clock tree path (CPPR – Clock Path Pessimism Removal) in both hold and setup. This simplifies the clock tree design for this design which is simple.

```
@innovus 10> set_db design_process_node 65
##  Process: 65            (User Set)
##     Node: (not set)

##  Check design process and node:
##  Design tech node is not set.
```

```
Applying the recommended capacitance filtering threshold values for
65nm    process    node:    total_c_th=0,    relative_c_th=1    and
coupling_c_th=0.1.
     These values will be used by all post-route extraction engines,
including TQuantus, IQuantus and Quantus QRC extraction.
     Capacitance filtering mode(extract_rc_cap_filter_mode option of
the set_db) is 'relative_and_coupling' for all engines.
     The accuracy mode for post_route extract_rc_effort_level low
extraction will be set to 'high'.
     Default value for EffortLevel(extract_rc_effort_level option of
the set_db) in post_route extraction mode will be 'medium' if Quantus
QRC technology file is specified else 'low'.
1 65
```

Defining a process sets several parameters in innovus, and aligns the flow with the constraints associated with a 65nm technology process. It does select different types of RC extraction for coupling, threshold on R and C etc.

```
@innovus 11> set_db design_top_routing_layer M5
1 M5
@innovus 12> set_db design_bottom_routing_layer M2
1 M2
```

Here we define the Top and Bottom routing layers. In this flow, the tool will be allowed to route from M2 to M5.

Then, we initialize the design

```
Innovus > init_design


Set Default Net Delay as 1000 ps.
Set Default Net Load as 0.5 pF.
Set Default Input Pin Transition as 0.1 ps.


**WARN: (IMPFP-3961):     The techSite 'VLC_SITE_sramHD_64x64' has no related
standard cells in LEF/OA library. Cannot make calculations for this site type unless
standard cell models of this type exist in the LEF/OA library. If the SITE is not
used by the library you can ignore this warning or remove the SITE definition from
the LEF/OA to avoid this message.
Type 'man IMPFP-3961' for more detail.
```

This warning comes from the lef of the memory. You can ignore it at this point

```
Extraction setup Started
Initializing multi-corner RC extraction with 2 active RC Corners ...
Captable file(s) not specified in multi-corner setup. PreRoute extraction will use
technology file. For post_route extraction, default value for effort level would be
'medium' and effort level 'low' would not be allowed.
Generating auto layer map file.
Importing multi-corner technology file(s) for preRoute extraction...
/dkits/gf/65nm/65LPe/pdk/65LPe/PEX/QRC/EDA-CAD-65N-
EX038/Rev9/cmos10lpe_6_00_01_00_LB/cmos10lpe_FuncCmax_6_00_01_00_LB_effective/qrc
TechFile
Generating auto layer map file.
/dkits/gf/65nm/65LPe/pdk/65LPe/PEX/QRC/EDA-CAD-65N-
EX038/Rev9/cmos10lpe_6_00_01_00_LB/cmos10lpe_FuncCmin_6_00_01_00_LB_effective/qrc
TechFile
Generating auto layer map file.
Completed (cpu: 0:00:03.5 real: 0:00:04.0)
Set Shrink Factor to 1.00000
```

From the technology corners defined and the RC parasitic files (qrcTechFile) selected, the tool summarizes all the parameters.

```
Summary of Active RC-Corners :

 Analysis View: analysis_setup_wc
    RC-Corner Name          : corner_rcworst
    RC-Corner Index         : 0
    RC-Corner Temperature : 25 Celsius
    RC-Corner Cap Table   : ''
    RC-Corner PreRoute Res Factor         : 1
    RC-Corner PreRoute Cap Factor         : 1
    RC-Corner PostRoute Res Factor        : 1 {1 1 1}
    RC-Corner PostRoute Cap Factor        : 1 {1 1 1}
    RC-Corner PostRoute XCap Factor       : 1 {1 1 1}
    RC-Corner PreRoute Clock Res Factor   : 1 [Derived      from      postRoute_res
(effortLevel low)]
    RC-Corner PreRoute Clock Cap Factor   : 1 [Derived      from      postRoute_cap
(effortLevel low)]
    RC-Corner PostRoute Clock Cap Factor  : 1 {1 1 1}      [Derived            from
postRoute_cap (effortLevel low)]
    RC-Corner PostRoute Clock Res Factor  : 1 {1 1 1}      [Derived            from
postRoute_res (effortLevel low)]
    RC-Corner  Technology  file:  '/dkits/gf/65nm/65LPe/pdk/65LPe/PEX/QRC/EDA-CAD-
65N-
EX038/Rev9/cmos10lpe_6_00_01_00_LB/cmos10lpe_FuncCmax_6_00_01_00_LB_effective/qrc
TechFile'

 Analysis View: analysis_hold_bc
    RC-Corner Name          : corner_rcbest
    RC-Corner Index         : 1
    RC-Corner Temperature : 25 Celsius
    RC-Corner Cap Table   : ''
    RC-Corner PreRoute Res Factor         : 1
    RC-Corner PreRoute Cap Factor         : 1
    RC-Corner PostRoute Res Factor        : 1 {1 1 1}
    RC-Corner PostRoute Cap Factor        : 1 {1 1 1}
    RC-Corner PostRoute XCap Factor       : 1 {1 1 1}
    RC-Corner PreRoute Clock Res Factor   : 1 [Derived      from      postRoute_res
(effortLevel low)]
    RC-Corner PreRoute Clock Cap Factor   : 1 [Derived      from      postRoute_cap
(effortLevel low)]
    RC-Corner PostRoute Clock Cap Factor  : 1 {1 1 1}      [Derived            from
postRoute_cap (effortLevel low)]
    RC-Corner PostRoute Clock Res Factor  : 1 {1 1 1}      [Derived            from
postRoute_res (effortLevel low)]
    RC-Corner  Technology  file:  '/dkits/gf/65nm/65LPe/pdk/65LPe/PEX/QRC/EDA-CAD-
65N-
EX038/Rev9/cmos10lpe_6_00_01_00_LB/cmos10lpe_FuncCmin_6_00_01_00_LB_effective/qrc
TechFile'
Updating RC grid for preRoute extraction ...
Initializing multi-corner resistance tables ...
Default   value   for   post_route   extraction   mode's   extract_rc_effort_level
(extract_rc_effort_level option of set_db) changed to 'medium'.
*Info: initialize multi-corner CTS.
Reading                  timing                  constraints                  file
'/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompi
ler/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc' ...
Current (total cpu=0:14:12, real=1:29:00, peak res=1204.3M, current mem=1078.5M)
**WARN: (TCLCMD-1461):    Skipped    unsupported    command:    set_units    (File
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc, Line 8).

INFO       (CTE):       Reading       of       timing       constraints       file
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc completed, with 1 WARNING
WARNING          (CTE-25):          Line:          10          of          File
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc   :   Skipped   unsupported
command: set_max_area
```

```
WARNING (CTE-25): Line: 9 of File
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc : Skipped unsupported
command: set_operating_conditions


Ending "Constraint file reading stats" (total cpu=0:00:00.1, real=0:00:00.0, peak
res=1095.6M, current mem=1095.6M)
Current (total cpu=0:14:12, real=1:29:00, peak res=1204.3M, current mem=1095.6M)
Reading timing constraints file
'/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompi
ler/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc' ...
Current (total cpu=0:14:12, real=1:29:00, peak res=1204.3M, current mem=1095.6M)
**WARN: (TCLCMD-1461): Skipped unsupported command: set_units (File
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc, Line 8).

INFO (CTE): Reading of timing constraints file
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc completed, with 1 WARNING
WARNING (CTE-25): Line: 10 of File
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc : Skipped unsupported
command: set_max_area


WARNING (CTE-25): Line: 9 of File
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_2024/DesignCompil
er/SDC/11072023/10h50m41s/postSynthesis_svt_clk_4.5.sdc : Skipped unsupported
command: set_operating_conditions


Ending "Constraint file reading stats" (total cpu=0:00:00.0, real=0:00:00.0, peak
res=1095.9M, current mem=1095.9M)
Current (total cpu=0:14:12, real=1:29:00, peak res=1204.3M, current mem=1095.9M)
```

Here the tool lists all the cells available for the design to come

```
Total number of combinational cells: 954
Total number of sequential cells: 174
Total number of tristate cells: 0
Total number of level shifter cells: 0
Total number of power gating cells: 0
Total number of isolation cells: 0
Total number of power switch cells: 0
Total number of pulse generator cells: 0
Total number of always on buffers: 0
Total number of retention cells: 0
List of usable buffers: SEN_BUF_1 SEN_BUF_10 SEN_BUF_12 SEN_BUF_16 SEN_BUF_2
SEN_BUF_1P5 SEN_BUF_20 SEN_BUF_24 SEN_BUF_3 SEN_BUF_4 SEN_BUF_6 SEN_BUF_8
SEN_BUF_AS_1 SEN_BUF_AS_0P5 SEN_BUF_AS_10 SEN_BUF_AS_12 SEN_BUF_AS_16 SEN_BUF_AS_2
SEN_BUF_AS_1P5 SEN_BUF_AS_3 SEN_BUF_AS_4 SEN_BUF_AS_5 SEN_BUF_D_1 SEN_BUF_AS_6
SEN_BUF_AS_8 SEN_BUF_D_12 SEN_BUF_D_16 SEN_BUF_S_1 SEN_BUF_D_2 SEN_BUF_D_3
SEN_BUF_D_4 SEN_BUF_D_6 SEN_BUF_D_8 SEN_BUF_S_12 SEN_BUF_S_16 SEN_BUF_S_2
SEN_BUF_S_3 SEN_BUF_S_4 SEN_BUF_S_6 SEN_BUF_S_8
Total number of usable buffers: 40
List of unusable buffers:
Total number of unusable buffers: 0
List of usable inverters: SEN_INV_0P5 SEN_INV_0P65 SEN_INV_1 SEN_INV_0P8 SEN_INV_10
SEN_INV_12 SEN_INV_16 SEN_INV_1P25 SEN_INV_2 SEN_INV_1P5 SEN_INV_20 SEN_INV_24
SEN_INV_3 SEN_INV_2P5 SEN_INV_32 SEN_INV_4 SEN_INV_5 SEN_INV_6 SEN_INV_8
SEN_INV_AS_1 SEN_INV_AS_0P5 SEN_INV_AS_10 SEN_INV_AS_12 SEN_INV_AS_16 SEN_INV_AS_2
SEN_INV_AS_1P5 SEN_INV_AS_3 SEN_INV_AS_4 SEN_INV_AS_5 SEN_INV_AS_6 SEN_INV_AS_8
SEN_INV_S_1 SEN_INV_S_0P5 SEN_INV_S_12 SEN_INV_S_16 SEN_INV_S_2 SEN_INV_S_1P5
SEN_INV_S_3 SEN_INV_S_32 SEN_INV_S_4 SEN_INV_S_6 SEN_INV_S_8
Total number of usable inverters: 42
List of unusable inverters:
Total number of unusable inverters: 0
List of identified usable delay cells: SEN_DEL_L4_1 SEN_DEL_L4_2 SEN_DEL_L4_4
SEN_DEL_L4_8 SEN_DEL_L6_1 SEN_DEL_L6_2 SEN_DEL_L6_4 SEN_DEL_L6_8
```

```
Total number of identified usable delay cells: 8
List of identified unusable delay cells:
Total number of identified unusable delay cells: 0
```

4.5.5. INITIALIZE THE METRICS EXTRACTION

In the latest versions of innovus, it does embed some feature called metrics.

It does enable you do save the state of your design and some metrics in a html file which can be opened at the end of the flow.

Let's enable the metrics feature so that we can check the reports later.

```
innovus > enable_metrics -on
```

- What this function does is storing all the metrics of the design (area, slack, reports etc.) at a certain point in time. You will see that we will use some snapshot-related commands later on.
- This relies on 2 commands :

```
>push_snapshot_stack
>pop_snapshot_stack
```

Everything happening between these two commands will be tracked and saved in the final html file.

4.5.6. AUTOMATE THE FLOW

As for all the other tools you used until now, this step can be automatically done through a script written in the tcl language.

1. From the terminal, using the gedit tool, open the following files and understand their content :

```
INNOVUS/BIN/init.tcl
INNOVUS/BIN/top32_mmmc_svt_mem64_4p5ns.view
```

- These two files contain all the commands you did write before. From a new instance of innovus, you can reach back the point where we are here, by running the following commands.
- Note that init.tcl imports the .view file through the read_mmmc command, so sourcing the init.tcl file will also setup the mmmc.

For innovus, we propose a different way to write a tcl script. If you take a look at the BIN folder, you will notice several files, and a file called "fullflow.tcl" which contains source commands to all the other files. This way allows you to handle longer scripts in a more comfortable way. And ease the process of progressively go through the flow. Though, when making different versions of the same flow, it does then require the designer to make an organizational effort.

4.5.7. SAVE AND RESTORE THE DESIGN STATE

As opposed to dc_shell, where you noticed that restoring the ddc file was not that straightforward, innovus does provide a complete save/restore design function which allows you to fully save and restore the actual state of the tool at any point.

As the place and route is a fairly longer process than logic synthesis, being able to save and restore the tool state can actually be useful.

1. To save the state of a design, run the following command :

```
Innovus> write_db DB/tutorial/top_32b_INIT
#% Begin save design ... (date=11/08 11:26:36, mem=1133.9M)
% Begin Save ccopt configuration ... (date=11/08 11:26:36, mem=1137.0M)
% End Save ccopt configuration ... (date=11/08 11:26:36, total cpu=0:00:00.0, real=0:00:00.0, peak res=1137.8M, current mem=1137.8M)
% Begin Save netlist data ... (date=11/08 11:26:36, mem=1137.9M)
Writing Binary DB to DB/tutorial/top_32b_INIT/top_32b.v.bin in single-threaded mode...
% End Save netlist data ... (date=11/08 11:26:36, total cpu=0:00:00.0, real=0:00:00.0, peak res=1138.0M, current mem=1138.0M)
Saving symbol-table file ...
Saving congestion map file DB/tutorial/top_32b_INIT/top_32b.route.congmap.gz ...
% Begin Save AAE data ... (date=11/08 11:26:37, mem=1138.6M)
Saving AAE Data ...
% End Save AAE data ... (date=11/08 11:26:37, total cpu=0:00:00.0, real=0:00:00.0, peak res=1138.6M, current mem=1138.6M)
Saving preference file DB/tutorial/top_32b_INIT/gui.pref.tcl ...
Saving mode setting ...
Saving root attributes to be loaded post write_db ...
Saving global file ...
Saving root attributes to be loaded previous write_db ...
% Begin Save floorplan data ... (date=11/08 11:26:38, mem=1142.9M)
Saving floorplan file ...
% End Save floorplan data ... (date=11/08 11:26:38, total cpu=0:00:00.0, real=0:00:00.0, peak res=1143.4M, current mem=1143.4M)
Saving Drc markers ...
... No Drc file written since there is no markers found.
% Begin Save placement data ... (date=11/08 11:26:38, mem=1143.5M)
** Saving stdCellPlacement_binary (version# 2) ...
Save Adaptive View Pruning View Names to Binary file
% End Save placement data ... (date=11/08 11:26:38, total cpu=0:00:00.0, real=0:00:01.0, peak res=1143.7M, current mem=1143.7M)
% Begin Save routing data ... (date=11/08 11:26:39, mem=1143.7M)
Saving route file ...
*** Completed saveRoute (cpu=0:00:00.0 real=0:00:00.0 mem=1283.5M) ***
% End Save routing data ... (date=11/08 11:26:39, total cpu=0:00:00.0, real=0:00:00.0, peak res=1147.9M, current mem=1147.9M)
Saving property file DB/tutorial/top_32b_INIT/top_32b.prop
*** Completed saveProperty (cpu=0:00:00.0 real=0:00:00.0 mem=1286.5M) ***
Saving preRoute extracted patterns in file 'DB/top_32b_INIT/top_32b.techData.gz' ...
Saving preRoute extraction data in directory 'DB/tutorial/top_32b_INIT/extraction/' ...
% Begin Save power constraints data ... (date=11/08 11:26:39, mem=1149.1M)
```

```
% End Save power constraints data ... (date=11/08 11:26:39, total
cpu=0:00:00.0,    real=0:00:00.0,    peak    res=1149.2M,    current
mem=1149.2M)
Generated self-contained design top_32b_INIT
#% End save design ... (date=11/08 11:26:40, total cpu=0:00:01.5,
real=0:00:04.0, peak res=1153.6M, current mem=1153.6M)
*** Message Summary: 0 warning(s), 0 error(s)
```

Essentially here the tool is saving all the details of the design.

In the same way, from a new version of innovus (if you exit it and start it again), you can restore a db.

1. To restore the init state you just saved, from a new innovus instance

```
innovus > read_db DB/tutorial/top_32b_INIT.db
```

Note that here, innovus will go through the configuration of the mmmc file again, as this step is kind of mandatory. But restoring more advanced part of the design will be faster.

4.6. FLOORPLANNING THE DESIGN

The logic cells in the gate-level netlist obtained from synthesis will be placed in rows in the so-called *core area*. All physical cells in the standard cell library have the same height and different widths. They all have a power rail at their top and a ground rail at their bottom. Physical cells are abutted in a row so their power and ground rails connect between cells. The rows in the core area are further flipped so power and ground rails of neighboring rows coincide.

The floorplan defines the actual form, or aspect ratio, of the core area, the global and detailed routing grids, the rows to host the core cells and the I/O pad cells (if required), the area for power rings, the (pre)placement of blocks/macros, and the location of the corner cells (if required). After the design import, an initial default floorplan is displayed in the display area.

4.6.1. DEFINE THE FLOORPLAN

Run the following command :

```
@innovus 7> create_floorplan -site CP65_DST -core_size 980 110
20 20 20 20
**WARN: (IMPFP-3961): The techSite 'VLC_SITE_sramHD_64x64' has no
related standard cells in LEF/OA library. Cannot make calculations
for this site type unless standard cell models of this type exist in
the LEF/OA library. If the SITE is not used by the library you can
ignore this warning or remove the SITE definition from the LEF/OA to
avoid this message.
Type 'man IMPFP-3961' for more detail.
```

You can ignore this warning which comes from the memory macro which also has a SITE being defined. Though here we use the site reference of the standard cell library (see after).

```
**WARN: (IMPFP-325): Floorplan of the design is resized. All current
create_floorplan objects are automatically derived based on specified
new create_floorplan. This may change blocks, fixed standard cells,
existing routes and blockages.
```

Here we purposely create a suboptimal floorplan which you will need to update and optimize later on.

The create_floorplan command has many parameters. Here we use the -site CP65_DST which comes from the standard cell library. Line 11 of the lef file (cp65npksdst.lef) you will find the following syntax. This defines the size of the rows of standard cells. The 1.8um corresponds to the height of the standard cell track. Using a different standard cell height (for e.g. a HS cell) will require the -site to be updated accordingly.

```
# Placement site definition for this library.
SITE CP65_DST
  SYMMETRY Y ;
  CLASS core ;
  SIZE 0.2 BY 1.8 ;
END CP65_DST
```

The lef can be found in

```
IPS/STDCELLS/hd/base/svt/5.00a/lef/5.6/cp65npksdst.lef
```

The `-core_size` command is described in the innovus TCR document in the description of the create_floorplan command page 814.

It is documented as follows :

```
-core_size {w h left bottom right top}
Specifies the core size and the spacing, in micrometers, between the
core edge, which is the margin between the outside edge of the core
(head) box.
```

- `w`: Specifies the core box's width value.
- `h`: Specifies the core box's height value.
- `left`: Specifies the margin from the outside edge of the core box to the left.
- `bottom`: Specifies the margin from the outside edge of the core box to the bottom.
- `right`: Specifies the margin from the outside edge of the core box to the right.
- `top`: Specifies the margin from the outside edge of the core box to the top.

```
The following command creates floorplan by specifying a core size of
1000 x 1000, and the spacing between core edge to each die edge of
300.
```

Then, some examples are given.

In other words, here, we created a 980um wide by 110um tall box, with 20um space on all sides.

The corresponding floorplan can be observed in the design display area.

Click on the floorplan view  to print the design units. The floorplan is now visible

Note the three memory macros on the right. And on the left, the two largest blocks (I_MULT and I_ADDSUB).

4.6.2. REPORT THE FLOORPLAN UTILIZATION

At this point you can already report the projected density of your design. To make sure that your floorplan makes sense from a design standpoint.

```
innovus 9> check_floorplan -report_density
Checking routing tracks.....
Checking other grids.....
Checking FINFET Grid is on Manufacture Grid.....
Checking core/die box is on Grid.....
Checking snap rule ......
Checking Row is on grid......
Checking AreaIO row.....
Checking row out of die ...
Checking routing blockage.....
Checking components.....
Checking IO Pads out of die...
Checking constraints (guide/region/fence).....
Checking groups.....

Checking Preroutes.....
No. of regular pre-routes not on tracks : 0

Reporting Utilizations.....
```

Here the core utilization is projected to be 54% which is low enough to make sure the design can be routed. There is no absolute rule here, as everything depends on your design and floorplan, though consider that you should generally not go above 80% at this point.

```
Core utilization  = 54.142904
Effective Utilizations
**ERROR: (IMPSP-365): Design      has      inst(s)      with      SITE
'VLC_SITE_sramHD_64x64', but the floorplan has no rows defined for
this site. Any locations found for such insts will be illegal; create
rows for this site to avoid this.
Type 'man IMPSP-365' for more detail.
```

This error can be ignored at this point. You can use the “man IMPSP-365” command to read about it (we do it after in this code snippet).

```
Average module density = 0.542.
Density for the design = 0.542.
       = (stdcell_area 35621 sites (12824 um^2) + block_area 126507
sites (45542 um^2)) / alloc_area 298900 sites (107604 um^2).
Pin Density = 0.04940.
```

```
             = total # of pins 14766 / total area 298900.

*** Summary of all messages that are not suppressed in this session:
Severity  ID               Count  Summary
ERROR     IMPSP-365            1  Design has inst(s) with SITE '%s',
but t...
*** Message Summary: 0 warning(s), 1 error(s)

0
@innovus 15> man IMPSP-365
IMPSP-365(20.14)                                        IMPSP-365(20.14)


NAME
       IMPSP-365

SUMMARY
       Design  has  inst(s)  with  SITE  '%s',  but  the floorplan has no rows
       defined for this site. Any locations found for such insts will be ille-
       gal; create rows for this site to avoid this.


DESCRIPTION
       This  warning  will be reported by commands such as check_floorplan and
       place_design
       when the floorplan does not contain any rows for  the  specified  site.
       Each
       standard  cell  in the LEF should have a SITE defined for it. Instances
       of this
       cell can only be placed in rows defined for this site.

       Example:
       --------
       Run the following command(replace  RAM_site  with  your  specific  site
       name) to report the instances which use this site:

         dbGet [dbGet -p3 top.insts.cell.site.name RAM_site].name

       When  you  create the initial floorplan, it should automatically create
       rows for
       the standard cells defined in the netlist. You can add additional  rows
       using
       Floorplan - Row - Create Core Row or using the create_row text command.
       If this
       warning is occurring on standard cells you must define rows for them or
       else
       they will not be placed.

       If  this problem occurs for hard macros such as memories you can ignore
       it.
       Rows are not required by the hard macros. Remove the SITE definition in
       the LEF for the
       hard macros to avoid this message for them.

       If this occurs for IO pads you can also ignore it. Rows for IO pads are
       optional. You can create IO rows using Floorplan - Row - Create I/O Row
       or
       using the create_io_row text command.

                                                        IMPSP-365(20.14)
```

Note that here the problem occurs for hard macros. So you can ignore the issues related to VLC_SITE_sramHD…

This is one good example of the designer role in the process of filtering useful errors and warnings.

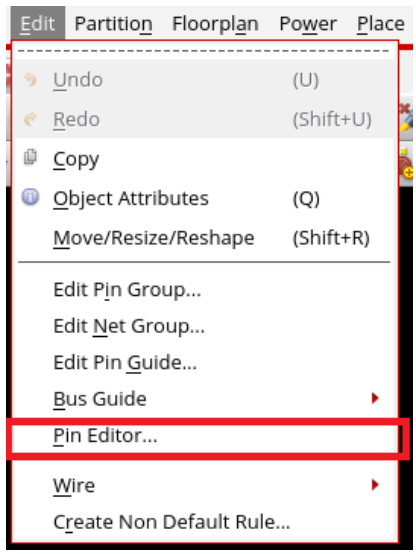
Save the design state as `DB/tutorial/top_32b_fplan`.

4.7. PLACING IO PINS

One important part of the floorplan, is the placement of input and output pins. Here in this proposed floorplan, we will place all the IOs on the top.

The IO file could be generated by hand, however, in this case, we will generate it using the GUI for simplicity reasons.

1. In the Innovus main menu, select **Edit > Pin Editor...**.

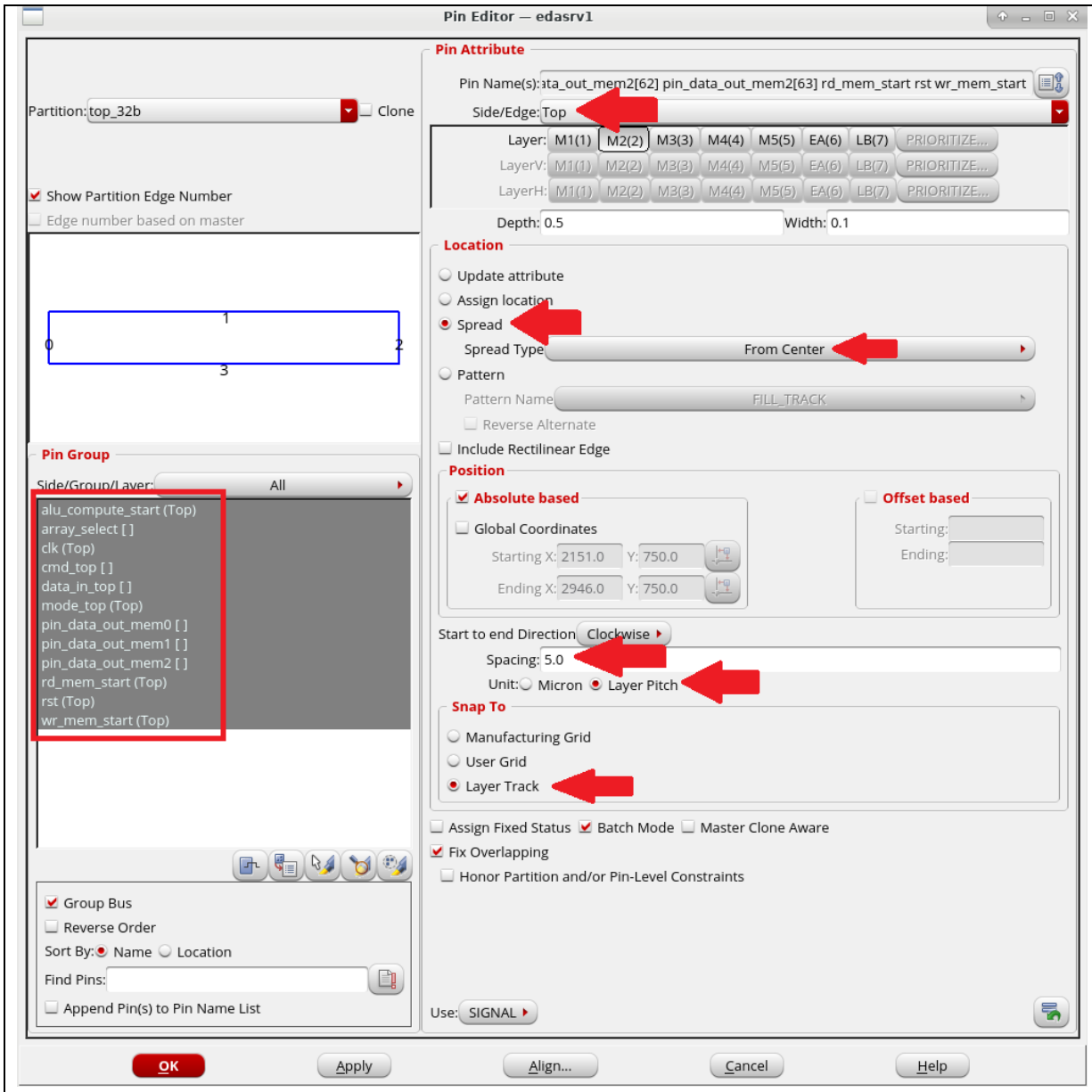


It can happen that a "overlay" window superposes with your innovus window, on the top-left corner. You can catch it and close it, or simply slightly move the window to bring the "edit" menu outside of the top left corner.

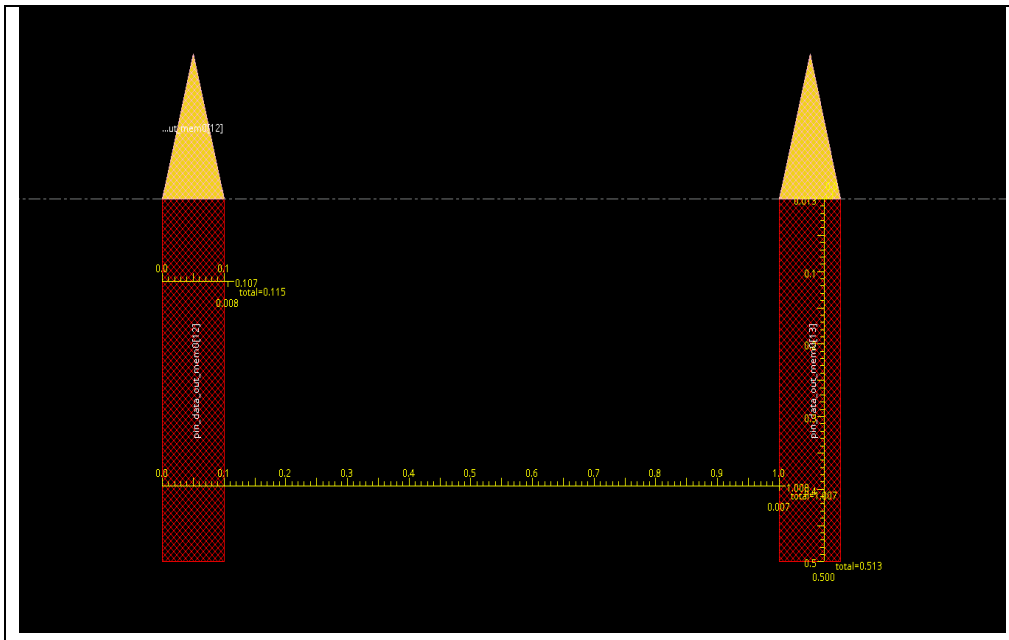
2. In the **Pin Editor** window:

- Start by selecting all the pins from the pin group panel on the left. Use shift-click to select them all.
- Then select spread and center in the location panel
- Then in the pin attributes select TOP as side/edge
- Select M2 for the layer of the pins
- Input 0.5 for depth and 0.1 for width – this is the size of the pin
- Make sure that the position is absolute based
- Select clockwise
- Put a 5.0 spacing and make sure the unit is in layer pitch – this represents the spacing between the pins – here 5*200ns for a 1um min pitch on M2.

3. Click apply and check the results in the design window.



4. In the design window you should see the pins appear on the top
5. Use the  button to create rulers, and check the sizes of the pins. These should correspond to the data you did input. Then press shift-k to remove the rulers and  to pass back to normal selection (and not ruler).

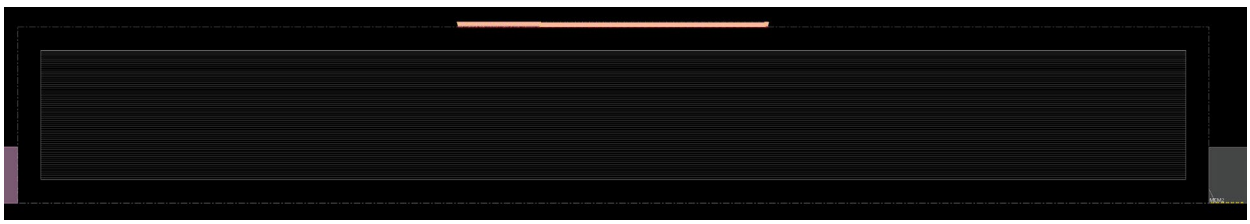


The pitch used in this technology can be seen from the tech lef file line 135 onwards

```
LAYER  M2
  TYPE  ROUTING ;
  OFFSET 0.00  0.00 ;
  AREA  0.052 ;
  WIDTH  0.10 ;
  PITCH  0.20 ;
```

The lef file can be found in :

```
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt
/5.00a/lef/5.6/cp65npksdst_m06f0f1.lef
```



4.7.1. SAVING THE I/O PLACEMENT

Once you have created your IO placement. Let's save the file inside the CONF folder.

```
innovus> write_io_file -locations CONF/top_32b_tutorial.io
Dumping FTerm of cell top_32b to file
```

You can then open the file and have a look at the syntax of the IO :

```
(globals
    version = 3
    io_order = default
```

```
)
(iopin
    (bottom
     (pin name="wr_mem_start"   offset=87.4000 layer=2 width=0.1000
depth=0.5200 place_status=placed  )
     (pin name="start_compute"  offset=88.0000 layer=2 width=0.1000
depth=0.5200 place_status=placed  )
     (pin name="rst" offset=88.6000       layer=2       width=0.1000
depth=0.5200 place_status=placed  )
     (pin name="rd_mem_start"   offset=89.2000 layer=2 width=0.1000
depth=0.5200 place_status=placed  )
     (pin name="pin_data_out_mem2[63]"    offset=89.8000    layer=2
width=0.1000 depth=0.5200 place_status=placed  )
     (pin name="pin_data_out_mem2[62]"    offset=90.4000    layer=2
width=0.1000 depth=0.5200 place_status=placed  )
     (pin name="pin_data_out_mem2[61]"    offset=91.0000    layer=2
width=0.1000 depth=0.5200 place_status=placed  )
     (pin name="pin_data_out_mem2[60]"    offset=91.6000    layer=2
width=0.1000 depth=0.5200 place_status=placed  )
     (pin name="pin_data_out_mem2[59]"    offset=92.2000    layer=2
width=0.1000 depth=0.5200 place_status=placed  )
     (pin name="pin_data_out_mem2[58]"    offset=92.8000    layer=2
width=0.1000 depth=0.5200 place_status=placed  )
     (pin name="pin_data_out_mem2[57]"    offset=93.4000    layer=2
width=0.1000 depth=0.5200 place_status=placed  )
```

Each pin is detailed with its name, position, layer, width, depth and status.

Save the design state as `DB/tutorial/top_32b_io` with the write_db command

4.8. PLACING THE MACRO BLOCK

The next step consists in placing the memories MEM0 MEM1 and MEM2.

In this floorplan, we place them on the bottom, pins facing up, with MEM0 on the left, MEM1 in the middle and MEM2 on the right.

For this, we will use the create_relative_flooplan command. Check page 845 of the innovus TCR documentation. Page 847 shows examples of how to understand the command.

```
innovus > delete_relative_floorplan -all
```

to remove the existing relative floorplan – this is generally a good practice

```
innovus > create_relative_floorplan -ref_type core_boundary -
orient   mx   -horizontal_edge_separate   {3      0      3}   -
vertical_edge_separate {0  0  0} -place MEM0 -ref top_32b
```

place MEM0.

Here we reference the core_boundary created before during the floorplan.

Then, we select the orientation as mx (mirror along the x axis to flip the memory vertically).

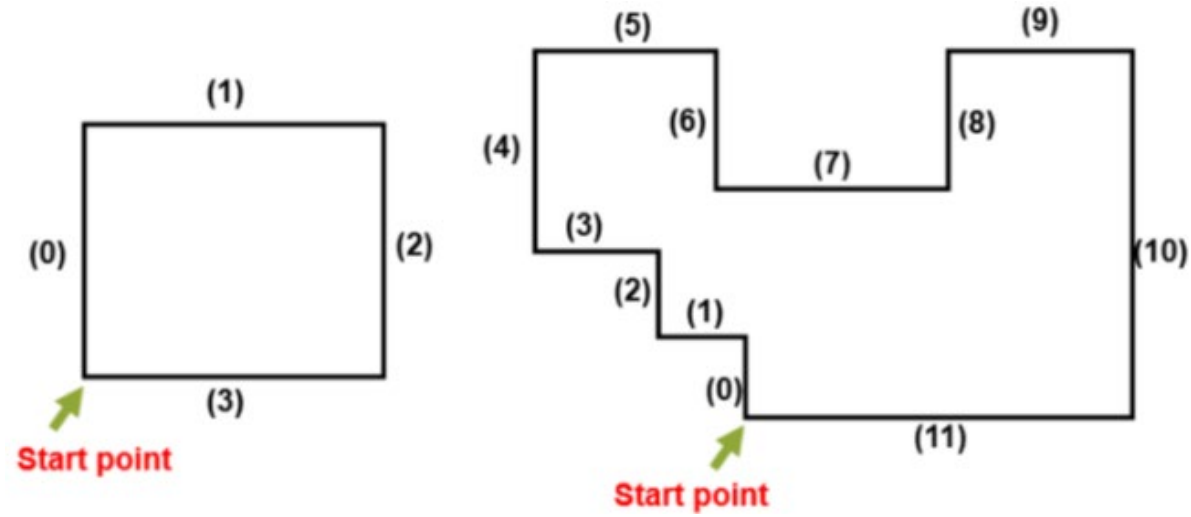
We select the horizontal edges to align. From the documentation, the edges calculations are done from the most bottom left edge, and counting clock-wise. Here both the core_boundary and MEM0 are rectangles (here the left picture). Considering the already flipped memory, We want to align the edge 3 of MEM0 with the edge 3 of core_boundary with a 0 offset.

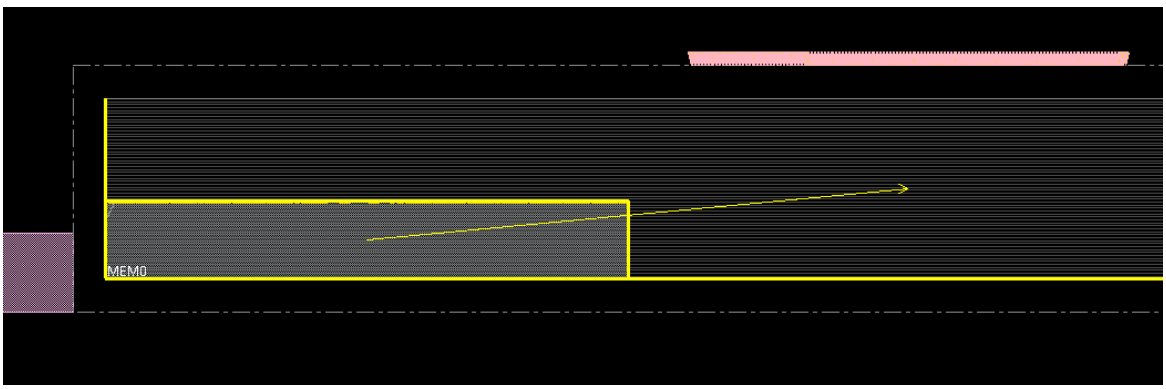
Leading to **-horizontal_edge_separate {3 0 3}**

Then, we do the same with the vertical edge. Edge 0 aligned on edge 0 with a 0 offset : **vertical_edge_separate {0 0 0}**

- Examples are given page 847 of the documentation

Finally we select the instance to place (MEM0), and it's reference design (top_32b)





Then we do the same with MEM1 :

```
innovus > create_relative_floorplan -ref_type core_boundary -
orient   mx   -horizontal_edge_separate   {3      0      3}   -
vertical_edge_separate {0  330  0} -place MEM1 -ref top_32b
```

and with MEM2

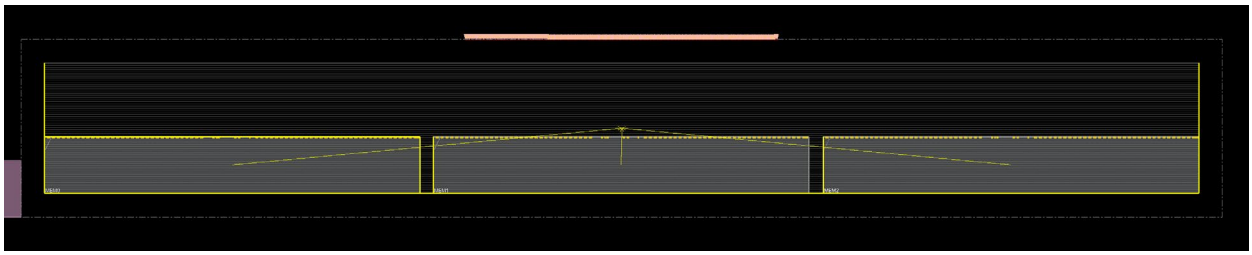
```
innovus > create_relative_floorplan -ref_type core_boundary -
orient   mx   -horizontal_edge_separate   {3      0      3}   -
vertical_edge_separate {2  0  2} -place MEM2 -ref top_32b
```

QUESTION 4-5: Why is the horizontal_edge_separate the same for the 3 memories ?

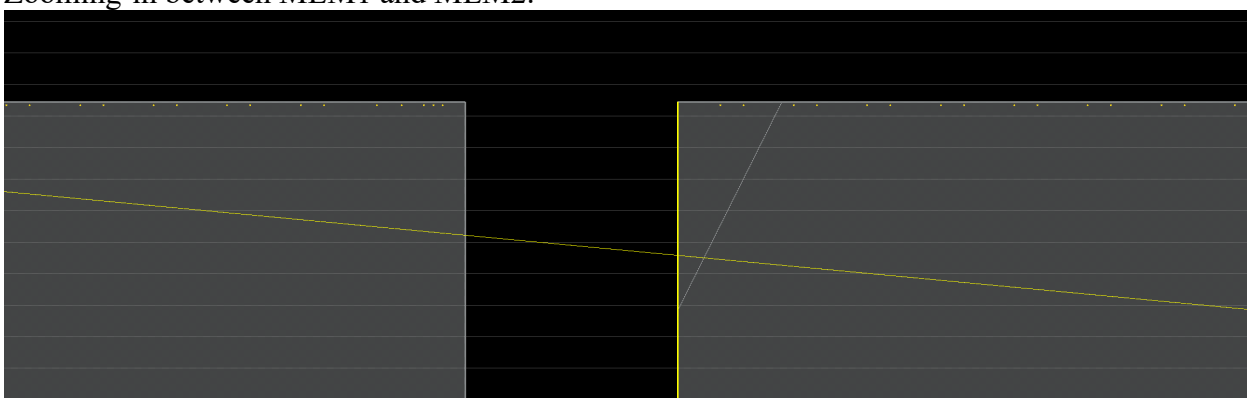
QUESTION 4-6: Why is the vertical edge separate of mem1 set as {0 330 0} and the one of mem2 as {3 0 3}?

4.8.1. ADDING PLACEMENT HALO AROUND THE MEMORIES

At this point you can note that there is some space between the memories. With rows where standard cells could be placed.



Zooming-in between MEM1 and MEM2.



However, it can be that placing standard cells too close to the memory causes issues. Also the available space between the memories may not allow to properly place and route gates. You can note that the memory macros partially occupy half a row of cells. Finally, as the pins are on the top, we want to make sure that we have enough space for the router to place connections there.

One way to cope with that is to add a Halo where nothing can be placed. This will also avoid issues if the lef of the memory is not properly designed (it can happen).

4.8.2. ADD A PLACEMENT HALO AROUND THE MEMORIES

```
innovus> delete_place_halo
innovus> create_place_halo -insts MEM0 -halo_deltas 0 0 7 5
innovus> create_place_halo -insts MEM1 -halo_deltas 7 0 7 5
innovus> create_place_halo -insts MEM2 -halo_deltas 7 0 0 5
```

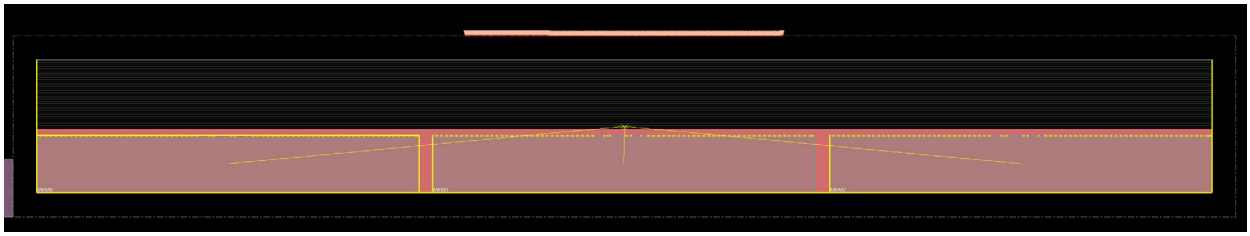
The -halo_delta command takes the following arguments left bottom right top.

MEM0 gets a halo on top and right

MEM1 gets a halo on top, left and right

MEM2 gets a halo on left and top.

By putting the halo on 7um on the left and right sides, we ensure overlap between them, and no placement of standard cells being done between them.



Save the design state as `DB/tutorial/top_32b_floorplan` with the write_db command

4.9. ADDING TAP CELLS

As in full custom design, tap cells must be introduced in the design, to make sure that no biasing issue will happen at the end. To do so, TAP cells must be included in the design before placing the cells.

The technology documentation says the following :

3.4.5 Tap Cell Placement

Cells in this library do not contain any substrate or well taps. A tap cell must be placed during floorplanning at locations and a frequency that satisfies design objectives as well as process design rule requirements. It consists of an nwell tap to VDD and a pwell tap to VSS. The tap is 2-grids wide.

The tap cellname is:

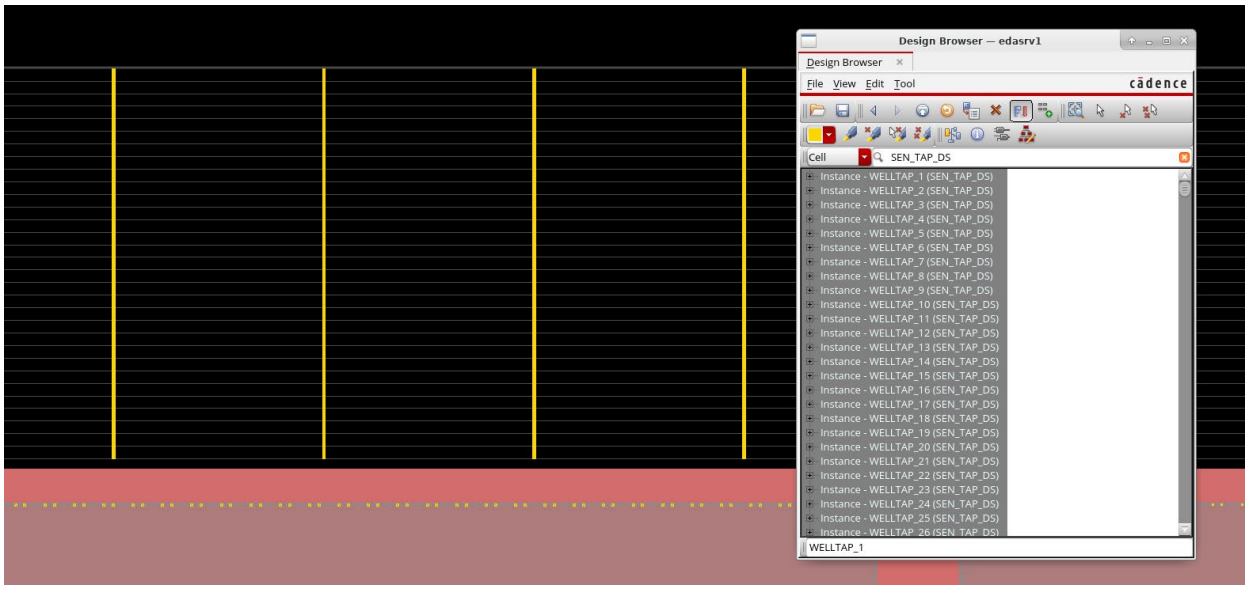
SEN_TAP_DS - VSS and VSS connected to Nwell/Pwell

```
Innovus> add_well_tap -cell "SEN_TAP_DS" -cell_interval 30 -
prefix "WELLTAP"
For 1054 new insts, *** Applied 0 GNC rules (cpu = 0:00:00.0)
Inserted 1054 well-taps <SEN_TAP_DS> cells (prefix WELLTAP).
```

- Note that if you had several power domains with different global power/ground nets, you could select a different power net for them.

You can check the cell placed through the design Browser (**tools>design browser**). In the search field, type **SEN_TAP_DS** and press **enter**. Press **Ctrl+A** to select them all to be able to highlight them.

- To highlight the selected cells, select a cell, select a color with the menu  and highlight it with  . To remove a highlighting, use the  button. To remove all of the highlights, use  .



4.10. CREATING THE POWER STRUCTURE

The goal here is to add a power ring and a ground ring around the core and the macro blocks as well as make sure that the macro power nets are properly connected to the global power ring[10] [11] [12].

The imported Verilog netlist does not include any power and ground connections. However, the cells that will be placed do have power/ground pins that will need to be routed to the global power/ground nets. It is the required to logically connect all the power nets of the cells to the global power (`VDD`) and ground (`VSS`) nets[13].

4.10.1. LOGICALLY CONNECT ALL POWER/GROUND PINS AND NETS

This step consists in connecting the circuit global power/ground nets (that have been defined in the design environment – cf init.tcl) to the power nets of the different instances being used in the design. The names of the power nets for the standard cell library and memory macro can be found in the corresponding lef files (they turn out to be aligned with the IPs we use, but it's not always the case). The standard cell library and memories power nets are `VDD`/`VSS`. The tap cells substrate biasing is `VBP` for pmos and `VBN` for the nmos.

Thereby, in this step, you want to connect the global power net `VDD` to `VDD` and `VBP`. While you want to connect the global ground net `VSS` to `VSS` and `VBN`.

```
innovus > connect_global_net VDD -type pg_pin -pin_base_name VDD
-override -verbose
```

[10] You can take a look at this reference: http://vlsibyjim.blogspot.com/2015/03/power-planning.html
[11] Inserting a pad ring is not addressed here. It is assumed that the considered design is a block to be included in a larger design.
[12] Additionally, you may want to add horizontal or vertical power stripes over the core area. This may be required for large designs to limit voltage drops in the row power rails.
[13] The global power and ground net names are defined in the technology LEF file.

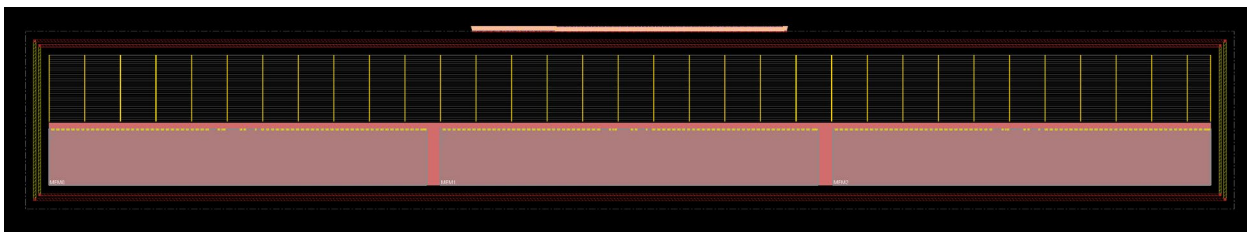
```
4732 new pwr-pin connections were made to global net 'VDD'.
innovus > connect_global_net VDD -type pg_pin -pin_base_name VBP
-override -verbose
3675 new pwr-pin connections were made to global net 'VDD'.
innovus > connect_global_net VSS -type pg_pin -pin_base_name VSS
-override -verbose
4732 new gnd-pin connections were made to global net 'VSS'.
innovus > connect_global_net VSS -type pg_pin -pin_base_name VBN
-override -verbose
3675 new gnd-pin connections were made to global net 'VSS'.
```

4.10.2. CREATING POWER AND GROUND RINGS AROUND THE CORE

The next step consists in creating power rings around the core. Power ring are useful as they enable an easy connectivity for the logic blocks from all sides with the power network. It is a good way to avoid IR drop effects.

Here we use the 20um available space around the core to insert two power rings (one for VDD and one for VSS) on all sides. Top and bottom parts of the rings are on Metal 5 (M5) while left and right are in Metal 4 (M4). The width of each metal line in the ring is 2um with a 2um spacing. The `-type core_ring` makes the ring designed around the "core", i.e., the main design area of this design. The `-center` option makes the ring centred on the core. This means that ring will be centred between the core area and the sides.

```
innovus > add_rings -nets {VDD VSS} -type core_rings -center 1
-layer {top M5 bottom M5 right M4 left M4} -width 2 -spacing 2
#% Begin add_rings (date=11/08 16:38:27, mem=2366.1M)

Loading cell geometries (cpu: 0:00:00.0, real: 0:00:00.0, peak mem:
2998.0M)
Ring generation is complete.
vias are now being generated.
add_rings created 8 wires.
ViaGen created 8 vias, deleted 0 via to avoid violation.
+--------+----------------+----------------+
|  Layer |     Created    |     Deleted    |
+--------+----------------+----------------+
|   M4   |        4       |       NA       |
|   V4   |        8       |        0       |
|   M5   |        4       |       NA       |
+--------+----------------+----------------+
#%  End  add_rings  (date=11/08  16:38:27,  total  cpu=0:00:00.0,
real=0:00:00.0, peak res=2367.9M, current mem=2367.9M)
```



4.10.3. CREATING POWER AND GROUND RING AND STRIPES TO CONNECT VDD/VSS PINS FOR THE MACRO BLOCK

Macro or IP blocks sometimes have non-conventional or non-easy-to-connect pins. Thereby, the designers must make sure that the power lines are well connected to the macro. Also, the designers must make sure that the power lines are large and dense enough to supply the macro or IP.

From this perspective, various options are possible. Designers could want to make rings around their macro blocks, or connect them with metal stripes. Also, as you did in the first phase of the labs on EDA, you must make sure that your standard cells connections to VDD and VSS are dense enough, to avoid IRdrop there.

First, let's have a look at the power structure in the design.

Enable the pin shapes in the blocks to see how are VDD and VSS pins organized. As you can see, there are vertical stripes in MEM0 MEM1 and MEM2. As the memories are aligned with each other, a couple of horizontal stripes could be enough to connect them all to VDD and VSS.

Then, for the standard cells, VDD and VSS Metal1 lines will be organized horizontally, as standard cells contain horizontal VSS and VSS lines. Inserting vertical stripes at regular intervals will allow to cut them and reduce their equivalent resistance. Thereby, we propose here to insert here 2 sets of vertical stripes between the memory macros.

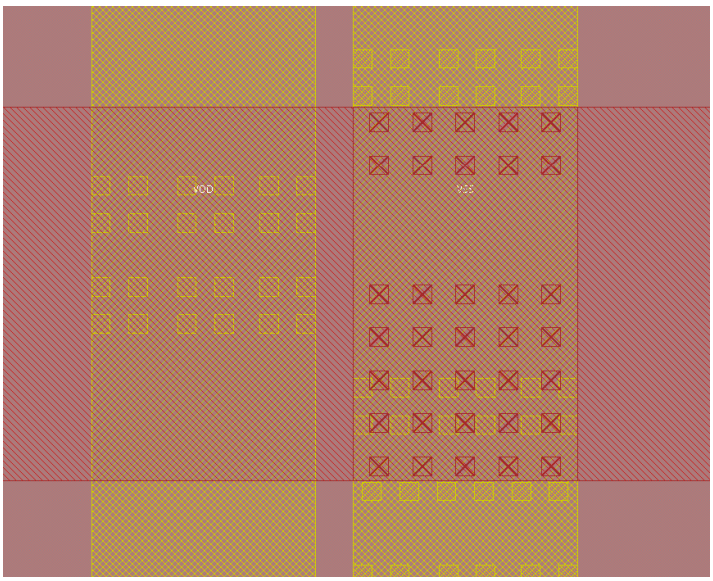
Check the documentation of the add_stripes (p2834) and add_rings (p2819) commands in the innovus TCR document.

```
innovus > add_stripes -direction horizontal -nets {VDD VSS} -
width 2 -spacing 2 -layer M5 -start_offset 20 -number_of_sets
1
#% Begin add_stripes (date=11/10 09:03:48, mem=2450.3M)

Initialize fgc environment(mem: 3105.9M) ...  fail and won't use fgc
to check drc(cpu: 0:00:00.0, real: 0:00:00.0, peak mem: 3105.9M)
**WARN: (IMPPP-133): The block boundary of the 'WELLTAP_1' instance
was increased to (20.000000 74.000000) (20.400000 75.849998) because
the (20.000000 75.750000) (20.400000 75.849998) cell geometry was
outside the original block boundary.
Type 'man IMPPP-133' for more detail.
...
**WARN: (IMPPP-133): The block boundary of the 'WELLTAP_19' instance
was increased to (560.000000 74.000000) (560.400024 75.849998)
because the (560.000000 75.750000) (560.400024 75.849998) cell
geometry was outside the original block boundary.
Type 'man IMPPP-133' for more detail.
**WARN: (IMPPP-133): The block boundary of the 'WELLTAP_20' instance
was increased to (590.000000 74.000000) (590.400024 75.849998)
because the (590.000000 75.750000) (590.400024 75.849998) cell
geometry was outside the original block boundary.
Type 'man IMPPP-133' for more detail.
**WARN: (EMS-27):    Message (IMPPP-133) has exceeded the current
message display limit of 20.
To increase the message display limit, refer to the product command
reference manual.
```

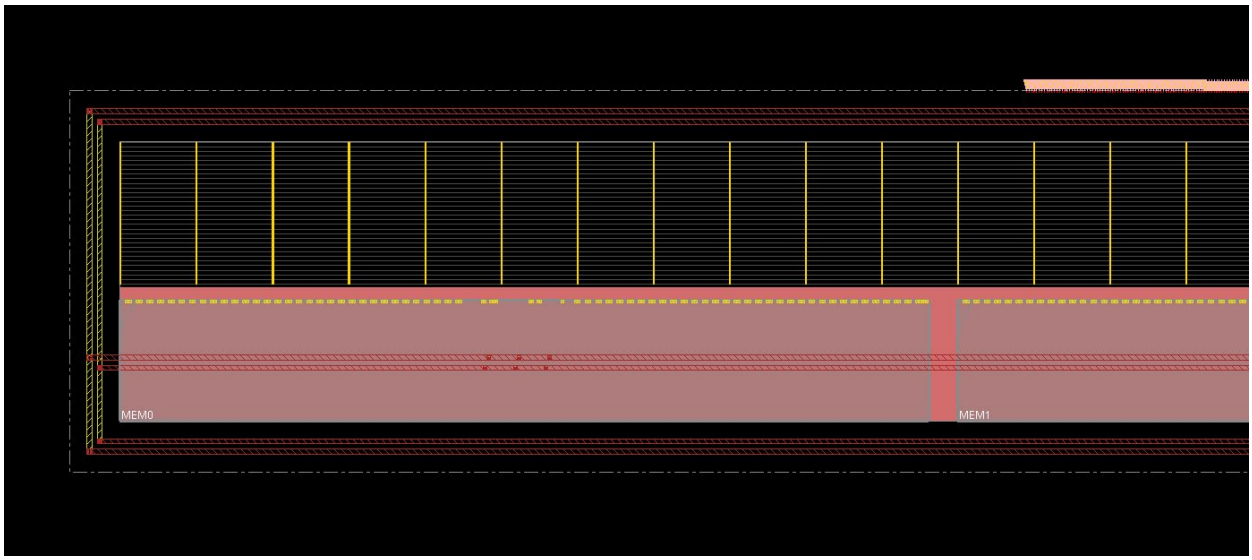
You can ignore these warnings

```
Loading cell geometries (cpu: 0:00:00.0, real: 0:00:00.0, peak mem:
3105.9M)
Loading wires (cpu: 0:00:00.0, real: 0:00:00.0, peak mem: 3105.9M)
Loading via instances (cpu: 0:00:00.0, real: 0:00:00.0, peak mem:
3105.9M)
Starting stripe generation ...
Non-Default Mode Option Settings :
  NONE
Stripe generation is complete.
vias are now being generated.
**WARN: (IMPPP-532): ViaGen Warning: The top layer and bottom layer
have same direction but only orthogonal via is allowed between layer
M3 & M5 at (20.00, 40.99) (20.80, 41.38).
...
**WARN: (IMPPP-4500):Extended number of geometries exist around
850.215000, 45.000000 between the M4 and M5 layers. This may increase
the run time.
Type 'man IMPPP-4500' for more detail.
```

Here the tool does complain about a complex geometry between M4 amd M5. You can use the suggested command to understand the warning. Looking at the 850-45 coordinates, you can see the following. Indeed the vias organization is complex. This part would need to be controlled post PnR with a DRC to make sure everything is correct.




```
add_stripes created 2 wires.
ViaGen created 247 vias, deleted 0 via to avoid violation.
+--------+----------------+----------------+
|  Layer |     Created    |     Deleted    |
+--------+----------------+----------------+
|   V4   |       247      |        0       |
|   M5   |         2      |       NA       |
+--------+----------------+----------------+
#%  End  add_stripes  (date=11/10  09:03:48,  total  cpu=0:00:00.1,
real=0:00:00.0, peak res=2450.5M, current mem=2450.5M)
```

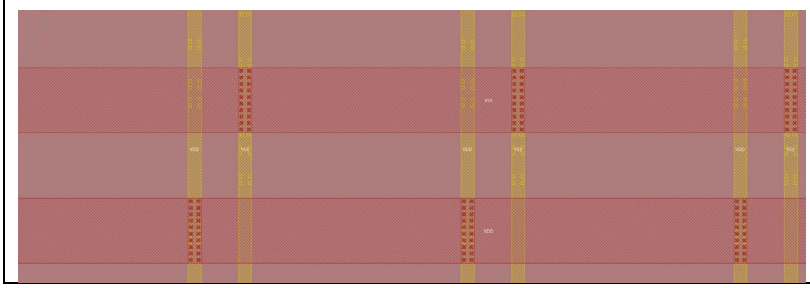


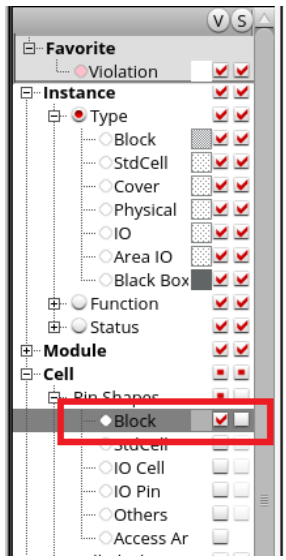
Zoom in the drawing and observe the vias and connections.

Enable the block shapes by ticking the cell>pin shapes>block , V, and look at the details of the connections.

Note how all the pin are organized for the memory.

Look how the tool did automatically connect the horizontal stripes with the VDD and VSS pins all along the width of the memr





Then let's create vertical stripes between the memories.

```
innovus > add_stripes -direction vertical -nets {VDD VSS} -width 2 -
spacing 2 -layer M4 -start_offset 320 -number_of_sets 1
innovus > add_stripes -direction vertical -nets {VDD VSS} -width 2 -
spacing 2 -layer M4 -start_offset 650 -number_of_sets 1
```

we do not report all the warnings though the same as the ones before appear.

Make the calculations on the x offset to understand 320 and 650

Note 1 : here we did not setup the vdd and vss lines for the standard cells. Only for the sram memories

Note 2 : here we do not insert specific rings around the memory macros, though we could with the following commands :

```
>select_obj MEM0
>add_rings -nets {VDD VSS} -type block_rings -around selected  -center
0 -skip_side {left bottom} -layer {top M5 bottom M5 right M4 left M4}
-width 1 -spacing 1
>deselect_obj -all
```

Note 3 : using the select_obj and deselect_obj is a good way to make sure that the tool does not try to draw a ring around a block you are not trying to use.

QUESTION 4-7 : We could have drawn an horizontal stripe by using the add_rings command and place it above the macros. Try to come up with a command that does that. Hint 1: save your design before playing with it, so that you can always restore it's state. Hint 2 : you can delete metal lines either with the undo button, or by selecting them and pressing the delete key on your keyboard.

4.10.4. ROUTE POWER NETS

Let's now connect the power nets with the special route command. The special_route does simply connect power structures. It can be generally used to connect the VDD and VSS nets of the standard cells, but you could use it, with the good arguments, to connect many power nets. Though, always remember that the commands you input do only do what you tell them to do[14]. Generally, discard any configuration that gives unexpected results.

Here we limit the -connect to core_pins to only create the standard cells power routing.

```
innovus > route_special -connect {core_pin} -net {VSS}
#% Begin route_special (date=11/10 09:27:35, mem=2468.3M)
*** Begin SPECIAL ROUTE on Fri Nov 10 09:27:35 2023 ***
SPECIAL          ROUTE          ran          on          directory:
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_202
4/INNOVUS
SPECIAL   ROUTE   ran   on   machine:   edasrv1   (Linux   3.10.0-
1160.83.1.el7.x86_64 Xeon 3.00Ghz)

Begin option processing ...
srouteConnectPowerBump set to false
routeSelectNet set to "VSS"
routeSpecial set to true
srouteConnectBlockPin set to false
srouteConnectConverterPin set to false
srouteConnectPadPin set to false
srouteConnectStripe set to false
srouteFollowCorePinEnd set to 3
srouteFollowPadPin set to false
srouteJogControl set to "preferWithChanges differentLayer"
sroutePadPinAllPorts set to true
sroutePreserveExistingRoutes set to true
srouteRoutePowerBarPortOnBothDir set to true
End option processing: cpu: 0:00:00, real: 0:00:00, peak: 440.00
megs.

Reading DB technology information...
Finished reading DB technology information.
Reading floorplan and netlist information...
```

[14] As we generally say : "crap-in, crap-out". Wrong configuration gives wrong results.

```
Finished reading floorplan and netlist information.
**WARN: (IMPSR-4302): Cap-table/qrcTechFile is found in the design,
so the same information from the technology file will be ignored.
Read in 15 layers, 7 routing layers, 1 overlap layer
Read in 1184 macros, 110 used
Read in 1164 components
  1161 core components: 107 unplaced, 0 placed, 1054 fixed
  3 block/ring components: 0 unplaced, 0 placed, 3 fixed
Read in 266 physical pins
  266 physical pins: 0 unplaced, 266 placed, 0 fixed
Read in 4 blockages
Read in 266 nets
Read in 2 special nets, 2 routed
Read in 2808 terminals
1 net selected.

Begin power routing ...
CPU time for FollowPin 0 seconds
  Number of Core ports routed: 32
  Number of Followpin connections: 16
End power routing: cpu: 0:00:00, real: 0:00:00, peak: 451.00 megs.



 Begin updating DB with routing results ...
 Updating DB with 266 io pins ...
 Updating DB with 0 via definition ...
route_special created 48 wires.
ViaGen created 192 vias, deleted 0 via to avoid violation.
+--------+----------------+----------------+
|  Layer |     Created    |     Deleted    |
+--------+----------------+----------------+
|   M1   |       48       |       NA       |
|   V1   |       64       |        0       |
|   V2   |       64       |        0       |
|   V3   |       64       |        0       |
+--------+----------------+----------------+
#% End route_special (date=11/10 09:27:36, total cpu=0:00:00.3,
real=0:00:01.0, peak res=2477.8M, current mem=2477.8M)

innovus > route_special -connect {core_pin} -net {VDD}
#% Begin route_special (date=11/10 09:28:14, mem=2473.6M)
*** Begin SPECIAL ROUTE on Fri Nov 10 09:28:14 2023 ***
SPECIAL          ROUTE          ran          on          directory:
/home/levisse/projects/CLASSROOM/PHASE2/testdirs/edalabs_phase2_202
4/INNOVUS
SPECIAL   ROUTE   ran   on   machine:   edasrv1   (Linux   3.10.0-
1160.83.1.el7.x86_64 Xeon 3.00Ghz)

Begin option processing ...
srouteConnectPowerBump set to false
routeSelectNet set to "VDD"
routeSpecial set to true
srouteConnectBlockPin set to false
srouteConnectConverterPin set to false
srouteConnectPadPin set to false
srouteConnectStripe set to false
srouteFollowCorePinEnd set to 3
```

```
srouteFollowPadPin set to false
srouteJogControl set to "preferWithChanges differentLayer"
sroutePadPinAllPorts set to true
sroutePreserveExistingRoutes set to true
srouteRoutePowerBarPortOnBothDir set to true
End option processing: cpu: 0:00:00, real: 0:00:00, peak: 451.00
megs.

Reading DB technology information...
Finished reading DB technology information.
Reading floorplan and netlist information...
Finished reading floorplan and netlist information.
**WARN: (IMPSR-4302):Cap-table/qrcTechFile is found in the design,
so the same information from the technology file will be ignored.
Read in 15 layers, 7 routing layers, 1 overlap layer
Read in 1184 macros, 110 used
Read in 1164 components
  1161 core components: 107 unplaced, 0 placed, 1054 fixed
  3 block/ring components: 0 unplaced, 0 placed, 3 fixed
Read in 266 physical pins
  266 physical pins: 0 unplaced, 266 placed, 0 fixed
Read in 4 blockages
Read in 266 nets
Read in 2 special nets, 2 routed
Read in 2808 terminals
1 net selected.

Begin power routing ...
CPU time for FollowPin 0 seconds
  Number of Core ports routed: 32
  Number of Followpin connections: 16
End power routing: cpu: 0:00:00, real: 0:00:00, peak: 451.00 megs.



 Begin updating DB with routing results ...
 Updating DB with 266 io pins ...
 Updating DB with 0 via definition ...
route_special created 48 wires.
ViaGen created 192 vias, deleted 0 via to avoid violation.
+--------+----------------+----------------+
|  Layer |     Created    |     Deleted    |
+--------+----------------+----------------+
|   M1   |       48       |       NA       |
|   V1   |       64       |        0       |
|   V2   |       64       |        0       |
|   V3   |       64       |        0       |
+--------+----------------+----------------+
#% End route_special (date=11/10 09:28:14, total cpu=0:00:00.3,
real=0:00:00.0, peak res=2474.7M, current mem=2474.7M)
```

check the documentation of the route_special command p2915.

4.10.5. LEARN HOW TO DEAL WITH MACRO PROBLEMS

The digital design implementation flow can be seen from two different angles. On one hand, your job is to setup the flow correctly and not make mistakes. On the other hand, you also have to deal with issues that may have come before in the design flow.

Sometimes (mostly when the problem is too big to be handled) you can propagate the issues and ask for corrections to the development team or the provider. Sometimes things are just as they are, and corrections cannot be provided by the IP providers.

Reasons for this can be numerous :

- The block has been designed by another design team, and applying a patch does not allow to meet the deadline.
- The block is provided as is, and/or the provider does not provide support.
    - This is generally the case when you are a small entity (i.e., not Apple), a university, or when dealing with “old” nodes.
- The block has been automatically generated, and the issue may not be easily patchable, or does not occur in different configurations

In all these cases you must correct the issue yourself at your level and most importantly document it, so that other users know how to face it in the future.

Here, as you can see in the screenshot, on the left side of the memories generated with the memory compiler, when showing the pin, the lef is incorrectly extracted. From the GDS (right picture) you can see that the horizontal metal lines for VDD and VSS continue, while in the LEF (left picture) they stop. The VSS pin is correctly connected to the vertical pin, while the VDD is not. This leads to incorrect connectivity checks when running verification as innovus does identify these VDD pins as floating.

Run the command

```
innovus > check_connectivity
VERIFY_CONNECTIVITY use new engine.

******** Start: VERIFY CONNECTIVITY ********
Start Time: Fri Nov 10 09:31:50 2023

Design Name: top_32b
Database Units: 2000
Design Boundary: (0.0000, 0.0000) (1020.0000, 150.0000)
Error Limit = 1000; Warning Limit = 50
Check all nets
**WARN: (IMPVFC-96): Instance pin D of net cmd_top[1] has not been
placed. Please make sure instance reg_cmd_top_reg_1_ is placed and
rerun verifyConnectivity.
...
**WARN: (IMPVFC-96): Instance pin CK of net clk has not been placed.
Please make sure instance reg_data_out_mem1_reg_19_ is placed and
rerun verifyConnectivity.
**WARN: (EMS-27):    Message (IMPVFC-96) has exceeded the current
message display limit of 20.
To increase the message display limit, refer to the product command
reference manual.
```

It does here complain about unplaced cells, as we did not do the placement yet. You can ignore these warnings.

```
Net clk: no routing.
Net n_Logic0_: no routing.
Net VDD: has an unconnected terminal, has special routes with opens.
```

This last “net VDD” violation is the issue we were mentioning before.

Look at the left side of the memories.

```
Begin Summary
    2 Problem(s) (IMPVFC-98): Net has no global routing and no
special routing.
    63 Problem(s) (IMPVFC-96): Terminal(s) are not connected.
    1 Problem(s) (IMPVFC-200): Special Wires: Pieces of the net are
not connected together.
    66 total info(s) created.
End Summary

End Time: Fri Nov 10 09:31:50 2023
Time Elapsed: 0:00:00.0

******** End: VERIFY CONNECTIVITY ********
  Verification Complete : 66 Viols.  0 Wrngs.
  (CPU Time: 0:00:00.0  MEM: 0.000M)
```

Run the

```
innovus > delete_drc_markers
```

to remove the white crosses

Two solutions are possible :

1. Wave (ignore) these errors, knowing these are not actual errors
2. Correct them to feel better and have cleaner logs

Here we propose to use the special_route to connect them.

```
route_special -inst MEM0 -connect block_pin -nets {VDD} -block_pin
left_boundary
route_special -inst MEM1 -connect block_pin -nets {VDD} -block_pin
left_boundary
route_special -inst MEM2 -connect block_pin -nets {VDD} -block_pin
left_boundary
```

QUESTION 4-8 : explain with your own words what happens with the left side of the memory. You could open the gds from the memory on embedit and comment on the gds.

QUESTION 4-9 : Explain the content of the route_special command used there. Inspire from the documentation p2915.

QUESTION 4-10 : Check the state of your design again by running a connectivity check. What does it complain about now ?

Save the design state as `DB/tutorial/top_32b_power` with the write_db command

4.11. CHECKING THE DRC RULES

At any point in your design, as for the connectivity check, it is important to make sure that the metal lines you draw, and the floorplan you design is correct from a design rule check standpoint.

Innovus does integrate a simple drc checker based on the technology lef you did provide when setting up the tool.

Run a drc check with

```
innovus > check_drc
```

remove the white cross with :

```
innovus > delete_drc_markers
```

QUESTION 4-11 : explain the results of the drc_check. Can you fully trust this drc check based on the above explanation ? how different is that compared to the DRC from Calibre?

4.12. PLACING CORE CELLS

This step places the standard cells in the rows, according to the imported Verilog netlist. Placement also follows legalization rules, such as cells cannot overlap each other and takes into account short and spacing DRC rules required by routing.

4.12.1. PLACE THE CORE CELLS

Let's first place the design cells. Start by a push_snapshot_stack to track the timing info.

```
innovus > push_snapshot_stack
innovus > place_design
Extracting standard cell pins and blockage ......
Pin and blockage extraction finished
*** Starting place_design default flow ***
*** Start delete_buffer_trees ***
Info: Detect buffers to remove automatically.
Analyzing netlist ...
Updating netlist
AAE DB initialization (MEM=3184.36 CPU=0:00:00.0 REAL=0:00:01.0)

*summary: 73 instances (buffers/inverters) removed
*** Finish delete_buffer_trees (0:00:00.6) ***
```

First it does clean a bit the design, removing buffers that could have been inserted by some hold correction before

```
**INFO: Enable pre-place timing setting for timing analysis
Set Using Default Delay Limit as 101.
**WARN: (IMPDC-1629): The default delay limit was set to 101. This is
less than the default of 1000 and may result in inaccurate delay
calculation for nets with a fanout higher than the setting.  If
needed, the default delay limit may be adjusted by running the command
'set delaycal_use_default_delay_limit'.
Set Default Net Delay as 0 ps.
Set Default Net Load as 0 pF.
**INFO: Analyzing IO path groups for slack adjustment
```

```
Effort level <high> specified for reg2reg_tmp.75869 path_group
#############################################################################
###############
# Design Stage: PreRoute
# Design Name: top_32b
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
#############################################################################
###############
Start delay calculation (fullDC) (1 T). (MEM=3193.9)
Total number of fetched objects 4188
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
End delay calculation. (MEM=3251.78 CPU=0:00:00.8 REAL=0:00:01.0)
End   delay   calculation   (fullDC).   (MEM=3205.62   CPU=0:00:01.3
REAL=0:00:01.0)
**INFO: Disable pre-place timing setting for timing analysis
Set Using Default Delay Limit as 1000.
Set Default Net Delay as 1000 ps.
Set Default Net Load as 0.5 pF.
**INFO: Pre-place timing setting for timing analysis already disabled
Deleted 0 physical inst  (cell - / prefix -).
Did not delete 1054 physical insts as they were marked preplaced.
INFO: #ExclusiveGroups=0
INFO: There are no Exclusive Groups.
*** Starting "NanoPlace(TM) placement v#9 (mem=3187.9M)" ...
*** Build Buffered Sizing Timing Model (Used Compact Buffer Set)
...(cpu=0:00:01.5 mem=3203.9M) ***
*** Build Virtual Sizing Timing Model
(cpu=0:00:02.8 mem=3218.9M) ***
No user-set net weight.
Net fanout histogram:
2          : 2982 (71.2%) nets
3          : 800 (19.1%) nets
4     -    14    : 305 (7.3%) nets
15    -    39    : 92 (2.2%) nets
40    -    79    : 7 (0.2%) nets
80    -    159   : 0 (0.0%) nets
160   -    319   : 0 (0.0%) nets
320   -    639   : 2 (0.0%) nets
640   -    1279  : 0 (0.0%) nets
1280  -    2559  : 0 (0.0%) nets
2560  -    5119  : 0 (0.0%) nets
5120+      : 0 (0.0%) nets
Options:    timingDriven    clkGateAware    ignoreScan    pinGuide
congEffort=auto gpeffort=medium
Scan chains were not defined.
#std cell=4661 (1054 fixed + 3607 movable) #buf cell=0 #inv cell=365
#block=3 (0 floating + 3 preplaced)
#ioInst=0   #net=4188   #term=14630   #term/net=3.49,   #fixedIo=0,
#floatIo=0, #fixedPin=0, #floatPin=266
stdCell: 4661 single + 0 double + 0 multi
Total standard cell length = 7.4916 (mm), area = 0.0135 (mm^2)
Average module density = 0.245.
Density for the design = 0.245.
```

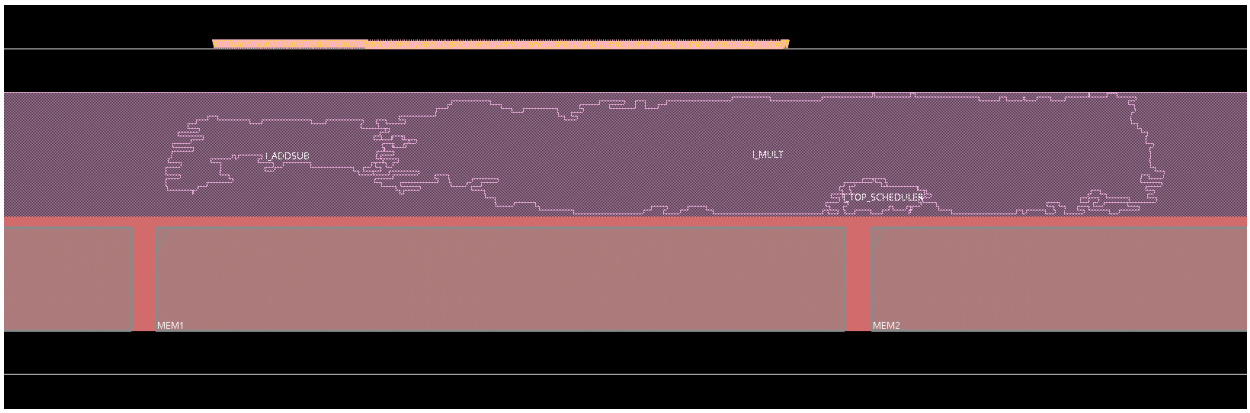
```
       = stdcell_area 35350 sites (12726 um^2) / alloc_area 144321
sites (51956 um^2).
Pin Density = 0.04895.
            = total # of pins 14630 / total area 298900.
=== lastAutoLevel = 10
Clock gating cells determined by native netlist tracing.
Iteration  1: Total net bbox = 1.471e+05 (1.21e+05 2.58e+04)
               Est.  stn bbox = 1.535e+05 (1.27e+05 2.64e+04)
               cpu = 0:00:00.0 real = 0:00:00.0 mem = 3329.5M
Iteration  2: Total net bbox = 1.471e+05 (1.21e+05 2.58e+04)
               Est.  stn bbox = 1.535e+05 (1.27e+05 2.64e+04)
               cpu = 0:00:00.0 real = 0:00:00.0 mem = 3329.5M
Iteration  3: Total net bbox = 1.538e+05 (1.29e+05 2.52e+04)
               Est.  stn bbox = 1.704e+05 (1.44e+05 2.63e+04)
               cpu = 0:00:00.3 real = 0:00:00.0 mem = 3334.9M
Active setup views:
    analysis_setup_wc
Iteration  4: Total net bbox = 1.482e+05 (1.23e+05 2.53e+04)
               Est.  stn bbox = 1.629e+05 (1.36e+05 2.64e+04)
               cpu = 0:00:00.2 real = 0:00:00.0 mem = 3334.9M
Iteration  5: Total net bbox = 1.728e+05 (1.48e+05 2.53e+04)
               Est.  stn bbox = 1.988e+05 (1.72e+05 2.65e+04)
               cpu = 0:00:00.3 real = 0:00:00.0 mem = 3334.9M
Iteration  6: Total net bbox = 1.656e+05 (1.39e+05 2.66e+04)
               Est.  stn bbox = 1.916e+05 (1.64e+05 2.80e+04)
               cpu = 0:00:00.4 real = 0:00:01.0 mem = 3337.2M

Iteration  7: Total net bbox = 1.661e+05 (1.39e+05 2.74e+04)
               Est.  stn bbox = 1.920e+05 (1.63e+05 2.88e+04)
               cpu = 0:00:00.1 real = 0:00:00.0 mem = 3342.5M
Iteration  8: Total net bbox = 1.661e+05 (1.39e+05 2.74e+04)
               Est.  stn bbox = 1.920e+05 (1.63e+05 2.88e+04)
               cpu = 0:00:01.6 real = 0:00:01.0 mem = 3342.5M
Iteration  9: Total net bbox = 1.677e+05 (1.35e+05 3.29e+04)
               Est.  stn bbox = 1.963e+05 (1.60e+05 3.59e+04)
               cpu = 0:00:00.6 real = 0:00:01.0 mem = 3342.5M
Iteration 10: Total net bbox = 1.677e+05 (1.35e+05 3.29e+04)
               Est.  stn bbox = 1.963e+05 (1.60e+05 3.59e+04)
               cpu = 0:00:01.6 real = 0:00:02.0 mem = 3342.5M
Iteration 11: Total net bbox = 1.754e+05 (1.40e+05 3.55e+04)
               Est.  stn bbox = 2.094e+05 (1.70e+05 3.91e+04)
               cpu = 0:00:00.9 real = 0:00:01.0 mem = 3342.5M
Iteration 12: Total net bbox = 1.754e+05 (1.40e+05 3.55e+04)
               Est.  stn bbox = 2.094e+05 (1.70e+05 3.91e+04)
               cpu = 0:00:01.6 real = 0:00:01.0 mem = 3342.5M
Iteration 13: Total net bbox = 1.822e+05 (1.45e+05 3.75e+04)
               Est.  stn bbox = 2.187e+05 (1.78e+05 4.10e+04)
               cpu = 0:00:02.4 real = 0:00:03.0 mem = 3342.5M
Iteration 14: Total net bbox = 1.822e+05 (1.45e+05 3.75e+04)
               Est.  stn bbox = 2.187e+05 (1.78e+05 4.10e+04)
               cpu = 0:00:00.0 real = 0:00:00.0 mem = 3342.5M
Iteration 15: Total net bbox = 1.822e+05 (1.45e+05 3.75e+04)
               Est.  stn bbox = 2.187e+05 (1.78e+05 4.10e+04)
               cpu = 0:00:00.0 real = 0:00:00.0 mem = 3342.5M
*** cost = 1.822e+05 (1.45e+05 3.75e+04) (cpu for global=0:00:10.4)
real=0:00:12.0***
Info: 0 clock gating cells identified, 0 (on average) moved 0/7
Solver runtime cpu: 0:00:04.6 real: 0:00:04.5
```

```
Core Placement runtime cpu: 0:00:05.1 real: 0:00:06.0
**WARN: (IMPSP-9025): No scan chain specified/traced.
Type 'man IMPSP-9025' for more detail.
*** Starting place_detail (6:56:12 mem=3358.5M) ***
Total net bbox length = 1.822e+05 (1.447e+05 3.752e+04) (ext =
2.608e+04)
Move report: Detail placement moves 3607 insts, mean move: 0.88 um,
max move: 19.34 um
     Max move on inst (reg_data_in_top_reg_47_): (424.99, 114.73) -
-> (432.40, 102.80)
     Runtime: CPU: 0:00:00.3 REAL: 0:00:00.0 MEM: 3358.5MB
Summary Report:
Instances move: 3607 (out of 3607 movable)
Instances flipped: 0
Mean displacement: 0.88 um
Max displacement: 19.34 um (Instance: reg_data_in_top_reg_47_)
(424.988, 114.727) -> (432.4, 102.8)
     Length: 23 sites, height: 1 rows, site name: CP65_DST, cell
type: SEN_FDPRBQ_D_1
Total net bbox length = 1.793e+05 (1.420e+05 3.733e+04) (ext =
2.554e+04)
Runtime: CPU: 0:00:00.4 REAL: 0:00:00.0 MEM: 3358.5MB
*** Finished place_detail (6:56:12 mem=3358.5M) ***
*** End of Placement (cpu=0:00:14.6, real=0:00:15.0, mem=3346.5M)
***
default core: bins with density > 0.750 = 41.04 % ( 158 / 385 )
Density distribution unevenness ratio = 29.646%
*** Free Virtual Timing Model ...(mem=3346.5M)
UM:  flow.cputime  flow.realtime  timing.setup.tns  timing.setup.wns
snapshot
UM:*                                                              final
      flow.cputime          flow.realtime          timing.setup.tns
timing.setup.wns  snapshot
UM:*                                                       global_place
**INFO: Enable pre-place timing setting for timing analysis
Set Using Default Delay Limit as 101.
**WARN: (IMPDC-1629): The default delay limit was set to 101. This is
less than the default of 1000 and may result in inaccurate delay
calculation for nets with a fanout higher than the setting.  If
needed, the default delay limit may be adjusted by running the command
'set delaycal_use_default_delay_limit'.
Set Default Net Delay as 0 ps.
Set Default Net Load as 0 pF.
**INFO: Analyzing IO path groups for slack adjustment
Effort level <high> specified for reg2reg_tmp.75869 path_group
###############################################################
###############
# Design Stage: PreRoute
# Design Name: top_32b
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
###############################################################
###############
Start delay calculation (fullDC) (1 T). (MEM=3318.75)
Total number of fetched objects 4188
```

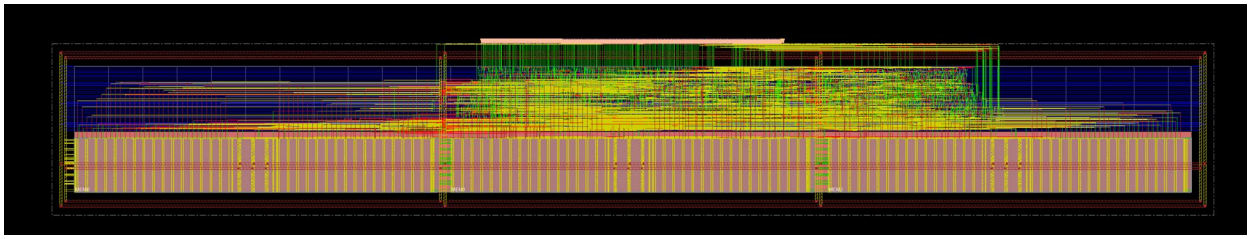
```
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
End delay calculation. (MEM=3354.17 CPU=0:00:00.6 REAL=0:00:00.0)
End   delay   calculation   (fullDC).   (MEM=3354.17   CPU=0:00:00.8
REAL=0:00:00.0)
**INFO: Disable pre-place timing setting for timing analysis
Set Using Default Delay Limit as 1000.
Set Default Net Delay as 1000 ps.
Set Default Net Load as 0.5 pF.
Info: Disable timing driven in postCTS congRepair.

Starting congRepair ...
[NR-eGR] Num Prerouted Nets = 0  Num Prerouted Wires = 0
[NR-eGR] Read numTotalNets=4188  numIgnoredNets=0
[NR-eGR] There are 1 clock nets ( 0 with NDR ).
[NR-eGR] ============ Routing rule table ============
[NR-eGR] Rule id: 0  Nets: 4188
[NR-eGR] ========================================
[NR-eGR]
[NR-eGR] Layer group 1: route 4188 net(s) in layer range [2, 5]
[NR-eGR] Early Global Route overflow of layer group 1: 0.13% H +
0.00% V. EstWL: 1.890486e+05um
[NR-eGR] Overflow after Early Global Route (GR compatible) 0.07% H +
0.00% V
[NR-eGR] Overflow after Early Global Route 0.10% H + 0.00% V
Early Global Route congestion estimation runtime: 0.16 seconds, mem
= 3344.4M
Local HotSpot Analysis: normalized max congestion hotspot area =
0.44, normalized total congestion hotspot area = 0.89 (area is in
unit of 4 std-cell row bins)
Skipped repairing congestion.
[NR-eGR] ------------------------------------------------------------
----------------
[NR-eGR]      M1  (1F) length: 0.000000e+00um, number of vias: 12874
[NR-eGR]      M2  (2H) length: 6.319813e+04um, number of vias: 22430
[NR-eGR]      M3  (3V) length: 3.538140e+04um, number of vias: 3637
[NR-eGR]      M4  (4H) length: 8.191603e+04um, number of vias: 757
[NR-eGR]      M5  (5V) length: 1.120127e+04um, number of vias: 0
[NR-eGR] Total length: 1.916968e+05um, number of vias: 39698
[NR-eGR] ------------------------------------------------------------
----------------
[NR-eGR] Total eGR-routed clock nets wire length: 2.471223e+03um
[NR-eGR] ------------------------------------------------------------
----------------
Early Global Route wiring runtime: 0.10 seconds, mem = 3247.4M
Tdgp not successfully inited but do clear! skip clearing
End of congRepair (cpu=0:00:00.3, real=0:00:00.0)
*** Finishing place_design default flow ***
***** Total cpu  0:0:19
***** Total real time  0:0:20
**place_design ... cpu = 0: 0:19, real = 0: 0:20, mem = 3247.4M **
Tdgp not successfully inited but do clear! skip clearing

*** Summary of all messages that are not suppressed in this session:
Severity  ID               Count  Summary
WARNING   IMPDC-1629           2  The default delay limit was set to
%d. T...
WARNING   IMPSP-9025           1  No scan chain specified/traced.
```

```
*** Message Summary: 3 warning(s), 0 error(s)

      flow.cputime           flow.realtime          timing.setup.tns
timing.setup.wns  snapshot
UM:*                                                                 final
-------------------------------------------------------------
     Current design flip-flop statistics

Single-Bit FF Count  :          500
Multi-Bit FF Count   :            0
Total Bit Count      :          500
Total FF Count       :          500
Bits Per Flop        :        1.000
-------------------------------------------------------------
      flow.cputime           flow.realtime          timing.setup.tns
timing.setup.wns  snapshot
UM:                          24915.7                              164110
place_design
0
```

The default placing option for place_design is timing-driven. If design views (here bc for hold and wc for setup) are defined, it will place the cells considering timing constraints. This way, wires are appearing. But these wires are not yet the final ones. Click on the  button to see the shape of the placed entities. See how is the different blocks are placed.



You can click back on  to see the actual placement of wires. At this point the wires are not yet optimally placed, and timing is not yet met.



Run the check place command :

```
@innovus 346> check_place
Begin checking placement ... (start mem=3433.8M, init mem=3450.6M)
*info: Placed = 4928           (Fixed = 1057)
*info: Unplaced = 0
Placement Density:24.83%(13391/53925)
Placement Density (including fixed std cells):25.88%(14150/54684)
Finished check_place (total: cpu=0:00:00.1, real=0:00:00.0; vio
checks: cpu=0:00:00.1, real=0:00:00.0; mem=3450.6M)
0
```

As you can see from the density check, the density is only around 25%. The remaining space will be used to insert buffers or inverters (e.g., when synthesizing the clock tree) or to replace cells with other cells with the same function, but with higher drive capabilities (e.g., when performing timing-driven optimizations).

Defining a floorplan that allows a sufficiently low core utilization is important, though, as we will see in the following part, is not always enough. Targeting 60% can be a good starting point, here we are much below, which means we have a lot of room to play with.

4.12.2. DO THE PRECTS TIMING OPTIMIZATION

The opt_design command can be used at various stages of the design to reoptimize the netlist under various constraints. It does optimize the design in the context of the timing constraints. Options such as -pre_cts, -post_cts, -post_route control general optimization associated with these stages. Specific optimizations such as -hold, -setup can be used to meet the timing constraints in hold or setup.

Latest optimizations of the command suggest to use place_opt_design, merging both place and opt_design -pre_cts commands.

```
innovus > opt_design -pre_cts
**opt_design ... cpu = 0:00:00, real = 0:00:00, mem = 2555.9M,
totSessionCpu=6:56:43 **
Executing: place_opt_design -opt
**INFO: User settings:
delaycal_default_net_delay                          1000ps
delaycal_default_net_load                           0.5pf
delaycal_default_net_load_ignore_for_ilm            0
delaycal_ignore_net_load                            false
delaycal_use_default_delay_limit                    1000
setAnalysisMode -cts                                postCTS
setAnalysisMode -virtualIPO                         false
setDelayCalMode -engine                             aae
design_bottom_routing_layer                         M2
design_process_node                                 65
design_top_routing_layer                            M5
extract_rc_coupling_cap_threshold                   0.1
extract_rc_engine                                   pre_route
extract_rc_relative_cap_threshold                   1.0
extract_rc_total_cap_threshold                      0.0
```

```
***   place_opt_design   #1   [begin]   :   totSession   cpu/real   =
6:56:42.7/45:38:53.7 (0.2), mem = 3288.9M
No user sequential activity specified, applying default sequential
activity of "0.2" for Dynamic Power reporting.
'set_default_switching_activity' finished successfully.
*** Starting GigaPlace ***
#optDebug: fT-E <X 2 3 1 0>
***   GlobalPlace   #1   [begin]   :   totSession   cpu/real   =
6:56:42.8/45:38:53.7 (0.2), mem = 3318.9M
*** GlobalPlace #1 [finish] : cpu/real = 0:00:00.0/0:00:00.0 (0.0),
totSession cpu/real = 6:56:42.8/45:38:53.7 (0.2), mem = 3318.9M
**opt_design ... cpu = 0:00:00, real = 0:00:00, mem = 2563.7M,
totSessionCpu=6:56:43 **
**INFO: set_db design_flow_effort standard -> setting 'set_db
opt_all_end_points true' for the duration of this command.
*** InitOpt #1 [begin] : totSession cpu/real = 6:56:42.8/45:38:53.7
(0.2), mem = 3318.9M
GigaOpt running with 1 threads.
Updating RC grid for preRoute extraction ...
Initializing multi-corner resistance tables ...
AAE DB initialization (MEM=3324.89 CPU=0:00:00.0 REAL=0:00:00.0)
Setting timing_disable_library_data_to_data_checks to 'true'.
Setting timing_disable_user_data_to_data_checks to 'true'.
**opt_design ... cpu = 0:00:02, real = 0:00:03, mem = 2549.1M,
totSessionCpu=6:56:45 **
*** opt_design -pre_cts ***
DRC Margin: user margin 0.0; extra margin 0.2
Setup Target Slack: user slack 0; extra slack 0.0
Hold Target Slack: user slack 0
**WARN: (IMPOPT-3195):     Analysis mode has changed.
Type 'man IMPOPT-3195' for more detail.
Multi-VT timing optimization disabled based on library information.
[NR-eGR] Started Early Global Route kernel ( Curr Mem: 3324.89 MB )
[NR-eGR] Num Prerouted Nets = 0  Num Prerouted Wires = 0
[NR-eGR] Read numTotalNets=4188  numIgnoredNets=0
[NR-eGR] There are 1 clock nets ( 0 with NDR ).
[NR-eGR] ============ Routing rule table ============
[NR-eGR] Rule id: 0  Nets: 4188
[NR-eGR] ========================================
[NR-eGR]
[NR-eGR] Layer group 1: route 4188 net(s) in layer range [2, 5]
[NR-eGR]  Early Global Route overflow of layer group 1: 0.13% H +
0.00% V. EstWL: 1.894608e+05um
[NR-eGR] Overflow after Early Global Route (GR compatible) 0.08% H +
0.00% V
[NR-eGR] Overflow after Early Global Route 0.11% H + 0.00% V
[NR-eGR]  ------------------------------------------------------------
----------------
[NR-eGR]      M1  (1F) length: 0.000000e+00um, number of vias: 12874
[NR-eGR]      M2  (2H) length: 6.356315e+04um, number of vias: 22583
[NR-eGR]      M3  (3V) length: 3.553300e+04um, number of vias: 3653
[NR-eGR]      M4  (4H) length: 8.196267e+04um, number of vias: 720
[NR-eGR]      M5  (5V) length: 1.113550e+04um, number of vias: 0
[NR-eGR] Total length: 1.921943e+05um, number of vias: 39830
[NR-eGR]  ------------------------------------------------------------
----------------
[NR-eGR] Total eGR-routed clock nets wire length: 2.591127e+03um
```

```
[NR-eGR] ------------------------------------------------------------
---------------
[NR-eGR] Finished Early Global Route kernel ( CPU: 0.23 sec, Real:
0.27 sec, Curr Mem: 3324.81 MB )
Extraction called for design 'top_32b' of instances=4664 and
nets=4190 using extraction engine 'pre_route' .
pre_route RC Extraction called for design top_32b.
RC Extraction called in multi-corner(2) mode.
RCMode: PreRoute
      RC Corner Indexes             0       1
Capacitance Scaling Factor   : 1.00000 1.00000
Resistance Scaling Factor    : 1.00000 1.00000
Clock Cap. Scaling Factor    : 1.00000 1.00000
Clock Res. Scaling Factor    : 1.00000 1.00000
Shrink Factor                : 1.00000
PreRoute extraction is honoring NDR/Shielding/ExtraSpace for clock
nets.
Using Quantus QRC technology file ...
Updating RC grid for preRoute extraction ...
Initializing multi-corner resistance tables ...
PreRoute RC Extraction DONE (CPU Time: 0:00:00.1   Real Time:
0:00:00.0  MEM: 3324.809M)
Starting delay calculation for Setup views
########################################################################
###############
# Design Stage: PreRoute
# Design Name: top_32b
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
########################################################################
###############
Start delay calculation (fullDC) (1 T). (MEM=3337.36)
Total number of fetched objects 4188
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
End delay calculation. (MEM=3428.46 CPU=0:00:00.9 REAL=0:00:01.0)
End delay calculation (fullDC). (MEM=3372.77 CPU=0:00:01.3
REAL=0:00:01.0)
***  Done  Building  Timing  Graph  (cpu=0:00:01.7  real=0:00:02.0
totSessionCpu=6:56:48 mem=3372.8M)

------------------------------------------------------------------
             Initial Summary
------------------------------------------------------------------

Setup views included:
 analysis_setup_wc

+--------------------+---------+
|     Setup mode     |   all   |
+--------------------+---------+
|           WNS (ns):| -23.927 |
|           TNS (ns):|-11866.6 |
|    Violating Paths:|   629   |
|          All Paths:|  1408   |
+--------------------+---------+
```

This is reporting the current status of the slack (WNS – worst negative slack), which is above 20ns. As the place command did not optimize the timing, this is expected.

```
+----------------+--------------------------------+-----------------
-+
|                |              Real              |       Total      |
|     DRVs       +------------------+-------------+------------------
-|
|                |  Nr nets(terms)  | Worst Vio   |  Nr nets(terms)
|
+----------------+------------------+-------------+------------------
-+
|   max_cap      |     50 (50)      |   -1.214    |     50 (50)      |
|   max_tran     |    228 (1413)    |  -28.054    |    228 (1413)    |
|   max_fanout   |      7 (7)       |    -444     |      7 (7)       |
|   max_length   |      0 (0)       |      0      |      0 (0)       |
+----------------+------------------+-------------+------------------
-+

Density: 23.599%
```

The initial density is around 24%, and this will be the starting point of the optimization.

```
---------------------------------------------------------------------
**opt_design ... cpu = 0:00:05, real = 0:00:06, mem = 2616.2M,
totSessionCpu=6:56:48 **
*** InitOpt #1 [finish] : cpu/real = 0:00:05.3/0:00:06.5 (0.8),
totSession cpu/real = 6:56:48.1/45:39:00.2 (0.2), mem = 3343.0M
** INFO : this run is activating medium effort placeOptDesign flow
*** Starting optimizing excluded clock nets MEM= 3343.0M) ***
*info: No excluded clock nets to be optimized.
*** Finished optimizing excluded clock nets (CPU Time= 0:00:00.0
MEM= 3343.0M) ***
The useful skew maximum allowed delay is: 0.3
***   SimplifyNetlist   #1   [begin]   :   totSession   cpu/real   =
6:56:48.4/45:39:00.5 (0.2), mem = 3343.0M
Info: 1 clock net  excluded from IPO operation.

Footprint cell information for calculating maxBufDist
*info: There are 22 candidate Buffer cells
*info: There are 20 candidate Inverter cells


Netlist preparation processing...
Removed 0 instance
*info: Marking 0 isolation instances dont touch
*info: Marking 0 level shifter instances dont touch
*** SimplifyNetlist #1 [finish] : cpu/real = 0:00:02.2/0:00:02.2
(1.0), totSession cpu/real = 6:56:50.6/45:39:02.7 (0.2), mem =
3397.4M
Begin: GigaOpt high fanout net optimization
GigaOpt HFN: use maxLocalDensity 1.2
GigaOpt Checkpoint: Internal optDRV -useLevelizedBufferTreeOnly -
auxMaxFanoutCountLimit   500   -largeScaleFixing   -maxIter   1   -
```

```
maxLocalDensity      1.2      -numThreads      1      -preCTS      -
preRouteDontEndWithRefinePlaceIncrSteinerRouteDC
*** DrvOpt #1 [begin] : totSession cpu/real = 6:56:50.7/45:39:02.7
(0.2), mem = 3397.4M
Info: 1 clock net  excluded from IPO operation.
*** DrvOpt #1 [finish] : cpu/real = 0:00:02.0/0:00:02.0 (1.0),
totSession cpu/real = 6:56:52.7/45:39:04.7 (0.2), mem = 3397.4M
GigaOpt HFN: restore maxLocalDensity to 0.98
End: GigaOpt high fanout net optimization
Begin: GigaOpt DRV Optimization
GigaOpt Checkpoint: Internal optDRV -max_tran -max_cap -
maxLocalDensity 1.2 -numThreads 1 -largeScaleFixing -maxIter 2 -
preCTS -preRouteDontEndWithRefinePlaceIncrSteinerRouteDC
*** DrvOpt #2 [begin] : totSession cpu/real = 6:56:52.7/45:39:04.7
(0.2), mem = 3397.4M
Info: 1 clock net  excluded from IPO operation.
+--------------------------------------------------------------------------------------------
-----------------------------------------------------+
|       max-tran       |        max-cap        | max-fanout |  max-length |       setup
|        |       |      |       |       |      |
+--------------------------------------------------------------------------------------------
-----------------------------------------------------+
| nets | terms|  wViol  | nets | terms|  wViol  | nets | terms| nets | terms|   WNS   |   TNS
|  #Buf  |  #Inv  | #Resize|Density|   Real   |   Mem   |
+--------------------------------------------------------------------------------------------
-----------------------------------------------------+
|   263|  1688|   -28.49|    70|    70|    -1.23|     7|     7|     0|     0|   -23.93|-
11866.63|       0|       0|       0| 23.60%|          |         |
|     3|     3|    -0.02|     0|     0|     0.00|    12|    12|     0|     0|    -3.30|  -
154.13|     146|       5|      57| 24.07%| 0:00:01.0|  3476.8M|
|     0|     0|     0.00|     0|     0|     0.00|    12|    12|     0|     0|    -3.30|  -
154.13|       0|       0|       3| 24.08%| 0:00:00.0|  3476.8M|
+--------------------------------------------------------------------------------------------
-----------------------------------------------------+
```

Check these tables from the innovus report, which is easier to read

```
*** Finish DRV Fixing (cpu=0:00:01.2 real=0:00:01.0 mem=3476.8M) ***

*** DrvOpt #2 [finish] : cpu/real = 0:00:02.2/0:00:02.2 (1.0),
totSession cpu/real = 6:56:54.8/45:39:06.9 (0.2), mem = 3409.7M
End: GigaOpt DRV Optimization
GigaOpt DRV: restore maxLocalDensity to 0.98
**opt_design ... cpu = 0:00:12, real = 0:00:13, mem = 2696.1M,
totSessionCpu=6:56:55 **

Active setup views:
 analysis_setup_wc
  Dominating endpoints: 0
  Dominating TNS: -0.000

Begin: GigaOpt Global Optimization
*info: use new DP (enabled)
GigaOpt Checkpoint: Internal globalOpt -maxLocalDensity 1.2 -
numThreads 1 -preCTS -rebufferAll -
preRouteDontEndWithRefinePlaceIncrSteinerRouteDC -
enableHighLayerOpt -maxIter 50 -maxIterForLEPG 50
Info: 1 clock net  excluded from IPO operation.
*** GlobalOpt #1 [begin] : totSession cpu/real = 6:56:54.9/45:39:07.0
(0.2), mem = 3409.7M
*info: 1 clock net excluded
*info: 2 special nets excluded.
*info: 2 no-driver nets excluded.
```

```
** GigaOpt Global Opt WNS Slack -3.299  TNS Slack -154.130
+--------+--------+---------+------------+--------+----------------
-+---------+-----------------------------------------+
|  WNS   |  TNS   | Density |     Real     |  Mem   |    Worst View
|Pathgroup|                End Point                |
+--------+--------+---------+------------+--------+----------------
-+---------+-----------------------------------------+
| -3.299|-154.130|   24.08%|   0:00:00.0| 3428.8M|analysis_setup_wc|
default| I_MULT/res_reg_63_/D                        |
| -2.579|-112.102|   24.17%|   0:00:02.0| 3474.5M|analysis_setup_wc|
default| I_MULT/res_reg_30_/D                        |
| -1.436| -49.799|   24.63%|   0:00:02.0| 3474.5M|analysis_setup_wc|
default| I_MULT/res_reg_30_/D                        |
| -1.436| -49.799|   24.63%|   0:00:00.0| 3474.5M|analysis_setup_wc|
default| I_MULT/res_reg_30_/D                        |
| -0.509| -11.941|   25.25%|   0:00:03.0| 3478.6M|analysis_setup_wc|
default| I_MULT/res_reg_50_/D                        |
| -0.340|  -7.344|   25.28%|   0:00:02.0| 3483.1M|analysis_setup_wc|
default| I_MULT/res_reg_25_/D                        |
| -0.293|  -4.873|   25.37%|   0:00:01.0| 3485.1M|analysis_setup_wc|
default| I_MULT/res_reg_25_/D                        |
| -0.293|  -4.873|   25.37%|   0:00:00.0| 3485.1M|analysis_setup_wc|
default| I_MULT/res_reg_25_/D                        |
| -0.030|  -0.177|   25.49%|   0:00:00.0| 3485.1M|analysis_setup_wc|
default| I_MULT/res_reg_59_/D                        |
|  0.000|   0.000|   25.51%|   0:00:00.0| 3485.1M|              NA|
NA| NA                                              |
+--------+--------+---------+------------+--------+----------------
-+---------+-----------------------------------------+

*** Finish pre-CTS Global Setup Fixing (cpu=0:00:09.8 real=0:00:10.0
mem=3485.1M) ***

***  Finish  pre-CTS  Setup  Fixing  (cpu=0:00:09.8  real=0:00:10.0
mem=3485.1M) ***
** GigaOpt Global Opt End WNS Slack 0.000  TNS Slack 0.000
*** GlobalOpt #1 [finish] : cpu/real = 0:00:12.3/0:00:12.3 (1.0),
totSession cpu/real = 6:57:07.2/45:39:19.2 (0.2), mem = 3416.0M
End: GigaOpt Global Optimization
*** Timing Is met
*** Check timing (0:00:00.0)
GigaOpt Checkpoint: Internal reclaim -numThreads 1 -preCTS -force -
doRemoveUselessTerm      -tgtSlackMult      3      -routeType      -
noRouteTypeResizePolish   -noViewPrune   -weedwhack   -nonLegal   -
nativePathGroupFlow
Info: 1 clock net  excluded from IPO operation.
Begin: Area Reclaim Optimization
*** AreaOpt #1 [begin] : totSession cpu/real = 6:57:07.3/45:39:19.4
(0.2), mem = 3435.1M
Reclaim Optimization WNS Slack 0.000  TNS Slack 0.000 Density 25.51
+---------+---------+--------+--------+------------+--------+
| Density | Commits |  WNS   |  TNS   |    Real    |  Mem   |
+---------+---------+--------+--------+------------+--------+
|   25.51%|        -|   0.000|   0.000|   0:00:00.0| 3437.1M|
|   25.51%|        0|   0.000|   0.000|   0:00:00.0| 3437.1M|
|   25.49%|        5|   0.000|   0.000|   0:00:00.0| 3479.8M|
|   24.95%|      282|   0.000|   0.000|   0:00:03.0| 3479.8M|
|   24.92%|       23|   0.000|   0.000|   0:00:01.0| 3479.8M|
```

```
|   24.92%|        0|   0.000|   0.000|   0:00:00.0| 3479.8M|
|   24.92%|        0|   0.000|   0.000|   0:00:00.0| 3479.8M|
+---------+---------+--------+--------+------------+--------+
Reclaim Optimization End WNS Slack 0.000  TNS Slack 0.000 Density
24.92
End: Core Area Reclaim Optimization (cpu = 0:00:05.4) (real =
0:00:06.0) **
*** AreaOpt #1 [finish] : cpu/real = 0:00:05.4/0:00:05.4 (1.0),
totSession cpu/real = 6:57:12.7/45:39:24.8 (0.2), mem = 3479.8M
Executing incremental physical updates
Executing incremental physical updates
End: Area Reclaim Optimization (cpu=0:00:05, real=0:00:06,
mem=3417.68M, totSessionCpu=6:57:13).
*** IncrReplace #1 [begin] : totSession cpu/real =
6:57:12.8/45:39:24.9 (0.2), mem = 3417.7M

*** Start incrementalPlace ***
No Views given, use default active views for adaptive view pruning
SKP will enable view:
  analysis_setup_wc
[NR-eGR] Num Prerouted Nets = 0  Num Prerouted Wires = 0
[NR-eGR] Read numTotalNets=4444  numIgnoredNets=0
[NR-eGR] There are 1 clock nets ( 0 with NDR ).
[NR-eGR] ============ Routing rule table ============
[NR-eGR] Rule id: 0  Nets: 4444
[NR-eGR] ========================================
[NR-eGR]
[NR-eGR] Layer group 1: route 4444 net(s) in layer range [2, 5]
[NR-eGR] Early Global Route overflow of layer group 1: 0.15% H +
0.00% V. EstWL: 1.895004e+05um
[NR-eGR] Overflow after Early Global Route (GR compatible) 0.12% H +
0.00% V
[NR-eGR] Overflow after Early Global Route 0.16% H + 0.00% V
Early Global Route congestion estimation runtime: 0.17 seconds, mem
= 3417.7M
Local HotSpot Analysis: normalized max congestion hotspot area =
0.44, normalized total congestion hotspot area = 0.89 (area is in
unit of 4 std-cell row bins)

=== incrementalPlace Internal Loop 1 ===
*** Finished SKP initialization (cpu=0:00:00.4, real=0:00:01.0)***
Iteration  8: Total net bbox = 1.770e+05 (1.41e+05 3.56e+04)
               Est.  stn bbox = 2.078e+05 (1.69e+05 3.89e+04)
               cpu = 0:00:00.4 real = 0:00:00.0 mem = 3397.2M
Iteration  9: Total net bbox = 1.752e+05 (1.39e+05 3.63e+04)
               Est.  stn bbox = 2.045e+05 (1.65e+05 3.97e+04)
               cpu = 0:00:00.5 real = 0:00:01.0 mem = 3395.2M
Iteration 10: Total net bbox = 1.753e+05 (1.39e+05 3.66e+04)
               Est.  stn bbox = 2.046e+05 (1.65e+05 4.00e+04)
               cpu = 0:00:08.3 real = 0:00:08.0 mem = 3393.2M
Iteration 11: Total net bbox = 1.766e+05 (1.39e+05 3.78e+04)
               Est.  stn bbox = 2.057e+05 (1.64e+05 4.12e+04)
               cpu = 0:00:05.6 real = 0:00:06.0 mem = 3399.2M
Iteration 12: Total net bbox = 1.778e+05 (1.39e+05 3.84e+04)
               Est.  stn bbox = 2.069e+05 (1.65e+05 4.18e+04)
               cpu = 0:00:00.5 real = 0:00:00.0 mem = 3395.2M
Move report: Timing Driven Placement moves 3863 insts, mean move:
15.59 um, max move: 92.46 um
```

```
     Max move on inst (I_MULT/FE_OFC239_reg_op2_30): (560.40,
115.40) --> (647.16, 121.10)

Finished Incremental Placement (cpu=0:00:16.2, real=0:00:16.0,
mem=3395.2M)
**WARN: (IMPSP-9025):No scan chain specified/traced.
Type 'man IMPSP-9025' for more detail.
*** Starting place_detail (6:57:29 mem=3395.9M) ***
Total net bbox length = 1.788e+05 (1.402e+05 3.864e+04) (ext =
2.577e+04)
Move report: Detail placement moves 3863 insts, mean move: 0.75 um,
max move: 27.92 um
     Max move on inst (FE_OFC149_addr_mem2_1): (769.90, 73.98) -->
(797.80, 74.00)
     Runtime: CPU: 0:00:00.3 REAL: 0:00:01.0 MEM: 3395.9MB
Summary Report:
Instances move: 3863 (out of 3863 movable)
Instances flipped: 0
Mean displacement: 0.75 um
Max displacement: 27.92 um (Instance: FE_OFC149_addr_mem2_1) (769.9,
73.981) -> (797.8, 74)
     Length: 4 sites, height: 1 rows, site name: CP65_DST, cell type:
SEN_BUF_1
     Violation at original loc: Placement Blockage Violation
Total net bbox length = 1.760e+05 (1.373e+05 3.864e+04) (ext =
2.520e+04)
Runtime: CPU: 0:00:00.3 REAL: 0:00:01.0 MEM: 3395.9MB
*** Finished place_detail (6:57:30 mem=3395.9M) ***
[NR-eGR] Num Prerouted Nets = 0  Num Prerouted Wires = 0
[NR-eGR] Read numTotalNets=4444  numIgnoredNets=0
[NR-eGR] There are 1 clock nets ( 0 with NDR ).
[NR-eGR] ============ Routing rule table ============
[NR-eGR] Rule id: 0  Nets: 4444
[NR-eGR] ========================================
[NR-eGR]
[NR-eGR] Layer group 1: route 4444 net(s) in layer range [2, 5]
[NR-eGR] Early Global Route overflow of layer group 1: 0.15% H +
0.00% V. EstWL: 1.840842e+05um
[NR-eGR] Overflow after Early Global Route (GR compatible) 0.10% H +
0.00% V
[NR-eGR] Overflow after Early Global Route 0.13% H + 0.00% V
Early Global Route congestion estimation runtime: 0.17 seconds, mem
= 3393.9M
Local HotSpot Analysis: normalized max congestion hotspot area =
0.89, normalized total congestion hotspot area = 1.33 (area is in
unit of 4 std-cell row bins)
[NR-eGR] ------------------------------------------------------------
----------------
[NR-eGR]      M1  (1F) length: 0.000000e+00um, number of vias: 13416
[NR-eGR]      M2  (2H) length: 6.043116e+04um, number of vias: 22887
[NR-eGR]      M3  (3V) length: 3.702324e+04um, number of vias: 3751
[NR-eGR]      M4  (4H) length: 7.965043e+04um, number of vias: 739
[NR-eGR]      M5  (5V) length: 9.729430e+03um, number of vias: 0
[NR-eGR] Total length: 1.868343e+05um, number of vias: 40793
[NR-eGR] ------------------------------------------------------------
----------------
[NR-eGR] Total eGR-routed clock nets wire length: 2.269565e+03um
```

```
[NR-eGR] ------------------------------------------------------------
---------------
Early Global Route wiring runtime: 0.11 seconds, mem = 3393.9M
0 delay mode for cte disabled.

*** Finished incrementalPlace (cpu=0:00:17.1, real=0:00:17.0)***
Start to check current routing status for nets...
All nets are already routed correctly.
End to check current routing status for nets (mem=3373.9M)
Extraction called for design 'top_32b' of instances=4920 and
nets=4446 using extraction engine 'pre_route' .
pre_route RC Extraction called for design top_32b.
RC Extraction called in multi-corner(2) mode.
RCMode: PreRoute
      RC Corner Indexes            0       1
Capacitance Scaling Factor   : 1.00000 1.00000
Resistance Scaling Factor    : 1.00000 1.00000
Clock Cap. Scaling Factor    : 1.00000 1.00000
Clock Res. Scaling Factor    : 1.00000 1.00000
Shrink Factor                : 1.00000
PreRoute extraction is honoring NDR/Shielding/ExtraSpace for clock
nets.
Using Quantus QRC technology file ...
Updating RC grid for preRoute extraction ...
Initializing multi-corner resistance tables ...
PreRoute RC Extraction DONE (CPU Time: 0:00:00.1  Real Time:
0:00:00.0  MEM: 3373.914M)
Compute RC Scale Done ...
**opt_design ... cpu = 0:00:48, real = 0:00:49, mem = 2671.7M,
totSessionCpu=6:57:31 **
################################################################
###############
# Design Stage: PreRoute
# Design Name: top_32b
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
################################################################
###############
Start delay calculation (fullDC) (1 T). (MEM=3378.46)
Total number of fetched objects 4444
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
End delay calculation. (MEM=3421.88 CPU=0:00:00.8 REAL=0:00:01.0)
End delay calculation (fullDC). (MEM=3421.88 CPU=0:00:01.0
REAL=0:00:01.0)
*** IncrReplace #1 [finish] : cpu/real = 0:00:19.4/0:00:19.5 (1.0),
totSession cpu/real = 6:57:32.3/45:39:44.4 (0.2), mem = 3421.9M
*** Timing NOT met, worst failing slack is -0.193
*** Check timing (0:00:00.0)
Begin: GigaOpt Optimization in WNS mode
GigaOpt Checkpoint: Internal optTiming -maxLocalDensity 1.0 -
maxLocalDensityForHardenOpt 0.92 -numThreads 1 -preCTS -wtns -
integratedAreaOpt -pgMode all -ipoTgtSlackCoef 1.5 -effTgtSlackCoef
1    -nativePathGroupFlow    -NDROptEffortAuto    -usefulSkew    -
nonLegalPlaceEcoBumpRecoveryInWNSOpt
Info: 1 clock net  excluded from IPO operation.
```

```
*** WnsOpt #1 [begin] : totSession cpu/real = 6:57:32.4/45:39:44.5
(0.2), mem = 3437.9M
*info: 1 clock net excluded
*info: 2 special nets excluded.
*info: 2 no-driver nets excluded.
** GigaOpt Optimizer WNS Slack -0.193 TNS Slack -0.422 Density 24.92
OptDebug: Start of Optimizer WNS Pass 0:
+----------+------+------+
|Path Group|   WNS|   TNS|
+----------+------+------+
|default   | 0.643| 0.000|
|reg2reg   |-0.193|-0.422|
|HEPG      |-0.193|-0.422|
|All Paths |-0.193|-0.422|
+----------+------+------+

Active Path Group: reg2reg
+--------+---------+--------+---------+---------+------------+-----
---+-----------------+---------+-----------------------------------
------+
|  WNS   | All WNS |  TNS   | All TNS | Density |    Real    |  Mem
|   Worst View    |Pathgroup|               End Point               |
+--------+---------+--------+---------+---------+------------+-----
---+-----------------+---------+-----------------------------------
------+
|  -0.193|   -0.193|  -0.422|   -0.422|   24.92%|   0:00:00.0|
3473.0M|analysis_setup_wc|                  reg2reg|         MEM0/ME
|
|   0.008|    0.008|   0.000|    0.000|   24.95%|   0:00:00.0|
3499.6M|analysis_setup_wc|         reg2reg|    I_MULT/res_reg_63_/D
|
|   0.015|    0.015|   0.000|    0.000|   24.96%|   0:00:01.0|
3499.6M|analysis_setup_wc|         reg2reg|    I_MULT/res_reg_63_/D
|
|   0.029|    0.029|   0.000|    0.000|   24.98%|   0:00:00.0|
3499.6M|analysis_setup_wc|         reg2reg|    I_MULT/res_reg_56_/D
|
|   0.029|    0.029|   0.000|    0.000|   24.98%|   0:00:00.0|
3499.6M|analysis_setup_wc|         reg2reg|    I_MULT/res_reg_56_/D
|
+--------+---------+--------+---------+---------+------------+-----
---+-----------------+---------+-----------------------------------
------+
```

Here it is progressively optimizing the reg2reg slack using the wc corner.

```
***   Finish   Core   Optimize   Step   (cpu=0:00:00.6   real=0:00:01.0
mem=3499.6M)  ***

*** Finished Optimize Step Cumulative (cpu=0:00:00.6 real=0:00:01.0
mem=3499.6M)  ***
OptDebug: End of Optimizer WNS Pass 0:
+----------+-----+-----+
|Path Group|  WNS|  TNS|
+----------+-----+-----+
|default   |1.209|0.000|
|reg2reg   |0.029|0.000|
|HEPG      |0.029|0.000|
|All Paths |0.029|0.000|
```

```
+----------+-----+-----+

** GigaOpt Optimizer WNS Slack 0.029 TNS Slack 0.000 Density 24.98
*** Starting place_detail (6:57:36 mem=3499.6M) ***
Total net bbox length = 1.768e+05 (1.381e+05 3.869e+04) (ext =
2.520e+04)

Starting Small incrNP...
Skipped incrNP (cpu=0:00:00.0, real=0:00:00.0, mem=3499.6M)
End of Small incrNP (cpu=0:00:00.0, real=0:00:00.0)
Move report: Detail placement moves 25 insts, mean move: 0.97 um,
max move: 3.00 um
     Max move on inst (I_MULT/FE_RC_2_0): (534.20, 108.20) -->
(537.20, 108.20)
     Runtime: CPU: 0:00:00.1 REAL: 0:00:00.0 MEM: 3502.6MB
Summary Report:
Instances move: 25 (out of 3872 movable)
Instances flipped: 0
Mean displacement: 0.97 um
Max displacement: 3.00 um (Instance: I_MULT/FE_RC_2_0) (534.2, 108.2)
-> (537.2, 108.2)
     Length: 8 sites, height: 1 rows, site name: CP65_DST, cell type:
SEN_INV_AS_5
Total net bbox length = 1.768e+05 (1.381e+05 3.870e+04) (ext =
2.520e+04)
Runtime: CPU: 0:00:00.1 REAL: 0:00:00.0 MEM: 3502.6MB
*** Finished place_detail (6:57:36 mem=3502.6M) ***
*** maximum move = 3.00 um ***
*** Finished re-routing un-routed nets (3499.6M) ***

***   Finish   Physical   Update   (cpu=0:00:00.3   real=0:00:00.0
mem=3499.6M) ***
** GigaOpt Optimizer WNS Slack 0.029 TNS Slack 0.000 Density 24.98
OptDebug: End of Setup Fixing:
+----------+-----+-----+
|Path Group|  WNS|  TNS|
+----------+-----+-----+
|default   |1.209|0.000|
|reg2reg   |0.029|0.000|
|HEPG      |0.029|0.000|
|All Paths |0.029|0.000|
+----------+-----+-----+


***  Finish  pre-CTS  Setup  Fixing  (cpu=0:00:01.2  real=0:00:01.0
mem=3499.6M) ***

***  WnsOpt  #1  [finish]  :  cpu/real  =  0:00:03.5/0:00:03.5  (1.0),
totSession cpu/real = 6:57:35.9/45:39:48.0 (0.2), mem = 3433.6M
End: GigaOpt Optimization in WNS mode
*** Timing Is met
*** Check timing (0:00:00.0)
GigaOpt Checkpoint: Internal reclaim -numThreads 1 -customPhyUpdate
-noGCompAndPhase   -ensureOneAreaReclaim   -force   -svrReclaim   -
rtrShortNets -preCTS -tgtSlackMult 2 -wtns -noRouteTypeResizePolish
-noViewPrune -nativePathGroupFlow
Info: 1 clock net  excluded from IPO operation.
Begin: Area Reclaim Optimization
```

```
*** AreaOpt #2 [begin] : totSession cpu/real = 6:57:36.0/45:39:48.2
(0.2), mem = 3452.6M
Reclaim Optimization WNS Slack 0.000  TNS Slack 0.000 Density 24.98
+---------+---------+--------+--------+------------+--------+
| Density | Commits |  WNS   |  TNS   |    Real    |  Mem   |
+---------+---------+--------+--------+------------+--------+
|   24.98%|        -|   0.000|   0.000|   0:00:00.0| 3452.6M|
|   24.98%|        0|   0.000|   0.000|   0:00:00.0| 3452.6M|
|   24.96%|        4|   0.000|   0.000|   0:00:00.0| 3495.3M|
|   24.84%|       77|   0.000|   0.000|   0:00:02.0| 3495.3M|
|   24.83%|        7|   0.000|   0.000|   0:00:00.0| 3495.3M|
|   24.83%|        0|   0.000|   0.000|   0:00:00.0| 3495.3M|
|   24.83%|        0|   0.000|   0.000|   0:00:00.0| 3495.3M|
+---------+---------+--------+--------+------------+--------+
Reclaim Optimization End WNS Slack 0.000  TNS Slack 0.000 Density
24.83
End: Core Area Reclaim Optimization (cpu = 0:00:02.9) (real =
0:00:03.0) **
*** Starting place_detail (6:57:39 mem=3495.3M) ***
Total net bbox length = 1.768e+05 (1.381e+05 3.870e+04) (ext =
2.520e+04)
Move report: Detail placement moves 0 insts, mean move: 0.00 um, max
move: 0.00 um
     Runtime: CPU: 0:00:00.0 REAL: 0:00:00.0 MEM: 3495.3MB
Summary Report:
Instances move: 0 (out of 3868 movable)
Instances flipped: 0
Mean displacement: 0.00 um
Max displacement: 0.00 um
Total net bbox length = 1.768e+05 (1.381e+05 3.870e+04) (ext =
2.520e+04)
Runtime: CPU: 0:00:00.1 REAL: 0:00:00.0 MEM: 3495.3MB
*** Finished place_detail (6:57:39 mem=3495.3M) ***
*** maximum move = 0.00 um ***
*** Finished re-routing un-routed nets (3495.3M) ***

***   Finish   Physical   Update   (cpu=0:00:00.2   real=0:00:00.0
mem=3495.3M) ***
*** AreaOpt #2 [finish] : cpu/real = 0:00:03.1/0:00:03.1 (1.0),
totSession cpu/real = 6:57:39.2/45:39:51.3 (0.2), mem = 3495.3M
End:   Area   Reclaim   Optimization   (cpu=0:00:03,   real=0:00:03,
mem=3433.23M, totSessionCpu=6:57:39).
Begin: GigaOpt postEco DRV Optimization
GigaOpt   Checkpoint:   Internal   optDRV   -inPostEcoStage   -
smallScaleFixing -maxIter 3 -max_tran -max_cap -maxLocalDensity 0.98
-numThreads  1  -preRouteDontEndWithRefinePlaceIncrSteinerRouteDC  -
preCTS
*** DrvOpt #3 [begin] : totSession cpu/real = 6:57:39.2/45:39:51.3
(0.2), mem = 3433.2M
Info: 1 clock net  excluded from IPO operation.
+-------------------------------------------------------------------
-------------------------------------------------------------------
------------------+
|        max-tran       |        max-cap        |  max-fanout |  max-
length |        setup        |         |         |         |        |
|         |
```

```
+--------------------------------------------------------------------
--------------------------------------------------------------------
------------------+
| nets | terms|  wViol  | nets | terms|  wViol  | nets | terms| nets
| terms|    WNS    |    TNS    |   #Buf   |   #Inv   | #Resize|Density|
Real    |   Mem    |
+--------------------------------------------------------------------
--------------------------------------------------------------------
------------------+
|      3|      3|    -0.02|      0|      0|      0.00|     12|     12|
0|      0|      0.01|      0.00|        0|        0|        0| 24.83%|
|          |
|      0|      0|      0.00|      0|      0|      0.00|     12|     12|
0|      0|      0.01|      0.00|        3|        0|        0| 24.83%|
0:00:00.0|  3498.1M|
|      0|      0|      0.00|      0|      0|      0.00|     12|     12|
0|      0|      0.01|      0.00|        0|        0|        0| 24.83%|
0:00:00.0|  3498.1M|
+--------------------------------------------------------------------
--------------------------------------------------------------------
------------------+

*** Finish DRV Fixing (cpu=0:00:00.1 real=0:00:00.0 mem=3498.1M) ***

***  DrvOpt  #3  [finish]  :  cpu/real  =  0:00:01.0/0:00:01.0  (1.0),
totSession cpu/real = 6:57:40.2/45:39:52.3 (0.2), mem = 3435.0M
End: GigaOpt postEco DRV Optimization
Running refinePlace -preserveRouting true -hardFence false
*** Starting place_detail (6:57:40 mem=3435.0M) ***

Starting Small incrNP...
Skipped incrNP (cpu=0:00:00.0, real=0:00:00.0, mem=3435.0M)
End of Small incrNP (cpu=0:00:00.0, real=0:00:00.0)
Move report: Detail placement moves 3 insts, mean move: 0.67 um, max
move: 1.00 um
     Max move on inst (U543): (334.40, 74.00) --> (335.40, 74.00)
     Runtime: CPU: 0:00:00.1 REAL: 0:00:00.0 MEM: 3435.0MB
Summary Report:
Instances move: 3 (out of 3871 movable)
Instances flipped: 0
Mean displacement: 0.67 um
Max displacement: 1.00 um (Instance: U543) (334.4, 74) -> (335.4,
74)
     Length: 4 sites, height: 1 rows, site name: CP65_DST, cell type:
SEN_NR2_S_0P5
Runtime: CPU: 0:00:00.1 REAL: 0:00:00.0 MEM: 3435.0MB
*** Finished place_detail (6:57:40 mem=3435.0M) ***

Active setup views:
 analysis_setup_wc
  Dominating endpoints: 0
  Dominating TNS: -0.000

Extraction  called  for  design  'top_32b'  of  instances=4928  and
nets=4454 using extraction engine 'pre_route' .
pre_route RC Extraction called for design top_32b.
RC Extraction called in multi-corner(2) mode.
RCMode: PreRoute
```

```
      RC Corner Indexes             0       1
Capacitance Scaling Factor   : 1.00000 1.00000
Resistance Scaling Factor    : 1.00000 1.00000
Clock Cap. Scaling Factor    : 1.00000 1.00000
Clock Res. Scaling Factor    : 1.00000 1.00000
Shrink Factor                : 1.00000
PreRoute extraction is honoring NDR/Shielding/ExtraSpace for clock
nets.
Using Quantus QRC technology file ...
Skipped RC grid update for preRoute extraction.
Initializing multi-corner resistance tables ...
PreRoute RC Extraction DONE (CPU Time: 0:00:00.1   Real Time:
0:00:00.0  MEM: 3399.273M)
Skewing Data Summary (End_of_FINAL)
--------------------------------------------------
 Total skewed count:0
--------------------------------------------------
Starting delay calculation for Setup views
########################################################################
###############
# Design Stage: PreRoute
# Design Name: top_32b
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
########################################################################
###############
Start delay calculation (fullDC) (1 T). (MEM=3403.82)
Total number of fetched objects 4452
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
End delay calculation. (MEM=3439.23 CPU=0:00:00.8 REAL=0:00:01.0)
End  delay  calculation  (fullDC).  (MEM=3439.23  CPU=0:00:01.0
REAL=0:00:01.0)
***  Done  Building  Timing  Graph  (cpu=0:00:01.2  real=0:00:01.0
totSessionCpu=6:57:42 mem=3439.2M)
[NR-eGR] Num Prerouted Nets = 0  Num Prerouted Wires = 0
[NR-eGR] Read numTotalNets=4452  numIgnoredNets=0
[NR-eGR] There are 1 clock nets ( 0 with NDR ).
[NR-eGR] ============ Routing rule table ============
[NR-eGR] Rule id: 0  Nets: 4452
[NR-eGR] ========================================
[NR-eGR]
[NR-eGR] Layer group 1: route 4452 net(s) in layer range [2, 5]
[NR-eGR] Early Global Route overflow of layer group 1: 0.14% H +
0.00% V. EstWL: 1.849824e+05um
[NR-eGR] Overflow after Early Global Route (GR compatible) 0.08% H +
0.00% V
[NR-eGR] Overflow after Early Global Route 0.11% H + 0.00% V
[NR-eGR] Finished Early Global Route kernel ( CPU: 0.15 sec, Real:
0.20 sec, Curr Mem: 3447.24 MB )
[hotspot] +------------+---------------+---------------+
[hotspot] |            |   max hotspot | total hotspot |
[hotspot] +------------+---------------+---------------+
[hotspot] | normalized |          0.89 |          1.78 |
[hotspot] +------------+---------------+---------------+
```

```
Local HotSpot Analysis: normalized max congestion hotspot area =
0.89, normalized total congestion hotspot area = 1.78 (area is in
unit of 4 std-cell row bins)
[hotspot] top 3 congestion hotspot bounding boxes and scores of
normalized hotspot
[hotspot] +-----+---------------------------------------+------------
---+
[hotspot] | top |            hotspot bbox               | hotspot score
|
[hotspot] +-----+---------------------------------------+------------
---+
[hotspot] |  1  |   331.40    72.20   345.80    86.60 |        0.89
|
[hotspot] +-----+---------------------------------------+------------
---+
[hotspot] |  2  |   662.60    65.00   677.00    79.40 |        0.44
|
[hotspot] +-----+---------------------------------------+------------
---+
[hotspot] |  3  |   662.60    79.40   677.00    93.80 |        0.44
|
[hotspot] +-----+---------------------------------------+------------
---+
Reported timing to dir ./timingReports
**opt_design ... cpu = 0:00:59, real = 0:01:00, mem = 2747.5M,
totSessionCpu=6:57:42 **

------------------------------------------------------------------------
     opt_design Final Summary
------------------------------------------------------------------------

Setup views included:
 analysis_setup_wc

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.005  |  0.005  |  1.209  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  1408   |   650   |   762   |
+--------------------+---------+---------+---------+

+----------------+-----------------------------------+-----------------
-+
|                |              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   |     12 (12)      |    -25     |     12 (12)      |
|   max_length   |      0 (0)       |     0      |      0 (0)       |
+----------------+------------------+------------+-----------------
-+
```

```
Density: 24.833%
Routing Overflow: 0.11% H and 0.00% V
------------------------------------------------------------------
**opt_design ... cpu = 0:01:00, real = 0:01:02, mem = 2750.1M,
totSessionCpu=6:57:42 **
**WARN: (IMPOPT-3195):     Analysis mode has changed.
Type 'man IMPOPT-3195' for more detail.
*** Finished opt_design ***
      flow.cputime          flow.realtime          timing.setup.tns
timing.setup.wns  snapshot
UM:*                                        0.000 ns          0.005
ns  final
------------------------------------------------------------
     Current design flip-flop statistics

Single-Bit FF Count  :          500
Multi-Bit FF Count   :            0
Total Bit Count      :          500
Total FF Count       :          500
Bits Per Flop        :        1.000
------------------------------------------------------------
      flow.cputime          flow.realtime          timing.setup.tns
timing.setup.wns  snapshot
UM:                          88.08                                222
opt_design_prects
#optDebug: fT-D <X 1 0 0 0>
**place_opt_design ... cpu = 0:01:00, real = 0:01:03, mem = 3373.5M
**
*** Finished GigaPlace ***

*** Summary of all messages that are not suppressed in this session:
Severity  ID               Count  Summary
WARNING   IMPSP-9025           1  No scan chain specified/traced.
WARNING   IMPOPT-3195          2  Analysis mode has changed.
*** Message Summary: 3 warning(s), 0 error(s)

*** place_opt_design #1 [finish] : cpu/real = 0:01:00.1/0:01:03.1
(1.0), totSession cpu/real = 6:57:42.8/45:39:56.8 (0.2), mem =
3373.5M
```

At this point the optimization is finished and the tool did not point any warnings or errors.

Let's check the worst setup path and check it. The -late option puts the report_timing in setup check, and extracts the corresponding worst setup time. Here as you can see the path being reported is in the multiplier. As post synthesis, it does show you the details of the path and the contribution of each gate in the timing analysis. Parasitics contributions are collapsed on the timing of the gates, the fanout column gives some information on the load of the corresponding gate.

```
@innovus 353> report_timing -late
############################################################
#  Generated by:      Cadence Innovus 20.14-s095_1
#  OS:                Linux x86_64(Host ID edasrv1)
#  Generated on:      Fri Nov 10 11:08:28 2023
#  Design:            top_32b
#  Command:           report_timing -late
############################################################
Path 1: MET (0.005 ns) Setup Check with Pin I_MULT/res_reg_22_/CK->D
```

```
               View: analysis_setup_wc
              Group: clk
         Startpoint: (R) I_MULT/reg_op1_reg_2_/CK
              Clock: (R) clk
           Endpoint: (R) I_MULT/res_reg_22_/D
              Clock: (R) clk

                       Capture       Launch
        Clock Edge:+     4.500        0.000
       Src Latency:+     0.000        0.000
       Net Latency:+     0.000 (I)    0.000 (I)
           Arrival:=     4.500        0.000

             Setup:-     0.143
       Cppr Adjust:+     0.000
     Required Time:=     4.357
      Launch Clock:=     0.000
         Data Path:+     4.352
             Slack:=     0.005

#-------------------------------------------------------------------------------
---------------
# Timing Point               Flags  Arc    Edge  Cell              Fanout  Trans
Delay  Arrival
#                                                                            (ns)
(ns)     (ns)
#-------------------------------------------------------------------------------
---------------
  I_MULT/reg_op1_reg_2_/CK   C1C2  CK     R     (arrival)            503  0.000
-    0.000
  I_MULT/reg_op1_reg_2_/Q    C1C2  CK->Q  R     SEN_FDPRBQ_D_1P5       9  0.000
0.234    0.234
  I_MULT/U1607/X             C1C2  A2->X  F     SEN_EO2_S_0P5          3  0.162
0.150    0.384
  I_MULT/U681/X              C1C2  B->X   R     SEN_ND2B_V1DG_1        6  0.244
0.160    0.544
  I_MULT/FE_OFC217_n1369/X   C1C2  A->X   R     SEN_BUF_1             29  0.178
0.572    1.116
  I_MULT/U1600/X             C1C2  A2->X  F     SEN_OAI21_S_0P5        1  1.071
0.533    1.649
  I_MULT/U2025/X             C1C2  A2->X  F     SEN_EO2_S_0P5          2  0.485
0.250    1.899
  I_MULT/U2026/X             C1C2  A1->X  F     SEN_OR2_DG_1           2  0.230
0.212    2.110
  I_MULT/U989/X              C1C2  A2->X  R     SEN_AOI21_G_1          2  0.078
0.114    2.224
  I_MULT/U75/X               C1C2  A2->X  F     SEN_OAI21_G_1          2  0.130
0.104    2.328
  I_MULT/U2033/X             C1C2  A1->X  R     SEN_AOI21_G_1          2  0.118
0.122    2.449
  I_MULT/U2036/X             C1C2  A1->X  F     SEN_OAI21_G_1          2  0.137
0.111    2.560
  I_MULT/U2038/X             C1C2  A2->X  R     SEN_AOI21_T_1          2  0.115
0.121    2.681
  I_MULT/U2039/X             C1C2  A2->X  F     SEN_OAI21_G_2          2  0.140
0.106    2.787
  I_MULT/U2049/X             C1C2  A1->X  R     SEN_AOI21_T_2          2  0.111
0.095    2.882
  I_MULT/U689/X              C1C2  A1->X  F     SEN_OAI21_G_2          2  0.101
0.095    2.976
  I_MULT/U811/X              C1C2  A1->X  R     SEN_AOI21_3            2  0.112
0.111    3.087
  I_MULT/U2070/X             C1C2  A2->X  F     SEN_OAI21_S_3          2  0.135
0.121    3.208
  I_MULT/U114/X              C1C2  A->X   R     SEN_INV_0P8            3  0.119
0.105    3.313
  I_MULT/U103/X              C1C2  A1->X  F     SEN_OAI21_S_1          1  0.133
0.107    3.420
  I_MULT/U2298/X             C1C2  A2->X  R     SEN_EN2_0P5            1  0.109
0.577    3.997
  I_MULT/U77/X               C1C2  A1->X  R     SEN_AO22_DG_1          1  1.299
0.354    4.352
```

```
  I_MULT/res_reg_22_/D         C1C2   D        R      SEN_FDPRBQ_D_1          1   0.138
0.000     4.352
#-------------------------------------------------------------------------------
---------------
```

Save the design state as `DB/tutorial/top_32b_place` with the write_db command

Save the current state of the design in the snapshot stack and call it place

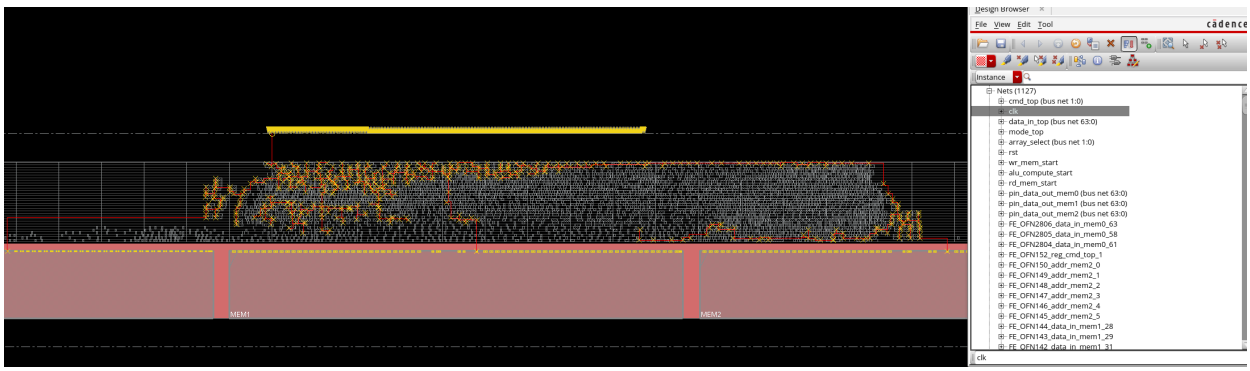
```
pop_snapshot_stack
create_snapshot -name place
```

4.13. SYNTHESIZING THE CLOCK TREE

As the paths that will propagate the clock signal in the design are not necessarily balanced, some registers may receive the active clock edge later than others (clock skew) and may therefore violate the assumed synchronous design operation. The clock tree synthesis (or optimization) aims at minimizing the clock skew by inserting clock buffers[15].

4.13.1. DISPLAY THE INITIAL CLOCK TREE

1. In the Innovus main menu, select **Tools > Design Browser...**.
2. In the **Design Browser** window, select the `clk` net in the **Net** part.
3. Deselect the visibility toggle **V** of the Net item.
4. After having observed the clock tree, exit the **Design Browser** window.
5. Select the visibility toggle **V** of the Net item.



4.13.2. SYNTHESIZE THE CLOCK TREE

As you can see in the previous screenshot, the clock signal is extremely long and does connect to all the flipflops and the clock input of the memory. Let's now replace this by a realistic clock tree which will make sure that hold violations do not occur. The Clock Tree Synthesis (CTS) can also be used to balance paths and reclaim a bit of time which can be lost in some parts of the circuit because of the routing.

[15] See also http://www.signoffsemi.com/cts-part-1-2

Start with a push in the metrics snapshots

```
innovus > push_snapshot_stack
```

First we define a few variables which will be used to optimize the routing of the clock tree. This may not always be mandatory, but as usual, you can always customize things if you need so.

```
innovus> set_db opt_useful_skew_ccopt standard
```

This enables the ccopt command to perform clock skew optimization and use it in a standard way to improve the timing. Putting it to extreme would enable better potential optimization gains at the cost of a higher runtime.

```
innovus> create_route_type -name leaf_rt -top_preferred_layer M4 -
bottom_preferred_layer M2
innovus> create_route_type -name trunk_rt -top_preferred_layer M5 -
bottom_preferred_layer M4
innovus> set_db cts_route_type_leaf leaf_rt
innovus> set_db cts_route_type_trunk trunk_rt
```

Here we customize the construction of the clock tree by separating the leaf and the trunk routing and matching each to different metal layers.

```
innovus> ccopt_design

...

------------------------------------------------------------------
     opt_design Final Summary
------------------------------------------------------------------

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.003  |  0.003  |  1.261  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  1408   |   650   |   762   |
+--------------------+---------+---------+---------+

+----------------+-----------------------------------+------------------+
|                |               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   |     12 (12)      |    -25     |     18 (18)      |
|   max_length   |      0 (0)       |     0      |      0 (0)       |
+----------------+------------------+------------+------------------+

Density: 24.961%
Routing Overflow: 0.12% H and 0.00% V
------------------------------------------------------------------

innovus > check_place
```

```
Begin checking placement ... (start mem=3512.7M, init mem=3512.7M)
*info: Placed = 4937           (Fixed = 1063)
*info: Unplaced = 0
Placement Density:24.96%(13460/53925)
Placement Density (including fixed std cells):26.00%(14219/54684)
Finished check_place (total: cpu=0:00:00.2, real=0:00:00.0; vio checks:
cpu=0:00:00.0, real=0:00:00.0; mem=3512.7M)
```

At this point you could run the following commands to check the status of your clock tree

```
report_clock_trees
report_skew_groups
```

QUESTION 4-12 : explain in a few lines what these report tell you

Let's now optimize the design post clock tree synthesis. As we now have a design that has a clock tree, we can optimize now for both setup and hold.

```
innovus > opt_design -post_cts -hold -setup


...
------------------------------------------------------------------------
             Initial Summary
------------------------------------------------------------------------

Setup views included:
 analysis_setup_wc

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.003  |  0.003  |  1.261  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  1408   |   650   |   762   |
+--------------------+---------+---------+---------+

+----------------+-----------------------------------+-----------------
-+
|                |              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   |     12 (12)     |    -25    |     18 (18)     |
|   max_length   |      0 (0)      |     0     |      0 (0)      |
+----------------+-----------------+-----------+-----------------
-+

Density: 24.961%
Routing Overflow: 0.12% H and 0.00% V
------------------------------------------------------------------------


...
------------------------------------------------------------------------
```

```
     Hold Opt Initial Summary
--------------------------------------------------------------------

Setup views included:
 analysis_setup_wc
Hold views included:
 analysis_hold_bc

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.007  |  0.007  |  1.255  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  1408   |   650   |   762   |
+--------------------+---------+---------+---------+

+--------------------+---------+---------+---------+
|     Hold mode      |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):| -0.034  |  0.103  | -0.034  |
|           TNS (ns):| -1.960  |  0.000  | -1.960  |
|    Violating Paths:|   71    |    0    |   71    |
|          All Paths:|  1408   |   650   |   762   |
+--------------------+---------+---------+---------+

+----------------+-----------------------------------+-----------------
-+
|                |              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   |     12 (12)      |    -25     |      18 (18)      |
|   max_length   |      0 (0)       |     0      |       0 (0)       |
+----------------+------------------+------------+-----------------
-+

Density: 24.957%
--------------------------------------------------------------------


...
--------------------------------------------------------------------
     opt_design Final Summary
--------------------------------------------------------------------

Setup views included:
 analysis_setup_wc
Hold views included:
 analysis_hold_bc

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
```

```
|           WNS (ns):|  0.007   |  0.007   |  1.254   |
|           TNS (ns):|  0.000   |  0.000   |  0.000   |
|    Violating Paths:|    0     |    0     |    0     |
|          All Paths:|  1408    |   650    |   762    |
+--------------------+----------+----------+----------+

+--------------------+----------+----------+----------+
|     Hold mode      |   all    | reg2reg  | default  |
+--------------------+----------+----------+----------+
|           WNS (ns):|  0.002   |  0.103   |  0.002   |
|           TNS (ns):|  0.000   |  0.000   |  0.000   |
|    Violating Paths:|    0     |    0     |    0     |
|          All Paths:|  1408    |   650    |   762    |
+--------------------+----------+----------+----------+

+----------------+-----------------------------------+-----------------
-+
|                |              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   |     12 (12)       |    -25        |     18 (18)      |
|   max_length   |      0 (0)        |     0         |      0 (0)       |
+----------------+-------------------+---------------+------------------
-+

Density: 25.147%
Routing Overflow: 0.14% H and 0.00% V
------------------------------------------------------------------------
**opt_design ... cpu = 0:00:27, real = 0:00:29, mem = 2863.6M,
totSessionCpu=7:14:13 **
*** Finished opt_design ***
      flow.cputime          flow.realtime          timing.setup.tns
timing.setup.wns  snapshot
UM:*                                        0.000 ns          0.007
ns  final
------------------------------------------------------------------
     Current design flip-flop statistics

Single-Bit FF Count  :          500
Multi-Bit FF Count   :            0
Total Bit Count      :          500
Total FF Count       :          500
Bits Per Flop        :        1.000
------------------------------------------------------------------
      flow.cputime          flow.realtime          timing.setup.tns
timing.setup.wns  snapshot
UM:                        119.09                                  637
opt_design_postcts
Info: Destroy the CCOpt slew target map.
*** opt_design #1 [finish] : cpu/real = 0:00:27.4/0:00:29.2 (0.9),
totSession cpu/real = 7:14:13.9/47:21:16.3 (0.2), mem = 3525.9M
```

You can note that the density increased slightly. Run a report_timing -late and a report_timing -early.

```
@innovus 423> report_timing -late
#####################################################################
#  Generated by:      Cadence Innovus 20.14-s095_1
#  OS:                Linux x86_64(Host ID edasrv1)
#  Generated on:      Fri Nov 10 11:33:04 2023
#  Design:            top_32b
#  Command:           report_timing -late
#####################################################################
Path 1: MET (0.007 ns) Setup Check with Pin I_MULT/res_reg_41_/CK->D
               View: analysis_setup_wc
              Group: clk
         Startpoint: (R) I_MULT/reg_op1_reg_28_/CK
              Clock: (R) clk
           Endpoint: (R) I_MULT/res_reg_41_/D
              Clock: (R) clk

                      Capture       Launch
        Clock Edge:+    4.500        0.000
       Src Latency:+   -0.129       -0.129
       Net Latency:+    0.128 (P)    0.129 (P)
           Arrival:=    4.499       -0.000

             Setup:-    0.081
       Cppr Adjust:+    0.000
     Required Time:=    4.418
      Launch Clock:=   -0.000
         Data Path:+    4.410
             Slack:=    0.007

#-------------------------------------------------------------------------
-------------------------
# Timing Point                Flags  Arc     Edge  Cell               Fanout
Trans   Delay  Arrival
#                                                                       (ns)
(ns)     (ns)
#-------------------------------------------------------------------------
-------------------------
  I_MULT/reg_op1_reg_28_/CK   C1C2  CK      R     (arrival)             89
0.222       -   -0.000
  I_MULT/reg_op1_reg_28_/Q    C1C2  CK->Q   R     SEN_FDPRBQ_D_1P5       2
0.222   0.366    0.366
  I_MULT/U1377/X              C1C2  A1->X   R     SEN_EN2_0P5            2
0.257   0.257    0.623
  I_MULT/U1530/X              C1C2  A2->X   R     SEN_AN3_1              7
0.362   0.291    0.914
  I_MULT/FE_OFC301_n79/X      C1C2  A->X    R     SEN_BUF_D_3           26
0.239   0.236    1.150
  I_MULT/U1016/X              C1C2  C1->X   F     SEN_AOI222_0P5         1
0.241   0.186    1.336
  I_MULT/U39/X                C1C2  B->X    R     SEN_OAI21_G_1          1
0.238   0.091    1.427
  I_MULT/U1687/X              C1C2  A2->X   R     SEN_EO2_S_0P5          1
0.162   0.157    1.584
  I_MULT/U1756/S              C1C2  CI->S   F     SEN_ADDF_0P5           1
0.367   0.355    1.939
  I_MULT/U1745/CO             C1C2  CI->CO  F     SEN_ADDF_0P5           1
0.159   0.265    2.204
  I_MULT/U2145/CO             C1C2  B->CO   F     SEN_ADDF_0P5           1
0.139   0.277    2.482
  I_MULT/U2158/S              C1C2  CI->S   R     SEN_ADDF_0P5           1
0.134   0.325    2.806
```

```
  I_MULT/U2156/S               C1C2  B->S    F     SEN_ADDF_0P5           1
0.138   0.279    3.085
  I_MULT/U2165/S               C1C2  CI->S   R     SEN_ADDF_0P5           2
0.123   0.365    3.450
  I_MULT/U196/X                C1C2  A1->X   F     SEN_NR2_G_1            2
0.236   0.092    3.543
  I_MULT/U2186/X               C1C2  A1->X   R     SEN_NR2_G_1            2
0.108   0.120    3.663
  I_MULT/U2187/X               C1C2  A1->X   F     SEN_ND2_1              3
0.179   0.136    3.799
  I_MULT/U128/X                C1C2  A->X    R     SEN_INV_0P8            4
0.176   0.125    3.924
  I_MULT/U1486/X               C1C2  A1->X   F     SEN_ND2_S_0P5          1
0.155   0.101    4.024
  I_MULT/U2226/X               C1C2  A2->X   R     SEN_OAI21_G_1          1
0.111   0.146    4.171
  I_MULT/U2227/X               C1C2  A2->X   R     SEN_EN2_S_1            1
0.199   0.115    4.286
  I_MULT/U2228/X               C1C2  A1->X   R     SEN_AO22_1             1
0.134   0.124    4.410
  I_MULT/res_reg_41_/D         C1C2  D       R     SEN_FDPRBQ_D_1         1
0.109   0.000    4.410
#-------------------------------------------------------------------------
-------------------------

@innovus 424> report_timing -early
#####################################################################
#  Generated by:      Cadence Innovus 20.14-s095_1
#  OS:                Linux x86_64(Host ID edasrv1)
#  Generated on:      Fri Nov 10 11:33:25 2023
#  Design:            top_32b
#  Command:           report_timing -early
#####################################################################
#####################################################################################
########
# Design Stage: PreRoute
# Design Name: top_32b
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
#####################################################################################
########
Start delay calculation (fullDC) (1 T). (MEM=3532.44)
Total number of fetched objects 4532
AAE_INFO: Total number of nets for which stage creation was skipped for all
views 0
End delay calculation. (MEM=3575.06 CPU=0:00:00.8 REAL=0:00:01.0)
End delay calculation (fullDC). (MEM=3575.06 CPU=0:00:01.0 REAL=0:00:01.0)
Path 1: MET (0.002 ns) Hold Check with Pin reg_rd_mem_start_reg/CK->D
               View: analysis_hold_bc
              Group: clk
         Startpoint: (F) rd_mem_start
              Clock:
           Endpoint: (F) reg_rd_mem_start_reg/D
              Clock: (R) clk

                     Capture       Launch
        Clock Edge:+   0.000        0.000
       Src Latency:+  -0.063        0.000
       Net Latency:+   0.068 (P)    0.000 (I)
           Arrival:=   0.005        0.000

             Hold:+    0.026
       Cppr Adjust:-   0.000
```

```
      Required Time:=    0.031
       Launch Clock:=    0.000
          Data Path:+    0.032
              Slack:=    0.002

#-------------------------------------------------------------------------
---------------------------
# Timing Point             Flags  Arc           Edge  Cell            Fanout
Trans   Delay  Arrival
#                                                                        (ns)
(ns)     (ns)
#-------------------------------------------------------------------------
---------------------------
  rd_mem_start             -      rd_mem_start  F     (arrival)           1
0.004   0.000    0.000
  FE_PHC376_rd_mem_start/X   C1C2  A->X          F     SEN_BUF_1           1
0.004   0.032    0.032
  reg_rd_mem_start_reg/D     C1C2  D             F     SEN_FDPRBQ_D_1      1
0.017   0.000    0.032
#-------------------------------------------------------------------------
---------------------------
```

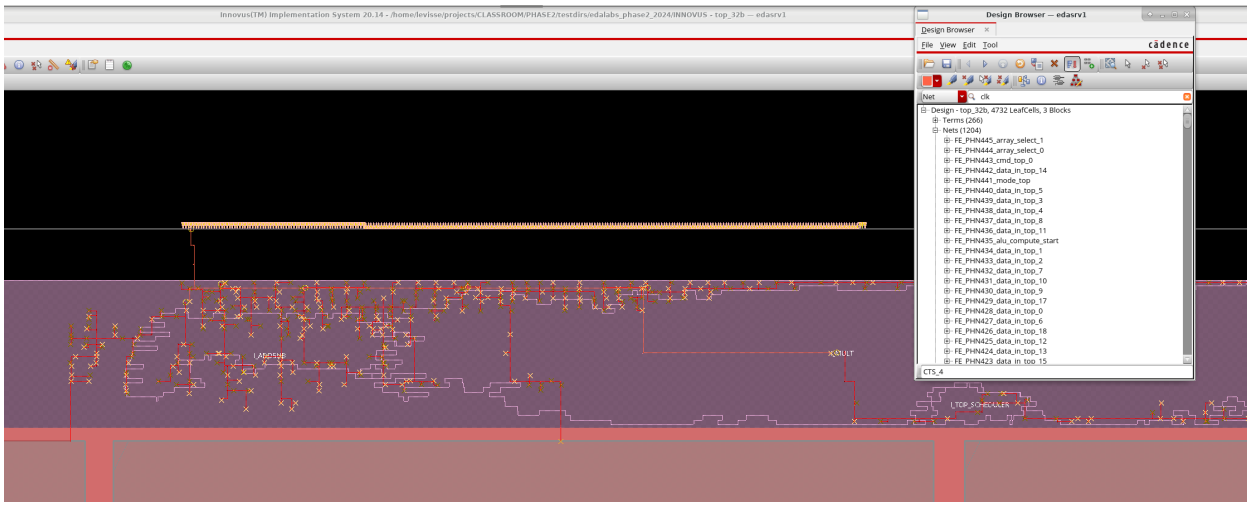
4.13.3. DISPLAY THE OPTIMIZED CLOCK TREE

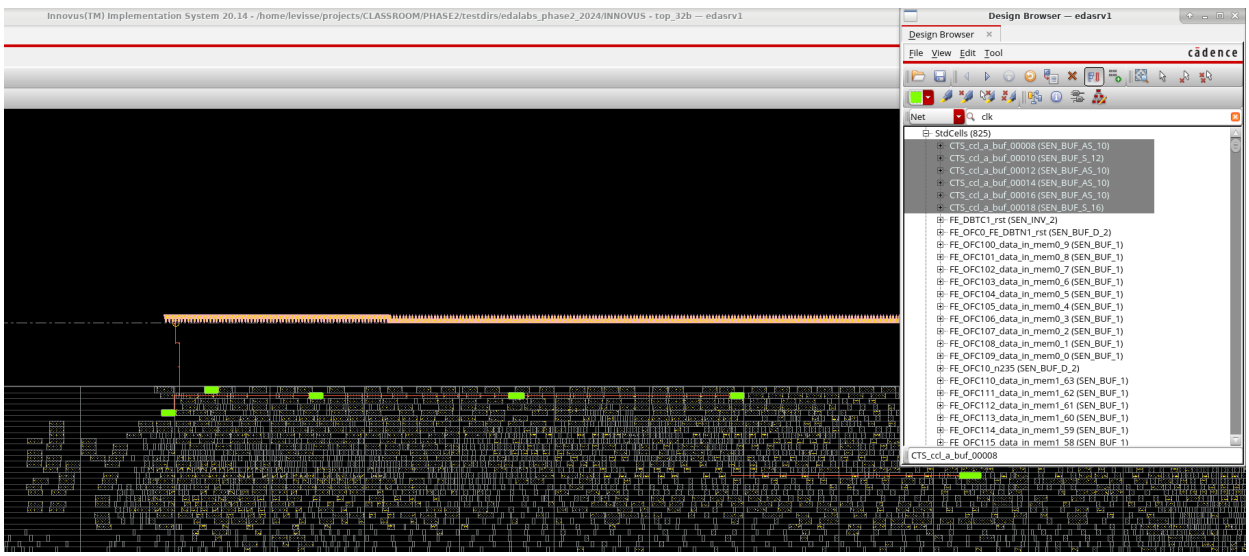
1. To see the clock nets post CTS:
   - In the Innovus main menu, select **Tools > Design Browser...**.
   - In the nets subsection, search for “clk” and “CTS_..”
   - Select them (CTRL+Shift) and highlight the nets with a specific color.

Your design may be different from the screenshot.

2. To see the buffers post CTS :
   - From the design browser, in stdcells
   - Search for CTS_... select them and highlight them with a different color

To see the cells, you must switch to  and disable the metals, to see something as shown in the screenshot.





Save the design state as `DB/tutorial/top_32b_cts` with the write_db command

Save the current state of the design in the snapshot stack and call it cts

```
pop_snapshot_stack
create_snapshot -name cts
```

4.14. ROUTING THE DESIGN

This steps performs global and detailed routing.

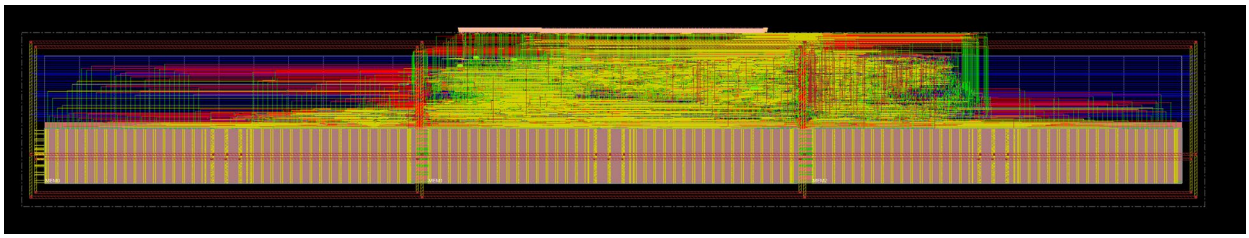
In the *global routing* phase, the router breaks the routing portion of the design into rectangles called *global routing cells* (gcells) and assigns the signal nets to the gcells. The global router attempts to find the shortest path through the gcells, but does not make actual connections or assign nets to specific tracks within the gcells. It tries to avoid assigning more nets to a gcell than the tracks can accommodate.

In the *detailed routing* phase, the router follows the global routing plan and lays down actual wires that connect the pins to their corresponding nets. The router runs search-and-repair routing: it locates shorts and spacing violations and reroutes the affected areas to eliminate as many of the violations as possible. The primary goal of detailed routing is to complete all of the required interconnect without leaving shorts or spacing violations.

4.14.1. TO ROUTE THE DESIGN

Here we use the route_design command which takes as input the parameters defined at the beginning of the design. i.e., the routing will only be done between M2 and M5.

```
innovus > push_snapshot_stack
innovus > route_design
```



You can note at this point that the placement is much more organized than before.

QUESTION 4-13: Zoom in the design above MEM0, what do you see in M2 ? what kind of issues do you believe you will face there when (i) increasing the frequency, (ii) reducing the available place.

QUESTION 4-14: run a report_timing -early and a report_timing -late. Comment the results.

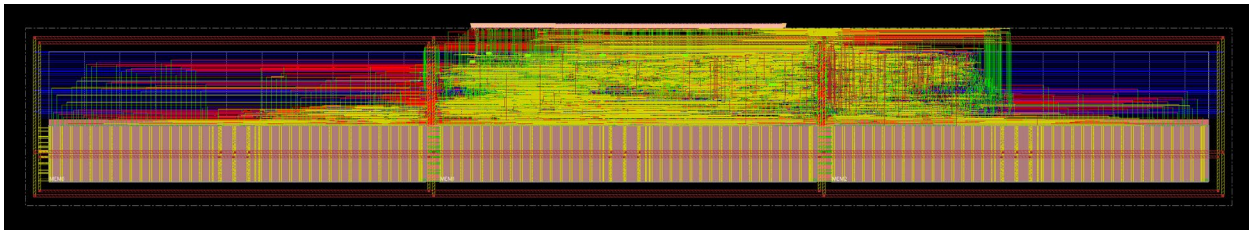
4.14.2. DO A POST-ROUTE TIMING OPTIMIZATION

Post-route timing optimization is better done using text commands:

```
> opt_design -post_route -hold -setup
```

QUESTION 4-15: run a report_timing -early and a report_timing -late. Why is it different compared to before ?

The design display area now shows the routed design.



At this point, it could be that your design still does not pass the setup check. This is because the floorplan we propose here is highly suboptimal. Running the opt_design command again could solve the issue, as it gives the tool another chance to converge to a better optimum.

Still, remember that generally, when a flow requires an arbitrary amount of operations to converge, it probably means that you are close to the failing point, and most likely need to relax some constraint somewhere.

Save the design state as `DB/tutorial/top_32b_routed` with the write_db command

Save the current state of the design in the snapshot stack and call it cts

```
pop_snapshot_stack
create_snapshot -name routed
```

4.15. ADDING FILLER CELLS

Now that the design is finalized, let's insert filler cells to fill the remaining holes in the rows and ensure the continuity of power/ground rails and N+/P+ wells. But also will ensure that Front end of line (active, diffusions, polysilicon, contacts etc.) density DRC constraints are met in the placed area.

The list of filler cells can be found in the technology documentation of the standard cells.

Page 17 onward of the documentation presents the special cells utilization. While page 41 onwards shows the list of special cells. For Filler cells, the site option range from 1 to 64 (1, 2, 4, 8, 16, 32, 64).

```
/dkits/synopsys/DesignWare_logic_libs/commonplatform65nlp/hd/base/svt
/latest/doc/lped0186_cp65npkslogcasdst000f.pdf
```

```
> push_snapshot_stack
> add_fillers -base_cells SEN_FILL1 SEN_FILL2 SEN_FILL4
SEN_FILL8 SEN_FILL16 SEN_FILL32 SEN_FILL64 -prefix FILLER
**WARN: (IMPSP-5217): add_fillers command is running on a postRoute
database. It is recommended to be followed by eco_route -target
command to make the DRC clean.
Type 'man IMPSP-5217' for more detail.
*INFO: Adding fillers to top-module.
*INFO:   Added 954 filler insts (cell SEN_FILL64 / prefix FILLER).
*INFO:   Added 165 filler insts (cell SEN_FILL32 / prefix FILLER).
*INFO:   Added 1015 filler insts (cell SEN_FILL16 / prefix FILLER).
*INFO:   Added 1208 filler insts (cell SEN_FILL8 / prefix FILLER).
*INFO:   Added 2185 filler insts (cell SEN_FILL4 / prefix FILLER).
*INFO:   Added 2079 filler insts (cell SEN_FILL2 / prefix FILLER).
*INFO:   Added 2244 filler insts (cell SEN_FILL1 / prefix FILLER).
*INFO: Swapped 0 special filler inst.
*INFO: Total 9850 filler insts added - prefix FILLER (CPU:
0:00:01.4).
For 9850 new insts, 9850 new pwr-pin connections were made to global
net 'VDD'.
9850 new pwr-pin connections were made to global net 'VDD'.
9850 new gnd-pin connections were made to global net 'VSS'.
9850 new gnd-pin connections were made to global net 'VSS'.
*** Applied 4 GNC rules (cpu = 0:00:00.0)
*INFO: Filler mode add_fillers_with_drc is default true to avoid
gaps, which may add fillers with violations. Please check the
violations for FILLER_incr* fillers and fix them before
routeDesign. Set it to false can avoid the violation but may leave
gaps.
*INFO: Second pass addFiller without DRC checking.
*INFO: Adding fillers to top-module.
*INFO:      Added 12 filler insts (cell SEN_FILL64 / prefix
FILLER_incr).
*INFO:      Added 27 filler insts (cell SEN_FILL32 / prefix
FILLER_incr).
*INFO:      Added 42 filler insts (cell SEN_FILL16 / prefix
FILLER_incr).
*INFO:      Added 49 filler insts (cell SEN_FILL8 / prefix
FILLER_incr).
*INFO:      Added 60 filler insts (cell SEN_FILL4 / prefix
FILLER_incr).
*INFO:      Added 750 filler insts (cell SEN_FILL2 / prefix
FILLER_incr).
*INFO:      Added 61 filler insts (cell SEN_FILL1 / prefix
FILLER_incr).
*INFO: Swapped 0 special filler inst.
*INFO: Total 1001 filler insts added - prefix FILLER_incr (CPU:
0:00:00.0).
For 1001 new insts, 1001 new pwr-pin connections were made to global
net 'VDD'.
1001 new pwr-pin connections were made to global net 'VDD'.
1001 new gnd-pin connections were made to global net 'VSS'.
1001 new gnd-pin connections were made to global net 'VSS'.
*** Applied 4 GNC rules (cpu = 0:00:00.0)
Pre-route DRC Violation:   1001
```

If when running a `>check_drc` you still get drc violations you can do the following :

```
innovus > check_filler
*INFO: Total number of padded cell violations: 0
*INFO: Total number of gaps found: 0
```

To make sure there are no gaps and issues with the filling

Then, you can run `>route_eco -fix_drc` which will solve the drc errors by moving wires around.

Though it could induce timing violations which you can check with `>report_timing -late` and `>report_timing_early`

Finally, running an `>opt_design -post_route -hold -setup` would finally solve the issue.

Alternatively, you could probably insert the fillers before the first opt_design -post_route

Save the design state as `DB/tutorial/top_32b_filled` with the write_db command

Save the current state of the design in the snapshot stack and call it cts

```
pop_snapshot_stack
create_snapshot -name filled
```

4.16. VERIFY THE DESIGN

4.16.1. CONNECTIVITY CHECK

The *connectivity verification* detects problems such as opens, unconnected wires, unconnected pins, loops, partial routing, and unrouted nets:

```
innovus > check_connectivity
VERIFY_CONNECTIVITY use new engine.

******** Start: VERIFY CONNECTIVITY ********
Start Time: Fri Nov 10 13:52:56 2023

Design Name: top_32b
Database Units: 2000
Design Boundary: (0.0000, 0.0000) (1020.0000, 150.0000)
Error Limit = 1000; Warning Limit = 50
Check all nets

Begin Summary
  Found no problems or warnings.
End Summary

End Time: Fri Nov 10 13:52:56 2023
Time Elapsed: 0:00:00.0

******** End: VERIFY CONNECTIVITY ********
  Verification Complete : 0 Viols.  0 Wrngs.
  (CPU Time: 0:00:00.4  MEM: -0.375M)
```

4.16.2. GEOMETRY VERIFICATION (DRC)

As in phase 1, DRC verification check the spacing, and the internal geometry of objects and the wiring between them. this drc check is not based on the DRC sign-off rules but on the tech lef. This cannot replace an actual DRC check, but ensures that the routing has been done correctly.

```
innovus > check_drc
 *** Starting Verify DRC (MEM: 2551.4) ***

  VERIFY DRC ..... Starting Verification
  VERIFY DRC ..... Initializing
  VERIFY DRC ..... Deleting Existing Violations
  VERIFY DRC ..... Creating Sub-Areas
  VERIFY DRC ..... Using new threading
  VERIFY DRC ..... Sub-Area: {0.000 0.000 86.400 76.320} 1 of 24
  VERIFY DRC ..... Sub-Area : 1 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {86.400 0.000 172.800 76.320} 2 of 24
  VERIFY DRC ..... Sub-Area : 2 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {172.800 0.000 259.200 76.320} 3 of 24
  VERIFY DRC ..... Sub-Area : 3 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {259.200 0.000 345.600 76.320} 4 of 24
  VERIFY DRC ..... Sub-Area : 4 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {345.600 0.000 432.000 76.320} 5 of 24
  VERIFY DRC ..... Sub-Area : 5 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {432.000 0.000 518.400 76.320} 6 of 24
  VERIFY DRC ..... Sub-Area : 6 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {518.400 0.000 604.800 76.320} 7 of 24
  VERIFY DRC ..... Sub-Area : 7 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {604.800 0.000 691.200 76.320} 8 of 24
  VERIFY DRC ..... Sub-Area : 8 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {691.200 0.000 777.600 76.320} 9 of 24
  VERIFY DRC ..... Sub-Area : 9 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {777.600 0.000 864.000 76.320} 10 of 24
  VERIFY DRC ..... Sub-Area : 10 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {864.000 0.000 950.400 76.320} 11 of 24
  VERIFY DRC ..... Sub-Area : 11 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {950.400 0.000 1020.000 76.320} 12 of 24
  VERIFY DRC ..... Sub-Area : 12 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {0.000 76.320 86.400 150.000} 13 of 24
  VERIFY DRC ..... Sub-Area : 13 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {86.400 76.320 172.800 150.000} 14 of 24
  VERIFY DRC ..... Sub-Area : 14 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {172.800 76.320 259.200 150.000} 15 of 24
  VERIFY DRC ..... Sub-Area : 15 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {259.200 76.320 345.600 150.000} 16 of 24
  VERIFY DRC ..... Sub-Area : 16 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {345.600 76.320 432.000 150.000} 17 of 24
  VERIFY DRC ..... Sub-Area : 17 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {432.000 76.320 518.400 150.000} 18 of 24
  VERIFY DRC ..... Sub-Area : 18 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {518.400 76.320 604.800 150.000} 19 of 24
  VERIFY DRC ..... Sub-Area : 19 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {604.800 76.320 691.200 150.000} 20 of 24
  VERIFY DRC ..... Sub-Area : 20 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {691.200 76.320 777.600 150.000} 21 of 24
  VERIFY DRC ..... Sub-Area : 21 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {777.600 76.320 864.000 150.000} 22 of 24
```

```
  VERIFY DRC ..... Sub-Area : 22 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {864.000 76.320 950.400 150.000} 23 of 24
  VERIFY DRC ..... Sub-Area : 23 complete 0 Viols.
  VERIFY DRC ..... Sub-Area: {950.400 76.320 1020.000 150.000} 24 of 24
  VERIFY DRC ..... Sub-Area : 24 complete 0 Viols.

  Verification Complete : 0 Viols.

 *** End Verify DRC (CPU: 0:00:01.3  ELAPSED TIME: 1.00  MEM: 0.0M) ***

1
```

4.16.3. PERFORM A CONGESTION ANALYSIS

One important step when routing a design with dense interconnect is identify the routing hotspots. This will give you important information on the bottlenecks that may exist in a design, and allow you to properly adapt your floorplan strategy.

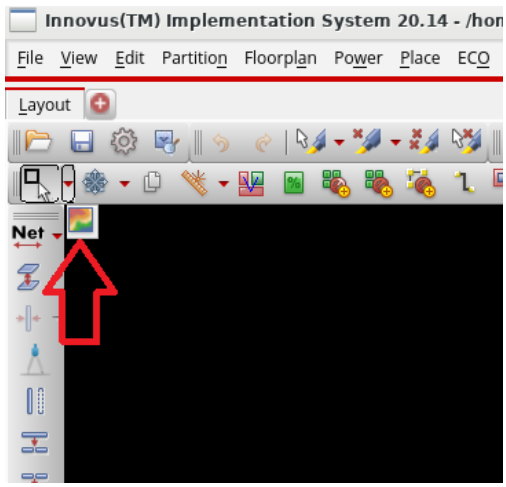
```
innovus > report_congestion -3d -hotspot
the report here may be different from yours
[hotspot] +------------+---------------+---------------+-------------------------------------+
[hotspot] |   layer    |  max hotspot  | total hotspot |            hotspot bbox             |
[hotspot] +------------+---------------+---------------+-------------------------------------+
[hotspot] |   M1(H)    |          4.00 |         13.33 |   468.00   108.00   482.40   122.40 |
[hotspot] |   M2(H)    |          0.00 |          0.00 |   (none)                            |
[hotspot] |   M3(V)    |          0.00 |          0.00 |   (none)                            |
[hotspot] |   M4(H)    |          3.11 |          4.44 |   331.20    79.20   345.60    93.60 |
[hotspot] |   M5(V)    |          0.00 |          0.00 |   (none)                            |
[hotspot] +------------+---------------+---------------+-------------------------------------+
[hotspot] |   worst    | (M1)     4.00 | (M1)    13.33 |                                     |
[hotspot] +------------+---------------+---------------+-------------------------------------+
[hotspot] | all layers |          0.00 |          0.00 |                                     |
[hotspot] +------------+---------------+---------------+-------------------------------------+
Local HotSpot Analysis (3d): normalized congestion max/total hotspot area = 0.00/0.00 (area is in unit of 4
std-cell row bins)
```

QUESTION 4-16 : explain with your own words what does the report_congestion reports. describe the difference between “max hotspot” and “total hotspot” from the TCR documentation page 2545 onwards.

However, while this command gives you the place where the main congestion hotspots happen (bbox), it is not really easy to read and take actions from this report.

Click on Route>NanoRoute>analyze Congestion

This will open a small colored square on the top left of the innovus window, click on it

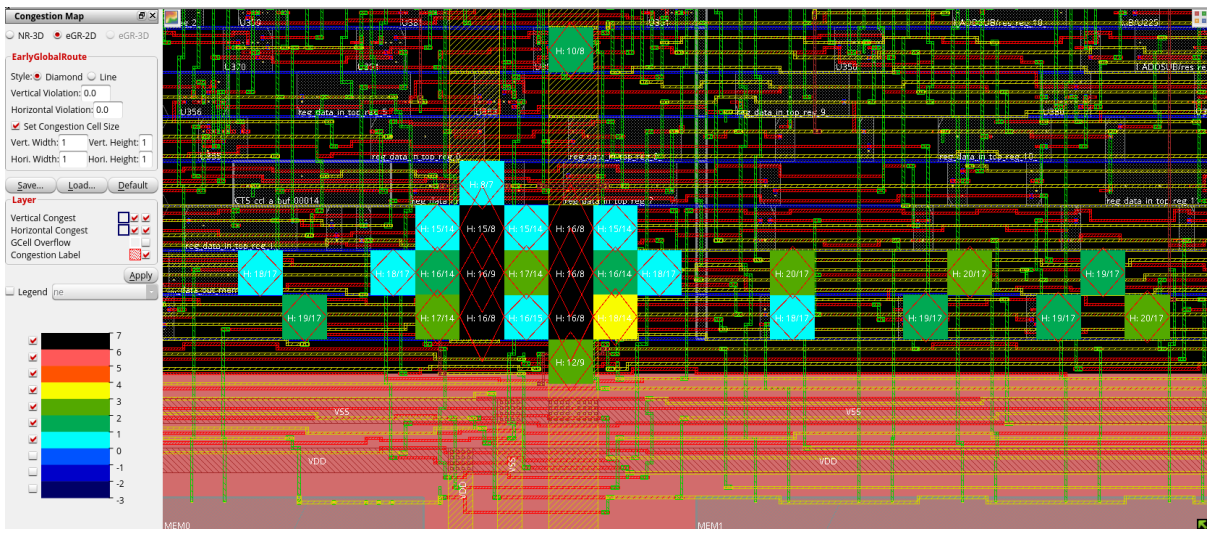


Select the following options in the congestion analysis panel :

- eGR-2D
- Diamond
- Vertical and horizontal violations to 0
- Tick Set congestion set size
- Put Vert. and Horiz. width and length to 1

Right click on the white color on the scale for 7, and put it black → this will make the result readable as white text on white squares is unreadable.

Click apply, and zoom in your design to identify congestion areas.



The data visible in the diamonds shows the following “H: X/Y”

- H means horizontal (respectively V means vertical). The congestion is on the horizontal layer
- X stands for the number of required metal tracks (a track means minimum width + minimum spacing for a metal layer) to route your design
- Y stands for the number of available metal tracks.

When X is higher than Y in this configuration, the tool will highlight the cell.

Here is a list of things a designer could do with the congestion analyzer :

- Flag negative congestion, to identify zones that are under risks of getting congested.
- Consider larger or smaller congestion area to check for congestion risks at various scales.

Of course, at this point, considering 4.5ns clock period, your design is not too congested. However, this tool will get extremely handy when experimenting a new floorplan, or check how things go when increasing the clock frequency.

QUESTION 4-17 : at this point, where does the congestion happen in your design ? what could explain it ? you could add screenshots if useful for your explanation.

4.17. GENERATING REPORTS

Let's now generate and save reports. It is recommended to group reports related to the same constraints in a specific RPT subdirectory. In our case, it is the `RPT/tutorial/` directory.

You can generally save them with the -out_file option or by using a ">" operator

4.17.1. DESIGN SUMMARY

A report on the entire design includes statistics for the following categories:

- General design information
- General library information
- Netlist information
- Timing information
- Floorplan/Placement information

```
innovus > report_summary -out_file RPT/tutorial/summary
Start to collect the design information.
Build netlist information for Cell top_32b.
Finished collecting the design information.
Generating macro cells used in the design report.
Generating standard cells used in the design report.
Analyze library ...
Analyze netlist ...
Analyze timing ...
Analyze floorplan/placement ...
Analysis Routing ...
Report saved in file RPT/tutorial/summary
```

The display of the HTML version of the report may take same time to happen.

4.17.2. DESIGN AREA

Use the `report_area` command to report on the design area (output edited):

```
innovus > report_area -verbose
Hinst Name          Module Name          Inst Count          Total Area              Buffer              Inverter
Combinational                Flop               Latch           Clock Gate               Macro            Physical
```

```
------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------
top_32b                                      3956           59193.330             651.960             426.960
8414.280            4157.640              0.000               0.000           45542.490               0.000
 I_ADDSUB           addsub_32b               364            1393.560               2.880              41.040
546.480             803.160               0.000               0.000               0.000               0.000
 I_MULT             mult_32b                2686            8918.280             276.840             249.840
7307.280            1084.320              0.000               0.000               0.000               0.000
 I_TOP_SCHEDULER    top_scheduler              76             201.600               1.440              12.960
104.400              82.800               0.000               0.000               0.000               0.000

innovus > report_area -verbose > RPT/tutorial/area
```

4.17.3. CRITICAL PATH TIMING

Use the `report_timing` command to report on the timing of the most critical path (output edited):

```
innovus> report_timing -late -max_paths 10
...
Path 1: MET (0.001 ns) Setup Check with Pin I_MULT/res_reg_47_/CK->D
               View: analysis_setup_wc
              Group: clk
         Startpoint: (R) I_MULT/reg_op2_reg_15_/CK
              Clock: (R) clk
           Endpoint: (R) I_MULT/res_reg_47_/D
              Clock: (R) clk

                       Capture       Launch
        Clock Edge:+     4.500        0.000
       Src Latency:+    -0.129       -0.129
       Net Latency:+     0.121 (P)    0.128 (P)
           Arrival:=     4.492       -0.001

             Setup:-     0.083
       Cppr Adjust:+     0.000
     Required Time:=     4.409
      Launch Clock:=    -0.001
         Data Path:+     4.409
             Slack:=     0.001

#-------------------------------------------------------------------------------------------------
# Timing Point               Flags  Arc     Edge  Cell            Fanout  Trans   Delay  Arrival
#                                                                          (ns)    (ns)      (ns)
#-------------------------------------------------------------------------------------------------
  I_MULT/reg_op2_reg_15_/CK   C1C2  CK      R     (arrival)           89  0.216       -   -0.001
  I_MULT/reg_op2_reg_15_/Q    C1C2  CK->Q   R     SEN_FDPRBQ_D_4      37  0.216   0.465    0.464
  I_MULT/U425/X               C1C2  A2->X   F     SEN_NR2_G_2          2  0.419   0.110    0.573
  I_MULT/U688/X               C1C2  A2->X   R     SEN_NR2_G_1          2  0.131   0.118    0.691
  I_MULT/U390/X               C1C2  A2->X   F     SEN_ND2_1            2  0.148   0.094    0.785
  I_MULT/U372/X               C1C2  A2->X   R     SEN_NR2_G_1          1  0.109   0.103    0.888
  I_MULT/U699/X               C1C2  A2->X   F     SEN_AOI21_G_2        1  0.131   0.086    0.974
...
  I_MULT/U173/X               C1C2  A1->X   F     SEN_ND2_1            3  0.132   0.104    3.565
  I_MULT/U150/X               C1C2  A2->X   R     SEN_OAI21_G_1        3  0.140   0.155    3.720
  I_MULT/U484/X               C1C2  A1->X   F     SEN_AOI21_G_1        1  0.206   0.116    3.835
  I_MULT/U1548/X              C1C2  B->X    F     SEN_OA21_2           1  0.128   0.125    3.960
  I_MULT/U2331/X              C1C2  B->X    F     SEN_OA21_8          20  0.070   0.113    4.073
  I_MULT/U2369/X              C1C2  A1->X   R     SEN_OAI21_G_1        1  0.087   0.101    4.174
  I_MULT/U2370/X              C1C2  A2->X   R     SEN_EN2_S_1          1  0.144   0.107    4.280
  I_MULT/U479/X               C1C2  A1->X   R     SEN_AO22_1           1  0.155   0.127    4.408
  I_MULT/res_reg_47_/D        C1C2  D       R     SEN_FDPRBQ_D_1       1  0.110   0.000    4.408
#-------------------------------------------------------------------------------------------------
...
Path 10: MET (0.013 ns) Setup Check with Pin I_MULT/res_reg_45_/CK->D
               View: analysis_setup_wc
              Group: clk
         Startpoint: (R) I_MULT/reg_op2_reg_15_/CK
              Clock: (R) clk
           Endpoint: (R) I_MULT/res_reg_45_/D
              Clock: (R) clk

                       Capture       Launch
        Clock Edge:+     4.500        0.000
       Src Latency:+    -0.129       -0.129
       Net Latency:+     0.119 (P)    0.128 (P)
           Arrival:=     4.490       -0.001

             Setup:-     0.083
       Cppr Adjust:+     0.000
```

```
      Required Time:=    4.407
       Launch Clock:=   -0.001
          Data Path:+    4.395
              Slack:=    0.013

#-------------------------------------------------------------------------------------------------
# Timing Point               Flags  Arc     Edge  Cell            Fanout  Trans   Delay  Arrival
#                                                                          (ns)    (ns)     (ns)
#-------------------------------------------------------------------------------------------------
  I_MULT/reg_op2_reg_15_/CK   C1C2  CK      R     (arrival)           89  0.216       -   -0.001
  I_MULT/reg_op2_reg_15_/Q    C1C2  CK->Q   R     SEN_FDPRBQ_D_4      37  0.216   0.465    0.464
  I_MULT/U425/X               C1C2  A2->X   F     SEN_NR2_G_2          2  0.419   0.110    0.573
  I_MULT/U688/X               C1C2  A2->X   R     SEN_NR2_G_1          2  0.131   0.118    0.691
  I_MULT/U390/X               C1C2  A2->X   F     SEN_ND2_1            2  0.148   0.094    0.785
  I_MULT/U372/X               C1C2  A2->X   R     SEN_NR2_G_1          1  0.109   0.103    0.888
  I_MULT/U699/X               C1C2  A2->X   F     SEN_AOI21_G_2        1  0.131   0.086    0.974
...
  I_MULT/U2331/X              C1C2  B->X    F     SEN_OA21_8          20  0.070   0.112    4.072
  I_MULT/U2344/X              C1C2  A1->X   R     SEN_OAI21_G_1        1  0.086   0.098    4.170
  I_MULT/U2345/X              C1C2  A2->X   R     SEN_EN2_S_1          1  0.140   0.100    4.269
  I_MULT/U461/X               C1C2  A1->X   R     SEN_AO22_1           1  0.144   0.124    4.394
  I_MULT/res_reg_45_/D        C1C2  D       R     SEN_FDPRBQ_D_1       1  0.111   0.000    4.394
#-------------------------------------------------------------------------------------------------
innovus> report_timing -late -max_paths 10 > RPT/tutorial/timing_setup
```

4.17.4. HOLD CHECK

Same as for the setup, you can save the 10 fastest paths

```
innovus> report_timing -early -max_paths 10 > RPT/tutorial/timing_hold
```

4.17.5. NETLIST STATISTICS

```
innovus > report_netlist_statistics
*** Statistics for net list top_32b ***
Number of cells      = 15932
Number of nets       = 4534
Number of tri-nets   = 0
Number of degen nets = 0
Number of pins       = 15330
Number of i/os       = 266

Number of nets with    2 terms = 3242 (71.5%)
Number of nets with    3 terms = 802 (17.7%)
Number of nets with    4 terms = 214 (4.7%)
Number of nets with    5 terms = 44 (1.0%)
Number of nets with    6 terms = 16 (0.4%)
Number of nets with    7 terms = 17 (0.4%)
Number of nets with    8 terms = 12 (0.3%)
Number of nets with    9 terms = 10 (0.2%)
Number of nets with >=10 terms = 177 (3.9%)

*** 164 Primitives used:
Primitive sramHD_64x64 (3 insts)
Primitive SEN_TIE0_1 (1 insts)
Primitive SEN_OR2_DG_1 (32 insts)
Primitive SEN_OR2_2 (1 insts)
Primitive SEN_OR2_1 (12 insts)
Primitive SEN_OAOI211_0P5 (2 insts)
```

```
Primitive SEN_OAI31_G_0P5 (1 insts)
Primitive SEN_OAI22_0P5 (1 insts)
Primitive SEN_OAI21B_3 (1 insts)
Primitive SEN_OAI21B_2 (4 insts)
Primitive SEN_OAI21B_1 (1 insts)
Primitive SEN_OAI21_G_2 (8 insts)
Primitive SEN_OAI21_G_1 (122 insts)
Primitive SEN_OAI21_T_3 (1 insts)
...
innovus > report_netlist_statistics > RPT/tutorials/netlist_stats
```

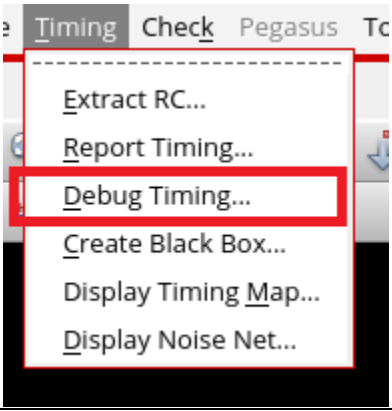
4.17.6. VISUALLY CHECK THE TIMING

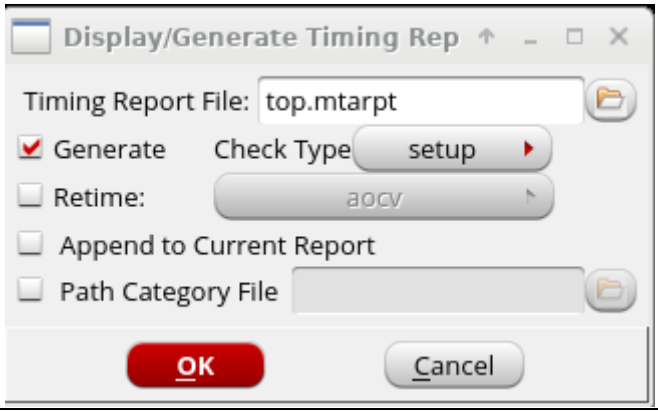
As in design compiler, you can generate histograms. This allows you to track signals and identify why something is wrong in your timing analysis

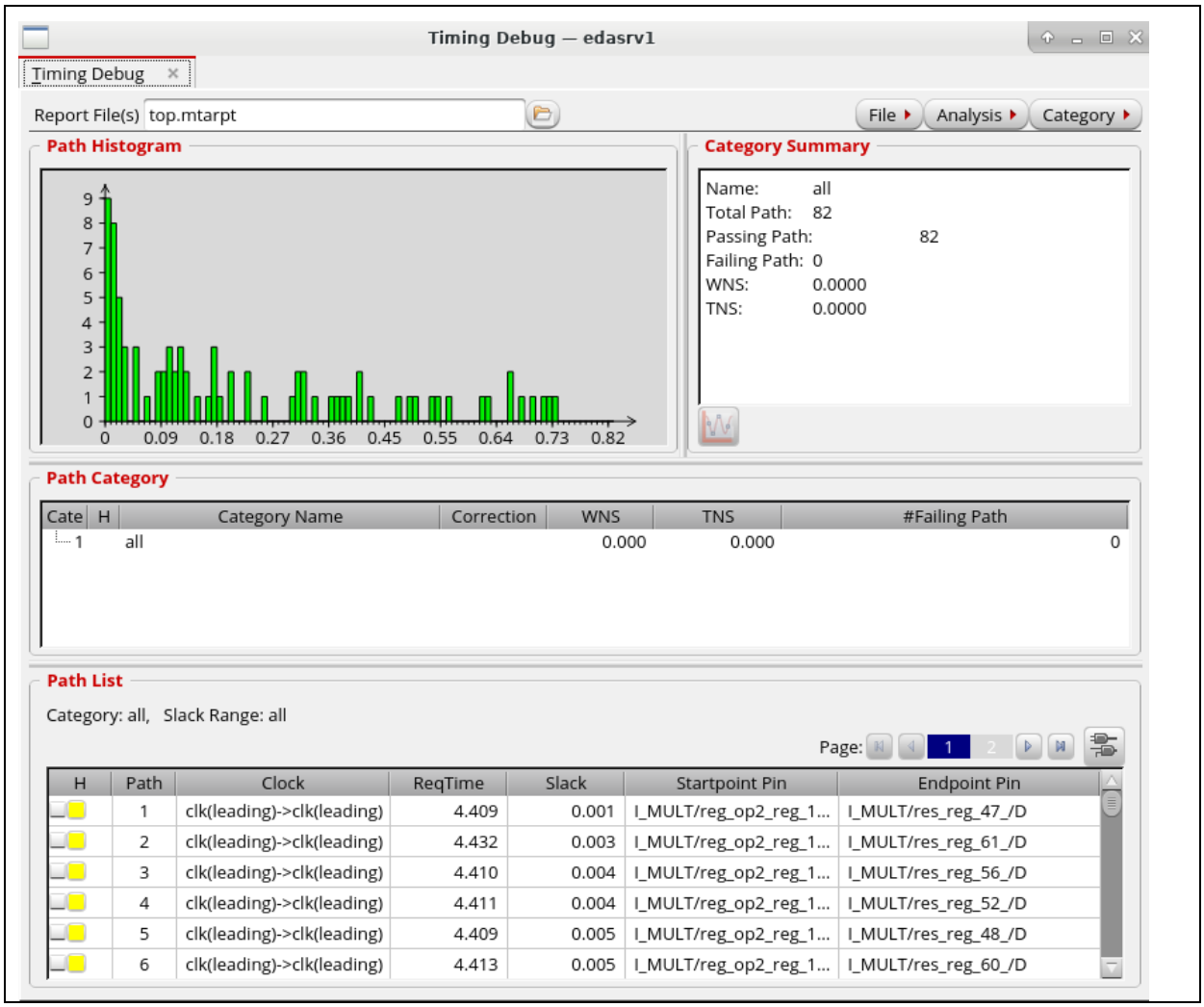
In the main innovus window, **Timing>Debug Timing**

Then make sure it is on "**setup**" and press **OK**.

From there you can analyse the timing in your circuit.







4.17.7. GENERATE THE SNAPSHOTS OUTPUTS AND ANALYZE THEM

run the following commands

```
>report_metric -format vivid -file RPT/tutorial/top_32b_snapshots.html
```

Open the top_32b_snapshots.html file and explore the results. It allows you to track visually many reports and identify bottlenecks in your design.

QUESTION 4-18 : report the CPU runtime of your design for the different steps. Which part took the most time ? does the computation time make sense ? which parts would you expect take the most time in a more complex design ?

4.18. EXPORTING THE DESIGN

This steps generates all the data that is required for post-place+route gate-level simulation and the design import in Virtuoso.

4.18.1. GENERATE THE VERILOG NETLIST

At the end of the place and route flow, you want to generate a new verilog netlist, which you will use for LVS verification, but also for logic simulation.

We create two netlists, one with all the cells and pg pins (VDD and VSS) in the standard cells, which will be used for LVS. And one without fillers and without pg pins, used for logic simulation.

Removing the fillers for simulation is not mandatory, but could sometimes simplify your life.

```
innovus> write_netlist -include_pg_ports
../HDL/PLACED/tutorial/top_32b_placed_lvs.v

Writing Netlist "../HDL/PLACED/tutorial/top_32b_placed_lvs.v" ...
Pwr name (VDD).
Gnd name (VSS).
1 Pwr names and 1 Gnd names.


innovus > write_netlist -exclude_leaf_cells -
exclude_insts_of_cells {SEN_FILL1 SEN_FILL2 SEN_FILL4
SEN_FILL8 SEN_FILL16 SEN_FILL32 SEN_FILL64}
../HDL/PLACED/tutorial/top_32b_placed_sim.v


Writing Netlist "../HDL/PLACED/tutorial/top_32b_placed_sim.v" ...
```

4.18.2. GENERATING THE SDF TIMING FILE

As for synthesis, we now want to generate sdf file that will be used during post PnR simulation.

```
@innovus 65> set_db timing_enable_simultaneous_setup_hold_mode true
1 true
@innovus 66> set_db timing_recompute_sdf_in_setuphold_mode true
1 true
@innovus 69> write_sdf TIM/tutorial/top_32b_placed.sdf -precision 3
-min_period_edges posedge -recompute_parallel_arcs -
adjust_setup_hold_for_zero_hold_slack -version 2.1 -min_view
analysis_hold_bc -max_view analysis_setup_wc


**WARN: (TCLCMD-1465): The write_sdf option -
recompute_parallel_arcs has been deprecated and replaced by the -
recompute_delay_calc option. It will continue to function in this
release, but you should update your scripts to use the new option.
Refer to the command reference for additional information on the new
option.
```

```
Starting SI iteration 1 using Infinite Timing Windows
#############################################################################
##############
# Design Stage: PostRoute
# Design Name: top_32b
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: SPEF/RCDB
# Signoff Settings: SI On
#############################################################################
##############
AAE_INFO: 1 threads acquired from CTE.
Start delay calculation (fullDC) (1 T). (MEM=2555.4)
Total number of fetched objects 4534
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
AAE_INFO-618: Total number of nets in the design is 4536,  100.0
percent of the nets selected for SI analysis
Total number of fetched objects 4534
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
AAE_INFO-618: Total number of nets in the design is 4536,  100.0
percent of the nets selected for SI analysis
End delay calculation. (MEM=2567.2 CPU=0:00:07.0 REAL=0:00:07.0)
End   delay   calculation   (fullDC).   (MEM=2567.2   CPU=0:00:07.2
REAL=0:00:07.0)
Loading CTE timing window with TwFlowType 0...(CPU = 0:00:00.0, REAL
= 0:00:00.0, MEM = 2567.2M)
Add other clocks and setupCteToAAEClockMapping during iter 1
Loading CTE timing window is completed (CPU = 0:00:00.0, REAL =
0:00:00.0, MEM = 2567.2M)
Starting SI iteration 2
Start delay calculation (fullDC) (1 T). (MEM=2538.41)
Glitch Analysis: View analysis_setup_wc -- Total Number of Nets
Skipped = 0.
Glitch Analysis: View analysis_setup_wc -- Total Number of Nets
Analyzed = 0.
Total number of fetched objects 4534
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
AAE_INFO-618: Total number of nets in the design is 4536,  17.2
percent of the nets selected for SI analysis
Glitch Analysis: View analysis_hold_bc -- Total Number of Nets
Skipped = 380.
Glitch Analysis: View analysis_hold_bc -- Total Number of Nets
Analyzed = 4534.
Total number of fetched objects 4534
AAE_INFO: Total number of nets for which stage creation was skipped
for all views 0
AAE_INFO-618: Total number of nets in the design is 4536,  4.5 percent
of the nets selected for SI analysis
End delay calculation. (MEM=2586.62 CPU=0:00:01.3 REAL=0:00:02.0)
End   delay   calculation   (fullDC).   (MEM=2586.62   CPU=0:00:01.4
REAL=0:00:02.0)
```

The generated SDF file is different from the one generated after synthesis. It now includes the proper interconnect wiring delays.

```
...
  (CELL
    (CELLTYPE  "top_32b")
    (INSTANCE)
      (DELAY
        (ABSOLUTE
        (INTERCONNECT FE_PHC447_wr_mem_start/X FE_PHC375_wr_mem_start/A
(0.000::0.000) (0.000::0.000))
        (INTERCONNECT FE_PHC446_rd_mem_start/X FE_PHC376_rd_mem_start/A
(0.000::0.000) (0.000::0.000))
        (INTERCONNECT cmd_top[1] reg_cmd_top_reg_1_/D  (0.000::0.001)
(0.000::0.001))
        (INTERCONNECT cmd_top[0] FE_PHC443_cmd_top_0/A  (-0.000::0.001) (-
0.000::0.001))
        (INTERCONNECT clk CTS_ccl_a_buf_00018/A  (0.010::0.008)
(0.010::0.008))
        (INTERCONNECT clk CTS_ccl_a_buf_00016/A  (0.003::0.003)
(0.003::0.003))
        (INTERCONNECT clk CTS_ccl_a_buf_00014/A  (0.008::0.007)
(0.008::0.007))
        (INTERCONNECT clk CTS_ccl_a_buf_00012/A  (0.002::0.002)
(0.002::0.002))
        (INTERCONNECT clk CTS_ccl_a_buf_00010/A  (0.005::0.005)
(0.005::0.005))
        (INTERCONNECT clk CTS_ccl_a_buf_00008/A  (0.001::0.002)
(0.001::0.002))
        (INTERCONNECT   data_in_top[63]   FE_PHC377_data_in_top_63/A      (-
0.000::0.000) (-0.000::0.000))

...

  (CELL
    (CELLTYPE  "SEN_FDPRBQ_D_1")
    (INSTANCE  I_MULT/res_reg_11_)
      (DELAY
        (ABSOLUTE
        (IOPATH (posedge CK) Q  (0.102::0.281) (0.098::0.265))
        (COND CK (IOPATH RD Q  () (0.152::0.454)))
        (COND (~(CK)&D) (IOPATH RD Q  () (0.153::0.458)))
        (COND (~(CK)&~(D)) (IOPATH RD Q  () (0.153::0.456)))
        )
      )
      (TIMINGCHECK
        (SETUPHOLD (posedge D) (COND RD (posedge CK)) (0.032::0.086) (-
0.007::-0.006))
        (SETUPHOLD (negedge D) (COND RD (posedge CK)) (0.010::0.017)
(0.025::0.080))
        (WIDTH (COND cond2 (posedge CK)) (0.140::0.310))
        (WIDTH (COND cond2 (negedge CK)) (0.122::0.301))
        (WIDTH (COND cond3 (posedge CK)) (0.140::0.310))
        (WIDTH (COND cond3 (negedge CK)) (0.122::0.301))
```

- Note that all the three-value sets `min:typ:max` have an empty `typ` field. It will then be important to select the proper delay type in simulation.

4.18.3. GENERATING THE GDS2 FILE

The GDS2 (or GDSII) format is a binary file format commonly used for representing and exchanging geometrical layout data.

The gds is extracted with the following inputs :

- The map file which contains a list of layers and maps each layer used in innovus to its actual layer from the technology PDK. If the map file is not defined properly, the foundry will not be able to use the GDS. A good mapping of the layers can be checked by importing the layout inside a virtuoso library attached to the technology and check that the layers match. The map file is not always provided by the foundry. Though generally IP providers give it with the lef files.
- The unit which should generally match with the smallest unit provided by the IPs (generally in the lef). Here 2000.
- The gds files to be merged. Here the memory macro gds from the compiler and the gds of the standard cells.

```
innovus > write_stream -map_file streamout.map -unit 2000
STREAM/tutorial/top_32b_placed.gds -merge
{../IPS/MEMORIES/sramHD_64x64/sramHD_64x64.gds
../IPS/STDCELLS/hd/base/svt/latest/gds/cp65npksdst2well.gds}

Merge file: ../IPS/MEMORIES/sramHD_64x64/sramHD_64x64.gds has version
number: 600
Merge file: ../IPS/STDCELLS/hd/base/svt/latest/gds/cp65npksdst2well.gds has
version number: 600
Parse flat map file...
**WARN: (EMS-27):      Message (IMPOGDS-387) has exceeded the current
message display limit of 20.
To increase the message display limit, refer to the product command
reference manual.
** NOTE: Created directory path 'STREAM/tutorial' for file
'STREAM/tutorial/top_32b_placed.gds'.
Writing GDSII file ...
        ****** db unit per micron = 2000 ******
        ****** output gds2 file unit per micron = 2000 ******
        ****** unit scaling factor = 1 ******
Output for instance
Output for bump
Output for physical terminals
Output for logical terminals
Output for regular nets
Output for special nets and metal fills
Output for via structure generation total number 46
Statistics for GDS generated (version 600)
----------------------------------------
Stream Out Layer Mapping Information:
GDS Layer Number          GDS Layer Name
----------------------------------------
    62                               COMP
    69                                 LB
```

```
    111                                 EA
    70                                  VV
    31                                  M5
    174                                 NT
    21                                  M4
    18                                  V2
    15                                  M1
    17                                  M2
    22                                  V4
    19                                  M3
    20                                  V3
    16                                  V1


Stream Out Information Processed for GDS version 600:
Units: 2000 DBU

Object                             Count
----------------------------------------
Instances                          15932

Ports/Pins                           266
    metal layer M2                   266

Nets                               63801
    metal layer M2                 33260
    metal layer M3                 21056
    metal layer M4                  8979
    metal layer M5                   506

    Via Instances                  33839

Special Nets                         194
    metal layer M1                    96
    metal layer M3                    63
    metal layer M4                    29
    metal layer M5                     6

    Via Instances                    714

Metal Fills                            0

    Via Instances                      0

Metal FillOPCs                         0

    Via Instances                      0

Metal FillDRCs                         0

    Via Instances                      0

Text                                 268
    metal layer M2                   266
    metal layer M5                     2


Blockages                              0
```

```
Custom Text                              0


Custom Box                               0

Trim Metal                               0

Scanning GDS file ../IPS/MEMORIES/sramHD_64x64/sramHD_64x64.gds to register
cell name ......
Scanning GDS file
../IPS/STDCELLS/hd/base/svt/latest/gds/cp65npksdst2well.gds to register
cell name ......
Merging GDS file ../IPS/MEMORIES/sramHD_64x64/sramHD_64x64.gds ......
        ****** Merge file: ../IPS/MEMORIES/sramHD_64x64/sramHD_64x64.gds has
version number: 600.
        ****** Merge file: ../IPS/MEMORIES/sramHD_64x64/sramHD_64x64.gds has
units: 1000 per micron.
        ****** unit scaling factor = 2 ******
Merging GDS file
../IPS/STDCELLS/hd/base/svt/latest/gds/cp65npksdst2well.gds ......
        ****** Merge file:
../IPS/STDCELLS/hd/base/svt/latest/gds/cp65npksdst2well.gds has version
number: 600.
        ****** Merge file:
../IPS/STDCELLS/hd/base/svt/latest/gds/cp65npksdst2well.gds has units: 1000
per micron.
        ****** unit scaling factor = 2 ******

######Streamout is finished!
```

4.18.4. GENERATING THE SPEF FILE

The SPEF[16] file is used to store the metal parasitic resistance and capacitance values of all the wires. It will be needed to perform a precise power analysis on primetime.

First, we generate the parasitics on innovus

```
innovus > extract_rc
#Start Inst Signature in MT(0)
#Start Net Signature in MT(21210294)
#Calculate SNet Signature in MT (55174126)
#Run time and memory report for RC extraction:
#RC extraction running on  Xeon 3.00GHz 36608KB Cache 96CPU.
#Run Statistics for snet signature:
#    Cpu time = 00:00:00, elapsed time = 00:00:00 .
#    Increased memory =     0.00 (MB), total memory =  1687.09 (MB),
peak memory =  1860.69 (MB)
#Run Statistics for Net Final Signature:
#    Cpu time = 00:00:00, elapsed time = 00:00:00 .
#    Increased memory =     0.00 (MB), total memory =  1687.09 (MB),
peak memory =  1860.69 (MB)
#Run Statistics for Net launch:
#    Cpu time = 00:00:00, elapsed time = 00:00:00 .
#    Increased memory =     0.00 (MB), total memory =  1687.09 (MB),
peak memory =  1860.69 (MB)
```

[16] https://en.wikipedia.org/wiki/Standard_Parasitic_Exchange_Format

```
#Run Statistics for Net init_dbsNet_slist:
#   Cpu time = 00:00:00, elapsed time = 00:00:00 .
#   Increased memory =     0.00 (MB), total memory =  1687.09 (MB),
peak memory =  1860.69 (MB)
#Run Statistics for net signature:
#   Cpu time = 00:00:00, elapsed time = 00:00:00 .
#   Increased memory =     0.00 (MB), total memory =  1687.09 (MB),
peak memory =  1860.69 (MB)
#Run Statistics for inst signature:
#   Cpu time = 00:00:00, elapsed time = 00:00:00 .
#   Increased memory =   -68.85 (MB), total memory =  1687.09 (MB),
peak memory =  1860.69 (MB)
The design is extracted. Skipping TQuantus.
```

Then we write the spef files for the rcbest and rcworst corners

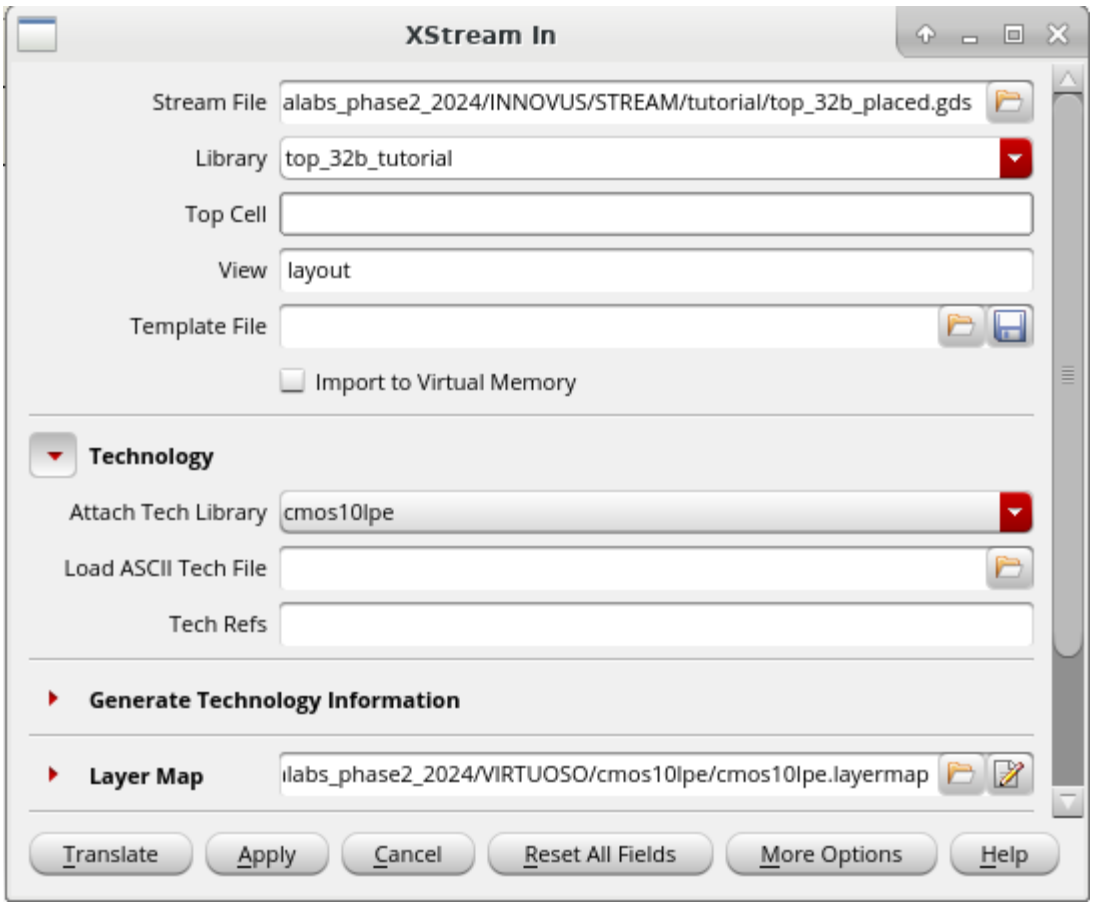
```
innovus>                write_parasitics                -spef_file
SPEF/tutorial/top_32b_rcworst.spef -rc_corner corner_rcworst

Reading RCDB with compressed RC data.
RC Out has the following PVT Info:
   RC:corner_rcworst
Dumping Spef file.....
Printing D_NET...
rcOut completed:: 100 %
RC Out from RCDB Completed (CPU Time= 0:00:00.4  MEM= 2538.6M)

innovus>                write_parasitics                -spef_file
SPEF/tutorial/top_32b_rcbest.spef -rc_corner corner_rcbest

Reading RCDB with compressed RC data.
RC Out has the following PVT Info:
   RC:corner_rcbest
Dumping Spef file.....
Printing D_NET...
rcOut completed:: 100 %
RC Out from RCDB Completed (CPU Time= 0:00:00.3  MEM= 2542.6M)
```

5. RUNNING A DRC FROM VIRTUOSO

5.1. IMPORTING THE DESIGN ON VIRTUOSO

1. Go in the VIRTUOSO folder
2. Source the sourceme.csh file >source sourceme.csh
3. Start virtuoso with >virtuoso &
4. In the **Virtuoso** window, select **File > Import > Stream...**.
5. In the **XStream In** window:
   - Define the **Stream File** field as `INNOVUS/STREAM/tutorial/top_32b_placed.gds`.
   - Define the **Library** field as `top_32b_tutorial`.
   - Define the **View** field as `layout`.
   - In the **Technology** section, define the **Attach Tech Library** as `cmos10lpe`.
6. Click the **More Options** button.

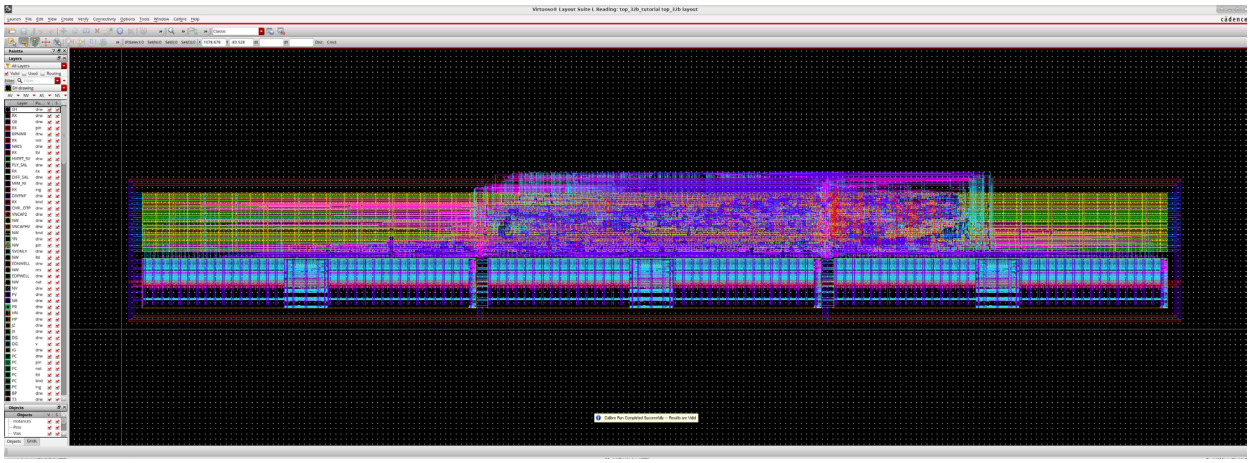
7. In the **XStream In** window, click **Translate** or **Apply**.



- The tool will issue some warning. You should have around 40-50 warnings or something about layers in the map file.

Search for the top_32b_tutorial library. Open the layout as read-only (right click on the layout > open as read-only) from the newly created top_32b cellview.

The full imported layout is then as follows. You can use the shift+F/ctrl+F keys to shwow/hide the details of the IPs (memories and standard cells).



5.2. RUNNING A DRC CHECK

From the layout viewer, select calibre>run nmDRC.

> Note that here we use the latest version of calibre. i.e., new GUI.

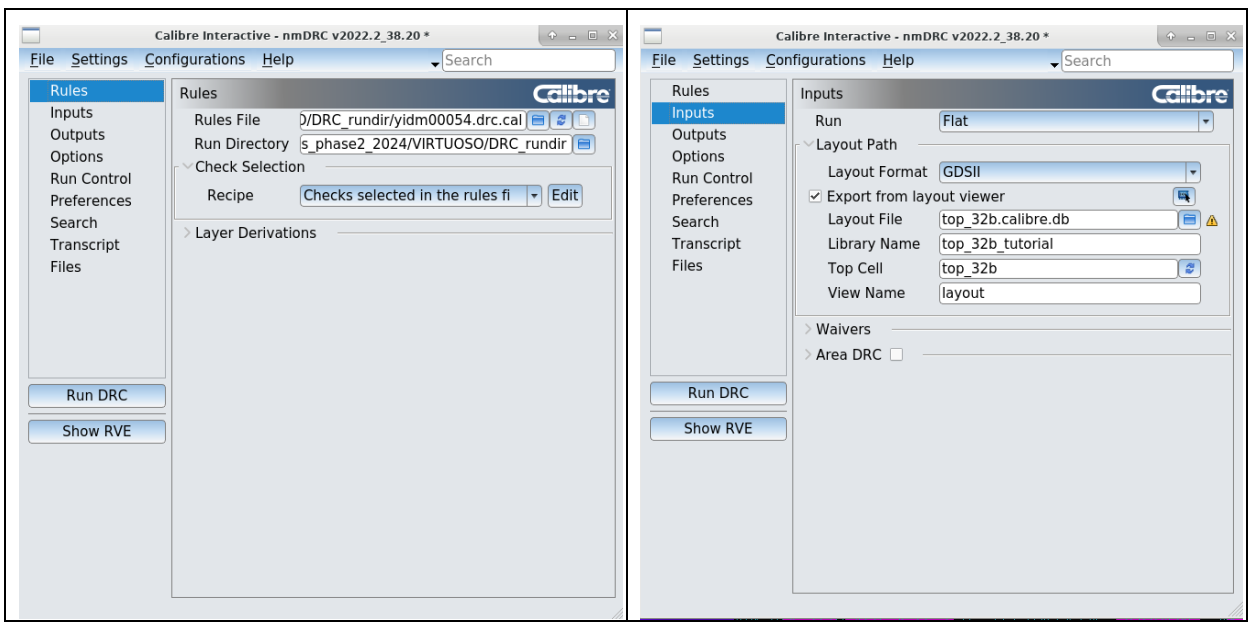
In the Rules panel :

- First, set the Run Directory to the DRC_rundir inside the VIRTUOSO folder.
- Then, set the Rules Files to tge yidm00054.drc.cal file inside the DRC_rundir folder.

In the input panel,

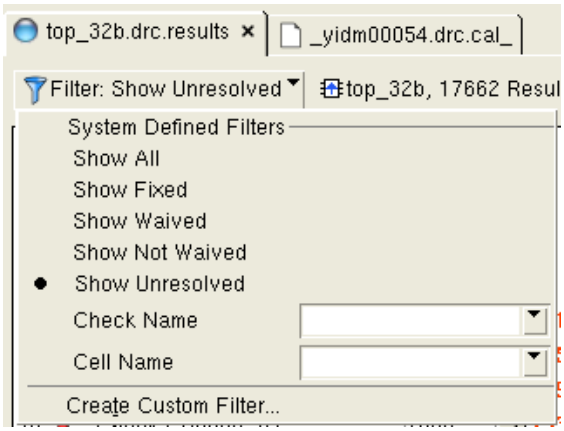
- select Flat in front of Run.
- Keep recipe as “check selected in rule files”

Click on Run DRC



If you get errors about environment variables, you may have forgotten to source the sourceme.csh file. In that case, quit virtuoso, source the file and start again.

In the results viewer, to see errors, switch the filter to “show Unresolved”



You should still get some errors, though you can ignore/wave the GRA999a and GRCHIPEDGE errors.

6. Place and Route the mapped design using TCL scripts inside Innovus

6.1. Using TCL scripts

As for the RTL synthesis, when the design complexity increases, or for speeding up the design exploration, it becomes much more convenient to use scripts and to run the place+route tool from the Linux command line. Scripts also conveniently capture the place+route flow and make it reusable. Cadence Innovus supports the Tcl language for building scripts.

In the BIN folder, you will find a folder called tutorial. This folder contains the following :

```
BIN/
└── tutorial
    ├── cts.tcl
    ├── default_mmmc.view
    ├── fillers.tcl
    ├── floorplan.tcl
    ├── fullflow.tcl
    ├── init.tcl
    ├── io.tcl
    ├── place.tcl
    ├── power.tcl
    ├── reports.tcl
    ├── route.tcl
    └── top32_mmmc_svt_mem64_4p5ns.view
```

Open the fullflow.tcl script with a text editor

```
#this file can be used to run the whole innovus flow at once
#update the SCRIPT_NAME variable to where all the steps of your flow
are
#by default a SCRIPT_NAME tutorial is used
```

```
#make a copy of this folder and change the SCRIPT_NAME variable
accordingly


set SCRIPT_NAME tutorial
set PATH_TO_SCRIPT BIN/${SCRIPT_NAME}

#these variables are used to create the proper folders in the results
hierarchy
set systemTime [clock seconds]
set time [clock format $systemTime -format %Hh%Mm%Ss]
set date [clock format $systemTime -format %D]
set date [string map {"/" ""} $date]

#creating the corresponding results folders
mkdir -p HDL/PLACED/${SCRIPT_NAME}/${date}/${time}
mkdir -p RPT/${SCRIPT_NAME}/${date}/${time}
mkdir -p TIM/${SCRIPT_NAME}/${date}/${time}
mkdir -p STREAM/${SCRIPT_NAME}/${date}/${time}
mkdir -p SPEF/${SCRIPT_NAME}/${date}/${time}
mkdir -p DB/${SCRIPT_NAME}/${date}/${time}

source ${PATH_TO_SCRIPT}/init.tcl
source ${PATH_TO_SCRIPT}/floorplan.tcl
source ${PATH_TO_SCRIPT}/power.tcl
source ${PATH_TO_SCRIPT}/io.tcl
source ${PATH_TO_SCRIPT}/place.tcl
source ${PATH_TO_SCRIPT}/cts.tcl
source ${PATH_TO_SCRIPT}/route.tcl
source ${PATH_TO_SCRIPT}/fillers.tcl
source ${PATH_TO_SCRIPT}/reports_exports.tcl
```

6.2. SETTING UP A NEW SET OF SCRIPTS FOR A NEW DESIGN

With this proposed flow, in order to make a test with a new floorplan or a new configuration, you should proceed as follows :

1. Copy the whole tutorial folder into a folder with a name that matches the new configuration
    - In the BIN folder >cp -r tutorial my_test_name (replace my_test_name with your naming convention)
2. In the fullflow.tcl, update the variable SCRIPT_NAME with my_test_name as you defined it before
3. Update the init.tcl and .view files according to the constraints and configurations you want to explore
4. Make sure the io file is pointing toward an io file you matched with your floorplan
5. From a new innovus instance, source the fullflow.tcl script
    - All the subfolders with the good names and places will be created accordingly

You could comment any line in the fullflow by adding a “#”. For e.g., to run the flow until the io placement, you could do :

```
source ${PATH_TO_SCRIPT}/init.tcl
source ${PATH_TO_SCRIPT}/floorplan.tcl
source ${PATH_TO_SCRIPT}/power.tcl
```

```
#source ${PATH_TO_SCRIPT}/io.tcl
#source ${PATH_TO_SCRIPT}/place.tcl
#source ${PATH_TO_SCRIPT}/cts.tcl
#source ${PATH_TO_SCRIPT}/route.tcl
#source ${PATH_TO_SCRIPT}/fillers.tcl
#source ${PATH_TO_SCRIPT}/reports_exports.tcl
```

6.3. AUTOMATICALLY SAVING THE DATABASE

Each sub-file of the script does contain lines such as

```
write_db DB/${SCRIPT_NAME}/${date}/${time}/top_32b_CTS
```

At its end. Thereby, if you have a look at the DB folder, you will find a folder organization for each run of you scripts.

QUESTION 6-1 : run the >du -sh DB command from the INNOVUS folder, and report the size of the folder.

This should highlight the need for you to clean this folder once in a while, and not always save everything you do.

7. RUN POST PNR SIMULATION

As for post synthesis simulation, you could at this point run a post place and route simulation which allows you to check the functionality of your circuit considering the parasitics.

1. Take the same configuration as for post synthesis simulation.
2. Take as a reference the environment you did setup in section 6 of the first document. With the following changes :
   - Create a new project which you could call TOP32_PLACED_4.5_SVT
   - Use the Verilog file from post PnR.
   - Use the sdf file from post PnR
3. Create a VCD file which you save in the ACTIVITY folder and call it activity_PLACED_4.5ns_SVT.vcd
   - A reference document for Post-PNR power estimation will soon be available.

8. DESIGN SPACE EXPLORATION

8.1. FIND THE MAXIMUM FREQUENCY FOR THE PROPOSED FLOORPLAN

Follow the methodology from section 6.2 on how to make new scripts for INNOVUS. Make a copy of the tutorial folder in the BIN folder of INNOVUS, and call it top_32b_XXns. Where XX is the maximum frequency you achieved during synthesis (it should be around 2 or 2.5ns).

In the new folder

1. update the mmmc file
   a. change its name
   b. in the file, update the path to the sdc file (line 55)
2. update accordingly the init.tcl :
   a. the path to the post synthesis Verilog file
   b. the name of the new mmmc file
3. update the fullflow.tcl

a. update the script_name variable at the beginning of the file

to explore the design space you have 2 options :

1. Reuse the results from several synthesis runs and for each, use the corresponding sdc and verilog
2. Use the fastest design, and edit the sdc to make the clock period slower and slower (slower means more ns).
    a. The second approach may seem simpler, though your designs will always be larger than with the first approach as by definition more optimized.

QUESTION 8-1 : without changing the floorplan, can you achieve the same operating frequency as during synthesis ? what fails first ? density ? timing ? metal routing congestion?

Hint : start from the max frequency you could achieve during synthesis. And then progressively reduce it until PnR can pass.

QUESTION 8-2 : what solutions could you imagine to make the design work when it fails? Propose a list of ideas and guidelines you would follow to make it work.

At this point, check the size of the DB folder (>du -sh DB) and clean it if needed

8.2. OPTIMIZE THE FLOORPLAN

The proposed floorplan is obviously bad. Let's now explore how one could improve it.

QUESTION 8-3 : propose a new floorplan for the top_32b architecture. Make a drawing before proceeding and report it here. Update accordingly the scripts and show, in the report how you handle the power supply (stripes and/or rings) – share screenshots.

Hint : Start with a relaxed frequency at first (4.5ns for e.g. as in the tutorial). And then, see if this floorplan makes your design go faster

QUESTION 8-4 : can you achieve a better frequency with this floorplan ? is it denser ? make sure your designs are clear from the perspective of the check_connectivity and check_drc. No need to push it down to virtuoso for the verification.

At this point, check the size of the DB folder (>du -sh DB) and clean it if needed

8.3. TAKE A SMALLER MEMORY

In this part, let's explore the consequencies of using a smaller memory on the performances, area -and power of the whole circuit. **Read both questions 7-5 and 7-6 before starting.**

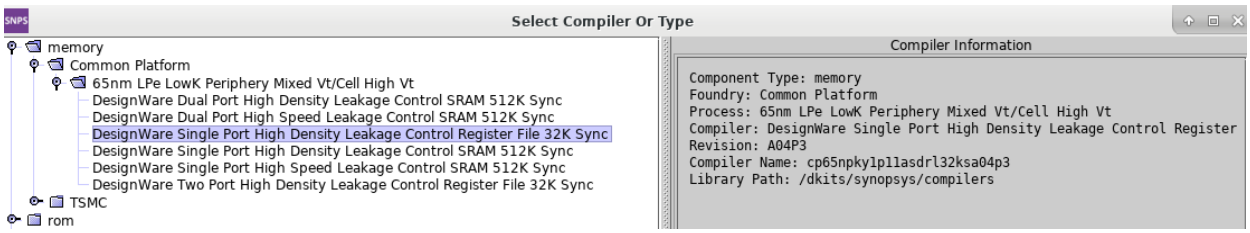
Close innovus, open Embedit (cf. section 2.2) and let's generate some memories. To restore the state of the memories created before, on embedit, click on project>open and select EDALABS (name of the project you created before).

QUESTION 8-5: make the same but with a 16x64 sramHD memory (this one should be already generated). How does the area, timing and power changes post synthesis? And then post-layout ? can you come up with a different floorplan ?

Hints :

1. make sure to properly update the vhdl code (alu_pkg and the top) – cf section 3
2. make sure you properly generate the db file from the lib file using library compiler – cf section 3.2
3. make sure you properly update the path to the memory IP during synthesis (in the link commands) and during PnR in the .view file and init.tcl

QUESTION 8-6 : this question is complementary to QUESTION 7-5. If you do this project with a friend, try the following : one does 7-5 with a 16x64 sramHD while the other one uses a Single Port High Density leakage Control Register File. Cf screenshot.



9. ADDITIONAL EXPLORATIONS

9.1. EXPLORING POST PNR POWER ON PRIMETIME

As we did in the first session, it is possible to extract the power from the circuit considering post PnR parasitics.

Let's start with the tutorial test-case. Svt cells, 4.5ns clock period, default floorplan.

1. Follow the same flow as what you did for primtime in the tuto 1
2. Make a copy of the ptime_svt.tcl script and call the new file ptime_svt_pnr.tcl
3. Open the newly created file with a text editor (e.g., gedit)
4. If you did not do it before, make sure that the path to the memory is correct.

The corrected line 14 as described on the moodle forum should be :

```
set target_library "$target_library IPS/MEMORIES/sramHD_64x64/ss1p08v125c/sramHD_64x64.db"
```

5. Update line 29 to point to the netlist you extract post place and route

This could be for e.g. :

```
read_verilog ../HDL/PLACED/tutorial/DATE/TIME/top_32b_placed_sim.v
```

6. Comment or delete lines 32 33 34 and 35
7. Save the file and close it.

Start pt_shell and source the file you just edited. At this point, the script will import the libs, the design, and define the clock constraints.

```
pt_shell > source BIN/ptime_svt_pnr.tcl
```

Make sure that the tool does not return you errors. You should be comfortable enough understanding the logs from pt_shell, as you did that in the first part of this tutorial.

run a update and report power of the design at this point

```
pt_shell> update_power
pt_shell> report_power
```

QUESTION 9-1 : how different is that from the report power you did with the same conditions post synthesis ? and why should it be different ?

Let's now add the RC parasitics you did generate during PnR (called SPEF file).

```
pt_shell> read_parasitics -format spef
../../INNOVUS/SPEF/tutorial/DATE/TIME/top_32b_rcbest.spef
```

Make sure you properly update the DATE and TIME

At this point the tool may issue some warnings about some extrapolation issues. You can ignore these warnings here. Though do not take this as a general rule.

```
pt_shell> update_power
pt_shell> report_power
```

QUESTION 9-2 : how does it compare to the results of question 9-1? Why and what's different now ?