Advanced Vlsi Project

Bitcoin

Stage 3 - Floorplan

שמות המגישים	אור שאול	עמרי אורן	סטאניסלאב קוגן	נועה פרקש

Workarea path for Stage 3: /project/advvlsi/users/stanislavk2/ws/bitcoin/stage3

Warmup Questions:

1. **[#P3.1_Q1]** Why is the floorplan stage important in physical design?
The floorplan stage is essential in physical design for various reasons:

- **Chip Area Optimization:** Efficiently placing different blocks and modules can reduce the overall chip area, resulting in cost savings and enhanced performance.
- **Design Closure Facilitation:** A robust floorplan lays a strong foundation for later stages like placement and routing, helping to meet design specifications and constraints.
- **Performance Enhancement:** A well-planned floorplan can shorten interconnect lengths between blocks, reducing signal delays and improving the chip's overall speed.
- **Power and Signal Integrity Management:** Effective floorplanning ensures proper power distribution and signal integrity by minimizing noise and crosstalk.
- **Manufacturing Considerations:** The floorplan must account for manufacturing capabilities and limitations to ensure efficient and effective production.

In summary, the floorplan stage establishes the foundation for the entire physical design process, influencing performance, power consumption, manufacturability, and cost.

2. **[#P3.1_Q2]** What are some of the factors to consider when creating a floorplan?
When creating a floorplan, several factors need to be considered:

- **Block Placement:** Functional blocks must be optimally placed on the chip to minimize delays and optimize performance, while also reducing wire length and congestion.
- **Heat Dissipation:** Plan for effective heat dissipation by allocating regions for thermal pads and heat sinks, optimizing the floorplan to prevent localized heating and thermal issues.
- **Design Constraints:** Consider any design constraints imposed by the target technology, design rules, performance targets, and functional requirements.
- **Power Distribution:** Allocate regions for power supplies, decoupling capacitors, and voltage regulators to ensure proper voltage levels and minimize power noise, considering power routing and voltage drops across the chip.

3. **[#P3.1_Q3]** Congestion:

    3.1. What is congestion in a floorplan?

    In floorplanning, congestion refers to the situation where a specific area of the integrated circuit (IC) layout has a high density of routing resources like metal wires or vias. This happens when there is insufficient space for routing paths, making it challenging to route signals efficiently.

3.2. Why is congestion a concern in floorplanning?
Congestion in floorplanning poses several concerns:

- **Timing Issues:** Congestion can cause longer interconnect lengths, adding delays and affecting the design's overall timing requirements.
- **Power Distribution Challenges:** It can limit power distribution resources, making it difficult to ensure proper power delivery and potentially causing power supply noise issues.
- **Routing Difficulties:** High congestion makes it challenging to find appropriate routing paths for signals, leading to longer routes, increased delays, and degraded signal integrity.
- **Manufacturability Concerns:** Congestion complicates manufacturing processes like lithography or etching, which can result in yield issues or manufacturing defects.

In summary, congestion impacts timing, power distribution, routing, and manufacturability, making it a significant concern in floorplanning.

4. **[#P3.1_Q4]** In what situations might a worker need to do manual work to assist an automatic program in floorplanning?
Manual work in floorplanning is required when automatic tools face challenges with complex or irregular layouts, optimizing critical paths, managing power and thermal distribution, resolving high interconnect congestion, or correcting design rule violations, necessitating human intervention to refine and ensure a manufacturable design.

Option 1: Automatic floorplan

Read the relevant tcl script located at the directory and answer:

2.1. Explain how the parameters core_utilization and side_ratio in set_auto_floorplan_constraints procedure effect the automatic floorplan:
Core_utilization- This refers to the percentage of core area taken up by standard cells. In the tcl script provided to us it is 0.5, i.e. 50% of core area is taken up by standard cells.
Side_ratio- The side ratio given to us in tcl script = {1 2}. This means that longer side of the floorplan is twice the length of shorter side.
2.2. What metals are we using for the power grid?
The metal layers used for the power grid are – M1, M5, M6, M7, M8 and M9.

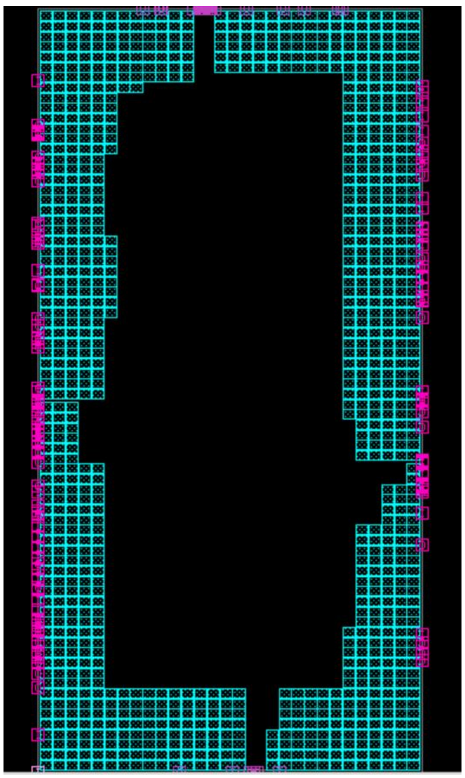
Task X:

[#P3.2_Q1]

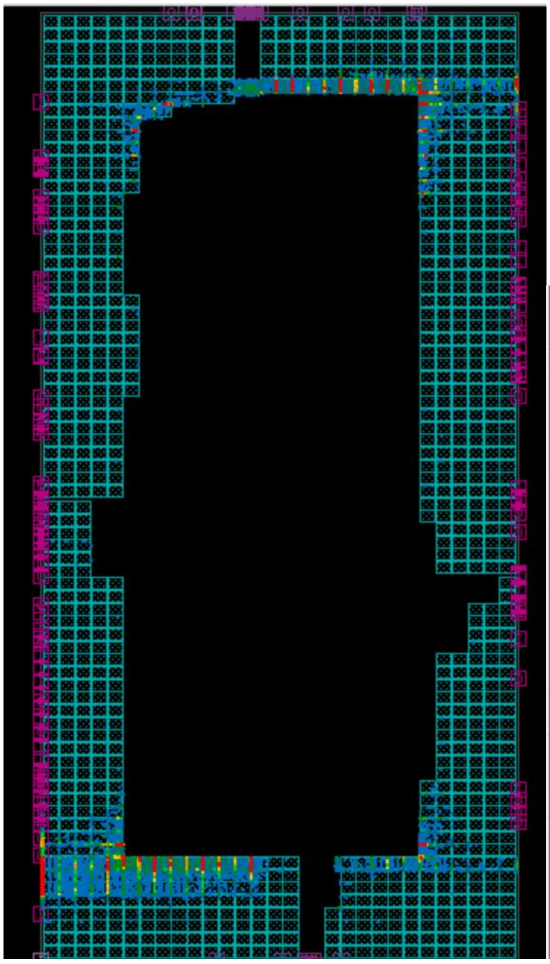
2.1.

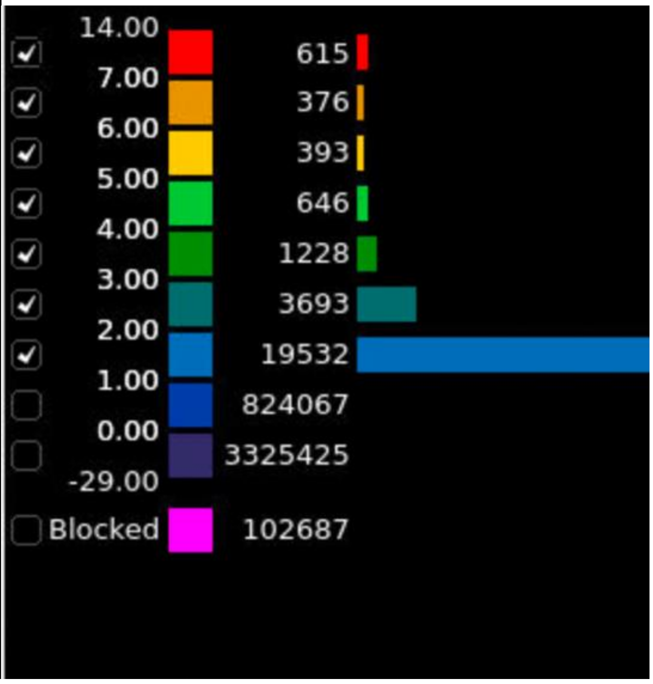
It's important to note that this section was completed with the assistance of the Elad due to several technical issues that arose. Therefore, I am not confident in the accuracy of the results obtained.

Layout without congestion map:



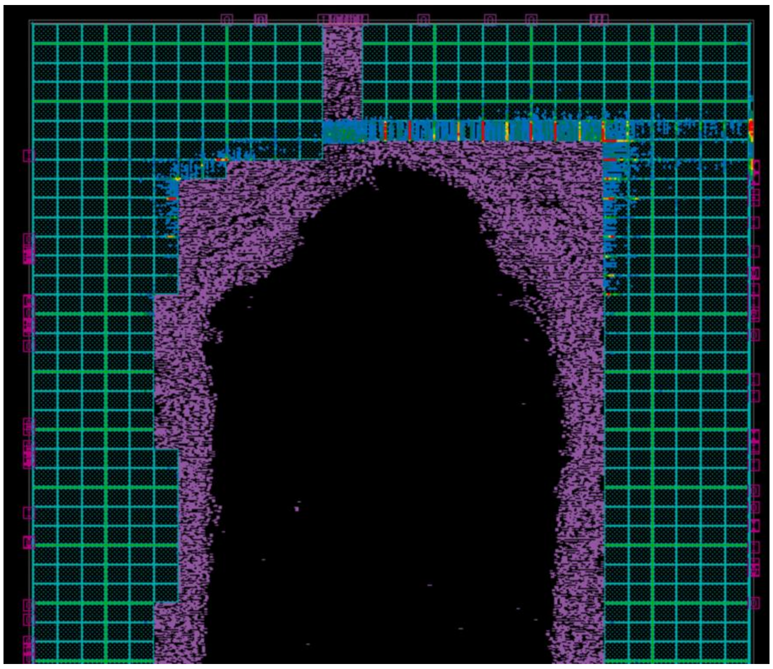
Layout with congestion map:

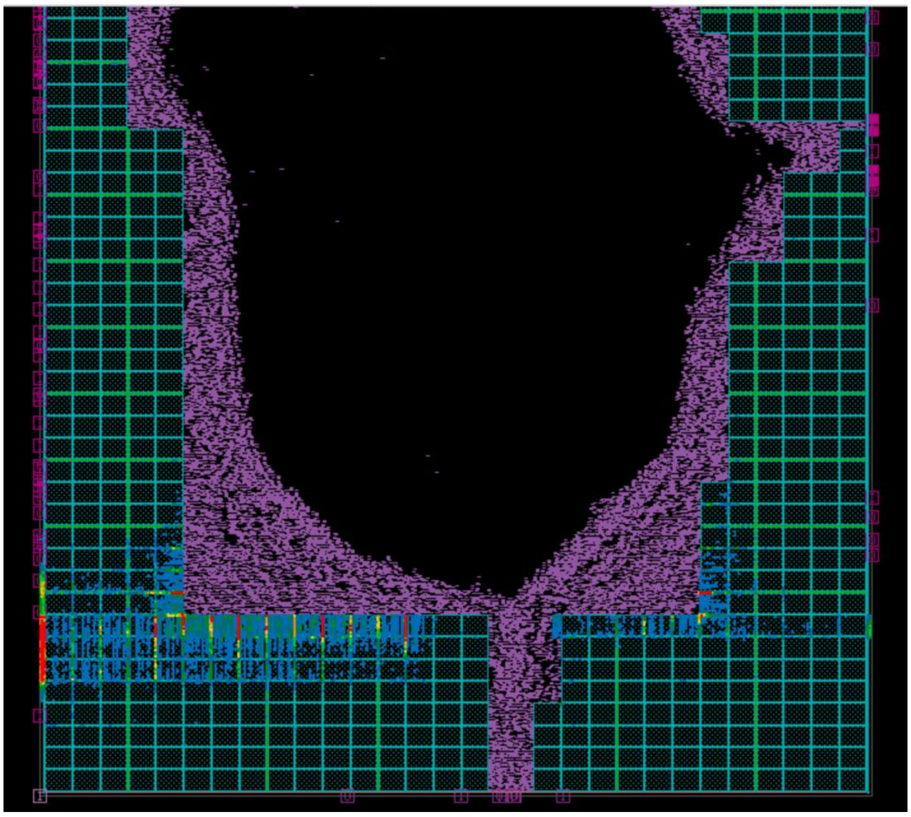




```
   14.00
✓        🟥     615
    7.00
✓        🟧     376
    6.00
✓        🟨     393
    5.00
✓        🟩     646
    4.00
✓        🟩    1228
    3.00
✓        🟦    3693
    2.00
✓        🟦   19532
    1.00
☐        🟦  824067
    0.00
☐        🟪 3325425
  -29.00
☐ Blocked 🟪  102687
```

Layout with congestion map zoomed in:





2.2. List the major pros and cons of the floorplan option:

Pros:

1. he design features a clear separation between standard cells and macros (SRAMs), with standard cells located centrally and macros placed along the boundaries.
2. This layout tends to reduce congestion, as there are minimal blockages and ample space for routing between standard cells, providing large routing channels.

Cons:

1. The boundaries have a high density of SRAMs, with limited space between macros compared to option 2. This results in numerous overflows, as indicated by the congestion map.
2. The high density of macros in the corners creates narrow channels with many pins, potentially leading to significant congestion in those areas.

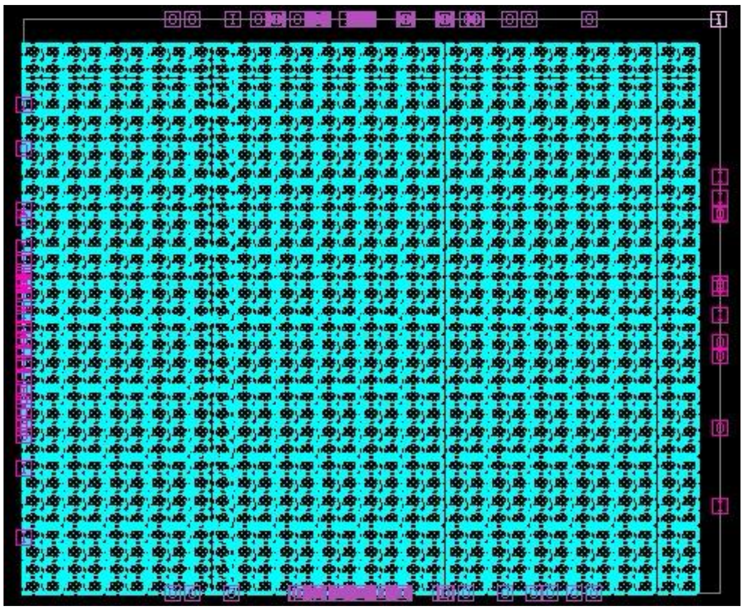
2.3. The floorplan can be optimized by increasing core utilization, which is currently only at 50%. A closer inspection of the layout reveals a significant amount of unused space in the central region. This empty space could be filled with standard cells, freeing up more room for macros to be distributed more evenly. As a result, this would significantly reduce congestion.

Option 2: Re-organized the location of the SRAMs into a matrix shape (fixed spaces)

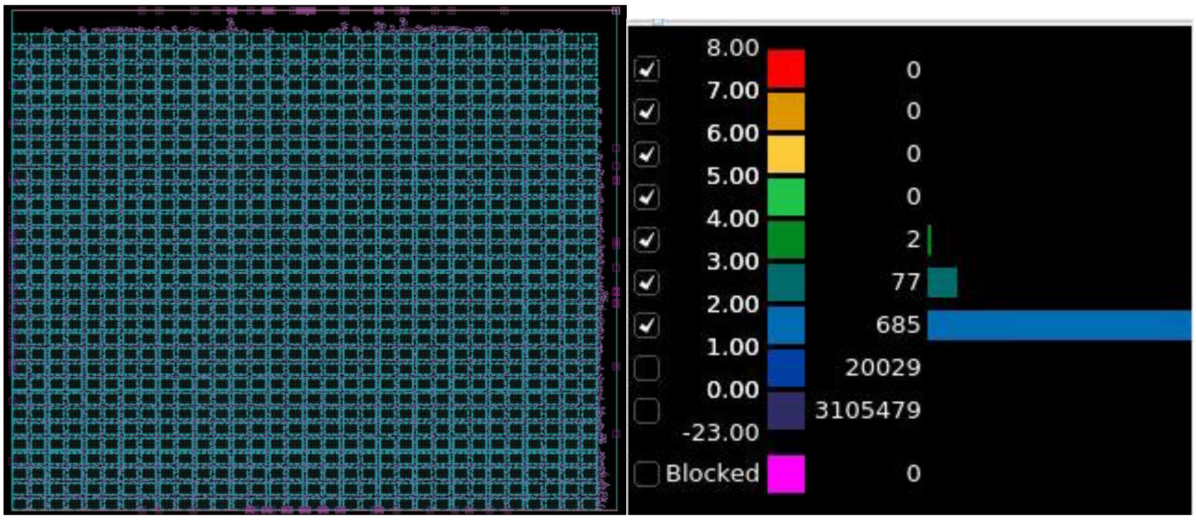
Task X:

2.1. Print screen of the layout with and without the congestion map.

Layout without congestion map:



Layout with congestion map:



2.2. List the major pros and cons of the floorplan option:

Pros:

1. The current design has no significant overflows.
2. Macros are more evenly distributed, providing more routing channels between them compared to the previous design.

Cons:

1. There isn't a distinct separation between regions for standard cells and macros. This lack of separation complicates routing between standard cells and macros, as many macros obstruct the path, leading to congestion in the limited routing channels.

2.3. A clearly defined central area dedicated to standard cells should be established, with the surrounding space allocated for macros. This arrangement minimizes the risk of bottlenecks and reduces congestion. Additionally, proper utilization of blockages and adequate spacing between macros will enhance pin accessibility.

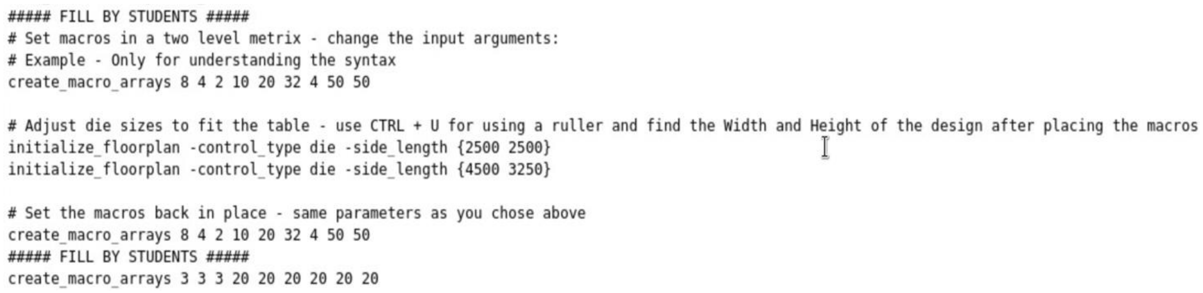
Option 3: Re-organized the location of the SRAMs into a matrix shape (unfixed spaces)

2.1. What does the function do?
The `create_macro_arrays` function plays a key role in determining the floorplan. The parameters we input into this function dictate the arrangement of the macros, the spacing between them, and the number of columns and rows in the final layout.

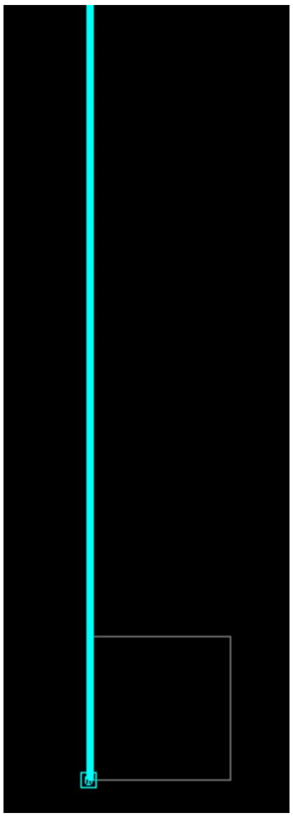
2.2. How can we set the input arguments to get a floorplan like option 2?
In this option, after reviewing the layout, we found that the size of each macro array is set to 1, with a spacing of 15 units between consecutive macros. Each row contains a total of 32 macros. Therefore, the input argument should be: 1 1 1 15 15 15 32 32 32.

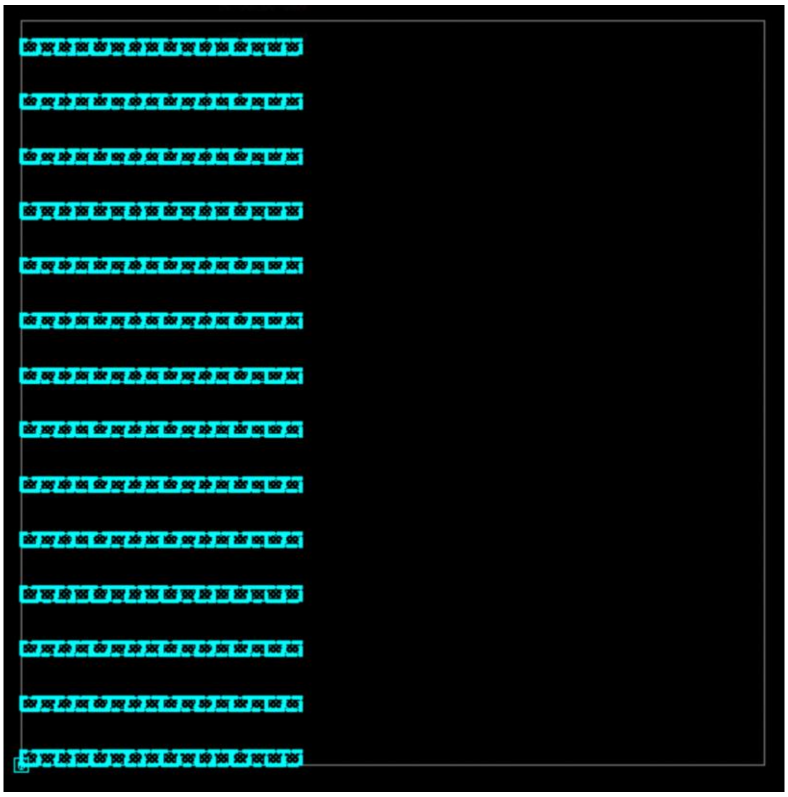
3. Run the "create_macro_arrays" procedure with different input arguments to understand better its functionality and its effect on the final floorplan. Then, decide on the best configuration and write it down in your report. Explain your decision.
Upon close examination, we discovered that the first three bits define the size of each macro-array. The next three bits determine the spacing between consecutive macros within a macro-array, and the final three bits specify the number of macro-arrays per row. We tested the design with different input parameters, as illustrated in the examples shown in the commented section of the screenshot below.

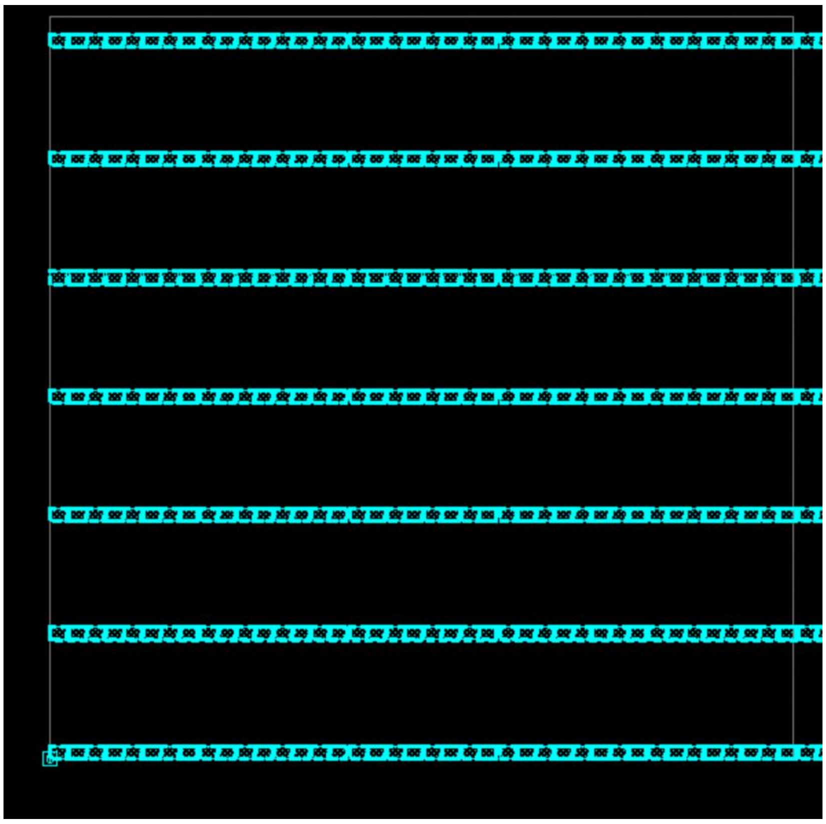
Code:



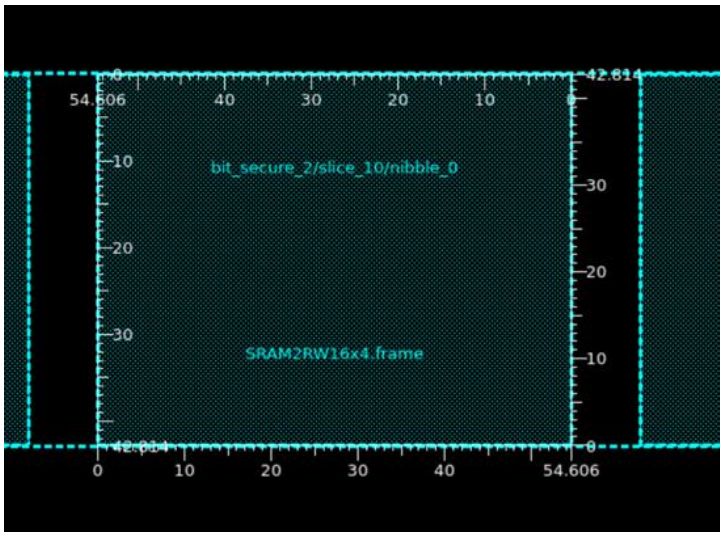
```
##### FILL BY STUDENTS #####
# Set macros in a two level metrix - change the input arguments:
# Example - Only for understanding the syntax
create_macro_arrays 8 4 2 10 20 32 4 50 50

# Adjust die sizes to fit the table - use CTRL + U for using a ruller and find the Width and Height of the design after placing the macros
initialize_floorplan -control_type die -side_length {2500 2500}
initialize_floorplan -control_type die -side_length {4500 3250}

# Set the macros back in place - same parameters as you chose above
create_macro_arrays 8 4 2 10 20 32 4 50 50
##### FILL BY STUDENTS #####
create_macro_arrays 3 3 3 20 20 20 20 20 20
```


For 1 1 1 1 1 1 1 1 1


For 4 4 4 4 4 4 4 4 4



For 8 8 8 8 8 8 8 8 8



For 8 8 8 8 8 8 8 8 8 zoomed in

After testing various input arguments and applying the insights we've gained, we suggest using the following input argument: 3 3 3 20 20 20 20 20 20.
In this configuration, each macro-array is sized at 3 units, the distance between consecutive macros is 20 units, and there are 20 macro-arrays per row.

Task X:

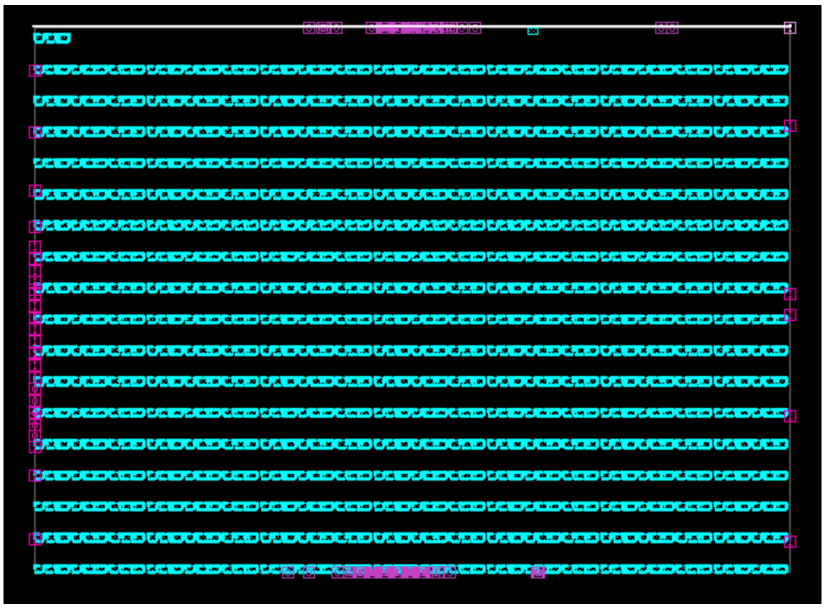
[#P3.2_Q1]

It's important to note that this section was completed with the assistance of the Elad due to several technical issues that arose. Therefore, I am not confident in the accuracy of the results obtained.

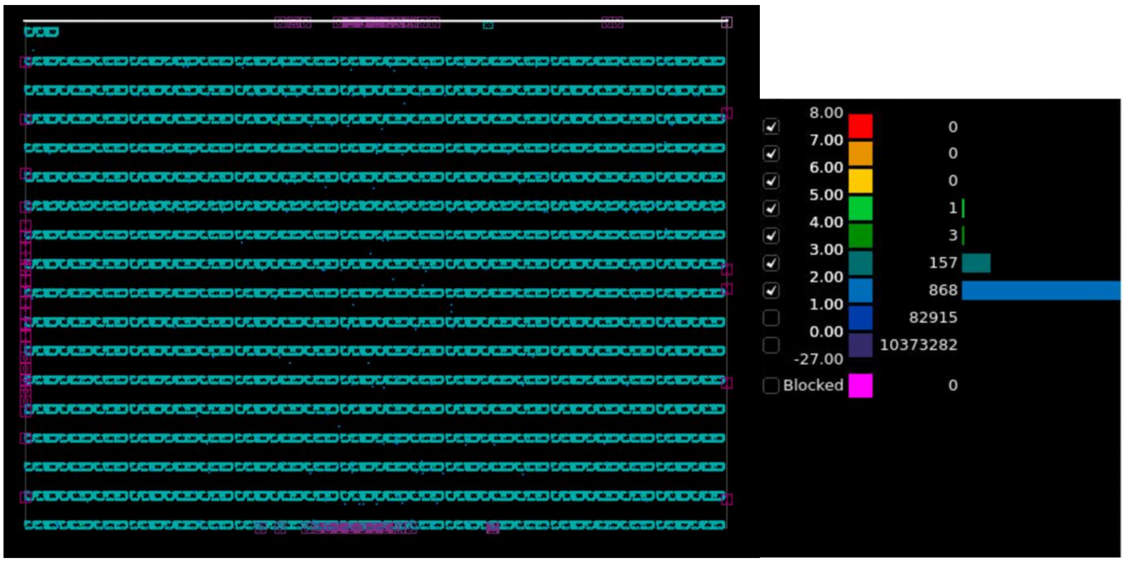
2.1.Print screen of the layout with and without the congestion map.

For 3 3 3 20 20 20 20 20 20:

Layout without congestion map:



Layout with congestion map:



2.2. List the major pros and cons of the floorplan option:

Pros:
The substantial space between each pair of macro-arrays can be utilized to place a significant number of standard cells, creating distinct zones for both standard cells and macros. This separation can help reduce congestion. Additionally, the relatively large spacing between macros within each array increases routing resources and minimizes bottlenecks.

Cons:
Because of the matrix-style design, there is still no designated area for placing standard cells. It is generally more effective to have a dedicated space for standard cells, free from macros as obstacles. This approach helps to reduce routing congestion and bottlenecks.

2.3. We can enhance the design by designating a large area exclusively for standard cells, while using the remaining space for macros. As previously mentioned, this layout helps to reduce congestion and bottlenecks. It's important to ensure that there remains enough space between the macros so that they are easily accessible to both other macros and standard cells.

Option 4: Divide and conquer

2.2. List the major pros and cons of the floorplan option:

Pros:
This is an instance of a secondary hierarchy. The bit_coin level is composed of 16 bit_top levels, so our primary task is to ensure that the bit_top level is constructed correctly. Once that's done, we simply replicate and align the rows to create the bit_coin stage. This approach significantly reduces manual effort and greatly enhances accuracy.

Cons:
This method is only applicable in situations where a secondary hierarchy is present, making it useful in a limited range of scenarios.

[#P3.2_Q2] After going through all the floorplan options, choose the best floorplan option from those options and explain your decision.

Out of the four options, we believe that option 3 is the best choice. It has the least congestion and fewest overflows compared to the other options, as shown in the congestion map. We are selecting option 3 over option 4 because option 4 is not suitable for most situations. Therefore, we will proceed with option 3 using the input argument `create_macro_array function = 3 3 3 20 20 20 20 20 20`.