Project06: OpenROAD (Final Project)

期末報告

1. Write a step-by-step running flow report of this open-source project. You are required to
   detail the incremental progress of each step based on benchmark designs. Provide several
   screenshots (and/or outputs) of the selected designs in every design-flow stage. Discuss
   the progress of these stages.
2. You are required to submit your preference list of the overall six major design-flow stages,
   synthesis, floorplanning, placement, clock tree synthesis, routing, and finishing,
   before December 7. A lottery-based bidding process will be held to determine the major
   design-flow stage you are going to have an in-depth survey. Based on the announcement
   of bidding result, which will be announced on December 8, perform in-depth experiments
   and analyze and compare the algorithm flows and outcomes of the selected design-flow
   stage. You are encouraged to adjust the internal scripts and to evaluate/observe the effects.
3. Complete a comprehensive report, which contains technical review comments and your

OpenROAD installation Path

https://openroad.readthedocs.io/en/latest/user/BuildLocally.html#prerequisites (與實際安裝過程順序有些微不同)

Step-by-step Running

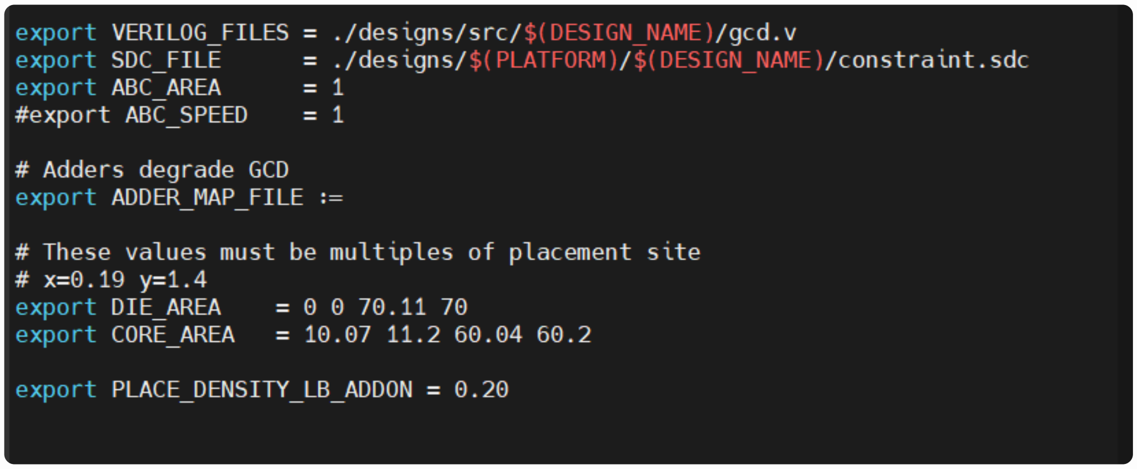
在本節測試流程分成Synthesis, Floorplanning, Placement, Clock Tree Synthesis, Finishing五項流程的結果觀察與分析

Synthesis Explorations

在OpenROAD中是使用yosys
Yosys相關連結:https://yosyshq.net/yosys/about.html

1. **進行區域優化比較(使用nangate45/gcd)**

   - **gcd 的 config.mk file 內容**



```
export VERILOG_FILES = ./designs/src/$(DESIGN_NAME)/gcd.v
export SDC_FILE      = ./designs/$(PLATFORM)/$(DESIGN_NAME)/constraint.sdc
export ABC_AREA      = 1
#export ABC_SPEED    = 1

# Adders degrade GCD
export ADDER_MAP_FILE :=

# These values must be multiples of placement site
# x=0.19 y=1.4
export DIE_AREA    = 0 0 70.11 70
export CORE_AREA   = 10.07 11.2 60.04 60.2

export PLACE_DENSITY_LB_ADDON = 0.20
```

   - **當ABC_AREA =0時，Design area 653 u^2 27% utilization.:**

Number of wires	Number of wire bits	Number of cells
371	417	346

   - **當ABC_AREA =1時，Design area 584 u^2 24% utilization.:**

Number of wires	Number of wire bits	Number of cells
387	433	362

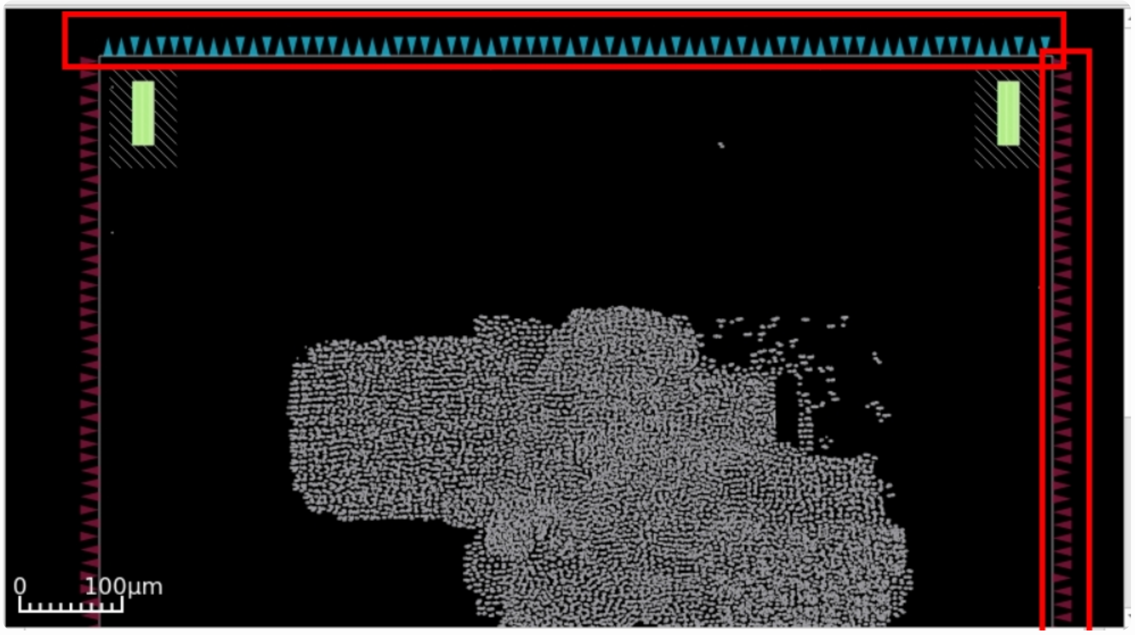
Floorplanning

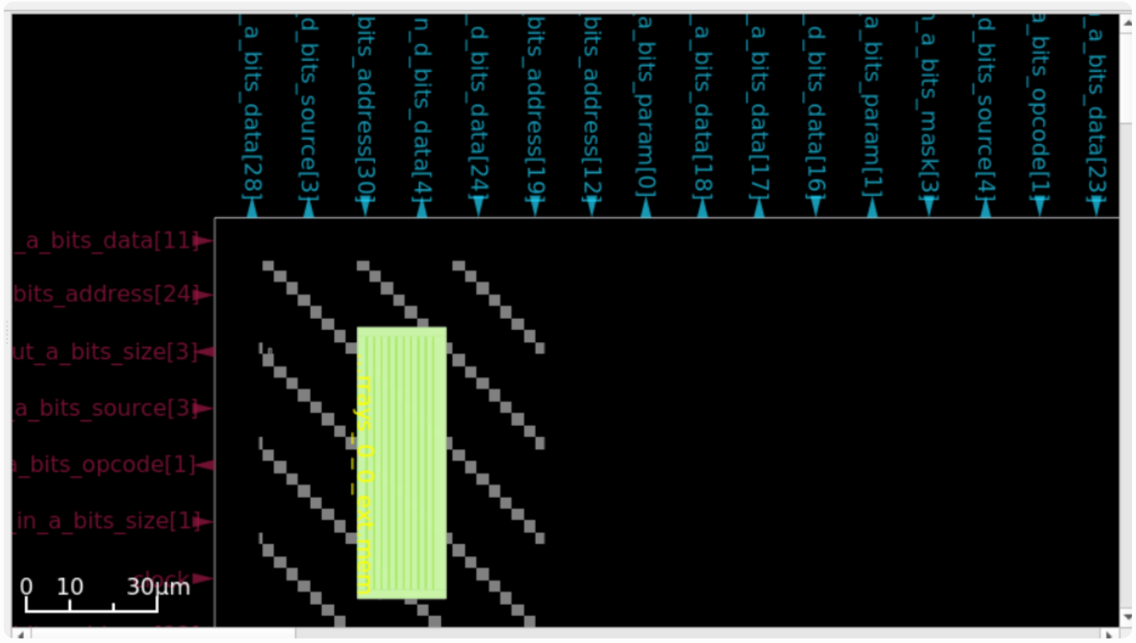
在OpenROAD的Floorplanning中可以細分成Translate Verilog、IO、TDMS、Macro、Tapcell、PDN步驟

1. **IO Placement (Random)**

   - **說明**
     IO pin腳位置隨機生成

- **輸出結果**

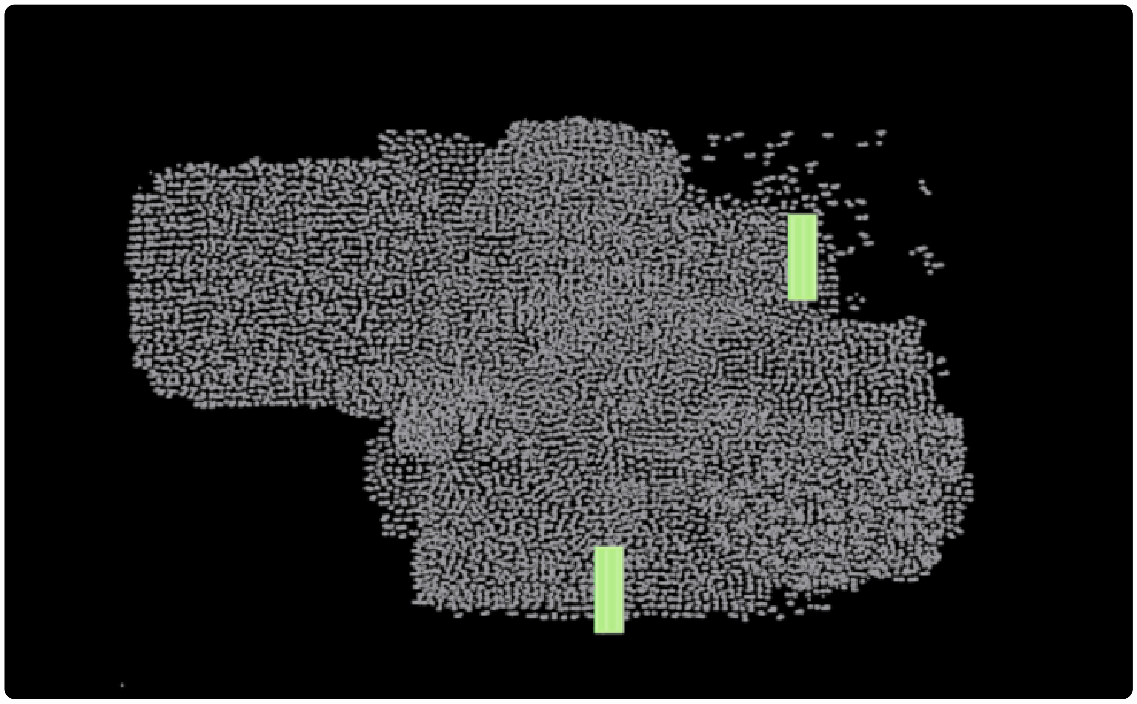




2. **Timing Driven Mixed Sized Placement(不懂這部分)**

   - **說明**
     猜這段是根據所有的元件及module時序進行Floorplanning

- **輸出結果**

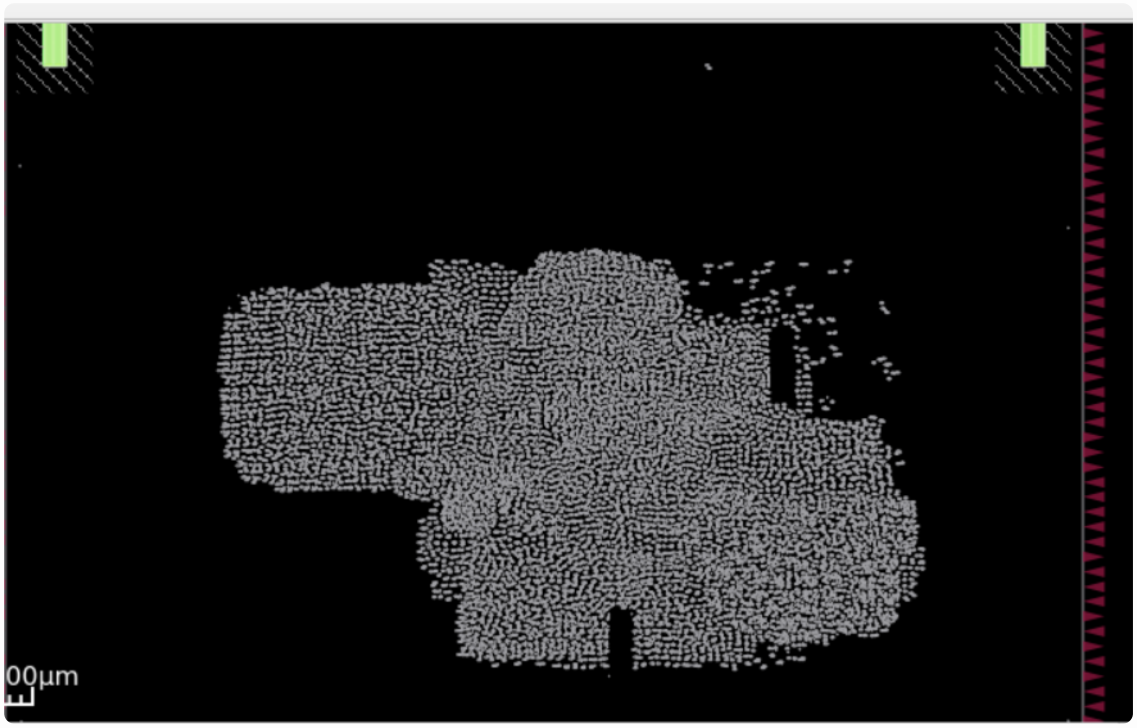
  

3. **Macro Placement**

   - **說明**

     Macros 可以分成 hard macros, firm macros 和 soft macros, 這個部分將進行macros and standard cells的擺放，對於好的擺放對於後續Placement時序和面積的影響性相當重要。

   - **輸出結果**

4. **Tapcell and Welltie insertion(不是很懂這部分)**

- **說明**

  文件上介紹

  > Tap cells are non-functional cells that can have a well tie, substrate tie or both. They are typically used when most or all of the standard cells in the library contain no substrate or well taps. Tap cells help tie the VDD and GND levels and thereby prevent drift and latch-up.
  >
  > The end cap cell or boundary cell is placed at both the ends of each placement row to terminate the row. They protect the standard cell gate at the boundary from damage during manufacturing.
  >
  > Tap cells are placed after the macro placement and power rail creation. This stage is called the pre-placement stage. Tap cells are placed in a regular interval in each row of placement. The maximum distance between the tap cells must be as per the DRC rule of that particular technology library.
  >
  > The figures below show two examples of tapcell insertion. When only the `-tapcell_master` and `-endcap_master` masters are given, the tapcell placement is similar to Figure 1. When the remaining masters are give, the tapcell placement is similar to Figure 2.
  >
  > Refer to the GUI figures to highlight well tap and end cap cells. The image does not differentiate and just shows a bunch of rectangles.

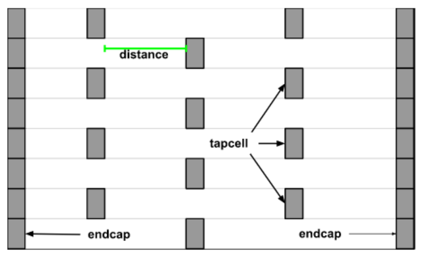


Figure 1: Tapcell insertion representation

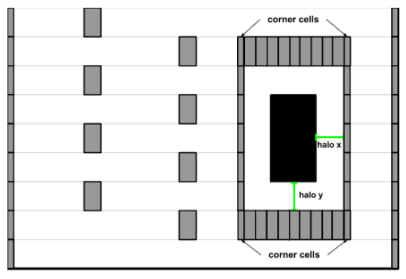
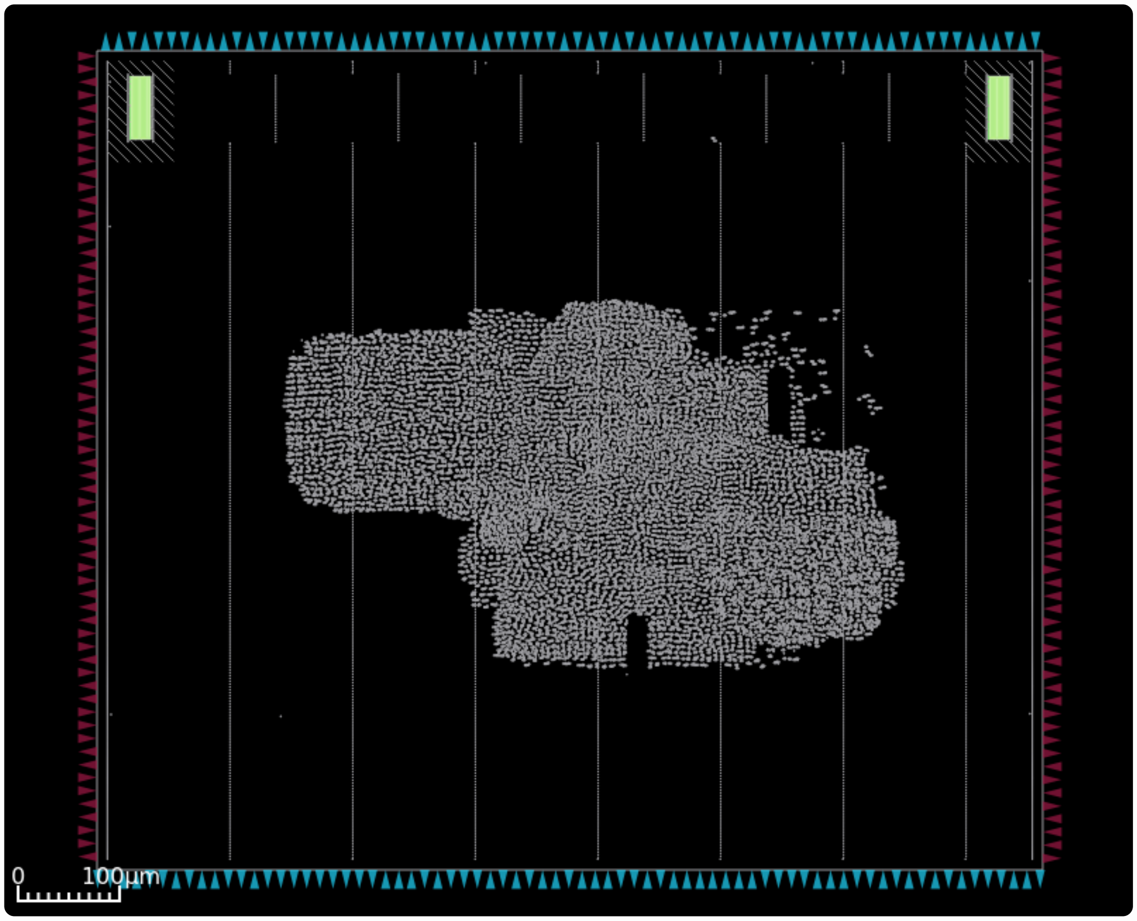


Figure 2: Tapcell insertion around macro representation

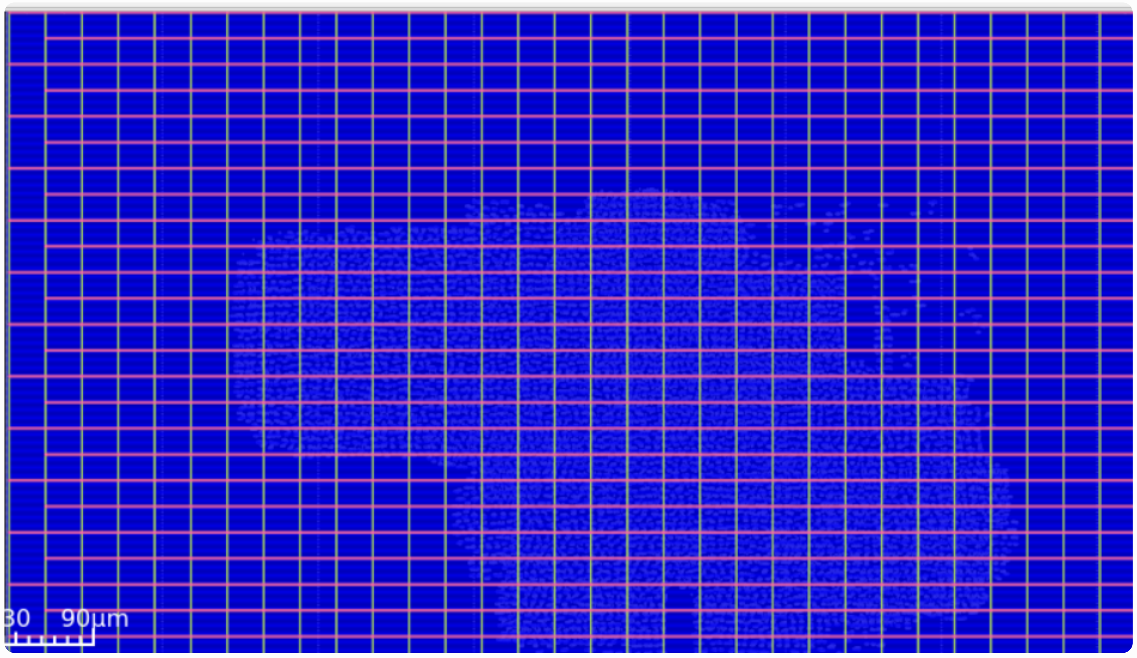
- **輸出結果**

  圖中垂直的線



5. **PDN generation**

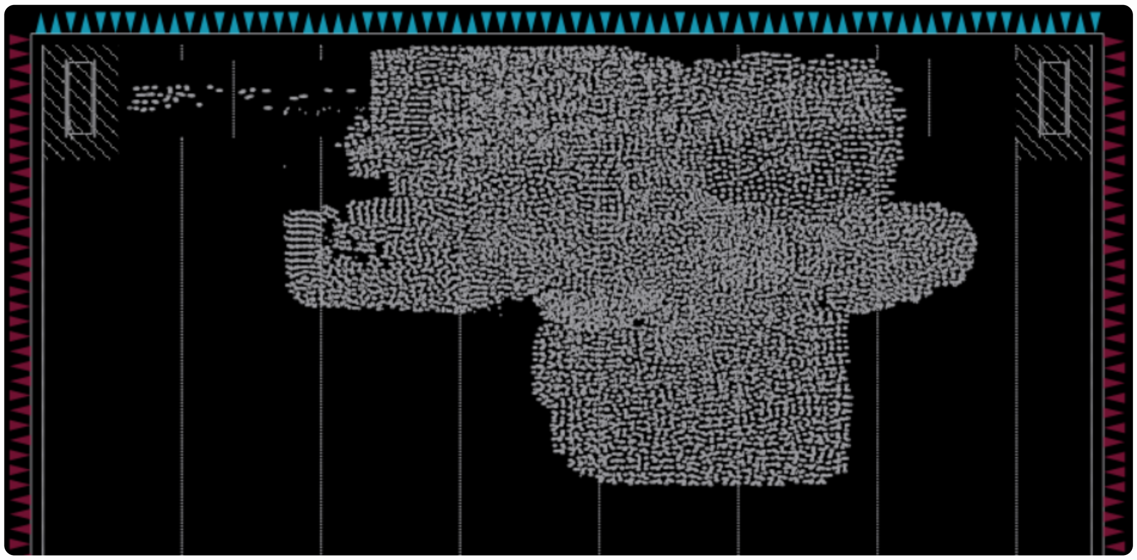
- **說明**
  PDN(Power Delivery Network) 此部分進行電力布局，產生VDD(紅線)與VSS(黃線)
- **輸出結果**



Placement

1. **Global placement without placed IOs, timing-driven, and routability-driven**

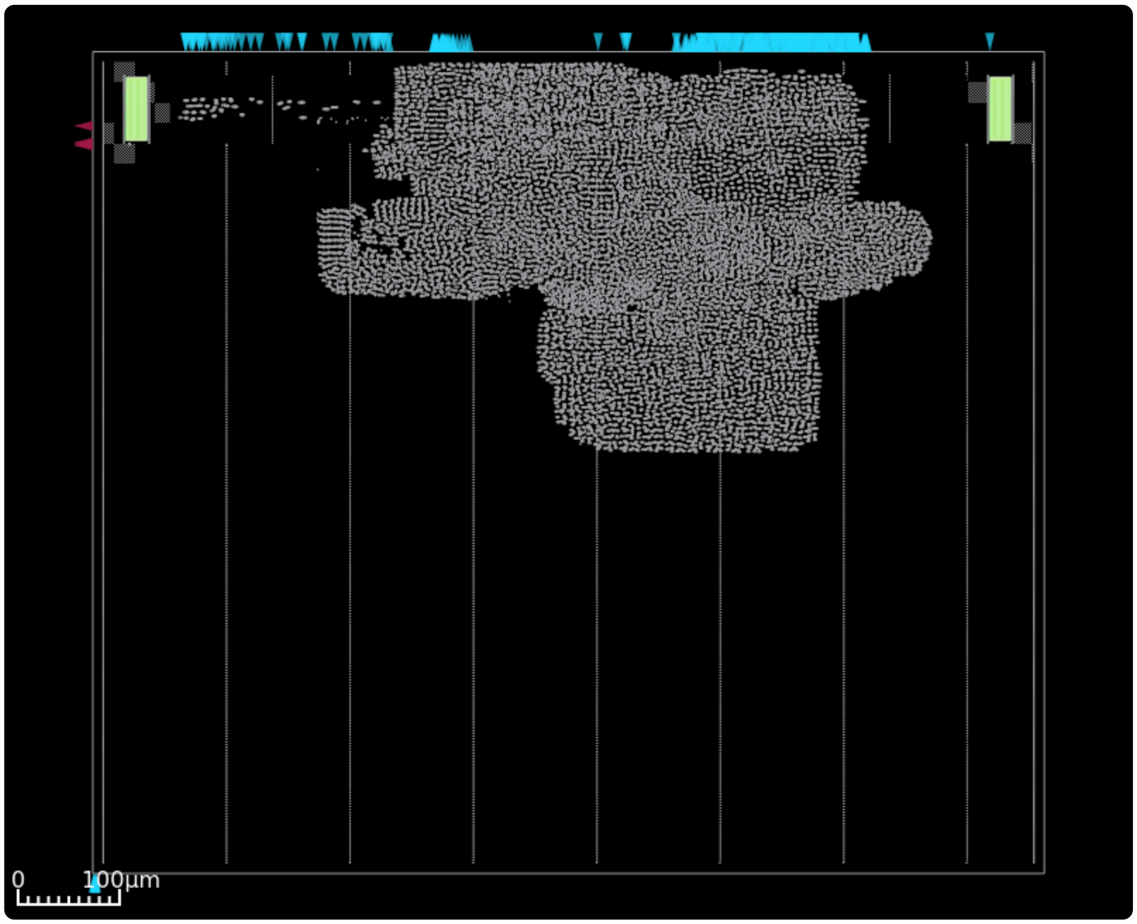
   - **說明**
     針對邏輯閘等元件進行位置布局
   - **輸出結果**



2. **IO placement**

   - **說明**
     於cell的placemet後進行IO的placement，將pin腳以最大程度地減少淨線長放置

- **輸出結果**

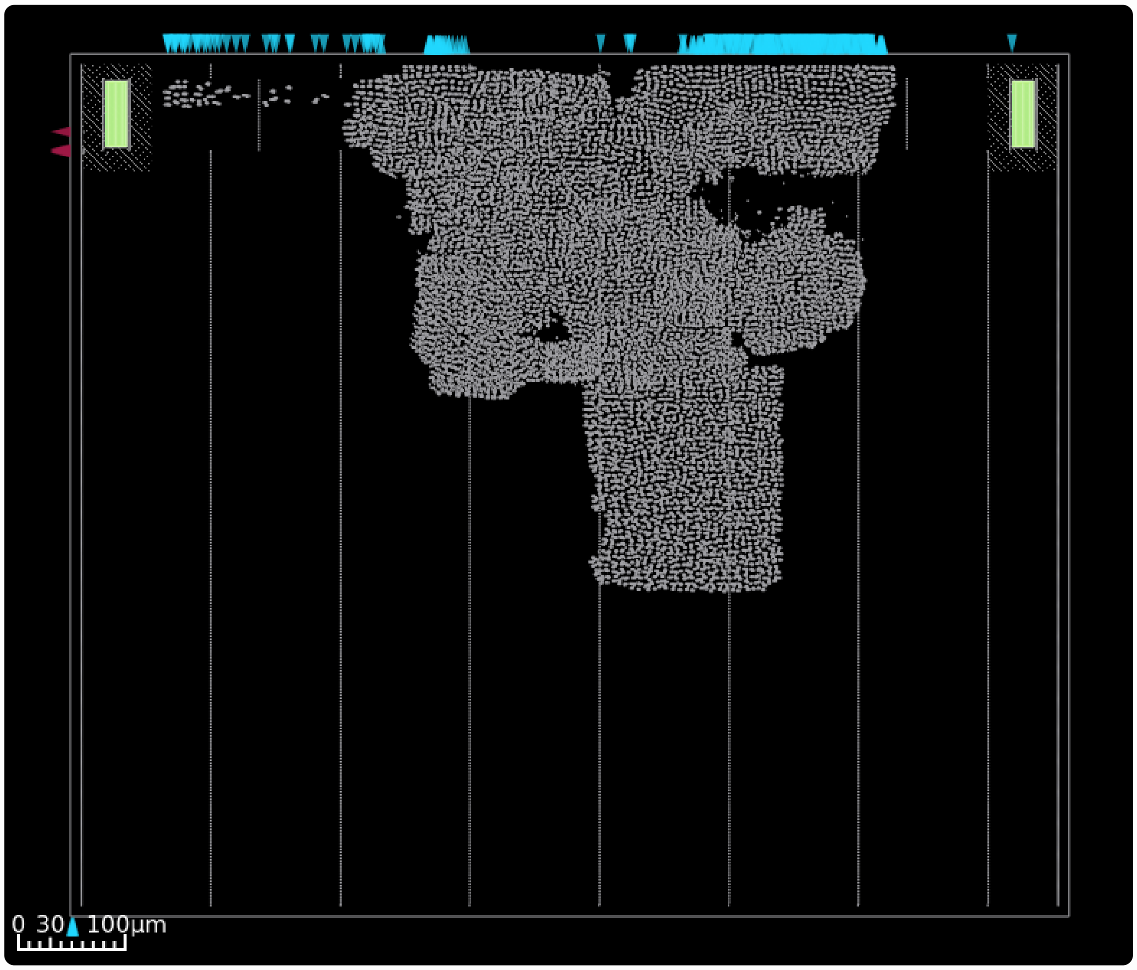
  

3. **Global placement with placed IOs, timing-driven, and routability-driven**

   - **說明**

     對第一步與第二步進行更完整的調整，內容包含IO、timing、routing的可行性以及時序上調整

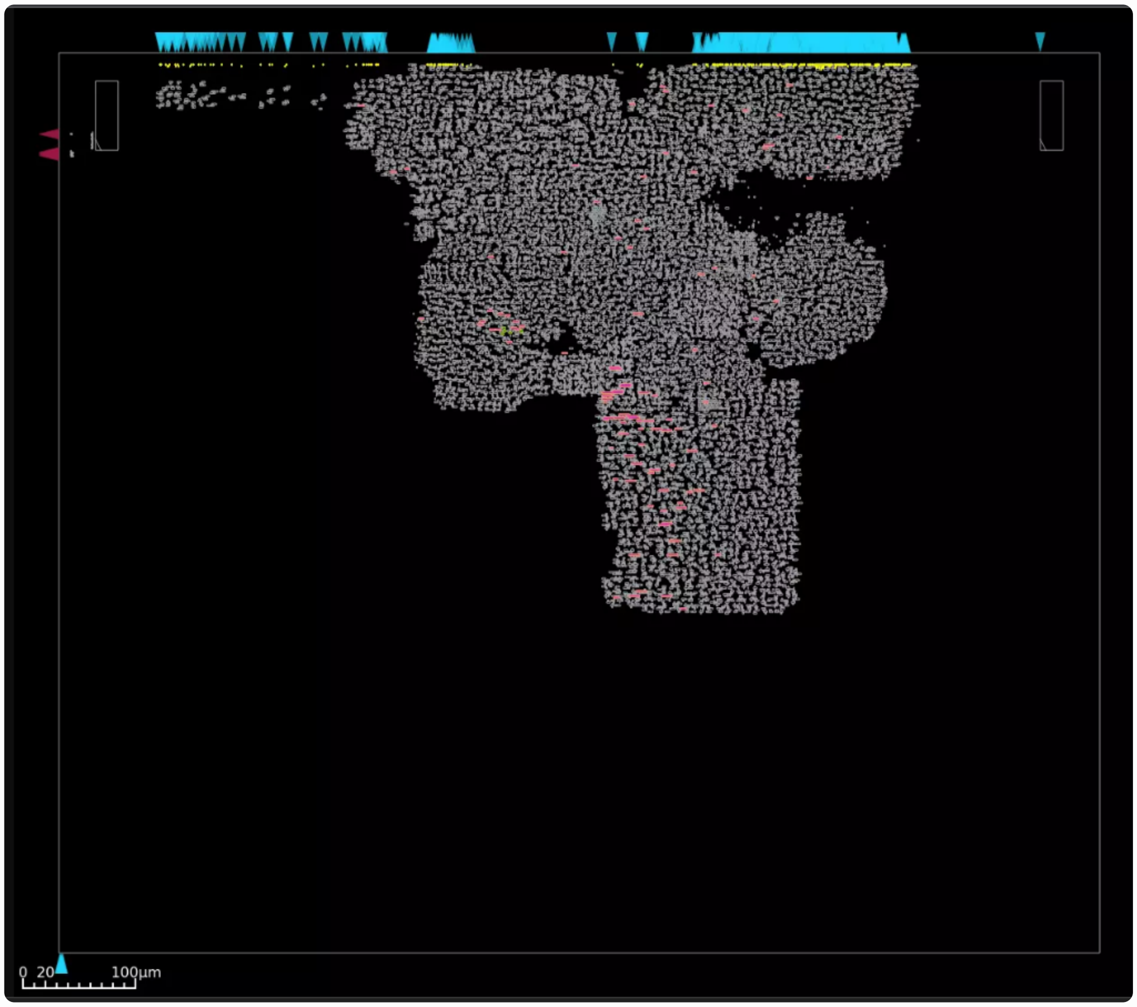
- **輸出結果**



Design area 49155 u^2 7% utilization.

4. **Resizing & Buffering**

- **輸出結果**

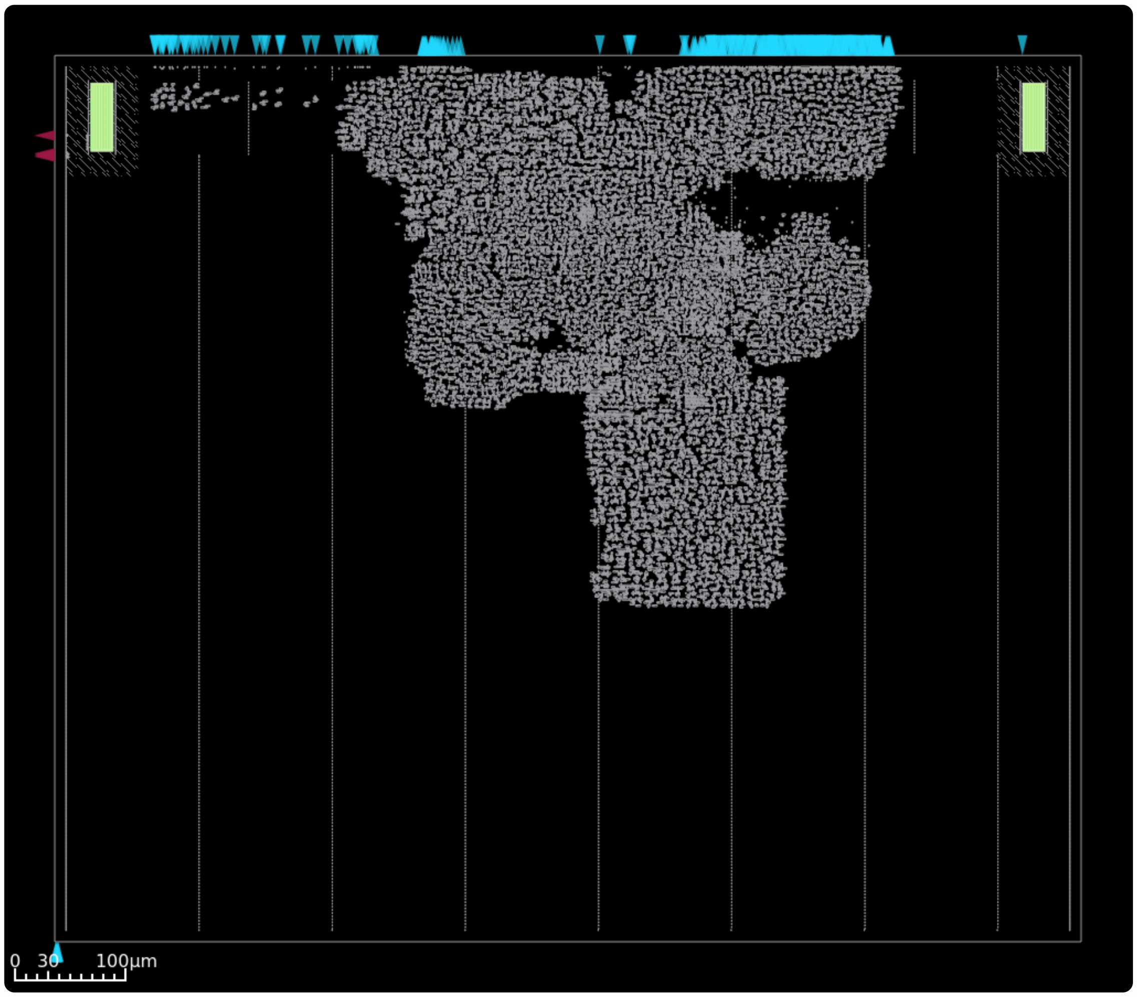


5. **Detail placement**

    - **說明**

      對於Global placement後結果在更細部進行微調，從Resizing & Buffering的部分有一些地方是可以進行調整的

- **輸出結果**



Design area 53289 u^2 8% utilization.

- **結果分析**
  在Global中的placement中使用了3個步驟分別對Cell、IO初步擺設最後在綜合Cell、IO進行更完整的放置，雖然感覺起來有點多此一舉，但是在擺放的效率上可以更加快速的確定Core物件、IO的整體位置，而Global與Detail之間無法直接看出差異，但是從Design area可以比較出Detail在使用率較Global提升

Clock Tree Synthesis

此部分分為2項TritonCTSFiller、cell insertion

1. **TritonCTS**
   - **輸出結果**
     可以透過進行查看CLK的線路布局，且可以使用TCL commands下達clock_tree_synthesis之命令來優化

2. **Filler cell insertion**

    - **說明(文件介紹)**
      Filler cells fills gaps between detail-placed instances to connect the power and ground rails in the rows. Filler cells have no logical connectivity. These cells are provided continuity in the rows for VDD and VSS nets and it also contains substrate nwell connection to improve substrate biasing.
    - **輸出結果**

Finishing

- **說明**
  在Routing後進行最後的wrapped macros跟GDS

Major design-flow Stage

在Routing中Net分成Signal、Power、Clock、Ground4種線路，而在Routing中透過兩個步驟來進行整體線路的配置以及優化分別是Global Routing和Detail Routing。

- **Goal**:

    1. 使用最少的 DRC(Design Rule Checking) 完成所有signal nets
    2. 優化時序、DRC 和Power的路徑

- **Design Rule Checking(Synopsys說明)**
  DRC checking is an essential part of the physical design flow and ensures the design meets manufacturing requirements and will not result in a chip failure. The process technology rules are provided by process engineers and/or fabrication facility.
  Here are some basic and common types of DRC rules

    1. Minimum width
    2. Minimum spacing
    3. Minimum area
    4. Wide metal jog
    5. Misaligned via wire
    6. Special notch spacing
    7. End of line spacing

1. **Global Routing**

- **說明**
  對於整個電路進行大方向的線路布局，與Global placement功用相似，演算法的部分使用fast approximation algorithm，以確保每個Net的Steiner tree結構。
- **輸出結果**
  Report

2. **Detail Routing**

   - **說明**
     OPENROAD使用TritonRoute進行Detail Routing，TritonRoute將LEF和放置的DEF然後為兩個信signal nets執行詳細的佈線和clock nets給出了route 指南格式的Global Routing解決方案在Detail Routing，（i）TritonRoute使用breadth-first search對Global結果進行預處理，以減少後期產生環路的可能性，同時保留Net連通性(ii) TritonRoute 識別考慮方向和routing軌道偏移的獨特實例，並生成pin腳訪問模式以幫助連接到pin腳訪問模式以幫助連接到pin腳。

   - **輸出結果**
     Report

     第一次優化結果

     最後優化結果

     整體電路 Routing 結果

     電路 Clock Nets 結果

3. **Debugging Problems in Global Routing**
   Global Routing有一些有用的功能來理解設計中的高擁塞問題以及違反DRC規定。

   - Heat Maps

   - DRC Viewer
     查看在Routing時是否有發生違規，查看report中的5_route_drc.rpt

RELATED Analysis

IR Drop Analysis

IR drop is the voltage drop in the metal wires constituting the power grid before it reaches the power pins of the standard cells. It becomes very important to limit the IR drop as it affects the speed of the cells and overall performance of the chip.

- Features:

  1. Report worst IR drop.
  2. Report worst current density over all nodes and wire segments in the power distribution network, given a placed and PDN-synthesized design.
  3. Check for floating PDN stripes on the power and ground nets.
  4. Spice netlist writer for power distribution network wire segments.

- Report

  1. VDD

  2. VSS

Review

藉由OpenROAD的實際操作對於整個Design過程更加了解，特別是在flootplaning的部分，比想像中要處理的事情還要多，而在Timing上，經過上次講師的說明後對於IC設計上相對重要，這點可以從OpenROAD運作流程中發現有一項是特別為了時序進行的，那這次剛好可以抽到Routing，因為以前有對電路板Layout的經驗然後，過去只有一層正反面，無須使用到演算法來進行繞線，還能自己手動繞線時常發生會繞到無處可鑽的窘境，偶爾會藉由自動繞線的方式來支援，深深的體會到是IC的佈線跟我們過去操作的PCB板的佈線技巧完全不同級別的問題，那也因為在這個領域還是處於初心者程度，其實OpenROAD還有許多Tool可以去套用到電路中只是實力還沒到達可以是隨心所欲的操作蠻可惜的部分。

參考網站:

OpenROAD Github:
https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts

OpenROAD文件:
https://openroad.readthedocs.io/en/latest/tutorials/FlowTutorial.html#running-the-automated-rtl-to-gds-flow

VLSI步驟與單元介紹:
https://ivlsi.com/macros-in-vlsi-physical-design/