Digital System Design Project 4 – OpenROAD

1. 下載OpenROAD(取自老師給的下載流程)
   1. 下載OpenROAD

git clone --recursive https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts

cd OpenROAD-flow-scripts

git checkout 14994593a4eefbbea55e812b3bb6a1ef1ffb2d1e

* 1. 安裝OpenROAD相依性套件

sudo ./setup.sh

* 1. 若虛擬機內存不足，則需要透過swap分區創建

安裝指令

sudo mkdir -p /var/cache/swap/

sudo dd if=/dev/zero of=/var/cache/swap/swap0 bs=128M count=64

sudo chmod 0600 /var/cache/swap/swap0

sudo mkswap /var/cache/swap/swap0

sudo swapon /var/cache/swap/swap0

確認指令

sudo swapon -s

刪除指令

sudo swapoff /var/cache/swap/swap0

sudo rm /var/cache/swap/swap0

<https://blog.csdn.net/weixin_44796670/article/details/121234446>

* 1. 安裝 OpenROAD 相依性套件

sudo ./setup.sh

* 1. 編譯 OpenROAD

./build\_openroad.sh --local

* 1. 驗證 OpenROAD 是否正確編譯完成

環境參數設定

source ./env.sh

確認 yosys 及 openroad 可以執行並出現 help 說明

yosys -help

openroad -help

* 1. 執行 OpenROAD

cd ~/OpenROAD-flow-scripts/flow

make

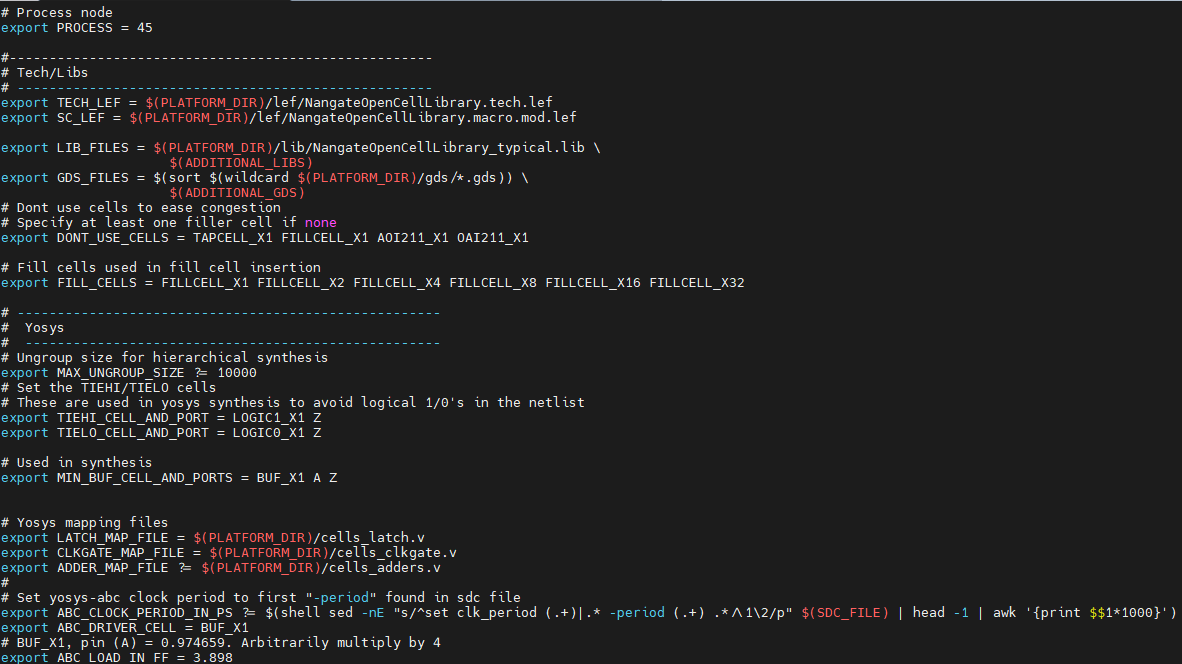
* 1. 可觀察比較之結果

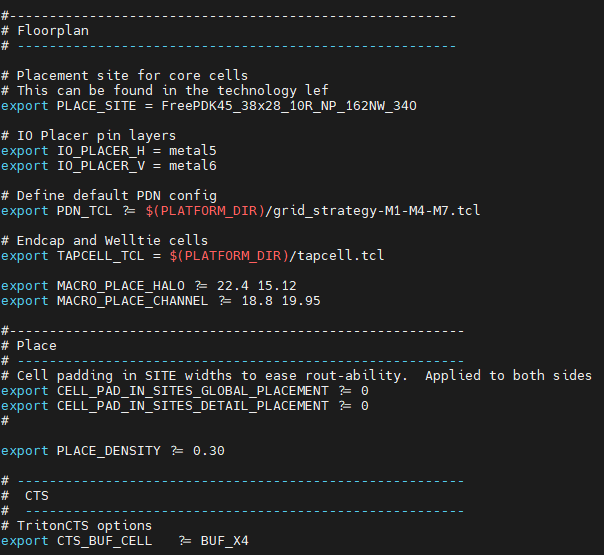
make gui\_ <TAB>

例如: make gui\_final

1. GCD晶片的設計流程
   1. 前處理
      1. 分析製程: 這個文件可以分析使用的製程(process)、使用的 Floorplan、Place、CTS 參數等等。

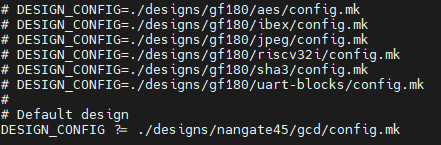
less ./platforms/nangate45/config.mk





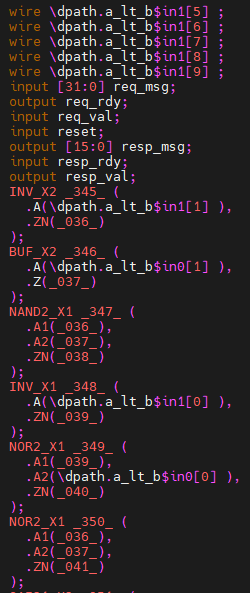
* + 1. 編譯檔案: 此報告將使用預設 nangate45 的 gcd 做分析，不用改動Makefile即可開始後開始編譯

make

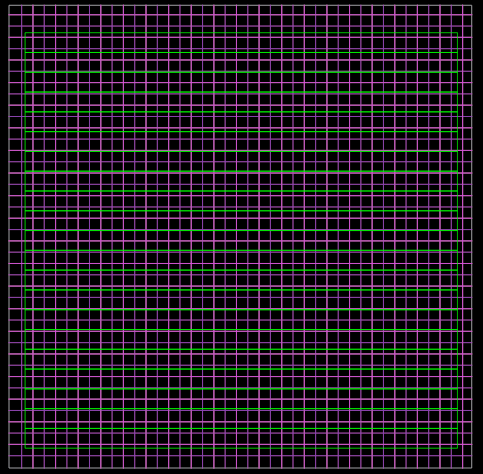


以下透過GUI分析每一步驟的設計配置和流程

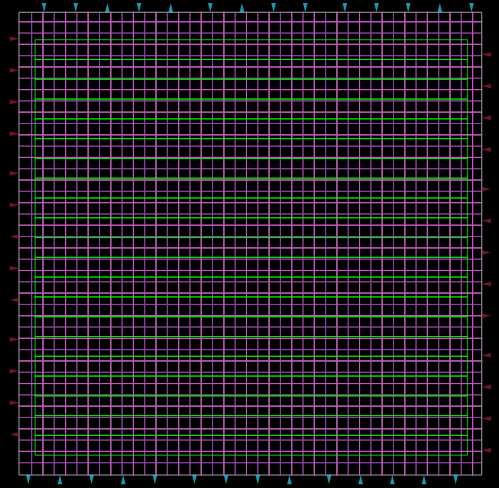
* 1. Synthesis: 用HDL code等方法，做出晶片的功能、面積等最佳化



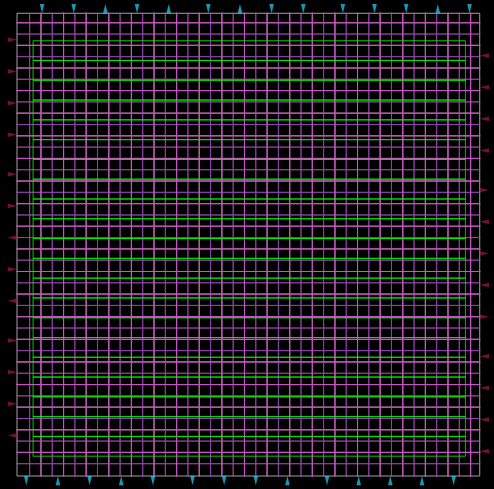
* 1. Floor Planning：決定晶片布局
     1. Translate verilog to odb (把verilog轉變成obd)



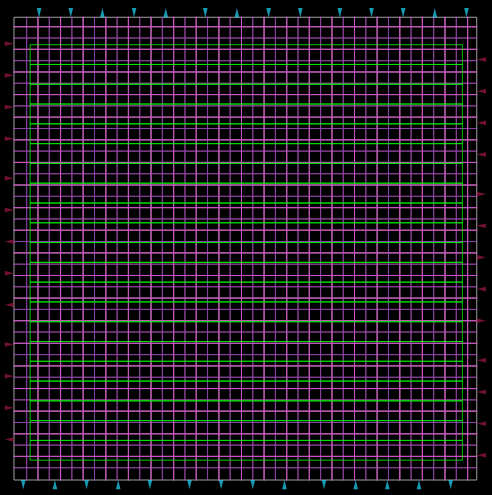
* + 1. IO Placement (隨機擺放IO)



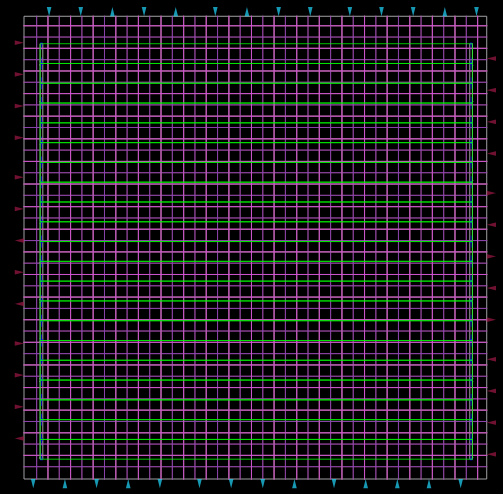
* + 1. Timing Driven Mixed Sized Placement(擺放)



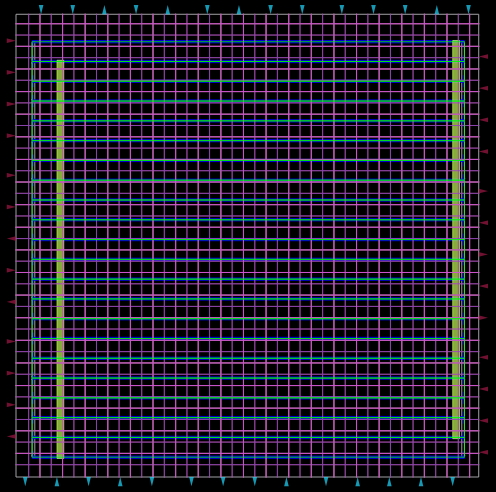
* + 1. Macro Placement(宏觀的擺放)



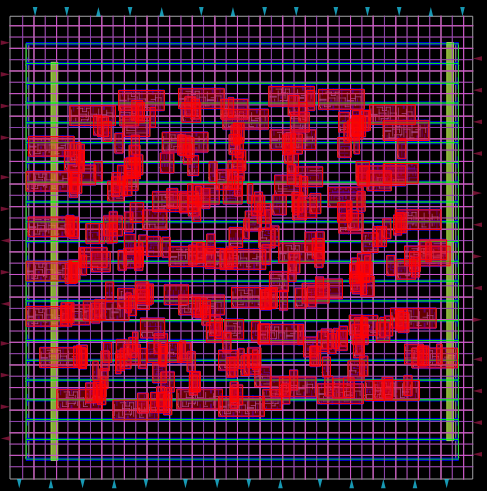
* + 1. Tapcell and Welltie insertion(tap cell與 well tie的擺放)



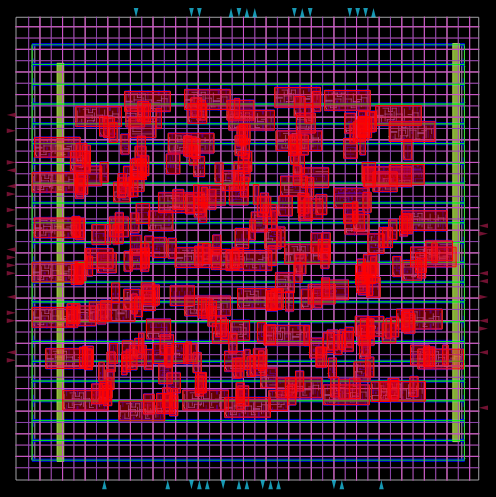
* + 1. PDN generation(電源規劃)



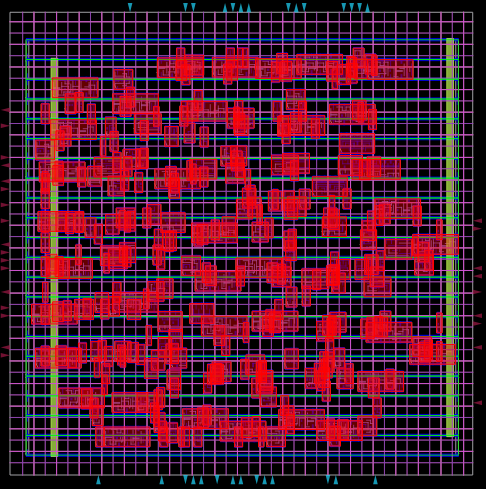
* 1. Placement：放入邏輯閘、元件
     1. Global placement without placed IOs, timing-driven, and routability-driven(大致上的布局，擺放flip flop等)



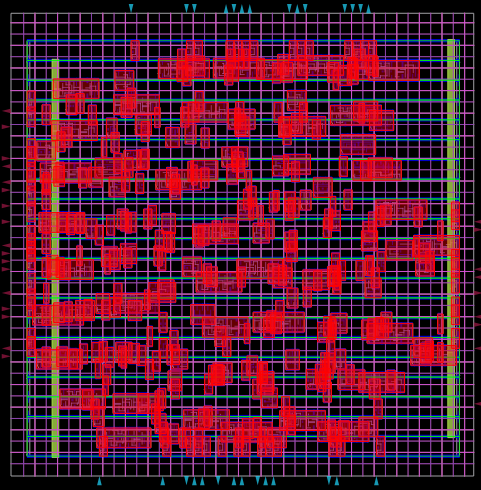
* + 1. IO placement (non-random)(再次布置IO)



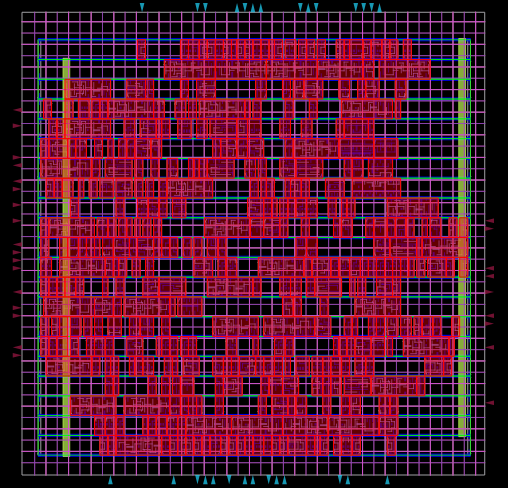
* + 1. Global placement with placed IOs, timing-driven, and routability-driven(根據IO等放置flip flop等)



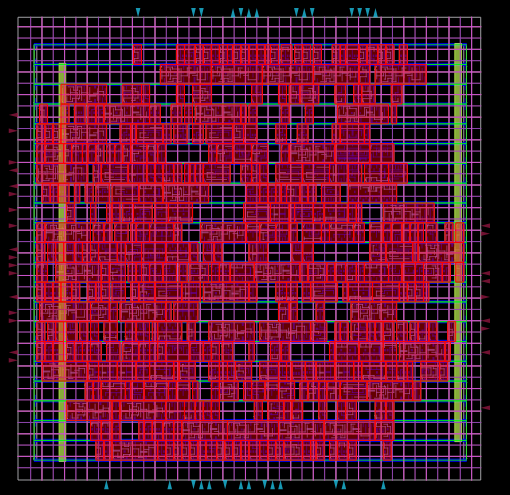
* + 1. Resizing & Buffering(重設大小)



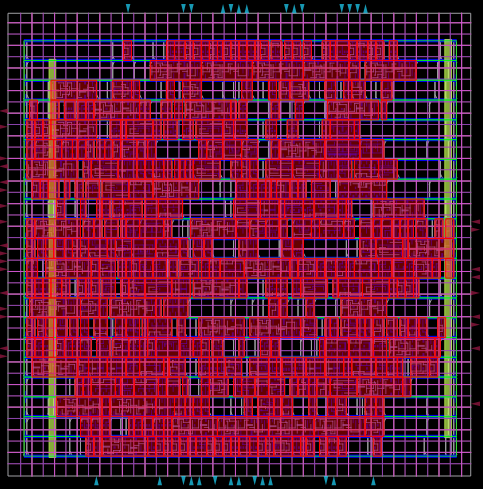
* + 1. Detail placement(詳細的調整，flip flop對齊電源網路等等)



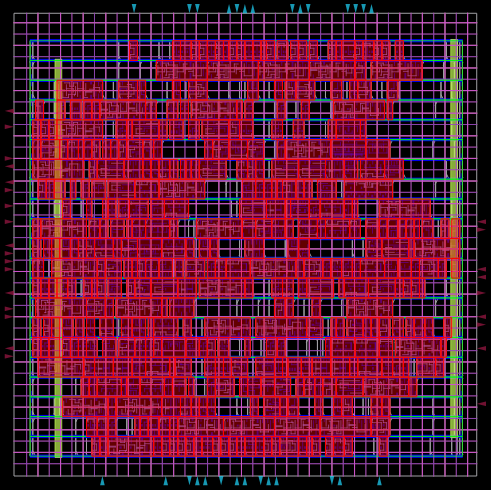
* 1. Clock Tree Synthesis(CTS)：設定路徑把clock信號傳遞給所有flip flop
     1. CTS (確定clock連接點的位置，但在這一步中尚未加入filter)



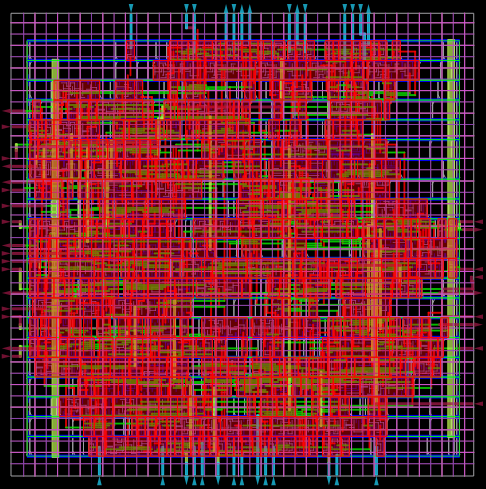
* + 1. CTS fill cell (使用filter填補還未安裝flip-flop的空位)



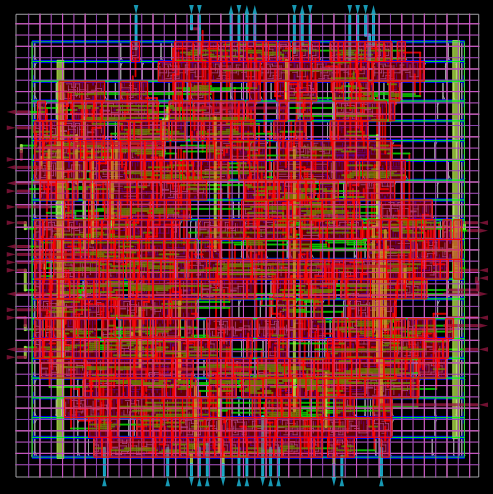
* 1. Routing：繞線
     1. Run global route



* + 1. Run detailed route



* 1. Finishing：完成設計



* 1. 參考資料

[[20231207] Digital System Design Flow (youtube.com)](https://www.youtube.com/watch?v=X7ybHCccvzA)

[10分鐘認識IC開發流程(IC Design Flow)｜方格子 vocus](https://vocus.cc/article/6381c54dfd8978000158baa9)

[PowerPoint 簡報 (nthu.edu.tw)](http://m105.nthu.edu.tw/~s105062901/ppt/RMaKnowsICDesignFlow.pdf)

[数字ASIC设计概要：Tap Cell - 非是非 (noyesno.net)](https://noyesno.net/page/asic/tap-cell.html)

<https://github.com/The-OpenROAD-Project/OpenROAD/blob/master/src/pdn/README.md>

[OpenROAD — OpenROAD documentation](https://openroad.readthedocs.io/en/latest/main/README.html)

1. 從 open-source IC design flow 的學習

我大概追蹤了一下流程，雖然沒有完全理解，但大概知道流程，從用寫verilog找最佳化、晶片布局、放入元件、設定clock、繞線到完成設計，慢慢地了解一點點IC設計的輪廓，不過由於沒有親自設計過，而且也沒接觸龐大的IC設計方法，仍有許多不了解的地方，期待未來若有機會，我能修課學會，並且能更看得懂這些流程。

1. 課程反饋

老師是我認為資工裡最認真教課的老師，除了會錄影給每個同學重複聽課的機會，對於課程的準備也很認真，而且可以藉由前三個Project，訓練程式能力，雖然我常常去找老師問問題，但老師還是不厭其煩的回答我的問題。

至於建議的部分，我希望能多一點socrative的練習題目，像是self-correct和counter divided by x等可以多幾題練習加強觀念；project3我曾經建議範例要修改，但後來我覺得過於麻煩，對於老師有點負擔，因此我認為至少劃出比較線，或是那一小段部分重錄會讓大家比較了解；在最後IC設計的流程，需要有實際的圖來說明會更好一點，只有流程圖，可能仍然不清楚，或是可以以比喻的方式來講解，像是有清大的學長以蓋房子的宏觀方式比喻IC的微觀設計。[PowerPoint 簡報 (nthu.edu.tw)](http://m105.nthu.edu.tw/~s105062901/ppt/RMaKnowsICDesignFlow.pdf)

最後，感謝老師這一年來的陪伴，讓我從連電路都不懂的高中生，學到了各種邏輯設計，希望未來有機會能修到老師的課。