FC ML/DSO.ai Macro Placement

Hoda Thabet

Infineon Technologies
Munich, Germany

ABSTRACT

This paper will discuss the evaluation of Synopsys Fusion Compiler ML/DSO.ai for achieving better PPA in SOC digital implementation through exploring its machine learning macro placement capabilities. The paper aims to leverage Fusion Compiler's machine learning auto macro placement feature to create the optimal macro placement for congestion, timing, and power.

Table of Contents

1. Introduction

In SOC digital implementation, designers face numerous challenges, such as meeting tight schedules, power budgets, and maintaining high performance. The evaluation of Fusion Compiler ML/DSO.ai's machine learning auto macro placement feature aims to address these challenges by optimizing PPA. The evaluation will focus on exploring key features and benefits of the tool, including its ability to enhance timing and congestion optimization through delivering the best floorplan.

The paper will discuss the evaluation process, including the methodology, tools used, and the results of the experiments. Attendees will gain insights into how machine learning can be leveraged to optimize macro placement in SOC digital implementation and how Fusion Compiler ML/DSO.ai can help achieve better PPA.

Overall, this paper aims to provide valuable insights for SOC digital designers looking to enhance their PPA optimization efforts by leveraging machine learning auto macro placement tools like Synopsys Fusion Compiler ML/DSO.ai.

2. Problem Statement

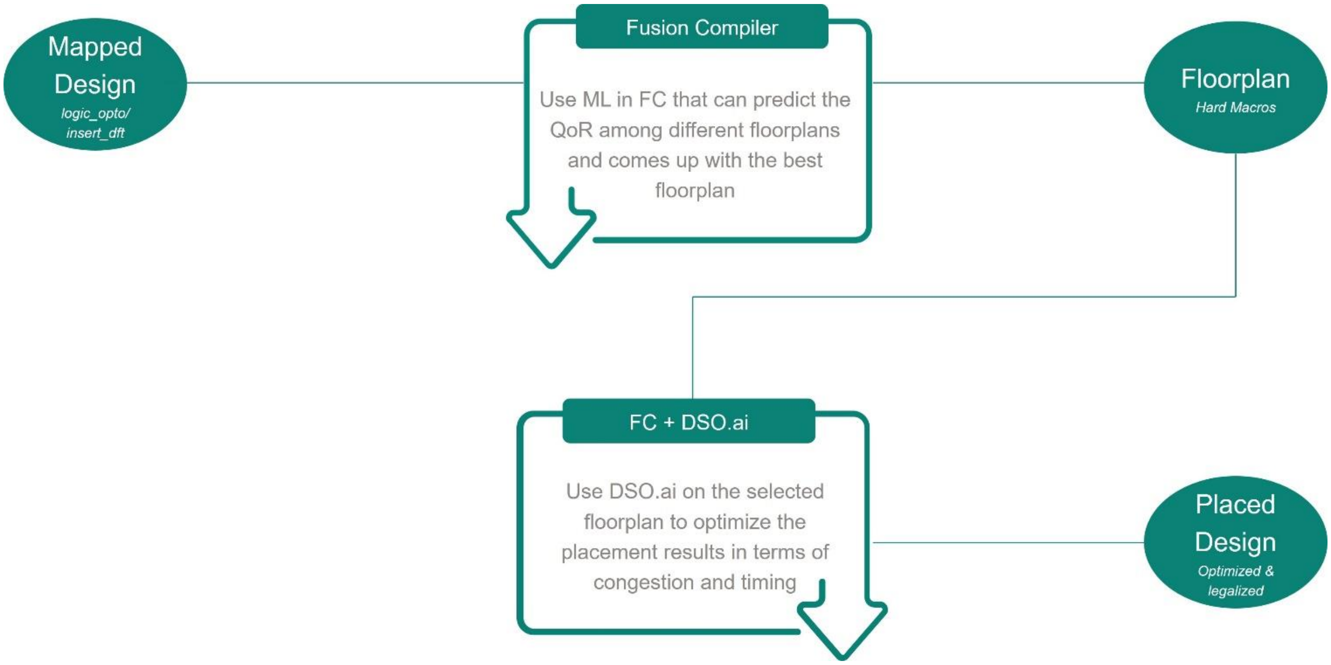
Floorplan and macro placement are critical steps in the design process of integrated circuits. They have a significant impact on the final product's performance, power consumption, and area. A well-optimized floorplan can provide better PPA (performance, power, and area) and reduce the overall cost of the product. However, creating a perfect floorplan takes time and effort. It can take several months to analyze and come up with a floorplan that provides the best PPA outcome. The process of designing the floorplan is often iterative, requiring several manual efforts to optimize the placement of macros. The good news is that modern design tools and algorithms can significantly reduce the manual effort and the iterative process required to find the best placement solution. These tools can help designers optimize their floorplans faster and with higher accuracy, leading to better PPA outcomes.

3. Methodology

Machine learning-based macro placement approach uses the data from previous macro placements to predict the QoR and applies machine learning to create floorplans that have the best macro placement.

When you use ML-based macro placement, the tool creates several floorplans. Each floorplan is referred to as one ML data. The trained model predicts congestion and TNS after post optimization. The tool creates the model from the ML data. The more data is available, the more accurate the model becomes.

Flow Overview


Mapped Design
logic_opto/ insert_dft
Fusion Compiler
Use ML in FC that can predict the QoR among different floorplans and comes up with the best floorplan
Floorplan
Hard Macros
FC + DSO.ai
Use DSO.ai on the selected floorplan to optimize the placement results in terms of congestion and timing
Placed Design
Optimized & legalized

How?

The ML floorplanning step can be run using one of the following commands. Each of these commands have the options to select the optimization objective, the runtime effort, and the macro placement style.

Command:

- place_macro_ml
- explore_macro_placement

Options:

- Mode : congestion | tns | power | both | wirelength | all
- Style : hybrid | on_edge | freeform | auto
- Effort : high | medium | low

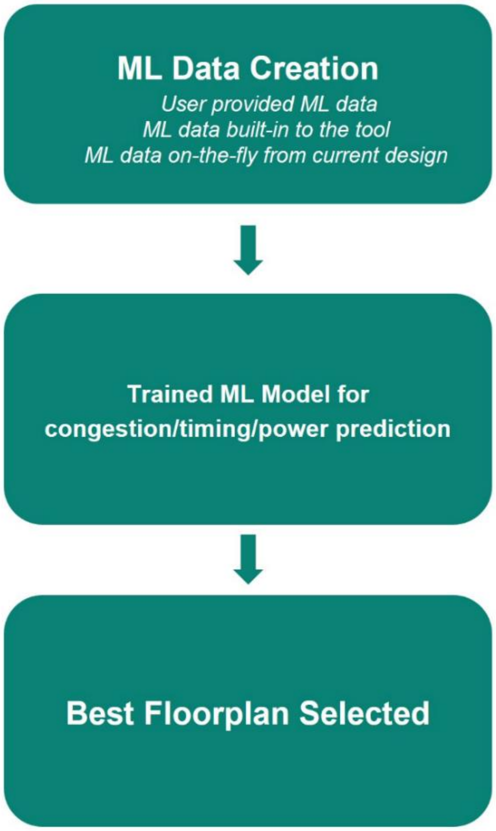
What Happens?

The tool explores possible macro placement results and create numerous floorplans. The tool then creates a trained model to predict the QoR among the created floorplans and comes up with the best floorplan.

The ML data is the used by the tool to build a trained model. The tool will use one or all of the following data to build the trained model for congestion, timing and power prediction.

- User Provided ML data
- ML data built-in to the tool
- ML data on-the-fly from current design.

The more ML data is made available, the more accurate the trained model will be.


ML Data Creation

User provided ML data
ML data built-in to the tool
ML data on-the-fly from current design

Trained ML Model for congestion/timing/power prediction

Best Floorplan Selected

Outputs

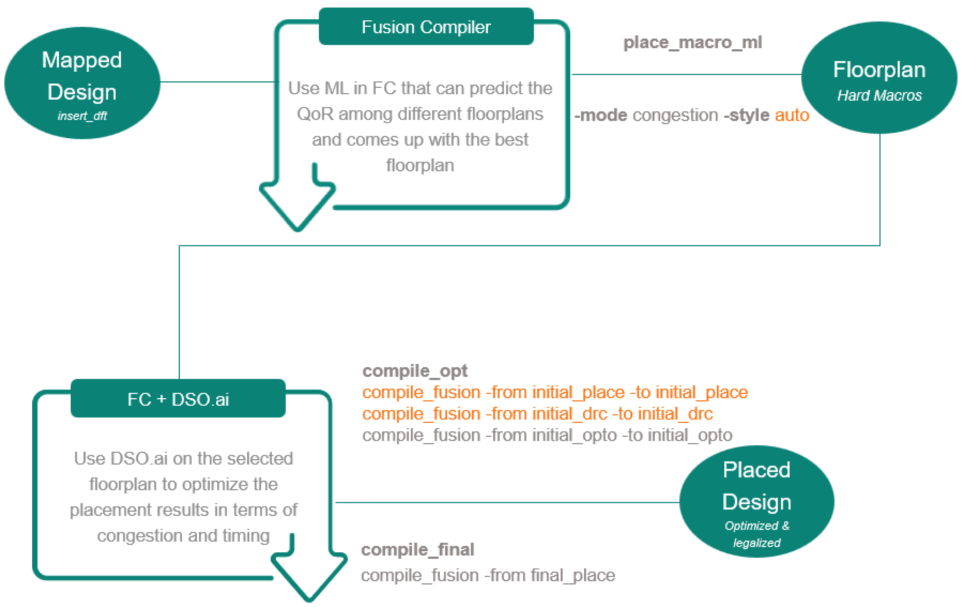
The outputs of the ML-macro placement step are as follows:

- **Floorplan**: It is saved in the output database (NDM). A DEF file also written by the tool. Snapshots for routing congestion map are output by the tool as well.

- **DSO.ai Permutons**: This is written by the tool in case of using ***explore_macro_placement*** command only.

- **ML Trained Data**: Contains multiple ML data. Each ML data corresponds to one explored floorplan. This data can be fed back to the tool in case of an incremental/more trained run.

4. Explored Flow



Use ML in FC that can predict the QoR among different floorplans and comes up with the best floorplan

```
place_macro_ml
-mode congestion -style auto
```

Use DSO.ai on the selected floorplan to optimize the placement results in terms of congestion and timing

```
compile_opt
compile_fusion -from initial_place -to initial_place
compile_fusion -from initial_drc -to initial_drc
compile_fusion -from initial_opto -to initial_opto

compile_final
compile_fusion -from final_place
```

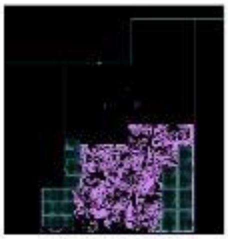
- Run the compile_fusion -to logic_opto command.
- Perform ML-based macro placement.
- If you used freeform or hybrid macro placement, continue with compile_fusion -from initial_drc. Otherwise, use compile_fusion -from initial_place.
- DSO.ai is then used starting compile_opt stage to optimize the standard cells placement in terms of congestion and timing.

5. Results

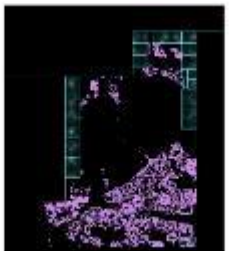
ML-macro placement has shown very good results with complete change in the macro placement compared to the original floorplan. The following figures show the cell density and global routing congestion maps of the best floorplan selected by the tool for the design under implementation. The QoR results reported are after the compile_final stage and are using custom defined congestion permutons.

Floorplan

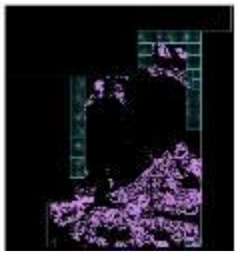
Baseline



FC ML + DSO.ai
Starting initial_place



FC ML + DSO.ai
Starting initial_opto

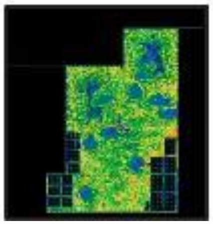


QoR

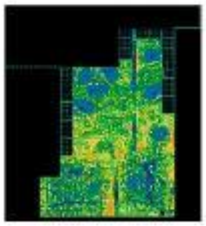
Run	ADES Improvement	R2R WNS Improvement	R2R TNS Improvement	CONGESTION Improvement
FC ML + DSO.ai *starting initial place*	60%	70%	80%	65%
FC ML + DSO.ai *starting initial opto*	50%	60%	70%	60%

Cell Density

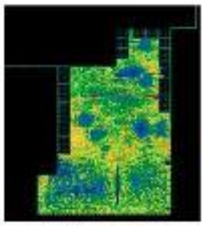
Baseline



FC ML + DSO.ai
Starting initial_place

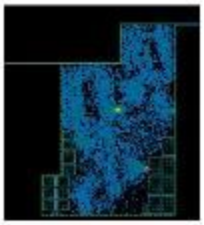


FC ML + DSO.ai
Starting initial_opto

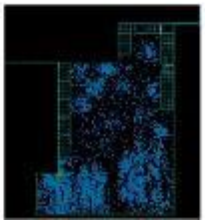


Congestion

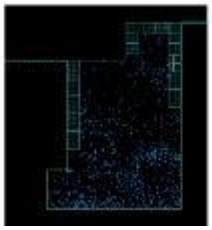
Baseline



FC ML + DSO.ai
Starting initial_place



FC ML + DSO.ai
Starting initial_opto



6. Conclusions

ML floorplan exploration can be used as a guidance on an initial floorplan. This guidance can provide a better congestion and timing results after full placement phase.

There are more things to explore for this flow that are under implementation.

- Run the full flow till route on the ML floorplan to verify that the design is routable.
- Involve DSO.ai in the ML Fusion Compiler floorplanning step to explore other ***mode*** & **s*tyle*** options.

7. References

[1] Fusion Compiler Design Planning User Guide