SAMSUNG

Scenario-based Mixed Signal Layout Generator using Generation APIs for Memory

Jeongyoon Lee, Kyeongrok Jo, Seungkwang Hong, Sichan Kim, Seunghwan Lee, Youngwook Kim, and Jungyun Choi

Samsung Electronics, Memory Business

62 THE CHIPS TO SYSTEMS CONFERENCE

SPONSORED BY CEDA® IEEE Council on Electronic Design Automation sigda acm

Motivation

- **Needs for Scenario-based Mixed Signal Layout Generator for Memory Analog IP**
  - The turn-around-time (TAT) for analog IP layout design significantly exceeds that for digital, despite their low quantity (Fig. 1 (a), (b)).
  - Although automated migration and generation solutions for analog IP have been proposed recently, their practical implementation remains challenging.
    - Parts of the schematic may be modified, or a new schematic can be added, which leads to the previous layout information no longer being applicable.
    - Even when the schematic is reused, layout reusability is often hindered by changes in design rule (DR), boundary condition, etc. (Fig. 2).
    - In the case of layout generation solutions, there is an issue of unpredictability regarding how much time it will take to update the layout based on the changes.
  - Transitioning to a new process requires layout exploration to select the optimal layout based on area and characteristics, but drawing multiple layouts manually takes excessively long.

➔ **A solution is needed to quickly generate new layout in response to changes.**

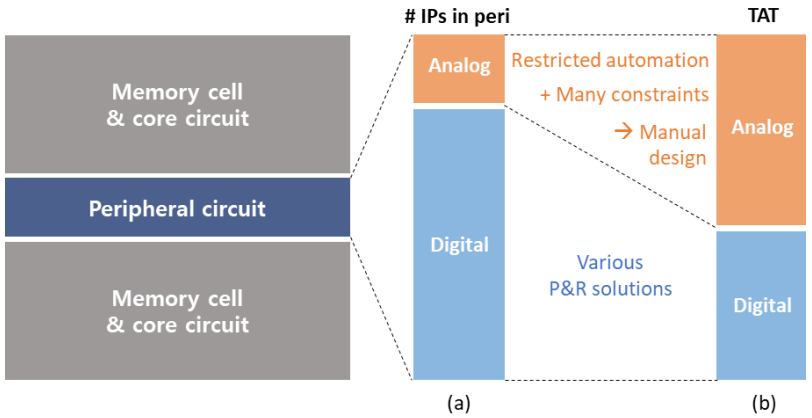
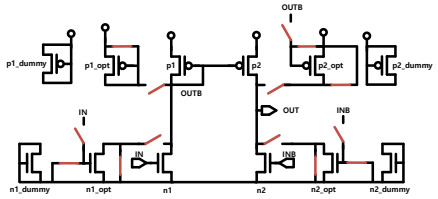


Fig. 1. Composition in peripheral circuit for memory.

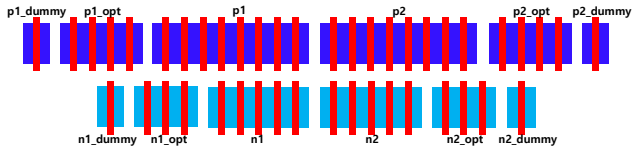
Fig. 2. Layout changes in a IP with the same schematic.
Different color represents the floorplan area for different type of components.*

Motivation (cont.)

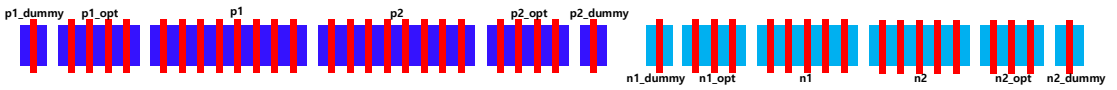
- **Advantages of Proposed Methods**
  - Layouts migrated from the previous one can be generated in real time.
  - When changes to the layout pattern are required, new layouts can be generated with simple modifications through the UI without the need to directly modify the generation code.
  - Rapid layout modifications enable layout exploration, allowing engineers to compare areas, characteristics, and other factors to make better decisions (Fig. 3).



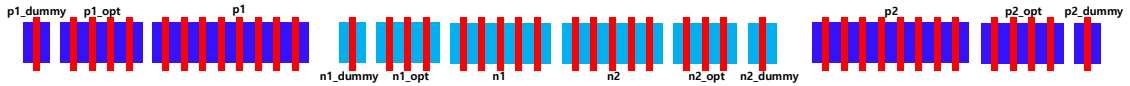
(a) Schematic



(b) Layout pattern type 1 (PN vertical)



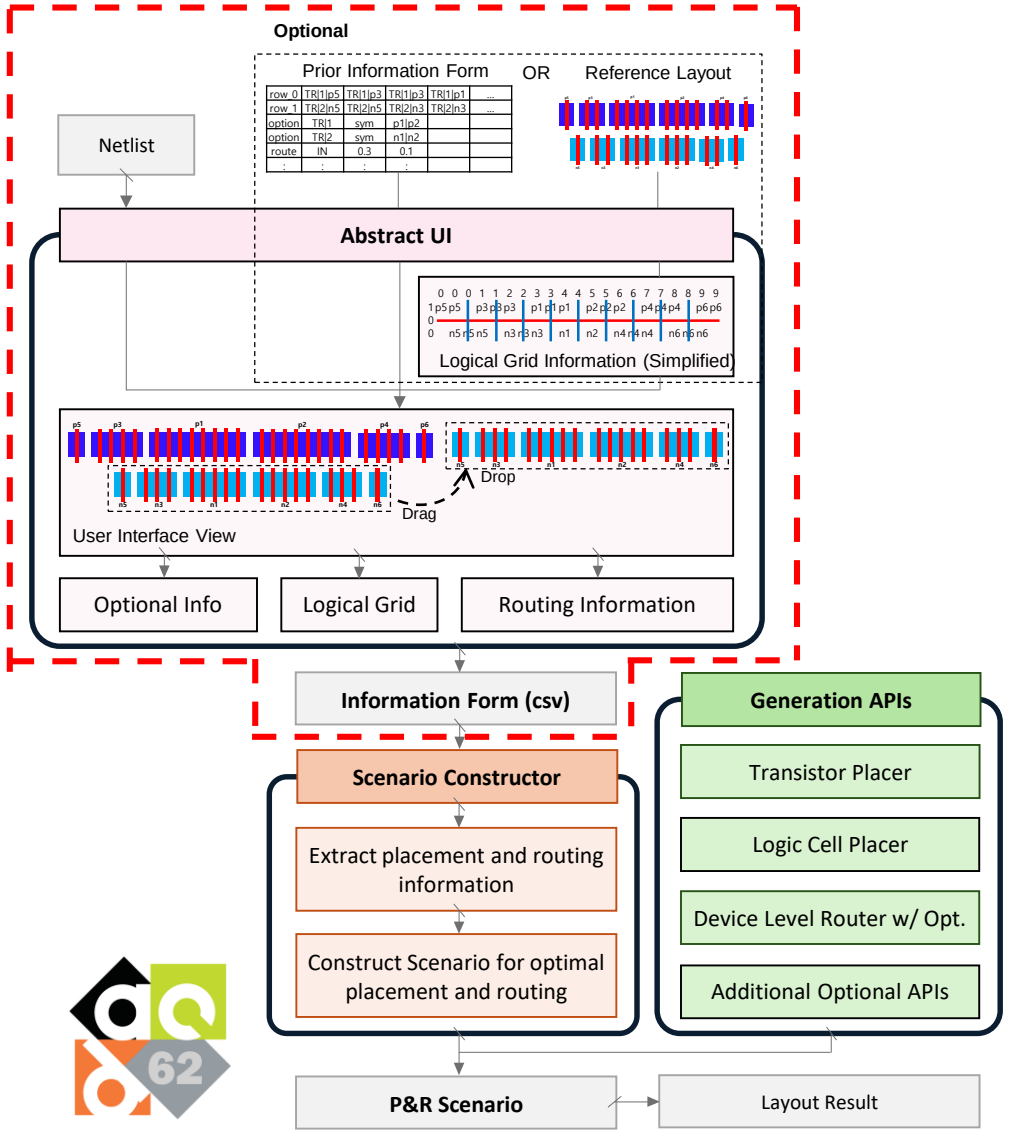
(c) Layout pattern type 2 (PN horizontal)



(d) Layout pattern type 3 (PNP symmetric)

Fig. 3. Layout changes in a IP with the same Differential Amplifier schematic with option.

Proposed Flow



- **Abstract UI (User Interface)**
  - It receives a netlist and displays initial placement results, considering routing, in a layout similar to the actual one.
  - When a reference layout or prior information form is provided, it extracts logical grid and adjusts it to match the netlist for display.
  - Engineers can easily modify the placement or routing to match their desired layout.
  - Information Form is generated from the abstract layout in UI, which contains the placement logical grid, routing information, and additional optional information(e.g. symmetry, alignment, etc.).
- **Information Form (csv)**
  - The information of each devices is stored with distinctions based on category*, index**, and instance name.
  - The routing information from the UI is converted into a width that reflects the position and non-default routing (NDR), then applied.

Fig. 4. Overall flow of Scenario-based Layout Generator

Proposed Flow (cont.)

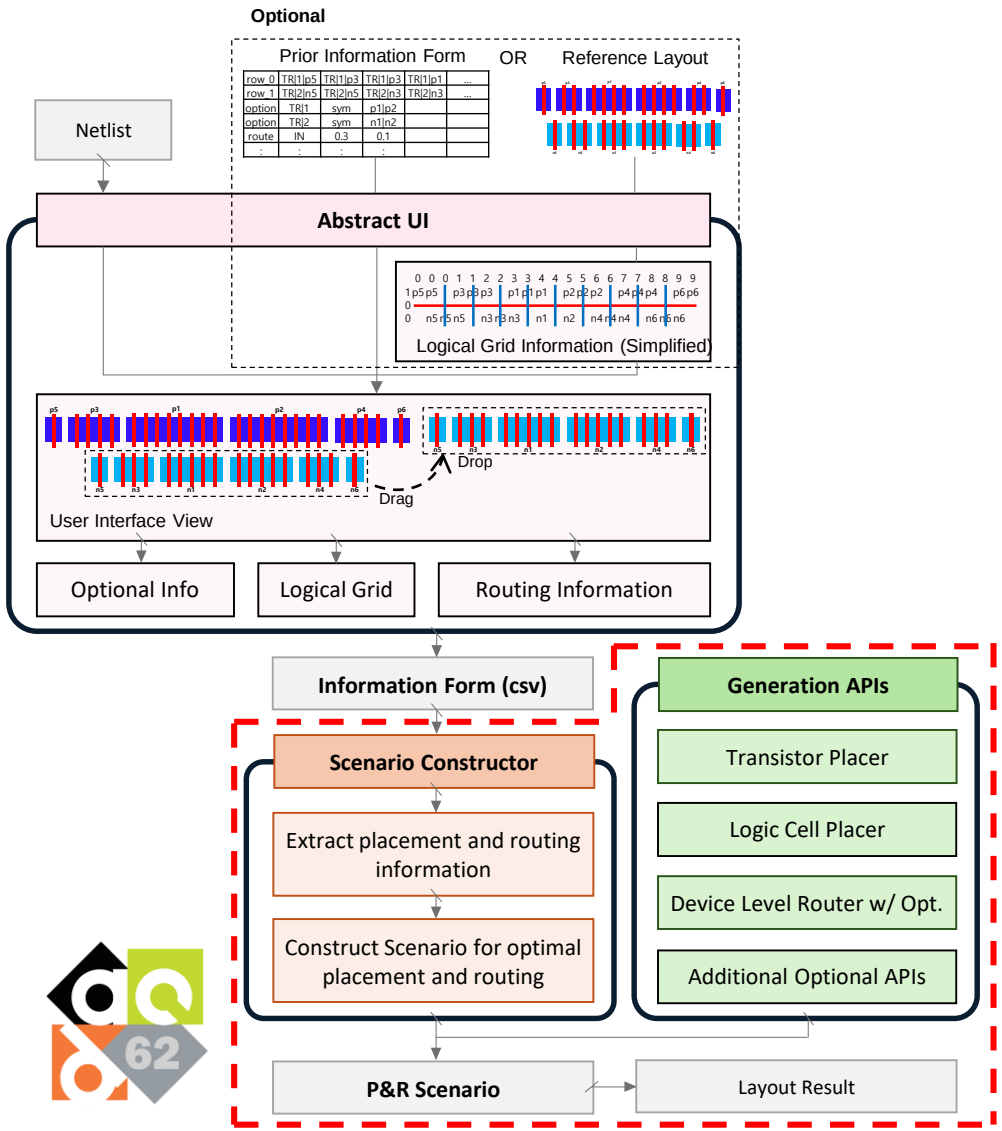


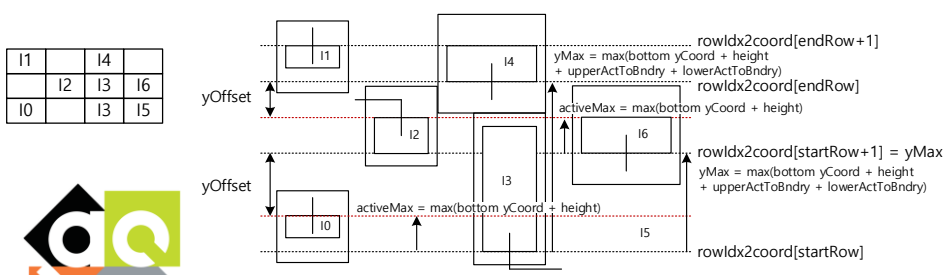
Fig. 4. Overall flow of Scenario-based Layout Generator

- **Scenario Constructor**
  - The placement and routing information of devices are extracted from the Information Form.
  - A Scenario is constructed, which means organizing the sequence of placement and routing (P&R) while considering the dependencies between individual elements(devices to be placed or nets to be routed), to achieve the desired layout.

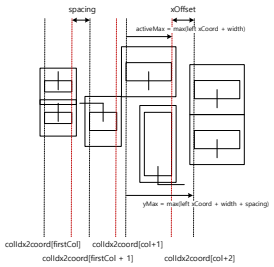
- **P&R Scenario**
  - Placement and routing are performed according to the given scenario sequence, utilizing APIs such as the Transistor Placer, Logic Cell*** Placer, Device-Level Router with Option Layer, and Additional Optional APIs.
  - This process enables the creation of new layouts without modifying the actual code.
  - Each API within the Generation APIs independently and fully performs its designated tasks.

Detailed Information of Generation APIs

- **Generation APIs**
  - **The Transistor Placer** receives logical grid information which contains the relative relationship of the transistors and converts it into physical placement.
    - It minimizes the area by 1) finding optimal spacing between transistors, maintaining relative position and considering DRs, and 2) sharing the diffusion with the same net.
    - It allows for various output to be generated for the same transistor, depending on the options provided.
      - e.g. diffusion sharing, using antenna, symmetric constraints and etc.
  - **The Logic Cell Placer** arranges logic cells with the same index within the same row
    - Avoid blocked ports and minimize the use of routing resources.
  - **The Device-Level Router** performs routing to the desired tracks, including NDR, shielding, and **optional routing**, ensuring the shortest possible path.
    - It is designed to prioritize the use of specified tracks.
      - Routing tracks need to be specified for effective connections with the upper hierarchy or neighboring layouts, as well as for characteristic-specific requirements.
    - It is also designed to support various optional routing tasks for ECO purposes.
  - **The Additional Optional APIs** include handling other categories such as guardring and another layout structures, as well as post-processes for tasks like Design Rule Check.



(a) Vertical direction



(b) Horizontal direction

Fig. 5. Conversion logic of grid for each direction in transistor placement

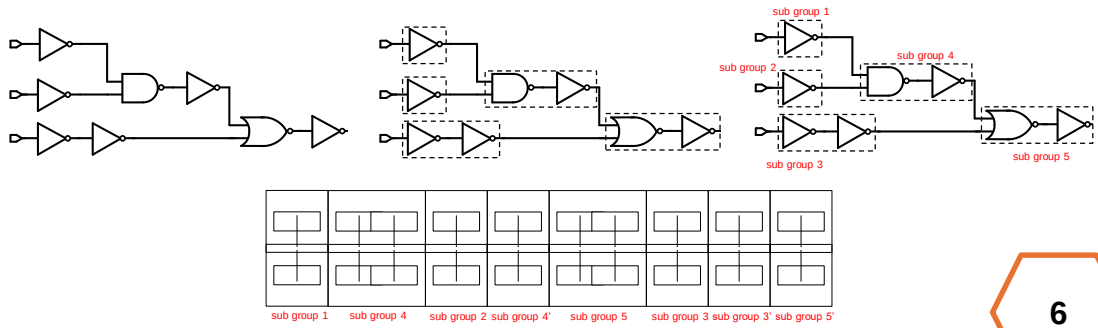


Fig. 6. Logic Cell placement

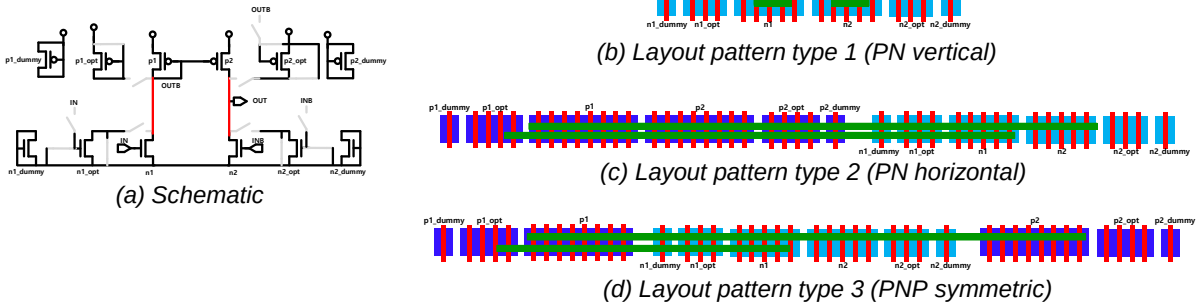
Environment & TAT Comparison

- Run on Linux machine with Intel Xeon® Gold 6334 CPU
- Implemented with Python and Skill Language
- Use two IPs for the experiment (Table 1)
- The Information Form used for measuring the runtime of the P&R Scenario was extracted from the reference layout.
- It has been observed that the TAT for manual layout processes can be reduced by approximately 99.65% for Latch-up Prevention Circuit and 99.88% for Differential Amplifier.

IP	TAT				
	Manual Drawing	Manual Drawing + Generation APIs Only		Runtime of P&R Scenario	
Latch-up Prevention Circuit	2h	1h	-50%	25s	-99.65%
Differential Amplifier	8h	1h	-87.5%	34s	-99.88%

Table 1. TAT comparison

Experimental Results



(a) Schematic

(b) Layout pattern type 1 (PN vertical)

(c) Layout pattern type 2 (PN horizontal)

(d) Layout pattern type 3 (PNP symmetric)

Fig. 6. Layout exploration results of Differential Amplifier

Differential Amplifier	Area(um^2)	Symmetry	Critical Net WL (um)	Total WL (um)	Modifying Time
Type 1 (PN Vertical)	100.869	O	37.3787	90.9811	Ref.
Type 2 (PN Horizontal)	89.1822	X	80.8479	156.5476	$<$ 10 min.
Type 3 (PNP Symmetric)	92.4111	O	66.5035	141.9633	$<$ 10 min.

Table 2. Comparison between layout pattern types

- Red lines in Fig. 6. (a) and green lines in (b) ~ (d) represent critical nets in Differential Amplifier.
- Using the same schematic, three types of layouts (Fig. 6. (b) ~ (d)) were generated by making simple modifications to the transistor placement through the UI.
- Modifying Abstract UI for the placement and routing takes less than 10 minutes.
- From an area perspective, Type 2 is optimal; however, when considering symmetry along with a combination of area and net wire length, Type 1 or Type 3 can be selectively the optimal solution depending on specific cases (Table 2).

Summary

- A **Scenario-Based Mixed Signal Layout Generator** was developed to significantly reduce layout TAT and enable efficient pattern exploration.
- The approach supports rapid pattern adjustments, allowing engineers to interact with the tool and generate layouts in their desired patterns.
- It is capable of handling complex patterns, including those utilizing optional routing and advanced layout configurations.
- By simply inputting a netlist, the tool analyzes the schematic to generate new layout patterns while ensuring compliance with analog layout constraints.
- This automation reduces manual effort for engineers, allowing them to focus on high-level design optimizations.
- **Contribution**
  - **Enhanced productivity**: Demonstrated substantial reductions in layout TAT.
  - **Reduced complexity**: Simplified adjustments for new design constraints or circuit modifications.
  - **Improved quality**: Facilitated exploration of multiple patterns, ensuring better silicon performance.
- **Future Work**
  - While the proposed work assists engineers in generating desired patterns and supports exploration, future developments aim to automate the exploration process entirely, including characteristic comparisons(e.g. post-layout simulation), to further enhance efficiency and accuracy.