Clock and Power Supply-Aware High Accuracy Phase Interpolator Layout Synthesis

Siou-Sian Lin, Shih-Yu Chen, Yu-Ping Huang, Tzu-Chuan Lin, Hung-Ming Chen and Wei-Zen Chen

Inst. of Electronics, National Yang Ming Chiao Tung University, Hsinchu City, Taiwan
zu00895077@gmail.com,jasonchen811221@gmail.com,nucweacia94fine@gmail.com,
reg8982@gmail.com,hmchen@nycu.edu.tw,wzchen@nycu.edu.tw

Abstract—Due to popular requests from the designers of clock and data recovery (CDR) regarding the inefficiency of generating high accuracy phase interpolator (PI), in this work, we have developed a layout generator for such circuit, different from conventional constraint-driven works. In the first stage, we propose a customized template floorplanning plus pin generation demanded by the users. In the second stage, in order to generate high accuracy layout, we implement a gridless router for signal, power supply and clock. Experiments with several configurations indicate that our approach can generate high-quality corresponding layouts that align with user expectations, and even surpass the quality of manual designs on structurally regular high-performance PIs, which are not easy and efficient to be generated by prior primitive/grid-based methods.

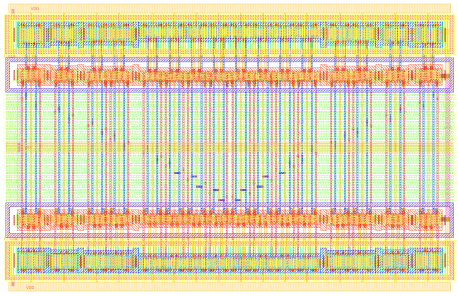


Fig. 1: The illustrated layout showing a manually designed 7-bit Inverter-based PI, fraught with complex constraints and typically demanding two full workdays to complete.

I. Introduction

Analog/mixed signal (AMS) circuit layout synthesis/automation has been studied for a long while, many research works and commercial product attempts were presented. To be more specific, extensive analog layout works have been conducted, such as ALIGN[1], MAGICAL[2] and BAG2[3], to name a few. In [1], parameterized cells and primitives were presented, but it could only synthesize low-frequency analog blocks. In [2] and recent adoption [4], they showed automated symmetry extraction in commonly-used analog blocks such opamp and OTAs. [3] among these three packages claimed to be a process portable generator. Based on our study, these approaches are insufficient in assisting the generation of effective layouts for high-performance circuits as they fail to meet the stringent requirements these circuits impose on their layouts. A high-frequency phase interpolator (PI) exemplifies such a high-performance circuit. The manual layout in Fig. 1, though simple, has strict placement and routing constraints due to its performance needs. Notably, this circuit includes 256 transistors, each with specific positioning and strict routing rules.

Besides aforementioned prior arts, analog layout automation was currently characterized in three categories: generation-based, migration-based and template-based approaches, mentioned in [5]. The generation-based method among the three is hampered by the high cost of algorithm design and a limited optimization space, constraining its practical application. While the migration-based method can directly transform layout styles and reduce development effort, it does not guarantee that all constraints are satisfied after migration. Furthermore, when the structure of the original circuit changes, the migration-based method may become ineffective, thus lacking flexibility. Therefore, we have chosen to employ a template-based synthesis method, which avoids the constraint-driven natures in generation- and migration-based approaches. This approach leverages a layout template encompassing all necessary information for circuit generation, including the placement of components and various routing details. It also incorporates the advantages of both generation and migration methods, boasting superior flexibility and efficiency.

In past research on analog circuits, many template-based synthesis methods have been proposed [6], [7]. Depending on the level of hierarchy, the description of the template can be divided into block level and circuit level. In order to preserve the freedom and richness of the templates, our study focuses on the circuit-level description to showcase various component placement styles. In high-performance circuits, MOS diffusion sharing and guard rings are of great importance. To generate practical circuits, we also take these factors into consideration.

In terms of routing, previous template-based methods primarily relied on constraints to guide routing behavior, which is insufficient to accurately describe routing patterns and may generate routing results that are inconsistent with expectations, especially special requirements from clock and power. To solve this problem, we propose a new gridless routing system that can precisely guide the detailed routing behavior by designers.

In all, our novel treatment starts with the layout synthesis of high accuracy PI, aligning accurately with layout designer expectations efficiently, adheres to the requirements of high-performance circuits, and accommodates designer preferences. We have the following contributions:

1) Our approach allows designers to customize layout templates for high-frequency phase interpolators planning,

978-3-9826741-0-0/DATE25/© 2025 EDAA

at the same time considering MOS diffusion sharing, guard ring generation, via generation and component pin creation.

2) Since routing of high-frequency circuits is depending on special treatment of clock and power supply routing, our router is implemented as gridless style in order to take care of non-uniform grid issues and to avoid redundant routes and dead spaces.
3) To accelerate our gridless routing, we further develop a multi-point-to-multi-point tile router. And to meet the unique requirements of power and clock nets, we develop "parallel" power routing and H-like clock routing, ensuring high-quality routing for high frequency nature of phase interpolator.

The remaining sections are organized as follows: our flow and methodology are depicted in Section II, while the process of the customized template generation is elaborated in Section III. Section IV motivates the usage of gridless routing for PI layout synthesis, and Section V presents our layout results on some performance indicators. Finally, a summary and conclusion of our work are shown in Section VI.

II. Algorithm Overview

We first illustrate the automatic flow integrating our work with Virtuoso. We utilize the Skill language to instruct Virtuoso for streaming out the netlist file (schematic) of the target circuit and the automatically constructed component entity information (GDS). This necessary information for constructing the layout is transferred to our work, and upon the completion of the layout synthesis, it is imported back into Virtuoso. Throughout the process, we use open-source tools to handle the data conversion between GDS and GDT (graphics data text) files.

Fig. 2 presents the overall flow of our work. The layout specification file is a custom document where users must provide detailed descriptions of the topological information between components in the circuit and various routing-related information to guide the layout process. By custom template generation, we can generate elements placement and pre-routing preparation. Via gridless power and clock routing, we can take care of specialized requests to accomplish high frequency and high accuracy PI layout synthesis. Without complicated constraints to drive layout generation, we are able to obtain good quality designs by customization.

III. Customized Template Generation

Using the information from the layout template, this section arranges the layout, implements MOS diffusion sharing, and generates guard rings. To ensure a higher-quality and accurate layout, we opt for an adjustable template over an area-optimized one, catering to designers' requirements. This approach also allows us to generate routing pin points for each MOS, completing all pre-routing preparations.

A. Placing Elements

This subsection aims to integrate automatically generated components (elements/primitives) from Virtuoso with layout template information, completing the placement of components and the setup of pins as well as pre-routing for PI layouts. The layout template information is represented as a

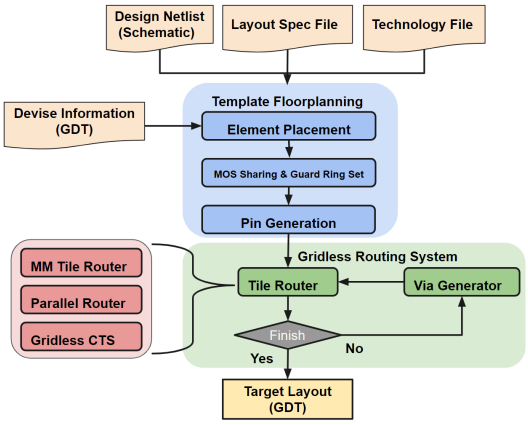


Fig. 2: Overall flow chart of our work.

two-dimensional string array, there are three types of elements when presenting physical space information: instances, spaces, and boxes. Boxes can be assigned to any layer, and when provided with additional information, they can form pins, pre-routes, or via-stacks, among others.

Based on the layout template information, elements are placed row by row from bottom to top. Each row contains an arbitrary number of varied elements. During row construction, we place elements from left to right. Once the placement of a row is complete, we have the freedom to decide whether to update the current y-coordinate. This mechanism ensures that any type of layout can be constructed.

Among the three types of elements, the size of an instance is fixed (as described by the .gdt file), while the sizes of spaces and boxes can be adjusted freely. We can either directly specify the coordinates of the box or use the coordinate information of other components to guide it. This mechanism ensures that, even after circuit sizing, the template information can still effectively guide the placement of the circuit. Fig. 3 shows an example of a 7-bit PI with this step applied.

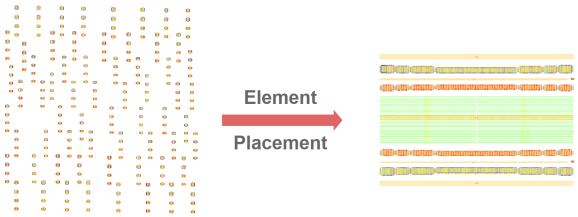


Fig. 3: The left part shows the device layout auto-generated with netlist in Virtuoso. The placement result is shown on the right.

B. MOS Diffusion Sharing and Guard Ring Generation

In practical layout design, especially for high-performance circuits, MOS diffusion sharing and guard ring generation are of paramount importance. However, these aspects have often been overlooked in previous research, limiting the practicality of the generated layouts in actual applications. In this work, we carefully address these features, each with its advantages and disadvantages. This allows users to evaluate and decide on their inclusion in the layout template.

Successful merging conditions are twofold: (1) the adjacent drain or source of the two has a connectedline on the netlist, and (2) both MOSFETs share the same type and transistor width. Regarding guard rings, distinct types for PMOS and NMOS are generated based on MOS position, ensuring Design Rule Check (DRC) compliance by adjusting related elements. Fig. 4 showcases two 7-bit Phase Interpolator (PI) layouts with guard rings: in Fig. 4(a), MOS sharing is not utilized, while in Fig. 4(b), MOS sharing is employed, leading to over 2 times reduction in area.

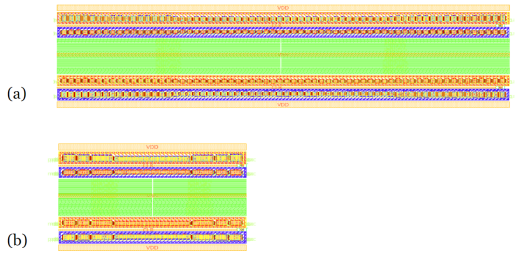


Fig. 4: 7-bit PI diffusion sharing layouts (with guard rings): (a) is without diffusion sharing and (b) activates diffusion sharing. The discrepancy in both area and critical net between the two exceeds twofold.

C. MOS Pin Generation

Unlike digital circuits whose basic elements are standard cells, analog circuits consist of MOS, capacitors and resistors. Despite their simple structure, ensuring successful pin access of these base components remains a crucial issue. For multi-finger MOS, it should be noted that although the MOS has only one point at each of the D, G, and S terminals in the abstract circuit, there may be multiple locations in the physical structure. Even though they are equivalent endpoints, they are not connected initially. At this stage, we need to merge the discrete MOS terminals into a single pin point to ensure connection accuracy and significantly reduce the complexity of subsequent routing.

From the perspective of the entire layout size, the impact of different MOS pin generation positions on routing length estimation seems negligible. But in real layout connection, the position of the MOS pin has a significant effect on routing behavior. The routing behavior may impact the routability and coupling effects. For high-performance circuits, unexpected routing can affect circuit performance, which is unacceptable. Therefore, the position of the pin needs precise control. We develop a method that meets both key considerations while following design rules. Initially, we create metal segments for each terminal from the user-defined layout spec file. Users/Designers can specify positions and routing layers, forming routing pin points. For multi-finger MOS types, merging of equivalent pin points is also performed at this stage. This planning will be used in the later routing stage.

IV. Comprehensive Gridless Routing System for PI Layout Synthesis

There have been numerous studies related to analog routing in the past. Basically all of these routers are designed based on a grid-based approach, one excellent example is [8], [3]. However, we learn from the designers that results of grid-based approach may limit the precision of routing and thus produce some negative effects: (1) Low resolution in grid-based router could lead to extra routing; (2) Dead space may occur during routing grid construction; (3) Management of various line widths and spacing cannot be performed. These factors potentially affect the routing outcome, leading to a decline in circuit performance. In contrast, gridless routers are not constrained by the grid and resolution, allowing for more flexible and precise reproduction of the anticipated routing behavior by users.

A. Routing System Overview

There have been some outstanding research studies on gridless routing system [9], [10] in the past, but they belong to the category of multi-layer routers. While multi-layer routers, with their comprehensive assessment mechanisms, may find shorter wire lengths than single-layer routers, this complexity makes the routing process unpredictable. This unpredictability does not enable users to anticipate and adjust routing behavior, contradicting the initial motivation of this work. To resolve this, we propose a single-layer based router and a via generator. By iteratively executing them, multi-layer routing can be accomplished, making the routing behavior simpler and more predictable.

In our gridless routing system, we start from the lowest metal layer. After the routing of a certain layer is complete, the via generator is invoked to generate corresponding via for pins that have not yet been routed. Then, corresponding pins are generated on the metal layer above the via. These operations are performed layer by layer. The entire process continues until all nets, including power supply and clock, are connected.

B. Multi-Source Multi-Target Tile Router

Our gridless router is implemented based on the corner stitch structure [11] as a tile router. We adopt the contour structure [12] proposed by Dion to ensure that the routing uses the correct width and that the routing space adheres to the design rule. [12] also proposed a point-to-point tile router (PP tile router), whose routing algorithm can be divided into two stages. The first stage is tile propagation, where a predetermined cost function (combined with the A* search algorithm) is used to find all feasible paths between adjacent space tiles to obtain the optimal space tile list. The second stage is path backtrace, where from the terminal of the best space tile list obtained in the first stage, the legal routing path is continuously backtracked and constructed until the list is fully traversed.

Dion's search method can efficiently find the least cost routing, but it only finds the optimal path between two tiles. According to the single-layer gridless routing structure we introduced above, what we hope to find is the shortest path between two pins. [12] and subsequent tile-router research did not mention how to solve the routing problem between multi-points. Since finding the shortest path between multiple points is an NP-complete problem, we compromise by dividing the pins into one source pin and the rest as target pins. A usual greedy method is to find the shortest path between the source pin and target pins each time, and after successful connection, merge the routing path and target pins into the source pin to

increase the routing search space, and improve routing quality. As shown in Fig. 5(a), the point-to-point tile router can ensure the minimum path between T1 (target pin-1) and S1-2 (tile-2 of source pin-1), but to confirm the shortest path between S1 and T1, PP tile router must be executed 12 (4 tiles in source x 3 tiles in target) times.

We then propose a multi-point-to-multi-point tile router (MM tile router). As shown in Fig. 5(b), it allows each tile in the source pin to perform path propagation simultaneously to achieve multi-source search capability. The predictive value in the original A* algorithm is calculated based on a single target tile. Here we modify to the following: for each individual path node, evaluate the estimated cost with the target tile closest to it. This modification can solve the problem that the A* search has only a single target. Although the number of initial path nodes will increase, and the prediction cost evaluation will take a little longer, the benefits of the A* search can still be retained when searching for the shortest path between source tiles and target tiles. Combined with the mechanisms of pruning paths and priority queues, the overall search time will not be much longer than a single PP tile router. The MM tile router can greatly improve the routing speed while maintaining the routing quality.

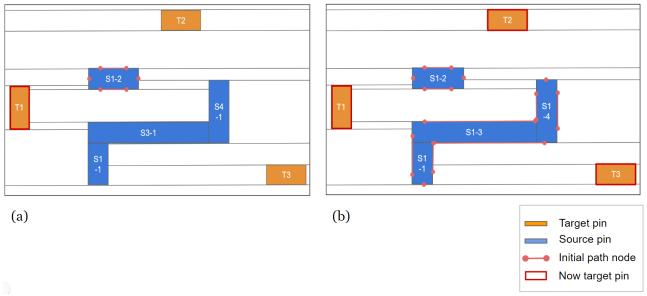


Fig. 5: (a) Point-to-point (PP) tile routing seeks the optimal path between single source tile and single target tile, while (b) our multi-point-to-multi-point (MM) tile routing can find the best path among source pin and other target pins.

C. Via Generaton

On cross-layer routing, the position of a via has a significant impact on the pin position of the upper layer routing. To make the routing behavior predictable and easy to adjust, we have designed a simple via generator. After completing single-layer routing, we process each route following the steps below:

1) If some route has completed the connection or there is a metal channel spanning multiple metal layers on its pin point (which can elevate the current routing to a higher level, avoiding the consumption of intermediate layer resources), there is no need to place a via.
2) According to the design rules of the via, considering the minimum size of the via and the enclosure specifications between this layer and the upper layer metal, a sandwich structure via stack is generated. The problem is transformed to how we place the via stack, simplifying the minimum size and metal enclosure issues.
3) We use the structure from [12] to find region that satisfy LVS and DRC. The LVS region refers to the area that can be connected to the pin of this layer's routing, while the DRC region refers to the area where the metal of this layer, via, and upper layer metal all meet the spacing rules after placing the via stack.
4) In the legal region, we find the preferred region to place vias, the regions are sorted according to the following two rules. Firstly, choose the legal region overlapping with the same routing area on the upper layer. Secondly, choose the legal position closest to the routing position on the upper layer. This step allows users to connect cross-layer routing with minimal cost through the upper layer's pre-routing guidance, avoiding generating unexpected additional routes.
5) Place the via stack on the preferred region and add the corresponding pin to the net on the upper layer routing layer.

D. Parallel Routing for Power Supply

Although the MM tile router ensures that the source pin can find the shortest distance target pin to merge with at each routing execution, this routing behavior may have adverse effects when routing power nets. Fig. 6 illustrates two routing patterns applied for the routing of the VSS net. Fig. 6(a) uses the MM tile router. The initial routing starts from the source pin and connects to one of the target pins above and below of source pin. Then the routing continuously extends to the surrounding target pins based on the shortest distance. This leads to a "series" type of routing formed by all pins above and below of source pin, resulting in excessively high current density. Such a scenario could potentially result in reduced signal transmission speed or cause the net to burn out.

To disperse the current, we hope that all target pins in the routing have independent routing paths from the source pin, and these paths do not intersect with each other. We call this type of routing "parallel routing". We have then modified the rules of the original router. Divergent from the MM tile router's approach of constantly merging the target pin and the path into the source pin, parallel routing does not merge the target pin and path after each completed routing. Instead, they are set as obstacles. Although this will increase the total routing length, it ensures that each target pin has an independent path from the source pin, as shown in Fig. 6(b). This parallel routing can prevent the problem of excessive current density.

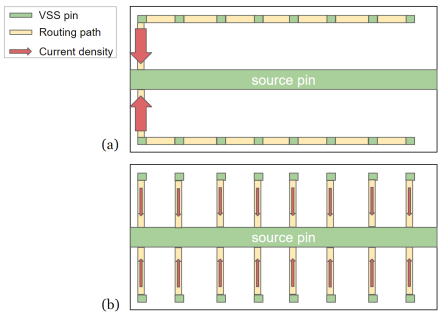


Fig. 6: Route VSS net by 2 methods: (a) MM tile routing and the (b) Parallel routing. Parallel routing can distribute current to prevent excessive current density on power nets.

E. Gridless Clock Tree Synthesis

In traditional clock routing algorithms, top-down based methods can control global information and yield shorter bus lengths. However, they tend to neglect sink-end information, leading to clock skew. Bottom-up methods, on the other hand, can reduce clock skew but may lack global information, leading to extra routing. When generating PI layouts with precise accuracy control, in response, we design a CTS tile router that integrates both top-down and bottom-up strategies to optimize the objective while complying with design constraints. We allow the users to control the spacing and maximum level of the clock tree.

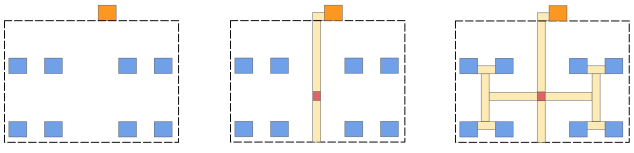


Fig. 7: Generating clock tree with gridless engine. (a) determines the domain of the clock routing region; (b) positions the root pin of the clock tree and completes routing between the root pin and source pin; (c) shows the result after executing those 4 steps.

Step 1: We identify sink pins in the clock net, i.e., those pins that need to satisfy clock routing constraints. These sink pins and the nearest source pin form a clock routing region, as shown in Fig. 7(a), which limits the routing range and reduces the time spent constructing the corner stitch layout.

Step 2: We identify the center of all sink pins and form a clock tree root pin. We complete routing from the clock tree root pin to the source pin on the boundary of the clock routing region. Furthermore, in order to satisfy the symmetry of the routing environment and reduce impedance mismatch, we need to extend this route as we need it, as illustrated in Fig. 7(b).

Step 3: We adopt and slightly modify the strategy in MMM algorithm [13]. We recursively partition the current layout in a top-down manner and obtain a clock (level) tree.

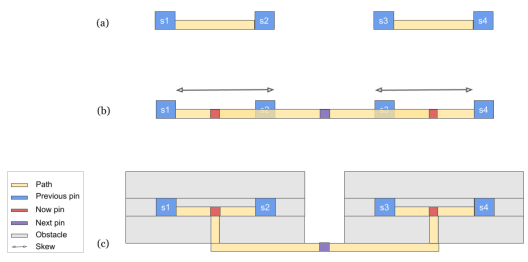


Fig. 8: (a) Starting point of our CTS; (b) result of Step 4 without obstacles, leading to clock skew due to lack of level separation. (c) Result of Step 4 with obstacles, preventing short circuits and ensuring spacing between different clock routes.

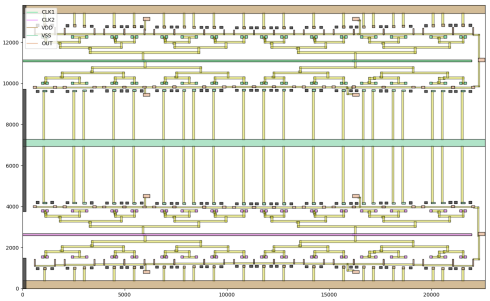


Fig. 9: Clock routing on M2 of 7 bit PI layout.

Step 4: We perform tile routing level by level according to the tree's hierarchy, from high to low. Each node is considered a separate routing segment. After all nodes at the current level complete their routing, the centers of all routings form a square pin point with length and width equal to the clock net width. This serves for the routing of lower-level tree nodes to ensure precise equal length clock routing. To fulfill the requirements of equal length and equal spacing for clock routing, we introduce a unique structure. We set obstacles outside the pin point and its required channel to control the behavior of the gridless routing, as shown in Fig. 8.

Fig. 9 displays the actual result of executing Gridless CTS on M2 metal for a 7-bit PI. Our proposed tile-based CTS enables clock routing to benefit from both top-down and bottom-up advantages. We have extended the concept of contours from [12], generating extra obstacles to effectively guide routing. Most importantly, we can freely adjust the line width and spacing at different levels, achieving better routing results than grid-based clock routing.

V. Experimental Result

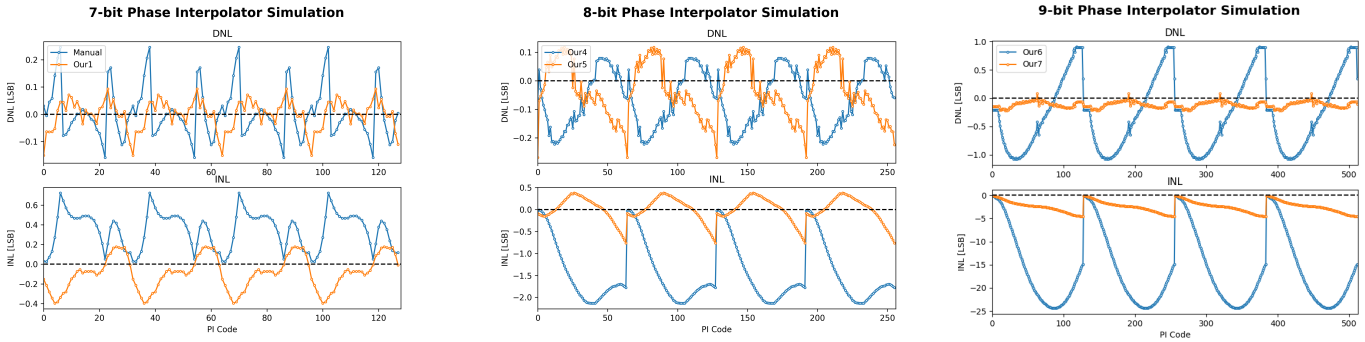
The work we propose has been implemented using the C++ programming language, running on a machine equipped with an Intel Xeon Gold 6240R, a 2.4 GHz CPU and 165 GB of memory. The experiments presented herein utilize the TSMC 28-nm planar technology. All results have successfully passed both the design rule check (DRC) and layout versus schematic (LVS) validations, also obtained post-layout simulation results.

Our target circuit is phase interpolator, which is a common but critical circuit used in high-speed transmission and clock data recovery (CDR) applications. Its operation involves adjusting the phase of a signal (typically the clock) by controlling the input voltage or current. The key performance indicators for this circuit include the Nonlinearity Error, which comprises the Differential Non-Linearity (DNL) and Integral Non-Linearity (INL). DNL measures the deviation between the actual and theoretical values of two consecutive outputs, while INL is the integral of DNL and evaluates the overall linearity performance.

Since our templates are designed for the aforementioned circuits, prior template-based approaches cannot be applied for comparison. In fact, we have tried our best to adopt [1] in our technology but failed. Table I and Fig. 10 present the comparison results of our work on 7-bit, 8-bit, and 9-bit 8GHz phase interpolators (layouts in Fig.11). The "Manual" (Fig. 1) and "Ours1" designs have the same specifications, with "Ours1" having slightly larger area and power consumption.

TABLE I: Comparison of Circuit Performance in 8GHz 7-bit/8-bit/9-bit PI. On circuit layouts with 8-bit and 9-bit resolutions, our tests indicate that using gridless CTS slightly expands the layout. However, it notably reduces the clock length bias from source to sink, significantly decreasing both DNL and INL.

	Design Specifications			Performance Metrics					
Design	R(bit)	Merged	CTS	Time	Area (um^2)	Clock Bias(um)	DNLmax(LSB)	INLpp(LSB)	P(mW)
Pre-sim	7	-	-	-	-	-	0.102	1.113	1.519
Manual	7	Yes	No	16h	220.8	15.76	0.246	0.697	1.551
Ours1	7	Yes	No	10s	257.64	8.46	0.151	0.574	1.621
Ours2	7	No	No	21s	536.94	21.02	0.115	0.867	1.582
Ours3	7	No	Yes	20s	649.98	0.11	0.192	0.915	1.657
Ours4	8	No	No	102s	1514.92	46.60	0.222	2.127	2.716
Ours5	8	No	Yes	103s	1748.58	0.66	0.269	1.145	2.852
Ours6	9	No	No	696s	4601.80	92.09	1.082	24.318	4.372
Ours7	9	No	Yes	688s	5247.80	0.13	0.224	4.573	4.850



(a) **(b)** **(c)**

Fig. 10: Simulation waveforms of 8GHz 7-bit/8-bit/9-bit PI. (a) compares the performance (DNL and INL) of the Manual and Ours1 (in Table I) under the same design specifications, showcasing our superior performance compared to the manual layout. (b) and (c) illustrate the difference with and without the use of CTS. Our Gridless CTS can significantly reduce nonlinear errors, improving post-sim performance.

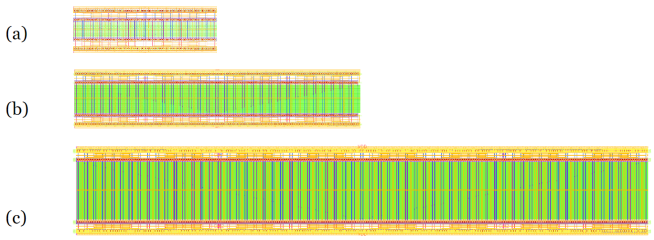


Fig. 11: Phase Interpolator (PI) layouts generated by our method, with sub-figures (a), (b) and (c) corresponding to our results for Ours3, Ours5 and Ours7 (setup in Table I), they are consisted of 256, 512, and 1024 elements respectively.

However, "Ours1" demonstrates significant improvements in DNL and INL, as we have employed precise placement and routing techniques, minimizing artificial errors and reducing the impact of impedance variations. Additionally, our approach offers much shorter runtime, freeing designers from tedious layout tasks.

In order to justify the necessity of our CTS, we compare the results of "Ours2,3", "Ours4,5", and "Ours6,7" to analyze the influence of CTS on the three different resolution PI cores (7-bit, 8-bit, and 9-bit). First, we observe that in all three experiments, enabling CTS requires larger area, and the clock tree structure introduces larger parasitic capacitance, resulting in increased power consumption. Next, we examine the impact of CTS on nonlinearity error. In the case of the 7-bit design, even though "Ours3" has a clock bias approaching 0, the scale of the 7-bit circuit is not large enough to outweigh the negative effects of the clock tree structure (dense structures of clock tree can result in higher impedance). Therefore, the performance of Ours3 (with CTS) is worse compared to Ours2 (without CTS). However, in the 8-bit and 9-bit experiments, we observe that clock path imbalance significantly affects nonlinearity error, particularly in the 9-bit circuit. Implementing CTS in the layout results in a reduction of the INL peak-to-peak value by 80%, indicating that CTS is indispensable for high-resolution PI designs.

VI. Conclusions

In this work, we propose a novel approach that employs customized layout templates to generate precise PI layouts to meet the various requirements of phase interpolators. Subsequently, we employ Gridless Routing System to generate lossless high-quality routing. Experimental results reveal that our methodology can substantially reduce the time spent on layout design, while achieving equivalent or superior performance compared to manual layouts. Future works include the extension of such methodology to other high accuracy analog circuit layouts.

REFERENCES

[1] T. Dhar, K. Kunal, Y. Li, M. Madhusudan, J. Poojary, A. K. Sharma, W. Xu, S. M. Burns, R. Harjani, J. Hu, D. A. Kirkpatrick, P. Mukherjee, S. Yaldiz, and S. S. Sapatnekar, "Align: A system for automating analog layout," *IEEE Design Test*, vol. 38, no. 2, pp. 8–18, 2021.

[2] B. Xu, K. Zhu, M. Liu, Y. Lin, S. Li, X. Tang, N. Sun, and D. Z. Pan, "Magical: Toward fully automated analog ic layout leveraging human and machine intelligence: Invited paper," in *2019 IEEE/ACM International Conference on Computer-Aided Design*, 2019, pp. 1–8.

[3] E. Chang, J. Han, W. Bae, Z. Wang, N. Narevsky, B. Nikolic, and E. Alon, "Bag2: A process-portable framework for generator-based ams circuit design," in *2018 IEEE Custom Integrated Circuits Conference*, 2018.

[4] P. Xu, J. Li, T.-Y. Ho, B. Yu, and K. Zhu, "Performance-driven analog layout automation: Current status and future directions," in *2024 Asia and South Pacific Design Automation Conference*. IEEE Press, 2024.

[5] M. P.-H. Lin, Y.-W. Chang, and C.-M. Hung, "Recent research development and new challenges in analog layout synthesis," in *2016 21st Asia and South Pacific Design Automation Conference*. IEEE Press, 2016, p. 617–622.

[6] R. Castro-Lopez, O. Guerra, E. Roca, and F. V. Fernandez, "An integrated layout-synthesis approach for analog ics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 7, pp. 1179–1189, 2008.

[7] R. Martins, N. Lourenço, and N. Horta, "Laygen ii—automatic layout generation of analog integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 11, pp. 1641–1654, 2013.

[8] J. Han, W. Bae, E. Chang, Z. Wang, B. Nikolic, and E. Alon, "Laygo: A template-and-grid-based layout generation engine for advanced cmos technologies," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 68, no. 3, pp. 1012–1022, 2021.

[9] J. Cong, J. Fang, and K.-Y. Khoo, "Dune-a multilayer gridless routing system," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 5, pp. 633–647, 2001.

[10] Y.-L. Li, H.-Y. Chen, and C.-T. Lin, "Nemo: A new implicit-connection-graph-based gridless router with multilayer planes and pseudo tile propagation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 4, pp. 705–718, 2007.

[11] J. Ousterhout, "Corner stitching: A data-structuring technique for vlsi layout tools," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 87–100, 1984.

[12] J. Dion and L. Monier, "Contour: a tile-based gridless router," in *Research Report 95/3*. Western Research Laboratory, 1995.

[13] M. Jackson, A. Srinivasan, and E. Kuh, "Clock routing for high-performance ics," in *27th ACM/IEEE Design Automation Conference*, 1990, pp. 573–579.