8-31-2022

RC Parasitic-Aware Layout Analysis and Routing Optimization Methodology

Mohamed Saleh Abouelyazid
The American University in Cairo (AUC), mohsaleh@aucegypt.edu

Sherif Hammouda

Yehea Ismail
The American University in Cairo (AUC), Y.ISMAIL@AUCEGYPT.EDU

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

RC Parasitic-Aware Layout Analysis and Routing Optimization Methodology

MOHAMED SALEH ABOUELYAZID[1,2], (Member, IEEE), SHERIF HAMMOUDA[1], AND YEHEA ISMAIL[2], (Fellow, IEEE)

[1]Siemens, EDA, 11835, Cairo, Egypt
[2]The American University in Cairo, 11835, Cairo, Egypt.

Corresponding author: Mohamed Saleh Abouelyazid (e-mail: mohamed_saleh@mentor.com, mohsaleh@aucegypt.edu).

ABSTRACT A parasitic-aware routing optimization and analysis methodology for integrated circuits is developed based on an incremental parasitic extraction and a fast optimization methodology. Existing routing optimization methodologies rely on many circuit simulations, detailed sensitivity analysis, and inefficient simple parasitic models to optimize routes. Moreover, they do not provide a mechanism to help layout designers in identifying problematic layout geometries that have a bad impact on a route's performance. The proposed methodology works on overcoming such problems by providing three features. First, it provides novel sensitivity circuit models to analyze the integrity of signals in layout routes. Such circuit models are based on an accurate matrix circuit representation, a cost function, and an accurate parasitic sensitivity extraction. The circuit models identify critical parasitic elements along with the corresponding layout geometries in a certain route, where they measure the sensitivity of a route's performance to corresponding layout geometries very fast. Therefore, they can correlate the problems of a route's performance to specific layout geometries. Second, the proposed methodology uses a nonlinear programming technique to optimize problematic routes with pre-determined degrees of freedom using the proposed circuit models. Third, the proposed methodology uses a novel incremental parasitic extraction method to extract parasitic elements of modified geometries efficiently. The incremental extraction is used as a part of the routing optimization process to improve the optimization runtime and increase the optimization accuracy. The proposed methodology is tested over different designs of 7nm and 65nm process nodes. The results show that the proposed methodology managed to identify and optimize the problematic geometries in critical routes efficiently with up to 10% performance improvements and a speedup of 3 to 9X as compared to traditional template-based methods.

INDEX TERMS layout routing; routing optimization; parasitic-aware; incremental extraction

I. INTRODUCTION

The continuous scaling down of process technology nodes enabled the integration of more functionalities and systems together on a single chip. Such an integration significantly increased the complexity and density of layouts introducing more parasitic elements. The impact of interconnect parasitic elements on the overall circuit performance keeps increasing from one technology generation to the next. Moreover, the number of interconnect parasitic elements significantly increased in recent advanced processes. Therefore, the effects of interconnect parasitic elements are no longer second order effects. They are now dominating the overall circuit performance [1]–[3]. As a result, it is very important to consider the parasitic effects during placement and routing processes to reduce the overall turn-around-time of a circuit design and improve the yield.

The current optimization flows do not deal with the effects of parasitic elements as dominant factors. They are still dealing with the parasitic effects as second order effects. Moreover, current flows do not provide proper layout analysis and debugging methodologies to help circuit designers in identifying the problematic parasitic elements and the corresponding layout geometries. As a result, circuit designers need to manually analyze the impact of interconnect parasitic elements on a circuit performance, which is a very time-consuming and error-prone operation.

Nowadays, the time consumed in analyzing the post-layout simulation results is more critical than post-layout simulation runtime itself. Therefore, there is an increasing demand to provide algorithms that help circuit designers in understanding the impact of parasitic elements on post-layout simulation results and identifying the most problematic parasitic elements along with the corresponding layout geometries in a given layout.

Automatic layout generation and optimization tools are used by layout designers to generate a layout that meets the required circuit specifications. Such tools are commonly used for digital circuit designs, where cell-based tools are employed to cover circuit synthesis, mapping, and physical design steps [4]. On the other hand, analog layout generation tools do not provide full automation environment for analog circuits, where analog circuit designers still need to do many manual analysis and layout modifications in order to meet the required circuit specifications. In analog designs, the layout optimization tools are usually used to determine device sizes, circuit topologies, and routing paths. However, they still deal with the effects of interconnect (i.e., route) parasitic elements as second order effects ignoring that the interconnect parasitic effects became one of the dominant factors on a circuit performance in advanced process nodes, especially those parasitic effects that are associated with critical nets. In order to control the effects of parasitic elements, the corresponding routes need to be routed in a way that reduces the associated parasitic elements [5], [6].

Routing is the process of creating connections between devices. The routing is mainly divided into two stages that include global and detailed routing. The global routing is responsible for identifying general paths of each connection. It usually divides the routing region into windows and identifies the general window-to-window paths for all connections (i.e., routes) [7]. On the other hand, the detailed routing is responsible for identifying exact paths, metal layers, and vias for each net in a certain layout. The routing processes usually consider multiple constraints, such as maintaining net symmetry, minimizing wire lengths, having a maximum number of vias, and minimizing parasitic elements [6], [8].

In net symmetry constraints, the layout geometrical matching is no longer enough to achieve a net symmetry as it does not necessarily provide a performance matching across the required nets. This problem significantly increases in advanced process technology nodes because layouts became more complicated and the parasitic coupling interactions with the surrounding polygons significantly increased. In order to achieve the performance matching, the parasitic elements of the target nets need to be considered while applying the net symmetry constraint. In other words, the accuracy requirements of parasitic-aware routing processes significantly increased in advanced process nodes requiring more accurate parasitic models [6].

Parasitic-aware routing processes aim to reduce the parasitic elements that are mainly associated with critical routes in order to meet the required circuit's specifications. This is done by modifying layout geometries of critical routes in a way that reduces the effects of associated parasitic elements. However, modifying layout geometries will not only impact the associated parasitic elements, but it will also impact the parasitic interactions among surrounding and nearby metals. Therefore, a full layout parasitic extraction is required with every change in routes in order to accurately measure the impact of modifying the routes [6].

Many efforts were done to provide parasitic-aware routing optimization methods; however, they use either simplified parasitic models such as in [5], [9]–[14], or a full layout parasitic extraction such as in [15]–[17] to extract the parasitic elements of a layout design during the optimization processes. As for the methods that use simplified parasitic models, they provide a faster layout routing optimization; however, they are less accurate, and their parasitic extraction accuracy cannot cope with the accuracy requirements of advanced process nodes [18], [19]. On the other hand, the methods that use a full layout parasitic extraction provide more accurate layout optimization; however, they are very slow as they require a full layout parasitic extraction with every iteration in the optimization loop. Therefore, such methods are inefficient for designs with many nets (e.g., more than 100K nets), where the runtime of a single full layout parasitic extraction, using a rule-based extractor, may exceed several hours according to our experimental results. Moreover, previous efforts did not provide a systematic method to analyze the impact of parasitic elements and geometry modifications on a circuit performance.

This paper aims to develop a new parasitic-aware routing optimization methodology. The proposed methodology can be applied either after or within the detailed routing step. The proposed methodology enables circuit designers to debug and analyze the impact of parasitic elements on a circuit performance. Also, it provides a mechanism to identify the problematic parasitic elements and correlate them with specific layout geometries. Moreover, it uses nonlinear programming to re-route the problematic paths (i.e., routes) in order to achieve the required specifications with a full consideration of the surrounding environment. The proposed methodology uses a novel incremental parasitic extraction method in order to extract the parasitic elements of a modified layout during the optimization process. The proposed incremental extraction method provides very accurate parasitic extraction results with a maximum error < 1% as compared to a full layout extraction.

The contributions of this paper are:

a. Circuit models to measure and analyze the impact of parasitic elements and corresponding layout geometries on a pre-defined cost function, such as net symmetry and maximum delay cost functions. In other words, they measure the sensitivity of system's performance cost function to layout geometries.

b. The proposed models are used in an algorithm that identifies the geometries and parasitic elements that the system's performance is most sensitive to without any circuit simulations.
c. A parasitic-aware routing optimization algorithm that uses nonlinear programming to automatically modify the most critical routes in order to meet the required performance cost function without circuit simulations. The proposed algorithm accepts pre-determined degrees of freedom (e.g., route's corners) and dynamic constraints. Therefore, the proposed routing algorithm optimizes a performance cost function taking the corresponding RC parasitic elements into consideration.
d. A novel incremental parasitic extraction methodology that considers second order parasitic capacitance effects efficiently. The proposed incremental methodology is applied on top of a full layout parasitic extraction tool, Calibre xRC, rule-based extractor [20]. It provides very accurate parasitic extraction results with a maximum error < 1% and a speedup of up to 40X as compared to a full layout extraction.
e. The testing of the proposed routing algorithm is performed by using a template-based layout optimization flow. We replaced the routing optimization algorithm of the template-based layout optimization flow with the proposed routing optimization algorithm.
f. The proposed methodology is tested on different designs of 7nm, 40nm, and 65nm process nodes.

The rest of the paper is organized as follows. Section II provides the related work. Section III provides a background on parasitic-aware layout optimization methods and system moments. Section IV introduces the proposed incremental parasitic extraction method. Section V describes the proposed parasitic-aware layout routing optimization methodology. Section VI shows the experimental results. Section VII provides the conclusion and future works. Moreover, Table 1 shows a list of abbreviations and symbols that are commonly used in this work.

Table 1. List of abbreviations and symbols.

Abbreviation or symbol	Definition
PDK	Process design kit
RO	Ring oscillator
VCO	Voltage-controlled oscillator
CF	Cost function
RCF	Relative cost function
DCF	Delay cost function
MR	Maximum coupling capacitance interaction range
BW	Band width
PM	Phase margin
GM	Gain margin
G	Admittance matrix
C	Capacitance matrix
P_i	A parasitic element
R_i	A parasitic resistive element
Cc_j	A parasitic capacitive element
m_k	A circuit moment at k^{th} order
r_t	A threshold voltage to a maximum voltage ratio
S_i	A circuit (i.e., network) response
G_e	Route geometries
Π	RC pi model

II. RELATED WORK

Most of existing parasitic-aware routing methods suffer from two problems. First, they use either simplified parasitic formulas or a full layout parasitic extraction in order to measure the parasitic elements for each layout modification in the design loop. The simplified parasitic models are not accurate and cannot cope with the increasing parasitic extraction accuracy requirements in advanced nodes leading to inaccurate layout optimization. On the other hand, the use of a full layout extraction is very time-consuming and not suitable for optimizing large layout designs. Second, the existing routing optimization methods do not provide a systematic way to help circuit designers in understanding the impact of parasitic elements and the corresponding layout geometries on a system's (i.e., route) performance.

In [21], a parasitic-aware routing method was developed based on simplified parasitic formulas. This approach aims to reduce the delay and routing area considering the interconnect parasitic elements of a given layout. This method identifies multiple candidate routes for each connection. Then, it evaluates the performance of each candidate until the candidates that meet the required performance are achieved. This method has three main problems. First, it uses simplified parasitic formulas that cannot cope with the new accuracy requirements of advanced process nodes [18], [19]. Second, this method does not deal with the parasitic effects as dominant factors on a circuit's performance. Third, this method relies on a pre-determined set of candidate routes that do not necessarily achieve the required performance.

In [10], an automatic optimization-based sizing and routing methodology was developed for analog circuits. This methodology uses a layout generator that computes the optimal electrical current correct wire topology and global routing in loop for each different sizing solution. Such a methodology relies on simplified parasitic models in order to achieve reasonable optimization runtime as it requires many optimization loops (i.e., iterations). This methodology has three main problems. First, it requires many iterations to achieve good results. Second, it uses simplified parasitic formulas that cannot cope with the new accuracy requirements of advanced process nodes. Third, it does not deal with the parasitic effects as dominant factors on circuit's performance.

In [22]–[24], parasitic-aware routing methodologies based on circuit moments were developed. The proposed methodologies aim to optimize layout routes by minimizing a cost function. The cost function considers parasitic resistance, capacitance, self-inductance, and mutual

coupling inductance effects (RLCK), and it provides a representation of the delay and ringing of the signals. Therefore, the minimization of the developed cost function helps in achieving a good balance between route's delay and ringing. These efforts have five problems. First, they require a full layout parasitic extraction in order to evaluate the corresponding cost function with every optimization iteration. Second, the cost function is only valid for delay and ringing effects. Third, they are not suitable for both net symmetry constraints and analog designs. Fourth, they do not provide good understanding to the impact of parasitic effects on a route's performance. Fifth, they do not correlate parasitic elements to certain geometries.

In [13], a template-based parasitic-aware layout optimization method was developed. As for the routing optimization, traditional template-based methods optimize layout routes in *x* and *y* directions separately. This method aims to overcome this problem by optimizing layout routes in *x* and *y* directions simultaneously. Such a method uses a hybrid algorithm that consists of nonlinear programming and graph-based algorithms in order to achieve more accurate layout optimization. However, this method has three problems. First, it does not deal with the parasitic effects as dominant factors on a circuit's performance as it uses very simple parasitic formulas to extract the parasitic elements of a given layout. Such formulas cannot cope with the new accuracy requirements of advanced process nodes. Second, it does not provide a mechanism to help circuit designers in understanding the impact of parasitic effects on a system's (i.e., route) performance. Third, it only considers rectilinear and Manhattan geometries, and it cannot handle non-Manhattan geometries.

In [5], [12], [14], [25], template-based parasitic-aware routing optimization methodologies were proposed. They aim to create a symbolic template with a set of constraints such as net symmetry, connectivity, parasitic bounds, and corresponding design rules. The calculations of parasitic bounds rely on multiple circuit simulations in order to identify a parasitic bound for each parasitic element. The parasitic model for each route is represented by a simple RC Π (i.e., pi) model in order to speed up the calculations of parasitic bounds. Such methodologies are fast; however, they are suffering from three problems. First, they use simplified parasitic formulas that cannot cope with the new accuracy requirements of advanced nodes. Second, they do not provide a mechanism to help circuit designers in understanding the impact of parasitic effects on a system's (i.e., route) performance. Third, most of them cannot handle non-Manhattan geometries.

In [26], analog layout design tool called LAYGEN II was developed. It uses a symbolic template (i.e., template-based) approach in order to perform placement and routing. This approach is very efficient in achieving a good initial layout for a given circuit design; however, it requires a lot of computational resources in order to handle large layouts.

In [27], an analog layout design tool was developed. As for the routing optimization, this method uses a combination of symbolic template (i.e., template-based) and optimization approaches in order to generate layouts. This method uses a template approach in order to reduce the search (i.e., solution) space. This method is efficient in achieving a good initial layout for a given circuit design; however, it requires a lot of computational resources in order to handle large layouts. Moreover, it is not designed to handle non-Manhattan geometries.

In [28], a routing algorithm was developed using a discrete particle swarm optimization and multi-stage transformation methods. The proposed algorithm optimizes layout routes using two types of Steiner minimal tree models that include Manhattan and non-Manhattan Steiner minimal trees. Therefore, the selected route structure can contain Manhattan and non-Manhattan geometries. This flow has two problems. First, it does not consider the impact of parasitic elements except for a route's delay. Second, it does not have a mechanism to help circuit designers in understanding the impact of parasitic elements on system's performance.

The problems of existing routing optimization methods can be summarized as below:

a. They do not provide a mechanism to help circuit designers in understanding the impact of parasitic elements on a system's (i.e., route) performance, such as identifying the problematic parasitic elements along with the corresponding layout geometries.
b. Most of existing efforts use either simplified parasitic formulas, such as in [5], [11]–[13], [27], [29], and [30], or a full layout extraction, such as in [15], [22]–[24], in order to extract the parasitic elements of a given layout. The methods that use simplified parasitic formulas suffer from an accuracy problem as the accuracy of such parasitic formulas cannot cope with the increasing accuracy requirements in advanced process nodes, whereas the methods that use a full layout extraction suffer from a long runtime problem as they require a full layout extraction with every optimization iteration.
c. Many efforts do not pay much attention to the nonlinear relationship between parasitic elements and layout geometries, such as in template-based approaches [5], [13], [14]. These efforts optimize layout routes in the *x* and *y* directions separately (one after another). Such a way of optimization cannot provide efficient results when it comes to the nonlinearity of parasitic constraints.
d. Many of existing efforts perform circuit simulations inside the optimization loops as in [31], [32].

This work focuses on overcoming these problems. First, it provides a routing optimization method that can be applied either after or within the detailed routing. Second, it provides sensitivity circuit models that help circuit designers in understanding the impact of parasitic elements and the corresponding layout geometries on a route's performance. Third, it uses a novel incremental parasitic extraction method to extract the parasitic elements of modified layouts during the optimization process. Such an incremental method provides very accurate results (<1% error) with a speedup of up to 40X as compared to a full layout extraction. Fourth, it does not require multiple circuit simulations. Table 2 provides a functional comparison among related works and our work.

Table 2. A comparison among state-of-the art routing optimization works including our work.

	Routing methodology and its main limitations	**Contrib-ution in circuit sizing**	**Parasitic extraction**	**Requires a circuit simulation**	**Handling of Non-Manhattan geometries**	**Models to analyze the impact of layout geometries on a system's performance**
Smey *et al*., [21]	**Routing Methodology:** • This method optimizes routes in two steps. First, it generates a set of candidate routes for each connection. Then, it selects the candidate route with minimum area and minimum delay for each connection. **Main limitations:** • It relies on pre-determined candidate routes that do not necessarily achieve the required performance. • It uses simplified RC parasitic formulas that cannot handle complicated layout structures in advanced nodes.	No	Simplified 2D cross-section models for RC elements.	Yes	No	No
Lourenco *et al*., [10]	**Routing Methodology:** • Automatic electromigration-aware wire topology and global routing in-loop for each different sizing solution. **Main limitations:** • It requires many iterations to achieve good results. • It uses simplified RC parasitic formulas that cannot handle complicated layout structures in advanced nodes.	Yes	Simplified 2D cross-section models for RC elements.	Yes	No	No
Bhaduri and Vemuri [22]–[24]	**Routing Methodology:** • They optimize routes by minimizing a cost function that provides a balance between the delay and ringing effects. • They use template-based approach to generate routing candidates. **Main limitations:** • They use a full layout parasitic extraction to evaluate the cost function, which consumes a lot of time. • The developed cost function has limited applications as it only considers the delay and ringing effects.	No	A full layout extraction for RLCK elements.	No	No	No
Zhang *et al*., [14], Liu and Zhang [5], [25], and Bhattacharya *et al*., [12].	**Routing Methodology:** • A symbolic template approach that is used to minimize parasitic effects and a route's area. **Main limitations:** • They use simple RC parasitic formulas that cannot handle complicated layout structures in advanced nodes.	No	Simplified 2D cross-section models for RC elements.	Yes	No	No
Naguib *et al*., [27]	**Routing Methodology:** • A symbolic template approach that is used to minimize parasitic effects and a route's area. **Main limitations:** • It requires a lot of computational resources in order to handle large layout designs.	Yes	Simplified 2D cross-section models for RC elements.	Yes	No	No
Liu *et al*., [28]	**Routing Methodology:** • Swarm optimization algorithms that are used to minimize parasitic effects and a route's area. **Main limitations:** • It does not consider the impact of parasitic elements except for a route's delay.	No	A full layout extraction for RC elements.	No	Yes	No
This work	**Routing Methodology:** • Nonlinear programming to minimize a performance cost function based on circuit moments and sensitivity models. **Main limitation:** • It only considers RC parasitic elements. Hence, this model is appropriate for local interconnect at any frequency and global interconnect at a lower frequency. For high frequency global interconnect, inductance and more complex models are needed.	No	Incremental layout parasitic extraction for RC elements	No	Yes	Yes

III. BACKGROUND

A. TEMPLATE-BASED PARASITIC-AWARE LAYOUT OPTIMIZATION

A layout optimization is the process of modifying and optimizing layout designs in order to meet the required circuit specifications. One of the most efficient layout optimization methods is the template-based method. The template-based method is used to either migrate a layout design from one process node to another or optimize an existing layout to meet the required constraints and specifications. It consists of two main steps that include symbolic template extraction and layout generation steps as shown in Figure 1.

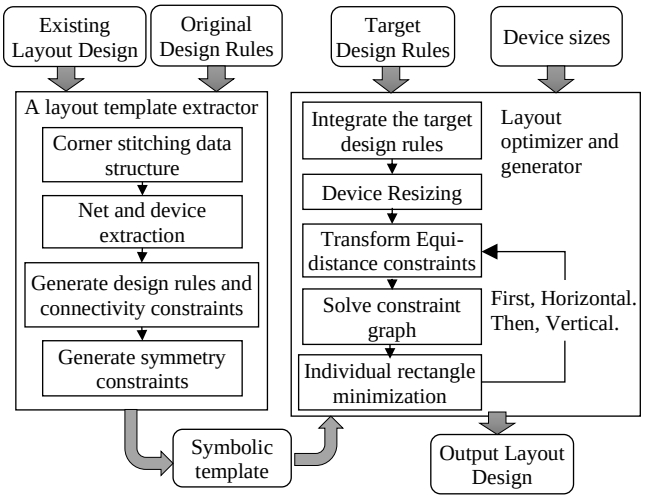
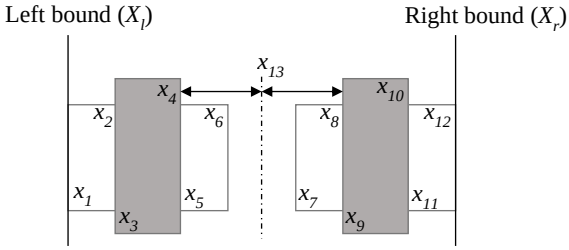


Figure 1. **Template-based layout optimization flow [13], [14].**

The Symbolic template step is responsible for generating a set of geometrical and electrical constraints (i.e., symbolic template) for an existing layout considering the required circuit specifications [5], [12]–[14]. The symbolic template is usually represented by mathematical formulas (e.g., compaction formulas) such as in Figure 2. On the other hand, the layout generation step is responsible for optimizing and generating a layout that meets the required specifications taking into consideration the obtained symbolic template constraints and the new design requirements. As shown in Figure 1, the layout generation (or optimization) step starts with a device sizing followed by a routing optimization, which is performed in the horizontal and vertical directions separately.

The routing optimization processes must consider the impact of parasitic elements on a circuit performance to achieve more accurate optimization results. Therefore, parasitic constraints are obtained and converted into geometrical constraints.



min(X_r - X_l), subject to:

$(x_1 - X_l) \geq 0$, $(x_3 - X_l) \geq 0$, …

$(X_r - x_{12}) \geq 0$, $(X_r - x_{10}) \geq 0$, …

$(x_2 - x_1) \geq 2$, $(x_{12} - x_{11}) \geq 2$, …

$(x_3 - x_2) = 0$, $(x_{11} - x_{10}) = 0$, …

$(x_4 - x_3) = (x_{10} - x_9)$, $(x_{13} - x_4) = (x_9 - x_{13})$, …

Figure 2. **An Example of template geometrical constraints, in the *x*-direction, for a simple layout [14].**

B. SYSTEM MOMENTS

Assuming an RC linear circuit, the corresponding general nodal analysis equations are given by:

$$G\,\underline{V} + C\,\dot{\underline{V}} = \underline{b}, \tag{1}$$

where G is an $n \times n$ admittance matrix that is obtained from the interconnections among the resistive elements, C is an $n \times n$ capacitance matrix that is obtained from the interconnections among the capacitive elements, $\underline{b}$ is a vector of size n that represent the inputs at each node, $\underline{V}$ is a vector with n state variables that represent the capacitor voltages (i.e., voltage response at each node), whereas n represent the number of nodes (or capacitor voltages) for a linear system with RC elements. The response, $V(s)$, at any node in a given linear circuit can be expressed by a Taylor series expansion as below [33]:

$$V(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \cdots, \tag{2}$$

where m_i represents the i^{th} moment of a given linear system at a given node.

Substitute (2) in (1), we get:

$$G[m_0 + m_1 s + m_2 s^2 + \cdots] + C\,s[m_0 + m_1 s + m_2 s^2 + \cdots] = \underline{b}\,. \tag{3}$$

Equating the coefficients of s^n in both sides of (3), we get [33]:

$$\begin{aligned} G\,\underline{m_0} &= \underline{b} \\ \underline{m_0} &= G^{-1}\,\underline{b} \\ \underline{m_1} &= G^{-1}\,C\,\underline{m_0} \\ \underline{m_2} &= G^{-1}\,C\,\underline{m_1} \\ &\vdots \\ \underline{m_n} &= G^{-1}\,C\,\underline{m_{n-1}}. \end{aligned} \tag{4}$$

Therefore, the moments of a linear system provide a detailed representation of its response (i.e., a system response) as shown in (2), and system moments can be obtained by (4) [33].

IV. INCREMENTAL RC PARASITIC EXTRACTION

The layout parasitic extraction is an essential step in conventional integrated circuit (IC) design flows. It is used to extract parasitic elements of a given layout in order to perform a post-layout simulation. If the post-layout simulation results did not meet the required circuit's specifications, layout designers would modify the corresponding layout until its post-layout simulation results meet the required specifications. Usually, this process requires several iterations of layout modifications, parasitic extractions, and post-layout simulations until convergence.

There are two approaches to reduce the turn-around-time of the layout parasitic extraction step in design loops. First, some approaches use simplified parasitic models to speed up the extraction process and reduce the parasitic network such as in [12]–[14]. This approach is not efficient in advanced process technology nodes as it handles the parasitic effects as second order effects ignoring that the interconnect parasitic effects became one of the dominant factors on a circuit's performance in such advanced nodes [18], [19], [34]. Second, other approaches may use an incremental parasitic extraction to limit the parasitic extraction process to the modified polygons in a given layout. As a result, the execution time (i.e., runtime) of the layout parasitic extraction step in design loops decreases significantly with minimal impact on the extraction accuracy as compared to the use of a full layout parasitic extraction.

The incremental parasitic extraction aims to identify the modified layout geometries, extract the corresponding parasitic elements, and update the corresponding circuit network (i.e., netlist) with the newly extracted parasitic elements. In our work, the incremental parasitic extraction is used to extract parasitic resistances and capacitances of modified areas in a given layout.

A. INCREMENTAL PARASITIC RESISTANCE EXTRACTION

As for parasitic resistances, they only depend on the geometrical shapes of modified layouts, and they do not depend on the surrounding environment. Therefore, the incremental parasitic resistance extraction identifies the modified layout polygons and re-extracts their parasitic resistances smoothly without any consideration of the surrounding environment. After that, the corresponding circuit network (i.e., netlist) are updated with the newly extracted parasitic resistive elements.

B. INCREMENTAL PARASITIC CAPACITANCE EXTRACTION

The incremental extraction of parasitic capacitances is more complicated than the incremental extraction of parasitic resistances because parasitic capacitances are highly correlated with the surrounding environment. In other words, if a layout polygon is modified, the modifications will not only impact the associated parasitic capacitive elements, but also, they will impact the parasitic capacitive elements among nearby metal polygons. Therefore, the incremental parasitic capacitance extraction needs to select and re-extract the parasitic capacitive elements that are impacted by layout modifications.

Existing incremental parasitic extraction methods can re-extract parasitic resistances efficiently; however, they cannot efficiently re-extract parasitic capacitance. This is because existing incremental methods only re-extract parasitic capacitances that are directly coupled with modified shapes (i.e., first order parasitic capacitances), and they ignore all coupling capacitances that are not directly coupled to modified layout shapes, such as second order coupling capacitances as shown in Figure 3, even if those capacitances are significantly impacted by layout modifications [35], [36]. As a result, they provide a low extraction accuracy as compared to a full layout parasitic capacitance extraction. Figure 4 shows an example of modifying the position of a nearby polygon on the second order coupling capacitance between two other fixed polygons.

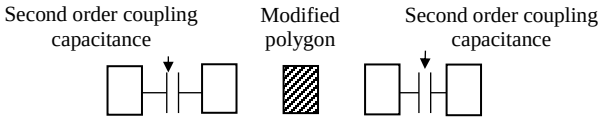


Figure 3. An example of second order coupling capacitances due to modifying a certain metal polygon.

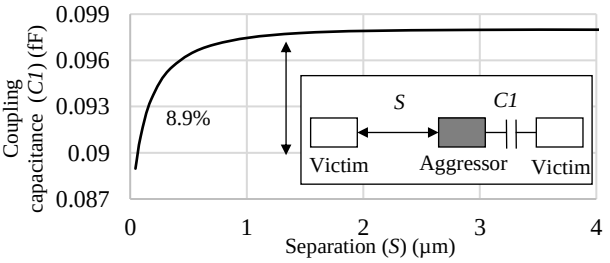


Figure 4. The impact of increasing the separation between the aggressor and left victim polygons on the coupling between the aggressor and right victim polygons. The experiment used metal5 layer of 28nm process technology node.

A novel incremental parasitic capacitance extraction method is developed to extract first and second order capacitances efficiently. The developed method provides outstanding accuracy results as compared to a full layout extraction with a maximum relative error < 1%. Moreover, the impact of extracting second order capacitances on the total extraction runtime is negligible, where the time required to extract second order capacitances represents < 5% of the total incremental extraction runtime. The developed method has three main steps. First, it identifies the modified shapes and the corresponding metal layers. Second, it calculates a maximum coupling capacitance interaction range (MR) for each metal layer. Third, it extracts all coupling capacitances that are enclosed inside the maximum interaction range, and it updates the corresponding circuit's network (i.e., netlist)

with the newly extracted parasitic capacitive elements. The three steps of the developed incremental capacitance extraction method are described in more details as follows.

1) IDENTIFY MODIFIED SHAPES

In this step, all metal polygons that are impacted by layout modifications are marked, where the modified metal polygons are marked, and the metal polygons that were previously interacting with the modified polygons (before modifications) are also marked. This is to ensure that all impacted parasitic capacitances are considered during the incremental extraction process.

2) CALCULATING THE MAXIMUM CAPACITANCE INTERACTION RANGE

In this step, a maximum capacitance interaction range (MR) is calculated for each metal layer and stored a pre-characterized library in order to be later used by the incremental extraction flow, given that each process technology node has a pre-characterized library of MR values. For a certain process node, the pre-characterized library is created only once and used numerous times by the incremental parasitic extraction flow for different layout designs.

The maximum capacitance interaction range (MR) of a polygon represents the range (i.e., distance) where coupling capacitances to other polygons are negligible and do not impact the accuracy of a parasitic capacitance extraction. In other words, the MR identifies the valid coupling range for each layout polygon in order to avoid unnecessarily capacitance computations. The calculation of an MR depends on the corresponding metal stack specifications, where each metal layer in a certain process node has a different MR value.

For a certain metal layer, the MR is calculated by constructing two adjacent metal polygons using the corresponding minimum dimensions. Then, an electrostatic simulator is used to extract the lateral coupling capacitance between the two polygons accurately. Also, the simulator performs a parametric sweep over lateral spacings while it measures the coupling capacitance between the two metal polygons until the MR is achieved, given that the MR represents the distance where the coupling capacitance between the two polygons is less than or equal to 1% of the total capacitance on one of the polygons as shown in Figure 5 [18], [19].

In [18] and [19], the MR is used in a full layout parasitic capacitance extraction to identify the maximum coupling interaction distance for each metal polygon; however, in this work, the MR is used in an incremental parasitic capacitance extraction to identify the capacitance elements that are impacted by layout modifications, given that such impacted capacitance elements do not necessarily have direct coupling interactions with any modified metal polygon.

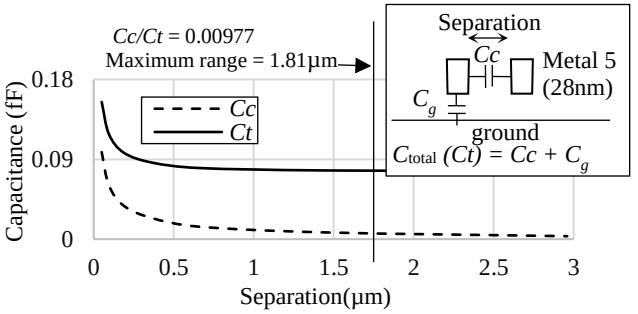


Figure 5. The impact of increasing the separation (i.e., spacing) between two metal polygons on the lateral coupling capacitance between them using metal5 of 28nm process technology node [18], [19].

3) CAPACITANCE EXTRACTION AND NETLIST UPDATE:

In this step, the maximum interaction ranges of all modified polygons are obtained from the corresponding pre-characterized library. Then, all parasitic capacitive elements that are enclosed inside this range are re-extracted including second order parasitic capacitances. This ensures that all impacted capacitive elements are extracted, whereas the capacitive elements that are not enclosed inside the maximum interaction ranges are not extracted as shown in Figure 6. Eventually, the corresponding circuit's network (i.e., netlist) is updated with the newly extracted parasitic capacitive elements.

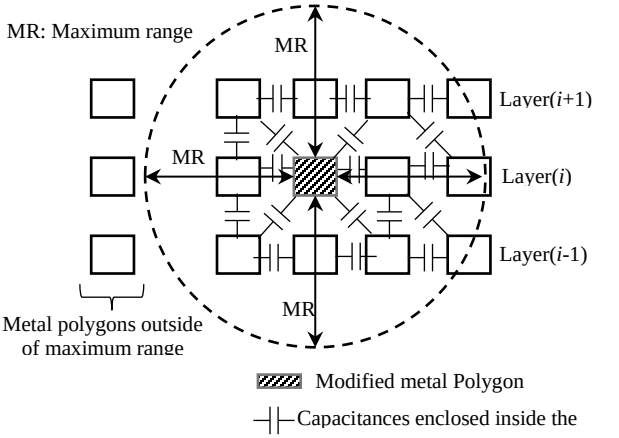


Figure 6. An illustrative example of 2D cross-section metal polygons showing some capacitive elements that are enclosed inside the maximum capacitance interaction range of a modified polygon.

V. PARASITIC-AWARE ROUTING OPTIMIZATION METHODOLOGY

Parasitic-aware routing optimization methodology based on circuit moments is developed. The proposed routing methodology is used as a part of a template-based layout optimization flow. The proposed methodology has three main benefits. First, it helps circuit designers in analyzing the performance of critical routes. This is done by developing a sensitivity circuit model that measures the sensitivity of a route's performance cost function to the corresponding metal geometries. Second, the proposed methodology efficiently considers the impact of parasitic elements during the optimization of critical routes by using a novel incremental

parasitic extraction method. Third, the proposed methodology optimizes critical routes very fast using a cost function and corresponding sensitivity circuit models. The critical routes represent the routes that either hold analog signals or have a considerable impact on a circuit's performance. Such routes are identified by circuit designers after performing a sensitivity analysis across different routes, i.e., the sensitivity of a circuit performance to a route's network including parasitic elements.

The proposed methodology consists of three main steps as shown in Figure 7. First, a performance cost function is developed, for example, a relative cost function that measures the performance difference between two routes. Second, sensitivity circuit models are derived to measure the sensitivities of a cost function to route's geometries. Third, a nonlinear programming is used to minimize a cost function subject to route's geometries considering the obtained sensitivity circuit models. The cost function minimization process considers different geometry constraints such as connectivity, blockages, and net symmetry constraints. Moreover, the optimization process can handle Manhattan and non-Manhattan geometries.

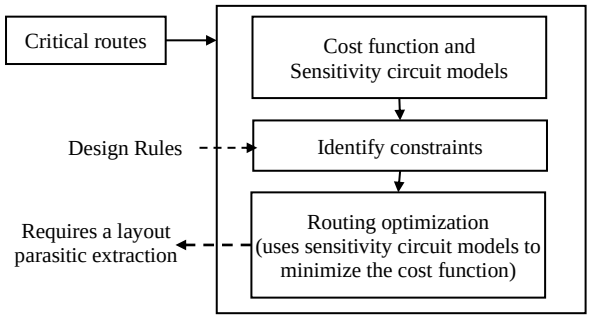


Figure 7. The proposed layout optimization flow for critical routes.

The nonlinear programming requires a layout parasitic extraction process with every optimization iteration to evaluate the developed cost function. Therefore, a novel incremental parasitic extraction method is developed, as described in section IV. The developed incremental extraction method employs a full layout extraction tool, Calibre xRC, rule-based extractor [20], in an incremental manner in order to reduce the parasitic extraction runtime. Moreover, it provides high accuracy numbers as compared to a full layout extraction (<1% error).

A. COST FUNCTION DEVELOPMENT

Two cost functions are developed. The first one represents a net matching (i.e., symmetry), whereas the second one represents a route's delay.

1) NET MATCHING COST FUNCTION:

A cost function that measures the performance difference between two systems (i.e., routes) is developed as follows. Assuming two systems with output responses S_1 and S_2. The systems can belong to the same net, as shown in Figure 8 (a), or different nets, as shown in Figure 8 (b). The corresponding responses at their terminals are expressed by Taylor series expansions as below:

$$S_1(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \cdots, \quad (5)$$

and

$$S_2(s) = m'_0 + m'_1 s + m'_2 s^2 + m'_3 s^3 + \cdots, \quad (6)$$

where m_i and m'_i are circuit moments at i^{th} order.

To ensure that the two systems have the same output response, a relative cost function (RCF) is developed as below:

$$\text{relative cost function (RCF)} = \sum_{i=0}^{q} \frac{(m_i - m'_i)^2}{{m'_i}^2}, \quad (7)$$

where q represents the required order of circuit moments. The purpose of using a relative formula is to normalize the weights for all required moments to ensure that all required moments are equally considered (regardless of their order of magnitude) during the optimization process.

The RCF has two main uses. First, it is used to meet net symmetry constraints as it measures the performance (or response) error between two routes. Second, it is used to optimize critical layout routes by measuring the performance error between a certain critical route and the corresponding shortest path route assuming no blockages.

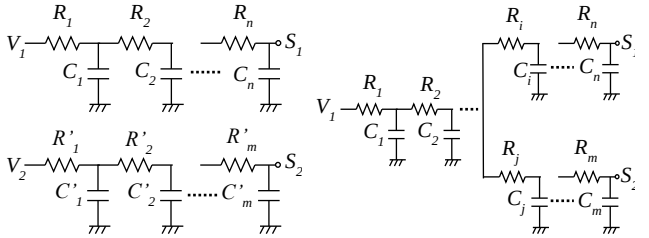


Figure 8. Two different RC systems that belong to (a) two different routes, or (b) the same route.

2) DELAY MINIMIZATION COST FUNCTION:

Another cost function is developed based on circuit moments in order to minimize a route's delay. According to [37], for a certain network, the crossing time ($t_{rt,q}$) represents the time required by a signal to reach a certain voltage as shown in Figure 9. The crossing time ($t_{rt,q}$) of a signal at a certain threshold ratio of a voltage (r_t) for q moments is given by:

$$t_{rt,q} = a_1 \cdot m_1 + a_2 \cdot \frac{m_2}{m_1} + a_3 \cdot \frac{m_3}{{m_1}^2} + \cdots + a_q \cdot \frac{m_q}{{m_1}^{(q-1)}}, \quad (8)$$

where the valid range of r_t is from 0 to 1, $t_{rt,q}$ is the time taken by the signal to achieve (or cross) the threshold voltage, q is the required order of moment, whereas a_1 to a_q are constant coefficients that might have different values based on the required threshold value (r_t). These constants were obtained using curve fitting operations as shown in [37].

In this work, a delay cost function is developed based on (8). The threshold voltage ratio of the crossing point is set to 0.5, and the maximum number of moments (q) is set to five moments, as recommended by [37] to achieve a good

accuracy. Therefore, the delay cost function (DCF) is given by:

$$\text{Delay cost function (DCF)} = t_{0.5,5}, \quad (9)$$

where the values of a_1 to a_5 coefficients are -3.05, 5.59, -4.36, 1.75, and -0.291, respectively as shown in [37].

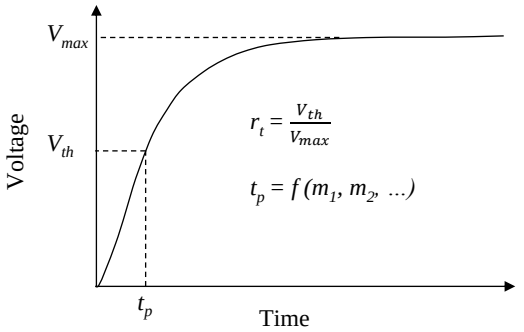


Figure 9. An illustrative example of the threshold ratio (r_t) that represents the threshold-crossing point (t_p, V_{th}), where a time t_p is required by the signal to reach V_{th} voltage.

B. SENSITIVITY CIRCUIT MODELS

In order to measure the impact of modifying layout geometries (i.e., route's geometries) on a cost function (CF), a circuit model that measures the sensitivity of CF to layout geometries is proposed and derived as below:

$$\frac{\overline{\partial \mathrm{CF}}}{\partial Ge} = \left[\frac{\partial \mathrm{CF}}{\partial P}\right]_{1\times n} \cdot \left[\frac{\partial P}{\partial Ge}\right]_{n\times m}, \quad (10)$$

where P represents the associated parasitic elements, Ge represents route's geometries, n is the number of parasitic elements, whereas m is the number of corresponding layout geometries. In order to correlate the cost function with layout geometries (Ge), the geometries are represented by using their coordinates (or vertices). Therefore, the sensitivity of a cost function (CF) to layout geometries is given by:

$$\frac{\overline{\partial CF}}{\partial Ge} = \begin{bmatrix} \frac{\partial CF}{\partial R_1} \\ \frac{\partial CF}{\partial R_2} \\ \vdots \\ \frac{\partial CF}{\partial R_i} \\ \frac{\partial CF}{\partial Cc_{i+1}} \\ \frac{\partial CF}{\partial Cc_{i+2}} \\ \vdots \\ \frac{\partial CF}{\partial Cc_n} \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial R_1}{\partial x_1} & \frac{\partial R_1}{\partial x_2} & \cdots & \frac{\partial R_1}{\partial y_{m-1}} & \frac{\partial R_1}{\partial y_m} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial Cc_n}{\partial x_1} & \frac{\partial Cc_n}{\partial x_2} & \cdots & \frac{\partial Cc_n}{\partial y_{m-1}} & \frac{\partial Cc_n}{\partial y_m} \end{bmatrix}, \quad (11)$$

where x and y represent the coordinates of route polygons as shown in Figure 10, R is a parasitic resistive element, whereas Cc is a parasitic capacitive element. In order to provide a degree of freedom, routes are fractured into quadrilateral polygons (e.g., rectangles). As a result, the sensitivity and cost function calculations consider either Manhattan or non-Manhattan geometries.

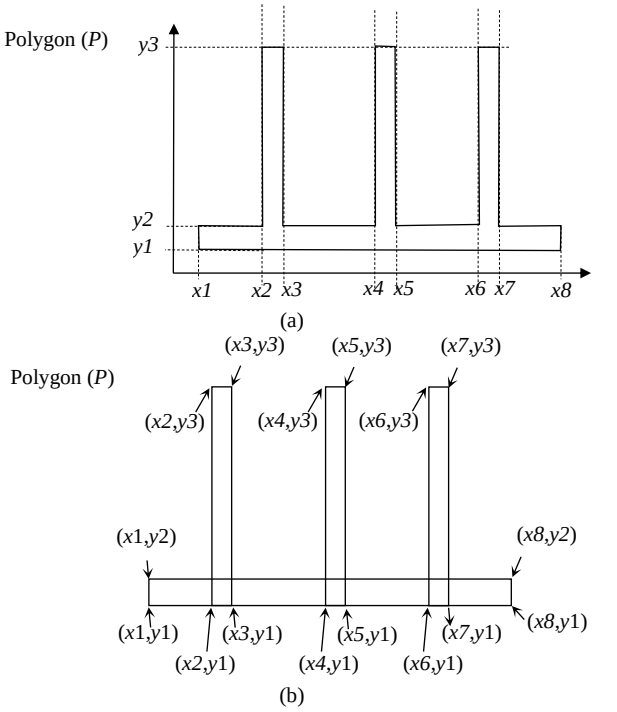


Figure 10. An illustrative example of a geometry representation in the proposed sensitivity models showing (a) an unfractured polygon and (b) a fractured polygon.

The proposed model in (10) has two main components. First, the CF sensitivity to parasitic elements ($\partial \mathrm{CF}/\partial P$), which is different from one cost function to another. Second, the sensitivity of parasitic elements to system (i.e., route) geometries ($\partial P/\partial Ge$).

As for a cost function sensitivity to parasitic elements ($\partial \mathrm{CF}/\partial P$), two sensitivity models are developed. First, the relative cost sensitivity to a parasitic element, which is derived from the relative cost function in (7). Second, the delay cost sensitivity to a parasitic element, which is derived from the delay cost function in (9). Both of them are derived for each parasitic element (P_i) in order to fill the corresponding matrix. As for the sensitivity of parasitic elements to system geometries, it does not rely on the used cost function, and it can be used in (11) regardless of the used cost function. The three sensitivity models are derived as follows.

1) THE RELATIVE COST FUNCTION SENSITIVITY TO A PARASITIC ELEMENT

As for the relative cost function sensitivity (RCF) to a parasitic element, it is obtained by differentiating (7) with a parasitic element (P_i) as below, given that the detailed derivations are found in the Appendix:

$$\frac{\partial \mathrm{RCF}}{\partial P_i} = \frac{\partial}{\partial P_i}\left(\frac{(m_0 - {m'}_0)^2}{{m'}_0{}^2} + \frac{(m_1 - {m'}_1)^2}{{m'}_1{}^2} + \cdots\right), \quad (12)$$

Let

$$\mathrm{RCF}_{\mathrm{mk}} = \frac{(m_k - {m'}_k)^2}{{m'}_k{}^2}. \quad (13)$$

Therefore, by using m_k as an intermediate variable,

$$\frac{\partial \mathrm{RCF}}{\partial P_i} = \sum_{k=0}^{q} \frac{\partial \mathrm{RCF_{mk}}}{\partial m_k} \quad \frac{\partial m_k}{\partial P_i}, \tag{14}$$

where m_k is a certain degree moment at a given node, q is the maximum required degree of moments, and $\mathrm{RCF_{mk}}$ is the relative cost function for a certain moment (i.e., relative moment cost function). This model has two components that include the sensitivity of a relative moment cost function to a circuit moment ($\partial \mathrm{RCF_{mk}}/\partial m_k$) and the sensitivity of a moment to a parasitic element ($\partial m_k/\partial P_i$).

As for the relative moment cost function sensitivity to a circuit moment, it is obtained by differentiating (13) with a moment (m_k) as below:

$$\frac{\partial \mathrm{RCF_{mk}}}{\partial m_k} = 2\,\frac{(m_k - m'_k)}{{m'_k}^2}. \tag{15}$$

As for the sensitivity of each moment to a parasitic element ($\partial m_k/\partial P_i$), it is obtained by differentiating (4) with a parasitic element (P_i) as below, given that the detailed derivations are found in the Appendix:

$$\frac{\partial m_0}{\partial P_i} = -G^{-1}\,\frac{\partial G}{\partial P_i} m_0, \tag{16}$$

and

$$\frac{\partial m_k}{\partial P_i} = -G^{-1}\left(\frac{\partial G}{\partial P_i}\, m_k + \frac{\partial C}{\partial P_i} m_{k-1} + C\,\frac{\partial m_{k-1}}{\partial P_i}\right), \quad k \geq 1 \tag{17}$$

where C is the capacitors matrix, G is the admittance matrix, and m_0 to m_k are circuit moments at a given node. Eventually, the sensitivity of an RCF to a parasitic element (P_i) is obtained by substituting (15), (16), and (17) in (14) as below, given that the detailed derivations are found in the Appendix:

$$\begin{aligned}\frac{\partial \mathrm{RCF}}{\partial P_i} &= 2\,\frac{(m_0 - m'_0)}{{m'_0}^2} \cdot\left(-G^{-1}\,\frac{\partial G}{\partial P_i} m_0\right) + \sum_{k=1}^{q}\left(2\,\frac{(m_k - m'_k)}{{m'_k}^2}\right. \\ &\left.\cdot\left(-G^{-1}\left(\frac{\partial G}{\partial P_i} m_k + \frac{\partial C}{\partial P_i} m_{k-1} + C\,\frac{\partial m_{k-1}}{\partial P_i}\right)\right)\right).\end{aligned} \tag{18}$$

2) THE DELAY COST FUNCTION SENSITIVITY TO A PARASITIC ELEMENT

As for the delay cost function (DCF) sensitivity to a parasitic element (P_i), it is obtained by differentiating (9) with a parasitic element (P_i) as below, given that the detailed derivations are found in the Appendix:

$$\begin{aligned}\frac{\partial DCF}{\partial P_i} &= a_1 \cdot \frac{\partial m_1}{\partial P_i} \\ &+ \sum_{k=2}^{q}\left[a_k\left(\frac{\partial m_k}{\partial P_i} \cdot \frac{1}{{m_1}^{k-1}} + m_k \cdot \frac{(1-k)}{{m_1}^k} \cdot \frac{\partial m_1}{\partial P_i}\right)\right],\end{aligned} \tag{19}$$

where $\partial m_k/\partial P_i$ is obtained in (17).

3) A PARASITIC SENSITIVITY TO LAYOUT GEOMETRIES:

As for parasitic sensitivities to layout geometries ($\partial P/\partial Ge$), they are measured by using the proposed incremental parasitic extraction flow which provides very fast and localized sensitivity numbers. For a certain parasitic element (P_i) and geometry parameter (x_j), the sensitivity is calculated using the below formula:

$$\frac{\partial P_i}{\partial x_j} = \frac{P_i(x_{j+1}) - P_i(x_j)}{x_{j+1} - x_j}, \tag{20}$$

where $P_i(x_{j+1})$ is the value of a parasitic element (P_i) when a geometry x equals x_{j+1}, $P_i(x_j)$ is the value of a parasitic element (P_i) when a geometry x equals x_j.

C. PERFORMANCE ANALYSIS TO IDENTIFY CRITICAL GEOMETRIES

It is very important to understand and analyze the impact of layout geometries on a route's performance. This would help identifying the most sensitive geometries to a route's performance cost function, speeding up the optimization process, and achieving better optimization results.

The performance analysis is performed by using the cost sensitivity to layout geometries model in (11). However, the sensitivity analysis mainly relies on the required performance cost function. In case of performing net matching analysis, the sensitivity models of the relative cost function in (11), (18), and (20) are used. In case of performing a delay analysis, the sensitivity models of the delay cost function in (11), (19), and (20) are used. The higher the sensitivity value, the higher the impact on a route's performance.

As for a general performance analysis, the sensitivity models of the relative cost function may be used in three steps. First, identify the critical routes. Second, create a shortest path route assuming no blockages as a reference route. Third, use (11), (18), and (20) in order to calculate the sensitivity of the RCF to route's geometries using the moments of a shortest path route as reference moments.

D. GEOMETRICAL CONSTRAINTS

Once the most sensitive geometries are selected, they are used as optimization parameters for the routing optimization process; however, this requires maintaining constraints such as the corresponding process design kit (PDK), net blockage constraints, connectivity, and net symmetry constraints. The constraints are obtained using a symbolic template approach.

E. LAYOUT ROUTING OPTIMIZATION PROCESS

The purpose of this step is to minimize a cost function with respect to the most sensitive route's geometries (i.e., coordinates) using a nonlinear programming. The sequential least squares quadratic programming (SLSQP) algorithm is used as a nonlinear programming algorithm because it is an iterative approach for nonlinear optimization problems that accepts multiple constraints. In order to provide degrees of freedom for the routing optimization process, the target routes are fractured into quadrilateral shapes. The number of fractured polygons relies on the required number of degrees of freedom. The fracturing is done in two steps. First, the polygons are scanned in the x direction and fractured vertically. Second, the polygons are scanned in the y direction and fractured horizontally as shown in Figure 10 (b). Each fractured polygon holds four vertices conforming a quadrilateral polygon. The fractured polygons are used to create and evaluate the sensitivity circuit models in (11).

The optimization algorithm is shown in Figure 11. The inputs of the algorithm are: 1) the target routes and 2) the constraints including the new design requirements, whereas the outputs are new routes that are represented by their coordinates. It is worth mentioning that the minimization of a cost function uses the derived sensitivity model, in (11), to create the Jacobean matrix that are used by the nonlinear programming algorithm.

VI. EXPERIMENTAL RESULTS

The testing covered the proposed incremental parasitic capacitance extraction method, the derived sensitivity models, and the proposed parasitic-aware routing optimization method. The testing used Calibre xRC, by Siemens EDA, as a rule-based layout parasitic extraction tool [20], and Eldo platform, by Siemens EDA, as a circuit simulator [38]. Moreover, the testing is performed on Intel Xeon(R) E5-2680, 2 CPUs, 2.50GHz, and 16GB of RAM.

A. TESTING THE PROPOSED INCREMENTAL CAPACITANCE EXTRACTION

The accuracy and runtime of the proposed incremental parasitic capacitance extraction were tested and compared against a full layout parasitic capacitance extraction across two designs that include Ring Oscillator (RO) (7nm) and voltage-controlled oscillator (VCO) (40nm) designs. Calibre xRC, rule-based extractor, is used as an extraction tool for both incremental and full layout parasitic extractions. The testing methodology involves modifying metal shapes for some critical nets. The modifications include deleting, moving, stretching, and adding new metal polygons. Each modified layout is tested by running a full layout parasitic extraction, the proposed incremental extraction, and the incremental extraction without considering the second order capacitances.

```
1  Inputs:
2    Routes[1..n]: List of routes that require optimization, and their count is n.
3    Constraints[1..m]: List of constraints, and their count is m.
4  Output:
5    New_Routes[1..n]: final list of optimized routes
6  Begin
7    Routes = initial current routes.
8    CF = initial values of a cost function across all routes.
9    for i ∈ [1..n]       //foreach route
10     R = Routes[i]   //in case of a delay optimization, it contains one route,
           // in case of a net matching optimization, it contains the two routes.
11     while (optimization is needed) //i.e., gradient is needed
12       Parasitics ← extract_parasitics(R) //extract parasitics of routes in (R)
             //using the proposed incremental parasitic extraction method.
13       dPdGe ← calculate_dPdGe(R, Parasitics) // ∂P/∂Ge using (20)
14       Moments ← calculate_moments(Parasitics) // using (4)
15       dCFdP ← calculate_dCFdP(Moments, Parasitics)
                                  // calculate ∂CF/∂P using (18) or (19)
16       dCFdGe ←calculate_dCFdGe (dCFdP, dPdGe)
                    // calculate ∂CF/∂Ge using (11) to identify the most
                   //sensitive geometries for optimization.
17       R ← optimize_route(R, dCFdGe, Constraints, SLSQP)
                                  // At this point, R holds an updated route.
18       New_Parasitics← extract_parasitics(R) //extract parasitics of new
             //routes in (R) using the proposed incremental parasitic
             //extraction method.
19       New_Moments←calculate_moments(New_Parasitics) //using (4)
20       CF[i]←calculate_cost_value(New_Moments)
                          // using (7) or (9) to calculate new cost value
21     end while
22     New_Routes[i] = R
23   end for
24 End
```

Figure 11. The proposed routing optimization algorithm pseudo code.

As for the RO (7nm), some input and output nets of RO stages were modified in three different ways: 1) modifying two metal layers with 1075 parasitic capacitive elements (i.e., small), 2) modifying three metal layers with 2037 parasitic capacitive elements (i.e., medium), and 3) modifying four metal layers with 3524 parasitic capacitive elements (i.e., large). As shown in Table 3, The maximum relative errors in the three scenarios after applying the proposed incremental parasitic extraction flow as compared to the full parasitic extraction are 0.14%, 0.25%, and 0.5%, respectively. Moreover, the relative speedup of the proposed incremental flow as compared to the full layout extraction in the three scenarios is 40.4, 27.8, and 21.15, respectively. Furthermore, the results show that the consideration of the second order parasitic capacitances has a very small impact on the runtime as compared to the incremental extraction that does not consider the second order parasitic capacitances.

Table 4 shows the simulated RO delay results in case of using the proposed incremental parasitic extraction and the full layout extraction across the three different modification scenarios. The simulation results show that the RO delay relative errors in three modification scenarios are 2.4e-4%, 0.001%, and 0.0057%, respectively.

Table 3. A comparison between the proposed incremental capacitance extraction method and a full layout capacitance extraction using an RO with 31 stages (7nm).

Component	Modification Type		
	Small	Medium	Large
Capacitive elements	1075	2037	3524
Metal layers	2	3	4
Max error of the proposed method	0.14%	0.25%	0.5%
Incremental extraction runtime in seconds (secs)	11 secs	16 secs	21 secs
Full extraction runtime (minutes)	7.4 minutes		
Relative speedup as compared to a full extraction run	40.4	27.8	21.15
Incremental extraction runtime without second order capacitances	10.4 secs	15.1 secs	19.5 secs

Table 4. The simulated delay results of an RO with 31 stages (7nm) in case of using the proposed incremental capacitance extraction method and a full layout capacitance extraction.

	Full layout parasitic extraction	Incremental layout parasitic extraction		
		Small modifications	Medium modifications	Large modifications
RO delay	8.47301ps	8.47303ps	8.4731ps	8.4735ps
Delay relative error as compared to a full layout parasitic extraction		2.4e-4%	0.001%	0.0057%

As for the VCO (40nm), several nets were modified in three different ways: 1) modifying two metal layers with 11768 parasitic capacitive elements (i.e., small), 2) modifying three metal layers with 12794 parasitic capacitive elements (i.e., medium), and 3) modifying four metal layers with 17724 parasitic capacitive elements (i.e., large). As shown in Table 5, the maximum errors in the three scenarios after applying the proposed incremental parasitic extraction flow as compared to the full parasitic extraction are 0.19%, 0.38%, and 0.63%, respectively. Moreover, the relative speedup of the proposed incremental flow as compared to the full layout extraction in the three scenarios is 54.2, 43.07, and 35.1, respectively.

Table 5. A comparison between the proposed incremental capacitance extraction method and a full layout capacitance extraction using a VCO (40nm).

Component	Modification Type		
	Small	Medium	large
Capacitive elements	11768	12794	17724
Metal layers	2	3	4
Max error of the proposed method	0.19%	0.38%	0.63%
Incremental extraction runtime in minutes (mins)	6.67 mins	8.4 mins	10.3 mins
Full extraction runtime	6.03 hours		
Relative speedup as compared to full run	54.2	43.07	35.1
Incremental extraction runtime without second order capacitances	6.35 mins	7.93 mins	9.65 mins

Table 6 shows the simulated VCO performance results in case of using the proposed incremental parasitic extraction and the full layout extraction across the three different modification scenarios. The simulation results show that the impact of the incremental parasitic extraction on the VCO performance is negligible as the center frequency, tuning ratios, and phase noise are almost identical in the case of using the full layout extraction and the incremental layout extraction.

Table 6. The simulated results of VCO (40nm) in case of using the proposed incremental capacitance extraction method and a full layout capacitance extraction.

	Full layout parasitic extraction	Incremental layout parasitic extraction		
		Small modifications	Medium modifications	Large modifications
Center frequency	250GHz	250GHz	250GHz	250.01GHz
Tuning Range (%)	4.4%	4.4%	4.4%	4.4%
Phase noise	-103.5 dBc/Hz at 1MHz	-103.5 dBc/Hz at 1MHz	-103.5 dBc/Hz at 1MHz	-103.51 dBc/Hz at 1MHz

Tables 3-6 summarize the experimental results of the RO (7nm) and VCO (40nm) designs, respectively. As shown in the tables, the proposed incremental extraction flow provides an outstanding accuracy as compared to full extraction with maximum errors < 1% and with huge runtime savings of up to 54X. Furthermore, the simulated results show that the consideration of the second order parasitic capacitances has a very small impact on the runtime as compared to the incremental extraction that does not consider the second order parasitic capacitances.

B. TESTING THE PROPOSED PARASITIC SENSITIVITY MODELS AND ROUTING OPTIMIZATION USING A SIMPLE INTERCONNECT STRUCTURE

The proposed sensitivity models were tested using the interconnect structure shown in Figure 12. This experiment has two purposes. First, it aims to measure the sensitivity of the relative cost function (RCF) to each layout geometry (i.e., coordinate) using (11), where the relative cost function measures V_{out2} moments relative to V_{out1} moments. Second, it aims to match the signal responses at V_{out1} and V_{out2} by optimizing the geometries of V_{out2} route. This is done by using a nonlinear programming to minimize the relative cost function in (7). The circuit response is measured using Eldo circuit simulator [38].

Figure 12 (a) shows the experimental interconnect structure. It contains one input pin, V_{in}, and two output pins that include V_{out1} and V_{out2}. The surrounding dielectric constant is set to 3.9, the elevation of the metal is set to 1 µm, the metal thickness is set to 0.1µm, whereas the sheet resistance is set to 3 Ω/□. The experiment aims to match the signal responses of V_{out1} and V_{out2} without moving the fixed nodes that represent the locations of input and output pins. The route of V_{out2} pin has four obstacles (i.e., blockages). Therefore, V_{out2} route should pass through such obstacles with minimal impact on the performance. The dimensions of the interconnect are shown in Figure 12 (b) and Figure 12 (c). The optimization process used Calibre xRC, rule-based extractor [20], to extract the parasitic elements of the interconnect structure.

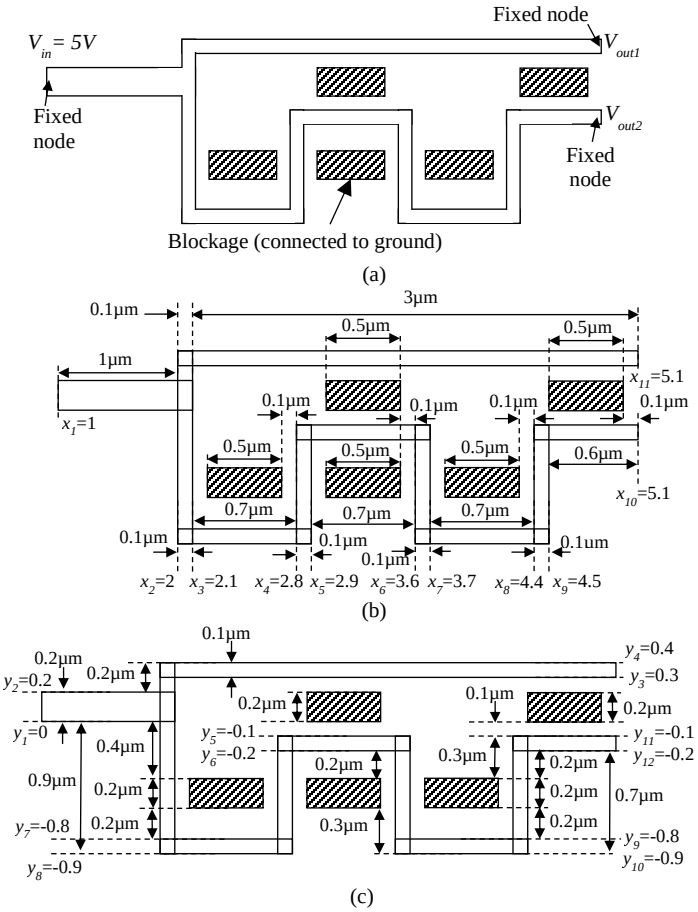


Figure 12. An experimental interconnect structure that is used for verifying the sensitivity circuit models and the optimization algorithm highlighting (a) the nodes, (b) the dimensions in the *x*-direction, and (c) the dimensions in the y-direction, given that all dimensions are in µm.

Table 7 shows the initial values (at the original interconnect dimensions) of the relative cost function sensitivities to the coordinates of V_{out2} route using (11). It is worth mentioning that the sensitivities are nonlinear. Therefore, they are calculated with every optimization iteration.

Table 7. The values of the sensitivity of the relative cost function to each Vout2 coordinate in the experimental interconnect structure.

Sensitivity parameter	Value	Sensitivity parameter	Value
$\partial CF/\partial x2$	181.226	$\partial CF/\partial y5$	-936.95
$\partial CF/\partial x3$	-1395.2	$\partial CF/\partial y6$	937.001
$\partial CF/\partial x4$	1416.3	$\partial CF/\partial y7$	-903.5
$\partial CF/\partial x5$	-1020.98	$\partial CF/\partial y8$	837.4
$\partial CF/\partial x6$	1307.1	$\partial CF/\partial y9$	926.7
$\partial CF/\partial x7$	-1120.98	$\partial CF/\partial y10$	-843.9
$\partial CF/\partial x8$	1902.3	$\partial CF/\partial y11$	884.2
$\partial CF/\partial x9$	-1813.7	$\partial CF/\partial y12$	-809.7

Moreover, a nonlinear programming is applied using SLSQP method in order to minimize the relative cost function. The nonlinear programming uses V_{out2} interconnect geometries (i.e., coordinates) as optimization parameters. Figure 13 shows the optimized interconnect structure. Figure 14 (a) shows the signal responses at V_{out1} and V_{out2} before the optimization process, whereas Figure 14 (b) shows the signal responses after the optimization process. As for the cost values, the value of the relative cost function before the optimization is 0.391, whereas the value of the relative cost function after the optimization is 0.002047.

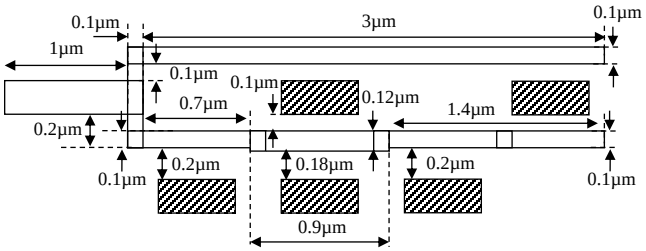


Figure 13. The experimental interconnect structure after the optimization process.

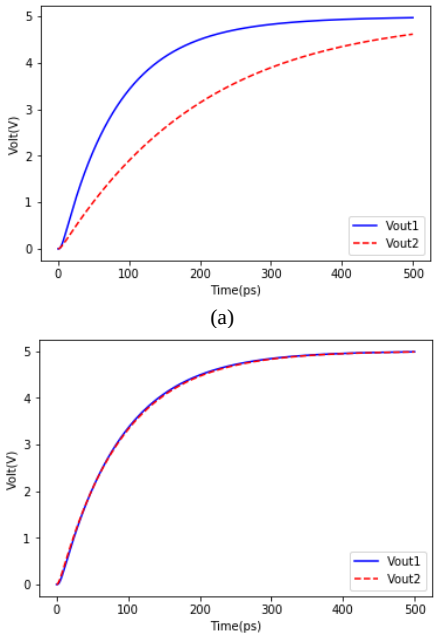


Figure 14. The output response of the experimental interconnect structure at Vout1 and Vout2 (a) before the optimization process and (b) after optimization process.

C. TESTING THE LAYOUT ROUTING OPTIMIZATION METHOD USING CIRCUIT DESIGNS

The routing optimization algorithm, shown in Figure 11, was tested across different designs that include Ring Oscillators (RO) of 7nm process node, folded cascode operational amplifiers with common mode feedback of 65nm process node, and voltage-controlled oscillator (VCO) of 40nm process node. The proposed routing algorithm was integrated in a template-based layout optimization flow, where the proposed routing optimization method replaced the template-based router. The performance of the proposed optimization method was tested in terms of the generated layout performance and the routing optimization runtime. The responses of generated layouts were measured in two steps. First, the parasitic elements of layouts were extracted using Calibre xRC, rule-based extractor [20], in order to be

used as inputs to a circuit simulator. Second, the circuit responses (or performances) were measured using Eldo circuit simulator [38]. Moreover, the simulated circuit responses of the layouts, which were generated by using the proposed optimization method, were compared against the simulated circuit responses of the layouts, which were generated by using the traditional template-based layout optimization method that is described in [13], [14], [25], [27].

1) RING OSCILLATOR (7NM)

As for the RO(7nm), six different RO designs each with 31 stages were tested using 0.75V as an operating voltage. The routing optimization, shown in Figure 11, used the delay cost function in (9) and its corresponding sensitivity circuit models. The testing of the proposed routing optimization algorithm considered two different scenarios of cost functions. The first one considered a cost function with three circuit moments, whereas the second one considered a cost function with five circuit moments. The optimization of RO routes included the input and output pins (i.e., input and output routes) for each RO stage. As shown in Table 8, the proposed routing method managed to achieve better simulated delay results in case of using a cost function with five moments. Moreover, the proposed method managed to reduce the delay of the six RO designs by 9.32%, 10.33%, 10.79%, 9.68%, 10.65%, and 11.1%, respectively, as compared to traditional template-based methods. The relative speedup of the proposed method (using five moments) as compared to the traditional template-based method for the six designs is 9.06, 8.91, 9.48, 8.7, 9.27, and 8.54, respectively.

The reason behind such improvements is that traditional template-based optimization methods use multiple circuit simulations in order to identify the parasitic bounds, and each simulation consumes around 29 minutes. As for the delay improvements, traditional template-based methods use simplified parasitic formulas that are not suitable for advanced process technology nodes, whereas the proposed method uses the proposed incremental extraction method. As for the area, both optimization methods provided almost the same area.

Table 8. The simulated results of the proposed routing optimization method as compared to a traditional template-based method across six different RO (7nm) designs.

	Traditional template-based routing Method		Proposed Method			
			three moments		five moments	
	Delay	Opt. runtime	Delay	Opt. runtime	Delay	Opt. runtime
RO1	7.51ps	4.23 hours	6.96ps	23 minutes	6.81ps	28 minutes
RO2	8.13ps	4.31 hours	7.83ps	24 minutes	7.29ps	29 minutes
RO3	9.27ps	4.11 hours	9.01ps	20 minutes	8.27ps	26 minutes
RO4	8.47ps	4.35 hours	8.07ps	24 minutes	7.65ps	30 minutes
RO5	8.26ps	4.17 hours	8.1ps	22 minutes	7.38ps	27 minutes
RO6	9.10ps	4.41 hours	8.91ps	26 minutes	8.09ps	31 minutes

Opt. runtime: Optimization runtime.

2) FOLDED CASCODE DIFFERENTIAL AMPLIFIER WITH COMMON MODE FEEDBACK (65NM)

Folded cascode differential amplifiers with common mode feedback (CMFB) circuits were tested using three different specifications. The Amplifiers were developed using 65nm process node. Figure 15 shows a block diagram of the amplifiers, whereas Figure 16 shows a schematic circuit design of the folded cascode differential amplifier.

The routing optimization, shown in Figure 11, used the relative cost function in (7) and its corresponding sensitivity circuit models using three and five circuit moments. The optimization was performed over seven routes, Route1 to Route7, as shown in Figure 16. The optimization aimed to match the responses (i.e., net matching) at the output terminal of each two similar routes, where Route1 was matched with Route2, Route3 was matched with Route4, and Route5 was matched with Route6. Moreover, the responses at the output terminals (i.e., t_1 and t_2) of Route7 were also matched.

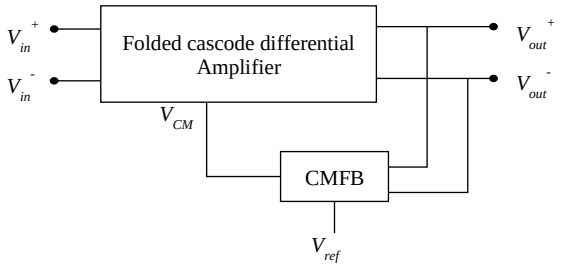


Figure 15. Block diagram of a fully differential folded cascode amplifier with common mode feedback circuit.

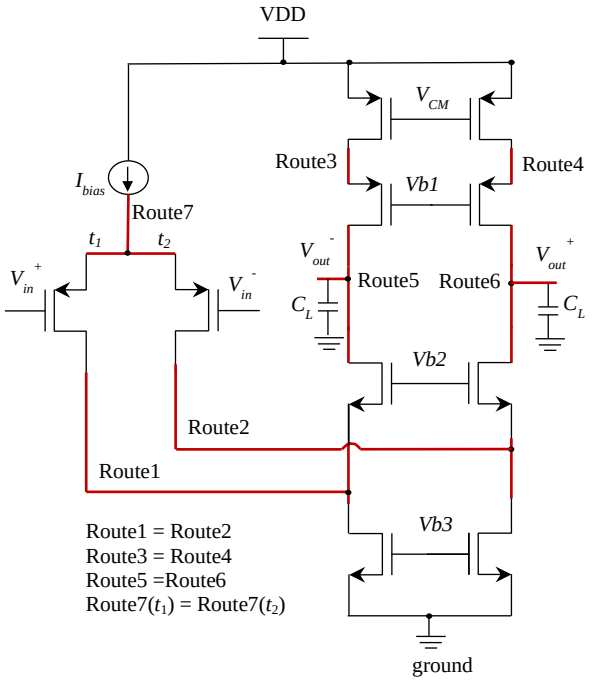


Figure 16. A circuit design of an experimental folded cascode operational amplifier (65nm) showing the optimized routes.

Tables 9, 10, and 11 show the layouts simulated performance results over the three different specifications in the case of: 1) removing interconnect (i.e., routes) parasitic elements, 2) using traditional template-based optimization method, 3) using the proposed optimization method with a cost function of three moments, and 4) using the proposed optimization method with a cost function of five moments. The simulated results show that the proposed optimization method (using five moments) managed to achieve better results that meet the required specifications as compared to the traditional template-based method with minimal impact on the area. Moreover, the optimization runtimes of the proposed method (using five moments) for the three specification scenarios were faster than the traditional template-based method with a speedup of 3.18X, 3.2X, 3.2X, respectively.

Table 9. The simulated results of the proposed routing optimization method as compared to a traditional template-based method over the first layout design of a folded cascode differential amplifier.

	Specs	Without routes parasitic elements	Opt. using Traditional Method	Opt. using the proposed Method	
				three moments	five moments
Gain (dB)	60.0	63.9	61.1	60.3	62.3
GBW (Hz)	350M	365M	361M	352M	362M
PM (º)	60.0	63.7	63.1	61.5	62.9
Output swing (V)	0.8	0.79	0.76	0.78	0.78
Routing optimization runtime			4.3 minutes	1.2 minutes	1.35 minutes
Area (μm)^2			2958	2962	2955

Opt. : Optimization.

Table 10. The simulated results of the proposed routing optimization method as compared to a traditional template-based method over the second layout design of a folded cascode differential amplifier.

	Specs	Without routes parasitic elements	Opt. using The traditional Method	Opt. using the proposed Method	
				three moments	five moments
Gain (dB)	50.0	53.3	51.2	51.1	52.7
GBW (Hz)	300M	313M	309M	303M	311M
PM (º)	50.0	53.1	51.8	50.7	51.7
Output swing (V)	0.9	0.89	0.88	0.89	0.89
Routing optimization runtime			4.5minutes	1.34 minutes	1.41 minutes
Area (μm)^2			3162	3150	3145

Opt. : Optimization.

Table 11. The simulated results of the proposed routing optimization method as compared to a traditional template-based method over the third layout design of a folded cascode differential amplifier.

	Specs	Without routes parasitic elements	Opt. using The traditional Method	Opt. using the proposed Method	
				three moments	five moments
Gain (dB)	60.0	62.7	61.3	60.8	61.7
GBW (Hz)	600M	615M	612M	604M	613M
PM (º)	55.0	59.1	57.1	55.6	58.8
Output swing (V)	0.8	0.79	0.78	0.79	0.79
Routing optimization runtime			4.7 minutes	1.38 minutes	1.47 minutes
Area (μm)^2			3364	3352	3348

Opt. : Optimization.

3) VOLTAGE CONTROLLED OSCILLATOR (40NM)

As for the VCO (40nm), the routing optimization, shown in Figure 11, used the relative cost function in (7) to optimize the matching nets and the delay cost function in (9) to optimize the oscillators nets along with the corresponding sensitivity circuit models. The testing of the proposed routing optimization algorithm considered two different scenarios of cost functions. The first one considered cost functions with three circuit moments, whereas the second one considered cost functions with five circuit moments.

Table 12 shows the simulated performance results of the VCO designs in the case of 1) removing interconnect (i.e., routes) parasitic elements, 2) using traditional template-based optimization method, 3) using the proposed optimization method with a cost function of three moments, and 4) using the proposed optimization method with a cost function of five moments. The simulated results show that the proposed routing optimization algorithm, using cost functions with five moments, managed to optimize the center frequency and the phase noise by percentages of 1.96%, 1.23%, and 7.1%, respectively, as compared to traditional template-based methods. Moreover, the optimization runtime of the proposed method is 6.8X faster than the traditional template-based method.

Table 12. The simulated results of the proposed routing optimization method as compared to a traditional template-based method over VCO (40nm) design.

	Without routes parasitic elements	Opt. using Traditional Method	Opt. using the proposed Method	
			three moments	five moments
Center frequency	269.9GHz	255GHz	256GHz	260GHz
Tuning Range (%)	6.1%	4.4%	4.46%	5.8%
Phase noise	-75.7 dBc/Hz at 1MHz	-87.5 dBc/Hz at 1MHz	-85.2 dBc/Hz at 1MHz	-81.3 dBc/Hz at 1MHz
Routing optimization runtime		3.4 hours	23 minutes	27 minutes
Area (μm)2		10,375	10,380	10,350

Opt. : Optimization.

VII. CONCLUSION AND FUTURE WORK

A parasitic-aware layout routing optimization methodology is developed. Existing layout routing optimization methods suffer from three main problems. First, they rely on many circuit simulations to calculate the parasitic bounds. Second, they rely on either simple parasitic models, which provide poor accuracy, or a full layout extraction, which consumes a lot of time, in order to extract the parasitic elements of a given layout during the optimization process. Third, they do not provide a mechanism to analyze the impact of parasitic elements and corresponding geometries on a system's performance. The proposed methodology overcomes such limitations by providing novel sensitivity circuit models that help circuit designers in analyzing the impact of parasitic elements and corresponding layout geometries on a system's performance. Moreover, it provides a novel incremental

parasitic capacitance extraction methodology that helps in providing a significant speeding up in the optimization runtime with minimal impact on the accuracy as compared to those methods that use a full layout extraction. The proposed optimization method uses a nonlinear programming technique to modify and optimize the problematic routes based on the proposed sensitivity circuit models. The proposed methodology is tested over different ring oscillator designs of 7nm process node and folded cascode differential amplifiers of 65nm process node. The experimental results show that the proposed methodology managed to achieve better accuracy and runtime results as compared to traditional template-based layout routing optimization methods. The proposed methodology managed to identify and optimize the problematic geometries in critical routes with up to 10% improvements in the performance and a speed up of 3 to 9X as compared to traditional template-based methods.

As for future works, the proposed methodology only considers the RC parasitic elements. Hence, their models are appropriate for local interconnect at any frequency and global interconnect at a lower frequency. For high frequency global interconnect, inductance and more complex models need to be included. Therefore, the future work aims to extend this work to consider the different inductance effects.

Appendix

A. MOMENTS SENSITIVITY TO A PARASITIC ELEMENT

The derivations of moments sensitivity to a parasitic element, in (16) and (17), are as below:
By differentiating (4) with a certain parasitic element (P_i) we get:
for m_0: differentiating ($G\,\underline{m_0} = \underline{b}$) with P_i

$$\frac{\partial}{\partial P_i}\left(G\,\underline{m_0}\right) = \frac{\partial}{\partial P_i}\left(\underline{b}\right), \tag{21}$$

Therefore,

$$\frac{\partial G}{\partial P_i}\,\underline{m_0} + G\,\frac{\partial \underline{m_0}}{\partial P_i} = 0. \tag{22}$$

Then,

$$\frac{\partial G}{\partial P_i}\,\underline{m_0} = -G\,\frac{\partial \underline{m_0}}{\partial P_i}, \tag{23}$$

Multiplying both sides by G^{-1}, we get:

$$\frac{\partial \underline{m_0}}{\partial P_i} = -G^{-1}\,\frac{\partial G}{\partial P_i}\underline{m_0}, \tag{24}$$

for m_1: differentiating ($G\,\underline{m_1} + C\,\underline{m_0} = 0$) with P_i

$$\frac{\partial}{\partial P_i}\left(G\,\underline{m_1}\right) + \frac{\partial}{\partial P_i}\left(C\,\underline{m_0}\right) = 0. \tag{25}$$

Therefore, (26)

$$\frac{\partial G}{\partial P_i}\,\underline{m_1} + G\,\frac{\partial \underline{m_1}}{\partial P_i} + \frac{\partial C}{\partial P_i}\,\underline{m_0} + C\,\frac{\partial \underline{m_0}}{\partial P_i} = 0,$$

Eventually,

$$\frac{\partial \underline{m_1}}{\partial P_i} = -G^{-1}\cdot\left(\frac{\partial G}{\partial P_i}\,\underline{m_1} + \frac{dC}{\partial P_i}\,\underline{m_0} + C\,\frac{\partial \underline{m_0}}{\partial P_i}\right). \tag{27}$$

Similarly, for m_2 till m_k, where ($G\,\underline{m_k} + C\,\underline{m_{k-1}} = 0$) :

$$\frac{\partial}{\partial P_i}\left(G\,\underline{m_k}\right) + \frac{\partial}{\partial P_i}\left(C\,\underline{m_{k-1}}\right) = 0.$$
$$\vdots$$
$$\frac{\partial \underline{m_k}}{\partial P_i} = -G^{-1}\cdot\left(\frac{\partial G}{\partial P_i}\,\underline{m_k} + \frac{\partial C}{\partial P_i}\underline{m_{k-1}} + C\frac{\partial \underline{m_{k-1}}}{\partial P_i}\right),\quad k \geq 1, \tag{28}$$

where $\underline{m_k}$ is an n vector of moments and n is the number of nodes in an RC network. This model represents a general model for moments sensitivity to a certain parasitic element. For a certain target node, the moment sensitivity to a parasitic element (P_i) is given by:

$$\frac{\partial m_0}{\partial P_i} = -G^{-1}\,\frac{\partial G}{\partial P_i}m_0, \quad \text{and} \tag{29}$$

$$\frac{\partial m_k}{\partial P_i} = -G^{-1}\cdot\left(\frac{\partial G}{\partial P_i}\,m_k + \frac{\partial C}{\partial P_i}m_{k-1} + C\frac{\partial m_{k-1}}{\partial P_i}\right),\quad k \geq 1, \tag{30}$$

where C is the capacitors matrix, G is the admittance matrix, and m_0 to m_k are circuit moments at a given node.
The parasitic element (P_i) in (29) and (30) can be either a resistive or capacitive element. The derivations for both cases are as follows.

1) MOMENTS SENSITIVITY TO A PARASITIC RESISTIVE ELEMENT:

The moment sensitivity to a parasitic resistive element (R_i) is obtained by substituting a parasitic element parameter (P_i) in (29) and (30) with a resistive element (R_i) as below:

$$\frac{\partial m_0}{\partial R_i} = -G^{-1}\,\frac{\partial G}{\partial R_i}m_0, \quad \text{and} \tag{31}$$

$$\frac{\partial m_k}{\partial R_i} = -G^{-1}\left(\frac{\partial G}{\partial R_i}\,m_k + \frac{\partial C}{\partial R_i}m_{k-1} + C\frac{\partial m_{k-1}}{\partial R_i}\right),\quad k \geq 1. \tag{32}$$

However, some terms might have special values when they are differentiated with a parasitic resistive element (R_i) as below:

$$\frac{\partial C}{\partial R_i} = 0, \tag{33}$$

because C is the capacitance matrix and differentiating it with a resistive element gives zero. Moreover, dG/dR_i is obtained as below:

$$\frac{\partial G}{\partial R_i} = \frac{\partial G}{\partial g_i}\,\frac{\partial g_i}{\partial R_i}, \tag{34}$$

where $g_i = (1/R_i)$. Therefore,

$$\frac{\partial G}{\partial R_i} = \frac{\partial G}{\partial g_i}\,\frac{\partial (1/R_i)}{\partial R_i}, \tag{35}$$

$$\frac{\partial G}{\partial R_i} = -\frac{1}{{R_i}^2}\,\frac{\partial G}{\partial g_i}. \tag{36}$$

As a result, the moments sensitivity to a parasitic resistive element (R_i) is given by:

for m_0:

substitute (36) in (31), we get:

$$\frac{\partial m_0}{\partial R_i} = G^{-1}\,\frac{1}{R^2}\,\frac{\partial G}{\partial g_i} m_0, \tag{37}$$

which represents the moment (m_0) sensitivity to a certain parasitic resistive element at a given node.

for m_k , $k \geq 1$, substitute (33) and (36) in (32), we get:

$$\frac{\partial m_k}{\partial R_i} = -G^{-1}\left(-\frac{1}{{R_i}^2}\,\frac{\partial G}{\partial g_i}\,m_k + C\frac{\partial m_{k-1}}{\partial R_i}\right), \quad k \geq 1, \tag{38}$$

which represents the moment (m_k) sensitivity to a certain parasitic resistive element when $k \geq 1$ at a given node.

2) MOMENTS SENSITIVITY TO A PARASITIC CAPACITIVE ELEMENT:

The moment sensitivity to a parasitic capacitive element (Cc_j) is obtained by substituting a parasitic element parameter (P_i) in (29) and (30) with a capacitive element (Cc_j) as below:

$$\frac{\partial m_0}{\partial Cc_j} = -G^{-1}\,\frac{\partial G}{\partial Cc_j} m_0, \quad \text{and} \tag{39}$$

$$\frac{\partial m_k}{\partial Cc_j} = -G^{-1}\left(\frac{\partial G}{\partial Cc_j}\,m_k + \frac{\partial C}{\partial Cc_j} m_{k-1} + C\frac{\partial m_{k-1}}{\partial Cc_j}\right), \quad k \geq 1. \tag{40}$$

However, some terms might have special values when they are differentiated with a parasitic capacitive element (Cc_j) as below:

$$\frac{\partial G}{\partial Cc_j} = 0, \tag{41}$$

because G is the admittance matrix and differentiating it with a capacitive element gives zero.

As a result, the moments sensitivity to a parasitic capacitive element (Cc_j) is given by:

for m_0, substitute (41) in (39), we get:

$$\frac{\partial m_0}{\partial Cc_j} = 0, \tag{42}$$

which represents the moment (m_0) sensitivity to a certain parasitic capacitive element at a given node.

for m_k , $k \geq 1$, substitute (41) in (40), we get:

$$\frac{\partial m_k}{\partial Cc_j} = -G^{-1}\left(\frac{\partial C}{\partial Cc_j} m_{k-1} + C\frac{\partial m_{k-1}}{\partial Cc_j}\right), k \geq 1 \tag{43}$$

which represents the moment (m_k) sensitivity to a certain parasitic capacitive element when $k \geq 1$ at a given node.

B. RELATIVE COST FUNCTION SENSITIVITY TO A PARASITIC ELEMENT

The derivations of the relative cost function sensitivity to a parasitic element, in (18), are as below:

Assuming two systems, the output response of the first system is given by:

$$S1(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \cdots, \tag{44}$$

whereas the output response of the second system is given by:

$$S2(s) = m'_0 + m'_1 s + m'_2 s^2 + m'_3 s^3 + \cdots. \tag{45}$$

Therefore, the relative cost function (RCF) between the two systems is given by:

$$\mathrm{RCF} = \sum_{i=0}^{q} \frac{(m_i - m'_i)^2}{{m'_i}^2}, \tag{46}$$

where q represents the required order of circuit moments.

differentiating (46) with a parasitic element (P_i) gives:

$$\frac{\partial \mathrm{RCF}}{\partial P_i} = \frac{\partial}{\partial P_i}\left(\frac{(m_0 - m'_0)^2}{{m'_0}^2} + \frac{(m_1 - m'_1)^2}{{m'_1}^2} + \cdots\right). \tag{47}$$

Let

$$\mathrm{RCF_{mk}} = \frac{(m_k - m'_k)^2}{{m'_k}^2}. \tag{48}$$

Therefore,

$$\frac{\partial \mathrm{RCF}}{\partial P_i} = \frac{\partial}{\partial P_i}\left(\mathrm{RCF_{m0}} + \mathrm{RCF_{m1}} + \cdots\right), \tag{49}$$

Use m_0 to m_k as intermediate variables for differentiation, we get:

$$\frac{\partial \mathrm{RCF}}{\partial P_i} = \frac{\partial \mathrm{RCF_{m0}}}{\partial m_0}\,\frac{\partial m_0}{\partial P_i} + \frac{\partial \mathrm{RCF_{m1}}}{\partial m_1}\,\frac{\partial m_1}{\partial P_i} + \cdots. \tag{50}$$

As a result,

$$\frac{\partial \mathrm{RCF}}{\partial P_i} = \sum_{k=0}^{n} \frac{\partial \mathrm{RCF_{mk}}}{\partial m_k}\,\frac{\partial m_k}{\partial P_i}. \tag{51}$$

This model has two components. The first component is ($\partial RCF_{mk}/\partial m_k$). It is obtained by differentiating (48) with a certain moment (m_k) as below:

$$\frac{\partial \mathrm{RCF_{mk}}}{\partial m_k} = 2\,\frac{(m_k - m'_k)}{{m'_k}^2}, \tag{52}$$

The second component ($\partial m_k/\partial P_i$) is already obtained in (29) and (30). By substituting (29), (30) and (52) in (51), we get:

$$\frac{\partial \mathrm{RCF}}{\partial P_i} = 2\,\frac{(m_0 - m'_0)}{{m'_0}^2} \cdot \left(-G^{-1}\,\frac{\partial G}{\partial P_i} m_0\right) + \sum_{k=1}^{n} \left(2\,\frac{(m_k - m'_k)}{{m'_k}^2} \cdot \left(-G^{-1}\left(\frac{\partial G}{\partial P_i} m_k + \frac{\partial C}{\partial P_i} m_{k-1} + C\,\frac{\partial m_{k-1}}{\partial P_i}\right)\right)\right), \quad (53)$$

which represents the relative cost function (RCF) sensitivity to a certain parasitic element (P_i) at a given node.

C. DELAY COST FUNCTION SENSITIVITY TO A PARASITIC ELEMENT

The derivations of the delay cost function sensitivity to a parasitic element, in (19), are as below:
The delay cost function (DCF) is given by, based on [37]:

$$\mathrm{DCF} = a_1 \cdot m_1 + a_2 \cdot \frac{m_2}{m_1} + a_3 \cdot \frac{m_3}{{m_1}^2} + \cdots + a_q \cdot \frac{m_q}{{m_1}^{(q-1)}}, \quad (54)$$

differentiating (54) with a parasitic element (P_i) gives:

$$\frac{\partial \mathrm{DCF}}{\partial P_i} = \frac{\partial}{\partial P_i}\left(a_1 \cdot m_1 + a_2 \cdot \frac{m_2}{m_1} + a_3 \cdot \frac{m_3}{{m_1}^2} + \cdots + a_q \cdot \frac{m_q}{{m_1}^{(q-1)}}\right). \quad (55)$$

Therefore,

$$\frac{\partial \mathrm{DCF}}{\partial P_i} = a_1 \cdot \frac{\partial m_1}{\partial P_i} + a_2 \cdot \left(\frac{\partial m_2}{\partial P_i}\frac{1}{m_1} + m_2\,(-{m_1}^{-2})\frac{\partial m_1}{\partial P_i}\right) + \cdots + a_q \cdot \left(\frac{\partial m_q}{\partial P_i}\frac{1}{{m_1}^{k-1}} + m_q\,(-(q-1){m_1}^{-q})\frac{\partial m_1}{\partial P_i}\right). \quad (56)$$

As a result,

$$\frac{\partial \mathrm{DCF}}{\partial P_i} = a_1 \cdot \frac{\partial m_1}{\partial P_i} + \sum_{k=2}^{q}\left[a_k \left(\frac{\partial m_k}{\partial P_i} \cdot \frac{1}{{m_1}^{k-1}} + m_k \cdot \frac{(1-k)}{{m_1}^{k}} \cdot \frac{\partial m_1}{\partial P_i}\right)\right], \quad (57)$$

which represents the delay cost function sensitivity to a parasitic element (P_i) at a given node.

References

[1] M. Bohr, "The new era of scaling in an SoC world," in *2009 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*, Feb. 2009, pp. 23–28. doi: 10.1109/ISSCC.2009.4977293.

[2] "New Parasitic Extraction Requirements In Custom Design For The Next Wave Of SoCs," *Semiconductor Engineering*, Jan. 30, 2020. https://semiengineering.com/new-parasitic-extraction-requirements-in-custom-design-for-the-next-wave-of-socs/ (accessed Jun. 21, 2021).

[3] J. H.-C. Chen, T. E. Standaert, E. Alptekin, T. A. Spooner, and V. Paruchuri, "Interconnect performance and scaling strategy at 7 nm node," in *IEEE International Interconnect Technology Conference*, May 2014, pp. 93–96. doi: 10.1109/IITC.2014.6831843.

[4] A. Domic, "Layout synthesis of MOS digital cells," in *27th ACM/IEEE Design Automation Conference*, Jun. 1990, pp. 241–245. doi: 10.1109/DAC.1990.114861.

[5] Zheng Liu and Lihong Zhang, "Performance-constrained parasitic-aware retargeting and optimization of analog layouts," in *2009 Canadian Conference on Electrical and Computer Engineering*, May 2009, pp. 1194–1197. doi: 10.1109/CCECE.2009.5090314.

[6] M. P. Lin, Y. Chang, and C. Hung, "Recent research development and new challenges in analog layout synthesis," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan. 2016, pp. 617–622. doi: 10.1109/ASPDAC.2016.7428080.

[7] M. D. Moffitt, "Global routing revisited," in *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, Nov. 2009, pp. 805–808. doi: 10.1145/1687399.1687549.

[8] N. Gockel, R. Drechsler, and B. Becker, "A multi-layer detailed routing approach based on evolutionary algorithms," in *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*, Apr. 1997, pp. 557–562. doi: 10.1109/ICEC.1997.592373.

[9] M. Ranjan, W. Verhaegen, A. Agarwal, H. Sampath, R. Vemuri, and G. Gielen, "Fast, layout-inclusive analog circuit synthesis using pre-compiled parasitic-aware symbolic performance models," in *Automation and Test in Europe Conference and Exhibition Proceedings Design*, Feb. 2004, vol. 1, pp. 604-609 Vol.1. doi: 10.1109/DATE.2004.1268911.

[10] N. Lourenço, R. Martins, and N. Horta, "Layout-aware sizing of analog ICs using floorplan amp; routing estimates for parasitic extraction," in *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, Mar. 2015, pp. 1156–1161.

[11] B. Cardoso, R. Martins, N. Lourenço, and N. Horta, "AIDA-PEx: Accurate parasitic extraction for layout-aware analog integrated circuit sizing," in *2015 11th Conference on Ph.D. Research in Microelectronics and Electronics (PRIME)*, Jun. 2015, pp. 129–132. doi: 10.1109/PRIME.2015.7251351.

[12] S. Bhattacharya, N. Jangkrajarng, and C.-R. Shi, "Template-driven parasitic-aware optimization of analog integrated circuit layouts," in *Proceedings. 42nd Design Automation Conference, 2005.*, Jun. 2005, pp. 644–647. doi: 10.1145/1065579.1065748.

[13] N. Jangkrajarng, L. Zhang, S. Bhattacharya, N. Kohagen, and C.-R. Shi, "Template-Based Parasitic-Aware Optimization and Retargeting of Analog and RF Integrated Circuit Layouts," in *2006 IEEE/ACM International Conference on Computer Aided Design*, Nov. 2006, pp. 342–348. doi: 10.1109/ICCAD.2006.320056.

[14] L. Zhang, N. Jangkrajarng, S. Bhattacharya, and C.-R. Shi, "Parasitic-Aware Optimization and Retargeting of Analog Layouts: A Symbolic-Template Approach," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 5, pp. 791–802, May 2008, doi: 10.1109/TCAD.2008.917594.

[15] H. Habal and H. Graeb, "Constraint-Based Layout-Driven Sizing of Analog Circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 8, pp. 1089–1102, Aug. 2011, doi: 10.1109/TCAD.2011.2158732.

[16] A. Toro-Frías, R. Castro-López, E. Roca, and F. V. Fernández, "An automated layout-aware design flow," in *2012 International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, Sep. 2012, pp. 73–76. doi: 10.1109/SMACD.2012.6339420.

[17] R. Castro-Lopez, O. Guerra, E. Roca, and F. V. Fernandez, "An Integrated Layout-Synthesis Approach for Analog ICs," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 7, pp. 1179–1189, Jul. 2008, doi: 10.1109/TCAD.2008.923417.

[18] M. S. Abouelyazid, S. Hammouda, and Y. Ismail, "Connectivity-Based Machine Learning Compact Models for Interconnect Parasitic Capacitances," in *2021 ACM/IEEE 3rd Workshop on Machine Learning for CAD (MLCAD)*, Aug. 2021, pp. 1–6. doi: 10.1109/MLCAD52597.2021.9531300.

[19] M. S. Abouelyazid, S. Hammouda, and Y. Ismail, "Fast and Accurate Machine Learning Compact Models for Interconnect Parasitic Capacitances Considering Systematic Process Variations," *IEEE Access*, vol. 10, pp. 7533–7553, 2022, doi: 10.1109/ACCESS.2022.3142330.

[20] "Calibre xRC | Siemens Digital Industries Software." https://eda.sw.siemens.com/en-US/ic/calibre-design/circuit-verification/xrc (accessed Apr. 11, 2021).

[21] R. M. Smey, B. Swartz, and P. H. Madden, "Crosstalk reduction in area routing," in *Automation and Test in Europe Conference and Exhibition 2003 Design*, Mar. 2003, pp. 862–867. doi: 10.1109/DATE.2003.1253714.

[22] A. Bhaduri and R. Vemuri, "Parasitic-Aware and Moment-driven Constraint Satisfying Non-Linear Routing Methodology," in *2006 49th IEEE International Midwest Symposium on Circuits and Systems*, Aug. 2006, vol. 2, pp. 84–88. doi: 10.1109/MWSCAS.2006.382214.

[23] A. Bhaduri and R. Vemuri, "Parasitic aware routing methodology based on higher order RLCK moment metrics," in *19th International Conference on VLSI Design held jointly with 5th International Conference on Embedded Systems Design (VLSID'06)*, Jan. 2006, p. 6 pp.-. doi: 10.1109/VLSID.2006.129.

[24] A. Bhaduri and R. Vemuri, "Inductive and capacitive coupling aware routing methodology driven by a higher order RLCK moment metric," in *Design, Automation and Test in Europe*, Mar. 2005, pp. 922-923 Vol. 2. doi: 10.1109/DATE.2005.182.

[25] Zheng Liu and L. Zhang, "A performance-constrained template-based layout retargeting algorithm for analog integrated circuits," in *2010 15th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan. 2010, pp. 293–298. doi: 10.1109/ASPDAC.2010.5419880.

[26] R. Martins, N. Lourenço, and N. Horta, "LAYGEN II—Automatic Layout Generation of Analog Integrated Circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 32, no. 11, pp. 1641–1654, Nov. 2013, doi: 10.1109/TCAD.2013.2269050.

[27] F. A. Naguib, S. Ahmed, S. Hamed, and M. Dessouky, "Expert Guided Analog Layout Placement and Routing Automation for Deep Nanotechnologies," in *2020 37th National Radio Science Conference (NRSC)*, Sep. 2020, pp. 240–247. doi: 10.1109/NRSC49500.2020.9235112.

[28] G. Liu, W. Zhu, S. Xu, Z. Zhuang, Y.-C. Chen, and G. Chen, "Efficient VLSI routing algorithm employing novel discrete PSO and multi-stage transformation," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16, 2020, doi: https://doi.org/10.1007/s12652-020-02659-8.

[29] R. Kastner, E. Bozorgzadeh, and M. Sarrafzadeh, "Coupling aware routing," in *Proceedings of 13th Annual IEEE International ASIC/SOC Conference (Cat. No.00TH8541)*, Sep. 2000, pp. 392–396. doi: 10.1109/ASIC.2000.880770.

[30] T.-Y. Ho, Y.-W. Chang, S.-J. Chen, and D. T. Lee, "A fast crosstalk- and performance-driven multilevel routing system," in *ICCAD-2003. International Conference on Computer Aided Design (IEEE Cat. No.03CH37486)*, Nov. 2003, pp. 382–387. doi: 10.1109/ICCAD.2003.159715.

[31] S. J. Patel and R. A. Thakker, "Parasitic-Aware Automatic Analog CMOS Circuit Design Environment," in *2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID)*, Jan. 2019, pp. 245–250. doi: 10.1109/VLSID.2019.00061.

[32] S. J. Patel and R. A. Thakker, "Parasitic Aware Automatic Analog CMOS Circuit Design Environment Using ABC Algorithm," in *2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID)*, Jan. 2018, pp. 445–446. doi: 10.1109/VLSID.2018.105.

[33] Y. I. Ismail, "Improved model-order reduction by using spacial information in moments," *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 11, no. 5, pp. 900–908, Oct. 2003, doi: 10.1109/TVLSI.2003.817138.

[34] W. Yu, M. Song, and M. Yang, "Advancements and Challenges on Parasitic Extraction for Advanced Process Technologies," in *2021 26th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan. 2021, pp. 841–846.

[35] W. PINELLO, A. Nieuwoudt, M. DRUT, and B. Qiu, "Determining eco aggressor nets during incremental extraction," US20160350470A1, Dec. 01, 2016 Accessed: Mar. 07, 2021. [Online]. Available: https://patents.google.com/patent/US20160350470A1/en

[36] K. Kalafala *et al.*, "Incremental parasitic extraction for coupled timing and power optimization," US9858383B2, Jan. 02, 2018 Accessed: Mar. 04, 2021. [Online]. Available: https://patents.google.com/patent/US9858383B2/en

[37] Y. I. Ismail and C. S. Amin, "Computation of signal threshold crossing times directly from higher order moments," in *IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004.*, Nov. 2004, pp. 246–253. doi: 10.1109/ICCAD.2004.1382581.

[38] "Eldo Platform," *Siemens Digital Industries Software*. https://eda.sw.siemens.com/en-US/ic/eldo/ (accessed Jul. 20, 2022).

MOHAMED SALEH ABOUELYAZID (Member, IEEE) received the B.Sc. and M.Sc. degrees in electronics engineering from the Faculty of Engineering, Cairo University, Egypt, in 2009 and 2013, respectively, and the Ph.D. degree from the School of Sciences and Engineering, The American University in Cairo, Egypt, in 2022. He has been working with Siemens EDA, Egypt, since 2009, where he is currently a Research and Development Software Staff Engineer in layout parasitic extraction team, Calibre design to silicon. Mohamed has a strong background in circuit verification, data modeling, machine learning, and electronic design automation.

SHERIF HAMMOUDA received the B.Sc. and M.Sc. degrees in electrical engineering from Ain Shams University, Cairo, Egypt, in 1996 and 2001, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Calgary, AB, Canada, in 2006. He has been with Siemens EDA, Cairo, since 1997, where he is also the Engineering Director managing research and development activities for Calibre design to silicon activities in Cairo. He holds a U.S. patent and coauthored a number of publications in various conferences focusing on EDA for IC Design.

YEHEA ISMAIL (Fellow, IEEE) is currently the Director of the Nanoelectronics and Devices Center, The American University in Cairo and Zewail City. He was a Tenured Professor with Northwestern University, USA, from 2000 to 2011. He has published more than 450 papers in top refereed journals and conferences and many patents. He has coauthored seven books, including *On-Chip Inductance in High Speed Integrated Circuits*, *Handbook on Algorithms for VLSI Physical Design*, *Temperature-Aware Computer Architecture*, *Arbitrary Modeling of TSVs for 3D Integrated Circuits*, and *Circuit Design Techniques for Microscale Energy Harvesting Systems*. He has many patents in the area of high-performance circuits and interconnect design and modeling. His work is some of the most highly cited in the VLSI area and is extensively used by industry. He has several awards, such as the USA National Science Foundation Career Award, the IEEE CAS Outstanding Author Award, the Best Teacher Award at Northwestern University, and many other best teaching awards and best paper awards. He

is a Distinguished Lecturer of IEEE CASS. He was the Editor-in-Chief of the IEEE TRANSACTION ON VERY LARGE SCALE INTEGRATION (TVLSI) (2011_2015) and the Chair Elect of the IEEE VLSI Technical Committee. He is on the editorial board of the *Journal of Circuits, Systems, and Computers*, was on the editorial board of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS_I: FUNDAMENTAL THEORY AND APPLICATIONS, and a Guest Editor for a special issue of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS on ``On-Chip Inductance in High Speed Integrated Circuits." He has also chaired many conferences, such as GLSVLSI, IWSOC, and ISCAS. He was the Chief Scientist of the Innovation and Entrepreneurship Center, Ministry of Communications and Information Technology, Egypt.