A Stack-Based Routing Methodology For Nanometric Analogue CMOS Devices

Stephanie Youssef, Damien Dupuis, Ramy Iskander, Marie-Minerve Louërat

A Stack-Based Routing Methodology For Nanometric Analogue CMOS Devices

Stephanie YOUSSEF, Damien DUPUIS, Ramy ISKANDER and Marie-Minerve LOUERAT
LIP6 Laboratory, University of Pierre and Marie-Curie, France
Email: stephanie.youssef@lip6.fr

***Abstract*— In this paper, we present a nanometric layout generation tool for analogue building blocks called *devices*. We focus on the procedural routing methods inside devices. A device may have one or more folded transistors' fingers merged into at least one *stack* depending on the chosen layout style. We present two routing methods: *intra*-stack and *inter*-stack to ease the routing of the wired segments. Taking advantage of both routing methods, the layout generation tool provides a range of transistor folding to respect the designer-defined constraints (either electrical or physical). Both routing methods are used to generate different layout styles. The layout generation for a differential pair device is illustrated using four layout styles: interdigitated, mirror, 2D common-centroid and M2 modules.**
Keywords: Migration, Layout Generation, Routing, Stress effects

I. INTRODUCTION

A lot of studies have targeted the automation of the layout generation of analogue circuits [1]–[12]. Their common goal is to improve the analogue IC design cycle [5], [13] to get closer to the efficiency of the digital one. The challenge is that the layout generation must support the evolution of CMOS down to the deep submicron technologies while meeting a wide range of constraints dealing with power supply, power consumption, biasing, high speed, area, symmetry, matching, coupling. With the migration to deep-sub-micron (DSM) technologies, a new constraint related to the MOS technology process has to be taken into consideration.

The DSM technologies use Shallow Trench Isolation (STI) for its accurate dimension control when compared with LOCOS isolation [14]. STI is implemented in the form of trenches etched into the wafer and filled with silicon dioxide to isolate the active area of the transistors. Although STI provides some degree of latch-up protection [15], this isolation technique induces mechanical stress on the transistor and hence degrades its performance [16]–[18]. As shown in [19], this mechanical stress is highly dependent on the layout style being used. To reduce the impact of mechanical stress, the layout must be designed so that all the transistors of the device are affected in the same way.

Analogue circuits are much more sensitive to the layout than the digital ones and their performance may be affected easily. Hence it is required to carefully control the routing from the transistor level in order to respect the different constraints that highly depend on the application. Examples are the parasitic capacitance and resistance induced by the routing wires to avoid the cross talk between wires [1], [6] or specific RF constraints [11]. The close link between the physical realization and the electrical behavior leads the circuit designer to compare several layout styles with appropriate routing of the basic cell (*device*) and choose the most suitable one, according to the specification of the whole circuit.

In this paper, we will focus on basic building blocks called *devices* to develop a library of parametrized analogue cells that support different layout styles (interdigitated, mirror, M2 module and 2D common-centroid). We introduce a parametric generic routing methodology. This methodology provides the designer a procedural way to perform a symmetrical routing and therefore a symmetrical layout. Its main advantage is to handle a large set of device topologies based on a generic Python method driven with a few input parameters. The resulting procedural description of a device may handle several technology processes and several aspect ratio. Based upon these generic devices, parametrized analogue blocks can be designed as reusable blocks matching use-case constraints (i.e. low-power, RF, ...). Taking advantage of such reusable blocks, the whole analogue IC design cycle may be improved.

The remainder of this paper is organized as follows. Section II introduces the context of the layout generation tool, then presents the definitions of the device and the stack object. Section III introduces the two routing methodologies. Section IV illustrates the routing results for a differential pair device with four layout styles: Interdigitated, mirror, 2D-common centroid and M2-Module styles. Finally, we conclude in section V.

II. LAYOUT OF THE DEVICE

A. Proposed Layout Generation Tool

The flow, still commonly used, for an analogue circuit consists of laborious iteration loops. Each iteration is composed of several steps: circuit sizing, layout generation, parasistics extraction and performance evaluation. While performances are not satisfactory, the loop is repeated. It is time consuming and subject to human errors. To speed up the design cycle, layout oriented methodolody were proposed [4], [6], [7], [9], [10], [20], [21].

[4] showed the advantage of providing a two ways communication between the sizing and layout generation as shown in Fig. 1. The idea is that the sizing tool provides the layout generation tool with the electrical parameters of the transistor such as the width (W), length (L), number of fingers (NF), etc... Once the layout is generated, the layout

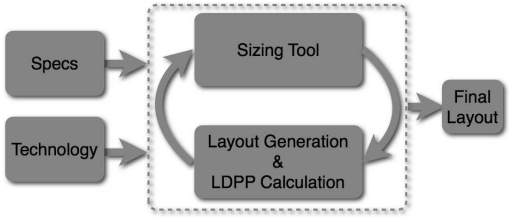

Fig. 1. Applied design flow.

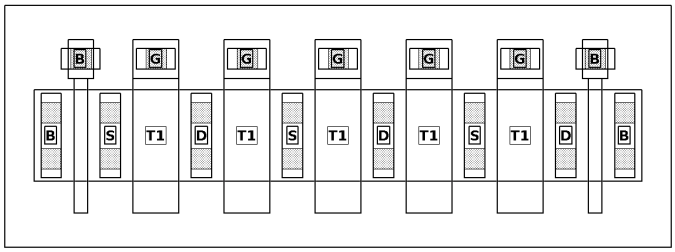

Fig. 2. Layout stack example $W = 2.0\mu$m, $L = 0.2\mu$m, $NFs = 7$, $Type = NMOS$ and $NB_{dummies} = 1$.

tool sends back the layout-dependent parasitic parameters such as the drain and source areas and perimeters, the stress effect parameters, etc... to re-evaluate the performance. This internal loop is repeated several times, with minimal designer intervention, till the targeted specifications are achieved. The final layout is then realized. This methodology minimizes the design time and possible errors. It has been implemented into our framework which is dedicated to analogue synthesis and technology migration for mixed-signal circuits in nanometric technologies. Our layout generation tool allows the generation of parametrized and shaped layouts, with different analogue dedicated layout styles [8], [12], [22].

B. The Device Definition

A device is defined as an atomic analogue function realized by a small set of transistors. The motivation to build a device is the following: the analogue electrical behavior of the set of transistors requires a dedicated layout with strong geometrical and robustness constraints. Therefore the layout of the transistors' set has to be designed as a whole. A typical library of analogue devices contains: a folded transistor, a differential pair, a current mirror and a cross coupled pair. Each device may have different layout styles to match use-case constraints. Here we will study four different styles of the differential pair: interdigitated, mirror, 2D common-centroid and M2 module [12] to illustrate our two routing methodologies.

C. The Stack Object

1) Stack object: Transistor folding technique is commonly used in analogue circuits to control the aspect ratio of the device layout and to reduce parasitic capacitance and gate resistance [3], [23] while allowing more accurate geometries and providing better electrical performance. Interdigitation and mirror styles are usually used to equally distribute process gradients along the device. Since the stack structure is a common feature of the analogue device, we have defined a 'Stack' object [19]. To create the layout of a complete stack, the designer of parametrized analogue devices simply calls the `createStack()` method with well specified input parameters as shown in Fig. 2. A device layout is made of a set of horizontal or vertical stacks.
The input parameters of the `createStack()` method are :

- **Type**: The type of the transistor NMOS or PMOS.
- **W**: The overall width of the transistor.
- **L**: The length of each finger (except dummies).
- **NFs**: The number of stack's fingers (including dummies).
- $\mathbf{NB_{dum}}$: The number of dummies at each stack ends (same for both ends).

Note that we assume that each finger of the stack, except dummies, has the same width and length. Based on this `createStack()` method, a dedicated Python API has been introduced to provide the designer with the possibility to describe a device as a stack arrangement. The generated layout passes design rule checking.

2) Layout Dependant Parameter Computation Methods: Three Layout Dependant Parameter (LDP) computation methods have been introduced in the stack object, described in Python. The first one computes the diffusion parameters like the area and perimeter of the drain and source zones. The second one computes the stress effect parameters introduced in BSIM4 [24] to model nanometric DSM effects. The third one computes the parasitic capacitance and resistance due to the routing. A dedicated Python API has been introduced to provide the device designer with the LDP values of a stack. Since each device uses at least one stack, the device designer may compute easily the device LDP from the stack LDP.

III. Routing Methodology

A. Intra-Stack Routing Methodology

The intra-stack routing addresses the routing between transistors' connectors inside the same stack. The idea is to reserve track(s) for routing each net of the stack according to the chosen pattern as shown in Fig. 3. Each device net is routed, one after the other, i.e. Drain (D), Bulk (B), Gate (G), Source (S), etc... Different patterns are supported: line, comb, serpentine or mixed between these three patterns. The goal is to set up a simple method that takes advantage of the regular structure of the stack while supporting its shape variation resulting from its parameter variation: transistor width, length and number of fingers. The designer should specify only a set of combination of '1's and '0's where '1' indicates a top track and '0' indicates a bottom track.

This combination is called *pattern*. For example if the designer specify '1' pattern to a net, this net will be routed in a top line pattern like Fig. 4.a, if the designer specify '0' pattern to a net, this net will be routed in a bottom line pattern like Fig. 4.b, if the designer specify '10' pattern to a net, this

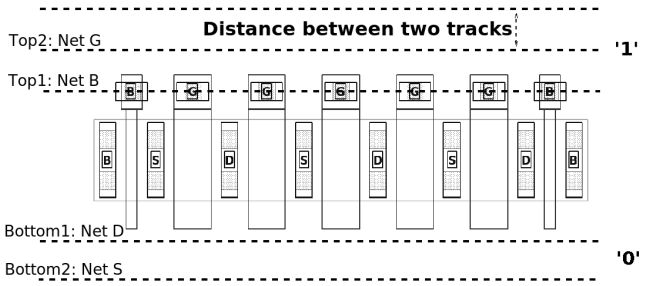

Fig. 3. One stack with reserved tracks.

net will be routed in a serpentine pattern like Fig. 4.c. Also the designer can choose any other random pattern like in Fig. 4.d. The first routed net has the first reserved track.

The advantage of the method is that the designer can specify any pattern and the router draws automatically the routing segments at the appropriate reserved tracks that passes the DRC check.

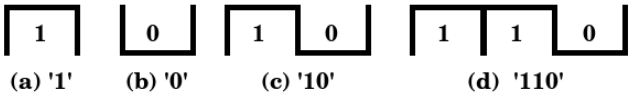

Fig. 4. Different inter-stack routing patterns.

B. Inter-Stack Routing Methodology

The inter-stack routing method addresses the routing between two routing segments of the same net belonging to two different stacks of the same device. This method `routeSegments()` performs segment to segment routing. It supports different patterns like Line, L-shape, U-shape or Z-shape. Unlike the intra-stack routing methodology, the inter-stack methodology allows routing around and between stacks.

This method has four parameters: the net to be routed, the first segment (Segment 1) and the second segment (Segment 2) to be routed and some coordinates. Depending on the coordinates specified by the device designer, this method is able to deduce the suitable pattern.

The special case of two horizontal segments routing is detailed in the following. Eight different parameters configurations are considered. In each case, the designer specifies at least the value of $X1$ and $X2$ belonging respectively to Segment 1 and Segment 2. Then, depending upon the remaining parameter, the method selects automatically the suitable shape:

1) $dx = 0$, L-Shape is deduced (Fig.5).
2) $0 < dx < X2 - X1$, Z-Shape is deduced (Fig.6).
3) $dx > X2 - X1$, U-Shape is deduced (Fig.7).
4) $0 > dx$, U-Shape is deduced (Fig.8).
5) $dy = 0$, L-Shape is deduced (Fig.9).
6) $Y2 - Y1 < dy < 0$, Z-Shape is deduced (Fig.10).
7) $dy < Y2 - Y1$, U-Shape is deduced (Fig.11).
8) $dy > 0$, U-Shape is deduced (Fig.12).

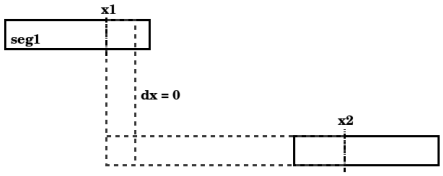

Fig. 5. L-Shape routing, dx specified.

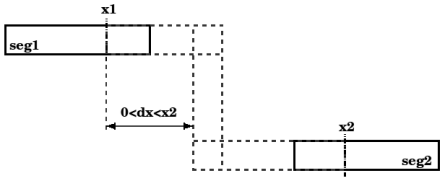

Fig. 6. Z-Shape routing, dx specified.

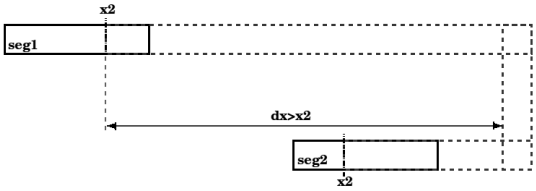

Fig. 7. U-Shape routing, dx specified.

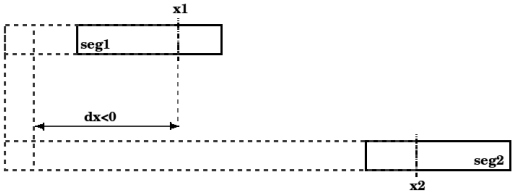

Fig. 8. U-Shape routing, dx specified.

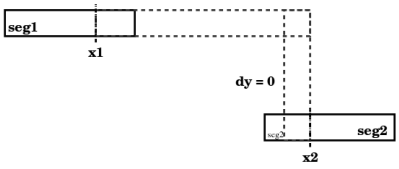

Fig. 9. L-Shape routing, dy specified.

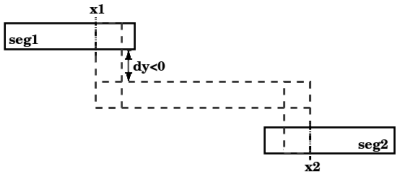

Fig. 10. Z-Shape routing, dy specified.

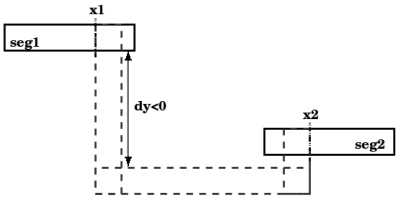

Fig. 11. U-Shape routing, dy specified.

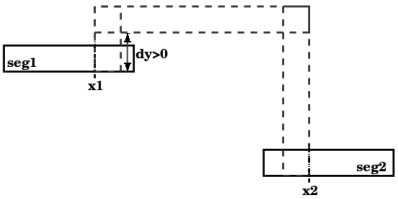

Fig. 12. U-Shape routing, dy specified.

Similar algorithms are introduced to route two vertical segments or one vertical segment with one horizontal segment.

IV. Results

In the following examples, we consider the differential pair device shown in Fig. 13 in CMOS 65nm technology. This device is composed of two transistors that have to be matched. Among the several ways to match and organize the two transistors in one stack, we consider the interdigitated and the mirror styles [22] to illustrate the intra-stack routing methodology. The multi-stack organization as well as the the inter-stack routing will be illustrated by the M2 Module and the 2D common-centroid styles [22]. The transistor $T1$ has top connector's grids (called $G1$) and the transistor $T2$ has bottom connector's grids (called $G2$).

Fig. 13. Differential Pair Schematic View.

A. Interdigitated Differential Pair

The interdigitated technique simply alternates n fingers of each transistor from left to right of the stack. Fig. 14 shows a differential pair in interdigitated style. The hatched segments show the inter-stack routing segments. The gate nets $G1$ and $G2$ are routed in a *line* pattern, while the two drain nets $D1$ and $D2$ are routed in a *comb* pattern and finally the source net S is routed in a *serpentine* pattern.

It can be noticed that the length of the routing segments of the drain net $D1$ is equal to the length of the drain net $D2$, and the length of the routing segments of the gate net $G1$ is equal to the length of the gate net $G2$. This deals with the decreasing of the mismatch in the electrical behavior between the two transistors since the parasitic capacitance and resistance are directly proportional to the length of the routing segments of each net. Also, the two transistors T1 and T2 are affected by the same stress effect.

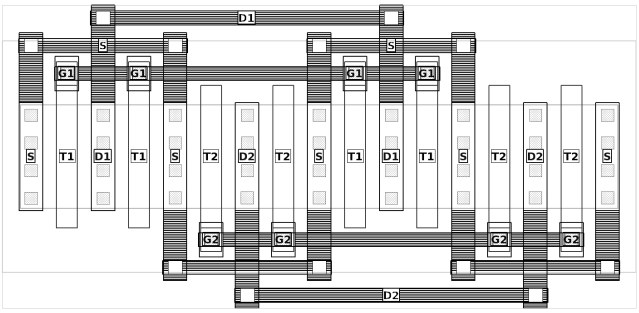

Fig. 14. Differential pair interdigitated style $W = 3.0\mu$m, $L = 0.15\mu$m, $NF = 4$, $Type = NMOS$ and $NB_{dummies} = 0$.

B. Mirror Differential Pair

The mirror technique alternates fingers of T1 and T2 starting from the middle of the stack. Fig. 15 shows a differential pair with mirror style. The gate nets $G1$ and $G2$ are routed in a *line* pattern, while the two drain nets $D1$ and $D2$ are routed in a *comb* pattern and finally the source net S is routed in a *mixed* pattern. It can be noticed that the routing segments' lengths of neither the drain nets $D1$ and $D2$, nor the gate nets $G1$ and $G2$ are equal. So there will be some mismatch in the electrical behavior between the two transistors. Also, T1 transistor, placed at the boundaries of the stack, suffers a more significant stress effect than T2 transistor. This is due to the fact that T1 is closer to the STI. It's important to note that although the mirror technique is preferred in the old technologies, the interdigitated layout style is much preferred in nanometric technologies to eliminate the stress effects, which are more significant.

In the interdigitated and the mirror styles we used only the inter-stack routing methodology because the device is made of a single stack.

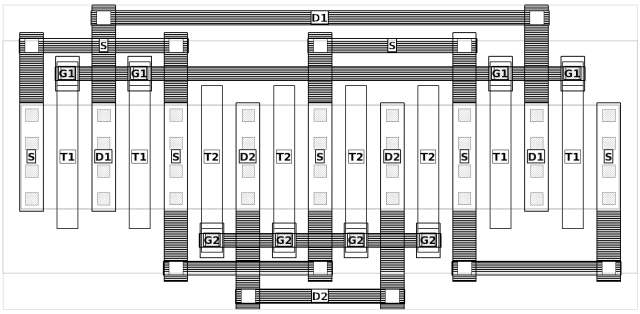

Fig. 15. Differential pair mirror style $W = 3.0\mu$m, $L = 0.15\mu$m, $NF = 4$, $Type = NMOS$ and $NB_{dummies} = 0$.

C. M2 Module Differential Pair

The idea of the M2 module [12] showed in Fig. 16, is to encapsulate every two fingers of the same transistor in a separated stack. The advantage of this approach is that although it takes more area, the stress effect of each stack is well defined and each stack can be placed in different topologies to control the overall placement of each transistor inside the device.

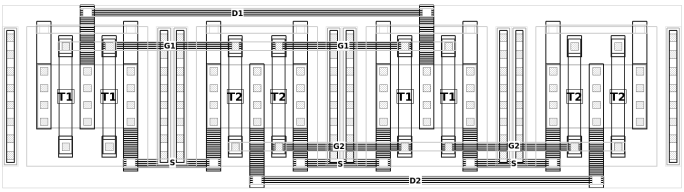

Fig. 16. Differential pair M2 module style $W = 3.0\mu$m, $L = 0.15\mu$m, $NF = 4$, $Type = NMOS$ and $NB_{dummies} = 0$.

Since this device is a multi-stack one, it takes advantage of both intra-stack and inter-stack routing methods. The hatched segments show the inter-stacks routing segments and the unhatched segments show the intra-stack routing segments (Fig. 16). The gate nets $G1$ and $G2$ are routed in a *line* pattern and the drain nets $D1$ and $D2$ as well as the source net S are routed in a *U-shape* pattern. The length of the routing segments of the gate and drain of each transistor is the same, allowing transistor matching.

D. 2D Common-Centroid Differential Pair

The idea of the two 2D common-centroid [8] is to split the device into different stacks and placed the finger of these stacks a specific way so that all the transistors' fingers of all the stacks have the a common center point.

This device being also a multi-stack one, it takes advantage of both intra-stack and inter-stack routing methods. The hatched segments show the inter-stacks routing segments and the unhatched segments show the intra-stack routing segments (Fig. 17). The source net S is routed in a *line* pattern and the other nets are routed in a *U-shape* pattern. It can be noticed that although the required area is large, the length of the routing segments of the gate and drain of each transistor is the same, allowing transistor matching.

E. Routing and Area Comparison

Table I shows the total length of each routed net for each transistor of the differential pair in a 65 nm technology, with $W = 3.0\mu m$, $L = 0.15\mu m$, $NF = 4$ (interdigitated, mirror and M2 module cases) or $NF = 8$ (2D common-centroid case), $Type = NMOS$ and $NB_{dummies}$=0 respectively for the four layout styles (interdigitated, mirror, M2 Module and 2D common-centroid). Nets G1 and D1 belong to transistor T1, nets G2 and D2 belong to transistor T2. Net S is shared between T1 and T2.

Table II compares area, aspect ratio and matching features of the differential pair in a 65 nm technology, with $W = 3.0\mu m$, $L = 0.15\mu m$, $NF = 4$ (interdigitated, mirror and M2 module cases) or $NF = 8$ (2D common-centroid case), $Type = NMOS$ and $NB_{dummies} = 0$ respectively for the four layout styles (interdigitated, mirror, M2 Module and 2D common-centroid). Note that the routing area is larger than the active area in all the cases.

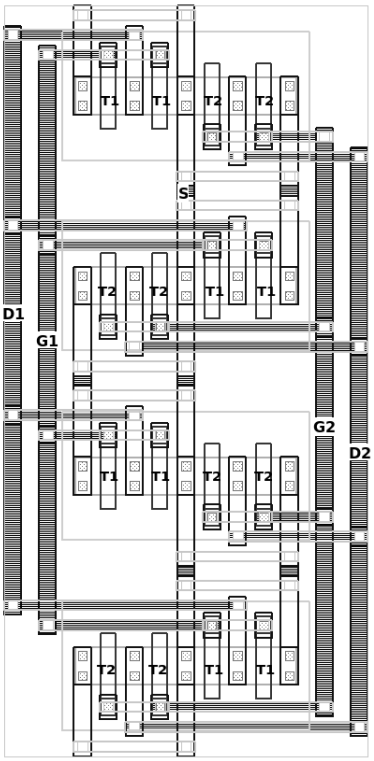

Fig. 17. Differential pair 2D common-centroid style $W = 3.0\mu$m, $L = 0.15\mu$m, $NF = 8$, $Type = NMOS$ and $NB_{dummies} = 0$.

TABLE I

ROUTING SEGMENTS LENGTH AS FUNCTION OF LAYOUT STYLE

	Routing segments length in μm.			
	Interdigitated	**Mirror**	**M2 Module**	**2D Common Centroid**
Net G1	6.94	7.98	12.88	13.24
Net G2	6.94	5.9	12.88	13.24
Net D1	10.8	11.825	16.745	17.17
Net D2	10.8	9.655	16.745	17.17
Net S	21.465	21.465	40.74	27.645

These tables provide the circuit designer with a clear vision of the advantage and drawbacks of various solutions to draw analogue device layouts.

F. Layout Dependant Parameter: Routing Capacitance

Let us take the example of the differential pair with 2D common-centroid style, with $W = 1.45\mu m$, $L = 0.18\mu m$ and $NF = 4$ in CMOS 65 nm technology. After the layout generation, the LDP, such as the parasitic routing capacitance and resistance can be retreived from the device. Table III

TABLE II

AREA AND MATCHING FEATURES AS FUNCTION OF LAYOUT STYLES

Areas in μm 2.				
	Interdigitated	**Mirror**	**M2 Module**	**2D Common Centroid**
Total area Active area height/width	10.01 3.23 0.48	10.01 3.23 0.48	17.70 5.46 0.27	27.67 13.65 2.07
horizontal gradient compensation	–	+	+	+
vertical gradient compensation	–	–	–	+
STI compensation	+	–	++	+

shows the values of the parasitic routing capacitances related to the gate net, drain net and source net of one of the two transistors of the differential pair.

TABLE III

PARASITIC ROUTING CAPACITANCES OF A 2D COMMON-CENTROID DIFFERENTIAL PAIR

Capacitance in fF.		
Gate Net	**Drain Net**	**Source Net**
0.0581	0.0937	0.0389

In this example the value of the parasitic routing capacitance of the 'G1' net is equal to the 'G2' one (Gate Net capacitance). The value of the parasitic routing capacitance of the 'D1' net is equal to the 'D2' one (Drain Net capacitance).

V. CONCLUSION

In this paper we have presented a method that allows simple and concise description for complex devices in nanometric technologies. It allows direct and accurate quantification of the layout-dependent parameters for different layout styles. The designer is therefore able to compare between layout styles and choose the suitable device layout for his circuit. We have investigated the two routing methods (intra-stack routing and inter-stack routing) for a differential pair in four different styles (interdigitated, mirror, M2 module and the 2D common-centroid). We showed that layout styles preferred for old technologies, such as mirror, may not be beneficial for new nanometric technologies.

REFERENCES

[1] D.J. Garrod, R.A. Rutenbar, and L.R. Carley. Automatic layout of custom analog cells in ANAGRAM. In *Computer-Aided Design, 1988. ICCAD-88. Digest of Technical Papers., IEEE International Conference on*, pages 544 –547, november 1988.

[2] J. M. Cohn, R. A. Rutenbar, and L. R. Carley. KOAN/ANAGRAM II: New Tools for Device-Level Analog Placement and Routing. *IEEE J. of Solid-State Circuits*, 26(3):330–342, march 1991.

[3] Ravindranath Naiknaware and Terri S. Fiez. Automated hierarchical CMOS analog circuit stack generation with intramodule connectivity and matching considerations. *IEEE J. of Solid-State Circuits*, 34(3):304–317, March 1999.

[4] Mohamed Dessouky, Marie-Minerve Louërat, and Jacky Porte. Layout-Oriented Synthesis of High Performance Analog Circuits . In *Proc. Design, Automation and Test in Europe*, pages 53–57, Paris, France, March 2000.

[5] Nuttorn Jangkrajarng, Sambuddha Bhattacharya, Roy Hartono, and C. J. Richard Shi. IPRAIl–intellectual property reuse-based analog IC layout automation. *Integration, the VLSI Journal*, 36(4):237 – 262, 2003. Analog and Mixed-signal IC Design and Design Methodologies.

[6] Mukesh Ranjan, Wim Verhaegen, Anuradha Agarwal, Hemanth Sampath, Ranga Vemuri, and Georges Gielen. Fast, layout-inclusive analog circuit synthesis using pre-compiled parasitic-aware symbolic performance models. In *Proc. Design, Automation and Test in Europe Conf.*, pages 604–609, Paris, France, February 2004.

[7] D. Stefanovic, M. Kayal, and M. Pastre. PAD: A new interactive knowledge-based analog design approach. *Analog Integrated Circuits and Signal Processing, Kluwer Academic Publishers*, pages 291–299, March 2005.

[8] Qiang Ma, E.F.Y. Young, and K.P. Pun. Analog placement with common centroid constraints. In *Computer-Aided Design, 2007. ICCAD 2007. IEEE/ACM International Conference on*, pages 579 –585, november 2007.

[9] Rafael Castro-Lopez, O. Guerra, E. Roca, and F.V. Fernandez. An Integrated Layout-Synthesis Approach for Analog ICs. *IEEE Trans. on Computer Aided Design*, 27(7):1179–1189, July 2008.

[10] H. Graeb, F. Balasa, R. Castro-Lopez, Y.W Chang, F.V. Fernandez, P.H Lin, and M. Strasser. Analog layout synthesis - Recent advances in topological approaches. In *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, pages 274 –279, april 2009.

[11] E. Avci, M. Strasser, H. Graeb, and U. Schlichtmann. A free-shape router for analog and RF applications. In *Circuit Theory and Design, 2009. ECCTD 2009. European Conference on*, pages 771 –774, august 2009.

[12] L. Lewyn, and N. Williams. Is a New Paradigm for Nanoscale Analog CMOS Design Needed? *Proceedings of the IEEE*, 99(1):3 –6, january 2011.

[13] R. Sommer, I. Rugen-Herzig, E. Hennig, U. Gatti, P. Malcovati, F. Maloberti, K. Einwich, C. Clauss, P. Schwarz, and G. Noessing. From system specification to layout: Seamless top-down design methods for analog and mixed-signal applications. In *Proc. Design Automation and Test in Europe Conf. (DATE)*, page 884, 2002.

[14] Darius Andriukaitis, Romualdas Anilionis, and Tomas Kersys. LOCOS CMOS process simulation. In *Conf. Information Technology Interfaces*, pages 489 –494, Catvat, Croatia, June 2006.

[15] Wang-Ling Goh, Kiat-Seng Yeo, Stephen Lazuardi, Wei Peng, Kam-Chew Leong, Lap Chan, and Alex See. Latchup characterization of 0.18-micron sti cobalt silicided test structures. *Microelectronics Journal*, 32(9):725 – 731, 2001.

[16] W.L. Goh *et al*. A comprehensive geometrical and biasing analysis for latchup in 0.18-μm C_oSI_2 STI CMOS structure. *Solid-State Electronics J.*, 48(12):2109 – 2114, 2004.

[17] V. Moroz, and G. Eneman, and P. Verheyen, and F. Nouri, and L. Washington, and L. Smith, and M. Jurczak, and D. Pramanik, and X. Xu. The impact of layout on stress-enhanced transistor performance. In *Int. Conf. on Simulation of Semiconductor Processes and Devices*, pages 143 – 146, Tokyo, Japan, September 2005.

[18] P.G. Drennan and M. L. Kniffin and D. R. Locascio. Implications of proximity effects for analog design. In *Custom Integrated Circuits Conf.*, pages 169 –176, San Diego, USA, September 2006.

[19] Stephanie Youssef and Damien Dupuis and Ramy Iskander and Marie-Minerve Louerat. Automatic Stress Effects Computation Based On A Layout Generation Tool For Analog IC. In *IEEE International Behavioral Modeling and Simulation Conference*, pages 7–12, September 2010.

[20] Vivek Joshi, Brian Cline, Dennis Sylvester, David Blaauw, and Kanak Agarwal. Stress aware layout optimization. In *Proc. Int. Symp. on Physical design*, pages 168–174, Portland, Oregon, April 2008.

[21] Ender Yilmaz and Gunhan Dundar. Analog layout generator for cmos circuits. *cad*, 28(1):32–45, january 2009.

[22] R. Jacob Baker. *CMOS: circuit design, layout, and simulation*. John Wiley and Sons, 2008.

[23] Behzad Razavi. *Design of Analog CMOS Integrated Circuits*. McGraw-Hill, 2000.

[24] W. Liu. *MOSFET Models for SPICE Simulation including BSIM3v3 and BSIM4*. Wiley Interscience, 2001.