Constructive Place-and-Route for FinFET-Based Transistor Arrays in Analog Circuits Under Nonlinear Gradients

Arvind K. Sharma, Meghna Madhusudan, Steven M. Burns,
Soner Yaldiz, Parijat Mukherjee, Ramesh Harjani, and Sachin S. Sapatnekar

***Abstract*—The design of active array structures in analog circuits requires careful matching to minimize the impact of variations. This work presents a constructive approach for building these arrays to directly incorporate shifts due to process variations, considering systematic first-order and second order gradients; to account for systematic layout effects, including parasitic mismatch and layout-dependent effects due to stress; and to ensure that the resulting layout delivers high performance. The proposed algorithms are targeted to FinFET technologies and are validated for multiple analog blocks in a commercial 12nm FinFET process. The layouts generated by the proposed method are demonstrated to provide better matching and performance than prior methods.**

Keywords: layout-dependent effects, mismatch, nonlinear gradients, routing parasitics, electromigration.

I. Introduction

Analog circuits such as amplifiers and digital-to-analog converters, which are widely used in systems-on-chip (SoCs), are very sensitive to device mismatch caused by on-chip variations. The mitigation of these effects has grown harder with technology scaling, where complex variation patterns cause greater unpredictability in circuit performance parameters. The use of optimized layout structures is critical to ensure the level of device matching that is essential for the correct functioning of high-performance analog circuits.

A taxonomy of variations shows several sources of variations: (1) During manufacturing, process variations cause the parameter values of active and passive devices on a chip to be perturbed fromvtheir nominal values. Some of these variations can be modeled *deterministically* as they are caused by effects that create non-random perturbations across a die, e.g., those due to optical proximity, lens aberration, well proximity, strained silicon, or chemical-mechanical polishing effects [1]. Systematic variations are dominated by linear (first-order) and quadratic (second-order) components [2], [3], and to reduce their impact, special layout techniques are used in transistor and capacitor array structures. The traditional way of canceling first-order systematic variations is through the use of common-centroid (CC) layout [4]: large structures are decomposed into arrays of smaller unit cells that are arrayed in a (typically) near-square shape. The unit cells of transistors or capacitors that are to be matched are placed so that their centroids coincide; under small perturbations and linear process variations, this can be shown to cancel out the impact of process variations [5]. Similarly, more advanced/complex layout patterns are used to cancel the impact of nonlinear variations, although this area is less widely studied. Although these layout placement schemes can, in principle, minimize the impact of systematic variations, it is difficult and time-consuming to manually generate an optimal CC placement. Moreover, the optimal CC placements must also be easily routable with low wirelength, few vias, and with low mismatch between the routing parasitics.

Other variations that affect analog circuits are not deterministic and are best described as random distributions (e.g., oxide thickness variations, fluctuations in the number and locations of dopant atoms in a transistor, or line edge roughness). Many of these effects have been modeled, and their effects accounted for, in the context of digital systems [6], [7], but there is relatively less work in the area of analog circuits. Random variations are well studied in the context of analog circuits, with a landmark paper by Pelgrom [8] showing how their variance is inversely proportional to device area: intuitively, larger transistors have fewer uncorrelated variations due to averaging over the device area.

As process technologies shrink to smaller nodes, the performance of digital designs has been greatly enhanced, but analog circuits typically perform worse at smaller nodes. Nevertheless, due to the importance of integrating digital and analog circuitry on a single chip, it is imperative to build analog circuits in advanced FinFET nodes. Several variation effects gain prominence at these nodes: in particular,

- layout-dependent effects (LDEs) that change the stress environment of a transistor, and hence its I–V characteristic, resulting in differential mismatch unless LDE-conscious considerations are incorporated into layout;
- increased interconnect parasitics associated with the smaller cross-sections seen in scaled technologies, and increased via resistances that penalize wires from shifting layers;
- lithography considerations that require wires to be routed in the same direction within a layer (as opposed to free-form wire direction changes from horizontal to vertical

A. K. Sharma was with the University of Minnesota, Minneapolis, MN, at the time this work was performed. M. Madhusudan, R. Harjani, and S. S. Sapatnekar are with the University of Minnesota, Minneapolis, MN. S. M. Burns was with Intel Labs, Hillsboro, OR, when this work was performed. S. Yaldiz and P. Mukherjee are with Intel Labs, Hillsboro, OR. This work was supported in part by the DARPA IDEA program, as part of the ALIGN project, under SPAWAR Contract N660011824048, and by the Semiconductor Research Corporation under Task 2810.083.

and vice versa in older technologies), particularly in lower metal layers; and

- electromigration (EM) effects caused by high current densities (i.e., high currents that pass through narrow wires): this can be mitigated by using wider wires.

Thus, designs in advanced FinFET-based technologies have many characteristics that are different from design in more classical bulk-based CMOS designs or BJT-based designs, and therefore require specific consideration in developing algorithms that are tailored to these technologies.

In the context of common-centroid transistor placement in FinFET-based circuits, the topic of this paper, another important consideration is the maximization of diffusion-sharing between adjacent transistors. If the source/drain nodes of adjacent transistors are adjacent, their diffusion regions can be connected directly, removing the need for metal connections that require vias (and the corresponding resistance penalty) and consume area due to design rule spacing requirements between vias and unconnected diffusion strips. The presence or absence of diffusion-sharing not only reduces area, but also determines LDEs and parasitics: therefore, maximizing diffusion sharing, and maintaining the same diffusion-sharing in matched transistors (i.e., matched device are constructed so that it has the same number of diffusion breaks), ensures close matching in performance metrics. Moreover, common-centroid design in FinFET technologies must also account for the rigid design rule requirements in these technologies that are enforced due to subwavelength lithography, such as quantized device widths and unidirectional routing in lower-level metal layers.

In industry practice, CC patterns are typically specified by the designer; even early versions of ALIGN [9], [10] used the same strategy. Automated minimization of systematic variations has attracted a great deal of research attention [5], [11]–[22] through the use of CC techniques, many of these approaches [11]–[14] address capacitor arrays and cannot be directly extended to transistor arrays as they are not required to consider transistor-specific effects such as LDEs and diffusion-sharing. CC transistor layout is addressed in [16], [17], but these methods consider a basic version of the problem that does not incorporate LDE, parasitic mismatch, or diffusion-sharing considerations. Diffusion-sharing is taken into account (but not LDEs or parasitic mismatch) in [18], where a graph representation of the arrayed structure, with edges joining source nodes and drain nodes of transistors, undergoes an Euler path discovery process for diffusion sharing, a classical technique from digital standard-cell layout. The method enumerates all Euler paths to determine the best layout, which can have considerable computation cost.

To consider higher-order variations and correlations, the idea of dispersion was used in [15]. Intuitively, if the unit cells of each element of an array structure are uniformly distributed throughout the layout, the cells are "well dispersed" and face similar noncancellable variations (e.g., quadratic variations) when averaged over the array. A metric based on dispersion is used to create array layouts that are common-centroid, and yet maximize dispersion; however, the work only considered arrays with two types of devices. Most prior works have focused on bulk transistor technologies, without incorporating FinFET-technology-specific effects. An exception is [19], which focused on a limited FinFET issue, gate misalignment, creating layouts that share diffusion and attempt to maximize dispersion; however, (a) LDEs, routing parasitic mismatches, and EM are not taken into consideration, and (b) the methods are specific to current mirror structures rather than general arrays.

Further, the algorithms presented in [16]–[19] only cancel linear systematic variations and do not optimize the placement for second-order gradients. However, second-order systematic variations are also important and should be minimized. For example, a linear gradient in the threshold voltage of a long-channel transistor across a die will result in a nonlinear change in its drain current [15]. In [23], the authors presented measurement results for distance-dependent variations for a FinFET technology. The authors shown that the variations have nonlinear component also and must be considered to optimize performance of larger circuits. In [15], [21], the authors presented algorithms to cancel nonlinear gradients. However, the algorithms are only applicable for two equally sized devices. Moreover, routing parasitic mismatch is not considered in the proposed algorithms.

Contributions of this work. This paper creates a fast, constructive approach for the placement of transistor arrays in FinFET circuits that minimizes both linear and nonlinear gradients, while maximizing diffusion sharing between devices, incorporating considerations of routing parasitic mismatch, building layouts that minimize LDE-mismatch. The work is an enhanced version of our earlier work in [22]. The consideration of nonlinear gradients on top of CC layout is a new contribution over the preliminary version, which only considers the impact of linear gradients; this paper presents algorithms to also minimize the impact of nonlinear gradients. We employ a routing algorithm that optimizes for EM: this is identical to that in [22], and therefore we do not reproduce its description in this work. We demonstrate experimentally that in comparison with existing approaches, the transistor arrays placed and routed using our approach perform better in the presence of systematic variations, LDEs, layout parasitics, and EM-induced degradation. We show improvement in the performance of transistor arrays compared to prior methods, including our earlier work in [22].

The remainder of the paper describes the details of our approach and the experimental results. Section II overviews the sources of variation in deeply-scaled FinFET technology nodes, and is followed by a description of our constructive placement algorithm. The base CC approach that minimizes linear gradients is described in Section III and is followed by an enhancement in Section IV that also reduces (typically smaller) second-order gradients by refining the placement, while maintaining its CC nature. Experimental results on a set of FinFET-based array structures, which constitute larger analog circuits, are presented in Section V, and we close the paper in Section VI.

II. Background

A. On-chip Variations

As stated in Section I, process variations may be either systematic or random in nature: systematic variations may be modeled by linear or nonlinear deterministic gradients, and random variations are represented by distributions whose statistics are described by [8]. In this section, we summarize these variations.

1) Spatial Variations: Analog circuit performance is typically predicated on reducing the differential variability between devices, often referred to as *mismatch*. One of the most widely-used transistor variation model in analog design was proposed by Pelgrom [8] in the 1980s, which quantifies the mismatch in a parameter P of two devices as the sum of two random variables corresponding to the uncorrelated component, u, and a spatially correlated component, s. The variance of the mismatch is given by:

$$\sigma^2(\Delta P) = \sigma_u^2 + \sigma_s^2 \tag{1}$$
$$\text{where } \sigma_u^2 = \frac{A_P^2}{WL} \;;\; \sigma_s^2 = S_P^2 r^2$$

where A_P and S_P are technology-dependent proportionality constants, W and L are the device width and length, respectively, r is the distance between the devices, and σ^2 represents the variance of the corresponding random variable. Note that the first component depends on the area of the transistor and can be diluted by using large-sized transistors, while the second depends on the distance between components, and can be mitigated by layouts that reduce the distance between devices.

2) Gradient-based Variations: An N^{th} order gradient in a parameter P as a function of location (x, y) on chip can be represented using the Taylor expansion to an N^{th} order polynomial:

$$P(x,y) = P_0 + C_0 + \sum_{i=1}^{N}\sum_{j=0}^{i} C_{i,j} x^j y^{i-j} \tag{2}$$

where P_0 is the parameter P value at the reference location (x_0, y_0), C_0 is the constant, and the C are gradient coefficients.

In prior work, most approaches have considered systematic variations to be dominated by a first-order linear gradient, and layout patterns should be optimized to cancel these (i.e., using a CC layout pattern). This is because process-induced systematic mismatch between devices is mainly caused by lithographic and technological effects during fabrication [24]. Under conventional models, the corresponding gradients are dominated by strong linear components. Further, even when across-die gradients are nonlinear, since the unit cell array is much smaller than the die size, nonlinear systematic variations have been approximated by linear gradients [25]. However, as we translate linear process gradients in transistor parameters to electrical performance metrics, we find that performance metrics vary nonlinearly with transistor parameters, resulting in nonlinear gradient in device performance parameters. For example, a linear gradient in the threshold voltage of a transistor across a die will result in nonlinear change in its drain current [15], particularly in analog circuits where long-channel devices are used to mitigate random variations. Therefore, it is not adequate to match only linear gradients using CC layout: nonlinear gradients should also be considered to optimize circuit performance.

As discussed earlier, mismatch-sensitive devices in analog circuits are divided into small unit cells. For each device parameter, P can be averaged over all the number of unit cells of the respective device. As each unit cells is small, the gradient in P within the unit cells can be ignored; consequently, the parameter can be represented by the parameter value at the center of the unit cells. For example, the average value of parameter P for device A having s_A unit cells is:

$$P_{avg}^{A} = \frac{1}{s_A}\sum_{i=1}^{s_A} P(x_i, y_i) \tag{3}$$

where (x_i, y_i) represent the center of the unit cells.

For a set of k devices, with each device i consisting of s_i segments; we denote the location of segment i of device j as $(x_{i,j}, y_{i,j})$. For these device the criteria to cancel out the first-order gradient can be expressed using (2) and (3):

$$\frac{1}{s_1}\sum_{i=1}^{s_1} x_{i,1} = \frac{1}{s_2}\sum_{i=1}^{s_2} x_{i,2} = \cdots = \frac{1}{s_k}\sum_{i=1}^{s_k} x_{i,k} \tag{4}$$
$$\frac{1}{s_1}\sum_{i=1}^{s_1} y_{i,1} = \frac{1}{s_2}\sum_{i=1}^{s_2} y_{i,2} = \cdots = \frac{1}{s_k}\sum_{i=1}^{s_k} y_{i,k} \tag{5}$$

Equation (4) and (5) shows that the centroids of the devices should coincide to cancel out the first order gradients (i.e., a CC pattern must be used). For example, Fig. 1(b) shows a CC layout pattern of a differential pair. The devices A and B are each divided into sixteen unit cells and placed such that the centroid coincides at **C**; i.e., the layout pattern will cancel first-order mismatch between device A and B.

Similarly the criteria to cancel out second-order gradients can be expressed using (2) and (3):

$$\frac{1}{s_1}\sum_{i=1}^{s_1} x_{i,1}^2 = \frac{1}{s_2}\sum_{i=1}^{s_2} x_{i,2}^2 = \cdots = \frac{1}{s_k}\sum_{i=1}^{s_k} x_{i,k}^2 \tag{6}$$
$$\frac{1}{s_1}\sum_{i=1}^{s_1} y_{i,1}^2 = \frac{1}{s_2}\sum_{i=1}^{s_2} y_{i,2}^2 = \cdots = \frac{1}{s_k}\sum_{i=1}^{s_k} y_{i,k}^2 \tag{7}$$
$$\frac{1}{s_1}\sum_{i=1}^{s_1} x_{i,1}y_{i,1} = \frac{1}{s_2}\sum_{i=1}^{s_2} x_{i,2}y_{i,2} = \cdots = \frac{1}{s_k}\sum_{i=1}^{s_k} x_{i,k}y_{i,k} \tag{8}$$

Note that the layout pattern in Fig. 1(b) satisfies the linear gradient cancellation requirements in Eqs. (4) and (5), but not (6)–(8). An alternative layout pattern for the differential pair, shown in Fig. 1(c), satisfies (4)–(8); i.e., the layout pattern will cancel both first-order and second-order mismatch between devices A and B.

In the example above, the special layout patterns minimizes the impact of systematic variations for this structure, but for a general structure, it is difficult and time-consuming to generate an optimal layout placement manually. Moreover, the placements must also be routing-friendly: for example, resistive parasitics at the terminals of device A and B in a

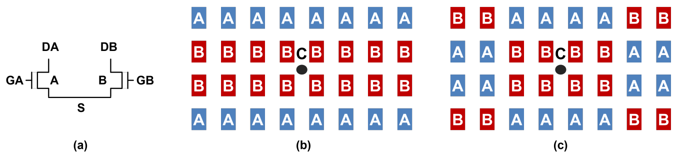
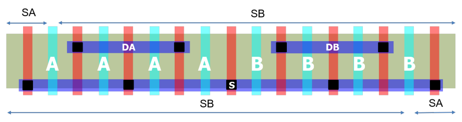


Figure 1: (a) A schematic of a differential pair. (b) A CC pattern to cancel first-order gradients. (c) A CC pattern to cancel first- and second-order gradients.

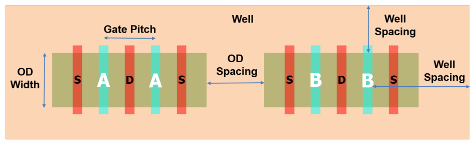
differential pair (Fig. 1(a)) impact the transistor transconductance, and should be small and matched. This can be achieved by considering routing parasitics while generating the optimal placements for systematic variations cancellation.

B. Layout-Dependent Effects

At advanced technology nodes, LDEs [26]–[28] induce shifts in transistor performance parameters stemming from relative position in the layout. The most common LDEs (Fig. 2) are discussed next.



(a) SA and SB parameters for LOD effect.



(b) OD width and spacing, gate pitch, and well proximity effects.

Figure 2: Layout-dependent effects.

Well proximity effect (WPE) At nanoscale CMOS nodes, to minimize the latchup effect high-energy ions are used to create a deep retrograde well profile [28]. However, the high-energy ions scatter at the edge of photo resist and change the doping profile that modifies V_{th} of a device based on its distance from the well edge. This effect is commonly knows as WPE [28]. The well spacing is shown for device B in Fig. 2 (b). The WPE induced mismatch can be minimized by keeping well edges far from devices or by maintaining equal well spacing for the devices to be matched.

Process-induced stress has been intentionally used at nanoscale nodes to improve a transistor performance. However, the improvement depends on a device layout and its proximity, therefore, result in LDEs. The main LDEs due to the stress are as follows:

Length of diffusion (LOD) One of the most significant LDEs is caused by LOD effect [29], whereby the stress on a transistor, and hence its V_{th}, varies with the length of the diffusion region. The impact of LOD [29] is described by two parameters, SA and SB, the distances from poly-gate to the diffusion/active edge on either side of the device. For a device of gate length L_g, and n unit cells [30]:

$$\Delta V_{th} \propto \frac{1}{\text{LOD}} = \sum_{i=1}^{n} \left(\frac{1}{\text{SA}_i + 0.5L_g} + \frac{1}{\text{SB}_i + 0.5L_g} \right) \quad (9)$$

Fig 2(a) shows SA and SB parameters for a unit cell of device A and B. Matched devices must have same values of SA and SB to match their threshold voltage shift, ΔV_{th}.

Oxide definition (OD) spacing and width Spacing between the OD regions (active areas), shown in Fig. 2(b), changes stress induced in a transistor; therefore, V_{th} varies as a function of OD spacing [26]. The effect is also known as oxide spacing effect (OSE). Moreover, stress induced in a transistor varies with the OD width (active area width). These effects can be avoided by maintaining same OD width and spacing for devices to be matched. For analog cells a unit cell based approach is used in which devices to be matched are divided into unit cells, therefore, same OD width is maintained for different devices. Further, the same OD spacing is used across unit cells. Moreover, the unit cells are placed such that devices to be matched have same number of diffusion breaks (i.e., OD breaks).

Gate pitch Stress induced in a transistor is also a function of gate pitch or poly pitch [26]. Gate pitch is shown in Fig. 2(b) for device A. As gate pitch increases the volume of the stressor material around the poly increases, this results in increased induced stress in the transistor channel, consequently, V_{th} varies. In analog cells, the effect is minimized by using a same poly pitch for devices to be matched.

In this work, we use a unit cell approach that is designed to cancel out all LDEs except LOD and WPE. Specifically, the gate/poly pitches are uniform for the analog blocks we place in CC; by construction, the unit cell approach ensures that the OD width is uniform; the y-direction OD spacing (OSE) is uniform for each transistor due to the use of a row-based unit cell placement approach, and the x-direction spacing is uniform due to diffusion-sharing. Therefore, we focus on optimizing LOD and WPE mismatch through the use of dummies and using placement techniques.

C. Parasitics

Parasitics are critical in analog layouts and can degrade the circuit performance considerably and also cause circuit failure. Nodes can be sensitive to resistive or capacitive parasitics or both. Typically, in nanometer-scale technology nodes, the resistivity of the lower metal layers is very high [31], [32]. Moreover, uni-directional routing for lower metal layers results in increased parasitics due to an increased number of vias. Hence, resistance parasitics tend to dominate at the analog cell level. In current mirrors, these routing parasitics can cause mismatch in the source voltage of matched devices and may result in a current ratio shift. In differential pair circuits, these parasitics affect the transconductance of the circuit (G_m) which can further degrade the performance of the analog circuit such as the gain and bandwidth of an operational transconductance amplifier (OTA) [33]. Therefore, it is important to place transistors in such a way that routing length is reduced. Moreover, as mentioned earlier, diffusion breaks can also result in increased wire and via parasitics, and our approach explicitly attempts to minimize the number of diffusion breaks.

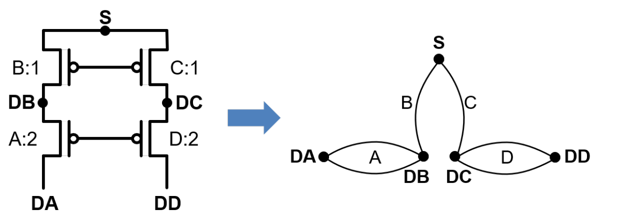


Figure 3: (a) A PMOS cascoded load and (b) its corresponding graph.

III. Placement and Routing of Array Structures Considering Linear Gradients

In this section, we develop a constructive CC placement algorithm for the transistor-based building-block cells that are commonly used in analog circuits (e.g., current mirrors, differential pairs, cascoded differential pair, cascoded load, etc.), which we will refer to as "analog cells". We begin with an explanation of the graph representation of analog cells in Section III-A, followed by a detailed explanation of the CC placement algorithm in Section III-B.

A. Graph Representation of Analog Cells

We represent the transistor netlist of an analog cell as a graph, $G(V, E)$. The set of vertices V represents nodes in the schematic/netlist, and the set of edges E corresponds to source-drain connections of the transistors, where the number of edges for a device is equal to the number of unit cells for the device. Fig. 3(a) shows the schematic of a PMOS cascoded load and its corresponding graph. The cascoded load has four devices, where devices A and D have two unit cells each, while B and C have one unit cell each. The corresponding graph is shown in Fig. 3(b).

B. Common-centroid Placement

In Algorithm 1, we present a procedure for placing the devices in an analog cell in CC pattern. In addition to canceling out systematic process variations in the devices, which is ensured by CC placement, the algorithm optimizes the area and source/drain parasitics of the layout by maximizing diffusion-sharing and incorporating LDEs. The inputs to the algorithm are the analog cell netlist, listing the number of unit cells for each device, and the unit cell aspect ratio (K).

The algorithm consists of four steps: (1) a *preprocessing* step that handles devices with an odd number of unit cells to reduce the problem to one with even unit cells for each device: this allows us to place half of the layout (which now has an integer number of cells) in CC fashion and reflect it to create a full CC layout; (2) an *aspect ratio calculation* step that determines the number of rows and columns of the array to keep the layout as close to a square as possible; (3) and (4) two placement steps for the unit cells for, respectively, the odd-numbered cells extracted in step (1), and for the half-layout for the remainder of the cells, which is then reflected about the CC point to build the full layout.

Next, we explain these four steps in detail for the current mirror bank in Fig. 4. Fig. 4(a) shows a schematic of the

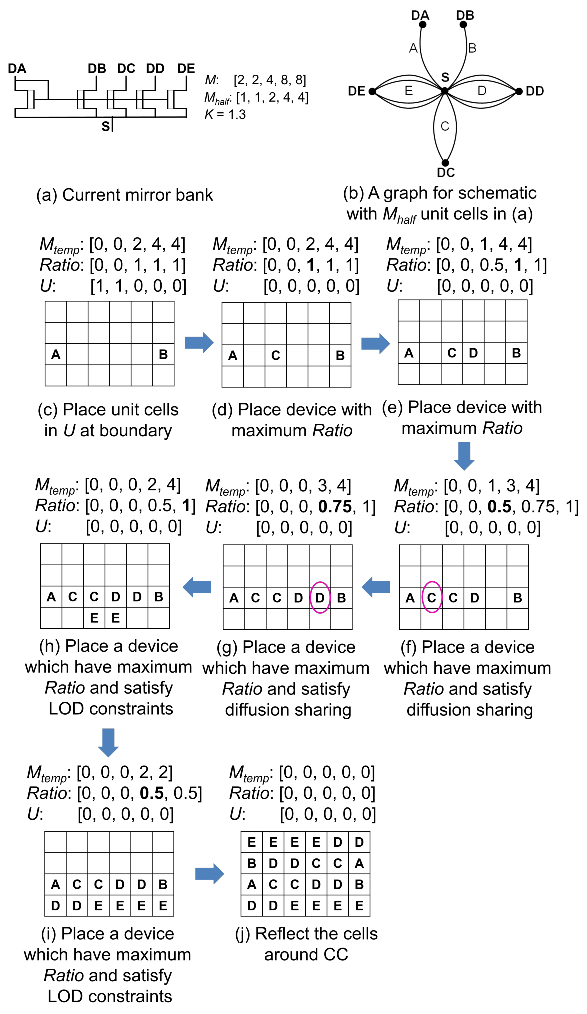

DA
DB
DC
DD
DE
S
M: [2, 2, 4, 8, 8]
M_half: [1, 1, 2, 4, 4]
K = 1.3
(a) Current mirror bank
DA
DB
A
B
DE
E
S
D
DD
C
DC
(b) A graph for schematic with M_half unit cells in (a)
M_temp: [0, 0, 2, 4, 4]
Ratio: [0, 0, 1, 1, 1]
U: [1, 1, 0, 0, 0]
(c) Place unit cells in U at boundary
M_temp: [0, 0, 2, 4, 4]
Ratio: [0, 0, 1, 1, 1]
U: [0, 0, 0, 0, 0]
(d) Place device with maximum Ratio
M_temp: [0, 0, 1, 4, 4]
Ratio: [0, 0, 0.5, 1, 1]
U: [0, 0, 0, 0, 0]
(e) Place device with maximum Ratio
M_temp: [0, 0, 0, 2, 4]
Ratio: [0, 0, 0, 0.5, 1]
U: [0, 0, 0, 0, 0]
(h) Place a device which have maximum Ratio and satisfy LOD constraints
M_temp: [0, 0, 0, 3, 4]
Ratio: [0, 0, 0, 0.75, 1]
U: [0, 0, 0, 0, 0]
(g) Place a device which have maximum Ratio and satisfy diffusion sharing
M_temp: [0, 0, 1, 3, 4]
Ratio: [0, 0, 0.5, 0.75, 1]
U: [0, 0, 0, 0, 0]
(f) Place a device which have maximum Ratio and satisfy diffusion sharing
M_temp: [0, 0, 0, 2, 2]
Ratio: [0, 0, 0, 0.5, 0.5]
U: [0, 0, 0, 0, 0]
(i) Place a device which have maximum Ratio and satisfy LOD constraints
M_temp: [0, 0, 0, 0, 0]
Ratio: [0, 0, 0, 0, 0]
U: [0, 0, 0, 0, 0]
(j) Reflect the cells around CC

Figure 4: The application of the proposed common-centroid algorithm on a current mirror bank. (b) is the graph for the schematic with M_{half}.

example circuit consists of five devices, A, B, C, D, and E, whose multiplicity matrix $M = [2, 2, 4, 8, 8]$ represents, in the same order, the number of unit cells of these five devices. The algorithm proceeds through the following steps:

Step 1: Preprocessing First, the list of devices with an odd number of unit cells in M is stored in a list U (line 4). These odd unit cells will be divided into half-cells (i.e., cells with the same height as unit cells, but with half the active width compared to the unit cells). This transformation ensures that the number of unit cells is even for all devices, thus enabling CC layout. However, since these half-cells cannot share diffusion with other "full-cells" and must be placed at the edges of the layout matrix, X, we add them to a list, U, of cells that must be at the edge of X.

Next, the remaining unit cells are divided in two halves and stored in the list M_{half} (line 5). In the succeeding steps, we will first place the unit cells in M_{half} in the matrix X in the lower half of the array; when the matrix has an odd number of rows, the left half of the middle row is also populated. Later, in Step 5, we will reflect this placement to the other half of the matrix through the CC point.

Algorithm 1 Common-centroid placement of analog cells

```
1: **Input**: Analog cell netlist; Unit cell height to width ratio, $K$; Device size vector
   $M = [M_1, M_2, \cdots, M_N]$, where, $M_i$ is the number of unit cells for device $i$
2: **Output**: Common-centroid placement $X$ in an array of size $r \times q$.
3: // Step 1: Preprocessing
4: $U \leftarrow$ list of $M_i$ with odd unit cells
5: $M_{half} \leftarrow [\lfloor M_1/2 \rfloor, \lfloor M_2/2 \rfloor, \cdots, \lfloor M_N/2 \rfloor]$
6: // Create layout for half the matrix; Reflect other half around CC
7: Create netlist graph $G(V, E)$, use $M_{half}$ for edge multiplicities
8: **for** $i = 1$ to $N$ **do** // Over all devices
9:    **if** $M_{half}[i]$ is odd $\wedge$ [(deg($M_{i,source}$)==$M_{half}[i]$) $\vee$
10:                                   (deg($M_{i,drain}$)==$M_{half}[i]$)] **then**
11:       $U.add(M_i)$ // Add odd unit cell of $M_i$ in $U$
12:       $M_{half}[i] = M_{half}[i] - 1$
13:    **end if**
14: **end for**
15: // Step 2: Aspect ratio $r \times q$ calculation
16: $r = Round\left(\sqrt{\sum M/K}\right)$ // Row calculation
17: **if** $len(U)$ is odd and $r$ is even **then**
18:    $r = r + 1$
19: **end if**
20: $q = \lceil \sum M/r \rceil$ // Column calculation
21: $q = 2\lfloor \frac{q+1}{2} \rfloor$ // Make column even
22: Calculate common-centroid point ($C_X$, $C_Y$) = ($\frac{r}{2}$, $\frac{q}{2}$)
23: // Step 3: Placement of devices in List U
24: **if** $len(U)$ is odd **then**
25:    $X[C_X][C_Y] = U[len(U)]$ // Place the last element of U
26:    $U.delete(len(U))$
27: **end if**
28: $t = len(U)$ // length of list U
29: $n \leftarrow 1$ // Counter
30: **for** $i = 0$ to $(\frac{t}{2} - 1)$ **do** // Place cells in $U$ at boundary
31:    $X[n * \frac{r}{2} - i][n] = U[1]$ and $X[n * \frac{r}{2} - i][q + 1 - n] = U[2]$
32:    $U.delete(1, 2)$
33:    **if** $(n * \frac{r}{2} - i - 1) == 0$ **then**
34:       $n = n + 1$
35:    **end if**
36: **end for**
37: // Step 4: Placement of devices in List $M_{half}$
38: Sort $M_{half}$ in ascending order; set $M_{temp} = M_{half}$
39: **for** $i = \lceil \frac{r}{2} \rceil$ to 1 **do** // Over half rows
40:    $Z = 1$ // If $Z = 1$, a cell is placed at the left of CC
41:    $Z_l = C_Y$ and $Z_r = C_Y + 1$ // Left and right counters
42:    **while** Row $i$ is not filled **do**
43:       $Ratio = [k/l$ for $(k, l)$ in $(M_{temp}, M_{half})]$
44:       **if** $Z == 1$ **then** // Place cell at the left of the CC
45:          Select $M_X$ which can share the diffusion region
46:                with device at $X[i][Z_l + 1]$ and have maximum Ratio
47:          **if** $M_X == X[j$ for $j$ in $(\frac{r}{2}, \frac{r}{2} - 1, \cdots, i - 1)][Z_l]$ or
48:             $M_X == X[j$ for $j$ in $(\frac{r}{2}, \frac{r}{2} - 1, \cdots, i - 1)][Z_r]$ **then**
49:                                           // $M_X$ already placed in the column
50:             Go to line 46 and select another $M_X$
51:                                           // To minimize LOD mismatch
52:          **end if**
53:          $X[i][Z_l] = M_X$
54:          $Z_l = Z_l - 1$ and $M_X = M_X - 1$
55:       **else** // Z=0, place cell at the right of the CC
56:          Select $M_X$ which can share the diffusion region
57:                with device at $X[i][Z_r - 1]$ and have maximum Ratio
58:          **if** $M_X == X[j$ for $j$ in $(\frac{r}{2}, \frac{r}{2} - 1, \cdots, i - 1)][Z_r]$ or
59:             $M_X == X[j$ for $j$ in $(\frac{r}{2}, \frac{r}{2} - 1, \cdots, i - 1)][Z_l]$ **then**
60:                                           // $M_X$ already placed in the column
61:             Go to line 57 and select another $M_X$
62:                                           // To minimize LOD mismatch
63:          **end if**
64:          $X[i][Z_r] = M_X$
65:          $Z_r = Z_r + 1$ and $M_X = M_X - 1$
66:       **end if**
67:       **if** ($r$ is even $\vee$ $i\ != \frac{r}{2}$) **then**
68:          $Z = \overline{Z}$
69:       **end if**
70:    **end while**
71: **end for**
72: // Step 5: Postprocessing
73: Reflect the remaining half devices around CC in $X$
74: Calculate $\Delta V_{th,LOD}^{max}$ using (9) between two devices
75: Calculate # of dummies using (9) to ensure $\Delta V_{th,LOD}^{max} < \varepsilon \cdot V_{th}$
```

To place this half, a graph $G(V, E)$ is created for the unit cells in M_{half} (line 7): note that the number of edges here is same as the number of unit cells in M_{half}. A graph for the current mirror bank testcase is shown in Fig. 4(b). Next, the unit cells with odd multiplicity in M_{half} are detected: these cells must lie at the end point of an Eulerian path and can only be placed without a diffusion break at the boundary of the CC placement matrix, X. All such cells are added to the list U (lines 8–14). This arises when an element of M_{half} is odd (i.e., it must be at an end point of an Eulerian path) and its source or drain has no other connections other than to the devices in M_{half}. For the current mirror bank testcase device A and B have odd number of unit cells in M_{half}, therefore, these unit cells are added to the list U as shown in Fig. 4(c).

Step 2: Aspect ratio calculation (lines 16–22) Traditionally, the aspect ratio of a CC layout has been chosen to be close to a square [4], for which the maximum distance from the origin is smaller than any other rectangle. This helps ensure the validity of the assumption that process shifts are small and can be approximated by a linear Taylor series approximation. Therefore, in this step, the number of rows and columns ($r \times q$) of the matrix X are calculated so that a near-square aspect ratio is obtained. The number of rows is calculated using line 16 and adjusted according to the unit cells in list U (lines 17–19): we will elaborate on this in Step 3. Finally, the number of columns and CC point (C_X, C_Y) are calculated (lines 20–22). For the current mirror bank testcase, the array size is 4×6, and CC point is at (3, 2).

Step 3: Placement of unit cells in U (lines 24–36) In this step, the unit cells in U are placed at the boundary in X. If the total number of unit cells is odd in U (i.e., length of U is odd), then one of the unit cells is placed at the center of the odd row without a diffusion break (lines 24–27); in Step 2, we had ensured that when the length of U is odd, the total number of rows is odd (lines 17–19).

The remaining unit cells in U are placed at the boundary of X (lines 28–36). For this, first, we initialize a counter n (line 29), which selects a column from the leftmost and rightmost ends of X for unit cell placement. Once the leftmost and rightmost columns are filled, the counter is increased and the next columns are selected (lines 33–35). For example, in the current mirror testcase there are two unit cells in U one each from device A and B. These are placed at the left and right boundary location as shown in Fig. 4(c). In this case, one column at each edge suffices; the role of the counter is to populate a second or third column, if necessary.

Step 4: Placement of unit cells in M_{half} The unit cells in M_{half} are sorted in ascending order and stored in M_{temp} (line 38), which represents the set of cells that are yet to be placed. Thereafter, the unit cells are placed over half of the rows, starting at line 39. These unit cells in each row are placed alternately at the left/right of the CC point. The starting location for the cells to be placed in a row is set by two variables Z_l and Z_r (line 41). Initially, Z_l and Z_r are set to C_Y and $C_Y + 1$, respectively (line 41). After placement of a unit cell at the left (right) of the CC point, Z_l (Z_r) is decreased (increased) by one and the location is updated. In other words, Z_l and Z_r move to the left/right of the CC location after a cell placement at the left/right of the CC. The cells from M_{temp} are then successively placed in a row until it gets filled (line 42).

The order in which the unit cells are populated into rows is based on the parameter, $Ratio$, that is computed for each device (line 43): this is the ratio of unplaced unit cells for that device in M_{temp} to the total number of unit cells M_{half}. The principle is that we choose a device for placement if, relatively speaking, a smaller fraction of its unit cells have been placed so far. This helps ensure better dispersion of the devices. Using this principle, the algorithm now selects a device from M_{temp} (if possible, that can share the diffusion region) and has maximum $Ratio$ (lines 46 and 57).

In each row, the method alternately places cells to the left and to the right of the CC point. The Boolean counter Z is used to enforce this by verifying whether it is 0 or 1. The exception to this alternation is when the total number of rows is odd and the CC placement occurs in the middle row: in this row, the cells are placed at the left of the CC only. As we will explain later, this left half-row will be reflected to right half-row in Step 5 about the CC point. Thus, the Boolean counter Z is inverted each time a unit cell is placed in a row, except when the total number of rows is odd and cells are placed in the middle row (lines 67–69). Uniform distribution of devices unit cells over columns will reduce mismatch between SA and SB parameters (shown in Fig. 2) of different devices, and consequently will minimize LOD mismatch. Therefore, to minimize LOD mismatch if the device has already been placed in the column (in a different row), other devices are prioritized over this one (lines 47–52 and 58–63).

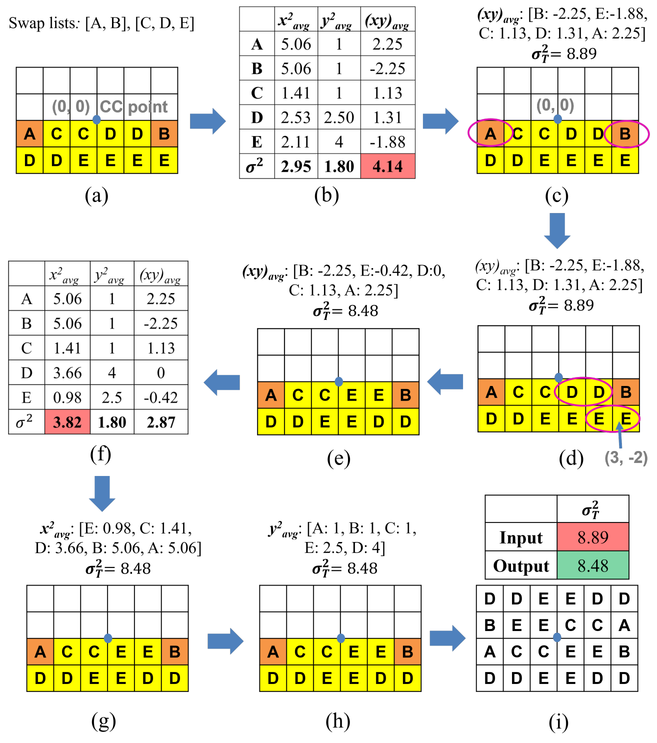
For example, in the current mirror bank testcase, first, device C is selected: at this point, no device can share the diffusion region, and C is the device with the highest $Ratio$ value. Its placement in X is shown in Fig. 4(d). Thereafter, $Ratio$ is updated, and device D, which now has the largest value in $Ratio$ and can share the diffusion region with the device C, is placed as shown in Fig. 4(e). Further, for the next two available locations (as circled in Fig. 4(f)–(g)), only devices C and D can share the diffusion, therefore, placed as shown in Fig. 4(f)–(g). At this point, the row is filled and we move to the next row. The procedure is repeated until all cells are placed, as shown in Fig. 4(h)–(i).

Step 5: Postprocessing The algorithm, as explained so far, places half of the devices (in M_{half}) in the lower part array. The remaining half of the devices are reflected around the CC point in X. The reflection is carried out about a horizontal line through the CC point. If the number of rows is odd, an additional step is required for the row in the middle: its left half is mirrored on to the right half to create CC symmetry. This is illustrated in Fig. 4(j) (line 73).

Finally, the maximum threshold voltage mismatch between two devices, $\Delta V_{th,LOD}^{max}$, due to the LOD effect is calculated using (9). The SA/SB values for each unit cell are first calculated from the placement, and thereafter (9) is used to calculate $\Delta V_{th,LOD}^{max}$. The mismatch can be minimized using dummies on the left/right of X (this will increase SA and SB as shown in Fig. 2, and consequently will reduce $\Delta V_{th,LOD}^{max}$). To minimize $\Delta V_{th,LOD}^{max}$ within $\varepsilon \cdot V_{th}$ (ε is a user-defined tolerance) the values of SA/SB are calculated using (9), and the required number of dummy unit cells on the left/right of X are calculated to meet the SA/SB criteria. WPEs are also best addressed through the use of dummy cells that ensure a minimum distance to the well edge.

IV. CC Placement to Minimize Nonlinear Gradients

In this section we present a constructive approach to refine the CC placement of our proposed algorithm to minimize the impact of second-order gradients. In principle, the approach is extensible to higher than second-order gradients, but in practice it is unlikely that such an analysis is necessary in real-world scenarios. Our approach refines the output of our CC placement algorithm to minimize the impact of second-order gradients while maintaining the CC nature of the placement, and maintaining the advantages of the original CC placement (e.g., diffusion-sharing and CC placement). The proposed algorithm tries to meet the criteria (6)–(8) by swapping unit cell locations within the array.



	x^2_{avg}	y^2_{avg}	$(xy)_{avg}$
A	5.06	1	2.25
B	5.06	1	-2.25
C	1.41	1	1.13
D	2.53	2.50	1.31
E	2.11	4	-1.88
σ^2	**2.95**	**1.80**	**4.14**

	x^2_{avg}	y^2_{avg}	$(xy)_{avg}$
A	5.06	1	2.25
B	5.06	1	-2.25
C	1.41	1	1.13
D	3.66	4	0
E	0.98	2.5	-0.42
σ^2	**3.82**	**1.80**	**2.87**

	σ_T^2
Input	8.89
Output	8.48

Figure 5: The application of the proposed algorithm on a current mirror bank: (a) Input placement from the CC algorithm. (b) Initial lists based on the x_{avg}^2, y_{avg}^2, and $(xy)_{avg}$, and σ^2 values for the input placement. (c) Sorted list $(xy)_{avg}$, and an illustration of the layout, showing the devices that can be swapped. (d) The CC layout, showing unit cells that can be swapped. (e) Updated placement and the updated list $(xy)_{avg}$ after swapping. (f) Updated lists x_{avg}^2, y_{avg}^2, and $(xy)_{avg}$ and their σ^2 values after the swapping. (g) Sorted list x_{avg}^2 and input placement to minimize the component. (h) Sorted list y_{avg}^2 and input placement to minimize the component. (i) Final output of the algorithm after reflection around the CC point.

Algorithm 2 Placement optimization considering nonlinear gradients

```
1: Input: Placement $X_{half}$ of transistors from Algorithm 1; analog cell netlist; Device size vector $M = [M_1, M_2, \cdots, M_N]$, where, $M_i$ is the number of unit cells for device $i$
2: Output: Refined placement to minimize the impact of second-order gradients
3:
4: **Function** DefineLists ($X_{half}$)
5: $x^2_{avg} \leftarrow []$, $y^2_{avg} \leftarrow []$, $(xy)_{avg} \leftarrow []$
6: **for** $i = 1$ to $N$ **do**
7:     $E_X = \frac{2}{M_i} \sum_{M_i} x_j^2$ // Average of $x^2$ from over all unit cells of device $i$
8:     $E_Y = \frac{2}{M_i} \sum_{M_i} y_j^2$ // Average of $y^2$ from over all unit cells of device $i$
9:     $E_{XY} = \frac{2}{M_i} \sum_{M_i} x_j y_j$ // Average of $xy$ over all unit cells of device $i$
10:    $x^2_{avg}$.add($E_X$), $y^2_{avg}$.add($E_Y$), $(xy)_{avg}$.add($E_{XY}$)
11: **end for**
12: $\sigma^2_T = \sigma^2(x^2_{avg}) + \sigma^2(y^2_{avg}) + \sigma^2((xy)_{avg})$ // Optimization function
13: **return** $\sigma^2_T$
14: **EndFunction**
15:
16: **Function** Swap (MaxList, $X_{half}$, $U$, $M_{half}$)
17: DevList $\leftarrow$ [] This list will store deviation from the mean for devices
18: **for** $k = 1$ to len(MaxList) **do**
19:     DevList.add(|MaxList[$k$]$-\mu$(MaxList)|)
20: **end for**
21: Sort DevList in descending order
22: **for** $i = 1$ to len(DevList) **do** // Select device in order of max. deviation from mean
23:     **for** $j =$ len(DevList) to $i + 1$ **do**
24:         **if** (DevList[$i$] and DevList[$j$] $\in U$) OR (DevList[$i$] and DevList[$j$] $\in M_{half}$) **then** // belong to the same swap list
25:             **repeat**
26:                 $X_c = X_{half}$ // Make a local copy of $X_{half}$
27:                 $\sigma^2_{old}$ = DefineLists($X_c$)
28:                 Swap the unit cells of device DevList[$i$] and DevList[$j$] in $X_{half}$
29:                 $\sigma^2_{new}$ = DefineLists($X_{half}$)
30:                 **if** $\sigma^2_{new} \geq \sigma^2_{old}$ **then**
31:                     $X_{half} = X_c$ // Restore $X_{half}$
32:                 **end if**
33:             **until** no swap possible between unit cells of device DevList[$i$] and DevList[$j$]
34:         **end if**
35:     **end for**
36: **end for**
37: **EndFunction**
38:
39: main()
40: // Step 1: Define swap lists
41: Use Step 1 (lines 4-14) of Algorithm 1 to define swap lists $U$ and $M_{half}$
42: // Step 2: Initialize $(xy)_{avg}$, $x^2_{avg}$, $y^2_{avg}$ lists; compute $\sigma^2_T$
43: $\sigma^2_T$ = DefineLists($X_{half}$)
44: Calculate $\sigma^2(x^2_{avg})$, $\sigma^2(y^2_{avg})$, and $\sigma^2((xy)_{avg})$
45: // Step 3: Swap unit cells, update (10) for Stage 1 of Step 3
46: **repeat**
47:     MaxList = List (out of $xy$, $x^2$, $y^2$) with the maximum value of $\sigma^2$
48:     Swap(MaxList, $X_{half}$, $U$, $M_{half}$)
49:     $\sigma^2_T$ = DefineLists($X_{half}$) // Update lists and $\sigma^2_T$
50:     Calculate $\sigma^2(x^2_{avg})$, $\sigma^2(y^2_{avg})$, and $\sigma^2((xy)_{avg})$ // Stage 2 of Step 3
51:     MaxList = List with the maximum value of $\sigma^2$ and not used in Stage 1
52:     Swap(MaxList, $X_{half}$, $U$, $M_{half}$) // Stage 3 of Step 3
53:     $\sigma^2_T$ = DefineLists($X_{half}$) // Update lists and $\sigma^2_T$
54:     Calculate $\sigma^2(x^2_{avg})$, $\sigma^2(y^2_{avg})$, and $\sigma^2((xy)_{avg})$
55:     MaxList = List not used in Stages 1 and 2
56:     Swap(MaxList, $X_{half}$, $U$, $M_{half}$)
57: **until** no swap possible in the current iteration
58: // Step 4: Postprocessing
59: Use Step 5 (lines 73–75) of Algorithm 1
```

For a given cell netlist with the number of unit cells listed for each device, the input to the algorithm is the symmetric half placement pattern from our CC algorithm, generated after Step 4 of Algorithm 1. We will explain the algorithm using half of the pattern, X_{half}. This approach refines the CC layout of X_{half} in four steps: (1) while maintaining the CC property of the layout, we generate *swap lists* of candidate unit cells that may be swapped to reduce the impact of nonlinear gradients; (2) we determine an objective function that quantifies the impact of nonlinear gradients; (3) we choose cells from the swap list that should be exchanged to reduce the magnitude of the objective function; and (4) we perform a final postprocessing step that reflects X_{half} about the CC point to obtain the full CC layout.

We continue with the example from Fig. 4(a)), and begin with X_{half} as shown in Fig. 5. The four steps of the algorithm are described below for this example:

Step 1: Creating swap lists. (lines 40–41) Since the main idea of our approach is to swap the unit cells of devices to meet the criteria in (6)–(8) in order to minimize the impact of second-order gradients, we create a list of devices that are good candidates for swapping. These candidates are chosen so as to preserve the good properties of the CC placement (e.g., diffusion-sharing).

For example, in Fig. 5(a), the unit cells of A and B can be swapped without creating a new diffusion break, and the same is true of the unit cells of and C, D, and E. These swap lists are the same as those created after the preprocessing step in Algorithm 1 (i.e., lists U and M_{half} in Step 1). For our testcase, the swap lists (i.e., devices in U and M_{half}) are shown in Fig. 5(a). The unit cells of the devices can be swapped only if they are in the same list as highlighted in the same color in Fig. 5.

Step 2: Computing the objective function. Equations (6)–(8) show that the impact of second-order gradients can be canceled if each device has same average value of x^2, y^2, and xy. Therefore, the objective function attempts to minimize the difference between the average values of x^2, y^2, and xy for all devices. We capture this objective using the variance, σ^2, in the distributions of x^2, y^2, and xy, over all devices, to minimize the impact of second-order gradients based on the following optimization objective function (line 12):

$$\sigma^2_T = \sigma^2(x^2_{avg}) + \sigma^2(y^2_{avg}) + \sigma^2((xy)_{avg}) \qquad (10)$$

where x^2_{avg}, y^2_{avg}, and $(xy)_{avg}$ are the average values of x^2, y^2, and xy, respectively, for all devices, where the average is computed over all unit cells in each device. These lists are created as follows (lines 6–11): for list x^2_{avg}, for each device, we calculate the average value of x^2 from the CC point and store the value in a list; similarly lists y^2_{avg} and $(xy)_{avg}$ are created based on average value of y^2 and xy, respectively.[1] The above optimization criterion equally weights components associated with x, y, and xy because for a given layout on a specific die on a wafer, we have no further information on which of these is most important. If such information is available, the function can be appropriately weighted.

For example, Fig. 5(b) shows the lists x^2_{avg}, y^2_{avg}, and $(xy)_{avg}$ for our testcase. The figure also shows the variance for each list: $\sigma^2(x^2_{avg}) = 2.95$, $\sigma^2(x^2_{avg}) = 1.80$, and $\sigma^2((xy)_{avg}) = 4.14$.

Step 3: Swapping unit cells to optimize the objective function. Next, we reduce the objective function (10) by swapping the unit cells of devices. We examine the three components of the objective function, and in each of three stages, we choose the swap criterion to be based on the largest

[1]Note that these lists correspond to the variances in the coordinates over all unit cells of each device in the circuit, and should not be confused with the swap lists in Step 1.

of these components. For example, if $\sigma^2((xy)^2_{avg})$ is the largest component, as in our example, we first use $(xy)_{avg}$ as the swap criterion; after these swaps are complete, we update the two remaining components (here, $\sigma^2(x^2_{avg})$ and $\sigma^2(y^2_{avg})$) and choose the larger of the two as the swap criterion, and then finally, we swap to optimize the remaining component. These three stages are repeated iteratively until no further improvements are possible (lines 46–57).

In each stage, we process the devices in the order of their distance from the mean (lines 17–21). In our example, when the swap criterion is $(xy)_{avg}$, we process the device whose unit cells are farthest from the mean of the $(xy)_{avg}$ list to reduce $\sigma^2((xy)_{avg})$. For each such device, we consider swaps with every other device on the list that has not already been processed; obviously, the candidate device must be on the same swap list (line 22). In our example, the devices that is first chosen is A (ties are broken arbitrarily), and the only other device on its swap list is B. Having selected the devices, next we select the unit cells of the two devices that will be swapped. Having selected the devices, next we select the unit cells of the two devices that will be swapped (lines 25–33). We select one or two unit cells to be swapped, minimizing the number of additional diffusion breaks.

For our example, the unit cells of devices A and B are highlighted in Fig. 5(c). In this case, it is clear that since they are symmetrically placed, swapping these devices will not alter the $(xy)_{avg}$ value of either device.

We then go to the next highest distance from the mean on the $(xy)_{avg}$ list, which corresponds to device E on the example and consider swaps with the unit cells of C and D, which lie on its swap list. As an example, let us consider swaps between the unit cells of E and D. In this case, selecting one unit cell will create a diffusion break, and we select pairs of unit cells for swapping. Let us consider the unit cells of E at $(2, -2), (3, -2)$ as candidates for swapping with unit cells of D at $(1, -1), (2, -1)$, as circled in Fig. 5(d). Swapping of these unit cells reduces the value of σ^2_T from 8.89 to 8.48 as shown in Fig. 5(e). The remaining candidate swaps between the unit cells of E and D are considered using the updated placement: we find that no further swaps will improve σ^2_T. The values of the components of (10) are then updated (Fig. 5(f)), and the stage using $(xy)_{avg}$ as the swap criterion is complete.

We then move to the next higher component of (10), which is x^2_{avg}, and repeat the swap procedure using this swap criterion. As shown in Fig. 5(g), no further improvements are achievable for this example. Finally, we move to the remaining component, y^2_{avg}, and it is found that no swaps can improve (10) here either.

Step 4: Postprocessing. Finally, a postprocessing step, similar to Algorithm 1 employed. The final placement is reflected around the CC point, and the final result shown in Fig. 5(i). We can see from Fig. 5(i) that our algorithm reduces σ^2_T from 8.89 to 8.48 ($\sim 5\%$ reduction) for this example.

V. Results and Discussion

Components of analog circuits such as OTAs, comparators, and DACs commonly contain transistor groups such as current mirrors and differential pairs, which use CC layout techniques. In this section, we apply our placement algorithms on a set of analog cells: current mirror banks and cascoded differential pairs. These algorithms are also valid for other analog cells that are mismatch sensitive and require special layouts – cross-coupled-pairs, differential and cascoded loads, etc. We present a qualitative comparison of our approach with prior methods for several circuit examples, highlighting the advantages of our approach, and also show the post-layout simulation results for a subset of these circuits. Our layout performs both placement, using the methods described in this paper, and routing, using the methods described in the precursor paper, [22].

A. Qualitative Comparison

We begin by validating our placement algorithms using test-cases shown in Fig. 6–7. We compare our algorithms with the algorithms presented in [19], [22]; recall that [22] is a preliminary version of this work. The results are compared based on four figures-of-merit (FOMs) that were discussed in Section I and II:

(1) Tolerance of linear systematic variations: this checks whether the placement is CC.
(2) Tolerance of systematic nonlinear variations: this checks whether the placement can minimize nonlinear gradients by comparing the maximum value of threshold voltage mismatch between the devices, $\triangle V_{th}^{max}$, due to systematic variations only. A placement with the smaller value of $\triangle V_{th}^{max}$ will be better in minimizing nonlinear gradients. We use first- and second-order gradient values characterized from a test chip [23] to calculate $\triangle V_{th}^{max}$.
(3) Diffusion-sharing: this checks whether diffusion-sharing is maximized.
(4) LDE: this considers whether the placement considers the impact of LDE.

We consider several testcases:

Four- and five-device CMBs. In Fig. 6, three testcases with current mirror banks (CMBs) are shown: two with four devices and one with five devices. A critical performance metric for a CMB is the current ratio, and it is degraded significantly by $\triangle V_{th}^{max}$ between the devices. LDEs and linear/nonlinear gradient based variations are the main cause of V_{th} mismatch. In our approach we incorporate these during placement by positioning the devices appropriately and adding dummies where necessary. Finally, diffusion-sharing is also important for CMBs as it reduces area and output capacitance, which will be critical for high-speed designs.

Fig. 6(a)–(c) show the layouts for the three testcases. The placements employing our proposed approach and algorithms in [19], [22] are also shown in the figure. We compare the four FOMs for these approaches.

• All approaches use *CC* placement and routing, and hence the results are tolerant to first-order systematic variations. However, the algorithms in [19], [22] do not consider nonlinear gradients.

• In all three testcases, the placement using [19] has diffusion breaks between devices (shown by shaded cells in Figs. 6(a)–(c)), which results in an increase in area and higher parasitic at the corresponding nodes.

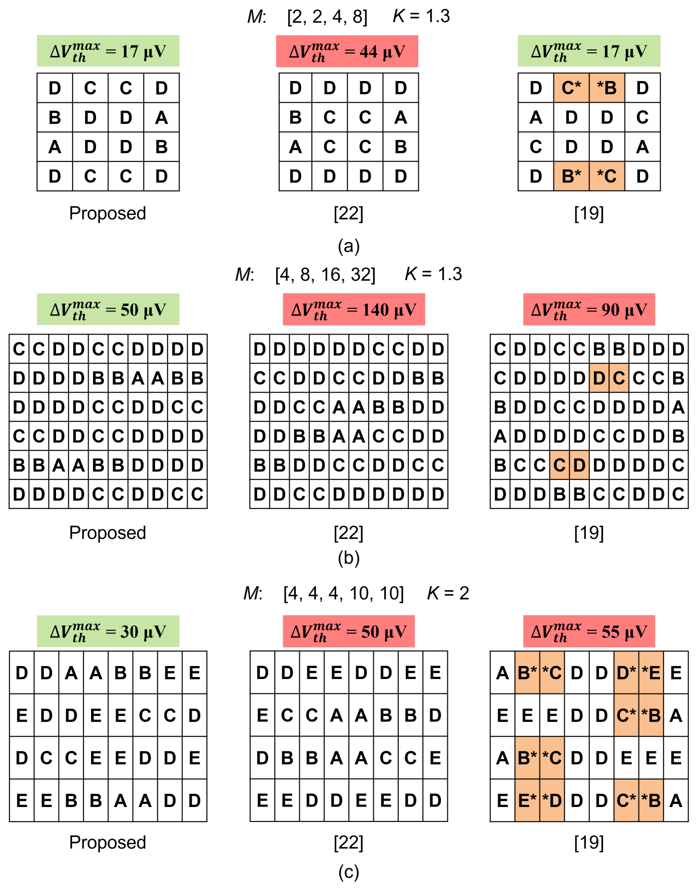

M: [2, 2, 4, 8] K = 1.3
ΔV_{th}^{max} = 17 µV
ΔV_{th}^{max} = 44 µV
ΔV_{th}^{max} = 17 µV
Proposed
[22]
[19]
(a)
M: [4, 8, 16, 32] K = 1.3
ΔV_{th}^{max} = 50 µV
ΔV_{th}^{max} = 140 µV
ΔV_{th}^{max} = 90 µV
Proposed
[22]
[19]
(b)
M: [4, 4, 4, 10, 10] K = 2
ΔV_{th}^{max} = 30 µV
ΔV_{th}^{max} = 50 µV
ΔV_{th}^{max} = 55 µV
Proposed
[22]
[19]
(c)

Figure 6: Testcases with (a), (b) four-device and (c) five-device current mirror banks. The proposed algorithm is compared with the results of [19], [22]. The shaded cells show locations where diffusion breaks must be inserted.

• Fig. 6(a) shows a four-device CMB where $M = [2, 2, 4, 8]$, $K = 1.3$. The placement using [19] has higher parasitic due to diffusion breaks and LDE mismatch, and will consequently see a current ratio mismatch.

• Fig. 6(c) shows a five-device CMB, with $M = [4, 4, 4, 10, 10], K = 2$, and its placements generated using the proposed technique and the algorithms in [19], [22]. The placement using [19] has LDE and higher parasitic. In our approach, LDE mismatch is minimized further by adding one column of unit cells corresponding to dummy transistors on the left and right sides, and parasitic are less due to diffusion sharing.

• For all three testcases, the placement from this work reduces $\triangle V_{th}^{max}$ due to nonlinear gradients by $\sim 2-3\times$ as compared to [19], [22].

A cascoded differential pair. Finally, Fig. 7 shows placements of a cascoded differential pair, obtained by employing our proposed approach and that used in [18], [22]. Note that since the algorithm presented in [19] is specific to current mirrors, and is not applicable for this testcase; therefore, we use another prior work presented in [18] instead. For this circuit, the absolute and mismatch parasitic resistances are critical as they impact the effective transconductance (G_m) and input offset. We consider this by limiting the value of resistive parasitics.

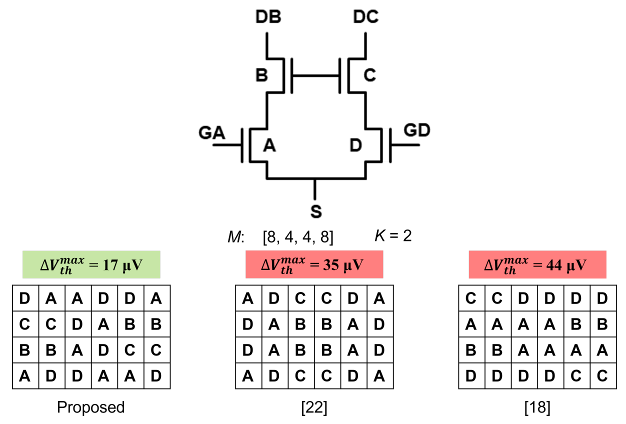

DB
DC
B
C
GA
A
D
GD
S
M: [8, 4, 4, 8] K = 2
ΔV_{th}^{max} = 17 µV
ΔV_{th}^{max} = 35 µV
ΔV_{th}^{max} = 44 µV
Proposed
[22]
[18]

Figure 7: Comparing the proposed algorithm with the results of [18], [22] on a cascoded differential pair testcase.

As before, diffusion-sharing is critical as it reduces output capacitance. We can also see that our placement from this work reduces $\triangle V_{th}^{max}$ due to nonlinear gradients by $\sim 2\times$ as compared to [18], [22].

B. Impact of second-order gradients

We now consider larger testcases to show the impact of second-order gradients. We select six-, eight-, and ten-device CMBs which are used in binary-weighted DACs. We compare the value of $\triangle V_{th}^{max}$ and the maximum percentage deviation in current ratio compared to the ideal value for the placements generated using each approach. As mentioned before, we use first- and second-order gradient trends similar to those characterized from a test chip [23] to calculate $\triangle V_{th}^{max}$. For layouts optimized for LDE effects, the impact of nonlinear gradients is the most significant contributor for large layouts. Therefore, in this analysis, we focus purely on the impact of nonlinear gradients on threshold voltage shift. We use the following I–V characteristic for long-channel devices used in CMBs to calculate the current:

$$I_D = (\beta/2)(V_{GS} - V_{th})^2(1 + \lambda V_{DS}) \qquad (11)$$

where V_{GS} is the gate-source voltage, V_{DS} is the drain-source voltage, β is the current factor, and λ is the channel length modulation factor.

Table I shows a comparison of $\triangle V_{th}^{max}$ and the maximum percentage deviation in current ratio for six-, eight- and ten-device CMBs for the placements generated using the proposed approach and the algorithms presented in [19], [22]. In the table, we also show the standard deviation of the uncorrelated random variation for the smallest device (with four unit cells) used in the CMBs, based on Pelgrom's model [3]. Please note that we did not use the technique presented in [18] for a comparison for the testcases in Table I due to their large size. As stated in Section I, the approach in [18] enumerates all Euler paths in the graph. This enumeration scales poorly for testcases with a large number of unit cells, and therefore the method is unable to handle these testcases.

We can observe from the table that the impact of second-order gradients increases significantly with the layout size.

The placements generated using [19], [22] have similarly high values of $\triangle V_{th}^{max}$ and the maximum deviation in current ratio compared to the proposed work as shown in Table I. For example, for a ten-device CMB the proposed work reduces the maximum deviation in the current ratio to 1.54% which is $\sim 14 - 17\times$ less compared to the placements generated using algorithms in [19], [22]. The table also shows that while $\triangle V_{th}^{max}$ from [19], [22] is larger than the uncorrelated random variation, its value is reduced to be lower than the uncorrelated random variation in our approach. For reference, we show a six-device CMB testcase and its placements generated using the proposed technique and the algorithms in [19], [22] in Fig. 8.

Table I: Comparison of CMBs with nonlinear gradients.

CMB multiplicity vector M	$\triangle V_{th}^{max}$ (mV)			Max. deviation in current ratio (%)			Uncorrelated
	Proposed	[22]	[19]	Proposed	[22]	[19]	random σ (mV)
[4:8:16:32:64:128]	0.14	1.66	1.67	0.15	1.84	1.90	1.34
[4:8:16:32:64:128:256:512]	0.14	6.58	5.48	0.16	7.18	6.50	1.34
[4:8:16:32:64:128:256:512:1024:2048]	1.20	25.52	16.87	1.54	26.35	21.89	1.34

The runtime of our proposed algorithm for the CMB test-cases are shown in Table II. These runtimes are measured on a personal computer with an Intel(R) Core i7 8665U CPU @1.90 GHz and indicate that the method scales sublinearly with the number of unit cells.

Table II: Runtime for the proposed algorithm for CMBs.

# devices	6	8	10
# unit cells	252	1020	4092
CPU time	8s	79s	314s

C. Post-layout Simulation Results

In the previous sub-sections, we have shown that our proposed placement is better than the algorithms presented in [19], [22] in the presence of second-order gradients. Here, we show that our proposed placement is comparable to [22] and better than [18], [19] in the presence of all variations: linear/nonlinear systematic variations, LDEs, and routing parasitics. For this experiment, we use the same routing technique as in our work [22] and compare the layouts in a commercial 12nm FinFET process. Post-layout simulation results are necessary to show the impact of LDEs and parasitics on the layout, and these are presented for two CMBs and a cascoded differential pair (DP) in Table III.

Table III: Post-layout simulations results for a cascoded differential pair (DP) and two CMBs.

Specification	Proposed	[22]	[19]	[18]
Fig. 8(a): **CMB1** (Schematic: M: [2: 2: 4: 8])				
Current ratio	2.00:2.00:3.99:7.99	2.00:2.00:4.00:7.98	2.00:1.99:3.99:7.60	2:2.01:3.67:7.88
Max. deviation (%)	–0.25	–0.25	–5.00	–8.25
Max. IR drop (mV)	1.7	1.8	4.0	3.7
Fig. 8(c): **CMB2** (Schematic: M: [4: 4: 4: 10: 10])				
Current ratio	4.00: 4.00: 3.99: 9.99: 10.01	4.00:3.97:3.97:9.98:9.98	4.00:4.02:4.11:9.61:10.10	4:3.64:3.61:9.39:9.48
Max. deviation (%)	–0.25	–0.75	–3.90	–9.75
Max. IR drop (mV)	5.4	5.7	19.5	18.5
Fig. 10: **Cascoded DP** (Schematic: G_m = 825μA/V; Offset = 0V; Output capacitance (C) = 0.8fF)				
G_m(μA/V)	804	764	NA	786
Offset (μV)	7	13	NA	899
C (fF)	1.26	1.30	NA	1.35

For CMB1, four-device CMB in Fig. 6(a), and CMB2, five-device CMB in Fig. 6(c), our proposed placement is better than in [18], [19] since it considers LDEs, has no diffusion breaks and has better routing parasitics (due to no diffusion breaks). In addition, our placement is comparable to [22] while considering LDEs and parasitics and better when considering second-order gradients (shown in the previous sub-section). We are able to achieve current ratios close to the ideal value while considering LDEs as shown in Table III. We also tabulate the maximum deviation of the current ratio from the ideal value for any device in the array and this is -0.25% for CMB1 and CMB2.

We also show results for the cascoded differential pair (DP) in Fig. 7 and compare the input-referred offset, G_m, and output capacitance. The algorithms presented in [19] can only be applied to current mirror banks, therefore, it is not applicable to generate the placement for the cascoded differential pair. Table III shows that our G_m and output capacitance are in a similar range as [18], while the offset is greatly improved.

VI. Conclusion

This paper develops an approach for constructively creating common-centroid layouts, accounting for linear and nonlinear process gradients, layout-dependent effects, and minimizing parasitics by maximimizing diffusion sharing. The approach is applied to FinFET technology nodes to create common-centroid layouts of arrays of unit cells of transistors. The method is applied to several commonly-encountered analog cells – a differential pair, current mirror banks, and a cascoded cell – and significant improvements over prior methods are demonstrated.

References

[1] K. Qian, "Variability modeling and statistical parameter extraction for CMOS devices," Ph.D. dissertation, University of California–Berkeley, Berkeley, CA, 2015.

[2] Y. A. Lin, "Parametric wafer map visualization," *IEEE Computer Graphics and Applications*, vol. 19, no. 4, pp. 14–17, 1999.

[3] M. Pelgrom, H. Tuinhout, and M. Vertregt, "Modeling of MOS matching," *Compact Modeling: Principles, Techniques and Applications*, pp. 453–490, 2010.

[4] A. Hastings, *The Art of Analog Layout*. Upper Saddle River, NJ: Prentice-Hall, 2001.

[5] N. Karmokar, M. Madhusudan, A. K. Sharma, R. Harjani, M. P. Lin, and S. S. Sapatnekar, "Common-centroid layout for active and passive devices: A review and the road ahead," in *Proceedings of the Asia-South Pacific Design Automation Conference*, 2022, pp. 114–121.

[6] D. Blaauw, K. Chopra, A. Srivastava, and L. Scheffer, "Statistical timing analysis: From basic principles to state of the art," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 4, pp. 589–607, 2008.

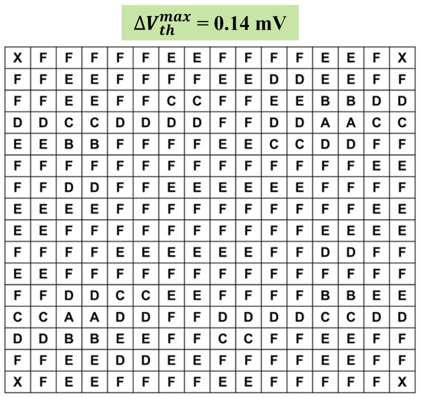
[7] S. S. Sapatnekar, "Overcoming variations in nanometer-scale technologies," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 1, pp. 5–18, 2011.

[8] M. J. Pelgrom, A. C. J. Duinmaijer, and A. G.Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid-State Circuits*, vol. 24, no. 5, pp. 1433–1439, 1989.

[9] K. Kunal, M. Madhusudan, A. K. Sharma, W. Xu, S. M. Burns, R. Harjani, J. Hu, D. A. Kirkpatrick, and S. S. Sapatnekar, "ALIGN: Open-source analog layout automation from the ground up," in *Proceedings of the ACM/IEEE Design Automation Conference*, 2019, pp. 77–80.

[10] T. Dhar, K. Kunal, Y. Li, M. Madhusudan, J. Poojary, A. K. Sharma, W. Xu, S. M. Burns, R. Harjani, J. Hu, D. A. Kirkpatrick, P. Mukherjee, S. Yaldiz, and S. S. Sapatnekar, "ALIGN: A system for automating analog layout," *IEEE Design & Test*, vol. 38, no. 2, pp. 8–18, 2020.

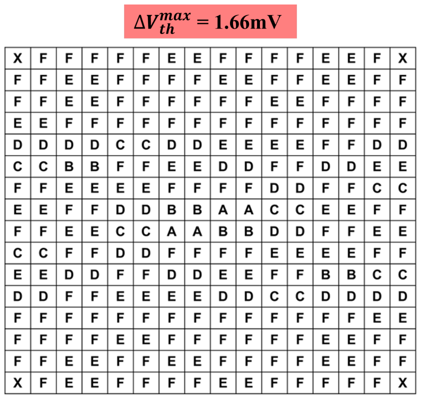
[11] M. Dessouky and D. Sayed, "Automatic generation of common-centroid capacitor arrays with arbitrary capacitor ratio," in *Proceedings of Design, Automation and Test in Europe*, 2002, pp. 576–580.



$\Delta V_{th}^{max} = 0.14$ mV

X	F	F	F	F	F	E	E	F	F	F	F	E	E	F	X
F	F	E	E	F	F	F	F	E	E	D	D	E	E	F	F
F	F	E	E	F	F	C	C	F	F	E	E	B	B	D	D
D	D	C	C	D	D	D	D	F	F	D	D	A	A	C	C
E	E	B	B	F	F	F	F	E	E	C	C	D	D	F	F
F	F	F	F	F	F	F	F	F	F	F	F	F	F	E	E
F	F	D	D	F	F	E	E	E	E	E	E	F	F	F	F
E	E	E	E	F	F	F	F	F	F	F	F	F	F	E	E
E	E	F	F	F	F	F	F	F	F	F	F	E	E	E	E
F	F	F	F	E	E	E	E	E	E	F	F	D	D	F	F
E	E	F	F	F	F	F	F	F	F	F	F	F	F	F	F
F	F	D	D	C	C	E	E	F	F	F	F	B	B	E	E
C	C	A	A	D	D	F	F	D	D	D	D	C	C	D	D
D	D	B	B	E	E	F	F	C	C	F	F	E	E	F	F
F	F	E	E	D	D	E	E	F	F	F	F	E	E	F	F
X	F	E	E	F	F	F	F	E	E	F	F	F	F	F	X

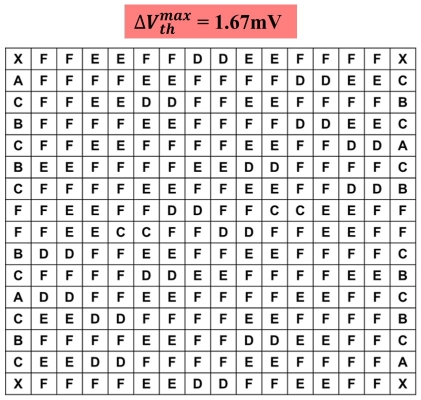
(a) Proposed



$\Delta V_{th}^{max} = 1.66$mV

X	F	F	F	F	F	E	E	F	F	F	F	E	E	F	X
F	F	E	E	F	F	F	F	E	E	F	F	E	E	F	F
F	F	E	E	F	F	F	F	F	F	E	E	F	F	F	F
E	E	F	F	F	F	F	F	F	F	F	F	F	F	F	F
D	D	D	D	C	C	D	D	E	E	E	E	F	F	D	D
C	C	B	B	F	F	E	E	D	D	F	F	D	D	E	E
F	F	E	E	E	E	F	F	F	F	D	D	F	F	C	C
E	E	F	F	D	D	B	B	A	A	C	C	E	E	F	F
F	F	E	E	C	C	A	A	B	B	D	D	F	F	E	E
C	C	F	F	D	D	F	F	F	F	E	E	E	E	F	F
E	E	D	D	F	F	D	D	E	E	F	F	B	B	C	C
D	D	F	F	E	E	E	E	D	D	C	C	D	D	D	D
F	F	F	F	F	F	F	F	F	F	F	F	F	F	E	E
F	F	F	F	E	E	F	F	F	F	F	F	E	E	F	F
F	F	E	E	F	F	E	E	F	F	F	F	E	E	F	F
X	F	E	E	F	F	F	F	E	E	F	F	F	F	F	X

(b) [22]



$\Delta V_{th}^{max} = 1.67$mV

X	F	F	E	E	F	F	D	D	E	E	F	F	F	F	X
A	F	F	F	F	E	E	F	F	F	F	D	D	E	E	C
C	F	F	E	E	D	D	F	F	E	E	F	F	F	F	B
B	F	F	F	F	E	E	F	F	F	F	D	D	E	E	C
C	F	F	E	E	F	F	F	F	E	E	F	F	D	D	A
B	E	E	F	F	F	F	E	E	D	D	F	F	F	F	C
C	F	F	F	F	E	E	F	F	E	E	F	F	D	D	B
F	F	E	E	F	F	D	D	F	F	C	C	E	E	F	F
F	F	E	E	C	C	F	F	D	D	F	F	E	E	F	F
B	D	D	F	F	E	E	F	F	E	E	F	F	F	F	C
C	F	F	F	F	D	D	E	E	F	F	F	F	E	E	B
A	D	D	F	F	E	E	F	F	F	F	E	E	F	F	C
C	E	E	D	D	F	F	F	F	E	E	F	F	F	F	B
B	F	F	F	F	E	E	F	F	D	D	E	E	F	F	C
C	E	E	D	D	F	F	F	F	E	E	F	F	F	F	A
X	F	F	F	F	E	E	D	D	F	F	E	E	F	F	X

(c) [19]

Figure 8: Comparing the proposed algorithm with [19], [22] on a six-device current mirror with M: [4, 8, 16, 32, 64, 128], $K = 1$.

[12] C. W. Lin, J. M. Lin, Y. C. Chiu, C. P. Huang, and S. J. Chang, "Mismatch-aware common-centroid placement for arbitrary-ratio capacitor arrays considering dummy capacitors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 12, pp. 1789–1802, 2012.

[13] F. Burcea, H. Habal, and H. E. Graeb, "A new chessboard placement and sizing method for capacitors in a charge-scaling DAC by worst-case analysis of nonlinearity," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1397–1410, 2015.

[14] P.-Y. Chou, N.-C. Chen, M. P.-H. Lin, and H. Graeb., "Matched-routing common-centroid 3-D MOM capacitors for low-power data converters," *IEEE Transactions on VLSI Systems*, vol. 25, no. 8, pp. 2234–2247, 2017.

[15] C. C. McAndrew, "Layout symmetries: Quantification and application to cancel nonlinear process gradients," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 1, pp. 1–14, 2016.

[16] V. Borisov, K. Langner, J. Scheible, and B. Prautsch, "A novel approach for automatic common-centroid pattern generation," in *Proceedings of the IEEE International Conference on Synthesis, Modeling, Analysis and Simulation Methods, and Applications to Circuit Design*, 2017, 4 pages.

[17] M. P.-H. Lin, H. Zhang, M. D. F. Wong, and Y.-W. Chang, "Thermal-driven analog placement considering device matching," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 3, pp. 325–336, 2011.

[18] D. Long, X. Hong, and S. Dong, "Optimal two-dimension common centroid layout generation for MOS transistors unit-circuit," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, 2005, pp. 2999–3002.

[19] P. H. Wu, M. P. H. Lin, X. Li, and T. Y. Ho, "Parasitic-aware common-centroid FinFET placement and routing for current-ratio matching," *ACM Transactions on Design Automation of Electronic Systems*, vol. 21, no. 3, pp. 39:1–39:12, 2016.

[20] N. Karmokar, A. K. Sharma, J. Poojary, M. Madhusudan, R. Harjani, and S. S. Sapatnekar, "Constructive common-centroid placement and routing for binary-weighted capacitor arrays," in *Proceedings of Design, Automation and Test in Europe*, 2022, pp. 166–171.

[21] X. Dai, C. He, H. Xing, D. Chen, and R. Geiger, "An nth order central symmetrical layout pattern for nonlinear gradients cancellation," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, 2005, pp. 4835–4838.

[22] A. K. Sharma, M. Madhusudan, S. M. Burns, Y. Soner, P. Mukherjee, R. Harjani, and S. S. Sapatnekar, "Performance-aware common-centroid placement and routing of transistor arrays in analog circuits," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, 2021, 9 pages.

[23] M. Madhusudan, J. Poojary, A. K. Sharma, Ramprasath S., K. Kunal, S. S. Sapatnekar, and R. Harjani, "Understanding distance-dependent variations for analog circuits in a FinFET technology," in *Proceedings of the IEEE European Solid-State Device Research Conference*, 2023.

[24] R. Gregor, "On the relationship between topography and transistor matching in an analog CMOS technology," *IEEE Transactions on Electron Devices*, vol. 39, no. 2, pp. 275–282, 1992.

[25] M.-F. Lan, A. Tammineedi, and R. Geiger, "Current mirror layout strategies for enhancing matching performance," *Analog Integrated Circuits and Signal Processing*, vol. 28, pp. 9–26, Jul. 2001.

[26] J. V. Faricelli, "Layout-dependent proximity effects in deep nanoscale CMOS," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, 2010, 8 pages.

[27] M. G. Bardon, V. Moroz, G. Eneman, P. Schuddinck, M. Dehan, D. Yakimets, D. Jang, G. van de Plas, A. Mercha, A. Thean, and D. Verkest, "Layout-induced stress effects in 14nm & 10nm FinFETs and their impact on performance," in *Proceedings of the IEEE International Symposium on VLSI Technology*, 2013, pp. T114–T115.

[28] T. Hook, J. Brown, P. Cottrell, E. Adler, D. Hoyniak, J. Johnson, and R. Mann, "Lateral ion implant straggle and mask proximity effect," *IEEE Transactions on Electron Devices*, vol. 50, no. 9, pp. 1946–1951, 2003.

[29] K. W. Su, Y. M. Sheu, C. K. Lin, S. J. Yang, W. J. Liang, X. Xi, C. S. Chiang, J. K. Her, Y. T. Chia, C. H. Diaz, and C. Hu, "A scaleable model for STI mechanical stress effect on layout dependence of MOS electrical characteristics," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, 2003, pp. 245–248.

[30] P. G. Drennan, M. L. Kniffin, and D. R. Locascio, "Implications of proximity effects for analog design," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, 2006, pp. 169–176.

[31] S. Yang, Y. Liu, M. Cai, J. Bao, P. Feng, and P. R. Chidi, "10nm high performance mobile SoC design and technology co-developed for performance, power, and area scaling," in *Proceedings of the IEEE International Symposium on VLSI Circuits*, 2017, pp. T70–T71.

[32] A. L. S. Loke, D. Yang, T. T. Wee, J. L. Holland, P. Isakanian, K. Rim, S. Yang, J. S. Schneider, G. Nallapati, S. Dundigal, H. Lakdawala, B. Amelifard, C. Lee, B. McGovern, P. S. Holdaway, X. Kong, and B. M. Leary, "Analog/mixed-signal design challenges in 7-nm CMOS and beyond," in *Proc. CICC*, 2018, pp. 1–8.

[33] A. K. Sharma, M. Madhusudan, S. M. Burns, P. Mukherjee, S. Yaldiz, R. Harjani, and S. S. Sapatnekar, "Common-centroid layouts for analog circuits: Advantages and limitations," in *Proceedings of Design, Automation and Test in Europe*, 2021, pp. 1224–1229.