Mismatch-Aware Common-Centroid Placement for Arbitrary-Ratio Capacitor Arrays Considering Dummy Capacitors

Cheng-Wu Lin, Jai-Ming Lin, Yen-Chih Chiu, Chun-Po Huang, and Soon-Jyh Chang, *Member, IEEE*

***Abstract*—Switched capacitors are commonly used in analog circuits to increase the accuracy of analog signal processing and lower power consumption. To take full advantage of switched capacitors, it is very important to achieve accurate capacitance ratios in the layout of the capacitor arrays, which are affected by systematic and random mismatches. A good capacitor placement should have a common-centroid structure with the highest possible degree of dispersion to mitigate mismatches. Several dummy units should be inserted to make the placement shape more square and compact. This paper proposes a simulated-annealing-based approach for mismatch-aware common-centroid placement under the above constraints. A pair-sequence representation is used to record a placement, and a couple of associated operations are developed to find better solutions. The experimental results show that the proposed placements achieve smaller oxide-gradient-induced mismatch and larger overall correlation coefficients (i.e., higher degree of dispersion) than those of previous works.**

***Index Terms*—Analog placement, capacitor array, common-centroid constraint, mismatch minimization, pair sequence.**

I. Introduction

SWITCHED-CAPACITOR (SC) circuits are a cost-effective approach to implementing various analog integrated circuits. The SC approach allows designers to replace large-area resistors with smaller SCs, which not only improves the precision of analog circuit functions [2] but also lowers power consumption. However, in order to fully take advantage of SC circuits, the capacitor size must be precisely controlled. Fig. 1 shows a biquad filter implemented using the SC technique [3]. This circuit is composed of two operational amplifiers and two capacitor arrays, namely, $\{C_1, C_2, C_3, C_4\}$ and $\{C_5, C_6\}$. The accuracy of this circuit mainly depends on the ratio of capacitances. Fig. 2 shows a successive-approximation-register analog-to-digital converter (SAR ADC), which has low power consumption and is thus widely used in biomedical chips and portable instruments. The most important component in a SAR ADC is the capacitor array, which comprises a set of capacitors $C_1, \ldots, C_{n+1}$. If these capacitors physically approach a predefined capacitance ratio (i.e., $C_1 = C_2$, $C_{i+1} = 2C_i, i = 2, \ldots, n$), the capacitor array is regarded as matched. It is very important to obtain a matched capacitor array because the linearity of the SAR ADC is mainly determined by the matching behavior.

A nominally matched capacitor array may be mismatched after integrated-circuit manufacturing. Circuit mismatches can generally be divided into two types, namely, systematic mismatches and random mismatches [4], [5].

For systematic mismatch, process biases have equal effects on identical devices. Ideally, if two devices have identical layouts, they will be matched because they suffer equal mismatches from the process biases. Therefore, given a set of devices with different sizes, designers prefer to divide them into several identical-layout-size subdevices to achieve better matching. However, the presence of process gradients also causes systematic mismatch [6]. The subdevices of a given device should exhibit symmetry in an array to average the effects induced by process gradients. In order to reduce systematic mismatch, designers usually adopt a common-centroid layout structure [7] to achieve these goals.

Random mismatch is caused by statistical fluctuations in processing conditions or material properties. The random variations in physical quantities of two identical devices can be modeled as two terms for mismatch, one that is inversely proportional to the device area and one that depends on the distance between the two devices [8]. Thus, the random mismatch can be reduced by increasing the device area. Closer devices have higher correlation in their parameters, and thus have less mismatch [9]. For a set of devices, their correlation can be determined by the dispersion degree of their subdevices. Device correlation increases with increasing dispersion degree of subdevices [9]. Therefore, the placement of subdevices in an array should exhibit the highest degree of correlation and dispersion [10] for reducing mismatch.

Fig. 3 shows the effects of various placement styles on the two types of mismatch. The capacitor array consists of four capacitors, whose capacitances have an integer ratio of 1:2:16:45. Each capacitor is further divided into several identical-layout-size unit capacitors, which are denoted by u_i, $i = 1, \ldots, 4$. Fig. 3(a) shows a placement with a

Manuscript received December 16, 2011; revised March 8, 2012 and April 26, 2012; accepted May 31, 2012. Date of current version November 21, 2012. This work was supported in part by the National Science Council of Taiwan, under Grant NSC 98-2221-E-006-156-MY3. A preliminary version of this paper was presented at the 2011 ACM/IEEE Design Automation Conference [1]. This paper was recommended by Associate Editor H. E. Graeb.

The authors are with the Department of Electrical Engineering, National Cheng Kung University, Tainan 701, Taiwan (e-mail: lcw@sscas.ee.ncku.edu.tw; jmlin@ee.ncku.edu.tw; chew@livemail.tw; gppo@sscas.ee.ncku.edu.tw; soon@mail.ncku.edu.tw).

Digital Object Identifier 10.1109/TCAD.2012.2204993

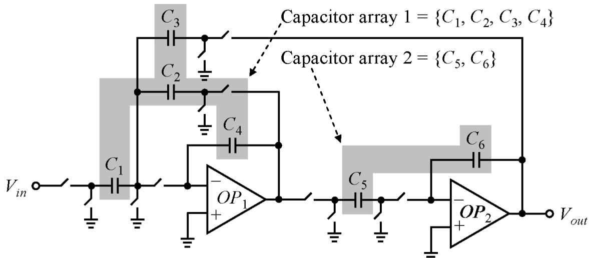

C_3
Capacitor array 1 = $\{C_1, C_2, C_3, C_4\}$
C_2
Capacitor array 2 = $\{C_5, C_6\}$
C_4
C_6
C_1
C_5
V_{in}
OP_1
OP_2
V_{out}

Fig. 1. Architecture of an SC biquad filter.

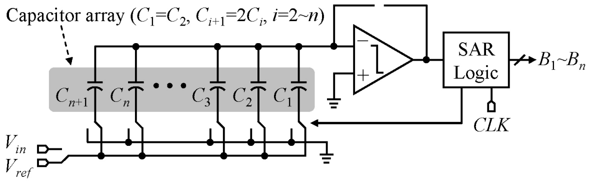
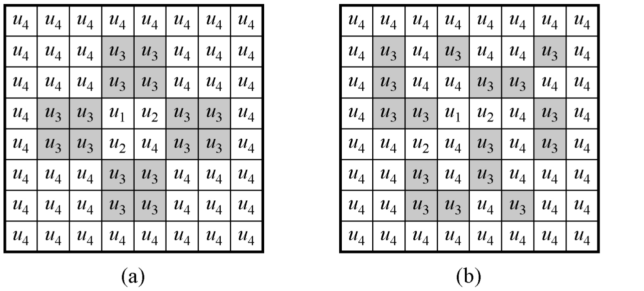

Capacitor array (C_1=C_2, C_{i+1}=2C_i, i=2~n)
C_{n+1} C_n • • • C_3 C_2 C_1
SAR Logic
B_1~B_n
CLK
V_{in}
V_{ref}

Fig. 2. Architecture of an n-bit SAR ADC.



u_4	u_4	u_4	u_4	u_4	u_4	u_4	u_4
u_4	u_4	u_4	u_3	u_3	u_4	u_4	u_4
u_4	u_4	u_4	u_3	u_3	u_4	u_4	u_4
u_4	u_3	u_3	u_1	u_2	u_3	u_3	u_4
u_4	u_3	u_3	u_2	u_4	u_3	u_3	u_4
u_4	u_4	u_4	u_3	u_3	u_4	u_4	u_4
u_4	u_4	u_4	u_3	u_3	u_4	u_4	u_4
u_4	u_4	u_4	u_4	u_4	u_4	u_4	u_4

(a)

u_4	u_4	u_4	u_4	u_4	u_4	u_4	u_4
u_4	u_3	u_4	u_3	u_4	u_4	u_3	u_4
u_4	u_3	u_4	u_4	u_3	u_3	u_4	u_4
u_4	u_3	u_3	u_1	u_2	u_4	u_3	u_4
u_4	u_4	u_2	u_4	u_3	u_4	u_3	u_4
u_4	u_4	u_3	u_4	u_3	u_4	u_4	u_4
u_4	u_4	u_3	u_3	u_4	u_3	u_4	u_4
u_4	u_4	u_4	u_4	u_4	u_4	u_4	u_4

(b)

Fig. 3. Two different placements for a four-capacitor array. Each capacitor is divided into several identical-layout-size unit capacitors u_i, i=1…4. The unit capacitors denoted by u_3 are colored in gray. (a) Common-centroid placement, which is good for reducing systematic mismatch. (b) Placement (from [10]) exhibiting a higher degree of dispersion, which is better for reducing random mismatch.

common-centroid structure. The unit capacitors are placed symmetrically with respect to the center of the layout, and thus the centroid of each capacitor is exactly on or close to the center of the layout. Compared with the symmetric placement shown in Fig. 3(a), Fig. 3(b) shows a placement with a higher random distribution (constructed based on the methodology in [10]). Since the placement in Fig. 3(a) has better symmetry, it better reduces systematic mismatch [3], [11]. The placement in Fig. 3(b) exhibits a higher degree of dispersion, which better reduces random mismatch [10]. To reduce systematic and random mismatches simultaneously, a good arrangement should have a common-centroid structure with the highest possible degree of dispersion.

Many circuits, such as filters, modulators, and oscillators [12], employ noninteger capacitance ratios (e.g., the capacitance ratio of capacitor array 1 in Fig. 1 is 1:1.1:15.6:44.8, which is sourced from [3]). A noninteger-ratio capacitor array is usually divided into an integer portion and a noninteger portion, in which the layout of the integer portion is constructed using an integer number of unit capacitors and that of the noninteger portion is implemented using nonunit capacitors. The ratio of a nonunit capacitor relative to a unit capacitor is between 1 and 2. Based on the slotted rectangle technique or the stub technique [13], [14], two adjacent unit capacitors are required to implement a nonunit capacitor. One of the two adjacent unit capacitors is stretched to the other side and the other one is shrunk to form a dummy capacitor. Fig. 4 illustrates the two layout techniques. Fig. 4(a) shows the original placement of two adjacent unit capacitors. The difference between the two layout techniques is that the slotted rectangle technique uses a slot [see Fig. 4(b)] to keep the same overall perimeter-to-area ratio as that of the unit capacitor whereas the stub technique uses a stub [see Fig. 4(c)].

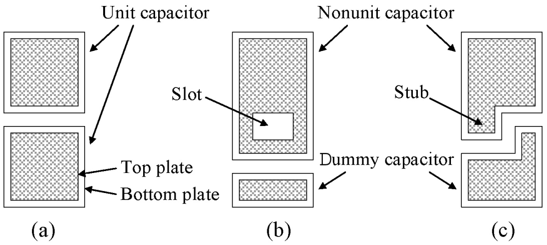


Fig. 4. Two layout techniques [13] for a nonunit capacitor. (a) Two originally adjacent unit capacitors. (b) Nonunit capacitor in the slotted rectangle technique. (c) Nonunit capacitor in the stub technique.

Two adjacent unit capacitors are required to implement a feasible layout for a nonunit capacitor, which is referred to as the adjacency constraint in this paper. In a 2-D capacitor placement, the area of two adjacent unit capacitors can be vertical or horizontal. The direction of the two adjacent unit capacitors depends on the routing channel direction. According to the routing rules in [13], routing wires are either horizontal or vertical to prevent intersection or overlap capacitance, and the routing of upper capacitor plates is separated from that of lower capacitor plates to avoid additional coupling and overlap capacitance. Fig. 5 shows a layout structure with vertical routing channels, where the upper-plate and lower-plate routing channels are interchanged. Therefore, the two adjacent unit capacitors for a nonunit capacitor should be placed either vertically or horizontally according to the routing channel direction, as shown in Fig. 6. Without loss of generality, the adjacency constraint refers to vertically adjacent unit capacitors. Unit capacitors at the edges of the array suffer less fringing than those internal to the array due to adjacent structures, and these unequal effects cause mismatch [13]. Thus, the array should be surrounded by dummy unit capacitors to act as adjacent structures. As shown in Fig. 5, the unit capacitors connected by the ground (GND) are dummy ones inserted for this purpose. In the following sections, the placements do not show these dummy unit capacitors. However, they should be inserted after capacitor placement to reduce the edge effects. Regular routing channels as those shown in Fig. 5 facilitate the interconnection of unit capacitors, but it does not mean that the matching of interconnection can be achieved more easily in this structure. Since this paper focuses only on the placement of unit capacitors, the interconnection issue is left as a further research problem.

For simplicity, a matrix is used to represent the placement of a capacitor array, where each entry represents one unit capacitor. The number of entries in the matrix should be equal to or larger than the number of unit capacitors in the capacitor

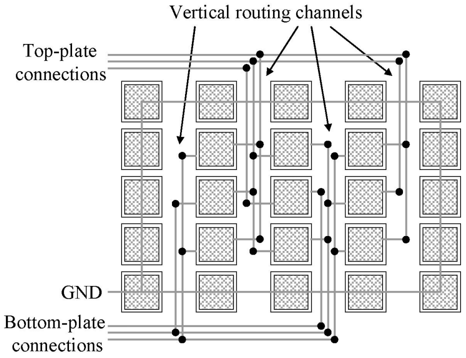


Fig. 5. Layout structure with vertical routing channels.

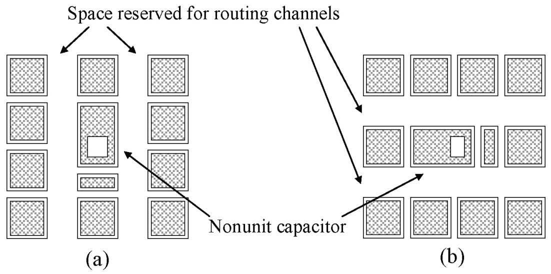
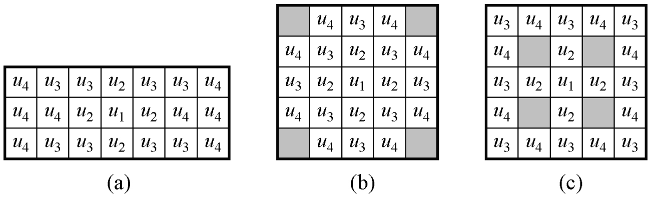


Fig. 6. Nonunit capacitors that occupy the area of two unit capacitors that are (a) vertically adjacent and (b) horizontally adjacent.



(a)

u_4	u_3	u_3	u_2	u_3	u_3	u_4
u_4	u_4	u_2	u_1	u_2	u_4	u_4
u_4	u_3	u_3	u_2	u_3	u_3	u_4

(b)

▨	u_4	u_3	u_4	▨
u_4	u_3	u_2	u_3	u_4
u_3	u_2	u_1	u_2	u_3
u_4	u_3	u_2	u_3	u_4
▨	u_4	u_3	u_4	▨

(c)

u_3	u_4	u_3	u_4	u_3
u_4	▨	u_2	▨	u_4
u_3	u_2	u_1	u_2	u_3
u_4	▨	u_2	▨	u_4
u_3	u_4	u_3	u_4	u_3

Fig. 7. (a) 3×7 matrix and an experimental placement result. (b) 5×5 matrix and an experimental placement result. (c) 5×5 matrix and the placement result after exchanging the locations of dummy unit capacitors and other unit capacitors (dummy unit capacitors are colored in gray).

array. For matrices whose entries outnumber unit capacitors, dummy unit capacitors should be inserted into empty entries. Although dummy unit capacitors do not induce mismatches, their location affects mismatch. For example, given a capacitor array that comprises four capacitors with a capacitance ratio of 1:4:8:8, Fig. 7(a) shows a placement for a 3×7 matrix. Since [7] suggests that an array should be nearly square for layout compactness, a square matrix can be used to obtain a more compact common-centroid placement. Fig. 7(b) shows a placement for the capacitor array in a 5×5 matrix, where four dummy unit capacitors are inserted. Fig. 7(c) shows another placement with the locations of the dummy unit capacitors changed. Since the placements in Fig. 7(b) and (c) are different, they induce different mismatches. To obtain a good placement, dummy unit capacitor placement must be considered.

A. Previous Work

To achieve better matching, the placement of a capacitor array is usually implemented with a common-centroid structure. Numerous works [11], [15]–[19] have studied common-centroid placement. Sayed and Dessouky [11] introduced an oxide gradient model to estimate oxide-gradient-induced mismatch. Based on this model, they presented a deterministic procedure for constructing a common-centroid placement. Some other works used topological representations, such as B*-trees [15], sequence-pairs [16], and center-based corner block list (C-CBL) [17], to tackle analog placement with common-centroid constraint. Strasser *et al.* [15] applied B*-trees to hierarchically bounded enumeration of basic building blocks and used enhanced shape functions for representing and combining the possible placements of the basic building blocks. Their placement algorithm is deterministic, and it can generate the Pareto front of placements with respect to different aspect ratios. Xiao and Young [16] explored the feasible conditions for the common-centroid and 1-D symmetry constraints based on the sequence-pair representation, and they showed a simulated-annealing-based placement tool that can handle the two constraints simultaneously. Ma *et al.* [17] proposed the C-CBL representation for denoting the placement of a common-centroid group. They presented a simulated-annealing-based placement method that can deal with symmetry, common-centroid, and general placement constraints at the same time. The works in [15]–[17] are general approaches that can handle common-centroid placement for subdevices with arbitrary shapes and dimensions.

For handling arrays of subdevices with regular shape and size, Ma *et al.* [18] proposed a grid-based approach that can distribute the subdevices uniformly to average out the influence of parasitic effects. In their approach, the assignment of subdevices was not constrained into a prespecified array. Given a group of subdevices, they generated a collection of feasible placement solutions with different column numbers and then used a pruning procedure to eliminate the redundant solutions dominated by others. The positions of subdevices in an array can be further exchanged through the simulated annealing (SA) process. Lin *et al.* [19] presented a thermal-driven common-centroid placement algorithm to assign the subdevices evenly into rows under the consideration of thermal profile. For a matching group, their approach generated all possible common-centroid placements with different aspect ratios (i.e., different numbers of rows), which is the same as the manner in [18].

For yield enhancement, Luo *et al.* [9], [20] introduced a spatial correlation model and discussed the relationship between correlation and variation of capacitance ratio. They showed that a capacitor placement with higher correlation coefficients achieves better matching. Furthermore, they proposed a heuristic algorithm [10], which partially exhaustively searches possible combinations and computes the correlation coefficients for each combination, to obtain a placement with the highest or near-highest correlation coefficients for yield improvement. Although their placement results exhibit a high degree of dispersion, their placements are not in a common-centroid structure. None of the above works, except for [11], considered noninteger-ratio capacitors in their method. Therefore, none of the existing works presented a method for constructing a common-centroid placement with the highest possible degree of dispersion for arbitrary-ratio capacitor arrays.

B. Our Contributions

It was suggested in [7] that a common-centroid layout should have the property of dispersion. Therefore, in this paper, an algorithm is proposed for constructing a placement that has a common-centroid structure and exhibits the highest possible degree of dispersion to reduce systematic and random mismatches simultaneously. The algorithm considers the adjacency constraint and the resulting placement is always feasible for the layout of noninteger-ratio capacitors.

Based on SA [21], a pair-sequence representation is used for the common-centroid placement. Since each pair of subdevices in a pair sequence can be automatically placed to symmetric locations in a layout, the corresponding placement can easily satisfy the common-centroid constraint. Three operations are proposed for perturbing a pair sequence, which can increase the degree of dispersion without breaking the common-centroid constraint in the resulting placement. Finally, a procedure for maintaining a feasible placement that fulfills the adjacency constraint after each perturbation is proposed. To make the proposed approach more efficient, the operations use a bucket data structure, redundant operations are removed, and multiple operations are combined to enhance the convergence of the SA process. The experimental results show that the placements obtained using the proposed method can achieve a smaller oxide-gradient-induced mismatch and larger overall correlation coefficients (i.e., higher degree of dispersion) than those of previous works.

The remainder of this paper is organized as follows. Section II formulates the common-centroid placement problem for arbitrary-ratio capacitor arrays. Section III describes the transformation between a pair sequence and its corresponding placement. Section IV shows how to initialize a pair sequence for an arbitrary-ratio capacitor set. Section V presents the operations for perturbing a pair sequence, the procedure for maintaining a feasible placement that fulfills the adjacency constraint, and the techniques for increasing efficiency. Section VI reports the experimental results. Finally, conclusions are drawn in Section VII.

II. Problem Formulation

In this paper, a common-centroid capacitor placement that fulfills several requirements for an arbitrary-ratio capacitor array is considered. First, the placement should have a common-centroid structure with the highest possible degree of dispersion to reduce systematic and random mismatches simultaneously. The resulting placement must satisfy the adjacency constraint if noninteger-ratio capacitors exist. Matrices whose entries outnumber the unit capacitors in a capacitor array must use dummy unit capacitors.

Let $C = \{C_1, C_2, \ldots, C_t\}$ denote a set of t capacitors, and the ratio of $C_1 : C_2 : \ldots : C_t$ be $q_1 : q_2 : \ldots : q_t$. Without loss of generality, assume that q_i is a rational number and that $q_i \leq q_j$ if $i < j$, where $1 \leq i,\ j \leq t$. Each capacitor C_i occupies b_i unit capacitors in the corresponding layout, where $b_i = \lceil q_i \rceil$. Let w denote the total number of unit capacitors in C (i.e., $w = b_1 + b_2 + \ldots + b_t$). Let $A_{r\times s}$ denote the $r \times s$ matrix used to place the unit capacitors of C; each entry in $A_{r\times s}$ can place one unit capacitor. Let n denote the size of the matrix, where $n = r \times s$ (i.e., $A_{r\times s}$ at most can place n unit capacitors).

Given an arbitrary-ratio capacitor array C and an arbitrary matrix $A_{r\times s}$, the capacitor placement problem is to assign each unit capacitor of C to a unique entry of $A_{r\times s}$. If n equals w, each entry in $A_{r\times s}$ contains exactly one unit capacitor. However, if $n > w$, entries which are not occupied by unit capacitors have to contain dummy unit capacitors. These dummy unit capacitors form an additional type of capacitor, named dummy capacitor C_D. Let b_D denote the number of dummy unit capacitors, where $b_D = n - w$.

Sayed and Dessouky proposed the oxide gradient model [11] for estimating the oxide-gradient-induced ratio mismatch of a capacitor array. Their model is used here to compute the process-gradient-induced mismatch. The spatial correlation model [9], [20] is applied to measure the degree of dispersion for a placement. Thus, the objective of the capacitor placement problem is to construct a placement that minimizes the oxide-gradient-induced mismatch and maximizes the overall correlation coefficients simultaneously. Note that the mismatches caused by dummy unit capacitors need not be evaluated. To achieve a feasible layout, the adjacency constraint must be fulfilled in the resulting placement if there exist noninteger-ratio capacitors.

In the following sections, the formulas for the oxide-gradient-induced mismatch [11] and the overall correlation coefficients [9], [20] are briefly reviewed. The two evaluation models are adopted in this paper for fair comparison with previous capacitor placement algorithms. The two models may not be accurate enough for modern technology or for different shapes of unit capacitors. To reflect the actual effects in the required technology, more accurate models can be substituted for the present ones in the proposed methodology.

A. Review of the Oxide Gradient Model

For a group of capacitors with arbitrary ratio, the ratio mismatch is calculated for each pair of capacitors and the largest value is retained. For t capacitors with a capacitance ratio of $C_1 : C_2 : \ldots : C_t$, which after the parallel unit-capacitor layout becomes $C_1^* : C_2^* : \ldots : C_t^*$, the capacitance ratio mismatch M is defined as

$$\begin{aligned} M &= \max\left(\left|\frac{\frac{C_i^*}{C_j^*} - \frac{C_i}{C_j}}{\frac{C_i}{C_j}}\right|\right) \times 100\% \\ &= \max\left(\left|\frac{C_i^*}{C_j^*}\frac{C_j}{C_i} - 1\right|\right) \times 100\% \qquad (1) \end{aligned}$$

for all i and j, where $1 \leq i,\ j \leq t$.

Let $\{u_{i1}, u_{i2}, \ldots, u_{ik}\}$ denote the set of k unit capacitors belonging to C_i. The capacitance of C_i after the parallel unit-capacitor layout is the summation of the capacitance of the unit capacitors

$$C_i^* = \sum_k u_{ik}. \qquad (2)$$

Assume that the unit-capacitor array is affected by a linear oxide gradient γ in the direction specified by an angle θ, as

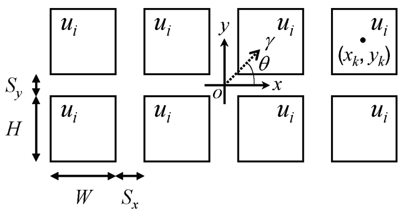


Fig. 8. Geometry of a unit-capacitor array for the estimation of oxide-gradient-induced mismatch.

shown in Fig. 8. Since the gradient angle cannot be predicted, the mismatch is estimated through 0° $\leq \theta \leq$ 180° and the maximum mismatch is retained. In Fig. 8, W and H are the width and height of a unit capacitor, and S_x and S_y are the horizontal and vertical spacing between unit capacitors. Let t_0 denote the oxide thickness at the origin O and the capacitance of a unit capacitor located at the origin be C_0. Due to the oxide gradient, a unit capacitor located at a different location experiences different oxide thickness and thus has different capacitance

$$u_{ik} = C_0 \frac{t_0}{t_k} f_k \tag{3}$$

where t_k is the equivalent oxide thickness at location (x_k, y_k)

$$t_k = t_0 + \gamma(x_k \cos\theta + y_k \sin\theta). \tag{4}$$

In (3), if the unit capacitor is treated as the decimal part of a nonunit capacitor, then f_k is the decimal value; otherwise, f_k = 1. Assume that the unit-capacitor array is achieved by an $r \times s$ matrix. If the unit capacitor u_{ik} is located at the entry in the r_kth row and s_kth column, the coordinates (x_k, y_k) of u_{ik} can be computed as

$$x_k = \left(s_k - \frac{s+1}{2}\right) \times (W + S_x) \tag{5}$$

$$y_k = \left(\frac{r+1}{2} - r_k\right) \times (H + S_y). \tag{6}$$

B. Review of the Spatial Correlation Model

Assume that the unit-capacitor array is achieved by an $r \times s$ matrix. For any two unit capacitors u_i and u_j, which are located at the entries in the r_ith row and s_ith column and the r_jth row and s_jth column, their correlation coefficient ρ_{ij} is defined as

$$\rho_{ij} = \rho_u^{D(i,j)} \tag{7}$$

$$D(i, j) = \sqrt{(r_i - r_j)^2 + (s_i - s_j)^2} \times l \tag{8}$$

where $0 < \rho_u < 1$, and l depends on process and size of devices. To simplify the experiment, assume that $l = 1$ to observe the relation between correlation and mismatch [10].

Let L denote the overall correlation coefficients of a capacitor array. For t capacitors, L is the summation of the correlation coefficients of all the capacitor pairs

$$L = \sum_{i=1}^{t-1} \sum_{j=i+1}^{t} R_{ij} \tag{9}$$

where R_{ij} is the correlation coefficient of two capacitors C_i and C_j. Assume that C_i consists of μ unit capacitors and C_j comprises ν unit capacitors. R_{ij} can be calculated as

$$R_{ij} = \frac{\sum_{a=1}^{\mu}\sum_{b=1}^{\nu} \rho_{ab}}{\sqrt{\left(\mu + 2\sum_{a=1}^{\mu-1}\sum_{b=a+1}^{\mu} \rho_{ab}\right)\left(\nu + 2\sum_{a=1}^{\nu-1}\sum_{b=a+1}^{\nu} \rho_{ab}\right)}}. \tag{10}$$

III. Pair-Sequence Representation

In this section, the pair-sequence representation for the placement of all elements in a matrix is introduced. Then, the method used for mapping each element in the representation to a unique entry of the matrix is described.

A. Matrix to Pair-Sequence Transformation

In this section, the pair-sequence $P_{r\times s}$ representation is proposed for denoting the placement of n elements in matrix $A_{r\times s}$. The pair sequence $P_{r\times s} = [p_1, p_2, \ldots, p_m]$ is an array of m elements, with each element p_i in $P_{r\times s}$ denoting a pair of symmetric entries in $A_{r\times s}$ except the first element p_1, where $m = \lceil n/2 \rceil$ and $n = r \times s$. For even and odd n, the first element p_1 denotes a pair of entries in the matrix and only one entry in the matrix, respectively. The subscript $r \times s$ of the representation indicates the dimensions of the matrix mapped by the representation. Since the placement of elements can have different dimensions, it is necessary to distinguish pair sequences with the same number of elements mapped to matrices with different dimensions. For example, although the sizes of matrices $A_{2\times6}$, $A_{6\times2}$, $A_{3\times4}$, and $A_{4\times3}$ are identical, their dimensions are different. Subscript $r \times s$ is thus appended to each pair sequence to denote the dimensions of the corresponding matrix.

The entries of a matrix $A_{r\times s}$ are encoded into the pair-sequence representation as follows. Let a_{ij} denote the entry in the ith row and jth column, where $1 \leq i \leq r$ and $1 \leq j \leq s$. The coordinate of a_{ij} is denoted by (x_j, y_i), where $x_j = j$ and $y_i = i$. The matrix $A_{r\times s}$ has a unique center whose coordinates are denoted by (x_c, y_c), where $x_c = (s+1)/2$ and $y_c = (r+1)/2$. Let d denote the distance from a_{ij} to the center of $A_{r\times s}$, which can be computed as

$$d = \sqrt{(x_j - x_c)^2 + (y_i - y_c)^2}. \tag{11}$$

Entries with the same distance d form a circle, which is denoted by R_d. Two entries are considered as a pair if they are in the same circle and opposite each other with respect to the center of $A_{r\times s}$, i.e., the two entries are symmetric to each other. After all entries of the matrix are classified into circles, the pairs of entries in a circle are selected serially in counterclockwise order from the 12-o'clock position. Then, a pair sequence is composed of these pairs from the inner circle to the outer circle. If the matrix comprises an odd number of entries, the innermost circle contains only one entry. In this case, the first pair in the corresponding pair sequence has only one element.

Fig. 9(a) shows a matrix $A_{3\times3}$. The entries can be divided into three circles, namely, R_0, R_1, and $R_{\sqrt{2}}$, according to

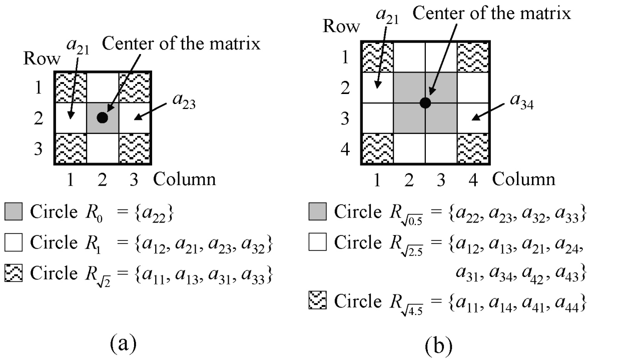

Row
a_{21}
Center of the matrix
a_{23}
1 2 3 Column
Circle R_0 = {a_{22}}
Circle R_1 = {$a_{12}, a_{21}, a_{23}, a_{32}$}
Circle $R_{\sqrt{2}}$ = {$a_{11}, a_{13}, a_{31}, a_{33}$}
(a)
Row
a_{21}
Center of the matrix
a_{34}
1 2 3 4 Column
Circle $R_{\sqrt{0.5}}$ = {$a_{22}, a_{23}, a_{32}, a_{33}$}
Circle $R_{\sqrt{2.5}}$ = {$a_{12}, a_{13}, a_{21}, a_{24}, a_{31}, a_{34}, a_{42}, a_{43}$}
Circle $R_{\sqrt{4.5}}$ = {$a_{11}, a_{14}, a_{41}, a_{44}$}
(b)

Fig. 9. (a) Circles of matrix $A_{3\times3}$, (a_{21}, a_{23}) is an example of a pair. (b) Circles of matrix $A_{4\times4}$, (a_{21}, a_{34}) is an example of a pair.

their distances to the center of $A_{3\times3}$. The entries in each circle are paired as follows. Since circle R_0 contains only one entry, a_{22} cannot be paired. For the entries in circle R_1, entry a_{12} is first selected because it is located at the 12-o'clock position. Since a_{12} and a_{32} are symmetric with respect to the center of the matrix, (a_{12}, a_{32}) is considered a pair. Following counterclockwise order, entry a_{21} and its opposite entry a_{23} are selected and considered as another pair. Similarly, the entries in circle $R_{\sqrt{2}}$ are paired; pairs (a_{11}, a_{33}) and (a_{31}, a_{13}) are obtained in sequence. After the entries in all circles are paired, a pair sequence is constructed by collecting the ordered pairs from the inner circle to the outer circle. Thus, the pair sequence $P_{3\times3}$ = [a_{22}, (a_{12}, a_{32}), (a_{21}, a_{23}), (a_{11}, a_{33}), (a_{31}, a_{13})] is derived. Fig. 9(b) shows another matrix $A_{4\times4}$. Similarly, the pair sequence $P_{4\times4}$ = [(a_{22}, a_{33}), (a_{32}, a_{23}), (a_{12}, a_{43}), (a_{21}, a_{34}), (a_{31}, a_{24}), (a_{42}, a_{13}), (a_{11}, a_{44}), (a_{41}, a_{14})] is derived.

B. Pair Sequence to Matrix Transformation

The procedure for deriving a pair sequence from a matrix was introduced in Section III-A. The method used for obtaining the corresponding placement once the materials are arranged into a pair sequence is described here.

Given a pair sequence $P_{r\times s}$, an $r \times s$ matrix $A_{r\times s}$ is first constructed according to the subscript of $P_{r\times s}$. Since each element in a pair corresponds to a unique location in the matrix, only materials located in a pair are placed in the corresponding entries in the matrix serially. For example, given a pair sequence $P_{3\times3}$ = [a, (b, c), (d, e), (f, g), (h, i)], a 3 × 3 matrix $A_{3\times3}$ is first constructed. Since each element in a pair is associated with a unique location in the matrix, material "a" in the first pair is placed in the corresponding entry a_{22} in $A_{3\times3}$, as shown in Fig. 10(a). Then, materials "b" and "c" in the second pair are placed in entries a_{12} and a_{32}, respectively, and so on. Once the pairs (d, e) and (h, i) in the original $P_{3\times3}$ are swapped, the pair sequence becomes $P_{3\times3}$ = [a, (b, c), (h, i), (f, g), (d, e)]. Thus, the corresponding placement is changed, as shown in Fig. 10(b).

IV. Placement Initialization

In this section, the procedure for constructing an initial placement for an arbitrary-ratio capacitor set is introduced. In the initial placement, all capacitors are arranged in a common-centroid structure and each noninteger-ratio capacitor satisfies the adjacency constraint. The first two sections, respectively, show the preferred method of arranging the dummy unit capacitors and the adjacent unit capacitors in a placement. Next, the procedure for pairing all the unit capacitors of a capacitor set is presented. Then, the method for arranging these unit-capacitor pairs to obtain a pair sequence is described. Finally, the whole algorithm of pair-sequence initialization is given in the last section.

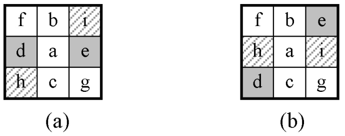


Fig. 10. (a) Placement result of $P_{3\times3}$ = [a, (b, c), (d, e), (f, g), (h, i)]. (b) Placement result of $P_{3\times3}$ = [a, (b, c), (h, i), (f, g), (d, e)], where the pairs (d, e) and (h, i) are exchanged in the original $P_{3\times3}$ (the two exchanged pairs are colored in different shades of gray).

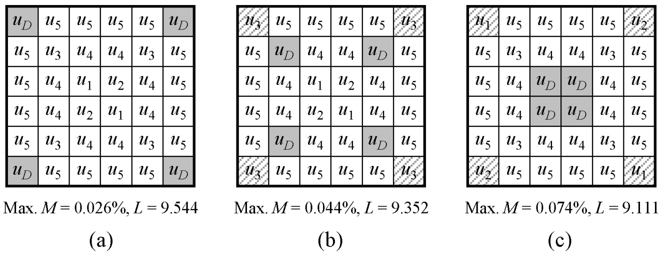
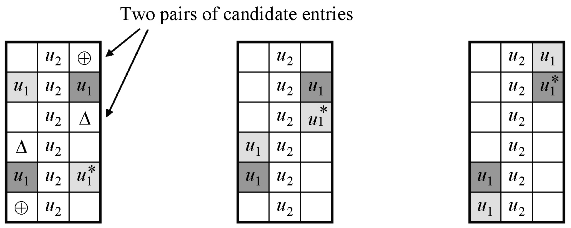

u_D u_5 u_5 u_5 u_5 u_D
u_5 u_3 u_4 u_4 u_3 u_5
u_5 u_4 u_1 u_2 u_4 u_5
u_5 u_4 u_2 u_1 u_4 u_5
u_5 u_3 u_4 u_4 u_3 u_5
u_D u_5 u_5 u_5 u_5 u_D
Max. M = 0.026%, L = 9.544
(a)
u_3 u_5 u_5 u_5 u_5 u_3
u_5 u_D u_4 u_4 u_D u_5
u_5 u_4 u_1 u_2 u_4 u_5
u_5 u_4 u_2 u_1 u_4 u_5
u_5 u_D u_4 u_4 u_D u_5
u_3 u_5 u_5 u_5 u_5 u_3
Max. M = 0.044%, L = 9.352
(b)
u_1 u_5 u_5 u_5 u_5 u_2
u_5 u_3 u_4 u_4 u_3 u_5
u_5 u_4 u_D u_D u_4 u_5
u_5 u_4 u_D u_D u_4 u_5
u_5 u_3 u_4 u_4 u_3 u_5
u_2 u_5 u_5 u_5 u_5 u_1
Max. M = 0.074%, L = 9.111
(c)

Fig. 11. (a) Experimental placement where u_Ds are placed at the corners. (b) Placement after exchanging u_Ds with u_3s in the original. (c) Placement after exchanging u_Ds with u_1s and u_2s in the original. M: oxide-gradient-induced mismatch. L: overall correlation coefficients.

A. Dummy Unit Capacitors

The locations of dummy unit capacitors affect the mismatch of a placement. Fig. 11 shows three placements for a capacitor set {C_1, C_2, C_3, C_4, C_5} with a capacitance ratio of 2:2:4:8:16. Assume that the matrix used to place the capacitor set is $A_{6\times6}$, which means that four dummy unit capacitors have to be inserted into the placement (i.e., b_D = 4). Let C_i = {u_i} denote the set of unit capacitors belonging to C_i, where u_i is one unit capacitor of C_i and the size of C_i is b_i. Similarly, C_D = {u_D} denotes the set of unit capacitors belonging to C_D and the size of C_D is b_D. Fig. 11(a) shows a common-centroid placement, in which all dummy unit capacitors are placed at the corners of the placement. Fig. 11(b) and (c) shows the placements after the dummy unit capacitors u_Ds have been exchanged with the unit capacitors u_3s and with u_1s and u_2s, respectively. Based on the oxide gradient model and the spatial correlation model, the three placements were evaluated. The following observations were made.

1) Capacitors placed closer to the center of the matrix lead to larger correlation coefficients [10]. Thus, it is preferred to place capacitors close to the center, if possible. Since some entries closer to the center are occupied by the dummy unit capacitors in Fig. 11(b), the overall correlation coefficients of the placement in Fig. 11(b) are smaller than those in Fig. 11(a).



Max. M = 0.109%, L = 0.977 (a)

Max. M = 0.084%, L = 0.969 (b)

Max. M = 0.128%, L = 0.960 (c)

Fig. 12. (a) Experimental placement with two pairs of candidate entries for the adjacency constraint. (b) Unit-capacitor pair moved from (a_{21}, a_{53}) to (a_{41}, a_{33}). (c) Unit-capacitor pair moved from (a_{21}, a_{53}) to (a_{61}, a_{13}). The unit capacitor marked by "*" is selected as the decimal of a nonunit capacitor. M: oxide-gradient-induced mismatch. L: overall correlation coefficients.

2) A small capacitor (e.g., C_1 or C_2) usually has a larger difference from its ideal capacitance value if it is placed away from the center of the matrix [11]. The mismatch of the placement in Fig. 11(c) is larger than that in Fig. 11(a) because the entries near the center are occupied by the dummy unit capacitors.

Based on the above observations, dummy unit capacitors should be placed close to the boundary of a matrix and small capacitors should be placed near the center of a matrix.

B. Adjacent Unit Capacitors

For the layout of an arbitrary-ratio capacitor set $C = \{C_1, C_2, \ldots, C_t\}$ with a capacitance ratio of $q_1 : q_2 : \ldots : q_t$, each capacitor C_i requires b_i unit capacitors, where $b_i = \lceil q_i \rceil$. The placement of each noninteger-ratio capacitor must fulfill the adjacency constraint (i.e., at least two unit capacitors are placed at adjacent locations in the resulting placement). In this section, the suitable placement for adjacent unit capacitors is determined.

Given a capacitor set $\{C_1, C_2\}$ with a capacitance ratio of 3.7:6 to be placed in a matrix $A_{6\times3}$, four and six unit capacitors, respectively, are required to implement C_1 and C_2. The placement of unit capacitors u_1s has to satisfy the adjacency constraint. Fig. 12(a) shows a common-centroid placement for the capacitor set. Since no two u_1s are placed at adjacent locations, the placement does not satisfy the adjacency constraint. To obtain a feasible placement, the unit-capacitor pair in entries (a_{21}, a_{53}) is selected and moved to the adjacent locations of the other pair in entries (a_{51}, a_{23}). There are two pairs of candidate entries, namely, (a_{41}, a_{33}) and (a_{61}, a_{13}), and Fig. 12(b) and (c) shows the resulting placements of the two choices, respectively. The three placements in Fig. 12 were evaluated using the oxide gradient model and the spatial correlation model. The following observation was made.

1) The mismatch of the placement in Fig. 12(c) is larger than that in Fig. 12(b) because C_1, which is a smaller capacitor relative to C_2, is placed farther from the center of the matrix.

The adjacency constraint is intrinsically easier to fulfill for a larger capacitor because it contains more unit capacitors. For smaller capacitors, candidate entries closer to the center of a matrix are preferred.

C. Unit-Capacitor Pairing

Unit-capacitor pairing is used to iteratively pair two unit capacitors until each unit capacitor of a capacitor set belongs to one pair. Then, the resulting pairs are arranged to form a pair sequence. According to the definition in Section III, the associated two unit capacitors in a pair are placed in symmetric locations with respect to the center of the matrix. Therefore, any two unit capacitors belonging to a given capacitor are preferred to be paired to obtain a symmetric placement. However, if a capacitor comprises an odd number of unit capacitors, one unit capacitor must be left out. Thus, for a capacitor with an odd number of unit capacitors, one unit capacitor is first selected such that the remaining unit capacitors can be paired.

Given an arbitrary-ratio capacitor set C and a matrix $A_{r\times s}$, dummy unit capacitors are generated if $n > w$. They form a dummy capacitor $C_D = \{u_D\}$ whose size is b_D. C_D is appended to the original set C to construct a new set $C' = \{C_1, C_2, \ldots, C_t, C_D\}$. It is clear that the entries of $A_{r\times s}$ can be filled by the unit capacitors of C'. If a capacitor in C contains only one unit capacitor, the capacitor is classified into subset C_{unit}. If a capacitor in C comprises an odd number of unit capacitors and the number is greater than two (i.e., 3, 5, …), the capacitor is classified into subset C_{odd}. For a noninteger-ratio capacitor C_i whose ratio is less than two (i.e., $1 < q_i < 2$), its layout consists of only two unit capacitors that must be placed at adjacent locations. In order to deal with the adjacency constraint for such a capacitor more easily, it is divided into two capacitors: C_i' and C_i'', each of which comprises only one unit capacitor. Thus, both C_i' and C_i'' are classified into the subset C_{unit}.

Let k and l denote the numbers of capacitors in C_{unit} and C_{odd}, respectively. Let z = 1 and 0 denote that b_D is odd and even, respectively. If a unit capacitor from each capacitor in C_{unit} and C_{odd} and also from capacitor C_D is selected when z = 1, the number of extracted unit capacitors is $k+l+z$. Thus, the number of remaining unit capacitors in each capacitor becomes even, and they can be paired. The selected unit capacitors (i.e., k+l+z unit capacitors) can be paired according to the following cases.

Case 1) *Both k and (l+z) are odd*: A unit capacitor is selected from k unit capacitors, and another one is selected from l unit capacitors if $l \neq 0$ (or from z unit capacitors if l = 0). The two selected unit capacitors are considered as a pair. Thus, the remaining unit capacitors can form ($k-1$)/2 and (l+$z$$-1$)/2 pairs.

Case 2) *k is odd and (l+z) is even*: A unit capacitor is selected from k unit capacitors and treated as a single-unit pair (i.e., a pair which contains only one unit capacitor). Thus, the remaining unit capacitors can form ($k-1$)/2 and (l+z)/2 pairs.

Case 3) *k is even and (l+z) is odd*: if $k \neq 0$, a unit capacitor is selected from k unit capacitors and considered as a single-unit pair, and thus the remaining unit capacitors can be handled by Case 1; if k = 0, a unit capacitor is selected from l unit capacitors if $l \neq 0$ (or from z unit capacitors if l = 0) and treated as a single-unit pair, and the remaining unit capacitors can form (l+z−1)/2 pairs.

Case 4) *Both k and (l+z) are even*: The k unit capacitors form k/2 pairs, and the (l+z) unit capacitors form (l+z)/2 pairs.

After the $k + l + z$ unit capacitors are paired, they can be classified into one of five types.

1) $S_{(\text{single},)}$: Single-unit pairs with each pair having only one unit capacitor.
2) $S_{(\text{unit, unit})}$: Two unit capacitors from two different capacitors in C_{unit}.
3) $S_{(\text{unit, odd})}$: One unit capacitor belongs to a capacitor in C_{unit}, and the other belongs to a capacitor in C_{odd}.
4) $S_{(\text{odd, odd})}$: Two unit capacitors from two different capacitors in C_{odd}.
5) $S_{(\text{odd, dummy})}$: One unit capacitor belongs to a capacitor in C_{unit} or C_{odd}, and the other belongs to C_D.

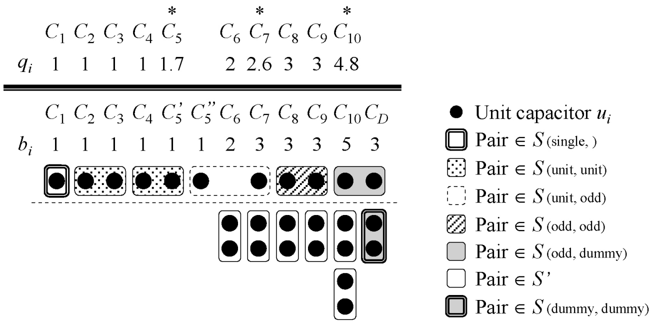
After the above procedure, the number of remaining unit capacitors in each capacitor becomes even, and thus the unit capacitors can be paired. These unit-capacitor pairs can be classified into one of two types.

1) S': Two unit capacitors in each pair come from the same capacitor except C_D.
2) $S_{(\text{dummy, dummy})}$: Two unit capacitors belong to C_D.

Given a capacitor set {C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}} with a capacitance ratio of 1:1:1:1:1.7:2:2.6:3:3:4.8 and a matrix $A_{5\times5}$, the numbers of unit capacitors required by C_i are 1, 1, 1, 1, 2, 2, 3, 3, 3, and 5, respectively. The number of dummy unit capacitors is 3. According to the aforementioned definition, C_{unit} = {C_1, C_2, C_3, C_4, C'_5, C''_5} and C_{odd} = {C_7, C_8, C_9, C_{10}}. Then, a unit capacitor is selected from each capacitor in C_{unit} and C_{odd}, and also from C_D, respectively (i.e., u_1, u_2, u_3, u_4, u_5, u_5, u_7, u_8, u_9, u_{10}, and u_D are extracted). Since k = 6, l = 4, and z = 1, the selected unit capacitors are paired according to Case 3. Assume that u_1 is selected from the first six unit capacitors (i.e., u_1, u_2, u_3, u_4, u_5, and u_5) and is considered as a single-unit pair [i.e., (u_1,) ∈ $S_{(\text{single},)}$]. Next, the remaining five unit capacitors (i.e., u_2, u_3, u_4, u_5, and u_5) and the last five unit capacitors (i.e., u_7, u_8, u_9, u_{10}, and u_D) are paired based on Case 1. Thus, a unit capacitor is selected from each of the two groups (assume that u_5 and u_7 are chosen). The two unit capacitors are considered as a pair [i.e., (u_5, u_7) ∈ $S_{(\text{unit, odd})}$]. Finally, the remaining unit capacitors u_2, u_3, u_4, u_5, u_8, u_9, u_{10}, and u_D are paired, which form (u_2, u_3) and (u_4, u_5) ∈ $S_{(\text{unit, unit})}$, (u_8, u_9) ∈ $S_{(\text{odd, odd})}$, and (u_{10}, u_D) ∈ $S_{(\text{odd, dummy})}$. After the above process, the numbers of remaining unit capacitors in capacitors C_i and C_D becomes even, and these unit capacitors can be paired and classified into the types S' and $S_{(\text{dummy, dummy})}$, respectively. Fig. 13 shows the results of unit-capacitor pairing obtained using the above procedure.

D. Pair Arrangement

After all unit capacitors of a capacitor set are paired, the pairs can be arranged to obtain a pair sequence, and then an initial placement from the pair sequence can be derived. To obtain a suitable placement, it is necessary to determine an order for assigning these pairs into the pair sequence.



*
*
*
C_1 C_2 C_3 C_4 C_5 C_6 C_7 C_8 C_9 C_{10}
q_i 1 1 1 1 1.7 2 2.6 3 3 4.8
C_1 C_2 C_3 C_4 C'_5 C''_5 C_6 C_7 C_8 C_9 C_{10} C_D
b_i 1 1 1 1 1 1 2 3 3 3 5 3
Unit capacitor u_i
Pair ∈ $S_{(\text{single},\ )}$
Pair ∈ $S_{(\text{unit, unit})}$
Pair ∈ $S_{(\text{unit, odd})}$
Pair ∈ $S_{(\text{odd, odd})}$
Pair ∈ $S_{(\text{odd, dummy})}$
Pair ∈ S'
Pair ∈ $S_{(\text{dummy, dummy})}$

Fig. 13. Example of unit-capacitor pairing. *: noninteger-ratio capacitor.

Before the procedure is described, the properties of various kinds of capacitor are first analyzed. If a capacitor belongs to C_{unit}, it is best to place it near the center of a matrix such that its centroid can be close to the matrix's center. If a capacitor belongs to C_{odd}, one of its unit capacitors will be a single-unit capacitor and the others will form several unit-capacitor pairs. Since each unit-capacitor pair can be automatically placed in symmetric locations in the matrix, the capacitor's centroid can be located close to the center of the matrix if the single-unit capacitor is also placed near the matrix's center. Finally, if a capacitor comprises an even number of unit capacitors, its unit capacitors can be paired completely, and thus its centroid is exactly at the center of the matrix.

Based on the above analysis, some unit capacitors should have higher priority than others to be placed closer to the matrix's center for a common-centroid placement. According to the observations in Section IV-A, dummy unit capacitors should be placed close to the boundary of a matrix and small capacitors should be placed near the center of a matrix. If a pair can be arranged near to the head of a pair sequence, the placement of the associated unit capacitors would be close to the matrix's center; otherwise, their placement is far from the matrix's center. Therefore, the following priorities are assigned to the pair types:

$$S_{(\text{single},)} > S_{(\text{unit, unit})} > S_{(\text{unit, odd})} > S_{(\text{odd, odd})} > S_{(\text{odd, dummy})} > S' > S_{(\text{dummy, dummy})}$$

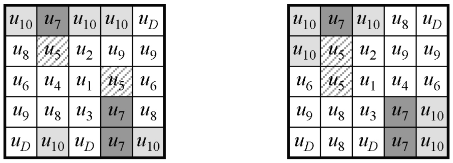
where $S_{(\text{single},)} > S_{(\text{unit, unit})}$ means that $S_{(\text{single},)}$ has a higher priority than $S_{(\text{unit, unit})}$, and so on. A pair whose type has higher priority will be arranged closer to the head of a pair sequence.

E. Algorithm for Pair-Sequence Initialization

This section gives a complete procedure for initializing a pair sequence (the pseudocode is shown in Algorithm 1). Given a capacitor array and a matrix for placement, the number of required unit capacitors for each capacitor and the number of dummy unit capacitors are first calculated (see Line 1). Next, unit capacitors are paired and classified into different types (see Line 2). Then, the pairs are placed into a pair sequence serially according to their priorities. Since the pairs belonging to the same type have identical priority, they are arranged into a pair sequence in deterministic order (see Lines 4–8). Finally, the adjacency constraint for noninteger-ratio capacitors is checked. If a placement violates the adjacency

Algorithm 1 InitializePairSequence(Capacitor array, Matrix size)

```
 1: Calculate the number of required unit capacitors for
    each capacitor;
    /* All unit capacitors are paired and classified into seven
    pair types; */
 2: Unit-capacitor pairing for the capacitor array;
 3: Allocate an empty pair sequence $P$;
 4: for highest-priority pair type to lowest-priority pair type do
 5:   for all unit-capacitor pairs belonging to this pair type do
 6:       Add the pair that has a unit capacitor $u_i$ with the
          smallest index $i$ to the end of the pair sequence $P$;
 7:   end for
 8: end for
 9: $P_{r \times s}$ ← append the matrix size $r \times s$ to the pair sequence $P$;
10: Check the adjacency constraint and maintain a feasible
       placement;
11: return $P_{r \times s}$,
```



u_{10}	u_7	u_{10}	u_{10}	u_D
u_8	u_5	u_2	u_9	u_9
u_6	u_4	u_1	u_5	u_6
u_9	u_8	u_3	u_7	u_8
u_D	u_{10}	u_D	u_7	u_{10}

(a)

u_{10}	u_7	u_{10}	u_8	u_D
u_{10}	u_5	u_2	u_9	u_9
u_6	u_5	u_1	u_4	u_6
u_9	u_8	u_3	u_7	u_{10}
u_D	u_8	u_D	u_7	u_{10}

(b)

Fig. 14. (a) Initial placement for the capacitor array shown in Fig. 13, which violates the adjacency constraint. (b) Feasible placement for the noninteger-ratio capacitors C_5, C_7, and C_{10}. Noninteger-ratio capacitors are colored in different shades of gray.

constraint, it is transformed into a feasible one (see Line 10). The details of maintaining a feasible placement are described in Section V-B.

Assume that the capacitor array shown in Fig. 13 is to be placed into a matrix $A_{5\times5}$. A pair sequence $P_{5\times5}$ = [u_1, (u_2, u_3), (u_4, u_5), (u_5, u_7), (u_8, u_9), (u_{10}, u_D), (u_6, u_6), (u_7, u_7), (u_8, u_8), (u_9, u_9), (u_{10}, u_{10}), (u_{10}, u_{10}), (u_D, u_D)] is first generated by Lines 1–9 of Algorithm 1. From the pair sequence, the corresponding placement can be derived, as shown in Fig. 14(a). Since capacitors C_5, C_7, and C_{10} are noninteger-ratio capacitors, their placements have to satisfy the adjacency constraint. However, the placements of C_5 and C_{10} shown in Fig. 14(a) violate the adjacency constraint. This condition is checked and modified by Line 10 of Algorithm 1 to achieve a feasible placement, as shown in Fig. 14(b), for noninteger-ratio capacitors.

V. Placement Algorithm

After an initial placement has been generated, the SA algorithm [21] is used to obtain better results. The SA algorithm repeatedly perturbs the pair sequence until a predefined termination condition is fulfilled. In the following sections, three operations for perturbing a pair sequence without breaking the symmetry property are introduced. To guarantee that the adjacency constraint is fulfilled after perturbation, a procedure for maintaining a feasible placement is presented. The third section gives the cost function of the proposed SA-based placement algorithm. Several techniques for increasing efficiency are described in the fourth section. Finally, the last section shows the global algorithm for capacitor placement.

Algorithm 2 MaintainFeasiblePlacement($P_{r\times s}$, Cap-for-check $\{C_t\}$)

```
 1: Derive a placement $T$ from $P_{r\times s}$;
 2: check_list ← $\{C_t\}$;
 3: while (check_list $\neq \emptyset$)
 4:   $C_i$ ← extract_one_cap(check_list);
 5:   if $C_i$ violates the adjacency constraint in $T$ then
 6:     for all feasible candidate entries for $C_i$ do
 7:       Calculate a priority value for this candidate entry;
 8:     end for
 9:     Select the candidate entry with the highest priority
        for exchange;
10:     if the selected candidate entry is originally
        occupied by $C_j$ that is another noninteger ratio
        capacitor then
11:       check_list ← check_list $\cup\{C_j\}$;
12:     end if
13:     Update $P_{r\times s}$ and $T$;
14:   end if
15: end while
```

A. Perturbation

The symmetry property of a common-centroid placement should be maintained during perturbation. To avoid breaking this property, the following operations are proposed.

1) *Op1*: It chooses one pair p_i that belongs to $S_{(\text{unit,odd})}$, $S_{(\text{odd,odd})}$, or $S_{(\text{odd,dummy})}$, and then reverses the order of the two unit capacitors in p_i.
2) *Op2*: It chooses two pairs p_i and p_j from any pair type except S', and then exchanges one unit capacitor u_k in p_i with another unit capacitor u_l in p_j.
3) *Op3*: It chooses two pairs p_i and p_j from any pair type except $S_{(\text{single},)}$, and then exchanges the order of p_i and p_j in the pair sequence.

Since Op2 exchanges the contents of two selected pairs, the types of the two pairs have to be updated after this operation is applied.

B. Maintaining a Feasible Placement

A feasible placement, which meets the adjacency constraint, may become infeasible after perturbation. To guarantee that the resulting placement is always feasible for noninteger-ratio capacitors, a procedure for maintaining a feasible placement after each perturbation is proposed (pseudocode is shown in Algorithm 2).

Given a pair sequence, the algorithm first derives a placement from the pair sequence (see Line 1), and then assigns noninteger-ratio capacitors $\{C_t\}$ into a checklist for a feasibility check (see Line 2). In the beginning, $\{C_t\}$ contains all noninteger-ratio capacitors of C; thus, the feasibility of all noninteger-ratio capacitors has to be checked. However, to avoid unnecessary feasibility checks during the SA process, only the noninteger-ratio capacitors involved by a perturbation are included into $\{C_t\}$. Then, the capacitors in the checklist are checked in sequence (see Lines 3 and 4). If any capacitor C_i violates the adjacency constraint, all candidate entries that can be exchanged with the invalid unit capacitors of C_i are traced, and each of these candidate entries is evaluated to determine its priority (see Lines 5–8). According to the observations in Section IV-B, the candidate entries that are closest to the center of a matrix are preferred under the adjacency constraint. Therefore, the priority of a candidate entry is determined by the distance from the entry to the matrix's center, with a shorter distance leading to a higher priority. However, if the candidate entry is occupied by another capacitor that is also a noninteger-ratio capacitor or has a smaller capacitance ratio than that of C_i, its priority is modified to a lower level. After the priority of all candidate entries has been evaluated, the candidate entry with the highest priority is selected for exchange with the invalid unit capacitor of C_i (see Line 9). Only Op1 and Op3 are used in the exchange process; thus, the types of associated pairs are not changed. If the selected candidate entry is originally occupied by another noninteger-ratio capacitor C_j, C_j will be appended to the checklist because its placement may violate the adjacency constraint after the exchange (see Lines 10–12). Finally, the pair sequence and the corresponding placement are updated (see Line 13).

For example, given the placement shown in Fig. 14(a) and a noninteger-ratio capacitor set $\{C_5, C_7, C_{10}\}$, which requires a feasibility check, Algorithm 2 first checks C_5 and finds that C_5 violates the adjacency constraint. Thus, the entries a_{12}, a_{32}, a_{24}, and a_{44} are traced as candidate entries. Depending on their distances to the matrix's center, the initial priority is $a_{32} > a_{24} = a_{44} > a_{12}$. To move u_5 from the original entry a_{34} to the first candidate entry a_{32} for fulfilling the adjacency constraint, Op1 is performed on the pair (a_{32}, a_{34}). However, if u_5 is to be moved from the original entry a_{22} to the second candidate entry a_{24}, Op1 is first executed to reverse the sequence in the original pair (a_{22}, a_{44}), and then Op3 is performed to swap the pairs (a_{22}, a_{44}) and (a_{42}, a_{24}). To move u_5 from the original entry a_{22} to the third candidate entry a_{44}, Op1 is executed on the pair (a_{22}, a_{44}). Since both the second and third cases involve the entry a_{44}, which is originally occupied by another noninteger-ratio capacitor C_7, the priority of the two candidate entries is modified to a lower level. Similarly, the fourth candidate entry a_{12} is also occupied by the noninteger-ratio capacitor C_7, and thus a lower priority is given. According to the final priority, the candidate entry a_{32} has the highest priority and is selected for exchange. Next, the feasibility of C_7 is checked; its placement satisfies the adjacency constraint. Finally, Algorithm 2 checks C_{10} and finds that C_{10} violates the adjacency constraint. The feasible candidate entries for C_{10} include a_{23}, (a_{21}, a_{45}), and (a_{42}, a_{24}). After a similar evaluation process is conducted on the candidate entries, the candidate entries (a_{21}, a_{45}) have the highest priority and are selected for exchange. Since the entry pair (a_{21}, a_{45}) can be exchanged with the pair (a_{13}, a_{53}) or (a_{52}, a_{14}), the latter is randomly chosen and then Op3 is executed on the pairs (a_{21}, a_{45}) and (a_{52}, a_{14}). Thus, the unit-capacitor pair (u_{10}, u_{10}) is moved from the original entries (a_{52}, a_{14}) to (a_{21}, a_{45}), and a feasible placement for C_{10} is obtained. The above procedure maintains the feasibility of all noninteger-ratio capacitors. The resulting placement is shown in Fig. 14(b).

C. Cost Function

The proposed method has two major objectives, namely, to minimize oxide-gradient-induced mismatch and to maximize the overall correlation coefficients. The oxide-gradient-induced mismatch and overall correlation coefficients of a placement can be evaluated using the oxide gradient model and the spatial correlation model, respectively. For a placement T, let M_T denote its oxide-gradient-induced mismatch and L_T denote its overall correlation coefficients. During the SA process, the average values of the oxide-gradient-induced mismatch and the overall correlation coefficients, which are respectively denoted by M_{avg} and L_{avg}, are kept. The cost function $\Phi(T)$ is computed as

$$\Phi(T) = \alpha \times \frac{M_{\text{avg}} - M_T}{M_{\text{avg}}} + (1 - \alpha) \times \frac{L_T - L_{\text{avg}}}{L_{\text{avg}}} \qquad (12)$$

where α is a user-specified parameter, $0 \le \alpha \le 1$. The goal of the algorithm is to find a placement with the maximum Φ.

D. Efficiency Improvements

Since the SA is a time-consuming process, the following techniques are proposed to decrease its runtime.

1) *Bucket data structure*: To facilitate the operations during perturbation, a bucket list data structure is used to record necessary information. It includes two bucket arrays H_B and H_P, which denote different pair types and a pair sequence, respectively. Each pair associated with a pair type is connected to the entry of the corresponding type in H_B by a doubly linked list. There is also another point referring to the pair from the pair sequence H_P. Fig. 15 shows the data structure used in the program. With this data structure, the program can easily extract a pair with the specified type during perturbation. If the type of pair changes after perturbation, only the information in the associated doubly linked lists has to be updated.
2) *Redundancy elimination*: The program ignores redundant operations during perturbation. For example, if two capacitors belonging to C_{unit} are chosen for position exchange, the resulting placements before and after the exchange are geometrically identical because the two capacitors are intrinsically the same if their index numbers are disregarded. Such a perturbation is considered as a redundant operation.
3) *Operation combination*: The convergence of the SA process can be enhanced by applying an operation that can lead to a large change. Thus, Op1 and Op3 are combined to get a new perturbation. The operation is to choose two pairs p_i and p_j from any pair type except $S_{(\text{single},)}$, and perform Op3 on p_i and p_j followed by one of the following actions.
   a) Execute Op1 on p_i.
   b) Execute Op1 on p_j.
   c) Execute Op1 on both p_i and p_j.

Note that these operations might deteriorate the solution quality when the SA process is converging. Therefore, a much low probability of applying these operations is given at low temperature.

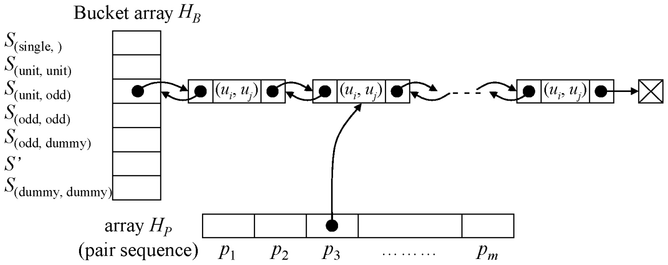


Fig. 15. Bucket data structure used in the proposed program.

Algorithm 3 CapacitorPlacement(Capacitor array, Matrix size)

```
1: $P_{r\times s}$ ← initializePairSequence(Capacitor array, Matrix size);
2: Derive a placement $T$ from $P_{r\times s}$;
3: Set an initial temperature $E$ and a temperature length $G$;
4: while ($E$ > freezing point)
5:   for $k = 1$ to $G$ do
6:     $P^*{}_{r\times s}$ ← perturb($P_{r\times s}$);
7:     maintainFeasiblePlacement($P^*{}_{r\times s}$, Cap-for-check $\{C_t\}$);
8:     Derive a placement $T^*$ from $P^*_{r\times s}$;
9:     if $\Phi(T^*) \geq \Phi(T)$ then
10:      $P_{r\times s}$ ← $P^*_{r\times s}$;
11:      $T$ ← $T^*$;
12:    else
13:      Calculate an acceptance rate $A$ according to
           $\Phi(T)$, $\Phi(T^*)$, and $E$;
14:      Generate a random number $R$;
15:      if $A > R$ then
16:        $P_{r\times s}$ ← $P^*{}_{r\times s}$;
17:        $T$ ← $T^*$;
18:      end if
19:    end if
20:  end for
21:  $E$ ← $E$ × cooling ratio;
22: end while
```

E. Global Algorithm

The global algorithm for capacitor placement is shown in Algorithm 3. After an initial placement is generated (see Lines 1 and 2), the SA algorithm repeatedly perturbs the pair sequence until the termination condition is fulfilled (see Lines 3–22). The adjacency constraint for noninteger-ratio capacitors must be maintained after each perturbation (see Lines 6 and 7). The objective is to find a placement with the maximum Φ (see Lines 9–11), which achieves the minimum oxide-gradient-induced mismatch and maximum overall correlation coefficients simultaneously.

VI. Experimental Results

The proposed capacitor placement algorithm was implemented in the C++ programming language and run on a 2.5-GHz Intel Core2 Quad PC. Three sets of experiments were performed: 1) arbitrary-ratio capacitor arrays; 2) integer-ratio capacitor arrays; and 3) capacitor arrays with dummy capacitors. To keep the comparisons fair, the experimental setups were the same as those in [10] and [11]: the oxide thickness t_0 was 40 nm, the oxide gradient γ was 10 ppm, the correlation coefficient between unit capacitors ρ_u was 0.9, and the geometry of unit capacitors was that depicted in Fig. 16(a).

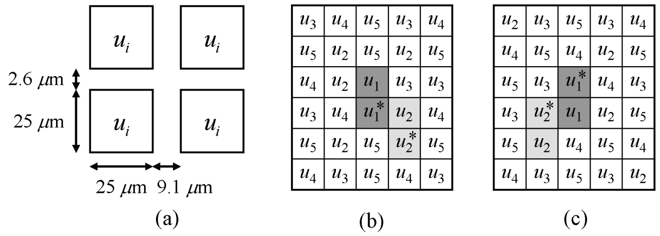

u_i u_i
u_i u_i
2.6 μm
25 μm
25 μm 9.1 μm
(a)
u_3 u_4 u_5 u_3 u_4
u_5 u_2 u_5 u_2 u_5
u_4 u_2 u_1 u_3 u_3
u_3 u_4 u_1^* u_2 u_4
u_5 u_2 u_5 u_2^* u_5
u_4 u_3 u_5 u_4 u_3
(b)
u_2 u_3 u_5 u_3 u_4
u_4 u_5 u_4 u_2 u_5
u_5 u_3 u_1^* u_4 u_3
u_3 u_2^* u_1 u_2 u_5
u_5 u_2 u_4 u_5 u_4
u_4 u_3 u_5 u_3 u_2
(c)

Fig. 16. (a) Experiment setup for the geometry of unit capacitors. (b) Placement constructed from [11] for SCF_1. (c) Placement for SCF_1 obtained using proposed method (adjacent unit capacitors for the layout of nonunit capacitors are colored in different shades of gray, and the unit capacitor marked by "*" is selected as the decimal of a nonunit capacitor).

A. Arbitrary-Ratio Capacitor Arrays

The first set of experiments is based on three arbitrary-ratio capacitor arrays, SCF_1, SCF_2, and SCF_3, which are sourced from [3] and [11]. These capacitor arrays are designed for SC filters, and they contain noninteger-ratio capacitors. Since no previous studies have considered systematic and random mismatches at the same time for the placement of arbitrary-ratio capacitor arrays, the proposed method is compared with the systematic algorithm [11], which can handle arbitrary-ratio capacitor arrays but only considers systematic mismatch.

The experimental results are shown in Table I. For each approach, the value of the oxide-gradient-induced mismatch (denoted by M), the value of the overall correlation coefficients (denoted by L), and the runtime are listed. The experimental results show that the placements obtained using the proposed method have smaller oxide-gradient-induced mismatches than those obtained by [11] for SCF_1 and SCF_3. The overall correlation coefficients of the placements obtained using the proposed method are larger than those obtained by [11] in all test cases. Fig. 16(b) and (c) shows the resulting placements of SCF_1 based on [11] and the proposed approach, respectively. Although the two placements have a common-centroid structure and both satisfy the adjacency constraint, the placement obtained using the proposed method achieves a smaller oxide-gradient-induced mismatch and larger overall correlation coefficients (i.e., higher degree of dispersion).

B. Integer-Ratio Capacitor Arrays

For the placement of integer-ratio capacitor arrays, the proposed approach is compared with the heuristic algorithm proposed by Chen *et al.* [10] based on six integer-ratio capacitor arrays. The first three capacitor arrays, named SCF_1_I, SCF_2_I, and SCF_3_I, are derived from SCF_1, SCF_2, and SCF_3 by modifying the noninteger-ratio capacitors to integer-ratio capacitors. The others are sourced from the capacitor arrays used in self-designed SAR ADCs. Since the resolutions of the ADCs are 8 bits, 9 bits, and 10 bits, the capacitor arrays are named SAR_8bit, SAR_9bit, and SAR_10bit, respectively.

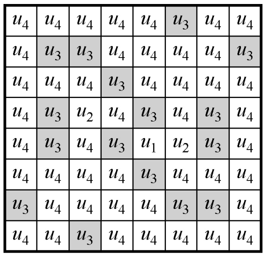
Since [10] only showed the placement result of SCF_3_I, their algorithm was implemented here to obtain the placement

TABLE I

Comparison of the Systematic Algorithm [11] and Present Work for Arbitrary-Ratio Capacitor Arrays

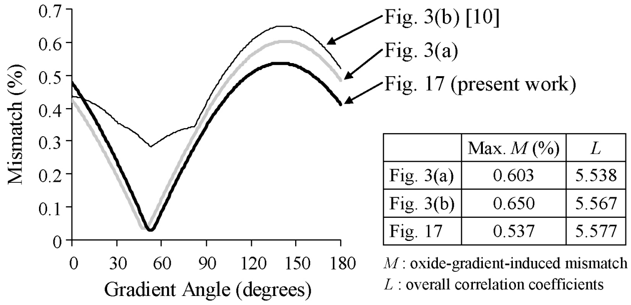
Array Name	No. of Cap.	Capacitance Ratio	No. of Unit Cap.	Matrix Size	Systematic Algorithm [11]			Present Work		
					Max. M (%)	L	Time (s)	Max. M (%)	L	Time (s)
SCF_1	5	1.2:5.8:7:7:8	30	6 × 5	0.324	9.625	N/A	0.290	9.644	71
SCF_2	5	1:1.4:2:9.2: 17	32	8 × 4	0.878	9.267	N/A	0.878	9.293	73
SCF_3	4	1:1.1:15.6:44.8	64	8 × 8	1.064	5.491	N/A	0.858	5.510	535

N/A: the runtime was not reported in [11].



u_4	u_4	u_4	u_4	u_4	u_3	u_4	u_4
u_4	u_3	u_3	u_4	u_4	u_4	u_4	u_3
u_4	u_4	u_4	u_3	u_4	u_4	u_4	u_4
u_4	u_3	u_2	u_4	u_3	u_4	u_3	u_4
u_4	u_3	u_4	u_3	u_1	u_2	u_3	u_4
u_4	u_4	u_4	u_4	u_3	u_4	u_4	u_4
u_3	u_4	u_4	u_4	u_4	u_3	u_3	u_4
u_4	u_4	u_3	u_4	u_4	u_4	u_4	u_4

Fig. 17. Placement result of SCF_3_I obtained using proposed method (the unit capacitors denoted by u_3 are colored in gray for comparison with Fig. 3).



	Max. M (%)	L
Fig. 3(a)	0.603	5.538
Fig. 3(b)	0.650	5.567
Fig. 17	0.537	5.577

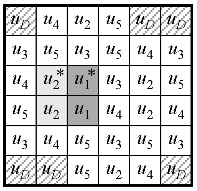
M : oxide-gradient-induced mismatch

L : overall correlation coefficients

Fig. 18. Oxide-gradient-induced mismatches and overall correlation coefficients of three different placements for SCF_3_I.

results of the other test cases, which are listed in Table II. The experimental results show that the placements obtained using the proposed method achieve smaller oxide-gradient-induced mismatches and larger overall correlation coefficients simultaneously than those obtained by [10] in all test cases. Although the proposed approach is slower than [10] for smaller capacitor arrays, it runs much faster than [10] for larger arrays. Since [10] partially exhaustively searches possible combinations and computes the correlation coefficients for each combination to obtain the best one, its computational time increases significantly with the number of available entries. Therefore, [10] cannot obtain results for the largest capacitor array (SAR_10bit) within an acceptable time. In contrast, the proposed approach applies SA to enhance results and uses several techniques to increase efficiency. Its results and runtime are thus better than those of [10].

Fig. 17 shows the placement result of SCF_3_I obtained using the proposed method. Compared with the placement shown in Fig. 3(b), which is sourced from [10], the placement obtained using the proposed method has comparable overall correlation coefficients (i.e., high degree of dispersion). The symmetry property in the placement is also better (i.e., common-centroid structure). Based on the oxide gradient model, the mismatch variation with gradient angle for three different placements of SCF_3_I was calculated; the results are shown in Fig. 18. The placement based on [10] has the largest oxide-gradient-induced mismatch because it does not have a common-centroid structure. Although the placement in Fig. 3(a) and the placement obtained using the proposed method have a common-centroid structure, the latter has a smaller oxide-gradient-induced mismatch and larger overall correlation coefficients because the proposed approach enhances the common-centroid placement.



u_D	u_4	u_2	u_5	u_D	u_D
u_3	u_5	u_3	u_5	u_4	u_3
u_4	u_2^*	u_1^*	u_3	u_2	u_5
u_5	u_2	u_1	u_4	u_2	u_4
u_3	u_4	u_5	u_3	u_5	u_3
u_D	u_D	u_5	u_2	u_4	u_D

Fig. 19. Placement result of SCF_1 with dummy unit capacitors in square matrix $A_{6\times6}$ (dummy unit capacitors and adjacent unit capacitors for nonunit capacitors are colored in different shades of gray; the unit capacitor marked by "*" is selected as the decimal of a nonunit capacitor).

C. Capacitor Arrays With Dummy Capacitors

In this set of experiments, capacitor arrays with non-square placements were placed in a square using dummy capacitors. Since a square matrix provides a more compact placement, it sometimes achieves better matching than a rectangular matrix. However, the size of a square matrix might be larger than the number of unit capacitors in a capacitor array. Thus, the arrangement of dummy unit capacitors in the resulting placement must be considered.

Table III compares the placements of the capacitor arrays with and without dummy capacitors. SCF_1, SCF_2, and SAR_9bit were used because other capacitor arrays are in square placements. The column "Without Dummy Capacitors" represents the condition that the matrix size is equal to the number of unit capacitors (i.e., no dummy unit capacitors are needed). The data in this column was taken from Tables I and II directly. The column "With Dummy Capacitors" represents the condition that the capacitor array is placed into a square matrix and some dummy unit capacitors are inserted. In the placement of SAR_9bit, only the issue of dummy unit capacitors must be considered. However, both the adjacency constraint and dummy unit capacitors must be considered in the placements of SCF_1 and SCF_2 because they contain noninteger-ratio capacitors.

According to the placement results of SCF_1, the matching of the placement in a square matrix $A_{6\times6}$ is not better than that

TABLE II

COMPARISON OF THE HEURISTIC ALGORITHM [10] AND PRESENT WORK FOR INTEGER-RATIO CAPACITOR ARRAYS

Array Name	No. of Cap.	Capacitance Ratio	No. of Unit Cap.	Matrix Size	Heuristic Algorithm [11]			Present Work		
					Max. M (%)	L	Time (s)	Max. M (%)	L	Time (s)
SCF_1_I	5	2:6:7:7:8	30	6 × 5	0.138	9.651	1	0.085	9.693	7
SCF_2_I	5	1:2:2:10:17	32	8 × 4	0.679	9.318	1	0.559	9.354	7
SCF_3_I	4	1:2:16:45	64	8 × 8	0.650	5.567	2[a]	0.537	5.577	17
SAR_8bit	9	1:1:2:4:8:16:32:64:128	256	16 × 16	0.800	32.074	602	0.695	32.111	235
SAR_9bit	10	1:1:2:4:8:16:32:64:128:256	512	32 × 16	1.077	38.072	20503	0.878	38.654	681
SAR_10bit	11	1:1:2:4:8:16:32:64:128:256:512	1024	32 × 32	–	–	–[b]	1.146	45.515	5130

[a] This runtime reported in [10] is 46.9 s, but the same algorithm implemented by us took 2 s. This discrepancy may be caused by the difference in programming and runtime platforms.

[b] No result is reported because the runtime was too long.

TABLE III

COMPARISON OF CAPACITOR ARRAYS WITH AND WITHOUT DUMMY CAPACITORS BASED ON PROPOSED ALGORITHM

Array Name	No. of Cap.	Capacitance Ratio	No. of Unit Cap.	Without Dummy Capacitors				With Dummy Capacitors			
				Matrix Size	Max. M (%)	L	Time (s)	Matrix Size	Max. M (%)	L	Time (s)
SCF_1	5	1.2:5.8:7:7:8	30	6 × 5	0.290	9.644	71	6 × 6	0.543	9.633	88
SCF_2	5	1:1.4:2:9.2:17	32	8 × 4	0.878	9.293	73	6 × 6	0.878	9.339	91
SAR_9bit	10	1:1:2:4:8:16:32:64:128:256	512	32 × 16	0.878	38.654	681	23 × 23	0.695	39.243	820

u_{10}	u_D	u_D	u_D	u_{10}	u_{10}	u_{10}	u_8	u_9	u_{10}	u_9	u_9	u_{10}	u_{10}	u_8	u_8	u_9	u_{10}	u_D	u_9	u_D	u_D	u_{10}
u_{10}	u_9	u_{10}	u_{10}	u_9	u_{10}	u_9	u_8	u_{10}	u_7	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_D	u_{10}	u_{10}
u_{10}	u_9	u_{10}	u_9	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_9	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_9	u_{10}	u_9	u_9	u_9	u_D
u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_6	u_{10}	u_8	u_{10}	u_7	u_{10}	u_{10}	u_{10}	u_9	u_{10}	u_{10}	u_{10}	u_7	u_{10}	u_{10}	u_9	u_{10}	u_8
u_{10}	u_{10}	u_8	u_{10}	u_9	u_{10}	u_9	u_{10}	u_9	u_{10}	u_9	u_9	u_9	u_9	u_6	u_9	u_{10}	u_{10}	u_9	u_{10}	u_{10}	u_{10}	u_{10}
u_9	u_{10}	u_7	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_9	u_9	u_{10}	u_8	u_8	u_{10}	u_9	u_{10}	u_{10}	u_9	u_{10}	u_7	u_{10}	u_9	u_{10}
u_9	u_8	u_{10}	u_9	u_9	u_{10}	u_8	u_9	u_8	u_8	u_8	u_{10}	u_{10}	u_8	u_7	u_8	u_9	u_8	u_9	u_{10}	u_{10}	u_9	u_8
u_{10}	u_9	u_8	u_7	u_{10}	u_8	u_9	u_{10}	u_6	u_7	u_{10}	u_{10}	u_7	u_8	u_9	u_{10}	u_9	u_8	u_5	u_{10}	u_{10}	u_{10}	u_8
u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_9	u_8	u_{10}	u_{10}	u_7	u_9	u_{10}	u_7	u_7	u_{10}	u_9	u_{10}	u_4	u_{10}	u_{10}	u_9	u_{10}	u_9
u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_6	u_9	u_9	u_9	u_{10}	u_9	u_5	u_6	u_8	u_{10}	u_8	u_9	u_{10}	u_8	u_{10}	u_{10}	u_6	u_{10}
u_8	u_7	u_9	u_9	u_5	u_{10}	u_8	u_7	u_8	u_9	u_4	u_D	u_9	u_3	u_7	u_9	u_9	u_8	u_9	u_{10}	u_{10}	u_{10}	u_9
u_9	u_{10}	u_9	u_{10}	u_7	u_{10}	u_9	u_8	u_6	u_5	u_6	u_1	u_6	u_5	u_6	u_8	u_9	u_{10}	u_7	u_{10}	u_9	u_{10}	u_9
u_9	u_{10}	u_{10}	u_{10}	u_9	u_8	u_9	u_9	u_7	u_3	u_9	u_2	u_4	u_9	u_8	u_7	u_8	u_{10}	u_5	u_9	u_9	u_7	u_8
u_{10}	u_6	u_{10}	u_{10}	u_8	u_{10}	u_9	u_8	u_{10}	u_8	u_6	u_5	u_9	u_{10}	u_9	u_9	u_9	u_6	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}
u_9	u_{10}	u_9	u_{10}	u_{10}	u_4	u_{10}	u_9	u_{10}	u_7	u_7	u_{10}	u_9	u_7	u_{10}	u_{10}	u_8	u_9	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}
u_8	u_{10}	u_{10}	u_{10}	u_5	u_8	u_9	u_{10}	u_9	u_8	u_7	u_{10}	u_{10}	u_7	u_6	u_{10}	u_9	u_8	u_{10}	u_7	u_8	u_9	u_{10}
u_8	u_9	u_{10}	u_{10}	u_9	u_8	u_9	u_8	u_7	u_8	u_{10}	u_{10}	u_8	u_8	u_8	u_9	u_8	u_{10}	u_9	u_9	u_{10}	u_8	u_9
u_{10}	u_9	u_{10}	u_7	u_{10}	u_9	u_{10}	u_{10}	u_9	u_{10}	u_8	u_8	u_{10}	u_9	u_9	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_7	u_{10}	u_9
u_{10}	u_{10}	u_{10}	u_{10}	u_9	u_{10}	u_{10}	u_9	u_6	u_9	u_9	u_9	u_9	u_{10}	u_9	u_{10}	u_9	u_{10}	u_9	u_{10}	u_8	u_{10}	u_{10}
u_8	u_{10}	u_9	u_{10}	u_{10}	u_7	u_{10}	u_{10}	u_{10}	u_9	u_{10}	u_{10}	u_{10}	u_7	u_{10}	u_8	u_{10}	u_6	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}
u_D	u_9	u_9	u_9	u_{10}	u_9	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_9	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_9	u_{10}	u_9	u_{10}
u_{10}	u_{10}	u_D	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_{10}	u_7	u_{10}	u_8	u_9	u_{10}	u_9	u_{10}	u_{10}	u_9	u_{10}
u_{10}	u_D	u_D	u_9	u_D	u_{10}	u_9	u_8	u_8	u_{10}	u_{10}	u_9	u_9	u_{10}	u_9	u_8	u_{10}	u_{10}	u_{10}	u_D	u_D	u_D	u_{10}

Fig. 20. Placement result of SAR_9bit with dummy unit capacitors in square matrix $A_{23\times23}$. Dummy unit capacitors are colored in gray.

in a rectangular matrix $A_{6\times5}$. Since the shape of $A_{6\times5}$ is very close to a square, it can be regarded as a compact placement for SCF_1. Since $A_{6\times6}$ is larger than $A_{6\times5}$ by one column, it increases the relative distances between unit capacitors, which may induce mismatch. Fig. 19 shows the placement result of SCF_1 with dummy unit capacitors. The placement is not as compact as that in Fig. 16(c).

For the placement of SCF_2, the original matrix $A_{8\times4}$ has a shape that is far from a square. If the placement is changed into a square matrix $A_{6\times6}$, a more compact placement for SCF_2 can be obtained. Since $A_{6\times6}$ provides a more compact placement than $A_{8\times4}$, the placement of SCF_2 with dummy unit capacitors has larger overall correlation coefficients than that without dummy unit capacitors. Similarly, the shape of the original matrix $A_{32\times16}$ for SAR_9bit is far from a square. A more compact placement for SAR_9bit can be obtained if it is placed into a square matrix $A_{23\times23}$. The placement results show that using square matrix $A_{23\times23}$ achieves a smaller oxide-gradient-induced mismatch and larger overall correlation coefficients than those of the rectangular matrix $A_{32\times16}$. Fig. 20 shows the resulting placement of SAR_9bit with dummy unit capacitors.

Based on the experimental results, the following observations were made.

1) If the shape of a rectangular matrix is very close to a square, it can be regarded as a compact placement. Thus, changing the placement into a square matrix is not necessary.
2) If the shape of an original matrix is far from a square, changing it into a square matrix by inserting some dummy unit capacitors will make it a more compact placement and obtain better matching.

VII. CONCLUSION

An SA-based approach for implementing a common-centroid placement with the highest possible degree of dispersion for an arbitrary-ratio capacitor array under the consideration of systematic and random mismatches was proposed. A pair-sequence representation for a common-centroid placement and associated perturbations were also proposed. The adjacency constraint for noninteger-ratio capacitors in a common-centroid placement and the arrangement of dummy capacitors in a compact square placement were discussed. The experimental results showed that the proposed common-centroid placement approach was effective in reducing capacitor mismatch and that the resulting placements were always feasible for noninteger-ratio capacitor layouts.

References

[1] C.-W. Lin, J.-M. Lin, Y.-C. Chiu, C.-P. Huang, and S.-J. Chang, "Common-centroid capacitor placement considering systematic and random mismatches in analog integrated circuits," in *Proc. ACM/IEEE Des. Autom. Conf.*, Jun. 2011, pp. 528–533.

[2] D. B. Ribner and M. A. Copeland, "Biquad alternatives for high-frequency switched-capacitor filters," *IEEE J. Solid-State Circuits*, vol. 20, no. 6, pp. 1085–1095, Dec. 1985.

[3] D. Khalil, M. Dessouky, V. Bourguet, M.-M. Louerat, A. Cathelin, and H. Ragai, "Evaluation of capacitor ratios in automated accurate common-centroid capacitor arrays," in *Proc. IEEE Int. Symp. Qual. Electron. Des.*, Mar. 2005, pp. 143–147.

[4] M. J. McNutt, S. LeMarquis, and J. L. Dunkley, "Systematic capacitance matching errors and corrective layout procedures," *IEEE J. Solid-State Circuits*, vol. 29, no. 5, pp. 611–616, May 1994.

[5] J.-B. Shyu, G. C. Temes, and F. Krummenacher, "Random error effects in matched MOS capacitors and current sources," *IEEE J. Solid-State Circuits*, vol. 19, no. 6, pp. 948–956, Dec. 1984.

[6] E. Felt, A. Narayan, and A. Sangiovanni-Vincentelli, "Measurement and modeling of MOS transistor current mismatch in analog IC's," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, Nov. 1994, pp. 272–277.

[7] A. Hastings, *The Art of Analog Layout*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2006.

[8] M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, "Matching properties of MOS transistors," *IEEE J. Solid-State Circuits*, vol. 24, no. 5, pp. 1433–1440, Oct. 1989.

[9] P.-W. Luo, J.-E. Chen, C.-L. Wey, L.-C. Cheng, J.-J. Chen, and W.-C. Wu, "Impact of capacitance correlation on yield enhancement of mixed-signal/analog integrated circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 11, pp. 2097–2101, Nov. 2008.

[10] J.-E. Chen, P.-W. Luo, and C.-L. Wey, "Placement optimization for yield improvement of switched-capacitor analog integrated circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 29, no. 2, pp. 313–318, Feb. 2010.

[11] D. Sayed and M. Dessouky, "Automatic generation of common-centroid capacitor arrays with arbitrary capacitor ratio," in *Proc. ACM/IEEE Design Autom. Test Eur. Conf. Exhibit.*, Aug. 2002, pp. 576–580.

[12] D. A. Johns and K. Martin, *Analog Integrated Circuit Design*. New York: Wiley, 1997.

[13] D. Khalil, M. Dessouky, V. Bourguet, M.-M. Louerat, A. Cathelin, and H. Ragai, "Compensated layout for automated accurate common-centroid capacitor arrays," in *Proc. IEEE Int. Conf. Elect. Electron. Comput. Eng.*, Sep. 2004, pp. 481–484.

[14] R. C. J. Taylor, "Switched capacitor filters," in *Analog Circuit Design*, R. J. van de Plassche, W. M. C. Sansen, and J. H. Huijsing, Eds. Norwell, MA: Kluwer, 1995, pp. 203–225.

[15] M. Strasser, M. Eick, H. Graeb, U. Schlichtmann, and F. M. Johannes, "Deterministic analog circuit placement using hierarchically bounded enumeration and enhanced shape functions," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, Nov. 2008, pp. 306–313.

[16] L. Xiao and E. F. Y. Young, "Analog placement with common centroid and 1-D symmetry constraints," in *Proc. IEEE/ACM Asia South Pacific Des. Autom. Conf.*, Jan. 2009, pp. 353–360.

[17] Q. Ma, L. Xiao, Y.-C. Tam, and E. F. Y. Young, "Simultaneous handling of symmetry, common centroid, and general placement constraints," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 1, pp. 85–95, Jan. 2011.

[18] Q. Ma, E. F. Y. Young, and K. P. Pun, "Analog placement with common centroid constraints," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, Nov. 2007, pp. 579–585.

[19] M. P.-H. Lin, H. Zhang, M. D. F. Wong, and Y.-W. Chang, "Thermal-driven analog placement considering device matching," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 3, pp. 325–336, Mar. 2011.

[20] J.-E. Chen, P.-W. Luo, and C.-L. Wey, "Yield evaluation of analog placement with arbitrary capacitor ratio," in *Proc. IEEE Int. Symp. Qual. Electron. Des.*, Mar. 2009, pp. 179–184.

[21] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

Cheng-Wu Lin received the B.S. degree from National Central University, Jhongli, Taiwan, in 2001, and the M.S. degree from National Cheng Kung University, Tainan, Taiwan, in 2006, where he is currently pursuing the Ph.D. degree, all in electrical engineering.

His current research interests include sizing and placement for analog and mixed-signal circuits.

Jai-Ming Lin received the B.S., M.S., and Ph.D. degrees in computer science from National Chiao Tung University, Hsinchu, Taiwan, in 1996, 1998, and 2002, respectively.

From 2002 to 2007, he was an Assistant Project Leader with the Computer-Aided Design Team, Realtek Corporation, Hsinchu. He is currently an Assistant Professor with the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan. His current research interests include floorplanning, placement, routing, and clock tree synthesis.

Yen-Chih Chiu received the M.S. degree in electrical engineering from National Cheng Kung University, Tainan, Taiwan, in 2011.

He is currently an Engineer with the Global Unichip Corporation, Hsinchu, Taiwan, where he is involved in electronic design automation flow. His current research interests include floorplanning, placement, and routing.

Chun-Po Huang received the B.S. degree from National Cheng Kung University, Tainan, Taiwan, in 2008, where he is currently pursuing the Ph.D. degree, both in electrical engineering.

His current research interests include design automation for high-speed and low-power analog-to-digital converters.

Soon-Jyh Chang (M'03) received the B.S. degree in electrical engineering from National Central University, Jhongli, Taiwan, in 1991, and the M.S. and Ph.D. degrees in electronic engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1996 and 2002, respectively.

Since 2003, he has been with the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, where he is currently a Professor and the Director of the Electrical Laboratories. He has authored or co-authored over 100 technical papers and holds seven patents. His current research interests include design, testing, and design automation for analog and mixed-signal circuits.

Dr. Chang was a recipient of many technical awards, including the Chip Implementation Center Outstanding Chip Award in 2008, 2011, and 2012, the Best Paper Award of the Institute of Electronics, Information, and Communication Engineers in 2010, the Gold Prize of the Macronix Golden Silicon Award in 2010, the Best Gold Member Award from the IEEE Tainan Section in 2010, the International Solid State Circuits Conference/Design Automation Conference Student Design Contest in 2011, and the International Symposium on Integrated Circuits Chip Design Competition in 2011. Since 2009, he has been the Chair of the IEEE Solid-State Circuits Society, Tainan Chapter. He was the Technical Program Co-chair of the IEEE International Symposium on Next-Generation Electronics in 2010. He was a Committee Member of the IEEE Asian Test Symposium in 2009, the Asia and South Pacific Design Automation Conference in 2010, the International Symposium on VLSI Design, Automation, and Test in 2009, 2010, and 2012, and the Asian Solid-State Circuits Conference in 2009 and 2011.