A Novel Design Flow for Dummy Fill Using Boolean Mask Operations

Tseng-Chin Luo, Mango C.-T. Chao, Philip A. Fisher, and Chun-Ren Kuo

Abstract—**Dummy fill has been demonstrated to be an effective technique to reduce process variation and improve manufacturability for advanced integrated circuit (IC) designs. However, the computation load, often several days for a realistic IC design, is a significant portion of the cycle time for delivering first silicon on new or modified designs. In this paper, we propose a novel design flow and dummy-fill algorithm based on Boolean operations, which greatly improves computational efficiency and pattern density uniformity, and enables dummy generation to be combined with the mask-preparation Boolean operations performed by the mask-fabrication facility. Mask data preparation can be performed in parallel with dummy generation and post-dummy simulation checks at the design house, resulting in improved first-silicon cycle time. Experimental results demonstrate these benefits in the context of an advanced foundry process technology.**

Index Terms—**Boolean operation, dummy fill.**

I. Introduction

As PROCESS technologies continually scale, the process variation due to each process step has greatly increased and has gradually become one of the most critical factors for integrated circuit (IC) manufacturing. In order to reduce the level of process variability, several physical-design techniques have been proposed to constrain or modify the original IC layout for improved manufacturability. Such design-for-manufacturability (DFM) techniques include: 1) symmetry or matching constraints [1], [2], which can reduce the potential voltage offset and power-supply rejection ratio for analog circuits; 2) via doubling [3], [4] and wire spreading [5], [6], which can lower the impact of random defects by adding redundancy; 3) microregularity constraints, such as single orientation of critical dimension (CD) lines and constant poly pitch, for resolution enhancement technique compatibility [7]; and 4) dummy fill, which inserts dummy layout patterns (which are not electrically active) into the design layout to improve pattern uniformity and thereby improve the uniformity of planarization, stress balance, and thermal anneal. In this paper, we will focus on improvements to the dummy-fill algorithm and its use in the design flow.

Conventional dummy fill requires two steps. The first step is to define the layout pattern of *dummy cells* [8]– [11], which usually follows the guidance of IC foundries based on their process characteristics. The second step is the *dummy-cell placement*, which places the defined dummy cells into the empty space of a design layout to balance its feature density. Several algorithms for dummy-cell placement have been proposed according to various density models, such as interlayer dielectric (ILD), chemical mechanical polishing (CMP) models [12]– [14], Cu CMP models [15], [16], and a hybrid model combining both CMP and electroplating topography models [17].

Fig. 1(a) shows the conventional design and tape-out flow, in which the dummy fill is applied by the design house after the design layout has passed design rule checking (DRC) and layout versus schematic check (LVS), resulting in the final graphic database system (GDS) file to be delivered to a IC foundry for tape-out (hereafter, the "tape-out GDS file"). After the dummy cells are placed into the layout, a final simulation verification (the post-dummy simulation step) is performed, and the tape-out GDS file is sent to an IC foundry, which then performs the following steps: 1) computation of the corresponding mask layers; 2) visual inspection of the mask layer data, commonly referred to as "Jobview," and optical proximity correction (OPC); and 3) generation of the reticle set for wafer processing. Traditionally, particularly for technologies up to 65 nm, the post-dummy simulation is generally used only as a safety verification check to assess small changes in the design performance due to parasitics and layout-dependent effects introduced by the dummy patterns. Unless serious timing impact is observed, the results of this simulation very rarely drive modifications to the original design before tape-out because repeated iterations of layout-change, dummy-fill, and simulation are costly in terms of both design cycle time and computational overhead. In this design flow, although the design is known to be functional after DRC and LVS, before wafer processing can begin, several days must be spent on dummy generation and post-dummy simulation. Furthermore, the GDS file size increases dramatically with the addition of dummy patterns, often by a factor of 10 or more, particularly for advanced

Manuscript received November 6, 2010; revised January 20, 2012 and February 5, 2012; accepted February 28, 2012. Date of publication March 8, 2012; date of current version July 31, 2012.

T.-C. Luo is with the Taiwan Semiconductor Manufacturing Corporation, Hsinchu 300, Taiwan, and also with the Department of Electronics Engineering, Institute of Electronics, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: tclo@tsmc.com).

M. C.-T. Chao and C.-R. Kuo are with the Department of Electronics Engineering, Institute of Electronics, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: mango@faculty.nctu.edu.tw; plk1986@gmail.com).

P. A. Fisher is with the Taiwan Semiconductor Manufacturing Corporation, Hsinchu 300, Taiwan (e-mail: philip_a_fisher@yahoo.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSM.2012.2190431

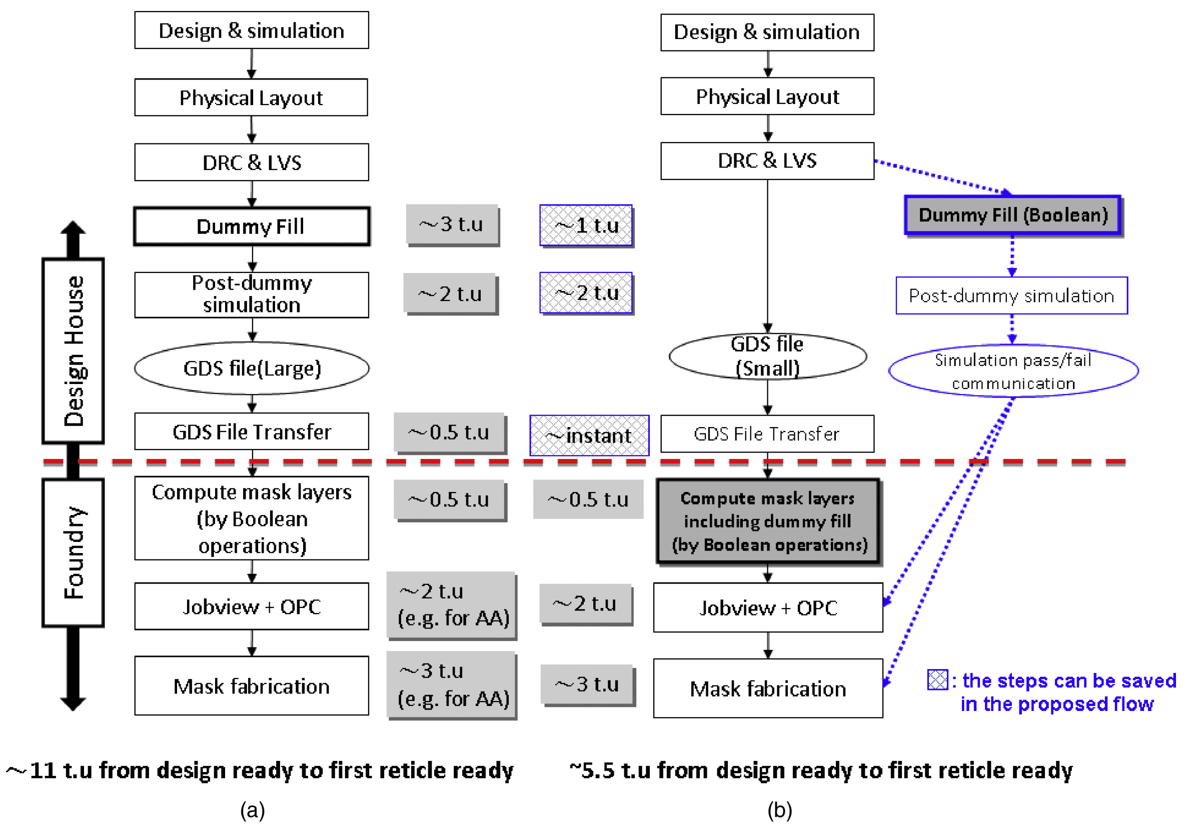


Fig. 1. Conventional dummy fill versus proposed dummy fill. Units of t.u. are arbitrary units of time, on the order of 24 h, although precise execution times of course vary for different facilities.

process technologies. This large GDS file is often difficult to handle, and dummy generation by typical algorithms and post-dummy simulation is often a large computational load for the workstations used by a typical design house for DRC and LVS.

It should be noted that although generally all design modifications are based on performance simulations before dummy fill, for high-performance designs in advanced technologies below 65 nm, the parasitic impact of dummy patterns on product performance may potentially be large enough to require at least minor layout modification, and possibly one or more iteration of layout change, dummy fill, and simulation. Although such cases are relatively rare in the authors' experience, if multiple iterations of layout change and post-dummy simulation were required, the benefits of our proposed algorithm would in fact be multiplied because each iteration would benefit from the improved computational efficiency and database hierarchy. However, in order to make our discussion applicable to the largest range of design cases, we focus on the case where only a single post-dummy simulation is required, while noting that the benefits of our proposed method are multiplicative in the rare cases where multiple iterations are needed.

Although the use of Boolean operations for computation of mask layers from the "tape-out GDS" has been common practice for many technology generations, to the best of our knowledge, this paper is the first report of successful use of Boolean operations to accelerate dummy-pattern generation for advanced logic technologies, while simultaneously achieving improvements in layout density uniformity and a more rapid tape-out flow. In this paper, we propose a novel design and tape-out flow illustrated in Fig. 1(b). In our proposed flow, the dummy-fill operation is greatly simplified by defining the dummy fill in terms of Boolean operations (hereafter, referred to as *Boolean dummy fill*). Instead of performing the dummy fill by repeatedly searching for regions of empty space in the design layout and inserting predefined dummy cells, as is commonly practiced in the conventional flow, we perform the dummy fill by applying proper Boolean operations similar to those that are already used to compute the mask layers based on the tape-out GDS. In other words, the dummy fill in the proposed flow is performed by the same method as the computation of mask layers by the IC foundry. Thus, the dummy generation may be merged with mask computation Booleans and may be performed by the foundry. Because the Boolean operations for dummy fill are much simpler than those typically required for mask computation, they represent only a negligible additional computational load, as foundries typically have the computational resources appropriate to the much more complicated mask computation and OPC operations.

In advanced technology nodes beyond 65 nm, most design houses perform timing checks both before and after dummy fill to insure chip performance because the dummy fill may introduce a significant parasitic capacitance load on the main circuit patterns. The parasitic capacitance change after dummy fill depends mainly on two factors: 1) dummy cell geometry, and 2) the distance between the dummy cells and the main circuit pattern. As a result, dummy-fill operations have the potential to affect chip timing and this effect becomes more significant in more advanced technology nodes because more and increasingly complex dummy patterns are required for the optimization of process variation. Generally, the performance impact of the dummy patterns results in a simple performance shift, but does not result in changes to either design layout or dummy fill. The post-dummy simulation is thus generally a necessary safety check to insure that the design delivers adequate performance, but rarely drives changes in the database. However, for demanding high-performance designs, in principle, it is possible that the results of post-dummy simulations may result in one or more iterations of either design layout change or fine-tuning of the dummy generation to compensate for timing impact. Our proposed method would remain beneficial even in such cases. The main benefit of our method is due to: 1) the database retaining a high degree of hierarchy when the dummy fill is performed, and 2) the use of Boolean operations rather than an iterative algorithm to perform the dummy fill. Our proposed method permits fine-tuning of dummy cell geometry to the same degree as conventional dummy fill, while still retaining the benefits of improved hierarchy and speed of execution. If additional post-dummy simulations or iterations on dummy-fill geometry are required, our proposed method provides an even greater net gain because each iteration can be performed more quickly on a smaller hierarchical database.

Further efficiency can be obtained because the Boolean dummy-fill and post-dummy simulations to assess the effect of dummy patterns on product performance, not only incur a much reduced computational load by the use of Boolean dummy fill but also these tasks can be performed simultaneously with the computation of the mask layers and OPC by the foundry, so several days of overhead can be eliminated. Once post-dummy simulation is complete, approval for mask fabrication (or mask release) can be communicated and mask fabrication (or wafer processing) can proceed immediately. In principle, the design house could completely forgo dummy-fill and postsimulation, but even when it is desired, it imposes no delay in the tape-out flow. Although the work of [19] demonstrated a dummy-insertion flow with some features that are present in our proposed flow, and may have been implemented by Boolean operations, there was no detailed discussion of the method of implementation, performance metrics, or the use of this flow to accelerate the overall tape-out process, and the scope was limited to adding dummy poly to address a specific process issue in a relatively old technology having poly spacing of $\sim$0.35 μm. Nevertheless, because the work of [19] is the only previous report of a possible partial implementation of our proposed flow of which we are aware, similarities and differences with this paper will be specifically addressed in the relevant sections below.

In summary, Boolean dummy fill and our proposed design and tape-out flow provides and enables the following benefits. First, the runtime overhead of the proposed dummy fill depends on the runtime of performing additional Boolean mask operations among the GDS layers, which is much faster than performing extensive GDS searching operations as is done in conventional dummy-fill algorithms, and thus is a smaller computational load for the design house. Second, Boolean dummy fill delivers improved pattern uniformity because the pattern uniformity of the Boolean dummy fill is not limited by the dimensions of the predefined dummy cells as is the case with conventional dummy-fill algorithms. Third, dummy generation and postsimulation at the design house are performed in parallel with dummy generation, mask-layer computation, and OPC (and potentially even mask writing), eliminating several days of overhead from the tape-out flow, enabling more rapid silicon verification of new or modified designs. The experimental results below, based on advanced process technologies will further demonstrate the efficiency and effectiveness of the proposed design and tape-out flow enabled by Boolean dummy fill.

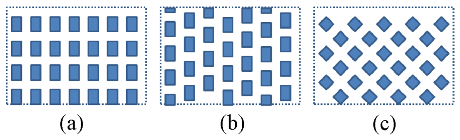


Fig. 2. Example of different dummy-pattern styles. (a) Griding. (b) Staggered. (c) Diamond.

II. Background

A. Long-Range Versus Short-Range Dummy Patterns

Dummy patterns can be categorized as either *short-range* or *long-range* dummy patterns. Short-range dummy patterns are placed close to the active patterns to eliminate potential local process variation, such as mechanical-stress effects [18] and proximity effects [19], [20]. The mechanical-stress effect is caused by the lattice mismatch of different films. The level of strain of different active regions may affect the silicon bandgap, diffusivity of impurities, and mobility, and thus may result in variation of devices' threshold voltage, transconductance, saturation drain current, and off-state drain current, even though the CDs of the devices are the same [18]. Dummy patterns can be inserted as stress buffers to balance this mechanical stress. The optical proximity effect is generated by optical diffraction during lithography such that the CD of an isolated layout pattern may be different from that of a nominally identical pattern in a dense layout. To resolve this proximity effect, dummy patterns with comparable dimensions and regular spacing must be inserted in the immediate neighborhood of isolated active patterns. Thus, the dimensions of short-range dummy patterns are relatively small [19], [20].

On the other hand, long-range dummy patterns are placed at a greater distance from the active patterns to eliminate potential global process variations caused by unbalanced layout density and low layout uniformity over distances of several micrometers or tens of micrometers. The sources of such longer range process variation include thermal anneal (rapid thermal annealing, Flash, and Laser) [21], CMP [9], [22], and etch microloading [9], [23]. Since the length scale associated with these process variations is larger, the dimensions of long-range dummy pattern cells are relatively large.

B. Dummy-Pattern Style

Fig. 2 shows different layout styles of dummy patterns. The first style (also the most conventional one) is the grid style, in which dummy patterns are placed in identical rectangles and aligned in the Y-direction with a fixed X-direction spacing. The second style is the staggered style, in which rectangular patterns in adjacent columns are placed with a fixed offset in the Y-direction. The staggered style results in more uniform fluid dynamics and reduces the competition for etch reactants to achieve a more uniform etch rate, and thus this style is preferred for reducing microloading effects [9]. The third style is the diamond-shape style, in which the dummy patterns are identical diamond shapes and placed with fixed spacing. This diamond-shape style is most commonly used for dummy metals because it minimizes the effective coupling capacitance [10]. The choice of the dummy-pattern style is determined by the characteristics of the adopted process technology and may differ for different IC foundries.

Note that the example shown in Fig. 2 is only for one GDS layer. In practice, dummy patterns are simultaneously added into different layers, such as poly, diffusion, and metal. Thus, dummy patterns are required for each corresponding layer. In fact, the dummy patterns for both poly and diffusion layers are designed and inserted simultaneously. The dummy pattern for metal layers are designed and inserted independently from the poly and diffusion layers.

C. Computation of Mask Layers Using Boolean Operations

Boolean operations have been widely used for IC mask generation [24], [25]. When the GDS file of a design is sent to an IC foundry or mask vendor, all the mask layers must first be computed by Boolean operations from the tape-out GDS layers (often referred to as the computer-aided design layers). Note that the number of GDS layers present in a typical tape-out GDS file is around 20, whereas the number of mask layers in a typical CMOS mask set is around 40 (depending on the specific process technology). Some mask layers, such as PW, Ldd, and SiGe, are not present in the tape-out GDS, but can be computed by applying Boolean mask operations (such as NOT, OR, SIZING) on the tape-out GDS layers. In general, the runtime of computing a mask layer is less than 1 h. This runtime is only a small portion of the entire process of mask-set generation, whose computation time is dominated by OPC. To perform OPC on one mask layer may take days for large advanced designs.

III. Boolean-Operation Dummy Fill

A. Overall Flow

According to the design flow shown in Fig. 1(b), the proposed Boolean dummy fill is performed at the same time as the mask layers are computed from the GDS layers. In other words, with Boolean dummy fill, it is no longer necessary to search for empty space in the design layout and insert dummy cells as in the conventional flow. Instead, proper Boolean mask operations must be designed for directly generating the corresponding mask layers with dummy patterns inserted. In principle, the responsibility for performing dummy fill can be transferred from design houses to IC foundries, where it is performed much more efficiently as part of the mask-layer Boolean operations. Design houses need only provide their verified design layout without the addition of dummy patterns, and the IC foundries can proceed as suggested in the flow of Fig. 1(b). If postsimulation is required, the design house can perform the small subset of the mask Boolean operations that perform the dummy fill, and carry out post-dummy simulation while the foundry is performing the OPC computations for the first layers in the process flow. This achieves the following benefits. First, less computation time is required at the design house due to the improved computational efficiency of Boolean dummy generation. Second, the computation time for dummy generation and post-dummy simulation at the design house is no longer a bottleneck in the tape-out flow due to the parallel generation of dummy patterns at the design house and foundry. Third, the size of the tape-out GDS file transmitted from the design house to the foundry is dramatically reduced, so this transmission delay becomes negligible. Since the runtime of computing the additional Boolean operations for dummy fill is far less than that of the original Boolean operations used to compute the mask layers, there is relatively little additional computational overhead at the foundry.

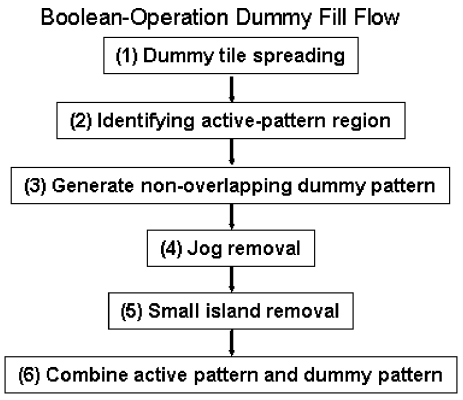

Boolean-Operation Dummy Fill Flow
(1) Dummy tile spreading
(2) Identifying active-pattern region
(3) Generate non-overlapping dummy pattern
(4) Jog removal
(5) Small island removal
(6) Combine active pattern and dummy pattern

Fig. 3. Steps of the proposed Boolean-operation dummy fill.

The following are the input and output of the proposed Boolean-operation dummy fill.

1) *Input.*
   a) Active patterns in the tape-out GDS file provided by the design house.
   b) The dummy-pattern style of both long-range and short-range dummy patterns.
   c) The spacing between the active patterns and dummy patterns.
   d) The placement range (distance to the active pattern) of short-range and long-range dummy patterns.
2) *Output.*
   a) A set of Boolean mask operations that can generate the mask layers with dummy patterns inserted.

B. Detailed Steps of Boolean-Operation Dummy Fill

Fig. 3 lists the six steps used in the proposed Boolean-operation dummy fill, including: 1) dummy-tile spreading; 2) identifying active-pattern regions; 3) generating nonoverlapping dummy patterns; 4) jog removal; 5) small-island removal; and 6) combining active patterns and dummy patterns. The details of each step are presented in the following sections. Steps 1–4 and 6 were performed by a single Mentor Calibre script. For reasons of convenience, Step 5 was performed by a separate script as will be explained below, but the entire flow could easily be performed by a single integrated script, which would result in runtime reductions even greater than those we report here. GDS file format is used throughout the flow. Note that in the following example, we generate dummy patterns only on the poly layer and use only the short-range dummy cells. A similar method can be applied to generate dummy patterns for other layers. We will later show how to generate dummy patterns with the use of both short-range and long-range dummy cells simultaneously. A similar flow for dummy poly insertion is described in [19] in a successful addition of dummy poly patterns to reduce the difference in contact ILD deposition and etch behaviors between gates with neighboring gates on both sides and gates with no neighboring gate on one side. Our flow and the flow in [19] both exploit: 1) dummy-tile spreading; 2) identifying active-pattern region; 3) generating nonoverlapping dummy patterns; 4) jog removal; 5) small-island removal; and 6) combining active patterns and dummy patterns; these steps may have been implemented using Boolean operations, but [19] does not state the software tool (e.g., Mentor Calibre) or algorithm (e.g., Boolean operation or search algorithm) used to implement the flow in their work. Because the flow of [19] begins with identifying the poly-edge candidates for dummy pattern addition prior to dummy-tile spreading, it appears probable that a search algorithm was employed in this paper for this initial step. Furthermore, the algorithm implemented in [19] appeared to be insufficiently robust for fully automated use, because the flow in [19] explicitly included a final step where the layout after dummy insertion must again be searched for noncompliant poly edges, and these must be edited manually. This final search of the database after dummy insertion is likely to incur substantial computational overhead. Finally, [19] was implemented on a relatively old technology (having poly pitch of $\sim$0.35 μm), and the scope appears to be limited to the addition of dummy poly patterns to address a specific process issue, without a discussion of the computational overhead or optimization of the implementation algorithm. Our paper is broadly applicable to multiple sizes of dummy patterns on multiple layers and has been successfully used in a fully automated mode on multiple customer products; here-in we will show that implementation by Boolean operation achieves superior computational performance, improved pattern uniformity, and accelerated tape-out compared to the current state-of-the-art dummy-insertion tool provided by a major foundry.

1) *Dummy-Tile Spreading:* The objective of the dummy-tile spreading is to generate the dummy patterns (short-range or long-range) for the entire poly, diffusion, or metal layer.

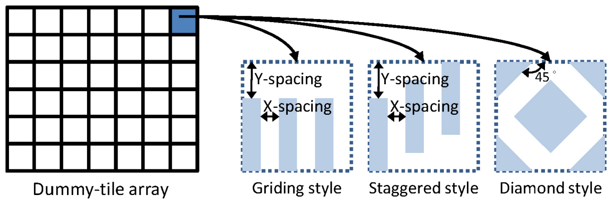


Fig. 4. Dummy-tile design.

Those full-layer dummy patterns are then used as the operands of the Boolean mask operations to compute the final physical dummy patterns. This full-layer dummy pattern is generated by duplicating the predefined dummy tile over the entire design window. The predefined dummy tile is designed according to the adopted dummy-pattern style. Fig. 4 shows examples of designing the corresponding dummy tile for dummy-tile spreading. Once the dummy tile is defined, this layout spreading can be easily performed by a commercial layout tool, such as Virtuoso, Laker, or Mentor Calibre, as was used in this paper.

2) *Identifying Active-Pattern Region:* Before generating the final dummy patterns, it is first necessary to identify the active-pattern region and eliminate this region from the full-layer dummy pattern. This active-pattern region includes the area of the active patterns as well as the spacing between the active patterns and dummy patterns, which can be computed by (1). Note that when calculating the active-pattern region for dummy poly patterns, the active patterns should include both poly and diffusion patterns since a poly dummy line cannot overlap with active diffusion patterns. The active metal patterns are excluded from the above active-pattern region. In the following discussion, GDS layer IDs (PO;0) and (PO;1) represent the active patterns and the dummy patterns on the poly layer, respectively. (AA;0) and (AA;1) represent the active patterns and the dummy patterns on the diffusion layer, respectively, as follows:

$$(\text{AA;0 OR PO;0}) \text{ SIZING } s_1. \tag{1}$$

In (1), (PO;0 OR AA:0) represents the union of active poly patterns and active diffusion patterns. The (SIZINGs_1) operation enlarges the designated patterns by s_1 μm, where s_1 denotes the minimum spacing between active patterns and dummy patterns.

3) *Generating Nonoverlapping Dummy Pattern:* In this step, we compute the final dummy patterns, denoted as *DP*, by eliminating the active-pattern region from the full-layer dummy pattern. Equation (2) lists the corresponding Boolean operations as follows:

$$\text{DP} = \text{PO;1 NOT ((AA;0 OR PO;0) SIZING } s_1). \tag{2}$$

Note that the operation "NOT" represents the difference operation. The output of the operation (A NOT B) is the layout patterns that are in A but do not overlap with the patterns in B.

4) *Jog Removal:* After eliminating the active-pattern region from the full-layer dummy patterns, some jog patterns or small-island patterns may exist among the dummy patterns as

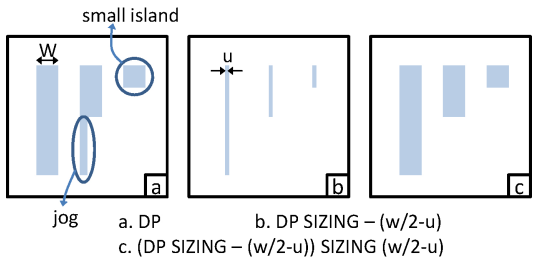


Fig. 5. Example of the jog removal.

shown in Fig. 5(a). The jog patterns and small-island patterns could become a source of defectivity during wafer processing, resulting in degraded yield [26]. In addition, these small-island patterns also significantly increase the computational overhead of the OPC due to their irregular shapes.

The objective of the current step is to remove the jog patterns from the dummy patterns by using the following Boolean operations. First, (3) is used to shrink the dummy patterns by x μm, where $x = \frac{w}{2} - u$, w represents the width of the adopted dummy pattern, and u represents the minimum layout unit. The minus sign used in the SIZING operation denotes shrinking the feature size. As illustrated in Fig. 5(b), after performing (3), the jog patterns completely disappear and the original regular patterns have been reduced to the minimum permitted dummy feature width as follows:

$$\text{DP SIZING -}x. \tag{3}$$

Next, (4) is used to enlarge the shrunk patterns by x μm. Since the jog patterns have already been eliminated, (4) simply restores the regular patterns back to their original size as illustrated in Fig. 5(c), and hence obtains dummy patterns without jogs (denoted as DP_JR) as follows:

$$\text{DP\_JR} = (\text{DP SIZING -}x) \text{ SIZING } x. \tag{4}$$

In other words, by applying both (3) and (4), any pattern with a width less than w will be eliminated and the other patterns are unchanged.

5) *Small-Island Removal:* The next step focuses on removing the small-island patterns, i.e., the patterns whose area is smaller than the minimum area permitted by the design rules of the particular process technology. The previously described jog removal can only eliminate the patterns whose width is smaller than a predefined constraint, but in many process technologies a shape with dimensions exceeding the minimum permitted width may still be forbidden by a minimum area design rule. Thus, small-island removal is performed by a Mentor Calibre DRC script, which identifies and removes all such small-island patterns. This step was performed by a DRC script that was separate from the main Calibre script used for the other Boolean operations only because such a DRC script was conveniently available. Performing this step with a separate script caused additional runtime overhead because the GDS output of the jog-removal operation was flattened for input into the small-island-removal script, but its runtime was still much faster than conventional dummy insertion as shown in Fig. 1.

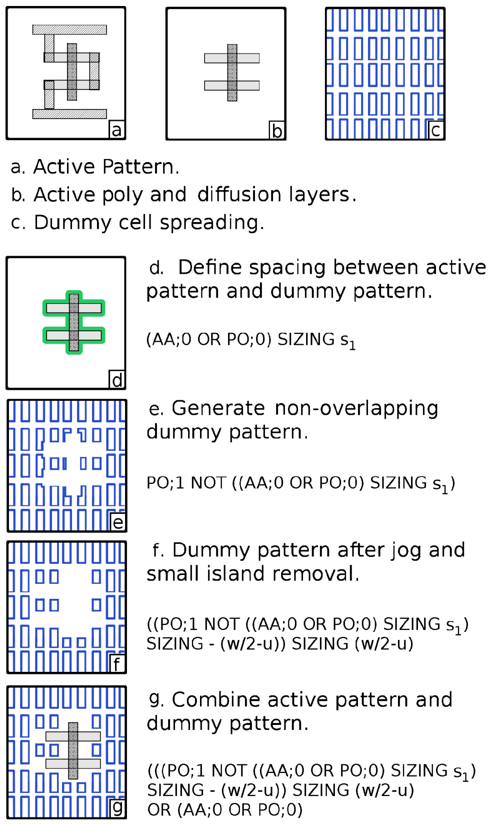

a
b
c
a. Active Pattern.
b. Active poly and diffusion layers.
c. Dummy cell spreading.
d
d. Define spacing between active pattern and dummy pattern.
(AA;0 OR PO;0) SIZING s1
e
e. Generate non-overlapping dummy pattern.
PO;1 NOT ((AA;0 OR PO;0) SIZING s1)
f
f. Dummy pattern after jog and small island removal.
((PO;1 NOT ((AA;0 OR PO;0) SIZING s1) SIZING - (w/2-u)) SIZING (w/2-u)
g
g. Combine active pattern and dummy pattern.
(((PO;1 NOT ((AA;0 OR PO;0) SIZING s1) SIZING - (w/2-u)) SIZING (w/2-u) OR (AA;0 OR PO;0)

Fig. 6. Example of Boolean-operation dummy fill using only short-range dummy patterns.

6) *Combining Active Patterns and Dummy Patterns:* The final step is to combine the generated dummy patterns with the active patterns (PO;0) by using the Boolean operation OR. Equation (5) shows the Boolean operations combining all the above steps, which can generate the final poly layer combining the dummy patterns and the active patterns as follows:

$$\begin{aligned} &\text{DP\_JR OR (AA; 0 OR PO; 0)} \\ = \quad &(((\text{PO; 1 NOT ((AA; 0 OR PO; 0) SIZING } s_1)) \\ &\text{SIZING} - x) \text{ SIZING } x) \text{ OR (AA; 0 OR PO; 0)}. \end{aligned} \tag{5}$$

7) *Example:* Fig. 6 shows an example of applying the proposed Boolean-operation dummy fill with only short-range dummy patterns. In this example, the result and the Boolean operations for each step are shown as subfigures. The active pattern used in this example is an inverter.

C. Dummy Fill With Long and Short-Range Dummy Patterns

In the following section, the use of Boolean-operation dummy fill to simultaneously place both long-range and short-range dummy patterns is demonstrated. Again, the example used is the case of generating dummy poly, but the dummy patterns for other layers can be obtained in a similar manner. In the following discussion, (PO;1) and (PO;2) represent the full-layer dummy patterns after dummy-tile spreading for the short-range dummy pattern and the long-range dummy pattern, respectively. The dummy generation is determined by the following parameters: s_1 represents the minimum spacing between active pattern and short-range dummy patterns, s_2 represents the farthest distance outside the active-pattern region, where a short-range dummy pattern can be placed, and s_3 represents the minimum spacing between the short-range patterns and the long-range patterns. These three parameters are determined by the selected process technology and predefined as inputs of the Boolean dummy fill.

The Boolean dummy fill including both long-range and short-range dummy patterns, illustrated in Fig. 7, follows the same steps shown in Fig. 3, but the step of generating nonoverlapping dummy patterns is slightly more complicated than that described in Section III-B3. Short-range dummy patterns can only be placed within a distance s_1 and a distance (s_1+s_2) of the active patterns. Thus, to compute the short-range dummy patterns, the short-range dummy patterns excluding the active-pattern region are generated by using (6), where the ((AA:0 OR PO;0) SIZING s_1) represents the active-pattern region. The resulting patterns are denoted as SRD_in and illustrated in Fig. 7(c) as follows:

$$\text{SRD\_in} = \text{PO};1 \text{ NOT } ((\text{AA}:0 \text{ OR } \text{PO};0) \text{ SIZING } s_1). \quad (6)$$

Of course a simple modification of (6) permits different values of s_1 for AA:0 and PO;0 layers, if required, but generally the same value can be used because the spacing of these two layers follows a fixed relationship set by the technology design rules. Second, the short-range dummy patterns placed outside the feasible range of the short-range patterns are generated by (7), where ((AA:0 OR PO;0) SIZING (s_1+s_2)) forms the largest region within which short-range dummy patterns can be placed around the active patterns. This pattern, denoted SRD_out, is illustrated in Fig. 7(d) as follows:

$$\text{SRD\_out} = \text{PO};1 \text{ NOT } ((\text{AA}:0 \text{ OR } \text{PO};0) \\ \text{SIZING}\,(s_1 + s_2)). \quad (7)$$

From the above intermediate results, the final short-range dummy patterns, denoted as SRD, are calculated by eliminating SRD_out from SRD_in as shown in (8). Fig. 7(e) illustrates the resulting SRD as follows:

$$\text{SRD} = \text{SRD\_in NOT SRD\_out}. \quad (8)$$

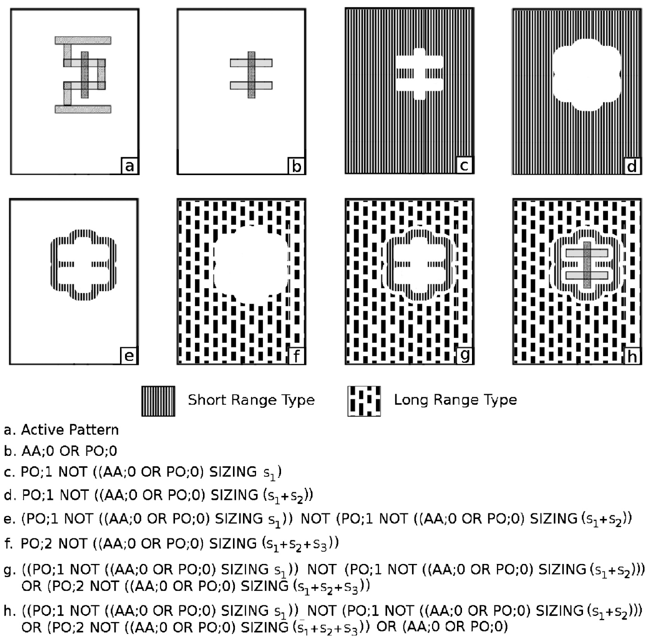
Next, the long-range dummy patterns need to be placed outside the region that is at a distance s_3 from the short-range patterns, i.e., (s_1 + s_2 + s_3) a distance from the active patterns. Thus, the long-range dummy patterns, denoted as LRD, are obtained by eliminating the invalid region from the full-layer long-range dummy patterns as shown in (9). Fig. 7(f) illustrates the resulting LRD as follows:

$$\text{LRD} = \text{PO;2 NOT ((AA:0 OR PO;0) SIZING } (s_1 + s_2 + s_3)). \quad (9)$$

Finally, the final poly dummy patterns, denoted as FPD, can be computed by combining SDR and LRD as shown in (10). Fig. 7(g) illustrates the final poly dummy patterns as follows:

$$\text{FPD} = \text{SDR OR LRD}. \quad (10)$$

Fig. 7(h) illustrates the final poly layer including both dummy and active patterns.


a
b
c
d
e
f
g
h
Short Range Type
Long Range Type
a. Active Pattern
b. AA;0 OR PO;0
c. PO;1 NOT ((AA;0 OR PO;0) SIZING s1)
d. PO;1 NOT ((AA;0 OR PO;0) SIZING (s1+s2))
e. (PO;1 NOT ((AA;0 OR PO;0) SIZING s1)) NOT (PO;1 NOT ((AA;0 OR PO;0) SIZING (s1+s2))
f. PO;2 NOT ((AA;0 OR PO;0) SIZING (s1+s2+s3))
g. ((PO;1 NOT ((AA;0 OR PO;0) SIZING s1)) NOT (PO;1 NOT ((AA;0 OR PO;0) SIZING (s1+s2))) OR (PO;2 NOT ((AA;0 OR PO;0) SIZING (s1+s2+s3))
h. ((PO;1 NOT ((AA;0 OR PO;0) SIZING s1)) NOT (PO;1 NOT ((AA;0 OR PO;0) SIZING (s1+s2))) OR (PO;2 NOT ((AA;0 OR PO;0) SIZING (s1+s2+s3)) OR (AA;0 OR PO;0)
Fig. 7. Example of Boolean-operation dummy fill using both short-range and long-range dummy patterns.

Note that the above discussion focused primarily on the steps required for generating nonoverlapping dummy patterns. The Boolean operations used for jog removal were omitted in order to simplifying the illustration. In practice, the complete Boolean operations required for dummy fill with both short-range and long-range dummy patterns are actually more than those listed in the above equations and Fig. 7.

D. Dummy Cell Initial Definition for Optimizing Layout Density Uniformity

In the above discussion, Boolean dummy insertion has been described for the case where a single set of short-range and long-range dummy cells is utilized throughout the chip, following the standard practice for conventional dummy pattern insertion. It will be shown by our experimental results that even with this constraint, Boolean dummy insertion delivers improved cross-chip pattern density uniformity. However, because Boolean dummy insertion is performed by simple chip-level Boolean operations, further optimization of layout density uniformity can be achieved by the appropriate initial definition of different dummy cells for different regions of the chip based on an initial survey of the layout density of the tape-out GDS. Before beginning the Boolean dummy insertion flow, the tape-out GDS is surveyed and the layout density is calculated for each rectangular subsection of the chip as shown in Fig. 8, where the size of the subsection is determined by process technology requirements. Then, short-range (and, if necessary, long-range) dummy cells can be initially defined with layout densities determined by the initial layout density of each subsection in the tape-out GDS. These cells can either be defined to attempt to achieve a target layout density or may simply be adjusted to tighten the statistical distribution of the layout density. In either case, the easy application of Boolean dummy insertion, utilizing dummy cells based on the initial

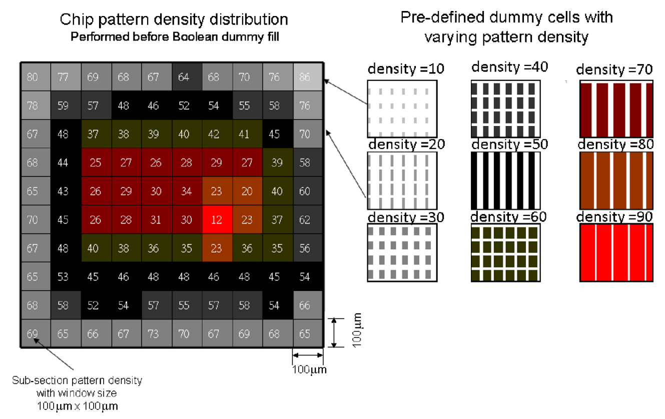

Chip pattern density distribution
Performed before Boolean dummy fill
Pre-defined dummy cells with varying pattern density
density =10
density =20
density =30
density =40
density =50
density =60
density =70
density =80
density =90
100μm
100μm
Sub-section pattern density with window size 100μm x 100μm

Fig. 8. Definition of dummy cells for different chip subregions based on initial layout density.

layout density, is certain to improve layout density uniformity even further than the improvements that are demonstrated by our experimental results. Of course the same concept can be applied to predefine dummy cells for conventional search-based dummy fill algorithms, but, as noted above, the improved computational efficiency of Boolean dummy fill enables this technique to be applied more easily.

IV. Experimental Results

A. *Conventional Dummy Fill and Test Cases*

In our experiments, we compare the proposed Boolean-operation dummy fill with a conventional dummy-fill flow, which has been applied to several product lines and is provided to customers by a well-known IC foundry. The concept of this conventional dummy fill is to insert the predefined dummy cells, which are also provided by the IC foundry, into the empty space of the design layout based on an insertion algorithm. Each predefined dummy cell has different dimensions as well as different insertion rules. Also, the predefined dummy cells can be designed with multiple-layer patterns and are sorted by their cell size. When empty space is found, the conventional dummy-fill algorithm will always try to insert the largest dummy cell possible according to the cell's insertion rules. Note that the final GDS-file size is directly proportional to the number of dummy cells inserted. Thus, by inserting the largest possible dummy cell, the GDS-file size can be minimized. The conventional dummy fill is implemented by a Calibre script, and has been extensively optimized under the constraints imposed by the need to maintain a reasonable file size and runtime. The test cases in our experiments are various representative products manufactured by the same IC foundry. The test cases include a 16 MB SRAM design, a mixed-signal design, and two logic designs developed with an advanced process technology. All the reported results were computed on a PC with a 2.8 GHz Opteron CPU (4 cores) and 64 GB main memory.

B. *Comparison of Runtime and GDS File Size*

Table I shows the results for applying the conventional dummy fill to each test case. In Table I, column 2 lists the

TABLE I
Resulting GDS File Size and Runtime for Conventional Dummy Fill

Test Case	Chip Size (μm^2)	GDS File Size (Byte)		Ratio b/a	Runtime (s)
		Before fill (a)	After fill (b)		
Memory	3800×4000	29 818 306	1 684 161 736	56.5x	11 619
Mix_signal	2200×660	191 690 752	2 526 404 608	13.2x	3653
Logic1	2200×6240	39 909 376	930 647 608	23.3x	18 003
Logic2	3300×3300	360 988 072	8 359 665 856	23.2x	4207
Average				29.1x	9371

chip size, columns 3 and 4 list the GDS-file size before and after applying the conventional dummy fill, respectively, and column 5 lists the ratio of these two file sizes. Column 6 lists the runtime for applying conventional dummy fill. As the results show, the GDS-file size may increase dramatically (29.1X in average) after the conventional dummy fill. This ratio ranges from 13.2X to 56.5X for different test cases and depends on the layout density of the original design, which in turn affects the number of dummy cells to be inserted. File sizes for the largest test case before and after the conventional dummy fill are 360 MB and 8.4 GB, respectively. The runtime of the conventional dummy fill is determined by the chip size and the number of inserted dummy cells. The longest runtime of the examples presented here was slightly more than 5 h.

Note that the test cases used in this experiment are substantially smaller than typical present-day advanced designs. However, the resulting GDS-file size already exceeds the memory limitation of an average PC. In fact, we attempted to run this experiment by reducing the main memory of the PC from 64 GB to 16 GB, but these experiments failed to run due to insufficient memory. This result demonstrates the large computational overhead of conventional dummy fill. Also, it can be expected that the GDS-file size increase by the conventional dummy fill may become even worse in the near future when the need of short-range patterns further increases for more advanced processes and thus a larger number of smaller dummy cells must be inserted. This experimental result further demonstrates the advantage of the proposed Boolean-operation dummy fill and eliminates the need for the tape-out GDS to contain dummy-fill patterns.

Next, Table II shows the runtime of performing the proposed Boolean-operation dummy fill. In Table II, column 2 lists the runtime of performing the Boolean operations only for computing original mask layers (without dummy fill). Column 3 lists the extra runtime of performing the Boolean operations for generating the mask layers including dummy fill, i.e., steps 1–4, and step 6 in Fig. 3. Column 4 lists the runtime for applying the small-island removal Calibre script (step 5 in Fig. 3). Although in the implementation reported in this paper, for convenience of implementation, the small-island removal step was implemented with a separate, readily available Calibre DRC script, which was different from the script used for the remainder of the steps in Fig. 3, combining all operations in Fig. 3 into a single Calibre script that is a straightforward programming exercise. Because much of the processing time in column 4 is due to the disruption of

TABLE II
RUNTIME OF THE PROPOSED BOOLEAN-OPERATION DUMMY FILL

Test Case	(a) Original Mask Layers Operation	Proposed Flow			(%) Overhead (d/a)
		(b) Dummy Fill Operations Steps 1–4 and 6	(c) Small-Island Removal	(d) Total = b + c	
Memory	8265	275	68	343	4.2
Mix_signal	825	95	59	154	18.7
Logic1	4965	180	540	720	14.5
Logic2	945	450	113	563	59.6
Average	3750	250	195	445	24.3

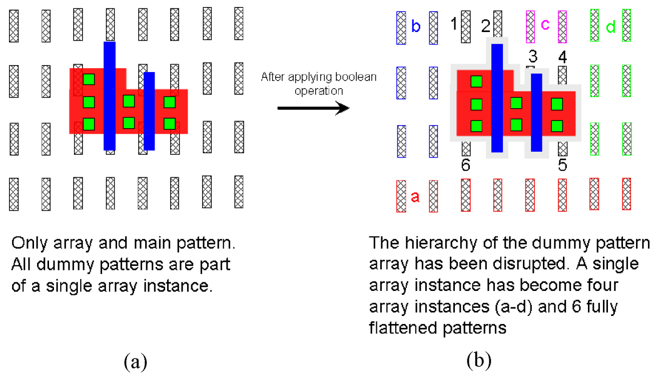

b
1
2
c
d
3
4
6
5
a
After applying boolean operation
Only array and main pattern. All dummy patterns are part of a single array instance.
The hierarchy of the dummy pattern array has been disrupted. A single array instance has become four array instances (a-d) and 6 fully flattened patterns
(a)
(b)

Fig. 9. Illustration of disruption of the GDS hierarchy of a dummy pattern array after applying one Boolean operation.

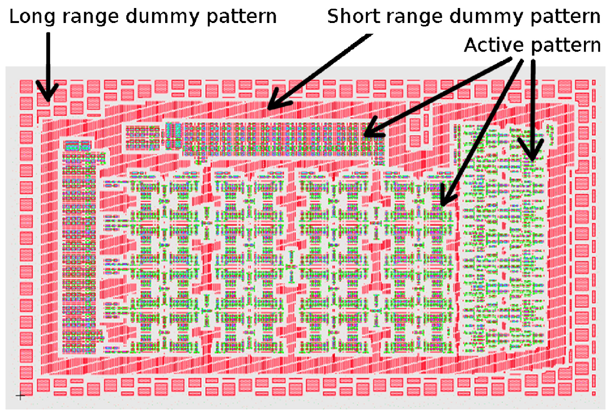


Fig. 10. Portion of the layout of the test case Logic1 after performing the Boolean-operation dummy fill.

the GDS hierarchy for input into the DRC script, a single integrated script would have much lower execution time for this operation. Fig. 9 illustrates the disruption of the GDS hierarchy of an array off dummy patterns after applying one Boolean operation.

The overall runtime overhead of the proposed dummy fill is listed in column 5, which is the summation of columns 3 and 4. The average runtime overhead of the proposed dummy fill is 445 s, which is 1/21 of that of the conventional flow (9371 s). This result demonstrates the efficiency of performing dummy fill based on Boolean mask operations. In this case, the same computing platform was used for both conventional dummy fill and the proposed dummy fill. In practice, the computers typically used for mask set computation and OPC are much more powerful than the workstations generally used in design houses. Thus, the actual runtime overhead of the proposed dummy fill would be even shorter if performed on a computing platform typically utilized for OPC. In addition, the additional runtime overhead of the proposed dummy fill, on average, is around 1/4 of that of computing the original mask layers, which is also a small portion of the entire process of mask-set generation when the long runtime of OPC is considered.

The different amounts of overhead for these respective designs can be qualitatively explained by the percentage of die area devoted to memory, logic, and mixed-signal intellectual property (IP), respectively. For the "Memory" test case, most of the die area consisted of high-density memory, so relatively few dummy patterns needed to be added. As a result, the nonoverlapping dummy pattern area (generated by step 3 of Fig. 3) is quite small, and the remaining steps in the flow consume relatively little computation time. On the other extreme, for the test case "Logic 2," a relatively large portion of the chip area consisted of random logic with a layout style that was not particularly dense. As a result, a relatively large number of dummy patterns were added. Note, however, that the small-island removal time for this case is comparable to the memory case, because the low density layout style required less use of small dummy cells, and thus less small-island removal. The test case "Mix_signal" had a relatively high percentage of mixed-signal and/or radio frequency IP, and for this IP, the dummy patterns had been drawn in the design GDS to optimize matching and to meet electrical requirements for some of the dummy patterns (e.g., requirements that some dummy patterns be grounded and others be connected to power planes). Thus, similar to the "Memory" test case, there was a relatively small area that required dummy fill, resulting in a small nonoverlapping dummy pattern area and a low small-island removal time. Test case "Logic 1" utilized a layout style of much higher density than test case "Logic 2," so while the amount of dummy pattern area required was not as large as "Logic 1," a larger number of small dummy patterns were required, resulting in a larger overhead for small-island removal. Only in the case of "Logic 2" was the overhead more than 50% of the computation time required for mask layer generation.

Table III shows the file sizes obtained by applying the proposed dummy fill to each test case. Following the convention of Table I, in Table III, column 2 lists the chip size and columns 3 and 4 list the GDS-file size before and after applying the proposed dummy fill. The ratio of the file size after dummy fill to before dummy fill is below 5X, which is a relatively small rate of file size increase compared with

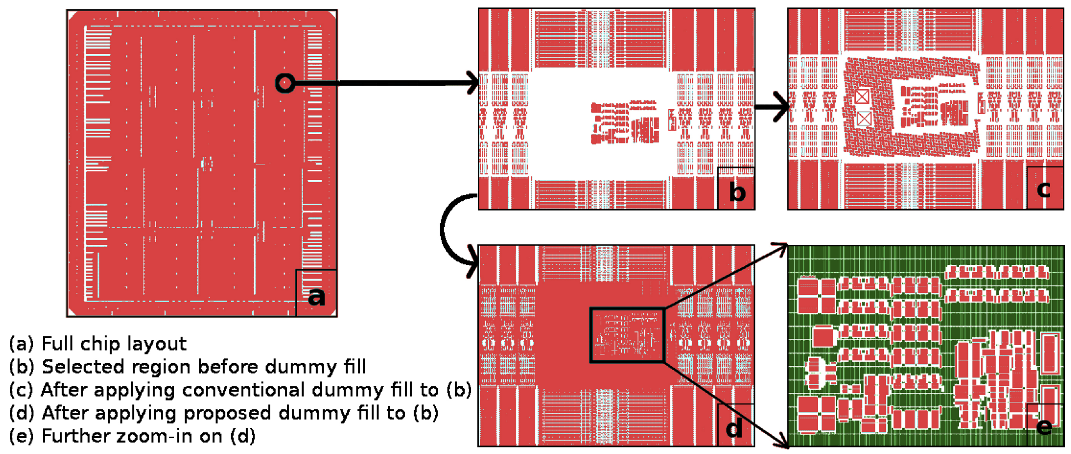

a
b
c
d
e
(a) Full chip layout
(b) Selected region before dummy fill
(c) After applying conventional dummy fill to (b)
(d) After applying proposed dummy fill to (b)
(e) Further zoom-in on (d)

Fig. 11. Layout of the test case "Memory" after applying conventional dummy fill and Boolean-operation dummy fill, respectively.

TABLE III
RESULTING GDS FILE SIZE AND RUNTIME FOR THE PROPOSED BOOLEAN-OPERATION DUMMY FILL

Test Case	Chip Size (μm^2)	GDS File Size (Byte)		Ratio
		Before Fill (a)	After Fill (b)	b/a
Memory	3800 × 4000	29 818 306	145 272 352	4.84x
Mix_signal	2200 × 660	191 690 752	876 345 355	4.57x
Logic1	2200 × 6240	39 909 376	91 074 334	2.28x
Logic2	3300 × 3300	360 988 072	424 575 146	1.18x
Average				3.22x

conventional dummy fill. This small rate of increase can be obtained because the GDSII file after the proposed dummy fill still maintains a certain degree of hierarchy, resulting in much less file size increase compared to conventional dummy fill that does not take advantage of database hierarchy.

Due to increasing design complexity, in recent years, an increasing portion of the design community is adopting OASIS file format rather than the traditional GDSII. Typically, transitioning from GDSII to OASIS reduces database file size by a factor of roughly 7–10X, where the exact value is a function of the degree of cell hierarchy. Based on experience in advanced technology tape-outs, which submit databases in OASIS format, it has been observed that the size ratio between GDSII and OASIS remains nearly constant before and after dummy fill, i.e., approximately 7–10X. The benefits we describe in this paper exist no matter which of these two file formats is used. However, the precise size ratio between the two formats for a given design depends on the degree of cell hierarchy employed.

C. Chip Layout After Dummy Fill

Fig. 10 shows the layout of a subblock in the test case "Logic1" after the Boolean-operation dummy fill is applied, where both the short-range and long-range dummy patterns can be clearly observed. Fig. 11(a) and (b) shows the layout of the full chip and the layout of a selected region for the SRAM test case, respectively. Fig. 11(c) and (d) shows the layouts of the selected region after applying the conventional dummy

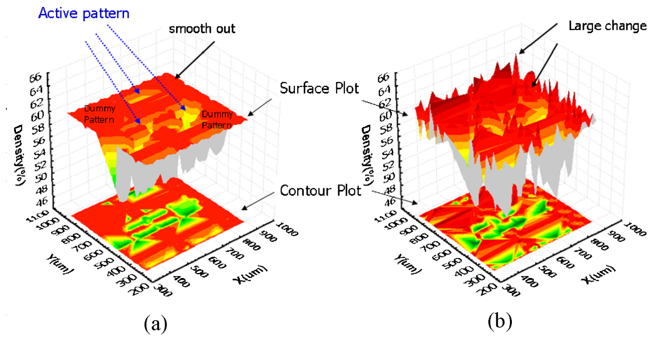


Fig. 12. Surface and contour plot of pattern density after performing the (a) proposed Boolean and (b) traditional dummy fill.

fill and the Boolean-operation dummy fill, respectively. The uniformity of the dummy patterns in Fig. 11(d) is higher than that in Fig. 11(c). This is because the dummy patterns are inserted based on the unit of a dummy cell in the conventional flow and the empty space remaining in the layout may not be large enough to place the smallest dummy cell. This problem can be solved by providing smaller predefined dummy cells. However, this would almost certainly increase the runtime and GDS file-size of the conventional flow because larger numbers of smaller dummy cells would need to be inserted. On the other hand, the dummy patterns in the proposed flow are generated by eliminating the active-pattern region from a full-layer dummy pattern, so empty space is eliminated more effectively than in the case of the conventional flow. This result demonstrates the high pattern uniformity achieved by the proposed Boolean-operation dummy fill.

D. Pattern Density After Dummy Fill

The pattern density of a real product chip after performing dummy fill was extracted by the Mentor Calibre tool with a window size of 20 × 20 μm. Fig. 12 compares the surface and contour plots for the pattern density of the AA layer after performing both traditional and Boolean dummy fill. As shown, there are three major areas where active patterns were originally present in the design. In the case of traditional dummy fill, many large changes in layout density occur in the immediate neighborhood of the active patterns. These density changes are caused by insufficient open area for insertion of predefined dummy cells. However, these changes in pattern density can be significantly reduced by the proposed Boolean dummy fill methodology, as shown in left plot in same figure. This improvement is primarily due to the Boolean "NOT" Boolean following the predefined dummy cell spreading. Dummy insertion is performed by Boolean dummy fill as long as the open area is larger than S1 and the dummy features pass small-island removal as described in Section III-B. In other words, the proposed Boolean dummy-fill algorithm can improve the chip pattern density uniformity, especially on areas very close to active patterns, which is very difficult to achieve using traditional dummy-insertion algorithms. It is possible that comparable pattern density uniformity could be achieved by the traditional algorithm by a "brute force" approach of adding additional smaller predefined dummy cells, but such an approach would greatly increase the algorithm runtime as well as the file size after dummy insertion. We further note that in this example, pattern density uniformity was improved even when the same set of short and long-range dummy cells was utilized for the entire chip. Further improvement is clearly possible based on an appropriate initial definition of different dummy cells for different subregions of the chip where the density of the dummy cells is derived from the initial layout density, as described previously.

V. Conclusion

In this paper, we have presented a novel dummy-fill flow, which applies Boolean mask operations to directly generate the mask layers with dummy patterns inserted, and performs dummy generation and mask computation at the foundry simultaneously with dummy fill and post-dummy simulation at the design house. The proposed flow not only improves the efficiency of dummy generation but also eliminates the delays in tape-out due to dummy insertion and post-dummy simulation, and dramatically reduces the tape-out GDS file size, enabling more rapid first silicon delivery. In comparison with a conventional dummy-fill flow currently widely used throughout the IC industry, the proposed dummy-fill flow can achieve better pattern uniformity with less runtime and smaller GDS file size. The savings in runtime and GDS file size will be even more significant in future advanced process technologies as more short-range dummy patterns with smaller pattern dimensions are required.

References

[1] J. M. Cohn, D. J. Garrod, R. A. Rutenbar, and L. R. Carley, *Analog Device-Level Layout Automation*. Norwell, MA: Kluwer Academic, 1994.

[2] F. Balasa, S. C. Maruvada, and K. Krishnamoorthy, "On the exploration of the solution space in analog placement with symmetry constraints," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 2, pp. 177–191, Feb. 2004.

[3] H.-Y. Chen, M.-F. Chiang, and Y.-W. Chang, "Novel full-chip gridless routing considering double-via insertion," in *Proc. IEEE Des. Automat. Conf.*, Sep. 2006, pp. 755–760.

[4] K.-Y. Lee, T.-C. Wang, and K.-Y. Chao, "Post-routing redundant via insertion and line end extension with via density consideration," in *Proc. IEEE Int. Conf. Comput.-Aided Des.*, Nov. 2006, pp. 633–640.

[5] O. Rizzo and H. Melzner, "Concurrent wire spreading, widening, and filling," in *Proc. IEEE Des. Automat. Conf.*, Jun. 2007, pp. 350–353.

[6] M.-C. Tsai, Y.-C. Lin, and T.-C. Wang, "An MILP-based wire spreading algorithm for PSM-aware layout modification," in *Proc. IEEE Asia South Pacific Des. Automat. Conf.*, Mar. 2008, pp. 364–369.

[7] V. Kheterpal, V. Rovner, T. G. Hersan, D. Motiani, Y. Takegawa, A. J. Strojwas, and L. Pileggi "Design methodology for IC manufacturability based on regular logic-bricks," in *Proc. Annu. ACM IEEE Des. Automat. Conf.*, Jun. 2005, pp. 353–358.

[8] A. Kurokawa, T. Kanamoto, A. Kasebe, Y. Inoue, and H. Masuda, "Efficient capacitance extraction method for interconnects with dummy fills," in *Proc. IEEE Custom Integr. Circuit Conf.*, Oct. 2004, pp. 485–488.

[9] A. B. Kahng and K. Samadi, "CMP fill synthesis: A survey of recent studies," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 1, pp. 3–19, Jan. 2008.

[10] C.-J. Hung, "Diamond metal-filled patterns achieving low parasitic coupling capacitance," U.S. Patent 6 998 716, Feb. 14, 2006.

[11] M. M. Nelson, "Optimized pattern fill process for improved CMP uniformity and interconnect capacitance," in *Proc. Univ./Govern./Industr. Microelectron. Symp.*, 2003, pp. 374–375.

[12] R. Tian, D. F. Wong, and R. Boone, "Model-based dummy feature placement for oxide chemical-mechanical polishing manufacturability," in *Proc. IEEE Des. Automat. Conf.*, Jun. 2000, pp. 667–670.

[13] Y. Chen, A. B. Kahng, G. Robins, and A. Zelikovsky, "Practical iterated fill synthesis for CMP uniformity," in *Proc. IEEE Des. Automat. Conf.*, Jun. 2000, pp. 671–674.

[14] M. Mukherjee and K. Chakraborty, "A randomized greedy algorithm for the pattern fill problem for DFM applications," in *Proc. Int. Symp. Qual. Electron. Des.*, Mar. 2008, pp. 344–347.

[15] C. Chiang and J. Kawa, *Design for Manufacturability and Yield for Nano-Scale CMOS*. New York: Springer, 2007.

[16] J. Luo, Q. Su, C. Chiang, and J. Kawa, "A layout dependent full-chip copper electroplating topography model," in *Proc. IEEE Int. Conf. Comput.-Aided Des.*, Nov. 2005, pp. 133–140.

[17] S. Sinha, J. Luo, and C. Chiang, "Model based layout pattern dependent metal filling algorithm for improved chip surface uniformity in the copper process," in *Proc. IEEE Asia South Pacific Des. Automat. Conf.*, Jan. 2007, pp. 1–6.

[18] R. Li, L. Yu, H. Xin, Y. Dong, K. Tao, and C. Wang, "A comprehensive study of reducing the STI mechanical stress effect on channel-width-dependent Idsat, *Semicond. Sci. Technol.*, vol. 22, no. 12, pp. 1292–1297, 2007.

[19] O. Pohland, J. Spieker, C.-T. Huang, S. Govindaswamy, and A. Balasinski, "New type of dummy layout pattern to control ILD etch rate," *Proc. SPIE*, vol. 6798, p. 679804, Dec. 2007.

[20] S. Nakao, K. Tsujita, I. Arimoto, and W. Wakamiya, "0.32-um pitch random line pattern formation by dense dummy pattern and double exposure in KrF wavelength," *Proc. SPIE*, vol. 4000, pp. 1123–1133, Jul. 2000.

[21] B. A. Anderson, H. S. Landis, and E. J. Nowak, "Variable overlap of dummy shapes for improved rapid thermal anneal uniformity," U.S. Patent 7 537 941 B2, May 26, 2009.

[22] L. Remy, P. Coll, and F. Picot, "Metal filling impact on standard cells: Definition of the metal fill corner concept," in *Proc. 21st Annu. Symp. Integr. Circuits Syst. Des.*, Sep. 2008, pp. 16–21.

[23] Y. Uematsu, "Integrated circuit device having dummy pattern effective against micro loading effect," U.S. Patent 5 598 010, Jan. 28, 1997.

[24] J. A. Wilmore, "Efficient boolean operations on IC masks," in *Proc. IEEE Des. Automat. Conf.*, Jun. 1981, pp. 571–579.

[25] U. Lauther, "An O(N log N) algorithm for boolean mask operations," in *Proc. IEEE Des. Automat. Conf.*, Jun. 1981, pp. 555–562.

[26] Y.-M. Kim, S.-U. Lee, J.-H. Kang, J.-H. Kim, and K.-H. Kim, "Application of modified jog-fill DRC rule on LFD OPC flow," *Proc. SPIE Photomask Technol. Conf.*, vol. 6730, no. 3, p. 67303W, 2007.

Tseng-Chin Luo received the M.S. degree in material science and engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1994, where he is currently pursuing the Ph.D. degree in electrical engineering.

He joined Winbond Electronics, Hsinchu, in 1996, and then was transferred to Worldwide Semiconductor Manufacturing Corporation, Hsinchu, in 1997. His major focus was parametric testing and process integration during this period. Since 1998, he has been developing processes for 0.18 μm and 0.13 μm technology in the Logic Technology Research and Development Division, Taiwan Semiconductor Manufacturing Corporation (TSMC), Hsinchu. He is currently a Project Manager with Fast Parametric Test Solutions, TSMC. His current research interests include developing test structures and fast test methodology for process characterization, design manufacturing and yield optimization, and establishing infrastructure to improve design effectiveness and analysis quality.

Mango C.-T. Chao received the B.S. and M.S. degrees from the Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, in 1998 and 2000, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of California at Santa Barbara, Santa Barbara, in 2006.

He then joined the Department of Electronics Engineering, National Chiao Tung University, where he is currently an Assistant Professor. His current research interests include memory testing, on-chip test compression/decompression, wafer acceptance test test-structure design, power-related testing, and physical design automation.

Philip A. Fisher received the B.S. degree in physics from the University of Washington, Seattle, in 1991, and the M.A. and Ph.D. degrees in physics from Harvard University, Cambridge, MA, in 1994 and 1999, respectively.

From 1999 to 2000, he was employed as a Foundry Process Support Engineer with Analog Devices Corporation, Norwood, MA. From 2000 to 2008, he was employed in the Process Integration and Advanced SOI Transistor Development with Advanced Micro Devices Corporation, Sunnyvale, CA, during which time he was promoted from a Senior Integration Engineer to a Senior Member of Technical Staff in recognition of his contributions to the transistor development for the 130, 90, 65, 45, and 32-nm technologies. Since 2008, he has been a Manager with the Department of Advanced Device Technology, Taiwan Semiconductor Manufacturing Corporation, Hsinchu, Taiwan. His Ph.D. research focused on the development of a thermoelectric microrefrigerator based on superconducting tunnel junctions, which was also reviewed in *The Economist*, Mar. 21, 1998. He is the author or co-author of 27 peer-reviewed publications and 27 U.S. patents. His current research interests include development and characterization of high-performance and low-power transistors.

Chun-Ren Kuo was born in Kaohsiung, Taiwan, in 1986. He received the B.S. degree from National Cheng Kung University, Tainan, Taiwan, in 2008, and the M.S. degree in electrical engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2011. His M.S. research focused on diagnosis and design for testability, which investigated scan-reordering methods based on low-power fill on stuck-at fault patterns to improve bridge fault coverage.