Mask Optimization for Directed Self-Assembly Lithography: Inverse DSA and Inverse Lithography

Seongbo Shim and Youngsoo Shin
School of Electrical Engineering, KAIST
Daejeon 34141, Korea

Abstract—**In directed self-assembly lithography (DSAL), a mask contains the images of guide patterns (GPs), which are patterned on a wafer through optical lithography; the wafer then goes through DSA process to pattern contacts. Mask design for DSAL, which is the opposite of the above processes, consists of two key steps, inverse DSA and inverse lithography, which we address in this paper. In inverse DSA, we progressively refine GPs until they produce target contacts as closely as possible. GP is defined as a function of a few geometry parameters, and how sensitive the contacts are to the parameters are calculated which then guides how much the GP should be refined. In inverse lithography, mask is progressively refined so that target GPs are produced. Mask is defined by pixel values and their gradient guides the direction that the mask should be refined. There are too many pixels for gradient calculation; the method to approximate calculation is proposed. Inverse DSA and inverse lithography are extended to handle process variations. We modify basic inverse lithography so that the resulting mask becomes less sensitive to lithography variations; basic inverse DSA is modified so that it provides the way this sensitivity can be checked.**

I. Introduction

A device of sub 10-nm node requires contacts and vias smaller than 40nm, which is beyond the resolution limit of traditional optical lithography. A directed self-assembly lithography (DSAL) has been proposed as an alternative patterning solution and is being widely studied for industrial deployment. DSA process is illustrated in Fig. 1: a large hole called a guide pattern is created (through optical lithography); the hole is filled with block copolymers (BCPs), which are strings of hydrophilic and hydrophobic polymers, and the two polymers are self-arranged (once heated) due to forces between polymers and guide pattern; contacts (or vias) are formed after one type of polymer is etched away.

Fig. 2 compares how mask is designed in optical lithography and DSAL. In optical lithography shown in (a), a lithography image (litho image for short) is obtained for a given layout; it then goes through optical proximity correction (OPC) to yield a final mask image. Mask design in DSAL is more complex as illustrated in (b). A DSA image is extracted from a layout; note that DSA image is a result of DSA process (see Fig. 1) while litho image results from lithography process, which is why they are named differently even though the same process p1 is applied to extract them. Nearby contacts in DSA image are clustered [1] and a guide pattern image (GP image) is compiled for each cluster. Only the contours in GP image are taken (now called GP litho image), from which the final mask image is designed.

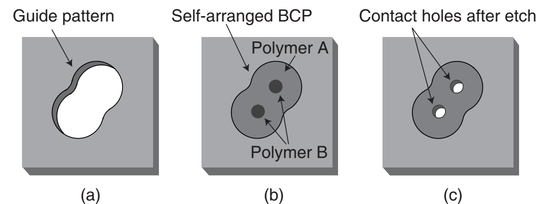

Fig. 1. DSA process: (a) a guide pattern, (b) a guide pattern filled with BCPs which are then self arranged, and (c) contact holes after polymer B is etched away.

A. Contribution

Inverse DSA: A key process in Fig. 2(b) is inverse DSA, labeled p3. There have been a few studies for the opposite process, e.g. form various shapes of GP images on a wafer and check how contacts are formed for each GP [2], and verify whether a given GP image causes any unexpected contact shapes [3]. But inverse DSA has never been studied before. An analytical solution to inverse DSA does not exist because a GP image is not uniquely determined for a DSA image. We propose an iterative solution (Section II), in which a GP image is progressively refined while the resulting DSA image, obtained through DSA simulation, is assessed against a target DSA image. A GP image is defined as a function of a few geometry parameters; it is refined by using sensitivity matrix, which contains the extent of how sensitive the DSA image is to each parameter change.

Inverse Lithography: Another important process in Fig. 2(b) is p2* named inverse lithography. This is similar to p2 of Fig. 2(a). A prime difference exists in how mask image is defined. In conventional OPC, a mask image is defined by a set of edges. Even in some DSAL techniques, edge-based mask image has been used to simplify mask design process [4]. But GP mask image should be more finely defined to reflect complex curve shapes that GP litho image contains. In our approach in Section III, we define mask image as a collection of pixels. A mask image is progressively refined using gradient descent method, and resulting litho image is assessed for each refinement. Gradient calculation is approximated without resorting to explicit lithography simulation, which speeds up the process.

We extend inverse DSA and inverse lithography to account for DSA- and lithography-variations (Section IV). Two problems are addressed. For a given error tolerance of DSA image, we want to derive the error tolerance of GP image, specifically

978-1-4673-9569-4/16/$31.00 ©2016 IEEE

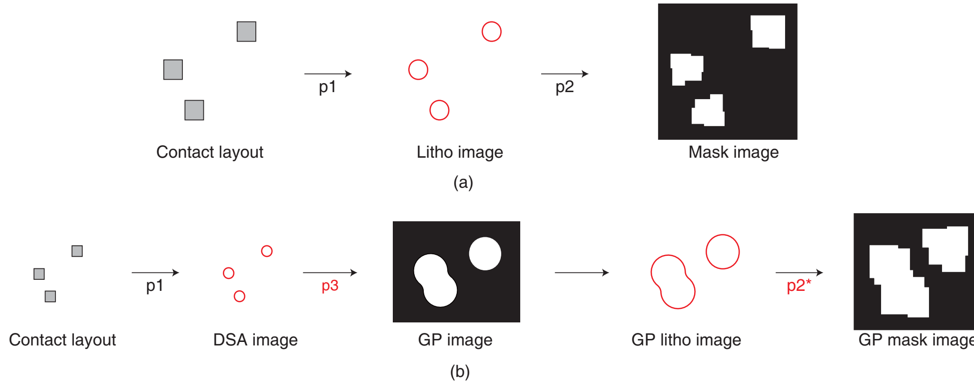

Fig. 2. Mask design for (a) optical lithography and (b) DSAL.

Input: a DSA image $\mathcal{D}_{in}$ of a cluster of contacts
Output: a GP image $\mathcal{G}$

L1: $\mathcal{G} \leftarrow$ an initial GP image
L2: $\mathcal{D} \leftarrow$ DSA_Simulation($\mathcal{G}$)
L3: $\mathbf{e} \leftarrow$ Measure_EPE($\mathcal{D}_{in}$, $\mathcal{D}$)
L4: **repeat** for max_iterations
L5: $\mathbf{M} \leftarrow$ Calc_Matrix($\mathcal{D}_{in}$, $\mathcal{G}$)
L6: $\mathcal{G} \leftarrow f(\mathbf{g})$, where $\mathbf{g}^{\mathrm{T}} \leftarrow \mathbf{g}^{\mathrm{T}} - \mathbf{M}^{-1} \times \mathbf{e}^{\mathrm{T}}$
L7: $\mathcal{D} \leftarrow$ DSA_Simulation($\mathcal{G}$)
L8: $\mathbf{e} \leftarrow$ Measure_EPE($\mathcal{D}_{in}$, $\mathcal{D}$)
L9: **if** $\max_{\forall i}(|e_i|) \leq$ max_EPE **then** Exit loop
L10: **return** $\mathcal{G}$

Fig. 3. Algorithm for inverse DSA.

the outer and inner boundaries within which the actual GP image has to reside; this is solved by applying inverse DSA multiple times. In the second problem, we aim to modify the basic inverse lithography of Section III, in a way that final mask yields GP image that always resides within the error tolerance of GP image even though lithography settings are varied.

II. Inverse DSA

The algorithm to solve inverse DSA (process p3 of Fig. 2(b)) is shown in Fig. 3. It receives one cluster of contacts in DSA image ($\mathcal{D}_{in}$), and returns corresponding GP image. An initial GP image is constructed (L1) by placing circles (each is a concentric circle for a circle in DSA image with radius being set to the length of one BCP string), taking only the boundary arc of merged circles, and smoothing the boundary (see Fig. 4). A DSA simulation is performed on the initial GP image (L2) to obtain DSA image ($\mathcal{D}$), which is, in most cases, different from input DSA image. The edge placement errors (EPEs) between the two DSA images are measured at certain points (four for each circle), and they are arranged as a vector $\mathbf{e} = (e_1, e_2, \ldots, e_m)$ (L3; see Fig. 4).

Our goal is to refine the GP image so that resulting DSA image resembles the input DSA image as much as possible; this is achieved through iteration (L4 to L9). To guide the refinement, GP image is defined as a function of geometry parameters, i.e.

$$\mathcal{G} = f(\mathbf{g}) = f(g_1, g_2, g_3, \cdots, g_n). \tag{1}$$

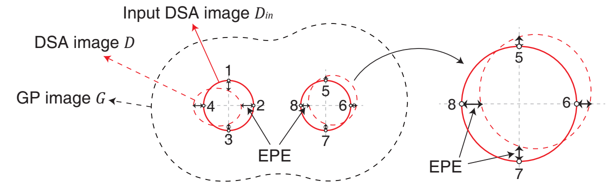

Fig. 4. GP image ($\mathcal{G}$) and its DSA image ($\mathcal{D}$); each contact of input DSA image has 4 EPEs at certain measurement points.

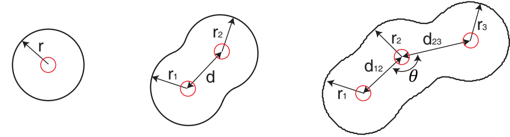

Fig. 5. Geometry parameters of GP image.

The parameters include the radius of each constituent circle, the center-to-center distance between adjacent circles, and the angle between adjacent center-to-center lines (see Fig. 5). We perturb each parameter to obtain a new GP image, perform a DSA simulation that returns corresponding DSA image, and assess its EPEs; a set of EPE changes allows us to introduce the sensitivity matrix:

$$\mathbf{M} = \begin{bmatrix} \frac{\partial e_1}{\partial g_1} & \frac{\partial e_1}{\partial g_2} & \cdots & \frac{\partial e_1}{\partial g_n} \\ \frac{\partial e_2}{\partial g_1} & \frac{\partial e_2}{\partial g_2} & \cdots & \frac{\partial e_2}{\partial g_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_m}{\partial g_1} & \frac{\partial e_m}{\partial g_2} & \cdots & \frac{\partial e_m}{\partial g_n} \end{bmatrix}. \tag{2}$$

It can be shown that the following holds:

$$\mathbf{M} \times \Delta\mathbf{g}^{\mathrm{T}} = \Delta\mathbf{e}^{\mathrm{T}}, \tag{3}$$

where $\Delta\mathbf{e}$ is EPE variations due to some extents of the parameter changes ($\Delta\mathbf{g}$). Rearranging (3) yields

$$\Delta\mathbf{g}^{\mathrm{T}} = \mathbf{M}^{-1} \times \mathbf{e}^{\mathrm{T}}, \tag{4}$$

where $\Delta\mathbf{g}$ now indicates how much we should adjust each parameter to compensate current EPEs (L6) to obtain a new GP image[1]. DSA simulation and EPE measurement then

[1] If $\mathbf{M}$ is not square, $\mathbf{g}^{\mathrm{T}} - (\mathbf{M}^{\mathrm{T}} \times \mathbf{M})^{-1} \times \mathbf{M}^{\mathrm{T}} \times \mathbf{e}^{\mathrm{T}}$ replaces $\mathbf{g}^{\mathrm{T}}$ or singular value decomposition (SVD) is applied in L6.

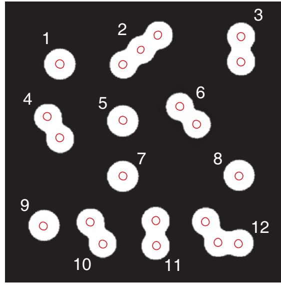

Fig. 6. GP images of clustered DSA images.

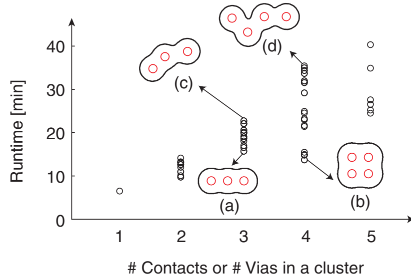

Fig. 7. Runtime of inverse DSA for various contact/via clusters.

follow (L7–L8). If the largest EPE does not exceed a certain threshold ($\max_{\forall i}(|e_i|) \leq$ max_EPE), iteration stops; otherwise iteration continues until user defined maximum iterations are reached.

The algorithm in Fig. 3 synthesizes a GP image for one cluster, so it has to be applied to all clusters (of contacts and vias) in a layout. Fortunately, DSA process illustrated in Fig. 1 is localized within a GP image[2], and so the same DSA image of a cluster should correspond to the same GP image. As an example of Fig. 6, a GP image is synthesized and shared for clusters of {1, 5, 7, 8, 9}; the same applies to {3, 4, 6, 10, 11}, {2}, and {12}; note that {3, 4, 6, 10, 11} are not exactly the same but they share the same geometry parameters. In our test layout that contains 804 clusters, only 31 (or about 4%) required inverse DSA.

A. Numerical Results

Our inverse DSA was implemented in MATLAB and C++; DSA simulator is based on self consistent field theory [6]. A few test circuits from Opencores [7] were synthesized by using 15nm NanGate library [8], and layouts in contact and via layers were scaled down so that the layouts follow the design rules of 10nm technology. In contact, size is 22nm and minimum pitch is 45nm; via has a size of 25nm with minimum pitch of 50nm.

The number of geometry parameters determines the number of DSA simulations to calculate the sensitivity matrix (2), and a DSA simulation takes more time as the area of GP image increases. Fig. 7 illustrates the runtime of inverse DSA for various types of clusters. For the same number of contacts (or vias) in a cluster, the runtime varies substantially. Clusters (a) and (b) have smaller number of parameters due to existence of symmetry, while (c) and (d) have more parameters and they occupy more area.

[2]It has been reported that DSA process is affected by GP density due to under-filling or over-filling of GP [1]. But, this can be overcome by additional process steps, e.g. DSA planarization [5].

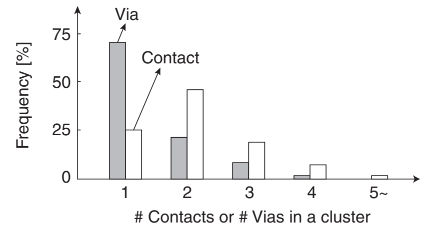

Fig. 8. Frequency of clusters with different number of contacts/vias.

Fig. 8 illustrates how the clusters with different number of contacts and vias are distributed for one sample circuit. Generally, clusters have smaller number of vias (shaded bars) because recent physical design tends to avoid routing congestion thereby spreading out vias. In contact layer (white bars), contrarily, there are many large clusters, which contain two or more contacts. Distribution of contact clusters is more affected by how cells are designed rather than by how cells are placed. This is partly because many cells have whitespace in-between. Note that a few very large clusters containing more than 5 contacts occur at cell boundary of abutted cells, and the number of them may increase as placement density increases.

III. Inverse Lithography

The input to inverse lithography (process p2* in Fig. 2(b)) is a GP litho image, which is simply a GP contour[3]. In conventional OPC of process p2, mask image is constructed by repeatedly adjusting edges. In our approach, on the other hand, GP mask image is obtained by adjusting pixels.

The algorithm of inverse lithography is illustrated in Fig. 9. An initial GP mask image is constructed (L1) by digitizing GP litho image, i.e. if pixel is within the litho image, its value is set to 1; otherwise its value becomes 0. The value indicates whether the corresponding pixel is transparent to light (1) or is opaque (0). A lithography simulation is performed on the initial mask image to obtain litho image (L2).

The cost is defined by the sum of EPEs between current litho image ($\mathcal{L}$) and input litho image ($\mathcal{L}_{in}$). We define a few points on input litho image to measure EPEs, as shown in Fig. 10. A line is drawn such that it passes the centers of adjacent contacts; orthogonal lines are also drawn from the centers of contacts. EPEs are measured at the points where the lines and input litho image intersect (points at peak); another points are also located at concave (points at valley); some more points are identified (intermediate points) between point at peak and point at valley in regular interval.

The gradient of cost (∇C) is defined by

$$\nabla C = \left(\frac{\partial C}{\partial g_1}, \frac{\partial C}{\partial g_2}, \frac{\partial C}{\partial g_3}, \cdots, \frac{\partial C}{\partial g_n} \right), \tag{5}$$

[3]In fact, the input is the whole GP contours or some GP contours in case of parallel processing, but we will consider one GP contour for simplicity of explanation.

Input: a GP litho image $\mathcal{L}_{in}$
Output: a GP mask image $\mathcal{M}$

L1: $\mathcal{M} \leftarrow$ an initial GP mask image
L2: $\mathcal{L} \leftarrow$ Litho_Simulation($\mathcal{M}$)
L3: $C \leftarrow$ Cost($\mathcal{L}_{in}$, $\mathcal{L}$)
L4: **repeat** for max_iterations
L5: $\quad\mathcal{M} \leftarrow \mathcal{M} - k \cdot \nabla C$
L6: $\quad\mathcal{M} \leftarrow$ Convert $\mathcal{M}$ to a binary mask
L7: $\quad\mathcal{L} \leftarrow$ Litho_Simulation($\mathcal{M}$)
L8: $\quad C \leftarrow$ Cost($\mathcal{L}_{in}$, $\mathcal{L}$)
L9: $\quad$**if** C increases OR $|\nabla C| \leq \varepsilon$ **then**
L10: $\quad\quad$Roll back $\mathcal{M}$; exit loop
L11: **return** $\mathcal{M}$

Fig. 9. Algorithm for inverse lithography.

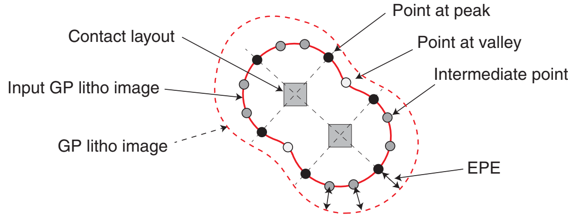

Fig. 10. EPE measurement points.

where parameters (g_i) are now pixel values rather than geometry properties; its calculation is addressed in Section III-A. A new mask image is constructed by adjusting each parameter of current mask image in the direction of negative gradient with step size k (L5), which is empirically determined. Some pixel values become non-binary in this process and so they are digitized (L6), i.e. if pixel value is larger than 1, it is set to 1; otherwise it is set to 0. We then perform lithography simulation to obtain litho image (L7), assess its cost through EPE calculation (L9). If cost increases or refined mask image yields insignificant change in litho image ($|\nabla C| \leq \varepsilon$), iteration stops with previously refined mask image; otherwise iteration continues until user defined maximum iterations are reached.

A. Approximation of Cost Gradient

An element of ∇C ($\partial C/\partial g_i$) can be obtained by perturbing one pixel value (g_i), performing lithography simulation, and assessing litho image through EPE calculation. This is computationally too expensive since there are many pixels; we instead approximate gradient calculation without explicit lithography simulation.

The light energy exposed on a wafer through mask image ($\mathcal{M}$), called intensity, determines litho image:

$$I(x,y) = |\mathcal{M}(x,y) \otimes H(x,y)|^2, \tag{6}$$

where H models optical element including light source and lenses, and $\otimes$ indicates a convolution. Let pixel value at (x_i, y_i) be perturbed; a new mask image can be represented by

$$\mathcal{M}'(x,y) = \mathcal{M}(x,y) + \delta(x - x_i, y - y_i), \tag{7}$$

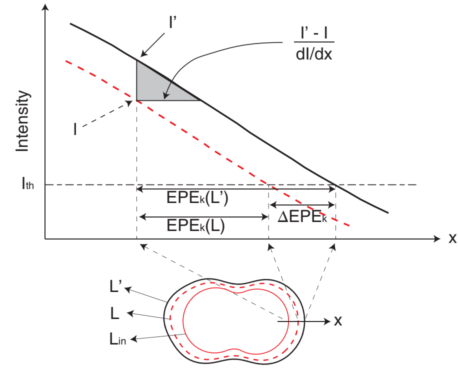

Fig. 11. Approximation of ΔEPE_k.

where δ is Dirac delta function. Corresponding intensity is

$$\begin{aligned} I' &= |\mathcal{M}' \otimes H|^2 \\ &= |(\mathcal{M} + \delta(x - x_i, y - y_i)) \otimes H|^2 \\ &= I + |H(x - x_i, y - y_i)|^2 + H^*(x - x_i, y - y_i)(\mathcal{M} \otimes H) \\ &\quad + H(x - x_i, y - y_i)(\mathcal{M} \otimes H)^*, \end{aligned} \tag{8}$$

where * denotes conjugate transpose.

We now try to approximate $\partial C/\partial g_i$ using (8). By definition of cost:

$$\begin{aligned} \partial C/\partial g_i &= \sum_k \left(\text{EPE}_k(\mathcal{L}', \mathcal{L}_{in}) - \text{EPE}_k(\mathcal{L}, \mathcal{L}_{in})\right) \\ &= \sum_k \Delta\text{EPE}_k, \end{aligned} \tag{9}$$

where $\mathcal{L}'$ is a potential new litho image if pixel value g_i is perturbed, and measurement point is indexed by k. Fig. 11 plots the two intensities I and I' with x-axis corresponding to the direction of EPE measurement. I_{th} denotes a threshold intensity, i.e. pattern is formed on a wafer unless intensity is smaller than I_{th}. We assume that I and I' are parallel lines, which is a approximation in small range of x. By referring to Fig. 11, it can be verified that EPE_k is equal to the bottom side of the shaded triangle, i.e.

$$\Delta\text{EPE}_k = \frac{I' - I}{dI(x_k, y_k)/dx}. \tag{10}$$

Note that $I' - I$ is readily obtained from (8); $\mathcal{M} \otimes H$ is available from lithography simulation (L2 or L7), and $H(x - x_i, y - y_i)$, which is in fact $H(x_k - x_i, y_k - y_i)$ in (10), is simply obtained from known H.

B. Evaluation

Our algorithm was implemented in MATLAB and C++, and applied to 6 test layouts (contact and via layers of 3 sample circuits). Since the layout is too large ($120\mu m$ by $120\mu m$ on average) to handle at the same time, it was divided into smaller tiles of $18\mu m$ by $18\mu m$ regions; the algorithm was then serially applied to each region and aggregated results yield a whole mask image; to handle optical proximity effect in the boundaries of the regions, we in fact took $20\mu m$ by $20\mu m$ region even though the result is taken from central $18\mu m$ by

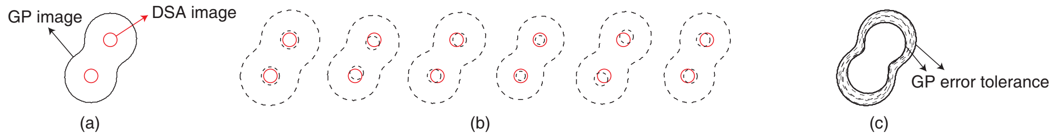

Fig. 12. Process to derive GP error tolerance: (a) DSA and GP images without error; (b) DSA and GP images (dotted) with various errors; (c) merged GP images from (b) and GP error tolerance (inner and outer boundaries).

TABLE I
COMPARISON OF EXACT AND APPROXIMATE INVERSE LITHOGRAPHY

Layout	Exact method			Approximate method		
	# Iter	Time (hours)	EPE_{max} (nm)	# Iter	Time (hours)	EPE_{max} (nm)
Via 1	5	1	0.3	11	0.1	0.9
Via 2	5	1.8	0.6	13	0.2	1.1
Via 3	6	4.2	0.6	14	0.5	0.7
Contact 1	7	4.8	0.5	18	0.4	0.8
Contact 2	7	6.8	0.5	17	1.1	0.7
Contact 3	10	16.7	0.7	25	2.9	1.4
Average	6.7	5.9	0.5	16.3	0.9	0.9

18μm region. ArF immersion lithography process was assumed with 1.35 numerical aperture (NA) and an annular shape of illumination.

We assess our algorithm (named approximate method) against the one that performs explicit lithography simulations (named exact method) for gradient calculations. The maximum number of iterations for all tiles, algorithm runtime (total runtime for all tiles using a single cpu), and maximum EPE value (out of all GPs in the test layout) of final litho image are compared in Table I. For the same circuit, via takes less time than contact, e.g. Via 1 vs Contact 1, because vias are usually sparser than contacts, for instance, one tile contains 269 GP contours on average in Contact 1, while Via 1 has just 48 GP contours within one tile. Approximate method requires more iterations due to inaccuracy of gradient computation implying that mask refinement is not always performed in ideal direction; runtime nevertheless is reduced by 85% on average thanks to approximation, which is not associated with any simulations. Note that runtime for a full-chip layout of 10mm by 10mm area is estimated to be about 10 hours when the industrial computing environment is assumed (e.g. distributed process with 500 cores), which is practically reasonable runtime [9]. Maximum EPE slightly increases, but is still within an acceptable range[4].

Contact requires more iterations because of worse accuracy in approximation. This is because contacts are usually denser than vias, so GP contours are closer to each other, which increases optical proximity effect and leads to decreased slope of intensity curve and thus smaller value of dI/dx (see Fig. 11); delta EPE value in (10) increases as a result.

IV. DSAL MASK DESIGN WITH PROCESS VARIATIONS

DSA and optical lithography processes are inherently associated with process variations. We address the mask design for DSAL (Fig. 2(b)) while we take such variations into account.

[4]About 1∼2% error in GP image incurs about 4∼5% error in DSA image. Patterning margin in DSA image is ±10%.

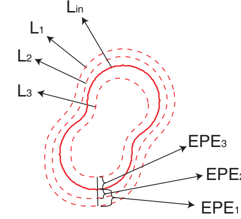

Fig. 13. EPE calculation in inverse lithography with lithography variations.

A. Inverse DSA and Inverse Lithography

Consider inverse DSA, which accepts DSA image and outputs GP image (Fig. 12(a)). DSA image is typically associated with error tolerance, e.g. ±10%. It is used to alter geometry parameters of DSA image such as circle size and distance between circles, which yields a series of altered DSA images (Fig. 12(b)); inverse DSAs are then applied to obtain GP images. All contours of GP images are overlapped (Fig. 12(c)); we define the inner and outer boundaries of aggregated contours as the error tolerance of GP image.

Consider now inverse lithography; we aim to modify the algorithm in Fig. 9 so that we take account of lithography variations:

- We insert assist features (AFs) while we synthesize an initial mask (L1), so that mask is less affected by lithography variations. AFs are also accounted for when we refine the mask (L5). The details of AF insertion are presented in Section IV-B.
- To calculate cost (L3 and L8), we perform lithography simulations multiple times while we vary lithography settings such as scanner focus, exposure energy, and mask error [10]. The simulations now yield multiple litho images ($\mathcal{L}_1$, $\mathcal{L}_2$, and $\mathcal{L}_3$ in Fig. 13); the maximum of all EPE values (at particular measurement point) is considered to be the final EPE value.

Once a final mask is returned, we also perform lithography simulations multiple times, and check whether all the resulting litho images reside within the error tolerance of GP image.

B. Insertion of DSA-Aware AF

We insert AFs in a way that the light that goes through AFs and regular patterns on a wafer interferes in constructive fashion. Light interference can be predicted by an interference map, which is a convolution of input mask image with $H(x,y)$ in (6). The regions of positive values indicate the potential locations where the light constructively interferes. We insert AFs into the region where the value is higher than some threshold.

If AF becomes too large during mask refinement process, it may be patterned out on a final wafer, which is not desirable.

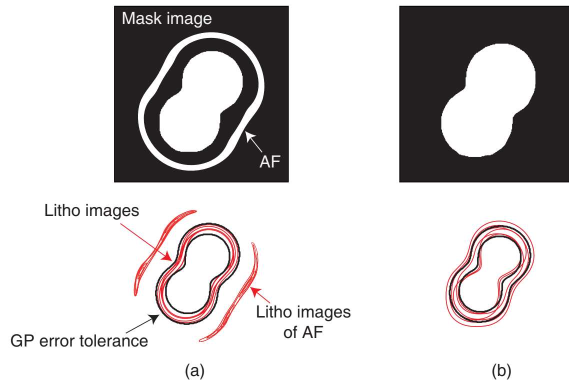

Fig. 14. Mask image (a) with AF and (b) without AF; litho images at various lithography settings are compared with GP error tolerance.

This can be prevented by restricting the AF size in a way that the width of its litho image does not exceed a certain value, which is determined by preliminary test.

Fig. 14(a) shows a GP mask image with curvilinear AF[5] and its litho images at various lithography settings, which are shown with the GP error tolerance. The litho images reside within the GP error tolerance, which implies that the variations of DSA image due to the process variations will not exceed the given error tolerance of the contact layout; note that AF also yields litho images, but which are not patterned out on a final wafer. But, some litho images from the GP mask image without AF reside outside of the GP error tolerance as shown in Fig. 14(b).

C. Assessment

The new inverse lithography algorithm was tested against the basic algorithm of Fig. 9, and the result is shown in Table II. The same test cases from Table I were used. Each algorithm was applied to generate a mask image, which was then submitted to lithography simulator as we vary lithography settings to 27 combinations (of scanner focus, exposure energy, and mask error). The resulting 27 litho images (or 27 GP contours) were checked to see whether they all reside within GP error tolerance; the corresponding GP is marked violated if checking fails. As Table I indicates, the new algorithm causes no violations while the basic algorithm incurs 6.6% on average.

The maximum distance between inner and outer contours of 27 litho images is called process variation band (PVB), which indicates the extent of sensitivity to process variations. The maximum PVB value of all GPs was picked and shown under columns ‘Max PVB’. Better result is also observed in this regard (on average of 4.1nm vs 9.0nm).

V. Conclusion

We have addressed two key problems of DSAL mask design. In inverse DSA, the GP has been progressively refined to produce desired contacts (or vias). The sensitivity matrix of contacts to a few geometry parameters of GP guides the refinement. In inverse lithography, mask has been refined for desired GP. The refinement has been guided by cost gradient with mask being defined by pixels. But, because of large number of pixels for gradient calculation, approximating gradient calculation without lithography simulation is important for runtime. Inverse DSA and inverse lithography have been extended to handle process variations. In particular, basic inverse lithography algorithm has been modified such that it produces a mask less sensitive to lithography variations.

DSA simulation takes two or three orders of more runtime than lithography simulation. Thus, the bottleneck in DSAL mask design process is inverse DSA, which we need to speed up as a future work.

[5]The curvilinear AF indeed is manufacturable by advanced mask writing technique [11].

TABLE II
Basic inverse lithography vs modified inverse lithography with lithography variations

Layout	Inverse litho		Inverse litho with litho variations	
	# Violations (%)	Max PVB (nm)	# Violations (%)	Max PVB (nm)
Via 1	7.9	8.3	0.0	3.6
Via 2	6.3	7.2	0.0	3.2
Via 3	6.8	8.8	0.0	3.3
Contact 1	5.3	9.4	0.0	4.4
Contact 2	8.8	9.8	0.0	4.7
Contact 3	4.3	10.5	0.0	5.5
Average	6.6	9.0	0.0	4.1

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2015R1A2A2A01008037)

References

[1] H. Yi *et al.*, “Design strategy of small topographical guiding templates for sub-15nm integrated circuits contact hole patterns using block copolymer directed self assembly,” in *Proc. SPIE Advanced Lithography*, Mar. 2013, pp. 1–9.

[2] ——, “Flexible control of block copolymer directed self-assembly using small, topographical templates: potential lithography solution for integrated circuit contact hole patterning,” *Advanced Materials*, vol. 14, no. 23, pp. 3107–3114, Jun. 2012.

[3] Z. Xiao *et al.*, “Directed self-assembly (DSA) template pattern verification,” in *Proc. Design Automation Conf.*, Jun. 2014, pp. 1–6.

[4] L. Azat *et al.*, “Computational simulations and parametric studies for directed self-assembly process development and solution of the inverse directed self-assembly problem,” *Japanese Journal of Applied Physics*, vol. 53, no. 6S, pp. 1–8, Jun. 2014.

[5] P. Barros *et al.*, “DSA planarization approach to solve pattern density issue,” in *Proc. SPIE Advanced Lithography*, Mar. 2015, pp. 1–10.

[6] N. Laachi *et al.*, “Self-consistent field theory investigation of directed self-assembly in cylindrical confinement,” *Journal of Polymer Science: Part B Polymer Physics*, vol. 53, no. 2, pp. 142–153, Jan. 2015.

[7] “Opencores,” http://www.opencores.org/.

[8] “Nangate 15nm open cell library,” http://www.nangate.com/.

[9] Private communication.

[10] L. Liebmann *et al.*, “Reducing DfM to practice: the lithography manufacturability assessor,” in *Proc. SPIE Advanced Lithography*, Feb. 2006, pp. 786–98.

[11] A. Fujimura *et al.*, “Improvement of mask write time for curvilinear assist feature,” in *Proc. SPIE Advanced Lithography*, Mar. 2010, pp. 1–10.