A Fast Look Up Table Based Lithography Simulator with SOCS Model for OPC Algorithm

TAHSIN BINTE SHAMEEM †1 ATSUSHI TAKAHASHI †1
HIROYOSHI TANABE †1 YUKIHIDE KOHIRA †2 CHIKAAKI KODAMA †3

With the advancement of technology node, nano lithography is becoming more complex and OPC algorithm is becoming more aggressive. For most OPC (Optical Proximity Correction) algorithm, it is not necessary to know the intensities of all the points in the simulation region. Only checking the intensities of certain points of mask (also known as tap point intensities), the behavior of the algorithm can be decided. In this research, a look up table and sum of coherent system (SOCS) model based lithography simulator has been proposed to give enough information to algorithm faster without generating the aerial image by complex mask pattern. The goal of this research is to build a simulator for intensity modelling that can provide the tap point intensity of mask as fast as possible and to guide OPC algorithm based on tap point intensity evaluation.

1. Introduction

With the advancement of technology node, nano lithography is becoming more complex and the OPC algorithm is becoming more aggressive. In order to achieve the wafer image quality, resolution enhancement technique (RET) is adopted[1].

In order to realize all the features on silicon wafer properly, optical proximity correction (OPC) is applied to modify mask pattern. OPC is one of the resolution enhancement technique (RET) which is used when the Optical Proximity Effect (OPE) of the image is large[2]. OPE is observed in the differences between target pattern and wafer image [3].

The purpose of the OPC algorithm is to provide best mask solution of target pattern so that the difference between target pattern and wafer image is minimum within process window[4]–[6]. This best solution of target pattern is an optimized modified mask pattern which includes several RET techniques. In order to obtain this solution of target pattern, most algorithms estimate aerial image on wafer derived by mask pattern, which is time consuming.

However, OPC algorithms do not require whole aerial image during iterations, though a complete areal image might be required in final quality checking. Typically, the behavior of algorithms is guided by some specific points on layout region; these points are also known as tap points or measurement points. The computation time of aerial image of mask increases with the advances of process technology. So generating a complete aerial image to guide OPC algorithms is redundant.

The purpose of this research is to guide OPC algorithms without generating aerial image of mask pattern in every iteration. It is done by obtaining intensity only at tap points.

In this paper, we propose a lithography simulation method that obtains the intensity of a point in simulation region caused by a mask pattern in the region. Our method obtains the intensity of a point in linear time of the complexity of the mask pattern (the number of corners of polygons).

†1 Tokyo Institute of Technology
†2 The University of Aizu
†3 KIOXIA Corporation

In the proposed method, intensity data of the center of simulation region caused by a rectangle whose one corner is on the origin of the region is stored in look up table (LUT). The size of LUT is linear to the size of simulation region. The intensity of an arbitrary point of the region is obtained by using the data in LUT, and the number of references is equal to the number of corners of polygons in the mask pattern in the region.

In lithography simulation, the pattern in a region under evaluation is extracted with the pattern with its surrounding region. Here, the extracted pattern is repeated infinitely under periodic boundary condition since the result for the region under evaluation is accurate enough when the surrounding region is large enough. Under this assumption, when a pattern is moved with parallel shift, the simulation result is moved accordingly. Therefore, the intensity of a point caused by a pattern can be obtained by the intensity of the center point of the simulation region after moving the pattern so that the point is moved to the center with parallel shift. In the proposed method, the intensity of the center point is obtained by using the data stored in LUT.

The LUT prepared by this research contains the intensity information of the center point of the simulation region caused by the effect of each of the rectangle whose upper left corner is located at the origin of the region. The quick computation of generating a tap point intensity arises from the fact of using this LUT because the simulator accesses the precomputed values in constant time complexity. The contribution of this research:

- A fast intensity modelling of aerial image at tap points is done by SOCS model.
- A novel approach of creating LUT that stores the intensity data of lithography simulation for a given optical kernel system such that the size of LUT is linearly proportional to the area of the simulation region. So, there is less memory complexity.

In this approach. ICCAD 2013 Simulator is considered as the benchmark simulator[7]. The rest of the paper is arranged as follows: Previous works are presented in section 2. Preliminaries and Problem Description are added in section 3. Methodology is described in section 4. Section 5 represents the experimental results and confirms the effectiveness of the proposed method. Finally, in section 6, the paper ends with conclusion and future work.

2. Previous Work

There are two kinds of OPC algorithms: Rule based and Model based. In order to find the solution of a mask faster, rule based OPC is widely adopted. Though rule based OPC is faster but it is not scalable as that of Model Based OPC. EPE minimization is one of the popular techniques in the optimization of OPC algorithms. Again, the MEEF matrix (Mask Error Enhancement Factor) matrix is adopted to guide the shifting of edges in EPE based OPC algorithms[8], [9]. There has been a proposal of a fast intensity based algorithms, but it does not consider dense pattern[10]. Presently, exploration has been made in the field of ILT (Inverse Lithography Technology) to find robust mathematical model for OPC algorithms, but because of the pixel based nature of ILT, the solution of the mask is hard to manufacture[11][12]. Similarly, there exist some aggressive OPC algorithms which give a complex mask pattern as output, but these are difficult to manufacture as well because of the pixel based nature[13][14]. Another pixel based algorithm is MOSAIC which exploits EPE under each process condition. But, as a matter of fact the output of this algorithm is complex mask pattern also which slows down the OPC algorithms[15]. In order to reduce the number of iteration of OPC algorithms, a neural network has been proposed which gives significantly better initial guess[16]. There are some algorithms which consider both EPE and PV band area called process window based OPC (PW-OPC)[17][18]. However, wafer image needs to be simulated in each process window and this is time consuming and computationally heavy. A fast printability verification method has been proposed which simulates the wafer image with less number of kernel. Though this process accelerates OPC algorithms, using more kernels will require more iterations.

However, most researches are focused on the defect or problem of previous work and try to resolve that problem. This manuscript provides a way to find tap point intensity quickly. This work is not directly related to above mentioned researches which use time consuming simulator.

In the proposed method, only tap point intensity of a mask pattern is obtained in an innovative way which provides extremely fast computation time to OPC algorithms.

3. Preliminaries and Problem Description

3.1 Optical Lithography

Lithography is a process of printing nano circuits (miniature circuits of nano meter length and beyond) into silicon wafer. In optical lithography system, light of a particular wavelength is projected through the condenser lens on the mask pattern. The light passing through the mask pattern undergoes diffraction which then passes the projection lens and falls on the silicon wafer.

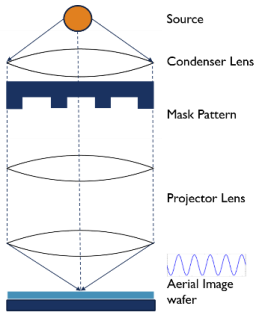


Figure 1: Lithography Process

When the light hits the mask, diffraction occurs and this creates diffraction pattern. When this diffraction pattern passes through the projection lens, the image that is formed is called aerial image. The image that forms on the silicon wafer is resist image.

3.2 Bench Mark Simulator

In this research, ICCAD 2013 simulator is used in the background for LUT generating purpose. ICCAD 2013 simulator takes a mask pattern as input and generates a wafer image as output. The intensity modelling is based on SOCS Model. It uses optical system kernel with a 193 nm wavelength and the CTR model of I_{th} = 0.225. Each pixel is 1 nm x 1 nm. The source of process variations are dose and focus which has a ±25 nm defocus range and a ±2% dose range.

Figure 2: Input and output of lithography simulator

ICCAD 2013 simulator has 24 kernels. The higher the kernel number of ICCAD 2013, the less the weight of the kernel. Kernels with low weights have less contribution to intensity. In this paper, only first kernel with weight 86.9432 is chosen for LUT generation purpose in experiments because it has the maximum weight.

3.3 Sum of Coherent System (SOCS) Model

Sum of Coherent systems model or SOCS model is useful in OPC algorithm because it provides extremely fast computation time.

In SOCS Model a mask M is given input to the model and is convoluted with each coherent kernel. Then scaler multiplication occurs with the weight of the kernel. After summation, the aerial image intensity $I(M)$ is obtained. (Figure 3).

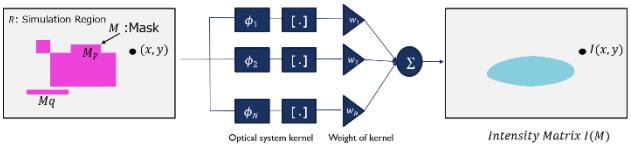


Figure 3: SOCS Model

Here $I(M)$ is the intensity matrix of mask M. Intensity map

$I(M)$ of simulation region R caused by mask $M \subset R$ consists of intensity value $I(x,y,M)$ of pixel $(x,y) \in R$.

$$I(M) = \bigcup_{(x,y)\in R} I(x,y,M)$$

In SOCS model, optical system is decomposed into the set of coherent kernels, and intensity map is formulated as follows:

$$I(M) = \sum_k w_k |M \otimes \phi_k|^2 = \sum_k w_k I_k(M) \quad (1)$$

where w_k and ϕ_k are the eigenvalue and the Eigen function of k^{th} kernel, respectively, and $\otimes$ denotes convolution operation.

The intensity of pixel $I(x,y,M)$ is the sum of the intensities derived by kernels. Let $A_k(x,y,M) = A_k^{\text{Re}}(x,y,M) + jA_k^{\text{Im}}(x,y,M)$ be the k^{th} amplitude of pixel $(x,y) \in R$ caused by mask $M \in R$. That is, let $I_k(x,y,M) = |A_k(x,y,M)|^2$. Then, the following equation holds.

$$\begin{aligned} I(x,y,M) &= \sum_k w_k I_k(x,y,M) \\ &= \sum_k w_k |A_k(x,y,M)|^2 \\ &= \sum_k w_k \left|A_k^{\text{Re}}(x,y,M) + jA_k^{\text{Im}}(x,y,M)\right|^2 \\ &= \sum_k w_k \left(A_k^{\text{Re}}(x,y,M)^2 + A_k^{\text{Im}}(x,y,M)^2\right) \end{aligned}$$

Note that $I_k(x,y,M)$ is defined as follows.

$$\begin{aligned} I_k(x,y,M) &= \left|\sum_{s_x\in S}\sum_{s_y\in S} M(x-s_x, y-s_y)\phi_k(s_x,s_y)\right|^2 \\ &= \left(\sum_{s_x\in S}\sum_{s_y\in S} M(x-s_x,y-s_y)\phi_k^{\text{Re}}(s_x,s_y) + j\sum_{s_x\in S}\sum_{s_y\in S} M(x-s_x,y-s_y)\phi_k^{\text{Im}}(s_x,s_y)\right)^2 \\ &= \left(\sum_{s_x\in S}\sum_{s_y\in S} M(x-s_x,y-s_y)\phi_k^{\text{Re}}(s_x,s_y)\right)^2 + \left(\sum_{s_x\in S}\sum_{s_y\in S} M(x-s_x,y-s_y)\phi_k^{\text{Im}}(s_x,s_y)\right)^2 \\ &= A_k^{\text{Re}}(x,y,M)^2 + A_k^{\text{Im}}(x,y,M)^2 \quad (2) \end{aligned}$$

where $\phi_k(s_x,s_y) = \phi_k^{\text{Re}}(s_x,s_y) + j\phi_k^{\text{Im}}(s_x,s_y)$, $S = \{-\lfloor H/2s \rfloor, \ldots, -2s, -s, 0, s, 2s, \ldots, \lfloor H/2s \rfloor\}$, H is the height or width of the simulation region, and s is the unit size of the simulation.

3.4 Decomposition of SOCS Model

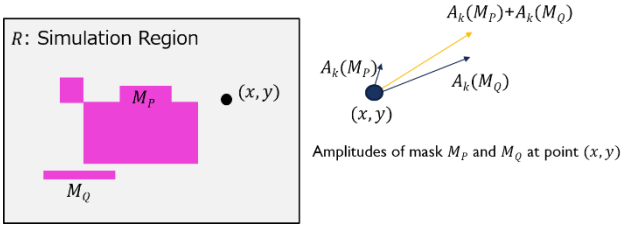


Figure 4: Decomposition of SOCS Model

This section is a continuation of section 3.3 . Let $M_P \subset R$ and $M_Q \subset R$ be masks where $M_P \cap M_Q = \emptyset$. Then,

$$A_k(M_P \cup M_Q) = A_k(M_P) + A_k(M_Q) \quad (3)$$

since

$$\begin{aligned} A_k(M_P \cup M_Q) &= A_k^{\text{Re}}(M_P \cup M_Q) + jA_k^{\text{Im}}(M_P \cup M_Q) \\ &= \left(A_k^{\text{Re}}(M_P) + A_k^{\text{Re}}(M_Q)\right) + j\left(A_k^{\text{Im}}(M_P) + A_k^{\text{Im}}(M_Q)\right) \\ &= \left(A_k^{\text{Re}}(M_P) + jA_k^{\text{Im}}(M_P)\right) + \left(A_k^{\text{Re}}(M_Q) + jA_k^{\text{Im}}(M_Q)\right) \\ &= A_k(M_P) + A_k(M_Q). \end{aligned}$$

Also, note that $I_k(M_P \cup M_Q) \neq I_k(M_P) + I_k(M_Q)$ since

$$\begin{aligned} I_k(M_P \cup M_Q) &= |A_k(M_P \cup M_Q)|^2 = |A_k(M_P) + A_k(M_Q)|^2 \\ &= \left(A_k^{\text{Re}}(M_P) + A_k^{\text{Re}}(M_Q)\right)^2 + \left(A_k^{\text{Im}}(M_P) + A_k^{\text{Im}}(M_Q)\right)^2 \\ &\neq A_k^{\text{Re}}(M_P)^2 + A_k^{\text{Re}}(M_Q)^2 + A_k^{\text{Im}}(M_P)^2 + A_k^{\text{Im}}(M_Q)^2 \\ &= |A_k(M_P)|^2 + |A_k(M_Q)|^2 = I_k(M_P) + I_k(M_Q). \end{aligned}$$

3.5 Tap point Intensity

Tap points are the points which are primarily defined on the target pattern and evaluated in the aerial image to check the quality of the wafer image. In the Figure 5, a rectangle shape mask pattern and its corresponding wafer image is shown. At tap point p_1, $I_{threshold\_min} < I_{p_1} < I_{threshold\_max.}$ $I_{threshold\_max}$ and $I_{threshold\_min}$ defines the acceptable range of intensity of tap point, respectively. At p_2, $I_{p_2} < I_{threshold\_min}$. Again, at p_3, $I_{p_3} > I_{threshold\_max.}$ So, though at p_1 wafer image quality is protected, but at p_2 and p_3 it is not.

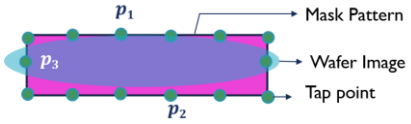


Figure 5: Tap points in a mask pattern.

3.6 Periodic Boundary Condition

In this research, experiments are done under periodic boundary condition which states that: the simulation region R is large enough such that the outside of simulation region R does not affect the intensity of the center region of a mask pattern and R is repeated infinitely in 2 dimensional plane. Such assumption is practically correct if simulation region contains peripheral area at least as wide as the optical radius.

3.7 Parallel shift

Following the periodic boundary condition mentioned in section 3.6, a mask is parallel shifted in 2 dimensional simulation region R in this research. In Figure 6, a mask M is defined in R which has a tap point $p(x_p, y_p)$. The mask is shifted by d_x in x direction and d_y in y direction by the following parallel shift operation: $shift(M, d_x, d_y)$

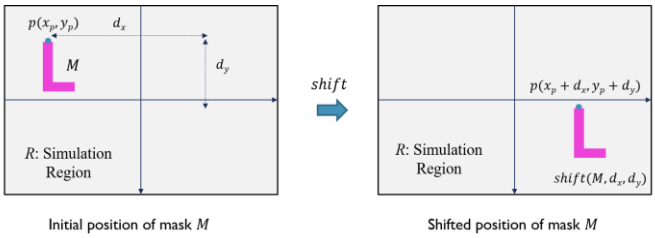


Figure 6: Demonstration of parallel shift

So, the new position of p becomes $(x_p + d_x, y_p + d_y)$. Here,

$$I\left(x_p + d_x, y_p + d_y, shift(M, d_x, d_y)\right) = I(x_p, y_p, M) \quad (4)$$

where I is the intensity of M. Equation (4) is true because the

relative position of M and p is maintained under periodic boundary condition. So, when a relative position of pattern M and tap point p is maintained, parallel shift does not affect the intensity of the tap point.

3.8 Problem Description

The target pattern can be a rectangle or a polygon. In this research, a polygon is decomposed into rectangles and evaluated.

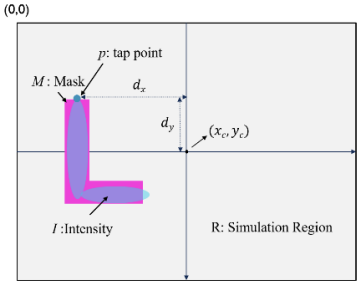


Figure 7: Target pattern in a Simulation Region

Here, I is the intensity caused by mask polygon M in the simulation region which corresponds to the wafer image. Simulation region R is a rectangular region whose upper left corner is at (0,0) which is considered as the origin. The target of this research is to calculate the tap point intensity $I(p, M)$ at point p in the fastest possible way. Finding this intensity faster will increase the efficiency and decrease the computation time of OPC algorithm.

In order to find the intensity value $I(p, M)$ at the point p, the polygon is positioned in such a way that the tap point p is located at the center point of the region (x_c, y_c) by shift operation shift(M, d_x, d_y).

Here, $shift(M, d_x, d_y)$ is the parallel shift operation of mask in which mask $M \subset R$ is shifted in x and y direction by d_x and d_y respectively in such a way that mask M as well as point p is shifted by shift operation so that p is positioned at (x_c, y_c). As it is mentioned before, here the relative position of M and p is maintained. So, a parallel shift will not change the intensity caused by M at point p even if it is shifted to the center. So,

$$I(p, M) = I(x_c, y_c, \text{shift}(M, d_x, d_y))$$

The polygon is then decomposed into rectangles as shown in the Figure 8.

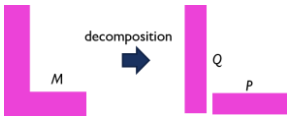


Figure 8: Decomposition of M into rectangles P and Q

The amplitude of intensity in a kernel caused by M on tap point p is the sum of amplitudes of intensities caused by the decomposed rectangles. The center point amplitude of each of the decomposed rectangle is obtained by center point amplitudes of four rectangles whose upper left corners are the origin of region.

So, the final target of the research is to know the amplitude of the center point of simulation region in the most efficient and quickest way.

4. Methodology

The proposed lithography simulator has two modules.

1. Simulator Unit
2. Preprocessing Unit

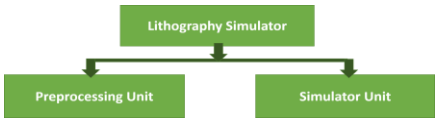


Figure 9: Overview of proposed lithography simulator

4.1 Simulator Unit

In the simulator unit, SOCS model is implemented to calculate the intensity. Let us suppose, we want to calculate the intensity of tap point $p(x_p, y_p)$ caused by rectangle M as a mask. Let, the upper left corner of rectangle M is (x, y). The upper left corner of the simulation region R is (0, 0) and the center point is (x_c, y_c).

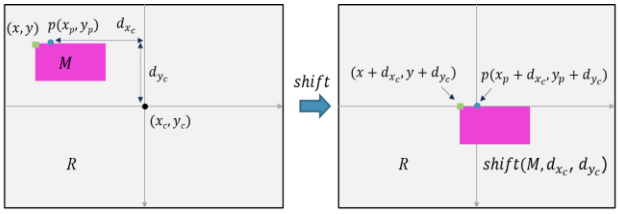


Figure 10: Target pattern M in with tap point p

Initially, the simulator performs parallel shift operation $shift(M, d_{x_c}, d_{y_c})$. By this operation, both M and p is moved by d_{x_c} in x direction and d_{y_c} in y direction such that p is positioned at the center point (x_c, y_c) of the simulation region R. So, the new position of p becomes $(x_p + d_{x_c}, y_p + d_{y_c})$. Here, by equation (4):

$$I(x_p, y_p, M) = I(x_p + d_{x_c}, y_p + d_{y_c}, shift(M, d_{x_c}, d_{y_c}))$$

The rectangle $shift(M, d_{x_c}, d_{y_c})$ is represented by four rectangles $M(x+d_{x_c}, y + d_{y_c})$, $M(x + d_{x_c} + w, y + d_{y_c})$, $M(x + d_{x_c}, y + d_{y_c} + h)$ and $M(x + d_{x_c} + w, y + d_{y_c} + h)$. The lower right corner points are represented by small green rectangles in the Figure 11(a)-11(d). The upper left corner of rectangle $M(x+d_{x_c}, y + d_{y_c})$ is at origin, but the lower right corner coincides with that of rectangle $shift(M, d_{x_c}, d_{y_c})$ which is $(x+d_{x_c}, y + d_{y_c})$. (Figure 11(a))

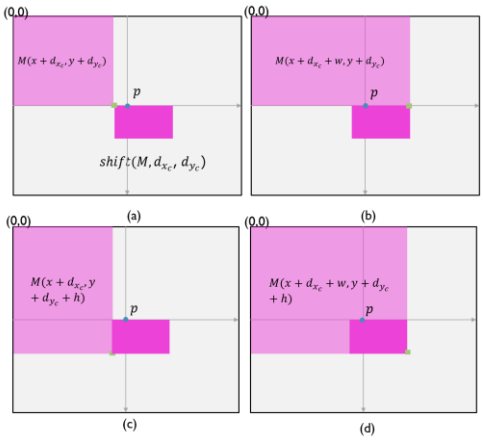


Figure 11: Representation of the upper left (a), upper right (b), lower left (c) and lower right corner (d) corners of M

Following SOCS Model and by the associative property of convolution operation, the amplitude of intensity at a point of simulation region can expressed by the sum of amplitudes of intensities caused by $M(x+d_{x_c}, y+d_{y_c})$, $M(x+d_{x_c}+w, y+d_{y_c})$, $M(x+d_{x_c}, y+d_{y_c}+h)$ and $M(x+d_{x_c}+w, y+d_{y_c}+h)$. So, the amplitude of intensity of a point p caused by rectangle M at k^{th} kernel is given as follows:

$$A_k(p,M) = A_k\left(p, M(x+d_{x_c}+w, y+d_{y_c}+h)\right) - A_k\left(p, M(x+d_{x_c}+w, y+d_{y_c})\right) - A_k\left(p, M(x+d_{x_c}, y+d_{y_c}+h)\right) + A_k\left(p, M(x+d_{x_c}, y+d_{y_c})\right) \quad (5)$$

where $A_k(M(x,y))$ is the amplitude of intensity of rectangle $M(x,y)$ for k^{th} kernel. So, intensity at point p for k^{th} kernel becomes:

$$I_k(p,M) = A_k(p,M)^2 \quad (6)$$

So, the summation of amplitudes of each component of equation (5) at point p of simulation region gives the resultant intensity $I_k(p,M)$. So if the values of the amplitudes caused by a rectangles whose upper left corner is at the origin are known quickly, $I_k(p,M)$ can be calculated easily. The amplitudes of intensity of these rectangles can be obtained by using amplitudes stored in LUT with shift operation.

So, in our implementation, M is shifted first so that point p is at the center point, and then M is represented by four rectangles whose upper left corner is the origin. By this approach, the computation time becomes extremely fast and less computation resource is needed to run the simulator unit.

4.2 Preprocessing Unit

This unit generates LUT for a given optical system of kernels. The data stored in the LUT are the center point amplitudes values of the rectangle whose upper left corner is at the origin of the simulation region (0, 0). These rectangles (whose upper left corner is the origin of the simulation region) are the building blocks of LUT. The purpose of LUT is to make the computation faster by previously computing the convolution of mask and optical kernels[19].

4.2.1 Division of Layout

Initially, in our experiments simulation region and step size is provided. The dimension of layout is 2048 nm x 2048 nm. With step size =s nm, the total number of building blocks or rectangles with upper left corner at origin of layout =layout area/$step\ size^2$ = $2048/s^2$

So, after division of layout, we get a number of rectangles whose upper left corner is at (0, 0).

4.2.2 Creation of rectangle at (0, 0)

Rectangles are created whose upper left corner is at origin (0, 0). In this explanation, there are $2048/s^2$ rectangles. The number of rectangles can also be represented in the following way: $(0,0) \rightarrow (i * s, j * s)$; where i and j represent the index of axis in x and y direction respectively.

4.2.3 Calculation of small tile and parallel shift

A small tile is defined as the first basic rectangle which is generated to initialize the computation of preprocessing of unit. Basically, a small tile is a square whose length of one side is equal to step size measurement.

For this example, $2048/s^2$ small tiles are required to complete the computation of $2048/s^2$ rectangles. The first small tile is calculated using ICCAD 2013 lithography simulator. In the following figure, the basic small tile is shown whose upper left corner is at (0, 0) and lower right corner is at *(s, s)* nm. In this research:

step size, s = length of shift = length of one side of basic small tile

Figure 12: ICCAD 2013 simulator generates the basic small tile

So, if the amplitude of intensity of a small tile at an arbitrary position is calculated, a parallel shift is done to the basic tile with appropriate shift amount. In the following figure, the basic small tile $M(s, s)$ whose upper left corner is at (0, 0) is shifted by $3s$ in x direction and $2s$ in y direction such that the new small tile $shift(M(s, s), 4s, 3s)$ is obtained.

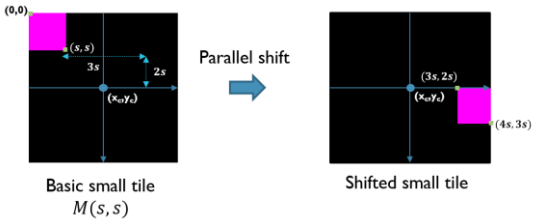


Figure 13 Demonstration of parallel shift of small tile

Hence,

$$M(s,s) \rightarrow shift(M(s,s), 4s, 3s)$$

Here, by equation (4):

$I(x_c, y_c, shift(M(s,s), 4s, 3s)) = I(x_c - 4s, , y_c - 3s, M(s,s))$

Note that $I(x_c - 4s, , y_c - 3s, M(s,s))$ is obtained by ICCAD 2013 simulator. In general, $I(x_c, y_c, shift(M(s,s), dx, dy))$ is given as $I(x_c - d_x, y_c - d_y, M(s,s))$.

4.2.4 Dynamic Programming Approach

Let $M(w,h) \subset R$ be the rectangle mask placed on the upper left corner of R whose width and height are w and h, respectively.

In our LUT T, the amplitudes of pixels are stored. More precisely, the complex value $T(w,h) = A_k(x_c, y_c, M(w,h))$ is stored where (x_c, y_c) be the center pixel of simulation region R, $w,h \in \{s, 2s, \dots, \lfloor H/s \rfloor\}$, H is the height and width of simulation region R, and s is the step size.

In our preprocessing unit to generate LUT, the following formula of dynamic programming is used.

$$\begin{aligned} A_k(x_c, y_c, M(w,h)) &= A_k(x_c, y_c, \text{shift}(M(s,s), w-s, h-s)) \\ &+ A_k(x_c, y_c, M(w-s, h)) \\ &+ A_k(x_c, y_c, M(w, h-s)) \\ &- A_k(x_c, y_c, M(w-s, h-s)) \end{aligned} \quad (7)$$

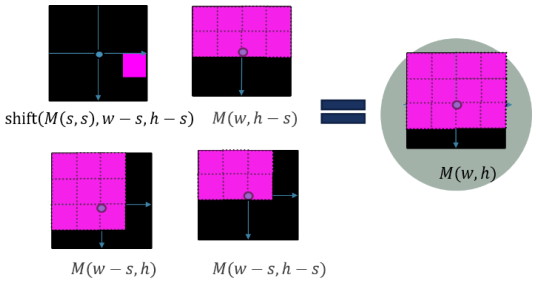


Figure 14 Dynamic Programming Approach

Note that $A_k(x_c, y_c, \text{shift}(M(s,s), w-s, h-s))$ is required in the recurrence formula. All the values required are obtained by one time lithography simulation for rectangle $M(s,s)$ (Figure 14) since $A_k(x_c, y_c, \text{shift}(M(s,s), w-s, h-s)) = A_k(x_c - w + s, y_c - h + s, M(s,s))$.

The amplitude of pixel (x_c, y_c) caused by an arbitrary rectangle mask $\text{shift}(M(w,h), d_x, d_y)$ is derived as follows.

$$A_k\left(x_c, y_c, \text{shift}(M(w,h), d_x, d_y)\right)$$

$$= A_k\left(x_c, y_c, M(w + d_x, h + d_y)\right) - A_k\left(x_c, y_c, M(w + d_x, d_y)\right)$$

$$- A_k\left(x_c, y_c, M(d_x, h + d_y)\right)$$

$$+ A_k\left(x_c, y_c, M(d_x, d_y)\right)$$

In order to get the intensity value of pixel $(x, y) \in R$ caused by mask $\text{shift}(M(w,h), d_x, d_y)$, the following formula is used.

$$I\left(x, y, \text{shift}(M(w,h), d_x, d_y)\right)$$

$$= I\left(x_c, y_c, \text{shift}(M(w,h), d_x - x + x_c, d_y - y + y_c)\right)$$

$$= \sum_k w_k I_k\left(x_c, y_c, \text{shift}(M(w,h), d_x - x + x_c, d_y - y + y_c)\right)$$

$$= \sum_k w_k \left|A_k\left(x_c, y_c, \text{shift}(M(w,h), d_x - x + x_c, d_y - y + y_c)\right)\right|^2$$

4.2.5 Look Up Table (LUT)

In this research, a LUT is prepared which stores the amplitudes of the center point of the simulation region caused by the rectangles whose upper left corner is at the origin of the region following equation (7).

The size of LUT depends on the number of rectangle. As the number of rectangle is s^2 in this example, so the number of data points stored in the LUT is also s^2.

4.3 Algorithm Flow of Proposed lithography simulator

The overall flow of the algorithm of proposed simulator is given in Figure 15.

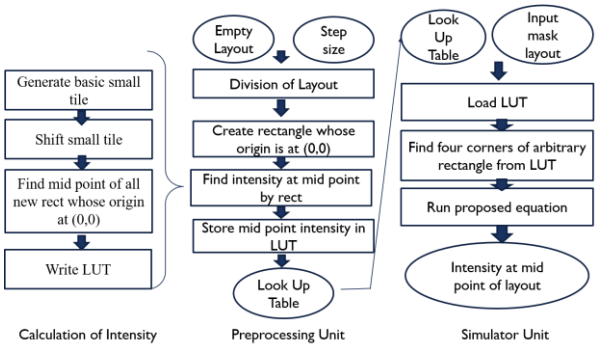


Figure 15: Flowchart of proposed simulator

5. Experimental Results

The two separate units (Preprocessing and Simulator) are separately tested and verified.

5.1 Preprocessing Unit Results

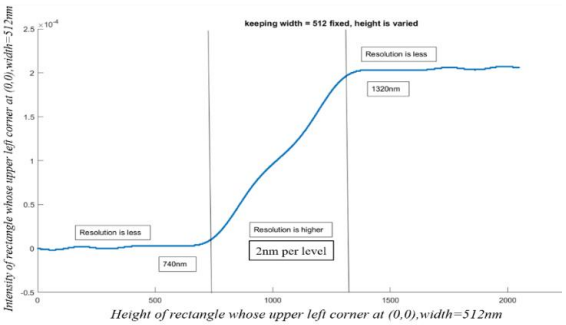
As mentioned before, the step size is the input to preprocessing unit which determines the size and precision of LUT. The step size of LUT is carefully chosen to be 2 nm. In order to determine this the following experiment is done.

5.1.1 Finding the minimum feature size

An experiment is carried to find the minimum step size of the proposed simulator. ICCAD 2013 simulator is used to generate the basic tile for the proposed simulator, so minimum feature which the ICCAD simulator can print is also the minimum feature which the proposed simulator can print.

In the layout, a rectangle of whose upper left corner at (0, 0) and width 512 nm and height 0 nm is chosen and the effect of intensity of this rectangle at center point of layout is recorded. Gradually, the height of this rectangle is increased by 1 nm and for each increment, the effect of intensity of this rectangle at center point is recorded. The height is varied from 0 nm to 2048 nm [the upper limit of layout]. Finally, a graph is plotted with the recorded intensity values and height.

X axis represents, the varying height of this particular rectangle and Y axis represent the intensity values.



Height Vs Intensity Graph of Benchmark simulator: ICCAD Lithosim 2013

Figure 16: Graph to find minimum step size *s*

Analyzing the graph, the following features are found:

- For a particular rectangle of width = 512 nm, if height < 740 nm, the intensity value saturates and is close to 0. Saturation of intensity means, for a change in height of this rectangle, change of intensity does not occur. So for any large number

rectangles of different heights, the intensity value is same.

- For a particular rectangle of width = 512 nm, if height > 1320 nm, the intensity value saturates.
- Sharp change of intensity occurs if the height is varied from 740 nm to 1320 nm for this particular rectangle. Because it is relatively near the center point of the layout. The change of intensity is not linear to the distance but fluctuated due to lithographic behavior. Studying this sharp change of intensity value, it is found that: for change of height of any 2 rectangles, change of intensity does not occur. For example: rectangle of height 1000 nm and 1001 nm with width 512 nm will have the same intensity value.

From this study, the minimum feature size of ICCAD simulator, hence the proposed simulator is determined to be 2 nm.

5.1.2 Time Complexity of Preprocessing Unit

In order to determine the time complexity of preprocessing unit, an experiment is done by varying the step size and recording the computation time required to generate the LUT for that particular step size.

The time complexity is linear for preprocessing unit. When the step size is 32, there are 4096 data point and the time taken by the preprocessing unit is 98 second. When step size is 4, the number of the data point is 262144 and time needed by preprocessing unit is 13500 second.

5.1.3 Features of 2nm LUT

This LUT is input to the simulator unit. In Table 1, the features of 2nm step size LUT is tabulated.

Table 1: Features of 2 nm LUT

Features of 2 nm LUT	
Size of LUT	282 mb
Step size	2 nm
# of data point	1048576

As the time complexity of preprocessing unit is linear, the program can be efficiently used for generating LUT for larger layout area and different optical system.

5.2 Simulator Unit Results

Using the 2 nm LUT, some test cases of mask pattern is evaluated to validate the simulator unit. A comparison of result of proposed simulator and the result of the same test from ICCAD simulator is shown below.

5.2.1 Case 1: Straight lines

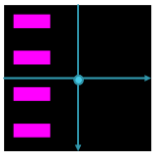


Figure 17: Layout for case 1

Case-1 includes four equidistant straightlines as shown in Figure 17.

5.2.2 Case 2: Polygon

Case 2 includes a polygon. In the proposed simulator, a L shaped polygon is broken down in to 3 rectangles that forms the L shape pattern. In Figure 18, the L-shape polygon in Case 2 is shown..

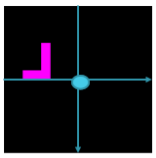


Figure 18: Layout for case 2

Let amplitude of L shaped polygon at center is $A_1(poly)$ for the 1st kernel. The amplitudes of intensities of $rect_0$, $rect_1$ and $rect_2$ are $A_1(rect_0)$, $A_1(rect_1)$ and $A_1(rect_2)$ respectively for 1st kernel. So, from equation (4),

$$A_1(poly) = A_1(rect_0) + A_1(rect_1) + A_1(rect_2)$$

5.2.3 Evaluation of tests:

A comparison of intensities of proposed simulator and bench mark simulator is done and shown in Table 2. Here A_1^{Re} and A_1^{Im} are the real and imaginary part of amplitudes for the first kernel, respectively and I is the intensity. For the experiments of this research, only 1st kernel is used. For comparison, smaller values are considered to show the accuracy of the simulator. The simulator works accurately even if the precise values of intensities are taken for comparison. Precision of simulator is up to 6 decimal places

Table 2: Evaluation of tests$(\times 10^{-5})$

Simulator	Benchmark			Proposed		
Case	A_1^{Re}	A_1^{Im}	I	A_1^{Re}	A_1^{Im}	I
1	9.9	57.7	3.0	9.9	57.7	3.0
2	-1.1	-4.4	0.0	-1.1	-4.4	0.0

5.2.4 Time Complexity and Performance Analysis

To evaluate the time complexity of simulator unit, several data points are generated. Here the data points represent arbitrary rectangles located at random positions in the layout (constraint: no overlapping region). The time to compute the data point is noted and a graph is plotted based on these values.

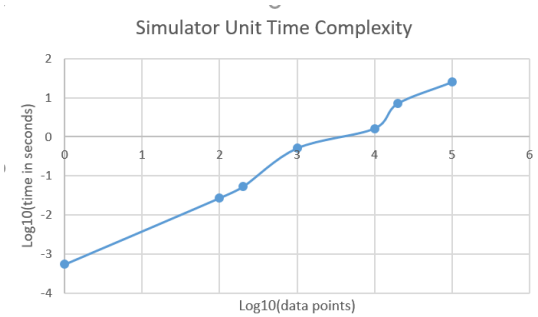


Figure 19 Time complexity of simulator unit

In Figure 19, a graph is shown which shows the time complexity of the simulator unit. In the X axis, there is data point

and in the Y axis there is time required to generate that amount of data points. So, the graph shows that, time complexity is piecewise linear. It means that the slope of the line changes if data points is added 10 time more than before.

Evaluating the above graph and also evaluating the different mask layout in section 5, the following result is obtained.

Table 3: Performance analysis of simulator unit

Performance Analysis of Simulator	
Simulation time of 1 rectangle	340 us (average)

Simulation time for one tap point in ICCAD 2013 simulator is 60 seconds in an average because it generates complete aerial image.

6. Conclusion and Future Work

In this paper, we have presented a lithography simulator that gives an innovative solution to calculating intensity of tap point in a mask. Additionally, this will help to guide OPC algorithm (for example: SRAF placement algorithm, hammer insertion algorithm etc.). The experimental results show the validity of the proposed method and also the time efficiency.

The proposed method is only evaluated in nominal process condition. One of the future work is to enhance the robustness of the simulator to other process condition. Finally, an OPC algorithm will be developed based on this simulator which will be able to deal with complicated mask model in any process condition.

Acknowledgments Thanks to Shimpei Sato, Assistant Professor, Tokyo Institute of Technology for his contribution to the manuscript.

Reference

[1] R. F. Pease and S. Y. Chou, “Lithography and Other Patterning Techniques for Future Electronics,” *Proc. IEEE*, vol. 96, no. 2, pp. 248–270, Feb. 2008.

[2] C. Spence, “Full-chip lithography simulation and design analysis: how OPC is changing IC design,” in *SPIE*, 2005, pp. 1–14.

[3] C. Mack, *Fundamentals of optical lithography*. Cambridge: Cambridge University Press, 2007.

[4] A. Awad, A. Takahashi, S. Tanaka, and C. Kodama, “A Fast Process-Variation-Aware Mask Optimization Algorithm with a Novel Intensity Modeling,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 3, pp. 998–1011, 2017.

[5] A. Awad, A. Takahashi, and C. Kodama, “A fast mask manufacturability and process variation aware OPC algorithm with exploiting a novel intensity estimation model,” *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E99A, no. 12, pp. 2363–2374, 2016.

[6] A. Awad, A. Takahashi, S. Tanaka, and C. Kodama, “Intensity Difference Map (IDM) Accuracy Analysis for OPC Efficiency Verification and Further Enhancement,” *IPSJ Trans. Syst. LSI Des. Methodol.*, vol. 10, pp. 28–38, 2017.

[7] S. Banerjee, Z. Li, and S. R. Nassif, “ICCAD-2013 CAD contest in mask optimization and benchmark suite,” *IEEE/ACM Int. Conf. Comput. Des. Dig. Tech. Pap. ICCAD*, no. c, pp. 271–274, 2013.

[8] N. B. Cobb and Y. Granik, “Model-based OPC using the MEEF matrix,” *22nd Annu. BACUS Symp. Photomask Technol.*, vol. 4889, p. 1281, 2002.

[9] J. Lei, L. Hong, G. Lippincott, and J. Word, “Model-based OPC using the MEEF matrix II,” *Opt. Microlithogr. XXVII*, vol. 9052, p. 90520N, 2014.

[10] P. Yu and D. Z. Pan, “A novel intensity based optical proximity correction algorithm with speedup in lithography simulation,” *IEEE/ACM Int. Conf. Comput. Des. Dig. Tech. Pap. ICCAD*, pp. 854–859, 2007.

[11] L. Pang, Y. Liu, and D. Abrams, “Inverse Lithography Technology (ILT): what is the impact to the photomask industry?,” *Photomask Next-Generation Lithogr. Mask Technol. XIII*, vol. 6283, p. 62830X, 2006.

[12] Y. Liu, D. Abrams, L. Pang, and A. Moore, “Inverse lithography technology principles in practice: unintuitive patterns,” 2005, p. 599231.

[13] S. Tanaka, S. Inoue, T. Kotani, K. Izuha, and I. Mori, “Impact of OPC aggressiveness on mask maufacturability,” *Photomask Next-Generation Lithogr. Mask Technol. X*, vol. 5130, p. 23, 2003.

[14] N. Cobb, “Flexible sparse and dense OPC algorithms,” 2005, p. 693.

[15] J. R. Gao, X. Xu, B. Yu, and D. Z. Pan, “MOSAIC: Mask optimizing solution with process window aware inverse correction,” *Proc. - Des. Autom. Conf.*, 2014.

[16] W. C. Huang *et al.*, “Intelligent model-based OPC,” 2006, p. 615436.

[17] P. Yu, S. X. Shi, and D. Z. Pan, “Process variation aware OPC with variational lithography modeling,” *Proc. - Des. Autom. Conf.*, pp. 785–790, 2006.

[18] P. Yu, “True process variation aware optical proximity correction with variational lithography modeling and model calibration,” *J. Micro/Nanolithography, MEMS, MOEMS*, vol. 6, no. 3, p. 031004, 2007.

[19] N. Cobb, “Fast Optical and Process Proximity Correction Algorithms for Integrated Circuit Manufacturing,” *www.video.eecs.berkeley.edu/paper*, p. 139, 1998.