Design-Aware Mask Inspection

Abde Ali Kagalwalla, Puneet Gupta, Christopher J. Progler, and Steve McDonald

***Abstract*—Mask inspection has become a major bottleneck in the manufacturing flow taking up as much as 40% of the total mask manufacturing time. In this paper, we explore techniques to improve the reticle inspection flow by increasing its design awareness. We develop an algorithm to locate nonfunctional features in a postoptical proximity correction layout without using any design information. Using this, and the timing information of the design (if available), the smallest defect size that could cause the design to fail is assigned to each reticle feature. The criticality of various reticle features is then used to partition the reticle such that each partition is inspected at a different pixel size and sensitivity so that the false and nuisance defect count is reduced without missing any critical defect. We also develop an analytical model to estimate the false and nuisance defect count. Using those models, our simulation results show that this design-aware mask inspection can reduce the false and nuisance defect count for a critical polysilicon layer from 80 defects down to 49 defects, leading to substantial reduction in defect review load. We also develop a model to estimate first pass yield (FPY) and show that our method can improve the FPY for a polysilicon layer from 11% to 30%. Apart from the polysilicon layer, the potential benefit of this approach is analyzed for active, contact and all the metal/via layers.**

***Index Terms*—Computer-aided design (CAD), design for manufacturability (DFM), mask inspection, mask manufacturing, reticle, semiconductor manufacturing.**

I. Introduction

A RETICLE (mask) is basically a stencil that determines what patterns eventually print on the wafer. The increasing aggressiveness of various resolution-enhancement techniques such as optical proximity correction (OPC), phase shift mask, and subresolution assist features (SRAFs) along with decreasing feature sizes has increased the complexity, and therefore the cost, of reticles considerably [1]. Keeping mask cost in control is extremely critical, especially for low-volume designs. The use of double patterning for 22 nm and beyond will substantially increase mask cost. The problem is likely to get even worse for future patterning technologies (e.g., multilayer extreme ultraviolet lithography masks and nanoimprint templates).

Reticle inspection is a significant contributor to the mask cost. In fact, mask inspection is more challenging (and expensive) than mask writing itself [2]. High-resolution reticle inspection tools are required to detect every potential printable defect in order to prevent yield loss, but inspection tools can report a large number of defects that do not affect yield. As a result postinspection review of the yield impact of defects has become very time consuming. This slow, and often manual inspection flow, has a considerable impact on mask cost and turnaround time (TAT). Hence, there is a strong need to improve the inspection flow.

In this paper, we develop a methodology to assign criticality to different mask features based on their design impact. Mask inspection tools can use this information to adapt their resolution locally for different regions without missing any critical defects, thereby saving inspection time. We now present a brief introduction of current mask inspection methodology, followed by a survey of some related work and a summary of our contributions.

A. Mask Inspection Primer

A comprehensive inspection of the reticle must be done by the mask shop before sending it to the fabs. The basic steps of inspection are shown in Fig. 1.

Initially, the reticle is passed through an inspection tool such as KLA-Tencor's Terascan [3] or NEC's LM series [4] that takes an image of die on the mask and compares it to a reference database or another die (die–database or die–die modes). The difference between the two image intensities is found and if the difference exceeds a predefined threshold, the difference pattern is labeled a defect. The inverse of this threshold is referred to as sensitivity. These tools can have a pixel size as low as 55 *nm* and can detect critical dimension (CD) defects as small as 20 *nm* on the mask at maximum sensitivity (minimum threshold) [3].

Inspection tools can generate a very large number of defects (100+), most of which do not impact the final design. Defects can be classified as shown in Fig. 2. A *false defect* is an incorrect detection reported by the inspection tool due to vibration, misalignment, optical distortion, error in database rendering (die–database mode), etc. Real defects are caused either due to misalignment or vibration of the mask writer (CD defects) or due to contamination of the mask (contamination defects). Inspection tools typically have different algorithms to detect these two categories of defects and hence have different sensitivities for these defects. Many real defects do not print on the wafer. Among printable defects, some lie on noncritical regions of the design such as dummy fill or redundant vias. Only a small fraction of the defects reported by the inspection tool really matter. All the nonprintable and noncritical defects are also called *nuisance defects*. Reducing the number of false and nuisance defects reported by the inspection tool is

Manuscript received July 5, 2011; revised October 27, 2011; accepted December 2, 2011. Date of current version April 20, 2012. This work was supported by the IMPACT UC Discovery Grant and NSF CAREER Award No. 0846196. This paper was recommended by Associate Editor D. Z. Pan.

A. A. Kagalwalla and P. Gupta are with the Department of Electrical Engineering, University of California at Los Angeles, Los Angeles, CA 90095 USA (e-mail: abdeali@ucla.edu; puneet@ee.ucla.edu).

C. J. Progler and S. McDonald are with Photronics, Inc., Allen, Dallas, TX 75013 USA.

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/TCAD.2011.2181909

0278-0070/$31.00 © 2012 IEEE

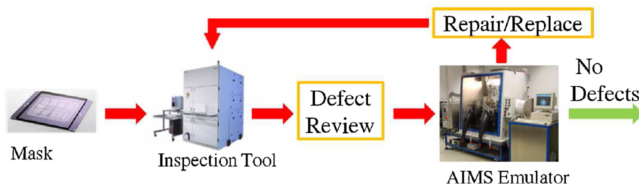

Fig. 1. Key steps of reticle inspection.

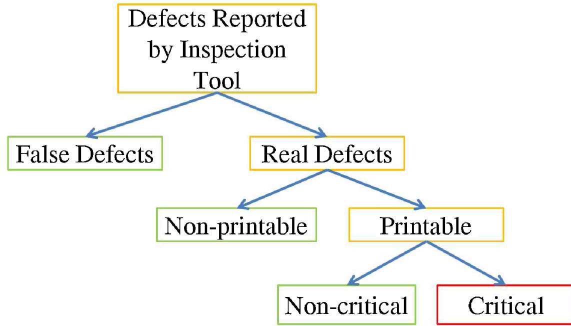

Fig. 2. Various categories of defects reported by inspection tool.

essential to reduce inspection cost. Reducing nuisance defects is particularly important to mask shops as it impacts first pass yield (FPY), which is the fraction of total masks manufactured that can be shipped without repair or detailed review.

The next step in mask inspection is defect review where each defect reported by the inspection tool is checked to find out if it really matters. False, nonprintable, and noncritical defects are filtered out during this step. Images of defects reported by the inspection tool are analyzed using software tools [5], [6] or manually. Often defect images need to be recaptured at a better resolution. For this, the inspection tool could be reused (online review) or an e-beam inspection is employed [7]. After pruning out a significant fraction of false/nonprintable/noncritical defects, the mask is passed through an aerial imaging tool. Aerial image measurement system (AIMS) [8] is essentially a hardware emulator of the wafer stepper that operates at optical settings similar to the stepper and gives a very accurate estimate of the printability of defects. Although extremely accurate, AIMS is slow and cumbersome. Hence, minimizing the number of defects that have to pass through AIMS tools is important in order to ensure reasonable TAT. Defects that are found to be printable by the AIMS tool are then either repaired or if they are unrepairable the reticle must be replaced. The repaired or replaced reticle must again go through this inspection cycle. Because of the manual steps and use of AIMS tool, defect review is typically the slowest part of reticle inspection.

B. Related Work

There has been considerable work to improve mask manufacturing by using design intent although most of it has focused on OPC. For instance, Banerjee *et al.* [9] used estimates of on/off current of transistors, based on simulated resist contours, to reduce OPC runtime and mask complexity. Zhang *et al.* [10] modeled the impact of corner rounding in printed transistors on saturation current and integrated their model into a OPC framework. Similarly, [11] and [12] used device performance estimates to tune the aggressiveness of optical correction achieving up to 93% reduction in mask complexity. Gupta *et al.* [13] used electrical and design metrics to reduce OPC runtime and mask write cost. Chan *et al.* [14] used estimates of design metrics like delay and power to improve the evaluation of process window. These approaches indicate that considerable benefit can be derived by using design intent to reduce the inherent pessimism in various mask manufacturing steps, including inspection.

The traditional approach to mask inspection discussed in the previous section does not use any design information to assess the criticality of defects. Defect disposition is done only on the basis on printability that is determined using software tools like Virtual Stepper [5], [6] along with AIMS emulation [8]. It assumes that all printable defects larger than a threshold size (say 10% of mask CD) are critical. If design information is available to mask shops, they may be able to avoid the expensive process of repair/replacement of the mask due to printable but noncritical defects. Design information can also be used to reduce false and nuisance defects reported by the inspection tool.

Communicating design intent to the inspection tool in the form of additional control layers has been suggested before [15]–[17]. Mask shops can use design information to lower the inspection sensitivity of noncritical regions in order to reduce the number of false and nuisance defects. Hedges *et al.* [16] have shown that up to 100× reduction in nuisance defect count is possible just by using variable sensitivity during reticle inspection. Current inspection tools allow the user to define inspection sensitivity on a per pixel basis. But memory requirements to store this sensitivity information are impractical since a reticle can have up to 10^{12} pixels. These approaches assume that mask shops know the design criticality of the layout that is rarely the case. Driessen *et al.* [18] analyzed a post-OPC layout to extract some noncritical features in the absence of any design data. Stoler *et al.* [19] extracted some criticality information as part of manufacturing rule check. Both these approaches focus on extracting assist features from the layout that are a major source of nuisance defects.

C. Our Work

This paper is an extension of [20]. The key contributions of this paper are as follows.

1) We develop a graph-based algorithm to locate nonfunctional features (redundant and dummy features) in a post-OPC layout (flat and 10× more complex than pre-OPC layout) in the absence of any design information.
2) We assign minimum defect size that impacts the design to each feature of the reticle for both CD and contamination defects. This is inferred using the timing slack of critical paths and the location of nonfunctional features found using the method mentioned above. This analysis is done for poly, active, contact, and all the back-end layers.
3) Using the minimum defect size of each feature of a reticle, we partition the layout using a recursive algorithm that is an improvement over the scanline-based heuristic in [20], where each partition is assigned a different pixel size and sensitivity to minimize false

and nuisance defects. First-order models for false and nuisance defects are developed to do this partitioning.

4) We develop a model to estimate FPY of masks and show the improvement achieved by our design-aware mask inspection.

The remainder of this paper is organized as follows. Section II discusses the nonfunctional feature-finding problem. Section III describes the methodology to assign criticality (minimum defect size that matters) to each layout feature. Section IV develops models for the inspection process. Section V then develops a partitioning algorithm that uses the criticality assignment and inspection process models to develop a design-aware inspection flow. The results are covered in Section VI. Section VII concludes this paper.

II. Nonfunctional Feature Finding

In this section, the following problem is explored. Given a post-OPC layout, identify nonfunctional features of the layout. We focus on locating redundant vias and dummy fill geometrically. Other nonfunctional features such as nontree routes and assist features can also be found using our graph-based methodology, but are not explored in this paper. The layout is assumed to have only rectilinear shapes, and that floating dummy fill in different metal layers are not connected through a via.[1] This is consistent with most commercial fill synthesis tools.

In order to identify nonfunctional features, we first fracture the layout into rectangles. A scanline-based algorithm is used to construct a neighborhood graph for these rectangles. The neighborhood graph is then simplified using some edge contraction operations. This reduced neighborhood graph (RNG) can then help identify dummy fill and redundant vias. The various steps of our approach are detailed below.

1) *Algorithm Steps.*

a) *Fracturing polygons:* The rectilinear polygons are fractured into rectangles using a simple horizontal slicing method [21]. The rectangles are then stored in different sets based on their layer. For example, a rectangle corresponding to a Metal 2 shape is stored in two sets, M_2V_1 and M_2V_2. A set M_iV_j corresponds to all rectangles belonging to the same/adjacent metal or via layers, where a via layer V_j connects metal layer M_j and M_{j+1}.[2]

b) *Neighborhood graph construction:* The new layout with fractured polygons is used to construct an undirected neighborhood graph, $G(V, E)$ in which every rectangle of the fractured layout corresponds to a vertex and edge $(u, v) \in E$ if the two corresponding rectangles are physically in contact with each other in the layout.

A scanline-based one-pass, optimal algorithm is used to solve the rectangle intersection problem as described in [22]. The problem is reduced to two subproblems, an interval query, and a point query. Interval tree and range tree are two "semi-dynamic" tree data-structures that are used to solve this problem [23]. We shall refer to these two sets of trees as scanline trees. A separate scanline is used for each set M_iV_j but there is a single graph for the entire layout. Both these trees can perform INTERSECTSEARCH,[3] INSERT, and DELETE operations in $O(log(m))$, where m is the number of nodes in the tree [23].

c) *Edge contraction:* All neighboring vertices of the neighborhood graph that correspond to rectangles of the same layer are merged. At the end of this operation, each vertex has an edge only to vertices belonging to an adjacent layer. Hence, a vertex corresponding to Metal 2 in RNG will have edges only to vertices of Via 1 or Via 2 and so on.

d) *Graph analysis:* Floating fill is identified by looking for isolated vertices. Cycles in the RNG correspond to redundant vias that can be identified using depth first search (DFS). Double and even multicut vias can be identified by scanning the reported cycles and identifying the set of vias connected to the same pair of metal-layer vertices in RNG.

2) *Runtime Improvement Techniques.*

a) *Routing-aware scanline:* The routing direction of each set of rectangles, M_iV_j, can be found by taking the larger of the average length and width of all rectangles in the set. If the routing direction is X (Y), we define y (x) coordinates of the rectangles as scanline events so that the average duration for which a rectangle needs to be stored in the tree reduces, thus improving INTERSECTSEARCH time.

b) *Shape simplification:* The complexity of layout features increases tremendously after OPC. This results in a large number of rectangles after fracturing and slows down the algorithm presented in Section 1. Before fracturing the polygons into rectangles, we perform shape approximation on the post-OPC polygons to reduce the number of rectangles created after fracturing. We create two sets of buckets for the coordinates of each polygon. Each point is included in two buckets, one in x-direction and another in y-direction such that x (or y) coordinate of each point in a bucket is within a certain threshold distance of others. All the x (or y) coordinates of a bucket are then changed to the average x (or y) coordinate of the corresponding bucket. This approach reduces small deviations along a straight line as shown in Fig. 3 and hence reduces rectangle count, while preserving connectivity.

Algorithm 1 summarizes the entire algorithm. Fig. 4 illustrates the complete algorithm for a sample double via.

[1] This constraint can be relaxed, if needed.

[2] Although storing each rectangle twice leads to redundant computation, we actually found this method be be faster than storing all the rectangles in a single set due to smaller interval and segment tree size during scan line.

[3] INTERSECTSEARCH returns all rectangles stored in the scanline tree that intersect the input rectangle and constructs edges in the neighborhood graph between the input and all returned rectangles.

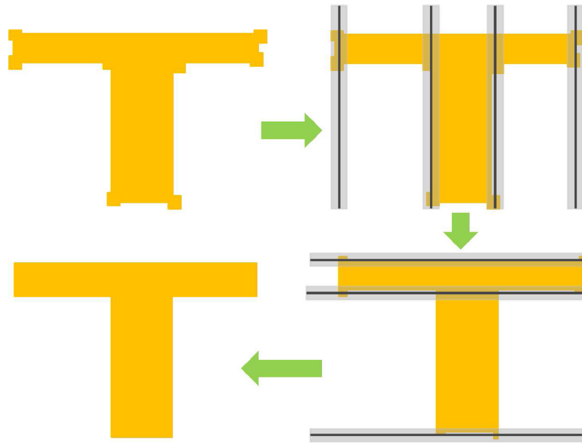

Fig. 3. Shape simplification for a distorted T-shape.

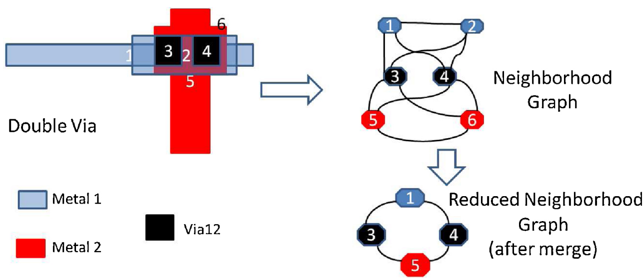

Fig. 4. Illustration of various steps of nonfunctional feature finding.

Algorithm 1 Nonfunctional feature finding

Require: Shapes of all metal and via layers, S.
1: **for all** Shape $s \in S$ **do**
2: SHAPE-SIMPLIFICATION(s)
3: Set of rectangles, B_s = FRACTURE(s)
4: Store B_s in set M_iV_j corresponding to shape layer
5: **end for**
//EVENT DEFINITION
6: Find routing direction R of each rectangle set, M_iV_j
7: **if** Routing direction R is X (Y) **then**
8: Store bottom (left) and top (right) of each rectangle in set as separate events in E_{ij}.
9: **end if**
//SCANLINE
10: **for all** Events $e \in E_{ij}$ for each set E_{ij} **do**
11: **if** e is bottom (left) **then**
12: INTERSECTSEARCH(Scanline tree, $e.rect$)
13: INSERT(Scanline tree, $e.rect$)
14: **else**
15: DELETE(Scanline tree, $e.rect$)
16: **end if**
17: **end for**
//EDGE CONTRACTION
18: Edge Contract $G(V, E)$ to obtain RNG $G(V', E')$
//GRAPH ANALYSIS
19: Mark all isolated vertices as dummy fill
20: Find cycles in $G(V', E')$ using DFS to detect redundant vias

We can now analyze the runtime complexity of our approach. If a layout has N rectangles, then the neighborhood graph construction can be done in $O(Nlog(N)+E)$ time, where E is the number of intersecting pairs of rectangles [22]. The neighborhood graph has N vertices and E edges. We can sort the vertices in $O(Nlog(N))$, and perform edge contraction in $O(E)$. RNG will then have N' vertices ($N' < N$) and E' edges ($E' < E$). N' and E' depend on the particular design layout and the aggressiveness of OPC. We can then perform DFS on the RNG to identify redundant vias and dummy fill in $O(N'+E')$ time. Hence, the overall complexity of our approach is $O(Nlog(N) + E)$.

III. Criticality Assignment

This section focuses on the following problem. Given the timing of critical paths and nonfunctional features identified in Section II, find the minimum-size reticle defect at each location in the layout that can cause failure.

On the basis of geometry, reticle defects are classified as pindots, pinholes, intrusion, and extrusion. Intrusion and extrusion defects are considered CD defects. Pindots and pinholes are usually classified as contamination defects. These two categories of defects are detected using different approaches and hence we treat them separately during criticality assignment. Apart from the size, type, and location of defect, CD impact of a defect on the wafer also depends on the type of reticle (bright-field or dark-field), type of resist (positive or negative), and mask error enhancement factor (MEEF) at the defect location.

Reticle and resist type depends on the mask layer under consideration. MEEF, on the other hand, changes within a mask itself. It is a function of neighborhood mask features and the optical parameters of the lithography system. There are three potential methods of accounting for MEEF in criticality assignment, which we shall explore further in Section VI.

1) Rely on modern inspection tools that support adaptive thresholding, i.e., the threshold value is dynamically changed by the tool depending on online MEEF estimation [3]. In this case, we can choose MEEF=1 since the inspection tool can adjust for it.
2) Find the worst-case MEEF (across process window) for all fragments for each mask shape through lithographic simulation and assign a MEEF value to each mask shape.
3) Find the worst-case MEEF for the entire mask for each layer type and use that value for assigning criticality of every shape of that reticle.

Note that our criticality analysis is focused on binary defects only. Phase defects are not considered since defect data from a commercial mask shop suggests that they are rare.[4] A square approximation is used to model defect shape (similar to most critical area analysis methods).

Details of criticality assignment for different reticle layers is detailed in the the following sections. For the polysilicon layer, we use the timing slack of various paths to assign the minimum size defect for each polysilicon shape corresponding to a transistor pair (PMOS + NMOS) on the critical path.

[4]In data for over 700 reticles, we did not see any phase defects.

TABLE I
GLOSSARY OF TERMINOLOGY USED IN THIS SECTION

Term	Definition
a	Size of square defect
$a_{\min}$	Minimum detectable defect size of the inspection tool
$W_{\min}$	Width design rule of given layer
$S_{\min}$	Spacing design rule of given layer
$Df_{\min}^{\text{CD}}$	Minimum tolerable CD defect
$Df_{\min}^{\text{Con}}$	Minimum tolerable contamination defect

TABLE II
DESIGN IMPACT OF DIFFERENT DEFECT TYPES IN POLYSILICON LAYER

Type	Gate Length	Design Impact
Intrusion	Decrease	Open/delay decrease
Extrusion	Increase	Short/delay increase
Pinhole	Decrease	Open
Pindot	No change	None

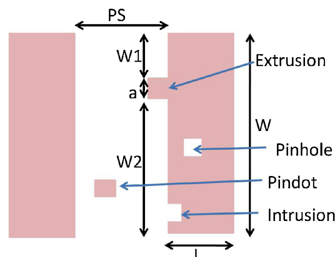

Fig. 5. Illustration of various defect types on polysilicon layer.

Since defects are very small compared to layout shapes, we assume that their impact on parasitic or coupling capacitance is negligible for all layers. Minor change in dimensions of back-end layer shapes do not affect circuit metrics like delay or power, as shown in [24]. Hence, prevention of opens or shorts is the only concern for assigning criticality to back-end layer reticles. As a result, we only utilize location of redundant vias and dummy fill to assign minimum size defect for via and metal layers, respectively. For all our analysis, we assume that assigning a tolerable defect size of 20%, the minimum width/space design rule is sufficient to prevent shorts or opens.[5] We also assume that a single layout shape is not affected by more than one defect since mask defect density is typically very low (order of few tens of defects for a full reticle). Table I lists the notation used in this section.

A. Polysilicon Layer

Polysilicon layer printing typically uses bright-field masks with positive photoresist. The impact of different reticle defect types is illustrated in Fig. 5 and summarized in Table II.

Since extrusion defects can cause timing failure, we must estimate the minimum size of an extrusion defect that can cause timing failure. Consider an extrusion defect as shown in Fig. 6. In order to estimate the delay change caused by this defect, the transistor is sliced into three parts as shown in Fig. 6 (similar to [24]) to estimate the effective $\frac{W}{L}$ as shown in (1). Using first-order transistor models, we can then estimate the change in drive current caused by this defect, which is then used as a pessimistic approximation of change in cell delay as shown in (2). Assuming that at most K defects lie on a critical path,[6] we can evaluate the minimum defect size that changes the delay of each affected transistor by less than T_{slack}/K and hence ensure timing correctness of the path. Here, the timing slack must be obtained from a timing report that designers need to pass on to mask shops. The maximum tolerable defect size, a_{critical} can then be estimated as shown in (3), with an additional guardband of α_{cycle} to allow for other sources of variation

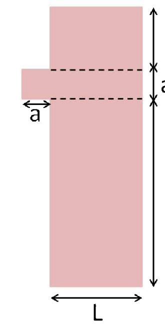

Fig. 6. Illustration of various defect types on polysilicon layer.

$$\left(\frac{W}{L}\right)_{\text{new}} = \frac{W1}{L} + \frac{W2}{L} + \frac{a}{L-a} \quad (1)$$

$$\frac{\triangle\text{Delay}}{\text{Delay}_{\text{nom}}} = -\frac{\triangle\frac{W}{L}}{\frac{W}{L}} = \frac{a^2}{WL} \quad (2)$$

$$a_{\text{critical}} = a_{\min}, \quad \text{if} \quad T_{\text{slack}} < \alpha_{\text{cycle}}$$
$$= \sqrt{\frac{(T_{\text{slack}} - \alpha_{\text{cycle}})/K}{\text{Delay}_{\text{nom}}}WL}, \quad \text{otherwise.} \quad (3)$$

To guardband against process variations downstream, we set the minimum defect size as 20% the width (opens) and spacing (shorts) dimensions. We assume that pinholes do not have any parametric impact and can only cause an open if they are bigger than the gate length. Hence, we can assign the minimum size of CD defects and contamination defects for any polysilicon feature as shown in (4) and (5), respectively

$$Df_{\min}^{\text{CD}} = \frac{\min(0.2W_{\min}, 0.2S_{\min}, a_{\text{critical}})}{\text{MEEF}} \quad (4)$$

$$Df_{\min}^{\text{Con}} = \frac{0.2W_{\min}}{\text{MEEF}}. \quad (5)$$

B. Active Layer

The potential impact of any active defect is determined by the location of the defect relative to an overlapping polysilicon or contact shape as shown in Fig. 7.

Active layer is usually patterned using bright-field masks with positive photoresist. Hence, we can summarize the design impact of any defect on an active layer reticle as shown as Table III. Note that an intrusion defect on the active layer

[5] For simplicity and pessimism, we use minimum DR rules instead of using exact design values.

[6] A critical path typically consists of only 20–50 transistors and hence the area occupied by a critical path is very small compared to the area of the chip. We take $K = 10$ as a pessimistic value.

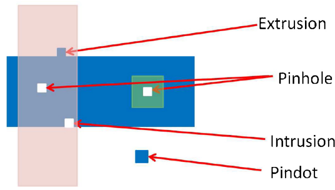

Fig. 7. Illustration of various defect types on active layer.

TABLE III
DESIGN IMPACT OF DIFFERENT DEFECT TYPES IN ACTIVE LAYER

Type	Design Impact
Intrusion	Delay increase
Extrusion	Active short
Pinhole	Contact/polysilicon open
Pindot	No impact

reticle can cause change in delay of a transistor if it lies on the overlap area with poly. Although the exact analysis would require technology computer-aided design (TCAD) simulations, we make the pessimistic assumption that an intrusion defect of size a reduces the transistor width by the same amount. Hence, we can calculate the maximum defect size that does not cause timing failure using a similar analysis as that of the polysilicon layer, as shown in the following equations:

$$\left(\frac{W}{L}\right)_{\text{new}} = \frac{W-a}{L} \tag{6}$$

$$\frac{\triangle\text{Delay}}{\text{Delay}_{\text{nom}}} = W\left(\frac{1}{W-a} - \frac{1}{W}\right) \tag{7}$$

$$\begin{aligned} a_{\text{critical}} &= a_{\min}, \quad \text{if} \quad T_{\text{slack}} < \alpha_{\text{cycle}} \\ &= W\left(\frac{T_{\text{slack}} - \alpha_{\text{cycle}}/K}{\text{Delay}_{\text{nom}}}\right), \quad \text{otherwise}. \end{aligned} \tag{8}$$

Pindot defects have no impact but pinhole defects can cause an open contact or a malfunctioning transistor.[7] The maximum tolerable pinhole defect size is, therefore, determined by poly/contact design rules since it can cause an open contact or transistor. Extrusion defects do have not a significant impact unless they cause a short with another active shape.

Based on the above analysis, we can assign maximum acceptable defect size for CD and contamination defects as follows:

$$Df_{\min}^{\text{CD}} = \frac{\min(0.2W_{\min}, 0.2S_{\min}, a_{\text{critical}})}{\text{MEEF}} \tag{9}$$

$$Df_{\min}^{\text{Con}} = \frac{\min(0.2W_{\min}^{\text{poly}}, 0.2W_{\min}^{\text{contact}})}{\text{MEEF}}. \tag{10}$$

C. Metal Layer

Dark-field masks with positive resist are typically used to make trenches for depositing copper (dual damascene process). The impact of various types of defects is shown in Table IV.

[7]Modeling the delay change on a transistor due to a pinhole defect on the transistor also requires more elaborate TCAD-based simulation study that is not dealt with in this paper.

TABLE IV
DESIGN IMPACT OF DIFFERENT DEFECT TYPES IN METAL LAYER

Type	Wire Width	Design Impact
Intrusion	Increase	Short
Extrusion	Decrease	Open
Pinhole	No change	None
Pindot	Decrease	Resistance change

TABLE V
DESIGN IMPACT OF DIFFERENT DEFECT TYPES IN VIA/CONTACT LAYER

Type	Via Width	Design Impact
Intrusion	Increase	Short
Extrusion	Decrease	Open/resistance increase
Pinhole	None	Metal short
Pindot	Decrease	Resistance increase

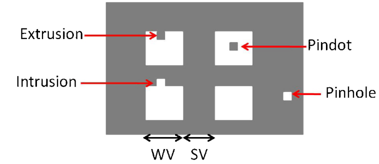

Fig. 8. Illustration of different defect types on via layer.

Small changes in back-end layers are known to have little impact on timing [24]. Hence, we focus only on opens and shorts for assigning criticality to metal-layer shapes. Dummy fill do not have any design impact and can be assigned a relaxed defect size tolerance for both CD and contamination defects. Hence, for a nondummy metal-layer feature, minimum defect size for CD defects and contamination defects can be assigned as follows:

$$Df_{\min}^{\text{CD}} = \frac{0.2\min(W_{\min}, S_{\min})}{\text{MEEF}} \tag{11}$$

$$Df_{\min}^{\text{Con}} = \frac{0.2S_{\min}}{\text{MEEF}}. \tag{12}$$

D. Contact and Via Layer

Dark-field masks with positive resist are typically used to print via layer. Impact of various defect types on via layer is summarized in Table V and shown in Fig. 8.

Similar to metal layers, we assume that the impact of mask defects on electrical metrics is negligible and we only consider opens and shorts while assigning criticality. Note that regions where the nonfill shapes on adjacent metal layers overlap must be assigned minimum detectable defect size of the inspection tool for contamination defects since even the smallest pinhole defect could cause a short. Similar to the other layers, 20% change in via area is taken as the constraint to assign defect size for CD and contamination (pindot) defects. Redundant vias will have a larger tolerance for defects. We can write the minimum size defects for a set of mXn redundant vias

($m = 1, n = 1$ for single via) as follows:

$$Df_{\min}^{\mathrm{CD}} = \frac{0.2\max(m,n)\min(W_{\min}, S_{\min})}{\mathrm{MEEF}} \quad (13)$$

$$Df_{\min}^{\mathrm{Cont}} = \frac{0.2\max(m,n)W_{\min}}{\mathrm{MEEF}} = a_{\min} \text{ for metal intersect regions.} \quad (14)$$

IV. Modeling the Inspection Process

In this section, we develop a model for two key inspection tool properties: resolution and defect count. Defect count of the inspection tool is subdivided into false defects, nuisance defect, and line edge roughness (LER) defects. All these properties are modeled in terms of pixel size and sensitivity, which are the two key-tunable parameters for mask inspection. In addition to this, we develop a model to estimate FPY, which is a key metric that determines mask cost.

A. Resolution

The resolution of any digital imaging system scales linearly with pixel size. Also, increasing the sensitivity helps in detecting smaller features. Hence, for an inspection with pixel size, p and sensitivity S, we shall model resolution as shown in (15) for both CD and contamination defects. Current inspection tools are capable of inspecting a 20-nm defect (on the mask), which corresponds to 5 nm on the wafer (MEEF=1) at a pixel size of 55 nm and sensitivity of 100 [3]. Hence, we take $K_c \approx 9$ for our experiments[8] as follows:

$$R_{\min} = K_c \frac{p}{S}. \quad (15)$$

B. Defect Model

1) *False Defects:* Due to the presence of random temporal noise,[9] the intensity falling on each pixel of the inspection tool sensor during image capture can be modeled as a Gaussian random variable is shown as follows:

$$p(I) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(I-I_m)^2}{2\sigma^2}} \quad (16)$$

where $p(I)$ is the probability of the intensity value being equal to I, I_m is the average intensity at the pixel under consideration, and σ is the temporal noise [25].

Now, suppose die-to-database inspection is done with optimum biasing settings such that intensity at each pixel of the reference database is equal to mean intensity of the corresponding mask. Let us assign the threshold for intensity as T, i.e., any pixel is labeled as a defect if $|I - I_m| > T$. Hence, the probability of a particular pixel being labeled as defective due to the Gaussian noise is given by

$$P(\text{defect}) = 1 - \int_{\mathrm{I_m - T}}^{\mathrm{I_m + T}} \mathrm{p(I)dI} = 2\mathrm{erfc}\left(\frac{\mathrm{T}}{\sqrt{2}\sigma}\right). \quad (17)$$

[8] K_c is slightly different for CD and contamination types of defects but we assume a constant value for simplicity.

[9] We assume that fixed point noise sources can be compensated for by post-capture image processing.

The various components of temporal noise in a typical charge-coupled device sensor are reset noise, shot noise, and read noise. Reset noise is typically compensated by correlated double sampling. The most critical component of noise is shot noise, comprising dark current, and photon shot noise [25]. We assume that the inspection system is photon noise limited. Photon noise is caused due to the randomness in the number of photons exposed to each pixel. The number of photons falling on a pixel follows a Poisson distribution [25]. Hence, we can model the noise σ as follows:

$$\sigma = \text{Noise} = K\sqrt{N_{\mathrm{sig}}} = K_n\, pix \quad (18)$$

where N_{sig} is the number of photons falling on a pixel, K, K_n are constants, and pix is the pixel size of the sensor used in the inspection tool.

Apart from changing the pixel size for inspection, mask engineers can also adjust sensitivity that is related to the threshold for detection of defects. Increasing sensitivity corresponds to reduction of threshold and greater false defect count. For simplicity, we assume that sensitivity is inversely proportional to the threshold. The value of threshold also depends on the background intensity that falls on each pixel, which is proportional to the pixel area. Hence, $T = K_t pix^2/S$, where S is the sensitivity used for inspection. Using this, we can estimate number of false defects as follows:

$$\text{False defects} = K_a \frac{A}{pix^2} erfc\left(\frac{K_t pix^2/S}{\sqrt{2}K_n pix}\right) = K_a \frac{A}{pix^2} erfc\left(K_m \frac{pix}{S}\right). \quad (19)$$

Now since CD and contamination defects are flagged using different algorithms whose sensitivity can be set independently [3], we calculate false defects reported by the two methods separately to get the overall false defect count of the inspection tool as follows:

$$\text{False defects} = \frac{A}{pix^2}\left(K^{\mathrm{CD}} erfc\left(K_m^{\mathrm{CD}} \frac{pix}{S^{\mathrm{CD}}}\right) + K^{\mathrm{Con}} erfc\left(K_m^{\mathrm{Con}} \frac{pix}{S^{\mathrm{Con}}}\right)\right) \quad (20)$$

where K^{CD}, K^{Con}, K_m^{CD}, and K_m^{Con} are constants that depend on the inspection tool.

We used a commercial mask shop's inspection data from over 800 reticles with inspection area ranging from 8000 to 15 000 mm^2, pixel size ranging from 72 to 250 nm, and sensitivities ranging from 75 to 100 to fit these parameters to get $K^{\mathrm{CD}} = 30.33$, $K^{\mathrm{Con}} = 34.21$, $K_m^{\mathrm{CD}} = 0.071$, and $K_m^{\mathrm{Con}} = 12.99$ if the inspected area is taken in mm^2 and the pixel size in nm.

2) *Nuisance Defects:* The number of nuisance defects depends on the design and the total number of real defects, which are the non-nuisance defects. Assuming that the defect distribution for a reticle follows the same negative binomial distribution as wafer defects,[10] we can derive a model for the

[10] Though there is no published study of reticle defect distribution (to the best of our knowledge), the similarity of mask writing process to wafer patterning suggests a similar defect distribution.

total number of real defects for a reticle of area A, inspected with pixel size p and sensitivities S^{CD} and S^{Con} using

$$\begin{aligned}\text{Real defects} &= A \times \sum_{\text{Defect types}} \int_{R_{\min}}^{\infty} \frac{K_2}{D^{\beta}} dD \\ &= A \times \sum_{\text{Defect types}} \frac{K_2}{\beta - 1}\left(K_c \frac{p}{S}\right)^{\beta-1} \\ &= A \times T^{CD}\left(\frac{p}{S^{CD}}\right)^{\beta^{CD}-1} \\ &+ A \times T^{\mathrm{Con}}\left(\frac{p}{S^{\mathrm{Con}}}\right)^{\beta^{\mathrm{Con}}-1}. \end{aligned} \tag{21}$$

The constants were fitted using the same mask shop data used to fit false defects to obtain $T^{\mathrm{CD}} = 0.0002555$, $\beta^{\mathrm{CD}} = 1.3$, $T^{\mathrm{Con}} = 0.00008208$, and $\beta^{\mathrm{Con}} = 0.88$. Note that this measure of real defects considers both critical and nuisance defects.

3) *LER Defects:* LER has become a major source of noise reported by inspection tools. Although LER can be considered a part of nuisance defects since they are actually present on the mask, the key difference from the analysis of nuisance defects above is that LER is a spatially high frequency source of noise that affects only the edges of features instead of the entire reticle. Hence, a model for LER depends on the total perimeter of all the polygons in a reticle pattern. If the total perimeter of all the polygon edges is P, the total number of pixels at the edge is P/pix and we can model every edge as an independent and identically distributed Gaussian distribution as shown in (22), where L_0 is the average edge location and σ_{LER}, the variance of the edge position, is the LER parameter that is determined by the manufacturing process. Equation (23) essentially gives the number of LER defects by multiplying the number of pixels on polygon edges to the probability of an edge deviating beyond the inspection resolution $R_{\min}$. Equation (24) gives a closed-form expression for the number of LER defects reported by the inspection tool. Since we do not have any industrial data for this defect type, we will ignore these in our experiments

$$L = \frac{1}{\sqrt{2\pi}\sigma_{\mathrm{LER}}} exp(-\frac{(L - L_0)^2}{2\sigma_{\mathrm{LER}}^2}) \tag{22}$$

$$\text{LER defects} = 2\frac{P}{pix}\int_{L_0+R_{\min}}^{\infty} L.dL \tag{23}$$

$$\text{LER defects} = 2\frac{P}{pix}(erfc\left(\frac{R_{\min}}{\sqrt{2}\sigma_{\mathrm{LER}}}\right). \tag{24}$$

C. First Pass Yield

FPY of masks is defined as the number of masks that can be passed without any repair or review. This is the most important metric for mask shops as it strongly dictates the manufacturing cost of masks. In this section, we develop a simple critical area-based methodology to estimate FPY.

Let us assume that the defect distribution on the mask is $P(r)$, which corresponds to the probability of a defect to be of size r. Let the maximum defect size on the mask be x_M. Let us also assume that the spatial distribution of defects on the mask is uniform. If a mask area A_M was inspected at a single resolution R, then the probability that a given defect will be detected by the inspection tool is given by P_{dd} in (25). Assuming $P(r)$ to be inversely proportional to r^3 (similar to wafer defect distribution) [26] and $x_M \to \infty$, we can calculate P_{dd} as shown in (26) and (27). We can then calculate the expected number of defects, N_d, and consequently the expected number of detected defects, N_{dd}, as shown in (28) and (29), respectively, where D_{avg} is the average number of defects on the reticle. Assuming that the number of potential defect sites are very large, we can treat the number of detected defects as a Poisson distribution with expected value N_{dd}. Hence, the probability of detecting no defects is given by (30).

$$P_{\mathrm{dd}} = \int_{R}^{x_M} P(r).dr \tag{25}$$

$$P(r) = \frac{K}{r^3} \tag{26}$$

$$P_{\mathrm{dd}} = \frac{K}{2}\frac{1}{R^2} \tag{27}$$

$$N_d = D_{\mathrm{avg}} A_M \tag{28}$$

$$N_{\mathrm{dd}} = P_{\mathrm{dd}} N_d \tag{29}$$

$$P_Y^{A_M} = \exp^{-N_{\mathrm{dd}}}. \tag{30}$$

If the total mask area is partitioned into N regions of area $A_{M1}, A_{M2}, \ldots, A_{Mk}$, each of which is inspected at a different pixel size and sensitivity (different resolution), then the FPY of the full mask can be expressed as a product of the probabilities of no defect detection from any partition as follows:

$$\text{First pass yield} = \prod_{k=1}^{k=N} P_Y^{A_{Mk}}. \tag{31}$$

V. Partitioning

In this section, we present a method to partition the reticle where each partition is assigned a pixel size and sensitivity such that the number of false and nuisance defects reported by the inspection tool are minimized without missing any critical defects.

We wish to perform inspection of different regions of the reticle at different pixel size and sensitivities. When we perform inspection for a particular pixel size, partitions marked for inspection at a different pixel size must be labeled as do not inspect regions (DNIRs) during the current pixel size inspection. DNIR rules specify that a DNIR can be as small as one pixel but there is a 40 pixel band in each direction that is not inspected. For our partitioning problem, this essentially means that a partition must have dimensions of at least 80 pixels (recall that multiple pixel sizes are implemented as multiple scans with DNIRs). For simplicity, we assume the same partition for both pixel size and sensitivity and use the largest pixel size in our experiments to define minimum dimension of a partition.

Hence, the design-aware reticle partitioning problem can be formally stated as follows.

Given a reticle with minimum size defect for each feature, create a partitioning such that a partition j of width W_j and height H_j is assigned a pixel size p_j, and sensitivities S_j^{CD}, S_j^{Con} such that the following function is minimized:

$$F = \text{False defects} + \gamma_1 \text{Real defects} + \gamma_2 \text{LER defects} \tag{32}$$

and the following constraints are obeyed:

1) minimum dimension constraint:

$$\min(W_j, H_j) > L_{\min}; \quad (33)$$

2) for any feature with minimum size defect D_{CD} and D_{Con} lying in the jth block of the partition:

$$D^{\text{CD}} > K_c \frac{p_j}{S_j^{\text{CD}}}, \quad D^{\text{Con}} > K_c \frac{p_j}{S_j^{\text{Con}}} \quad (34)$$

where γ_1 and γ_2 are weighting factors of the cost function and $L_{\min}$ is the minimum dimension constraint.

We use a recursive partitioning heuristic to reduce the defect count metric. At any iteration, if we have k partitions, we find an optimal vertical or horizontal split line that minimizes the defect count for each of the k partitions. Computing the false/nuisance defect cost for any partition requires scanning the entire partition to find the feature with the minimum defect size assigned. This value of tolerance dictates the resolution for inspection. We then pick the pixel size, sensitivity option that minimizes the cost while keeping the inspection resolution equal to the minimum defect size. Note that splitting a partition can never decrease the total defect count. If both new partitions after splitting have the same value of minimum tolerance then the cost remains the same. If one of them has a higher tolerance then it can be inspected at a lower resolution that would reduce false/nuisance defect count. If a partition has reached the minimum dimension constraint ($L_{\min}$) or no split line reduces the cost then we mark that partition as "optimized" and do not analyze it in any future iterations. This step helps to improve runtime. To locate the optimal split line for any partition, we exhaustively search all the potential horizontal and vertical lines at increments of $L_{\min}$ and pick the line that minimizes the cost. The algorithm terminates when all partitions have been labeled as "optimized" or a fixed maximum number of iterations have been reached. The overall algorithm has been summarized in Algorithm 2.

Algorithm 2 Design-aware reticle partitioning for inspection

Require: Criticality value of each shape of reticle, minimum dimension constraint $L_{\min}$.
1: Define partition array P.
2: Initialize P with one partition, the full reticle.
3: **while** $iter <$ MAX-ITER AND $\text{num}_{\text{opt}} < \text{size}(P)$ **do**
4: **for all** $p_i \in P$ **do**
5: **if** p_i NOT "optimized" **then**
6: Find minimum cost split line SL for pi that reduces cost by ΔCost.
7: **if** $\Delta\text{Cost} > 0$ **then**
8: Partition p_i using SL to get new partitions p_{iA} and p_{iB}.
9: Insert p_{iA} and p_{iB} into P.
10: **else**
11: p_i is "optimized."
12: **end if**
13: **end if**
14: **end for**
15: Count number of "optimized" partitions in P, num_{opt}.
16: $iter++$.
17: **end while**

TABLE VI
RECTANGLE COUNT BEFORE AND AFTER SHAPE SIMPLIFICATION FOR ALL LAYERS

Design Name	No. of Gates	Area (μm^2)	No. of Rectangles (Before)	No. of Rectangles (After)
Mips	19 983	15 947	2 582 260	1 591 124
AESCipher	11 395	19 678	2 893 906	1 850 315
Nova	62 800	169 628	20 621 302	13 626 203

TABLE VII
NONFUNCTIONAL FEATURE FINDING RESULTS

Design Name	No. of Double Vias	No. of Dummy Fill	Runtime (min)	Memory Usage (MB)
Mips	23 562	20 040	3	1212
AESCipher	24 267	5308	683	1143
Nova	156 774	144 727	135	5888

VI. EXPERIMENTAL RESULTS

A. Nonfunctional Feature Finding

We implement our neighborhood graph-based algorithm to identify redundant vias and dummy fill in C++ using OpenAccess (OA) API [27]. Layouts of some benchmark circuits implemented in 45-nm Nangate OpenCell library along with the insertion of double vias and dummy fill was done in Cadence Encounter [28]. OPC was performed on the generated GDSII files using Mentor Calibre [29]. All the implementation was done on a 2.0 GHz Intel Xeon machine with 4 GB memory.

The size of the post-OPC benchmark layouts that we considered along with improvement in the rectangle count due to shape simplification is shown in Table VI. The threshold for bucketing was taken as 20 nm, which is less than the minimum metal width for 45-nm FreePDK [30]. Around 50% reduction in number of shapes is observed for the three benchmark circuit layouts considered.

Table VII summarizes the results of redundancy finding for all the layers. Note that the benchmark design AESCipher takes a long time despite the small number of rectangles. This is because the design is heavily congested and has a very large number of edges in the neighborhood graph. Runtime can be easily improved by partitioning the layout into smaller blocks and using a separate graph for each region. The algorithm can also be parallelized easily by running the critical graph construction step for each set $M_i V_j$ in parallel. These techniques are left for future work.

The number of redundant vias and dummy fill reported by our approach are verified with the number obtained from DEF file of the corresponding design. Double vias are reported with 100% accuracy by our approach and there is less than 1% error in dummy fill due to some outliers.

Table VIII shows the percentage noncritical regions for the benchmarks that indicates the potential benefits that can be derived from design-aware inspection of metal and via layers. For contact and via layers, we mention the total number of

TABLE VIII
LAYER-BY-LAYER NONCRITICAL REGIONS

Design	Via Layer	No. of Vias	No. of Redundant	Metal Layer	% Dummy Area
Mips	Contact	171 272	6	Metal1	3.47
	Via1	30 737	3140	Metal2	26.54
	Via2	40 715	29 592	Metal3	26.68
	Via3	15 731	9874	Metal4	42.29
	Via4	7666	3056	Metal5	53.37
	Via5	2811	1468	Metal6	81.69
AESCipher	Contact	190 230	0	Metal1	0.00003
	Via1	46 857	8987	Metal2	4.0
	Via2	52 950	27 461	Metal3	4.9
	Via3	27 872	10 641	Metal4	11.8
	Via4	15 854	2022	Metal5	14.3
	Via5	10 287	1650	Metal6	31.9
Nova	Contact	1 399 817	62	Metal1	0.0009
	Via1	237 337	22 878	Metal2	19.97
	Via2	300 009	206 249	Metal3	19.69
	Via3	104 190	58 682	Metal4	43.14
	Via4	39 005	17 708	Metal5	58.54
	Via5	14 897	8054	Metal6	83.63

contact/vias along with the number of redundant. Contact layer has redundancy only at the standard cell level. Since very few Nangate cells have redundant contacts, the number of redundant contacts is very low. For metal layers, dummy area is reported as a percentage of the total die area. Higher metal layers typically have less congestion after routing and hence have a greater percentage of dummy area. Note that we have not considered active or polysilicon fill and hence the criticality assignment of polysilicon and active layers is based on the slack of timing critical paths only.

B. Criticality Assignment and Reticle Partitioning

For assigning minimum size defect to each layout feature, we use 45-nm design rules from FreePDK [30]. Location of dummy fill and redundant via was used for metal and via layers, respectively. For polysilicon and active layers, we need slack values that was obtained from the timing analysis of the postrouted design using Cadence Encounter [28]. The criticality assignment method assumes MEEF=1 unless otherwise stated. The tolerance guardband, α_{cycle}, used for criticality assignment of polysilicon and active layers is taken as 1% of the design cycle time.

Using the criticality assignment, reticle partitioning was implemented in C++ using OA API [27]. From the fitting results of false and nuisance defects, we found that the false defect count is typically at least 10× the number of nuisance defects. But nuisance defects are more important to mask shops as they help improve FPY. Hence, we took $\gamma_1 = 10$ in the cost function for these experiments. Since we did not have any data for LER defects, we take $\gamma_2 = 0$. Only two pixel sizes, 72 nm and 90 nm, were used in our experiments. The minimum dimension constraint was taken as $2\,\mu m$, which is slightly larger than the dimension of 80 pixels at 90 nm pixel size.

We tested our partitioning algorithm for poly, active, contact, and all the back-end layer reticles for the same three designs for which the nonfunctional features have been reported. Since the designs we consider are very small compared to real reticle sizes, we find out the number of copies of these benchmark designs that can fit on an industrial reticle

TABLE IX
IMPROVEMENT IN DEFECT COUNT AFTER PARTITIONING

Design Name	Layer	Before		After		Runtime
		# False	# Real	# False	# Real	(s)
Mips	Polysilicon	70.59	5.52	39.95	3.48	24
	Active	70.59	5.52	38.16	2.37	17
	Contact	66.60	3.36	60.44	3.05	157
	Via1	66.60	3.36	56.78	2.95	64
	Via2	65.94	3.17	57.84	2.83	82
	Via3	65.94	3.17	47.90	2.46	68
	Via4	56.83	2.07	30.86	1.25	30
	Via5	56.83	2.07	14.02	0.65	30
	Metal1	42.62	2.80	42.62	2.80	18
	Metal2	42.20	2.57	39.37	2.40	30
	Metal3	42.20	2.57	40.40	2.46	15
	Metal4	36.37	1.14	32.38	1.02	8
	Metal5	36.37	1.14	32.56	1.02	8
	Metal6	36.37	1.14	22.55	0.72	8
AESCipher	Polysilicon	74.95	5.83	52.60	4.44	42
	Active	74.95	5.83	60.72	4.44	22
	Contact	70.71	3.54	70.71	3.54	129
	Via1	45.24	2.98	44.06	2.91	26
	Via2	44.76	2.71	44.48	2.71	36
	Via3	44.76	2.71	42.54	2.57	20
	Via4	60.30	2.22	33.31	1.04	10
	Via5	60.30	2.22	28.73	0.90	8
	Metal1	45.24	2.98	45.24	2.98	67
	Metal2	44.76	2.71	44.76	2.71	33
	Metal3	44.76	2.71	44.76	2.71	41
	Metal4	38.58	1.18	38.24	1.18	22
	Metal5	38.58	1.18	38.10	1.18	15
	Metal6	38.58	1.18	35.39	1.11	15
Nova	Polysilicon	73.86	5.78	44.77	3.90	2132
	Active	73.86	5.78	45.74	2.98	801
	Contact	69.69	3.51	67.61	3.40	7684
	Via1	69.69	3.51	42.02	2.76	1217
	Via2	44.16	2.68	42.08	2.56	1000
	Via3	68.99	3.32	35.19	2.15	542
	Via4	59.46	2.17	19.05	0.65	272
	Via5	59.46	2.17	8.89	0.36	548
	Metal1	44.60	2.93	44.60	2.93	2444
	Metal2	44.16	2.68	43.06	2.62	981
	Metal3	44.16	2.68	43.38	2.63	509
	Metal4	38.05	1.19	35.78	1.13	236
	Metal5	38.05	1.19	33.61	1.06	141
	Metal6	38.05	1.19	22.03	0.70	195

($104\,mm \times 132\,mm$). Since defect count is proportional to inspection area, the false and real defect count of one design layout are scaled by the number of copies of the design on a full field reticle in order to demonstrate the potential benefits of our approach for a full field reticle. Table IX shows these results. The false and real defect count after partitioning is compared to the case where inspection is done at a single value of pixel size and CD and contamination sensitivities for the entire reticle.

The results of Table IX demonstrate the reduction of false and real defects with our design-aware inspection methodology. Note that the improvement in the real defect count is due to the reduction in nuisance defects only as the partitioning method is constrained to not miss any critical defects. Highly congested layers with very few noncritical features such as the lower metal/via layers show little benefit of design-aware inspection since most areas of the reticle are very critical. Polysilicon and active layers show significant improvement due to different timing criticality of various features. Higher via and metal layers, which have a significant amount of redundancy as shown in Table VIII, show up to 4× improvement

TABLE X
COMPARISON OF PIXEL SIZE AND SENSITIVITY (P+S) PARTITIONING VERSUS SENSITIVITY-ONLY (S) PARTITIONING

Design Name	Layer	P+S Partitioning		S Partitioning	
		# False	# Real	# False	# Real
Mips	Polysilicon	39.95	3.48	62.34	3.48
	Active	38.16	2.37	58.82	2.37
	Contact	60.44	3.05	60.44	3.05
	Via1	56.78	2.95	57.75	2.95
	Via2	57.84	2.83	59.47	2.83
	Via3	47.90	2.46	49.98	2.46
	Via4	30.86	1.25	33.97	1.25
	Via5	14.02	0.65	17.05	0.65
	Metal1	42.62	2.80	66.60	2.80
	Metal2	39.37	2.40	61.52	2.40
	Metal3	40.40	2.46	63.13	2.46
	Metal4	32.38	1.02	50.59	1.02
	Metal5	32.56	1.02	50.87	1.02
	Metal6	22.55	0.72	35.23	0.72

in false and real defect count. Note that the metal/via layer processing does not require any explicit timing information while the polysilicon layer leverages it heavily.

Inspection tools typically allow inspection of different mask regions at different sensitivities in a single scan of the mask. But inspection at different pixel sizes implies that the reticle needs to be scanned multiple times. Hence, it is important to evaluate the additional benefit achieved by varying the pixel size over the reticle area. Table X compares the false and nuisance defect count after partitioning using both pixel size and sensitivity as parameters versus using sensitivity as the only tunable parameter (pixel size taken as 72 nm). The results show that pixel size is a significant knob and using it might be worthwhile despite the need for multiple scans of the reticle. Note that we retain the minimum partition size constraint for sensitivity-only case even though it is not a strict requirement as too many partitions are impractical to store in the inspection tool.

C. First Pass Yield

Using the formulation for FPY described in Section VI-C, we can estimate the FPY if the entire reticle is inspected at a single resolution and compare it to the design-aware approach of inspecting different regions of the mask at different resolution. A reticle is assumed to yield only when all copies of the design on the full field reticle work. Table XI shows the result of this computation for the three benchmark designs we partitioned. Note that the improvement in FPY correlates well with the reduction in real defect count in Table IX. For example, contact and lower metal/via layers show little improvement in FPY whereas poly, active, and higher via layers show an increase of 30% in FPY in some cases. The only exception to this similarity is the higher metal layers, which have a high FPY even with the conventional approach and hence only a small improvement in FPY. The reason for this is the relaxed design rules of the higher metal layers.

D. Accounting for Nonunity MEEF

Our previous results have assumed that the inspection tool can report MEEF-adjusted defect dimensions [3], hence we used MEEF=1. In this section, we do not rely on this feature

TABLE XI
IMPROVEMENT IN FPY WITH DESIGN-AWARE MASK INSPECTION

Design Name	Layer	FPY Before (%)	FPY After (%)
Mips	Polysilicon	12.74	33.72
	Active	12.74	51.71
	Contact	29.17	32.69
	Via1	29.17	34.00
	Via2	31.21	36.08
	Via3	31.21	40.94
	Via4	42.92	61.81
	Via5	42.92	79.14
	Metal1	49.32	49.32
	Metal2	54.36	56.63
	Metal3	54.36	55.79
	Metal4	85.87	87.31
	Metal5	85.87	87.25
	Metal6	85.87	90.98
AESCipher	Polysilicon	11.22	23.50
	Active	11.22	21.38
	Contact	27.04	27.04
	Via1	47.22	48.19
	Via2	52.36	52.60
	Via3	52.36	54.08
	Via4	40.74	86.65
	Via5	40.74	87.80
	Metal1	47.22	47.22
	Metal2	52.36	52.37
	Metal3	52.36	52.38
	Metal4	85.07	85.19
	Metal5	85.07	85.25
	Metal6	85.07	86.30
Nova	Polysilicon	11.58	29.55
	Active	11.58	41.14
	Contact	27.55	28.62
	Via1	27.55	49.81
	Via2	52.85	54.46
	Via3	29.57	60.01
	Via4	41.27	90.00
	Via5	41.27	93.17
	Metal1	47.73	47.73
	Metal2	52.85	53.69
	Metal3	52.85	53.44
	Metal4	85.26	86.10
	Metal5	85.26	86.87
	Metal6	85.26	91.18

and instead use a single worst-case value of MEEF for each reticle layer during criticality assignment. Computing the MEEF value separately for each feature would be more accurate but due to the minimum partition size limitation of inspection tools, it is unlikely to yield much benefit. Hence, we chose the simplistic approach of applying a single pessimistic MEEF correction for each layer.

In order to compute MEEF, we used Mentor Calibre [29] that can compute MEEF value for each layout fragment separately. Since typical MEEF values are a function of technology node and OPC recipe, we computed the MEEF for different layers of a $10\,\mu$m $\times$ $10\,\mu$m snippet of post-OPC Mips layout and used the worst-case value across all fragments of this snippet layout. MEEF was computed using this approach at two defocus values, 0 nm and 50 nm, and the worst-case value is chosen. The MEEF values for different layers are shown in Table XII. These values are larger than previously reported data from the industry [31]–[33]. This is due to the lack of a well-optimized optical correction recipe that should include SRAF insertion and retargeting. Despite this limitation, these MEEF values can be used to validate the feasibility and

TABLE XII
IMPROVEMENT IN DEFECT COUNT AND FPY AFTER PARTITIONING WHEN MEEF CORRECTION APPLIED

Design Name	Layer	Worst-Case MEEF	Before			After		
			No. of False	No. of Real	FPY	No. of False	No. of Real	FPY
Mips	Polysilicon	2	70.59	5.87	8.73	64.15	5.34	10.91
	Active	2	70.59	5.87	8.73	63.47	4.71	15.39
	Contact	6.5	70.59	5.87	8.73	64.06	5.33	10.94
	Via1	5.5	70.59	5.87	8.73	61.21	5.15	11.70
	Via2	20	70.59	5.87	8.73	63.67	5.31	11.01
	Via3	2.5	70.59	5.87	8.73	53.50	4.56	14.85
	Via4	2	65.94	3.17	31.21	35.80	1.95	50.03
	Via5	1.5	63.30	2.65	36.91	15.62	0.85	74.91
	Metal1	9	70.59	5.87	8.73	70.59	5.87	8.73
	Metal2	6.5	70.59	5.87	8.73	65.85	5.48	10.28
	Metal3	6	70.59	5.87	8.73	67.58	5.63	9.69
	Metal4	9	70.59	5.87	8.73	62.84	5.23	11.41
	Metal5	6.5	70.59	5.87	8.73	63.18	5.26	11.28
	Metal6	4.5	70.59	5.87	8.73	43.76	3.65	22.06

potential benefit of our approach in the presence of high MEEF mask features.

After correcting for the MEEF values of different layers during criticality assignment, results obtained from partitioning are shown in Table XII for a Mips design. Note that the defect count and FPY before partitioning are worse compared to the case when MEEF=1. This is expected since the minimum tolerable defect size across the entire reticle worsens due to larger MEEF values. Despite the pessimistic and high MEEF values, the results demonstrate the benefits of our partitioning-based design-aware inspection.

VII. CONCLUSION

In this paper, we developed a comprehensive design-aware mask inspection flow.

1) We proposed a graph-based algorithm that finds non-functional features (dummy fill and redundant vias) in a post-OPC layout with almost 100% accuracy.
2) We formulated a method to assign a minimum size defect to each feature of a reticle for poly, active, contact, and all the back-end layers.
3) We developed a recursive partitioning algorithm to inspect different regions of the layout with different pixel size and sensitivity and up to 4× reduction in nuisance and false defects was observed along with up to 4× improvement in FPY coming from reduction in nuisance defects.

We also demonstrated the importance of pixel size as a paramater in achieving the full benefit of design-aware inspection. Despite the overhead of additional scans of the reticle for each pixel size, the significantly lower defect count suggested that it is a parameter that needs to be exploited in any design-aware inspection flow.

The design-aware methodology that we proposed can be applied easily by captive mask shops since they have access to the design database. Merchant mask shops would need additional information from their customers in the form of either timing report for front-end layer reticles or the database of all the back-end layers so that redundant and dummy features can be identified. In case mask shops cannot get access to design database and are limited to die–die inspection mode, the criticality partitioning can be done at the design end.

In the future, we plan to test our approach in an actual commercial mask shop and explore the implications of our methodology if all mask layers are not available. Only amplitude defects are dealt with in this paper. We plan to extend this methodology to model the design impact of phase defects as well.

ACKNOWLEDGMENT

The authors would like to thank Prof. A. B. Kahng, University of California at San Diego, San Diego, and Dr. S. Stokowski, KLA-Tencor, Milpitas, CA, for their valuable input and contributions.

REFERENCES

[1] P. Gupta, A. B. Kahng, D. Sylvester, and J. Yang, "A cost-driven lithographic correction methodology based on off-the-shelf sizing tools," in *Proc. DAC*, Jun. 2003, pp. 16–21.

[2] K. Hosono and K. Kato, "PMJ panel discussion overview on mask complexities, cost, and cycle time in 32-nm system LSI generation: Conflict or concurrent?" *Proc. SPIE*, vol. 7122, p. 712207, 2008.

[3] A. Dayal, B. Mu, V. Iyer, P. Lim, A. Goonesekera, and B. Broadbent, "Results from the KLA-Tencor TeraScanXR reticle inspection tool," *Proc. SPIE*, vol. 7122, p. 71223G, 2008.

[4] H. Moribe, T. Bashomatsu, K. Matsumura, A. Uehara, and H. Takahashi, "Improvement of image quality and inspection speed in LM7500 reticle inspection system," *Proc. SPIE*, vol. 7028, p. 70282K, 2008.

[5] L. Pang, A. Lu, J. Chen, E. Guo, L. Cai, and J.-H. Chen, "Enhanced dispositioning of reticle defects for advanced masks using virtual stepper with automated defect severity scoring," *Proc. SPIE*, vol. 5256, pp. 461–473, 2003.

[6] L. Karklin, M. M. Altamirano, L. Cai, K. A. Phan, and C. A. Spence, "Automatic defect severity scoring for 193-nm reticle defect inspection," *Proc. SPIE*, vol. 4346, pp. 898–906, 2001.

[7] T.-Y. Kang, H.-C. Lee, H. Zhang, K. Yamada, Y. Kitayama, K. Kobayashi, and P. Fiekowsky, "Auto-classification and simulation of mask defects using SEM and CAD images," *Proc. SPIE*, vol. 7122, p. 71221F, 2008.

[8] A. C. Dürr, A. M. Zibold, and K. Böhm, "An advanced study for defect disposition through 193-nm aerial imaging," *Proc. SPIE*, vol. 6152, p. 61522M, 2006.

[9] S. Banerjee, P. Elakkumanan, L. Liebmann, and M. Orshansky, "Electrically driven optical proximity correction based on linear programming," in *Proc. ICCAD*, 2008, pp. 473–479.

[10] Q. C. Zhang and P. van Adrichem, "Determining OPC target specifications electrically instead of geometrically," *Proc. SPIE Photomask Technol.*, vol. 6730, p. 67303V, 2007.

[11] S. Teh, C. Heng, and A. Tay, "Design-process integration for performance-based OPC framework," in *Proc. DAC*, Jun. 2008, pp. 522–527.

[12] K. Koike, K. Nakayama, K. Ogawa, and H. Ohnuma, "OPC to reduce variability of transistor properties," *Proc. SPIE*, vol. 6521, p. 65210J, 2007.

[13] P. Gupta, A. B. Kahng, D. Sylvester, and J. Yang, "Performance-driven optical proximity correction for mask cost reduction," *J. Micro/Nanolithography, MEMS MOEMS*, vol. 6, no. 3, p. 031005, 2007.

[14] T. Chan, A. Kagalwalla, and P. Gupta, "Measurement and optimization of electrical process window," *J. Micro/Nanolithography, MEMS MOEMS*, vol. 10, no. 1, p. 013014, Jan.–Mar. 2011.

[15] W. W. Volk, C. Hess, W. Ruch, Z. Yu, W. Ma, L. Fisher, C. Vickery, and Z. M. Ma, "Investigation of smart inspection of critical layer reticles using additional designer data to determine defect significance," *Proc. SPIE*, vol. 5256, pp. 489–499, 2003.

[16] S. Hedges, C. Le, M. Eickhoff, M. Wylie, T. Simmons, V. Vellanki, and J. McMurran, "Novel mask inspection flow using sensitivity control layers (SCL) on the TeraScanHR-587 platform," *Proc. SPIE*, vol. 7122, p. 71221G, 2008.

[17] H. Tsuchiya, M. Tokita, T. Nomura, and T. Inoue, "Die-to-database mask inspection with variable sensitivity," *Proc. SPIE*, vol. 7028, p. 70282I, 2008.

[18] F. A. J. M. Driessen, J. Gunawerdana, Y. Saito, H. Tsuchiya, and Y. Tsuji, "Flexible sensitivity inspection with TK-CMI software for criticality-awareness," *Proc. SPIE*, vol. 7122, p. 712222, 2008.

[19] D. Stoler, W. Ruch, W. Ma, S. Chakravarty, S. Liu, R. Morgan, J. Valadez, B. Moore, and J. Burns, "Optimizing defect inspection strategy through the use of design-aware database control layers," *Proc. SPIE*, vol. 6730, p. 67303L, 2007.

[20] A. Kagalwalla, P. Gupta, C. Progler, and S. McDonald, "Design-aware mask inspection," in *Proc. ICCAD*, Nov. 2010, pp. 93–99.

[21] K. Gourley and D. Green, "A polygon-to-rectangle conversion algorithm," *IEEE Comput. Graphics Applicat.*, vol. 3, no. 1, pp. 31–36, Jan. 1983.

[22] H. Six and D. Wood, "The rectangle intersection problem revisited," *BIT Numeric. Math.*, vol. 20, no. 4, pp. 426–433, 1980.

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: MIT Press, 2001.

[24] T.-B. Chan, R. Ghaida, and P. Gupta, "Electrical modeling of lithographic imperfections," in *Proc. VLSI Design*, Jan. 2010, pp. 423–428.

[25] J. Nakamura, *Image Sensors and Signal Processing for Digital Still Cameras*. Boca Raton, FL: CRC Press, 2005.

[26] S. Levasseur and F. Duvivier, "Application of a yield model merging critical areas and defectivity to industrial products," in *Proc. IEEE Defect Fault Tolerance VLSI Syst.*, Oct. 1997, pp. 11–19.

[27] *Openaccess API* [Online]. Available: http://www.si2.org

[28] *Cadence SOC Encounter*. (2008) [Online]. Available: http://www.cadence.com

[29] *Mentor Calibre*. (2008) [Online]. Available: http://www.mentor.com

[30] *FreePDK 45 nm Design Rules*. (2009) [Online]. Available: http://www.eda.ncsu.edu/wiki/FreePDK45:Contents

[31] I.-S. Kim, S. Suh, S. Jung, E. Lee, Y.-S. Kang, S. Lee, S.-G. Woo, and H. Cho, "Toward DFM: Process worthy design and OPC through verification method using MEEF, TF-MEEF, and MTT," *Proc. SPIE*, vol. 6156, p. 61560L, 2006.

[32] G. Xiao, T. Cecil, L. Pang, B. Gleason, and J. McCarty, "Source optimization and mask design to minimize MEEF in low k1 lithography," *Proc. SPIE*, vol. 7028, p. 70280T, 2008.

[33] S.-H. Choi, A.-Y. Je, J.-S. Hong, M.-H. Yoo, and J.-T. Kong, "MEEF-based correction to achieve OPC convergence of low-k1 lithography with strong OAI," *Proc. SPIE*, vol. 6154, p. 61540P, 2006.

Abde Ali Kagalwalla received the B.Tech. degree from the Indian Institute of Technology, Bombay, India, in 2009, and the M.S. degree from the University of California at Los Angeles, Los Angeles, in June 2011, both in electrical engineering. Currently, he is pursuing the Ph.D. degree from the Department of Electrical Engineering, University of California at Los Angeles.

His current research interests include computer-aided design of very large-scale integrated circuits, semiconductor design-manufacturing interface, lithography, and algorithms.

Puneet Gupta received the B.Tech. degree in electrical engineering from the Indian Institute of Technology Delhi, New Delhi, India, in 2000, and the Ph.D. degree from the University of California at San Diego, San Diego, in 2007.

He is currently a Faculty Member with the Department of Electrical Engineering, University of California at Los Angeles, Los Angeles. He co-founded Blaze DFM, Inc. (acquired by Tela, Inc.), Sunnyvale, CA, in 2004, where he served as a Product Architect until 2007. He has authored over 70 papers, ten U.S. patents, and a book chapter. His current research interests include building high-value bridges across application–architecture–implementation–fabrication interfaces for lowered cost and power, increased yield, and improved predictability of integrated circuits and systems.

Dr. Gupta was a recipient of the NSF CAREER Award, the ACM/SIGDA Outstanding New Faculty Award, the European Design Automation Association Outstanding Dissertation Award, and the IBM Ph.D. Fellowship. He has given tutorial talks at DAC, ICCAD, the International VLSI Design Conference, and the SPIE Advanced Lithography Symposium. He has served on the Technical Program Committee of DAC, ICCAD, ASPDAC, ISQED, ICCD, SLIP, and VLSI Design. He was the Program Chair of the IEEE Design for Manufacturability Workshop in 2009 and 2010.

Christopher J. Progler received the B.S. and M.S. degrees in optics from the Institute of Optics, University of Rochester, Rochester, NY, and the D.Phil. degree in electrical engineering from the University of Texas at Dallas, Dallas, with an emphasis on nonlinear optimization linked to physical processes.

Since March 2004, he has been the Chief Technology Officer with Photronics, Inc., Boise, ID, where he manages its intellectual property, establishing strategies for future intellectual property development, identifying external partnerships, which complement Photronics' core technical position, and new technology-driven market opportunities. He also is the Vice President of Photronics. Prior to joining Photronics as a Chief Scientist, he was the Manager of the Advanced Imaging and Technology Computer-Aided Design, IBM Semiconductor Research and Development Center, East Fishkill, NY. At IBM, his team developed innovative imaging methods, computation tools, and characterization techniques required to support IBM's aggressive lithography roadmap. He is a Visiting Lecturer with the University of Texas at Dallas teaching graduate-level courses in optics and lithography and mentoring research students at multiple universities. Since April 2008, he has been a Scientific Advisory Board Member of Cymer, Inc., San Diego, CA. He has authored more than 70 papers and patents in lithography, optics, and microfabrication. He is recognized throughout the industry for his contributions to the lithographic sciences.

Dr. Progler has served as the Chair of the SPIE Conference on Optical Lithography. He co-founded SPIE's Advanced Microelectronic Manufacturing Symposium. He has organized and co-chaired a new SPIE-sponsored microlithography conference to be held with Photonics Asia in Beijing, China, in 2008. He is an Associate Editor of the *SPIE Journal of Microlithography, Microfabrication, and Microsystems* and is currently under contract for a microlithography book edition. He is a frequently Invited Speaker at industrial events and serves on a number of industry advisory committees. In 2004, he was elected a fellow of SPIE, the International Society of Optical Engineering. His contributions to SPIE Technical Programs are indeed noteworthy.

Steve McDonald is the Director of NanoFab Global Sales and Applications Engineering Department, Photronics, Inc., Boise, ID.