Image Processing Techniques for Wafer Defect Cluster Identification

Chenn-Jung Huang
National Taitung Teachers College

Chi-Feng Wu
Philips Semiconductor

Chua-Chin Wang
National Sun Yat-Sen University

An automatic, wafer-scale, defect cluster identifier finds and marks all defective dies, further automating wafer testing. A prototype tool screens 45,000 wafers, saving a Philips Semiconductor test facility $100,000 in expenses per month.

ELECTRICAL TESTING determines whether each die on a wafer functions as originally designed. But these tests don't detect all the defective dies in clustered defects on the wafer, such as scratches, stains, or localized failed patterns. So instead, five to 10 people visually check wafers and hand mark the defective dies in, or close to, these flawed regions. Although this manual checking prevents many defective dies from continuing on to assembly, it does not detect localized failure patterns—caused by the fabrication process—because they are invisible to the naked eye.

To solve these problems, we propose an automatic, wafer-scale, defect cluster identifier. This software tool uses a median filter and a clustering approach to detect the defect clusters and to mark all defective dies. Our experimental results verify that the proposed algorithm effectively detects defect clusters, although it introduces an additional 1% yield loss of electrically good dies. More importantly, it makes automated wafer testing feasible for application in the wafer-probing stage. Such a test would occur early in the manufacturing process in Figure 1 and avoid the cost of assembling and testing defective dies.

Other researchers have proposed algorithms that detect defects or provide defect density distribution for yield prediction.[1-5] To our knowledge, however, there are no tools that detect defect clusters and automatically mark potentially bad dies during wafer probing.

We base our software tool on an algorithm that works with image processing techniques[6,7] to detect the defect clusters on a wafer. The tool implements the algorithm immediately following wafer probing because failed dies marked in the wafer map file assist in locating the flawed areas. Incorporating this technique during the testing stage eliminates manual operation and permits a more fully automated test process.

Defect cluster detection algorithm

Several process stages make up our algorithm, which we summarize as follows:

1. Read in the wafer map file's data format and convert it into a binary matrix.
2. Apply the median filter to the binary matrix to remove the isolated defective die.

0740-7475/02/$17.00 © 2002 IEEE

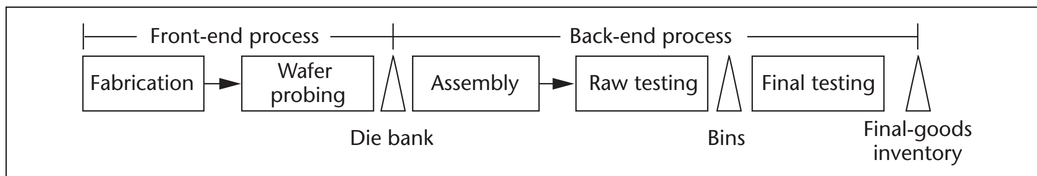

Figure 1. IC manufacturing process. Triangles represent storage of the dies.

3. Perform nearest-neighbor clustering on the binary matrix.
4. Build the linked lists for the defect clusters.
5. Go through the linked lists and mark all the eight-adjacent neighbors as defective dies to compensate for the unexpected erosion from median filter application. (We define two defective dies as eight-adjacent if they share either a side or a corner.)
6. Mark every isolated interior die (a good die surrounded by eight defective dies) in the defect cluster.
7. Generate the output file for further processing.

Wafer map conversion

The basic data format the prober uses is the Semiconductor Equipment Communication Standard (SECS) format developed by the Semiconductor Equipment and Materials Institute (SEMI). The prober's wafer map includes the formatting information defined by SEMI (such as lists, single and multibyte integers, and so on) as well as the map and header data.[8] The header data defines the wafer's orientation during probing. This data consists of a list of individual reference points used to compare the physical wafer to the logical wafer (the wafer map data file), the die size, the total number of rows and columns, and the number of dies to be processed on the wafer. The wafer map's data section records die count in the row, the direction of x-axis travel from one die to the next, and the dies' actual binary codes.

Our tool reads in the wafer map file reported by the prober and transforms it into a bounded matrix with m by n entries, where m denotes the total number of columns on the wafer and n denotes the total number of rows. To simplify processing, our tool converts the binary codes stored in the wafer map into binary values for each matrix entry, where 0 represents a good die and 1, a defective die. This simplification does not differentiate the defective dies in the way the probe instrument does. A wafer map might contain different kinds of defects, and binary codes are assigned to defective dies for these different defects. Although our tool uses this simplified data structure, its *output* wafer map file still follows the SECS standard.

Median filter application

This algorithm separates the wafer's defect clusters from its isolated defective dies. We considered several image filtering techniques for stripping off the isolated defective dies, including Wiener, inverse, spatial-frequency, and median filtering. We also examined wafer map samples with defective-die distribution provided by Philips Semiconductor, Kaohsiung, Taiwan. With respect to the clustered defects, the isolated defective dies on these samples appeared to be salt-and-pepper type noise. Thus, the median filter is a good candidate for the filtering task because of its ability to remove salt-and-pepper type noise.

A median filter finds the median of all the pixels within a region of an image called a *window*. Applying the filter to an image removes random noise with minimum image blurring. Random noise includes negative-exponential and salt-and-pepper type noise. The median filter easily removes outlier noise from images where less than 50% of the pixels are outliers.

To perform median filtering in a pixel's window, we sort both the pixel's and its neighbors' values, determine the median, and finally assign this value to the pixel. That is,

$$\mathbf{out}(x, y) = \text{median}\ \{in(x - m, y - n), (m, n) \in W\}$$

where $in(x, y)$ and $out(x, y)$ are the input and output images, and W is the chosen window.

40	10	20
25	15	20
35	30	20

Figure 2. A 3 × 3 window.

For example, in a 3×3 window, the median is the fifth largest value; in a 5×5 window, it is the thirteenth largest value; and so forth. Assume that a 3×3 window has values 40, 10, 20, 25, 15, 20, 35, 30, and 20, as Figure 2 shows. Sorting of the window's values as 10, 15, 20, 20, 20, 25, 30, 35, and 40, results in a median of 20.

The approach permits for an even simpler median filter design because each matrix entry contains only a 1 or a 0. This design can just count the total number of 1s that appear in the 3×3 neighborhood. Then the pixel's value can be set to 1 if the number of 1s is greater than or equal to five. After application of the median filter, all 1s left in the matrix will be treated as defective dies in the defect cluster. For median filtering, we do not consider edge dies to be part of a defect cluster. So our algorithm temporarily marks these dies as good, so they will bypass the filtering process.

Nearest-neighbor clustering

After the tool removes all isolated defective dies, it uses a region-based segmentation step to group various regions into an image with similar features.[9] Clustering techniques proposed in prior pattern recognition literature[10] have similar objectives, so we can apply these techniques to segmentation. We adopted the nearest-neighbor method to identify defective dies of different defect clusters.[10] We implemented this method by defining the distance between two defect clusters as the shortest distance between two defective dies, where one defective die is in each defect cluster. If S_i and S_j are two defect clusters, the distance between them is

$$Dist(S_i, S_j) = \min_{a \in S_i,\, b \in S_j} d(a, b) \geq 2$$

where $d(a, b)$ denotes the Hamming distance between defective dies a and b.

Suppose that the wafer contains N defective dies at the initial state and each forms an individual cluster. We merge any two clusters together if the distance between them is less than two. The merging process proceeds sequentially until the distance between the closest neighbors of different clusters surpasses one.

Building linked lists

After applying the nearest-neighbor algorithm, the algorithm constructs linked lists to represent members of different defect clusters on the wafer. This list consists of three fields:

- *x*-die, which denotes the matrix row index of the defective die;
- *y*-die, representing the matrix column index; and
- next-die, which points to the next element in the linked list.

The algorithm forms the list by linking up all the dies in a cluster. Each linked list represents one cluster, and the list's length equals the count of the cluster's defective dies. We pick up a defective die in the linked list and use the first and second fields of the die—*x*- and *y*-die—to compute the distances between two clusters where the two closest dies in each cluster are chosen. In addition, we can link any two clusters together by setting the next die field of the last defective die of one list to point to the first defective die of the other. This data provides test engineers with information they need either to back trace or to find out what could have caused the defect clusters.

Defective-die marking

Some defective dies in the defect clusters can still pass the electrical test (wafer probe). For this reason, we propose going through all the elements in the linked lists and marking their eight-adjacent neighbors as defective dies even if these neighbors pass the electrical test.[10] Figure 2 clarifies the meaning of eight-adjacent; the entry denoted by 15 in Figure 2 connects to all of its eight neighbors.

The last step marks an isolated interior die as a defective die. Then the tool generates an output file for the next processing stage.

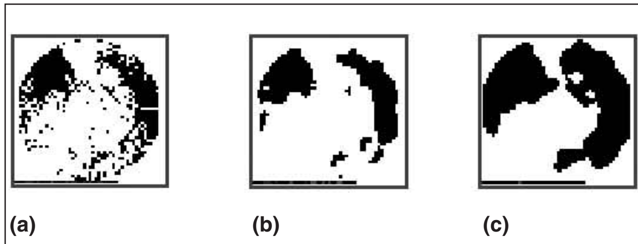

Figure 3. Defect cluster detection, sample 1: original wafer map file (a), median filter output (b), and the detected defect clusters (c).

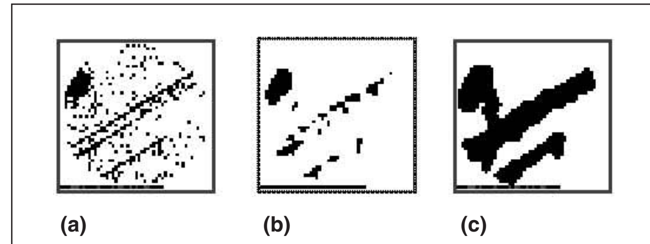

Figure 4. Defect cluster detection, sample 2: original wafer map file (a), median filter output (b), and the detected defect clusters (c).

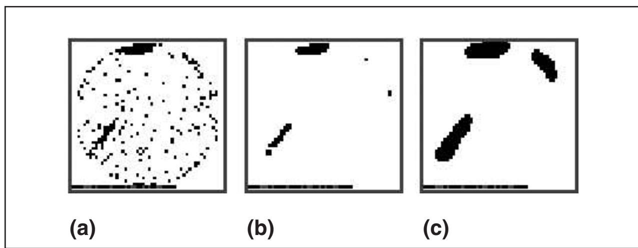

Figure 5. Defect cluster detection, sample 3: original wafer map file (a), median filter output (b), and the detected defect clusters (c).

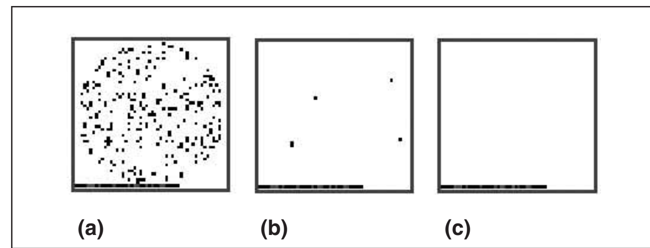

Figure 6. Defect cluster detection on a wafer with no defect clusters: original wafer map file (a), median filter output (b), and the empty defect cluster map (c).

Experimental results

Figures 3a, 4a, and 5a illustrate three original sample map files from the hundreds in the experiment. Although a person can easily recognize defect clusters on these wafers, support testers cannot identify these defect clusters without efficient software. Applying the median filter yields the binary matrices in Figures 3b, 4b, and 5b. Finally, the tool marks the eight-adjacent neighbors and the isolated interior dies as defective. Figures 3c, 4c, and 5c show the results of this process. This method recognizes bad and suspect dies and also labels dies located in flawed areas. Nevertheless, the median filter's limitations imply that it cannot recognize defect clusters where the probe detects fewer than five failed dies.

Figure 6 shows a typical result for our tool when applied to a wafer sample with no clustered defects. The tool treats a cluster as noise when it is too small, as Figure 6b shows. Therefore, this approach does not mistake isolated defective dies for defect clusters.

Besides this experiment, Philips Semiconductor performed a series of burn-in tests, which screen out early-lifetime-failures-under-use conditions. The burn-in tests did not report failures caused by clustered defects.

OUR TOOL SCREENS up to 45,000 wafers per month for Philips, saving at least $100,000 in operating costs, including manual testing expenses. Test results verify that the tool detects all the defect clusters.

Subsequent research will incorporate intelligent tools such as neural networks, fuzzy logic, and genetic algorithms to recognize scattered defective dies and defect clusters of fewer than five failed dies. ■

Acknowledgment

We thank the National Science Council of the Republic of China for financial support under Contract No. NSC 88-2219-E-110-001.

■ References

1. M. Taubenlatt and J. Batchelder, "Patterned Wafer Inspection Using Spatial Filtering for Cluster Environment," *Applied Optics*, vol. 31, no. 17, June 1992, pp. 3354-3362.

2. F.-L. Chen and S.-F. Liu, “A Neural-Network Approach to Recognize Defect Spatial Pattern in Semiconductor Fabrication.” *IEEE Trans. Semiconductor Manufacturing*, vol. 13, no. 3, Aug. 2000, pp. 366-373.
3. A. Vacca, B. Eynon, and S. Yeomans, “Improving Wafer Yields at Low k1 with Advanced Photomask Defect Detection,” *Solid State Technology*, June 1998, pp. 185-190.
4. R. Nurani et al., “In-Line Yield Prediction Methodologies Using Patterned Wafer Inspection Information,” *IEEE Trans. Semiconductor Manufacturing*, vol. 11, no. 1, Feb. 1998, pp. 40-47.
5. C. Hess and L. Weiland, “Extraction of Wafer-Level Defect Density Distribution to Improve Yield Prediction,” *IEEE Trans. Semiconductor Manufacturing*, vol. 12, no. 2, May 1999, pp. 175-183.
6. B. Jahne, *Digital Image Processing: Concepts, Algorithms, and Scientific Applications*, Springer-Verlag, Berlin, 1997.
7. G. Baxes, *Digital Image Processing: Principles and Applications*, John Wiley & Sons, New York, 1994.
8. T. Baumgartner, *SECS II Wafer Map Format*, Electroglas, San Jose, Calif., 1996.
9. N. Pal and S. Pal, “A Review on Image Segmentation Techniques,” *Pattern Recognition*, vol. 26, no. 9, Sept. 1993, pp. 1277-1294.
10. G. Earl, R. Johnsonbaugh, and S. Jost, *Pattern Recognition and Image Analysis*, Prentice Hall, Upper Saddle River, N.J., 1996.

Chenn-Jung Huang is an associate professor in the Department of Computer Science and Information Education at National Taitung Teachers College, Taiwan. His research interests include computer communication networks, image processing, neural networks, fuzzy logic, and computer arithmetic. Huang has a PhD in electrical engineering from National Sun Yat-Sen University, Taiwan.

Chua-Chin Wang is a professor in the Department of Electrical Engineering at National Sun Yat-Sen University, Taiwan. His research interests include VLSI design, low-power and high-speed logic circuit design, neural networks, and wireless communication. Wang has a PhD in electrical engineering from the State University of New York in Stony Brook. He is a member of the IEEE.

Chi-Feng Wu is a factory manager at Philips Semiconductor’s Wafer Testing Factory, Kaohsiung, Taiwan. His research interests include VLSI design, low-power logic, and circuit design. Wu has an MS and a PhD in electrical engineering from National Sun Yat-Sen University, Taiwan.

■ Direct questions and comments about this article to Chenn-Jung Huang, National Taitung Teachers College, Department of Computer Science and Information Education, 684, Section 1, Chunghua Road, Taitung, Taiwan 950; cjh@cc.ntttc.edu.tw.

For further information on this or any other computing topic, visit our Digital Library at http://computer.org/publications/dlib.