Analysis of IC Manufacturing Process Deformations: An Automated Approach Using SRAM Bit Fail Maps

T. Zanon, M. Ferdman, K. Komeyli*, and W. Maly

Carnegie Mellon University, Pittsburgh, PA, USA
** Intel Corporation, Santa Clara, CA, USA*

Abstract

SRAM bit fail maps (BFM) are routinely collected during earlier phases of yield ramping, providing a rich source of information for IC failure and deformation learning. In this paper, we present an automated approach to analyzing BFM data efficiently. We also demonstrate the usability of our analysis framework using real BFM test data from a large, modern SRAM test vehicle.

Introduction

To facilitate the discussion, we first present important terms used extensively throughout this paper.

Terminology

- A *deformation* is a difference between the IC structure of the fabricated device and the desired structure of the nominal device.
- A *deforming event* is the deviation from nominal manufacturing conditions resulting in an IC deformation.
- A *failure* is the manifestation or impact of a deformation on the behavior of a fabricated IC, detectable at test.
- A *bit fail map (BFM)* is a matrix of 0s and 1s describing passing and failing SRAM cells at test (see Figure 1).
- A *colored bit fail map* is a bit fail map with additional information about the way a SRAM cell failed during test. Bit fail map cells are colored based on this information.
- A *failure pattern instance (FPI)* is a group of failing SRAM cells attributed to the same deforming event.
- A *failure pattern type (FPT)* represents a set of similar failure pattern instances (e.g., there is only one single-bit FPT, but many single-bit FPIs in a test data set).
- A *Euclidean minimum spanning tree (EMST)* is the minimum length acyclic graph G connecting all vertices of a set V (see Figure 3b) [1,2].
- An *artificial neural network (ANN)* is a mathematical model that emulates some of the observed properties of biological nervous systems and draws on the analogies of adaptive biological learning. Thus, neural networks are very suitable for learning interrelations in complex data (e.g., complex FPIs) [3].

Application of SRAM bit fail maps in new technologies
SRAM bit fail maps have been successfully used in addressing a number of major tasks in semiconductor manufacturing such as process control monitoring, defect monitoring, and many others in the yield improvement domain [4,5,6,7]. There are, however, three important issues that are becoming stumbling blocks to using SRAM bit fail maps for analysis of the newest IC technologies. First, the volume of bit fail maps harvested during manufacturing of moderns ICs is overwhelming, especially in the starting phase of yield ramping. Second, a significant percentage of large modern memory devices are affected by multiple independent deformations on the same die (*multiplicity* of deforming events). This is especially true for earlier stages of the product cycle, where the assumption of having a single spot defect [6] per die does not hold at all. Third, many failure patterns in modern SRAMs are geometrically very complex and expected to become even more complex in future due to more sophisticated manufacturing processes (*IC failure-layout interaction*).

Consequently, new techniques that could effectively assist yield improvement tasks are likely to become indispensable for any modern IC manufacturing operation.

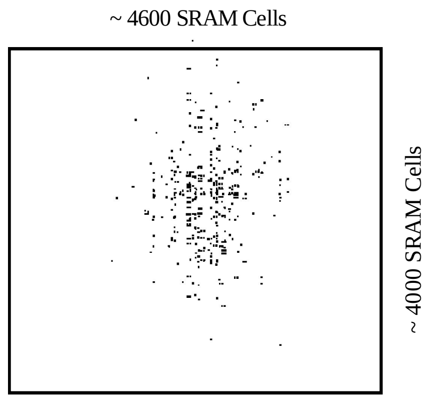


Figure 1: SRAM bit fail map showing a large deformation. Black dots represent SRAM cells that failed at test.

Objective of the paper
This paper describes one possible approach to address the above stated goal of research oriented towards yield improvement. The approach presented in this paper focuses on an automated analysis of SRAM bit fail maps to enable fast yield learning from a large volume of potentially highly complex BFMs found in modern SRAM devices.

The paper is organized as follows: First, a software framework that enables automated analysis of bit fail maps is introduced. Then, fundamental assumptions used in the development of the framework are assessed using bit fail map data from a large, modern SRAM test vehicle. Finally, potential failure analysis applications (e.g., Pareto chart analysis and deformation analysis) that benefit from the use of the presented automated BFM analysis approach are discussed.

SRAM bit fail map analysis approach

To address the need for SRAM bit fail map analysis defined in the introduction, a set of software utilities for processing BFM data has been developed. In this section, an overview of the algorithms embedded in this software as well as the framework organizing interactions between various BFM analysis software components is provided. The flow diagram of this framework is shown in Figure 2. Key functions performed by the major components of the software framework are:

1. *Segmentation of bit fail maps* into failure pattern instances (FPI). Multiple IC structure deforming events that may be present on a single die are separated into individual FPIs.
2. *Extraction of basic characteristics* of failure pattern instances. Complexity of the description of failure patterns instances (FPI representation) is reduced.
3. *Clustering of failure patterns instances* with similar characteristics into failure pattern types (FPT). FPIs are grouped into FPTs based on the similarity of FPI characteristics.

Details of the implementation of each of these three key functions are described below.

Bit fail map data pre-processing
In general, bit fail maps consist of matrices of 0s and 1s describing the pass/fail state of each tested SRAM cell. The amount of data that has to be stored for SRAM devices with large number of failing cells is significant. Further, any processing of BFM data in their raw format is computationally very inefficient and impractical. Therefore, all raw bit fail maps are initially compressed using a data reduction algorithm that combines neighboring failing cells into rectangular *fail regions*. These fail regions are then stored in a database (here: *PostgreSQL* [8]) for further analysis.

Segmentation of bit fail maps
Experience shows that in modern manufacturing processes a significant number of bit fail maps are likely to be affected by more than one deforming event. This is especially pronounced during yield ramping. Furthermore, it is possible that some bit fail maps consist of many individual fail regions, which are

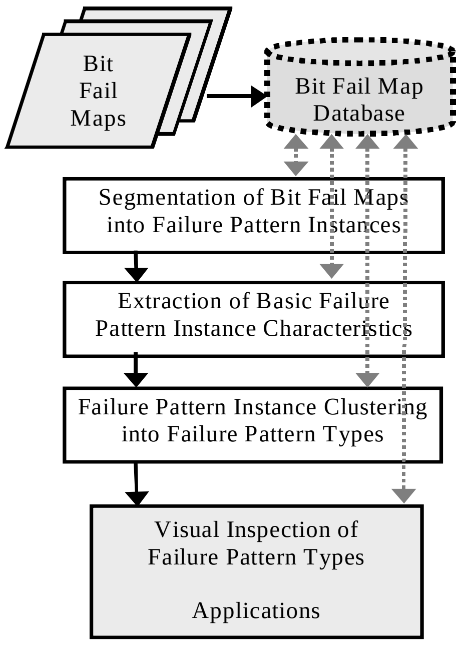

Bit Fail Maps
Bit Fail Map Database
Segmentation of Bit Fail Maps into Failure Pattern Instances
Extraction of Basic Failure Pattern Instance Characteristics
Failure Pattern Instance Clustering into Failure Pattern Types
Visual Inspection of Failure Pattern Types
Applications

Figure 2: Flow diagram of bit fail map analysis framework.

actually caused by the same deforming event. An example is illustrated in Figure 1. A large number of single and double-bit failures in this die were evidenty caused by the same deforming event. The ultimate objective of segmentation is to identify sets of SRAM fail regions that are likely caused by a single deforming event. This means, for example, that all single-bit and double-bit failures of the large area deformation in Figure 1 are grouped into a single failure pattern instance.

A divisive clustering technique [9] to separate fail regions into failure pattern instances is used. In order to perform segmentation, the algorithm examines the relationships between fail regions and uses four empirically determined parameters to decide whether a set of fail regions should belong to the same FPI. In a divisive clustering approach, all fail regions are first considered to belong to one FPI. This failure pattern instance is then iteratively split into smaller ones, as long as the overall quality of the segmentation improves. The procedure terminates when no further improvement can be achieved by splitting any of the resulting FPIs.

The algorithm begins from a bit fail map of fail regions (Figure 3a). First, a Euclidean minimum spanning tree (EMST) connecting all fail regions (EMST vertices) on a die is created. The distance between the closest points of the fail regions determines the edges of the EMST (Figure 3b). The EMST edges are then traversed, starting from the longest edge. Based on four clustering parameters, each edge is determined to either span fail regions belonging to the same or two different FPIs. If the edge is determined to span two FPIs, the edge is cut (a detailed description of the *edge cutting procedure* is given below) and the procedure continues with the next largest edge. If the edge is determined to connect two fail regions within a single FPI, all fail regions still connected to these two are considered to belong to the same FPI, and no cuts are made to

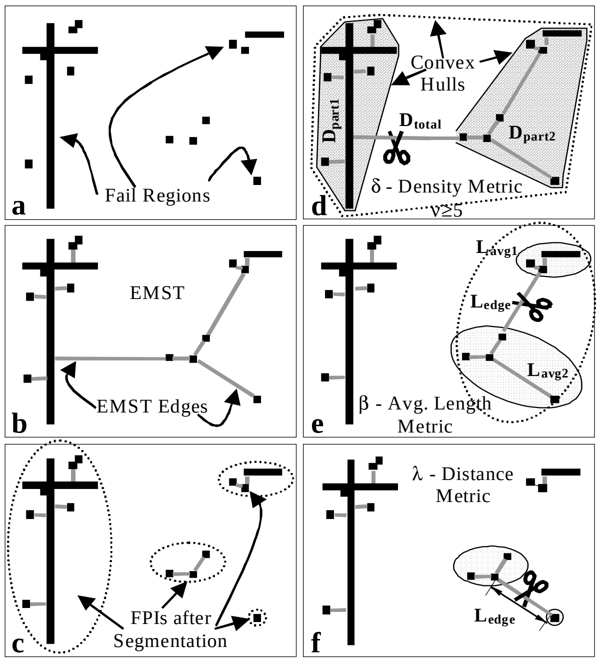


Figure 3: EMST-based bit fail map segmentation into FPIs. The original multi-deformation bit fail map is depicted in a), the resulting segmentation is shown in c). Details of the segmentation procedure are illustrated in d) through f).

separate them. When no more EMST edges that do not connect fail regions within a previously determined FPI remain, the procedure terminates (Figure 3c).

Figure 3d through Figure 3f graphically illustrate how the four parameters (ν, δ, β and λ), are used to determine whether an EMST edge connects fail regions in different FPIs. The parameter ν dictates the smallest number of fail regions that are necessary for a *density metric* (number of defective cells divided by area of a convex hull enclosing these failing cells) to be valid. If there is a sufficient number of fail regions present on at least one side of the edge, the densities of the FPIs before and after cutting are computed (see D_{part1}, D_{part2} and D_{total} in Figure 3d). If $(1-D_{total}/D_{part1})>\delta$ or $(1-D_{total}/D_{part2})>\delta$, cutting this edge would “improve” (increase) the density of at least one resulting FPI, and the edge in question is removed from the EMST. If there are fewer than ν fail regions on at least one side of the edge with length L_{edge} considered for cutting, the average lengths L_{avg} or L_{avg2} of edges spanning from the side of this edge are computed (*average length metric*). If $(1-L_{avg1}/L_{edge})>\beta$ or $(1-L_{avg2}/L_{edge})>\beta$, the edge is cut because this cut would separate an FPI with significantly shorter edges from a larger FPI (Figure 3e). If only one fail region is present on at least one side of the edge in question, neither density nor average edge lengths can be computed for this side of the edge. In this case, the fourth and last parameter λ is used (*distance metric*). λ is compared to the length L_{edge} of the EMST edge in question, and if the edge is longer than λ, it is cut (Figure 3f).

Extraction of basic failure pattern characteristics

In order to make failure pattern instances accessible to failure pattern type instance clustering (the last step of the proposed analysis approach), their geometric complexity has to be reduced. Consequently, the main objective of this step is to extract basic geometric characteristics that represent suitable building blocks for a simplified description of failure pattern instances.

The FPI characterization approach adopted in this research is related to previous work on bit fail map analysis reported in [10] and on defect shape analysis for computer vision applications [11]. The key idea is to describe any geometric pattern by very intuitive and simplified characteristics (note the fundamental difference to approaches based on more complex and less intuitive characteristics using, e.g., statistical moments [12] or Fourier coefficients) In this research, these ideas were modified and extended to accommodate for the intricate failure patterns encountered in the available SRAM test vehicle. A detailed discussion of the basic FPI characteristics that are considered very promising candidates for adequate simplified characterization of FPIs follows in the remainder of this section.

Inspection of failure pattern instances showed that many failure pattern instances do not adequately represent the shape of the underlying deformation. In these cases, the observed failure pattern instance is a response to both the deformation and the layout of the SRAM test vehicle (in general, the word-line and bit-line structure of a memory). As can be seen in the example in Figure 4, consideration of the entire failure pattern instance strongly misrepresents the actual deformation. Consequently, the decision was made to identify and differentiate these *layout-induced failure components* from the remaining failure components of an FPI. These layout-induced components are referred to as *gray-colored* failure components (see Figure 4). The remaining *black-colored* components now serve as a much better representation of the actual area of deformation. Nevertheless, the gray-colored failure components should not be regarded as useless. The strong relationship of the gray-colored components to the SRAM layout points to specific metal layers and makes the gray-colored components a valuable means for pinpointing a potentially defective manufacturing step. Gray-colored failure components are used as basic characteristics of an FPI. In particular, the following three Boolean characteristics

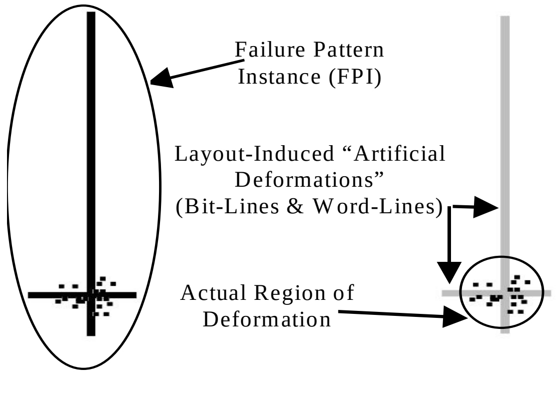

Failure Pattern Instance (FPI)
Layout-Induced “Artificial Deformations” (Bit-Lines & Word-Lines)
Actual Region of Deformation

Figure 4: Examples of SRAM failure pattern instance where the inclusion of layout-induced (gray) failure components strongly misrepresents the underlying deformed memory region.

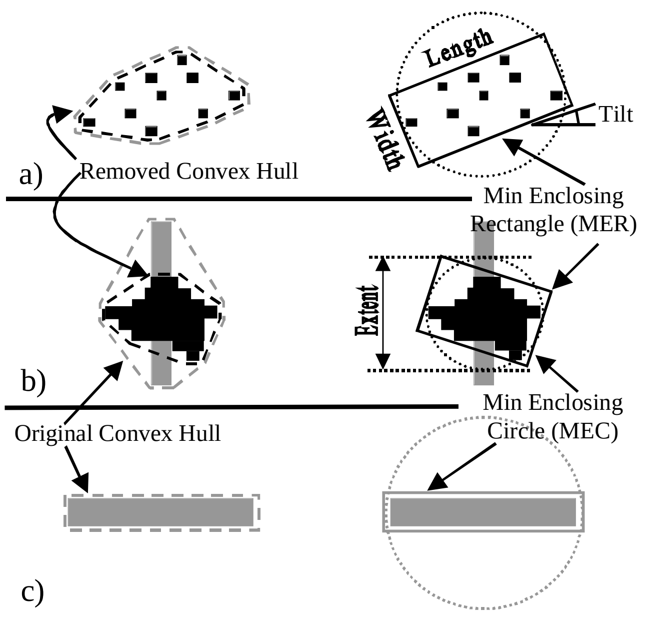


Figure 5: Geometric features used for simplified description of failure pattern instances and deformations. Examples show failure pattern instances with a) black-colored only, b) black and gray-colored, and c) gray-colored only failure components. The geometric features are computed using the Computational Geometry Algorithms Library CGAL [13].

are used: 1) *gray-vertical*, 2) *gray-horizontal*, and 3) *gray-block*, indicating an FPI with at least one vertical, horizontal, or entire SRAM block layout-induced failure component.

Appropriate FPI characteristics are needed to convey the notion of area of a deformed SRAM region. The area of a convex hull appears to be a suitable indicator of the deformed area. Thus, convex hull areas of failure pattern instances before and after "removal" of layout-induced gray-colored failure components (see *original* and *removed convex hulls* in Figure 5) are computed and used as basic FPI characteristics.

Although the knowledge of gray-colored failure components and the area of deformed regions provide considerable information for failure and deformation analysis, it was found experimentally that additional characteristics are required for proper description of many failure pattern instances. Important FPI characteristics are (this list is by no means exhaustive): extent, aspect ratio, principal direction, defective cell density within the deformed region and closeness of the shape of a deformed region to a perfect circle. Some of these characteristics can be derived from more fundamental ones (e.g., the aspect ratio of a deformed region can be described by the ratio of length to width of the deformed region). For the purpose of the research presented in this paper, it was found that the extraction of only three additional, relatively simple geometric features is sufficient to characterize the majority of encountered failure pattern instances according to the above-mentioned list of important characteristics. These geometric features are: 1) the *minimum enclosing circle (MEC)*, 2) the *minimum enclosing rectangle (MER)*, and 3) the *total number of failing SRAM cells* of the deformed SRAM region. These geometric features including convex hulls are illustrated in Figure 5. The FPIs in this figure also demonstrate the focus of the proposed characterization approach on the underlying deformation of a failure pattern instance. If a failure pattern instance has any black-colored failure components, only these components are considered for the MEC and MER computation (see Figure 5a and b). On the other hand, if the entire FPI consists of gray-colored failure components only (see Figure 5c), computation of geometric features is based on the gray failure components to improve the clustering quality of such failure pattern instances.

A summary of how the basic characteristics can be utilized is given in Table 1. Of course, we do not claim that this list of basic geometric characteristics covers every single aspect of failure pattern instance description, but the selected set works extremely well with our data sets.

Clustering of failure pattern instances

Once all bit fail maps have been successfully segmented into failure pattern instances and described using a suitable set of simple FPI characteristics, interesting observations can be made based on this representation of failure pattern instances (see experiment and application sections). However, many useful bit fail map analysis applications (e.g., generation of failure pattern type Pareto charts) require failure pattern instances to be clustered into unique groups. These groups of failure pattern instances are called failure pattern types (FPT). The large number of failure pattern instances hinders manual clustering or at the very least makes manual clustering a time-consuming and error-prone task. Consequently, the main objective of the final step of the proposed analysis approach is to automatically cluster FPIs into appropriate FPTs.

Random inspection of failure pattern instances from available SRAM test data sets has shown that a significant portion of the FPTs would not have been anticipated. Hence, any knowledge-based [14,15] clustering approach would have generated a very large bin of unknown failure pattern instances (FPIs not assigned to any particular FPT). A large bin of "unknowns"

Table 1: Extracted basic FPI characteristics.

Extracted basic FPI characteristic	**Potential use**
Gray-horizontal Gray-vertical Gray-block	Diagnosis of defective manufacturing step; emphasis on deformation
Area of original and removed hulls	Area of the deformed region; Defective cell density
MEC radius MEC center	Extent of deformed region; location of deformed region on die; closeness to circle (e.g., traditional circular spot defect)
MER length MER width MER tilt	Alignment and approximate size of deformed region; aspect ratio (e.g., straight scratch detection)
Defective cell count	Defective cell density; texture of deformed region

corresponds to a tedious inspection process of each failure pattern instance in this bin. Consequently, a clustering approach was needed that does not require any a-priori knowledge of potential clusters in the input data space (here: the space of characterized FPIs). One clustering technique that meets this requirement is based on *growing hierarchical self-organizing maps (GHSOM)* [16,17,18], a particular type of *unsupervised artificial neural networks*. GHSOM-based clustering is a completely input data-driven procedure. Hence, clusters in the space of failure pattern instances are extracted without any specific structural knowledge of this space and without any external interaction throughout the entire exploration (clustering) phase. The resolution of clustering, in essence the desired "crispness" of extracted failure pattern types, can be controlled via two parameters τ_m (breadth growth parameter) and τ_u (depth growth parameter). Exact definition and use of these parameters as well as a detailed description of the operation of the GHSOM clustering procedure is outside the scope of this paper [17,18]. Here only a brief outline of how self-organizing maps facilitate failure pattern instance clustering and how they tie into the presented automated bit fail map analysis approach is given.

A simplified illustration of the FPI clustering procedure using (GH)SOM networks is depicted in Figure 6. First, all failure pattern instances to be clustered are described by an appropriate set of basic and derived characteristics (obtained from the FPI characterization analysis step). The specific characterization of each FPI is stored in a SOM input feature vector. Subsequently, the complete sequence of input feature vectors is fed to the SOM learning algorithm (usually several times). An SOM neural network consists of neurons organized on a regular, usually rectangular, two-dimensional grid. The core of each neuron is a prototype vector, which is of the same dimension as the input feature vector. During learning, the prototype vectors of neurons are affected not only by the stream of input feature vectors but also by changes of the prototype vectors of neighboring neurons in the map. At the end of the training, the prototype vectors of all neurons in the SOM have organized themselves such that prototype vectors close in the input feature space (here: vector of FPI characteristics) are spatially close in

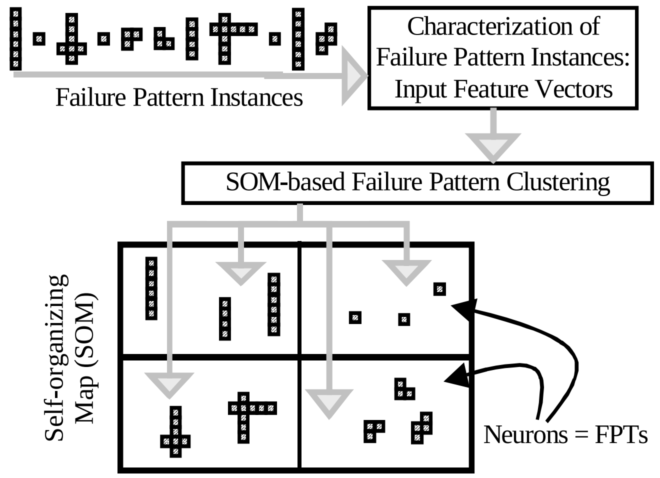

Failure Pattern Instances
Characterization of Failure Pattern Instances: Input Feature Vectors
SOM-based Failure Pattern Clustering
Self-organizing Map (SOM)
Neurons = FPTs

Figure 6: Illustration of FPI clustering using SOM networks.

the SOM network, which explains the name "self-organizing". At this stage, each failure pattern instance can be assigned to exactly one neuron (the neuron having a prototype feature vector closest to the vector of the FPI) and each neuron represents a failure pattern type. Consequently, we have obtained a clustering of failure pattern instances into failure pattern types. The key characteristics of the failure pattern type are reflected in the respective prototype vector of its corresponding neuron.

Experimental verification of framework operation

To assess the correctness of principles guiding the development of the presented three-step SRAM BFM analysis framework, extensive experiments on a large set of real BFM test data from large, modern SRAM products were conducted. Key results of this experiment are reported in the following subsections.

SRAM test vehicle and test data statistics

The test vehicle used for this research is a relatively large static memory device (256K x 72) manufactured in a six-metal 0.13µm L_{eff} process technology. Memory devices of this size consist in general of two or more hierarchy levels of sense amplifier and word access control. For this research, word or bit-lines in the first level of hierarchy are called *local* and those in all higher levels of hierarchy are referred to as *global*. This SRAM property was extensively used for failure pattern characterization (gray-colored failure component). Test data was collected from wafers with two different SRAM cell layouts (manufactured in alternating wafer columns) as well as from two slightly different processes (test data set A and B). To demonstrate the usefulness of the proposed framework without making the procedure overly complicated, BFM analysis in this experiment was based on test data from functional fail testing only. Functional fail testing of the SRAM is carried out at minimum acceptable power supply voltage and nominal clock frequency using an appropriate test algorithm. All failure information was obtained in the previously described format of fail region. Table 2 summarizes the basic test data statistics.

Segmentation of bit fail maps

As previously mentioned, many large SRAM devices fabricated in modern process technologies may be affected by multiple deforming events. After inspection of over 1000 randomly selected SRAM bit fail maps from both test data sets, it was concluded that the proposed failure pattern instance segmentation step is imperative for proper analysis of bit fail maps.

Table 2: Statistics of the collected test data. The gray boxes highlight the number of SRAMs analyzed in this experiment.

		Data set A	Data set B
Wafers		453	439
Total	Dies	46,280	46,156
	Fail regions	6,385,984	3,737,521
Functional fail test only	Dies	22,966	19,036
	Fail regions	5,508,208	2,922,258

Segmentation of bit fail maps is controlled by a set of four parameters ν, δ, β and λ as described in detail in the previous section. In the current stage of research, a suitable set of parameters has to be determined by application of a number of different parameter sets to a reasonably large and representative number of SRAM bit fail maps with subsequent visual inspection of the quality of the segmentation results. Extensive segmentation experiments were conducted on over 1000 SRAM bit maps randomly selected for visual inspection. Finally, the parameter set leading to the highest quality segmentation results was chosen as the parameter set for subsequent segmentation experiments on all available dies (from functional fail test). The values of the parameters selected are listed in Table 3.

Subsequently, segmentation experiments were conducted on both test data sets. The number of failure pattern instances extracted was 38,368 and 28,879, respectively. Inspection of several hundred segmented bit fail maps again confirmed the appropriateness of the chosen segmentation parameter set. The number of extracted failure patterns instances clearly indicates that large, modern SRAM devices are affected on average by more than one deforming event per die. Comparing the number of total extracted FPIs to the number of dies processed results in an average of 1.58 FPI/die and 1.51 FPI/die for test data set A and B, respectively. These results clearly demonstrate the necessity of a BFM segmentation step for large modern memory devices. According to the histogram shown in Figure 7, the bit fail maps of 60-65% of the analyzed die can be explained by a single failure pattern instance (potential occurrence of a single deforming event). A significant portion of more than 30% of the bit fail maps have multiple failure pattern instances and therefore cannot be represented by any model based on the assumption of single defect occurrence (e.g., traditional spot defect model). This is a key observation that must be considered for many tasks in IC fabrication (e.g., test generation or redundancy design for yield improvement).

Clustering of failure pattern instances
Once the SRAM bit fail maps from the collected test data sets were segmented into failure pattern instances and simple and intuitive primary geometric features were computed for each FPI, the proposed GHSOM-based FPI clustering approach could be carried out to obtain failure pattern types. Clustering of FPIs in the presented experiments focused predominantly on the size of deformations. Thus, an appropriate set of characteristics that meets this requirement was selected (focus on, e.g., shape requires a different set to work satisfactorily). The complete set of selected characteristics for the presented experiments is listed in Table 4.

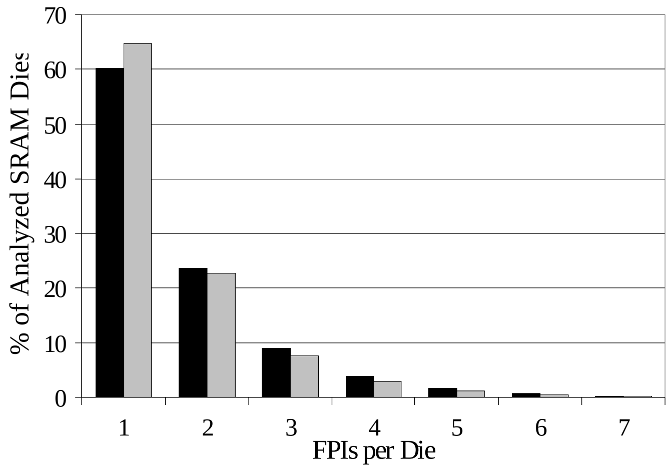


Figure 7: Percentage of dies from test data set A (black) and B (gray) versus number of FPIs found per die.

Table 3: Parameters used for the experiments described in this section.

	Parameters for segmentation of bit fail maps	
ν	Min. number of fail region for density metric	3
δ	Density metric threshold	0.94
β	Average length metric threshold	0.8
λ	Distance metric threshold	200
	Parameters for failure pattern instance clustering	
τ_m	GHSOM breadth growth parameter	0.5
τ_u	GHSOM hierarchy depth growth parameter	0.5e-3

The GHSOM neural network clustering experiments were conducted with both available test data sets using the parameters given in Table 3. These parameters were chosen empirically. However, this selection was not difficult to establish for high quality clustering Figure 8 shows the hierarchical self-organizing map obtained after learning failure pattern clusters in test data set A. Each rectangle of this map represents a neuron. A numbered neuron represents a designated failure pattern type (here: 70). The three unnumbered neurons are empty and are not used as failure pattern types.

Table 4: FPI characteristics used as components of the input feature vector for the demonstrated clustering experiment.

Input feature vector component	
Gray-vertical Gray-horizontal Gray-block	Boolean; separation of FPI with various gray failure components
MER tilt	Three-valued parameter; distinction between word-line, bit-line or no layout alignment of an FPI
MER length MER width	Approximate size and shape of FPI
Original & removed Convex hull areas	Area of deformed region; improved clustering quality of gray-only FPIs
Defective cell density	“Texture” of deformed region
Number of defective cells	Empirically found to be beneficial for FPI clustering.

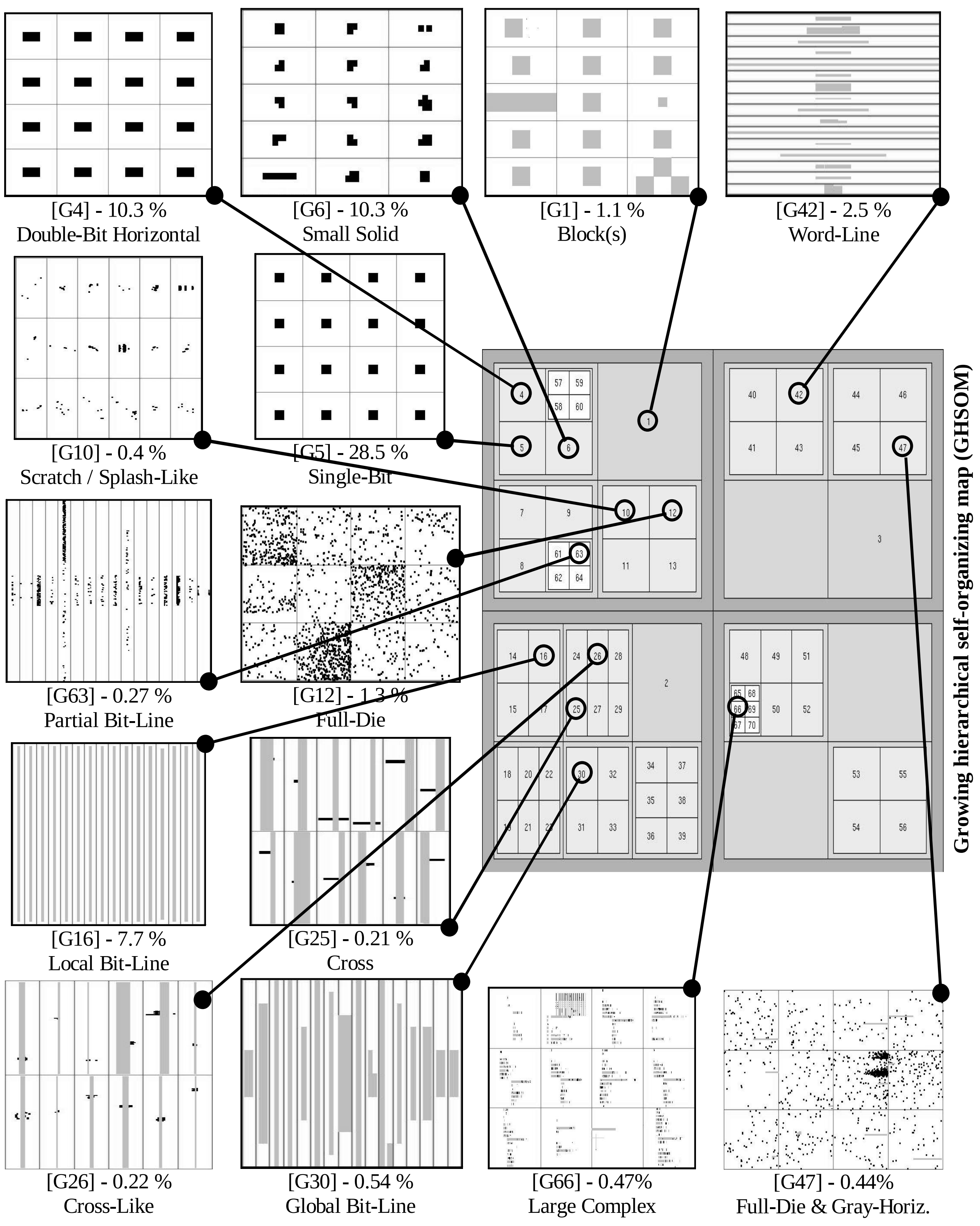


Figure 8: Sample failure pattern type clustering for test data set A. Shown are multiple failure pattern instances for some FPTs. Simple failure patterns (e.g., Single-Bit [G5]) as well as more complex deformations (e.g., Small Solid [G6], Partial Bit-line [G63], Full- Die [G12] and [G47]) have been successfully identified.

Different shades of gray of the boxes indicate different levels (here: four) of hierarchy in the map (with lighter shades representing deeper levels) and thus different levels in the failure pattern instance space. Also shown are several representative failure pattern types that were selected to highlight the benefits of GHSOM-based clustering for investigative BFM analysis.

- All *major "crisp"* (identical) and *highly frequent* FPIs are reliably assigned to unique FPTs (e.g., single-bit [G5], double-bit horizontal [G4] or local bit-line [G16] failures). Although similar results for these *expected* failure pattern instances can be obtained using a knowledge-based clustering approach, it is still important to note that a neural network deals with them appropriately without any additional effort.
- Clustering *less frequent similar* failure pattern instances into the same failure pattern type (e.g., global bit-line [G30], partial bit-line [G63], small dense clusters [G6] or word-line [G42] failures) also works extremely well. The failure pattern instances in these groups, although not identical, appear to share their "strongest" characteristic(s) (e.g., gray-vertical in the case of group [G30]). The clustering behavior of the GHSOM (the desired level of similarity in the FPIs that fall into a single FPT) can be controlled via the parameters τ_m and τ_u.
- Completely *unexpected* failure pattern instanced (e.g., large complex failures [G66], which were observed in test data set A, but not in test data set B) are identified and clustered into meaningful FPTs. Thus, the GHSOM deals with these FPIs easily without any prior knowledge of their existence. Any knowledge-based clustering approach would have assigned this and many other unexpected failure patterns instances to the "unknown" bin, requiring tedious inspection of many FPIs in this bin before this FPT is discovered.
- Some initially quite similar looking FPIs with fine differences are clustered into separate failure pattern type bins. This behavior of the GHSOM potentially increases the diagnostic capability of BFM analysis. E.g., the full-die failures [G12] and [G47] look very similar at first glance, but upon closer inspection are separated because of one or more gray-colored horizontal failure components in failure pattern instances of FPT [G47]. Thus, FPT [G47] allows a more detailed diagnosis of the underlying manufacturing problems.

BFM analysis run-time
The experiments have been conducted on a Sun Sparc Fire 280R with a 900 MHz CPU and 4 GBytes of RAM. Typical run-times for each of the three main analysis steps of the framework are listed in Table 5. Analysis execution times were found to be reasonably fast, even without code optimization. Especially the first step of the analysis framework can be expected to become significantly faster with an optimized implementation of the EMST algorithm.

Table 5: Analysis run- times for the three main analysis steps.

Step 1: BFM Segmentation	75 minutes for 10,000 BFM (100 wafers) → 0.45 seconds / BFM
Step 2: FPI characterization	7 minutes for 10,000 failure pattern instances → 0.042 seconds / FPI
Step 3: FPI clustering	< 2 minutes for clustering approx. 30,000 failure pattern instances

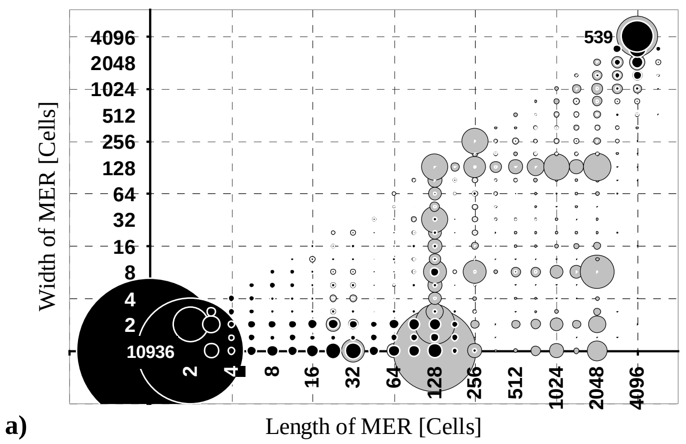
Verification experiment - Conclusions
The presented experiment has decisively shown that the proposed BFM analysis approach can be utilized for efficient investigations of large sets of complex bit fail maps with multiple deformation events per die. This is achieved by quickly and automatically processing bit fail map information and condensing it to a small number of "intelligently" clustered failure pattern types for further failure analyses and diagnosis. It was also found that clustering results of excellent quality can be obtained using simplified characterization of FPIs in combination with a GHSOM neural network. Finally, the run-time of the software at the current stage of development was found to be fast enough to perform SRAM BFM analysis of several hundred of wafers in reasonable time.

Applications

We investigated potential failure analysis and diagnosis applications for IC manufacturing that can benefit from the proposed SRAM BFM analysis framework. Three example applications, including preliminary results, are briefly illustrated in this section.

Building a deformation size portfolio
The extracted FPI characteristics can be directly used to learn about specific deformation properties using standard statistical means (e.g., using histograms). An example is shown in Figure 9a. The bubble chart bins the deformed regions from test data set A according to their length (horizontal axis) and width (vertical axis) (using length and width of the MER). The area of a bubble represents the number of FPIs in a given bin. The bubble's color differentiates FPIs with black-colored failure components only (black bubbles) and FPIs with at least one gray-colored failure component (gray bubbles). Two very distinct observations can be made. First, an obviously large number of small failure pattern instances exist (black bubbles in the lower-left corner of graph, representing, e.g., single and double-bit failures) as well as a quite considerable number of full-die failures. Second, the impact of the layout-induced failure components (gray-colored failure components) is evident. This once again underlines the importance of handling gray-colored components in a special manner.

Size analysis of deformed regions can be further improved by taking failure pattern types (FPT) into account. Failure pattern types were generated according to a variety of FPI characteristics and potentially represent different deforming events. Thus, based on FPT information, one can now more reliably study the approximate size of specific deforming events. Examples are illustrated in Figure 9b. The failure pattern instances of two FPTs from test data set A were binned according to their length and width of the deformed region. The black bubbles represent small solid FPIs ([G6] in Figure 8) as a likely result of spot defects. The gray bubbles are for slightly elongated "splash-like" FPIs with lower defect cell density



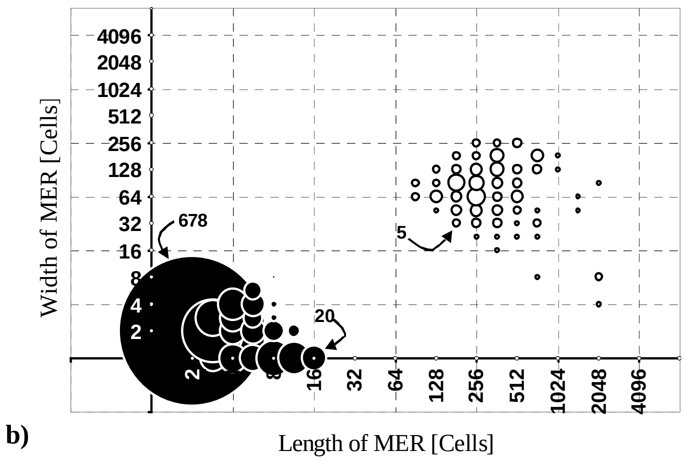


Figure 9: Bubble charts showing (approximate) sizes and shapes of deformations (FPI) using the MER information from the second analysis step. a) FPIs with black-colored failure components only (black) and FPIs with at least one gray-colored failure component (gray); b) Small solid FPIs [G6] (black) and scratch/splash-like FPIs [G10] (white).

[G10] caused by some directional deforming event (e.g., wafer handling issue).

Generation of failure type Pareto chart

The number of failure pattern types obtained from the presented analysis framework is reasonably small, therefore enabling manual inspection and processing of BFM data. Visual inspection and labeling of failure pattern types has become, although still a human task, relatively simple and manageable. E.g., the generation of a FPT Pareto chart, which is a very popular failure analysis means, can be achieved in minutes (see Figure 10). This chart can then be used to support learning of key yield detractors and to identify particular failure pattern types for further, more detailed failure analysis (e.g., feeding a dictionary-based inductive fault analysis with failure patterns instances that are suitable for such a framework or identifying large area failures for spatial frequency analysis to learn about *deformation-layout interactions*).

Failure pattern type-based process monitoring

A more advanced application that benefits from the proposed

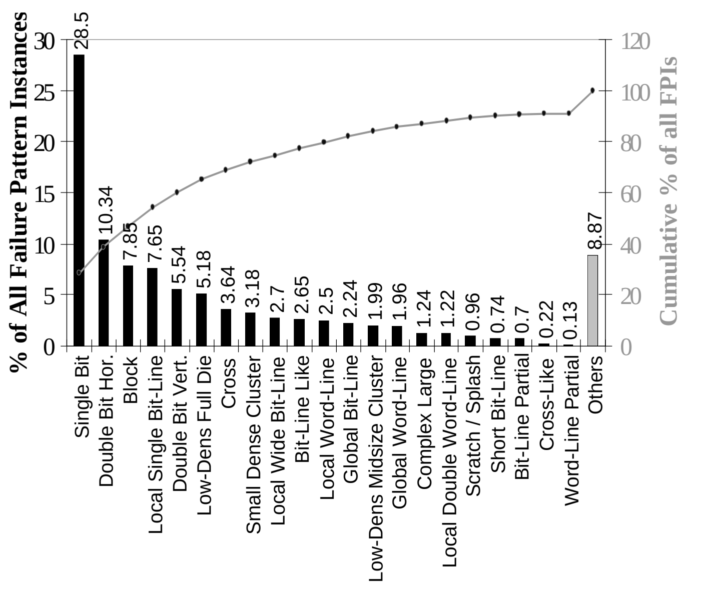
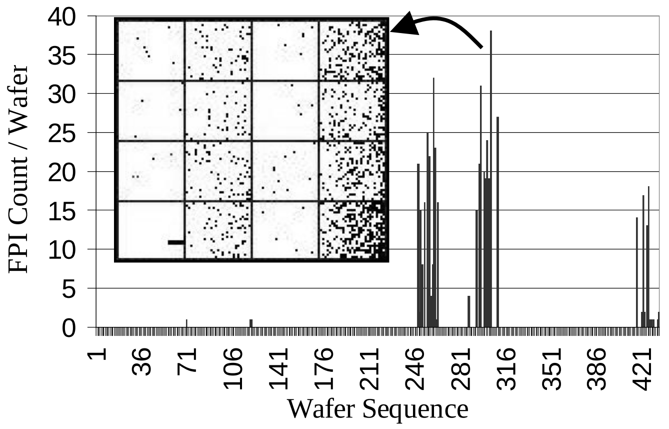


Figure 10: Pareto chart of major failure pattern types (FPT) found in SRAM test data set A.



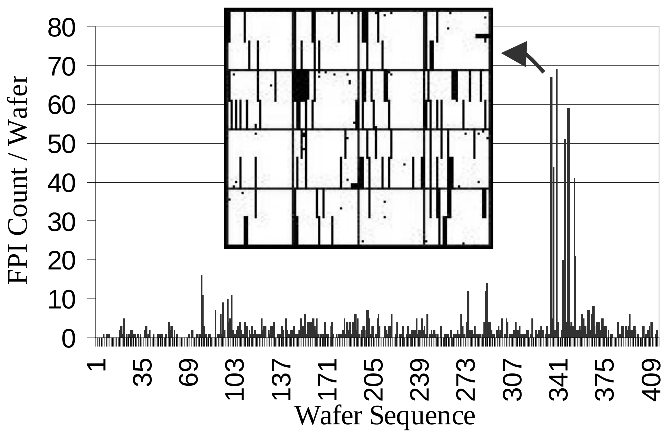


Figure 11: Plot showing the number of failure pattern instance of specific failure pattern types per wafer versus the wafer sequence. FPT dependent excursions can be clearly identified, potentially indicating different root causes of the observed excursions.

BFM analysis framework is failure pattern type-based process monitoring for excursion identification or trend analyses. Established means already exist, of course, to catch excursions (e.g., yield monitoring based on e-test and in-line test data). Nevertheless, knowing the predominant failure patterns type(s) during an excursion period could greatly benefit root cause analysis. Thus, process monitoring based on failure pattern types extracted by the presented analysis framework can enhance existing excursion identification practices. An illustration is given in Figure 11. These plots show the number of occurrences of selected failure pattern types per wafer with the wafers arranged in the order of manufacturing. The two types are: a) low-defective cell density, full-die failures [G12] in Figure 8 and b) eight-column-wide, long-bit-line fails [G16] of test data set B. The plots aid in monitoring the respective failure pattern types over time. Various excursion periods can be easily spotted and linked to potential root causes. The failure pattern type in Figure 11a indicates a parametric front-end-issue spread over the entire wafer (note the different impact of the deforming event on the two products on the wafer). The failure pattern type in Figure 11b can most likely be traced back to a sense amplifier issue, since the eight-column-wide bit-line fails are aligned with the locations of these circuits in the layout.

Conclusions

In this paper, we presented an automated three-step approach for the analysis of bit fail maps from modern static memories. Experiments based on real SRAM bit fail map data demonstrate that the proposed approach can be very beneficial for failure and deformation learning in any semiconductor manufacturing operation.

Key observations are:

- A significant portion of modern large SRAMs (and likely other products) is affected by more than one deforming event.
- A large number of failure pattern types encountered in bit fail maps of modern static memories exhibit a very complex nature. This is especially true in the yield ramping phase where SRAM bit fail map data plays a major role in failure analysis. We found that a neural network approach for SRAM BFM analysis is extremely beneficial for successful data exploration.
- More comprehensive bit fail map data very likely improves the diagnostic capabilities and resolution of the SRAM bit fail map analysis framework. More complete knowledge of failures and deformations could be established by applying the presented analysis approach to *colored bit fail maps* that are colored according to: a) the failed test, b) the actual steps within a functional test that failed a cell or c) current measurements of failing SRAM cells. Investigation of these opportunities, of course, requires significant support from semiconductor manufacturers by testing and providing necessary test data.

Acknowledgements

The authors would like to thank Intel Corporation for providing valuable data from their SRAM test vehicle for this research as well as financial assistance. The authors would further like to thank Alisa Yurovsky for countless hours of software development, the CMU/CSSI test group for discussions and Thomas Vogels for his very constructive proofreading.

References

1. R.L. Graham and P. Hell, *On the History of the Minimum Spanning Tree Problem, Annals of the History of Computing*, 7, 43-57 (1985).
2. J.B. Kruskal, *On the Shortest Spanning Tree of a Graph and the Traveling Salesman Problem, Proc. of the American Mathematical Society*, 7, 48-50 (1956).
3. L. Fausett, *Fundamentals of Neural Networks*, Prentice-Hall, Upper Saddle River, New Jersey (1994).
4. C. H. Stapper, *On Yield, Fault Distributions, and Clustering of Particles, IBM Journal of Research and Development*, 30(3), 326-338 (1986).
5. S. Lopez and D. Bakker, *Correlating Defects to Bit Map Failures Using Automated Patterned Wafer Inspection Systems*, presented at the UltraClean Manufacturing Symposium, (1992).
6. J. Khare, et.al., *Yield-Oriented Computer-Aided Defect Diagnosis, IEEE Transactions on Semiconductor Manufacturing*, 8(2), 195-206 (1995).
7. J. Khare and W. Maly, *Contamination Diagnosis Using Contamination-Defect_Fault (CDF) Simulation, Proc. of International Symposium for Testing and Failure Analysis*, 109-114 (1996)
8. *PostgreSQL User's Guide*, http://www.postgresql.org.
9. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, Wiley-Interscience, New York (1973).
10. R.S. Collica, et.al., *SRAM bitmap Shape Recognition and Sorting Using Neural Networks, IEEE Transactions on Semiconductor Manufacturing*, 8(3), 326-334 (1995).
11. M. Peura and J.Iivarinen, *Efficiency of Simple Shape Descriptors, 3rd International Workshop on Aspects in Visual Forms*, 443-51, (1997).
12. S.S. Gleason, et.al., *Rapid Yield Learning through Optical Defect and Electrical Test Analysis, SPIE's International Symposium on Metrology, Inspection, and Process Control for Microlithography*, (1998)
13. *CGAL User's Guide*, http://www.cgal.org.
14. B.B. Sindahl, *Interactive Graphical Analysis of Bit Fail Map Data Using Interactive Pattern Recognitions, Proc. of International Test Conference*, 687-695 (1987).
15. C.A.Gloor, *Embedded Memory Analysis for Standard Cell ASIC Yield Enhancement, Proc. of International Symposium for Testing and Failure Analysis*, 69-75 (2000).
16. T. Kohonen, *Self-Organization and Associative Memory*, Springer Verlag, New York (1988).
17. *The Growing Hierarchical Self-Organzing Map Project*, http://www.ifs.tuwien.ac.at/~andi/ghsom.
18. M. Dittenbach, D. Merkl, and A. Rauber, *The Growing Hierarchical Self-Organizing Map, Proc. of the International Join Conference on Neural Networks*, 6, 15-19 (2000).