A Die-Selection Method Using Search-Space Conditions for Yield Enhancement in 3D Memory

Joohwan Lee, Kihyun Park, and Sungho Kang

Three-dimensional (3D) memories using through-silicon vias (TSVs) as vertical buses across memory layers will likely be the first commercial application of 3D integrated circuit technology. The memory dies to stack together in a 3D memory are selected by a die-selection method. The conventional die-selection methods do not result in a high-enough yields of 3D memories because 3D memories are typically composed of known-good-dies (KGDs), which are repaired using self-contained redundancies. In 3D memory, redundancy sharing between neighboring vertical memory dies using TSVs is an effective strategy for yield enhancement. With the redundancy sharing strategy, a known-bad-die (KBD) possibly becomes a KGD after bonding. In this paper, we propose a novel die-selection method using KBDs as well as KGDs for yield enhancement in 3D memory. The proposed die-selection method uses three search-space conditions, which can reduce the search space for selecting memory dies to manufacture 3D memories. Simulation results show that the proposed die-selection method can significantly improve the yield of 3D memories in various fault distributions.

Keywords: Yield enhancement, 3D memory, 3D memory stacking, inter-die redundancy, die-selection method.

I. Introduction

Three-dimensional (3D) integrated circuits (ICs) with through-silicon vias (TSVs) exhibit promise for overcoming well-known wall problems, such as interconnects [1]. Memory plays an important role in high-performance systems and will likely be the first commercial application of 3D IC technology [2], [3]. 3D memory is implemented with TSVs as vertical buses across memory layers. Thus, 3D memory can reduce memory access latency and increase memory access bandwidth [2].

As the capacity and density of 3D memory are extremely high, 3D memory defects are easily manufactured. It results in yield drop and quality degradation [4], [5]. Memory repair is used for achieving reasonable yields [6]-[9]. 3D memory is typically composed of known-good-dies (KGDs), which are repaired using self-contained redundancies. To verify a KGD, a number of repair algorithms can be used, which have been presented for two-dimensional (2D) memory using both spare rows and spare columns [10]-[16].

The majority of redundancy analysis (RA) algorithms use self-contained redundancies for repairing memory faults. When a memory die is repaired, the unused redundancies may remain. If the redundancies in a memory die are designed conventionally, these unused and remaining redundancies are wasted. However, if the remaining redundancies can be used in another memory die, more KGDs are additionally obtained. In [10], the unused redundancies are shared to increase memory yield for traditional 2D memory, but this strategy is not popular in 2D memory because of its high routing overhead. In 3D memory, however, the redundancy sharing between vertical memory layers is a practicable and effective strategy because the short electrical path length of TSVs can make routing easy.

Manuscript received Feb. 25, 2011; revised Apr. 16, 2011; accepted May 6, 2011.

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2010-0026822).

Joohwan Lee (phone: +82 2 2123 2775, email: eldsich@yonsei.ac.kr), Kihyun Park (email: cimabear@yonsei.ac.kr), and Sungho Kang (corresponding author, email: shkang@yonsei.ac.kr) are with the Department of Electrical and Electronic Engineering, Yonsei University, Seoul, Rep. of Korea.

http://dx.doi.org/10.4218/etrij.11.0111.0108

When the redundancy sharing strategy is used, it is inefficient that 3D memory is composed of only KGDs because the unused redundancies in 3D memory are wasted. A known-bad-die (KBD) is an irreparable memory die because the number of redundancies in a KBD is insufficient. A KBD is not repaired using self-contained redundancies. However, with the redundancy sharing strategy, a KBD possibly becomes a KGD after bonding. Intuitively, to obtain more 3D memories without waste, 3D memories should be manufactured using KBDs as well as KGDs. In other words, the remaining redundancies in a memory die can be used to repair faults of another memory die for manufacturing a good 3D memory. Both KGDs and KBDs should be carefully selected. If the number of unused redundancies in a KGD is more than the number of insufficient redundancies in a KBD, a good 3D memory is manufactured but there are still unused redundancies remaining. Conversely, if the number of unused redundancies in a KGD is less than the number of insufficient redundancies in a KBD, a 3D memory is irreparable after stacking. Therefore, it is very important to select an adequate die for enhancing 3D memory yield without too much waste of redundancies.

Recently, two die-selection methods have emerged for yield enhancement in 3D memory. Jiang and others [8] use irrespective sub-bipartite graphs to selectively match memory dies together. The die-selection method in [8] is able to enhance 3D memory yield. However, the technique in [8] needs an exhaustive search to obtain the maximum yield, which is too time-consuming to be used in practice. Furthermore, since different memory dies require distinct fault bitmaps, the method in [8] is not cost-effective. Chou and others [9] exactly match one die to another die with inter-die redundancies. That is to say, the number of unused redundancies in one die is the same as the number of insufficient redundancies in another die. Contrary to the technique in [8], the die-selection method in [9] does not use much time because it does not need an exhaustive search. The technique in [9] does not need fault bitmaps, either. Therefore, the die-selection method in [9] is cost-effective. However, the 3D memory yield using the technique in [9] can be lower than that using a die-selection method with considering the property of single cell faults. The property of single cell faults is that a single cell fault which does not share a row address and a column address with other faults can be repaired by either a spare row or a spare column. In this paper, we propose a die-selection method using three search-space conditions for yield enhancement in 3D memory. The proposed die-selection method needs neither exhaustive search nor fault bitmaps. The property of single cell faults is used to improve the yield of 3D memories. Thus, the proposed method is cost-effective and enhances the 3D memory yield.

The rest of this paper is organized as follows. Section II introduces background information concerning a die-selection method. Section III describes the proposed die-selection method with a simple example. Experimental results of fault distributions, the yield of 3D memories, and the area overhead are shown and analyzed in section IV. Finally, section V concludes this paper.

II. Background

1. Stacking Flow of 3D Memory

3D memory can be built by three possible integration methods [8], [9]: wafer-to-wafer (W2W), die-to-wafer (D2W), and die-to-die (D2D). W2W integration technology has a simple manufacturing process, but the yield of 3D memories can be quite low. D2D (D2W) integration technology, on the other hand, requires a more complex manufacturing process. However, the yield of D2D-manufactured 3D memories can be much higher than that of W2W-manufactured 3D memories. In this paper, D2D integration technology is used to form 3D memory stacking. However, it can be modified for W2W or D2W integration technology by adding the constraint of the position of a die in a wafer.

Figure 1 shows the stacking flow of 3D memory based on D2D integration technology. First of all, memory dies are classified into four types (fault-free die, self-repairable die, inter-repairable die, and irreparable die) according to the states after pre-bond test and repair, as shown in Fig. 1. The classified dies are selected to stack 3D memories before stacking. The yield of 3D memories is strongly influenced according to the die-selection method. After stacking, post-bond test and repair is carried out in order to finally ensure the reliability of 3D memory. Then, good 3D memories are shipped. Otherwise, faulty 3D memories are scrapped.

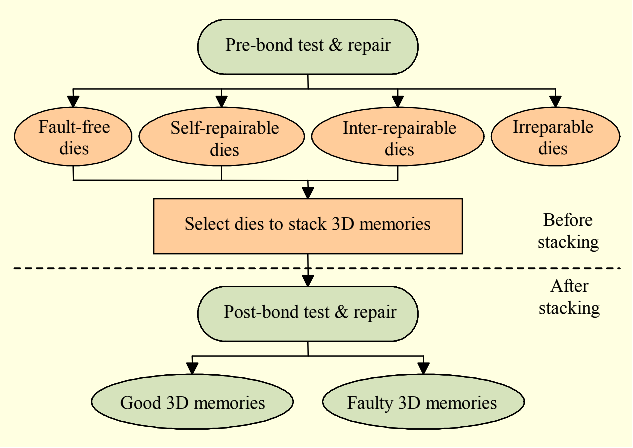


Fig. 1. Stacking flow of 3D memory with inter-die redundancy.

2. Classification of Memory Dies

The memory dies with inter-die redundancies are classified into the following four types in the phase of pre-bond test and repair: fault-free die, self-repairable die, inter-repairable die, and irreparable die, as shown in Fig. 1. A fault-free die has no faults and uses no redundancies. A self-repairable die can be repaired by the self-contained redundancies and may leave unused redundancies. An inter-repairable die cannot be repaired by the self-contained redundancies but can be repaired by the inter-die redundancies. An irreparable die cannot be repaired by the inter-die redundancies or by the self-contained redundancies. Current 3D memories are produced using fault-free dies and self-repairable dies since they do not use inter-die redundancies. The 3D memories which use inter-die redundancies, however, are stacked with fault-free dies, self-repairable dies, and inter-repairable dies. Therefore, the inter-die redundancy scheme is able to increase 3D memory yield.

3. Architecture of 3D Memory with Inter-die Redundancies

Diverse types of redundancies are considered for the repair of defective memories with inter-die redundancies. In [8], both a programmable decoder and multiplexers are used for redundancy sharing. The shift reconfiguration mechanism is used to exchange a defective element with an inter-die redundancy, as described in [9]. Other redundancy architectures which use a repair address register [11], [13]-[16] can also be used for inter-die redundancies. The repair address register used in 3D memory requires an additional single bit compared to that of 2D memory to determine which die was repaired.

Figure 2 depicts the inter-die redundancy scheme used in the proposed die-selection method. A spare row and two spare columns on each memory die are connected to the programmable decoder of other layers using TSVs as well as the decoder on its own layer. The routing overhead to support inter-die redundancies is quite low due to the use of short TSVs. The multiplexers control which memory dies use the corresponding redundancies.

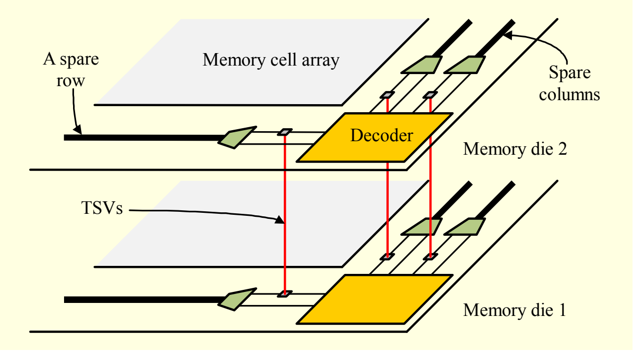


Fig. 2. Inter-die redundancy scheme.

4. Property of Single Cell Faults

The majority of state-of-the-art memories adopt 2D spare architecture [11], [13]-[16], which is generally composed of spare row lines and spare column lines. A memory with 2D spare architecture obeys a line replacement policy. A line replacement policy [13], [16] dictates that any fault in a memory has to be replaced with a spare line. A faulty cell should be replaced with a spare row line or a spare column line. Among the faults assigned to spare lines, there are single cell faults which do not share a row address and a column address with other faults. A single cell fault can be repaired by either a spare row line or a spare column line. This property of single cell faults is used to improve the yield of 3D memories in the proposed die-selection method.

III. Proposed Die-Selection Method for 3D Memories

An efficient die-selection method using three search-space conditions is proposed to maximize the yield of 3D memories with inter-die redundancies. Before the proposed die-selection method is applied, the information about memory dies should be collected. It is assumed that the built-in self-repair (BISR) of each memory die collects the information for the proposed die-selection method during pre-bond test and repair, as shown in Fig. 3. In Fig. 3, when the BISR is started, a built-in self-test (BIST) generates test patterns, such as the March test, to determine whether the memory under test (MUT) has a fault. If a fault is not detected, the MUT is regarded as a fault-free die and the information of the MUT is stored. Otherwise, the built-in redundancy analysis (BIRA) allocates a redundancy to repair the memory. After the BIRA, if the MUT is not repairable by

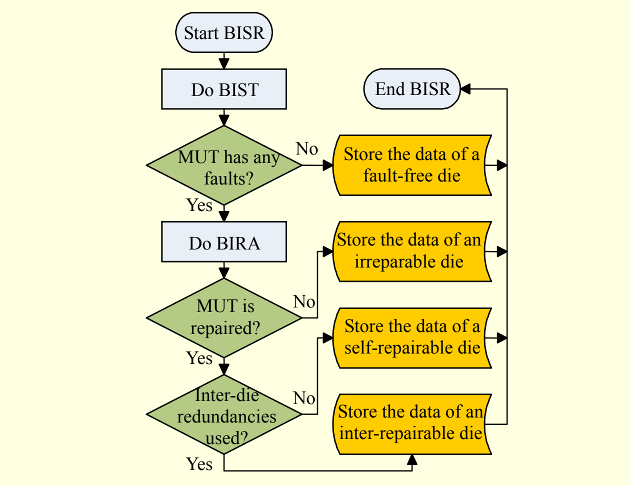

Start BISR
Do BIST
End BISR
MUT has any faults?
No
Store the data of a fault-free die
Yes
Do BIRA
Store the data of an irreparable die
MUT is repaired?
No
Yes
Store the data of a self-repairable die
Inter-die redundancies used?
No
Store the data of an inter-repairable die
Yes

Fig. 3. BISR flow for proposed die-selection method.

using the self-contained redundancy as well as the inter-die redundancy, then the MUT is classified as an irreparable die and the information of the MUT is recorded. Conversely, if the MUT can be repaired by any type of redundancy, the MUT is determined as a self-repairable die or an inter-repairable die according to the use of inter-die redundancies and the information of the MUT is stored. After all the memory dies are collected, the pre-bond test and repair is ended.

With the information collected by the BISR, the proposed die-selection method selects a target die to be stacked and then identifies a counterpart die which meets the three search-space conditions. Search-space condition 1 sets limits on the state of spare rows for the counterpart dies, and search-space condition 2 sets limits on the state of spare columns for the counterpart dies. These two conditions are as follows. R_T (C_T), R_C (C_C), and R_S (C_S) in search-space condition 1 (search-space condition 2) denote the value of the state of spare rows (columns) in the target die after pre-bond test and repair, the value of the state of spare rows (columns) in the counterpart die after pre-bond test and repair, and the number of the total spare rows (columns) in a die, respectively. R_T, C_T, R_C, and C_C can be positive numbers, negative numbers, or zeros. After pre-bond test and repair, if unused spare rows (columns) remain in the target die and the counterpart die, R_T and R_C (C_T and C_C) are positive numbers, respectively. However, if spare rows (columns) in the target die and the counterpart die are insufficient, R_T and R_C (C_T and C_C) are negative numbers, respectively. On the other hand, both R_S and C_S are always positive numbers.

$$\text{Search-space condition 1:} \quad -R_T \le R_C \le R_S,$$
$$\text{Search-space condition 2:} \quad -C_T \le C_C \le C_S.$$

The previous die-selection method [9] uses exact matching, but the proposed die-selection method using search-space conditions 1 and 2 finds the counterpart die within the search space of the conditions. Since the two search-space conditions increase the number of counterpart die candidates as compared with the previous die-selection method [9], the target die is easily matched with the counterpart die. In other words, it is easy to manufacture 3D memories when the proposed die-selection method is used. Therefore, the proposed method using search-space conditions 1 and 2 can increase the 3D memory yield.

A memory die classification map is used for the purpose of finding the counterpart die as fast as possible. The information of a memory die for the proposed die-selection method is recorded in the memory die classification map right after the memory die is classified. Figure 4 shows the memory die classification map for the memory die with one spare row and two spare columns. In Fig. 4, *SSR* (*SSC*) denotes the state of spare rows (columns) in a die after pre-bond test and repair.

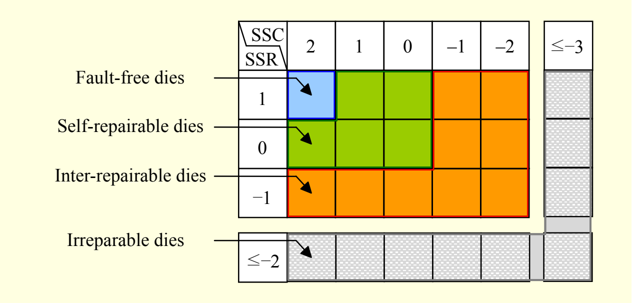
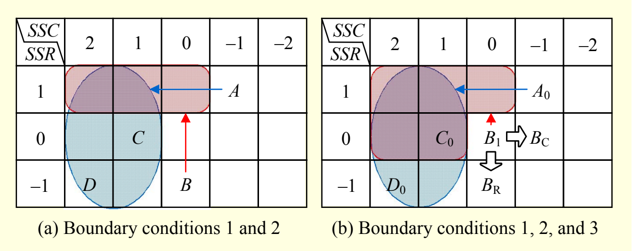


Fig. 4. Memory die classification map (R_S = 1 and C_S = 2).



(a) Boundary conditions 1 and 2 (b) Boundary conditions 1, 2, and 3

Fig. 5. Example of proposed die-selection method.

The positive value of *SSR* (*SSC*) indicates that there are unused redundancies in a repaired memory die. On the other hand, the negative value of *SSR* (*SSC*) represents that a tested memory die needs inter-die redundancies to repair the memory die. In this work, since the proposed die-selection method is used for two-layer 3D memory and the memory dies have one spare row and two spare columns, the value of *SSR* (*SSC*) is bounded from –1 (–2) to 1 (2). If the value of *SSR* is less than or equal to –2 or the value of *SSC* is less than or equal to –3, the memory die is an irreparable die. There is no need to store irreparable dies in the memory die classification map because irreparable dies are not used for 3D memory stacking. The classified memory dies except irreparable dies are recorded in the memory die classification map. The exact locations of classified memory dies are determined by the results of the BISR.

Figure 5 shows an example of how the proposed die-selection method finds the counterpart die for four memory dies with one spare row and two spare columns. In Fig. 5(a), the previous die-selection method [9] is unable to identify a counterpart die because there is no exact matching die; however, the proposed die-selection method using search-space conditions 1 and 2 is able to find a counterpart die. Since the search space for finding counterpart dies is instantly determined, the inter-repairable die *A* can be matched with the self-repairable die *C* or the inter-repairable die *D*, as shown in Fig. 5(a). If a counterpart die can be selected, a good 3D memory is manufactured using the target die and the counterpart die. In Fig. 5(a), however, the inter-repairable die *B* is still unmatched.

A single cell fault that does not share a row address and a column address with other faults can be repaired by either a spare row or a spare column. According to this property of single cell faults, both a spare row and a spare column can be repair solutions for a single cell fault. Therefore, the repair decision for single cell faults can be postponed until the other faults in a 3D memory have been determined. The property of a single cell fault is considered in search-space condition 3, in which S_T and S_C denote the number of single cell faults in the target die and the counterpart die, respectively.

Search-space condition 3: $-R_T-C_T+S_T+S_C \le R_C+C_C \le R_S+C_S$.

Figure 5(b) shows the extended search space when the proposed die-selection method additionally uses search-space condition 3 as well as search-space conditions 1 and 2 with the same state as that in Fig. 5(a). When die *B* in Fig. 5(a) has a single cell fault, it can be represented as die B_1 in Fig. 5(b). In a similar way, dies *A*, *C*, and *D* which have no single cell fault can be expressed as dies A_0, C_0, and D_0, respectively. It is assumed that the single cell fault of die *B* is repaired by a spare row. Therefore, the value of *SSR* is changed from –1 of die *B* to 0 of die B_1, as shown in Fig. 5. The *SSC* value of die B_1 in Fig. 5(b) is the same as that of die *B* in Fig. 5(a). In conclusion, die B_1 has more choices than die *B* does. As search-space condition 3 is additionally considered in the proposed die-selection method, the counterpart die C_0 can be newly selected for die B_1. If a single cell fault of die B_1 is repaired by a spare row after selection of the counterpart die, B_1 is expressed as B_R in Fig. 5(b). However, since die C_0 has no available spare rows but does have one available spare column, a single cell fault of die B_1 is repaired by a spare column of die C_0. Therefore, B_C in Fig. 5(b) is the appropriate repair solution for die B_1 in a 3D memory. Dies B_C and A_0 can be matched with dies C_0 and D_0, respectively. As a result, the proposed die-selection method using three search-space conditions can further increase the 3D memory yield.

Figure 6 shows the pseudo-code of the proposed die-selection method. SELECT_DIES represents the process of selecting dies to stack 3D memories. This function does not terminate until all dies are used. Initially, if an unused die exists, the target die (*t_die*) with the minimum sum of the number of available spares is selected at SELECT_T_DIE_MIN_SUM_AVAIL_SPA, and the selected target die is marked as used at MARK_USED. When the target die is determined, the presence of candidates for a counterpart die is analyzed using the three search-space conditions according to CHECK_SEARCH_SPACE_CONDITIONS. If there is at least one candidate die, the counterpart die (*c_die*) with the minimum sum of available spares within the limits is selected at SELECT_C_DIE_MIN_SUM_AVAIL_SPA, and the selected counterpart die is marked as used at MARK_USED. After the counterpart die is determined, the results for both the target die and the counterpart die are recorded.

```
SELECT_DIES() {
    while(unused die exists) {
        t_die = SELECT_T_DIE_MIN_SUM_AVAIL_SPA();
        MARK_USED(t_die);
        if(CHECK_SEARCH_SPACE_CONDITIONS(t_die)) {
            c_die = SELECT_C_DIE_MIN_SUM_AVAIL_SPA(t_die);
            MARK_USED(c_die);
            RECORD_RESULT(t_die, c_die);
        }
    }
}
```

Fig. 6. Pseudo-code of proposed die-selection method.

As stated above, the proposed method is explained for use in D2D integration technology. However, it can easily be extended for W2W integration technology or D2W integration technology by adding a constraint for the position of a die in a wafer.

IV. Simulation Results and Analysis

To estimate the performance of the die-selection method, we developed a simulation tool in C-language, called yield enhancement simulator (YES). The overall diagram of YES is shown in Fig. 7. Three kinds of input data are required to execute YES: general information, memory die information, and algorithm information. According to the information, YES generates faulty addresses at random. Then, memory dies are classified with their states after pre-bond test and repair. The classified memory dies are selected for 3D memory stacking. After evaluating the 3D memory yield by YES, following

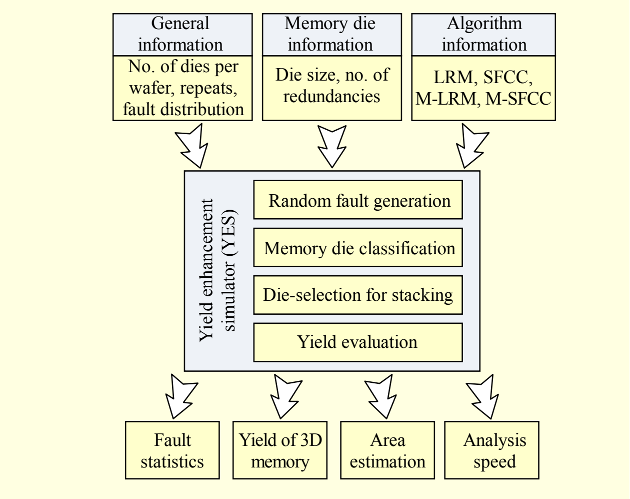
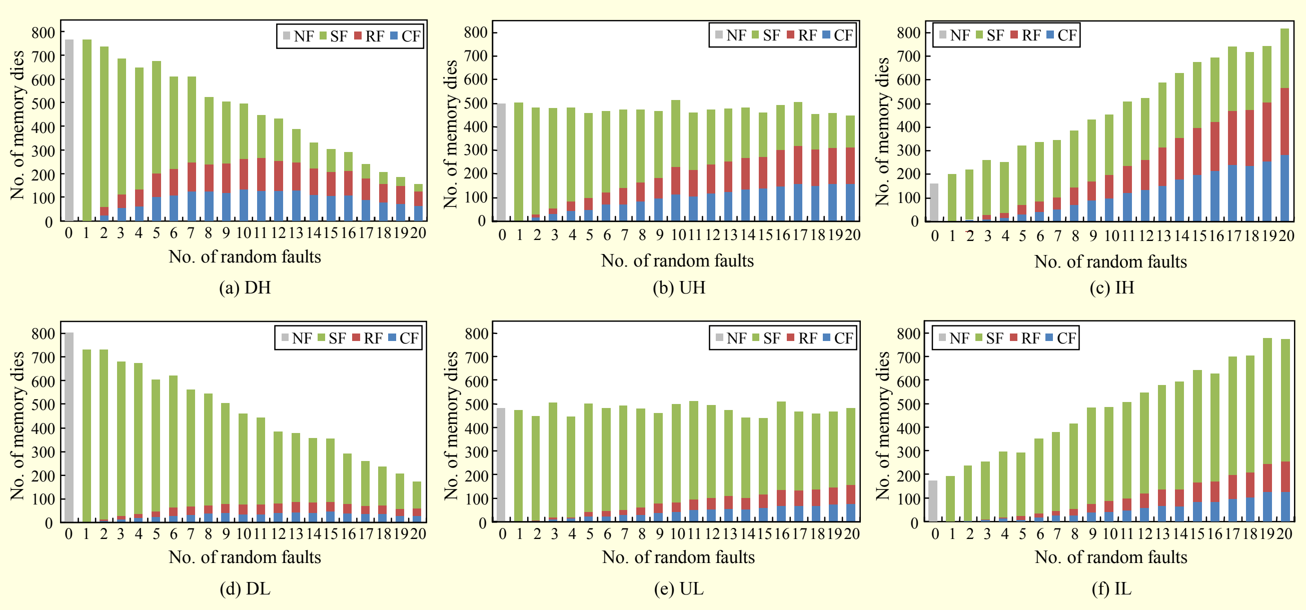


Fig. 7. Overview of simulation tool YES.



(a) DH

(b) UH

(c) IH

(d) DL

(e) UL

(f) IL

Fig. 8. Six random fault sets: DH, UH, IH, DL, UL, and IL.

output data is generated: fault statistics, yield of 3D memory, area estimation, and analysis speed.

The simulator YES can analyze four RA algorithms: local repair-most (LRM) algorithm [11], selected fail count comparison (SFCC) algorithm [13], modified local repair-most (M-LRM) algorithm [9], and modified selected fail count comparison (M-SFCC) algorithm. The LRM and SFCC algorithms are the traditional RA algorithms for 2D memory. These algorithms do not use inter-die redundancies. On the contrary, the M-LRM and M-SFCC algorithms use inter-die redundancies for repairing memory dies. The previous die-selection method [9] is evaluated by the LRM algorithm without inter-die redundancies and the M-LRM algorithm with inter-die redundancies. To evaluate the proposed die-selection method, the SFCC algorithm without inter-die redundancies and the M-SFCC algorithm with inter-die redundancies are utilized because they are able to classify single cell faults.

For the comparison of yields, we assumed nine cases of 1,024×1,024 memory dies with combinations of two to four spare rows and two to four spare columns. To evaluate the die-selection methods in a variety of situations, we prepared six random fault sets composed of combinations of the three fault distributions and the two types of fault density. The combinations of six random fault sets can cover the majority of important cases because six random fault sets present basic forms. When the number of injected faults in a die increases, the probability of the die's existence is decreased for decremental fault distribution and is unchanged for uniform fault distribution; however, it increases for incremental fault distribution. The two types of fault density are high density and low density. The fault distribution with high density has more line faults than that with low density. Six random fault sets are presented in Fig. 8. Each *x*-axis represents the number of injected random faults in a memory die, and each *y*-axis indicates the number of existing memory dies. When the number of random faults is 0, there is no fault (NF) in a memory die. In other cases, there are row line faults (RFs), column line faults (CFs), and single cell faults (SFs) in memory dies. For decremental fault distribution with high density (DH), 27.80% RFs, 27.44% CFs, and 44.77% SFs were injected into the simulated memory dies, as shown in Fig. 8(a). For decremental fault distribution with low density (DL), 9.84% RFs, 9.72% CFs, and 80.43% SFs were injected into the simulated memory dies, as shown in Fig. 8(d). In a similar way, additional fault sets are also shown in Fig. 8: uniform fault distribution with high density (UH) in Fig. 8(b), uniform fault distribution with low density (UL) in Fig. 8(e), incremental fault distribution with high density (IH) in Fig. 8(c), and incremental fault distribution with low density (IL) in Fig. 8(f). To provide accurate simulation results, each simulation was repeated 10,000 times with fault numbers from 0 to 20.

Table 1 summarizes the simulation results of 3D memory yields without (LRM, SFCC) and with inter-die redundancies (M-LRM, M-SFCC) between the previous die-selection method presented in [9] and the proposed die-selection method. In Table 1, *i*R/*j*C denotes a memory die with *i* spare rows and *j* spare columns. Since both *i* and *j* vary from 2 to 4, there are different nine cases for 3D memory stacking. From Table 1, we can observe that both the previous die-selection method and the proposed die-selection method with inter-die redundancies

Table 1. Comparison results of yields for 3D memories without and with inter-die redundancies between using method of [9] and proposed die-selection method.

	DH (27.80% RFs, 27.44% CFs, and 44.77% SFs)				UH (26.44% RFs, 26.23% CFs, and 47.34% SFs)				IH (28.29% RFs, 28.58% CFs, and 43.13% SFs)			
	[9]		Proposed		[9]		Proposed		[9]		Proposed	
	w/o	with	w/o	with	w/o	with	w/o	with	w/o	with	w/o	with
2R/2C	42.30 %	64.56 %	42.37 %	70.96 %	27.30 %	41.62 %	27.33 %	45.58 %	12.97 %	18.66 %	13.01 %	20.92 %
2R/3C	51.11 %	81.88 %	51.26 %	86.30 %	33.62 %	54.02 %	33.69 %	58.26 %	17.43 %	26.40 %	17.49 %	29.04 %
2R/4C	59.31 %	88.38 %	59.74 %	97.58 %	40.18 %	57.20 %	40.29 %	71.86 %	22.70 %	28.52 %	22.80 %	39.94 %
3R/2C	51.16 %	69.26 %	51.25 %	86.32 %	12.08 %	16.20 %	12.08 %	21.20 %	17.49 %	19.52 %	17.52 %	29.16 %
3R/3C	59.68 %	94.76 %	59.98 %	97.42 %	40.29 %	66.46 %	40.39 %	71.08 %	22.88 %	35.30 %	22.95 %	38.92 %
3R/4C	68.02 %	99.84 %	68.47 %	99.92 %	47.79 %	79.94 %	47.95 %	84.68 %	29.27 %	46.56 %	29.54 %	50.42 %
4R/2C	59.56 %	82.72 %	59.78 %	97.68 %	11.46 %	15.32 %	11.47 %	20.60 %	22.82 %	25.26 %	22.82 %	39.98 %
4R/3C	68.12 %	97.34 %	68.51 %	99.92 %	47.84 %	67.50 %	47.98 %	84.68 %	29.31 %	36.00 %	29.54 %	50.34 %
4R/4C	76.42 %	100 %	77.08 %	100 %	55.58 %	93.46 %	55.92 %	96.74 %	37.25 %	59.62 %	37.65 %	64.00 %
	DL (9.84% RFs, 9.72% CFs, and 80.43% SFs)				UL (11.44% RFs, 11.31% CFs, and 77.26% SFs)				IL (12.63% RFs, 12.29% CFs, and 75.09% SFs)			
	[9]		Proposed		[9]		Proposed		[9]		Proposed	
	w/o	with	w/o	with	w/o	with	w/o	with	w/o	with	w/o	with
2R/2C	37.43 %	56.98 %	37.43 %	59.48 %	24.60 %	37.36 %	24.60 %	39.74 %	12.05 %	17.36 %	12.05 %	18.90 %
2R/3C	44.35 %	69.86 %	44.35 %	71.04 %	30.04 %	47.64 %	30.04 %	49.92 %	15.52 %	23.54 %	15.52 %	25.22 %
2R/4C	51.03 %	73.96 %	51.03 %	81.46 %	35.28 %	49.56 %	35.29 %	60.44 %	19.69 %	24.38 %	19.71 %	32.92 %
3R/2C	44.35 %	60.28 %	44.35 %	71.04 %	30.04 %	38.66 %	30.04 %	49.92 %	15.52 %	18.00 %	15.52 %	25.22 %
3R/3C	51.03 %	79.84 %	51.04 %	81.46 %	35.29 %	57.28 %	35.30 %	60.46 %	19.69 %	30.76 %	19.71 %	32.92 %
3R/4C	57.39 %	89.90 %	57.39 %	90.30 %	41.14 %	68.44 %	41.15 %	71.16 %	24.14 %	38.76 %	24.15 %	41.64 %
4R/2C	51.03 %	68.34 %	51.04 %	81.44 %	35.29 %	44.98 %	35.29 %	60.46 %	19.68 %	21.26 %	19.70 %	32.92 %
4R/3C	57.39 %	83.96 %	57.39 %	90.30 %	41.13 %	58.12 %	41.14 %	71.16 %	24.14 %	30.92 %	24.15 %	41.64 %
4R/4C	63.40 %	96.02 %	63.46 %	96.60 %	46.87 %	78.94 %	46.89 %	82.20 %	29.74 %	48.52 %	29.76 %	51.56 %

significantly increase 3D memory yields when compared to the cases where 3D memories are manufactured without inter-die redundancies. Thus, the use of inter-die redundancies is very helpful to boost the yield of 3D memories.

The yield of the proposed die-selection method with inter-die redundancies is greater than or equal to that of the previous die-selection method with inter-die redundancies for each of the cases in Table 1. The previous die-selection method with inter-die redundancies is evaluated by the M-LRM algorithm, and the proposed die-selection method with inter-die redundancies is assessed by the M-SFCC algorithm. Although the different algorithms are used to judge the performance of the previous and the proposed die-selection methods with inter-die redundancies, the use of different algorithms can be ignored. In spite of the repair rate differences between the LRM algorithm and the SFCC algorithm, the results of the previous die-selection method without inter-die redundancies evaluated by the LRM algorithm is almost identical to that of the proposed die-selection method without inter-die redundancies assessed by the SFCC algorithm, as shown in Table 1. In other words, RA algorithms do not exert a strong influence on the yields of 3D memories using die-selection methods with inter-die redundancies. Therefore, the application of the three search-space conditions creates the meaningful yield enhancement between the proposed die-selection method and the previous die-selection method.

The areas for die-selection methods using four different RA algorithms (LRM, SFCC, M-LRM, and M-SFCC) are estimated by YES. When applying the BISR flow for each die-selection method, such as that for the proposed die-selection method in Fig. 3, the area of the BIRA dominates the area of the whole BISR. Also, the majority of the area of the BIRA is the area of storage requirements. We assume an M by N memory die using a 2D spare architecture with R_S spare rows and C_S spare columns. The equations to calculate storage requirements for BIRAs used in die-selection methods are shown below. A_{LRM}, A_{SFCC}, $A_{M\text{-}LRM}$, and $A_{M\text{-}SFCC}$ from (1) to (4) represent the number of bits required for each RA algorithm. A_{LRM} and A_{SFCC} are introduced in [11] and [13], respectively. $A_{M\text{-}LRM}$ [9] and $A_{M\text{-}SFCC}$ is obtained by modifying A_{LRM} and A_{SFCC}. All estimated area overhead equations consist of two kinds of variables: the number of spares (R_S and C_S) and the size of the memory die (M and N). We compare the required storage cells with the number of spares and the size of the memory die.

$$\begin{aligned}A_{LRM} &= m \times n \\ &\quad + m \times \left[\left(\lceil \log_2 M \rceil + 1 \right) + \lceil \log_2 (C_S + 1) \rceil \right] \\ &\quad + n \times \left[\left(\lceil \log_2 N \rceil + 1 \right) + \lceil \log_2 (R_S + 1) \rceil \right] \\ &\quad + R_S \times \left(\lceil \log_2 M \rceil + 1 \right) + C_S \times \left(\lceil \log_2 N \rceil + 1 \right), \quad (1)\end{aligned}$$

where $m = R_S + C_S \times (R_S + 1)$, $n = C_S + R_S \times (C_S + 1)$,

$$\begin{aligned}A_{SFCC} &= (R_S + C_S) \times \left(\lceil \log_2 M \rceil + \lceil \log_2 N \rceil + 1 \right) \\ &\quad + (R_S + C_S) \times \left(\lceil \log_2 R_S \rceil + \lceil \log_2 C_S \rceil \right) \\ &\quad + \left[R_S \times (C_S - 1) + C_S \times (R_S - 1) \right] \\ &\quad \times \left[\lceil \log_2 (\max(M, N)) \rceil + \lceil \log_2 (R_S + C_S) \rceil + 2 \right] \\ &\quad + R_S \times \left(\lceil \log_2 M \rceil + 1 \right) + C_S \times \left(\lceil \log_2 N \rceil + 1 \right), \quad (2)\end{aligned}$$

$$\begin{aligned}A_{M\text{-}LRM} &= p \times q \\ &\quad + p \times \left[\left(\lceil \log_2 M \rceil + 1 \right) + \lceil \log_2 (2C_S + 1) \rceil \right] \\ &\quad + q \times \left[\left(\lceil \log_2 N \rceil + 1 \right) + \lceil \log_2 (2R_S + 1) \rceil \right] \\ &\quad + 2R_S \times \left(\lceil \log_2 M \rceil + 1 \right) + 2C_S \times \left(\lceil \log_2 N \rceil + 1 \right), (3)\end{aligned}$$

where $p = 2R_S + 2C_S \times (2R_S + 1)$, $q = 2C_S + 2R_S \times (2C_S + 1)$,

$$\begin{aligned}A_{M\text{-}SFCC} &= (2R_S + 2C_S) \times \left(\lceil \log_2 M \rceil + \lceil \log_2 N \rceil + 1 \right) \\ &\quad + (2R_S + 2C_S) \times \left(\lceil \log_2 2R_S \rceil + \lceil \log_2 2C_S \rceil \right) \\ &\quad + \left[2R_S \times (2C_S - 1) + 2C_S \times (2R_S - 1) \right] \\ &\quad \times \left[\lceil \log_2 (\max(M, N)) \rceil + \lceil \log_2 (2R_S + 2C_S) \rceil + 2 \right] \\ &\quad + 2R_S \times \left(\lceil \log_2 M \rceil + 1 \right) + 2C_S \times \left(\lceil \log_2 N \rceil + 1 \right). \\ &\qquad (4)\end{aligned}$$

Since two-layer 3D memory is used in this paper, $A_{M\text{-}LRM}$ in (3) and $A_{M\text{-}SFCC}$ in (4) can be derived from applying twice as many redundancies as A_{LRM} in (1) and A_{SFCC} in (2), respectively. Area estimation with different spare columns for four different RA algorithms is presented in Fig. 9, and that with different memory die sizes is shown in Fig. 10. Based on the area estimation results generated by YES, the die-selection methods without inter-die redundancies (A_{LRM} and A_{SFCC}) have small storage requirements compared to those with inter-die redundancies ($A_{M\text{-}LRM}$ and $A_{M\text{-}SFCC}$), as shown in both Figs. 9 and 10. However, the yields of the die-selection methods without inter-die redundancies are much lower than that with inter-die redundancies. The area of the proposed die-selection method with inter-die redundancies is much smaller in all cases than that of the previous die-selection method [9] with inter-die redundancies, as shown in Figs. 9 and 10. Therefore, the proposed die-selection method has the advantage of area overhead.

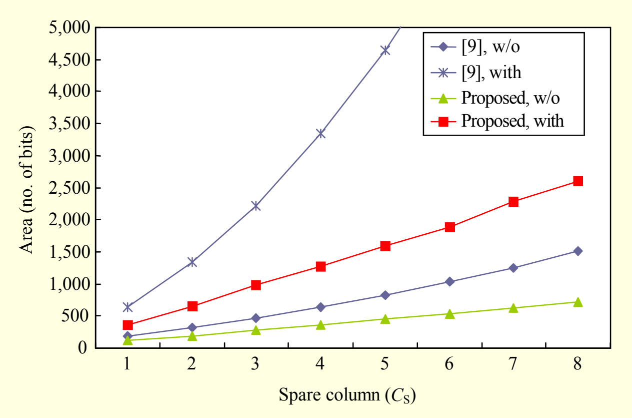


Fig. 9. Area estimation with different spare columns (M = 1,024, N = 1,024, and R_S = 2).

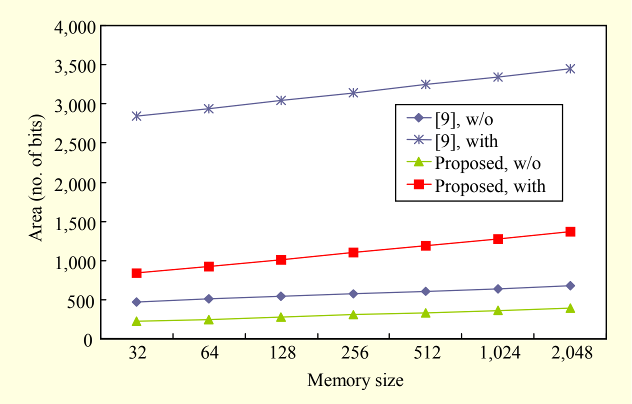


Fig. 10. Area estimation with different memory die sizes (R_S = 2 and C_S = 4).

For the last output data, analysis speed can be generated by YES. The analysis speed of the previous die-selection method using the LRM and the M-LRM algorithms is a bit faster than that of the proposed die-selection method using the SFCC and the M-SFCC algorithms, especially in the complex cases. However, although four RA algorithms used in YES have different time overheads, the analysis speed for both the previous die-selection method and the proposed die-selection method is almost identical regardless of inter-die redundancies. This is because the test time for executing the March test in the BIST is much longer than that for carrying out the other procedures for 3D memory stacking. The time overhead of the BIST strongly depends on the size of memory die. Therefore, the difference between each die-selection method can be negligible.

V. Conclusion

A die-selection method using three search-space conditions was proposed for yield enhancement in 3D memory with little waste of redundancies. The previous die-selection method exactly matches one die to another die with inter-die redundancies. However, the proposed die-selection method uses three search-space conditions to manufacture 3D memories. The use of search-space conditions, which consider the state of spare rows and spare columns in a die after pre-bond test and repair, increases the number of candidates for the counterpart memory dies. Since a single cell fault can be repaired by either a spare row or a spare column, the repair decision for single cell faults can be postponed until the other faults in a 3D memory have been determined. Simulation results showed that the proposed die-selection method in various fault distributions can further improve 3D memory yields. The area and the analysis time are also analyzed for the die-selection methods. In conclusion, the proposed die-selection method using three search-space conditions enhances the yield of 3D memories with feasible area and time overhead.

References

[1] V.F. Pavlidis and E.G. Friedman, "Interconnect-Based Design Methodologies for Three-Dimensional Integrated Circuits," *Proc. IEEE*, vol. 97, no. 1, Jan. 2009, pp. 123-140.

[2] H. Sun et al., "3D DRAM Design and Application to 3D Multicore Systems," *IEEE Design Test Comput.*, vol. 26, no. 5, Sept. 2009, pp. 36-47.

[3] W.R. Davis et al., "Demystifying 3D ICs: the Pros and Cons of Going Vertical," *IEEE Design Test Comput.*, vol. 22, no. 6, Nov. 2005, pp. 498-510.

[4] E.J. Marinissen and Y. Zorian, "Testing 3D Chips Containing Through-Silicon Vias," *Proc. Int. Test Conf.*, Nov. 2009, pp. 1-11.

[5] H.-H.S. Lee and K. Chakrabarty, "Test Challenges for 3D Integrated Circuits," *IEEE Design Test Comput.*, vol. 26, no. 5, Sept. 2009, pp. 26-35.

[6] S.S. Iyer et al., "Process-Design Considerations for Three Dimensional Memory Integration," *Proc. Symp. VLSI Tech.*, June 2009, pp. 60-63.

[7] Y.-F. Chou, D.-M. Kwai, and C.-W. Wu, "Memory Repair by Die Stacking with Through Silicon Vias," *Proc. Int. Workshop Memory Tech., Design, and Testing*, Aug. 2009, pp. 53-58.

[8] L. Jiang, R. Ye, and Q. Xu, "Yield Enhancement for 3D-Stacked Memory by Redundancy Sharing across Dies," *Proc. Int. Conf. Comput.-Aided Design*, Nov. 2010, pp. 230-234.

[9] C.-W. Chou, Y.-J. Huang, and J.-F. Li, "Yield-Enhancement Techniques for 3D Random Access Memories," *Proc. Int. Symp. VLSI Design Automat. Test*, Apr. 2010, pp. 104-107.

[10] T. Yamagata et al., "A Distributed Globally Replaceable Redundancy Scheme for Sub-Half-Micron ULSI Memories and Beyond," *IEEE J. Solid-State Circuits*, vol. 31, no. 2, Feb. 1996, pp. 195-201.

[11] C.-T. Huang et al., "Built-In Redundancy Analysis for Memory Yield Improvement," *IEEE Trans. Relia.*, vol. 52, no. 4, Dec. 2003, pp. 386-399.

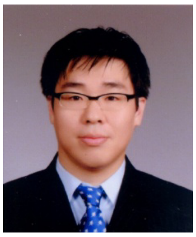
[12] M.-H. Yang et al., "A Novel BIRA Method with High Repair Efficiency and Small Hardware Overhead," *ETRI J.*, vol. 31, no. 3, June 2009, pp. 339-341.

[13] W. Jeong et al., "A Fast Built-In Redundancy Analysis for Memories with Optimal Repair Rate Using a Line-Based Search Tree," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 17, no. 12, Dec. 2009, pp. 1665-1678.

[14] H. Cho, W. Kang, and S. Kang, "A Built-In Redundancy Analysis with a Minimized Binary Search Tree," *ETRI J.*, vol. 32, no. 4, Aug. 2010, pp. 638-641.

[15] T. Han et al., "High Repair Efficiency BIRA Algorithm with a Line Fault Scheme," *ETRI J.*, vol. 32, no. 4, Aug. 2010, pp. 642-644.

[16] W. Jeong et al., "An Advanced BIRA for Memories with an Optimal Repair Rate and Fast Analysis Speed by Using a Branch Analyzer," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 12, Dec. 2010, pp. 2014-2026.



Joohwan Lee received the BS and MS in electrical and electronic engineering from Yonsei University, Seoul, Rep. of Korea in 2003 and 2005, respectively. He was a research engineer with the ASIC Research Center, Yonsei University. He is currently working toward the PhD in electrical and electronic engineering at Yonsei University. His current research interests include fault simulation, fault diagnosis, built-in self-repair (BISR), built-in self-test (BIST), built-in redundancy analysis (BIRA), RA algorithm, reliability, and very large scale integration (VLSI) design.

Kihyun Park received the BS in electrical and electronic engineering from Yonsei University, Seoul, Rep. of Korea, in 2007. He was also a research engineer with the ASIC Research Center, Yonsei University. He is currently working toward a combined MS/PhD in electrical and electronic engineering at Yonsei University. His current research interests include built-in self-repair (BISR), built-in self-test (BIST), built-in redundancy analysis (BIRA), test algorithms, and very large scale integration (VLSI) design.

Sungho Kang received the BS from Seoul National University, Seoul, Rep. of Korea, and the MS and PhD in electrical and computer engineering from the University of Texas at Austin in 1992. He was a research scientist with the Schlumberger Laboratory for Computer Science, Schlumberger Inc., and a senior staff engineer with the Semiconductor Systems Design Technology, Motorola Inc. Since 1994, he has been a professor with the Department of Electrical and Electronic Engineering, Yonsei University, Seoul, Rep. of Korea. His main research interests include VLSI design and testing, design for testability, BIST, defect diagnosis, and design for manufacturability.