

# 基于相邻多层冗余共享的三维存储器堆叠方法

刘 军<sup>1,2</sup>, 李 进<sup>1,2</sup>, 王 伟<sup>1,2</sup>, 任福继<sup>3</sup>

(1 合肥工业大学 计算机与信息学院, 安徽 合肥 230009;

2 情感计算与先进智能机器安徽省重点实验室, 安徽 合肥 230009;

3 日本德岛大学 先端技术科学教育部, 日本 德岛 7708502)

**摘 要:** 为提高三维存储器的成品率和减少冗余行列所需的 TSVs 的数量, 提出每一层存储块都与相邻上下两层共享冗余的三维存储器冗余共享结构, 并给出了一种新的堆叠方案, 将故障较多与较少的芯片交替堆叠. 实验结果表明, 该结构在有效控制 TSVs 数量增长的前提下, 提高了三维存储器的修复率.

**关键词:** 三维存储器; 成品率提高; 冗余共享; 堆叠方法

**中图分类号:** TP391

**文献标识码:** A

**文章编号:** 1000—7180(2017)07—0001—06

## A Stacking Scheme Using Adjacent Redundancy Across Dies for 3D-stacked Memory

LIU Jun<sup>1,2</sup>, LI Jin<sup>1,2</sup>, WANG Wei<sup>1,2</sup>, REN Fu-ji<sup>3</sup>

(1 School of Computer and Information, Hefei University of Technology, Hefei 230009, China;

2 Anhui Province Key Laboratory of Affective Computing and Advanced Intelligent Machine,

Hefei 230009, China; 3 Department of Information Science & Intelligent Systems,

Faculty of Engineering, The University of Tokushima, Tokushima 7708502, Japan)

**Abstract:** To improve the yield and the number of TSVs used in spare rows and columns, this paper proposed a sharing structure of adjacent redundancy across dies. Besides, based on the new redundancy sharing structure, a die-selection method stacking alternative memory layers with different number of faults was also presented. Experimental results show that the proposed method can significantly increase memory yield with relatively small area overhead of TSVs.

**Key words:** 3D memory; yield enhancement; redundancy sharing; die-selection method

### 1 引言

随着集成电路制造工艺的发展, 通过垂直集成来提高电路集成度的三维芯片设计, 已成为当前半导体行业发展最快的技术之一<sup>[1]</sup>.

随着集成电路行业的发展, 芯片中嵌入式存储器成为 SoC 的主要组成部分<sup>[2-3]</sup>. 因此如何修复存储器中的故障单元成为提高其成品率的关键技术之一.

存储器修复技术, 就是采用冗余单元来替换故障单元, 从而达到修复故障的目的. 在三维存储器

中, 采用垂直集成的方法, 能有效缩短各芯片之间的布线长度, 减少数据传输延时, 使得三维存储器中不同层芯片之间共享冗余单元修复故障变得可行, 从而大大提高了冗余单元的利用率, 提高了三维存储器的成品率.

根据不同的冗余替换策略, 三维存储器的修复方法大致分为三类: 全局冗余(Global Redundancy)共享、层间冗余(Inter-Layer Redundancy)共享和整层冗余(Layer Redundancy)替换. 全局冗余共享的方法中<sup>[4-5]</sup>, 面积开销比较大. 层间共享冗余的修复

方法,通过在不同层之间利用 TSVs 建立数据连接,使得本层的冗余单元,不仅能替换本层的故障单元,还可以被其它层利用<sup>[6]</sup>.但层间冗余方法的成品率要低于全局冗余的成品率.整层冗余替换的方法虽然该方案的硬件开销较小,但该方案会造成较大的资源浪费,并不能有效提高三维存储器的成品率.

本文提出了一种基于相邻层的冗余共享策略,每层芯片与其上下相邻层之间均通过 TSVs 连接共享冗余,替换修复故障单元.并在此基础上提出了一种新的堆叠方法,将故障数目较多的芯片与较少的交替堆叠,最终形成的三维存储器堆叠结构中,故障数目较多的芯片均能从相邻层获取冗余修复自身.所提方法的硬件开销较小,而且由于每层芯片均能利用相邻上下层的冗余来修复自身故障单元,更加有效地利用了三维存储器的冗余单元,提高了成品率.

## 2 背景介绍

### 2.1 存储器修复

为了提高存储器的成品率,一般采用增加冗余单元通过地址重映射的方法来替换存储器中的故障单元的修复方法<sup>[7-8]</sup>.图 1 所示为一种利用可编程逻辑器件直接重定向故障单元地址的电路结构.当存储单元 01 出现故障时,则将原本存储单元 01 前的电路断开,其地址信息写到冗余编码 0 中;当访问存储单元 01 时,则地址和冗余编码 0 中信息一致,通过  $hit_0$  激活冗余单元 00 进行相关读写操作,实现了故障单元的替换<sup>[9-10]</sup>.

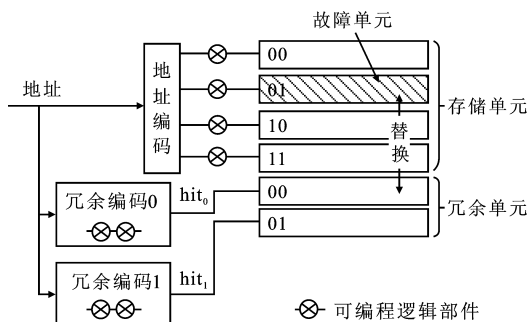


图 1 一种冗余替换的修复结构

### 2.2 三维芯片的堆叠

三维芯片有三种堆叠方式<sup>[4,6,11]</sup>:裸片到裸片(D2D, Die-to-Die)堆叠,裸片到晶圆(D2W, Die-to-Wafer)堆叠,晶圆到晶圆(W2W, Wafer-to-Wafer)堆叠. W2W 的方法简单易行,能大批量绑定生成三维芯片,但无法通过有效的堆叠策略来提高冗余的利用效

率,因此 W2W 堆叠的成品中,芯片修复率低,资源浪费较大.相比之下,D2W 和 D2D 的堆叠方式中,可以按照一定的算法来挑选合适的芯片进行绑定,成品率会大大提高.本文所提出的冗余共享修复策略是用来修复使用 D2D 堆叠方式的三维存储器.

## 3 本文方案

### 3.1 冗余共享结构

本文提出的相邻多层冗余共享的三维存储器修复策略的如图 2 所示.在图 2 中,假设三维存储器由四块存储块堆叠而成.位于中间两层的存储块,可以共享上下相邻两层的冗余,

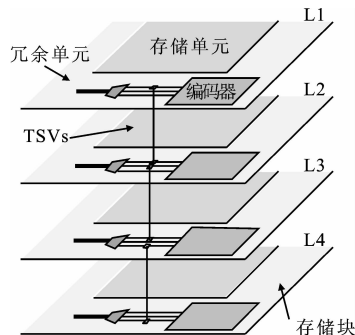


图 2 相邻多层冗余共享修复结构

第 3 层存储块可以使用第 2 层和第 4 层的即图中第 2 层存储块可以使用第 1、第 3 层的冗余;而位于最底层和最高层的存储块,只能使用与其相邻的一层存储块的冗余,即图中第 1 层存储块可以使用第 2 层的冗余,第 4 层存储块只能使用第 3 层的冗余.

在本文所提出的三维存储器冗余共享结构中,由于每一层存储块的冗余只需与其上下相邻层之间通过 TSVs 实现冗余共享,因此与文献[5]中的全局冗余共享策略比较,相邻多层冗余共享结构能够有效地减少 TSVs 的数量,降低硬件面积开销.与文献[6]中的相邻两层存储单元之间共享冗余的方案相比较,本文方法能够有效地提高冗余的利用率,提高三维存储器成品率.

### 3.2 芯片堆叠策略

在相邻多层冗余共享结构中,某一层是否能够被修复,不仅和自身的故障信息、冗余数量等有关,还与其相邻层是否有足够的冗余有关.因此,只有采取恰当的芯片堆叠策略,才能保证充分利用各层冗余,提高三维存储器成品率.

针对相邻多层冗余共享结构,本文提出了一种故障较多存储块与故障较少存储块交替堆叠的策

略. 所提方法从三维存储器的中间层开始, 依次选择上下相邻层进行堆叠, 直到完成整个三维存储器的堆叠.

在进行三维存储器的绑定前首先要剔除不可被修复的存储块. 由于每一层存储块能连接并利用的冗余单元有限, 若修复自身所需要的冗余数大于该存储块所能利用的所有冗余数目, 则可以认为此存储块不可修复. 本文的三维存储器冗余共享结构中, 除最底层和最高层存储块外, 其余各层均能利用自身和上下相邻两层冗余来修复自身故障. 若设每个存储块有  $R$  个冗余, 则在三维存储器结构中, 一个存储块最多可以使用  $3R$  个冗余. 因此, 当某一存储块进行故障修复时所需要的冗余量超过  $3R$ , 则此存储块是不可被修复的, 需要剔除.

将不可以修复的存储块剔除掉以后, 接着需要进行可修复存储块的堆叠. 假设剔除不可被修复存储块后, 剩余存储块可堆叠成为  $N$  个三维存储器, 每个三维存储器的层数为  $M(M > 2)$ , 本文所提的堆叠流程如图 3 所示.

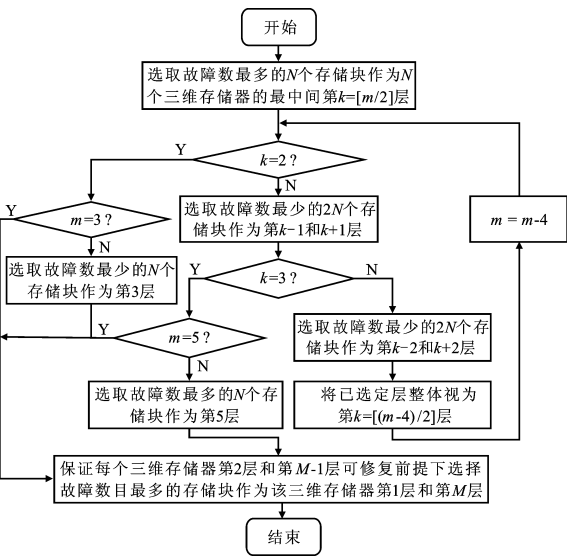


图 3 三维存储器堆叠过程

第一步是确定哪些存储块作为三维存储器的最中间层. 令  $m = M, k = \lceil m/2 \rceil$ , 选取可修复存储块中故障数目最多的  $N$  个存储块依次作为  $N$  个三维存储器的第  $k$  层. 由于中间层能与相邻的上下两层共享冗余, 相对于最外层而言能被修复的可能性更大, 此时选择故障数目最少的  $2N$  个存储块作为第  $k$  层的相邻两层, 即第  $k-1$  层和第  $k+1$  层, 能最大程度保证故障数目最多的存储块可以被修复. 此时第  $k-1$  层和第  $k+1$  层存储块故障数目少, 能向相邻层提供冗余修复故障, 所以选取余下故障数目最多的

$2N$  个存储块作为向外的第  $k-2$  层和第  $k+2$  层.

至此已经堆叠了五层, 本文将此五层存储块视为一个整体作为三维存储器的一层, 此时三维存储器的层数  $m = m-4$ , 该层为第  $k = \lceil (m-4)/2 \rceil$  层. 再依照前文的堆叠策略选取其它层存储块, 直至  $N$  个三维存储器除最底层和最高层外都已选定.

若三维存储器的层数较少, 不足以完成以上堆叠的整个过程时, 则在其堆叠过程中, 一旦发现选择的存储块在三维存储器的最外层, 立即停止堆叠, 进入后续步骤.

完成以上步骤以后开始堆叠最外层. 对于每一个三维存储器, 此时第  $2, 3, \dots, M-1$  层都已经确定, 根据第 2 层存储块的故障信息和可用冗余数量, 在保证第 1 层和第 2 层可修复, 即这两层可用冗余不少于修复故障所需冗余数的前提下, 选取余下故障数目最多的存储块作为第 1 层. 同理选择合适的存储块作为第  $M$  层, 完成整个三维存储器的堆叠.

通过一个例子来说明本文所提的芯片堆叠过程. 假设有 996 个存储块, 每个存储块中冗余数量  $R = 2$ , 存储块内的故障数目及对应的存储块数量如表 1 所示<sup>[12]</sup>.

表 1 模拟故障信息

| 存储块内故障数目 | 存储块数目 |
|----------|-------|
| 0        | 222   |
| 1        | 334   |
| 2        | 251   |
| 3        | 125   |
| 4        | 47    |
| 5        | 14    |
| 6        | 3     |

假设三维存储器层数为 6, 即每个三维存储器需要由 6 个存储块绑定而成, 则表 1 中的 996 个存储块绑定后能形成 166 个三维存储器. 依照图 3 的流程, 具体堆叠过程如下:

(1) 确定中间层存储块. 三维存储器层数为 6, 中间层为第三层, 即  $k = 3$ . 从表 1 中取出故障数目最多的 166 个存储块作为 166 个三维存储器的第三层, 如图 4(a) 所示. 在这 166 个存储块中, 有 3 个包含 6 个故障的存储块, 14 个包含 5 个故障的存储块, 47 个包含 4 个故障的存储块, 102 个包含 3 个故障的存储块.

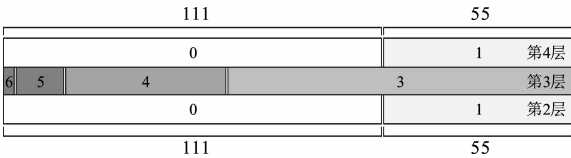
(2) 选取第二至第五层存储块. 因为  $k-1 = 2 \neq 1$  且  $k+1 = 4 \neq 6$ , 此时第  $k$  层的相邻两层都不是最边缘层, 因此选取余下存储块中故障数目最少的 332 个存储块, 分别作为这 166 个三维存储器的第二

层和第四层,如图 4(b) 所示. 接着选取余下存储块中故障数目最多 166 个存储块的作为这 166 个三维存储器的第 5 层即可完整所有中间层的堆叠过程,结果如图 4(c) 所示.

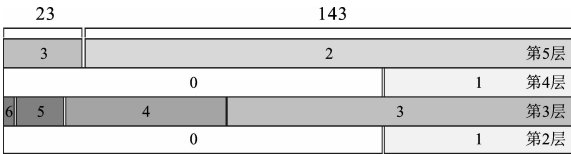
(3)选取最高层和最底层的存储块,完成堆叠. 如图 4(d)所示,在剩余的存储块中,选择所有满足修复条件的存储块中故障数目较多的存储块作为三维存储器的最高和最低层,从而达到充分利用冗余单元的目的,提高整体修复率,结果如图 4(d)所示.



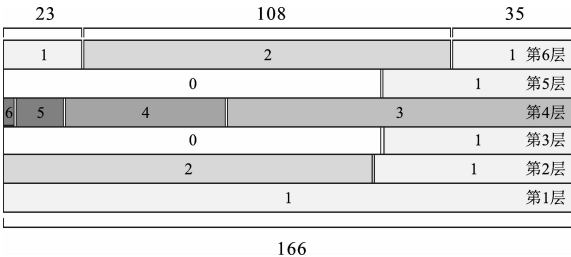
(a)确定中间第 3 层



(b)选取第 2 层和第 4 层



(c)选取第 5 层



(d)完成堆叠绑定

图 4 三维存储器堆叠过程

996 个存储块的最终堆叠结果如表 2 所示. 表 2 中第一列表示三维存储器的个数,第二列堆叠结构中的数字从左到右依次为三维存储器从最底层到最高层存储块所含的故障数目,第三列表示这些三维存储器中的含有不同故障数目的存储块的个数. 如“1-0-6-0-3-1”表示这个三维存储器从第 1 层到第 6 层存储块中的故障数目依次为 1、0、6、0、3、1,且这样的三维存储器有 3 个;这 3 个三维存储器中,包含了 3 块有 6 个故障的存储块、3 块有 3 个故障的存储块、6 块有 1 个故障的存储块以及 6 个零故障存储块. 从表中可以看出,对于给定的 996 个存储块,本文所提三维存储器冗余共享修复的结构以及对应的堆叠策略,能完成所有故障存储块的绑定和修复.

表 2 堆叠绑定结果

| 三维存储器个数 | 堆叠结构        | 不同故障数的存储块数量 |    |    |     |     |     |     |
|---------|-------------|-------------|----|----|-----|-----|-----|-----|
|         |             | 6           | 5  | 4  | 3   | 2   | 1   | 0   |
| 3       | 1-0-6-0-3-1 | 3           | 0  | 0  | 3   | 0   | 6   | 6   |
| 14      | 1-0-5-0-3-1 | 0           | 14 | 0  | 14  | 0   | 28  | 28  |
| 6       | 1-0-4-0-3-1 | 0           | 0  | 6  | 6   | 0   | 12  | 12  |
| 41      | 1-0-4-0-2-2 | 0           | 0  | 41 | 0   | 82  | 41  | 82  |
| 47      | 1-0-3-0-2-2 | 0           | 0  | 0  | 47  | 94  | 47  | 94  |
| 20      | 1-1-3-1-2-2 | 0           | 0  | 0  | 20  | 40  | 60  | 0   |
| 35      | 1-1-3-1-2-1 | 0           | 0  | 0  | 35  | 35  | 140 | 0   |
| 166     | -           | 3           | 14 | 47 | 125 | 251 | 334 | 222 |

4 实验结果及分析

为验证本文所提方案的有效性,本部分将本文所提方案的实验结果与两层冗余共享、全局冗余共享的结果进行了比较. 在实验中,假设芯片中故障分布服从如下的 Polya-Eggenberger 模型<sup>[13-14]</sup>:

$$P(K) = \frac{\Gamma(K + \alpha) \left(\frac{\lambda}{\alpha}\right)^K}{K! \Gamma(\alpha) \left(1 + \frac{\lambda}{\alpha}\right)^{(\alpha+K)}} \quad (1)$$

其中  $P(K)$  表示芯片中有  $K$  个故障的概率; $\lambda$  表示平均每个芯片中包含的故障个数; $\alpha$  表示故障密度, $\alpha$  值越大,说明故障分布越集中,反之则越分散,本文实验中取  $\alpha = 2.382$ .

成品率是三维存储器的重要指标,计算如式(2)所示. 成品率表示绑定后可修复三维存储器个数与总体数目的比率,能有效反映三维存储器的冗余共享结构设计以及堆叠策略在故障修复方面的有效性.

$$\text{成品率} = \frac{\text{可修复三维存储器数目}}{\text{总三维存储器数目}} \times 100\% \quad (2)$$

本文分别考虑不同故障数、冗余数量和三维存储器堆叠层数等变化因素,对本文提出的相邻多层冗余共享修复的三维存储器结构和文献[6] 中两层冗余共享以及文献[5] 中全局冗余共享策略的成品率的影响. 表 3 给出了在不同故障数目、不同冗余数量的情况下,三维存储器层数分别为 4 层、6 层和 8 层时,本文方法以及两层冗余共享、全局冗余共享结构的成品率的结果,其中  $F_{\max}$  表示芯片中注入的最大故障数量, $F_{\text{avg}}$  表示所有芯片中的平均故障个数.

由表 3 可以看出,本文所提出相邻多层冗余共

享结构的三维存储器的成品率高于两层冗余共享结构. 由于相邻多层冗余共享结构中,有故障的存储块能利用的冗余数量,比两层冗余共享结构能利用的多,能修复更过的故障,因此在 $F_{\max}$ 和 $F_{\text{avg}}$ 以及冗余数量相等的情况下,相邻多层冗余结构的成品率始

终比两层冗余共享结构高. 在全局冗余共享结构中,整个三维存储器中的冗余,都通过 TSVs 连接到每一个存储块上,其冗余可利用率已经达到最大,因此成品率最高. 表中数据表明,本文所提方案与全局冗余共享结构的成品率比较接近.

表 3 三维存储器成品率仿真实验结果

| 故障数目       |                  |     | 冗余数量  | 不同绑定层数的成品率/% |       |       |       |       |       |       |       |  |
|------------|------------------|-----|-------|--------------|-------|-------|-------|-------|-------|-------|-------|--|
|            |                  |     |       | 4 层          |       |       | 6 层   |       |       | 8 层   |       |  |
| $F_{\max}$ | $F_{\text{avg}}$ | R/C | 本文方法  | 两层共享         | 全局共享  | 本文方法  | 两层共享  | 全局共享  | 本文方法  | 两层共享  | 全局共享  |  |
| 10         | 2.441            | 1   | 58.33 | 57.54        | 59.52 | 59.04 | 58.43 | 60.24 | 57.94 | 57.14 | 59.52 |  |
|            |                  | 2   | 90.08 | 82.94        | 90.87 | 90.96 | 83.73 | 92.17 | 89.68 | 82.54 | 90.48 |  |
| 10         | 3.217            | 2   | 72.22 | 71.03        | 73.02 | 72.89 | 71.69 | 74.10 | 72.22 | 70.63 | 73.02 |  |
|            |                  | 3   | 94.84 | 87.30        | 95.63 | 95.78 | 88.55 | 96.99 | 94.44 | 87.30 | 95.24 |  |
| 15         | 4.320            | 3   | 78.17 | 76.59        | 78.97 | 78.92 | 77.11 | 80.12 | 77.78 | 76.19 | 78.57 |  |
|            |                  | 4   | 94.84 | 87.30        | 95.63 | 95.78 | 88.55 | 96.39 | 94.44 | 87.30 | 95.24 |  |
| 15         | 5.093            | 4   | 84.13 | 81.35        | 84.92 | 84.94 | 82.53 | 85.54 | 84.13 | 80.95 | 84.92 |  |
|            |                  | 5   | 97.22 | 89.68        | 98.02 | 98.19 | 90.96 | 99.40 | 96.83 | 89.68 | 97.62 |  |
| 20         | 6.224            | 5   | 86.11 | 82.54        | 86.51 | 86.75 | 83.13 | 87.35 | 85.71 | 82.54 | 86.51 |  |
|            |                  | 6   | 96.83 | 88.89        | 98.41 | 98.19 | 89.76 | 99.40 | 96.83 | 88.89 | 98.41 |  |
| 25         | 8.074            | 7   | 90.08 | 85.71        | 90.87 | 90.96 | 86.75 | 91.57 | 89.68 | 85.71 | 90.48 |  |
|            |                  | 8   | 98.02 | 90.48        | 98.41 | 99.40 | 91.57 | 99.40 | 97.62 | 90.48 | 98.41 |  |

对于全局冗余共享结构,尽管能充分利用冗余,对故障芯片的修复率高,但此种结构将每层之间都通过 TSVs 连接,导致三维存储器中 TSVs 数量较多,硬件开销比较大. 并且 TSVs 作为三维存储器中各层之间的数据传输载体,制造工艺复杂,故障率较高. 因此有效控制 TSVs 数量,不仅能避免三维存储器中的非存储单元的面积开销过大,还能减少 TSVs 的测试修复成本.

本文所提的相邻多层冗余共享的方法,与全局冗余共享方案相比,能有效地减少 TSVs 数量,降低面积开销. 假设每一层存储块的冗余数量为 1,则对于本文以及文献[6]中两层冗余共享结构和文献[5]中全局冗余共享结构,TSVs 数量随着三维存储器层数的增加而变化的结果如图 5 所示. 从图中可以看出,全局共享结构中 TSVs 数量随着层数的增加而快速增大,当三维存储器层数较多时,会造成大量的面积开销. 而本文所提出的相邻多层冗余共享的修复结构,随着三维存储器层数的增加而增多的 TSVs 数量相对较少. 从图中可以看出,当三维存储器层数到达 8 层时,本文所提方法的 TSVs 数量只比两层冗余共享结构多 4,而此时全局冗余共享的

方法所需要的 TSVs 数量远远高于前两者. 可以说,本文所提出的相邻多层冗余共享的修复结构,能很好的控制 TSVs 数量,降低硬件面积开销.

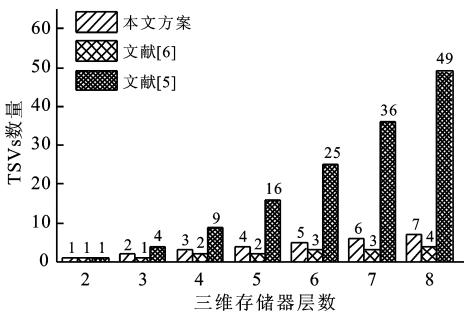


图 5 TSVs 数量对比实验结果

5 结束语

本文提出了一种相邻多层共享修复的三维存储器冗余共享修复策略,在提高三维存储器成品率的同时减少了 TSVs 的数量,降低了硬件开销. 在本文所提出的三维存储器的冗余共享结构中,每一层存储块都与相邻的上下两层共享冗余,提高了三维存储器的成品率. 针对这种相邻多层冗余共享结构,还提出了一种新的芯片堆叠方法,将故障数目较多与

较少的芯片交替堆叠,以达到充分利用冗余来修复芯片故障的目的.实验结果表明,与其它的冗余共享修复策略相比,本文所提方案在提高成品率和降低硬件面积开销方面效果明显.

## 参考文献:

- [1] Banerjee K, Souri S J, Kapur P, et al. 3-D ICs: a novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration [J]. *Proceedings of the IEEE*, 2001, 89(5): 602-633.
- [2] Kiyoo Itoh B S. Embedded memories: progress and a look into the future [J]. *Design & Test of Computers*, IEEE, 2011, 28(1): 10-13.
- [3] Allan A, Edenfeld D, Joyner W H, et al. 2001 technology roadmap for semiconductors [J]. *Computer*, 2002, 35(1): 42-53.
- [4] Lee J, Park K, Kang S. A die-selection method using search-space conditions for yield enhancement in 3d memory [J]. *Etri Journal*, 2011, 33(6): 904-913.
- [5] Lee J, Park K, Kang S. Yield enhancement techniques for 3d memories by redundancy sharing among all layers [J]. *Etri Journal*, 2012, 34(3): 388-98.
- [6] Li J, Rong Y, Qiang X. Yield enhancement for 3D-stacked memory by redundancy sharing across dies [C] // *Proceedings of the Computer-Aided Design (IC-CAD)*, 2010 IEEE/ACM International Conference on. San Jose, 2010: 230-234.
- [7] Schuster S E. Multiple word/bit line redundancy for semiconductor memories [J]. *Solid-State Circuits*, IEEE Journal of, 1978, 13(5): 698-703.
- [8] Zorian Y, Shoukourian S. Embedded-memory test and repair: infrastructure IP for SoC yield [J]. *Design & Test of Computers*, IEEE, 2003, 20(3): 58-66.
- [9] Kuo S Y, Fuchs W K. Efficient spare allocation for reconfigurable arrays [J]. *Design & Test of Computers*, IEEE, 1987, 4(1): 24-31.
- [10] Chin-Long W, Lombardi F. On the repair of redundant RAMs [J]. *Computer-Aided Design of Integrated Circuits and Systems*, IEEE Transactions on, 1987, 6(2): 222-231.
- [11] Che Wei C, Yu Jen H, Jin Fu L. Yield-enhancement techniques for 3D random access memories [C] // *Proceedings of the VLSI Design Automation and Test (VLSI-DAT)*, 2010 International Symposium on. Hsinchu, Taiwan 2010: 104-7.
- [12] Rab M T, Bawa A A, Toubia N A. Reducing cost of yield enhancement in 3-d stacked memories via asymmetric layer repair capability [J]. *Ieee T Vlsi Syst*, 2014, 22(9): 2017-2024.
- [13] Stapper C H. On a composite model to the IC yield problem [J]. *Solid-State Circuits*, IEEE Journal of, 1975, 10(6): 537-539.
- [14] Stapper C H, McLaren A N, Dreckmann M. Yield model for productivity optimization of vlsi memory chips with redundancy and partially good product [J]. *IBM Journal of Research and Development*, 1980, 24(3): 398-409.

## 作者简介:

刘 军 男,(1978-),副教授,硕士生导师.研究方向为数字系统设计自动化、集成电路测试、嵌入式系统等.

李 进(通讯作者) 男,(1992-),硕士研究生.研究方向为三维存储器测试修复等. E-mail:ahlijin@163.com.

王 伟 男,(1974-),副教授,硕士生导师.研究方向为微系统、嵌入式系统、低功耗技术等.

任福继 男,(1959-),教授,博士生导师.研究方向为人工智能、情感机器人、自然语言理解等.