



(12) 发明专利

(10) 授权公告号 CN 101379566 B

(45) 授权公告日 2014. 06. 25

(21) 申请号 200680049881. X

(22) 申请日 2006. 12. 14

(30) 优先权数据

11/322, 988 2005. 12. 30 US

(85) PCT国际申请进入国家阶段日

2008. 06. 30

(86) PCT国际申请的申请数据

PCT/US2006/047717 2006. 12. 14

(87) PCT国际申请的公布数据

W02007/078830 EN 2007. 07. 12

(73) 专利权人 英特尔公司

地址 美国加利福尼亚州

(72) 发明人 M · J · 德姆普塞 J · A · 迈斯

(74) 专利代理机构 中国专利代理(香港)有限公司 72001

代理人 柯广华 王丹昕

(51) Int. Cl.

G11C 29/00 (2006. 01)

(56) 对比文件

US 2004255209 A1, 2004. 12. 16,

US 6006311 A, 1999. 12. 21,

US 6671822 B1, 2003. 12. 30, 全文.

FONG J Y 等. Nonvolatile Repair

Caches Repair Embedded SRAM and New
Nonvolatile Memories. 《DEFECT AND FAULT
TOLERANCE IN VLSI SYSTEMS, 2004》. DFT
2004. PROCEEDINGS. 19TH IEEE INTERNATIONAL
SYMPOSIUM ON CANNES, FRANCE 10-13 OCT.
2004, 2004, 247-255.

审查员 刘渊

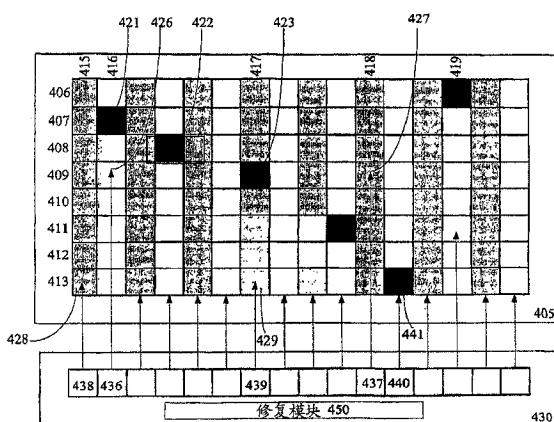
权利要求书3页 说明书10页 附图8页

(54) 发明名称

用于修复高速缓存阵列中单元的装置、系统
和方法

(57) 摘要

本文描述用于修复高速缓存存储器 / 阵列的方法和装置。高速缓存包括多个行，并且在逻辑上可见呈多个列。耦合到该高速缓存的修复高速缓存包括映射到每个在逻辑上可见的列的修复位。修复模块基于任何个别因素或因素组合来确定列中待修复的坏位，这些因素如高速缓存的每行中的错误数量、高速缓存的每行中可以通过错误校正码 (ECC) 校正的错误数量、位故障率或其它考虑因素。在访问包括坏位的高速缓存行时，通过映射到包括坏位的列的修复位来透明地修复坏位。



1. 一种用于修复高速缓存阵列中单元的装置，包括：
具有多个字的高速缓存，所述高速缓存在逻辑上可见呈多个列；
具有多个修复路的修复高速缓存，所述多个修复路中的每一修复路与所述多个列中的一个列关联；
修复模块，用于以所述多个修复路中的与包含第一位的第一列关联的第一路中的修复位来修复第一位，其中所述修复模块动态地确定待修复的第一位，并且确定待修复的第一位基于优化所述多个字中的每个字的错误校正以及第一位的故障率。
2. 如权利要求 1 所述的装置，其中所述高速缓存实现 1 位错误校正码(ECC)。
3. 如权利要求 1 所述的装置，其中所述高速缓存实现所述多个字的每个字的多位错误校正码(ECC)。
4. 如权利要求 1 所述的装置，其中所述多个修复路中的每一修复路包含一个修复位，并且所述多个修复路的所述一个修复位直接映射到所述多个列中的一个列。
5. 如权利要求 1 所述的装置，其中所述多个修复路中的每一修复路包含多个修复位，并且使用替换算法来选择第一路中的修复位来修复第一位。
6. 如权利要求 1 所述的装置，其中利用所述多个修复路中的与第一列关联的第一路中的修复位来修复第一位包括：
确定高速缓存写入是否引用所述多个字中的包含第一位的高速缓存字；
将待通过所述高速缓存写入而写入到第一位的逻辑信息写入到所述修复高速缓存中的修复位。
7. 如权利要求 1 所述的装置，其中利用所述多个修复路中的与第一列关联的第一路中的修复位来修复第一位包括：
确定高速缓存读取是否引用所述多个字中的包含第一位的高速缓存字；
用来自所述修复高速缓存内的修复位的逻辑信息来替换待从第一位读取的逻辑信息。
8. 如权利要求 7 所述的装置，其中用来自所述修复高速缓存内的修复位的逻辑信息来替换逻辑信息对于所述高速缓存是透明的。
9. 一种用于修复高速缓存阵列中单元的装置，包括：
在逻辑上组织成 M 行和 N 列的存储器阵列；
具有 N 个替换位的修复阵列，所述 N 个替换位中的每个替换位直接映射到所述 N 列中的一列；以及
模块，在对包含第一位的行访问时，所述模块找出所述 N 列的第一列中的待修复的第一位并用所述 N 个替换位中的直接映射到第一列的替换位来修复第一位，其中所述存储器阵列是高速缓存存储器，并且所述修复阵列是修复高速缓存，以及所述装置还包括用于修补所述 M 行的每行中的 X 个位的 X 位错误校正码(ECC)，其中所述模块基于用于将所述 M 行中的每行的错误数量减少至 X 个错误的算法来找出第一位。
10. 如权利要求 9 所述的装置，其中 M 是正整数，N 是 16 的整数倍。
11. 如权利要求 9 所述的装置，其中所述模块基于用于将所述 M 行中的每行的错误数量减少至 X 个错误的算法来动态地重新找出第一位。
12. 一种用于修复高速缓存阵列中单元的系统，包括：

微处理器,所述微处理器包括 :

具有多个行的高速缓存,其中所述高速缓存在逻辑上可见呈多个列;

修复模块,用于 :

将修复位与所述多个列中的每一列关联,

确定所述多个列的第一列中的待修复的第一位,以及

在对包含第一位的行访问时,用与第一列关联的修复位来修复第一位;

包括耦合到所述微处理器的多个元件的系统存储器,其中所述高速缓存用于存储所述多个元件的内容的本地副本,

其中确定所述多个列的第一列中的待修复的第一位包括 :

跟踪第一位的故障率;

确定在包含第一位的行中的错误数量;

基于第一位的故障率和包含第一位的行中的错误数量来确定待修复的第一位。

13. 如权利要求 12 所述的系统,其中所述微处理器能够进行并行乱序的整数和浮点执行。

14. 如权利要求 12 所述的系统,其中所述高速缓存具有从由以下高速缓存组成的组中选出的组织:直接映射高速缓存、组关联高速缓存和全关联高速缓存。

15. 如权利要求 12 所述的系统,其中所述修复逻辑包括固件,并且通过将修复位直接映射到所述多个列中的每一列来将修复位与所述多个列中的每一列关联。

16. 如权利要求 12 所述的系统,其中在对包含第一位的行访问时用与第一列关联的修复位来修复第一位包括 :

如果对包含第一位的行的访问是对包含第一位的行的写入,则将待存储在第一位中的值存储到与第一列关联的修复位;

如果对包含第一位的行的访问是从包含第一位的行读取,则用存储在与第一列关联的修复位中的值来替换待从第一位写出的值。

17. 如权利要求 12 所述的系统,其中所述系统存储器是从由随机存取存储器(RAM)、双倍数据速率(DDR)存储器设备以及静态 RAM(SRAM)存储器设备组成的组中选出的存储器设备。

18. 一种用于修复高速缓存阵列中单元的方法,包括 :

接收访问高速缓存的请求,所述请求引用第一高速缓存行;

确定第一高速缓存行是否包含待修复的第一高速缓存单元;以及

如果第一高速缓存行包含待修复的第一高速缓存单元,则在完成访问所述高速缓存的请求时,用与第一高速缓存单元关联的修复高速缓存单元来修复第一高速缓存单元,

其中待修复的第一高速缓存单元基于第一高速缓存单元的故障率和包含第一高速缓存单元的高速缓存行中的错误数量来确定。

19. 如权利要求 18 所述的方法,其中访问所述高速缓存的请求是对所述高速缓存的写入,并且修复第一高速缓存单元包括:在完成对所述高速缓存的写入时,将待写入到第一高速缓存单元的逻辑值透明地写入到与第一高速缓存单元关联的修复高速缓存单元。

20. 如权利要求 18 所述的方法,其中访问所述高速缓存的请求是从所述高速缓存读取,并且修复第一高速缓存单元包括:在完成从所述高速缓存读取时,从与第一高速缓存单

元关联的修复高速缓存单元透明地读取待从第一高速缓存单元读取的逻辑值。

21. 如权利要求 18 所述的方法, 其中确定第一高速缓存行是否包含待修复的第一高速缓存单元包括 :

如果查找模块没有将第一高速缓存单元与第一高速缓存行关联, 则确定第一高速缓存行不包含待修复的第一高速缓存单元;

如果所述查找模块将第一高速缓存单元与第一高速缓存行关联, 则确定第一高速缓存行包含待修复的第一高速缓存单元。

22. 如权利要求 18 所述的方法, 其中将所述修复高速缓存单元与第一高速缓存单元关联包括将所述修复高速缓存单元直接映射到所述高速缓存的包含第一高速缓存单元的列。

23. 如权利要求 22 所述的方法, 其中第一高速缓存单元是第一位, 并且修复高速缓存单元是修复位。

24. 如权利要求 23 所述的方法, 其中对第一高速缓存行的引用包括用于识别所述高速缓存的第一行的标签值。

25. 如权利要求 18 所述的方法, 其中访问所述高速缓存的请求是对所述高速缓存的写入, 并且修复第一高速缓存单元包括 : 在完成对所述高速缓存的写入时, 将待写入到第一高速缓存单元的逻辑值透明地写入到与第一高速缓存单元关联的修复高速缓存单元。

26. 一种用于修复高速缓存阵列中单元的方法, 包括 :

确定待用修复高速缓存来修复的高速缓存中的多个位的优化配置, 所述修复高速缓存包含直接映射到所述高速缓存的每个逻辑垂直带的修复位; 以及

在对所述高速缓存的访问引用所述高速缓存的包含第一位的字时, 修复经确定待用直接映射到包含第一位的逻辑垂直带的修复位来修复的所述高速缓存中的所述多个位的第一位,

其中确定待修复的高速缓存中的多个位的优化配置基于将所述高速缓存的每个字中的第一错误数量减少至所述高速缓存的每个字中的可校正的错误数量。

27. 如权利要求 26 所述的方法, 其中所述高速缓存包括 2 位错误校正码 (ECC), 并且所述高速缓存的每个字中的可校正的错误数量是 2。

28. 如权利要求 26 所述的方法, 其中确定待修复的高速缓存中的多个位的优化配置基于所述多个位的故障率。

29. 如权利要求 26 所述的方法, 其中修复所述多个位中的第一位包括 :

如果引用包含第一位的字的对所述高速缓存的访问是写操作, 则将待存储在第一位中的值写入到直接映射到包含第一位的逻辑垂直带的修复位;

如果引用包含第一位的所述高速缓存的字的对所述高速缓存的访问是读操作, 则读取存储在直接映射到包含第一位的逻辑垂直带的修复位中的值。

30. 如权利要求 26 所述的方法, 其中动态地重新评估待修复的所述多个位的优化配置的确定。

用于修复高速缓存阵列中单元的装置、系统和方法

技术领域

[0001] 本发明涉及高速缓存存储器领域,具体来说,涉及对高速缓存阵列中的单元的修复。

背景技术

[0002] 通过允许各种大小和关联性选择来提供高速缓存的设计灵活性,同时保持高速缓存找出 / 存储所请求的元素的速度,这对于利用高速缓存的体系结构来说是非常有利的。但是,当涉及半导体技术时,功率节省始终是一个普遍问题,它通常会导致限制性能或牺牲可靠性。

[0003] 在涉及高速缓存阵列时,一种典型的节省功率的方法包括在比设备的其余部分(如处理器、芯片集或其它集成电路)的电压低的电压下运行高速缓存阵列。然而,电压越低,会使高速缓存阵列内的单元越易出现软错误,即出现导致错误的位反转。此外,高速缓存内的单元通常会因为设计、制造、或在制造事件之后永久地损坏,这通常称为硬错误。

[0004] 传统上,为了校正硬错误,用备用行来替换包括硬错误的高速缓存行。与之相比,软错误通常可以通过高速缓存阵列所用的错误校正码(ECC)来校正。ECC 通常是指用于检测并且可以潜在地找出以及修补错误的逻辑。作为一个实例,许多高速缓存阵列利用 1 位 ECC 来校正每个字或高速缓存行的单个位错误。

[0005] 参照图 1,示出现有技术的高速缓存 100。通常,高速缓存存储器就是一个存储器阵列;但是,它也可以在物理上组织成或在逻辑上视为是具有多个行 / 字,如行 106-113。此外,高速缓存的每一行中的每个位或每组位都可以在逻辑上视为是形成一列,如列 115、116、117 和 118。假定高速缓存 100 包括 1 位 ECC,则可以检测和修补每行中的单个错误,如高速缓存行 106 中的位错误 130、高速缓存行 109 中的位错误 131 和高速缓存行 111 中的位错误 132。

[0006] 但是,当高速缓存 100 的电压下降到低于临界电压时,某些位会开始发生故障。因此,为了确保高速缓存 100 中的可靠性,供应给高速缓存 100 的电压只有在诸如高速缓存行 113 的行开始具有多个位错误(如位错误 120 和 125)之前才可能降低到临界电压。因此,为了确保可靠性,不会再进一步降低电压;但是,这是以牺牲功率节省为代价的。

附图说明

- [0007] 通过附图中的各图举例而不是限制性地示出本发明。
- [0008] 图 1 示出现有技术的高速缓存阵列的实施例。
- [0009] 图 2 示出包括高速缓存、修复高速缓存和修复模块的处理器的实施例。
- [0010] 图 3 示出高速缓存阵列的物理组织和从逻辑角度看的组织的实施例。
- [0011] 图 4a 示出用于修复高速缓存中的多个错误的修复高速缓存的实施例。
- [0012] 图 4b 示出查找表的实施例。
- [0013] 图 5 示出用于修复具有多个组和路的高速缓存阵列中的错误的修复高速缓存的

实施例。

[0014] 图 6a 示出用于修复高速缓存单元的流程图的实施例。

[0015] 图 6b 示出如图 6a 所示的用于修复高速缓存单元的流程图的特定实施例。

[0016] 图 7 示出用于基于高速缓存中待修复的多个位的优化配置来修复高速缓存中的位的流程图的实施例。

具体实施方式

[0017] 在以下描述中,阐述了众多特定细节,如特定高速缓存组织、高速缓存设置、高速缓存大小等的实例,以便充分理解本发明。但是,本领域的技术人员将明白,无需采用这些特定细节也可以实现本发明。在其它情况下,没有详细示出熟知的元件或方法,如错误校正、高速缓存设计和高速缓存接口等,以免不必要地使本发明晦涩难懂。

[0018] 本文描述的方法和设备是用于修复高速缓存存储器 / 阵列的。在一个实施例中,在诸如微处理器、嵌入式处理器、单元处理器或其它整数或浮点执行设备的处理设备上实现高速缓存。但是,用于修复高速缓存的方法和装置不限于此,它们可以在任何集成电路设备上实现,或与任何集成电路装置相关联地实现。

处理器的实施例

[0020] 图 2 示出处理器 200,它包括处理器逻辑 220、高速缓存 205、修复高速缓存 210 和修复模块 215。处理器逻辑 220 可以包括用于执行指令或对数据进行操作的任何逻辑。在一个实施例中,处理器逻辑 220 包括用于乱序并行执行整数和浮点数据的逻辑。作为另一个说明性实例,处理器逻辑 220 包括用于与外部设备对接的接口逻辑、用于提取和解码数据或指令的前端逻辑、用于辅助指令的乱序执行的乱序逻辑和用于执行指令和 / 或对数据进行操作的至少一个执行单元。

[0021] 高速缓存 205 包括与处理器 200 关联的任何高速缓存存储器阵列,包括诸如一级高速缓存的低级高速缓存、诸如跟踪高速缓存或二级高速缓存的中级高速缓存、或诸如三级高速缓存的较高级高速缓存。尽管图中将高速缓存 205、修复高速缓存 210 和修复模块 215 示为是位于处理器 200 中,但每一个也可以集成在另一个内,同时也可以位于处理器 200 之外。作为一个实例,在修复高速缓存 210 中实现修复模块 215。诸如模块 215 的任何模块可以在硬件、软件、固件或其任意组合中实现。通常,在不同的实施例中,模块边界会有所变化,并且可以共同或独立地实现各个功能。修复高速缓存 210 用于根据修复模块 215 的判断修复高速缓存 205 中的高速缓存单元。下文将参照图 4a、4b 和 5 更详细地论述修复高速缓存 210 和模块 215。

高速缓存的物理和逻辑组织

[0023] 转到图 3,示出高速缓存阵列 302 的组织。通常,高速缓存在物理上组织成阵列。图中将高速缓存阵列 302 示为是一维阵列;但是,高速缓存可以是多维的,并且在物理上可以按任何方式组织。然而,在逻辑上,高速缓存可以视为是具有多个组、路、行 / 字、列、以及单元 / 位。此外,高速缓存单元是指分解成高速缓存字 / 行的任何粒度。例如,高速缓存单元包括单个位,或者在备选实施例中,是指位的分组。高速缓存 302 包括用作数据或指令的中间存储设备的任何类型的存储器,如随机存取存储器(RAM)设备或静态 RAM(SRAM)设备。

[0024] 传统上,使用三种类型的高速缓存组织:全关联、k-路组关联以及直接映射高速

缓存组织。在全关联高速缓存中，在任何高速缓存项中存储主存储器的任一行，这使得高速缓存比较变得复杂。在组关联高速缓存中，通常将高速缓存在逻辑上分解成多路，其中来自主存储器的单元存储在每路内的特定的预定单元中，由此简化高速缓存查找过程。可用于存储主存储单元的每条路内的单元通常在逻辑上集合在一起并称为组。直接映射高速缓存实际上是 1 路组关联高速缓存，它具有用于潜在地存储具有共同寻址属性的多个存储单元的一个单元。

[0025] 为了简化对高速缓存内的行和列的描述，假定阵列 302 可以在逻辑上视为是具有高速缓存行 / 字 305、310 和 315 的单个路。每个高速缓存行 305、310 和 315 具有多个位，如位 306、311 和 316。物理视图 300 示出在物理上组织成一维阵列的高速缓存行。逻辑视图 350 示出行和列配置，其中高速缓存行 305、310 和 315 是行，并且这些高速缓存行的相同位置中的每个位形成一列。作为实例，列 320 包括来自高速缓存行 305、310 和 315 的第一位 306、311 和 316。

[0026] 作为另一个实例，阵列 302 在逻辑上组织成 M 行和 N 列。通常，高速缓存具有任意整数数量的 M 行，并在每行中具有 16 的倍数的 N 个位。但是，高速缓存不限于此，因为 M 和 N 可以是任意正整数。如上所述，高速缓存可以按照与如图 3 所示的方式不同的方式在物理上和逻辑上组织。实际上，图 5 中示出并参照图 5 论述组关联高速缓存的一个实例。

[0027] 修复高速缓存的实施例

[0028] 接下来转到图 4a，示出高速缓存 405 和修复高速缓存 430。高速缓存 405 具有多个字 406-413，这些字又称为行、行或元素。如上所述，参照图 3，在它们的相应字内具有相同位置或偏移的字 406-413 的每个位在逻辑上可以视为是列或在逻辑上组织成列。例如，列 415 包括行 406-413 中的每一行内的第一位，而列 416 包括行 406-413 的第二位。

[0029] 如图所示，高速缓存 405 在例如单元 421-423 和 426-429 处包括一些错误。这些错误可以是硬错误、软错误、或硬错误和软错误的组合。硬错误是其中位或高速缓存单元因某个制造、硬件或设计缺陷而发生故障的错误。软错误也可包括因制造缺陷引起的可预测的错误和不太可能两次影响相同单元的随机错误，如由于低电压供应引起的位的无限反转。高速缓存 405 还描绘修复后的单元 426、427、428 和 429。修复后的单元 426-429 将参照修复高速缓存 430 进行更详细地论述。

[0030] 尽管图中将修复高速缓存 430 示为与高速缓存 405 分离并耦合到高速缓存 405，但修复高速缓存 430 不限于此。实际上，在一个实施例中，是在高速缓存 450 中实现修复高速缓存 430 的。修复高速缓存 430 包括多个“路”。与分解成组和路的高速缓存组织方案类似，修复高速缓存 430 中的路包括映射到高速缓存 405 中的列的至少一个位或单元。作为一个简化的说明性实例，假定每条高速缓存行为 128 位宽，则修复高速缓存 430 包括 128 路。还值得注意的是，修复高速缓存 430 不需要具有与高速缓存 405 中的位数量相同的路。作为一个实例，即使与修复高速缓存关联的高速缓存具有 128 位的行宽，修复高速缓存仍可包括 64 路。在该实例中，修复高速缓存的每一路与高速缓存的两列关联或映射到高速缓存的两列。在图 4a 的实例中，修复高速缓存 430 中的每一路具有 1 个位；但是，修复高速缓存 430 中的每一路也可以包括多个位，稍后如图 5 所示。

[0031] 将修复路 / 位与高速缓存的列关联

[0032] 修复高速缓存 430 内的路与高速缓存 405 中的从逻辑上可见的列 / 垂直带关联。

为简单起见,图 4a 示出修复高速缓存 430 中只具有一个位的每一路,如包括位 438 的第一路和包括位 436 的第二路。位 438 与高速缓存 405 的列 415 关联。在一个实施例中,列 415 直接映射到位 438。因此,列 415 内的错误映射到位 438 并通过位 438 来修复。

[0033] 在使用不同数量的路或每一路中使用多个位的实施例中,可以使用其它熟知的映射技术和替换算法来将路与列关联。在一个实例中,使用上次使用或时间替换算法来选择修复位用于修复高速缓存中的错误。下文将参照图 5 论述其它实例。

[0034] 将修复位或修复路与高速缓存的在逻辑上可见的列关联可以在任何模块中实现。在一个实施例中,通过硬件来将修复位 438 与列 415 关联,其中修复位 438 在物理上与引用列 415 的地址或地址的一部分关联。在另一个实施例中,通过硬件和软件的组合来将修复位 438 与列 415 关联。作为一个实例,使用引用列 415 中的单元的物理或虚拟地址的特定位来将位 438 与列 415 关联。因此,当存在引用那些特定位的访问时,使用正确的修复位 438 来修复该单元。作为另一个实例,当使用修复位 438 来修复列 415 中的单元(如修复后的单元 428)时,查找表将修复位 438 与单元列 415 和单元 428 关联。

[0035] 确定待修复的单元

[0036] 诸如高速缓存 405 的高速缓存内待修复的单元或位的确定或找出 (target) 可以基于任何数量的因素或考虑因素。通常,在确定待修复的单元或位时,首先确定某个位或单元处是否存在错误。上文论述的高速缓存 405 示出具有诸如位 421、422 和 423 以及位 426–429 的错误的多个单元。如上所述,错误 421–423 和 426–429 包括硬错误、软错误或硬错误和软错误的组合。硬错误相对易于检测,因为某个硬件或制造缺陷往往会始终如一地包含一个错误。但是,软错误常常更难检测,因为这一次位或单元有效,而到下一次访问时,就会存在错误。当供应给高速缓存的电压降低到高速缓存单元的临界电压附近时尤其如此,在临界电压附近,某些单元开始发生故障。

[0037] 因此,最初确定单元或位是否具有错误也可以基于多个因素。在第一实施例中,将产生反转位或错误值的任何单元确定为是坏位或出错位。在另一个实施例中,使用不那么刚性的策略来检测和确定为故障位。作为一个实例,使用预测故障分析来确定位是否出错。这里,跟踪位或单元发生故障的次数,即位故障率。在发生预定次数的故障或足够高的故障率之后,确定该位为待修复的坏位或故障位。这潜在地避免在过少或过多故障时将位标记为故障位或错误位。可以使用其它熟知的错误检测方法来检测和 / 或识别坏高速缓存单元,这里不再详细描述这些熟知的错误检测方法,以免使本发明晦涩难懂。

[0038] 当将诸如位 421–423 和 426–429 的多个位视为是出错或故障位时,则找出待修复的一个位或多个位。下文将参照图 5 更详细地论述高速缓存中位的修复。在一个实施例中,确定待修复的位的优化配置。如上所述,利用多种因素来确定待修复的位的优化配置,这些因素如位在字和列中的位置、位的故障率、位的故障率的预测分析、每个字中的错误数量、每个字中可校正的错误数量、优化每个字中的错误校正以及其他熟知的用于校正或修复高速缓存单元的考虑因素。在确定待修复的位时,可以个别或相互结合地考虑这些考虑因素中的每个因素。

[0039] 作为特定的说明性实例,假定高速缓存 405 实现单个位错误校正码 (ECC)。因此,ECC 可以校正每个字或行中的 1 位。以前,在单个高速缓存行中具有两个坏位的部分(如高速缓存)会使这部分不能再用,这是因为单个位校正 ECC 无法恢复正确的数据。但是,利用修

复高速缓存 430, 可以降低电压, 并修复每个字中更多的错误。在该实例中, 高速缓存行 409 和 413 各具有三个位错误。因此, 为了将每个高速缓存行中的错误数量减少至可校正的量(在本实例中, 因为实现 1 位 ECC, 所以为 1 个位), 要修复高速缓存的行 409 和 413 中的两个位。注意, 在列 416 中, 用修复位 436 来修复位 426 而不是修复位 421。基于每个字中的错误数量和每个字中可校正的错误数量找出待修复的位 426。具体来说, 如果使用修复位 436 来修复位 421, 则字 409 将具有三个错误, 其中只能修复一个位, 即位 427。因此, 1 位 ECC 将不能校正两个错误, 并且就无法访问字 409。

[0040] 在备选方法中, 如果高速缓存 405 实现多位校正 ECC, 如 X 位错误校正, 则确定待修复的位的优化配置以将高速缓存 405 的每个字中的错误数量减少至 X 个错误。具体来说, 如果实现 2 位校正 ECC, 则优化每个字中的错误校正包括将每个字中的错误数量减少至 2 个或更少。然而, 确定待修复的位也可以基于前述考虑因素采用层级方式进行。在一个实施例中, 第一级考虑因素是将高速缓存的每个字中的错误数量减少至两个错误。在第二级, 在修复两个位之间存在同等选择的情形下, 选择基于其它次级因素。

[0041] 为了说明, 注意, 列 417 中的位 423 或位 429 是待通过修复位 439 来修复的候选位。在用修复位 439 来修复位 429 的所示实施例的备选实施例中, 修复位 439 改为修复位 423。为了对此变化做出补偿, 用修复位 440 来修复位 441 以使行 413 中只保留 1 位错误。因此, 为了在位 423 和 429 之间选择, 使用诸如故障率的其它次级考虑因素来确定要修复哪个位。为了进一步说明该实例, 假定位 429 发生 20 次(75% 的时间)故障, 而位 423 发生 5 次(20% 的时间)故障, 则基于故障次数或它们的故障率来选择其中任何一个位为待修复。注意, 不是每个错误或位都需要修复的, 因为可以通过 ECC 校正的错误的数量仍可留下。

[0042] 上述这些实施例和实例纯粹是示例性的, 这如以下事实证明, 即高速缓存 405 可以实现多位 ECC 或不需要实现任何级的 ECC, 并且可以通过任何已知方法将位确定为坏 / 故障位或基于任何数量的因素找出待修复的位。

[0043] 尽管图中描绘为位于修复高速缓存 430 中, 但用于确定高速缓存 405 内待修复的位的模块 450 也可以在高速缓存 405、修复高速缓存 430、包括高速缓存 405 和修复高速缓存 430 的处理器、或其任意组合中实现。模块 450 包括用于确定高速缓存 405 内待修复的位的任何硬件、软件、固件、代码、电路或其组合。诸如模块 450 的任何模块都可以在硬件、软件、固件或其任意组合中实现。通常, 在不同的实施例中, 模块边界会有所改变, 并且各个功能可以一起或者独立实现。固件通常是指用于执行某个功能的硬件和软件 / 微代码例行程序的组合。

[0044] 在模块 450 的一个实施例中, 使用包括 ECC 的高速缓存 405 中的逻辑来确定哪些位发生了故障, 而修复高速缓存 430 中的固件则跟踪这些位的故障次数和 / 或故障率并基于每个字中的错误数量和故障率找出待修复哪些位。在另一个实施例中, 仅仅与修复高速缓存 430 关联的固件只确定高速缓存 405 中的待修复的位。

[0045] 作为模块 450 的另一个简化的说明性实例, 固件跟踪高速缓存 405 中的位故障。固件例行程序基于上述因素中的任何个别因素或因素组合来根据算法确定待修复的位, 这些因素如每个字中的错误数量、每个字中可校正的错误数量、故障率和其它考虑因素。此外, 固件存储用于将以下元素的任意组合彼此关联的查找表: 待修复的高速缓存单元 / 位、包括待修复的高速缓存单元 / 位的高速缓存行 / 字、修复单元 / 位、或包括待修复的高速缓存

单元 / 位的高速缓存列。如下所论述,在访问高速缓存时,查找表潜在地帮助修复高速缓存单元。

[0046] 修复高速缓存单元

[0047] 如上文参照图 4a 所示,修复高速缓存或修复模块(如修复高速缓存 430 和修复模块 450)用与高速缓存 405 中包括待修复的位的列关联的修复位(如修复位 436–440)来修复诸如高速缓存 405 的高速缓存中的位。用修复高速缓存 430 中的位来修复高速缓存 405 中的位包括:用修复位来替换待修复的位;将待存储在待修复的位中的值存储在修复位中;提供修复位的内容以取代待修复的位;或以其它方式帮助修复高速缓存中待修复的位。

[0048] 在一个实施例中,使用修复位来修复高速缓存中的位包括:确定对高速缓存的请求是否引用高速缓存中包括待修复的位的字;以及用存储在修复位中的逻辑信息来替换待从高速缓存的该位读取的逻辑信息。作为一个实例,假定修复位 428 被确定为是待修复的位。则将位 428 的内容复制到位 438,或者在之前写入到行 413 时,将待存储在位 428 中的位存储在位 438 中。因此,如果请求对行 413 的读取,则在完成该读取时,通过读取值,即存储在修复位 438 中的逻辑值,来修复位 428。从不同的角度看,因为单元 428 被确定为是具有潜在不正确信息的坏单元,所以请求读取的设备从高速缓存 405 接收具有来自修复单元 438 的位而不是具有存储在单元 428 中的位的高速缓存行 413。

[0049] 在另一个实施例中,使用修复位来修复高速缓存中的位包括:确定对高速缓存的请求是否引用高速缓存中包括待修复的位的字;以及将待写入到高速缓存的该位内的逻辑信息写入/存储到修复位。作为一个实例,假定修复位 426 被确定为是待修复的位。因此,如果存在对字 409 的写入,则通过将待写入到位 426 的值写入到修复位 436 来修复位 426。从不同的角度看,设备写入是写入到高速缓存 405 中的字 409,并且待写入到单元 426 的位是写入到修复单元 436。

[0050] 由上文可见,从高速缓存 405 的角度看,修复高速缓存 430 可以透明地工作,它也可以作为中间层不透明地工作。

[0051] 在一个实施例中,修复高速缓存 430 和关联的逻辑用作访问高速缓存 405 的设备和高速缓存 405 之间的中间层。在该实施例中,修复高速缓存 430 将接收诸如读取和写入的所有请求,并在读取和写入操作期间修复高速缓存单元。例如,如果设备请求从高速缓存 405 的行 413 中读取,则修复高速缓存 430 将截获该请求,并在完成该请求时,通过用修复位 438 替换位 428 并用位 439 替换位 429 来向所请求的设备提供高速缓存行 413 的内容。

[0052] 但是,用作中间层会潜在地减缓某些高速缓存访问时间。例如,在访问高速缓存行 406 时,不需要通过修复高速缓存 430 来修复任何位,在此情况下,修复高速缓存没有理由截获和转发该请求。

[0053] 因此,在另一个实施例中,修复高速缓存 430 透明地修复高速缓存 405 中的位。透明工作的修复高速缓存 430 通过与以上实例对比来示出。在访问高速缓存行 413 时,修复高速缓存 430 查看请求所引用的单元,而不是截获和转发该请求。如果请求是读取并引用高速缓存行 413,则在完成该请求时,从高速缓存 405 发送出高速缓存行 413,并且修复高速缓存 430 用修复位 438 来修复高速缓存单元 428 并用修复位 439 来修复高速缓存单元 439。由此可见,修复高速缓存 430 的操作基本上对高速缓存 405 透明,因为高速缓存 405 在不受修复高速缓存替换位 428 和 429 的影响下接收请求并完成该请求。此外,在写入到高速缓

存行 413 时,对于高速缓存 405,写入像它之前那样继续,因为所有值都写入到高速缓存 405 中。但是,同时也将高速缓存单元 428 和 429 的值写入到修复单元 438 和 439 中。

[0054] 如上所述,在一个实施例中,模块 450 可以跟踪将通过修复高速缓存 430 中的修复位来修复高速缓存 405 中的那个位。例如,在查找表中,如在如图 4b 所示的查找表 460 中,列 465 中所列的修复位与它们要修复的高速缓存 405 中的位关联,高速缓存 405 中的这些位如列 470 中的对应项所列。因此,当访问引用包括单元 428 和 429 的高速缓存行 413 时,通过检查查找表 460,便能够确定位 428 和 429 为待修复的位。此外,表 460 包括关于将通过位 438 和 439 来修复它们的信息。

[0055] 但是,模块 450 不限于如图 4b 所示的查找表 460 的特定实现。例如,除了列 470 中所列的位之外,或者取代列 470 中所列的位,查找表可以包括对高速缓存行的地址的引用。在另一个实施例中,该表包括修复高速缓存 430 中的所有修复位,并在将要修复位时设置标志位或写入对应的位单元。在备选方法中,如图 4b 所示,只有在修复位将要修复坏单元时才在表 460 中创建各项。

[0056] 作为另一个说明性实例,假定模块 450 将高速缓存 405 中的位 426 确定为是需要修复的位。还假定,因为 ECC 已经检测并且固件已经跟踪到位 426 具有高故障率,所以模块 450 确定要修复位 426。此外,包括位 426 的行 409 具有两个其它错误,并且只实现 1 位 ECC。因此,模块 450 将高速缓存行 409 的起始地址和大小、位 426 的地址以及修复位 436 的地址(因为修复位 436 与列 416 关联)存储在表中。因此,在访问高速缓存行 409 时,检查该表,并确定引用高速缓存行 409。在完成该请求时,模块 450 用修复位 436 来修复位 426,这是因为在表中它们与该高速缓存行关联。作为修正,多个高速缓存单元和修复位与多个高速缓存行关联。

[0057] 作为说明性实例,以上实例证明了修复模块 450 和查找表 460 的巨大的可能性。例如,通过硬件、软件、固件或其组合,表 460 可以将以下元素中的每个元素彼此关联以帮助修复高速缓存单元:待修复的高速缓存单元/位、包括待修复的高速缓存单元/位的高速缓存行/字、修复单元/位、或包括待修复的高速缓存单元/位的高速缓存列。还应注意,可以存储待关联的地址、地址引用、地址的若干部分或上述各项的其它表示。

[0058] 系统中的修复高速缓存的实施例

[0059] 转到图 5,示出具有耦合到存储器 561 的微处理器 505 的系统。尽管图中没有示出,但控制器集线器或其它集成电路/设备可以潜在地耦合在存储器 561 与微处理器 505 之间。微处理器 505 包括用于对数据进行操作和执行指令的任何逻辑或模块。如图所示,微处理器 505 包括高速缓存 510 和修复高速缓存 550。微处理器 505 还可包括用于执行指令、对数据进行操作或与外部设备进行通信的其它处理逻辑。

[0060] 在所示实例中,将高速缓存 510 示为是双路组关联高速缓存;但是,高速缓存 510 可以组织成直接映射高速缓存、关联高速缓存、全关联高速缓存、组关联高速缓存或路关联高速缓存。通常,主存储器阵列 561 在逻辑上分解成多页,如页 565-580。主存储器 561 包括任何存储器组或阵列,如随机存取存储器(RAM)、静态随机存取存储器(SRAM)、动态 RAM(DRAM)或其它主存储器设备。

[0061] 主存储地址 566 引用存储器 561 的页内单元,如页 565 内的单元 566。在地址 566 内,某些位对于页 565 内的每个单元来说是相同的,并且有些位指定页 565 内的单元 566 的

偏移,即页偏移值。地址 566 的段 567 (又称为标签值)通常用于将单元 566 映射到高速缓存 510 内的一个组,如组 540。实际上,页 565 或者是页 565-580 中的每一页中具有相同偏移的单元都可以通过段 567 映射到一个组。组 540 包括地址单元 566 所映射到的第一路 511 中的高速缓存行和第二路 512 中的高速缓存行。因此,在对主存储单元 566 请求时,高速缓存只需检查组 540,以便查看存储单元 566 的副本是否存在。高速缓存 510 中的每一路都包括 M 个行,其中每一行包括 N 个位。在图 5 中,M 和 N 分别示为是每路 8 行和每行 7 个位。

[0062] 图中将修复高速缓存 550 示为是耦合到高速缓存 510。修复高速缓存 550 包括 14 路,即高速缓存 510 的 M 行的 N 个位中的每一位对应 1 路,其中每一路与 N 个位中的至少一个位关联。如上所述,在另一个实施例中,利用 7 路来实现修复高速缓存,其中每一路具有两个位并映射到高速缓存 510 的两列。但是,如图所示,修复高速缓存 550 包括 14 路,每一路具有两个位并与高速缓存 510 的一列关联。例如,路 555 包括位 556 和位 557,并与列 541 关联。

[0063] 在一个实施例中,高速缓存 510 实现 ECC。作为说明性实例,高速缓存 510 实现具有在高速缓存 510 的每行中校正两个位的能力的 2 位 ECC。行 521 包括三个故障位,即位 545-547,它们全都无法通过 2 位 ECC 来校正。因此,使用与列 541 关联的路 555 中的修复位 556 来修复位 545。此外,行 522 包括 4 个坏位。因此,使用路 555 中的第二位(即修复位 557)来修复高速缓存单元 548。

[0064] 修复高速缓存 550 还包括修复模块 560。如上所述,修复模块 560 可以在修复高速缓存 550、高速缓存 510、微处理器 505 或其任意组合中或之间实现。修复模块 560 将每一路和修复位与高速缓存的列关联。在一个实施例中,路 555 直接映射到列 541。在该实例中,将路 555 直接映射到列 541 包括用路 555 中的修复位只修复列 545 中的位。一个映射实例包括:将识别列 541 中的位的地址的一部分与路 555 关联,这可以通过比较逻辑、映射表、查找逻辑、或其它常见的关联技术来完成。

[0065] 修复模块 560 还确定各列中待修复的位。如上所述,修复模块 560 基于优化配置来选择高速缓存 510 中待修复的位。在确定优化配置和待修复的位时,可以考虑通过算法、软件、硬件或其它机制的任何数量的因素,如位故障率、每个高速缓存行中的错误数量、每个高速缓存行中可校正的错误数量、实现的 ECC 类型、可用的修复位的数量、修复列中的一个位而不修复其它高速缓存行或列上的另一个位的影响、以及其它考虑因素。在一个实施例中,修复逻辑 560 基于上述因素动态地分析和重新确定待校正的位。

[0066] 此外,修复模块 560 修复经确定要修复的位。如上所述,在对包括坏位的高速缓存行进行读取时,从修复位读取与坏位所在的列关联的修复位,并在完成读取请求时用该修复位来替换坏位。作为一个实例,假定处理器 505 做出引用映射到组 540 的主存储地址 566 的读取请求。高速缓存 510 在行 521 中包括主存储单元 566 的副本。修复模块 560 确定要修复位 545。作为一个说明性实例,假定,修复逻辑 560 之前已经确定位 545 是故障位,并在表中创建了一个项以形成将用修复位 557 来修复坏位 545 的关联。然后,从高速缓存 510 中读出高速缓存行 521,并用存储在修复位 557 中的值来替换来自坏位 545 的值。最后,因为用来自修复位 557 的有效信息替换了坏位 545,所以微处理器 505 从高速缓存 510 接收到主存储单元 566 的有效副本。类似地,在写入到高速缓存 510 时,将待写入到坏单元 545 的

值写入到修复位 557。

[0067] **修复高速缓存单元的实施例**

[0068] 接下来参照图 6a, 示出修复高速缓存单元的流程图的实施例。在方框 605, 接收访问高速缓存的请求, 其中该请求引用第一高速缓存行。如上所述, 请求可以用多种方式引用高速缓存行, 这些方式包括与该高速缓存行关联的标签值、与该高速缓存行关联的物理地址、与该高速缓存行关联的虚拟地址、复制到该高速缓存行的物理 / 虚拟主存储地址、或任何其它常见的引用高速缓存的字 / 行的方法。

[0069] 接着, 在方框 610, 确定该高速缓存行是否包含待修复的第一高速缓存单元。在一个实施例中, 使用查找模块来确定该高速缓存行是否包含待修复的第一高速缓存单元。当确定要修复第一高速缓存单元时, 创建或修改查找模块中的项以便将修复单元与高速缓存单元和 / 或包含该高速缓存单元的高速缓存行关联。因此, 在访问时, 检查查找模块, 以便查看第一高速缓存行是否包含待修复的第一单元。

[0070] 在另一个实施例中, 使用多层查找模块来首先确定该高速缓存行是否包含任何待修复的单元。这可以通过在查找模块中列出对包括待修复的单元的高速缓存行的引用来完成。如果没有匹配的高速缓存行, 则不需要做进一步的分析。如果列出 / 有匹配的高速缓存行, 则确定该高速缓存单元是否在第一高速缓存行内或哪些高速缓存单元要修复。这可以用多种方式来完成, 例如在查找模块中输入信息或使用地址比较来确定该高速缓存单元在高速缓存行的地址边界内。

[0071] 可以使用其它机制和模块来确定高速缓存行是否包括待修复的第一高速缓存单元。例如, 固件可以存储对具有待修复的单元的高速缓存行的引用。关于确定高速缓存行是否包含待修复的第一单元的更多论述, 请参照上文中的修复高速缓存单元的部分。

[0072] 在方框 615, 如果第一高速缓存行包含待修复的第一高速缓存单元, 则在完成访问高速缓存的请求时, 用与第一高速缓存单元关联的修复高速缓存单元来修复第一高速缓存单元。如上所述, 第一高速缓存单元可以通过包括查找表的模块来关联。在一个实施例中, 通过将修复高速缓存单元映射到包括第一高速缓存单元的逻辑上可见的列来将第一高速缓存单元与修复高速缓存单元关联。这里, 将修复高速缓存单元直接映射到高速缓存的列, 并在确定要修复第一高速缓存单元时, 使用直接映射到包括第一高速缓存单元的列的修复高速缓存单元。

[0073] 转到图 6b, 示出用修复高速缓存单元来修复第一高速缓存单元的流程图的特定实施例。在方框 616, 在完成对高速缓存的写入请求时, 将待写入到第一高速缓存单元的逻辑值透明地写入到与第一高速缓存单元关联的修复高速缓存单元。一个实例包括写入到高速缓存行, 其中该高速缓存行具有第一坏位。从高速缓存的角度看, 对高速缓存行的写入像正常一样发生。但是, 当如在方框 610 那样确定写入是写入到包括坏位的高速缓存行时, 则也将待写入到第一坏位的值写入到映射到包括第一坏位单元的高速缓存的逻辑列的修复位。

[0074] 在方框 617, 在完成从高速缓存读取的请求时, 从与第一高速缓存单元关联的修复高速缓存单元读取待从第一高速缓存单元读取的逻辑值。继续上文的实例, 在方框 610, 确定对高速缓存行的请求包含第一坏位单元。从高速缓存的角度看, 如完成正常请求一样, 读取高速缓存行, 并由高速缓存提供该高速缓存行。但是, 从修复位单元读取存储在与第一坏位关联的修复位中的值, 并用该值替换来自第一坏位单元的值。因此, 高速缓存正常工作,

而为坏位存储和读出的数据位于修复位中。从高速缓存的观点看的这种视角是指写入 / 存储和读取 / 替换的透明度，就像高速缓存正常工作一样。

[0075] 转到图 7，示出基于待修复的多个位的优化配置来修复高速缓存的流程图的实施例。在方框 705，确定待用修复高速缓存来修复的高速缓存中的多个位的优化配置。修复高速缓存包括直接映射到高速缓存的每个逻辑垂直带 (stripe) / 列的修复位。

[0076] 如上所述，高速缓存可以具有任何数量的确定为坏位的位。因此，要修复哪些位的选择可以基于任何数量的因素做出。在一个实施例中，为了选择优化配置，使用基于以下因素中的任何一个因素或任何因素组合来选择待修复的位的配置的算法：位故障率，每个高速缓存行中坏位的数量，每个高速缓存行中可校正的坏位数量，实现的 ECC 类型，可用的修复位的数量，以及修复垂直带中的一个位、而不修复其它高速缓存行或列内的另一个位的影响。

[0077] 在一个实施例中，动态地重新评估或动态地确定位的优化配置的确定。例如，假定基于它的故障率和在具有其它坏位的行内的位置而确定要修复第一位。但是，如果在不同的时间情形发生变化，则可以选择要修复位于与第一位相同的垂直带内的另一个位来取代修复第一位。这允许修复高速缓存具有最大的灵活性来确保在变化的情形下确定待修复的位的最佳配置。

[0078] 接着，在方框 710，在对高速缓存的访问引用包含第一位的高速缓存的字时，修复经确定待用直接映射到包含第一位的逻辑垂直带的修复位来修复的高速缓存内的多个位中的第一位。在一个实施例中，修复多个位中的第一位包括：如果引用包括第一位的字的对高速缓存的访问是写操作，则将待存储在第一位中的值写入到修复位。在另一个实例中，修复多个位中的第一位包括：如果引用包括第一位的高速缓存的字的对高速缓存的访问是读操作，则读取存储在修复位中的值。

[0079] 如上所示，可以使用修复高速缓存来增加在较低电压下运行的高速缓存的效率和准确度。以前，高速缓存无法将电源电压降低到更低的电压极限，因为这会使每个高速缓存行中存在太多的无法通过 ECC 校正的错误。但是，通过包含修复高速缓存，可以修复高速缓存的每个逻辑列中的错误，从而允许降低所供应的电压，而不会牺牲通过 ECC 校正的错误的数量。另外，不是高速缓存内的每个错误都需要修复。潜在地，通过与校正 ECC 配合使用，只需修复足以确保有效数据的错误。此外，被选择要校正 / 修复的位可以随着时间动态地变化，以选择修复位的更优化配置。因此，节省了功率，而不会牺牲高速缓存的准确度。

[0080] 在以上说明中，参照具体示例性实施例给出了详细描述。但是，很明显，在不偏离如随附权利要求所述的本发明的更广的精神和范围的前提下，可以对本发明做出各种修改和改变。因此，应将说明书和附图视为是说明性意义而不是限制性意义。此外，上文使用的实施例和其它示例性语言不一定是指相同的实施例或相同的实例，而是可以指完全不同的实施例，也可以潜在地指相同的实施例。

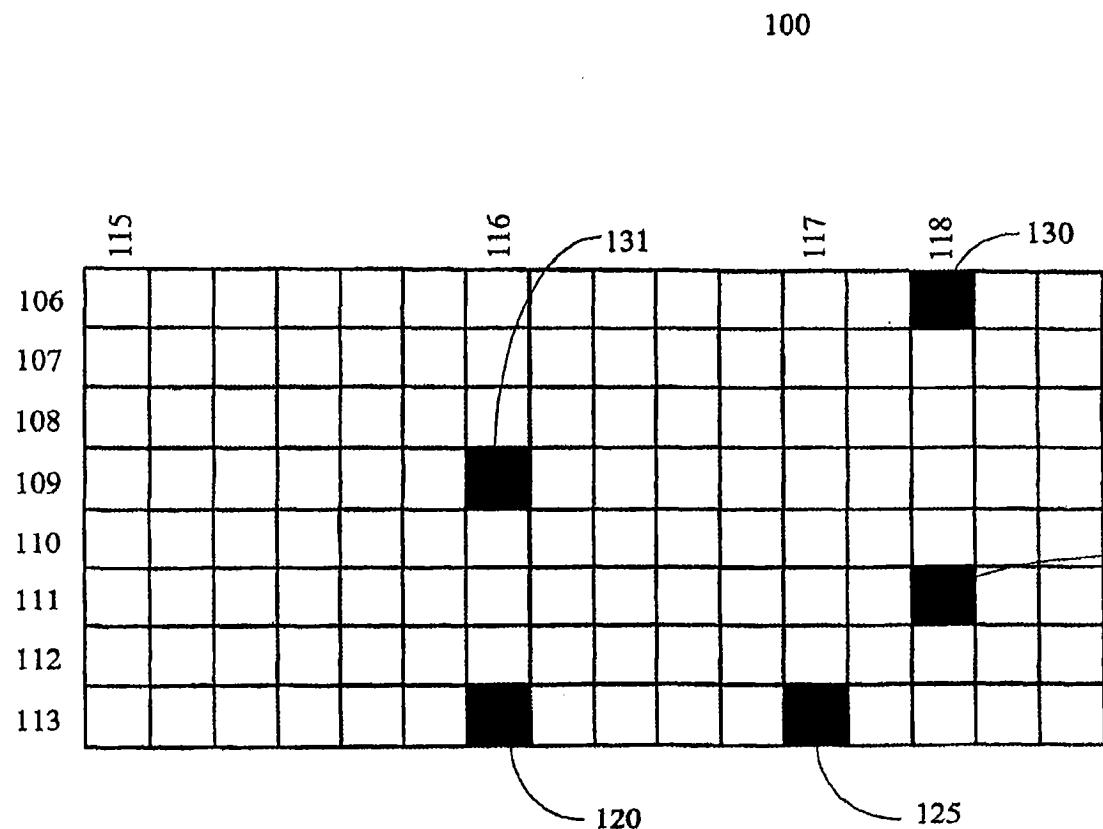


图 1

现有技术

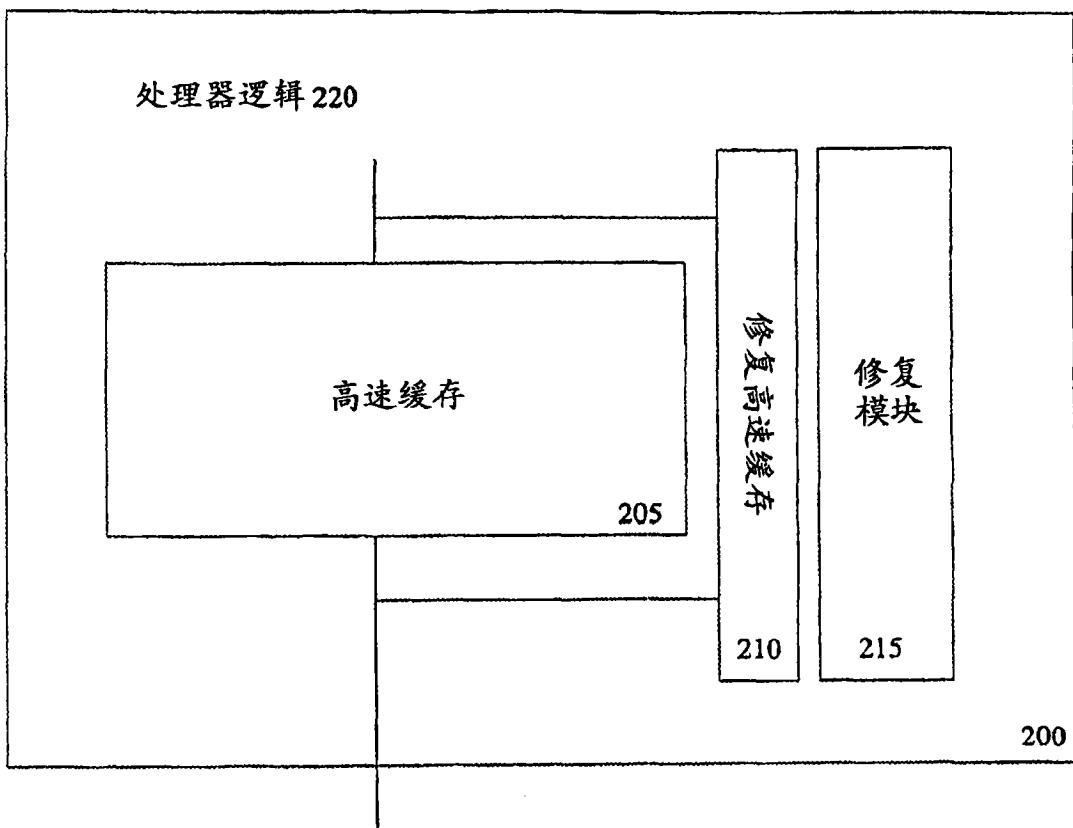


图 2

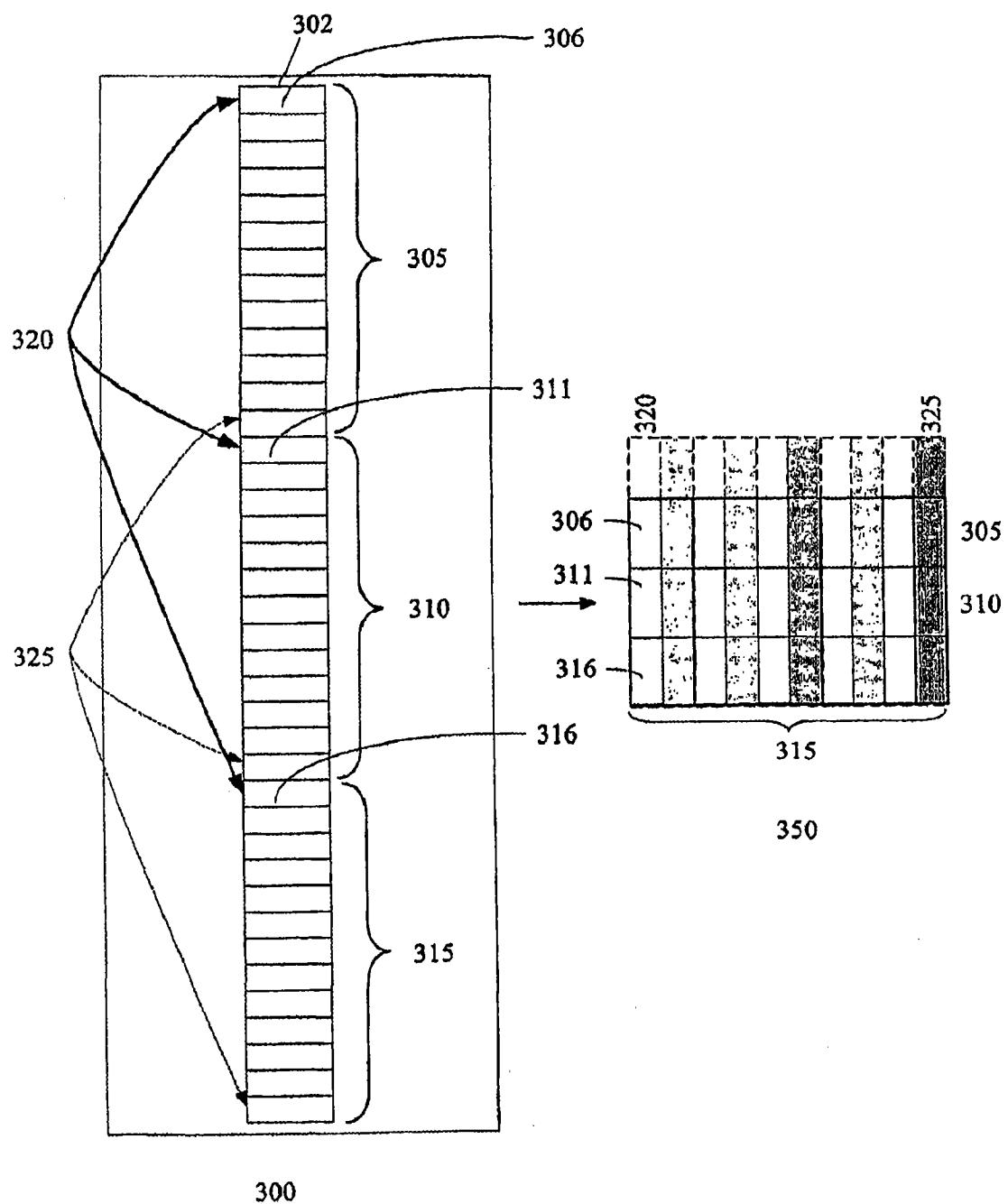


图 3

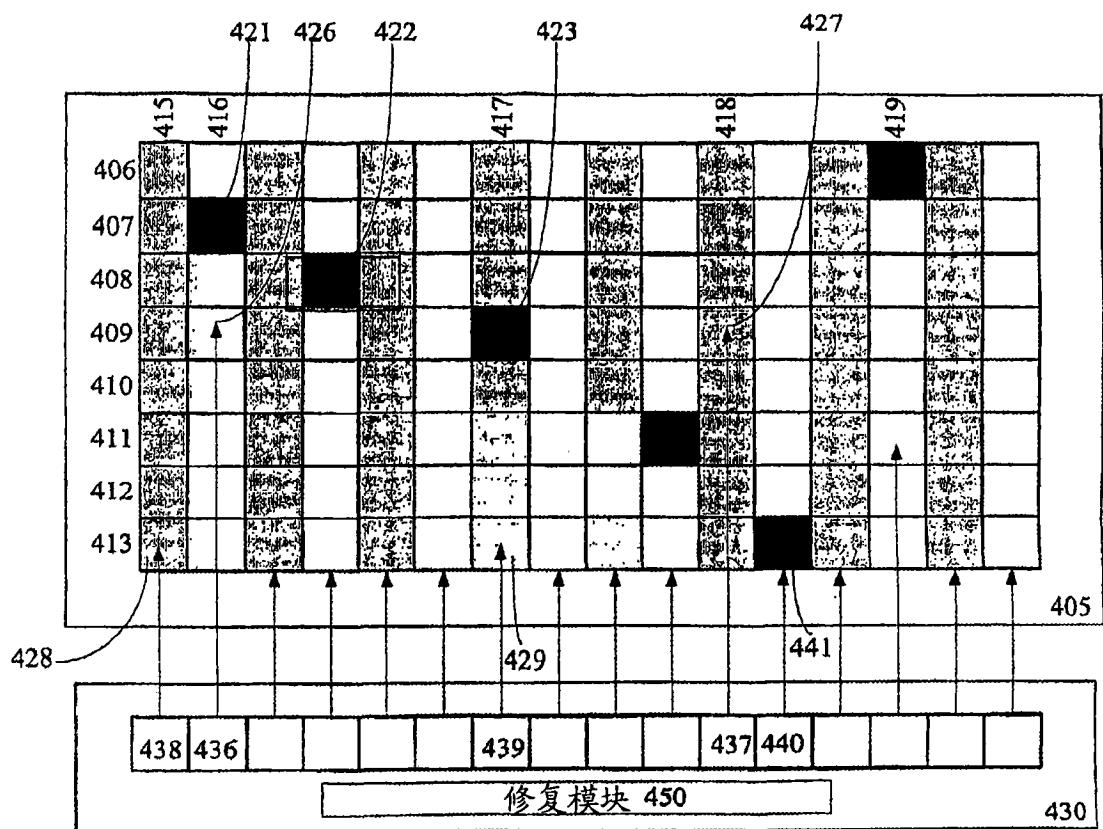


图 4a

460	465 修复位	470 坏位
	438	428
	436	426
	439	429
	437	427
	440	441

图 4b

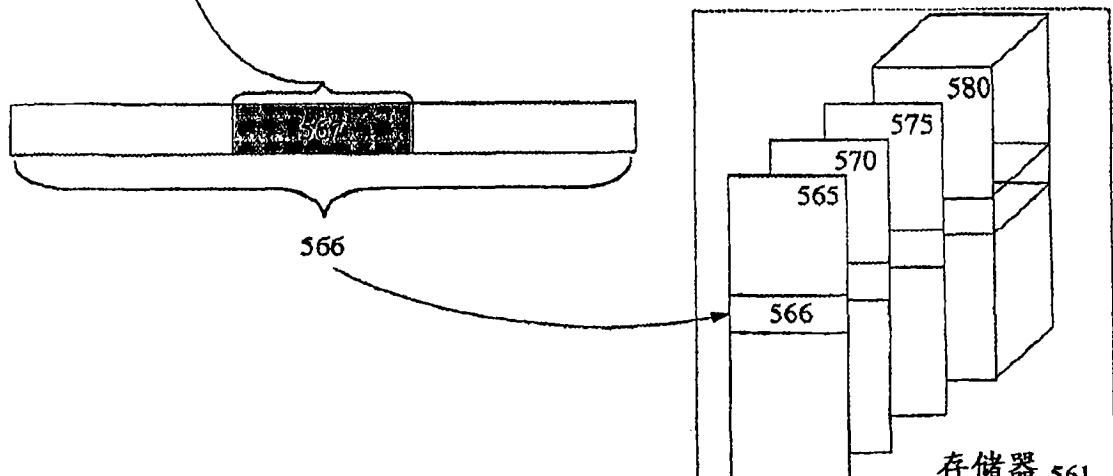
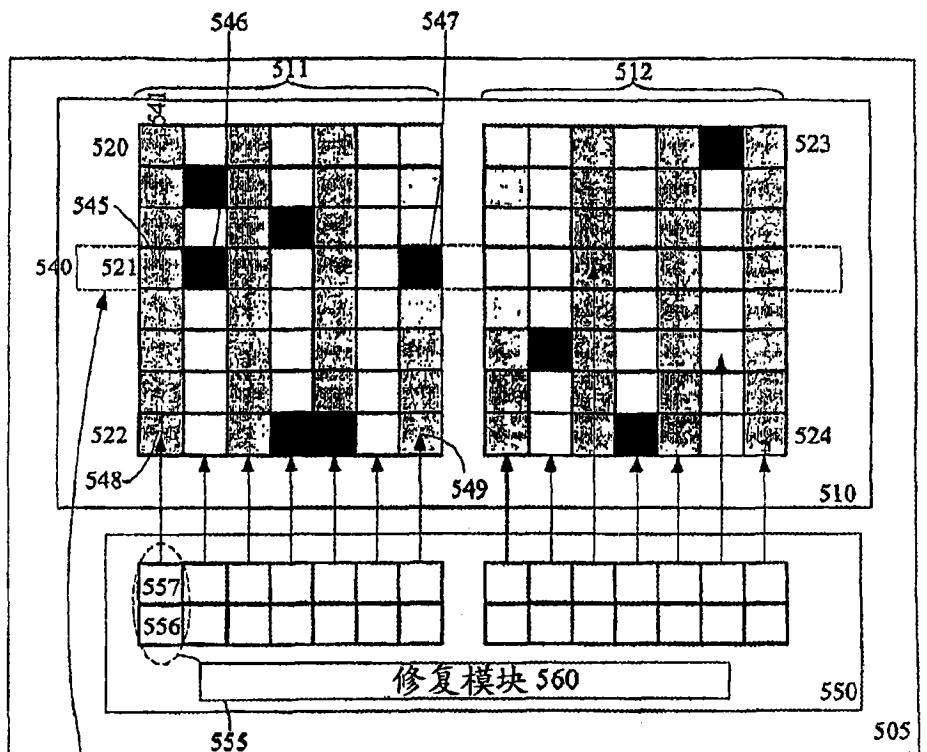


图 5

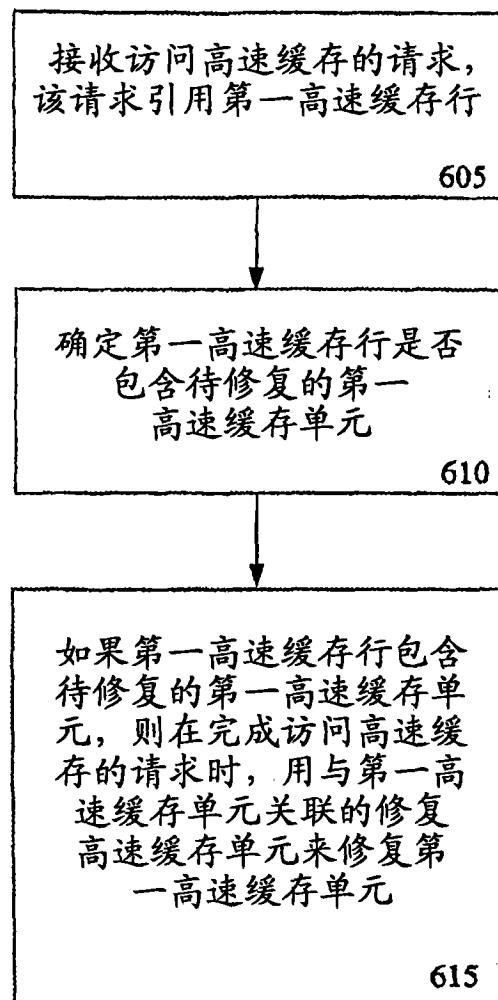


图 6a

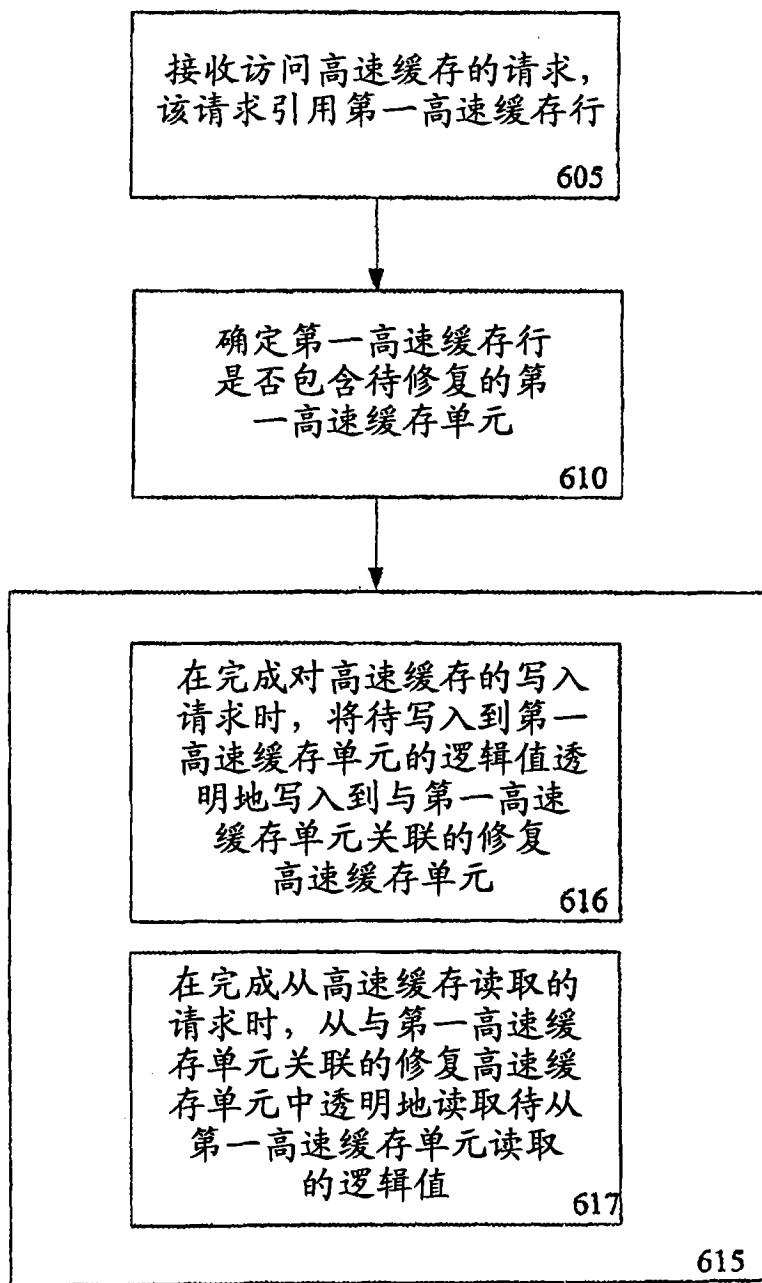


图 6b

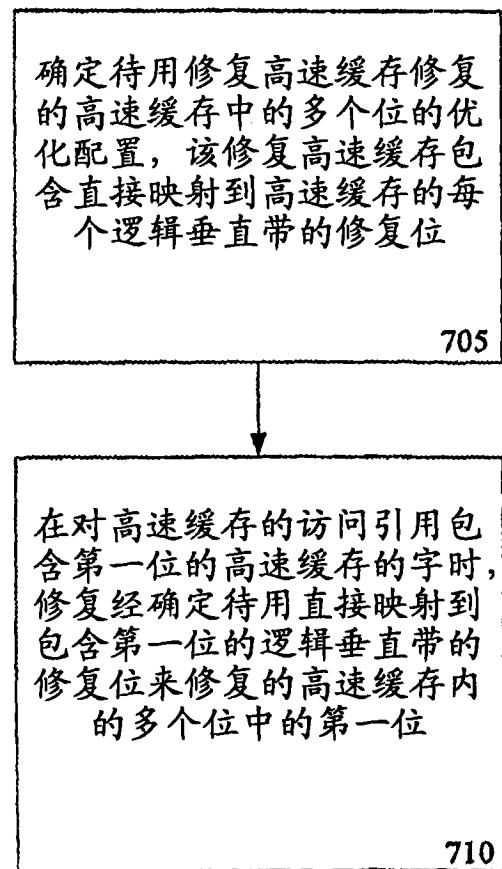


图 7