(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 102945201 B
(45)授权公告日 2016.02.17

(21)申请号 201210333854.2

(22)申请日 2012.09.11

(30)优先权数据
13/229736 2011.09.11 US

(73)专利权人 微软技术许可有限责任公司
地址 美国华盛顿州

(72)发明人 D. 摩斯 K. 梅拉 R. 纳加
S. 费尔马 S. 拉帕尔

(74)专利代理机构 中国专利代理(香港)有限公司 72001
代理人 董宁 汪扬

(51)Int.Cl.
G06F 11/34(2006.01)

(56)对比文件
US 2009/0106334 A1,2009.04.23,说明书第2段到第52段以及附图1-8.
US 2007/0174580 A1,2007.07.26,说明书第88段到第92段.
JP 特開 2001-51806 A,2001.02.23,全文.
US 2005/0034012 A1,2005.02.10,全文.
US 2005/0071593 A1,2005.03.31,全文.

审查员 李晓阳

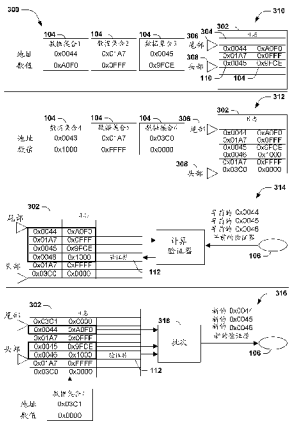
权利要求书3页 说明书21页 附图11页

(54)发明名称

已验证数据集合的非易失性介质日志记录

(57)摘要

本发明涉及已验证数据集合的非易失性介质日志记录。将数据集合存储在存储集合中(例如被写入到构成RAID阵列的硬盘驱动器的数据集合)可能会通过非顺序写入而降低所述存储集合的性能,特别在存储器件所迅速写入的数据集合之后有顺次跟随的数据集合的情况下尤其如此。此外,由于数据集合和验证器(例如校验和)的非原子写入和其间的故障(比如RAID写入空洞的发生),存储集合可能会表现出不一致。相反,可以在提交到存储集合之前首先把数据集合和验证器写入以存储在存储器件的非易失性介质上。这样的写入可以被顺序地写入到日志,而不管各个数据集合在存储集合中的位置如何;并且故障恢复可以简单地涉及重新提交日志中的各条一致记录以便校正对存储集合的不完整写入。



CN 102945201 B

1. 一种在由至少一个存储器件提供的存储集合中存储数据集合的方法，所述方法涉及具有处理器的计算机并且包括：

在处理器上执行使得计算机实施以下操作的指令：

在存储器件上生成日志，所述日志被配置成存储分别与一个验证器相关联的各个数据集合；

在接收到针对在存储集合中的某一位置处存储数据集合的请求之后：

为该数据集合计算一个验证器；以及

将所述验证器和所述数据集合存储在日志中；

从日志选择包括将要提交到存储集合的第一数据集合和第二数据集合的一个批次，使得将第一数据集合和第二数据集合一起写入到存储集合快于将第一数据集合和第二数据集合单独地写入到存储集合；

在从日志去除任何数据集合之前，对于所述批次的对应数据集合，将第一数据集合、第一数据集合的验证器、第二数据集合和第二数据集合的第二验证器存储在存储集合中；以及

仅仅在把所述批次的所有数据集合存储在存储集合中之后，从日志中去除所述批次的第一数据集合和第二数据集合。

2. 如权利要求 1 中所述的方法，其中选择批次还包括：在检测到从包括以下各项的提交事件集合当中选择的提交事件之后从日志选择将要存储在存储集合中的第一数据集合和第二数据集合：

涉及日志的容量的日志容量事件；

涉及存储在日志中的数据集合的持续时间的持续时间事件；

涉及针对把日志中的至少一个数据集合提交到存储集合的请求的提交请求事件；以及

涉及存储集合的至少一个存储器件的工作负荷的存储器件工作负荷事件。

3. 如权利要求 1 中所述的方法，选择所述数据集合的批次包括：

选择存储在日志中并且将被存储在存储集合中的第一位置处的第一数据集合以包括在所述批次中；以及

选择存储在日志中并且将被存储在存储集合中的靠近第一位置的第二位置处的第二数据集合以包括在所述批次中。

4. 如权利要求 1 中所述的方法：

对应的请求指定存储集合中的针对所述数据集合的位置；以及

对于数据集合计算验证器包括：

对于完全记录在日志中的数据集合，从该数据集合计算验证器；以及

对于没有完全记录在日志中的数据集合：

读取存储集合中的所述位置处的数据集合的原始版本；

读取该数据集合的原始版本的原始验证器；

从原始验证器中去除该数据集合的原始版本；以及

把该数据集合包括在原始验证器中。

5. 如权利要求 1 中所述的方法：

所述计算机包括易失性存储器；并且

所述指令被配置成：

在易失性存储器中生成存储在存储器件上的日志的易失性存储器表示；以及

在把数据集合存储在日志中之后，把该数据集合存储在日志的易失性存储器表示中。

6. 如权利要求 5 中所述的方法，所述指令被配置成在接收到针对读取数据集合的请求之后：

在确定所述数据集合被存储在易失性存储器内的日志的易失性存储器表示中之后，给出存储在易失性存储器表示中的数据集合；

在确定所述数据集合被存储在存储器件上的日志中之后，给出存储在日志中的数据集合；以及

在确定所述数据集合被存储在存储于存储器件上的存储集合中之后，给出存储在存储器件的存储集合中的数据集合。

7. 如权利要求 5 中所述的方法，所述指令被配置成在把一个批次存储到存储集合之后从日志的易失性存储器表示中去除所述批次。

8. 如权利要求 7 中所述的方法：

所述指令被配置成：

在把数据集合存储在日志中之后，把标记为不可去除的所述数据集合存储在日志的易失性存储器表示中；以及

在把数据集合存储在存储集合中之后，把存储在日志的易失性存储器表示中的所述数据集合标记为可去除；以及

把数据集合从存储在易失性存储器内的日志的易失性存储器表示中去除包括：从日志的易失性存储器表示中仅仅去除被标记为可去除的数据集合。

9. 如权利要求 5 中所述的方法：

日志的易失性存储器表示具有数据集合的容量；以及

在易失性存储器中生成日志的易失性存储器表示包括：在易失性存储器中保留容量以便存储：

日志的易失性存储器表示中的数据集合的容量；

构成日志的易失性存储器表示的容量的数据集合的验证器；以及

被配置成在把日志中所存储的数据集合存储在存储集合中的同时对将要存储在存储集合中的数据集合进行存储的缓冲器。

10. 一种把数据存储在包括至少一个存储器件的存储集合上的方法，所述方法涉及具有处理器的计算机并且包括：

向器件发送指令，所述指令在器件的处理器上被执行时：

在存储器件上生成包括记录、头部指针和尾部指针的序列的日志；

在接收到将要存储在存储集合中的某一位置处的数据集合之后：

前提日志的头部指针经过新的记录，并且

把所述数据集合存储在所述新的记录中；

从日志选择包括将要提交到存储集合的第一数据集合和第二数据集合的一个批次，使得第一数据集合和第二数据集合在日志的尾部指针附近；

在从日志去除第一数据集合和第二数据集合之前，把第一数据集合、第一数据集合的

验证器、第二数据集合和第二数据集合的验证器提交到存储集合；以及

在将第一数据集合和第二数据集合提交到存储集合之后，前提日志的尾部指针经过所述批次的第一数据集合和第二数据集合。

已验证数据集合的非易失性介质日志记录

背景技术

[0001] 在计算领域内，许多情形涉及在一个或更多非易失性存储器件(例如基于盘片的磁性和／或光学硬盘驱动器、固态存储器件以及非易失性存储器电路)上存储数据。数据存储的许多细节可以不同，比如字尺寸、寻址方法、将存储器件的存储空间划分成一个或更多分区以及将存储器件内的已分配空间暴露为计算环境内的一个或更多卷。

[0002] 在许多这样的存储情形中，可以利用各种技术来检测对于数据的非意定改变。例如，器件的读取或存储逻辑中的错误、缓冲区欠载或溢出、存储介质中的瑕疵或者外部干扰(比如宇宙射线)都可能不时导致在存储于存储介质上的数据中或者在从存储介质读取数据中的非有意的改变。因此，在许多这样的情形中，根据涉及对于对应的数据集合(例如不同的字、扇区、区段或其他数据集合)计算的验证器(例如奇偶校验位或校验和)的检错方案将数据存储在存储器件上。所述验证器可以被用来证实数据集合的内容已经被有效地存储到存储器件上和／或从存储器件中读取。作为一个这样的例子，在存储包括一个比特集合的数据集合的情境中，可以对各个比特应用异或(XOR)运算，从而得到可以存储并且与该数据集合相关联的一个奇偶校验位。当后来读取所述数据集合时，可以对其应用另一次异或运算，并且可以将结果与所述奇偶校验位进行比较。任一个比特的改变都会导致这些 XOR 计算的失配，从而表明数据已被错误地存储、改动或者从存储器件错误地读取。可以标识许多类型的验证器，其在某些特征方面可能有所不同(例如易于计算，标识出数据集合的*哪*一个比特被改变的能力，以及能够借以校正错误地读取的数据部分的纠错能力)。

[0003] 检错方案常常被用在廉价盘冗余阵列(RAID)阵列中，比如共用在一起以便获得各种聚集属性(比如改进吞吐量和自动数据镜像)的硬盘驱动器集合。作为一个这样的例子，RAID 4 阵列涉及两个或更多盘的集合，其中在所述阵列中包括一个盘，该盘不用来存储用户数据，而是用来存储被存储在其他盘上的数据的验证器。例如，对于一个涉及每个存储一兆兆字节数据的四个盘的 RAID 4 阵列，前三个盘的容量被共用从而形成用于用户数据的三兆兆字节存储空间，而第四个盘被包括在所述阵列中以保存用于存储在前三个盘上的各个数据集合的验证器(例如对于分别存储在其他三个盘上的每三个 64 比特字，第四个盘包括一个验证所述三个 64 比特字的完整性的 64 比特验证器)。RAID 阵列控制器包括被配置成实施针对所提供的驱动器集合的所选 RAID 级别的细节的电路(例如在接收到一个数据集合之后，将所述数据自动分配到三个用户数据盘上，计算所述数据集合的验证器，并且将验证器存储在第四个盘上)。所使用的 RAID 技术还可以实现附加的保护或特征；例如如果 RAID 4 阵列中的任何单个存储器件发生故障，则可以通过使用剩余的存储器件完全重建存储在故障器件上的数据。

发明内容

[0004] 提供本发明内容以简化形式介绍下面在具体实施方式中进一步描述的一组概念。本发明内容不意图标识出所要求保护的主题内容的关键因素或本质特征，也不意图被用来限制所要求保护的主题内容的范围。

[0005] 把数据写入到存储器件可能会存在几个低效源头和潜在问题。作为第一个例子，被写入到存储器件的数据集合可能涉及一个数据序列，比如写入到存储器件上的一个物理地址序列的数据。通过根据该序列写入数据集合(例如作为邻接数据块的顺序写入)，存储器件可以获得更快的查找时间、更高的吞吐量以及 / 或者由于查找时间和写入操作的减少而导致的降低功率消耗和物理损耗。然而由于各种情况，存储器件可能把所述数据序列作为两个或更多子序列写入，并且可能无法实现这些效率。作为第一个例子，针对写入数据集合的请求可能包括两个或更多针对写入所述序列的一部分的请求(例如针对写入地址 1000-1015 的第一请求和针对写入地址 1016 到 1031 的第二请求)，并且存储器件可以单独提交(commit)所述序列的每一个部分而不是一起提交整个序列。作为第二个例子，存储器件可能接收到几个写入请求，并且在写入序列的第一部分和序列的第二部分之间可能把一个不同的数据集合写入在一个不同位置处，从而在第一部分与第二部分之间导致两次附加的查找。这些和其他情况可以被标识为错失了提升性能效率、功率效率以及存储器件的使用寿命的机会。

[0006] 在存储基于验证器的存储集合时可能发生的第二个问题涉及存储数据集合及其验证器(或者反之亦然)之间的延迟。作为第一个例子，许多存储器件只支持每次对一个位置(例如处于硬盘驱动器的写入头下方的位置，或者由固态存储器件中的地址寄存器指定的位置)的写入，并且对于数据的顺序存储涉及在验证器之前写入数据集合，或者反之亦然。作为第二个例子，如果数据集合和验证器被存储在不同的存储器件上，则可能难以把第一存储器件存储数据集合的时刻与第二存储器件存储数据集合的验证器的时刻同步。在这些和其他例子中，存储数据集合和相应的验证器可能不是按照同步方式发生的，而是按照顺序方式发生的。然而在存储数据集合之后并且在存储验证器之前可能发生(一个或多个)存储器件的故障，比如掉电、硬件故障、软件崩溃、或者从阵列当中意外地去除存储器件。因此，验证器与验证器所表示的数据不匹配。由非原子写入导致的这一问题有时被标识为 RAID 写入空洞，其可能在所导致的许多后果当中显现。例如，(一个或多个)存储器件可能难以确定如何补救这一错误，例如所述失配表示错误的验证器、数据集合的非意定改变(例如存储集合的误比特率(BER)的显现)还是对于数据集合或验证器的错误读取。这一信息缺乏甚至可能危害最近没有被写入的数据集合的一部分的准确性的置信度。例如，如果其中一个存储器件发生故障，则针对(利用错误的验证器)从剩余的存储器件恢复该存储器件上的数据的尝试可能导致错误的数据重建。例如，为了从已经丢失或受到破坏的特定卷恢复数据并且用修复的或替换存储器件来替代，可以通过把相同位置(例如其他器件上的相同物理或逻辑地址)处的各个字与所述地址处的字集合的校验和 XOR 在一起来计算所丢失的存储器件上的数据的每一个字，并且该结果得到丢失的字。然而如果校验和已经非有意地改变，则所述 XOR 运算得到错误的结果，并且会用错误的数据来替换所替代卷上的字。作为另一个例子，如果对于包括部分 A 和 B 的数据集合存储验证器 C，并且在更新 A 和 C 时发生灾难性故障，则计算机可能能够识别出数据集合 [A,B] 与验证器 C 之间的失配。这一无能可能不仅会削弱在灾难性故障时正参与写入的 A 和 C 的置信度，而且还会削弱可能甚至长时间没有被访问过的 B 的置信度。

[0007] 这里给出了用于降低由于比如 RAID 写入空洞之类的问题所导致的数据丢失和恢复时间延长的风险以及提高存储集合的效率的技术。根据这些技术，在存储集合当中的一

个或更多存储器件上可以生成一个日志，其被配置成存储将被提交到存储集合的数据集合。所述日志例如可以包括被结构化为一个循环数组的记录序列，其中每一条记录具有用以存储一个数据集合以及为该数据集合计算的验证器的容量。可以首先按照接收顺序把将要写入到存储集合的所有数据集合连同为该数据集合计算的验证器存储在日志中。

[0008] 这些技术可以通过提供一种机制来减轻 RAID 写入空洞的后果，通过所述机制可以在提交到存储集合中的位置之前把非原子写入存储在存储器件的非易失性存储器上。如果在把数据集合写入到日志的同时发生故障，则存储在存储集合中的所述数据集合的版本保持完好；并且如果在把数据集合写入到存储集合的同时发生故障，则可以通过重新发起从日志到存储集合写入所述数据集合来恢复所述故障。此外，日志的使用可以通过促进顺序写入来改进存储器件的性能。作为第一个例子，首先将非顺序数据集合顺序地写入到日志，从而即使对于非顺序数据集合也始终提供快速的顺序写入。作为第二个例子，所述日志可以操作为写入请求与存储集合之间的写入缓冲区，从而实现对包括存储在存储器件的邻接物理位置中的数据序列的各个写入请求的合并。例如通过生成存储在非易失性存储器件上的日志的易失性存储器表示，可以获得另外的性能改进，其中所述易失性存储器表示充当抵达高速缓存和 / 或写入缓冲区。通过使用这里给出的技术，可以可获得这些和其他优点。

[0009] 为了实现前述和相关目的，下面的描述和附图阐述了特定的说明性方面和实现方式。这仅仅表明可以采用一个或更多方面的多种方式当中的几种。通过结合附图考虑下面的详细描述，本公开内容的其他方面、优点和新颖特征将变得显而易见。

附图说明

[0010] 图 1 给出了构成一个存储集合的各个数据集合在存储器件上的示例性存储的图示。

[0011] 图 2 给出了一种示例性情形的图示，其中描绘出在存储集合内的写入操作期间发生故障的后果。

[0012] 图 3 给出了根据这里给出的技术的构成一个存储集合的各个数据集合在存储器件上的示例性存储的图示。

[0013] 图 4 给出了图示根据这里给出的技术的将构成一个存储集合的各个数据集合存储在至少一个存储器件上的第一种示例性方法的流程图。

[0014] 图 5 给出了图示根据这里给出的技术的将构成一个存储集合的各个数据集合存储在至少一个存储器件上的第二种示例性方法的流程图。

[0015] 图 6 是包括处理器可执行指令的示例性计算机可读存储介质的图示，所述处理器可执行指令被配置成根据这里给出的技术将构成一个存储集合的各个数据集合存储在至少一个存储器件上。

[0016] 图 7 是特征在于计算数据集合的验证器的各种技术的一种示例性情形的图示。

[0017] 图 8 是特征在于使用存储在存储器件的非易失性介质上的日志的易失性存储器表示的一种示例性情形的图示。

[0018] 图 9 是特征在于与存储在存储器件上的日志和所述日志的易失性存储器表示之间的存储器件的写入缓冲区的交互的第一示例性情形的图示。

[0019] 图 10 是特征在于与存储在存储器件上的日志和所述日志的易失性存储器表示之间的存储器件的写入缓冲区的交互的第二示例性情形的图示。

[0020] 图 11 示出了可以在其中实施这里所阐述的一项或更多项规定的示例性计算环境。

具体实施方式

[0021] 下面参照附图描述所要求保护的主题内容，其中相同的附图标记始终被用来指代相同的元件。在下面的描述中，出于解释的目的，阐述了许多具体细节以便提供对于所要求保护的主题内容的透彻理解。然而可以明显看出，可以在没有这些具体细节的情况下实践所要求保护的主题内容。在其他事例中，以方框图的形式示出了各个结构和器件以便于描述所要求保护的主题内容。

[0022] A、介绍

[0023] 在计算领域内，许多情形涉及将包括一系列数据集合的存储集合存储在一个或更多存储器件的集合上。例如，用户可能希望在一个硬盘驱动器集合上创建一份档案，并且可以在所述档案内存储一个或更多数据集合（例如字节、字、数据块或序列、文件或记录）。在一些情形中，（一个或多个）存储器件可以完全被分派来存储数据；例如一个两兆兆字节硬盘驱动器可以被配置成提供一个两兆兆字节存储集合。在其他情形中，可以按照促进数据可访问性和 / 或恢复的方式将所述存储集合冗余地存储在各个存储器件上；例如一个一兆兆字节数据集合可以被完全相同地存储在两个一兆兆字节硬盘驱动器上，以便在任一份拷贝受到破坏的情况下提供备份。可以将多个存储器件配置成按照多种方式互操作来存储所述存储集合。

[0024] 在廉价盘冗余阵列（RAID）的各种变型中包括了许多这样的存储方案和特征。作为第二个例子，在 RAID 0 存储方案中，两个或更多硬盘驱动器的整个存储空间可以被分派来存储数据，从而第一硬盘驱动器可以提供对于存储在所述存储集合的一个部分中的数据的访问，而第二硬盘驱动器则并行地提供对于存储在所述存储集合的另一个部分中的数据的访问，从而实际上将对于数据集合的访问速率加倍（并且可能包括存储在其他硬盘驱动器上的所述存储集合的各个部分的其他倍增）。作为第二个例子，在 RAID 1 存储方案中，第一硬盘驱动器可以被完全指派来存储一个存储集合，并且操作为镜像的附加盘驱动器可以存储该存储集合的完全相同的拷贝。可以通过对于由不同硬盘驱动器提供的存储集合的同时访问来获得性能改进。此外，可以在任何硬盘驱动器上访问所述存储集合的一份完整拷贝，并且可以在不危害包括在其中的数据可用性的情况下替换（例如由于破坏、无响应、缺失或损坏而）发生故障的任何硬盘驱动器。然而 RAID 1 方案会显著减少存储集合的容量（例如添加硬盘驱动器不会增大存储集合的容量）。附加的 RAID 变型可以平衡 RAID 0 和 RAID 1 阵列的可访问性、性能和故障恢复属性，同时最大化存储集合的容量。例如，在包括特定尺寸的各个硬盘驱动器的集合的 RAID 4 阵列中，除了其中一个硬盘驱动器之外的所有硬盘驱动器的全部容量可以提供存储空间，而保留的硬盘驱动器则可以存储奇偶校验信息（例如对存储在其他硬盘驱动器上的每一个数据集合的异或（XOR）计算）。这种配置使得存储空间最大化（例如包括四个一兆兆字节硬盘驱动器的 RAID 4 阵列提供三兆兆字节的存储空间），同时还容许一个驱动器的故障；例如如果其中任一个硬盘驱动器完全故障，则可以用替换

硬盘驱动器来将其替换，并且可以利用存储在剩余的硬盘驱动器上的数据来重建故障硬盘驱动器上的数据。例如，可以简单地通过对于存储在各个硬盘驱动器上的对应的数据集合重新计算 XOR 数值来重建故障的奇偶校验硬盘驱动器；并且可以通过使用可用的数据集合和所述 XOR 奇偶校验数值来重建存储在其他硬盘驱动器当中的故障硬盘驱动器上的数据。

[0025] 然而在涉及到把包括各个数据集合的存储集合存储在一个或更多存储器件上的情形中，可能会出现影响所述存储集合的性能和 / 或可靠性的各种低效和问题。图 1 和 2 描绘出可以通过这里给出的技术解决的两个这种问题的实例。

[0026] 在图 1 的示例性情形 100 中，可以把包括一系列数据集合 104（例如存储集合 102 的各个字节、字、数据块、文件或记录）的存储集合 102 存储在存储器件 106（例如硬盘驱动器）上。生成或访问存储集合 102 的过程可以生成涉及各个数据集合 104 的读取和 / 或写入请求集合，并且可以被存储器件 106 接收及实现。例如，硬盘驱动器可以包括悬置在旋转物理介质上方的读取 / 写入头 108，其能够读取在读取 / 写入头 108 下方旋转的物理介质的任何扇区（例如径向数据线）下存储的数据。因此，当接收到将要存储在存储集合 102 中的第一位置 110 处的第一数据集合 104 时，硬盘驱动器可以旋转所述物理介质，直到与存储集合 102 中的位置 110 匹配的物理位置被旋转到读取 / 写入头 108 下方为止，并且随后可以把数据集合 104 写入到物理介质上。然而这样的硬盘驱动器的性能常常受到在把物理介质旋转到适当位置时的延迟的限制。通过顺序访问可以缓解该延迟；例如可以相继写入构成物理介质上的一个物理位置序列的三个数据集合 104，从而把在将各个位置正确地定位在读取 / 写入头 108 下方中的旋转延迟的数目从三个减少到一个。由于旋转延迟常常是硬盘驱动器的吞吐量中的速率限制因素，因此顺序访问可以显著改进存储器件 106 的吞吐量。此外，对于每一个数据集合 104，计算验证器 112（在图 1 的示例性情形 100 中被表示为对于每一个四字节数据集合 104 计算的一个奇偶校验字节）并且将其与数据集合 104 存储在一起，其可以被用来验证数据集合 104 的完整性。图 1 的示例性情形 100 通过在扇区中把验证器 112 附加到数据集合 104 而给出了效率改进，从而可以通过施行一次旋转查找和写入请求来存储数据集合 104 和验证器 112。

[0027] 图 1 图示了特征在于把存储集合 102 的数据集合 104 存储到存储器件 106 的示例性情形 110。在该示例性情形 100 中，由存储器件 106 接收并处理四个数据集合 104 的序列。在第一时间点 114，接收第一数据集合 104 以便存储在存储集合 102 中的第一位置 110 处，并且存储器件 106 施行第一次旋转查找以便将读取 / 写入头 108 定位在物理介质上的相应位置上方，并且将第一数据集合 104 及其验证器 112 写入到物理介质。在第二时间点 116，接收第二数据集合 104 以便存储在存储集合 102 中的第一位置 110 处，并且存储器件 106 施行第二次旋转查找以便将读取 / 写入头 108 定位在物理介质上的相应位置上方，并且将第二数据集合 104 及其验证器 112 写入到物理介质。分别在第三时间点 108 和第四时间点 110 实施附加的旋转查找操作，以便存储第三数据集合 104 和第四数据集合 104。

[0028] 然而，图 1 的示例性情形 100 描绘出在把各个数据集合 104 写入到存储集合 102 中的一些低效源头。作为第一个例子，第一数据集合 104 和第三数据集合 104 构成一个序列（例如存储在存储集合 102 中的接连位置 110 处的数据集合 102），第二存储集合 102 和第四存储集合 102 也构成一个序列。例如，第一过程可以请求针对存储集合 102 的第一写入序列，而第二过程同时请求针对存储集合 102 的一个不同部分的第二写入序列，并且存储器

件 106 可以按照交织方式接收请求。然而在该示例性情形 100 中，存储器件 106 按照接收顺序把各个数据集合 104 写入到物理介质，从而施行四次旋转查找以便写入各个数据集合 104。虽然该写入过程可以严格保留接收到各个写入请求的顺序，但是通过施行第一次旋转查找来存储第一数据集合 104 和第三数据集合 104 并且施行第二次旋转查找来存储第二数据集合 104 和第四数据集合 104，可以获得吞吐量改进。

[0029] 图 1 中描绘的低效的第二个源头来自对于存储集合 102 不必要地写入数据，其迅速被后续写入覆写。作为第一个例子，第二数据集合 104 和第四数据集合 104 都被写入到存储集合 102 中的相同位置 110。对于覆写的识别可以提供通过只写入最近一次写入 202 而改进存储器件 106 的性能的机会(特别在所述覆写所写入的数据与早前的写入相同的情况下)。然而在该示例性情形 100 中，存储器件 106 无法获得这一识别，并且对于存储集合 102 中的相同位置 110 不必要地施行了相同数据集合 104 的两次写入。作为第二个例子，可以对于第一组数据集合 104 (其中包括第一数据集合 104 和第三数据集合 104)计算第一验证器 112 (例如位置 0x0044-0047)，并且可以对于第二组数据集合 104 (其中包括第二数据集合 104 和第四数据集合 104)计算第二验证器 112(例如位置 0xA0F0-A0F3)。由于接收到各个写入请求的序列，存储器件 106 将每一个验证器 110 计算并写入两次(例如对于第一数据集合 104 施行了第一验证器 110 的第一次计算并且与第一数据集合 104 存储在一起；对于第二数据集合 104 施行了第二验证器 110 的第一次计算并且与第二数据集合 104 存储在一起；对于第三数据集合 104 施行了第一验证器 110 的重新计算并且与第三数据集合 104 存储在一起；并且对于第四数据集合 104 施行了第二验证器 110 的重新计算并且与数据集合 104 存储在一起)。这些重新计算本来可以避免从而减少计算和写入的次数，这是通过对于第一数据集合 104 和第三数据集合 104 计算一次第一验证器 112 并且对于第二数据集合 104 和第四数据集合 104 计算一次第二验证器 112 而实现的。这些和其他低效可能是由于存储器件 106 不能标识出减少对于顺序地存储在存储集合 102 中的各个数据集合 104 的写入请求所涉及的计算和 / 或写入的机会而导致的。

[0030] 图 2 给出了示例性情形 200 的图示，其中描绘出可能出现在存储集合 102 中的第二类问题。在该示例性情形 200 中，一组存储器件 106 互操作来存储具有一组验证器 112 的存储集合 102。具体来说，其中三个存储器件 106 存储与特定位置 110 相关联的三个数据集合 104，并且第四存储器件 106 存储所述三个数据集合 104 的验证器 112；例如每一个数据集合 104 可以包括单个比特，并且可以通过将三个比特一起 XOR 来计算验证器 112。(图 2 的示例性情形 200 将每一个数据集合 104 描绘为单个比特以便简化下面的解释，但是可以认识到，数据集合 104 可以具有任何尺寸。) 当写入任何数据集合 104 时，存储在第四存储器件 106 上的验证器 112 被更新以便匹配更新后的数据集合 104。然而如该示例性情形 200 中所示，对于各个存储器件 106 的写入可能不是按照原子并且严格同时的方式发生的，而是可能在不同时间发生；例如可能接收到针对更新存储在一个存储器件 106 上的数据集合 104 和第四数据集合 104 上的验证器 112 两者的请求，但是如果一个存储器件 106 可能空闲而另一个存储器件 106 正进行写入操作，则第一存储器件 106 可能在第二存储器件 106 之前发起和 / 或完成其写入操作。各个存储器件 104 之间的性能(例如旋转速度)和状况(例如写入位置与物理介质的当前旋转位置的距离)的变化也可能导致各个存储器件 106 之间的定时差异。例如，在第一时间点 204，可能请求写入 202 以便更新存储在第三存储器件

106 上的数据集合 104 和存储在第四存储器件 106 上的相应验证器 112。然而第三存储器件 106 可能在第一时间点 204 开始和 / 或完成写入 202，而第四存储器件 106 则可能在第二时间点 206 完成对于验证器 112 的写入 202。

[0031] 在图 2 的示例性情形 200 中描绘的各个存储器件 106 的不完美同步可能在存储服务发生故障的情况下产生不一致。例如，在第三时间点 208，可能请求针对由第三存储器件 106 存储的数据集合 104 和由第四存储器件 106 存储的相应验证器 112 两者的另一次写入 202。然而在该写入过程期间可能发生计算环境的故障 210（例如电力故障或者硬件或软件崩溃）。虽然存储器件 106 和存储阵列常常被设计成耐受许多这样的故障 210，但是该故障 210 可能在第三存储器件 106 已经完成对更新后的数据集合 104 的写入之后但是在第四存储器件 106 写入更新后的验证器 112 之前发生。在第四时间点 212，当各个存储器件 106 再次可访问时（例如当电力恢复时），由第三存储器件 106 存储的数据集合 104 的更新和第四存储器件 106 对验证器 112 的更新失败会给出不一致 ：验证器 112 与相应的数据不再匹配。在使用单个存储器件 106 时可能发生类似的情形 ；例如如果在数据集合 104 的第一拷贝和第二拷贝的更新之间发生故障 210，则存储在具有单个读取 / 写入头 108 的存储器件 106 上的数据集合 104 的各个冗余拷贝可能会给出不一致。

[0032] 这种不一致有时也被标识为“RAID 写入空洞”，其可能导致几个问题。作为第一个例子，可能无法标识出一个或更多数据集合 104 和 / 或验证器 112 当中的哪一个是错误的（例如所述故障可能类似地发生在第四存储器件 106 更新了验证器 112 之后但是在第三存储器件 106 完成更新数据集合 104 之前），从而会危害由验证器 112 所表示的所有数据集合 104 的完整性——即使在写入 202 中甚至没有涉及存储在第一和第二存储器件 106 上的数据集合 104。作为第二个例子，这种不一致可能没有被迅速发现，而是可能在存储集合 102 中持续。随后，如果一个存储器件 106 变为不可用（例如如果第一存储器件 106 完全故障或者被移除），针对重建第一存储器件 106 上的数据的尝试可以利用其他存储器件上的数据，但是所述不一致可能导致错误的数据重建。因此所述阵列就无法提供从单个存储器件 106 的故障进行恢复的预期能力。这些和其他问题可能由各个存储器件 106 在把有关的数据集合 104 存储到存储集合 102 时的互操作中的不完美同步导致。

[0033] B、所给出的技术

[0034] 这里所给出的技术是用于解决在存储情形中可能出现的其中一些问题和 / 或低效的技术，其中可能包括在图 1 和图 2 的示例性情形中所图示的那些问题和 / 或低效。根据这些技术，可以在存储存储集合 102 的其中一个或更多存储器件 106 上生成日志。可以首先把将要写入到存储集合 102 的各个数据集合 104 和验证器 112 写入到所述日志。此外，所述日志可以被结构化为按照各个数据集合 104 的写入顺序的序列结构化的各个数据集合 104 的序列，从而使得接收到针对把各个数据集合 104 写入在存储集合 102 的各个（非顺序）位置 110 中的请求流的存储器件 106 可以首先把各个数据集合 104 顺序地存储在日志中。此外，为了把数据提交到存储集合 102 中的所请求的位置 110，可以选择能够可以作为一个批次被写入的若干数据集合。例如，所述日志可以把将要写入到存储集合 102 的各个数据集合 104 顺序地排入队列，并且可以从队列的前方周期性地选择能够在相同批次中写入的一组数据集合 104。

[0035] 图 3 给出了示例性情形 330 的图示，其中描绘出通过使用存储在存储器件 106 的

非易失性介质上的日志 302 而把各个数据集合 104 和相应的验证器 112 存储到存储集合 102。在该示例性情形 300 中，在被配置成存储构成存储集合 102 的至少一部分的各个数据集合 104 的存储器件 106 上生成日志 302，其被配置成在把所述数据集合 104 和相应的验证器 112 提交到存储集合 102 之前存储这样的数据。具体来说，在该示例性情形 300 中，日志 302 被结构化为各条记录 304 的序列，所述记录 304 存储一个数据集合 104、该数据集合 104 在存储集合 102 中的位置 110 以及该数据集合 104 的验证器 112。日志 302 的各条记录 304 通过尾部指针 306 和头部指针 308 被结构化为一个队列，其中尾部指针 306 标识队列的开头（即日志 304 中的最早数据集合 104），头部指针 308 标识队列的末尾（即被写入到日志 304 的最近数据集合 104）。在第一时间点 310，日志 302 最初为空（即头部指针 308 和尾部指针 306 指向同一条记录 304）；在接收到将被存储在存储集合 102 中的三个数据集合 104 的序列之后，存储器件 106 可以按顺序将所述三个数据集合 104 记录到日志 302（例如通过移动头部指针 308 以分派各条记录 304，并且随后把各个数据集合 104 写入到各条记录 304 中）。在第二时间点 312，可以接收针对写入三个附加数据集合 104 的请求的序列，并且可以通过递增头部指针 308 并且把各个数据集合 104 写入到日志 302 的各条记录 304 来将其存储在日志 302 中。此外，在第三时间点 314，存储器件 106 可以对于每一个数据集合 104 计算一个验证器 112，并且把每一个验证器 112 写入到日志 302（可能利用存储在日志 302 和 / 或存储集合 102 中的其他数据集合 104）。在第四时间点 316，存储器件 106 可以将一个批次 318 的数据集合 104 提交到存储集合 102，这例如是通过从日志 392 当中选择将要提交的一个批次 318 的数据集合 104 并且将所述数据集合 104 和相应的验证器 112 写入到存储集合 102 而实现的。随后可以把针对已被提交到存储集合 102 的数据集合 104 的记录 304 从日志 302 中去除（例如通过将尾部指针 306 前提经过所述记录 302）。按照这种方式，可以根据这里所给出的技术将数据集合 104 提交到存储集合 102。

[0036] 图 3 中对这里给出的其中一些技术的示例性使用的描绘图示了可以从中可获得的一些潜在优点。作为第一个例子，对数据集合的分批可以把能够被写入到存储集合 102 中的位置 110 的连续序列的两个或更多数据集合 104 合并成一个序列，即使在以下情况下也可以实现：针对构成所述序列的各个数据集合 104 的写入请求与将要存储在存储集合 102 中的其他位置 110 处的其他数据集合 104 交织；这样的数据集合 104 不是按照严格顺序的次序接收到的；以及 / 或者在针对写入各个数据集合 104 的请求之间发生简短的延迟。作为第二个例子，对于数据集合 104 的分批可以改进存储器件 106 的效率，这是通过减少在较短时帧内接收到的相同数据的覆写而实现的。例如，可以把针对覆写特定数据集合 104 的多个请求分组成一个批次，并且可以通过把数据集合 104 单次写入到存储集合 102 中的位置 110 而实现。例如，图 3 的示例性情形 300 中对批次 318 的选择省略了靠近尾部指针的一个数据集合 104，但是该数据集合 104 被存储在日志 302 中的后续写入 202 覆写。作为第三个例子，针对写入由相同验证器 112 表示的各个数据集合 104 的单独要求如果被分组成同一批次的话可以导致对验证器 112 的单次计算和写入，而不是对验证器 112 的几次单独的更新。作为第四个例子，通过保守地选择批次 318（例如不是激进地清空日志 302，而是在其中留下一些记录 304），所述技术可以标识并获得针对将来提高效率的机会。例如，对于批次 318 没有选择将要写入到位置 0x03C0 的数据集合 104，这是因为其最近被接收并且可能后面很快就有针对写入附加数据集合 104 的请求。因此，当针对将一个数据集合 104

写入到位置0x03C1的后续请求被接收到并且被存储在日志302中时，可以对于未来的批次318选择全部两个数据集合104，从而顺次一起提交全部两个数据集合104，而不是发出两次单独的写入202。

[0037] 在图3的示例性情形300中图示的当前公开的技术的第二个潜在优点是减少了RAID写入空洞的发生和后果。把针对更新数据集合104的请求首先提交到存储器件106的非易失性介质上的日志并且随后将数据集合104和验证器112移动到存储集合102，可以避免其间的不一致。例如，如果在把数据集合104和/或验证器112写入到日志的同时发生故障210，则恢复过程可以检测到日志被不完整地写入，并且可以丢弃日志的不完整部分。虽然这一丢弃可能导致写入202的丢失，但是这样的写入202还没有被提交到存储集合102并且没有对请求所述写入202的过程证实，因此可以被安全地丢失。此外，存储集合102的一致性不会受到在RAID写入空洞中涉及的不完整写入的危害。此外，如果在把日志的内容提交到存储集合102的同时发生故障210，则存储器件106可以通过把存储在日志中的数据集合104重新提交到存储集合102而从故障210恢复。按照这种方式，可以改进针对存储集合102的写入202的性能，并且/或者可以减少由于比如RAID写入空洞之类的问题所导致的不一致的发生。通过根据这里给出的技术将数据集合104存储到存储集合102，可以可获得这些和其他优点。

[0038] C、示例性实施例

[0039] 图4给出了这些技术的第一示例性实施例的图示，其被描绘为将数据集合104存储在由至少一个存储器件106提供的存储集合102中的第一示例性方法400。所述示例性方法400例如可以被实施为存储在器件(例如存储器电路、硬盘驱动器的盘片、固态存储器组件或者磁盘或光盘)的存储器组件中的一个指令集合，其在由器件的处理器执行时使得所述器件施行这里所给出的技术。示例性方法400开始于402，并且涉及在处理器上执行404所述指令。具体来说，所述指令被配置成在至少一个存储器件106上生成406日志302，所述日志302被配置成存储分别与一个验证器112相关联的各个数据集合104。所述指令还被配置成在接收到针对把一个数据集合104存储在存储集合102中的位置110处的请求之后，把该数据集合104存储408在日志302中。所述指令还被配置成选择410存储在日志302中的一个批次318的数据集合104，其被提交到存储集合102。对于所述批次318的对应的412数据集合104，所述指令可以通过以下操作实现针对存储集合102的这一提交：计算414一个验证器112；将验证器112存储416在日志302中；以及将数据集合104和所述数据集合104的验证器112存储418在存储集合102中。所述指令还被配置成在把数据集合104和所述数据集合104的验证器112存储418在存储集合102中之后，从日志302中去除该数据集合104。按照这种方式，所述指令就根据这里给出的技术获得了将各个数据集合104存储在存储集合102中，因此示例性方法400结束于420。

[0040] 图5给出了这些技术的第二实施例的图示，其被描绘为将数据集合104存储在由至少一个存储器件106提供的存储集合102中的第二示例性方法500。所述示例性方法500例如可以被实施为存储在器件(例如存储器电路、硬盘驱动器的盘片、固态存储器组件或者磁盘或光盘)的存储器组件中的一个指令集合，其在由器件的处理器执行时使得所述器件施行这里所给出的技术。示例性方法500开始于502，并且涉及向器件发送504指令。具体来说，所述指令被配置成在存储器件106上生成506日志302，所述日志302包括记录304

的序列、头部指针 308 和尾部指针 306。所述指令还被配置成在接收到 508 将要存储在存储集合 102 中的位置 110 处的数据集合 104 之后，将日志 302 的头部指针 308 前提 510 经过新记录 304，并且把所述数据集合 104 存储 512 在该新记录 304 中。所述指令还被配置成选择 514 靠近日志 302 的尾部指针 306 的至少一个所选数据集合 104 以便提交到存储集合 102。对于对应的 516 所选数据集合 104，所述指令被配置成施行以下操作：计算 518 所选数据集合 104 的验证器 112；将所选数据集合 104 的验证器 112 存储 520 在日志 302 中；以及将所选数据集合 104 和所述数据集合 104 的验证器 112 提交 522 到存储集合 102。所述指令还被配置成将日志 302 的尾部指针 306 前提 524 经过包括所述至少一个所选数据集合 104 的记录 304。按照这种方式，所述指令就根据这里给出的技术获得了将各个数据集合 104 存储在存储集合 102 中，因此示例性方法 500 结束于 526。

[0041] 图 6 给出了这些技术的第三示例性实施例，其被图示为包括处理器可执行指令 602 的示例性计算机可读介质 600，所述处理器可执行指令 602 被配置成应用这里给出的技术。这样的计算机可读介质例如可以包括涉及有形器件的计算机可读存储介质，比如存储器半导体（例如利用静态随机存取存储器（SRAM）、动态随机存取存储器（DRAM）和 / 或同步动态随机存取存储器（SDRAM）技术的半导体）、硬盘驱动器的盘片、闪存器件或者磁盘或光盘（比如 CD-R、DVD-R 或软盘），其编码一个计算机可读指令集合，当由比如计算机之类的器件 610 的处理器 612 执行时，所述计算机可读指令使得器件 610 实施这里给出的技术。这样的计算机可读介质还可以包括（作为与计算机可读存储介质不同的一类技术的）各种类型的通信介质，比如可以通过各种物理现象传播的信号（例如电磁信号、声波信号或光学信号）以及在各种有线情形（例如通过以太网或光纤线缆）和 / 或无线情形（例如，比如 WiFi 之类的无线局域网（WLAN）、比如 Bluetooth 之类的个人区域网（PAN）或者蜂窝或无线电网络）中传播的信号，其编码一个计算机可读指令集合，当由器件的处理器执行时，所述计算机可读指令使得所述器件实施这里给出的技术。在一个这样的实施例中，处理器可执行指令 602 可以被配置成施行一种将数据集合 104 存储在由至少一个存储器件 106 提供的存储集合 102 中的方法，比如图 4 的第一示例性方法 400 或图 5 的第二示例性方法 500。本领域普通技术人员可以设想到被配置成根据这里给出的技术操作的许多此类计算机可读介质。

[0042] D、变型

[0043] 在许多方面可以设想到这里讨论的技术的变型，并且一些变型关于这些和其他技术的其他变型可以给出附加的优点和 / 或减少缺点。此外，一些变型可以被组合实施，并且通过协同合作，一些组合可以特征在于附加的优点和 / 或减少的缺点。所述变型可以被合并在各个实施例中（例如图 4 的第一示例性方法 400 和图 5 的第二示例性方法 500），以便为这样的实施例赋予单独的和 / 或协同的优点。

[0044] D1、各种情形

[0045] 在这些技术的各个实施例之间可能有所不同的第一方面涉及可以在其中利用这样的技术的情形。作为第一种变型，这些技术可以被用来管理许多类型的存储集合 102 和数据集合 104，其中包括：分别包括存储在对应地址处的一个数值集合的一个或更多卷；分别包括一个文件集合的文件系统；分别包括一个记录集合的数据库；分别包括一个媒体对象集合的媒体库；分别包括一组应用的应用集合；以及分别包括一个卷和 / 或虚拟机存储器的集合的计算环境服务器。此外，在存储集合 102 内对一个数据集合 104 的标识在不同

情形之间的粒度可以有所不同；例如包括一个卷的存储集合 102 可以利用这些技术进行日志记录并且向存储集合 102 提交包括比特、字节、各种长度的字、各种长度的数据块或者扇区的数据集合 104。

[0046] 作为第二种变型，这些技术可以被用来管理把存储集合 102 和数据集合 104 存储在各种类型的易失性和非易失性存储器件 106 上，其中包括硬盘驱动器、固态存储器件、磁性或光学带存储驱动器以及磁盘或光盘。在存储集合 102 的存储中所涉及的存储器件 106 的数目也可以有所不同；例如这些技术可以被用来管理把存储集合 102 存储在单个存储器件 106 上，存储在存储器件 106 的小且紧密集成的集合（例如 RAID 阵列）上，或者存储在可能潜在地大并且 / 或者潜在地分布广的存储器件 106 的松散集成的集合（例如布置在世界的不同地区并且通过因特网通信的存储器件 106 的集合）上。作为仅仅一个例子，这些技术可以被适配成用于实施在存储器件 106 的各种类型的 RAID 阵列中的不同 RAID 级别。此外，对存储集合 102 进行存储的存储器件 106 还可以是混合的类型，并且可以根据各种分层设置来组织（例如存储集合 102 可以首先被存储在相对高性能的主存储器件 106 上，其被备份到相对低性能的异地档案存储器件 106）。所述技术还可以也根据并且适应于存储集合 102 和存储器件 106 的各种属性来实施，其中包括成本、可用性、可靠性、性能需求、应用于存储集合 102 的敏感性和安全性措施以及存储器件 106 的能力。

[0047] 该第一方面的第三种变型涉及日志 302 与存储器件 106 的关系，特别是在存储集合 102 跨越多个存储器件 106 的情况下。作为第一个这样的例子，对于被分派在一个或更多单独的存储器件 106 上的存储集合 102，可以将日志 302 排他地存储在一个存储器件 106 上。替换地，可以把日志 302 存储在与存储集合 102 的一部分或全部相同的存储器件 106 上。作为第二个这样的例子，可以在各个存储器件 106 当中生成多个日志 302。例如，对于跨越几个存储器件 106 的存储集合 102，可以在每一个存储器件 106 上对于存储在该存储器件 106 上的存储集合 102 部分中所存储的各个数据集合 104 生成日志 302。替换地，各个单独的存储器件 106 上的日志 302 可以不与存储集合 102 中的特定位置相关联；例如将要写入到存储集合 102 的数据集合 104 可以在提交到存储集合 102 之前被存储在任何日志中。这种变型可以提供分散式日志记录过程；例如可以把数据集合 104 写入到具有最短 I/O 队列的存储器件 106 的日志 302，或者对于在地理上分布的存储器件 106 的集合当中共享的存储集合 102，可以写入到给出对写入过程的最高可访问性的存储器件 106（例如在地理上最靠近所述过程并且 / 或者在与写入过程通信时特征在于最低等待时间或最高带宽的存储器件 106）。作为第三个这样的例子，可以将日志 302 冗余地存储为相同存储器件 106 上的两份或更多份拷贝，可以将其存储为两个或更多存储器件 106 上的镜像拷贝，或者将其（例如通过分块）分布在两个或更多存储器件 106 上，以便为日志 302 赋予类似于通过各种 RAID 存储方案所提供的故障包容特征。

[0048] 该第一方面的第四种变型涉及被用来验证对应的数据集合 104 的完整性的验证器 112 的类型。在一些情形中，可以使用相对简单的验证器 112，比如被计算为数据集合 104 的 XOR 的奇偶校验位，或者数据集合 104 的总和或散列码。简单的验证器 112 可能适合于相对低价值的存储集合 102，相对低功率的存储器件 106（例如具有相对慢的硬件、容量受限的存储器和有限电池寿命的便携式器件上的存储器件），以及 / 或者其性能高度重要的存储集合 102，从而可更加快速地计算的验证器 112 可能是有利的。在其他情形中，可以使用

相对复杂的验证器 112，其可以提供附加的数据安全性特征。例如，纠错验证器 112（比如 Hamming 码）可以被用来不仅确定数据集合 104 是否是准确的，还可以确定是否由于其中一个数据集合 104 和 / 或验证器 112 的改变而导致了不一致。此外，对于存储集合 102 中的数据集合 104 的不同集合或类型可以利用不同类型的验证器 112（例如对于更有价值的数据可以利用更加复杂但是耐久的验证器 112）。本领域普通技术人员可以在具有这些和其他类型的变型和细节的许多情形中实施这里给出的技术。

[0049] D2、各种元素

[0050] 在各个实施例当中可能有所不同的第二方面涉及这些技术的各种元素的变型。作为第一种变型，许多技术可以在生成 406 日志 302 时使用。例如，日志 302 可能包括许多类型的数据结构，比如数组、链表、表格、数据库、栈、队列、堆或二元树。不同的实现方式可能会给出各种优点和缺点（例如性能、易于更新、空间效率、计算经济性以及与存储器件 106 和 / 或存储集合 102 的特性的兼容性）。还可以在对存储集合 102 进行存储的不同存储器件 106 上以及 / 或者对于不同类型的数据集合 104 实施不同类型的日志 302。例如，日志 302 可以被结构化为包括头部指针 308 和尾部指针 306 的数组，其可以提供以下优点 ：对于日志 302 的任何记录 304 的快速索引（例如 O(1) 的访问时间），通过操纵头部指针 308 和尾部指针 306 实现的对记录 304 的高效分派和重复使用，以及数据集合 104 在日志 302 中的高效存储（例如通过简单地将新数据集合 104 附加到构成日志 302 的记录 304 的序列）。

[0051] 作为该第二方面的第二种变型，可以按照许多方式来选择 410 将要提交到存储集合 102 的批次 318。作为第一个例子，可以通过许多类型的事件来发起所述选择 410。例如，实施这些技术的器件 610、存储器件 106 或者其他类型的器件可以在检测到许多类型的提交事件之后发起对批次 318 的选择 410。这样的提交事件（构成一个示例性提交事件集合）的一些例子包括 ：涉及日志 302 的容量的日志容量事件（例如日志 302 变满）；涉及存储在日志 302 中的数据集合 104 的持续时间的持续时间事件（例如数据集合 104 早于特定时限，比如超过一分钟之前被存储在日志 302 中的数据集合 104）；涉及针对把日志 302 中的至少一个数据集合 104 提交到存储集合 102 的请求的提交请求事件（例如请求写入 202 数据集合 104 的过程可以请求把数据集合 104 提交到存储集合 102）；以及涉及存储集合 102 的至少一个存储器件 106 的工作负荷的存储器件工作负荷事件（例如存储器件 106 可能检测到输入 / 输出工作的空闲时刻，并且可以利用该空闲时刻刷新来自日志 302 的一些数据集合 104）。许多其他类型的事件可以提示发起把数据集合 104 提交到存储集合 102 的过程。

[0052] 作为该第二方面的该第二种变型的第二个例子，可以按照许多方式选择将要提交到存储集合 102 的数据集合 104 的批次 318。例如，可能有利的是在接收到写入请求之后把第一数据集合 104 到存储集合 102 的提交推迟一段简短的时间，以免后续的写入 202 指明对第一数据集合 104 的覆写以及 / 或者提供顺次跟在第一数据集合 104 之后的附加数据集合 104，其因此可以一起被写入到存储集合 102（如在图 3 的示例性情形 300 中的第四时间点 316 所示）。然而当即将接收到顺次跟随在后的第二数据集合 104 的概率减小的价值比不上在把数据集合 104 存储为日志 302 的一条记录 304 时所涉及的成本和复杂度时，把数据集合 104 的提交推迟一段较长时间可能是不利的。此外，可能有利的是选择构成一个批次 318 的各个数据集合 104，以便改进提交到存储集合 102 的效率。作为第一个这样的例子，当存储在日志 302 中的第一数据集合 104 被选择包括在将要存储在存储集合 102 中的

第一位置 110 处的批次 318 中时，一个实施例还可以选择同样存储在日志 302 中并且将被存储在存储集合 102 中的靠近第一位置 110 的第二位置 110 处的第二数据集合 104 以便包括在所述批次 318 中(例如在存储器件 106 的物理介质上是接连的或者至少在物理上靠近的各个数据集合 104，其可以高效地被一起写入在同一批次 318 中)。作为第二个这样的例子，这些技术的一个实施例可以从某一批次 318 中省略存储在日志 302 中并且将被存储在存储集合 102 中的某一位置 110 处的第一数据集合 104，如果所述实施例确定日志 302 还包括比第一数据集合 104 更新并且将被存储在存储集合 102 中的相同位置 110 处的第二数据集合 104 (即后续覆写)。取代将数据集合 104 包括在批次 318 中，所述实施例可以简单地从日志 302 中去除较早的数据集合 104。

[0053] 作为该第二方面的第三种变型，可以按照许多方式计算 414 验证器 112。作为第一方面的一种明显的变型，在这样的情形中可以利用许多类型的验证器 112，但是另外还可以按照多种方式从可用数据计算验证器 112。作为第一个例子，可以基于其所表示的当前数据集合 104 完全重新计算验证器 112。然而作为第二个例子，当验证器 112 表示几个数据集合 104 并且其中的数据集合 104 的一个子集改变时，可以有可能(并且有时更加高效)从验证器 112 中去除陈旧的数据集合 104 并且在验证器 112 中包括更新后的数据集合 104，而不是从当前数据集合 104 重新计算验证器 112，后者可能涉及从存储集合 102 获取数据集合 104 的剩余部分。

[0054] 图 7 给出了特征在于计算验证器 112 的不同方式的示例性情形 700 的图示。在第一时间点 702，在存储集合 102 中为数据集合 104 和相应的验证器 112 分派了空间。然而数据集合 104 的任何部分和验证器 112 都还没有被写入到存储集合 102。在第二时间点 704，已经接收到数据集合 104 的一部分，并且已经发起针对把数据集合 104 提交到存储集合 102 的请求。为了计算验证器 112，这些技术的一个实施例可以标识出构成数据集合 104 的剩余数据既不存在于日志 302 中也不存在与存储集合 102 中。相应地，所述实施例可以利用数据集合 104 的已提供部分计算验证器 112，并且可以对于数据集合 104 的剩余部分推断出并使用默认数值(例如零)。因此，在第二时间点 704，仅仅利用数据集合 104 的已有部分计算验证器 112。在第三时间点 706，利用数据集合 104 的当前数据计算验证器 112。例如，可能已经提供了完全指明数据集合 104 的新的和更新后的数据(例如构成完全数据集合 104 的一系列写入 202 可能存在于日志 302 中，或者可以从存储集合 102 获取数据集合 104 的不存在于日志 302 中的部分)，并且可以利用构成数据集合 104 的当前数据完全重新计算验证器 112 (例如作为所有当前数据的 XOR)。然而在第四时间点 708，仅仅利用原始验证器 112 以及数据集合 104 的原始版本和新版本根据数据集合 104 的改变来重新计算验证器 112。例如，可以通过把原始验证器 112 与数据集合 104 的原始版本进行 XOR 来计算新的验证器 112，从而从原始验证器 112 反转对数据集合 104 的该部分的添加，并且随后将该数值与数据集合 104 的新版本进行 XOR。这一重新计算对于没有完全被存储在日志 302 中的数据集合 104 可能更加高效；例如这一重新计算可以避免从存储集合 102 读取数据集合 104 的没有被存储在日志 302 中的部分。此外，可以基于从存储集合 102 获取数据集合 104 的该部分的比较成本而在这些重新计算技术之间做出选择(例如对于表示一个大数据集合 104 的验证器 112，可能更加高效的是去除并且包括所述数据集合 104 的小部分的更新，并且从所述数据集合 104 的大部分的更新的当前数据重新计算验证器 112)。

[0055] 作为该第二方面的第四种变型，如果存储集合 102（例如电力故障或软件崩溃）和 / 或一个或更多存储器件 106(例如与存储器件 106 的通信中断，存储器件 106 的硬件、固件或驱动程序故障，或者存储器件 106 的移除或损坏，其后是重新建立通信或者替换存储器件 107）发生了故障 210，则这些技术的一个实施例可以按照许多方式利用日志 302 从故障 210 恢复。作为该第二方面的该第四种变型的第一个例子，这些技术的一个实施例可以简单地检查日志 302，丢弃日志 302 中的任何不完整或不一致的记录 304（例如在故障 210 时被不完整地写入的记录），并且随后重新开始从日志 302 向存储集合 102 提交数据集合 104。在所述过程中，在恢复过程期间可以正确地重写可能被不完整地写入到存储集合 102 的任何数据集合 104，甚至无需检测数据集合 104 到存储集合 102 的不完整写入。

[0056] 作为该第二方面的该第四种变型的第二个例子，可以按照分阶段方式施行从故障 210 的恢复。例如，可能有利的是尽可能快速地从故障 210 恢复（例如为了缩短利用存储集合 102 的服务的停用时间），同时还确保对存储集合 102 的访问提供有效且一致的数据。相应地，在所述恢复的第一阶段期间，这些技术的一个实施例可以首先读取日志 302 的内容（例如在该处将数据集合 104 存储在日志 302 中的存储集合 102 内的位置 110），以便确定是从日志 302 还是从存储集合 102 实现对存储集合 102 的访问。所述恢复随后可以继续到第二阶段，所述第二阶段涉及重新开始从日志 302 向存储集合 102 提交数据集合 104，以便校正由于故障 210 而导致不完整地和 / 或不一致地写入的数据集合 104。因此，所述实施例可以阻断。

[0057] 恢复过程的附加示例性实施例例如可以涉及以下操作 ：扫描存储集合 102 的一部分或全部，以便验证其完整性 ；仅仅对在故障中所涉及的存储器件 106 应用恢复过程（例如仅仅把数据集合 104 从日志 302 重写到被暂时移除的存储器件 106）；以及对于不同的存储器件 106 和 / 或不同的数据集合 104 应用不同的恢复过程（例如在对存储在第二存储器件 106 上的第二日志 302 应用恢复过程之前，对存储在第一存储器件 106 上的第一日志 302 应用恢复过程并且完成其恢复）。本领域普通技术人员可以设想到改变这里给出的技术的各种元素以便应用在不同情形中的许多此类方式。

[0058] D3、日志的易失性存储器表示

[0059] 在这些技术的各个实施例当中可能有所不同的第三方面涉及在实施此类技术的器件 610 的易失性存储器中生成日志 302 的易失性存储器表示。例如，除了在存储器件 106 的非易失性介质上生成的日志 302 之外，这些技术的一个实施例还可以生成易失性存储器表示，其也对存储在日志 302 中的数据集合 104 进行存储并且被保持与日志 302 同步。虽然对日志 302 的易失性存储器表示的生成和维护可能会增加复杂度并且消耗附加的计算资源，但是所述易失性存储器表示在这些技术的实施例中可以提供许多潜在的用途和优点。作为第一个示例性优点，所述易失性存储器表示可以充当对日志 302 的写入缓冲区 ；例如取代把各个单独的数据集合 104 写入到日志 302，一个实施例可以最初把数据集合 104 存储在易失性存储器表示中，并且可以把一块数据集合 104 提交到日志 302，从而利用对日志 302 的分批写入 202 扩展了将数据集合 104 顺序写入 202 到日志 302 的效率增益。作为第二个示例性优点，易失性存储器表示可以充当最近写入的数据集合 104 的读取高速缓存 ；例如取代从存储在相对慢的存储器件 106 上的日志 302 中读取最近写入的数据集合 104，一个实施例可以从易失性存储器表示提供数据集合 104。因此，这些技术的一个实施例可以

尝试根据其在日志 302 和易失性存储器表示中的可用性来获取所请求的数据集合 104。例如，在确定数据集合 104 被存储在易失性存储器内的日志 302 的易失性存储器表示中之后，所述实施例可以获取并给出存储在易失性存储器表示中的数据集合 104；在确定数据集合 104 被存储在存储器件 106 上的日志 302 中之后，获取并给出存储在日志 302 中的数据集合 104；并且在其他情况下可以获取并给出存储在存储器件 106 上的存储集合 102 中的数据集合 104。一个数据集合 104 可能还跨越这些来源当中的两个或更多；例如所请求的数据集合 104 的第一部分可能存在于日志 302 中并且从该日志 302 获取，而所请求的数据集合 104 的第二部分则可能存在于易失性存储器表示中并且从该易失性存储器表示获取（不管该第二部分是否存在于可访问性较低的日志 302 和 / 或存储集合 104 中）。作为第三个示例性优点，与评估日志 302 的内容相比，通过评估易失性存储器表示（其常常提供更加快速的访问）的内容，可以更加高效地施行关于存储在日志 302 中的数据集合 104 的决定（比如对批次 318 的选择 410）。数据集合 104 在器件 610 的快速但是易失性的存储器中的可访问性的这些优点可以与通过将其存储在非易失性存储器件 106 上的日志 302 中而实现的数据集合 104 的耐久性并行地获得。

[0060] 作为该第三方面的第一种变型，所述易失性存储器表示可以被结构化成类似于日志 302，或者可以按照不同的方式生成。例如，虽然可能有利的是把日志 302 结构化成促进在比如硬盘驱动器之类的存储器件 106 上的顺序写入，但是这一优点在提供相对等效的顺序和随机存取的存储器电路中就可能被减弱；因此可以按照另一种方式生成易失性存储器表示，比如根据将在该处存储数据集合 104 的存储集合 102 中的位置 110 进行索引，比如散列表或者比如 Adelson-Velskii-Landis 树之类的 B 树。

[0061] 作为该第三方面的第二种变型，易失性存储器表示可以存储与日志 302 中所存储的相同的数据集合 104，或者可以存储不同的存储集合 104。作为第一个这样的例子，易失性存储器表示可以按照写入缓冲区的形式暂时聚集将要一起写入到日志 302 的新数据集合 104。作为第二个这样的例子，易失性存储器表示可以保留从日志 302 去除的数据集合 104，另外还把易失性存储器表示的多余容量用作易失性存储器读取高速缓存。例如，在把数据集合 104 提交到存储集合 102 并且从日志 302 中去除（并且可能甚至被覆写）之后，易失性存储器表示可以将数据集合 104 保留在存储器中，这是由于过程可能请求最近写入的数据集合 104 的概率相对高。只要易失性存储器表示中仍然有容量，就可以（在把数据集合 104 提交到日志 302 和 / 或存储集合 102 之后）继续这种把数据集合 104 保留在易失性存储器表示中，并且易失性存储器表示可以逐出先前提交的数据集合 104，以便为新近接收到的并且迄今尚未提交的数据集合 104 提供容量。在这种情形中，可能有利的是令易失性存储器表示区分已被提交到日志 302 和 / 或存储集合 102 的数据集合 104 与尚未提交的数据集合 104。例如，在把一个数据集合 104 存储在日志 302 中之后，一个实施例可以把该数据集合 104 存储在日志 302 的易失性存储器表示中，并且把该数据集合 104 标记为不可去除；在把存储于日志 302 中的一个数据集合 104 提交到存储集合 102 之后，所述实施例可以把存储在易失性存储器表示中的该数据集合 104 标记为可去除。随后，为了释放易失性存储器表示中的容量，所述实施例可以仅仅从日志 302 的易失性存储器表示中安全地去除被标记为可去除的数据集合 104。这种变型保持了日志 302 与易失性存储器表示的同步性，同时有利地把易失性存储器表示的闲置容量利用作为读取高速缓存。

[0062] 相反，并且作为该第三方面的第三种变型，可能有利的是在存储已提交或未提交数据集合 104 时不要穷尽易失性存储器表示的容量，而是在易失性存储器内的易失性存储器表示中保留足够的容量以便存储传入的数据集合104。具体来说，可以为一个被配置成对将被存储在存储集合 102 中的数据集合 104 进行存储的缓冲区保留足够的容量，同时日志 302 则从事将其他数据集合 104 提交到日志 302。这种变型进一步把易失性存储器表示利用作为写入缓冲区，以便在不中断存储器件 106 把数据集合 104 从日志 302 提交到存储集合 102 的任务的情况下接受传入的数据集合 104。

[0063] 作为该第三方面的第四种变型，故障 210 的恢复还可以涉及重建日志 302 的易失性存储器表示 802。例如，所述恢复过程可以开始于读取日志 302 以便重新生成易失性存储器表示 302。按照这种方式开始所述重建可能是有利的，例如通过重新建立易失性存储器表示 802 的读取高速缓存和 / 或写入缓冲区特征，从而减轻存储器件 106 存储日志 302 的读取 / 写入工作负荷并且便于把日志 302 中的数据集合 104 提交到存储集合 102 以便覆写由于故障 210 导致的不完整或不一致写入 202 的任务。

[0064] 图 8 给出了一种示例性情形 800 的图示，其中把在非易失性存储器件 106 上生成的日志 302 与器件 610 的易失性存储器中的日志 302 的易失性存储器表示 802 配对。在该示例性情形 800 中，易失性存储器表示 802 被存储为根据存储在日志 302 中的各个数据集合 104 的位置 110 的十六进制地址而组织的 B 树。在第一时间点 804，日志 302 和易失性存储器表示 802 可以存储特定的一组数据集合 104 ；并且当在第二时间点 806，接收将要写入到日志 302 的第二数据集合 104 时，所述第二数据集合 104 可以被存储在易失性存储器表示 802 中并且随后被写入（直接写入或者通过写入缓冲区写入）到日志 302。此外，在第三时间点 806，当把存储在日志 302 中的一个或更多数据集合 104 提交到存储在存储器件 106 上的存储集合 102 时，可以把所述数据集合 104 从日志 302 中去除（例如通过前提尾部指针 306 经过包含所提交的数据集合 104 的记录 302），但是如果易失性存储器表示 802 的空间容量可用，则可以把这些数据集合 104 保留在易失性存储器表示 802 中但是将其标记为可去除（在图 8 的示例性情形 800 中由虚线边界示出）。因此易失性存储器表示 802 可以为新近接收的数据集合 104 提供容量（例如在第四时间点 810 接收的第四数据集合 104）。因此可以根据数据集合 104 在易失性存储器表示 802、日志 302 和存储集合 102 中的可用性来实现读取请求。例如，在第五时间点 812，可能接收到针对三个数据集合 104 的三个读取请求，其中第一个数据集合 104 可以从易失性存储器表示 802 提供（尽管从日志 302 中逐出但是存在于易失性存储器表示 802 中）；第二个数据集合 104 可以从日志 302 提供（在被提交到日志 302 之后已经从易失性存储器表示 802 中去除）；并且第三个数据集合 104 （其已经被从易失性存储器表示 802 和日志 302 中逐出）可以从存储集合 102 获取并提供。这样，在图 8 的示例性情形 800 中图示的这些技术的实施例就通过实施日志 302 的易失性存储器表示 802 而实现了几个优点。本领域普通技术人员可以设想到根据这里给出的技术对日志 302 的易失性存储器表示 802 的此类使用。

[0065] D4、与写入缓冲区的互操作

[0066] 在这些技术的各个实施例当中可能有所不同的第四方面涉及在对存储集合 102 进行存储的存储器件 106 中包括及利用写入缓冲区。在许多情况下，存储器件 106 可以有利地利用写入缓冲区来改进性能，这例如是通过在易失性存储器中对数据集合 104 的写入

202 进行分批直到发起刷新请求为止，并且随后将所有数据集合 104 提交到存储于存储器件 106 上的存储集合 102。然而，存储器件 106 上的写入缓冲区的操作可能会降低这里给出的技术的性能，并且实际上可能导致一些问题。例如，如果针对在日志 302 结果中存储数据集合 104 的请求在易失性写入缓冲区中被延迟，则在发生故障 210 的情况下可能会丢失数据集合 104。具体来说，写入缓冲区常常是按照透明的方式实施的，从而使得操作系统或过程可能难以确定数据集合 104 是否实际已被提交到日志 302（除非肯定地请求了刷新操作并且将其验证为完成）或者甚至难以确定对于存储器件 104 是否存在写入缓冲区。因此，当过程请求把数据集合 104 写入到日志 302 时，存储器件 106 可以迅速向所述过程表明所述请求已被实现，即使所述写入被存储在易失性写入缓冲区中而不是存储在日志 302 的非易失性存储装置中。因此，所述应用可能会错误地操作仿佛数据集合 104 已被提交，并且如果在存储器件 106 从写入缓冲区刷新数据集合 104 之前发生故障 210 则可能会出现不一致和预料之外的数据丢失。类似地，日志 302 与存储集合 102 之间的写入缓冲区的操作可能导致日志 302 会错误地操作仿佛数据集合 104 已被持久地存储；例如所述日志可能会去除尚未被提交到存储集合 102 的数据集合 104，从而在写入缓冲区被刷新之前发生故障 210 的情况下会导致不完整和不一致的数据集合 104。此外，写入缓冲区可能提出的优点（例如分批的写入 202、顺序写入 202 的合并以及减少覆写）已经由这里给出的技术的其他组件提供。因此可以认识到，写入缓冲区的存在和操作会导致增加复杂度、增加开销、潜在的性能降低以及预料之外的结果，并且只会提供很少的或者没有尚未通过这里给出的技术实现的优点。

[0067] 鉴于这些潜在的缺点，可以根据写入缓冲区的存在来调节这些技术的实施例。作为该第四方面的第一种变型，这些技术的一个实施例可以通过多种方式避免写入缓冲区的使用和效果。作为该第一种变型的第一个例子，当把数据集合 104 和验证器 112 写入到日志 302 时，绕过写入缓冲区，这例如是通过把针对日志 302 的写入作为直写（write-through）请求发出，或者通过简单地禁用存储器件 106 上的写入缓冲区。作为该第一种变型的第二个例子，所述实施例可以通过在针对存储在存储器件 106 上的日志 302 和 / 或存储集合 102 的每一次写入 202 之后发出刷新请求来取消写入缓冲区的效果。虽然频繁发出刷新请求可能会降低存储器件 106 的性能，但是可以通过多种方式来减少所述性能损失；例如如果存储集合 102 和 / 或日志 302 被分布在分别可能包括或者可能不包括写入缓冲区的一组存储器件 106 上，则这些技术的一个实施例可以被配置成只对存储最近写入的数据集合 104 的存储器件 106 发出刷新请求。

[0068] 作为该第四方面的第二种变型，这些技术的一个实施例可以与写入缓冲区互操作，并且可以把写入缓冲区的操作与日志 302 和 / 或日志 302 的存储器内表示 802 的操作进行协调。作为该第二种变型的第一个例子，当刷新存储日志 302 的存储器件 106 时，可以标识出日志 302 的刷新点，其表示已被刷新到日志 302 的数据集合 104（与写入请求已被发出到日志 302 但是可能仍然保留在写入缓冲区中的数据集合 104 相对）。例如，在特征在于日志 302 的易失性存储器表示 802 的一个实施例中，最初可以把存储在易失性存储器表示 802 中的数据集合 104 标记为不可去除，并且可以保持如此标记直到日志 302 的冲刷点被移动经过所述数据集合 104 为止，此时易失性存储器表示 802 可以把所述数据集合 104 标记为可去除。

[0069] 图 9 给出了一种示例性情形 900 的图示，其特征在于将易失性存储器表示 802 适

配成与存储日志 302 的存储器件 106 的写入缓冲区 902 互操作。在该示例性情形 900 中，将要写入到存储集合 102 的数据集合 104 首先被存储在易失性存储器表示 802 中，并且随后在提交到存储集合 102 之前被写入到日志 302。然而写入缓冲区 902 可能在存储日志 302 的存储器件 106 上存在，并且可能会导致不一致以及问题，例如如果被认为已经写入到日志 302 的数据集合 104 相反地被存储在写入缓冲区 902 的易失性存储器中，并且故障 210 会导致数据集合 104 丢失而不会被写入到日志 302。相应地，易失性存储器表示 902 可以记录对应的数据集合 104 的状态。例如，在第一时间点 904，在发起针对把一组数据集合 104 移动到日志 302 的请求之后，易失性存储器表示 802 可以记录数据集合 104 的状态为处于被写入到存储器件 106 的过程中(即表明已经发起了针对把数据集合 104 写入到存储器件 106 的请求，但是存储器件 106 还没有表明已经接收到写入请求)。在第二时间点 906，存储器件 106 做出响应表明接收到数据集合 104。然而易失性存储器表示可能无法确定数据集合 104 是否已被提交到日志 302，或者数据集合 104 是否驻留在写入缓冲区 902 中。相应地，在第二时间点 906，易失性存储器表示 902 把所述数据集合 104 标记为已由存储器件 106 缓冲。与此同时，尚未被存储器件 104 确认为完全接收到的其他数据集合 104 可以继续被标记为处于写入到存储器件 106 的过程中。作为第三时间点 908，易失性存储器表示 802 可以发出针对刷新写入缓冲区 902 的请求，并且写入缓冲区 902 可以开始把已经完全接收到的数据集合 104 提交到包括日志 302 的非易失性存储介质。在第四时间点 910，当存储器件 106 表明所述刷新请求已被实现时，易失性存储器表示可以把先前被标记为已缓冲的所有数据集合 104 (即存储器件 106 确认为在刷新请求之前完全接收到的所有数据集合 104)标记为已被完全记入日志并且可选地标记为可去除。在第五时间点 912，存储器件 106 随后可以表明已经完全接收到附加的数据集合 104，易失性存储器表示 802 可以把这些数据集合 104 标记为已缓冲并且准备好通过第二个刷新请求提交到日志 302。通过这种方式，易失性存储器表示 802 跟踪数据集合 104 关于存储器件 106 的写入缓冲区 902 的状态。

[0070] 作为该第四方面的第三种变型，写入缓冲区 902 还可以作为从日志 302 向存储器件 106 提交数据集合 104 的中介并且可能与之发生干扰。按照类似的方式，存储在易失性存储器表示 802 和 / 或日志 302 中的数据集合 104 的状态可以表明数据集合 104 是否已从日志 302 被刷新到存储集合 102。例如在检测到从写入缓冲区 902 向存储集合 102 提交数据集合 104 之后(例如对刷新请求的确认)，这些技术的一个实施例可以把日志 302 和 / 或易失性存储器表示 902 中的数据集合 104 标记为已提交，并且可以只把被标记为已提交到存储集合 104 的数据集合 104 从日志 302 和 / 或易失性存储器表示 902 中去除。

[0071] 图 10 给出了一种示例性情形 1000 的图示，其特征在于调节日志 302 以便与对存储集合 102 进行存储的存储器件 106 的写入缓冲区 902 互操作。在该示例性情形 1000 中，日志 302 存储将要提交到分布在三个存储器件 106 上的存储集合 102 的数据集合 104，其中每一个存储器件 106 包括一个写入缓冲区 902。为了确保将数据集合 104 完全提交到存储有存储集合 102 的各个存储器件 106 的物理介质，日志 302 可以记录已经由日志 302 请求写入到存储集合 102 的数据集合 104 的状态。例如，在第一时间点 1002，在标识出将要提交的一个批次 318 的数据集合 104 之后，日志 302 可以发送针对每一个数据集合 104 去到对该数据集合 104 进行存储的存储器件 106 的写入请求。然而当存储器件 106 确认接收到所述数据集合 104 时，日志 302 可能不认为所述数据集合 104 已被提交到存储集合 102，而

是可能被存储在每一个存储器件 106 上的易失性写入缓冲区 902 中，因此可能把日志 302 中的每一个数据集合 104 标记为已被缓冲。在第二时间点 1004，日志 302 可以向各个存储器件 106 发出刷新请求（并且具体来说是只向存储已缓冲数据集合 106 的存储器件 106 发送；例如第三存储器件 106 没有存储任何最近提交的数据集合 104，因此不向其发出刷新请求）。在第三时间点 1004，当存储器件 106 表明已经实现了刷新请求时，日志 302 可以把所述数据集合 104 标记为已被提交。日志 302 还可以在头部指针 308 与尾部指针 306 之间标识出一个刷新点 1008，从而使得所述刷新点与尾部指针 306 之间的所有数据集合 104 都已被提交到存储集合 102。在第四时间点 1010，日志 302 随后可以通过把尾部指针 308 移动到刷新点 1008 而逐出各个数据集合 104，这是因为其间的任何数据集合 104 都已被完全提交到存储集合 102。按照这种方式，日志 302 可以被适配成考虑到存储有存储集合 102 的存储器件 106 的写入缓冲区 902 的操作。在实施这里给出的技术的同时，本领域普通技术人员可以设想出考虑写入缓冲区 902 的存在和操作的许多方式。

[0072] E、计算环境

[0073] 图 11 给出了可以在其中实施这里给出的技术的计算器件 1102 内的示例性计算环境的图示。示例性的计算器件包括（但不限于）个人计算机、服务器计算机、手持式或膝上型器件、移动器件（比如移动电话、个人数字助理（PDA）、媒体播放器等等）、多处理器系统、消费电子装置、小型计算机、大型计算机以及包括任何前述系统或器件的分布式计算环境。

[0074] 图 11 图示了包括被配置成实施这里提供的一个或更多实施例的计算器件 1102 的系统 1100 的一个例子。在一种配置中，计算器件 1102 包括至少一个处理器 1106 和至少一个存储器组件 1108。取决于计算器件的确切配置和类型，存储器组件 1108 可以是易失性的（比如 RAM）、非易失性的（例如比如 ROM、闪存等等）或者中间或混合类型的存储器组件。这种配置在图 11 中由虚线 1104 图示。

[0075] 在一些实施例中，器件 1102 可以包括附加的特征和 / 或功能。例如，器件 1102 可以包括一个或更多附加的存储组件 1110，其中包括（但不限于）硬盘驱动器、固态存储器件以及 / 或者其他可移除或不可移除磁性或光学介质。在一个实施例中，实施这里提供的一个或更多实施例的计算机可读和处理器可执行指令被存储在存储组件 1110 中。存储组件 1110 还可以存储其他数据对象，比如操作系统的组件、构成一个或更多应用的可执行二进制档、编程库（例如应用编程接口（API））、媒体对象以及文档。计算机可读指令可以被加载到存储器组件 1108 中以便由处理器 1106 执行。

[0076] 计算器件 1102 还可以包括一个或更多通信组件 1116，其允许计算器件 1102 与其他器件进行通信。所述一个或更多通信组件 1116 可以包括（例如）调制解调器、网络接口卡（NIC）、射频发送器 / 接收器、红外端口以及通用串行总线（USB）USB 连接。这样的通信组件 1116 可以包括有线连接（通过物理线绳、线缆或连线连接到网络）或无线连接（比如通过可见光、红外或者一个或更多无线电频率与联网器件无线通信）。

[0077] 计算器件 1102 可以包括一个或更多输入组件 1114，比如键盘、鼠标、笔、语音输入器件、触摸输入器件、红外摄影机或视频输入器件，以及 / 或者一个或更多输出组件 1112，比如一个或更多显示器、扬声器和打印机。输入组件 1114 和 / 或输出组件 1112 可以通过有线连接、无线连接或其任何组合连接到计算器件 1102。在一个实施例中，来自另一个计算器件的输入组件 1114 或输出组件 1112 可以被用作计算器件 1102 的输入组件 1114 和 / 或

输出组件 1112。

[0078] 计算器件 1102 的各个组件可以通过各种互连(比如总线)进行连接。这样的互连可以包括外围组件互连(PCI)(比如 PCI Express)、通用串行总线(USB)、火线(IEEE 1394)、光学总线结构等等。在另一个实施例中,计算器件 1102 的各个组件可以通过网络互连。例如,存储器组件 1108 可以包括位于通过网络互连的不同物理位置的多个物理存储器单元。

[0079] 本领域技术人员将认识到,被利用来存储计算机可读指令的存储器件可以分布在网络上。例如,可通过网络 1118 访问的计算器件 1120 可以存储用以实施这里所提供的一个或更多实施例的计算机可读指令。计算器件 1102 可以访问计算器件 1120,并且下载所述计算机可读指令的一部分或全部以便执行。替换地,计算器件 1102 可以下载所需要的多条计算机可读指令,或者一些指令可以在计算器件 1102 处执行,并且一些指令可以在计算器件 1120 处执行。

[0080] F、术语的使用

[0081] 虽然用特定于结构特征和/或方法动作的语言描述了本主题内容,但是要理解的是,在所附权利要求中限定的主题内容不一定受限于上面描述的具体特征或动作。相反,上面描述的具体特征和步骤是作为实施权利要求的示例性形式而公开的。

[0082] 在本申请中使用的术语“组件”、“模块”、“系统”、“接口”等等通常意图指代与计算机有关的实体——硬件、硬件与软件的组合、软件或执行中的软件。例如,组件可以是(但不限于)运行在处理器上的过程、处理器、对象、可执行程序、执行线程、程序和/或计算机。作为说明,运行在控制器上的应用和控制器都可以是一个组件。一个或更多组件可以驻留在一个过程和/或执行线程内,并且一个组件可以位于一台计算机上和/或分布在两台或更多台计算机之间。

[0083] 此外,所要求保护的主题内容可以被实施为一种利用标准编程和/或工程技术来产生软件、固件、硬件或其任何组合以便控制计算机实施所公开的主题内容的方法、设备、或者制造产品。这里使用的术语“制造产品”意图包含可从任何计算机可读器件、载体或介质访问的计算机程序。当然,本领域技术人员将认识到,在不背离所要求保护的主题内容的范围或精神的情况下,本领域技术人员可以对这种配置做出许多修改。

[0084] 这里提供了各个操作实施例。在一个实施例中,所描述的其中一项或更多项操作可以构成存储在一个或更多计算机可读介质上的计算机可读指令,在由计算器件执行时所述计算机可读指令将使得所述计算器件施行所描述的操作。描述其中一部分或全部操作的顺序不应被理解为意味着这些操作一定是顺序相关的。受益于本说明书,本领域技术人员将认识到替换的排序。此外还将理解的是,不一定所有操作都存在于这里提供的每一个实施例中。

[0085] 此外,这里使用的“示例性”一词意图充当一个实例、事例或说明。在这里被描述为“示例性”的任何方面或设计不一定应被理解为比其他方法或设计有利。相反,使用“示例性”一词是意图以具体的方式呈现概念。本申请中所使用的术语“或者”意图意味着包含性的“或者”而不是互异性的“或者”。也就是说,除非另行指明或者从上下文明显看出,否则“X 采用 A 或 B”意图意味着任何自然的包含性排列。也就是说,如果 X 采用 A、X 采用 B 或者 X 同时采用 A 和 B,则“X 采用 A 或 B”在任何前述事例下都得以满足。此外,除非另行指明或者从上下文中明显看出是针对单数形式,否则用在本申请和所附权利要求书中的冠

词“一个 / 一项”和“某一”通常可以被理解为意味着“一个 / 一项或更多个 / 更多项”。

[0086] 此外，虽然关于一种或更多种实现方式示出并描述了本公开内容，但是基于阅读并理解本说明书和附图，本领域技术人员将会想到等效的更改和修改。本公开内容包括所有这样的修改和更改，并且仅由所附权利要求书的范围限制。特别关于由上面描述的组件（例如元件、资源等等）所施行的各项功能，被用来描述这样的组件的术语意图对应于（除非另行表明）施行所述组件的指定功能的任何组件（例如功能上等效），尽管其在结构上不等效于在本公开内容的这里说明的示例性实现方式中施行所述功能的所公开的结构。此外，虽然本公开内容的某一项具体特征可能是关于几种实现方式当中的仅仅一种公开的，但是这样的特征可以与其他实现方式的一项或更多项其他特征组合，正如可能对于任何给定或特定应用所期望且有利的那样。此外，就在详细描述或权利要求书中使用“包括”、“具有”、“带有”或其变型而言，这样的术语以与术语“包括”类似的方式都意图是包含性的。

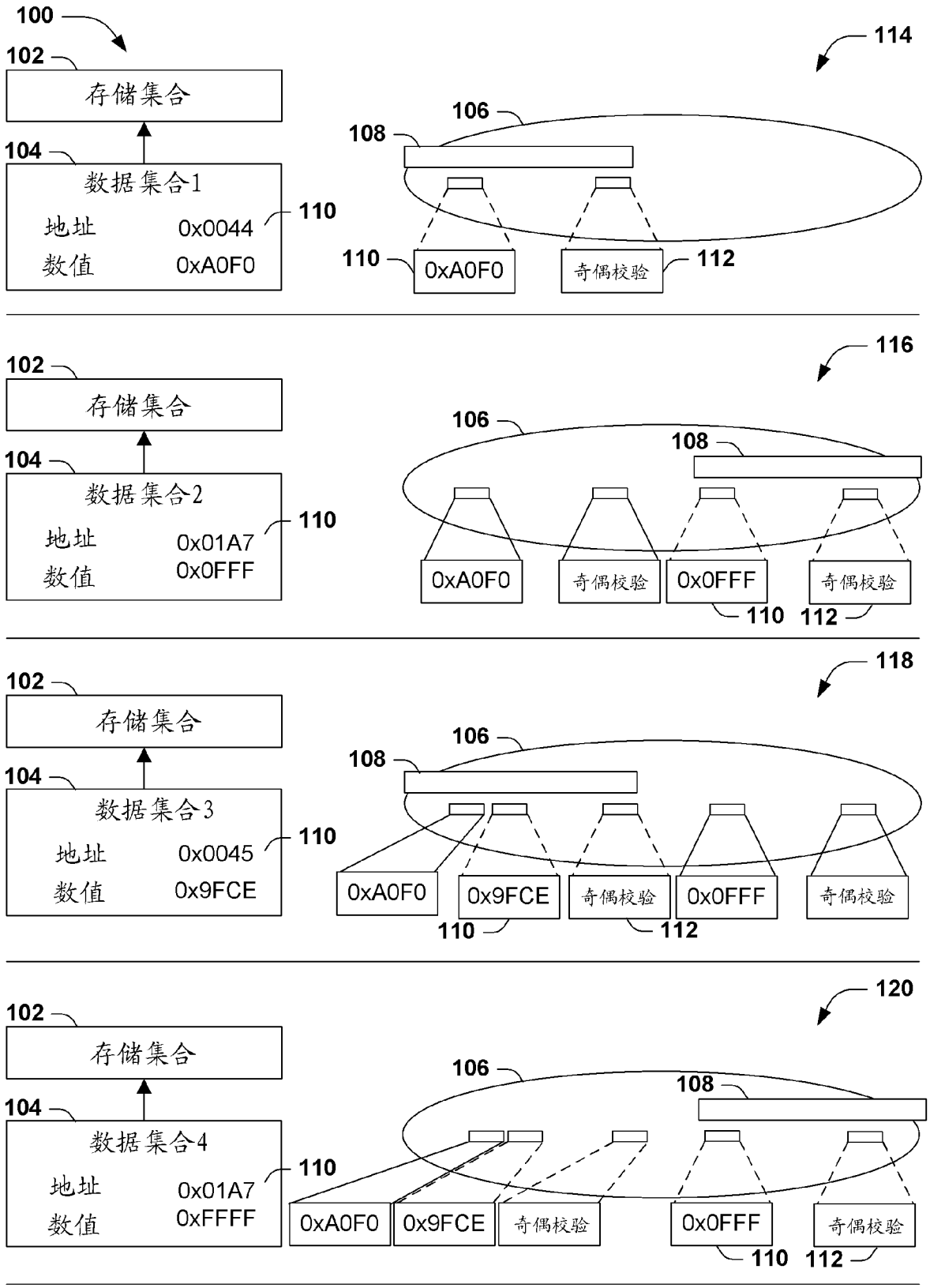

100
102
存储集合
104
数据集合1
地址 0x0044
数值 0xA0F0
110
114
106
108
110
0xA0F0
奇偶校验
112
102
存储集合
104
数据集合2
地址 0x01A7
数值 0x0FFF
110
116
106
108
0xA0F0
奇偶校验
0x0FFF
奇偶校验
110
112
102
存储集合
104
数据集合3
地址 0x0045
数值 0x9FCE
110
118
106
108
0xA0F0
0x9FCE
奇偶校验
0x0FFF
奇偶校验
110
112
102
存储集合
104
数据集合4
地址 0x01A7
数值 0xFFFF
110
120
106
108
0xA0F0
0x9FCE
奇偶校验
0x0FFF
奇偶校验
110
112

图 1

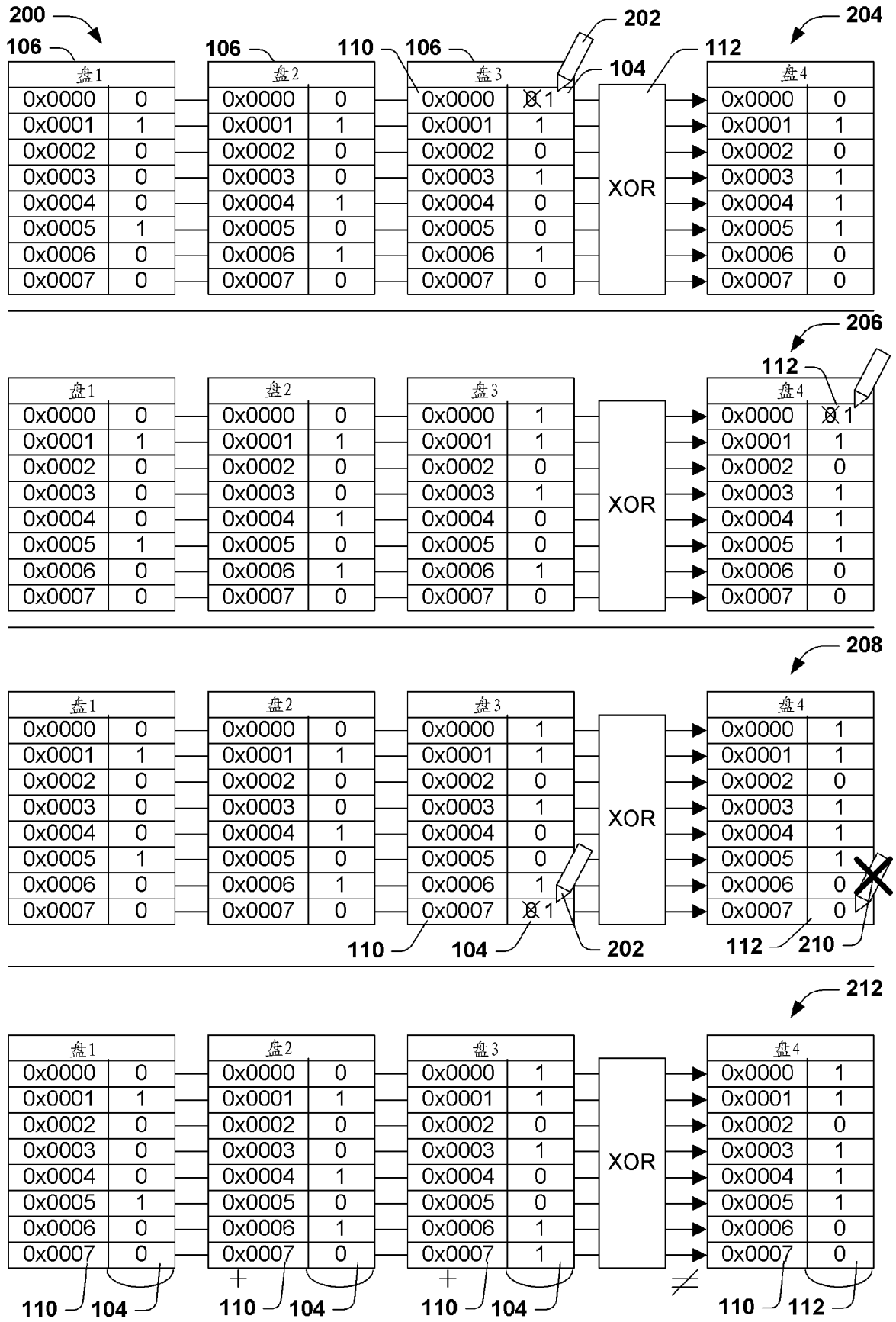

200
106
106
110
106
202
204
104
112
盘1
盘2
盘3
盘4
0x0000 0
0x0001 1
0x0002 0
0x0003 0
0x0004 0
0x0005 1
0x0006 0
0x0007 0
0x0000 1
0x0001 1
0x0002 0
0x0003 1
0x0004 0
0x0005 0
0x0006 1
0x0007 0
XOR
206
208
210
212
110
104
202
112
≠

图 2

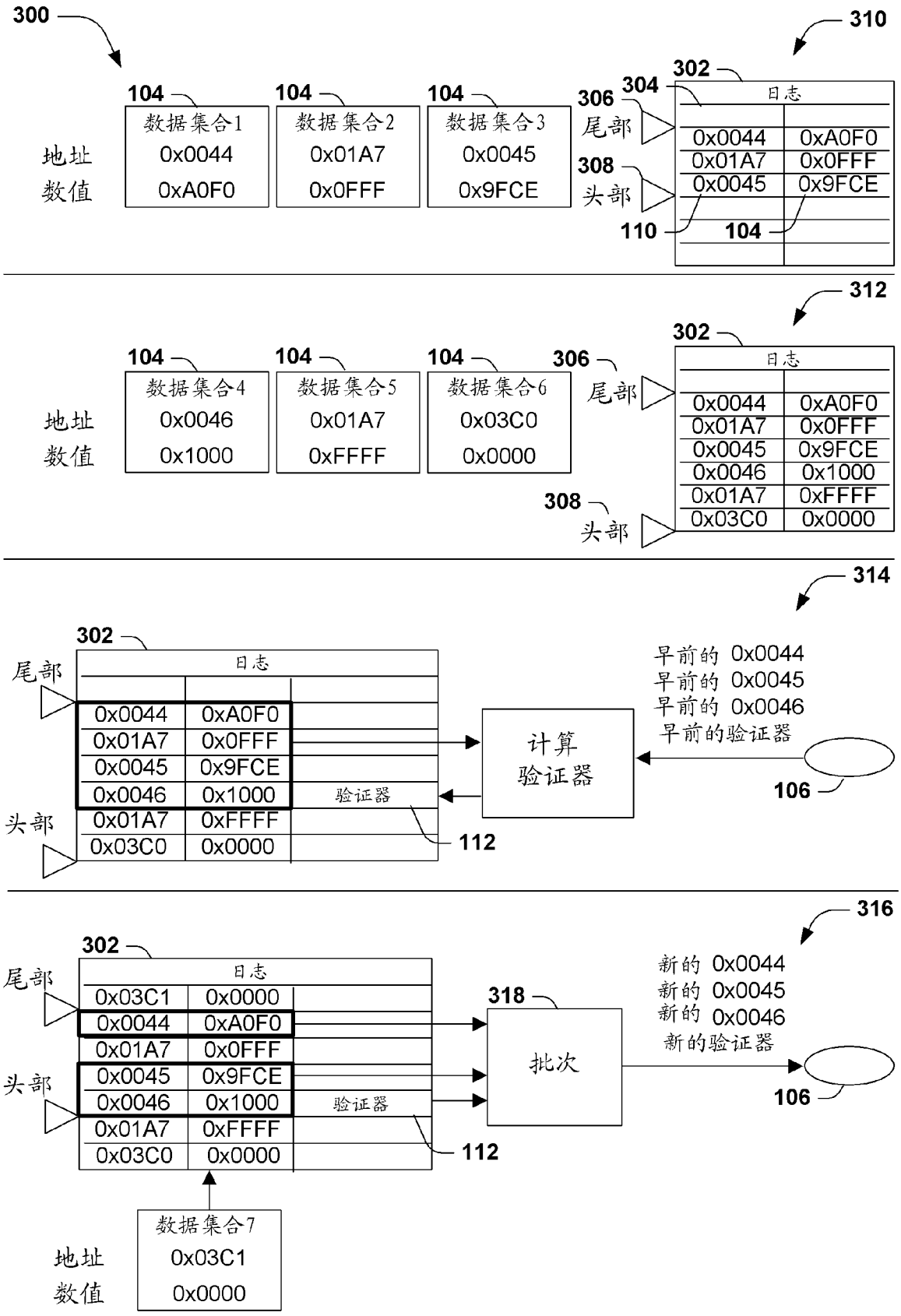

300
310
302
304
306
尾部
308
头部
110
104
104
104
数据集合1
数据集合2
数据集合3
地址
数值
0x0044
0xA0F0
0x01A7
0x0FFF
0x0045
0x9FCE
日志
0x0044 0xA0F0
0x01A7 0x0FFF
0x0045 0x9FCE
312
302
306
尾部
308
头部
104
104
104
数据集合4
数据集合5
数据集合6
地址
数值
0x0046
0x1000
0x01A7
0xFFFF
0x03C0
0x0000
日志
0x0044 0xA0F0
0x01A7 0x0FFF
0x0045 0x9FCE
0x0046 0x1000
0x01A7 0xFFFF
0x03C0 0x0000
314
302
尾部
头部
日志
0x0044 0xA0F0
0x01A7 0x0FFF
0x0045 0x9FCE
0x0046 0x1000
0x01A7 0xFFFF
0x03C0 0x0000
验证器
112
计算
验证器
早前的 0x0044
早前的 0x0045
早前的 0x0046
早前的验证器
106
316
302
尾部
头部
日志
0x03C1 0x0000
0x0044 0xA0F0
0x01A7 0x0FFF
0x0045 0x9FCE
0x0046 0x1000
0x01A7 0xFFFF
0x03C0 0x0000
验证器
112
318
批次
新的 0x0044
新的 0x0045
新的 0x0046
新的验证器
106
数据集合7
地址
数值
0x03C1
0x0000

图 3

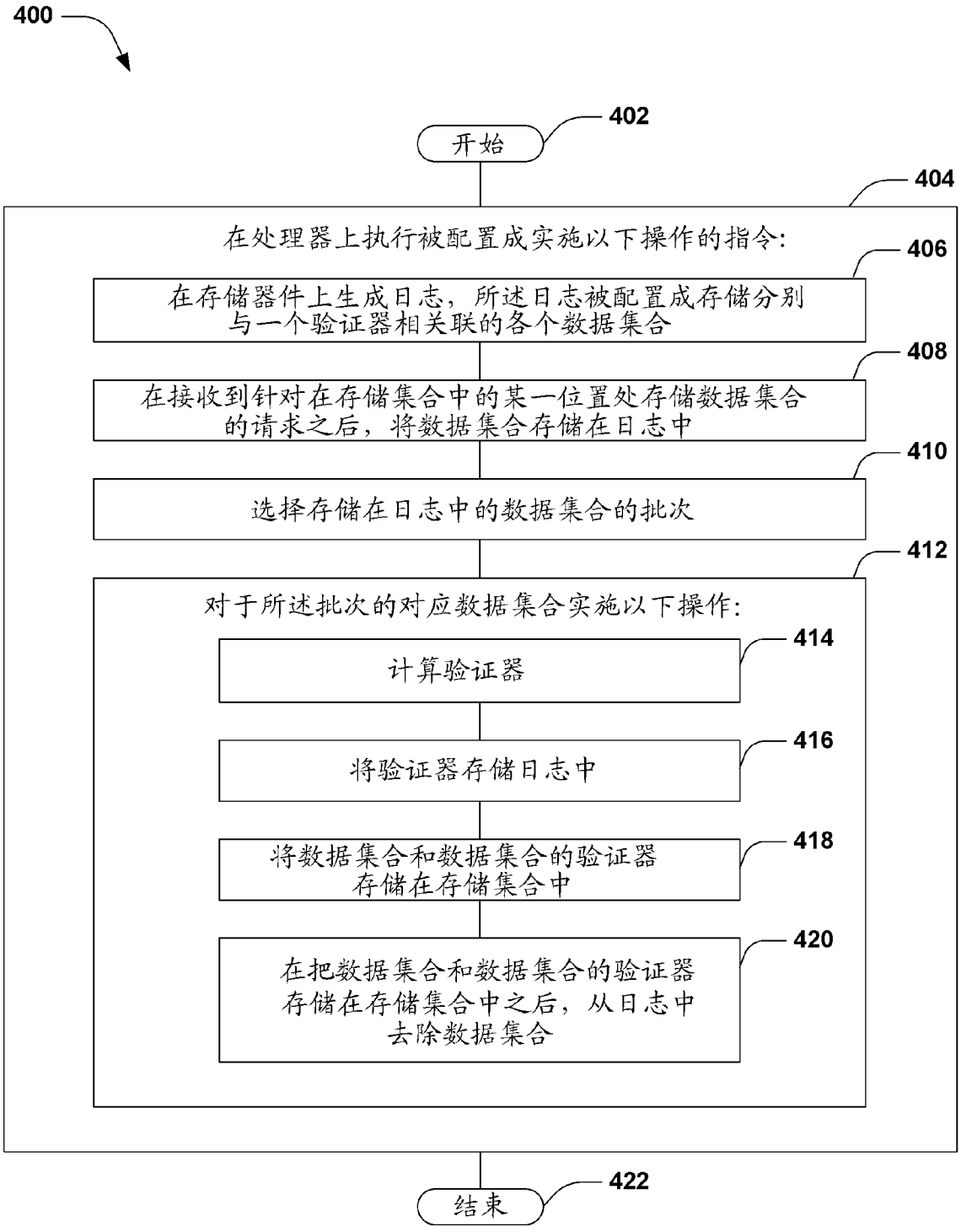

400
开始
402
404
在处理器上执行被配置成实施以下操作的指令:
406
在存储器件上生成日志，所述日志被配置成存储分别与一个验证器相关联的各个数据集合
408
在接收到针对在存储集合中的某一位置处存储数据集合的请求之后，将数据集合存储在日志中
410
选择存储在日志中的数据集合的批次
412
对于所述批次的对应数据集合实施以下操作:
414
计算验证器
416
将验证器存储日志中
418
将数据集合和数据集合的验证器存储在存储集合中
420
在把数据集合和数据集合的验证器存储在存储集合中之后，从日志中去除数据集合
422
结束

图 4

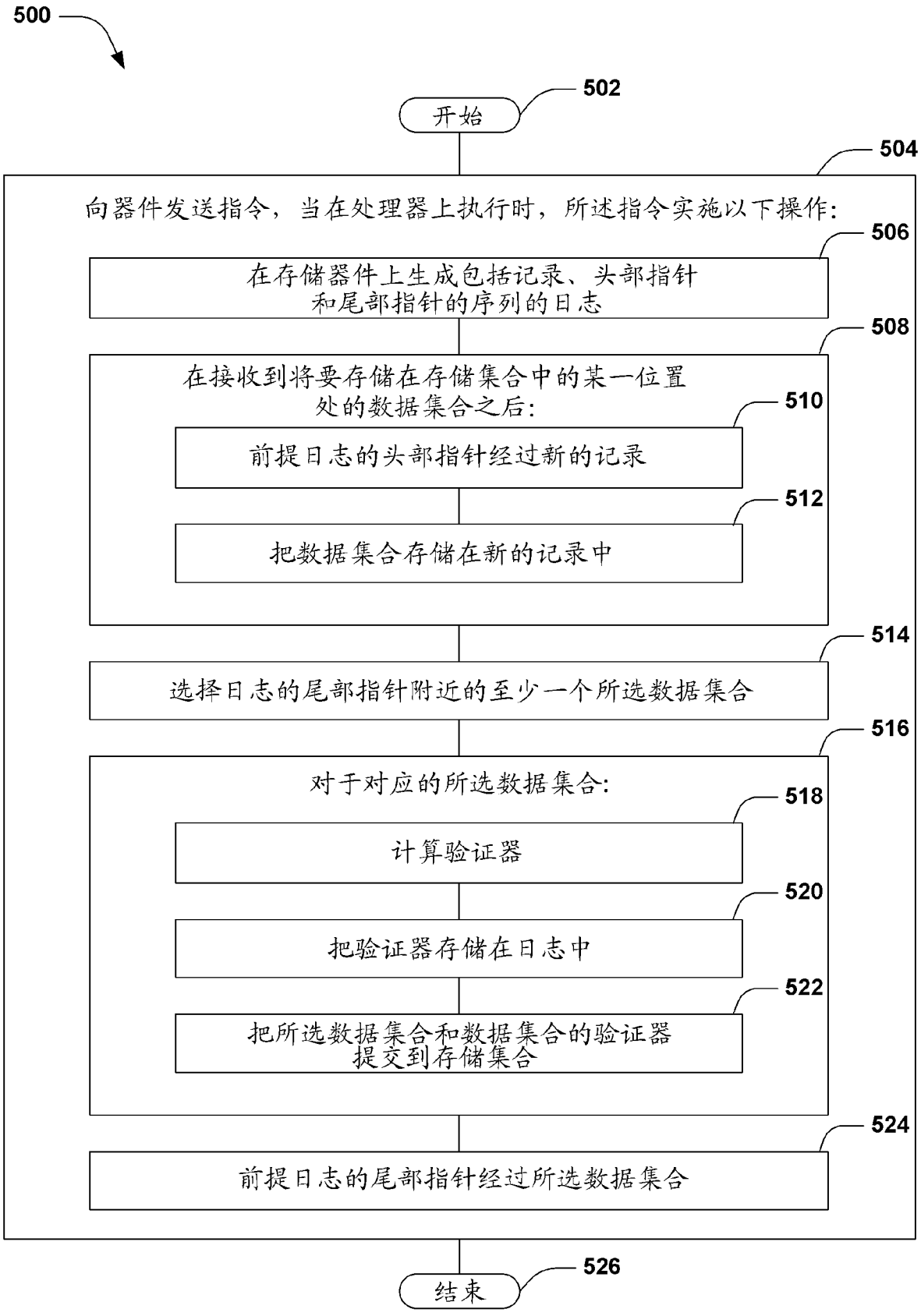

500
开始
502
504
向器件发送指令，当在处理器上执行时，所述指令实施以下操作：
506
在存储器件上生成包括记录、头部指针和尾部指针的序列的日志
508
在接收到将要存储在存储集合中的某一位置处的数据集合之后：
510
前提日志的头部指针经过新的记录
512
把数据集合存储在新的记录中
514
选择日志的尾部指针附近的至少一个所选数据集合
516
对于对应的所选数据集合：
518
计算验证器
520
把验证器存储在日志中
522
把所选数据集合和数据集合的验证器提交到存储集合
524
前提日志的尾部指针经过所选数据集合
结束
526

图 5

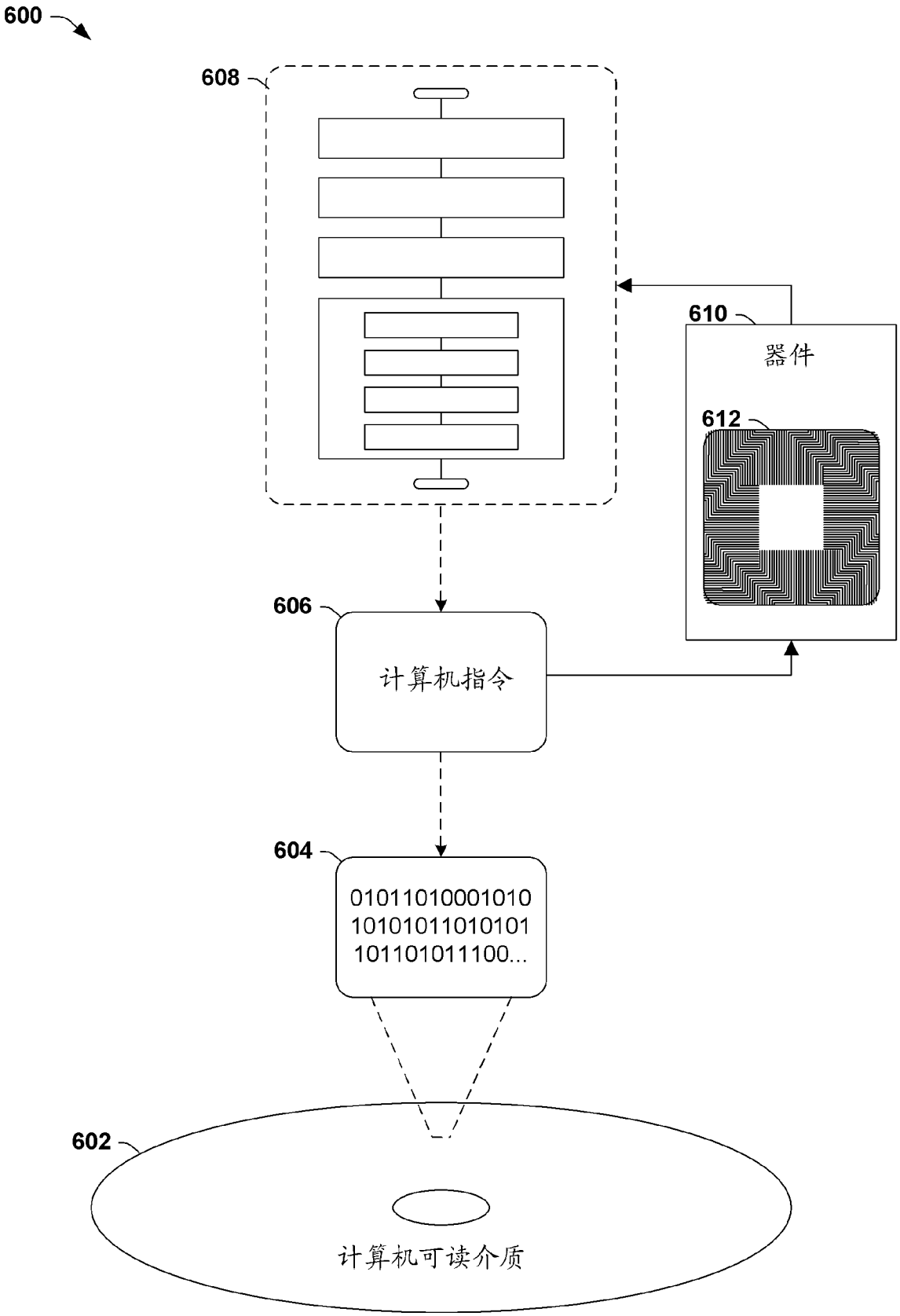

600
608
610
器件
612
606
计算机指令
604
01011010001010
10101011010101
101101011100...
602
计算机可读介质

图 6

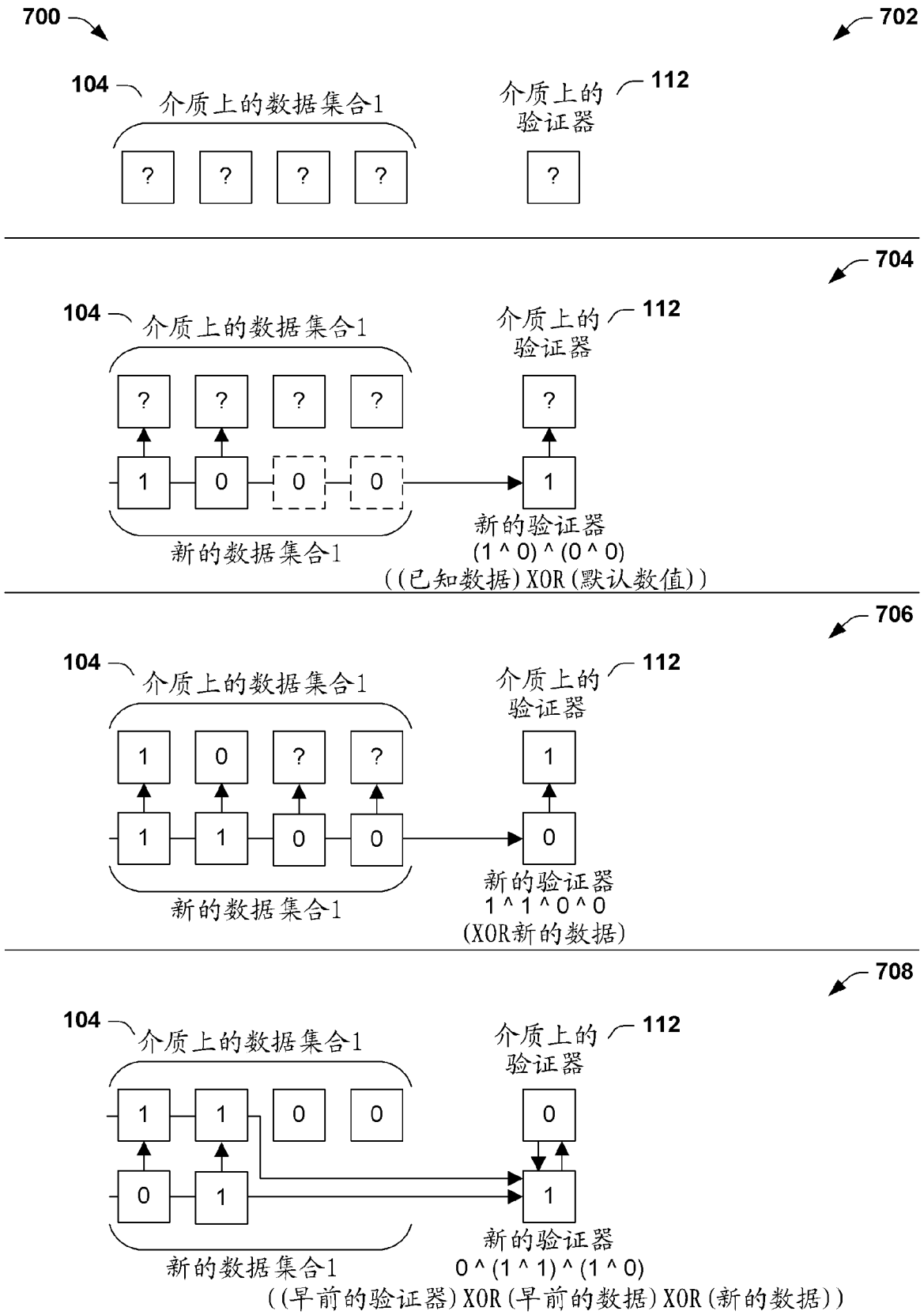

700
702
104
介质上的数据集合1
介质上的
验证器
112
? ? ? ?
?
704
104
介质上的数据集合1
介质上的
验证器
112
? ? ? ?
?
1 0 0 0
1
新的数据集合1
新的验证器
(1 ^ 0) ^ (0 ^ 0)
((已知数据) XOR (默认数值))
706
104
介质上的数据集合1
介质上的
验证器
112
1 0 ? ?
1
1 1 0 0
0
新的数据集合1
新的验证器
1 ^ 1 ^ 0 ^ 0
(XOR新的数据)
708
104
介质上的数据集合1
介质上的
验证器
112
1 1 0 0
0
0 1
1
新的数据集合1
新的验证器
0 ^ (1 ^ 1) ^ (1 ^ 0)
((早前的验证器) XOR (早前的数据) XOR (新的数据))

图 7

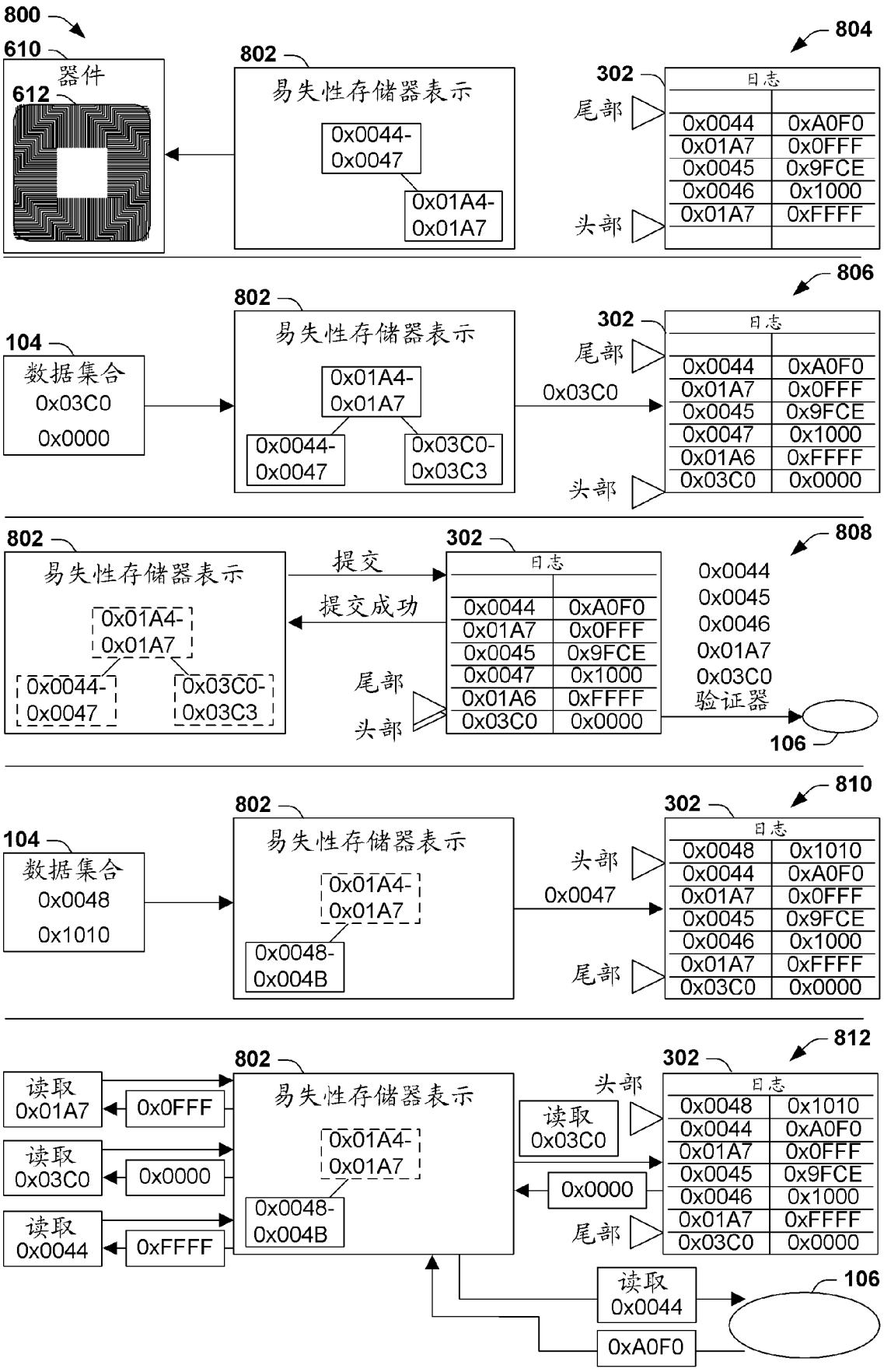

800
610
器件
612
802
易失性存储器表示
0x0044-
0x0047
0x01A4-
0x01A7
804
302
日志
尾部
头部
0x0044 0xA0F0
0x01A7 0x0FFF
0x0045 0x9FCE
0x0046 0x1000
0x01A7 0xFFFF
806
104
数据集合
0x03C0
0x0000
0x03C0
0x0047 0x1000
0x01A6 0xFFFF
0x03C0 0x0000
808
提交
提交成功
0x0044
0x0045
0x0046
0x01A7
0x03C0
验证器
106
810
0x0048 0x1010
0x0047
812
读取
0x01A7
0x0FFF
0x03C0
0x0000
0x0044
0xFFFF
0x0048-
0x004B
0x03C0
读取
0x0044
0xA0F0

图 8

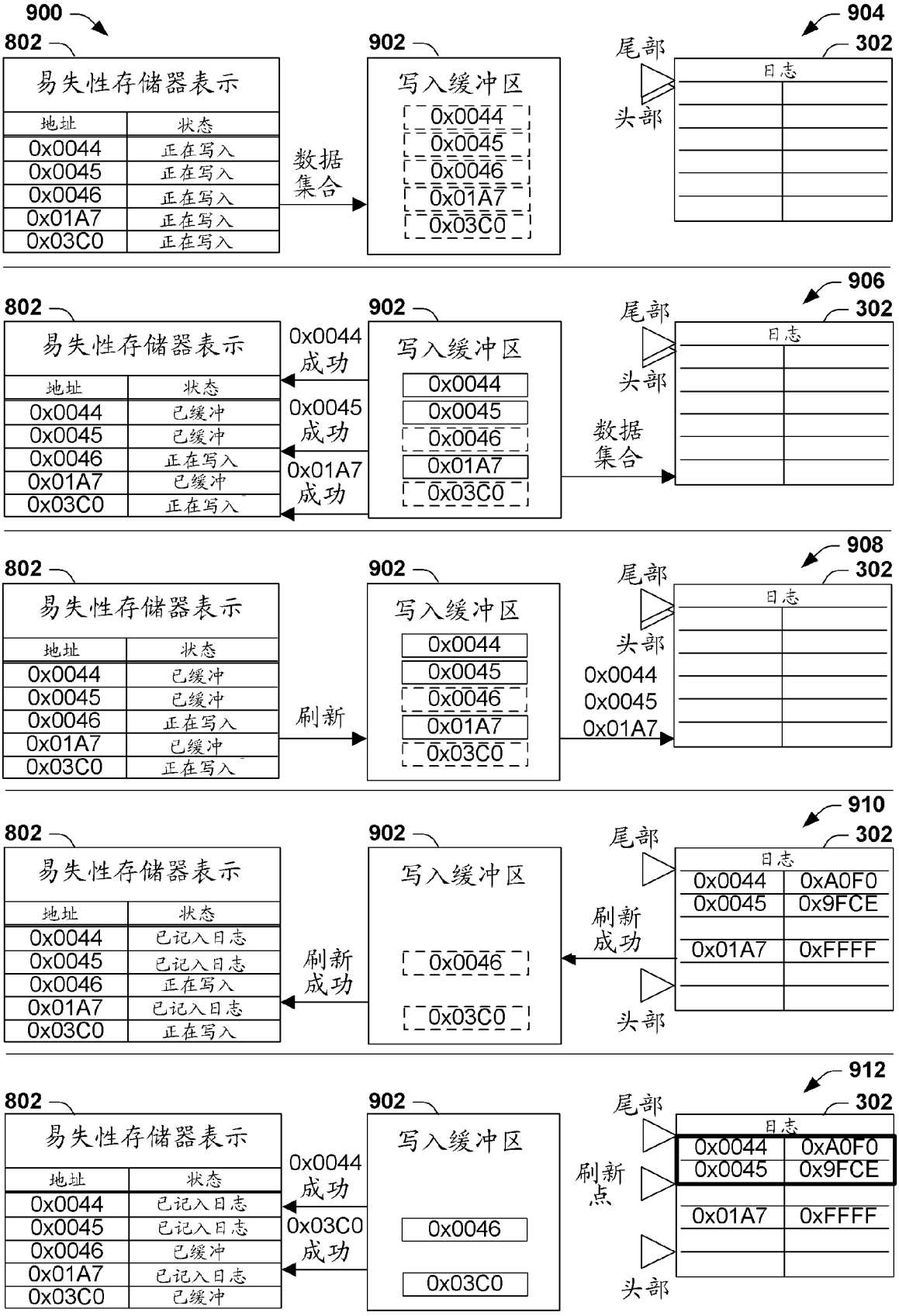

900
802
易失性存储器表示
地址 状态
0x0044 正在写入
0x0045 正在写入
0x0046 正在写入
0x01A7 正在写入
0x03C0 正在写入
数据集合
902
写入缓冲区
0x0044
0x0045
0x0046
0x01A7
0x03C0
904
尾部
头部
302
日志
906
0x0044 成功
0x0045 成功
0x01A7 成功
908
刷新
910
刷新成功
0x0044 0xA0F0
0x0045 0x9FCE
0x01A7 0xFFFF
912
刷新点

图 9

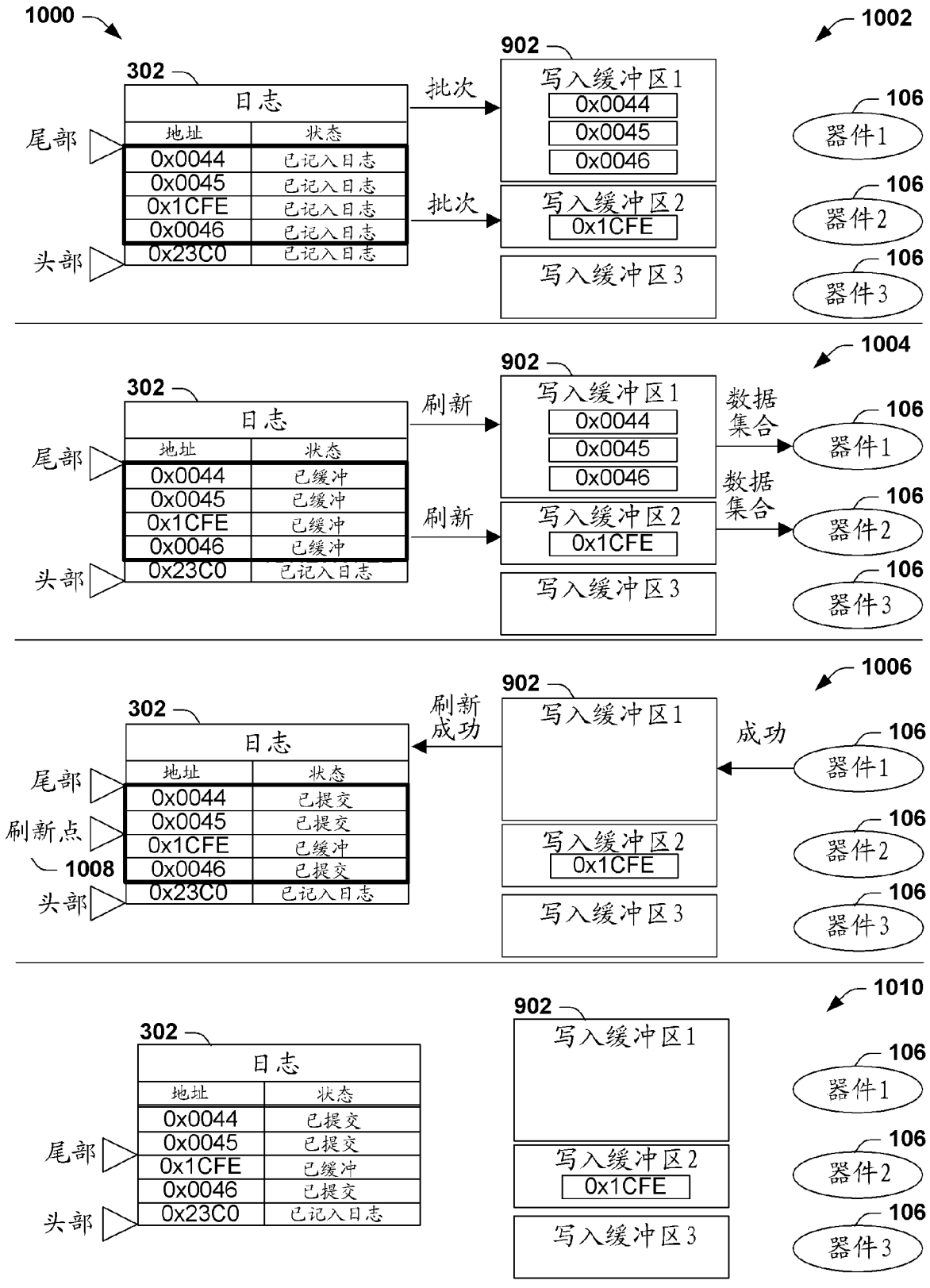

1000
1002
902
302
日志
地址
状态
0x0044 已记入日志
0x0045 已记入日志
0x1CFE 已记入日志
0x0046 已记入日志
0x23C0 已记入日志
尾部
头部
批次
批次
写入缓冲区1
0x0044
0x0045
0x0046
写入缓冲区2
0x1CFE
写入缓冲区3
106
器件1
器件2
器件3
1004
刷新
刷新
数据集合
数据集合
0x0044 已缓冲
0x0045 已缓冲
0x1CFE 已缓冲
0x0046 已缓冲
0x23C0 已记入日志
1006
刷新成功
成功
0x0044 已提交
0x0045 已提交
0x1CFE 已缓冲
0x0046 已提交
0x23C0 已记入日志
刷新点
1008
1010

图 10

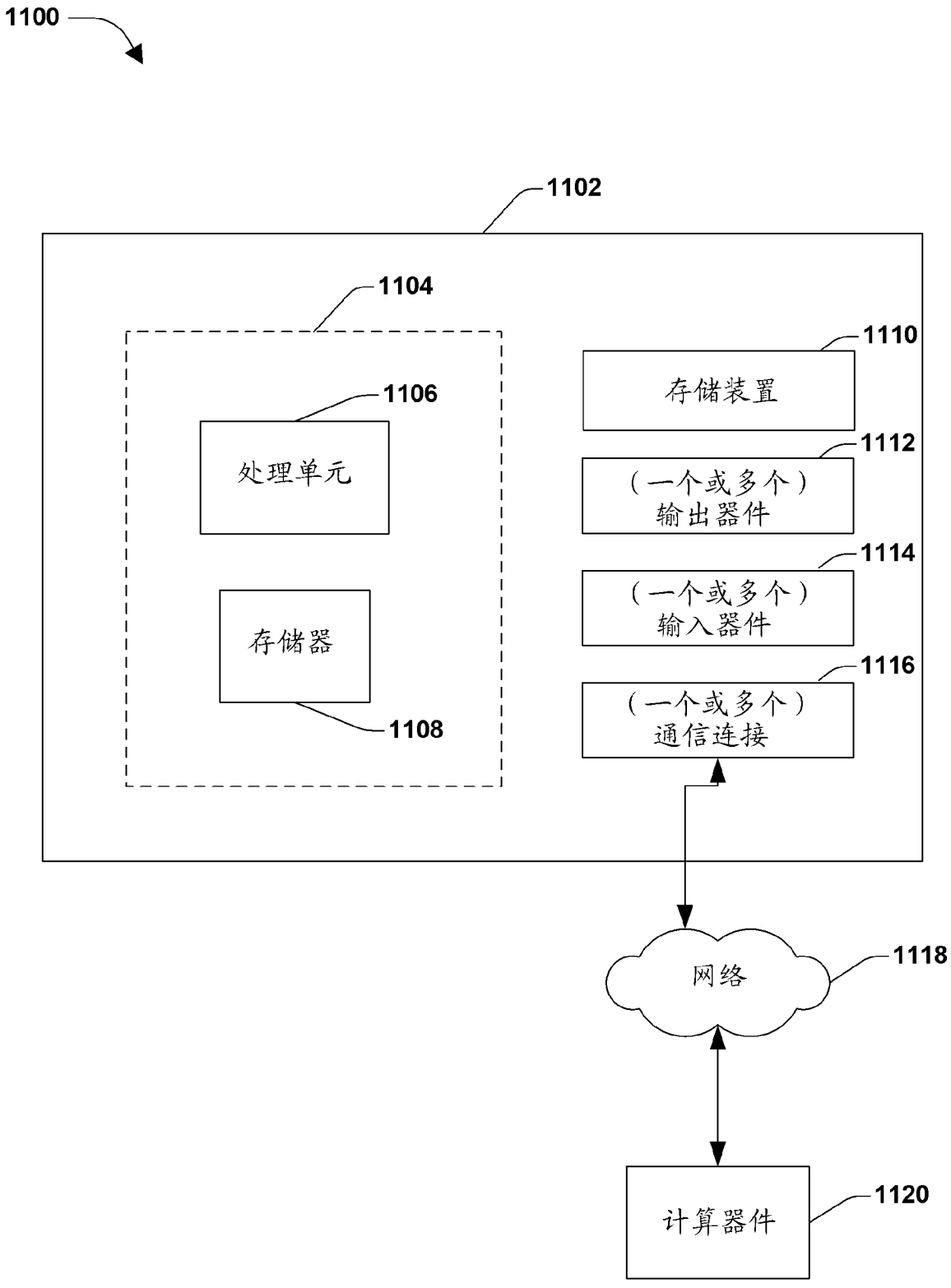

1100
1102
1104
1106
处理单元
存储器
1108
1110
存储装置
1112
（一个或多个）
输出器件
1114
（一个或多个）
输入器件
1116
（一个或多个）
通信连接
网络
1118
计算器件
1120

图 11