(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 103092782 A
(43)申请公布日 2013.05.08

(21)申请号 201210244914.3

(22)申请日 2012.07.13

(30)优先权数据
61/507,646 2011.07.14 US
13/432,394 2012.03.28 US

(71)申请人 LSI 公司
地址 美国加利福尼亚州

(72)发明人 维奈·阿肖克·苏曼纳切
蒂莫西·W·斯瓦托什
帕米拉·S·亨普斯特德
杰克逊·L·埃利斯
迈克尔·S·希肯 马丁·S·德尔

(74)专利代理机构 北京康信知识产权代理有限责任公司 11240
代理人 余刚 吴孟秋

(51)Int.Cl.
G06F 13/16(2006.01)

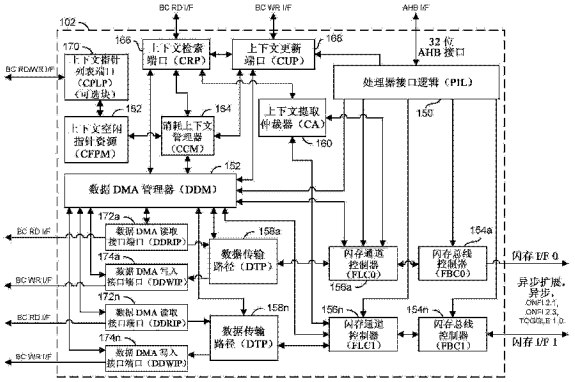
权利要求书1页 说明书20页 附图7页

(54)发明名称

用于闪存器件的闪存控制器硬件架构

(57)摘要

本发明公开了用于闪存器件的闪存控制器硬件架构，该闪存介质控制器包括一个或多个专用数据传输路径、一个或多个闪存通道控制器、和一个或多个闪存总线控制器。一个或多个闪存通道控制器通常耦接至一个或多个专用数据传输路径。一个或多个闪存总线控制器通常耦接至一个或多个闪存通道控制器。



CN 103092782 A

1. 一种闪存介质控制器，包括：

一个或多个专用数据传输路径；

一个或多个闪存通道控制器，耦接至所述一个或多个专用数据传输路径；以及

一个或多个闪存总线控制器，耦接至所述一个或多个闪存通道控制器。

2. 根据权利要求 1 所述的闪存介质控制器，其中，所述闪存介质控制器实现包含执行与闪存库的交易所需的信息的上下文。

3. 根据权利要求 2 所述的闪存介质控制器，其中，所述闪存介质控制器实现所述闪存库中每个逻辑单元的独立的上下文的链接列表。

4. 根据权利要求 1 所述的闪存介质控制器，其中，所述闪存介质控制器被配置为从多个上下文链中提取上下文。

5. 根据权利要求 1 所述的闪存介质控制器，还包括被配置为提供单个接口的消耗上下文管理器，在所述单个接口中，报告来自与所述闪存介质控制器相关联的所有逻辑单元的所有完成状态。

6. 根据权利要求 1 所述的闪存介质控制器，其中，所述一个或多个闪存通道控制器均包括：裸片管理表，保持附加到所述闪存通道控制器的各个裸片的提取上下文的当前状态；以及上下文管理器，被配置为对由所述闪存介质控制器执行上下文进行管理。

7. 根据权利要求 1 所述的闪存介质控制器，其中，在所述一个或多个闪存通道控制器的每一个中均支持全双工操作。

8. 根据权利要求 1 所述的闪存介质控制器，其中，所述闪存介质控制器实现多个闪存通道，每个闪存通道均具有包括一个相应的专用数据传输路径的独立通道结构。

9. 一种装置，包括：

至少两个专用数据传输路径；

至少两个闪存通道控制器，耦接至所述至少两个专用数据传输路径；以及

至少两个闪存总线控制器，耦接至所述至少两个闪存通道控制器。

10. 根据权利要求 9 所述的装置，还包括对应于至少两个独立通道被配置的多个闪存存储器件。

11. 根据权利要求 10 所述的装置，其中，所述装置包括片上系统。

12. 根据权利要求 9 所述的装置，其中每个闪存交易均由上下文表示。

13. 根据权利要求 12 所述的装置，其中，所述上下文包括一数据结构，该数据结构包含由所述装置执行与闪存库的交易或者将数据移动到系统缓冲器中的位置或从所述系统缓冲器中的位置移出所述数据所需的所有信息。

14. 根据权利要求 13 所述的装置，其中，所述数据结构进一步被配置为对附接至所述装置的每一闪存单元提供上下文的链接列表。

用于闪存器件的闪存控制器硬件架构

[0001] 本申请要求于 2011 年 7 月 14 日提交的美国临时申请第 61/507,646 号和 2012 年 3 月 28 日提交的美国申请第 13/432,394 号的优先权，其全部内容结合于此作为参考。

技术领域

[0002] 本发明总体涉及闪存介质，特别地，涉及用于实现闪存器件的闪存控制器硬件架构的方法和 / 或装置。

背景技术

[0003] 闪存介质控制器经由诸如 ONFI 2.X 的闪存接口与闪存器件通信。在单个闪存接口上，可以连接固定数量的闪存目标（target）。在单个闪存接口上连接多个闪存目标使得多个闪存目标之间共享闪存接口。在多个闪存目标之间共享接口会产生对闪存器件的瓶颈。

[0004] 期望实现一种用于闪存器件的闪存控制器硬件架构，其解决与闪存介质存储相关的挑战。

发明内容

[0005] 本发明涉及闪存介质控制器，其包括一个或多个专用数据传输路径、一个或多个闪存通道控制器和一个或多个闪存总线控制器。一个或多个闪存通道控制器通常耦接至一个或多个专用数据传输路径。一个或多个闪存总线控制器通常耦接至一个或多个闪存通道控制器。

[0006] 本发明的目标、特征和优点包括提供用于实现闪存器件的闪存控制器硬件架构的方法和 / 或装置，该闪存控制器硬件架构可以（i）为各个闪存交易提供上下文（context），（ii）提供上下文处理，（iii）通过消耗（consumed）上下文管理器（CCM）提供状态报告，（iv）在闪存通道控制器中提供裸片（die）管理表和上下文管理器块，（v）提供独立闪存通道结构，包括专用数据传输路径，（vi）在闪存通道控制器和缓冲控制器接口中提供全双工操作支持，和 / 或（vii）提供处理器控制模式。

附图说明

[0007] 从下面的详细说明书和所附权利要求书及附图中，上述和其他目的、特征和优点将变得显而易见，其中：

[0008] 图 1 是示出了在单芯片系统（SOC）环境（context）下实现的闪存介质控制器的框图；

[0009] 图 2 是示出了根据本发明实施方式的示例闪存介质控制器（FMC）结构的框图；

[0010] 图 3 是示出了根据本发明实施方式的示例闪存通道控制器结构的框图；

[0011] 图 4 是示出了图 3 的上下文管理器模块的示例子模块的示图；

[0012] 图 5 是示出了图 3 的裸片管理模块的示例子模块的示图；

[0013] 图 6 是示出了图 3 的闪存操作管理器模块的示例子模块的示图；

[0014] 图 7 是示出了图 3 的数据流管理器模块的示例子模块的示图；

[0015] 图 8 是示出了实现了图 3 的上下文管理器模块的示例子模块的示图；以及

[0016] 图 9 是示出了根据本发明实施方式的示例闪存介质上下文布局的示图。

具体实施方式

[0017] 在一个实施方式中，根据本发明的系统可以被设计为通过各种大容量存储协议进行操作，包括 SAS（“串行连接 SCSI”）、FC（“光纤通道”）和 FC-AL（“光纤通道仲裁环路”），所有这些都是基于小型计算机系统接口（“SCSI”）协议和串行 ATA（“SATA”）协议的。本领域普通技术人员应当熟悉这些大容量存储协议，因此，这样的协议不会在本文中进一步讨论。除非在调用特定协议的情况下，本文所公开的系统和方法不依赖于正在使用的特定协议，并被设计为通过所有协议进行正确操作。此外，根据本发明实施方式的系统和方法可以适用于与目前在使用或将来开发的其他类似协议一起使用，这些协议包括用于企业级应用的协议以及用于诸如最终用户的其他应用协议。本文所述的系统包括用于实现闪存器件的闪存控制器硬件架构的新方法和 / 或装置。

[0018] 参照图 1，其示出了通过根据本发明实施方式的闪存介质控制器所实现的系统 100 的框图。在一个示例中，系统（或结构）100 可包括块（或电路）102、多个块（或电路）104a 至 104n、多个块（或电路）106a 至 106n、块（或电路）18、块（或电路）110、块（或电路）112、块（或电路）114、和块（或电路）116。电路 102 至 116 可以表示被实现为硬件、固件、软件、硬件、固件和 / 或软件的组合或者其他的模块和 / 或块。

[0019] 在一个示例中，块 102 可以实现根据本发明实施方式的闪存介质控制器（FMC）。块 104a 至 104n 可以被实现为第一数量的闪存存储器件或组件。块 104a 至块 104n 可以耦接至块 102 的第一闪存通道。块 102 的第一闪存通道可以被配置为对各个块 104a 至 104n 提供独立的芯片启用（CE）信号。块 106a 至块 106n 可以被实现为第二数量的闪存存储器件或组件。块 106a 至块 106n 可耦接至块 102 的第二闪存通道。块 102 的第二闪存通道可以被配置为对各个块 106a 至块 106n 提供独立的芯片启用（CE）信号。尽管 FMC102 以两个闪存通道的示例进行了说明，对本领域技术人员显而易见的是，可以相应地实现另外的闪存通道以满足特定实现的设计标准。闪存器件 104a 至 104n 和 106a 至 106n 可以被实现为包括一个或多个裸片的单个闪存组（flash package）。闪存器件 104a 至 104n 和 106a 至 106n 通过使用 NAND 和 / 或 NOR 闪存器件来实现。块 102 可以包括用于 NAND 闪存和 / 或 NOR 闪存的适当的物理层支持（PHY）。

[0020] 块 108 可以实现可耦接至块 102 的外部 FMC 处理器（FARM）。块 110 可以实现可被配置为将静态随机存取存储器（SRAM）和 / 或动态随机存取存储器（DRAM）耦接至块 102 的存储器控制器。块 112 可以被实现为一个或多个 SRAM 器件。块 114 可以被实现为一个或多个 DRAM 装置。块 116 可以实现耦接块 110 和块 114 的双倍数据速率物理层（PHY）接口。在一个示例中，块 102、108、110、112、114 和 116 可以实现单芯片系统（SOC）结构。

[0021] 块 102 可以实现为被配置为协助各种应用程序使用闪存器件 104a 至 104n 和闪存器件 106a 至 106n 的软 IP 块。正如本文使用的，术语“软 IP 块”通常是指可以以软件（例如，HDL 代码、RTL 代码等）提供的集成电路的构建块（building block）。块 102 通常支持

与闪存器件的多个闪存接口。块 102 通常不包括处理器(例如 ARM)。然而,在一个示例中,块 102 可以实现被配置为将块 102 耦接至外部处理器 108 的接口(例如 32 位的 AHB 等)。块 102 通常被配置为处理由块 104a 至 104n 和块 106a 至 106n 形成的闪存介质海量存储阵列的管理。在一个示例中,块 102 可以利用多例示(multiply-instantiated)闪存通道控制器(FLC),其能够执行与附接有多个独立闪存器件的单个闪存数据通道相关联的大部分管理功能。从块 102 可能对闪存访问了解甚少这个意义上说,块 102 的功能可能有点宽泛。块 102 通常更多地涉及将闪存感知(flash-aware)通道编织(weave)成单个硬件实体。在一个示例中,实现块 102 的软 IP 可以进行参数化以支持用于应用程序的最大可能通道。例如,在一个实现中,通道的数量可以为 2。在另一实现中,数量可以为 8。

[0022] 在一个示例中,块 102 可以支持的特征包括 :(i) 两个闪存通道 ;(ii) 在每个闪存通道上的高达八个芯片启用信号(CE);(iii) 闪存接口,包括异步正常模式、异步扩展模式、Toggle1.0、ONFI2.3、ONFI2.1、Toggle2.0 ;(iv) 硬件可配置的多个通道之间的专用 ECC 或共享 ECC (例如实现块 102 的软 IP 包的参数化特征);(v) 闪存接口上的 8 位数据 ;(vi) Toggle2.0 或 ONFI2.3 的闪存接口规范的闪存接口上的高达 200MHz 的 DDR 速率 ;(vii) 部分读取命令,(viii) 随机读取命令 ;(ix) 关于闪存写入 / 读取的 CRC 删除 / 插入(strip/insert) 选项 ;(x) 对 4K 字节数据高达 64 位的校正 ;(xi) 512、2K、4K 字节数据上的可配置的 n 位校正(n 最大值 =64);(xii) 用于寄存器编程的 32 位 AHB 接口 ;(xiii) 上下文命令在外部存储器(如 DRAM 或 SRAM) 上的存储 ;(xiv) 在闪存通道控制器中的直通缓冲器(cut-through buffer);(xv) 提供更好性能的独立闪存读写数据路径 ;(xvi) 对各个闪存单元号(FUN)报告的有序状态 ;(xvii) 针对每个闪存通道的数据路径支持一个读取缓冲控制器(BC)接口和一个写入缓冲控制器接口 ;(xviii) 支持用于上下文检索的读取 BC 接口 ;(xix) 支持用于上下文更新的写入 BC 接口 ;(xx) 支持用于空闲资源指针(CFRP) 的读取 / 写入 BC 接口。

[0023] 参考图 2,图 2 示出了图 1 的块 102 的更详细的框图,该框图示出了根据本发明实施方式的示例闪存介质控制器(FMC) 结构。在一个示例中,块 102 可以实现三个主要功能接口 :缓冲控制器(BC)接口、闪存器件接口以及处理器接口(例如,32 位 AHB 等)。在框图的左侧和左上侧示出了缓冲控制器(BC)接口。在一个示例中,可以实现七个缓冲控制器接口(例如,三个读取接口 BC_RD_I/F、三个写入接口 BC_WR_I/F 和一个读取 / 写入接口 BC_RD/WR_I/F)。在框图的右侧示出了闪存器件接口。在一个示例中,可以实现两个闪存通道接口(例如,FLASH_I/F_0 和 FLASH_I/F_1)。在框图的右上侧示出了 32 位 AHB 接口。在一个示例中,32 位 AHB 接口可用于对块 102 中的寄存器进行编程、读取状态及使用诊断寄存器。

[0024] 块 102 通常包括块(或电路) 150、块(或电路) 152、多个块(或电路) 154a 至 154n、多个块(或电路) 156a 至 156n、多个块(或电路) 158a 至 158n、块(或电路) 160、块(或电路) 162、块(或电路)164、块(或电路)166、块(或电路)168、块(或电路)170、多个块(或电路)172a 至 172n 以及多个块(或电路) 174a 至 174n。电路 150 到 174a 至 174n 可以表示可实现为硬件、固件、软件、以及硬件、固件和 / 或软件的组合或者其他实施方式的模块和 / 或块。块 150 可以实现处理器接口逻辑(PIL)。块 152 可以实现数据 DMA 管理器(DDM)。块 154a 至 154n 可以实现闪存总线控制器(FBC)。块 156a 至 156n 可以实现闪存通道控制器(FLC)。块 158a 至 158n 可以实现数据传输路径(DTP)。块 160 可以实现上下文提取仲裁器(contexts

fetch arbiter，CA)。块 162 可以执行上下文空闲指针资源(CFPM)。块 164 可以实现消耗上下文管理器(CCM)。块 166 可以实现上下文检索端口(CRP)。块 168 可以实现上下文更新端口(CUP)。块 170 可以实现上下文指针列表端口(CPLP)。块 170 通常是可选的。块 172a 至 172n 可以实现数据 DMA 读取接口端口(DDRIP)。块 174a 至 174n 可以实现数据 DMA 写入接口端口(DDWIP)。并且，块 172a 至 172n 和块 174a 至 174n 通常形成数据 DMA 接口端口(DDIP)。

[0025] 在一个示例中，块 150 可以提供从块 108 至 102 的可寻址资源(例如，经由 AMBAAHB-Lite 接口)的接口。块 150 可以向所有可寻址资源提供接口并且向块 102 中的不位于块 156a 至 156n 内的子模块的配置和状态寄存器提供直接接口。块 150 也可以向各个块 156a 至 156n 中的可寻址资源提供接口。此外，块 150 可以包含一个上下文构成缓冲器(CCB)，其中，处理器固件可以将实际介质上下文写入块 102 以经由块 168 存储到系统缓冲器。在一个示例中，块 150 可以包括以下特征：至 108 的 32 位 AMBAAHB-Lite 从属接口；可以为输入时钟值(例如，HCLK)的一些划分值(或相同)的系统时钟(例如，SYS_CLK)；对块 102 中的所有配置寄存器和状态寄存器以及处理器可寻址空间的访问；由处理器固件用于构建存储在系统缓冲器中的上下文(context)的上下文构成缓冲器(CCB)；分布到块 156a 至 156n 中的每一个的处理器接口，其中，对各个寻址资源的访问由处理器访问端口(PAP)处理，并且其包含可被块 102 中的多个子模块使用的寄存器。块 150 可以对没有逻辑地存储在块 156a 至 156n 中的可寻址资源执行所有寄存器解码和所有读取数据复用(multiplex)。

[0026] 块 152 通常管理两个数据传输，一个用于闪存编程(例如，从缓冲器到闪存器件的数据交易)，另一个用于闪存读取(例如，从闪存器件到缓冲器的数据交易)。DMA 数据路径通常包括从块 156a 至 156n 到相应的块 158a 至 158n、数据 DMA 接口端口(DDIP)块 172a 至 172n 以及块 174a 至 174n 的分离的 32 位读写数据总线。块 158a 至 158n 可以包含 ECC 功能。DMA 数据传输通常包括一系列事件，该一序列事件可以包括通过块 102 的其他子块(或端口块)访问相应的上下文。在一个示例中，一个 DMA 传输可以包括 FLC 请求、检索上下文操作、数据传输和 FLC 完成阶段。

[0027] 在 FLC 的请求步骤中，数据传输可以随着块 156a 至 156n 中的一个升高各自的请求线而开始。在检索上下文操作中，可以经由上下文检索端口(CRP)接口 166 从一个缓冲控制器检索相应的上下文。数据传输可以发生在 DDIP、DTP 和 FLC 块之间，在该过程中，上下文可以发送至 DDIP 并且可以被写回或不被写回。在 FLC 完成阶段，到选定块 156a 至 156n 的完成线可以升高以表示传输结束。DDM152 可以执行检索上下文并向 DTP 块提供输入，以方便数据交易。

[0028] 块 154a 至 154n 通常在相应的闪存通道上执行至一组 NAND 闪存器件的低级接口信令。通常对各个闪存通道控制器(FLC)156a 至 156n 存在一个闪存总线控制器(FBC)154a 至 154n。块 154a 至 154n 通常管理用于几个接口类型的闪存接口协议的各个周期的时序以及给定类型的不同时序模式(例如，异步、ONFI2.0 同步、ONFI2.3 同步、三星 Toggle1.0、三星 Toggle2.0 等)。在一个示例中，通过一组内部时序寄存器中存储的时序计数，可以控制周期的定时。块 154a 至 154n 的核心逻辑通常在与其余的块 102 不同的时钟域下操作。一般地，只有时序寄存器组(set)位于与其余的块 156a 至 156n 相同的时钟域中。通常在寄存器和 FBC 核心之间通常不需要同步逻辑，因为只有在 FBC 是静态(quiescent)时(例如，

没有未完成(outstanding)操作)寄存器才被写入,所以寄存器被视为静态(static)的。

[0029] 块156a至156n通常执行对各个裸片的命令的调度。块156a至156n管理各个相应闪存通道上的命令序列。块156a至156n提供控制寄存器和状态寄存器,通过它们,固件能够对裸片(die)编程并观察状态。每个块156a至156n均包括上下文管理和裸片管理。块156a至156n通常负责处理上下文。

[0030] 块158a至158n的每一个路由数据流量,并针对块154a至154n中的每一个、可选的内部ECC编码器/解码器以及相应的数据DMA接口端口(DDIP)之间的数据流启动每个接口的流控制。在一个示例中,可在块158a至158n内实现内部ECC编码器/解码器。可选地,块158a至158n中的每一个可被配置为共享单个ECC编码器/解码器模块。可以通过相应的数据DMA管理器(DDM)模块152和相应的数据DMA接口端口(DDIP)块172a至172n和块174a至174n针对每次传送对块158a至158n进行编程。各个块158a至158n可以包括可在全双工操作模式下工作的独立闪存读写路径。块158a至158n保持数据传输中的当前区域计数以及各个区域中的当前双字(dword)计数。块158a至158n通常执行DDIP、ECC编码器及解码器和FLC块之间的流控制转化。块158a至158n保持每次传输的运行可校正(running correctable)ECC错误和,并在传输结束后向块152提供该最终值。块158a至158n可以包含用于对ECC编码器及解码器编程的FMC寄存器。块150可以通过寄存器接口来访问寄存器。ECC模块通常能够进行4K字节数据上的64位校正。然而,可以相应地实现其他级别的校正以满足特定实现的设计标准。在一个示例中,解码器门计数可以为415K门,而编码器门计数可以为75K门。

[0031] 块160通常负责从块156a至156n接收对上下文的请求,从系统缓冲器(例如,通过缓冲控制器进行访问的DRAM)中检索请求的上下文,然后将上下文分发给块156a至156n。检索实际上可以经由对上下文检索访问端口(CRP)166的请求来执行。上下文为FMC的基本控制单元。上下文通常包含FLC执行命令所需的、或者FMC执行到系统缓冲器或来自系统缓冲器的相关数据传输(DMA)所需的所有信息。FLC的动作完全自主;因此,FLC需要仲裁经由缓冲控制器对系统缓冲器的访问,其包含由固件构建的上下文的链接列表(linked list of context)。块160通常提供仲裁,以及发起对块166的请求。然后,块160将检索到的上下文清楚地路由到相应的FLC目的地。块162通常被实现为块102的子块以提供空闲指针在其中对固件可用的单点(single point)。

[0032] 块164通常被实现为块102的子块以提供由固件可以在完成后检测已完成的上下文的单点。块164通常在多个FLC源之间执行仲裁。FLC提供与上下文指针相关联的PASS/FAILE CC状态。一旦获得上下文,块164更新上下文状态字段,然后将上下文提供给固件。在固件需要较长时间来读取已完成的上下文、以及块164内的内部存储器将变满的情况下,块164可以使用缓冲器来存储在当前报告的上下文之后进行排队的已完成的上下文。

[0033] 块166至174n通常实现端口接口。端口接口可用于与缓冲控制器通信。在一个示例中,在端口接口内可以实现QBFIFO块。下面的端口接口也可以被实现为端口接口的一部分:上下文检索端口(CRP)166、上下文更新端口(CUP)168、上下文指针列表接口端口(CPLIP)170(可选)、数据DMA读取接口端口(DDRIP)172a至172n、数据DMA写入接口端口(DDWIP)174a至174n。在一个示例中,块102的信号接口可被分为4个主要接口:AHB接口、

缓冲控制器接口、NAND 和 / 或 NOR 闪存物理层(PHY)接口以及混杂(MISC)接口。缓冲控制器接口可以包括(i)用于通道 0 和通道 1 的 DDIP BC 写入接口,(ii)用于通道 0 和通道 1 的 DDIP BC 读取接口(iii)CRP BC 读取接口,(iv)CUP BC 写入接口,以及(v)CPLIP BC 读取 / 写入接口。

[0034] 在一个示例中,块 102 可以以三个时钟实现。块 102 中的大部分逻辑可以在被称为系统时钟(例如,SYS_CLK)的时钟域上工作。系统时钟可以为 AHB 时钟。系统时钟通常具有 FMC 处理器(FARM) 112 的工作频率的一半的频率。第二时钟可以为所谓的闪存时钟(例如,FBC_CLK)。闪存总线控制器(FBC)154A 至 154n 可以完全工作在闪存时钟域。在一个示例中,可在块 154a 至 154n 的数据流管理器(DM) 模块中实现先进先出缓冲器(FIFO),以管理时钟 FBC_CLK 和 SYS_CLK 之间的频率。第三时钟可以为缓冲控制器时钟(例如,BC_CLK)。所有带有 BC 的接口端口均工作在缓冲控制器时钟域。可以在缓冲控制器时钟 BC_CLK 和系统时钟 SYS_CLK 之间实现缓冲器元件(例如,QBFIFO)。

[0035] 参考图 3,其示出了图解根据本发明实施方式的示例闪存通道控制器结构的块 200 的示图。在一个示例中,块 200 可以用于实现图 2 中的块 154a 至 154n 和块 156a 至 156n。在一个示例中,块(或电路) 200 可以包括块(或电路) 202、块(或电路) 204、块(或电路) 206、块(或电路) 208、块(或电路) 210、块(或电路) 212 以及块(或电路) 214。电路 202 至 210 可以表示被实现为硬件、固件、软件、以及硬件和固件和 / 或软件的组合或其他实施方式的模块和 / 或块。在一个示例中,块 202 可以实现上下文处理协调器(CPC)。在一个示例中,块 204 可以实现上下文管理器(CM)。在一个示例中,块 206 可以实现裸片管理模块(DMM)。在一个示例中,块 208 可以实现闪存操作管理器(FOM)。在一个示例中,块 210 可以实现处理器访问端口(PAP)。在一个示例中,块 212 可以实现闪存总线控制器(FBC)。在一个示例中,块 214 可以实现数据流管理器(DFM)。

[0036] 块 202 可以帮助上下文信息流入和流出块 200。上下文流可以由块 204 发起。块 202 主要涉及响应于提取或配置上下文的请求。为了获取上下文,块 202 响应块 204 对新的上下文的请求。首先,块 202 可以向在块 200 管理的裸片中进行仲裁并将选定的裸片或逻辑单元号(LUN) 转发至块 202 的块 206 发起请求。然后,块 202 发出一个试图从系统缓冲器提取上下文的提取至上下文提取仲裁器(CFA) (例如,图 2 中的块 160)。

[0037] 一旦提取到上下文,将上下文提交给块 202。块 202 对上下文执行一些解释并转发上下文至块 204。如果块 206 没有可用来发起上下文执行的裸片(LUN),则块 206 通知块 202 缺乏可用的裸片,然后块 202 通知(communicate)块 204 缺乏可用的裸片。块 202 还协助块 200 完成上下文的配置。此外,启动该流程的是块 204,并且,向实现消耗上下文管理器(CCM)的块(例如,图 2 中块 164)发出配置消息的是块 202。当由 CCM 接收到配置消息并启动工作时,块 202 通知块 204,然后块 204 可以继续处理上下文的执行。

[0038] 块 202 通常执行上下文的一些解释。具体地,为了确定上下文是否为处理器控制模式(PCM)上下文,块 202 可以解释上下文。当接收到 PCM 上下文时,上下文的提取(追加)应当停止。然后,块 202 等待块 204 开始执行 PCM 上下文并且在处理器控制模式完成时继续(resume)“标准”操作。在处理器控制模式间隔期间,块 202 确定提取到的上下文是否是 15 个双字上下文,而不是 4 个双字闪存上下文,块 202 在“标准”操作下将其发送至块 204。

[0039] 在一个示例中,块 204 可以包括上下文状态机(CSM)、上下文提取管理器(CFM)、上

下文配置引擎(CDE)、上下文解释器(CI)。块204通常负责管理正在由块200有效处理的上下文。块204通常执行有效上下文的簿记(bookkeeping)。上下文是向系统缓冲器提供闪存介质控制器(FMC)执行闪存交易和DMA所需的所有信息的数据结构。块204管理在闪存通道控制器级别上的上下文,由此主要涉及闪存交易有关的上下文管理。块204维持由块208执行对闪存通道上的闪存裸片执行命令和数据传输所用的信息。

[0040] 块206通常负责维持块200的操作所需的基于裸片的信息。块206管理裸片管理表中的各个裸片的信息并且在裸片之间仲裁对上下文表排队的访问。在一个示例中,块206可以包括更新裸片状态的裸片状态机。块206可以执行/监控多裸片操作。块206通常负责闪存命令,包括但不仅限于:READ、COPYBACK READ/COPYBACK WRITE、BLOCK ERASE、PAGE PROGRAM,以及目标级命令,其中,目标级命令包括但不仅限于READ ID、READ PARAMETER PAGE、GET FEATURES、SET FEATURES、SYNCHRONOUS RESET、以及RESET。

[0041] 块208通常处理应用到闪存通道的各个闪存操作序列。通常针对闪存介质控制器的每个闪存通道控制器(FLC)来实现一个块208。块208在块204的上下文表中的命令之间进行仲裁,并应用命令至块212。在一个示例中,块208本身支持来自ONFI2.0命令列表中的最常用的命令,以及在三星NAND闪存器件中出现的一些特定的(和类似的)命令。此外,可以通过纳米序列器(nano-sequencer)(在下面图9到图11中详细说明)支持其他现有和将来的命令。自身支持的命令可以无需处理器干预来操作,但其他命令通常使用一些级别的处理器支持。

[0042] 闪存命令可以被分解成可串行地应用于由块208控制的实际闪存裸片的原子“周期”。因为闪存命令通常涉及漫长的等待时间(例如,在数据从芯片中有效读取之前,页读取可能需要25μs),“命令周期”可以经常针对闪存通道的不同裸片“一个接一个”地(back-to-back)操作,从而减少了有效的、累积的等待时间。块208通常通过随着应用每个闪存循环而更新裸片的状态来管理闪存裸片。然后块208读取更新的上下文表来确定接下来什么循环应该(或可以)执行。NAND闪存存储器操作通常包括一个或多个闪存周期。通常存在四种类型的闪存周期:命令、地址、数据输出(w.r.t闪存器件,例如,读取)、数据输入(w.r.t闪存器件,例如,写入)。周期类型大致转化为在块208和块212之间定义的操作类型。

[0043] 块210通常实现提供了从FMC 100的AHB-Lite从属接口至200的可寻址资源的处理器访问的接口块。在本文讨论的大部分资源主要用于诊断目的,正如所有配置信号采用全局级别(作为共享的配置寄存器块的一部分)。例如,可以通过块210实现闪存通道数据缓冲器的完全访问。访问可以单纯作为早期验证支架(scaffold)而提供。然而,对闪存通道数据缓冲器的访问,也可以支持需要直接访问内部表的固件补丁(pitch)。这样的访问可以通过块210提供。

[0044] 块210的特点可以包括:简单的访问接口,符合AHB-Lite从属协议并且由FMC中的处理器接口逻辑(PIL)进行缓冲;提供对寄存器资源、上下文表、上下文高速缓存以及裸片管理表的读写访问;提供对位于块214中的闪存通道数据缓冲器存储器资源的读写访问。块210通常支持添加每通道配置寄存器的能力,尽管大部分配置寄存器通常被提供为块200的输入。类似地,可以支持状态寄存器和中断寄存器访问,尽管大多数状态寄存器和中断寄存器通常在块200以外产生。块210的主要逻辑组可以包括:界面管理器(IF_MGR)、

数据流管理接口(DM_IF)、寄存器块解码器(REG_DEC)、寄存器块复用器(REG_MUX)、中断处理器(INT_HND)和FLC全局寄存器(GLOB_REGS)。

[0045] 参照图4,示出了图解图3的上下文管理模块204的子模块的示图。在一个示例中,块204可以包括上下文表(CT)220、上下文状态机(CSM)222、上下文高速缓存(CC)224和上下文队列控制器(CQC)226。块204通常划分和执行闪存通道控制器上的操作的阶段,维持闪存通道上的所有有效上下文的优先级顺序,维持各个闪存通道上下文的状态,提供(例如,通过上下文高速缓存)执行完整交易所需的上下文的临时片上存储的最低量,维持正在执行的处理中的各个上下文的缓冲器指针,并通过使用上下文状态机(CSM)222确定下一状态的上下文以为各个上下文提供代理(agency)。可以在上下文表(CT)220中保持最小的上下文信息。表220通常提供了目前正在执行的上下文的优先级队列。上下文队列控制器(CQC)226可被配置为从上下文表220中移除已完成的上下文并压缩上下文表220,以消除间隙。

[0046] 参照图5,示出了图解图3的裸片管理模块206的子模块的示图。在一个示例中、块206可以包括裸片状态机230、裸片服务仲裁器232以及裸片管理表234。

[0047] 参照图6,示出了图解图3的闪存操作管理器(FOM)208的子模块的示图。在一个示例中,块208可被分为四个子模块:命令仲裁器(CA)240、数据传输仲裁器(DTA)242、闪存操作格式化器(FOF)244、纳米序列器246。命令仲裁器240通常扫描上下文表以得到要应用的命令,然后与闪存操作格式化器(FOF)244通信以发送信号至闪存缓冲控制器(FBC)。一旦所有的“命令”部分已经运行,闪存为“数据阶段”准备就绪,数据传输仲裁器242发起FBC和数据流管理(DM)214之间的传输。最后,纳米序列器246解释特殊的“软上下文”以应用任何闪存可能需要的命令序列,即使命令序列本身不支持。

[0048] 参照图7,示出了图解图3的数据流管理子模块214的示图。数据流管理214通常提供闪存通道数据缓冲器存储器资源。在一个示例中,闪存通道数据缓冲器存储器资源可以包括直通缓冲器250和252。在一个示例中,直通缓冲器250和252可以被实现为其大小是可编程的。例如,缓冲器250和252的大小可以调整,以满足带宽指标。在一个示例中,缓冲器250和252可以包括静态随机存取存储器(SRAM)。然而,可以相应地实现其他类型的存储器以满足特定实施的设计标准。通常地,各个闪存通道实现两个直通缓冲器。

[0049] 参照图8,示出了图3的上下文管理器(CM)204的示例实施的示图。上下文管理器(CM)204通常负责管理相应的闪存通道控制器(FLC)正在有效地处理的上下文。CM 204通常执行有效上下文的簿记。如前所述,上下文是向系统缓冲器提供闪存介质控制器(FMC)102执行闪存交易和DMA所使用的所有信息的数据结构。CM 204管理FLC级别处的上下文,由此主要涉及与闪存交易有关的上下文管理。CM 204维持闪存操作管理器(FOM)对闪存通道上的闪存裸片执行命令和数据传输所用的信息。

[0050] CM 204通常被配置为:(i)发起并执行各自的闪存通道控制器上的操作的阶段,(ii)维持相应的闪存通道上的所有有效上下文的优先级顺序,(iii)维持相应的闪存通道上的各上下文的状态,(iv)提供用于执行交易的上下文的临时片上存储的最小量(或将该量最小化)(例如,通过上下文高速缓存224),(v)维持正在执行的处理中的各上下文的缓冲器指针,(vi)通过使用上下文状态机(CSM)222确定下一状态的上下文以为各上下文提供代理,(vii)维持目前正在执行的上下文的优先级队列(例如,上下文表220)中的最小的

上下文信息。上下文队列控制器 226 通常被配置为从上下文表 220 中移除已完成的上下文并且压缩上下文表 220，以消除间隙。

[0051] 上下文队列控制器（CQC）226 是对上下文表（CT）220 执行修改的逻辑块。在一个示例中，CT220 可以被实现为寄存器块，其针对每个排队上下文被组织成一个实体。CQC 226 是对按照优先级队列组织的表执行操作的块。CQC 226 通常启动和执行上下文处理，并负责对上下文表上执行处理。主要处理通常包括追加、等待、修改、配置和压缩。处理由 CQC 226 进行发起和执行。

[0052] 追加阶段是 FMC 提取新上下文并将这些上下文的条目添加至上下文表 220 的阶段。CQC 226 检查闪存上下文的内容以及由 CPC 202 提供的上下文信息，并且基于内容和上下文信息来追加和创建条目。在一个示例中、上下文表的条目可以包括表示上下文表的条目是否有效的位（或标志）、表示上下文的状态的值、表示上下文高速缓存索引的值、表示闪存操作值、表示闪存裸片的值、上下文指针、指示是否禁用数据传输的位（或标志）以及表示平面地址的值。新条目通常以“有效”位设置（例如，为“1”）以及被设置为“QUEUED”的“上下文状态”的值开始。如果闪存操作是非法的，初始状态可以被设置为值“ILLEGAL”，并且该上下文条目会在处理阶段被删除。通常，通过 CQC 226 所提供的上下文和信息来确定其他字段。新的条目通常都附加到压缩的上下文表 220 的末尾。因此，CQC226 通常知道 220 上下文表的深度。

[0053] 当 CQC 226 不再等待未完成的数据传输完成并且 CQC 226 在给定的闪存操作周期内已经试图进行至少一次追加操作的时候，CQC 226 通常退出“追加”阶段。当上下文表 220 或上下文高速缓存 224 中不再有任何有效空间时，CQC 226 还可以离开“追加”阶段。

[0054] 上下文管理器 204 在全闪存操作周期之间可以或可以不被强制等待。上下文管理器 204 通常能够强制（enforce）最低闪存操作时期（例如，通过闪存操作周期寄存器）。例如，在闪存里除了编程或擦除命令之后的轮询以外基本上是闲置的情况下，这样的最小周期是所期待的。在这种情况下，因为没有追加或配置，所以上下文阶段需要很短的时间来执行。因此，通道倾向于处于这样的状态，其中，通道为忙碌的连续轮询闪存裸片（continuously polling flash die），从而在不应该有能量消耗时在闪存接口上消耗能量。CQC226 通常停留在等待阶段，直到预定的时间已经过期（例如，时间可在“闪存操作计时器”寄存器中指定）。当预定的时间已过期时，CQC 226 可以进入“修改”阶段。

[0055] CQC 发起的下一阶段通常为“修改”阶段。在修改阶段，上下文表 220 基于由闪存操作管理器（FOM）执行的闪存操作以及通过来自数据路径传输的结果而被修改。更新通常涉及到上下文的状态并因此通常通过上下文状态机（CSM）222 发起。当状态更新发生时，CSM 222 发送更新状态和上下文表索引至 CQC 226。然后，CQC 226 更新上下文表 220 中的条目。当 FOM 结束其闪存接口处理周期时，修改阶段结束。FOM 可通过发出一个信号（例如，FOM_CM_FLASH_PROC_CMPLT）通知上下文管理器 204 闪存接口处理已完成。一旦修改阶段完成，CQC 226 可以对上下文表 220 上的上下文执行配置、压缩和追加。在此期间，上下文表 220 对 FOM 是不可访问的。CQC 226 可通过在特定的时钟周期内解除（de-assert）向 FOM 表示上下文表读取条目和上下文高速缓存读取数据有效的信号（例如，CM_FOM_CT_VALID）的方式，使上下文表 220 对 FOM 是不可访问的。

[0056] 当修改阶段完成后，CPC 202 发起“配置”动作。配置动作将 CQC 226 置入 CQC 226

搜寻检索已完成执行的条目的上下文表 220 的模式。CQC226 将条目是否已完成执行的确定建立在上下文状态上。在上下文处于“完成”状态时，上下文可由 CQC 226 配置。在一个示例中，上下文可以处于这样的状态，其中 CQC 226 正在等待来自数据路径的有关上下文完成状态的通知。例如，在读取操作的情况下，上下文可以处于 DATA_TRANSFER_DONE 状态中，并等待 ECC 校验的结果。在这种情况下，CQC 226 可以临时暂停配置处理并等待从数据路径返回的状态。在此期间，CQC 226 允许“追加”发生。然而，一旦等待状态返回，上下文可由 CQC 226 配置，并且消耗的上下文记录可以转发给 CPC 202（并最终到消耗上下文管理器（CCM）164）。

[0057] 当 CQC 226 配置了上下文，CQC 226 清除上下文表 220 相应的条目的“有效”位。这个过程一直持续到 CQC 226 已审阅了上下文表 220 中的每个上下文。当 CQC 226 到达上下文表 220 中的有效上下文的末尾时，配置阶段完成。

[0058] 已由 CQC 226 配置的上下文清除各自的表条目中的“有效”位。在没有一个机制来移动该表以填满空位（hole）的情况下，有效条目会在上下文表 220 中变得分散（或分段）。分散的上下文将使得上下文表难以扫描以及“追加”阶段更为复杂。为了确保上下文表 220 维持其优先级队列的特征，上下文表 220 可以被压缩。在压缩过程中，当 CQC 226 配置了上下文，CQC 226 立即将被释放的条目之后的所有条目上移一个位置。当该处理完成后，所有有效条目都在优先级顺序列表的前面并且所有的空位被移除。与其他动作的情况一样，当压缩过程完成时，CQC 226 维护“完成”信号（或位）。在最后压缩阶段结束之后，CQC 226 可以开始追加阶段。

[0059] CQC 226 通常知道处理器的控制模式。在处理器控制模式中，整个 CM 204 暂停标准操作并且进入这样的模式，其中，FLC 的操作本质上被闪存操作管理器 208 中的纳米序列器 246 执行的“软上下文”所驱动。软上下文具有与标准闪存上下文不同的尺寸。在一个示例中，软上下文可以包括全部的十五个 32 位双字，然而“闪存上下文”，即 FLC 执行的全部介质上下文的部分，通常仅包括四个 32 位双字。

[0060] 通常，当处理器控制模式（PCM）的“闪存操作”字段被设置为 PROCESSOR_CONTROL_MODE 的上下文出现在上下文队列的顶部时，处理器控制模式（PCM）开始。通常，在上下文表 220 中，在 PCM 上下文后面应当没有有效条目，因为一旦 CQC 226 对 PCM 上下文排队，CQC226 应当暂停标准上下文的检索。当 PCM 开始时，CQC 226 可以经由信号（例如 CM_CPC PROC_CNTL_MODE）通知 CPC 202。响应于该通知，CPC 202 可以取得在 PCM 上下文中的位置出现的“软上下文”。从提供给 FOM 的角度来看，FOM 通常不知道在上下文表 220 中存在 PCM 上下文，而 PCM 上下文是在上下文表 220 中的其他有效条目的后面。上下文表 220 中的 PCM 上下文条目将其“有效”位作为 0 提供给 FOM，直到 CM 204 为 FOM 做好开始执行软上下文的准备。

[0061] 当 FOM 开始读取软上下文时，因为操作由存储软上下文的上下文高速缓存 224 提供给 FOM 208，CQC 226 监视（snoop）操作。当操作涉及 DMA 上下文（例如预取数据，设置读取数据缓冲器，或配置上下文指针）时，CQC 226 在上下文表 220 中指定（co-opt）上下文表中现在未使用的存储，并在上下文表中放置指针以用于跟踪。在这些 DMA 上下文完成时，FOM 208 通知上下文管理器 204，其之后用标准方式配置上下文。

[0062] CQC 226 在监视的同时，还寻找“取得下一软上下文”操作。当 CQC226 发现一个

时，CQC 226 维持至取得下一软上下文的 CPC 202 的信号(例如 CM_CQCPCM_NEXT_CONTEXT)。在 FOM 208 通知 CM 204 软上下文执行已完成时，FOM 208 在“FOM/CM”命令接口上通知 CM 204。然后，CQC 226 解除到 CPC 的信号(例如 CM_CPC PROC_CNTL_MODE)，然后，标准操作继续。在一个示例中，CM_CPC PROC_CNTL_MODE 可以被维持为表示 CM 204 已经进入处理器控制模式并且现在准备好接收软上下文的信号的级别。

[0063] CQC 226 的另一重要功能为监控超时情况。在一个示例中，CQC 226 可以包括计数器，其被配置为对系统时钟(SYS_CLK)周期的数量进行计数，使得同样的上下文表的条目位于上下文表 220 的顶部(即，在条目 0)。如果计数值达到一个可编程“超时”计数器的值，上下文表 220 顶端的条目可以被认为已经超时。当条目被认为已经超时时，条目可以从上下文表 220 中移除，上下文指针返回到消耗上下文接口上的上下文处理协调器(CPC) 202。

[0064] 上下文的返回状态是两个可能的“超时”状态之一。在第一种情况中，超时会可能起因于这样的情形，其中，闪存通道上的另一裸片忙碌并且正在降低 R/B 线。在这种情况下，状态表示超时可能是起因于另一裸片的超时。在第二种情况中，上下文的裸片被视为事故原因(culprit)。在这里，可以返回表示裸片是事故原因的不同状态。

[0065] 上下文表 220 本质上为条目的存储介质。上下文表的深度是参数化的。例如，在能支持每通道十六个裸片)的芯片的情况下，可以实现十六个条目。如果每个裸片(die)可以管理一个以上的操作，增加深度可以是有利的。上下文表 220 有最小函数(minimal function)。CQC 226 对上下文表 220 实现大部分更深入的处理。然而，可以通过多重读取接口以及各个读取接口的复用逻辑来实施上下文表 220。在一个示例中，针对读取访问能力，可以通过 FOM 208 的接口以及到上下文状态机(CSM) 222 的接口来实施上下文表 220。上下文表 220 也可以具有到 CQC 226 的读取接口。上下文表 220 也可以被处理器访问。

[0066] 上下文表 220 还具有用于表的压缩阶段的“移动(shift)”能力。除此之外，CQC 226 可以使用简单的写入接口来更新上下文表 220。在一个示例中，上下文表 220 可以在触发器(flip-flop)中实施。当上下文表 220 在触发器中实施时，不需要读取访问所需的仲裁。如果上下文表 220 的规模增加到超过大约 1000 个触发器，上下文表 220 可以在寄存器堆或 SRAM 中实现，但还应当实现额外的管理和访问仲裁。

[0067] 上下文高速缓存 224 是类似于上下文表 220 的另一上下文数据存储元件。上下文高速缓存 224 通常包括参数化的条目数。在一个示例中，条目的数量可以为八个。然而，可以实现其他数量的条目以达到特定实施方式的设计标准。例如，条目的数量可以被设置为比实际上是全流水线操作所需的数量多一个或两个。数量通常应当被设置得足够大，以允许在处理器控制模式中用于全“软上下文”的足够空间。如上所述，一个完全的上下文可以包括十五个 32 位双字。全部介质上下文的子集称为“闪存上下文”。闪存上下文通常是全部介质上下文的前四个双字(或 DWORD/ 双字)。闪存上下文的四个双字通常包含由 FLC 用来执行由固件指定的完整操作的所有信息。在标准操作(例如，当 FLC 不处于处理器控制模式)中，只有闪存上下文的前两个双字存储在上下文高速缓存 224。闪存上下文的其余部分一般存储在上下文表 220 中。

[0068] 上下文高速缓存 224 通常维持各个条目上的状态。在一个示例中，状态可以包括表示条目是空闲(FREE)还是已用(USED)的位。在一个示例中，可以在上下文高速缓存 224 中实现 8 个这样的位。当闪存上下文被写入到上下文高速缓存 224 的位置时，位置的状态

变为 USED。当 CQC 226 在状态变换时接收到使位置清空的信息时，位置状态返回到 FREE。在标准操作期间，上下文高速缓存 224 基于状态位通知 CQC 226 上下文高速缓存 224 具有空闲条目的空间。如果存在空闲的位置，CQC 226 可自由地从 CPC 202 请求上下文。在 CPC 202 取得新的闪存上下文时，CPC 202 将闪存上下文作为一串（a burst）32 位双字数据提供给上下文高速缓存数据 224。当数据有效时，可以维持信号（例如 CPC CM_ENQ_CTX_VALID）。上下文高速缓存 224 将数据写入空闲的位置。上下文高速缓存 224 预期 CPC 202 将仅写入一个闪存上下文。

[0069] 在上下文表 220 顶端处的条目被表示为 PROCESSOR_CONTROL_MODE 操作的处理器控制模式中，上下文高速缓存 224 应当是完全空闲的。在处理器控制模式中，上下文高速缓存 224 应当预期从 CPC 202 接收软上下文。上下文高速缓存 224 也可以预期软上下文包括 15 个双字。实质上，上下文高速缓存 224 用作从属，接收由 CPC202 提供的任何数据。CPC 202 负责将适量数据写入上下文高速缓存 224。上下文高速缓存 224 对 FOM 208 是可访问的，其在对闪存单元实现实际的命令时使用全部的闪存上下文信息。FOM 208 向 32 位双字提供地址，并且上下文高速缓存 224 在后续时钟周期响应请求的双字。在处理器控制模式期间，来自上下文高速缓存 224 的读取响应被可以基于操作的内容实现行动的上下文队列控制器（CQC）226 监控。正如上下文表 220，上下文高速缓存 224 也可以通过处理机接口访问。

[0070] FOM 208 读取来自上下文表 220 的条目并且执行操作。在执行操作之后，FOM 208 更新上下文状态机（CSM）222。CSM 222 将该状态存储在上下文表 220 的特定条目中。FOM 208 然后继续执行下一条目等。在到达上下文的末尾之后，FOM 208 回滚并且再一次执行条目以检查执行的下一部分。执行上下文表 220 中的多条目的处理通常提供对闪存接口上的命令的流水线执行。闪存接口上的命令的流水线执行通常提供闪存接口的有效利用。通过一部分一部分地执行多个条目以及恢复上下文表 220 中的状态，FOM 208 提供比常规的技术更有效的闪存接口的利用。

[0071] 参考图 9，示出了图解根据本发明实施方式的示例闪存介质上下文布局 300 的示图。各个闪存交易通常由上下文表示。上下文是包括系统硬件执行与闪存库（flash bank）的交易和 / 或将移动数据到系统缓冲器中的位置或从中移动该数据所使用的所有信息的数据结构。上下文通常由固件构建，并被作为到上下文内容实际所处的缓冲控制器（BC）里的指针而提供给闪存通道控制器（FLC）。固件可以选择构建这些上下文的链接列表（例如上下文列表）从而允许更大的闪存操作的硬件自动化。各个被管理的闪存单元（例如裸片、LUN 等）通常有一个上下文列表。

[0072] 通常，在普通操作期间，全部上下文只有一部分被 FLC 用于执行闪存交易。因此，只有被 FLC 所使用的部分存储在 FLC 的上下文高速缓存中。在正常操作期间均被存储在 FLC 的上下文高速缓存中的所有上下文的一部分通常被称为“闪存上下文”。正如可以从闪存介质上下文布局 300 中可以看到的，在一个示例中，闪存上下文可以仅包括全部十五个双字上下文的前四个双字。

[0073] 在一个示例中，根据本发明实施方式实现的上下文可以包括下列内容和 / 或字段：闪存操作字段、闪存行地址字段、块描述指针 / 回拷（copyback）行地址字段、构成位字段、下一上下文指针字段、DMA 跳过（skip）掩码字段、数据缓冲器指针字段、备用

(alternate)数据缓冲器指针字段、逻辑块地址(LBA)字段、状态字段、闪存单元号(FUN)字段、可以包括元数据或构成数据指针、元数据、和/或闪存构成数据的字段、可以包括元数据或者闪存构成数据的字段、如果使用64位LBA模式字段则可以包括元数据或逻辑块地址(LBA)的上部的字段、可以包括上下文的错误校正编码技术(ECC)的字段。闪存操作字段可以包括操作码。在一个示例中,操作码可以包括八个位。操作码可以由固件产生,以向硬件传递要执行的操作。通常,操作涉及对闪存行列的至少一次访问。固件可用的基本操作码可以包括表示下列操作的值,包括例如RESET、SYNCHRONOUS_RESET、READ_ID、READ_PARAMETER_PAGE、GET_FEATURES、SET_FEATURES、READ_PAGE、PROGRAM_PAGE、ERASE_BLOCK、COPYBACK_READ/PROGRAM、READ_STATUS、READ_STATUS_ENHANCED、PROCESSOR_CONTROL_MODE、MUILT PLANE PROGRAM PAGE以及MUILT PLANE PROGRAM PAGE END。

[0074] 闪存行地址字段可以包括闪存存储器中的访问的页的行地址。在一个示例中,闪存行地址字段可以包括与页面地址串接的块地址。闪存行地址可以在三(3)个行地址操作周期期间提供给闪存阵列。对于清除操作,仅提供闪存存储器中的访问的块地址。对于仅使用一个字节地址的READ_ID操作,在一个示例中,字节可以被放入闪存行地址的最左边的字节(MSB)。在闪存操作为PROCESSOR_CONTROL_MODE时,闪存行地址字段也可以用作软上下文指针。在遇到处理器控制模式操作时,FLC可以使用闪存行地址字段的值作为软上下文的系统缓冲器地址。

[0075] 在将数据移至系统缓冲器的位时,DMA跳过掩码字段可以基于位的状态(例如‘1’或‘0’)而允许转发或者跳过页扇区(分区)。在一个示例中,DMA跳过掩码字段可以被实现为低有效(active low)跳过掩码或转发掩码(例如‘1’表示发送而‘0’表示跳过相应页)。DMA跳过掩码字段可以主要用于读取/修改/写入(RMW)操作,这对少量扇区的交易而言是共同的。例如,在8K的页器件上,页通常被分成十六(16)个扇区。在DMA跳过掩码字段的特定位为1时,在一个示例中,相应的扇区可以被写入系统缓冲器或从中读出。在该位为0时,相应的扇区可以从被写入系统缓冲器中或从系统缓冲器读出中省略。在一个示例中,也可以存在允许数据从两个分离的源写入的模式:包含未修改的页数据的缓冲器块,以及包括修改的页数据缓冲器块。对于这些情况,状态(例如‘0’)可以用于表示扇区从包含更新的页数据的块而写入,其他状态(例如‘1’)用于表示扇区从包含未修改的页数据的块而写入。位的位置也可以是有意义(significant)的。例如,最高有效位(MSB)通常对应于扇区中的第一页或最小数量的页的位。最低有效位(LSB)通常是对应于扇区的最大数量的页的位。如果在页中存在小于16个的分区,最低有效位可以被视为是可忽略的。然而,可以实现其他数量的条目以达到特定实施方式的设计标准。在一个示例中,在实现博斯乔赫里霍克文黑姆(bose_chaudhuri_hocquenghem)错误校正码(ECC)时,掩码粒度可以是一(1)个信息字。

[0076] 下一上下文指针字段可以指向包括由固件构建的上下文列表(例如上下文的链接列表)中的下一上下文的指针。如果“下一上下文指针”等于被固件编程到FLC里的“列表末尾指针”,硬件可以假定检索已到达末尾。硬件在继续遍历链接列表之前可以等待“列表末尾指针”值的改变。也可以定义“空指针”值,其将停止硬件的表检索。

[0077] 在一个示例中,块描述指针(chunk descriptor pointer)/回拷行地址字段,通常对于回拷命令将要回写入的缓存数据提供闪存存储器页的行地址。行地址为块地址和页面

地址的串接。在另外的示例中，回拷闪存行地址字段与非回拷上下文的未使用的字段共享。例如，块描述符指针 / 回拷行地址字段可以用来提供缓冲器分配管理器（BAM）的辅助字段。BAM 辅助字段可以包括由缓冲器分配管理器（BAM）管理的数据缓冲器中的数据块的描述符的指针。

[0078] 块描述符字段可以用于缓存管理。块描述符可以包括缓冲器管理中所用的块地址（例如在系统缓冲器中）、有效位、脏（dirty）位、传输暂停计数、状态、LBA 和指针。在 DMA 已经完成之后，块描述符指针通常被闪存介质控制器（FMC）中的 DMA 管理器传送至 BAM。块描述符指针也可以被传送到 BAM，以便如果标志位（例如 BAM 查找所需要的位）被设置则获得缓冲器数据指针以开始 DMA。块描述符指针 /BAM 辅助字段也可以被固件设置为通用 BAM 辅助字段，并且内容可以由固件根据 BAM 序列器具体编程而确定。通常，块描述符指针 /BAM 辅助字段对硬件是完全显而易见的。

[0079] 数据缓冲器指针字段通常提供指向将要从缓冲控制器传送到闪存介质或者从闪存介质传送到缓冲控制器的实际数据的指针。在上下文起初被取得时，数据缓冲器指针字段存在于上下文中，或者如果设置 BAM 查找所需要的位，数据缓冲器指针字段可以被 BAM 通过 BAM 查找而填入（populate）。数据缓冲器指针字段通常提供块中的第一个字节的地址，而不一定为第一个有效 LBA。

[0080] 备用数据缓冲器指针字段也是从缓冲控制器传送到闪存介质或者从闪存介质传送到缓冲控制器的数据的指针。在操作使用进 / 出缓冲器的多源或多目的地时，使用备用数据缓冲器指针字段。备用数据缓冲器指针主要可以用于读取 / 修改 / 写入（RMW）操作。在主机（host）执行小于整个页的操作时，主机进行 RMW 操作以更新页。备用数据缓冲器指针可以用于读取操作，以便将旧的页数据存储在一个临时块中。然后，在实施写入操作时，跳过掩码可以用于从包括未修改的页数据的介质块或包括要被更新的数据的主机块中选择扇区（或分区）的写入源。

[0081] 元数据字段通常包括与闪存页有关的元数据（例如，诸如 LBA、顺序编号、坏块指示符等）。通常可以预期所有的元数据会容纳在上下文中。否则，一个元数据缓冲器指针可以包含在上下文中以代替元数据，从而元数据可以存储在系统缓冲器中或从系统缓冲器中检索。对于写入，元数据字段通常被固件填入并且被硬件插入到页中。对于读取，对于固件来说，通常期望请求读取页中的元数据的内容。对于这些情况，上下文可以被配置为从页中读取元数据到上下文中。在一个示例中，可以由元数据大小寄存器确定以字节表示的元数据的大小。

[0082] 闪存配置数据字段通常与元数据字段共享字节位置，并通常用于传输有限数量字节的操作（例如 READ_ID、GET_FEATURES、SET_FEATURES、READ_STATUS 等）。由于与这样的传输有关的字节数通常较小（例如 SET_FEATURES 和 GET_FEATURES 可以使用 4 个字节 ;READ_ID 通常使用 5 个字节 ;READ_STATUS 可以仅使用 1 个字节），数据仅在上下文中而不是在一个独立的缓冲器位置中传输。如果与这样的传输有关的数据超出在上下文中分配的空间，可以使用闪存配置数据指针字段。“GET_FEATURES 和 SET_FEATURES”命令往往假定使用 4 个字节。在一个示例中，从 READ_ID 命令寄存器的读取字节的数量中可以获得 READ_ID 命令所需的字节的数量。在闪存配置数据字段用于配置数据时，所有的 7 个双字均被硬件更新，并且固件仅需要读取适当的字节。

[0083] 元数据缓冲器指针字段包括页元数据(例如包含在闪存页的用户数据中的管理数据)的位置的地址。因为在许多应用中所有的元数据可以存储在上下文自身中是预期的,元数据缓冲器指针字段可以省略。在要从外部的系统缓冲器中检索元数据时,可以使用元数据缓冲器指针字段。通常只有元数据被储存在外部并且不在上下文中,才使用元数据缓冲器指针字段。在一个示例中,可以实现配置位,以指定元数据是储存在外部还是储存在上下文之内。在一个示例中,外部系统缓冲器可以被实现为包括闪存介质控制器(FMC)的芯片的外部的存储器(例如 DDR RAM)。在又一示例中,外部的系统缓冲器可以被实现为 FMC IP 的外部的片上 RAM。在另一示例中,外部的系统缓冲器可以包括 FMC IP 内的存储器。通常,外部的系统缓冲器可以为上下文的外部的任何存储。

[0084] 闪存配置数据指针字段通常提供取得的或需要被写入一个闪存单元的构成数据的位置地址。由于元数据被分配给上下文字段以用于数据交易,所以与 READ_ID、GET_FEATURES 和 SET_FEATURES 命令有关的闪存配置数据可以被存储在上下文中。然而,就像元数据的情况那样,如果数据大小超过上下文中为闪存配置数据所分配的空间,系统缓冲器的数据的指针可以被用于访问。

[0085] 逻辑块地址(LBA)字段通常提供页中的第一个数据分区的 LBA。针对系统缓冲器和闪存的各个扇区,LBA 通常被编码到数据保护。由于页为连续的 LBA 组,只有该组中的第一个 LBA 用作上下文的一部分。LBA 可以用于为扇区页的缓冲器 CRC 做种(seed)。针对元数据的 LBA 部分,LBA 也可以有选择地进行校验。例如,可以实现配置位以选择是否相对于元数据的 LBA 部分来校验 LBA。例如,在配置位被设置时,LBA 可以对元数据的 LBA 部分进行校验。在一个示例中,只有低 32 位可以用于做种。

[0086] 闪存单元号(FUN)字段可以由固件用作用于确定哪个闪存单元将应用上下文的标识符。该确定对硬件而言是清楚的。闪存单元号字段被固件单独地用于管理目的,使得在上下文已经被硬件消耗之后,一旦标识符被提供回固件,固件可以协调上下文。

[0087] 构成位字段通常包括用于构成上下文并以硬件流中的各种点确定上下文(以及上下文表示的传输)的处理的所有位。在一个示例中,上下文配置位字段中的可用的上下文位和字段可以包括但不限于以下:消耗上下文管理器中断启用;部分命令启用位,使用闪存接口上的部分命令功能启用/禁用执行 DMA 跳过掩码;禁用对系统缓冲器的 DMA;禁用任何从或到 FLC 本地缓冲器的数据传输;扰码器功能启用,启用/禁用扰码器功能;ECC 错误检测中断启用;ECC 宏旁路,启用/禁用读取数据的解码或附加奇偶性至写入数据;缓冲器 CRC 启用;忽略元数据信号(位),使元数据在读取时不被转发到系统数据缓冲器、系统缓冲器或上下文,并在写入时防止从任何源插入元数据(例如,叶子元数据字段空白);信号(位),使用于 DMA 的用户数据保持元数据(例如,在读取时转发元数据到数据缓冲器,以及在写入时从数据缓冲器接收元数据);信号(位),指示是否要将元数据中的“开始 LBA”字段与上下文中的开始 LBA 字段相校验;信号(位),指示是否在缓冲器中保持 ECC 奇偶校验字段(例如,在读取时将 ECC 奇偶校验字节传输到系统缓冲器,以及在写入时将 ECC 从系统缓冲器传输到闪存);信号(位),指示是否使用平面缓冲器(例如,数据传输进/出非托管存储器区域)而不是托管数据缓冲器(例如,可用于确定在量子突发中传输多少数据到系统缓冲器);信号(位),指示是否使用用户数据扇区长度配置以确定扇区的数据长度(例如,如果该位是被清除的,扇区的数据长度可以根据保留区域扇区长度配置而确定);信号(位),指示是否传

输闪存配置数据(例如,将闪存数据读取入到上下文中,或写入来自上下文的闪存数据);信号(位)指示是否传输全部(full)原始闪存页面到系统缓冲器或从中传输全部(full)原始闪存页面;信号(位)指示是否使用备用缓冲器跳过掩码(例如,使跳跃掩码用于标识(mark)将被传输进/出被备用数据缓冲器的指针所指向的缓冲器块的扇区(分区),而不是省略扇区的传输;在启用扰乱器时为各个分区定义扰乱器种子的字段。指示是否传输一个全部原始闪存页面到或从系统缓冲器的信号,其实质上与大多数其他选项是互斥的。指示是否传输一个完整的原始闪存页面到或从系统缓冲器的信号通常超越于(overide)于其他位的设置。通常传输一个完整的原始闪存页到或从系统缓冲器的特征使固件掌握(get at)不属于任何分区的页的未使用区域。传输闪存配置数据信号可用于诸如 ops、READID 命令,GETFEATURES,以及 SETFEATURE。数据的大小,可由配置数据长度寄存器确定。数据将出现在元数据中通常由上下文占据的位置。

[0088] 状态字段通常包括可用于向固件返回状态的位。在一个示例中,在状态字段可以包含编程/擦除操作所用的通过(pass)/失败(fail)状态。在另一示例中,状态字段还可以包含读取操作中 ECC 逻辑检测到的错误数量。然而,可以实现该字段的其他用途,以满足特定实现的设计标准。在一个示例中,状态字段可配置为表示闪存页面中的错误数量和用于上下文的管理器的完成代码(例如、清除(clean)、编程/擦除错误、校正的读取错误、无法校正的读取错误、从系统缓冲器中的预取(prefetch)上的缓冲器 CRC 错误、操作超时、LBA/元数据不匹配、非法操作等)。

[0089] 如上所述,上下文是确定了命令执行的数据结构。上下文为固件和硬件之间的通信单元。参考图 2 和图 3,一般的上下文流可以总结如下。固件与 150 内存指针一起写入 FMC102 的处理器接口逻辑(PIL)的上下文结构缓冲器(CCB)中的上下文。多个上下文可以通过使用上下文结构中的下一指针字段而由上下文中的固件链接。硬件通过上下文更新端口(CUP)168 来执行到内存指针相关联的位置的上下文的存储器写入。例如,固件可以通过各自的闪存通道控制器(FLC)156a 至 156n 中的裸片管理模块(DMM)206 中的裸片管理寄存器来启用特定的裸片的执行。可以为各个裸片提供一套独立的管理裸片寄存器。DMM 206 将寄存器中的信息发送(communicate)至各自的闪存通道控制器(FLC)156a 至 156n 中的上下文管理器(CM)204。CM204 可以通过上下文检索端口(CRP)166 从内存中提取启用上下文,并且调度对各自的闪存通道总线上的命令的执行。CM 204 通常调度命令并且指示用于面向数据操作的数据 DMA 管理器(DDM)块 152。DDM 块 152 通常从存储器中提取上下文以通过 CRP 166 从上下文中提取数据参数。在由 DDM 152 和 CM 204 进行的上下文成功地完成以后,状态被更新到消耗上下文管理器(CCM)164。在完成后,可由硬件产生中断并且固件可以读取出给定上下文下的完成状态。

[0090] 包括块 172a 至 172n 和块 174A 至 174n 的数据 DMA 接口端口(DDIP)通常负责 DTP 158a 到 DTP 158n、系统缓冲控制器和相关上下文之间的实时数据路由。各个数据传输可包括一个或多个分区。各个分区可以包括一个或多个区域。实际格式来自 FMC(例如,使用寄存器)以及来自相关上下文的配置设置而确定。设置可按各传输、各分区以及各区域基础进行分类。路由可根据数据传输格式的区域类型以及其他配置而确定。路由通常包括合并和分片(padding and stripping)功能。各个数据传输涉及的所有上下文的双字。因此,DDM 152 发起上下文对 DDIP 的访问,其中 DDIP 在实际数据移动之前,将所有上下文双字读取入

其自己的上下文高速缓存区域。

[0091] 一旦检索到相关上下文，DDIP 以适当的配置设置来编程其内部控制逻辑，并提供一些配置信息至相应的 DTP 158a 到 DTP 158n。然后 DDIP 跟踪各个分区内的各个区域，并且建立用于适当路由的逻辑。在实际数据传输中，如果数据被重定向到上下文，则 DDIP 会切换上下文访问至写入模式。当传输完成后，DDIP 会通知 DDM 152 传输和上下文访问都已完成。DDIP 可以执行错误校验。例如，DDIP 可配置为执行系统缓冲器 CRC 校验和 / 或确认接收到的 LBA 符合来自上下文的预期的 LBA。这些校验以及系统缓冲器 ECC 字节错误（启用时）的结果通常都传回 DDM 块 152。DDM 块 152 可以转而将信息传至选定的 FLC 156a 至 156n。DDIP 也可以配置为生成和插入 CRC。可以通过从各分区收集所有用户数据字节而构建整个用户数据。同样，可以通过从各分区收集元数据字节而构建整个元数据。在一个示例中，可以收集元数据直到达到预定的上限。在一个示例中，DDIP 可以具有默认配置，其中 DDIP 仅路由已用用户数据区域的部分进 / 出系统缓冲器，并且仅路由已用数据区域部分进 / 出上下文元。ECC 区域可以分片 / 合并。可以修改默认配置以提供多种变化。例如，DDIP 可以被配置为 :（1）保持整个用户数据区完好，包括未使用的字节，（2）使进 / 出系统缓冲器域的元数据区域保持用户数据，（3）分片 / 合并元数据区域，（4）保持 ECC 完好，和 / 或（5）保持整个分区完好（例如，没有分片 / 合并）。

[0092] 上下文布局结构 300（在上面结合图 9 所述的）可用于任何目标中的页的编程。顶端模块之间的示例编程数据流可以概括如下。固件程序以 CCB 中的关联存储器指针、在处理器逻辑接口（PIL）150 中、以命令对上下文编程。CCB 通过 CUP_168 执行到存储器的写入操作。固件可以启用各自 FLC 156a 至 156n 中的 DMM 206 以允许上下文的执行。各自的 FLC156a 至 156n 中的 CM 204 通过 CRP 166 从内存提取上下文，并且处理用于调度各自的闪存总线控制器（FBC）154a 至 154n 上的命令的上下文部分。DDM 与 FLC 152 针对上下文进行通信，DDM 152 通过 CRP 166 提取上下文，并且提供数据块指针至各自的 DTP 块 158a 至 158n。各自的 DTP 块 158a 至 158n 通过从上下文中解码的指针从内存请求数据。该请求是通过块 172a 至 172n 和块 174a 至 174n 组成的 DMA 接口数据端口（DDIP）进行的。数据 DMA 接口端口（DDIP）在量子突发中从存储器读取数据。

[0093] 从内存中读取的数据通常通过由块 172a 至 172n 和块 174a 至 174n 形成的 DDIP 传递到 DTP。在 DTP 中，如果启用的话，数据可以进行 CRC 校验，并且如果启用的话，ECC 可以添加到用户数据并传递到各自的 FLC156a 至 156n。如果数据的完整性失败，状态更新到 CCM 块 164。各自的 FLC 156a 至 156n 中的闪存操作管理器（FOM）208 调度编程周期有关的命令和数据至各自的闪存总线控制器（FBC）154a 至 154n。各自的 FBC154a 至 154n 以闪存接口规范对闪存总线接口执行编程操作。各自的 FBC154a 至 154n 读取闪存操作完成的状态。在操作完成时，各自 FBC 154a 至 154n 与各自 FLC 156A 至 156N 通信，并且各自的 FLC 156a 至 156n 与 CCM 164 通信。CCM 164 维持中断至固件以通知位于在 CCM 164 相关的上下文寄存器中的上下文已完成，并且准备好固件的检查。然后，固件从 CCM 164 读取上下文。当固件完成上下文的读取和处理时，固件可以维持允许加载另一消耗的上下文的信号（例如，位）。

[0094] 基本的闪存读取数据流可描述如下。一个上下文结构用于读取任何目标页。固件程序以 CCB 中的关联存储器指针在处理器逻辑接口（PIL）150 中以命令对上下文编程。CCB

通过CUP_168执行到存储器的写入操作。CCB通过CUP 168执行对存取器的写入．固件启用各自的FLC 156a至156n中的DMM_206以允许执行上下文。各自的FLC 156a至156n中的CM 204通过CRP 166从存储器提取上下文，并且处理调度被用于各自的FBC 154a至154n的命令所需的上下文部分。各自的FLC 156a至FLC 156n与各自的FLC 154a至154n进行通信以执行对目标的读取命令。各自的FLC 156a至156n与DDM 152针对上下文进行通信，DDM 152通过CRP 166提取上下文，并且提供数据块的指针至各自的DTP块158a至158n。

[0095] 在接收到从闪存通过各自的FBC 154a到FBC 154n到各自的DTP158a到DTP 158n的数据时，各自的DTP 158a到DTP 158n通过来自上下文的解码的指针请求要写入存储器的数据。各自的DTP 158a到DTP 158n也通过由块172a至172n和块174a至174n(DDIP)形成的DMA接口数据端口这样做。DDIP在量子突发中将数据写入到存储器。在DTP中，如果启用的话，数据进行ECC校验，并且如果启用的话，CRC添加到用户数据并且该数据传到DDIP。如果数据的完整性失败，状态更新到CCM块164。在操作完成后，读取操作的状态被提供到CCM块164。CCM块164维持中断至固件以通知固件，在CCM块164的上下文寄存器中位于的上下文是完整的，并以为固件检查做好准备。固件读取来自CCM块164的上下文。当固件完成上下文的读取和处理，固件可以维持允许加载另一消耗上下文的信号(例如，位)。

[0096] 根据本发明的闪存控制器硬件架构总体上被设计用于提高闪存系统的整体性能。该设计也提高了闪存总线接口效率，并通过固件编程与上下文交互。由多个硬件特征所定义的独特的硬件体系结构了提高了闪存总线接口的效率。通过独特的上下文流提高了固件的交互。通过支持对经由由固件编程的更高层闪存命令上下文到闪存器件的多重闪存的命令序列的自动硬件处理，根据本发明的硬件结构也可以有助于支持更有效的固件设计。

[0097] 根据本发明实施方式的闪存介质控制器可包括以下特征：1)各个闪存交易的上下文；2)上下文处理；3)经由消耗上下文管理器(CCM)的状态报告；4)各个闪存的通道控制器内的裸片管理表以及上下文管理器块；5)独立通道结构，包括专用数据传输路径；6)支持在闪存通道控制器和缓冲控制器接口中的全双工操作；7)处理器控制模式。

[0098] 对于每个闪存交易的上下文，每个闪存交易由上下文表示。上下文是包含硬件执行与闪存库的交易和/或移动数据到系统缓冲器的位置或从中移动数据所需的全部信息的数据结构。上下文一般由固件构建，并以指向缓冲控制器的上下文内容实际所处位置的指针而提交给各自的闪存通道控制器。固件可以选择构建上下文列表，以便允许更大的闪存操作的硬件自动化。通常，各托管闪存单元(裸片，LUN)存在一个上下文。因为各个LUN存在独立的上下文，使之能够产生流水线并行操作。

[0099] 根据本发明实施方式的闪存介质控制器还可以提供上下文处理。闪存介质控制器可以从多个上下文链(例如，可专用于LUN的上下文链)中提取上下文。这将产生线程并行处理。并行处理线程有助于执行多个上下文，这意味着在以下几个方面中的流水线操作：(一)在多重闪存通道之间内，(b)在闪存通道内(c)在闪存目标内，以及(d)通过使用多通道命令在LUN内。

[0100] 通常，按照本发明实施方式的消耗上下文管理器(CCM)提供可以报告来自所有LUN的所有完成状态的单一接口。CCM可以被实现为闪存介质控制器(FMC)的子块。CCM通常提供在完成后可以通过固件检查已完成的上下文的单点。CCM通常在多个FLC源之间执行仲裁。各个FLC源提供上下文指针有关的PASS/FAIL和ECC的状态。一旦取得上下文，

CCM 更新上下文状态字段。然后 CCM 提交该上下文至固件。当固件需要较长时间来读取已完成的上下文以及内部存储器内的 CCM 即将变满时，CCM 使用缓冲器来存储在当前的报告上下文之后排队的已完成的上下文。

[0101] 在一个示例中，裸片管理表和上下文管理器块可位于闪存通道控制器内。裸片管理表可以实现为保持各个裸片(或 LUN)的提取上下文的当前状态的硬件。通过实现这一方法，硬件保持通道内目前正在进行的并行线程的完整信息。这有助于管理上下文的并行线程处理。上下文管理器可以实现为管理上下文在闪存总线接口上的执行的硬件模块。这有助于在闪存总线接口上执行多个命令。如图 5 中所示，裸片管理表可以保持多个条目，其中条目数取决于在给定的闪存通道上的 LUN 数量。各个条目可以专用于一个 LUN。各个条目可以保持链的下一指针、链的尾指针和上下文当前状态的信息。

[0102] 上下文管理器通常保持多个条目，其中，条目数量通常确定了可以在通道上执行的并行闪存交易的数量。这里的各个条目专门用于闪存操作。各个条目保持上下文指针、裸片数量、操作类型和上下文当前状态。

[0103] 本发明提供独立通道结构。独立通道结构包括专用的数据传输路径。专用的数据传输路径特征有助于在各个闪存通道上独立地执行上下文线程。使用根据本发明的结构，通过增加闪存通道的数量可以增加数据带宽。例如，各个闪存通道可以支持多达 8 个闪存目标(例如，启用 8 个芯片)。各个闪存目标可以具有多个 LUN。

[0104] 本发明还可以提供在闪存通道控制器和缓冲控制器接口内的全双工操作支持。闪存编程和读取交易可以在闪存总线接口上顺序地执行，这也意味着闪存总线接口是半双工的。然而，闪存控制器内部实现全双工数据路径从而使控制器是全双工的。全双工控制器有助于减少在闪存接口的方向变化时可能发生的延迟。例如，在闪存上的总线接口的编程命令中，这意味着数据在总线上从闪存控制器传输到闪存器件，并且在闪存控制器内部的同一时间，先前执行的读取命令所接收的数据可以传输到缓冲器，反之亦然。这有助于减少在闪存总线接口上的方向变化中的周转时间。

[0105] 本发明还提供了一个处理器控制模式。FMC 自身只支持获得最大标称(maximum-quoted)性能所需的最基本命令的子集。即，硬件自动化特征只对那些获得最高性能所需的命令进行了优化。其目的是为了简化硬件自动化和简化固件设计，同时保证与所有现有的 NAND 闪存的互操作性。然而，这并不是说硬件限制发布这些命令。相反，硬件可直接由固件指示来执行几乎所有可以在闪存上执行的原子操作，并且固件可以承担 FMC 内嵌的硬件资源的直接控制，以促进闪存的控制和数据移动的控制。该能力被称为处理器控制模式。

[0106] 正如相关领域技术人员所显而易见的，图 1 到图 8 所示的功能可以使用一个或多个传统的通用处理器、数字式计算机、微处理器、微控制器、RISC (精简指令集计算机)处理器、CISC (复杂指令集计算机)处理器、SIMD (单指令多数据)处理器、数字信号处理器、中央处理单元(CPU)、算术逻辑单元(ALU)、视频数字信号处理器(VDSP)和 / 或类似计算机来实现，并且可以根据本说明书的教导进行编程。还正如相关领域技术人员所显而易见的，熟练的编程员可以基于本发明的说明来编制合适的软件、固件、编码、编程、指令、操作码、微码、和 / 或编程模块。软件通常从一个中等或多个介质中由机器实施的一个或多个处理器来执行。

[0107] 本发明也可以通过ASIC（专用集成电路）、平台ASIC、FPGA（现场可编程门阵列）、PLD（可编程逻辑器件）、CPLD（复杂可编程逻辑器件）、海量门（sea-of-gates）、RFIC（射频集成电路）、ASSP（特定应用标准产品）、一个或多个单片集成电路、排列为倒装芯片（flip-chip）模块和/或多芯片模块的一个或多个芯片或裸片、或与常规部件电路网络适当互连的芯片的准备方式而实现，如本文所描述的，其修改对于本领域技术人员是显而易见的。

[0108] 因此，本发明还可以包括计算机产品，其可以是包括指令的存储介质和/或传输介质，其中，该指令可用于对机器编程以执行一个或多个根据本发明的处理或方法。由机器进行的计算机产品中所含指令的执行，随着周围电路的操作，可以将输入数据转换成存储介质上的一个或多个文件，和/或转换成表示物理对象或物质的诸如音频和/或视觉描述（depiction）的一个或多个输出信号。存储介质可以包括但不仅限于软盘、硬盘、磁盘、光盘、CD-ROM DVD和磁光盘的任何类型的圆盘，诸如ROM（只读取存储器）、RAM（随机存取存储器）、EPROM（可擦除可编程ROM）、EEPROM（电可擦除可编程ROM）、UVPROM（紫外线可擦除可编程ROM）、闪存、磁卡、光卡的电路，和/或任何合适的用于存储电子指令的介质类型。

[0109] 本发明的元件可以形成一个或多个装置、单元、部件、系统、机器和/或装置的一部分或全部。这些装置可以包括但不限于服务器、工作站、存储阵列控制器、存储系统、单个电脑、笔记本电脑、笔记本电脑、掌上电脑、单个数字助理、便携式电子器件、电池供电器件、机顶盒编码器、解码器、转码器、压缩机、解压缩、预处理器、后处理器、发射器、接收器、收发器、密码电路、蜂窝电话、数码相机、定位和/或导航系统、医疗器件、头戴式显示器、无线器件、录音、音频存储和/或音频播放器件、录像、视频存储和/或视频播放器件、游戏平台、外围器件和/或多芯片模块。相关领域技术人员应当了解，本发明的元件可以在其他类型的装置中实现，以满足特定应用的标准。

[0110] 尽管本发明已参考其优选实施方式进行了特别说明和描述，但是相关领域技术人员应当理解，在不背离本发明的范围的前提下可以在形式和细节上进行各种变化。

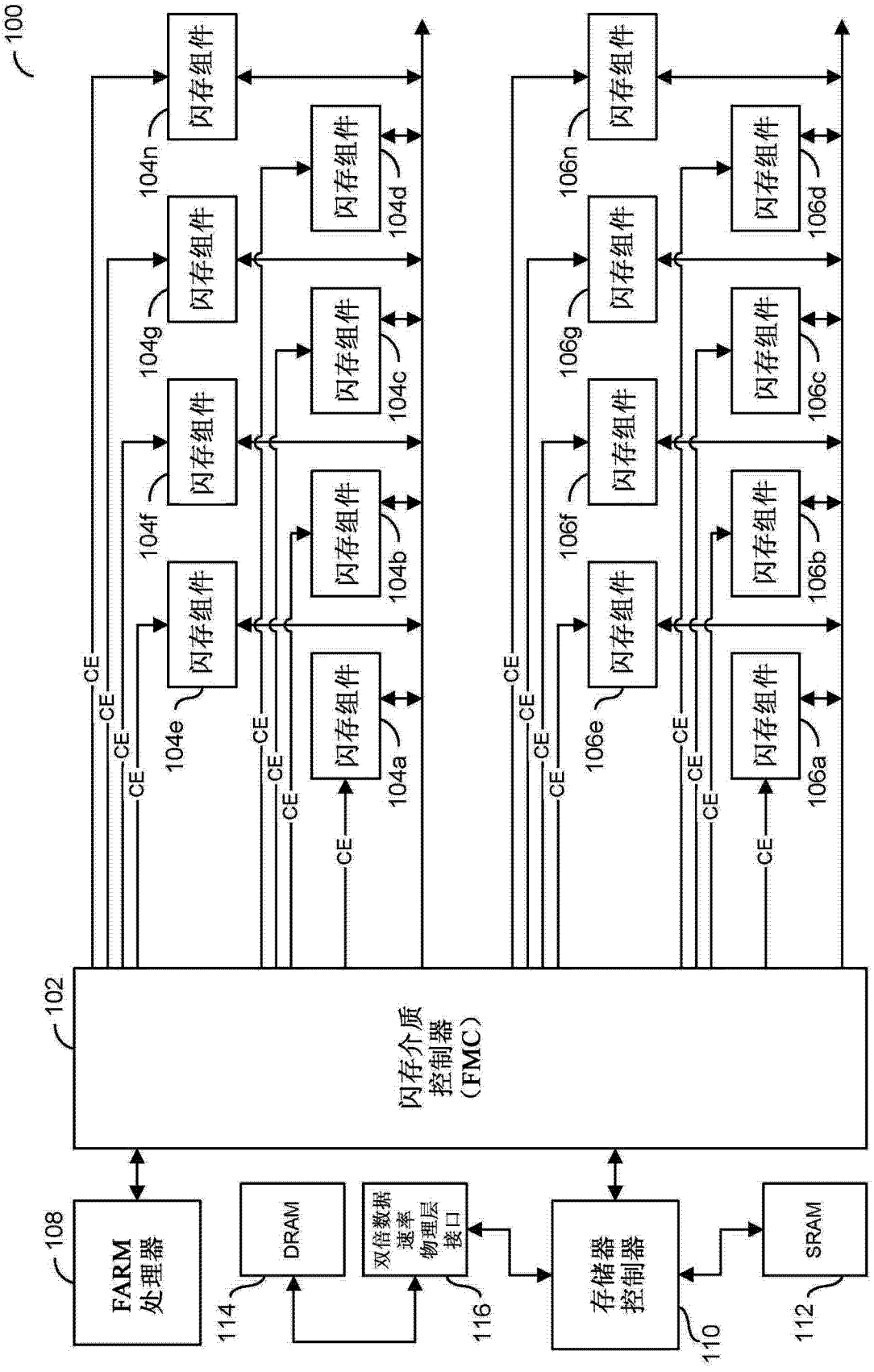

100
102
闪存介质控制器（FMC）
108
FARM 处理器
114
DRAM
双倍数据速率物理层接口
116
存储器控制器
110
112
SRAM
闪存组件
104a
104b
104c
104d
104e
104f
104g
104n
106a
106b
106c
106d
106e
106f
106g
106n
CE

图 1

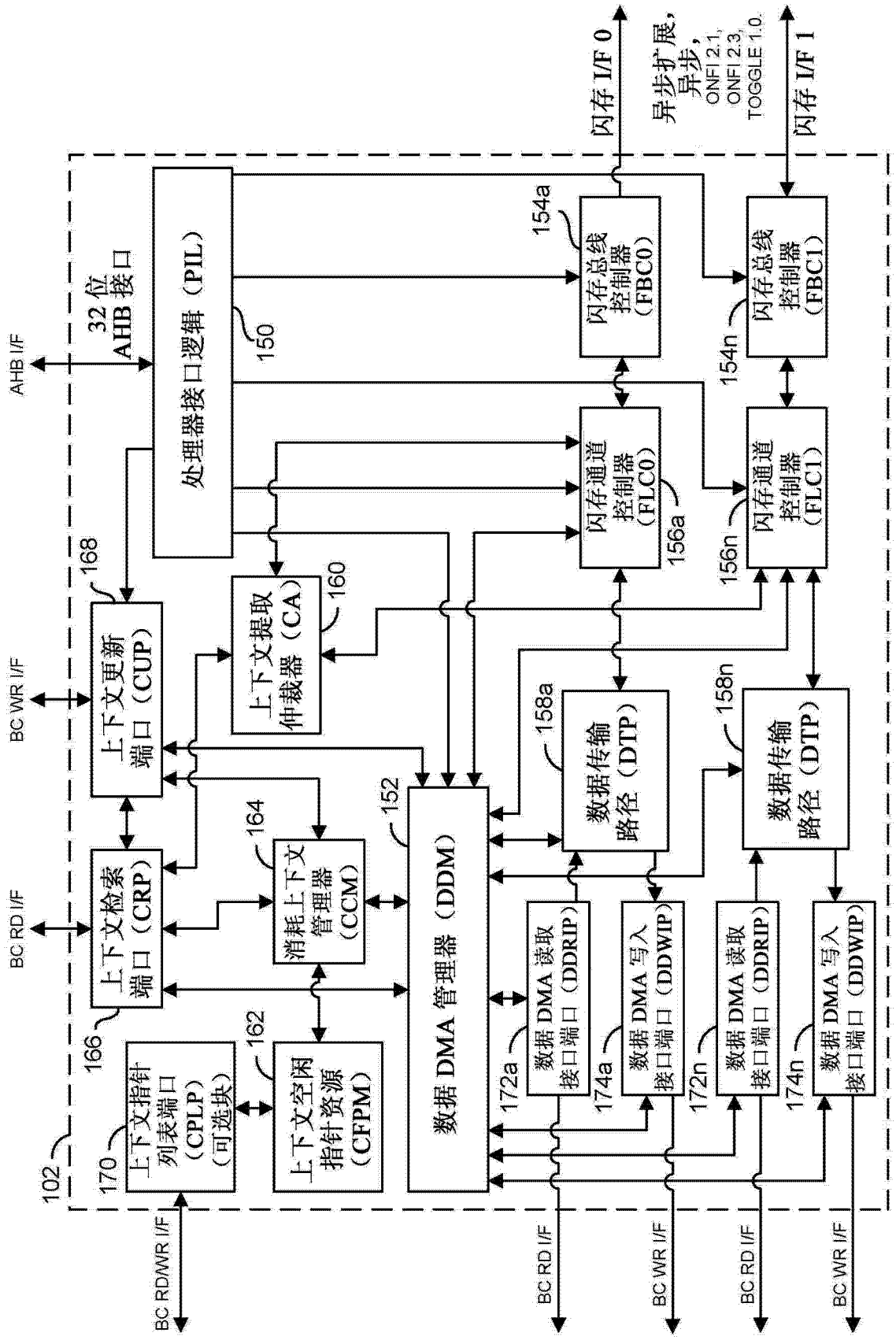

AHB I/F
32 位 AHB 接口
处理器接口逻辑（PIL）
150
闪存 I/F 0
异步扩展，异步，ONFI 2.1, ONFI 2.3, TOGGLE 1.0.
闪存 I/F 1
154a
闪存总线控制器（FBC0）
154n
闪存总线控制器（FBC1）
闪存通道控制器（FLC0）
156a
闪存通道控制器（FLC1）
156n
168
上下文更新端口（CUP）
BC WR I/F
上下文提取仲裁器（CA）
160
BC RD I/F
上下文检索端口（CRP）
164
消耗上下文管理器（CCM）
152
数据 DMA 管理器（DDM）
158a
数据传输路径（DTP）
158n
数据传输路径（DTP）
166
162
170
上下文指针列表端口（CPLP）（可选块）
上下文空闲指针资源（CFPM）
102
数据 DMA 读取接口端口（DDRIP）
172a
数据 DMA 写入接口端口（DDWIP）
174a
数据 DMA 读取接口端口（DDRIP）
172n
数据 DMA 写入接口端口（DDWIP）
174n
BC RD/WR I/F
BC RD I/F
BC WR I/F
BC RD I/F
BC WR I/F

图 2

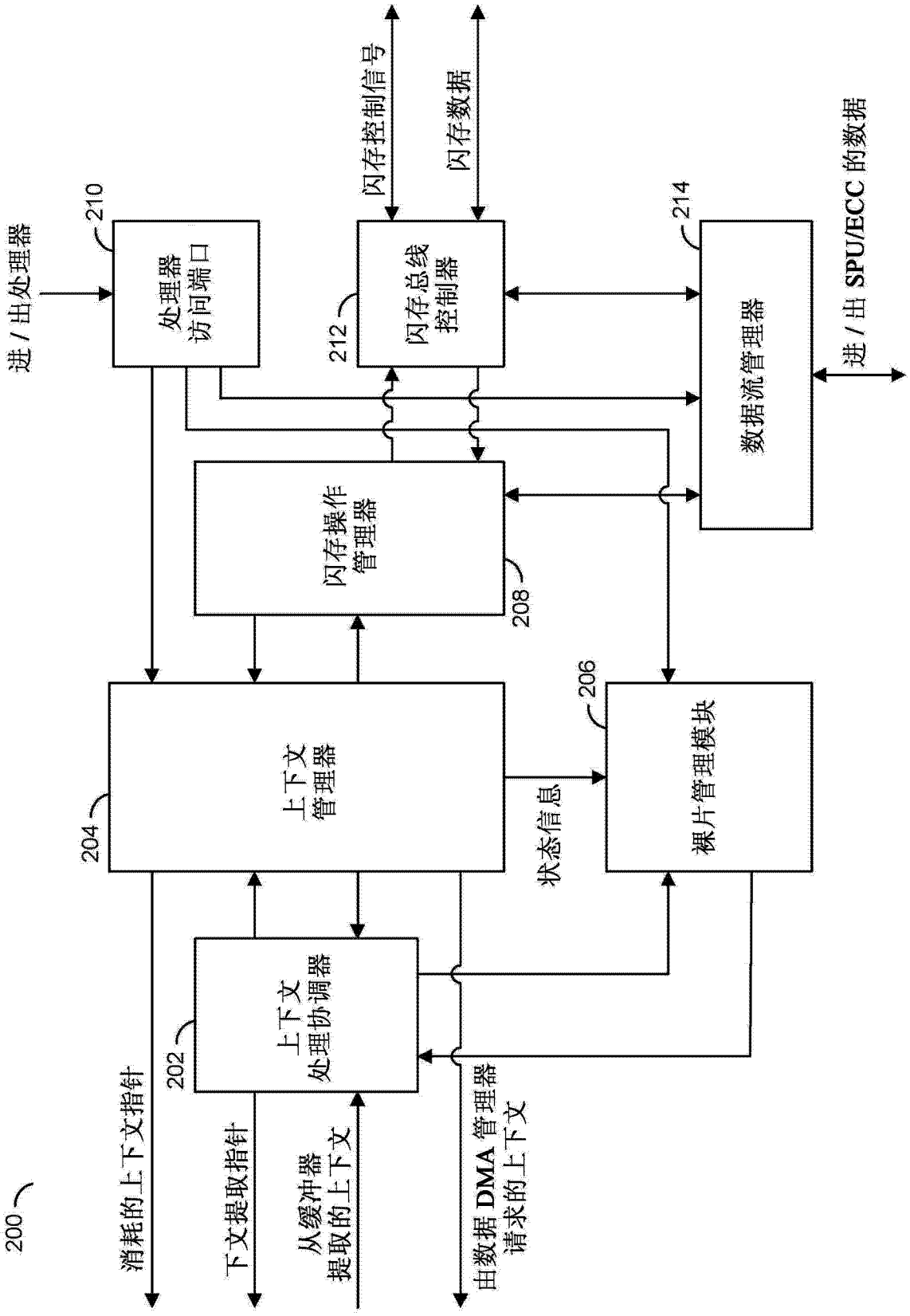

200
进 / 出处理器
210
处理器访问端口
闪存控制信号
闪存数据
212
闪存总线控制器
214
数据流管理器
进 / 出 SPU/ECC 的数据
闪存操作管理器
208
204
上下文管理器
206
裸片管理模块
状态信息
202
上下文处理协调器
消耗的上下文指针
下文提取指针
从缓冲器提取的上下文
由数据 DMA 管理器请求的上下文

图 3

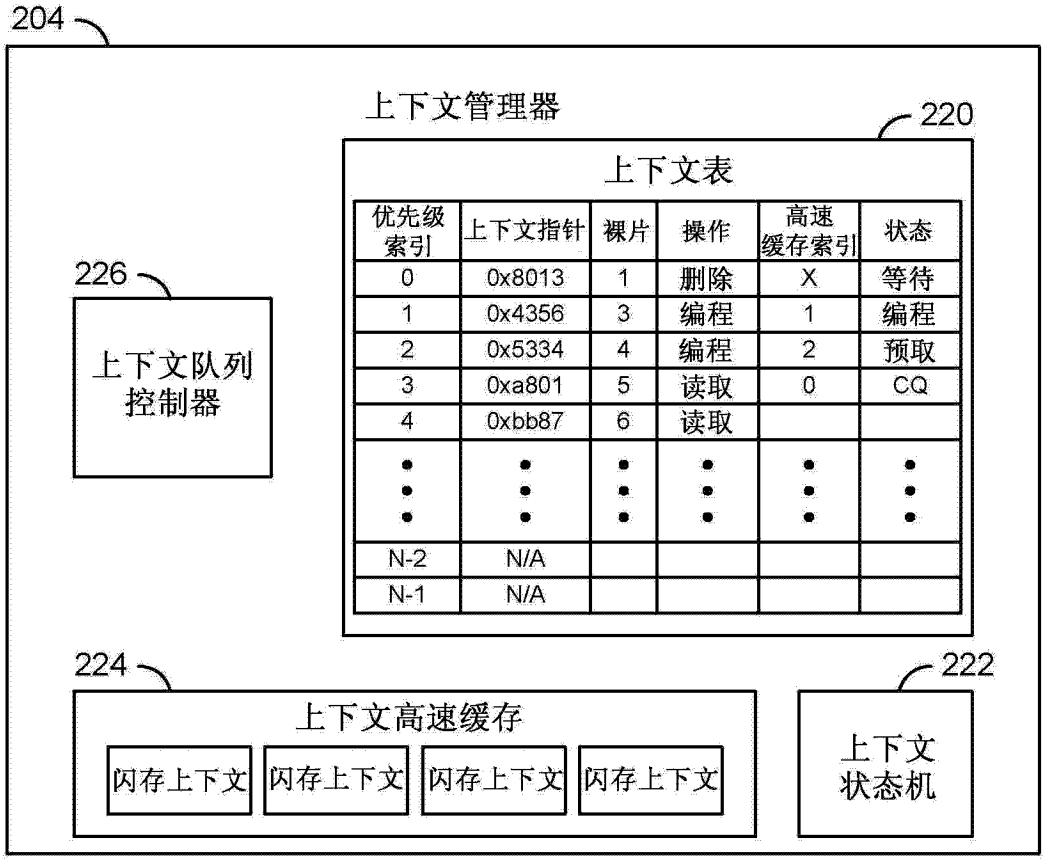

204
上下文管理器
220
上下文表
优先级索引
上下文指针
裸片
操作
高速缓存索引
状态
0
0x8013
1
删除
X
等待
1
0x4356
3
编程
1
编程
2
0x5334
4
编程
2
预取
3
0xa801
5
读取
0
CQ
4
0xbb87
6
读取
N-2
N/A
N-1
N/A
226
上下文队列控制器
224
上下文高速缓存
闪存上下文
闪存上下文
闪存上下文
闪存上下文
222
上下文状态机

图 4

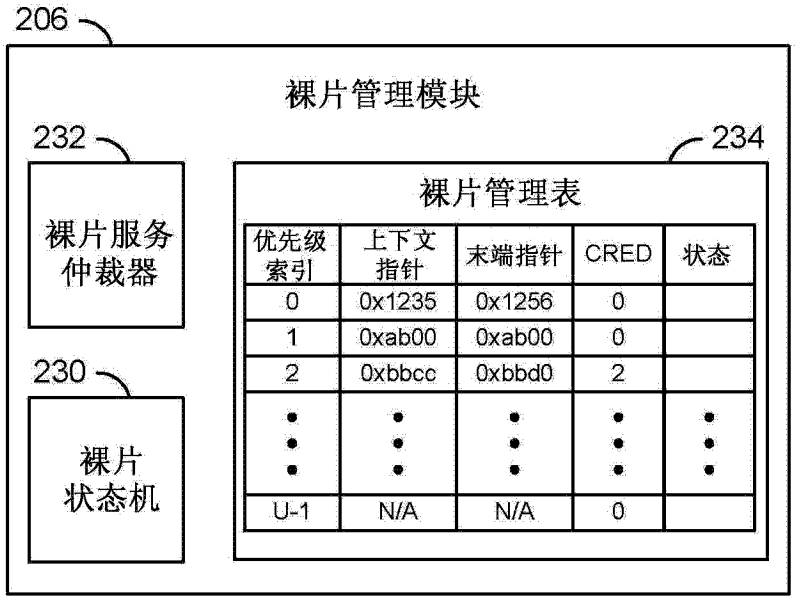

206
裸片管理模块
232
裸片服务仲裁器
230
裸片状态机
234
裸片管理表
优先级索引
上下文指针
末端指针
CRED
状态
0
0x1235
0x1256
0
1
0xab00
0xab00
0
2
0xbbcc
0xbbd0
2
U-1
N/A
N/A
0

图 5

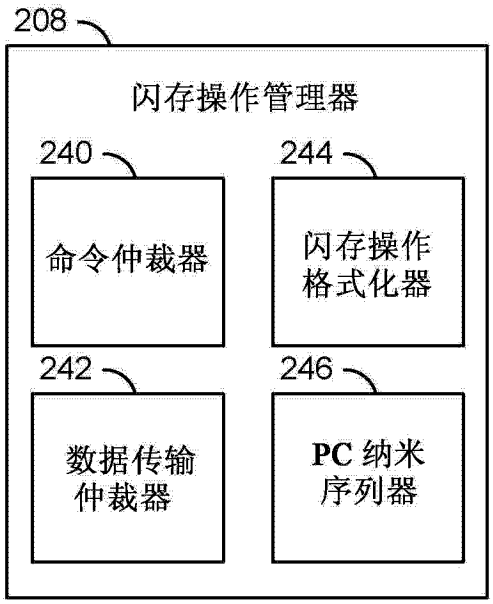

208
闪存操作管理器
240
命令仲裁器
244
闪存操作格式化器
242
数据传输仲裁器
246
PC 纳米序列器

图 6

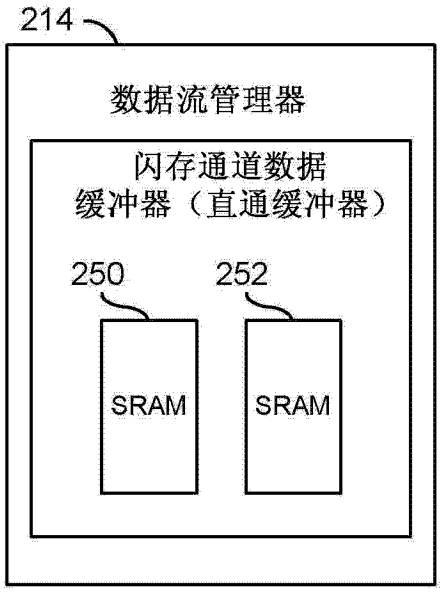

214
数据流管理器
闪存通道数据缓冲器（直通缓冲器）
250
SRAM
252
SRAM

图 7

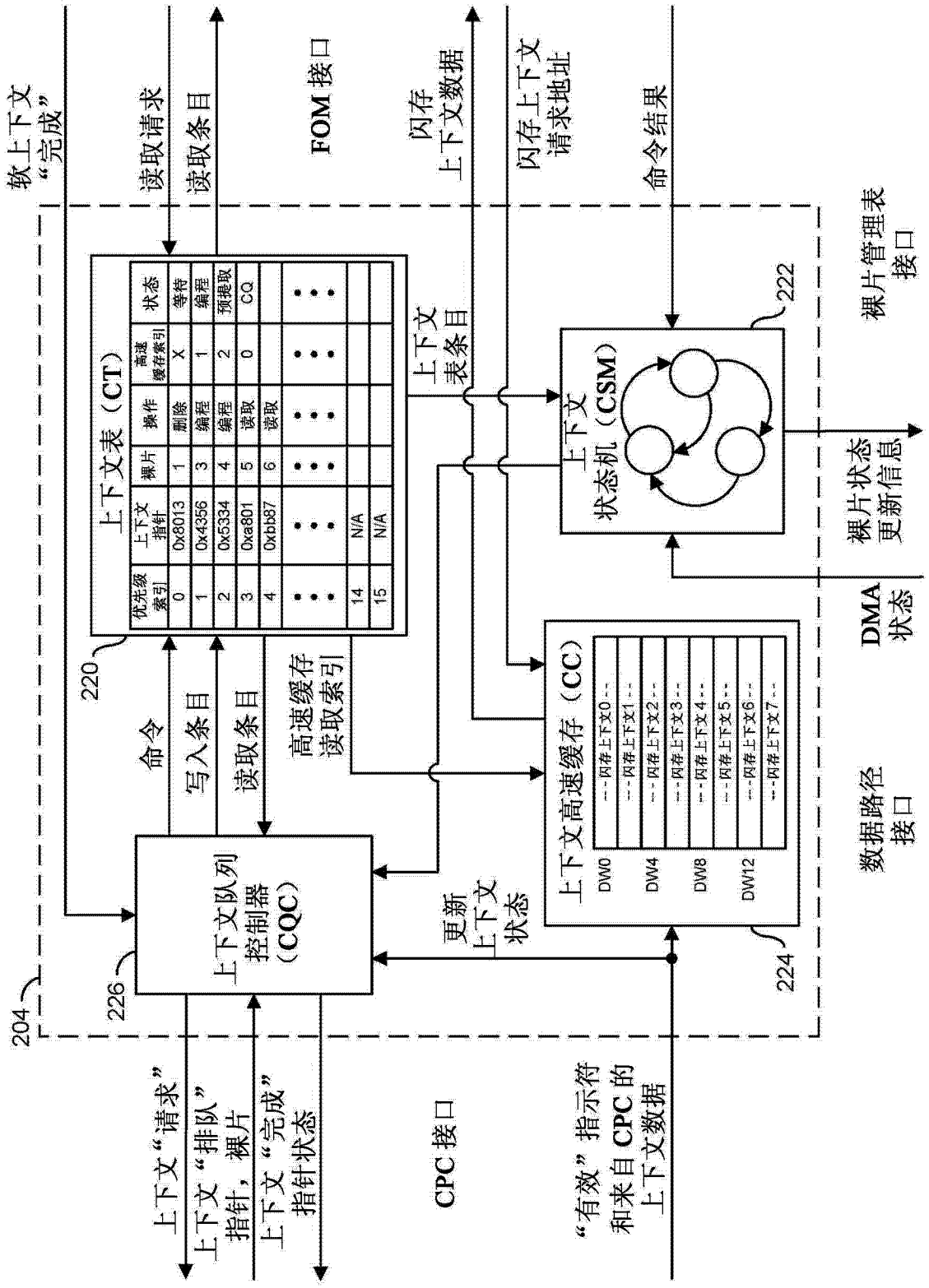

软上下文“完成”
读取请求
读取条目
FOM 接口
闪存上下文数据
闪存上下文请求地址
命令结果
裸片管理表接口
上下文表（CT）
优先级索引
上下文指针
裸片
操作
高速缓存索引
状态
0 0x8013 1 删除 X 等待
1 0x4356 3 编程 1 编程
2 0x5334 4 编程 2 预提取
3 0xa801 5 读取 0 CQ
4 0xbb87 6 读取
14 N/A
15 N/A
220
上下文表条目
上下文状态机（CSM）
222
裸片状态更新信息
DMA状态
命令
写入条目
读取条目
高速缓存读取索引
上下文高速缓存（CC）
DW0
DW4
DW8
DW12
闪存上下文0
闪存上下文1
闪存上下文2
闪存上下文3
闪存上下文4
闪存上下文5
闪存上下文6
闪存上下文7
224
数据路径接口
更新上下文状态
上下文队列控制器（CQC）
226
204
上下文“请求”
上下文“排队”指针，裸片
上下文“完成”指针状态
CPC 接口
“有效”指示符和来自 CPC 的上下文数据

图 8

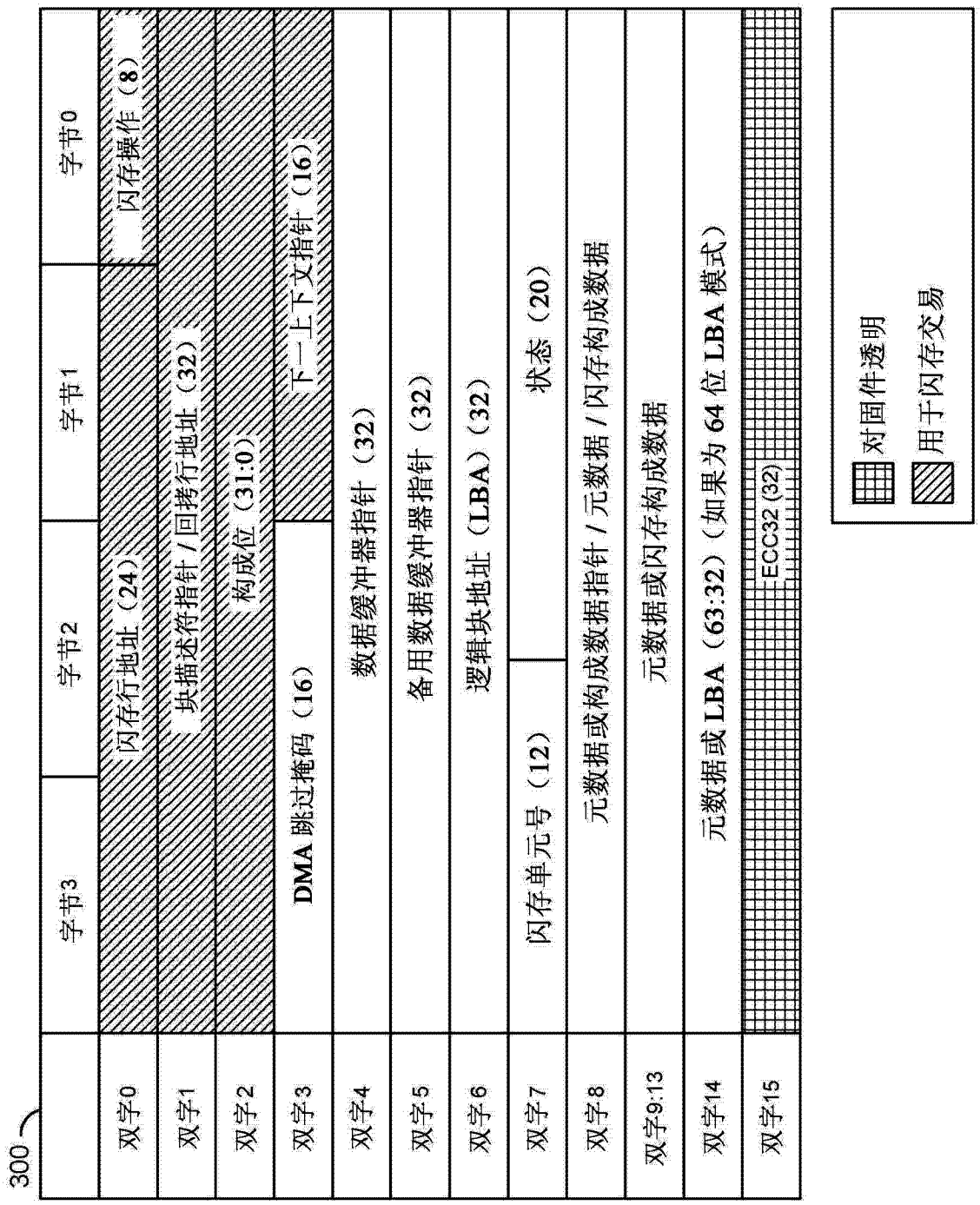

300
字节3
字节2
字节1
字节0
双字0
闪存行地址（24）
闪存操作（8）
双字1
块描述符指针 / 回拷行地址（32）
双字2
构成位（31:0）
双字3
DMA 跳过掩码（16）
下一上下文指针（16）
双字4
数据缓冲器指针（32）
双字5
备用数据缓冲器指针（32）
双字6
逻辑块地址（LBA）（32）
双字7
闪存单元号（12）
状态（20）
双字8
元数据或构成数据指针 / 元数据 / 闪存构成数据
双字9:13
元数据或闪存构成数据
双字14
元数据或 LBA（63:32）（如果为 64 位 LBA 模式）
双字15
ECC32 (32)
对固件透明
用于闪存交易

图 9