(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 105814560 B
(45)授权公告日 2019.06.18

(21)申请号 201480067463.8
(22)申请日 2014.12.02
(65)同一申请的已公布的文献号
申请公布号 CN 105814560 A
(43)申请公布日 2016.07.27
(30)优先权数据
14/100,250 2013.12.09 US
(85)PCT国际申请进入国家阶段日
2016.06.08
(86)PCT国际申请的申请数据
PCT/US2014/068228 2014.12.02
(87)PCT国际申请的公布数据
WO2015/088837 EN 2015.06.18
(73)专利权人 赛灵思公司
地址 美国加利福尼亚州
(72)发明人 M·伯洛特 L·刘 K·A·维瑟斯
(74)专利代理机构 北京市君合律师事务所 11517
代理人 顾云峰 吴龙瑛
(51)Int.Cl.
G06F 3/06(2006.01)
(56)对比文件
CN 102436420 A,2012.05.02,
US 6446141 B1,2002.09.03,
US 2010325199 A1,2010.12.23,
US 2001019509 A1,2001.09.06,
US 8700683 B2,2014.04.15,
US 2013290599 A1,2013.10.31,
审查员 邓丽婉

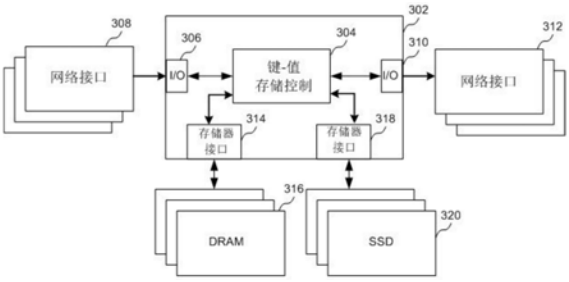
权利要求书2页 说明书11页 附图10页

(54)发明名称

用于实现高吞吐量键-值存储的存储器设置

(57)摘要

本申请描述了一种用于处理数据的电路。该电路包括:输入(306),其用于接收对实现键-值存储数据处理的请求;与不同存储类型相关联的多个存储器接口(314,318),其用于使得能够访问与键-值存储相关联的多个存储器设备(316,320);以及,用于基于数据传输标准通过多个存储器接口来控制数据路由的存储器管理电路(108,304)。



CN 105814560 B

1.一种用于处理数据的电路,其特征在于,所述电路包括:

输入,用于接收用于实现键值存储数据处理的请求,所述键值存储数据处理使得能够访问与键相关联的存储;

与不同存储器类型相关联的多个存储器接口,其使得能够访问与键值存储相关联的多个存储器设备;

第一类型的第一存储器设备,用于存储与所述键值存储数据处理相关联的存储的值的第一部分;

第二类型的第二存储器设备,用于存储与所述键值存储数据处理相关联的所述存储的值的第二部分;以及

存储器管理电路,用于基于数据传输标准控制通过所述多个存储器接口的数据路由;

其中,所述存储器管理电路使得能够访问与所述键值存储数据处理相关联的所述存储的值的第一部分,以及与所述键值存储数据处理相关联的所述存储的值的第二部分。

2.如权利要求1所述的电路,其特征在于,所述存储器管理电路控制一个数据块路由到第一数据部分和第二数据部分。

3.如权利要求2所述的电路,其特征在于,所述存储器管理电路使得能够通过第一存储器接口路由所述数据块的所述第一数据部分,并且通过第二存储器接口路由所述数据块的所述第二数据部分。

4.如权利要求3所述的电路,其特征在于,所述存储器管理电路基于访问标准将一个数据块的一部分路由至选定的存储器类型。

5.如权利要求4所述的电路,其特征在于,所述访问标准取决于所述数据块的数据的大小。

6.如权利要求4所述的电路,其特征在于,所述访问标准取决于所述数据块的数据的访问频率。

7.如权利要求3至6中任一项所述的电路,其特征在于,所述多个存储器接口中的一个存储器接口使得能够将数据路由到DRAM存储器,而所述第二存储器接口使得能够路由数据到SSD存储器。

8.如权利要求3至6中任一项所述的电路,其特征在于,通过所述第二存储器接口路由的数据被存储在多个存储器设备中。

9.一种用于处理数据的方法,其特征在于,所述方法包括:

接收用于实现与键值存储相关联的数据处理的请求,所述数据处理使得能够访问与键相关联的存储;

基于数据传输标准,控制通过多个存储器接口的数据路由,其中第一存储器类型的第一存储器设备存储与所述数据处理相关联的存储的值的第一部分,第二存储器类型的第二存储器设备存储与所述数据处理相关联的所述存储的值的第二部分;

使得能够通过第一存储器接口访问与所述键值存储相关联的所述第一存储器设备的存储器位置;以及,

使得能够通过第二存储器接口访问与所述键值存储相关联的所述第二存储器设备的存储器位置;

其中,存储器管理电路使得能够访问与所述数据处理相关联的所述存储的值的第一部

分，以及与所述数据处理相关联的所述存储的值的第二部分。

10.如权利要求9所述的方法，其特征在于，基于数据传输标准控制通过所述多个存储器接口的数据路由包括：通过所述第一存储器接口路由一个数据块的第一部分，并且通过所述第二存储器接口路由所述数据块的第二部分。

11.如权利要求9或权利要求10所述的方法，其特征在于，基于数据传输标准控制通过所述多个存储器接口的数据路由包括：根据数据的大小而通过所述第一存储器接口或所述第二存储器接口来路由数据。

12.如权利要求9或权利要求10所述的方法，其特征在于，基于数据传输标准控制通过所述多个存储器接口的数据路由包括：根据数据的访问频率而通过所述第一存储器接口或者所述第二存储器接口来路由数据。

13.如权利要求9或权利要求10所述的方法，其特征在于，使得能够通过第一存储器接口访问与所述键值存储相关联的所述第一存储器设备的存储器位置包括：将数据路由到SSD存储器。

14.如权利要求13所述的方法，其特征在于，使得能够通过第二存储器接口访问与所述键值存储相关联的所述第二存储器设备的存储器位置包括：将数据路由到DRAM。

用于实现高吞吐量键-值存储的存储器设置

技术领域

[0001] 本发明的一个或多个应用大体涉及集成电路，更具体地，涉及用于处理数据的电路与方法。

背景技术

[0002] 集成电路可以实现在设备中以用于处理数据。进一步地，可以根据不同的协议来处理数据，例如是符合数据传输标准的专有协议或公共协议。然而，不同的协议可能会要求不同的硬件元件。不同硬件元件的实现可能会影响设备的性能，例如吞吐量(throughput)、延迟和功率消耗。这些性能标准中的每一个标准在评价设备性能时仍然都很重要。也就是说，电路开发者在降低设备功率的同时，也继续在提高设备的速度。

[0003] 此外，人们为了减少实现集成电路的成本已经付出了巨大的努力。减少实现集成电路的成本的方法之一是使得一特定电路能够被用于不同的应用场合。也就是说，如果可以通过编程使一个电路可以在不同配置下运行并且实现不同的电路，那么我们就可以得到一个更节约成本的具有集成电路的设备。可以从电路改进中得益的一个数据处理领域是键-值存储(Key-value store,KVS)的实现。常规服务器可以使用其主存储器进行键-值存储，并且用网络适配器与客户端设备连接。虽然不同类型的存储器都可以用来存储数据，但是不同类型的存储器具有不同的运行参数，并且可能会影响到KVS的运行。因此，需要具有灵活度且因而可以提高数据处理性能的设备。

发明内容

[0004] 本发明描述了一种用于处理数据的电路。该电路包括：输入，其用于接收对实现键-值存储数据处理的请求；与不同存储器类型相关联的多个存储器接口，其使得能够访问与键-值存储相关联的多个存储器设备；以及存储器管理电路，其用于基于数据传输标准控制通过所述多个存储器接口的数据路由。

[0005] 用于处理数据的电路可选地可以包括：输入，其用于接收对实现键-值存储数据处理的请求；第一存储器接口，其使得能够访问与键-值存储相关联的第一类型存储器的存储器位置；第二存储器接口，其使得能够访问与键-值存储相关联的第二类型存储器的存储器位置；以及耦接在输入和第一及第二存储器接口之间的存储器管理电路，该存储器管理电路基于数据传输标准控制通过第一和第二存储器接口的数据路由。

[0006] 本发明还描述了一种用于处理数据的方法。该方法包括：接收对实现键-值存储相关联的数据处理的请求；基于数据传输标准，控制通过多个存储器接口的数据路由；使得能够通过第一存储器接口访问与键-值存储相关联的第一类型存储器的存储器位置；以及使得能够通过第二存储器接口访问与键-值存储相关联的第二类型存储器的存储器位置。

附图说明

[0007] 图1是一种用于处理数据的电路的框图；

[0008] 图2是图1所示电路的存储器接口的框图；

[0009] 图3是用于使得键-值存储能够实现多个存储器类型的电路的框图；

[0010] 图4是用于使得键-值存储能够实现多个存储器类型并且并行地使用多个接口以提高接入带宽的电路的框图；

[0011] 图5是第一类型存储器的存储器控制器的框图；

[0012] 图6是第二类型存储器的存储器控制器的框图；

[0013] 图7是示出了在实现键-值存储时，将一个数据块的数据分配到不同类型的存储器中的示意图；

[0014] 图8是动态随机存储器(DRAM)的框图；

[0015] 图9是图8中DRAM的存储器元件的框图；

[0016] 图10是固态驱动器(SSD)存储器的框图；

[0017] 图11是图9中固态驱动器的存储器元件的框图；

[0018] 图12是处理与键-值存储相关联的数据的流程图；

[0019] 图13包括了体现键-值存储操作的表格；

[0020] 图14是用于实现键-值存储的数据包的各种字段的示意图；

[0021] 图15是用于处理与键-值存储相关联的数据的流程图；

[0022] 图16是用于对有可编程资源的设备进行编程的系统的框图；

[0023] 图17是具有可编程资源的集成电路的框图；以及

[0024] 图18是图17中集成电路中可配置逻辑元件的框图。

具体实施方式

[0025] 键-值存储(KVS)是大部分数据中心中非常重要的中间件层。键-值存储的功能是通过通信或网络接口存储并取回与键相关联的值(而不是用地址索引)。近几年，键-值存储在数据中心被广泛地使用，以通过缓存最频繁访问的对象来加速它们的网络性能。因为KVS服务器通常被用来缓存大型数据库的内容，因此关于这类KVS服务器的一个重要方面就是它的存储密度，缓存越大，KVS效率越高。进一步地，KVS服务器的吞吐量和延迟也是十分重要的指标，这是因为其基本上可以决定网站的集合速度。最后，随着服务器运行费用(取决于其能耗)开始逐渐大于它的资本开支，能耗的考量也变得越来越关键。

[0026] 下述的各种电路和方法描述了固态驱动器(SSD)辅助的数据流架构，其用于在可编程逻辑或定制集成电路中实现键-值存储。在各种实施方式中，一方面通过不同类型的存储器的组合，可以获得十分高的吞吐量(例如，80Gbps)、很高的存储密度(例如，兆兆字节)以及很低的延迟与功耗。下述的这些电路和方法允许用户通过使用例如同时含有SSD和DRAM的混合存储器架构来调整需要的存储密度到多个兆兆字节，同时依然可以提供高数据吞吐量、很低的延迟以及高能效。更具体地，可以通过在DRAM和SSD之间智能分配数据以及在很多SSD中并行地存储数据来获得这种性能，从而增加接入带宽，尽管这会过量供应(overprovision)。虽然该架构既可以在可编程逻辑设备中实现，例如后面会详述的可编程门阵列(FPGA)，也可以在专用集成电路(ASIC)中实现，但是要求改进的KVS架构能提供灵活性决定了其更倾向于FPGA或者一些其他的可编程逻辑设备。

[0027] 首先参考图1，该图示出了用于处理数据的电路的框图。具体来说，网络L2-L4处理

模块102将与KVS相关联的数据提供给KVS协议处理模块104。该网络层处理模块可以依照开放系统互连(OSI)参考模型对数据进行处理，其中L2层涉及数据链路层，其定义了用于操作通信链路以及检测并纠正数据包错误的程序，L3层确定数据是如何在网络设备之间传输的，以及使得能够将数据包路由到特定的网络设备地址，而L4层涉及管理网络中端至端信息传递的运输层。虽然图1中的电路涉及到OSI参考模型并且指的是网络层L2-L4，但是需要理解的是，这里所参考的这些网络层仅为示例，并且这些电路和方法可以被实现在任意类型的数据网络中，特别是可以实现键-值存储的任意类型的数据网络。

[0028] 在存储器管理电路108的帮助下，由KVS协议处理模块104产生的数据被提供给哈希表控制器106。存储器管理电路108控制与值存储控制器110相关联的运行存储器。存储器管理电路108在值存储中分配地址页，向哈希表提供一个或多个地址，并且确定何种类型的存储器接收数据。哈希表存储器接口111使得能够访问哈希存储器(hash memory)，其例如为DRAM或SRAM，或者能够访问将在下文结合图17详述的可编程逻辑设备的BRAM，或者任意其他的存储器类型。哈希表的实现以及KVS的运行将在下文结合图12-14给出更详细的描述。值存储控制器110产生的数据被提供给KVS协议格式化电路112，其输出可以由网络L2-L4处理模块进行处理。

[0029] 值存储控制器110也会通过存储器管理电路108与多个存储器接口进行通信，该存储器管理电路108根据数据的地址将数据路由到多个存储器接口。更具体地，存储器接口类型A 120、存储器接口类型B 122和存储器接口类型n 124中的每一个都可以访问n种不同类型的存储器元件。如下文详细描述的，存储器接口类型A 120可以与第一类型存储器接口，例如DRAM。存储器接口类型B 122可以和第二类型存储器接口，该第二类型存储器与第一类型存储器不同，例如为SSD。存储器接口类型n可以是访问某种其他类型的存储器的接口，这种其他类型的存储器与前两种存储器类型不同，例如为常规的磁盘驱动器。下文将会结合图2详细说明每种存储器接口都可以与多个存储器设备进行通信。因此，图1中的电路通过选择性实现不同存储器设备来实现最高效的键-值存储，从而充分利用不同存储器设备的特点。

[0030] 更具体地，图1的电路提供了使用混合存储器架构(例如结合使用SSD和DRAM)的架构，该架构使得存储密度大幅度增加。这种混合存储器架构实现了极高吞吐量和高密度KVS，同时功耗也大大降低。进一步地，具有SSD的定制硬件数据流架构的特殊实现与传统设备相比还改进了响应时间。

[0031] 可以通过一个或多个网络接口来接收与KVS相关联的数据包。网络L2-L4处理电路102实现了处理层1-4，激活了将被处理的KVS协议层以确定进行存储操作还是取回操作，并且确定用于键-值存储操作的键和值的大小。哈希表控制器106，其通常与存储在DRAM中的哈希表相关联，被访问以确定键-值对在存储器中所处的位置。存储器管理电路108决定了在不同类型的存储器设备(例如DRAM和SSD设备)之间的数据分配，使得值存储控制器110能够在数据包被KVS协议格式化电路112基于KVS协议重新格式化并通过网络L2-L4处理电路114输出之前访问合适的存储器(例如，合适的DRAM或SSD或这两者)。

[0032] 现在参考图2，该图示出了图1中电路的存储器接口的框图。具体来说，存储器接口120-124中的每一个都可以包括与多个存储器控制器以及相应存储器设备相耦接的判优器(arbiter)202。例如，第一存储器控制器204与第二存储器设备206相耦接，第二存储器控制

器208与第二存储器设备210相耦接，第n个存储器控制器212与第n个设备214相耦接。正如下文中结合图5和图6详细描述地，每一个存储器控制器使得能够与例如DRAM或SSD的选定类型的存储器进行交互，并也有可能只针对于特定类型的存储器。下文也会描述，存储器的类型的选择是基于使得在键-值存储中实现最有效的数据处理的一个或多个标准。

[0033] 现在参考图3，图3示出了用于使得一个键-值存储可以实现多个存储器类型的电路的框图。集成电路302可以实现为具有键-值存储控制电路304，该键-值存储控制电路304与I/O模块306相接，该I/O模块306与一个或多个网络接口308耦接，用以接收数据，而I/O模块310与一个或多个网络接口312耦接以输出数据。基于图3的实现，存储器接口314使得能够与由第一多个存储器设备316构成的一个或多个存储器设备通信，这些存储器设备如图中所示为DRAM，并且第二存储器接口318使得能够与由第二多个存储器设备320构成的一个或多个存储器设备通信，这些存储器设备如图中所示为SSD。

[0034] SSD所提供的许多好处中包括较低的成本和功耗。然而，更重要的是，SSD使得键-值存储可以通过较少的引脚(例如4个引脚)连接到较大的存储密度(例如500GB存储器)。也就是说，因为SSD利用高速串行接口，例如串行高级技术附件(SATA)、串行连接SCSI(SAS)，或者外围组件互联快速(PCIe)，所以只需要较少的输入/输出(I/O)引脚。相反地，一个DRAM SODIMM需要大约140个I/O引脚来连接8GB的存储。虽然SSD的访问速度通常比DRAM慢，但是当数据被存储在合适的存储器设备中时，例如当较大的数据块被存储在SSD中时，电路的运行就可以得到改善。SSD可以是基于NAND的闪存，或者是具有高速串行接口和低I/O引脚要求的一些其它类型的闪存。

[0035] 存储器接口314和318可以例如根据图2中的存储器接口来实现。尽管DRAM和SSD被用作示例，但是需要理解的是其他类型的存储器元件也可以被实现，其中对于特定类型的存储器设备的选择是基于该存储器设备的运行参数，对此下文会给出更详细的介绍。尽管所示的第一组和第二组多个存储器设备316和320与集成电路302相分离，但是应该理解的是所示存储器设备也可以被实现在集成电路302上，并且用于实现与键-值存储相关联的不同类型的多个存储器设备的电路与方法均可被实现在单个集成电路中。

[0036] 对用来访问键-值存储的数据的一个或多个存储器元件的选择使得键-值存储的运行有效率，这是通过考虑不同存储器设备的访问特性从而根据键-值存储的不同数据来确定采用何种类型的存储器设备来实现的。根据一个实施例，存储器管理电路将一个数据块分成第一数据部分和第二数据部分，其中第一数据部分和第二数据部分被提供给不同的存储器类型。根据另一个实施例，基于访问标准例如访问速度或访问频率，存储器管理电路将数据(完整的数据块或者一部分数据块)路由到选定的存储器类型。

[0037] SSD的一个重要方面涉及到它的按块访问特征和低I/O引脚要求。尽管该访问特征对于实现键-值存储的某些应用可能是不利的，但是这个访问特征在实现键-值存储的其他应用中可能是有优势的。例如，访问比页尺寸小(例如，4千字节页大小)的SSD是十分低效的。因此，为了最大化访问性能，除了SSD之外，还提供了DRAM以将数据在DRAM和SSD之间进行分配。然而为了传输数据，DRAM对于I/O引脚有较高的要求。因此，对混合实现中存储器元件的选择需要考虑许多因素。

[0038] 根据一个实施例，在KVS中访问或存储的数据被分成2个部分。包含有多个页尺寸的值的部分被存储在SSD中，而剩下的被存储在DRAM中。因此，SSD仅以全页大小访问，从而

最大化了访问性能。尽管维持一个列表以及在DRAM和SSD之间移动值可能需要额外的开销，但是混合结构具有显著改进的性能。一个数据分割的例子将在后文结合图7给出更详细的说明。

[0039] 可替代地，尺寸小于阈值的值被存储在DRAM中，而具有较大尺寸的值则被存储在SSD中。也就是说，不同于分割一个数据块，可以根据数据块的尺寸将与键-值存储相关的数据块存储在DRAM或者SSD中。这个替代实施方式适用于一些应用，这些应用具有与DRAM密度和值密度相匹配的值大小分配。

[0040] 根据另一个实施例，在不同存储器类型之间的数据分配也可以基于访问频率。因为对于给定存储器类型的访问时间可能与其他存储器类型的访问时间不同，所以将经常访问的数据存储于具有较快访问时间的存储器类型是十分有益的。也就是说，经常访问的数据值被存储在DRAM中。这样的数据块分割解释了这样的一个实现方式，其中DRAM实际上是SSD的缓存，并且其最适用于以较高频率访问较小值的一些情况。

[0041] 现在参考图4，图4示出了一个用于使得键-值存储能够实现多个存储器类型并且可以并行地进行存储操作的电路的框图。SSD可以过量地供应，以有效减少页尺寸。根据图4中的实施例，除了多个存储器320，还提供了被示为SSD的多个存储器402。这些存储器402存储了与存储器320相同的内容，并且可以与存储器320并行地访问。尽管如所述方式并行地提供许多SSD并不会增加值存储本身的尺寸，但是由于相同内容现在可以通过多个链接访问，因此SSD的访问带宽将会增加。

[0042] 因为使用了不同存储器类型，所以需要有不同的存储器接口。图5示出了用于第一类型存储器的存储器接口的框图。举例来说，图5中的存储器控制器可以适用于SSD。图5中的存储器控制器包括用户接口502和物理接口504。用户接口502接收用来与物理接口504通信的指令(cmd)和地址(addr)信息。通过物理接口，数据被作为发送(tx)或接收(rx)数据来被通信。当被用作SSD的控制器时，发射和接收数据代表串行数据。因此，图5的电路可以用于多个存储器320和402中的每一个SSD。用于另一种存储器类型的用户接口602也可以接收用来与物理接口604通信的指令(cmd)和地址(addr)信息。除了通过物理接口604通信的数据，还可以发送控制(ctrl)、时钟(clk)和地址(addr)信号。当配有DRAM时，图6的物理接口也可以通信并行数据。

[0043] 现在参考图7，该图示出了将一个数据块的数据分配给不同类型的存储器的示意图。如前所述，通过将数据块中对应于SSD的SSD访问块大小的部分存储于一个或多个SSD并将剩余数据的存储在DRAM，可以使键-值存储的运行更为有效。也就是说，可以实现取模函数(modulo function)来确定将被存储于DRAM中的剩余的值(该值小于SSD访问块大小)。如图7所示，一个有m+1字节的数据块包括两个具有SSD块大小的块和剩余的数据。因此，这两个块中的每一个都被存储于SSD，同时剩余数据被存于DRAM。作为示例，如果一个数据块大小在8到12千字节之间(例如8千字节又10字节)，那么两个4千字节的块会被存在SSD中，而10字节会被存在DRAM中。

[0044] 现在参考图8，该图示出了一个动态随机存取存储器(DRAM)的框图。该DRAM模块400包括了多个单元802，其中每一个单元802都与一个地址译码器804耦接。地址译码器804接收地址线806，并且产生与多个存储器单元802相连的地址。地址译码器804接收芯片使能信号，并且每一个存储器单元802接收参考电压Vref。写入模块810使得能够从存储器单元

802中读取数据或将数据写入存储器单元802。具体来说，如果写入模块810被提供了写使能信号使得能够通过门814向存储器单元802写入数据，那么通过数据线812提供的数据就可以被写入到提供给地址译码器的地址。如果写使能信号禁止向存储器单元802写入数据，那么与反相器816耦接的数据会在数据线818上产生。如同下面即将详细描述的，存储在DRAM模块的单元中的数据必须被周期性地刷新以保持数据。

[0045] 现在参考图9，该图示出了图8中DRAM的存储器元件的框图。具体来说，第一晶体管902具有耦接到参考位线的漏极904和耦接到电容器908的源极906。可以通过参考字线在栅极控制该晶体管902。第二晶体管914具有耦接到位线的漏极916和耦接到电容器920的源极918。可以通过字线在栅极控制该晶体管914。通过拉高字线并且对位线施加所需存储的值以对电容器920充电可，以实现数据的写入。为了从存储器中读取数据，感测放大器926可以检测到存储在参考位线处的电容器912上的电压和存储在位线处的电容器924上的电压之间的差，从而产生一个输出值。在该位线的值被确定之后，通过将合适的电荷存储在电容器924上，可以将该值重新写入位线。也就是说，电容器924被周期性地充电以保持该单元所存储数据的正确值。虽然由于单个DRAM存储器单元相较于SRAM单元更小因而其具有优势，但是DRAM存储器单元必须被周期性刷新以维持电容器上代表了存储数据的电压，并且其相较于SRAM单元具有较长的访问时间。

[0046] 现在参考图10，该图示出了固态驱动器(SSD)的框图。该SSD包括一个带有处理器1004和缓冲电路1006的SSD控制器1002，其中的每一个都与复用器/解复用器1008相接口。复用器/解复用器电路1008从多个闪存设备1010至1016中的一个接收数据，并提供数据给多个闪存设备1010-1016中的一个。接口电路1018使得能够在一个设备的存储器接口和SSD之间传输数据，该存储器接口例如FPGA 302的存储器接口318。

[0047] 现在参考图11，该图示出了图10中的固态磁盘的存储器元件的框图。图11中的存储器元件包括衬底1102，该衬底包括源极1104和由位线控制的漏极1106。字线控制栅极元件，该栅极元件包括由氧化层1110分隔开的控制栅极1108与位于氧化层1114之上的浮动栅极1112。虽然这里以与DRAM和SSD存储器设备相关联的电路为例，需要理解的是本发明也可以在其他电路上实现。

[0048] 现在参考图12，该流程图示出了与键-值存储相关联的数据处理。根据一个实施例，电路模块1202的流水线包括实现键-值存储的模块。根据图12的电路模块1202的该流水线包括哈希计算电路1204、读取哈希表电路1206、写入哈希表电路1208和读取键-值(KV)电路1210、更新值电路1212、缓存管理电路1214、写入KV表1216、以及格式化响应数据包(response packet)模块1218。如存储器或控制电路的电路模块108的流水线外部的各种电路都可以被采用，例如存储器管理电路108或者哈希表存储器接口111。

[0049] 虽然在图12中示出了在电路模块108的流水线中的各个电路块连接的特定顺序，但是需要理解的是，所述电路模块也可以被布置为不同的顺序，而图12中所示的特定顺序仅为示例需要。此外，虽然电路模块的流水线使得能够实现键-值存储，但是电路模块的流水线也可以包括实现不同功能的其他电路模块。处理状态信息也可以作为数据包的报头的一部分，在图14中所示的字段之一进行通信，或者作为状态信息的单独报头字段进行通信。举例来说，处理状态可以向流水线的电路模块提供一系列的功能，例如(执行哈希、执行读取哈希、执行写入哈希、执行读取键等)。可替代地，状态信息也可以被作为子信道用共用的

流控制发送。

[0050] 存储器管理电路108将一个“哈希”值映射到一个键-值存储内的特定存储器地址，该“哈希”值可以容易地由键计算得到，在图13中键-值存储被显示为两个存储器元件，并且更具体地是多个存储器设备316的键-值存储存储器A和多个存储器设备320的键-值存储存储器B。如图13所示，哈希函数将所有可能的键1302映射到相关联的哈希表1304中的地址。该哈希表随后映射到实际键-值存储中的一个地址。该键-值存储的每一个值都包括一个键和一个对应的值，其中该存储的值可以被划分到多个存储器设备中。可替代地，该键可能会与在键-值存储中对应的地址一同被存储到该哈希表中，其作用为值存储。因此，该键-值存储可以通过存储有键的哈希表和存储有与键相关的值的存储器来实现。该哈希表产生用于访问存储在多个存储器设备上的数据所需的地址。如图13所示，与该键-值存储存储器A相关联的值A-1和与该键-值存储存储器B相关联的值A-2都可以基于与键A相关联的地址来访问。如图13所示，与其他键(例如键C)相关联的值也被存储在键-值存储存储器A和键-值存储存储器B中，同时与其他键(例如键I)相关联的值仅被存储在单个存储器中。

[0051] 在运行时，图1的电路通过网络接口104接收输入的请求。网络L2-L4处理块102随后对基础网络协议进行处理。通常这会包括媒体访问控制(MAC)模块，并且后跟UDP和TCP处理。网络处理模块102产生的数据包随后被传输给KVS协议处理模块104，该处理模块确定了通过处理流水线的数据包流。该处理模块需要被激活，其也可以被认为是一个处理事务状态，随后连同数据包一起被传输通过流水线。处理流水线中的每一级随后检查相关状态位，从而决定在将数据包传输到下一级之前对其进行什么处理(如果有的话)。

[0052] 因此，处理流水线包括多个模块用于计算哈希函数、读取或写入哈希表、读取或写入键-值存储、修改值、格式化响应数据包以及其他必要的基本操作。因此，所述处理流水线有效地实现了所有属于键-值存储处理协议一部分的基本操作的超集。需要注意的是，这些基本操作可以以许多不同方式实现，例如通过可编程资源、专用硬件模块或者常规的微处理器来实现。

[0053] 现在参考图14，示出了用于实现键-值存储的数据包的字段。具体地，该数据包包括协议报头部分和数据包正文。该协议报头部分有不同字段，包括幻数字段、操作码字段、键长度字段、特定处理ID和特定条目ID。该数据包正文可包括额外字段、键和值。

[0054] 现在参考图15，图15示出了处理与键-值存储相关联的数据的方法的流程图。在步骤1502，接收对实现与键-值存储相关联的数据处理的请求。在步骤1504，根据数据传输标准控制通过多个存储器接口的数据路由。在步骤1506，使得能够访问与第一存储器接口相关联的第一类型存储器的存储器位置。在步骤1508，使得能够访问与键-值存储第二存储器接口相关联的第二类型存储器的存储器位置。图15所示的方法可以通过上述图1-14中的电路实现，并且也可以在具有可编程资源的集成电路中实现，实现方法将参考图16-18在下文进行描述。

[0055] 现在参考图16，根据一个实施例，图16示出了一个用于对具有可编程资源的设备进行编程的系统的框图。具体地，计算机1602被耦接为从存储器1606接收应用设计1604，并且生成配置比特流，该配置比特流被存储在非易失性存储器1608中。如同下面即将要详细描述的，该电路设计可以是一个高层次的设计，例如由硬件描述语言(HDL)描述的电路设计。另外，该计算机可以被配置为运行用以生成配置比特流的软件，该配置比特流被存储在

非易失性存储器1608中并且被提供给集成电路1610，这个集成电路可以是可编程集成电路，例如如图17中所述的集成电路。

[0056] 如本领域技术人员所熟知的，用于实现在可编程集成电路中的电路设计的软件流包括综合、打包、布局和布线。综合步骤包括将高层次设计的电路设计转换成可编程集成电路中的元件的配置。例如，计算机运行的综合工具可以实现用于在配置逻辑块(CLB)中或数字信号处理(DSP)块中实现特定功能的电路设计的一部分。综合工具的一个例子是由位于加州圣何塞的Xilinx,Inc.所提供的ISE工具。打包步骤包括将电路设计的多个部分分组到该设备中的各个定义的模块中，例如CLB。布局步骤包括确定在打包步骤中定义的设备的模块的位置。最后，布线步骤包括在可编程集成电路中选择互连元件例如可编程互连的路径。在布局和布线结束之后，所有的功能、位置和连接都已知，随后即生成配置比特流。该配置比特流可以由软件模块BitGen生成，该软件可以由位于加州圣何塞的Xilinx.Inc提供。该比特流可以通过缆线下载或编程到EPROM中，以传输给可编程集成电路。

[0057] 现在参考图17，图17示出了一个具有可编程资源的集成电路的框图。虽然具有可编程资源的设备可以在任意类型的集成电路设备中实现，例如具有可编程资源的专用集成电路(ASIC)，但是其他设备还会包括专用可编程逻辑器件(PLD)。PLD的一种类型是复杂可编程逻辑器件(CPLD)。CPLD包括两个或多个互相连接在一起的“功能模块”，并且用以通过互连开关矩阵输入/输出(I/O)资源。CPLD的每一个功能模块都包括一个两级与/或结构，该结构与在可编程逻辑阵列(PLA)或可编程阵列逻辑(PAL)设备中使用的结构相似。PLD的另一种类型是现场可编程门阵列(FPGA)。在一个典型的FPGA中，可配置逻辑模块(CLB)的阵列被耦接到可编程输入/输出模块(IOB)。该CLB和IOB通过可编程路由资源的层级互连。通过向FPGA的配置存储器单元加载配置比特流，通常是从片外存储器，这些CLB、IOB和可编程路由资源可以根据情况而定制。对于这两种可编程逻辑设备，设备的功能可以通过以控制目的提供给该设备的配置比特流的配置数据比特来控制。配置数据比特可以被存储在易失性存储器中(例如FPGA和一些CPLD中的静态存储器单元)，非易失性存储器(例如，在一些CPLD中的闪存)中，或者任意其他类型的存储器单元中。

[0058] 图17中的设备包括一个FPGA架构1700，该架构包括大量不同的可编程单元块，这些可编程单元块包括多千兆位收发器(MGT)1701、CLB 1702、随机存取存储器模块(BRAM)1703、输入/输出模块(IOB)1704、配置和时钟逻辑(CONFIG/CLOCKS)1705、数字信号处理模块(DSP)1706、专用输入/输出模块(I/O)1707(如配置端口和时钟端口)、以及其他可编程逻辑1708，例如数字时钟管理器、模-数转换器、系统监视逻辑等等。一些FPGA也包括可以用作例如实现软件应用的专用处理器模块(PROC)1710。

[0059] 在一些FPGA中，每一个可编程单元块包括可编程互连元件(INT)1711，该元件包括去向和来自位于每一个相邻单元块中的对应互连元件的标准化连接。因此，所有可编程互连元件一起可以实现针对所示FPGA的可编程互连结构。如图17顶部所示的例子，该可编程互连元件1711也包括去向和来自位于同一个单元块中的可编程逻辑元件的连接。

[0060] 例如，一个CLB 1702可以包括可配置逻辑元件(CLE)1712，可以通过对其编程并加上单个可编程互连元件1711来实现用户逻辑。除了一个或多个可编程互连元件之外，BRAM 1703还可以包括BRAM逻辑元件(BRL)1713。该BRAM包括与配置逻辑块的分布式RAM分开的专用存储器。通常，一个单元块中包括的互连元件的数量取决于该单元块的高度。在图示的实

施例中，一个BRAM单元块与5个CLB具有相同的高度，然而这里也可以换成其他数字。DSP单元块1706除了适当数量的可编程互连元件外还包括DSP逻辑元件(DSPL)1714。除了该可编程互连元件1711的一个实例以外，IOB 1704还可以还包括例如输入/输出逻辑元件(IOL)1715的两个实例。该设备的连接的位置可以通过被提供给其的配置数据流的配置数据比特来控制。响应于配置比特流的比特，该可编程互连使得能够实现连接，该连接包括互连现，其用来将多种信号耦合到可编程逻辑中实现的电路，或者其他电路例如BRAM或处理器。

[0061] 在图示的实施例中，靠近芯片中心的列状区域被用于配置、时钟和其他控制逻辑。从此列延伸出的配置/时钟分配区域1209被用来跨越FPGA的宽度分配时钟和配置信号。一些利用如图17所示架构的FPGA包括额外的逻辑模块，该逻辑模块会破坏占FPGA大部分的规则的列状结构。该额外的逻辑模块可以是可编程模块和/或专用逻辑。例如，如图17中所示的处理器模块PROC 1710横跨CLB和BRAM的若干列。

[0062] 注意图17仅是为了说明一个示例性FPGA架构。一列中逻辑块的数量、列的相对宽度、列的数量和顺序、列中包含的逻辑模块类型、逻辑模块的相对大小以及图17顶部所示的互连/逻辑的实施例都仅仅是示例性的。例如，不论CLB在哪，在实际的FPGA中通常包括多于一个的CLB相邻列，以便于用户逻辑的有效实现。尽管图17的实施例涉及一个具有可编程资源的集成电路，但是需要理解的是下文会详述的电路和方法可以在任何类型的ASIC中实施。

[0063] 在一个实施例中，MGT 1701包括特征接收机(characterized receiver)，并且通过利用存储在BRAM 1703中的误差指示，一些CLB 1702被配置以实现眼扫描控制器和PRBS数据检查程序。

[0064] 现在参考图18，该图示出了图17中的集成电路的一个可配置逻辑元件的框图。具体地，图18以简化形式描述了图17中的配置逻辑模块1702的可配置逻辑元件。在图18的实施例中，片M 1801包括4个查找表(LUTM)1801A-1801D，每一个查找表都是通过6个LUT数据输入接口A1-A6、B1-B6、C1-C6和D1-D6驱动，并且每一个都提供两个LUT输出信号O5和O6。来自LUT 1801A-1801D的O6输出端子分别驱动片输出端子A-D。该LUT数据输入信号是由FPGA互连结构通过输入多路复用器提供的，其可以通过可编程互连元件1811实现，并且该LUT输出信号也被提供给该互连结构。片M还包括：用于驱动输出端子AMUX-DMUX的输出选择多路复用器1811A-1811D；用于驱动存储器元件1802A-1802D的数据输入端子的输出选择多路复用器1812A-1812D；组合多路复用器1816、1818和1819；反弹(bounce)多路复用器电路1822-1823；由反相器1805和多路复用器1806(这两者在输入时钟路径上提供了可选的翻转)代表的电路；并且具有多路复用器1814A-1814D、1815A-1815D、1820-1821和异或门1813A-1813D的进位逻辑(carry logic)。所有这些元件都如图18所示耦接在一起。在图18中没有示出多路复用器的选择输入的地方，这些选择输入都受配置存储器单元控制。也就是说，存储在配置存储器单元中的配置比特流的配置比特被耦合到该多路复用器的选择输入，从而为多路复用器选择正确的输入。为了清楚起见，在图18以及其他所选的图中略去了这些已知的配置存储器单元。

[0065] 在图示的实施例中，可以对每一个存储器元件1802A-1802D编程以使其用作为同步或异步触发器或锁存器。对于一个片中的所有四个存储器元件的同步或异步功能之间的选择可以通过对同步/异步选择电路1803编程实现。当对存储器元件编程以使S/R(置位/复

位)输入信号提供置位功能时,REV输入端子提供复位功能。当对存储器元件编程以使该S/R(置位/复位)输入信号提供复位功能时,REV输入端子提供置位功能。存储器元件1802A-1802D通过时钟信号CK来时钟驱动,该信号可以例如由全局时钟网络或互连结构提供。这样的可编程存储器元件在FPGA设计领域众所周知。每一个存储器元件1802A-1802D都向互连结构提供一个寄存的输出信号AQ-DQ。因为每一个LUT 1801A-1801D都提供两个输出信号O5和O6,所以该LUT可以被配置为用作为带有5个共用输入信号(IN1-IN5)的两个5输入LUT,或者作为带有输入信号IN1-IN6的一个6输入LUT。

[0066] 在图13的实施例中,每一个LUTM 1801A-1801D都可以以几种模式中的任意一种运行。在查找表模式下,每一个LUT具有6个数据输入信号IN1-IN6,这些信号由FPGA互连结构通过输入多路复用器提供。在RAM模式下,每一个LUT都可以作为单个的64位RAM或共享地址的两个32位RAM。该RAM写入数据通过输入端子DI 1(通过用于LUT1301A-1301C的多路复用器1817A-1817C)提供给64位RAM,或通过输入端子DI 1和DI2提供给两个32位RAM。在LUT RAM中的RAM写入操作受来自于多路复用器1806的时钟信号CK和来自于多路复用器1807的写使能信号WEN控制,其可以选择性通过时钟使能信号CE或通过写使能信号WE。在移位寄存器模式下,每一个LUT作为两个16位移位寄存器使用,或者与两个16位移位寄存器串联耦接以得到一个32位移位寄存器。移入(shift-in)信号通过输入端子DI1和DI2中的一个或两个来提供。16位和32位移出信号可以通过LUT输出端子提供,并且该32位移出信号也可以更直接地通过LUT输出端子MC31来提供。LUT 1801A的32位移出信号MC31也可以通过输出选择多路复用器1811D和CLE输出端子DMUX提供给用于移位寄存链(shift regi ster chaining)的通用互连结构。因此,上述电路和方法可以在一个例如图17和18中的设备或者其他合适的设备中实现。

[0067] 这里描述的一个示例性装置大体涉及一种用于处理数据的电路。该电路包括:输入,其用于接收对实现键-值存储数据处理的请求;与不同存储器类型相关联的多个存储器接口,其使得能够访问与键-值存储相关联的多个存储器设备;以及存储器管理电路,其用于基于数据传输标准控制通过所述多个存储器接口的数据路由。

[0068] 在一些这样的装置中,该存储器管理电路可以控制一个数据块路由到第一数据部分和第二数据部分。

[0069] 在一些这样的装置中,该存储器管理电路可以使得能够通过第一存储器接口路由所述数据块的第一部分,并且通过第二存储器接口路由所述数据块的第二部分。

[0070] 在一些这样的装置中,该存储器管理电路可以基于访问标准将一个数据块的一部分路由至选定的存储器类型。

[0071] 在一些这样的装置中,该访问标准可以取决于该数据块的数据的大小。

[0072] 在一些这样的装置中,该访问标准可以取决于该数据块的数据的访问频率。

[0073] 在一些这样的装置中,多个存储器接口可以包括第一存储器接口使得能够将数据路由到DRAM,和第二存储器接口使得能够将数据路由到SSD存储器。

[0074] 这里描述的另一个示例性装置大体涉及到一个用于处理数据的电路。该电路可替代性地包括:输入,其用于接收对实现键-值存储数据处理的请求;第一存储器接口,其使得能够访问与键-值存储相关联的第一类型存储器的存储器位置;第二存储器接口,其使得能够访问与键-值存储相关联的第二类型存储器的存储器位置;以及耦接在输入与第一及第

二存储器接口之间的存储器管理电路，其基于数据传输标准通过第一和第二存储器接口控制数据路由。

[0075] 在一些这样的装置中，该存储器管理电路可以通过第一存储器接口路由一个数据块的第一部分，通过第二存储器接口路由该数据块的第二部分。

[0076] 在一些这样的装置中，取决于该数据的大小，该存储器管理电路可以通过第一存储器接口或第二存储器接口路由数据。

[0077] 在一些这样的装置中，取决于该数据的访问频率，该存储器管理电路可以通过第一存储器接口或第二存储器接口路由数据。

[0078] 在一些这样的装置中，该第一存储器接口可以使得能够路由数据到DRAM存储器，该第二存储器接口可以使得能够路由数据到SSD存储器。

[0079] 在一些这样的装置中，通过该第二存储器接口路由的数据可以被存储在多个存储器设备中。

[0080] 在一些这样的装置中，该存储器管理电路可以使得能够同时访问多个存储器设备中的不同数据块。

[0081] 这里描述的一个示例性的方法大体涉及到处理数据。该方法包括：接收对实现与键-值存储相关联的数据处理的请求；基于数据传输标准，控制通过多个存储器接口的数据路由；使得能够通过第一存储器接口访问与键-值存储相关联的第一类型存储器的存储器位置；使得能够通过第二存储器接口访问与键-值存储相关联的第二类型存储器的存储器位置。

[0082] 在一些这样的方法中，基于数据传输标准，通过多个存储器接口控制数据路由可以包括通过第一存储器接口路由一个数据块的第一部分和通过第二存储器接口路由该数据块的第二部分。

[0083] 在一些这样的方法中，基于数据传输标准，通过多个存储器接口控制数据路由可以包括根据数据大小而通过第一存储器接口或通过第二存储器接口来路由数据。

[0084] 在一些这样的方法中，基于数据传输标准，通过多个存储器接口控制数据路由可以包括根据对数据的访问频率而通过第一存储器接口或通过第二存储器接口来路由数据。

[0085] 在一些这样的方法中，通过第一存储器接口使得能够访问与键-值存储相关联的第一类型存储器的存储器位置包括路由数据到SSD存储器。

[0086] 在一些这样的方法中，通过第二存储器接口使得能够访问与键-值存储相关联的第二类型存储器的存储器位置包括路由数据到DRAM。

[0087] 因此，上述电路和方法涉及用于处理数据的新的电路与方法。虽然具体参考了键-值存储，但是图1-3的电路布局也可以被延伸到其他与“大数据”相关的应用，例如查询处理器、数据扫描等等。该说明书和描述的实施例仅仅为示例，实施例的实际范围和实质应参考所附的权利要求。

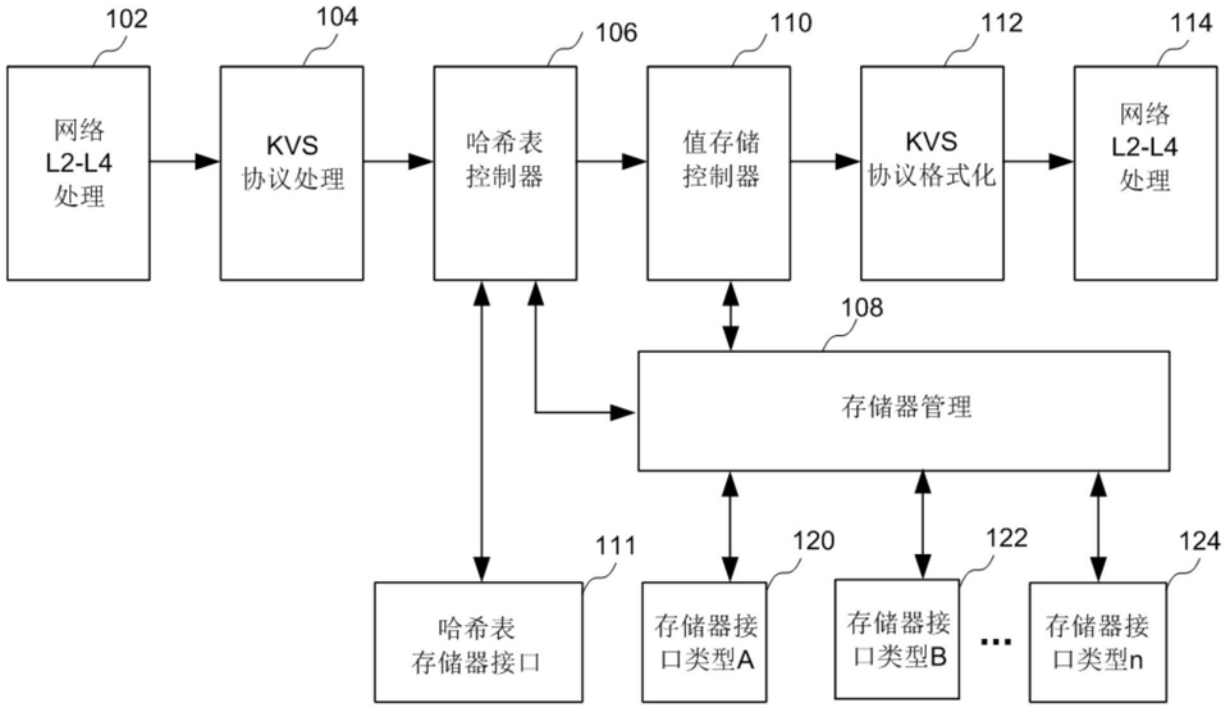

102
网络
L2-L4
处理
104
KVS
协议处理
106
哈希表
控制器
110
值存储
控制器
112
KVS
协议格式化
114
网络
L2-L4
处理
108
存储器管理
111
哈希表
存储器接口
120
存储器接
口类型A
122
存储器接
口类型B
...
124
存储器接
口类型n

图1

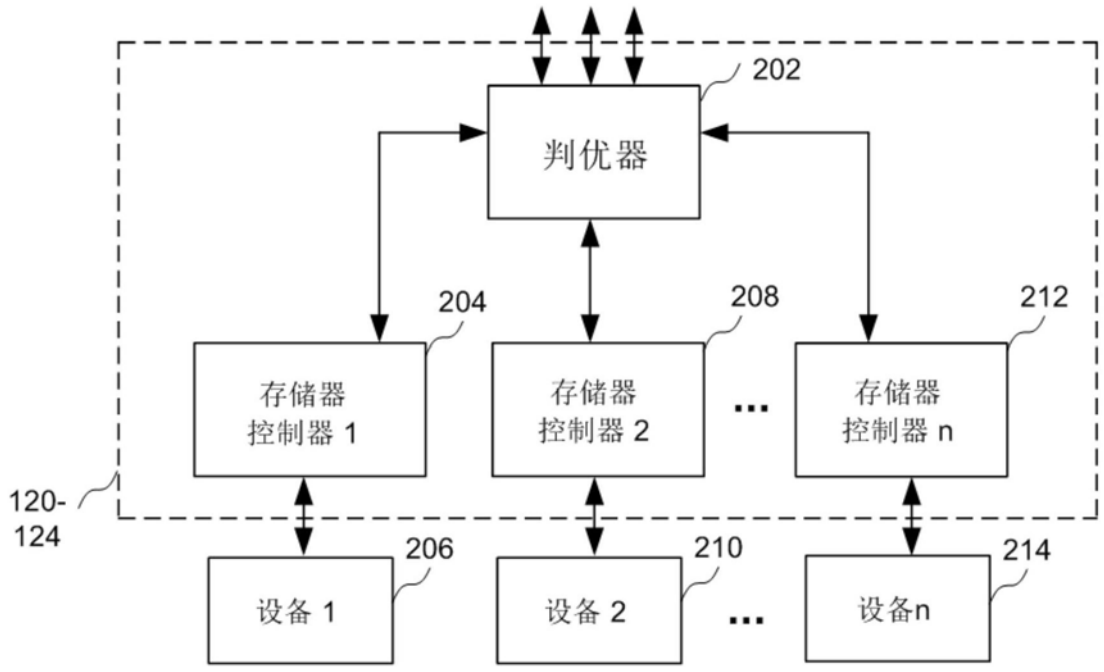

202
判优器
204
存储器
控制器 1
208
存储器
控制器 2
...
212
存储器
控制器 n
120-
124
206
设备 1
210
设备 2
...
214
设备n

图2

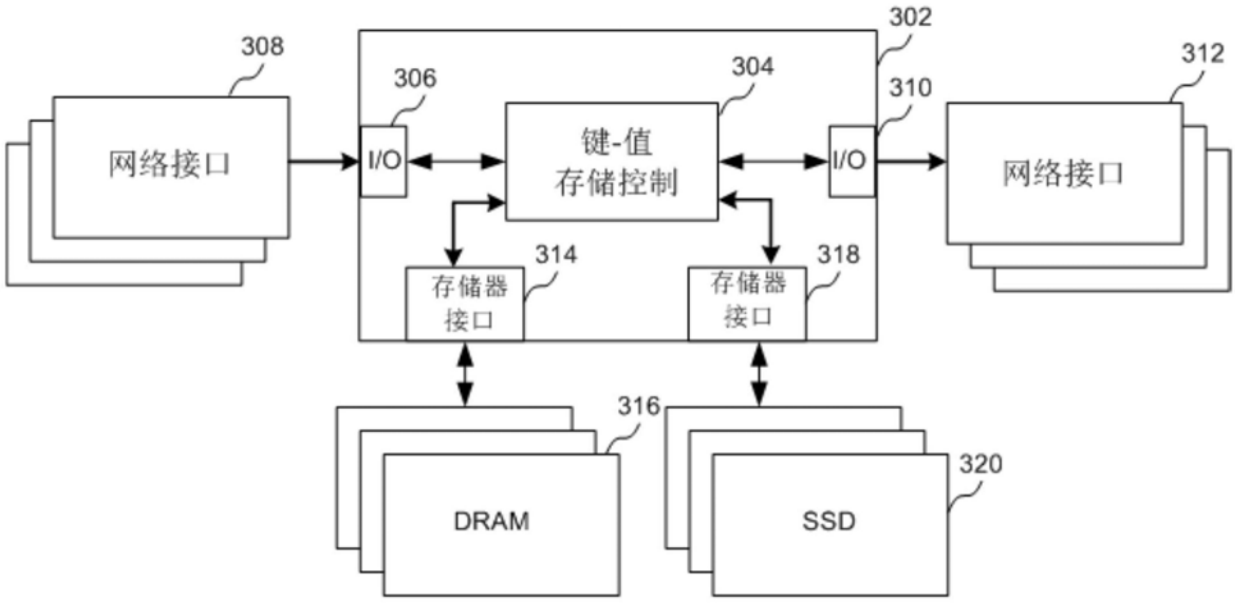

308
网络接口
306
I/O
键-值
存储控制
304
302
310
I/O
312
网络接口
314
存储器
接口
318
存储器
接口
316
DRAM
320
SSD

图3

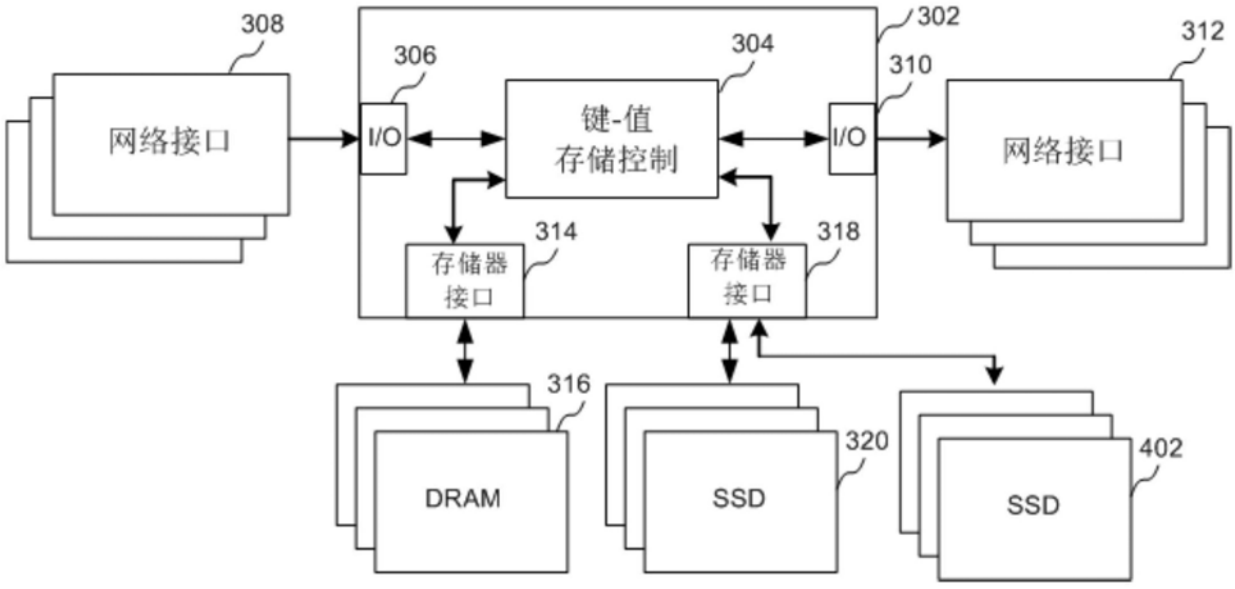

308
网络接口
306
I/O
键-值
存储控制
304
302
310
I/O
312
网络接口
314
存储器
接口
318
存储器
接口
316
DRAM
320
SSD
402
SSD

图4

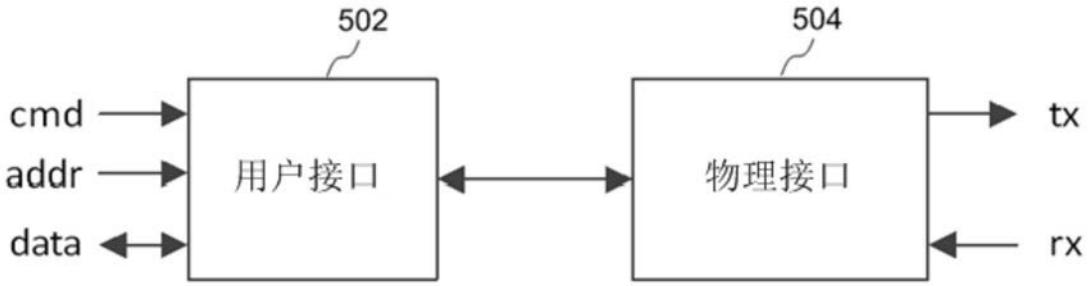

502
504
cmd
addr
data
用户接口
物理接口
tx
rx

图5

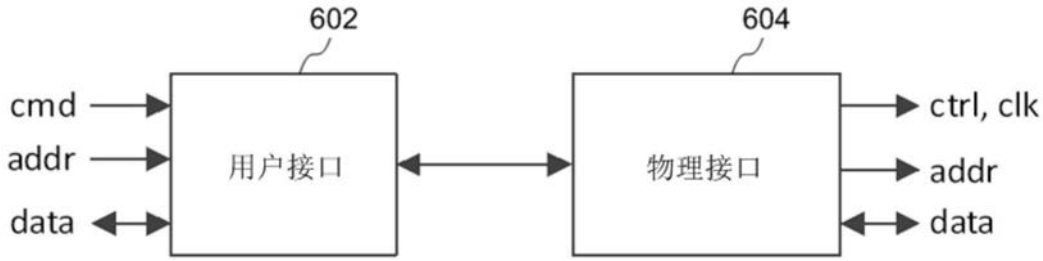

602
604
cmd
addr
data
用户接口
物理接口
ctrl, clk
addr
data

图6

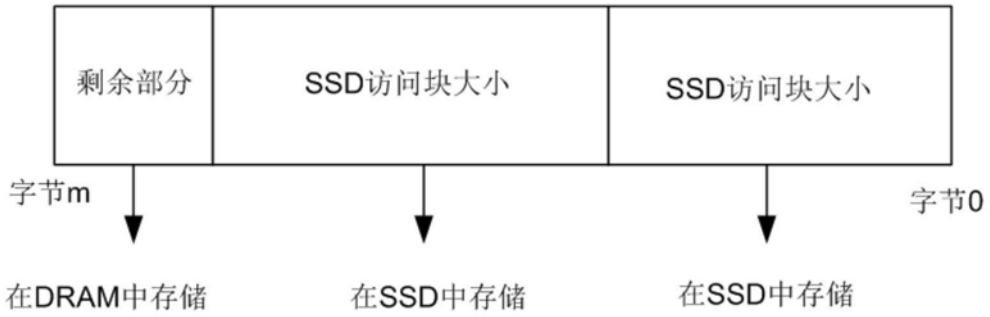

剩余部分
SSD访问块大小
SSD访问块大小
字节m
字节0
在DRAM中存储
在SSD中存储
在SSD中存储

图7

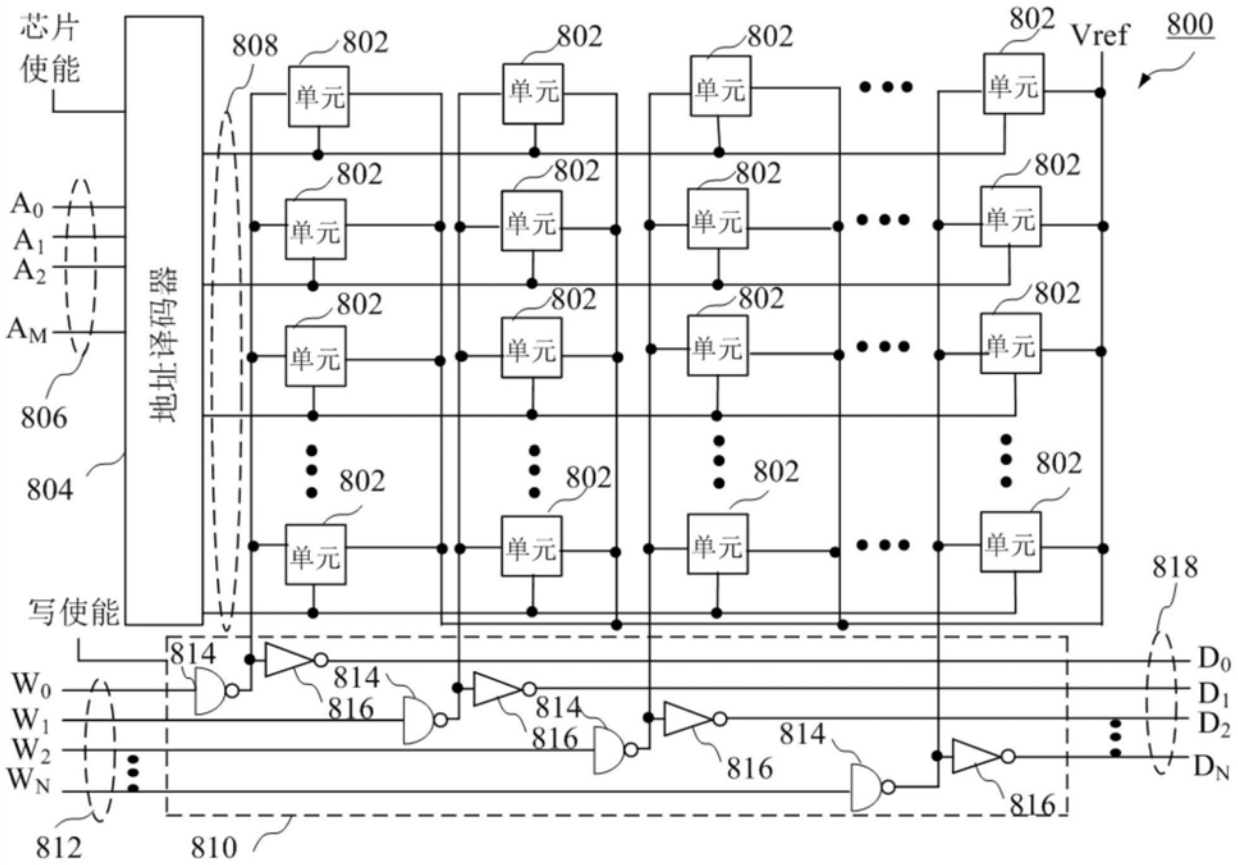

芯片使能
地址译码器
写使能
单元
802
808
804
806
812
810
814
816
818
800
Vref
A_0
A_1
A_2
A_M
W_0
W_1
W_2
W_N
D_0
D_1
D_2
D_N

图8

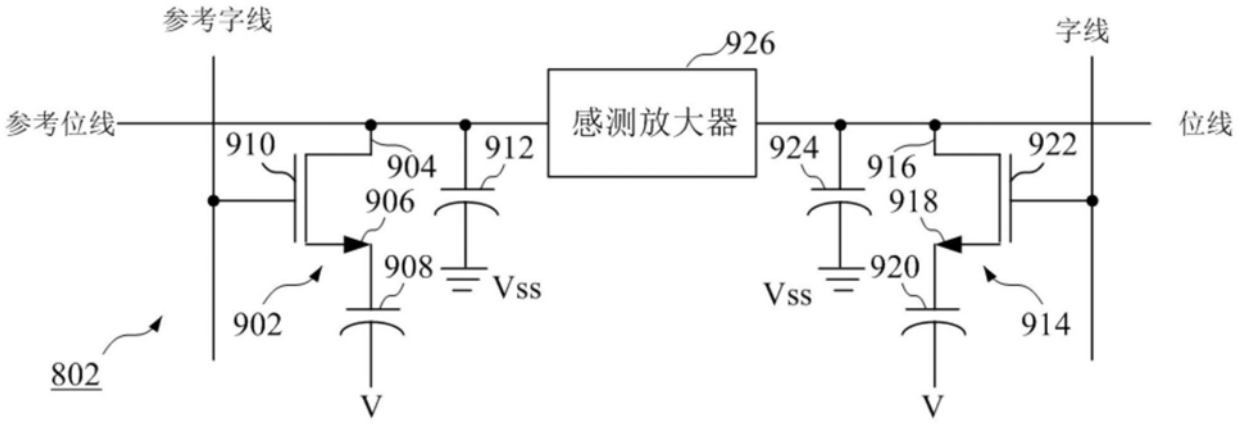

参考字线
字线
参考位线
位线
感测放大器
926
910
904
912
906
908
902
802
Vss
V
924
916
922
918
920
914

图9

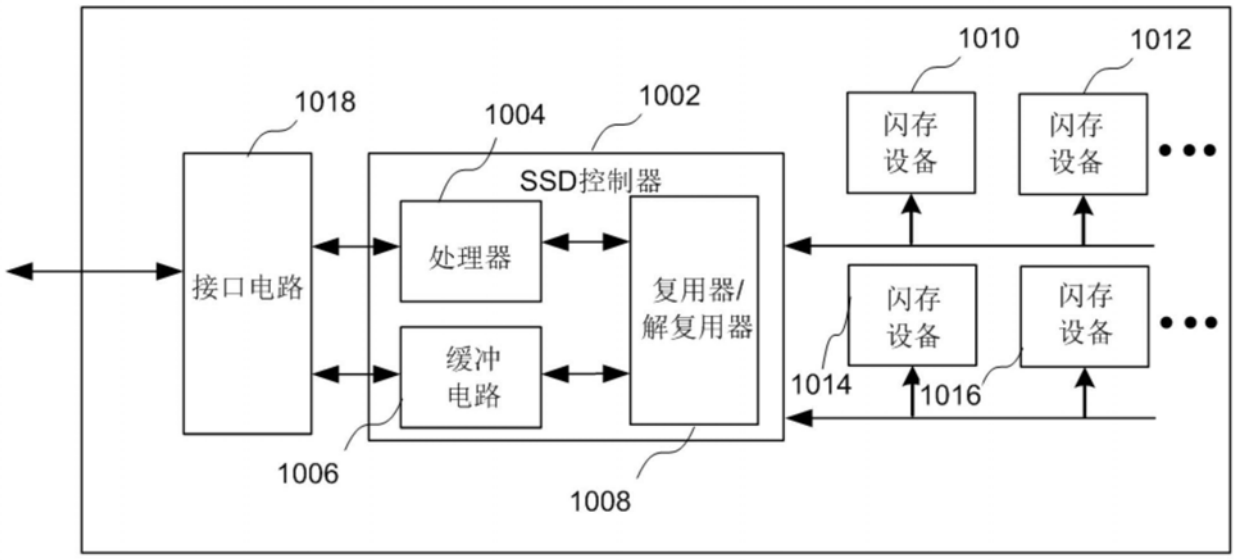

1010
1012
1018
1004
1002
闪存
设备
闪存
设备
SSD控制器
处理器
接口电路
复用器/
解复用器
闪存
设备
闪存
设备
缓冲
电路
1014
1016
1006
1008

图10

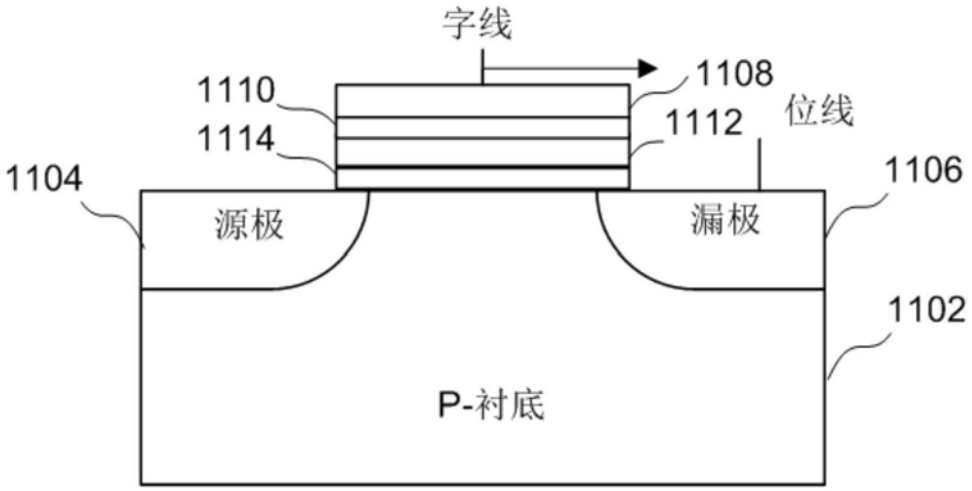

字线
1110
1108
1114
1112
位线
1104
1106
源极
漏极
1102
P-衬底

图11

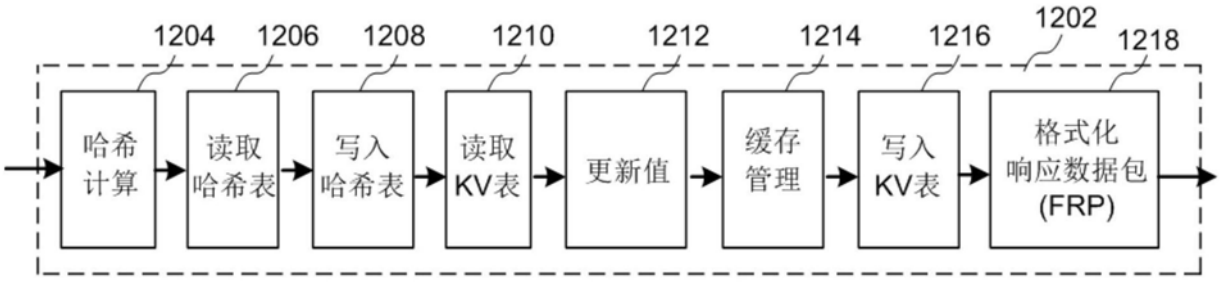

1204
1206
1208
1210
1212
1214
1216
1202
1218
哈希
计算
读取
哈希表
写入
哈希表
读取
KV表
更新值
缓存
管理
写入
KV表
格式化
响应数据包
(FRP)

图12

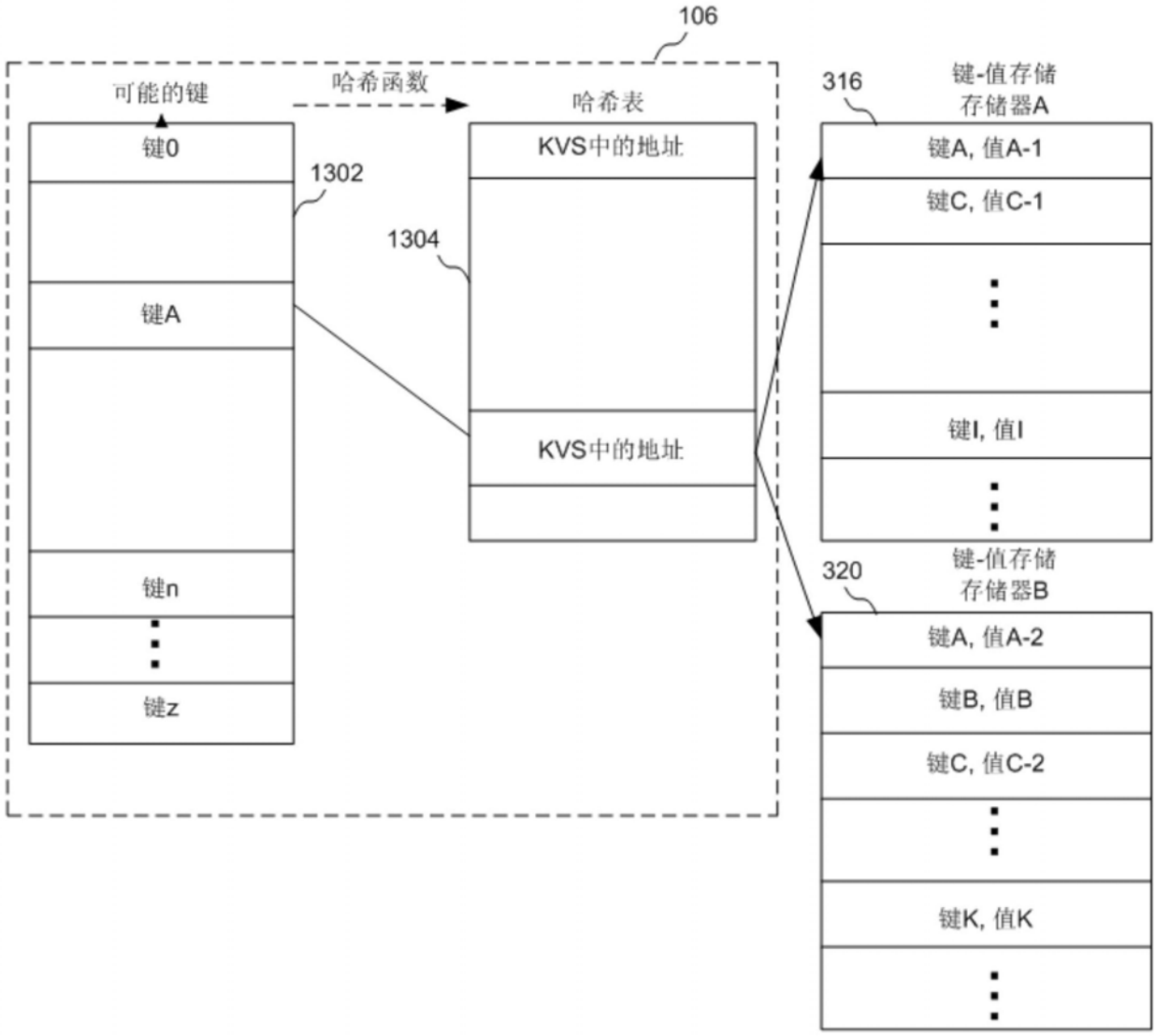

106
可能的键
哈希函数
哈希表
键0
1302
1304
键A
键n
键z
KVS中的地址
KVS中的地址
316
键-值存储
存储器A
键A, 值A-1
键C, 值C-1
键I, 值I
320
键-值存储
存储器B
键A, 值A-2
键B, 值B
键C, 值C-2
键K, 值K

图13


“幻数”
操作码 (SET/GET …)
键长度(字节)
额外长度(字节)
数据类型
状态
正文长度 (字节)
特定处理ID
特定条目ID
额外字段(例如,到期时间) (如果存在)
键 (如果存在)
值 (如果存在)
协议报头部分
数据包正文

图14

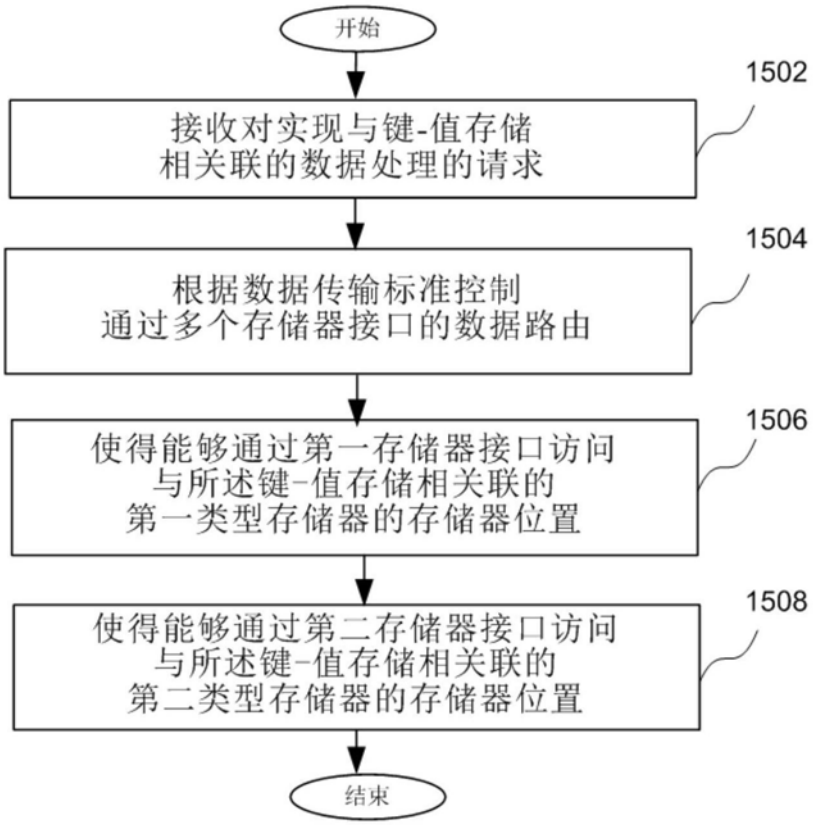

开始
1502
接收对实现与键-值存储相关联的数据处理的请求
1504
根据数据传输标准控制通过多个存储器接口的数据路由
1506
使得能够通过第一存储器接口访问与所述键-值存储相关联的第一类型存储器的存储器位置
1508
使得能够通过第二存储器接口访问与所述键-值存储相关联的第二类型存储器的存储器位置
结束

图15

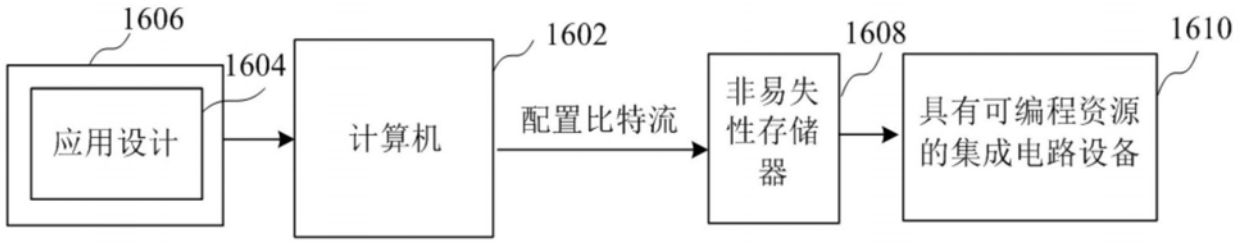

1606
1604
应用设计
计算机
1602
配置比特流
1608
非易失性存储器
1610
具有可编程资源的集成电路设备

图16

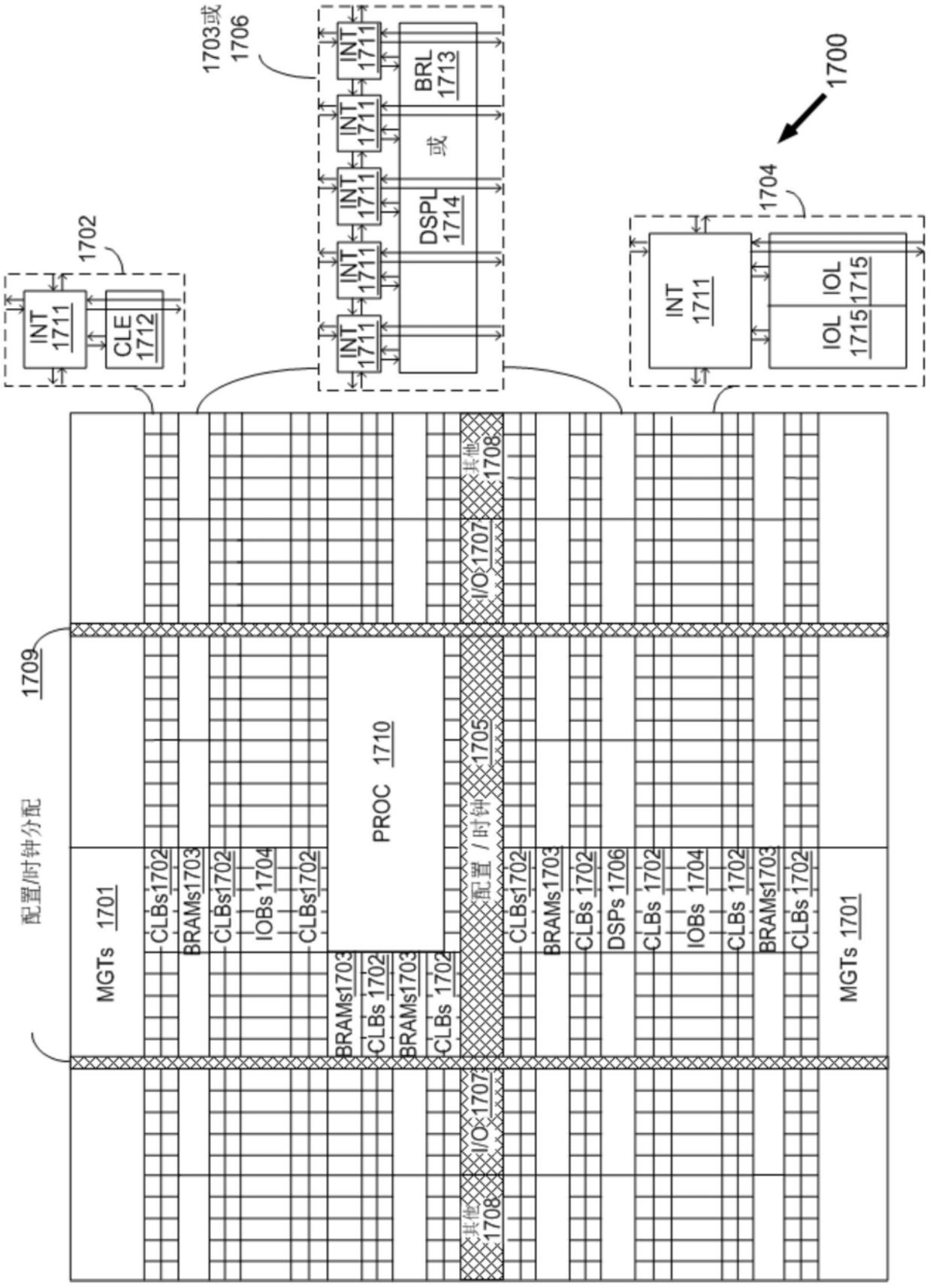

1703或1706
1700
1704
1702
INT 1711
CLE 1712
BRL 1713
或
DSPL 1714
IOL 1715
1709
配置/时钟分配
MGTs 1701
CLBs1702
BRAMs1703
IOBs 1704
PROC 1710
其他1708
I/O1707
配置/时钟1705
DSPs 1706

图17

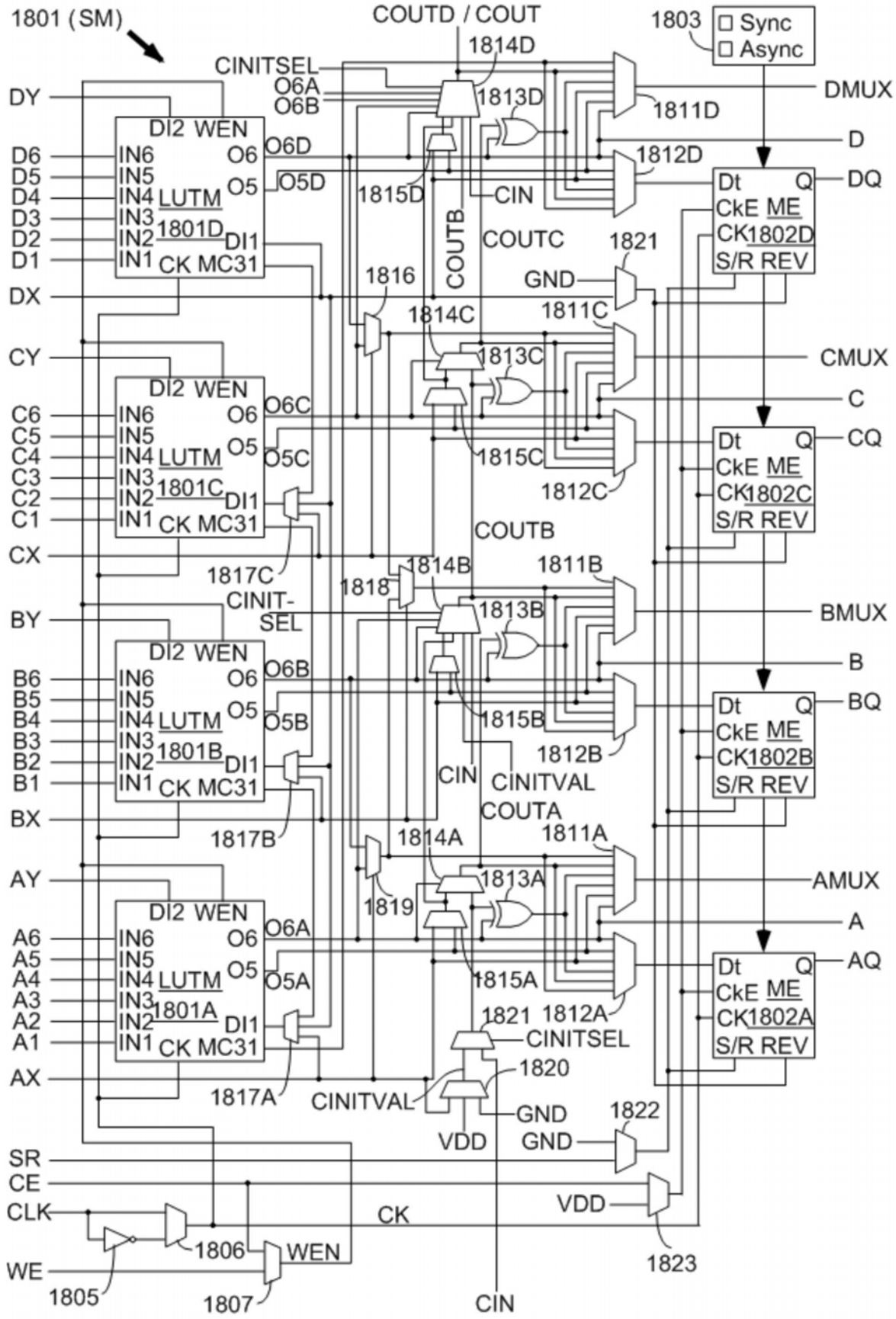

1801 (SM)
COUTD / COUT
1814D
1803
Sync
Async
CINITSEL
O6A
O6B
1813D
DMUX
1811D
DY
DI2 WEN
O6
O6D
D
D6
IN6
D5
IN5
D4
IN4 LUTM
O5
O5D
1815D
CIN
1812D
DQ
D3
IN3
D2
IN2 1801D
DI1
COUTB
COUTC
1821
D1
IN1 CK MC31
1816
GND
DX
1814C
1811C
CY
1813C
CMUX
C
C6
C5
C4
C3
C2
C1
CQ
1815C
1812C
CX
COUTB
1817C
1814B
1811B
1818
CINIT-
SEL
1813B
BMUX
BY
B
B6
B5
B4
B3
B2
B1
BQ
1815B
1812B
CIN
CINITVAL
COUTA
BX
1817B
1814A
1811A
1813A
AMUX
AY
1819
A
A6
A5
A4
A3
A2
A1
AQ
1815A
1821
1812A
CINITSEL
AX
1820
1817A
CINITVAL
GND
1822
VDD
GND
SR
CE
VDD
CLK
CK
1806
WEN
1823
WE
1805
1807
CIN

图18