



(12) 发明专利申请

(10) 申请公布号 CN 103229155 A

(43) 申请公布日 2013. 07. 31

(21) 申请号 201180056676. 7

(51) Int. Cl.

(22) 申请日 2011. 09. 23

G06F 13/14 (2006. 01)

(30) 优先权数据

G06F 13/16 (2006. 01)

61/386, 237 2010. 09. 24 US

G06F 12/00 (2006. 01)

G11C 7/10 (2006. 01)

(85) PCT申请进入国家阶段日

2013. 05. 24

(86) PCT申请的申请数据

PCT/US2011/053129 2011. 09. 23

(87) PCT申请的公布数据

W02012/040649 EN 2012. 03. 29

(71) 申请人 德克萨斯存储系统股份有限公司

地址 美国德克萨斯州

(72) 发明人 H. H. 弗罗斯特 R. 哈特赛尔

(74) 专利代理机构 北京市柳沈律师事务所

11105

代理人 高巍

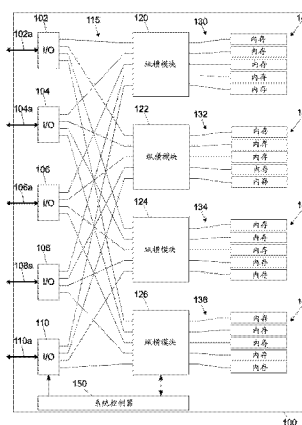
权利要求书3页 说明书20页 附图21页

(54) 发明名称

高速内存系统

(57) 摘要

本发明所公开的实施例涉及具有高速串行通信的基于闪存的内存模块。该基于闪存的内存模块包括：多个 I/O 模块，其分别被配置成通过一个或多个外部通信链路与外部装置通信、多个基于闪存的内存卡，其分别包括多个闪存式内存装置，以及多个纵横交换元件，其分别被连接至这些基于闪存的内存卡中相应的一个上，并被配置成使 I/O 模块中的每一个能够与这些基于闪存的内存卡中相应的一个进行通信。每一个 I/O 模块通过高速串行通信链路被连接至每一个纵横交换元件上，并且每一个纵横交换元件通过多个并行通信链路被连接至这些基于闪存的内存卡中相应的一个上。



1. 一种具有高速串行通信的基于闪存的内存模块,包括:

多个输入/输出(I/O)模块,每一个I/O模块被配置成通过一个或多个外部通信链路与外部装置通信;

多个基于闪存的内存卡,每一个基于闪存的内存卡包括多个闪存式内存装置,每一个闪存式内存装置具有物理内存空间,其被划分成块,每一个块进一步被划分成页面,每一个页面代表单独的可寻址内存位置,在该位置上执行内存操作,在一个块组群中可同时擦除多个这样的内存位置;以及

多个纵横交换元件,每一个纵横交换元件被连接至所述基于闪存的内存卡中相应的一个内存卡上,并且被配置成使所述I/O模块中的每一个I/O模块与所述基于闪存的内存卡中所述相应的一个内存卡进行通信;

其中,每一个I/O模块通过高速串行通信链路被连接至每一个纵横交换元件,每一个高速串行通信链路使每一个I/O模块向每一个纵横交换元件发送代表命令、指令和/或数据的比特和从每一个纵横交换元件接收代表命令、指令和/或数据的比特;并且

其中,每一个纵横交换元件通过多个并行通信链路被连接至所述基于闪存的内存卡中所述相应的一个内存卡上,每个并行通信链路将一个纵横交换元件连接至所述基于闪存的内存卡中的所述相应的一个内存卡的所述闪存式内存装置中的一个闪存式内存装置上。

2. 如权利要求1所述的基于闪存的内存模块,其中,由所述I/O模块发送的代表命令、指令和/或数据的所述比特是直接内存访问(DMA)请求的一部分,这样的DMA请求包括读取-修改-写入DMA请求。

3. 如权利要求1所述的基于闪存的内存模块,其中,所述I/O模块被配置成通过所述一个或多个外部通信链路与所述外部装置通信,所述一个或多个外部通信链路使用高速通信协议,其包括以下协议中的一个:光纤信道、无限带宽、以太网,以及前面板数据端口。

4. 如权利要求3所述的基于闪存的内存模块,其中,每个I/O模块包括高速接口,其被配置成使所述I/O模块通过所述一个或多个外部通信链路与所述外部装置通信,所述一个或多个外部通信链路使用所述高速通信协议。

5. 如权利要求4所述的基于闪存的内存模块,其中,每个I/O模块进一步包括高速控制器,其具有多个多千兆位收发器,被连接至所述I/O模块的每一个高速串行通信链路使用一个多千兆位收发器,每一个I/O模块被配置成使用所述多千兆位收发器向每一个纵横交换元件发送代表命令、指令和/或数据的所述比特,和从每一个纵横交换元件接收代表命令、指令和/或数据的所述比特。

6. 如权利要求5所述的基于闪存的内存模块,其中,每一个I/O模块进一步包括CPU和共享交换元件,所述共享交换元件被配置成调节所述CPU、经由所述一个或多个外部通信链路的所述外部装置,以及所述多千兆位收发器之间的通信。

7. 如权利要求1所述的基于闪存的内存模块,其中,每一个纵横交换元件包括高速控制器,其具有多个多千兆位收发器,被连接至所述纵横交换元件的每一个高速串行通信链路使用一个多千兆位收发器,每一个纵横交换元件被配置成使用所述多千兆位收发器向每一个I/O模块发送代表命令、指令和/或数据的所述比特,和从每一个I/O模块接收代表命令、指令和/或数据的所述比特。

8. 一种可扩展高速内存卡,包括:

印刷电路板 (PCB) ;

接口电路,其被安装到所述 PCB 上并被配置成使所述高速内存卡通过一个或多个高速串行通信链路从一个或多个外部装置接收代表指令、命令、和 / 或数据的比特 ;

多个内存装置,其被安装到所述 PCB 上,每一个内存装置具有在其中执行内存操作的物理内存空间 ;以及

控制器,其被安装到所述 PCB 上并被连接至所述接口电路和所述多个内存装置,所述控制器被配置成控制所述接口电路和所述内存装置之间的通信以执行所述内存操作 ;

其中,所述接口电路通过多个高速串行通信线被连接至所述控制器,每一个高速串行通信线对应于所述高速串行通信链路中的一个 ;并且

其中,所述控制器通过预定数量的并行通信线被连接至所述多个内存装置,所述控制器被配置成将来自所述高速串行通信链路的代表命令、指令和 / 或数据的所述比特从串行格式转换成并行格式。

9. 如权利要求 8 所述的可扩展高速内存卡,其中,所述一个或多个高速串行通信链路包括 24 个高速串行通信链路,并且其中,每一个高速串行通信链路包括两个全双工串行通信信道,每一个串行通信信道被配置成运载独立于所述其它串行通信信道的通信。

10. 如权利要求 8 所述的可扩展高速内存卡,其中,将所述控制器连接至所述多个内存装置的所述预定数量的并行通信线包括 288 个并行通信线。

11. 如权利要求 8 所述的可扩展高速内存卡,其中,所述控制器包括串行器 / 解串行器和分组模块,所述串行器 / 解串行器和分组模块被配置成依据串行格式输入来自所述接口电路的代表命令、指令和 / 或数据的所述比特,并依据并行格式重新配置代表命令、指令和 / 或数据的所述比特。

12. 如权利要求 11 所述的可扩展高速内存卡,其中,所述串行器 / 解串行器和分组模块使用驻留在所述串行器 / 解串行器和分组模块中的多千兆位收发器和存储缓存器,依据并行格式重新配置代表命令、指令和 / 或数据的所述比特。

13. 如权利要求 11 所述的可扩展高速内存卡,其中,所述控制器进一步包括多信道缓冲器,其被配置成接收通过所述串行器 / 解串行器和分组模块以并行格式被重新配置的代表命令、指令和 / 或数据的所述比特,并根据代表命令、指令和 / 或数据的所述比特构建预定数量的字,每一个字包括预定数量的比特。

14. 如权利要求 13 所述的可扩展高速内存卡,其中,所述控制器进一步包括至少一个纠错和数据保护电路,其被配置成从所述多信道缓冲器接收所述字,使用所述字生成一个或多个纠错码比特,将用于每一个字的所述纠错码比特添加到所述字中,并且输出已添加所述纠错码比特的每一个字。

15. 如权利要求 14 所述的可扩展高速内存卡,其中,所述控制器进一步包括输出缓冲器,其被连接至所述至少一个纠错和数据保护电路,所述输出缓冲器被配置成从所述至少一个纠错和数据保护电路依次接收已添加所述纠错码比特的所述字,使得来自所述至少一个纠错和数据保护电路的第一个字被提供给所述输出缓冲器中的一个输出缓冲器,并且来自所述至少一个纠错和数据保护电路的下一个字被提供给所述输出缓冲器中的另一个输出缓冲器。

16. 如权利要求 15 所述的可扩展高速内存卡,其中,所述控制器进一步包括内存缓冲

器,其被配置成从所述输出缓冲器接收已添加所述纠错码比特的所述字,并以预定的方式将预定数量的所述字组合成超字。

17. 一种具有高速串行通信的内存模块,包括:

第一多个输入处理块和第二多个输入处理块,每一个输入处理块被配置成依据串行格式接收代表命令、指令和 / 或数据的比特,并依据并行格式重新配置代表命令、指令和 / 或数据的所述比特;

多个内存装置,每一个内存装置具有在其中执行内存操作的物理内存空间;以及

控制器,其被连接至所述第一和第二多个输入处理块和所述内存装置,所述控制器被配置成控制在所述第一和第二多个输入处理块与每一个内存装置之间的通信,以执行所述内存操作,所述控制器包括:

(a) 第一多信道缓冲器和第二多信道缓冲器,其分别被连接至所述第一和第二多个输入处理块,每一个多信道缓冲器被配置成分别从所述第一和第二多个输入处理块以并行格式接收代表命令、指令和 / 或数据的所述比特,并且根据代表命令、指令和 / 或数据的所述比特构建预定数量的字,每一个字包括预定数量的比特;

(b) 第一纠错和数据保护电路和第二纠错和数据保护电路,其被分别连接至所述第一和第二多信道缓冲器,所述第一和第二纠错和数据保护电路被配置成分别从所述第一和第二多信道缓冲器接收所述字,使用所述字生成一个或多个纠错码比特,将用于每一个字的所述纠错码比特添加到所述字中,并且输出已添加所述纠错码比特的每一个字;

(c) 第一输出缓冲器和第二输出缓冲器,其分别被连接至所述第一和第二纠错和数据保护电路,所述第一和第二输出缓冲器被配置成从所述第一和第二纠错和数据保护电路依次接收已添加所述纠错码比特的所述字,使得来自所述第一和第二纠错和数据保护电路中的一个的第一个字被提供给所述第一和第二输出缓冲器中的一个,并且来自所述第一和第二纠错和数据保护电路中的另一个的下一个字被提供给所述第一和第二输出缓冲器中的另一个;以及

(d) 内存缓冲器,其被配置成从所述第一和第二输出缓冲器接收具有被添加到其中的所述纠错码比特的所述字,并以预定的方式将预定数量的所述字组合以形成超字。

18. 如权利要求 17 所述的内存模块,其中,所述控制器进一步包括判决模块,其被配置成确定每一个内存操作是 READ 操作还是 WRITE 操作,所述判决模块进一步被配置成优先处理作为 READ 操作的内存操作,除非作为 WRITE 操作的内存操作满足预定的条件。

19. 如权利要求 17 所述的内存模块,其中,所述内存模块为基于闪存的内存模块,所述基于闪存的内存模块包括一个或多个所述下列类型的闪存式内存:单级单元闪存式内存,和多级单元闪存式内存。

20. 如权利要求 17 所述的内存模块,其中,所述内存模块为基于 RAM 的内存模块,所述基于 RAM 的内存模块包括一个或多个所述下列类型的 RAM:DDR RAM、DDR2RAM,和 DDR3RAM。

高速内存系统

[0001] 相关申请的交叉引用

[0002] 本申请要求根据提交于 2010 年 9 月 24 日的第 61/386,237 号, 名称为“High-Speed Memory System”的美国临时申请的优先权的利益, 在此结合该临时申请的全部内容作为参考。

技术领域

[0003] 本公开内容一般涉及通过外部主机装置和 / 或外部通信装置访问的存储系统。

背景技术

[0004] 存储系统经常使用各种各样的方法和装置来接收入局数据(incoming data)和数据请求, 并对那些请求进行处理以存储或重新取回数据。通常这样的存储系统受限于它们的带宽, 这是因为它们的输入 / 输出端口数量有限, 以及 / 或者因为该系统中存在瓶颈。出现这样的瓶颈可能是由于使用了相对较慢的数据总线, 以及使用了复杂的转换或传输构架和 / 或协议。

[0005] 据此, 需要一种更有效、更高效且优化的高速内存系统。

发明内容

[0006] 所公开的实施例涉及用于提供更有效、更高效和优化的高速内存系统的方法和设备。通常, 一方面, 所公开的实施例涉及具有高速串行通信的基于闪存的内存模块。这种基于闪存的内存模块包括: 多个输入 / 输出(I/O) 模块, 每一个 I/O 模块被配置成通过一个或多个外部通信链路与外部装置通信; 多个基于闪存的内存卡, 每一个基于闪存的内存卡包括多个闪存式内存装置, 每一个闪存式内存装置具有物理内存空间, 其被划分成块, 每一块进一步被划分成页面, 每一个页面代表了可单独寻址的内存位置, 内存操作在该位置上执行, 在一个块组群中的多个这样的内存位置可同时被擦除, 以及, 多个纵横交换元件(crossbar switching element), 每一个纵横交换元件被连接至基于闪存的内存卡中相应的一个内存卡上, 并被配置成使得 I/O 模块中的每一个都与该相应的一个基于闪存的内存卡通信。每一个 I/O 模块都通过高速串行通信链路被连接至每一个纵横交换元件上, 每一个高速串行通信链路使得每一个 I/O 模块都向每一个纵横交换元件发送代表命令、指令和 / 或数据的比特和从每一个纵横交换元件接收代表命令、指令和 / 或数据的比特, 并且, 每一个纵横交换元件都通过多个并行通信链路被连接至基于闪存的内存卡中相应的一个上, 每一个并行通信链路将一个纵横交换元件连接至基于闪存的内存卡中的相应的一个内存卡的闪存式内存装置中的一个上。

[0007] 通常, 另一方面, 所公开的实施例涉及可扩展高速内存。该可扩展高速内存包括: 印刷电路板(PCB); 接口电路, 其被安装在 PCB 上并被配置成使高速内存卡通过一个或多个高速串行通信链路从一个或多个外部装置接收代表指令、命令和 / 或数据的比特; 多个内存装置, 其被安装在 PCB 上, 每一个内存装置具有物理内存空间, 在其中执行内存操作, 以

及,控制器,其被安装在 PCB 上并被连接至接口电路和多个内存装置,该控制器被配置成控制接口电路和每一个内存装置之间的通信以执行这些内存操作。接口电路通过多个高速串行通信线被连接至控制器,每一个高速串行通信线对应于高速串行通信链路中的一个,并且该控制器通过预定数量的并行通信线被连接至多个内存装置,该控制器被配置成将来自高速串行通信链路的代表命令、指令和 / 或数据的比特从串行格式转换成并行格式。

[0008] 通常,再一方面,所公开的实施例涉及具有高速串行通信的内存模块。该内存模块包括:第一多个输入处理块和第二多个输入处理块,每一个输入处理块被配置成依据串行格式接收代表命令、指令,和 / 或数据的比特,并依据并行格式重新配置这些代表命令、指令,和 / 或数据的比特;多个内存装置,每一个内存装置具有物理内存空间,在其中执行内存操作,以及,控制器,其被连接至第一和第二多个输入处理块和内存装置,该处理器被配置成控制该第一和第二多个输入处理块与每一个内存装置之间的通信以执行内存操作。该控制器包括:(a) 第一多信道缓冲器和第二多信道缓冲器,其分别被连接至第一和第二多个输入处理块,每一个多信道缓冲器被配置成分别从第一和第二多个输入处理块以并行格式接收代表命令、指令和 / 或数据的比特,并将根据这些代表命令、指令,和 / 或数据的比特构建预定数量的字,每个字包括预定数量的比特;(b) 第一纠错和数据保护电路以及第二纠错和数据保护电路,其分别被连接至第一和第二多信道缓冲器,该第一和第二纠错和数据保护电路被配置成分别从第一和第二多信道缓冲器接收字,用这些字生成一个或多个纠错码比特,将用于每一个字的纠错码比特添加到该字中,并输出已添加纠错码比特的每一个字;(c) 第一输出缓冲器和第二输出缓冲器,其分别被连接至第一和第二纠错和数据保护电路,该第一和第二输出缓冲器被配置成从第一和第二纠错和数据保护电路依次接收具有已添加纠错码比特的字,使得来自第一和第二纠错和数据保护电路中的一个的第一个字被提供给第一和第二输出缓冲器中的一个,并且来自第一和第二纠错和数据保护电路中另一个的下一个字被提供给第一和第二输出缓冲器中的另一个,以及 (d) 内存缓冲器,其被配置成从第一和第二输出缓冲器接收已添加纠错码比特的字,并以预定方式将预定数量的字进行组合以形成超字(super-word)。

附图说明

[0009] 图 1 示出了依据本公开内容的一定的教导而构建的高速存储系统;

[0010] 图 1A 至图 1C 示出了 DMA 读取 - 修改 - 写入操作的示例,其可在本公开内容中的高速存储系统中执行;

[0011] 图 2 示出了 I/O 模块的示例性实施例,其可被用于本公开内容中的高速存储系统;

[0012] 图 3 示出了 I/O 模块的可替换示例性实施例,其可被用于本公开内容中的高速存储系统;

[0013] 图 4 示出了用于实现在以上结合图 1 说明的该类纵横模块的一个示例性手段;

[0014] 图 5 示出了内存卡的示例性实施例,其可被用于本公开内容中的系统;

[0015] 图 6 示出了内存卡的可替换实施例,其可被用于本公开内容中的高速存储系统;

[0016] 图 7 示出了可替换的高速存储系统,其中 I/O 模块与内存板直接通信,而未使用纵横模块;

- [0017] 图 8 示出了高速内存板的示例性实施例,其可被用于图 7 中的高速存储系统;
- [0018] 图 9 更详细地示出了示例性高速内存板的部件;
- [0019] 图 10A 示出了用于高速内存板的串行器 / 解串行器和分组模块的示例性实施例;
- [0020] 图 10B 至图 10D 示出了用于高速内存板的先入 - 先出 (FIFO) 存储缓冲器的示例性实施例;
- [0021] 图 11 示出了用于高速内存板的多信道 FIFO 缓冲器的示例性实施例;
- [0022] 图 12 和图 12A 示出了将数据从纠错和数据保护电路移至高速内存板的输出 FIFO 缓冲器的示例性处理;
- [0023] 图 13A 和图 13B 说明了用于通过多个单独的高速串行通信链路转移数据的示例性处理;
- [0024] 图 14A 至图 14D 示出了用于在高速串行通信链路上转移数据时,避免来自不同的写入操作的数据不适当地“混合”的示例性处理;
- [0025] 图 15 示出了本公开内容中的高速存储系统中,用于在访问在高速串行通信链路上移动的数据的 READ (读取) 和 WRITE (写入) 之间进行裁定的示例性处理;
- [0026] 图 16 示出了在本公开内容中的高速存储系统中的示例性控制器,以及在该控制器和物理 RAM 内存之间的连接;
- [0027] 图 17 示出了在本公开内容中的高速存储系统中的两个基本内存周期内对图 16 中的控制器的示例性操作。

具体实施方式

[0028] 以上说明的附图和下面对特定构造和功能的文字说明并不体现为限制本申请所创造出的发明或随附的权利要求书的范围。而是,提供这些附图和文字说明是用于教导本领域任何技术人员以制造和使用这些寻求专利保护的发明。那些本领域技术人员将体会到,为了清楚和便于理解的原因,并未说明或示出了本发明的商用实施例的所有特征。本领域技术人员还将体会到,对结合本发明各方面的实际商用实施例的研发将需要做出许多具体实施决策,以达到研发者对该商用实施例的最终目标。这样的具体实施决策可包括,且很可能并不限于,符合系统相关的、商业相关的、政府相关的以及其它的规定,其可能根据具体的实施方式、位置以及时间变迁而改变。虽然研发者的努力可能在绝对意义上是很复杂并且很耗时的,然而,这样的努力对于那些获得本公开内容利益的本领域技术人员是日常工作。必须理解的是,本文所公开和教导的发明可具有大量的且各种各样的变形和可替换形式。最后,单数性术语,例如但不限于“某一个”,的使用并非旨在限制对象的数量。而且,在本文字性说明书中的关系性术语,例如但不限于“顶部”、“底部”、“左”、“右”、“上部”、“下部”、“向下”、“向上”、“侧面”等都是为了具体参考附图时清楚的目的而使用的,而非意图限制本发明的或随附的权利要求书的范围。

[0029] 示例性存储系统

[0030] 转向附图并尤其参考图 1,示出了依据本公开内容的一定的教导而构建的高速存储系统 100。通常,该高速存储系统 100 从外部主机装置接收数据相关的请求,诸如 READ 和 WRITE 请求,并处理那些请求,以向物理内存存储数据和 / 或从物理内存重新取回数据。

[0031] 该示例性存储系统 100 包括并使用了多个输入 / 输出模块、高速串行通信信道、可

配置纵横模块、并行内存总线,和物理内存卡,以提供连接的高带宽存储系统。在下面提供了该系统的额外的细节及其许多新颖的方面。

[0032] 参考图 1,系统 100 包括多个输入 / 输出 (I/O) 接口模块 102、104、106、108 和 110。在该示例中,示出了五个 I/O 模块,但是 I/O 模块的数量并不是一定的,而是可以改变的。每一个 I/O 模块经由主机通信信道(分别为 102a、104a、106a、108a 和 110a) 被耦接至一个或多个外部主机装置。这些信道 102a、104a、106a、108a 和 110a 使得外部装置 - 诸如服务器或可作为主机操作的任意其它的装置 - 向 / 从该 I/O 接口模块 102、104、106、108 和 110 提供和 / 或接收命令、指令和数据。

[0033] 这些主机通信信道可采用不同的形式并且附带不同的协议,这些协议包括但不限于:光纤信道(Fibre Channel)、无限带宽(InfiniBand)、以太网,和前面板数据端口(FPDP)。虽然由本公开系统提供的好处具有完全的优势,但是这些通信链路的精确的物理和逻辑构架并不一定是所公开的系统这样的。可优选的是,这些主机通信链路能够支持高带宽数据传输。

[0034] 在图 1 的示例中,I/O 模块 102、104、106、108 和 110 中的每一个都通过多个高速串行通信信道(在图 1 中全部标记为 115)被耦接至四个可配置纵横模块 120、122、124 和 126。这些信道使 I/O 模块将命令、指令和 / 或数据发送至纵横模块,正如下面详细说明书的。

[0035] 在所示出的实施例中具有四个纵横模块 120、122、124 和 126,但是可以体会到的是,纵横模块的数量并不是一定的,也可以是不同的。在图 1 中,四个纵横模块中的每一个经由分别的串行系统通信信道被耦接至五个 I/O 模块中的每一个上。可以设想到,在可替换的实施例中,可将一个或多个纵横模块耦接至全部 I/O 模块中较少的几个上。

[0036] 如图 1 所反映出的,在该示例性实施例中,每一个纵横模块都通过多个并行内存总线被耦接至成组的内存卡上。例如,在所示出的示例中,纵横模块 120 通过第一组并行内存总线 130 被耦接至第一组内存卡 140。通过类似的方式,纵横模块 122、124 和 126 分别经由并行内存总线组群 132、134 和 136 被耦接至内存卡组群 142、144 和 146 上。在所示出的示例中,每一个纵横模块通过五个多路并行总线被耦接至一组五个内存卡,其可分别被用于通信控制、命令,或监控信息,以及传输数据。将体会到的是,内存卡的数量以及内存接口总线的类型可能不同,而不背离本公开内容的教导。

[0037] 在所示出的示例中,每一个内存卡,并且因此每一组内存卡,将一般地对应于特定范围的存储位置。该特定范围可以是实际的物理地址范围(例如,在该地址范围中,每一个地址对应于与该内存卡相关联的特定物理存储位置),或者可以是逻辑地址范围(例如,在该逻辑地址范围中,每一个逻辑地址可通过内存卡上的控制器或者外部控制器映射到与该内存卡相关联的物理存储位置)。

[0038] 在图 1 的示例中,总体系统 100 将向外部主机提供许多可用的存储地址,并且每一个内存卡将通常对应于由该系统 100 所提供的特定范围的地址。此外,因为每一组内存卡与特定范围的地址相关联,所以每一个纵横块(crossbar)也与对应于其连接的该组内存卡的地址范围相关联。

[0039] 在图 1 的系统 100 中,系统控制器 150 被耦接至 I/O 模块 102、104、106、108 和 110 中的每一个以及纵横模块 120、122、124 和 126 中的每一个上。虽然在图 1 中该系统控制器连接至全部五个 I/O 模块,但是仅示出了一个这样的连接。同样地,在系统控制器和纵横模

块之间的四个连接中仅示出了一个连接。通常,系统控制器 150 提供了定时和同步信号,但是其也可被用于执行监控、报告,以及其它监管任务。

[0040] 在一般的操作中,每一个 I/O 模块通过其主机通信链路从连接该 I/O 模块的外部主机装置接收外部数据相关的请求。每一个 I/O 模块处理这些数据相关的请求,以能够向存储系统 100 存储数据和 / 或从存储系统 100 重新取回数据。在图 1 的实施例中是以不同的通信“分组”形式接收数据请求并且返回响应的,但是也可使用可替换的通信协议。

[0041] 虽然精确的协议细节取决于应用,但是应该针对低通信开销选择外部协议,这有利于高数据吞吐量。在图 1 的系统中, I/O 模块接收和处理的通信分组典型地包括头部(header)以及,如果该分组要传送数据,则包括数据有效载荷(payload)。虽然头部的准确内容和格式根据不同应用而不同,但是头部将典型地包括:(a) 指示请求类型的标识符(例如,该请求是要将数据存储在系统中的 WRITE 请求,还是要重新取回先前被存储在系统中的数据的 READ 请求);(b) 指示与该请求相关联的特定 ADDRESS 的标识符(例如,特定的逻辑或物理“地址”,其对于 WRITE 请求将通常标识出与该请求相关联的数据存储的开始位置,或者,对于 READ 请求为重新获取与该请求相关联的数据的起始位置 - 这些地址中的每一个有时被称为“TARGET ADDRESS (目标地址)”) ;以及,对于一些系统,(c) 对该数据相关的请求的源的识别(有时被称为“SOURCE ADDRESS (源地址)”) ;以及,(d) 指示与该请求相关联的数据量的标识符(例如,对将作为该请求的一部分被存储到系统或从系统重新取回的字节数、页面数或其它数据量的指示)。对于一些通信协议,将所有外部数据相关的请求推断为与数据的特定量相关联,诸如特定大小的数据页面,或特定数量的数据字。在这样的协议中,可能不需要具体指明与特定外部请求相关联的数据量,因为对于该特定协议,该请求涉及的数据的标准数量将隐含在内。

[0042] 正如外部装置向它们附带的 I/O 模块发出数据相关的请求那样, I/O 模块也向它们连接的纵横模块以及它们附带的内存卡发出请求,这被称为 DMA (direct memory access, 直接内存访问) 请求,简称为 DMA。每一个外部请求可要求访问一定范围的系统地址,这些地址跨越多个内存卡和纵横模块。即使特定的外部请求仅要求访问一个内存卡,与该外部请求相关联的数据量也可能超过与单个 DMA 请求相关联的最大数据量。由于这个原因, I/O 模块可针对其接收的每一个外部请求发出多于一个的 DMA 请求。在这种情景下, DMA 请求以这样的方式被引导至内存卡,即:在该系统内的正确的物理位置处进行数据存储或重新读取。在外部 READ 请求的情况下,例如, I/O 模块将发出一个或多个 DMA READ 请求,以便从其被存储的不同位置“收集”被请求的数据。在外部 WRITE 请求的情况下, I/O 模块一般将发出一个或多个 DMA WRITE 请求,以便将所有被接收的数据“传递”至其应该被存储的不同的位置。

[0043] 通常, DMA WRITE 请求被整合到“分组”中,该请求及其相关联的数据驻留在同一个分组内。另一方面, DMA READ 请求通常仅包括请求信息(地址、数据量等)。对这些请求类型中每一种请求的响应是大致的补充。对于由 I/O 模块发出的每一个 DMA WRITE 请求,纵横模块将返回 DMAWRITE 响应分组,其最低限度包括指示 WRITE 操作成功或失败的状态信息。对于由 I/O 模块发出的每一个 DMA READ 请求,纵横模块将返回 DMAREAD 响应分组,其包含所请求的数据,以及,最低限度包含指示所返回数据有效性的状态信息。

[0044] 在一个实施例中,系统内的内存卡中的每一个都采用了闪存式内存,每一个 DMA

请求指定了对固定量数据的传输,该固定量等于被存储在单个闪存式内存“页面”中的数据量。在该示例中,页面一般是可被分散地写入该闪存式内存的最小信息量。

[0045] 在图 1 的系统 100 中,DMA 请求通常被定义成向与固定的地址边界对准的地址和从与固定的地址边界对准的地址传输固定的数据量(4Kbytes)。在一些情况下,外部 WRITE 请求所跨越的地址范围有可能并不与这些固定的 DMA 请求地址边界对准。如图 1A 至图 1C 所示,外部 WRITE 请求可能在并不对应于所允许的 DMA 请求的起始地址的地址处开始。类似地,DMA WRITE 请求可能在并不对应于 DMA 请求的结束地址的地址处结束。当出现这种“不对准”时,I/O 卡可执行一次或多次读取-改变-写入(RMW)操作。在一次读取-改变-写入操作中,I/O 模块发出 DMA READ 请求,以从系统内存中从跨外部 WRITE 请求的开始和/或结尾的地址范围中重新取回数据。然后 I/O 模块使用来自未对准的 WRITE 请求的外部数据改变(重置)一部分被重新取回的数据,并且然后通过发出补充 DMA WRITE 请求,将改变后的数据返回系统内存。通过这种方式,I/O 模块能够将数据存储至系统内存中没有符合单独 DMA 请求的对准限制的地址范围处。

[0046] 图 1A 示出了“包含在内”的情况的示例,在这种情况下,外部 WRITE 请求指定完全被包含在一个页面之内的地址和数据量 160,但是这将导致读取-改变-写入操作被执行,因为起始地址并不与 DMA 地址边界对准。在这种情况下,仅需要一个 DMA READ 请求。使用与 DMA READ 请求相同(对准)的地址发出的 DMA WRITE 请求代替先前存储的数据覆盖新(被改变)的数据。图 1B 示出了具有跨两个闪存页面的指定地址和数据量 170 的外部 WRITE 请求。这种“跨地址”的情况将产生两个重新取回数据的 DMA READ 请求,并随后产生两个 DMA WRITE 请求以返回新的数据图像。图 1C 示出了外部 WRITE 请求,该外部 WRITE 请求的指定地址和数据量 180 横跨了三个页面。在这种“横跨”的情况下,I/O 卡将发出针对第一页面的 DMAREAD 请求,覆盖新数据,并发出 DMA WRITE 请求以适应这些改变。由于下一个页面将完全被新的数据改写,I/O 卡将无需在该页面上执行读取-改变-写入操作,并将简单地向该对准的地址发出 DMA WRITE 请求。但是,新数据集剩下的数据并未完全填满第三页面。因此,I/O 卡将以前述方式针对该页面执行读取-改变-写入。将体会到的是,这三种情况仅是可能进行操作的外部 WRITE 请求操作类型的子集,并且仅仅旨在说明读取-改变-写入操作。

[0047] 虽然外部请求的结构通常是取决于协议的,但是对于特定的系统,内部 DMA 请求/响应协议通常是固定的,并且被设计成使系统性能最大化。在图 1 的系统中,DMA 请求和 DMA 响应被构造成分组那样,包括头部和,如果该分组传输数据,则包括数据有效载荷。

[0048] DMA 请求典型地包括:(a) 指示 DMA 请求类型的标识符(即,DMA 请求是要将数据存储到系统中的 WRITE 请求,还是要重新取回前面被存储到系统中的数据的 READ 请求);(b) 指示与该 DMA 请求相关联的特定的 ADDRESS 的标识符(例如,特定的逻辑或物理“地址”,其对于 DMA WRITE 请求将一般标识出与请求相关联的数据存储开始的位置,或者,对于 DMA READ 请求,将标识出与该请求相关联的数据的重新取回应该开始的位置);(c) 指示该 DMA 请求起源的 I/O 模块的标识符;以及(d)“标签(tag)”,其对于每一个 I/O 模块,唯一地标识出起源于该模块的特定的 DMA 请求。对于 DMA WRITE 请求,该请求还包括要被存储在用于该请求的系统内存中的数据。在图 1 的系统 100 中,与每一个 DMA 请求相关联的数据量是固定的。对于数据量可变的系统,头部可还包括(e)指示与该请求相关联的数据量的标

识符。

[0049] DMA 响应典型地包括：(a) 指示 DMA 响应类型的标识符(即，该响应是对应于 DMA WRITE 请求，还是 DMA READ 请求)，(b) 用于每一个 I/O 模块的“标签”，其可被 I/O 模块用于将每一个 DMA 响应与其对应的 DMA 请求唯一地关联起来，以及，(c) 状态标识符，用于 DMA WRITE 响应的状态标识符指示了对应的写入操作是成功还是失败，或者，用于 DMA READ 请求的状态标识符指示了重新取回的数据的有效性。对于 DMA READ 响应，该响应还包括通过原始 DMA READ 请求所请求的数据。

[0050] 在图 1 的系统 100 中，I/O 模块将通过最低限度地处理该请求来响应外部数据相关的请求，以确定其为 READ 请求还是 WRITE 请求，并且确定对应的 DMA 请求要被引导至其上的纵横块(并在可能时确定内存板)，以便满足外部请求。

[0051] 如上所提供的，I/O 模块在最低程度上处理了外部数据相关的请求之后，该系统的组件将进行操作以满足该请求。例如，I/O 模块将初始处理每一个外部数据相关的请求，以识别出其类型(是 READ 还是 WRITE)，以及确定需要发出的特定的 DMA 请求，以满足该外部请求。然后 I/O 模块将所需的 DMA 请求传达给适宜的纵横模块，用于进一步处理。该纵横模块依次将 (a) 确定内存卡，每一个接收到的 DMA 请求应该被引导至该内存卡上，(b) 将每一个接收到的 DMA 请求由其串行格式转变成可适用于呈现给内存卡的并行格式，以及然后 (c) 将每一个 DMA 请求的所有或者部分转发给内存卡，该内存卡的地址范围横跨与该 DMA 请求相关联的地址。如在下面更详细地说明的，该内存卡中的电路将进一步处理 DMA 请求，以将所提供的数据存储于物理内存中(针对 DMA WRITE 请求)或者，从物理内存重新取回数据(针对 DMA READ 请求)。重要的是注意与每一个 DMA WRITE 请求相关联的数据有效载荷通常被作为该请求的一部分进行传达。在一些实施例中，在纵横模块和内存卡之间可具有确认(ACK) 信号以确定该数据已经被成功地发送给这些内存卡。

[0052] 通过使用用于 I/O 模块和纵横模块的高速串行系统通信链路、可配置纵横模块，以及，用于纵横模块和内存卡的高速并行内存通信链路的组合，存储系统 100 提供了甚高带宽(extremely high-bandwidth) 系统。

[0053] 在以上所讨论的示例中，将 I/O 接口模块耦接至纵横模块的高速串行通信链路被用于运载命令以及控制信息(例如，在 DMA 请求的头部中发现的类型信息) 和数据(例如，在数据有效载荷中所提供的信息)。可替换的实施例被设想成，在其中，一些附加的最小串行通信链路被用于提供不涉及传输在数据有效载荷中所接收的类型数据的通信。例如，将实施例设想成，其中相关的低带宽、低速串行通信链路可被用于特定的通信类别，从而从该高速链路上卸载一定量的业务量(traffic)，并在这些链路性能上获得相应的加速。例如，每一个 DMA READ 请求和 DMA WRITE 响应可以这种方式被发送，针对 DMA WRITE 请求和 DMA READ 响应分别预留高速串行链路，这两者都在它们各自的分组内传输数据。由于 DMA READ 请求和 DMA WRITE 响应并不传送任何数据有效载荷，它们通常在尺寸上小得多，并可利用较慢的通信链路，而无需牺牲较大的性能作为代价。在可替换的实施例中，DMA READ 请求和 DMA WRITE 响应可在可替换的低带宽串行信道上发送至所有纵横模块，或者从所有纵横模块通过单独共享的高速链路被广播。

[0054] 高速串行链路还可被用于使用多点总线将所有纵横块连接至所有 I/O 模块，以类似于数据的信号速率提供这种边带数据。可替换地，这种边带链路可实现成环形构造

(ring structure)。在这两种情况下,其可在额外的判决逻辑和硬件的代价下减少系统中点到点连接的数量,但额外的好处是带来了更高的数据吞吐量和带宽。

[0055] 在下面提供了涉及以上所讨论的特定元件的额外的细节,以及元件和 / 或总体系统的可替换的实施例。

[0056] I/O 模块

[0057] 图 1 中的 I/O 模块可采用多种形式。在图 2 中提供了可适用的 I/O 模块的一个示范性实施。

[0058] 参考图 2,示出了示范性 I/O 模块 200。在该示例中, I/O 模块 200 提供了具有两个分开的光纤信道(“FC”)端口(201 和 203)的接口,每个都使特定的 I/O 模块 200 连接至光纤信道主机总线适配器(未示出),从而使外部主机作为 SCSI 装置访问系统 100。每一个端口(201 或 203)包括单个发送链路和单个接收链路,每一个链路以每秒 2.125gigabits (Gb/s) 或 4.25Gb/s 工作。每一个端口可支持点到点和判决环光纤信道协议(Arbitrated Loop Fibre Channel protocols)。将体会到的是,这种物理接口和 SCSI 和光纤信道(FC)协议是示例性的,而且也可使用其它的物理接口和协议。

[0059] 参考图 2,示例性 I/O 模块 200 包括光电转换器 202a 和 202b,用于在光和电的格式之间转换信息。通常,这些装置(202a 和 202b)中的每一个将从其外部光纤接收链路(201 或 203)上接收的信号转换成电信号,然后将该电信号转发至高速控制器 204。同样地,这些装置中的每一个将从高速控制器 204 接收到的电信号转换成在其外部光发送链路上被发送的光信号。已知的光电转换器可被使用。

[0060] 如以上所提到的光电转换器被耦接至高速控制器 204 上。在图 2 的示例中,高速控制器 204 为经配置的场可编程门阵列(FPGA),例如 Xilinx Virtex-4FPGA 装置。控制器 204 通过通信链路被耦接至可编程微处理器(CPU)206,诸如 Freescale MPC8547 处理器 206,并连接至内存 208。在所说明的示例中,内存 208 是由 DDP 内存组件组成的。

[0061] 正如可看出的,控制器 204 提供了四个高速串行系统通信信道 210、212、214 和 216。处理器 206 和控制器 204 一起实现了在外部光纤信道端口(201 和 203)和四个串行系统通信信道 210、212、214 和 216 之间的协议变换功能。在所示出的示例中,串行系统通信信道中的每一个都包括全双工物理层,其包括发送(TX)子信道和接收(RX)子信道。每一个子信道进一步包括两个区别为 5Gb/s 的串行通信链路,它们被绑定在一起,以形成能够具有 10Gb/s 数据传输速度的单个子信道。在该示例中,在串行系统通信信道发送数据之前将数据进行 8B/10B 编码,产生 8Gb/Sec (或每秒 1GByte (1GB/sec))的数据传输速率。可替换的编码方案(例如,64B/66B 编码)也可被采用。在图 2 的示例中,使用了在 Xilinx Vertex-4FPGA 装置中可用的多千兆位收发器(MGT)模块来实现每一个串行通信信道,但是可体会到的是,也可采用其它类似的实现方法,而不背离本公开内容的教导。绑定通信链路的有效过程,诸如那些可通过 FPGA 装置中的 Xilinx Virtex-4 族实现的,可被用于形成通信信道。可替换地,与其它串行化 / 解串行化(SerDes)协议相关联的其它的绑定方案也可被采用。

[0062] 在图 2 的示例中, I/O 接口模块 200 提供了四个串行系统通信信道 210、212、214 和 216,其分别连接至不同的纵横模块。对这四个全双工串行系统通信信道的使用共同地向每一个 I/O 模块提供了潜在的 4GB/sec 的读取带宽和 4GB/sec 的写入带宽。假设图 1 的

系统 100 使用五个 I/O 模块的情况下, 示例性示出的系统将具有 20GB/sec 的总读取带宽和 20GB/sec 的总写入带宽。

[0063] 通常, 高速控制器 204 接收并处理在 FC 接口 201 和 203 上接收的外部数据相关的请求。如以上所说明的, 每一个外部数据相关的请求将典型地包括头部, 其指示了: (a) 该请求是 READ 还是 WRITE 请求, 以及 (b) TARGET ADDRESS (目标地址)。头部可选择性地指示 (c) SOURCE ADDRESS (源地址), 以及 (d) 该请求相关联的数据量的指示。对于 WRITE 请求, 这些请求将还包括数据有效载荷。高速控制器 204 将处理该请求, 并可将许多头部信息(但无任何数据有效载荷)提供给处理器或 CPU206。该 CPU206 将管理该 FC 协议的至少一些部分并辅助总系统接口的管理。

[0064] 如前面所讨论的, 高速控制器 204 和 CPU206 一起工作, 以管理外部光纤信道链路。除一些最低限度的 CPU 辅助外, 该高速控制器 204 在大多数实施例中独自负责发出内部 DMA 请求, 其需要满足外部数据相关的请求中的每一个请求。尤其是, 为了满足每个外部需求, 高速控制器 204 确定了必须被发出的特定的 DMA 请求, 以及该 DMA 请求必须被发送到的纵横模块。一旦该高速控制器识别出适用的纵横模块和满足外部请求所需要的 DMA 请求, 其向适当的纵横模块开始发出 DMA 请求。通常, 可由 I/O 模块 200 发出的待处理 DMA 请求(尚未接收到响应的那些请求)的数量受 DMA 头部中标签字段(tag field)的大小的限制。对于相对较小的传输, 所需 DMA 请求的数量将是较小的, 并且高速控制器 204 可发出它们所有, 而不用等待任何响应的接收。对于较大的传输, 所需 DMA 请求的数量可能是较大的, 并且高速控制器 204 可能被迫延迟对 DMA 请求的发出。例如在给定标签值为 8bit 的情况下, 该高速控制器可初始地发出 256 个 DMA 请求而不必重复使用任何标签值。每一个待处理的 DMA 请求将被唯一地贴上标签, 使高速控制器能够将每一个 DMA 响应与其对应的 DMA 请求关联起来。在返回对这些 DMA 请求的响应时, 高速控制器可开始重复使用这些来自自己完成的 DMA 请求(那些已经接收到 DMA 响应的 DMA 请求)的标签, 从而能够发出后续 DMA 请求。

[0065] 通常, DMA 请求和它们对应的 DMA 响应在串行系统通信信道 210、212、214 和 216 上传输。因为每一个 I/O 模块提供了四个串行通信信道, 其可同时或接近同时地向不同纵横模块独立地发送数据和 / 或从不同纵横模块独立地接收数据。

[0066] 如早前所说明的, 为了满足每一个外部数据相关的请求, 高速控制器 204 负责将 READ 和 WRITE DMA 请求传达给正确的纵横模块。在一个实施例中, 高速控制器 204 确定了哪个纵横模块应该通过使用在该高速控制器内存 208 中所维护的查询表格接收每一个 DMA 请求。在该实施例中, 控制器 204 将访问并维护查询表格, 其将包括信息, 该信息将由系统所呈现的总地址范围内的特定地址范围的信息与特定纵横模块相关联。这种方法的一个好处是, 其在对系统进行配置和重配置方面提供了一定程度的灵活性。在再进一步的实施例中, 在内存 208 中的查询表格将进一步将所呈现的特定地址范围与给定的内存卡关联起来, 并将不同的内存卡与纵横模块关联起来, 使得所接收的地址将与给定的纵横模块相对应, 并且与被耦接至该给定的纵横模块的特定的内存卡相对应。

[0067] 在可替换的实施例中, 给定的纵横模块和所接收的地址之间的关系是“硬件连线的”, 使得给定的地址范围将总是对应于特定的纵横模块。

[0068] 在另一个实施例中, 给定的纵横模块和所接收到的地址之间的关系是动态的, 从而使失败的或其它不可用的内存卡或纵横模块能够被“重新映射”到另一个内存卡或纵横模块

上。

[0069] 通过 I/O 模块(诸如图 2 中的 I/O 接口模块 200)发出的 DMA 传输,可指定固定或可变数据量的传输。在 DMA 传输尺寸固定的实施例中,DMA 请求可不必提供关于要被传输的数据量的信息。在内存卡使用闪存式内存的实施例中,DMA 传输大小被固定成在单个闪存式内存页面中所存储的数据量。

[0070] DMA 操作可传输可变的数据量。在这样的实施例中,每一个 DMA 请求将典型地包括对要被传输的数据量的指示(通常按照最小数据量的倍数)。

[0071] 在 DMA WRITE 操作过程中,不管 DMA 传输是使用固定的还是可变的传输尺寸执行的,高速控制器 204 将从连接的外部主机装置接收的数据排成队列。对于包括多个 DMA WRITE 请求的较大的数据传输,I/O 接口在开始发出 DMA WRITE 请求之前不需要等待接收到所有的外部数据。而是,只要数据足够支持每一个 DMA WRITE 请求,高速控制器就可开始发出 DMA WRITE 请求,并继续发出 DMA WRITE 请求。

[0072] 在 DMA READ 操作过程中,高速控制器将在相反方向(离开系统内存)上流动的数据排成队列。在这种情况下,高速控制器将发出多个 DMA READ 请求,并然后等待要从各个纵横模块被返回 DMA READ 响应。由于这些响应可以与它们对应的 DMA READ 请求不同的顺序被接收,高速控制器必须将从纵横模块接收的数据排成队列,直到数据可以其正确的顺序被传送。在假设外部数据请求需要发出 256 个 DMA READ 请求的场景下,可能发生的是,最后的 DMA READ 响应将对应于第一个 DMA READ 请求,从而需要 I/O 接口将来自 256 个 DMA 传输的数据排成队列。只要数据可按顺序被传送,I/O 接口就可以仅将数据卸载到外部请求器(主机装置)中。

[0073] 通过上述方式,I/O 接口模块 200 可从外部主机接收数据相关的请求,访问系统 100 内的内存,以完成该请求,并在外部将所请求的数据提供给主机装置。

[0074] 应该体会到的是,图 2 中的 I/O 接口模块只是可被用在本公开内容中的系统中的 I/O 接口模块的一个示例。也可以使用可替换的 I/O 接口。在图 3 中提供了可替换接口的一个示例。

[0075] 图 3 示出了可替换的 I/O 接口模块 300,其能够通过多个高速无限带宽端口(一起被称为 301)接收数据相关的请求。

[0076] 参考图 3,示例性可替换 I/O 模块 300 包括无限带宽接口芯片 302,其能够在多个无限带宽的端口 301 上接收和发送数据相关的请求和响应。该无限带宽接口芯片可以是标准的无限带宽芯片,诸如那些可从 Mellanox 获得的。

[0077] 通常,无限带宽接口芯片 302 经由无限带宽端口 301 接收数据相关的请求,并在最低程度上将所接收到的请求至少解码成识别出命令头部,并且对于外部 WRITE 请求解码出数据有效载荷的程度。然后接口芯片 302 将使得命令头部信息通过使用转换元件、路由元件或共享内存元件 305(其在图 3 的示例中为 PCI Express (PCIe)转换器)能够用于控制 CPU306。使用转换/路由/内存元件 305,接口芯片 302 还将使接收到的数据可被高速控制器 304(其可以是经配置的 FPGA)使用。该高速控制器 304 将多个串行系统通信信道 310、312、314 和 316 提供给纵横模块,并且对所接收的数据相关的请求进行操作以与以上结合图 2 中的高速控制器 200 说明的相似的方式,向纵横模块提供信息和数据且从纵横模块接收信息和数据。

[0078] 将体会到的是,以上所讨论的 I/O 接口模块 200 和 300 仅是示例性的,并且还可使用其它类型的 I/O 接口模块,包括那些使用不同的物理层的、不同通信协议的、以及分立的装置(与经配置的 FPGA 相对)的 I/O 接口模块,而不背离本公开的内容。

[0079] 通常,对 I/O 模块的主要要求是,其能够接收和响应外部提供的数据相关的请求,并且其能够在多个高速系统通信信道上将那些请求转达给多个纵横模块。

[0080] 纵横模块

[0081] 如以上所说明的,本公开内容的系统中每一个纵横模块的总体一般功能为:(a) 从 I/O 模块接收 DMA 请求并将那些 DMA 请求(加上所有关联的数据有效载荷)提供给连接至该纵横模块的适当的内存卡,以及(b) 从被连接至纵横模块的内存卡接收重新取回的数据,并将重新取回的数据(和潜在的一些相关信息)提供给与该重新取回的数据相关联的 I/O 模块。

[0082] 可使用不同的手段来实现本公开系统中的纵横模块。图 4 示出了用于实现以上结合图 1 说明的类型的纵横模块的一个示例性手段。

[0083] 参考图 4,示出了示例性纵横模块 400。虽然该纵横模块可通过使用分立电路、特殊应用集成电路(ASIC),或者两者混合来实现,但是在图 4 的实施例中,纵横模块 400 是通过使用经配置的 FPGA(诸如 Xilinx Vertex4FPGA)实现的。

[0084] 在该示例中,经配置的 FPGA 被配置成提供五个串行系统通信端口 402、404、406、408 和 410。每一个串行通信端口被配和被连接以提供了在其本身和 I/O 模块的高速通信端口中的一个端口之间的高速的、双向通信信道。因此,在该示例中,由于纵横模块 400 提供了五个全双工通信模块,其能够同时且分别地与系统 100 中的 I/O 模块中的每一个进行通信。因此,可将通信端口 402 耦接至 I/O 接口模块 102 的串行通信端口中的一个上,将端口 404 耦接至 I/O 接口模块 104 的端口上等等,使得在每一个 I/O 模块和纵横模块 400 之间存在串行连接。

[0085] 除提供了以上所说明的五个串行通信端口以外,图 4 中的纵横模块 400 还提供了多个并行的通信总线,其被用于使得在纵横模块 400 和多个内存卡之间能够通信。在图 4 的示例中有五个内存卡 412、414、416、418 和 420,并且纵横模块 400 使用并行总线与内存卡通信。在图 4 的示例中,纵横模块 400 和每一个单独的内存卡之间的通信是使用并行总线完成的,其在本示例中对于每一个内存卡包括十六比特(16)宽数据总线和五比特(5)宽控制总线。

[0086] 在一般操作中,纵横模块 400 主要起到一个装置的作用,用于经由串行系统链路中的一个链路接收 DMA 请求,识别出 DMA 请求应被引导到其上的特定内存卡,将接收到的 DMA 请求由串行格式转换成并行格式,以及将并行格式的 DMA 请求提供给适当的内存卡。此外,纵横模块 400 将典型地接收由内存卡提供的数据、确定数据将用于哪个 I/O 模块,以及通过串行系统通信链路中的一个通信链路将所接收的数据作为 DMA 响应传送给适当的 I/O 模块。

[0087] 在该示例性的情况下,纵横模块 400 使用串行接口总线与每一个 I/O 模块通信,并使用并行接口总线与每一个内存卡通信。应该注意到,I/O 接口总线的实现取决于 I/O 模块,并且内存接口总线取决于内存卡。理论上,这两种总线将被实现成串行或并行接口,假设该总线可在要求的数据速率下执行。接口总线可取决于应用,并且对于那些本领

域技术人员,影响特定总线选择的设计决策应该是清楚明了的。例如,为了使连接器密度最小化,内存卡接口可被实现成高速串行连接,其与 I/O 模块本质上相似,使纵横块到内存(crossbar-to-memory)的接口简单化。

[0088] 由于将纵横块端口连接至 I/O 模块的总线中的每一个总线都是点到点连接,所以使判决最小化,并且由被设计成在必要时使纵横块能够拖延 I/O 总线的机制进行调节,以防止交叉运行(over-run)。

[0089] 对 FPGA 进行配置来执行这些任务对于那些受益于本公开内容的本领域普通技术人员来说应该是显而易见的。

[0090] 内存卡

[0091] 可在本公开系统中使用的内存卡可采用许多种形式。其可利用各种各样的内存,包括不同类型的随机访问内存(诸如,DDR RAM、DDR2RAM、DDR3RAM 等),以及不同类型的闪存式内存(包括 MLC 闪存式内存和 SLC 闪存式内存等)。部分地取决于所使用的内存类型,这些内存卡的构造也可以是不同的。

[0092] 图 5 示出了内存卡 500 的一个示例性实施例,其采用了闪存式内存并且可被用于本公开内容的系统中。在图 5 的示例中,该闪存式内存为 SLC 闪存式内存,但是采用 MLC 闪存式内存的实施例也是可以设想到的。

[0093] 参考图 5,闪存式内存卡 500 包括系统控制器 502,其通过将系统控制器耦接到纵横模块中的一个模块(图 5 中未示出)的并行总线 504 接收信息。如上所提及的,该并行总线 504 包括十六比特(16)宽数据总线和五比特(5)宽控制总线。由纵横模块提供的 DMA 请求是由系统控制器 502 接收并处理的。系统控制器 502 在内存卡和纵横模块之间管理通信协议,并可实现诸如对总线错误进行管理的纠错的功能。此外,系统控制器 502 还可部分地处理所接收的 TARGET ADDRESS(目标地址),以确定在内存卡上的哪个特定内存元件与该 TARGET ADDRESS 相关联。

[0094] 在示例性的系统中,系统控制器 502 通过直接的连接与许多个单独的闪存控制器 506、508、510 和 512 通信。系统控制器 502 还与 CPU503 通信,CPU503 可同时与单独的闪存控制器通信。这种通信可以是通过与闪存控制器的直接连接(如图 5 所示),或者通过“直接通过(pass-through)”连接进行的,其中,为了访问闪存控制器,CPU503 与系统控制器 502 通信。这样做一般是为了减少总线扇出(fan-out),并且是出于让那些本领域技术人员更清楚的设计考虑的。

[0095] 在所示出的示例中,系统控制器 502 和闪存控制器 506、508、510 和 512 之间的通信是使用独立的十六比特(16)宽并行总线完成的,一个这样的独立并行总线被耦接在系统控制器 502 与这些单独的闪存控制器 506、508、510 和 512 中的每一个之间。类似的独立十六比特(16)宽并行总线可被用在 CPU503 与每一个闪存式控制器 506、508、510 和 512 直接通信的实施例中。

[0096] 这些单独的闪存式控制器中的每一个分别被耦接至物理闪存式内存空间 513、514、515 和 516,并且被耦接至控制器内存 517、518、519 和 520(其可以是例如 DDR RAM 内存)。在所示出的示例中,每一个物理闪存式内存空间(513-516)是由十个独立的闪存式内存芯片形成的。在操作中,闪存式控制器 502 将处理 DMA 请求,以将所提供的数据存储指定的物理闪存式内存中,并重新取回所请求的数据,并将其提供给系统控制器 502。可采用

许多个不同类型的闪存控制器。在提交于 2009 年 9 月 5 日的第 12/554,888、12/554,891 和 12/554,892 号共同未决的美国专利申请中更详细地说明了一个优选的控制器及其操作,在此结合其内容作为参考。

[0097] 可注意到,因为本公开内容的系统 100 是以这样的方式进行处理,即,由 I/O 接口模块接收的 TARGET ADDRESS 为被提供给内存卡的地址,所以本系统可以使用闪存式内存(其最终需要将所接收的 TARGET ADDRESS 转译成物理闪存地址位置)和 RAM 内存(或者无需进行逻辑到物理地址转译的其它内存)。而且,由于从 TARGET ADDRESS 到物理闪存地址的任何转换都是在将 DMA 请求提供给内存卡之后发生的,所以闪存式内存卡和 RAM 内存卡都能够被本系统使用。这是因为闪存控制器将所接收到的 TARGET ADDRESS 映射到特定的物理闪存地址上的操作对于将纵横模块耦接至内存卡的接口总线来说在很大程度上是透明的。

[0098] 图 6 示出了内存卡 600 的可替换的实施例,其采用了 RAM 内存,并且可被用在本公开内容的系统中。在该实施例中,由纵横模块提供给内存卡的地址被用作访问 RAM 内存的物理地址。参考图示,内存卡 600 包括许多个元件,其可全部被定位在并固定到单个多层印刷电路板上。

[0099] 通常地,内存卡 600 包括系统控制器 602,就像图 5 中的控制器 502,其从纵横模块接收 DMA 请求,并使用适合于该系统的协议(其可包括或结合纠错)处理该请求。然后系统控制器将该请求和所有数据有效载荷传递给内存控制器 604,其可将为所接收数据的快速 DMA 传输分配对应于 TARGET ADDRESS (针对 WRITE 请求)的适当的物理地址,或者使用 DMA 传输从 RAM 内存 606 (其可以是 DDR、DDR2、DDR3 或任意其它的高速 RAM 内存)重新取回数据,并将该数据提供给系统控制器 602 (针对 READ 请求)。将体会到的是,图 6 中示出的箭头的指向仅表示对 WRITE 操作示例的描述。还将体会到的是,为了实现的目的,控制器 602 和控制器 604 都可被合并到一个 FPGA 装置中。

[0100] 可替换系统实施例

[0101] 在以上所说明的系统 100 中,纵横模块被用于接收串行发送数据相关的请求,以及将那些请求转换成被提供给内存卡的并行发送的请求。可以设想可替换的实施例,其中,可除去纵横模块,并且可将数据相关的请求从 I/O 模块直接串行发送至内存卡。

[0102] 图 7 示出了可替换的存储系统 700,其中,I/O 模块或装置可使用高速串行通信链路与内存板直接通信,而不使用纵横模块。

[0103] 参考图 7,可替换的存储系统 700 包括多个 I/O 模块 701、702、703、704、705、706、707、708、709、710、711 和 712,其经由一个或多个通信信道分别被耦接至一个或多个外部主机。在所示出的示例中,每一个 I/O 模块提供了多个高速双向全双工串行通信信道,其使每一个 I/O 模块与多个内存板 740、742、744 和 746 中的每一个进行通信。在图 7 的示例中,每一个 I/O 模块 701 至 712 提供了八个串行通信端口,并且连接 I/O 模块和内存板的每一条线都旨在反映出两个分开的全双工通信信道。

[0104] I/O 模块 701 至 712 以这样的方式被耦接至内存板 740、742、744 和 746,即,能够使用甚高带宽通信。例如,第一 I/O 模块 701 提供了八个串行数据端口(每一个内存板两个端口)。因此,I/O 模块 701 可接受的最大数据速率将是串行通信信道最大带宽的 8 倍。

[0105] 在操作中,I/O 模块 701 至 712 的操作类似于以上结合图 2 和图 3 所说明的 I/O 模块。但是,由于图 7 中的 I/O 模块提供了更多个串行通信信道,它们能够对更多数据相关的

请求或响应进行同时且独立地传送。这在通常用于存储系统 700 的而且特别是在 I/O 模块 701 至 712 和内存板 740、742、744 和 746 之间产生了更高的带宽。例如,由于 I/O 模块 701 提供了八个串行通信端口(每个内存板两个),其能够向内存板 740、742、744 和 746 中的每一个同时地(或接近同时地)并且独立地发出两个 WRITE 请求,总共八个同时地(或接近同时地)被处理的 WRITE 请求。在一些实施例中,没有纵横块,而是将与参考原始的系统 100 中相同的判决和拖延机制放置在每一个内存板上。

[0106] 在图 7 的示例中,每一个串行系统通信链路被用于向 I/O 模块或通信装置和内存板传递数据和控制信息,并且从 I/O 模块或通信装置和内存板传递数据和控制信息。此外,每一个内存板将接收数字数据并将响应于 WRITE 请求(以将数字数据存储在该内存板的物理内存空间中特定的位置)并且将响应于 READ 请求(以重新取回和提供被存储在该板上物理内存的特定位置处的数字数据)。因为每一个高速串行通信链路可以非常高的速率(在本示例中为 625MB/sec)接收或发送串行数据,所以在图 7 中的系统可以非常高的数据速率存储和重新取回数据。

[0107] 图 8 示出了关于图 7 的系统 700 中使用的示例性内存板 800 的构造的细节。

[0108] 转到图 8,内存板 800 包括由多个单独的内存芯片组成的存储控制单元 802 和物理内存空间 804。该内存板进一步包括适当的接口电路 806,以允许内存板接收多个串行通信链路。在图 8 中,该接口电路 806 使内存板 800 从 24 个串行通信链路上接收输入。在图 8 的示例中,各种组件都被放置在同一个印刷电路板 800 上。

[0109] 存储控制单元 802 可通过使用场可编程门阵列,或者“FPGA”(诸如可从 Xilinx 获得的 Virtex-6FPGA (XC6VLX240T-2FFG1156CES) 其工作在 333MHz) 组成。但是,应该理解的是,存储控制单元 802 可以可替换地使用其它类型的 FPGA 装置、分立电路、编程的微处理器,或者以上中的任何的或所有的组合来实现。

[0110] 在所示出的示例中,物理内存空间 804 是使用了多个双倍数据速率动态随机访问内存芯片、DDR(诸如可从 Micron Technologies, Inc. 获得的 DDR3-800(MT4J128M8BY-25E),其工作在 333MHz)构建成的。但是将理解到的是,也可以采用其它形式的内存,诸如可替换的 DDR 装置、闪存和其它类型的内存,而不背离本公开内容的教导。

[0111] 在本示例中,存储控制单元 802 驱动 288 外部并行数据链路(在图 9 中更好地看出)至物理内存空间 804。因为内存空间使用了主频在 333MHz 的 DDR 内存,图 8 中的高速内存板将能够在每一个时钟周期内在物理内存空间中存储高达 576 比特的数字信息(512 比特数据和 64 比特 ECC),因此提供的总体存储速率对于数据和 ECC 为大约 24000MB/秒,单独用于数据的为 20.83GB/秒。

[0112] 图 9 详细示出了示例性高速内存板的结构的部件。尤其是,图 9 示出了图 8 中的存储控制单元 802 内的结构,其可被用于从串行通信链路接收数字数据并对这样的数据进行操作以存储在物理内存空间 804 中。

[0113] 参考图 9,示出了高速串行通信链路中的一个,串行链路 900,其被耦接至存储控制单元 802 上。高速串行通信链路 900 被提供给串行器/解串行器以及分组模块 912,其将被放置在存储控制电路 802 中。在图 10A 中进一步说明的这种串行器/解串行器和分组模块 912 描绘了被输入多千兆位收发器(MGT) 1010 的串行链路,该 MGT1010 将被放置在存储控制单元 802 中。该 MGT1010 将从串行通信链路接收串行数据,并且,在周期性的基础上,

将所接收到的串行数据转换成并行数据。进而参考图 10A 的示例, MGT1010 接收串行数据, 并以 156.25MHz 提供并行的 32 比特宽数据。

[0114] 在图 10A 的示例中, 来自 MGT1010 的 32 比特宽的并行数据以这样的方式被提供给 256 比特宽的多字深先进先出(FIFO)存储缓存 1012, 即, 从 MGT1010 所接收到的 32 比特宽的数据被分组成 256 比特宽的数据字。在图 10B 至图 10D 中一般地说明了这种处理。

[0115] 参考图 10B, 示意性地说明了 256 比特宽的 FIFO 存储器 1012, 其被分组成 256 比特宽字 A1 至 A8。所示出的系统将来自 MGT1010 的 32 比特宽数据分组以形成 256 比特宽字的方法如下: 首先, 从 MGT1010 接收初始的 32 比特宽字, 在本例中为 A1。该初始的 32 比特宽字 A1 将被存储在初始的数据位置, 如图 10B 所反映的。从 MGT1010 接收下一个 32 比特宽字, 在本例中为 A2, 其将被存储在第二个位置, 第三个字 A3 将被存储在第三个位置, 等等, 直到八个字 A1 至 A8 已经被接收并被存储在 FIFO1012 中以形成 256 比特宽的字 A1 至 A8, 如图 10B 所反映的。

[0116] 一旦通过以上所说明的处理形成了 256 比特宽字 A1 至 A8, 并且从 MGT1010 接收到另一个 32 比特宽字, 本例中为 B1, 初始的字 A1 至 A8 将被“向下”转移至 FIFO1012 中的另一个位置, 并且新接收到的 32 比特宽的字将被存储在该初始位置, 如图 10C 所反映的。该处理将重复直到构建成第二个全部 256 比特宽的字 B1 至 B8, 如图 10C 所反映的。从 MGT1010 接收 32 比特字、将所接收到的 32 比特的字分组成 256 比特的字, 以及通过 FIFO1012 转移 256 比特宽的字处理将持续直到在该 FIFO1012 中的 256 比特宽的内存位置被完全移满。在图 10D 中示出了更加完全移满的 FIFO1012 的示例。

[0117] 虽然堆栈 FIFO1012 的“深度(depth)”可能根据实现方式而不同, 但是图 9 和图 10A 至图 10D 的示例中, FIFO1012 具有至少 32 个 256 比特宽字的深度。这是因为, 在本示例中, 一组 32 个 256 比特宽字是用于向物理内存空间 804 传送数据和从物理内存空间 804 传送数据的基本单元。这样, 在此每一组 32 个 256 比特宽字将被称为基本内存周期(“BMC”)单元。在本示例中, 每一个 BMC 单元将包括 8,192 数据比特(32×256)或 1KB 数据。

[0118] 参考回到图 9, 来自分组 FIFO912 的数据被提供给多信道 FIFO 缓存 914, 在图 9 中仅示出了其中一个信道。在所示出的示例中, 多信道 FIFO 缓存 914 为 256 比特宽, 其能够接收以 256 比特/秒的速率接收数据, 并且以 27.7MHz 的速率作为主频。使用“突发(burst)”传输将数据从串行器/解串行器和分组模块 912 被提供给多信道 FIFO 缓冲器 914, 在此情况下, 完整的数据的 BMC 单元在 32 个时钟周期内被传输。

[0119] 一旦完整的数据 BMC 单元从串行器/解串行器和分组模块 912 被传送到多信道 FIFO 缓冲器 914, 该多信道 FIFO 缓冲器 914 将包括具有 32 个连贯的 256 比特“字”的形式的数据 BMC 单元。在所示出的示例中, 多信道 FIFO 缓冲器 914 的深度为, 其将能够存储多个数据 BMC 单元。这在图 11 中大体反映出, 在此情况下, 多信道 FIFO 缓冲器 914 的内容被描述为包括三个数据 BMC 单元(由框 1140a、1140b 和 1140c 体现出)。

[0120] 参考回图 9, 被存储在多信道 FIFO 缓冲器 914 中的数据经由工作在 333MHz 的 256 比特宽并行数据总线被提供给纠错和数据保护电路 916。该纠错和数据保护电路 916 处理所接收到的数据, 以引入一个或多个纠错码比特(“ECC”比特), 并对数据进行处理, 以加强保护防止数据出错。对于那些本领域普通技术人员来说, ECC 处理的示例是已知的, 并且任意可用的 ECC 处理机制都是可被使用的。此外, 纠错和数据保护电路也可实现其它

数据保护技术或变换,诸如“Chipkill”和常用于加强数据保护的其它技术。如那些本领域普通技术人员所理解的,Chipkill 是由 IBM 研发的先进的 ECC 形式,其保护计算机内存系统,防止任意的一个内存芯片出错,以及防止单个内存芯片的任何部分的多比特错误。关于 Chipkill 的额外的信息可在以下文件中找到,在此结合其内容作为参考:Timothy. J.Dell,A White Paper on the Benefits of Chipkill-Correct ECC for PC Server Main Memory (1997),IBM Microelectronics Division。

[0121] 作为纠错和数据保护电路 916 的操作结果,将在被提供给该电路的每一个 256 比特“字”中添加比特,并且针对每 256 比特的输入,在本示例中,该电路的合成的输出将是 288 比特。这 288 比特输出将对应于被提供给 ECC 和数据保护电路 916 的输入数据,并对应于由电路 916 加入的保护和 ECC 比特。

[0122] 来自数据保护电路 916 的这 288 比特宽的输出经由 288 比特宽的并行总线被提供给两个输出 FIFO 缓冲器 918 和 920。在本示例中,该并行总线的主频是 333MHz。来自数据保护电路的数据以“ping-pong”的方式被提供给输出 FIFO 缓冲器 918 和 920,使得来自该电路的第一个 288 比特“字”被提供个输出 FIFO 缓冲器中的一个(例如,输出 FIFO 缓冲器 918),同时下一个 288 比特“字”在随后的时钟周期被提供给输出 FIFO 缓冲器中的另一个(例如,FIFO 缓冲器 920)。以这种方式,对应于完整的数据单元的数据被放置在两个输出 FIFO 缓冲器 918 和 920 中,使得每一个输出 FIFO 缓冲器中有一半的数据。在本示例中,每一个输出 FIFO 缓冲器 918 和 920 的深度使得每一个缓冲器可存储对应于多个数据单元的数据。

[0123] 在图 12 和图 12A 中一般地示出了将数据从电路 916 移至输出 FIFO 缓冲器 918 和 920 的处理,其中示出了在数据单元传输之后的输出 FIFO 缓冲器 918 和 920 的示例性内容。如图所反映的,在两个缓冲器中都包含了示例性数据单元的数据,并且弥补该完整的数据单元的数据被“交织”穿过两个输出 FIFO 缓冲器 918 和 920。

[0124] 参考回图 9,来自两个输出 FIFO 缓冲器 918 和 920 的数据被提供给高速的 576 比特宽的输出内存缓冲器 922。在图 9 的示例中,每一个 FIFO 输出缓冲器 918 和 920 每个时钟周期将 288 比特的数据和错误保护信息提供给高速输出内存缓冲器 922,并且进入该高速内存缓冲器 922 的输入在本示例中主频是 333MHz。在该速度和数据传输速率下,完整的数据数据单元(32 个字)可在 16 个时钟周期内被传输。此时,将 16 个字存储在 FIFO 输出缓冲器 918 和 920 中的每一个中,并且因此传输整个数据单元需要 16 个周期。

[0125] 输出 FIFO 缓冲器 918 和 920 与高速输出内存缓冲器 922 之间的耦接使得之前被存储在输出 FIFO 缓冲器 918 和 920 中的“交织”的数据合并形成单个的顺序正确的 576 比特“超字”,在这里数据反映出了其最初穿过高速串行通信链路 900 而被接收的顺序。

[0126] 参考回图 9,将来自高速输出缓冲器的数据通过主频为 333MHz 的 DDR 的 288 比特宽并行总线提供给物理内存 804 (未示出)。由于 DDR 时钟在两个时钟边沿发送数据,来自输出内存缓冲器 922 的数据以 667MHz 的速率被有效传输。结果,传输至图 9 中电路的内存上的有效数据传输速率为 20.83GB/s。

[0127] 为了解释的目的,图 9 的示例仅结合单个高速串行输入,在本示例中为串行输入 701(见图 7),示出并说明了对本系统的操作。在全部示例性系统中,可提供高达 24 个单独的高速串行通信链路。在图 13A 至图 13B 中反映出了由本系统对来自这样的具有多个串行

通信链路的系统的数据进行处理的方法。

[0128] 首先参考图 13A, 示出了与以上结合图 9 所讨论的非常相似的系统。但是, 在图 13A 的示例中, 用于该示例性高速输入的输入 MGT1010 和分组 FIFO 缓冲器 1012 被合并以形成输入处理块 1370。另外, 还示出了 11 个其它的串行输入 1371 至 1381, 其分别具有其本身对应的输入处理块。这些输入处理块中的每一个就像以上结合图 9 和图 10A 至图 10D 所说明的电路那样工作, 以能够以突发模式的方式将完整的数据 BMC 单元提供给多信道 FIFO 缓冲器 914。

[0129] 由于图 13A 中的系统的设计, 该系统可以非常高的速率接收数据, 使得, 在峰值操作条件下, 来自 WRITE 操作的数据几乎总是被提供给多信道 FIFO 缓冲器 914, 而来自多信道 FIFO 缓冲器 914 的数据几乎总是被提供给纠错和数据保护电路 916。

[0130] 但是, 图 13A 的系统优选地处理来自由所说明的系统所提供的一半高速串行通信链路的数据。如图 13B 所示, 该示例性系统包括另一组电路, 类似于那些以上所说明的, 包括另一个多信道 FIFO 缓冲器 914', 其处理来自剩余的 12 个串行通信链路的数据, 并将数据提供给第二纠错和数据保护电路 916'。

[0131] 如图 13B 所反映的, 第二纠错和数据保护电路 916' 以“ping-pong”的方式将其输出提供给 FIFO 缓冲器 918 和 920。但是, 在该示例性系统中, 用于从第二纠错和数据保护电路 916' 传输数据的过程与第一纠错和数据保护电路 916 “异相(out-of-phase)”操作, 使得, 虽然第一数据保护电路 916 正在将数据传输至输出 FIFO 缓冲器中的一个(例如, 输出 FIFO 缓冲器 918), 但是第二纠错和数据传输电路 916' 正在将数据传输至另一个输出 FIFO 缓冲器(例如, 输出 FIFO 缓冲器 920)。在下一个时钟周期期间, 该传输将“交换”。以这种方式, 数据可在每一个时钟周期期间总是被传送至两个输出 FIFO 缓冲器 918 和 920。

[0132] 为了避免将来自不同的 WRITE 操作的数据错误地“混合”, 通过第一和第二纠错和数据保护电路 916 和 916' 提供的数据优选地以这样的方式被传输至输出 FIFO 缓冲器 918 和 920, 即, 其能够被重新组合成看起来像是在单个串行通信链路上接收的数据。在图 14A 和图 14B 中反映出了一种实现该方法的手段。

[0133] 参考图 14A 和图 14B, 在第一数据传输操作期间, 已经准备好从第一纠错和数据保护电路 916 被传输的 BMC 单元的第一偶数字(例如, word0, a0)将被第一电路 916 传输至被预留给第一电路 916 的输出 FIFO 缓冲器 918 的一部分。同时, 已经准备好从第二纠错和数据保护电路 916' 传输的来自 BMC 单元的第一偶数字(word0, b0)将被传输至被预留给用于存储来自第二电路 916' 的数据的输出 FIFO 缓冲器 920 的一部分。在下一个时钟周期期间, 来自第一电路 916 中的 BMC 的第一奇数字(例如, Word1, a1)将被存储在输出 FIFO 缓冲器 920 的预留空间中, 而且来自 FIFO 916' 中的 BMC 的第一奇数字(例如, word1, b1)将被存储在输出 FIFO 缓冲器 918 中的预留空间中。在该方法中, 数据总是被传输至两个输出 FIFO 缓冲器 918 和 920, 使得最大带宽得到保持。

[0134] 在可替换的方法中, 对来自第二纠错和数据保护电路 916' 的数据进行的写入操作可被延迟一个周期, 使得所有偶数字被存储在输出 FIFO 缓冲器 918 中, 而所有奇数字被存储在输出 FIFO 缓冲器 920 中(见图 14C 和 14D)。

[0135] 除了如图 9 所反映出的以及以上其它图所讨论的那样提供电路用于接收和处理数据, 每一个内存卡也可包括类似的构造(本质上进行反向操作), 用于以高数据速率从 RAM

内存重新取回数据,并将所接收到的数据拆包,使得其能够通过高速串行通信信道中的一个,被传输至 I/O 接口模块,然后 I/O 接口模块可将所请求的数据提供给适当的主机。

[0136] 在所示出的示例中,将内存控制器 802 耦接到物理 RAM 内存 804 的数据和地址线使得,在任意的给定的时刻,只能完成 READ 访问或只能完成 WRITE 访问。因此,为提供最佳性能,控制器 802 必须实现一些形式的判决,如图 15 一般所示的。在一个实施例中,通过在控制器 802 中包括判决模块 1504 来完成这种判决,该判决模块 1504 将接收每一数据相关请求的命令头部信息并将该信息存储在与提供该请求的 I/O 接口模块相关联的缓冲器中。在优选的实施例中,该判决模块 1504 将维护用于 READ 操作的分开的缓冲器 1500 和 1503,以及用于 WRITE 操作的分开的缓冲器 1501 和 1503,使得控制器 802 将以 I/O 接口模块接着 I/O 接口模块为基础,在缓冲存储器中维护从每一个 I/O 接口模块接收的 READ 和 WRITE 请求或命令的列表。通常,该列表可在时间戳或时间顺序基础上维护,并且每一个列表入口可包括与从 I/O 接口模块接收到的每一数据相关的请求相关联的命令信息,也就是 TARGET ADDRESS、字计数(或要被传输的数据量的其它指示),以及传输方向的指示(例如,其为一个 READ 还是 WRITE 操作)。

[0137] 为了优化性能,可对 READ 和 WRITE 请求进行如下的处理。

[0138] 通常,判决模块 1504 将给予 READ 请求优先权,并且,除非遇到以下所讨论的特定情况,将在先进先被处理的基础上处理在用于不同的 I/O 接口模块的 READ 请求缓冲器中的 READ 请求。

[0139] 如果判决模块 1504 和控制器 802 已经处理了所有的 READ 请求,使得没有 READ 请求仍然待处理,则该判决模块 1504 和控制器 802 将在先进先被处理的基础上(first-come, first-served)上处理在 WRITE 请求缓冲器中的所有待处理的 WRITE 请求。

[0140] 为了避免具有太多的 WRITE 请求待处理,并帮助解决等待和相干扰的问题,如果确定待处理 WRITE 请求的数量已经超出特定的阈值数,则判决模块 1504 和控制器 802 将处理 WRITE 请求。判决模块 1504 通过考虑到以下的因素来进行阈值超出判断,即:(i) 总计来自所有 I/O 接口模块的待处理 WRITE 请求的总数量;(ii) 来自给定的 I/O 接口模块的待处理 WRITE 请求的总数量;或者(iii) 以上所述的一些组合。例如,如果对给定的 I/O 接口模块的待处理 WRITE 请求的总数量和 WRITE 请求的数量两者都予以考虑,则在下列两种情况中的任一种情况下,判决模块 1504 可使控制器 802 处理 WRITE 请求,即:(i) 待处理 WRITE 请求的总数量超出第一阈值(在该情况下,待处理 WRITE 请求可以先进先被处理为基础进行处理),或者,(ii) 对于给定的 I/O 接口模块的待处理 WRITE 请求的数量超出第二阈值(其可比第一阈值低),在该情况下,判决模块 1504 将使控制器 802 在先进先被处理的基础上处理来自该与超阈值缓冲器相关联的特定 I/O 接口模块的 WRITE 请求。

[0141] 而且,如果确定在与接收数据相关联的输出 FIFO 缓冲器中的一个 FIFO 缓冲器中所存储的数据已经超出了一定的限制,诸如超出该输出 FIFO 缓冲器存储能力 1/2 的量,该判决器可处理 WRITE 请求,在该情况下,将处理与输出 FIFO 缓冲器相关联的 WRITE 请求,使得 I/O 接口模块可持续发出数据和数据相关的请求,而不出现中断或节流(throttling)。

[0142] 除以上所述以外,判决模块 1504 还可控制如何处理 READ 和 WRITE 请求,使得数据相干性得到保护。例如,如果确定出待处理的 READ 请求被引导至特定的地址,并且那里存在更早时间的待处理 WRITE 请求,则判决模块 1504 可使得控制器 802 处理该待处理 WRITE

请求,以保证响应于该 WRITE 请求返回适宜的数据。

[0143] 内存芯片访问

[0144] 存在各种各样的手段和方法可使内存控制器 802 用于将数据传输到物理 RAM 内存或从物理 RAM 内存输出数据。在一个实施例中,控制器 802 可作为,或包括,DDR3 内存控制模块操作,其随着起始指示符从判决模块 1504 接收命令信息以控制数据传输的定时。通常,由判决模块 1504 提供的命令信息可包括:被存储在可用于判决目的缓冲器中的相同的信息,也就是 TARGET ADDRESS,以及更特定地,用于传输的起始地址;要被传输的数据量的标识符,其可以是字计数;以及传输方向。然后起到内存控制模块作用的控制器 802 将如 TARGET ADDRESS (起始地址)所指示的,启动适当的内存芯片。

[0145] 如果传输方向指示到内存的 WRITE,控制器 802 将配合向内存芯片发出命令(例如,定时和地址命令),同时传输来自输出内存缓冲器 922 的数据。

[0146] 如果传输方向指示从 RAM 内存 READ,控制器 802 将配合向内存芯片发出适当的地址和定时命令,同时将来自这些内存芯片的数据传输到与数据被指引到的 I/O 接口模块相关联的输出 FIFO 缓冲器。

[0147] 在一个示例中,控制器 802 将在与前面的命令相关联的数据传输完成之前接收下一组命令信息,因此,使得该系统能够在最大的带宽下操作。

[0148] 在图 16 和图 17 中大体上反映出了在控制器 802 将数据传输进物理 RAM 内存和从物理 RAM 内存传输数据时对其的一般操作。

[0149] 首先参考图 16,示出了一种物理 RAM 内存的特定的布置。在图 16 的实施例中,示出了控制器 802,并且示出了在控制器 802 和物理 RAM 内存之间的连接。在该示例中,被耦接至控制器 802 的物理 RAM 内存采用了七十二(72)个内存芯片,其被划分成四个区段(1602、1604、1606 和 1608),每一个区段具有 18 个内存芯片。在该示例中,每一个内存芯片具有四个存储体(A、B、C、D)。将内存芯片耦接起来,使得它们共享:(a) 地址和控制线(本示例中为 22),以及 (b) 数据线(本示例中为 72)。在图 16 中示出的这种耦接只是将物理内存芯片连接至控制器 802 的一种方法。所有四个区段,尽管具有单独的控制总线,但是它们操作一致,并且在同一时刻向同一存储体发出命令。

[0150] 在图 17 中可找到关于内存控制器 802 的操作的额外的细节。图 17 示出了在两个基本内存周期内对控制器 802 的操作(在此所处的第二个基本内存周期未完成)。通常,每一个内存周期包括将 32 个 288 比特字的传输到(或出)DDR 内存。在特定的示例中,传输发生在 16 个实际的时钟周期内。针对每一个内存卡在 333MHz 下提供了每 48ns 传输 1KB 的数据传输速率,或者 20.83GB/sec (仅数据)。考虑到该系统,如图 7 所反映的,可具有许多个内存卡,该系统的总体带宽可明显高于 20.83GBytes/sec,并且,如果使用五个或更多个内存卡,则可超过 100GBytes/sec。

[0151] 参考图 17,基本内存周期由通过与每一个区段相关联的命令总线启动内存的存储体 Bank A 而被初始化。这在图 17 的示例中是在时钟周期 1 中发生的。四个时钟周期之后,在时钟周期 5,激活命令被提供给存储体 Bank B。在其四个周期之后,在时钟周期 9,存储体 Bank C 被激活,并且在四个周期之后,在时钟周期 13,存储体 Bank D 被激活。由于存储体 Bank A 将在周期 6 和周期 19 之间进行预充电,在发出后续的激活 Bank A 命令时,这种可选性地激活不同的存储体的手段考虑到了不会被妨碍的任意的物理预充电时间。

[0152] 对于不同存储体中的每一个,在存储体被实施激活之后的五个周期,将针对该特定的存储体推断出指示数据传输是 READ 还是 WRITE 的命令。因此,这是在 Bank A 的时钟周期 6、Bank B 的时钟周期 10、Bank C 的时钟周期 14,以及 Bank D 的时钟周期 18 发生的。

[0153] 在提供该传输是 READ 还是 WRITE 的指示之后的五个时钟周期,数据将在四个连续周期中被提供给数据线。因为是以双倍的数据速率(DDR)传输的,所以每一个内存时钟周期将传输两个字。因此,对于每一个存储体,在整个基本内存周期内将传输八个 288 比特字数据(图 17 中的数据总线的符号, W0, 包含两个 288 比特字)。

[0154] 根据讨论,在基本内存周期的数据传输过程期间,下一个内存周期的基本内存周期将通过断定出在时钟周期 19 时用于 Bank A 的激活命令而被初始化。在剩余的后续基本内存周期后跟随的是前面以上关于第一基本内存周期所讨论的。

[0155] 正如以上所指示的,每一个基本内存周期将传输完整的 1KB 数据。在一个实施例中,该系统提供了“半写入(half-write)”周期,在此仅传输 512 字节。对于这样的“半写入”周期,仅断定出对两个连续的存储体(例如, Bank A 和 B 或者 Bank C 和 D)的激活。在该周期期间其它存储体将保持空闲。用于该传输的 TARGET ADDRESS 将确定四个存储体中的哪两个被激活用于该传输。

[0156] 以上实施例是说明性的,并不构成限制感。还可以利用以上所说明的本发明的一个或多个方面设计出其它的和进一步的实施例,而不背离所公开的实施例的精神。而且,在此所说明的步骤的顺序可按照不同的顺序出现,除非另外特别限制。在此所说明的这些不同的步骤可与其它步骤合并,插入一定的步骤,和 / 或分裂成多个步骤。类似地,元件已经在功能上进行了说明,而且可被实施成分开组件,或可被合并成具有多个功能的组件。

[0157] 在优选的和其它的实施例的上下文中已经说明了所公开的实施例,而并非说明了本发明的每一个实施例。对于那些本领域普通技术人员来说,所说明的实施例的明显的变形和可替换也是可用的。本公开的和未公开的实施例并非旨在限制或约束申请者所考虑的本发明的应用范围。申请者意图完全地保护落入随附的权利要求书的等价物的范围或程度的所有这样的变形和改进。

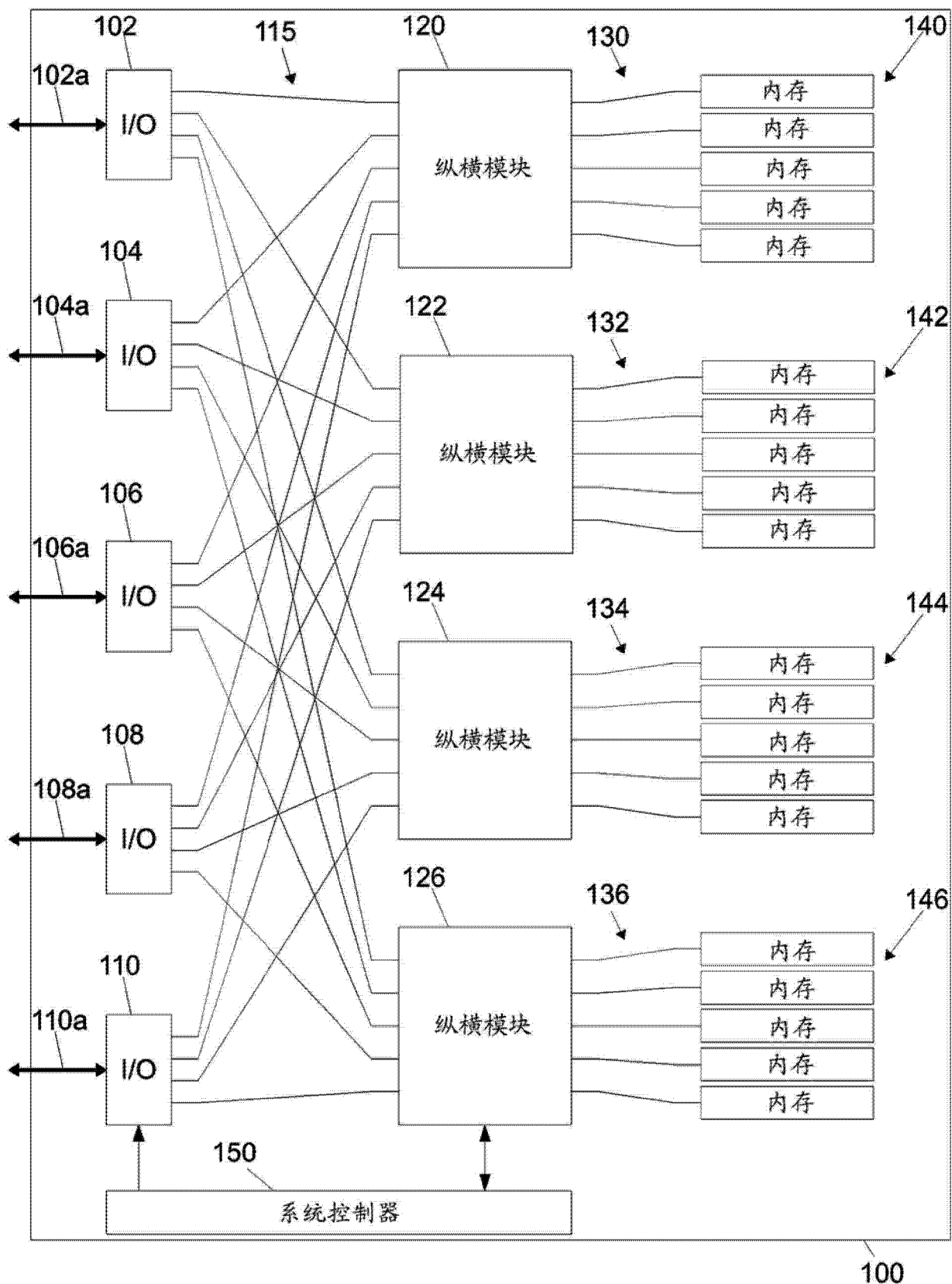


图 1

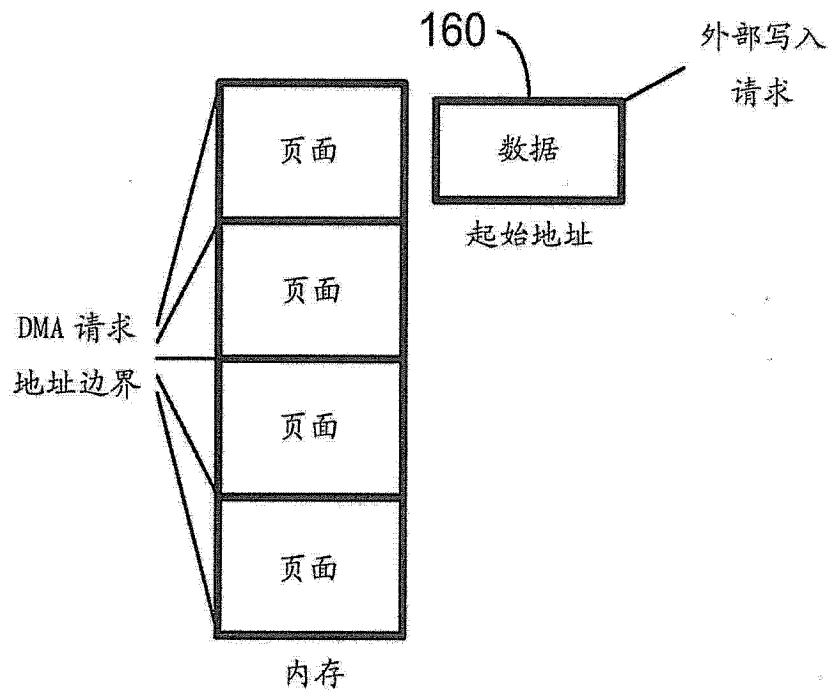


图 1A

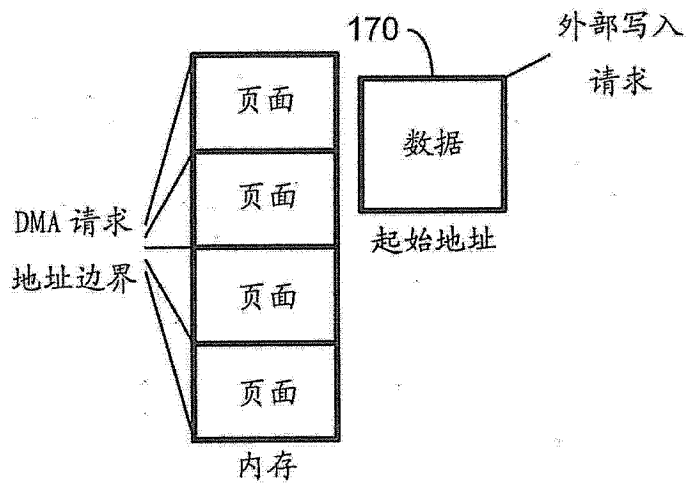


图 1B

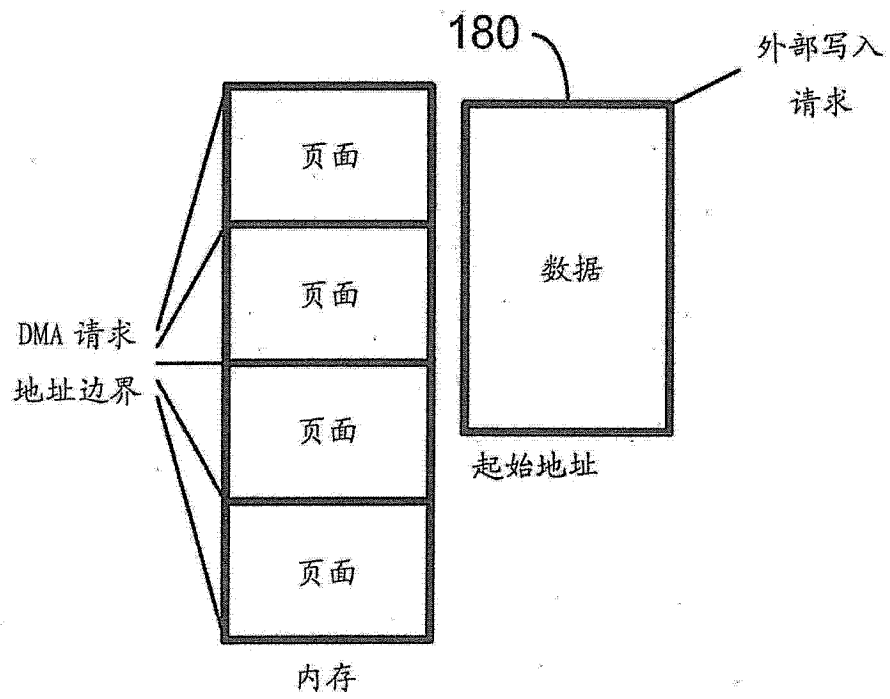


图 1C

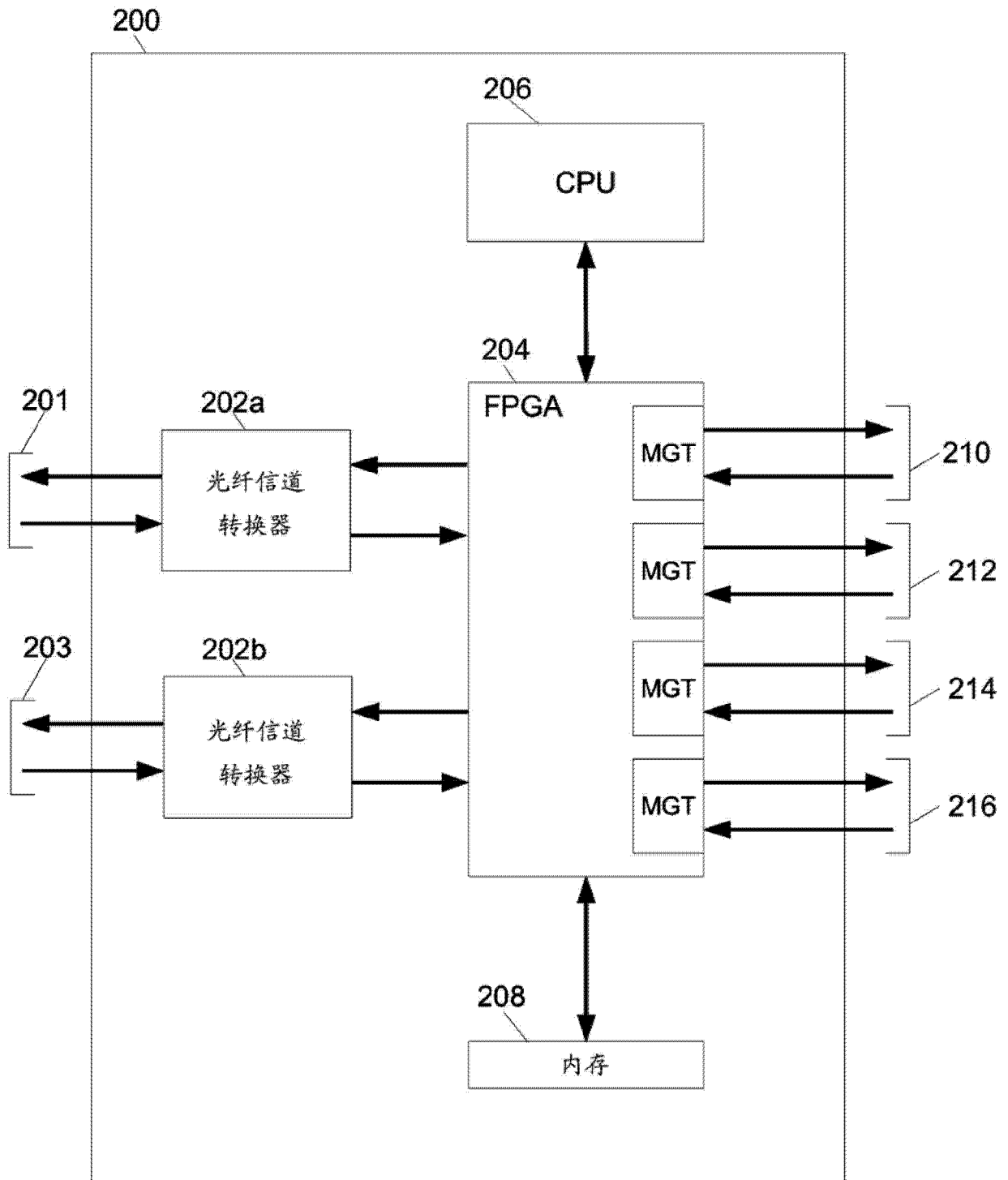


图 2

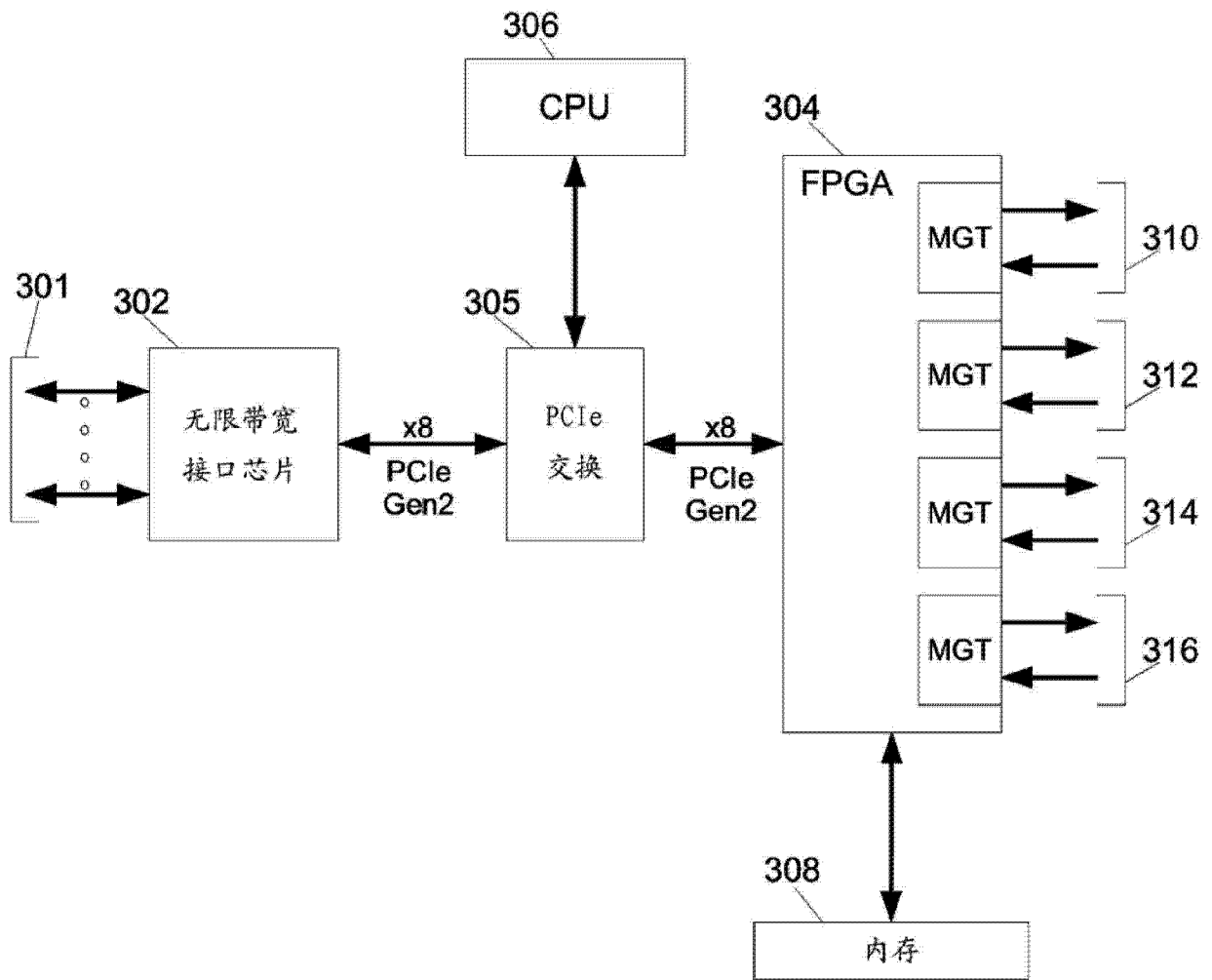


图 3

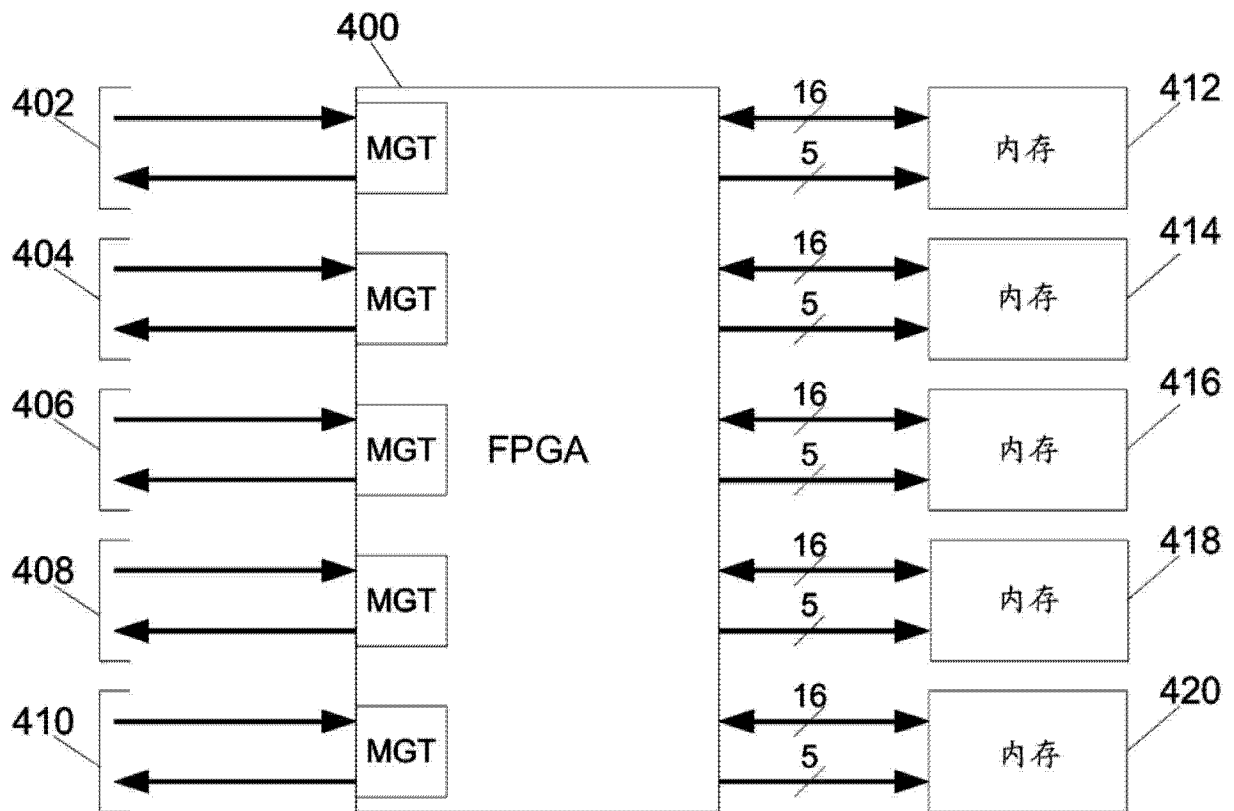


图 4

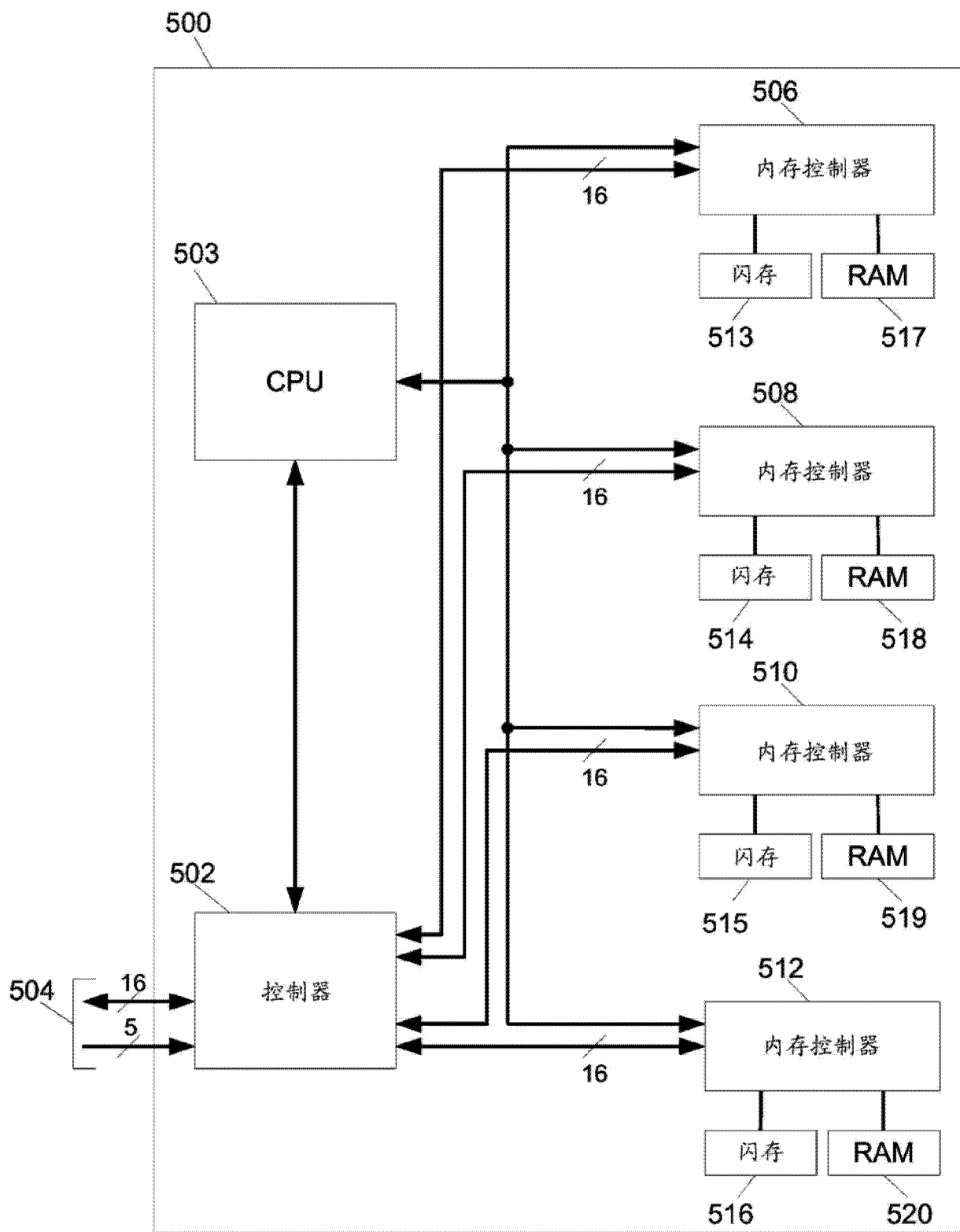


图 5

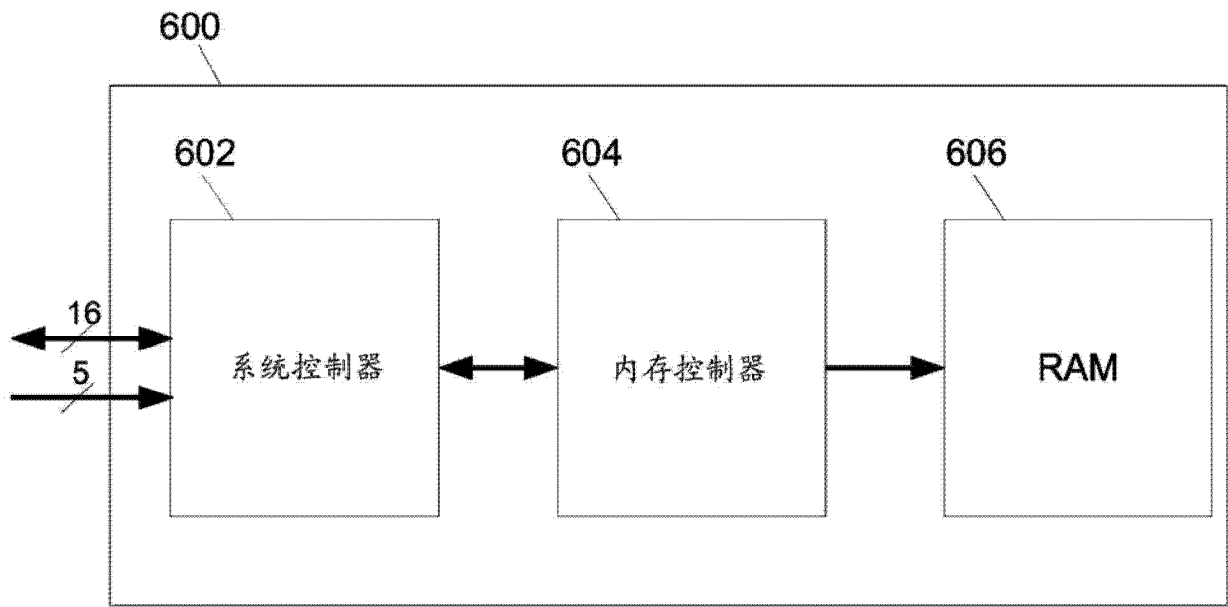


图 6

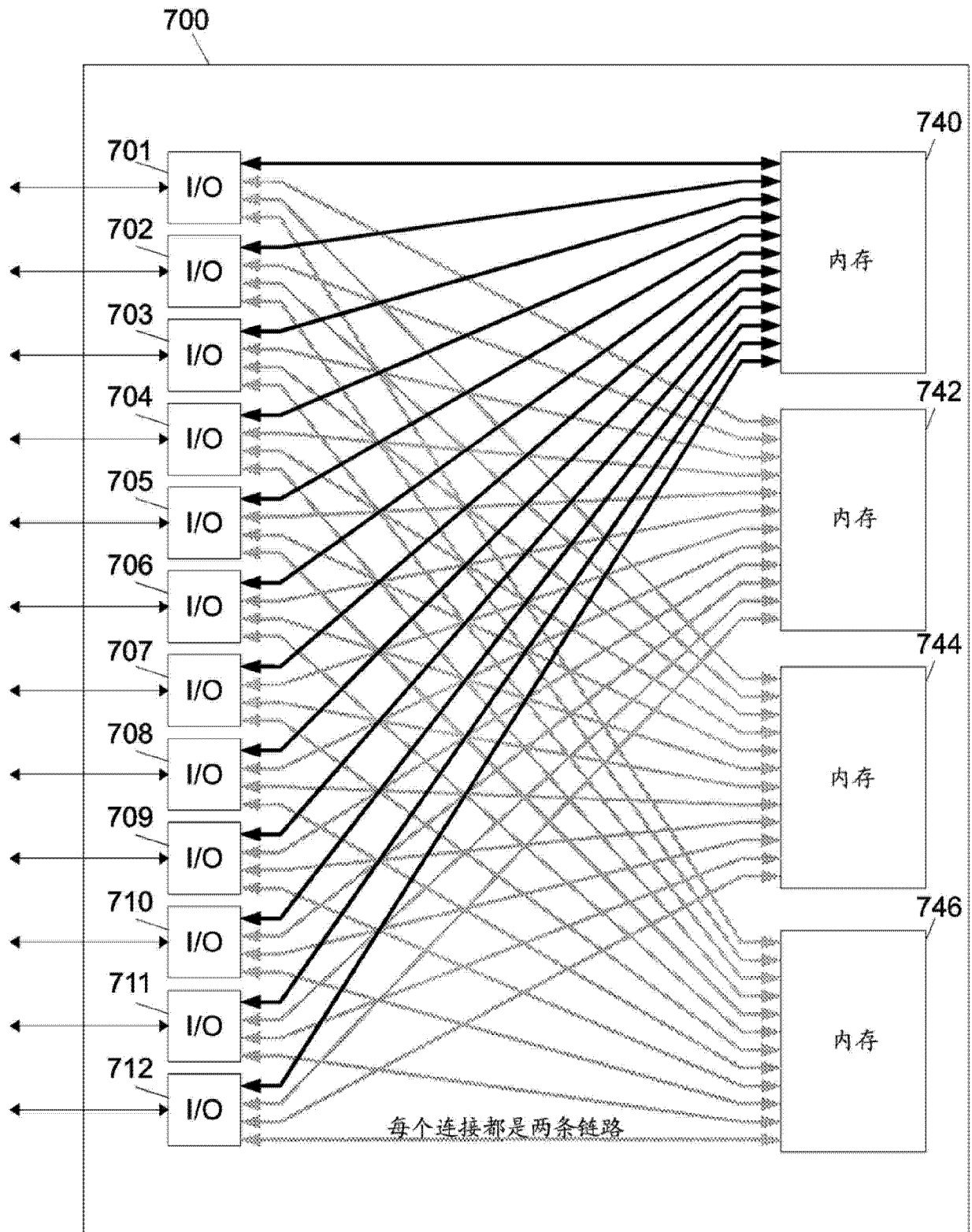


图 7

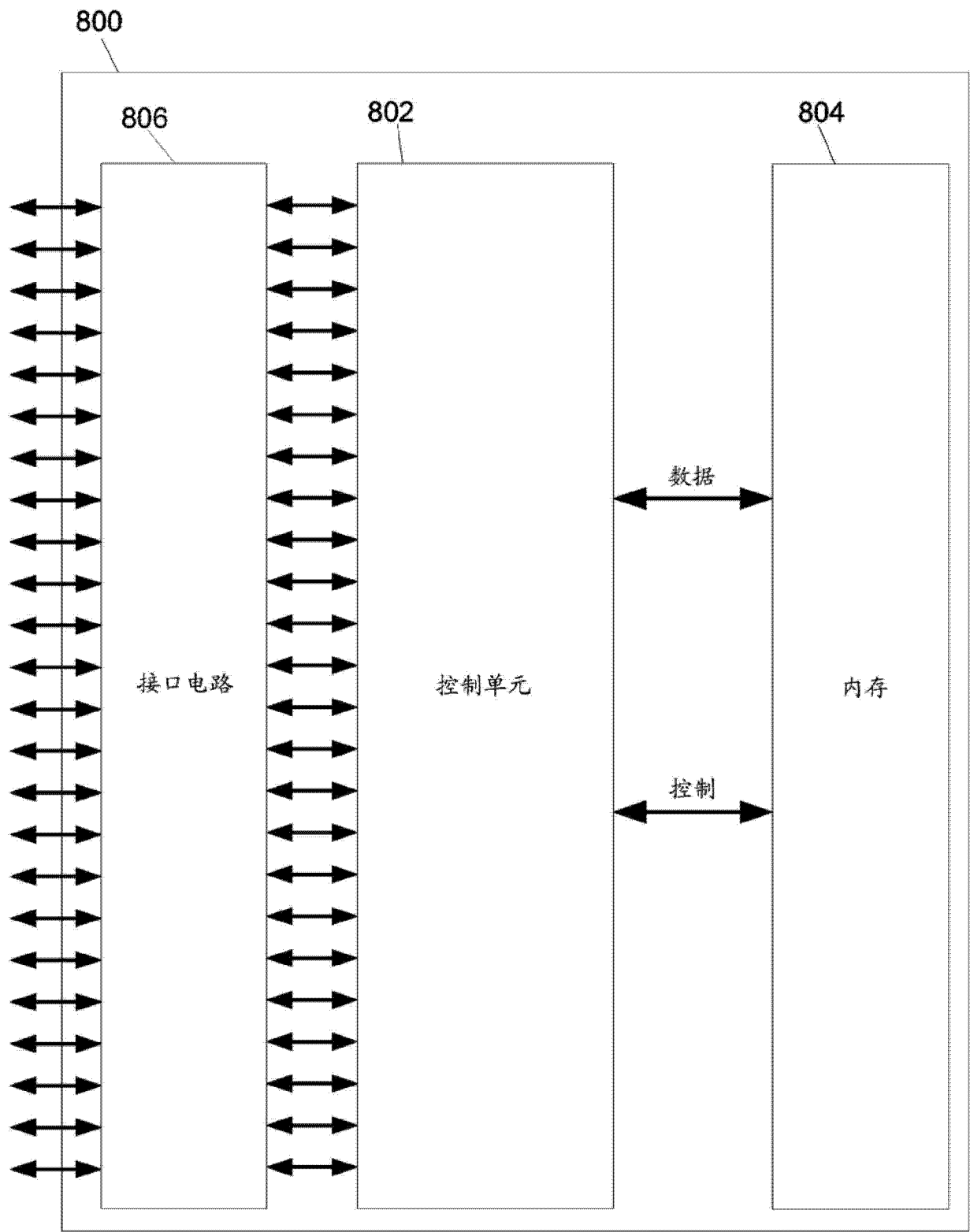


图 8

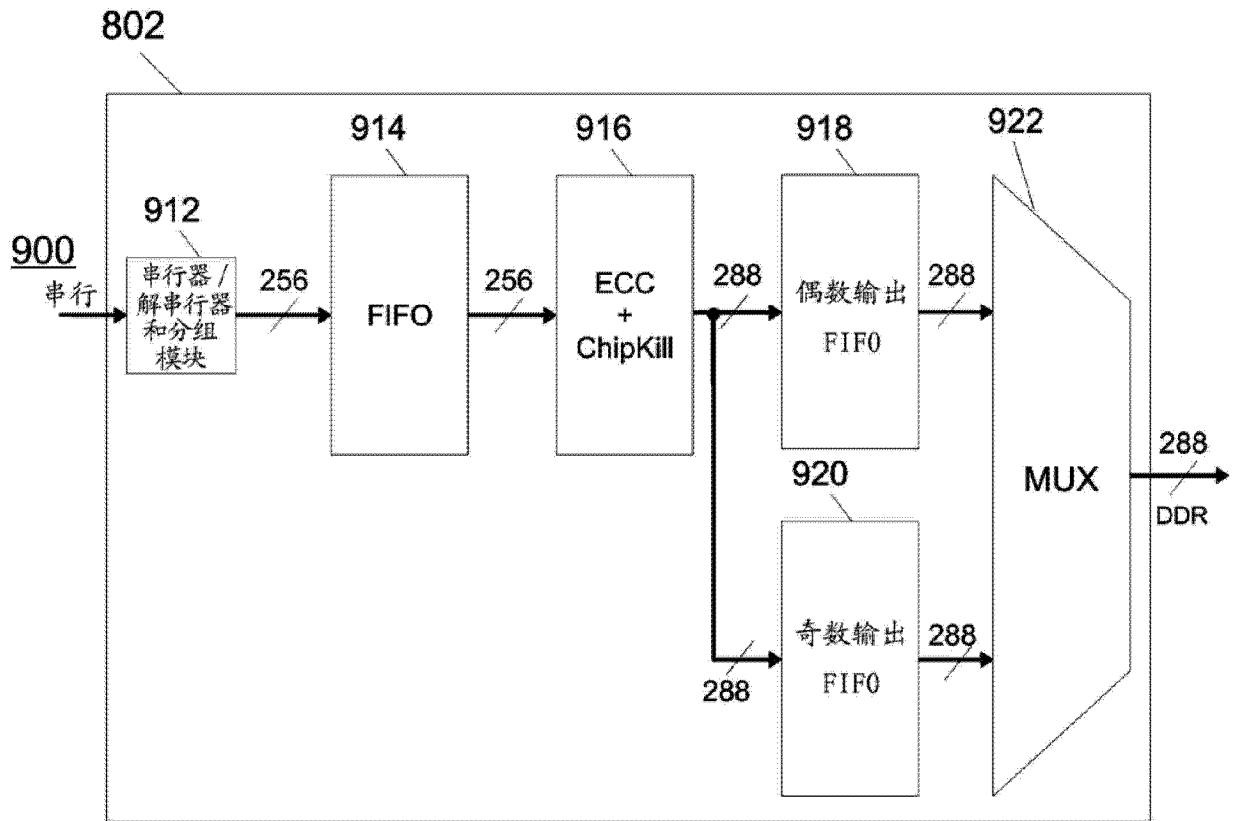


图 9

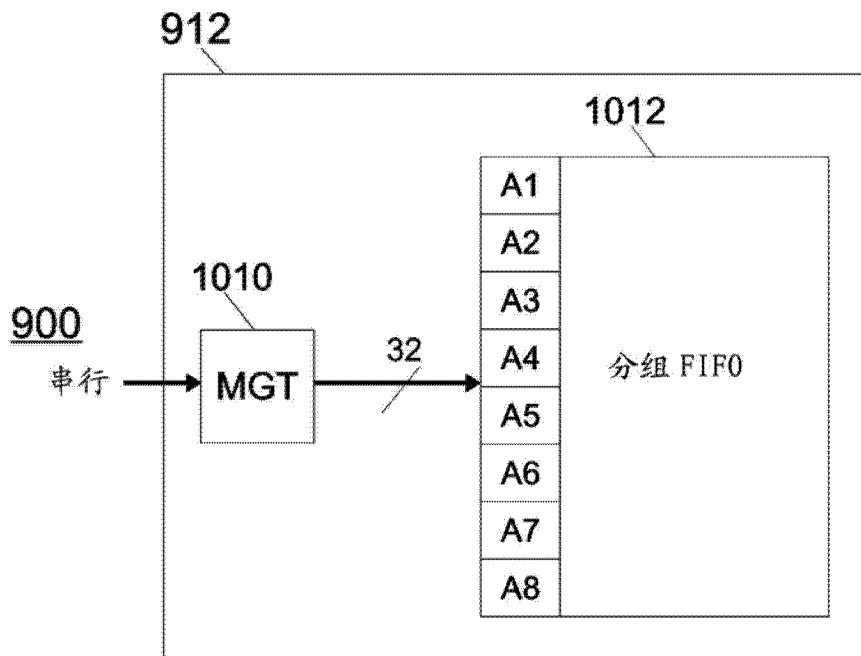


图 10A

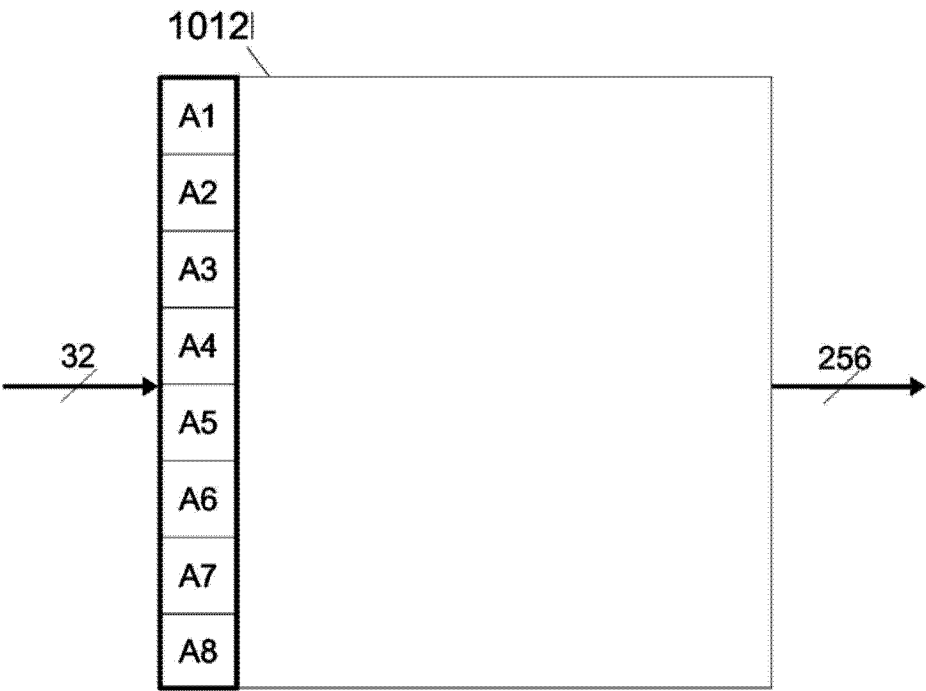


图 10B

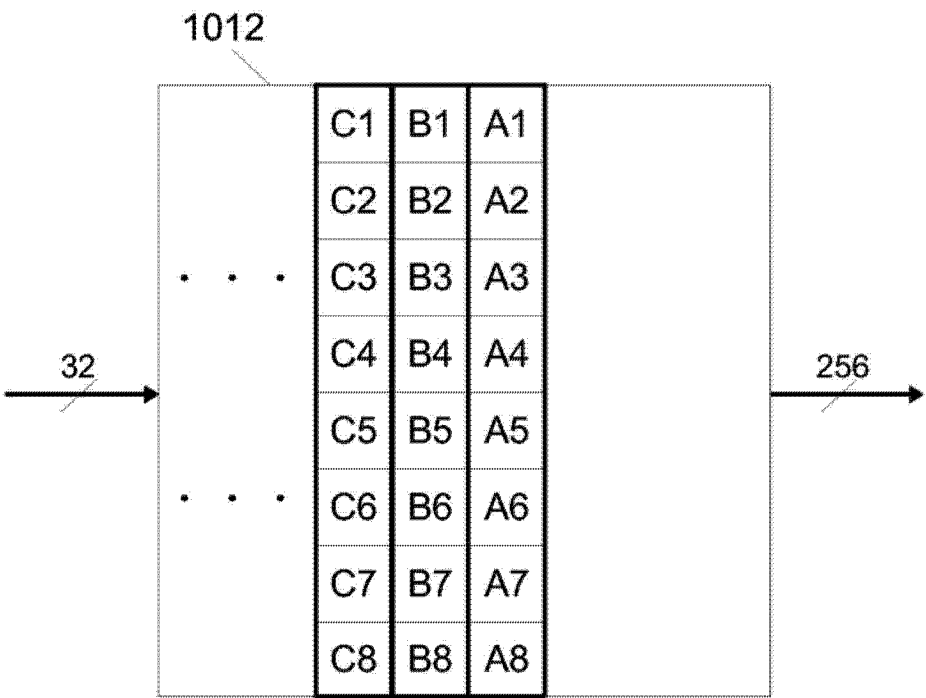


图 10C

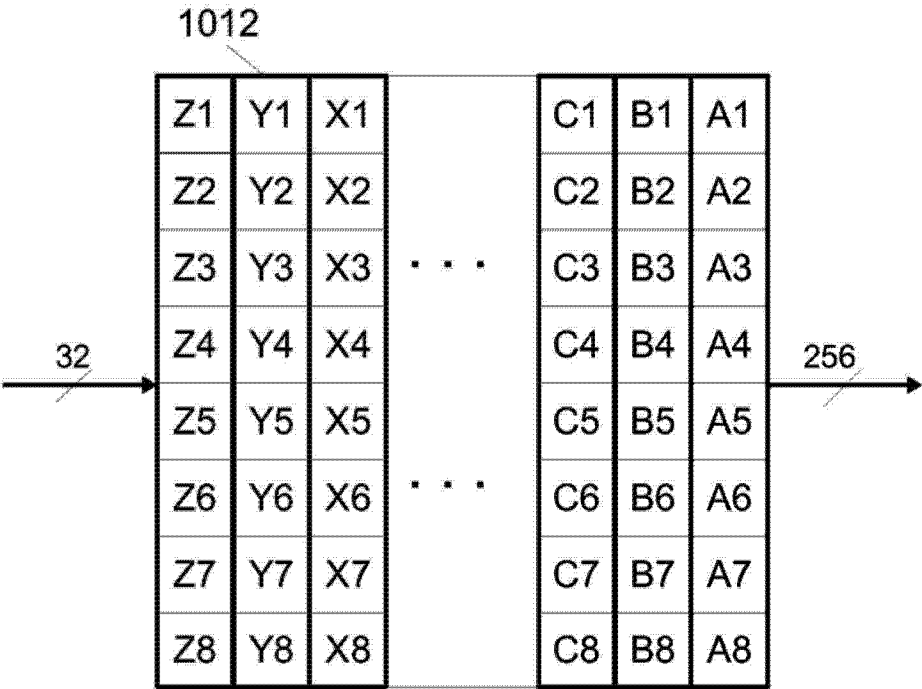


图 10D

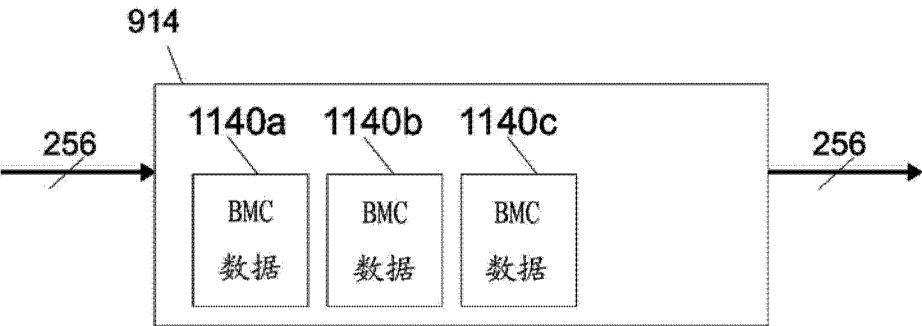


图 11

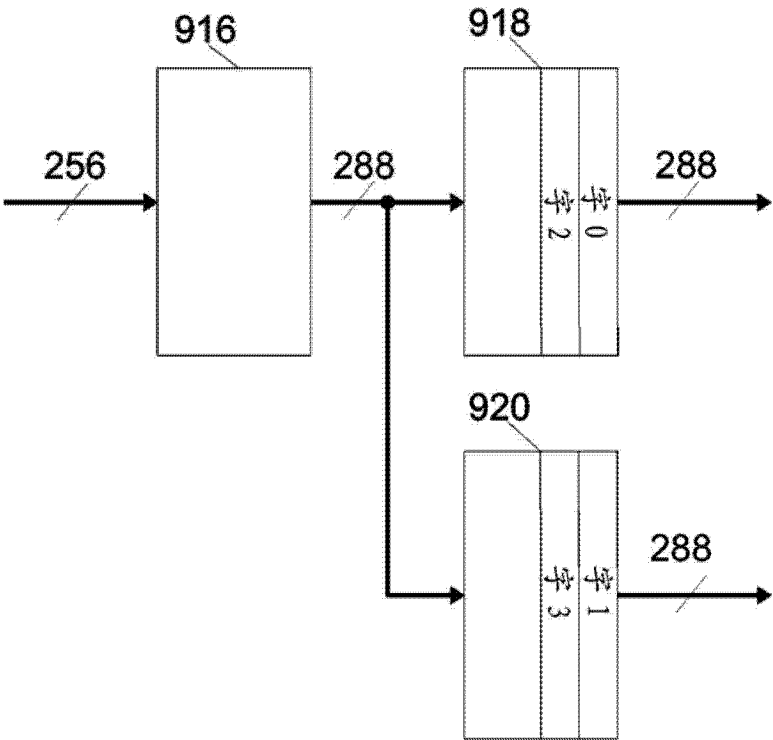


图 12

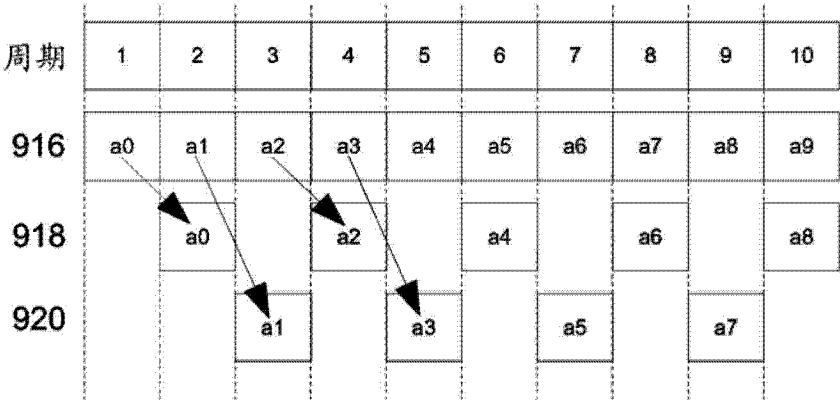


图 12A

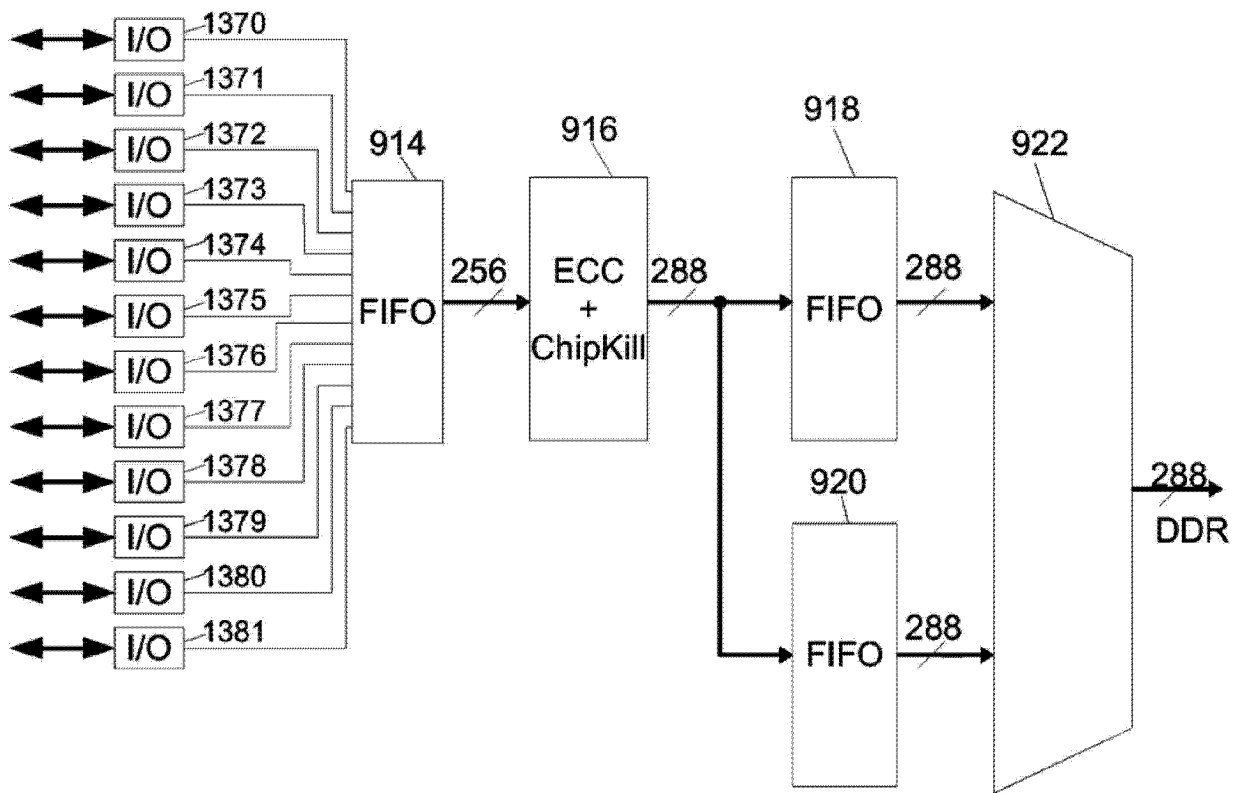


图 13A

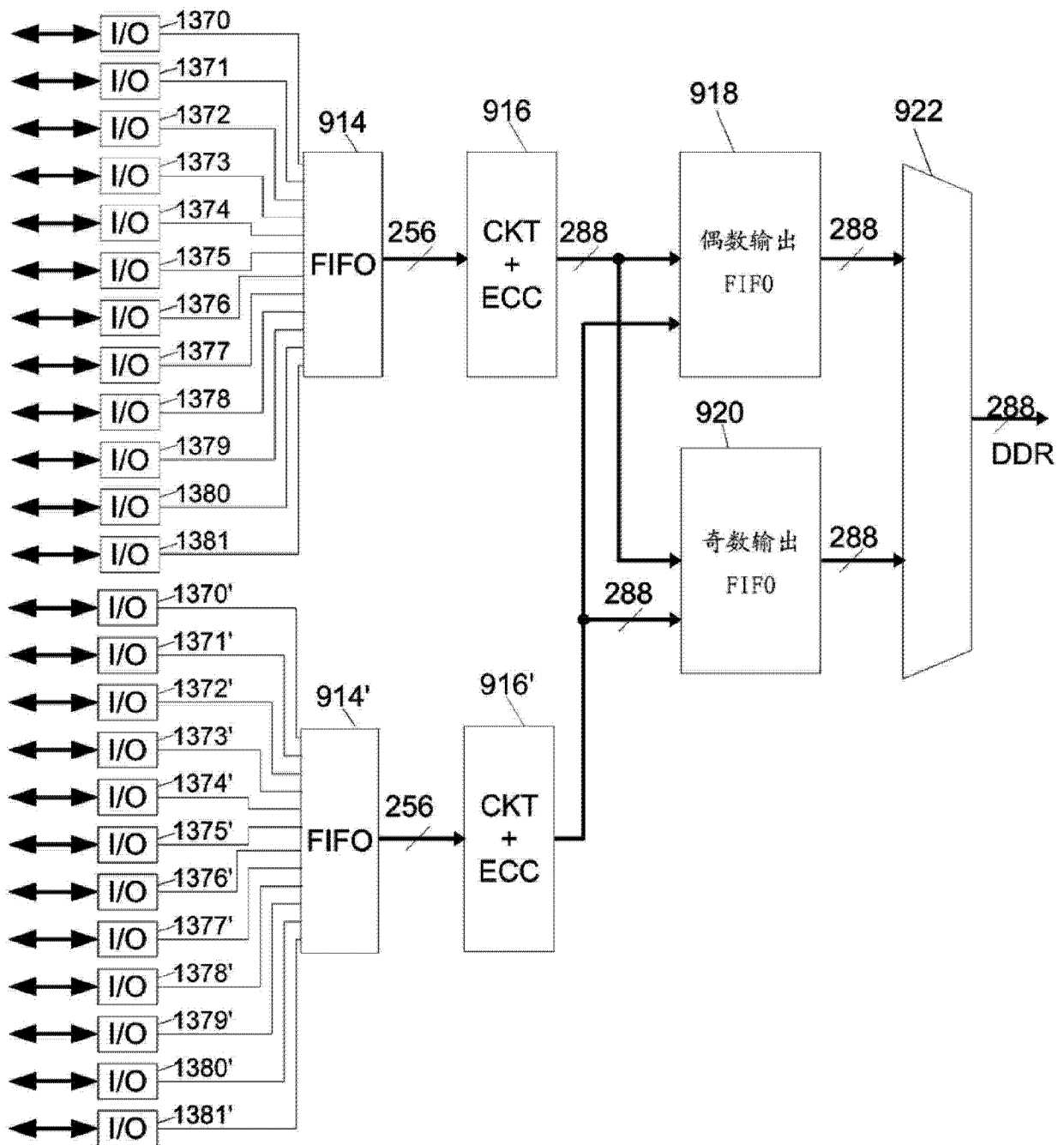


图 13B

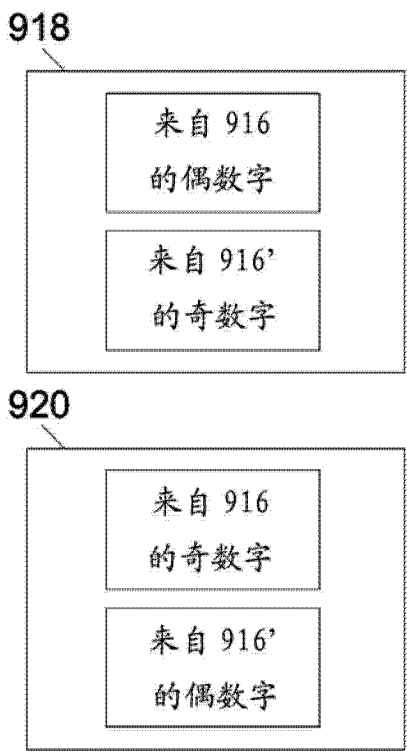


图 14A

周期	1	2	3	4	5	6	7	8	9	10
916	a0	a1	a2	a3	a4	a5	a6	a7	a8	a9
916'	b0	b1	b2	b3	b4	b5	b6	b7	b8	b9
918		a0	b1	a2	b3	a4	b5	a6	b7	a8
920		b0	a1	b2	a3	b4	a5	b6	a7	b8

图 14B

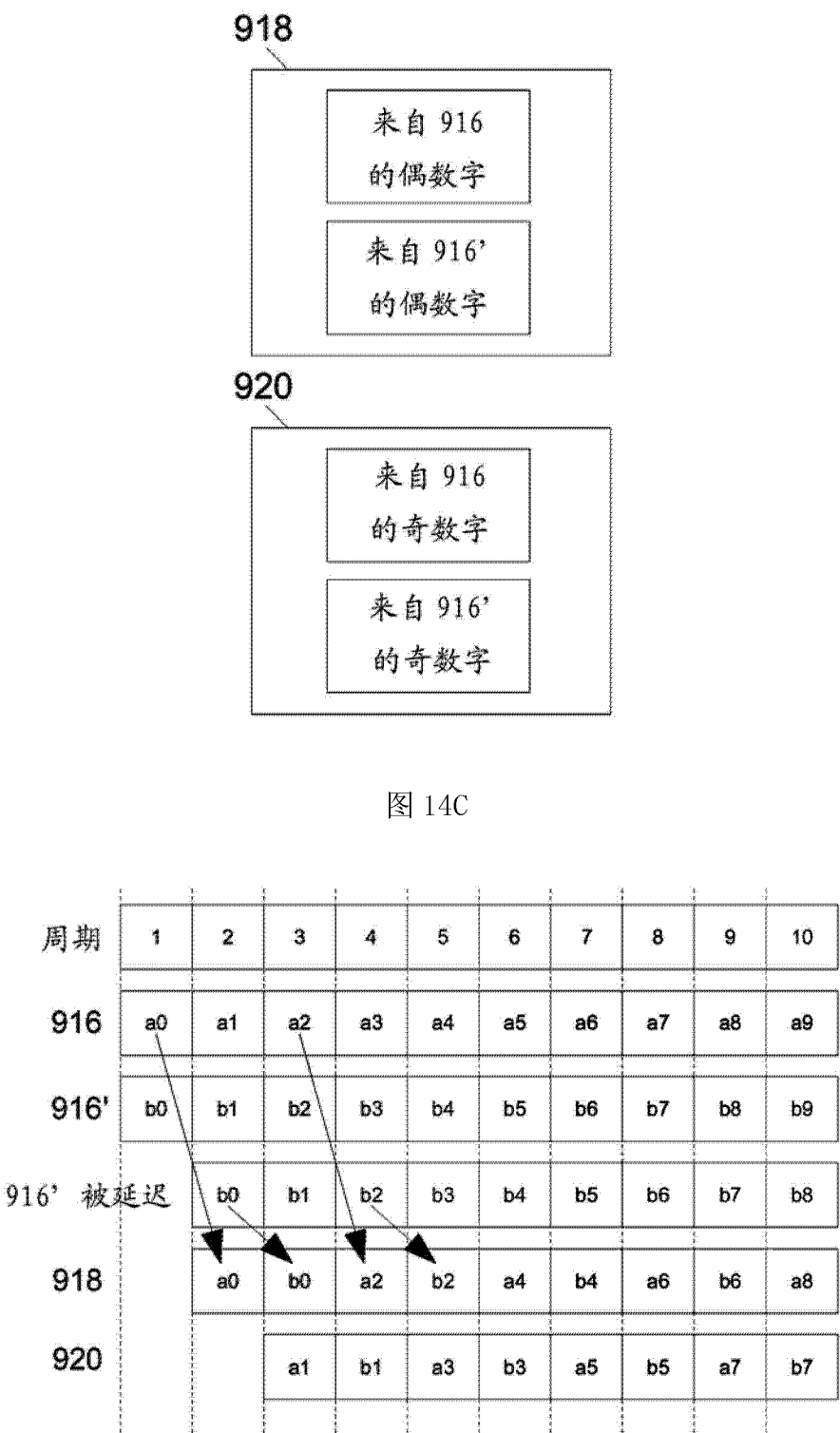


图 14C

图 14D

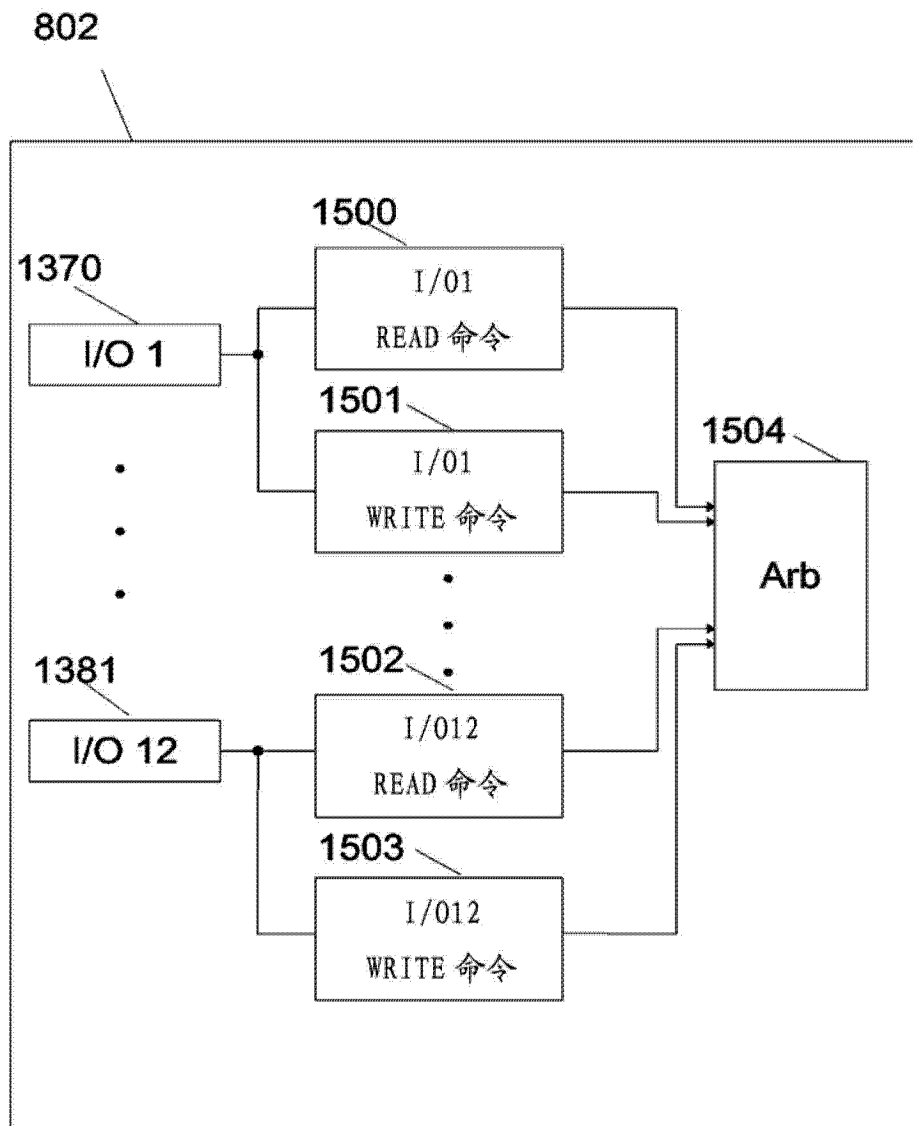


图 15

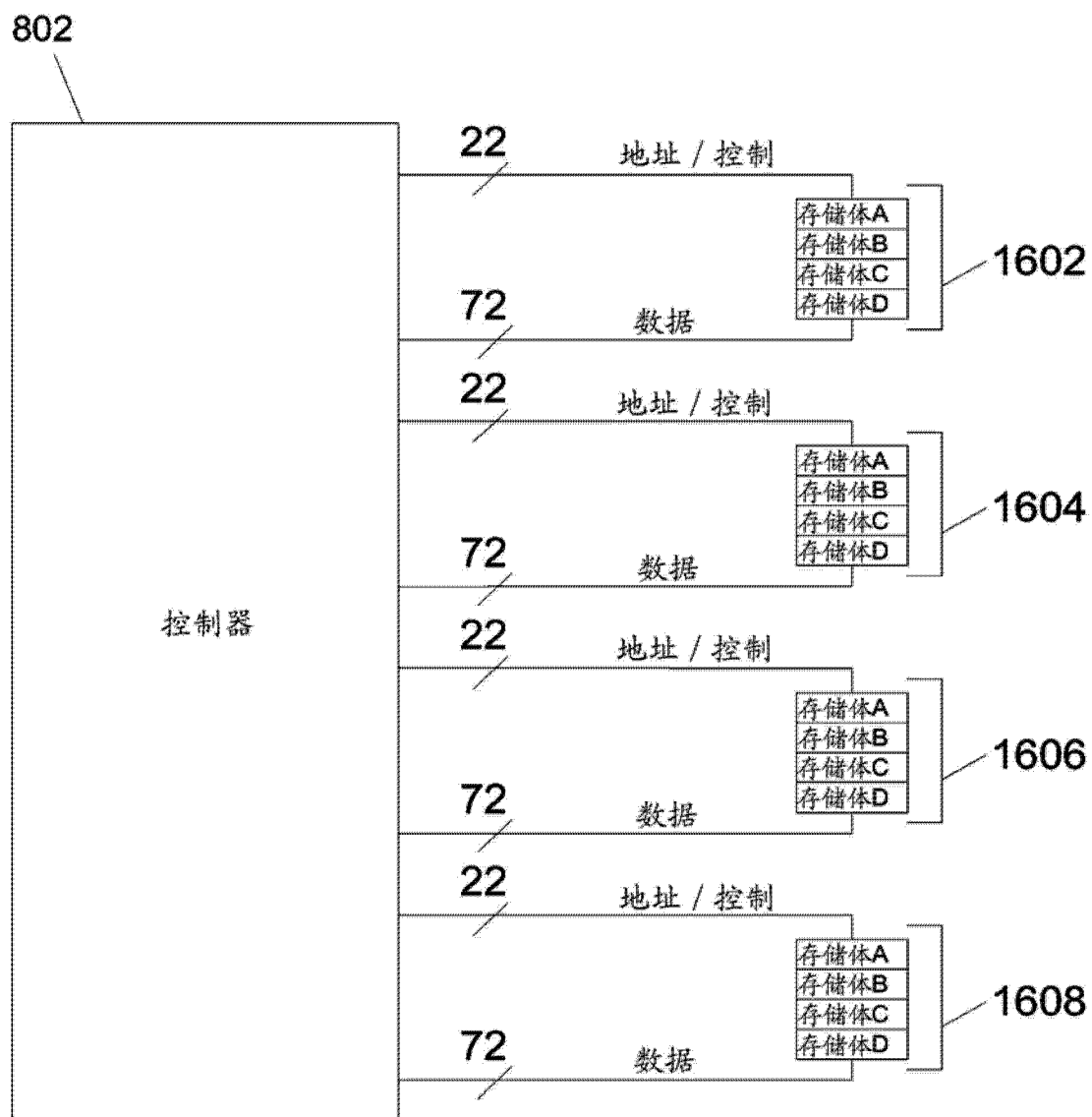


图 16

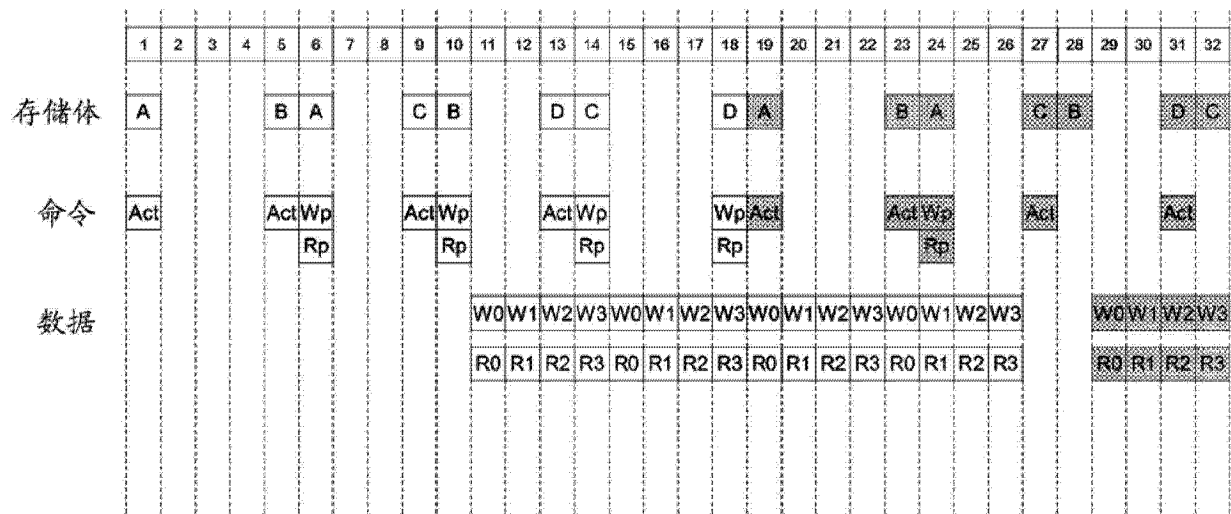


图 17