



(12) 发明专利申请

(10) 申请公布号 CN 117561542 A

(43) 申请公布日 2024. 02. 13

(21) 申请号 202280045630.3

R • W • 席尔瓦

(22) 申请日 2022.10.28

(74) 专利代理机构 上海专利商标事务所有限公司 31100

(30) 优先权数据

专利代理师 徐倩 黄嵩泉

17/561,666 2021.12.23 US

(85) PCT国际申请进入国家阶段日

(51) Int.Cl.

2023.12.26

G06T 9/00 (2006.01)

(86) PCT国际申请的申请数据

PCT/US2022/048288 2022.10.28

(87) PCT国际申请的公布数据

W02023/121770 EN 2023.06.29

(71) 申请人 英特尔公司

地址 美国加利福尼亚州

(72) 发明人 K • A • 谢尔森 P • 萨蒂

V • 克里希南 A • 纳瓦勒

A • R • 阿普 A • 考克

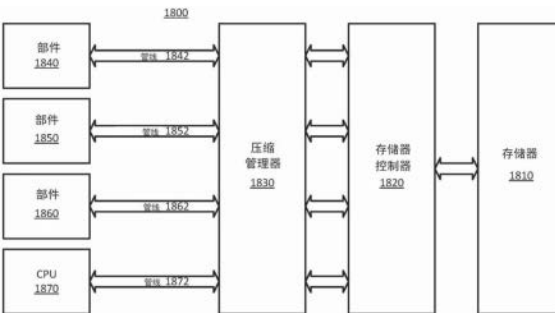
权利要求书3页 说明书42页 附图31页

(54) 发明名称

用于通用可消耗压缩的统一无状态压缩系统

(57) 摘要

系统包括存储被嵌入在压缩数据中的压缩格式信息的压缩引擎。压缩格式信息可以被包括在包括压缩控制表面 (CCS) 信息的报头中。系统包括共享存储器, 该共享存储器用于为多个硬件管线存储压缩数据, 其中压缩数据的各个块具有共同存储器占用空间和压缩报头。压缩引擎可以压缩要存储在共享存储器中的数据, 这包括生成报头。压缩引擎可以对从共享存储器读取的数据进行解压缩, 这包括从报头标识压缩格式。



1. 一种图形处理器, 包括:

存储器控制器, 用于耦合到用于为多个图形部件存储压缩数据的共享存储器, 其中所述压缩数据的块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面 (CCS) 信息的报头; 以及

压缩引擎, 用于压缩要存储在所述共享存储器中的数据, 以及对从所述共享存储器读取的数据进行解压缩, 其中压缩要存储的数据包括生成所述报头, 并且其中对读取的数据进行解压缩包括从所述报头标识压缩格式。

2. 如权利要求1所述的图形处理器, 其中, 所述压缩格式信息包括用于指示压缩类型的查找表的值。

3. 如权利要求1至2中任一项所述的图形处理器, 其中, 所述压缩引擎包括压缩器/解压缩器核心以及用于存储CCS信息的控制平面缓存。

4. 如权利要求3所述的图形处理器, 其中, 所述压缩引擎用于从所述多个图形部件中的一个图形部件接收要被压缩的数据, 包括接收到的数据要以无状态格式被压缩的指示符, 并且响应于所述指示符, 所述压缩引擎用于生成具有压缩信息的所述报头。

5. 如权利要求1至4中任一项所述的图形处理器, 其中, 所述压缩数据的块具有统一存储器布局。

6. 如权利要求5所述的图形处理器, 其中, 所述统一存储器布局包括行为主格式布局。

7. 如权利要求1至6中任一项所述的图形处理器, 进一步包括:

图形部件之间的通信结构, 其中所述通信结构用于基于块压缩比支持多个数据传输大小。

8. 如权利要求7所述的图形处理器, 其中, 所述通信结构支持压缩数据的传输和未压缩数据的传输。

9. 如权利要求1至8中任一项所述的图形处理器, 其中, 所述压缩引擎包括第一压缩引擎, 并且进一步包括:

第二压缩引擎, 其中所述第一压缩引擎和所述第二压缩引擎位于所述图形处理器的不同层处。

10. 一种计算机系统, 包括:

中央处理单元 (CPU), 用于执行一般计算;

图形单元, 耦合到所述CPU, 所述图形单元包括:

存储器控制器, 用于耦合到用于为多个图形部件存储压缩数据的共享存储器, 其中所述压缩数据的块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面 (CCS) 信息的报头; 以及

压缩引擎, 用于压缩要存储在所述共享存储器中的数据, 以及对从所述共享存储器读取的数据进行解压缩, 其中压缩要存储的数据包括生成所述报头, 并且其中对读取的数据进行解压缩包括从所述报头标识压缩格式。

11. 如权利要求10所述的计算机系统, 其中, 所述压缩格式信息包括用于指示压缩类型的查找表的值。

12. 如权利要求10至11中任一项所述的计算机系统, 其中, 所述压缩引擎用于从多个图形客户端单元中的一个图形客户端单元接收要被压缩的数据, 包括接收到的数据要以无状

态格式被压缩的指示符,并且响应于所述指示符,所述压缩引擎用于生成具有所述压缩格式信息和所述CCS信息的所述报头。

13.如权利要求10至12中任一项所述的计算机系统,其中,所述压缩数据的块具有统一存储器布局,其中所述统一存储器布局包括行为主格式布局。

14.如权利要求10至13中任一项所述的计算机系统,进一步包括:

硬件管线之间的通信结构,其中所述通信结构用于基于块压缩比支持多个数据传输大小。

15.如权利要求14所述的计算机系统,其中,所述通信结构支持压缩数据的传输和未压缩数据的传输。

16.如权利要求10至15中任一项所述的计算机系统,其中,所述多个硬件管线包括以下各项中的一项或多项:

媒体管线;

颜色管线;

显示管线;

视频捕捉管线;或

人工智能(AI)加速器管线。

17.如权利要求10至16中任一项所述的计算机系统,其中,所述压缩引擎包括第一压缩引擎,并且进一步包括:

第二压缩引擎,其中所述第一压缩引擎和所述第二压缩引擎位于所述图形单元的不同层处。

18.如权利要求10至17中任一项所述的计算机系统,进一步包括:

主系统存储器,所述主系统存储器与所述图形单元的所述共享存储器分开;

其中,所述压缩引擎用于提供所述主系统存储器中的数据的压缩和解压缩,以供所述CPU访问。

19.一种用于压缩的方法,包括:

将压缩数据在针对多个图形部件的共享存储器中存储为压缩数据的块,所述压缩数据的块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面(CCS)信息的报头;

压缩要存储在所述共享存储器中的数据,包括生成所述报头;以及

对从所述共享存储器读取的数据进行解压缩,包括从所述报头标识压缩格式。

20.如权利要求19所述的方法,进一步包括:

使用通信结构在图形部件之间传送压缩数据的块,所述通信结构基于块压缩比支持多个数据传输大小。

21.如权利要求19至20中任一项所述的方法,其中,所述通信结构支持压缩数据的传输和未压缩数据的传输。

22.如权利要求19至21中任一项所述的方法,进一步包括:

从所述多个图形部件中的一个图形部件接收要被压缩的数据,包括接收接收到的数据要以无状态格式被压缩的指示符;以及

响应于所述指示符,生成具有压缩信息的所述报头。

23. 如权利要求19至22中任一项所述的方法,其中,所述压缩数据的块具有行为主格式布局。

用于通用可消耗压缩的统一无状态压缩系统

优先权要求

[0001] 本申请要求2021年12月23日提交的题为“UNIFIED STATELESS COMPRESSION SYSTEM FOR UNIVERSALLY CONSUMABLE COMPRESSION (用于通用可消耗压缩的统一无状态压缩系统)”的美国专利申请第17/561,666号的优先权权益。

技术领域

[0002] 描述总体上涉及图形处理,并且更具体的描述涉及图形子系统内的压缩。

背景技术

[0003] 在计算机图形系统中越来越多地应用压缩来减少传输大量数据的影响。压缩是通过将数据应用压缩算法减少表示信息所需的数据量来完成的。图形处理系统中的部件之间的压缩可以处理表面,表面是由GPU(graphics processing unit,图形处理单元)处理的二维(two dimensional,2D)图像。

[0004] 传统上,移动压缩数据需要有关像素格式和表面使用的信息(诸如,颜色、z数据、线性缓冲器)。固定存储器(pinned memory)传统上用于具有特定特性的表面(其中存储器的某个区域用于纹理),从而使目标无法被移出存储器。然而,有限量的存储器可以被固定。动态渲染目标传统上是被固定的。由于系统涉及多个产生和消耗压缩数据的部件,因此部件之间缺乏类似的接口可导致显著的性能影响。

附图说明

[0005] 以下描述包括对附图的讨论,这些附图具有通过实现的示例的方式给出的例示。这些附图应被理解成作为示例,而不是作为限制。如本文中所使用,对一个或多个示例的引用应被理解为描述包括在本发明的至少一种实现方式中的特定特征、结构或特性。本文中出现的诸如“在一个示例中”或“在替代性的示例中”之类的短语提供了本发明的实现方式的示例,并且不一定全部指代相同的实现方式。然而,它们也不一定是相互排他的。

[0006] 图1是根据示例的处理系统的框图。

[0007] 图2A-图2D图示由本文中描述的示例提供的计算系统和图形处理器。

[0008] 图3A-图3C图示由本文中描述的示例提供的附加的图形处理器和计算加速器体系结构的框图。

[0009] 图4是根据一些示例的图形处理器的图形处理引擎的框图。

[0010] 图5A-图5B图示根据本文中描述的示例的线程执行逻辑,该线程执行逻辑包括在图形处理器核心中采用的处理元件的阵列。

[0011] 图6图示根据示例的附加的执行单元。

[0012] 图7是图示根据一些示例的图形处理器指令格式的框图。

[0013] 图8是图形处理器的另一示例的框图。

[0014] 图9A是图示根据一些示例的图形处理器命令格式的框图。

- [0015] 图9B是图示根据示例的图形处理器命令序列的框图。
- [0016] 图10图示根据一些示例的用于数据处理系统的示例性图形软件体系结构。
- [0017] 图11A是图示根据示例的可用于制造集成电路以执行操作的IP核心开发系统的框图。
- [0018] 图11B图示根据本文中描述的一些示例的集成电路封装组件的截面侧视图。
- [0019] 图11C图示封装组件,该封装组件包括连接到衬底的多个单元的硬件逻辑小芯片。
- [0020] 图11D图示根据示例的包括可互换小芯片的封装组件。
- [0021] 图12、图13A和图13B图示根据本文中所述的各示例的可以使用一个或多个IP核心制造的示例性集成电路和相关联的图形处理器。
- [0022] 图14图示具有向量单元和矩阵单元的图形处理器的示例。
- [0023] 图15图示具有并行计算集群的图形处理单元的示例。
- [0024] 图16图示具有用于在图形处理器之间传输的压缩数据分组的系统的示例。
- [0025] 图17图示图形压缩体系结构的示例。
- [0026] 图18图示提供通用可消耗的数据压缩的系统的示例。
- [0027] 图19图示具有可以传输压缩数据的数据结构的系统的示例。
- [0028] 图20图示具有分布式压缩和解压缩的系统的示例。
- [0029] 图21图示片上系统中具有统一压缩的系统的示例。
- [0030] 图22是生成通用可消耗的压缩数据的示例的流程图。
- [0031] 图23是对通用可消耗的压缩数据解压缩的示例的流程图。
- [0032] 接下来是某些细节和实现方式的描述,包括对附图的非限制性描述,这些附图可以描绘一些或所有示例,以及其他潜在实现方式。

具体实施方式

- [0033] 如本文所述,系统包括存储被嵌入在压缩数据中的压缩格式信息的压缩引擎。压缩格式信息可以被包括在包括压缩控制表面(compression control surface,CCS)信息的报头中。系统包括共享存储器,用于存储多个硬件管线的压缩数据,其中压缩数据块具有共同存储器占用空间和压缩报头。压缩引擎可以压缩要存储在共享存储器中的数据,包括生成报头。压缩引擎可以对从共享存储器读取的数据进行解压缩,包括从报头标识压缩格式。
- [0034] 将压缩信息嵌入在压缩数据中的系统可以实现可缩放的压缩体系结构,该体系结构允许在不同部件处进行本地最大化压缩,同时维持跨SOC(system on chip,片上系统)上的所有部件的兼容性。在某些用例中,体系结构可以允许与CPU(central processing unit,中央处理单元)共享压缩缓冲器。与CPU以及GPU共享的能力为AI(artificial intelligence,人工智能)工作负载或涉及CPU和GPU或其他加速器(它们必须在计算任务中进行协作以完成操作)之间大量数据传输的其他工作负载提供显著益处。
- [0035] 通用可消耗压缩的示例可以包括相机产生视频并将其存储在固定大小的缓冲器中的场景。媒体编码器可以将数据从初始缓冲器发送到另一缓冲器。诸如图形、媒体或显示器之类的部件可以消耗视频。因此,各种不同的部件可以提供用于存储的数据。在一个示例中,使用存储器侧缓存来压缩数据。通过分立设备中的存储器侧缓存,部件可以利用存储器侧缓存来创建压缩片。然而,为了存储器侧缓存的更高效使用,某些段可能需要数据在存储

器侧缓存之前被压缩。

[0036] 通用可消耗压缩使系统能够缩放,即使没有类似的接口。任何客户端单元或部件可以使用解压缩器来消耗压缩数据。存储在存储器中的数据可被消耗,而无需单独的元数据。

[0037] 系统概述

[0038] 图1是根据示例的处理系统100的框图。系统100可在以下各项中被使用:单处理器桌面型电脑系统、多处理器工作站系统或具有大量处理器102或处理器核心107的服务器系统。在一个示例中,系统100是被并入在片上系统(system-on-a-chip, SoC)集成电路内的处理平台,该片上系统(SoC)集成电路用于在移动设备、手持式设备或嵌入式设备中使用,诸如,用于在具有至局域网或广域网的有线或无线连接性的物联网(Internet-of-things, IoT)设备内使用。

[0039] 在一个示例中,系统100可包括以下各项,可与以下各项耦合,或可集成在以下各项内:基于服务器的游戏平台;游戏控制台,包括游戏和媒体控制台;移动游戏控制台、手持式游戏控制台或在线游戏控制台。在一些示例中,系统100是移动电话、智能电话、平板计算设备或移动互联网连接的设备(诸如,具有低内部存储容量的膝上型电脑)的部分。处理系统100也可包括以下各项,与以下各项耦合,或被集成在以下各项内:可穿戴设备,诸如,智能手表可穿戴设备;智能眼镜或服装,其利用增强现实(augmented reality, AR)或虚拟现实(virtual reality, VR)特征来增强,以提供视觉、音频或触觉输出来补充现实世界视觉、音频或触觉体验或以其他方式提供文本、音频、图形、视频、全息图像或视频、或触觉反馈;其他增强现实(AR)设备;或其他虚拟现实(VR)设备。在一些示例中,处理系统100包括电视机或机顶盒设备,或者是电视机或机顶盒设备的部分。在一个示例中,系统100可包括自动驾驶交通工具,与自动驾驶交通工具耦合,或集成在自动驾驶交通工具内,该自动驾驶交通工具诸如,公共汽车、拖拉机拖车、汽车、电机或电力循环、飞机或滑翔机(或其任何组合)。自动驾驶交通工具可使用系统100来处理在该交通工具周围感测到的环境。

[0040] 在一些示例中,一个或多个处理器102各自包括一个或多个处理器核心107,该一个或多个处理器核心107用于处理指令,这些指令当被执行时,执行用于系统和用户软件的操作。在一些示例中,一个或多个处理器核心107中的至少一个处理器核心被配置成用于处理特定的指令集109。在一些示例中,指令集109可促进复杂指令集计算(Complex Instruction Set Computing, CISC)、精简指令集计算(Reduced Instruction Set Computing, RISC)或经由超长指令字(Very Long Instruction Word, VLIW)的计算。一个或多个处理器核心107可以处理不同的指令集109,不同的指令集109可包括用于促进对其他指令集的仿真的指令。处理器核心107还可包括其他处理设备,诸如,数字信号处理器(Digital Signal Processor, DSP)。

[0041] 在一些示例中,处理器102包括缓存存储器104。取决于体系结构,处理器102可具有单个内部缓存或多级的内部缓存。在一些示例中,缓存存储器在处理器102的各种部件之间被共享。在一些示例中,处理器102也使用外部缓存(例如,第三级(L3)缓存或最后一级缓存>Last Level Cache, LLC)(未示出),可使用已知的缓存一致性技术在处理器核心107之间共享该外部缓存。寄存器堆106可附加地被包括在处理器102中,并且可包括用于存储不同类型的数据的不同类型的寄存器(例如,整数寄存器、浮点寄存器、状态寄存器以及指令

指针寄存器)。一些寄存器可以是通用寄存器,而其他寄存器可以专用于处理器102的设计。

[0042] 在一些示例中,一个或多个处理器102与一个或多个接口总线110耦合,以在处理器102与系统100中的其他部件之间传送通信信号,诸如,地址、数据、或控制信号。在一个示例中,接口总线110可以是处理器总线,诸如,直接媒体接口(Direct Media Interface, DMI)总线的某个版本。然而,处理器总线不限于DMI总线,并且可包括一个或多个外围部件互连总线(例如,PCI、PCI快速)、存储器总线或其他类型的接口总线。在一个示例中,(一个或多个)处理器102包括集成存储器控制器116和平台控制器中枢130。存储器控制器116促进存储器设备与系统100的其他部件之间的通信,而平台控制器中枢(platform controller hub, PCH) 130提供经由本地I/O总线至I/O设备的连接。

[0043] 存储器设备120可以是动态随机存取存储器(dynamic random-access memory, DRAM)设备、静态随机存取存储器(static random-access memory, SRAM)设备、闪存设备、相变存储器设备或具有合适的性能以充当进程存储器的某个其他存储器设备。在一个示例中,存储器设备120可以作为用于系统100的系统存储器来操作,以存储数据122和指令121,用于在一个或多个处理器102执行应用或进程时使用。存储器控制器116也与任选的外部图形处理器118耦合,该任选的外部图形处理器118可与处理器102中的一个或多个图形处理器108通信以执行图形操作和媒体操作。在一些示例中,可由加速器112辅助图形操作、媒体操作和/或计算操作,该加速器112是可被配置成用于执行专业的图形操作、媒体操作或计算操作的集合的协处理器。例如,在一个示例中,加速器112是用于优化机器学习或计算操作的矩阵乘法加速器。在一个示例中,加速器112是光线追踪加速器,该光线追踪加速器可用于与图形处理器108协同地执行光线追踪操作。在一个示例中,可替代加速器112使用外部加速器119,或可与加速器112协同地使用外部加速器119。

[0044] 在一些示例中,显示设备111可连接到(一个或多个)处理器102。显示设备111可以是以下各项中的一项或多项:内部显示设备,如在移动电子设备或膝上型电脑设备中;或经由显示接口(例如,显示端口、嵌入式显示端口、MIPI、HDMI等)外接的外部显示设备。在一个示例中,显示设备111可以是头戴式显示器(head mounted display, HMD),诸如,用于在虚拟现实(VR)应用或增强现实(AR)应用中使用的立体显示设备。

[0045] 在一些示例中,平台控制器中枢130使外围设备能够经由高速I/O总线连接到存储器设备120和处理器102。I/O外围设备包括但不限于音频控制器146、网络控制器134、固件接口128、无线收发器126、触摸传感器125、数据存储设备124(例如,非易失性存储器、易失性存储器、硬盘驱动器、闪存、NAND、3D NAND、3D Xpoint等)。数据存储设备124可以经由存储接口(例如,SATA(serial advanced technology attachment, 串行高级技术附件))或经由外围总线(诸如,外围部件互连总线(例如,PCI、PCI快速))连接。触摸传感器125可以包括触摸屏传感器、压力传感器或指纹传感器。无线收发器126可以是Wi-Fi收发器、蓝牙收发器或移动网络收发器,该移动网络收发器诸如3G、4G、5G或长期演进(Long-Term Evolution, LTE)收发器。固件接口128启用与系统固件的通信,并且可以例如是统一可扩展固件接口(unified extensible firmware interface, UEFI)。网络控制器134可启用至有线网络的网络连接。在一些示例中,高性能网络控制器(未示出)与接口总线110耦合。在一个示例中,音频控制器146是多声道高清音频控制器。在一个示例中,系统100包括用于将传统(例如,个人系统2(Personal System 2, PS/2))设备耦合至系统的任选的传统I/O控制器140。平台

控制器中枢130还可以连接到一个或多个通用串行总线(Universal Serial Bus,USB)控制器142连接输入设备,诸如,键盘和鼠标143组合、相机144或其他USB输入设备。

[0046] 将会理解,所示的系统100是示例性的而非限制性的,因为也可以使用以不同方式配置的其他类型的数据处理系统。例如,存储器控制器116和平台控制器中枢130的实例可以被集成到分立的外部图形处理器中,该分立的外部图形处理器诸如外部图形处理器118。在一个示例中,平台控制器中枢130和/或存储器控制器116可以在一个或多个处理器102外部。例如,系统100可包括外部存储器控制器116和平台控制器中枢130,该外部存储器控制器116和平台控制器中枢130可以被配置为在与(一个或多个)处理器102通信的系统芯片组内的存储器控制器中枢和外围控制器中枢。

[0047] 例如,可使用电路板(“撬板(sled)”),在该电路板上被放置的部件(诸如,CPU、存储器和其他部件)经设计以实现提升的热性能。在一些示例中,诸如处理器之类的处理部件位于撬板的顶侧上,而诸如DIMM(dual inline memory module,双列直插式存储器模块)之类的附近存储器位于撬板的底侧上。作为由该设计提供的增强的气流的结果,部件能以比在典型系统中更高的频率和功率等级来操作,由此提高性能。此外,撬板被配置成用于盲配机架中的功率和数据通信线缆,由此增强它们被快速地移除、升级、重新安装和/或替换的能力。类似地,位于撬板上的各个部件(诸如,处理器、加速器、存储器和数据存储驱动器)由于它们距彼此的增加的间距而被配置成易于升级。在说明性示例中,部件附加地包括用于证明它们的真实性的硬件认证特征。

[0048] 数据中心可利用支持多个其他网络体系结构的单个网络体系结构(“结构”),多个其他网络体系结构包括以太网和全方位路径。撬板可经由光纤耦合至交换机,这提供比典型的双绞线布线(例如,5类、5e类、6类等)更高的带宽和更低的等待时间。由于高带宽、低等待时间的互连和网络体系结构,数据中心在使用中可集中在物理上分散的诸如存储器、加速器(例如,GPU、图形加速器、FPGA(field programmable gate array,现场可编程门阵列)、ASIC、神经网络和/或人工智能加速器等)和数据存储驱动器之类的资源,并且根据需要将它们提供给计算资源(例如,处理器),从而使计算资源能够就好像被集中的资源在本地那样访问这些被集中的资源。

[0049] 功率供应或功率源可将电压和/或电流提供给系统100或本文中描述的任何部件或系统。在一个示例中,功率供应包括用于插入到墙壁插座中的AC到DC(交流到直流)适配器。此类AC功率可以是可再生能源(例如,太阳能)功率源。在一个示例中,功率源包括DC功率源,诸如,外部AC到DC转换器。在一个示例中,功率源或功率供应包括用于通过接近充电场来充电的无线充电硬件。在一个示例中,功率源可包括内部电池、交流供应、基于动作的功率供应、太阳能功率供应或燃料电池源。

[0050] 图2A-图2D图示由本文中描述的示例提供的计算系统和图形处理器。图2A-图2D的具有与本文中的任何其他附图的元件相同的附图标记(或名称)的元件能以与本文中其他地方描述的方式类似的任何方式进行操作或运行,但不限于此。

[0051] 图2A是处理器200的示例的框图,该处理器200具有一个或多个处理器核心202A-202N、集成存储器控制器214以及集成图形处理器208。处理器200可包括附加的核心,这些附加的核心最多为由虚线框表示的附加核心202N并包括由虚线框表示的附加核心202N。处理器核心202A-202N中的每一个包括一个或多个内部缓存单元204A-204N。在一些示例中,

每个处理器核心也具有对一个或多个共享缓存单元206的访问权。内部缓存单元204A-204N和共享缓存单元206表示处理器200内的缓存存储器层次体系。缓存存储器层次体系可包括每个处理器核心内的至少一个级别的指令和数据缓存以及一个或多个级别的共享的中级缓存,诸如,第二级(L2)、第三级(L3)、第四级(L4)或其他级别的缓存,其中,在外部存储器之前的最高级别的缓存被分类为LLC。在一些示例中,缓存一致性逻辑维持各缓存单元206与204A-204N之间的一致性。

[0052] 在一些示例中,处理器200还可包括一个或多个总线控制器单元的集合216和系统代理核心210。一个或多个总线控制器单元216管理外围总线的集合,诸如,一个或多个PCI总线或PCI快速总线。系统代理核心210提供对各处理器部件的管理功能。在一些示例中,系统代理核心210包括用于管理对各种外部存储器设备(未示出)的访问的一个或多个集成存储器控制器214。

[0053] 在一些示例中,处理器核心202A-202N中的一个或多个处理器核心包括针对同步多线程操作的支持。在此类示例中,系统代理核心210包括用于在多线程处理期间协调并操作核心202A-202N的部件。系统代理核心210可附加地包括功率控制单元(power control unit, PCU),该功率控制单元(PCU)包括用于调节处理器核心202A-202N和图形处理器208的功率状态的逻辑和部件。

[0054] 在一些示例中,处理器200附加地包括用于执行图形处理操作的图形处理器208。在一些示例中,图形处理器208与共享缓存单元的集合206以及系统代理核心210耦合,该系统代理核心210包括一个或多个集成存储器控制器214。在一些示例中,系统代理核心210还包括用于将图形处理器输出驱动到一个或多个经耦合的显示器的显示控制器211。在一些示例中,显示控制器211还可以是经由至少一个互连与图形处理器耦合的单独模块,或者可以集成在图形处理器208内。

[0055] 在一些示例中,基于环的互连单元212用于耦合处理器200的内部部件。然而,可以使用替代的互连单元,诸如,点到点互连、交换式互连、或其他技术,包括本领域中公知的技术。在一些示例中,图形处理器208经由I/O链路213与环互连212耦合。

[0056] 示例性I/O链路213表示多个各种各样的I/O互连中的至少一种,包括促进各处理器部件与高性能嵌入式存储器模块218(诸如,eDRAM模块)之间的通信的封装上I/O互连。在一些示例中,处理器核心202A-202N中的每一个以及图形处理器208可将嵌入式存储器模块218用作共享的最后一级缓存。

[0057] 在一些示例中,处理器核心202A-202N是执行相同的指令集体系结构的同构核心。在另一示例中,处理器核心202A-202N在指令集体系结构(instruction set architecture, ISA)方面是异构的,其中,处理器核心202A-202N中的一个或多个执行第一指令集,而其他核心中的至少一个执行第一指令集的子集或不同的指令集。在一个示例中,处理器核心202A-202N在微体系结构方面是异构的,其中,具有相对较高功耗的一个或多个核心与具有较低功耗的一个或多个功率核心耦合。在一个示例中,处理器核心202A-202N在计算能力方面是异构的。此外,处理器200可在一个或多个芯片上实现,或者被实现为除其他部件之外还具有有所图示的部件的SoC集成电路。

[0058] 图2B是根据本文中所描述的一些示例的图形处理器核心219的硬件逻辑的框图。图2B的具有与本文中任何其他附图的元件相同的附图标记(或名称)的元件能以与在本文

中其他地方描述的方式类似的任何方式进行操作或运行,但不限于此。图形处理器核心219(有时称为核心切片)可以是模块化图形处理器内的一个或多个图形核心。图形处理器核心219的示例是一个图形核心切片,并且基于目标功率包络和性能包络,如本文中所描述的图形处理器可以包括多个图形核心切片。每个图形处理器核心219可包括固定功能块230,该固定功能块230与多个子核心221A-221F(也称为子切片)耦合,多个子核心221A-221F包括模块化的通用和固定功能逻辑的块。

[0059] 在一些示例中,固定功能块230包括几何/固定功能管线231,该几何/固定功能管线231例如在较低性能和/或较低功率的图形处理器实现方式中可由图形处理器核心219中的所有子核心共享。在各示例中,几何/固定功能管线231包括3D固定功能管线(例如,如在下文描述的图3和图4中的3D管线312)、视频前端单元、线程生成器和线程调遣器、以及统一返回缓冲器管理器,该统一返回缓冲器管理器管理统一返回缓冲器(例如,如下文所描述的在图4中的统一返回缓冲器418)。

[0060] 在一个示例中,固定功能块230还包括图形SoC接口232、图形微控制器233和媒体管线234。图形SoC接口232提供图形处理器核心219与片上系统集成电路内的其他处理器核心之间的接口。图形微控制器233是可被配置成管理图形处理器核心219的各种功能的可编程子处理器,这些功能包括线程调遣、调度和抢占。媒体管线234(例如,图3和图4的媒体管线316)包括用于促进对包括图像数据和视频数据的多媒体数据进行解码、编码、预处理和/或后处理的逻辑。媒体管线234经由对子核心221A-221F内的计算或采样逻辑的请求来实现媒体操作。

[0061] 在一个示例中,SoC接口232使图形处理器核心219能够与通用应用处理器核心(例如,CPU)和/或SoC内的其他部件进行通信,其他部件包括诸如共享的最后一级缓存存储器之类的存储器层次体系元件、系统RAM、和/或嵌入式片上或封装上DRAM。SoC接口232还可启用与SoC内的诸如相机成像管线之类的固定功能设备的通信,并且启用全局存储器原子性的使用和/或实现全局存储器原子性,该全局存储器原子性可在图形处理器核心219与SoC内的CPU之间被共享。SoC接口232还可实现针对图形处理器核心219的功率管理控制,并且启用图形核心219的时钟域与SoC内的其他时钟域之间的接口。在一个示例中,SoC接口232使得能够从命令流转化器和全局线程调遣器接收命令缓冲器,该命令流转化器和全局线程调遣器被配置成用于将命令和指令提供给图形处理器内的一个或多个图形核心中的每一个图形核心。当媒体操作要被执行时,这些命令和指令可以被调遣给媒体管线234,或者当图形处理操作要被执行时,这些命令和指令可以被调遣给几何和固定功能管线(例如,几何和固定功能管线231、几何和固定功能管线237)。

[0062] 图形微控制器233可被配置成用于执行针对图形处理器核心219的各种调度任务和管理任务。在一个示例中,图形微控制器233可对子核心221A-221F内的执行单元(execution unit, EU)阵列222A-222F、224A-224F内的各个图形并行引擎执行图形和/或计算工作负载调度。在该调度模型中,在包括图形处理器核心219的SoC的CPU核心上执行的主机软件可将工作负载提交到多个图形处理器门铃(doorbell)中的一个图形处理器门铃,这调用了对适当的图形引擎的调度操作。调度操作包括:确定接下来要运行哪个工作负载,将工作负载提交到命令流转化器,抢占在引擎上运行的现有工作负载,监测工作负载的进度,以及当工作负载完成时通知主机软件。在一个示例中,图形微控制器233还可促进图形处理

器核心219的低功率或空闲状态,从而向图形处理器核心219提供独立于操作系统和/或系统上的图形驱动器软件跨低功率状态转变来保存和恢复图形处理器核心219内的寄存器的能力。

[0063] 图形处理器核心219可具有多于或少于所图示的子核心221A-221F,最多N个模块化子核心。对于N个子核心的每个集合,图形处理器核心219还可包括共享功能逻辑235、共享和/或缓存存储器236、几何/固定功能管线237、以及用于加速各种图形和计算处理操作的附加的固定功能逻辑238。共享功能逻辑235可包括与图4的共享功能逻辑420(例如,采样器逻辑、数学逻辑、和/或线程间通信逻辑)相关联的、可由图形处理器核心219内的每N个子核心共享的逻辑单元。共享和/或缓存存储器236可以是用于图形处理器核心219内的N个子核心221A-221F的集合的最后一级缓存,并且还可以充当可由多个子核心访问的共享存储器。几何/固定功能管线237而不是几何/固定功能管线231可被包括在固定功能块230内,并且几何/固定功能管线237可包括相同或类似的逻辑单元。

[0064] 在一个示例中,图形处理器核心219包括附加的固定功能逻辑238,该附加的固定功能逻辑238可包括供由图形处理器核心219使用的各种固定功能加速逻辑。在一个示例中,附加的固定功能逻辑238包括供在仅位置着色中使用的附加的几何管线。在仅位置着色中,存在两个几何管线:几何/固定功能管线237、231内的完全几何管线;以及剔除管线,其是可被包括在附加的固定功能逻辑238内的附加的几何管线。在一个示例中,剔除管线是完全几何管线的精简版本。完全管线和剔除管线可以执行同一应用的不同实例,每个实例具有单独的上下文。仅位置着色可以隐藏被丢弃三角形的长剔除运行,从而在一些实例中使得能够更早地完成着色。在一个示例中,附加的固定功能逻辑238内的剔除管线逻辑可以与主应用并行地执行位置着色器,并且通常比完全管线更快地生成关键结果,因为剔除管线仅取得顶点的位置属性并对顶点的位置属性进行着色,而不向帧缓冲器执行对像素的栅格化和渲染。剔除管线可以使用所生成的关键结果来计算所有三角形的可见性信息,而无需考虑那些三角形是否被剔除。完全管线(其在本实例中可以被称为重放(replay)管线)可以消耗该可见性信息以跳过被剔除的三角形,从而仅对最终被传递到栅格化阶段的可见的三角形进行着色。

[0065] 在一个示例中,附加的固定功能逻辑238还可包括机器学习加速逻辑,诸如,固定功能矩阵乘法逻辑,该机器学习加速逻辑用于包括针对机器学习训练或推断的优化的实现方式。

[0066] 在每个图形子核心221A-221F内包括可用于响应于由图形管线、媒体管线、或着色器程序作出的请求而执行图形操作、媒体操作和计算操作的执行资源的集合。图形子核心221A-221F包括:多个EU阵列222A-222F、224A-224F;线程调遣和线程间通信(thread dispatch and inter-thread communication,TD/IC)逻辑223A-223F;3D(例如,纹理)采样器225A-225F;媒体采样器226A-226F;着色器处理器227A-227F;以及共享的本地存储器(shared local memory,SLM)228A-228F。EU阵列222A-222F、224A-224F各自包括多个执行单元,这些执行单元是能够执行浮点和整数/定点逻辑操作以服务于图形操作、媒体操作或计算操作(包括图形程序、媒体程序或计算着色器程序)的通用图形处理单元。TD/IC逻辑223A-223F执行针对子核心内的执行单元的本地线程调遣和线程控制操作,并且促进在子核心的执行单元上执行的线程之间的通信。3D采样器225A-225F可将纹理或其他3D图形相

关的数据读取到存储器中。3D采样器可基于所配置的样本状态以及与给定纹理相关联的纹理格式以不同方式读取纹理数据。媒体采样器226A-226F可基于与媒体数据相关联的类型和格式来执行类似的读取操作。在一个示例中,每个图形子核心221A-221F可以交替地包括统一3D和媒体采样器。在子核心221A-221F中的每一个子核心内的执行单元上执行的线程可利用每个子核心内的共享的本地存储器228A-228F,以使在线程组内执行的线程能够使用片上存储器的公共池来执行。

[0067] 图2C图示图形处理单元(graphics processing unit,GPU) 239,该GPU 239包括被布置为多核心组240A-240N的专用的图形处理资源集合。虽然提供仅单个多核心组240A的细节,但是将理解,其他多核心组240B-240N可配备有相同或类似的图形处理资源集合。

[0068] 如所图示,多核心组240A可包括图形核心的集合243、张量核心的集合244以及光线追踪核心的集合245。调度器/调遣器241调度和调遣图形线程以用于在各个核心243、244、245上执行。寄存器堆的集合242存储在执行图形线程时由核心243、244、245使用的操作对象值。这些寄存器堆可包括例如用于存储整数值的整数寄存器、用于存储浮点值的浮点寄存器、用于存储紧缩(packed)数据元素(整数和/或浮点数据元素)的向量寄存器以及用于存储张量/矩阵值的片寄存器。在一个示例中,片寄存器被实现为向量寄存器的经组合的集合。

[0069] 一个或多个经组合的第一级(L1)缓存和共享存储器单元247在本地将图形数据存储在每个多核心组240A内,图形数据诸如纹理数据、顶点数据、像素数据、光线数据、包围体数据等。一个或多个纹理单元247也可用于执行纹理操作,诸如,纹理映射和采样。由所有多核心组240A-240N或多核心组240A-240N的子集共享的第二级(L2)缓存253存储用于多个并发的图形线程的图形数据和/或指令。如所图示,可跨多个多核心组240A-240N共享L2缓存253。一个或多个存储器控制器248将GPU 239耦合至存储器249,该存储器249可以是系统存储器(例如,DRAM)和/或专用图形存储器(例如,GDDR6存储器)。

[0070] 输入/输出(Input/output,I/O)电路250将GPU 239耦合至一个或多个I/O设备252,这一个或多个I/O设备252诸如数字信号处理器(digital signal processor,DSP)、网络控制器或用户输入设备。片上互连可用于将I/O设备252耦合至GPU 239和存储器249。I/O电路250的一个或多个I/O存储器管理单元(I/O memory management unit,IOMMU)251直接将I/O设备252耦合至系统存储器249。在一个示例中,IOMMU 251管理用于将虚拟地址映射到系统存储器249中的物理地址的多个页表集合。在该示例中,I/O设备252、(一个或多个)CPU 246和(一个或多个)GPU 239可共享相同的虚拟地址空间。

[0071] 在一个实现方式中,IOMMU 251支持虚拟化。在这种情况下,IOMMU 251可以管理用于将宾客/图形虚拟地址映射到宾客/图形物理地址的第一页表集合以及用于将宾客/图形物理地址映射到(例如,系统存储器249内的)系统/主机物理地址的第二页表集合。第一页表集合和第二页表集合中的每一个的基址可被存储在控制寄存器中,并且在上下文切换时被换出(例如,使得新上下文被提供有对相关页表集合的访问权)。虽然未在图2C中图示,但是核心243、244、245和/或多核心组240A-240N中的每一个可包括转译后备缓冲器(TLB),这些TLB用于对宾客虚拟至宾客物理转译、宾客物理至主机物理转译以及宾客虚拟至主机物理转译进行缓存。

[0072] 在一个示例中,CPU 246、GPU 239和I/O设备252被集成在单个半导体芯片和/或芯

片封装上。图示的存储器249可集成在同一芯片上,或者可经由片外接口被耦合至存储器控制器248。在一个实现方式中,存储器249包括共享与其他物理系统级存储器相同的虚拟地址空间的GDDR6存储器,但是本发明的基本原理不限于该特定的实现方式。

[0073] 在一个示例中,张量核心244包括专门被设计成用于执行矩阵操作的多个执行单元,这些矩阵操作是用于执行深度学习操作的基本计算操作。例如,可将同步矩阵乘法操作用于神经网络训练和推断。张量核心244可使用各种操作对象精度来执行矩阵处理,各种操作对象精度包括单精度浮点(例如,32比特)、半精度浮点(例如,16比特)、整数字(16比特)、字节(8比特)和半字节(4比特)。在一个示例中,神经网络实现方式提取每个经渲染场景的特征,从而潜在地组合来自多个帧的细节,以构建高质量的最终图像。

[0074] 在深度学习实现方式中,可调度并行的矩阵乘法工作以用于在张量核心244上执行。神经网络的训练尤其需要大量矩阵点积操作。为了处理 $N \times N \times N$ 矩阵乘法的内积公式化,张量核心244可包括至少 N 个点积处理元件。在矩阵乘法开始之前,一个完整的矩阵被加载到片寄存器中,并且对于 N 个循环中的每个循环,第二矩阵的至少一列被加载。对于每个循环,存在被处理的 N 个点积。

[0075] 取决于特定的实现方式,能以不同精度来存储矩阵元素,包括16比特的字、8比特的字节(例如,INT8)以及4比特的半字节(例如,INT4)。可为张量核心244指定不同的精度模式以确保将最高效的精度用于不同的工作负载(例如,诸如推断工作负载,其可容忍至字节和半字节的量化(quantization))。

[0076] 在一个示例中,光线追踪核心245加速用于实时光线追踪实现方式和非实时光线追踪实现方式两者的光线追踪操作。具体而言,光线追踪核心245包括光线遍历/相交电路,该光线遍历/相交电路用于使用包围体层次体系(bounding volume hierarchy, BVH)来执行光线遍历并标识封围在BVH体积内的光线与基元之间的相交。光线追踪核心245还可包括用于执行深度测试和剔除(例如,使用Z缓冲器或类似布置)的电路。在一个实现方式中,光线追踪核心245与本文中描述的图像降噪技术协同地执行遍历和相交操作,该图像降噪技术的至少部分可在张量核心244上执行。例如,在一个示例中,张量核心244实现深度学习神经网络以执行对由光线追踪核心245生成的帧的降噪。然而,(一个或多个)CPU 246、图形核心243和/或光线追踪核心245还可实现全部的降噪和/或深度学习算法或降噪和/或深度学习算法中的部分。

[0077] 此外,如上文所描述,可采用对于降噪的分布式方法,其中,GPU 239在通过网络或高速互连而耦合至其他计算设备的计算设备中。在该示例中,经互连的计算设备共享神经网络学习/训练数据,以改善整个系统学习执行用于不同类型的图像帧和/或不同的图形应用的降噪的速度。

[0078] 在一个示例中,光线追踪核心245处理所有的BVH遍历和光线-基元相交,从而使图形核心243免于被针对每条光线的数千个指令过载。在一个示例中,每个光线追踪核心245包括用于执行包围盒测试(例如,用于遍历操作)的第一专业电路集合和用于执行光线-三角形相交测试(例如,使已被遍历的光线相交)的第二专业电路集合。因此,在一个示例中,多核心组240A可简单地启动光线探测,并且光线追踪核心245独立地执行光线遍历和相交,并将命中数据(例如,命中、无命中、多个命中等)返回到线程上下文。当光线追踪核心245执行遍历和相交操作时,其他核心243、244被释放以执行其他图形或计算工作。

[0079] 在一个示例中,每个光线追踪核心245包括用于执行BVH测试操作的遍历单元和执行光线-基元相交测试的相交单元。相交单元生成“命中”、“无命中”或“多个命中”响应,该相交单元将这些响应提供给适当的线程。在遍历和相交操作期间,其他核心(例如,图形核心243和张量核心244)的执行资源被释放以执行其他形式的图形工作。

[0080] 在下文描述的一个特定示例中,使用在其中工作被分布在图形核心243与光线追踪核心245之间的混合式栅格化/光线追踪方法。在一个示例中,光线追踪核心245(和/或其他核心243、244)包括对光线追踪指令集的硬件支持,光线追踪指令集诸如:微软的DirectX光线追踪(DirectX Ray Tracing,DXR),其包括DispatchRays命令;以及光线生成着色器、最近命中着色器、任何命中着色器和未命中着色器,它们使得能够为每个对象指派唯一的着色器和纹理集合。可由光线追踪核心245、图形核心243和张量核心244支持的另一光线追踪平台是Vulkan 1.1.85。然而,要注意,本发明的基本原理不限于任何特定的光线追踪ISA。

[0081] 一般而言,各个核心245、244、243可支持包括用于以下各项的指令/函数的光线追踪指令集:光线生成、最近命中、任何命中、光线-基元相交、逐基元和层次体系包围盒构建、未命中、拜访和异常。更具体地,一个示例包括用于执行以下功能的光线追踪指令:

[0082] 光线生成——可为每个像素、样本或其他用户定义的工作指派执行光线生成指令。

[0083] 最近命中——可执行最近命中指令以对场景内光线与基元的最近交点定位。

[0084] 任何命中——任何命中指令标识场景内光线与基元之间的多个相交,从而潜在地标识新的最近交点。

[0085] 相交——相交指令执行光线-基元相交测试并输出结果。

[0086] 逐基元包围盒构建——该指令围绕给定的基元或基元组建立包围盒(例如,当建立新BVH或其他加速数据结构时)。

[0087] 未命中——指示光线未命中场景或场景的指定区域内的所有几何体。

[0088] 拜访——指示光线将遍历的子容体。

[0089] 异常——包括各种类型的异常处置器(例如,针对各种错误条件被调用)。

[0090] 图2D是根据本文中描述的示例的通用图形处理单元(GPGPU)270的框图,该GPGPU 270可被配置为图形处理器和/或计算加速器。GPGPU 270可经由一个或多个系统和/或存储器总线与主机处理器(例如,一个或多个CPU 246)和存储器271、272互连。在一个示例中,存储器271是可与一个或多个CPU 246进行共享的系统存储器,而存储器272是专用于GPGPU 270的设备存储器。在一个示例中,GPGPU 270和设备存储器272内的部件可被映射到能够由一个或多个CPU 246访问的存储器地址中。可经由存储器控制器268来促进对存储器271和272的访问。在一个示例中,存储器控制器268包括内部直接存储器存取(direct memory access,DMA)控制器269,或可包括用于执行否则将由DMA控制器执行的操作的逻辑。

[0091] GPGPU 270包括多个缓存存储器,这些缓存存储器包括L2缓存253、L1缓存254、指令缓存255以及共享存储器256,该共享存储器256的至少部分也可被分区为缓存存储器。GPGPU 270还包括多个计算单元260A-260N。每个计算单元260A-260N包括向量寄存器的集合261、标量寄存器的集合262、向量逻辑单元的集合263以及标量逻辑单元的集合264。计算单元260A-260N还可包括本地共享存储器265和程序计数器266。计算单元260A-260N可与常

量缓存267耦合,该常量缓存267可用于存储常量数据,该常量数据是在GPGPU 270上执行的内核程序或着色器程序的运行期间不会改变的数据。在一个示例中,常量缓存267是标量数据缓存,并且经缓存的数据可被直接取到标量寄存器262中。

[0092] 在操作期间,一个或多个CPU 246可将命令写入到GPGPU 270中的寄存器中,或写入到GPGPU 270中的、已经被映射到可访问地址空间中的存储器中。命令处理器257可从寄存器或存储器读取命令,并且确定将如何在GPGPU 270内处理那些命令。随后可使用线程调遣器258来将线程调遣给计算单元260A-260N以执行那些命令。每个计算单元260A-260N可独立于其他计算单元来执行线程。此外,每个计算单元260A-260N可被独立地配置成用于有条件计算,并且可有条件地将计算的结果输出到存储器。当所提交的命令完成时,命令处理器257可中断一个或多个CPU 246。

[0093] 图3A-图3C图示由本文中描述的示例提供的附加的图形处理器和计算加速器体系结构的框图。图3A-图3C的具有与本文中的任何其他附图的元件相同的附图标记(或名称)的元件能以与本文中其他地方描述的方式类似的任何方式进行操作或运行,但不限于此。

[0094] 图3A是图形处理器300的框图,该图形处理器300可以是分立的图形处理单元,或可以是与多个处理核心或其他半导体器件集成的图形处理器,其他半导体器件诸如但不限于存储器设备或网络接口。在一些示例中,图形处理器经由至图形处理器上的寄存器的存储器映射的I/O接口并且利用被放置到处理器存储器中的命令进行通信。在一些示例中,图形处理器300包括用于访问存储器的存储器接口314。存储器接口314可以是至本地存储器、一个或多个内部缓存、一个或多个共享的外部缓存和/或至系统存储器的接口。

[0095] 在一些示例中,图形处理器300还包括用于将显示输出数据驱动到显示设备318的显示控制器302。显示控制器302包括用于显示器的一个或多个叠加平面以及多层的视频或用户界面元素的合成的硬件。显示设备318可以是内部或外部显示设备。在一个示例中,显示设备318是头戴式显示设备,诸如,虚拟现实(VR)显示设备或增强现实(AR)显示设备。在一些示例中,图形处理器300包括用于将媒体编码到一种或多种媒体编码格式,从一种或多种媒体编码格式对媒体解码,或在一种或多种媒体编码格式之间对媒体转码的视频编解码器引擎306,这一种或多种媒体编码格式包括但不限于:移动图片专家组(Moving Picture Experts Group, MPEG)格式(诸如, MPEG-2)、高级视频译码(Advanced Video Coding, AVC)格式(诸如, H.264/MPEG-4AVC、H.265/HEVC、开放媒体联盟(Alliance for Open Media, AOMedia) VP8、VP9)、以及电影和电视工程师协会(the Society of Motion Picture & Television Engineers, SMPTE) 421M/VC-1、和联合图像专家组(Joint Photographic Experts Group, JPEG)格式(诸如, JPEG、以及运动JPEG(Motion JPEG, MJPEG)格式)。

[0096] 在一些示例中,图形处理器300包括块图像传输(block image transfer, BLIT)引擎304,用于执行二维(2D)栅格化器操作,包括例如,比特边界块传输。然而,在一个示例中,使用图形处理引擎(graphics processing engine, GPE) 310的一个或多个部件执行2D图形操作。在一些示例中, GPE 310是用于执行图形操作的计算引擎,这些图形操作包括三维(3D)图形操作和媒体操作。

[0097] 在一些示例中, GPE 310包括用于执行3D操作的3D管线312,该3D操作诸如,使用作用于3D基元形状(例如,矩形、三角形等)的处理函数来渲染三维图像和场景。3D管线312包括可编程和固定功能元件,这些可编程和固定功能元件执行元件内的各种任务和/或生成

到3D/媒体子系统315的执行线程。虽然3D管线312可用于执行媒体操作,但是GPE 310的示例还包括媒体管线316,该媒体管线316专门用于执行媒体操作,诸如,视频后处理和图像增强。

[0098] 在一些示例中,媒体管线316包括固定功能或可编程逻辑单元,用于代替、或代表视频编解码器引擎306来执行一个或多个专业的媒体操作,诸如,视频解码加速、视频去隔行以及视频编码加速。在一些示例中,媒体管线316附加地包括线程生成单元,用于生成线程以供在3D/媒体子系统315上执行。所生成的线程在3D/媒体子系统315中所包括的一个或多个图形执行单元上执行用于媒体操作的计算。

[0099] 在一些示例中,3D/媒体子系统315包括用于执行由3D管线312和媒体管线316生成的线程的逻辑。在一个示例中,管线将线程执行请求发送到3D/媒体子系统315,该3D/媒体子系统315包括用于对于对可用的线程执行资源的各种请求进行仲裁和调遣的线程调遣逻辑。执行资源包括用于处理3D线程和媒体线程的图形执行单元的阵列。在一些示例中,3D/媒体子系统315包括用于线程指令和数据的一个或多个内部缓存。在一些示例中,子系统还包括用于在线程之间共享数据并用于存储输出数据的共享存储器,其包括寄存器和可寻址存储器。

[0100] 图3B图示根据本文中描述的示例的图形处理器320,该图形处理器320具有分片体系结构。在一个示例中,图形处理器320包括图形处理引擎集群322,该图形处理引擎集群322在图形引擎片310A-310D内具有图3A的图形处理引擎310的多个实例。每个图形引擎片310A-310D可经由片互连的集合323A-323F被互连。每个图形引擎片310A-310D还可经由存储器互连325A-325D被连接到存储器模块或存储器设备326A-326D。存储器设备326A-326D可使用任何图形存储器技术。例如,存储器设备326A-326D可以是图形双倍数据速率(GDDR)存储器。在一个示例中,存储器设备326A-326D是高带宽存储器(high-bandwidth memory, HBM)模块,这些HBM模块可与其相应的图形引擎片310A-310D一起在管芯上。在一个示例中,存储器设备326A-326D是可被堆叠在其相应的图形引擎片310A-310D的顶部上的堆叠式存储器设备。在一个示例中,每个图形引擎片310A-310D和相关联的存储器326A-326D驻留在分开的小芯片上,这些分开的小芯片被接合到基础管芯或基础衬底,如在图11B-图11D中进一步详细地所描述。

[0101] 图形处理引擎集群322可与片上或封装上结构(fabric)互连324连接。结构互连324可启用图形引擎片310A-310D与诸如视频编解码器306和一个或多个复制引擎304之类的部件之间的通信。复制引擎304可用于将数据移出存储器设备326A-326D和在图形处理器320外部的存储器(例如,系统存储器),将数据移入存储器设备326A-326D和在图形处理器320外部的存储器(例如,系统存储器),并且在存储器设备326A-326D与在图形处理器320外部的存储器(例如,系统存储器)之间移动数据。结构互连324还可用于将图形引擎片310A-310D互连。图形处理器320可以可选地包括显示控制器302,用于启用与外部显示设备318的连接。图形处理器还可被配置为图形加速器或计算加速器。在加速器配置中,显示控制器302和显示设备318可被省略。

[0102] 图形处理器320可经由主机接口328连接到主机系统。主机接口328可启用图形处理器320、系统存储器和/或其他系统部件之间的通信。主机接口328可以是例如PCI快速总线或另一类型的主机系统接口。

[0103] 图3C图示根据本文中描述的示例的计算加速器330。计算加速器330可包括与图3B的图形处理器320的体系结构类似性,并且针对计算加速进行优化。计算引擎集群332可包括计算引擎片340A-340D的集合,计算引擎片340A-340D的集合包括针对并行或基于向量的通用计算操作优化的执行逻辑。在一些示例中,计算引擎片340A-340D不包括固定功能图形处理逻辑,但是在一个示例中,计算引擎片340A-340D中的一个或多个可包括用于执行媒体加速的逻辑。计算引擎片340A-340D可经由存储器互连325A-325D连接到存储器326A-326D。存储器326A-326D和存储器互连325A-325D可以是与在图形处理器320中类似的技术,或者可以是不同的技术。图形计算引擎片340A-340D还可经由片互连323A-323F的集合被互连,并且可与结构互连324连接和/或由结构互连324互连。在一个示例中,计算加速器330包括可被配置为设备范围的缓存的大型L3缓存336。计算加速器330还能以与图3B的图形处理器320类似的方式经由主机接口328连接到主机处理器和存储器。

[0104] 图形处理引擎

[0105] 图4是根据一些示例的图形处理器的图形处理引擎410的框图。在一个示例中,图形处理引擎(GPE)410是图3A中示出的GPE 310的某个版本,并且还可表示图3B的图形引擎片310A-310D。图4的具有与本文中的任何其他附图的元件相同的附图标记(或名称)的元件能以与本文中其他地方描述的方式类似的任何方式进行操作或运行,但不限于此。例如,图3A的3D管线312和媒体管线316被图示。媒体管线316在GPE 410的一些示例中是任选的,并且可以不显式地被包括在GPE 410内。例如并且在至少一个示例中,单独的媒体和/或图像处理处理器被耦合至GPE 410。

[0106] 在一些示例中,GPE 410与命令流转化器403耦合或包括命令流转化器403,该命令流转化器403将命令流提供给3D管线312和/或媒体管线316。在一些示例中,命令流转化器403与存储器耦合,该存储器可以是系统存储器、或内部缓存存储器和共享缓存存储器中的一个或多个。在一些示例中,命令流转化器403从存储器接收命令,并且将这些命令发送至3D管线312和/或媒体管线316。这些命令是从环形缓冲器取得的指示,该环形缓冲器存储用于3D管线312和媒体管线316的命令。在一个示例中,环形缓冲器可附加地包括存储批量的多个命令的批量命令缓冲器。用于3D管线312的命令还可包括对存储在存储器中的数据引用,这些数据诸如但不限于用于3D管线312的顶点数据和几何数据和/或用于媒体管线316的图像数据和存储器对象。3D管线312和媒体管线316通过经由相应的管线内的逻辑执行操作或者通过将一个或多个执行线程调遣至图形核心阵列414来处理命令和数据。在一个示例中,图形核心阵列414包括一个或多个图形核心块(例如,(一个或多个)图形核心415A、(一个或多个)图形核心415B),每个块包括一个或多个图形核心。每个图形核心包括图形执行资源的集合,该图形执行资源的集合包括:用于执行图形操作和计算操作的通用和图形专用执行逻辑;以及固定功能纹理处理逻辑和/或机器学习和人工智能加速逻辑。

[0107] 在各示例中,3D管线312可包括用于通过处理指令并将执行线程调遣给图形核心阵列414来处理一个或多个着色器程序的固定功能和可编程逻辑,该一个或多个着色器程序诸如,顶点着色器、几何着色器、像素着色器、片段着色器、计算着色器或其他着色器程序。图形核心阵列414提供统一的执行资源块,以供在处理这些着色器程序时使用。图形核心阵列414的(一个或多个)图形核心415A-415B内的多功能执行逻辑(例如,执行单元)包括对各种3D API着色器语言的支持,并且可执行与多个着色器相关联的多个同步执行线程。

[0108] 在一些示例中,图形核心阵列414包括用于执行诸如视频和/或图像处理之类的媒体功能的执行逻辑。在一个示例中,除了图形处理操作之外,执行单元还包括可编程以执行并行的通用计算操作的通用逻辑。通用逻辑可并行地或结合图1的(一个或多个)处理器核心107或如图2A中的核心202A-202N内的通用逻辑来执行处理操作。

[0109] 由在图形核心阵列414上执行的线程生成的输出数据可以将数据输出到统一返回缓冲器(unified return buffer,URB)418中的存储器。URB 418可存储用于多个线程的数据。在一些示例中,URB 418可用于在图形核心阵列414上执行的不同线程之间发送数据。在一些示例中,URB 418可附加地用于在图形核心阵列上的线程与共享功能逻辑420内的固定功能逻辑之间的同步。

[0110] 在一些示例中,图形核心阵列414是可缩放的,使得阵列包括可变数量的图形核心,每个图形核心都具有基于GPE 410的目标功率和性能等级的可变数量的执行单元。在一个示例中,执行资源是动态地可缩放的,使得执行资源可根据需要被启用或禁用。

[0111] 图形核心阵列414与共享功能逻辑420耦合,该共享功能逻辑420包括在图形核心阵列中的图形核心之间被共享的多个资源。共享功能逻辑420内的共享功能是将专业的补充功能提供给图形核心阵列414的硬件逻辑单元。在各示例中,共享功能逻辑420包括但不限于采样器421逻辑、数学422逻辑和线程间通信(inter-thread communication,ITC)423逻辑。此外,一些示例实现共享功能逻辑420内的一个或多个缓存425。

[0112] 至少在其中对于给定的专业功能的需求不足以包括在图形核心阵列414内的情况下实现共享功能。相反,那个专业功能的单个实例化被实现为共享功能逻辑420中的独立实体,并且在图形核心阵列414内的执行资源之间被共享。在图形核心阵列414之间被共享并被包括在图形核心阵列414内的确切的功能集因示例而异。在一些示例中,共享功能逻辑420内的由图形核心阵列414广泛使用的特定共享功能可被包括在图形核心阵列414内的共享功能逻辑416内。在各个示例中,图形核心阵列414内的共享功能逻辑416可包括共享功能逻辑420内的一些或所有逻辑。在一个示例中,共享功能逻辑420内的所有逻辑元件可以在图形核心阵列414的共享功能逻辑416内被复制。在一个示例中,共享功能逻辑420被排除以有利于图形核心阵列414内的共享功能逻辑416。

[0113] 执行单元

[0114] 图5A-图5B图示根据本文中所描述的示例的线程执行逻辑500,该线程执行逻辑500包括在图形处理器核心中采用的处理元件的阵列。图5A-图5B的具有与本文中任何其他附图的元件相同的附图标记(或名称)的元件能以与在本文中其他地方描述的方式类似的任何方式进行操作或运行,但不限于此。图5A-图5B图示线程执行逻辑500的概述,该线程执行逻辑500可表示以图2B中的每个子核心221A-221F图示的硬件逻辑。图5A表示通用图形处理器内的执行单元,而图5B表示可在计算加速器内被使用的执行单元。

[0115] 如在图5A中所图示,在一些示例中,线程执行逻辑500包括着色器处理器502、线程调遣器504、指令缓存506、包括多个执行单元508A-508N的可缩放执行单元阵列、采样器510、共享本地存储器511、数据缓存512、以及数据端口514。在一个示例中,可缩放执行单元阵列可通过基于工作负载的计算要求启用或禁用一个或多个执行单元(例如,执行单元508A、508B、508C、508D,一直到508N-1和508N中的任一个)来动态地缩放。在一个示例中,所包括的部件经由互连结构而互连,该互连结构链接到部件中的每个部件。在一些示例中,线

程执行逻辑500包括通过指令缓存506、数据端口514、采样器510、以及执行单元508A-508N中的一个或多个到存储器(诸如,系统存储器或缓存存储器)的一个或多个连接。在一些示例中,每个执行单元(例如,508A)是能够执行多个同步硬件线程同时针对每个线程并行地处理多个数据元素的独立式可编程通用计算单元。在各示例中,执行单元508A-508N的阵列是可缩放的以包括任何数量的单独执行单元。

[0116] 在一些示例中,执行单元508A-508N主要用于执行着色器程序。着色器处理器502可处理各种着色器程序,并且可经由线程调遣器504来调遣与着色器程序相关联的执行线程。在一个示例中,线程调遣器包括用于对来自图形管线和媒体管线的线程发起请求进行仲裁并在执行单元508A-508N中的一个或多个执行单元上实例化所请求的线程的逻辑。例如,几何管线可将顶点着色器、曲面细分着色器或几何着色器调遣给线程执行逻辑以用于处理。在一些示例中,线程调遣器504还可处理来自执行的着色器程序的运行时线程生成请求。

[0117] 在一些示例中,执行单元508A-508N支持包括对许多标准3D图形着色器指令的原生支持的指令集,使得以最小的转换执行来自图形库(例如,Direct 3D和OpenGL)的着色器程序。这些执行单元支持顶点和几何处理(例如,顶点程序、几何程序、顶点着色器)、像素处理(例如,像素着色器、片段着色器)以及通用处理(例如,计算和媒体着色器)。执行单元508A-508N中的每个执行单元都能够进行多发出单指令多数据(single instruction multypedata,SIMD)执行,并且多线程操作在面对较高等待时间的存储器访问时启用高效的执行环境。每个执行单元内的每个硬件线程都具有专用的高带宽寄存器堆和相关联的独立线程状态。对于能够进行整数操作、单精度浮点操作和双精度浮点操作、能够具有SIMD分支能力、能够进行逻辑操作、能够进行超越操作和能够进行其他混杂操作的管线,执行是针对每个时钟多发出(multi-issue per clock)的。在等待来自存储器或共享功能中的一个共享功能的数据时,执行单元508A-508N内的依赖性逻辑使等待的线程休眠,直到所请求的数据已被返回。当等待的线程正在休眠时,硬件资源可致力于处理其他线程。例如,在与顶点着色器操作相关联的延迟期间,执行单元可以执行针对像素着色器、片段着色器或包括不同顶点着色器的另一类型的着色器程序的操作。各示例可应用以使用利用单指令多线程(Single Instruction Multiple Thread,SIMT)的执行,作为对SIMD的使用的替代,或作为对SIMD的使用的附加。对SIMD核心或操作的引用也可应用于SIMT,或应用于SIMD与SIMT的组合。

[0118] 执行单元508A-508N中的每个执行单元对数据元素的阵列进行操作。数据元素的数量是“执行大小”、或用于指令的通道。执行通道是用于指令内的数据元素访问、掩码、和流控制的执行的逻辑单元。通道的数量可独立于用于特定图形处理器的物理算术逻辑单元(Arithmetic Logic Unit,ALU)或浮点单元(Floating Point Unit,FPU)的数量。在一些示例中,执行单元508A-508N支持整数和浮点数据类型。

[0119] 执行单元指令集包括SIMD指令。各种数据元素可以作为紧缩(packed)数据类型被存储在寄存器中,并且执行单元将基于元素的数据大小来处理各个元素。例如,当对256比特宽的向量进行操作时,向量的256比特被存储在寄存器中,并且执行单元将向量操作作为四个单独的54比特紧缩(packed)数据元素(四字(Quad-Word,QW)大小数据元素)、八个单独的32比特紧缩数据元素(双字(Double Word,DW)大小数据元素)、十六个单独的16比特紧缩数

据元素(字(Word,W)大小数据元素)、或三十二个单独的8比特数据元素(字节(byte,B)大小数据元素)。然而,不同的向量宽度和寄存器大小是可能的。

[0120] 在一个示例中,可以将一个或多个执行单元组合到融合执行单元509A-509N中,该融合执行单元509A-509N具有对于融合EU而言共同的线程控制逻辑(507A-507N)。可以将多个EU融合到EU组中。融合的EU组中的每个EU可以被配置成用于执行单独的SIMD硬件线程。融合的EU组中的EU的数量可以根据示例而有所不同。此外,可以逐EU地执行各种SIMD宽度,包括但不限于SIMD8、SIMD16和SIMD32。每个融合图形执行单元509A-509N包括至少两个执行单元。例如,融合执行单元509A包括第一EU 508A、第二EU 508B、以及对于第一EU 508A和第二EU 508B而言共同的线程控制逻辑507A。线程控制逻辑507A控制在融合图形执行单元509A上执行的线程,从而允许融合执行单元509A-509N内的每个EU使用共同的指令指针寄存器来执行。

[0121] 一个或多个内部指令缓存(例如,506)被包括在线程执行逻辑500中,以对用于执行单元的线程指令进行缓存。在一些示例中,一个或多个数据缓存(例如,512)被包括,以在线程执行期间对线程数据进行缓存。在执行逻辑500上执行的线程还可将被显式地管理的数据存储在共享本地存储器511中。在一些示例中,采样器510被包括以为3D操作提供纹理采样并且为媒体操作提供媒体采样。在一些示例中,采样器510包括专业的纹理或媒体采样功能,以便在向执行单元提供采样数据之前在采样过程期间处理纹理数据或媒体数据。

[0122] 在执行期间,图形管线和媒体管线经由线程生成和调遣逻辑将线程发起请求发送到线程执行逻辑500。一旦几何对象的组已经被处理并被栅格化为像素数据,着色器处理器502内的像素处理器逻辑(例如,像素着色器逻辑、片段着色器逻辑等)就被调用以进一步计算输出信息,并且使得结果被写入到输出表面(例如,颜色缓冲器、深度缓冲器、模版印刷缓冲器等)。在一些示例中,像素着色器或片段着色器计算各顶点属性的值,各顶点属性的值将跨经栅格化的对象而被插值。在一些示例中,着色器处理器502内的像素处理器逻辑随后执行应用编程接口(application programming interface,API)供应的像素着色器程序或片段着色器程序。为了执行着色器程序,着色器处理器502经由线程调遣器504将线程调遣给执行单元(例如,508A)。在一些示例中,着色器处理器502使用采样器510中的纹理采样逻辑来访问存储在存储器中的纹理图中的纹理数据。对纹理数据和输入几何数据的算术操作计算针对每个几何片段的像素颜色数据,或丢弃一个或多个像素而不进行进一步处理。

[0123] 在一些示例中,数据端口514提供存储器访问机制,供线程执行逻辑500将经处理的数据输出至存储器以便在图形处理器输出管线上进一步处理。在一些示例中,数据端口514包括或耦合至一个或多个缓存存储器(例如,数据缓存512),以便对数据进行缓存供经由数据端口进行存储器访问。

[0124] 在一个示例中,执行逻辑500还可包括可提供光线追踪加速功能的光线追踪器505。光线追踪器505可支持光线追踪指令集,该光线追踪指令集包括用于光线生成的指令/函数。光线追踪指令集可与由图2C中的光线追踪核心245支持的光线追踪指令集类似或不同。

[0125] 图5B图示根据示例的执行单元508的示例性内部细节。图形执行单元508可包括指令取得单元537、通用寄存器堆阵列(general register file array,GRF)524、体系结构寄存器堆阵列(architectural register file,ARF)526、线程仲裁器522、发送单元530、分支

单元532、SIMD浮点单元(floating point unit,FPU)的集合534、以及在一个示例中的专用整数SIMD ALU的集合535。GRF 524和ARF 526包括与可在图形执行单元508中活跃的每个同步硬件线程相关联的通用寄存器堆和体系结构寄存器堆的集合。在一个示例中,每线程体系结构状态被维持在ARF 526中,而在线程执行期间使用的数据被存储在GRF 524中。每个线程的执行状态,包括用于每个线程的指令指针,可以被保存在ARF 526中的线程特定的寄存器中。

[0126] 在一个示例中,图形执行单元508具有作为同步多线程(Simultaneous Multi-Threading,SMT)与细粒度交织多线程(Interleaved Multi-Threading,IMT)的组分的体系结构。该体系结构具有模块化配置,该模块化配置可以基于同步线程的目标数量和每个执行单元的寄存器的数量而在设计时进行微调,其中跨用于执行多个同步线程的逻辑来划分执行单元资源。可由图形执行单元508执行的逻辑线程的数量不限于硬件线程的数量,并且可将多个逻辑线程指派给每个硬件线程。

[0127] 在一个示例中,图形执行单元508可协同发出多条指令,这些指令可以各自是不同的指令。图形执行单元线程508的线程仲裁器522可以将指令调遣给以下各项中的一项以供执行:发送单元530、分支单元532或(一个或多个)SIMD FPU 534。每个执行线程可访问GRF 524内的128个通用寄存器,其中,每个寄存器可存储可作为具有32比特数据元素的SIMD 8元素向量访问的32个字节。在一个示例中,每个执行单元线程具有对GRF 524内的4个千字节的访问权,但是示例并不限于此,并且在其他示例中可以提供更多或更少的寄存器资源。在一个示例中,图形执行单元508被分区为可独立地执行计算操作的七个硬件线程,但是每个执行单元的线程数量也可根据示例而有所不同。例如,在一个示例中,支持最多16个硬件线程。在其中七个线程可以访问4个千字节的示例中,GRF 524可以存储总共28个千字节。在16个线程可访问4个千字节的情况下,GRF 524可存储总共64个千字节。灵活的寻址模式可准许对寄存器一起进行寻址,从而有效地建立更宽的寄存器或者表示跨步式矩形块数据结构。

[0128] 在一个示例中,经由通过消息传递发送单元530执行的“发送”指令来调遣存储器操作、采样器操作以及其他较长等待时间的系统通信。在一个示例中,分支指令被调遣给专用分支单元532,以促进SIMD分散和最终的汇聚。

[0129] 在一个示例中,图形执行单元508包括用于执行浮点操作的一个或多个SIMD浮点单元(FPU) 534。在一个示例中,(一个或多个)FPU 534还支持整数计算。在一个示例中,(一个或多个)FPU 534可以SIMD执行最多数量M个32比特浮点(或整数)操作,或者SIMD执行最多2M个16比特整数或16比特浮点操作。在一个示例中,(一个或多个)FPU中的至少一个提供支持高吞吐量超越数学函数和双精度64比特浮点的扩展数学能力。在一些示例中,8比特整数SIMD ALU的集合535也存在,并且可专门优化成执行与机器学习计算相关联的操作。

[0130] 在一个示例中,可以在图形子核心分组(例如,子切片)中对图形执行单元508的多个实例的阵列进行实例化。为了可缩放性,产品架构师可以选择每子核心分组的执行单元的确切数量。在一个示例中,执行单元508可以跨多个执行通道来执行指令。在进一步的示例中,在不同通道上执行在图形执行单元508上执行的每个线程。

[0131] 图6图示根据示例的附加的执行单元600。执行单元600可以是用于在例如图3C中的计算引擎片340A-340D中使用的计算优化的执行单元,但不限于此。执行单元600的变体

也可在如图3B中的图形引擎片310A-310D中使用。在一个示例中,执行单元600包括线程控制单元601、线程状态单元602、指令取得/预取单元603、以及指令解码单元604。执行单元600附加地包括寄存器堆606,该寄存器堆606存储可被指派给执行单元内的硬件线程的寄存器。执行单元600附加地包括发送单元607和分支单元608。在一个示例中,发送单元607和分支单元608能以与图5B的图形执行单元508的发送单元530和分支单元532类似的方式操作。

[0132] 执行单元600还包括计算单元610,该计算单元610包括多个不同类型的功能单元。在一个示例中,计算单元610包括ALU单元611,该ALU单元611包括算术逻辑单元的阵列。ALU单元611可被配置成执行64比特、32比特和16比特的整数和浮点操作。可同时执行整数和浮点操作。计算单元610还可包括脉动阵列612和数学单元613。脉动阵列612包括数据处理单元的宽W且深D的网络,其可用于以脉动方式执行向量或其他数据并行操作。在一个示例中,脉动阵列612可被配置成执行矩阵操作,诸如,矩阵点积操作。在一个示例中,脉动阵列612支持16比特浮点操作以及8比特和4比特整数操作。在一个示例中,脉动阵列612可被配置成加速机器学习操作。在此类示例中,脉动阵列612可被配置有对bfloat16比特浮点格式的支持。在一个示例中,数学单元613可被包括以便以高效的且比ALU单元611更低功率的方式执行数学操作的特定子集。数学单元613可包括可在由其他示例提供的图形处理引擎的共享功能逻辑中发现的数学逻辑(例如,图4中的共享功能逻辑420的数学逻辑422)的变体。在一个示例中,数学单元613可被配置成执行32比特和64比特浮点操作。

[0133] 线程控制单元601包括用于控制执行单元内的线程的执行的逻辑。线程控制单元601可包括线程仲裁逻辑,该线程仲裁逻辑用于启动、停止以及抢占执行单元600内线程的执行。线程状态单元602可用于存储用于被指派以在执行单元600上执行的线程的线程状态。将线程状态存储在执行单元600内使得能够在线程变得被阻止或空闲时快速抢占那些线程。指令取得/预取单元603可从较高级别执行逻辑的指令缓存(例如,如图5A中的指令缓存506)取得指令。指令取得/预取单元603还可基于对当前执行线程的分析来发出对要被加载到指令缓存中的指令的预取请求。指令解码单元604可用于对要由计算单元执行的指令进行解码。在一个示例中,指令解码单元604可被用作次级解码器以将复杂指令解码为组成的微操作。

[0134] 执行单元600附加地包括寄存器堆606,该寄存器堆606可由在执行单元600上执行的硬件线程使用。寄存器堆606中的寄存器可跨用于执行执行单元600的计算单元610内的多个同步线程的逻辑而被划分。可由图形执行单元600执行的逻辑线程的数量不限于硬件线程的数量,并且可将多个逻辑线程指派给每个硬件线程。基于所支持的硬件线程的数量,寄存器堆606的大小可因示例而有所不同。在一个示例中,可使用寄存器重命名来动态地将寄存器分配给硬件线程。

[0135] 图7是根据一些示例的图示图形处理器指令格式700的框图。在一个或多个示例中,图形处理器执行单元支持具有按照多种格式的指令的指令集。实线框图示通常被包括在执行单元指令中的组成部分,而虚线包括任选的或仅被包括在指令的子集中的组成部分。在一些示例中,所描述和图示的指令格式700是宏指令,因为它们供应至执行单元的指令,这与产生自一旦指令被处理就进行的指令解码的微操作相反。

[0136] 在一些示例中,图形处理器执行单元原生地支持128比特指令格式710的指令。基

于所选择的指令、指令选项和操作对象数量,64比特紧凑指令格式730可用于一些指令。原生的128比特指令格式710提供对所有指令选项的访问,而一些选项和操作在64比特格式730中受限。64比特格式730中可用的原生指令因示例而异。在一些示例中,使用索引字段713中的索引值的集合将指令部分地压缩。执行单元硬件基于索引值来引用压缩表的集合,并使用压缩表输出来重构128比特指令格式710的原生指令。可以使用其他大小和格式的指令。

[0137] 针对每种格式,指令操作码712限定执行单元要执行的操作。执行单元跨每个操作对象的多个数据元素并行地执行每个指令。例如,响应于加法指令,执行单元跨表示纹理元素或图片元素的每个颜色通道执行同步加法操作。默认地,执行单元跨操作对象的所有数据通道执行每个指令。在一些示例中,指令控制字段714启用对某些执行选项(诸如,通道选择(例如,谓词(predication))和数据通道顺序(例如,拌和(swizzle)))的控制。针对128比特指令格式710的指令,执行大小字段716限制将被并行地执行的数据通道的数量。在一些示例中,执行大小字段716不可用于64比特紧凑指令格式730。

[0138] 一些执行单元指令具有最多三个操作对象,包括两个源操作对象src0 720、src1 722以及一个目的地718。在一些示例中,执行单元支持双目的地指令,其中目的地中的一个隐式的。数据操纵指令可具有第三源操作对象(例如,src2 724),其中指令操作码712确定源操作对象的数量。指令的最后一个源操作对象可以是与指令一起被传递的立即数(例如,硬编码的)值。

[0139] 在一些示例中,128比特指令格式710包括访问/寻址模式字段726,该访问/寻址模式字段726例如指定使用直接寄存器寻址模式还是间接寄存器寻址模式。当使用直接寄存器寻址模式时,由指令中的比特直接提供一个或多个操作对象的寄存器地址。

[0140] 在一些示例中,128比特指令格式710包括访问/寻址模式字段726,该访问/寻址模式字段726指定指令的寻址模式和/或访问模式。在一个示例中,访问模式用于限定指令的数据访问对齐。一些示例支持包括16字节对齐访问模式和1字节对齐访问模式的访问模式,其中,访问模式的字节对齐确定指令操作对象的访问对齐。例如,当处于第一模式时,指令可将字节对齐的寻址用于源操作对象和目的地操作对象,并且当处于第二模式时,指令可将16字节对齐的寻址用于所有的源操作对象和目的地操作对象。

[0141] 在一个示例中,访问/寻址模式字段726的寻址模式部分确定指令要使用直接寻址还是间接寻址。当使用直接寄存器寻址模式时,指令中的比特直接提供一个或多个操作对象的寄存器地址。当使用间接寄存器寻址模式时,可以基于指令中的地址寄存器值和地址立即数字段来计算一个或多个操作对象的寄存器地址。

[0142] 在一些示例中,基于操作码712比特字段对指令进行分组从而简化操作码解码740。针对8比特的操作码,比特4、比特5、和比特6允许执行单元确定操作码的类型。所示出的确切的操作码分组仅是示例。在一些示例中,移动和逻辑操作码组742包括数据移动和逻辑指令(例如,移动(mov)、比较(cmp))。在一些示例中,移动和逻辑操作码组742共享五个最高有效的比特(most significant bit,MSB),其中,移动(mov)指令采用0000xxxxb的形式,而逻辑指令采用0001xxxxb的形式。流控制指令组744(例如,调用(call)、跳转(jmp))包括0010xxxxb(例如,0x20)形式的指令。混杂指令组746包括指令的混合,包括0011xxxxb(例如,0x30)形式的同步指令(例如,等待(wait)、发送(send))。并行数学指令组748包括

0100xxxxb (例如, 0x40) 形式的逐分量的算术指令 (例如, 加、乘 (mul))。并行数学组 748 跨数据通道并行地执行算术操作。向量数学组 750 包括 0101xxxxb (例如, 0x50) 形式的算术指令 (例如, dp4)。向量数学组对向量操作对象执行算术, 诸如, 点积计算。在一个示例中, 所图示的操作码解码 740 可用于确定执行单元的哪个部分将用于执行经解码的指令。例如, 一些指令可被指定为将由脉动阵列执行的脉动指令。其他指令 (诸如, 光线追踪指令 (未示出)) 可被路由至执行逻辑的切片或分区内的光线追踪核心或光线追踪逻辑。

[0143] 图形管线

[0144] 图 8 是图形处理器 800 的另一示例的框图。图 8 的具有与本文中的任何其他附图的元件相同的附图标记 (或名称) 的元件能以与本文中其他地方描述的方式类似的任何方式进行操作或运行, 但不限于此。

[0145] 在一些示例中, 图形处理器 800 包括几何管线 820、媒体管线 830、显示引擎 840、线程执行逻辑 850 以及渲染输出管线 870。在一些示例中, 图形处理器 800 是包括一个或多个通用处理核心的多核心处理系统内的图形处理器。图形处理器通过至一个或多个控制寄存器 (未示出) 的寄存器写入或者经由通过环形互连 802 发出至图形处理器 800 的命令被控制。在一些示例中, 环形互连 802 将图形处理器 800 耦合至其他处理部件 (诸如, 其他图形处理器或通用处理器)。由命令流转化器 803 解释来自环形互连 802 的命令, 该命令流转化器 803 将指令供应至几何管线 820 或媒体管线 830 的各个部件。

[0146] 在一些示例中, 命令流转化器 803 引导顶点取得器 805 的操作, 该顶点取得器 805 从存储器读取顶点数据, 并执行由命令流转化器 803 提供的顶点处理命令。在一些示例中, 顶点取得器 805 将顶点数据提供给顶点着色器 807, 该顶点着色器 807 对每一个顶点执行坐标空间变换和照明操作。在一些示例中, 顶点取得器 805 和顶点着色器 807 通过经由线程调遣器 831 将执行线程调遣给执行单元 852A-852B 来执行顶点处理指令。

[0147] 在一些示例中, 执行单元 852A-852B 是具有用于执行图形操作和媒体操作的指令集的向量处理器的阵列。在一些示例中, 执行单元 852A-852B 可具有专用于每个阵列或在阵列之间被共享的所附接的 L1 缓存 851。缓存可以被配置为数据缓存、指令缓存、或被分区为在不同分区中包含数据和指令的单个缓存。

[0148] 在一些示例中, 几何管线 820 包括用于执行 3D 对象的硬件加速曲面细分的曲面细分部件。在一些示例中, 可编程外壳着色器 811 配置曲面细分操作。可编程域着色器 817 提供对曲面细分输出的后端评估。曲面细分器 813 在外壳着色器 811 的指示下进行操作, 并且包含用于基于粗糙的几何模型来生成详细的几何对象集合的专用逻辑, 该粗糙的几何模型作为输入被提供给几何管线 820。在一些示例中, 如果不使用曲面细分, 则可以绕过曲面细分部件 (例如, 外壳着色器 811、曲面细分器 813 和域着色器 817)。

[0149] 在一些示例中, 完整的几何对象可由几何着色器 819 经由被调遣给执行单元 852A-852B 的一个或多个线程来处理, 或者直接行进至裁剪器 829。在一些示例中, 几何着色器对整个几何对象操作, 而不是像在图形管线的先前的阶段中那样对顶点或顶点的补片进行操作。如果曲面细分被禁用, 则几何着色器 819 从顶点着色器 807 接收输入。在一些示例中, 几何着色器 819 是可由几何着色器程序编程的, 以便在曲面细分单元被禁用的情况下执行几何曲面细分。

[0150] 在栅格化之前, 裁剪器 829 处理顶点数据。裁剪器 829 可以是固定功能裁剪器或具

有裁剪和几何着色器功能的可编程裁剪器。在一些示例中,渲染输出管线870中的栅格化器和深度测试部件873调遣像素着色器以将几何对象转换为逐像素表示。在一些示例中,像素着色器逻辑被包括在线程执行逻辑850中。在一些示例中,应用可绕过栅格化器和深度测试部件873,并且经由流出单元823访问未栅格化的顶点数据。

[0151] 图形处理器800具有互连总线、互连结构、或允许数据和消息在处理器的主要部件之间传递的某个其他互连机制。在一些示例中,执行单元852A-852B和相关联的逻辑单元(例如,L1缓存851、采样器854、纹理缓存858等)经由数据端口856进行互连,以执行存储器访问并且与处理器的渲染输出管线部件进行通信。在一些示例中,采样器854、缓存851、858和执行单元852A-852B各自具有单独的存储器访问路径。在一个示例中,纹理缓存858也可被配置为采样器缓存。

[0152] 在一些示例中,渲染输出管线870包含栅格化器和深度测试部件873,该栅格化器和深度测试部件873将基于顶点的对象转换为相关联的基于像素的表示。在一些示例中,栅格化器逻辑包括用于执行固定功能三角形和线栅格化的窗口器/掩码器单元。在一些示例中,相关联的渲染缓存878和深度缓存879也是可用的。像素操作部件877对数据执行基于像素的操作,但是在一些实例中,与2D操作相关联的像素操作(例如,利用混合的比特块图像传输)由2D引擎841执行,或者在显示时由显示控制器843使用叠加显示平面来代替。在一些示例中,共享的L3缓存875可用于所有的图形部件,从而允许在不使用主系统存储器的情况下共享数据。

[0153] 在一些示例中,图形处理器媒体管线830包括媒体引擎837和视频前端834。在一些示例中,视频前端834从命令流转化器803接收管线命令。在一些示例中,媒体管线830包括单独的命令流转化器。在一些示例中,视频前端834在将媒体命令发送到媒体引擎837之前处理该媒体命令。在一些示例中,媒体引擎837包括用于生成线程以用于经由线程调遣器831调遣给线程执行逻辑850的线程生成功能。

[0154] 在一些示例中,图形处理器800包括显示引擎840。在一些示例中,该显示引擎840在处理器800外部,并且可经由环形互连802、或某个其他互连总线或结构来与图形处理器耦合。在一些示例中,显示引擎840包括2D引擎841和显示控制器843。在一些示例中,显示引擎840包含能够独立于3D管线进行操作的专用逻辑。在一些示例中,显示控制器843与显示设备(未示出)耦合,该显示设备可以是如在膝上型电脑中的系统集成的显示设备或经由显示设备连接器而附连的外部显示设备。

[0155] 在一些示例中,几何管线820和媒体管线830可配置成用于基于多个图形和媒体编程接口来执行操作,并且不专用于任何一个应用编程接口(API)。在一些示例中,用于图形处理器的驱动器软件将专用于特定图形或媒体库的API调用转换为可由图形处理器处理的命令。在一些示例中,为全部来自Khronos Group的开放图形库(Open Graphics Library, OpenGL)、开放计算语言(Open Computing Language, OpenCL)和/或Vulkan图形和计算API提供支持。在一些示例中,也可以为来自微软公司的Direct3D库提供支持。在一些示例中,可支持这些库的组合。还可以为开源计算机视觉库(Open Source Computer Vision Library, OpenCV)提供支持。如果可进行从未来API的管线到图形处理器的管线的映射,则具有兼容3D管线的未来API也将受到支持。

[0156] 图形管线编程

[0157] 图9A是图示根据一些示例的图形处理器命令格式900的框图。图9B是图示根据示例的图形处理器命令序列910的框图。图9A中的实线框图示一般被包括在图形命令中的组成部分,而虚线包括任选的或仅被包括在图形命令的子集中的组成部分。图9A的示例性图形处理器命令格式900包括用于标识命令的客户端902、命令操作代码(操作码) 904和数据906的数据字段。子操作码905和命令大小908也被包括在一些命令中。

[0158] 在一些示例中,客户端902指定图形设备的、处理命令数据的客户端单元。在一些示例中,图形处理器命令解析器检查每一个命令的客户端字段以调整对命令的进一步的处理,并且将命令数据路由至适当的客户端单元。在一些示例中,图形处理器客户端单元包括存储器接口单元、渲染单元、2D单元、3D单元和媒体单元。每个客户端单元具有处理命令的对应的处理管线。一旦由客户端单元接收到命令,客户端单元就读取操作码904以及子操作码905(如果存在)以确定要执行的操作。客户端单元使用数据字段906中的信息来执行命令。针对一些命令,预期显式的命令大小908指定命令的大小。在一些示例中,命令解析器基于命令操作码自动地确定命令中的至少一些命令的大小。在一些示例中,命令经由双字的倍数被对齐。可使用其他命令格式。

[0159] 图9B中的流程图示示例性图形处理器命令序列910。在一些示例中,以图形处理器的示例为特征的数据处理系统的软件或固件使用所示出的命令序列的某个版本来建立、执行并终止图形操作的集合。仅出于示例目的示出并描述样本命令序列,因为示例不限于这些特定的命令或该命令序列。此外,命令可以作为批量的命令在命令序列中被发出,使得图形处理器将以至少部分地并发的方式处理命令序列。

[0160] 在一些示例中,图形处理器命令序列910能以管线转储清除命令912开始,以使任何活跃的图形管线完成用于管线的当前未决的命令。在一些示例中,3D管线922和媒体管线924不并发地操作。执行管线转储清除以使活跃的图形管线完成任何未决命令。响应于管线转储清除,用于图形处理器的命令解析器将暂停命令处理,直到活跃的绘画引擎完成未决操作并且相关的读缓存被无效。任选地,渲染缓存中被标记为“脏”的任何数据可以被转储清除到存储器。在一些示例中,管线转储清除命令912可用于管线同步,或可在将图形处理器置于低功率状态之前被使用。

[0161] 在一些示例中,当命令序列需要图形处理器在管线之间显式地切换时,使用管线选择命令913。在一些示例中,在发出管线命令之前在执行上下文中仅需要一次管线选择命令913,除非上下文是发出针对这两条管线的命令。在一些示例中,紧接在经由管线选择命令913进行的管线切换之前需要管线转储清除命令912。

[0162] 在一些示例中,管线控制命令914配置用于操作的图形管线,并且用于对3D管线922和媒体管线924进行编程。在一些示例中,管线控制命令914为活跃的管线配置管线状态。在一个示例中,管线控制命令914用于管线同步,并且用于在处理批量的命令之前清除来自活跃管线内的一个或多个缓存存储器的数据。

[0163] 在一些示例中,返回缓冲器状态命令916用于将用于相应管线的返回缓冲器的集合配置成用于写入数据。一些管线操作需要对一个或多个返回缓冲器的分配、选择或配置,在处理期间操作将中间数据写入这一个或多个返回缓冲器中。在一些示例中,图形处理器还使用一个或多个返回缓冲器以存储输出数据并执行跨线程通信。在一些示例中,返回缓冲器状态916包括选择要用于管线操作的集合的返回缓冲器的大小和数量。

[0164] 命令序列中的其余命令基于用于操作的活跃管线而不同。基于管线判定920,命令序列被定制成用于以3D管线状态930开始的3D管线922、或者在媒体管线状态940处开始的媒体管线924。

[0165] 用于配置3D管线状态930的命令包括用于顶点缓冲器状态、顶点元素状态、常量颜色状态、深度缓冲器状态、以及将在处理3D基元命令之前配置的其他状态变量的3D状态设置命令。这些命令的值至少部分地基于使用中的特定3D API来确定。在一些示例中,3D管线状态930命令也能够将在不会使用某些管线元件的情况下选择性地禁用或绕过那些元件。

[0166] 在一些示例中,3D基元932命令用于提交要由3D管线处理的3D基元。经由3D基元932命令传递给图形处理器的命令和相关联的参数被转发到图形管线中的顶点取得功能。顶点取得功能使用3D基元932命令数据来生成顶点数据结构。顶点数据结构被存储在一个或多个返回缓冲器中。在一些示例中,3D基元932命令用于经由顶点着色器对3D基元执行顶点操作。为了处理顶点着色器,3D管线922将着色器执行线程调遣给图形处理器执行单元。

[0167] 在一些示例中,3D管线922经由执行934命令或事件来触发。在一些示例中,寄存器写入触发命令执行。在一些示例中,经由命令序列中的“去往(go)”或“踢除(kick)”命令来触发执行。在一个示例中,命令执行使用管线同步命令以通过图形管线对命令序列转储清除来触发。3D管线将执行针对3D基元的几何处理。一旦操作完成,就对所得到的几何对象进行栅格化,并且像素引擎对所得到的像素进行着色。对于那些操作,还可以包括用于控制像素着色和像素后端操作的附加命令。

[0168] 在一些示例中,当执行媒体操作时,图形处理器命令序列910遵循媒体管线924路径。一般而言,针对媒体管线924进行编程的特定用途和方式取决于要执行的媒体或计算操作。在媒体解码期间,特定的媒体解码操作可被迁移到媒体管线。在一些示例中,也可绕过媒体管线,并且可使用由一个或多个通用处理核心提供的资源完全地或部分地执行媒体解码。在一个示例中,媒体管线还包括用于通用图形处理器单元(GPGPU)操作的元件,其中,图形处理器用于使用计算着色器程序来执行SIMD向量操作,这些计算着色器程序并不显式地与图形基元的渲染相关。

[0169] 在一些示例中,以与3D管线922类似的方式来配置媒体管线924。用于配置媒体管线状态940的命令的集合在媒体对象命令942之前被调遣或被放置到命令序列中。在一些示例中,用于媒体管线状态940的命令包括用于配置将被用于处理媒体对象的媒体管线元件的数据。这包括用于在媒体管线内配置视频解码和视频编码逻辑的数据,诸如编码或解码格式。在一些示例中,用于媒体管线状态940的命令还支持使用指向包含批量的状态设置的“间接”状态元素的一个或多个指针。

[0170] 在一些示例中,媒体对象命令942供应指向用于由媒体管线处理的媒体对象的指针。媒体对象包括存储器缓冲器,该存储器缓冲器包含要处理的视频数据。在一些示例中,在发出媒体对象命令942之前,所有的媒体管线状态必须是有效的。一旦管线状态被配置并且媒体对象命令942被排队,就经由执行命令944或等效的执行事件(例如,寄存器写入)来触发媒体管线924。随后可通过由3D管线922或媒体管线924提供的操作对来自媒体管线924的输出进行后处理。在一些示例中,以与媒体操作类似的方式配置和执行GPGPU操作。

[0171] 图形软件体系结构

[0172] 图10图示根据一些示例的用于数据处理系统1000的示例性图形软件体系结构。在

一些示例中,软件体系结构包括3D图形应用1010、操作系统1020以及至少一个处理器1030。在一些示例中,处理器1030包括图形处理器1032以及一个或多个通用处理器核心1034。图形应用1010和操作系统1020各自在数据处理系统的系统存储器1050中执行。

[0173] 在一些示例中,3D图形应用1010包含一个或多个着色器程序,该一个或多个着色器程序包括着色器指令1012。着色器语言指令可以采用高级着色器语言,诸如,Direct3D的高级着色器语言(High-Level Shader Language, HLSL)、OpenGL着色器语言(OpenGL Shader Language, GLSL),等等。应用还包括采用适于由通用处理器核心1034执行的机器语言的可执行指令1014。应用还包括由顶点数据限定的图形对象1016。

[0174] 在一些示例中,操作系统1020是来自微软公司的微软®Windows®操作系统、专属的类UNIX操作系统或使用Linux内核的变体的开放源类UNIX操作系统。操作系统1020可支持图形API 1022,诸如,Direct3D API、OpenGL API或Vulkan API。当Direct3D API在使用中时,操作系统1020使用前端着色器编译器1024以将采用HLSL的任何着色器指令1012编译成较低级的着色器语言。编译可以是即时(just-in-time, JIT)编译或者应用可执行着色器预编译。在一些示例中,在3D图形应用1010的编译期间,高级着色器被编译为低级着色器。在一些示例中,着色器指令1012以中间形式提供,该中间形式诸如,由Vulkan API使用的标准便携式中间表示(Standard Portable Intermediate Representation, SPIR)的某个版本。

[0175] 在一些示例中,用户模式图形驱动器1026包含后端着色器编译器1027以将着色器指令1012编译为硬件特定表示。当OpenGL API在使用中时,将采用GLSL高级语言的着色器指令1012传递至用户模式图形驱动器1026以用于编译。在一些示例中,用户模式图形驱动器1026使用操作系统内核模式功能1028来与内核模式图形驱动器1029通信。在一些示例中,内核模式图形驱动器1029与图形处理器1032通信以调遣命令和指令。

[0176] IP核心实现方式

[0177] 至少一个示例的一个或多个方面可以由存储在机器可读介质上的代表性代码实现,该机器可读介质表示和/或限定集成电路(诸如,处理器)内的逻辑。例如,机器可读介质可包括表示处理器内的各种逻辑的指令。当由机器读取时,指令可使机器制造用于执行本文所描述的技术的逻辑。此类表示(被称为“IP核心”)是集成电路的逻辑的可重复使用单元,这些可重复使用单元可以作为描述集成电路的组织的硬件模型而被存储在有形的、机器可读介质上。可以将硬件模型供应至在制造集成电路的制造机器上加载硬件模型的各客户或制造设施。可以制造集成电路,使得电路执行与本文中描述的示例中的任一示例相关联地描述的操作。

[0178] 图11A是图示根据示例的可用于制造集成电路以执行操作的IP核心开发系统1100的框图。IP核心开发系统1100可以用于生成可并入到更大的设计中或用于构建整个集成电路(例如,SOC(system on a chip,片上系统)集成电路)的模块化、可重复使用的设计。设计设施1130可生成采用高级编程语言(例如,C/C++)的IP核心设计的软件仿真1110。软件仿真1110可用于使用仿真模型1112来设计、测试并验证IP核心的行为。仿真模型1112可包括功能仿真、行为仿真和/或时序仿真。随后可从仿真模型1112创建或合成寄存器传输级(register transfer level, RTL)设计1115。RTL设计1115是对硬件寄存器之间的数字信号的流进行建模的集成电路(包括使用建模的数字信号来执行的相关联的逻辑)的行为的抽

象。除了RTL设计1115之外,还可创建、设计或合成逻辑级或晶体管级的较低级别设计。由此,初始设计和仿真的特定细节可有所不同。

[0179] 可由设计设施进一步将RTL设计1115或等效方案合成到硬件模型1120中,该硬件模型1120可以采用硬件描述语言(hardware description language,HDL)或物理设计数据的某种其他表示。可以进一步仿真或测试HDL以验证IP核心设计。可使用非易失性存储器1140(例如,硬盘、闪存或任何非易失性存储介质)来存储IP核心设计以用于递送至第三方制造设施1165。替代地,可通过有线连接1150或无线连接1160(例如,经由互联网)来传送IP核心设计。制造设施1165随后可制造至少部分地基于IP核心设计的集成电路。所制造的集成电路可被配置成用于执行根据本文中描述的至少一个示例的操作。

[0180] 图11B图示根据本文中所描述的一些示例的集成电路封装组件1170的截面侧视图。集成电路封装组件1170图示如本文中所描述的一个或多个处理器或加速器设备的实现方式。封装组件1170包括连接至衬底1180的多个硬件逻辑单元1172、1174。逻辑1172、1174可至少部分地在可配置逻辑或固定功能逻辑硬件中实现,并且可包括本文中描述的(一个或多个)处理器核心、(一个或多个)图形处理器或其他加速器设备中的任一者的一个或多个部分。每个逻辑单元1172、1174可在半导体管芯内实现,并且经由互连组织1173与衬底1180耦合。互连组织1173可被配置成用于在逻辑1172、1174与衬底1180之间路由电信号,并且可包括互连,该互连诸如但不限于凸块或支柱。在一些示例中,互连组织1173可被配置成路由电信号,诸如例如,与逻辑1172、1174的操作相关联的输入/输出(I/O)信号和/或功率或接地信号。在一些示例中,衬底1180是基于环氧树脂的层压衬底。在其他示例中,衬底1180可包括其他合适类型的衬底。封装组件1170可经由封装互连1183连接到其他电气设备。封装互连1183可耦合至衬底1180的表面以将电信号路由到其他电气设备,诸如主板、其他芯片组或多芯片模块。

[0181] 在一些示例中,逻辑单元1172、1174与桥接器1182电耦合,该桥接器1182被配置成用于在逻辑1172与逻辑1174之间路由电信号。桥接器1182可以是电信号提供路由的密集互连组织。桥接器1182可包括由玻璃或合适的半导体材料构成的桥接器衬底。电路由特征可形成在桥接器衬底上,以提供逻辑1172与逻辑1174之间的芯片到芯片连接。

[0182] 尽管图示了两个逻辑单元1172、1174和桥接器1182,但是本文中所描述的示例可包括在一个或多个管芯上的更多或更少的逻辑单元。这一个或多个管芯可以由零个或更多个桥接器连接,因为当逻辑被包括在单个管芯上时,可以排除桥接器1182。替代地,多个管芯或逻辑单元可以由一个或多个桥接器连接。此外,多个逻辑单元、管芯和桥接器可按其他可能的配置(包括三维配置)被连接在一起。

[0183] 图11C图示封装组件1190,该封装组件1190包括连接到衬底1180(例如,基础管芯)的硬件逻辑小芯片的多个单元。如本文中所描述的图形处理单元、并行处理器和/或计算加速器可由分开制造的各种硅小芯片组成。在该上下文中,小芯片是至少部分地被封装的集成电路,该至少部分地被封装的集成电路包括能够与其他小芯片一起被组装到更大的封装中的不同的逻辑单元。具有不同IP核心逻辑的各种小芯片集合能够被组装到单个器件中。此外,小芯片可使用有源中介层(interposer)技术而被集成到基础管芯或基础小芯片中。本文中描述的概念启用GPU内的不同形式的IP之间的互连和通信。IP核心能够使用不同的工艺技术来制造并在制造期间被构成,这避免了尤其是对于具有若干风格的IP的大型SoC

的将多个IP汇聚到同一制造工艺的复杂性。允许使用多种工艺技术改善了上市时间,并提供具有成本效益的方法来创建多个产品SKU。此外,分解的IP更易修改以被独立地功率门控,对于给定工作负载不在使用中的部件能够被关断,从而降低总功耗。

[0184] 硬件逻辑小芯片可包括专用硬件逻辑小芯片1172、逻辑或I/O小芯片1174、和/或存储器小芯片1175。硬件逻辑小芯片1172以及逻辑或I/O小芯片1174可以至少部分地实现在可配置逻辑或固定功能逻辑硬件中,并且可包括本文中描述的(一个或多个)处理器核心、(一个或多个)图形处理器、并行处理器或其他加速器设备中的任一个的一个或多个部分。存储器小芯片1175可以是DRAM(例如,GDDR、HBM)存储器或缓存(SRAM)存储器。

[0185] 每个小芯片可被制造为单独的半导体管芯,并且经由互连组织1173与衬底1180耦合。互连组织1173可被配置成用于在衬底1180内的各种小芯片与逻辑之间路由电信号。互连组织1173可包括互连,诸如但不限于凸块或支柱。在一些示例中,互连组织1173可被配置成用于路由电信号,诸如例如,与逻辑、I/O和存储器小芯片的操作相关联的输入/输出(input/output,I/O)信号和/或功率或接地信号。

[0186] 在一些示例中,衬底1180是基于环氧树脂的层压衬底。在其他示例中,衬底1180可包括其他合适类型的衬底。封装组件1190能够经由封装互连1183连接到其他电气设备。封装互连1183可耦合至衬底1180的表面以将电信号路由到其他电气设备,诸如,主板、其他芯片组或多芯片模块。

[0187] 在一些示例中,逻辑或I/O小芯片1174和存储器小芯片1175能够经由桥接器1187被电耦合,该桥接器1187被配置成用于在逻辑或I/O小芯片1174与存储器小芯片1175之间路由电信号。桥接器1187可以是为电信号提供路由的密集互连组织。桥接器1187可包括由玻璃或合适的半导体材料构成的桥接器衬底。电路特征可形成在桥接器衬底上以提供逻辑或I/O小芯片1174与存储器小芯片1175之间的芯片至芯片连接。桥接器1187还可被称为硅桥接器或互连桥接器。例如,在一些示例中,桥接器1187是嵌入式多管芯互连桥接器(Embedded Multi-die Interconnect Bridge,EMIB)。在一些示例中,桥接器1187可以简单地是从一个小芯片到另一小芯片的直接连接。

[0188] 衬底1180可包括用于I/O 1191、缓存存储器1192和其他硬件逻辑1193的硬件部件。结构1185可被嵌入在衬底1180中以启用衬底1180内的各种逻辑小芯片与逻辑1191、1193之间的通信。在一个示例中,I/O 1191、结构1185、缓存、桥接器和其他硬件逻辑1193可被集成到层叠在衬底1180的顶部上的基础管芯中。

[0189] 在各示例中,封装组件1190可包括由结构1185或一个或多个桥接器1187互连的更少或更多数量的部件和小芯片。封装组件1190内的小芯片能以3D布置或2.5D布置来布置。一般而言,桥接器组织1187可用于促进例如逻辑或I/O小芯片与存储器小芯片之间的点到点互连。结构1185可用于将各种逻辑和/或I/O小芯片(例如,小芯片1172、1174、1191、1193)与其他逻辑和/或I/O小芯片互连。在一个示例中,衬底内的缓存存储器1192可充当用于封装组件1190的全局缓存,充当分布式全局缓存的部分,或充当用于结构1185的专用缓存。

[0190] 图11D图示根据示例的包括可互换小芯片1195的封装组件1194。可互换小芯片1195可被组装到一个或多个基础小芯片1196、1198上的标准化插槽中。基础小芯片1196、1198可经由桥接器互连1197被耦合,该桥接器互连1197可与本文中描述的其他桥接器互连类似,并且可以是例如EMIB。存储器小芯片也可经由桥接器互连被连接到逻辑或I/O小芯

片。I/O和逻辑小芯片可经由互连结构进行通信。基础小芯片各自都能以用于逻辑或I/O或存储器/缓存中的一者的标准化格式来支持一个或多个插槽。

[0191] 在一个示例中,SRAM和功率递送电路可被制造到基础小芯片1196、1198中的一个或多个中,基础小芯片1196、1198可使用相对于可互换小芯片1195不同的工艺技术来制造,可互换小芯片1195堆叠在基础小芯片的顶部上。例如,可使用较大工艺技术来制造基础小芯片1196、1198,同时可使用较小工艺技术来制造可互换小芯片。可互换小芯片1195中的一个或多个可以是存储器(例如,DRAM)小芯片。可基于针对使用封装组件1194的产品的功率和/或性能来为封装组件1194选择不同的存储器密度。此外,可在组装时基于针对产品的功率和/或性能来选择具有不同数量的类型的功能单元的逻辑小芯片。此外,可将包含具有不同类型的IP逻辑核心的小芯片插入到可互换小芯片插槽中,从而启用可混合并匹配不同技术的IP块的混合式处理器设计。

[0192] 示例性片上系统集成电路

[0193] 图12和图13A-图13B图示根据本文中描述的各个示例的可使用一个或多个IP核心制造的示例性集成电路和相关联的图形处理器。除了所图示的内容之外,还可包括其他逻辑和电路,包括附加的图形处理器/核心、外围接口控制器或通用处理器核心。

[0194] 图12是图示根据示例的可使用一个或多个IP核心来制造的示例性片上系统集成电路1200的框图。示例性集成电路1200包括一个或多个应用处理器1205(例如,CPU)、至少一个图形处理器1210并且可附加地包括图像处理器1215和/或视频处理器1220,图像处理器1215和视频处理器1220中的任一者可以是来自相同的设计设施或多个不同的设计设施的模块化IP核心。集成电路1200包括外围或总线逻辑,包括USB控制器1225、UART控制器1230、SPI/SDIO控制器1235和I²S/I²C控制器1240。此外,集成电路可包括显示设备1245,该显示设备1245耦合至高清晰度多媒体接口(high-definition multimedia interface, HDMI)控制器1250和移动行业处理器接口(mobile industry processor interface, MIPI)显示接口1255中的一个或多个。可以由闪存子系统1260(包括闪存和闪存控制器)来提供存储。可以经由存储器控制器1265来提供存储器接口以获得对SDRAM(synchronous dynamic random access memory, 同步动态随机存取存储器)或SRAM存储器设备的访问。一些集成电路附加地包括嵌入式安全引擎1270。

[0195] 图13A-图13B是图示根据本文中所描述的示例的用于在SoC内使用的示例性图形处理器的框图。图13A图示根据示例的可以使用一个或多个IP核心来制造的片上系统集成电路的示例性图形处理器1310。图13B图示根据示例的可以使用一个或多个IP核心来制造的片上系统集成电路的附加的示例性图形处理器1340。图13A的图形处理器1310是低功率图形处理器核心的示例。图13B的图形处理器1340是较高性能的图形处理器核心的示例。图形处理器1310和1340中的每个图形处理器可以是图12的图形处理器1210的变体。

[0196] 如图13A中所示,图形处理器1310包括顶点处理器1305和一个或多个片段处理器1315A-1315N(例如,1315A、1315B、1315C、1315D,一直到1315N-1和1315N)。图形处理器1310可以经由单独的逻辑执行不同的着色器程序,使得顶点处理器1305被优化以执行用于顶点着色器程序的操作,而一个或多个片段处理器1315A-1315N执行用于片段或像素着色器程序的片段(例如,像素)着色操作。顶点处理器1305执行3D图形管线的顶点处理阶段,并生成基元和顶点数据。(一个或多个)片段处理器1315A-1315N使用由顶点处理器1305生成的基

元数据和顶点数据来产生被显示在显示设备上的帧缓冲器。在一个示例中, (一个或多个) 片段处理器1315A-1315N被优化以执行如在OpenGL API中提供的片段着色器程序, 这些片段着色器程序可以用于执行与如在Direct 3D API中提供的像素着色器程序类似的操作。

[0197] 图形处理器1310附加地包括一个或多个存储器管理单元 (MMU) 1320A-1320B、(一个或多个) 缓存1325A-1325B以及 (一个或多个) 电路互连1330A-1330B。该一个或多个MMU 1320A-1320B为图形处理器1310 (包括为顶点处理器1305和/或 (一个或多个) 片段处理器1315A-1315N) 提供虚拟到物理地址映射, 除了存储在一个或多个缓存1325A-1325B中的顶点数据或图像/纹理数据之外, 该虚拟到物理地址映射还可以引用存储在存储器中的顶点数据或图像/纹理数据。在一个示例中, 一个或多个MMU 1320A-1320B可以与系统内的其他MMU同步, 使得每个处理器1205-1220可以参与共享或统一的虚拟存储器系统, 系统内的其他MMU包括与图12的一个或多个应用处理器1205、图像处理器1215和/或视频处理器1220相关联的一个或多个MMU。根据示例, 一个或多个电路互连1330A-1330B使得图形处理器1310能够经由SoC的内部总线或经由直接连接来与SoC内的其他IP核心对接。

[0198] 如图13B中所示, 图形处理器1340包括图13A的图形处理器1310的一个或多个MMU 1320A-1320B、缓存1325A-1325B和电路互连1330A-1330B。图形处理器1340包括一个或多个着色器核心1355A-1355N (例如, 1355A、1355B、1355C、1355D、1355E、1355F, 一直到1355N-1和1355N), 其提供统一的着色器核心体系结构, 其中, 单个核心或单个类型的核心可执行所有类型的可编程着色器代码, 包括用于实现顶点着色器、片段着色器和/或计算着色器的着色器程序代码。存在的着色器核心的确切数量可以因示例和实现方式而异。此外, 图形处理器1340包括核心间任务管理器1345, 该核心间任务管理器1345充当用于将执行线程调遣给一个或多个着色器核心1355A-1355N的线程调遣器和用于加速对基于片的渲染的分片操作的分片单元1358, 在基于片的渲染中, 针对场景的渲染操作在图像空间中被细分, 例如以利用场景内的局部空间一致性或优化内部缓存的使用。

[0199] 图14图示具有向量单元和矩阵单元的图形处理器的示例。图形单元1400是图形处理单元的示例, 其可以包括集成到共同SOC (system on a chip, 片上系统) 上的多个芯片。图形单元1400可以是图形处理器。

[0200] 图形单元1400包括硬件调度器1402, 硬件调度器1402表示用于跨图形单元1400的不同计算核心执行调度的电路。图形单元1400包括共享指令缓存1404, 共享指令缓存1404表示用于不同计算单元的指令的临时存储装置。在一个示例中, 每个切片1410具有在该切片上的计算核心之间共享的单独的指令缓存。在一个示例中, 共享指令缓存1404表示硬件调度器1402的部分, 该硬件调度器1402用于分发指令以供在图形单元1400中并行执行。

[0201] 在一个示例中, 图形单元1400包括由N个切片 (切片1410[0: (N-1)], 统称为切片1410) 共享的纹理单元1462。在一个示例中, 图形单元1400包括由切片1410共享的栅格化器1464。图形单元1400可以包括多个切片1410[10: (N-1)] 或分区, 这些切片或分区各自包括多个图形核心。切片1410可以包括支持逻辑, 支持逻辑包括本地指令缓存1412、本地线程调度器1414和线程调遣器1416。本地指令缓存1412可以缓存用于个体切片中的核心的指令, 本地线程调度器1414可以将指令分发给该切片中的并行计算核心。线程调遣器1416可以负责时间并行指令分发。在一个示例中, 切片1410包括寄存器/寄存器堆1418, 寄存器/寄存器堆1418表示用于存储数据或指令以供计算核心使用的寄存器。在一个示例中, 寄存器/寄存

器堆1418表示寄存器的结构化集合,以保存用于迭代计算的值。

[0202] 在一个示例中,切片1410包括向量单元1422和矩阵乘法(matrix multiplication,MXM)单元1424。向量单元1422和MXM单元1424可以执行逻辑操作。向量单元1422和MXM单元1424可以包括用于执行逻辑操作的计算的计算单元。计算单元可以包括附加功能单元(additional function unit,AFU)1432、浮点单元(floating-point unit,FPU)1434、整数算术逻辑单元(integer arithmetic logic unit,ALU)1436、地址计算单元(address computational unit,ACU)1438、双精度浮点单元(double-precision floating-point unit,DPFPU)1440和矩阵处理单元(matrix processing unit,MPU)1442。向量单元1422可能并非包括所有的计算单元(例如,向量单元1422可以不包括MPU 1442)。类似地,MXM 1424可能并非包括所有的计算单元。

[0203] 这些计算单元中的一些计算单元以特定精度进行操作。例如,FPU 1434可执行单精度(32比特)和半精度(16比特)浮点操作,而DPFPU 1440执行双精度(64比特)浮点操作。ALU 1436可以以8比特精度、16比特精度和32比特精度执行可变精度整数操作,并且可以被配置用于混合精度操作。MPU 1442也可以被配置用于混合精度矩阵操作,包括半精度浮点操作和8比特整数操作。MPU 1442可执行各种矩阵操作以加速机器学习应用框架,包括启用对加速的通用矩阵-矩阵乘法(general matrix to matrix multiplication,GEMM)的支持。AFU 1432可执行不被浮点单元或整数单元支持的附加逻辑操作,包括三角函数操作(例如,正弦、余弦等)。

[0204] 在一个示例中,切片1410包括在计算单元之间共享的渲染单元。渲染1450表示渲染单元,其可以包括光线追踪1452和采样器1454。渲染1450可以包括其他渲染单元。在一个示例中,图形单元1400包括在切片1410之间共享的缓存/共享存储器1470。在一个示例中,缓存/共享存储器1470表示第二级(L2)缓存。在一个示例中,缓存/共享存储器1470是可以作为计算单元之间的通信结构操作的、存储器结构的部分。

[0205] 图15图示具有并行计算集群的图形处理单元的示例。系统1500表示适合部署在多芯片模块或SOC上的高度并行的通用图形处理单元(general-purpose graphics processing unit,GPGPU)。

[0206] 系统1500使得能够由图形处理单元的阵列执行高度并行的计算操作。在一个示例中,系统1500表示直接被链接到其他GPGPU实例的GPGPU实例,以创建具有高带宽、高速操作的多GPU集群,以用于诸如深度神经网络的更快训练之类的应用。系统1500包括主机接口1512,以对诸如CPU或CPU SOC之类的主机处理器提供接口。在一个示例中,主机接口1512是PCI快速(PCI Express)接口。替代地,主机接口1512可以是供应方特定的通信接口或通信结构。

[0207] 系统1500包括全局调度器1514,以用于将与由主机接口1512接收的命令相关联的执行线程分发给计算集群1520[0:3](统称为计算集群1520)。系统1500图示四个计算集群1520,但是将理解,系统1500可以包括更多或更少的计算机集群。在一个示例中,计算集群1520共享缓存存储器1530。缓存存储器1530为计算集群1520内的缓存存储器(未具体示出)提供较高级别的缓存。

[0208] 系统1500包括存储器1552A和存储器1552B,统称为存储器1552。存储器1552A和存储器1552B可以表示不同的存储器通道。系统1500可以包括更多或更少的存储器通道。无论

系统1500中包括多少存储器通道或多少存储器,存储器1552都可以通过存储器控制器1550A和存储器控制器1550B(统称为存储器控制器1550)而与计算集群1520耦合,以分别与存储器1552A和存储器1552B耦合。存储器控制器1550管理对相关联的存储器设备的访问。存储器1552可以是或可以包括不同类型的存储器设备,诸如,动态随机存取存储器(dynamic random-access memory, DRAM)或图形随机存取存储器,诸如,同步图形随机存取存储器(synchronous graphics random access memory, SGRAM),包括图形双倍数据速率(graphics double data rate, GDDR)存储器。

[0209] 在一个示例中,每个计算集群1520表示根据图形单元1400的切片1410的示例的切片。计算集群1520可以各自包括多种类型的整数和浮点逻辑单元,以便以不同的精度执行计算操作。例如,计算集群1520中的每一个计算集群1520中的浮点单元的至少子集可被配置成用于执行16比特或32比特浮点操作,而浮点单元的不同子集可被配置成用于执行64比特浮点操作。

[0210] 计算集群1520可以使用不同的同步和数据交换机制来彼此通信。在一个示例中,计算集群1520经由主机接口1512进行通信。在一个示例中,系统1500包括将系统1500与GPU链路1544耦合的I/O中枢1542。在一个示例中,GPU链路1544耦合至专用的GPU至GPU桥接器,该GPU至GPU桥接器实现系统1500的多个实例之间、以及不同的计算集群之间的通信和同步。在一个示例中,GPU链路1544与高速互连耦合,以将数据传送和接收至其他GPGPU或并行处理器。在一个示例中,系统1500的多个实例可以位于分开的数据处理系统中,并且经由网络设备进行通信,该网络设备可经由主机接口1512来访问。在一个示例中,附加于或替代于主机接口1512,GPU链路1544可以被配置成用于启用至主机处理器的连接。

[0211] 通用可消耗的数据压缩

[0212] 图16图示具有用于在图形处理器之间传输的压缩数据分组的系统的示例。系统1600表示具有CPU 1610和GPU 1650[0:(N-1)](统称为GPU 1650)的计算机系统。GPU 1650表示图形单元,其可以是独立的图形处理单元,或者是图形处理单元的部分或分区。在一个示例中,GPU 1650表示具有计算核心的切片或芯片。

[0213] CPU 1610表示通用处理器。CPU 1610可以包括一个或多个核心1612。核心1612执行通用操作(general purpose operation, GP OP)1614。GP OP 1614可以表示计算机设备上的任何类型的计算或操作。CPU 1610可以访问存储器,并且可以基于由核心1612执行的操作向GPU 1650发送操作。

[0214] 在一个示例中,系统1600包括非易失性存储器1632,用于存储不被GPU 1650或CPU 1610活跃使用的数据。非易失性存储器1632表示系统1600中的、即使在至系统的电力被中断时也维持确定的状态的存储装置。CPU 1610可以包括文件系统1630作为CPU 1610上的非易失性文件系统应用或其他文件系统服务,作为操作系统(operating system, OS)或主要执行例程的部分。OS提供包括系统硬件接口管理和驱动器的软件平台,以允许在OS下运行的应用访问计算环境的硬件资源。OS可以通过文件系统1630管理对非易失性存储器1632的访问。

[0215] 统一存储器1642表示系统1600的系统存储器或主存储器。统一存储器1642通常是或包括易失性存储器,易失性存储器在至存储器的电力被中断长于短时间段(即,毫秒)的情况下具有不确定的状态。在一个示例中,统一存储器1642包括GPGPU存储器1644,GPGPU存

存储器1644表示被分配以供GPU 1650使用的存储器。CPU 1610的存储器控制器1620管理CPU 1610对统一存储器1642的访问。

[0216] CPU 1610可以通过总线1640通信地耦合至GPU 1650和统一存储器1642。总线1640表示CPU 1610与GPU 1650以及存储器之间的高速通信连接。在一个示例中,总线1640是PCIe(peripheral component interconnect express,外围组件互连快速)总线。可以在CPU 1610与GPU 1650之间使用其他总线或互连。CPU 1610可以通过发起GPU 1650上的应用的实例来将操作迁移到GPU 1650。在一个示例中,系统1600包括用于互连GPU 1650的GPU链路1660。在一个示例中,GPU链路1660与总线1640相同。在一个示例中,GPU 1650包括单独的GPU到GPU链路或结构来进行互连,以供进行GPU 1650之间的数据交换。

[0217] GPU 1650包括图形核心1652或其他图形处理硬件。在一个示例中,GPU 1650包括缓存1654,缓存1654表示核心1652可用的一个或多个级别的缓存。在一个示例中,缓存1654包括第一级缓存(L1或L1\$(L1缓存))和第二级缓存(L2或L2\$(L2缓存))。在一个示例中,GPU 1650包括存储压缩数据的L2缓存,并且当数据从L1移动到L2时系统提供按需压缩,并且当数据从L2移动到L1时系统提供按需解压缩。压缩/解压缩(Compression/decompression, COMP/DECOMP) 1658表示GPU 1650中用于提供按需压缩/解压缩操作的压缩和解压缩引擎。压缩/解压缩1658可以是或者包括分布式压缩管理硬件。

[0218] 在一个示例中,GPU 1650包括存储器1656,存储器1656表示特定GPU实例的本地存储器资源。存储器1656可以表示特定于GPU实例的易失性存储器资源。系统1600可以包括用于GPU 1650的共享存储器。存储器1656表示一个GPU上的非共享资源。

[0219] NIC(network interface circuit,网络接口电路) 1662[0:(N-1)](统称为NIC 1662)表示使得各个GPU 1650能够通过GPU链路1660访问其他GPU的电路。系统1600可以包括第一图形处理器(例如,GPU 1650[0]),第一图形处理器将数据块作为压缩数据存储在缓存1654或存储器1656中。压缩数据具有数据块,这些数据块具有共同的占用空间(footprint)但具有可变的数据量,这取决于存储在压缩块中的数据的压缩比。例如,无损压缩系统可以以8:8(无压缩)到8:1(最高压缩)之间的任意比率来压缩数据。

[0220] 在一个示例中,CPU 1610包括压缩管理器1622,压缩管理器1622表示CPU 1610中的压缩逻辑。在一个示例中,系统1600包括压缩管理器1670,压缩管理器1670表示可由系统1600的多个部件访问的压缩管理器。无论是在整个系统1600中以分布式方式实现,还是在统一的压缩/解压缩引擎中(诸如,利用压缩管理器1670)实现,系统1600中的(一个或多个)压缩管理器都可以为系统1600中的通用消耗提供压缩和解压缩。通用消耗可以指使用不同压缩格式的部件可以彼此压缩和共享数据的事实。因此,使用一种压缩格式的部件可以压缩数据,该数据随后可以被使用相同或不同压缩格式的另一部件解压缩并使用。系统1600使得解压缩器能够仅利用指向压缩数据块的指针来对数据解压缩,而不需要伴随数据的其他信息。

[0221] 在一个示例中,系统1600具有单个压缩器以及分布式或分散式解压缩,该单个压缩器由压缩管理器1670表示,可以与统一存储器1642相关联。分布式或分散式解压缩使得系统1600内的不同部件能够在本地接收压缩数据并将该压缩数据解压缩以供使用,而不论该数据是否使用相同类型的压缩被压缩。本地解压缩可以能够对不同的压缩格式解压缩,从而允许对不同压缩类型的压缩数据的访问。在一个示例中,系统1600具有分布式压缩,利

用不同的压缩管理器来以无状态格式压缩数据,该数据可以在其他分布式位置处被读取和解压缩。

[0222] 传统上,片是受到压缩的数据的生产方,并且压缩块无法在片之间被共享。传统上,表面所有权需要与块边界对齐,并且是只读的。通用可消耗压缩的应用可以包括数据的无状态压缩和无状态解压缩。

[0223] 通过无状态压缩,表面状态格式变得不相关,并且压缩数据可以被不同的部件共享。在一个示例中,无状态压缩数据将CMF (compression format,压缩格式) 应用于不使用表面状态的客户端单元的单个硬编码值。单个硬编码值可以是用于标识CMF格式的查找表的值。在一个示例中,压缩管理器可以在客户端处对CMF值进行硬编码,或者将CMF值附接到可编程MMIO(memory mapped input/output,存储器映射的输入/输出) 寄存器或寄存器堆。

[0224] 通过无状态解压缩,压缩管理器可以在不知道压缩数据的任何表面细节的情况下执行解压缩。例如,如果CPU 1610想要从统一存储器1642访问压缩数据的页,则无论使用什么格式来压缩数据,压缩管理器都可以使用复制引擎对数据就地解压缩。系统1600可以将页移动到另一位置(诸如,交换文件或不同的物理地址),或者读入到CPU中以供执行。因此,经解压缩的页可以安全地恢复到系统1600内的不同位置并由GPU 1650使用。

[0225] 作为块压缩的部分,压缩引擎可以将经压缩的代码嵌入到压缩数据中。因此,解压缩可以在压缩数据的读回时访问压缩信息。在一个示例中,可以通过生成压缩数据的报头来执行嵌入压缩信息,该报头包括CMF指示符。因此,解压缩器仅需要使用压缩数据作为输入,而不需要状态信息。

[0226] 在一个示例中,通用可消耗的压缩数据具有标准化的格式化以实现无状态压缩。例如,压缩数据可以具有有着共同占用空间的块,这些块具有基于有多少数据被压缩的变化的数据量。在一个示例中,压缩数据不具有相关联的表面类型。传统上,压缩数据具有相关联的表面类型,这可能在表面类型不匹配的情况下导致压缩数据与消耗方不兼容。在一个示例中,压缩数据具有统一布局。所有数据可以共享相同的格式,而不需要配对比特来提供格式信息。因此,对表面进行写入的任何部件可以使用相同的格式对该表面进行写入,并在报头中提供CMF信息,从而使得其他部件能够仅利用压缩数据块作为输入来对数据进行解压缩和使用。在一个示例中,数据具有统一存储器布局。例如,数据可以全部以行为主格式被布局。在一个示例中,数据全部以列为主格式被布局。

[0227] 在一个示例中,来自缓存1654的数据可以通过应用通用可消耗压缩格式的按需压缩被驱逐到存储器1656。数据随后可以被其他GPU或其他部件使用。另外,压缩数据可以从存储器1656被移动到统一存储器1642(诸如,在GPGPU存储器1644中),并随后被读入CPU 1610。因此,压缩的存储器可以通过解压缩器从存储器流出PCIe链路或其他通信链路,并进入另一个部件。压缩数据不需要被固定到系统1600内的特定存储器设备。

[0228] 图17图示图形压缩体系结构的示例。体系结构1700表示根据系统1600的示例的系统的压缩体系结构。体系结构1700提供用于压缩数据的数据分组的替代结构。

[0229] 存储器1710表示压缩数据的存储器存储装置。如所图示的,压缩数据在缓存或存储器设备中使用恒定的占用空间,但存储的数据量根据压缩比而变化。例如,考虑D2处存储的数据量(压缩比为8:1)与D1处的数据(压缩比为8:6)相比的差异。所图示的其他压缩比是8:2、8:4和8:5。所存储的数据量不一定是按比例,但不同的数据块说明了基于压缩比的

相对差异。

[0230] 在一个示例中,体系结构1700表示具有内部无损压缩的GPU体系结构的压缩细节。在一个示例中,体系结构1700包括CCS(compression control surface,压缩控制表面)表1720。CCS表1720包括跟踪存储器1710的分立的存储器块的压缩大小的多个CCS条目。每个CCS条目可以表示块压缩比状态。CCS表1720被图示出为具有与存储器1710相当的布局,以表示数据块具有用于指示其压缩比的相关联的CCS条目的事实。由此,像存储器1710一样,CCS表1720具有由字母(A、B、C、D、E、...)指示的行和由数字(1、2、3、4、...)指示的列。将要理解的是,并非所有数据均被图示。可以以数据块组的形式访问数据。

[0231] 在一个示例中,存储器1710的数据块的大小是256B并且支持以32B大小增量的、8:1至8:8压缩之间的压缩比。在一个示例中,块以恒定的占用空间布局、作为压缩数据被存储在本地存储器中。使用恒定的占用空间布局,每个块在存储器中的起始位置基于完整块大小,而不论压缩比如何。由此,压缩不一定会节省存储器空间,但会节省用于从存储器读取的存储器带宽。

[0232] 存储器1710的数据及其相关联的CCS条目可以被串接在一起,以形成固定大小的分组,以用于在系统中的部件之间传输。在一个示例中,分组大小是固定的,其中将对每个传输使用相同的分组大小。然而,分组中所包括的数据块的数量可以基于压缩比而变化。由此,固定大小的分组将包含可变的数据量。

[0233] 块1740表示具有报头1750的数据块实现方式,具有多个压缩数据块。报头1750可以是用于存储CCS元数据和压缩格式(CMF)信息的报头。在一个示例中,报头1750包括CMF 1752以指示用于对块1740进行压缩的压缩格式。

[0234] 在一个示例中,报头1750包括CCS条目1754作为针对块的CCS信息。在一个示例中,报头1750包括其他信息1756。其他信息1756可以包括可以被存储在报头中以指示关于压缩块的某些信息的任何其他元数据。块1740包括压缩数据1742,压缩数据1742表示块1740中的压缩数据块。

[0235] 报头1750可以在存储块的属性时被引用。在一个示例中,CMF 1752可以具有多个比特(例如,4比特或其他数量)来指示不同的压缩实现方式。在一个示例中,CMF 1752指示作为用于解压缩器标识解压缩格式并应用其对数据块进行解压缩的查找表的条目的值。

[0236] 在一个示例中,报头1750不需要一个或多个配对比特,因为系统将共同布局应用于所有压缩格式。在一个示例中,CMF 1752包括像素格式信息,从而允许向量提供像素格式和压缩格式信息。这样,解压缩器可以从向量提取像素格式信息,并驱动该解压缩器执行解压缩。

[0237] 在一个示例中,CMF 1752指示表示压缩代码的值。CCS代码被设计用于传达压缩比和clear_n(清除_n)值,而CMF 1752是报头中的字段,用于提供可标识用于压缩/解压缩的算法的代码。压缩算法的示例可以包括双峰压缩和Z平面压缩。同一系统内也可能有其他压缩算法或压缩格式。

[0238] 在一个示例中,系统提供清除值的硬件管理。系统可以应用M个(在一个示例中,M等于8)硬编码的全局值,该M个硬编码的全局值与CCS代码中的M个Clear_N代码相对应。在一个示例中,如果整个数据块与来自查找表的Clear_N值中的一个匹配,则压缩器将分配匹配的Clear_N代码。在一个示例中,数据块具有256B。在一个示例中,无论CMF格式设置如何,

都考虑所有的Clear_N。在解压缩器中,系统可以将CCS Clear_N代码转换回未压缩的数据。

[0239] 图18图示提供通用可消耗的数据压缩的系统的示例。系统1800表示根据系统1600的示例的系统。系统1800包括存储器1810,存储器1810表示统一存储器或可由生成压缩数据、或消耗压缩数据、或既生成又消耗压缩数据的多个不同部件访问的存储器。

[0240] 存储器控制器1820管理对存储器1810的访问。在一个示例中,系统1800包括压缩管理器1830,用于管理去往和来自存储器1810的数据的压缩。压缩管理器1830可以提供根据本文的任何示例的压缩和解压缩。在一个示例中,压缩管理器1830是共享压缩引擎。在一个示例中,压缩管理器1830表示分散式压缩引擎,其中利用遍及系统的压缩器和解压缩器的多个实例向部件提供压缩服务。

[0241] 系统1800示出通过管线1842连接到存储器1810的部件1840、通过管线1852连接到存储器1810的部件1850、通过管线1862连接到存储器1810的部件1860、以及通过管线1872连接到存储器1810的CPU 1870。部件1840、部件1850和部件1860中的任一个可以表示不同的功能单元,诸如媒体堆栈或媒体管线,诸如视频堆栈、3D彩色管线、显示器、相机、AI (artificial intelligence,人工智能) 加速器、或其他部件。CPU 1870可以被认为用于压缩数据的目的的部件,因为它可以访问压缩数据并且需要将其解压缩以用于计算。系统1800可以包括多个一种类型的功能单元、多个不同类型的功能单元、或者多个功能单元的组合,在该组合中至少两个功能单元是相同类型的功能单元并且至少一个功能单元是不同类型的功能单元。

[0242] 在一个示例中,系统1800表示具有分立图形单元的系统。分立图形单元具有作为与计算机设备的中央处理单元分开的部件的图形处理单元。分立图形单元通过延伸超出处理器封装的通信链路耦合到处理器。

[0243] 在一个示例中,系统1800表示具有集成图形单元的系统。集成图形单元具有作为中央处理器SOC的部件的图形处理单元。集成图形单元具有通信链路,该通信链路通过处理器SOC封装内的通信链路耦合到处理器。

[0244] 通过分立图形单元实现方式中的存储器侧缓存,部件有机会利用存储器侧缓存来创建压缩片。为了存储器侧缓存的更高效使用,某些段可能需要数据在存储器侧缓存之前被压缩。

[0245] 在一个示例中,系统1800允许部件单元预压缩到L2缓存中,或者依赖于共享缓存来创建压缩片并且随后对数据进行压缩。与解压缩类似,在整个系统中对于在核心块附近放置解压缩器存在灵活性。在一个示例中,核心块可以是具有媒体管线的媒体块、具有颜色管线的3D块、具有显示管线的显示块、具有视频捕捉计算管线的视频捕捉块或具有人工智能(AI) 加速器管线的AI加速器或可以使用存储器结构中的共同解压缩器的其他块。在一个示例中,系统1800实现缩放到具有不同面积和功率/性能要求的不同段。

[0246] 在一个示例中,系统1800是可缩放的,因为消耗压缩数据的任何类型的部件可以从生成压缩数据的任何其他类型的部件访问压缩数据。考虑其中部件1840是被压缩存储在存储器1810中的压缩数据的生产方而部件1850是消耗方的示例。一些部件可以通过L2或L3 (第三级) 缓存被耦合,而其他部件不通过L2或L3 (第三级) 缓存被耦合。允许部件从存储器1810访问压缩数据使得系统1800能够允许部件共享压缩数据。

[0247] 在一个示例中,部件1840、部件1850、部件1860和CPU 1870中的每一个可以产生压

缩数据、消耗压缩数据、或者既产生又消耗压缩数据。消耗压缩数据是指从存储器读取压缩数据以用于在部件处进行本地处理。产生压缩数据是指生成以压缩形式被存储在存储器中的内容。产生压缩数据可以包括生成具有CMF信息以向压缩器指示如何解释数据的报头。压缩器可以选择适当的格式适配设置以及内部压缩器管线配置来解释数据。

[0248] 在压缩操作期间,格式用于获得最大压缩比。在一个示例中,格式被存储在最终的压缩向量中,以使解压缩器能够反转所有格式适配步骤以恢复原始数据。通用可消耗的压缩数据通过允许在压缩期间选择适当的格式来允许最佳压缩,同时允许读取侧的通用消耗。读取期间支持通用消耗,因为客户端不需要发送CMF,而是可以从压缩数据本身推断出CMF。

[0249] 在一个示例中,通用消耗允许任何部件支持解压缩(只要该部件具有解压缩器)。由此,系统1800可以支持对不支持压缩的数据目标目的地的内联解压缩。内联解压缩的使用的一个示例是在分立的GFX(graphics,图形)中,至系统存储器的写入在数据流向系统存储器时引起自动的内联解压缩。

[0250] 在一个示例中,压缩部件可以在通信结构或数据结构上发送预压缩通信量(traffic)以及控制数据。在一个示例中,具有地址的附加数据可用于携带与压缩块(例如,256B)相关联的控制平面数据。在一个示例中,压缩部件可以指示绕过压缩。例如,CPU 1870可以在某些情况下绕过压缩。对于读取,CPU SOC可以支持将压缩请求发送回客户端部件或将未压缩请求发送回客户端部件。

[0251] 在一个示例中,多个压缩部件可以对所有读取数据量应用远端解压缩器。例如,部件可以通过内联解压缩器请求数据,从而接收解压缩数据,而不是接收压缩数据并将该压缩数据本地解压缩到客户端。在一个示例中,压缩部件可以请求从存储器1810返回压缩数据。各个压缩部件或客户端可以具有靠近客户端的解压缩器以优化性能。此类部件可以包括解压缩器,并且潜在地在部件内部包括超取得(overfetch)缓存。

[0252] 在一个示例中,管线1842、管线1852、管线1862和管线1872中的一个或多个表示用于与存储器1810交换压缩数据的硬件管线。在一个示例中,硬件管线中的一个或多个支持压缩数据和未压缩数据两者的传输。在一个示例中,存储在存储器1810中的压缩数据具有共同的存储器占用空间大小。压缩数据块可以包括具有压缩格式信息和CCS的报头。压缩管理器1830可以表示压缩引擎,该压缩引擎可以对要存储在存储器1810中的数据进行压缩,并且对从存储器1810读取的数据进行解压缩。

[0253] 图19图示具有可以传输压缩数据的数据结构的系统的示例。系统1900表示用于与根据系统1600或系统1800的系统一起使用的通信结构。系统1900实现用于压缩数据的通信结构。

[0254] 在一个示例中,系统1900包括存储器1910以存储多个部件的数据。在一个示例中,系统1900包括对存储器1910提供接口的存储器桥接器1920并且包括交叉开关1930,交叉开关1930表示选择性地将一个链路与另一链路连接以实现跨部件的通信的切换结构。在一个示例中,交叉开关1930和存储器桥接器1920在共享被压缩存储的数据的图形部件之间提供通信结构。在一个示例中,通信结构基于块压缩比支持多个数据传输大小。在一个示例中,通信结构支持压缩数据的传输和未压缩数据的传输。

[0255] 在一个示例中,系统1900包括压缩器/解压缩器(compressor/decompressor,

COMP/DECOMP) 1940以执行数据的压缩和解压缩。在一个示例中,压缩器/解压缩器1940可以对以压缩格式跨通信结构传输的数据执行内联解压缩。在一个示例中,压缩器/解压缩器1940可以执行内联压缩,以跨部件之间的通信结构、以压缩格式发送数据。在一个示例中,可以绕过压缩器/解压缩器1940以使得能够在实现发送压缩数据的系统中发送未压缩/解压缩数据。

[0256] 在一个示例中,系统1900中表示的通信结构可以支持不同的块大小。在一个示例中,系统1900使得部件能够将32字节(32B)到256字节(256B)的任何值放置在结构上。在一个示例中,系统1900支持64字节的未压缩请求,而可压缩/压缩请求可以是从32字节到256字节请求的任何值。在一个示例中,系统1900的结构支持用于写入的可压缩和不可压缩通信量。

[0257] 节点1950[0:(M-1)](统称为节点1950)表示系统1900中的M个节点。节点1950可以表示作为数据生产方或数据消耗方的各种客户端单元中的任一个。节点1950可以选择性地通过压缩器/解压缩器1940彼此耦合和/或耦合到存储器1910,以用于跨通信结构的数据的按需压缩服务。

[0258] 图20图示具有分布式压缩和解压缩的系统的示例。系统2000表示传输数据的系统,该数据可以包括根据系统1600、系统1800或系统1900的示例的压缩数据。在一个示例中,系统2000表示图形单元。在一个示例中,系统2000表示分立的图形单元。

[0259] 在一个示例中,系统2000包括存储器2010以存储数据2012。数据2012表示压缩数据。存储器2010可以存储与数据2012相关联的CCS信息。在一个示例中,系统2000包括L2缓存2030或由多个图形部件共享的其他较高级别的缓存。在一个示例中,L2缓存2030可以存储由数据2032表示的未压缩数据和由数据2034表示的压缩数据。

[0260] 在系统2000的一个示例中,系统具有分散式压缩管理。在一个示例中,系统2000包括靠近存储器2010的压缩引擎或压缩管理器。压缩管理器可以被实现为硬件压缩管理器或者被实现为硬件和软件的组合。压缩器2022表示压缩引擎的压缩器,用于接收未压缩数据(灰色箭头)、对其进行压缩、并将压缩数据(阴影线箭头)提供至存储器2010以供存储。解压缩器表示压缩引擎的解压缩器,用于接收压缩数据、对其进行解压缩、并将解压缩数据提供至L2缓存2030。

[0261] 如本文所述,压缩器2022可以生成具有CMF信息和CCS信息的报头,以产生通用可消耗的压缩数据。解压缩器2024可以仅通过对数据的访问(诸如,指向数据块的指针)来对压缩数据进行解压缩,以取回被嵌入在压缩数据中的压缩信息并使用该压缩信息来对数据进行解压缩。

[0262] 在一个示例中,压缩器2052可以从系统2000的部件的未压缩数据生成压缩数据作为压缩数据2034以存储在L2缓存2030中。在一个示例中,解压缩器2054可以将来自L2的压缩数据2034解压缩到系统2000的部件。压缩器2052和解压缩器2054可以是系统2000中用于分布式压缩的另一压缩引擎的部分。

[0263] 系统2000的其他部件表示可以通过L2缓存2030被连接的部件的示例。在一个示例中,系统2000包括客户端单元2040,该客户端单元2040包括L1缓存2042。L1缓存2042可以存储客户端单元2040的未压缩数据2044。在一个示例中,客户端单元2040是压缩数据的消耗方和生产方。L1缓存2042可以直接耦合到L2缓存2030,而不是通过压缩引擎耦合到L2缓存

2030。L1缓存2042可以与L2缓存2030交换未压缩数据2032。

[0264] 在一个示例中,系统2000包括客户端单元2060,该客户端单元2060是压缩数据的消耗方或生产方。客户端单元2060表示通过压缩管理器耦合到L2缓存2030的、不具有本地L1缓存的部件。客户端单元2060可以在没有缓存设备的情况下缓冲所产生的数据或缓冲数据以供消耗。

[0265] 在一个示例中,系统2000包括客户端单元2070,该客户端单元2070包括L1缓存2072。L1缓存2072可以存储客户端单元2070的未压缩数据2074。在一个示例中,客户端单元2070是压缩数据的消耗方和生产方。L1缓存2072可以通过压缩引擎耦合到L2缓存2030。压缩器2052和解压缩器2054可以提供从L1缓存2072到L2缓存2030的数据的压缩以及从L2缓存2030到L1缓存2072的压缩数据的解压缩。

[0266] 在一个示例中,系统2000允许绕过压缩引擎。例如,L2缓存2030可以直接向存储器2010提供压缩数据,并且存储器2010可以将压缩数据在不被解压缩的情况下直接传递给L2缓存2030。在一个示例中,L1缓存2072可以直接从L2缓存2030接收未压缩数据,而不需要由压缩引擎解压缩。在一个示例中,客户端单元2060可以直接从L2缓存2030接收未压缩数据,而不需要由压缩引擎解压缩。

[0267] 在一个示例中,存储器2010表示动态随机存取存储器(dynamic random access memory, DRAM)设备。在一个示例中,DRAM设备是高带宽存储器(high bandwidth memory, HBM)设备。在一个示例中,L1缓存2072和L1缓存2042表示静态随机存取存储器(static random access memory, SRAM)设备。在一个示例中,L1缓存表示寄存器堆或寄存器组。在一个示例中,L2缓存2030是SRAM设备。

[0268] 在一个示例中,系统2000包括耦合到L2缓存2030的NIC 2080。在一个示例中,NIC 2080可以与L2缓存2030交换压缩数据。在一个示例中,NIC 2080可以与L2缓存2030交换未压缩数据。在一个示例中,系统2000包括NIC 2080与L2缓存2030之间的压缩管理器。在一个示例中,NIC 2080可以通过与一个或多个其他部件共享的压缩管理器传递数据。

[0269] 系统2000中的压缩管理器被示为位于部件与L2缓存“之间”。在一个示例中,压缩管理器是部件的部分。分布在部件中的压缩管理器可以允许本地缓存的最大化使用,并通过减少结构带宽来减少SOC负载。在一个示例中,压缩管理器包括压缩器/解压缩器核心和用于存储CCS信息的控制平面缓存。压缩管理器的分布可以将压缩/解压缩核心设置在系统2000的不同层处。

[0270] 在一个示例中,使用无状态压缩的每个客户端单元将写事务的CMF设置为充当无状态指示符的特定值。当写请求在途中经由转译后备缓冲器(translation lookaside buffer, TLB)转译而传递到较高级别的缓存(例如,L2缓存2030)时,特殊CMF代码可以由在控制无状态压缩的全局MMIO寄存器中被编程的CMF字段替代。因此,系统可以避免将寄存器直接管线化到所有客户端单元,同时维持一定程度的可编程控制。如果针对页的压缩启用比特被置位,则较高级别的缓存可以基于更新的CMF在驱逐时对数据进行压缩。

[0271] 图21图示片上系统中具有统一压缩的系统的示例。系统2100表示传输数据的系统,该数据可以包括根据系统1600、系统1800或系统1900的示例的压缩数据。在一个示例中,系统2100表示图形单元。在一个示例中,系统2100表示集成的图形单元。

[0272] 在一个示例中,系统2100包括存储器2110以存储数据。该数据可以包括压缩数据。

存储器2110可以存储与压缩数据相关联的CCS信息。在一个示例中,系统2100包括存储器控制器2120,用于管理对存储器2110的访问。在一个示例中,系统2100包括压缩管理器,用于执行要被存储在存储器2110中的数据的压缩和解压缩。在一个示例中,压缩引擎被分布到系统2100的部件或客户端单元中。压缩引擎包括压缩器2132,用于压缩要存储在存储器2110中的数据(阴影线箭头)。压缩引擎包括解压缩器2134,用于对来自存储器2110的数据进行解压缩(灰色箭头)。在一个示例中,系统2100包括用于不需要被压缩或解压缩来与存储器2110交换的压缩数据的绕过路径(长阴影线箭头)。

[0273] 接口2130表示到压缩管理器的接口。在一个示例中,CPU核心2140可以耦合到接口2130。在一个示例中,GPU核心2150可以耦合到接口2130。在一个示例中,加速器2160可以耦合到接口2130。在一个示例中,I/O(input/output,输入/输出)2170可以耦合到接口2130。

[0274] CPU核心2140表示可以消耗未压缩数据的一个或多个CPU核心。GPU核心2150表示可以消耗或生成数据的一个或多个GPU核心。在一个示例中,GPU核心2150可以交换压缩数据(例如,来自缓存)或未压缩数据。加速器2160表示系统2100中可以对数据执行操作的加速器服务。I/O 2170表示到诸如相机或其他传感器之类的部件的连接,或者到其他部件的链路。在一个示例中,I/O 2170可以交换压缩或未压缩数据。

[0275] 在一个示例中,压缩器2132和解压缩器2134的压缩管理器表示用于系统2100的多个或所有客户端单元的共同压缩服务。在一个示例中,压缩管理器包括核心压缩器/解压缩器和控制平面缓存。在一个示例中,压缩管理器向CPU SOC的部件提供压缩服务。部件的示例可以包括媒体、图形、视频或其他部件。

[0276] 图22是生成通用可消耗的压缩数据的示例的流程图。过程2200表示用于生成通用可消耗的压缩数据的过程。在一个示例中,在2202处,客户端单元生成数据。客户端单元可以根据本文中的任何示例。

[0277] 在2204处,客户端单元可以可选地将未压缩的数据存储在L1缓存中。在一个示例中,L1缓存将数据驱逐至L2缓存。在2206处,系统可以确定是否要针对L2或另一本地存储器来对数据进行压缩。

[0278] 如果系统要对数据进行压缩,则在2208处“是”分支处,在一个示例中,系统在2210处利用通用压缩器来应用压缩。通用压缩器可以生成可由客户端部件通用地消耗的压缩数据。在2212处,压缩器可以生成压缩信息并将该压缩信息嵌入在压缩数据中。

[0279] 在一个示例中,如果系统不压缩数据,则在2208“否”分支处,系统可以在2214处将未压缩的数据发送至L2缓存或存储器。在一个示例中,在压缩数据被生成之后,系统可以在2214处将压缩的数据发送至L2缓存或存储器。

[0280] 图23是对通用可消耗的压缩数据解压缩的示例的流程图。过程2300表示用于对通用可消耗的压缩数据进行解压缩的过程。在一个示例中,在2302处,客户端单元请求数据。客户端单元可以根据本文中的任何示例。

[0281] 在2304处,系统可以确定L2缓存或本地存储器中的数据是否以压缩格式被存储。如果L2缓存或存储器中的数据被压缩存储,则在2306处“是”分支,在一个示例中,系统在2308处访问嵌入有压缩信息的压缩数据。在2310,通用解压缩器基于被存储在压缩数据中的压缩信息来应用解压缩。

[0282] 在一个示例中,如果数据没有被压缩,则在2306处“否”分支,系统可以在2312处将

未压缩的数据提供至L1缓存。在一个示例中,在压缩数据被解压缩之后,系统可以在2312处将未压缩数据提供至L1缓存。

[0283] 一般而言,根据本文中的描述,在一个示例中,一种图形处理器包括:存储器控制器,用于耦合到用于为多个图形部件存储压缩数据的共享存储器,其中压缩数据的块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面CCS信息的报头;以及压缩引擎,用于压缩要存储在共享存储器中的数据,以及对从共享存储器读取的数据进行解压缩,其中压缩要存储的数据包括生成报头,并且其中对读取的数据进行解压缩包括从报头标识压缩格式。

[0284] 在图形处理器的一个示例中,压缩格式信息包括用于指示压缩类型的查找表的值。根据图形处理器的任一先前示例,在一个示例中,压缩引擎包括压缩器/解压缩器核心以及用于存储CCS信息的控制平面缓存。根据图形处理器的任一先前示例,在一个示例中,压缩引擎用于从多个图形部件中的一个图形部件接收要被压缩的数据,包括接收到的数据要以无状态格式被压缩的指示符,并且响应于该指示符,压缩引擎用于生成具有压缩信息的报头。根据图形处理器的任一先前示例,在一个示例中,压缩数据的块具有统一存储器布局。根据图形处理器的任一先前示例,在一个示例中,统一存储器布局包括行为主格式布局。根据图形处理器的任一先前示例,在一个示例中,图形处理器包括:图形部件之间的通信结构,其中通信结构用于基于块压缩比支持多个数据传输大小。根据图形处理器的任一先前示例,在一个示例中,通信结构支持压缩数据的传输和未压缩数据的传输。根据图形处理器的任一先前示例,在一个示例中,压缩引擎包括第一压缩引擎,图形处理器包括:第二压缩引擎,其中第一压缩引擎和第二压缩引擎位于图形处理器的不同层处。

[0285] 一般而言,根据本文中的描述,在一个示例中,一种计算机系统包括:中央处理单元(CPU),用于执行一般计算;图形单元,耦合到CPU,图形单元包括:存储器控制器,用于耦合到用于为多个图形部件存储压缩数据的共享存储器,其中压缩数据的块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面CCS信息的报头;以及压缩引擎,用于压缩要存储在共享存储器中的数据,以及对从共享存储器读取的数据进行解压缩,其中压缩要存储的数据包括生成报头,并且其中对读取的数据进行解压缩包括从报头标识压缩格式。

[0286] 在计算机系统的一个示例中,压缩格式信息包括用于指示压缩类型的查找表的值。根据计算机系统的任一先前示例,在一个示例中,压缩引擎用于从多个图形客户端单元中的一个图形客户端单元接收要被压缩的数据,包括接收到的数据要以无状态格式被压缩的指示符,并且响应于该指示符,压缩引擎用于生成具有压缩格式信息和CCS信息的报头。根据计算机系统的任一先前示例,在一个示例中,压缩数据的块具有统一存储器布局,其中统一存储器布局包括行为主格式布局。根据计算机系统的任一先前示例,在一个示例中,计算机系统包括:硬件管线之间的通信结构,其中该通信结构用于基于块压缩比支持多个数据传输大小。根据计算机系统的任一先前示例,在一个示例中,通信结构支持压缩数据的传输和未压缩数据的传输。根据计算机系统的任一先前示例,在一个示例中,多个硬件管线包括以下各项中的一项或多项:媒体管线;颜色管线;显示管线;视频捕捉管线;或人工智能(AI)加速器管线。根据计算机系统的任一先前示例,在一个示例中,压缩引擎包括第一压缩引擎,并且进一步包括:第二压缩引擎,其中第一压缩引擎和第二压缩引擎位于图形单元的

不同层处。根据计算机系统的任一先前示例,在一个示例中,计算机系统包括:主系统存储器,与图形单元的共享存储器分开;其中,压缩引擎用于提供主系统存储器中的数据的压缩和解压缩,以供CPU访问。

[0287] 一般而言,根据本文中的描述,在一个示例中,一种用于压缩的方法包括:将压缩数据在针对多个图形部件的共享存储器中存储为压缩数据的块,压缩数据的这些块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面 (CCS) 信息的报头;压缩要存储在共享存储器中的数据,包括生成报头;以及对从共享存储器读取的数据进行解压缩,包括从报头标识压缩格式。

[0288] 在方法的一个示例中,方法包括:使用通信结构在图形部件之间传送压缩数据的块,该通信结构基于块压缩比支持多个数据传输大小。根据方法的任一先前示例,在一个示例中,通信结构支持压缩数据的传输和未压缩数据的传输。

[0289] 一般而言,根据本文中的描述,在一个示例中,一种图形处理器包括:用于多个图形客户端单元的硬件管线;共享存储器,用于存储多个硬件管线的压缩数据,其中压缩数据块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面 (CCS) 信息的报头;以及压缩引擎,用于压缩要存储在共享存储器中的数据,以及对从共享存储器读取的数据进行解压缩,其中压缩要存储的数据包括生成报头,并且其中对读取的数据进行解压缩包括从报头标识压缩格式。

[0290] 在先前段落的图形处理器的一个示例中,压缩格式信息包括用于指示压缩类型的查找表的值。根据图形处理器的任一示例,在一个示例中,压缩引擎包括压缩器/解压缩器核心以及用于存储CCS信息的控制平面缓存。根据图形处理器的任一示例,在一个示例中,压缩引擎用于从多个图形部件中的一个图形部件接收要被压缩的数据,包括接收到的数据要以无状态格式被压缩的指示符,并且响应于该指示符,压缩引擎用于生成具有压缩信息的报头。根据图形处理器的任一示例,在一个示例中,压缩数据的块具有统一存储器布局。根据图形处理器的任一示例,在一个示例中,统一存储器布局包括行为主格式布局。根据图形处理器的任一示例,在一个示例中,图形处理器包括:硬件管线之间的通信结构,其中该通信结构用于基于块压缩比支持多个数据传输大小。根据图形处理器的任一示例,在一个示例中,通信结构支持压缩数据的传输和未压缩数据的传输。根据图形处理器的任一示例,在一个示例中,压缩引擎包括第一压缩引擎,并且图形处理器包括:第二压缩引擎,其中第一压缩引擎和第二压缩引擎位于图形处理器的不同层处。

[0291] 一般而言,根据本文中的描述,在一个示例中,一种计算机系统包括:中央处理单元 (CPU),用于执行一般计算;图形单元,包括:用于多个图形客户端单元的硬件管线;以及共享存储器,用于为多个硬件管线存储压缩数据,其中压缩数据的块具有共同存储器占用空间并且包括具有压缩格式信息和压缩控制表面 (CCS) 信息的报头;以及耦合到图形单元的压缩引擎,压缩引擎用于压缩要存储在共享存储器中的数据,以及对从共享存储器读取的数据进行解压缩,其中压缩要存储的数据包括生成报头,并且其中对读取的数据进行解压缩包括从报头标识压缩格式。

[0292] 在计算机系统的一个示例中,压缩格式信息包括用于指示压缩类型的查找表的值。根据计算机系统的任一先前示例,在一个示例中,压缩引擎用于从多个图形客户端单元中的一个图形客户端单元接收要被压缩的数据,包括接收到的数据要以无状态格式被压缩

的指示符,并且响应于该指示符,压缩引擎用于生成具有压缩格式信息和CCS信息的报头。根据计算机系统的任一先前示例,在一个示例中,压缩数据的块具有统一存储器布局,其中统一存储器布局包括行为主格式布局。根据计算机系统的任一先前示例,在一个示例中,计算机系统包括:硬件管线之间的通信结构,其中该通信结构用于基于块压缩比支持多个数据传输大小。根据计算机系统的任一先前示例,在一个示例中,通信结构支持压缩数据的传输和未压缩数据的传输。根据计算机系统的任一先前示例,在一个示例中,多个硬件管线包括以下各项中的一项或多项:媒体管线;颜色管线;显示管线;视频捕捉管线;或人工智能(AI)加速器管线。根据计算机系统的任一先前示例,在一个示例中,计算机系统包括:主系统存储器,与图形单元的共享存储器分开;其中,压缩引擎用于提供主系统存储器中的数据的压缩和解压缩,以供CPU访问。

[0293] 本文中图示的流程图提供各种过程动作的序列的示例。流程图可以指示要由软件或固件例程执行的操作、以及物理操作。流程图可以说明有限状态机(finite state machine,FSM)的状态的实现方式的示例,有限状态机可以以硬件和/或软件来实现。尽管以特定顺序或次序示出,但是除非另行指出,否则这些动作的次序可被更改。由此,所图示的图应当仅被理解为示例,并且过程可按不同次序来执行,并且一些动作可被并行地执行。此外,可以省略一个或多个动作;由此,并非所有实现方式都将执行所有动作。

[0294] 在本文中描述的各种操作或功能的程度上,它们可被描述或定义为软件代码、指令、配置和/或数据。内容可以是可直接执行的(“对象”或“可执行”形式)、源代码、或差异代码(“Δ”或“补丁”代码)。本文中所描述的内容中的软件内容可以经由具有存储于其上的内容的制品来提供,或者可经由操作通信接口以经由此通信接口来发送数据的方法来提供。机器可读存储介质可以使机器执行所描述的功能或操作,并且包括以机器(例如,计算设备、电子系统等)可访问的形式存储信息的任何机制,诸如可记录/不可记录的可记录介质(例如,只读存储器(ROM)、随机存取存储器(RAM)、磁盘存储介质、光存储介质、闪存设备等)。通信接口包括对硬连线的、无线的、光学介质等中的任一者提供接口以向另一设备通信的任何机制,诸如,存储器总线接口、处理器总线接口、网际连接、盘控制器等。通信接口可通过提供配置参数和/或发送信号来配置,以预备通信接口提供描述软件内容的数据信号。可经由发送至该通信接口的一个或多个命令或信号来访问该通信接口。

[0295] 本文中所描述的各种部件可以是用于执行所描述的操作或功能的装置。本文中所描述的每个部件包括软件、硬件、或者软件和硬件的组合。部件可以被实现为软件模块、硬件模块、专用硬件(例如,专用硬件、专用集成电路(application specific integrated circuit,ASIC)、数字信号处理器(digital signal processor,DSP)等)、嵌入式控制器、硬连线电路系统等。

[0296] 除了本文中所描述的内容之外,还可以对本发明的所公开的内容和实现方式进行各种修改而不背离它们的范围。因此,本文中的说明和示例应当被解释成说明性的,而非限制性的。本发明的范围应当仅通过参照所附权利要求书来界定。

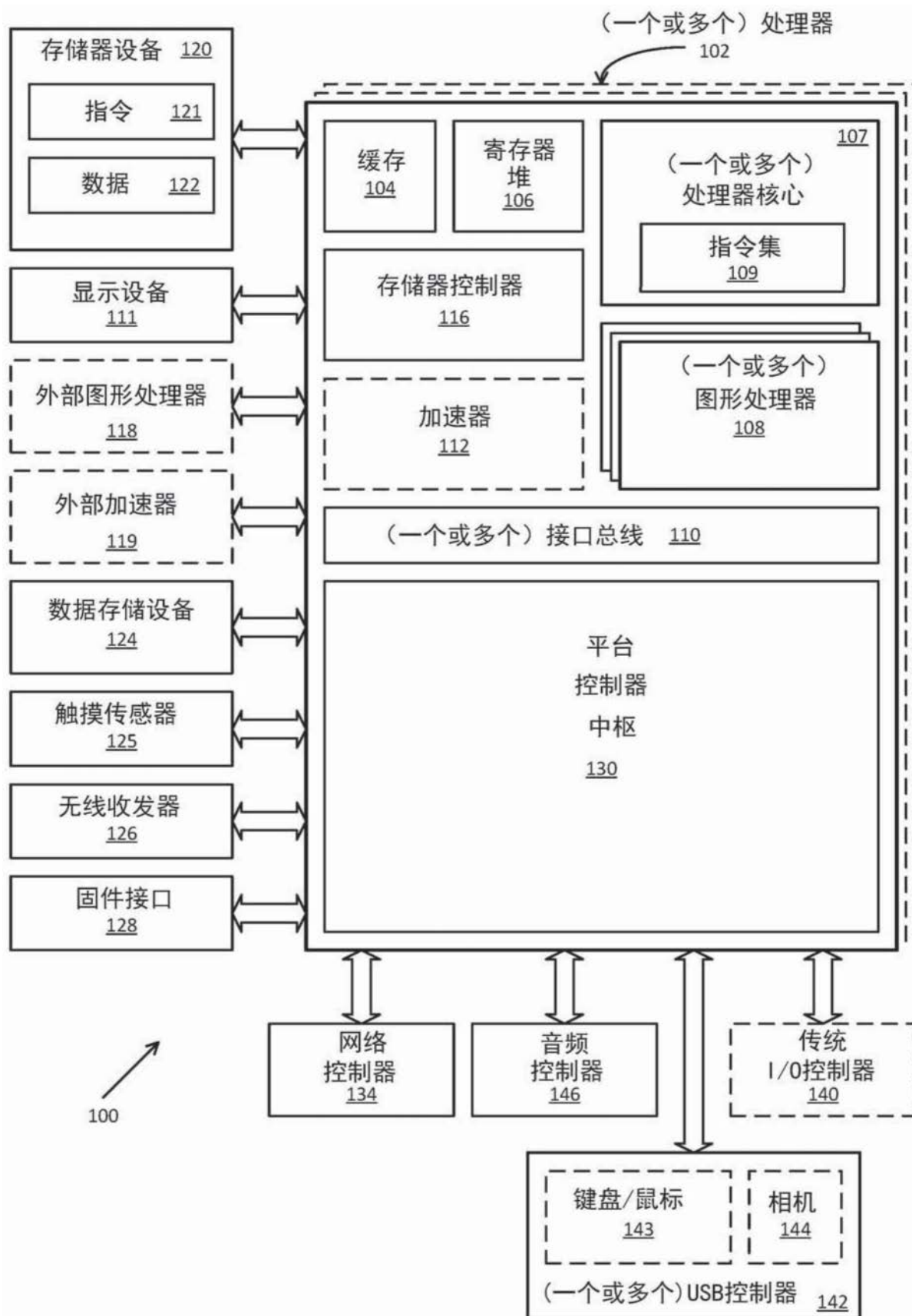


图1

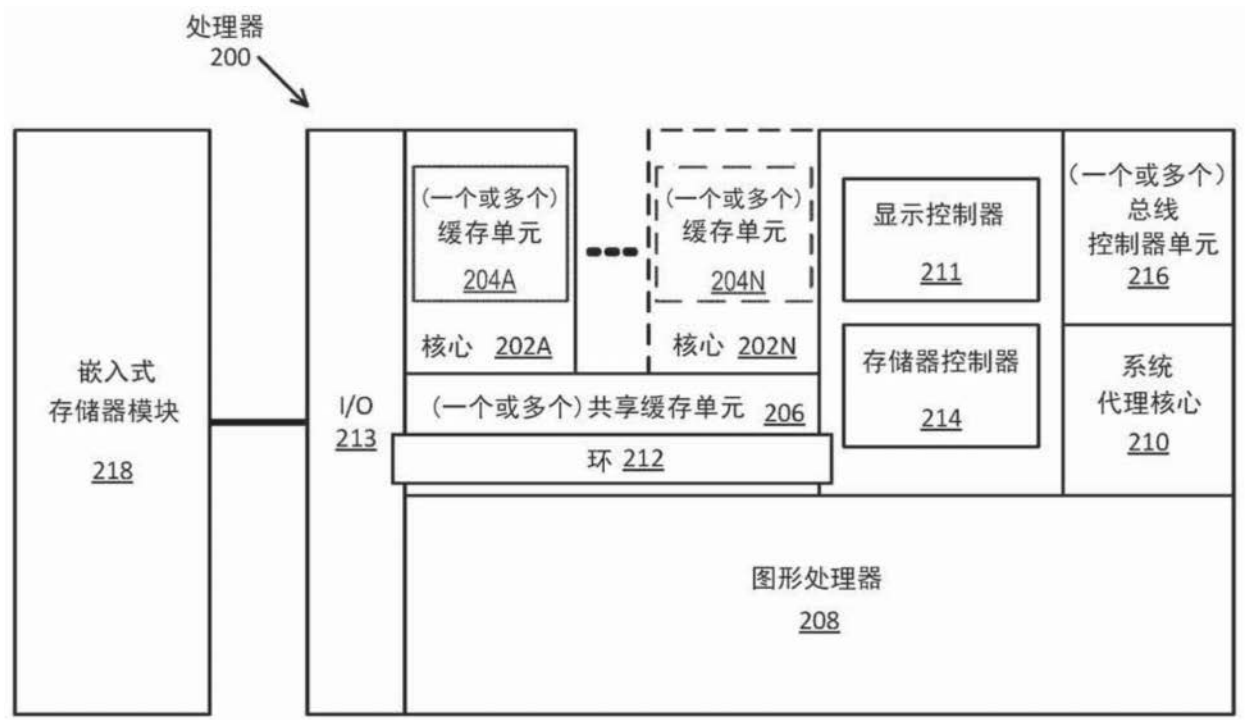


图2A

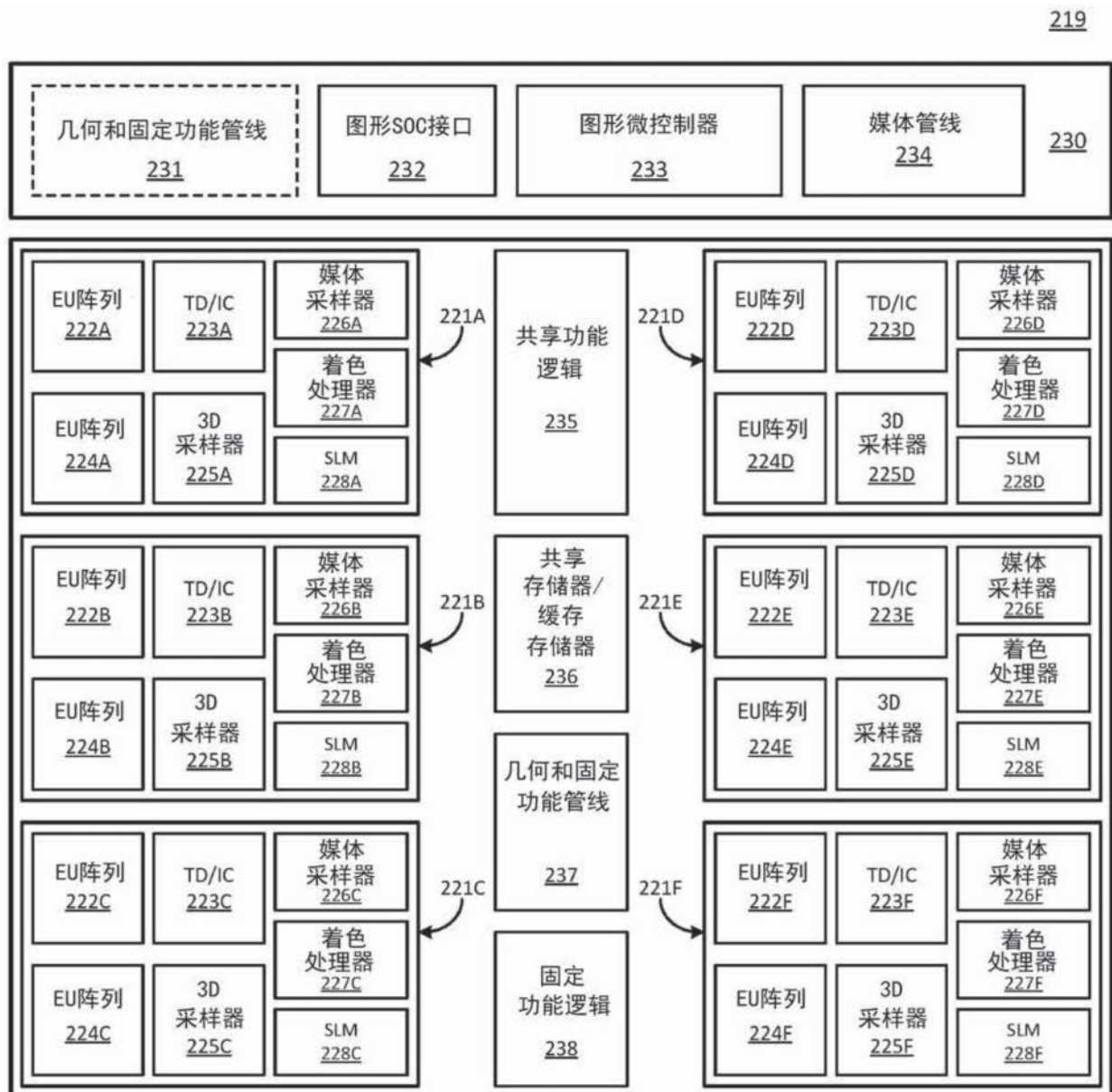


图2B

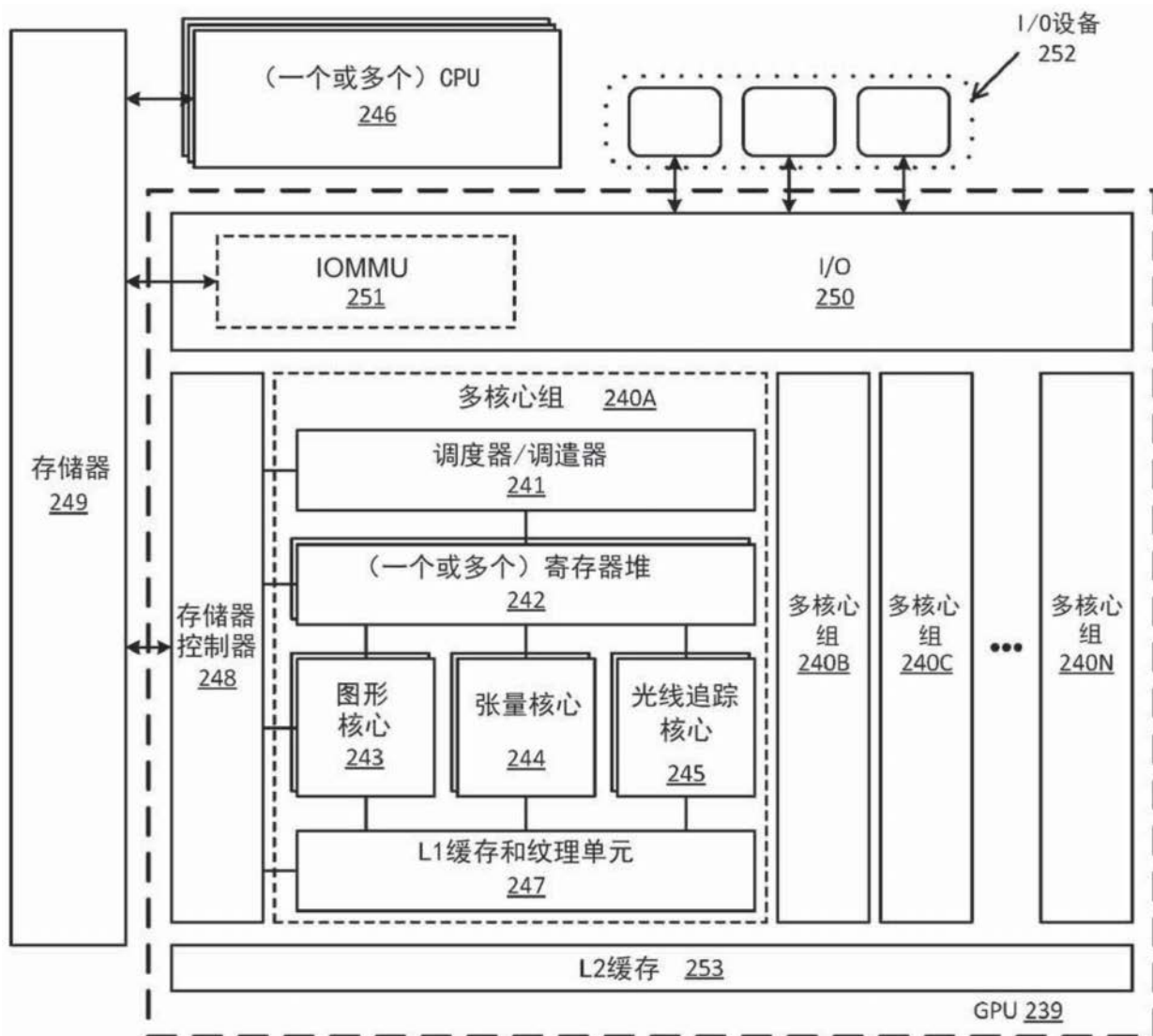


图2C

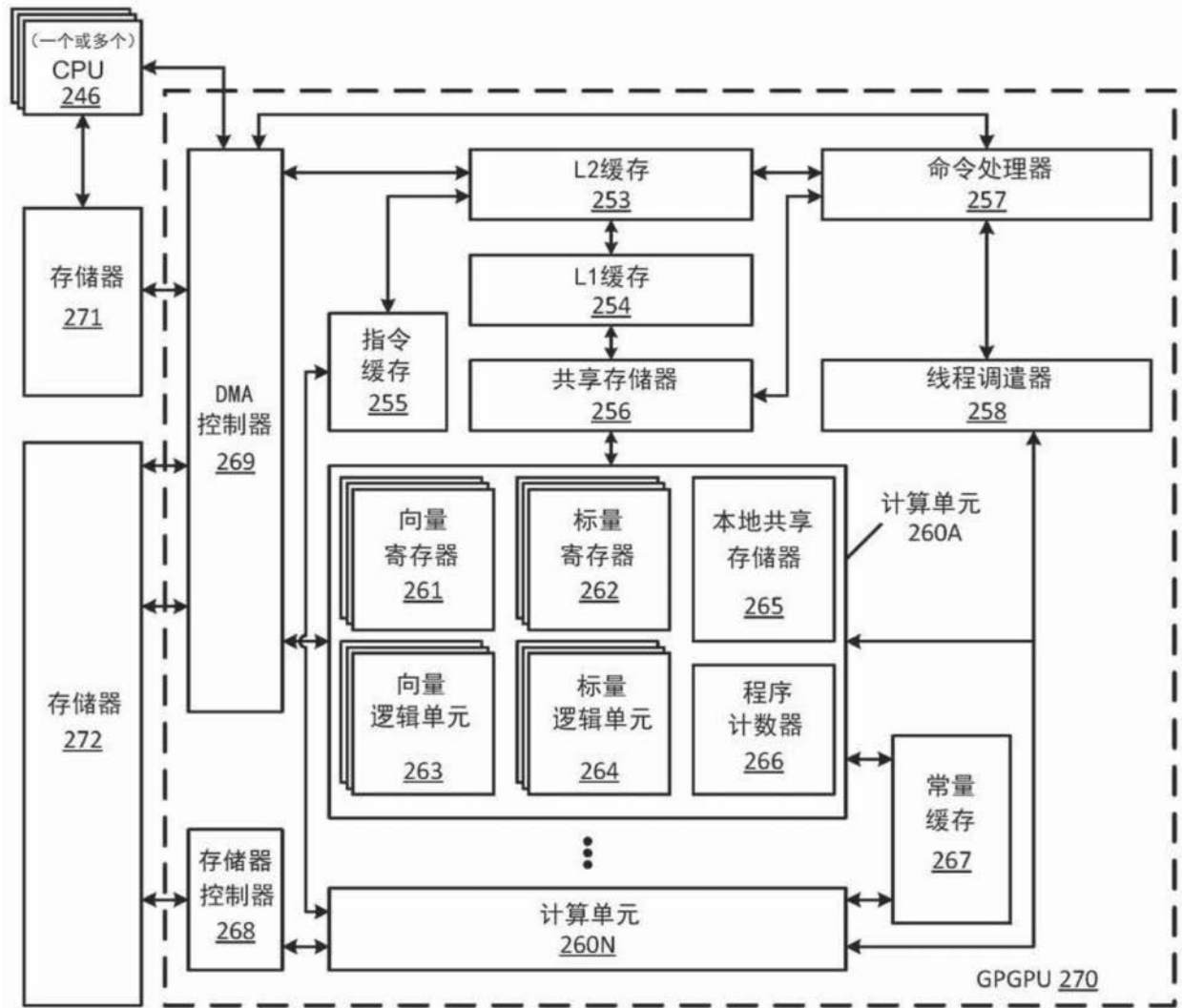


图2D

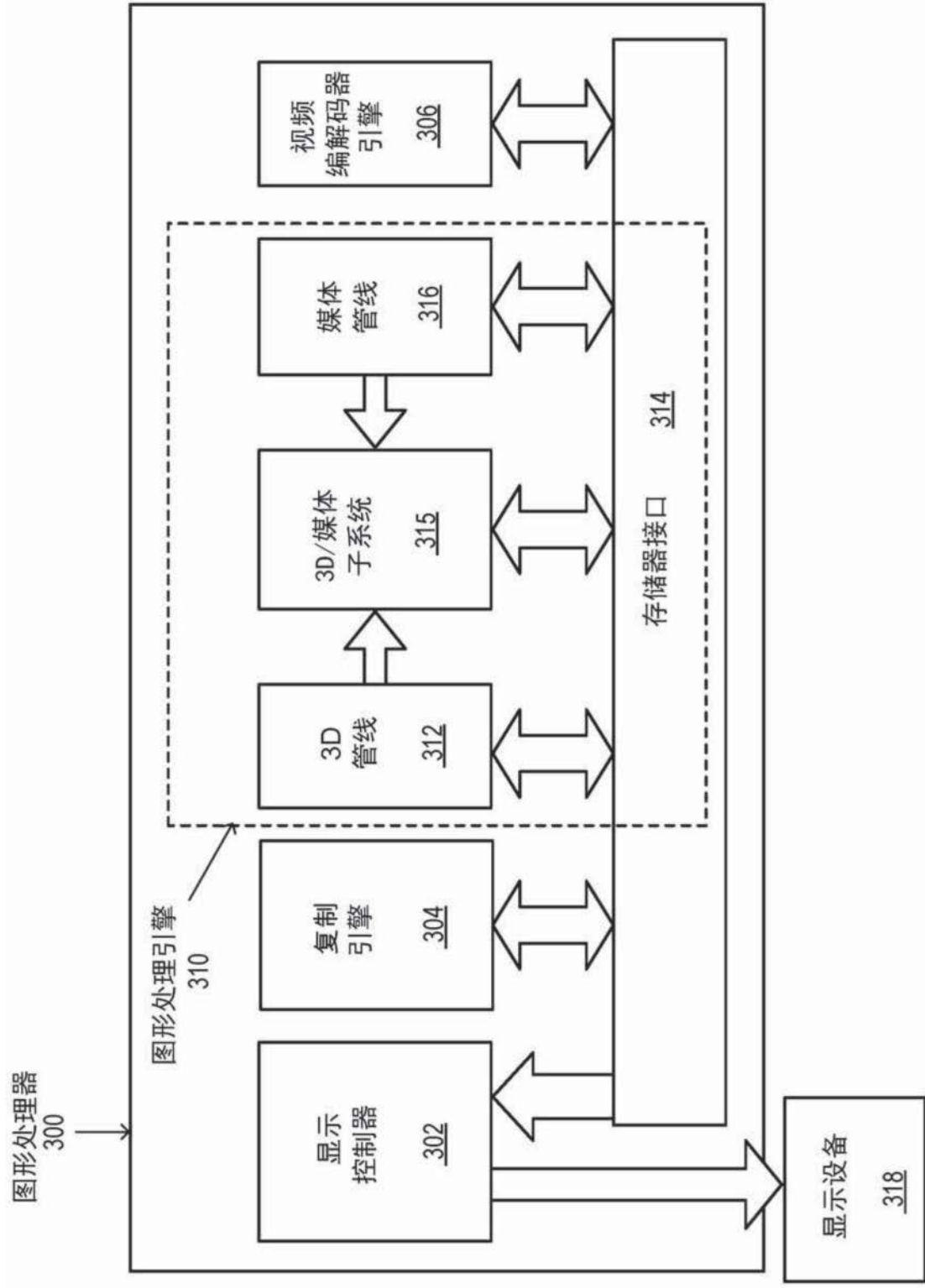


图3A

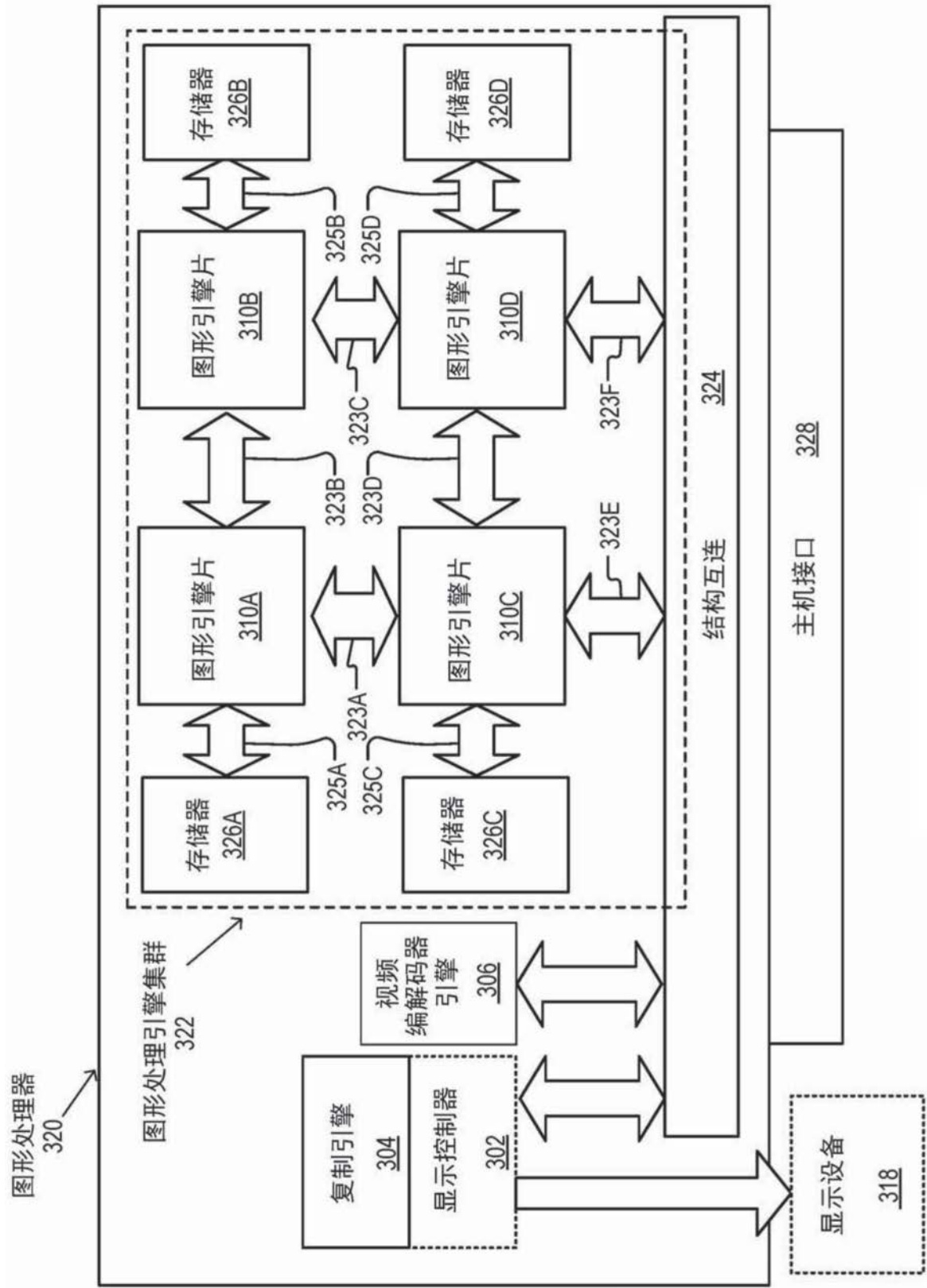


图3B

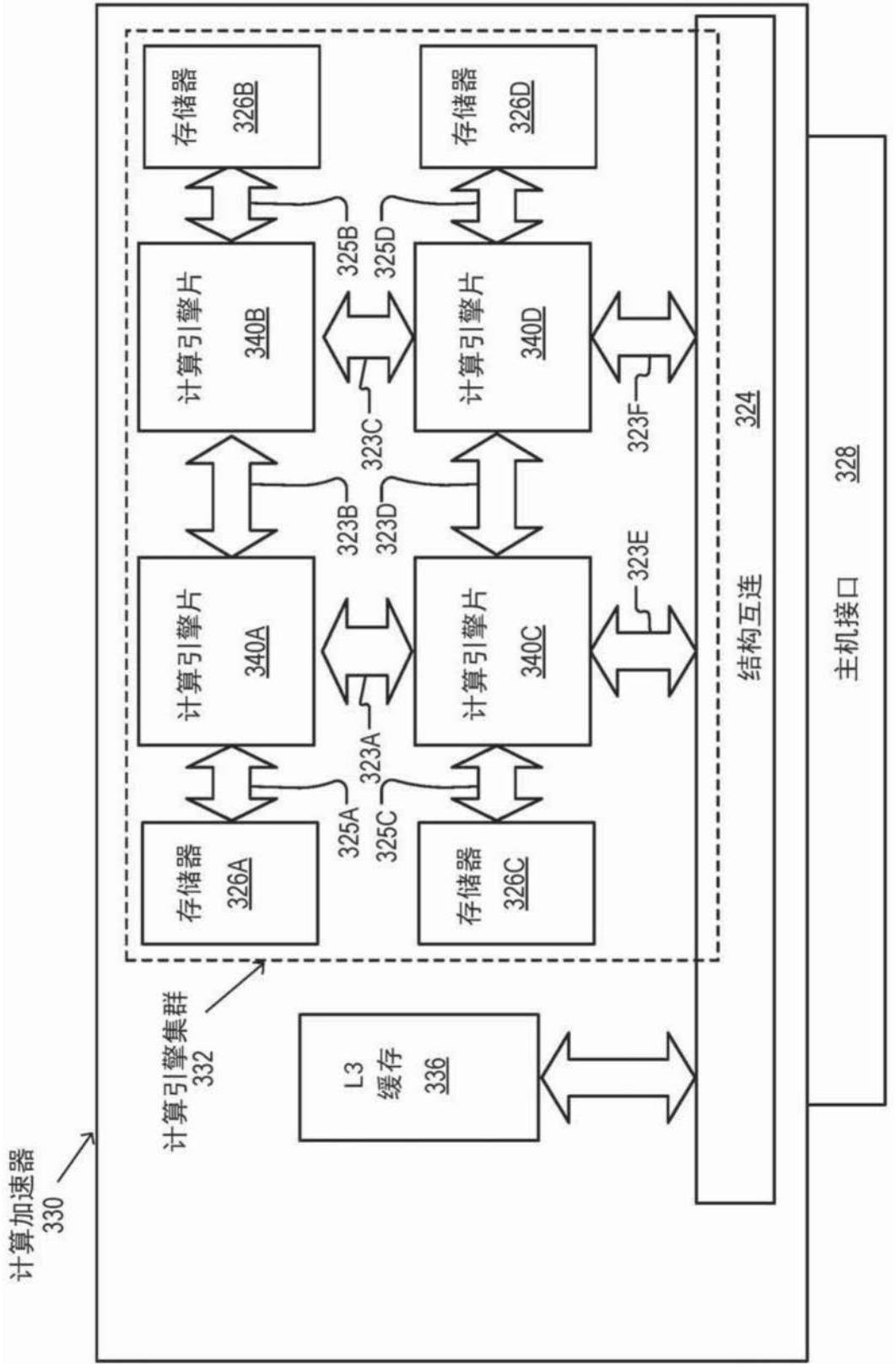


图3C

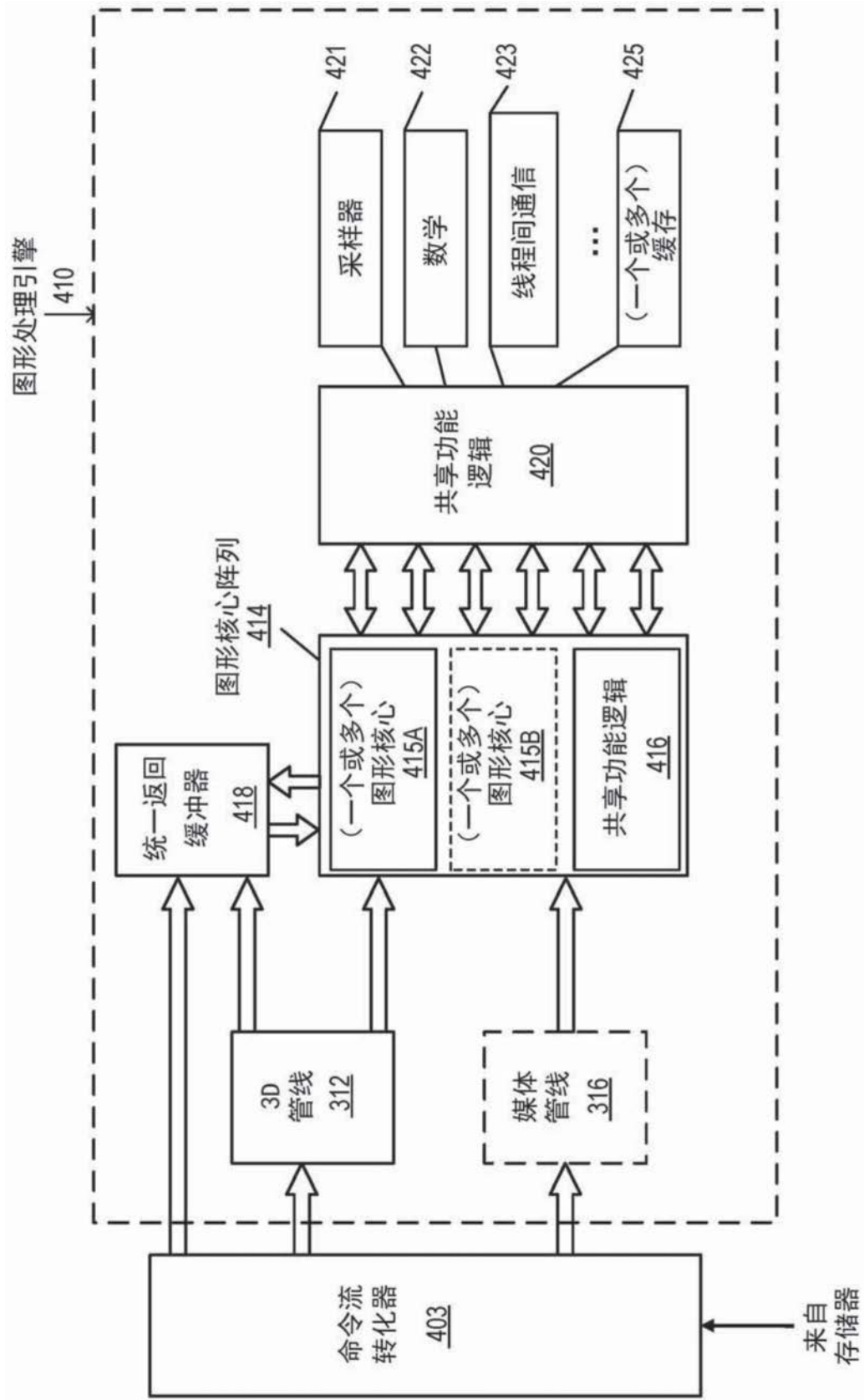


图4

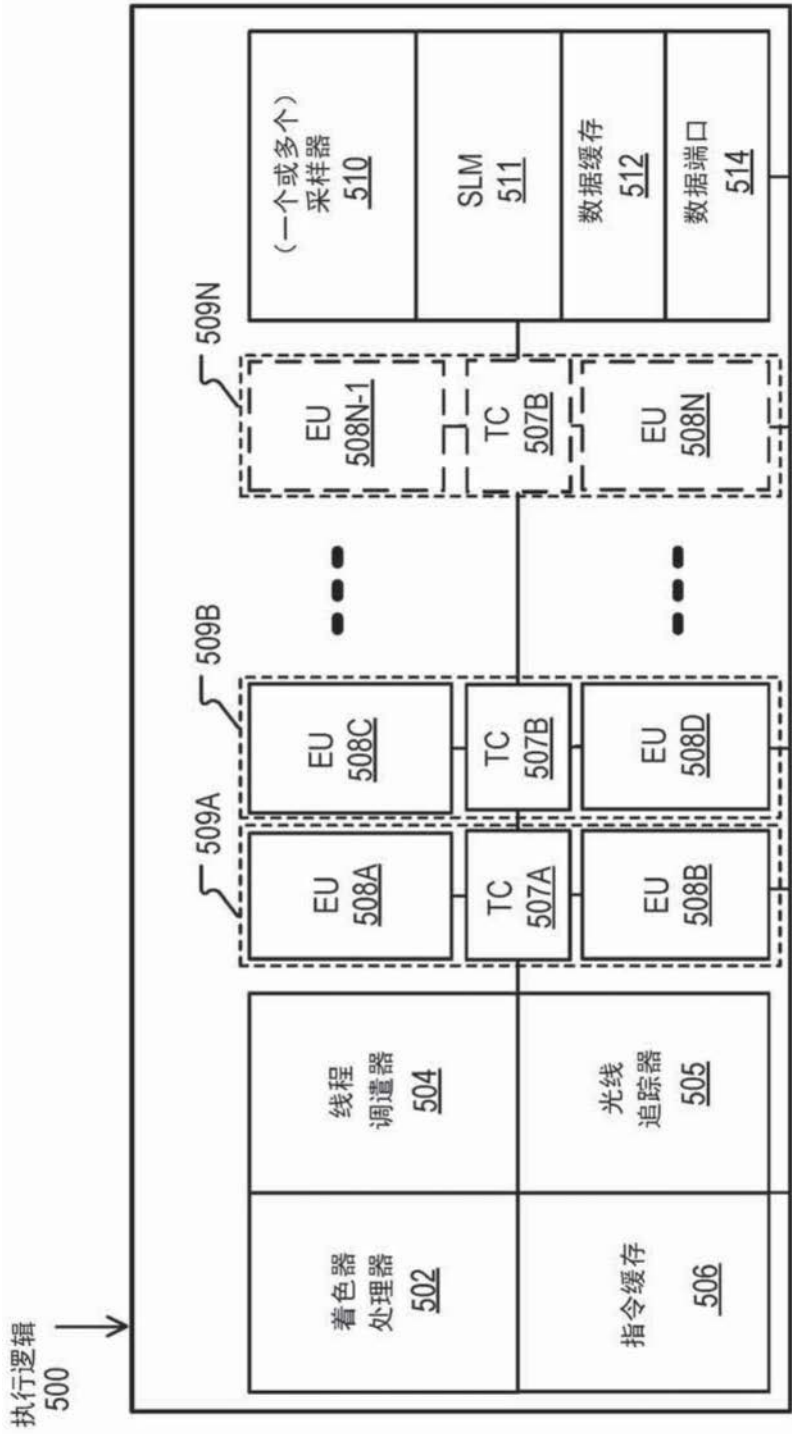


图5A

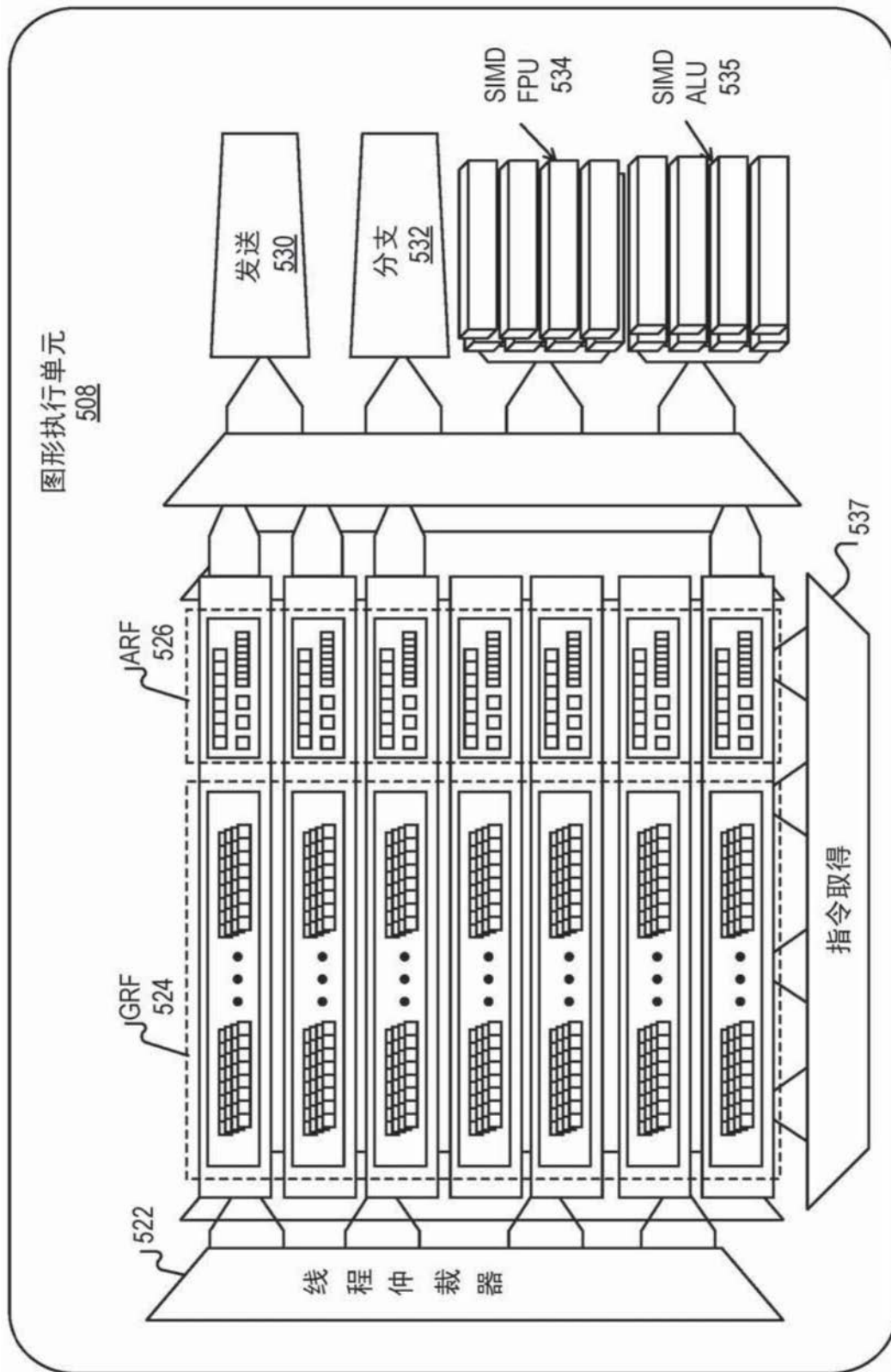


图5B

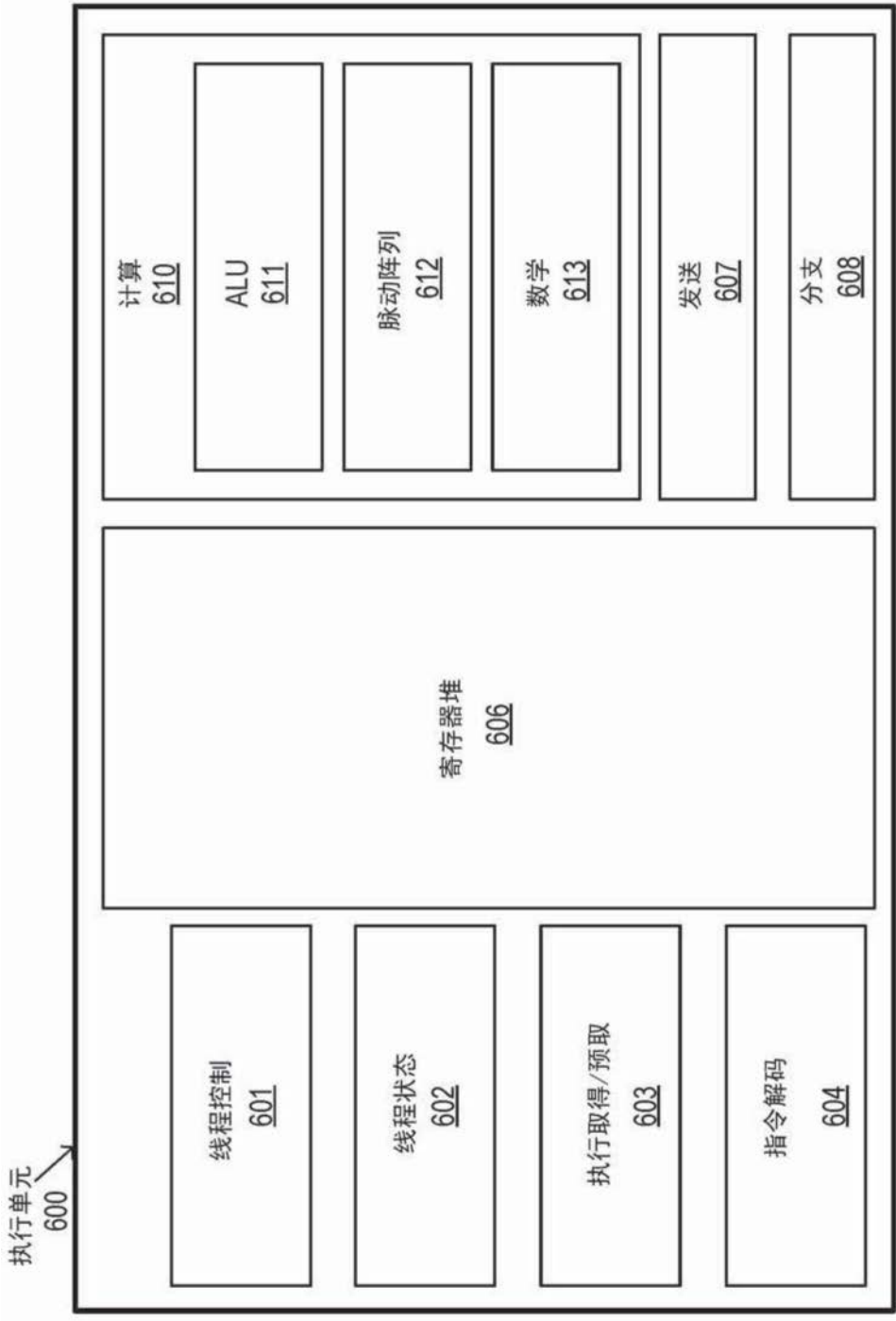


图6

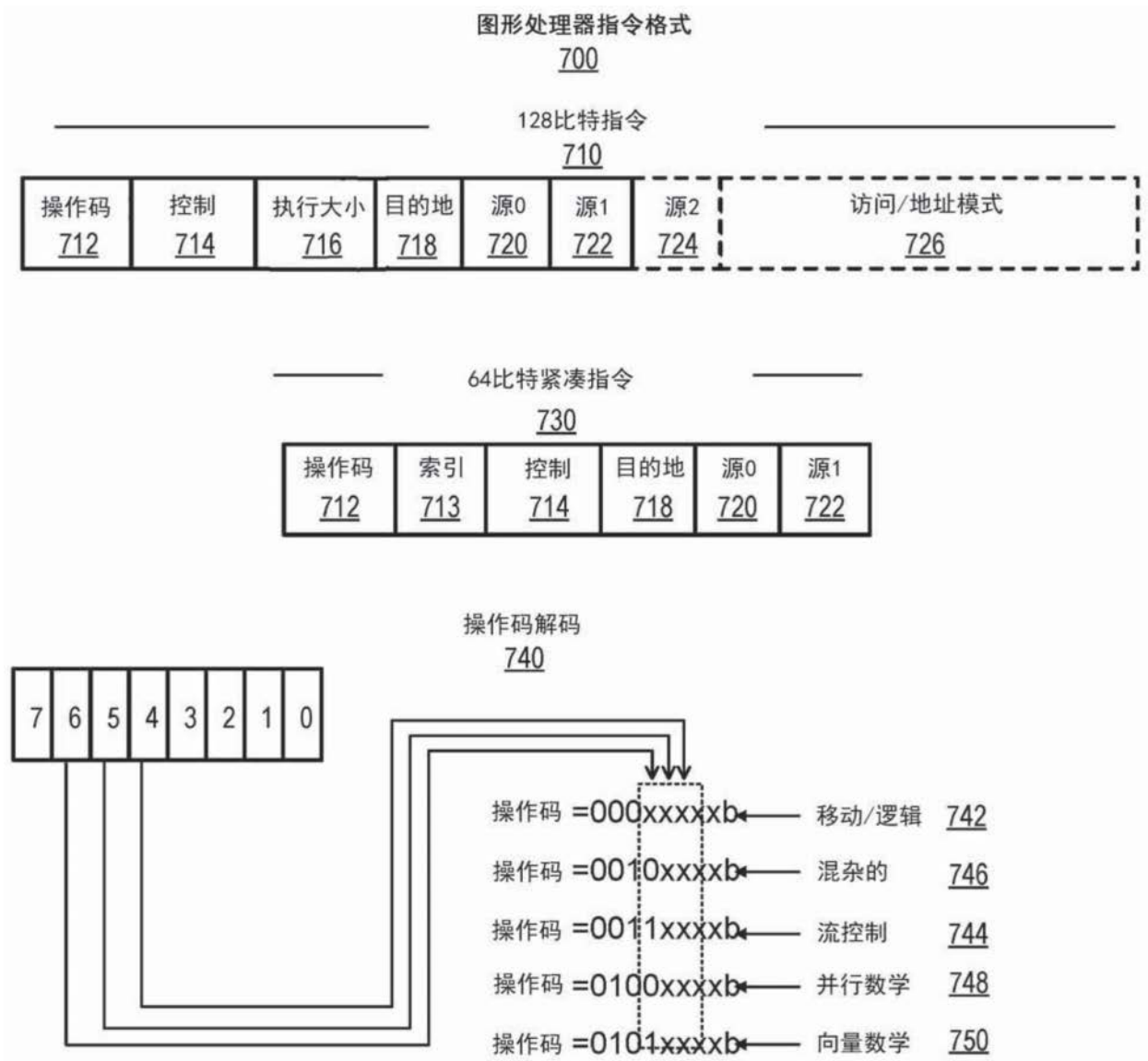


图7

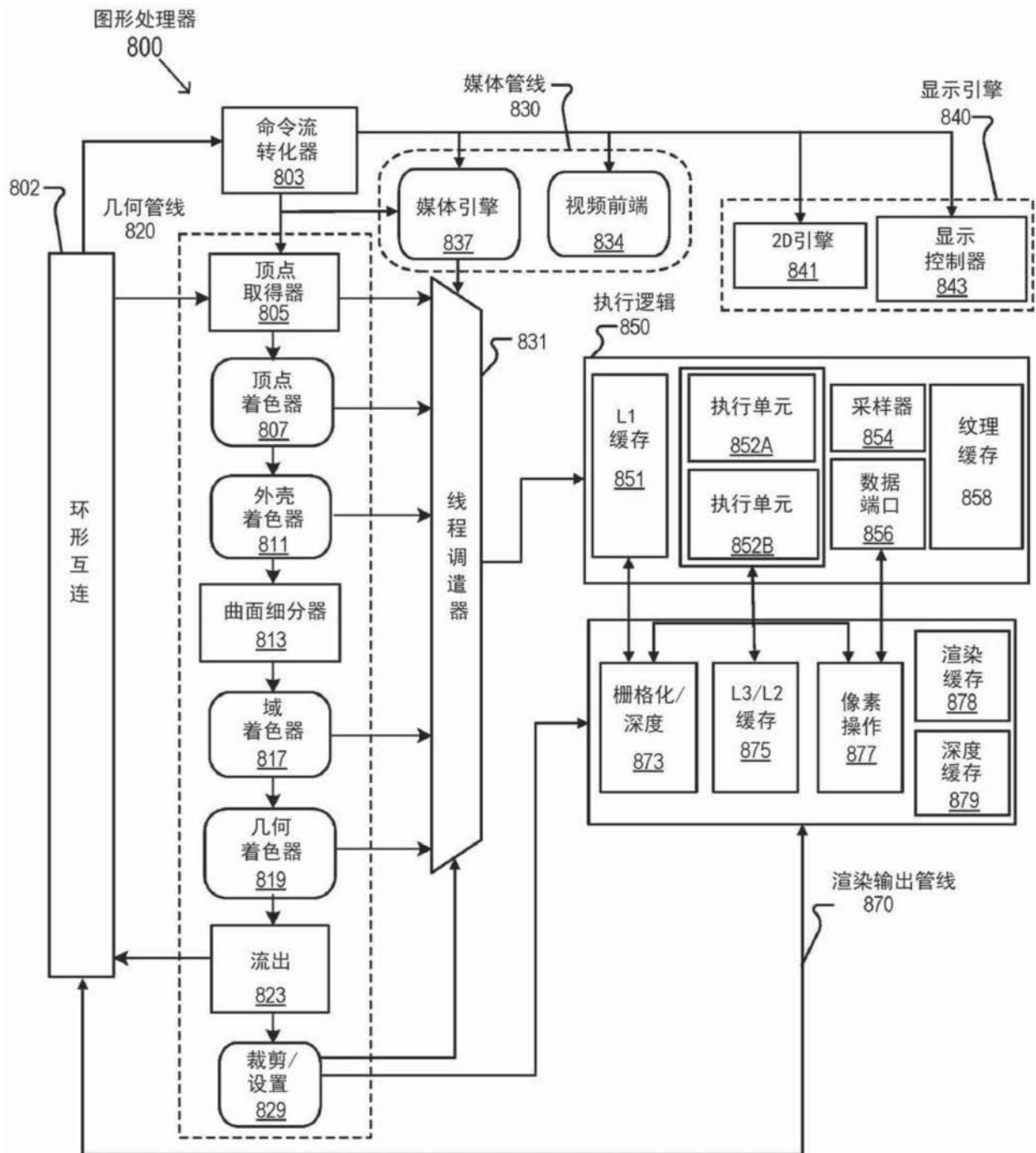


图8



图9A

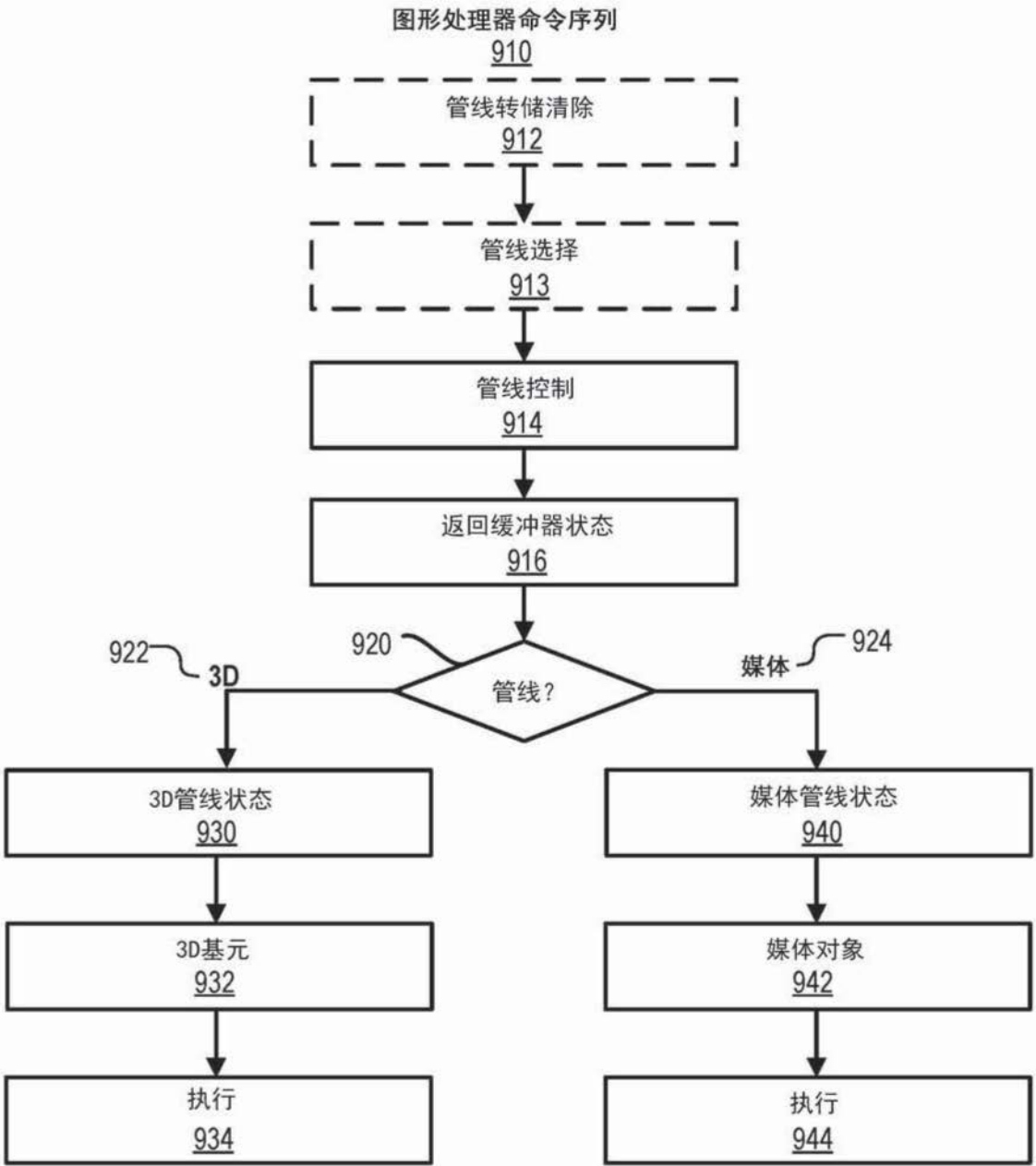


图9B

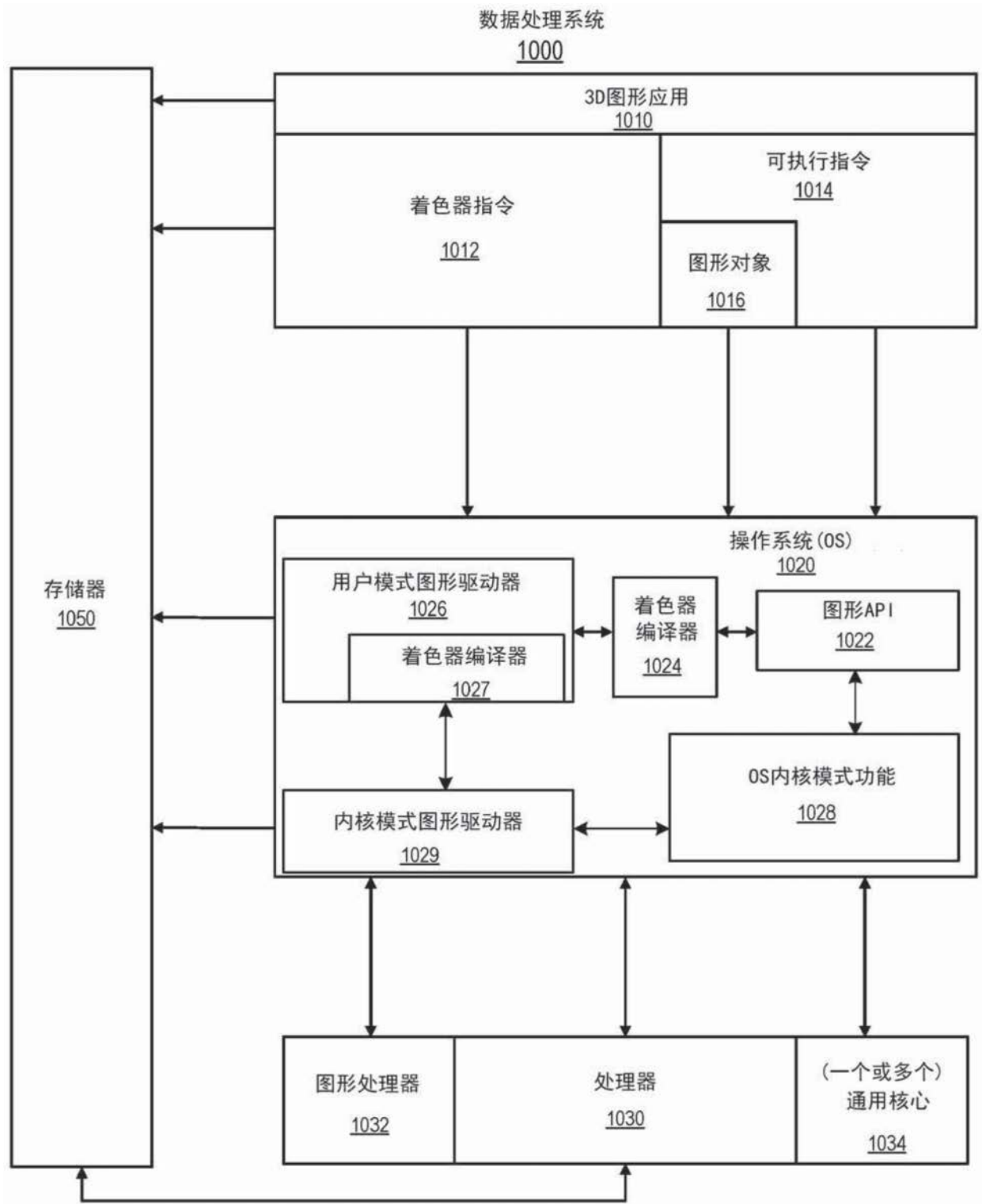


图10

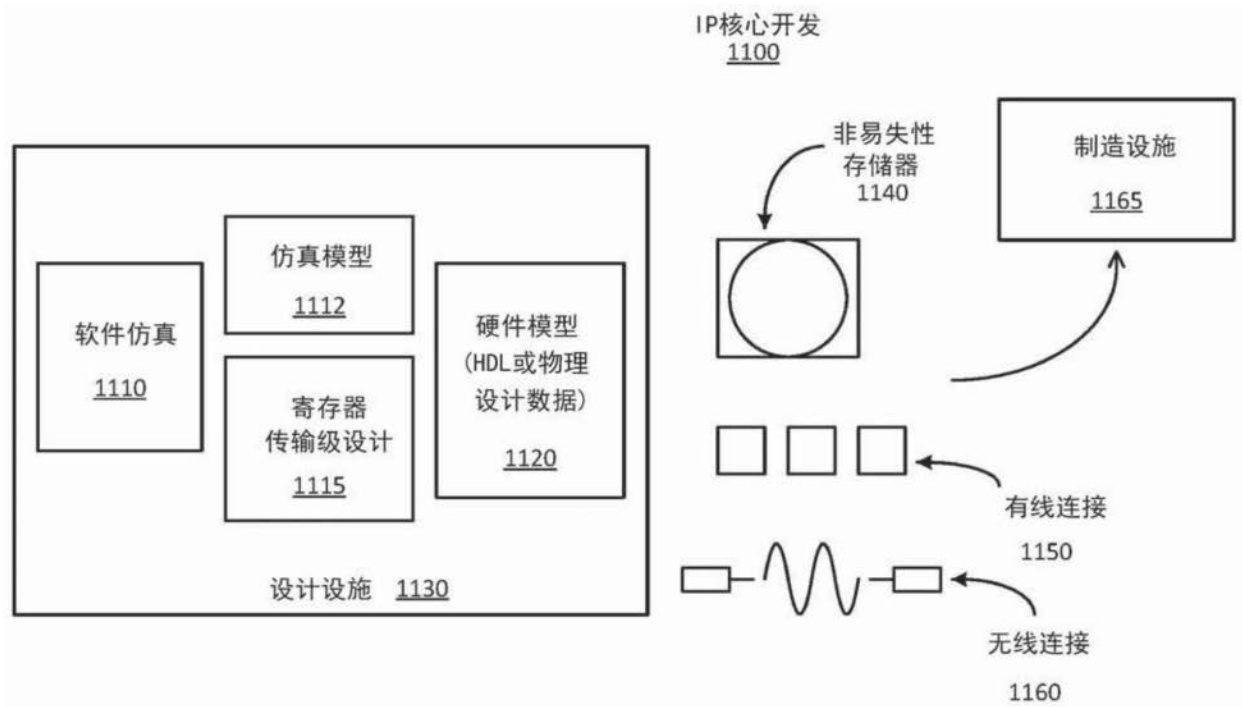


图11A

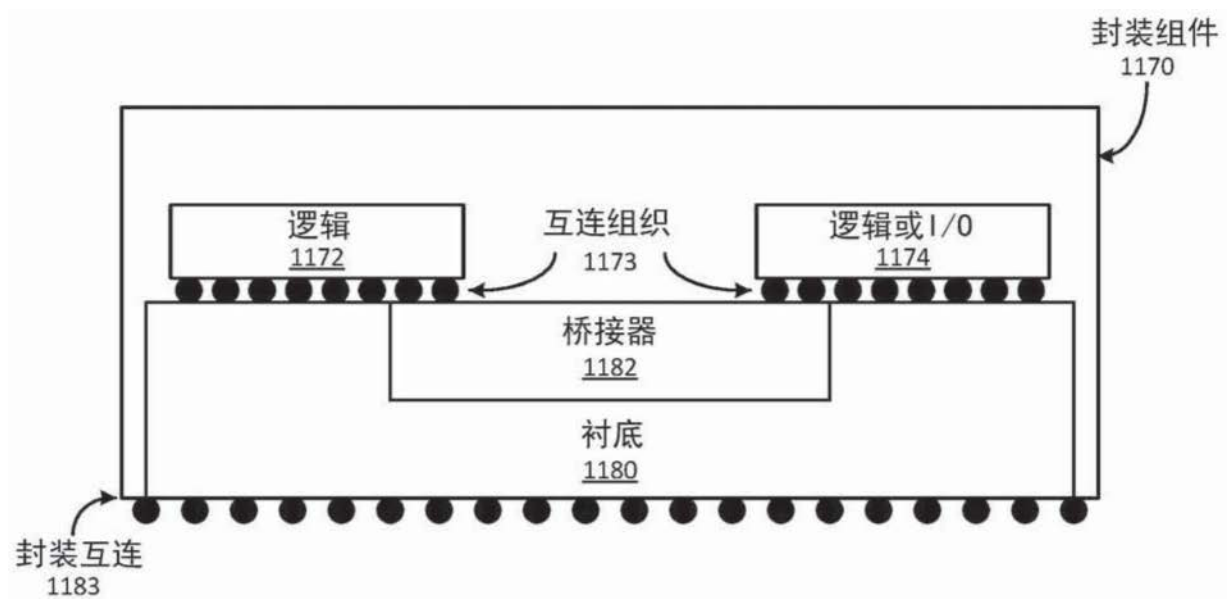


图11B

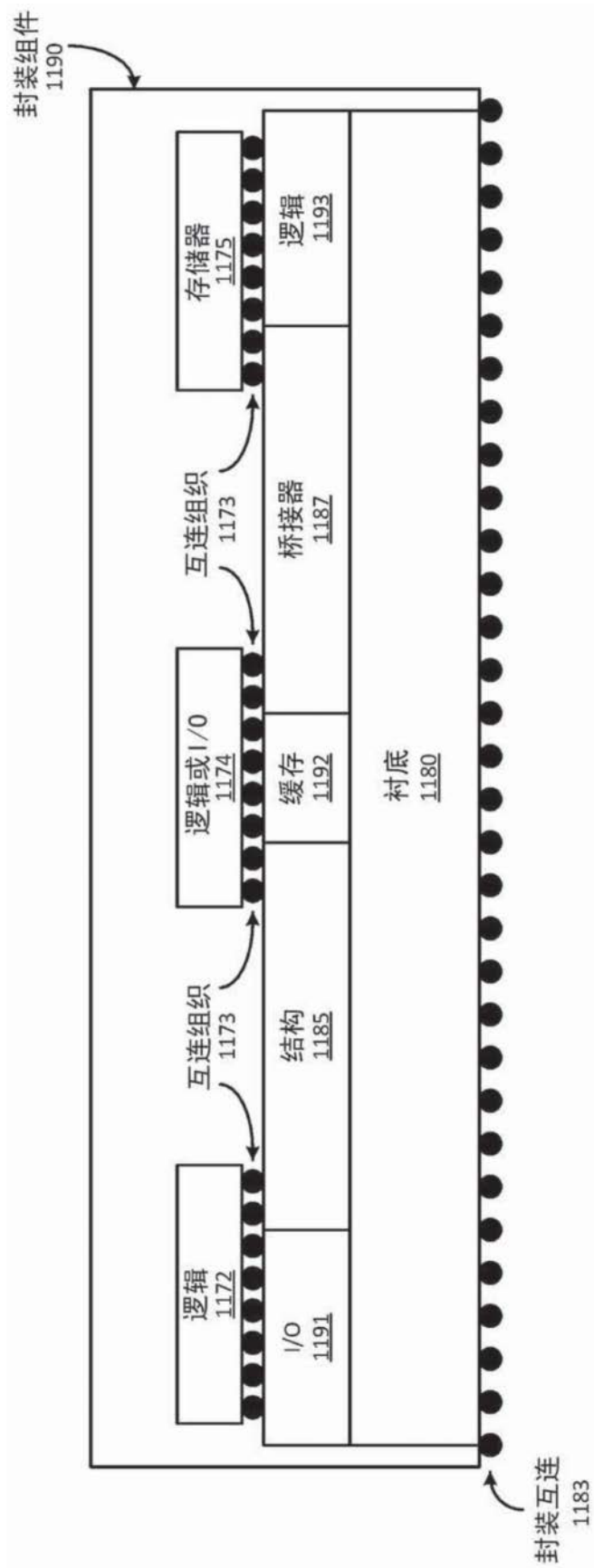


图11C

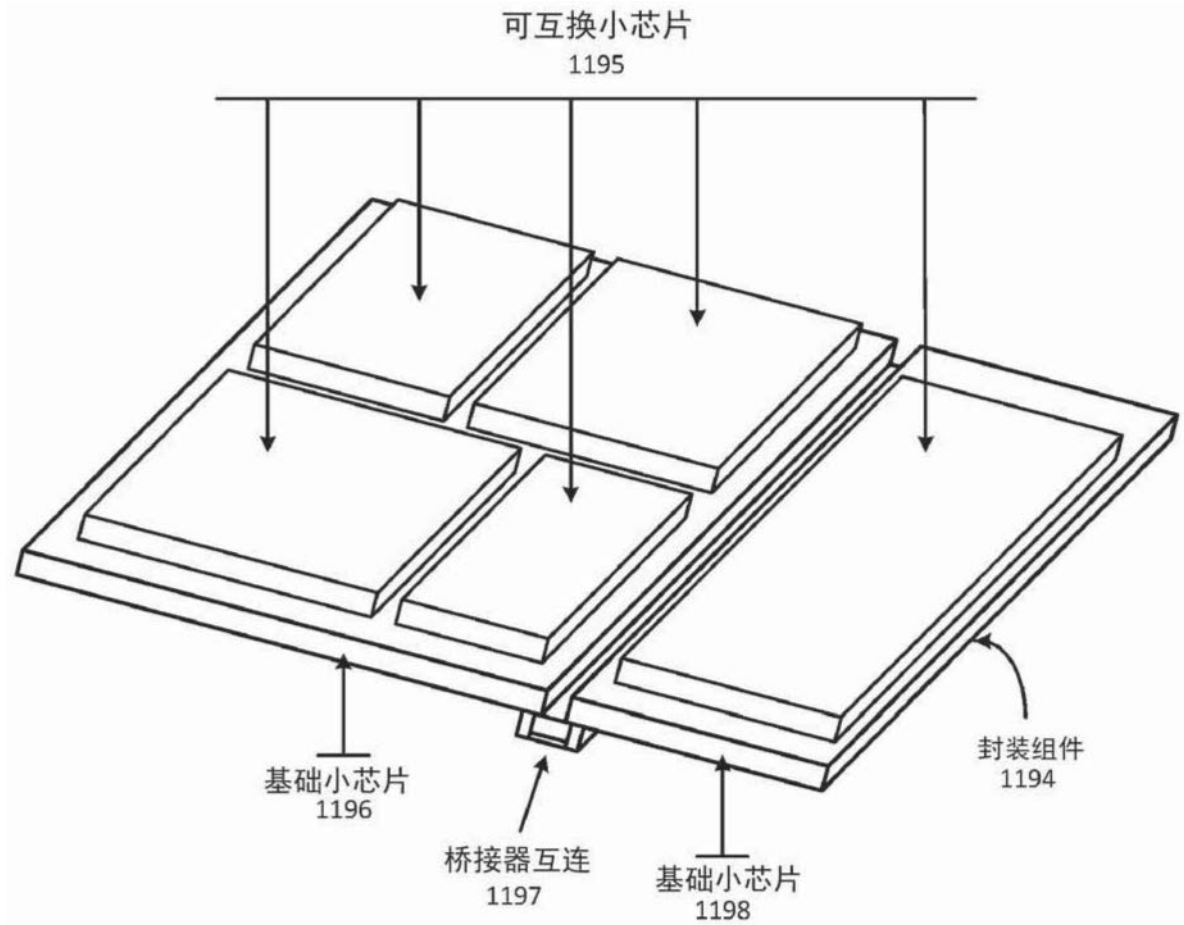


图11D

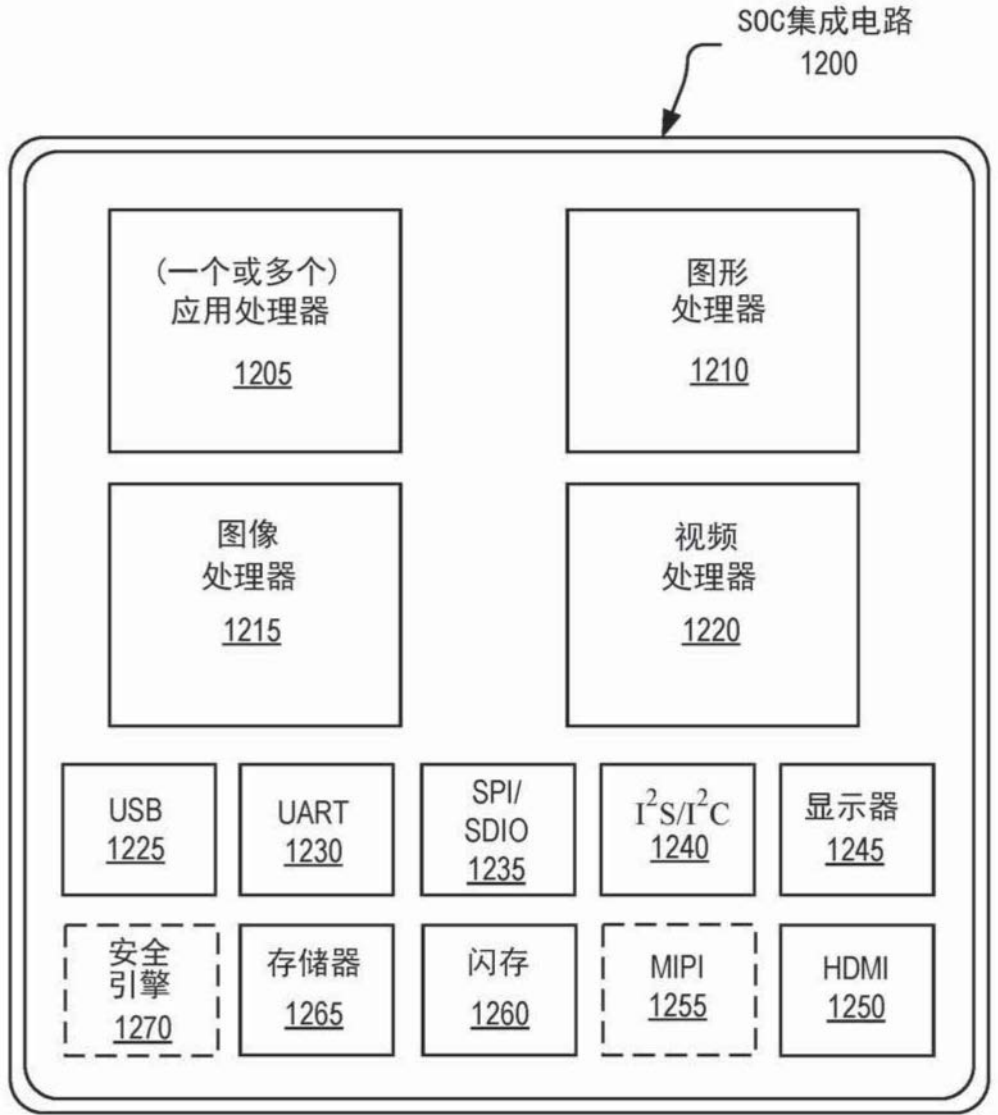


图12

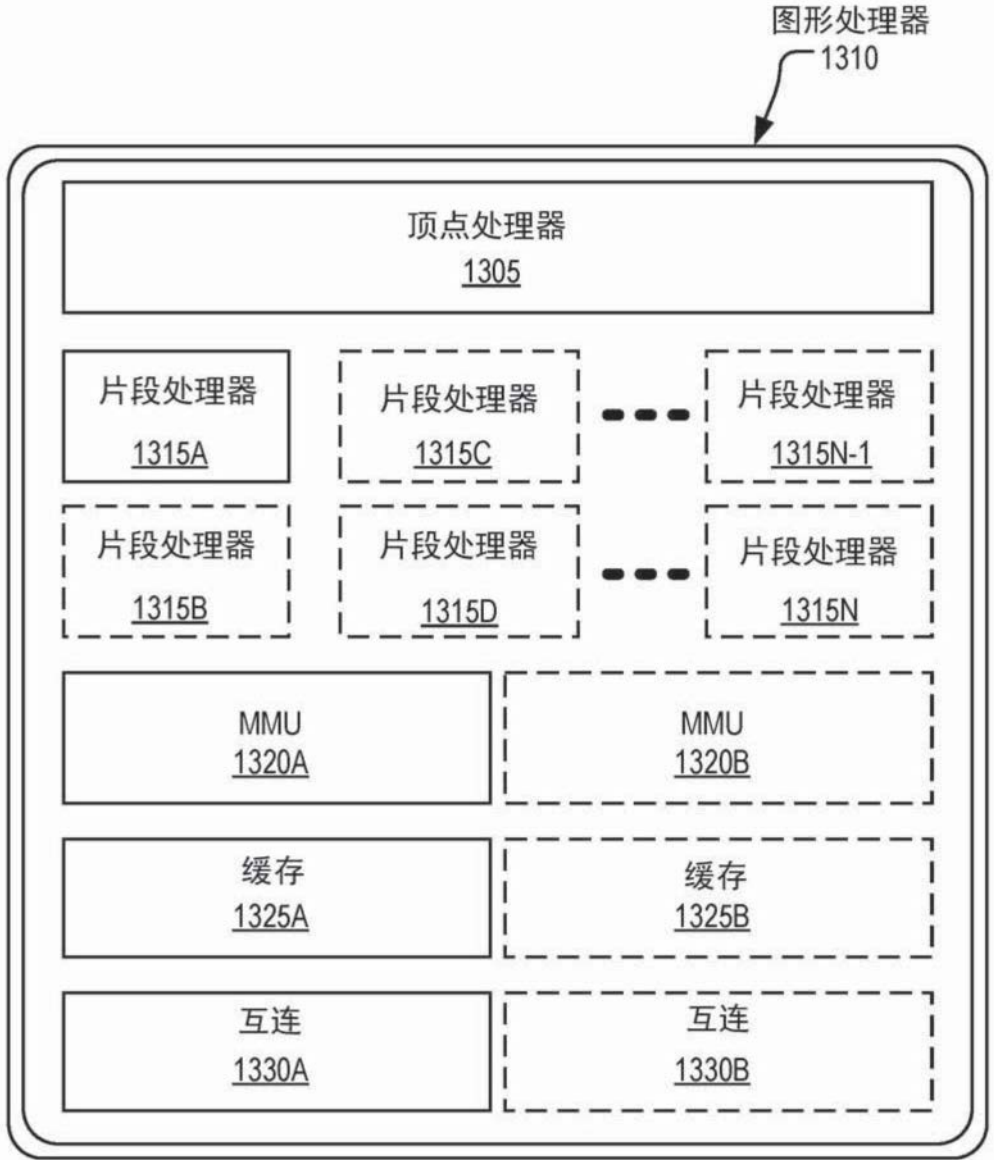


图13A

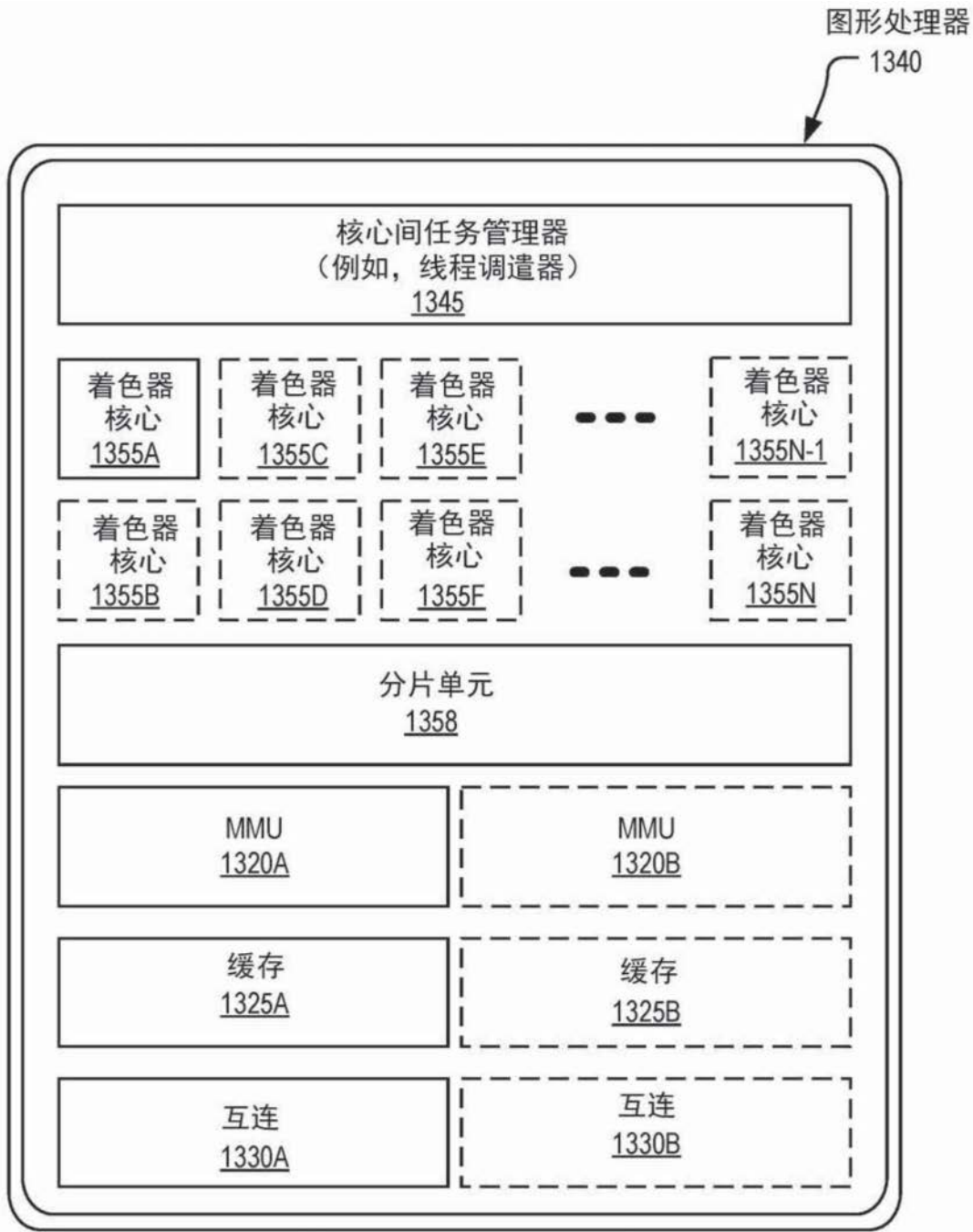


图13B

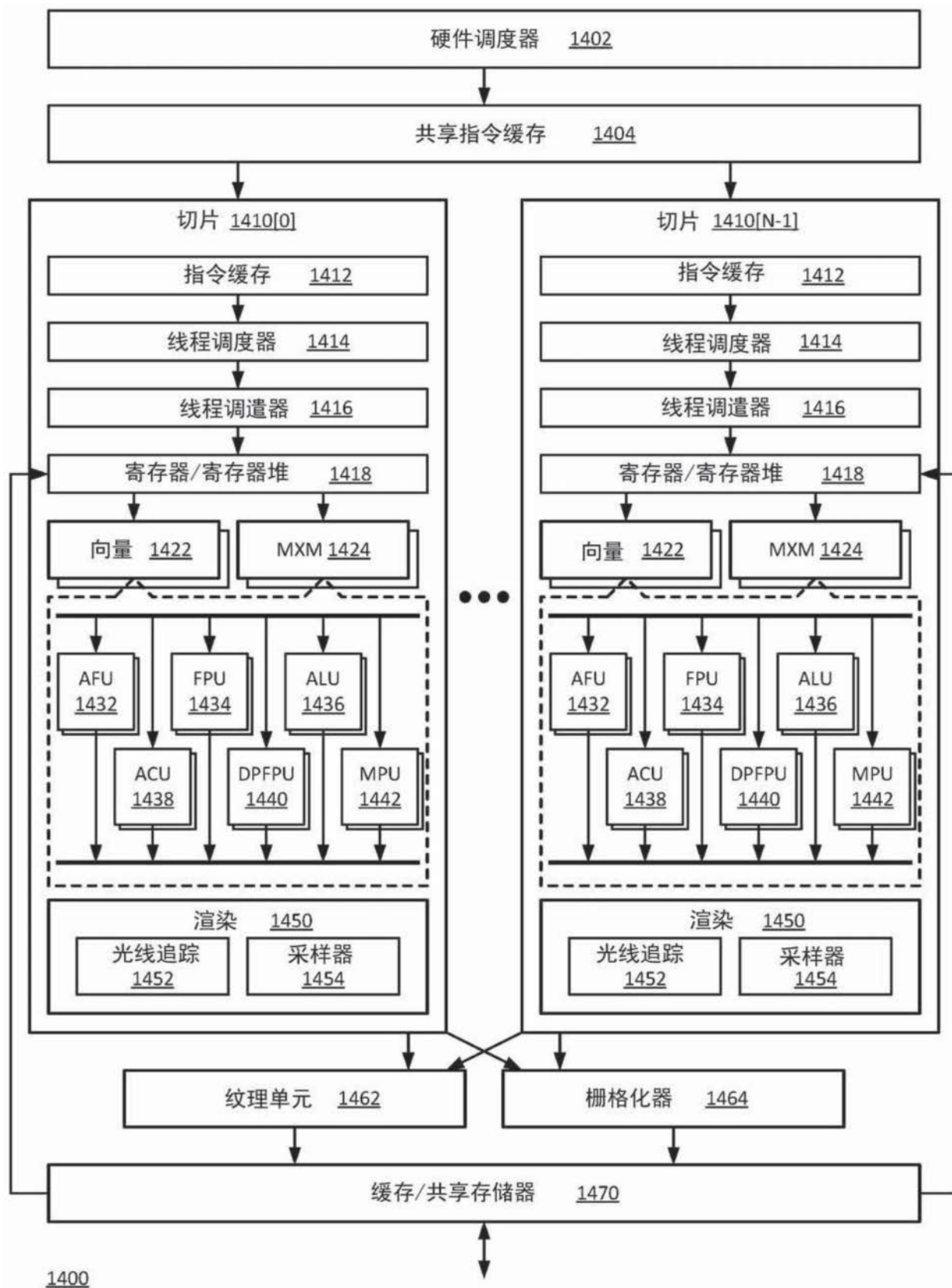


图14

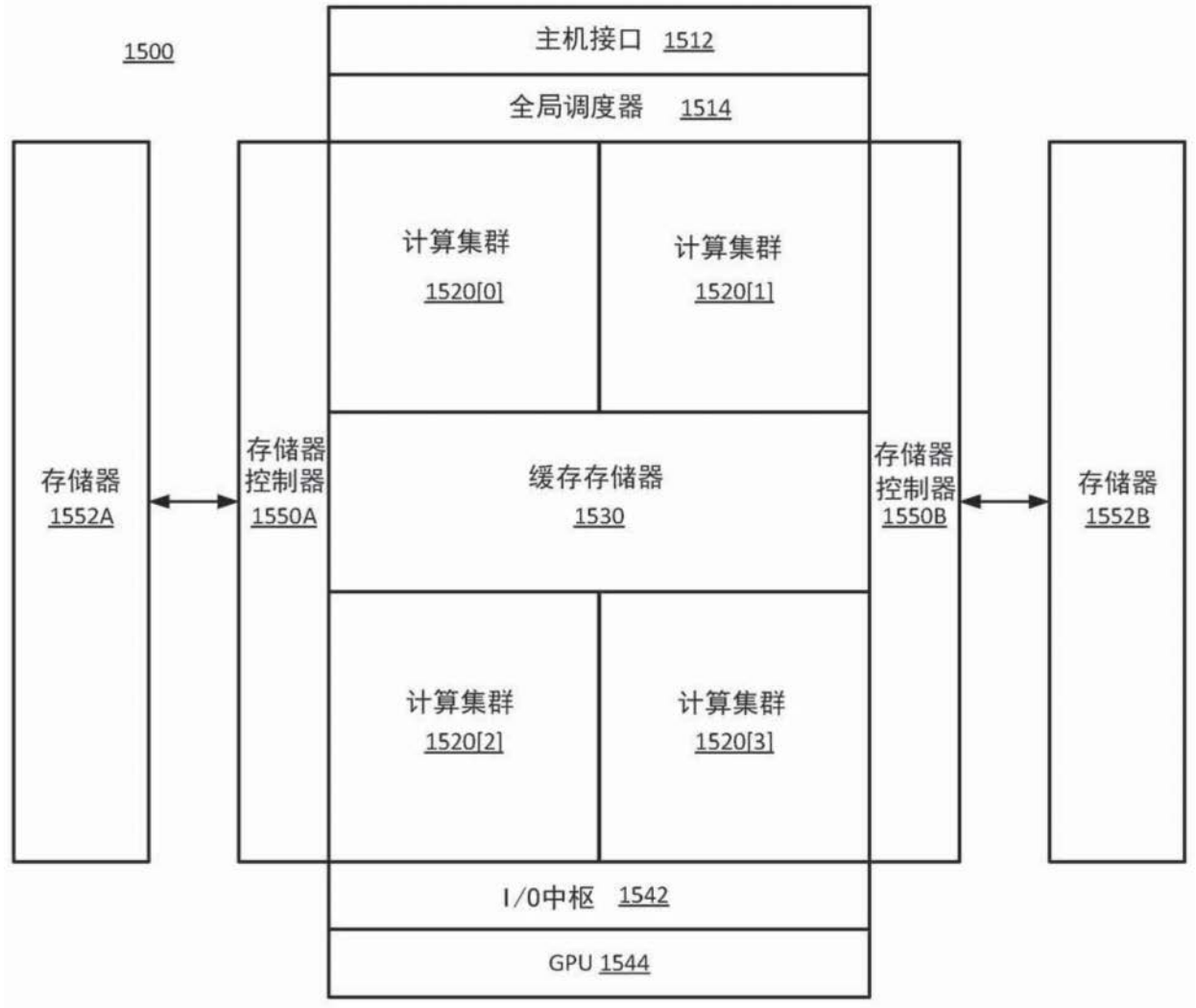


图15

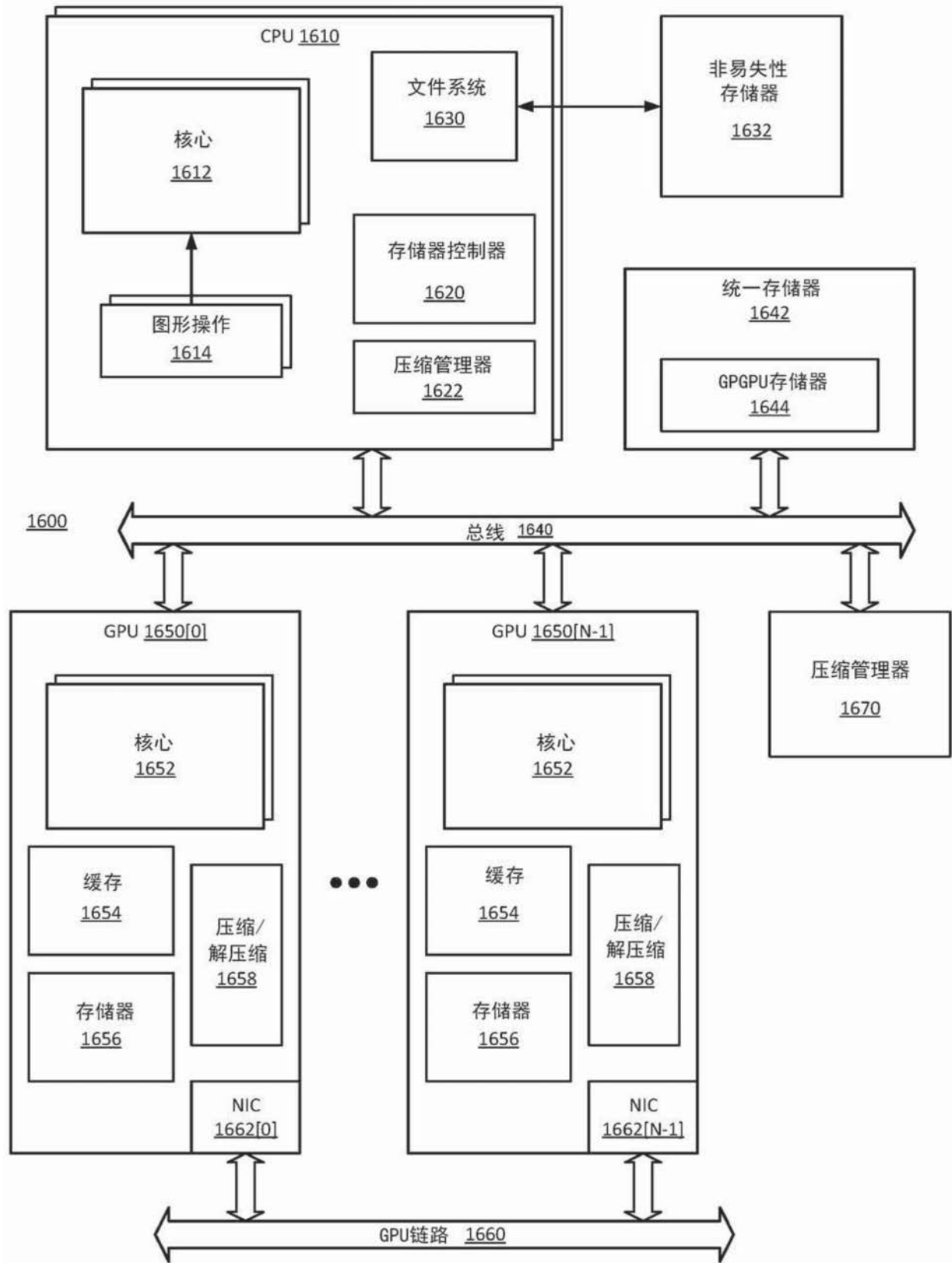


图16

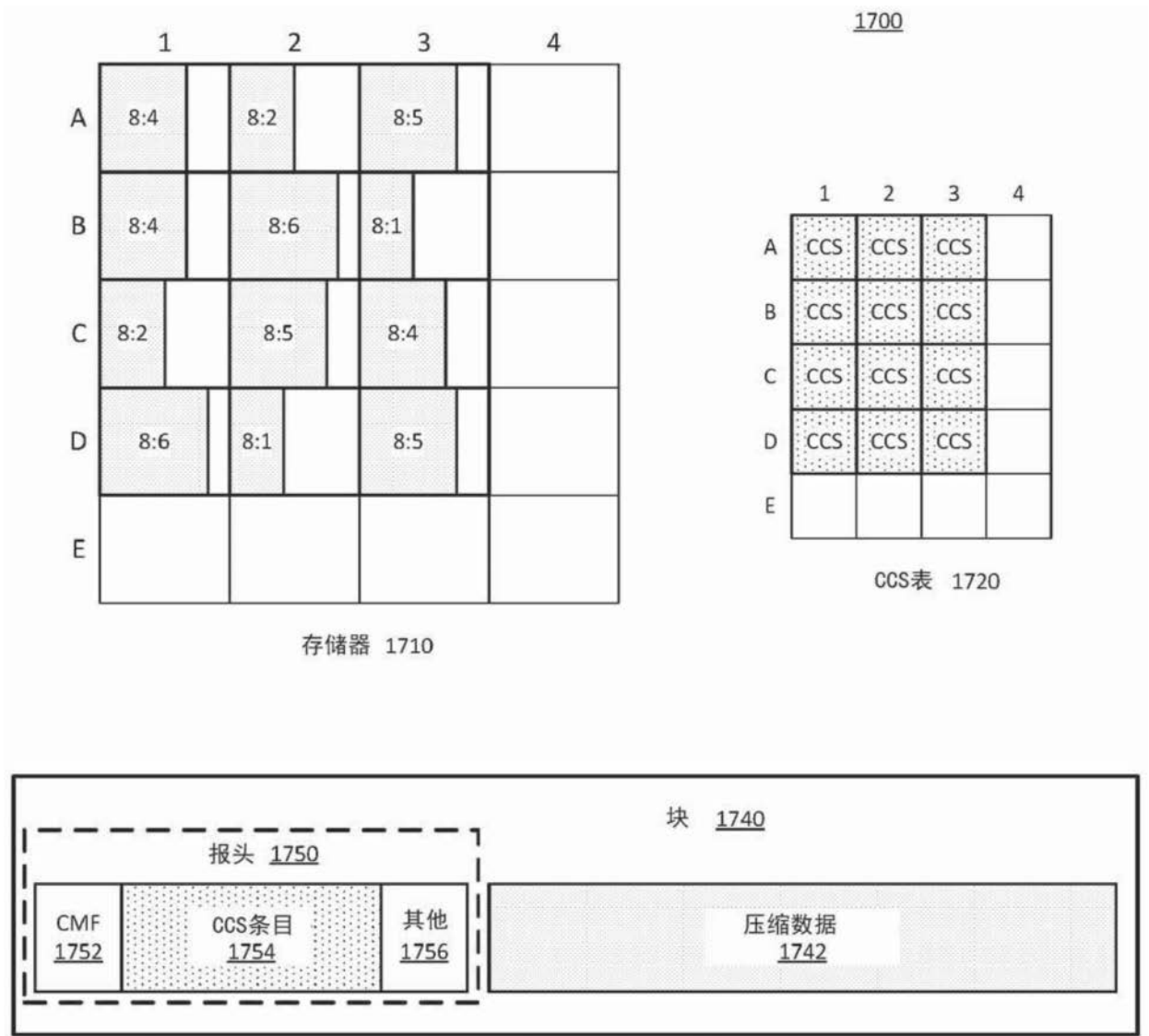


图17

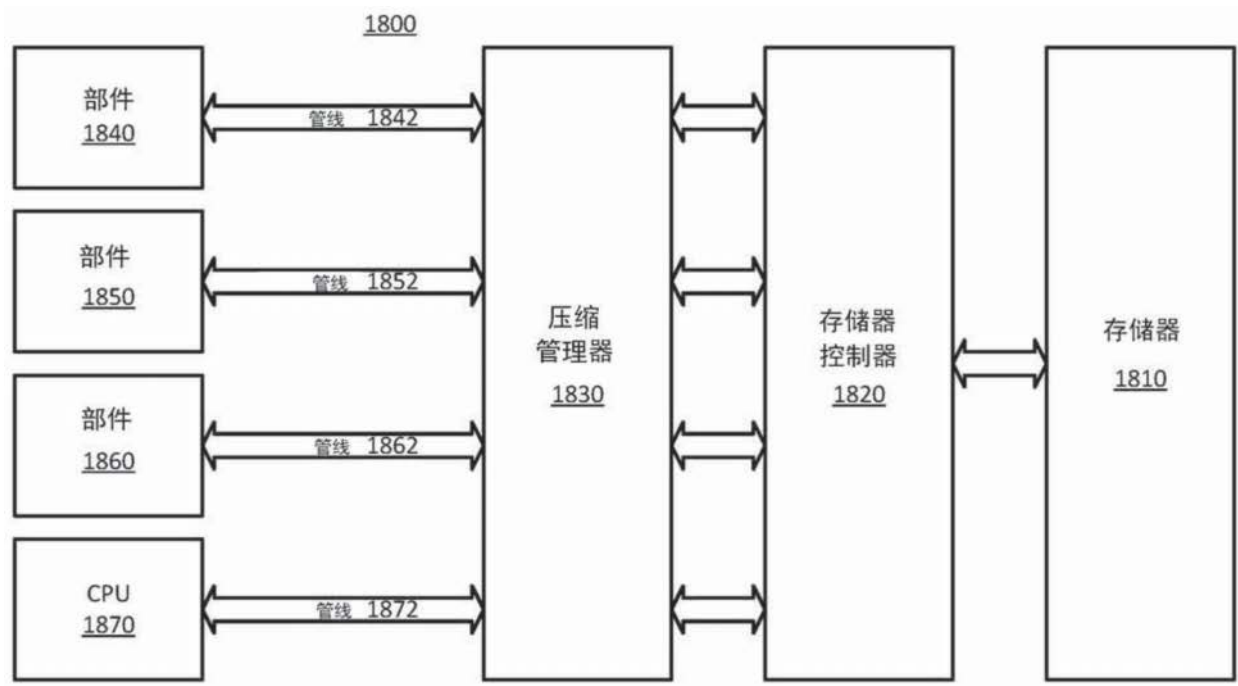


图18

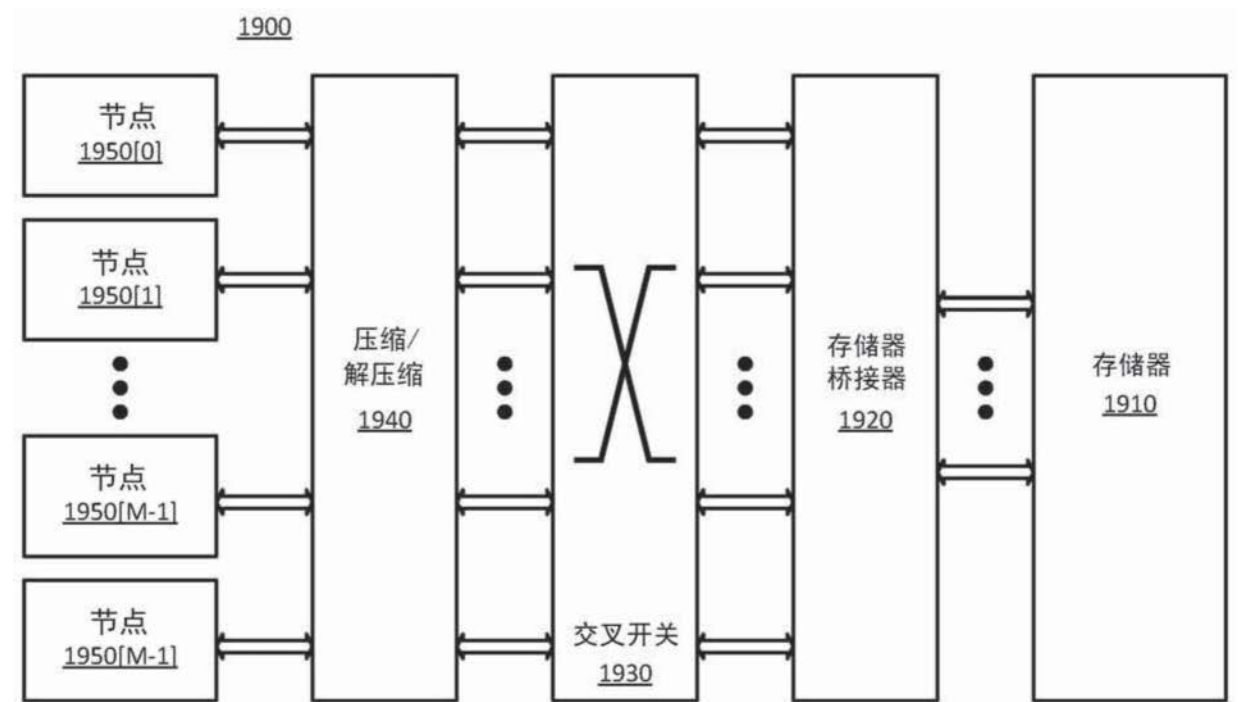


图19

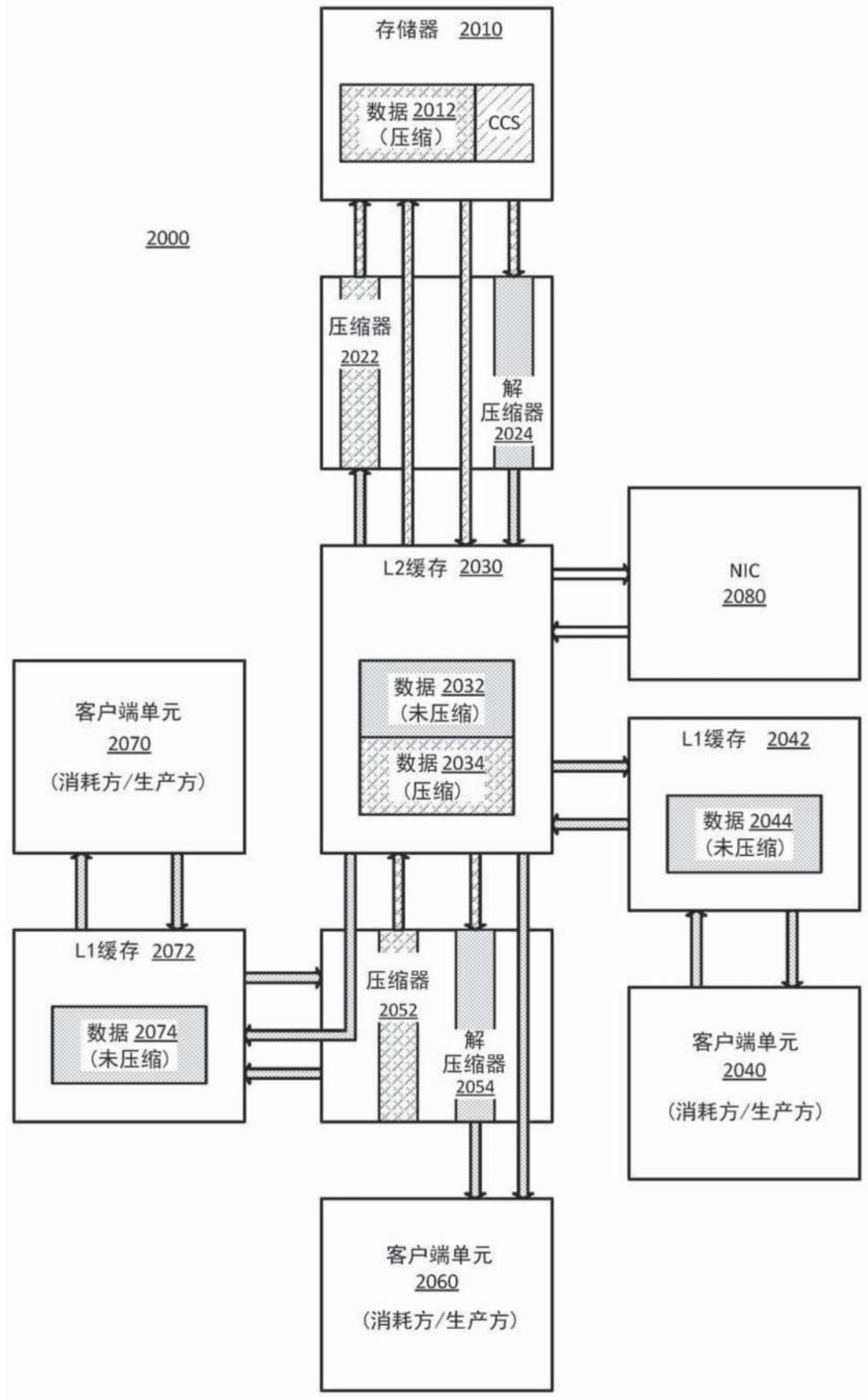


图20

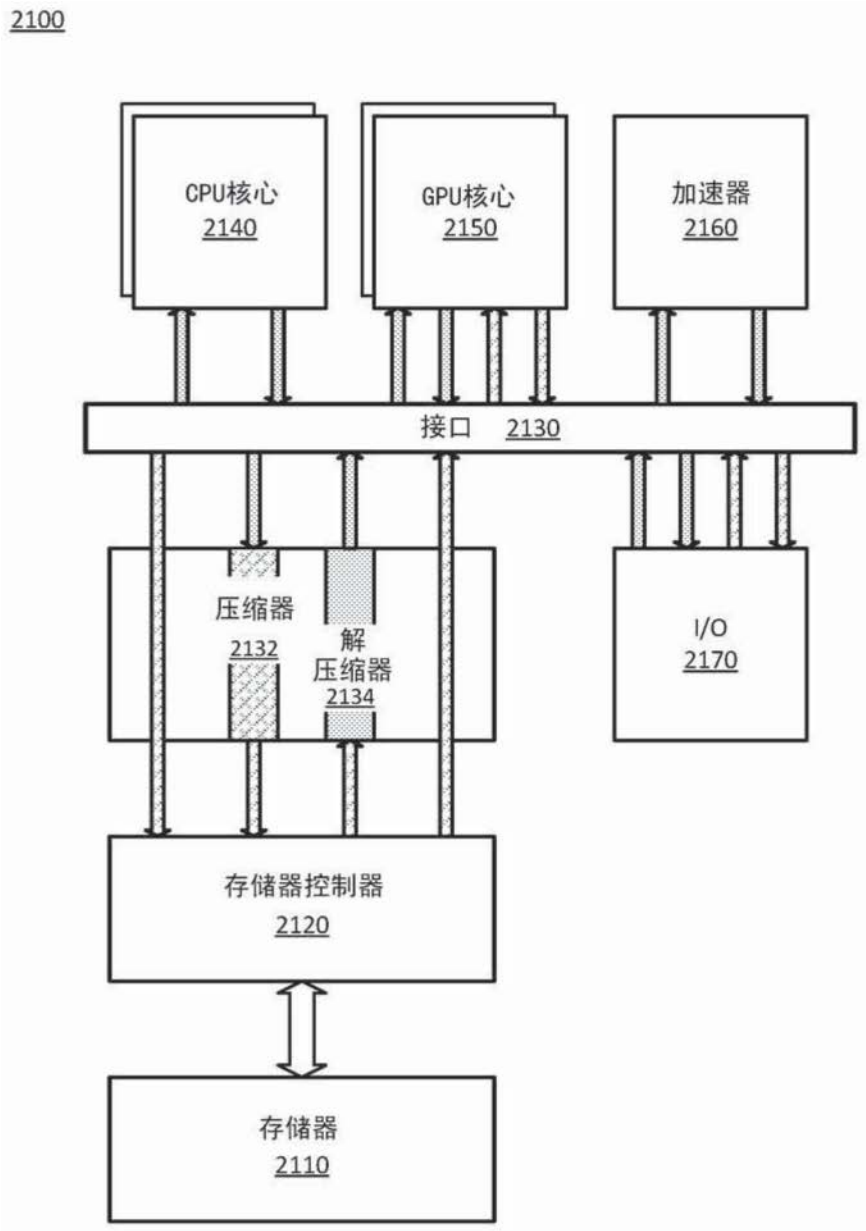


图21

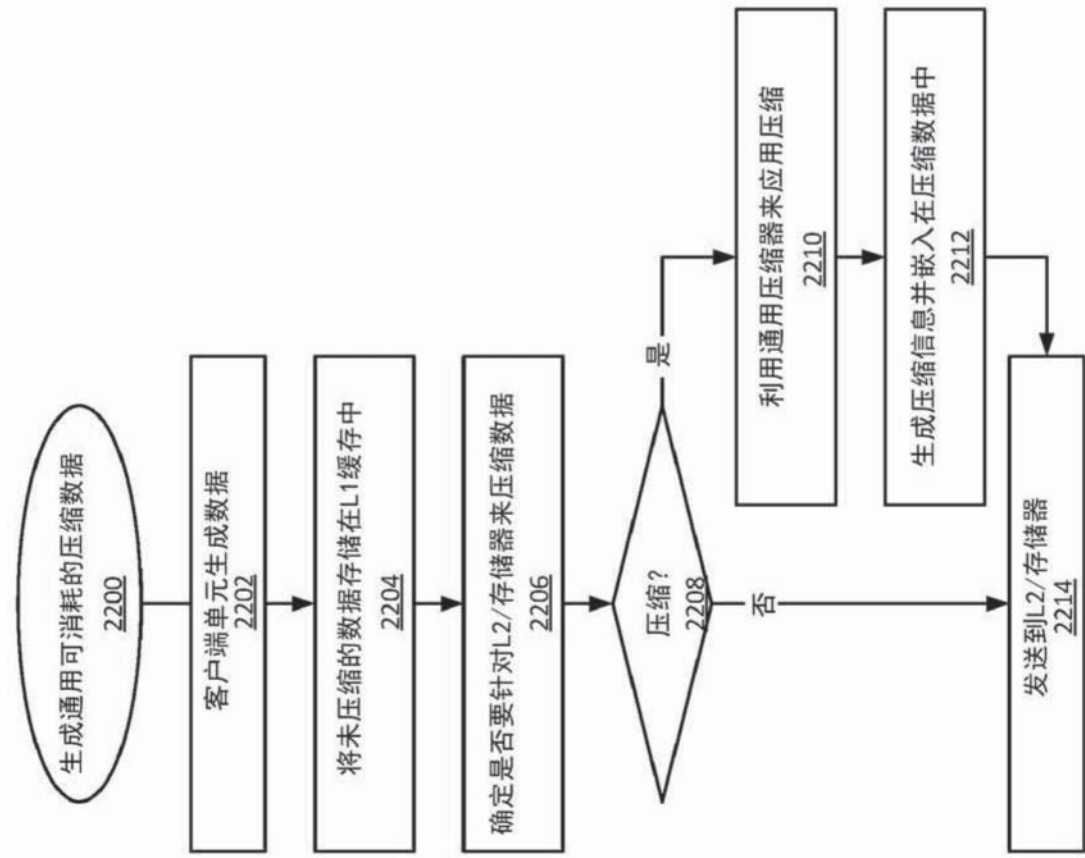


图22

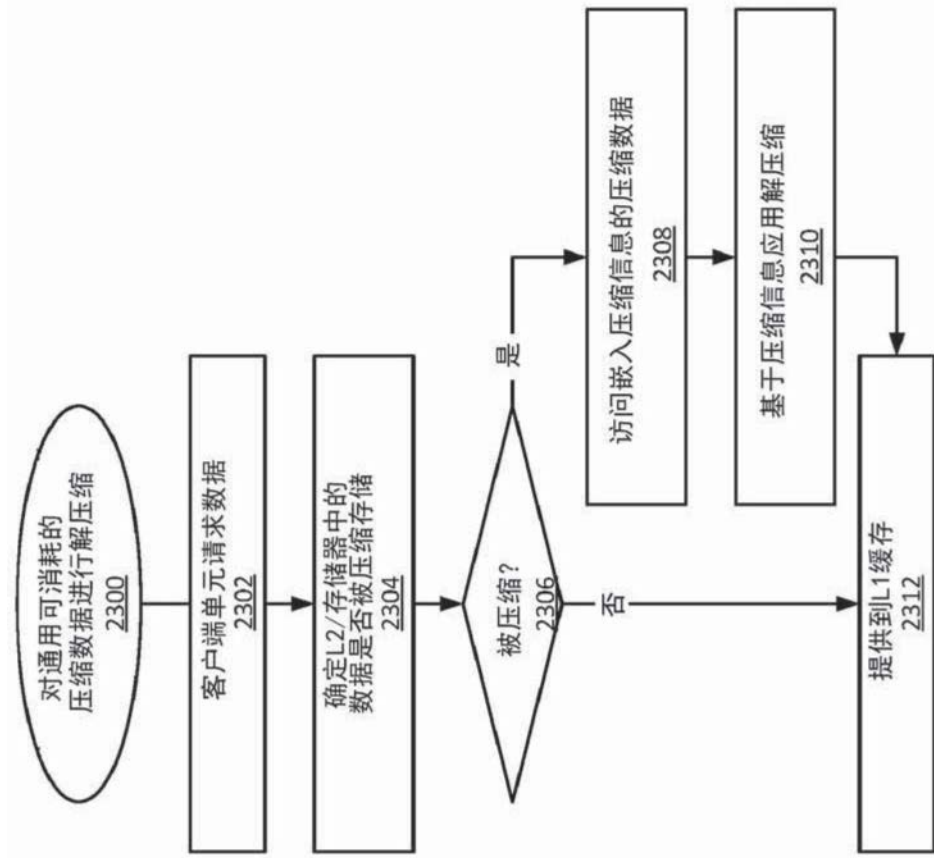


图23