(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 108694376 B
(45)授权公告日 2025.02.14

(21)申请号 201810283630.2

(22)申请日 2018.04.02

(65)同一申请的已公布的文献号
申请公布号 CN 108694376 A

(43)申请公布日 2018.10.23

(30)优先权数据
15/477,000 2017.04.01 US

(73)专利权人 英特尔公司
地址 美国加利福尼亚州

(72)发明人 J·D·吴 A·R·阿普 S·J·巴兰 S-H·李 A·H·默哈姆德 H-F·R·陈 J·雷

(74)专利代理机构 上海专利商标事务所有限公司 31100
专利代理师 李炜 黄嵩泉

(51)Int.Cl.
H04N 19/146(2014.01)
H04N 19/172(2014.01)
H04N 19/42(2014.01)
H04N 19/44(2014.01)

(56)对比文件
US 2011141230 A1,2011.06.16
US 2011211119 A1,2011.09.01
US 2015281638 A1,2015.10.01
US 6625333 B1,2003.09.23
Alatan等.Utilization of 3-D Motion Models in Video Coding:Occlusion Detection and Motion Compensated Temporal Interpolation.Advances in Digital Image Communication, Proceedings of 2nd Erlangen Symposium.1997, 第85-92页.

审查员 陈丹丹

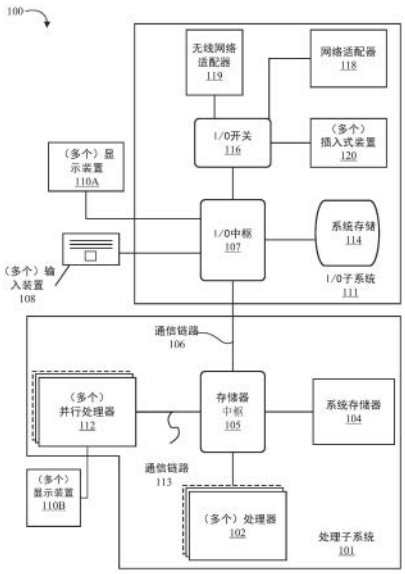
权利要求书3页 说明书44页 附图41页

(54)发明名称

包括静态场景确定、阻塞检测、帧率变换和调整压缩率的视频运动处理

(57)摘要

本申请提供了包括静态场景确定、阻塞检测、帧率变换和调整压缩率的视频运动处理。系统和方法可提供帧率变换中的阻塞检测。检测阻塞允许更准确地执行帧率变换。在一些实施例中,可使用一个或多个立体深度相机来确定移动物体的深度,从而更准确地确定阻塞。在一些实施例中,可调整压缩率来平衡帧率和功率以帮助确保符合功率预算。在至少一些实施例中,将相机的运动可从3D渲染管传递到编码器以避免运动计算并由此节省功率。



CN 108694376 B

1.一种用于视频运动处理的系统,包括:

可编程处理器;

存储器,所述存储器包括一组指令,所述一组指令如果由所述可编程处理器执行,则使所述可编程处理器从3D流水线接收视频信号;以及

逻辑,用于:

确定所述视频中的场景是静态的还是变化的;

在确定所述场景是变化的,并且所述场景中的物体在移动时:

接收帧的运动向量;

增加视频的帧率;

检测所述视频的一个或多个帧内的物体的运动;

在帧率变换过程中估计中间帧内的所述物体的阻塞,其中

所述阻塞是基于二维(2D)到三维(3D)结构估计而被估计的;以及

在确定所述视频中的场景是静态的且所述场景中没有对象移动

时,确定所述帧的视口是否是静态的,并且

在所述视口已改变时,将所述视口的编码传递给编码引擎以进行所述帧的编码,以及

在所述视口没有改变时,不对所述帧执行进一步的处理。

2.如权利要求1所述的系统,其特征在于,增加所述帧率改善图像震颤。

3.如权利要求2所述的系统,其特征在于,所述帧率变换包括将所述帧率变成两倍。

4.如权利要求1-3中的任一项所述的系统,其特征在于,所述2D到3D结构估计基于所述运动向量的位置来估计所述阻塞位置。

5.如权利要求1所述的系统,其特征在于,所述中间帧被内插在所述一个或多个帧中的两个帧之间。

6.如权利要求5所述的系统,其特征在于,所述移动物体被估计为被内插在遮挡的前方。

7.如权利要求5所述的系统,其特征在于,所述移动物体被估计为被内插在遮挡的后方。

8.一种用于视频运动处理的系统,包括:

一个或多个深度相机,用于确定移动物体的深度;

可编程处理器;

存储器,所述存储器包括一组指令,所述一组指令如果由所述可编程处理器执行,则使所述可编程处理器从3D流水线接收视频信号;以及

逻辑,用于:

确定所述视频中的场景是静态的还是变化的;

在确定所述场景是变化的,并且所述场景中的物体在移动时,

接收帧的运动向量;

增加视频的帧率;

检测所述视频的一个或多个帧内的物体的运动;

基于来自所述一个或多个深度相机的输入来检测所述物体的深度;在帧率变换过程中估计中间帧内的所述物体的阻塞,其中所述阻塞是基于二维(2D)到三维(3D)结构估计而被

估计的;

在帧率变换过程中内插所述中间帧;

在确定所述视频中的场景是静态的且所述场景中没有对象移动时,确定所述帧的视口是否是静态的,并且

在所述视口已改变时,将所述视口的编码传递给编码引擎以进行所述帧的编码,以及

在所述视口没有改变时,不对所述帧执行进一步的处理。

9.如权利要求8所述的系统,其特征在于,增加所述帧率改善图像震颤。

10.如权利要求9所述的系统,其特征在于,所述帧率变换包括将所述帧率变成两倍。

11.如权利要求8所述的系统,其特征在于,所确定的深度用于改善阻塞估计。

12.如权利要求11所述的系统,其特征在于,所述深度相机是从包括飞行时间传感器和已编码光传感器的组中选择的。

13.如权利要求8-11中的任一项所述的系统,其特征在于,所述移动物体被估计为被内插在遮挡的前方。

14.如权利要求8-11中的任一所述的系统,其特征在于,所述移动物体被估计为被内插在遮挡的后方。

15.一种用于视频运动处理的方法,包括:

从3D流水线接收视频信号,并确定所述视频中的场景是静态的还是变化的;

在确定所述场景是变化的,并且所述场景中的物体在移动时:

接收帧的运动向量;

增加视频的帧率;

检测所述视频的一个或多个帧内的物体的运动;

估计中间帧内的物体的阻塞;

在帧率变换过程中内插所述中间帧,其中所述阻塞是基于二维(2D)到三维(3D)结构估计而被估计的;以及

在确定所述视频中的场景是静态的且所述场景中没有对象移动时,确定所述帧的视口是否是静态的,并且

在所述视口已改变时,将所述视口的编码传递给编码引擎以进行所述帧的编码,以及

在所述视口没有改变时,不对所述帧执行进一步的处理。

16.如权利要求15所述的方法,其特征在于,增加所述帧率改善图像震颤。

17.如权利要求16所述的方法,其特征在于,所述帧率变换包括将所述帧率变成两倍。

18.如权利要求15-17中的任一项所述的方法,其特征在于,所述2D到3D结构估计基于所述运动向量的位置来估计所述阻塞位置。

19.如权利要求15所述的方法,其特征在于,所述中间帧被内插在所述一个或多个帧中的两个帧之间。

20.如权利要求19所述的方法,其特征在于,所述移动物体被估计为被内插在遮挡的前方或后方。

21.一种视口编码系统,包括:

一个或多个相机;

可编程处理器;

存储器,所述处理器包括一组指令,所述一组指令如果由所述可编程处理器执行,则使所述可编程处理器接收与所述一个或多个相机有关的视口信息;以及

逻辑,用于:

确定所述一个或多个相机是否是移动的;

确定所述视口中的场景是否是静态的以及所述视口是否是静态的,其中当所述场景不是静态的时,编码被传递给阻塞检测器,其中所述阻塞检测器用于估计中间帧内的物体的阻塞,并且其中所述阻塞是基于二维(2D)到三维(3D)结构估计而被估计的;以及

在确定所述视口中的场景是静态的,且所述视口已改变时,将所述视口的编码传递给编码引擎,其中所述编码在无需计算运动的情况下被传递。

22.如权利要求21所述的系统,其特征在于,场景状态是经由所述编码引擎中的API确定的。

23.如权利要求22所述的系统,其特征在于,当所述场景是静态的,并且所述视口没有改变时,所述逻辑结束处理。

包括静态场景确定、阻塞检测、帧率变换和调整压缩率的视频运动处理

[0001] 相关申请的交叉引用

[0002] 本申请是2017年4月1日提交的题为“VIDEO MOTION PROCESSING INCLUDING STATIC SCENE DETERMINATION,OCCLUSION DETECTION,FRAME RATE CONVERSION,AND ADJUSTING COMPRESSION RATIO(包括静态场景确定、阻塞检测、帧率变换和调整压缩率的视频运动处理)”的美国专利申请第15/477000号的部分接续并要求其优先权。

技术领域

[0003] 实施例总体上涉及视频运动处理,并更具体地涉及确定静态场景、检测移动物体的遮挡、转换帧率和调整压缩率。

背景技术

[0004] 先进的视频处理技术已经启用了包括例如三维(3D)和虚拟内容的许多新的、令人激动视频应用。可能涉及向用户递送3D、虚拟现实(VR)、增强现实(AR)和其他浸入式观看环境的这些应用会需要在维持高准确度的同时快速处理。而且,运动检测和处理会倾向于涉及复杂的编码和处理,并聚有高功率要求。

附图说明

[0005] 实施例的各种优点将通过阅读以下说明和所附权利要求以及通过参考以下附图而变得为本领域技术人员所显而易见,在附图中:

[0006] 图1是展示计算机系统的框图,所述计算机系统被配置成实现本文描述的实施例的一个或多个方面;

[0007] 图2A到图2D展示了根据实施例的并行处理器部件;

[0008] 图3A到图3B是根据实施例的图形多处理器的框图;

[0009] 图4A到图4F展示了示例性架构,其中多个GPU通信地耦合至多个多核处理器;

[0010] 图5展示根据实施例的图形处理流水线;

[0011] 图6是根据各种实施例的视频处理系统的示例的图;

[0012] 图7A根据实施例的运动检测过程的示例的流程图;

[0013] 图7B是根据实施例的检测视频中的运动的过程的示例;

[0014] 图8A-B是根据实施例的阻塞检测过程的示例的图;

[0015] 图8C是根据实施例的用于检测视频中的移动物体的阻塞的过程的示例;

[0016] 图9A是根据实施例的阻塞深度检测系统的示例的图;

[0017] 图9B是根据实施例的用于确定移动阻塞的深度的过程的示例;

[0018] 图10A是根据实施例的视频压缩系统的示例;

[0019] 图10B是根据实施例的用于压缩视频的过程的示例;

[0020] 图11是根据实施例的具有本地背光能力的显示器的示例的框图;

[0021] 图12A是根据实施例的数据处理装置的示例的框图;

[0022] 图12B是根据实施例的距离确定的示例的展示;

[0023] 图13是根据实施例的分层显示架构的示例的框图;

[0024] 图14是根据实施例的显示架构的示例的框图,所述显示架构包括多个显示单元;

[0025] 图15是根据实施例的云辅助的媒体递送架构的示例的框图;

[0026] 图16到图18是根据实施例的数据处理系统的概述的示例的框图;

[0027] 图19是根据实施例的图形处理引擎的示例的框图;

[0028] 图20到图22是根据实施例的执行单元的示例的框图;

[0029] 图23是根据实施例的图形流水线的示例的框图;

[0030] 图24A到图24B是根据实施例的图形流水线的示例的框图;

[0031] 图25是根据实施例的图形软件架构的示例的框图;

[0032] 图26是根据实施例的知识产权(IP)核开发系统的示例的框图;以及

[0033] 图27是根据实施例的片上系统集成电路的示例的框图。

具体实施方式

[0034] 在以下描述中,阐述了许多细节以提供对本发明的更透彻的理解。然而对于本领域技术人员显而易见的是,没有这些具体细节中的一个或多个也可实践本发明。在其他实例中,未描述公知的特征以避免使本发明含糊。

[0035] 系统概述

[0036] 图1是展示计算系统100的框图,所述计算系统被配置成实现本文描述的实施例的一个或多个方面。计算系统100包括处理子系统101,所述处理子系统具有一个或多个处理器102和系统存储器104,所述处理器与所述系统存储器经由可包括存储器中枢105的互连路径来通信。存储器中枢105可以是芯片组部件内的单独的部件,或可以集成在一个或多个处理器102内。存储器中枢105经由通信链路106与I/O子系统111耦合。I/O子系统111包括I/O中枢107,所述I/O中枢可以使得计算系统100能够从一个或多个输入装置108接收输入。另外,I/O中枢107可以使得显示控制器能够将输出提供给一个或多个显示装置110A,所述显示控制器可被包括在一个或多个处理器102中。在一个实施例中,与I/O中枢107耦合的一个或多个显示装置110A可以包括本地、内部或嵌入式显示装置。

[0037] 在一个实施例中,处理子系统101包括一个或多个并行处理器112,所述并行处理器经由总线或其他通信链路113耦合至存储器中枢105。通信链路113可以是任何数目的基于标准的通信链路技术或协议中的一者(比如但不限于,PCI快速总线),或可以是供应方特定的通信接口或通信结构。在一个实施例中,一个或多个并行处理器112形成计算上集中的并行或向量处理系统,所述系统包括大量处理核和/或处理集群(比如,集成众核(MIC)处理器)。在一个实施例中,一个或多个并行处理器112形成图形处理子系统,所述图形处理子系统可以将像素输出到经由I/O中枢107耦合的一个或多个显示装置110A中的一者。一个或多个并行处理器112还可以包括显示控制器和显示接口(未示出)以使得能够直接连接到一个或多个显示装置110B。

[0038] 在I/O子系统111内,系统存储单元114可以连接到I/O中枢107以提供用于计算系统100的存储机制。I/O开关116可以用于提供接口机制以实现I/O中枢107与其他部件(比

如,可集成到平台中的网络适配器118和/或无线网络适配器119,以及可以经由一个或多个插入式装置120添加的各种其他装置)之间的连接。网络适配器118可以是以太网适配器或另一有线网络适配器。无线网络适配器119可以包括以下各者中的一者或多者:Wi-Fi、蓝牙、近场通信(NFC)、或包括一个或多个无线电装置的其他网络装置。

[0039] 计算系统100可以包括未明确示出的其他部件,包括USB或其他端口连接件、光学存储驱动器、视频捕获装置等等,它们也可连接到I/O中枢107。使图1中的各种部件互连的通信路径可使用任何合适的协议来实现,比如基于PCI(外围部件互连)的协议(例如,PCI快速总线)或任何其他总线或点对点通信接口和/或协议(比如,NV-Link高速互连、或本领域中已知的互连协议)。

[0040] 在一个实施例中,一个或多个并行处理器112包括针对图形和视频处理进行优化的电路(包括(例如),视频输出电路),并且构成图形处理单元(GPU)。在另一个实施例中,一个或多个并行处理器112包括针对通用处理进行优化的电路,同时保持本文更详细地描述的底层计算架构。在又一个实施例中,计算系统100的部件可与一个或多个其他系统元件一起集成在单个集成电路上。例如,一个或多个并行处理器112、存储器中枢105、处理器102和I/O中枢107可以集成到芯片上系统(SoC)集成电路中。可替代地,计算系统100的部件可以集成到单个封装中以形成封装中系统(SIP)配置。在一个实施例中,计算系统100的部件的至少一部分可以集成到多芯片模块(MCM)中,所述MCM可以与其他多芯片模块一起互连到模块化计算系统中。

[0041] 将认识到的是,本文示出的计算系统100是展示性的,并且各种变型和修改是有可能的。可根据需要修改连接拓扑,包括桥的数目和排、(多个)处理器102的数目和(多个)并行处理器112的数目。例如,在一些实施例中,系统存储器104直接而非通过桥连接到(多个)处理器102,而其他装置经由存储器中枢105和(多个)处理器102与系统存储器104通信。在其他替代性拓扑中,(多个)并行处理器112连接到I/O中枢107或直接连接到一个或多个处理器102中的一者,而非连接到存储器中枢105。在其他实施例中,I/O中枢107和存储器中枢105可集成到单个芯片中。一些实施例可包括经由多个插口附接的两组或更多组处理器102,它们可以与(多个)并行处理器112的两个或更多个实例耦合。

[0042] 本文中示出的一些特定部件是可选的,并且可以不被包括在计算系统100的所有实现方式中。例如,可支持任何数量的插入式卡或外设,或可消除一些部件。此外,一些架构可对于与图1中展示的那些部件类似的部件使用不同的术语。例如,在一些架构中,存储器中枢105可称为北桥,而I/O中枢107可称为南桥。

[0043] 图2A展示了根据实施例的并行处理器200。并行处理器200的各种部件可使用一个或多个集成电路装置来实现,比如可编程处理器、专用集成电路(ASIC)或现场可编程门阵列(FPGA)。根据实施例,所展示的并行处理器200是图1中所示的一个或多个并行处理器112的变体。

[0044] 在一个实施例中,并行处理器200包括并行处理单元202。所述并行处理单元包括I/O单元204,所述I/O单元实现与其他装置(包括并行处理单元202的其他实例)的通信。I/O单元204可直接连接到其他装置。在一个实施例中,I/O单元204经由使用中枢或开关接口(比如,存储器中枢105)来与其他装置连接。存储器中枢105与I/O单元204之间的连接形成通信链路113。在并行处理单元202内,I/O单元204与主机接口206和存储器交叉开关

(memory crossbar)216连接,其中,主机接口206接收涉及执行处理操作的命令,并且存储器交叉开关216接收涉及执行存储器操作的命令。

[0045] 当主机接口206经由I/O单元204接收命令缓冲时,主机接口206可以将用于执行那些命令的工作操作导引至前端208。在一个实施例中,前端208与调度器210耦合,该调度器210被配置成将命令或其他工作项目分布至处理集群阵列212。在一个实施例中,调度器210确保在任务被分布至处理集群阵列212的处理集群之前,处理集群阵列212被适当地配置且处于有效状态。在一个实施例中,调度器210是经由在微控制器上执行的固件逻辑实现的。微控制器实现的调度器210可配置用于以粗粒度和细粒度执行复杂的调度和工作分布操作,从而实现在处理阵列212上执行的线程的快速抢占和上下文切换。在一个实施例中,主机软件可以经由多个图像处理门铃中的一个来证明工作负荷以用于在处理阵列212上调度。随后工作负荷可以由调度器微控制器内的调度器210逻辑跨处理阵列212自动地分布。

[0046] 处理集群阵列212可以包括多达“N”个处理集群(例如,集群214A、集群214B、直到集群214N)。处理集群阵列212的每一个集群214A-214N都可以执行大量的并发线程。调度器210可以使用各种调度和/或工作分布算法来将工作分配给处理集群阵列212的集群214A-214N,各种调度和/或工作分布算法可取决于为每一种类型的程序或计算而产生的工作负荷而变化。调度可以由调度器210动态地处置,或者可以在配置用于由处理集群阵列212执行的程序逻辑的编译期间部分地由编译器逻辑辅助。在一个实施例中,可以将处理集群阵列212的不同集群214A-214N分配用于处理不同类型的程序,或用于执行不同类型的计算。

[0047] 可以将处理集群阵列212配置成执行各种类型的并行处理操作。在一个实施例中,将处理集群阵列212配置成执行通用并行计算操作。例如,处理集群阵列212可以包括用于执行处理任务的逻辑,处理任务包括过滤视频和/或音频数据、执行建模操作(包括物理操作)、以及执行数据变换。

[0048] 在一个实施例中,处理集群阵列212被配置成执行并行的图形处理操作。在其中并行处理器200被配置成执行图形处理操作的实施例中,处理集群阵列212可以包括用于支持执行这样的图形处理操作的附加逻辑,包括但不限于用于执行纹理操作的纹理采样逻辑、以及曲面细分逻辑和其他顶点处理逻辑。另外,处理集群阵列212可以被配置成执行与图形处理有关的着色器程序,比如但不限于顶点着色器、曲面细分着色器、几何着色器和像素着色器。并行处理单元202可以经由I/O单元204传递来自系统存储器的数据以供处理。在处理期间,可以将所传递的数据在处理期间存储到芯片上存储器(例如,并行处理器存储器222),然后将其写回到系统存储器。

[0049] 在一个实施例中,当并行处理单元202用于执行图形处理时,调度器210可以被配置成将处理工作负荷划分成近似等规模的任务,以更好地使得能够将图形处理操作分布到处理集群阵列212中的多个集群214A至214N。在一些实施例中,处理集群阵列212的多个部分可以被配置成执行不同类型的处理。例如,第一部分可被配置成执行顶点着色和拓扑生成,第二部分可被配置成执行曲面细分和几何着色,并且第三部分可被配置成执行像素着色或其他屏幕空间操作,以产生供显示的渲染图像。由集群214A至214N中的一者或多者产生的中间数据可存储在缓冲器中以允许在集群214A至214N之间传输所述中间数据以供进一步处理。

[0050] 在操作期间,处理集群阵列212可以经由调度器210来接收待执行的处理任务,所

述调度器从前端208接收定义处理任务的命令。针对图形处理操作,处理任务可以包括待处理的数据(例如,表面(补片(patch))数据、图元数据(primitive data)、顶点数据和/或像素数据)的索引以及状态参数和定义要如何处理数据(例如,要执行什么程序)的命令。调度器210可被配置成获取与任务相对应的索引,或可从前端208接收这些索引。前端208可以被配置成确保在发起由进入的命令缓冲(例如,分批缓冲、推动缓冲等)指定的工作负荷之前处理集群阵列212被配置成有效状态。

[0051] 并行处理单元202的一个或多个实例中的每一个都可以与并行处理器存储器222耦合。并行处理器存储器222可以经由存储器交叉开关216来访问,存储器交叉开关216可以从处理集群阵列212以及I/O单元204接收存储器请求。存储器交叉开关216可以经由存储器接口218访问并行处理器存储器222。存储器接口218可以包括多个分区单元(例如,分区单元220A、分区单元220B、直到分区单元220N),每一个分区单元都可以耦合至并行处理器存储器222的一部分(例如,存储器单元)。在一种实现方式中,将分区单元220A-220N的数量配置成等于存储器单元的数量,使得第一分区单元220A具有对应的第一存储器单元224A,第二分区单元220B具有对应的存储器单元224B,并且第N分区单元220N具有对应的第N存储器单元224N。在其他实施例中,分区单元220A-220N的数量可以不等于存储器装置的数量。

[0052] 在各种实施例中,存储器单元224A至224N可以包括各种类型的存储器装置,包括动态随机存取存储器(DRAM)或图形随机存取存储器(比如,同步图形随机存取存储器(SGRAM),包括图形双数据速率(GDDR)存储器)。在一个实施例中,存储器单元224A至224N还可包括3D堆叠式存储器,包括但不限于高带宽存储器(HBM)。本领域技术人员将认识到,存储器单元224A至224N的具体的实现方式可以变化,并且可以选自各种常规设计中的一者。渲染目标(比如,帧缓冲器或纹理映射(texture map))可跨越存储器单元224A至224N存储,从而允许分区单元220A至220N并行写入每个渲染目标的多个部分以高效地使用并行处理器存储器222的可用带宽。在一些实施例中,可排除并行处理器存储器222的本地实例,以有利于结合本地高速缓存存储器来利用系统存储器的统一的存储器设计。

[0053] 在一个实施例中,处理集群阵列212的集群214A-214N中的任何一个都可以处理将被写入并行处理器存储器222内的存储器单元224A-224N中的任何一个的数据。可以将存储器交叉开关216配置成将每一个集群214A-214N的输出传递到可以对输出执行附加的处理操作的任何分区单元220A-220N或另一集群214A-214N。每一个集群214A-214N都可以通过存储器交叉开关216与存储器接口218通信,以便从各种外部存储器装置读取或向各种外部存储器装置写入。在一个实施例中,存储器交叉开关216具有到存储器接口218的连接以与I/O单元204通信,并具有到并行处理器存储器222的本地实例的连接,从而使不同的处理集群214A-214N内的处理单元能够与系统存储器或不在并行处理单元202本地的其他存储器通信。在一个实施例中,存储器交叉开关216可以使用虚拟通道以分离集群214A-214N与分区单元220A-220N之间的业务流。

[0054] 虽然在并行处理器200内展示了并行处理单元202的单个实例,但是可以包括并行处理单元202的任何数目的实例。例如,可以在单个插入式卡上提供并行处理单元202的多个实例,或可以将多个插入式卡互连。并行处理单元202的不同实例可以被配置成即使这些不同实例具有不同数目的处理核、不同量的本地并行处理器存储器和/或其他配置差异而仍互操作。例如且在一个实施例中,并行处理单元202的一些实例可以相对于其他实例包括

更高精度浮点单元。包括并行处理单元202或并行处理器200的一个或多个实例的系统可以以多种配置和形状因数来实现,包括但不限于台式、膝上型或手持式个人计算机、服务器、工作站、游戏控制台和/或嵌入式系统。

[0055] 图2B是根据实施例的分区单元220的框图。在一个实施例中,分区单元220是图2A的分区单元220A至220N中的一者的实例。如所展示,分区单元220包括L2高速缓存221、帧缓冲器接口225和ROP 226(光栅操作单元)。L2高速缓存221是读/写高速缓存,其被配置成执行从存储器交叉开关216和ROP 226接收的加载和存储操作。由L2高速缓存221将读未命中(read miss)和紧急回写请求输出到帧缓冲器接口225以供处理。也可以经由帧缓冲器接口225将更新发送到帧缓冲器以供处理。在一个实施例中,帧缓冲器接口225与并行处理器存储器中的存储器单元(比如,图2的存储器单元224A至224N(例如,在并行处理器存储器222内))中的一者交界。

[0056] 在图形应用中,ROP 226是执行诸如模板印刷(stencil)、z测试、混合等等的光栅操作的处理单元。随后ROP 226输出存储在图形存储器中的处理过的图形数据。在一些实施例中,ROP 226包括压缩逻辑,该压缩逻辑用于压缩写入到存储器的深度或颜色数据,并且解压缩从存储器读取的深度或颜色数据。压缩逻辑可以是利用多种压缩算法的一种或多种的无损压缩逻辑。由ROP 226执行的压缩的类型可以基于待压缩的数据的统计特性而变化。例如,在一个实施例中,Δ颜色压缩逐图块地对深度和颜色数据执行。

[0057] 在一些实施例中,ROP 226被包括在每个处理集群(例如,图2的集群214A至214N)内而非包括在分区单元220内。在这样的实施例中,经由存储器交叉开关216来传输针对像素数据的读和写请求而非像素片段数据。已处理的图形数据可在显示装置(比如,图1的一个或多个显示装置110中的一者)上显示、被路由以供由(多个)处理器102进一步处理、或被路由以供由图2A的并行处理器200内的处理实体中的一者进一步处理。

[0058] 图2C是根据实施例的并行处理单元内的处理集群214的框图。在一个实施例中,处理集群是图2的处理集群214A至214N中的一者的实例。处理集群214可以被配置成并行执行许多线程,其中,术语“线程”是指在一组特定的输入数据上执行的特定程序的实例。在一些实施例中,在不提供多个独立的指令单元的情况下,使用单指令多数据(SIMD)指令发布技术以支持对大量线程的并行执行。在其他实施例中,在使用共同指令单元的情况下,使用单指令多线程(SIMT)技术以支持对大量一般为同步的线程的并行执行,所述共同指令单元被配置成将指令发布到处理集群中的每一者内的一组处理引擎。不同于SIMD执行制度(其中,所有处理引擎通常执行相同的指令),SIMT执行允许不同的线程更容易沿着分歧的执行路径通过给定的线程程序。本领域技术人员将理解,SIMD处理制度表示SIMT处理制度的功能性子集。

[0059] 可以经由流水线管理器232来控制处理集群214的操作,所述流水线管理器将处理任务分布到SIMT并行处理器。流水线管理器232从图2的调度器210接收指令,并且经由图形多处理器234和/或纹理单元236来管理对那些指令的执行。所展示的图形多处理器234是SIMT并行处理器的示例性实例。然而,具有不同架构的各种类型的SIMT并行处理器可被包括在处理集群214内。图形多处理器234的一个或多个实例可以被包括在处理集群214内。图形多处理器234可以处理数据,并且数据交叉开关240可以用于将已处理的数据分布到多个可能的目的地(包括其他着色器单元)中的一者。流水线管理器232可以通过指定待经由数

据交叉开关240分布的已处理的数据的目的地来促进已处理的数据的分布。

[0060] 处理集群214内的每一个图形多处理器234都可以包括完全相同的一组功能执行逻辑(例如,算术逻辑单元、加载-存储单元等)。能以流水线方式配置功能执行逻辑,在流水线方式中,在先前的指令完成之前,可发布新指令。功能执行逻辑支持多种多样的操作,包括整数和浮点算术、比较操作、布尔操作、位移位和各种代数函数的计算。在一个实施例中,可以利用同一功能性单元硬件来执行不同的操作,并且可以存在功能单元的任何组合。

[0061] 传输至处理集群214的指令构成线程。跨一组并行处理引擎而执行的一组线程是线程组。线程组对不同的输入数据执行同一程序。可以将线程组内的每一个线程分配给图形多处理器234内的不同的处理引擎。线程组可包括比图形多处理器234内的处理引擎的数量少的线程。当线程组包括比处理引擎的数量少的线程时,处理引擎中的一个或多个在线程组正在被处理的周期期间可以是空闲的。线程组也可包括比图形多处理器234内的处理引擎的数量多的线程。当线程组包括比图形多处理器234内的处理引擎的数量多的线程时,处理可以在连续的时钟周期上执行。在一个实施例中,可在图形多处理器234上并发地执行多个线程组。

[0062] 在一个实施例中,图形多处理器234包括内部高速缓存存储器以执行加载和存储操作。在一个实施例中,图形多处理器234可以放弃内部高速缓存,并且使用处理集群214内的高速缓存存储器(例如,L1高速缓存308)。每个图形多处理器234还有权访问在所有处理集群214当中共享并且可用于在线程之间转移数据的分区单元(例如,图2的分区单元220A至220N)内的L2高速缓存。图形多处理器234还可访问芯片外全局存储器,所述芯片外全局存储器可以包括本地并行处理器存储器和/或系统存储器中的一者或多者。可将在并行处理单元202外部的任何存储器用作全局存储器。多个实施例(其中处理集群214包括图形多处理器234的多个实例)可以共享共同的指令和数据,这些指令和数据可存储在L1高速缓存308中。

[0063] 每个处理集群214可包括MMU 245(存储器管理单元),所述MMU被配置成将虚拟地址映射到物理地址中。在其他实施例中,MMU 245的一个或多个实例可驻留在图2的存储器接口218内。MMU 245包括:一组页表条目(PTE),用于将图块(tile)(更多地讨论分块(tiling))的虚拟地址映射到物理地址;以及可选地高速缓存行索引。MMU 245可包括可驻留在图形多处理器234或L1高速缓存或处理集群214内的地址转换后备缓冲器(TLB)或高速缓存。物理地址经处理以分布表面数据存取局部性,从而允许在分区单元当中实现高效的请求交错。高速缓存行索引可用于确定针对高速缓存行的请求是命中还是未命中。

[0064] 在图形和计算应用中,处理集群214可被配置成使得每个图形多处理器234耦合至纹理单元236以用于执行纹理映射操作,例如确定纹理样本位置、读取纹理数据和过滤纹理数据。根据需要,从内部纹理L1高速缓存(未示出)或在一些实施例中从图形多处理器234内的L1高速缓存读取纹理数据,并且从L2高速缓存、本地并行处理器存储器或系统存储器获取所述纹理数据。每个图形多处理器234将已处理的任务输出到数据交叉开关240以将已处理的任务提供给另一个处理集群214,以供进一步处理或以经由存储器交叉开关216将已处理的任务存储在L2高速缓存、本地并行处理器存储器或系统存储器中。preROP 242(例如,预光栅操作单元)被配置成从图形多处理器234接收数据、将数据导引到ROP单元,这些ROP单元可与如本文描述的分区单元(例如,图2的分区单元220A至220N)位于一起。preROP 242

单元可以执行针对颜色混合的优化、组织像素颜色数据和执行地址转换。

[0065] 将认识到的是,本文描述的核架构是展示性的,并且各种变型和修改是有可能的。任何数目的处理单元(例如,图形多处理器234、纹理单元236、preROP 242等)可被包括在处理集群214内。此外,虽然仅示出了一个处理集群214,但是如本文描述的并行处理单元可以包括处理集群214的任何数目的实例。在一个实施例中,每个处理集群214可以被配置成使用单独的且截然不同的处理单元、L1高速缓存等独立于其他处理集群214来操作。

[0066] 图2D示出了根据一个实施例的图形多处理器234。在这样的实施例中,图形多处理器234与处理集群214的流水线管理器232耦合。图形多处理器234具有执行流水线,包括但不限于:指令高速缓存252、指令单元254、地址映射单元256、寄存器堆258、一个或多个通用图形处理单元(GPGPU)核262和一个或多个加载/存储单元266。GPGPU核262和加载/存储单元266经由存储器和高速缓存互连268与高速缓存存储器272和共享存储器270耦合。

[0067] 在一个实施例中,指令高速缓存252从流水线管理器232接收待执行的指令流。这些指令被高速缓存在指令高速缓存252中,并且由指令单元254分派以供执行。指令单元254可以将指令分派为线程组(例如,线程束),其中线程组的每个线程被指派给GPGPU核262内的一不同执行单元。指令可以通过指定统一地址空间内的地址来访问本地、共享或全局地址空间中的任一者。地址映射单元256可以用于将统一地址空间中的地址转换成可以由加载/存储单元266访问的截然不同的存储器地址。

[0068] 寄存器堆258为图形多处理器324的功能单元提供一组寄存器。寄存器堆258为连接到图形多处理器324的功能单元(例如,GPGPU核262、加载/存储单元266)的数据路径的操作数提供临时存储。在一个实施例中,在这些功能单元中的每一者之间划分寄存器堆258,使得每个功能单元分配有寄存器堆258的专用部分。在一个实施例中,在由图形多处理器324执行的不同线程束之间划分寄存器堆258。

[0069] GPGPU核262可以各自包括浮点单元(FPU)和/或整数算术逻辑单元(ALU),这些FPU和整数ALU用于执行图形多处理器324的指令。根据实施例,GPGPU核262可以在架构上是类似的,或可以在架构上是不同的。例如且在一个实施例中,GPGPU核262的第一部分包括单精度FPU和整数ALU,而GPGPU核的第二部分包括双精度FPU。在一个实施例中,FPU可以针对浮点算术实现IEEE 754-2008标准,或可以实现可变精度浮点算术。图形多处理器324可以另外包括一个或多个固定功能或特殊功能单元以执行特定的功能(比如,复制矩形或像素混合操作)。在一个实施例中,GPGPU核中的一者或多者也可以包括固定或特殊功能逻辑。

[0070] 在一个实施例中,GPGPU核262包括能够对多组数据执行单条指令的SIMD逻辑。在一个实施例中,GPGPU核262可以物理地执行SIMD4、SIMD8和SIMD16指令,并且逻辑地执行SIMD1、SIMD2和SIMD32指令。用于GPGPU核的SIMD指令可以由着色器编译器在编译时生成,或者可以在执行为单程序多数据(SPMD)或SIMT架构编写和编译的程序时自动生成。为SIMT执行模型而配置的程序的多个线程可以经由单条SIMD指令而执行。例如,在一个实施例中,执行相同或类似操作的八个SIMT线程可以经由单个SIMD8逻辑单元并行地执行。

[0071] 存储器和高速缓存互连268是互连网络,该互连网络将图形多处理器234的功能单元中的每一个连接到寄存器堆258,并连接到共享存储器270。在一个实施例中,存储器和高速缓存互连268是交叉开关互连,该交叉开关互连允许加载/存储单元266在共享存储器270与寄存器堆258之间实现加载和存储操作。寄存器堆258能以与GPGPU核262相同的频率操

作,由此在GPGPU核262与寄存器堆258之间的数据传递是非常低等待时间的。共享存储器270可以用来实现在图形多处理器234内的功能单元上执行的线程之间的通信。高速缓存存储器272可以用作例如数据高速缓存,以便对功能单元与纹理单元236之间通信的纹理数据进行高速缓存。共享存储器270也可以用作程序管理的高速缓存。在GPGPU核262上执行的线程能以程序方式还将除了存储在高速缓存存储器272内的经自动高速缓存的数据之外的数据存储在共享存储器内。

[0072] 图3A到图3B展示了根据实施例的附加图形多处理器。所展示的图形多处理器325、350是图2C的图形多处理器234的变体。所展示的图形多处理器325、350可以被配置为能够同时执行大量执行线程的流传送多处理器(SM)。

[0073] 图3A示出了根据附加实施例的图形多处理器325。图形多处理器325相对于图2D的图形多处理器234包括执行资源单元的多个附加实例。例如,图形多处理器325可以包括指令单元332A至332B、寄存器堆334A-334B和纹理单元344A-344B的多个实例。图形多处理器325还包括多组图形或计算执行单元(例如,GPGPU核336A至336B、GPGPU核337A至337B、GPGPU核338A至338B)和多组加载/存储单元340A至340B。在一个实施例中,执行资源单元具有共同的指令高速缓存330、纹理和/或数据高速缓存存储器342以及共享存储器346。

[0074] 各种部件可以经由互连结构327通信。在一个实施例中,互连结构327包括一个或多个交叉开关以启用图形多处理器325的各种部件之间的通信。在一个实施例中,互连结构327是分开的高速网络结构层,图形多处理器325的每一个部件堆叠在该高速网络结构层上。图形多处理器325的部件经由互连结构327与远程部件通信。例如,GPGPU核336A-336B、337A-337B以及338A-338B可以各自经由互连结构327与共享存储器346通信。互连结构327可以仲裁图形多处理器325内的通信以确保部件之间的公平的带宽分配。

[0075] 图3B示出了根据附加实施例的图形多处理器350。图形处理器包括多组执行资源356A至356D,其中,每一组执行资源包括多个指令单元、寄存器堆、GPGPU核和加载存储单元,如图2D和图3A中所展示。执行资源356A至356D可以与纹理单元360A至360D一致地工作以进行纹理操作,同时共享指令高速缓存354和共享存储器362。在一个实施例中,执行资源356A至356D可以共享指令高速缓存354和共享存储器362以及纹理和/或数据高速缓存存储器358A至358B的多个实例。各种部件可以经由类似于图3A的互连结构327的互连结构352来通信。

[0076] 本领域技术人员将理解,图1、图2A至图2D以及图3A至图3B中所描述的架构就本实施例的范围而言是描述性的和非限制性的。因此,在不背离本文描述的实施例的范围的情况下,本文描述的技术可在任何正确配置的处理单元上实现,所述处理单元包括但不限于一个或多个移动应用处理器、一个或多个台式计算机或服务器中央处理单元(CPU)(包括多核CPU)、一个或多个并行处理单元(比如,图2的并行处理单元202)以及一个或多个图形处理器或专用处理单元。

[0077] 在一些实施例中,如本文描述的并行处理器或GPGPU通信地耦合至主机/处理器核以加速图形操作、机器学习操作、模式分析操作和各种通用GPU(GPGPU)功能。GPU可经由总线或其他互连(例如,比如PCIe或NVLink的高速互连)通信地耦合至主机处理器/核。在其他实施例中,GPU可集成在与这些核相同的封装或芯片上,并且经由内部的处理器总线/互连(即,在所述封装或芯片的内部)通信地耦合至这些核。不管连接GPU的方式如何,处理器核

都可用工作描述符中所包含的命令/指令序列的形式将工作分配给GPU。GPU然后使用专用的电路/逻辑来高效地处理这些命令/指令。

[0078] 用于GPU至主机处理器互连的技术

[0079] 图4A展示了示例性架构,其中多个GPU 410至413经由高速链路440至443(例如,总线、点对点互连等)通信地耦合至多个多核处理器405至406。在一个实施例中,取决于实现方式,高速链路440至443支持4GB/s、30GB/s、80GB/s或更高的通信吞吐量。可使用各种互连协议,包括但不限于PCIe 4.0或5.0以及NVLink 2.0。然而,本发明的基本原理并不限于任何特定的通信协议或吞吐量。

[0080] 另外,在一个实施例中GPU 410至413中的两者或更多者经由高速链路444至445互连,这些高速链路可使用与用于高速链路440至443的协议/链路相同或不同的协议/链路来实现。类似地,多核处理器405至406中的两者或更多者可经由高速链路433相连接,所述高速链路可以是以20GB/s、30GB/s、120GB/s或更高操作的对称多处理器(SMP)总线。可替代地,图4A中所示的各种系统部件之间的所有通信可使用相同的协议/链路(例如,经由共同的互连结构)来实现。然而,如所提到,本发明的基本原理并不限于任何特定类型的互连技术。

[0081] 在一个实施例中,每个多核处理器405至406分别经由存储器互连430至431通信地耦合至处理器存储器401至402,并且每个GPU 410至413分别经由GPU存储器互连450至453通信地耦合至GPU存储器420至423。存储器互连430至431以及450至453可利用相同或不同的存储器访问技术。通过示例的方式且不受限制地,处理器存储器401至402和GPU存储器420至423可以是易失性存储器,比如动态随机存取存储器(DRAM)(包括堆叠式DRAM)、图形DDR SDRAM(GDDR)(例如,GDDR5、GDDR6)或高带宽存储器(HBM),和/或可以是非易失性存储器,比如3D XPoint或纳米随机存取存储器。在一个实施例中,存储器的某一部分可以是易失性存储器,并且另一部分可以是非易失性存储器(例如,使用两级存储器(2LM)层级结构)。

[0082] 如下文所描述,虽然各种处理器405至406和GPU 410至413可以分别物理地耦合至特定的存储器401至402、420至423,但是可实现统一存储器架构,其中相同的虚拟系统地址空间(也称为“有效地址”空间)被分布在所有各个物理存储器当中。例如,处理器存储器401至402可各自包括64GB的系统存储器地址空间,并且GPU存储器420至423可各自包括32GB的系统存储器地址空间(在本示例中产生总共256GB的可寻址存储器)。

[0083] 图4B展示了根据一个实施例的针对多核处理器407与图形加速模块446之间的互连的附加细节。图形加速模块446可包括集成在线卡上的一个或多个GPU芯片,所述线卡经由高速链路440耦合至处理器407。可替代地,图形加速模块446可集成在与处理器407相同的封装或芯片上。

[0084] 所展示的处理器407包括多个核460A至460D,每个核具有转换后备缓冲器(translation lookaside buffer)461A至461D和一个或多个高速缓存462A至462D。这些核可包括用于执行指令和处理数据的各种其他部件,未展示这些部件以避免使本发明的基本原理模糊(例如,指令获取单元、分支预测单元、解码器、执行单元、重排序缓冲器等)。高速缓存462A至462D可包括1级(L1)和2级(L2)高速缓存。另外,一个或多个共享的高速缓存426可被包括在缓存层级结构中并且由多组核460A至460D共享。例如,处理器407的一个实施例

包括24个核,每个核具有其自身的L1高速缓存、12个共享的L2高速缓存和12个共享的L3高速缓存。在本实施例中,L2和L3高速缓存中的一者由两个邻近的核共享。处理器407和图形加速器集成模块446与系统存储器441连接,所述系统存储器可包括处理器存储器401至402。

[0085] 经由通过一致性总线464上的核间通信来保持存储在各种高速缓存462A至462D、456和系统存储器441中的数据和指令的一致性。例如,每个高速缓存可具有与其相关联的高速缓存一致性逻辑/电路以响应于检测到的至特定高速缓存行的读或写来经由一致性总线464进行通信。在一个实现方式中,经由一致性总线464来实现高速缓存监听协议,以监听高速缓存访问。高速缓存监听/一致性技术是为本领域技术人员所充分理解的,并且此处将不加以详细描述以避免使本发明的基本原理模糊。

[0086] 在一个实施例中,代理电路425将图形加速模块446通信地耦合至一致性总线464,从而允许图形加速模块446作为核的对等物来参与到高速缓存一致性协议中。特定地,接口435经由高速链路440(例如,PCIe总线、NVLink等)提供至代理电路425的连接性,并且接口437将图形加速模块446连接到链路440。

[0087] 在一个实现方式中,加速器集成电路436代表图形加速模块446的多个图形处理引擎431、432、N来提供高速缓存管理、存储器访问、上下文管理和中断管理服务。图形处理引擎431、432、N可各自包括单独的图形处理单元(GPU)。可替代地,图形处理引擎431、432、N可包括GPU内的不同类型的图形处理引擎,比如图形执行单元、媒体处理引擎(例如,视频编码器/解码器)、采样器和位块传输(blit)引擎。换句话说,图形加速模块可以是具有多个图形处理引擎431至432、N的GPU,或图形处理引擎431至432、N可以是集成在共同的封装、线卡或芯片上的个体GPU。

[0088] 在一个实施例中,加速器集成电路436包括存储器管理单元(MMU)439以用于执行各种存储器管理功能,比如虚拟至物理存储器转换(也称为有效至真实存储器转换)和用于访问系统存储器441的存储器访问协议。MMU 439还可包括转换后备缓冲器(TLB)(未示出)以用于将虚拟/有效缓存到物理/真实地址转换。在一个实现方式中,高速缓存438存储命令和数据以供由图形处理引擎431至432、N进行有效访问。在一个实施例中,存储在高速缓存438和图形存储器433至434、N中的数据与核高速缓存462A至462D、456以及系统存储器411保持一致。如所提到,这可经由代理电路425来实现,所述代理电路代表高速缓存438和存储器433至434、N来参与到高速缓存一致性机制中(例如,将与对处理器高速缓存462A至462D、456上的高速缓存行的修改/访问有关的更新发送到高速缓存438,和从高速缓存438接收更新)。

[0089] 一组寄存器445存储用于由图形处理引擎431至432、N执行的线程的上下文数据,并且上下文管理电路448管理线程上下文。例如,上下文管理电路448可在上下文切换期间执行保存和恢复操作以保存和恢复各种线程的上下文(例如,其中,第一线程被保存并且第二线程被存储,使得可以由图形处理引擎执行第二线程)。例如,在上下文切换时,上下文管理电路448可将当前寄存器值存储到存储器中的指定的区域(例如,由上下文指针标识)。然后,其可在返回到上下文时恢复这些寄存器值。在一个实施例中,中断管理电路447接收并处理从系统装置接收到的中断。

[0090] 在一个实现方式中,由MMU 439将来自图形处理引擎431的虚拟/有效地址转换成

系统存储器411中的真实/物理地址。加速器集成电路436的一个实施例支持多个(例如,4、8、16个)图形加速器模块446和/或其他加速器装置。图形加速器模块446可专用于在处理器407上执行的单个应用,或可在多个应用之间共享。在一个实施例中,呈现虚拟化图形执行环境,其中多个应用或虚拟机器(VM)共享图形处理引擎431至432、N的资源。这些资源可被再分为“切片(slice)”,这些切片基于与不同的VM和/或应用相关联的处理要求和优先权来被分配给这些VM和/或应用。

[0091] 因此,加速器集成电路起至图形加速模块446的系统的桥的作用,并且提供地址转换和系统存储器高速缓存服务。另外,加速器集成电路436可为主机处理器提供虚拟化设施,以管理图形处理引擎的虚拟化、中断和存储器管理。

[0092] 由于图形处理引擎431至432、N的硬件资源被显式地映射到由主机处理器407所见的真实地址空间,所以任何主机处理器都可以使用有效地址值来直接寻址这些资源。在一个实施例中,加速器集成电路436的一个功能是物理地分离图形处理引擎431至432、N,使得它们向系统呈现为独立的单元。

[0093] 如所提到,在所展示的实施例中,一个或多个图形存储器433至434、M分别耦合至图形处理引擎431至432、N中的每一者。图形存储器433至434、M存储由图形处理引擎431至432、N中的每一者处理的指令和数据。图形存储器433至434、M可以是易失性存储器,比如DRAM(包括堆叠式DRAM)、GDDR存储器(例如,GDDR5、GDDR6)或HBM,和/或可以是非易失性存储器,比如3D XPoint或纳米随机存取存储器(Nano-Ram)。

[0094] 在一个实施例中,为减少链路440上的数据业务,使用偏置技术以确保存储在图形存储器433至434、M中的数据是将被图形处理引擎431至432、N使用最频繁并且优选地不被核460A至460D使用(至少不是频繁地)的数据。类似地,偏置机制试图将由核(且优选地不是由图形处理引擎431至432、N)所需的数据保存在这些核的高速缓存462A至462D、456和系统存储器411内。

[0095] 图4C展示了另一个实施例,其中加速器集成电路436被集成在处理器407内。在本实施例中,图形处理引擎431至432、N经由接口437和接口435(再次,这些接口可利用任何形式的总线或接口协议)通过高速链路440来直接通信到加速器集成电路436。加速器集成电路436可执行与关于图4B所描述的操作相同的操作,但考虑到其与一致性总线462和高速缓存的462A至462D、426极为接近而潜在地处于更高的吞吐量。

[0096] 一个实施例支持不同的编程模型,包括专用进程编程模型(没有图形加速模块虚拟化)和共享的编程模型(具有虚拟化)。后者可包括由加速器集成电路436控制的编程模型和由图形加速模块446控制的编程模型。

[0097] 在专用进程模型的一个实施例中,图形处理引擎431至432、N在单一操作系统下专用于单个应用或进程。单个应用可以将其他应用请求汇集(funnel)到图形引擎431至432、N,从而在VM/分区内提供虚拟化。

[0098] 在专用进程编程模型中,可由多个VM/应用分区共享图形处理引擎431至432、N。共享的模型需要系统管理程序虚拟化图形处理引擎431至432、N,以允许由每个操作系统进行访问。针对无管理程序的单分区系统,图形处理引擎431至432、N由操作系统所拥有。在两种情况下,操作系统可以虚拟化图形处理引擎431至432、N以提供对每个进程或应用的访问。

[0099] 针对共享的编程模型,图形加速模块446或个别图形处理引擎431至432、N使用进

程句柄来选择进程要素。在一个实施例中,进程要素存储在系统存储器411中,并且可使用本文描述的有效地址至真实地址转换技术来寻址。进程句柄可以是在向图形处理引擎431至432、N登记其上下文(也就是说,调用系统软件以将进程要素添加到进程要素链表)时被提供给主机进程的特定于实现方式的值。进程句柄的较低的16位可以是进程要素链表内的进程要素的偏移。

[0100] 图4D展示了示例性加速器集成切片490。如本文所使用,“切片”包括加速器集成电路436的处理资源的指定部分。系统存储器411内的应用有效地址空间482存储进程要素483。在一个实施例中,响应于来自处理器407上执行的应用480的GPU调用481来存储进程要素483。进程要素483包含对应的应用480的进程状态。包含在进程要素483中的工作描述符(WD)484可以是由应用请求的单作业,或可包含指向作业队列的指针。在后一种情况下,WD 484是指向应用的地址空间482中的作业请求队列的指针。

[0101] 图形加速模块446和/或个别图形处理引擎431至432、N可以由系统中的所有进程或进程子集共享。本发明的实施例包括用于设置进程状态并将WD 484发送到图形加速模块446以在虚拟化环境中开始作业的基础设施。

[0102] 在一个实现方式中,专用进程编程模型是特定于实现方式的。在这个模型中,单进程拥有图形加速模块446或个别图形处理引擎431。由于图形加速模块446由单进程所拥有,所以在指派图形加速模块446的时候,管理程序针对拥有的分区初始化加速器集成电路436并且操作系统针对拥有的进程初始化加速器集成电路436。

[0103] 在操作中,加速器集成切片490中的WD获取单元491获取下一个WD 484,所述下一个WD包括待由图形加速模块446的图形处理引擎中的一者完成的工作的指示。来自WD 484的数据可存储在寄存器445中,并且由如所展示的MMU 439、中断管理电路447和/或上下文管理电路446使用。例如,MMU 439的一个实施例包括用于访问OS虚拟地址空间485内的段/页表486的段/页行走电路。中断管理电路447可处理从图形加速模块446接收的中断事件492。当执行图形操作时,由MMU 439将由图形处理引擎431至432、N产生的有效地址493转换为真实地址。

[0104] 在一个实施例中,针对每个图形处理引擎431至432、N和/或图形加速模块446复制一组相同的寄存器445,并且可由管理程序或操作系统来初始化这些寄存器。这些所复制的寄存器中的每一者可被包括在加速器集成切片490中。

[0105] 表1中示出了可由管理程序初始化的示例性寄存器。

[0106] 表1-管理程序初始化的寄存器

[0107]

1	切片控制寄存器
2	真实地址(RA)调度的进程区域指针
3	权限掩码覆盖寄存器(Authority Mask Override Register)
4	中断向量表条目偏移
5	中断向量表条目限制
6	状态寄存器
7	逻辑分区ID
8	真实地址(RA)管理程序加速器利用记录指针
9	存储描述寄存器

[0108] 表2中示出了可由操作系统初始化的示例性寄存器。

[0109] 表2-操作系统初始化的寄存器

[0110]

1	进程和线程标识
2	有效地址(EA)上下文保存/恢复指针
3	虚拟地址(VA)加速器利用记录指针
4	虚拟地址(VA)存储段表指针
5	权限掩码
6	工作描述符

[0111] 在一个实施例中,每个WD 484是特定于特定的图形加速模块446和/或图形处理引擎431至432、N的。其包含图形处理引擎431至432、N完成其工作所需的全部信息,或其可以是指向存储器位置(在所述存储器位置处,应用已设置了待完成的工作的命令队列)的指针。

[0112] 图4E展示了共享模型的一个实施例的附加细节。本实施例包括其中存储有进程要素列表499的管理程序真实地址空间498。管理程序真实地址空间498可经由管理程序496来访问,所述管理程序虚拟化用于操作系统495的图形加速模块引擎。

[0113] 共享的编程模型允许来自系统中的所有分区或分区子集的所有进程或进程子集使用图形加速模块446。存在两个编程模型,其中,图形加速模块446由多个进程和分区共享:时间切片共享和图形定向共享(graphics directed shared)。

[0114] 在这个模型中,系统管理程序496拥有图形加速模块446,并且使其功能可用于所有操作系统495。为使图形加速模块446支持由系统管理程序496进行的虚拟化,图形加速模块446可遵循以下要求:1)应用的作业请求必须是自主的(即,无需在作业之间保持状态),或图形加速模块446必须提供上下文保存和恢复机制。2)由图形加速模块446保证在指定的时间量内完成应用的作业请求(包括任何转换故障),或图形加速模块446提供抢占作业的处理的能力。3)当在定向共享的编程模型中操作时,必须保证图形加速模块446在进程之间的公平性。

[0115] 在一个实施例中,针对共享模型,需要应用480利用图形加速模块446类型、工作描述符(WD)、权限掩码寄存器(AMR)值和上下文保存/恢复区域指针(CSRP)来进行操作系统495系统调用。图形加速模块446类型描述了用于系统调用的目标加速度函数。图形加速模块446类型可以是特定于系统的值。WD专门针对图形加速模块446被格式化,并且可以呈图形加速模块446命令、指向用户定义的结构的有效地址指针、指向命令队列的有效地址指针或用于描述待由图形加速模块446完成的工作的任何其他数据结构的形式。在一个实施例中,AMR值是待用于当前进程的AMR状态。被传递到操作系统的值类似于设定AMR的应用。如果加速器集成电路436和图形加速模块446实现方式不支持用户权限掩码覆盖寄存器(UAMOR),那么操作系统可将当前UAMOR值应用于AMR值,之后在管理程序调用中传递AMR。可选地,管理程序496可应用当前权限掩码覆盖寄存器(AMOR)值,之后将AMR放到进程要素483中。在一个实施例中,CSRP是寄存器445中的一者,其包含在应用的地址空间482中的区域的有效地址以用于使图形加速模块446保存和恢复上下文状态。如果不需要在作业之间保存状态或当作业被抢占时,这个指针是可选的。上下文保存/恢复区域可以是固定的(pinned)系统存储器。

[0116] 在接收到系统调用时,操作系统495可验证应用480已注册并且已被给予使用图形加速模块446的权限。然后,操作系统495利用表3中所示的信息来调用管理程序496。

[0117] 表3-OS至管理程序调用参数

[0118]

1	工作描述符(WD)
2	权限掩码寄存器(AMR)值(潜在地被掩码)
3	有效地址(EA)上下文保存/恢复区域指针(CSRP)
4	进程ID(PID)和可选线程ID(TID)
5	虚拟地址(VA)加速器利用记录指针(AURP)
6	存储段表指针(SSTP)的虚拟地址
7	逻辑中断服务号(LISN)

[0119] 在接收到管理程序调用时,管理程序496验证操作系统495已注册并且已被给予使用图形加速模块446的权限。然后,管理程序496将进程要素483放入对应的图形加速模块446类型的进程要素链表中。进程要素可包括表4中所示的信息。

[0120] 表4-进程要素信息

[0121]

1	工作描述符（WD）
2	权限掩码寄存器（AMR）值（潜在地被掩码）
3	有效地址（EA）上下文保存/恢复区域指针（CSRP）
4	进程 ID（PID）和可选线程 ID（TID）
5	虚拟地址（VA）加速器利用记录指针（AURP）
6	存储段表指针（SSTP）的虚拟地址
7	逻辑中断服务号（LISN）
8	从管理程序调用参数导出的中断向量表
9	状态寄存器（SR）值
10	逻辑分区 ID（LPID）

[0122]

11	真实地址（RA）管理程序加速器利用记录指针
12	存储器描述符寄存器（SDR）

[0123] 在一个实施例中,管理程序初始化多个加速器集成切片490寄存器445。

[0124] 如图4F中所展示,本发明的一个实施例采用可经由共同的虚拟存储器地址空间寻址的统一存储器,所述共同的虚拟存储器地址空间用于访问物理处理器存储器401至402和GPU存储器420至423。在这种实现方式中,在GPU 410至413上执行的操作利用相同的虚拟/有效存储器地址空间来访问处理器存储器401至402且反之亦然,由此简化可编程性。在一个实施例中,虚拟/有效地址空间的第一部分被分配给处理器存储器401,第二部分被分配给第二处理器存储器402,第三部分被分配GPU存储器420,等。由此得以跨越处理器存储器401至402和GPU存储器420至423中的每一者来分布整个虚拟/有效存储器空间(有时称为有

效地址空间),从而允许任何处理器或GPU利用被映射到所述存储器的虚拟地址来访问任何物理存储器。

[0125] 在一个实施例中,在MMU 439A至439E中的一者或多者内的偏置/一致性管理电路494A至494E确保主机处理器(例如,405)与GPU 410至413的高速缓存器之间的高速缓存一致性,并且实现指示其中应存储有某些类型的数据的物理存储器的偏置技术。虽然图4F中展示了偏置/一致性管理电路494A至494E的多个实例,但是可在一个或多个主机处理器405的MMU内和/或在加速器集成电路436内实现偏置/一致性电路。

[0126] 一个实施例允许GPU附加存储器420至423被映射为系统存储器的一部分并且使用共享虚拟存储器(SVM)技术来访问,但不经受与完全系统高速缓存一致性相关联的典型性能缺陷。GPU附加存储器420至423被作为系统存储器来访问而无繁重的高速缓存一致性开销的这种能力为GPU卸载提供了有益的操作环境。这种安排允许主机处理器405软件设置操作数和访问计算结果,而没有传统I/O DMA数据复制的开销。这样的传统的复制涉及驱动器调用、中断和存储器映射I/O(MMIO)访问,它们相对于简单的存储器访问来说全部都是低效率的。同时,访问GPU附加存储器420至423而无高速缓存一致性开销的能力对于已卸载的计算的执行时间来说可以是关键的。在具有实质流传送写存储器业务的情况下,例如,高速缓存一致性开销可以显著减少由GPU 410至413所见的有效写带宽。操作数设置的效率、结果访问的效率和GPU计算的效率在确定GPU卸载的有效性中全部都起到一定的作用。

[0127] 在一个实现方式中,由偏置跟踪器数据结构来驱动在GPU偏置与主机处理器偏置之间的选择。可使用偏置表,例如,其可以是每GPU附加存储器页包括1或2个位的页粒度结构(即,被控制在存储器页的粒度下)。可以采用一个或多个GPU附加存储器420至423的被偷取的存储器范围来实现偏置表,其中在GPU 410至413中具有或不具有偏置高速缓存(例如,用于缓存偏置表的频繁使用/最近使用的条目)。可替代地,可将整个偏置表保持在GPU内。

[0128] 在一个实现方式中,在实际访问GPU存储器之前访问与每一次访问GPU附加存储器420至423相关联的偏置表条目,从而引起以下操作。首先,来自GPU 410至413的在GPU偏置中寻找其页的本地请求(这些本地请求发现它们的页处于GPU偏置)被直接转发到对应的GPU存储器420至423。来自GPU的本地请求(这些本地请求发现它们的页处于主机偏置)被转发到处理器405(例如,经由如上文所讨论的高速链路)。在一个实施例中,来自处理器405的在主机处理器偏置中寻找所请求的页的请求完成类似于正常存储器读取的请求。可替代地,可将针对GPU偏置页的请求转发到GPU 410至413。然后,如果GPU当前不使用页,那么其可将所述页转变到主机处理器偏置。

[0129] 可以由基于软件的机制、硬件辅助的基于软件的机制抑或针对一组有限的情况由纯粹基于硬件的机制来改变页的偏置状态。

[0130] 用于改变偏置状态的一个机制采用API调用(例如,OpenCL),所述API调用转而调用GPU的装置驱动器,所述装置驱动器转而发送消息(或为命令描述符排队)到GPU,从而指导其改变偏置状态并且针对一些转变在主机中执行高速缓存转储清除(cache flushing)操作。高速缓存转储清除操作对于从主机处理器405偏置转变到GPU偏置来说是需要的,但对于反向转变来说是不需要的。

[0131] 在一个实施例中,通过暂时渲染不可由主机处理器405缓存的GPU偏置页来保持高速缓存一致性。为了访问这些页,处理器405可请求来自GPU 410的访问,这取决于实现方式

可立即授予访问权或可不立即授予访问权。因此,为减少处理器405与GPU 410之间的通信,确保GPU偏置页是为GPU所需但非为主机处理器405所需(且反之亦然)的那些页是有利的。

[0132] 图形处理流水线

[0133] 图5展示根据实施例的图形处理流水线500。在一个实施例中,图形处理器可以实现所展示的图形处理流水线500。所述图形处理器可以被包括在如本文描述的并行处理子系统内,所述并行处理子系统为比如图2的并行处理器200,在一个实施例中,其是图1的(多个)并行处理器112的变体。各种并行处理系统可以经由如本文描述的并行处理单元(例如,图2的并行处理单元202)的一个或多个实例来实现图形处理流水线500。例如,着色器单元(例如,图3的图形多处理器234)可被配置成执行顶点处理单元504、曲面细分控制处理单元508、曲面细分评估处理单元512、几何处理单元516和片段/像素处理单元524中的一者或多者的功能。数据组装器502、图元组装器506、514、518、曲面细分单元510、光栅化器522和光栅操作单元526的功能也可由处理集群(例如,图3的处理集群214)内的其他处理引擎和对应的分区单元(例如,图2的分区单元220A至220N)执行。还可使用用于一个或多个功能的专用处理单元来实现图形处理流水线500。在一个实施例中,可以由通用处理器(例如,CPU)内的并行处理逻辑来执行图形处理流水线500的一个或多个部分。在一个实施例中,图形处理流水线500的一个或多个部分可以经由存储器接口528来访问芯片上存储器(例如,如图2中的并行处理器存储器222),所述存储器接口可以是图2的存储器接口218的实例。

[0134] 在一个实施例中,数据组装器502是收集表面和图元的顶点数据的处理单元。数据组装器502随后将包括顶点属性的顶点数据输出至顶点处理单元504。顶点处理单元504是可编程执行单元,该可编程执行单元执行顶点着色器程序,按顶点着色器程序所指定来照亮并变换顶点数据。顶点处理单元504读取存储在高速缓存、本地或系统存储器中的数据以供在处理顶点数据时使用,并且顶点处理单元504可被编程为将顶点数据从基于物体的坐标表示变换到世界空间坐标空间或归一化装置坐标空间。

[0135] 图元组装器506的第一实例从顶点处理单元504接收顶点属性。图元组装器506按照需要读取所存储的顶点属性,并且构建图形图元以供由曲面细分控制处理单元508处理。图形图元包括由各种图形处理应用编程接口(API)所支持的三角、线段、点、补片(patch)等等。

[0136] 曲面细分控制处理单元508将输入顶点视为用于几何补片的控制点。控制点是从来自补片的输入表示(例如,补片的基底)转换到适于由曲面细分评估处理单元512在表面评估中使用的表示。曲面细分控制处理单元508也可以计算几何补片的边缘的曲面细分因数。曲面细分因数应用于单个边缘,并且对与该边缘相关联的依赖于视图的细节等级进行量化。将曲面细分单元510配置成接收补片的边缘的曲面细分因数,并且将补片曲面细分为诸如线、三角或四边形图元的多个几何图元,多个几何图元被传输到曲面细分评估处理单元512。曲面细分评估处理单元512对经再分的补片的参数化坐标操作以生成与几何图元相关联的每一个顶点的表面表示和顶点属性。

[0137] 图元组装器514的第二实例从曲面细分评估处理单元512接收顶点属性,按照需要读取所存储的顶点属性,并且构建图形图元以供由几何处理单元516处理。几何处理单元516是可编程执行单元,该可编程执行单元执行几何着色器程序以便按几何着色器程序所指定来变换从图元组装器514接收的图形图元。在一个实施例中,将几何处理单元516编程

为将图形图元再分成一个或多个新图形图元,并且计算用于对新图形图元进行光栅化的参数。

[0138] 在一些实施例中,几何处理单元516可在几何流中增加或删除元素。几何处理单元516将指定新图形图元的参数和顶点输出到图元组装器518。图元组装器518从几何处理单元516接收参数和顶点,并且构建用于由视口缩放、拣选和剪辑单元520处理的图形图元。几何处理单元516读取存储在并行处理器存储器或系统存储器中的数据以供在处理几何数据时使用。视口缩放、拣选和剪辑单元520执行剪辑、拣选和视口缩放,并将经处理的图形图元输出到光栅化器522。

[0139] 光栅化器522可以执行深度拣选和其他基于深度的优化。光栅化器522也执行对新图形图元的扫描转换以生成片段,并且将那些片段和相关联的覆盖数据输出到片段/像素处理单元524。片段/像素处理单元524是被配置成执行片段着色器程序或像素着色器程序的可编程执行单元。片段/像素处理单元524按片段或像素着色器程序所指定来变换从光栅化器522接收的片段或像素。例如,可将片段/像素处理单元524编程为执行操作产生输出到光栅操作单元526的经着色的片段或像素,这些操作包括但不限于纹理映射、着色、混合、纹理校正和透视校正。片段/像素处理单元524可以读取存储在并行处理器存储器或系统存储器中的数据以供在处理片段数据时使用。可将片段或像素着色器程序配置成以取决于为处理单元配置的采样率的样本、像素、图块或其他粒度来着色。

[0140] 光栅操作单元526是处理单元,该处理单元执行包括但不限于模板印刷、z测试、混合等的光栅操作,并将像素数据作为经处理的图形数据输出,以存储在图形存储器(例如,如图2中的并行处理器存储器222和/或图1中的系统存储器104)中,显示在一个或多个显示装置110上,或者供由一个或多个处理器102或(多个)并行处理器112中的一个来进一步处理。在一些实施例中,将光栅操作单元526配置成压缩被写入到存储器的z或颜色数据,并且解压缩从存储器读取的z或颜色数据。

[0141] 包括静态场景确定、阻塞检测、帧率变换和调整压缩率的视频运动处理

[0142] 现在转向图6,所展示的是根据各种实施例的视频处理系统的概述。在至少一些实施例中,视频处理系统600包括运动检测器610、阻塞检测器620、压缩调整器630和编码器650。运动检测器610可从一个或多个相机接收输入(下文中更详细地讨论)。运动检测器610可确定从一个或多个相机接收的视频输入是否包括运动,即,接收到的帧内的物体的运动。基于帧是静态的还是移动的确定,运动检测器610可生成该物体的运动向量,并决定将该帧(包括运动向量)传递给阻塞检测器620(即,当在帧中检测到运动时)或者编码器650(即,当帧中没有检测到运动时)。阻塞检测器620可估计物体的运动以便执行帧率变换以使帧“平滑”。阻塞检测器620也可确定对帧内移动物体的估计是否呈现与另一个物体的阻塞(即,遮挡),并估计由移动物体呈现的阻塞。压缩调整器630可基于视频处理系统600的功率预算而调整帧率的压缩率。编码器650可对所接收的视频帧进行编码,并输出经编码的帧用于显示。在至少一些实施例中,视频处理系统600可通过避免在静态帧中编码运动、通过估计运动和阻塞并通过调整压缩率来节省编码、处理和功率带宽。

[0143] 继续参考图6和图7A-7B,这些图为根据实施例的运动检测解决方案。图7A展示了运动检测过程700的示例的图。所展示的元素702表示3D流水线。所展示的元素704表示从3D流水线接收视频信号,并确定所接收的帧中的场景是静态的或还是非静态(即,变化的)的。

运动检测可由例如运动检测器(例如,运动检测器610)中的应用接口(API)来执行。如果在704处为“否”,即,确定了场景不是静态的,并且场景中的物体是移动的,则将帧传递给表示阻塞检测器(例如,阻塞检测器620)的元素706。如果在704处为“是”,即,确定了场景是静态的,并且场景中没有物体在移动,则将帧传递给所展示的元素708。所展示的元素708表示接收有静态场景的帧,并确定帧的视口是否由于例如场景可能是静态的而用户可能正在观看场景的不同部分而已经改变。视口的改变可由运动检测器610确定。如果在708处为“是”,即,确定了视口已改变,则将帧传递给表示编码引擎(例如,编码器650)的所展示的元素710,用于帧的编码。如果在708处为“否”,即,确定了视口没有改变,则不对帧执行进一步的处理,并且过程继续到所展示的元素712并结束。系统700通过确定场景是静态的并且相机是移动的(即,视口已改变),系统避免计算运动(即,估计运动和阻塞),并将帧的编码传递给编码引擎,由此在编码引擎上节省功率。

[0144] 图7B示出根据实施例的检测运动的方法的示例。方法750可一般地实现为一组逻辑指令,这组逻辑指令存储在诸如随机存取存储器(RAM)、只读存储器(ROM)、可编程ROM(PROM)、闪存等的机器或计算机可读的存储介质中,存储在诸如例如可编程逻辑阵列(PLA)、现场可编程门阵列(FPGA)、复杂可编程逻辑器件(CPLD)的可配置逻辑中,存储在使用诸如例如数字信号处理器(DSP)、专用集成电路(ASIC)、互补金属氧化物半导体(CMOS)或晶体管-晶体管逻辑(TTL)技术、或其任何组合的电路技术的固定功能逻辑硬件中。

[0145] 所展示的处理框752提供了接收与一个或多个相机有关的视口信息。所展示的处理框754提供了确定相机是否是移动的。相机的移动可由运动检测器(例如,运动检测器610)确定。运动确定可基于例如视口改变、加速计或类似的运动传感器。所展示的处理框756提供了确定视口中的场景是否是静态的。场景中的移动的状态(即,静态的或移动的)可例如由例如运动检测器(例如,运动检测器610)中的应用接口(API)来确定。所展示的处理框758提供将视口的编码传递给编码器(例如,编码器650)。在至少一些实施例中,无需计算运动而将编码传递给编码器。过程750允许系统700在将帧的编码传递给编码引擎之前确定场景是静态的以及相机是移动的(即,视口已改变)以便由此避免计算运动(即,估计运动和阻塞)。系统700由此在编码引擎上节省功率。

[0146] 继续参考图6和8A-B,这些图为根据实施例示出对场景的阻塞检测过程的示例。图8A展示可由视频处理系统(例如,视频处理系统600)处理的帧的场景800。该场景可包括例如球810和人820。球810或另一物体可处于运动中(从左向右移动),并且人820或另一物体可基本上静止。场景800示出在时刻t处(左边)和在时刻t+1处(右边)的球。场景800可表示标准帧率,诸如例如,每秒30帧(30帧/秒)。然而,该标准帧率会在与移动物体(例如,球810)有关的帧中引入震颤,其中移动物体会显得以不均匀的方式移动。为了减少震颤并使物体的移动“平滑化”,系统600可通过例如增加帧显示率来执行帧率变换。可例如通过使帧显示率增加为两倍来增加帧显示率以使帧中的物体的移动“平滑化”。结果是,可将每秒30帧(30帧/秒)(如图8A中所展示)增加到每秒60帧(60帧/秒)(如图8B中所展示)。为了完成该帧率变换,系统600可估计现有帧之间的帧的运动(即,运动检测),即,估计处于时刻t+0.5处的新帧800,并在时刻t和t+1处的帧之间引入该新帧800。帧率变换可按需求持续地执行。在至少一些实施例中,帧率可进行调整,如下文中更详细地所讨论。相应地,新帧可包括时刻t、t+0.5以及t+1、t+1.5、t+2、t+2.5、t+3等。时刻t+0.5的新估计帧在图8B中示出,其中将(时刻

t+0.5处的)球810估计为已经移到(时刻t处的)第一帧位置与(时刻t+1处的)第二帧位置之间的中间。然而,随着球移动,由球810产生的阻塞区域改变。因此,视频处理系统600也可例如经由阻塞检测器629估计移动物体(即,球810)是处于前景(即,在人820前面)中还是处于背景(即,在人820后面)中。

[0147] 为了估计阻塞,视频处理系统600可利用二维(2D)到三维(3D)结构以基于包括例如输入相机的位置的若干因素来估计来确定基于2D取向的物体的结构。例如,估计可假定帧800的底部附近的移动接近相机且更可能对应于帧800的前景。类似地,例如,估计可假定帧800的顶部附近的移动更远离相机且更可能对应于天空。结果是,视频处理系统600可估计例如移动物体(即,球810)处于背景中的概率是百分之八十(80%)。因此,帧率变换和估计导致时刻t+0.5处的中间帧中的球810的运动被生成在人820后面(即,在背景中)。系统600可因此通过在帧率变换中利用2D到3D结构估计来减少帧中的移动物体的震颤(即,使移动"平滑化")以检测运动并更准确地估计移动物体的阻塞。

[0148] 图8C示出根据实施例的检测移动物体的阻塞的方法的示例。方法850可一般地实现为一组逻辑指令,该组逻辑指令存储在诸如RAM、ROM、PROM、闪存等的机器或计算机可读的存储介质中,存储在诸如例如PLA、FPGA、CPLD的可配置逻辑中,存储在使用诸如例如DSP、ASIC、CMOS或TTL技术的电路技术的固定功能逻辑硬件中,或存储在上述各项的任何组合中。

[0149] 所展示的处理框852提供了接收视频帧的运动向量。该运动向量可从运动检测器(例如,运动检测器610)接收。所展示的处理框854提供了增加视频的帧率。所展示的处理框856提供了检测视频的一个或多个帧内的物体的运动。所展示的处理框858提供了确定移动物体的阻塞。如上文所讨论,该阻塞可基于对移动物体的2D到3D结构估计来进行估计。过程850可因此使视频处理系统600能够通过以下方式来减少帧中的移动物体的震颤(即,使移动"平滑化"):在帧率变换中利用2D到3D结构估计以检测运动并更准确地估计移动物体的阻塞。

[0150] 继续参考图6以及图9A-B,这些图展示根据实施例展示的对场景的阻塞检测过程的示例的图。图9A展示与上文中参照图8A概述的过程类似的过程。场景900可包括可由视频处理系统(例如,视频处理系统600)处理的多个帧(例如,帧910、920和930)。场景900的每一个帧901、902和903可包括例如球910和人920。球910或其他物体可处于运动中(从左向右移动),并且人920或另一物体可基本上静止。帧901示出处于时刻n处(左边)和在时刻n+1处(右边)的球。场景900可表示标准帧率,诸如例如,每秒30帧(30帧/秒)。然而,该标准帧率会在与移动物体(例如,球910)有关的帧中引入震颤,其中移动物体会显得以不均匀的方式移动。为了减少震颤并使物体的移动"平滑化",系统600可通过例如增加帧显示率来执行帧率变换。可通过例如将帧显示速率增加为两倍以使帧中的物体的移动"平滑化"。结果是,可通过将内插的帧903包括在标准帧901与902之间来将每秒30帧(30帧/秒)增加到每秒60帧(60帧/秒)。为了完成该帧率变换,系统600可内插帧903以估计现有帧之间的球910的运动(即,运动检测),即,估计处于时刻n+0.5的新帧903,并将该新帧903内插在分别处于时刻n和n+1处的标准帧901与902之间。帧率变换和内插可按需求持续地执行。在至少一些实施例中,如下文中将更详细地所讨论,帧率可进行调整。相应地,新帧可包括时刻n、n+0.5以及n+1、n+1.5、n+2、n+2.5、n+3等。时刻n+0.5处的新估计帧被内插在帧903处,其中将(时刻n+0.5处

的)球910估计为已经移动到(处于时刻t处的)第一帧位置与(处于时刻n+1处的)第二帧位置之间的中间。然而,随着球910移动,由球910产生的阻塞区域改变。

[0151] 为了估计阻塞,视频处理系统600可经由阻塞检测器620估计并内插由球910产生的阻塞。视频处理系统600也可包括来自深度传感器(未示出)的输入,以提供移动物体(即,球910)的深度,该深度传感器例如为立体相机或与捕获帧901和902的相机通信的立体输入。深度输入允许视频处理系统600确定移动物体是处于前景(即,在人920前面)中还是处于背景(即,在人920后面)中。为了执行估计,视频处理系统600可利用深度输入来确定要将移动物体(即,球910)内插什么深度。深度输入可因此帮助通知视频处理系统600关于当执行帧率变换时相对于其他物体将在哪里内插移动物体,即,球910是在人920前面还是后面。相应地,帧率变换和内插过程导致球910的运动被内插在时刻n+0.5处的中间帧中在人920后面。系统600可因此通过在帧率变换中利用深度相机检测运动和物体的深度而使得由物体产生的阻塞可以被内插在中间帧中来减少移动物体的震颤(即,使移动"平滑化")。

[0152] 图9B示出根据实施例的检测移动物体的阻塞的方法的示例。方法950可一般地实现为一组逻辑指令,这组逻辑指令存储在诸如RAM、ROM、PROM、闪存等的机器或计算机可读的存储介质中,存储在诸如例如PLA、FPGA、CPLD的可配置逻辑中,存储在使用诸如例如DSP、ASIC、CMOS或TTL技术的电路技术的固定功能逻辑硬件中,或存储在上述各项的任何组合中。

[0153] 所展示的处理框952提供了接收视频的运动向量。运动向量可从运动检测器(例如,运动检测器610)接收。所展示的处理框954提供了增加视频的帧率。所展示的处理框956提供了检测视频的一个或多个帧内的物体的运动。该运动可由运动检测器(例如,运动检测器610)检测。所展示的处理框958提供了基于来自一个或多个深度相机的输入来确定移动物体的深度。所展示的处理框960提供了估计中间帧中的移动物体的阻塞。该阻塞可由阻塞模块(例如,阻塞检测器620)估计。所展示的处理框962提供了在帧率变换过程中内插中间帧。过程950可因此使视频处理系统600能够通过以下方式来减少帧中移动物体的震颤(即,使移动"平滑化"):在帧率变换中利用2D到3D结构估计来以检测运动并更准确地估计移动物体的阻塞。

[0154] 继续参考图6和10A-B,根据实施例示出视频压缩系统的示例。图10A展示包括编码器1010和帧率模块1020的可调整压缩率反馈环路1000的示例的图。反馈环路1000可在具有压缩调整器和编码器的视频处理系统(例如,包括压缩调整器630和编码器650的视频处理系统600)中实现。反馈环路1000可基于接收到的帧率,并且可以是可调整的。编码器1010可从3D流水线接收一个或多个帧以进行编码。编码器1010也可接收进入系统的功率预算输入以确保系统处于功率预算内。可调整反馈环路1000以将帧率与功率预算匹配,从而确保基于功率预算的最佳压缩率。例如,如果帧率降低,则反馈环路1000可降低压缩率来匹配帧率,即,在检测到帧率的降低之后,帧率模块1020可向编码器1010提供反馈输入以实现功率预算匹配。结果是,反馈环路1000用于节省处理功率。反馈环路1000可在包括例如图形处理器的硬件或软件中实现。基于前述内容,反馈环路1000帮助确保编码器(例如,编码器650)在所提供的功率预算内操作以便由此节省功率。

[0155] 图10B示出了根据实施例的检测移动物体的阻塞的方法的示例。方法1050可一般地实现为一组逻辑指令,这组逻辑指令存储在诸如RAM、ROM、PROM、闪存等的机器或计算机

可读的存储介质中,存储在诸如例如PLA、FPGA、CPLD的可配置逻辑中,存储在使用诸如例如DSP、ASIC、CMOS或TTL技术的电路技术的固定功能逻辑硬件中,或存储在上述各项的任何组合中。

[0156] 所展示的处理框1052提供了接收一个或多个视频帧以进行编码。如本文中所讨论,这一个或多个视频帧可从例如3D图形流水线接收。所展示的处理框1054提供了接收编码引擎的功率预算。编码引擎可以是编码器,例如,编码器1010或编码器650,并且功率预算可以是进入系统600的输入以确保系统在所定义的功率限制内操作。所展示的处理框1056提供了在编码引擎内确定反馈环路的帧率。该帧率可由帧率模块(例如,帧率模块1020)确定。帧率模块也可向编码器1010提供反馈以向编码器1010供应压缩率调整并由此完成反馈环路1000。所展示的处理框1058提供调整帧率压缩率以确保符合功率预算。帧率压缩率可由压缩调整器(例如,压缩调整器630)实现。编码器1010和帧率模块1020由此用于确保帧率与编码器功率要求之间的平衡,并且帮助确保压缩率在所定义的功率预算内执行。反馈环路1000由此用于确保功率预算符合性并节省编码功率。

[0157] 显示技术

[0158] 现在转向图11,示出性能增强的计算系统1100。在所展示的示例中,处理器1110耦合至显示器1120。处理器1110一般可生成将显示在显示器1120的LCD面板1150上的图像。在一个示例中,处理器1110包括通信接口,诸如例如,视频图形阵列(VGA)、显示端口(DP)接口、嵌入式显示端口(eDP)接口、高清晰度多媒体接口(HDMI)、数字视觉接口(DVI)等等。处理器1110可以是处理图形数据并生成显示在LCD面板1150上的图像(例如,视频帧、静止图像)的图形处理器(例如,图形处理单元/GPU)。此外,处理器1110可包括生成像素数据的一个或多个图像处理流水线。图像处理流水线可符合OPENGL架构或其他适合的架构。另外,处理器1110可连接到主机处理器(例如,中央处理单元/CPU),其中主机处理器执行控制处理器1100和/或与处理器1110交互的一个或多个设备驱动器。

[0159] 所展示的显示器1120包括定时控制器(TCON)1130,该定时控制器1130可对LCD面板1150上的不同像素单独地寻址,并且逐刷新周期地更新LCD面板1150上的每一个单独像素。就此而言,LCD面板1150可包括多个液晶元件,诸如例如,液晶和集成的颜色过滤器。LCD面板1150的每一个像素都可包括分别具有红色、绿色和蓝色过滤器的液晶元件三元组。LCD面板1150能以二维(2D)阵列安排像素,该二维阵列经由行驱动器1152和列驱动器1154控制以更新正由LCD面板1150显示的图像。因此,TCON 1130可驱动行驱动器1152和列驱动器1154以对LCD面板1150的特定像素寻址。TCON 1130也可调整提供给像素中的液晶元件的电压以改变穿过三个液晶元件中的每一个的光强度,并因此改变显示在LCD面板1150的表面上的像素的颜色。

[0160] 背光1160可包括安排在LCD面板1150的边缘处的多个发光元件,诸如例如,发光二极管(LED)。相应地,由LED生成的光可由散射器(未示出)分散通过LCD面板1150。在另一示例中,以一配置将LED安排在LCD面板1150正后方的2D阵列中,由于每个LED使光分散通过定位在该LED前方的LCD面板1150的一个或多个对应像素,因此,该配置有时称为直接背光。发光元件也可包括沿LCD面板1150的一个或多个边缘安排的紧凑型荧光灯(CFL)。为了消除多个边缘,可更改边缘的组合以实现对区域的选择性照明,其中以较少的功率使用少于整组照明元件。

[0161] 发光元件也可包括放置在LCD面板1150后面的一个或多个电致发光材料的薄片。在此类情况下,来自薄片的表面的光可分散通过LCD面板1150的像素。此外,可将薄片划分成多个区域,诸如例如,象限。在一个示例中,单独控制每一个区域以仅照亮LCD面板1150的一部分。也可使用其他背光解决方案。

[0162] 所展示的显示器1120也包括将电压提供给背光1160的发光元件的背光控制器(BLC)1140。例如,BLC 1140可包括脉宽调制(PWM)驱动器(未示出)以生成激活背光1160的发光元件的至少一部分的PWM信号。PWM信号的占空比和频率可使由发光元件生成的光变暗。例如,100%占空比可对应于发光元件完全开启,而0%占空比可对应于发光元件完全关闭。因此,中间级占空比(例如,25%、50%)通常使发光元件被开启达循环周期的、与占空比的百分比成比例的部分。该循环周期可以足够快,使得发光元件的闪烁对人眼是不可察觉的。此外,对用户的影响可能在于,由背光1160发出的光的级别比背光1160完全激活时要低。BLC 1140可与TCON 1130分离,或者可合并进TCON 1130中。

[0163] 可替代地,可使用发射性显示系统,其中LCD面板1150将被发射性显示面板(例如,有机发光二极管/OLED)取代,背光1160将被省略,并且行驱动器1152和列驱动器1154可分别用于直接对像素颜色和亮度调制。

[0164] 基于距离的显示分辨率

[0165] 图12A示出其中用户1218与包含显示单元1228的数据处理装置1200交互的场景。显示处理装置1200可包括例如,笔记本计算计、台式计算机、平板计算机、可转换平板、移动网际装置(MID)、个人数字助理(PDA)、可穿戴装置(例如,头戴式显示器/HMD)、媒体播放器等,或上述各项的任何组合。所展示的数据处理装置1200包括耦合至存储器1222的处理器1224(例如,嵌入式控制器、微控制器、主机处理器、图形处理器),存储器1222可包括通过处理器1224可寻址的存储位置。如将更详细地所讨论,距离传感器1210可启用相对于显示单元1228的、基于距离的显示分辨率。

[0166] 所展示的存储器1222包括将在显示单元1228上渲染的显示数据1226。在一个示例中,处理器1224在将显示数据1226呈现在显示单元1228上之前,执行对显示数据1226的数据转换。后处理引擎1214可在处理器1224上执行以接收显示数据1226和距离传感器1210的输出。后处理引擎1214可修改显示数据1226以增强显示单元1228上的屏幕内容的可读性,降低数据处理装置1200中的功耗,等等,或可进行上述操作的任何组合。

[0167] 所展示的存储器1222除了操作系统1212和应用1220之外还存储显示分辨率设置1216。显示分辨率设置1216可指定将沿长度维度和宽度维度在显示单元1228上呈现的显示数据1226的像素数量。如果如由应用1220生成的显示数据1226与显示单元1228的格式不兼容,则处理器1224可配置显示数据1226的缩放比例以匹配显示单元1228的格式。就此而言,显示分辨率设置1216可与定义显示单元1228的其他设置的配置数据相关联和/或可合并入该配置数据。此外,可在单位距离或面积(例如,每英寸像素/PPI)或其他合适的参数方面对显示分辨率设置1216进行定义。

[0168] 应用1220可生成用户界面,其中用户1218可与用户界面交互以从通过该用户界面提供的一个或多个选项中选择显示分辨率设置1216,将显示分辨率设置1216作为所请求的值键入,等等。因此,可在渲染在显示单元1228上之前调整显示数据1226的大小以适配显示分辨率设置1216。

[0169] 距离传感器1210可跟踪用户1218与显示单元1228之间的距离,其中距离感测可通过与数据处理装置1200/显示单元1228相关联的物理按钮、通过由应用1220和/或操作系统1220的加载所提供的用户界面等等而触发。例如,在数据处理装置1200的引导期间,操作系统1212可执行自动进程以触发背景或前景中的距离感测。距离感测可定期地或持续性地进行。

[0170] 图12B示出距离感测场景的一个示例。在所展示的示例中,距离传感器1210使用收发机1208以在用户1218的方向上发射电磁波束1202。因此,收发机1202可定位在数据处理装置1200(图12A)的前向表面上。电磁波束1202可影响用户1218,并且可作为返回电磁波束1204从用户1218反射/散射。返回电磁波束1204可由例如处理器1224(图12A)和/或后处理引擎1214(图12A)分析以确定用户1218与显示单元1228(图12A)之间的距离1206。距离1206可用于调整显示分辨率设置1216。

[0171] 显示层

[0172] 现在转向图13,示出显示系统1300,其中级联的显示层1361、1362和1363用于实现显示组件1360中的空间/时间超分辨率。在所展示的示例中,处理器1310经由总线1320向系统1300提供原始图形数据1334(例如,视频帧、静止图像)。级联的显示程序1331可存储在存储器1330中,其中级联的显示程序1331可以是与显示组件1360相关联的显示驱动器的一部分。所展示的存储器1330也包括原始图形数据1334和已分解的图形数据1335。在一个示例中,级联的显示程序1331包括时间分解分量1332和空间分解分量1333。时间分解分量1332可执行时间分解计算,而空间分解分量可执行空间分解计算。级联的显示程序331可基于用户配置和原始图形数据1334而导出经分解的图形数据1335用于在每一个显示层1361、1362和1363上呈现。

[0173] 显示组件1360可实现为用在例如头戴式显示器(HMD)应用中的LCD(液晶显示器)。更具体地,显示组件1360可包括LCD面板、接口板、透镜附件等的堆栈。每一个面板都能以例如1280*1280的原生分辨率并以60Hz刷新率操作。可使用其他原生分辨率、刷新率、显示面板技术和/或层配置。

[0174] 多个显示单元

[0175] 图14示出了图形显示系统1400,该图形显示系统1400包括一组显示单元1430(1430a-1430n),这组显示单元1430一般可用于输出宽屏(例如,全景)演示1440,该宽屏演示1440包括有聚合力的且结构化的拓扑形式的协调内容。在所展示的示例中,数据处理装置1418包括处理器1415,该处理器1415将逻辑功能1424应用于通过网络1420从一组显示单元1430接收的硬件配置文件数据1402。当未发现硬件配置文件数据与硬件配置文件查找表1412中的一组设置的匹配时,将逻辑功能1424应用于硬件配置文件数据1402可创建一组自动拓扑设置1406。所展示的一组自动拓扑设置1406从显示处理装置1418通过网络1420被传输到显示单元1430。

[0176] 处理器1415可在从显示驱动器1410接收到逻辑功能1424之后执行并运行逻辑功能1424。就此而言,显示驱动器1410可包括自动拓扑模块1408,该自动拓扑模块1408自动地配置并构建显示单元1432的拓扑以创建演示1440。在一个示例中,显示驱动器1410是一组指令,这组指令当由处理器1415执行时使数据处理装置1418与显示单元1430、视频卡等通信,并执行自动拓扑生成操作。

[0177] 数据处理装置1418可包括例如,服务器、台式机、笔记本计算机、平板计算机、可转换平板、MID、PDA、可穿戴装置、媒体播放器等等。因此,显示处理装置1418可包括硬件控制模块1416、存储装置1414、随机读取存储器(RAM,未示出)、包括一个或多个视频控制器卡的控制器卡,等等。在一个示例中,显示单元1430是彼此协同以产生演示1440的平板显示器(例如,液晶、有源矩阵、等离子体等)、HMD、视频投影装置,等等。此外,演示1440可基于存储在存储装置1414中的媒体文件而生成,其中媒体文件可包括例如,电影、视频剪辑、动画、广告等、或上述各项的任何组合。

[0178] 可将术语“拓扑”认为是第一显示单元1430a、第二显示单元1430b、第三显示单元1430n等的数量、缩放、形状和/或其他配置参数。相应地,显示单元1430的拓扑可使得演示1440能够被一致地视觉地呈现,使得演示1440的各个段是与正通过显示单元1430播放的媒体的原始尺度和范围成比例且兼容的。因此,拓扑可构成不受在演示1440中渲染的内容的形状或大小的持续变化影响的空间关系和/或几何属性。在一个示例中,自动拓扑模块1408包括定时模块1426、控制模块1428、信号监视器模块1432和信号显示模块1434。定时模块1426可将一组显示单元1430中的特定显示单元指定为样本显示单元。在此类情况下,定时模块1426可将剩下的显示模块1430指定为附加的显示单元。在一个示例中,定时模块1426自动地将形状因数设置成与硬件配置文件数据1402兼容,其中演示1440是由图形信号的序列1422自动发起的。

[0179] 在一个示例中,控制模块1428修改一组自动拓扑设置1406。此外,信号监视器模块1432可自动地监视图形信号的序列1422,并且触发存储装置1414将一组自动拓扑设置1406与硬件配置文件查找表1412相关联。此外,信号监视器模块1432可根据一组变化标准自动地检测一组显示单元1430中的改变,并且自动地生成对应于一组显示单元1430中的改变的新拓扑配置文件。由此,可将新拓扑配置文件应用于一组显示单元1430。如果图形信号的序列1422不能满足一组准则,则信号监视器模块1432也可触发信号显示模块1434重应用一组自动拓扑设置1406。如果硬件配置文件数据1402不支持图形信号的序列1422的自动拓扑显示,则数据处理装置1418可报告错误,并在错误日志1413中记录该错误。

[0180] 云辅助的媒体递送

[0181] 现在转向图15,云游戏系统1500包括通过网络1510耦合至服务器1520的客户端1540。客户端1540一般可以是在服务器1520上被容纳、处理和渲染的图形(例如,游戏、虚拟现实/VR、增强现实/AR)内容的消费方。所展示的可缩放的服务器1520具有同时(例如,通过利用并行的和分摊的处理和渲染资源)向多个客户端提供图形内容的容量。在一个示例中,服务器1520的可缩放性由网络1510的容量限制。相应地,可以存在某个客户端的阈值数量,超过该阈值数量,则对所有客户端的服务都降级。

[0182] 在一个示例中,服务器1520包括图形处理器(例如,GPU)1530、主机处理器(例如,CPU)1524和网络接口卡(NIC)1552。NIC 1522可从客户端1540接收对于图形内容的请求。来自客户端1540的请求可导致图形内容经由在主机处理器1524上执行的应用从存储器被获取。主机处理器1524可执行高级操作,诸如例如,在给定场景中确定物体的位置、碰撞和运动。基于这些高级操作,主机处理器1524可生成与场景数据组合并由图形处理器1530执行的渲染命令。渲染命令可使图形处理器1530对于将经由客户端1540呈现的场景定义场景几何、着色、照明、运动、纹理、相机参数等。

[0183] 更具体地,所展示的图形处理器1530包括图形渲染器1532,该图形渲染器1532根据由主机处理器1524生成的渲染命令来执行渲染过程。图形渲染器1532的输出可以是提供给帧捕获器1534的原始视频帧流。所展示的帧捕获器1534耦合至编码器1536,该编码器1536可压缩/格式化原始视频流以在网络1510上传输。编码器1536可使用各种视频压缩算法,诸如例如,来自国际电信联盟电信标准化部门(ITUT)的H.264标准、来自国际标准化组织/国际电工委员会(ISO/IEC)的MEPG4高级视频编码(AVC)标准,等等。

[0184] 所展示的客户端1540(其可以是台式计算机、笔记本计算机、平板计算机、可转换计算机、可穿戴装置、MID、PDA、媒体播放器等)包括NIC 1542以从服务器1520接收所传输的视频流。NIC 1522可包括物理层和客户端1540中的网络接口的软件层的基础以促进网络1510上的通信。客户端1540也可包括采用与编码器1536相同的格式化/压缩方案的解码器1544。因此,解压缩的视频流可从解码器1544提供给视频渲染器1546。所展示的视频渲染器1546耦合至视觉地呈现图形内容的显示器1548。

[0185] 如已经记录的,图形内容可包括游戏内容。就此而言,客户端1540可执行实时交互式串流,该实时交互式串流涉及从输入装置1550收集用户输入以及经由网络1510将用户输入递送到服务器1520。云游戏的这种实时交互式部分会提出关于等待时间的挑战。

[0186] 附加的系统概述示例

[0187] 图16是根据实施例的处理系统1600的框图。在各种实施例中,系统1600包括一个或多个处理器1602和一个或多个图形处理器1608,并且可以是单处理器台式计算机系统、多处理器工作站系统或具有大量处理器1602或处理器核1607的服务器系统。在一个实施例中,系统1600是包括在系统芯片(SoC)中的处理平台以供用于移动装置、手持式装置或嵌入式装置中。

[0188] 系统1600的实施例可以包括以下各者或可以包括在以下各者内: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。在一些实施例中,系统1600是移动电话、智能电话、平板计算装置或移动互联网装置。数据处理系统1600还可以包括以下各者、与以下各者耦合或被集成在以下各者中:穿戴式装置,比如智能手表穿戴式装置、智能眼镜装置、增强现实装置或虚拟显示装置。在一些实施例中,数据处理系统1600是电视或机顶盒装置,其具有一个或多个处理器1602和由一个或多个图形处理器1608产生的图形接口。

[0189] 在一些实施例中,一个或多个处理器1602各自包括用于处理指令的一个或多个处理器核1607,这些指令在被执行时执行系统和用户软件的操作。在一些实施例中,一个或多个处理器核1607中的每一者被配置成处理具体的指令集1609。在一些实施例中,指令集1609可以促进复杂指令集计算(CISC)、精简指令集计算(RISC)、或经由超长指令字(VLIW)的计算。多个处理器核1607可以各自处理不同的指令集1609,所述指令集可以包括用于促进对其他指令集进行仿真的指令。处理器核1607还可包括其他处理装置,比如数字信号处理器(DSP)。

[0190] 在一些实施例中,处理器1602包括高速缓存存储器1604。取决于架构,处理器1602可以具有单个内部高速缓存或多级内部高速缓存。在一些实施例中,在处理器1602的各种部件当中共享高速缓存存储器。在一些实施例中,处理器1602还使用外部高速缓存(例如,3级(L3)高速缓存或最后一级高速缓存(LLC))(未示出),可使用已知的高速缓存一致性技术

在处理器核1607之间共享所述外部高速缓存。寄存器堆1606被另外包括在处理器1602中，所述寄存器堆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、状态寄存器和指令指针寄存器)。一些寄存器可以是通用寄存器,而其他寄存器可以是特定于处理器1602的设计的。

[0191] 在一些实施例中,处理器1602耦合至处理器总线1610,以在处理器1602与系统1600中的其他部件之间传输通信信号(比如,地址、数据或控制信号)。在一个实施例中,系统1600使用示例性‘中枢’系统架构,包括存储器控制器中枢1616和输入输出(I/O)控制器中枢1630。存储器控制器中枢1616促进存储器装置与系统1600的其他部件之间的通信,而I/O控制器中枢(ICH)1630经由本地I/O总线来提供至I/O装置的连接。在一个实施例中,存储器控制器中枢1616的逻辑被集成在处理器内。

[0192] 存储器装置1620可以是动态随机存取存储器(DRAM)装置、静态随机存取存储器(SRAM)装置、闪存装置、相变存储器装置或具有合适的性能以充当进程存储器的某一其他存储器装置。在一个实施例中,存储器装置1620可以作为系统1600的系统存储器来操作,以存储数据1622和供在一个或多个处理器1602执行应用或进程时使用的指令1621。存储器控制器中枢1616也与可选的外部图形处理器1612耦合,所述外部图形处理器可与处理器1602中的图形处理器1608耦合,以执行图形和媒体操作。

[0193] 在一些实施例中,ICH 1630使得能够经由高速I/O总线将外围装置连接到存储器装置1620和处理器1602。I/O外围装置包括但不限于:音频控制器1646、固件接口1628、无线收发机1626(例如,Wi-Fi、蓝牙)、数据存储装置1624(例如,硬盘驱动器、闪存等)和用于将传统(例如,个人系统2(PS/2))装置耦合至系统的传统I/O控制器1640。一个或多个通用串行总线(USB)控制器1642连接输入装置(比如,键盘和鼠标1644组合)。网络控制器1634还可以耦合至ICH 1630。在一些实施例中,高性能网络控制器(未示出)耦合至处理器总线1610。将认识到的是,所示出的系统1600是示例性的而非限制性的,因为还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,I/O控制器中枢1630可以集成在所述一个或多个处理器1602内,或者存储器控制器中枢1616和I/O控制器中枢1630可以集成在分立式外部图形处理器(比如外部图形处理器1612)内。

[0194] 图17是处理器1700的实施例的框图,所述处理器具有一个或多个处理器核1702A至1702N、集成式存储器控制器1714和集成式图形处理器1708。图17中具有与本文任何其他图的元件相同的参考数字(或名称)的那些元件可以以与本文别处描述的方式类似的任何方式来操作或起作用,但并不限于此。处理器1700可包括多达且包括由虚线框表示的附加核1702N的附加核。处理器核1702A至1702N中的每一者包括一个或多个内部高速缓存单元1704A至1704N。在一些实施例中,每个处理器核还能够访问一个或多个共享高速缓存单元1706。

[0195] 内部高速缓存单元1704A至1704N和共享高速缓存单元1706表示处理器1700内的高速缓存存储器层级结构。高速缓存存储器层级结构可包括每个处理器核内的至少一级指令和数据高速缓存以及共享中间级高速缓存的一个或多个级(比如,2级(L2)、3级(L3)、4级(L4)或其他级高速缓存),其中,在外部存储器前面的最高级高速缓存被归类为LLC。在一些实施例中,高速缓存一致性逻辑保持各种高速缓存单元1706和1704A至1704N之间的一致性。

[0196] 在一些实施例中,处理器1700还可包括一组一个或多个总线控制器单元1716以及系统代理核1710。所述一个或多个总线控制器单元1716管理一组外围总线,比如一个或多个外围部件互连总线(例如,PCI、PCI快速总线)。系统代理核1710提供对各处理器部件的管理功能。在一些实施例中,系统代理核1710包括一个或多个集成式存储器控制器1714,所述集成式存储器控制器用于管理对各种外部存储器装置(未示出)的访问。

[0197] 在一些实施例中,处理器核1702A至1702N中的一者或多者包括对同时多线程处理的支持。在这样的实施例中,系统代理核1710包括用于在多线程处理期间协调和操作核1702A至1702N的部件。系统代理核1710可另外包括功率控制单元(PCU),所述PCU包括用于调节处理器核1702A至1702N和图形处理器1708的功率状态的逻辑和部件。

[0198] 在一些实施例中,处理器1700另外包括用于执行图形处理操作的图形处理器1708。在一些实施例中,图形处理器1708与一组共享高速缓存单元1706和系统代理核1710包括一个或多个集成式存储器控制器1714耦合。在一些实施例中,显示控制器1711与图形处理器1708耦合以便将图形处理器输出驱动到一个或多个耦合的显示器。在一些实施例中,显示控制器1711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1708或系统代理核1710内。

[0199] 在一些实施例中,使用基于环形的互连单元1712来耦合处理器1700的内部部件。然而,可以使用替代性互连单元,比如点到点互连、切换式互连、或其他技术,包括本领域众所周知的技术。在一些实施例中,图形处理器1708经由I/O链路1713与环形互连1712耦合。

[0200] 示例性I/O链路1713表示多种I/O互连中的至少一者,包括促进各种处理器部件与高性能嵌入式存储器模块1718(比如eDRAM模块)之间的通信的封装(on package)I/O互连。在一些实施例中,处理器核1702至1702N中的每一者和图形处理器1708将嵌入式存储器模块1718用作共享的最后一级高速缓存。

[0201] 在一些实施例中,处理器核1702A至1702N是执行相同的指令集架构的同质核。在另一个实施例中,处理器核1702A至1702N就指令集架构(ISA)而言是异质的,其中,处理器核1702A至1702N中的一者或多者执行第一指令集,而其他核中的至少一者执行第一指令集的子集或不同的指令值。在一个实施例中,处理器核1702A至1702N就微架构而言是异质的,其中,具有相对更高功率消耗的一个或多个核与具有更低功率消耗的一个或多个功率核耦合。另外,处理器1700可以实现在一个或多个芯片上或者被实现为具有除其他部件之外的所展示的部件的SoC集成电路。

[0202] 图18是图形处理器1800的框图,所述图形处理器可以是分立的图形处理单元,或可以是与多个处理核集成的图形处理器。在一些实施例中,图形处理器经由到图形处理器上的寄存器的映射I/O接口并且利用被放置在处理器存储器中的命令与存储器进行通信。在一些实施例中,图形处理器1800包括用于访问存储器的存储器接口1814。存储器接口1814可以是到本地存储器、一个或多个内部高速缓存、一个或多个共享外部高速缓存、和/或到系统存储器的接口。

[0203] 在一些实施例中,图形处理器1800还包括用于将显示输出数据驱动到显示装置1820的显示控制器1802。显示控制器1802包括用于显示器的一个或多个重叠平面的硬件以及多层视频或用户接口元件的组成。在一些实施例中,图形处理器1800包括用于编码、解码、或者向、从或在一个或多个媒体编码格式之间进行媒体代码转换的视频编解码器引擎

1806,包括但不限于:运动图像专家组(MPEG)格式(比如MPEG-2)、高级视频译码(AVC)格式(比如H.264/MPEG-4AVC)、以及电影&电视工程师协会(SMPTE)421M/VC-1、和联合图像专家组(JPEG)格式(比如JPEG、以及运动JPEG(MJPEG)格式)。

[0204] 在一些实施例中,图形处理器1800包括用于执行二维(2D)光栅化器操作的块图像传输(BLIT)引擎1804,所述2D光栅化器操作包括(例如)位边界块传输。然而,在一个实施例中,使用图形处理引擎(GPE)1810的一个或多个部件执行2D图形操作。在一些实施例中,图形处理引擎1810是用于执行图形操作的计算引擎,所述图形操作包括三维(3D)图形操作和媒体操作。

[0205] 在一些实施例中,GPE 1810包括用于执行3D操作的3D流水线1812,所述3D操作为比如使用作用于3D图元形状(例如,矩形、三角形等)的处理功能来渲染三维图像和场景。3D流水线1812包括在元件和/或生成的执行线程内向3D/媒体子系统1815执行各种任务的可编程和固定功能元件。虽然3D流水线1812可以用于执行媒体操作,但是GPE 1810的实施例还包括媒体流水线1816,所述媒体流水线具体地用于执行媒体操作,比如视频后处理和图像增强。

[0206] 在一些实施例中,媒体流水线1816包括用于代替或代表视频编解码器引擎1806执行一个或多个专门的媒体操作的固定功能或可编程逻辑单元,所述专门的媒体操作为比如视频解码加速、视频解交织和视频编码加速。在一些实施例中,媒体流水线1816另外包括线程生成单元以便生成用于在3D/媒体子系统1815上执行的线程。所生成的线程对3D/媒体子系统1815中所包括的一个或多个图形执行单元执行对媒体操作的计算。

[0207] 在一些实施例中,3D/媒体子系统1815包括用于执行由3D流水线1812和媒体流水线1816生成的线程的逻辑。在一个实施例中,流水线向3D/媒体子系统1815发送线程执行请求,所述3D/媒体子系统包括用于仲裁并将各请求分派到可用的线程执行资源的线程分派逻辑。执行资源包括用于处理3D和媒体线程的图形执行单元的阵列。在一些实施例中,3D/媒体子系统1815包括用于线程指令和数据的一个或多个内部高速缓存。在一些实施例中,所述子系统还包括共享存储器(包括寄存器和可寻址存储器),以便在线程之间共享数据并存储输出数据。

[0208] 3D/媒体处理

[0209] 图19是根据一些实施例的图形处理器的图形处理引擎1910的框图。在一个实施例中,GPE 1910是图18中所示的GPE 1810的一个版本。图19中具有与本文任何其他图的元件相同的参考数字(或名称)的元件可以以与本文别处描述的方式类似的任何方式来操作或起作用,但并不限于此。

[0210] 在一些实施例中,GPE 1910与命令流转化器1903耦合,所述命令流转化器将命令流提供给GPE的3D流水线1912和媒体流水线1916。在一些实施例中,命令流转化器1903耦合至存储器,所述存储器可以是系统存储器,或可以是内部高速缓存存储器和共享高速缓存存储器中的一者或多者。在一些实施例中,命令流转化器1903从存储器接收命令,并且将命令发送给3D流水线1912和/或媒体流水线1916。所述命令是从存储用于3D流水线1912和媒体流水线1916的环形缓冲器获取的指示。在一个实施例中,所述环形缓冲器可另外包括存储多批多命令的批命令缓冲器。3D流水线1912和媒体流水线1916通过经由各自流水线内的逻辑执行操作或者通过将一个或多个执行线程分派至执行单元阵列1914来处理所述命令。

在一些实施例中,执行单元阵列1914是可缩放的,使得所述阵列基于GPE 1910的目标功率和性能级别而包括可变数目的执行单元。

[0211] 在一些实施例中,采样引擎1930与存储器(例如,高速缓存存储器或系统存储器)以及执行单元阵列1914耦合。在一些实施例中,采样引擎1930为执行单元阵列1914提供存储器访问机制,所述存储器访问机制允许执行阵列1914从存储器读取图形和媒体数据。在一些实施例中,采样引擎1930包括用于执行针对媒体的专门图像采样操作的逻辑。

[0212] 在一些实施例中,采样引擎1930中的专门的媒体采样逻辑包括去噪/解交织模块1932、运动估计模块1934以及图像缩放和过滤模块1936。在一些实施例中,去噪/解交织模块1932包括用于对经解码的视频数据执行去噪或解交织算法中的一者或多者的逻辑。解交织逻辑将经交织的视频内容的交变长组合为视频的单个帧。去噪逻辑从视频和图像数据减少或去除数据噪声。在一些实施例中,所述去噪和解交织逻辑是运动自适应的并且使用基于在视频数据中检测到的运动量的空间或时间过滤。在一些实施例中,去噪/解交织模块1932包括专门的运动检测逻辑(例如,在运动估计引擎1934内)。

[0213] 在一些实施例中,运动估计引擎1934通过对视频数据执行视频加速度函数(诸如,运动向量估计和预测)来提供对视频操作的硬件加速度。运动估计引擎确定描述连续视频帧之间的图像数据变换的运动向量。在一些实施例中,图形处理器媒体编解码器使用视频运动估计引擎1934来对宏块级视频执行操作,对于其利用通用处理器来执行可以另外地是太计算密集型的。在一些实施例中,运动估计引擎1934通常可用于图形处理器部件以便辅助视频解码和处理功能,所述视频解码和处理功能对于视频数据内的运动的方向或幅度是敏感或自适应的。

[0214] 在一些实施例中,图像缩放和过滤模块1936执行图像处理操作,以提高所产生的图像和视频的视觉质量。在一些实施例中,缩放和过滤模块1936在向执行单元阵列1914提供数据之前在采样操作期间处理图像和视频数据。

[0215] 在一些实施例中,GPE 1910包括数据端口1944,所述数据端口提供用于使图形子系统访问存储器的附加机制。在一些实施例中,数据端口1944针对操作促进存储器访问,所述操作包括渲染目标写入、恒定缓冲器读取、暂时存储器空间读取/写入、和媒体表面访问。在一些实施例中,数据端口1944包括用于高速缓存对存储器的访问的高速缓存存储器空间。高速缓存存储器可以是单个数据高速缓存,或被分离成用于经由数据端口来访问存储器的多个子系统的多个高速缓存(例如,渲染缓冲高速缓存、恒定缓冲器高速缓存等)。在一些实施例中,执行在执行单元阵列1914中的执行单元上的线程通过经由数据分布互连交换消息来与数据端口通信,所述数据分布互连耦合GPE 1910的每个子系统。

[0216] 执行单元

[0217] 图20是图形处理器2000的另一个实施例的框图。图20中具有与本文任何其他图的元件相同的参考数字(或名称)的元件可以以与本文别处描述的方式类似的任何方式来操作或起作用,但并不限于此。

[0218] 在一些实施例中,图形处理器2000包括环形互连2002、流水线前端2004、媒体引擎2037和图形核2080A至2080N。在一些实施例中,环形互连2002将图形处理器耦合至其他处理单元,包括其他图形处理器或者一个或多个通用处理器核。在一些实施例中,图形处理器是集成在多核处理系统内的多个处理器之一。

[0219] 在一些实施例中,图形处理器2000经由环形互连2002接收多批命令。由流水线前端2004中的命令流转化器2003翻译传入的命令。在一些实施例中,图形处理器2000包括用于经由图形核2080A至2080N来执行3D几何处理和媒体处理的可缩放执行逻辑。针对3D几何处理命令,命令流转化器2003将命令供应给几何流水线2036。针对至少一些媒体处理命令,命令流转化器2003将命令供应给视频前端2034,所述视频前端与媒体引擎2037耦合。在一些实施例中,媒体引擎2037包括用于视频和图像后处理的视频质量引擎(VQE)2030以及用于提供硬件加速的媒体数据编码和解码的多格式编码/解码(MFX)2033引擎。在一些实施例中,几何流水线2036和媒体引擎2037各自生成执行线程,所述执行线程用于由至少一个图形核2080A提供的线程执行资源。

[0220] 在一些实施例中,图形处理器2000包括以模块化核2080A至2080N(有时称为核切片)为特征的可缩放线程执行资源,每个模块化核具有多个子核2050A至2050N、2060A至2060N(有时称为核子切片)。在一些实施例中,图形处理器2000可以具有任意数量的图形核2080A至2080N。在一些实施例中,图形处理器2000包括图形核2080A,所述图形核至少具有第一子核2050A和第二子核2060A。在其他实施例中,图形处理器是具有单个子核(例如,2050A)的低功率处理器。在一些实施例中,图形处理器2000包括多个图形核2080A至2080N,每个图形核包括一组第一子核2050A至2050N和一组第二子核2060A至2060N。所述一组第一子核2050A至2050N中的每个子核至少包括第一组执行单元2052A至2052N和媒体/纹理采样器2054A至2054N。所述一组第二子核2060A至2060N中的每个子核至少包括第二组执行单元2062A至2062N和采样器2064A至2064N。在一些实施例中,每个子核2050A至2050N、2060A-2060N共享一组共享资源2070A至2070N。在一些实施例中,这些共享资源包括共享高速缓存存储器和像素操作逻辑。其他共享资源也可包括在图形处理器的各种实施例中。

[0221] 图21展示了可线程执行逻辑2100,包括在GPE的一些实施例中所采用的处理元件的阵列。图21中具有与本文任何其他图的元件相同的参考数字(或名称)的那些元件可以以与本文别处描述的方式类似的任何方式来操作或起作用,但并不限于此。

[0222] 在一些实施例中,线程执行逻辑2100包括像素着色器2102、线程分派器2104、指令高速缓存2106、可缩放执行单元阵列,包括多个执行单元2108A至2108N、采样器2110、数据高速缓存2112和数据端口2114。在一个实施例中,这些所包括的部件经由互连结构而互连,所述互连结构链接到这些部件中每一者。在一些实施例中,通过指令高速缓存2106、数据端口2114、采样器2110和执行单元阵列2108A至2108N中的一者,线程执行逻辑2100包括至存储器(比如,系统存储器或高速缓存存储器)的一个或多个连接。在一些实施例中,每个执行单元(例如,2108A)是个别向量处理器,能够执行多个同时的线程并且针对每个线程来并行处理多个数据元素。在一些实施例中,执行单元阵列2108A至2108N包括任何数目的个别执行单元。

[0223] 在一些实施例中,执行单元阵列2108A至2108N主要用于执行“着色器”程序。在一些实施例中,阵列2108A至2108N中的执行单元执行包括对许多标准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,直接3D和OpenGL)的着色器程序。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)和通用处理(例如,计算和媒体着色器)。

[0224] 执行单元阵列2108A至2108N中的每个执行单元对数据元素的阵列进行操作。数据

元素的数目是“执行大小“或用于指令的通道的数目。执行通道是用于数据元素访问、掩码和指令内的流控制的逻辑执行单元。通道的数目可以与针对特定图形处理器的物理算术逻辑单元(ALU)或浮点单元(FPU)的数目无关。在一些实施例中,执行单元2108A至2108N支持整数和浮点数据类型。

[0225] 执行单元指令集包括单指令多数据(SIMD)。可以将各种数据元素作为压缩数据类型存储在寄存器中,并且执行单元将基于各种元素的数据大小来处理这些元素。例如,当在256位宽的向量上进行操作时,所述256位的向量存储在寄存器中,并且所述执行单元作为四个单独64位压缩数据元素(四倍字长(QW)大小的数据元素)、八个单独32位压缩数据元素(双倍字长(DW)大小的数据元素)、十六个单独16位压缩数据元素(字长(W)大小的数据元素)、或三十二个单独8位数据元素(字节(B)大小的数据元素)在所述向量上进行操作。然而,不同的向量宽度和寄存器大小是可能的。

[0226] 一个或多个内部指令高速缓存(例如,2106)被包括在线程执行逻辑2100中,以高速缓存用于执行单元的线程指令。在一些实施例中,一个或多个数据高速缓存(例如,2112)被包括用于高速缓存在线程执行期间的线程数据。在一些实施例中,采样器2110被包括用于为3D操作提供纹理采样并且为媒体操作提供媒体采样。在一些实施例中,采样器2110包括专门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或媒体数据。

[0227] 在执行期间,图形流水线和媒体流水线经由线程生成和分派逻辑将线程发起请求发送给线程执行逻辑2100。在一些实施例中,线程执行逻辑2100包括本地线程分派器2104,所述本地线程分派器仲裁来自图形流水线和媒体流水线的线程发起请求并在一个或多个执行单元2108A至2108N上实例化所请求的线程。例如,几何流水线(例如,图20的2036)将顶点处理、曲面细分或几何处理线程分派给线程执行逻辑2100(图21)。在一些实施例中,线程分派器2104还可以处理来自执行着色器程序的运行时间线程生成请求。

[0228] 一旦一组几何对象已被处理并被光栅化为像素数据,就调用像素着色器2102以进一步计算输出信息并导致将结果写入到输出表面(例如,颜色缓冲器、深度缓冲器、模板印刷缓冲器等)。在一些实施例中,像素着色器2102计算各顶点属性的值,所述各顶点属性跨栅格化对象被内插。在一些实施例中,像素着色器2102然后执行应用编程接口(API)供应的像素着色器程序。为了执行所述像素着色器程序,像素着色器2102经由线程分派器2104将线程分派给执行单元(例如,2108A)。在一些实施例中,像素着色器2102使用采样器2110中的纹理采样逻辑来访问存储器中所存储的纹理图中的纹理数据。对纹理数据和输入几何进行的算术运算计算每个几何片段的像素颜色数据,或放弃一个或多个像素以供进一步处理。

[0229] 在一些实施例中,数据端口2114提供用于使线程执行逻辑2100将已处理的数据输出到存储器以供在图形处理器输出流水线上处理的存储器访问机制。在一些实施例中,数据端口2114包括或耦合至一个或多个高速缓存存储器(例如,数据高速缓存2112)从而经由数据端口高速缓存数据以供存储器访问。

[0230] 图22是根据一些实施例的示意图形处理器指令格式2200的框图。在一个或多个实施例中,图形处理器执行单元支持具有多种格式的指令的指令集。实线框展示通常包括在执行单元指令中的分量,而虚线包括可选的或仅包括在指令的子集中的分量。在一些实施

例中,所描述和展示的指令格式2200是宏指令,因为它们是供应至执行单元的指令,这与从指令解码产生的微操作相反(一旦所述指令被处理)。

[0231] 在一些实施例中,图形处理器执行单元原生地支持呈128位格式2210的指令。64位紧凑指令格式2230可用于基于所选的指令、指令选项和操作数的数目的一些指令。原生128位格式2210提供对所有指令选项的访问,而一些选项和操作在64位格式2230中则被限制。以64位格式2230可用的原生指令根据实施例而变化。在一些实施例中,使用索引字段2213中的一组索引值将指令部分地紧凑。执行单元硬件基于这些索引值参考一组压缩表,并且使用压缩表输出来以128位格式2210重建原生指令。

[0232] 针对每种格式,指令操作码2212定义执行单元要执行的操作。执行单元跨越每个操作数的多个数据元素并行执行每个指令。例如,响应于加法指令,执行单元跨越表示纹理元素或图片元素的每个颜色通道来执行同时加法运算。默认情况下,执行单元跨越操作数的所有数据通道执行每个指令。在一些实施例中,指令控制字段2214使得能控制某些执行选项,比如通道选择(例如,预测)以及数据通道排序(例如,混合)。针对128位指令2210,执行大小字段2216限制将被并行执行的数据通道的数目。在一些实施例中,执行大小字段2216不可用于64位紧凑指令格式2230。

[0233] 一些执行单元指令具有至多三个操作数,包括两个源操作数src0 2220、src1 2222和一个目的地2218。在一些实施例中,执行单元支持双目的地指令,其中,这些目的地之一是隐式的。数据操纵指令可以具有第三源操作数(例如,SRC2 2224),其中,指令操作码2212确定源操作数的数目。指令的最后一个源操作数可以是通过所述指令传递的立即(例如,硬编码)值。

[0234] 在一些实施例中,128位指令格式2210包括访问/地址模式信息2226,所述访问/地址模式信息指定(例如)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,直接由指令2210中的位来提供一个或多个操作数的寄存器地址。

[0235] 在一些实施例中,128位指令格式2210包括访问/地址模式字段2226,所述访问/地址模式字段指定所述指令的地址模式和/或访问模式。在一个实施例中,访问模式定义所述指令的数据访问对齐。一些实施例支持包括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数的访问对齐。例如,当处于第一模式时,指令2210可针对源操作数和目的地操作数使用字节对齐寻址,并且当处于第二模式时,指令2210可针对所有的源操作数和目的地操作数使用16字节对齐寻址。

[0236] 在一个实施例中,访问/地址模式字段2226的地址模式部分确定指令将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2210中的位直接提供一个或多个操作数的寄存器地址。当使用间接寄存器寻址模式时,可基于地址寄存器值和指令中的地址立即字段来计算一个或多个操作数的寄存器地址。

[0237] 在一些实施例中,基于操作码2212位字段对指令分组,以简化操作码解码2240。针对8位操作码,位4、5和6允许执行单元确定操作码的类型。所示出的精确操作码分组仅是示例性的。在一些实施例中,移动和逻辑操作码组2242包括数据移动和逻辑指令(例如,移动(mov)、比较(cmp))。在一些实施例中,移动和逻辑组2242共享五个最高有效位(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2244(例如,调用(call)、跳(jmp))包括采用0010xxxxb形式(例如,0x20)的指令。混杂指令

组2246包括指令的混合体,这些指令包括采用0011xxxxb形式(例如,0x30)的同步指令(例如,等待、发送)。并行数学指令组2248包括采用0100xxxxb形式(例如,0x40)的按分量逐个作出的(component-wise)算术指令(例如,加(add)、减(mul))。并行数学组2248跨越数据通道并行执行算术运算。向量数学组2250包括采用0101xxxxb形式(例如,0x50)的算术指令(例如,dp4)。向量数学组执行比如对向量操作数的点积计算的算术。

[0238] 图形流水线

[0239] 图23是图形处理器2300的另一个实施例的框图。图23中具有与本文任何其他图的元件相同的参考数字(或名称)的元件可以以与本文别处描述的方式类似的任何方式来操作或起作用,但并不限于此。

[0240] 在一些实施例中,图形处理器2300包括图形流水线2320、媒体流水线2330、显示引擎2340、线程执行逻辑2350和渲染输出流水线2370。在一些实施例中,图形处理器2300是包括一个或多个通用处理核的多核处理系统内的图形处理器。图形处理器受到至一个或多个控制寄存器(未示出)的寄存器写入的控制或者经由环形互连2302经由发布至图形处理器2300的命令被控制。在一些实施例中,环形互连2302将图形处理器2300耦合至其他处理部件,比如其他图形处理器或通用处理器。由命令流转化器2303翻译来自环形互连2302的命令,所述命令流转化器将指令供应给图形流水线2320或媒体流水线2330的个别部件。

[0241] 在一些实施例中,命令流转化器2303指导顶点获取器2305的操作,所述顶点获取器从存储器读取顶点数据并执行由命令流转化器2303提供的顶点处理命令。在一些实施例中,顶点获取器2305将顶点数据提供给顶点着色器2307,所述顶点着色器向每个顶点执行坐标空间变换和照明操作。在一些实施例中,顶点获取器2305和顶点着色器2307通过经由线程分派器2331将执行线程分派给执行单元2352A、2352B来执行顶点处理指令。

[0242] 在一些实施例中,执行单元2352A、2352B是具有用于执行图形和媒体操作的指令集的向量处理器的阵列。在一些实施例中,执行单元2352A、2352B具有特定用于每个阵列或在阵列之间共享的附加L1高速缓存2351。所述高速缓存可以被配置为数据高速缓存、指令高速缓存或单个高速缓存,所述单个高速缓存被分割成将数据和指令包含在不同的分区中。

[0243] 在一些实施例中,图形流水线2320包括用于执行对3D对象的硬件加速曲面细分的曲面细分部件。在一些实施例中,可编程的外壳着色器2311配置曲面细分操作。可编程的域着色器2317提供对曲面细分输出的后端评估。曲面细分器2313在外壳着色器2311的方向上进行操作并且包含专用逻辑,所述专用逻辑用于基于粗糙几何模型来生成详细的几何对象集合,所述粗糙几何模型作为输入被提供至图形流水线2320。在一些实施例中,如果未使用曲面细分,则可以对曲面细分部件2311、2313、2317进行旁路。

[0244] 在一些实施例中,完整的几何对象可以由几何着色器2319经由被分派给执行单元2352A、2352B的一个或多个线程来处理,或可以直接继续进行至剪辑器2329。在一些实施例中,几何着色器在整个几何对象(而非顶点或者如图形流水线的先前级中的顶点补片)上进行操作。如果曲面细分被禁用,那么几何着色器2319从顶点着色器2307接收输入。在一些实施例中,几何着色器2319可由几何着色器程序编程以便在曲面细分单元被禁用时执行几何曲面细分。

[0245] 在光栅化之前,剪辑器2329处理顶点数据。剪辑器2329可以是固定功能的剪辑器

或者具有剪裁和几何着色器功能的可编程剪辑器。在一些实施例中,渲染输出流水线2370中的光栅化器2373(例如,深度测试部件)分派像素着色器以将几何对象转换为它们的逐像素表示。在一些实施例中,像素着色器逻辑包括在线程执行逻辑2350中。在一些实施例中,应用可以对光栅化器2373进行旁路,并且经由流出单元2323来访问未光栅化的顶点数据。

[0246] 图形处理器2300具有互连总线、互连结构或某种其他互连机制,其允许在处理器主要部件当中传递数据和消息。在一些实施例中,执行单元2352A、2352B和(多个)关联高速缓存2351、纹理和媒体采样器2354以及纹理/采样器高速缓存2358经由数据端口2356互连,以执行存储器访问并与处理器的渲染输出流水线部件通信。在一些实施例中,采样器2354、高速缓存2351、2358以及执行单元2352A、2352B各自具有单独的存储器访问路径。

[0247] 在一些实施例中,渲染输出流水线2370包含光栅化器2373,所述光栅化器将基于顶点的对象转换为关联的基于像素的表示。在一些实施例中,光栅化器逻辑包括用于执行固定功能三角形和线光栅化的窗口器/掩码器单元。相关联的渲染高速缓存2378和深度高速缓存2379在一些实施例中也是可用的。像素操作部件2377对数据执行基于像素的操作,不过在一些示例中,与2D操作相关联的像素操作(例如,位块图像传输和混合)由2D引擎2341执行,或在显示时间由使用重叠显示平面的显示控制器2343代替。在一些实施例中,共享的L3高速缓存2375可用于所有的图形部件,从而允许在无需使用主系统存储器的情况下共享数据。

[0248] 在一些实施例中,图形处理器媒体流水线2330包括媒体引擎2337和视频前端2334。在一些实施例中,视频前端2334从命令流转化器2303接收流水线命令。在一些实施例中,媒体流水线2330包括单独的命令流转化器。在一些实施例中,视频前端2334在将所述命令发送给媒体引擎2337之前处理媒体命令。在一些实施例中,媒体引擎2337包括用于生成线程以用于经由线程分派器2331分派给线程执行逻辑2350的线程生成功能。

[0249] 在一些实施例中,图形处理器2300包括显示引擎2340。在一些实施例中,显示引擎2340在处理器2300外部并且经由环形互连2302、或某个其他互连总线或结构耦合至图形处理器。在一些实施例中,显示引擎2340包括2D引擎2341和显示控制器2343。在一些实施例中,显示引擎2340包含能够独立于3D流水线而操作的专用逻辑。在一些实施例中,显示控制器2343与显示装置(未示出)耦合,所述显示装置可以是系统集成式显示装置(如在膝上型计算机中),或可以是经由显示装置连接器所附接的外部显示装置。

[0250] 在一些实施例中,图形流水线2320和媒体流水线2330可配置成基于多个图形和媒体编程接口来执行操作,并且不特定于任何一个应用编程接口(API)。在一些实施例中,图形处理器的驱动器软件将特定于特定图形或媒体库的API调度转换成可由图形处理器处理的命令。在一些实施例中,为来自科纳斯(Khronos)集团的开放图形库(OpenGL)和开放计算语言(OpenCL)、来自微软公司的Direct 3D库提供支持、或者可以向OpenGL和D3D两者提供支持。还可以为开源计算机视觉库(OpenCV)提供支持。如果可以进行从未来API调用的流水线至图形处理器的流水线的映射,那么还将支持具有兼容的3D流水线的未来API。

[0251] 图形流水线编程

[0252] 图24A是根据一些实施例的示意图形处理器命令格式2400的框图。图24B是根据实施例的示意图形处理器命令序列2410的框图。图24A中的实线框展示通常包括在图形命令中的分量,而虚线包括可选的或仅包括在图形命令的子集中的分量。图24A的示例性图形处

理器命令格式2400包括用于标识命令的目标客户端2402、命令操作代码(操作码)2404和命令的相关数据2406的数据字段。在一些命令中还包括子操作码2405和命令大小2408。

[0253] 在一些实施例中,客户端2402指定处理命令数据的图形装置的客户端单元。在一些实施例中,图形处理器命令解析器检查每个命令的客户端字段以便调整对命令的进一步处理并将命令数据路由至合适的客户端单元。在一些实施例中,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流水线。一旦命令被客户端单元接收,客户端单元就读取操作码2404以及(如果存在的话)子操作码2405以确定待执行的操作。客户端单元使用数据字段2406中的信息来执行命令。针对一些命令,期待显式命令大小2408以指定命令的大小。在一些实施例中,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。在一些实施例中,经由双倍字长的倍数对命令进行对齐。

[0254] 图24B中的流程图示出了示例性图形处理器命令序列2410。在一些实施例中,以图形处理器的实施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版本来启动、执行并终止图形操作集合。仅出于示例性目的示出并描述了样本命令序列,因为实施例并不限于这些特定命令或者此命令序列。此外,所述命令可以作为一批命令以命令序列被发布,从而使得图形处理器将以至少部分同时的方式处理命令序列。

[0255] 在一些实施例中,图形处理器命令序列2410可以流水线转储清除命令2412开始,以使任何活跃的图形流水线完成所述流水线的当前未决命令。在一些实施例中,3D流水线2422和媒体流水线2424不同时进行操作。执行流水线转储清除以使活跃的图形流水线完任何未决命令。响应于流水线转储清除,图形处理器的命令解析器将暂停命令处理,直到活跃的绘图引擎完成未决操作且相关的读取高速缓存无效。可选地,渲染高速缓存中被标记为'脏(dirty)'的任何数据可以被转储清除到存储器。在一些实施例中,可以针对流水线同步或在将图形处理器放置处于低功率状态之前使用流水线转储清除命令2412。

[0256] 在一些实施例中,当命令序列要求图形处理器在流水线之间作明确切换时,使用流水线选择命令2413。在一些实施例中,在发布流水线命令之前在执行情境中仅需要一次流水线选择命令2413,除非所述情境要发布针对两条流水线的命令。在一些实施例中,在经由流水线选择命令2413作流水线切换之前立即需要流水线转储清除命令是2412。

[0257] 在一些实施例中,流水线控制命令2414配置用于操作的图形流水线,并用于对3D流水线2422和媒体流水线2424编程。在一些实施例中,流水线控制命令2414配置活跃流水线的流水线状态。在一个实施例中,流水线控制命令2414被用于流水线同步,以及用于在处理一批命令之前将数据从活跃的流水线内的一个或多个高速缓存存储器中清除。

[0258] 在一些实施例中,使用返回缓冲器状态命令2416来配置用于使相应的流水线写入数据的一组返回缓冲器。一些流水线操作需要分配、选择或配置一个或多个返回缓冲器,这些操作在处理期间将中间数据写入到所述返回缓冲器中。在一些实施例中,图形处理器还使用一个或多个返回缓冲器以便存储输出数据并且执行跨线程通信。在一些实施例中,返回缓冲器状态2416包括选择返回缓冲器的大小和数量以用于流水线操作集合。

[0259] 命令序列中的剩余命令基于用于操作的活跃流水线而不同。基于流水线确定2420,根据3D流水线2422和媒体流水线2424来定制命令序列,所述3D流水线以3D流水线状态2430开始,所述媒体流水线始于媒体流水线状态2440处。

[0260] 用于3D流水线状态2430的命令包括用于以下各者的3D状态设置命令:顶点缓冲器状态、顶点元素状态、恒定颜色状态、深度缓冲器状态和将在处理3D图元命令之前配置的其他状态变量。至少部分地基于使用中的特定3D API来确定这些命令的值。在一些实施例中,3D流水线状态2430命令还能够选择性地禁用或旁路掉特定流水线元件(如果将不使用那些元件的话)。

[0261] 在一些实施例中,3D图元2432命令用于提交待由3D流水线处理的3D图元。经由3D图元2432传递到图形处理器的命令和关联的参数被转发到图形流水线中的顶点获取函数。顶点获取函数使用3D图元2432命令数据来产生顶点数据结构。顶点数据结构被存储在一个或多个返回缓冲器中。在一些实施例中,3D图元2432命令用于经由顶点着色器对3D图元执行顶点操作。为了处理顶点着色器,3D流水线2422将着色器执行线程分派给图形处理器执行单元。

[0262] 在一些实施例中,经由执行2434命令或事件来触发3D流水线2422。在一些实施例中,寄存器写入触发命令执行。在一些实施例中,经由命令序列中的'go'或'kick'命令来触发执行。在一个实施例中,使用流水线同步命令来触发命令执行,以通过图形流水线来转储清除命令序列。3D流水线将执行针对3D图元的几何处理。一旦操作完成,便对所得几何对象光栅化,并且像素引擎给所得像素上色。针对那些操作还可包括用于控制像素着色和像素后端操作的附加命令。

[0263] 在一些实施例中,当执行媒体操作时,图形处理器命令序列2410遵循媒体流水线2424路径。一般地,媒体流水线2424的特定用途和编程方式取决于待执行的媒体或计算操作。在媒体解码期间,可将特定的媒体解码操作卸载到媒体流水线。在一些实施例中,还可以对媒体流水线进行旁路,并且可以使用由一个或多个通用处理核提供的资源来整体地或部分地执行媒体解码。在一个实施例中,媒体流水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,所述计算着色器程序与图形图元的渲染不明确相关。

[0264] 在一些实施例中,以与3D流水线2422类似的方式配置媒体流水线2424。在媒体对象命令2442之前将一组媒体流水线状态命令2440分派到或放置到命令队列中。在一些实施例中,媒体流水线状态命令2440包括用于配置媒体流水线元件的数据,所述媒体流水线元件将用于处理媒体对象。这包括用于配置媒体流水线内的视频解码和视频编码逻辑的数据(比如编码或解码模式)。在一些实施例中,媒体流水线状态命令2440还支持将一个或多个指针用于包含一批状态设置的"间接"状态元件。

[0265] 在一些实施例中,媒体对象命令2442将指针供应给供由媒体流水线处理的媒体对象。媒体对象包括包含待处理的视频数据的存储器缓冲器。在一些实施例中,在发布媒体对象命令2442之前,所有的媒体流水线状态必须是有效的。一旦流水线状态被配置并且媒体对象命令2442被排队,则经由执行命令2444或等效的执行事件(例如,寄存器写入)来触发媒体流水线2424。然后可以通过由3D流水线2422或媒体流水线2424提供的操作对来自媒体流水线2424的输出进行后处理。在一些实施例中,以与媒体操作类似的方式来配置和执行GPGPU操作。

[0266] 图形软件架构

[0267] 图25展示根据一些实施例的数据处理系统2500的示例性图形软件架构。在一些实

施例中,软件架构包括3D图形应用2510、操作系统2520、以及至少一个处理器2530。在一些实施例中,处理器2530包括图形处理器2532以及一个或多个通用处理器核2534。图形应用2510和操作系统2520各自在数据处理系统的系统存储器2550中执行。

[0268] 在一些实施例中,3D图形应用2510包含一个或多个着色器程序,所述着色器程序包括着色器指令2512。着色器语言指令可以呈高阶着色器语言,比如高阶着色器语言(HLSL)或OpenGL着色器语言(GLSL)。所述应用还包括呈适合于由通用处理器核2534执行的机器语言的可执行指令2514。所述应用还包括由顶点数据定义的几何对象2516。

[0269] 在一些实施例中,操作系统2520是来自微软公司的Microsoft® Windows®操作系统、使用Linux内核的变体的专属类UNIX操作系统或开源类UNIX操作系统。当Direct3D API在使用中时,操作系统2520使用前端着色器编译器2524以将呈HLSL的任何着色器指令2512编译为低阶着色器语言。所述编译可以是即时(JIT)编译,或者所述应用可执行着色器预编译。在一些实施例中,在对3D图形应用2510进行编译期间,将高阶着色器编译成低阶着色器。

[0270] 在一些实施例中,用户模式图形驱动器2526包含后端着色器编译器2527,所述后端着色器编译器用于将着色器指令2512转换为硬件特定表示。当OpenGL API在使用中时,呈GLSL高阶语言的着色器指令2512被传递到用户模式图形驱动器2526以供编译。在一些实施例中,用户模式图形驱动器2526使用操作系统内核模式函数2528来与内核模式图形驱动器2529进行通信。在一些实施例中,内核模式图形驱动器2529与图形处理器2532进行通信以便分派命令和指令。

[0271] IP核实现方式

[0272] 至少一个实施例的一个或多个方面可由存储在机器可读介质上的代表性代码来实现,所述机器可读介质表示和/或定义集成电路(比如,处理器)内的逻辑。例如,机器可读介质可以包括表示处理器内的各个逻辑的指令。当由机器读取时,所述指令可以使所述机器制造用于执行本文描述的技术的逻辑。这类表示(称为“IP核”)是集成电路的逻辑的可重复使用单元,所述可重复使用单元可以作为对集成电路的结构进行描述的硬件模型而存储在有形、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消费者或制造设施。可以制造集成电路,从而使得所述电路执行与在此描述的实施例中的任一实施例相关联地描述的操作。

[0273] 图26是根据实施例的展示IP核开发系统2600的框图,所述IP核开发系统可用于制造集成电路以执行操作。IP核开发系统2600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集成电路)的模块化、可重复使用设计。设计设施2630可采用高阶编程语言(例如,C/C++)生成对IP核设计的软件仿真2610。软件仿真2610可用于设计、测试并验证IP核的行为。然后可由仿真模型2600来创建或合成寄存器传输级(RTL)设计。RTL设计2615是对硬件寄存器之间的数字信号的流动进行建模的集成电路(包括使用建模的数字信号执行的相关联逻辑)的行为的抽象。除了RTL设计2615之外,还可以创建、设计或合成逻辑电平或晶体管电平处的较低层次设计。由此,初始设计和仿真的具体细节可以发生变化。

[0274] 可由设计设施进一步将RTL设计2615或等效物合成为硬件模型2620,所述硬件模型可以呈硬件描述语言(HDL)或物理设计数据的某个其他表示。可以进一步仿真或测试HDL

以验证IP核设计。可以使用非易失性存储器2640(例如,硬盘、闪存或任何非易失性存储介质)来存储IP核设计以供递送到第3方制造设施2665。可替代地,可以通过有线连接2650或无线连接2660来传输(例如,经由互联网)IP核设计。制造设施2665然后可以制造至少部分地基于IP核设计的集成电路。所制造的集成电路可被配置用于执行根据本文描述的至少一个实施例的操作。

[0275] 图27是根据实施例的展示示例性片上系统集成电路2700的框图,可使用一个或多个IP核来制造所述系统芯片集成电路。示例性集成电路包括一个或多个应用处理器2705(例如,CPU)、至少一个图形处理器2710,并且可以另外包括图像处理器2715和/或视频处理器2720,其中的任一中可以是来自相同或多个不同设计设施的模块化IP核。集成电路包括外围或总线逻辑,包括USB控制器2725、UART控制器2730、SPI/SDIO控制器2735、I^2S/I^2C控制器2740。另外,集成电路可以包括显示装置2745,所述显示装置耦合至高清晰度多媒体接口(HDMI)控制器2750和移动行业处理器接口(MIPI)显示接口2755中的一者或多者。可以由闪存子系统2760(包括闪存和闪存控制器)来提供存储。可经由存储器控制器2765来提供存储器接口以用于访问SDRAM或SRAM存储器装置。一些集成电路另外包括嵌入式安全引擎2770。

[0276] 另外,其他逻辑和电路可被包括在集成电路2700的处理器中,这些逻辑和电路包括附加的图形处理器/核、外围接口控制器或通用处理器核。

[0277] 附加注释和示例:

[0278] 示例1可包括阻塞估计系统,该阻塞估计系统包括可编程处理器和存储器,该存储器包括一组指令,这组指令如果由可编程处理器执行,则使可编程处理器接收视频帧的运动向量。该系统还可包括用于以下操作的逻辑:增加视频的帧率;检测视频的一个或多个帧的物体的运动;以及在帧率变换过程中估计中间帧内的物体的阻塞,其中阻塞是基于二维(2D)到三维(3D)结构估计而被估计的。

[0279] 示例2可包括示例1的系统,其中增加帧率改善图像震颤。

[0280] 示例3可包括示例2的系统,其中帧率变换包括将帧率变成两倍。

[0281] 示例4可包括示例1到3的系统中的任何一个,其中2D到3D结构估计基于运动向量的位置来估计阻塞位置。

[0282] 示例5可包括示例1的系统,其中,中间帧被内插在一个或多个帧中的两个帧之间。

[0283] 示例6可包括示例5的系统,其中移动物体被估计为被内插在遮挡的前方。

[0284] 示例7可包括示例5的系统,其中移动物体被估计为被内插在遮挡的后方。

[0285] 示例8可包括阻塞估计设备,该阻塞估计设备包括:可编程处理器,用于接收视频帧的运动向量;以及固定功能逻辑,用于:增加视频的帧率;检测视频的一个或多个帧内的物体的运动;以及在帧率变换过程中估计中间帧内的物体的阻塞,其中阻塞是基于二维(2D)到三维(3D)结构估计而被估计的。

[0286] 示例9可包括示例8的设备,其中增加帧率改善图像震颤。

[0287] 示例10可包括示例9的设备,其中帧率变换包括将帧率变成两倍。

[0288] 示例11可包括示例8到10中的任何一项的设备,其中2D到3D结构估计基于运动向量的位置来估计阻塞位置。

[0289] 示例12可包括示例8的设备,其中,中间帧被内插在一个或多个帧中的两个帧之间。

[0290] 示例13可包括示例12的设备,其中移动物体被估计为被内插在遮挡的前方。

[0291] 示例14可包括示例8的设备,其中移动物体被估计为被内插在遮挡的后方。

[0292] 示例15包括阻塞估计方法,该阻塞估计方法包括:接收视频帧的运动向量;增加视频的帧率;检测视频的一个或多个帧内的物体的运动;以及在帧率变换过程中估计中间帧内的物体的阻塞,其中阻塞是基于二维(2D)到三维(3D)结构估计而被估计的。

[0293] 示例16可包括示例15的方法,其中增加帧率改善图像震颤。

[0294] 示例17可包括示例16的方法,其中帧率变换包括将帧率变成两倍。

[0295] 示例18可包括示例15到17中的任何一项的方法,其中2D到3D结构估计基于运动向量的位置来估计阻塞位置。

[0296] 示例19可包括示例15的方法,其中中间帧被内插在一个或多个帧中的两个帧之间。

[0297] 示例20可包括示例19的方法,其中移动物体被估计为被内插在遮挡的前方。

[0298] 示例21可包括示例19的方法,其中移动物体被估计为被内插在遮挡的后方。

[0299] 示例22可包括至少一种非瞬态计算机可读存储介质,其包括一组指令,该组指令如果由计算装置执行,则使计算装置:接收视频帧的运动向量;增加视频的帧率;检测视频的一个或多个帧内的物体的运动;以及在帧率变换过程中估计中间帧内的物体的阻塞,其中阻塞是基于二维(2D)到三维(3D)结构估计而被估计的。

[0300] 示例23可包括示例15到22中的至少一种非瞬态计算机可读存储介质,其中2D到3D结构估计基于运动向量的位置来估计阻塞位置。

[0301] 示例24可包括示例22或23的至少一种非瞬态计算机可读存储介质,其中中间帧被内插在一个或多个帧中的两个帧之间。

[0302] 示例25可包括示例22或23的至少一种非瞬态计算机可读存储介质,其中移动物体被估计为被内插在遮挡的前方。

[0303] 示例26可包括帧率变换系统,该帧率变换系统包括:一个或多个深度相机,用于确定移动物体的深度;可编程处理器;以及存储器,该存储器包括一组指令,这组指令如果由可编程处理器执行,则使可编程处理器接收视频帧的运动向量。该系统还可包括逻辑,该逻辑用于:增加视频的帧率;检测视频的一个或多个帧内的物体的运动;基于来自一个或多个深度相机的输入来检测物体的深度;估计中间帧中的物体的阻塞;以及在帧率变换过程中内插中间帧。

[0304] 示例27可包括示例26的系统,其中增加帧率改善图像震颤。

[0305] 示例28可包括示例27的系统,其中帧率变换包括将帧率变成两倍。

[0306] 示例29可包括示例26的系统,其中所确定的深度用于改善阻塞估计。

[0307] 示例30可包括示例29的系统,其中深度相机是从包括飞行时间传感器和已编码光传感器的组中选择的。

[0308] 示例31可包括示例26到30中任何一项的系统,其中移动物体被估计为被内插在遮挡的前方。

[0309] 示例32可包括示例26到30中任何一项的系统,其中移动物体被估计为被内插在遮挡的后方。

[0310] 示例33可包括帧率变换设备,该帧率变换设备包括:可编程处理器,用于接收视频

帧的运动向量;以及逻辑,用于:增加视频的帧率;检测视频的一个或多个帧内物体的运动;基于来自一个或多个深度相机的输入来检测物体的深度;估计中间帧中的物体的阻塞;以及在帧率变换过程中内插中间帧。

[0311] 示例34可包括示例33的设备,其中增加帧率改善图像震颤。

[0312] 示例35可包括示例34的设备,其中帧率变换包括将帧率变成两倍。

[0313] 示例36可包括示例33的设备,其中所确定的深度用于改善阻塞估计。

[0314] 示例37可包括示例36的设备,其中深度相机是从包括飞行时间传感器和已编码光传感器的组中选择的。

[0315] 示例38可包括示例33到37中任何一项的设备,其中将移动物体估计为在遮挡前方被内插。

[0316] 示例39可包括示例33到37中任何一项的设备,其中移动物体被估计为被内插在遮挡的后方。

[0317] 示例40可包括帧率变换方法,该帧率变换方法包括:接收视频帧的运动向量;增加视频的帧率;检测视频的一个或多个帧内的物体的运动;基于来自一个或多个深度相机的输入来检测物体的深度;估计中间帧中的物体的阻塞;以及在帧率变换过程中内插中间帧。

[0318] 示例41可包括示例40的方法,其中增加帧率改善图像震颤。

[0319] 示例42可包括示例41的方法,其中帧率变换包括将帧率变成两倍。

[0320] 示例43可包括示例40的方法,其中所确定的深度用于改善阻塞估计。

[0321] 示例44可包括示例40的方法,其中深度相机是从包括飞行时间传感器和已编码光传感器的组中选择的。

[0322] 示例45可包括示例40到44中任何一项的方法,其中移动物体被估计为被内插在遮挡的前方。

[0323] 示例46可包括示例40到44中任何一项的方法,其中移动物体被估计为被内插在遮挡的后方。

[0324] 示例47可包括至少一种非瞬态计算机可读存储介质,该储存介质包括一组指令,该组指令如果由计算装置执行,则使计算装置:接收视频帧的运动向量;增加视频的帧率;检测视频的一个或多个帧内的物体的运动;基于来自一个或多个深度相机的输入来检测物体的深度;估计中间帧中的物体的阻塞;以及在帧率变换过程中内插中间帧。

[0325] 示例48可包括示例47的至少一种非瞬态计算机可读存储介质,其中深度相机是从包括飞行时间传感器和已编码光传感器的组中选择的。

[0326] 示例49可包括示例47或48的至少一种非瞬态计算机可读存储介质,其中将移动物体估计为在遮挡前方被内插。

[0327] 示例50包括示例47或48的至少一种非瞬态计算机可读存储介质,其中移动物体被估计为被内插在遮挡的后方。

[0328] 示例51可包括帧率变换系统,该帧率变换系统包括:可编程处理器;存储器,该存储器包括一组指令,这组指令如果由可编程处理器执行,则使可编程处理器接收一个或多个视频帧用于编码;以及逻辑,用于:接收编码引擎的功率预算;确定编码引擎内反馈环路的帧率;以及调整帧率压缩率以确保符合功率预算。

[0329] 示例52可包括示例51的系统,其中指令如果由可编程处理器执行,还使可编程处

理器在确定了帧率已经降低之后降低帧率压缩率。

[0330] 示例53可包括示例52的系统,其中降低帧率压缩率节省处理功率。

[0331] 示例54可包括示例51的系统,其中一个或多个视频帧是从3D图形流水线接收的。

[0332] 示例55可包括示例51到54中任何一项的系统,其中帧率压缩率是由帧率模块调整的。

[0333] 示例56可包括示例55的系统,其中帧率模块向编码器输出反馈以形成反馈环路。

[0334] 示例57可包括帧率压缩设备,该帧率压缩设备包括:可编程处理器,用于接收一个或多个视频帧用于编码;以及逻辑,用于:接收编码引擎的功率预算;确定编码引擎内反馈环路的帧率;以及调整帧率压缩率以确保符合功率预算。

[0335] 示例58可包括示例57的设备,其中指令如果由可编程处理器执行,则还使可编程处理器在确定了帧率已经降低之后降低帧率压缩率。

[0336] 示例59可包括示例58的设备,其中降低帧率压缩率节省处理功率。

[0337] 示例60可包括示例57的设备,其中一个或多个视频帧是从3D图形流水线接收的。

[0338] 示例61可包括示例57到60中任何一项的设备,其中帧率压缩率是由帧率模块调整的。

[0339] 示例62可包括示例61的设备,其中帧率模块向编码器输出反馈以形成反馈环路。

[0340] 示例63可包括帧率压缩方法,该帧率压缩方法包括:接收一个或多个视频帧用于编码;接收编码引擎的功率预算;确定编码引擎内反馈环路的帧率;以及调整帧率压缩率以确保符合功率预算。

[0341] 示例64可包括示例63的方法,还包括:在确定了帧率已经降低之后降低帧率压缩率。

[0342] 示例65可包括示例64的方法,其中降低帧率压缩率节省处理功率。

[0343] 示例66可包括示例63的方法,其中一个或多个视频帧是从3D图形流水线接收的。

[0344] 示例67可包括示例63到66中任何一个的方法,其中帧率压缩率是由帧率模块调整的。

[0345] 示例68可包括示例67的方法,其中帧率模块向编码器输出反馈以形成反馈环路。

[0346] 示例69可包括至少一种非瞬态计算机可读存储介质,其包括一组指令,这组指令如果由计算装置执行,则使计算装置:接收一个或多个视频帧用于编码;接收编码引擎的功率预算;确定编码引擎内的反馈环路的帧率;以及调整帧率压缩率以确保符合功率预算。

[0347] 示例70可包括示例69的至少一种非瞬态计算机可读存储介质,其中指令如果由可编程处理器执行,则还使可编程处理器在确定了帧率已经降低之后降低帧率压缩率。

[0348] 示例71可包括示例70的至少一种非瞬态计算机可读存储介质,其中降低帧率压缩率节省处理功率。

[0349] 示例72可包括示例69的至少一种非瞬态计算机可读存储介质,其中一个或多个视频帧是从3D图形流水线接收的。

[0350] 示例73可包括示例69到72中任何一项的至少一种非瞬态计算机可读存储介质,其中帧率压缩率是由帧率模块调整的。

[0351] 示例74可包括示例73的至少一种非瞬态计算机可读存储介质,还包括向编码器输出反馈以形成反馈环路。

[0352] 示例75可包括视口编码系统,该视口编码系统包括:一个或多个相机;可编程处理器;以及存储器,该存储器包括一组指令,这组指令如果由可编程处理器执行,则使可编程处理器接收与一个或多个相机有关的视口信息。该系统还可包括逻辑,该逻辑用于:确定一个或多个相机是否是移动的;确定视口中的场景是否是静态的;以及将视口的编码传递给编码引擎,其中编码在无需计算运动的情况下被传递。

[0353] 示例76可包括示例75的系统,其中场景状态是经由编码引擎中的API确定的。

[0354] 示例77可包括示例76的系统,其中当场景是静态的,并且视口没有改变时,逻辑结束处理。

[0355] 示例78可包括示例75到77中任何一项的系统,其中当场景是静态的,并且视口没有改变时,视口的编码被传递给编码引擎。

[0356] 示例79可包括示例78的系统,其中当场景不是静态时,编码被传递给阻塞模块。

[0357] 示例80可包括视口编码设备,该视口编码设备包括:可编程处理器,用于接收与一个或多个相机有关的视口信息;以及逻辑,用于:确定一个或多个相机是否是移动的;确定视口中的场景是否是静态的;以及将视口的编码传递给编码引擎,其中编码在无需计算运动的情况下被传递。

[0358] 示例81可包括示例80的设备,其中场景状态是经由编码引擎中的API确定的。

[0359] 示例82可包括示例81的设备,其中当场景是静态的,并且视口没有改变时,逻辑结束处理。

[0360] 示例83可包括示例80到82中任何一项的设备,其中当场景是静态的,并且视口没有改变时,视口的编码被传递给编码引擎。

[0361] 示例84可包括示例83的设备,其中当场景不是静态的时,编码被传递给阻塞模块。

[0362] 示例85可包括视口编码方法,该视口编码方法包括:接收与一个或多个相机有关的视口信息;确定一个或多个相机是否是移动的;确定视口中的场景是否是静态的;以及将视口的编码传递给编码引擎,其中编码在无需计算运动的情况下被传递。

[0363] 示例86可包括示例85的方法,其中场景状态是经由编码引擎中的API确定的。

[0364] 示例87可包括示例86的方法,其中当场景是静态的,并且视口没有改变时,逻辑结束处理。

[0365] 示例88可包括示例85到87中任何一项的方法,其中当场景是静态的,并且视口没有改变时,视口的编码被传递给编码引擎。

[0366] 示例89可包括示例88的方法,其中当场景不是静态的时,编码被传递给阻塞模块。

[0367] 示例90可包括至少一种非瞬态计算机可读存储介质,其包括一组指令,这组指令如果由计算装置执行时,则使计算装置:接收与一个或多个相机有关的视口信息;确定一个或多个相机是否是移动的;确定视口中的场景是否是静态的;以及将视口的编码传递给编码引擎,其中编码在无需计算运动的情况下被传递。

[0368] 示例91可包括示例90的至少一种非瞬态计算机可读存储介质,其中场景状态是经由编码引擎中的API确定的。

[0369] 示例92可包括示例91的至少一种非瞬态计算机可读存储介质法,其中当场景是静态的,并且视口没有改变时,逻辑结束处理。

[0370] 示例93可包括示例90到92中任何一项的至少一种非瞬态计算机可读存储介质,其

中当场景是静态的,并且视口没有改变时,视口的编码被传递给编码引擎。

[0371] 示例94可包括示例93的至少一种非瞬态计算机可读存储介质,其中当场景不是静态的时,编码被传递给阻塞模块。

[0372] 术语“耦合”在本文汇总可以用来指所讨论的部件之间任何类型的、直接或间接的关系,并可以应用于电气、机械、流体、光学、电磁、电机或其他连接。此外,术语“第一”、“第二”等在本文中可仅用于便于讨论,并且除非另外指示,否则不传递任何具体的时域或时序的显著性。此外,应理解,不定冠词“一”或“一个”承载“一个或多个”或“至少一个”的意思。

[0373] 如在本申请和权利要求书中所使用,由术语“一个或多个”描述的项目列表可意指所列表项目的任何组合。例如,短语“A、B和C中的至少一个”意味着A、B、C;A和B;A和C;B和C;或A、B和C。

[0374] 上文中已经参考具体实施例描述了实施例。然而,本领域技术人员将理解,可以对其进行各种修改和改变而不偏离如在所附的权利要求书中所阐述的实施例的更宽泛的精神和范围。因此认为前述说描述和附图是说明性的而不是限制性的。

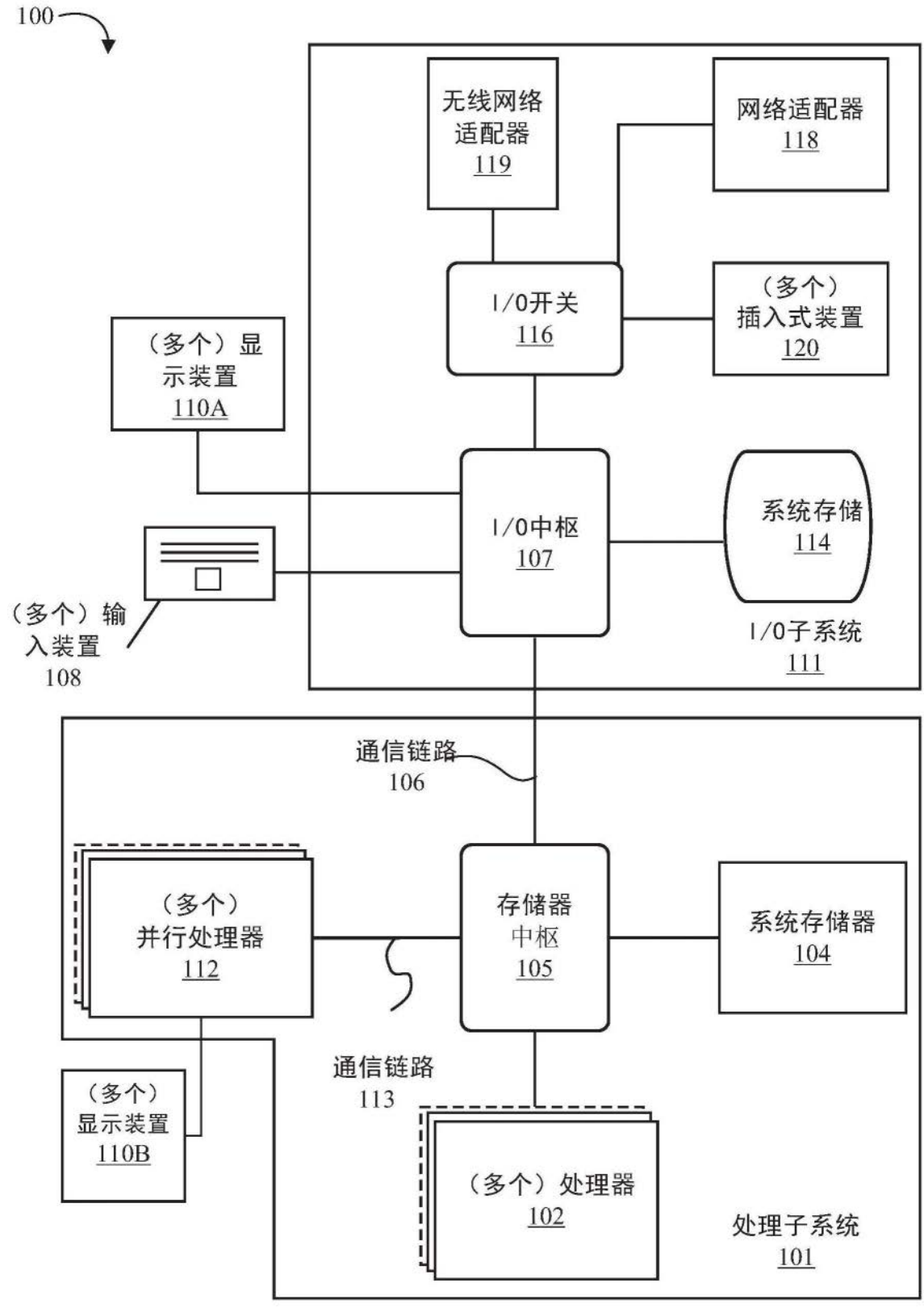

100
无线网络适配器 119
网络适配器 118
I/O开关 116
（多个）插入式装置 120
（多个）显示装置 110A
I/O中枢 107
系统存储 114
（多个）输入装置 108
I/O子系统 111
通信链路 106
（多个）并行处理器 112
存储器中枢 105
系统存储器 104
通信链路 113
（多个）显示装置 110B
（多个）处理器 102
处理子系统 101

图1

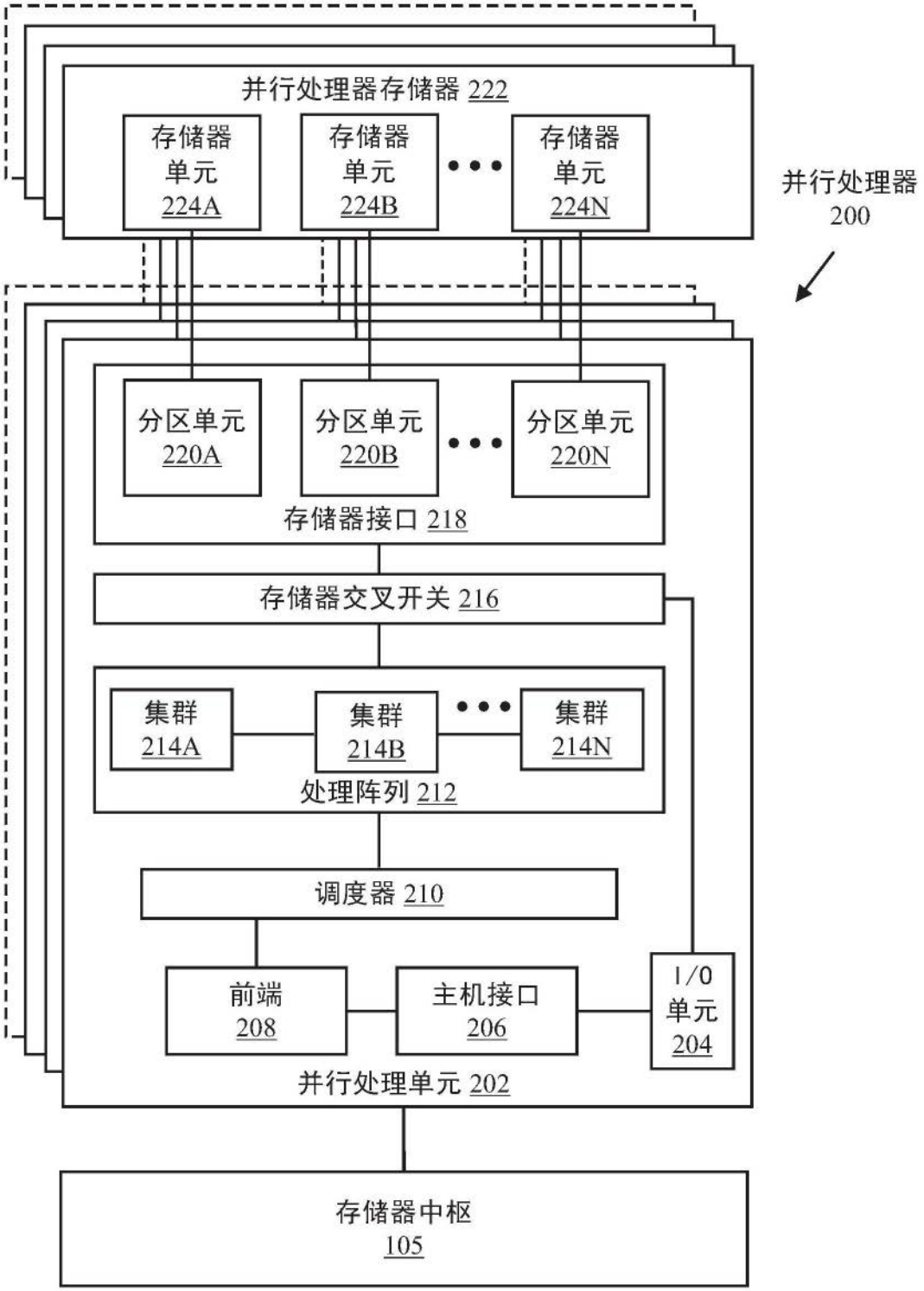

并行处理器存储器 222
存储器单元 224A
存储器单元 224B
存储器单元 224N
并行处理器 200
分区单元 220A
分区单元 220B
分区单元 220N
存储器接口 218
存储器交叉开关 216
集群 214A
集群 214B
集群 214N
处理阵列 212
调度器 210
前端 208
主机接口 206
I/O 单元 204
并行处理单元 202
存储器中枢 105

图2A

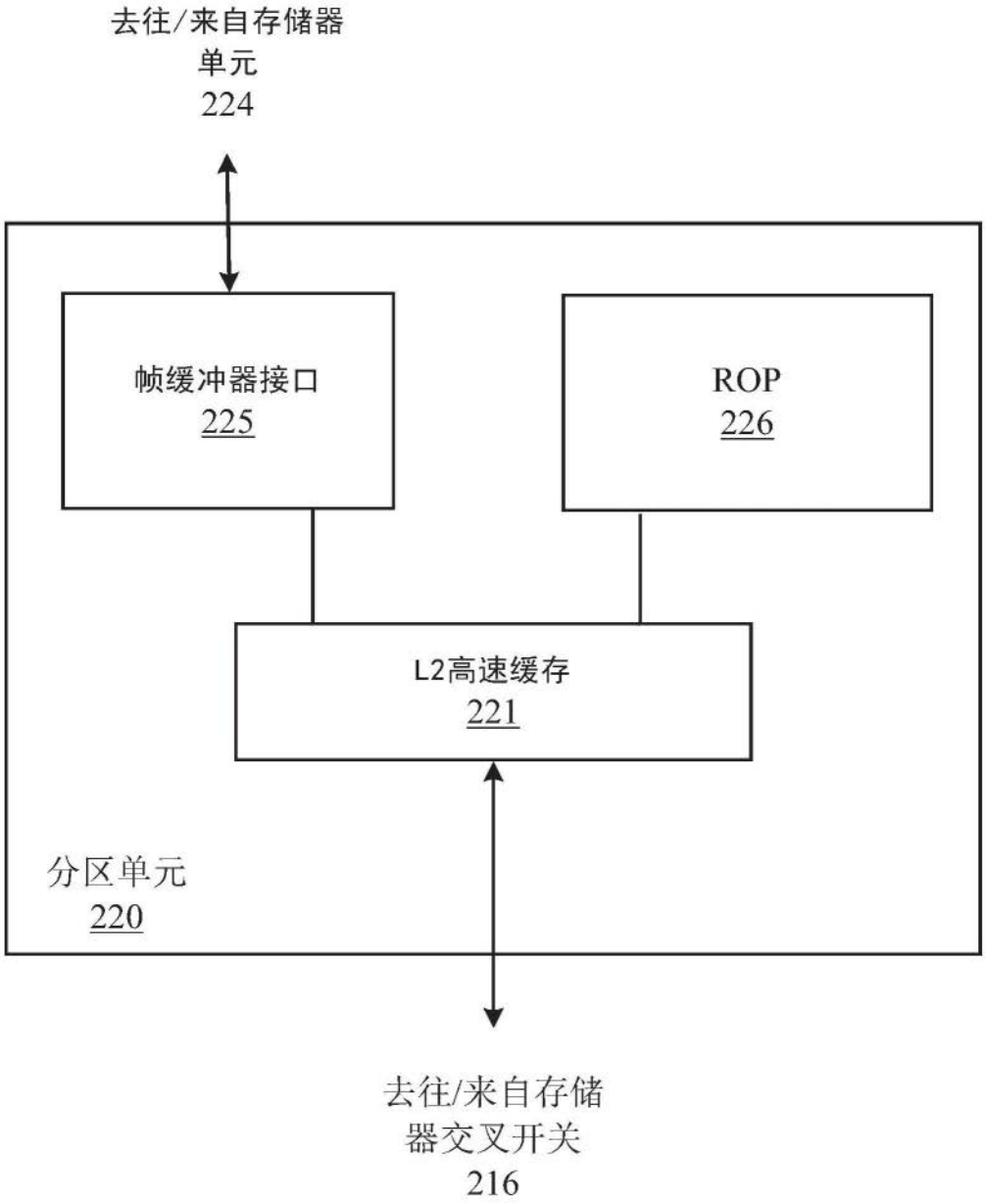

去往/来自存储器
单元
224
帧缓冲器接口
225
ROP
226
L2高速缓存
221
分区单元
220
去往/来自存储
器交叉开关
216

图2B

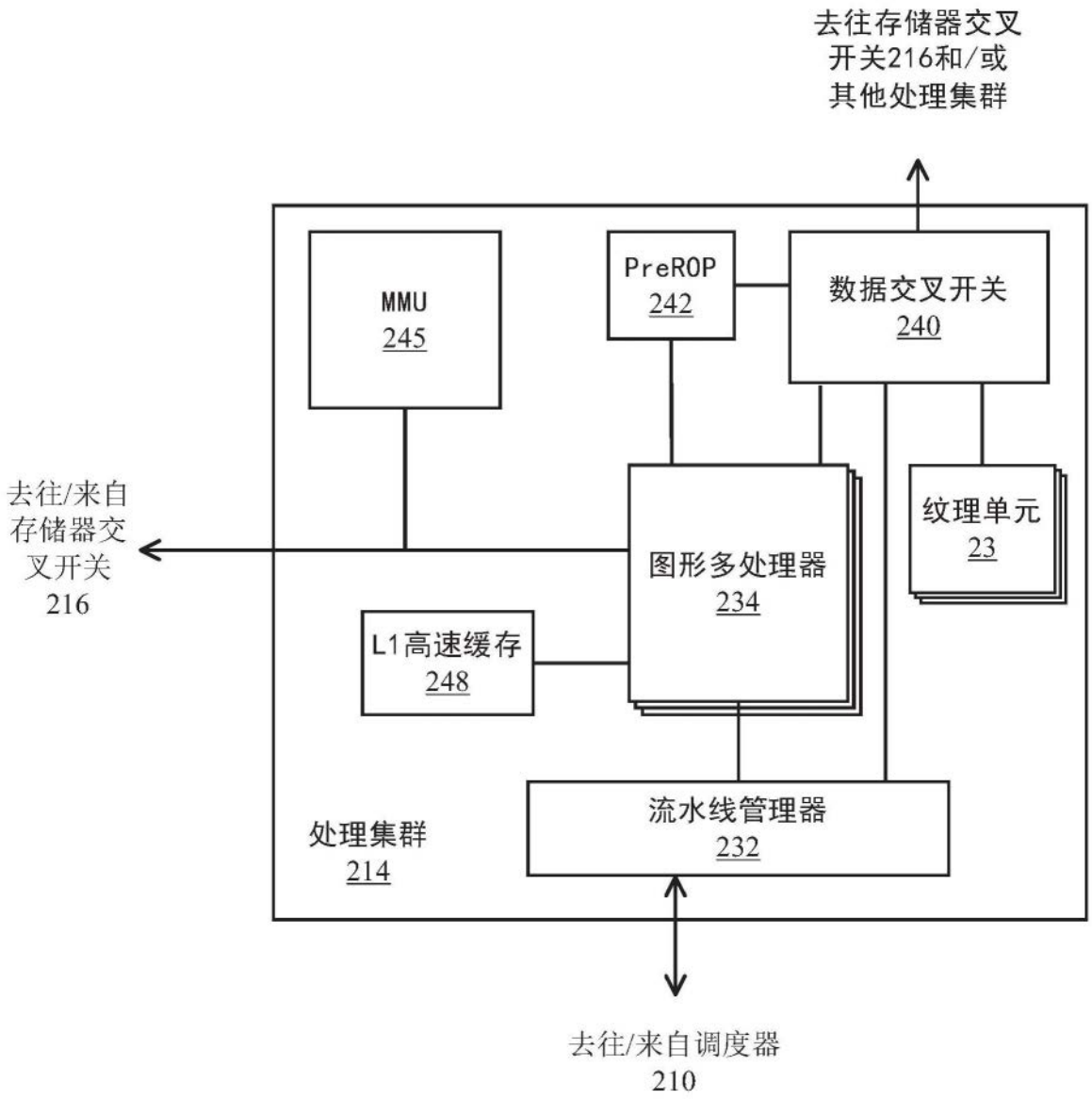

去往存储器交叉
开关216和/或
其他处理集群
MMU
245
PreROP
242
数据交叉开关
240
去往/来自
存储器交
叉开关
216
图形多处理器
234
纹理单元
23
L1高速缓存
248
流水线管理器
232
处理集群
214
去往/来自调度器
210

图2C

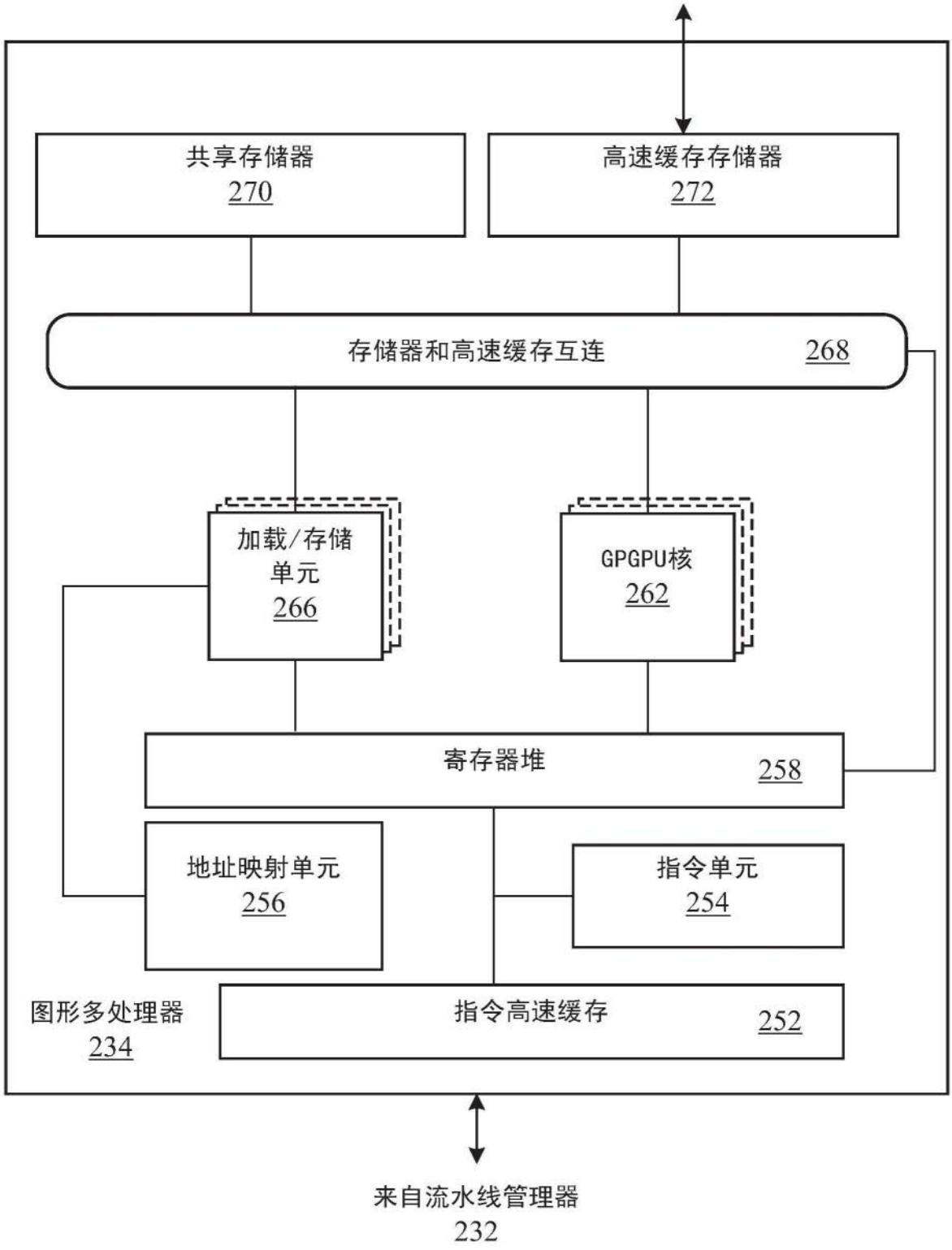

共享存储器
270
高速缓存存储器
272
存储器和高速缓存互连
268
加载/存储
单元
266
GPGPU核
262
寄存器堆
258
地址映射单元
256
指令单元
254
图形多处理器
234
指令高速缓存
252
来自流水线管理器
232

图2D


图形多处理器 325
互连结构 327
共享存储器 346
（多个）纹理单元 344A
（多个）纹理单元 344B
高速缓存存储器 342
加载/存储单元 340A
GPGPU 核 338A
GPGPU 核 337A
GPGPU 核 336A
加载/存储单元 340B
GPGPU 核 338B
GPGPU 核 337B
GPGPU 核 336B
寄存器堆 334A
寄存器堆 334B
指令单元 332A
指令单元 332B
指令高速缓存 330

图3A


图形多处理器 350
互连结构 352
共享存储器 362
（多个）纹理单元 360A
（多个）纹理单元 360B
高速缓存存储器 358A
执行资源 356A
执行资源 356B
（多个）纹理单元 360C
（多个）纹理单元 360D
高速缓存存储器 358B
执行资源 356C
执行资源 356D
指令高速缓存 354

图3B

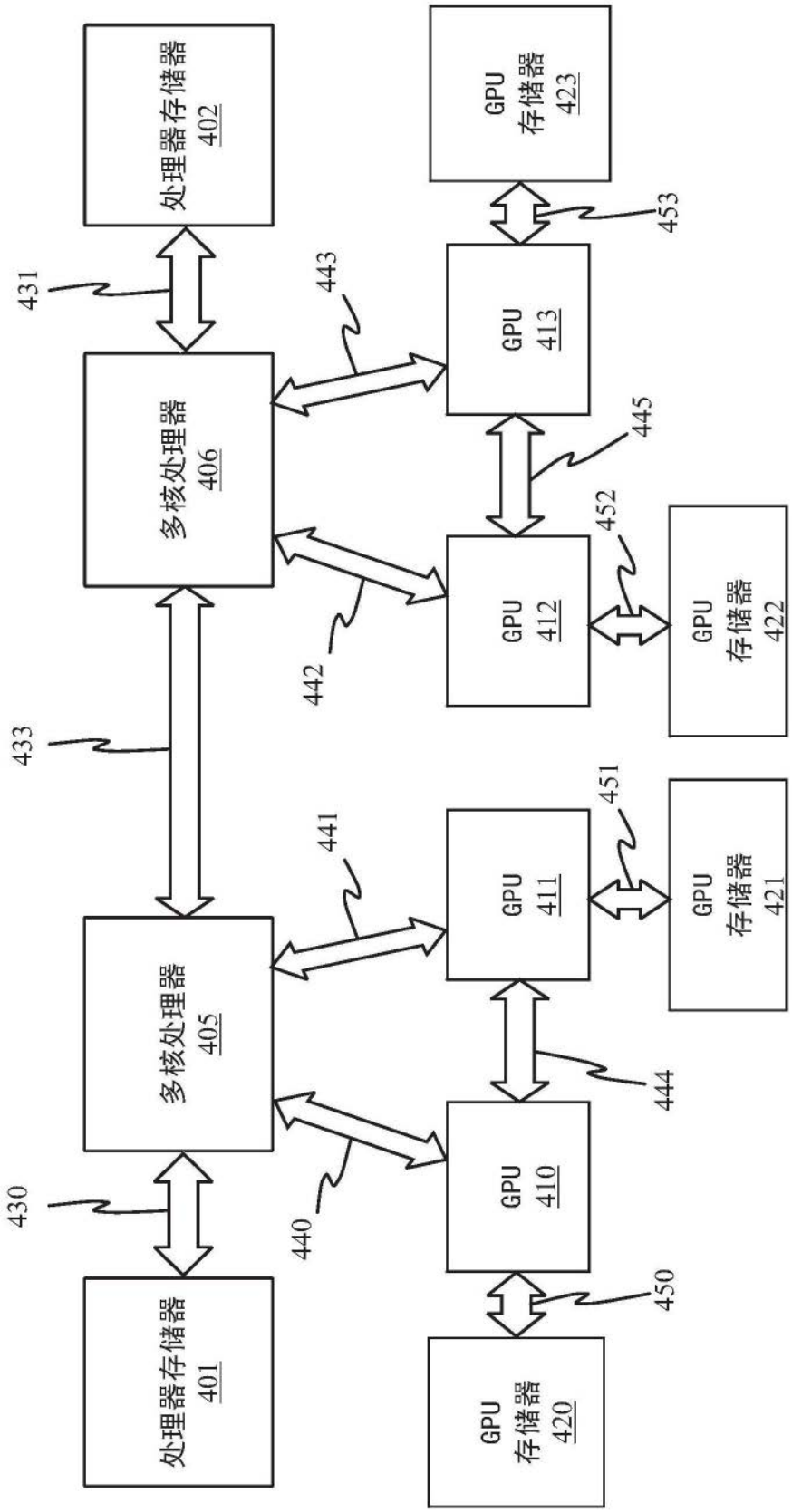

处理器存储器 402
GPU 存储器 423
453
431
443
GPU 413
多核处理器 406
445
452
GPU 412
GPU 存储器 422
442
433
441
451
GPU 411
GPU 存储器 421
多核处理器 405
444
430
440
GPU 410
450
处理器存储器 401
GPU 存储器 420

图4A

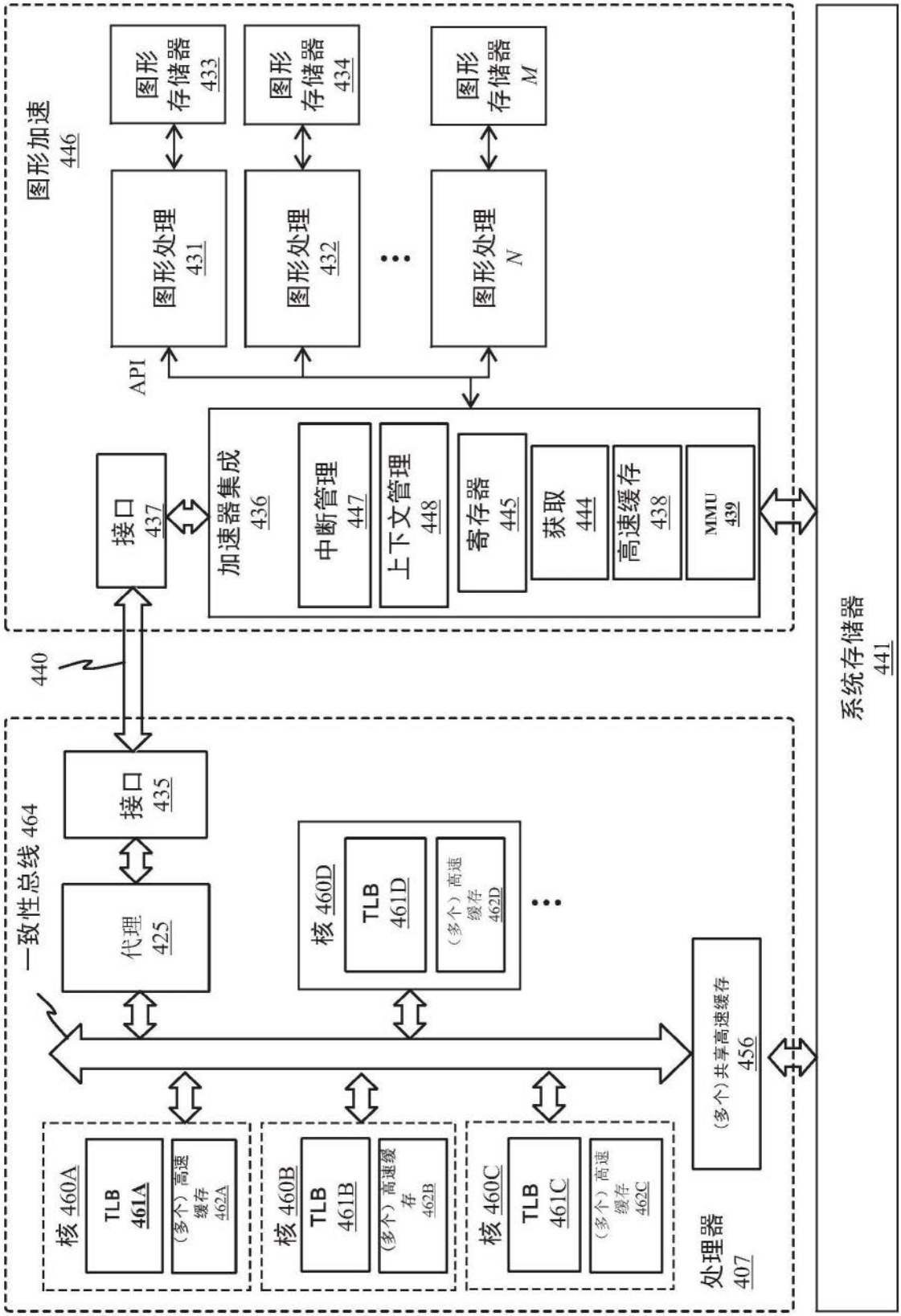

图形加速
446
图形存储器
433
图形存储器
434
图形存储器
M
图形处理
431
图形处理
432
图形处理
N
API
接口
437
加速器集成
436
中断管理
447
上下文管理
448
寄存器
445
获取
444
高速缓存
438
MMU
439
440
一致性总线 464
接口
435
代理
425
核 460D
TLB
461D
（多个）高速缓存
462D
核 460A
TLB
461A
（多个）高速缓存
462A
核 460B
TLB
461B
（多个）高速缓存
462B
核 460C
TLB
461C
（多个）高速缓存
462C
（多个）共享高速缓存
456
处理器
407
系统存储器
441

图4B

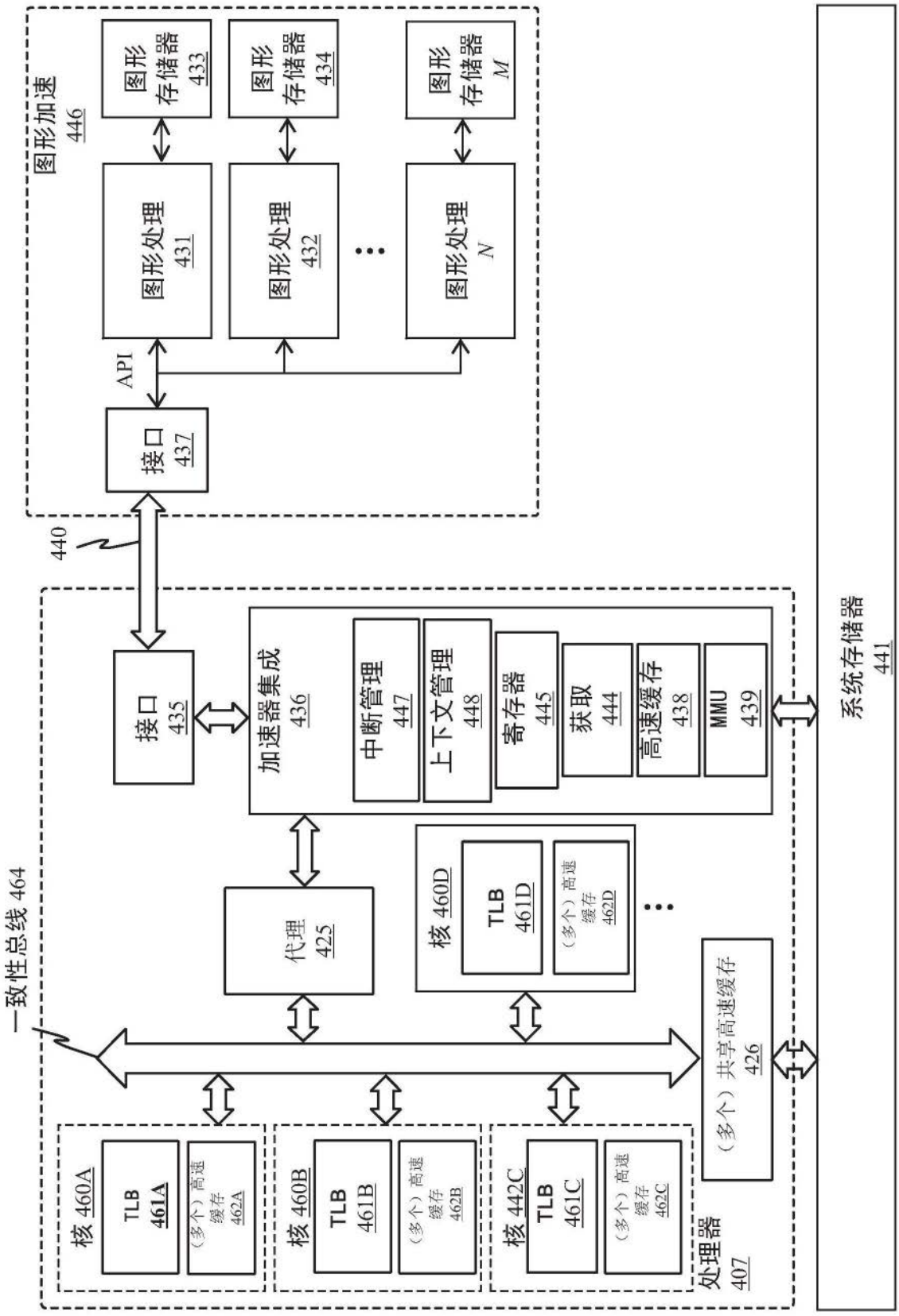

图形加速 446
图形存储器 433
图形存储器 434
图形存储器 M
图形处理 431
图形处理 432
图形处理 N
API
接口 437
440
接口 435
加速器集成 436
中断管理 447
上下文管理 448
寄存器 445
获取 444
高速缓存 438
MMU 439
代理 425
核 460D
TLB 461D
（多个）高速缓存 462D
一致性总线 464
（多个）共享高速缓存 426
系统存储器 441
核 460A
TLB 461A
（多个）高速缓存 462A
核 460B
TLB 461B
（多个）高速缓存 462B
核 442C
TLB 461C
（多个）高速缓存 462C
处理器 407

图4C

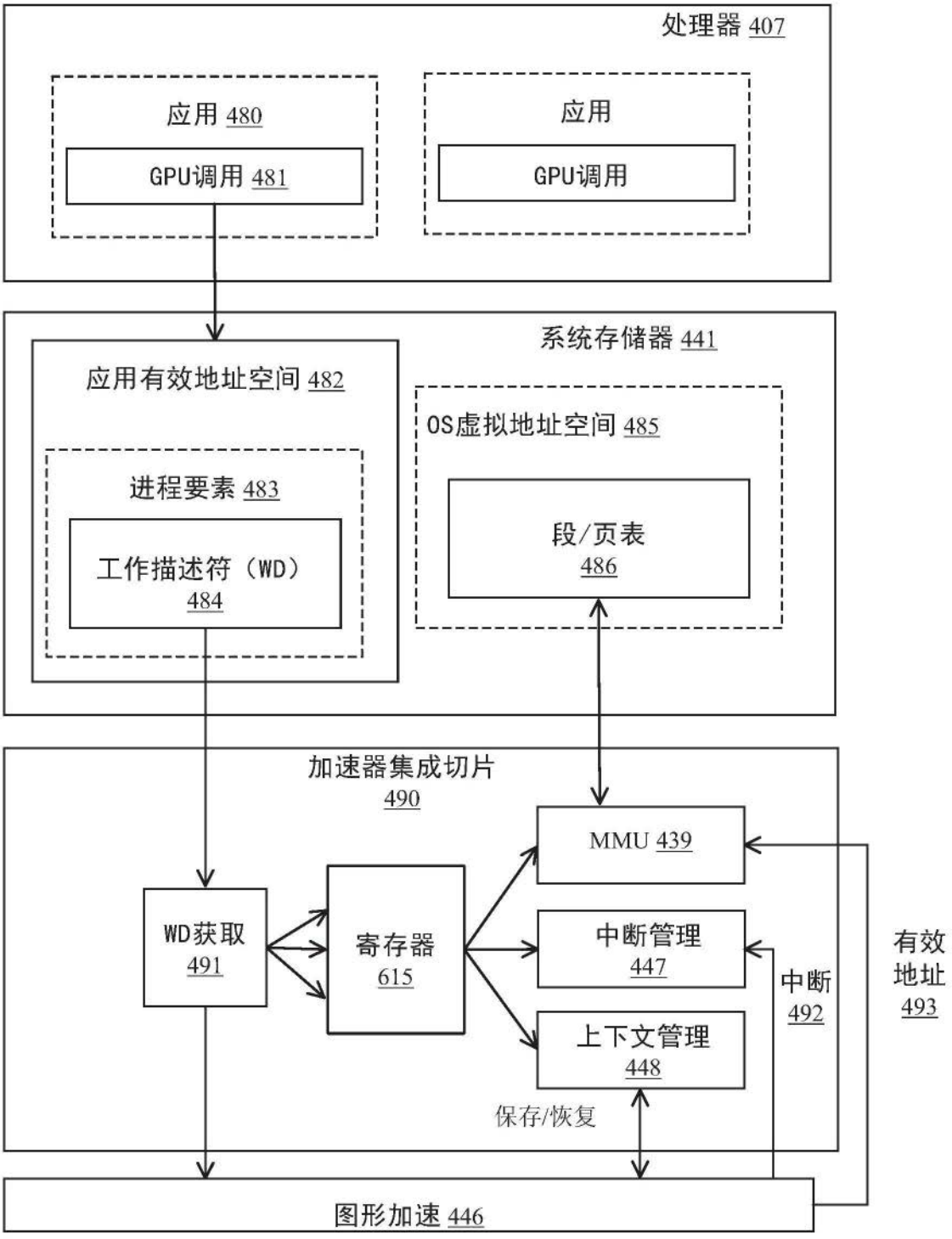

处理器 407
应用 480
GPU调用 481
应用
GPU调用
系统存储器 441
应用有效地址空间 482
进程要素 483
工作描述符（WD）
484
OS虚拟地址空间 485
段/页表
486
加速器集成切片
490
MMU 439
WD获取
491
寄存器
615
中断管理
447
上下文管理
448
保存/恢复
中断
492
有效
地址
493
图形加速 446

图4D

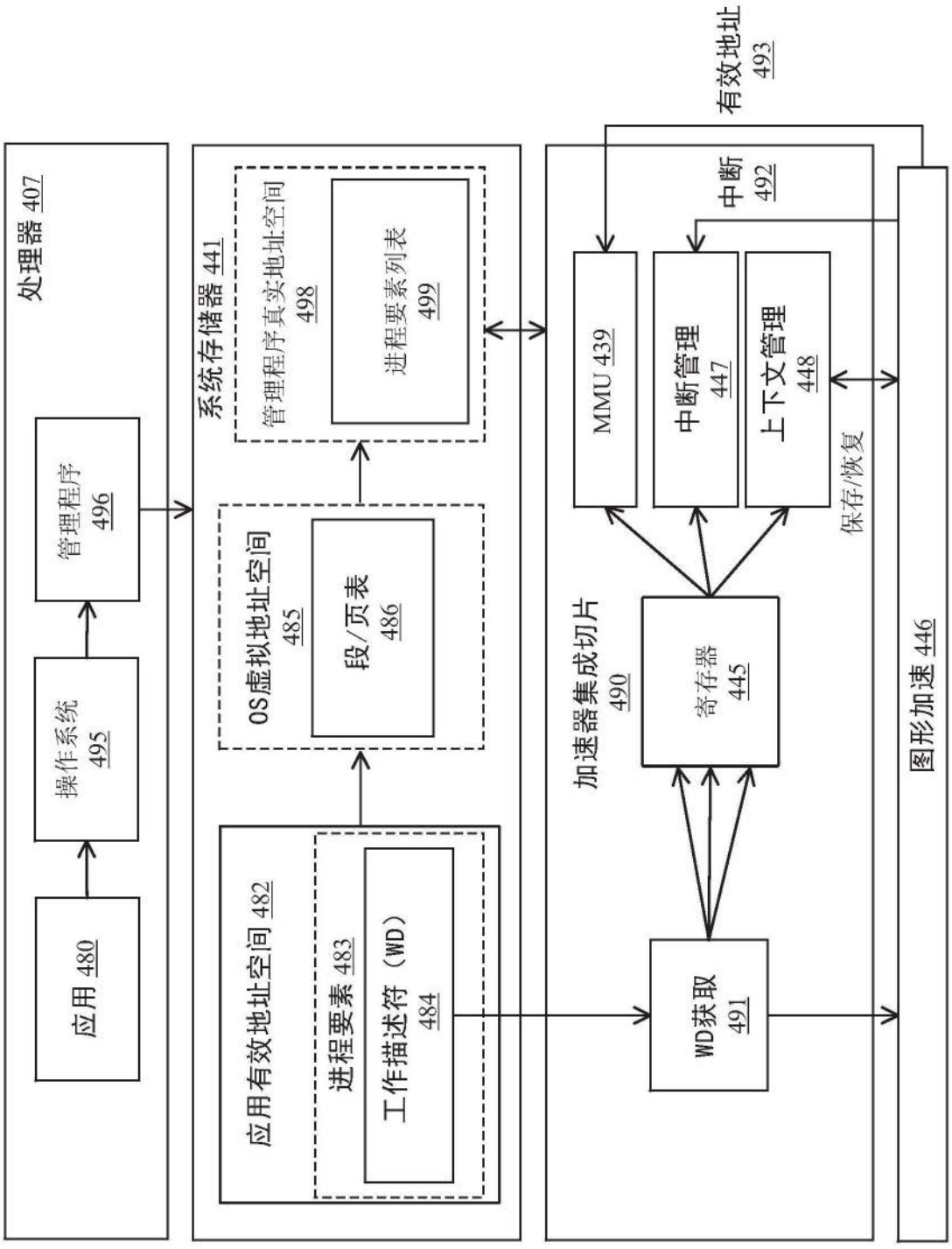

处理器 407
应用 480
操作系统 495
管理程序 496
系统存储器 441
应用有效地址空间 482
进程要素 483
工作描述符（WD） 484
OS虚拟地址空间 485
段/页表 486
管理程序真实地址空间 498
进程要素列表 499
加速器集成切片 490
WD获取 491
寄存器 445
MMU 439
中断管理 447
上下文管理 448
保存/恢复
中断 492
有效地址 493
图形加速 446

图4E

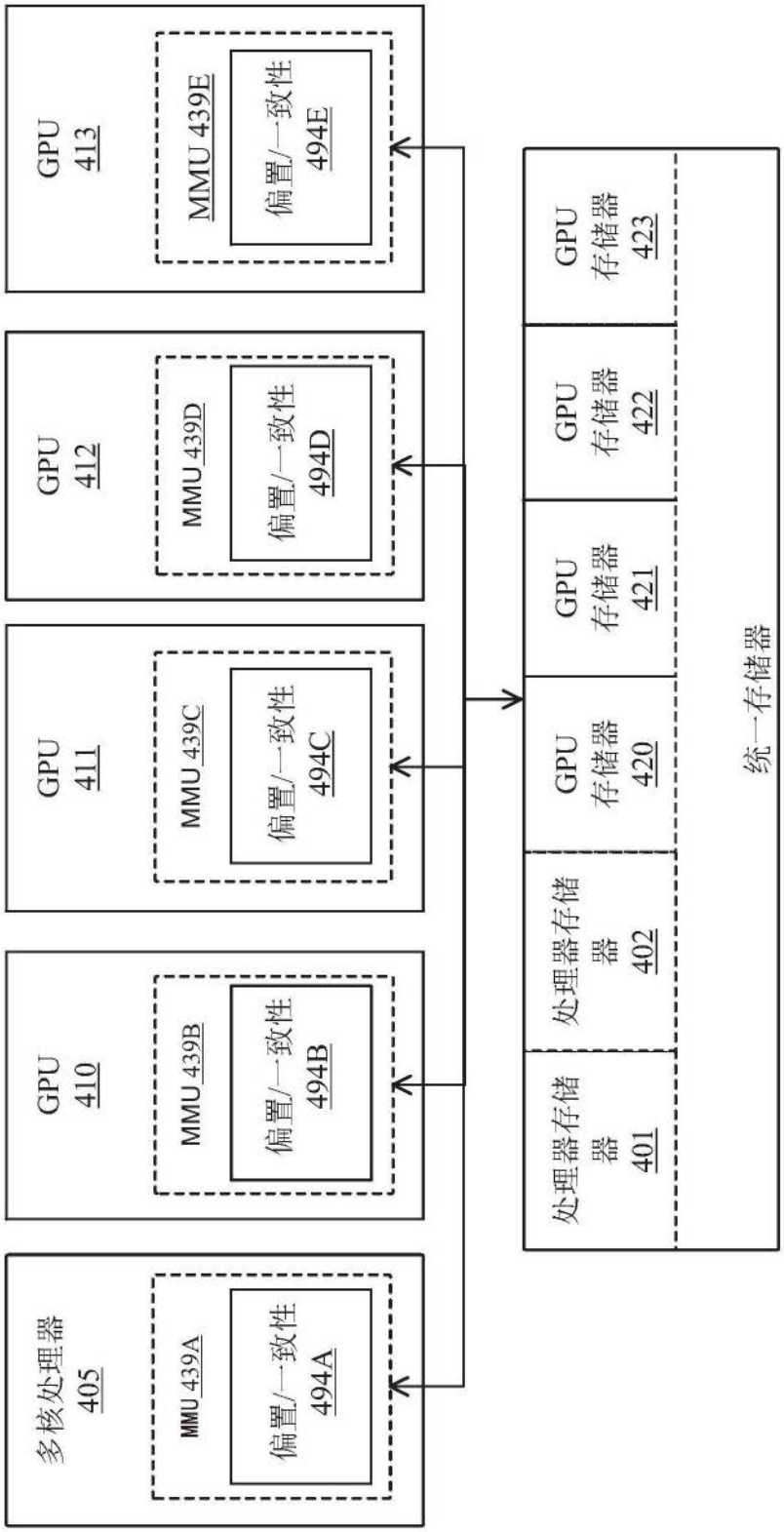

多核处理器 405
MMU 439A
偏置/一致性 494A
GPU 410
MMU 439B
偏置/一致性 494B
GPU 411
MMU 439C
偏置/一致性 494C
GPU 412
MMU 439D
偏置/一致性 494D
GPU 413
MMU 439E
偏置/一致性 494E
处理器存储器 401
处理器存储器 402
GPU 存储器 420
GPU 存储器 421
GPU 存储器 422
GPU 存储器 423
统一存储器

图4F

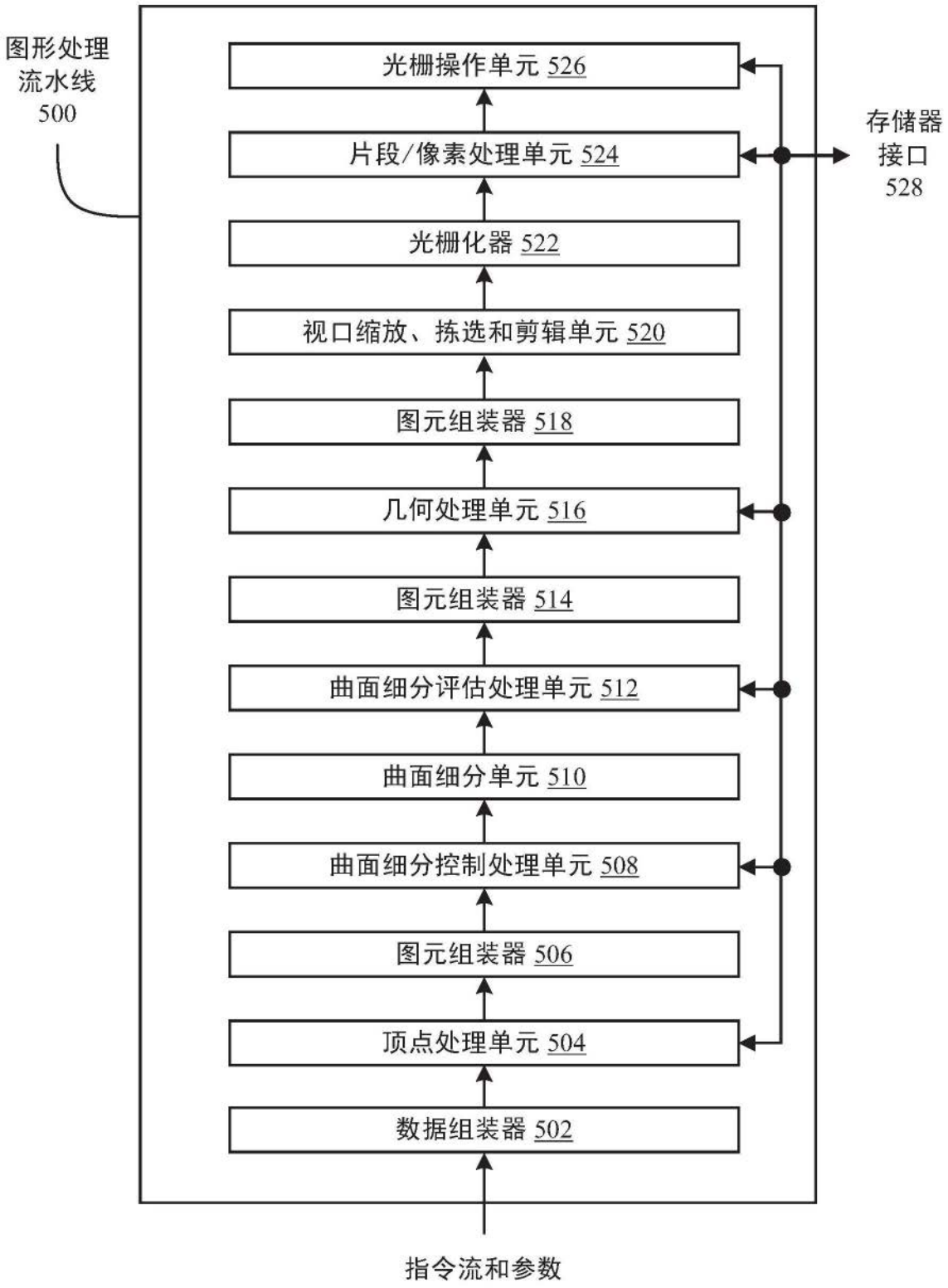

图形处理
流水线
500
光栅操作单元 526
片段/像素处理单元 524
光栅化器 522
视口缩放、拣选和剪辑单元 520
图元组装器 518
几何处理单元 516
图元组装器 514
曲面细分评估处理单元 512
曲面细分单元 510
曲面细分控制处理单元 508
图元组装器 506
顶点处理单元 504
数据组装器 502
存储器
接口
528
指令流和参数

图5

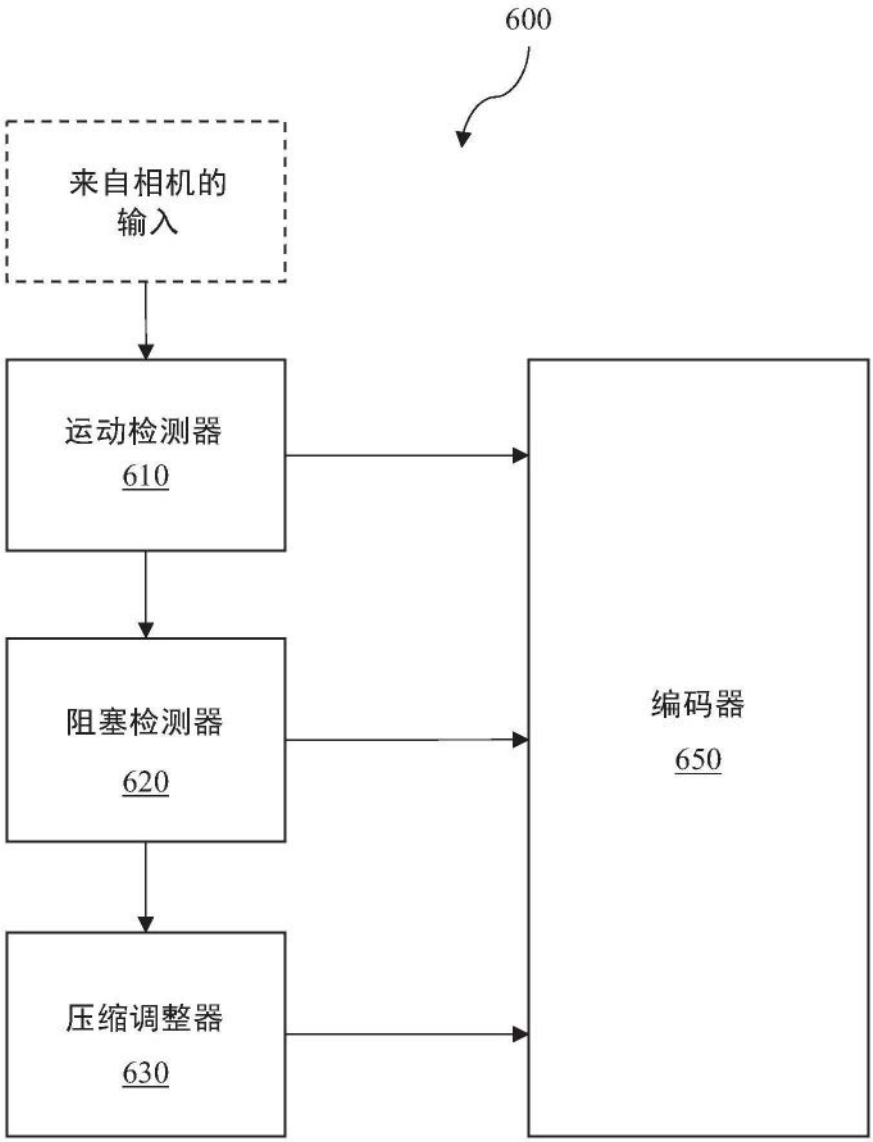


图6

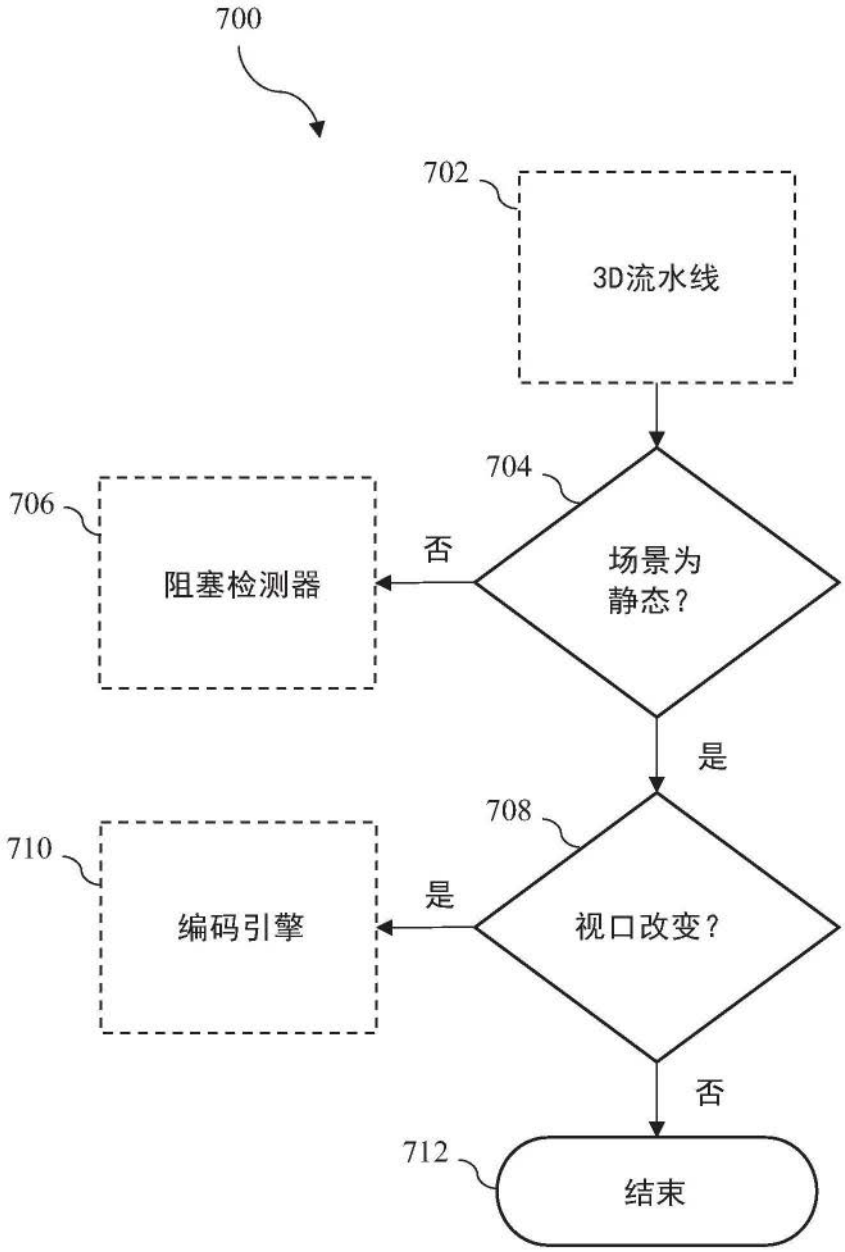

700
702
3D流水线
704
场景为静态？
否
706
阻塞检测器
是
708
视口改变？
是
710
编码引擎
否
712
结束

图7A

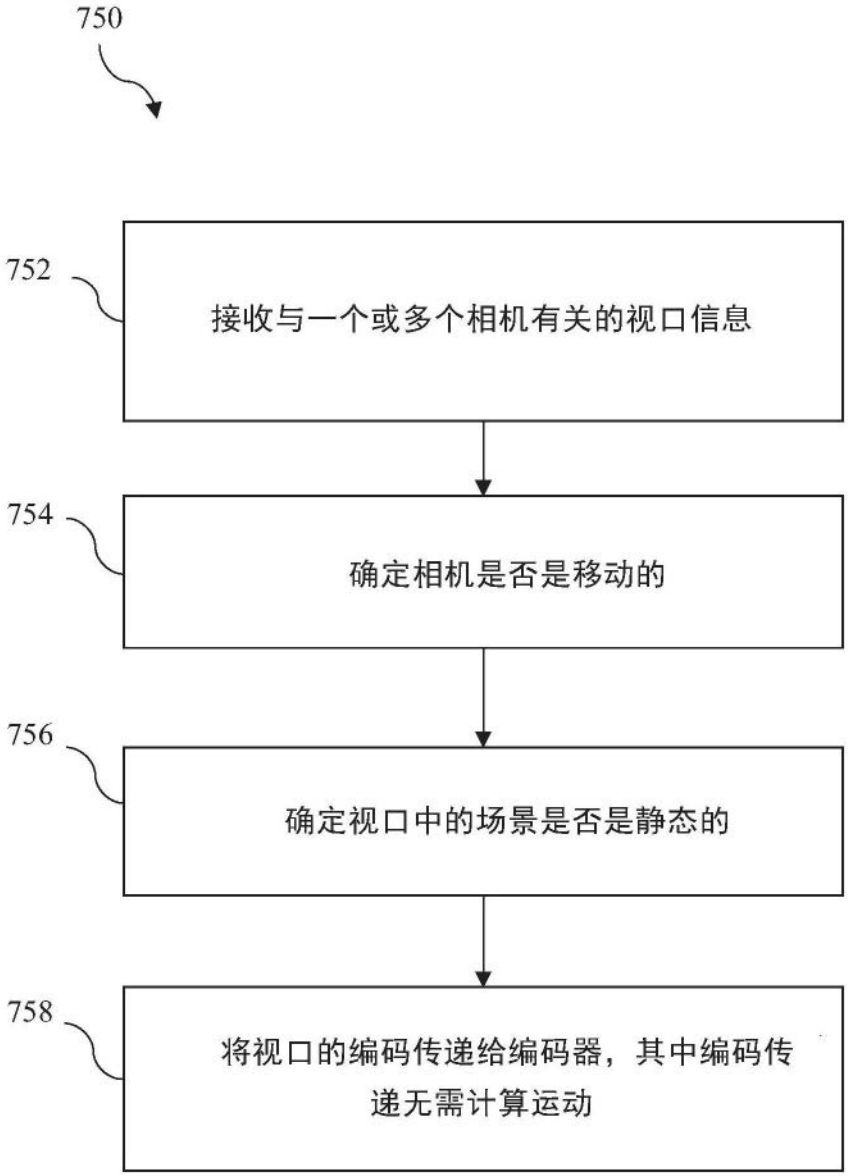

750
752
接收与一个或多个相机有关的视口信息
754
确定相机是否是移动的
756
确定视口中的场景是否是静态的
758
将视口的编码传递给编码器，其中编码传递无需计算运动

图7B

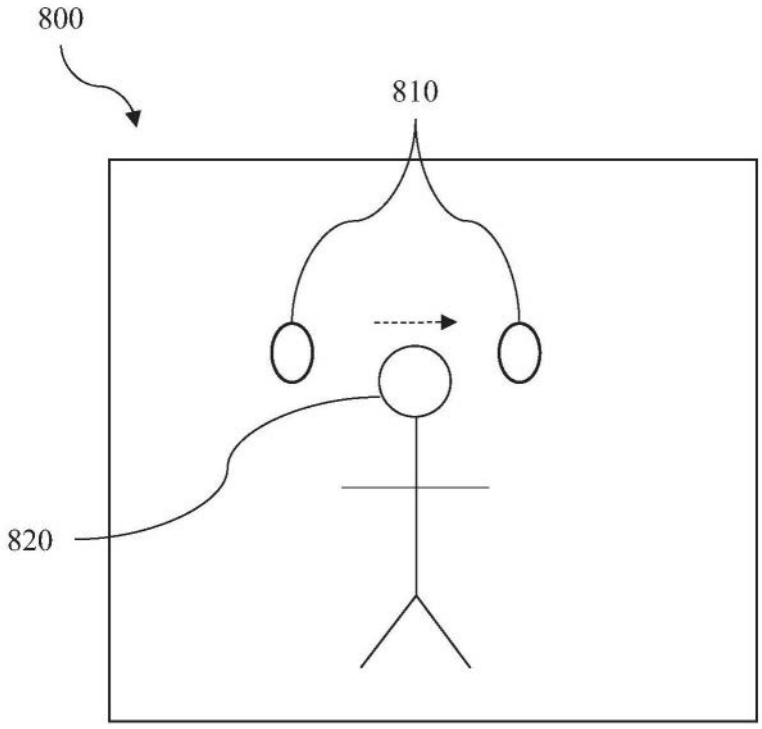


图8A

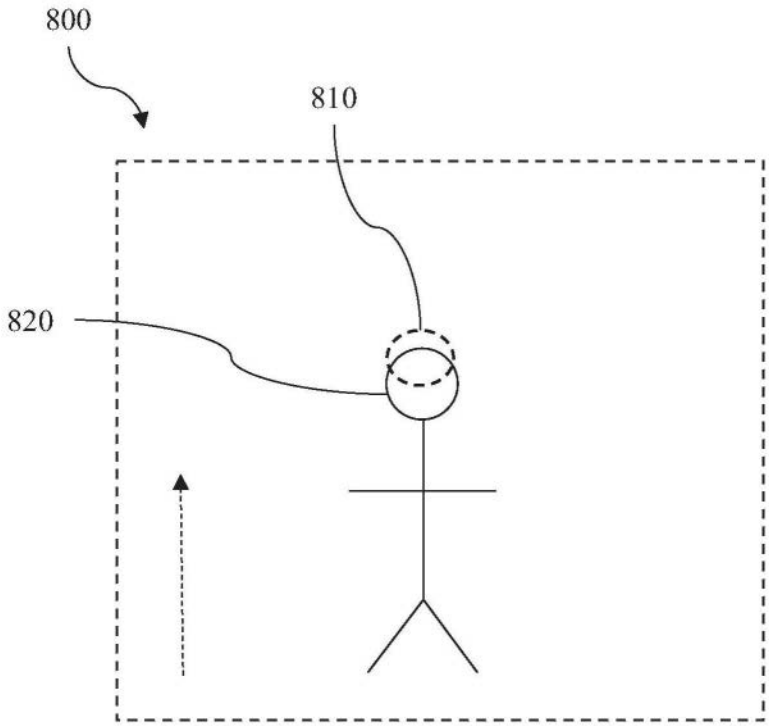


图8B

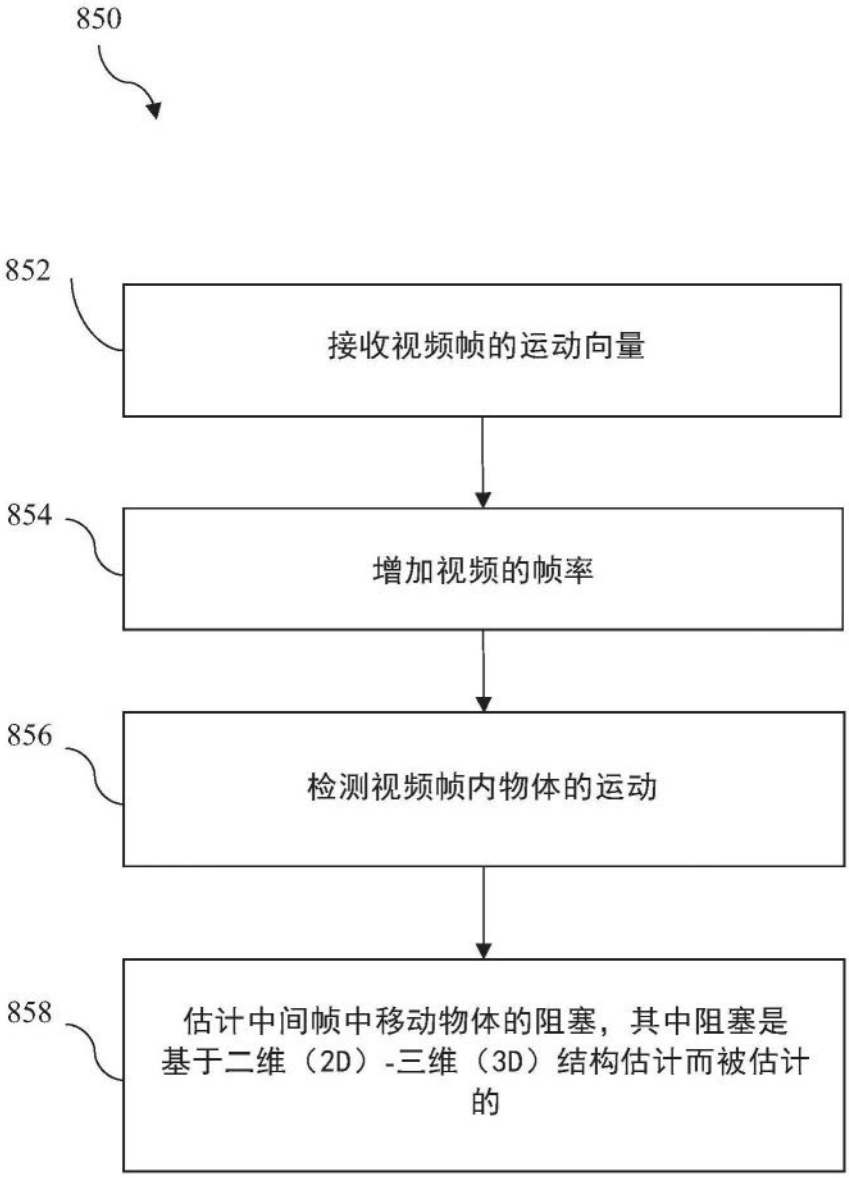

850
852
接收视频帧的运动向量
854
增加视频的帧率
856
检测视频帧内物体的运动
858
估计中间帧中移动物体的阻塞，其中阻塞是基于二维（2D）-三维（3D）结构估计而被估计的

图8C

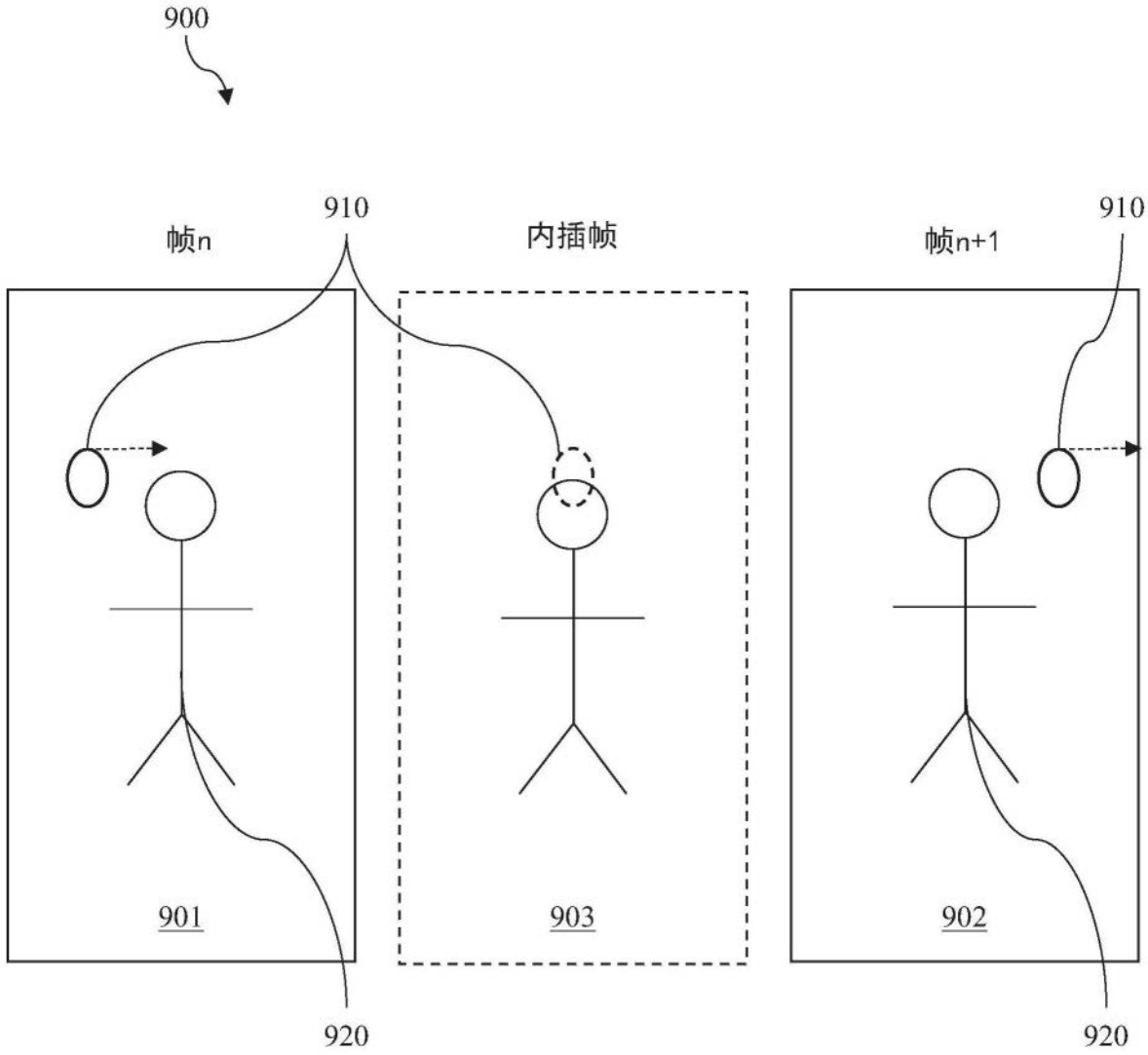


图9A

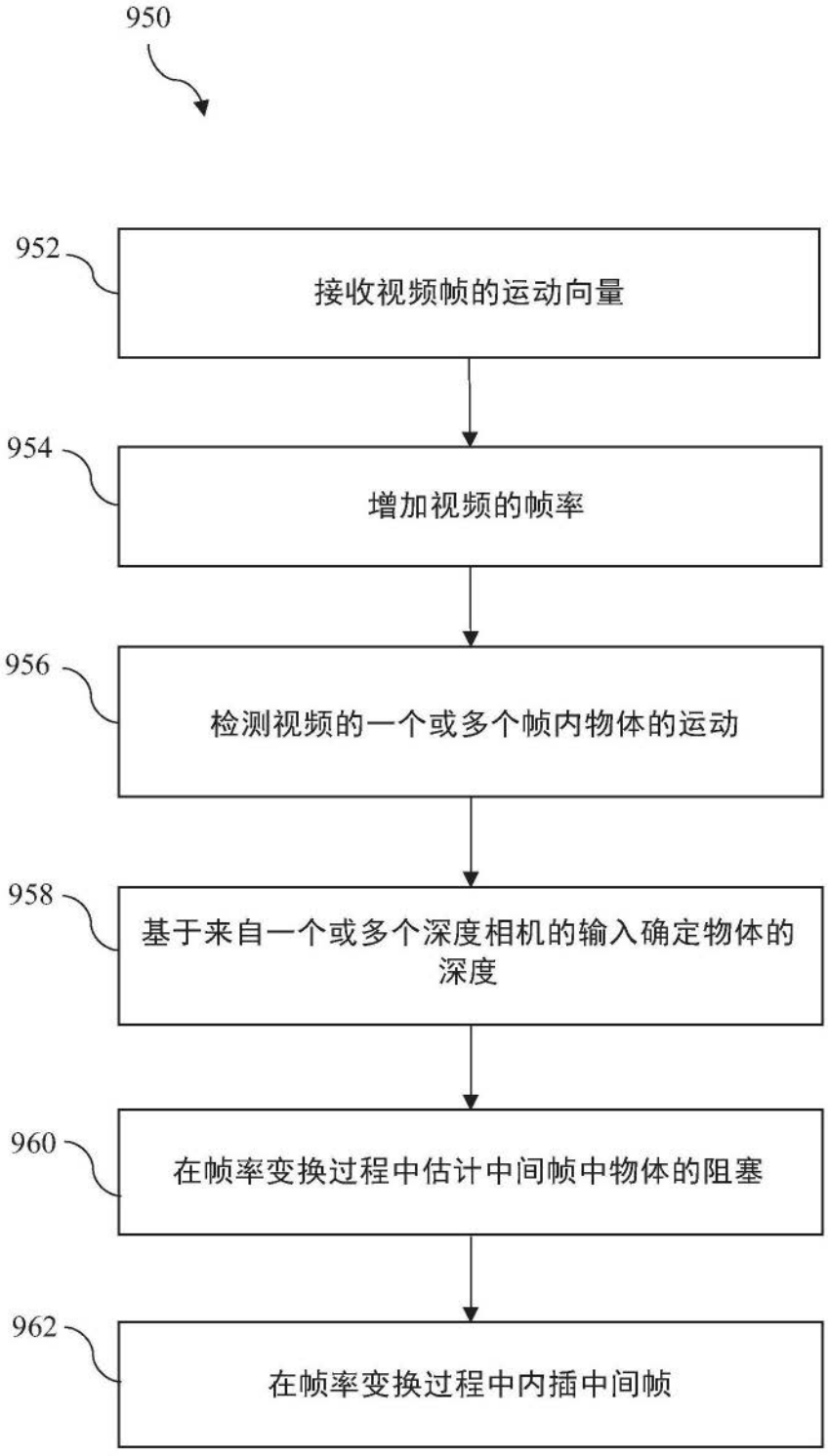

950
952
接收视频帧的运动向量
954
增加视频的帧率
956
检测视频的一个或多个帧内物体的运动
958
基于来自一个或多个深度相机的输入确定物体的深度
960
在帧率变换过程中估计中间帧中物体的阻塞
962
在帧率变换过程中内插中间帧

图9B

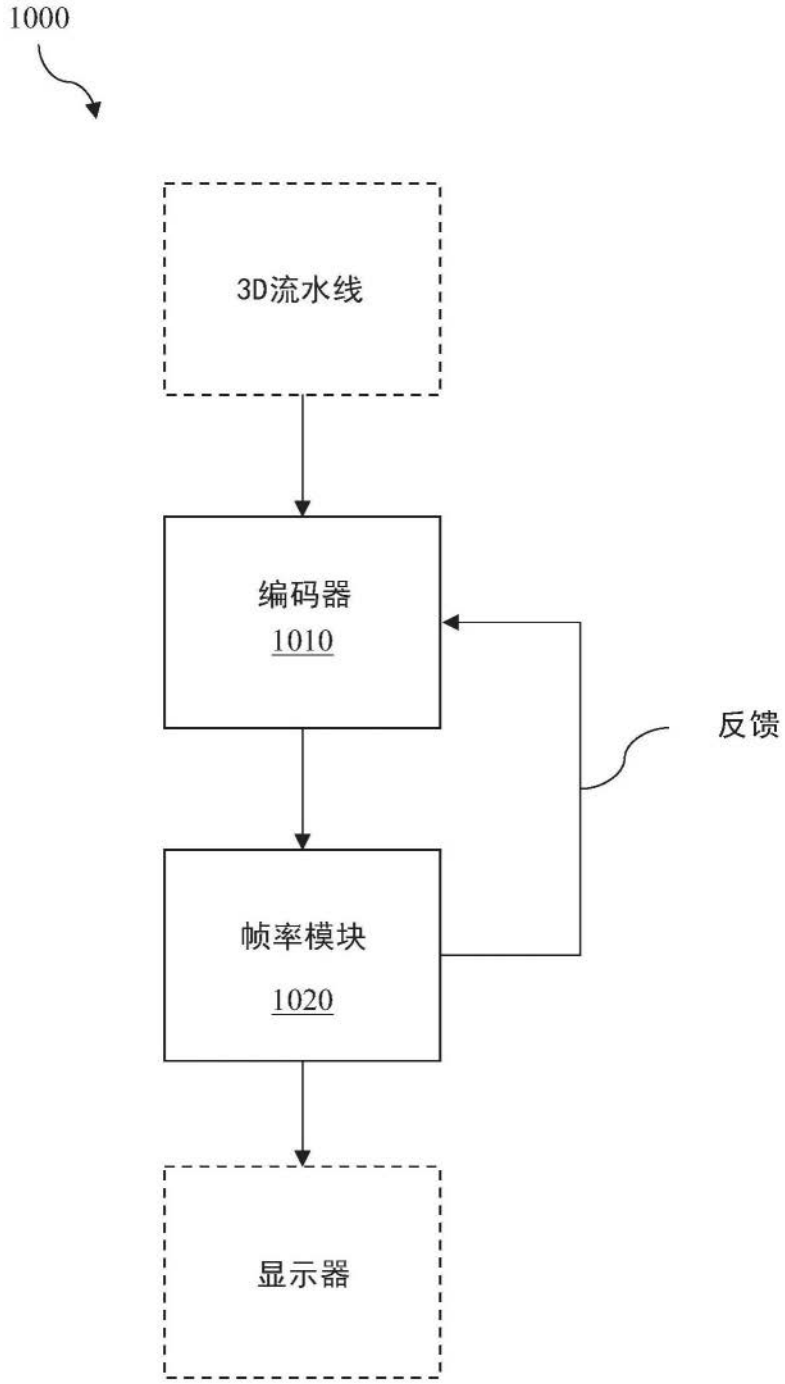


图10A

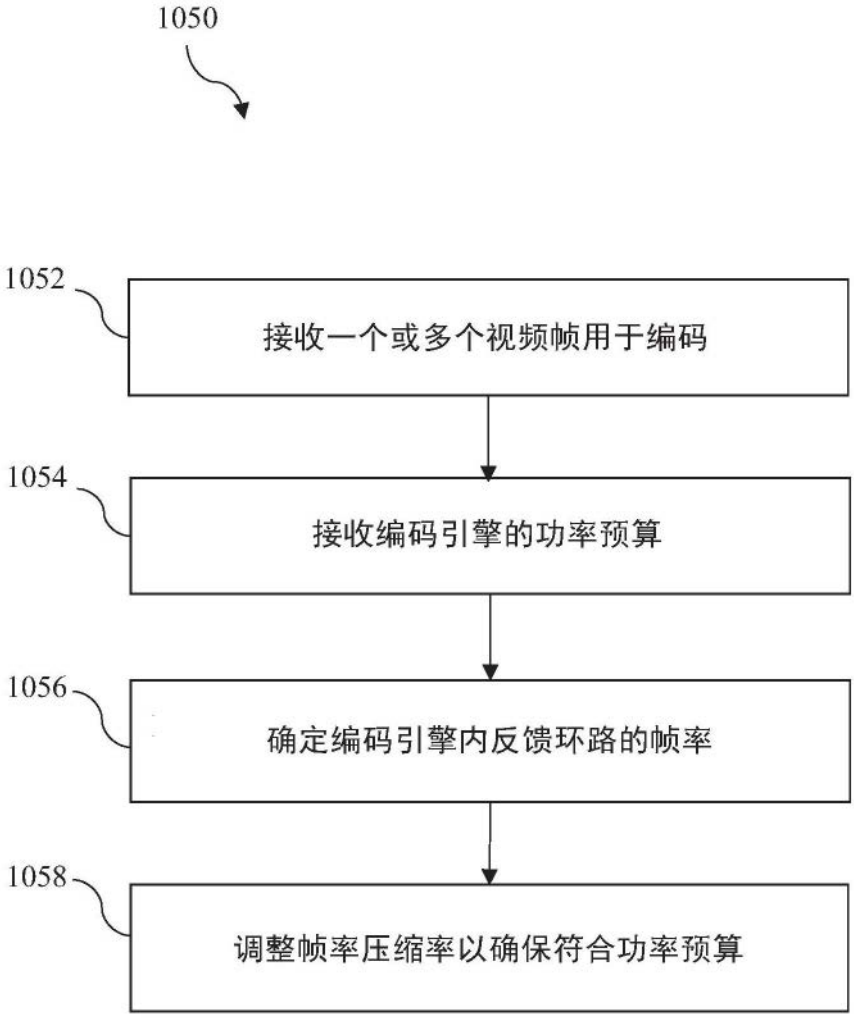

1050
1052
接收一个或多个视频帧用于编码
1054
接收编码引擎的功率预算
1056
确定编码引擎内反馈环路的帧率
1058
调整帧率压缩率以确保符合功率预算

图10B

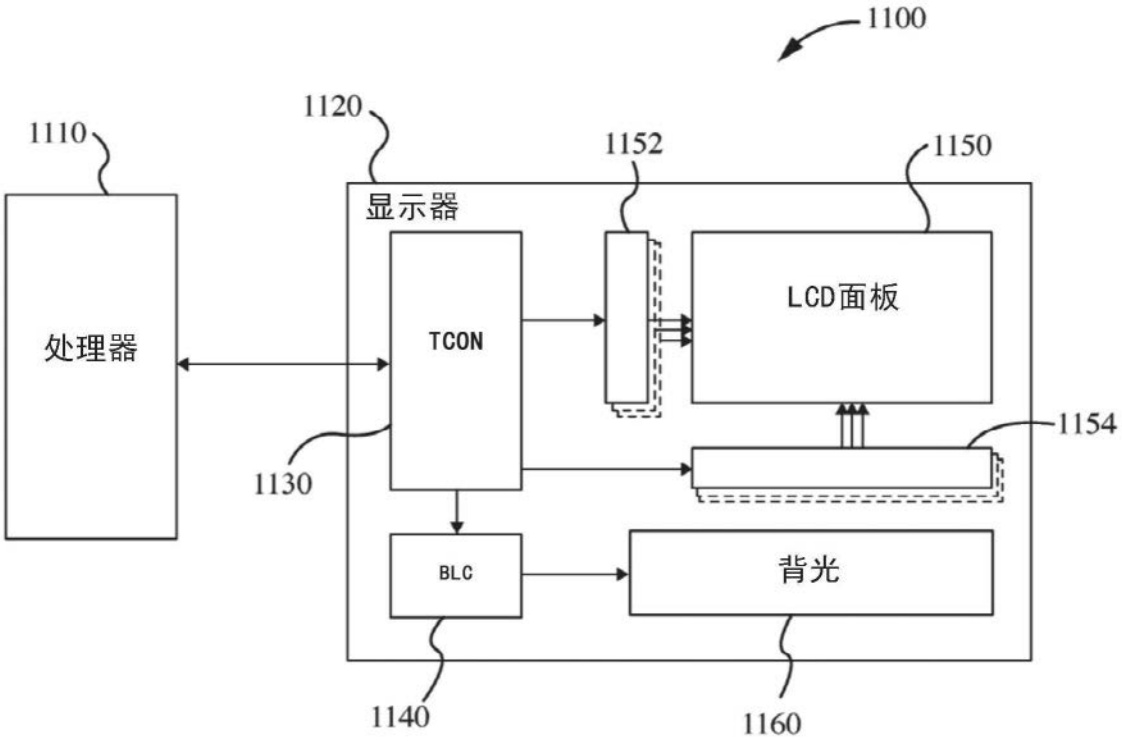

1100
1110
1120
1152
1150
显示器
处理器
TCON
LCD面板
1154
1130
BLC
背光
1140
1160

图11

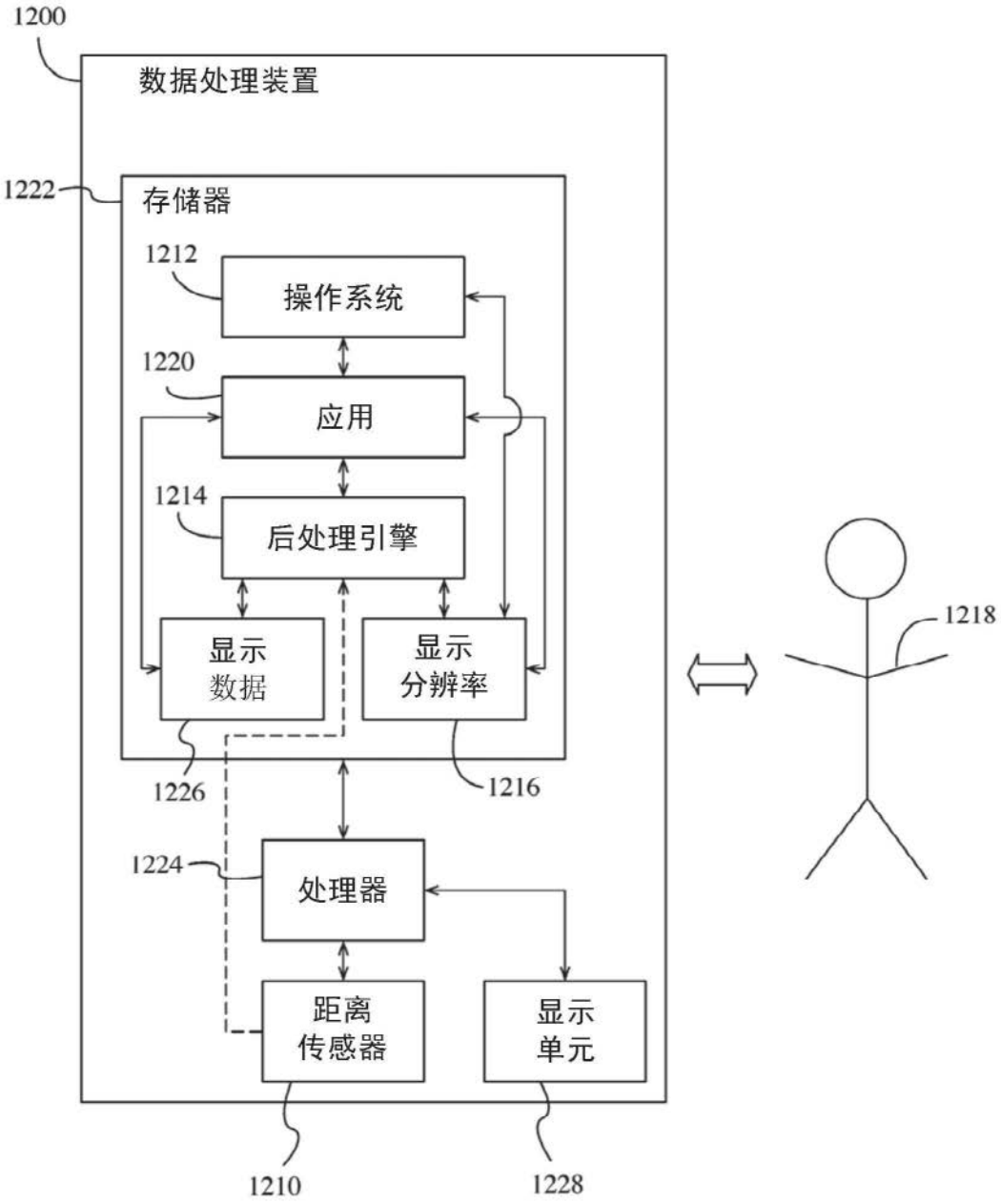

1200
数据处理装置
1222
存储器
1212
操作系统
1220
应用
1214
后处理引擎
显示
数据
显示
分辨率
1226
1216
1224
处理器
距离
传感器
显示
单元
1210
1228
1218

图12A

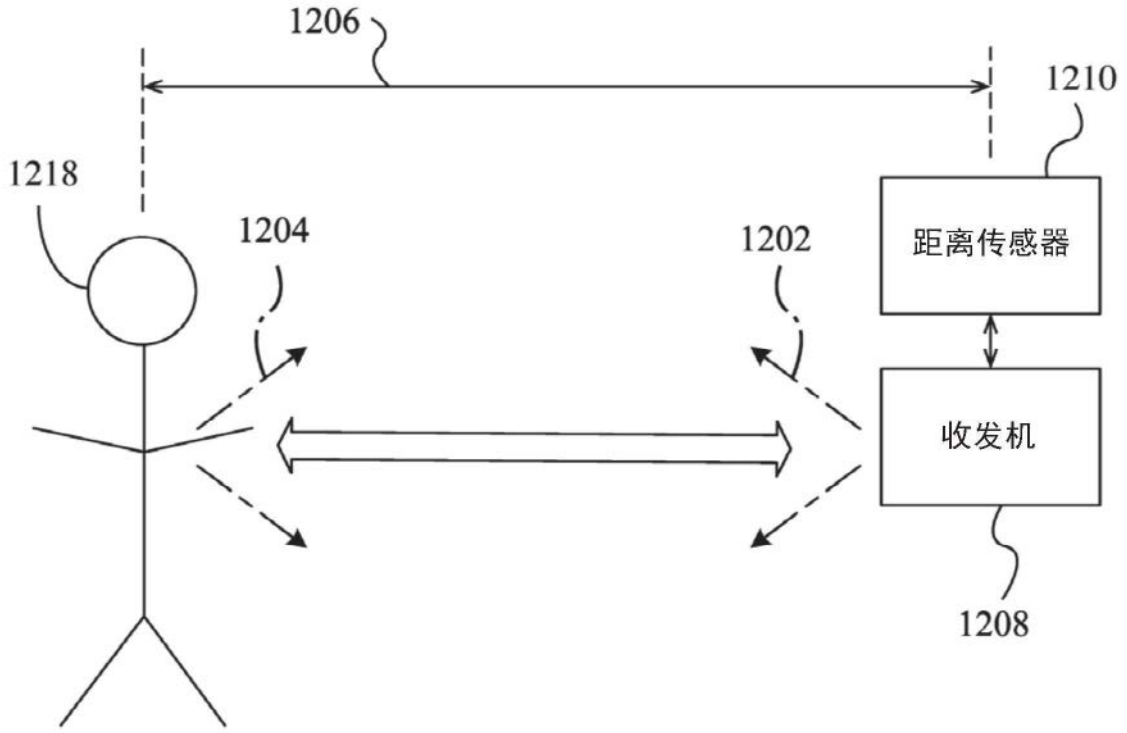


图12B

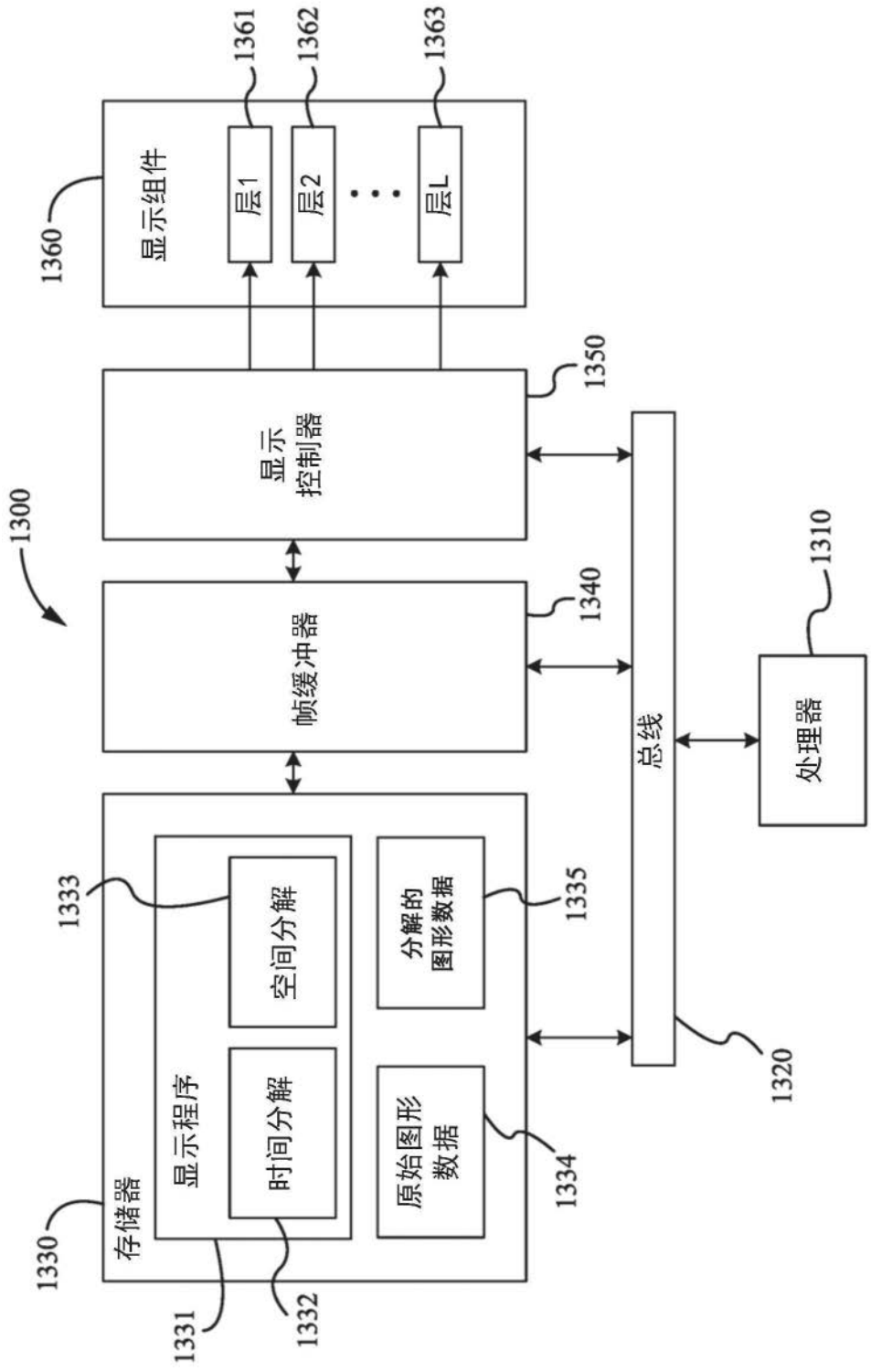

1361
1362
1363
显示组件
层1
层2
层L
1360
显示
控制器
1350
1300
帧缓冲器
1340
1310
总线
处理器
1333
空间分解
分解的
图形数据
1335
显示程序
时间分解
原始图形
数据
1334
存储器
1330
1320
1331
1332

图13

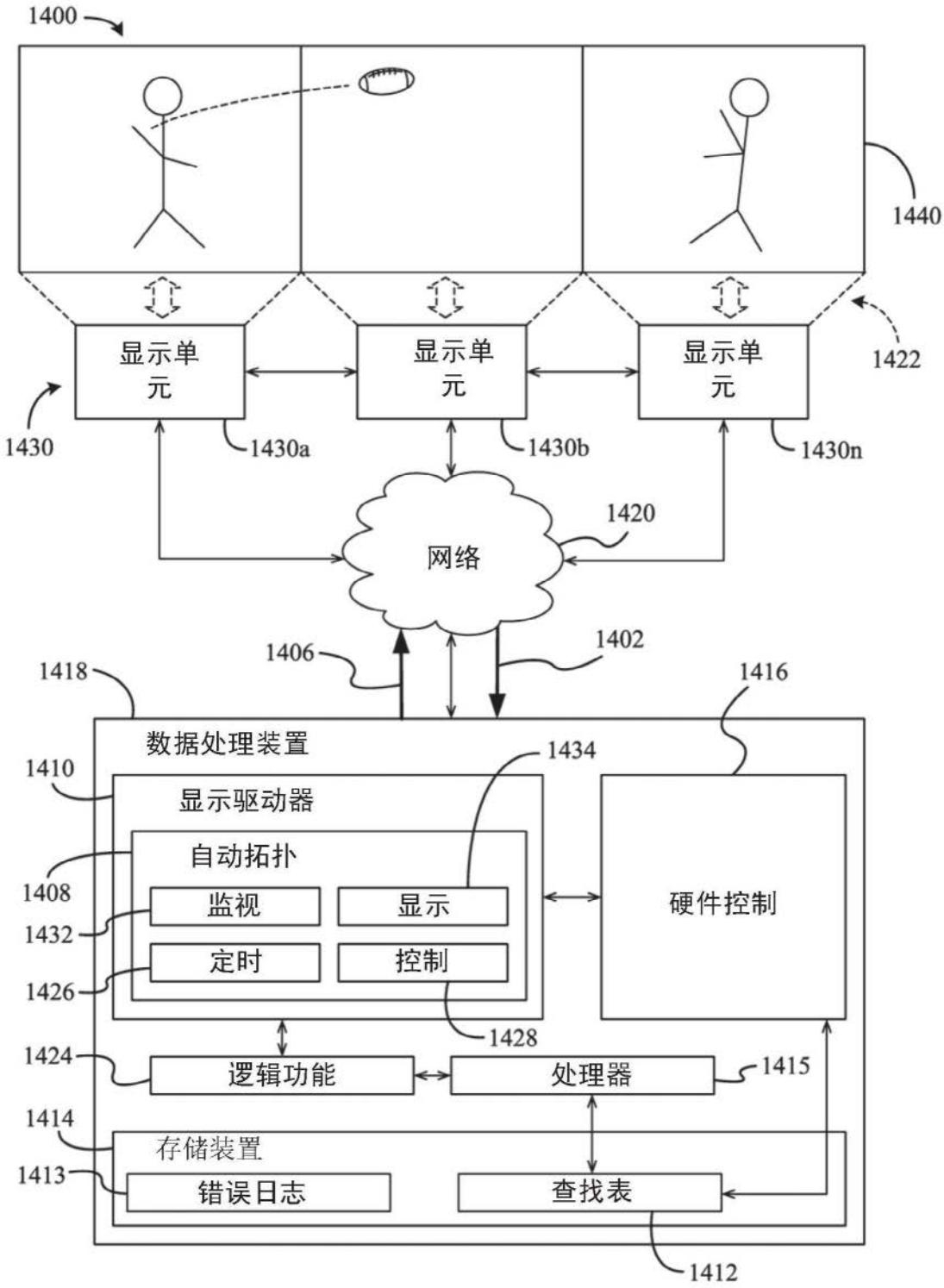

1400
1440
显示单元
显示单元
显示单元
1422
1430
1430a
1430b
1430n
1420
网络
1406
1402
1418
1416
数据处理装置
1410
1434
显示驱动器
自动拓扑
1408
监视
显示
1432
定时
控制
1426
硬件控制
1428
1424
逻辑功能
处理器
1415
1414
存储装置
1413
错误日志
查找表
1412

图14

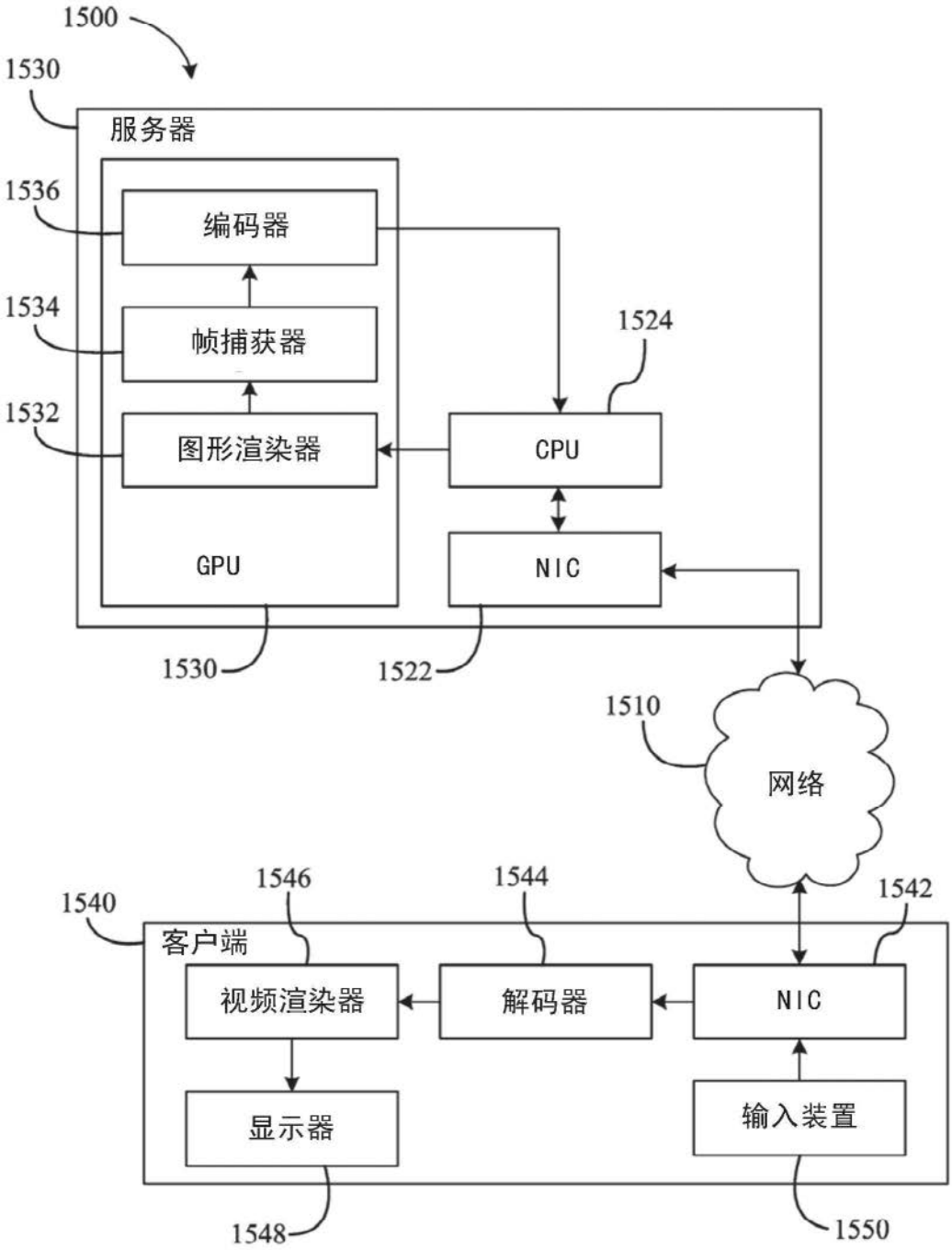

1500
1530
服务器
1536
编码器
1534
帧捕获器
1532
图形渲染器
GPU
1524
CPU
NIC
1530
1522
1510
网络
1540
客户端
1546
1544
1542
视频渲染器
解码器
NIC
显示器
输入装置
1548
1550

图15

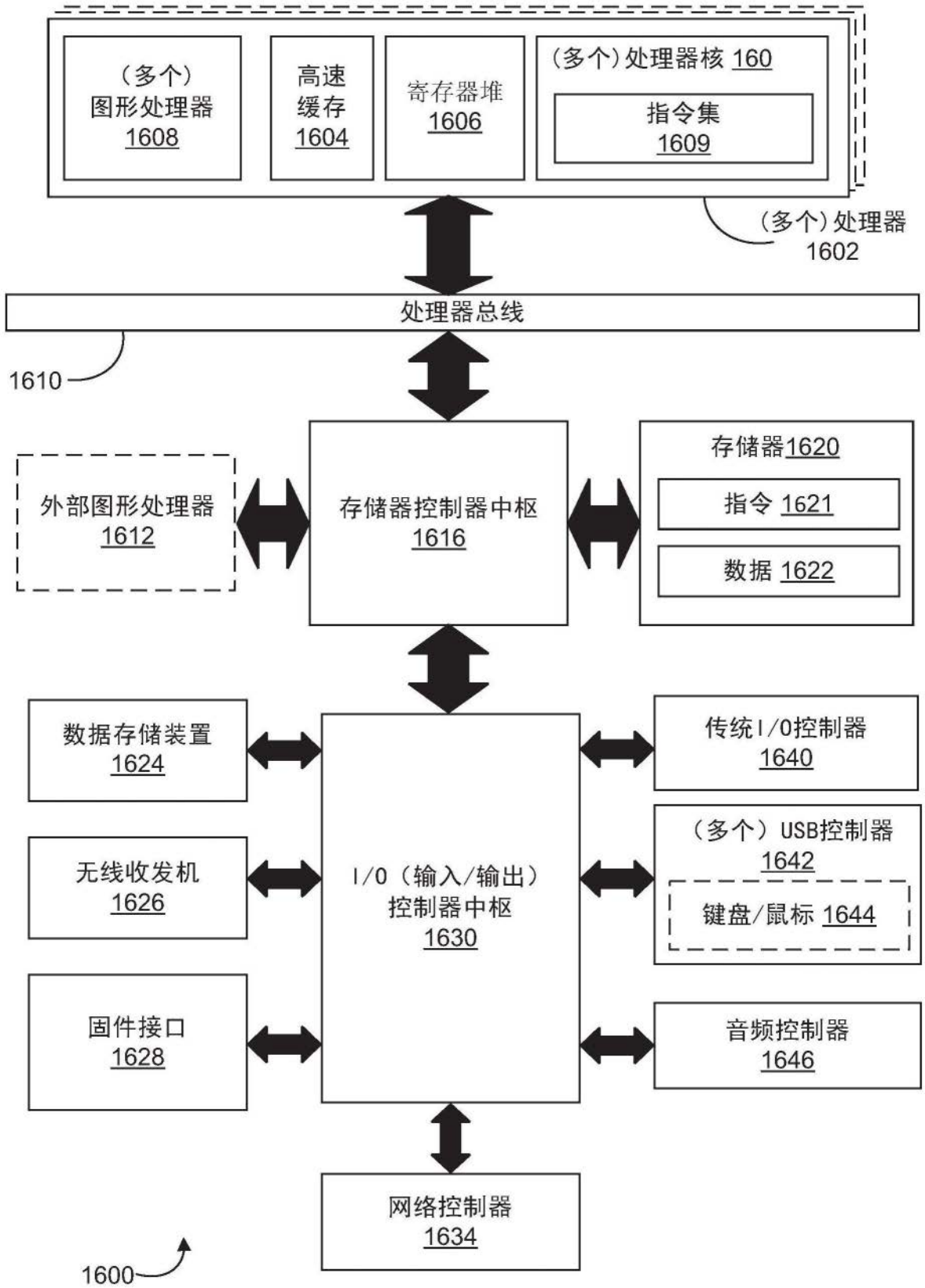

（多个）
图形处理器
1608
高速
缓存
1604
寄存器堆
1606
（多个）处理器核 160
指令集
1609
（多个）处理器
1602
处理器总线
1610
外部图形处理器
1612
存储器控制器中枢
1616
存储器1620
指令 1621
数据 1622
数据存储装置
1624
无线收发机
1626
固件接口
1628
I/O（输入/输出）
控制器中枢
1630
传统I/O控制器
1640
（多个）USB控制器
1642
键盘/鼠标 1644
音频控制器
1646
网络控制器
1634
1600

图16

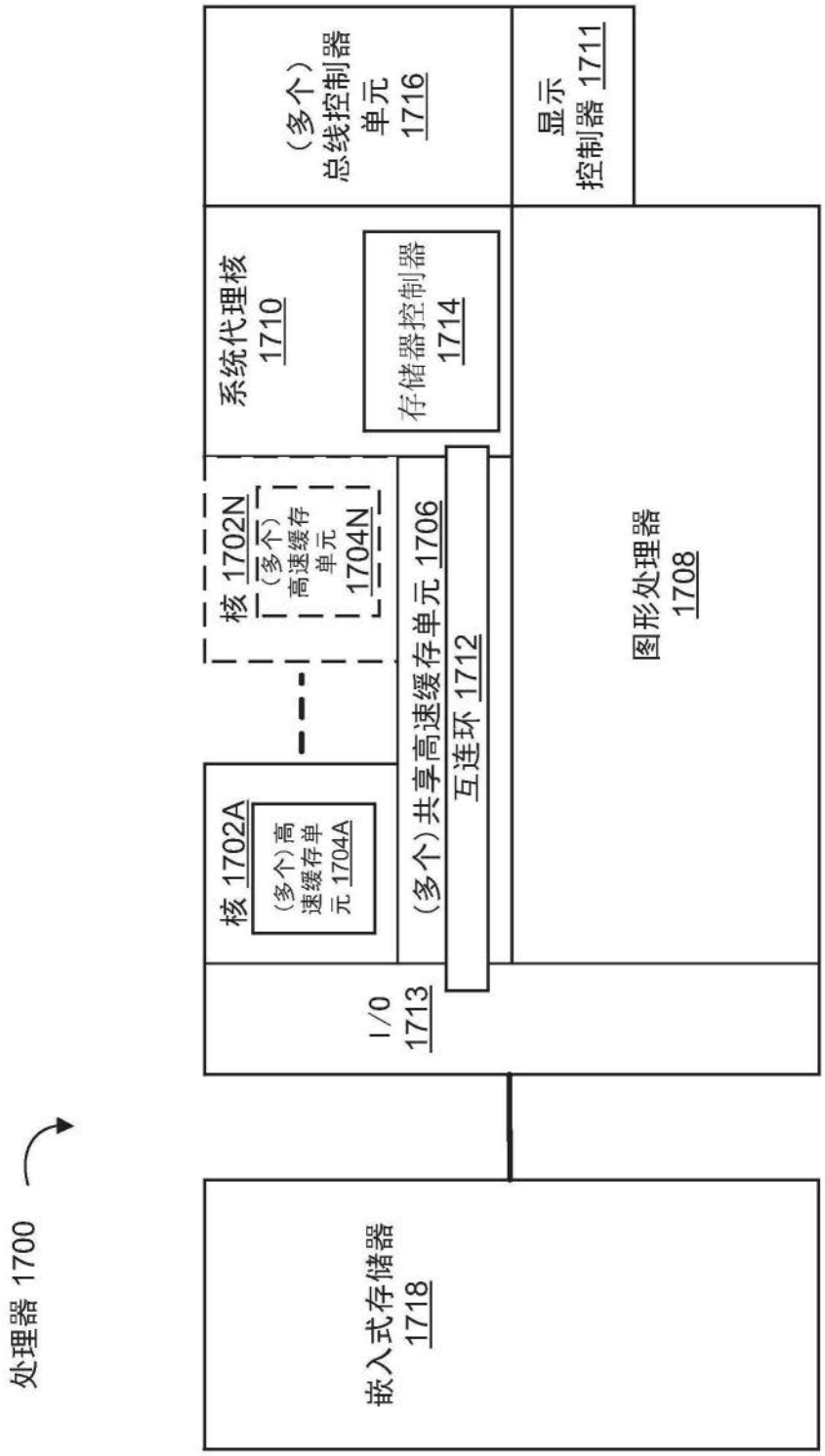

处理器 1700
嵌入式存储器 1718
I/O 1713
核 1702A
(多个)高速缓存单元 1704A
核 1702N
(多个)高速缓存单元 1704N
(多个)共享高速缓存单元 1706
互连环 1712
系统代理核 1710
存储器控制器 1714
(多个)总线控制器单元 1716
显示控制器 1711
图形处理器 1708

图17

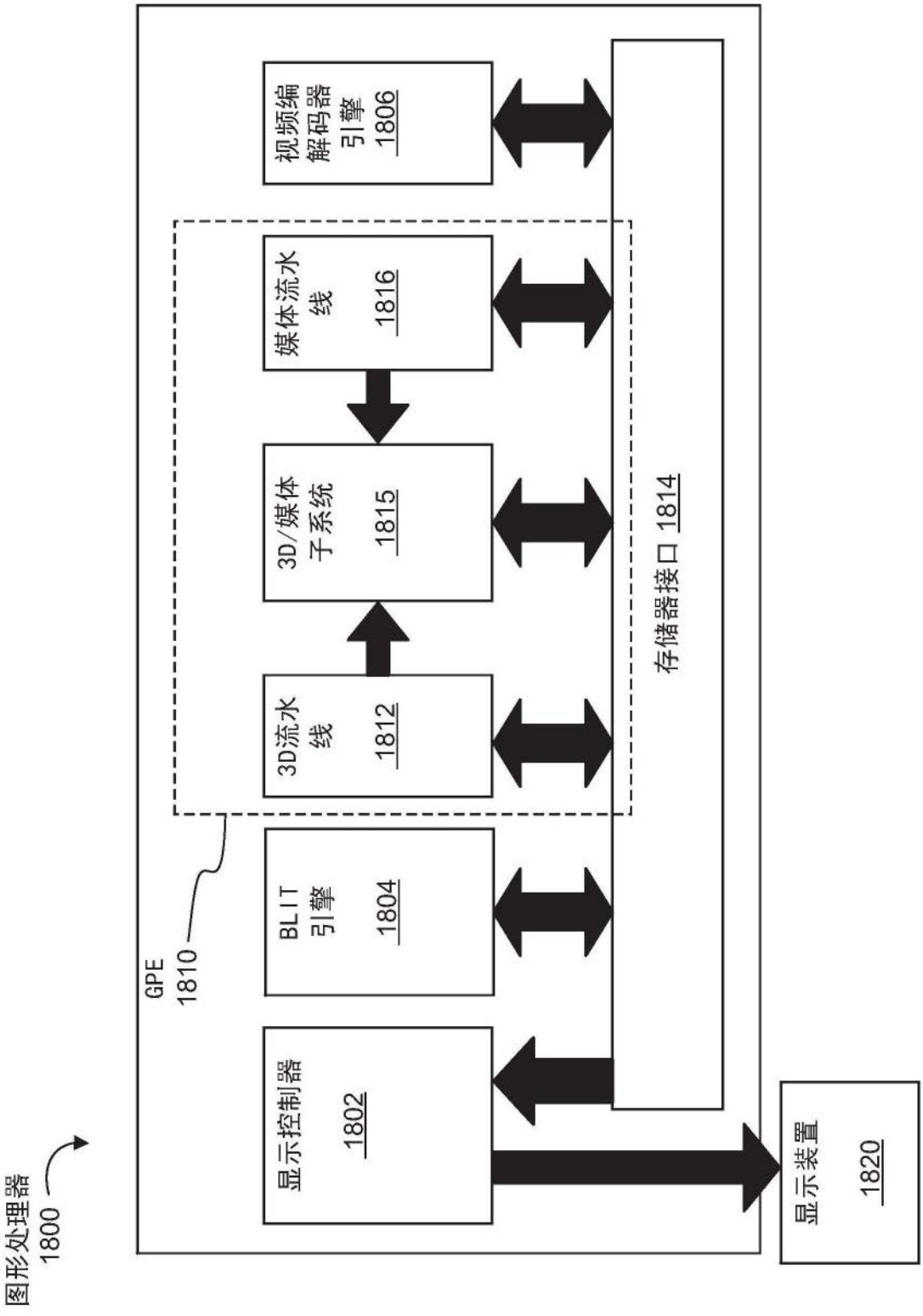

图形处理器
1800
显示控制器
1802
BLIT
引擎
1804
GPE
1810
3D流水
线
1812
3D/媒体
子系统
1815
媒体流水
线
1816
视频编
解码器
引擎
1806
存储器接口 1814
显示装置
1820

图18

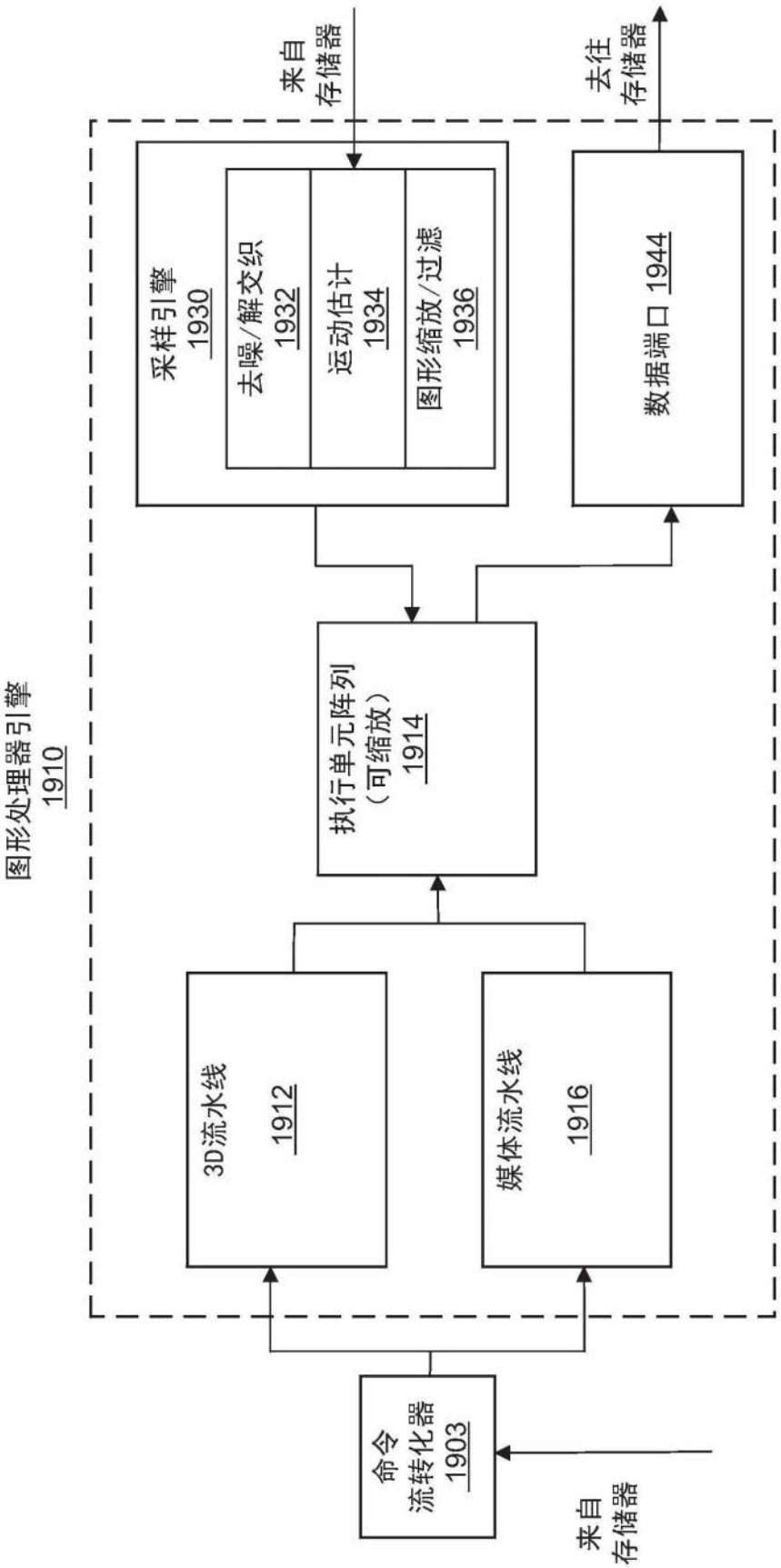

来自
存储器
去往
存储器
采样引擎
1930
去噪/解交织
1932
运动估计
1934
图形缩放/过滤
1936
数据端口 1944
执行单元阵列
（可缩放）
1914
图形处理器引擎
1910
3D流水线
1912
媒体流水线
1916
命令
流转化器
1903
来自
存储器

图19

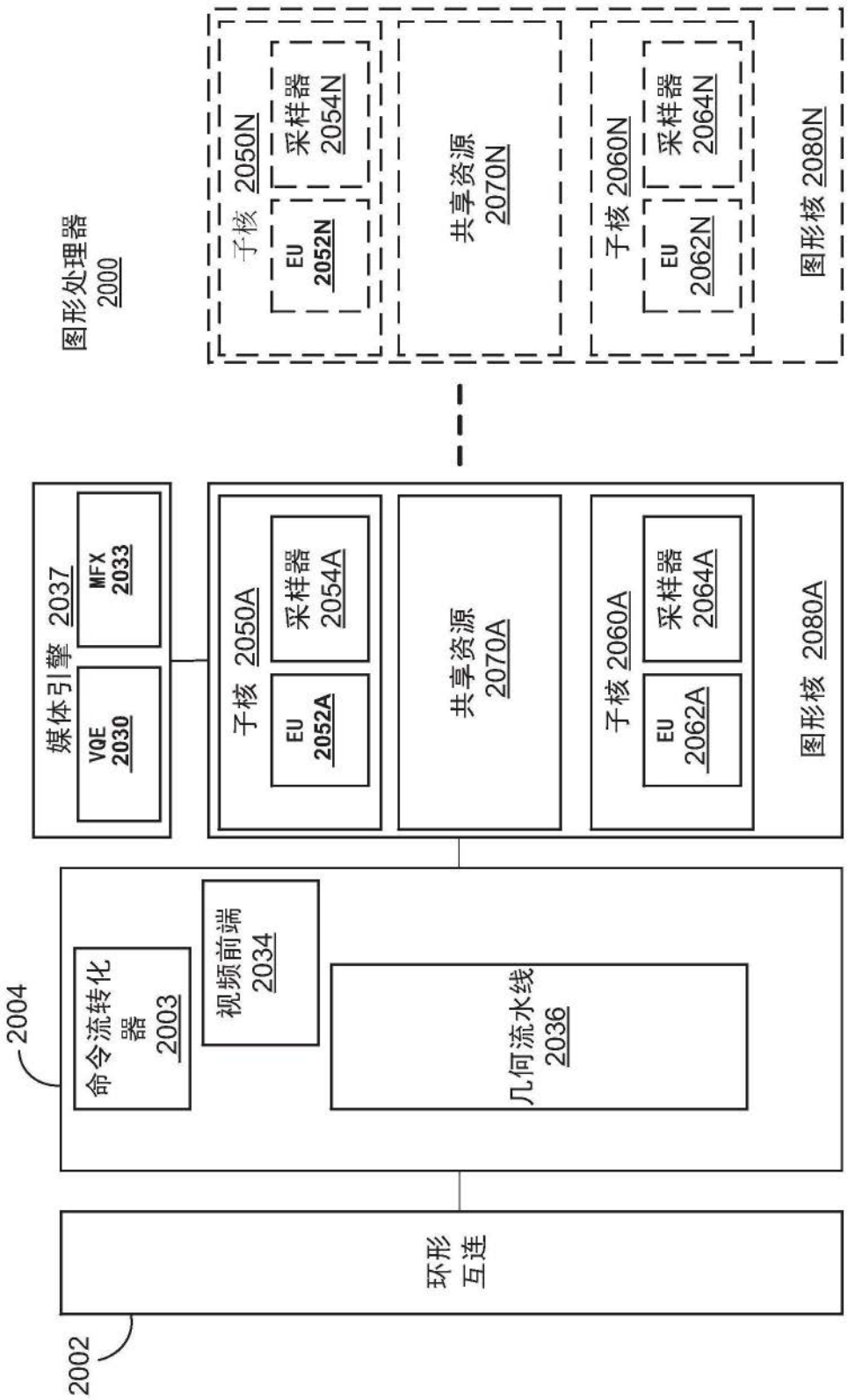

图形处理器 2000
子核 2050N
EU 2052N
采样器 2054N
共享资源 2070N
子核 2060N
EU 2062N
采样器 2064N
图形核 2080N
媒体引擎 2037
VQE 2030
MFX 2033
子核 2050A
EU 2052A
采样器 2054A
共享资源 2070A
子核 2060A
EU 2062A
采样器 2064A
图形核 2080A
2004
命令流转化器 2003
视频前端 2034
几何流水线 2036
环形互连
2002

图20

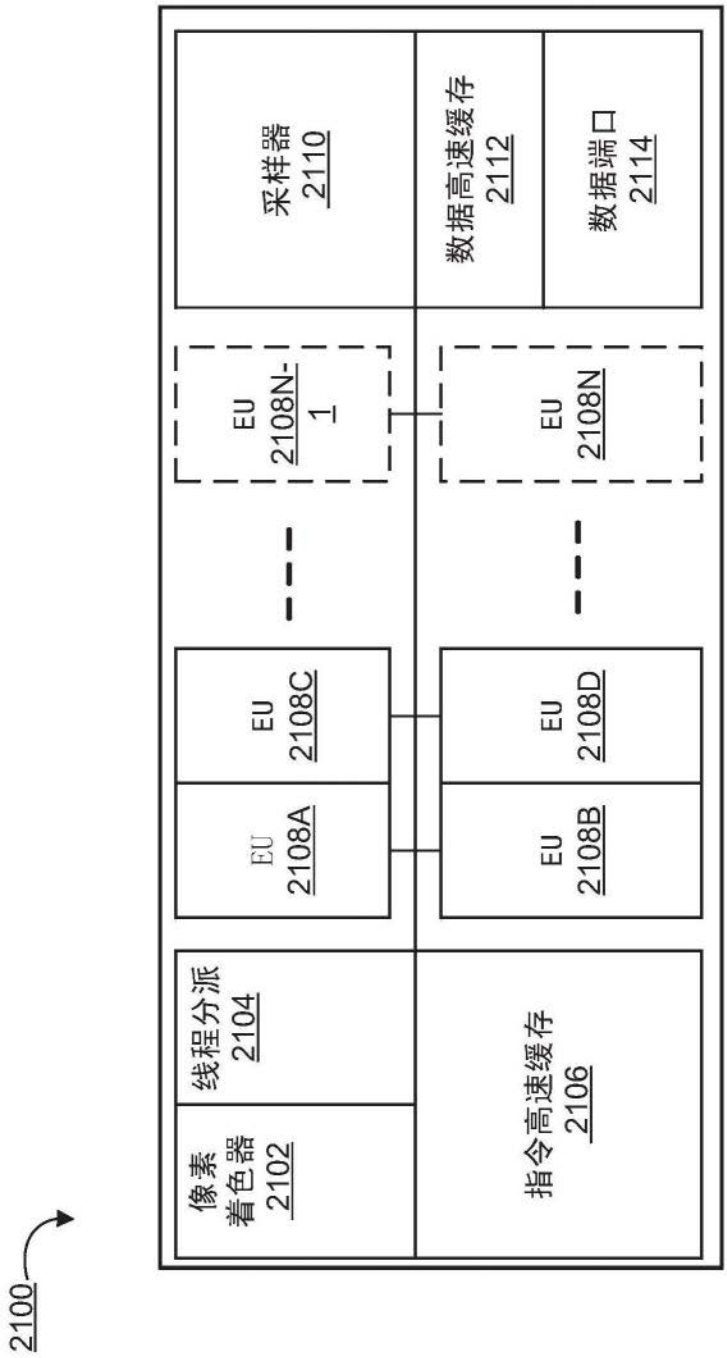

2100
像素着色器 2102
线程分派 2104
指令高速缓存 2106
EU 2108A
EU 2108C
EU 2108B
EU 2108D
EU 2108N-1
EU 2108N
采样器 2110
数据高速缓存 2112
数据端口 2114

图21

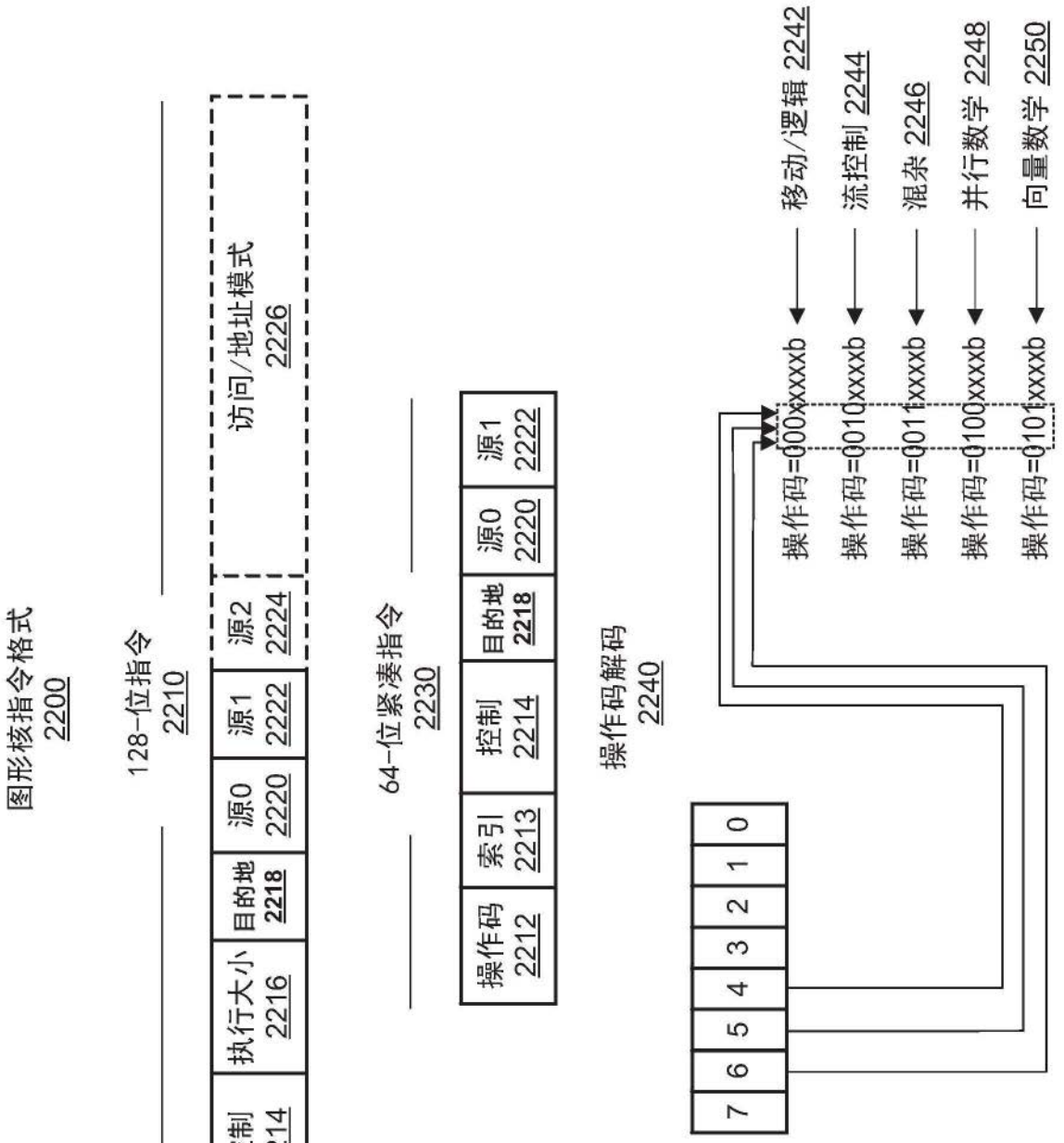

图形核指令格式
2200
128-位指令
2210
操作码
2212
控制
2214
执行大小
2216
目的地
2218
源0
2220
源1
2222
源2
2224
访问/地址模式
2226
64-位紧凑指令
2230
操作码
2212
索引
2213
控制
2214
目的地
2218
源0
2220
源1
2222
操作码解码
2240
7
6
5
4
3
2
1
0
操作码=000xxxxxb
操作码=0010xxxxb
操作码=0011xxxxb
操作码=0100xxxxb
操作码=0101xxxxb
移动/逻辑 2242
流控制 2244
混杂 2246
并行数学 2248
向量数学 2250

图22

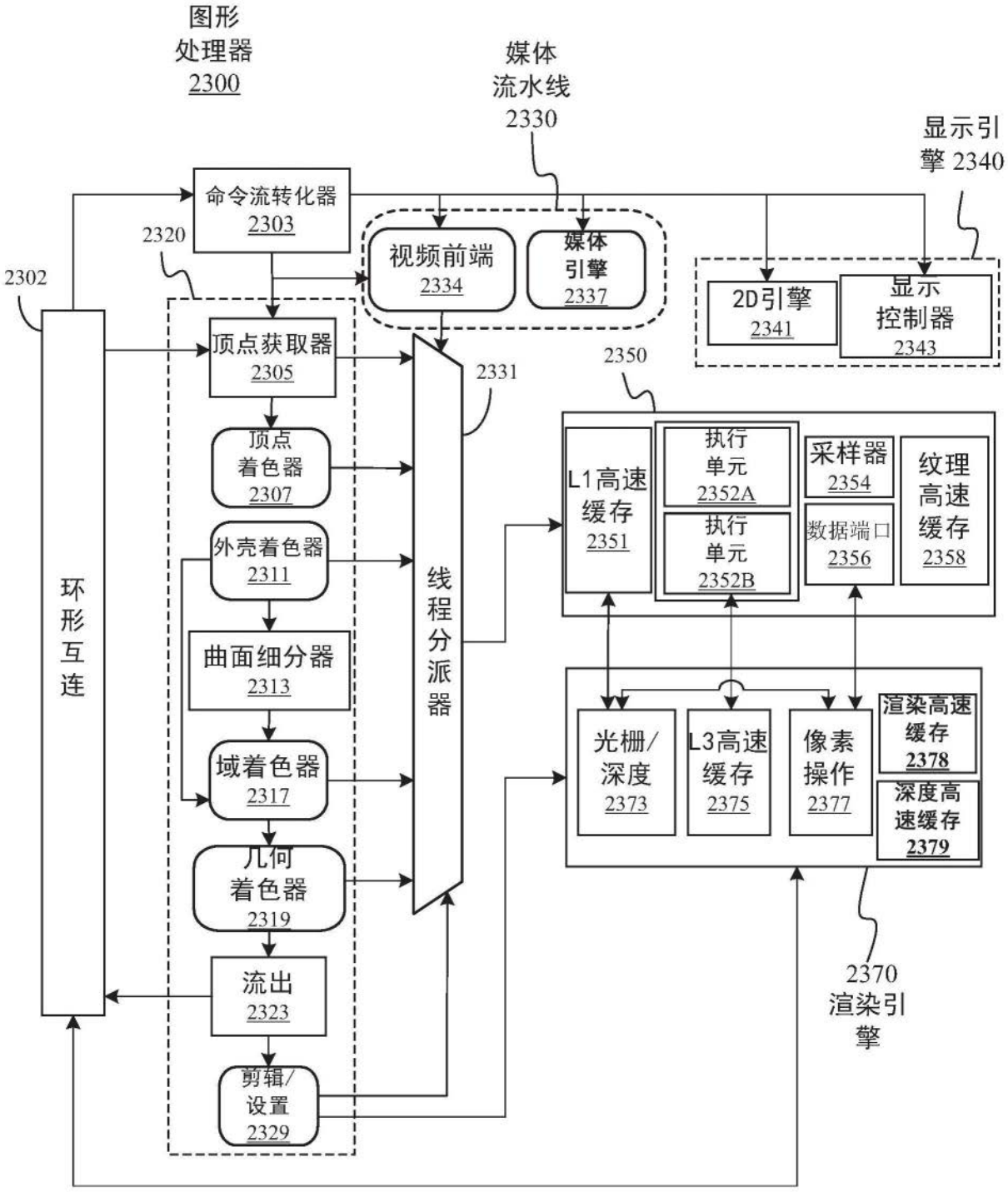

图形
处理器
2300
媒体
流水线
2330
显示引
擎 2340
命令流转化器
2303
2320
2302
视频前端
2334
媒体
引擎
2337
2D引擎
2341
显示
控制器
2343
顶点获取器
2305
2331
2350
顶点
着色器
2307
外壳着色器
2311
曲面细分器
2313
域着色器
2317
几何
着色器
2319
流出
2323
剪辑/
设置
2329
环
形
互
连
线
程
分
派
器
L1高速
缓存
2351
执行
单元
2352A
执行
单元
2352B
采样器
2354
数据端口
2356
纹理
高速
缓存
2358
光栅/
深度
2373
L3高速
缓存
2375
像素
操作
2377
渲染高速
缓存
2378
深度高
速缓存
2379
2370
渲染引
擎

图23

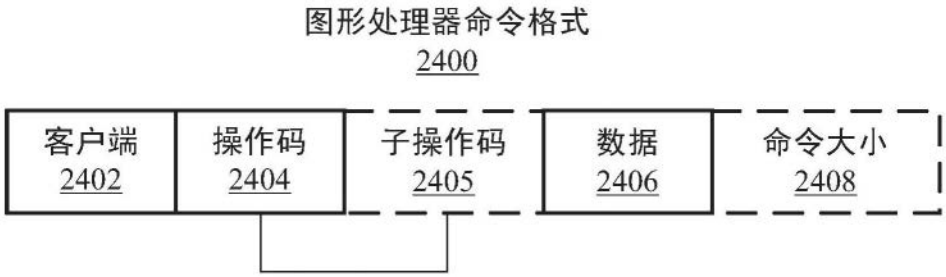

图形处理器命令格式
2400
客户端
2402
操作码
2404
子操作码
2405
数据
2406
命令大小
2408

图24A

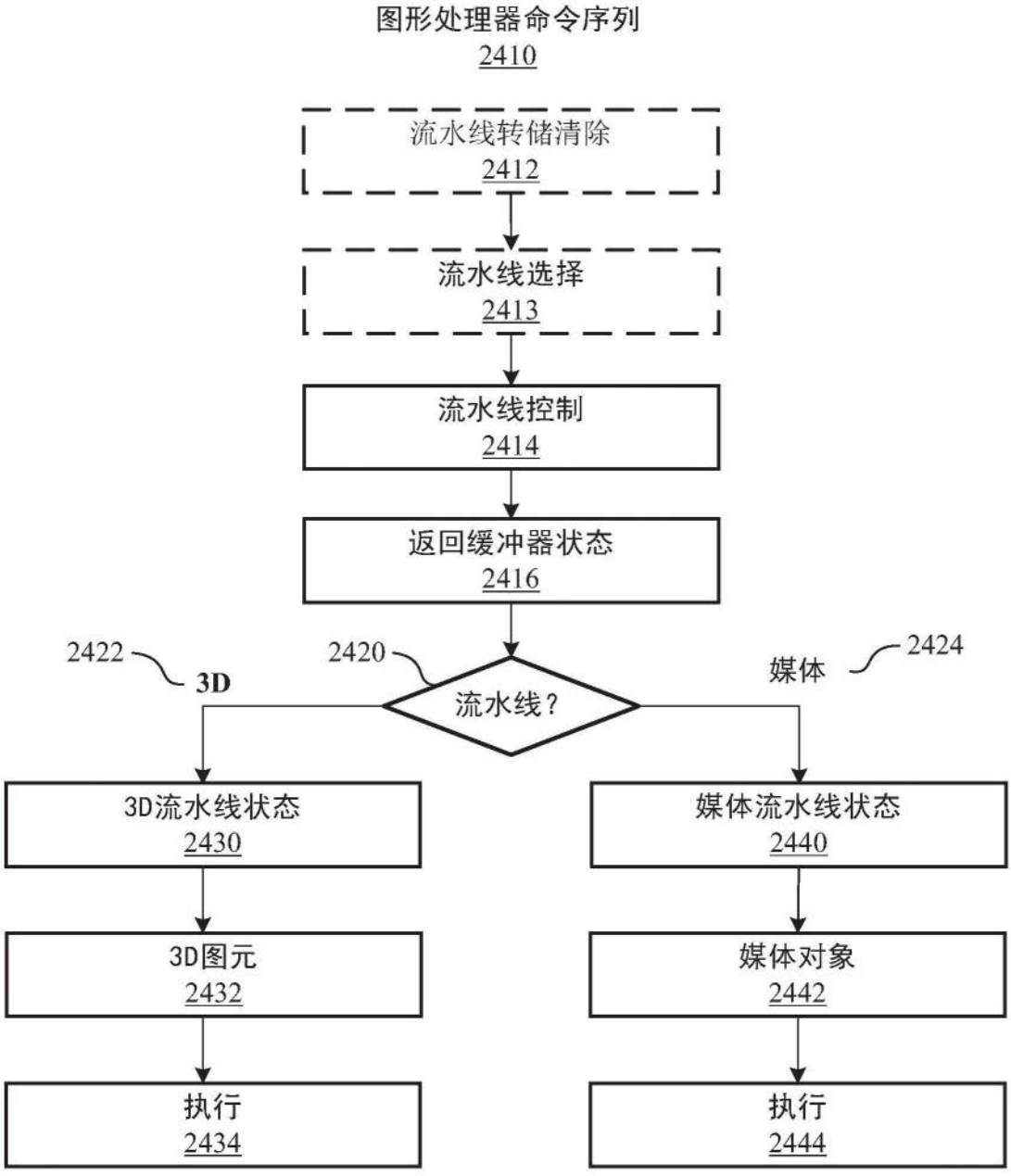

图形处理器命令序列
2410
流水线转储清除
2412
流水线选择
2413
流水线控制
2414
返回缓冲器状态
2416
2422
3D
2420
流水线？
媒体
2424
3D流水线状态
2430
3D图元
2432
执行
2434
媒体流水线状态
2440
媒体对象
2442
执行
2444

图24B

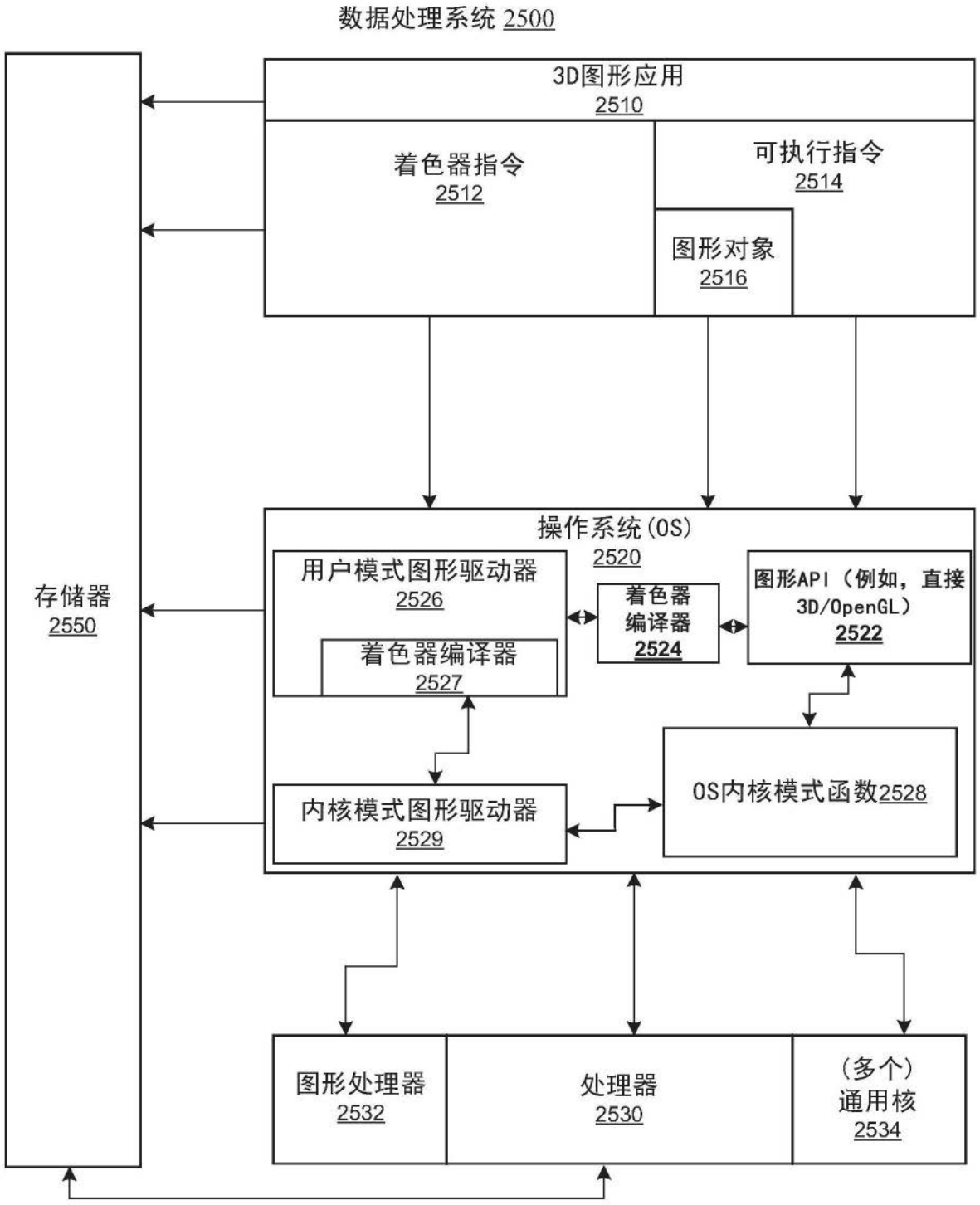

数据处理系统 2500
3D图形应用
2510
着色器指令
2512
可执行指令
2514
图形对象
2516
操作系统(OS)
2520
用户模式图形驱动器
2526
着色器编译器
2527
着色器
编译器
2524
图形API（例如，直接
3D/OpenGL）
2522
存储器
2550
内核模式图形驱动器
2529
OS内核模式函数2528
图形处理器
2532
处理器
2530
(多个)
通用核
2534

图25

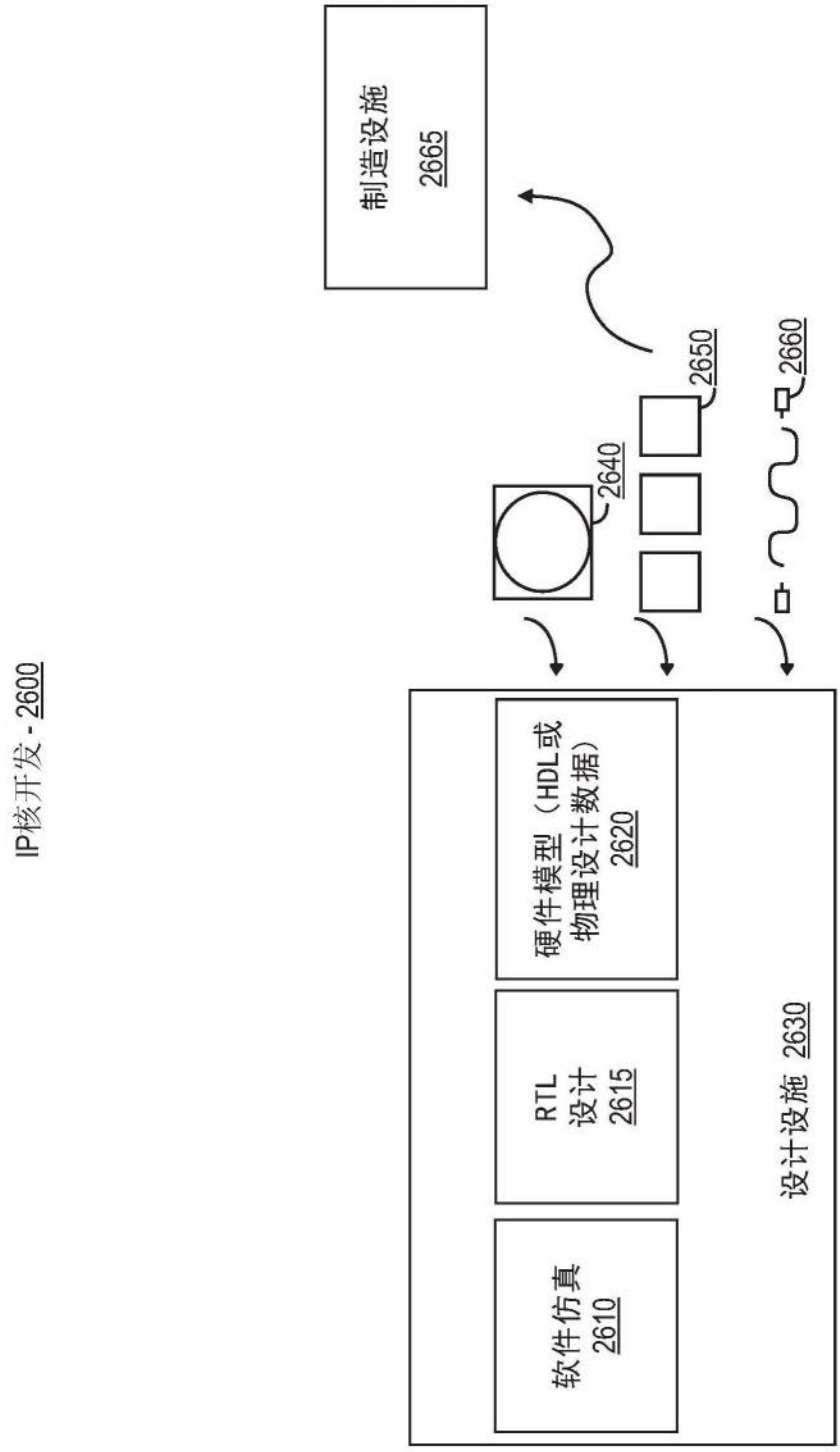

IP核开发 - 2600
软件仿真 2610
RTL 设计 2615
硬件模型（HDL或物理设计数据） 2620
设计设施 2630
2640
2650
2660
制造设施 2665

图26

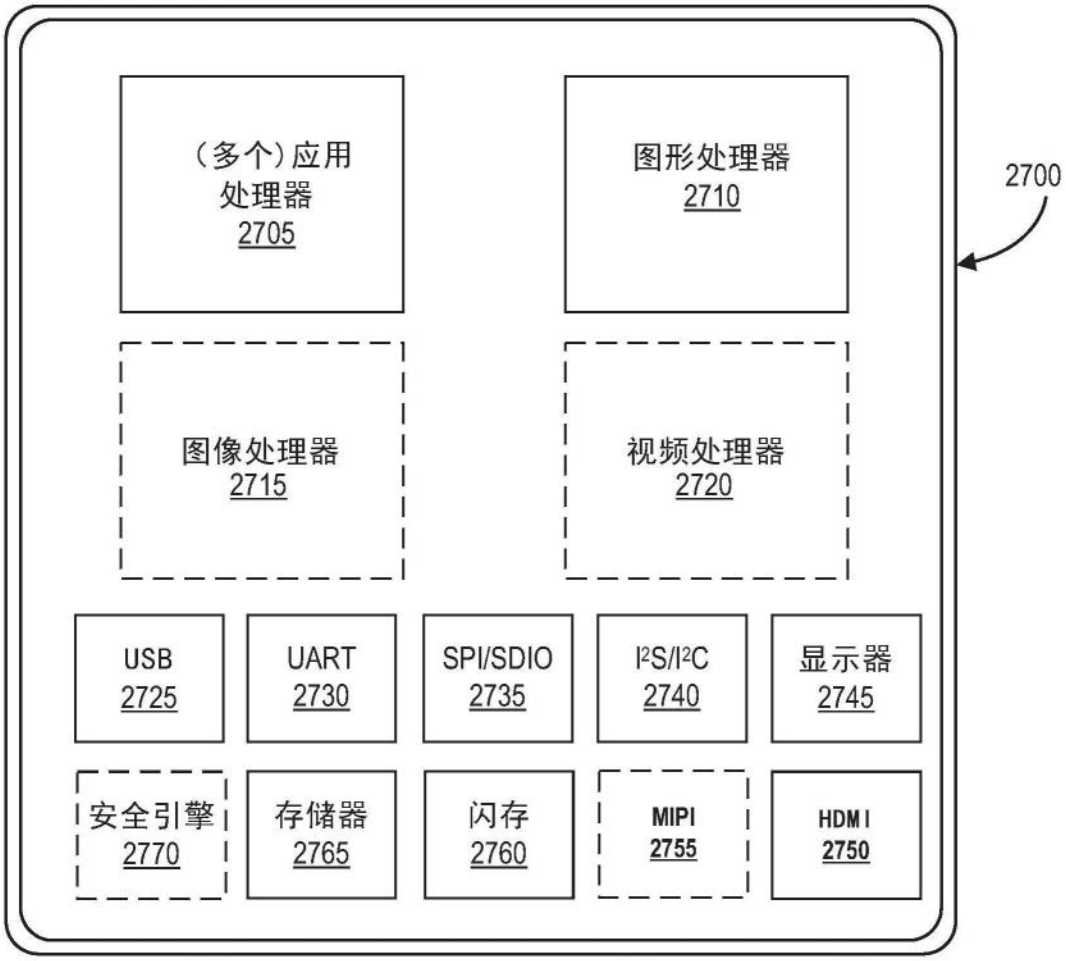

(多个)应用
处理器
2705
图形处理器
2710
2700
图像处理器
2715
视频处理器
2720
USB
2725
UART
2730
SPI/SDIO
2735
I²S/I²C
2740
显示器
2745
安全引擎
2770
存储器
2765
闪存
2760
MIPI
2755
HDMI
2750

图27