



(12) 发明专利

(10) 授权公告号 CN 114402285 B

(45) 授权公告日 2025. 03. 18

(21) 申请号 202080065063.9

(22) 申请日 2020.09.28

(65) 同一申请的已公布的文献号
申请公布号 CN 114402285 A

(43) 申请公布日 2022.04.26

(30) 优先权数据
16/591,306 2019.10.02 US

(85) PCT国际申请进入国家阶段日
2022.03.14

(86) PCT国际申请的申请数据
PCT/US2020/053122 2020.09.28

(87) PCT国际申请的公布数据
W02021/067198 EN 2021.04.08

(73) 专利权人 辉达公司
地址 美国加利福尼亚州

(72) 发明人 A·克尔 M·墨菲 M·哈戈格
J·德穆思 J·德兰

(74) 专利代理机构 北京市磐华律师事务所
11336

专利代理师 高伟

(51) Int.Cl.
G06F 8/41 (2018.01)
G06F 9/455 (2006.01)
G06N 20/10 (2019.01)

(56) 对比文件
CN 104011679 A, 2014.08.27
US 2018293490 A1, 2018.10.11
US 2018322390 A1, 2018.11.08
Jingyue Wu.gpucc: An Open-Source
GPGPU Compiler.《2016 IEEE/ACM
International Symposium on Code
Generation and Optimization (CGO)》.2016,
说明书正文第1节,第2.1-2.4节,第5节.

审查员 曹春燕

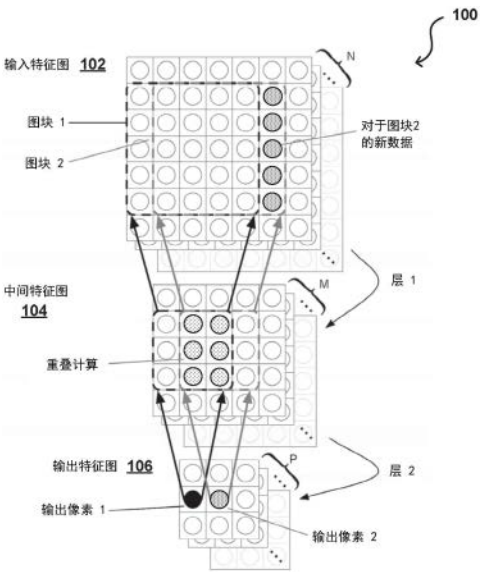
权利要求书2页 说明书59页 附图42页

(54) 发明名称

机器学习的内核融合

(57) 摘要

介绍了用于编译代码的装置、系统和技术。在至少一个实施例中,一个或多个编译器用于利用一个或多个代码部分的一个或多个中间表示来编译一个或多个已编译代码部分。



1. 一种处理器, 包括:

一个或多个电路, 用于执行编译器来编译一个或多个源代码软件程序和一个或多个目标代码软件程序, 以生成单个目标代码软件程序。

2. 如权利要求1所述的处理器, 其中所述一个或多个源代码软件程序包括: 中间表示, 所述中间表示包括一个或多个卷积内核或矩阵乘法的一个或多个单独编译的表示。

3. 如权利要求1所述的处理器, 其中所述一个或多个源代码软件程序包括: 一个或多个中间表示, 所述一个或多个中间表示与一个或多个应用程序特定函数对象相对应。

4. 如权利要求3所述的处理器, 其中所述应用程序特定函数对象是通过应用程序编程接口 (API) 从应用程序接收的。

5. 如权利要求3所述的处理器, 其中与所述一个或多个应用程序特定函数对象相对应的至少一个中间表示被插入一个或多个回调位置。

6. 如权利要求1所述的处理器, 其中所述一个或多个电路进一步用于编译成位码以在一个或多个库中分发。

7. 如权利要求1所述的处理器, 其中所述一个或多个电路进一步用于在使用一个或多个即时代码生成过程进行编译之前执行一个或多个优化。

8. 一种系统, 包括:

一个或多个处理器, 用于执行编译器来编译一个或多个源代码软件程序和一个或多个目标代码软件程序, 以生成单个目标代码软件程序; 以及
存储器, 用于存储已编译代码软件程序。

9. 如权利要求8所述的系统, 其中所述一个或多个源代码软件程序包括: 中间表示, 所述中间表示包括一个或多个卷积内核或矩阵乘法的一个或多个单独编译的表示。

10. 如权利要求8所述的系统, 其中所述一个或多个源代码软件程序包括: 一个或多个中间表示, 所述一个或多个中间表示与一个或多个应用程序特定函数对象相对应。

11. 如权利要求10所述的系统, 其中所述应用程序特定函数对象是通过应用程序编程接口 (API) 从应用程序接收的。

12. 如权利要求10所述的系统, 其中与所述一个或多个应用程序特定函数对象相对应的至少一个中间表示被插入一个或多个回调位置。

13. 如权利要求8所述的系统, 其中所述一个或多个电路进一步用于编译成位码以在一个或多个库中分发。

14. 如权利要求8所述的系统, 其中所述一个或多个处理器进一步用于在使用一个或多个即时代码生成过程进行编译之前执行一个或多个优化。

15. 一种非暂时性机器可读介质, 其上存储有一组指令, 如果由一个或多个处理器执行所述一组指令, 则使所述一个或多个处理器至少:

执行编译器来编译一个或多个源代码软件程序和一个或多个目标代码软件程序, 以生成单个目标代码软件程序。

16. 如权利要求15所述的非暂时性机器可读介质, 其中所述一个或多个源代码软件

程序包括:中间表示,所述中间表示包括一个或更多个卷积内核或矩阵乘法的一个或更多个单独编译的表示。

17.如权利要求15所述的非暂时性机器可读介质,其中所述一个或更多个源代码软件程序包括:一个或更多个中间表示,所述一个或更多个中间表示与一个或更多个应用程序特定函数对象相对应。

18.如权利要求17所述的非暂时性机器可读介质,其中所述应用程序特定函数对象是通过应用程序编程接口(API)从应用程序接收的。

19.如权利要求17所述的非暂时性机器可读介质,其中与所述一个或更多个应用程序特定函数对象相对应的至少一个中间表示被插入一个或更多个回调位置。

20.如权利要求15所述的非暂时性机器可读介质,其中所述一个或更多个处理器进一步用于编译成位码以在一个或更多个库中分发。

21.如权利要求15所述的非暂时性机器可读介质,其中所述一个或更多个处理器进一步用于在使用一个或更多个即时代码生成过程进行编译之前执行一个或更多个优化。

22.一种方法,包括:

执行编译器来编译一个或更多个源代码软件程序和一个或更多个目标代码软件程序,以生成单个目标代码软件程序。

23.如权利要求22所述的方法,其中所述一个或更多个源代码软件程序包括:中间表示,所述中间表示包括一个或更多个卷积内核或矩阵乘法的一个或更多个单独编译的表示。

24.如权利要求22所述的方法,其中所述一个或更多个源代码软件程序包括:一个或更多个中间表示,所述一个或更多个中间表示与一个或更多个应用程序特定函数对象相对应。

25.如权利要求24所述的方法,其中所述应用程序特定函数对象是通过应用程序编程接口(API)从应用程序接收的。

26.如权利要求24所述的方法,其中与所述一个或更多个应用程序特定函数对象相对应的至少一个中间表示被插入一个或更多个回调位置。

27.如权利要求22所述的方法,其中一个或更多个电路进一步用于编译成位码以在一个或更多个库中分发。

28.如权利要求22所述的方法,其中一个或更多个电路进一步用于在使用一个或更多个即时代码生成过程进行编译之前执行一个或更多个优化。

机器学习的内核融合

[0001] 相关申请的交叉引用

[0002] 这是PCT申请并且要求2019年10月2日提交的题为“机器学习的内核融合 (KERNEL FUSION FOR MACHINE LEARNING)”的美国申请第16/591,306号的优先权；该申请的全部公开内容出于所有目的通过引用并入本文。

技术领域

[0003] 至少一个实施例涉及用于执行和促进人工智能的处理资源。例如，至少一个实施例涉及用于根据本文描述的各种新颖技术来训练神经网络的处理器或计算系统。

背景技术

[0004] 以特定处理器架构为目标的应用程序可能难以最大化特定设备的性能。硬件提供者可以为不同的设备类型提供专门的计算实现，但可能不希望公开与这些专门实现相关的函数。

附图说明

[0005] 将参考附图描述根据本公开的各个实施例，其中：

[0006] 图1示出了根据至少一个实施例的图块表示；

[0007] 图2示出了根据至少一个实施例的计算架构；

[0008] 图3示出了根据至少一个实施例的编译器管线；

[0009] 图4示出了根据至少一个实施例的编译器架构；

[0010] 图5示出了根据至少一个实施例的用于编译融合功能的过程；

[0011] 图6A示出了根据至少一个实施例的推理和/或训练逻辑；

[0012] 图6B示出了根据至少一个实施例的推理和/或训练逻辑；

[0013] 图7示出了根据至少一个实施例的示例数据中心系统；

[0014] 图8示出了根据至少一个实施例的计算机系统；

[0015] 图9示出了根据至少一个实施例的计算机系统；

[0016] 图10示出了根据至少一个实施例的计算机系统；

[0017] 图11示出了根据至少一个实施例的计算机系统；

[0018] 图12A示出了根据至少一个实施例的计算机系统；

[0019] 图12B示出了根据至少一个实施例的计算机系统；

[0020] 图12C示出了根据至少一个实施例的计算机系统；

[0021] 图12D示出了根据至少一个实施例的计算机系统；

[0022] 图12E和图12F示出了根据至少一个实施例的共享编程模型；

[0023] 图13示出了根据至少一个实施例的示例性集成电路和相关联的图形处理器；

[0024] 图14A-14B示出了根据至少一个实施例的示例性集成电路和相关联的图形处理器；

- [0025] 图15A-15B示出了根据至少一个实施例的附加的示例性图形处理器逻辑；
- [0026] 图16示出了根据至少一个实施例的计算机系统；
- [0027] 图17A示出了根据至少一个实施例的并行处理器；
- [0028] 图17B示出了根据至少一个实施例的分区单元；
- [0029] 图17C示出了根据至少一个实施例的处理集群；
- [0030] 图17D示出了根据至少一个实施例的图形多处理器；
- [0031] 图18示出了根据至少一个实施例的多图形处理单元 (GPU) 系统；
- [0032] 图19示出了根据至少一个实施例的图形处理器；
- [0033] 图20示出了根据至少一个实施例的处理器器的微架构；
- [0034] 图21示出了根据至少一个实施例的深度学习应用处理器；
- [0035] 图22示出了根据至少一个实施例的示例神经形态处理器；
- [0036] 图23和图24示出了根据至少一个实施例的图形处理器器的至少部分；
- [0037] 图25示出了根据至少一个实施例的图形处理器器核心的至少部分；
- [0038] 图26A-26B示出了根据至少一个实施例的图形处理器器核心的至少部分；
- [0039] 图27示出了根据至少一个实施例的并行处理单元 (“PPU”)；
- [0040] 图28示出了根据至少一个实施例的通用处理集群 (“GPC”)；
- [0041] 图29示出了根据至少一个实施例的并行处理单元 (“PPU”) 的存储器分区单元；以及
- [0042] 图30示出了根据至少一个实施例的流式多处理器。

具体实施方式

[0043] 在至少一个实施例中,可以对计算密集型内核、卷积或矩阵执行单独编译。在至少一个实施例中,单独编译可以涉及处理内核,例如用于即时内核融合的图形处理单元 (GPU) 的内核。在至少一个实施例中,活动组件可用于融合单独编译并以低级可移植表示分布的 GPU 内核。

[0044] 在至少一个实施例中,程序员可以已经用高级编程语言实现了一个或更多个计算,例如来自 NVIDIA 公司的 C++ 或 CUDA C++。在至少一个实施例中,这些计算可以在矩阵乘积或卷积内核的关键阶段具有明确配置的回调。在至少一个实施例中,这样的回调可以具有明确定义的函数签名和语义。在至少一个实施例中,可以使用编译堆栈的各个阶段来编译该高级表示以编译成中间表示,然后该中间表示可以作为位码分布在库中。在至少一个实施例中,编译可以从诸如 CUDA 之类的表示进行到 NVVM 中间表示 (IR), 得到的 NVVM 位码分布在库中。在至少一个实施例中,调用站点可以保持明确表示,例如通过使用 NVVM 函数调用句法。

[0045] 在至少一个实施例中,在应用程序运行时,活动组件可以公开应用程序编程接口 (API), 其中客户端应用可以为一些或所有回调提供函数对象。

[0046] 在至少一个实施例中,这可以包括一个或更多个 CUDA 设备函数对象。在至少一个实施例中,这些源表示可以由具有实时编译器的活动组件编译,产生中间表示,例如 NVVMIR, 或者通过单独编译或程序生成的中间表示。在至少一个实施例中,活动组件可以执行编译器内联传递以将这些函数插入到包含内核内的适当调用站点处。在至少一个实施例

中,一旦内联,可以执行过程间优化,然后是即时代码生成过程。在至少一个实施例中,这种优化和代码生成可以产生完整的GPU二进制。在至少一个实施例中,内联和优化可以在寄存器分配和最终指令调度之前发生,使得由活动组件融合的内核具有零额外开销。

[0047] 在至少一个实施例中,在计算内核和应用程序定义的一个或多个运算符已经被活动组件内联和优化之后,可以将得到的低级中间表示可以是高速缓存或编译为本地机器代码,并且然后高速缓存。在至少一个实施例中,这种机制可以促进两阶段部署策略,其中首先融合运算符并调整性能。在至少一个实施例中,不变式表示被部署用于在第二阶段中执行。在至少一个实施例中,这种能力可以与深度学习软件库一起被利用,以避免长时间的开销,并且可以在融合内核需要安全认证的安全关键环境中使用。

[0048] 在至少一个实施例中,部分地执行内核融合以获得针对各种加速器处理器架构的应用程序的改进性能。在至少一个实施例中,可以使用单独的编译来避免在为特定类型的设备生成的实现中定义专用挂钩后面的函数。在至少一个实施例中,可以通过将运算与计算受限的工作负载(诸如矩阵乘法或卷积内核)相结合来获得性能优势,这可以帮助以应用程序特定的方式扩展函数或通过利用局部性来提高性能。在至少一个实施例中,如果矩阵乘法计算结果,则对应的数据驻留在芯片上,并且另一运算可以使用该数据而不必从设备存储器中提取该数据。

[0049] 在至少一个实施例中,矩阵乘法卷积内核可以被部分编译,然后与用户或应用程序提供的函数的较低级别表示融合。在至少一个实施例中,这随后可以用专门的编译器工具链来编译。在至少一个实施例中,用户或应用程序将不会接收到该矩阵乘法的高级形式,或直接访问专用编译器。在至少一个实施例中,用户将仍然能够将他们的运算与供应商提供的实现融合并实现最佳性能。

[0050] 在至少一个实施例中,计算受限的工作负载的实现可以假设输入将被划分为图块,并且以整个图块为单位被提取或流式传输到芯片或设备中。在至少一个实施例中,可以利用图块的表示,如图1的示例地图视图100中所示。在至少一个实施例中,可以在框架之间使用抽象层来生成运行时内核。在至少一个实施例中,这包括一组输入特征图102,其可用于构建特征图的其他层,例如一层中间特征图104和输出特征图106。在至少一个实施例中,每个地图可以由一组图块组成,重叠的图块部分对应于重叠的计算。在至少一个实施例中,对于输出特征图,每个图块可以对应于输出像素值。在至少一个实施例中,可以表示张量运算的图并且可以应用编译器算法进行优化。在至少一个实施例中,可以构造内核以及专门的通用矩阵乘法(GEMM)和卷积变体。在至少一个实施例中,对于内核融合,逐元素运算可以与GEMMS、卷积和归约融合,并且连续的卷积运算也可以融合。在至少一个实施例中,可以在框架和代码生成器之间提供丰富的接口。在至少一个实施例中,数据流可以具有几个点,在这些点组合其他运算是可行的,而不影响执行相应数学计算的内部循环。

[0051] 在至少一个实施例中,可以使用计算编译架构202,如图2的视图200所示。在至少一个实施例中,该架构可以包括计算前端206,其可以将输入传递给编译器优化模块208,在至少一些实施例中,编译器优化模块208可以提供优化和分析通道。在至少一个实施例中,优化可以利用来自一个或多个设备库204的信息用于目标类型的设备。在至少一个实施例中,可以应用210一种或更多种计算降低和优化,以提供诸如矢量化之类的函数。在至少一个实施例中,该架构可以利用DCI降低212以及一个或多个库,如可以包括引擎库214和

ELF库216。

[0052] 在至少一个实施例中,可以使用统一的中间表示,例如可以包括统一的NVVMIR。在至少一个实施例中,这种统一的中间表示可以利用对ORI的增强,例如寄存器分配、优化指示和指令调度。在至少一个实施例中,这种方法还可以提供改进的编译器内在函数,其能够提供对新类型指令的支持。在至少一个实施例中,可以使用可以提供可重用抽象作为C++模板的深度学习的模板库。在至少一个实施例中,模板参数可以包括图块大小、输入类型、累积类型和数学运算。在至少一个实施例中,正交分量可以在可能的情况下解耦。在至少一个实施例中,优化指令和特殊函数可以表示为内在函数和代码生成器通道。在至少一个实施例中,二进制的部署可以通过静态库进行,API到编译的模板实例中,或者将源代码部署为仅标头库。

[0053] 在至少一个实施例中,可以使用编译器管线300,如图3所示。在至少一个实施例中,混合模式输入文件302被提供给计算平台前端304。在至少一个实施例中,前端304可以传递主机IR 306和设备IR 308,其中设备IR 308可以是使用设备代码生成器310处理。在至少一个实施例中,设备代码生成器包括IR优化器312和代码生成器314。在至少一个实施例中,设备代码生成器310产生PTX、ELF和LLVM-IR,它们是打包成胖二进制316。在至少一个实施例中,胖二进制316连同来自主机代码生成器318的代码一起被提供给主机编译器320,主机编译器320可以将代码编译成单个输出对象。

[0054] 在至少一个实施例中,可以使用可以为函数、图块和指令强制构造IR对象的过程代码生成方法。在至少一个实施例中,这可以在图块大小、数据类型和数学运算上参数化。在至少一个实施例中,部署策略可以是嵌入过程代码生成器、IR代码高速缓存、二进制和JIT,并提供编译管理器组件以调用代码生成。在至少一个实施例中,这种方法可以是灵活的,因为可以显式地构造任何IR码序列。在至少一个实施例中,编译时间是有利的,因为没有代码解析或模板专门化。在至少一个实施例中,代码不向用户公开,因为用户只接收构建程序的二进制。

[0055] 在至少一个实施例中,用户可能尝试将运算与矩阵乘法相结合。在至少一个实施例中,矩阵乘法具有非常规则的结构化数据访问模式。在至少一个实施例中,硬件供应商可以在代码中留下挂钩,这些挂钩是注入和编译自定义函数的机会。在至少一个实施例中,编译器工具链可以将该工作负载编译成中间表示,例如LVM-IR。在至少一个实施例中,这些挂钩或融合站点可以作为设备函数调用保留在该中间表示中。在至少一个实施例中,应用程序程序员可以至少部分地基于这些挂钩中的每一个可能的语义来定义函数,并且该编译器可以将它们的函数编译为中间表示,将其内嵌在调用站点,在计算受限的工作负载内,并应用专门的编译器变体来生成最终降低的机器代码。在至少一个实施例中,要包括的运算的类型可以至少部分地取决于相应的应用程序和用例。

[0056] 在至少一个实施例中,可以使用包括混合IR构建的平台400,例如图4中所示。在至少一个实施例中,程序员已经编写了一个带有挂钩的基本卷积的实现,如由将由语言适合的编译器404编译的静态内核402所表示的。在至少一个实施例中,程序代码生成器408或其他应用程序接口可以编译用户提供的函数以与内核融合。在至少一个实施例中,编译管理器416可以协助编译,其可以包括要融合的编译代码406的各种实例。在至少一个实施例中,转换和优化模块412可以如本文所讨论的那样与代码生成器模块414一起用于生成最终编

译的二进制或降低的机器代码以供执行。在至少一个实施例中,这种编译架构可以在低级深度学习框架内运行,例如来自NVIDIA公司的TensorRT,其中大部分函数可以在一个或更多个GPU上执行。

[0057] 在至少一个实施例中,一个或更多个应用程序提供的函数可以与一个或更多个计算受限工作负载的编码实现相融合。在至少一个实施例中,这种融合可以提供相对于单独执行的性能和能量改进。在至少一个实施例中,一种架构可以利用专用编译器行为来公开编译整个应用程序是不可行的。在至少一个实施例中,这样的架构可以使硬件供应商能够将敏感IP或专有函数与用户提供的函数相结合,而无需公开专有函数。在至少一个实施例中,这种融合还可以减少函数的整体代码大小。

[0058] 在至少一个实施例中,可以使用如图5所示的编译过程500。在至少一个实施例中,在应用程序运行时通过API接收502一个或更多个函数对象。在至少一个实施例中,这些函数对象由该应用程序提供或为该应用程序提供。在至少一个实施例中,每个函数对象被编译504以获得中间表示。在至少一个实施例中,获得506计算受限操作的中间或部分编译表示,诸如卷积内核。在至少一个实施例中,执行508内联传递以便插入在对应于部分编译内核中的挂钩的一个或更多个调用位置处的一个或更多个编译函数。在至少一个实施例中,执行一种或更多种优化。在至少一个实施例中,执行512最终编译以生成输出,其可以采用最终可执行代码的形式。

[0059] 推理和训练逻辑

[0060] 图6A示出了用于执行与一个或更多个实施例相关联的推理和/或训练操作的推理和/或训练逻辑615。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。

[0061] 在至少一个实施例中,推理和/或训练逻辑615可以包括但不限于代码和/或数据存储器601,用于存储前向和/或输出权重和/或输入/输出数据,和/或在一个或更多个实施例的方面中配置被训练为和/或用于推理的神经网络的神经元或层的其他参数。在至少一个实施例中,训练逻辑615可以包括或耦合到用于存储图形代码或其他软件以控制时序和/或顺序的代码和/或数据存储器601,其中权重和/或其他参数信息被加载以配置逻辑,包括整数和/或浮点单元(统称为算术逻辑单元(ALU))。在至少一个实施例中,代码(诸如图代码)基于该代码所对应的神经网络的架构将权重或其他参数信息加载到处理器ALU中。在至少一个实施例中,代码和/或数据存储器601存储在使用一个或更多个实施例的方面训练和/或推理期间的输入/输出数据和/或权重参数的前向传播期间结合一个或更多个实施例训练或使用的神经网络的每个层的权重参数和/或输入/输出数据。在至少一个实施例中,代码和/或数据存储器601的任何部分都可以包括在其他片上或片外数据存储内,包括处理器的L1、L2或L3高速缓存或系统存储器。

[0062] 在至少一个实施例中,代码和/或数据存储器601的任何部分可以在一个或更多个处理器或其他硬件逻辑设备或电路的内部或外部。在至少一个实施例中,代码和/或代码和/或数据存储器601可以是高速缓存存储器、动态随机可寻址存储器(“DRAM”)、静态随机可寻址存储器(“SRAM”)、非易失性存储器(例如闪存)或其他存储。在至少一个实施例中,对代码和/或代码和/或数据存储器601是在处理器的内部还是外部的选择,例如,或者由DRAM、SRAM、闪存或某种其他存储类型组成,可以取决于存储片上与片外的可用存储空间,正在执行训练和/或推理功能的延迟要求,在神经网络的推理和/或训练中使用的数据的批大小或这些

因素的某种组合。

[0063] 在至少一个实施例中,推理和/或训练逻辑615可以包括但不限于代码和/或数据存储605,用于存储与在一个或多个实施例的方面中被训练为和/或用于推理的神经网络的神经元或层相对应的反向和/或输出权重和/或输入/输出数据。在至少一个实施例中,在使用一个或多个实施例的方面训练和/或推理期间,代码和/或数据存储605存储在输入/输出数据和/或权重参数的反向传播期间结合一个或多个实施例训练或使用的神经网络的每个层的权重参数和/或输入/输出数据。在至少一个实施例中,训练逻辑615可以包括或耦合到用于存储图代码或其他软件以控制时序和/或顺序的代码和/或数据存储605,其中权重和/或其他参数信息被加载以配置逻辑,该逻辑包括整数和/或浮点单元(统称为算术逻辑单元(ALU))。在至少一个实施例中,代码(诸如图代码)基于该代码所对应的神经网络的架构将权重或其他参数信息加载到处理器ALU中。在至少一个实施例中,代码和/或数据存储605的任何部分可以与其他片上或片外数据存储一起包括,包括处理器的L1、L2或L3高速缓存或系统存储器。在至少一个实施例中,代码和/或数据存储605的任何部分可以在一个或多个处理器或其他硬件逻辑设备或电路上的内部或外部。在至少一个实施例中,代码和/或数据存储605可以是高速缓存存储器、DRAM、SRAM、非易失性存储器(例如闪存)或其他存储。在至少一个实施例中,代码和/或数据存储605是在处理器的内部还是外部的选择,例如,是由DRAM、SRAM、闪存还是其他某种存储类型组成,取决于片上与片外的可用存储,正在执行的训练和/或推理功能的延迟要求,在神经网络的推理和/或训练中使用的数据的批大小或这些因素的某种组合。

[0064] 在至少一个实施例中,代码和/或数据存储601以及代码和/或数据存储605可以是分开的存储结构。在至少一个实施例中,代码和/或数据存储601以及代码和/或数据存储605可以是相同的存储结构。在至少一个实施例中,代码和/或数据存储601以及代码和/或数据存储605可以部分是相同的存储结构而部分是分开的存储结构。在至少一个实施例中,代码和/或数据存储601以及代码和/或数据存储605的任何部分可以与其他片上或片外数据存储包括在一起,包括处理器的L1、L2或L3高速缓存或系统存储器。

[0065] 在至少一个实施例中,推理和/或训练逻辑615可以包括但不限于一个或多个算术逻辑单元(“ALU”)610(包括整数和/或浮点单元),用于至少部分地基于训练和/或推理代码(例如,图代码)或由其指示来执行逻辑和/或数学运算,其结果可能会产生存储在激活存储620中的激活(例如,来自神经网络内部的层或神经元的输出值),其是存储在代码和/或数据存储601和/或代码和/或数据存储605中的输入/输出和/或权重参数数据的函数。在至少一个实施例中,激活响应于执行指令或其他代码,根据由ALU610执行的线性代数和/或基于矩阵的数学生成在激活存储620中存储的激活,其中存储在代码和/或数据存储605中和/或代码和/或数据存储601中的权重值用作操作数以及其他值,例如偏置值、梯度信息、动量值或其他参数或超参数,可以将任何或所有这些存储在代码和/或数据存储605或代码和/或数据存储601或其他片上或片外存储中。

[0066] 在至少一个实施例中,一个或多个处理器或其他硬件逻辑设备或电路包括一个或多个ALU 610,而在另一实施例中,一个或多个ALU 610可以在处理器或其他硬件逻辑设备或使用它们(例如协处理器)的电路外。在至少一个实施例中,可以将一个或多个ALU 610包括在处理器的执行单元之内,或者以其他方式包括在由处理器的执行单元可

访问的ALU库(bank)中,该处理器的执行单元可以在同一处理器内或者分布在不同类型的不同处理器之间(例如,中央处理单元、图形处理单元、固定功能单元等)。在至少一个实施例中,代码和/或数据存储601、代码和/或数据存储605以及激活存储620可以在相同处理器或其他硬件逻辑设备或电路上,而在另一实施例中,它们可以在不同的处理器或其他硬件逻辑设备或电路中,或在相同和不同处理器或其他硬件逻辑设备或电路的某种组合中。在至少一个实施例中,激活存储620的任何部分可以与其他片上或片外数据存储包括在一起,包括处理器的L1、L2或L3高速缓存或系统存储器。此外,推理和/或训练代码可以与处理器或其他硬件逻辑或电路可访问的其他代码一起存储,并可以使用处理器的提取、解码、调度、执行、退出和/或其他逻辑电路来提取和/或处理。

[0067] 在至少一个实施例中,激活存储620可以是高速缓存存储器、DRAM、SRAM、非易失性存储器(例如,闪存)或其他存储。在至少一个实施例中,激活存储620可以完全地或部分地在一个或多个处理器或其他逻辑电路内部或外部。在至少一个实施例中,可以根据片上与片外的可用存储、执行训练和/或推理功能的延迟要求、在推理和/或训练神经网络中使用的数据的批大小或这些因素的某种组合,选择激活存储620是在处理器的内部还是外部,例如,或者包含DRAM、SRAM、闪存或其他存储类型。在至少一个实施例中,图6A中所示的推理和/或训练逻辑615可以与专用集成电路(“ASIC”)结合使用,例如来自Google的TensorFlow®处理单元、来自Graphcore™的推理处理单元(IPU)或来自Intel公司的Nervana®(例如“Lake Crest”)处理器。在至少一个实施例中,图6A所示的推理和/或训练逻辑615可与中央处理单元(“CPU”)硬件、图形处理单元(“GPU”)硬件或其他硬件(例如现场可编程门阵列(“FPGA”))结合使用。

[0068] 图6B示出了根据至少一个实施例的推理和/或训练逻辑615。在至少一个实施例中,推理和/或训练逻辑615可以包括但不限于硬件逻辑,其中计算资源被专用或以其他方式排他地连同对应于神经网络内的一层或更多层神经元的权重值或其他信息一起使用。在至少一个实施例中,图6B中所示的推理和/或训练逻辑615可以与专用集成电路(ASIC)结合使用,例如来自Google的TensorFlow®处理单元,来自Graphcore™的推理处理单元(IPU)或来自Intel公司的Nervana®(例如“Lake Crest”)处理器。在至少一个实施例中,图6B中所示的推理和/或训练逻辑615可以与中央处理单元(CPU)硬件、图形处理单元(GPU)硬件或其他硬件(例如现场可编程门阵列(FPGA))结合使用。在至少一个实施例中,推理和/或训练逻辑615包括但不限于代码和/或数据存储601以及代码和/或数据存储605,其可以用于存储代码(例如,图代码)、权重值和/或其他信息,包括偏置值、梯度信息、动量值和/或其他参数或超参数信息。在图6B中所示的至少一个实施例中,代码和/或数据存储601以及代码和/或数据存储605中的每一个都分别与专用计算资源(例如计算硬件602和计算硬件606)相关联。在至少一个实施例中,计算硬件602和计算硬件606中的每一个包括一个或多个ALU,这些ALU仅分别对存储在代码和/或数据存储601和代码和/或数据存储605中的信息执行数学函数(例如线性代数函数),执行函数的结果被存储在激活存储620中。

[0069] 在至少一个实施例中,代码和/或数据存储601和605以及相应的计算硬件602和606中的每一个分别对应于神经网络的不同层,使得从代码和/或数据存储601和计算硬件602的一个“存储/计算对601/602”得到的激活提供作为代码和/或数据存储605和计算硬件606的下一个“存储/计算对605/606”的输入,以便反映神经网络的概念组织。在至少一个实

施例中,每个存储/计算对601/602和605/606可以对应于一个以上的神经网络层。在至少一个实施例中,在推理和/或训练逻辑615中可以包括在存储计算对601/602和605/606之后或与之并行的附加存储/计算对(未示出)。

[0070] 数据中心

[0071] 图7示出了可以使用至少一个实施例的示例数据中心700。在至少一个实施例中,数据中心700包括数据中心基础设施层710、框架层720、软件层730和应用程序层740。

[0072] 在至少一个实施例中,如图7所示,数据中心基础设施层710可以包括资源协调器712、分组计算资源714和节点计算资源(“节点C.R.”)716(1)-716(N),其中“N”代表正整数。在至少一个实施例中,节点C.R.716(1)-716(N)可以包括但不限于任何数量的中央处理单元(“CPU”)或其他处理器(包括加速器、现场可编程门阵列(FPGA)、图形处理器等),存储器设备(例如动态只读存储器)、存储设备(例如,固态硬盘或磁盘驱动器),网络输入/输出(“NW I/O”)设备,网络交换机,虚拟机(“VM”),电源模块和冷却模块等。在至少一个实施例中,节点C.R.716(1)-716(N)中的一个或多个节点C.R.可以是具有一个或多个上述计算资源的服务器。

[0073] 在至少一个实施例中,分组计算资源714可以包括容纳在一个或多个机架内的节点C.R.的单独分组(未示出),或者容纳在各个地理位置的数据中心内的许多机架(也未示出)。分组的计算资源714内的节点C.R.的单独分组可以包括可以被配置或分配为支持一个或多个工作负载的分组的计算、网络、存储器或存储资源。在至少一个实施例中,可以将包括CPU或处理器的几个节点C.R.分组在一个或多个机架内,以提供计算资源来支持一个或多个工作负载。在至少一个实施例中,一个或多个机架还可以包括任何数量的电源模块、冷却模块和网络交换机,以任意组合。

[0074] 在至少一个实施例中,资源协调器712可以配置或以其他方式控制一个或多个节点C.R.716(1)-716(N)和/或分组的计算资源714。在至少一个实施例中,资源协调器712可以包括用于数据中心700的软件设计基础结构(“SDI”)管理实体。在至少一个实施例中,资源协调器可以包括硬件、软件或其某种组合。

[0075] 在至少一个实施例中,如图7所示,框架层720包括作业调度器722、配置管理器724、资源管理器726和分布式文件系统728。在至少一个实施例中,框架层720可以包括支持软件层730的软件732和/或应用程序层740的一个或多个应用程序742的框架。在至少一个实施例中,软件732或应用程序742可以分别包括基于Web的服务软件或应用程序,例如由Amazon Web Services,Google Cloud和Microsoft Azure提供的服务或应用程序。在至少一个实施例中,框架层720可以是但不限于一种免费和开放源软件网络应用程序框架,例如可以利用分布式文件系统728来进行大范围数据处理(例如“大数据”)的Apache Spark™(以下称为“Spark”)。在至少一个实施例中,作业调度器732可以包括Spark驱动器,以促进对数据中心700的各个层所支持的工作负载进行调度。在至少一个实施例中,配置管理器724可以能够配置不同的层,例如软件层730和包括Spark和用于支持大规模数据处理的分布式文件系统728的框架层720。在至少一个实施例中,资源管理器726能够管理映射到或分配用于支持分布式文件系统728和作业调度器722的集群或分组计算资源。在至少一个实施例中,集群或分组计算资源可以包括数据中心基础设施层710上的分组的计算资源714。在至少一个实施例中,资源管理器726可以与资源协调器712协调以管理这些映射的或分配的

计算资源。

[0076] 在至少一个实施例中,包括在软件层730中的软件732可以包括由节点C.R.716(1)-716(N)的至少一部分,分组的计算资源714和/或框架层720的分布式文件系统728使用的软件。一种或更多种类型的软件可以包括但不限于Internet网页搜索软件、电子邮件病毒扫描软件、数据库软件和流视频内容软件。

[0077] 在至少一个实施例中,应用程序层740中包括的一个或更多个应用程序742可以包括由节点C.R.716(1)-716(N)的至少一部分、分组计算资源714和/或框架层720的分布式文件系统728使用的一种或更多种类型的应用程序。一种或更多种类型的应用程序可以包括但不限于任何数量的基因组学应用程序、认知计算、和机器学习应用程序,包括训练或推理软件,机器学习框架软件(例如PyTorch、TensorFlow、Caffe等)或其他与一个或更多个实施例结合使用的机器学习应用程序。

[0078] 在至少一个实施例中,配置管理器724、资源管理器726和资源协调器712中的任何一个可以基于以任何技术上可行的方式获取的任何数量和类型的数据来实现任何数量和类型的自我修改动作。在至少一个实施例中,自我修改动作可以减轻数据中心700的数据中心操作员做出可能不好的配置决定并且可以避免数据中心的未充分利用和/或执行差的部分。

[0079] 在至少一个实施例中,数据中心700可以包括工具、服务、软件或其他资源,以根据本文所述的一个或更多个实施例来训练一个或更多个机器学习模型或者使用一个或更多个机器学习模型来预测或推理信息。例如,在至少一个实施例中,可以通过使用上文关于数据中心700描述的软件和计算资源,根据神经网络架构通过计算权重参数来训练机器学习模型。在至少一个实施例中,通过使用通过本文所述的一种或更多种训练技术计算出的权重参数,可以使用上面与关于数据中心700所描述的资源,使用对应于一个或更多个神经网络的经训练的机器学习模型来推理或预测信息。

[0080] 在至少一个实施例中,数据中心可以使用CPU、专用集成电路(ASIC)、GPU、FPGA或其他硬件来使用上述资源来执行训练和/或推理。此外,上述的一个或更多个软件和/或硬件资源可以配置成一种服务,以允许用户训练或执行信息推理,例如图像识别、语音识别或其他人工智能服务。

[0081] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在图7的系统中使用,用于至少部分地基于使用神经网络训练操作、神经网络函数和/或体系架构、或者本文所述的神经网络用例计算的权重参数推理或预测操作。

[0082] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0083] 计算机系统

[0084] 图8是示出了根据至少一个实施例示例性计算机系统的框图,该示例性计算机系统可以是具有互连的设备和组件的系统,片上系统(SOC)或它们的某种形成有处理器的组合,该处理器可以包括执行单元以执行指令。在至少一个实施例中,根据本公开,例如本文所述的实施例,计算机系统800可以包括但不限于组件,例如处理器802,其执行单元包括逻

辑以执行用于过程数据的算法。在至少一个实施例中,计算机系统800可以包括处理器,例如可从加利福尼亚圣塔克拉拉的英特尔公司(Intel Corporation of Santa Clara, California)获得的PENTIUM®处理器家族、Xeon™、Itanium®、XScale™和/或StrongARM™, Intel®Core™或Intel®Nervana™微处理器,尽管也可以使用其他系统(包括具有其他微处理器的PC、工程工作站、机顶盒等)。在至少一个实施例中,计算机系统800可以执行可从华盛顿州雷蒙德市的微软公司(Microsoft Corporation of Redmond, Wash.)获得的WINDOWS操作系统版本,尽管其他操作系统(例如UNIX和Linux)、嵌入式软件和/或图形用户界面也可以使用。

[0085] 实施例可以用在其他设备中,例如手持设备和嵌入式应用程序。手持设备的一些示例包括蜂窝电话、互联网协议(Internet Protocol)设备、数码相机、个人数字助理(“PDA”)和手持PC。在至少一个实施例中,嵌入式应用程序可以包括微控制器、数字信号处理器(“DSP”)、片上系统、网络计算机(“NetPC”)、机顶盒、网络集线器、广域网(“WAN”)交换机,或根据至少一个实施例可以执行一个或更多个指令的任何其他系统。

[0086] 在至少一个实施例中,计算机系统800可包括但不限于处理器802,该处理器802可包括但不限于一个或更多个执行单元808,以根据本文描述的技术执行机器学习模型训练和/或推理。在至少一个实施例中,计算机系统800是单处理器台式机或服务器系统,但是在另一实施例中,计算机系统800可以是多处理器系统。在至少一个实施例中,处理器802可以包括但不限于复杂指令集计算机(“CISC”)微处理器、精简指令集计算(“RISC”)微处理器、超长指令字(“VLIW”)微处理器、实现指令集组合的处理器,或任何其他处理器设备,例如数字信号处理器。在至少一个实施例中,处理器802可以耦合到处理器总线810,该处理器总线810可以在处理器802与计算机系统800中的其他组件之间传输数据信号。

[0087] 在至少一个实施例中,处理器802可以包括但不限于1级(“L1”)内部高速缓存存储器(“cache”)804。在至少一个实施例中,处理器802可以具有单个内部高速缓存或多级内部缓存。在至少一个实施例中,高速缓存存储器可以驻留在处理器802的外部。根据特定的实现和需求,其他实施例也可以包括内部和外部高速缓存的组合。在至少一个实施例中,寄存器文件806可以在各种寄存器中存储不同类型的数据,包括但不限于整数寄存器、浮点寄存器、状态寄存器和指令指针寄存器。

[0088] 在至少一个实施例中,包括但不限于执行整数和浮点运算的逻辑的执行单元808,其也位于处理器802中。在至少一个实施例中,处理器802还可以包括微码(“ucode”)只读存储器(“ROM”),用于存储某些宏指令的微代码。在至少一个实施例中,执行单元808可以包括用于处理封装指令集809的逻辑。在至少一个实施例中,通过将打包指令集809包括在通用处理器的指令集中,以及要执行指令的相关电路,可以使用通用处理器802中的打包(packed)数据来执行许多多媒体应用程序使用的操作。在一个或更多个实施例中,可以通过使用处理器的数据总线的全宽度来在封装的数据上执行操作来加速和更有效地执行许多多媒体应用程序,这可能不需要在该处理器的数据总线上传输较小的数据单元来一次执行一个数据元素的一个或更多个操作。

[0089] 在至少一个实施例中,执行单元808也可以用在微控制器、嵌入式处理器、图形设备、DSP和其他类型的逻辑电路中。在至少一个实施例中,计算机系统800可以包括但不限于存储器820。在至少一个实施例中,存储器820可以实现为动态随机存取存储器(“DRAM”)设

备、静态随机存取存储器 (“SRAM”) 设备、闪存设备或其他存储器设备。在至少一个实施例中, 存储器820可以存储由处理器802可以执行的由数据信号表示的指令819和/或数据821。

[0090] 在至少一个实施例中, 系统逻辑芯片可以耦合到处理器总线810和存储器820。在至少一个实施例中, 系统逻辑芯片可以包括但不限于存储器控制器集线器 (“MCH”) 816, 并且处理器802可以经由处理器总线810与MCH 816通信。在至少一个实施例中, MCH 816可以提供到存储器820的高带宽存储器路径818以用于指令和数据存储以及用于图形命令、数据和纹理的存储。在至少一个实施例中, MCH 816可以在处理器802、存储器820和计算机系统800中的其他组件之间启动数据信号, 并且在处理器总线810、存储器820和系统I/O接口822之间桥接数据信号。在至少一个实施例中, 系统逻辑芯片可以提供用于耦合到图形控制器的图形端口。在至少一个实施例中, MCH 816可以通过高带宽存储器路径818耦合到存储器820, 并且图形/视频卡812可以通过加速图形端口 (Accelerated Graphics Port) (“AGP”) 互连814耦合到MCH 816。

[0091] 在至少一个实施例中, 计算机系统800可以使用系统I/O接口822, 其是将MCH 816耦合到I/O控制器集线器 (“ICH”) 830的专有集线器接口总线。在至少一个实施例中, ICH 830可以通过本地I/O总线提供与某些I/O设备的直接连接。在至少一个实施例中, 本地I/O总线可以包括但不限于用于将外围设备连接到存储器820、芯片组和处理器802的高速I/O总线。示例可以包括但不限于音频控制器829、固件集线器 (“Flash BIOS”) 828、无线收发器826、数据存储824、包含用户输入和键盘接口825的传统I/O控制器823、串行扩展端口827 (例如通用串行总线 (USB)) 和网络控制器834。数据存储824可以包括硬盘驱动器、软盘驱动器、CD-ROM设备、闪存设备或其他大容量存储设备。

[0092] 在至少一个实施例中, 图8示出了包括互连的硬件设备或“芯片”的系统, 而在其他实施例中, 图8可以示出示例性片上系统 (SoC)。在至少一个实施例中, 图8中示出的设备可以与专有互连、标准化互连 (例如, PCIe) 或其某种组合互连。在至少一个实施例中, 计算机系统800的一个或多个组件使用计算快速链路 (CXL) 互连来互连。

[0093] 推理和/或训练逻辑615用于执行与一个或多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中, 推理和/或训练逻辑615可以在图8的系统中使用, 用于至少部分地基于使用神经网络训练操作、神经网络函数和/或体系架构、或者本文所述的神经网络用例计算的权重参数推理或预测操作。

[0094] 在至少一个实施例中, 此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0095] 图9是示出根据至少一个实施例的用于利用处理器910的电子设备900的框图。在至少一个实施例中, 电子设备900可以是, 例如但不限于, 笔记本电脑、塔式服务器、机架服务器、刀片服务器、膝上型计算机、台式机、平板电脑、移动设备、电话、嵌入式计算机或任何其他合适的电子设备。

[0096] 在至少一个实施例中, 系统900可以包括但不限于通信地耦合到任何合适数量或种类的组件、外围设备、模块或设备的处理器910。在至少一个实施例中, 处理器910使用总线或接口耦合, 诸如I²C总线、系统管理总线 (“SMBus”)、低引脚数 (LPC) 总线、串行外围接口 (“SPI”)、高清音频 (“HDA”) 总线、串行高级技术附件 (“SATA”) 总线、通用串行总线 (“USB”)

(1、2、3版等)或通用异步接收器/发送器(“UART”)总线。在至少一个实施例中,图9示出了系统,该系统包括互连的硬件设备或“芯片”,而在其他实施例中,图9可以示出示例性片上系统(SoC)。在至少一个实施例中,图9中所示的设备可以与专有互连线、标准化互连(例如,PCIe)或其某种组合互连。在至少一个实施例中,图9的一个或多个组件使用计算快速链路(CXL)互连线来互连。

[0097] 在至少一个实施例中,图9可以包括显示器924、触摸屏925、触摸板930、近场通信单元(“NFC”)945、传感器集线器940、热传感器946、快速芯片组(“EC”)935、可信平台模块(“TPM”)938、BIOS/固件/闪存(“BIOS,FW Flash”)922、DSP960、驱动器920(例如固态硬盘(“SSD”)或硬盘驱动器(“HDD”))、无线局域网单元(“WLAN”)950、蓝牙单元952、无线广域网单元(“WWAN”)956、全球定位系统(GPS)955、相机(“USB 3.0相机”)954(例如USB 3.0相机)和/或以例如LPDDR3标准实现的低功耗双倍数据速率(“LPDDR”)存储器单元(“LPDDR3”)915。这些组件可以各自以任何合适的方式实现。

[0098] 在至少一个实施例中,其他组件可以通过本文所述的组件通信地耦合到处理器910。在至少一个实施例中,加速度计941、环境光传感器(“ALS”)942、罗盘943和陀螺仪944可以通信地耦合到传感器集线器940。在至少一个实施例中,热传感器939、风扇937、键盘936和触摸板930可以通信地耦合到EC 935。在至少一个实施例中,扬声器963、耳机964和麦克风(“mic”)965可以通信地耦合到音频单元(“音频编解码器和D类放大器”)962,其又可以通信地耦合到DSP 960。在至少一个实施例中,音频单元962可以包括例如但不限于音频编解码器/解码器(“编解码器”)和D类放大器。在至少一个实施例中,SIM卡(“SIM”)957可以通信地耦合到WWAN单元956。在至少一个实施例中,组件(诸如WLAN单元950和蓝牙单元952以及WWAN单元956)可以被实现为下一代形式因素(NGFF)。

[0099] 推理和/或训练逻辑615用于执行与一个或多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在图9的系统中使用,用于至少部分地基于使用神经网络训练操作、神经网络函数和/或体系架构、或者本文所述的神经网络用例计算的权重参数推理或预测操作。

[0100] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0101] 图10示出了根据至少一个实施例的计算机系统1000。在至少一个实施例中,计算机系统1000配置为实现贯穿本公开描述的各种过程和方法。

[0102] 在至少一个实施例中,计算机系统1000包括但不限于至少一个中央处理单元(“CPU”)1002,该中央处理单元(“CPU”)1002连接到使用任何合适协议实现的通信总线1010,诸如PCI(“外围设备互联”)、外围组件互连Express(“PCI-Express”)、AGP(“加速图形端口”)、超传输或任何其他总线或点对点通信协议。在至少一个实施例中,计算机系统1000包括但不限于主存储器1004和控制逻辑(例如,实现为硬件、软件或其组合),并且数据可以采取随机存取存储器(“RAM”)的形式存储在主存储器1004中。在至少一个实施例中,网络接口子系统(“网络接口”)1022提供到其他计算设备和网络的接口,用于从计算机系统1000接收数据并将数据传输到其他系统。

[0103] 在至少一个实施例中,计算机系统1000在至少一个实施例中包括但不限于输入设

备1008、并行处理系统1012和显示设备1006,它们可以使用常规的阴极射线管(“CRT”)、液晶显示器(“LCD”)、发光二极管(“LED”)、等离子显示器或其他合适的显示技术实现。在至少一个实施例中,从输入设备1008(诸如键盘、鼠标、触摸板、麦克风等)接收用户输入。在至少一个实施例中,前述模块中的每一个可以位于单个半导体平台上以形成处理系统。

[0104] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在图10的系统中使用,用于至少部分地基于使用神经网络训练操作、神经网络函数和/或体系架构、或者本文所述的神经网络用例计算的权重参数推理或预测操作。

[0105] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0106] 图11示出了根据至少一个实施例的计算机系统1100。在至少一个实施例中,计算机系统1100包括但不限于计算机1110和USB盘1120。在至少一个实施例中,计算机1110可以包括但不限于任何数量和类型的处理器(未示出)和存储器(未示出)。在至少一个实施例中,计算机1110包括但不限于服务器、云实例、膝上型计算机和台式计算机。

[0107] 在至少一个实施例中,USB盘1120包括但不限于处理单元1130、USB接口1140和USB接口逻辑1150。在至少一个实施例中,处理单元1130可以是任何指令执行系统、装置或能够执行指令的设备。在至少一个实施例中,处理单元1130可以包括但不限于任何数量和类型的处理核心(未示出)。在至少一个实施例中,处理核心1130包括专用集成电路(“ASIC”),该专用集成电路被优化为执行与机器学习相关联的任何数量和类型的操作。例如,在至少一个实施例中,处理核心1130是张量处理单元(“TPC”),其被优化以执行机器学习推理操作。在至少一个实施例中,处理核心1130是视觉处理单元(“VPU”),其被优化以执行机器视觉和机器学习推理操作。

[0108] 在至少一个实施例中,USB接口1140可以是任何类型的USB连接器或USB插座。例如,在至少一个实施例中,USB接口1140是用于数据和电源的USB 3.0 Type-C插座。在至少一个实施例中,USB接口1140是USB 3.0 Type-A连接器。在至少一个实施例中,USB接口逻辑1150可以包括使处理单元1130能够经由USB连接器1140与设备(例如计算机1110)相连接的任何数量和类型的逻辑。

[0109] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在图11的系统中使用,用于至少部分地基于使用神经网络训练操作、神经网络函数和/或体系架构、或者本文所述的神经网络用例计算的权重参数推理或预测操作。

[0110] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0111] 图12A示出了示例性架构,其中多个GPU 1210-1213通过高速链路1240-1243(例如,总线/点对点互连等)通信地耦合到多个多核心处理器1205-1206。在一个实施例中,高速链路1240-1243支持4GB/s、30GB/s、80GB/s或更高的通信吞吐量。可以使用各种互连协议,包括但不限于PCIe 4.0或5.0以及NVLink 2.0。

[0112] 此外,在一个实施例中,两个或更多个GPU 1210-1213通过高速链路1229-1230互连,该高速链路可以使用与用于高速链路1240-1243的协议/链路相同或不同的协议/链路来实现。类似地,两个或更多个多核心处理器1205-1206可以通过高速链路1228连接,该高速链路可以是以20GB/s、30GB/s、120GB/s或更高的速度运行的对称多处理器(SMP)总线。可替代地,可以使用类似的协议/链路(例如,通过公共互连结构)来完成图12A中所示的各种系统组件之间的所有通信。

[0113] 在一个实施例中,每个多核心处理器1205-1206分别经由存储器互连1226-1227通信地耦合到处理器存储器1201-1202,并且每个GPU 1210-1213分别通过GPU存储器互连1250-1253通信地耦合到GPU存储器1220-1223。存储器互连1226-1227和1250-1253可以利用相同或不同的存储器访问技术。作为示例而非限制,处理器存储器1201-1202和GPU存储器1220-1223可以是易失性存储器,诸如动态随机存取存储器(DRAM)(包括堆叠的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,例如3D XPoint或Nano-Ram。在一个实施例中,处理器存储器1201-1202的某些部分可以是易失性存储器,而另一部分可以是非易失性存储器(例如,使用两级存储器(2LM)层次结构)。

[0114] 如下面所述,尽管各种处理器1205-1206和GPU 1210-1213可以分别物理地耦合到特定存储器1201-1202、1220-1223,可以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)分布在各个物理存储器之间。例如,处理器存储器1201-1202可以各自包含64GB的系统存储器地址空间,并且GPU存储器1220-1223可以各自包含32GB的系统存储器地址空间(在该示例中,导致总计256GB的可寻址存储器)。

[0115] 图12B示出了根据一个示例性实施例的用于多核心处理器1207和图形加速模块1246之间互连的附加细节。图形加速模块1246可以包括集成在线路卡上的一个或更多个GPU芯片,该线路卡经由高速链路1240耦合到处理器1207。替代地,图形加速模块1246可以集成在具有处理器1207的同一封装或芯片上。

[0116] 在至少一个实施例中,所示的处理器1207包括多个核心1260A-1260D,每个核心都具有转换后备缓冲区1261A-1261D和一个或更多个高速缓存1262A-1262D。在至少一个实施例中,核心1260A-1260D可以包括未示出的各种其他组件,用于执行指令和处理数据。高速缓存1262A-1262D可以包括级别1(L1)和级别2(L2)高速缓存。此外,一个或更多个共享高速缓存1256可以被包括在高速缓存1262A-1262D中,并且由各组核心1260A-1260D共享。例如,处理器1207的一个实施例包括24个核心,每个核心具有其自己的L1高速缓存,十二个共享的L2高速缓存,和十二个共享的L3高速缓存。在该实施例中,两个相邻核心共享一个或更多个L2和L3高速缓存。处理器1207和图形加速模块1246与系统存储器1214连接,该系统存储器1214可以包括图12A中的处理器存储器1201-1202。

[0117] 通过一致性总线1264经由核心间通信为存储在各个高速缓存1262A-1262D、1256和系统存储器1214中的数据和指令维护一致性。在至少一个实施例中,例如,每个高速缓存可以具有与其相关联的高速缓存一致性逻辑/电路,以响应于检测到对特定高速缓存行的读取或写入通过一致性总线1264进行通信。在一个实现方式中,通过一致性总线1264实现高速缓存监听协议,以监听(snoop)高速缓存访问。

[0118] 在至少一个实施例中,代理电路1225将图形加速模块1246通信地耦合到一致性总

线1264,从而允许图形加速模块1246作为核心1260A-1260D的对等方参与高速缓存一致性协议。特别地,在至少一个实施例中,接口1235通过高速链路1240(例如,PCIe总线、NVLink等)提供到代理电路1225的连接,并且接口1237将图形加速模块1246连接到链路1240。

[0119] 在一个实现方式中,加速器集成电路1236代表图形加速模块1246的多个图形处理引擎1231,1232,N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。在至少一个实施例中,图形处理引擎1231,1232,N可各自包括单独的图形处理单元(GPU)。替代地,图形处理引擎1231,1232,N可以包括GPU内的不同类型的图形处理引擎,诸如图形执行单元、媒体处理引擎(例如,视频编码器/解码器)、采样器和blit引擎。在至少一个实施例中,图形加速模块1246可以是具有多个图形处理引擎1231-1232,N的GPU,或者图形处理引擎1231-1232,N可以是集成在通用封装、线路卡或芯片上的各个GPU。

[0120] 在一个实施例中,加速器集成电路1236包括存储器管理单元(MMU)1239,用于执行各种存储器管理功能,例如虚拟到物理存储器转换(也称为有效到真实存储器转换),还包括用于访问系统存储器1214的存储器访问协议。MMU 1239还可包括转换后备缓冲区(TLB)(未示出),用于高速缓存虚拟/有效到物理/真实地址转换。在一个实现方式中,高速缓存1238存储命令和数据,用于图形处理引擎1231-1232,N有效地访问。在一个实现方式中,将存储在高速缓存1238和图形存储器1233-1234,M中的数据与核心高速缓存1262A-1262D、1256和系统存储器1214保持一致。如前所述,可以经由代表高速缓存1238和图形存储器1233-1234,M的代理电路1225来完成该任务(例如,将与处理器高速缓存1262A-1262D、1256上的高速缓存行的修改/访问有关的更新发送到高速缓存1238,并从高速缓存1238接收更新)。

[0121] 在至少一个实施例中,一组寄存器1245存储由图形处理引擎1231-1232,N执行的线程的上下文数据,并且上下文管理电路1248管理线程上下文。例如,上下文管理电路1248可以执行保存和恢复操作,以在上下文切换期间保存和恢复各个线程的上下文(例如,其中保存第一线程并且存储第二线程,以便可以由图形处理引擎执行第二线程)。例如,上下文管理电路1248在上下文切换时,可以将当前寄存器值存储到存储器中的(例如,由上下文指针标识的)指定区域。然后,当返回上下文时可以恢复寄存器值。在一个实施例中,中断管理电路1247接收并处理从系统设备接收的中断。

[0122] 在一个实现方式中,MMU 1239将来自图形处理引擎1231的虚拟/有效地址转换为系统存储器1214中的真实/物理地址。在至少一个实施例中,加速器集成电路1236支持多个(例如,4、8、16)图形加速器模块1246和/或其他加速器设备。图形加速器模块1246可以专用于在处理器1207上执行的单个应用程序,或者可以在多个应用程序之间共享。在至少一个实施例中,呈现了虚拟化的图形执行环境,其中图形处理引擎1231-1232,N的资源与多个应用程序或虚拟机(VM)共享。在至少一个实施例中,可以基于处理要求和与VM和/或应用程序相关联的优先级,将资源细分为“切片”,其被分配给不同的VM和/或应用程序。

[0123] 在至少一个实施例中,加速器集成电路1236作为图形加速模块1246的系统的桥来执行,并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1236可以为主机处理器提供虚拟化设施,以管理图形处理引擎1231-1232,N的虚拟化、中断和存储器管理。

[0124] 由于图形处理引擎1231-1232,N的硬件资源被明确地映射到主机处理器1207看到的真实地址空间,因此任何主机处理器都可以使用有效地址值直接寻址这些资源。加速器

集成电路1236的一个功能是物理分离图形处理引擎1231-1232,N,使得它们在系统看来为独立的单元。

[0125] 在至少一个实施例中,一个或多个图形存储器1233-1234,M分别耦合到每个图形处理引擎1231-1232,N。图形存储器1233-1234,M存储指令和数据,所述指令和数据由每个图形处理引擎1231-1232,N处理。在至少一个实施例中,图形存储器1233-1234,M可以是易失性存储器,例如DRAM(包括堆叠的DRAM)、GDDR存储器(例如,GDDR5,GDDR6)或HBM,和/或可以是非易失性存储器,例如3D XPoint或Nano-Ram。

[0126] 在一个实施例中,为了减少链路1240上的数据流量,使用偏置技术以确存储存储在图形存储器1233-1234,M中的数据是图形处理引擎1231-1232,N最常使用的,并且最好核心1260A-1260D不使用(至少不经常使用)的数据。类似地,偏置机制试图将核心(并且优选地不是图形处理引擎1231-1232,N)需要的数据保持在核心的高速缓存1262A-1262D、1256和系统存储器1214中。

[0127] 图12C示出了另一个示例性实施例,其中加速器集成电路1236被集成在处理器1207内。在该实施例中,图形处理引擎1231-1232,N经由接口1237和接口1235(同样可以利用任何形式的总线或接口协议)通过高速链路1240直接与加速器集成电路1236通信。加速器集成电路1236可以执行与关于图12B描述的操作相同的操作。但是由于它紧密靠近一致性总线1264和高速缓存1262A-1262D、1256,可能具有更高的吞吐量。至少一个实施例支持不同的编程模型,包括专用进程编程模型(无图形加速模块虚拟化)和共享编程模型(具有虚拟化),所述编程模型可以包括由加速器集成电路1236控制的编程模型和由图形加速模块1246控制的编程模型。

[0128] 在至少一个实施例中,图形处理引擎1231-1232,N专用于单个操作系统下的单个应用程序或进程。在至少一个实施例中,单个应用程序可以将其他应用程序请求汇聚(funnel)到图形处理引擎1231-1232,N,从而在VM/分区内提供虚拟化。

[0129] 在至少一个实施例中,图形处理引擎1231-1232,N可以被多个VM/应用程序分区共享。在至少一个实施例中,共享模型可以使用系统管理程序来虚拟化图形处理引擎1231-1232,N,以允许每个操作系统进行访问。对于没有管理程序的分分区系统,操作系统拥有图形处理引擎1231-1232,N。在至少一个实施例中,操作系统可以虚拟化图形处理引擎1231-1232,N,以提供对每个进程或应用程序的访问。

[0130] 在至少一个实施例中,图形加速模块1246或个体图形处理引擎1231-1232,N使用进程句柄来选择进程元素。在至少一个实施例中,进程元素被存储在系统存储器1214中,并且可使用本文所述的有效地址到真实地址转换技术来寻址。在至少一个实施例中,进程句柄可以是特定于实现方式的值,其在向图形处理引擎1231-1232,N注册其上下文时提供给主机进程(即,调用系统软件以将进程元素添加到进程元素链接列表)。在至少一个实施例中,进程句柄的较低16位可以是进程元素在进程元素链接列表中的偏移量。

[0131] 图12D示出了示例性加速器集成切片1290。如本文所使用的,“切片”包括加速器集成电路1236的处理资源的指定部分。系统存储器1214中的应用程序有效地址空间1282存储进程元素1283。在一个实施例中,响应于来自在处理器1207上执行的应用程序1280的GPU调用1281,存储进程元素1283。进程元素1283包含相应的应用程序1280的进程状态。包含在进程元素1283中的工作描述符(WD)1284可以是由应用程序请求的单个作业,或者可以包含指

向作业队列的指针。在至少一个实施例中,WD 1284是指向应用程序的地址空间1282中的作业请求队列的指针。

[0132] 图形加速模块1246和/或各个图形处理引擎1231-1232,N可以由系统中所有进程或进程子集共享。在至少一个实施例中,可以包括用于设置进程状态并将WD 1284发送到图形加速模块1246以在虚拟化环境中开始作业的基础设施。

[0133] 在至少一个实施例中,专用进程编程模型是特定于实现方式的。在该模型中,单个进程拥有图形加速模块1246或个体图形处理引擎1231。当图形加速模块1246由单个进程拥有时,管理程序初始化用于所拥有的分区的加速器集成电路1236,当指派了图形加速模块1246时,操作系统初始化用于所拥有的进程的加速器集成电路1236。

[0134] 在操作中,加速器集成切片1290中的WD获取单元1291获取下一个WD 1284,其包括要由图形加速模块1246的一个或更多个图形处理引擎完成的工作的指示。来自WD 1284的数据可以存储在寄存器1245中,并由MMU 1239、中断管理电路1247和/或上下文管理电路1248使用,如图所示。例如,MMU 1239的一个实施例包括用于访问OS虚拟地址空间1285内的段/页表1286的段/页漫游电路。中断管理电路1247可以处理从图形加速模块1246接收的中断事件1292。当执行图形操作时,由图形处理引擎1231-1232,N生成的有效地址1293被MMU 1239转换为真实地址。

[0135] 在一个实施例中,为每个图形处理引擎1231-1232,N和/或图形加速模块1246复制相同的一组寄存器1245,并且所述寄存器1245可以由管理程序或操作系统初始化。这些复制的寄存器中的每一个可以被包括在加速器集成切片1290中。可以由管理程序初始化的示例性寄存器在表1中示出。

表 1—管理程序初始化的寄存器	
1	切片控制寄存器
2	实地址 (RA) 调度的进程区域指针
3	权限屏蔽覆写寄存器
4	中断向量表条目偏移量
5	中断向量表条目限制
6	状态寄存器
7	逻辑分区 ID
8	实地址 (RA) 管理程序加速器利用率记录指针
9	存储描述寄存器

[0138] 表2中示出了可由操作系统初始化的示例性寄存器。

[0139]

表 2—操作系统初始化的寄存器	
寄存器#	描述
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) 加速器利用率记录指针
4	虚拟地址 (VA) 存储段表指针
5	权限屏蔽
6	工作描述符

[0140] 在至少一个实施例中,每个WD 1284特定于特定的图形加速模块1246和/或图形处理引擎1231-1232,N。它包含图形处理引擎1231-1232,N完成工作所需的所有信息,或者它可以是指向存储器位置的指针,在该存储器位置应用程序已经设置了要完成的工作的命令队列。

[0141] 图12E示出了共享模型的一个示例性实施例的附加细节。该实施例包括管理程序真实地址空间1298,其中存储了进程元素列表1299。可经由管理程序1296来访问管理程序实地址空间1298,所述管理程序1296虚拟化用于操作系统1295的图形加速模块引擎。

[0142] 在至少一个实施例中,共享编程模型允许来自系统中全部分区或分区子集的全部进程或进程子集使用图形加速模块1246。存在两种编程模型,其中图形加速模块1246由多个进程和分区共享,即,时间切片共享和图形定向共享。

[0143] 在至少一个实施例中,在该模型中,系统管理程序1296拥有图形加速模块1246,并使其功能可用于所有操作系统1295。对于图形加速模块1246通过系统管理程序1296支持虚拟化,图形加速模块1246可以遵守一下要求:1) 应用程序的作业请求必须是自主的 (即,不需要在作业之间保持状态),或者图形加速模块1246必须提供上下文保存和恢复机制,2) 图形加速模块1246保证应用程序的作业请求在指定的时间量内完成,包括任何转换错误,或者图形加速模块1246提供了抢占作业处理的能力,并且3) 在有向共享编程模型中进行操作时,必须确保图形加速模块1246进程之间的公平性。

[0144] 在至少一个实施例中,需要应用程序1280使用图形加速模块1246类型、工作描述符 (WD)、权限屏蔽寄存器 (AMR) 值和上下文保存/恢复区域指针 (CSRP) 进行操作系统1295系统调用。在至少一个实施例中,图形加速模块1246类型描述了用于系统调用的目标加速函数。在至少一个实施例中,图形加速模块1246类型可以是系统特定的值。在至少一个实施例中,WD是专门为图形加速模块1246格式化的,并且可以采用图形加速模块1246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的形式,或描述要由图形加速模块1246完成的工作的任何其他数据结构。在一个实施例中,AMR值是用于当前进程的AMR状态。在至少一个实施例中,传递给操作系统的值与设置AMR的应用程序类似。如果加速器集成电路1236和图形加速模块1246的实现不支持用户权限屏蔽覆写寄存器 (UAMOR),则在管理程序调用中传递AMR之前,操作系统可以将当前UAMOR值应用于AMR值。管理程序1296可以在将AMR放入进程元素1283中之前选择性地应用当前权限屏蔽覆写寄存器 (AMOR) 值。

在至少一个实施例中,CSRP是寄存器1245中的一个,所述寄存器包含应用程序的有效地址空间1282中的区域的有效地址,供图形加速模块1246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者当作业被抢占时,则该指针是可选的。在至少一个实施例中,上下文保存/恢复区域可以是固定的系统存储器。

[0145] 在接收到系统调用时,操作系统1295可以验证应用程序1280已经注册并且被授予使用图形加速模块1246的权限。然后,操作系统1295使用表3中所示的信息来调用管理程序1296。

表 3—操作系统到管理程序的调用参数	
1	工作描述符 (WD)
2	权限屏蔽寄存器 (AMR) 值 (可能被屏蔽)
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) 加速器利用率记录指针 (AURP)
6	存储段表指针的虚拟地址 (SSTP)
7	逻辑中断服务号 (LISN)

[0148] 在接收到管理程序调用时,管理程序1296验证操作系统1295已注册并被授予使用图形加速模块1246的权限。然后,管理程序1296将进程元素1283放入相应的图形加速模块1246类型的进程元素链接列表中。进程元素可以包括表4中所示的信息。

表 4—进程元素信息	
1	工作描述符 (WD)
2	权限屏蔽寄存器 (AMR) 值 (可能被屏蔽)
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) 加速器利用率记录指针 (AURP)
6	存储段表指针的虚拟地址 (SSTP)
7	逻辑中断服务号 (LISN)
8	从管理程序调用参数派生的中断向量表
9	状态寄存器 (SR) 值
10	逻辑分区 ID (LPID)
11	真实地址 (RA) 管理程序加速器利用率记录指针
12	存储描述符寄存器 (SDR)

[0150] 在至少一个实施例中,管理程序初始化多个加速器集成切片1290寄存器1245。

[0151] 如图12F所示,在至少一个实施例中,使用统一存储器,所述统一存储器可经由用于访问物理处理器存储器1201-1202和GPU存储器1220-1223的公共虚拟存储器地址空间来寻址。在该实现方式中,在GPU1210-1213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1201-1202,反之亦然,从而简化了可编程性。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1201,第二部分被分配给第二处理器存储器1202,第三部分被分配给GPU存储器1220,以此类推。在至少一个实施例中,整个虚拟/有效存储器空间(有时称为有效地址空间)由此分布在处理器存储器1201-1202和GPU存储器1220-1223的每一个中,从而允许任何处理器或GPU采用映射到任何物理存储器的虚拟地址访问该存储器。

[0152] 在一个实施例中,一个或更多个MMU 1239A-1239E内的偏置/一致性管理电路1294A-1294E确保一个或更多个主机处理器(例如,1205)与GPU 1210-1213的高速缓存之间的高速缓存一致性,并实现指示应在其中存储某些类型的数据的物理存储器的偏置技术。虽然在图12F中示出了偏置/一致性管理电路1294A-1294E的多个实例,但可以在一个或更多个主机处理器1205的MMU内和/或在加速器集成电路1236内实现偏置/一致性电路。

[0153] 一个实施例允许将GPU附加的存储器1220-1223映射为系统存储器的一部分,并使用共享虚拟存储器(SVM)技术进行访问,但不会遭受与完整系统高速缓存一致性相关的性能缺陷。在至少一个实施例中,将GPU附加的存储器1220-1223作为系统存储器来访问而无需繁重的高速缓存一致性开销的能力为GPU卸载提供了有利的操作环境。该布置允许主机处理器1205的软件设置操作数并访问计算结果,而没有传统的I/O DMA数据拷贝的开销。这样的传统拷贝包括驱动程序调用、中断和存储器映射I/O(MMIO)访问,相对于简单的存储器访问而言,这些访问效率均较低。在至少一个实施例中,在没有高速缓存一致性开销的情况下访问GPU附加的存储器1220-1223的能力对于卸载的计算的执行时间可能是关键的。例如,在具有大量流式写入存储器流量的情况下,高速缓存一致性开销可以显著降低GPU 1210-1213所看到的有效写入带宽。在至少一个实施例中,操作数设置的效率、结果访问的效率和GPU计算的效率可能会在确定GPU卸载的有效性方面发挥作用。

[0154] 在至少一个实施例中,GPU偏置和主机处理器偏置的选择由偏置跟踪器数据结构驱动。例如,可以使用偏置表,所述偏置表可以是页面粒度结构(例如,以存储器页面的粒度来控制),该页面粒度结构包括每个GPU附加的存储器页面1或2位。在至少一个实施例中,在GPU 1210-1213中具有或不具有偏置高速缓存(例如,用于高速缓存偏置表的频繁/最近使用的条目)的情况下,可以在一个或更多个GPU附加的存储器1220-1223的被盗存储器范围中实现偏置表。替代地,可以在GPU内维护整个偏置表。

[0155] 在至少一个实施例中,在实际访问GPU存储器之前,访问与对GPU附加的存储器1220-1223的每次访问相关联的偏置表条目,从而引起以下操作。首先,来自GPU 1210-1213的在GPU偏置中找到其页面的本地请求被直接转发到对应的GPU存储器1220-1223。来自GPU的在主机偏置中找到其页面的本地请求被转发至处理器1205(例如,通过本文所述的高速链路)。在一个实施例中,来自处理器1205的在主机处理器偏置中找到所请求页面的请求完成了与正常存储器读取类似的请求。替代地,可以将指向GPU偏置页面的请求转发到GPU 1210-1213。在至少一个实施例中,如果GPU当前不使用页面,则GPU可随后将页面迁移到主

机处理器偏置。在至少一个实施例中,页面的偏置状态可以通过基于软件的机制、基于硬件辅助的软件的机制、或者在有限的情况下通过纯粹基于硬件的机制来改变。

[0156] 一种用于改变偏置状态的机制采用API调用(例如OpenCL),所述API调用随后调用GPU的设备驱动程序,所述设备驱动程序随后发送消息(或使命令描述符入队)到GPU,引导GPU改变偏置状态,并在某些迁移中在主机中执行高速缓存刷新操作。在至少一个实施例中,高速缓存刷新操作用于从主机处理器1205偏置到GPU偏置的迁移,但是不用于相反的迁移。

[0157] 在一个实施例中,高速缓存一致性是通过暂时渲染主机处理器1205无法高速缓存的GPU偏置页面来维护的。为了访问这些页面,处理器1205可以请求来自GPU 1210的访问,GPU 1210可以或可以不立即授予访问权限。因此,为了减少处理器1205和GPU 1210之间的通信,确保GPU偏置页面是GPU所需的页面而不是主机处理器1205所需的页面是有益的,反之亦然。

[0158] 推理和/或训练逻辑615用于执行一个或更多个实施例。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。

[0159] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0160] 图13示出了根据本文所述的各个实施例的示例性集成电路和相关联的图形处理器,其可以使用一个或更多个IP核心来制造。除了图示之外,在至少一个实施例中可以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。

[0161] 图13是示出根据至少一个实施例的可使用一个或更多个IP核心制造的芯片集成电路1300上的示例性系统的框图。在至少一个实施例中,集成电路1300包括一个或更多个应用程序处理器1305(例如,CPU)、至少一个图形处理器1310,并且可以另外包括图像处理器1315和/或视频处理器1320,其中任意一个可能是模块化IP核心。在至少一个实施例中,集成电路1300包括外围或总线逻辑,其包括USB控制器1325、UART控制器1330、SPI/SDIO控制器1335和I22S/I22C控制器1340。在至少一个实施例中,集成电路1300可以包括显示设备1345耦合到高清多媒体接口(HDMI)控制器1350和移动工业处理器接口(MIPI)显示接口1355中的一个或更多个。在至少一个实施例中,存储可以由闪存子系统1360提供,包括闪存和闪存控制器。在至少一个实施例中,可以经由存储器控制器1365提供存储器接口以用于访问SDRAM或SRAM存储器设备。在至少一个实施例中,一些集成电路还包括嵌入式安全引擎1370。

[0162] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在集成电路1300中用于至少部分地基于使用神经网络训练操作、神经网络功能和/或架构或本文描述的神经网络用例计算的权重参数来推理或预测操作。

[0163] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0164] 图14A-14B示出了根据本文描述的各个实施例的可使用一个或更多个IP核心制造的示例性集成电路和相关联的图形处理器。除了所示出的内容之外,在至少一个实施例中

可以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。

[0165] 图14A-14B是示出了根据本文描述的实施例的在SoC内使用的示例性图形处理器的框图。图14A示出了根据至少一个实施例的可使用一个或更多个IP核心制造的片上系统集成电路的示例性图形处理器1410。图14B示出了根据至少一个实施例的可以使用一个或更多个IP核心制造的片上系统集成电路的附加示例性图形处理器1440。在至少一个实施例中,图14A的图形处理器1410是低功率图形处理器核心。在至少一个实施例中,图14B的图形处理器1440是更高性能的图形处理器核心。在至少一个实施例中,图形处理器1410、1440中的每个图形处理器可以是图13的图形处理器1310的变体。

[0166] 在至少一个实施例中,图形处理器1410包括顶点处理器1405和一个或更多个片段处理器1415A-1415N(例如,1415A、1415B、1415C、1415D至1415N-1和1415N)。在至少一个实施例中,图形处理器1410可经由单独逻辑执行不同的着色器程序,使得顶点处理器1405被优化为执行用于顶点着色器程序的操作,而一个或更多个片段处理器1415A-1415N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。在至少一个实施例中,顶点处理器1405执行3D图形管线的顶点处理阶段且产生图元和顶点数据。在至少一个实施例中,一个或更多个片段处理器1415A-1415N使用由顶点处理器1405生成的图元和顶点数据来产生显示在显示设备上的帧缓冲区。在至少一个实施例中,片段处理器1415A-1415N被优化为执行如OpenGL API中提供的片段着色器程序,其可用于执行与如直接3D API中提供的像素着色器程序类似的操作。

[0167] 在至少一个实施例中,图形处理器1410另外包括一个或更多个存储器管理单元(MMU) 1420A-1420B、一个或更多个高速缓存1425A-1425B和一个或更多个电路互连1430A-1430B。在至少一个实施例中,一个或更多个MMU 1420A-1420B为图形处理器1410(包括为顶点处理器1405和/或片段处理器1415A到1415N)提供虚拟到物理地址映射,除了存储在一个或更多个高速缓存1425A-1425B中的顶点或图像/纹理数据之外,其还可参考存储在存储器中的顶点或图像/纹理数据。在至少一个实施例中,一个或更多个MMU 1420A-1420B可与系统内的其他MMU同步,包含与一个或更多个应用程序处理器1305、图像处理器1315、和/或图13的视频处理器1320相关联的一个或更多个MMU,从而使得每个处理器1305-1320可以参与共享或统一的虚拟存储器系统。在至少一个实施例中,一个或更多个电路互连1430A-1430B使得图形处理器1410能够经由SoC的内部总线或经由直接连接与SoC内的其他IP核心接合。

[0168] 在至少一个实施例中,图形处理器1440包括图14A的图形处理器1410的一个或更多个MMU 1420A-1420B、一个或更多个高速缓存1425A-1425B和一个或更多个电路互连1430A-1430B。在至少一个实施例中,图形处理器1440包括一个或更多个着色器核心1455A-1455N(例如,1455A、1455B、1455C、1455D、1455E、1455F到1455N-1和1455N),其提供统一着色器核心架构,其中单个核心或类型或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。在至少一个实施例中,着色器核心的数量可以变化。在至少一个实施例中,图形处理器1440包括核心间任务管理器1445,其用作线程分派器,用于将执行线程分派到一个或更多个着色器核心1455A-1455N和分块单元1458,以加速用于基于图块的渲染的分块操作,其中在图像空间中细分场

景的渲染操作,例如,以利用场景内的局部空间相干性或优化内部高速缓存的使用。

[0169] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,可以在集成电路14A和/或14B中使用推理和/或训练逻辑615,用于至少部分地基于使用在此描述的神经网络训练操作、神经网络功能和/或架构或神经网络用例计算的权重参数来推理或预测操作。

[0170] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0171] 图15A-15B示出了根据本文描述的实施例的附加示例性图形处理器逻辑。在至少一个实施例中,图15A示出了可以包括在图13的图形处理器1310内的图形核心1500,并且在至少一个实施例中,其可以是如图14B所示的统一着色器核心1455A-1455N。图15B示出了在至少一个实施例中的适用于在多芯片模块上部署的高度并行的通用图形处理单元1530。

[0172] 在至少一个实施例中,图形核心1500包括共享指令高速缓存1502、纹理单元1518和高速缓存/共享存储器1520,它们对于图形核心1500内的执行资源是通用的。在至少一个实施例中,图形核心1500可包括多个切片1501A-1501N或每个核心的分区,并且图形处理器可包括图形核心1500的多个实例。切片1501A-1501N可包括支持逻辑,所述逻辑包括本地指令高速缓存1504A-1504N、线程调度器1506A-1506N、线程分派器1508A-1508N和一组寄存器1510A-1510N。在至少一个实施例中,切片1501A-1501N可以包括一组附加功能单元 (AFU 1512A-1512N)、浮点单元 (FPU 1514A-1514N)、整数算术逻辑单元 (ALU 1516A-1516N)、地址计算单元 (ACU 1513A-1513N)、双精度浮点单元 (DPFPU 1515A-1515N) 和矩阵处理单元 (MPU 1517A-1517N)。

[0173] 在至少一个实施例中,FPU 1514A-1514N可以执行单精度 (32位) 和半精度 (16位) 浮点运算,而DPFPU 1515A-1515N则执行双精度 (64位) 浮点运算点操作。在至少一个实施例中,ALU 1516A-1516N可以以8位、16位和32位精度执行可变精度整数运算,并且可以配置为混合精度运算。在至少一个实施例中,MPU 1517A-1517N还可被配置用于混合精度矩阵运算,包括半精度浮点运算和8位整数运算。在至少一个实施例中,MPU 1517-1517N可以执行各种矩阵运算以加速机器学习应用程序框架,包括使得能够支持加速的通用矩阵到矩阵乘法 (GEMM)。在至少一个实施例中,AFU 1512A-1512N可以执行浮点数或整数单元不支持的附加逻辑运算,包括三角运算 (例如,正弦,余弦等)。

[0174] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在图形核心1500中用于至少部分地基于使用在此描述的神经网络训练操作、神经网络功能和/或架构或神经网络使用情况计算的权重参数来推理或预测操作。

[0175] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0176] 图15B示出了在至少一个实施例中的通用处理单元 (GPGPU) 1530,其可以被配置为使得高度并行的计算操作能够由一组图形处理单元来执行。在至少一个实施例中,GPGPU 1530可以直接链接到GPGPU 1530的其他实例,以创建多GPU集群以提高用于深度神经网络

的训练速度。在至少一个实施例中, GPGPU 1530包括主机接口1532, 以实现与主机处理器的连接。在至少一个实施例中, 主机接口1532是PCI Express接口。在至少一个实施例中, 主机接口1532可以是厂商专用的通信接口或通信结构。在至少一个实施例中, GPGPU 1530接收主机处理器的命令, 并使用全局调度器1534, 以将与那些命令相关联的执行线程分配给一组计算集群1536A-1536H。在至少一个实施例中, 计算集群1536A-1536H共享高速缓存存储器1538。在至少一个实施例中, 高速缓存存储器1538可以用作计算集群1536A-1536H内的高速缓存存储器的更高级别的高速缓存。

[0177] 在至少一个实施例中, GPGPU 1530包括存储器1544A-1544B, 所述存储器1544A-1544B经由一组存储器控制器1542A-1542B与计算集群1536A-1536H耦合。在至少一个实施例中, 存储器1544A-1544B可以包括各种类型的存储器设备, 包括动态随机存取存储器(DRAM)或图形随机存取存储器, 例如同步图形随机存取存储器(SGRAM), 其包括图形双倍数据速率(GDDR)存储器。

[0178] 在至少一个实施例中, 计算集群1536A-1536H每个都包括一组图形核心, 例如图15A的图形核心1500, 所述图形核心可以包括多种类型的整数和浮点逻辑单元, 所述逻辑单元可以在计算机各种精度范围上执行计算操作, 包括适用于机器学习计算的精度。例如, 在至少一个实施例中, 每个计算集群1536A-1536H中的浮点单元的至少一个子集可以被配置为执行16位或32位浮点运算, 而浮点单元的不同子集可以配置为执行64位浮点运算。

[0179] 在至少一个实施例中, GPGPU 1530的多个实例可以被配置为用作计算集群。在至少一个实施例中, 计算集群1536A-1536H用于同步和数据交换的通信在实施例之间变化。在至少一个实施例中, GPGPU 1530的多个实例通过主机接口1532进行通信。在至少一个实施例中, GPGPU 1530包括I/O集线器1539, 所述集线器将GPGPU 1530与GPU链路1540耦合, 使得能够直接连接到GPGPU 1530的其他实例。在至少一个实施例中, GPU链路1540耦合到专用GPU到GPU桥, 所述桥使得GPGPU 1530的多个实例之间能够通信和同步。在至少一个实施例中, GPU链路1540与高速互连耦合, 以向其他GPGPU或并行处理器发送和接收数据。在至少一个实施例中, GPGPU 1530的多个实例位于单独的数据处理系统中, 并通过可通过主机接口1532访问的网络设备进行通信。在至少一个实施例中, GPU链路1540可被配置为使得能够连接到主机除主机接口1532之外或作为其替代的处理器。

[0180] 在至少一个实施例中, GPGPU 1530可以被配置为训练神经网络。在至少一个实施例中, 可以在推理平台内使用GPGPU 1530。在至少一个实施例中, 在其中使用GPGPU 1530进行推理的情况下, 相对于使用GPGPU 1530训练神经网络时, GPGPU 1530可以包括更少的计算集群1536A-1536H。在至少一个实施例中, 与存储器1544A-1544B相关联的存储器技术可以在推理和训练配置之间有所不同, 其中更高带宽的存储器技术专用于训练配置。在至少一个实施例中, GPGPU 1530的推理配置可以支持推理特定指令。例如, 在至少一个实施例中, 推理配置可以提供对一个或多个8位整数点积指令的支持, 该指令可以在部署的神经网络的推理操作期间使用。

[0181] 推理和/或训练逻辑615用于执行与一个或多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中, 推理和/或训练逻辑615可以在GPGPU 1530中用于至少部分地基于使用在此描述的神经网络训练操作、神经网络功能和/或架构或神经网络使用情况计算的权重参数来推理或预

测操作。

[0182] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0183] 图16示出了根据至少一个实施例的计算机系统1600的框图。在至少一个实施例中,计算机系统1600包括具有一个或更多个处理器1602的处理子系统1601和系统存储器1604,所述系统存储器1604经由可包括存储器集线器1605的互连路径通信。在至少一个实施例中,存储器集线器1605可以是芯片组部件内的单独部件,或者可以集成在一个或更多个处理器1602内。在至少一个实施例中,存储器集线器1605通过通信链路1606与I/O子系统1611耦合。在一个实施例中,I/O子系统1611包括I/O集线器1607,所述I/O集线器可以使计算机系统1600能够接收来自一个或更多个输入设备1608的输入。在至少一个实施例中,I/O集线器1607可以使显示控制器向一个或更多个显示设备1610A提供输出,所述显示控制器可以包括在一个或更多个处理器1602中。在至少一个实施例中,与I/O集线器1607耦合的一个或更多个显示设备1610A可以包括本地,内部或嵌入式显示设备。

[0184] 在至少一个实施例中,处理子系统1601包括经由总线或其他通信链路1613耦合到存储器集线器1605的一个或更多个并行处理器1612中。在至少一个实施例中,通信链路1613可以使用任何一种许多基于标准的通信链路技术或协议,例如但不限于PCI Express,或者可以是特定于供应商的通信接口或通信结构。在至少一个实施例中,一个或更多个并行处理器1612形成计算集中的并行或矢量处理系统,所述系统可以包括大量处理核心和/或处理集群,例如多集成核心(MIC)处理器。在至少一个实施例中,一个或更多个并行处理器1612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集线器1607耦合的一个或更多个显示设备1610A之一。在至少一个实施例中,一个或更多个并行处理器1612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更多个显示设备1610B。

[0185] 在至少一个实施例中,系统存储单元1614可以连接到I/O集线器1607,以提供用于计算机系统1600的存储机制。在至少一个实施例中,I/O交换机1616可以用于提供一个接口机制,以实现I/O集线器1607与其他组件之间的连接,例如可以集成到平台中的网络适配器1618和/或无线网络适配器1619,以及可以通过一个或更多个附加设备1620添加的各种其他设备。在至少一个实施例中,网络适配器1618可以是有以太网适配器或另一有线网络适配器。在至少一个实施例中,无线网络适配器1619可以包括Wi-Fi、蓝牙、近场通信(NFC)中的一个或更多个,或包括一个或更多个无线电设备的其他网络设备。

[0186] 在至少一个实施例中,计算机系统1600可以包括未明确示出的其他组件,所述其他组件包括USB或其他端口连接、光学存储驱动器、视频捕获设备等,所述其他组件也可以连接到I/O集线器1607。在至少一个实施例中,可以使用任何合适的协议(例如基于PCI(外围组件互连)的协议(例如PCI-Express)或其他总线或点对点通信接口和/或协议)来实现互连图16中各个组件的通信路径,例如NV-Link高速互连或互连协议。

[0187] 在至少一个实施例中,一个或更多个并行处理器1612包括为图形和视频处理而优化的电路,所述电路包括例如视频输出电路,并构成图形处理单元(GPU)。在至少一个实施例中,一个或更多个并行处理器1612包括为通用处理而优化的电路。在至少一个实施例中,计算机系统1600的组件可以与单个集成电路上的一个或更多个其他系统元件集成。例如,

在至少一个实施例中,并行处理器1612、存储器集线器1605、处理器1602和I/O集线器1607,可以被集成到片上系统(SoC)集成电路中。在至少一个实施例中,计算机系统1600的组件可以被集成到单个封装中,以形成系统级封装(SIP)配置。在至少一个实施例中,计算机系统1600的组件的至少一部分可以被集成到多芯片模块(MCM)中,所述多芯片模块可以与其他多芯片模块互连到模块化计算机系统中。

[0188] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在系统1600中用于至少部分地基于使用在此描述的神经网络训练操作、神经网络功能和/或架构或神经网络使用情况计算的权重参数来推理或预测操作。

[0189] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0190] 处理器

[0191] 图17A示出了根据至少一个实施例的并行处理器1700。在至少一个实施例中,并行处理器1700的各种组件可以使用一个或更多个集成电路设备来实现,例如可编程处理器、专用集成电路(ASIC)或现场可编程门阵列(FPGA)。在至少一个实施例中,所示的并行处理器1700是根据示例性实施例的图16所示的一个或更多个并行处理器1612的变体。

[0192] 在至少一个实施例中,并行处理器1700包括并行处理单元1702。在至少一个实施例中,并行处理单元1702包括I/O单元1704,其使得能够与其他设备进行通信,包括并行处理单元1702的其他实例。在至少一个实施例中,I/O单元1704可以直接连接到其他设备。在至少一个实施例中,I/O单元1704通过使用集线器或交换机接口(例如,存储器集线器1605)与其他设备连接。在至少一个实施例中,存储器集线器1605与I/O单元1704之间的连接形成通信链路1613。在至少一个实施例中,I/O单元1704与主机接口1706和存储器交叉开关1716连接,其中主机接口1706接收用于执行处理操作的命令,而存储器交叉开关1716接收用于执行存储器操作的命令。

[0193] 在至少一个实施例中,当主机接口1706经由I/O单元1704接收命令缓冲区时,主机接口1706可以引导工作操作以执行那些命令到前端1708。在至少一个实施例中,前端1708与调度器1710耦合,调度器1710配置成将命令或其他工作项分配给处理集群阵列1712。在至少一个实施例中,调度器1710确保在将任务分配给处理集群阵列1712之前,处理集群阵列1712被正确地配置并且处于有效状态。在至少一个实施例中,调度器1710通过在微控制器上执行的固件逻辑来实现。在至少一个实施例中,微控制器实现的调度器1710可配置成以粗粒度和细粒度执行复杂的调度和工作分配操作,从而实现对在处理阵列1712上执行的线程的快速抢占和上下文切换。在至少一个实施例中,主机软件可以证明用于通过多个图形处理门铃(doorbell)之一在处理阵列1712上进行调度的工作负载。在至少一个实施例中,工作负载然后可以由包括调度器1710的微控制器内的调度器1710逻辑在处理阵列1712上自动分配。

[0194] 在至少一个实施例中,处理集群阵列1712可以包括多达“N”个处理集群(例如,集群1714A、集群1714B到集群1714N)。在至少一个实施例中,处理集群阵列1712的每个集群1714A-1714N可以执行大量并发线程。在至少一个实施例中,调度器1710可以使用各种调度

和/或工作分配算法将工作分配给处理集群阵列1712的集群1714A-1714N,其可以根据每种程序或计算类型产生的工作负载而变化。在至少一个实施例中,调度可以由调度器1710动态地处理,或者可以在配置为由处理集群阵列1712执行的程序逻辑的编译期间部分地由编译器逻辑来辅助。在至少一个实施例中,可将处理集群阵列1712的不同的集群1714A-1714N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。

[0195] 在至少一个实施例中,处理集群阵列1712可以配置成执行各种类型的并行处理操作。在至少一个实施例中,处理集群阵列1712配置成执行通用并行计算操作。例如,在至少一个实施例中,处理集群阵列1712可以包括执行处理任务的逻辑,该处理任务包括对视频和/或音频数据的过滤,执行建模操作,包括物理操作以及执行数据转换。

[0196] 在至少一个实施例中,处理集群阵列1712配置成执行并行图形处理操作。在至少一个实施例中,处理集群阵列1712可以包括附加逻辑以支持这种图形处理操作的执行,包括但不限于执行纹理操作的纹理采样逻辑,以及镶嵌逻辑和其他顶点处理逻辑。在至少一个实施例中,处理集群阵列1712可以配置成执行与图形处理有关的着色器程序,例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。在至少一个实施例中,并行处理单元1702可以经由I/O单元1704从系统存储器传送数据以进行处理。在至少一个实施例中,在处理期间,可以在处理期间将传送的数据存储到片上存储器(例如,并行处理器存储器1722),然后将其写回到系统存储器。

[0197] 在至少一个实施例中,当并行处理单元1702用于执行图形处理时,调度器1710可以配置成将处理工作负载划分为近似相等大小的任务,以更好地将图形处理操作分配给处理集群阵列1712的多个集群1714A-1714N。在至少一个实施例中,处理集群阵列1712的部分可以配置成执行不同类型的处理。例如,在至少一个实施例中,第一部分可以配置成执行顶点着色和拓扑生成,第二部分可以配置成执行镶嵌和几何着色,并且第三部分可以配置成执行像素着色或其他屏幕空间操作,以生成用于显示的渲染图像。在至少一个实施例中,可以将由集群1714A-1714N中的一个或多个产生的中间数据存储在缓冲区中,以允许在集群1714A-1714N之间传输中间数据以进行进一步处理。

[0198] 在至少一个实施例中,处理集群阵列1712可以经由调度器1710接收要执行的处理任务,该调度器1710从前端1708接收定义处理任务的命令。在至少一个实施例中,处理任务可以包括要被处理的数据的索引,例如,表面(补丁)数据、原始数据、顶点数据和/或像素数据,以及状态参数和定义如何处理数据的命令(例如,要执行什么程序)。在至少一个实施例中,调度器1710可以配置成获取与任务相对应的索引,或者可以从前端1708接收索引。在至少一个实施例中,前端1708可以配置成确保在启动由传入命令缓冲区(例如,批缓冲区(batch-buffer)、推送缓冲区等)指定的工作负载之前,处理集群阵列1712配置成有效状态。

[0199] 在至少一个实施例中,并行处理单元1702的一个或多个实例中的每一个可以与并行处理器存储器1722耦合。在至少一个实施例中,可以经由存储器交叉开关1716访问并行处理器存储器1722,所述存储器交叉开关1716可以接收来自处理集群阵列1712以及I/O单元1704的存储器请求。在至少一个实施例中,存储器交叉开关1716可以经由存储器接口1718访问并行处理器存储器1722。在至少一个实施例中,存储器接口1718可以包括多个分区单元(例如,分区单元1720A、分区单元1720B到分区单元1720N),其可各自耦合至并行处

理器存储器1722的一部分(例如,存储器单元)。在至少一个实施例中,多个分区单元1720A-1720N为配置为等于存储器单元的数量,使得第一分区单元1720A具有对应的第一存储器单元1724A,第二分区单元1720B具有对应的存储器单元1724B,第N分区单元1720N具有对应的第N存储器单元1724N。在至少一个实施例中,分区单元1720A-1720N的数量可以不等于存储器设备的数量。

[0200] 在至少一个实施例中,存储器单元1724A-1724N可以包括各种类型的存储器设备,包括动态随机存取存储器(DRAM)或图形随机存取存储器,例如同步图形随机存取存储器(SGRAM),包括图形双倍数据速率(GDDR)存储器。在至少一个实施例中,存储器单元1724A-1724N还可包括3D堆叠存储器,包括但不限于高带宽存储器(HBM)。在至少一个实施例中,可以跨存储器单元1724A-1724N来存储诸如帧缓冲区或纹理映射的渲染目标,从而允许分区单元1720A-1720N并行地写入每个渲染目标的部分,以有效地使用并行处理器存储器1722的可用带宽。在至少一个实施例中,可以排除并行处理器存储器1722的本地实例,以有利于利用系统存储器与本地高速缓存存储器结合的统一存储器设计。

[0201] 在至少一个实施例中,处理集群阵列1712的集群1714A-1714N中的任何一个都可以处理将被写入并行处理器存储器1722内的任何存储器单元1724A-1724N中的数据。在至少一个实施例中,存储器交叉开关1716可以配置为将每个集群1714A-1714N的输出传输到任何分区单元1720A-1720N或另一个集群1714A-1714N,集群1714A-1714N可以对输出执行其他处理操作。在至少一个实施例中,每个集群1714A-1714N可以通过存储器交叉开关1716与存储器接口1718通信,以从各种外部存储设备读取或写入各种外部存储设备。在至少一个实施例中,存储器交叉开关1716具有到存储器接口1718的连接以及与I/O单元1704通信,以及到并行处理器存储器1722的本地实例的连接,从而使不同处理集群1714A-1714N内的处理单元与系统存储器或不是并行处理单元1702本地的其他存储器进行通信。在至少一个实施例中,存储器交叉开关1716可以使用虚拟通道来分离集群1714A-1714N和分区单元1720A-1720N之间的业务流。

[0202] 在至少一个实施例中,可以在单个插入卡上提供并行处理单元1702的多个实例,或者可以将多个插入卡互连。在至少一个实施例中,并行处理单元1702的不同实例可以配置成相互操作,即使不同实例具有不同数量的处理核心,不同数量的本地并行处理器存储器和/或其他配置差异。例如,在至少一个实施例中,并行处理单元1702的一些实例可以包括相对于其他实例而言更高精度的浮点单元。在至少一个实施例中,结合并行处理单元1702或并行处理器1700的一个或更多个实例的系统可以以各种配置和形式因素来实现,包括但不限于台式机、膝上型计算机或手持式个人计算机、服务器、工作站、游戏机和/或嵌入式系统。

[0203] 图17B是根据至少一个实施例的分区单元1720的框图。在至少一个实施例中,分区单元1720是图17A的分区单元1720A-1720N之一的实例。在至少一个实施例中,分区单元1720包括L2高速缓存1721、帧缓冲区接口1725和光栅操作单元(“ROP”)1726。L2高速缓存1721是读/写高速缓存,其配置成执行从存储器交叉开关1716和ROP 1726接收的加载和存储操作。在至少一个实施例中,L2高速缓存1721将读取未命中和紧急回写请求输出到帧缓冲区接口1725以进行处理。在至少一个实施例中,还可以经由帧缓冲区接口1725将更新发送到帧缓冲区以进行处理。在至少一个实施例中,帧缓冲区接口1725与并行处理器存储器

中的存储器单元 (诸如图17A的存储器单元1724A-1724N (例如,在并行处理器存储器1722内))之一相互作用。

[0204] 在至少一个实施例中,ROP 1726是一种处理单元,其执行光栅操作,诸如模版、z测试、混合等。在至少一个实施例中,ROP 1726然后输出存储在图形存储器中的处理后的图形数据。在至少一个实施例中,ROP 1726包括压缩逻辑以压缩被写入存储器的深度或颜色数据并解压缩从存储器读取的深度或颜色数据。在至少一个实施例中,压缩逻辑可以是利用多种压缩算法中的一种或更多种的无损压缩逻辑。ROP 1726执行的压缩逻辑可以基于要压缩的数据的统计特性而变化。例如,在至少一个实施例中,基于每图块基础上的深度和颜色数据执行增量颜色压缩。

[0205] 在至少一个实施例中,ROP 1726包括在每个处理集群内 (例如,图17A的集群1714A-1714N),而不是在分区单元1720内。在至少一个实施例中,通过存储器交叉开关1716而不是像素片段数据传输对像素数据的读取和写入请求。在至少一个实施例中,经处理的图形数据可以在显示设备上 (诸如图16的一个或更多个显示设备1610之一) 显示,由处理器1602路由以供进一步处理,或者由图17A的并行处理器1700内的处理实体之一路由以供进一步处理。

[0206] 图17C是根据至少一个实施例的并行处理单元内的处理集群1714的框图。在至少一个实施例中,处理集群是图17A的处理集群1714A-1714N之一的实例。在至少一个实施例中,处理集群1714中的一个或更多个可以配置成并行执行许多线程,其中“线程”是指在特定的一组输入数据上执行的特定程序的实例。在至少一个实施例中,单指令多数据 (SIMD) 指令发布技术用于支持大量线程的并行执行而无需提供多个独立的指令单元。在至少一个实施例中,使用单指令多线程 (SIMT) 技术来支持并行执行大量一般同步的线程,这使用了公共指令单元,该公共指令单元配置成向每个处理集群内的一组处理引擎发出指令。

[0207] 在至少一个实施例中,可以通过将处理任务分配给SIMT并行处理器的管线管理器1732来控制处理集群1714的操作。在至少一个实施例中,管线管理器1732从图17A的调度器1710接收指令,通过图形多处理器1734和/或纹理单元1736管理这些指令的执行。在至少一个实施例中,图形多处理器1734是SIMT并行处理器的示例性实例。然而,在至少一个实施例中,处理集群1714内可以包括不同架构的各种类型的SIMT并行处理器。在至少一个实施例中,在处理集群1714内可以包括图形多处理器1734的一个或更多个实例。在至少一个实施例中,图形多处理器1734可以处理数据,并且数据交叉开关1740可以用于将处理后的数据分发到多个可能的目的 (包括其他着色器单元) 地之一。在至少一个实施例中,管线管理器1732可以通过指定要经由数据交叉开关1740分配的处理后的数据的目的地来促进处理后的数据的分配。

[0208] 在至少一个实施例中,处理集群1714内的每个图形多处理器1734可以包括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)。在至少一个实施例中,可以以管线方式配置功能执行逻辑,其中可以在先前的指令完成之前发出新的指令。在至少一个实施例中,功能执行逻辑支持多种操作,包括整数和浮点算术、比较操作、布尔运算、移位和各种代数函数的计算。在至少一个实施例中,可以利用相同的功能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。

[0209] 在至少一个实施例中,传送到处理集群1714的指令构成线程。在至少一个实施例

中,跨一组并行处理引擎执行的一组线程是线程组。在至少一个实施例中,线程组在不同的输入数据上执行程序。在至少一个实施例中,线程组内的每个线程可被分配给图形多处理器1734内的不同处理引擎。在至少一个实施例中,线程组可包括比图形多处理器1734内的多个处理引擎更少的线程。在至少一个实施例中,当线程组包括的线程数少于处理引擎的数量时,一个或更多个处理引擎在正在处理该线程组的循环期间可能是空闲的。在至少一个实施例中,线程组还可以包括比图形多处理器1734内的多个处理引擎更多的线程。在至少一个实施例中,当线程组包括比图形多处理器1734内的处理引擎更多的线程时,可以在连续的时钟周期内执行处理。在至少一个实施例中,可以在图形多处理器1734上同时执行多个线程组。

[0210] 在至少一个实施例中,图形多处理器1734包括内部高速缓存存储器,以执行加载和存储操作。在至少一个实施例中,图形多处理器1734可以放弃内部高速缓存并使用处理集群1714内的高速缓存存储器(例如,L1高速缓存1748)。在至少一个实施例中,每个图形多处理器1734还可以访问分区单元(例如,图17A的分区单元1720A-1720N)内的L2高速缓存,这些分区单元在所有处理集群1714之间共享并且可以用于在线程之间传输数据。在至少一个实施例中,图形多处理器1734还可以访问片外全局存储器,其可以包括本地并行处理器存储器和/或系统存储器中的一个或更多个。在至少一个实施例中,并行处理单元1702外部的任何存储器都可以用作全局存储器。在至少一个实施例中,处理集群1714包括图形多处理器1734的多个实例,它们可以共享可以存储在L1高速缓存1748中的公共指令和数据。

[0211] 在至少一个实施例中,每个处理集群1714可以包括配置成将虚拟地址映射为物理地址的存储器管理单元(“MMU”)1745。在至少一个实施例中,MMU 1745的一个或更多个实例可以驻留在图17A的存储器接口1718内。在至少一个实施例中,MMU 1745包括一组页表条目(PTE),其用于将虚拟地址映射到图块的物理地址以及可选地映射到高速缓存行索引。在至少一个实施例中,MMU 1745可以包括地址转换后备缓冲区(TLB)或可以驻留在图形多处理器1734或L1高速缓存或处理集群1714内的高速缓存。在至少一个实施例中,处理物理地址以分配表面数据访问局部性,以便在分区单元之间进行有效的请求交织。在至少一个实施例中,高速缓存行索引可以用于确定对高速缓存行的请求是命中还是未命中。

[0212] 在至少一个实施例中,可以配置处理集群1714,使得每个图形多处理器1734耦合到纹理单元1736,以执行纹理映射操作,所述操作确定纹理样本位置、读取纹理数据以及过滤纹理数据。在至少一个实施例中,根据需从内部纹理L1高速缓存(未示出)或从图形多处理器1734内的L1高速缓存中读取纹理数据,并从L2高速缓存、本地并行处理器存储器或系统存储器中获取纹理数据。在至少一个实施例中,每个图形多处理器1734将处理后的任务输出到数据交叉开关1740,以将处理后的任务提供给另一处理集群1714以进行进一步处理或将处理后的任务存储在L2高速缓存、本地并行处理器存储器、或经由存储器交叉开关1716的系统存储器中。在至少一个实施例中,preROP 1742(光栅前操作单元)配置成从图形多处理器1734接收数据,将数据引导至ROP单元,该ROP单元可以与本文所述的分区单元(例如,图17A的分区单元1720A-1720N)一起定位。在至少一个实施例中,PreROP 1742单元可以执行用于颜色混合的优化、组织像素颜色数据以及执行地址转换。

[0213] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例

中,推理和/或训练逻辑615可在图形处理集群1714中使用,用于至少部分地基于使用本文描述的神经网络训练操作、神经网络功能和/或架构或神经网络使用情况计算的权重参数来推理或预测操作。

[0214] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0215] 图17D示出了根据至少一个实施例的图形多处理器1734。在至少一个实施例中,图形多处理器1734与处理集群1714的管线管理器1732耦合。在至少一个实施例中,图形多处理器1734具有执行管线,该执行管线包括但不限于指令高速缓存1752、指令单元1754、地址映射单元1756、寄存器文件1758、一个或更多个通用图形处理单元(GPGPU)核心1762和一个或更多个加载/存储单元1766。GPGPU核心1762和加载/存储单元1766通过存储器和高速缓存互连1768与高速缓存存储器1772和共享存储器1770耦合。

[0216] 在至少一个实施例中,指令高速缓存1752从管线管理器1732接收要执行的指令流。在至少一个实施例中,将指令高速缓存在指令高速缓存1752中并将其分派以供指令单元1754执行。在一个实施例中,指令单元1754可以分派指令作为线程组(例如,线程束),将每个线程组分配给GPGPU核心1762内的不同执行单元。在至少一个实施例中,指令可以通过指定统一地址空间内的地址来访问任何本地、共享或全局地址空间。在至少一个实施例中,地址映射单元1756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1766访问的不同的存储器地址。

[0217] 在至少一个实施例中,寄存器文件1758为图形多处理器1734的功能单元提供了一组寄存器。在至少一个实施例中,寄存器文件1758为连接到图形多处理器1734的功能单元(例如,GPGPU核心1762、加载/存储单元1766)的数据路径的操作数提供了临时存储。在至少一个实施例中,在每个功能单元之间划分寄存器文件1758,使得为每个功能单元分配寄存器文件1758的专用部分。在至少一个实施例中,寄存器文件1758在图形多处理器1734正在执行的不同线程束之间划分。

[0218] 在至少一个实施例中,GPGPU核心1762可以各自包括用于执行图形多处理器1734的指令的浮点单元(FPU)和/或整数算术逻辑单元(ALU)。GPGPU核心1762在架构上可以相似或架构可能有所不同。在至少一个实施例中,GPGPU核心1762的第一部分包括单精度FPU和整数ALU,而GPGPU核心的第二部分包括双精度FPU。在至少一个实施例中,FPU可以实现用于浮点算法的IEEE 754-2008标准或启用可变精度浮点算法。在至少一个实施例中,图形多处理器1734可以另外包括一个或更多个固定功能或特殊功能单元,以执行特定功能,诸如复制矩形或像素混合操作。在至少一个实施例中,GPGPU核心中的一个或更多个还可以包括固定或特殊功能逻辑。

[0219] 在至少一个实施例中,GPGPU核心1762包括能够对多组数据执行单个指令的SIMD逻辑。在一个实施例中,GPGPU核心1762可以物理地执行SIMD4、SIMD8和SIMD16指令,并且在逻辑上执行SIMD1、SIMD2和SIMD32指令。在至少一个实施例中,用于GPGPU核心的SIMD指令可以在编译时由着色器编译器生成,或者在执行针对单程序多数据(SPMD)或SIMT架构编写和编译的程序时自动生成。在至少一个实施例中,可以通过单个SIMD指令来执行为SIMT执行模型配置的程序的多个线程。例如,在至少一个实施例中,可以通过单个SIMD8逻辑单元并行执行执行相同或相似操作的八个SIMT线程。

[0220] 在至少一个实施例中,存储器和高速缓存互连1768是将图形多处理器1734的每个功能单元连接到寄存器文件1758和共享存储器1770的互连网络。在至少一个实施例中,存储器和高速缓存互连1768是交叉开关互连,其允许加载/存储单元1766在共享存储器1770和寄存器文件1758之间实现加载和存储操作。在至少一个实施例中,寄存器文件1758可以以与GPGPU核心1762相同的频率操作,从而在GPGPU核心1762和寄存器文件1758之间进行数据传输的延迟非常低。在至少一个实施例中,共享存储器1770可以用于启用在图形多处理器1734内的功能单元上执行的线程之间的通信。在至少一个实施例中,高速缓存存储器1772可以用作例如数据高速缓存,以高速缓存在功能单元和纹理单元1736之间通信的纹理数据。在至少一个实施例中,共享存储器1770也可以用作程序管理的高速缓存。在至少一个实施例中,除了存储在高速缓存存储器1772中的自动高速缓存的数据之外,在GPGPU核心1762上执行的线程还可以以编程方式将数据存储在共享存储器中。

[0221] 在至少一个实施例中,如本文所述的并行处理器或GPGPU通信地耦合到主机/处理器核心,以加速图形操作、机器学习操作、图案分析操作以及各种通用GPU (GPGPU) 功能。在至少一个实施例中,GPU可以通过总线或其他互连(例如,诸如PCIe或NVLink的高速互连)通信地耦合到主机处理器/核心。在至少一个实施例中,GPU可以与核心集成在同一封装或芯片上,并通过内部处理器总线/互连(即,在封装或芯片的内部)通信地耦合到核心。在至少一个实施例中,不管GPU连接的方式如何,处理器核心可以以工作描述符中包含的命令/指令序列的形式向GPU分配工作。在至少一个实施例中,该GPU然后使用专用电路/逻辑来有效地处理这些命令/指令。

[0222] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在图形多处理器1734中使用,用于至少部分地基于使用本文描述的神经网络训练操作、神经网络功能和/或架构或神经网络使用情况计算的权重参数来推理或预测操作。

[0223] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0224] 图18示出了根据至少一个实施例的多GPU计算系统1800。在至少一个实施例中,多GPU计算系统1800可包括处理器1802,其经由主机接口交换机1804耦合到多个通用图形处理单元(GPGPU) 1806A-D。在至少一个实施例中,主机接口交换机1804是将处理器1802耦合到PCI快速总线的PCI快速交换机设备,处理器1802可通过PCI快速总线与GPGPU1806A-D通信。GPGPU1806A-D可经由高速点对点GPU到GPU链路1816的集合互连。在至少一个实施例中,GPU到GPU链路1816经由专用GPU链路连接到GPGPU1806A-D中的每一者。在至少一个实施例中,P2PGPU链路1816实现GPGPU1806A-D中的每一个之间的直接通信,而不需要在处理器1802连接到的主机接口总线1804上的通信。在至少一个实施例中,通过指向P2PGPU链路1816的GPU到GPU流量,主机接口总线1804保持可用于系统存储器访问或例如经由一个或更多个网络设备与多GPU计算系统1800的其他实例通信。虽然在至少一个实施例中,GPGPU1806A-D经由主机接口交换机1804连接到处理器1802,但是在至少一个实施例中,处理器1802包括对P2PGPU链路1816的直接支持,并且可以直接连接到GPGPU1806A-D。

[0225] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练

操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在多GPU计算系统1800中使用,用于至少部分地基于使用本文描述的神经网络训练操作、神经网络功能和/或架构或神经网络使用情况计算的权重参数来推理或预测操作。

[0226] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0227] 图19是根据至少一个实施例的图形处理器1900的框图。在至少一个实施例中,图形处理器1900包括环形互连1902、管线前端1904、媒体引擎1937和图形核心1980A-1980N。在至少一个实施例中,环形互连1902将图形处理器1900耦合到其他处理单元,所述处理单元包括其他图形处理器或一个或更多个通用处理器核心。在至少一个实施例中,图形处理器1900是集成在多核心处理系统内的许多处理器之一。

[0228] 在至少一个实施例中,图形处理器1900经由环形互连1902接收多批命令。在至少一个实施例中,输入的命令由管线前端1904中的命令流转化器(streamer)1903解释。在至少一个实施例中,图形处理器1900包括可扩展执行逻辑,用于经由图形核心1980A-1980N执行3D几何处理和媒体处理。在至少一个实施例中,对于3D几何处理命令,命令流转化器1903将命令提供给几何管线1936。在至少一个实施例中,对于至少一些媒体处理命令,命令流转化器1903将命令提供给视频前端1934,该视频前端与媒体引擎1937耦合。在至少一个实施例中,媒体引擎1937包括用于视频和图像后处理的视频质量引擎(VQE)1930,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)1933引擎。在至少一个实施例中,几何管线1936和媒体引擎1937各自生成用于由至少一个图形核心1980A提供的线程执行资源的执行线程。

[0229] 在至少一个实施例中,图形处理器1900包括具有(featuring)模块核心1980A-1980N(有时被称为核心切片)的可扩展线程执行资源,每个图形核心具有多个子核心1950A-1950N,1960A-1960N(有时称为核心子切片)。在至少一个实施例中,图形处理器1900可以具有任意数量的图形核心1980A到1980N。在至少一个实施例中,图形处理器1900包括具有至少第一子核心1950A和第二子核心1960A的图形核心1980A。在至少一个实施例中,图形处理器1900是具有单个子核心(例如1950A)的低功率处理器。在至少一个实施例中,图形处理器1900包括多个图形核心1980A-1980N,每个图形核心包括一组第一子核心1950A-1950N和一组第二子核心1960A-1960N。在至少一个实施例中,第一子核心1950A-1950N中的每个子核心至少包括第一组执行单元1952A-1952N和媒体/纹理采样器1954A-1954N。在至少一个实施例中,第二子核心1960A-1960N中的每个子核心至少包括第二组执行单元1962A-1962N和采样器1964A-1964N。在至少一个实施例中,每个子核心1950A-1950N,1960A-1960N共享一组共享资源1970A-1970N。在至少一个实施例中,共享资源包括共享高速缓存存储器和像素操作逻辑。

[0230] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以在图形处理器1900中使用,用于至少部分地基于使用本文描述的神经网络训练操作、神经网络功能和/或架构或神经网络使用情况计算的权重参数来推理或预测操作。

[0231] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0232] 图20是根据至少一个实施例的说明用于处理器2000的微架构的框图,该处理器2000可以包括用于执行指令的逻辑电路。在至少一个实施例中,处理器2000可以执行指令,包括x86指令、ARM指令、用于专用集成电路(ASIC)的专用指令等。在至少一个实施例中,处理器2000可以包括用于存储封装数据的寄存器,例如作为加利福尼亚州圣克拉拉市英特尔公司采用MMX技术启用的微处理器中的64位宽MMX™寄存器。在至少一个实施例中,整数和浮点数形式可用的MMX寄存器可以与封装的数据元素一起运行,所述封装的数据元素伴随单指令多数据(“SIMD”)和流式SIMD扩展(“SSE”)指令。在至少一个实施例中,与SSE2、SSE3、SSE4、AVX或更高版本(一般称为“SSEx”)技术有关的128位宽XMM寄存器可以保存此类封装数据操作数。在至少一个实施例中,处理器2000可以执行指令以加速机器学习或深度学习算法、训练或推理。

[0233] 在至少一个实施例中,处理器2000包括有序前端(“前端”)2001,以提取要执行的指令并准备稍后在处理器管线中使用的指令。在至少一个实施例中,前端2001可以包括几个单元。在至少一个实施例中,指令预取器2026从存储器中获取指令并将指令提供给指令解码器2028,指令解码器2028又对指令进行解码或解释。例如,在至少一个实施例中,指令解码器2028将接收到的指令解码为机器可执行的所谓的“微指令”或“微操作”(也称为“微操作”或“微指令”)的一个或更多个操作。在至少一个实施例中,指令解码器2028将指令解析为操作码以及相应的数据和控制字段,其可以由微架构用来使用以根据至少一个实施例来执行操作。在至少一个实施例中,跟踪高速缓存2030可以将解码的微指令组装成微指令队列2034中的程序排序的序列或追踪以供执行。在至少一个实施例中,当追踪高速缓存2030遇到复杂指令时,微码ROM 2032提供完成操作所需的微指令。

[0234] 在至少一个实施例中,可以将一些指令转换成单个微操作,而另一些指令则需要几个微操作来完成全部操作。在至少一个实施例中,如果需要多于四个的微指令来完成一条指令,则指令解码器2028可以访问微码ROM 2032以执行指令。在至少一个实施例中,可以将指令解码为少量的微指令以在指令解码器2028处进行处理。在至少一个实施例中,如果需要多个微指令完成操作,则可以将指令存储在微码ROM 2032中。在至少一个实施例中,追踪高速缓存器2030参考入口点可编程逻辑阵列(“PLA”)以确定正确的微指令指针,用于根据至少一个实施例从微码ROM 2032读取微码序列以完成一个或更多个指令。在至少一个实施例中,在微码ROM2032完成对指令的微操作排序之后,机器的前端2001可以恢复从追踪高速缓存2030获取微操作。

[0235] 在至少一个实施例中,乱序执行引擎(“乱序引擎”)2003可以准备用于执行的指令。在至少一个实施例中,乱序执行逻辑具有多个缓冲区,以使指令流平滑并重新排序,以在指令沿管线下落并被调度执行时优化性能。在至少一个实施例中,乱序执行引擎2003包括但不限于分配器/寄存器重命名器2040、存储器微指令队列2042、整数/浮点微指令队列2044、存储器调度器2046、快速调度器2002、慢速/通用浮点调度器(“慢速/通用FP调度器”)2004和简单浮点调度器(“简单FP调度器”)2006。在至少一个实施例中,快速调度器2002、慢速/通用浮点调度器2004和简单浮点调度器2006也统称为“微指令调度器2002、2004、2006”。在至少一个实施例中,分配器/寄存器重命名器2040分配每个微指令按序列执行所

需要的机器缓冲区和资源。在至少一个实施例中,分配器/寄存器重命名器2040将逻辑寄存器重命名为寄存器文件中的条目。在至少一个实施例中,分配器/寄存器重命名器2040还为两个微指令队列之一中的每个微指令分配条目,存储器微指令队列2042用于存储器操作和整数/浮点微指令队列2044用于非存储器操作,在存储器调度器2046和微指令调度器2002、2004、2006的前面。在至少一个实施例中,微指令调度器2002、2004、2006基于它们的从属输入寄存器操作数源的就绪性和需要完成的执行资源微指令的可用性来确定何时准备好执行微指令。在至少一个实施例中,至少一个实施例的快速调度器2002可以在主时钟周期的每个一半上调度,而慢速/通用浮点调度器2004和简单浮点调度器2006可以在每个主处理器时钟周期调度一次。在至少一个实施例中,微指令调度器2002、2004、2006对调度端口进行仲裁,以调度用于执行的微指令。

[0236] 在至少一个实施例中,执行块2011包括但不限于整数寄存器文件/支路网络2008、浮点寄存器文件/支路网络(“FP寄存器文件/支路网络”)2010、地址生成单元(“AGU”)2012和2014、快速算术逻辑单元(“快速ALU”)2016和2018、慢速算术逻辑单元(“慢速ALU”)2020、浮点ALU(“FP”)2022和浮点移动单元(“FP移动”)2024。在至少一个实施例中,整数寄存器文件/支路网络2008和浮点寄存器文件/旁路网络2010在本文中也称为“寄存器文件2008、2010”。在至少一个实施例中,AGU 2012和2014、快速ALU 2016和2018、慢速ALU 2020、浮点ALU 2022和浮点移动单元2024在本文中也称为“执行单元2012、2014、2016、2018、2020、2022和2024”。在至少一个实施例中,执行块2011可以包括但不限于任意数量(包括零)和类型的寄存器文件、支路网络、地址生成单元和执行单元(以任何组合)。

[0237] 在至少一个实施例中,寄存器文件2008、2010可以布置在微指令调度器2002、2004、2006与执行单元2012、2014、2016、2018、2020、2022和2024之间。在至少一个实施例中,整数寄存器文件/支路网络2008执行整数运算。在至少一个实施例中,浮点寄存器文件/支路网络2010执行浮点操作。在至少一个实施例中,寄存器文件2008、2010中的每一个可以包括但不限于支路网络,该支路网络可以绕过或转发尚未写入寄存器文件中的刚刚完成的结果到新的从属对象。在至少一个实施例中,寄存器文件2008、2010可以彼此通信数据。在至少一个实施例中,整数寄存器文件/支路网络2008可以包括但不限于两个单独的寄存器文件、一个寄存器文件用于低阶32位数据,第二寄存器文件用于高阶32位数据。在至少一个实施例中,浮点寄存器文件/支路网络2010可以包括但不限于128位宽的条目,因为浮点指令通常具有宽度为64至128位的操作数。

[0238] 在至少一个实施例中,执行单元2012、2014、2016、2018、2020、2022、2024可以执行指令。在至少一个实施例中,寄存器文件2008、2010存储微指令需要执行的整数和浮点数据操作数值。在至少一个实施例中,处理器2000可以包括但不限于任何数量的执行单元2012、2014、2016、2018、2020、2022、2024及其组合。在至少一个实施例中,浮点ALU 2022和浮点移动单元2024,可以执行浮点、MMX、SIMD、AVX和SSE或其他操作,包括专门的机器学习指令。在至少一个实施例中,浮点ALU 2022可以包括但不限于64位乘64位浮点除法器,以执行除法、平方根和余数微操作。在至少一个实施例中,可以用浮点硬件来处理涉及浮点值的指令。在至少一个实施例中,可以将ALU操作传递给快速ALU 2016、2018。在至少一个实施例中,快速ALU 2016、2018可以以半个时钟周期的有效延迟执行快速操作。在至少一个实施例中,大多数复杂的整数运算进入慢速ALU 2020,因为慢速ALU 2020可以包括但不限于用于长延迟类

型操作的整数执行硬件,例如乘法器、移位、标志逻辑和分支处理。在至少一个实施例中,存储器加载/存储操作可以由AGU 2012、2014执行。在至少一个实施例中,快速ALU 2016、快速ALU 2018和慢速ALU 2020可以对64位数据操作数执行整数运算。在至少一个实施例中,可以实现快速ALU 2016、快速ALU 2018和慢速ALU 2020以支持包括十六、三十二、128、256等的各种数据位大小。在至少一个实施例中,浮点ALU 2022和浮点移动单元2024可以实现为支持具有各种宽度的位的一定范围的操作数。在至少一个实施例中,浮点ALU 2022和浮点移动单元2024可以结合SIMD和多媒体指令对128位宽打包数据操作数进行操作。

[0239] 在至少一个实施例中,微指令调度器2002、2004、2006在父加载完成执行之前调度从属操作。在至少一个实施例中,由于可以在处理器2000中推测性地调度和执行微指令,处理器2000还可以包括用于处理存储器未命中的逻辑。在至少一个实施例中,如果数据高速缓存中的数据加载未命中,则可能存在正在运行的从属操作,其使调度器暂时没有正确的数据。在至少一个实施例中,一种重放机制追踪并重新执行使用不正确数据的指令。在至少一个实施例中,可能需要重放从属操作并且可以允许完成独立操作。在至少一个实施例中,处理器的至少一个实施例的调度器和重放机制也可以设计为捕获用于文本串比较操作的指令序列。

[0240] 在至少一个实施例中,术语“寄存器”可以指代可以用作识别操作数的指令的一部分的机载处理器存储位置。在至少一个实施例中,寄存器可以是那些可以从处理器外部使用的寄存器(从程序员的角度来看)。在至少一个实施例中,寄存器可能不限于特定类型的电路。相反,在至少一个实施例中,寄存器可以存储数据、提供数据并执行本文描述的功能。在至少一个实施例中,本文描述的寄存器可以通过处理器内的电路使用多种不同技术来实现,例如专用物理寄存器、使用寄存器重命名动态分配的物理寄存器、专用和动态分配的物理寄存器的组合等。在至少一个实施例中,整数寄存器存储32位整数数据。至少一个实施例的寄存器文件还包含八个用于封装数据的多媒体SIMD寄存器。

[0241] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615可以并入执行块2011以及示出的或未示出的其他存储器或寄存器中。例如,在至少一个实施例中,本文描述的推理和/或训练技术可以使用在执行块2011中示出的一个或更多个ALU。此外,权重参数可以存储在片上或片外存储器和/或寄存器(示出的或未示出的)中,该权重参数将执行块2011的ALU配置为执行一个或更多个机器学习算法、神经网络架构、用例或本文描述的推理技术。

[0242] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0243] 图21示出了根据至少一个实施例的深度学习应用程序处理器2100。在至少一个实施例中,深度学习应用程序处理器2100使用指令,如果由深度学习应用程序处理器2100执行,则指令使深度学习应用程序处理器2100执行贯穿本公开描述的一些或全部过程和技术。在至少一个实施例中,深度学习应用程序处理器2100是专用集成电路(ASIC)。在至少一个实施例中,应用程序处理器2100执行矩阵乘法运算或者“硬连线”到硬件中,作为执行一个或更多个指令或两者的结果。在至少一个实施例中,深度学习应用程序处理器2100包括但不限于处理集群2110(1)-2110(12)、芯片间链路(“ICL”)2120(1)-2120(12)、芯片间控制

器 (“ICC”) 2130(1) - 2130(2)、存储器控制器 (“Mem Ctrlr”) 2142(1) - 2142(4)、高带宽存储器物理层 (“HBM PHY”) 2144(1) - 2144(4)、管理控制器中央处理单元 (“管理控制器CPU”) 2150、串行外设接口、内部集成电路和通用输入/输出模块 (“SPI、I2C、GPIO”)、外围组件互连快速控制器和直接存储器访问块 (“PCIe控制器和DMA”) 2170、以及十六通道外围组件互连快速端口 (“PCI Express x 16”) 2180。

[0244] 在至少一个实施例中,处理集群2110可以执行深度学习操作,包括基于一种或更多种训练技术计算的权重参数的推理或预测操作,包括本文所述的那些技术。在至少一个实施例中,每个处理集群2110可以包括但不限于任何数量和类型的处理器。在至少一个实施例中,深度学习应用程序处理器2100可以包括任何数量和类型的处理集群2100。在至少一个实施例中,芯片间链路2120是双向的。在至少一个实施例中,芯片间链路2120和芯片间控制器2130使多个深度学习应用程序处理器2100能够交换信息,包括从执行一个或更多个神经网络中体现的一种或更多种机器学习算法而产生的激活信息。在至少一个实施例中,深度学习应用程序处理器2100可以包括任意数量(包括零)和类型的ICL 2120和ICC 2130。

[0245] 在至少一个实施例中,HBM2 2140提供总共32GB的存储器。HBM2 2140(i)与存储器控制器2142(i)和HBM PHY 2144(i)都相关联。在至少一个实施例中,任何数量的HBM2 2140可以提供任何类型和总量的高带宽存储器,并且可以与任何数量(包括零)和类型的存储器控制器2142和HBM PHY 2144相关联。在至少一个实施例中,可以用任何数量和类型的块替换SPI、I2C、GPIO 3360、PCIe控制器2160和DMA 2170和/或PCIe2180,以任何技术上可行的方式实现任何数量和类型的通信标准。

[0246] 推理和/或训练逻辑1015用于执行与一个或更多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,深度学习应用程序处理器2100用于训练机器学习模型(例如神经网络),以预测或推理提供给深度学习应用程序处理器2100的信息。在至少一个实施例中,深度学习应用程序处理器2100用于基于已经由另一处理器或系统或由深度学习应用程序处理器2100训练的经训练的机器学习模型(例如,神经网络)来推理或预测信息。在至少一个实施例中,处理器2100可以用于执行本文所述的一个或更多个神经网络用例。

[0247] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0248] 图22是根据至少一个实施例的神经形态处理器2200的框图。在至少一个实施例中,神经形态处理器2200可以从神经形态处理器2200外部的源接收一个或更多个输入。在至少一个实施例中,这些输入可以被传输到神经形态处理器2200内的一个或更多个神经元2202。在至少一个实施例中,可以使用包括一个或更多个算术逻辑单元(ALU)的电路或逻辑来实现神经元2202及其组件。在至少一个实施例中,神经形态处理器2200可以包括但不限于成千上万个神经元2202的实例,但是可以使用任何合适数量的神经元2202。在至少一个实施例中,神经元2202的每个实例可以包括神经元输入2204和神经元输出2206。在至少一个实施例中,神经元2202可以生成可以传输到神经元2202的其他实例的输入的输出。在至少一个实施例中,神经元输入2204和神经元输出2206可以经由突触2208互连。

[0249] 在至少一个实施例中,神经元2202和突触2208可以互连,使得神经形态处理器2200操作以处理或分析由神经形态处理器2200接收的信息。在至少一个实施例中,当通过

神经元输入2204接收到的输入超过阈值时,神经元2202可以发送输出脉冲(或“触发”或“峰值”)。在至少一个实施例中,神经元2202可以对在神经元输入2204处接收到的信号进行求和或积分。例如,在至少一个实施例中,神经元2202可以实现为有泄漏的积分-触发神经元,其中如果求和(称为“膜电位”)超过阈值,则神经元2202可以使用诸如sigmoid或阈值函数的传递函数来产生输出(或“触发”)。在至少一个实施例中,泄漏的积分-触发神经元可以将神经元输入2204处接收到的信号求和成膜电位,并且可以应用程序衰减因子(或泄漏)以减小膜电位。在至少一个实施例中,如果在神经元输入2204处接收到足够快以超过阈值的多个输入信号(即,在膜电势衰减得太低而不能触发之前),则泄漏的积分-触发神经元可能会触发。在至少一个实施例中,神经元2202可以使用接收输入、将输入积分到膜电位、并衰减膜电位的电路或逻辑来实现。在至少一个实施例中,可以对输入求平均,或者可以使用任何其他合适的传递函数。此外,在至少一个实施例中,神经元2202可以包括但不限于当将传递函数应用程序于神经元输入2204的结果超过阈值时在神经元输出2206处产生输出尖峰的比较器电路或逻辑。在至少一个实施例中,一旦神经元2202触发,它可以通过例如将膜电位复位为0或另一合适的默认值来忽略先前接收的输入信息。在至少一个实施例中,一旦膜电位被重置为0,则神经元2202可以在合适的时间段(或修复期)之后恢复正常操作。

[0250] 在至少一个实施例中,神经元2202可以通过突触2208互连。在至少一个实施例中,突触2208可以操作以将从第一神经元2202的输出的信号传输到第二神经元2202的输入。在至少一个实施例中,神经元2202可以在一个以上的突触2208实例上传输信息。在至少一个实施例中,神经元输出2206的一个或多个实例可以通过突触2208的实例连接到同一神经元2202中神经元输入2204的实例。在至少一个实施例中,相对于突触2208的那个实例,神经元2202的实例产生要在突触2208的实例上传输的输出可以被称为“突触前神经元”。在至少一个实施例中,相对于突触2208的实例,神经元2202的实例接收通过突触2208的实例传输的输入可以被称为“突触后神经元”。在至少一个实施例中,关于突触2208的各种实例,因为神经元2202的实例可以接收来自一个或多个突触2208实例的输入,并且还可以通过一个或多个突触2208实例传输输出,因此神经元2202的单个实例可以既是“突触前神经元”又是“突触后神经元”。

[0251] 神经元2202可以被组织成一层或更多层。神经元2202的每个实例可以具有一个神经元输出2206,该神经元输出2206可以通过一个或多个突触2208扇出到一个或多个神经元输入2204。在至少一个实施例中,第一层2210中的神经元2202的神经元输出2206可以连接到第二层2212中的神经元2202的神经元输入2204。在至少一个实施例中,层2210可以被称为“前馈层”。在至少一个实施例中,在第一层2210的实例中神经元2202的每个实例可以扇出到第二层2212中的神经元2202的每个实例。在至少一个实施例中,第一层2210可以被称为“完全连接的前馈层”。在至少一个实施例中,在第二层2212的每个实例中的神经元2202的每个实例扇出到少于在第三层2214中的神经元2202的所有实例。在至少一个实施例中,第二层2212可以被称为“稀疏连接的前馈层”。在至少一个实施例中,第二层2212中的神经元2202可以扇出到多个其他层中的神经元2202,包括扇出到(相同的)第二层2212中的神经元2202。在至少一个实施例中,第二层2212可以被称为“循环层”。在至少一个实施例中,神经形态处理器2200可以包括但不限于循环层和前馈层的任何合适的组合,包括但不限于稀疏连接的前馈层和完全连接的前馈层。

[0252] 在至少一个实施例中,神经形态处理器2200可以包括但不限于可重新配置的互连架构或专用硬连线互连,以将突触2208连接到神经元2202。在至少一个实施例中,神经形态处理器2200可以包括但不限于电路或逻辑,其根据神经网络拓扑结构和神经元扇入/扇出,允许根据需要分配突触给不同神经元2202。例如,在至少一个实施例中,可以使用互连结构(诸如片上网络)或通过专用连接将突触2208连接到神经元2202。在至少一个实施例中,可以使用电路或逻辑来实现突触互连及其组件。

[0253] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0254] 图23示出了根据至少一个实施例的处理系统。在至少一个实施例中,系统2300包括一个或多个处理器2302和一个或多个图形处理器2308,并且可以是单处理器台式机系统、多处理器工作站系统或具有大量处理器2302或处理器核心2307的服务器系统。在至少一个实施例中,系统2300是结合在片上系统(SoC)集成电路内的处理平台,以在移动、手持或嵌入式设备使用。

[0255] 在至少一个实施例中,系统2300可以包括或结合在基于服务器的游戏平台中,包括游戏和媒体控制台的游戏控制台、移动游戏控制台、手持游戏控制台或在线游戏控制台。在至少一个实施例中,系统2300是移动电话、智能电话、平板计算设备或移动互联网设备。在至少一个实施例中,处理系统2300还可包括与可穿戴设备耦合或集成在可穿戴设备中,例如智能手表可穿戴设备、智能眼镜设备、增强现实设备或虚拟现实设备。在至少一个实施例中,处理系统2300是电视或机顶盒设备,其具有一个或多个处理器2302以及由一个或多个图形处理器2308生成的图形界面。

[0256] 在至少一个实施例中,一个或多个处理器2302每个包括一个或多个处理器核心2307,以处理指令,该指令在被执行时执行针对系统和用户软件的操作。在至少一个实施例中,一个或多个处理器核心2307中的每一个被配置为处理特定指令集2309。在至少一个实施例中,指令集2309可以促进复杂指令集计算(CISC)、精简指令集计算(RISC),或通过超长指令字(VLIW)进行计算。在至少一个实施例中,处理器核心2307可以各自处理不同的指令集2309,该指令集可以包括有助于仿真其他指令集的指令。在至少一个实施例中,处理器核心2307还可以包括其他处理设备,例如数字信号处理器(DSP)。

[0257] 在至少一个实施例中,处理器2302包括高速缓存存储器2304。在至少一个实施例中,处理器2302可以具有单个内部高速缓存或多个级别的内部高速缓存。在至少一个实施例中,高速缓存存储器在处理器2302的各个组件之间共享。在至少一个实施例中,处理器2302还使用外部高速缓存(例如,三级(L3)高速缓存或最后一级高速缓存(LLC))(未示出),可以使用已知的高速缓存一致性技术在处理器核心2307之间共享该外部高速缓存。在至少一个实施例中,处理器2302中另外包括寄存器文件2306,处理器可以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)。在至少一个实施例中,寄存器文件2306可以包括通用寄存器或其他寄存器。

[0258] 在至少一个实施例中,一个或多个处理器2302与一个或多个接口总线2310耦合,以在处理器2302与系统2300中的其他组件之间传输通信信号,例如地址、数据或控制信号。在至少一个实施例中,接口总线2310在一个实施例中可以是处理器总线,例如直接媒体接口(DMI)总线的版本。在至少一个实施例中,接口总线2310不限于DMI总线,并且可以包括

一个或多个外围组件互连总线(例如,PCI,PCI Express)、存储器总线或其他类型的接口总线。在至少一个实施例中,处理器2302包括集成存储器控制器2316和平台控制器集线器2330。在至少一个实施例中,存储器控制器2316促进存储器设备与处理系统2300的其他组件之间的通信,而平台控制器集线器(PCH)2330通过本地I/O总线提供到输入/输出(I/O)设备的连接。

[0259] 在至少一个实施例中,存储器设备2320可以是动态随机存取存储器(DRAM)设备、静态随机存取存储器(SRAM)设备、闪存设备、相变存储设备或具有适当的性能以用作处理器存储器。在至少一个实施例中,存储设备2320可以用作处理系统2300的系统存储器,以存储数据2322和指令2321,以在一个或多个处理器2302执行应用程序或过程时使用。在至少一个实施例中,存储器控制器2316还与可选的外部图形处理器2312耦合,其可以与处理器2302中的一个或多个图形处理器2308通信以执行图形和媒体操作。在至少一个实施例中,显示设备2311可以连接至处理器2302。在至少一个实施例中,显示设备2311可以包括内部显示设备中的一个或多个,例如在移动电子设备或膝上型设备或通过显示器接口(例如显示端口(DisplayPort)等)连接的外部显示设备中。在至少一个实施例中,显示设备2311可以包括头戴式显示器(HMD),诸如用于虚拟现实(VR)应用或增强现实(AR)应用中的立体显示设备。

[0260] 在至少一个实施例中,平台控制器集线器2330使外围设备能够通过高速I/O总线连接到存储设备2320和处理器2302。在至少一个实施例中,I/O外围设备包括但不限于音频控制器2346、网络控制器2334、固件接口2328、无线收发器2326、触摸传感器2325、数据存储设备2324(例如,硬盘驱动器、闪存等)。在至少一个实施例中,数据存储设备2324可以经由存储接口(例如,SATA)或经由外围总线来连接,诸如外围组件互连总线(例如,PCI、PCIe)。在至少一个实施例中,触摸传感器2325可以包括触摸屏传感器、压力传感器或指纹传感器。在至少一个实施例中,无线收发器2326可以是Wi-Fi收发器、蓝牙收发器或移动网络收发器,诸如3G、4G或长期演进(LTE)收发器。在至少一个实施例中,固件接口2328使能与系统固件的通信,并且可以是例如统一可扩展固件接口(UEFI)。在至少一个实施例中,网络控制器2334可以启用到有线网络的网络连接。在至少一个实施例中,高性能网络控制器(未示出)与接口总线2310耦合。在至少一个实施例中,音频控制器2346是多通道高清晰度音频控制器。在至少一个实施例中,处理系统2300包括可选的传统(legacy)I/O控制器2340,用于将传统(例如,个人系统2(PS/2))设备耦合到系统。在至少一个实施例中,平台控制器集线器2330还可以连接到一个或多个通用串行总线(USB)控制器2342,该控制器连接输入设备,诸如键盘和鼠标2343组合、相机2344或其他USB输入设备。

[0261] 在至少一个实施例中,存储器控制器2316和平台控制器集线器2330的实例可以集成到离散的外部图形处理器中,例如外部图形处理器2312。在至少一个实施例中,平台控制器集线器2330和/或存储器控制器2316可以在一个或多个处理器2302的外部。例如,在至少一个实施例中,系统2300可以包括外部存储器控制器2316和平台控制器集线器2330,其可以配置成在与处理器2302通信的系统芯片组中的存储器控制器集线器和外围控制器集线器。

[0262] 推理和/或训练逻辑615用于执行与一个或多个实施例相关联的推理和/或训练操作。本文结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例

中,部分或全部推理和/或训练逻辑615可以结合到图形处理器2300中。例如,在至少一个实施例中,本文描述的推理和/或推理技术可以使用一个或更多个ALU,所述ALU体现在图形处理器2312中。此外,在至少一个实施例中,本文描述的推理和/或训练操作可以使用除图6A或图6B所示的逻辑之外的逻辑来完成。在至少一个实施例中,权重参数可以存储在片上或片外存储器和/或寄存器(示出或未示出)中,其配置图形处理器2300的ALU,以执行一种或更多种本文所述的机器学习算法、神经网络架构、用例或训练技术。

[0263] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0264] 图24是根据至少一个实施例的具有一个或更多个处理器核心2402A-2402N、集成存储器控制器2414和集成图形处理器2408的处理器2400的框图。在至少一个实施例中,处理器2400可以包含附加核心,多达并包括以虚线框表示的附加核心2402N。在至少一个实施例中,每个处理器核心2402A-2402N包括一个或更多个内部高速缓存单元2404A-2404N。在至少一个实施例中,每个处理器核心还可以访问一个或更多个共享高速缓存单元2406。

[0265] 在至少一个实施例中,内部高速缓存单元2404A-2404N和共享高速缓存单元2406表示处理器2400内的高速缓存存储器层次结构。在至少一个实施例中,高速缓存存储器单元2404A-2404N可以包括每个处理器核心内的至少一级指令和数据高速缓存以及共享中级高速缓存中的一级或更多级缓存,例如2级(L2)、3级(L3)、4级(L4)或其他级别的高速缓存,其中将外部存储器之前的最高级别的高速缓存归类为LLC。在至少一个实施例中,高速缓存一致性逻辑维持各种高速缓存单元2406和2404A-2404N之间的一致性。

[0266] 在至少一个实施例中,处理器2400还可包括一组一个或更多个总线控制器单元2416和系统代理核心2410。在至少一个实施例中,一个或更多个总线控制器单元2416管理一组外围总线,例如一个或更多个PCI或PCIe总线。在至少一个实施例中,系统代理核心2410为各种处理器组件提供管理功能。在至少一个实施例中,系统代理核心2410包括一个或更多个集成存储器控制器2414,以管理对各种外部存储器设备(未示出)的访问。

[0267] 在至少一个实施例中,一个或更多个处理器核心2402A-2402N包括对多线程同时进行的支持。在至少一个实施例中,系统代理核心2410包括用于在多线程处理期间协调和操作核心2402A-2402N的组件。在至少一个实施例中,系统代理核心2410可以另外包括电源控制单元(PCU),该电源控制单元包括用于调节处理器核心2402A-2402N和图形处理器2408的一个或更多个电源状态的逻辑和组件。

[0268] 在至少一个实施例中,处理器2400还包括用于执行图处理操作的图形处理器2408。在至少一个实施例中,图形处理器2408与共享高速缓存单元2406和包括一个或更多个集成存储器控制器2414的系统代理核心2410耦合。在至少一个实施例中,系统代理核心2410还包括用于驱动图形处理器输出到一个或更多个耦合的显示器的显示器控制器2411。在至少一个实施例中,显示器控制器2411也可以是经由至少一个互连与图形处理器2408耦合的独立模块,或者可以集成在图形处理器2408内。

[0269] 在至少一个实施例中,基于环的互连单元2412用于耦合处理器2400的内部组件。在至少一个实施例中,可以使用替代性互连单元,例如点对点互连、交换互连或其他技术。在至少一个实施例中,图形处理器2408经由I/O链路2413与环形互连2412耦合。

[0270] 在至少一个实施例中,I/O链路2413代表多种I/O互连中的至少一种,包括促进各

种处理器组件与高性能嵌入式存储器模块2418(例如eDRAM模块)之间的通信的封装I/O互连。在至少一个实施例中,处理器核心2402A-2402N和图形处理器2408中的每一个使用嵌入式存储器模块2418作为共享的最后一级高速缓存。

[0271] 在至少一个实施例中,处理器核心2402A-2402N是执行公共指令集架构的同质核心。在至少一个实施例中,处理器核心2402A-2402N在指令集架构(ISA)方面是异构的,其中一个或多个处理器核心2402A-2402N执行公共指令集,而一个或多个其他处理器核心2402A-2402N执行公共指令集的子集或不同指令集。在至少一个实施例中,就微架构而言,处理器核心2402A-2402N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。在至少一个实施例中,处理器2400可以在一个或多个芯片上实现或被实现为SoC集成电路。

[0272] 推理和/或训练逻辑615用于执行与一个或多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615的部分或全部可以结合到处理器2400中。例如,在至少一个实施例中,本文描述的训练和/或推理技术可以使用在图形处理器2312、一个或多个图形核心2402A-2402N或图24中的其他组件中具体化的ALU中的一个或多个。此外,在至少一个实施例中,本文描述的推理和/或训练操作可以使用除了图6A或图6B中所示出的逻辑之外的逻辑来完成。在至少一个实施例中,可以将权重参数存储在片上或片外存储器和/或寄存器(示出或未示出的)中,该权重参数将图形处理器2400的ALU配置成用于执行一个或多个机器学习算法、神经网络架构、用例或本文描述的训练技术。

[0273] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0274] 图25是根据本文描述的至少一个实施例的图形处理器核心2500的硬件逻辑的框图。在至少一个实施例中,图形处理器核心2500包括在图形核心阵列内。在至少一个实施例中,图形处理器核心2500(有时被称为核心切片)可以是模块化图形处理器内的一个或多个图形核心。在至少一个实施例中,图形处理器核心2500是一个图形核心切片的示例,并且如本文描述的图形处理器可以包括基于目标功率和性能包络的多个图形核心切片。在至少一个实施例中,每个图形核心2500可以包括与多个子核心2501A-2501F(也被称为子切片)耦合的固定功能块2530,所述多个子核心包括具有通用和固定功能逻辑的模块化块。

[0275] 在至少一个实施例中,固定功能块2530包括几何和固定功能管线2536,例如,在较低性能和/或较低功率的图形处理器实施方式中,该几何和固定功能管线2536可以由图形处理器2500中的所有子核心共享。在至少一个实施例中,几何和固定功能管线2536包括3D固定功能管线、视频前端单元,线程产生器和线程分派器以及管理统一返回缓冲区的统一返回缓冲区管理器。

[0276] 在固定的至少一个实施例中,固定功能块2530还包括图形SoC接口2537、图形微控制器2538和媒体管线2539。在至少一个实施例中,图形SoC接口2537提供了图形核心2500以及片上集成电路系统中的其他处理器核心之间的接口。在至少一个实施例中,图形微控制器2538是可编程子处理器,其可配置为管理图形处理器2500的各种功能,包括线程分派、调度和抢占。在至少一个实施例中,媒体管线2539包括有助于对包括图像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。在至少一个实施例中,媒体管线2539经

由对子核心2501-2501F内的计算或采样逻辑的请求来实现媒体操作。

[0277] 在至少一个实施例中,SoC接口2537使图形核心2500能够与通用应用程序处理器核心(例如,CPU)和/或SoC内的其他组件通信,包括存储器层次结构元素,诸如共享的最后一级高速缓存、系统RAM和/或嵌入式片上或封装DRAM。在至少一个实施例中,SoC接口2537还可以使得能够与SoC内的固定功能设备(例如,相机成像管线)进行通信,并且使得能够使用和/或实现可以在图形核心2500和SoC内部的CPU之间共享的全局存储器原子。在至少一个实施例中,SoC接口2537还可以实现用于图形核心2500的电源管理控制,并且启用图形核心2500的时钟域与SoC内的其他时钟域之间的接口。在至少一个实施例中,SoC接口2537使得能够从命令流转化器和全局线程分派器接收命令缓冲区,其配置为向图形处理器内的一个或多个图形核心中的每一个提供命令和指令。在至少一个实施例中,当要执行媒体操作时,可以将命令和指令分派给媒体管线2539,或者当要执行图形处理操作时,可以将其分配给几何形状和固定功能管线(例如,几何形状和固定功能管线2536,几何形状和固定功能管线2514)。

[0278] 在至少一个实施例中,图形微控制器2538可以配置为对图形核心2500执行各种调度和管理任务。在至少一个实施例中,图形微控制器2538可以在子核心2501A-2501F中的执行单元(EU)阵列2502A-2502F、2504A-2504F内的各种图形并行引擎上执行图形和/或计算工作负载调度。在至少一个实施例中,在包括图形核心2500的SoC的CPU核心上执行的主机软件可以提交多个图形处理器门铃之一的工作负载,其调用适当的图形引擎上的调度操作。在至少一个实施例中,调度操作包括确定接下来要运行哪个工作负载、将工作负载提交给命令流转化器、抢先在引擎上运行的现有工作负载、监控工作负载的进度以及在工作负载完成时通知主机软件。在至少一个实施例中,图形微控制器2538还可以促进图形核心2500的低功率或空闲状态,从而为图形核心2500提供在图形核心2500内独立于操作系统和/或系统上的图形驱动程序软件的跨低功率状态转换的保存和恢复寄存器的能力。

[0279] 在至少一个实施例中,图形核心2500可以具有比所示的子核心2501A-2501F多或少达N个模块化子核心。对于每组N个子核心,在至少一个实施例中,图形核心2500还可以包括共享功能逻辑2510、共享和/或高速缓存存储器2512、几何/固定功能管线2514以及附加的固定功能逻辑2516以加速各种图形和计算处理操作。在至少一个实施例中,共享功能逻辑2510可以包括可由图形核心2500内的每个N个子核心共享的逻辑单元(例如,采样器、数学和/或线程间通信逻辑)。在至少一个实施例中,共享和/或高速缓存存储器2512可以是图形核心2500内的N个子核心2501A-2501F的最后一级高速缓存,并且还可以用作可由多个子核心访问的共享存储器。在至少一个实施例中,可以包括几何/固定功能管线2514来代替固定功能块2530内的几何/固定功能管线2536,并且可以包括相同或相似的逻辑单元。

[0280] 在至少一个实施例中,图形核心2500包括附加的固定功能逻辑2516,其可以包括供图形核心2500使用的各种固定功能加速逻辑。在至少一个实施例中,附加的固定功能逻辑2516包括用于仅位置着色中使用的附加的几何管线。在仅位置着色中,存在至少两个几何管线,而在几何和固定功能管线2514、2536内的完整几何管线和剔除管线中,其是可以包括在附加的固定功能逻辑2516中的附加几何管线。在至少一个实施例中,剔除管线是完整几何管线的修整版。在至少一个实施例中,完整管线和剔除管线可以执行应用程序的不同实例,每个实例具有单独的环境。在至少一个实施例中,仅位置着色可以隐藏被丢弃的三角

形的长剔除运行,从而在某些情况下可以更早地完成着色。例如,在至少一个实施例中,附加固定功能逻辑2516中的剔除管线逻辑可以与主应用程序并行执行位置着色器,并且通常比完整管线更快地生成关键结果,因为剔除管线获取并遮蔽顶点的位置属性,无需执行光栅化和将像素渲染到帧缓冲区。在至少一个实施例中,剔除管线可以使用生成的临界结果来计算所有三角形的可见性信息,而与这些三角形是否被剔除无关。在至少一个实施例中,完整管线(在这种情况下可以称为重播管线)可以消耗可见性信息来跳过剔除的三角形以仅遮盖最终传递到光栅化阶段的可见三角形。

[0281] 在至少一个实施例中,附加的固定功能逻辑2516还可包括机器学习加速逻辑,例如固定功能矩阵乘法逻辑,用于实现包括用于机器学习训练或推理的优化。

[0282] 在至少一个实施例中,在每个图形子核心2501A-2501F内包括一组执行资源,其可用于响应于图形管线、媒体管线或着色器程序的请求来执行图形、媒体和计算操作。在至少一个实施例中,图形子核心2501A-2501F包括多个EU阵列2502A-2502F、2504A-2504F,线程分派和线程间通信(TD/IC)逻辑2503A-2503F,3D(例如,纹理)采样器2505A-2505F,媒体采样器2506A-2506F,着色器处理器2507A-2507F和共享本地存储器(SLM)2508A-2508F。EU阵列2502A-2502F、2504A-2504F每个都包含多个执行单元,这些执行单元是通用图形处理单元,能够为图形、媒体或计算操作提供服务,执行浮点和整数/定点逻辑运算,包括图形、媒体或计算着色器程序。在至少一个实施例中,TD/IC逻辑2503A-2503F为子核心内的执行单元执行本地线程分派和线程控制操作,并促进在子核心的执行单元上执行的线程之间的通信。在至少一个实施例中,3D采样器2505A-2505F可以将与纹理或其他3D图形相关的数据读取到存储器中。在至少一个实施例中,3D采样器可以基于与给定纹理相关联的配置的采样状态和纹理格式来不同地读取纹理数据。在至少一个实施例中,媒体采样器2506A-2506F可以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。在至少一个实施例中,每个图形子核心2501A-2501F可以可替代地包括统一的3D和媒体采样器。在至少一个实施例中,在每个子核心2501A-2501F内的执行单元上执行的线程可以利用每个子核心内的共享本地存储器2508A-2508F,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。

[0283] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615的部分或全部可以被合并到图形处理器2510中。例如,在至少一个实施例中,本文描述的训练和/或推理技术可以使用在图形处理器2312、图形微控制器2538、几何和固定功能管线2514和2536或图25中的其他逻辑中体现的一个或更多个ALU。此外,在至少一个实施例中,本文描述的推理和/或训练操作可以使用除图6A或图6B所示的逻辑以外的逻辑来完成。在至少一个实施例中,权重参数可以存储在片上或片外存储器和/或寄存器(示出或未示出)中,其配置图形处理器2500的ALU以执行一种或更多种本文介绍的机器学习算法、神经网络架构、用例或训练技术。

[0284] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0285] 图26A-26B示出了根据至少一个实施例的包括图形处理器核心的处理元件的阵列的线程执行逻辑2600。图26A示出了至少一个实施例,其中使用了线程执行逻辑2600。图26B

示出了根据至少一个实施例的执行单元的示例性内部细节。

[0286] 如图26A中所示,在至少一个实施例中,线程执行逻辑2600包括着色器处理器2602、线程分派器2604、指令高速缓存2606、包括多个执行单元2608A-2608N的可缩放执行单元阵列、采样器2610、数据高速缓存2612和数据端口2614。在至少一个实施例中,可缩放执行单元阵列可以例如基于工作负载的计算要求,通过启用或禁用一个或更多个执行单元(例如,执行单元2608A、2608B、2608C、2608D至2608N-1和2608N中的任意一个)来动态缩放。在至少一个实施例中,可缩放执行单元通过链路到每个执行单元的互连结构互连。在至少一个实施例中,线程执行逻辑2600包括通过指令高速缓存2606、数据端口2614、采样器2610和执行单元2608A-2608N中的一个或更多个到存储器(诸如系统存储器或高速缓存存储器)的一个或更多个连接。在至少一个实施例中,每个执行单元(例如2608A)是独立的可编程通用计算单元,其能够执行多个同时的硬件线程,同时针对每个线程并行处理多个数据元素。在至少一个实施例中,执行单元2608A-2608N的阵列可缩放以包括任意数量的单独执行单元。

[0287] 在至少一个实施例中,执行单元2608A-2608N主要用于执行着色器程序。在至少一个实施例中,着色器处理器2602可以处理各种着色器程序并经由线程分派器2604来分派与着色器程序相关联的执行线程。在至少一个实施例中,线程分派器2604包括用于仲裁来自图形和媒体管线的线程初始化庆祝以及在执行单元2608A-2608N中的一个或更多个执行单元上实例化请求的线程的逻辑。例如,在至少一个实施例中,几何管线可以将顶点、镶嵌或几何着色器分派到线程执行逻辑以进行处理。在至少一个实施例中,线程分派器2604还可以处理来自执行着色器程序的运行时线程产生请求。

[0288] 在至少一个实施例中,执行单元2608A-2608N支持一种指令集,该指令集包括对许多标准3D图形着色器指令的本机支持,从而使图形库(例如Direct 3D和OpenGL)中的着色器程序只需最少的转换即可执行。在至少一个实施例中,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)。在至少一个实施例中,每个执行单元2608A-2608N包括一个或更多个算术逻辑单元(ALU),能够执行多发出单指令多数据(SIMD),并且多线程操作实现了高效的执行环境尽管有更高的延迟存储器访问。在至少一个实施例中,每个执行单元内的每个硬件线程具有专用的高带宽寄存器文件和相关的独立线程状态。在至少一个实施例中,执行是每个时钟到管线的多次发出,管线能够进行整数、单精度和双精度浮点运算、SIMD分支功能、逻辑运算、先验运算和其他其他运算。在至少一个实施例中,在等待来自存储器或共享功能之一的数据时,执行单元2608A-2608N内的依赖性逻辑使等待线程休眠直到返回了所请求的数据。在至少一个实施例中,当等待线程正在休眠时,硬件资源可以专用于处理其他线程。例如,在至少一个实施例中,在与顶点着色器操作相关联的延迟期间,执行单元可以对像素着色器、片段着色器或另一类型的着色器程序(包括不同的顶点着色器)执行操作。

[0289] 在至少一个实施例中,执行单元2608A-2608N中的每一个执行单元在数据元素的阵列上进行操作。在至少一个实施例中,多个数据元素是“执行大小”或指令的通道数。在至少一个实施例中,执行通道是用于指令内的数据元素访问、屏蔽和流控制的执行的逻辑单元。在至少一个实施例中,多个通道可以独立于用于特定图形处理器的多个物理算术逻辑

单元 (ALU) 或浮点单元 (FPU)。在至少一个实施例中,执行单元2608A-2608N支持整数和浮点数据类型。

[0290] 在至少一个实施例中,执行单元指令集包括SIMD指令。在至少一个实施例中,各种数据元素可以作为封装数据类型存储在寄存器中,并且执行单元将基于那些元素的数据大小来处理各种元素。例如,在至少一个实施例中,当对256位宽的向量进行操作时,将向量的256位存储在寄存器中,并且执行单元对向量进行操作,作为四个单独的64位封装数据元素(四字(QW)大小数据元素)、八个单独的32位封装数据元素(双字(DW)大小数据元素)、十六个单独的16位封装数据元素(单词(W)大小数据元素)或三十二个单独的8位数据元素(字节(B)大小的数据元素)。然而,在至少一个实施例中,不同的向量宽度和寄存器大小是可能的。

[0291] 在至少一个实施例中,一个或更多个执行单元可以被组合成具有与融合EU共用的线程控制逻辑(2607A-2607N)的融合执行单元2609A-2609N。在至少一个实施例中,可以将多个EU合并成一个EU组。在至少一个实施例中,融合EU组中的EU的数量可以配置为执行单独的SIMD硬件线程。融合的EU组中的EU的数量可根据各个实施例而变化。在至少一个实施例中,每个EU可以执行各种SIMD宽度,包括但不限于SIMD8、SIMD16和SIMD32。在至少一个实施例中,每个融合图形执行单元2609A-2609N包括至少两个执行单元。例如,在至少一个实施例中,融合执行单元2609A包括第一EU 2608A、第二EU 2608B以及与第一EU 2608A和第二EU 2608B共用的线程控制逻辑2607A。在至少一个实施例中,线程控制逻辑2607A控制在融合图形执行单元2609A上执行的线程,从而允许融合执行单元2609A-2609N内的每个EU使用公共指令指针寄存器来执行。

[0292] 在至少一个实施例中,一个或更多个内部指令高速缓存(例如2606)被包括在线程执行逻辑2600中以高速缓存用于执行单元的线程指令。在至少一个实施例中,包括一个或更多个数据高速缓存(例如2612)以在线程执行期间高速缓存线程数据。在至少一个实施例中,包括采样器2610以提供用于3D操作的纹理采样和用于媒体操作的媒体采样。在至少一个实施例中,采样器2610包括专门的纹理或媒体采样功能,以在将采样数据提供给执行单元之前在采样过程中处理纹理或媒体数据。

[0293] 在执行期间,在至少一个实施例中,图形和媒体管线通过线程产生和分派逻辑将线程发起请求发送到线程执行逻辑2600。在至少一个实施例中,一旦一组几何对象已经被处理并光栅化成像素数据,则在着色器处理器2602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)被调用以进一步计算输出信息并且导致将结果写入输出表面(例如,颜色缓冲区、深度缓冲区、模板缓冲区等)。在至少一个实施例中,像素着色器或片段着色器计算要在光栅化对象上插值的各种顶点属性的值。在至少一个实施例中,着色器处理器2602内的像素处理器逻辑然后执行应用程序接口(API)提供的像素或片段着色器程序。在至少一个实施例中,为了执行着色器程序,着色器处理器2602经由线程分派器2604将线程分派到执行单元(例如2608A)。在至少一个实施例中,着色器处理器2602使用采样器2610中的纹理采样逻辑来访问存储在存储器中的纹理贴图纹理数据。在至少一个实施例中,对纹理数据和输入几何数据的算术运算为每个几何片段计算像素颜色数据,或者丢弃一个或更多个像素以进行进一步处理。

[0294] 在至少一个实施例中,数据端口2614提供了一种用于线程执行逻辑2600的存储器

访问机制,以将处理后的数据输出到存储器以在图形处理器输出管线上进行进一步处理。在至少一个实施例中,数据端口2614包括或耦合到一个或多个高速缓存存储器(例如,数据高速缓存2612)以高速缓存数据以便经由数据端口进行存储器访问。

[0295] 如图26B所示,在至少一个实施例中,图形执行单元2608可以包括指令获取单元2637、通用寄存器文件阵列 (GRF) 2624、架构寄存器文件阵列 (ARF) 2626、线程仲裁器2622、发送单元2630、分支单元2632、一组SIMD浮点单元 (FPU) 2634,以及在至少一个实施例中,一组专用整数SIMD ALU 2635。GRF 2624和ARF 2626包括一组与可以在图形执行单元2608中活跃的每个同时硬件线程相关联的通用寄存器文件和架构寄存器文件。在至少一个实施例中,在ARF 2626中维护每个线程架构状态,而在线程执行期间使用的数据存储GRF 2624中。在至少一个实施例中,每个线程的执行状态,包括每个线程的指令指针,可以被保存在ARF 2626中的线程专用寄存器中。

[0296] 在至少一个实施例中,图形执行单元2608具有一种架构,该架构是同时多线程 (SMT) 和细粒度交错多线程 (IMT) 的组合。在至少一个实施例中,架构具有模块化配置,该模块化配置可以在设计时基于同时线程的目标数量和每个执行单元的寄存器数量来进行微调,其中执行单元资源在用于执行多个同时线程的逻辑上分配。

[0297] 在至少一个实施例中,图形执行单元2608可以共同发布多个指令,每个指令可以是不同的指令。在至少一个实施例中,图形执行单元线程2608的线程仲裁器2622可以将指令分派到发送单元2630、分支单元2632或SIMD FPU 2634之一以供执行。在至少一个实施例中,每个执行线程可以访问GRF 2624中的128个通用寄存器,其中每个寄存器可以存储32个字节,可以作为32位数据元素的SIMD 8元素向量进行访问。在至少一个实施例中,每个执行单元线程可以访问GRF 2624中的4KB,尽管实施例不限于此,并且在其他实施例中可以提供更多或更少的寄存器资源。在至少一个实施例中,尽管每个执行单元的线程数量也可以根据实施例而变化,但是最多可以同时执行七个线程。在其中七个线程可以访问4KB的至少一个实施例中,GRF 2624可以存储总共28KB。在至少一个实施例中,灵活的寻址模式可以允许将寄存器一起寻址以有效地建立更宽的寄存器或表示跨步的矩形块数据结构。

[0298] 在至少一个实施例中,经由由消息传递发送单元2630执行的“发送”指令来调度存储器操作、采样器操作和其他更长延迟的系统通信。在至少一个实施例中,将分支指令分派到专用分支单元2632促进SIMD发散和最终收敛。

[0299] 在至少一个实施例中,图形执行单元2608包括一个或多个SIMD浮点单元 (FPU) 2634,以执行浮点操作。在至少一个实施例中,一个或多个FPU 2634还支持整数计算。在至少一个实施例中,一个或多个FPU 2634可以SIMD执行多达M个32位浮点(或整数)运算,或者SIMD执行多达2M个16位整数或16位浮点运算。在至少一个实施例中,至少一个FPU提供扩展的数学能力以支持高吞吐量的先验数学函数和双精度64位浮点。在至少一个实施例中,还存在一组8位整数SIMD ALU 2635,并且可以被专门优化以执行与机器学习计算相关的操作。

[0300] 在至少一个实施例中,可以在图形子核心分组(例如,子切片)中实例化图形执行单元2608的多个实例的阵列。在至少一个实施例中,执行单元2608可以跨多个执行通道执行指令。在至少一个实施例中,在图形执行单元2608上执行的每个线程在不同的通道上执行。

[0301] 推理和/或训练逻辑615用于执行与一个或多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,推理和/或训练逻辑615的部分或全部可以被并入执行逻辑2600中。此外,在至少一个实施例中,可以使用除了图6A或图6B中所示的逻辑之外的逻辑来完成在此描述的推理和/或训练操作。在至少一个实施例中,权重参数可以存储在片上或片外存储器和/或寄存器(示出或未示出)中,其配置执行逻辑2600的ALU以执行一种或更多种机器学习算法、神经网络架构、用例或本文介绍的训练技术。

[0302] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0303] 图27示出了根据至少一个实施例的并行处理单元(“PPU”)2700。在至少一个实施例中,PPU 2700配置有机器可读代码,该机器可读代码如果由PPU 2700执行,则使得PPU 2700执行贯穿本公开描述的一些或全部过程和技术。在至少一个实施例中,PPU 2700是在一个或多个集成电路设备上实现的多线程处理器,并且利用多线程作为被设计为处理在多个线程上并行执行的计算机可读指令(也称为机器可读指令或简单的指令)的延迟隐藏技术。在至少一个实施例中,线程是指执行线程,并且是被配置为由PPU 2700执行的一组指令的实例。在至少一个实施例中,PPU 2700是图形处理单元(“GPU”),图形处理单元配置为实现用于处理三维(“3D”)图形数据的图形渲染管线,以便生成用于在显示设备(诸如液晶显示器(“LCD”)设备)上显示的二维(“2D”)图像数据。在至少一个实施例中,PPU 2700用于执行计算,诸如线性代数运算和机器学习运算。图27仅出于说明性目的示出了示例并行处理器,并且应被解释为在本公开的范围内设想的处理器架构的非限制性示例,并且可以采用任何适当的处理器来对其进行补充和/或替代。

[0304] 在至少一个实施例中,一个或多个PPU 2700配置成加速高性能计算(“HPC”)、数据中心和机器学习应用程序。在至少一个实施例中,PPU2700配置成加速深度学习系统和应用程序,包括以下非限制性示例:自动驾驶汽车平台、深度学习、高精度语音、图像、文本识别系统、智能视频分析、分子模拟、药物发现、疾病诊断、天气预报、大数据分析、天文学、分子动力学模拟、财务建模、机器人技术、工厂自动化、实时语言翻译、在线搜索优化以及个性化用户推荐等。

[0305] 在至少一个实施例中,PPU 2700包括但不限于输入/输出(“I/O”)单元2706、前端单元2710、调度器单元2712、工作分配单元2714、集线器2716、交叉开关(“Xbar”)2720、一个或多个通用处理集群(“GPC”)2718和一个或多个分区单元(“存储器分区单元”)2722。在至少一个实施例中,PPU 2700通过一个或多个高速GPU互连(“GPU互连”)2708连接到主机处理器或其他PPU 2700。在至少一个实施例中,PPU 2700通过互连2702连接到主机处理器或其他外围设备。在一实施例中,PPU 2700连接到包括一个或多个存储器设备(“存储器”)2704的本地存储器。在至少一个实施例中,存储器设备2704包括但不限于一个或多个动态随机存取存储器(“DRAM”)设备。在至少一个实施例中,一个或多个DRAM设备配置和/或可配置为高带宽存储器(“HBM”)子系统,并且在每个设备内堆叠有多个DRAM管芯。

[0306] 在至少一个实施例中,高速GPU互连2708可以指代系统使用其来进行缩放的基于线的多通道通信链路,并包括与一个或多个中央处理单元结合的一个或多个PPU 2700(“CPU”),支持PPU 2700和CPU之间的缓存相干以及CPU主控。在至少一个实施例中,高速GPU

互连2708通过集线器2716将数据和/或命令传输到PPU 2700的其他单元,例如一个或多个复制引擎、视频编码器、视频解码器、电源管理单元和/或在图27中可能未明确示出的其他组件。

[0307] 在至少一个实施例中,I/O单元2706配置为通过系统总线2702从主机处理器(图27中未示出)发送和接收通信(例如,命令、数据)。在至少一个实施例中,I/O单元2706直接通过系统总线2702或通过一个或多个中间设备(例如存储器桥)与主机处理器通信。在至少一个实施例中,I/O单元2706可以经由系统总线2702与一个或多个其他处理器(例如一个或多个PPU 2700)通信。在至少一个实施例中,I/O单元2706实现外围组件互连Express(“PCIe”)接口,用于通过PCIe总线进行通信。在至少一个实施例中,I/O单元2706实现用于与外部设备通信的接口。

[0308] 在至少一个实施例中,I/O单元2706对经由系统总线2702接收的分组进行解码。在至少一个实施例中,至少一些分组表示被配置为使PPU 2700执行各种操作的命令。在至少一个实施例中,I/O单元2706如命令所指定的那样将解码的命令发送到PPU 2700的各种其他单元。在至少一个实施例中,命令被发送到前端单元2710和/或被发送到集线器2716或PPU 2700的其他单元,例如一个或多个复制引擎、视频编码器、视频解码器、电源管理单元等(图27中未明确示出)。在至少一个实施例中,I/O单元2706配置为在PPU 2700的各种逻辑单元之间路由通信。

[0309] 在至少一个实施例中,由主机处理器执行的程序在缓冲区中对命令流进行编码,该缓冲区将工作负载提供给PPU 2700以进行处理。在至少一个实施例中,工作负载包括指令和要由那些指令处理的数据。在至少一个实施例中,缓冲区是可由主机处理器和PPU 2700两者访问(例如,读/写)的存储器中的区域—主机接口单元可以配置为访问经由I/O单元2706通过系统总线2702传输的存储器请求连接到系统总线2702的系统存储器中的缓冲区。在至少一个实施例中,主机处理器将命令流写入缓冲区,然后将指示命令流开始的指针发送给PPU 2700,使得前端单元2710接收指向一个或多个命令流指针并管理一个或多个命令流,从命令流中读取命令并将命令转发到PPU 2700的各个单元。

[0310] 在至少一个实施例中,前端单元2710耦合到调度器单元2712,该调度器单元2712配置各种GPC 2718以处理由一个或多个命令流定义的任务。在至少一个实施例中,调度器单元2712配置为跟踪与调度器单元2712管理的各种任务有关的状态信息,其中状态信息可以指示任务被分配给哪个GPC 2718,任务是活跃的还是非活跃的,与任务相关联的优先级等等。在至少一个实施例中,调度器单元2712管理在一个或多个GPC 2718上执行的多个任务。

[0311] 在至少一个实施例中,调度器单元2712耦合到工作分配单元2714,该工作分配单元2714配置为分派任务以在GPC 2718上执行。在至少一个实施例中,工作分配单元2714跟踪从调度器单元2712接收到的多个调度任务并且工作分配单元2714管理每个GPC 2718的待处理任务池和活跃任务池。在至少一个实施例中,待处理任务池包括多个时隙(例如32个时隙),这些时隙包含分配给要由特定的GPC 2718处理的任务;活跃任务池可包括用于由GPC 2718主动处理的任务的多个时隙(例如4个时隙),以使随着GPC 2718中的一个完成任务的执行,该任务将从GPC 2718的活动任务池中逐出,并且从待处理任务池中选择其他任务之一,并安排其在GPC 2718上执行。在至少一个实施例中,如果活跃任务在GPC 2718上处

于空闲状态,例如在等待数据依赖性解决时,则活跃任务从GPC 2718中驱逐并返回到待处理任务池,同时选择了待处理任务池中的另一个任务并调度在GPC 2718上执行。

[0312] 在至少一个实施例中,工作分配单元2714经由XBar 2720与一个或更多个GPC 2718通信。在至少一个实施例中,XBar 2720是互连网络,其将PPU 2700的许多单元耦合到PPU 2700的其他单元,并且可以配置为将工作分配单元2714耦合到特定的GPC 2718。在至少一个实施例中,一个或更多个PPU 2700的其他单元也可以通过集线器2716连接到XBar 2720。

[0313] 在至少一个实施例中,任务由调度器单元2712管理,并由工作分配单元2714分配给GPC 2718之一。GPC 2718配置为处理任务并产生结果。在至少一个实施例中,结果可以由GPC 2718中的其他任务消耗,通过XBar 2720路由到不同的GPC 2718或存储在存储器2704中。在至少一个实施例中,结果可以通过分区单元2722写到存储器2704中,其实现了用于向存储器2704写入数据或从存储器2704读取数据的存储器接口。在至少一个实施例中,结果可以经由高速GPU互连2708传输到另一PPU 2704或CPU。在至少一个实施例中,PPU 2700包括但不限于U个分区单元2722,其等于耦合到PPU 2700的分离且不同的存储器设备2704的数量。在至少一个实施例中,分区单元2722将在下面结合图29更详细地描述。

[0314] 在至少一个实施例中,主机处理器执行驱动器核心,该驱动程序核心实现应用程序编程接口(API),该应用程序编程接口使在主机处理器上执行的一个或更多个应用程序能够调度操作以在PPU 2700上执行。在一个实施例中,多个计算应用程序由PPU 2700同时执行,并且PPU 2700为多个计算应用程序提供隔离、服务质量(“QoS”)和独立的地址空间。在至少一个实施例中,应用程序生成指令(例如,以API调用的形式),该指令使驱动器核心生成一个或更多个任务以供PPU 2700执行,并且驱动器核心将任务输出至由PPU 2700处理的一个或更多个流。在至少一个实施例中,每个任务包括一个或更多个相关线程组,其可以被称为线程束(warp)。在至少一个实施例中,线程束包括可以并行执行的多个相关线程(例如32个线程)。在至少一个实施例中,协作线程可以指代多个线程,所述线程包括用于执行任务并且通过共享存储器交换数据的指令。在至少一个实施例中,结合图29根据至少一个实施例更详细地描述了线程和协作线程。

[0315] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,深度学习应用程序处理器用于训练机器学习模型(诸如神经网络),以预测或推理提供给PPU 2700的信息。在至少一个实施例中,PPU 2700用于基于已由另一处理器或系统或PPU 2700训练过的训练过的机器学习模型(例如,神经网络)推理或预测信息。在至少一个实施例中,PPU 2700可用于执行本文所述的一个或更多个神经网络用例。

[0316] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0317] 图28示出了根据至少一个实施例的通用处理集群(“GPC”)2800。在至少一个实施例中,GPC 2800是图27的GPC 2718。在至少一个实施例中,每个GPC 2800包括但不限于用于处理任务的多个硬件单元,并且每个GPC 2800包括但不限于管线管理器2802、预光栅操作单元(“preROP”)2804、光栅引擎2808、工作分配交叉开关(“WDX”)2816、存储器管理单元(“MMU”)2818、一个或更多个数据处理集群(“DPC”)2806,以及部件的任何合适组合。

[0318] 在至少一个实施例中, GPC 2800的操作由管线管理器2802控制。在至少一个实施例中, 管线管理器2802管理一个或更多个DPC 2806的配置, 以处理分配给GPC 2800的任务。在至少一个实施例中, 管线管理器2802配置一个或更多个DPC 2806中的至少一个以实现图形渲染管线的至少一部分。在至少一个实施例中, DPC 2806配置为在可编程流式多处理器 (“SM”) 2814上执行顶点着色器程序。在至少一个实施例中, 管线管理器2802配置为将从工作分配单元接收的数据包路由到GPC 2800内的适当逻辑单元, 以及在至少一个实施例中, 可以将一些数据包路由到preROP 2804和/或光栅引擎2808中的固定功能硬件单元, 而可以将其他数据包路由到DPC 2806以由原始引擎2812或SM 2814进行处理。在至少一个实施例中, 管线管理器2802配置DPC 2806中的至少一个以实现神经网络模型和/或计算管线。

[0319] 在至少一个实施例中, preROP单元2804配置为在至少一个实施例中将由光栅引擎2808和DPC 2806生成的数据路由到分区单元2722中的光栅操作 (“ROP”) 单元, 上面结合图27更详细地描述。在至少一个实施例中, preROP单元2804配置为执行用于颜色混合的优化、组织像素数据、执行地址转换等等。在至少一个实施例中, 光栅引擎2808包括但不限于配置为执行各种光栅操作的多个固定功能硬件单元, 并且在至少一个实施例中, 光栅引擎2808包括但不限于设置引擎、粗光栅引擎、剔除引擎、裁剪引擎、精细光栅引擎、图块聚合引擎及其任意合适的组合。在至少一个实施例中, 设置引擎接收变换后的顶点并生成与由顶点定义的几何图元相关联的平面方程; 平面方程式被传送到粗光栅引擎以生成基本图元的覆盖信息 (例如, 图块的x、y覆盖范围掩码); 粗光栅引擎的输出将传输到剔除引擎, 在剔除引擎中与z测试失败的图元相关联的片段将被剔除, 并传输到剪切引擎, 在剪切引擎中剪切位于视锥范围之外的片段。在至少一个实施例中, 将经过裁剪和剔除的片段传递给精细光栅引擎, 以基于设置引擎生成的平面方程式生成像素片段的属性。在至少一个实施例中, 光栅引擎2808的输出包括将由任何适当的实体 (例如, 由在DPC 2806内实现的片段着色器) 处理的片段。

[0320] 在至少一个实施例中, 包括在GPC 2800中的每个DPC 2806包括但不限于M管线控制器 (“MPC”) 2810; 图元引擎2812; 一个或更多个SM 2814; 及其任何合适的组合。在至少一个实施例中, MPC 2810控制DPC 2806的操作, 将从管线管理器2802接收的分组路由到DPC 2806中的适当单元。在至少一个实施例中, 将与顶点相关联的分组路由到图元引擎2812, 图元引擎2812配置为从存储器中获取与顶点关联的顶点属性; 相反, 可以将与着色器程序相关联的数据包发送到SM 2814。

[0321] 在至少一个实施例中, SM 2814包括但不限于可编程流式处理器, 其配置为处理由多个线程表示的任务。在至少一个实施例中, SM 2814是多线程的并且配置为同时执行来自特定线程组的多个线程 (例如32个线程), 并且实现单指令、多数据 (“SIMD”) 架构, 其中将一组线程 (例如, 线程束) 中的每个线程配置为基于相同的指令集来处理不同的数据集。在至少一个实施例中, 线程组中的所有线程执行相同的指令。在至少一个实施例中, SM 2814实施单指令、多线程 (“SIMT”) 架构, 其中一组线程中的每个线程配置为基于相同的指令来处理不同的数据集, 但是其中线程组中的各个线程允许在执行期间发散。在至少一个实施例中, 为每个线程束维护程序计数器、调用栈和执行状态, 从而当线程束中的线程发散时, 实现线程束和线程束内的串行执行之间的并发性。在另一个实施例中, 为每个单独的线程维护程序计数器、调用栈和执行状态, 从而使得在线程束内和线程束之间的所有线程之间具

有相等的并发性。在至少一个实施例中,为每个单独的线程维持执行状态,并且可以收敛并并行地执行执行相同的指令的线程以提高效率。下面更详细地描述SM 2814的至少一个实施例。

[0322] 在至少一个实施例中,MMU 2818在GPC 2800和存储器分区单元(例如,图27的分区单元2722)之间提供接口,并且MMU 2818提供虚拟地址到物理地址的转换、存储器保护以及存储器请求的仲裁。在至少一个实施例中,MMU 2818提供一个或多个转换后备缓冲区(“TLB”),用于执行虚拟地址到存储器中的物理地址的转换。

[0323] 推理和/或训练逻辑615用于执行与一个或多个实施例相关联的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,深度学习应用程序处理器用于训练机器学习模型(诸如神经网络),以预测或推理提供给GPC 2800的信息。在至少一个实施例中,GPC 2800用于基于已由另一处理器或系统或GPC 2800训练过的机器学习模型(例如,神经网络)推理或预测信息。在至少一个实施例中,GPC 2800可用于执行本文所述的一个或多个神经网络用例。

[0324] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0325] 图29示出了根据至少一个实施例的并行处理单元(“PPU”)的存储器分区单元2900。在至少一个实施例中,存储器分区单元2900包括但不限于光栅操作(“ROP”)单元2902;二级(“L2”)高速缓存2904;存储器接口2906;及其任何合适的组合。在至少一个实施例中,存储器接口2906耦合到存储器。在至少一个实施例中,存储器接口2906可以实现32、64、128、1024位数据总线,或者类似的实现方式用于高速数据传输。在至少一个实施例中,PPU包括U个存储器接口2906,每对分区单元2900一个存储器接口2906,其中每对分区单元2900连接到对应的存储器设备。例如,在至少一个实施例中,PPU可以连接至多达Y个存储器设备,例如高带宽存储器堆栈或图形双数据速率版本5同步动态随机存取存储器(“GDDR5 SDRAM”)。

[0326] 在至少一个实施例中,存储器接口2906实现高带宽存储器第二代(“HBM2”)存储器接口,并且Y等于U的一半。在至少一个实施例中,HBM2存储器堆栈与PPU位于同一物理封装上,与传统的GDDR5 SDRAM系统相比,可提供大量功率并节省面积。在至少一个实施例中,每个HBM2堆栈包括但不限于四个存储器管芯,且Y等于4,每个HBM2堆栈包括每个管芯两个128位通道,用于总共8个通道和1024位的数据总线宽度。在至少一个实施例中,存储器支持单错误校正双错误检测(“SECD”)错误校正码(“ECC”)以保护数据。在至少一个实施例中,ECC为对数据损坏敏感的计算应用程序提供更高的可靠性。

[0327] 在至少一个实施例中,PPU实现了多级存储器层次结构。在至少一个实施例中,存储器分区单元2900支持统一存储器以为中央处理单元(“CPU”)和PPU存储器提供单个统一虚拟地址空间,从而实现虚拟存储器系统之间的数据共享。在至少一个实施例中,追踪PPU对位于其他处理器上的存储器的访问频率,以确保将存储器页面移动到更频繁地访问页面的PPU的物理存储器。在至少一个实施例中,高速GPU互连2708支持地址转换服务,其允许PPU直接访问CPU的页表,并通过PPU提供对CPU存储器的完全访问。

[0328] 在至少一个实施例中,复制引擎在多个PPU之间或PPU与CPU之间传输数据。在至少一个实施例中,复制引擎可以为未被映射到页表中的地址生成页面错误,并且存储器分区

单元2900然后为页面错误提供服务,将地址映射到页表中,之后复制引擎执行传输。在至少一个实施例中,为多个处理器之间的多个复制引擎操作固定(即不可分页)存储器,从而实质上减少了可用存储器。在至少一个实施例中,在硬件页面故障的情况下,可以将地址传递给复制引擎,而无需考虑是否驻留存储器页,并且复制过程是透明的。

[0329] 根据至少一个实施例,来自图27的存储器2704或其他系统存储器的数据由存储器分区单元2900获取,并将其存储在L2高速缓存2904中,L2高速缓存2904位于芯片上并且在各种GPC之间共享。在至少一个实施例中,每个存储器分区单元2900包括但不限于与对应的存储器设备相关联的L2高速缓存的至少一部分。在至少一个实施例中,在GPC内的各个单元中实现较低级别的高速缓存。在至少一个实施例中,每个SM 2814可以实现一级(“L1”)高速缓存,其中L1高速缓存是专用于特定SM 2814的私有存储器,并且从L2高速缓存2904中获取数据并将其存储在每个L1高速缓存中,用于在SM 2814的功能单元中进行处理。在至少一个实施例中,L2高速缓存2904耦合到存储器接口2906和XBar 2720。

[0330] 在至少一个实施例中,ROP单元2902执行与像素颜色有关的图形光栅操作,诸如颜色压缩、像素混合等。在至少一个实施例中,ROP单元2902结合光栅引擎2808实施深度测试,从光栅引擎2808的剔除引擎接收与像素片段相关联的样本位置的深度。在至少一个实施例中,针对在与片段关联的样本位置的深度缓冲区中的相应深度测试深度。在至少一个实施例中,如果该片段通过了针对该样本位置的该深度测试,则ROP单元2902更新深度缓冲区,并将该深度测试的结果发送给光栅引擎2808。将意识到,分区单元2900的数量可以不同于GPC的数量,因此,可以在至少一个实施例中将每个ROP单元2902耦合到每个GPC。在至少一个实施例中,ROP单元2902追踪从不同GPC接收到的分组,并且确定ROP单元2902生成的结果通过XBar 2720被路由到哪个。

[0331] 图30示出了根据至少一个实施例的流式多处理器(“SM”)3000。在至少一个实施例中,SM 3000是图28的SM 2814。在至少一个实施例中,SM 3000包括但不限于指令高速缓存3002;一个或多个调度器单元3004;寄存器文件3008;一个或多个处理核心(“核心”)3010;一个或多个特殊功能单元(“SFU”)3012;一个或多个加载/存储单元(“LSU”)3014;互连网络3016;共享存储器/一级(“L1”)高速缓存3018;和其任何合适的组合。在至少一个实施例中,工作分配单元调度任务以在并行处理单元(“PPU”)的通用处理集群(“GPC”)上执行,并且每个任务被分配给GPC内部的特定数据处理集群(“DPC”),并且如果任务与着色器程序相关联,则将该任务分配给SM 3000之一。在至少一个实施例中,调度器单元3004从工作分配单元接收任务并管理分配给SM 3000的一个或多个线程块的指令调度。在至少一个实施例中,调度器单元3004调度线程块以作为并行线程的线程束来执行,其中每个线程块被分配至少一个线程束。在至少一个实施例中,每个线程束执行线程。在至少一个实施例中,调度器单元3004管理多个不同的线程块,将线程束分配给不同的线程块,然后在每个时钟周期内将来自多个不同的协作组的指令分派给各种功能单元(例如,处理核心3010、SFU 3012和LSU 3014)。

[0332] 在至少一个实施例中,协作组可以指用于组织通信线程组的编程模型,其允许开发者表达线程正在通信的粒度,从而能够表达更丰富、更有效的并行分解。在至少一个实施例中,协作启动API支持线程块之间的同步以执行并行算法。在至少一个实施例中,常规编程模型的应用程序提供了用于同步协作线程的单一、简单的构造:跨线程块的所有线程的

屏障(例如, `syncthreads()` 函数)。但是,在至少一个实施例中,程序员可以在小于线程块粒度的情形下来定义线程组,并在所定义的组内进行同步,以实现更高的性能、设计灵活性以及以集合组范围功能接口的形式实现软件重用。在至少一个实施例中,协作组使程序员能够以子块(即,小到单个线程)和多块粒度明确定义线程组,并执行集合操作,例如对协作组中的线程进行同步。在至少一个实施例中,该编程模型支持跨软件边界的干净组合,从而库和实用程序功能可以在其本地环境中安全地同步,而不必进行关于收敛的假设。在至少一个实施例中,协作组图元使协作并行的新图案成为可能,包括但不限于生产者-消费者并行,机会主义并行以及整个线程块网格上的全局同步。

[0333] 在至少一个实施例中,调度单元3006配置为将指令发送到功能单元中的一个或多个,并且调度器单元3004包括但不限于两个调度单元3006,该两个调度单元3006使得来自同一线程束的两个不同指令能够在每个时钟周期被调度。在至少一个实施例中,每个调度器单元3004包括单个调度单元3006或附加调度单元3006。

[0334] 在至少一个实施例中,每个SM 3000在至少一个实施例中包括但不限于寄存器文件3008,该寄存器文件3008为SM 3000的功能单元提供了一组寄存器。在至少一个实施例中,寄存器文件3008在每个功能单元之间划分,从而为每个功能单元分配寄存器文件3008的专用部分。在至少一个实施例中,寄存器文件3008在由SM 3000执行的不同线程束之间划分,并且寄存器文件3008为连接到功能单元的数据路径的操作数提供临时存储。在至少一个实施例中,每个SM 3000包括但不限于多个L个处理核心3010。在至少一个实施例中,SM 3000包括但不限于大量(例如128个或更多)不同的处理核心3010。在至少一个实施例中,每个处理核心3010在至少一个实施例中包括但不限于全管线、单精度、双精度和/或混合精度处理单元,其包括但不限于浮点算术逻辑单元和整数算术逻辑单元。在至少一个实施例中,浮点算术逻辑单元实现用于浮点算术的IEEE 754-2008标准。在至少一个实施例中,处理核心3010包括但不限于64个单精度(32位)浮点核心、64个整数核心、32个双精度(64位)浮点核心和8个张量核心。

[0335] 根据至少一个实施例,张量核心配置为执行矩阵运算。在至少一个实施例中,一个或多个张量核心包括在处理核心3010中。在至少一个实施例中,张量核心配置为执行深度学习矩阵算术,例如用于神经网络训练和推理的卷积运算。在至少一个实施例中,每个张量核心在 4×4 矩阵上操作并且执行矩阵乘法和累加运算 $D = A \times B + C$,其中A、B、C和D是 4×4 矩阵。

[0336] 在至少一个实施例中,矩阵乘法输入A和B是16位浮点矩阵,并且累加矩阵C和D是16位浮点或32位浮点矩阵。在至少一个实施例中,张量核心对16位浮点输入数据进行32位浮点累加运算。在至少一个实施例中,16位浮点乘法使用64个运算,并得到全精度乘积,然后使用32位浮点加法与其他中间乘积累加起来,以进行 $4 \times 4 \times 4$ 矩阵乘法。在至少一个实施例中,张量核心用于执行由这些较小元件构成的更大的二维或更高维度的矩阵运算。在至少一个实施例中,API(诸如CUDA9C++API)公开专门的矩阵加载、矩阵乘法和累加以及矩阵存储操作,以有效地使用来自CUDA-C++程序的张量核心。在至少一个实施例中,在CUDA级别,线程束级别接口假定跨越所有32个线程束线程的 16×16 大小的矩阵。

[0337] 在至少一个实施例中,每个SM 3000包括但不限于执行特殊功能(例如,属性评估、倒数平方根等)的M个SFU 3012。在至少一个实施例中,SFU 3012包括但不限于配置为遍历

分层树数据结构的树遍历单元。在至少一个实施例中, SFU 3012包括但不限于配置为执行纹理映射过滤操作的纹理单元。在至少一个实施例中, 纹理单元配置为从存储器中加载纹理映射(例如, 纹理像素的2D阵列)和采样纹理映射, 以产生采样的纹理值以供由SM 3000执行的着色器程序使用。在至少一个实施例中, 将纹理映射存储在共享存储器/L1高速缓存3018中。在至少一个实施例中, 根据至少一个实施例, 纹理单元使用mip映射(mip-maps)(例如, 细节级别不同的纹理映射)来实现纹理操作(诸如过滤操作)。在至少一个实施例中, 每个SM 3000包括但不限于两个纹理单元。

[0338] 在至少一个实施例中, 每个SM 3000包括但不限于实现共享存储器/L1高速缓存3018与寄存器文件3008之间的加载和存储操作的N个LSU 3014。在至少一个实施例中, 每个SM 3000包括但不限于互连网络3016, 其将每个功能单元连接到寄存器文件3008, 并且将LSU 3014连接到寄存器文件3008和共享存储器/L1高速缓存3018。在至少一个实施例中, 互连网络3016是交叉开关, 其可以配置为将任何功能单元连接到寄存器文件3008中的任何寄存器, 并且将LSU 3014连接到寄存器文件3008和共享存储器/L1高速缓存3018中的存储器位置。

[0339] 在至少一个实施例中, 共享存储器/L1高速缓存3018是片上存储器的阵列, 其在至少一个实施例中允许SM 3000与图元引擎之间以及SM3000中的线程之间的数据存储和通信。在至少一个实施例中, 共享存储器/L1高速缓存3018包括但不限于128KB的存储容量, 并且位于从SM 3000到分区单元的路径中。在至少一个实施例中, 共享存储器/L1高速缓存3018在至少一个实施例中用于高速缓存读取和写入。在至少一个实施例中, 共享存储器/L1高速缓存3018、L2高速缓存和存储器中的一个或多个是后备存储。

[0340] 在至少一个实施例中, 将数据高速缓存和共享存储器功能组合到单个存储器块中, 为两种类型的存储器访问提供了改进的性能。在至少一个实施例中, 容量由不使用共享存储器的程序使用或将其用作高速缓存, 例如如果共享存储器配置为使用一半容量, 并且纹理和加载/存储操作可以使用剩余容量。根据至少一个实施例, 在共享存储器/L1高速缓存3018内的集成使共享存储器/L1高速缓存3018能够用作用于流传输数据的高吞吐量管线, 同时提供对频繁重用的数据的高带宽和低延迟访问。在至少一个实施例中, 当配置用于通用并行计算时, 与图形处理相比, 可以使用更简单的配置。在至少一个实施例中, 绕过固定功能图形处理单元, 从而创建了更加简单的编程模型。在至少一个实施例中, 在通用并行计算配置中, 工作分配单元直接将线程的块分配和分布给DPC。在至少一个实施例中, 块中的线程执行同一程序, 在计算中使用唯一的线程ID以确保每个线程生成唯一的结果, 使用SM 3000执行程序并执行计算, 使用共享存储器/L1高速缓存3018在线程之间进行通信, 以及使用LSU 3014通过共享存储器/L1高速缓存3018和存储器分区单元来读写全局存储器。在至少一个实施例中, 当被配置用于通用并行计算时, SM 3000向调度器单元3004写入可以用来在DPC上启动新工作的命令。

[0341] 在至少一个实施例中, PPU被包括在台式计算机、膝上型计算机、平板电脑、服务器、超级计算机、智能电话(例如, 无线、手持设备)、个人数字助理(“PDA”)、数码相机、车辆、头戴式显示器、手持式电子设备等中或与之耦合。在至少一个实施例中, PPU被实现在单个半导体衬底上。在至少一个实施例中, PPU与一个或多个其他设备(例如附加的PPU、存储器、精简指令集计算机(“RISC”)CPU, 一个或多个存储器管理单元(“MMU”)、数模转换器

(“DAC”)等)一起被包括在片上系统(“SoC”)中。

[0342] 在至少一个实施例中,PPU可以被包括在包括一个或更多个存储设备的图形卡上。该图形卡可以配置为与台式计算机主板上的PCIe插槽相连接。在至少一个实施例中,该PPU可以是包括在主板的芯片组中的集成图形处理单元(“iGPU”)。

[0343] 推理和/或训练逻辑615用于执行与一个或更多个实施例相关的推理和/或训练操作。下面结合图6A和/或图6B提供关于推理和/或训练逻辑615的细节。在至少一个实施例中,深度学习应用程序处理器用于训练机器学习模型(诸如神经网络),以预测或推理提供给SM 3000的信息。在至少一个实施例中,SM 3000用于基于已由另一处理器或系统或由SM 3000训练过的机器学习模型(例如,神经网络)推理或预测信息。在至少一个实施例中,SM 3000可用于执行一个或更多个本文所述的神经网络用例。

[0344] 在至少一个实施例中,此类组件可用于编译与用户提供的函数相融合的计算受限运算。

[0345] 在至少一个实施例中,单个半导体平台可以指唯一的单一基于半导体的集成电路或芯片。在至少一个实施例中,可以使用具有增加的连接性的多芯片模块,其模拟芯片上的操作,并且相对于利用传统的中央处理单元(“CPU”)和总线实现方式进行了实质性的改进。在至少一个实施例中,根据用户的需求,各种模块也可以分开放置或以半导体平台的各种组合放置。

[0346] 在至少一个实施例中,机器可读的可执行代码或计算机控制逻辑算法形式的计算机程序被存储在主存储器1004和/或辅助存储中。根据至少一个实施例,如果由一个或更多个处理器执行,则计算机程序使系统1000能够执行各种功能。在至少一个实施例中,存储器1004、存储和/或任何其他存储是计算机可读介质的可能示例。在至少一个实施例中,辅助存储可以指代任何合适的存储设备或系统,例如硬盘驱动器和/或可移除存储驱动器,其代表软盘驱动器、磁带驱动器、光盘驱动器、数字多功能盘(“DVD”)驱动器、记录设备、通用串行总线(“USB”)闪存等。在至少一个实施例中,各个先前附图的架构和/或功能是在CPU 1002;并行处理系统1012;能够具有两个CPU 1002的至少部分能力的集成电路;并行处理系统1012;芯片组(例如,设计成作为执行相关功能的单元工作并出售的一组集成电路等);和集成电路的任何适当组合的环境中实现的。

[0347] 在至少一个实施例中,各个先前附图的架构和/或功能在通用计算机系统、电路板系统、专用于娱乐目的的游戏控制台系统、专用系统等的环境中实现。在至少一个实施例中,计算机系统1000可以采取台式计算机、膝上型计算机、平板电脑、服务器、超级计算机、智能电话(例如,无线、手持设备)、个人数字助理(“PDA”)、数码相机、车辆、头戴式显示器、手持式电子设备、移动电话设备、电视、工作站、游戏机、嵌入式系统和/或任何其他类型的逻辑的形式。

[0348] 在至少一个实施例中,并行处理系统1012包括但不限于多个并行处理单元(“PPU”)1014和相关联的存储器1016。在至少一个实施例中,PPU 1014经由互连1018和交换机1020或多路复用器连接到主机处理器或其他外围设备。在至少一个实施例中,并行处理系统1012在可并行化的PPU 1014上分配计算任务,例如,作为跨多个图形处理单元(“GPU”)线程块的计算任务分布的一部分。在至少一个实施例中,在PPU 1014中的一些或全部之间共享和访问存储器(例如,用于读取和/或写入访问),尽管这种共享存储器可能引发相对于

使用本地存储器和驻留在PPU1014上的寄存器的性能损失。在至少一个实施例中,通过使用命令(诸如__syncthreads())来同步PPU 1014的操作,其中块中的所有线程(例如,跨多个PPU 1014执行)在进行之前到达某个代码执行点。

[0349] 其他变化在本公开的精神内。由此,虽然所公开的技术易受不同修改和替代构造的影响,但其某些所解说的实施例在附图中示出并且已在上文详细地描述。然而,应当理解,并非旨在将本公开限制为所公开的一种或多种特定形式,而是相反,旨在覆盖落在如所附权利要求所限定的本公开的精神和范围内的所有修改、替代构造和等同物。

[0350] 在描述所披露的实施例的背景下(尤其在以下权利要求的背景下),术语“一个/一种(a/an)”和“该(the)”以及类似指称词的使用应被解释为覆盖单数和复数两者,除非在此另外指明或与上下文明显矛盾,并且不作为术语的定义。除非另外指出,术语“包含”、“具有”、“包括”和“含有”应被解释为开放式术语(意指“包括但不限于”)。术语“连接”在未修改和提及物理连接时应被解释为部分地或完全地包含在一起、附接至其上、或连接在一起,即使存在介入的东西。除非本文另有说明,本文列举的数值范围仅旨在用作单独提及落入范围内的每个单独数值的简略方法,并且每个单独数值并入说明书中,就好像其在本文单独列举一样。除非上下文另外指出或矛盾,否则术语“集合”(例如,“项目集合”)或“子集”的使用应被解释为包括一个或更多个成员的非空集合。进一步,除非上下文另外指出或矛盾,否则对应集合的术语“子集”不一定表示对应集合的适当子集,但是子集和对应集合可以是相等的。

[0351] 连接语言,如“A、B和C中的至少一项”形式的短语,“或“A、B和C中的至少一个”,除非另有明确说明或者与上下文明显矛盾,用上下文另外理解,所述上下文通常用于呈现项目、术语,等可以是A或B或C,或者A和B和C的集合的任何非空子集。例如,在具有三个成员的集合的说明性示例中,连接性短语“A、B和C中的至少一个”和“A、B和C中的至少一个”是指以下集合中的任何集合:{A}、{B}、{C}、{A、B}、{A、C}、{B、C}、{A、B、C}。因此,此类连接性语言一般不旨在暗示某些实施例要求各自存在A中的至少一个、B中的至少一个、以及C中的至少一个。此外,除非上下文另外指出或矛盾,否则术语“多个”指示为复数的状态(例如,“多个项目”指示多个项目)。多个是至少两个项目,但当明确地或通过上下文如此指示时可能更多。进一步地,除非另外指明或另外从上下文清楚,短语“基于”是指“至少部分地基于”而不是“仅基于。”

[0352] 在此描述的过程的操作可以按任何适合的顺序进行,除非在此另外指明或另外与上下文明显矛盾。在至少一个实施例中,诸如本文中描述的那些过程(或其变型和/或组合)的过程在配置有可执行指令的一个或更多个计算机系统的控制下执行,并被实现为在一个或更多个处理器上共同执行的代码(例如,可执行指令、一个或更多个计算机程序或一个或更多个应用),通过硬件或其组合。在至少一个实施例中,代码例如以包括可由一个或更多个处理器执行的多个指令的计算机程序的形式存储在计算机可读存储介质上。在至少一个实施例中,计算机可读存储介质是非瞬态计算机可读存储介质,该非瞬态计算机可读存储介质排除瞬态信号(例如,传播的瞬态电或电磁传输)但包括瞬态信号的收发机内的非瞬态数据存储电路(例如,缓冲区、高速缓存和队列)。在至少一个实施例中,代码(例如,可执行代码或源代码)被存储在其上存储有可执行指令(或用于存储可执行指令的其他存储器)的一个或更多个非瞬态计算机可读存储介质的集合上,当由计算机系统的一个或更多个处理

器执行时(即,由于被执行),使计算机系统执行本文描述的操作。在至少一个实施例中,一组非暂态计算机可读存储介质包括多个非暂态计算机可读存储介质和多个非暂态计算机可读存储介质中的各个非暂态计算机可读存储介质中的一个或更多个缺少全部代码,而多个非暂态计算机可读存储介质共同地存储全部代码。在至少一个实施例中,可执行指令被执行以使得不同的指令由不同的处理器执行。例如,非瞬态计算机可读存储介质存储指令,并且主中央处理单元(“CPU”)执行指令中的一些指令,而图形处理单元(“GPU”)执行其他指令。在至少一个实施例中,计算机系统的不同组件具有单独的处理器,并且不同的处理器执行指令的不同子集。

[0353] 因此,在至少一个实施例中,计算机系统被配置为实现单独地或共同地执行本文中描述的过程的操作的一个或更多个服务,并且此类计算机系统配置有能够实现操作的执行的可应用的硬件和/或软件。进一步,实现本公开的至少一个实施例的计算机系统是单个设备,并且在另一实施例中,是包括不同地操作的多个设备的分布式计算机系统,使得分布式计算机系统执行本文描述的操作并且使得单个设备不执行所有操作。

[0354] 在此提供的任何和所有示例或示例性语言(例如,“如”)的使用仅旨在更好地阐明本披露的实施例并且不对本披露的范围构成限制,除非另外要求。说明书中的语言不应被解释为指示任何未要求保护的元素对于实践本公开是必不可少的。

[0355] 在此引用的所有参考文献(包括出版物、专利申请、以及专利)均通过引用结合在此,其程度如同每个参考文献被单独地并且具体地指示为通过引用结合并且在此以其全文列出一样。

[0356] 在说明书和权利要求书中,可以使用术语“耦合”和“连接”及其派生词。应当理解,这些术语可能不旨在作为彼此的同义词。而是,在具体示例中,“连接”或“耦合”可以用于指示两个或更多个元件彼此直接或间接物理或电接触。“耦合”还可以指两个或更多个元件彼此不直接接触,但仍彼此合作或交互。

[0357] 除非另有明确说明,可以理解的是贯穿说明书术语,如“处理”、“计算”、“计算”、“确定”等指计算机或计算系统的动作和/或过程,或类似的电子计算设备,其操纵和/或变换被表示为物理的(诸如电子的)数据,计算系统的寄存器和/或存储器内的量到类似地表示为计算系统的存储器内的物理量的其他数据中,寄存器或其他此类信息存储、传输或显示装置。

[0358] 以类似的方式,术语“处理器”可以指处理来自寄存器和/或存储器的电子数据并且将该电子数据转换成可以存储在寄存器和/或存储器中的其他电子数据的任何设备或设备的部分。作为非限制性示例,“处理器”可以是CPU或GPU。“计算平台”可以包括一个或更多个处理器。如在此使用的,“软件”过程可以包括例如随着时间执行工作的软件和/或硬件实体,如任务、线程和智能代理。而且,每一过程可指代用于循序或并行地、连续地或间歇地执行指令的多个过程。术语“系统”和“方法”在此可互换地使用,只要系统可以体现一种或多种方法并且方法可以被认为是系统。

[0359] 在本文件中,可以参考获得、获取、接收、或输入模拟或数字数据到子系统、计算机系统、或计算机实现的机器中。获得、获取、接收或输入模拟和数字数据可以以各种方式完成,如通过接收作为函数调用或对应用程序编程接口的调用的参数的数据。在一些实现中,获得、获取、接收或输入模拟或数字数据的过程可通过经由串行或并行接口传送数据来完

成。在另一实现中,获得、获取、接收或输入模拟或数字数据的过程可通过经由计算机网络从提供实体向获取实体传送数据来完成。还可以参考提供、输出、传输、发送或呈现模拟或数字数据。在不同示例中,提供、输出、传送、发送或呈现模拟或数字数据的过程可通过作为函数调用的输入或输出参数、应用编程接口的参数或进程间通信机制来传送数据来完成。

[0360] 虽然以上论述陈述所描述的技术的举例实施例,但其他架构可用以实施所描述的功能性,且既定处于本发明的范围内。此外,尽管以上出于讨论的目的定义了职责的特定分布,但是取决于情况,可以以不同的方式分布和划分各种功能和职责。

[0361] 此外,尽管已经用结构特征和/或方法动作专用的语言描述了本主题,但应当理解的是,所附权利要求书中所要求保护的主题不必限于所描述的具体特征或动作。相反,特定特征和动作被公开为实现权利要求的示范性形式。

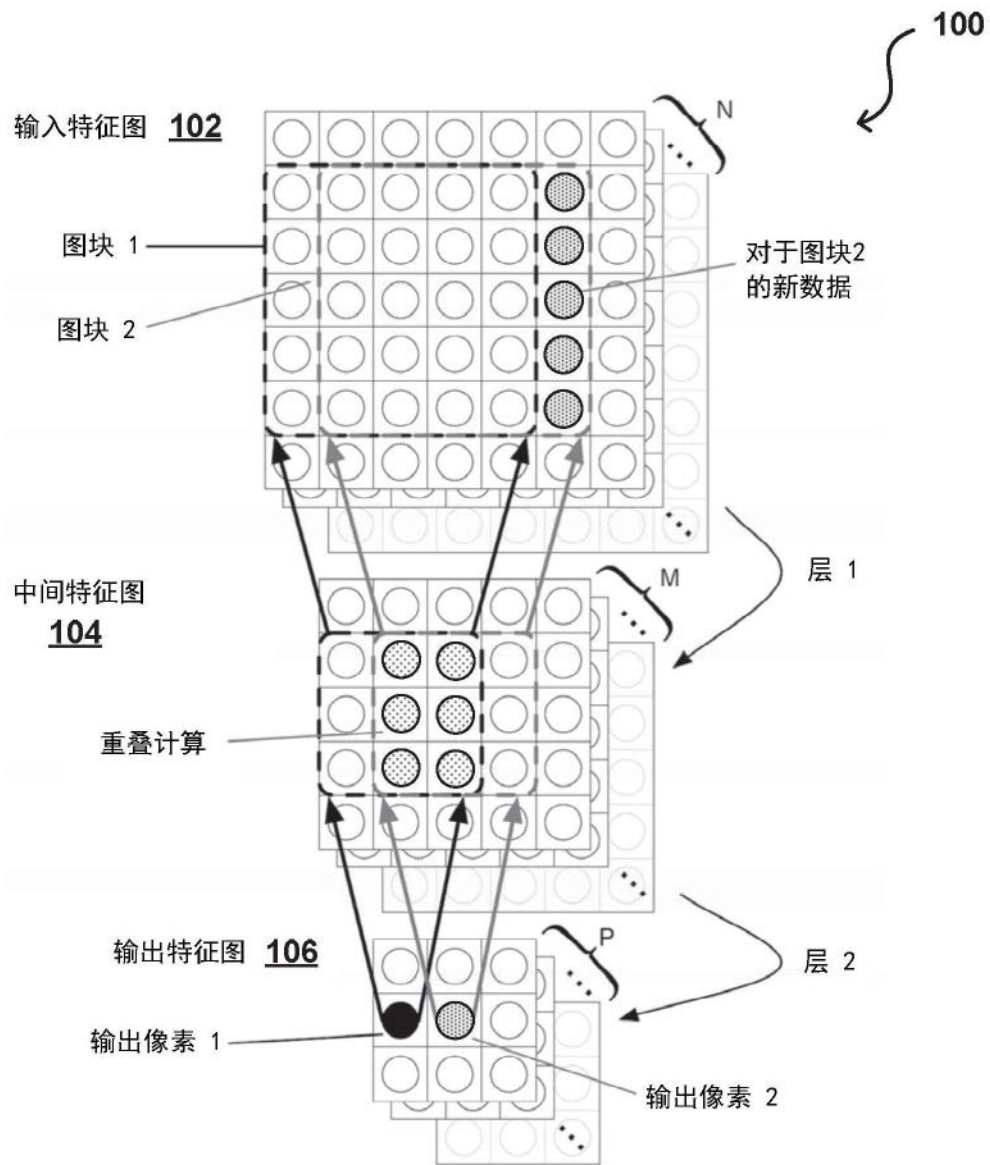


图1

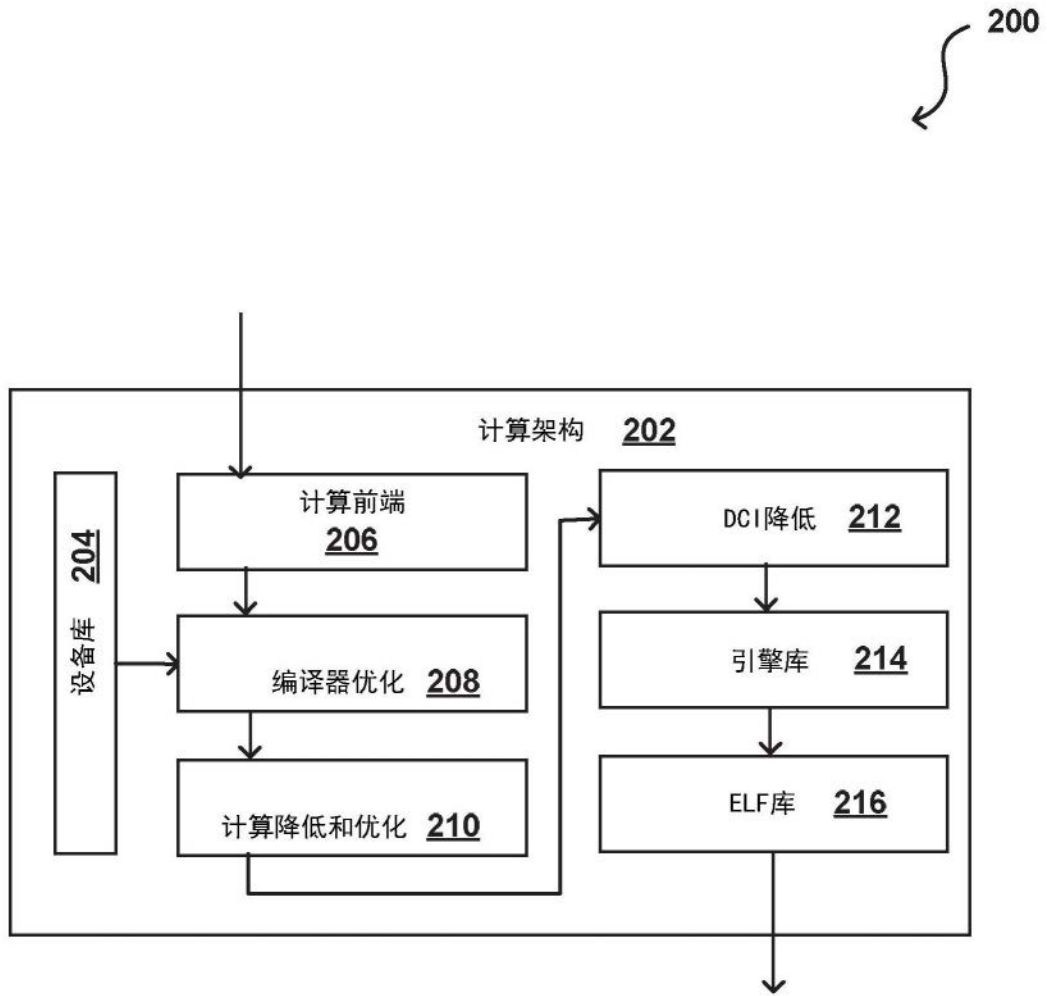


图2

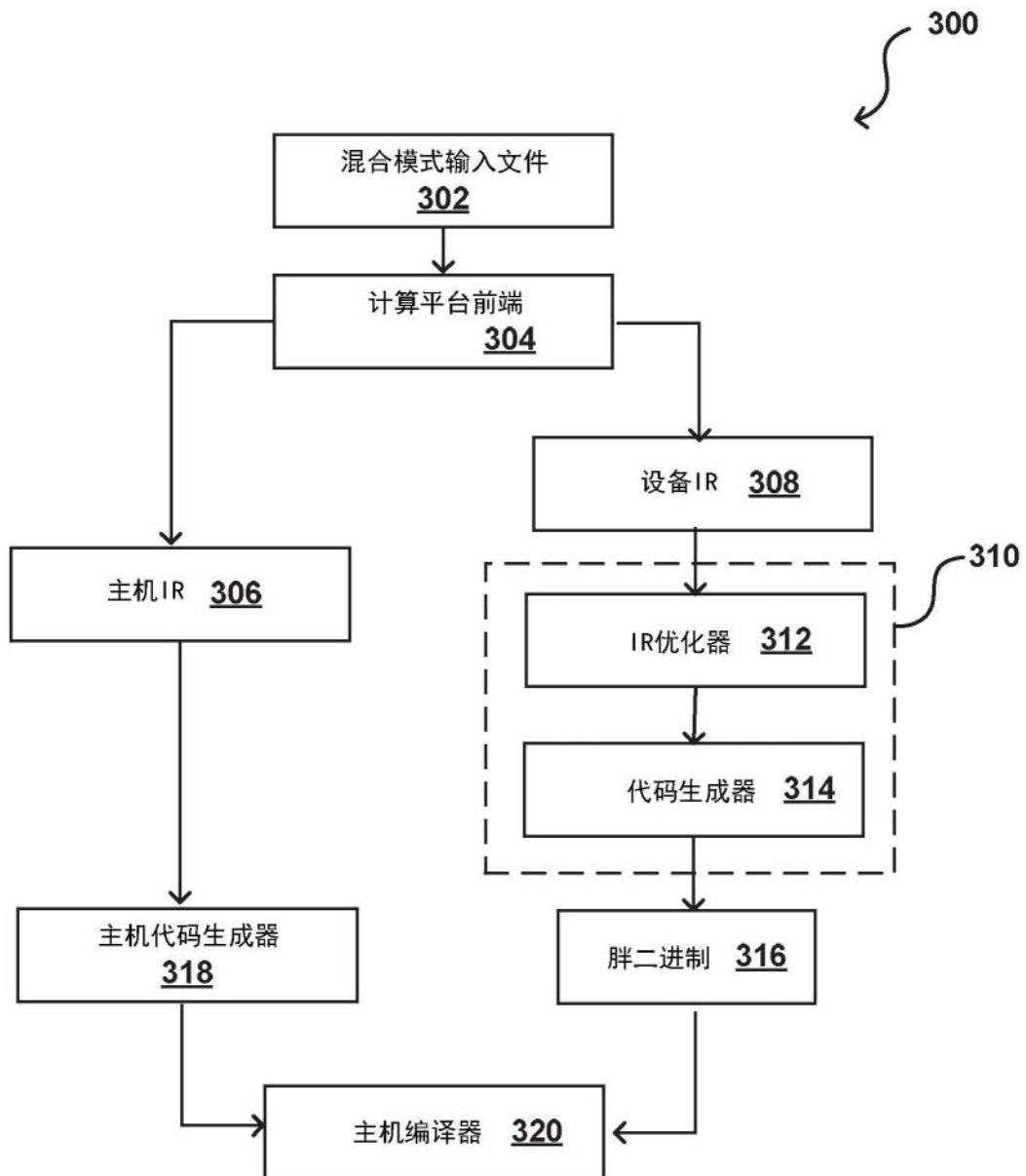


图3

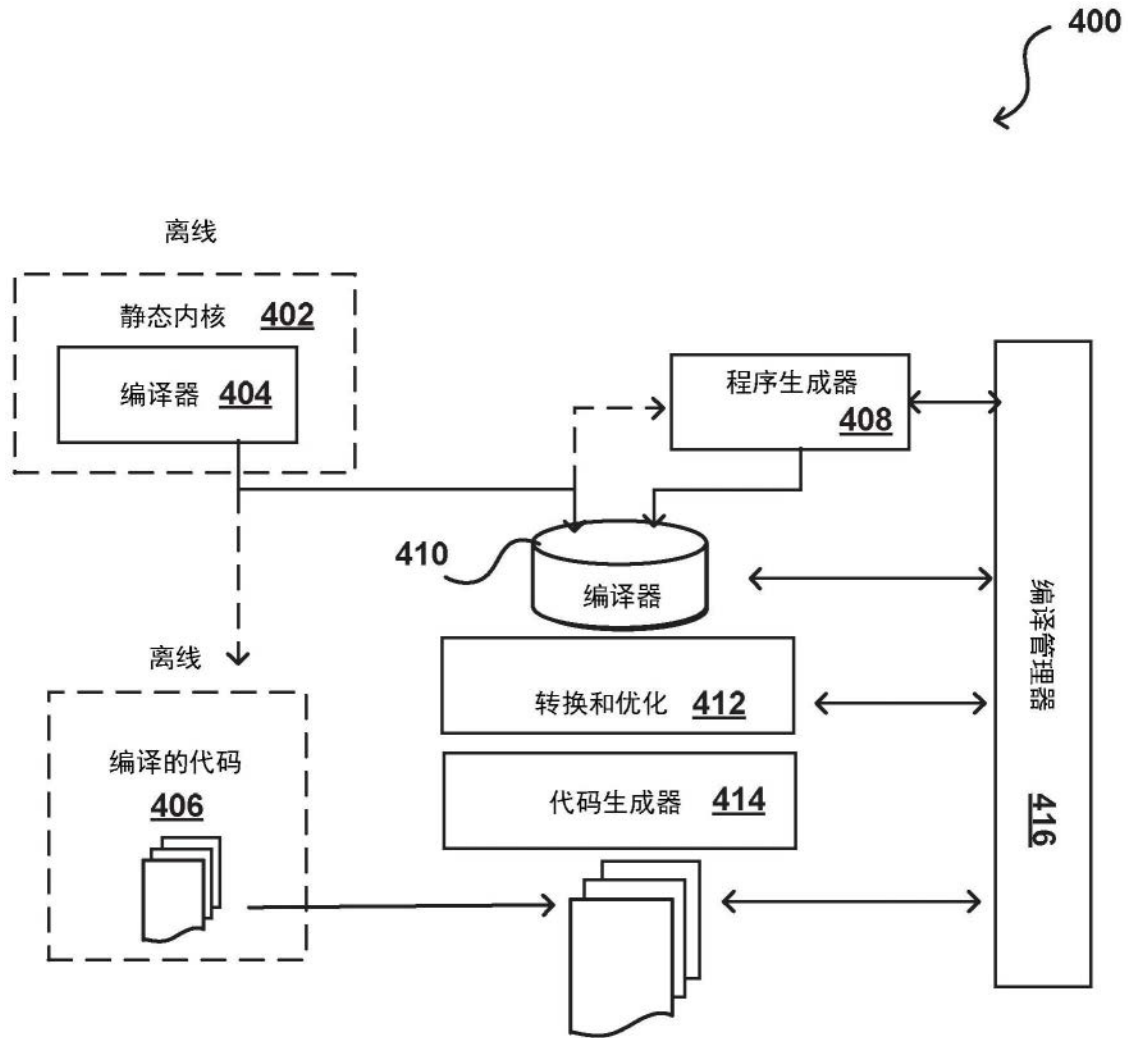


图4

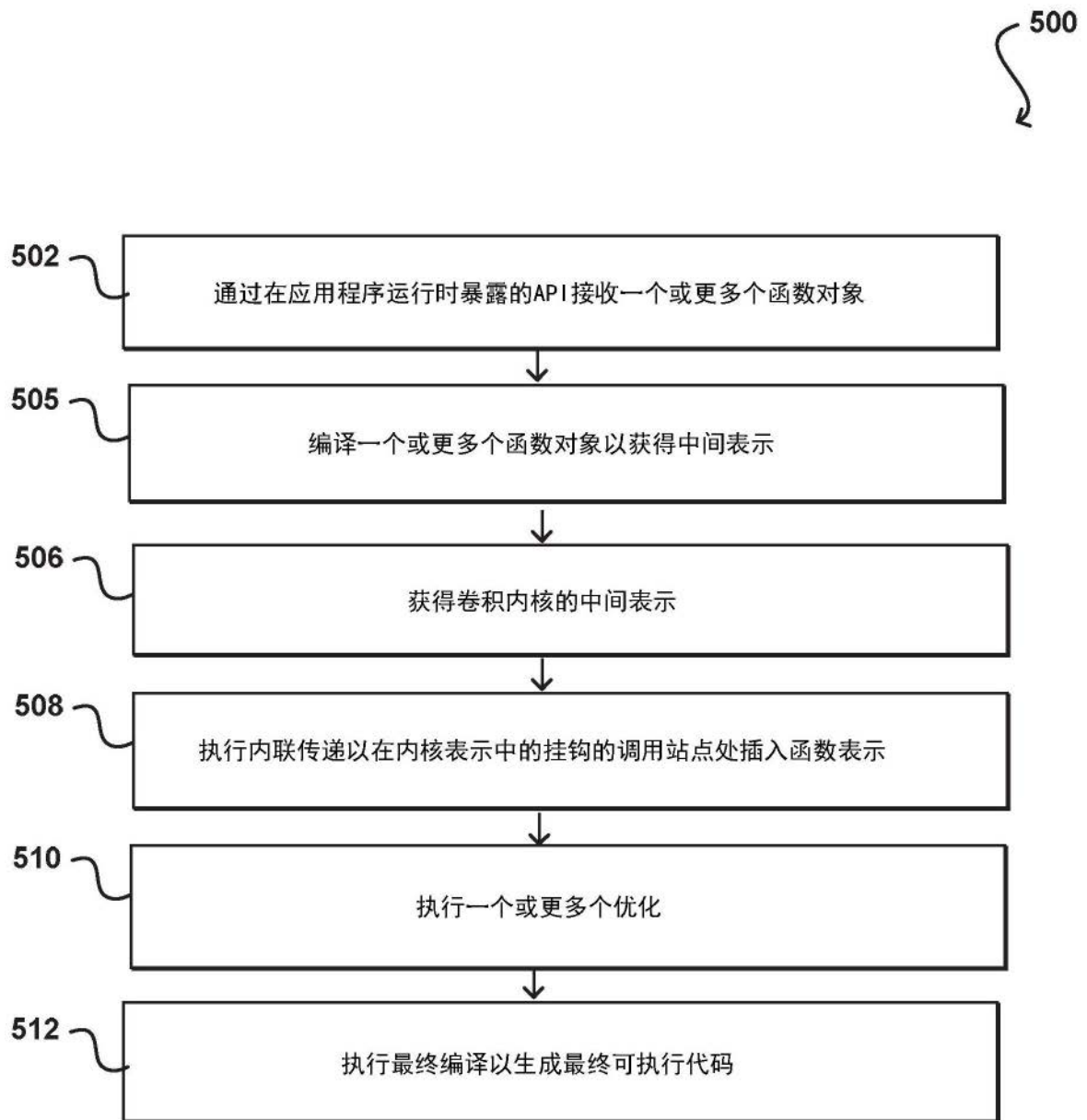


图5

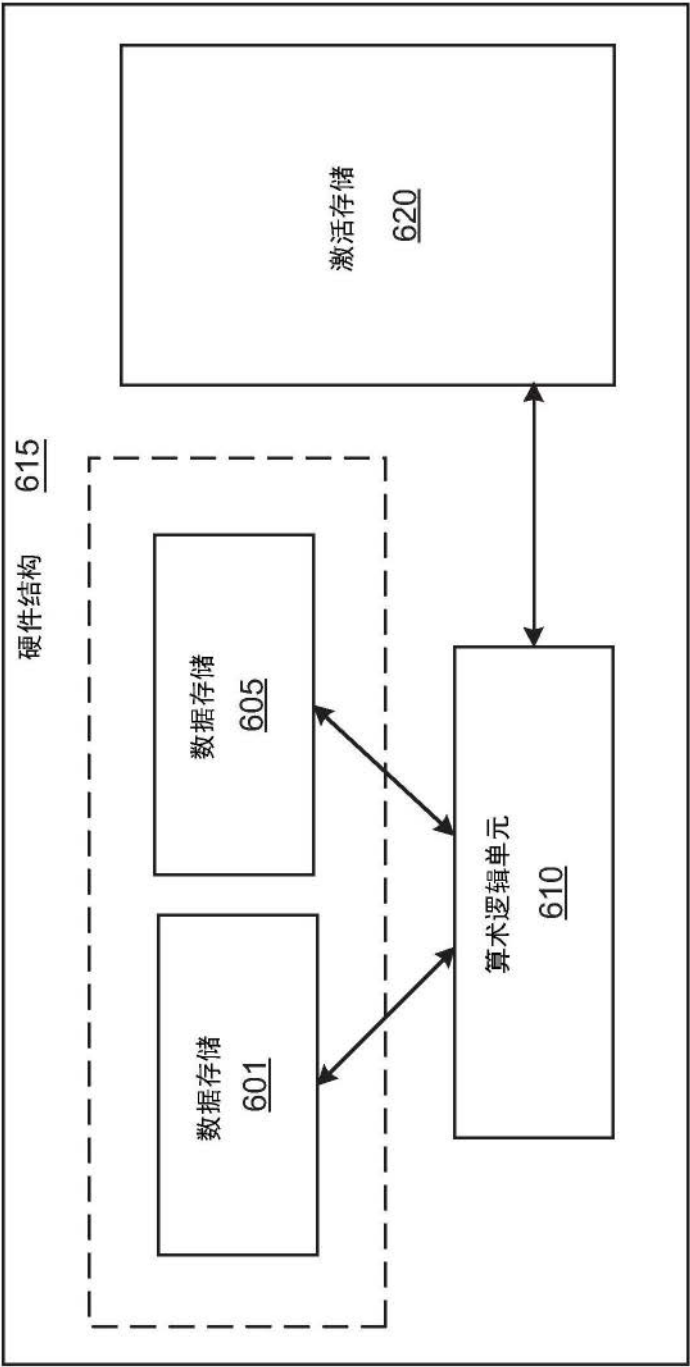


图6A

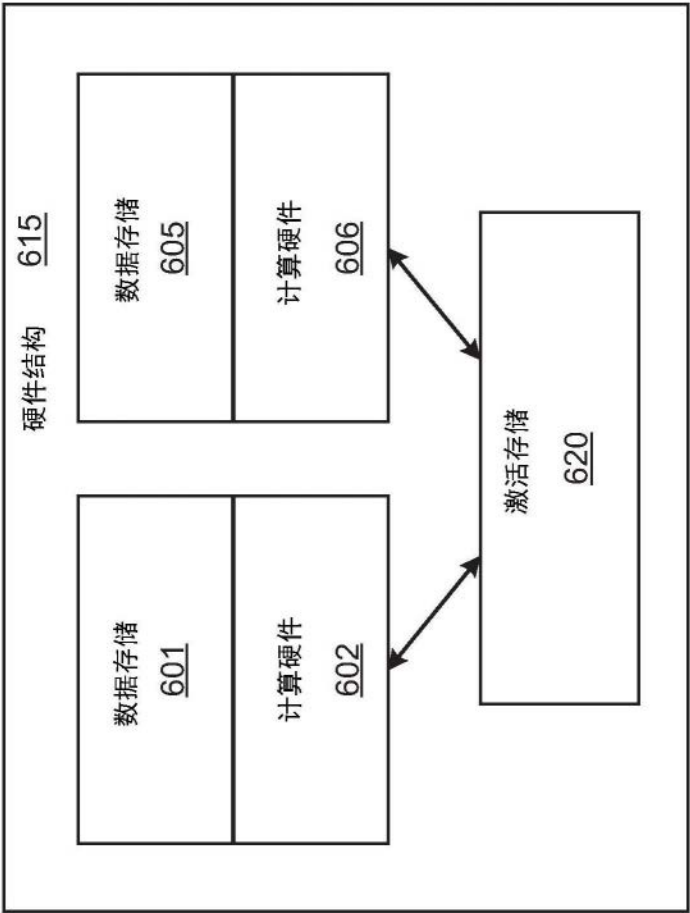


图6B

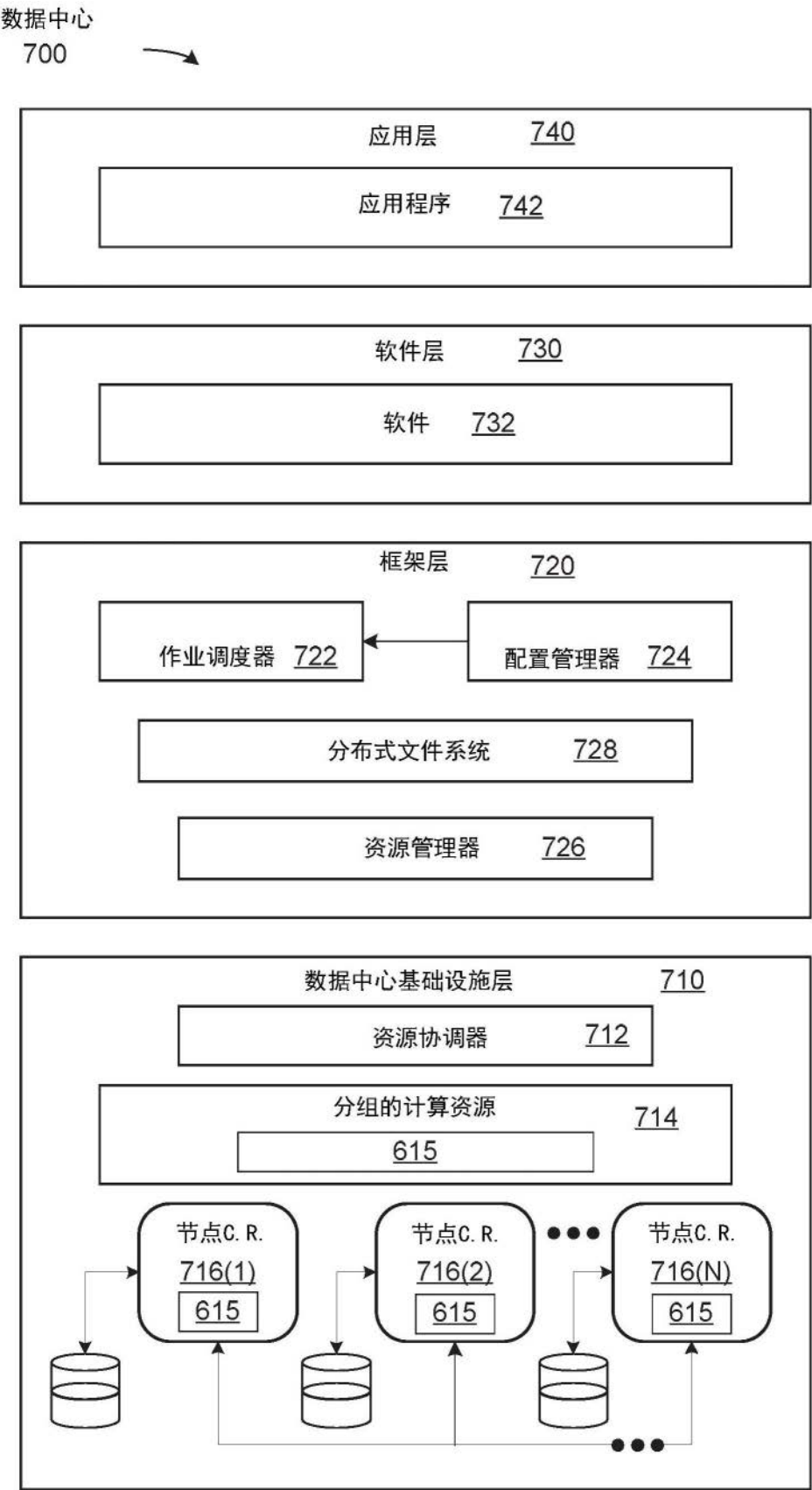


图7

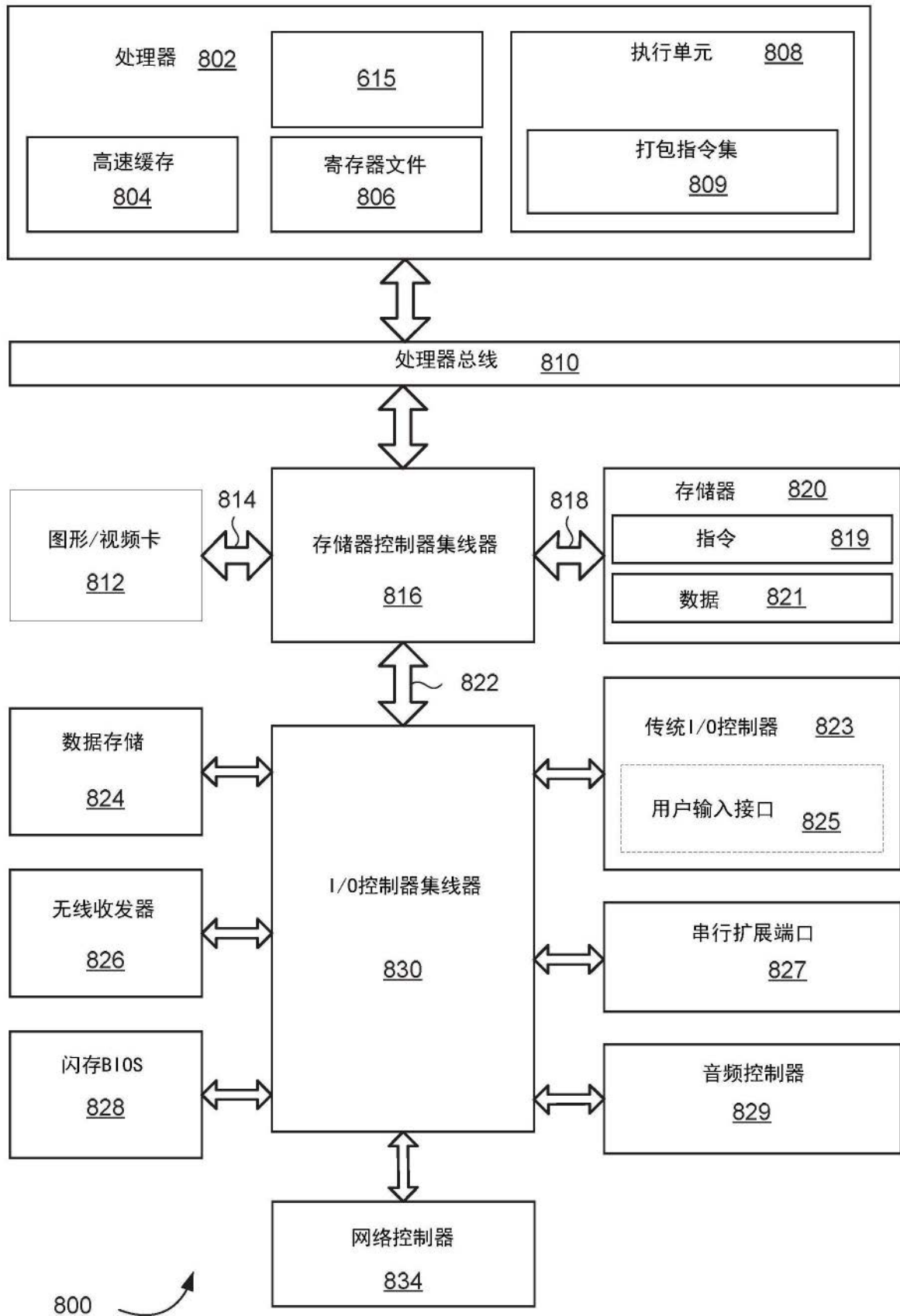


图8

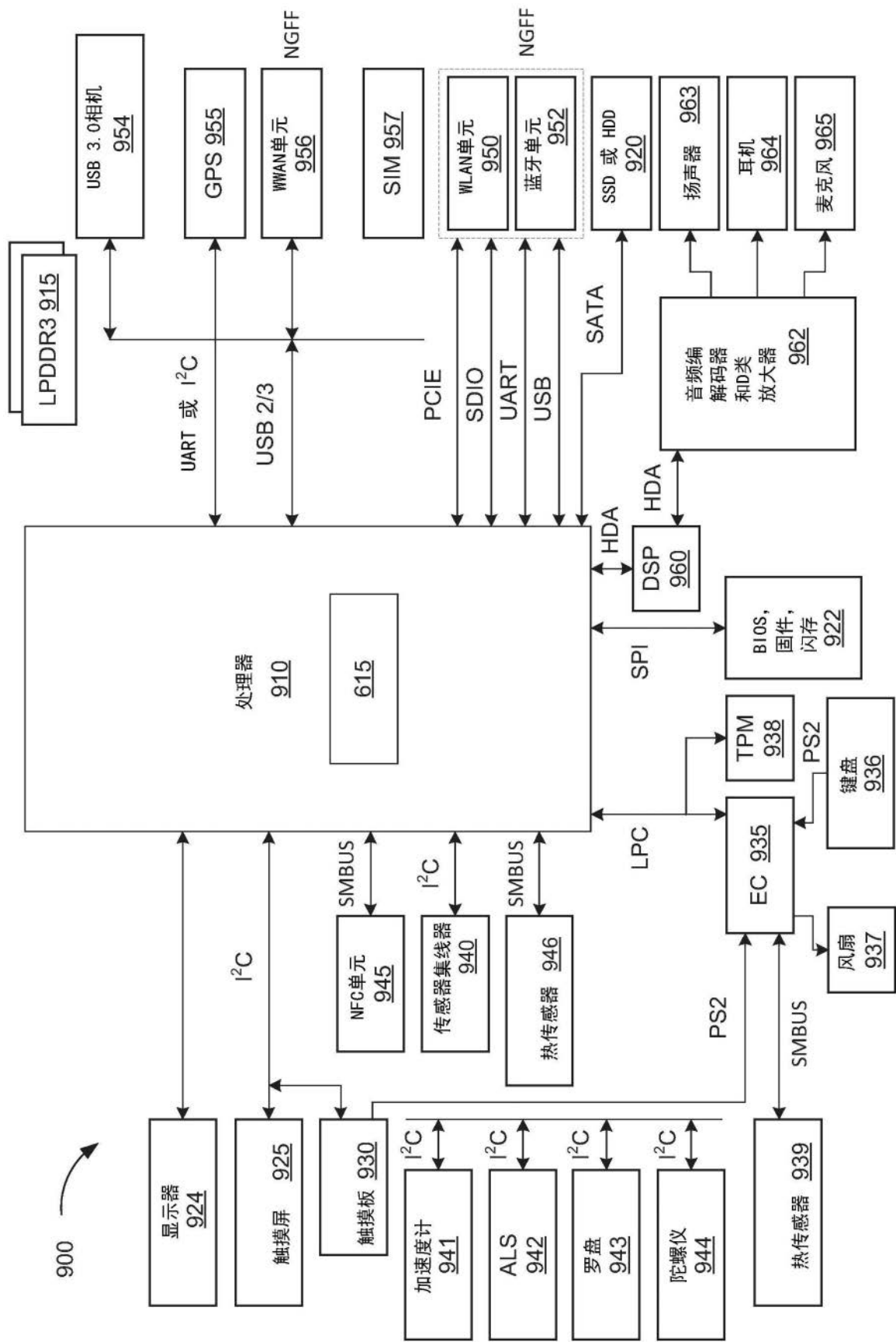


图9

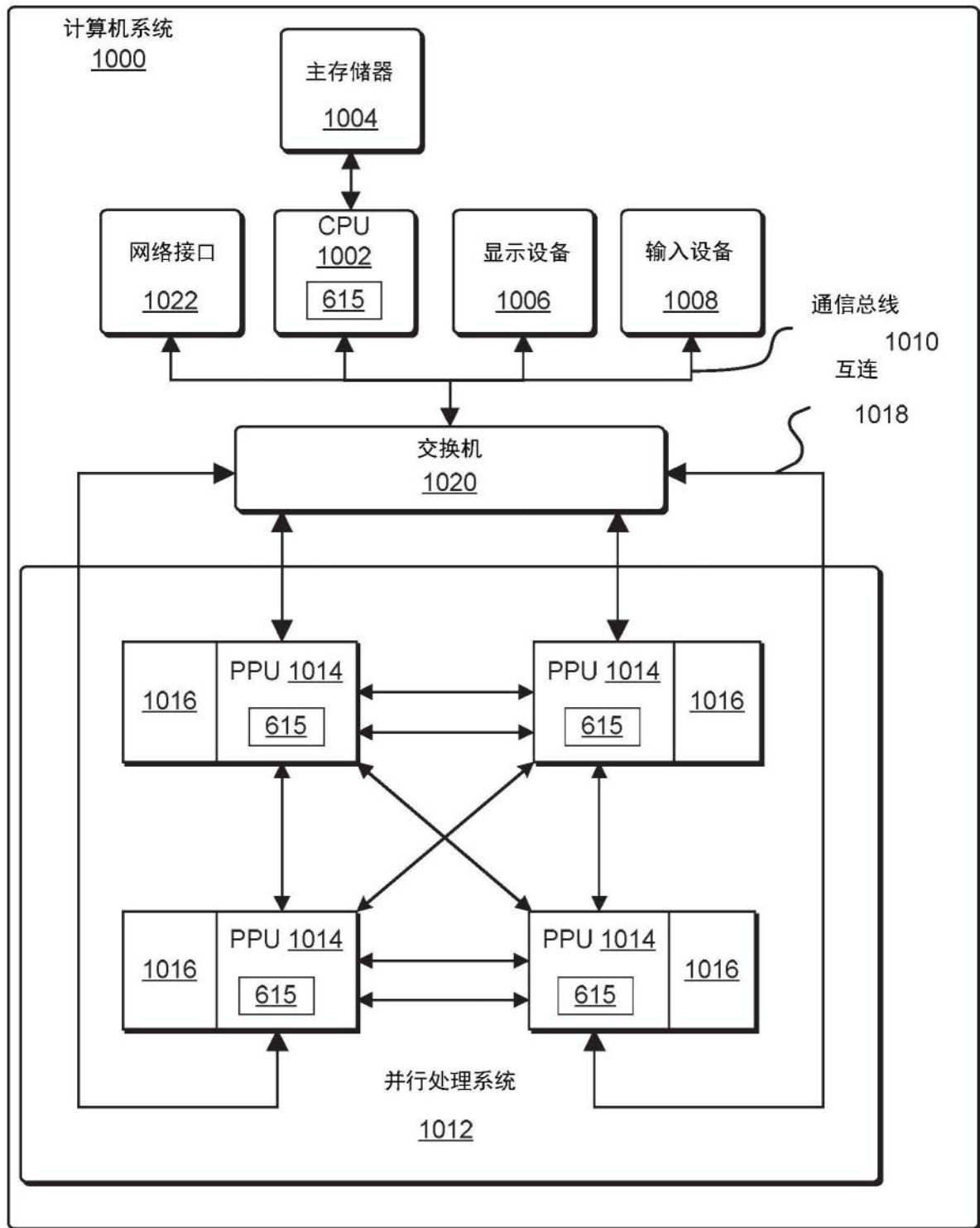


图10

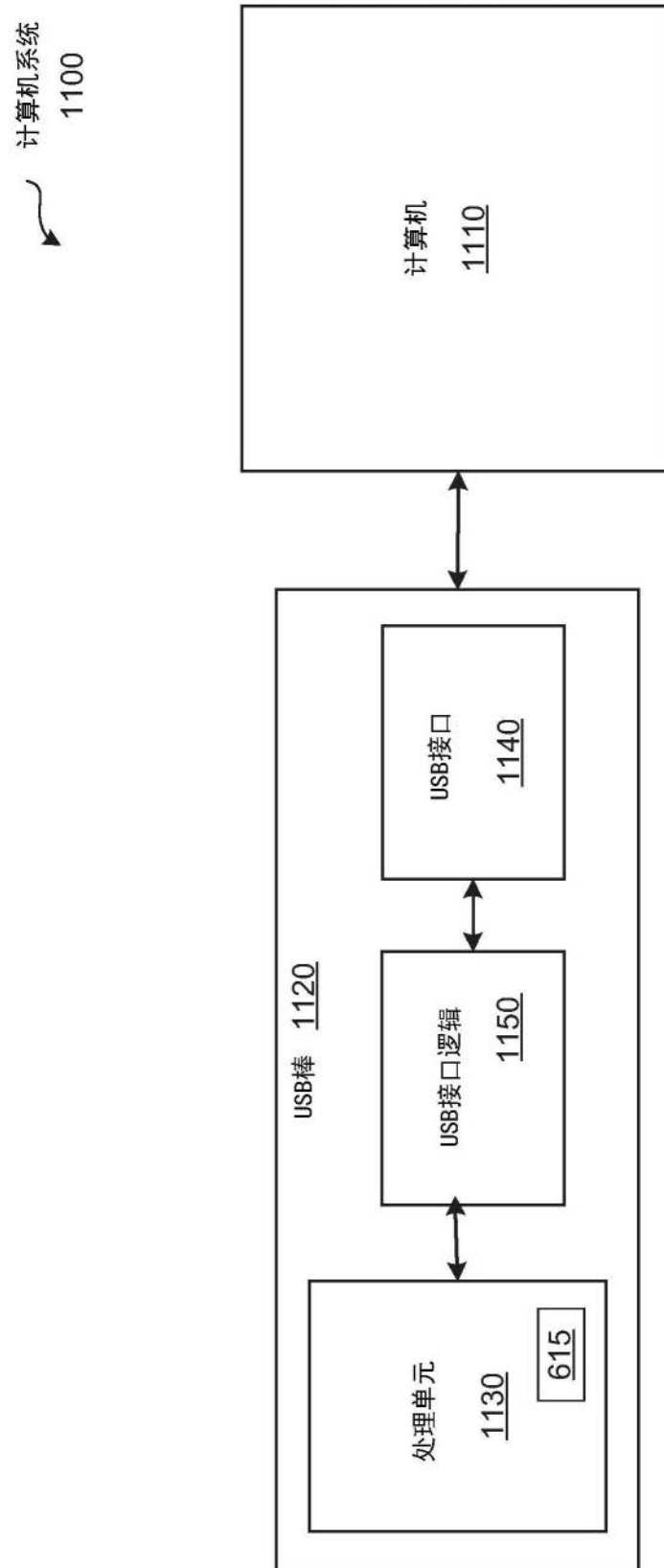


图11

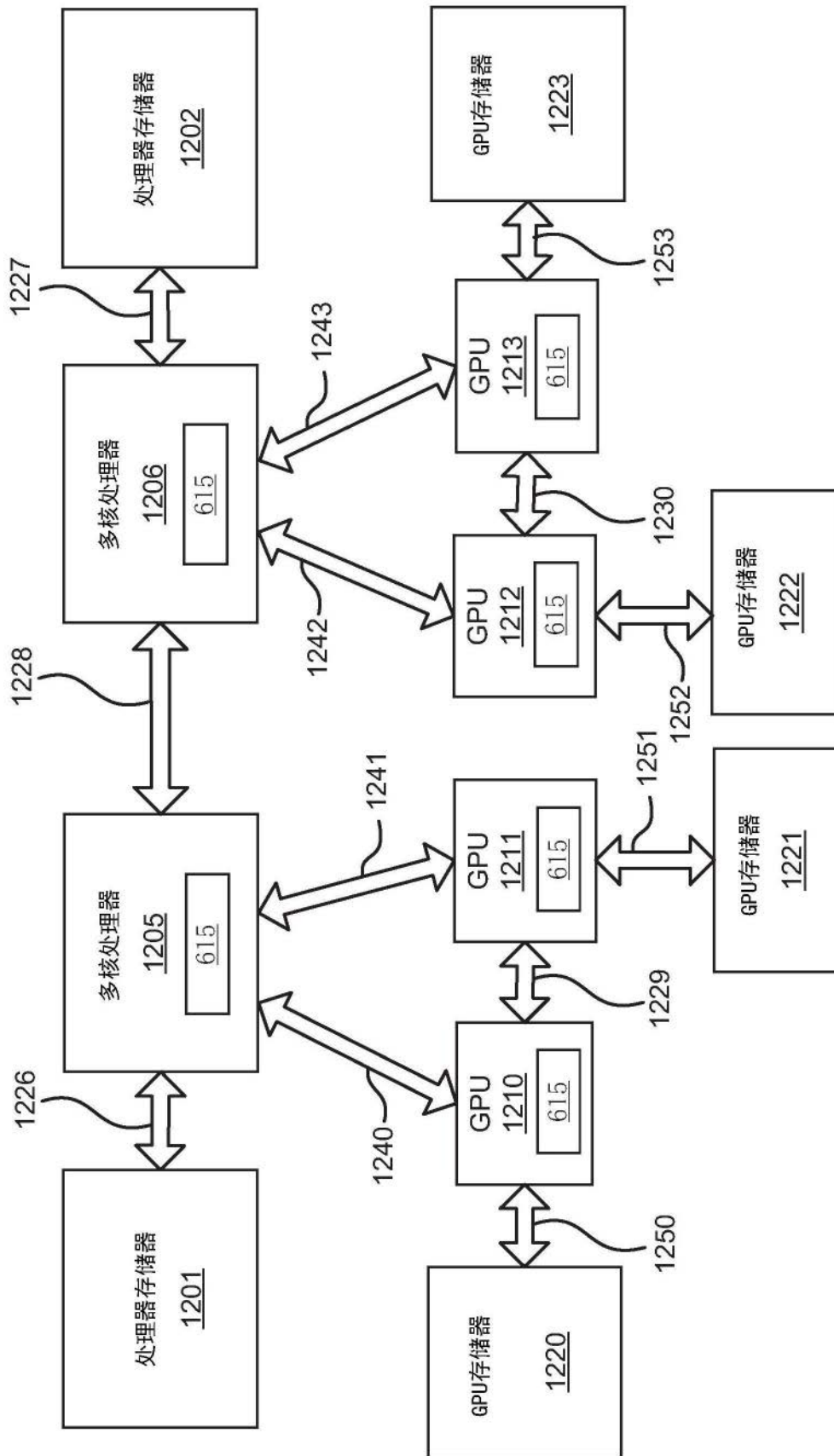


图12A

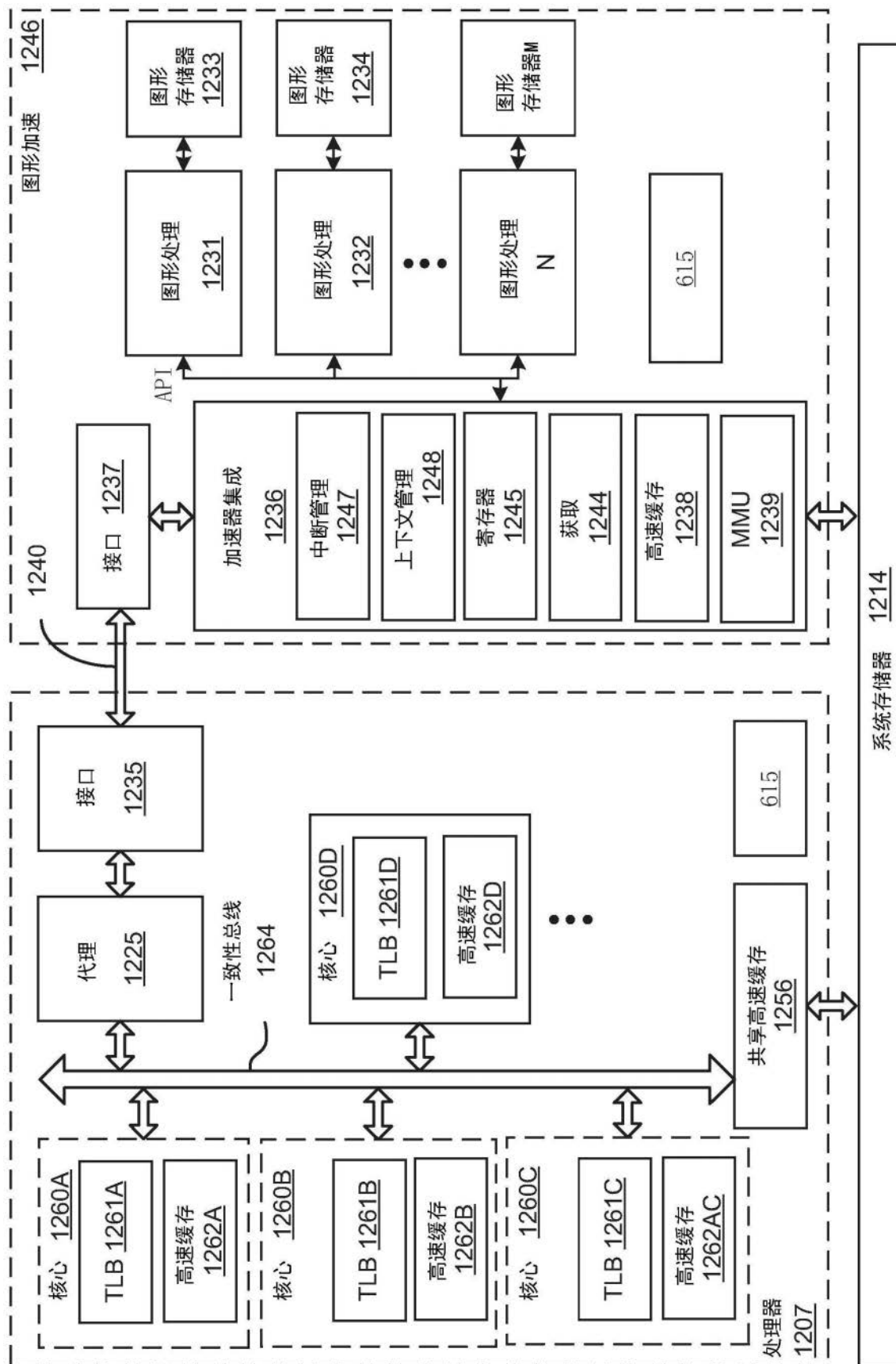


图12B

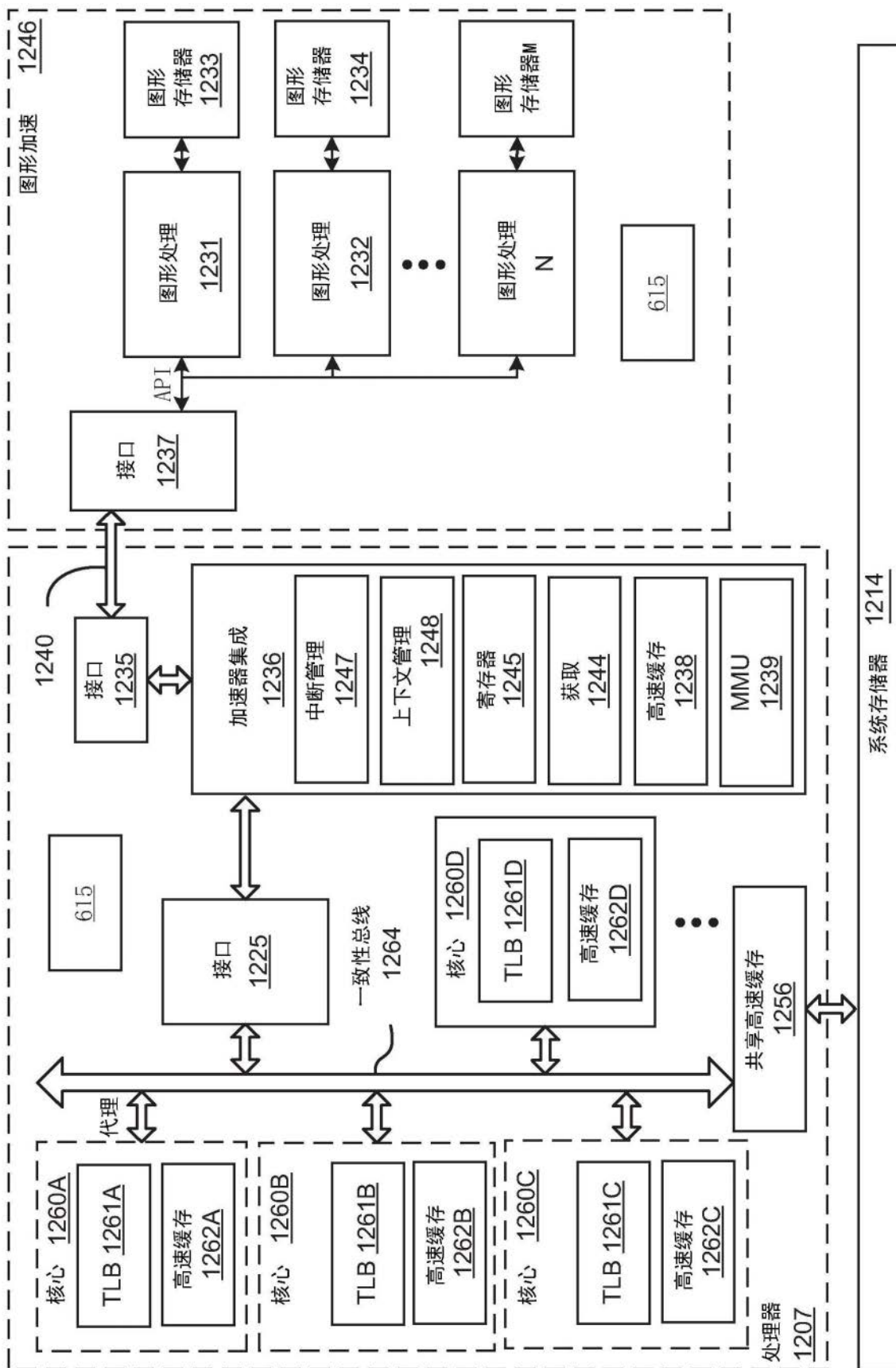


图12C

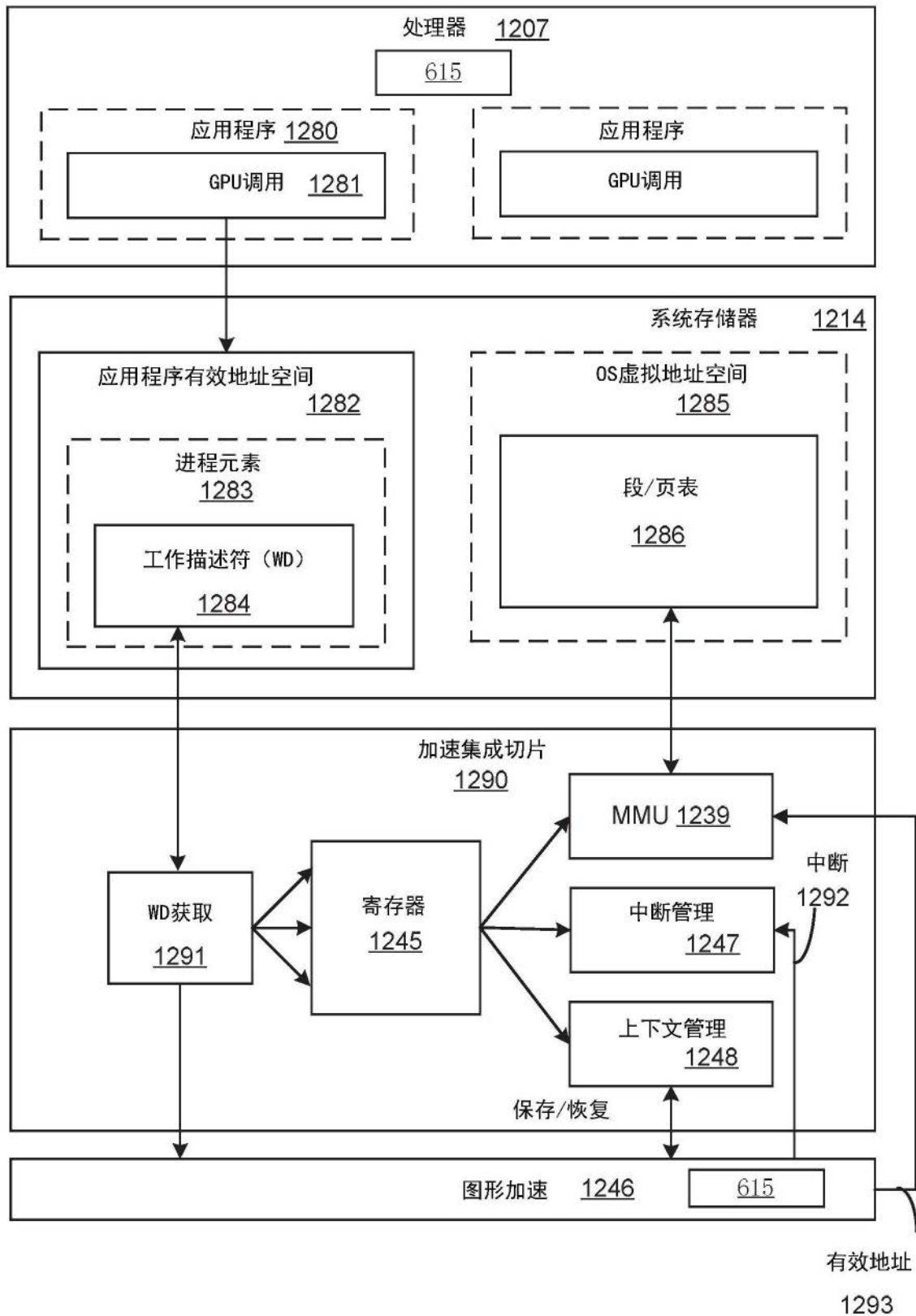


图12D

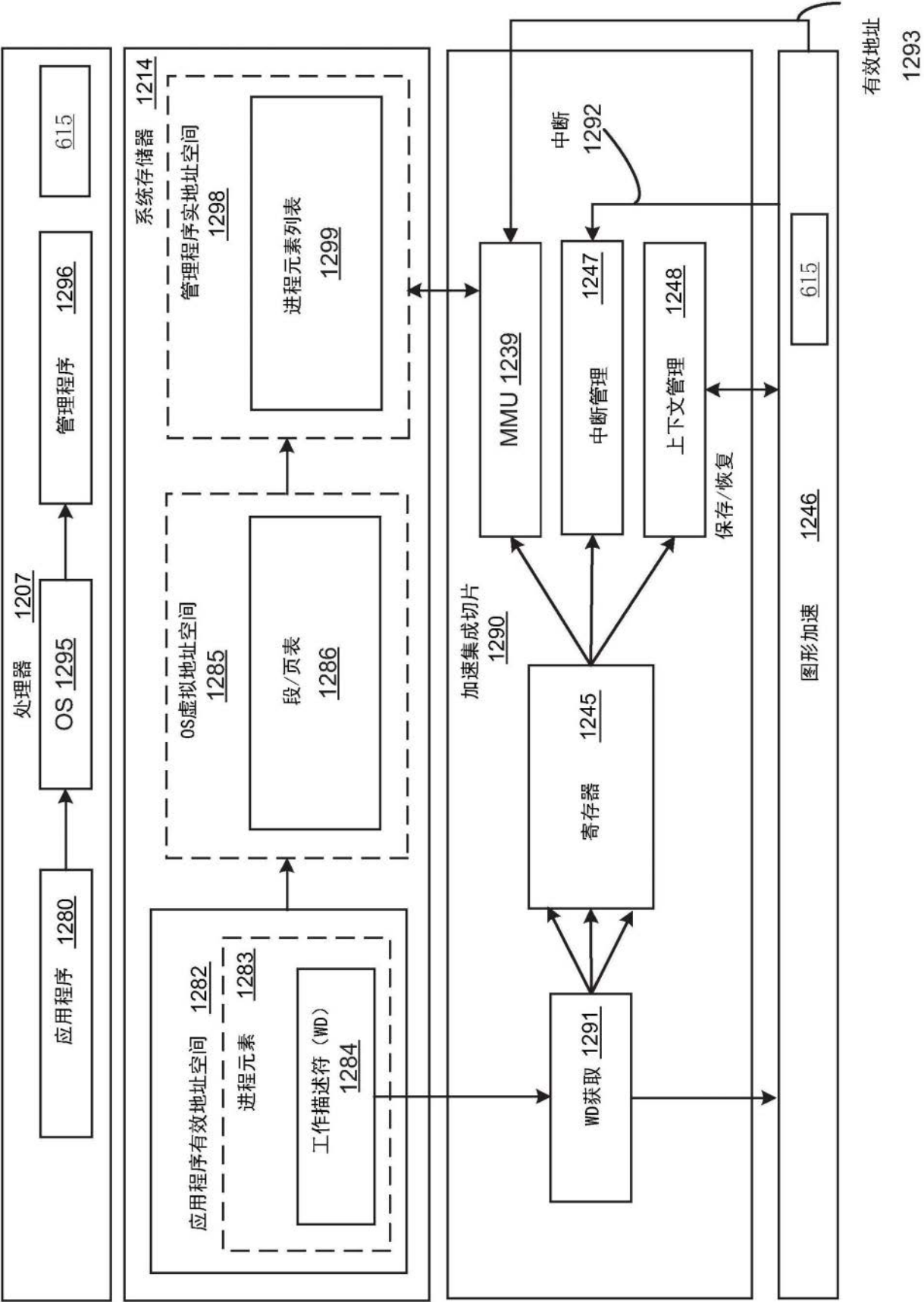


图12E

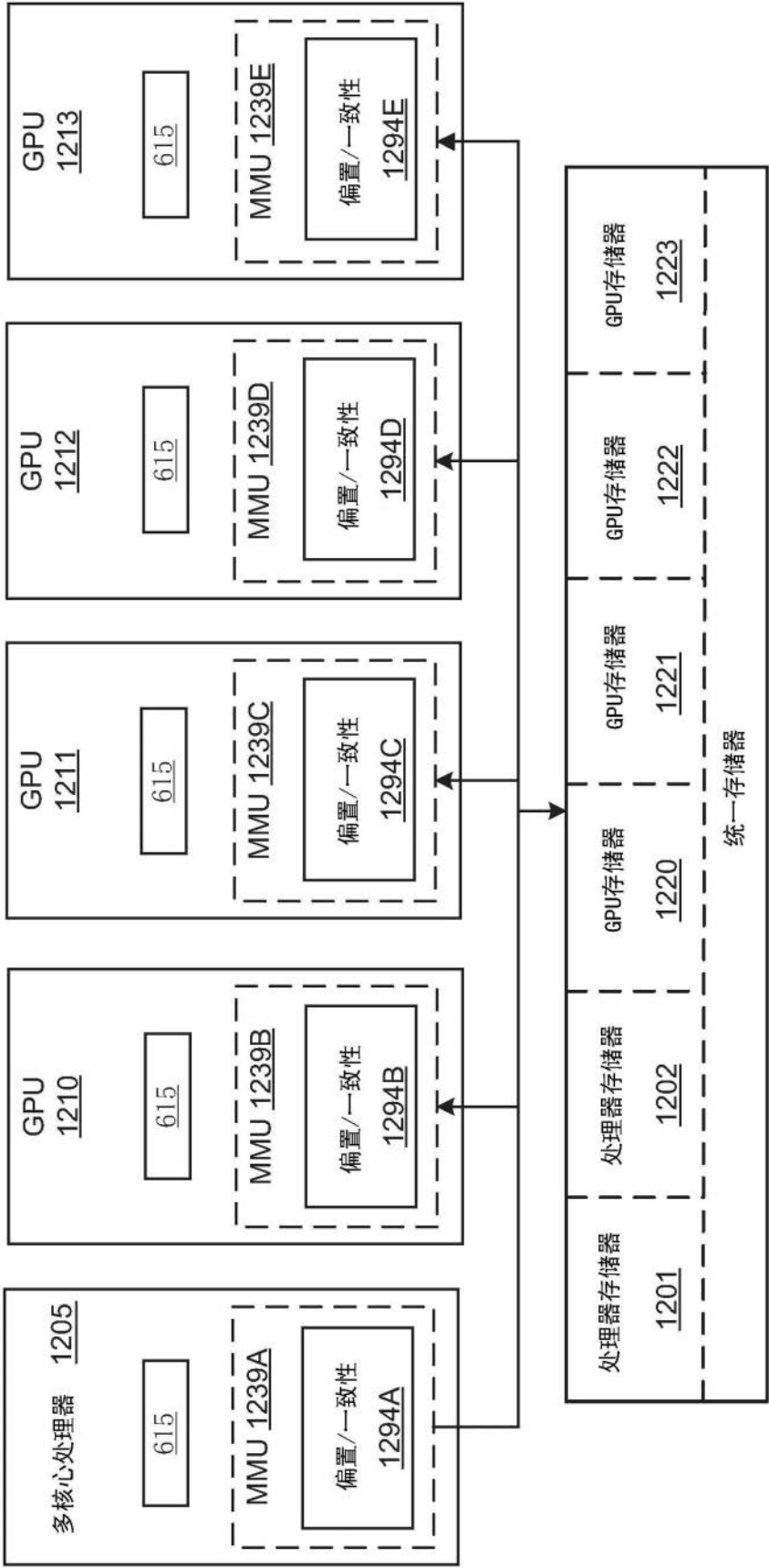


图12F

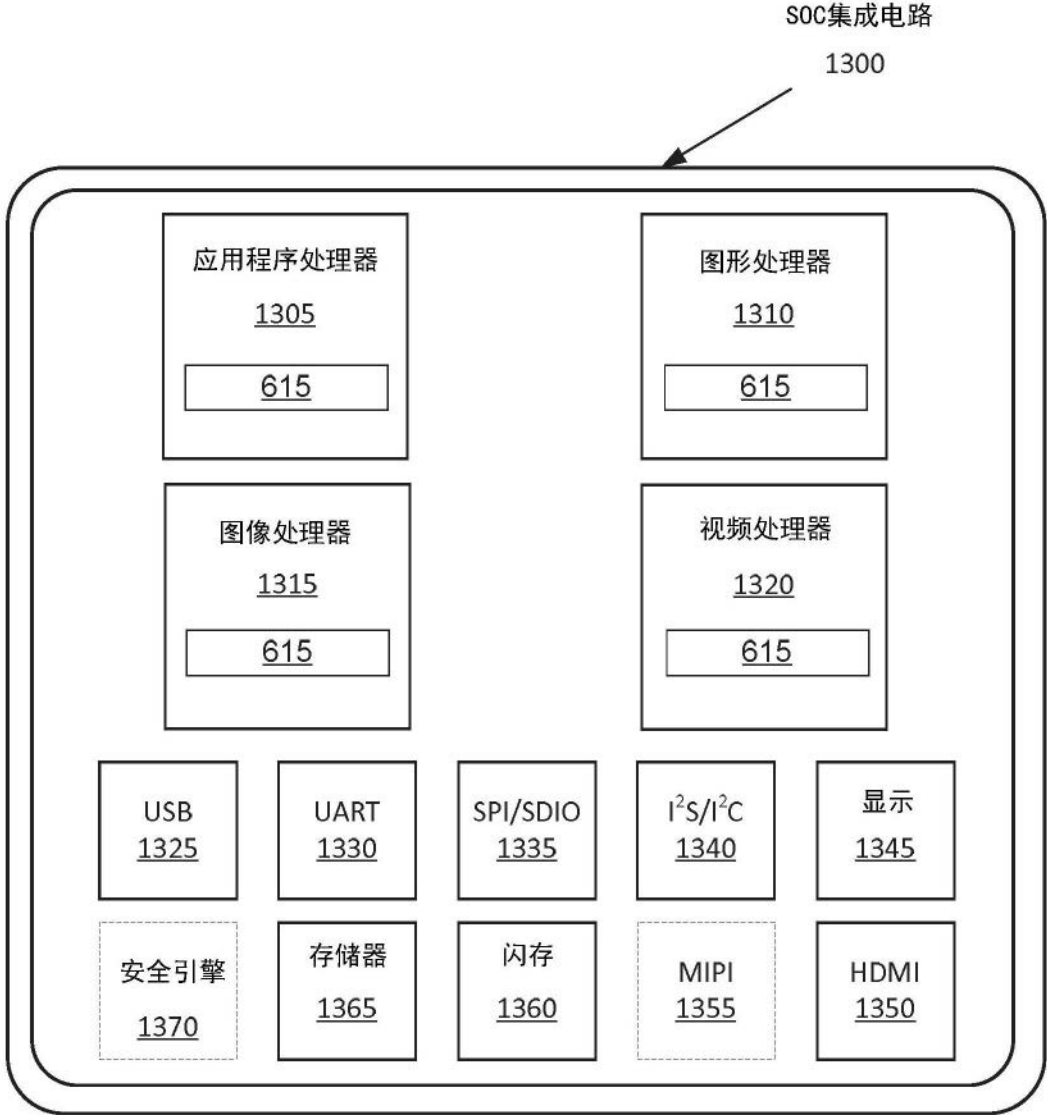


图13

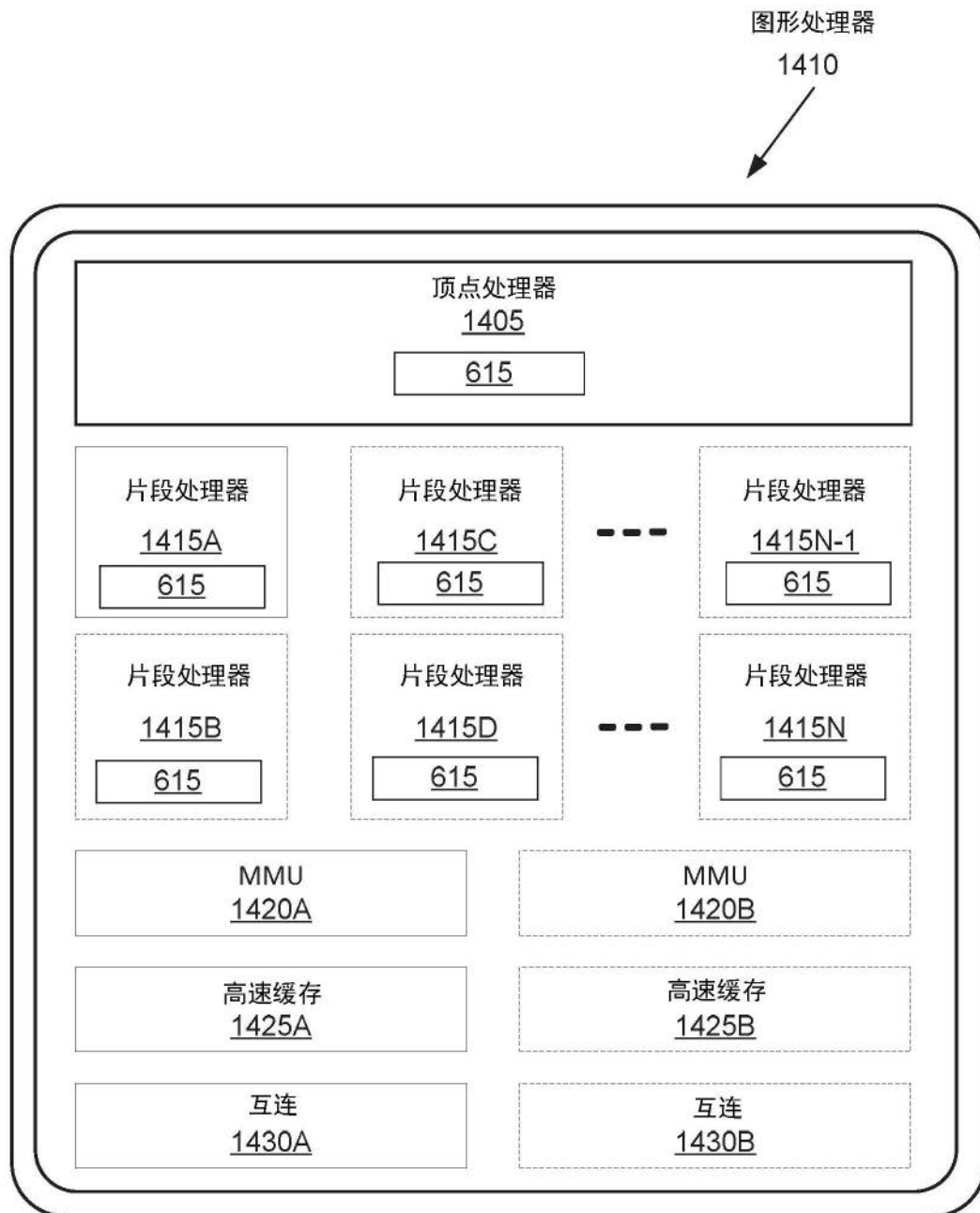


图14A

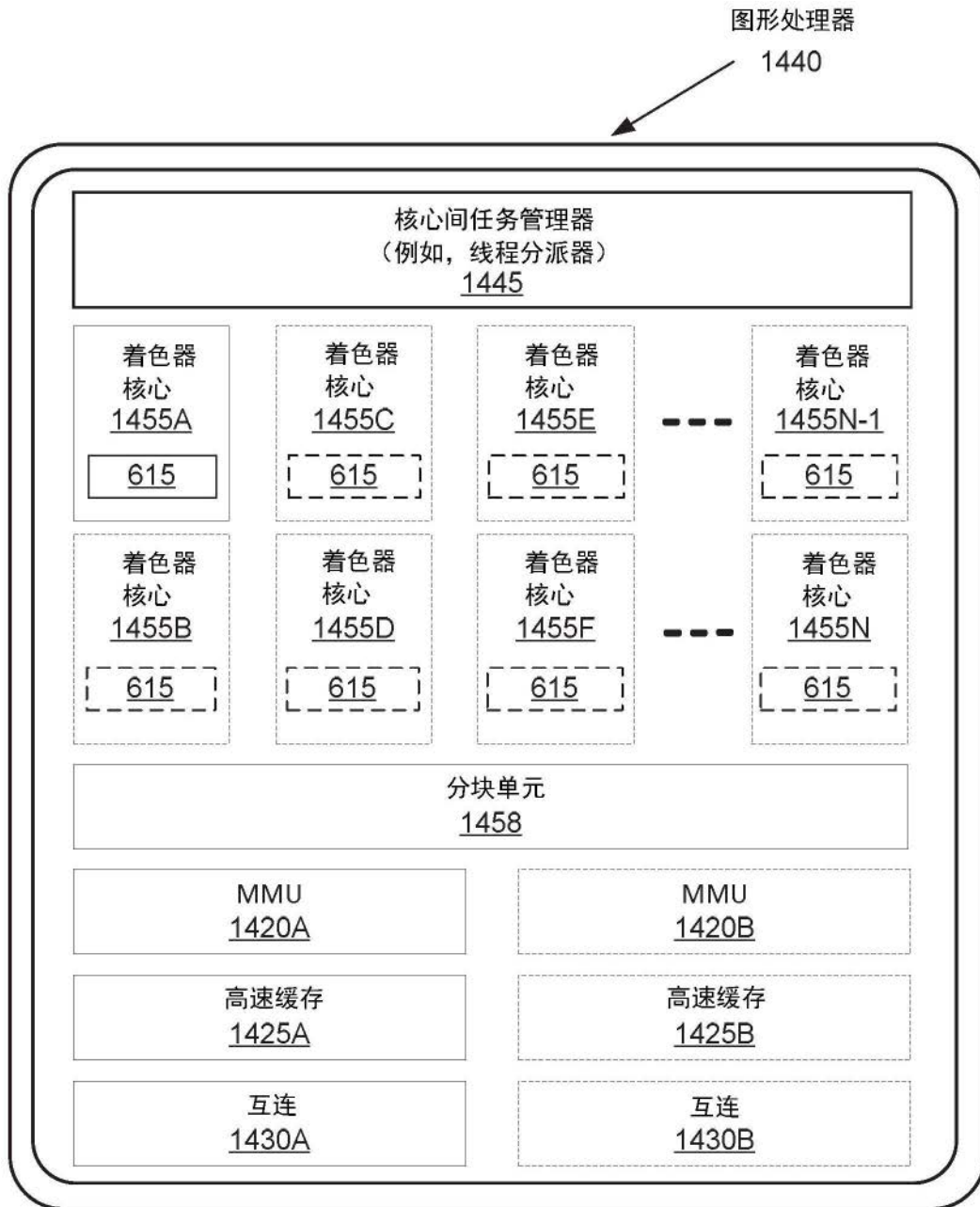


图14B

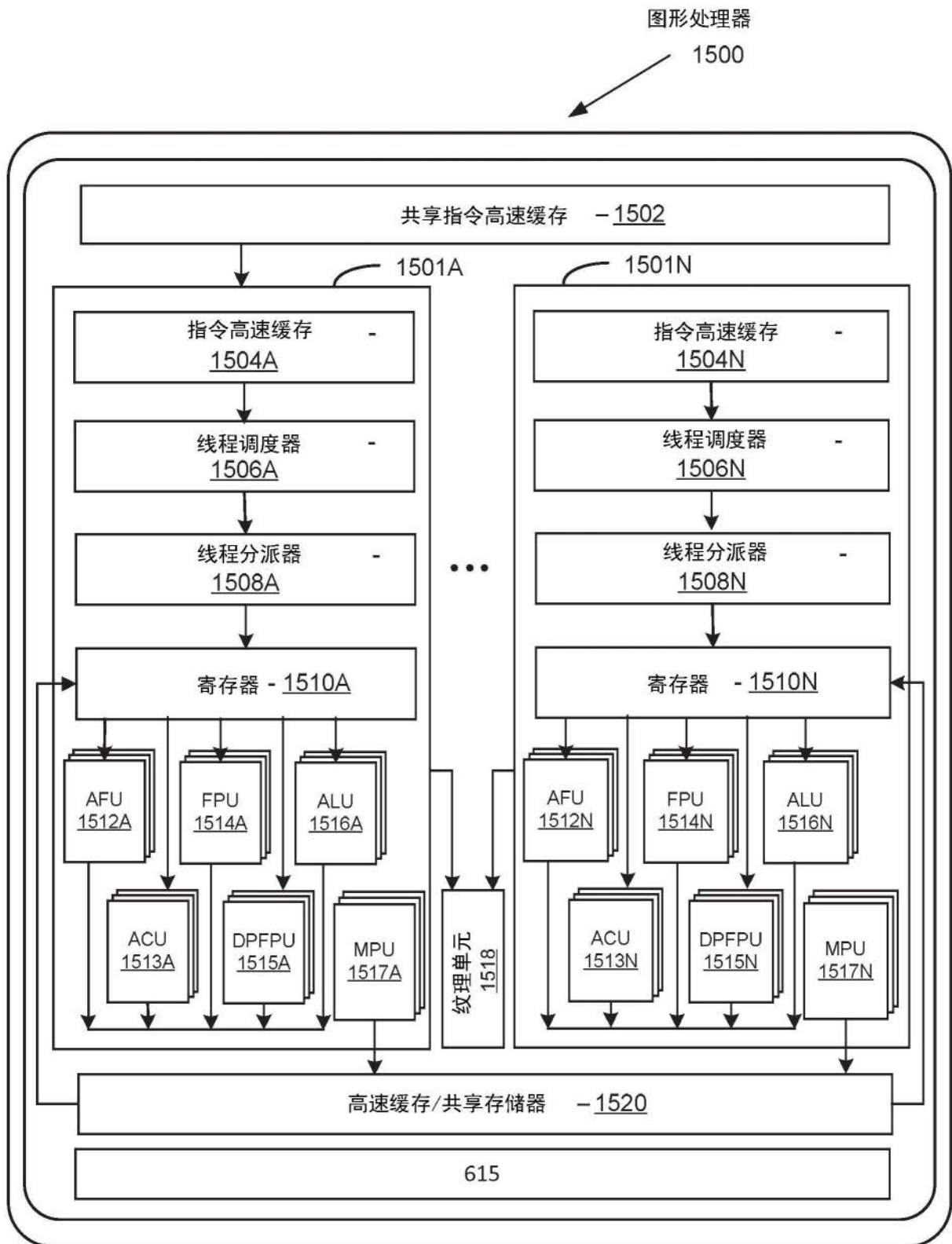


图15A

1530

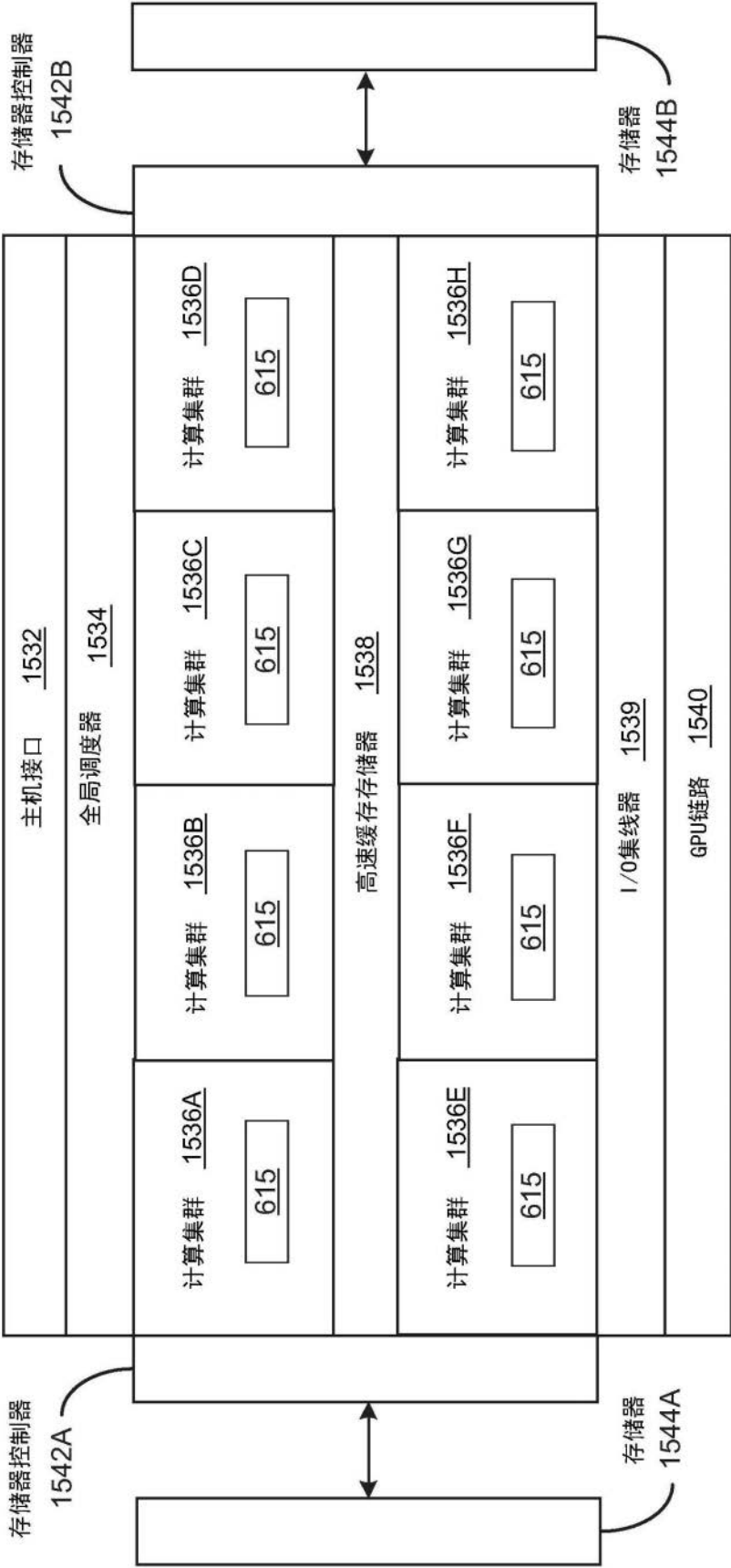


图15B

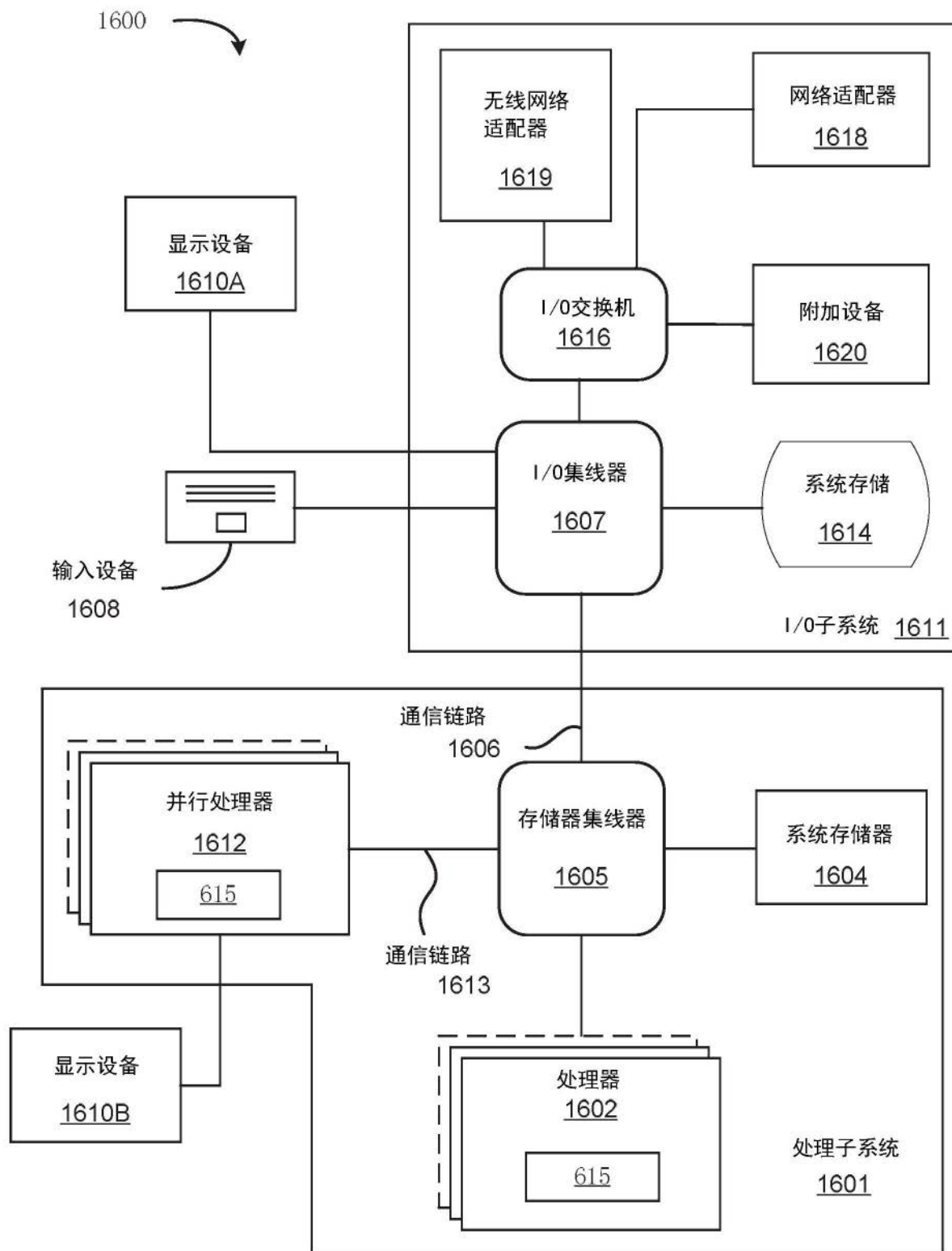


图16

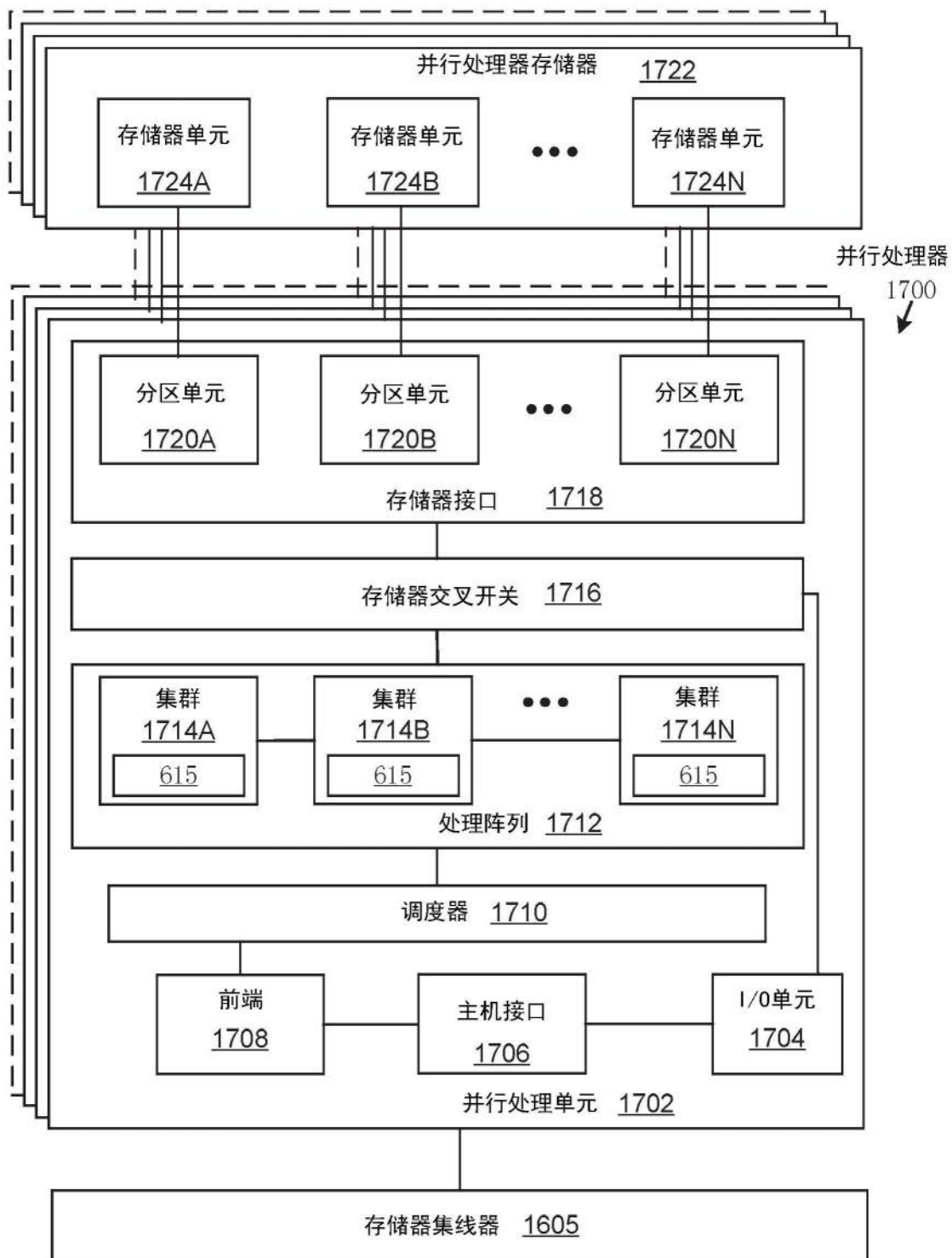


图17A

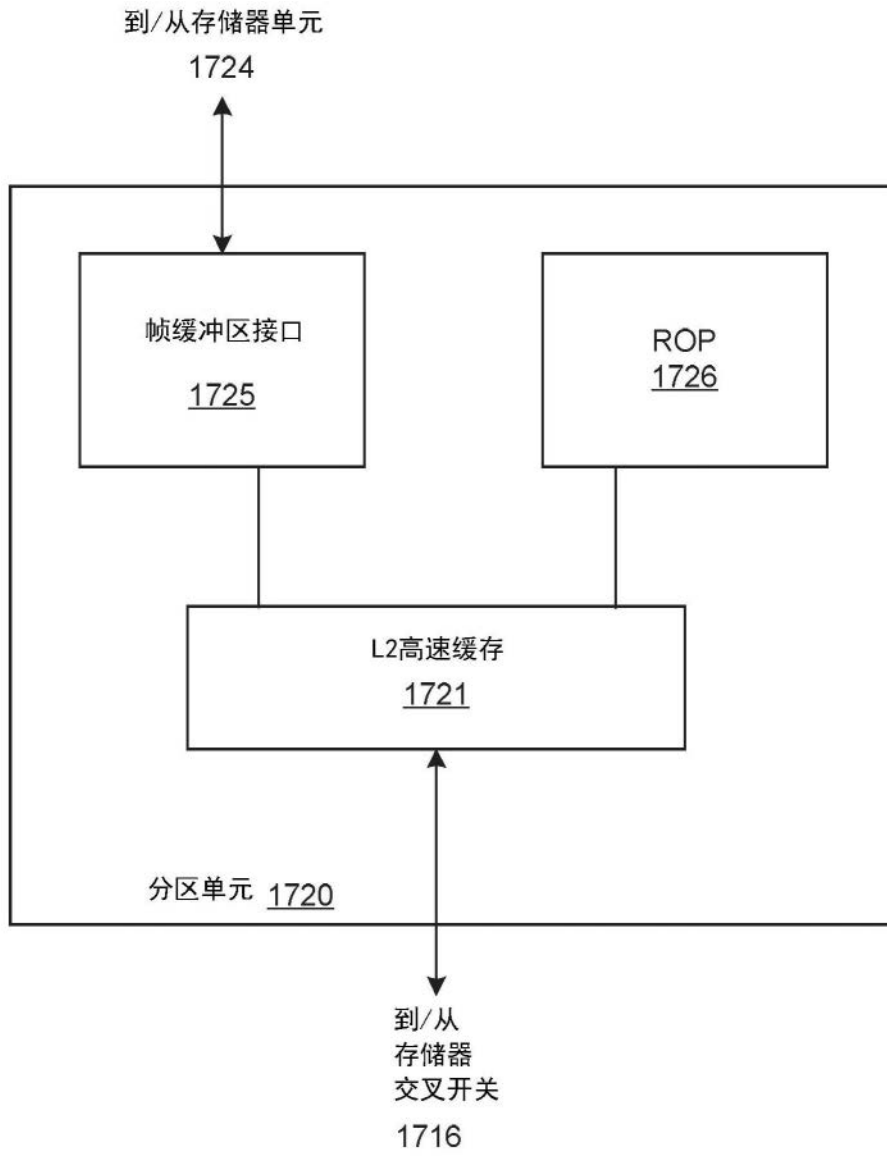


图17B

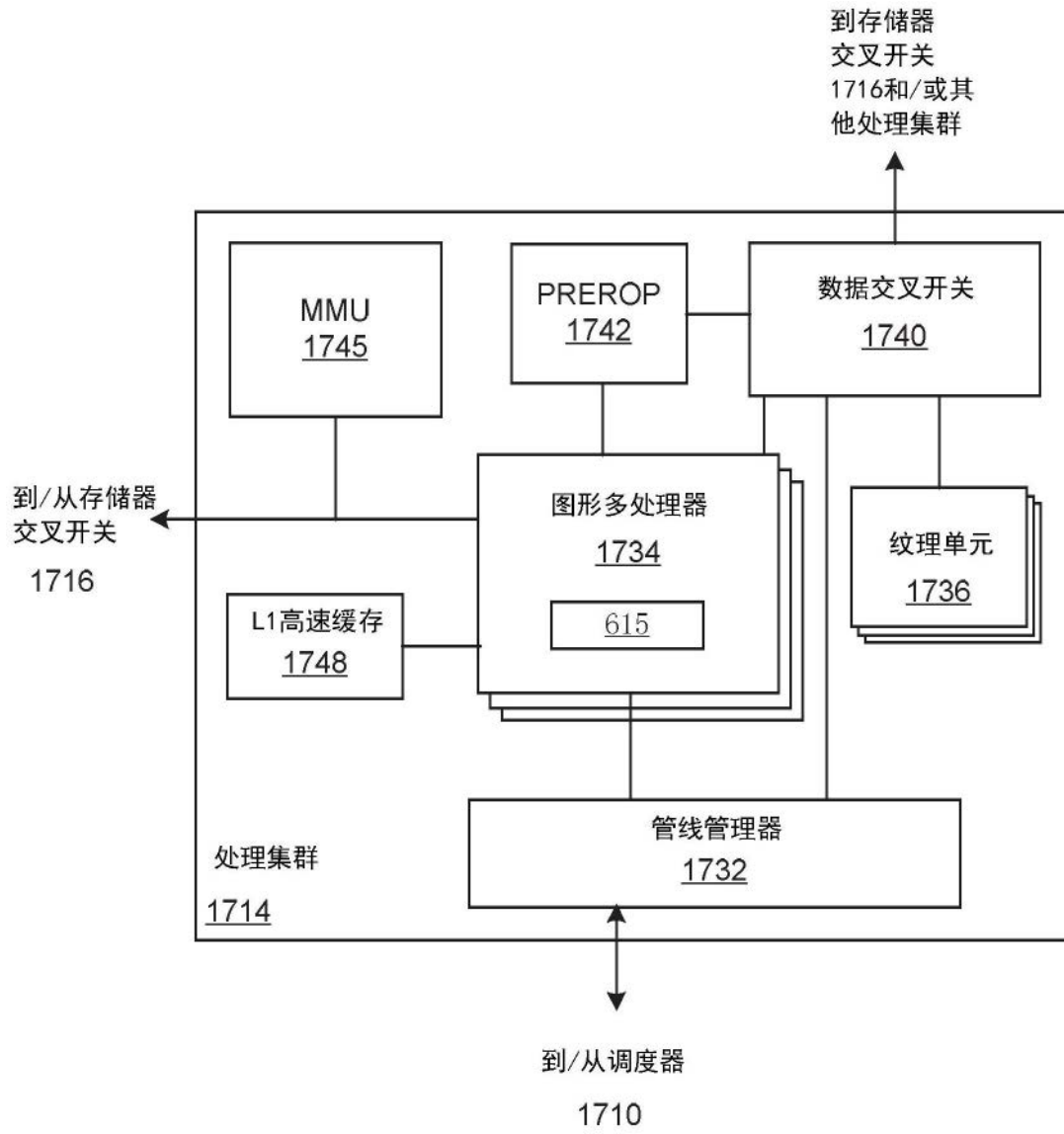


图17C

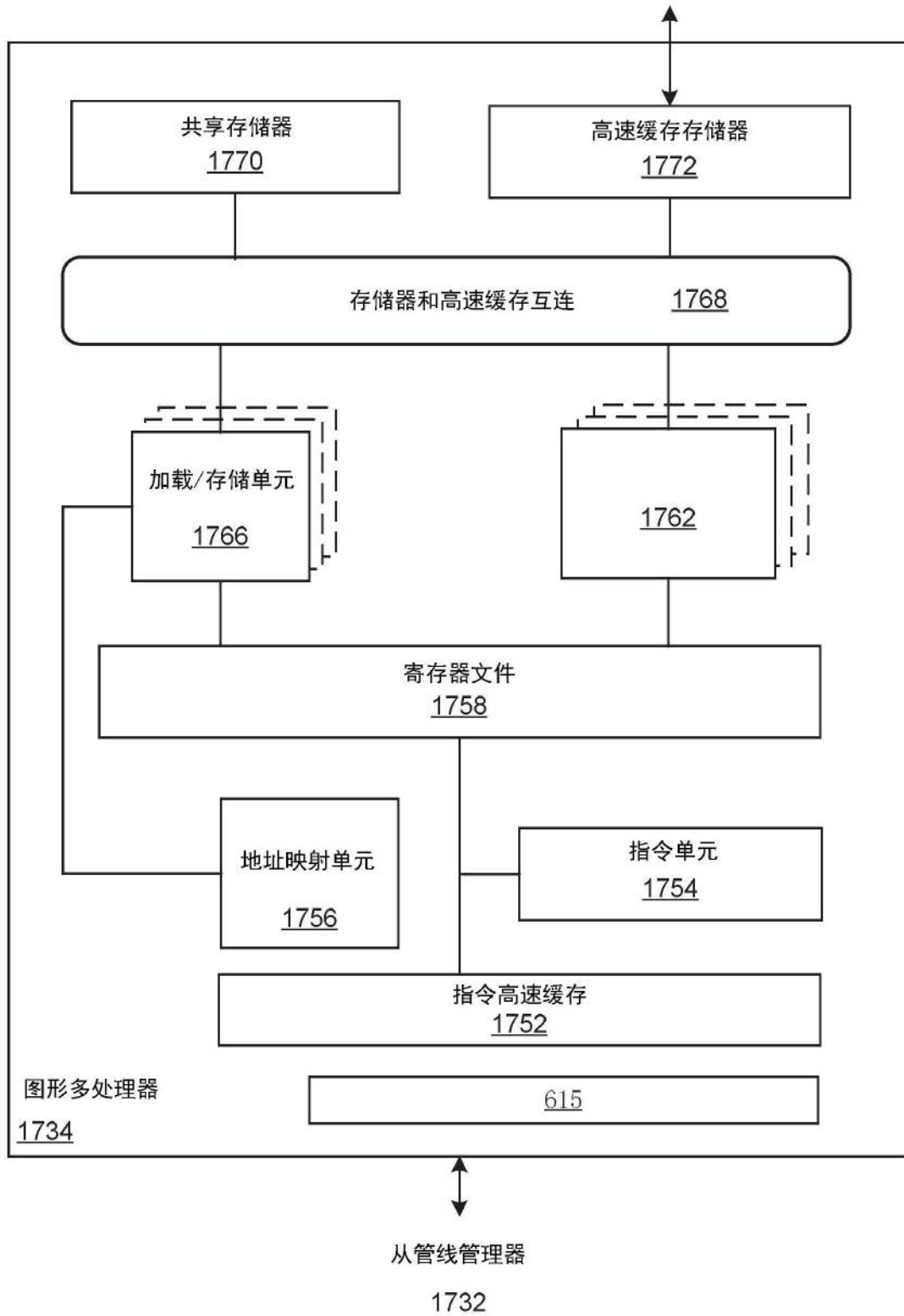


图17D

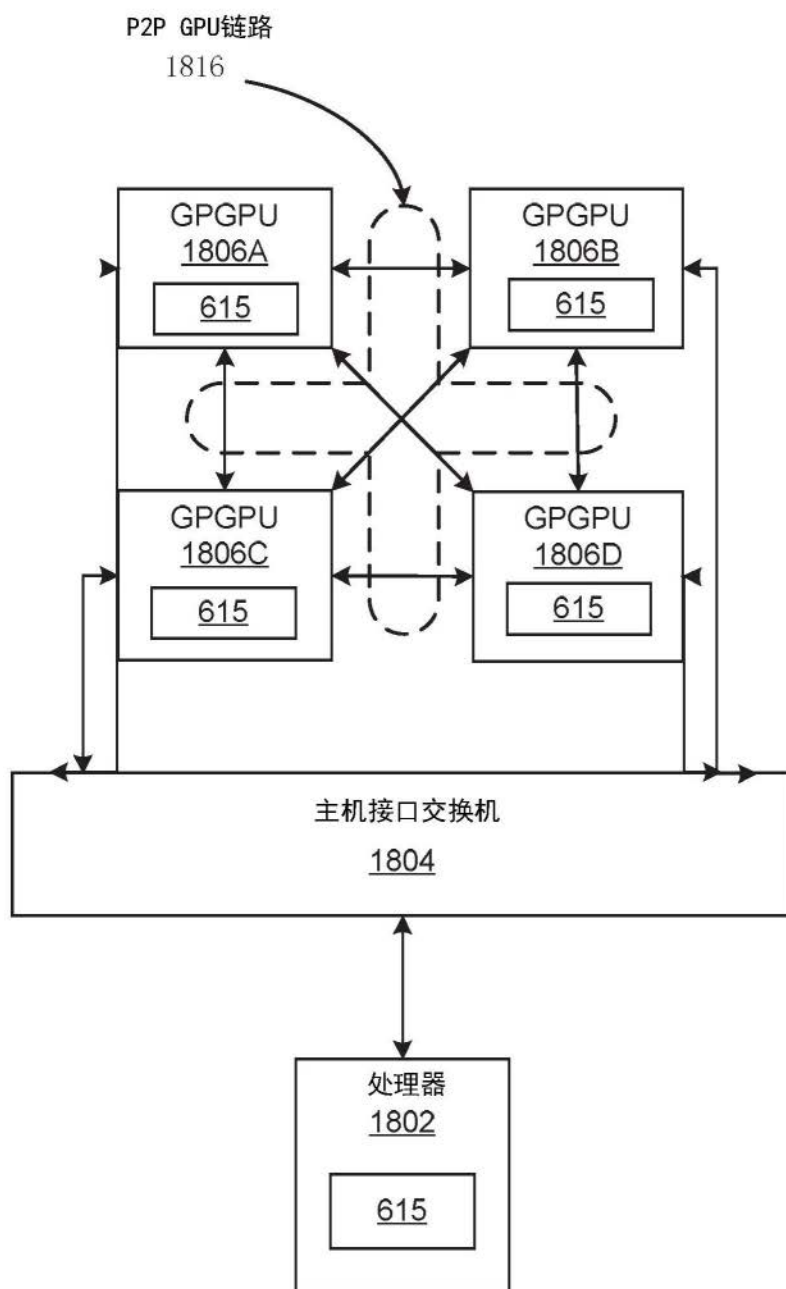
1800

图18

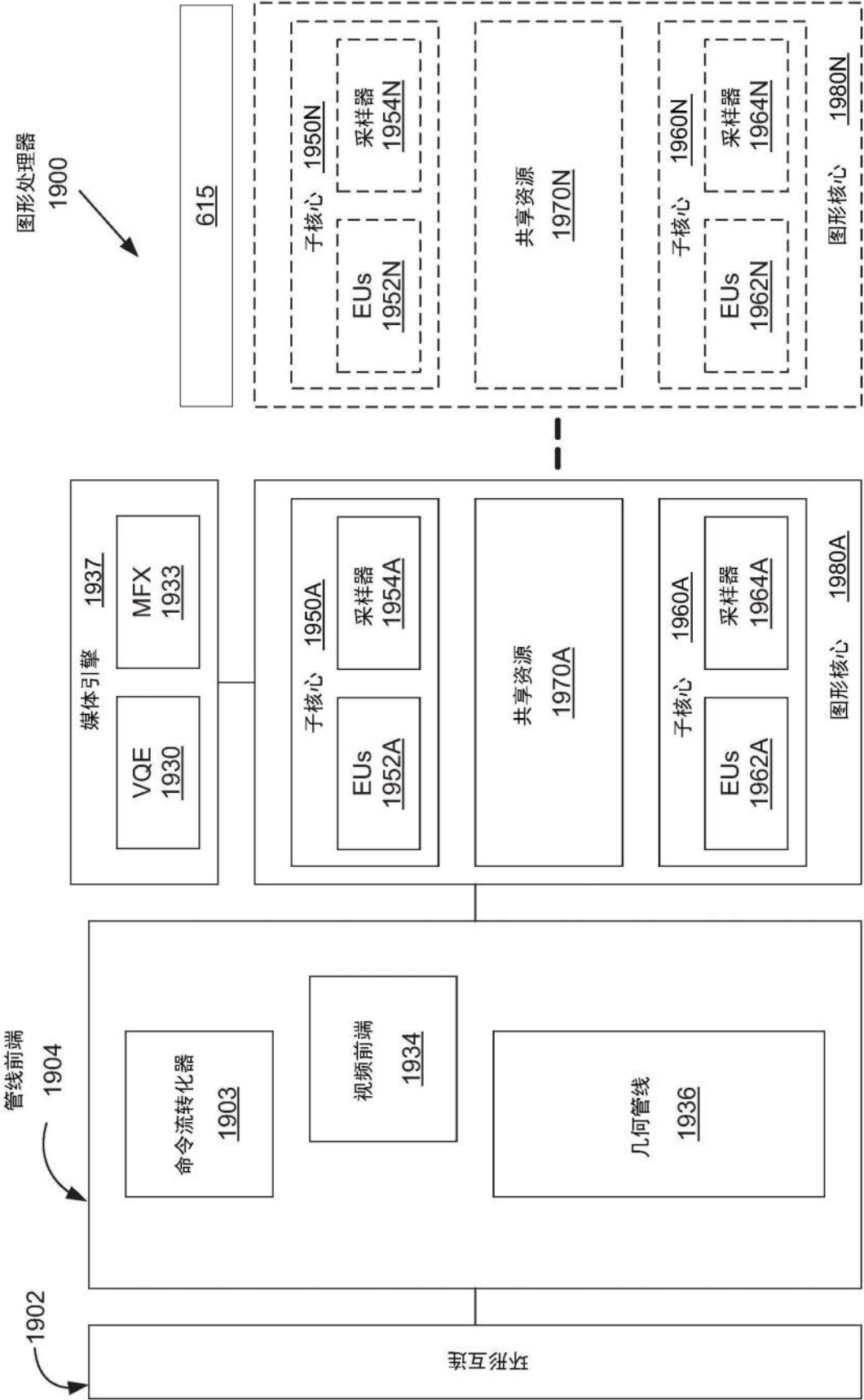


图19

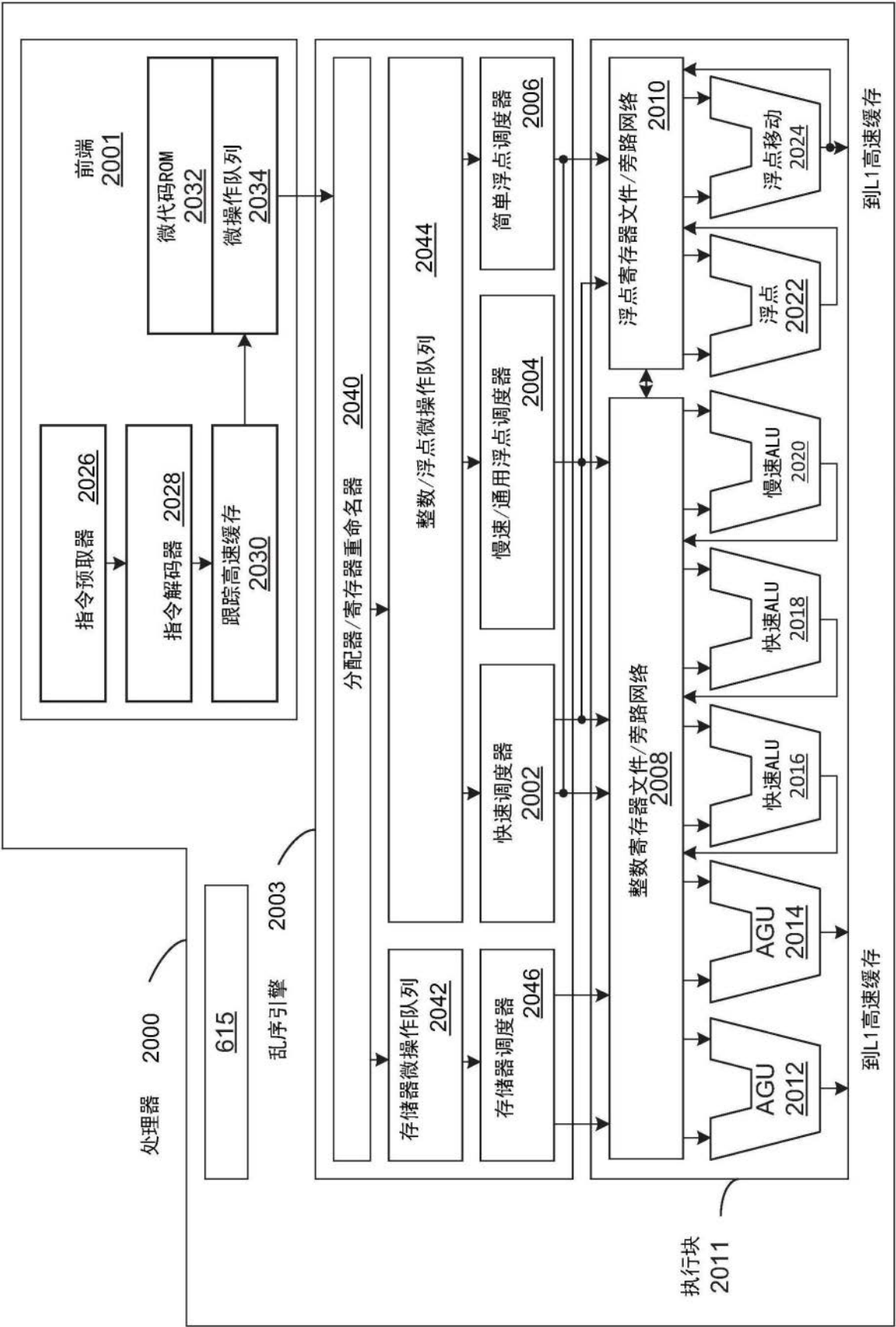


图20

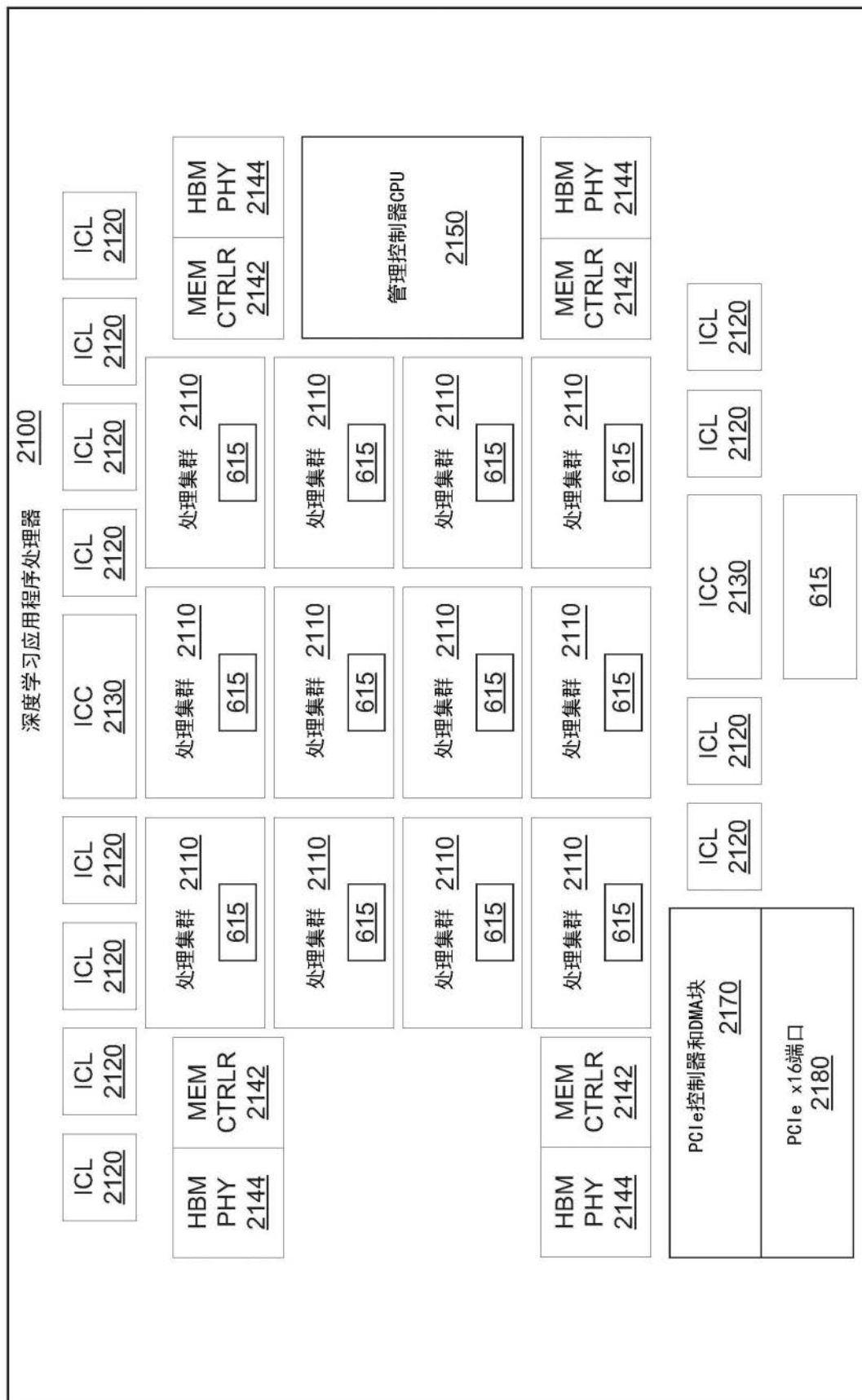


图21

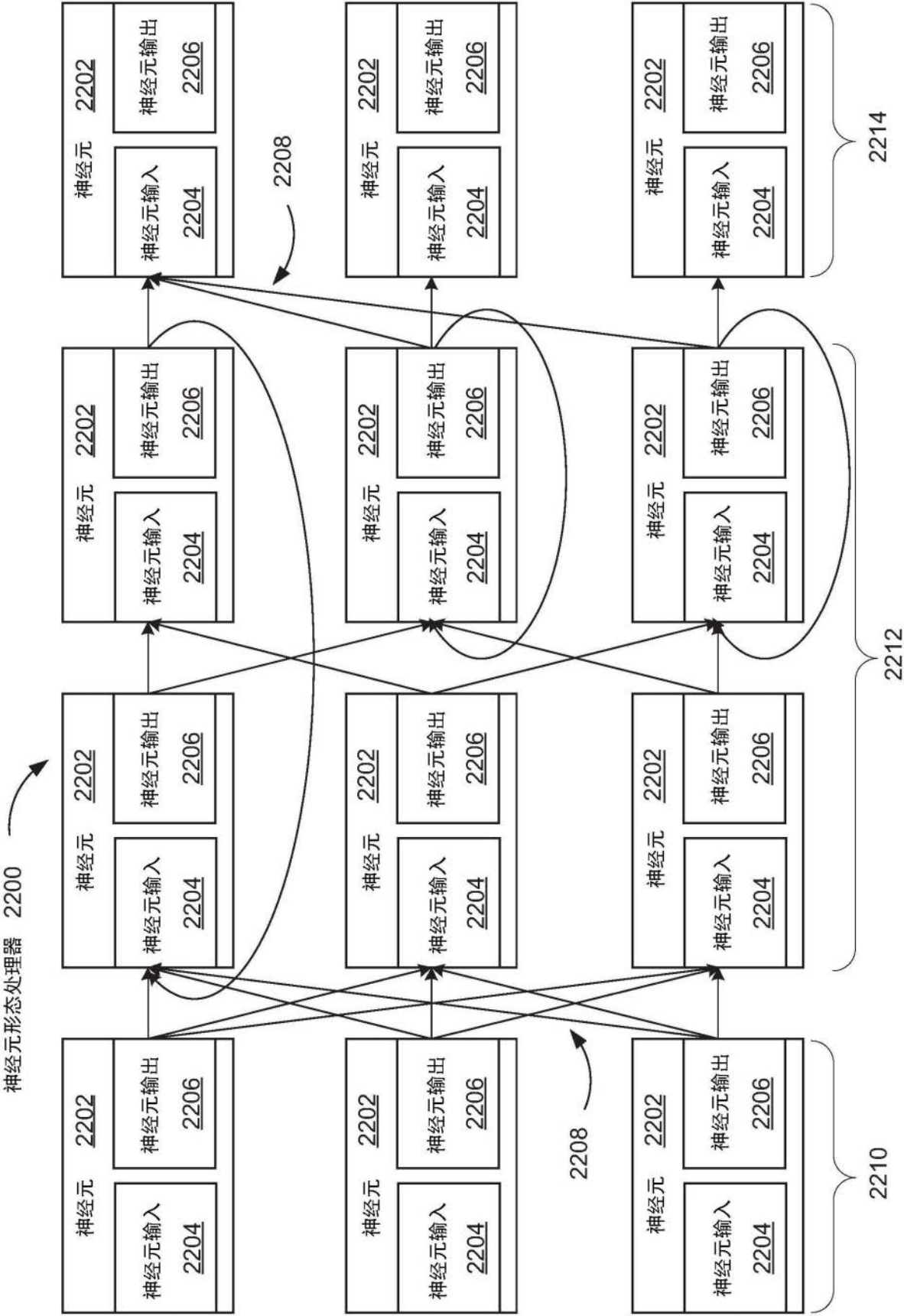


图22

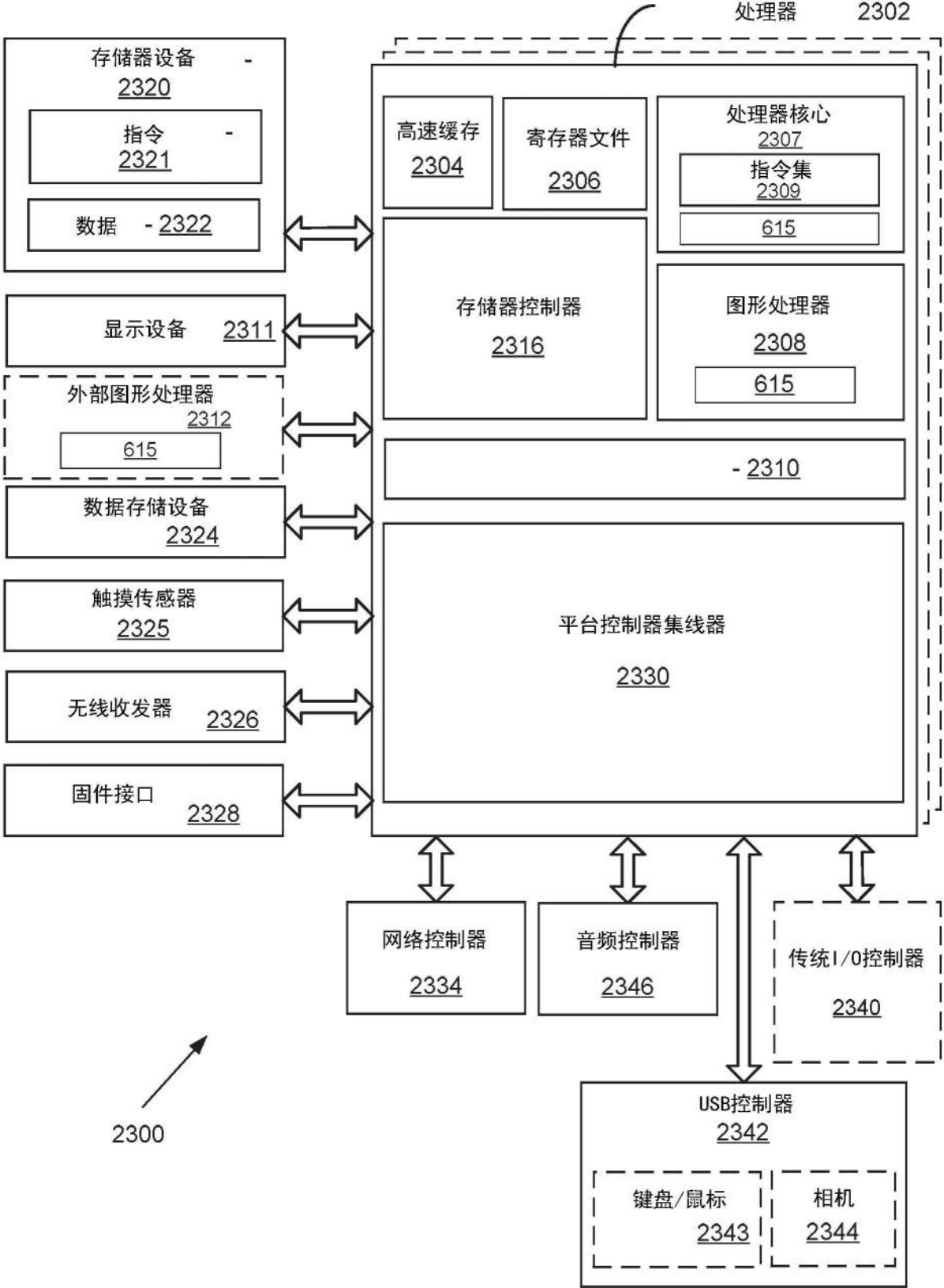


图23

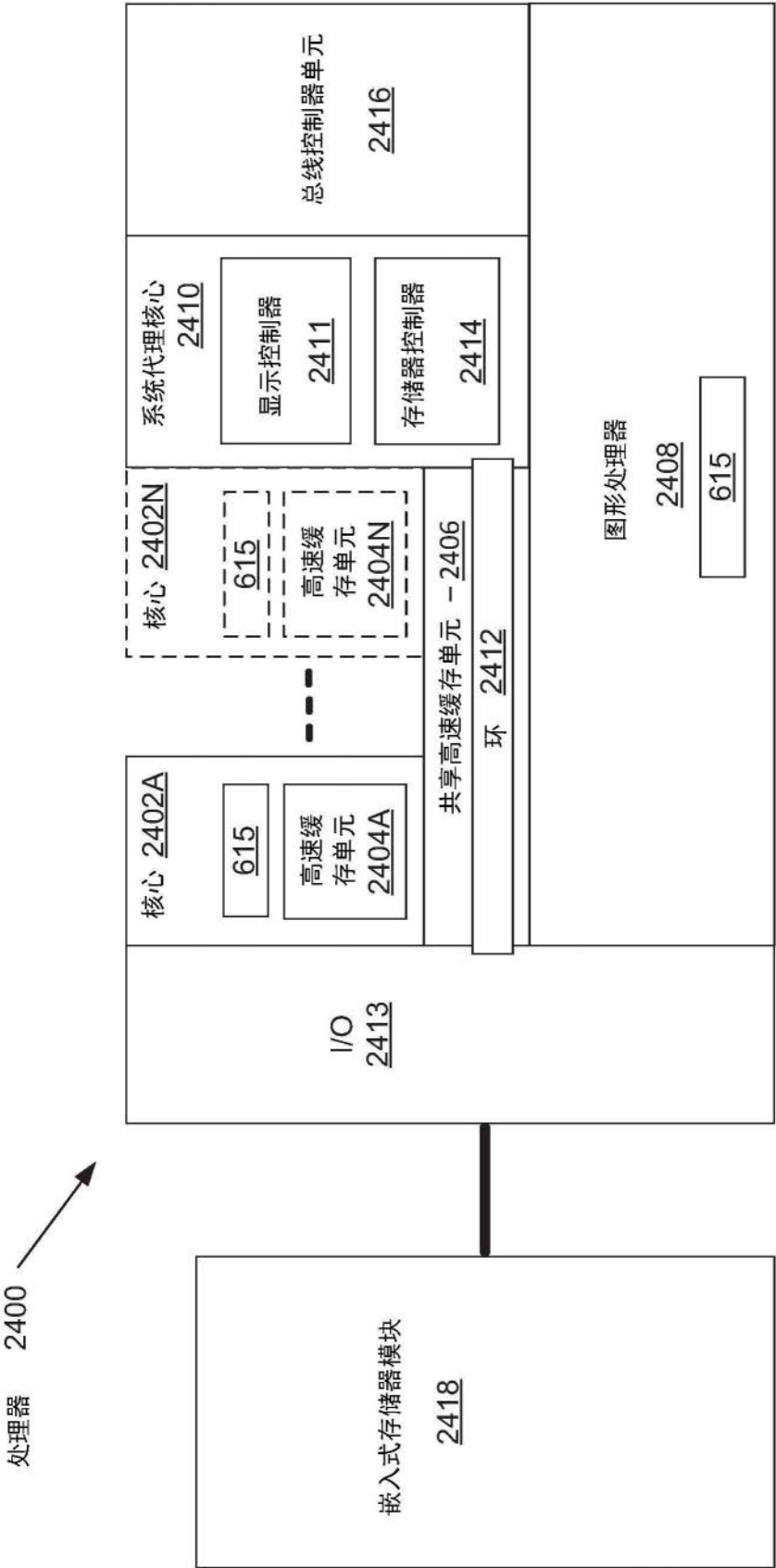


图24

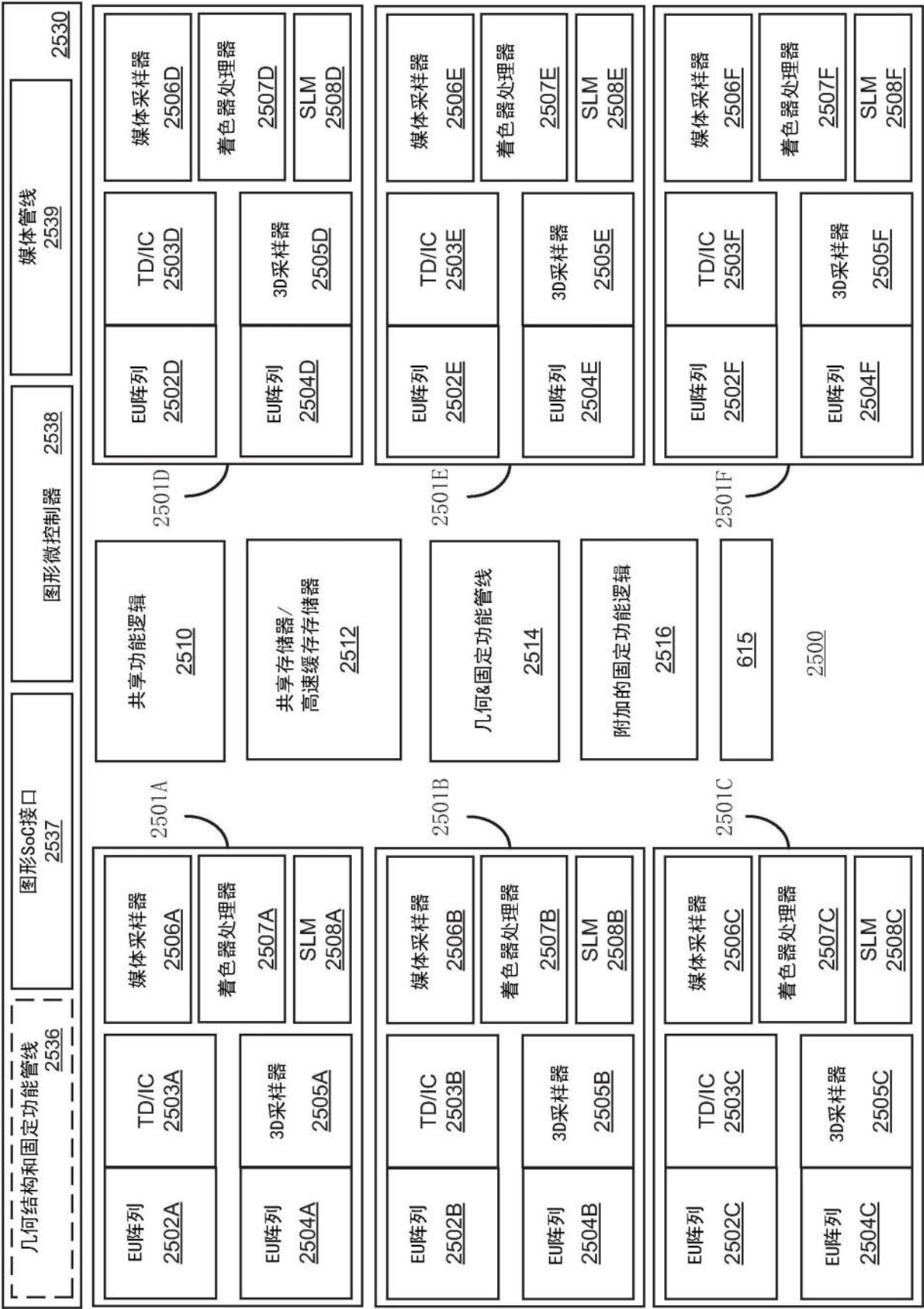


图25

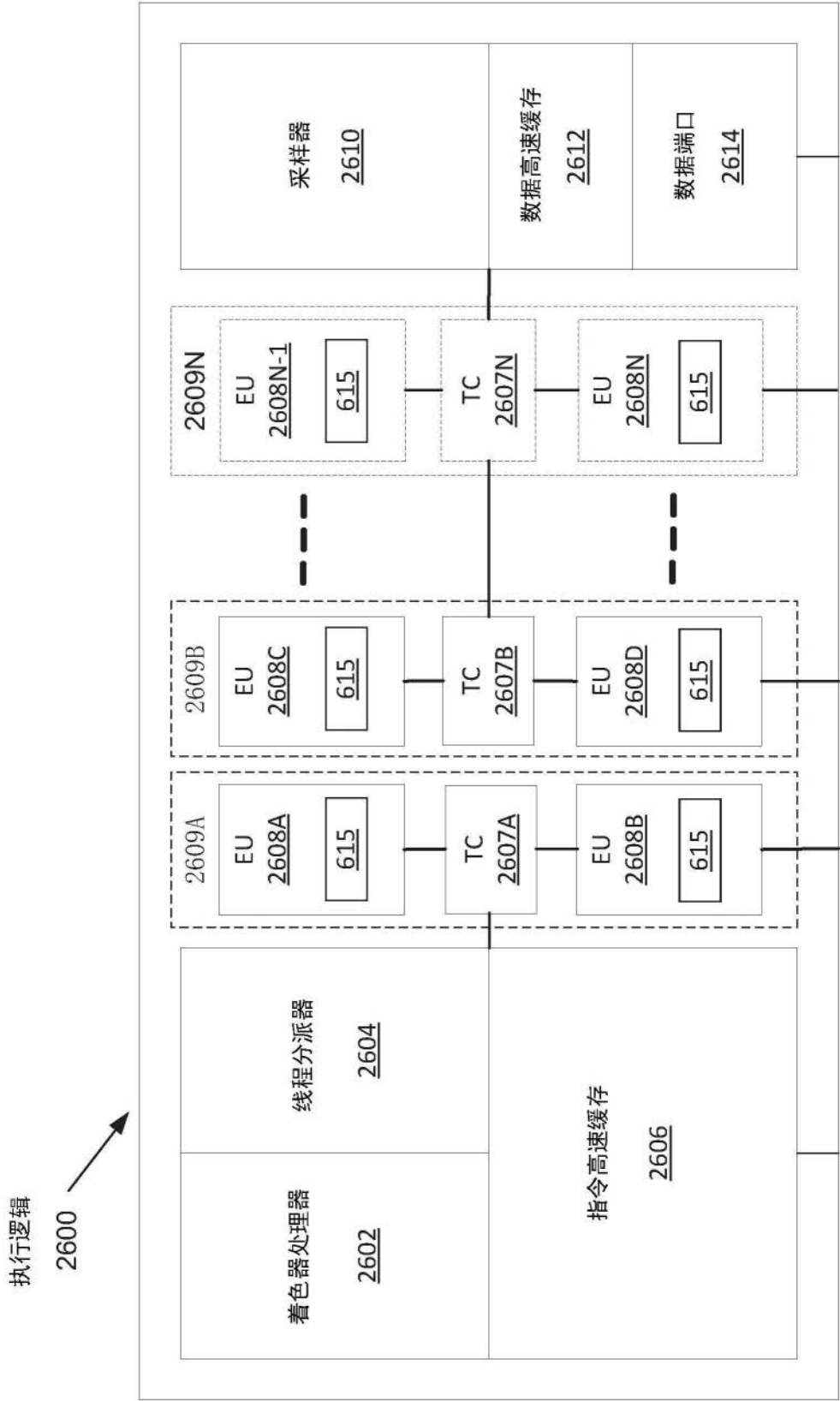


图26A

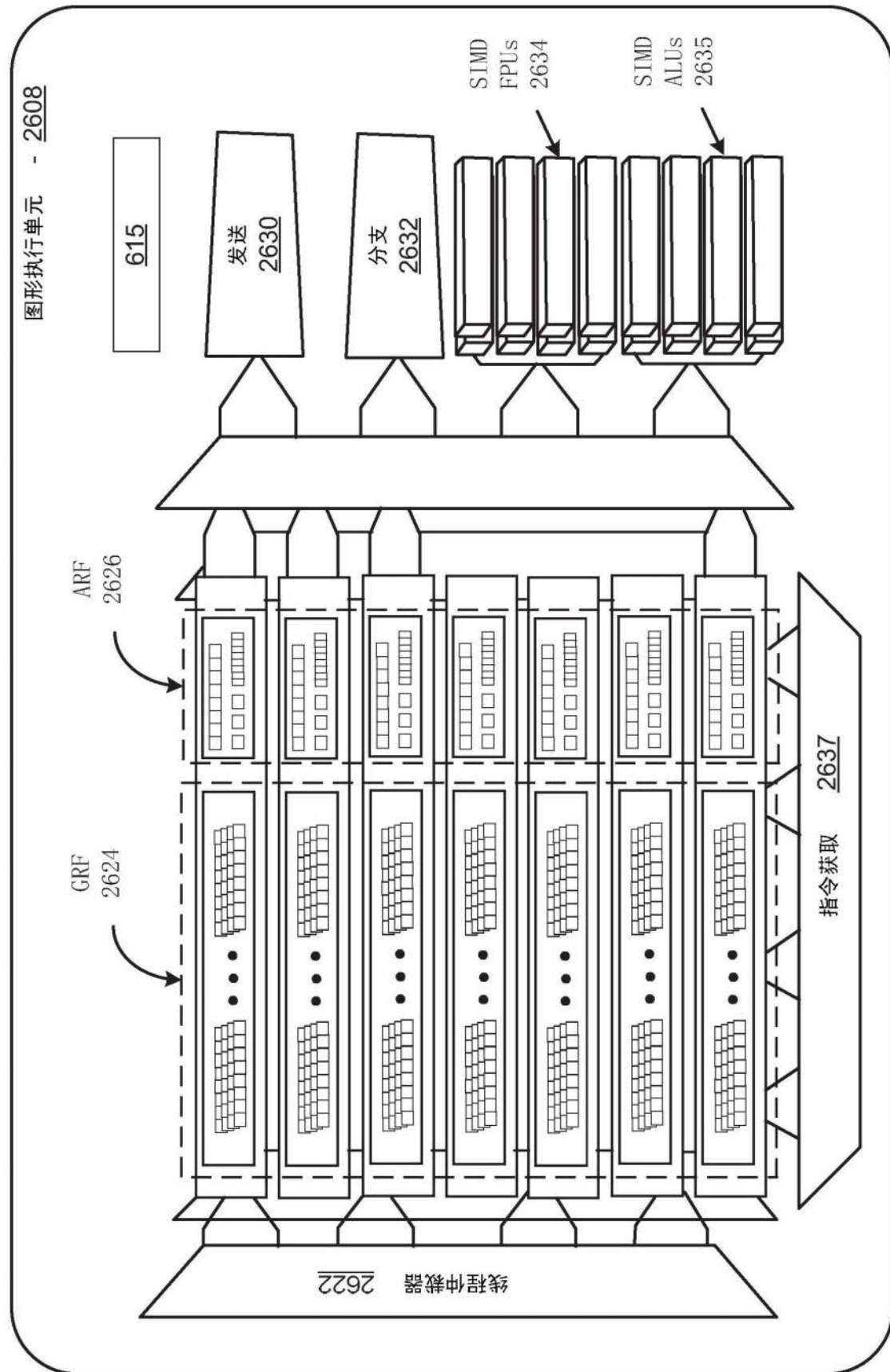


图26B

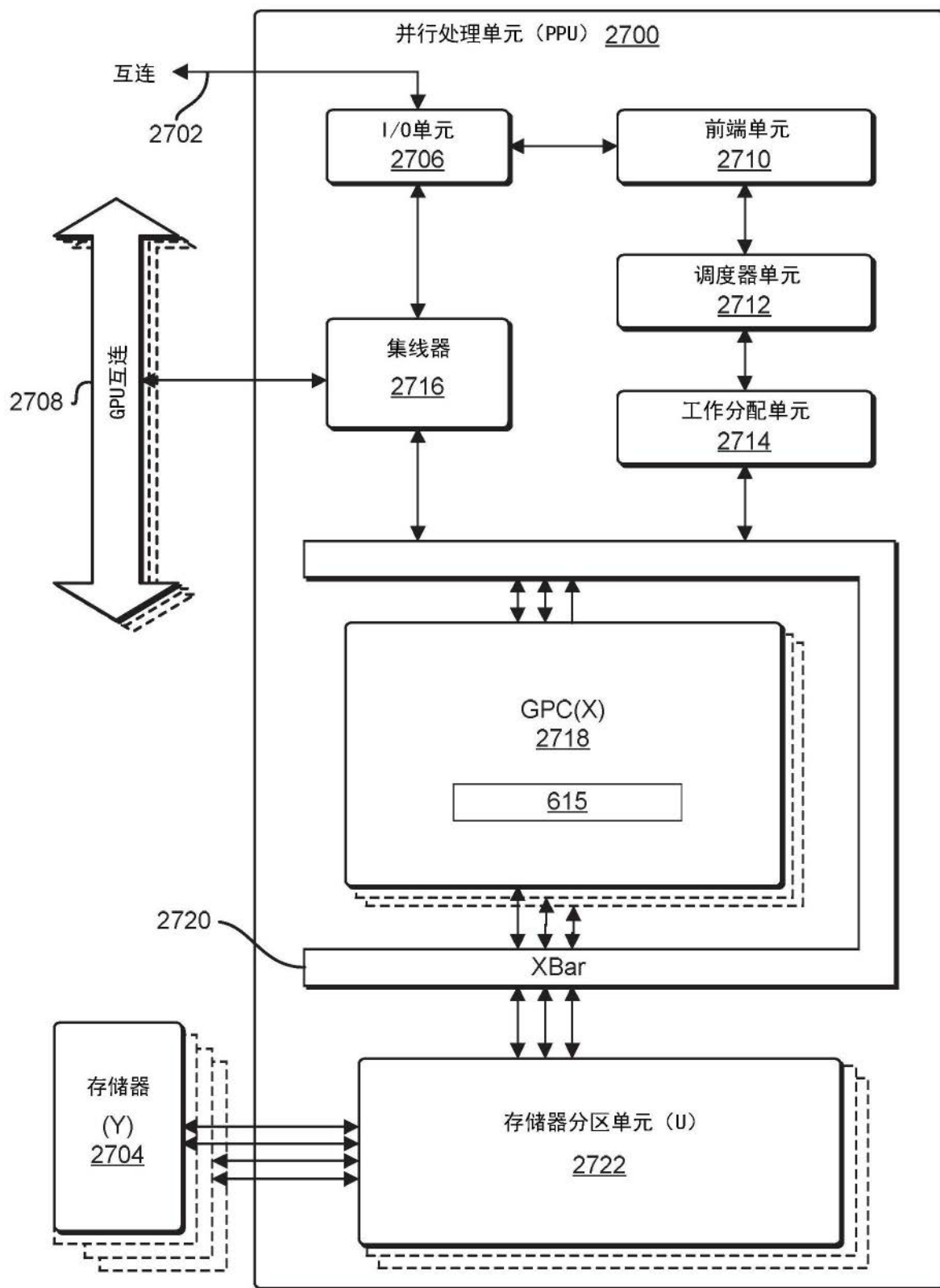


图27

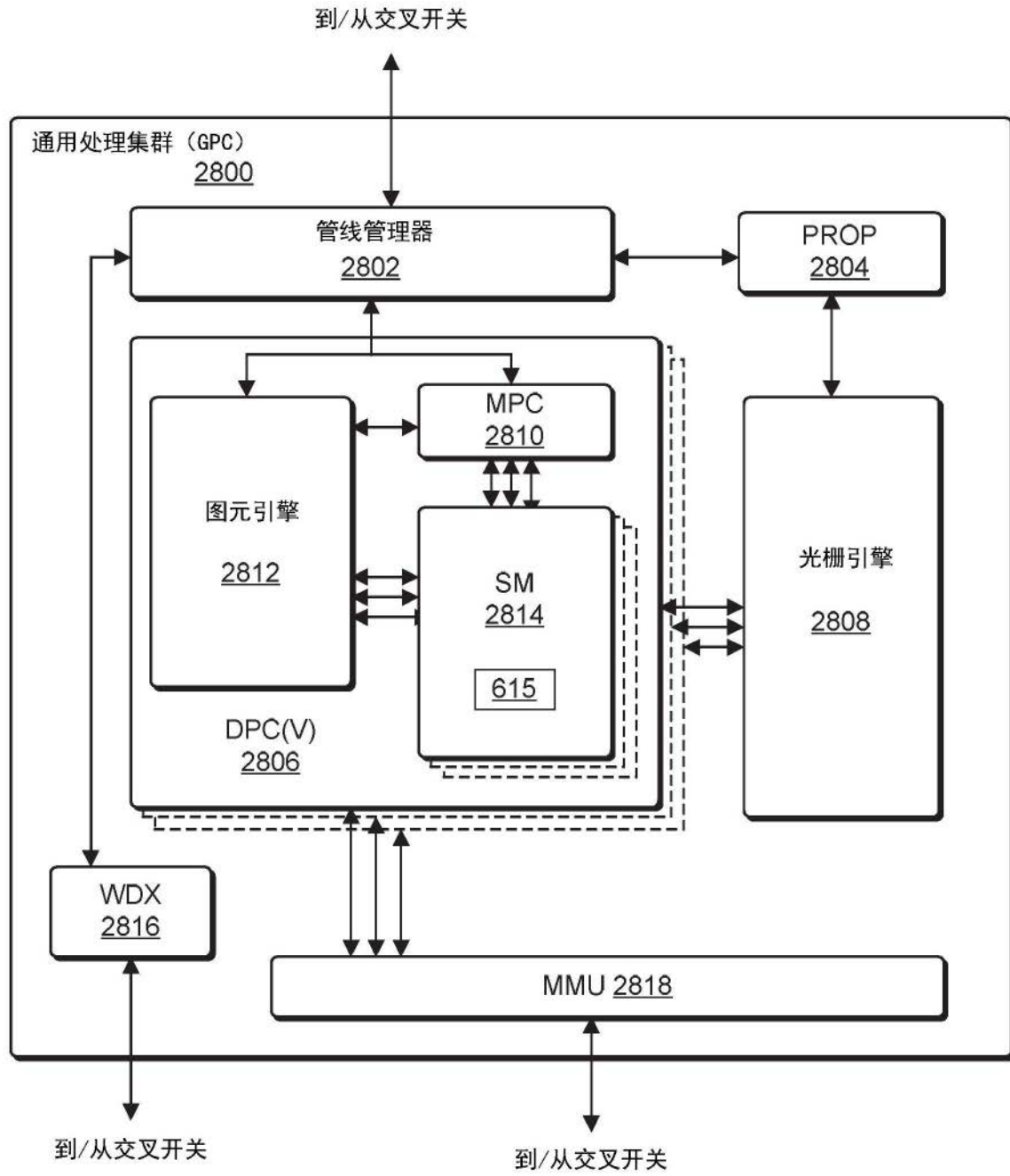


图28

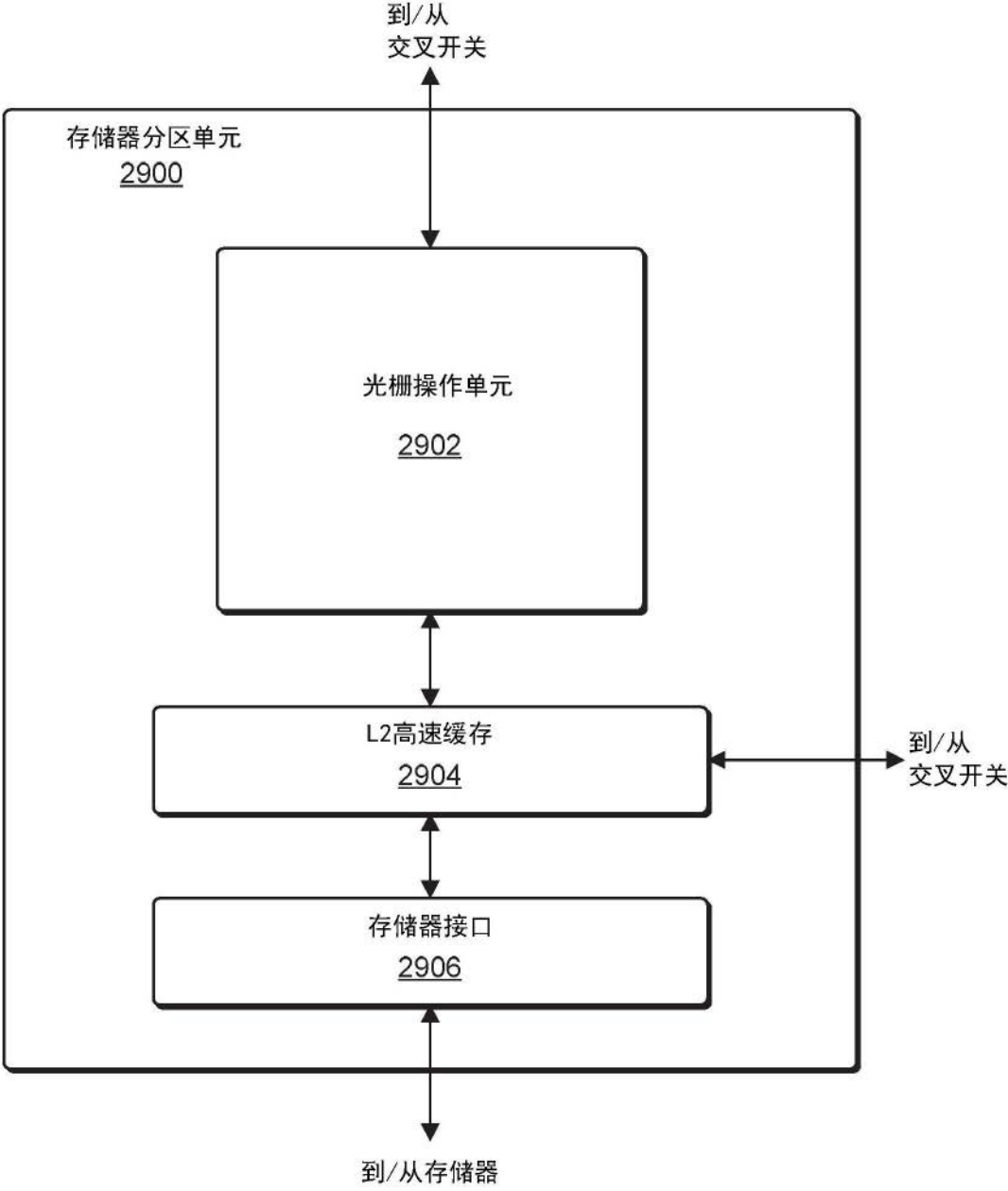


图29

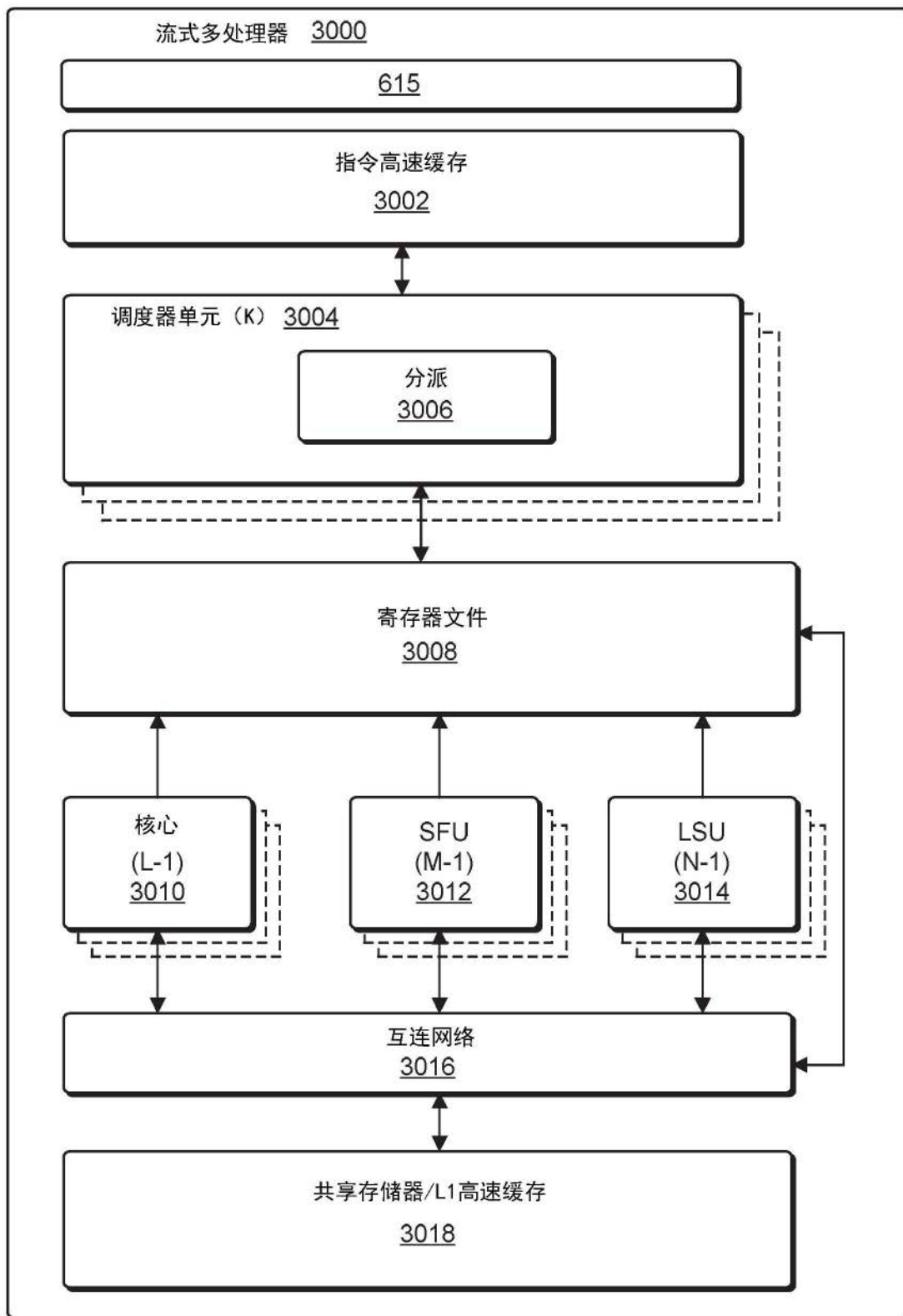


图30