(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 111737019 B
(45)授权公告日 2020.12.11

(21)申请号 202010896588.9

(22)申请日 2020.08.31

(65)同一申请的已公布的文献号
申请公布号 CN 111737019 A

(43)申请公布日 2020.10.02

(73)专利权人 西安芯瞳半导体技术有限公司
地址 710065 陕西省西安市高新区丈八街办丈八一路3号旺都1幢2单元11层21101号

(72)发明人 刘周平 王世凯 李洋

(74)专利代理机构 西安维英格知识产权代理事务所(普通合伙) 61253
代理人 李斌栋 沈寒酉

(51)Int.Cl.
G06F 9/50(2006.01)

(56)对比文件
CN 110928695 A,2020.03.27
CN 111209116 A,2020.05.29
CN 104572509 A,2015.04.29
US 2008136829 A1,2008.06.12
US 10713746 B2,2020.07.14
Jens Kehne 等."GPUswap: Enabling Oversubscription of GPU Memory through Transparent Swapping".《ACM SIGPLAN NOTICES》.2015,
孙立明."支持国产飞腾1500A处理器的计算机图形显示系统优化与实现".《中国优秀硕士学位论文全文数据库 信息科技辑》.2019,(第01期),第I138-2261页.

审查员 牛洪波

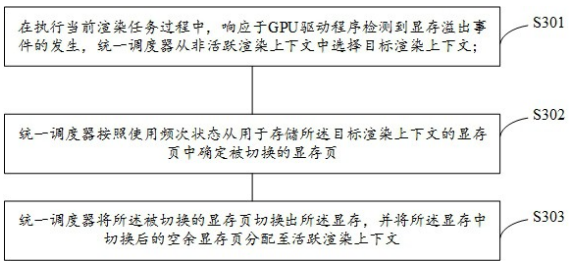
权利要求书2页 说明书9页 附图5页

(54)发明名称

一种显存资源的调度方法、装置及计算机存储介质

(57)摘要

本发明实施例公开了一种显存资源的调度方法、装置及计算机存储介质;该方法应用于CPU的显存管理技术,该方法可以包括:在执行当前渲染任务过程中,响应于图形处理器GPU驱动程序检测到显存溢出事件的发生,统一调度器从非活跃渲染上下文中选择目标渲染上下文;所述统一调度器按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页;所述统一调度器将所述被切换的显存页切换出显存,并将所述显存中切换后的空余显存页分配至活跃渲染上下文。通过上述方案,能够灵活地调度显存资源,并且提高显存资源的利用效率。



CN 111737019 B

1.一种显存资源的调度方法,其特征在于,所述方法包括:

在执行当前渲染任务过程中,响应于图形处理器GPU驱动程序检测到显存溢出事件的发生,统一调度器从非活跃渲染上下文中选择目标渲染上下文;其中,所述非活跃渲染上下文包括当前未执行的渲染上下文;

所述统一调度器按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页;

所述统一调度器将所述被切换的显存页切换出显存,并将所述显存中切换后的空余显存页分配至活跃渲染上下文;其中,所述活跃渲染上下文包括当前被执行的渲染上下文。

2.根据权利要求1所述的方法,其特征在于,所述GPU驱动程序检测到显存溢出事件的发生,包括:

所述GPU驱动程序在为所述活跃渲染上下文分配显存时检测到显存剩余资源不足以为所述活跃渲染上下文进行分配;

或者,在根据所述活跃渲染上下文执行当前渲染任务过程中,所述GPU驱动程序根据由所述GPU中的存储管理单元MMU上报的中断信号触发中断服务程序;其中,所述中断信号包括所述GPU中的MMU在执行当前渲染任务过程中发现产生的中间数据导致所述显存资源不足时所反馈的中断信号。

3.根据权利要求2所述的方法,其特征在于,所述在根据所述活跃渲染上下文执行当前渲染任务过程中,所述GPU驱动程序根据由所述GPU中的存储管理单元MMU上报的中断信号触发中断服务程序,包括:

所述GPU驱动程序接收由所述GPU的MMU反馈的中断信号;其中,所述中断信号为所述GPU的MMU发现缺页中断状态时所反馈的信号,且包括缺页中断的描述信息以及全局显存资源的使用信息;

所述GPU驱动程序触发中断服务程序,并向所述统一调度器传输所述缺页中断的描述信息以及全局显存资源的使用信息,以使得所述统一调度器基于所述中断服务程序的触发根据所述缺页中断的描述信息以及全局显存资源的使用信息选择目标渲染上下文。

4.根据权利要求1所述的方法,其特征在于,所述统一调度器从非活跃渲染上下文中选择目标渲染上下文,包括:

所述统一调度器从所述非活跃渲染上下文中将所述当前被执行的渲染上下文之前最接近的已执行完毕的渲染上下文选择为所述目标渲染上下文。

5.根据权利要求1所述的方法,其特征在于,所述统一调度器按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页,包括:

所述统一调度器从所述目标渲染上下文的显存页中,基于最近最少使用LRU算法选择最近最少使用的显存页作为所述被切换的显存页。

6.根据权利要求1至5任一项所述的方法,其特征在于,所述方法还包括:

所述统一调度器在系统内存内为所述目标渲染上下文分配存储空间;

所述统一调度器将所述被切换的显存页同步至所述GPU的MMU,以使得所述GPU调用所述GPU的MMU完成地址转换;

所述统一调度器启动所述GPU的直接存储器访问,根据转换完成的地址将所述被切换的显存页切换至所述系统内存中。

7.根据权利要求1至5任一项所述的方法，其特征在于，所述统一调度器针对每个渲染上下文均对应保存一张显存资源使用表，用于记录被对应的渲染上下文所存储的显存地址以及虚拟显存的使用情况，同时保存有所述被对应的渲染上下文中每个显存页的使用频次；相应地，所述方法还包括：

所述统一调度器将所述目标渲染上下文对应的显存资源使用表进行更新，以此保证当切换到下一次渲染上下文过程中，显存访问的正确性。

8.一种显存资源的调度装置，其特征在于，所述装置包括：选择部分，确定部分以及第一分配部分，其中，

所述选择部分，经配置为在执行当前渲染任务过程中，响应于图形处理器GPU驱动程序检测到显存溢出事件的发生，从非活跃渲染上下文中选择目标渲染上下文；其中，所述非活跃渲染上下文包括当前未执行的渲染上下文；

所述确定部分，经配置为按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页；

所述第一分配部分，经配置为将所述被切换的显存页切换出显存，并将所述显存中切换后的空余显存页分配至活跃渲染上下文；其中，所述活跃渲染上下文包括当前被执行的渲染上下文。

9.一种显存资源的调度装置，其特征在于，所述装置包括：CPU、图形处理器GPU、显存以及存储器；其中，所述存储器存储有GPU驱动程序以及用于执行权利要求1至7任一项所述的显存资源的调度方法步骤的显存资源的调度程序；所述CPU，经配置为在执行当前渲染任务过程中，基于所述存储器所存储的GPU驱动程序以及所述显存资源的调度程序对所述GPU的显存的资源进行调度。

10.一种计算机存储介质，其特征在于，所述计算机存储介质存储有显存资源的调度程序，所述显存资源的调度程序被至少一个处理器执行时实现权利要求1至7任一项所述的显存资源的调度方法的步骤。

一种显存资源的调度方法、装置及计算机存储介质

技术领域

[0001] 本发明实施例涉及图形处理器的显存管理技术，尤其涉及一种显存资源的调度方法、装置及计算机存储介质。

背景技术

[0002] 随着图形处理器(GPU,Graphic Processing Unit)的应用场景及需求的不断扩大，GPU中的显存容量逐渐成为影响GPU在执行应用处理相关任务的重要因素，也逐渐成为GPU在面对各式各样的应用处理时的瓶颈。因此，在有限的物理显存条件下，GPU在执行应用处理任务时需要尽可能地充分利用现有的显存资源。

发明内容

[0003] 有鉴于此，本发明实施例期望提供一种显存资源的调度方法、装置及计算机存储介质；能够灵活地调度显存资源，并且提高显存资源的利用效率。

[0004] 本发明实施例的技术方案是这样实现的：

[0005] 第一方面，本发明实施例提供了一种显存资源的调度方法，所述方法包括：

[0006] 在执行当前渲染任务过程中，响应于图形处理器GPU驱动程序检测到显存溢出事件的发生，统一调度器从非活跃渲染上下文中选择目标渲染上下文；其中，所述非活跃渲染上下文包括当前未执行的渲染上下文；

[0007] 所述统一调度器按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页；

[0008] 所述统一调度器将所述被切换的显存页切换出显存，并将所述显存中切换后的空余显存页分配至活跃渲染上下文；其中，所述活跃渲染上下文包括当前被执行的渲染上下文。

[0009] 第二方面，本发明实施例提供了一种显存资源的调度装置，所述装置包括：选择部分，确定部分以及第一分配部分，其中，

[0010] 所述选择部分，经配置为在执行当前渲染任务过程中，响应于图形处理器GPU驱动程序检测到显存溢出事件的发生，从非活跃渲染上下文中选择目标渲染上下文；其中，所述非活跃渲染上下文包括当前未执行的渲染上下文；

[0011] 所述确定部分，经配置为按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页；

[0012] 所述第一分配部分，经配置为将所述被切换的显存页切换出显存，并将所述显存中切换后的空余显存页分配至活跃渲染上下文；其中，所述活跃渲染上下文包括当前被执行的渲染上下文。

[0013] 第三方面，本发明实施例提供了一种显存资源的调度装置，其特征在于，所述装置包括：CPU、图形处理器GPU、显存以及存储器；其中，所述存储器存储有GPU驱动程序以及用于执行第一方面所述的显存资源的调度方法步骤的显存资源的调度程序；所述CPU，经配置

为在执行当前渲染任务过程中，基于所述存储器所存储的GPU驱动程序以及所述显存资源的调度程序对所述GPU的显存的资源进行调度。

[0014] 第四方面，本发明实施例提供了一种计算机存储介质，其特征在于，所述计算机存储介质存储有显存资源的调度程序，所述显存资源的调度程序被至少一个处理器执行时实现第一方面所述的显存资源的调度方法的步骤。

[0015] 本发明实施例提供了一种显存资源的调度方法、装置及计算机存储介质；通过引入针对显存的统一调度器来实现对显存资源的统一管理及调度，打破了不同渲染上下文之间彼此独立的限制，能够在发生显存溢出事件时，通过将被切换的显存页换出并将换出所获得空余显存页分配至执行当前渲染任务过程中的活跃渲染上下文以实现对不同渲染上下文的显存页进行换出换入操作，从而在显存溢出时，无需向CPU上报错误并等待已有渲染任务完成后释放显存资源以针对活跃渲染上下文的执行渲染操作，提高了显存资源的利用效率。

附图说明

[0016] 图1为本发明实施例提供的一种计算装置的组成示意图。

[0017] 图2为本发明实施例提供的详细说明图1中处理器、GPU和系统内存的实例实施方案的框图。

[0018] 图3为本发明实施例提供的一种显存资源的调度方法流程示意图。

[0019] 图4为本发明实施例提供的一种GPU驱动程序触发中断服务程序的流程示意图。

[0020] 图5为本发明实施例提供的一种显存溢出的实例示意图。

[0021] 图6为本发明实施例提供的一种显存换入换出示意图。

[0022] 图7为本发明实施例提供的一种显存资源的调度装置组成示意图。

[0023] 图8为本发明实施例提供的另一种显存资源的调度装置组成示意图。

[0024] 图9为本发明实施例提供的又一种显存资源的调度装置组成示意图。

具体实施方式

[0025] 下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述。

[0026] 针对前述出现的显存容量瓶颈情况，当前相关技术方案的主要解决思路是首先将GPU渲染任务所针对的渲染数据优先存放到系统内存中，当GPU开始执行渲染任务时，才将存储于系统内存的渲染数据搬移到显存中供GPU读取及处理，以此减少显存资源被耗尽的情况。比如图形执行管理器(GEM,Graphic Execution Manager)方案和转换表映射(TTM,Translation Table Maps)方案。

[0027] 需要说明的是，常见的系统内存容量通常为显存容量的数倍甚至数十倍，并且系统内存容量还可以借助虚拟内存技术使用硬盘的空间进行扩充；由此可知，系统内存容量由于远大于显存容量，相对于显存容量来说，可以几乎认为是“无限”的。而上述技术方案对于单任务渲染场景来说，由于可以优先使用“无限”的系统内存资源临时存放GPU执行渲染任务时所需的渲染数据。但对于多任务渲染场景来说，当GPU的显存被耗尽时，新的渲染任务就无法执行了，只能向CPU上报显存不够用的错误信息；这种情况的发生原因是对于目前

的相关方案中，每个渲染任务均对应一个渲染上下文，而多个渲染上下文则对应着多个渲染任务，不同的渲染上下文在显存内的分配与回收是完全独立的，且与当前渲染上下文紧密绑定，其他渲染上下文是不参与到当前渲染上下文的管理中。在这种情况下，若GPU执行了多个渲染上下文，而其中的某个渲染上下文占用了大量的GPU的显存资源且一直没有被释放掉，那么对于其他渲染上下文而言，不同的渲染上下文之间由于彼此独立，从而不能相互通知以让占用GPU显存资源的渲染任务释放部分其所占的GPU显存资源。这样的情况对于获取不到GPU显存资源的渲染上下文来说，只能向CPU上报错误退出，或者一直等待直到获取到显存资源后继续执行。根据上述针对目前相关技术方案的分析可以获知，目前的相关技术无法充分地利用GPU的显存资源，显存资源的利用效率低下。基于此，本发明实施例所阐述的技术方案期望通过打破不同渲染上下文之间彼此独立的限制以实现对显存资源进行统一管理及调度，从而能够充分利用GPU的显存资源。

[0028] 参见图1，其示出了能够实现本发明实施例技术方案的计算装置2，该计算装置2的实例包括但不限于：无线装置、移动或蜂窝电话(包含所谓的智能电话)、个人数字助理(PDA)、视频游戏控制台(包含视频显示器、移动视频游戏装置、移动视频会议单元)、膝上型计算机、桌上型计算机、电视机顶盒、平板计算装置、电子书阅读器、固定或移动媒体播放器等。在图1的实例中，该计算装置2可以包括：处理器6、系统内存10和GPU 12。计算装置2还可包含显示处理器14、收发器模块3、用户接口4和显示器8。收发器模块3和显示处理器14两者可为与处理器6和/或GPU 12相同的集成电路(IC)的部分，两者可在包含处理器6和/或GPU 12的一或多个IC的外部，或可形成于在包含处理器6和/或GPU 12的IC外部的IC中。

[0029] 为清楚起见，计算装置2可包含图1中未图示的额外模块或单元。举例来说，计算装置2可在其中计算装置2为移动无线电话的实例中包含扬声器和麦克风(两者均未在图1中示出)来实现电话通信，或在计算装置2为媒体播放器的情况下包含扬声器。计算装置2还可包含摄像机。此外，计算装置2中所示的各种模块和单元可能不是在计算装置2的每个实例中都是必需的。举例来说，在计算装置2为桌上型计算机或经装备以与外部用户接口或显示器连接的其它装置的实例中，用户接口4和显示器8可在计算装置2外部。

[0030] 用户接口4的实例包含(但不限于)轨迹球、鼠标、键盘和其它类型的输入装置。用户接口4还可为触摸屏，并且可作为显示器8的部分并入。收发器模块3可包含电路以允许计算装置2与另一装置或网络之间的无线或有线通信。收发器模块3可包含调制器、解调器、放大器和用于有线或无线通信的其它此类电路。

[0031] 处理器6可为微处理器，例如中央处理单元(CPU)，其经配置以处理供执行的计算机程序的指令。处理器6可包括控制计算装置2的运算的通用或专用处理器。用户可将输入提供到计算装置2，以致使处理器6执行一或多个软件应用程序。在处理器6上执行的软件应用程序可包含(例如)操作系统、文字处理器应用程序、电子邮件应用程序、电子表格应用程序、媒体播放器应用程序、视频游戏应用程序、图形用户接口应用程序或另一程序。另外，处理器6可执行用于控制GPU 12的运算的GPU驱动程序22。用户可经由一或多个输入装置(未图示)(例如，键盘、鼠标、麦克风、触摸垫或经由用户接口4耦合到计算装置2的另一输入装置)将输入提供到计算装置2。

[0032] 在处理器6上执行的软件应用程序可包含一或多个图形渲染指令，其指令处理器6来致使将图形数据渲染到显示器8。在一些实例中，所述软件指令可符合图形应用程序编程

接口(API),例如开放式图形库OpenGL API、开放式图形库嵌入系统(OpenGL ES)API、Direct3D API、X3D API、RenderMan API、WebGL API、开放式计算语言(OpenCLT M) 、RenderScript或任何其它异构计算API,或任何其它公用或专有标准图形或计算API。所述软件指令还可为针对无渲染算法(例如计算摄影、卷积神经网络、视频处理、科学应用程序等)的指令。为了处理图形渲染指令,处理器6可向GPU 12发出一或多个图形渲染命令(例如,通过GPU驱动程序22),以致使GPU 12执行图形数据的渲染中的一些或全部。在一些实例中,待渲染的图形数据可包含例如点、线、三角形、四边形、三角形带等图形图元的列表。

[0033] GPU 12可经配置以执行图形运算,从而将一或多个图形图元渲染到显示器8。因此,当在处理器6上执行的软件应用中的一者需要图形处理时,处理器6可将图形命令和图形数据提供到GPU 12以用于渲染到显示器8。图形数据可包含(例如)绘制命令、状态信息、图元信息、纹理信息等。在一些情况下,GPU 12可内置有高度并行结构,其提供比处理器6高效的对复杂图形相关运算的处理。举例来说,GPU 12可包含经配置以并行方式对多个顶点或像素进行运算的多个处理元件,例如着色器单元。在一些情况下,GPU 12的高度并行性质允许GPU 12比使用处理器6直接将场景绘制到显示器8更快速地将图形图像(例如,GUI和二维(2D)和/或三维(3D)图形场景)绘制到显示器8上。

[0034] 在一些情况下,可将GPU 12集成到计算装置2的母板中,从而形成集成显卡。在其它情况下,GPU 12可存在于图形卡上,所述图形卡安装在计算装置2的母板中的端口中,或可以其它方式并入在经配置以与计算装置2互操作的外围装置内,该图形卡也可称为独立显卡。GPU 12可包含一或多个处理器,例如一或多个微处理器、专用集成电路(ASIC)、现场可编程门阵列(FPGA)、数字信号处理器(DSP)或其它等效的集成或离散逻辑电路。GPU 12还可包含一或多个处理器核心,使得GPU 12可被称作多核处理器。

[0035] 在一些实例中,GPU 12可将完全形成的图像存储在系统内存10中。显示处理器14可从系统内存10检索图像,且输出致使显示器8的像素照亮以显示所述图像的值。显示器8可为计算装置2的显示器,其显示由GPU 12产生的图像内容。显示器8可为液晶显示器(LCD)、有机发光二极管显示器(OLED)、阴极射线管(CRT)显示器、等离子显示器或另一类型的显示装置。

[0036] 图2是进一步详细说明图1中处理器6、GPU 12和系统内存10的实例实施方案的框图。如图2所示,处理器6可执行至少一个软件应用程序18、图形API 20和GPU 驱动程序22,其中的每一者可为一或多个软件应用程序或服务。在一些实例中,图形API 20和GPU驱动程序22可实施为CPU 6的硬件单元。

[0037] 可供GPU 12使用的存储器可包含显存16,其可存储经渲染图像数据,例如像素数据,以及任何其它数据,因此,显存16还可被称为帧缓冲器。在具体实施过程中,比如在GPU 12集成到计算装置2的母板中以形成集成显卡的情况下,显存16可为系统内存10的部分;而在GPU 12存在于图形卡上,且所述图形卡安装在计算装置2的母板中的端口中,或可以其它方式并入在经配置以与计算装置2互操作的外围装置内的情况下,即GPU 12存在于独立显卡的情况下,显存16可与系统内存10分离,如图2中所示;需要说明的是,本发明实施例的技术方案可以应用于图2所示的实例方案,也就是说,在实施本发明实施例的技术方案时,为了清楚简洁地描述技术方案,通常可以认为显存16与系统内存10分离;当然,上述说明并不排除本发明实施例的技术方案应用于显存16作为系统内存10的部分的情况,后续不再赘

述。

[0038] 显存16存储GPU 12的目的地像素。每个目的地像素可与唯一屏幕像素位置相关联。在一些实例中，显存16可存储每个目的地像素的色彩分量和目的地α值。举例来说，显存16可存储每个像素的红色、绿色、蓝色、α(RGBA)分量，其中“RGB”分量对应于色彩值，并且“A”分量对应于目的地α值(例如，用于图像合成的不透明度值)。尽管图2将显存16和系统内存10说明为单独的存储器单元，但在其它实例中，例如集成显卡的情况下，显存16可以是系统内存10的一部分。此外显存16还可能够存储除像素之外的任何合适的数据。

[0039] 软件应用程序18可为利用GPU 12的功能性的任何应用程序。举例来说，软件应用程序18可为图形应用程序、操作系统、便携式制图应用程序、用于工程或艺术应用的计算机辅助设计程序、视频游戏应用程序或使用2D或3D图形的另一类型的软件应用程序。

[0040] 软件应用程序18可包含指令GPU 12渲染图形用户接口(GUI)和/或图形场景的一或多个绘制指令。举例来说，绘制指令可包含界定将由GPU 12渲染的一组一或多个图形图元的指令。在一些实例中，绘制指令可共同地界定用于GUI中的多个开窗表面的全部或部分。在额外实例中，所述绘制指令可共同地定义图形场景的全部或部分，所述图形场景包含在由应用程序定义的模型空间或世界空间内的一或多个图形对象。

[0041] 软件应用程序18可经由图形API 20调用GPU驱动程序22，以向GPU 12发出一或多个命令，以用于将一或多个图形图元渲染到可显示的图形图像中。举例来说，软件应用程序18可调用GPU驱动程序22，以向GPU 12提供图元定义。在一些情况下，图元定义可以例如三角形、矩形、三角形扇、三角形带等的绘制图元的列表的形式被提供到GPU 12。图元定义可包含指定与待呈现的图元相关联的一或多个顶点的顶点规格。所述顶点规格可包含每个顶点的位置坐标，且在一些情况下包含与顶点相关联的其它属性，例如色彩属性、法向量和纹理坐标。图元定义还可包含图元类型信息(例如，三角形、矩形、三角形扇、三角形带等)、缩放信息、旋转信息及类似者。

[0042] 基于由软件应用程序18向GPU驱动程序22发出的指令，GPU驱动程序22可调配指定供GPU 12执行的一或多个运算以便渲染图元的一或多个命令。当GPU 12接收到来自CPU 6的命令时，GPU 12可通过执行图形处理管线，以便对命令进行解码，并对图形处理管线进行配置以执行命令中所制定的渲染操作。此外，在执行渲染操作过程中，可以通过CPU 6调用GPU驱动程序22以对显存16的存储空间进行申请、数据拷贝以及释放等操作。

[0043] 基于上述针对图2所示实例的阐述，若期望实现通过打破不同渲染上下文之间彼此独立的限制，本发明实施例优选地在GPU 12的系统驱动层级，也就是GPU驱动程序22中引入针对显存16的统一调度器221，来实现对显存资源的统一管理及调度。在一些示例中，参见图3，其示出了本发明实施例提供的一种显存资源的调度方法，该方法可以包括：

[0044] S301：在执行当前渲染任务过程中，响应于GPU驱动程序22检测到显存溢出事件的发生，统一调度器221从非活跃渲染上下文中选择目标渲染上下文；其中，所述非活跃渲染上下文包括当前未执行的渲染上下文；

[0045] S302：统一调度器221按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页；

[0046] S303：统一调度器221将所述被切换的显存页切换出所述显存，并将所述显存中切换后的空余显存页分配至活跃渲染上下文；其中，所述活跃渲染上下文包括当前被执行的

渲染上下文。

[0047] 需要说明的是,通过统一调度器221执行上述技术方案,打破了不同渲染上下文之间彼此独立的限制,能够在发生显存溢出事件时,通过将被切换的显存页换出并将换出所获得空余显存页分配至执行当前渲染任务过程中的活跃渲染上下文以实现对不同渲染上下文的显存页进行换出换入操作,从而在显存溢出时,无需向CPU 6上报错误并等待已有渲染任务完成后释放显存资源以针对活跃渲染上下文的执行渲染操作,提高了显存资源的利用效率。

[0048] 对于图3所示的技术方案,在一些示例中,所述GPU驱动程序22检测到显存溢出事件的发生,可以包括:

[0049] GPU驱动程序22在为所述活跃渲染上下文分配显存时检测到显存剩余资源不足以为所述活跃渲染上下文进行分配;

[0050] 或者,在根据所述活跃渲染上下文执行当前渲染任务过程中,GPU驱动程序22根据由GPU 12中的存储管理单元(MMU,Memory Management Unit)121上报的中断信号触发中断服务程序;其中,所述中断信号包括所述GPU 12中的MMU在执行当前渲染任务过程中发现产生的中间数据导致显存资源不足时所反馈的中断信号。

[0051] 对于上述示例,具体来说,为了执行当前渲染任务,GPU驱动程序22首先就需要为被执行的活跃渲染上下文分配显存资源,若此时显存16中剩余的空闲资源不足以分配给活跃渲染上下文,那么就会导致显存溢出的情况出现,也就是说,上述显存溢出事件(后续简称为“第一显存溢出事件”)通常会发生在执行当前渲染任务过程的前段部分,响应于上述第一显存溢出事件,统一调度器221就可以按照图3所述的技术方案进行显存页的换出换入操作。

[0052] 此外,在为活跃渲染上下文分配至相应显存资源之后,当GPU 12根据所述活跃渲染上下文执行当前渲染任务的过程中,通常会产生一些中间数据缓存在显存16中,此时,如果中间数据过多,同样会导致显存溢出的情况出现,也就是说,上述显存溢出事件(后续简称为“第二显存溢出事件”)可能发生在执行当前渲染任务过程中的任意时刻,相应于第二显存溢出事件发生,统一调度器221同样可以按照图3所述的技术方案进行显存页的换出换入操作。

[0053] 具体地,对于上述第二显存溢出事件来说,当GPU 12根据所述活跃渲染上下文执行当前渲染任务的过程中,若产生的中间数据致使显存溢出的情况出现,举例来说,在GPU内部逻辑在动态访问显存过程中,若GPU 12中的MMU 121一旦发现所访问的显存已超出当前显存的界限后,会立即向主机端发送一个中断信号,主机端在捕获到GPU 12的MMU 121所发送的中断信号后,会触发GPU驱动程序22中预设的中断服务程序,该中断服务程序能够配合统一调度器221完成图3所示的显存页的换出换入操作。详细来说,参见图4,上述在根据所述活跃渲染上下文执行当前渲染任务过程中,GPU驱动程序22根据由GPU 12中的MMU 121上报的中断信号触发中断服务程序,其具体实施过程可以包括:

[0054] S41:GPU驱动程序22接收由GPU 12的MMU 121反馈的中断信号;其中,所述中断信号为GPU 12的MMU 121发现缺页中断状态时所反馈的信号,具体可以包括缺页中断的描述信息以及全局显存资源的使用信息;举例来说,缺页中断的描述信息可以包括当前缺页对应的渲染上下文、当前缺页对应的显存地址;全局显存资源的使用信息可以包括基于最近

最少使用(LRU,Least Recently Used)算法获得的当前显存使用的情况,比如可以被换出的显存资源以及不可以被换出的显存资源;

[0055] S42:GPU驱动程序22触发中断服务程序,并向所述统一调度器221传输所述缺页中断的描述信息以及全局显存资源的使用信息,以使得所述统一调度器221基于所述中断服务程序的触发根据所述缺页中断的描述信息以及全局显存资源的使用信息选择目标渲染上下文。

[0056] 对于图3所示的技术方案,在一些示例中,所述统一调度器221从非活跃渲染上下文中选择目标渲染上下文,可以包括:

[0057] 所述统一调度器221从所述非活跃渲染上下文中将当前被执行的渲染上下文之前最接近的已执行完毕的渲染上下文选择为所述目标渲染上下文。

[0058] 需要说明的是,渲染过程中,在当前被执行的渲染上下文之前最接近的已执行完毕的渲染上下文,也被称之为刚执行完毕的渲染上下文,在短时间内通常不会再次被用于执行渲染任务,因此,优先将目标渲染上下文选择为刚执行完毕的渲染上下文,能够避免短期内再次进行上述显存页的换出换入操作,提高了显存资源的使用效率。

[0059] 对于图3所示的技术方案,需要说明的是,在统一调度器221中,会针对每个渲染上下文均对应保存一张显存资源使用表,用于记录被对应的渲染上下文所存储的显存地址以及虚拟显存的使用情况,同时保存有被对应的渲染上下文中每个显存页的使用频次,比如最近最少使用情况,从而在完成选择目标渲染上下文后,确定被切换的显存页,基于此,在一些示例中,所述统一调度器221按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页,包括:

[0060] 所述统一调度器221从所述目标渲染上下文的显存页中,基于LRU算法选择最近最少使用的显存页作为所述被切换的显存页。

[0061] 基于上述图3所示的技术方案以及前述示例,当统一调度器221完成显存页换出换入操作之后,还需要针对换出的显存页进行妥善处理,否则在下一次调用目标渲染上下文执行渲染任务时,会再次引发并生成显存溢出的中断信号。基于此,在一些示例中,所述方法还包括:

[0062] 统一调度器221在系统内存10内为所述目标渲染上下文分配存储空间;

[0063] 统一调度器221将被切换的显存页同步至GPU 12的MMU 121,以使得GPU 12调用GPU 12的MMU 121完成地址转换;

[0064] 统一调度器221启动GPU 12的直接存储器访问(DMA,Direct Memory Access)122根据转换完成的地址将被切换的显存页切换至系统内存10中。

[0065] 完成上述示例后,当下次调用目标渲染上下文执行渲染任务时可以通过GPU 12的DMA 122将系统内存10保存的被切换的显存页调度至显存16中。

[0066] 此外,在一些示例中,在完成显存页的换入换出操作后,还可以包括:统一调度器221将目标渲染上下文对应的显存资源使用表进行更新,以此保证当切换到下一次渲染上下文过程中,显存访问的正确性。需要说明的是,在选择即将替换的显存页的过程中,可以将显存所保存的内容属性作为显存切换的依据。

[0067] 基于上述技术方案,参见图5所示的具体实施示例,显存16中已有的渲染上下文分别为CTX0、CTX1以及CTX2,当前执行渲染操作的活跃渲染上下文为CTX3,当为CTX3分配显存

时,发现显存16的剩余资源不足以完全分配给CTX3,会造成部分显存页(如图5中灰色方块所示)出现溢出,此时,统一调度器221可以将最近完成渲染任务的CTX1选为目标渲染上下文,并且从CTX1的显存页中根据LRU算法选择被切换出的显存页,如图5中的交叉线填充方块所示。统一调度器221可以将被切换出的显存页切换出显存,并且将切换所得到的显存16的空余空间分配给灰色方块所示的CTX3的显存页,如图6所示。对于交叉线填充方块所示的切换出的显存页CTX1',统一调度器221可以启动GPU 12的DMA 122将被切换的显存页切换至系统内存10。

[0068] 通过以上技术方案,采用统一调度器221对显存16的所有资源进行统一管理,从而可以灵活高效的进行显存16与系统内存10之间的换入换出操作,打破多个渲染上下文彼此独立的限制,使得显存资源得到充分的利用。

[0069] 基于前述技术方案相同的技术构思,参见图7,其示出了本发明实施例提供的一种显存资源的调度装置70,该装置70可以包括:选择部分701,确定部分702以及第一分配部分703,其中,

[0070] 所述选择部分701,经配置为在执行当前渲染任务过程中,响应于图形处理器GPU驱动程序检测到显存溢出事件的发生,从非活跃渲染上下文中选择目标渲染上下文;其中,所述非活跃渲染上下文包括当前未执行的渲染上下文;

[0071] 所述确定部分702,经配置为按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页;

[0072] 所述第一分配部分703,经配置为将所述被切换的显存页切换出显存,并将所述显存中切换后的空余显存页分配至活跃渲染上下文;其中,所述活跃渲染上下文包括当前被执行的渲染上下文。

[0073] 在一些示例中,所述GPU驱动程序检测到显存溢出事件的发生,包括:

[0074] 所述GPU驱动程序在为所述活跃渲染上下文分配显存时检测到显存剩余资源不足以为所述活跃渲染上下文进行分配;

[0075] 或者,在根据所述活跃渲染上下文执行当前渲染任务过程中,所述GPU驱动程序根据由所述GPU中的存储管理单元MMU 121上报的中断信号触发中断服务程序;其中,所述中断信号包括所述GPU中的MMU 121在执行当前渲染任务过程中发现产生的中间数据导致所述显存资源不足时所反馈的中断信号。

[0076] 基于上述示例,所述在根据所述活跃渲染上下文执行当前渲染任务过程中,所述GPU驱动程序根据由所述GPU中的存储管理单元MMU 121上报的中断信号触发中断服务程序,包括:

[0077] 所述GPU驱动程序接收由所述GPU的MMU 121反馈的中断信号;其中,所述中断信号为所述GPU的MMU 121发现缺页中断状态时所反馈的信号,且包括缺页中断的描述信息以及全局显存资源的使用信息;

[0078] 所述GPU驱动程序触发中断服务程序,并向所述显存资源的调度装置70传输所述缺页中断的描述信息以及全局显存资源的使用信息,以使得所述显存资源的调度装置70基于所述中断服务程序的触发根据所述缺页中断的描述信息以及全局显存资源的使用信息选择目标渲染上下文。

[0079] 在一些示例中,所述选择部分701,经配置为:从所述非活跃渲染上下文中将所述

当前被执行的渲染上下文之前最接近的已执行完毕的渲染上下文选择为所述目标渲染上下文。

[0080] 在一些示例中，所述确定部分702，经配置为从所述目标渲染上下文的显存页中，基于最近最少使用LRU算法选择最近最少使用的显存页作为所述被切换的显存页。

[0081] 在一些示例中，参见图8，所述装置70还可以包括：

[0082] 第二分配部分704，经配置为在系统内存内为所述目标渲染上下文分配存储空间；

[0083] 同步部分705，经配置为将所述被切换的显存页同步至所述GPU的MMU，以使得所述GPU调用所述GPU的MMU完成地址转换；

[0084] 启动部分706，经配置为启动所述GPU的DMA，根据转换完成的地址将所述被切换的显存页切换至所述系统内存中。

[0085] 在一些示例中，参见图9，所述装置70还可以包括：

[0086] 存储部分707，经配置为针对每个渲染上下文均对应保存一张显存资源使用表，用于记录被对应的渲染上下文所存储的显存地址以及虚拟显存的使用情况，同时保存有所述被对应的渲染上下文中每个显存页的使用频次；

[0087] 更新部分708，经配置为将所述目标渲染上下文对应的显存资源使用表进行更新，以此保证当切换到下一次渲染上下文过程中，显存访问的正确性。

[0088] 可以理解地，在本实施例中，“部分”可以是部分电路、部分处理器、部分程序或软件等等，当然也可以是单元，还可以是模块也可以是非模块化的。

[0089] 另外，在本实施例中的各组成部分可以集成在一个处理单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现，也可以采用软件功能模块的形式实现。

[0090] 所述集成的单元如果以软件功能模块的形式实现并非作为独立的产品进行销售或使用时，可以存储在一个计算机可读取存储介质中，基于这样的理解，本实施例的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台计算机设备(可以是个人计算机，服务器，或者网络设备等)或processor(处理器)执行本实施例所述方法的全部或部分步骤。而前述的存储介质包括：U盘、移动硬盘、只读存储器(ROM，Read Only Memory)、随机存取存储器(RAM，Random Access Memory)、磁碟或者光盘等各种可以存储程序代码的介质。

[0091] 因此，本实施例提供了一种计算机存储介质，所述计算机存储介质存储有显存资源的调度程序，所述显存资源的调度程序被至少一个处理器执行时实现上述技术方案中所述显存资源的调度方法步骤。

[0092] 需要说明的是：本发明实施例所记载的技术方案之间，在不冲突的情况下，可以任意组合。

[0093] 以上所述，仅为本发明的具体实施方式，但本发明的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本发明揭露的技术范围内，可轻易想到变化或替换，都应涵盖在本发明的保护范围之内。因此，本发明的保护范围应以所述权利要求的保护范围为准。

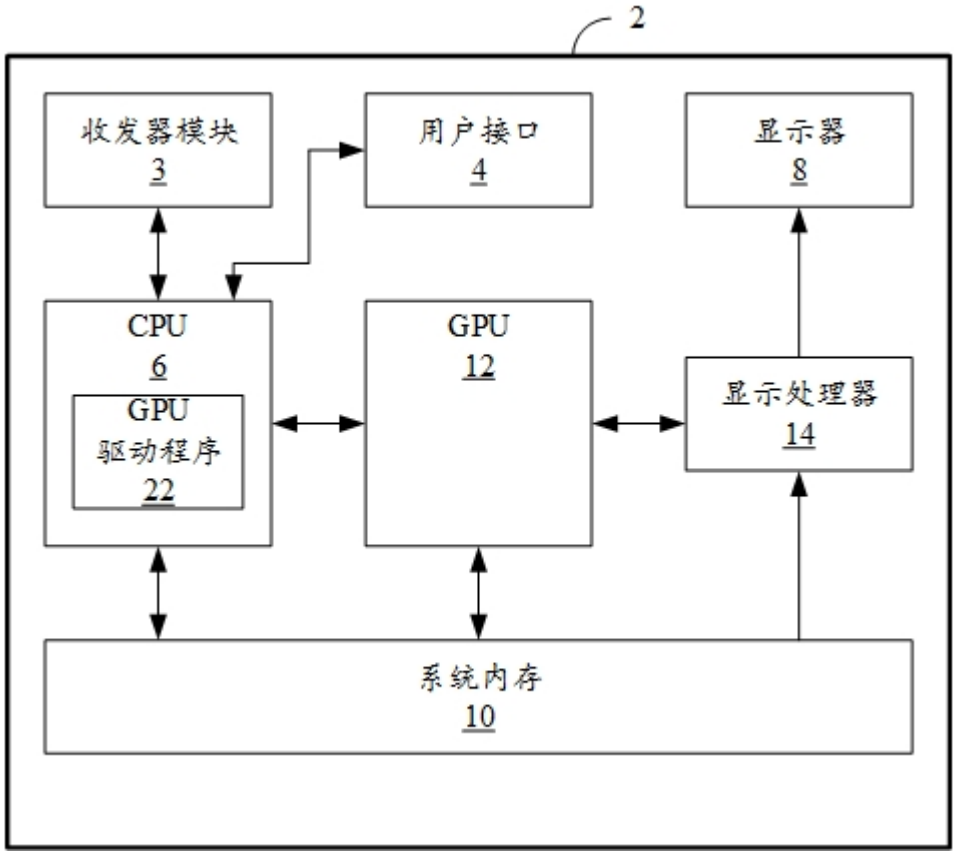


图 1

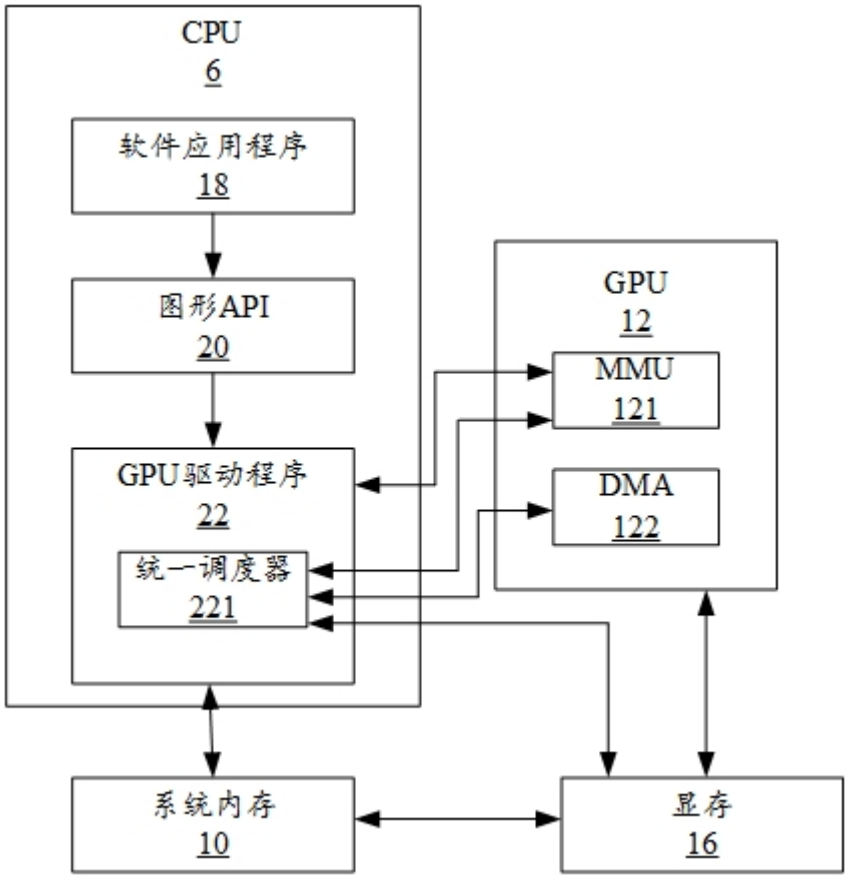

CPU
6
软件应用程序
18
图形API
20
GPU驱动程序
22
统一调度器
221
GPU
12
MMU
121
DMA
122
系统内存
10
显存
16

图 2

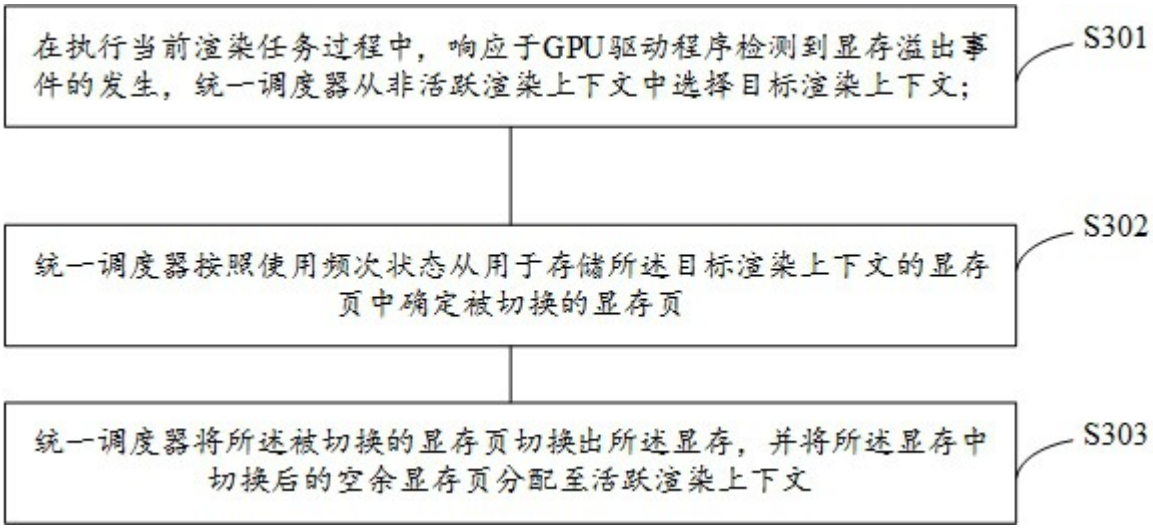

在执行当前渲染任务过程中，响应于GPU驱动程序检测到显存溢出事件的发生，统一调度器从非活跃渲染上下文中选择目标渲染上下文；
S301
统一调度器按照使用频次状态从用于存储所述目标渲染上下文的显存页中确定被切换的显存页
S302
统一调度器将所述被切换的显存页切换出所述显存，并将所述显存中切换后的空余显存页分配至活跃渲染上下文
S303

图 3

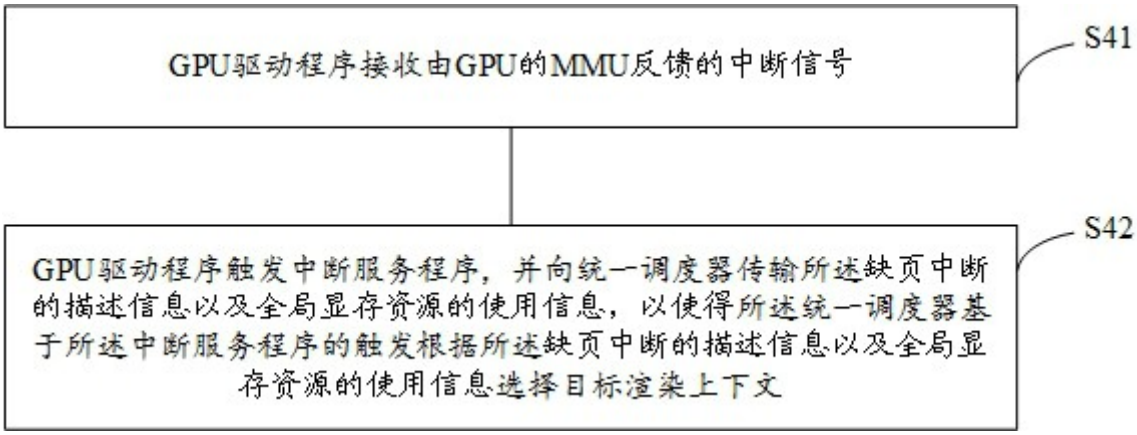

GPU驱动程序接收由GPU的MMU反馈的中断信号
S41
GPU驱动程序触发中断服务程序，并向统一调度器传输所述缺页中断的描述信息以及全局显存资源的使用信息，以使得所述统一调度器基于所述中断服务程序的触发根据所述缺页中断的描述信息以及全局显存资源的使用信息选择目标渲染上下文
S42

图 4

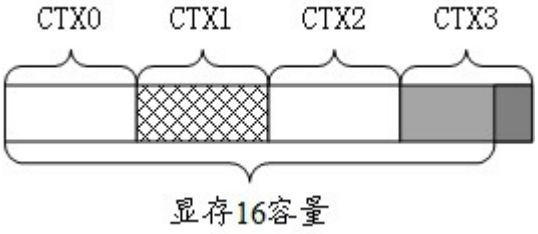

CTX0
CTX1
CTX2
CTX3
显存16容量

图 5

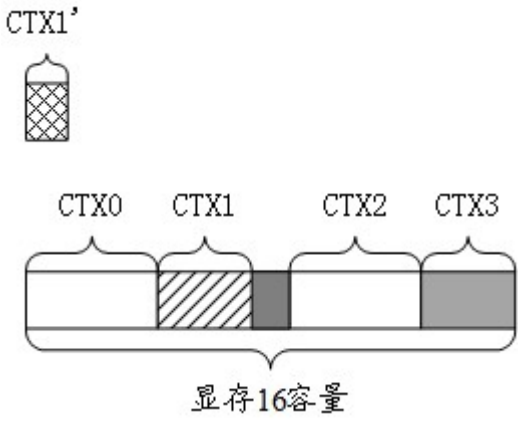

CTX1'
CTX0
CTX1
CTX2
CTX3
显存16容量

图 6


70
选择部分701
确定部分702
第一分配部分703
显存资源的调度装置

图 7


70
选择部分701
确定部分702
第一分配部分703
第二分配部分704
同步部分705
启动部分706
显存资源的调度装置

图 8

图 9