(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 113256478 A
(43)申请公布日 2021.08.13

(21)申请号 202110183183.5

(22)申请日 2021.02.10

(30)优先权数据
2002003.8 2020.02.13 GB
2002004.6 2020.02.13 GB

(71)申请人 畅想科技有限公司
地址 英国赫特福德郡

(72)发明人 杨喜乐

(74)专利代理机构 北京东方亿思知识产权代理有限责任公司 11258
代理人 董越

(51)Int.Cl.
G06T 1/20(2006.01)
G06T 1/60(2006.01)
G06F 9/38(2006.01)

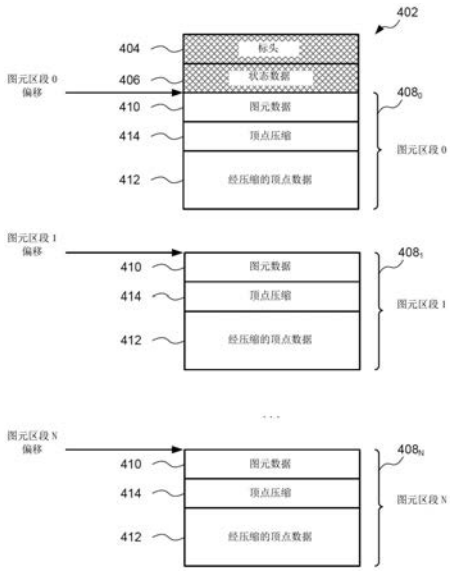
权利要求书3页 说明书23页 附图17页

(54)发明名称

用于在图形处理系统中存储图元的方法和图元块生成器

(57)摘要

用于在图形处理系统中存储图元的方法和图元块生成器。方法包括:接收与状态数据相关联的图元,状态数据定义图元将如何被渲染;确定与所接收的图元相关联的状态数据是否匹配当前图元块的状态数据;以及响应于确定所接收的图元的状态数据与当前图元块的状态数据匹配:基于一个或多个图元区段大小约束,确定是否要将所接收的图元添加到数据存储器中的当前图元块的当前图元区段;响应于确定要将所接收的图元添加到当前图元区段,将所接收的图元添加到当前图元区段;以及响应于确定不将所接收的图元添加到当前图元区段:输出当前图元区段;重新配置数据存储器以存储当前图元块的新图元区段;以及将所接收的图元添加到当前图元块的新图元区段。



CN 113256478 A

1.一种在图形处理系统中存储图元的方法(700),所述方法(700)包括:

接收与状态数据相关联的图元,所述状态数据定义所述图元将如何被渲染(702);

确定与所接收的图元相关联的所述状态数据是否与当前图元块的状态数据匹配(704);

响应于确定所接收的图元的所述状态数据与所述当前图元块的所述状态数据匹配:

基于一个或多个图元区段大小约束,确定是否要将所接收的图元添加到数据存储器中的所述当前图元块的当前图元区段(706);

响应于确定要将所接收的图元添加到所述当前图元区段,将所接收的图元添加到所述当前图元区段(710);

响应于确定不将所接收的图元添加到所述当前图元区段:

输出所述当前图元区段(712);

重新配置所述数据存储器以存储当所述前图元块的新图元区段(714);以及

将所接收的图元添加到所述当前图元块的所述新图元区段(710)。

2.根据权利要求1所述的方法(700),还包括响应于确定所接收的图元的所述状态数据不匹配所述当前图元块的所述状态数据:

输出所述当前图元块的所述当前图元区段(708);

重新配置所述数据存储器以存储具有与所接收的图元相关联的所述状态数据的新图元块的新图元区段(716);以及

将所接收的图元添加到所述新图元块的所述新图元区段(710)。

3.根据权利要求2所述的方法(700),其中重新配置所述数据存储器以存储具有与所接收的图元相关联的所述状态数据的新图元块的新图元区段包括清除与所述当前图元块相关的所述数据存储器的内容。

4.根据权利要求2所述的方法(700),其中与所述当前图元块相关联的所述状态数据被存储在所述数据存储器中;并且重新配置数据存储器以存储具有与所接收的图元相关联的所述状态数据的新图元块的新图元区段包括用与所接收的图元相关联的所述状态数据替换存储在所述数据存储器中的所述状态数据。

5.根据权利要求2所述的方法(700),其中所述当前图元块的标头被存储在所述数据存储器中;并且重新配置数据存储器以存储具有与所接收的图元相关联的所述状态数据的新图元块的新图元区段包括用所述新图元块的标头替换所述数据存储器中的所述标头,所述新图元块的所述标头指示所述新图元块包括单个图元区段。

6.根据权利要求1至5中任一项所述的方法(700),还包括根据压缩算法来压缩所输出的图元区段的全部或一部分。

7.如权利要求6所述的方法(700),还包括更新所输出的图元区段的压缩子区段以指示所输出的图元区段的全部或一部分已经被压缩和/或标识所述压缩算法的一个或多个参数。

8.根据权利要求1至5中任一项所述的方法(700),其中重新配置所述数据存储器以存储所述当前图元块的新图元区段包括清除与所述当前图元区段相关的所述数据存储器的所述内容。

9.根据权利要求1至5中任一项所述的方法(700),其中所述当前图元块的标头被存储

在所述数据存储器中;并且重新配置所述数据存储器以存储所述当前图元块的新图元区段包括更新所述数据存储器中的所述标头以反映新图元区段已经被添加到所述当前图元块。

10.根据权利要求1至5中任一项所述的方法(700),其中所接收的图元由一个或多个顶点形成;并且将所接收的图元添加到图元区段包括针对所接收的图元的每个新顶点将新顶点条目添加到所述图元区段的顶点数据子区段,以及将图元条目添加到所述图元区段的图元数据子区段。

11.根据权利要求10所述的方法(700),其中所述图元条目标识所述顶点数据子区段中的形成所述图元的所述顶点。

12.根据权利要求1至5中任一项所述的方法(700),其中所述当前图元块的标头被存储在所述数据存储器中;并且将所接收的图元添加到图元区段中包括更新所述数据存储器中的所述标头以反映已经将附加图元添加到所述当前图元区段。

13.根据权利要求1至5中的任一项所述的方法(700),还包括将所输出的图元区段或所压缩的输出图元区段存储在存储器中。

14.根据权利要求1至5中任一项所述的方法(700),其中所述一个或多个图元区段大小约束包括每个图元区段的图元的最大数目。

15.根据权利要求1至5中任一项所述的方法(700),其中所接收的图元由一个或多个顶点形成,并且所述一个或多个图元区段大小约束包括每个图元区段的顶点的最大数目。

16.一种在图形处理系统中生成渲染输出的方法,在所述图形处理系统中,所述渲染空间被划分成多个图块,所述方法包括:

变换多个图元以生成多个经过变换的图元;

针对所述多个经过变换的图元中的每一个执行权利要求1至5中的任一项所述的方法(700),以生成多个图元块;

针对所述多个图块中的每一个生成显示列表,所述显示列表包括信息,所述信息标识包括与渲染所述图块相关的至少一个图元的所述图元块,并且针对每个所标识的图元块标识与渲染所述图块相关的所述图元块的所述经过变换的图元;以及

基于在所述显示列表中标识为与渲染所述图块相关的所述经过变换的图元而渲染每个图块。

17.根据权利要求16所述的方法,其中标识与渲染所述图块相关的图元块的所述经过变换的图元的所述信息包括标识包括与渲染所述图块相关的至少一个图元的所述图元块的每个图元区段的信息,以及针对每个所标识图元区段标识所述图元区段中与渲染所述图块相关的所述图元的信息。

18.一种用于图形处理系统的图元块生成器(600),所述图元块生成器(600)包括:

数据存储器(602、808、908、1008),所述数据存储器被配置为存储:

当前图元块的当前图元区段(606);以及

所述当前图元块的状态数据(608),所述状态数据(608)定义所述当前图元块中的图元将如何被渲染;以及

图元块生成逻辑(604),所述图元块生成逻辑被配置为:

接收与状态数据(804、904、1004)相关联的图元(802、902、1002);

确定与所接收的图元(802、902、1002)相关联的所述状态数据(804、904、1004)是否匹

配所述当前图元块的所述状态数据(608);以及

响应于确定所接收的图元(802、902)的所述状态数据(804、904)与所述当前图元块的所述状态数据(608)匹配:

基于一个或多个图元区段大小约束条件,确定是否要将所接收的图元(802、902、1002)添加到所述当前图元区段(606);

响应于确定要将所接收的图元(802)添加到所述当前图元区段(606),将所接收的图元添加到所述当前图元区段(606);以及

响应于确定不将所接收的图元(902)添加到所述当前图元区段(606):

输出所述当前图元区段(606);

重新配置所述数据存储器(602、908)以存储所述当前图元块的新图元区段;以及

将所接收的图元(902)添加到所述当前图元块的所述新图元区段。

19.一种计算机可读存储介质,其上存储有计算机可读指令,所述计算机可读指令在计算机系统处执行时使所述计算机系统执行根据权利要求1至5中任一项所述的方法。

20.一种计算机可读存储介质,其上存储有如权利要求18所述的图元块生成器(600)的计算机可读描述,所述计算机可读描述当在集成电路制造系统中处理时致使所述集成电路制造系统制造体现所述图元块生成器(600)的集成电路。

用于在图形处理系统中存储图元的方法和图元块生成器

技术领域

[0001] 本申请涉及图形处理系统，并且更具体地涉及用于在图形处理系统中存储图元的数据结构、方法和图元块生成器。

背景技术

[0002] 图形处理系统被配置成例如从在计算机系统上运行的应用程序(例如游戏应用程序)接收图形数据，并对来自图形数据的图像进行渲染以提供渲染输出。例如，应用程序可以生成场景的3D模型并输出表示场景中的对象的几何结构数据。特定来说，应用程序可以使用一个或多个图元(即，简单的几何形状，例如但不限于可以被应用纹理的矩形、三角形、线和点)来表示每个对象，所述多个图元由一个或多个顶点的位置限定。在这些状况下，由应用程序输出的几何结构数据可以包含标识每个顶点的信息(例如顶点在世界空间中的坐标)和指示由顶点形成的图元的信息。然后，图形处理系统将接收到的几何结构数据转换成可以在屏幕上显示的图像。

[0003] 图形处理系统可以例如实施即时模式渲染(IMR)或基于图块的渲染(TBR)。在IMR中，将整个场景作为整体进行渲染。与此对比，在TBR中，使用被划分成被称为图块的子区段的渲染空间对场景进行渲染，其中可针对每个图块独立地执行渲染过程的至少一部分。图块可以具有任何合适形状，但通常为矩形(其中术语“矩形”包含正方形)。TBR的优点为，可以在渲染期间使用快速、片上存储器以用于颜色、深度和模板缓冲区操作，这与IMR相比允许显著减少系统存储器带宽，而不需要足够大以同时存储用于整个场景的数据的片上存储器。

[0004] TBR涉及两个关键阶段：几何结构处理阶段；以及光栅化阶段。在几何结构处理阶段期间，将从应用程序(例如游戏应用程序)接收的几何结构数据(例如限定图元的顶点)从世界空间坐标变换成屏幕空间坐标。然后创建至少部分地落在图块的边界内的经过变换的图元(例如三角形)的每图块列表。在光栅化阶段期间，对每个图块单独地渲染(即，将经过变换的图元映射到像素并且针对图块中的每个像素标识颜色)。这可以包括标识哪个(哪些)图元在每个像素处是可见的。接着可以由每个像素处的可见图元的外观确定所述像素的颜色，所述可见图元可以由应用于所述像素处的纹理和/或在所述像素上运行的像素着色器程序限定。像素着色器程序描述将针对给定像素执行的操作。对每个图块单独地渲染会使图形处理系统能够在光栅化阶段中对特定图块进行渲染时仅检索与所述图块相关的经过变换的图元数据，这会使针对存储器(例如中间缓冲区)的带宽要求保持为低。一旦已经针对图块中的每个像素标识出颜色值，就将图块的颜色值写出到存储器(例如帧缓冲区)。一旦已经渲染整个场景(即，一旦已经针对所有图块的像素确定了颜色值)，场景就可以例如显示在屏幕上。

[0005] 图1示出示例TBR图形处理系统100。系统100包括存储器102_1、102_2、102_3、102_4、几何结构处理逻辑104和光栅化逻辑106。存储器102_1、102_2、102_3和102_4中的两个或多个可以在同一物理存储器单元中实现。

[0006] 几何结构处理逻辑104实施TBR的几何结构处理阶段。几何结构处理逻辑104包括变换逻辑108、图元块生成器110，以及平铺引擎112。变换逻辑108从应用程序(例如游戏应用程序)接收几何结构数据(例如顶点、图元和/或补片)并将几何结构数据变换到渲染空间(例如屏幕空间)中。变换逻辑108还可以执行例如裁剪和剔除的功能以移除落在视锥外的几何结构数据(例如图元或补片)，和/或应用所属领域的技术人员所知的照明/属性处理。

[0007] 图元块生成器110将经过变换的图元(即，与其相关的经过变换的几何结构数据)分组成图元块，并将图元块存储在存储器102_2中。图元块是一种数据结构，其中一个或多个图元(例如，与之相关的经过变换的几何结构数据)被存储在一起。在图元块中存储图元可以允许将一组图元的经过变换的几何结构数据更有效地存储在存储器102_2中。具体而言，用于图元的经过变换的几何结构数据通常包括用于多个顶点的经过变换的顶点信息，其中这些顶点可以在多个图元之间共享(或为多个图元所共有)。因此，在同一图元块中的多个图元共享顶点的状况下，与所述顶点有关的数据只需在所述图元块中存储一次。

[0008] 可以使用任何合适的方法或技术将经过变换的图元分组成图元块。例如，在一些状况下，可以基于经过变换的图元到达图元块生成器110的顺序将经过变换的图元分组成图元块。在这些情况下，每个图元块可以具有最大大小(例如，就位或字节而言)、可以属于图元块的图元的最大数目、和/或可以属于图元块的顶点的最大数目。然后，图元块生成器110可被配置为将图元添加到当前图元块，直到达到最大值中的一个或多个为止。

[0009] 在其他情况下，可基于图元在渲染空间中的位置将所述图元分组成图元块，使得在渲染空间中具有空间上相似的位置的图元在同一图元块中。例如，可以将渲染空间划分成多个宏区域，这些宏区域可以包围多个图块(例如，被划分成1024个32×32的图块的1024×1024的渲染空间可以具有16个256×256的宏区域)，并且图元块生成器110可以被配置成维护每一个宏区域的图元块。然后，当图元块生成器110接收到图元时确定所述图元至少部分地落在哪个或哪些宏区域内。如果图元块生成器110确定图元至少部分地落在仅一个宏区域内，则图元块生成器110可以将图元(即，与图元相关的经过变换的几何结构数据)放置在该宏区域的图元块中。如果图元块生成器110确定图元落在一个以上的宏区域内，则图元块生成器110可以被配置成(i)选择图元落在的宏区域中的一个宏区域(例如，第一宏区域)，并且将图元(即，与之相关的经过变换的几何结构数据)放置在所选宏区域的图元块中；或者(ii)将图元(即，与之相关的经过变换的几何结构数据)放置在图元至少部分地落在的宏区域中的每一个宏区域的图元块中。

[0010] 将图元块与标识图元块在存储器中的位置的信息一起提供给平铺引擎112。平铺引擎112从经过变换的几何结构数据生成用于每个图块的经过变换的图元的列表，所述经过变换的图元至少部分地落在所述图块内。所述列表可以被称作显示列表或经过变换的显示列表。在一些状况下，经过变换的显示列表可以包括到与至少部分地落在图块内的图元相关的经过变换的几何结构数据(例如，顶点数据)的指针或链路。例如，图2示出了用于图块的示例显示列表202，所述示例显示列表包括用于每个图元块的图元块条目204、206，每个图元块包括至少部分地落在所述图块的边界内的至少一个图元。每个图元块条目204、206包括标识图元块在存储器中的位置(例如，图元块在存储器中的地址)的信息208和标识图元块的哪些图元至少部分地落在图块的边界内的信息210。如图2所示，标识图元块的哪些图元至少部分地落在图块内的信息可以是掩码的形式，所述掩码包括图元块中的每一个

图元的位，指示所述图元是否至少部分地落在图块的边界内。

[0011] 返回图1，光栅化逻辑106实现TBR的光栅化阶段。具体而言，光栅化逻辑106通过从存储器102_3获取图块的显示列表，然后从存储器102_2获取落在图块内的由所述图块的显示列表指示的图元的经过变换的几何结构数据，以逐图块的方式渲染图元；并基于经过变换的几何结构数据来渲染所述图块的图元。

[0012] 在一些状况下，光栅化逻辑106可包括光栅化器114、隐藏表面移除(HSR)逻辑116和纹理化/着色逻辑118。在这些状况下，光栅化器114从存储器102_3获取每个显示列表，并且针对每个显示列表从存储器102_2获取针对落在由对应的显示列表指定的图块内的图元的经过变换的几何结构数据，并将每个图元转换成图元片段的集合。术语“片段”在本文中用于意指采样点处的图元的样本，所述样本将被处理以对图像的像素进行渲染。在一些示例中，可以存在像素到片段的一对一映射。不过，在其他示例中，片段可以多于像素，并且此过采样可以允许像素值的较高质量渲染，例如通过促进可以应用于多个片段以用于对每个像素值进行渲染的抗混叠和其他滤波器。

[0013] 然后将特定图块的图元片段提供到HSR逻辑116，所述HSR逻辑通过对图元片段执行深度测试而移除隐藏(例如，被其他图元片段隐藏)的图元片段。接着将其余片段(在隐藏表面移除之后)传递到纹理化/着色逻辑118，所述纹理化/着色逻辑对图元片段执行纹理化和/或着色以确定被渲染图像的像素值。接着，将图块的被渲染像素值存储在存储器102_4(例如，帧缓冲区)中。

[0014] 光栅化逻辑106处理每个图块，并且当整个图像已经被渲染并存储在存储器102_4(例如，帧缓冲区)中时，图像可以从图形处理系统100输出并以任何合适的方式使用，例如，显示在显示器上、存储在存储器中或传输到另一设备等。在片段在由纹理化/着色逻辑118处理之前由HSR逻辑116处理的意义上，图1所展示的TBR图形处理系统100是“推迟的”渲染系统。在其他示例中，图形处理系统可能并非推迟的渲染系统，在此状况下，将会在将HSR应用于片段之前将纹理化/着色应用于那些片段。

[0015] 尽管几何结构处理逻辑在图中示为与光栅化逻辑分离，但在一些实施方案中，几何结构处理逻辑和光栅化逻辑可共享一些资源。例如，图形处理系统可以使用统一着色方法，其中相同的物理执行单元可以用于执行在几何结构处理阶段中使用的指令(例如，执行顶点处理)并执行在光栅化阶段中使用的指令(例如，执行片段处理)。

[0016] 下文描述的实施方案仅以示例的方式提供，并且不限制解决用于在图形处理系统中存储图元的已知方法和系统的任何或所有缺点的实现方式。

发明内容

[0017] 提供本发明内容是为了介绍在以下详细描述中进一步描述的一些概念。本发明内容不旨在标识所要求保护的主题的关键特征或必要特征，也不旨在用于限制所要求保护的主题的范围。

[0018] 本文描述了用于在图形处理系统中存储图元的数据结构、方法和图元块生成器。所述方法包括：接收与状态数据相关联的图元，所述状态数据定义所述图元将如何被渲染；确定与所接收的图元相关联的所述状态数据是否匹配当前图元块的状态数据；以及响应于确定所接收的图元的所述状态数据与所述当前图元块的所述状态数据匹配：基于一个或多

个图元区段大小约束，确定是否要将所接收的图元添加到数据存储器中的所述当前图元块的当前图元区段；响应于确定要将所接收的图元添加到所述当前图元区段，将所接收的图元添加到所述当前图元区段；以及响应于确定不将所接收的图元添加到所述当前图元区段：输出所述当前图元区段；重新配置数据存储器以存储所述当前图元块的新图元区段；以及将所接收的图元添加到所述当前图元块的所述新图元区段。

[0019] 第一方面提供了一种在图形处理系统中存储图元的方法，所述方法包括：接收与状态数据相关联的图元，所述状态数据定义所述图元将如何被渲染；确定与所接收的图元相关联的所述状态数据是否匹配当前图元块的状态数据；以及响应于确定所接收的图元的所述状态数据与所述当前图元块的所述状态数据匹配：基于一个或多个图元区段大小约束，确定是否要将所接收的图元添加到数据存储器中的所述当前图元块的当前图元区段；响应于确定要将所接收的图元添加到所述当前图元区段，将所接收的图元添加到所述当前图元区段；以及响应于确定不将所接收的图元添加到所述当前图元区段：输出所述当前图元区段；重新配置数据存储器以存储所述当前图元块的新图元区段；以及将所接收的图元添加到所述当前图元块的所述新图元区段。

[0020] 所述方法还可以包括，响应于确定所接收的图元的状态数据不匹配当前图元块的状态数据：输出当前图元块的当前图元区段；重新配置所述数据存储器以存储具有与所接收的图元相关联的状态数据的新图元块的新图元区段；以及将所接收的图元添加到所述新图元块的新图元区段。

[0021] 重新配置所述数据存储器以存储具有与所接收的图元相关的状态数据的新图元块的新图元区段可包括清除与当前图元块相关的数据存储器的内容。

[0022] 与当前图元块相关联的状态数据可以存储在数据存储器中；并且重新配置数据存储器以存储具有与所接收的图元相关联的状态数据的新图元块的新图元区段可以包括用与所接收的图元相关联的状态数据替换存储在数据存储器中的状态数据。

[0023] 当前图元块的标头可以存储在数据存储器中；并且重新配置数据存储器以存储具有与所接收的图元相关联的状态数据的新图元块的新图元区段可包括用新图元块的标头替换数据存储器中的标头，所述标头指示新图元块包括单个图元区段。

[0024] 所述方法还可以包括根据压缩算法压缩输出图元区段的全部或一部分。

[0025] 所述方法还可以包括更新输出图元区段的压缩子区段以指示输出图元区段的全部或一部分已被压缩和/或标识压缩算法的一个或多个参数。

[0026] 重新配置数据存储器以存储当前图元块的新图元区段可以包括清除与当前图元区段相关的数据存储器的内容。

[0027] 当前图元块的标头可以存储在数据存储器中；并且重新配置数据存储器以存储当前图元块的新图元区段可以包括更新数据存储器中的标头以反映新图元区段已被添加到当前图元块。

[0028] 所接收的图元可以由一个或多个顶点形成；并且将所接收的图元添加到图元区段可以包括：针对所接收的图元的每个新顶点，将新顶点条目添加到图元区段的顶点数据子区段；以及将图元条目添加到图元区段的图元数据子区段。

[0029] 图元条目可标识顶点数据子区段中形成图元的顶点。

[0030] 当前图元块的标头可以存储在数据存储器中；并且将所接收的图元添加到图元区

段可以包括更新数据存储器中的标头以反映已经将附加图元添加到当前图元区段。

[0031] 所述方法还可以包括将输出图元区段或所压缩的输出图元区段存储在存储器中。

[0032] 所述一个或多个图元区段大小约束可以包括每个图元区段的图元的最大数目。

[0033] 所接收的图元可以由一个或多个顶点形成，并且所述一个或多个图元区段大小约束包括每个图元区段的顶点的最大数目。

[0034] 将所接收的图元添加到当前图元区段可以包括将所接收的图元的信息添加到数据存储器中的当前图元区段。

[0035] 第二方面提供一种在图形处理系统中生成渲染输出的方法，在所述图形处理系统中所述渲染空间被划分成多个图块，所述方法包括：变换多个图元以生成多个经过变换的图元；针对所述多个经过变换的图元中的每一个执行所述第一方面的所述方法以生成多个图元块；针对所述多个图块中的每一个生成显示列表，所述显示列表包括信息，所述信息标识包括与渲染所述图块相关的至少一个图元的所述图元块，并且针对每个所标识的图元块标识与渲染所述图块相关的所述图元块的所述经过变换的图元；以及基于在所述显示列表中标识的与渲染所述图块相关的用于所述图块的经过变换的图元来渲染每个图块。

[0036] 标识与渲染所述图块相关的图元块的经过变换的图元的信息可以包括标识包括与渲染所述图块相关的至少一个图元的所述图元块的每个图元区段的信息，以及针对每个所标识的图元区段，标识所述图元区段中与渲染所述图块相关的所述图元的信息

[0037] 第三方面提供了一种用于图形处理系统的图元块生成器，所述图元块生成器包括：数据存储器，所述数据存储器被配置为存储：当前图元块的当前图元区段；以及当前图元块的状态数据，所述状态数据定义当前图元块中的图元将如何被渲染；以及图元块生成逻辑，所述图元块生成逻辑被配置为：接收与状态数据相关联的图元；确定与所接收的图元相关联的状态数据是否与当前图元块的状态数据匹配；以及响应于确定所接收的图元的状态数据与当前图元块的状态数据匹配：基于一个或多个图元区段大小约束，确定是否要将所接收的图元添加到当前图元区段；响应于确定要将所接收的图元添加到所述当前图元区段，将所接收的图元添加到所述当前图元区段；以及响应于确定不将所接收的图元添加到当前图元区段：输出当前图元区段；重新配置所述数据存储器以存储所述当前图元块的新图元区段；以及将所接收的图元添加到当前图元块的新图元区段。

[0038] 所述图元块生成逻辑可以被进一步配置为，响应于确定所接收的图元的状态数据与所述当前图元块的状态数据不匹配：输出当前图元块的当前图元区段和当前图元块的状态数据；重新配置数据存储器以存储具有与所接收的图元相关的状态数据的新图元块的新图元区段；以及将所接收的图元添加到所述新图元块的新图元区段。

[0039] 数据存储器可被进一步配置为存储当前图元块的标头；并且所述图元块生成逻辑可被配置为，通过用新图元块的标头替换所述数据存储器中的标头，重新配置所述数据存储器以存储具有与所接收的图元相关联的状态数据的新图元块的新图元区段，所述新图元块的标头指示所述新图元块包括单个图元区段。

[0040] 数据存储器可被进一步配置为存储当前图元块的标头；并且所述图元块生成逻辑可被配置为通过更新所述数据存储器中的标头以指示新图元区段已被添加到所述当前图元块，重新配置所述数据存储器以存储所述当前图元块的新图元区段。

[0041] 第四方面提供一种图元块生成器，所述图元块生成器被配置为执行第一方面或第

二方面的方法。

[0042] 第五方面提供一种图形处理系统，所述图形处理系统包括第三方面或第四方面的图元块生成器。

[0043] 所述图形处理系统还可以包括平铺引擎，所述平铺引擎包括：平铺逻辑，所述平铺逻辑被配置为：接收多个图元块，每个图元块包括一个或多个图元区段，每个图元区段包括一个或多个图元；以及针对所接收的图元块的每个图元区段，确定所述图元区段的图元是否至少部分地落在图块的边界内；以及显示列表生成器，所述显示列表生成器被配置为：针对包括至少部分地落在所述图块的所述边界内的至少一个图元的每个图元块，将标识所述图元块的信息添加到所述图块的显示列表；针对每个所标识的图元块，将信息添加到所述显示列表，所述信息标识包括至少部分地落在所述图块的所述边界内的至少一个图元的所述图元块的每个图元区段；以及针对每个所标识的图元区段，将信息添加到所述显示列表，所述信息标识所述图元区段中的至少部分地落在所述图块的所述边界内的图元。

[0044] 本文所述的图元块生成器、平铺引擎和图形处理系统可以以集成电路上的硬件来体现。可以提供一种在集成电路制造系统处制造体现本文所述的图元块生成器、平铺引擎或图形处理系统的集成电路的方法。可以提供集成电路限定数据集，当在集成电路制造系统中进行处理时，该集成电路限定数据集将系统配置为制造体现本文所述的平铺引擎、图元块生成器或图形处理系统的集成电路。可以提供一种非暂态计算机可读存储介质，在其上存储有本文所述的图元块生成器、平铺引擎或图形处理系统的计算机可读描述，当在集成电路制造系统中进行处理时，该可读描述致使所述集成电路制造系统制造体现图元块生成器、平铺引擎或图形处理系统的集成电路。

[0045] 可以提供一种集成电路制造系统，该集成电路制造系统包括：非暂态计算机可读存储介质，其上存储有本文所述的图元块生成器、平铺引擎或图形处理系统的计算机可读描述；布局处理系统，其被配置为处理计算机可读描述，以生成体现图元块生成器、平铺引擎或图形处理系统的集成电路的电路布局描述；以及集成电路生成系统，其被配置为根据电路布局描述制造体现平铺引擎或图形处理系统的集成电路。

[0046] 可以提供用于执行如本文所述的方法的计算机程序代码。可以提供其上存储有计算机可读指令的非暂态计算机可读存储介质，当在计算机系统处执行时，所述计算机可读指令使计算机系统执行如本文所述的方法。

[0047] 如对本领域的技术人员显而易见的，上述特征可以适当地组合，并且可以与本文所述的示例的任何方面组合。

附图说明

[0048] 现在将参考附图详细描述示例，在附图中：

[0049] 图1是示例的基于图块的渲染图形处理系统的框图；

[0050] 图2是第一示例显示列表的示意图；

[0051] 图3是一组示例图元块的示意图，其中每个图元块包括单个图元区段；

[0052] 图4是具有多个图元区段的示例图元块的示意图；

[0053] 图5是示例图元数据和顶点数据子区段的示意图；

[0054] 图6是被配置为用于生成图4的图元块的示例图元块生成器的框图；

[0055] 图7是在图4的图元块中存储图元的示例方法的流程图；

[0056] 图8是示出根据图7的方法的第一示例图元的处理的示意图；

[0057] 图9是示出根据图7的方法的第二示例图元的处理的示意图；

[0058] 图10是示出根据图7的方法的第三示例图元的处理的示意图；

[0059] 图11是示例平铺引擎的框图；

[0060] 图12是示出边界框平铺方法的示意图；

[0061] 图13是示出由图11的平铺引擎生成的第一示例显示列表的示意图；

[0062] 图14是示出由图11的平铺引擎生成的第二示例显示列表的示意图；

[0063] 图15是示出由图11的平铺引擎生成的第三示例显示列表的示意图；

[0064] 图16是生成图块的显示列表的示例方法的流程图；

[0065] 图17是包括图6的图元块生成器和图11的平铺引擎的示例图形处理系统的框图；

[0066] 图18是其中可以实现本文所述的图元块生成器、平铺引擎和/或图形处理系统的示例计算机系统的框图；并且

[0067] 图19是用于生成体现本文所述的图元块生成器、平铺引擎和/或图形处理系统的集成电路的示例集成电路制造系统的框图。

[0068] 附图示出了各种示例。技术人员将理解，附图中所示的元件边界(例如，框、框的组，或其他形状)表示边界的一个示例。在一些示例中，情况可能是一个元件可以被设计为多个元件，或者多个元件可以被设计为一个元件。在适当的情况下，贯穿各附图使用共同附图标记来指示相似特征。

具体实施方式

[0069] 通过示例的方式给出以下描述，以使本领域的技术人员能够制造和使用本发明。本发明不限于本文中描述的实施方案，并且对所公开的实施方案的各种修改对于所属领域的技术人员而言将是显而易见的。仅通过示例的方式描述实施方案。

[0070] 如上所述，在一些TBR图形处理系统中，在几何结构处理阶段中生成的经过变换的图元被分组成图元块，并且所述图元块是存储在存储器中的图元块。然后，每个图块的显示列表可包括用于每个图元块的图元块条目，所述图元区段包括落在所述图块的边界内的至少一个图元。每个图元块条目可以标识所述图元块以及所述图元块的与渲染所述图块相关的图元(例如，所述图元块中的至少部分落在所述图块的边界内的图元)。在光栅化阶段期间，通过检索每个图块的显示列表，然后检索并处理在显示列表中被标识为与渲染所述图块相关的经过变换的图元来单独地渲染所述图块。这可以包括，对于每个图块，获取在对应的显示列表中标识的每个图元块；然后处理在显示列表中被标识为与渲染所述图块相关的每个所取的图元块的图元。

[0071] 在一些情况下，每个经过变换的图元与指定将如何在光栅化阶段中光栅化所述图元的状态数据相关联。在这些情况下，可以以任何方式将图元分组成图元块，只要(i)图元块中的所有图元共享相同的状态数据；以及(ii)图元块中的顶点的数目不超过顶点阈值和/或图元块中的图元的数目不超过图元阈值。顶点阈值和/或图元阈值可以基于图形处理系统的硬件限制。例如，在一些情况下，顶点阈值可以基于可以存储在临时存储器中的顶点的最大数目。注意，虽然此类系统对于申请人来说可能是已知的，但是这并不是承认此类系

统是公知的。

[0072] 现在参考图3，其示出了可以在此类系统中生成的示例图元块302_0、302_1、302_2。每个图元块302_0、302_1、302_2包括标头区段304、状态数据区段306和图元区段308。

[0073] 标头区段304包括关于图元块的信息，例如但不限于图元块中的图元的数目和/或图元块中的顶点的数目。

[0074] 状态数据区段306包括图元块中的图元的状态数据。状态数据可以被描述为用于渲染图元块中的图元的方法。例如，状态数据可以包括但不限于标识深度比较模式、混合状态、纹理状态和/或图元类型的信息。

[0075] 图元区段308包括一个或多个图元的几何结构数据。在一些情况下，图元区段308可以包括图元数据子区段310和顶点数据子区段312。图元数据子区段310可包括用于图元块中的每个图元的图元条目，其标识形成所述图元的顶点。在一些情况下，可对顶点数据子区段312中的每个顶点指派索引，并且每个图元条目可包括顶点索引列表。索引可以是对于图元块是局部的索引(其在此可以被称为局部索引)。例如，如果图元为三角形且图元块中的第一图元由与索引0、1和3相关联的顶点形成，则所述图元的图元条目可列出顶点索引0、1和3。每个顶点索引可充当指向顶点数据子区段312的与所述顶点相关的部分的指针。

[0076] 顶点数据子区段312包括用于形成图元块中的图元的每个顶点的顶点条目。每个条目可包括描述对应顶点在渲染空间中的位置数据(例如，渲染空间中的一组坐标，例如X、Y和Z坐标)。每个条目还可包括用以描述顶点的外观的一组属性，例如纹理坐标(U,V)和/或应用于顶点的基色。顶点数据可以以压缩或非压缩格式存储在顶点数据子区段312中。任何合适的压缩算法或技术可用于压缩顶点数据。

[0077] 在顶点数据可以压缩格式存储到顶点数据子区段312中的情况下，图元部308还可以包括顶点压缩子区段314，其标识顶点数据是否已经被压缩，并且如果是，则标识用于压缩顶点数据的压缩算法的参数/配置。

[0078] 在一些图形处理系统中，尤其是硅面积有限的低预算系统中，用于在图元被存储在主存储器中之前临时存储图元的存储器已变得更小。这降低了顶点阈值和/或图元阈值，这导致更小的图元块。较小的图元块降低了压缩顶点数据的压缩比。这还意味着不是来自同一绘制调用的所有图元都可以被置于同一图元块中，从而导致具有相同状态数据的多个连续的图元块。

[0079] 已经做出努力以通过在存储器(例如查找表)中存储不同的状态数据组合并且在存储器中存储指示相关状态数据的位置的信息(例如到查找表中的索引)来减少重复的状态数据。例如，状态数据可以包括标识多个参数的状态的信息，其中每个参数由多个位限定。不是在图元块中明确地包括每个参数的信息，而是状态数据的每个可能的组合可以被存储在存储器中的状态数据表中，并且每个图元块的状态数据区段306可以包括指向状态数据表的条目之一的索引或指针。虽然这减少了存储的重复数据的量，但是它不能消除重复数据。

[0080] 因此，本文描述了图元块结构、以及用于在此类图元块中存储图元的方法和图元块生成器，其中每个图元块包括单个状态数据区段和一个或多个图元区段，其中每个图元区段包括用于一个或多个图元的经过变换的几何结构数据。每个图元块区段可具有最大数目的图元和/或顶点。本文描述的图元块格式减少了状态数据重复，因为它允许更多图元共

享相同的状态数据;并且还允许将图元压缩并存储在更小的组中。

[0081] 此外,在一些情况下,光栅化逻辑106的纹理化/着色逻辑118可以使用一个或多个SIMD(单指令多数据)处理器来实现,因为纹理化/着色逻辑通常将相同的变换(例如,相同的着色器)应用于多个顶点/片段。如本领域技术人员所知,SIMD处理器包括多个处理元件,每个处理元件对不同的数据集执行相同的操作。处理输入数据集合的每个处理元件被称为SIMD处理器的“通道”。当每个通道为“满”(即,正在处理数据)时,SIMD处理器最高效地操作。在一些情况下,纹理化/着色逻辑118的SIMD处理器可以包括32个通道。将图3所示结构的图元存储在图元块中通常导致纹理化/着色逻辑118的SIMD通道不满和/或可能需要时间来获得SIMD通道的数据并将其放在一起。由于本文描述的图元块格式允许每个图元块包括更多个图元,因此其使得在光栅化阶段中用来自处于一图元块的不同图元区段中的图元的片段来填充SIMD通道(例如,在纹理化/着色阶段期间)更容易。

[0082] 因此,本文所描述的图元块结构允许图形处理系统在有利时(例如,当在几何结构处理阶段中生成所述图标块时)利用较小图元/顶点组,并且在有利时(例如,当在光栅化阶段中填充SIMD通道时)利用较大图元/顶点组的益处。

[0083] 现在参考图4,其示出了根据实施方案的示例图元块402。与图3的图元块302_0、302_1、302_2类似,图元块402包括标头区段404和状态数据区段406。然而,虽然图3的图元块302_0、302_1、302_2仅包括单个图元区段308,但是图4的图元块402可以包括多个图元区段408(例如,N个图元区段,其中N是大于或等于一的整数)。所有的图元区段408共享相同的状态数据。

[0084] 标头区段404包括关于图元块的信息,例如但不限于块中的图元区段的数目和每个图元区段中的图元和/或顶点的数目。在一些情况下,标头区段404还可以包括每个图元区段的偏移地址。

[0085] 与图3的状态数据区段306类似,状态数据区段406描述了图元块中的图元将如何被渲染。例如,状态数据可以包括但不限于标识深度比较模式、混合状态、纹理状态和/或图元类型的信息。在一些情况下,状态数据区段406可以包括标识多个状态参数中的每一个的值的信息。在其他情况下,不同的状态参数组合可以存储在存储器(例如,查找表)中,并且图元块402的状态数据区段406可以包括指向状态参数的特定组合的指针或索引。在某些情况下,与存储实际状态参数值相反,在状态数据区段中存储指向特定状态参数组合的指针可以显著地减小状态数据区段的大小。

[0086] 每个图元区段408_0、408_1...408_N可对应于图3的图元区段308。例如,每个图元区段408_0,408_1...408_N可包括共享相同状态数据的一个或多个经过变换的图元的经过变换的几何结构数据。每个图元区段中的经过变换的图元的数目可由每个图元区段的图元的最大数目和/或每个图元区段的顶点的最大数目限制。例如,每个图元区段可具有最多64个图元和/或最多32或64个顶点。可基于图形处理系统的硬件限制且具体来说是其图元块生成器来设置图元的最大数目和/或顶点的最大数目。例如,每个图元区段的图元和/或顶点的最大数目可基于图元块生成器可临时存储的图元和/或顶点的最大数目。

[0087] 与图3的图元区段类似,每个图元区段408_0、408_1...图4的408_N包括图元数据子区段410和顶点数据子区段412,它们可以对应于以上参考图3所描述的图元数据子区段310和顶点数据子区段312。与图元区段中的每个图元有关的数据可以被添加到图元数据子区段

410，并且与形成或定义图元区段中的至少一个图元的每个顶点有关的数据可以被添加到顶点数据子区段412。

[0088] 可为图元区段中的每个顶点指派从0到n-1的索引，其中n为图元区段中的顶点的数目。例如，添加到图元区段的第一顶点可被分配索引0，添加到图元区段的第二顶点可被分配索引1，等等。由于索引对于图元区段是局部的，所以指派给图元区段中的顶点的索引可称为局部索引。局部索引可用于标识与顶点数据子区段412中的对应顶点相关的信息的位置。应注意，指派给图元区段中的顶点的局部索引可不同于指派给顶点的全局索引。例如，渲染中的顶点可以被分配全局索引，所述全局索引反映渲染中的顶点的排序，或者反映渲染中的一组顶点内的顶点的排序。现在请参照图5，其显示了包括两个三角形图元502与504的示例图元区段的图元数据与顶点数据子区段410、412。第一图元502由具有全局索引V1、V2、V3的顶点定义，并且第二图元504由具有全局索引V7、V8、V9的顶点定义。这六个顶点被添加到图元块，并且可以分别被分配局部索引V0、V1、V2、V3、V4和V5。

[0089] 顶点数据子区段412可包括每个顶点的顶点条目，所述顶点条目形成或定义图元区段408_0、408_1…408_N中的图元。例如，在图5中，在图元区段中存在六个顶点(分别具有全局索引V1、V2、V3、V7、V8、V9和局部索引V0、V1、V2、V3、V4、V5)，因此顶点数据子区段412可包含六个顶点条目506_0、506_1、506_2、506_3、506_4、506_5。

[0090] 如图5所示，每个顶点项目506_0、506_1、506_2、506_3、506_4、506_5可包含标识对应顶点在渲染空间中的位置或定位的信息(例如，顶点在渲染空间中的X、Y和Z坐标)。每个顶点条目506_0、506_1、506_2、506_3、506_4、506_5还可包括一组一个或多个属性以描述顶点的外观，例如纹理坐标(U,V)和/或应用于顶点的基色。

[0091] 顶点条目可以以压缩或未压缩的形式存储在顶点数据区段中。可以以不同的方式压缩(或不压缩)同一图元块的不同图元区段的顶点条目。例如，第一图元区段的顶点条目可以被解压缩，同一图元块的第二图元区段的顶点条目可以使用第一压缩算法来压缩，并且同一图元块的第三图元区段的顶点条目可以使用不同的第二压缩算法来压缩。这允许为每个图元区段选择最佳压缩算法。

[0092] 图元数据子区段410可以包括用于图元块中的每个图元的图元条目。例如，在图5的示例中，有两个图元502与504，因此有两个图元条目508_0，508_1，每个图元一个条目。如图5所示，每个图元条目可以包括标识顶点数据子区段412中的形成所述图元的顶点的信息。例如，在图5中，第一图元502的图元条目508_0标识具有局部索引V0、V1和V2的顶点，并且第二图元504的图元条目508_1标识具有局部索引V3、V4和V5的顶点。当在光栅化阶段中渲染图元时，从顶点数据子区段检索与所述图元相关联的顶点条目，并将其用于渲染所述图元。例如，为了渲染图5的示例中的第一图元502，从顶点数据子区段412检索具有局部索引V0、V1和V2的顶点条目，并且使用这些顶点条目来渲染所述图元。

[0093] 在顶点条目可以以压缩形式存储在顶点数据子区段412中的情况下，每个图元区段还可以包括顶点压缩子区段414，其标识顶点条目是否已经被压缩，以及可选地，标识使用了哪种压缩算法或技术来压缩顶点条目，和/或关于顶点数据子区段412中的顶点条目的压缩的其他信息。

[0094] 图元块402的每个图元区段408_0、408_1…408_N可以与所述图元块402的其他图元区段分开地写入存储器和从存储器读取。例如，如图4所示，存储器中每个图元区段408_0、

$408_1 \cdots 408_N$的开始可由从图元块中的点(例如,从标头部分404的开始或从状态数据区段406的结束)偏移的图元区段来标识。在一些情况下,图元块402的图元区段408_0、$408_1 \cdots 408_N$可连续地或背对背地存储在存储器中,使得每个图元区段408_0、$408_1 \cdots 408_N$的开始或偏移可从图元块中的先前图元区段的大小确定。例如,第二图元区段可以存储在第一图元区段之后的下一个可寻址存储器块处。在其他情况下,图元区段408_0、$408_1 \cdots 408_N$可不连续或背靠背存储在存储器中。例如,在一些情况下,在存储器中在图元块的第一图元区段和第二图元区段之间可存在其他数据。

[0095] 图元块生成器

[0096] 现在参考图6,其示出了被配置为生成具有图4所示的结构或格式的图元块的示例图元块生成器600。图元块生成器600包括数据存储器602和图元块生成逻辑604。

[0097] 数据存储器602是被配置为临时存储当前图元块的当前图元区段606、当前图元块的状态数据608和当前图元块的标头610的存储设备。当前图元区段606被配置为存储一个或多个图元。当前图元区段606可以具有上述任何格式。例如,当前图元区段606可以包括图元数据子区段、顶点数据子区段和可选的顶点压缩子区段。如上文所描述,状态数据608描述在光栅化阶段中图元将如何被渲染。当接收到集合中的第一图元时,当前图元块的状态数据608可以初始地被设置为无效值,以便启动新的图元块。

[0098] 图元块生成逻辑604被配置为接收各自与状态数据相关联的图元,所述状态数据标识在光栅化阶段中图元将如何被渲染。图元块生成逻辑604被配置为,对于每个所接收的图元,基于所述图元的状态数据和当前图元块的状态数据608,确定所述图元是否能够被添加到当前图元块。如果图元块生成逻辑604确定所接收的图元可添加到当前图元块,则图元块生成逻辑604被配置为基于一个或多个大小约束(例如,图元的最大数目和/或顶点的最大数目)来确定所接收的图元是否可添加到当前图元块的当前图元区段606。如果图元块生成逻辑604确定所接收的图元块可以被添加到当前图元块的当前图元块区段606,则图元块生成逻辑604将所接收的图元块添加到当前图元块区段606。然而,如果图元块生成逻辑604确定所接收的图元不能被添加到当前图元区段606,则图元块生成逻辑604被配置为致使当前图元区段606被输出(例如,存储于存储器(例如,外部或芯片外存储器)中),致使数据存储设备602经重新配置以存储当前图元块的新图元区段,并且将所接收的图元添加到当前图元块的新图元区段。

[0099] 图元块生成逻辑604可以包括状态数据比较逻辑612、图元区段分析逻辑614和控制器616。状态数据比较逻辑612被配置为将与所接收的图元相关联的状态数据与当前图元块的状态数据608进行比较,以确定它们是否匹配并将比较结果通知给控制器616。如上所述,状态数据描述了图元将如何被渲染。状态数据可以包括多个状态参数,每个状态参数可以具有多个值中的一个。在这些情况下,确定所接收的图元的状态数据与当前图元块的状态数据608匹配可以包括确定所接收的图元的多个状态参数中的每一个是否具有与当前图元块的对应状态参数相同的值。在一些情况下,如果状态数据比较逻辑612确定状态数据不匹配,则状态数据比较逻辑612可以向控制器616提供所接收的图元的状态数据,如下所述,所述状态数据可以用作新的图元块的状态数据。在其他情况下,控制器616或数据存储602可以直接或通过其他手段接收基本状态数据。

[0100] 图元区段析逻辑614配置成基于一个或多个图元区段大小约束来确定是否可以将

所接收的图元添加到当前图元区段606，并且向控制器616通知该确定。例如，图元区段分析逻辑614可以被配置为如果添加所接收的图元将不会违反一个或多个图元区段大小约束中的任何一个，则确定所接收的图元可以被添加到当前图元区段。大小约束可以基于数据存储器的大小和/或数据存储器的被分配用于存储当前图元区段606的部分。在一些情况下，一个或多个图元大小约束可包括图元区段中的图元的最大数目和/或图元区段中的顶点的最大数目。在这些情况下，如果将图元添加到当前图元区段将导致超过最大值的任一个，则图元区段分析逻辑614可确定所接收的图元不能被添加到当前图元区段。

[0101] 确定将所接收的图元添加到当前图元区段是否将违反图元的最大数目可以包括确定当前图元区段中的图元的数目，并且将图元的数目与图元的最大数目进行比较。如果当前图元区段中的图元数目小于最大值，则添加所述图元将不会违反所述大小约束。

[0102] 确定将所接收的图元添加到当前图元区段是否将违反当前图元区段中的顶点的最大数目可包括标识所接收的图元相对于当前图元区段的新顶点的数目。如上所述，图元可共享顶点。因此，如果所接收的图元的顶点中的一个或多个由已在当前图元区段中的至少一个图元共享，则所接收的图元的顶点中的一个或多个可能已在当前图元区段中。因此，确定将所接收的图元添加到当前图元区段是否将违反当前图元区段中的顶点的最大数目可包括通过将所接收的图元的顶点与当前图元块中的顶点进行比较来标识所接收的图元的新顶点的数目，并且接着确定将所标识数目的新顶点添加到当前图元区段是否将违反当前图元区段中的顶点的最大数目。

[0103] 在一些情况下，除了将所确定的结果提供给控制器616之外，图元区段分析逻辑614可以被配置为将所接收的图元提供给控制器616。在其他情况下，控制器616或数据存储器602可以直接或通过其他方式接收图元，以便能够将图元添加到当前图元块。

[0104] 控制器616可以被配置成基于从状态数据比较逻辑612和图元区段分析逻辑614接收的通知来控制所接收的图元在数据存储602和外部存储器中的存储。具体地，控制器616可以被配置为：

[0105] (i)将图元添加到当前图元区段：如果状态数据比较逻辑612指示所接收的图元的状态数据与当前图元块的状态数据匹配；图元区段分析逻辑614指示：所接收的图元可以被添加到当前图元区段中，而不违反一个或多个图元区段大小约束条件-使得所接收的图元被添加到当前图元区段606中；

[0106] (ii)开始新图元区段并将图元添加到新图元区段：如果状态数据比较逻辑612指示所接收的图元的状态数据与当前图元块的状态数据匹配；而图元区段分析逻辑614指示在不违反一个或多个图元区段大小约束的情况下不能将所接收的图元添加到当前图元区段中->：存储在数据存储器602中的当前图元区段被输出(例如存储在存储器620中)，数据存储器602被重新配置以存储当前图元块的新图元区段(例如清除当前图元区段606的内容)，并且所接收的图元被添加到新图元区段；以及

[0107] (iii)开始新图元块并将图元添加到其第一图元区段：如果状态数据比较逻辑612指示所接收的图元的状态数据不匹配当前图元块的状态数据->原因：数据存储器602中的数据与要输出的当前图元块(例如，存储在存储器620中)相关，数据存储器被重新配置为存储具有所接收的图元的状态数据的新图元块的新图元区段，以及要添加到新图元块的新图元区段的所接收的图元。

[0108] 在一些情况下，将所接收的图元添加到数据存储器中的当前图元区段可以包括针对与所接收的图元相关联的每个新顶点，将新顶点条目添加到当前图元区段的顶点数据子区段；以及将所述图元的新图元条目添加到图元数据子区段，所述图元条目标识顶点数据子区段中与所述图元相关的顶点。将图元添加到当前图元区段还可包括更新用于当前图元块的标头610以反映已将附加图元添加到当前图元区段，并且如果已添加一个或多个新顶点，则反映新顶点的数目。

[0109] 在一些情况下，图元块生成器600还可以包括压缩引擎618，其被配置为接收从数据存储602输出的图元区段，并且在将图元区段存储在存储器中之前压缩图元区段的全部或一部分。例如，压缩引擎618可被配置为压缩图元区段的顶点数据子区段。用于压缩所接收的图元区段的全部或一部分的压缩算法可以是预先配置的，或者可以基于所接收的图元区段中的数据来动态地选择。压缩引擎618还可被配置为更新图元区段的顶点压缩子区段以标识图元区段的全部或部分已被压缩，并且任选地标识哪一压缩算法用于压缩相关部分，和/或关于相关部分的压缩的其他信息。

[0110] 现在参考图7，其示出了用于在图形处理系统的存储器中存储经过变换的图元的示例方法，以便在可以由图6的图元块生成器600实现的渲染阶段中使用。

[0111] 在图7的方法700中，数据存储器用于临时存储标头、状态数据和当前图元块的当前图元区段。如果所接收的图元具有与当前图元块相同的状态数据，则将所接收的图元添加到数据存储器中的当前图元区段，并且可以在不违反图元区段大小约束的情况下将所接收的图元存储在当前图元区段中。如果状态数据匹配，但在当前图元区段中存储所接收的图元会违反大小约束，则输出当前图元区段，并且重新配置数据存储器以存储当前图元块的新图元区段，并将所接收的图元添加到其中。如果状态数据不匹配，则输出数据存储器中与当前图元块相关的数据，并且数据存储器被重新配置以存储与具有接收图元的状态数据的新图元块相关的数据，并且接收图元被添加到新图元块的第一图元区段。

[0112] 方法700开始于步骤702，其中接收与状态数据相关联的经过变换的图元。接收经过变换的图元可以包括接收定义图元的经过变换的几何结构数据(或标识几何结构数据的信息)。在一些情况下，定义经过变换的图元的经过变换的几何结构数据可包括用于定义所述图元的每个顶点的标识所述顶点在渲染空间中的位置的信息(例如，渲染空间中的X、Y和Z坐标)。对于每个顶点，经过变换的几何结构数据还可以包括其他信息，例如但不限于，描述顶点的外观的一组属性，例如纹理坐标(U,V)和/或应用于顶点的基色。一旦接收到图元，方法700就进行到步骤704。

[0113] 在步骤704处，确定与所接收的图元相关联的状态数据是否匹配与当前图元块相关联的状态数据。如上所述，状态数据描述了图元将如何被渲染。状态数据可以包括多个状态参数，每个状态参数可以具有多个值中的一个。在这些情况下，确定所接收的图元的状态数据与当前图元块的状态数据匹配可以包括确定形成所接收的图元的状态数据的多个状态参数中的每一个是否具有与当前图元块的状态数据608中的对应状态参数相同的值。如果在步骤704中确定所接收的图元的状态数据与当前图元块的状态数据匹配，则所述方法进行到步骤706。然而，如果在步骤704中确定所接收的图元的状态数据与当前图元块的状态数据不匹配，则方法700进行到步骤708。

[0114] 在步骤706处，确定是否可以将所接收的图元添加到数据存储器中的当前图元区

段而不违反一个或多个图元区段大小约束。如上所述，一个或多个大小约束被配置成控制图元区段的大小。一个或多个大小约束可以基于用于在将经过变换的图元输出到存储器之前临时存储经过变换的图元的数据存储的大小。在一些情况下，所述一个或多个大小约束可包括图元区段中的图元的最大数目和/或图元区段中的顶点的最大数目。上文描述了确定是否将通过将图元添加到数据存储器中的当前图元块而违反图元的最大数目，或确定是否将通过将所接收的图元添加到数据存储器中的当前图元块而违反顶点的最大数目的方法。如果在步骤706处确定通过将所接收的图元添加到当前图元区段中，没有违反图元区段大小约束，则方法700进行到步骤710。然而，如果在步骤706中确定通过将所接收的图元添加到数据存储器中的当前图元区段中会违反一个或多个图元区段大小约束，则方法700进行到步骤712。

[0115] 在步骤710处，将所接收的图元添加到当前图元区段。如上所述，在一些情况下，将所接收的图元添加到数据存储器中的当前图元区段可以包括：针对与所接收的图元相关联的每个新顶点(例如，所接收的图元的尚未在当前图元区段中的每个顶点)，将新顶点条目添加到当前图元区段的顶点数据子区段；以及将所述图元的新图元条目添加到当前图元区段的图元数据子区段，所述新图元条目标识与所述图元有关的顶点数据子区段中的顶点。将图元添加到当前图元区段还可包括更新当前图元块的标头610以反映已将附加图元添加到当前图元区段，并且如果已添加一个或多个新顶点，则将新顶点的数目添加到图元区段。一旦所接收的图元被添加到数据存储器中的当前图元区段，方法700结束。

[0116] 在步骤712处，在确定所接收的图元的状态数据与当前图元块的状态数据匹配，但将所接收的图元存储在当前图元块的当前图元区段中会违反一个或多个图元区段大小约束之后，输出与当前图元区段相关的数据存储器中的数据。在一些情况下，当前图元区段的数据可以被输出到外部存储器(例如，存储在外部存储器中)。在一些情况下，在被存储在外部存储器中之前，图元区段的全部或一部分可以被压缩。例如，在一些情况下，在被存储在存储器中之前，其顶点数据子区段可被压缩。可以使用任何适当的压缩算法或技术来压缩图元区段的全部或一部分。在将图元区段的全部或一部分存储在存储器中之前对其进行压缩的情况下，可以修改图元区段以包括已经对图元区段的全部或一部分进行了压缩的信息，并且可选地包括使用了哪种压缩算法来压缩图元区段的该部分的信息，和/或关于图元区段的该部分的压缩的其他信息。一旦已经输出了数据存储器中与当前图元区段相关的数据，方法700就进行到步骤714。

[0117] 在步骤714处，数据存储器被重新配置以存储当前图元块的新图元区段。重新配置数据存储器以存储当前图元块的新图元区段可以包括清除数据存储器中当前图元区段的内容，并更新标头610以指示已经将附加图元区段添加到当前图元块。一旦数据存储器被重新配置以存储当前图元块的新图元区段，方法700就进行到步骤710，在该步骤中，所接收的图元块被添加到数据存储器中的当前图元区段606。

[0118] 在步骤708处，在确定所接收的图元的状态数据不匹配当前图元块的状态数据之后，输出数据存储器中与当前图元块相关的数据。数据存储器中与当前图元块相关的数据可以包括标头610、状态数据608和当前图元区段606。数据存储器中与当前图元块相关的数据可以被输出到存储器。如上所述，关于步骤712，在将与当前图元区段相关的数据存储在存储器中之前，可以压缩图元区段的全部或一部分。可以使用任何合适的压缩方法或技术

来压缩图元区段的全部或一部分。优选地，使用无损压缩方法或技术。一旦数据存储器602中与当前图元块相关的数据被输出。方法700进行到步骤716。

[0119] 在步骤716处，数据存储器被重新配置以存储新的图元块的第一图元区段，所述图元块具有所接收的图元块的状态数据。重新配置数据存储器以存储新的图元块的第一图元区段可包括清除数据存储器中的当前图元区段606的内容，用与所接收的图元相关联的状态数据替换数据存储器602中的状态数据608，以及用新的图元块的标头(其可包括指示所述图元块包括当前不具有图元或顶点的单个图元区段的信息)替换数据存储器中的标头610。一旦数据存储器602已经被重新配置以存储新的图元块的第一图元区段，方法700就进行到步骤710，在该步骤中，所接收的图元被添加到当前图元区段(在这种情况下，其表示新的图元块的第一图元区段)。

[0120] 现在参考图8至图10，其示出了如何根据图7的方法处理图元的示例。具体而言，图8表示在当前的图元块的当前的图元部存储图元的示例，图9表示在当前的图元块的新的图元部存储图元的示例，图10表示在新的图元块的新的图元部存储图元的示例。

[0121] 特别地，图8示出了由具有全局索引V0、V1和V3的顶点形成的图元802。图元802与状态数据804ABCD相关联。如806所示，在接收图元802时，数据存储808包括用于与状态数据ABCD相关联的当前图元块的当前图元区段。当前图元区段包括一个图元(P0)和三个顶点(具有局部索引V0、V1、V2)。在此实例中，每个图元区段的最大图元数为三个，并且每个图元区段的最大顶点数为五个。图元802的状态数据804与当前图元块的状态数据匹配，因此图元802可以被添加到当前图元块。此外，将图元802添加到当前图元区段将不会违反图元区段大小限制中的任一者，因为如果将图元802添加到当前图元区段，则将仅存在两个图元和四个顶点，因为图元802的顶点(V0、V1)中的两者已在当前图元区段中。因此，如810所示，图元802被添加到当前图元区段。具体而言，顶点条目(V3＝X3、Y3、Z3、U3、V3)被添加到图元802的新顶点(具有本地索引V3)的当前图元区段，并且图元条目(P1＝V0、V1、V2)被添加到图元802的标识相关顶点的当前图元区段。在这种情况下，全局索引匹配局部索引，然而，在其他示例中，情况可能不是这样。

[0122] 图9示出了由具有全局索引V3、V4和V5的顶点形成的图元902。图元902与状态数据904ABCD相关联。如906所示，在接收到图元902时，数据存储908包括与状态数据ABCD相关联的当前图元块的当前图元区段。当前图元区段包括一个图元(P0)和三个顶点(具有局部索引V0、V1、V2)。在此实例中，每个图元区段的最大图元数为三个，并且每个图元区段的最大顶点数为五个。图元902的状态数据904与当前图元块的状态数据匹配，因此图元902可以被添加到当前图元块。然而，将图元902添加到当前图元区段将违反每个图元区段的顶点的最大数目，因为如果将图元902添加到当前图元区段，则图元区段中将存在六个顶点，因为图元902的所有三个顶点(具有全局索引V3、V4、V5的顶点)都是新的(即，尚未在当前图元区段中)。因此，当前图元区段的内容或数据被输出(例如，到存储器)，当前图元区段被重新配置以存储新图元区段(即，内容被刷新)，并且图元902被添加到当前图元区段。如910处所示，此导致当前图元区段包含图元902的每个顶点(具有被指派局部索引V0、V1及V2的全局索引V3、V4、V5)的顶点条目及标识相关顶点(具有局部顶点V0、V1及V2的顶点)的图元902的图元条目；并且更新所述标头以指示所述图元块具有包括一个图元和三个顶点的第二图元区段。应注意，在此实例中，全局顶点索引不匹配局部顶点索引。具体而言，全局顶点V3、V4、V5

分别成为新图元区段中的局部顶点V0、V1和V2。

[0123] 图10展示由具有全局索引V0、V1和V3的顶点形成的图元1002。图元1002与状态数据1004DEDE相关联。如1006处所示，在接收到图元1002时，数据存储1008包括与状态数据ABCD相关联的当前图元块的当前图元区段。当前图元区段包括一个图元(P0)和三个顶点(具有局部顶点V0、V1、V2)。在此实例中，每个图元区段的最大图元数为三个，并且每个图元区段的最大顶点数为五个。图元1002的状态数据1004与当前图元块的状态数据不匹配，因此图元1002不能被添加到当前图元块。因此，状态数据、标头和当前图元区段都被输出(例如输出到存储器)，然后数据存储器1008被重新配置以存储新的图元块(即，状态数据、标头和当前图元区段的内容被刷新)。然后，将状态数据设置为图元1002的状态数据1004，将图元1002添加到当前图元区段，并且修改标头以指定新图元块包括具有一个图元和三个顶点的一个图元区段。如1010处所示，这导致状态数据被设置为DEDE，并且当前图元区段包括图元1002的每个顶点(具有局部索引V0、V1、V2)的顶点条目和图元1002的标识相关顶点的图元条目；并且所述标头经设置以指示所述图元块包含具有一个图元和三个顶点的一个图元区段。应注意，在此实例中，全局顶点索引不匹配局部顶点索引。具体而言，全局顶点V0、V1、V3分别成为新的图元块的新图元区段中的局部顶点V0、V1和V2。

[0124] 在图6-10的示例中，基于状态数据和接收图元的顺序(例如，在图元块生成器600处)将图元分组成图元块。具体地，如果图元块具有与当前图元块相同的状态数据，则将所述图元添加到所述图元块，否则创建新的图元块，并将所接收的图元添加到所述图元块。然而，这仅是示例，并且在其他示例中，图元块生成器可以被配置为以另一种方式将图元分组成图元块。例如，在一些情况下，在将所接收的图元的状态数据与当前图元块的状态数据进行比较之前，图元块生成器可以被配置为使用其他标准来确定是否要将所接收的图元添加到当前图元块(例如，基于所接收的图元的空间位置与图元块中的图元的空间位置之间的距离)，并且仅在满足其他标准的情况下才比较状态数据。

[0125] 类似地，在图6-10的示例中，基于接收图元的顺序和图元区段大小约束(多个)将图元分组成图元区段。具体地，如果图元具有与所述图元区段中的图元相同的状态数据并且在所述图元区段中存在用于所述图元的足够空间(基于图元区段大小约束)，则将所述图元添加到所述图元区段。然而，这仅是示例，并且在其他示例中，在确定在图元区段中是否存在用于所述图元的足够的空间或空间之前(基于图元区段大小约束)，图元块生成器可以被配置为使用其他标准来确定所接收的图元是否要被添加到当前图元区段(例如，基于所接收的图元的空间位置与图元区段中的图元的空间位置之间的距离)，并且仅在满足该其他标准时，分析当前图元区段的大小约束。

[0126] 在图6-10的示例中，数据存储器被配置成存储单个图元块的数据。然而，在其他情况下，数据存储可以被配置为存储多个图元块的数据。例如，数据存储器可以被配置为存储多个图元块的标头、状态数据和当前图元区段。在这些情况下，在确定所接收的图元的状态数据是否与当前图元块的状态数据匹配之前，可以选择多个图元块中的一个作为当前图元块。例如，在一些情况下，渲染空间可被划分成多个区域(其可大于图块)，并且数据存储装置可被配置成存储用于多个区域中的每一个的图元块数据。在这些情况下，在确定所接收的图元的状态数据是否与当前图元块的状态数据匹配之前，图元块生成器可以被配置为确定所接收的图元落在哪个区域内，并且选择与这些区域相关联的图元块中的一个或多个作

为当前图元块。

[0127] 平铺引擎

[0128] 在经过变换的图元以上述结构和/或格式存储在图元块中的情况下，其中每个图元块可以包括一个或多个图元区段，所述平铺引擎可以被配置为生成每个图块的显示列表，所述显示列表具有与参照图1和图2所描述的格式不同的格式。具体地，图1中的平铺引擎被配置为生成每个图块的显示列表，如图2所示，所述显示列表针对包括至少部分地落在图块的边界内的至少一个图元的每个图元块包括标识所述图元块的信息(例如，存储器中的图元块的地址)和标识所述图元块的要用于渲染该平铺的图元的信息(例如，图元掩码)。相比之下，当图元以上述结构存储在图元块中时，平铺引擎可以被配置为生成每个图块的显示列表，针对包括至少部分地落在图块的边界内的至少一个图元的每个图元块，所述显示列表包括：标识所述图元块的信息(例如，所述图元块在存储器中的地址)；标识其包括至少部分地落在所述图块内的至少一个图元的每个图元区段的信息(例如，所述图元区段在存储器中的偏移)，及针对每个所标识的图元区段标识与渲染所述图块内相关的所述区段的图元的信息。这允许光栅化逻辑仅获取图元块的与渲染图块相关的那些图元区段，而不是获取整个图元块。

[0129] 现在参考图11，其示出了示例的平铺引擎1100，所述引擎被配置为生成图块的显示列表，所述显示列表标识用于渲染该图块的图元(例如，至少部分地落在该图块的边界内的图元)。平铺引擎1100包括平铺逻辑1102和显示列表生成器1104。

[0130] 平铺逻辑1102被配置为(i)接收多个图元块，其中每个图元块包括一个或多个图元区段，所述图元区段包括一个或多个图元(例如，与其相关的经过变换的几何结构数据)；(ii)针对每个所接收的图元区段确定所述图元区段的哪些图元至少部分地落在图块的边界内；以及(iii)输出所确定的结果。在一些情况下，输出可以是用于每个图元区段的图元掩码的形式。每个图元掩码可包括用于所述图元区段中的每个图元的位，所述位指示所述图元是否至少部分地落在所述图块的边界内。

[0131] 平铺逻辑1102可以使用任何合适的方法来确定图元是否至少部分地落在图块的边界内。例如，在一些状况下，平铺逻辑1102可以使用简单、准确性不高的方法(诸如简单的边界框平铺方法)来确定图元是否至少部分地落在图块内，以便将图元快速地分类到图块中。如所属领域的技术人员所知的，在边界框方法中，对包围基元的边界框进行标识(例如，包围基元顶点的与轴线对齐的最小边界框)。可以使用任何合适的方法来生成边界框。例如，平铺逻辑1102可以通过找到图元顶点的最小和最大的X和Y坐标并且根据这些坐标形成轴线对齐边界框来生成边界框。可以以任何粒度或分辨率生成边界框。例如，在一些状况下，边界框可以处于X和Y坐标分辨率(即，边界框可以由顶点的最大和最小的X和Y坐标限定)。在其他状况下，边界框可以处于图块分辨率(即，包围图元的最近的图块边缘)。一旦平铺逻辑1102已经标识出图元的边界框，如果边界框至少部分地与图块重叠或相交，平铺逻辑1102就可以确定图元至少部分地落在图块内。换句话讲，如果图元的边界框至少部分地落在图块的边界内，则可以确定所述图元至少部分地落在图块内。虽然可以使用边界框方法快速有效地确定图元是否至少部分地落在图块内，但这并不是“完美”的平铺，因为边界框通常大于图元，这可导致图元实际上未处于图块中时被确定为处于所述图块中。

[0132] 例如，图12示出了被分成四个图块1202、1204、1206和1208的示例渲染空间1200。

如果使用简单的轴线对齐边界框方法来确定图元1210至少部分地落在这些图块1202、1204、1206、1208中的哪些图块内，则生成围绕图元1210的边界框1212。由于边界框1212与所有图块1202、1204、1206、1208至少部分地重叠，因此即使图元1210实际上仅落在三个图块1204、1206、1208内或与它们重叠，也可以确定所述图元至少部分地落在四个图块1202、1204、1206、1208中的每一个图块内。然而，在图元实际上并未落在图块内时确定图元落在图块内不会导致出错，并且在光栅化阶段中会简单地丢弃所述图元。然而，在图元确实落在图块时确定图元并未落在图块内可导致光栅化阶段发生错误。因此，保守图块是有利的。换句话讲，与不包括实际上落在图块内的图元相比，更好的是指示图元落在图块内，即使图元实际上并未落在图块内。

[0133] 然而，在其他状况下，图块逻辑1102可以使用更复杂和/或更精确的方法(诸如完美平铺或近乎完美平铺方法)来确定图元是否落在图块内。可以由平铺逻辑1102使用的示例完美平铺方法在申请人已公开的申请号为2549789的英国专利申请中进行了描述，所述申请的全部内容以引用方式并入本文。

[0134] 显示列表生成器1104接收由平铺逻辑1102输出的平铺结果(例如图元掩码)、存储器中的每个图元块的地址和存储器中的每个图元区段的位置(例如偏移)，并且从其生成显示列表。具体地，显示列表生成器1104被配置为，针对包括至少部分地落在图块的边界内的至少一个图元的每个图元块，将图元块条目添加到显示列表，所述图元块条目标识所述图元块。在一些情况下，图元块条目可以包括图元块在存储器中的地址。显示列表生成器1104接着被配置为针对包括至少一个至少部分地落在图块的边界内的图元块的每个图元区段将标识所述图元区段在存储器中的位置且标识所述图元区段的与渲染所述图块相关的图元的信息包含在显示列表中。

[0135] 在一些情况下，显示列表生成器1104可以被配置为，针对包括至少部分地落在图块的边界内的至少一个图元的每个图元区段：(i)将标识所述图元区段在存储器中的位置的图元区段条目添加到所述图块的所述显示列表，以及(ii)将标识与渲染所述图块相关的所述图元区段的图元(即，至少部分地落在所述图块的边界内的图元)的相关图元条目添加到所述显示列表。在一些情况下，每个图元区段条目可以包括从标头的结束的偏移和标识存储器中的图元区段的开始的图元块的状态数据。

[0136] 例如，如图13所示，如果图元块包括编号为0至5的六个图元区段，并且第0、第3和第5图元区段包括至少部分地落在到图块的边界内的至少一个图元，则所述图块的显示列表1302可以包括标识所述图元块(例如，所述图元块在存储器中的地址)的图元块条目1304、标识所述图元区段在存储器中的位置(例如，偏移)的第0、第3和第5图元区段中的每一个的图元区段条目1306、1308、1310、以及标识所述图元区段的与所述图块相关的图元(例如，图元掩码)的第0、第3和第5图元区段中的每一个的相关图元条目1312、1314、1316。

[0137] 在一些情况下，图元块的第一图元区段的位置可以是已知的(例如，偏移可以是零)。在这些情况下，代替将用于第一图元区段的图元区段条目添加到图块的显示列表，可以存在两种类型的图元块条目，一种指定第一图元区段(即图元区段0)与图块相关，而另一种指定第一图元区段(即图元区段0)与图块不相关。在一些情况下，每个图元块条目可以包括标志或位，如果第一图元区段(即，图元区段0)与图块相关则设置所述标志或位，并且如果第一图元区段(即，图元区段0)与图块不相关则不设置所述标志或位，反之亦然。这消除

了在显示列表中包括第一图元区段的位置(例如偏移)的需要。

[0138] 例如,如图14所示,如果一个图元块包括编号为0至5的六个图元区段,并且第0、第3和第5图元区段包括至少一个至少部分落在图块边界内的图元,则所述图块的显示列表1402可以包括标识所述图元块(例如,所述图元块在存储器中的地址)并且指示第0图元区段相关的图元块条目1404、标识所述图元区段在存储器中的位置(例如,偏移)的第3和第5图元区段中的每一个的图元区段条目1406、1408、以及标识与所述图块相关的所述图元区段1412(例如,图元掩码)的第0、第3和第5图元区段中的每一个的相关图元条目1410、1414。因此,与图13的示例显示列表相比,图14的显示列表1402不包括第0图元区段的图元区段条目。

[0139] 图15中示出了另一个示例。在此实例中,图元块包括编号为0到4的五个图元区段,并且第3和第4图元区段包括至少部分地落在图块的边界内的至少一个图元。在该示例中,图块的显示列表1502可以包括标识图元块(例如,存储器中的图元块的地址)并且指示第0图元区段与当前区段不相关的图元块条目1504、标识图元区段在存储器中的位置(例如,和偏移)的用于第3和第4图元区段中的每一个的图元区段条目1506、1508、以及标识与区段相关的所述图元区段的图元(例如,图元掩码)的用于第3和第4图元区段中的每一个的相关图元条目1510、1512。

[0140] 现在参考图16,其示出了可由图11的平铺引擎1100实现的生成图块的显示列表的示例方法1600。方法1600开始于步骤1602,在此步骤接收图元块。所述图元块包括一个或多个图元区段,每个图元区段包括一个或多个图元。一旦已经接收到图元块,方法1600就进行到步骤1604。

[0141] 在步骤1604处,选择图元块的第一图元区段,并且方法1600进行到步骤1606。

[0142] 在步骤1606处,确定所选图元区段中的哪些图元至少部分地落在所述图块的边界内。可使用任何方法(例如上文所述的方法)来确定图元是否至少部分地落在图块的边界内。如果图元至少部分地落在图块的边界内,则所述图元被认为与渲染所述图块相关。一旦确定了所选图元区段中的哪些图元至少部分地落在图块的边界内,则方法1600进行到步骤1608。

[0143] 在步骤1608处,确定在所选图元区段中是否存在与渲染所述图块相关的至少一个图元(即,是否存在至少部分地落在所述图块的边界内的至少一个图元)。如果确定图元区段中没有图元与渲染图块有关,则方法1600进行到步骤1616。然而,如果确定存在与渲染所述拼图相关的图元区段的至少一个图元,则方法1600进行到步骤1610。

[0144] 在步骤1610处,确定这是否是当前图元块的第一图元区段,所述图元区段包括与渲染图块相关的图元。如果确定这是包括与渲染所述图块相关的图元的第一图元区段,则方法1600进行到步骤1612。然而,如果确定这不是包括与渲染图块相关的图元的第一图元区段,则方法1600直接进行到步骤1614。

[0145] 在步骤1612处,可以将标识所述图元块的信息添加到所述图块的显示列表。如上所述,将标识图元块的信息添加到显示列表可以包括将图元块条目添加到显示列表,所述图元块条目标识存储器中的图元块的位置(例如,图元块条目可以包括存储器中的图元块的地址)。一旦标识图元块的信息被添加到显示列表中,方法1600就进行到步骤1614。

[0146] 在步骤1614处,将标识所选图元区段的信息连同标识所选图元区段的与渲染图块

相关的图元的信息一起添加到显示列表。如上所述，将标识所选图元区段的信息添加到显示列表可包括将标识图元区段在存储器中的位置的图元区段条目添加到显示列表。图元区段在存储器中的位置可表示为偏移(例如，从标头的末尾的偏移和图元块的状态数据)。在一些情况下，在所选图元区段是图元块中的第一图元区段的情况下，将标识所选图元区段的信息添加到显示列表可以包括修改图元块条目以指示第一图元区段是相关的。如上所述，将标识与渲染图块相关的图元区段的图元的信息添加到所述图块的显示列表可包括将所述图元区段的相关图元条目添加到所述显示列表。相关图元条目可包括图元掩码，所述图元掩码包括用于图元区段中的每个图元的位，所述位指示所述图元是否至少部分地落在所述图块的边界内。一旦关于图元区段的信息已经添加到显示列表，方法1600就进行到步骤1616。

[0147] 在步骤1616处，确定在图元块中是否还有至少一个图元区段。如果确定在图元块中至少还有一个图元区段，则方法1600进行到步骤1618，在该步骤中选择下一个图元区段。然而，如果确定在图元块中不再有图元区段，则方法1600进行到步骤1620。

[0148] 在步骤1620处，确定是否还有至少一个图元块。如果确定至少还有一个图元块，方法1600返回到步骤1602，在此接收下一个图元块。然而，如果确定没有更多的图元块，则方法1600结束。

[0149] 图形处理系统

[0150] 现在参考图17，其示出了示例的基于图块的渲染图形处理系统1700，其包括图6的图元块生成器600和图11的平铺引擎1100。图17的图形处理系统1700与图1的图形处理系统100的相似之处在于，其包括几何结构处理逻辑1704和光栅化逻辑1706；几何结构处理逻辑1704包括对应于图1的变换逻辑108的变换逻辑1708；并且光栅化逻辑1706包括光栅化器1714、HSR逻辑1716和纹理化/着色逻辑1718(它们各自用作上述图1的对应部件)。然而，几何结构处理逻辑并不包括图元生成器，以将转换逻辑1708所生成的经过转换的图元储存于具有单一图元区段的图元中，而是包括图元生成器600，其用以将转换逻辑1708所生成的经过转换的图元储存于可包括一个以上图元区段的图元中。几何结构处理逻辑1704还包括平铺引擎1100，其被配置成生成显示列表，所述显示列表标识用于渲染图块的相关图元块，以及与渲染图块相关的其图元区段。

[0151] 图18示出了可以实现本文所述的图元块生成器、平铺引擎和/或图形处理系统的计算机系统。计算机系统包括CPU 1802、GPU 1804、存储器1806和其他装置1814，诸如显示器1816、扬声器1818和相机1820。在GPU 1804上实现处理块1810(其可对应于本文描述的图元块生成器、平铺引擎和/或图形处理系统)。在其他示例中，处理块1810可以在CPU 1802上实现。计算机系统的部件可经由通信总线1822彼此进行通信。

[0152] 图1、6、11和17的图元块生成器、平铺引擎和图形处理系统被示为包括多个功能块。这仅是示意性的，并不旨在限定此类实体的不同逻辑元件之间的严格划分。每个功能块可以任何合适的方式提供。应当理解，本文所述的由图元块生成器、平铺引擎或图形处理系统形成的中间值不需要在任何点由图元块生成器、平铺引擎或图形处理系统物理地生成，并且可以仅仅表示方便地描述由图元块生成器、平铺引擎或图形处理系统在其输入和输出之间执行的处理的逻辑值。

[0153] 本文所述的图元块生成器、平铺引擎和图形处理系统可以以集成电路上的硬件来

体现。本文所述的图元块生成器、平铺引擎和图形处理系统可以被配置成执行本文所述的任何方法。一般来讲，上文所述的功能、方法、技术或部件中的任一者可在软件、固件、硬件(例如，固定逻辑电路系统)或其任何组合中实现。本文可以使用术语“模块”、“功能”、“部件”、“元件”、“单元”、“块”和“逻辑”来概括地表示软件、固件、硬件或其任何组合。在软件实现方式的情况下，模块、功能、部件、元件、单元、块或逻辑表示程序代码，所述程序代码在处理器上执行时执行指定任务。本文中所描述的算法和方法可以由执行代码的一个或多个处理器执行，所述代码使处理器执行算法/方法。计算机可读存储介质的示例包括随机存取存储器(RAM)、只读存储器(ROM)、光盘、闪存存储器、硬盘存储器以及可以使用磁性、光学和其他技术来存储指令或其他数据并且可以由机器存取的其他存储器装置。

[0154] 如本文中所使用的术语计算机程序代码和计算机可读指令是指供处理器执行的任何种类的可执行代码，包含以机器语言、解释语言或脚本语言表达的代码。可执行代码包含二进制代码、机器代码、字节代码、定义集成电路的代码(例如硬件描述语言或网表)，以及用例如C、Java或OpenCL等编程语言代码表达的代码。可执行代码可以是例如任何种类的软件、固件、脚本、模块或库，当在虚拟机或其他软件环境中被适当地执行、处理、解释、编译、运行时，这些软件、固件、脚本、模块或库使支持可执行代码的计算机系统的处理器执行由所述代码指定的任务。

[0155] 处理器、计算机或计算机系统可以是任何种类的装置、机器或专用电路，或其集合或一部分，它具有处理能力使得可以执行指令。处理器可以是任何种类的通用或专用处理器，例如CPU、GPU、片上系统、状态机、媒体处理器、专用集成电路(ASIC)、可编程逻辑阵列、现场可编程门阵列(FPGA)等。计算机或计算机系统可以包括一个或多个处理器。

[0156] 本发明还意图包围限定如本文中所描述的硬件的配置的软件，例如硬件描述语言(HDL)软件，用于设计集成电路或用于配置可编程芯片以执行所要功能。也就是说，可以提供一种其上编码有集成电路限定数据集形式的计算机可读程序代码的计算机可读存储介质，当在集成电路制造系统中处理(即，运行)时，该计算机可读程序代码将该系统配置成制造被配置成执行本文所述任何方法的图元块生成器、平铺引擎或图形处理系统，或者制造包括本文所述任何装置的图元块生成器、平铺引擎或图形处理系统。集成电路定义数据集可以是例如集成电路描述。

[0157] 因此，可以提供一种在集成电路制造系统处制造如本文所述的图元块生成器、平铺引擎或图形处理系统的方法。此外，可以提供一种集成电路限定数据集，当其在集成电路制造系统中被处理时致使图元块生成器、平铺引擎或图形处理系统的制造方法被执行。

[0158] 集成电路定义数据集可以是计算机代码的形式，例如作为网表，用于配置可编程芯片的代码，作为定义适合于在集成电路中以任何级别制造的硬件描述语言，包括作为寄存器传输级(RTL)代码，作为高级电路表示法(诸如Verilog或VHDL)，以及作为低级电路表示法(诸如OASIS(RTM)和GDSII)。在逻辑上定义适合于在集成电路中制造的硬件的更高级表示法(诸如RTL)可以在计算机系统上处理，所述计算机系统被配置用于在软件环境的上下文中生成集成电路的制造定义，所述软件环境包括电路元件的定义和用于组合这些元件以生成由该表示法定义的集成电路的制造定义的规则。如通常软件在计算机系统处执行以便定义机器的情况一样，可能需要一个或多个中间用户步骤(例如，提供命令、变量等)，以便将计算机系统配置成生成集成电路的制造定义，以执行定义集成电路以便生成所述集成

电路的制造定义的代码。

[0159] 现在将参考图19描述在集成电路制造系统处处理集成电路限定数据集以便将系统配置为制造图元块生成器、平铺引擎或图形处理系统的示例。

[0160] 图19示出了集成电路(IC)制造系统1902的示例,其被配置为制造如本文的任何示例中所描述的图元块生成器、平铺引擎或图形处理系统。特别地,IC制造系统1902包括布局处理系统1904和集成电路生成系统1906。IC制造系统1902被配置为接收IC定义数据集(例如,定义如本文的任何示例中所描述的图元块生成器、平铺引擎或图形处理系统)、处理IC定义数据集、以及根据IC定义数据集(例如,其实现如本文的任何示例中所描述的图元块生成器、平铺引擎或图形处理系统)生成IC。IC限定数据集的处理将IC制造系统1902配置为制造体现如本文任何示例中所述的图元块生成器、平铺引擎或图形处理系统的集成电路。

[0161] 布局处理系统1904被配置为接收和处理IC定义数据集以确定电路布局。根据IC定义数据集确定电路布局的方法在本领域中是已知的,并且例如可以涉及合成RTL代码以确定要生成的电路的门级表示,例如就逻辑部件(例如NAND、NOR、AND、OR、MUX和FLIP-FLOP部件)而言。通过确定逻辑部件的位置信息,可以根据电路的门级表示来确定电路布局。这可以自动完成或者在用户参与下完成,以便优化电路布局。当布局处理系统1904确定了电路布局时,它可以将电路布局定义输出到IC生成系统1906。电路布局定义可以是例如电路布局描述。

[0162] 如本领域已知的,IC生成系统1906根据电路布局定义来生成IC。例如,IC生成系统1906可以实现生成IC的半导体器件制造工艺,其可以涉及光刻和化学处理步骤的多步骤序列,在此期间,在由半导体材料制成的晶片上逐渐形成电子电路。电路布局定义可呈掩码的形式,其可以在光刻工艺中用于根据电路定义来生成IC。替代性地,提供给IC生成系统1906的电路布局定义可以是计算机可读代码的形式,IC生成系统1906可以使用该计算机可读代码来形成用于生成IC的合适掩码。

[0163] 由IC制造系统1902执行的不同过程可以全部在一个位置例如由一方来实现。替代性地,IC制造系统1902可以是分布式系统,使得一些过程可以在不同位置执行,并且可以由不同的方来执行。例如,以下阶段中的一些可以在不同位置和/或由不同方来执行:(i)合成表示IC定义数据集的RTL代码,以形成要生成的电路的门级表示;(ii)基于门级表示来生成电路布局;(iii)根据电路布局来形成掩码;以及(iv)使用掩码来制造集成电路。

[0164] 在其他示例中,在集成电路制造系统中对集成电路限定数据集的处理可以将该系统配置成在不处理IC限定数据集以确定电路布局的状况下制造图元块生成器、平铺引擎或图形处理系统。例如,集成电路定义数据集可以定义例如FPGA的可重新配置的处理器的配置,并且对所述数据集进行的处理可以将IC制造系统配置成(例如,通过将配置数据加载到FPGA)生成具有所述定义的配置的可重新配置的处理器。

[0165] 在一些实施方案中,当在集成电路制造系统中处理时,集成电路制造定义数据集可以使集成电路制造系统生成如本文中描述的装置。例如,通过集成电路制造定义数据集,以上面参考图19描述的方式对集成电路制造系统的配置,可以制造出如本文所述的装置。

[0166] 在一些示例中,集成电路定义数据集可包括在数据集处定义的硬件上运行的软件,或者与在数据集处定义的硬件组合运行的软件。在图19所示的示例中,IC生成系统还可以由集成电路定义数据集进一步配置,以在制造集成电路时根据在集成电路定义数据集中

定义的程序代码将固件加载到所述集成电路上，或者以其他方式向集成电路提供与集成电路一起使用的程序代码。

[0167] 与已知的实现方式相比，在本申请中阐述的概念在装置、设备、模块和/或系统中(以及在本文中实现的方法中)的实现方式可以引起性能改进。性能改进可以包含计算性能提高、等待时间缩短、处理量增大和/或功耗降低中的一个或多个。在制造此类装置、设备、模块和系统(例如在集成电路中)期间，可以在性能改进与物理实施方案之间进行权衡，从而改进制造方法。例如，可以在性能改进与布局面积之间进行权衡，从而匹配已知实现方式的性能，但使用更少的硅。例如，这可以通过以串行方式重复使用功能块或在装置、设备、模块和/或系统的元件之间共享功能块来完成。相反，在本申请中阐述的引起装置、设备、模块和系统的物理实现方式的改进(诸如硅面积减小)的概念可以针对性能提高进行权衡。例如，这可以通过在预定义面积预算内制造模块的多个实例来完成。

[0168] 申请人据此独立地公开了本文中所描述的每个单独的特征以及两个或更多个此类特征的任意组合，到达的程度使得此类特征或组合能够根据本领域的技术人员的普通常识基于本说明书整体来实行，而不管此类特征或特征的组合是否解决本文中所公开的任何问题。鉴于前文描述，本领域的技术人员将清楚，可以在本发明的范围内进行各种修改。

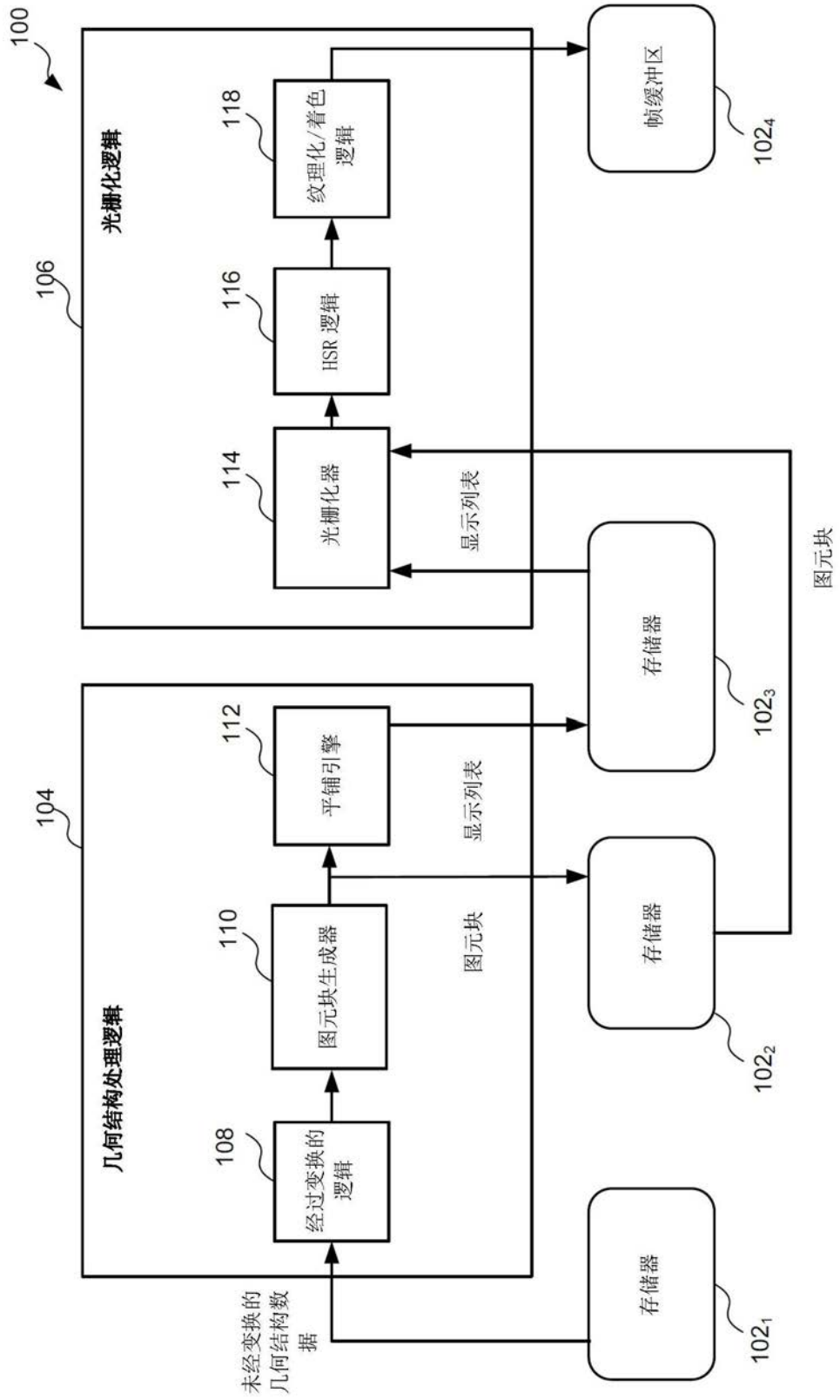

100
光栅化逻辑
106
几何结构处理逻辑
104
未经变换的几何结构数据
经过变换的逻辑
108
图元块生成器
110
平铺引擎
112
光栅化器
114
HSR 逻辑
116
纹理化/着色逻辑
118
显示列表
图元块
显示列表
图元块
帧缓冲区
1024
存储器
1023
存储器
1022
存储器
1021

图1

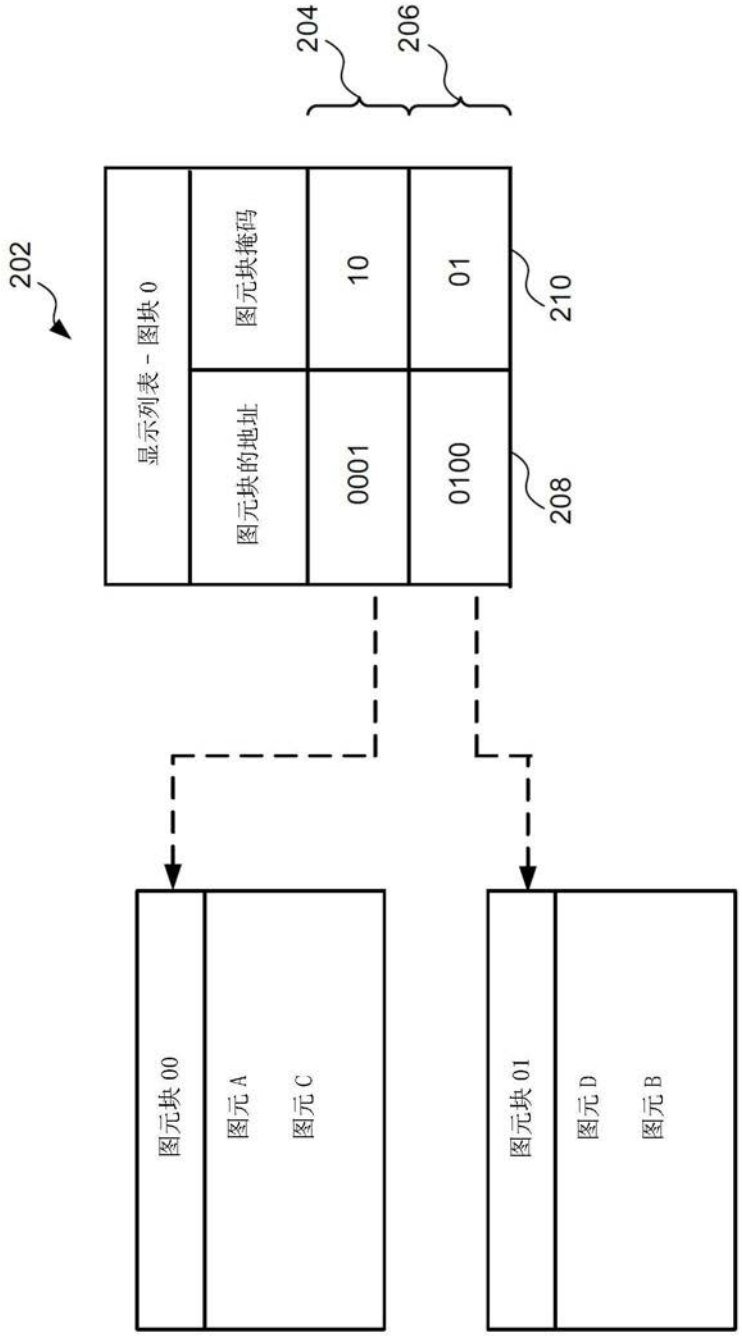

204
206
202
显示列表 - 图块 0
图元块掩码
10
01
210
图元块的地址
0001
0100
208
图元块 00
图元 A
图元 C
图元块 01
图元 D
图元 B

图2

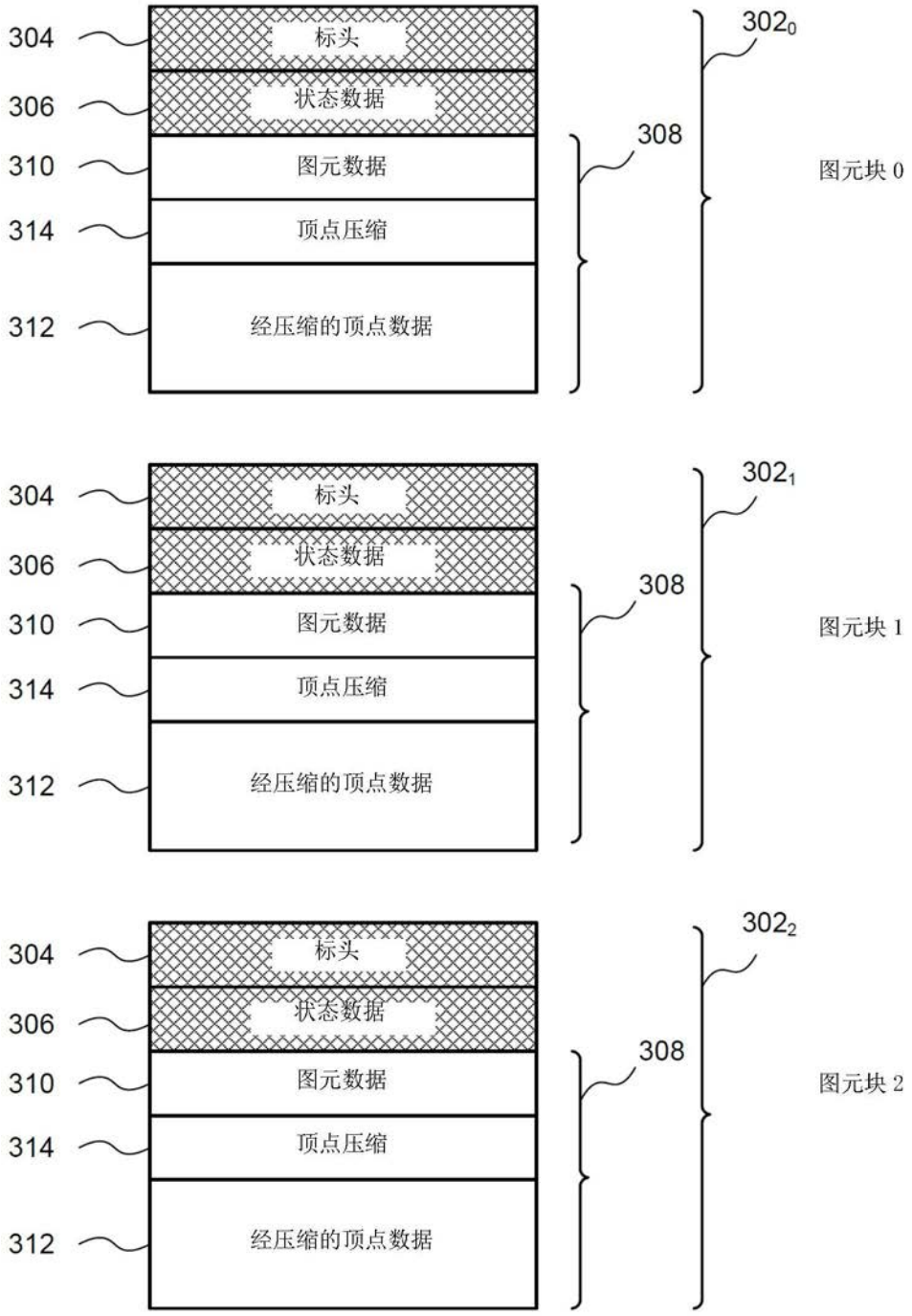

304
标头
306
状态数据
310
图元数据
314
顶点压缩
312
经压缩的顶点数据
308
302_0
图元块 0
304
标头
306
状态数据
310
图元数据
314
顶点压缩
312
经压缩的顶点数据
308
302_1
图元块 1
304
标头
306
状态数据
310
图元数据
314
顶点压缩
312
经压缩的顶点数据
308
302_2
图元块 2

图3

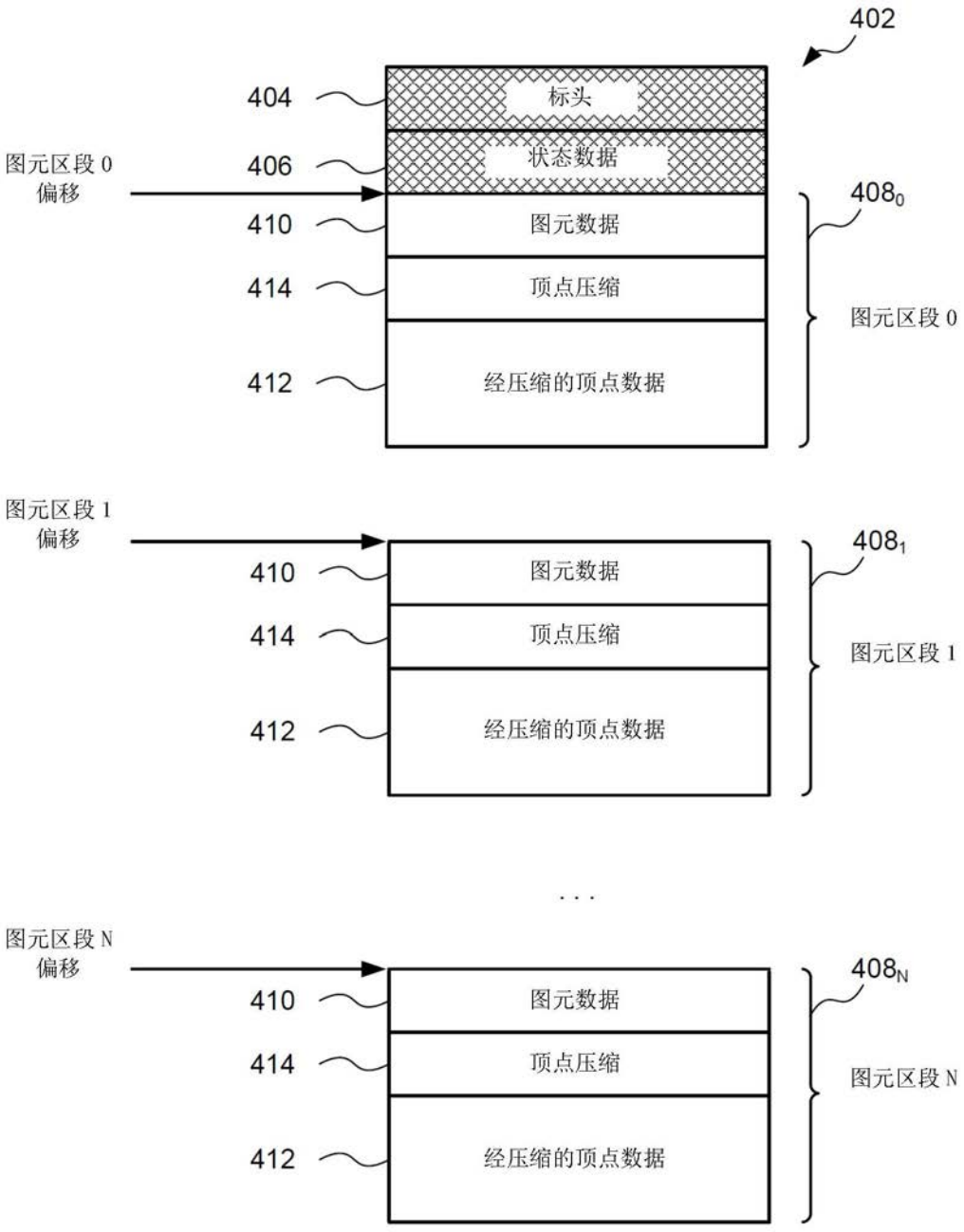

402
404
标头
406
状态数据
图元区段 0
偏移
410
图元数据
414
顶点压缩
412
经压缩的顶点数据
408_0
图元区段 0
图元区段 1
偏移
410
图元数据
414
顶点压缩
412
经压缩的顶点数据
408_1
图元区段 1
. . .
图元区段 N
偏移
410
图元数据
414
顶点压缩
412
经压缩的顶点数据
408_N
图元区段 N

图4

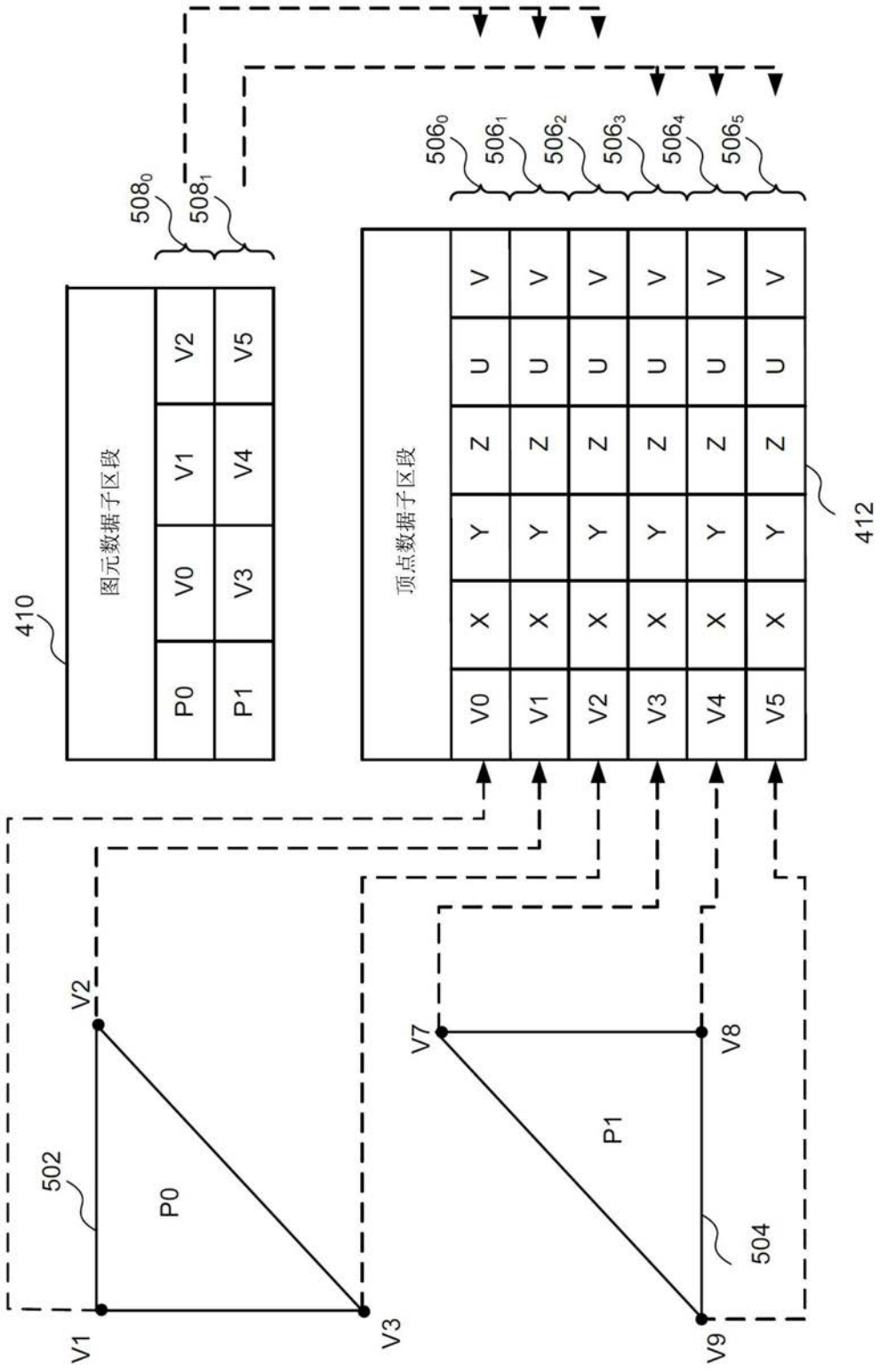

508₀
508₁
506₀
506₁
506₂
506₃
506₄
506₅
图元数据子区段
P0
P1
V0
V1
V2
V3
V4
V5
410
顶点数据子区段
X
Y
Z
U
V
412
502
504
V7
V8
V9

图5

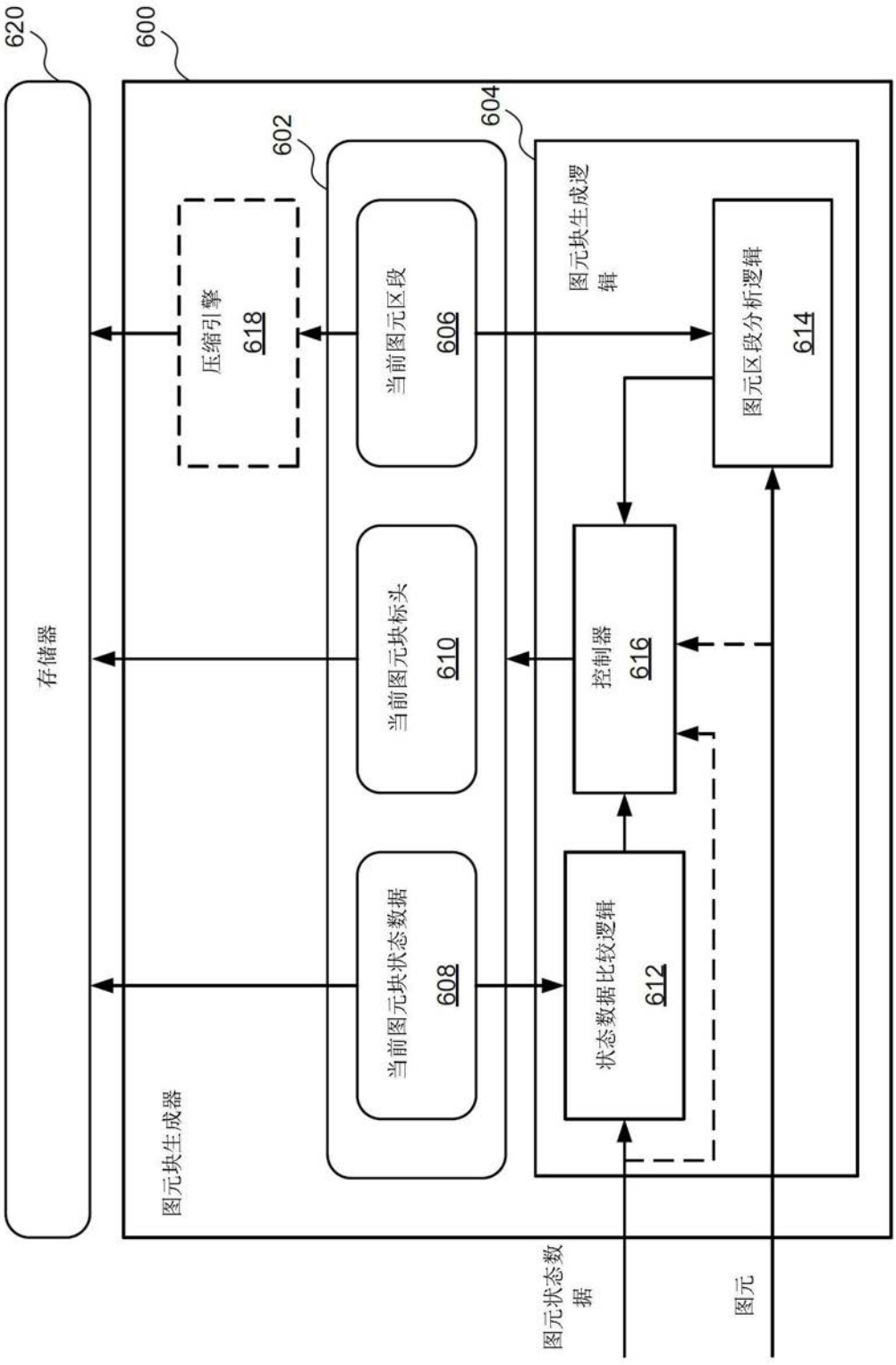

620
600
602
604
存储器
图元块生成器
压缩引擎
618
当前图元区段
606
当前图元块标头
610
当前图元块状态数据
608
图元块生成逻辑
图元区段分析逻辑
614
控制器
616
状态数据比较逻辑
612
图元状态数据
图元

图6

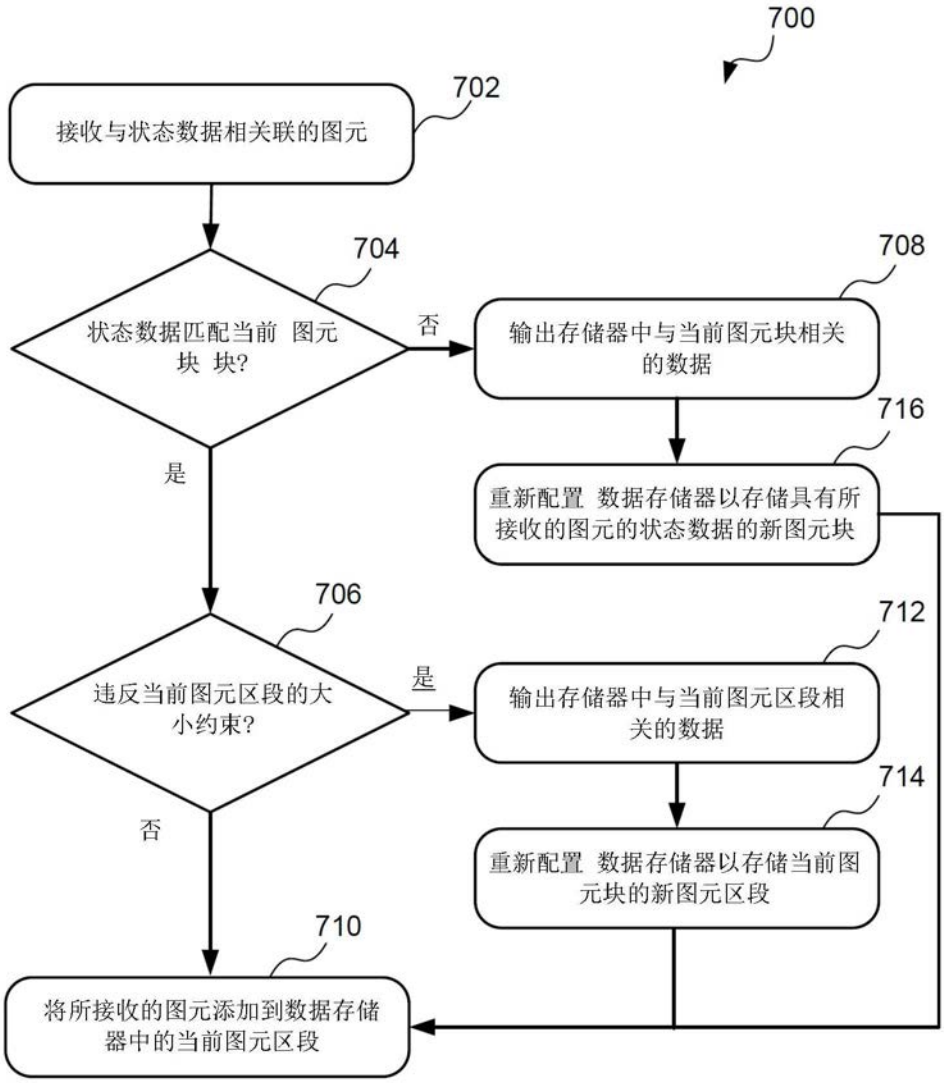

700
702
接收与状态数据相关联的图元
704
状态数据匹配当前 图元块 块?
否
是
708
输出存储器中与当前图元块相关的数据
716
重新配置 数据存储器以存储具有所接收的图元的状态数据的新图元块
706
违反当前图元区段的大小约束?
是
否
712
输出存储器中与当前图元区段相关的数据
714
重新配置 数据存储器以存储当前图元块的新图元区段
710
将所接收的图元添加到数据存储器中的当前图元区段

图7

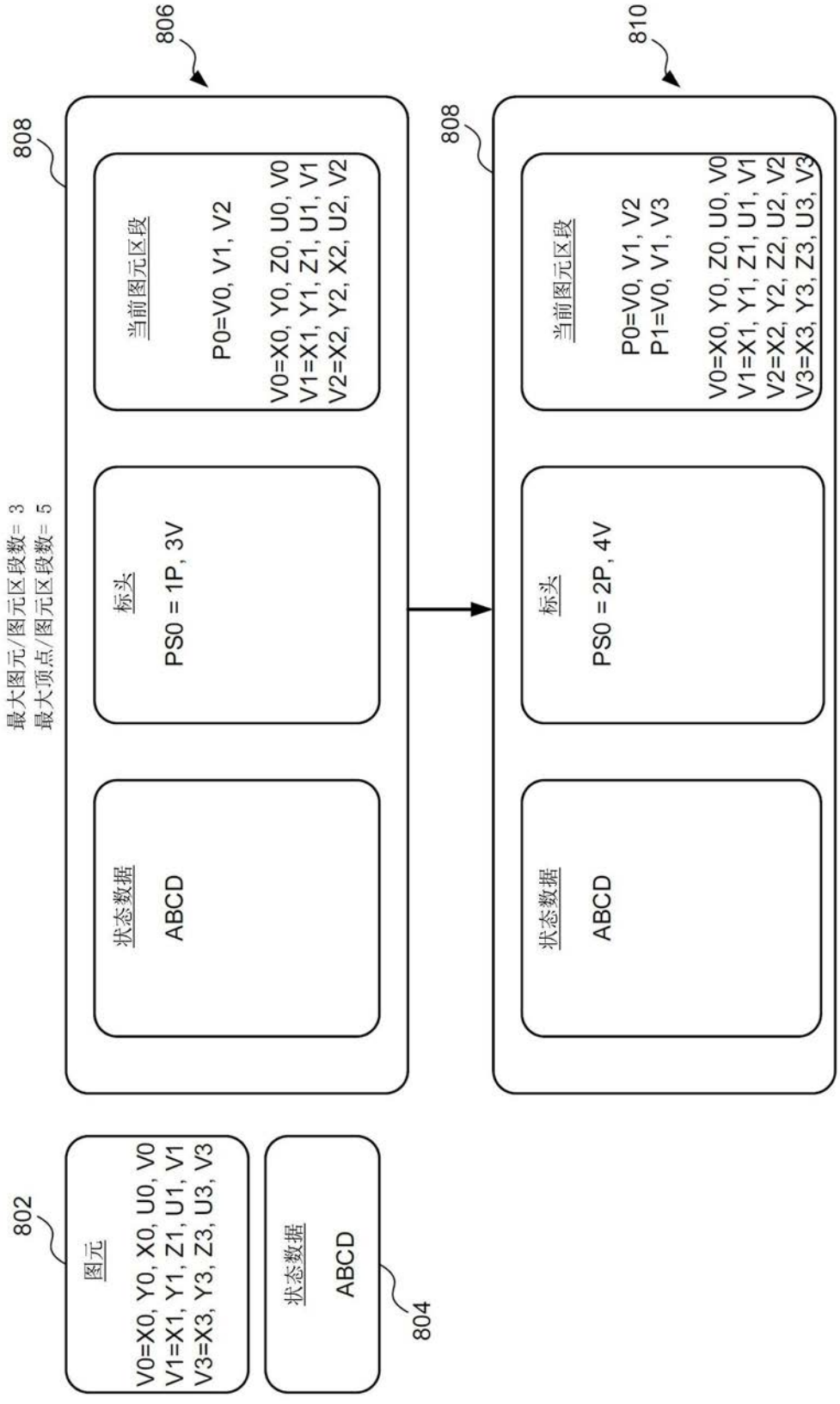

806
808
最大图元/图元区段数= 3
最大顶点/图元区段数= 5
状态数据
ABCD
标头
PS0 = 1P, 3V
当前图元区段
P0=V0, V1, V2
V0=X0, Y0, Z0, U0, V0
V1=X1, Y1, Z1, U1, V1
V2=X2, Y2, X2, U2, V2
810
808
状态数据
ABCD
标头
PS0 = 2P, 4V
当前图元区段
P0=V0, V1, V2
P1=V0, V1, V3
V0=X0, Y0, Z0, U0, V0
V1=X1, Y1, Z1, U1, V1
V2=X2, Y2, Z2, U2, V2
V3=X3, Y3, Z3, U3, V3
802
图元
V0=X0, Y0, X0, U0, V0
V1=X1, Y1, Z1, U1, V1
V3=X3, Y3, Z3, U3, V3
状态数据
ABCD
804

图8

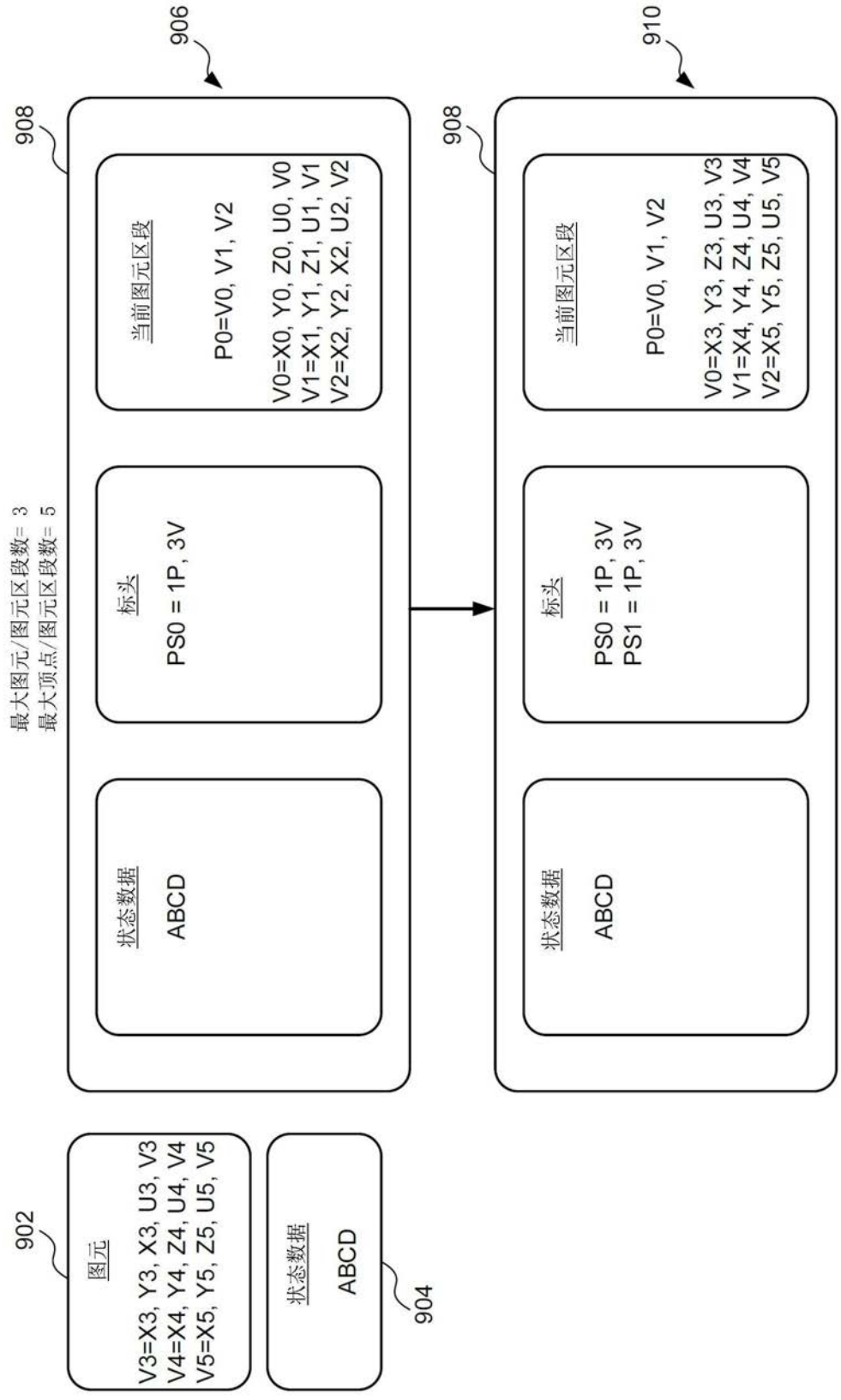

902
图元
V3=X3, Y3, X3, U3, V3
V4=X4, Y4, Z4, U4, V4
V5=X5, Y5, Z5, U5, V5
904
状态数据
ABCD
906
908
最大图元/图元区段数= 3
最大顶点/图元区段数= 5
状态数据
ABCD
标头
PS0 = 1P, 3V
当前图元区段
P0=V0, V1, V2
V0=X0, Y0, Z0, U0, V0
V1=X1, Y1, Z1, U1, V1
V2=X2, Y2, X2, U2, V2
910
908
状态数据
ABCD
标头
PS0 = 1P, 3V
PS1 = 1P, 3V
当前图元区段
P0=V0, V1, V2
V0=X3, Y3, Z3, U3, V3
V1=X4, Y4, Z4, U4, V4
V2=X5, Y5, Z5, U5, V5

图9

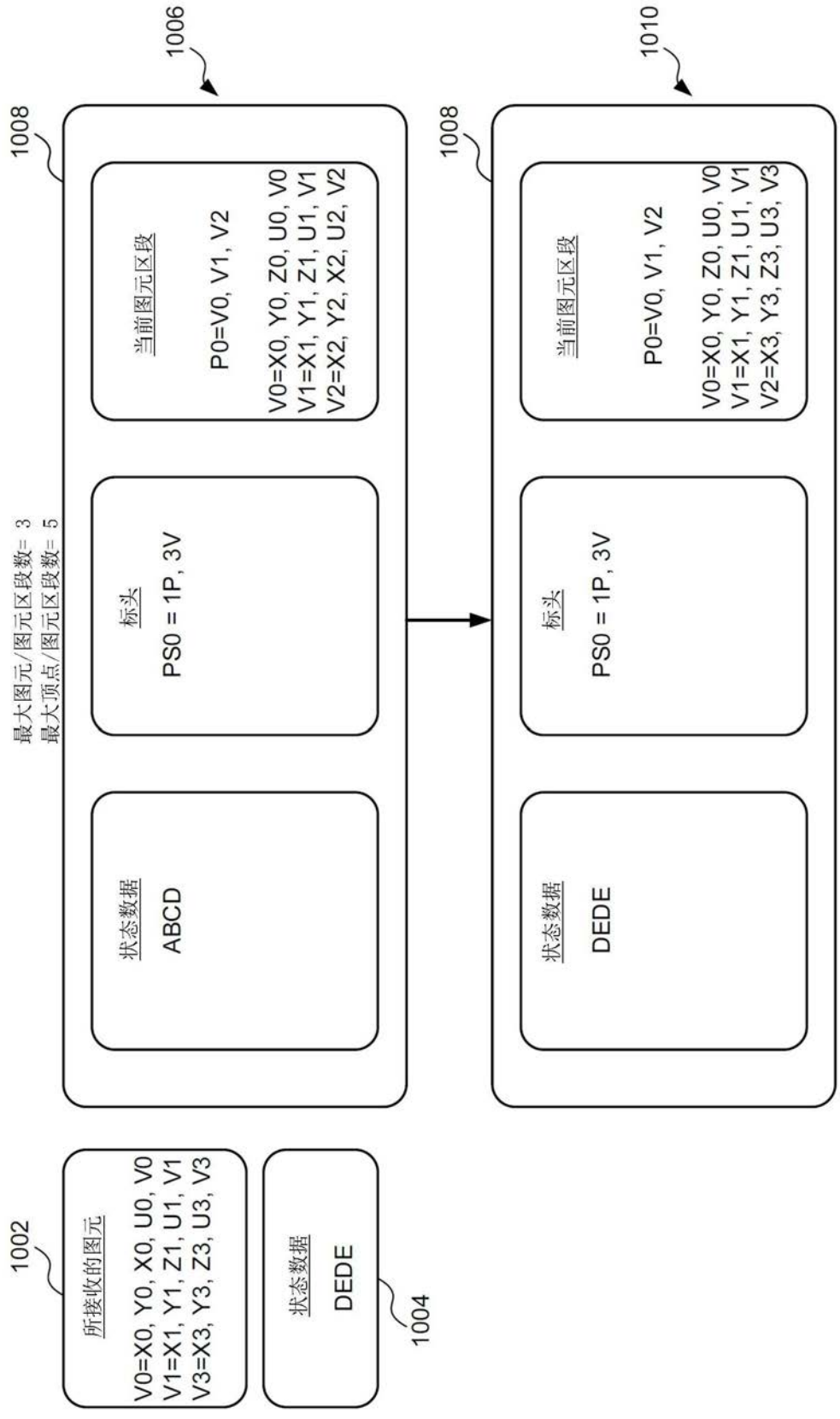

1002
所接收的图元
V0=X0, Y0, X0, U0, V0
V1=X1, Y1, Z1, U1, V1
V3=X3, Y3, Z3, U3, V3
状态数据
DEDE
1004
1006
1008
最大图元/图元区段数= 3
最大顶点/图元区段数= 5
状态数据
ABCD
标头
PS0 = 1P, 3V
当前图元区段
P0=V0, V1, V2
V0=X0, Y0, Z0, U0, V0
V1=X1, Y1, Z1, U1, V1
V2=X2, Y2, X2, U2, V2
1010
1008
状态数据
DEDE
标头
PS0 = 1P, 3V
当前图元区段
P0=V0, V1, V2
V0=X0, Y0, Z0, U0, V0
V1=X1, Y1, Z1, U1, V1
V2=X3, Y3, Z3, U3, V3

图10

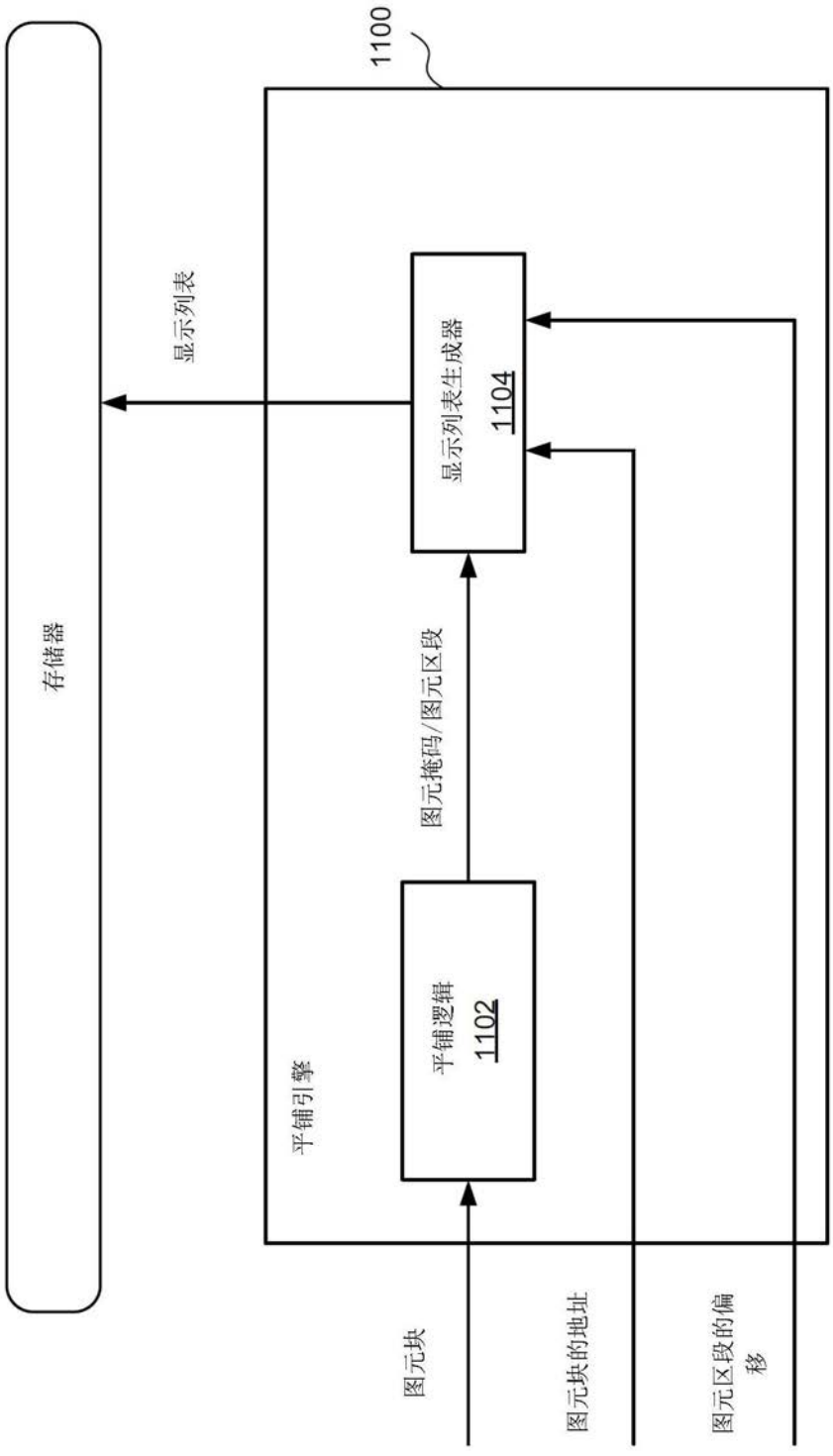

1100
存储器
显示列表
显示列表生成器
1104
图元掩码/图元区段
平铺引擎
平铺逻辑
1102
图元块
图元块的地址
图元区段的偏移

图11

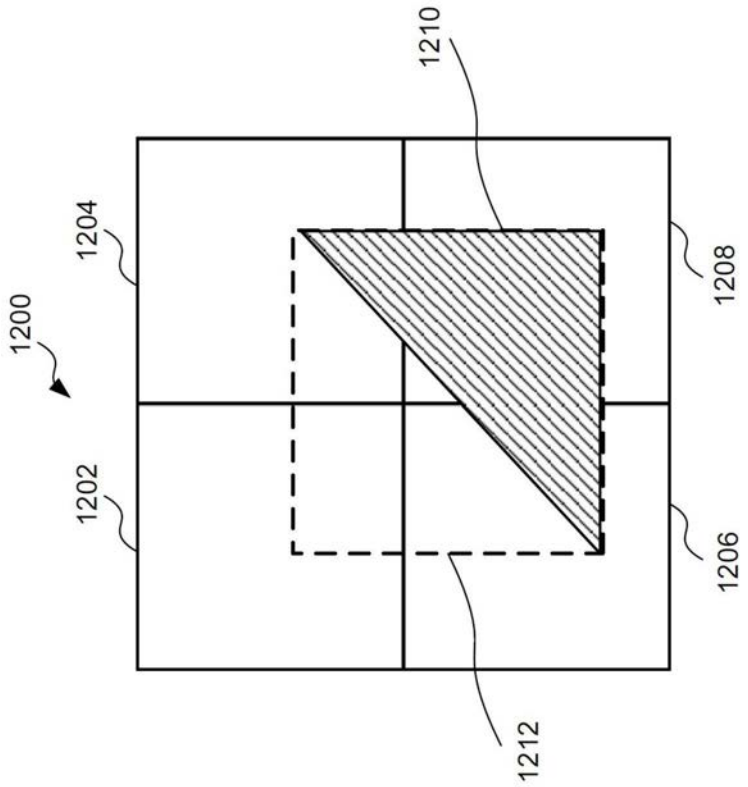

1210
1204
1208
1200
1202
1206
1212

图12

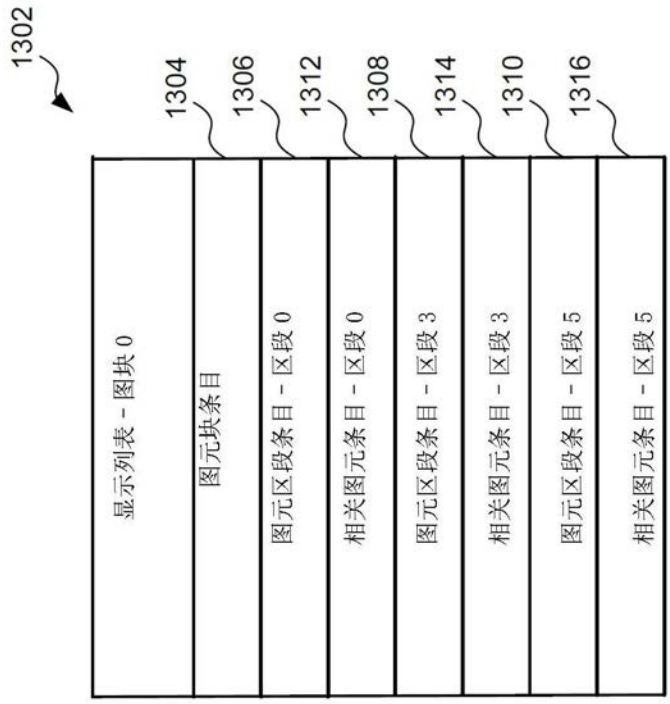

1302
1304
1306
1312
1308
1314
1310
1316
显示列表 - 图块 0
图元块条目
图元区段条目 - 区段 0
相关图元条目 - 区段 0
图元区段条目 - 区段 3
相关图元条目 - 区段 3
图元区段条目 - 区段 5
相关图元条目 - 区段 5

图13

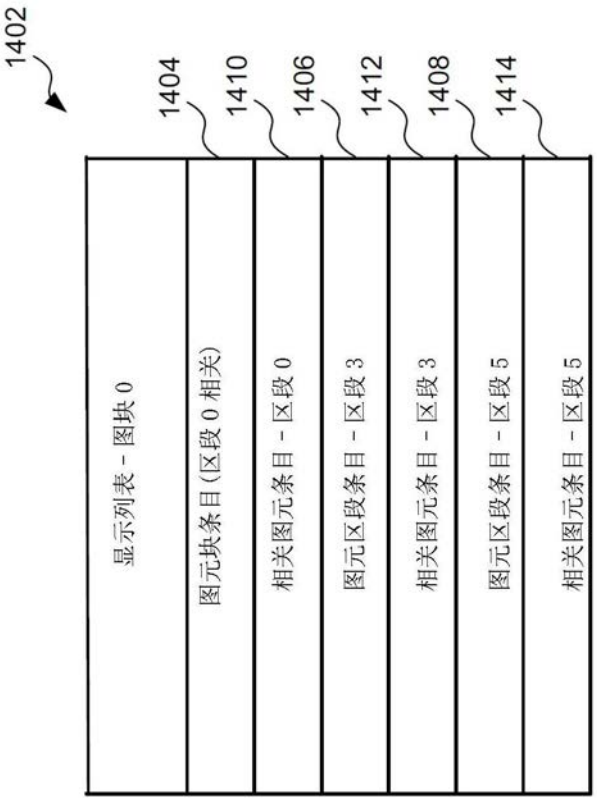

1402
1404
1410
1406
1412
1408
1414
显示列表 - 图块 0
图元块条目(区段 0 相关)
相关图元条目 - 区段 0
图元区段条目 - 区段 3
相关图元条目 - 区段 3
图元区段条目 - 区段 5
相关图元条目 - 区段 5

图14

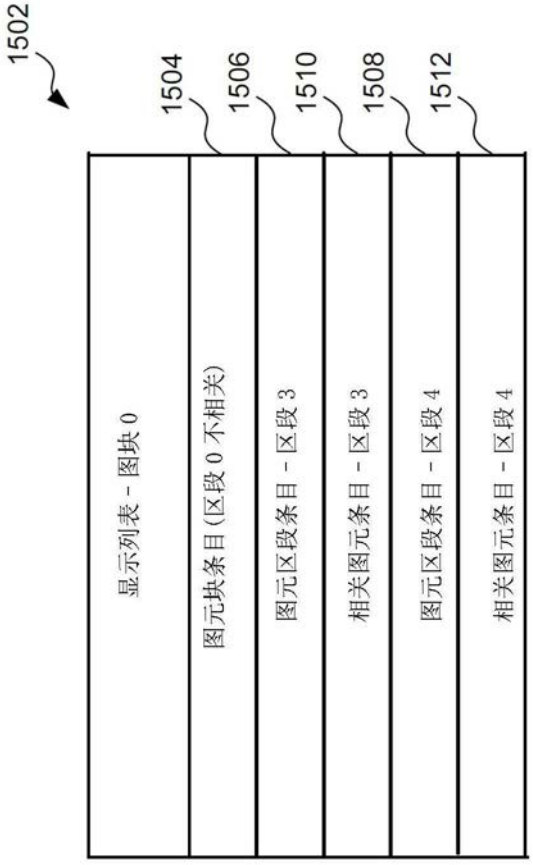

1502
1504
1506
1510
1508
1512
显示列表 - 图块 0
图元块条目(区段 0 不相关)
图元区段条目 - 区段 3
相关图元条目 - 区段 3
图元区段条目 - 区段 4
相关图元条目 - 区段 4

图15

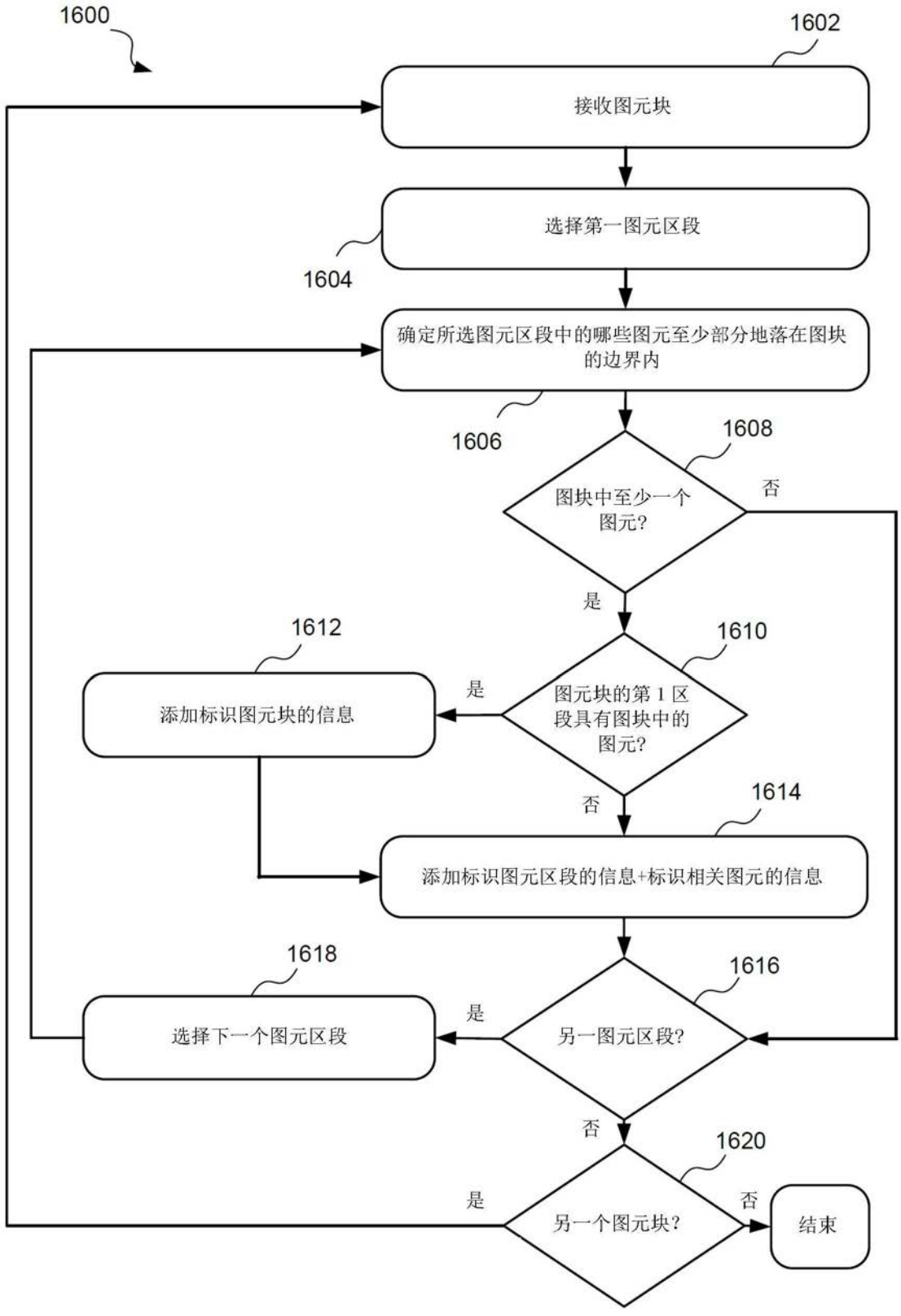

1600
1602
接收图元块
选择第一图元区段
1604
确定所选图元区段中的哪些图元至少部分地落在图块的边界内
1606
1608
图块中至少一个图元?
否
是
1612
1610
添加标识图元块的信息
是
图元块的第1区段具有图块中的图元?
否
1614
添加标识图元区段的信息+标识相关图元的信息
1618
1616
选择下一个图元区段
是
另一图元区段?
否
1620
是
另一个图元块?
否
结束

图16

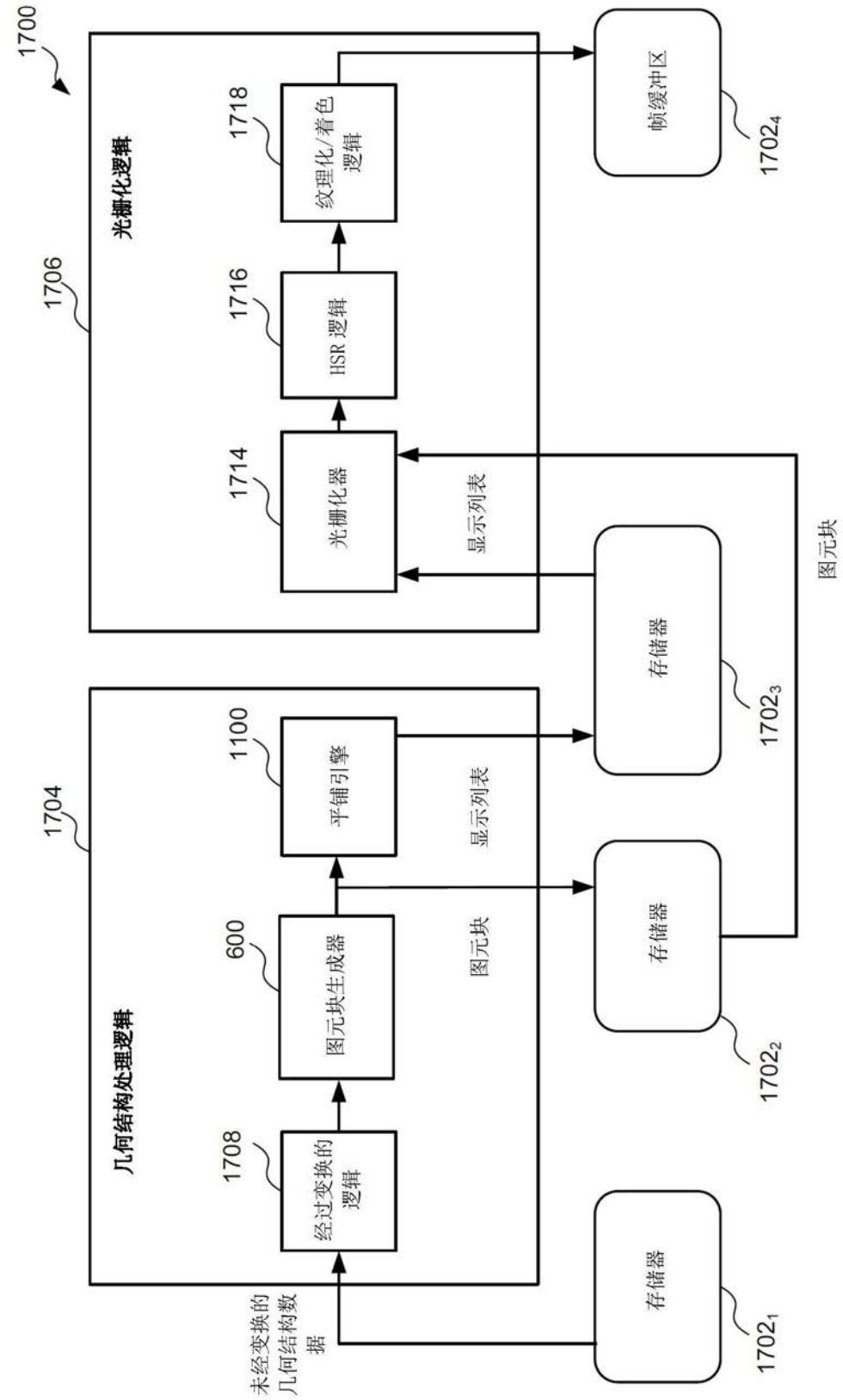

1700
光栅化逻辑
1706
1718
纹理化/着色逻辑
1716
HSR 逻辑
1714
光栅化器
显示列表
帧缓冲区
1702_4
图元块
存储器
1702_3
几何结构处理逻辑
1704
1100
平铺引擎
显示列表
600
图元块生成器
图元块
存储器
1702_2
1708
经过变换的逻辑
存储器
1702_1
未经变换的几何结构数据

图17

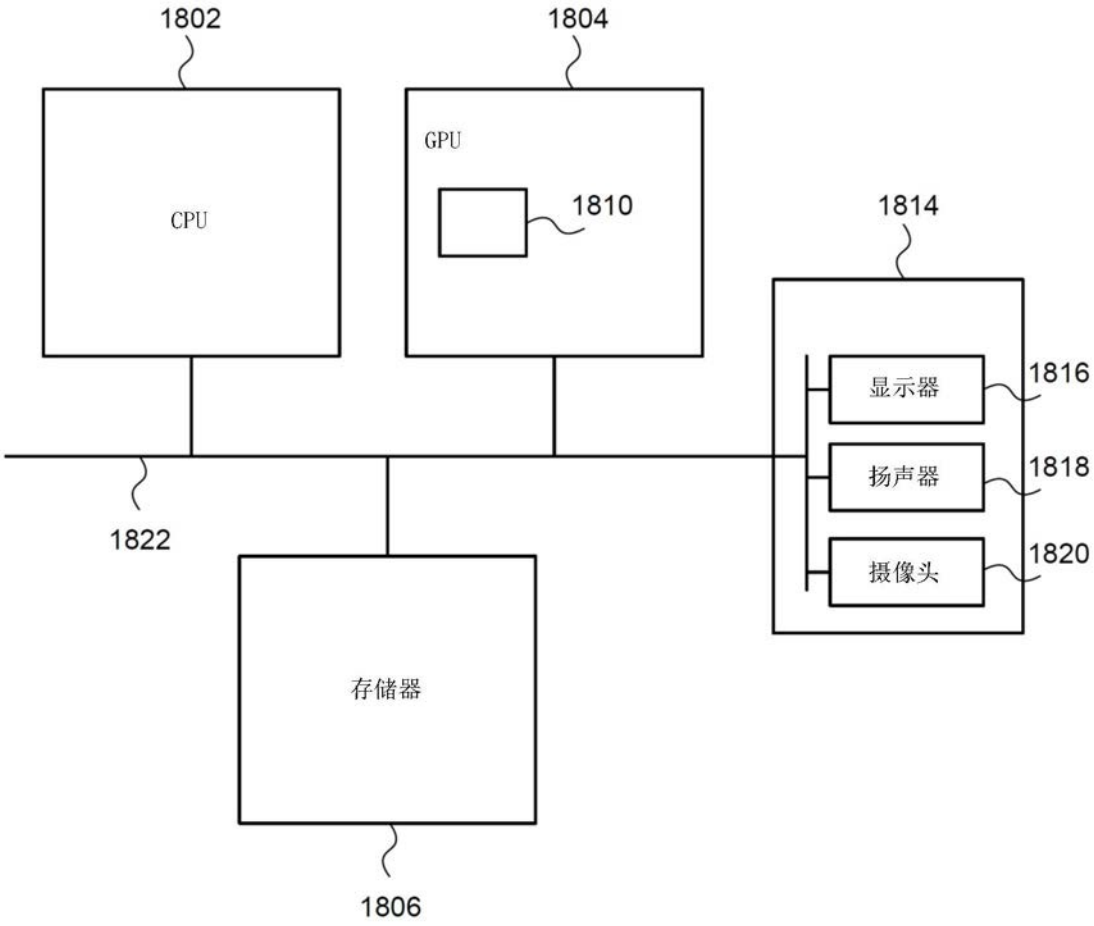

1802
1804
CPU
GPU
1810
1814
显示器
1816
扬声器
1818
摄像头
1820
1822
存储器
1806

图18

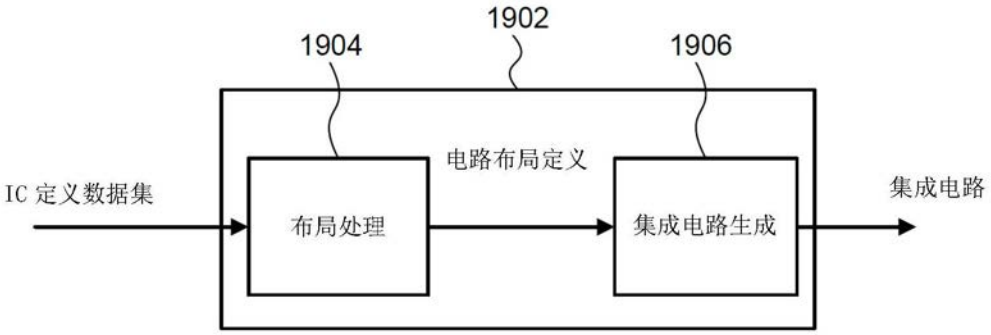

1902
1904
1906
电路布局定义
IC 定义数据集
布局处理
集成电路生成
集成电路

图19