[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
G06T 1/20

[12] 发明专利说明书

[21] ZL 专利号 97190237.2

[43] 授权公告日 2003 年 1 月 22 日

[11] 授权公告号 CN 1099655C

[22] 申请日 1997.2.6 [21] 申请号 97190237.2

[30] 优先权

[32] 1996. 2. 6 [33] JP [31] 20332/1996

[86] 国际申请 PCT/JP97/00297 1997.2.6

[87] 国际公布 WO97/29457 日 1997.8.14

[85] 进入国家阶段日期 1997.11.21

[71] 专利权人 索尼计算机娱乐公司

地址 日本东京

[72] 发明人 广井聪幸 冈正昭

[56] 参考文献

JP 平 6020063A 1994.01.28 G06F15/72

JP 平 6251166A 1994.09.09 G06F15/72

JP 平 7230555A 1995.08.29 G06T15/00、G06T5/20

JP 平 7296173A 1995.11.10 G06T11/00

审查员 韩岳

[74] 专利代理机构 北京市柳沈律师事务所

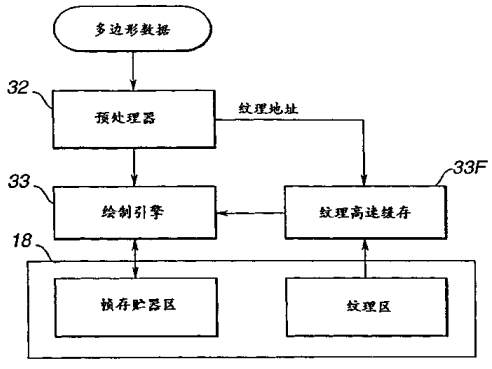
代理人 马 莹

权利要求书 1 页 说明书 19 页 附图 14 页

[54] 发明名称 图像绘制装置及图像绘制方法

[57] 摘要

一种用于图形计算机、特技设备或视频游戏机中的图像绘制装置和方法。 根据绘图命令，通过预处理器 32 的预处理产生图像绘制所需的数据，根据产生的数据通过纹理映射产生象素数据，绘制引擎 33 所需的纹理数据在预处理级从帧缓冲器 18 的纹理区传送到纹理高速缓存 33F，预处理器 32 和绘制引擎 33 以流水线方式工作。 这使纹理映射或 MIP 映射时不必停止图像绘制装置，降低了访问纹理存贮器的次数和时间，提高了整体图像绘制速度。



ISSN1008-4274

1.一种图像绘制装置，包括：

预处理部件，用于根据绘制由单元图形的组合定义的图像模型的绘制命令，在以单元图形为基础的绘制处理之前产生绘制处理所需的数据；

图像绘制部件，用于响应在图像存贮器中绘制图像的所述绘制命令，通过纹理映射处理以单位图形为基础产生像素数据；

存贮部件，用于临时存贮图像绘制装置进行纹理映射所需的纹理数据；

供给部件，用于响应由上述预处理装置输出的数据，把所述图像绘制装置进行纹理映射所需的纹理数据提供给所述存贮装置。

2. 根据权利要求1所述的图像绘制装置，其中所述图像绘制部件具有执行MIP映射的功能，所述图像绘制部件响应所述预处理部件输出的数据，选择具有所需分辨率的纹理数据，以提供所选择的纹理数据给所述存贮部件。

3. 一种图像绘制方法，包括下列步骤：

根据绘制由单元图形的组合定义的图像模型的绘图命令，以单元图形为基础，产生有关图像绘制处理的数据；

根据所述数据，把所述图像绘制处理中纹理映射所需的纹理数据提供给存贮部件；

通过纹理处理产生单元图形的象素数据以便在图像存贮器中绘图，并且在图像存贮器中绘制图像。

4. 根据权利要求3所述的图像绘制方法，其中把所述纹理数据供给所述存贮装置的所述步骤包括下述步骤：即响应有关所述图像绘制处理的数据，在所述图像绘制处理中提供MIP映射中所需分辨率的纹理数据。

图像绘制装置及图像绘制方法

技术领域

本发明涉及一种图像绘制装置和图像绘制方法，被用于图形计算机、特技设备或视频游戏机等利用计算机的视频设备。

背景技术

在家用 TV 游戏机、个人计算机或图形计算机中的图像生成装置（用于产生输出到电视接收机、监视器接收机或阴极射线管(CRT)显示装置上以便显示的图像数据）中，为了能高速处理，在中央处理单元(CPU)和帧缓冲器之间提供一专用绘制装置。

也就是说，在上述的图像生成装置中，在生成图形时 CPU 并不直接访问帧缓冲器，而是执行几何处理，如坐标变换、剪切或光源计算，将三维模型定义为基本单元图形如三角形或四边形的组合，以形成绘制三维图形的绘制命令，并把产生的绘制命令送到绘制装置．例如，为了形成一个三维物体，该物体被分成多个多边形，每个多边形的绘制命令从 CPU 送到绘制装置．绘制装置解释从 CPU 送来的绘制命令，执行写象素数据到帧缓冲器的成像处理，根据指定顶点的彩色数据和深度的 Z 值，考虑形成多边形的所有象素的 Z 值和彩色，以便在帧缓冲器中绘图． Z 值表示指定从视点沿深度方向的距离的信息．

例如，若一个三维物体被显示在上述图像生成装置中，该物体被分解成多个多边形，与每个多边形有关的绘制命令从 CPU 发送到绘制装置上．为了更逼真地显示物体，使用了称为纹理映射或 MIP 映射的技术．还有大家熟知的通过彩色查找表(CLUT)转换图像彩色数据的技术，在查找表中存有为改变显示彩色的彩色转换数据．

纹理映射是这样一种技术，单独提供一个作为纹理源图像的二维图像图案附加到组成物体的多边形表面．而 MIP 映射是内插象素数据的纹理映射技术中的一种，它使附加到多边形上的图案，在三维模型移向视点或离开视点时不会造成不自然感．

而且，图像的绘制速度取决于绘制引擎中对每个多边形的纹理映射或 MIP 映射的处理速度．还有，图像的绘制速度受从绘制引擎到帧缓冲器的写

速度的影响，这样如果帧缓冲器存取速度低，则绘制速度亦低。因此，如果为了增加绘制速度，采用昂贵的高速存贮器作为大容量帧缓冲器，则系统成本会过高地增加。而如果用便宜的动态随机存贮器(DRAM)，则系统的绘制速度会降低。

发明内容

鉴于上述本技术领域的状况，本发明的目的如下面所述。

即，本发明的一个目的是提供一种图像绘制装置和图像绘制方法，从而，即使使用便宜的存贮器，如 DRAM 作为帧缓冲器，也可以维持高绘制速度。

本发明的另一个目的是提供一种图像绘制装置和图像绘制方法，从而在通过绘制装置执行纹理映射的图像绘制装置中，绘制处理可以不停顿绘图装置而执行。

本发明还有一个目的是提供图像绘制装置和图像绘制方法，可以降低对图像存贮器的访问次数和访问时间而增加整个图像绘制速度。

根据本发明的图像绘制装置包括：预处理部件，用于根据绘制由单元图形的组合定义的图像模型的绘制命令，在以单元图形为基础的绘制处理之前产生绘制处理所需的数据；图像绘制部件，用于响应在图像存贮器中绘制图像的绘制命令，通过纹理映射处理以单位图形为基础形成像素数据；存贮部件，用于暂时存贮图像绘制装置进行纹理映射所需的纹理数据；以及供给部件，用于响应由预定处理部件输出的数据，把图像绘制部件进行纹理映射所需的纹理数据提供给存贮部件。

根据本发明的图像绘制装置中，图像绘制部件具有执行 MIP 映射的功能，图像绘制部件响应预处理装置输出的数据，选择具有所需分辨率的纹理数据，以提供所选择的纹理数据给存贮部件。

根据本发明的图像绘制方法包括以下步骤：根据绘制由单元图形的组合定义的图像模型的绘制命令，以单元图形为基础，产生有关图像绘制处理数据；根据该数据，把图像绘制处理中纹理映射所需的纹理数据提供给存贮部件；通过纹理处理产生单元图形的象素数据以便在图像存贮器中绘图，以及在图像存贮器绘制图像。

在根据本发明的图像绘制方法中，把纹理数据供给存贮部件的步骤包括下述步骤：即响应有关图像绘制处理的数据，提供图像绘制处理中 MIP 映射所需分辨率的纹理数据。

根据本发明的图像绘制装置和方法，图像绘制部件通过预处理部件，在预处理阶段，从纹理存贮器发送纹理映射所需的纹理数据到纹理高速缓存。预处理部件和图像绘制部件以流水线方式工作，使得图像绘制过程中不用停止图像绘制部件。

根据本发明的图像绘制装置和方法，图像绘制部件进行 MIP 映射所需分辨率的数据由预处理部件在预处理阶段从纹理存贮器上的纹理数据中选择，并传送到纹理高速缓存，以降低对纹理存贮器的访问次数和访问的时间，提高整个图像的绘制速度。

附图说明：

图 1 是显示采用本发明的视频游戏机的结构框图；

图 2 是显示视频游戏机中 GPU 的详细结构的框图；

图 3 是显示 GPU 的基本结构的框图；

图 4 显示了在 GPU 中纹理高速缓存中数据结构的示例；

图 5 是显示在 GPU 中用预处理器分解第一多边形的处理流程图；

图 6 是显示视频游戏机中第二总线转换器的结构框图；

图 7 说明了视频游戏机中第二总线转换器的结构；

图 8 说明了当对第一多边形形状内部访问时，要被访问的交错图案(interleaving pattern)；

图 9 说明了当对第一多边形形状内部访问时，基于地址访问情况下的屏蔽；

图 10 说明了通过屏蔽得到的访问地址；

图 11 说明了用(4×4)交错图案访问绘制于帧缓冲器的存贮体的第二多边形形状的内部；

图 12 说明了用(4×4)交错图案访问绘制于帧缓冲器的存贮体的第二多边形形状的内部时，所访问的交错图案；

图 13 说明了以地址为基础用(4×4)交错图案访问第二多边形形状内部情况下的屏蔽；

图 14 说明了用(8×2)交错图案对第二多边形的内部的访问；

图 15 说明了用(8×2)交错图案访问绘制于帧缓冲器的存贮体的第二多边形形状内部时，所访问的交错图案；

图 16 说明以地址为基础用(8×2)交错图案访问第二多边形形状内部情

况下的屏蔽;

图 17 说明了用(16 × 1)交错图案对第二多边形的内部的访问;

图 18 说明了用(16 × 1)的交错图案访问绘制于帧缓冲器的存贮体的第二多边形形状的内部时，所访问的交错图案;

图 19 说明了以地址为基础用(16 × 1)交错图案访问第二多边形形状内部情况下的屏蔽.

图 20 说明了计算绘制于帧缓冲器的存贮体的多边形纵横比的处理.

图 21 显示了具有 16 个地址的 5 种交错图案的图案.

图 22 是采用本发明的视频游戏机装置的平面图.

图 23 是该视频游戏装置的后视图.

图 24 是该视频游戏装置的侧视图.

图 25 是装载在该视频游戏装置上的 CD-ROM 的平面图.

具体实施方式

参考附图，下面详细说明本发明的优选实施例. 根据本发明的绘制装置用于图 1 所示的视频游戏装置中. 根据本发明的绘制方法在该视频游戏装置中被实现.

视频游戏机根据用户发出的指令通过读和执行存贮在辅助存贮装置如光盘中的游戏程序执行游戏，其结构如图 1 所示.

具体地，本视频游戏装置具有两类总线，即主总线 1 和副总线 2 .

主总线 1 和副总线 2 经过总线控制器 16 被互相连接.

主总线 1 上连接有: 由微处理器组成的中央处理单元(CPU) 11 ，由随机存取存贮器(RAM)组成的主存贮器 12，主动态存贮器存取控制器或主 DMAC 13， MPEG 解码器 14 和图像处理单元或图形处理单元(GPU) 15 . 副总线 2 上连接有: 由微处理器组成的辅助中央处理单元或副 CPU 21 ， 由随机存取存贮器(RAM)组成的辅助存贮器 22，辅助动态存贮器存取控制器或副 DMAC 23 ，其中存贮有程序例如操作系统的只读存贮器(ROM) 24 ， 声音处理单元(SPU) 25 ， 通信控制器或异步传送模式(ATM) 26 ， 辅助存贮装置 27 ， 输入装置 28 和 CD-ROM 驱动器 30 .

总线控制器 16 是主总线 1 上的一种装置，用于在主总线 1 和副总线 2 之间进行转换，并在初始状态是断开的.

主 CPU 11 是主总线 1 上的一种装置，通过主存贮器 12 上的程序操作. 由于总线控制器 16 在起动时是断开的，所以主 CPU 11 从副总线 2 上的 ROM

24 中读出引导程序，以在副总线 2 上从 ROM 24 复制应用程序和所需的数据，加载到主存贮器 12 或副总线 2 上的装置中．在主 CPU 11 上加载了一个几何变换引擎(GTE)17，以实现如坐标转换的处理．该 GTE 17 具有并行计算机制，以便并行地执行多个计算处理操作，并对 CPU 11 的计算处理请求作出响应，以使对例如坐标转换、光源计算、矩阵或向量运算实现快速处理运算．根据 GTE 17 计算处理运算的结果，主 CPU 11 把三维模型定义为基本单元图形如三角形或四边形的组合，形成与每个多边形相关的绘制命令以便绘制三维图像，且打包绘制命令，以便把结果命令包传送到 GPU 15 中．

主 DMAC 13 是主总线 1 上的装置，用来实现控制，例如主总线 1 上各装置的 DMA 传送．如果总线控制器 16 是断开的，则主 DMAC 13 也控制副总线 2 上的装置．

GPU 15 是主总线 1 上的一装置，起成像处理器的作用．GPU 15 解释从主 CPU 11 或主 DMAC 13 发送的命令包形式的绘制命令，根据形成多边形的所有象素的指定深度的 Z 值和彩色值，在帧缓冲器 18 中实现写象素数据的成像处理．

MDEC 14 是一个同主 CPU 11 可并行操作的 I/O 连接装置及作为图像扩展引擎的装置．MDEC 14 解码用正交变换，例如离散余弦变换压缩和编码的图像数据．

在副总线 2 上，副 CPU 21 是在副总线 2 上的装置，它根据副存贮器 22 中的程序工作．

副 DMAC 23 是副总线 2 上的装置，用来对例如副总线 2 上的装置执行控制如 DMAC 传送．该副 DMAC 23 只在总线控制器 16 被关闭时能够获取总线权．

SPU 25 是副总线 2 上起声音处理器作用的装置．SPU 25 响应来自副 CPU 21 或副 DMAC 23 的作为命令包送出的声音命令，从声音存贮器 29 读出声音源数据以输出该读出的数据．

ATM 26 是副总线 2 上的通信装置．

辅助存贮装置 27 是副总线 2 上的输入输出装置，由非易失性存贮器例如闪速存贮器组成．该辅助存贮装置 27 用来暂存数据如游戏进展或得分．

CD-ROM 驱动器 30 是副总线 2 上的数据输入装置，用来从 CD-ROM 上复制应用程序和所需的数据．

输入装置 28 是从其它一些设备输入的装置，如副总线 2 上的控制板、人－机接口、图像输入或语音输入。

也就是，在上面所述的视频游戏机中，几何处理系统执行几何处理，如坐标转换、剪切或光源计算．形成绘制命令，用于将三维模型定义为基本单元图形(多边形)如三角形或四边形的组合，以便绘制三维图形，且把同每个多边形相关的绘制命令作为命令包送到主总线 1，该几何处理系统由主总线 1 上的主 CPU 11 和 GTE 17 形成．另外，在视频游戏机中，成像处理系统由 GPU 15 组成，成像处理系统根据几何处理系统的绘制命令，形成每个多边形的象素数据，用于通过在帧缓冲器 18 中绘图的成像处理，在帧缓冲器 18 绘图．

上述 GPU 15 将予以详细说明．

参考图 2， GPU 15 包括一个连到如图 1 所示的主总线 1 上的包引擎 31，并且通过预处理器 32 和绘制引擎 33，实现将每个多边形的象素数据写入帧缓冲器 18 中的成像处理，根据通过主总线 1 从图 1 所示的 CPU 11 或主 DMAC 送到包引擎 31 的命令包形式的绘制命令，读出在帧缓冲器 18 中所绘图像的象素数据，并把所读出的象素数据作为视频信号经 CRT 控制器 34 提供给电视接收机或监视器接收机(未示出)．

包引擎 31 展开通过主总线 1 从图 1 所示的主 CPU 11 或主 DMAC 13 发送到一个寄存器(未示出)的命令包．

预处理器 32 根据按命令包送给包引擎 31 的绘制命令产生多边形数据，并实现预定的预处理，如在后面将说明的对多边形数据的多边形分割，同时产生各种数据，如绘制引擎 33 所需的各个多边形的顶点坐标信息、纹理或 MIP 映射纹理的地址信息，或象素交错的控制信息．

绘制引擎 33 包括：连到预处理器 32 上的 N 个多边形引擎 33A1、33A2、…、33AN，连到多边形引擎 33A1、33A2、…、33AN 上的 N 个纹理引擎 33B1、33B2、…、33BN；连到纹理引擎 33B1、33B2、…、33BN 的第一总线转换器 33C；连到第一总线转换器 33C 的 M 个象素引擎 33D1、33D2、…、33DM；连到象素引擎 33D1、33D2、…、33DM 的第二总线转换器 33E；连到第二总线转换器 33E 的纹理高速缓存 33F；和连到纹理高速缓存 33F 上的 CLUT 高速缓存 33G．

在绘制引擎 33 中，N 个多边形引擎 33A1、33A2、…、33AN 根据由

预处理器 32 预处理的多边形数据，按绘制命令顺序地产生多边形，以便逐个对多边形实现并行阴影处理。

N 个纹理引擎 33B1 、 33B2 、…、 33BN ，在多边形引擎 33A1 、33A2 、…、 33AN 产生的每个多边形上，基于经彩色查找表(CLUT)高速缓存 33G，从纹理高速缓存 33F 提供的纹理数据，并行地实现纹理映射或 MIP 映射。

预处理器 32 预先提供地址信息如附加到由 N 个纹理引擎 33B1 、33B2 、…、 33BN 处理的多边形上的纹理或 MIP 映射纹理给纹理高速缓存 33F 。根据上面的地址信息，纹理映射所需纹理数据从纹理区传送到帧缓冲器 18，同时只有 MIP 映射所需的分辨率数据从相关的纹理数据中被选择，以便作为 MIP 映射纹理数据被传送。在纹理绘制期间要参考的 CLUT 高速缓存 33G 的 CLUT 数据从 CLUT 区传到帧缓冲器 18 。

由 N 个纹理引擎 33B1 、 33B2 、…、 33BN 通过纹理映射或 MIP 映射处理的多边形数据，经第一总线转换器 33C 传到 M 个象素引擎 33D1 、33D2 、…、 33DM 。

M 个象素引擎 33D1 、 33D2 、…、 33DM 执行各种图像处理操作，如并行地进行 Z －缓冲器处理或抗混叠处理，以产生 M 个象素数据。

由 M 个象素引擎 33D1 、 33D2 、…、 33DM 产生的 M 个象素数据，经第二总线转换器 33E 被写到帧缓冲器 18 中。

第二总线转换器 33E 从预处理器 32 中得到象素交错控制信息。第二总线转换器 33E 具有下述功能，即，根据上述控制信息，从 M 个象素引擎 33D1 、33D2 、…、 33DM 产生的 M 个象素数据中选择 L 个象素数据，执行象素交错处理，用符合绘制在帧缓冲器 18 上的多边形形状的 M 个存贮地址作为存取单元，写 M 个象素数据。

绘制引擎 33 根据由预处理器 32 预处理的多边形数据，在帧缓冲器 18 中产生和写入每个多边形的全部象素数据，以在帧缓冲器 18 上绘制由绘制命令定义为多边形的图。因此，在帧缓冲器 18 绘制的图像的象素数据通过第二总线转换器 33E 被读出，以便作为视频信号经 CRTC 34 提供给电视接收机或监视器接收机(未示出)。

在上面所述的 GLUT 15 结构中，预处理器 32 根据多边形的顶点坐标[(X0， Y0)，(X1， Y1)，(X2， Y2)]及纹理坐标[(U0， V0)， (U1， V1)，

(U2 , V2)], 产生地址信息, 用于预读附加在由 N 个纹理引擎 33B1 、33B2 、…、 33BN 处理的多边形上的纹理. 此外, GLUT 15 根据多边形的各边的斜率[(X1 - X0)/(Y1 - Y0), (X2 - X0)/(Y2 - Y0), (X1 - X2)/(Y1 - Y2)], 纹理地址的斜率[(U1 - U0)/(Y1 - Y0), (U2 - U0)/(Y2 - Y0), (U1 - U2)/(Y1 - Y2)], [(V1 - V0)/(Y1 - Y0), (V2 - V0)/(Y2 - Y0), (V1 - V2)/(Y1 - Y2)], …或多边形面积重新产生 MIP 映射选择信息, 把信息提供给纹理高速缓存 33F . GLUT 15 按左边缘顶点的顺序[(X0 , Y0)→(X1 , Y1)→(X2 , Y2)]顺序, 或按右边沿顶点的顺序[(X2 , Y2)→(X1 , Y1)→(X0 , Y0)]或用扫描二端点或纹理地址的顺序排序多边形顶点坐标[(X0 , Y0), (X1 , Y1), (X2 , Y2)].

预处理器 32 存贮对应于工作存贮器中预处理的多边形数据的信息(未示出), 在绘图引擎 33 能处理下一个多边形的阶段, 能够处理一个多边形的信息可从工作存贮器传送到 N 个多边形引擎 33A1 、 33A2 、…、 33AN , 这使得绘制引擎 33 启动对新多边形的绘制处理.

也就是用本 GPU 15 , 预处理器 32 和绘制引擎 33 以流水线处理执行绘制处理, 以便绘制由绘制命令定义为多边形的组合的图像.

用该流水线处理进行的绘制处理在以后还会说明.

预处理器 32 对前面所述的多边形数据执行预定的预处理, 并提供各种数据给绘制引擎 33 , 这些数据如绘制引擎 33 所需的顶点坐标信息、用于纹理或 MIP 映射纹理的地址信息、或例如作象素交错的控制信息.

绘制引擎 33 从预处理器 32 接收数据, 以从纹理高速缓存 33D 中读出所需的纹理数据, 产生数据以便在帧缓冲器 18 中写所产生的象素数据. 纹理高速缓存 33D , 根据预处理器 32 预处理计算得到的需要的纹理地址, 从帧缓冲器 18 读出纹理区的纹理数据. 此纹理数据被读出, 以便在利用纹理数据的图像绘制开始之前完成数据的读出. 由于只从纹理区读对应于 MIP 映射所需分辨率的纹理数据, 访问纹理区的次数被减少了.

如图 4 中示例, 在纹理高速缓存 33F 中的数据结构包括: 由纹理地址形成的标签区 TAG , 存贮所需的纹理数据的存贮区 DATA , 和指出还未被使用的纹理数据的标志 L . 为利用标志 L 复位的项, 纹理高速缓存 33 从帧缓冲器 18 的纹理区读入纹理数据, 置位其标志 L . 绘制引擎 33 从标志置位的这个项读出相应纹理数据, 以便执行绘制处理, 在绘制结束的阶段复位此项的

标志L，该纹理数据已不再需要。

在绘制设备中，如上所述执行纹理映射处理，预处理器 32 和绘制引擎 33 按流水线构成，这样以使绘制引擎 33 所需的纹理数据从纹理存贮器即帧缓冲器 18 的纹理区通过预处理器 32，在预处理阶段送到高速缓存 33F，以便绘制处理能够被执行，无须停止绘制引擎 33。此外，由于只从纹理区读取同 MIP 映射所需分辨率相关的数据，可以减少访问纹理区的次数，从而提高了整个装置的绘制速度。

通过预处理器 32 的多边形分割处理按照图 5 所示的流程图执行。

具体地说，多边形分割处理开始时，指定多边形的数量的多边形计数 C 设定为 1。

在第一处理步骤 S1，判断是否需要分割多边形。在处理步骤 S1 的判断处理中，要判断是否现在处理的多边形包含在纹理高速缓存 33F 的范围内。为了这个判断处理，看是否多边形顶点的纹理坐标[(U0， V0)，(U1， V1)，(U2， V2)]计算值落在一个纹理页中已足够了。

如果在处理步骤 S1 的判断结果是“是”，即，如果多边形需要分割，处理转到步骤 S2，以将多边形分成 N 份。在处理步骤 S2 上，多边形的这种 N 分割通过在中性点上分割多边形的所有边而实现，如下所示:

$$X0'=(X0+X1)/2$$

$$Y0'=(Y0+Y1)/2$$

$$Z0'=(Z0+Z1)/2$$

$$X1'=(X1+X2)/2$$

$$Y1'=(Y1+Y2)/2$$

$$Z1'=(Z1+Z2)/2$$

$$X2'=(X2+X0)/2$$

$$Y2'=(Y2+Y0)/2$$

$$Z2'=(Z2+Z0)/2$$

$$U0'=(U0+U1)/2$$

$$V0'=(V0+V1)/2$$

$$Z0'=(Z0+Z1)/2$$

$$U1'=(U1+U2)/2$$

$$V1'=(V1+V2)/2$$

Z1'=(Z1+Z2)/2

U2'=(U2+U0)/2

V2'=(V2+V0)/2

Z2'=(Z2+Z0)/2

R0'=(R0+R1)/2

G0'=(G0+G1)/2

B0'=(B0+B1)/2

R1'=(R1+R2)/2

G1'=(G1+G2)/2

B1'=(B1+B2)/2

R2'=(R2+R0)/2

G2'=(G2+G0)/2

B2'=(B2+B0)/2

即，在处理步骤 S2，多边形的 N 分割中，多边形的所有边在其中点被分割，以将三角形状多边形分割为例如 N = 4 的新多边形。

在下一处理步骤 S2，通过设置多边形计数 C 为 C = C + N - 1 而改变多边形的数量。然后，处理回到第一处理步骤 S1，以便判断是否新被分割的多边形应该进一步再分割。重复执行处理步骤 S1 到 S3，直到新分割多边形被包括在纹理高速缓存的范围之内。

如果在第一处理步骤 S1 上判定的结果是“否”，也就是如果不需要分割多边形，则处理转到下一处理步骤 S4。

在处理步骤 S4 上，一个多边形的预处理信息被提供给多边形引擎 33A1、33A2、…、33AN 以启动成像处理，然后，不必等待成像处理的结束，处理转到下一个处理步骤 S5。

在处理步骤 S5 上，多边形计数 C 被减量。

在下一个处理步骤 S6 上，判断是否多边形计数 C 等于‘0’。如果在处理步骤 S6 处理结果是“否”，C ≠ 0，存在多边形要处理，则处理返回到第一处理步骤 S1，进入下一个多边形的处理。如果在处理步骤 S6 处理结果是“是”，也就是如果所有多边形已完成成像，因此无多边形要分割，处理结束。

也就是，预处理器 32 判断是否在绘制引擎 33 中，现在处理的多边形包含在纹理高速缓存 33F 中(判断条件 1)。预处理器 32 根据判断结果执行分割

处理，相应于绘制命令分割多边形，以使新分割的多边形被包括在纹理高速缓存 33F 中．这样允许根据绘制引擎 33 中从纹理高速缓存 33f 经过 CLUT 高速缓存 33G 读出的纹理数据，可靠而高效率地执行纹理映射处理．

用预处理器 32 的多边形分割处理中，可以检查上述第一处理步骤中是否多边形中的象素数小于规定的值(判断条件 2)，以检查是否多边形需要被分割及在处理步骤 S2 按绘制命令把多边形分成多个在二维空间中的多边形，以使新分割的多边形中的象素数不大于规定值．这样，要由绘制引擎处理的多边形的尺寸可以相等．多边形中的象素数的判断亦可以通过寻找多边形顶点的外积值求出面积及检查是否该值小于最佳值．

在用预处理器 32 的多边形分割处理中，在上述处理步骤 S2 对应于绘制命令的多边形能被分割成多个三维空间中的多边形．

在这种情况下，可在第一处理步骤 S1 中判断是否多边形顶点的 Z 值的最小值和最大值之间差包括在最佳范围中(判断条件 3)，以检查是否多边形需要被分割，并在处理步骤 S2 把对应于绘制命令的多边形分成多个三维空间中的多边形，以使新分的多边形的象素数包括在规定范围内，以限制多边形尺寸，根据从纹理高速缓存 33F 经 CLUT 高速缓存 33G 读出的纹理数据，以仅有很小纹理失真情况，执行纹理映射处理．

在这种情况，可在第一处理步骤 S1，判断是否多边形顶点的 Z 值的最小和最大值参考的 MIP 映射纹理被跨越(判断条件 4)，并根据判断结果在处理步骤 S2，把对应于绘制命令的多边形分成多个在三维空间的多边形，以限制在多边形中参考的 MIP 映射的参考范围，根据纹理高速缓冲器 33F 经 CLUT 高速缓存 33G 读出的 MIP 映射纹理数据，进行有效的 MIP 映射．

亦可以根据是否多边形中的象素数不大于预定值，来判断是否多边形需要分割，根据该判断结果，在处理步骤 S2 把对应于绘制命令的多边形分成多个在三维空间中的多边形，使得分割的新多边形内的象素数小于上述规定值．

同样，也可以根据例如多边形的象素数，预测绘制引擎 33 的绘制处理时间．在处理步骤 S1，根据是否预处理器 32 的预处理时间同绘制引擎 33 的绘制处理时间平衡(判断条件 5)判断是否多边形需要被划分，以及根据判断的结果，在处理步骤 S2，对应于绘制命令分割多边形，以使预处理器 32 的预处理时间同绘制引擎 33 的绘制处理时间平衡．这使得可平衡预处理器 32

的预处理时间和绘制引擎 33 的绘制处理时间彼此间关系，以便用流水线构建预处理器 32 和绘制引擎 33，以有效地实现高速绘图.

亦可以在处理步骤 S1 判断是否处理引擎 33 所处理的多边形的形状在上述处理步骤 S1 适合象素交错(判断条件 6)，并通过处理步骤 S2 对应于绘图命令把多边形分成形状适合于象素交错的多个新的多边形. 这使得可以通过绘制引擎 33 有效地访问帧缓冲器 18，以实现高速绘制处理.

同样可以在处理步骤 S1，根据上述各种各样判断条件的组合，判断是否多边形需要分割，并通过处理步骤 S2 对应于绘制命令把多边形分成多个新多边形，使新分割的多边形满足判断条件.

具体地说，可以在处理步骤 S1，根据判断条件 1 和 2 的组合，判断是否多边形需要分割，并通过处理步骤 S2 对应于绘制命令把多边形分成多个新多边形，使新分割的多边形满足判断条件 1 和 2，以便经绘制引擎 33 处理的多边形的尺寸相等，及根据从纹理高速缓存 33F 经 CLUT 高速缓存 33G 读出的纹理数据可靠有效地执行纹理映射处理.

也可以在处理步骤 S1，根据判断条件 1 和 3 的组合，判断是否多边形需要分割，并在处理步骤 S2，对应于绘制命令把多边形分成多个新多边形，以使新分的多边形满足判断条件 1 和 3. 这使得可以根据从纹理高速缓存 33F 经 CLUT 高速缓存 33G 读出的纹理数据，仅以很小的纹理失真可靠有效地实现纹理映射处理. 另外，如果判断条件 2 同上面的组合相组合，可以使由绘制引擎 33 处理的多边形的尺寸、也就是象素数相等，以便执行纹理映射.

也可以在处理步骤 S1 根据判断条件 1 和 4 的组合，判断是否多边形需要分割，且在处理步骤 S2，对应于绘制命令把多边形分成多个新的多边形，以使新分割多边形满足判断条件 1 和 4. 这使得可以根据从纹理高速缓存 33F 经 CLUT 高速缓存 33G 读出的纹理数据，可靠有效地实现 MIP 映射处理. 另外，如果判断条件 2 和 3 与上面的组合相组合，则可能使绘制引擎 33 处理的多边形尺寸，也就是其象素数相等，且减少纹理失真.

亦可以在处理步骤 S1，根据判断条件 1 和 5 的组合，判断是否多边形需要分割，且在处理步骤 S2，对应于绘制命令把多边形分成多个新的多边形，以使新分割多边形满足判断条件 1 和 5，这使得可以保持预处理器 32 的处理时间同绘制引擎 33 的处理时间均衡，以用流水线实现有效高速的纹理映射. 另外，如果判断条件 2 和 3 与上面的组合相组合，则可能使由绘制引

擎 33 处理的多边形尺寸，也就是其象素数相等，且减少纹理失真．判断条件 4 亦可以同上面的组合相组合，以实现 MIP 映射．

亦可以在处理步骤 S1，根据判断条件 1 和 6 的组合，判断是否多边形需要分割，并由处理步骤 S2 对应于绘制命令把多边形分成多个新多边形，以使新分的多边形满足判断条件 1 和 6．这使得可以通过绘制引擎 33 可靠有效地实现纹理映射，且有效地访问帧缓冲器 18 以实现高速绘制．另外，如果判断条件 2 和 3 同上面的组合相组合，则能使绘制引擎 33 处理的多边形尺寸，也就是其象素数相等，以减少纹理失真．该判断条件 4 亦可以同上面的组合相组合以实现 MIP 映射，或者判断条件 5 可以同上面的组合相组合，以便用流水线实现高速处理．

通过上述第二总线转换器 33E 进行的象素交错如下所述被实现．

参考图 6，第二总线转换器 33E 包括：一个控制电路 101，由图 2 所示的预处理器 32 的输出供给；一个选择器 102，由控制电路 101 的输出供给；以及多个多路转换器/多路分解器(MUX/DMUX) 103a、103b、103c、103d、…，它们都由选择器 102 的输出供给．

(MUX/DMUX) 103a、103b、103c、103d、…被连到帧缓冲器 18 及绘制引擎 33，如图 2 所示．

帧缓冲器 18 由多个存贮体[1]、[2]、…、[X]、…、[L]组成，如图 2 所示．每个存贮体[1]、[2]、…、[X]、…、[L]由被 16 个地址表示的矩形区(交错图案)组成，所以可同时访问 16 个地址．

因此，例如帧缓冲器 18 的存贮体[X]具有 16 个输入/输出端口 P_0 到 P_{15} 以访问地址 A_0 到 A_{15}．MUX/DMUX 103a、103b、103c、103d、…中的 4 个 MUX/DMUX 103a、103b、103c、103d 每个连到 16 个输入/输出端口 P_0 到 P_{15}．

另外，4 个 MUX/DMUX 103a、103b、103c、103d 被连到绘制引擎 33 的 4 个相关象素引擎 $33D_{X1}$、$33D_{X2}$、$33D_{X3}$ 和 $33D_{X4}$．

由于存贮体[X]以外的存贮体的构成类似于上面所述的存贮体[X]，因此为了简化起见，不对它们作详细说明．由第二总线转换器 33E 在其它存贮体上执行的访问操作类似于在以后要说明的、在存贮体[X]上由第二总线转换器 33E 执行的操作．因此，在下面说明中，只解释存贮体[X]上第二总线转换器 33E 执行的访问操作．

首先，解释第二总线转换器 33E 的一系列操作。

如果例如绘制在存贮体[X]上的多边形的形状是三角形 T_{ABC}(第一多边形的形状)，如图 7 所示，则控制电路 101 首先用象素交错控制信息由预处理器 32 馈送。根据预处理器 32 的象素交错控制信息，控制电路 101 转换交错图案成为例如(4 × 4)交错图案，用来访问三角形 T_{ABC} 的内部。

在控制电路 101 中转换交错图案的方法将在后面详述。

在存贮体[X]上形成多个交错图案，这种将被访问的交错图案，也就是说，这种允许三角形 T_{ABC} 的内部作为一个整体被访问的交错图案，用（ 4 × 4 ）的交错图案，通过控制电路 101 被检测。

因此，在三角形 T_{ABC} 中，如果在存贮体[X]上的每个交错图案用 P(在 x 方向上的图案索引和在 y 方向上的图案索引)来表示，用

$$
\begin{aligned}
P(x,\ y) = {} & P(3,\ 1),\ P(4,\ 1),\ P(1,\ 2),\\
& P(2,\ 2),\ P(3,\ 2),\ P(4,\ 2),\\
& P(1,\ 3),\ P(2,\ 3),\ P(3,\ 3),\\
& P(4,\ 3),\ P(5,\ 3),\ P(2,\ 4),\\
& P(3,\ 4),\ P(4,\ 4),\ P(5,\ 4),\\
& P(3,\ 5),\ P(4,\ 5),\ P(5,\ 5),\\
& P(4,\ 6),\ P(5,\ 6)
\end{aligned}
$$

所指定的总数为 20 的交错图案被检测，如图 8 所示。

控制电路 101 把如上所述检测的指定 20 个交错图案的图案信息，按交错图案送到选择器 102 。当按地址执行存贮器访问时，控制电路 101 把对应于三角形 T_{ABC} 形状的屏蔽信息送到选择器 102 。

基于从控制电路 101 按交错图案提供的图案信息，选择器 102 指示对应于 MUX/DMUX 103a 、 103b 、 103c 和 103d 所要访问的（ 4 × 4 ）交错图案 P 的地址。

如果从控制电路 101 得到屏蔽信息，选择器 102 根据屏蔽信息，以（ 4 × 4 ）交错图案 P 执行屏蔽的结果得到 MUX/DMUX 103a 到 103d 的访问地址，如图 9 所示。因此， 在图 9 所示的 P(4 ， 1)所指定的交错图案中，作为屏蔽结果得到的访问地址是 A4 、 A5 、 A6 、 A8 、 A9 、 A10 、 A13 、 A14 和 A15 ，如图 10 中阴影线所示。

MUX/DMUX 103a 、 103b 、 103c 和 103d 访问由选择器 102 指示的存

贮体[X]的地址 A_0 到 A_{15}.

象素引擎 $33D_{X1}$、$33D_{X2}$、$33D_{X3}$ 和 $33D_{X4}$ 分别输出象素数据给 MUX/DMUX 103a、103b、103c 和 103d，如上所述.

因此，MUX/DMUX 103a 访问选择器 102 所指示的地址，从象素引擎 Xa，经过对应于选择器 102 所指示地址的输入/输出端口 P_0 到 P_{15} 中的一个，在由上述地址所指示的存贮体[X]的区域中，写入象素数据.

MUX/DMUX 103a 访问选择器 102 所指示的地址，以经过对应于上面地址的输入/输出端口 P_0 到 P_{15} 中的一个，读出写入存贮体[X]上的地址指示的区域中的数据. MUX/DMUX 103a 对从存贮体[X]读出的数据，执行预定处理.

由于 MUX/DMUX 103b 到 103d 的操作类似于上述 MUX/DMUX 103a 的操作，因此，为了清楚，详细说明被省略.

对于在上述控制电路 101 中交错图案的改变方法，现在作具体的说明.

首先，说明以（4 × 4）交错图案 P 访问图 11 所示的绘制在存贮体[X]中的形状为横向拉长的三角形 T_{DEF}(第二多边形的形状) 的多边形的内部的次数.

在这种情况下，要被访问的交错图案是

$$\begin{aligned} P(x,\ y) = {} & P(1,\ 1),\ P(2,\ 1),\ P(3,\ 1), \\ & P(4,\ 1),\ P(5,\ 1),\ P(0,\ 2), \\ & P(1,\ 2),\ P(2,\ 2),\ P(3,\ 2), \\ & P(4,\ 2),\ P(5,\ 2),\ P(6,\ 2), \\ & P(7,\ 2),\ P(8,\ 2),\ P(7,\ 3), \\ & P(8,\ 3),\ P(9,\ 3), \end{aligned}$$

总数为 17，如图 12 所示.

也就是为了用（4 × 4）交错图案访问三角形 T_{DEF} 的内部，作为一个整体访问三角形 T_{DEF} 的内部的访问次数是 17.

在根据地址存取时，仅要求的存贮器地址可通过执行（4 × 4）交错图案 P 的屏蔽而被访问，就像访问上面所述三角形 T_{ABC} 的情况，如图 13 所示.

而如果三角形 T_{DEF} 的内部用图 14 所示的（8 × 2）交错图案被访问，要被访问的交错图案是

$$P_1(x,\ y) = P_1(3,\ 1),\ P_1(2,\ 2),$$

$P_1(0, 3), P_1(1, 3),$
$P_1(2, 3), P_1(0, 4),$
$P_1(1, 4), P_1(2, 4),$
$P_1(3, 4), P_1(1, 5),$
$P_1(2, 5), P_1(3, 5),$
$P_1(4, 5), P_1(3, 6),$
$P_1(4, 6)$

总数为 15，如图 15 所示.

也就是如果用（ 8 × 2 ）交错图案访问三角形 T_{DEF} 的内部，为访问三角形 T_{DEF} 的整个内部所需的访问次数为 15 .

在按地址访问时，在（ 8 × 2 ）交错图案 P_1 范围实现屏蔽，如上面所述的访问三角形 T_{ABC} 的情况，如图 16 所示，只访问所需的存贮器地址.

而如果如 17 所示用（ 16 × 1 ）交错图案 P_2 访问三角形 T_{DEF} 的内部，要被访问的交错图案是

$$P_2(x, y) = P_2(0, 5), P_2(1, 5),$$
$P_2(0, 6), P_2(1, 6),$
$P_2(0, 7), P_2(1, 7),$
$P_2(0, 8), P_2(1, 8),$
$P_2(0, 9), P_2(1, 9),$
$P_2(0, 10), P_2(1, 10),$
$P_2(2, 10), P_2(1, 11),$
$P_2(2, 11), P_2(1, 12),$
$P_2(2, 12), P_2(2, 13),$

总数为 18，如图 18 所示.

也就是如果用（ 16 × 1 ）交错图案访问三角形 T_{DEF} 的内部，为访问三角形 T_{DEF} 的整个内部所需的访问次数为 18 .

在按地址访问时，在（ 8 × 2 ）交错图案 P_2 的范围实现屏蔽，如上面所述，如同访问三角形 T_{ABC} 的情况，如图 19 中所示，只访问所需的存贮器地址.

如上所述，用（ 4 × 4 ）交错图案 P 访问三角形 T_{DEF} 内部的次数为 17，而用（ 8 × 2 ）交错图案 P_1 访问三角形 T_{DEF} 内部的次数为 15，用（ 16 × 1 ）

交错图案 P_2 访问三角形 T_{DEF} 内部的次数为 18．因此，用（8 × 2）交错图案 P_1 访问三角形 T_{DEF} 内部的次数为最小访问次数。可以看出，对三角形 T_{DEF} 最合适的交错图案是（8 × 2）交错图案 P_1。

因此，为了把用于访问存贮体[X]的交错图案转换成合适的交错图案以满足将被访问的多边形形状，控制电路 101 执行下述处理操作.

例如如果绘制在存贮体[X]的多边形的形状是一个三角形 T_{HIJ}，如图 20 所示，用于象素交错的控制信息如上所述从预处理器 32 提供给控制电路 101．例如，用于象素交错的控制信息是三角形 T_{HIJ} 的三个顶点的 xy 坐标 H(Xh， Yh)、I(Xi， Yi)或 J(Xj， Yj)这样的信息。

而控制电路 101 利用来自预处理器 32 用于象素交错的控制信息，用如下计算求三角形 T_{HIJ} 的纵横比 R：

$$R = dy/dx$$

$$= (MAXy - MINy)/(MAXx - MINx)$$

这里 MAXx 和 MINx 分别为 X 方向的最大和最小值，而 MAXy 和 MINy 分别为 Y 方向的最大和最小值，如图 20 所示。

在三角形 T_{HIJ} 中:

MAXx = Xj

MINx = Xi

MAXy = Yh

MINy = Yi

根据这样找到的纵横比 R，控制电路 101 选择(1 × 16)、(2 × 8)、(4 × 4)、(8 × 2)和(16 × 1)5 个交错图案 Pa 到 Pe 中的一个，如图 21 所示，且把访问三角形 T_{HIJ} 内部所用的交错图案转换成所选择的交错图案.

控制电路 101 具有一个关于纵横比 R 和交错图案的表(表 1)．在该表中，有预定的与各种纵横比 R 值相关的合适的交错图案，也就是最小访问次数的交错图案．于是控制电路 101 用上述表选择如上获得的同纵横比 R 相关的合适的交错图案.

表 1

纵横比 R	交错图案
~ 0.1	Pa(16 × 1)
0.1 ~ 0.5	Pb(8 × 2)

0.5 ~ 2.0	Pc(4 × 4)
2.0 ~ 8.0	Pd(2 × 8)
8.0 ~	Pe(1 × 16)

在第二总线转换器 33E 中，图 21 所示的 5 种交错图案 Pa 到 Pe 的合适的一种被选择以响应要绘制在存贮体[X]的多边形形状，该存贮体[X]将用所选择的交错图案访问，以使多边形用最小访问操作次数绘制到存贮体[X]. 因此，可以用第二总线转换器 33E 有效地实现存贮器访问.

GPU 15 通过针对提高存贮器访问效率的第二总线转换器 33E 来访问帧缓冲器 18，用于进行各种数据的处理操作，如上所述，以实现有效数据处理操作.

采用本发明的视频游戏机 201 的构成如图 22 的平面视图、图 23 的前视图及图 24 的侧视图所例示.

也就是说，本发明的视频游戏机 201 基本上由主体部件 202、和用电缆 227 同主体部件 202 相连接的操作装置 217 组成，如图 22 所示. 主体部件 202 上表面的中部安装一盘装载单元 203，一个示于图 25 的 CD-ROM 251 被装载在其中，盘装载单元 203 的左边装有一个电源开关 205，用来操作接通或关闭设备，还装有一个复位开关 204，用来操作临时复位游戏. 盘装载单元 203 的右边装有一个盘动作开关 206，用来相对于盘装载部件 203 进行装载或卸载 CD-ROM 251 的操作.

在主体部件 202 的前面装有连接部分 207A、207B，如图 23 所示. 这些连接部分 207A、207B 装有从操作装置 217 来的电缆 227 前端提供的连接端子 226，还装有一个连接端子插入部分 212，用来连接如存贮卡等记录装置 228、及记录/插入部分 208. 也就是说，每组有操作装置 217 和记录装置 218 的两组可连到主体部件 202.

图 23 的前视图显示了这种状态，其中连接端子部分 226 和记录装置 228 被装在右边连接部分 207B，既没有连接端子部分 226，也没有记录装置 228 装在左边连接部分 207A. 用于装记录装置 228 到其上的记录插入部分 208 装有门板 209，如图 23 所示. 当在主体部件 202 上装记录装置 228 时，门板被记录装置 228 的前端部分推开，以便进行装载.

连接端子部分 226 有一个滚花的把持部分 231A 起防滑作用. 同样地，记录装置 228 有一个滚花的把持部分 242A 起防滑作用. 参见侧视图 24，连

接端子部分226具有的长度L基本上等于记录装置228的长度.

操作装置227具有可用左右手把持的支持物220、221．支持物220、221的前端部分分别装有操作部件218、219．操作部件224、225用两手的食指操作，而操作部件218、219用两手的拇指操作.

在操作部件218和219之间提供一个选择开关222用于在游戏期间执行选择，还提供一个启动开关223用于启动游戏.

采用本视频游戏机201，装载在盘装载单元203上的CD-ROM 251，用CD-ROM驱动器30再现．操作装置217相当于上面所述的输入装置28，而记录装置228相当于辅助存贮装置27．

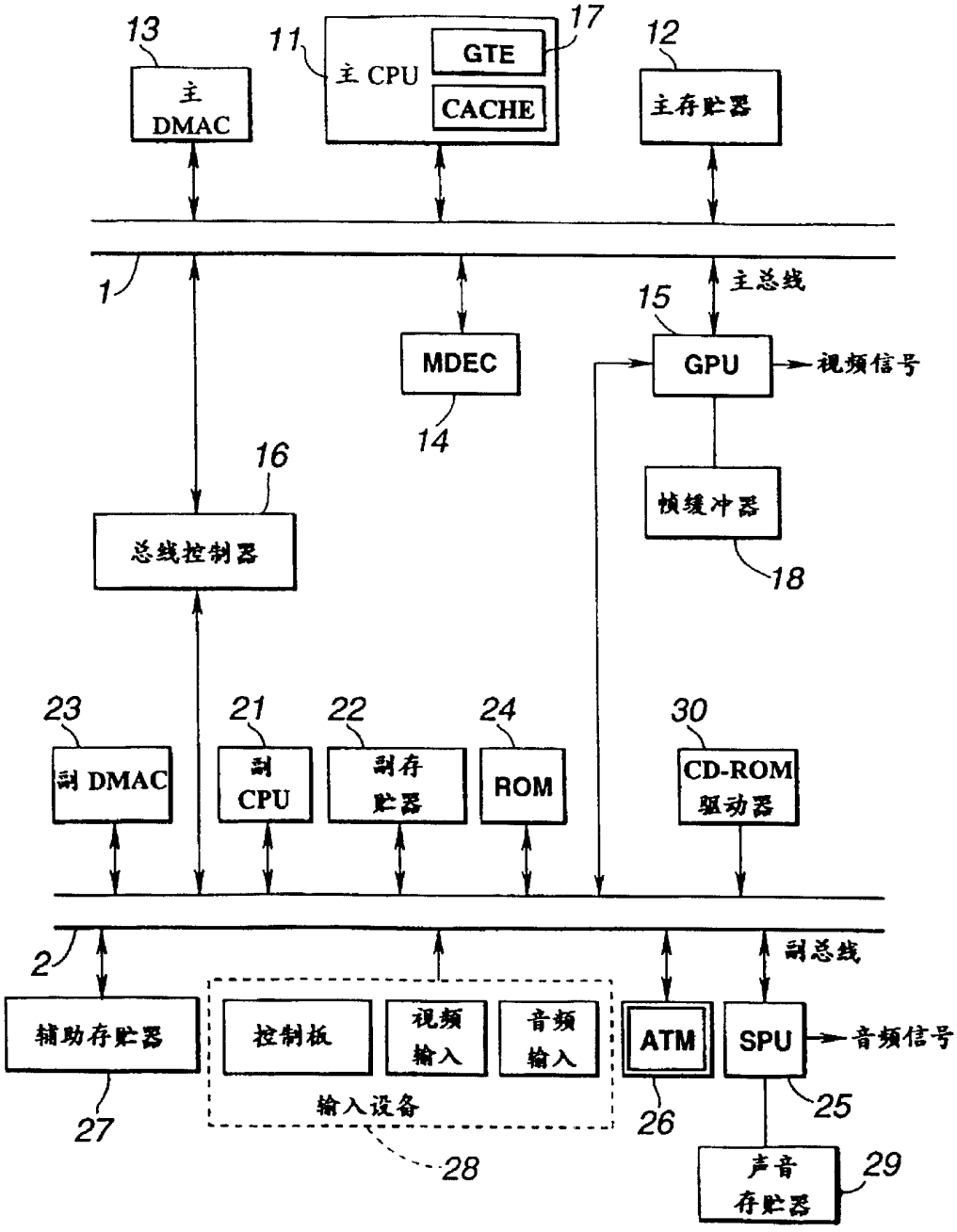

13
主
DMAC
11
主 CPU
GTE
CACHE
17
12
主存贮器
1
主总线
15
MDEC
14
GPU
视频信号
16
总线控制器
帧缓冲器
18
23
副 DMAC
21
副
CPU
22
副存
贮器
24
ROM
30
CD-ROM
驱动器
2
副总线
辅助存贮器
27
控制板
视频
输入
音频
输入
输入设备
28
ATM
26
SPU
音频信号
25
声音
存贮器
29

图 1

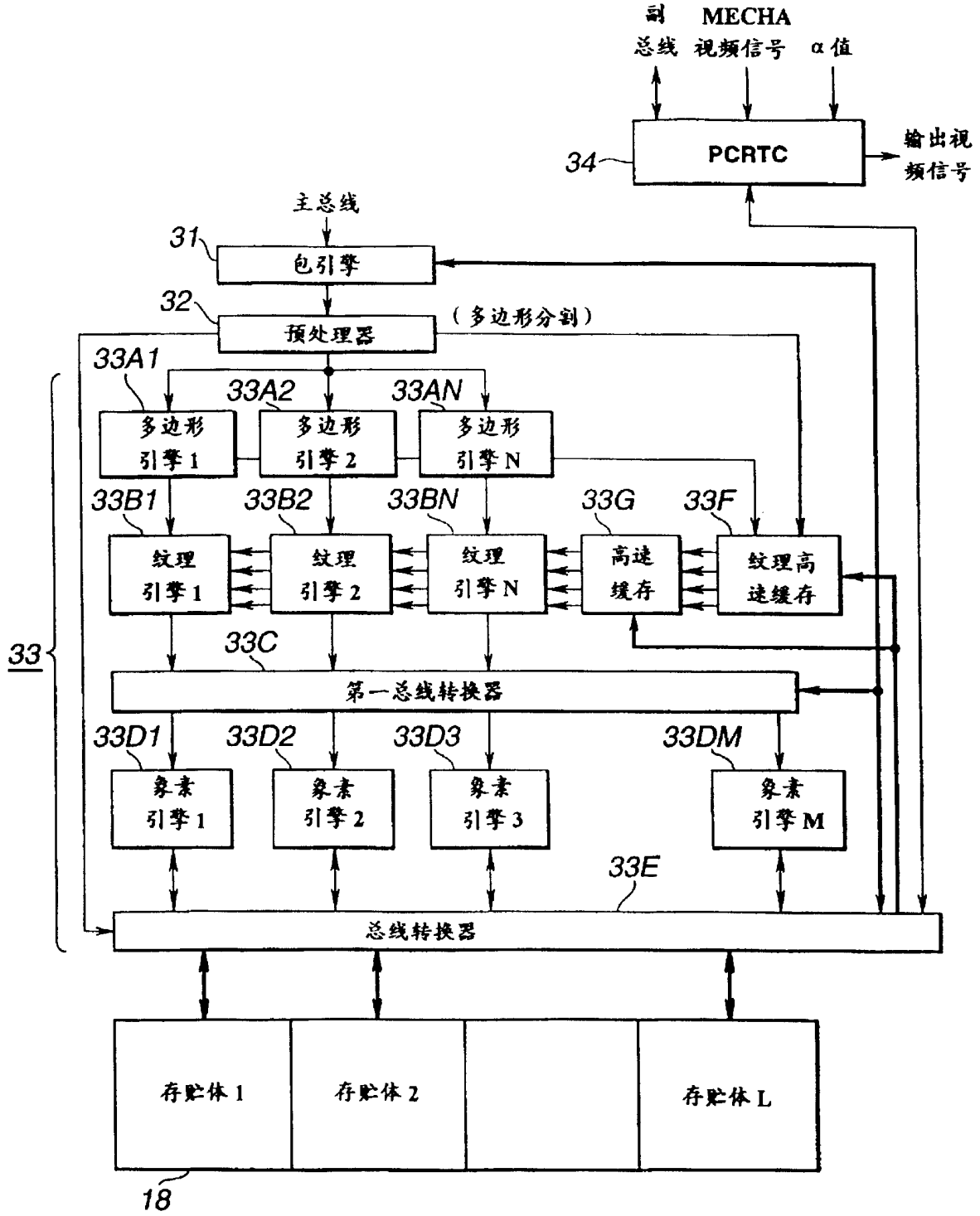

副 MECHA
总线 视频信号 α值
34
PCRTC
输出视频信号
主总线
31
包引擎
32
预处理器
(多边形分割)
33A1
33A2
33AN
多边形引擎1
多边形引擎2
多边形引擎N
33B1
33B2
33BN
33G
33F
纹理引擎1
纹理引擎2
纹理引擎N
高速缓存
纹理高速缓存
33
33C
第一总线转换器
33D1
33D2
33D3
33DM
象素引擎1
象素引擎2
象素引擎3
象素引擎M
33E
总线转换器
存贮体1
存贮体2
存贮体L
18

图 2

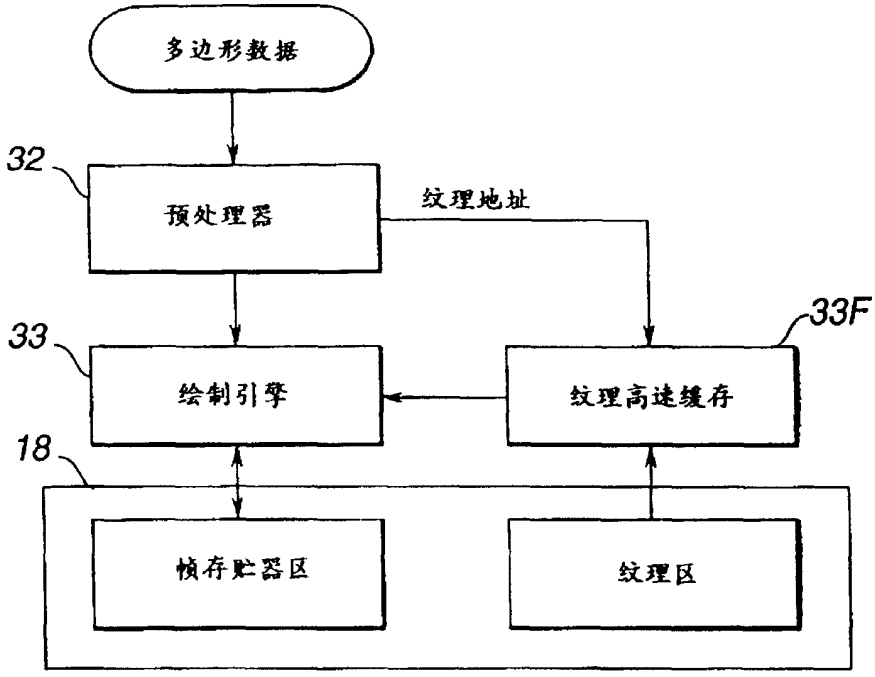


图 3

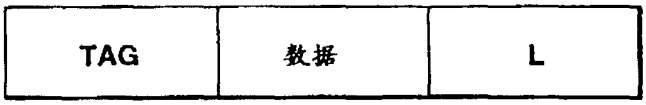


图 4

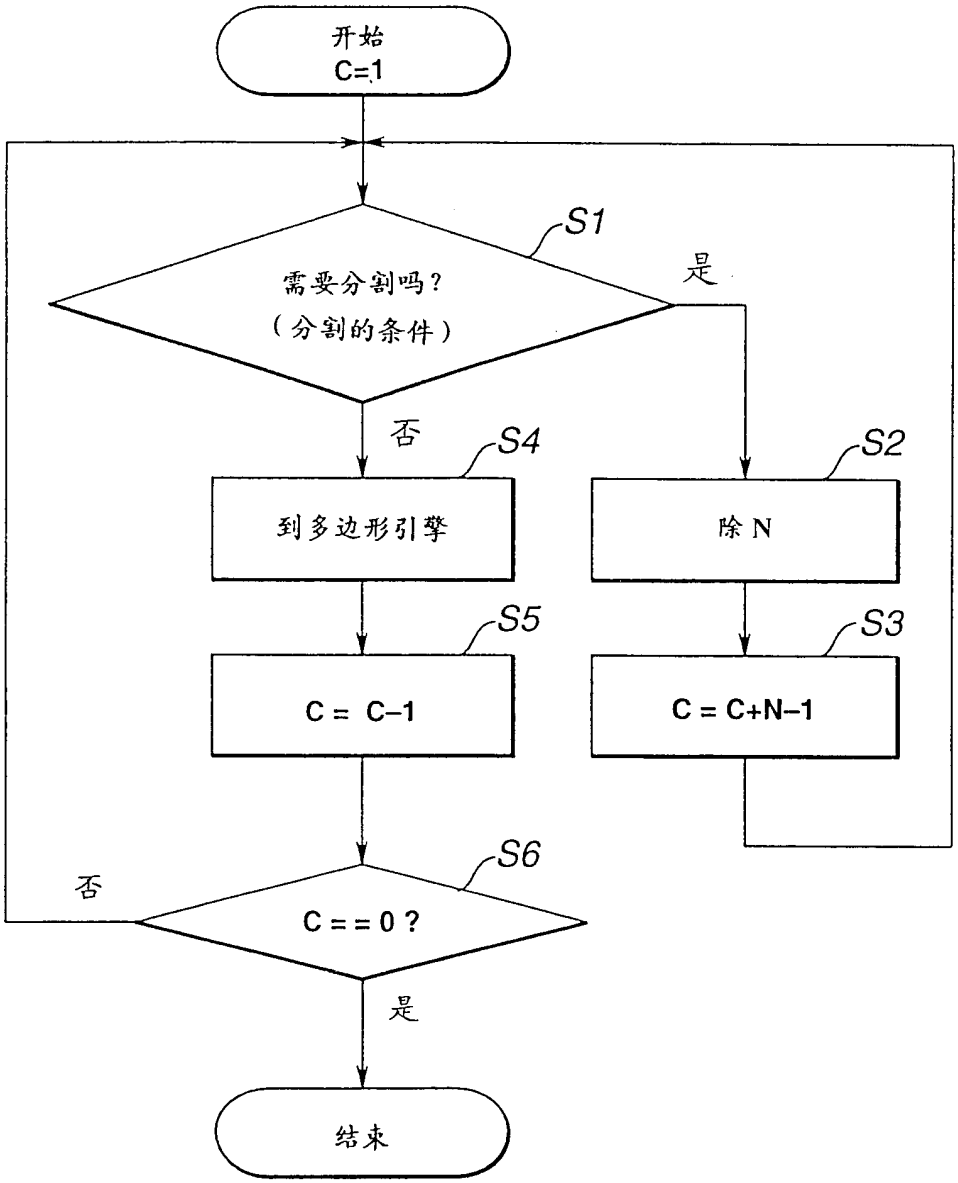


图 5

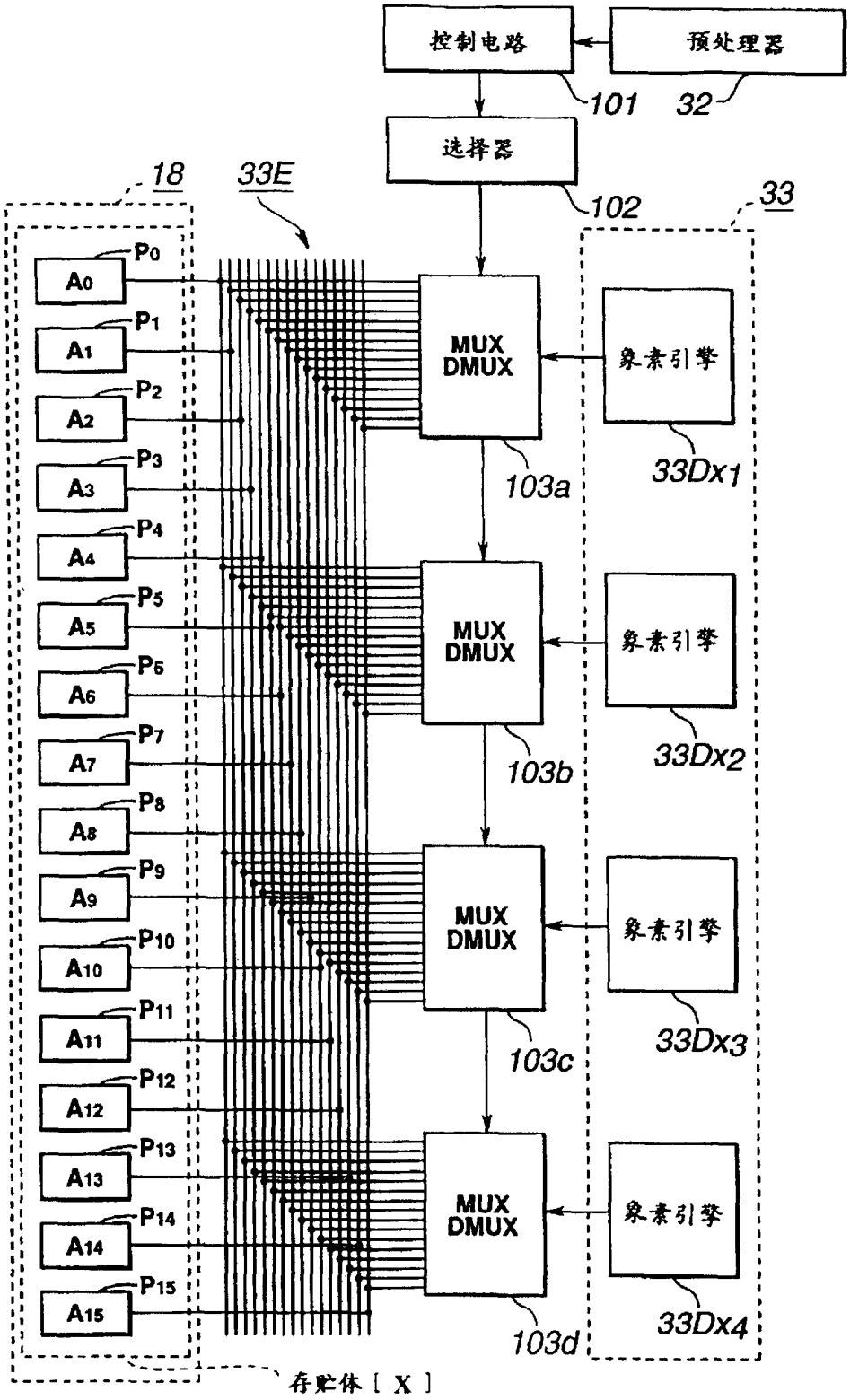

控制电路
预处理器
101
32
选择器
102
18
33E
33
P0
A0
P1
A1
P2
A2
P3
A3
P4
A4
P5
A5
P6
A6
P7
A7
P8
A8
P9
A9
P10
A10
P11
A11
P12
A12
P13
A13
P14
A14
P15
A15
MUX DMUX
MUX DMUX
MUX DMUX
MUX DMUX
象素引擎
象素引擎
象素引擎
象素引擎
103a
103b
103c
103d
33Dx1
33Dx2
33Dx3
33Dx4
存贮体 [X]

图 6

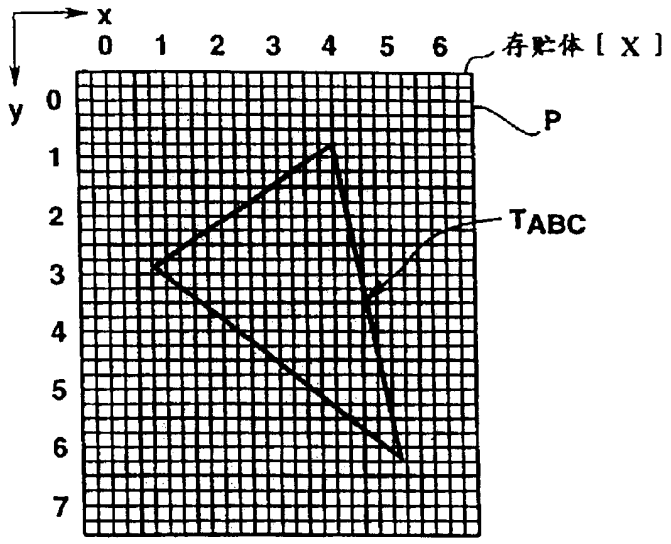

x
0 1 2 3 4 5 6
存贮体[X]
y
0
1
2
3
4
5
6
7
P
TABC

图 7

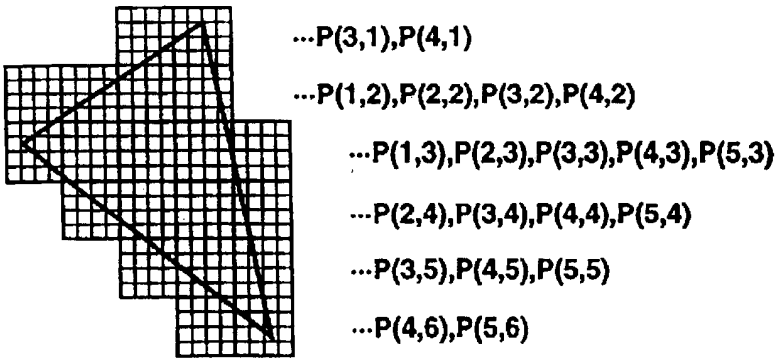

…P(3,1),P(4,1)
…P(1,2),P(2,2),P(3,2),P(4,2)
…P(1,3),P(2,3),P(3,3),P(4,3),P(5,3)
…P(2,4),P(3,4),P(4,4),P(5,4)
…P(3,5),P(4,5),P(5,5)
…P(4,6),P(5,6)

图 8

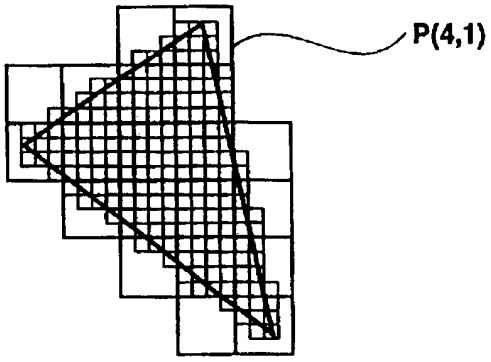

P(4,1)

图 9

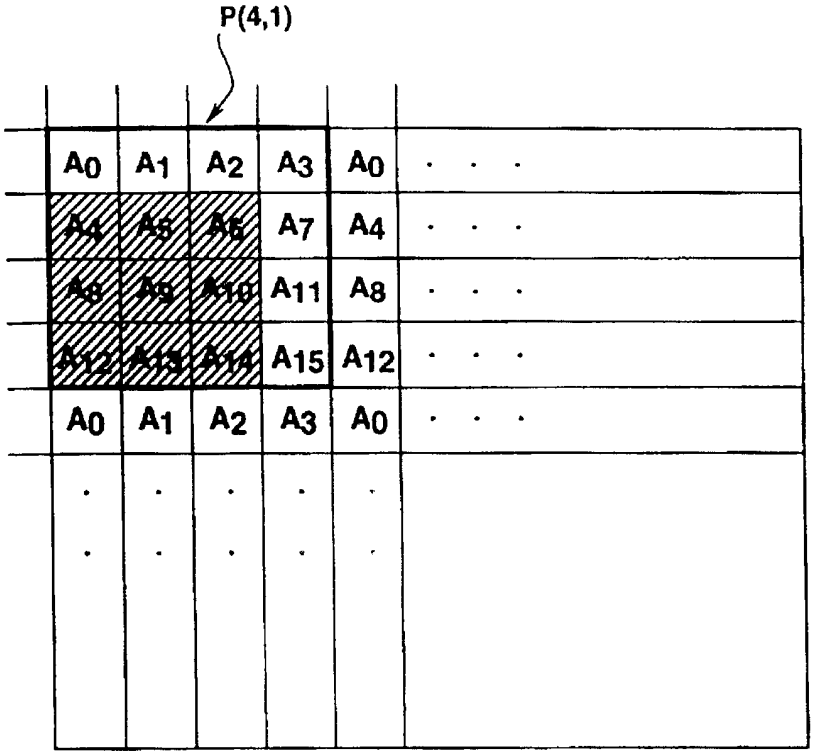

P(4,1)
A0 A1 A2 A3 A0
A4 A5 A6 A7 A4
A8 A9 A10 A11 A8
A12 A13 A14 A15 A12
A0 A1 A2 A3 A0

图 10

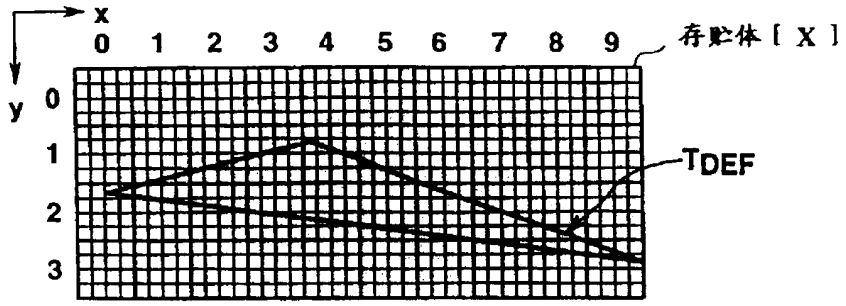

x
0 1 2 3 4 5 6 7 8 9
存贮体[X]
y
0
1
2
3
TDEF

图 11

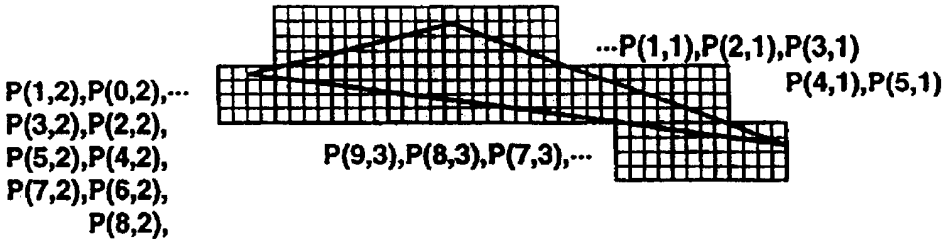

P(1,2),P(0,2),···
P(3,2),P(2,2),
P(5,2),P(4,2),
P(7,2),P(6,2),
P(8,2),
···P(1,1),P(2,1),P(3,1)
P(4,1),P(5,1)
P(9,3),P(8,3),P(7,3),···

图 12

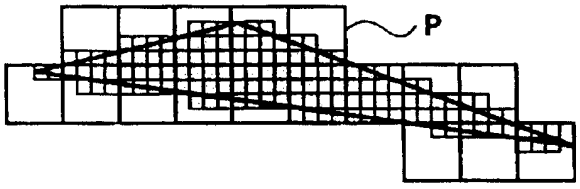

P

图 13

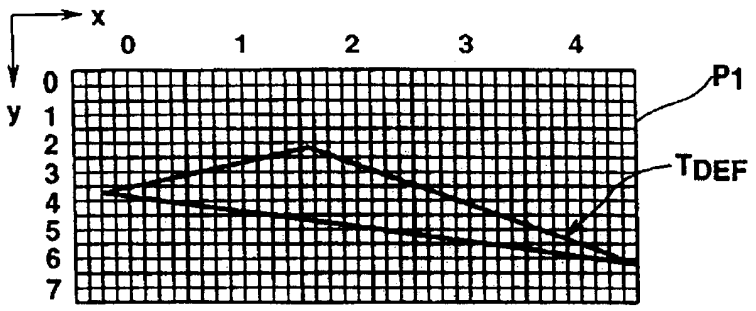

x
0
1
2
3
4
y
0
1
2
3
4
5
6
7
P1
TDEF

图 14

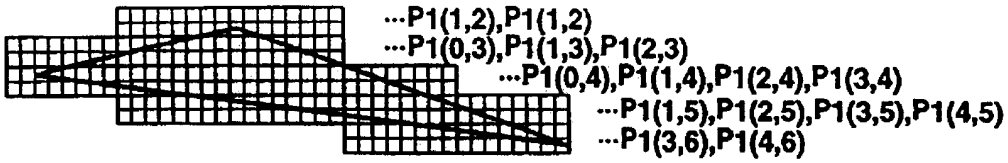

…P1(1,2),P1(1,2)
…P1(0,3),P1(1,3),P1(2,3)
…P1(0,4),P1(1,4),P1(2,4),P1(3,4)
…P1(1,5),P1(2,5),P1(3,5),P1(4,5)
…P1(3,6),P1(4,6)

图 15

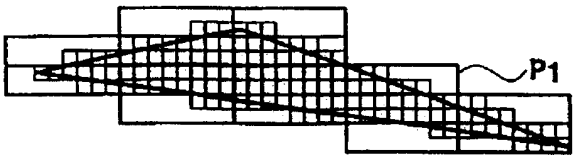

P1

图 16

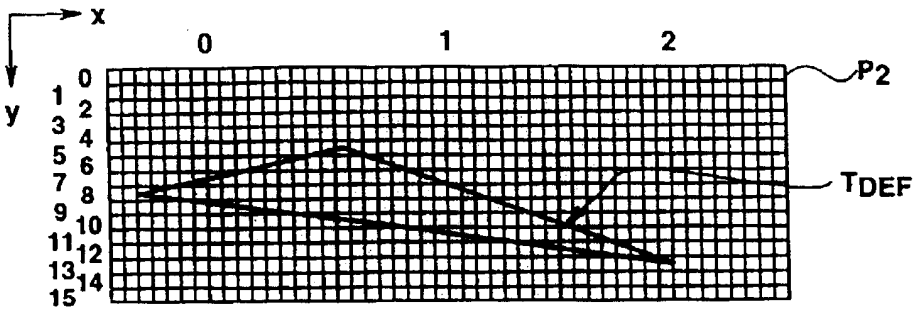

x
y
0
1
2
0
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15
P2
TDEF

图 17

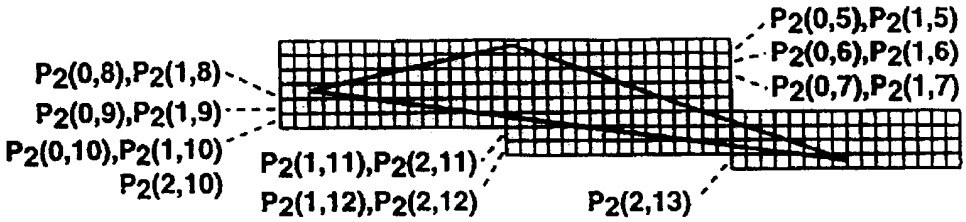

P2(0,5),P2(1,5)
P2(0,6),P2(1,6)
P2(0,7),P2(1,7)
P2(0,8),P2(1,8)
P2(0,9),P2(1,9)
P2(0,10),P2(1,10)
P2(2,10)
P2(1,11),P2(2,11)
P2(1,12),P2(2,12)
P2(2,13)

图 18

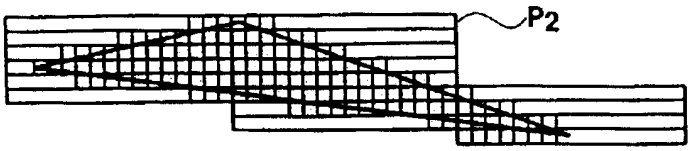

P2

图 19

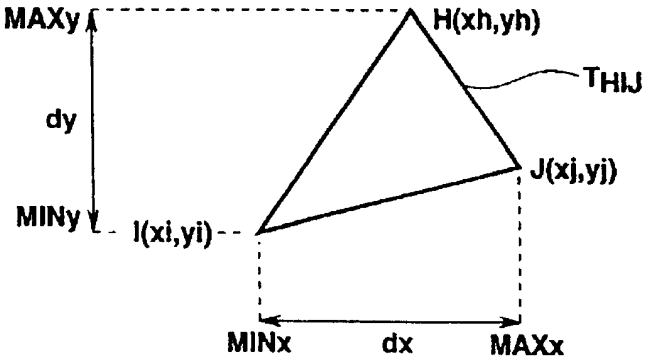


图 20

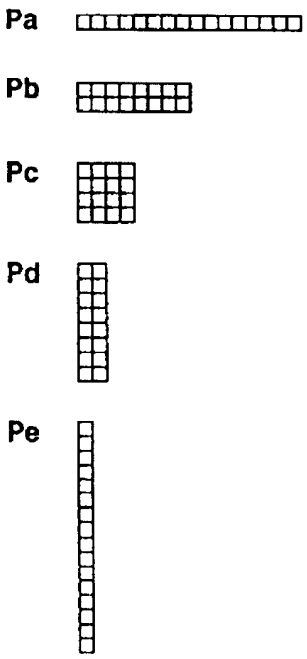


图 21

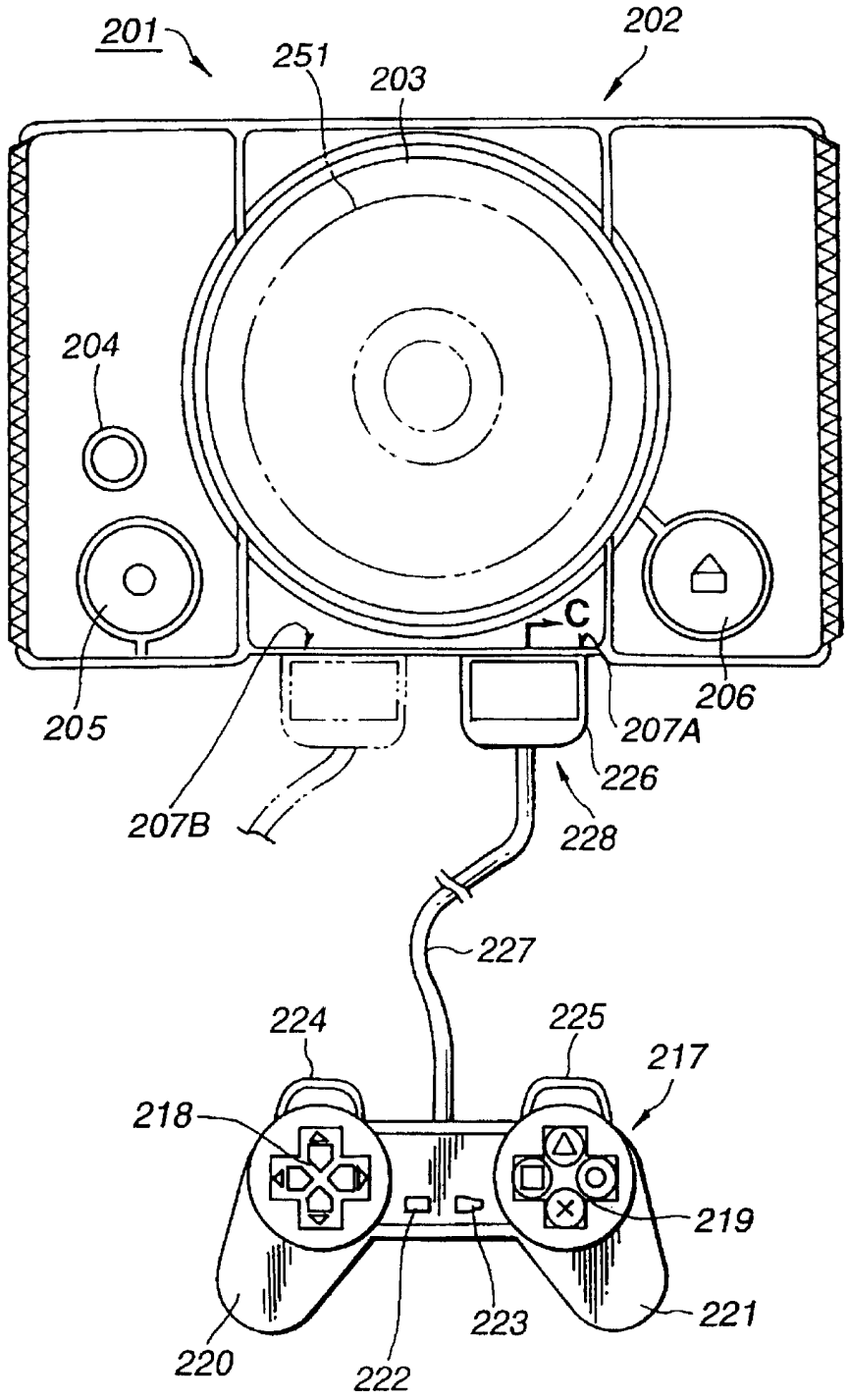

201
251
203
202
204
205
C
206
207A
207B
226
228
227
224
225
217
218
219
221
220
222
223

图 22

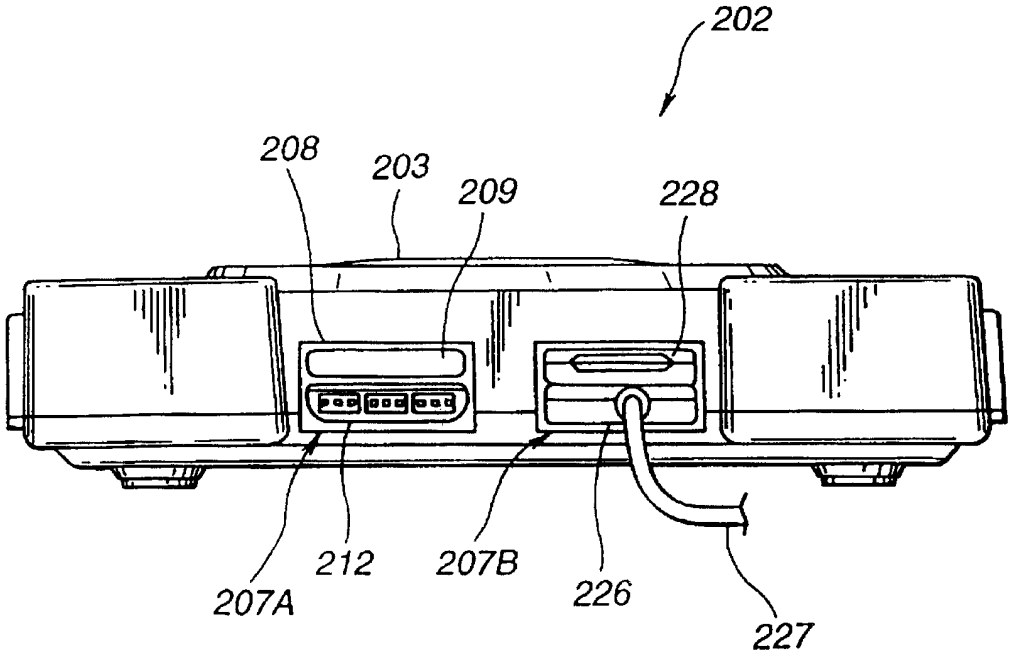


图 23

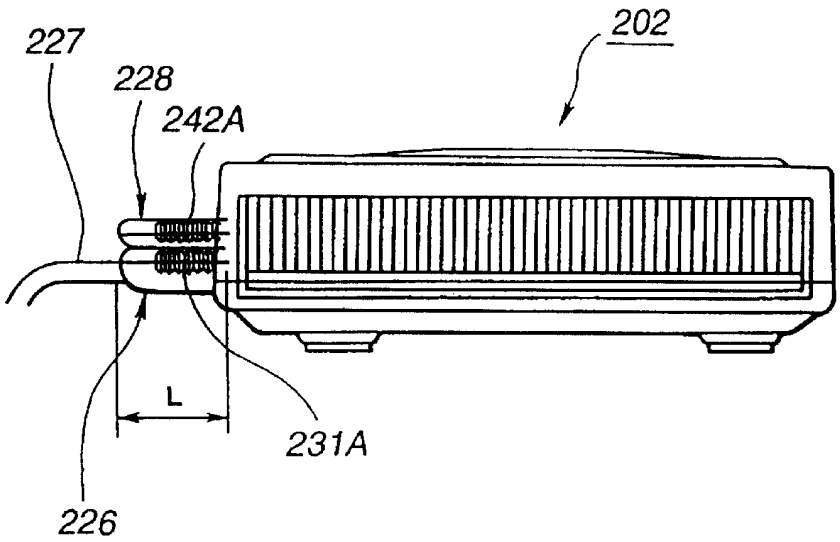


图 24

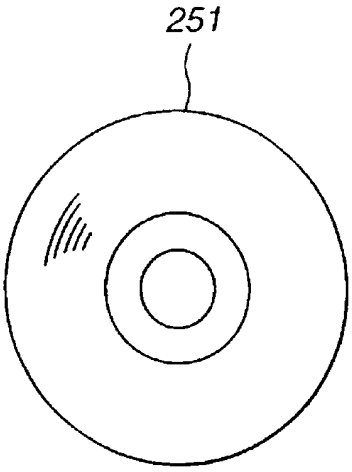


图 25