(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 101202091 B
(45)授权公告日 2012.11.28

(21)申请号 200710167989.5
(22)申请日 2004.07.09
(30)优先权数据
60/486,844 2003.07.11 US
(62)分案原申请数据
200480026201.3 2004.07.09
(73)专利权人 松下电器产业株式会社
地址 日本大阪府
(72)发明人 约瑟夫·麦克罗森 冈田智之
(74)专利代理机构 永新专利商标代理有限公司 72002
代理人 王英
(51)Int.Cl.
G11B 27/034(2006.01)
G11B 27/10(2006.01)
H04N 9/804(2006.01)
H04N 9/82(2006.01)

(56)对比文件
WO 2004036578 A1,2004.04.29,

审查员 马京京

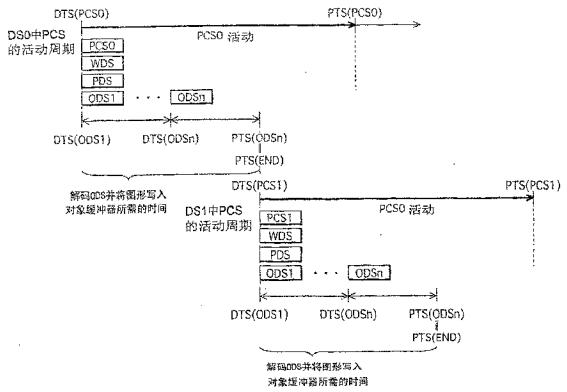
权利要求书 2 页 说明书 27 页 附图 38 页

(54)发明名称

记录方法、再现装置和方法

(57)摘要

在BD-ROM上记录通过复用视频流和图形流而生成的AV剪辑。所述图形流包括多个DS(显示集合),其每组段均用于图形显示。属于DS的ODS(对象定义段)包括参考ODS以及非参考ODS。所述非参考ODS是没有通过所述显示集合中包括的PCS(呈现合成段)而参考的ODS。END段设置为紧跟在所述显示集合中所述参考ODS和非参考ODS的序列中最后一个ODS之后。



CN 101202091 B

1. 一种再现装置，用于从记录介质读取数字流并再现在该数字流中复用的视频流和图形流，所述再现装置包括：

视频解码器 (5)，用于解码所述视频流以生成运动图像；以及

图形解码器 (12)，用于解码所述图形流以生成图形对象，其中：

所述图形流包括多个显示集合，每个所述显示集合用于图形显示；

所述显示集合包括呈现合成段 PCS、用于限定图形对象的对象定义段 ODS 的序列以及 END 段；

所述 PCS 包括时间信息，该时间信息用于在所述视频流的再现时间轴上指定所述显示集合中 PCS 的活动周期；

每个段包含在包中；

所述时间信息包括写在所述包中的解码时戳以及呈现时戳；

所述 ODS 的序列包括通过所述显示集合中 PCS 而参考的参考 ODS 以及没有通过所述显示集合中 PCS 而参考的非参考 ODS，所述显示集合中的 ODS 是顺序地设置的；以及

其中所述图形解码器还包括流图形处理器 (14)，其对设置在所述显示集合中 PCS 和 END 段之间的所述参考 ODS 和非参考 ODS 依次进行解码，并存储解码获得的图形对象到所述图形解码器的缓冲器中。

2. 如权利要求 1 所述的再现装置，其中：

在与所述 END 段相关的时戳显示的时间，完成通过所述图形解码器对所述参考 ODS 和非参考 ODS 的解码。

3. 一种记录方法，包括下面的步骤：

创建包括多个显示集合的图形流，每个所述显示集合都用于图形显示；

将所创建的图形流与视频流一同记录到 BD-ROM 的螺旋轨道上，其中每个所述显示集合包括如下功能段：呈现合成段 PCS、用于限定图形对象的对象定义段 ODS 以及 END 段，其中所述 PCS 具有时间信息，所述时间信息用于在所述视频流的再现时间轴上指定所述显示集合中 PCS 的活动周期；每个段包含在包中；所述时间信息包括写在包中的解码时戳以及呈现时戳；所述显示集合包括 ODS 序列，该 ODS 序列包括通过所述显示集合中 PCS 而参考的参考 ODS 以及没有通过所述显示集合中 PCS 而参考的非参考 ODS，

其中创建图形流的步骤包括：

将所述参考 ODS 和非参考 ODS 设置在所述显示集合中，所述显示集合中的 ODS 是顺序地设置的，以便以所述设置的顺序而依次解码；

紧跟在所述参考 ODS 和非参考 ODS 的序列中最后一个 ODS 之后设置所述 END 段；以及

在记录所述图形流之前，执行如下步骤：

将所述功能段转换为传输流数据包，以及；

向每个对应于所述功能段的传输流数据包和对应于所述视频流的传输流数据包分配到达时戳 ATS；其中

通过将带有所分配的 ATS 的传输流数据包记录到所述 BD-ROM 的所述螺旋轨道上来记录所述图形流。

4. 一种再现方法，用于在计算机上实现从记录介质读取数字流并再现所述数字流中复用的视频流和图形流的处理，所述方法包括下面的步骤：

解码所述视频流以生成运动图像；以及

解码所述图形流以生成图形对象，其中：

所述图形流包括多个显示集合，每个所述显示集合用于图形显示；

所述显示集合包括呈现合成段 PCS、用于限定图形对象的对象定义段 ODS 以及 END 段；

所述 PCS 包括时间信息，该时间信息用于在所述视频流的再现时间轴上指定所述显示集合中 PCS 的活动周期；

每个段包含在包中；

所述时间信息包括写在所述包中的解码时戳以及呈现时戳；

所述显示集合包括 ODS 序列，该 ODS 序列包括通过所述显示集合中 PCS 而参考的参考 ODS 以及没有通过所述显示集合中 PCS 而参考的非参考 ODS；

所述 ODS 顺序地设置在所述显示集合中；以及

在处理所述显示集合的时候，对所述图形流解码的步骤解码设置在所述显示集合中 PCS 和 END 段之间的参考 ODS 和非参考 ODS，并存储解码获得的图形对象到所述计算机的缓冲器中。

记录方法、再现装置和方法

[0001] 本申请是2004年7月9日提交的、申请号为200480026201.3、题为“记录介质、记录方法、再现装置和方法以及计算机可读程序”的专利申请的分案申请。

发明领域

[0002] 本发明涉及诸如BD-ROM和再现装置的记录介质，并特别涉及通过再现数字流来加字幕的技术，其中通过复用视频流和图形流生成该数字流。

技术背景

[0003] 通过渲染图形流显示的字幕对于不同语言地域的人享受外语电影来说是重要的手段。此类图形流与代表运动图像的视频流复用，并记录在记录介质上。图形流包括多个显示集合，每个显示集合由显示控制信息和图形数据组成。每个显示集合用于显示电影再现中的单独字幕。从记录介质中读取显示集合，并一个个处理，作为运动图像进行的再现，以与运动图像一同显示字幕。

[0004] 这里，如果只在前面紧邻的一个显示集合的处理完成之后处理每个显示集合，就出现处理延迟。特别当每个显示集合具有例如1920×1080的高分辨率时，出现明显的处理延迟。因此，当图形流包含多个显示集合时，就出现需要并行处理显示集合。

发明内容

[0005] 一个显示集合可以不仅携带其自身呈现合成段所参考的对象定义段，还可以携带后面显示集合的呈现合成段所参考的对象定义段。在传送不是呈现合成段参考的对象定义段的情况下，在将多个对象定义段加载到再现装置上的时候，属于一个显示集合的对象定义段和属于后面显示集合的对象定义段之间的边界不明确。

[0006] 本发明在于提供一种记录介质，其可以甚至在一个显示集合不仅包括其自身呈现合成段所参考的对象定义段，还包括后面显示集合的呈现合成段所参考的对象定义段的时候，说明对象定义段属于哪个显示集合。

[0007] 上述目的可以通过一种记录方法来实现。所述记录方法包括下面的步骤：创建包括多个显示集合的图形流，每个所述显示集合都用于图形显示；将所产生的数字流记录到记录介质上，其中每个所述显示集合包括如下功能段：呈现合成段PCS、用于限定图形对象的对象定义段ODS以及END段，其中所述PCS包括时间信息，所述时间信息用于在所述视频流的再现时间轴上指定所述显示集合中PCS的活动周期；每个段包含在包中；所述时间信息包括写在包中的解码时戳以及呈现时戳；所述显示集合包括ODS序列，该ODS序列包括通过所述显示集合中PCS而参考的参考ODS以及没有通过所述显示集合中PCS而参考的非参考ODS，其中创建图形流的步骤包括：将所述参考ODS和非参考ODS设置在所述显示集合中，所述显示集合中的ODS是顺序地设置的，以便以所述设置的顺序而依次解码；紧跟在所述参考ODS和非参考ODS的序列中最后一个ODS之后设置所述END段；将所述功能段转换为传输流数据包，并将所述传输流数据包与所述视频流一同记录在所述记录介质上，其中

所述视频流同样被转换为传输流数据包；向每个对应于所述功能段的传输流数据包和对应于所述视频流的传输流数据包给定称为到达时戳 ATS 的时戳；以及复用所述视频流和所述图形流以生成数字流。

[0008] 根据此结构，END 段设置在显示集合中属于该显示集合的对象定义段之后。通过参考 END 段，再现装置可以探测属于显示集合的对象定义段的传送结束。因此，即使在一个显示集合包括参考对象定义段和非参考对象定义段的时候，在处理显示集合时，很容易就可以识别需要从哪个点到哪个点执行解码。这使再现装置使用非参考对象定义段执行高速渲染。

附图说明

[0009] 图 1 是本发明所涉及的记录介质的示例性应用；

[0010] 图 2 示出了图 1 所示的 BD-ROM 的结构；

[0011] 图 3 示出了 AV 剪辑的结构；

[0012] 图 4A 示出了呈现图形流的结构；

[0013] 图 4B 示出了包含功能段的 PES 数据包；

[0014] 图 5 示出了由多种功能段构成的逻辑结构；

[0015] 图 6 示出了字幕显示位置和时元之间的关系；

[0016] 图 7A 示出了 ODS 的数据结构；

[0017] 图 7B 示出了 PDS 的数据结构；

[0018] 图 8A 示出了 WDS 的数据结构；

[0019] 图 8B 示出了 PCS 的数据结构；

[0020] 图 9 示出了用于显示字幕的 DS 的示例性描述；

[0021] 图 10 示出了 DS1 中的 PCS 和 WDS 的示例性描述；

[0022] 图 11 示出了 DS2 中的 PCS 的示例性描述；

[0023] 图 12 示出了 DS3 中的 PCS 的示例性描述；

[0024] 图 13 示出了当执行如图 10 至 12 所示的图形更新时对象缓冲器中的存储空间；

[0025] 图 14 示出了用于计算 DECODEDURATION 的示例性算法；

[0026] 图 15 是图 14 所示算法的流程图；

[0027] 图 16A 和 16B 是图 14 所示算法的流程图；

[0028] 图 17A 示出了一个窗口中存在一个图形对象的情形；

[0029] 图 17B 和 17C 是图 14 所示算法中所用的参数的时序图；

[0030] 图 18A 示出了一个窗口中存在两个图形对象的情形；

[0031] 图 18B 和 18C 是图 14 所示算法中所用的参数的时序图；

[0032] 图 19A 示出了两个图形对象分别存在于两个窗口中的情形；

[0033] 图 19B 是当解码时间 (2) 大于清除时间 (1) 与写入时间 (31) 之和时的时序图；

[0034] 图 19C 是当清除时间 (1) 与写入时间 (31) 之和大于解码时间 (2) 时的时序图；

[0035] 图 20 示出了一个 DS 的处理内容；

[0036] 图 21 示出了如何在一个流水线解码器模型中并行处理两个 DS；

[0037] 图 22 示出了在三个 DS 中重叠 PCS 的活动周期的实例；

[0038] 图 23 示出了每个 DS 中功能段的时戳的设定；

[0039] 图 24 示出了每个 DS 中 PCS 的时戳；

[0040] 图 25A 示出了两个 DS 中 PCS 的活动周期重叠的情况；

[0041] 图 25B 示出了两个 DS 中 PCS 的活动周期不重叠的情况；

[0042] 图 26 示出了指明传送完成的 END 段；

[0043] 图 27A 至 27C 示出了活动周期重叠和 object_id 分配之间的关系；

[0044] 图 28 示出了本发明的实施例所涉及的再现装置的内部构造；

[0045] 图 29 示出了图 28 中所示的图形平面、编码数据缓冲器和对象缓冲器的传输速率 Rx、Rc、Rd 和尺寸；

[0046] 图 30 是再现装置中的流水线处理的时序图；

[0047] 图 31 是当 ODS 解码在清除图形平面之前结束情况下的流水线处理的时序图；

[0048] 图 32 是合成缓冲器、对象缓冲器、编码数据缓冲器和图形平面的占用率变化的时序图；

[0049] 图 33 是加载功能段的操作的流程图；

[0050] 图 34 示出了执行跳节操作的情形；

[0051] 图 35 示出了当执行如图 34 所示的跳节操作时向编码数据缓冲器中加载 DS10 的情形；

[0052] 图 36 示出了当执行普通再现时的情形；

[0053] 图 37 示出了当执行如图 36 所示的普通再现时向编码数据缓冲器中加载 DS1 和 DS20 的情形；

[0054] 图 38 是图 28 所示的图形控制器的操作流程图；

[0055] 图 39 是图形控制器的操作流程图；

[0056] 图 40 是图形控制器的操作流程图；

[0057] 图 41 示出了 BD-ROM 的制造步骤。

具体实施方式

[0058] （第一个实施例）

[0059] 下面描述本发明第一个实施例所涉及的记录介质。下面首先介绍该记录介质的使用。图 1 示出了该记录介质的示例性应用。在图中，该记录介质为 BD-ROM 100。BD-ROM 100 用于在家庭影院系统中提供电影，该家庭影院系统包括再现装置 200、电视 300 和遥控 400。

[0060] 接着介绍记录介质的制造。通过改进 BD-ROM 的应用层，可以实现该记录介质。图 2 示出了 BD-ROM 100 的示例性结构。

[0061] 在图中，第四级示出了 BD-ROM 100，第三级示出了 BD-ROM100 上的轨道。图中所示的轨道向外延伸成直线，但实际上，轨道从 BD-ROM 100 的中心向外螺旋延伸。轨道包括引入区、容量区和引出区。容量区的层模型是物理层、文件系统层和应用层。第一级以目录结构的形式示出了 BD-ROM 100 的应用层格式（应用格式）。如图所示，BD-ROM 100 在根目录下有 BDMV 目录。BDMV 目录包含一个用于存储 AV 剪辑的文件 (XXX.M2TS)、一个用于存储 AV 剪辑的管理信息的文件 (XXX.CLPI) 和一个用于定义 AV 剪辑的逻辑播放路径（播放列表）

的文件 (YYY.MPLS)。通过生成这样的应用格式,可以实现 BD-ROM 100。如果上述各文件类型都有一个以上的文件,则在 BDMV 目录下设置三个名为 STREAM、CLIPNF 和 PLAYLIST 的目录,分别用来存储与 XXX.M2TS 类型相同的文件、与 XXX.CLPI 类型相同的文件以及与 YYY.MPLS 类型相同的文件。

[0062] 下面说明该应用格式中的 AV 剪辑 (XXX.M2TS)。

[0063] AV 剪辑 (XXX.M2TS) 是 MPEG-TS(传输流) 格式的数字流,它是通过复用一个视频流、至少一个音频流和一个呈现图形流而获得的。视频流代表电影的运动图像,音频流代表电影的音频,呈现图形流代表电影的字幕。图 3 示出了 AV 剪辑 (XXX.M2TS) 的结构。

[0064] 在图中,中间级示出了 AV 剪辑。可以如下创建该 AV 剪辑。将上方第一级中由多个视频帧 (图像 pj1、pj2、pj3……) 构成的视频流和由多个音频帧构成的音频流转换成上方第二级中的 PES 数据包,然后再转换成上方第三级中的 TS 数据包。同样,将下方第一级中的呈现图形流转换成下方第二级中的 PES 数据包,然后再转换成下方第三级中的 TS 数据包。将视频、音频和呈现图形流的这些 TS 数据包进行复用,从而形成 AV 剪辑。

[0065] 在图 3 所示的例子中,只有一个呈现图形流被复用到 AV 剪辑中。如果 BD-ROM 100 支持多种语言,则可以将每种语言的呈现图形流都复用到 AV 剪辑中。以上述方式生成的 AV 剪辑按照与计算机文件相同的方式被分成多个内容,并存储到 BD-ROM 100 上。

[0066] 下面说明呈现图形流。图 4A 示出了呈现图形流的结构。在图中,第一级显示的是构成该 AV 剪辑的 TS 数据包序列。第二级显示的是构成呈现图形流的 PES 数据包序列。该 PES 数据包序列是通过将第一级的 TS 数据包序列中具有预定 PID 的 TS 数据包的有效负载进行连接而形成的。

[0067] 第三级示出了呈现图形流的结构。呈现图形流是由多个功能段构成的,这些功能段包括 PCS(呈现合成段)、WDS(窗口定义段)、PDS(调色板定义段)、ODS(对象定义段) 和 END(显示集合段的末尾)。在这些功能段中,PCS 是画面合成段,WDS、PDS 和 ODS 是定义段。一个功能段对应于一个 PES 数据包或多个 PES 数据包。也就是说,一个功能段被转换成一个 PES 数据包并记录到 BD-ROM 100 上,或者,被分成多个片断,再转换成 PES 数据包,并记录到 BD-ROM 100 上。

[0068] 图 4B 示出了包含功能段的 PES 数据包。如图所示,每个 PES 数据包由包头和有效负载构成。有效负载携带功能段,包头携带与该功能段相关联的 DTS 和 PTS。后面,将包含一个功能段的 PES 数据包的包头中的 DTS 和 PTS 视为该功能段的 DTS 和 PTS。

[0069] 这些多种类型的功能段形成如图 5 所示的逻辑结构。在图中,第三级示出了功能段,第二级示出了 DS(显示集合),第一级示出了时元。

[0070] 第二级中的 DS 是呈现图形流中的一组功能段,用于创建一个图形画面。虚线 hk2 表明第三级中的功能段属于哪个 DS。从图中可以看出,功能段序列 PCS-WDS-PDS-ODS-END 组成一个 DS。再现装置 200 从 BD-ROM 100 中读取构成该 DS 的这些功能段,以产生一个图形画面。

[0071] 第一级中的时元指的是 AV 剪辑的再现时间轴上的连续存储器管理的一个时间单元以及分配给该时间单元的一组数据。这里所说的存储器包括 :图形平面,用于存储一个图形画面 ;对象缓冲器,用于存储解压缩的图形数据。连续存储器管理意味着 :在该时元内,不刷新图形平面和对象缓冲器,只在图形平面的预定矩形区域中执行图形的删除和呈现

（刷新意味着清除整个图形平面或整个对象缓冲器）。在该时元内，该矩形区域的大小和位置是固定的。只要在图形平面的该固定矩形区域中执行图形的删除和呈现，就可以保证视频和图形的同步。换言之，时元是 AV 剪辑的再现时间轴上的一个时间单位，其中，可以保证视频和图形的同步。为了改变图形平面中的图形删除 / 呈现区域，需要定义再现时间轴上的变化点以及从该点向前设置一个新的时元。在这两个时元之间的边界中，视频和图形的同步无法得到保证。

[0072] 对于字幕，时元是再现时间轴上的一个时间段，其中，字幕出现在平面上固定矩形区域内。图 6 示出了字幕显示位置和时元之间的关系。在图中，字幕显示位置的改变取决于图像的图案。更详细地说，三个字幕“Actually”、“I lied to you”和“Sorry”位于屏幕的底部，而两个字幕“三年已经过去了”和“从那以后。”位于屏幕的顶部。因此，为了增加可视性，字幕显示位置从一个边缘部分变化到另一边缘部分。在这种情况下，在 AV 剪辑的再现时间轴上，在屏幕底部显示字幕的时间段是时元 1，在屏幕顶部显示字幕的时间段是时元 2。这两个时元各有自己的字幕呈现区域。在时元 1 中，字幕呈现区域是 Window1，对应于屏幕的底部边缘部分。在时元 2 中，字幕呈现区域是 Window2，对应于屏幕的顶部边缘部分。在时元 1 和时元 2 中，对象缓冲器和图形平面的存储器管理是连续的，所以，字幕在该屏幕的相应边缘部分中无缝地显示。对时元的解释到此结束。

[0073] 下面解释 DS。

[0074] 在图 5 中，虚线 hk1 表示第二级中的 DS 属于哪个时元。如图所示，DS 序列 (Epoch Start DS、Acquisition Point DS 和 Normal Case DS) 构成第一级中的一个时元。在这里，Epoch Start（时元开始）、AcquisitionPoint（采集点）和 Normal Case（正常情况）是 DS 的类型。在图 5 中，Acquisition Point DS 位于 Normal Case DS 之前，但它们的排列顺序也可以反过来。

[0075] Epoch Start DS 提供显示效果“新显示”，并表示一个新时元的开始。Epoch Start DS 包含下一画面合成所需的全部功能段。提供 EpochStart DS 的位置是跳节操作的目的地，如电影中一章的开始。

[0076] Acquisition Point DS 提供显示效果“显示刷新”，并且与前面的 Epoch Start DS 相同。Acquisition Point DS 不是时元的开始，但却包含下一画面合成所需的全部功能段。因此，当从 Acquisition Point DS 开始再现时，能够可靠地显示图形。也就是说，通过 Acquisition PointDS，可以从时元的中间点进行画面合成。

[0077] 提供 Acquisition Point DS 的位置可以是跳节操作的目的地，例如可以通过时间搜索来指定的位置。时间搜索是一种操作，用于对与用户的时间输入（单位为分 / 秒）相对应的再现点进行定位。时间输入的单位较大，如 10 分或 10 秒。因此，提供 Acquisition Point DS 的位置可以通过单位是 10 分或 10 秒的时间搜索来指定。通过在可由时间搜索指定的位置中提供 Acquisition Point DS，当执行时间搜索时，可以流畅地再现图形流。

[0078] Normal Case DS 提供显示效果“显示更新”，并且仅包含与前一画面合成之间的差值。例如，如果 DSv 与它之前的 Dsu 具有相同的字幕，但却具有不同的画面合成，则 DSv 是仅包含一个 PCS 和一个 END 的 Normal Case DS。这样，就不必在 DSs 中提供覆盖的 ODS，从而可以减少 BD-ROM 100 上所存储的数据量。由于 Normal Case DS 只包含差值，所以，不能单用 Normal Case DS 来显示图形。

[0079] 下面说明 ODS、WDS 和 PDS（定义段）。

[0080] ODS 是用于定义图形对象的功能段。BD-ROM 上记录的 AV 剪辑的图像质量与高清晰度电视一样高。因此，以 1920×1080 的高分辨率来设置图形对象。由于该高分辨率，可以在 BD-ROM 上逼真地再现影院屏幕风格的字幕，即，雅致的手写体字幕。

[0081] 图形对象由多个游程数据构成。游程数据表示使用像素编码的像素串，是像素值和该像素值的连续长度。像素编码有 8 个比特，表示从 1 到 255 的值。通过使用该像素编码，游程数据设置全部颜色（16777216 色）中任意 256 个像素颜色。应当注意的是，为了将图形对象显示为字幕，需要将字符串放置在透明色的背景上。

[0082] ODS 定义根据图 7A 所示的数据结构的图形对象。如图所示，ODS 包括 :segment_type 字段，表示段类型“ODS”;segment_length 字段，表示 ODS 的数据长度 ;object_id 字段，标识时元中的图形对象 ;object_version_number 字段，表示时元中的 ODS 的版本 ; last_in_sequence_flag 字段 ;object_data_fragment 字段，承载与图形对象的部分或全部相对应的连续字节序列。

[0083] 更详细地讲，object_id 是一个标识符，用于标识图形对象，以及当对 ODS 进行解码和将图形对象缓冲到对象缓冲器中时该图形对象在对象缓冲器中所占用的存储区域。因此，当对象缓冲器中有一个或多个图形对象时，用 object_id 字段值标识对象缓冲器中的各存储区域。假设将一个 object_id 分配给两个或更多个 ODS。在这种情况下，在将与一个 ODS 相对应的图形对象存储到对象缓冲器之后，该图形对象会被与随后具有相同 object_id 的 ODS 相对应的图形对象覆盖。这样的更新意在防止对象缓冲器中出现很多小的自由空间和防止该对象缓冲器中的图形对象的分散。当显示图形时，对象缓冲器中的图形对象不断地被传输到图形平面。因此，如果对象缓冲器中存在很多小的自由空间或者一个图形对象分散在对象缓冲器中，则读取图形对象的开销导致从对象缓冲器到图形平面的传输效率降低。传输效率的这种降低会影响图形和视频的同步显示。为了防止这种问题，对象缓冲器中的现有图形对象被具有相同 object_id 的一个新图形对象覆盖。

[0084] 这里，覆盖现有图形对象的新图形对象需要在尺寸上等于现有图形对象，也就是说，新图形对象既不能小于现有图形对象，也不能大于现有图形对象。因此，在创作时，创作者需要使这些图形对象尺寸相等。该尺寸约束条件，即具有相同 object_id 的图形对象应该宽度和高度相等，只适用于一个时元内。具有相同 object_id 的图形对象不必尺寸相等，如果它们属于不同时元的话。

[0085] 接下来解释 last_in_sequence_flag 字段和 object_data_fragment 字段。由于 PES 数据包的有效负载的约束条件，构成一个字幕的解压缩图形可能无法包含在一个 ODS 中。在这种情况下，将图形分割成多个片断，将其中一个片断承载在 object_data_fragment 中。当将图形对象跨过多个 ODS 进行存储时，除最后片断之外的每个片断都具有相同的尺寸。也就是说，最后的片断小于或等于前面片断的尺寸。承载图形对象的这些片断的 ODS 顺序地出现在 DS 中。last_in_sequence_flag 字段表示图形对象的结束。尽管上述 ODS 数据结构所基于的方法将片断不留空隙地存储在连续 PES 数据包中，但是，也可以通过在 PES 数据包中留出一些空隙的方式将片断存储在 PES 数据包中。

[0086] PDS 是功能段，定义用于颜色转换的调色板。调色板是表示 1 至 255 的像素代码和像素值的组合的数据。这里所指的像素值是由红色差值分量 (Cr 值）、蓝色差值分量 (Cb

值）、亮度分量（Y 值）和透明色（T 值）构成的。将每个游程数据的像素代码代入调色板上的像素值中，从而产生颜色。图 7B 示出了 PDS 的数据结构。如图所示，PDS 包括 ：segment_type 字段，表示段类型“PDS”；segment_length 字段，表示 PDS 的数据长度 ；palette_id 字段，唯一地标识该调色板 ；palette_version_number 字段，表示该时元内的 PDS 的版本 ；以及 palette_entry 字段，承载每个条目的信息。palette_entry 字段表示每个条目的红色差值分量（Cr 值）、蓝色差值分量（Cb 值）和透明色（T 值）。

[0087] WDS 是功能段，用于定义图形平面上的矩形区域。如前所述，在图形平面上的固定矩形区域中执行清除和呈现的时元内，存储器管理是连续的。图形平面上的该矩形区域被称为窗口，它是用 WDS 来定义的。图 8A 示出了 WDS 的数据结构。如图所示，WDS 包括 ：window_id 字段，唯一地标识图形平面上的该窗口 ；window_horizontal_position 字段，指定图形平面上该窗口的左上方像素的水平位置 ；window_vertical_position 字段，指定图形平面上该窗口的左上方像素的垂直位置 ；window_width 字段，指定图形平面上该窗口的长度 ；window_height 字段，指定图形平面上该窗口的高度。

[0088] window_horizontal_position 字段、window_vertical_position 字段、window_width 字段和 window_height 字段可以取以下值。图形平面作为这些字段值的坐标系。图形平面具有由参数 window_width 和 window_height 定义的二维尺寸。

[0089] window_horizontal_position 字段指定图形平面上该窗口的左上方像素的水平位置，因此取值范围是 0 至 (video_width)-1。window_vertical_position 字段指定图形平面上该窗口的左上方像素的垂直位置，因此取值范围是 0 至 (video_height)-1。

[0090] window_width 字段指定图形平面上该窗口的长度，因此取值范围是 1 至 (video_width)-(window_horizontal_position)。window_height 字段指定图形平面上该窗口的高度，因此取值范围是 1 至 (video_height)-(window_vertical_position)。

[0091] 对于每个时元，使用 WDS 中的 window_horizontal_position、window_vertical_position、window_width 和 window_height 字段，可以定义窗口的位置和尺寸。因此，在创作时，创作者可以调整窗口，使其出现在时元中的每个图像的预期边缘部分中，从而不干扰该图像的图案。可以清楚地观看按照这种方式显示的字幕的图形。对于每个时元，可以定义 WDS。因此，当图像图案随时间而变化时，可以基于这样的变化，移动图形，从而不降低可视度。这将电影质量增强到与将字幕集成到运动图像中相同的等级。

[0092] 下面解释 END。END 是用于表示 DS 传输结束的功能段。END 位于 DS 中最后的 ODS 之后。END 包括 ：segment_type 字段，表示段类型“END”；segment_length，表示 END 的数据长度。这些字段不是本发明的主要特色，因此这里不再赘述。

[0093] 下面解释 PCS（合成段）。

[0094] PCS 是功能段，用于合成可与运动图像同步的画面。图 8B 示出了 PCS 的数据结构。如图所示，PCS 包括 ：segment_type 字段 ；segment_length 字段 ；composition_number 字段 ；composition_state 字段 ；palette_update_flag 字段 ；palette_id_ref 字段 ；composition_object(1) 至 composition_object(m) 字段。

[0095] composition_number 字段用从 0 至 15 的数字来唯一地标识 DS 中的图形更新。更详细地讲，对于每个图形更新，从时元的开始到 PCS，composition_number 字段加 1。

[0096] composition_state 字段表示 DS 是 Normal Case DS、AcquistionPoint DS、还是

时元 Start DS。

[0097] palette_update_flag 字段表示 PCS 是否描述仅调色板显示更新。仅调色板显示更新指的是只用新调色板替换先前调色板的更新。为了表示仅调色板显示更新，将 palette_update_flag 字段设置为 1。

[0098] palette_id 字段指定在该 DS 中使用的调色板。

[0099] composition_object(1) 至 composition_object(m) 字段各包含用于控制该 DS 中的单个窗口的信息。在图 8B 中，作为示例，虚线 wd1 表示 composition_object(i) 的内部结构。如图所示，composition_object(i) 包括 :object_id 字段 ;window_id 字段 ;object_cropped_flag 字段 ;object_horizontal_position 字段 ;object_vertical_position 字段 ;cropping_rectangle 信息 (1) 至 cropping_rectangle 信息 (n)。

[0100] object_id 字段表示对应于窗口中图形对象的 ODS 的标识符，该对象对应于 composition_object(i)。

[0101] window_id 字段表示在 PCS 中被分配了图形对象的窗口的标识符。至多两个图形对象可以被分配给一个窗口。

[0102] object_cropped_flag 字段表示是否要显示对象缓冲器中剪裁的图形对象。当将 object_cropped_flag 字段设为 1 时，显示对象缓冲器中剪裁的图形对象。当将 object_cropped_flag 字段设为 0 时，不显示对象缓冲器中剪裁的图形对象。

[0103] object_horizontal_position 字段表示图形对象在图形平面的左上方像素的水平位置。

[0104] object_vertical_position 字段表示图形对象在图形平面的左上方像素的垂直位置。

[0105] 当 object_cropped_flag 字段为 1 时，cropping_rectangle 信息 (1)cropping_rectangle 信息 (n) 是有效的。作为示例，虚线 wd2 指示了 cropping_rectangle 信息 (i) 的内部结构。如图所示，cropping_rectangle 信息 (i) 包括 :object_cropping_horizontal_position 字段 ;object_cropping_vertical_position 字段 ;object_cropping_width 字段 ;object_cropping_height 字段。

[0106] object_cropping_horizontal_position 字段表示剪裁矩形在图形对象中左上角的水平位置。剪裁矩形用于取出图形对象的一部分，其对应于 ETSI EN 300 743 中的“区域”。

[0107] object_cropping_vertical_position 字段表示剪裁矩形在图形对象中左上角的垂直位置。

[0108] object_cropping_width 字段表示剪裁矩形在图形对象中的水平长度。

[0109] object_cropping_height 字段表示剪裁矩形在图形对象中的垂直长度。

[0110] 下面具体描述 PCS，在所使用的例子中，当运动图像的再现进行时，通过三个向图形平面写入的操作，顺序地显示图 6 所示的三个字幕“Actually”、“I lied to you.”和“Sorry.”。图 9 示出了实现该字幕显示的示例性描述。在图中，一个时元具有 DS1(Epoch Start DS)、DS2(Normal Case DS) 和 DS3(Normal Case DS)。DS1 包括 :一个 WDS，用于定义显示字幕的窗口 ;一个 ODS，表示行“Actually I liedto you.Sorry.”;一个 PCS。DS2 包括一个 PCS。DS3 包括一个 PCS。

[0111] 下面描述各个 PCS。图 10 至 12 示出了属于 DS1 和 DS3 的 WDS 和 PCS。

[0112] 图 10 示出了 DS1 中的 PCS 和 WDS。在图中，WDS 中的 window_horizontal_position 字段值和 window_vertical_position 字段值指定图形平面上的窗口的左上方坐标 LP1，WDS 中的 window_width 字段值和 window_height 字段值指定窗口的宽度和高度。

[0113] PCS 中 cropping_rectangle 信息的 object_cropping_horizontal_position 字段值和 object_cropping_vertical_position 字段值指定剪裁矩形在坐标系中左上角坐标 ST1，该剪裁矩形的原点是对象缓冲器中图形对象的左上方坐标。剪裁矩形是一个区域，由从左上角坐标 ST1 的 object_cropping_width 字段值和 object_cropping_height 字段值进行定义。剪裁过的图形对象位于区域 cp1（被虚线框包围）中，所以，剪裁过的图形对象的左上角位于一个由坐标系中该图形平面的 object_horizontal_position 字段和 object_vertical_position 字段指定的像素。这样，将字幕“Actually I lied to you. Sorry.”中的“Actually”写入图形平面上的窗口中。字幕“Actually”覆盖在一张图片之上，然后，显示所得的图像。

[0114] 图 11 示出了 DS2 中的 PCS。由于图中的 WDS 的描述与图 10 相同，所以这里不再赘述。同时，cropping_rectangle 信息的描述与图 10 不同。在图 11 中，cropping_rectangle 信息中的 object_cropping_horizontal_position 字段值和 object_cropping_vertical_position 字段值指定与对象缓冲器中的字幕“I lied to you.”相对应的剪裁矩形的左上方坐标，而 object_cropping_width 字段值和 object_cropping_height 字段值指定剪裁矩形的宽度和高度。从而，将字幕“I lied to you.”写入图形平面上的窗口中。字幕“I lied to you.”覆盖在一张图片上，并显示所得的图像。

[0115] 图 12 示出了 DS3 中的 PCS。由于图中的 WDS 的描述与图 10 相同，所以这里不再赘述。同时，cropping_rectangle 信息的描述与图 10 不同。在图 12 中，object_cropping_horizontal_position 字段值和 object_cropping_vertical_position 字段值表示对象缓冲器中与字幕“Sorry.”相对应的剪裁矩形的左上角坐标，而 object_cropping_width 字段值和 object_cropping_height 字段值表示剪裁矩形的宽度和高度。从而，将字幕“Sorry.”写入图形平面上的窗口中。字幕“Sorry.”覆盖在一张图片上，并显示所得图像。

[0116] 图 13 示出了当执行如图 10 至 12 的图形更新时对象缓冲器的存储器空间。如图所示，对象缓冲器有四个存储区域 A 至 D，它们各有固定的高度和宽度以及固定的位置。在存储器区域 A 至 D 中，图 10 所示的字幕存储在存储区域 A 中。存储区域 A 至 D 中的每一个用与该存储区域中要存储的图形对象相对应的 object_id 标识。详细地讲，存储区域 A 用 object_id ＝ 1 标识，存储区域 B 用 object_id ＝ 2 标识，存储区域 C 用 object_id ＝ 3 标识，存储区域 D 用 object_id ＝ 4 标识。为了维持从对象缓冲器到图形平面的传输效率，存储区域 A 至 D 中的每一个区域的高度和宽度都是固定的。因此，当解码之后获得具有相同 object_id 的图形对象时，将该图形对象写入由该 object_id 标识的存储区域中，覆盖现有的图形对象。例如，为了在与图 10 至 12 中显示的字幕相同的位置用相同的尺寸显示字幕，需要在后续的 DS 中提供与 DS1 中的 ODS 具有相同 object_id 的 ODS。通过如此添加相同的 object_id，对象缓冲器中的图形对象被新的图形对象覆盖，新的图形对象的显示位置以及尺寸与被覆盖的图形对象相同。

[0117] 下面描述实现显示效果的约束条件。为了流畅地显示字幕，在窗口上需要执行清

除和呈现。当以视频帧的帧速率执行窗口清除和窗口呈现时，需要下面的从对象缓冲器到图形平面的传输速率。

[0118] 首先，检查对窗口尺寸的约束条件。假设 Rc 是从对象缓冲器到图形平面的传输速率。在最糟糕的情况下，需要以 Rc 执行窗口清除和窗口呈现。换言之，窗口清除和窗口呈现各需要以 Rc 的一半 (Rc/2) 执行。

[0119] 为了一个视频帧同步窗口清除和窗口呈现，需要满足：

[0120] （窗口尺寸）×（帧速率）≈ Rc/2。

[0121] 如果帧速率为 29.97，则

[0122] Rc ＝（窗口尺寸）×2×29.97

[0123] 为了显示字幕，窗口尺寸至少需要是整个图形平面的 25％至 33％。如果图形平面的像素总数量是 1920×1080，每个像素的索引的比特长度为 8 比特，则图形平面的总容量是 2M 字节（≈ 1920×1080×8）。

[0124] 假定窗口尺寸是图形平面的 1/4，即 500K 字节（＝ 2M 字节 /4）。将其代入上式，则得到 Rc ＝ 256Mbps(500K 字节 ×2×29.97)。

[0125] 因此，如果窗口尺寸是图形平面的大约 25％至 33％，那么，只要用 Rc ＝ 256Mbps 来显示字幕，就可以实现字幕的显示效果，而不会与运动图像失去同步。

[0126] 如果以视频帧速率的 1/2 或 1/4 执行窗口清除和窗口呈现，则在 Rc 相同的情况下窗口尺寸可以为两倍或四倍。下面描述窗口的位置和范围。如前所述，窗口的位置和范围在一个时元内是固定的，原因如下。

[0127] 如果窗口的位置或范围在时元中变化，则需要改变到图形平面的写地址。这会产生开销，从而导致从对象缓冲器到图形平面的传输速率 Rc 降低。

[0128] 当向图形平面传输解码后的图形对象时，为了降低开销，在一个窗口中可同时显示的图形对象的数量是有限的。当设置图形对象的边缘部分的地址时，会产生这里所说的开销发生。如果边缘部分的数量较大，则该开销会增加。

[0129] 如果对一个窗口中可显示的图形对象的数量没有限制，那么，当向图形平面传输图形对象时，开销会无限地发生，这会增加传输负载的变化。另一方面，如果一个窗口中的图形对象的数量被限制为 2，那么，基于最糟糕情形下开销数量为 4 的假设，可以设置传输速率 Rc。因此，可以很容易地确定传输速率 Rc 的最低标准。对窗口的解释到此结束。

[0130] 下面说明如何将承载诸如 PCS 和 ODS 之类功能段的 DS 分配到 AV 剪辑的再现时间轴上。时元是再现时间轴上的一个时间段，其中，存储器管理是连续的，并且，时元由一个或多个 DS 构成。因此，将 DS 有效地分配到 AV 剪辑的再现时间轴上是很重要的。这里所说的 AV 剪辑的再现时间轴是一个时间轴，用于定义构成 AV 剪辑中复用的视频流的各图片的解码时间和呈现时间。再现时间轴上的解码时间和呈现时间用 90KHz 的时间精度来表示。DS 中的 PCS 和 ODS 的 DTS 和 PTS 表示该再现时间轴上的同步控制的时机。换言之，通过使用 PCS 和 ODS 的 DTS 和 PTS 来实施同步控制，将 DS 分配到再现时间轴上。

[0131] 首先介绍用 PCS 和 ODS 的 DTS 和 PTS 来实施同步控制。

[0132] DTS 表示 ODS 的解码处理的开始时间，具有的精度是 90KHz。PTS 表示 ODS 的解码处理的结束时间，具有的精度是 90KHz。

[0133] 解码处理包括：对 ODS 进行解码；然后，将通过解码而生成的解压缩图形对象传输

到对象缓冲器。该解码处理并不是瞬间结束的，而是需要一定的时间量。ODS 的 DTS 和 PTS 分别表示 ODS 的解码开始时间和解码结束时间，以指定解码处理的开始和结束。

[0134] 由于 PTS 所示的时间是最终期限，所以，需要在 PTS 所示时间之前对 ODS 进行解码并将解压缩的图形对象存储到对象缓冲器中。

[0135] DSn 中任一 ODSj 的解码开始时间用 DTS(DSn[ODSj]) 表示，其精度是 90KHz。因此，DSn 中 ODSj 的解码结束时间（即 PTS(DSn[ODSj])）是 DTS(DSn[ODSj]) 与解码处理所需最长时间之和。

[0136] 假设 SIZE(DSn[ODSj]) 表示 ODSj 的尺寸，Rd 表示 ODS 解码速率。那么，解码处理所需的最长时间（以秒为单位）是 SIZE(DSn[ODSj])//Rd。符号“//”表示一种将小数部分取整的除法运算符。

[0137] 通过将该最长时间转换为 90KHz 的精度并将该结果加上 ODSj 的 DTS，则可以计算出 PTS 指定的 ODSj 的解码结束时间，精度为 90KHz。

[0138] DSn 中 ODSj 的该 PTS 可用下面的公式表示：

[0139] PTS(DSn[ODSj]) = DTS(DSn[ODSj])+

[0140] 90000×(SIZE(DSn[ODSj])//Rd)

[0141] 此外，DSn 中两个相邻的 ODS(ODSj 和 ODSj+1) 需要满足下面的关系：

[0142] PTS(DSn[ODSj]) ≤ DTS(DSn[ODSj+1])

[0143] DSn 中的 END 表示 DSn 的结束。因此，END 表示 DSn 中最后一个 ODS(ODSlast) 的解码结束时间。ODSlast 的 PTS(PTS(DSn[ODSlast])) 表示 ODSlast 的解码结束时间，所以，如下设置 END 的 PTS：

[0144] PTS(DSn[END]) = PTS(DSn[ODSlast])

[0145] 同时，如下设置 DSn 中的 PCS 的 DTS 和 PTS。

[0146] PCS 的 DTS 表示 DSn 中顶部 ODS(ODS1) 的解码开始时间或比它更早的时间。这是因为，需要将 PCS 加载到再现装置 200 的缓冲器中，与 ODS1 的解码开始时间 (DTS(DSn[ODS1])) 和 DSn 中的顶部 PDS(PDS1) 变为无效时 (PTS(DSn[PDS1])) 相同或更早。也就是说，PCS 的 DTS 需要满足下面的公式：

[0147] DTS(DSn[PCS]) ≤ DTS(DSn[ODS1])

[0148] DTS(DSn[PCS]) ≤ PTS(DSn[ODS1])

[0149] 另一方面，如下计算 PCS 的 PTS：

[0150] PTS(DSn[PCS]) ≥ DTS(DSn[PCS])+DECODEDURATION(DSn)

[0151] 在这里，DECODEDURATION(DSn) 表示 DSn 中的 PCS 所述的更新所用的全部图形对象的呈现和解码所需的时间。尽管 DECODEDURATION(DSn) 不是固定值，它也不会受诸如再现装置实现和状态差异之类因素的影响。当 DSn.PCS.OBJ[j] 表示 DSn 中的 PCS 描述的画面合成所用的图形对象时，可以通过以下方面来改变 DECODEDURATION(DSn)：(i) 窗口清除所需的时间；(ii) 对 DSn.PCS.OBJ[j] 进行解码所需的时间；(iii) 将 DSn.PCS.OBJ[j] 写到图形平面上所需的时间。因此，只要 Rd 和 Rc 是预先确定的，那么，DECODEDURATION(DSn) 就是相同的，而不管再现装置的实现方式。因此，在合成时，计算上述各时间段的长度，从而指定 PCS 的 PTS。

[0152] 基于图 14 所示的程序，执行 DECODEDURATION(DSn) 的计算。图 15 以及图 16A

和 16B 的流程图示出了该程序的算法。下面结合这些附图解释 DECODEDURATION(DSn) 的计算过程。在图 15 中，调用函数 PLANEINITIALIZATIONTIME，然后将返回值加上 decode_duration(S1)。函数 PLANEINITIALIZATIONTIME(图 16A) 是用于计算初始化图形平面从而产生 DSn 的显示所需时间的函数。在步骤 S1 中，用 DSn、DSn.PCS.OBJ[0] 和 decode_duration 作为参数，调用该 PLANEINITIALIZATIONTIME 函数。

[0153] 图 16A 示出了 PLANEINITIALIZATIONTIME 函数的过程。在图中，initialize_duration 是一个变量，用于表示 PLANEINITIALIZATIONTIME 函数的返回值。

[0154] 步骤 S2 判断 DSn 中的 PCS 的 composition_state 字段是不是 EpochStart。如果 composition_state 字段是 Epoch Start(S2 :是，图 14 中 DSn.PCS.composition_state == EPOCH_START)，则将清除图形平面所需的时间设置为 initialize_duration(S3)。

[0155] 如上所述，假设从对象缓冲器到图形平面的传输速率 Rc 是 256000000，图形平面的总尺寸是 (video_width)*(video_height)。那么，清除图形平面所需的时间 (单位是秒) 是 (video_width)*(video_height)//256000000。将其乘以 90000Hz，用 PTS 精度表示。因此，清除图形平面所需的时间是 90000×(video_width)*(video_height)//256000000。将其加上 intialize_duration，作为返回值返回。

[0156] 如果 composition_state 字段不是 Epoch Start(S2 : 否)，则对所有窗口 Window[i]，执行将清除 Window[i] 所需的时间加上 initialize_duration 的操作 (S4)。如前所述，假设从对象缓冲器到图形平面的传输速率 Rc 是 256000000，并且，Window[i] 的总尺寸是 Σ SIZE(WDS.WIN[i])。那么，清除全部 Window[i] 所需的时间 (单位是秒) 是 Σ SIZE(WDS.WIN[i])//256000000。将其乘以 90000Hz，用 PTS 精度表示。因此，清除所有 Window[i] 所需的时间是 90000× Σ SIZE(WDS.WIN[i])//256000000。将其加上 intialize_duration，作为返回值返回。对 PLANEINITIALIZATIONTIME 函数的描述到此结束。

[0157] 在图 15 中，步骤 S5 判断 DSn 中的图形对象的数量是 1 还是 2(图 14 中的 if(DSn.PCS.num_of_objects == 2)、if(DSn.PCS.num_of_objects == 1))。如果 DSn 中的图形对象的数量是 1(S5 :== 1)，则将用于等待该图形对象解码结束的等待时间加上 decode_duration(S6)。通过调用 WAIT 函数，计算该等待时间 (在图 14 中，(decode_duration+ = WAIT(DSn，DSn.PCS.OBJ[0]，decode_duration)))。使用 DSn、DSn.PCS.OBJ[0] 和 decode_duration 作为参数，调用 WAIT 函数，并将表示等待时间的 wait_duration 作为返回值返回。

[0158] 图 16B 示出了 WAIT 函数的过程。

[0159] 在 WAIT 函数中，current_duration 是一个变量，将 decode_duration 设置为该参数，object_definition_ready_time 是表示 DSn 中的图形对象 OBJ[i] 的 PTS 的变量。

[0160] 此外，current_time 是一个变量，表示 current_duration 与 DSn 中的 PCS 的 DTS 之和。如果 object_definition_ready_time 大于 current_time(S7 :是，图 14 中 if(current_time < object_definition_ready_time))，则将 object_definition_ready_time 和 current_time 之间的差值设为 wait_duration，然后作为返回值返回 (S8，图 14 中 wait_duration+ = object_definition_ready_time-current_time)。WAIT 函数到此结束。

[0161] 再回到图 15 中，WAIT 函数的返回值与用于在 OBJ[0] 所属窗口上进行呈现所需的时间 (90000× Σ SIZE(DSn.WDS.WIN[0])//256000000) 被设置为 decode_duration(S9)。

[0162] 上述过程涉及 DSn 中图形对象数量为 1 的情况。如果图形对象的数量是 2(S5 :

＝2，图 14 中的 if(DSn.PCS.num_of_objects ＝＝ 2))，则使用 DSn、DSn.PCS.OBJ[0] 和 decode_duration 作为参数，调用 WAIT 函数，然后将 WAIT 函数的返回值加到 decode_duration 中 (S10)。

[0163] 步骤 S11 判断 OBJ[0] 所属的窗口与 OBJ[1] 所属的窗口是否相同 (在图 14 中，if(DSn.PCS.OBJ[0].window_id ＝＝ DSn.PCS.OBJ[1].window_id))。如果判断结果是肯定的 (S11 :是)，则用 DSn、DSn.PCS.OBJ[1] 和 decode_duration 作为参数，调用 WAIT 函数，然后将 WAIT 函数的返回值加上 decode_duration(S12)。此外，将在 OBJ[0] 和 OBJ[1] 所属窗口上进行呈现所需的时间 (90000× Σ SIZE(DSn.WDS.WIN[0].window_id)//256000000) 加上 decode_duration(S13)。

[0164] 另一方面，如果判断结果是否定的 (S11 :否)，则将在 OBJ[0] 所属窗口上进行呈现所需的时间 (90000× Σ SIZE(DSn.WDS.OBJ[0].window_id)//256000000) 加上 decode_duration(S15)。此后，用 DSn、DSn.PCS.OBJ[1] 和 decode_duration 作为参数，调用 WAIT 函数，然后将 WAIT 函数的返回值加上 decode_duration(S16)。此外，将在 OBJ[1] 所属窗口上进行呈现所需的时间 (90000× Σ SIZE(DSn.WDS.OBJ[1].window_id)//256000000) 加上 decode_duration(S 17)。这样，计算出 DECODEDURATION(DSn)。

[0165] 下面通过具体的例子，说明如何设置一个 DS 中的 PCS 的 PTS。

[0166] 在图 17A 示出的情形中，与一个 ODS(ODS1) 相对应的一个 OBJ(OBJ1) 属于一个窗口。图 17B 和 17C 示出了图 14 中使用的参数之间的关系的时序图。这些时序图都有三级。在这三级中，“图形平面访问”级和“ODS 解码”级表示当再现 ODS 时并行执行的两个过程。上述算法基于这两个过程并行执行的假设。

[0167] 图形平面访问由清除时间 (1) 和写入时间 (3) 构成。清除时间 (1) 表示用于清除整个图形平面所需的时间 (90000×((图形平面尺寸)//256000000))，或用于清除图形平面上所有窗口所需的时间 (Σ (90000×((窗口 [i] 的尺寸)//256000000)))。

[0168] 写入时间 (3) 表示在整个窗口上进行呈现所需的时间 (90000×((窗口尺寸)//256000000))。

[0169] ODS 解码由解码时间 (2) 构成。解码时间 (2) 表示从 ODS1 的 DTS 到 PTS 的时间周期。

[0170] 清除时间 (1)、解码时间 (2) 和写入时间 (3) 的变化取决于要清除的范围、要解码的 ODS 的尺寸和要写入图形平面的图形对象的尺寸。在图 17 中，为简单起见，假设解码时间 (2) 的开始与清除时间 (1) 的开始相同。

[0171] 图 17B 所示的情形中，解码时间 (2) 比清除时间 (1) 要长。在这种情况下，decode_duration 是解码时间 (2) 与写入时间 (3) 之和。

[0172] 图 17C 所示的情形中，清除时间 (1) 比解码时间 (2) 要长。在这种情况下，decode_duration 是清除时间 (1) 与写入时间 (3) 之和。

[0173] 在图 18A 至 18C 所示情形中，与两个 ODS(ODS1 和 ODS2) 相对应的两个 OBJ(OBJ1 和 OBJ2) 属于一个窗口。在图 18B 和 18C 中，解码时间 (2) 表示对 ODS1 和 ODS2 进行解码所需的总时间。同样，写入时间 (3) 表示将 OBJ1 和 OBJ2 写入图形平面所需的总时间。尽管 ODS 的数量是 2，但是可以采用与图 17 相同的方式计算 decode_duration。详细地讲，如果 ODS1 和 ODS2 的解码时间 (2) 比清除时间 (1) 长，则 decode_duration 是解码时间 (2)

与写入时间 (3) 之和，如图 18B 所示。

[0174] 如果清除时间 (1) 比解码时间 (2) 长，则 decode_duration 是清除时间 (1) 与写入时间 (3) 之和，如图 18C 所示。

[0175] 图 19A 至 19C 所示的情形中，OBJ1 属于 Window1，而 OBJ2 属于 Window2。在这种情形中，如果清除时间 (1) 比 ODS1 和 ODS2 的解码时间 (2) 长，则 decode_duration 是清除时间 (1) 与写入时间 (3) 之和。另一方面，如果清除时间 (1) 比解码时间 (2) 短，则可以将 OBJ1 写入 Window1，而不必等待解码时间 (2) 的结束。在这种情况下，decode_duration 不仅仅是解码时间 (2) 与写入时间 (3) 之和。假设写入时间 (31) 表示将 OBJ1 写入 Window1 所需的时间，写入时间 (32) 表示将 OBJ2 写入 Window2 所需的时间。在图 19B 所示情形中，解码时间 (2) 比清除时间 (1) 与写入时间 (31) 之和要长。在这种情况下，decode_duration 是解码时间 (2) 与写入时间 (32) 之和。

[0176] 图 19C 所示的情形中，清除时间 (1) 与写入时间 (31) 之和比解码时间 (2) 长。在这种情况下，decode_duration 是清除时间 (1)、写入时间 (31) 以及写入时间 (32) 之和。

[0177] 根据播放机模型，图形平面的尺寸是固定的。窗口和 ODS 的尺寸和数量也是在合成时预先设置的。因此，计算出来的 decode_duration 可能是以下之一 ：清除时间 (1) 与写入时间 (3) 之和 ；解码时间 (2) 与写入时间 (3) 之和 ；解码时间 (2) 与写入时间 (32) 之和 ；清除时间 (1)、写入时间 (31) 与写入时间 (32) 之和。通过基于如此计算出的 decode_duration 来设置 PCS 的 PTS，可以使图形与图像数据同步，且具有高精度。如此准确的同步控制是通过定义窗口和限制窗口中的清除和呈现操作来实现的。因此，在创作中，对概念“窗口”的介绍是非常重要的。

[0178] 下面介绍如何设置 DSn 中 WDS 的 DTS 和 PTS。设置 WDS 的 DTS，以满足下面的公式 ：

[0179] DTS(DSn[WDS]) ≥ DTS(DSn[PCS])

[0180] WDS 的 PTS 表示启动向图形平面写入的最后期限。由于向图形平面写入只限于一个窗口，所以，通过从 PCS 的 PTS 所示时间减去在所有窗口上进行呈现所需的时间，可以确定启动向图形平面写入的时间。假设 Σ SIZE(WDS.WIN[i]) 是 Windows[i] 的总尺寸。那么，用于在所有 Windows[i] 上进行清除和呈现所需的时间是 Σ SIZE(WDS.WIN[i])//256000000。用 90000KHz 的精度表达该时间，得到 90000×SIZE(WDS.WIN[i])//256000000。

[0181] 因此，可以如下计算 WDS 的 PTS ：

[0182] PTS(DSn[WDS]) ＝ PTS(DSn[PCS])-

[0183] 90000×SIZE(WDS.WIN[i])//256000000

[0184] 由于 WDS 的 PTS 表示最后期限，所以，可以在该 PTS 所示的时间之前启动向图形平面的写入。也就是说，在属于这两个窗口之一的一个 ODS 解码结束之后，可以将通过解码获得的图形对象立即写入图 19 所示的窗口中。

[0185] 因此，使用 WDS 的 DTS 和 PTS，可以将窗口分配给 AV 剪辑的再现时间轴上的预期点。对 DSn 中 PCS 和 WDS 的 DTS 和 PTS 的介绍到此结束。

[0186] 每个 DS 中的 PCS 在从由其 DTS 示出的时间至由其 PTS 示出的时间中是活动的。在 PCS 为活动的该时间周期被称为 DS 中的 PCS 的活动周期。

[0187] 以下说明 DS 中的 PCS 的活动周期是如何重叠的。当图形流包含多个 DS 时，希望并行处理两个或更多 DS。为了使再现装置中能够实现此类并行处理，DS 中的 PCS 的活动周期需要重叠。同时，蓝光光盘只读格式保证以必需的最小结构的再现装置执行解码。

[0188] 蓝光光盘只读格式的解码器模型基于流水线处理（流水线解码模型）。流水线解码模型能够同时从对象缓冲器读取一个 DS 的图形对象到图形平面，同时地，解码并向对象缓冲器写入下一个 DS 的图形对象。

[0189] 当再现装置遵循流水线解码模型时，需要合理确定引入间隔。这里所指的引入间隔是从一个 DS 处理的开始到下一个 DS 处理的开始的时间周期。一个 DS 的处理包括，对象缓冲器可以被分成两个处理，也就是，解码 ODS 并将未压缩的图形对象写入对象缓冲器的处理，以及从对象缓冲器中读取未压缩图形对象并将其写入图形平面的处理。如此，一个 DS 中的 PCS 的活动周期可以被分解，如图 20 所示。如该图所示，一个 DS 的处理由解码 ODS 并将图形写入对象缓冲器所需的时间，以及从对象缓冲器中读取图形并将其写入图形平面所需的时间。

[0190] 流水线解码模型能够同时将图形写入对象缓冲器以及从对象缓冲器中读取图形。因此，可以并行处理两个 DS，如图 21 所示。图 21 示出在流水线解码模型中如何并行处理两个 DS(DS_n 和 DS_{n+1})。

[0191] 如图解，并行处理 DS_n 和 DS_{n+1}，使得从对象缓冲器中读取 DS_n 的时间与向对象缓冲器写入 DS_{n+1} 的时间重叠。

[0192] 在此类并行处理中，在将 DS_n 的图形对象写入对象缓冲器完成之后，DS_{n+1} 的图形对象被写入对象缓冲器。

[0193] DS_n 中一个 ODS 的解码停止时间由 DS_n 中 END 的 PTS 表示。同样，DS_{n+1} 中一个 ODS 的开始解码的最早时间由 DS_{n+1} 中 PCS 的 DTS 表示。因此，预先设定 DS_n 中 END 的时戳和 DS_{n+1} 中 PCS 的时戳，使得满足

[0194] $PTS(DS_n[END]) \leqslant DTS(DS_{n+1}[PCS])$

[0195] 通过以此方式设定引入间隔，在流水线解码模型中可以并行处理 DS_n 和 DS_{n+1}。

[0196] 图 22 示出三个 DS(DS0，DS1，和 DS2) 中 PCS 的活动周期重叠的情况。

[0197] 以下说明如何在再现时间轴上设定重叠 DS 中的功能段的时戳。图 23 示出每个 DS0 和 DS1 中功能段的时戳，其中 DS0 和 DS1 的 PCS 具有重叠的活动周期。在此图中，DS0 中 WDS，PDS，和最高级 ODS(ODS1) 的 DTS 被设定为等于 DS0 中 PCS 的 DTS。这意味着当 DS0 中 PCS 的活动周期开始，DS_n 中 ODS 的解码就立即开始。因此，在由 PCS 的 DTS 示出的时间开始 ODS1 的解码。同时，在由 DS0 中 END 的 PTS 示出的时间停止 ODS_n 的解码，其中 ODS_n 为 DS0 中的最后一个 ODS。这里应该注意，DS0 中 WDS 的 DTS，PDS，和最高级 ODS 的 DTS 可以改为设定为晚于 DS0 中 PCS 的 DTS。

[0198] DS1 中 PCS 的 DTS 示出一个时间，其等于或晚于由 DS0 中 END 的 PTS 示出的时间。因此，当在由 DS1 中 PCS 的 DTS 示出的时间开始 DS1 中 ODS 的解码时，可以在流水线解码模型中并行处理 DS0 和 DS1。

[0199] 以下检查在此类流水线处理中对图形平面的渲染处理。

[0200] 当并行处理 DS_n 和 DS_{n+1} 时，通过解码 DS_n 获得的图形对象和通过解码 DS_{n+1} 获得的图形对象可以同时写入图形平面，造成未能在屏幕上显示 DS_n 的图形对象。

[0201] 为了防止这一点，需要如下设定 DS_n 中 PCS 的 PTS 和 DS_{n+1} 中 PCS 的 PTS：

[0202] $PTS(DS_n[PCS])+(90,000\times \Sigma SIZE(DS_n[WDS]\cdot Window[i]))//256,000,000 \leq PTS(DS_{n+1}[PCS])$

[0203] 其中 $\Sigma SIZE(DS_n[WDS]\cdot Window[i])$ 为 Windows[i] 的总尺寸，$(90,000\times \Sigma SIZE(DS_n[WDS]\cdot Window[i]))//256,000,000$ 为渲染 Windows[i] 所需的时间。通过以此方式延迟 DS_{n+1} 的图形对象的显示时间，防止 DS_{n+1} 的图形对象覆盖 DS_n 的图形对象。图 24 示出根据此公式，DS0 至 DS2 中 PCS 的 PTS。

[0204] 当窗口的尺寸为图形平面的 1/4 时，$PTS(DS_n[PCS])$ 和 $PTS(DS_{n+1}[PCS])$ 之间的间隔等于视频流的一个帧周期。

[0205] 以下说明对 DS 中 PCS 的活动周期重叠的约束。如果属于一个 DS 的图形对象具有与属于前面紧邻的 DS 的图形对象相同的 object_id 以实现更新，这些 DS 中 PCS 的活动周期就不能重叠。假设 DS0 包括具有 object_id ＝ 1 的 ODS，DS1 包括具有相同 object_id ＝ 1 的 ODS。

[0206] 如果此 DS0 和 DS1 中 PCS 的活动周期重叠，DS1 中 ODS 载入再现装置 200，并在 DS0 停止之前解码。在此情况下，DS0 的图形对象被 DS1 的图形对象覆盖。这使得 DS1 的图形对象，而不是 DS0 的图形对象，出现在屏幕上。为了防止这一点，在图形更新的情况下，DS 中 PCS 的活动周期的重叠是被禁止的。

[0207] 图 25A 示出可以在一条流水线中处理两个 DS 的情况，而图 25B 示出不可以在一条流水线中处理两个 DS 的情况。在这些图中，DS0 具有 ODSA 和 ODSB，同时 DS1 具有 ODSC 和 ODSD。如果 DS1 中的 ODSC 和 ODSD 具有与 DS0 中的 ODSA 和 ODSB 不同的 object_id，DS0 和 DS1 中 PCS 的活动周期可以重叠，如图 25A 所示。如果 DS1 中的 ODSC 和 ODSD 具有与 DS0 中的 ODSA 和 ODSB 相同的 object_id，DS0 和 DS1 中 PCS 的活动周期不可以重叠，如图 25B 所示。

[0208] 可以通过以下“传送加速度”的方法克服该约束。例如，当 DS0 包含具有 object_id ＝ 1 的 ODSA，而且 DS1 包含 ODSC，用于更新 DS0 中 ODSA 的图形对象时，初始为 DS1 中 ODSC 给出不同于 object_id ＝ 1 的 object_id。只在 DS1 中 ODSC 的图形对象已经被存入对象缓冲器中之后，ODSC 的 object_id 改为 object_id ＝ 1，覆盖 DS0 中 ODSA 的图形对象。根据此方法，可以克服上述约束。这就是说，用于更新对象缓冲器中先前的图形对象的图形对象可以被载入对象缓冲器，无需等待显示先前的图形对象。

[0209] 因为以上方法可以用于图形更新中，一个 DS 不仅可以经常携带由其自身的 PCS 引用的 ODS，还可以经常携带由随后的 DS 的 PCS 引用的 ODS。在此情况下，必须向再现装置 200 指明哪些 ODS 属于 DS。为此，在 DS 自身中携带的所有 ODS 之后放置 END。再现装置 200 参考 DS 中的 END，以探测属于 DS 的 ODS 的末端。

[0210] 图 26 示出由 END 指明的 ODS 传送的末端。在此图中，第一层示出属于一个 DS 的功能段，而第二层示出 BD-ROM 100 上的这些功能段的安排。例如 PCS，WDS，PDS，和 ODS 的功能段被转换为 TS 数据包，并与视频流一同记录在 BD-ROM 100 上，该视频流同样被转换为 TS 数据包。

[0211] 每个对应于功能段的 TS 数据包和对应于视频流的 TS 数据包被给定称为 ATS 的时戳。对应于功能段的 TS 数据包和对应于视频流的 TS 数据包安排在 BD-ROM 100 上，使得具

有相同时戳的 TS 数据包彼此邻近。

[0212] 这意味着属于 DS 的 PCS,WDS,以及 PDS 在 BD-ROM 100 上是不连续,对应于视频流(由图中的字母 V 指明)的 TS 数据包插入其中。因此功能段相隔一定距离在 BD-ROM 100 上出现。当对应于功能段的 TS 数据包以一定间隔出现在 BD-ROM 100 上时,难以立即探测直到哪个 TS 数据包属于 DS。同样,DS 可以包括不由 DS 的 PCS 引用的 ODS,这使得探测更加困难。在此实施例中,然而,在属于 DS 的最后一个 ODS 之后提供 END。因此,即使当属于 DS 的功能段以一定间隔出现时,也容易探测到哪个 ODS 属于 DS。

[0213] 图 27 示出重叠的活动周期和 object_id 分配之间的关系。图 27A 示出四个 DS(DS0,DS1,DS2,和 DS3)。DS0 的 PCS 不描述任何图形对象的显示。DS1 的 PCS 描述屏幕上的对象 X 和 Y 的显示,DS2 的 PCS 描述屏幕上的对象 A,Y,和 C 的显示,而且 DS3 的 PCS 描述屏幕上的对象 D 的显示。

[0214] 图 27B 示出属于 DS 的 ODS 和 DS 中 PCS 的活动周期。DS0 包含对象 X 的 ODS。DS1 包含对象 Y 的 ODS。DS2 包含对象 A,B,和 C 的 ODS。DS3 包含对象 D 的 ODS。四个 DS 的每个中显示图形对象和传送 ODS 之间的矛盾可归于以上传送加速度。这些 DS 中 PCS 的活动周期部分重叠。图 27C 示出对象缓冲器中图形对象的安排。

[0215] 假定 object_id 0,1,2,3,和 4 分别分配给对象 X,Y,A,B,和 C。这里是这样一种情况,属于 DS3 的对象 D 可以分配给 object_id5,3,和 0 中的任何一个。

[0216] 因为 object_id 5 在 DS0 至 DS2 中未分配,该 object_id 是可能的。

[0217] 因为具有 object_id 3 的对象 B 包括在 DS2 中,但不由任何 DS 的 PCS 引用,该 object_id 是可能的。

[0218] 因为具有 object_id 0 的对象 X 显示在 DS1 中,该 object_id 是可能的。只要 DS1 中 PCS 的活动周期已经结束,显示对象 D 而不是对象 X 的问题就不会出现。

[0219] 相反地,不可能将 object_id 1,2,和 4 中的任何一个分配给对象 D。如果此类 object_id 的任何一个分配给对象 D,对象 D 而不是三个对象 A,Y,和 C 中的任何一个将会停止显示,且三个对象 A,Y,和 C 将显示在 DS2 中。

[0220] 因此,对象 D 可以分配给相同的 object_id,作为不在 DS 中 PCS 的活动周期中引用的对象,该活动周期与 DS3 中 PCS 的活动周期重叠,或作为由 DS 的 PCS 引用的对象,而其中该 PCS 的活动周期已经结束。

[0221] DS_n 和 DS_{n+1} 中 PCS 的活动周期的重叠基于 DS_n 和 DS_{n+1} 属于图形流中相同时元的前提。如果 DS_n 和 DS_{n+1} 属于不同时元,DS_n 和 DS_{n+1} 中 PCS 的活动周期不能重叠。这是因为如果在 DS_n 中 PCS 的活动周期结束之前,载入 DS_{n+1} 的 PCS 或 ODS,就不可能在 DS_n 中 PCS 的活动周期结束时刷新 (flush) 对象缓冲器和图形平面。

[0222] 当 DS_n 为 $EPOCH_m$ 的最后一个 DS(以下为“$EPOCH_m$DSlast[PCS]”),而 DS_{n+1} 为 $EPOCH_{m+1}$ 的第一个 DS(以下为“$EPOCH_{m+1}$DSfirst[PCS]”),DS_n 和 DS_{n+1} 的 PCS 的 PTS 需要满足以下公式:

[0223] $PTS(EPOCH_m\ DSlast[PCS]) \leqslant DTS(EPOCH_{m+1}DSfirst[PCS])$

[0224] 同样,DS_n 和 DS_{n+1} 中 PCS 的活动周期的重叠基于图形流为呈现图形流的前提。存在两种类型的图形流:呈现图形流;以及交互式图形流,该交互式图形流主要用于生成交互式显示。

[0225] 如果 DS_n 和 DS_{n+1} 属于交互式图形流，禁止 DS_n 和 DS_{n+1} 的重叠。在交互式图形流中，携带控制信息的段称为交互成分段 (ICS)。这样，需要设定 DS_n 和 DS_{n+1} 的时间信息，使得紧接在 DS_n 中 ICS 的活动周期之后，开始 DS_{n+1} 中 ICS 的活动周期。DS_n 中 ICS 的活动周期的结束通过 DS_n 中 ICS 的 PTS 示出，而 DS_{n+1} 中 ICS 的活动周期的开始通过 DS_{n+1} 中 ICS 的 DTS 示出。这里，$PTS(DS_n[ICS])$ 和 $DTS(DS_{n+1}[ICS])$ 需要满足以下公式：

[0226] $$PTS(DS_n[ICS]) \leqslant DTS(DS_{n+1}[ICS])$$

[0227] 这样完成对 DS 中 PCS 的活动周期重叠的说明。

[0228] 注意到以上说明的 DS 的数据结构 (PCS，WDS，PDS，和 ODS) 为以程序语言编写的类结构的实例。作者根据蓝光光盘只读格式中定义的语法编写类结构，在 BD-ROM 100 上创建这些数据结构。

[0229] 这样完成了对根据本发明第一实施例的记录介质的说明。以下说明根据本发明第一实施例的再现装置。图 28 示出了再现装置 200 的内部构造。再现装置 200 是基于该内部构造制造的。再现装置 200 主要包括三个部分：系统 LSI；驱动装置；微处理器系统。可以将这些部件装配到该装置的壳体和衬底上，从而制造再现装置 200。系统 LSI 是集成电路，包括各种处理单元，用于实现再现装置 200 的功能。再现装置 200 包括：BD 驱动器 1；读取缓冲器 2；PID 滤波器 3；传输缓冲器 4a、4b 和 4c；外围电路 4d；视频解码器 5；视频平面 6；音频解码器 7；图形平面 8；CLUT 单元 9；加法器 10；图形解码器 12。图形解码器 12 包括：编码数据缓冲器 13；外围电路 13a；流图形处理器 14；对象缓冲器 15；合成缓冲器 16；图形控制器 17。

[0230] BD 驱动器 1 执行 BD-ROM 100 的加载、读取和弹出。BD 驱动器 1 访问 BD-ROM 100。

[0231] 读取缓冲器 2 是 FIFO（先进先出）存储器。因此，从 BD-ROM 100 读取的 TS 数据包在读取缓冲器 2 中消除的次序与它们的到达次序相同。

[0232] PID 滤波器 3 对从读取缓冲器 2 输出的 TS 数据包执行滤波。更详细地讲，PID 滤波器 3 只把具有预定 PID 的 TS 数据包传递给传输缓冲器 4a、4b 和 4c。PID 滤波器 3 中没有缓冲机制。因此，进入 PID 滤波器 3 的 TS 数据包立即被写入传输缓冲器 4a、4b 和 4c。

[0233] 传输缓冲器 4a、4b 和 4c 是 FIFO 存储器，用于存储从 PID 滤波器 3 输出的 TS 数据包。传输速度 Rx 表示从传输缓冲器 4a 读取 TS 数据包的速度。

[0234] 外围电路 4d 具有连线逻辑，用于把从传输缓冲器 4a 读取的 TS 数据包转换成功能段。然后，将功能段存储在编码数据缓冲器 13 中。

[0235] 视频解码器 5 对从 PID 滤波器 3 输出的 TS 数据包进行解码，从而获得解压缩的图片，并将其写入视频平面 6。

[0236] 视频平面 6 是用于存储运动图像的平面存储器。

[0237] 音频解码器 7 对从 PID 滤波器 3 输出的 TS 数据包进行解码，并输出解压缩的音频数据。

[0238] 图形平面 8 是一个平面存储器，它具有一幅画面的存储区域，能够存储一幅画面的解压缩图形。

[0239] CLUT 电路 9 基于 PDS 中所示的 Y、Cr 和 Cb 值，对图形平面 8 上的解压缩图形的索引色 (index color) 进行转换。

[0240] 加法器 10 将 CLUT 电路 9 转换过的解压缩图形乘以 PDS 中所示的 T 值（透明色）。

然后，加法器 10 对所得的解压缩图形和视频平面 6 上的解压缩图片数据中的相应像素执行加法，并输出所得的图像。

[0241] 图形解码器 12 对图形流进行解码，从而获得解压缩的图形，并将解压缩的图形作为图形对象写入图形平面 8 中。对图形流进行解码之后，字幕和菜单就出现在屏幕上。

[0242] 图形解码器 12 从对象缓冲器 15 中读取属于 DSn 的图形对象，同时将属于 DSn+1 的图形对象写入对象缓冲器 15，从而执行流水线处理。

[0243] 图形解码器 12 包括：编码数据缓冲器 13；外围电路 13a；流图形处理器 14；对象缓冲器 15；合成缓冲器 16；图形控制器 17。

[0244] 编码数据缓冲器 13 用于将功能段与 DTS 和 PTS 一起存储。通过将传输缓冲器 4a 中存储的各 TS 数据包去除 TS 包头和 PES 包头以及将剩余的有效负载顺序地进行排列，得到这些功能段。去除的 TS 包头和 PES 包头中包含的 DTS 和 PTS 存储在编码数据缓冲器 13 中，与这些功能段相对应。

[0245] 外围电路 13a 具有连线逻辑，用于将来自编码数据缓冲器 13 的数据传输到流图形处理器 14，并将来自编码数据缓冲器 13 的数据传输到合成缓冲器 16。更详细地讲，如果当前时间到达 ODS 的 DTS，则外围电路 13a 将 ODS 从编码数据缓冲器 13 传输到流图形处理器 14。此外，如果当前时间到达 PCS 或 PDS 的 DTS，则外围电路 13a 将 PCS 或 PDS 从编码数据缓冲器 13 传输到合成缓冲器 16。

[0246] 流图形处理器 14 对 ODS 进行解码，从而获得具有索引色的解压缩图形，并将解压缩图形作为图形对象传输到对象缓冲器 15。流图形处理器 14 的解码是瞬时的，并且，通过解码获得的图形对象暂时存储在流图形处理器 14 中。尽管流图形处理器 14 的解码是瞬时的，但图形对象从流图形处理器 14 到对象缓冲器 15 的传输不是瞬时的。这是因为，对于蓝光只读光盘格式的播放机模型，到对象缓冲器 15 的传输是以 128Mbps 的传输速率执行的。DS 中的 END 的 PTS 给出了属于 DS 的所有图形对象到对象缓冲器 15 的传输的结束。因此，在 END 的 PTS 所示的时间之前，不会启动下一 DS 的处理。通过对各 ODS 进行解码而得到的图形对象向对象缓冲器 15 的传输开始于 ODS 的 DTS 所示的时间，结束于 ODS 的 PTS 所示的时间。

[0247] 如果 DSn 的图形对象和 DSn+1 的图形对象具有不同的 Object_id，那么，流图形处理器 14 将这两个图形对象写入对象缓冲器 15 中不同的存储区域。这样，就可以实现图形对象的流水线呈现，并且，DSn 的图形对象不会被 DSn+1 的图形对象覆盖。另一方面，如果 DSn 的图形对象和 DSn+1 的图形对象具有相同的 Object_id，那么，流图形存储器 14 将 DSn+1 的图形对象写入对象缓冲器 15 中也存储了 DSn 的图形对象的存储区域，从而覆盖 DSn 的图形对象。在这种情况下，不执行流水线处理。DS 也可以可以包括由该 DS 的 PCS 引用的 ODS 以及 PCS 未引用的 ODS。流图形处理器 14 不仅顺序地对该 PCS 引用的 ODS 进行解码，而且还对 PCS 未引用的 ODS 进行解码，然后，将解码所得的图形存储到对象缓冲器 15 中。

[0248] 对象缓冲器 15 对应于 ETSI EN 300 743 中的像素缓冲器。流图形处理器 14 解码的图形对象存储在对象缓冲器 15 中。对象缓冲器 15 的尺寸应当是图形平面 8 的尺寸的两倍或四倍。这是因为，为了实现滚读 (scrolling)，对象缓冲器 15 需要能够存储图形平面 18 两倍或四倍的图形。

[0249] 合成缓冲器 16 用于存储 PCS 和 PDS。当 DSn 和 DSn+1 中的 PCS 的激活时间重叠

时，组合缓冲器 16 存储 DSn 中的 PCS 和 DSn+1 中的 PCS。

[0250] 图形控制器 17 对合成缓冲器 16 中的 PCS 进行解码。基于解码结果，图形控制器 17 将 DSn+1 的图形对象写入对象缓冲器 15，同时从对象缓冲器 15 中读取 DSn 的图形对象，并将其呈现以供显示。在 DSn 中的 PCS 的 PTS 所示的时间，执行图形控制器 17 的呈现。DS_n 的图形对象的呈现和 DS_{n+1} 的图形对象的呈现之间的时间间隔如上所述。

[0251] 下面给出建议的传输速率以及 PID 滤波器 3、传输缓冲器 4a、4b 和 4c、图形平面 8、CLUT 单元 9、编码数据缓冲器 13、流图形处理器 14、对象缓冲器 15、合成缓冲器 16 和图形控制器 17 的缓冲器尺寸。图 29 示出了传输速率 Rx、Rc 和 Rd 以及图形平面 8、传输缓冲器 4a、编码数据缓冲器 13 和对象缓冲器 15 的尺寸。

[0252] 从对象缓冲器 15 到图形平面 8 的传输速率 Rc（像素合成速率）是再现装置 200 中最高的传输速率，根据窗口大小和帧速率，将其计算为 256Mbps（＝500K 字节 ×29.97×2）。

[0253] 与 Rc 不同，从流图形处理器 14 到对象缓冲器 15 的传输速率 Rd（像素解码速率）不必与帧速率一致，它可以是 Rc 的 1/2 或 1/4。因此，传输速率 Rd 是 128Mbps 或 64Mbps。

[0254] 从传输缓冲器 4a 到编码数据缓冲器 13 的传输速率 Rx（传输缓冲器泄漏速率）ODS 处于压缩状态下的传输速率。因此，通过用 ODS 的压缩率乘以 Rd，可以计算出传输速率 Re。例如，如果压缩率为 25％，则传输速率 Rx 是 16Mbps（＝64Mbps×25％）。

[0255] 给出的这些传输速率和缓冲器尺寸只是最低标准，传输速率和缓冲器大小也可以大于图 29 所示情况。

[0256] 在如上构造的再现装置 200 中，组成部件以流水线方式执行处理。

[0257] 图 30 的时序图示出了再现装置 200 中执行的流水线处理。在图中，第五级示出了 BD-ROM 100 上的 DS。第四级示出了 PCS、WDS、PDS、ODS 和 END 向编码数据缓冲器 13 的写入时间。第三级示出了流图形处理器 14 对 ODS 的解码时间。第二级示出了合成缓冲器 16 的存储内容。第一级示出了图形控制器 17 的操作。

[0258] ODS1 和 ODS2 的 DTS 分别给出了 t31 和 t32。因此，在 t31 和 t32 之前，需要将 ODS1 和 ODS2 缓冲在编码数据缓冲器 13 中。因此，在 t31 之前，ODS1 向编码数据缓冲器 13 的写入结束，在 t31，解码时间 dp1 开始，在 t31 之前，ODS2 向编码数据缓冲器 13 的写入结束，在 t31，解码时间 dp2 开始。

[0259] 同时，ODS1 和 ODS2 的 PTS 分别给出了 t32 和 t33。因此，在 t32 之前，流图形处理器 14 对 ODS1 的解码结束，在 t33 之前，流图形处理器 14 对 ODS2 的解码结束。因此，在 ODS 的 DTS 给出的时间之前，ODS 缓存在编码数据缓冲器 13 中，在 ODS 的 PTS 所示的时间之前，对所缓冲的 ODS 进行解码并将其传输到对象缓冲器 15。

[0260] 在第一级中，cd1 表示图形控制器 17 清除图形平面 8 所需的时间，td1 表示图形控制器 17 将对象缓冲器 15 中包含的图形写入图形平面 8 所需的时间。WDS 的 PTS 表示启动写入图形的最后期限。PCS 的 PTS 表示向图形平面 8 中写入图形结束并呈现所写入图形以供显示的时间。因此，在 PCS 的 PTS 所示的时间处，获得图形平面 8 上的一幅画面的解压缩图形。CLUT 单元 9 对解压缩的图形执行颜色转换，加法器 10 将该图形覆盖在视频平面 6 上存储的解压缩图片上。从而产生所得的图像。

[0261] 在图形解码器 12 中，当图形控制器 17 清除图形平面 8 时，流图形处理器 14 继续解码。该流水线处理的结果是，可以快速地显示图形。

[0262] 图 30 示出了在 ODS 解码之前图形平面 8 的清除结束的例子。另一方面，图 31 的时序图示出了 ODS 的解码在清除图形平面 8 之前结束的流水线处理。在这种情况下，ODS 解码结束之后，还不能把通过解码获得的图形写入图形平面 8。只有图形平面 8 的清除结束之后，才可以将该图形写入图形平面 8 中。

[0263] 图 32 的时序图示出了再现装置 200 中的缓冲器占用率的变化。在图中，第一至第四级分别示出了图形平面 8、对象缓冲器 15、编码数据缓冲器 13 和合成缓冲器 16 的占用率。这些变化是用线图的形式表示的，其中，水平轴表示时间，垂直轴表示占用率。

[0264] 第四级表示合成缓冲器 16 的占用率的变化。如图所示，合成缓冲器 16 的占用率的变化包括单调增加 Vf0，单调增加 Vf0 反映将从编码数据缓冲器 13 输出的 PCS 进行存储。

[0265] 第三级显示编码数据缓冲器 13 的占用率的变化。如图所示，数据缓冲器 13 的占用率的变化包括单调增加 Vf1 和 Vf2 以及单调减少 Vg1 和 Vg2，单调增加 Vf1 和 Vf2 反映 ODS1 和 ODS2 的存储，单调减少 Vg1 和 Vg2 反映流图形处理器 14 顺序地读取 ODS1 和 ODS2。单调增长 Vf1 和 Vf2 的斜率基于从传输缓冲器 4a 到编码数据缓冲器 13 的传输速率 Rx，而单调减少 Vg1 和 Vg2 是瞬时的，因为流图形处理器 14 的解码是瞬时执行的。也就是说，流图形处理器 14 瞬时地对每个 ODS 进行解码，并保存通过解码获得的解压缩图形。由于从流图形处理器 14 到对象缓冲器 15 的传输速率 Rd 是 128Mbps，所以，对象缓冲器 15 的占用率以 128Mbps 增加。

[0266] 第二级给出了对象缓冲器 15 的占用率的变化。如图所示，对象缓冲器 15 的占用率的变化包括单调增加 Vh1 和 Vh2，单调增加 Vh1 和 Vh2 反映将从流图形处理器 14 输出的 ODS1 和 ODS2 的图形对象进行存储。单调增加 Vh1 和 Vh2 的斜率基于从流图形处理器 14 到对象缓冲器 15 的传输速率 Rd。ODS1 和 ODS2 的解码时间分别对应于第三级中出现单调减少和第二级中出现单调增加的时间段。ODS 的 DTS 表示解码时间的开始，而 ODS 的 PTS 表示解码时间的结束。如果在 ODS 的 PTS 显示的时间之前将解压缩图形对象传输到对象缓冲器 15，则 ODS 的解码结束。在 ODS 的 PTS 所示时间之前将解压缩图形对象存储在对象缓冲器 15 中是很重要的。只要满足这一点，解码时间中的单调减少和单调增加不限于图 32 所示情形。

[0267] 第一级示出了图形平面 8 的占用率的变化。如图所示，图形平面 8 的占用率的变化包括单调增加 Vf3，单调增加 Vf3 反映将从对象缓冲器 15 输出的图形对象进行存储。单调增加 Vf3 的斜率基于从对象缓冲器 15 到图形平面 8 的传输速率 Rc。PCS 的 PTS 给出了单调增加 Vf3 的结束。

[0268] 通过使用 ODS 的 DTS 和 PTS、PCS 的 DTS 和 PTS 以及图 29 所示的缓冲器尺寸和传输速率，可以创建图 32 所示的图。该图使得当再现 BD-ROM 100 上的 AV 剪辑时创作者能够了解缓冲器状态如何改变。

[0269] 通过重写 DTS 和 PTS，可以调整缓冲器状态的这些变化。因此，创作者可以防止解码负载超过再现装置的解码器的规格，或者，防止再现期间的缓冲器溢出。这使得当开发再现装置 200 时硬件和软件的实现更容易。对再现装置 200 的内部构造的解释到此结束。

[0270] 下面说明如何实现图形解码器 12。通过让一个通用 CPU 执行用于实现图 33 所示操作的程序，可以实现图形解码器 12。下面结合图 33，描述图形解码器 12 的操作。

[0271] 图 33 的流程图示出了功能段的加载操作。在图中，SegmentK 是一个变量，表示一

个属于 DS 并且在再现 AV 剪辑期间被读取的 Segment(PCS、WDS、PDS 或 ODS)，忽略标志表示是忽略 SegmentK 还是加载 SegmentK。在该流程图中，在将忽略标志复位为 0 之后 (S20)，对于每个 SegmentK，执行步骤 S21 到 S24 和 S27 到 S31 的循环 (S25 和 S26)。

[0272] 步骤 S21 判断 SegmentK 是不是 PCS。如果 SegmentK 是 PCS，则操作转入步骤 S27。

[0273] 步骤 S22 判断忽略标志是 0 还是 1。如果忽略标志是 0，则操作转入步骤 S23。如果忽略标志是 1，则操作转入步骤 S24。在步骤 S23 中，将 SegmentK 加载到编码数据缓冲器 13 中。

[0274] 如果忽略标志是 1(S22 :否)，则忽略 SegmentK(S24)。这导致在步骤 S22 中对属于该 DS 的所有功能段的否定判断，因此，忽略 DS 的所有功能段。

[0275] 因此，忽略标志指示要忽略或加载 SegmentK。执行步骤 S27 至 S31 和 S34 至 S35，从而设置该忽略标志。

[0276] 步骤 S27 判断 PCS 的 composition_state 字段是不是 AcquisitionPoint。如果 composition_state 字段是 Acquisition Point，则操作转入步骤 S28。如果 composition_state 字段是 Epoch Start 或 Normal Case，则操作转入步骤 S31。

[0277] 步骤 S28 判断前一 DS 是否存在图形解码器 12 的某一缓冲器中 (编码数据缓冲器 13、流图形处理器 14、对象缓冲器 15 和合成缓冲器 16)。如果执行跳节操作的话，则前面的 DS 不存在于图形解码器 12 中。在这种情况下，需要从 Acquisition Point Ds 开始显示，从而使操作转入步骤 S30(S28 :否)。

[0278] 在步骤 S30 中，将忽略标志设置为 0，于是操作转入步骤 S22。

[0279] 另一方面，如果执行正常再现的话，则前面的 DS 存在图形解码器 12 中。在这种情况下，操作转入步骤 S29(S28 :是)。在步骤 S29 中，将忽略标志设置为 1，于是操作转入步骤 S22。

[0280] 步骤 S31 判断 composition_state 字段是否为 Normal Case。如果 composition_state 字段是 Normal Case，则操作转入步骤 S34。如果 composition_state 字段为 Epoch Start，则操作转入步骤 S30，其中，将忽略标志设置为 0。

[0281] 步骤 S34 与步骤 S28 相同，判断前面的 DS 是否存在图形解码器 12 中。如果前面的 DS 是否存在，则将忽略标志设置为 0(S30)。否则，则将忽略标志设置为 1，因为无法获得足以合成一幅画面图形的的功能段 (S35)。这样，当前面的 DS 不存在于图形解码器 12 中时，忽略 Normal Case DS 的功能段。

[0282] 下面结合图 34 给出加载 DS 的具体例子。在图 34 中，将三个 DS(DS0、DS1 和 DS2) 与视频进行复用。DS1 的 composition_state 字段是 Epoch Start，DS10 的 composition_state 字段是 Acquisition Point，DS20 的 composition_state 字段是 Normal Case。

[0283] 假设对复用了这三个 DS 和视频的 AV 剪辑中的图片数据 pt10 执行跳节操作，如箭头 am1 所示。在这种情况下，最接近 pt10 的 DS10 受到图 33 所示操作的影响。DS10 的 composition_state 字段是 Acquisition Point(S27 :是)，但是，前面的 DS(DS1) 不存在于编码数据缓冲器 13 中 (S28 :否)。因此，将忽略标志设置为 0(S30)。从而，将 DS10 加载到编码数据缓冲器 13 中，如图 35 中的箭头 md1 所示。另一方面，假设对 DS10 之后的图片数据执行跳节操作，如图 34 中的箭头 am2 所示。在这种情况下，DS20 是 Normal Case DS，之前的 DS(DS10) 不存在于编码数据缓冲器 13 中。因此，忽略 DS20，如图 35 中的箭头 md2 所

示。

[0284] 图 37 示出了当执行如图 36 所示的正常再现时如何加载 DS1、DS10 和 DS20。在这三个 DS 中，DS1 是一个 Epoch Start DS，被加载到编码数据缓冲器 13 中，如箭头 rd1 所示 (S23)。但是，对于作为 Acquisition Point DS 的 DS10，忽略标志被设置成 1(S29)，所以，DS10 的功能段不会被加载到编码数据缓冲器 13 中，而是被忽略，如箭头 rd2 所示 (S24)。同时，作为 Normal Case DS 的 DS20 被加载到编码数据缓冲器 13 中，如箭头 rd3 所示 (S23)。

[0285] 下面解释图形控制器 17 的操作。图 38 至 40 的流程图示出了图形控制器 17 的操作。

[0286] 步骤 S41 至 S44 构成一个主程序，其中，等待由步骤 S41 到 S44 中任意之一所表示的事件。

[0287] 在图 38 中，步骤 S41 判断当前再现时间是否为 PCS 的 DTS。如果是，则执行步骤 S45 至 S53。

[0288] 步骤 S45 判断 PCS 的 composition_state 字段是不是 Epoch Start。如果是，则在步骤 S46 中清除整个图形平面 8。否则，在步骤 S47 中清除由 WDS 的 window_horizontal_position 字段、window_vertical_position 字段、window_width 字段和 window_height 字段表示的窗口。

[0289] 在步骤 S46 至 S47 之后执行步骤 S48，判断是否已经超过任一 ODSx 的 PTS。清除整个图形平面 8 需要花费很长时间，所以，在清除整个图形平面 8 之前，ODSx 的解码可能已经结束。步骤 S48 检查这种可能性。如果没有超过 ODSx 的 PTS，则操作返回到主程序。如果超过了 ODSx 的 PTS，则执行步骤 S49 至 S51。步骤 S49 判断 object_cropped_flag 字段是否为 0。如果是，则将与 ODSx 对应的图形对象设置为不显示 (S50)。

[0290] 如果 object_cropped_flag 字段是 1，则将基于 objetct_cropping_horizontal_position 字段、object_cropping_vertical_position 字段、cropping_width 字段和 cropping_height 字段剪裁的图形对象写入图形平面 8 上的窗口中，所在位置用 object_horizontal_position 字段和 object_vertical_position 字段表示 (S51)。这样，就将图形对象写入了窗口。

[0291] 步骤 S52 判断是否超过另一 ODS(ODSy) 的 PTS。如果在将 ODSx 的图形对象写入图形平面 8 的期间完成了 ODSy 的解码，则将 ODSy 设置为 ODSx(S53)，然后，操作返回步骤 S49。因此，对 ODSy 执行步骤 S49 至 S51。

[0292] 在图 39 中，步骤 S42 判断当前再现时间是否为 WDS 的 PTS。如果是，则操作转入步骤 S54，判断窗口的数量是否为 1。如果窗口的数量为 2，则操作返回主程序。如果窗口的数量为 1，则执行步骤 S55 至 S59 的循环。在该循环中，对于窗口中要显示的最多两个图形对象中的每一个，执行步骤 S57 至 S59。步骤 S57 判断 object_cropped_flag 是否为 0。如果是，则将图形对象设置为不显示 (S58)。

[0293] 如果 object_cropped_flag 是 1，则将基于 object_cropping_horizontal_position 字段、object_cropping_vertical_position 字段、cropping_width 字段和 cropping_height 字段剪裁的图形对象写入图形平面 8 上的窗口中，所在位置用 object_horizontal_position 字段和 object_vertical_position 字段表示 (S59)。该循环之后，一个或多个图形对象被写入窗口中。

[0294] 步骤 S44 判断当前再现时间是否为 PCS 的 PTS。如果是，则操作转入步骤 S60，以判断 palette_update_flag 字段是否为 1。如果是，则将 palette_id 字段标识的调色板发送到 CLUT 单元 9(S61)。如果 palette_update_flag 字段为 0，则跳过步骤 S61。

[0295] 此后，CLUT 单元对图形平面 8 上的图形执行颜色转换。然后，将图形覆盖到视频上 (S62)。

[0296] 在图 40 中，步骤 S43 判断当前时间是否为 ODS 的 PTS。如果是，则操作转入步骤 S63，以判断窗口数量是否为 2。如果窗口数量为 1，则操作返回主程序。

[0297] 这里，步骤 S43 和 S63 中做出的判断具有以下意思。如果窗口数量为 2，则分别在两个窗口中显示两个图形对象。在这种情况下，每次 ODS 解码结束时，需要将通过解码得到的图形对象写入图形平面 8(参见图 19B)。因此，如果当前时间是 ODS 的 PTS 并且窗口数量是 2，则执行步骤 S64 至 S66，以将每个图形对象写入图形平面 8 中。步骤 S64 判断 object_cropped_flag 字段是否为 0。如果是，则将图形对象设置为不显示 (S65)。

[0298] 如果 object_cropped_flag 字段是 1，则将基于 object_cropping_horizontal_position 字段、object_cropping_vertical_position 字段、cropping_width 字段和 cropping_height 字段剪裁的图形对象写入图形平面 8 上的窗口中，所在位置用 object_horizontal_position 字段和 object_vertical_position 字段表示 (S66)。重复该处理，从而将两个图形对象分别写入这两个窗口。

[0299] 根据该实施例，在前面紧邻的 DS 中 PCS 的活动周期内开始一个 DS 的处理。换句话说，无需等待前面紧邻的 DS 中 PCS 的活动周期结束，就可以开始 DS 的处理。DS 的处理开始的定时是在前面紧邻的 DS 中 PCS 的活动周期内，前面紧邻的 DS 的图形的解码和传送完成的时候。因此，DS 的处理可以提前一个时间周期，其中该时间周期从前面紧邻的 DS 的图形的解码和传送完成到前面紧邻的 DS 中 PCS 的活动周期结束。

[0300] 即使当以此方式，在前面紧邻的 DS 中 PCS 的活动周期内开始一个 DS 的处理时，DS 的图形对象写入对象缓冲器的时间周期也不与前面紧邻的 DS 的图形对象写入对象缓冲器的时间周期重叠。因此，只要将可以同时读取和写入的双端口存储器用作为对象缓冲器，就可以用单个流图形处理器在流水线中处理两个或更多 DS。此类流水线处理提高解码效率，不会使再现装置 200 的内部结构复杂化。

[0301] (第二个实施例)

[0302] 本发明的第二个实施例涉及第一个实施例中介绍的 BD-ROM100 的制造过程。图 41 是 BD-ROM 100 的制造过程的流程图。

[0303] 该制造过程包括 :材料生产步骤，用于记录视频、声音等 (S201) ;创作步骤，使用创作设备创建应用格式 (S202) ;印制 (pressing) 步骤，用于创建 BD-ROM 100 的原始主盘 (original master)，并执行压制 (stamping) 和粘合 (bonding)，以完成 BD-ROM 100(S203)。

[0304] 在该制造过程中，创作步骤包括步骤 S204 至 S213。

[0305] 在步骤 S204 中，生成控制信息、窗口定义信息、调色板定义信息和图形。在步骤 S205 中，将控制信息、窗口定义信息、调色板定义信息和图形转换成功能段。在步骤 S206 中，基于要同步图片的时间，设置每个 PCS 的 PTS。在步骤 S207 中，基于 PTS[PCS] 设置 DTS[ODS] 和 PTS[ODS]。在步骤 S208 中，基于 DTS[ODS]，设置 DTS[PCS]、PTS[PDS]、DTS[WDS]

和 PTS[WDS]。在步骤 S209 中，用图表示播放机模型中每个缓冲器的占用率的变化。在步骤 S210 中，判断图示的变化是否满足播放机模型的约束条件。如果判断结果是否定的，则在步骤 S211 中，重写每个功能段的 DTS 和 PTS。如果判断结果是肯定的，则在步骤 S212 中生成图形流，并在步骤 S213 中将图形流与视频流和音频流进行复用，以形成 AV 剪辑。此后，将 AV 剪辑改编为蓝光只读光盘格式，从而完成该应用格式。

[0306] （修改）

[0307] 尽管通过上面的实施例对本发明进行了描述，但本发明不限于此。也可以用下面的修改 (A) 至 (P) 中任意之一来实现本发明。本申请的每项权利要求的发明都包括上述实施例的扩展和概括以及下面的修改。扩展和概括程度取决于做出本发明时本发明技术领域的技术发展水平。

[0308] (A) 在上述实施例描述的情形中，将 BD-ROM 用作记录介质。但是，本发明的主要特征在于记录介质上记录的图形流，而这并不依赖于 BD-ROM 的物理特性。因此，本发明同样适用于能够记录图形流的任何记录介质。这种记录介质的示例包括：光盘，如 DVD-ROM、DVD-RAM、DVD-RW、DVD-R、DVD+RW、DVD+R、CD-R 或 CD-RW；磁光盘，如 PD 和 MO；半导体存储卡，如 CF 卡、Smartmedia 卡、记忆棒、多媒体卡或 PCM-CIA 卡；磁记录盘，如软盘、SuperDisk、Zip 和 Click！；可移动硬盘，如 ORB、Jaz、SparQ、SyJet、EZFley 和 Microdrive；以及不可移动硬盘。

[0309] (B) 在上述实施例描述的情形中，再现装置对 BD-ROM 上的 AV 剪辑进行解码，并将解码后的 AV 剪辑输出到电视。当然，再现装置也可以只有 BD 驱动器，其他的组成部件由电视来提供。在这种情况下，可以将再现装置和电视集成到通过 IEEE 1394 连接器相连的家庭网络中。

[0310] 在上述实施例描述的情形中，再现装置连接到电视，但是，再现装置也可以与显示装置集成在一起。此外，再现装置也可以只包括系统 LSI（集成电路），其构成处理的实质性部分。该再现装置和该集成电路都是本说明书所描述的发明。因此，不管涉及再现装置还是涉及集成电路，基于第一个实施例中描述的再现装置的内部构造来制造再现装置的行为都是实施本发明的一种行为。此外，有偿转让（即销售）或无偿转让（即，作为礼物）、出租和进口该再现装置也是实施本发明的行为。同样，使用橱窗展示、商品目录或宣传册来许诺转让或出租该再现装置的行为也是实施本发明的行为。

[0311] (C) 使用上述流程图所示的程序的信息处理实际上是用硬件资源实现的。因此，描述这些流程图中所示操作过程的程序本身也是一项发明。在上述实施例所描述的情形中，程序是集成到再现装置中的，但是，这些程序的使用也可以独立于再现装置。实施这些程序的行为包括：(1) 制造行为；(2) 有偿或无偿转让的行为；(3) 出租行为；(4) 进口行为；(5) 经由双向电子通信网络提供给公众的行为；(6) 使用橱窗展示、商品目录或宣传册许诺销售或出租的行为。

[0312] (D) 在每个流程图中以时间序列执行的步骤的时间单元可被视为本发明的必要单元。因此，这些流程图所示的再现方法是一项发明。如果执行每个流程图中所示的处理，通过以时间顺序执行这些步骤，从而实现预期的目的和预期的效果，那么，这也是实施本发明的记录方法的行为。

[0313] (E) 当把 AV 剪辑记录到 BD-ROM 上时，可以向 AV 剪辑中的各个 TS 包添加扩展

头。该扩展头被称为 TP_extra_header，其包括 arrival_time_stamp 和 copy_permission_indicator，并且数据长度为 4 个字节。具有 TP_extra_header 的 TS 数据包（下文称之为“EX TS 包”）以 32 个数据包为单位进行分组，并且将每组写入 3 个扇区中。由 32 个 EX TS 数据包构成的一组具有 6144 个字节（= 32x192），这相当于 3 个扇区的尺寸（6144 个字节 = (2048x3)）。包含在这 3 个扇区中的这 32 个 EX TS 数据包被称为一个对准单元 (aligned unit)。

[0314] 在通过 IEEE1394 连接器相连的家庭网络中，该再现装置以下面的方式发送对准单元。该再现装置将对准单元中的 32 个 EX TS 数据包中的每一个都删除 TP_extra_header，并且根据 DTCP 规范加密各 TS 数据包的主体，然后输出加密的 TS 数据包。当输出 TS 数据包时，该再现装置将一个同步包插入相邻的 TS 数据包之间。根据 TP_extra_header 的 arrival_time_stamp 所示的时间来确定插入同步包的位置。该再现装置输出 DTCP_descriptor 以及 TS 包。DTCP_descriptor 对应于 TP_extra_header 中的 copy_permission_indicator。通过提供表明“禁止复制”的 DTCP_descriptor，当在 IEEE 连接器连接的家庭网络中使用 TS 数据包时，可以防止将 TS 数据包记录到其他设备上。

[0315] (F) 在上述实施例所描述的情形中，将 BD-ROM 格式的 AV 剪辑用作数字流，但本发明还可以用 DVD 视频格式或 DVD 视频记录格式的 VOB（视频对象）来实现。VOB 是符合 ISO/IEC13818-1 标准的节目流，它是通过复用视频流和音频流而获得的。同样，AV 剪辑中的视频流也可以是 MPEG4 视频流或 WMV 视频流。此外，AV 剪辑中的音频流可以是线性 PCM 音频流、Dolby AC-3 音频流、MP3 音频流、MPEG-AAC 音频流或 dts 音频流。

[0316] (G) 上述实施例中的电影可以是通过对经过模拟广播传输的模拟图像信号进行编码而获得的。此外，该电影也可以是经过数字广播传输的传输流构成的流数据。

[0317] 或者，还可以通过对记录在录像带中的模拟 / 数字图像信号进行编码来获得内容。此外，还可以通过对从视频照相机中直接装载的模拟或数字图像信号进行编码来获得内容。这些内容也可以是通过分发服务器分发的数字作品。

[0318] (H) 上述实施例中描述的图形对象是游程编码的栅格数据。游程编码用于压缩和编码图形对象，因为游程编码适于压缩和解压缩字幕。字幕的属性在于，水平方向上的相同像素值的连续长度较长。因而，通过使用游程编码进行压缩，可以获得高压缩率。此外，游程编码降低了解压缩的负担，因此适于用软件实现解密。但是，对图形对象使用游程编码不是限制本发明。例如，图形对象可以是 PNG 数据。此外，图形对象不必是栅格数据，也可以是矢量数据。此外，图形对象可以是透明图案。

[0319] (I) 根据再现装置中的语言设置选择的字幕图形可以实现 PCS 的显示效果。因此，可以通过根据该再现装置的语言设置显示的字幕图形来实现通过使用在常规 DVD 中视频主体内包含的字符而达到的显示效果。这有助于提高实用性。

[0320] 此外，根据该再现装置的显示设置而选择的字幕图形可以实现 PCS 的显示效果。例如，BD-ROM 上记录了各种显示模式的图形，如宽屏幕、平移与扫描 (Pan and Scan，即 4:3 模式）和邮箱 (letterbox，即 16:9 模式），再现装置根据与再现装置连接的电视的显示设置来选择一种显示模式，并显示对应的图形。由于 PCS 的显示效果适用于这些图形，所以可视性得到了增强因此，用根据显示设置显示的字幕图形可以实现用传统 DVD 中视频主体内包含的字符达到的显示鲜果。这有助于提高实用性。

[0321] (J) 在第一个实施例描述的情形中，设置从对象缓冲器到图形平面的传输速率 Rc，从而在一个视频帧内，清除图形平面和在窗口上呈现图形，窗口尺寸是图形平面的 25％。但是，可以将 Rc 设置成在垂直消隐期间完成清除和呈现。假定垂直消隐周期是 1/29.93 秒的 25％。那么，Rc 是 1Gbps。如此设置 Rc，可以更流畅地显示图形。

[0322] 此外，可以与垂直消隐期间的写操作一起，使用与行扫描同步的写操作。这样，可以 Rc ＝ 256Mbps 流畅地显示字幕。

[0323] (K) 在上述实施例所描述的情形中，再现装置包括图形平面上。但是，还可以将用于储存一行解压缩像素的线缓冲器代替图形平面。因为对于每行执行到图像信号的转换，所以，同样可以用线缓冲器执行到图像信号的转换。

[0324] (L) 在上述实施例所描述的情形中，图形是表示电影中的对话的字符串，但是，本发明同样适用于与视频精确同步显示的任何图形。这样的图形示例包括插图、图案、卡通人物和字符标记。这些图形的示例还包括构成商标的图案、字符和颜色的组合、国家饰章、国旗、国徽、国家政府使用的公共标记或图章、国际组织的饰章、旗帜或徽章，或者特殊项目的原始标记。

[0325] (M) 在第一个实施例所描述的情形中，基于字幕在屏幕顶部或底部上水平地显示的假设，将窗口设置在屏幕的顶部或底部。但是，也可以将窗口设置在屏幕的左侧或右侧，从而在屏幕的左侧和右侧上垂直地显示字幕。这样，就可以垂直地显示日文字幕。

[0326] (O) 当 DSn 和 DSn+1 属于图形流中同一时元时，图形解码器对 DSn 和 DSn+1 执行流水线处理。另一方面，当 DSn 和 DSn+1 属于不同的时元时，在 DSn 的图形显示开始之后，图形解码器开始处理 DSn+1。

[0327] 此外，有两种图形流，即 :呈现图形流，主要用于与视频进行同步 ;交互图形流，主要用于实现交互显示。当该图形流是呈现图形流时，图形解码器对 DSn 和 DSn+1 执行流水线处理 ;当该图形流是交互图形流时，不执行流水线处理。

[0328] 可以如上所述修改本发明。但是，本申请的每项权利要求的发明都反映了解决传统技术所遇到的技术问题的手段，所以，根据权利要求书的本发明的技术范围不会超过本领域技术人员认识该技术问题的技术范围。因此，根据权利要求书的本发明基本上对应于本申请中的说明书部分。

[0329] 工业应用性

[0330] 上述实施例披露了本发明所涉及的记录介质和再现装置的内部构造，可以基于所披露的内部构造，批量制造记录介质和再现装置。换言之，该记录介质和再现装置能够在工业上制造出来。因此，该记录介质和再现装置具有工业应用性。

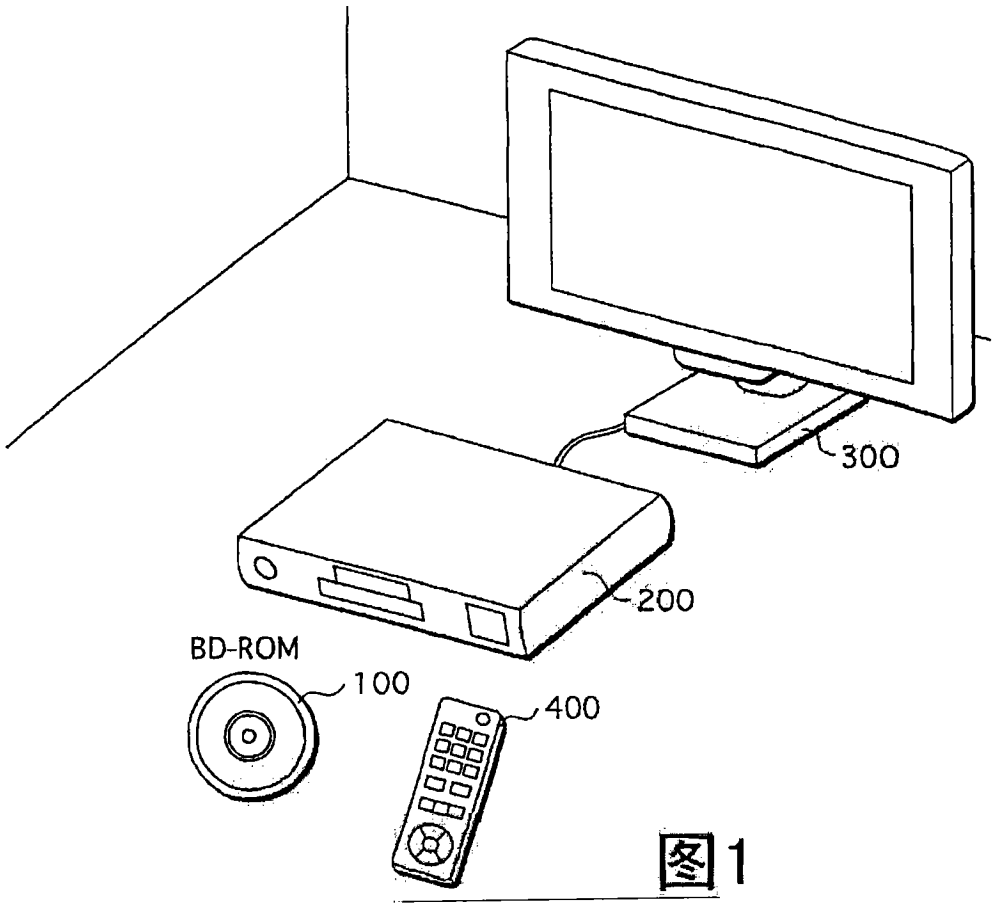


图1

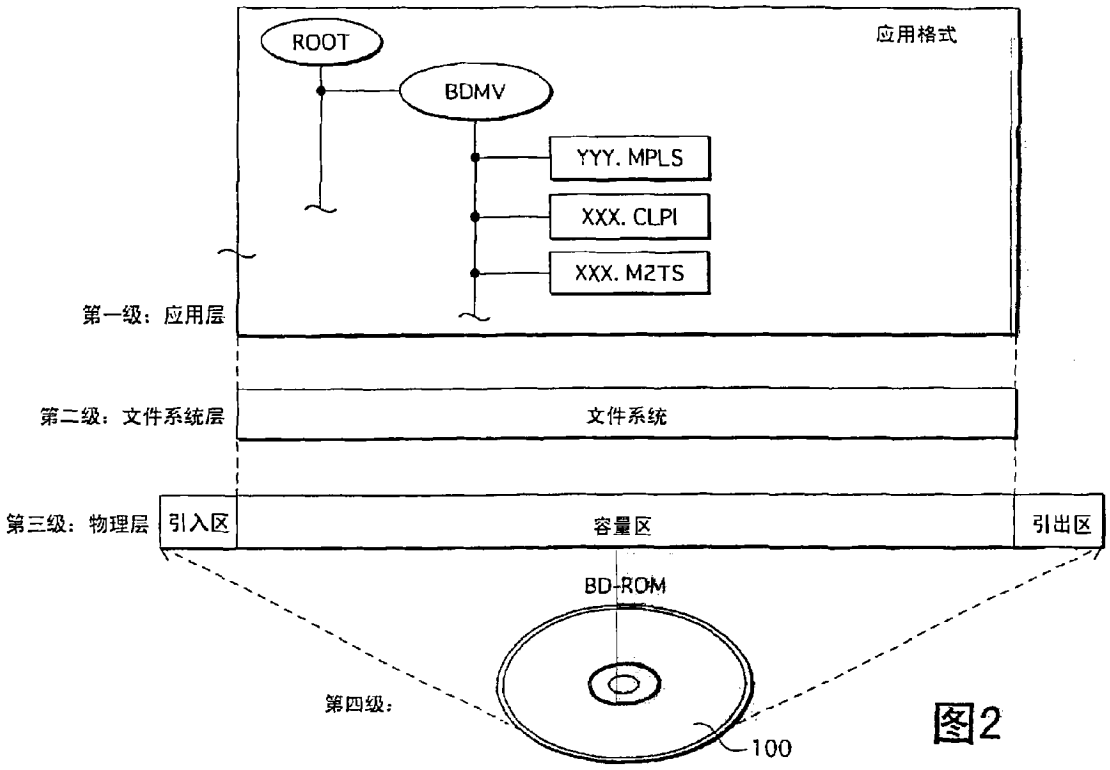

ROOT
BDMV
YYY. MPLS
XXX. CLPI
XXX. M2TS
应用格式
第一级：应用层
第二级：文件系统层
文件系统
第三级：物理层
引入区
容量区
引出区
BD-ROM
第四级：
100

图2

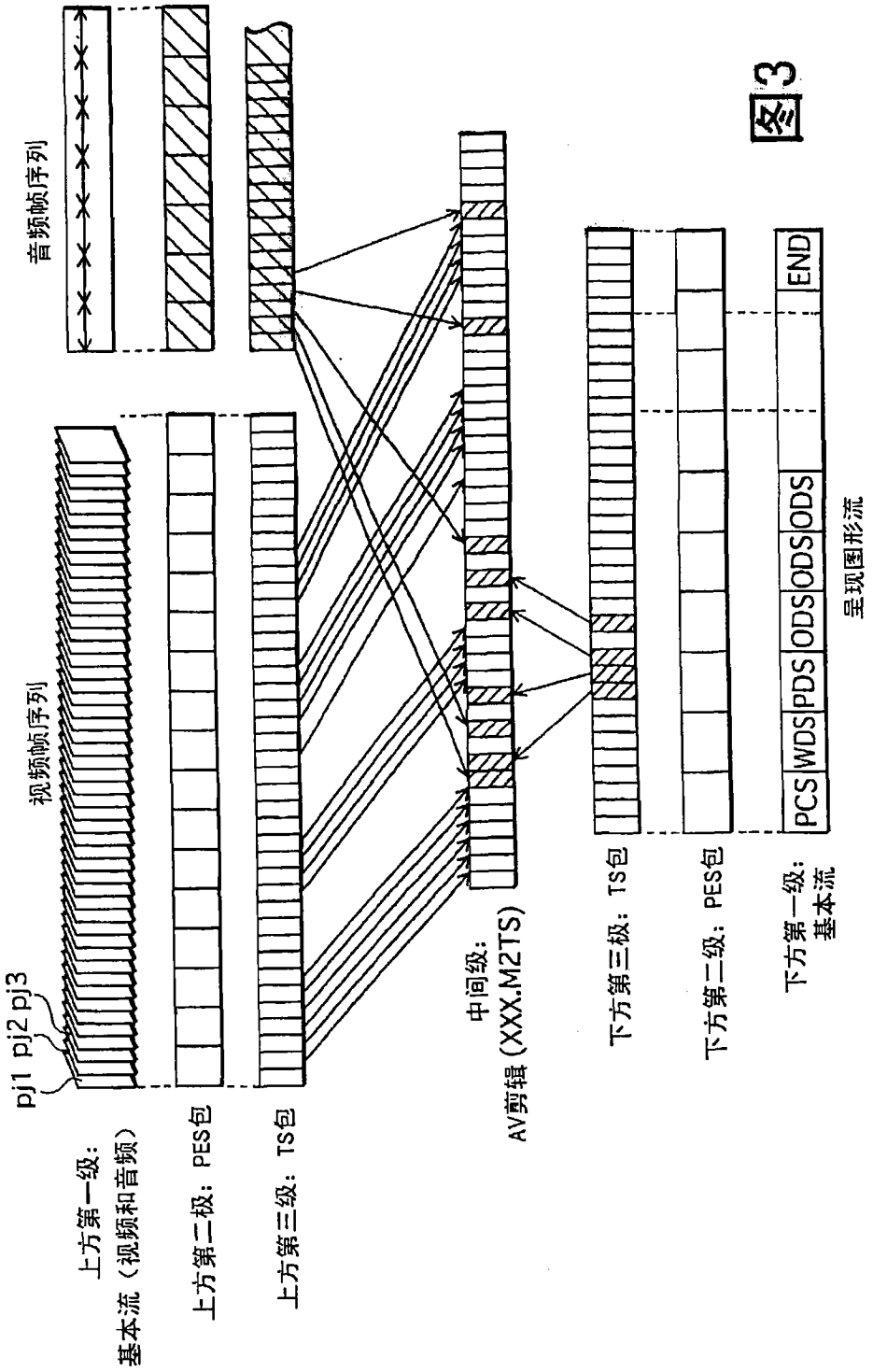

音频帧序列
视频帧序列
pj1 pj2 pj3
上方第一级：
基本流（视频和音频）
上方第二级：PES包
上方第三级：TS包
中间级：
AV剪辑（XXX.M2TS）
下方第三级：TS包
下方第二级：PES包
下方第一级：
基本流
PCS
WDS
PDS
ODS
ODS
ODS
END
呈现图形流

图3

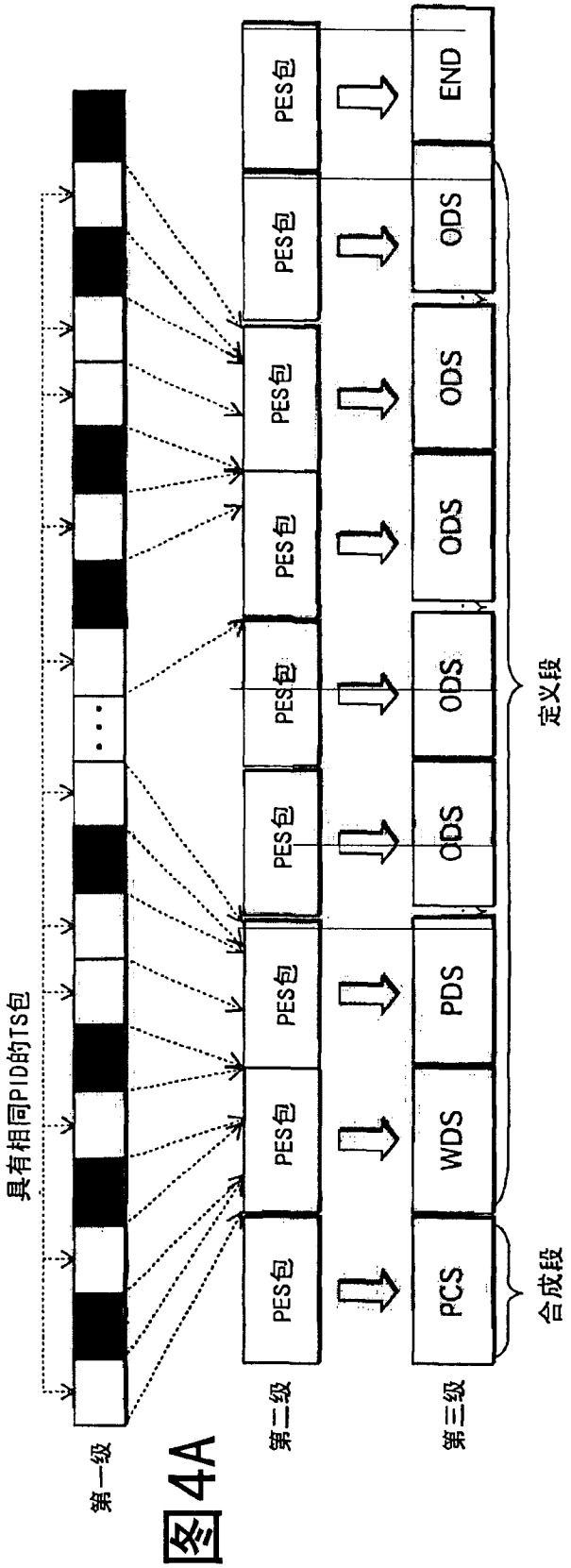

具有相同PID的TS包
第一级
第二级
第三级
PES包
PCS
WDS
PDS
ODS
END
定义段
合成段

图4A

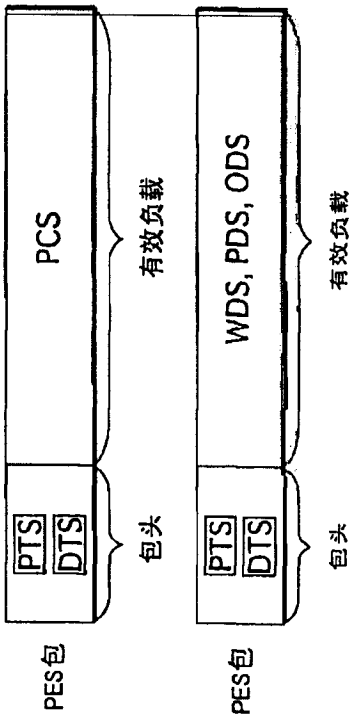

PES包
PTS
DTS
包头
PCS
有效负载
WDS, PDS, ODS

图4B

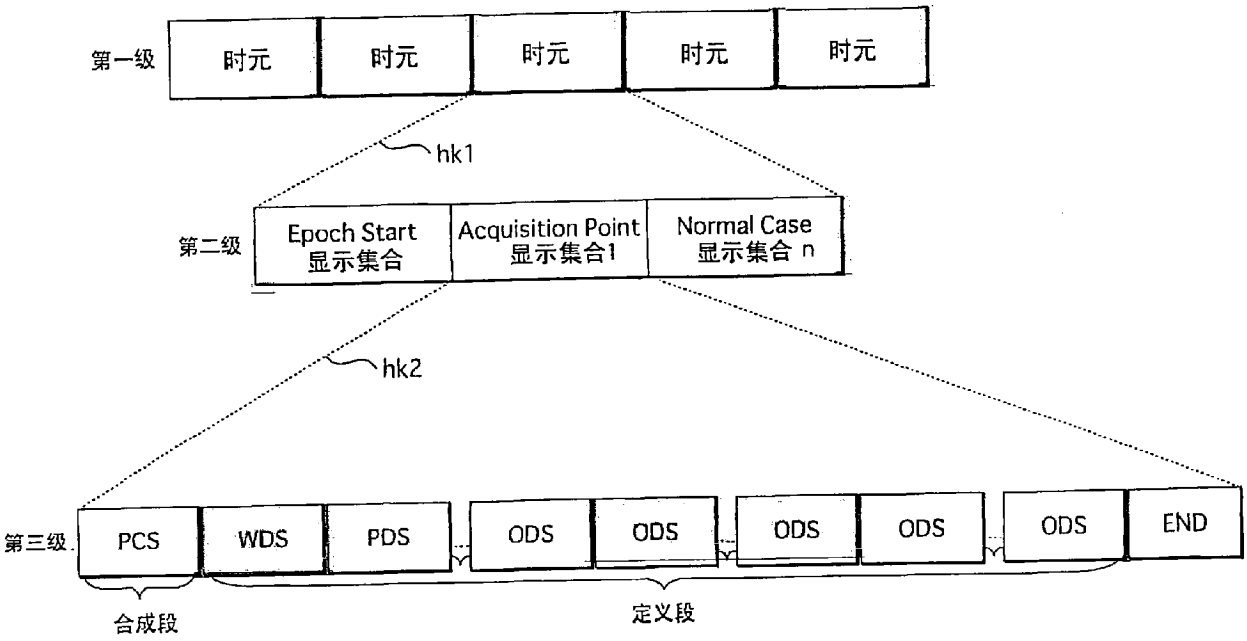

第一级
时元
时元
时元
时元
时元
hk1
第二级
Epoch Start
显示集合
Acquisition Point
显示集合1
Normal Case
显示集合 n
hk2
第三级
PCS
WDS
PDS
ODS
ODS
ODS
ODS
ODS
END
合成段
定义段

图5

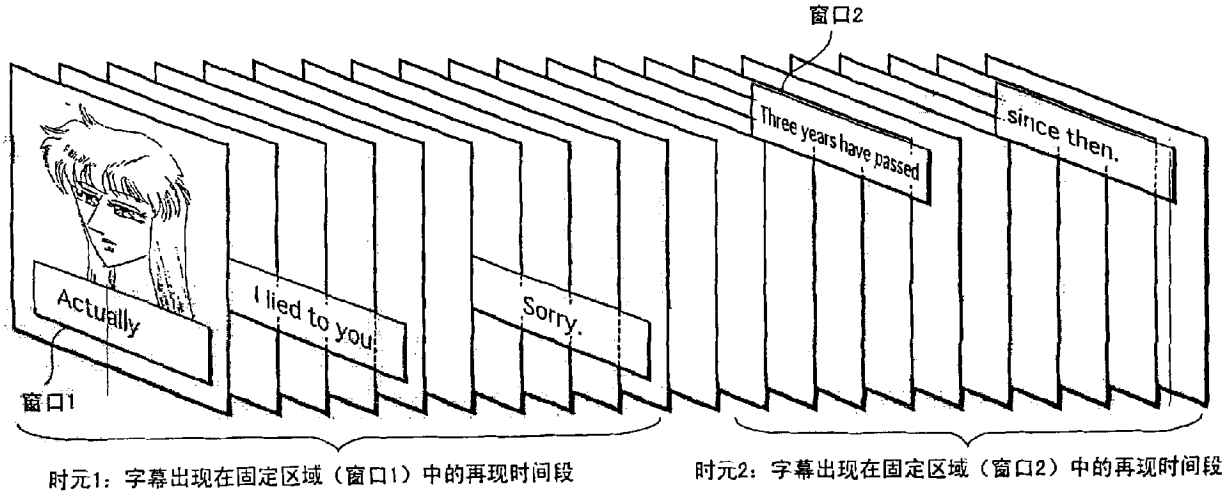

窗口2
Three years have passed
since then.
Actually
I lied to you.
Sorry.
窗口1
时元1：字幕出现在固定区域（窗口1）中的再现时间段
时元2：字幕出现在固定区域（窗口2）中的再现时间段

图6

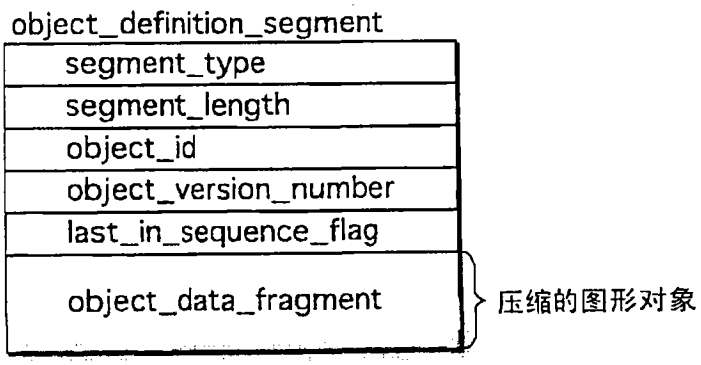

object_definition_segment
segment_type
segment_length
object_id
object_version_number
last_in_sequence_flag
object_data_fragment
压缩的图形对象

图7A

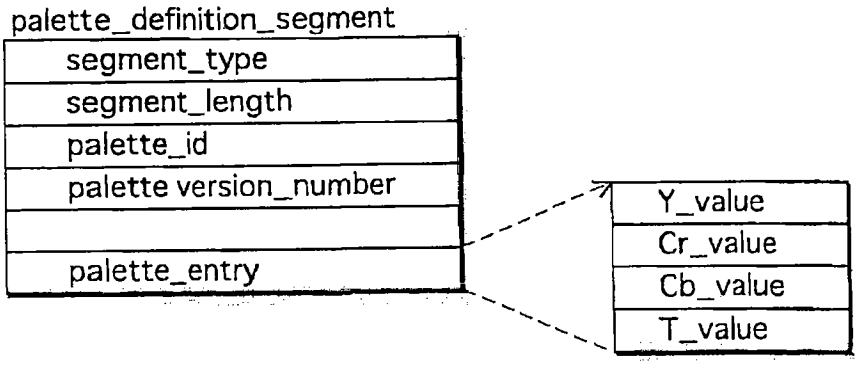

palette_definition_segment
segment_type
segment_length
palette_id
palette version_number
palette_entry
Y_value
Cr_value
Cb_value
T_value

图7B

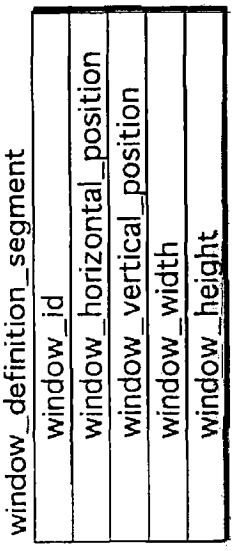

window_definition_segment
window_id
window_horizontal_position
window_vertical_position
window_width
window_height

图8A

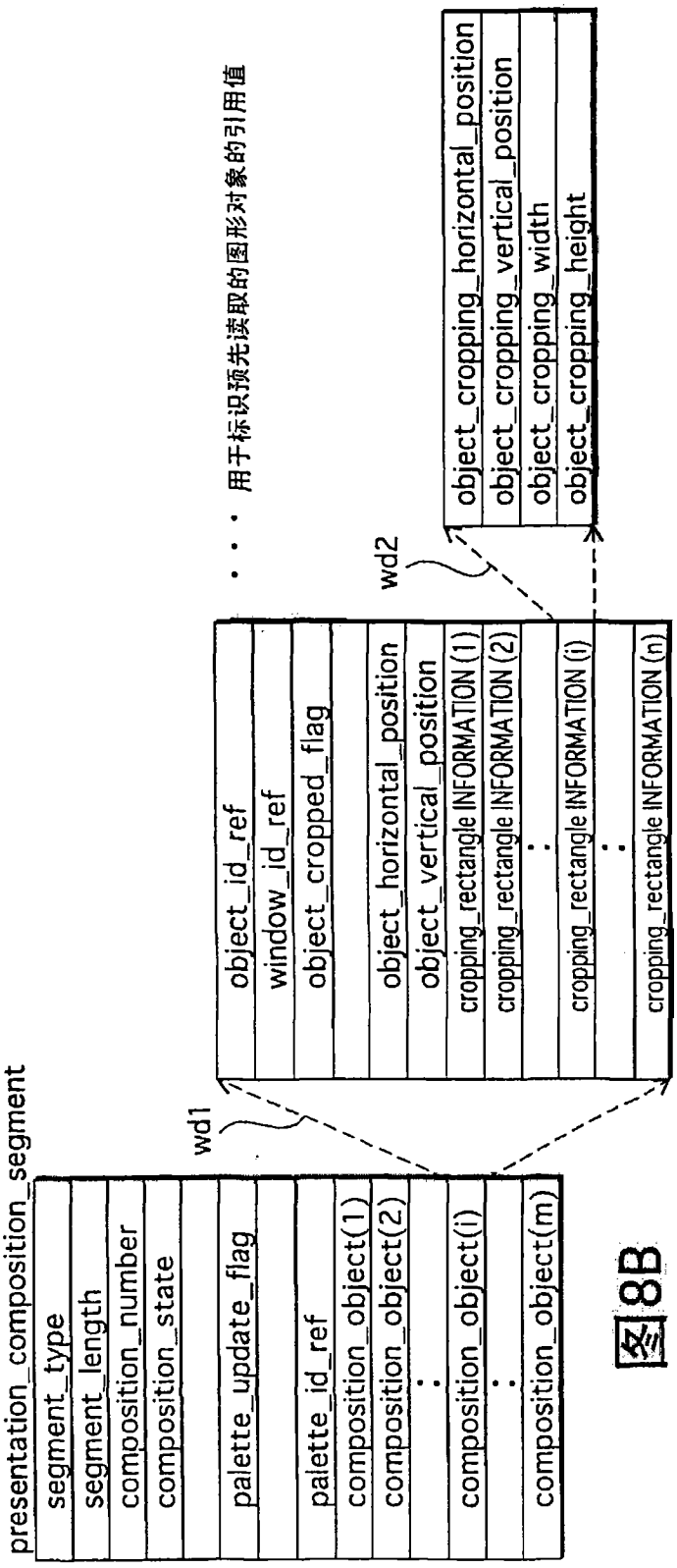

presentation_composition_segment
segment_type
segment_length
composition_number
composition_state
palette_update_flag
palette_id_ref
composition_object(1)
composition_object(2)
composition_object(i)
composition_object(m)
wd1
object_id_ref
window_id_ref
object_cropped_flag
object_horizontal_position
object_vertical_position
cropping_rectangle INFORMATION (1)
cropping_rectangle INFORMATION (2)
cropping_rectangle INFORMATION (i)
cropping_rectangle INFORMATION (n)
· · · 用于标识预先读取的图形对象的引用值
wd2
object_cropping_horizontal_position
object_cropping_vertical_position
object_cropping_width
object_cropping_height

图8B

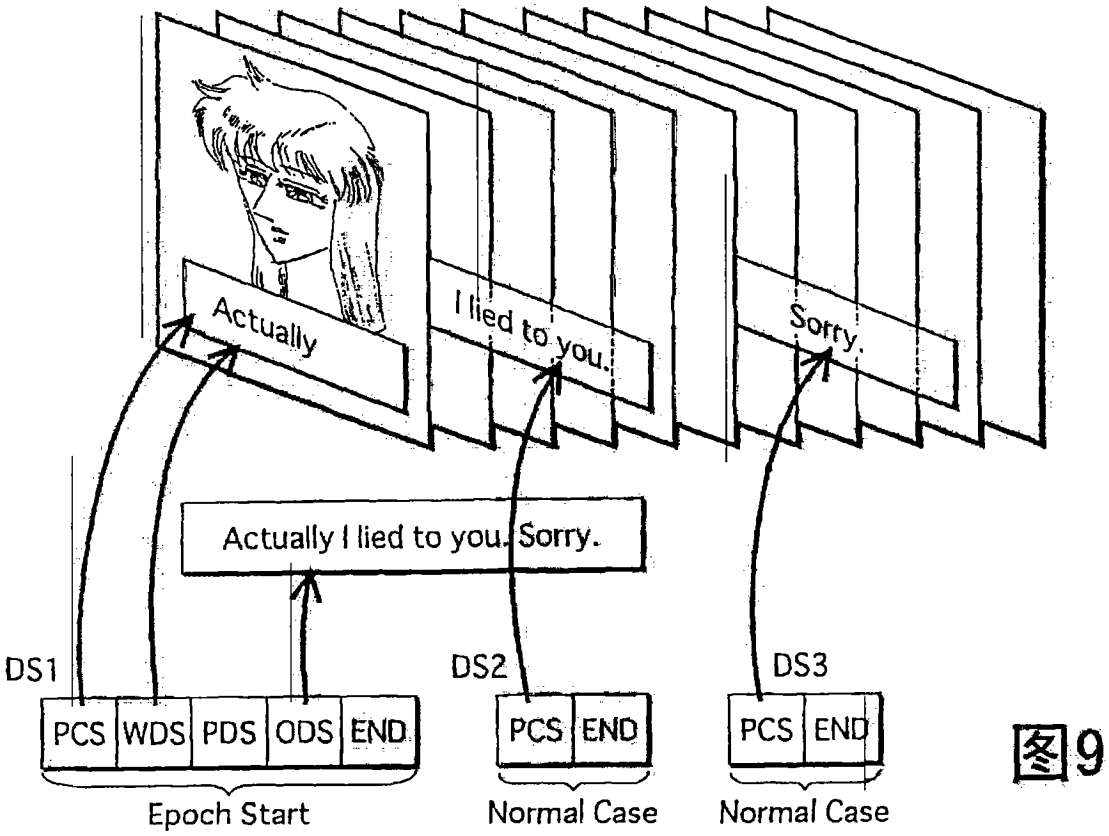

Actually
I lied to you.
Sorry.
Actually I lied to you. Sorry.
DS1
DS2
DS3
PCS
WDS
PDS
ODS
END
PCS
END
PCS
END
Epoch Start
Normal Case
Normal Case

图9

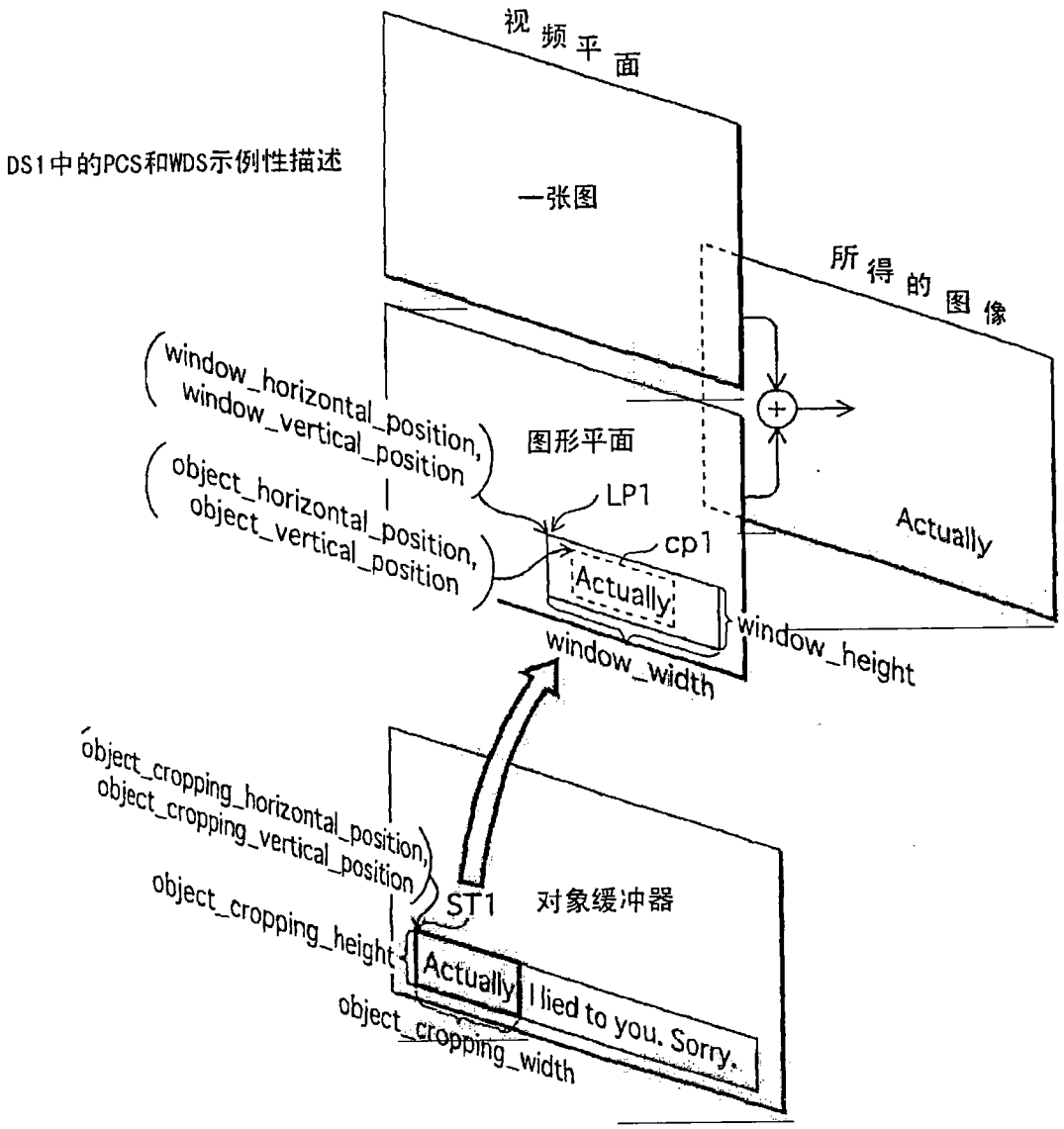

DS1中的PCS和WDS示例性描述
视频平面
一张图
所得的图像
window_horizontal_position,
window_vertical_position
图形平面
object_horizontal_position,
object_vertical_position
LP1
cp1
Actually
Actually
window_height
window_width
object_cropping_horizontal_position,
object_cropping_vertical_position
ST1
对象缓冲器
object_cropping_height
Actually
I lied to you. Sorry.
object_cropping_width

图10

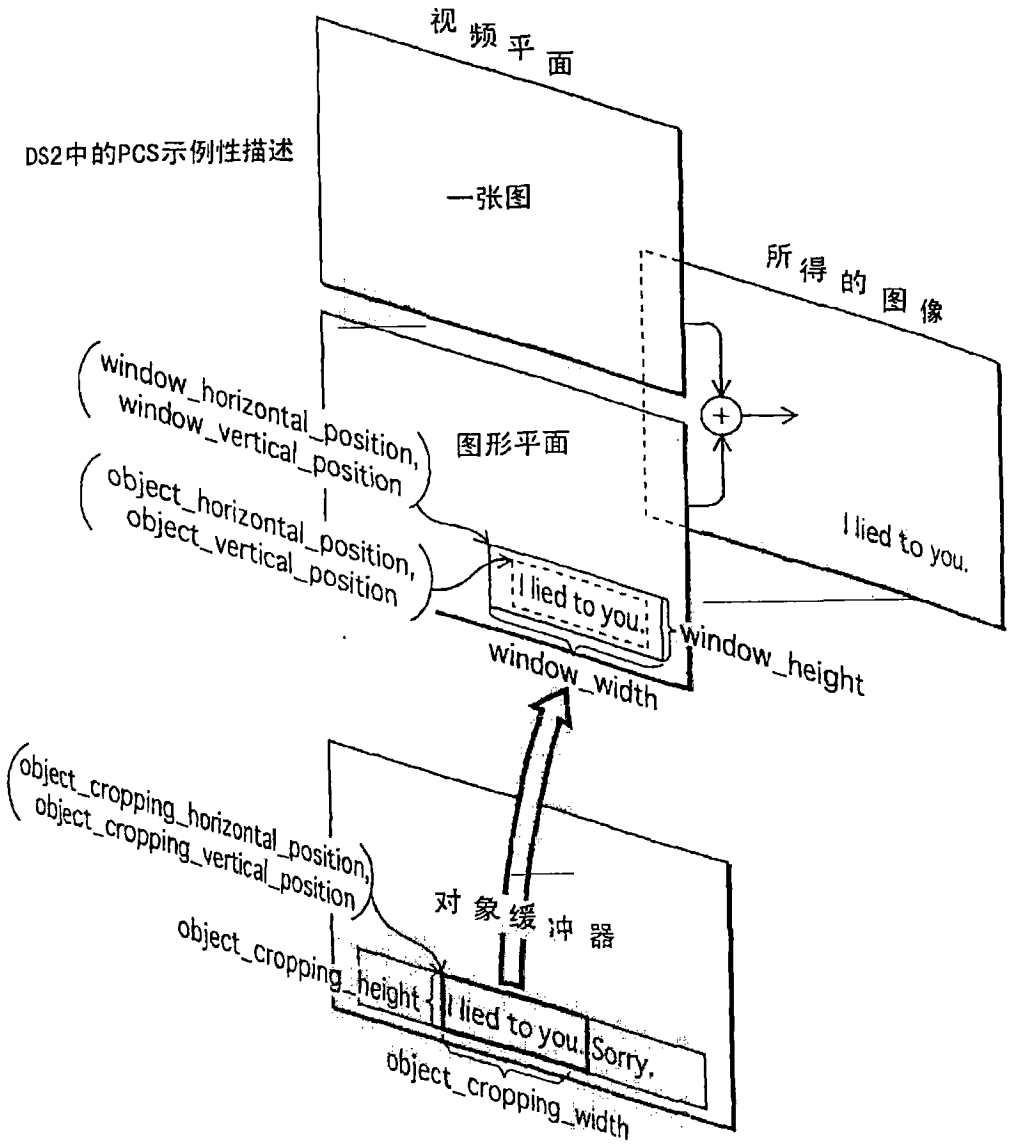

视频平面
DS2中的PCS示例性描述
一张图
所得的图像
window_horizontal_position,
window_vertical_position
图形平面
object_horizontal_position,
object_vertical_position
I lied to you.
I lied to you.
window_height
window_width
object_cropping_horizontal_position,
object_cropping_vertical_position
对象缓冲器
object_cropping_height
I lied to you.
Sorry.
object_cropping_width

图11

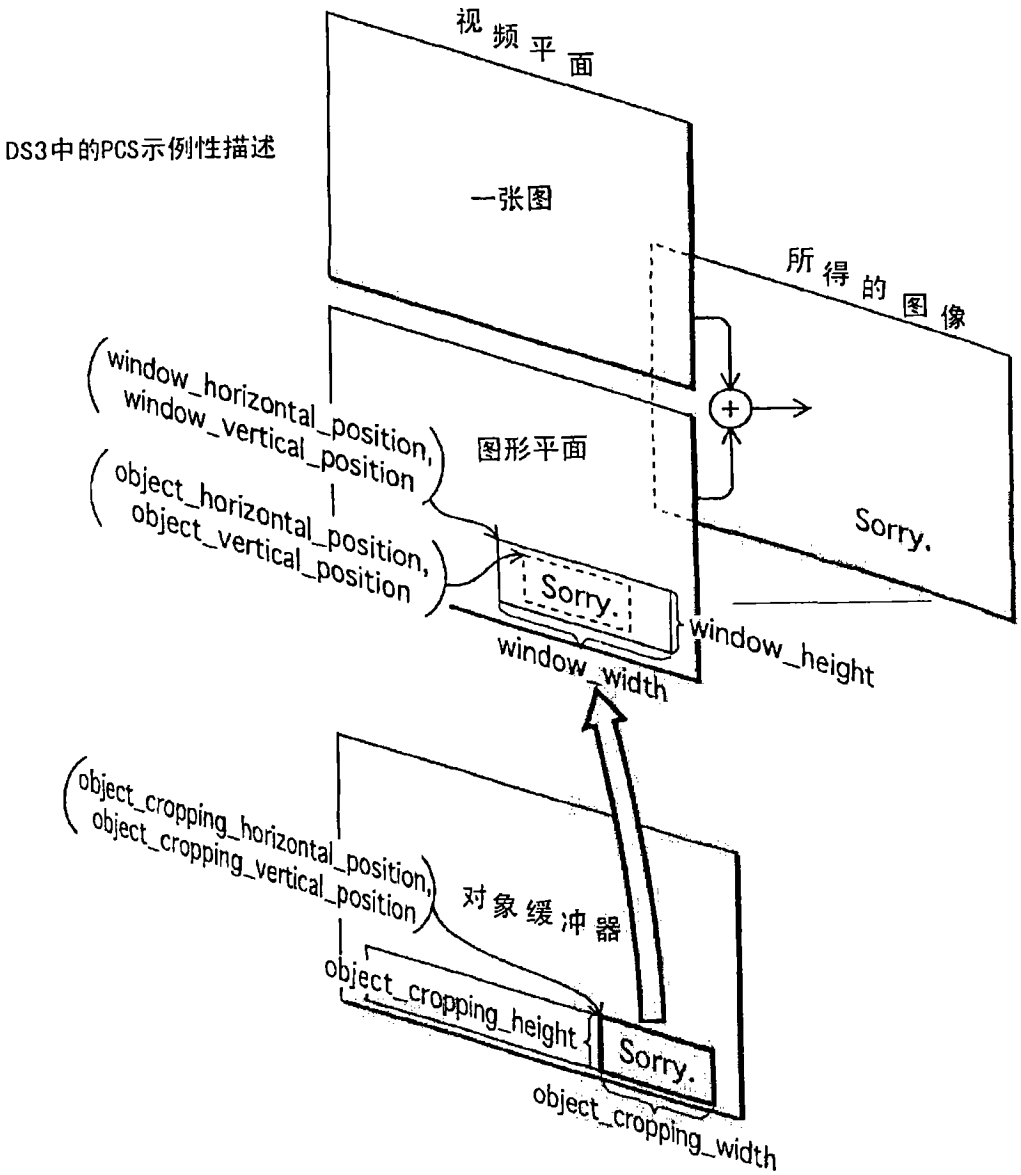

DS3中的PCS示例性描述
视频平面
一张图
所得的图像
window_horizontal_position,
window_vertical_position
图形平面
object_horizontal_position,
object_vertical_position
Sorry.
Sorry.
window_width
window_height
object_cropping_horizontal_position,
object_cropping_vertical_position
对象缓冲器
object_cropping_height
Sorry.
object_cropping_width

图12

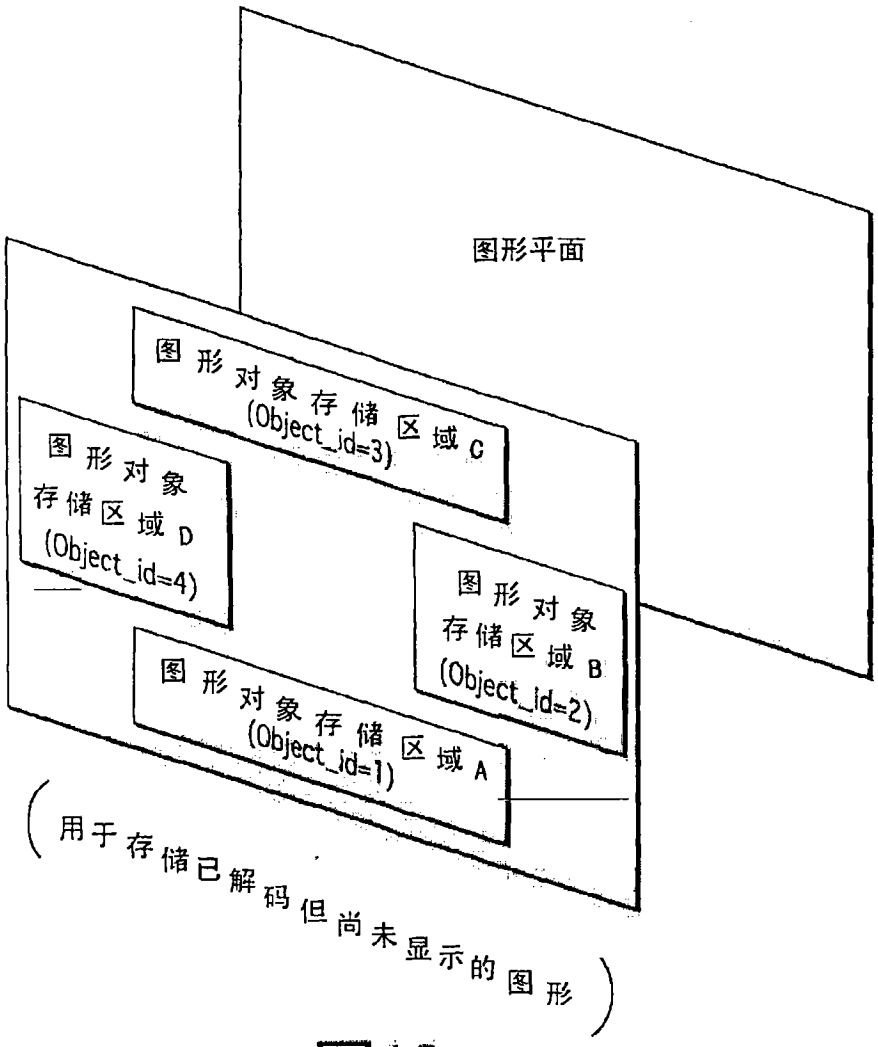

图形平面
图形对象存储区域C
(Object_id=3)
图形对象
存储区域D
(Object_id=4)
图形对象
存储区域B
(Object_id=2)
图形对象存储区域A
(Object_id=1)
用于存储已解码但尚未显示的图形

图13

```
        PTS( DSn[PCS)] )>=DTS( DSn[PCS] )+DECODEDURATION( DSn )
Where:
  •  DECODEDURATION( DSn ) is calculated as follows:
  decode_duration = 0 ;
  decode_duration += PLANEINITIALIZATIONTIME( DSn ) ;
  if( DSn. PCS. num_of_objects == 2 )
  {
      decode_duration += WAIT( DSn, DSn. PCS. OBJ[0], decode_duration ) ;
      if( DSn. PCS. OBJ[0]. window_id == DSn. PCS. OBJ[1]. window_id )
      {
              decode_duration += WAIT( DSn, DSn. PCS. OBJ[1], decode_duration ) ;
              decode_duration += 90000*( SIZE( DSn. PCS. OBJ[0]. window_id )//256*10^6 ) ;
      }
      else
      {
              decode_duration += 90000*( SIZE( DSn. PCS. OBJ[0]. window_id )//256*10^6 ) ;
              decode_duration += WAIT( DSn, DSn. PCS. OBJ[1], decode_duration ) ;
              decode_duration += 90000*( SIZE( DSn. PCS. OBJ[1]. window_id )//256*10^6 ) ;
      }
  }
  else if( DSn. PCS. num_of_objects ==1 )
  {
      decode_duration += WAIT( DSn, DSn. PCS. OBJ[0], decode_duration ) ;
      decode_duration += 90000*( SIZE( DSn. PCS. OBJ[0]. window_id )//256*10^6 ) ;
  }
  return decode_duration ;

  •  PLANEINITIALIZATIONTIME( DSn ) is calculated as follows:
  initialize_duration=0 ;
  if( DSn. PCS. composition_state= = EPOCH_START )
  {
    initialize_duration = 90000*( 8*video_width*video_height//256*10^6) ;
  }
  else
  {
      for( i=0 ; i < WDS. num_windows ; i++ )
      {
              if( EMPTY(DSn.WDS.WIN[i],DSn ) )
                    initialize_duration += 90000*( SIZE( DSn. WDS. WIN[i] )//256*10^6) ;
      }
  }
  return initialize_duration ;

  •  WAIT( DSn, OBJ, current_duration ) is calculated as follows:
  wait_duration = 0 ;
  if( EXISTS( OBJ. object_id, DSn ) )
  {
      object_definition_ready_time = PTS( GET( OBJ. object_id. DSn ) ) ;
      current_time = DTS( DSn. PCS )+current_duration ;
      if( current_time < object_definition_ready_time )
              wait_duration += object_definition_ready_time - current_time ) ;
  }
  return wait_duration ;
```

图14

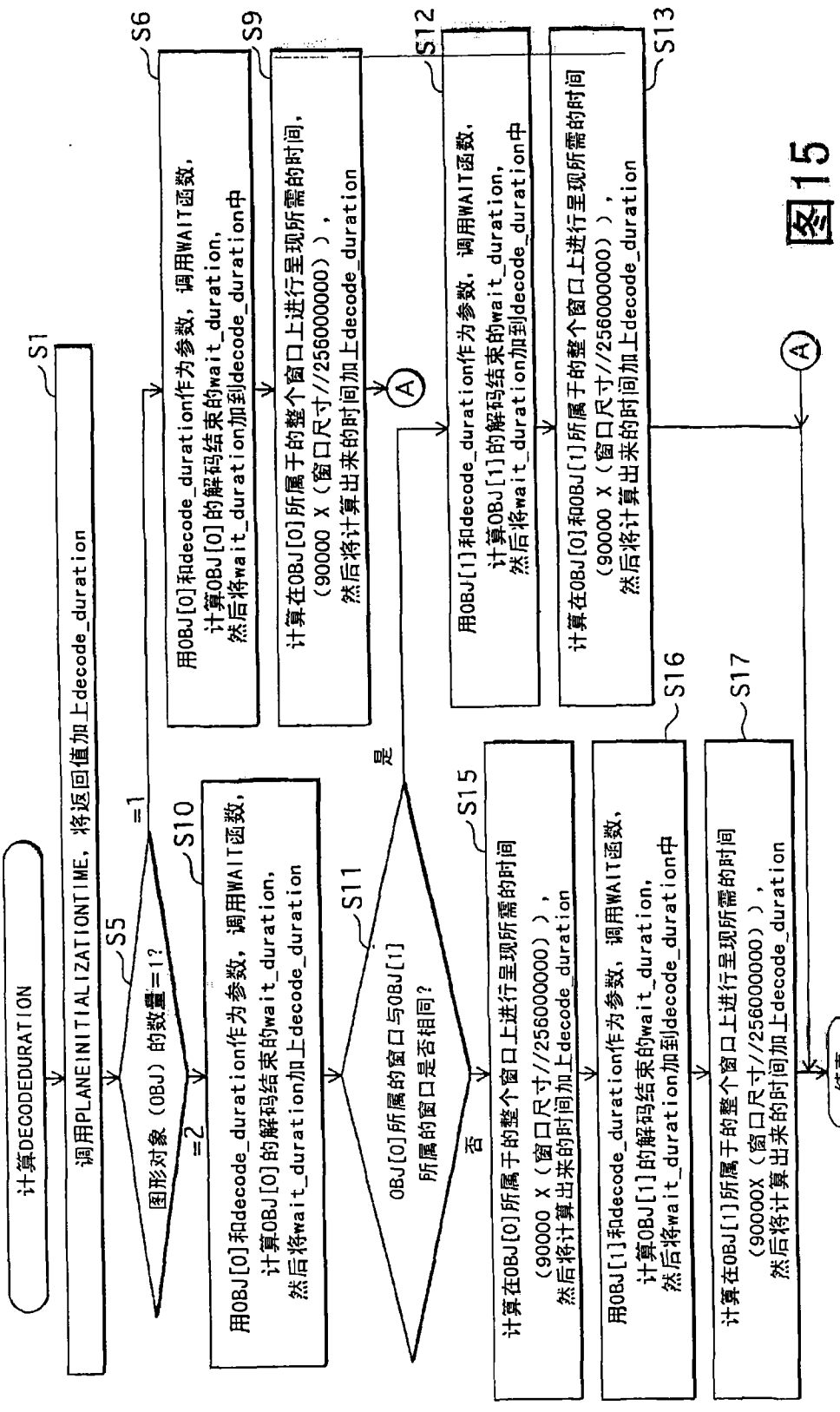

计算DECODEDURATION
S1
调用PLANEINITIALIZATIONTIME，将返回值加上decode_duration
S5
图形对象（OBJ）的数量=1?
=1
=2
S6
用OBJ[0]和decode_duration作为参数，调用WAIT函数，
计算OBJ[0]的解码结束的wait_duration，
然后将wait_duration加到decode_duration中
S9
计算在OBJ[0]所属于的整个窗口上进行呈现所需的时间，
（90000 X（窗口尺寸//256000000））；
然后将计算出来的时间加上decode_duration
S10
用OBJ[0]和decode_duration作为参数，调用WAIT函数，
计算OBJ[0]的解码结束的wait_duration，
然后将wait_duration加上decode_duration
S11
OBJ[0]所属的窗口与OBJ[1]
所属的窗口是否相同?
是
否
A
S12
用OBJ[1]和decode_duration作为参数，调用WAIT函数，
计算OBJ[1]的解码结束的wait_duration，
然后将wait_duration加到decode_duration中
S13
计算在OBJ[0]和OBJ[1]所属于的整个窗口上进行呈现所需的时间
（90000 X（窗口尺寸//256000000））；
然后将计算出来的时间加上decode_duration
S15
计算在OBJ[0]所属于的整个窗口上进行呈现所需的时间
（90000 X（窗口尺寸//256000000））；
然后将计算出来的时间加上decode_duration
S16
用OBJ[1]和decode_duration作为参数，调用WAIT函数，
计算OBJ[1]的解码结束的wait_duration，
然后将wait_duration加到decode_duration中
S17
计算在OBJ[1]所属于的整个窗口上进行呈现所需的时间
（90000X（窗口尺寸//256000000））；
然后将计算出来的时间加上decode_duration
A
结束

图15

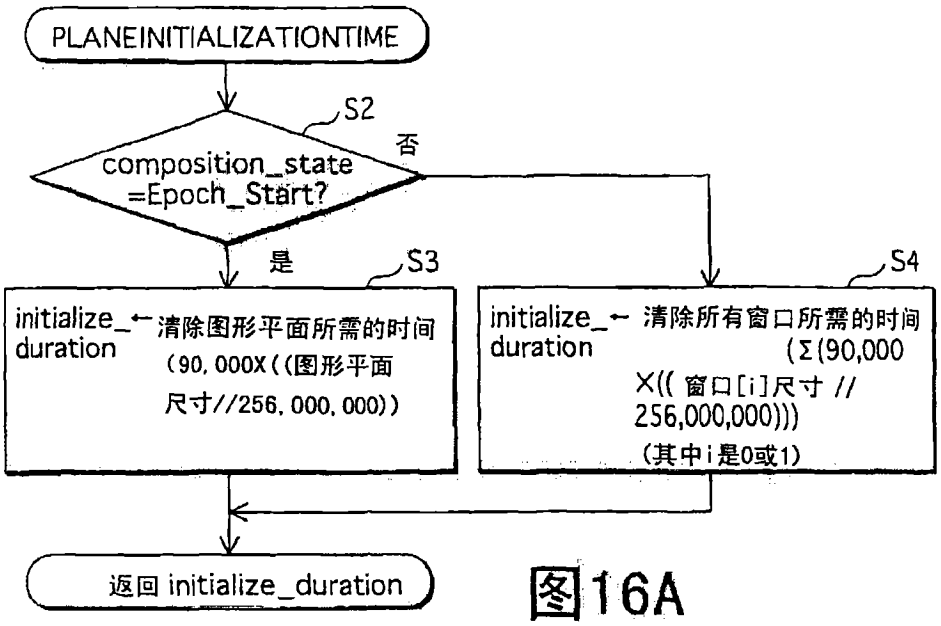

PLANEINITIALIZATIONTIME
S2
composition_state =Epoch_Start?
否
是
S3
initialize_duration ← 清除图形平面所需的时间 (90,000X((图形平面尺寸//256,000,000))
S4
initialize_duration ← 清除所有窗口所需的时间 (Σ(90,000 X((窗口[i]尺寸 // 256,000,000))) (其中i是0或1)
返回 initialize_duration

图16A

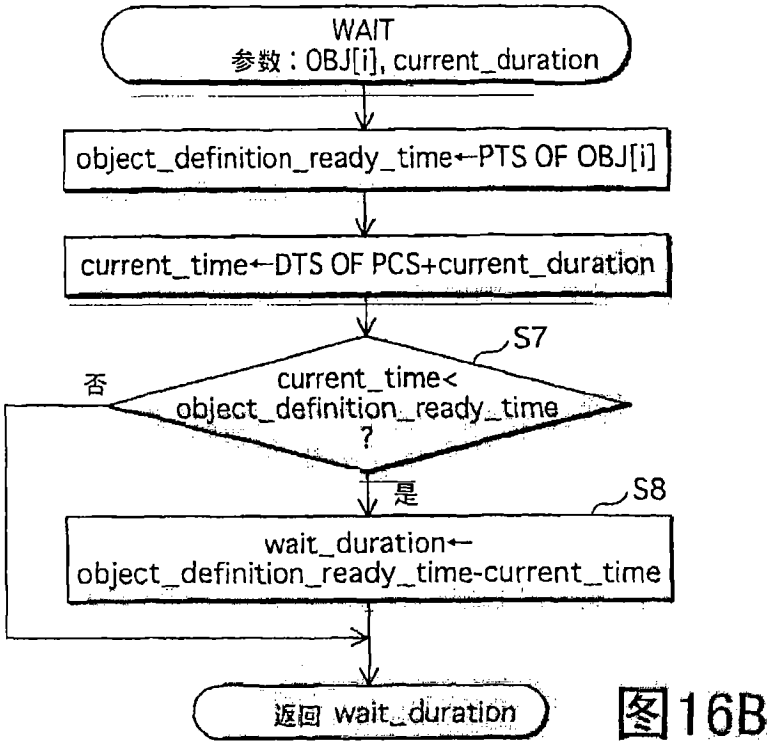
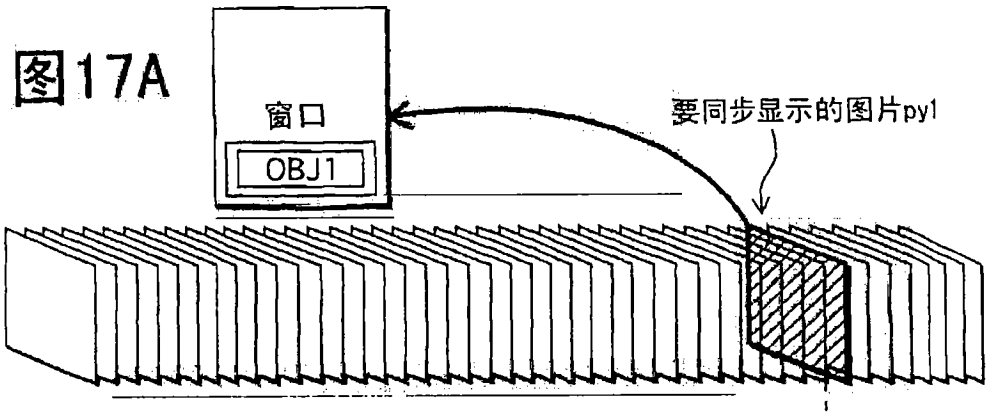

WAIT
参数：OBJ[i], current_duration
object_definition_ready_time←PTS OF OBJ[i]
current_time←DTS OF PCS+current_duration
S7
current_time< object_definition_ready_time ?
否
是
S8
wait_duration← object_definition_ready_time-current_time
返回 wait_duration

图16B

图17A



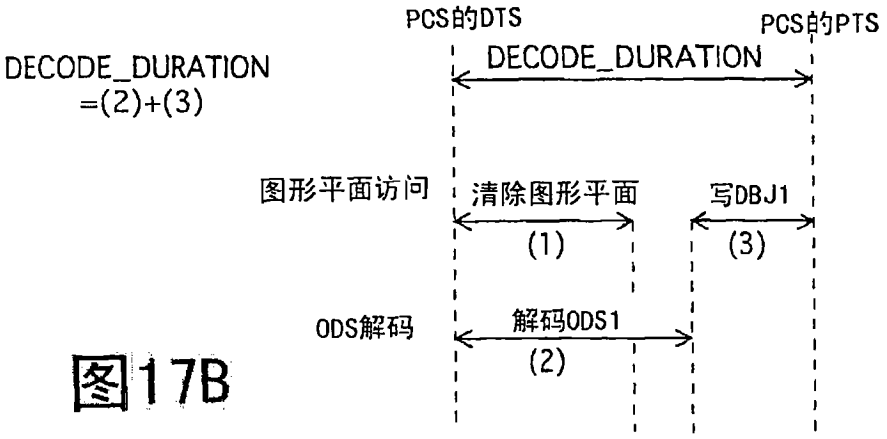


图17B

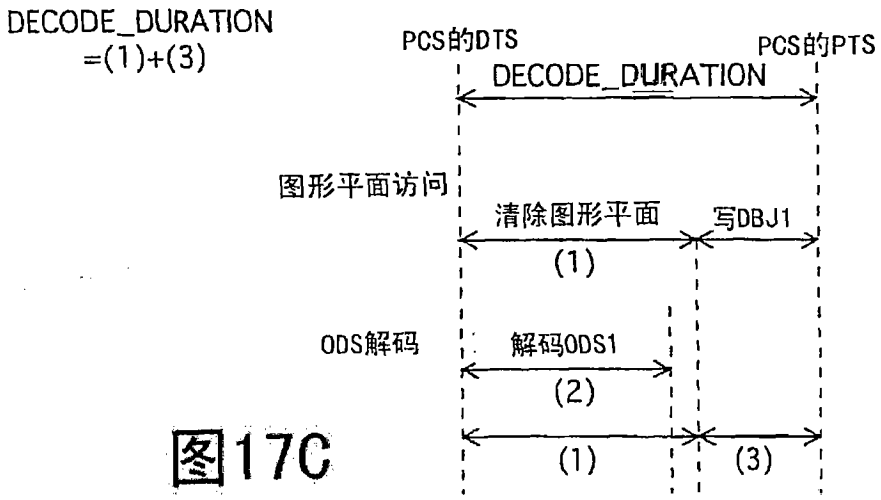


图17C

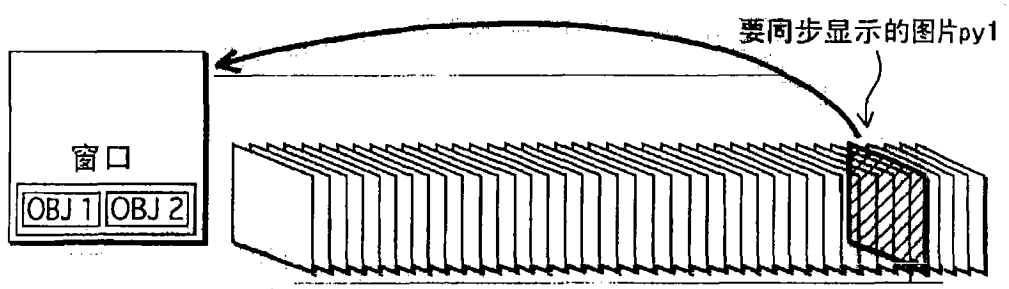


图18A

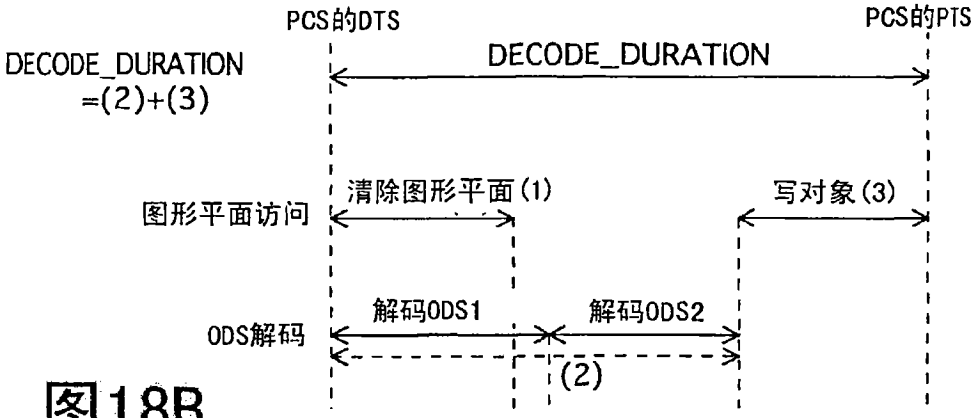


图18B

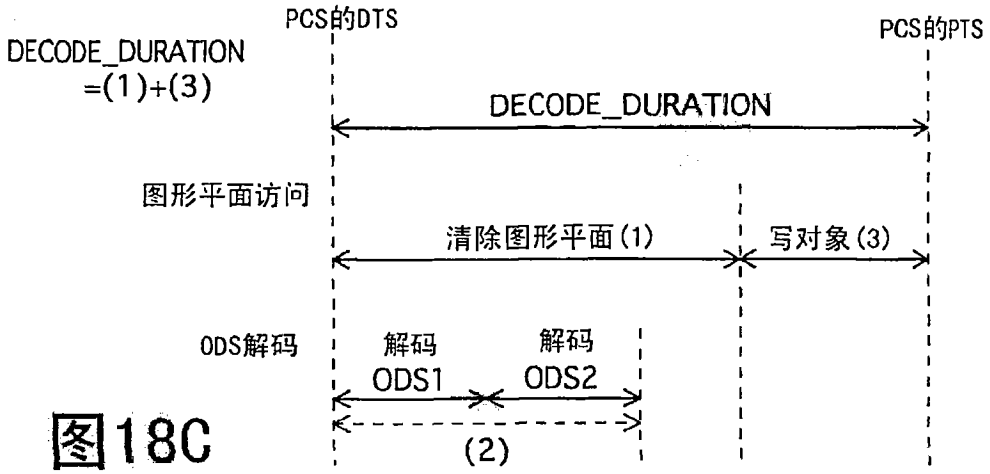


图18C

图19A

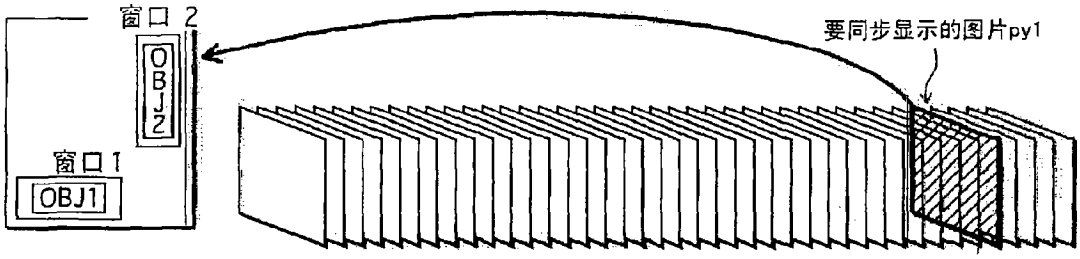


图19B

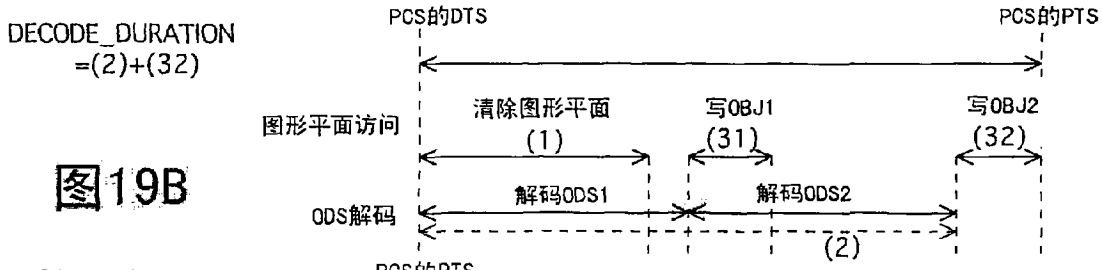
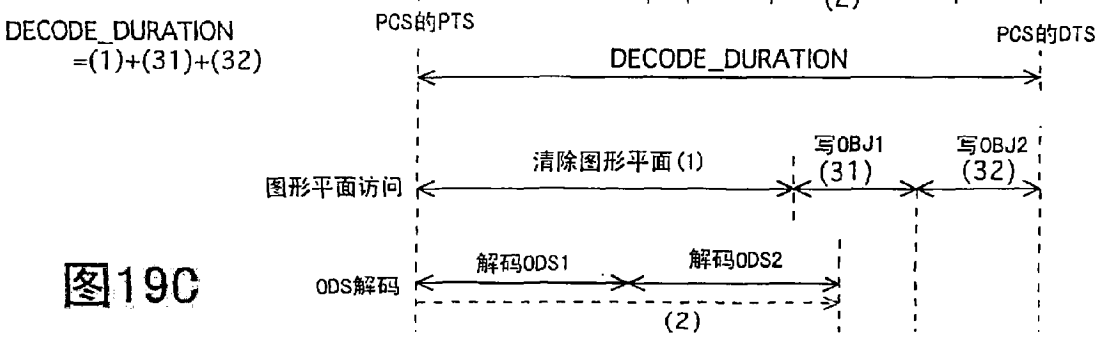


图19C



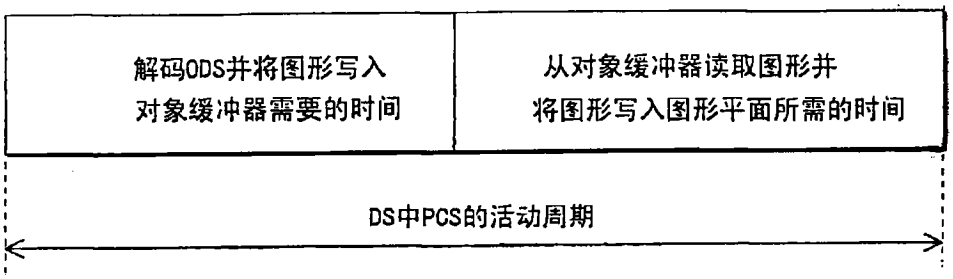

解码ODS并将图形写入
对象缓冲器需要的时间
从对象缓冲器读取图形并
将图形写入图形平面所需的时间
DS中PCS的活动周期

图20

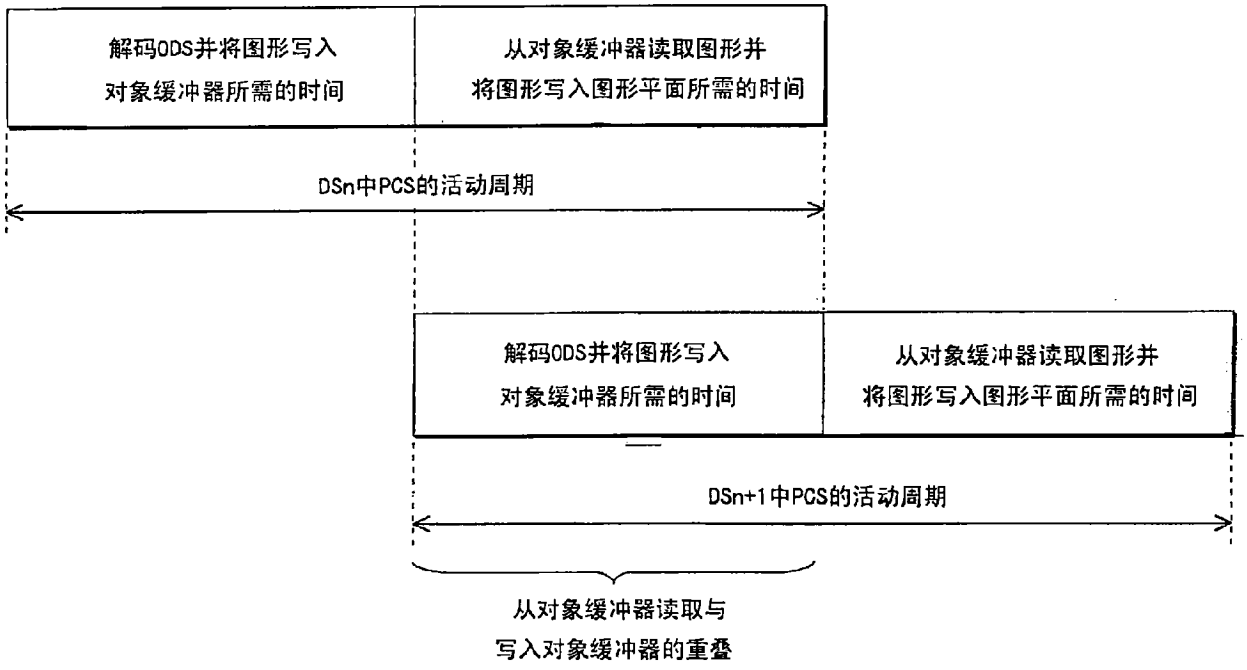

解码ODS并将图形写入
对象缓冲器所需的时间
从对象缓冲器读取图形并
将图形写入图形平面所需的时间
DSn中PCS的活动周期
解码ODS并将图形写入
对象缓冲器所需的时间
从对象缓冲器读取图形并
将图形写入图形平面所需的时间
DSn+1中PCS的活动周期
从对象缓冲器读取与
写入对象缓冲器的重叠

图21

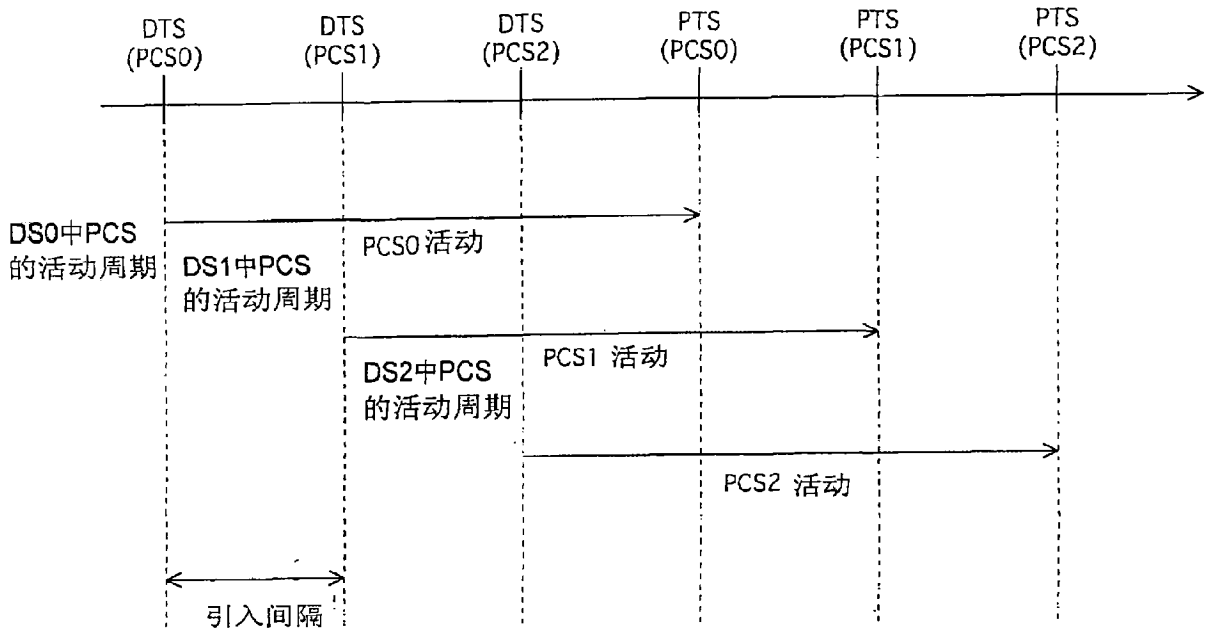

DTS (PCS0)
DTS (PCS1)
DTS (PCS2)
PTS (PCS0)
PTS (PCS1)
PTS (PCS2)
DS0中PCS 的活动周期
PCS0 活动
DS1中PCS 的活动周期
PCS1 活动
DS2中PCS 的活动周期
PCS2 活动
引入间隔

图22

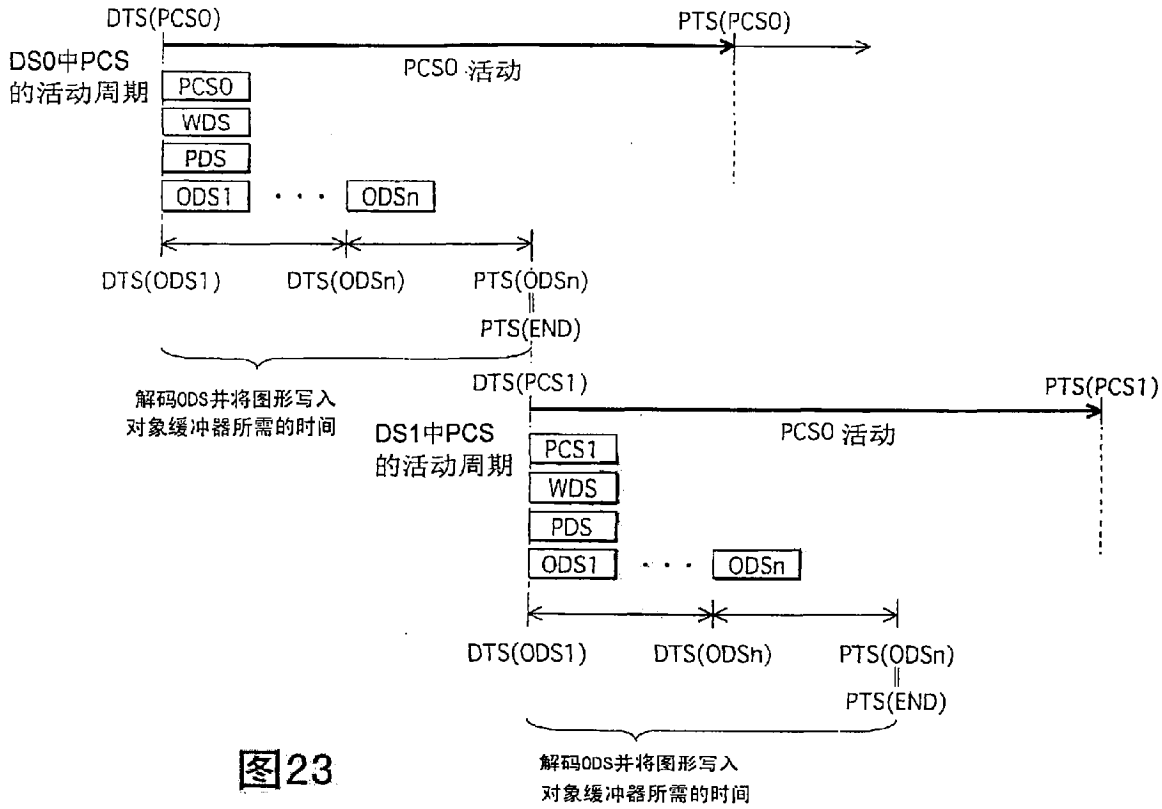

DTS(PCS0)
PTS(PCS0)
DS0中PCS 的活动周期
PCS0 活动
PCS0
WDS
PDS
ODS1
ODSn
DTS(ODS1)
DTS(ODSn)
PTS(ODSn)
PTS(END)
解码ODS并将图形写入 对象缓冲器所需的时间
DTS(PCS1)
PTS(PCS1)
DS1中PCS 的活动周期
PCS0 活动
PCS1
WDS
PDS
ODS1
ODSn
DTS(ODS1)
DTS(ODSn)
PTS(ODSn)
PTS(END)
解码ODS并将图形写入 对象缓冲器所需的时间

图23

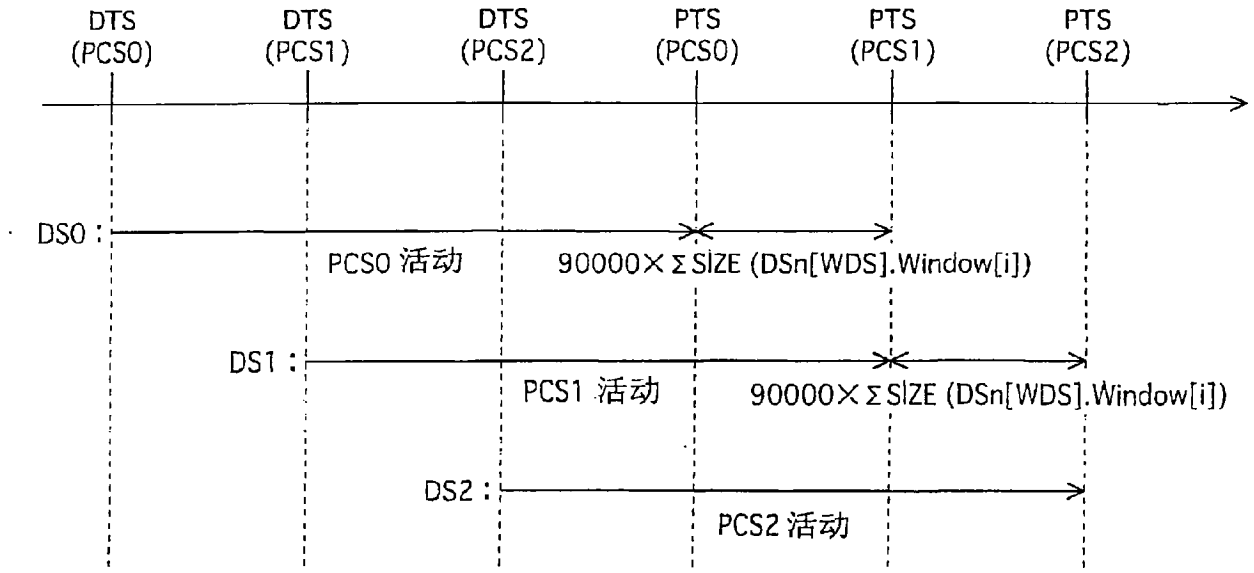

DTS
(PCS0)
DTS
(PCS1)
DTS
(PCS2)
PTS
(PCS0)
PTS
(PCS1)
PTS
(PCS2)
DS0：
PCS0 活动
90000×ΣSIZE (DSn[WDS].Window[i])
DS1：
PCS1 活动
90000×ΣSIZE (DSn[WDS].Window[i])
DS2：
PCS2 活动

图24

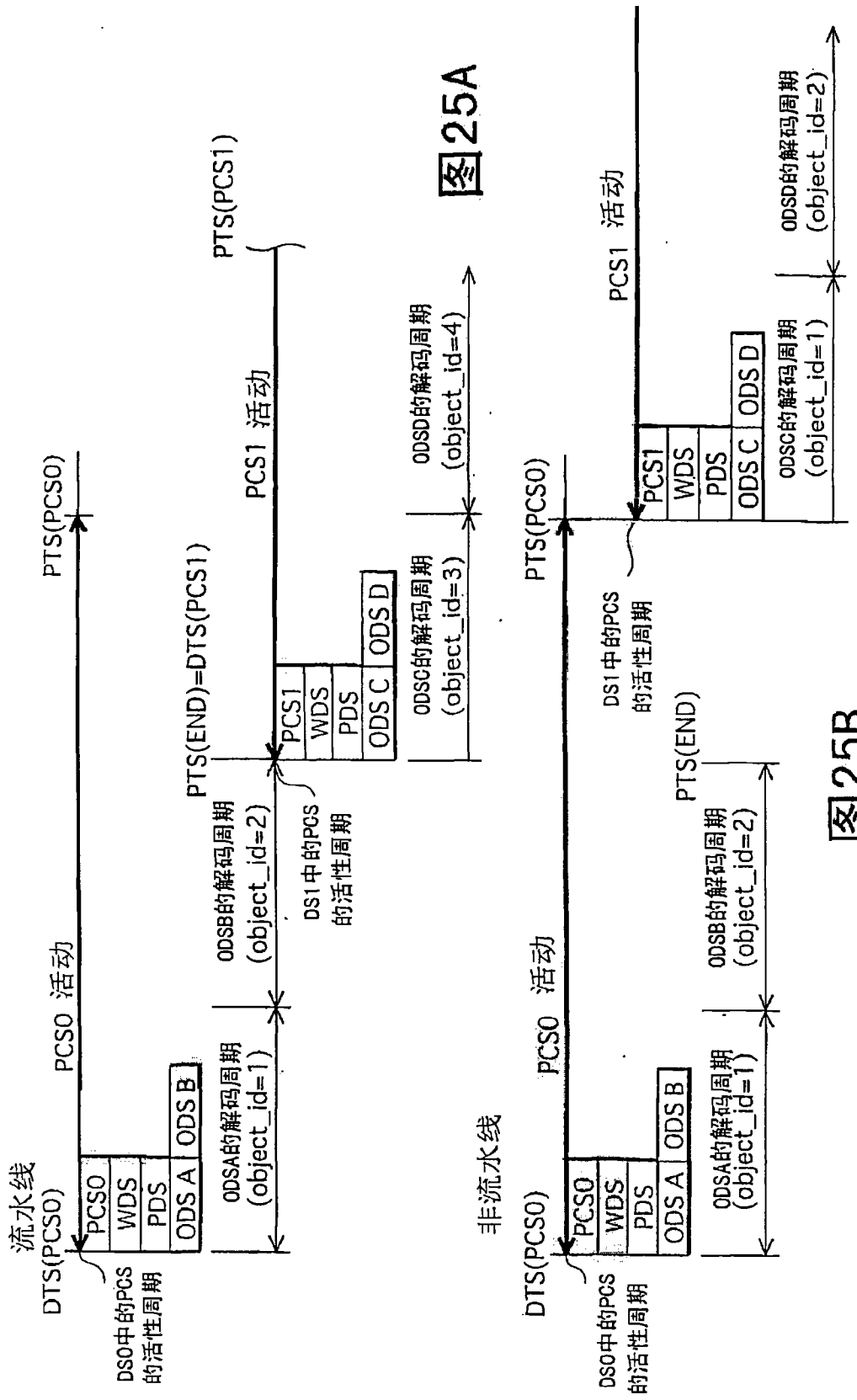

流水线
DTS(PCS0)
PTS(PCS0)
PCS0 活动
PCS0
WDS
PDS
ODS A
ODS B
DS0中的PCS的活性周期
ODSA的解码周期(object_id=1)
ODSB的解码周期(object_id=2)
PTS(END)=DTS(PCS1)
PTS(PCS1)
PCS1 活动
PCS1
WDS
PDS
ODS C
ODS D
DS1中的PCS的活性周期
ODSC的解码周期(object_id=3)
ODSD的解码周期(object_id=4)

图25A

非流水线
DTS(PCS0)
PTS(PCS0)
PCS0 活动
PCS0
WDS
PDS
ODS A
ODS B
DS0中的PCS的活性周期
ODSA的解码周期(object_id=1)
ODSB的解码周期(object_id=2)
PTS(END)
PCS1 活动
PCS1
WDS
PDS
ODS C
ODS D
DS1中的PCS的活性周期
ODSC的解码周期(object_id=1)
ODSD的解码周期(object_id=2)

图25B

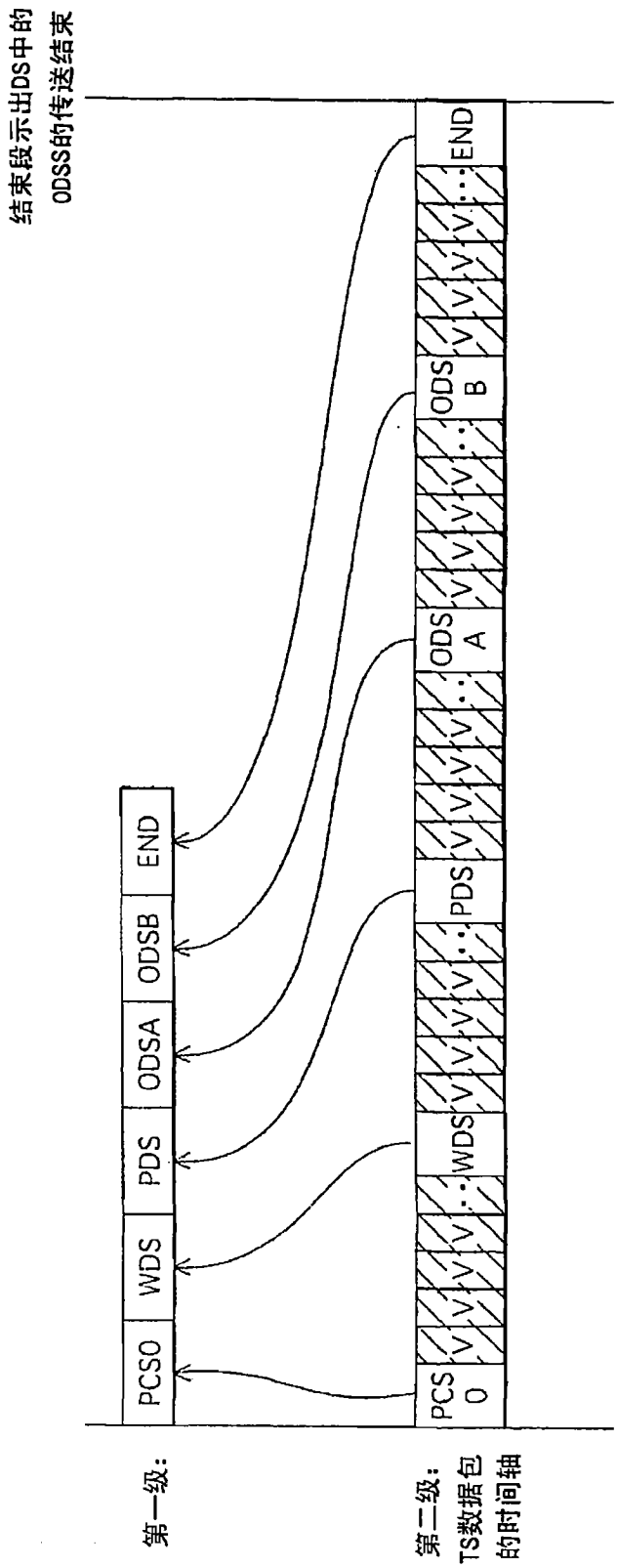

结束段示出DS中的ODSS的传送结束
PCSO
WDS
PDS
ODSA
ODSB
END
第一级：
PCS 0
WDS
PDS
ODS A
ODS B
END
第二级：TS数据包的时间轴

图26

屏幕合成

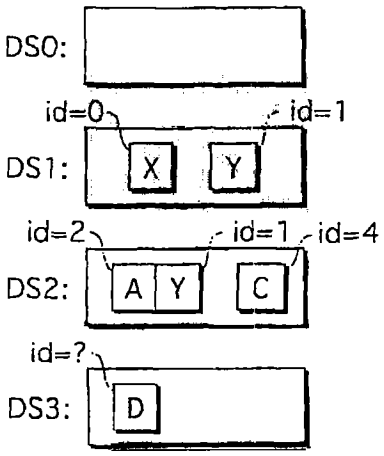


图27A

活动周期重叠和ODS传送

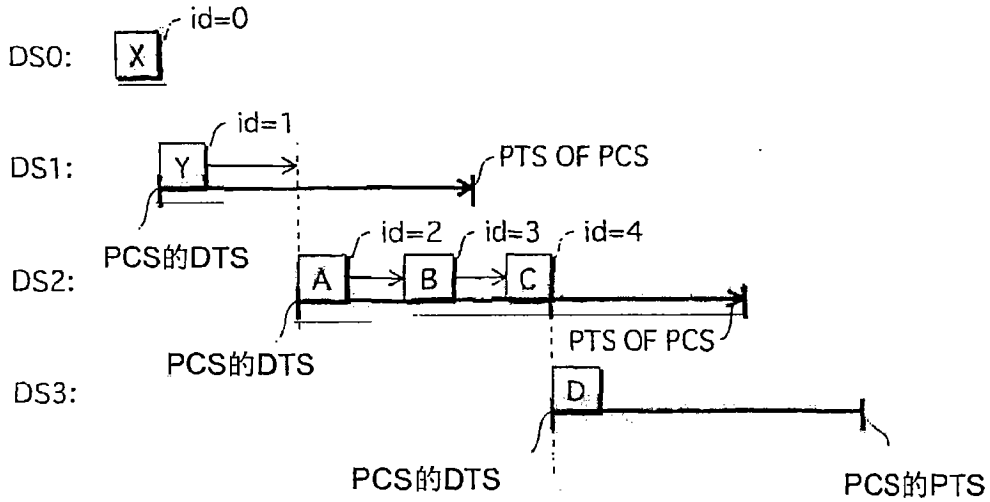


图27B

对象缓冲器中的设置

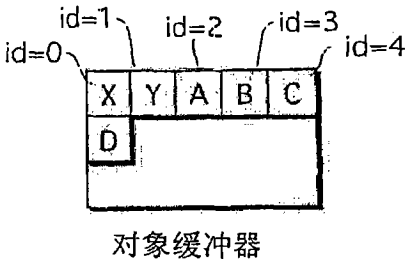


图27C

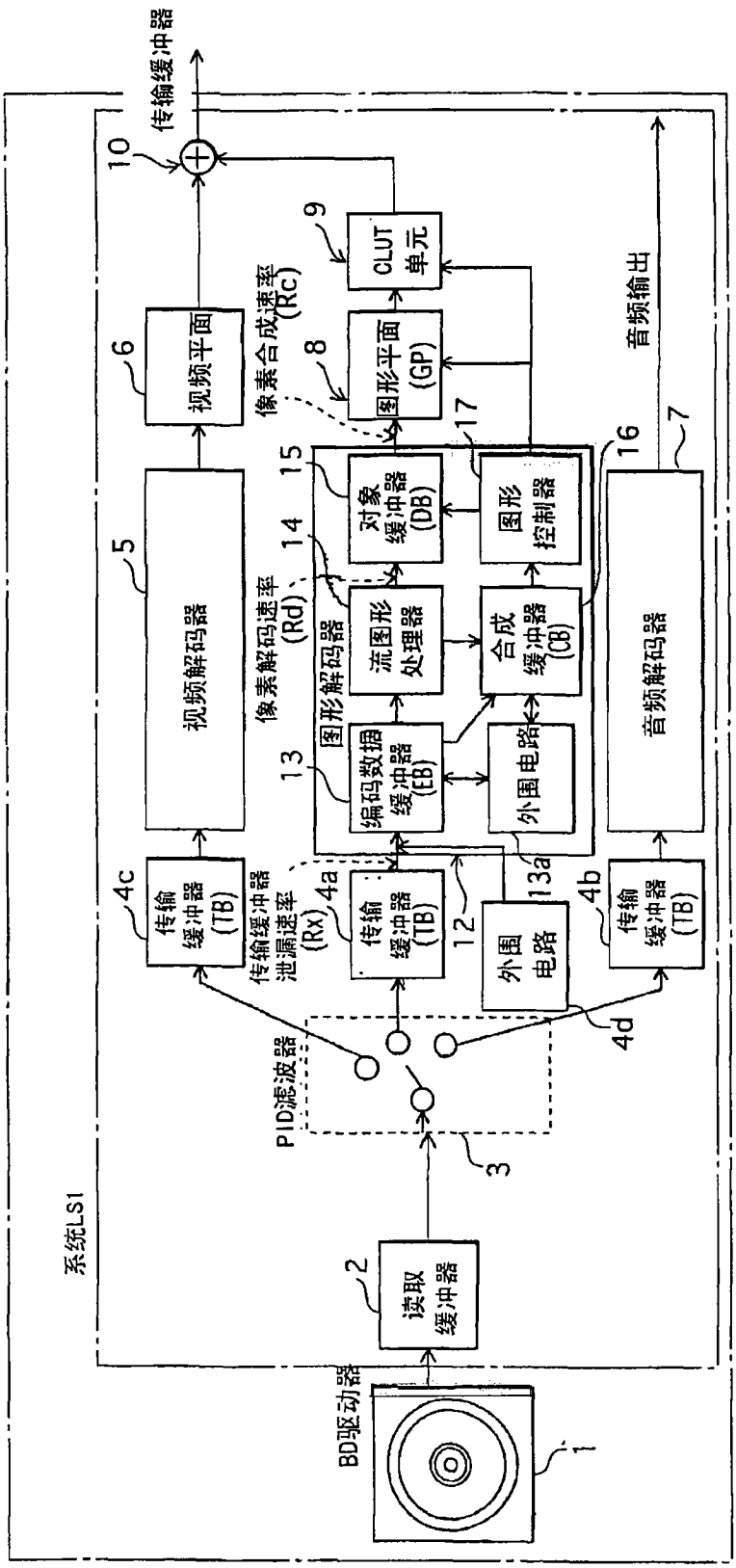

传输缓冲器
10
视频平面
6
像素合成速率(Rc)
9
CLUT单元
8
图形平面(GP)
17
对象缓冲器(DB)
15
图形控制器
16
音频输出
7
视频解码器
5
像素解码速率(Rd)、
14
流图形处理器
合成缓冲器(CB)
图形解码器
13
编码数据缓冲器(EB)
外围电路
音频解码器
传输缓冲器(TB)
4c
传输缓冲器泄漏速率(Rx)
4a
传输缓冲器(TB)
12
外围电路
13a
4b
传输缓冲器(TB)
PID滤波器
4d
3
系统LSI
2
读取缓冲器
BD驱动器
1

图28

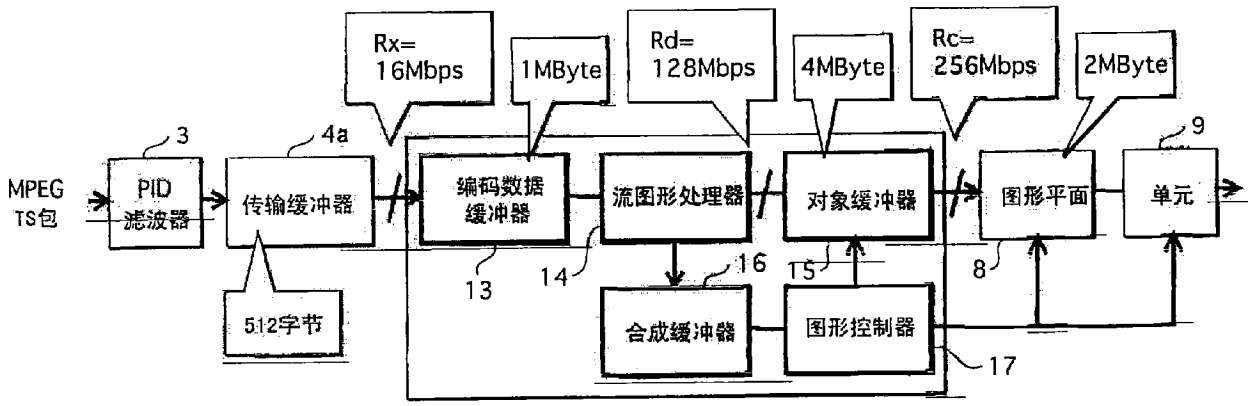

Rx=
16Mbps
1MByte
Rd=
128Mbps
4MByte
Rc=
256Mbps
2MByte
3
4a
9
MPEG
TS包
PID
滤波器
传输缓冲器
编码数据
缓冲器
流图形处理器
对象缓冲器
图形平面
单元
13
14
16
T5
8
512字节
合成缓冲器
图形控制器
17

图29

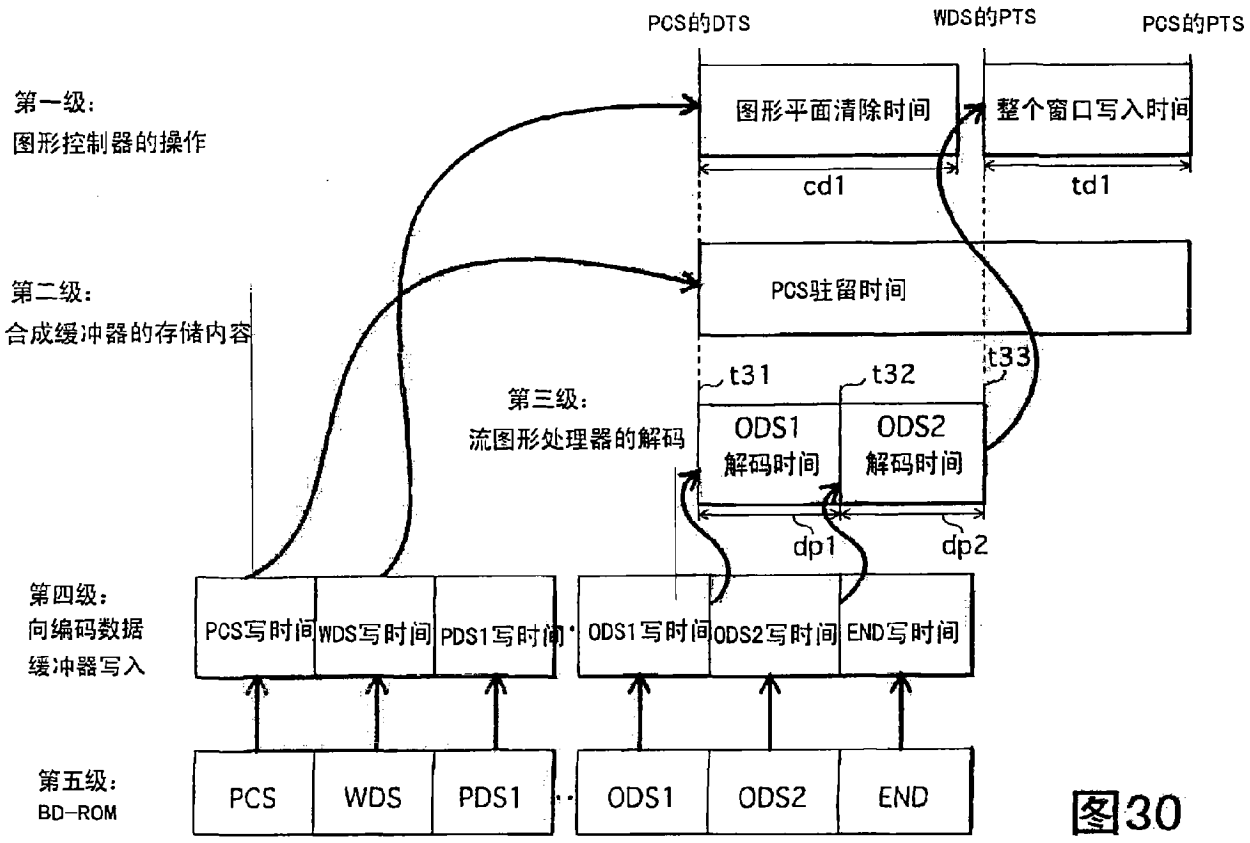

PCS的DTS
WDS的PTS
PCS的PTS
第一级：
图形控制器的操作
图形平面清除时间
整个窗口写入时间
cd1
td1
第二级：
合成缓冲器的存储内容
PCS驻留时间
t31
t32
t33
第三级：
流图形处理器的解码
ODS1
解码时间
ODS2
解码时间
dp1
dp2
第四级：
向编码数据
缓冲器写入
PCS写时间
WDS写时间
PDS1写时间
ODS1写时间
ODS2写时间
END写时间
第五级：
BD-ROM
PCS
WDS
PDS1
ODS1
ODS2
END

图30

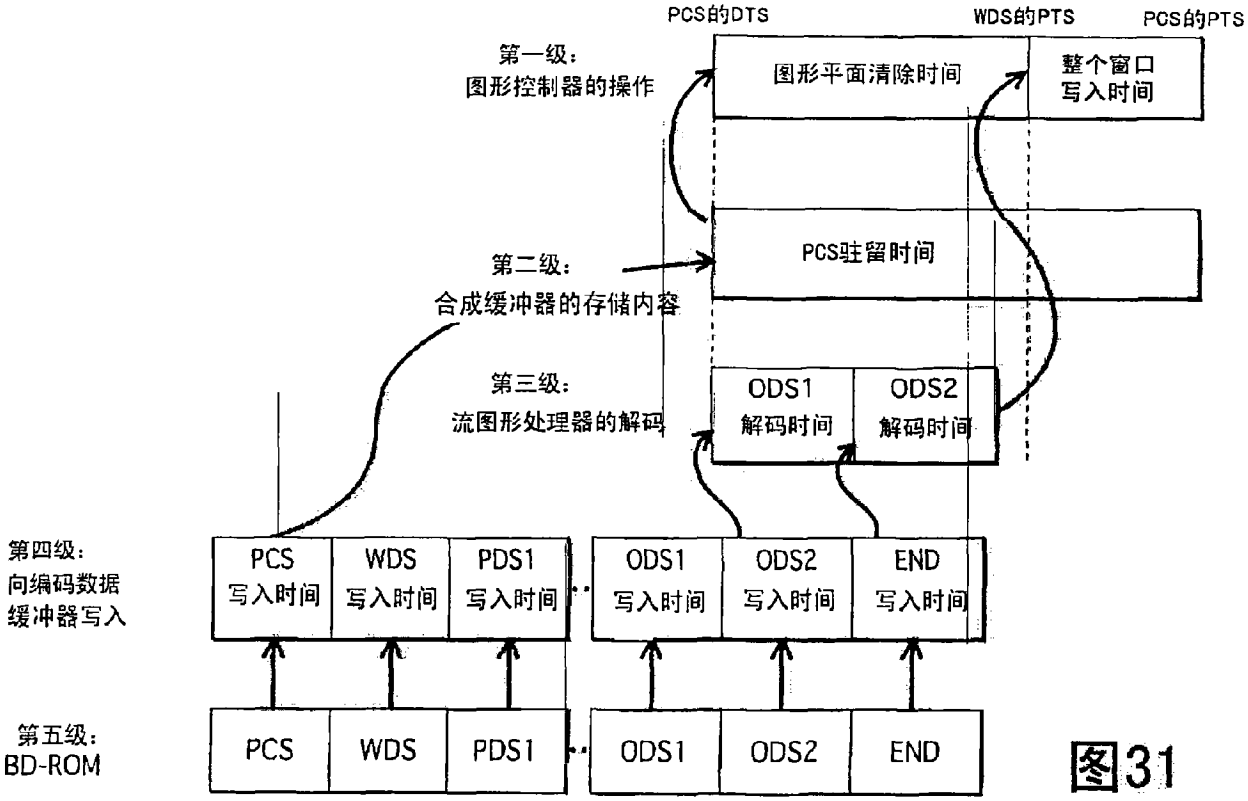

PCS的DTS
WDS的PTS
PCS的PTS
第一级：
图形控制器的操作
图形平面清除时间
整个窗口
写入时间
PCS驻留时间
第二级：
合成缓冲器的存储内容
第三级：
流图形处理器的解码
ODS1
解码时间
ODS2
解码时间
第四级：
向编码数据
缓冲器写入
PCS
写入时间
WDS
写入时间
PDS1
写入时间
ODS1
写入时间
ODS2
写入时间
END
写入时间
第五级：
BD-ROM
PCS
WDS
PDS1
ODS1
ODS2
END

图31

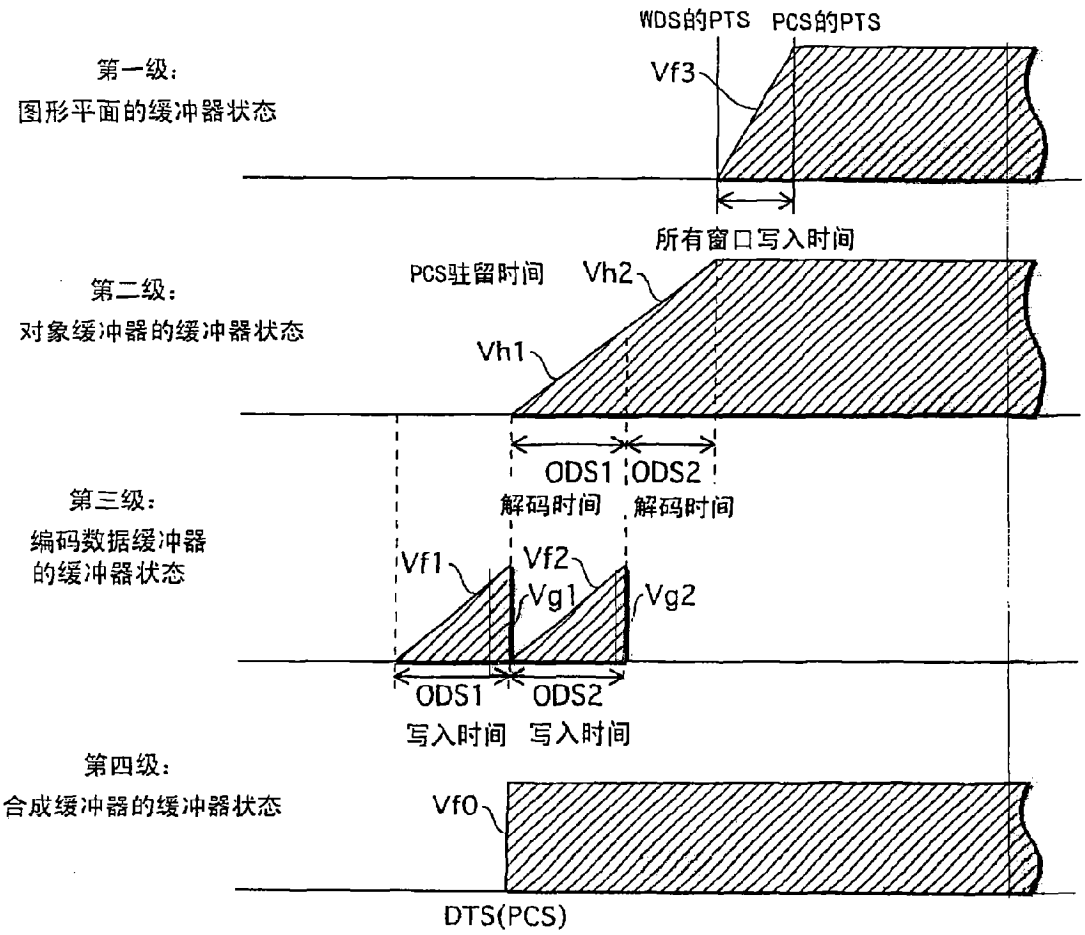

WDS的PTS
PCS的PTS
第一级：
图形平面的缓冲器状态
Vf3
所有窗口写入时间
PCS驻留时间
Vh2
第二级：
对象缓冲器的缓冲器状态
Vh1
ODS1
ODS2
解码时间
解码时间
第三级：
编码数据缓冲器
的缓冲器状态
Vf1
Vf2
Vg1
Vg2
ODS1
ODS2
写入时间
写入时间
第四级：
合成缓冲器的缓冲器状态
Vf0
DTS(PCS)

图32

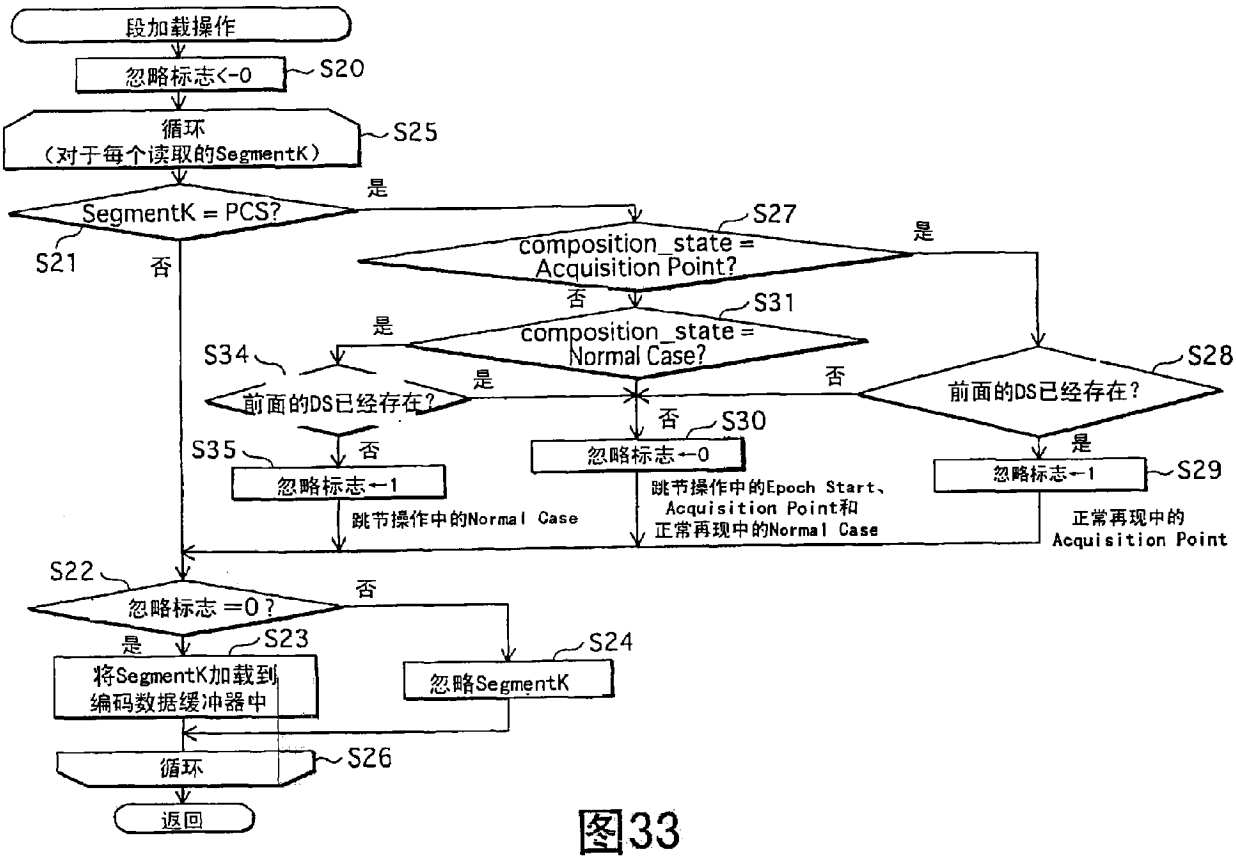

段加载操作
忽略标志<-0
S20
循环
（对于每个读取的SegmentK）
S25
SegmentK = PCS?
是
S21
否
S27
composition_state = Acquisition Point?
是
否
S31
composition_state = Normal Case?
是
S34
前面的DS已经存在?
是
否
S35
否
忽略标志←1
跳节操作中的Normal Case
否
S30
忽略标志←0
跳节操作中的Epoch Start、Acquisition Point和正常再现中的Normal Case
S28
前面的DS已经存在?
是
忽略标志←1
S29
正常再现中的Acquisition Point
S22
忽略标志 =0?
否
是
S23
将SegmentK加载到编码数据缓冲器中
S24
忽略SegmentK
循环
S26
返回

图33

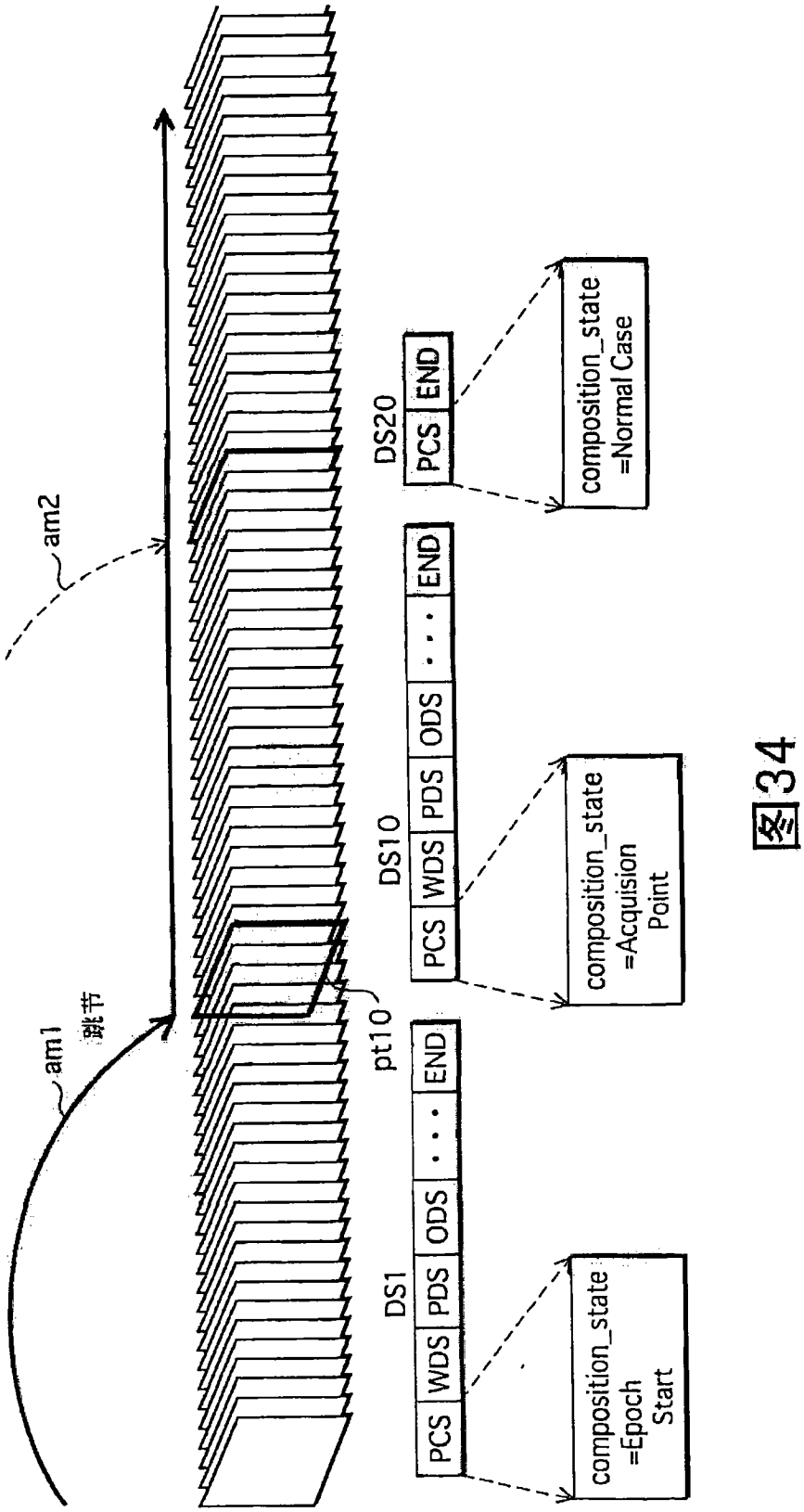

am2
am1
跳节
pt10
DS20
PCS
END
composition_state
=Normal Case
DS10
PCS
WDS
PDS
ODS
• • •
END
composition_state
=Acquisition
Point
DS1
PCS
WDS
PDS
ODS
• • •
END
composition_state
=Epoch
Start

图34

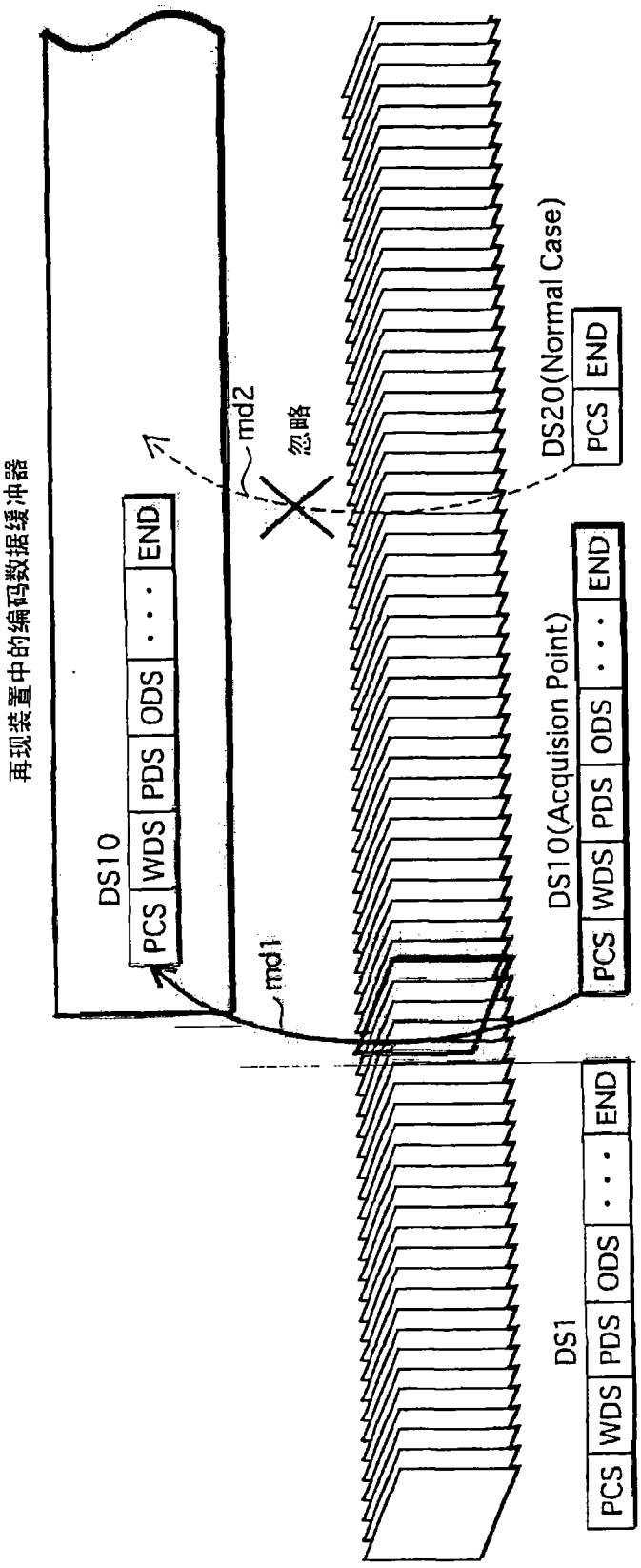

再现装置中的编码数据缓冲器
DS10
PCS
WDS
PDS
ODS
· · ·
END
md1
md2
忽略
DS20(Normal Case)
PCS
END
DS10(Acquision Point)
PCS
WDS
PDS
ODS
· · ·
END
DS1
PCS
WDS
PDS
ODS
· · ·
END

图35

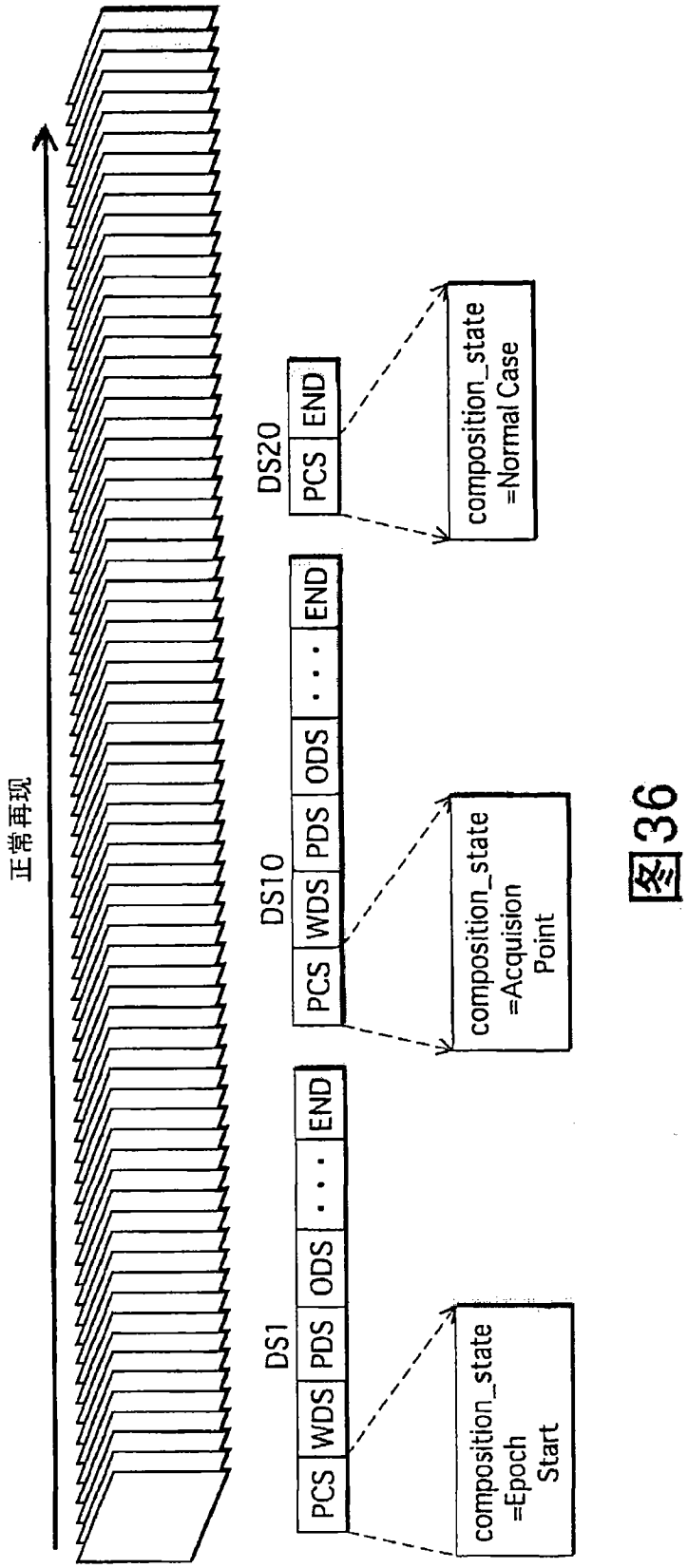

正常再现
DS1
PCS
WDS
PDS
ODS
· · ·
END
composition_state
=Epoch
Start
DS10
PCS
WDS
PDS
ODS
· · ·
END
composition_state
=Acquision
Point
DS20
PCS
END
composition_state
=Normal Case

图36

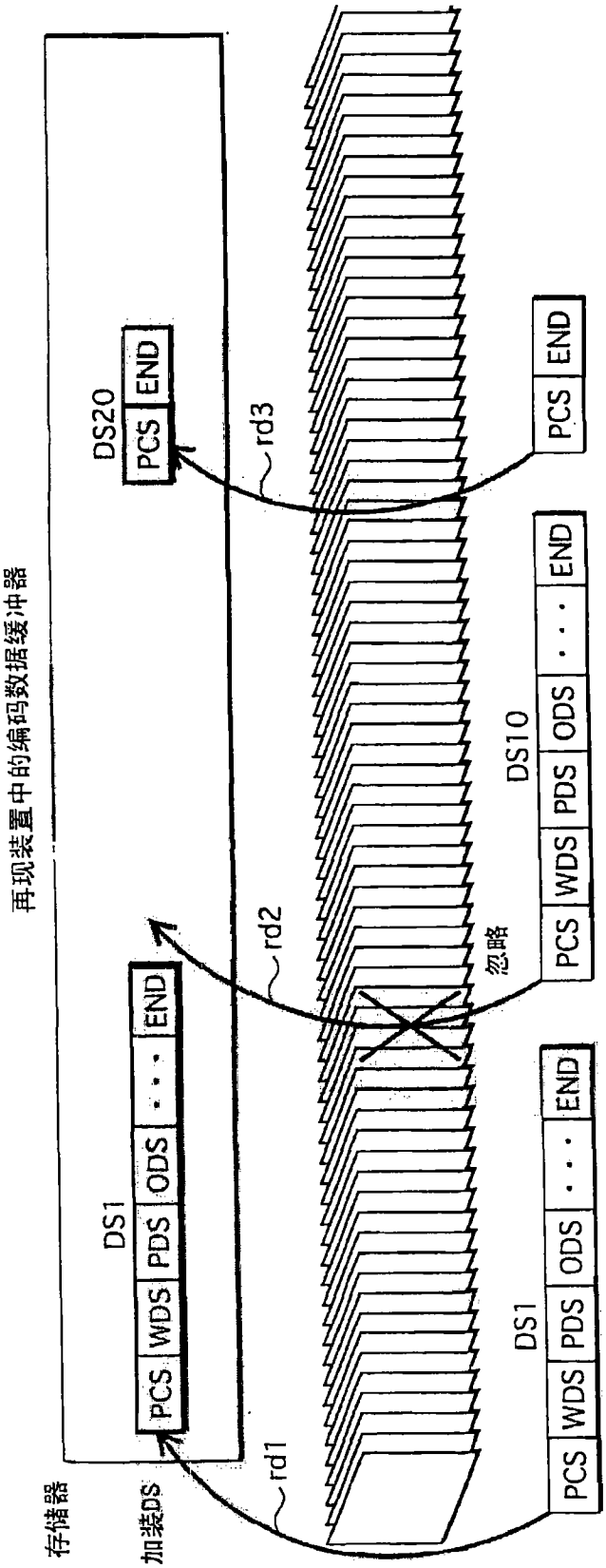

再现装置中的编码数据缓冲器
存储器
加载DS
rd1
rd2
rd3
忽略
DS1
PCS
WDS
PDS
ODS
· · ·
END
DS10
DS20

图37

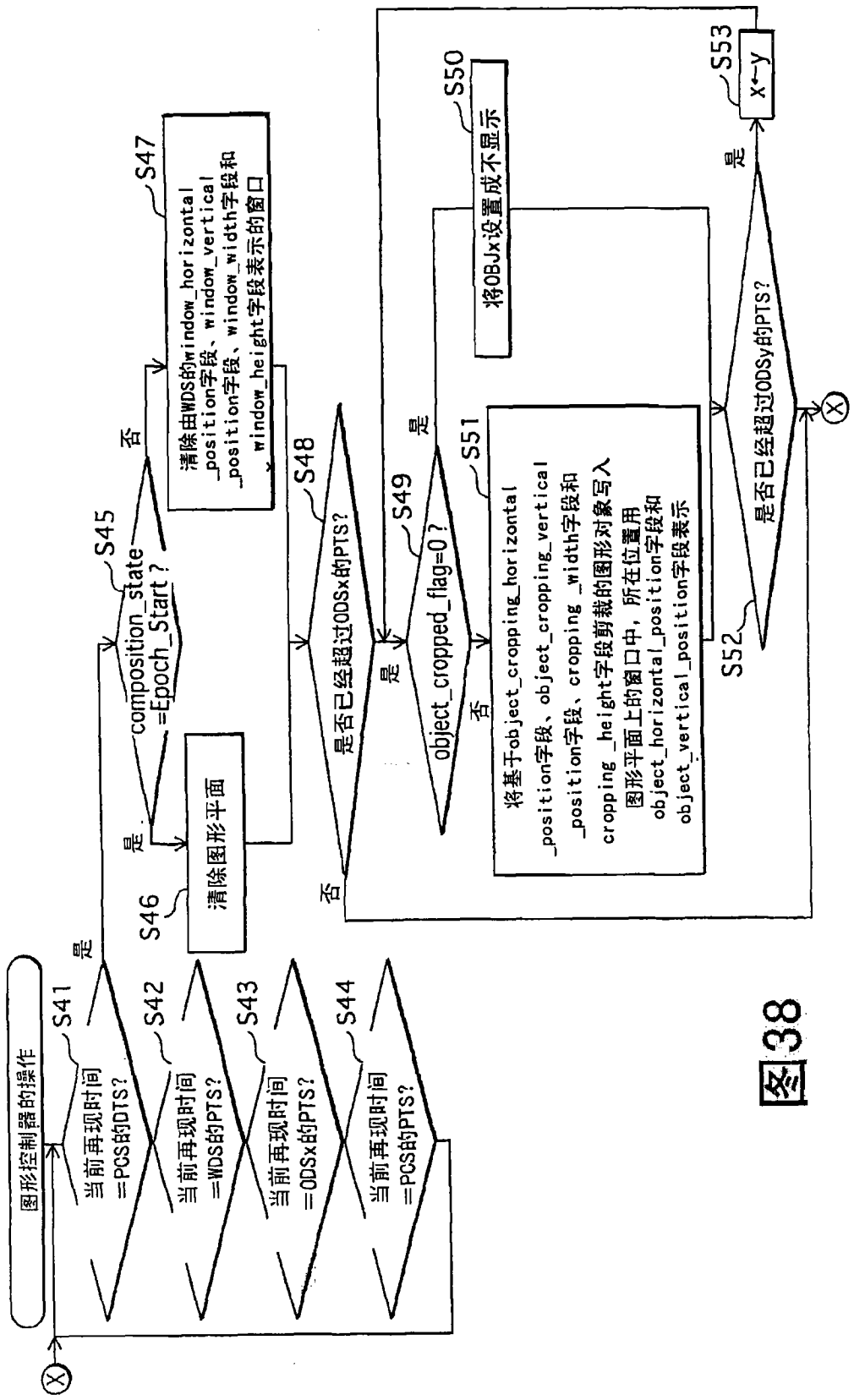

图形控制器的操作
S41
当前再现时间=PCS的DTS?
是
S42
当前再现时间=WDS的PTS?
是
S43
当前再现时间=ODSx的PTS?
是
S44
当前再现时间=PCS的PTS?
是
S45
composition_state=Epoch_Start?
否
S47
清除由WDS的window_horizontal_position字段、window_vertical_position字段、window_width字段和window_height字段表示的窗口
是
S46
清除图形平面
S48
是否已经超过ODSx的PTS?
否
是
S49
object_cropped_flag=0?
否
S51
将基于object_cropping_horizontal_position字段、object_cropping_vertical_position字段、cropping_width字段和cropping_height字段剪裁的图形对象写入图形平面上的窗口中，所在位置用object_horizontal_position字段和object_vertical_position字段表示
S50
将OBJx设置成不显示
S52
是否已经超过ODSy的PTS?
是
S53
x←y

图38

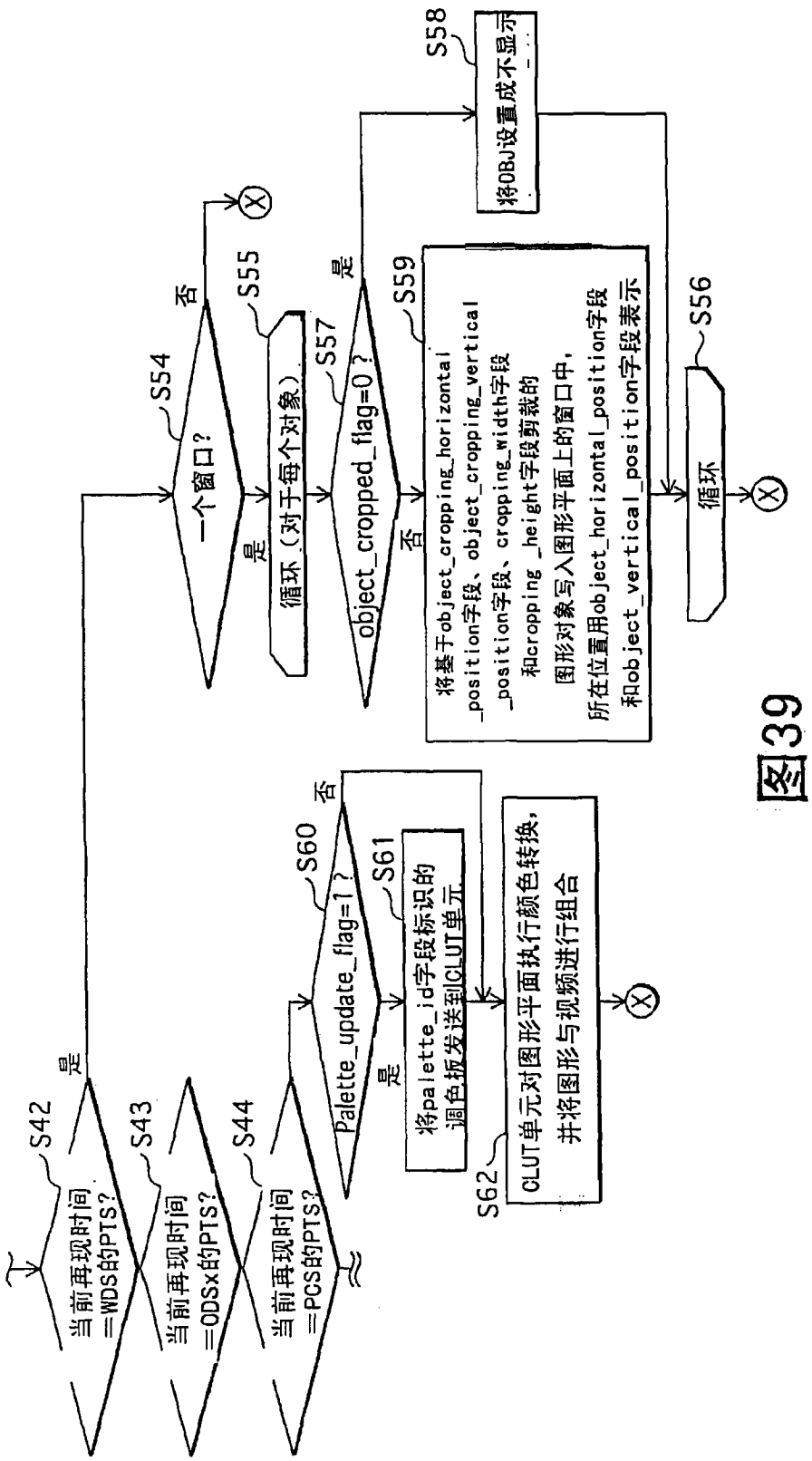

S42
当前再现时间=WDS的PTS?
是
S43
当前再现时间=ODSx的PTS?
S44
当前再现时间=PCS的PTS?
Palette_update_flag=1?
否
是
S60
S61
将palette_id字段标识的调色板发送到CLUT单元
S62
CLUT单元对图形平面执行颜色转换，并将图形与视频进行组合
S54
一个窗口?
否
是
S55
循环（对于每个对象）
S57
object_cropped_flag=0?
是
否
S59
将基于object_cropping_horizontal_position字段、object_cropping_vertical_position字段、cropping_width字段和cropping_height字段剪裁的图形对象写入图形平面上的窗口中，所在位置用object_horizontal_position字段和object_vertical_position字段表示
S58
将OBJ设置成不显示
S56
循环

图39

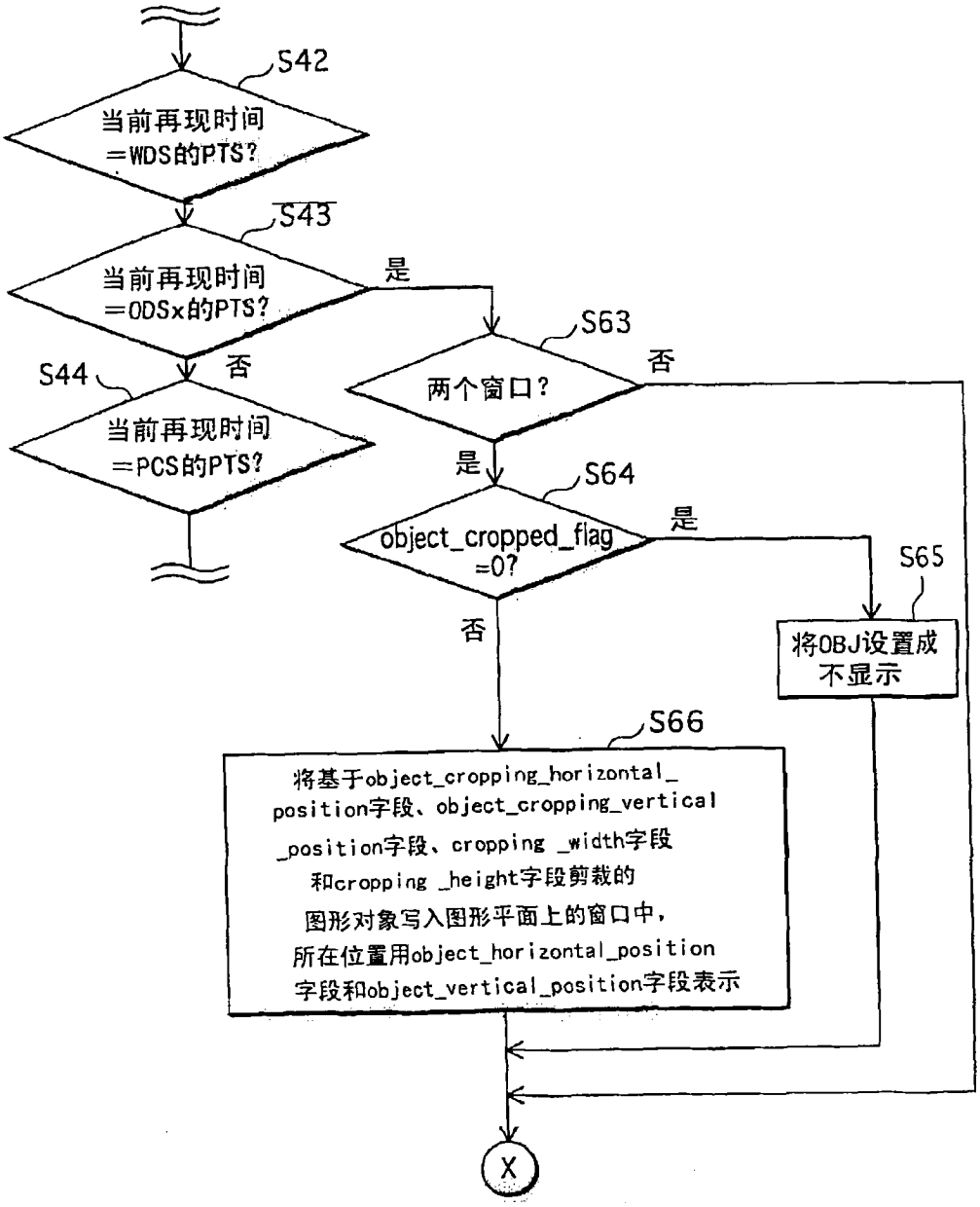

S42
当前再现时间
=WDS的PTS?
S43
当前再现时间
=ODSx的PTS?
是
否
S44
当前再现时间
=PCS的PTS?
S63
两个窗口?
否
是
S64
object_cropped_flag
=0?
是
否
S65
将OBJ设置成
不显示
S66
将基于object_cropping_horizontal_
position字段、object_cropping_vertical
_position字段、cropping _width字段
和cropping _height字段剪裁的
图形对象写入图形平面上的窗口中，
所在位置用object_horizontal_position
字段和object_vertical_position字段表示
X

图40

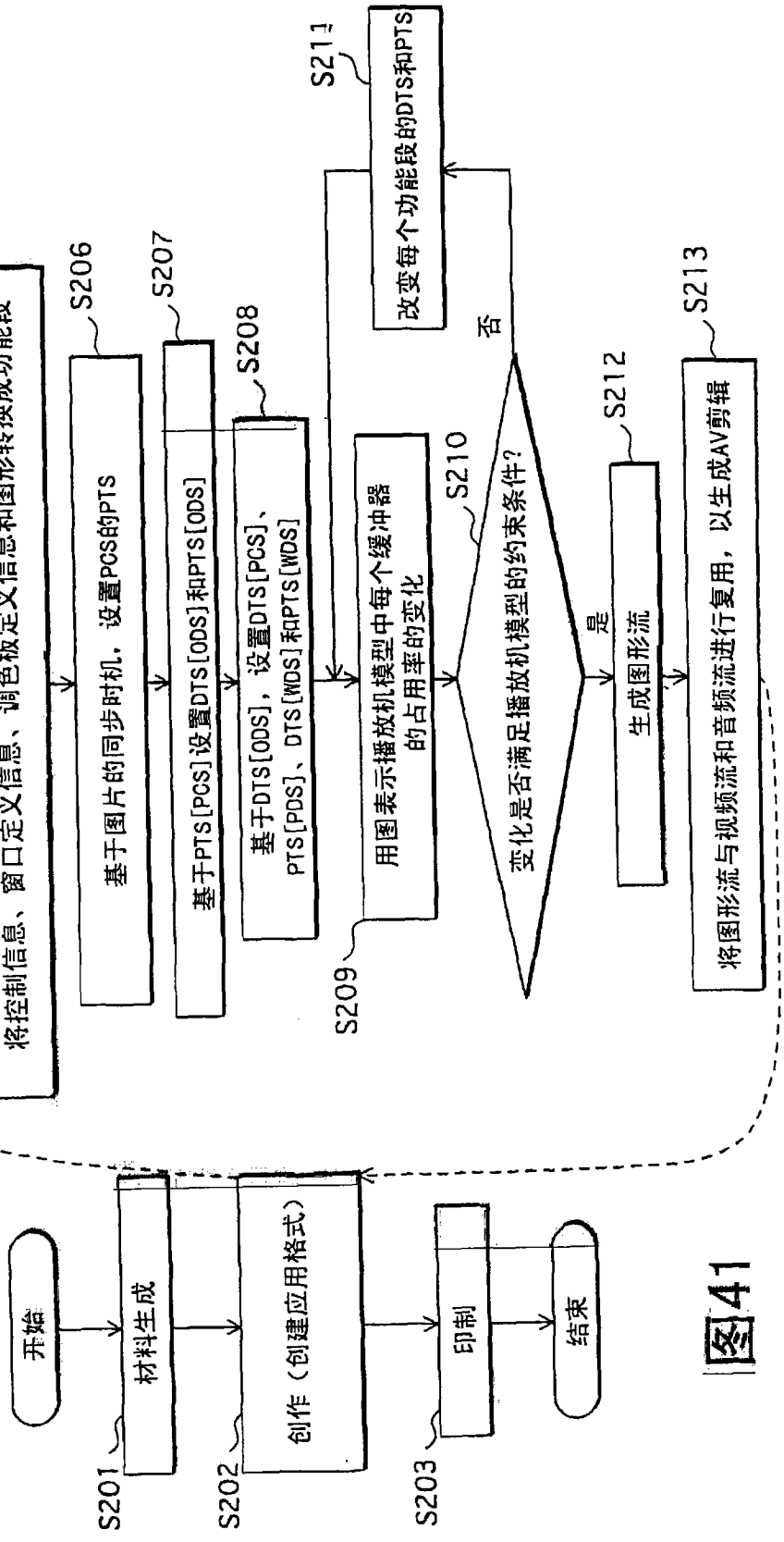

开始
材料生成
S201
创作（创建应用格式）
S202
印制
S203
结束
S204
写入控制信息、窗口定义信息、
调色板定义信息和图形
S205
将控制信息、窗口定义信息、调色板定义信息和图形转换成功能段
S206
基于图片的同步时机，设置PCS的PTS
S207
基于PTS[PCS]设置DTS[ODS]和PTS[ODS]
S208
基于DTS[ODS]，设置DTS[PCS]、
PTS[PDS]、DTS[WDS]和PTS[WDS]
S209
用图表示播放机模型中每个缓冲器
的占用率的变化
S210
变化是否满足播放机模型的约束条件?
否
是
S211
改变每个功能段的DTS和PTS
S212
生成图形流
S213
将图形流与视频流和音频流进行复用，以生成AV剪辑

图41