(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 106688251 B
(45)授权公告日 2019.10.01

(21)申请号 201580045969.3
(22)申请日 2015.07.27
(65)同一申请的已公布的文献号
申请公布号 CN 106688251 A
(43)申请公布日 2017.05.17
(30)优先权数据
62/031,723 2014.07.31 US
(85)PCT国际申请进入国家阶段日
2017.02.27
(86)PCT国际申请的申请数据
PCT/US2015/042190 2015.07.27
(87)PCT国际申请的公布数据
WO2016/018787 EN 2016.02.04
(73)专利权人 杜比实验室特许公司
地址 美国加利福尼亚
(72)发明人 T•J•埃格尔丁格 C•沃尔夫 A•C•诺埃尔 D•M•费舍尔 S•马蒂奈茨
(74)专利代理机构 中国国际贸易促进委员会专利商标事务所 11038
代理人 宋岩
(51)Int.Cl.
H04S 3/00(2006.01)
(56)对比文件
CN 103650539 A,2014.03.19,
CN 103354630 A,2013.10.16,
US 2006174267 A1,2006.08.03,
Achim Kuntz etc.Delay Handling in MPEG-H 3D audio.《109. MPEG MEETIING (MOTION PICTURE EXPERT GROUP OR ISO/IEC JTC1/SC29/WG11)》.2014,
审查员 刘柳群

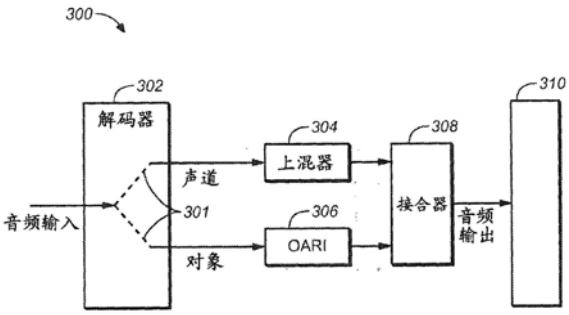
权利要求书3页 说明书18页 附图14页

(54)发明名称

音频处理系统和方法

(57)摘要

实施例针对通过以下方式对自适应音频内容进行处理,即,对于自适应音频比特流的每个音频段,确定音频类型是基于声道的音频和基于对象的音频中的一个;用指示对应音频段的音频类型的元数据定义对每个音频段进行标记;在声道音频渲染器部件中对被标记为基于声道的音频的音频段进行处理;并且在不同于声道音频渲染器部件的对象音频渲染器部件中对被标记为基于对象的音频的音频段进行处理。基于对象的音频通过对象音频渲染器接口渲染,所述对象音频渲染器接口基于元数据更新的定时和对齐以及最大/最小块大小参数来动态地调整对象音频段的处理块大小。



CN 106688251 B

1.一种对自适应音频内容进行处理的方法,包括:

对于包括多个音频段的自适应音频比特流的每个音频段,确定音频类型是基于声道的音频和基于对象的音频中的一个;

用指示对应音频段的音频类型的元数据定义对所述每个音频段进行标记;

在声道音频渲染器部件中对被标记为基于声道的音频的音频段进行处理;以及

在不同于声道音频渲染器部件的对象音频渲染器部件中对被标记为基于对象的音频的音频段进行处理,其中声道音频渲染器部件和对象音频渲染器部件具有不同的非零延时,并且,所述渲染器部件两者在它们首次初始化时被查询它们各自的、以采样计的延时以用于当在处理基于对象的音频段和基于声道的音频段之间切换时管理延时。

2.根据权利要求1所述的方法,还包括将所述元数据定义编码为音频类型元数据元素,所述音频类型元数据元素被编码为与每个音频段相关联的元数据有效载荷的一部分。

3.根据权利要求1所述的方法,其中,所述元数据定义包括二进制标志值,所述二进制标志值由解码器部件设置,并且被发送到声道音频渲染器部件和对象音频渲染器部件。

4.根据权利要求3所述的方法,其中,所述二进制标志值由声道音频渲染器部件和对象音频渲染器部件针对每个接收的音频段解码,并且其中,音频段中的音频数据由声道音频渲染器部件和对象音频渲染器部件中的一个基于解码的二进制标志值渲染。

5.根据权利要求1至4中的任何一项所述的方法,其中,所述基于声道的音频包括老式环绕声音频,所述声道音频渲染器部件包括上混器,并且进一步其中,所述对象音频渲染器部件包括对象音频渲染器接口。

6.一种对自适应音频进行渲染的方法,包括:

在解码器中接收输入音频,所述输入音频包括编码在音频比特流中的基于声道的音频段和基于对象的音频段;

在解码器中检测基于声道的音频段和基于对象的音频段之间的类型改变;

当检测到类型改变时,生成对于每种类型的音频段的元数据定义;

将所述元数据定义与合适的音频段相关联;以及

依赖于相关联的元数据定义在合适的后解码器处理部件中对每个音频段进行处理,其中每个后解码器处理部件具有不同于各个其他后解码器处理部件的非零延时,并且,后解码器处理部件在它们首次初始化时被查询它们各自的、以采样计的延时以用于当在处理基于对象的音频段和基于声道的音频段之间切换时管理延时。

7.根据权利要求6所述的方法,其中,所述基于声道的音频包括将通过自适应音频渲染系统的上混器渲染的老式环绕声音频,并且进一步其中,所述基于对象的音频通过所述自适应音频渲染系统的对象音频渲染器接口渲染。

8.根据权利要求6或权利要求7所述的方法,其中,所述元数据定义包括音频类型标志,所述音频类型标志被解码器编码为与所述音频比特流相关联的元数据有效载荷的一部分。

9.根据权利要求8所述的方法,其中,所述标志的第一状态指示相关联的音频段是基于声道的音频,所述标志的第二状态指示相关联的音频段是基于对象的音频。

10.一种对自适应音频进行渲染的系统,包括:

解码器,所述解码器接收具有音频内容和相关联的元数据的比特流中的输入音频,所述音频内容在任何一个时间具有以下音频类型:所述音频类型包括基于声道的音频或基于

对象的类型的音频中的一个；

上混器，所述上混器耦合到解码器，用于对基于声道的音频进行处理；

对象音频渲染器接口，所述对象音频渲染器接口与上混器并行地耦合到解码器，用于通过对象音频渲染器对基于对象的音频进行渲染；

元数据元素生成器，所述元数据元素生成器在解码器内，被配置为用第一元数据定义对基于声道的音频进行标记，并且用第二元数据定义对基于对象的音频进行标记；及

延时管理器，被配置为通过对初始化阶段期间的已知的延时差进行预先补偿来对任何两个连续的音频段之间的发送和处理延时进行调整，以便为该连续的音频段提供通过上混器和对象音频渲染器接口的不同信号路径的时间对齐的输出，其中上混器和对象音频渲染器两者具有不同的非零延时，并且上混器和对象音频渲染器在它们首次初始化时被查询它们的以采样计的延时。

11.根据权利要求10所述的系统，其中，所述上混器从解码器接收带标记的基于声道的音频和带标记的基于对象的音频两者，并且仅对基于声道的音频进行处理。

12.根据权利要求10所述的系统，其中，所述对象音频渲染器接口从解码器接收带标记的基于声道的音频和带标记的基于对象的音频两者，并且仅对基于对象的音频进行处理。

13.根据权利要求10所述的系统，其中，所述元数据元素生成器设置二进制标志，所述二进制标志指示从解码器发送到上混器和对象音频渲染器接口的音频段的类型，并且其中，所述二进制标志被解码器编码为与所述比特流相关联的元数据有效载荷的一部分。

14.根据权利要求10至13中的任何一项所述的系统，其中，所述基于声道的音频包括环绕声音频床，所述音频对象包括符合对象音频元数据格式的对象。

15.一种在基于声道的音频渲染和基于对象的音频渲染之间切换的方法，包括：

对于相关联的音频块，将元数据元素编码为具有指示基于声道的音频内容的第一状态或指示基于对象的音频内容的第二状态；

将所述元数据元素作为包括多个音频块的音频比特流的一部分发送到解码器；

在解码器中对每个音频块的所述元数据元素进行解码，如果所述元数据元素具有第一状态，则将基于声道的音频内容路由到声道音频渲染器，如果所述元数据元素具有第二状态，则将基于对象的音频内容路由到对象音频渲染器，其中声道音频渲染器和对象音频渲染器两者具有不同的非零延时，并且声道音频渲染器和对象音频渲染器在它们首次初始化时被查询它们的以采样计的延时以用于当在渲染基于对象的音频和基于声道的音频之间切换时管理延时。

16.根据权利要求15所述的方法，其中，所述元数据元素包括元数据标志，所述元数据标志与发送到解码器的脉冲编码调制音频比特流被带内发送。

17.根据权利要求15所述的方法，其中，所述声道音频渲染器包括上混器或直通节点中的一个，所述直通节点将基于声道的音频的输入声道映射到输出扬声器。

18.根据权利要求15至17中的任何一项所述的方法，其中，所述对象音频渲染器包括利用对象音频渲染器接口的渲染器，所述对象音频渲染器接口基于元数据更新的定时和对齐以及一个或多个其他的参数来动态地调整所述音频的处理块大小，所述一个或多个其他的参数包括最大块大小和最小块大小。

19.一种非暂时性计算机可读介质，包括存储在该非暂时性计算机可读介质上的指令，

所述指令当被执行时使得执行如权利要求1-9和权利要求15-18中任一项所述的方法。

20.一种音频处理设备,包括:

一个或多个处理器;以及

存储器,存储当被执行时使一个或多个处理器执行如权利要求1-9和权利要求15-18中任一项所述的方法的指令。

21.一种包括用于执行如权利要求1-9和权利要求15-18中任一项所述的方法的部件的装置。

音频处理系统和方法

[0001] 相关申请的交叉引用

[0002] 本申请要求2014年7月31日提交的美国临时专利申请No.62/031,723的优先权,该申请通过引用被整体结合于此。

技术领域

[0003] 一个或多个实现大体上涉及音频信号处理,更具体地涉及一种用于在基于声道的音频和基于对象的音频之间平滑地切换的方法以及用在自适应音频处理系统中的相关联的对象音频渲染器接口。

背景技术

[0004] 数字影院的引入和真实三维(“3D”)或虚拟3D内容的开发已经创建了声音的新标准(诸如多个声道的音频的合并)以允许内容创建者的更大创造力并且允许观众的听觉体验更包容和更真实。作为用于分布空间音频的手段,扩展超越传统的扬声器馈送和基于声道的音频是关键的,并且对于基于模型的音频描述一直存在相当大的兴趣,该基于模型的音频描述允许收听者选择期望的回放配置,其中音频是特别针对他们的选定配置进行渲染的。声音的空间表示利用音频对象,这些音频对象是带有相关联的、表观源位置(例如,3D坐标)、表观源宽度以及其他参数的参数源描述的音频信号。进一步的进展包括下一代空间音频(也称为“自适应音频”)格式已经被开发,该格式包括音频对象和传统的基于声道的扬声器馈送、以及音频对象的位置元数据的混合。在空间音频解码器中,声道被直接发送到它们相关联的扬声器,或者被下混到现存的扬声器组,并且音频对象被解码器以灵活的(自适应的)方式渲染。与每个对象相关联的参数源描述(诸如3D空间中的位置轨迹)与连接到解码器的扬声器的数量和位置一起被采用作为输入。然后渲染器利用某些算法(诸如平移定律)来在附接的扬声器组中分布与每个对象相关联的音频(“基于对象的音频”)。从而,每个对象的创作的空间意图通过存在于收听室中的特定扬声器配置最优地呈现。

[0005] 在传统的基于声道的音频系统中,音频后处理不会由于比特流内容改变而随时间改变。因为在整个系统中传载(carried)的音频一直是使用静态声道标识符(诸如左、右、中心等)来标识的,所以单独的音频后处理技术可以一直保持活动(active)。但是,基于对象的音频系统使用新的音频后处理机制,这些音频后处理机制使用专门的元数据来将基于对象的音频渲染到基于声道的扬声器布局。在实践中,基于对象的音频系统需要还支持并处理基于声道的音频,部分地是以便支持老式音频内容。因为基于声道的音频缺少使得能够进行音频渲染的专门的元数据,所以当编码的音频源包含基于对象的或基于声道的音频时,某些音频后处理技术可能是不同的。例如,上混器可以用于为扬声器生成在传入的基于声道的音频中不存在的内容,并且这样的上混器将不被应用于基于对象的音频。

[0006] 在大多数当前的系统中,音频节目通常仅包含一种类型的音频,或者是基于对象的音频,或者是基于声道的音频,因而处理链(渲染或上混)可以在初始化时被选择。但是,随着新的音频格式的出现,由于传输介质、创意选择、用户交互或其他类似的因素,节目中

的音频类型(声道或对象)可能随时间改变。在混合音频系统中,音频能够在不改变编解码器的情况下在基于对象的音频和基于声道的音频之间切换。在这种情况下,系统最优地是不表现出静音或音频延迟,而是通过在渲染的对象输出和上混的声道输出之间切换来将连续的音频流提供给它的所有扬声器输出,因为当前的音频系统中的一个问题是它们对于比特流中的这样的改变可能静音或者生成毛刺。

[0007] 对于具有对象和声道两者的自适应音频内容,现代的音频/视频接收器(AVR)系统(诸如可以利用**Dolby® Atmos®**技术或其他自适应音频标准的那些AVR系统)通常包括一个或多个数字信号处理器(DSP)芯片以及一个或多个微控制器芯片或单个芯片的一个或多个核(例如,片上系统,SoC)。微控制器负责管理DSP上的处理并且与用户进行交互,而DSP被专门优化来进行音频处理。当在基于对象的音频和基于声道的音频之间切换时,DSP能够发信号向微控制器通知改变,然后微控制器使用逻辑来将DSP重新配置为对新的音频类型进行处理。这种类型的信令被称为"带外"信令,因为它发生在DSP和微控制器之间。这样的带外信令由于诸如处理开销、发送延时、数据切换开销之类的因素必定会花费一定量的时间,并且如果DSP错误地处理音频数据,则这通常导致音频的不需要的静音,或者,可能生成毛刺。

[0008] 因而,所需的是在基于对象的内容和基于声道的内容之间切换的方式,该方式提供没有间隙、静音或毛刺的连续或平滑音频流。还需要的是允许音频处理DSP在无需与其他处理器或微控制器进行外部通信的情况下为传入的音频选择正确的处理链的机制。

[0009] 关于具有对象音频渲染器的对象音频渲染系统,基于对象的音频包括数字音频数据的部分(例如,PCM音频的采样),以及定义相关联的采样将怎样被渲染的元数据。因而,关于音频数据的对应采样对元数据更新进行合适定时对于音频对象的精确渲染是重要的。在具有许多对象和/或具有可以在声音空间周围快速移动的对象的动态音频节目中,元数据更新可以相对于音频帧速率非常快速地发生。当前的基于对象的音频处理系统通常能够对规律地并且以解码器和渲染处理器的处理能力之内的速率发生的元数据更新进行处理。这样的系统通常依赖于设定大小的音频帧以及以周期性均匀的速率应用的元数据更新。但是,当更新更快速地或者以周期性不均匀的方式发生时,对更新进行处理变得具有大得多的挑战性。通常,更新可能不与该更新所应用到的音频采样合适地对齐,这或者是因为更新发生得太快,或者是因为在元数据更新和对应的音频采样之间的同步错位(slip)。在这种情况下,音频采样可能根据不合适的元数据定义而渲染。

[0010] 还需要的是改变(adapt)编解码器解码的输出以便以可能的最高效的方式对用于自适应音频系统的元数据进行合适地缓冲和反序列化的机制。还需要的是这样的对象音频渲染器接口:该对象音频渲染器接口被配置为确保对象音频被以最少量的处理能力并且以高精度渲染,而且还可以针对客户需要进行调整,这依赖于它们的芯片架构。

[0011] 背景部分中讨论的主题不应该仅由于该主题在背景部分中被提及就被假定为是现有技术。类似地,在背景部分中提及的或者与背景部分的主题相关联的问题不应该被假定为以前已经在现有技术中被认识到。背景部分中的主题仅代表不同的方法,这些方法本身也可以是发明。Dolby、Dolby Digital Plus、Dolby TrueHD以及Atmos是杜比实验室许可公式(Dolby Laboratories Licensing Corporation)的商标。

发明内容

[0012] 实施例针对一种通过以下方式对自适应音频内容进行处理的方法，即，对于自适应音频比特流的每个音频段，确定音频类型是基于声道的还是基于对象的；使用指示对应音频段的音频类型的元数据定义对每个音频段进行标记；在声道音频渲染器部件中对被标记为基于声道的音频的音频段进行处理；并且在与基于声道的音频渲染器部件不同的对象音频渲染器部件中对被标记为基于对象的音频的音频段进行处理。所述方法还包括将所述元数据定义编码为音频类型元数据元素，该音频类型元数据元素被编码为与每个音频段相关联的元数据有效载荷的一部分。所述元数据定义可以包括二进制标志值，该二进制标志值由解码器设置，并且被发送到声道音频渲染器部件和对象音频渲染器部件。对于该实施例，二进制标志值由声道音频渲染器部件和对象音频渲染器部件针对每个接收的音频段解码，并且音频段中的音频数据由声道音频渲染器部件和对象音频渲染器部件中的一个基于解码的二进制标志值渲染。基于声道的音频可以包括立体声或老式环绕声音频，并且声道音频渲染器部件可以包括上混器或简单的直通节点，对象音频渲染器部件可以包括对象音频渲染器接口。所述方法还可以包括通过对初始化阶段期间的已知的延时差进行预先补偿来对任何两个连续的音频段之间的发送和处理延时进行调整。

[0013] 实施例还针对一种通过以下方式对自适应音频进行渲染的方法：在解码器中接收输入音频，该输入音频包括被编码在音频比特流中的基于声道的音频段和基于对象的音频段；在解码器中检测基于声道的音频段和基于对象的音频段之间的类型改变；当检测到类型改变时对每种类型的音频段生成元数据定义；将元数据定义与合适的音频段相关联；并且依赖于相关联的元数据定义在合适的后解码器处理部件中对每个音频段进行处理。基于声道的音频可以包括将通过自适应音频渲染系统的上混器渲染的老式环绕声音频，并且基于对象的音频可以通过系统的对象音频渲染器接口渲染。在实施例中，该方法还包括通过对初始化阶段期间的已知的延时差进行预先补偿来对任何两个连续的音频段之间的处理延时进行调整。用于所述方法的元数据定义可以包括音频类型标志，该音频类型标志被解码器编码为与音频比特流相关联的元数据有效载荷的一部分。对于该实施例，该标志的第一状态指示相关联的音频段是基于声道的音频，该标志的第二状态指示相关联的音频段是基于对象的音频。

[0014] 实施例还针对一种自适应音频渲染系统，该自适应音频渲染系统具有：解码器，该解码器接收具有音频内容和相关联的元数据的输入音频比特流，所述音频内容在任何时间具有这样的音频类型：所述音频类型包括基于声道的音频或基于对象的类型的音频中的一个；上混器，该上混器耦合到解码器，用于对基于声道的音频进行处理；对象音频渲染器接口，该对象音频渲染器接口与上混器并行地耦合到解码器，用于通过对象音频渲染器对基于对象的音频进行渲染；以及元数据元素生成器，该元数据元素生成器在解码器内，被配置为用第一元数据定义对基于声道的音频进行标记，并且用第二元数据定义对基于对象的音频进行标记。在该系统中，上混器从解码器接收带标记的基于声道的音频和带标记的基于对象的音频两者，并且仅对基于声道的音频进行处理；对象音频渲染器接口从解码器接收带标记的基于声道的音频和带标记的基于对象的音频两者，并且仅对基于对象的音频进行处理。元数据元素生成器可以被配置为设置二进制标志，该二进制标志指示从解码器发送到上混器和对象音频渲染器接口的音频段的类型，并且其中，该二进制标志被解码器编码

为与比特流相关联的元数据有效载荷的一部分。基于声道的音频可以包括环绕声音频床，音频对象可以包括符合对象音频元数据(OAMD)格式的对象。在实施例中，所述系统还包括延时管理器，该延时管理器被配置为通过对初始化阶段期间的已知的延时进行预先补偿来对任何两个连续的音频段之间的延时进行调整以便为该连续的音频段提供通过上混器和对象音频渲染器接口的不同信号路径的时间对齐的输出。在一些实施例中，可以使用将输入音频声道映射到输出扬声器的简单的直通节点替换上混器。

[0015] 实施例还针对一种通过以下方式对基于对象的音频进行处理的方法：在对象音频渲染器接口(OARI)中接收音频采样块以及一个或多个相关联的对象音频元数据有效载荷；从每个对象音频元数据有效载荷反序列化一个或多个音频块更新；将音频采样和音频块更新存储在相应的音频采样存储器高速缓存和音频块更新存储器高速缓存中；并且基于音频块更新相对于处理块边界的定时和对齐以及一个或多个其他的参数来动态地选择音频采样的处理块大小，所述一个或多个其他的参数包括最大/最小处理块大小参数。所述方法还可以包括在通过动态选择确定的大小的处理块中将基于对象的音频从OARI发送到OAR。每个元数据元素在具有采样偏移的元数据帧中被传递，所述采样偏移指示该帧应用于音频块中的哪个采样处。该方法还可以包括通过一个或多个过程准备包含元数据元素的元数据，所述一个或多个过程包括对象优先化、宽度移除、禁用对象处理、过于频繁的更新的过滤、到期望范围的空间位置裁剪以及将更新数据转换为期望格式。OAR可以支持有限数量的处理块大小，诸如长度为32、64、128、256、480、512、1024、1536或2048个采样，但不限于此。在实施例中，做出处理块大小选择使得音频块更新被定位于处理块大小选择参数所允许的处理块的第一采样附近。处理块大小可以被选择为在音频块更新位置、OAR处理块大小以及OARI最大块大小参数值和最小块大小参数值的约束下尽可能大。元数据帧可以包含元数据，该元数据定义关于音频采样块中的一个或多个对象的渲染的属性，这些属性是从包括以下各项的组选择的：内容类型属性，包括对话、音乐、效果、拟音(Foley)、背景以及周围环境定义；空间属性，包括3D位置、对象大小以及对象速率；以及扬声器渲染属性，包括对准(snap)到扬声器位置、声道权重、增益、斜坡以及低音管理信息。

[0016] 实施例还针对一种通过以下方式对音频对象进行处理的方法：在对象音频渲染器接口(OARI)中接收音频采样块以及相关联的元数据，该元数据定义音频采样在对象音频渲染器(OAR)中怎样被渲染，其中，该元数据随着时间被更新以定义音频对象的不同渲染属性；把将被OAR处理的音频采样和元数据更新缓冲到处理块布置中；基于元数据更新相对于块边界的定时和对齐以及一个或多个其他的参数来动态地选择处理块大小，所述一个或多个其他的参数包括：最大/最小块大小参数；并且在通过动态选择步骤确定的大小的块中将基于对象的音频从OARI发送到OAR。该方法还可以包括将对于每个块的音频数据和块更新存储在相应的音频存储器高速缓存和更新存储器高速缓存中，其中，块更新被编码在存储在对象音频元数据有效载荷中的元数据元素中。每个元数据元素可以在具有采样偏移的元数据帧中被传递，所述采样偏移指示该帧应用于处理块中的哪个采样处。可以做出处理块大小选择使得音频块更新被定位于最小输出块大小选择所允许的块的第一采样附近。在实施例中，块大小可以被选择为在块更新位置、OAR块大小以及OARI最大块大小参数值的约束下尽可能大。该方法还可以包括通过一个或多个过程准备包含元数据元素的元数据，该一个或多个过程包括对象优先化、宽度移除、禁用对象处理、过于频繁的更新的过滤、到期望

范围的空间位置裁剪以及将更新数据转换为期望格式。

[0017] 实施例又针对一种通过以下方式对自适应音频数据进行处理的方法:通过定义的元数据定义来确定将被处理的音频是基于声道的音频还是基于对象的音频;如果是基于声道的,则通过基于声道的音频渲染器(CAR)对该音频进行处理;如果是基于对象的,则通过基于对象的音频渲染器(OAR)对该音频进行处理,其中,OAR利用OAR接口(OARI),OAR接口基于元数据更新的定时和对齐以及一个或多个其他的参数来动态地调整音频的处理块大小,该一个或多个其他的参数包括最大块大小和最小块大小。

[0018] 实施例还针对一种通过以下方式在基于声道的或基于对象的音频渲染之间切换的方法:将具有第一状态或第二状态的元数据元素与音频块一起编码,其中第一状态指示基于声道的音频内容,第二状态指示基于对象的音频内容;将该元数据元素作为音频比特流的一部分发送到解码器;在解码器中对该元数据元素进行解码,如果该元数据元素具有第一状态,则将基于声道的音频内容路由到声道音频渲染器(CAR),如果该元数据元素具有第二状态,则将基于对象的音频内容路由到对象音频渲染器(OAR)。在该方法中,所述元数据元素包括元数据标志,该元数据标志与发送到解码器的脉冲编码调制(PCM)音频比特流一起在带内发送。CAR可以包括上混器或直通节点中的一个,所述直通节点将基于声道的音频的输入声道映射到输出扬声器;OAR包括利用OAR接口(OARI)的渲染器,OAR接口基于元数据更新的定时和对齐以及一个或多个其他的参数来动态地调整音频的处理块大小,该一个或多个其他的参数包括最大块大小和最小块大小。

[0019] 实施例又针对实现前述方法的数字信号处理系统和/或合并实现前述方法中的至少一些的电路的扬声器系统。

[0020] 通过引用并入

[0021] 本说明书中提及的每个出版物、专利和/或专利申请整体通过引用结合于此,达到如同每个单独的出版物和/或专利申请被特别地且单独地指示为通过引用结合一样的程度。

附图说明

[0022] 在以下附图中,相同的附图标记用于指代相同的元件。虽然以下附图描绘了各种示例,但是该一个或多个实现不限于附图中描绘的示例。

[0023] 图1示出了提供用于高度声道回放的高度扬声器的环绕系统(例如,9.1环绕)中的示例扬声器放置。

[0024] 图2示出了根据实施例的产生自适应音频混合的基于声道和对象的数据的组合。

[0025] 图3是根据实施例的对基于声道的和基于对象的音频进行处理的自适应音频系统的框图。

[0026] 图4A示出了根据实施例的自适应音频AVR系统中的用于基于声道的解码和上混的处理路径。

[0027] 图4B示出了根据实施例的图4A的自适应音频AVR系统中的用于基于对象的解码和渲染的处理路径。

[0028] 图5是示出了根据实施例的提供带内信令元数据以在基于对象的音频数据和基于声道的音频数据之间切换的方法的流程图。

[0029] 图6示出了根据实施例的、将元数据组织到被对象音频渲染器处理的分层结构中。

[0030] 图7示出了根据实施例的第一类型的编解码器内的元数据更新的应用以及元数据更新的组帧。

[0031] 图8示出了根据可替代实施例的第二类型的编解码器内的元数据更新的应用以及元数据更新的组帧。

[0032] 图9是示出根据实施例的、对象音频渲染器接口进行的过程步骤的流程图。

[0033] 图10示出了根据实施例的、对象音频渲染器接口的高速缓存和反序列化处理循环(cycle)。

[0034] 图11示出了根据实施例的、对象音频渲染器接口对于元数据更新的应用。

[0035] 图12示出了根据实施例的、对象音频渲染器接口进行的初始处理循环的示例。

[0036] 图13示出了在图12的示例处理循环之后的后续处理循环。

[0037] 图14示出了列出根据实施例的在计算内部数据结构中的偏移字段时使用的字段的表格。

具体实施方式

[0038] 描述了用于在自适应音频系统中在基于对象的音频和基于声道的音频之间切换的系统和方法,所述自适应音频系统允许没有间隙、静音或毛刺地回放连续音频流。还描述了关于相关联的对象音频渲染器接口的实施例,该对象音频渲染器接口生成动态选择的处理块大小以优化处理器效率和存储器使用、同时在自适应音频处理系统的对象音频渲染器中保持对象音频元数据与对象音频PCM数据的合适对齐。本文中所描述的一个或多个实施例的各方面可以在包括执行软件指令的一个或多个计算机或处理设备的混合、渲染和回放系统中的对源音频信息进行处理的音频或视听系统中实现。所描述的实施例中的任何一个都可以单独使用,或与另一实施例以任何组合一起使用。虽然各种实施例可能是由在本说明书中的一个或多个地方可能讨论的或暗指的现有技术的各种缺陷激发的,但是实施例不必需解决这些缺陷中的任何一个。换言之,不同实施例可以解决在本说明书中可能讨论的不同缺陷。一些实施例可以仅部分解决在本说明书中可能讨论的一些缺陷或者仅一个缺陷,一些实施例可以不解决这些缺陷中的任何一个。

[0039] 为了本说明书的目的,以下术语具有相关联的意义:术语“声道”意指音频信号加上元数据,在该元数据中,位置被编码为声道标识符,例如,左前或右上环绕;“基于声道的音频”是为通过预定义的具有相关联的标称位置(例如,5.1、7.1等)的一组扬声器区域回放而格式化的音频;术语“对象”或“基于对象的音频”意指具有参数源描述(诸如表观源位置(例如,3D坐标)、表观源宽度等)的一个或多个音频声道;“自适应音频”意指基于声道的和/或基于对象的音频信号加上元数据,其基于回放环境、使用音频流加上在其中位置被编码为空间中的3D位置的元数据来对音频信号进行渲染;术语“自适应流传输”是指可以自适应地改变(例如,从基于声道的变为基于对象的,或者再次变回来)的并且是在线流传输应用中常用的音频类型,在该在线流传输应用中,音频的格式必须针对变化的带宽约束进行缩放(即,随着对象音频趋向于达到更高数据速率,在更低带宽条件下回退(fallback)的通常是基于声道的音频);“收听环境”意指任何打开的、部分封闭的或完全封闭的区域(诸如可以用于单独回放音频内容或者与视频或其他内容一起回放音频内容的房间)并且可以在家

庭、影院、剧院、礼堂、工作室、游戏控制台等中实现。

[0040] 自适应音频格式和系统

[0041] 在实施例中，互连系统实现为被配置为与声音格式和处理系统一起工作的音频系统(可以称为“空间音频系统”、“混合音频系统”或“自适应音频系统”)的一部分。这样的系统是基于音频格式和渲染技术来使得观众沉浸程度增强、艺术控制更大、系统具有灵活性和可扩展性。整个自适应音频系统通常包括音频编码、分布和解码系统，该系统被配置为生成一个或多个比特流，该比特流包含传统的基于声道的音频元素和音频对象编码元素(基于对象的音频)两者。与单独采用基于声道的方法或基于对象的方法相比，这样的组合方法提供更大的编码效率和渲染灵活性。

[0042] 自适应音频系统和相关联的音频格式的示例实现是Dolby® Atmos®平台。这样的系统合并有可以实现为9.1环绕系统的高度(上/下)维度或类似环绕声配置。这样的基于高度的系统可以用不同的命名来指定，其中，通过x.y.z指定使高度扬声器区别于地板扬声器，其中，x是地板扬声器的数量，y是次低音扬声器(subwoofer)的数量，z是高度扬声器的数量。因而，9.1系统可以被称为5.1.4系统，该系统包括5.1系统以及4个高度扬声器。

[0043] 图1示出当前的提供用于高度声道回放的高度扬声器的环绕系统(例如，5.1.4环绕)中的扬声器放置。系统100的扬声器配置包括地板平面中的五个扬声器102以及高度平面中的四个扬声器104。通常，这些扬声器可以用于产生被设计为或多或少地精确地从房间内的任何位置发出的声音。预定义的扬声器配置(诸如图1所示的那些扬声器配置)可以自然地限制精确地表示给定声源的位置的能力。例如，声源的向左平移不能远于左扬声器本身。这适用于每一个扬声器，因而形成一维(例如，左-右)、二维(例如，前-后)或三维(例如，左-右、前-后、上-下)几何形状，在该几何形状中，下混受到约束。各种不同的扬声器配置和类型可以用在这样的扬声器配置中。例如，某些增强音频系统可以使用呈9.1、11.1、13.1、19.4或其他配置的扬声器，诸如x.y.z配置指定的那些。扬声器类型可以包括全范围直接扬声器、扬声器阵列、环绕扬声器、次低音扬声器、高音扬声器以及其他类型的扬声器。

[0044] 音频对象可以被认为是可以被感知为从收听环境中的特定的一个物理位置或多个物理位置发出的声音元素组。这样的对象可以是静态的(即，静止的)或动态的(即，移动的)。音频对象由元数据控制，元数据定义声音在给定时间点的位置，以及其他。当对象被回放时，它们是使用存在的扬声器、根据位置元数据渲染的，而不一定被输出到预定义的物理声道。会话中的轨道可以是音频对象，并且标准平移数据类似于位置元数据。这样，放置在屏幕上的内容可以有效地以与基于声道的内容相同的方式平移，但是放置在环绕中的内容可以被渲染到单个的扬声器，如果期望的话。虽然音频对象的使用提供了对离散效果的期望的控制，但是音轨(soundtrack)的其他方面可以在基于声道的环境中有效地工作。例如，许多环境效果或混响实际上得益于被馈送到扬声器阵列。虽然这些可以被看作具有足以填充阵列的宽度的对象，但是保留一些基于声道的功能是有益的。

[0045] 自适应音频系统被配置为除了音频对象之外还支持音频床，其中，床是有效地基于声道的子混合或干支(stem)。这些床可以被递送以用于依赖于内容创建者的意图而最终被单独回放(渲染)或者被组合成单个床而回放。这些床可以在不同的基于声道的配置(诸如，5.1、7.1和9.1)以及诸如图1所示的包括高架扬声器的阵列中创建。图2示出根据实施例的用于产生自适应音频混合的、基于声道的数据和基于对象的数据的组合。如过程200中所

示，基于声道的数据202(例如可以是以脉冲编码调制(PCM)数据的形式提供的5.1或7.1环绕声数据)与音频对象数据204组合，产生自适应音频混合208。音频对象数据204是通过将原始的基于声道的数据的元素与相关联的元数据组合而产生的，该相关联的元数据指定关于音频对象的位置的某些参数。如图2中概念性地示出的，创作工具提供创建同时包含扬声器声道组与对象声道的组合的音频节目的能力。例如，音频节目可以包含可选地组织成组(或轨道，例如，立体声或5.1轨道)的一个或多个扬声器声道、关于一个或多个扬声器声道的描述性元数据、一个或多个对象声道以及关于一个或多个对象声道的描述性元数据。

[0046] 对于自适应音频混合208，回放系统可以被配置为对通过一个或多个捕捉部件、预处理部件、创作部件和编码部件生成的音频内容进行渲染和回放，其中该编码部件将输入音频编码为数字比特流。自适应音频部件可以用于通过对输入音频的分析来自动地生成合适的元数据，该分析是通过对诸如源分离和内容类型的因素进行检查来进行的。例如，可以通过对于声道对(channel pairs)之间的相关的输入的相对水平的分析从多声道记录得出位置元数据。内容类型(诸如语音或音乐)的检测可以例如通过特征提取和分类来实现。某些创作工具允许通过优化录音师的创作意图的输入和汇编来创作音频节目，使得他可以一次就创建针对几乎任何回放环境中的回放进行优化的最终的音频混合。这可以通过使用音频对象以及相关联的与原始音频内容一起编码的位置数据来实现。一旦自适应音频内容已经被创作并且在合适的编解码器设备中被编码，它就被解码和渲染以便通过诸如图1所示的扬声器回放。

[0047] 图3是根据实施例的对基于声道的和基于对象的音频进行处理的自适应音频系统的框图。如系统300中所示，输入音频(包括基于对象的包括对象元数据的音频以及基于声道的音频)作为输入音频比特流(音频输入)被输入到解码/渲染(解码器)子系统302内的一个或多个解码器电路。比特流中的音频对具有相关联的扬声器或声道标识符的诸如声道(音频床)的各种音频分量以及具有相关联的对象元数据的各种音频对象(例如，静态的或动态的对象)进行编码。在实施例中，在任何特定时间只有一种类型的音频(对象或声道)被输入，但是音频输入流可以在节目的过程期间周期性地或者有点频繁地在这两种类型的音频内容之间切换。基于对象的流可以包含声道和对象两者，对象可以是不同的类型：床对象(即，声道)、动态对象以及ISF(中间空间格式)对象。ISF是通过将平移操作划分为以下两个部分来优化音频对象平移器的操作的格式：时变部分和静态部分，其他类似的对象也可以被所述系统处理。OAR同时对所有这些类型进行处理，而CAR用于进行老式的基于声道的内容的盲上混或者用作直通节点。

[0048] 解码器302之后的音频处理对于基于声道的音频与基于对象的音频通常是不同的。因而，对于图3的实施例，基于声道的音频被示为是通过上混器304或其他基于声道的音频处理器处理的，而基于对象的音频被示为是通过对象音频渲染器接口(OARI)306处理的。CAR部件可以包括如所示的上混器，或者它可以包括将输入音频声道映射到输出扬声器的简单的直通节点，或者它可以是任何其他合适的基于声道的处理部件。然后，处理后的音频在接合器(joiner)部件308或类似的组合电路中被复用或接合在一起，然后所得出的音频输出被发送到扬声器阵列(诸如图1的阵列100)中的合适的一个或多个扬声器310。

[0049] 对于图3的实施例，音频输入可以包括声道和对象，以及它们各自的相关联的元数据或标识符数据。当编码的音频比特流输入到解码器302时，该音频比特流因而包含两种类

型的音频数据。在实施例中，解码器302包含切换机制301，切换机制301利用带内信令元数据来在基于对象的音频数据和基于声道的音频数据之间切换以使每种特定类型的音频内容被路由到合适的处理器304或306。通过使用这样的信令元数据，编码的音频源可以发信号通知基于对象的音频和基于声道的音频之间的切换301。在实施例中，信令元数据信号与音频输入比特流“带内”地发送，并且用于启动下游过程，诸如音频渲染306或上混304。这使得实现没有间隙、静音、毛刺或音频/视频同步漂移的连续音频流。在初始化时，解码器302准备好对基于对象的音频和基于声道的音频这二者进行处理。当在音频类型之间发生改变时，元数据在解码器DSP的内部生成，并且在音频处理块之间发送。通过利用该元数据，可以允许DSP在无需与其他DSP或微控制器进行外部通信的情况下为传入的音频选择正确的处理链。这允许编码的音频源通过与音频内容一起发送的元数据信号来发信号通知基于对象的音频和基于声道的音频之间的切换。

[0050] 图4A和图4B示出根据实施例的自适应音频AVR系统中的、基于对象的解码和渲染与基于声道的解码和上混所穿过的不同的处理路径。图4A示出自适应音频AVR系统中的用于基于声道的解码和上混的处理路径和信号流，图4B示出相同的AVR系统中的用于基于对象的解码和渲染的处理路径和信号流。输入比特流(可以是Dolby Digital Plus或类似比特流)可以随时间在基于对象的内容和基于声道的内容之间改变。当内容改变时，解码器402(例如，Dolby Digital Plus解码器)被配置为输出对音频配置(对象对声道)进行编码或者指示的带内元数据。如图4A中所示，输入比特流内的基于声道的音频是通过上混器404处理的，上混器404还接收扬声器配置信息；如图4B中所示，输入比特流内的基于对象的音频是通过对象音频渲染器(OAR)406处理的，OAR 406还接收合适的扬声器配置信息。OAR通过图3中所示的对象音频渲染器接口(OARI)306与AVR系统411进行接口。与音频内容一起编码的并且对音频类型进行编码的带内元数据的使用允许上混器404和渲染器406选择合适的音频进行处理。因而，如图4A和图4B所示，上混器404将通过内联(in-line)元数据来检测基于声道的音频的存在，并且仅对基于声道的音频进行处理，而忽略基于对象的音频。同样地，渲染器406将通过内联元数据来检测基于对象的音频的存在，并且仅对基于对象的音频进行处理，而忽略基于声道的音频。该内联元数据有效地允许系统直接基于后解码器处理部件检测的音频内容的类型而在合适的后解码器处理部件(例如，上混器、OAR)之间切换，如虚拟开关403所示。

[0051] 当在渲染的音频(基于对象的)和上混的音频(基于声道的)之间切换时，管理延时也是重要的。上混器404和渲染器406两者可能具有不同的非零延时。如果延时不被考虑，则音频/视频同步可能受到影响，并且音频毛刺可能被感知到。延时管理可以单独处理，或者可以由渲染器或上混器处理。当渲染器或上混器首次被初始化时，诸如通过每个部件内的延时确定算法来查询每个部件的、以采样计的延时。当渲染器或上混器变为活动(active)时，通过部件算法生成的等于其延时的初始采样被丢弃。当渲染器或上混器变为不活动时，等于其延时的额外数量的零采样被处理。因而，输出的采样的数量精确地等于输入的采样的数量。没有前导零输出，并且没有过期数据留在部件算法中。这样的管理和同步是由系统400和411中的延时管理部件408提供的。延时管理器408还负责将上混器404和渲染器406的输出接合为一个连续音频流。在实施例中，可以通过根据延时处理规则对各相应的接收的音频段丢弃前导零并处理额外的数据而在上混器和渲染器两者的内部对实际的延时管理

功能进行处理。延时管理器因而确保不同信号路径的输出在时间上对齐。这允许所述系统在不产生可听到的且令人不快的伪影的情况下对比特流改变进行处理，否则可能会由于多个回放条件以及比特流中的改变的可能性而产生所述伪影。

[0052] 在实施例中，延时对齐通过对初始化阶段期间的已知的延时差进行预先补偿来发生。在连续的音频段期间，可能因为音频未对齐到最小帧边界大小(例如，在声道音频渲染器中)或者系统正在应用“淡化”来最小化瞬变而掉落采样。如图4A和图4B中所示，然后延时同步的音频通过一个或多个附加的后处理410被处理，后处理410可以利用启用自适应音频的扬声器的信息，该信息提供关于声音转向(steering)、对象轨迹、高度效果等的参数。

[0053] 在实施例中，为了使得能够关于比特流参数进行切换，上混器404必须保持在存储器中初始化。这样，当自适应音频内容的丢失被检测到时，上混器可以立即开始对基于声道的音频进行上混。

[0054] 图5是示出根据实施例的提供带内信令元数据以在基于对象的音频数据和基于声道的音频数据之间切换的方法的流程图。如图5的过程550中所示，具有在不同时间的基于声道的音频和基于对象的音频的输入比特流在解码器中接收，502。当解码器接收到比特流时，解码器检测音频类型的改变，504。解码器在内部生成指示音频的每个接收的段的音频类型的元数据，并且将该生成的元数据与音频的每个段一起编码以用于发送到下游处理器或处理块，506。因而，每个基于声道的音频段均与声道标识元数据定义一起编码(被标记为基于声道的)，每个基于对象的音频段均与对象标识元数据定义一起编码(被标记为基于对象的)。解码器之后的每个处理块都基于该内联信令元数据来检测传入的音频信号段的类型，并且相应地对它进行处理或者忽略它，508。因而，上混器或其他类似的过程将对被发信号通知是基于声道的音频段进行处理，OAR或类似过程将对被发信号通知是基于对象的音频段进行处理。连续的音频段之间的任何延时差通过系统内的或者每个下游处理块内的延时管理过程进行调整，并且音频流被接合以形成输出音频流，510。然后输出流被发送到环绕声扬声器阵列，512。

[0055] 通过利用带内元数据信令机制并且通过管理延时，图3的音频系统能够接收并处理随时间在对象和声道之间改变的音频，并且对于所有被请求的扬声器馈送保持恒定的音频输出，没有毛刺、静音或音频/视频同步漂移。这允许对在相同的比特流中包含新的(例如，Dolby Atmos音频/视频)内容和老式的(例如，环绕声音频)内容两者的音频内容进行分布和处理。通过使用合适的上混器304，AVR或其他设备可以在内容类型之间进行切换，引起最小的空间失真。这允许新开发的AVR产品能够接收比特流中(诸如比特率和声道配置)的改变，而不会导致任何的音频遗失或不期望的音频伪影，当产业走向新形式的内容递送和自适应流传输场景时，这是特别重要的。所描述的环绕上混技术对于帮助解码器处理这些比特流改变起重要作用。

[0056] 应该注意到，如在图4A和图4B中进一步详述的图3的系统代表了用于自适应音频的回放系统的示例，其他的配置、部件和互连也是可能的。例如，解码器302可以实现为耦合到用于上混和对象渲染的两个单独的处理器(DSP)的微控制器，这些部件可以实现为通过物理传输接口或网络耦合到一起的单独的设备。解码器微控制器和处理DSP均可以包含在单独的部件或子系统内，或者它们可以是包含在相同的子系统中的单独的部件，诸如集成的解码器/渲染器部件。可替代地，解码器和后解码器过程可以实现为单片集成电路设备内

的单独的处理部件。

[0057] 元数据定义

[0058] 在实施例中，自适应音频系统包括从原始的空间音频格式生成元数据的部件。所描述的系统的方法和部件包括音频渲染系统，该音频渲染系统被配置为对包含传统的基于声道的音频元素以及音频对象编码元素两者的一个或多个比特流进行处理。来自空间音频处理器的空间音频内容包括音频对象、声道以及位置元数据。元数据是在音频工作站中响应于工程师的混合输入而生成的，以提供控制空间参数(例如，位置、速率、强度、音色等)的渲染队列并且指定收听环境中的哪个(哪些)驱动器或扬声器在展示期间播放相应的声音。元数据在工作站中被与相应的音频数据相关联以被音频处理器进行封装和输送。

[0059] 在实施例中，音频类型(即，基于声道的音频或基于对象的音频)元数据定义被添加到作为被自适应音频处理系统处理的音频比特流的一部分而发送的元数据有效载荷，被编码在该元数据有效载荷内，或者被以其他的方式与该元数据有效载荷相关联。通常，用于自适应音频的创作和分布系统创建并递送允许经由固定扬声器位置(左声道、右声道等)以及基于对象的音频元素回放的音频，所述基于对象的音频元素具有广义3D空间信息，包括位置、大小和速率。系统通过元数据来提供关于音频内容的有用信息，所述元数据由内容创建者在内容创建/创作时与音频本体(essence)配对。元数据因而对关于在渲染期间可以使用的音频属性的详细信息进行编码。这样的属性可以包括内容类型(例如，对话、音乐、效果、拟音、背景/周围环境等)以及音频对象信息(诸如空间属性(例如，3D位置、对象大小、速率等))和有用的渲染信息(例如，对准到扬声器位置、声道权重、增益、斜坡、低音管理信息等)。音频内容和再现意图元数据可以或者由内容创建者手动创建，或者通过使用自动的媒体智能算法来创建，所述媒体智能算法可以在创作过程期间在后台运行，并且可以在最终质量控制阶段期间被内容创建者审阅，如果期望的话。

[0060] 在实施例中，存在一起工作来描述数据的数种不同的元数据类型。首先，在各个处理节点之间(诸如在解码器和上混器或渲染器之间)存在连接。该连接包含数据缓冲器和元数据缓冲器。如下面关于OARI更详细地描述的，元数据缓冲器实现为列表，该列表具有指向数据缓冲器的某些字节偏移的指针。节点到该连接的接口是通过“管脚”。节点可以具有零个或更多个输入管脚以及零个或更多个输出管脚。在一个节点的输入管脚和另一节点的输出管脚之间建立连接。管脚的一个特质是它的数据类型。也就是说，连接中的数据缓冲器可以表示各种不同类型的数据——PCM音频、编码的音频、视频等。节点的责任是通过它的输出管脚指示什么类型的数据正在被输出。处理节点还应该对它的输入管脚进行查询，使得它知道什么类型的数据正被处理。

[0061] 一旦节点对它的输入管脚进行了查询，然后它就可以决定怎样对传入的数据进行处理。如果传入的数据是PCM音频，则该节点需要准确地知道该PCM音频的格式是什么。音频的格式用“pcm_config”元数据有效载荷结构描述。该结构描述例如PCM音频的声道计数、步幅以及声道分配。它还包含标志“object_audio”，该标志如果被设置为1，则指示PCM音频是基于对象的，或者如果PCM音频是基于声道的，则被设置为0，但是其他标志设置值也是可能的。在实施例中，该pcm_config结构由解码器节点设置，并且被OARI节点和CAR节点两者接收。当渲染节点接收到pcm_config元数据更新时，它对object_audio标志进行检查，并且相应地做出反应，根据需要开始新的流或者结束当前流。

[0062] 许多其他的元数据类型可以由音频处理框架定义。通常，元数据包括标识符、有效载荷大小、到数据缓冲器中的偏移以及可选的有效载荷。许多元数据类型没有任何实际的有效载荷，纯粹是信息性的。例如，“sequence start(序列开始)”和“sequence end(序列结束)”信令元数据没有有效载荷，因为它们只是信号，没有另外的信息。实际的对象音频元数据被携带在“演变(evolution)”帧中，并且用于演变的元数据类型具有与演变帧的大小相等的有效载荷大小，该有效载荷大小不是固定的，可以从帧到帧改变。术语演变帧通常是指安全的可扩展的元数据封装和递送框架，在该框架中，帧可以包含一个或多个元数据有效载荷以及相关联的定时和安全信息。虽然关于演变帧描述了实施例，但是应该注意到，可以使用提供类似能力的任何合适的帧配置。

[0063] 对象音频渲染器接口

[0064] 如图3所示，基于对象的音频是通过对象音频渲染器接口306处理的，对象音频渲染器接口306包括或包装对基于对象的音频进行渲染的对象音频渲染器(OAR)。在实施例中，OARI 306从解码器302接收音频数据，并且对已经用合适的内联元数据发信号通知为是基于对象的音频的音频数据进行处理。OARI的工作通常是针对某些AVR产品和回放部件(诸如启用自适应音频的扬声器和条形音箱)对元数据更新进行过滤。它实现了以下的技术：诸如将元数据与传入的缓冲的采样合适地对齐；使系统适应变化的复杂性以满足处理器需要；对在块边界上未对齐的元数据更新进行智能过滤；以及针对诸如条形音箱和其他专门的扬声器产品之类的应用对元数据更新进行过滤。

[0065] 对象音频渲染器接口本质上是用于对象音频渲染器的进行两个操作的包装器：首先，它对演变框架和对象音频元数据比特流进行反序列化；第二，它在合适的时间、以合适的块大小缓冲将被OAR处理的输入采样和元数据更新。在实施例中，OARI实现异步输入/输出API(应用程序接口)，其中，采样和元数据更新被推到输入音频比特流上。在进行该输入调用之后，可用采样的数量被返回到调用器，然后这些采样被处理。

[0066] 对象音频元数据包含对自适应音频节目进行渲染所需要的所有相关信息，所述自适应音频节目具有来自解码器(例如，Dolby Digital Plus、Dolby TrueHD、Dolby MAT解码器或其他解码器)的相关联的一组基于对象的PCM音频输出。图6示出根据实施例将元数据组织为被对象音频渲染器处理的分层结构。如示图600中所示，对象音频元数据有效载荷被划分为节目分配以及相关联的对象音频元素。对象音频元素包括关于多个对象的数据，每个对象数据元素具有相关联的对象信息块，该对象信息块包含对象基本信息和对象渲染信息。对象音频元素还具有关于每个对象音频元素的元数据更新信息和块更新信息。

[0067] 输入音频比特流的PCM采样与定义这些采样怎样被渲染的某个元数据相关联。当对象和渲染参数改变时，针对新的或连续的PCM采样对元数据进行更新。关于元数据组帧，可以依赖于编解码器的类型不同地存储元数据更新。但是，通常，当编解码器特定的组帧被移除时，元数据更新应具有等同的定时和渲染信息，与它们的输送无关。图7示出根据实施例的第一类型的编解码器内的元数据更新的应用以及元数据更新的组帧。依赖于使用的数据编解码器，所有帧都包含元数据更新，该元数据更新可以将多个块包含在单个帧中，或者访问单元可以包含更新，通常每帧只有一个块。如示图700中所示，PCM采样702与周期性元数据更新704相关联。在该示图中，示出了五个这样的更新。在某些编解码器(诸如Dolby Digital Plus格式)中，一个或多个元数据更新可以存储在演变帧706中，演变帧706包含对

象音频元数据以及对于每个相关联的元数据更新的块更新。因而，图7的示例示出，前两个元数据更新存储在具有两个块更新的第一演变帧中，接下来的三个元数据更新存储在具有三个块更新的第二演变帧中。这些演变帧对应于统一的帧708和710，这些帧每个均具有限定数量的采样(例如，对于Dolby Digital Plus帧来说，长为1536个采样)。

[0068] 图7的实施例示出对于一种类型的编解码器(诸如Dolby Digital Plus编解码器)的元数据更新的存储。但是，可以使用其他编解码器和组帧方案。图8示出根据与不同编解码器(诸如Dolby TrueHD编解码器)一起使用的可替代组帧方案的元数据存储。如示图800中所示，每个元数据更新802均被封装到对应的演变帧804中，演变帧804具有对象音频元数据元素(OAMD)以及相关联的块更新。这些被组帧到某个数量的采样(例如，对于Dolby TrueHD编解码器来说，40个采样)的访问单元806中。虽然已经针对某些示例编解码器(诸如Dolby Digital Plus和Dolby TrueHD)描述了实施例，但是应该注意到，可以使用任何适合于基于对象的音频的编解码器，并且可以相应地对元数据组帧方案进行配置。

[0069] OARI操作

[0070] 对象音频渲染器接口负责将音频数据和演变元数据连接到对象音频渲染器。为了实现这一点，对象音频渲染器接口(OARI)将音频采样和附随元数据在可管理的数据部分或帧中提供给对象音频渲染器(OAR)。图7和图8示出元数据更新怎样被存储在进入OARI的音频中，图11、图12和图13示出用于OAR的音频采样和附随元数据。

[0071] 如图9的流程图900所示，对象音频渲染器接口操作包括多个离散的步骤或处理操作。图9的方法大体上示出了通过以下方式对基于对象的音频进行处理的过程，即，在对象音频渲染器接口(OARI)中接收音频采样块以及一个或多个相关联的对象音频元数据有效载荷；从每个对象音频元数据有效载荷反序列化一个或多个音频块更新；将音频采样和音频块更新存储在相应的音频采样存储器高速缓存和音频块更新存储器高速缓存中；并且基于音频块更新相对于处理块边界的定时和对齐以及一个或多个其他的参数来动态地选择音频采样的处理块大小，所述一个或多个其他的参数包括最大/最小处理块大小参数。在该方法中，基于对象的音频是在通过动态选择过程确定的大小的处理块中从OARI发送到OAR的。

[0072] 参考图9，对象音频渲染器接口首先接收音频采样块和经反序列化的演变元数据帧，902。音频采样块可以是任意大小，诸如高达在对象音频渲染器接口初始化期间传入的max_input_block_size参数。OAR可以被配置为支持有限数量的块大小，诸如长度为32、64、128、256、480、512、1024、1536和2048个采样的块大小，但不限于此，可以使用任何实用的块大小。

[0073] 元数据被作为具有二进制有效载荷(例如，数据类型evo_payload_t)和采样偏移的反序列化的演变框架帧传递，所述采样偏移指示演变框架帧应用于音频块中的哪个采样处。只有包含对象音频元数据的演变框架有效载荷被传递到对象音频渲染器接口。接下来，从对象音频元数据有效载荷反序列化音频块更新数据，904。块更新携带关于采样块的空间位置和其他元数据(诸如对象类型、增益和斜坡数据)。依赖于系统配置，多达例如八个块更新被存储在对象音频元数据结构中。除了各个块更新之外，偏移计算还合并了演变框架偏移、对象音频渲染器接口采样高速缓存的进展以及对象音频元数据的偏移值。然后将音频数据和块更新高速缓存，906。高速缓存操作在高速缓存中保持元数据和采样位置之间的关

系。如方框908中所示，对象音频渲染器接口为音频采样的处理块选择大小。然后为该处理块准备元数据，910。该步骤包括某些进程，诸如对象优先化、宽度移除、禁用对象处理、针对选定的块大小过滤过于频繁的更新、到对象音频渲染器支持的范围的空间位置裁剪(以确保没有负的Z值)以及将更新数据转换为供对象音频渲染器使用的特殊格式。然后用选定的处理块调用对象音频渲染器，912。

[0074] 在实施例中，对象音频渲染器接口步骤由API函数进行。一个函数(例如，oari_addsamples_evo)将对象音频元数据有效载荷解码为块更新，对采样和块更新进行高速缓存，并且选择第一处理块大小。第二函数(例如，第一oari_process)对一个块进行处理，并且选择下一个处理块大小。一个处理循环的示例调用序列如下：首先，调用oari_addsamples_evo，第二，零次或更多次调用oari_process，前提条件是处理块是可用的；对每个循环，重复这些步骤。

[0075] 如图9的步骤906中所示，OARI进行高速缓存和反序列化操作。图10更详细地示出根据实施例的对象音频渲染器接口的高速缓存和反序列化处理循环。如示图1000中所示，PCM采样的形式的对象音频数据被输入到PCM音频高速缓存1004，对应的元数据有效载荷通过对象音频元数据解析器1007被输入到更新高速缓存1008。块更新用编号的圆圈表示，每个与PCM音频高速缓存1004中的采样位置具有如箭头所示的固定关系。对于图10中所示的示例更新场景，最后两个更新与超过当前高速缓存的末尾的、与未来一个循环的音频相关联的采样有关。高速缓存过程涉及保留来自前一个处理循环的音频的任何未使用部分以及附随元数据。用于块更新的这个延缓高速缓存与更新高速缓存1008是分离的，因为对象音频元数据解析器总是将全套更新反序列化到主高速缓存1004中。音频高速缓存的大小受到在初始化时给定的输入参数的影响，诸如受max_input_block_size、max_output_block_size和max_objs参数的影响。元数据高速缓存大小是固定的，但是可以在对象音频渲染器接口实现内部改变OARI_MAX_EVO_MD参数，如果需要的话。

[0076] 为了为OARI_MAX_EVO_MD定义选择新的值，需要考虑选定的max_input_block_size参数。OARI_MAX_EVO_MD参数表示在对oari_addsamples_evo函数进行一次调用的情况下可以发送到对象音频渲染器接口的对象音频元数据有效载荷的数量。如果输入的采样块被多个对象音频元数据覆盖，则需要通过调用代码缩小输入大小以得到允许量的对象音频元数据。过量的音频和对象音频元数据通过在未来的处理循环中另外调用oari_addsamples_evo来进行处理。延缓更新被发送到音频高速缓存1004的延缓PCM部分1003。在某个实现中，对象音频元数据的数量在理论上的最糟情况是max_input_block_size/40，而更现实的最糟情况是max_input_block_size/128。可以在调用oari_addsamples_evo函数时处理变化的块大小的调用代码应该选择现实的最糟情况，而依赖于固定输入块大小的代码需要选择理论上的最糟情况。在这样的实现中，OARI_MAX_EVO_MD的默认值是16。

[0077] 对具有宽度(有时称为“大小”)的对象进行渲染通常需要比其他情况更多的处理能力。在实施例中，对象音频渲染器接口可以从一些或全部对象移除宽度。该特征由诸如max_width_objects参数之类的参数控制。从超过该计数的对象移除宽度。如果优先度信息在对象音频元数据中被指定或者由更高对象索引指定，则被选为移除宽度的对象的优先度更低。

[0078] 另外，对象音频渲染器接口对由对象音频渲染器中的限制器(limiter)引入的处

理延时进行补偿。这可以通过参数设置(诸如用b_compensate_latency参数)来启用或禁用。对象音频渲染器接口通过丢弃初始的无声并且在末尾处进行零清除(flushing)来进行补偿。

[0079] 如图9的步骤908所示,OARI进行处理块大小选择操作。处理块是具有零个或一个更新的采样块。在没有更新的情况下,对象音频渲染器继续对新的音频数据使用前一次更新的元数据。如上所述,对象音频渲染器可以被配置为支持有限数量的块大小:32、64、128、256、480、512、1,024、1,536和2,048个采样,但是其他大小也是可能的。通常,更大的处理块大小是更加CPU有效的。对象音频渲染器可以被配置为不支持元数据和处理块的开始之间的偏移。在这种情况下,块更新需要在处理块的开始或附近。通常,块更新被定位为在由最小输出块大小选择所允许的块的第一采样附近。处理块大小选择的目标是选择在块更新被定位在处理块的第一采样处的情况下尽可能大的处理块大小。该选择受到可用的对象音频渲染器块大小和块更新位置的约束。另外的约束来自于对象音频渲染器接口参数,诸如min_output_block_size和max_output_block_size参数。高速缓存大小和输入块大小不是处理块大小的选择中的因素。如果多于一次的更新在min_output_block_size个采样内发生,则只有第一次更新被保留,任何另外更新被丢弃。如果块更新不位于处理块的第一采样处,则元数据应用得太早,导致更新不精确。最大可能的不精确度由诸如min_output_block_size-1的参数值给出。没有任何块更新数据的初始采样导致无声的输出。如果对于多个采样尚未接收到更新数据,则输出也是静音的。直到错误情况被检测到为止的采样的数量由初始化时的参数max_lag_samples给出。

[0080] 图11示出根据实施例的对象音频渲染器接口对于元数据更新的应用。在该示例中,min_output_block_size被设置为128个采样,max_output_block_size被设置为512个采样。因而,如下四个可能的块大小可用于处理:128、256、480和512。图11示出选择发送到对象音频渲染器的采样的正确大小的过程。通常,确定合适的块大小是基于某些标准,这些标准是基于通过在给定的某些条件下调用可能的最大块大小来优化总体计算效率。对于第一条件,如果存在比最小块大小更靠近的两个更新,则应该在计算块大小确定之前移除第二更新。应该选择块大小使得:单个更新应用于将被处理的采样块,该更新尽可能地靠近将被处理的块中的第一采样;块大小需要不小于在初始化期间传递的min_output_block_size参数值;并且块大小需要不大于在初始化期间传递的max output_block_size参数值。

[0081] 图12示出根据实施例的由对象音频渲染器接口进行的初始处理循环的示例。如示图1200所示,元数据更新用从1到5的编号圆圈表示。处理循环从调用填充音频高速缓存和元数据高速缓存的oari_addsamples_evo函数1204开始,之后接着调用一系列oari_process渲染函数1206。因而,在调用函数1204之后,调用第一oari_process函数,该函数将第一音频块与更新0一起发送到对象音频渲染器。块和更新区域在图12中被示为阴影区域。随后,通过采样高速缓存的进展用每个函数调用1206示出。注意最大输出块大小怎样被实施,也就是说,每个阴影区域的大小不超过max_output_block_size 1202。对于所示的示例,更新2和更新3所具有的、与它们相关联的音频数据多于max_output_block_size参数所允许的音频数据,因而更新2和更新3被作为多个处理块发送。只有第一处理块具有更新元数据。处理块选择在下一轮等待附加采样以最大化处理块。随后调用oari_addsamples_evo函数,开始新的处理循环。在该图中可以看出,更新5应用于尚未被添加的音频。

[0082] 在随后的处理循环中，oari_addsamples_evo函数首先将所有的剩余的音频移动到高速缓存的开始，并且调整剩余的更新的偏移。图13示出在图12的示例处理循环之后的第二处理循环。然后oari_addsamples_evo函数将新的音频和元数据添加到高速缓存中的延缓内容之后。更新1的处理示出min_output_block_size参数的实施。更新0的第二处理块小于该参数，因而被扩展以与该最小大小匹配。结果，处理块现在包含更新1，更新1需要沿着该音频块处理。因为更新1未被定位在处理块的第一采样处，但是对象音频渲染器在那应用更新1，所以元数据被早地应用。这导致音频渲染的精度降低。

[0083] 关于元数据定时，实施例包括当将元数据应用于对象音频渲染器接口中的对象音频渲染器时保持准确定时的机制。一个这样的机制包括在内部数据结构中使用采样偏移字段。图14示出列出根据实施例的在计算内部oari_md_update数据结构中的偏移字段时使用的字段的表格(表1)。

[0084] 对于更高的采样速率，需要缩放所指示的采样偏移中的一些。以下位字段的时间尺度(time scale)是基于音频采样速率：

[0085] Timestamp

[0086] oa_sample_offset

[0087] block_offset_factor

[0088] oa_sample_offset位字段由oa_sample_offset_type、oa_sample_offset_code和oa_sample_offset字段的组合给出。需要按照如以下表2列出的依赖于音频采样频率的缩放因子来对这些位字段的值进行缩放。

[0089]

相关联的音频采样频率（kHz）	时间尺度基础 (kHz)	缩放因子
48	48	1
96	48	2
192	48	4
44.1	44.1	1
88.2	44.1	2
176.4	44.1	4

[0090] 表2

[0091] 例如，如果96kHz比特流演变框架有效载荷具有2,000个采样的有效载荷偏移，则需要按照缩放因子2对该值进行缩放，并且演变框架有效载荷中的时间戳需要指示1,000个采样。因为对象音频元数据有效载荷不知道音频采样速率，所以它采取48kHz的时间尺度基础，该时间尺度基础的缩放因子为1。重要的是注意到，在对象音频元数据内，斜坡持续时间值(由ramp_duration_code、use_ramp_table、ramp_duration_table和ramp_duration字段的组合给出)还使用48kHz的时间尺度基础。需要根据相关联的音频的采样频率来对ramp_duration值进行缩放。

[0092] 一旦缩放操作被进行，就可以进行最后的采样偏移计算。在实施例中，用于偏移值的整体计算的方程由以下程序例程给出。

```
/* N represents the number of metadata blocks in the object audio metadata payload and must be
in the range [1, 8] */

for (i=0; i<N; i++) {
    metadata_update_buffer[i].offset = sample_offset + (timestamp * fs_scale_factor) +
      (oa_sample_offset * fs_scale_factor) + (32 * block_offset_factor[i] * fs_scale_factor);

}
```

[0093]

[0094] 对象音频渲染器接口基于元数据更新的定时和对齐以及最大/最小块大小定义和其他可能的因素来动态地调整音频的处理块大小。这允许对于意图应用元数据的音频块最优地发生元数据更新。元数据因而可以以适应多个对象以及相对于数据块边界不均匀地更新的对象的方式、并且以允许系统处理器相对于处理器循环高效地运作的方式与音频本体配对。

[0095] 虽然已经关于一个或多个特定编解码器(诸如Dolby Digital Plus、MAT 2.0和TrueHD)中的实现描述和示出了实施例,但是应该注意到可以使用任何编解码器或解码器格式。

[0096] 本文中所描述的音频环境的各方面表示音频或音频/视觉内容通过合适的扬声器和回放设备的回放,并且可以表示在其中收听者正在体验所捕获的内容的回放的任何环境,诸如影院、音乐厅、室外剧院、家庭或房间、收听亭、汽车、游戏控制台、耳机或耳麦系统、公共地址(PA)系统或任何其他回放环境。虽然已经主要关于家庭剧院环境中的、空间音频内容与电视内容相关联的示例和实现描述了实施例,但是应该注意到实施例还可以在其他的基于消费者的系统(诸如游戏、筛选系统以及任何其他基于监视器的A/V系统)中实现。包括基于对象的音频和基于声道的音频的空间音频内容可以与任何相关的内容(相关联的音频、视频、图形等)结合使用,或者它可以包括独立的音频内容。回放环境可以是从耳机或近场监视器到小房间或大房间、汽车、露天竞技场、音乐厅等的任何合适的收听环境。

[0097] 本文中所描述的系统的各方面可以在合适的基于计算机的声音处理网络环境中实现,用于对数字或数字化音频文件进行处理。自适应音频系统的部分可以包括包含任何期望数量的单独机器的一个或多个网络,所述网络包括用于缓冲和路由在计算机之间传输的数据的一个或多个路由器(未示出)。这样的网络可以构建在各种不同的网络协议之上,并且可以是互联网、广域网(WAN)、局域网(LAN)或它们的任何组合。在网络包括互联网的实施例中,一个或多个机器可以被配置为通过web浏览器程序访问互联网。

[0098] 部件、块、过程或其他功能部件中的一个或多个可以通过控制系统的基于处理器的计算设备的执行的计算机程序来实现。还应该注意到,本文中所公开的各种功能可以使用硬件、固件的任何数量的组合来描述,和/或被描述为实现在各种机器可读或计算机可读介质中的、就它们的行为、寄存器传送、逻辑部件和/或其他特性而言的数据和/或指令。其中可以实现这样的格式化数据和/或指令的计算机可读介质包括但不限于各种形式的物理(非暂时性)非易失性存储介质,诸如光学、磁性或半导体存储介质。

[0099] 除非上下文另有明确要求,否则在整个说明书和权利要求书中,词语“包括”“包含”等要从与排他性或穷举性意义相反的包容性的意义上来解释;也就是说,从“包括但不

限于”的意义上来解释。使用单数或复数的词语分别还包括复数或单数。另外，词语“本文中”、“下文中”、“以上”、“下面”以及类似含义的词语是将本申请作为一个整体来引用，而不是引用本申请的任何特定部分。当词语“或”用于引用两个或更多个项的列表时，该词语涵盖该词语的以下所有解释：该列表中的任何一个项、该列表中的所有的项以及该列表中的项的任何组合。

[0100] 整个本说明书中所引用的“一个实施例”、“一些实施例”或“实施例”意味着与实施例结合描述的特定的特征、结构或特性被包括在所公开的(一个或多个)系统和(一种或多种)方法的至少一个实施例中。因而，短语“在一个实施例中”、“在一些实施例中”或“在实施例中”在整个本描述中各个地方的出现可以指示相同的实施例，或者可能不一定指示相同的实施例。此外，所述特定的特征、结构或特性可以以本领域的普通技术人员将清楚的任何合适的方式组合。

[0101] 虽然已经以示例的方式就特定实施例描述了一个或多个实现，但是应该理解，一个或多个实现不限于所公开的实施例。相反，意图涵盖本领域技术人员将清楚的各种修改和类似布置。因此，所附权利要求的范围应该被给予最广泛的解释，以便包含所有这样的修改和类似布置。

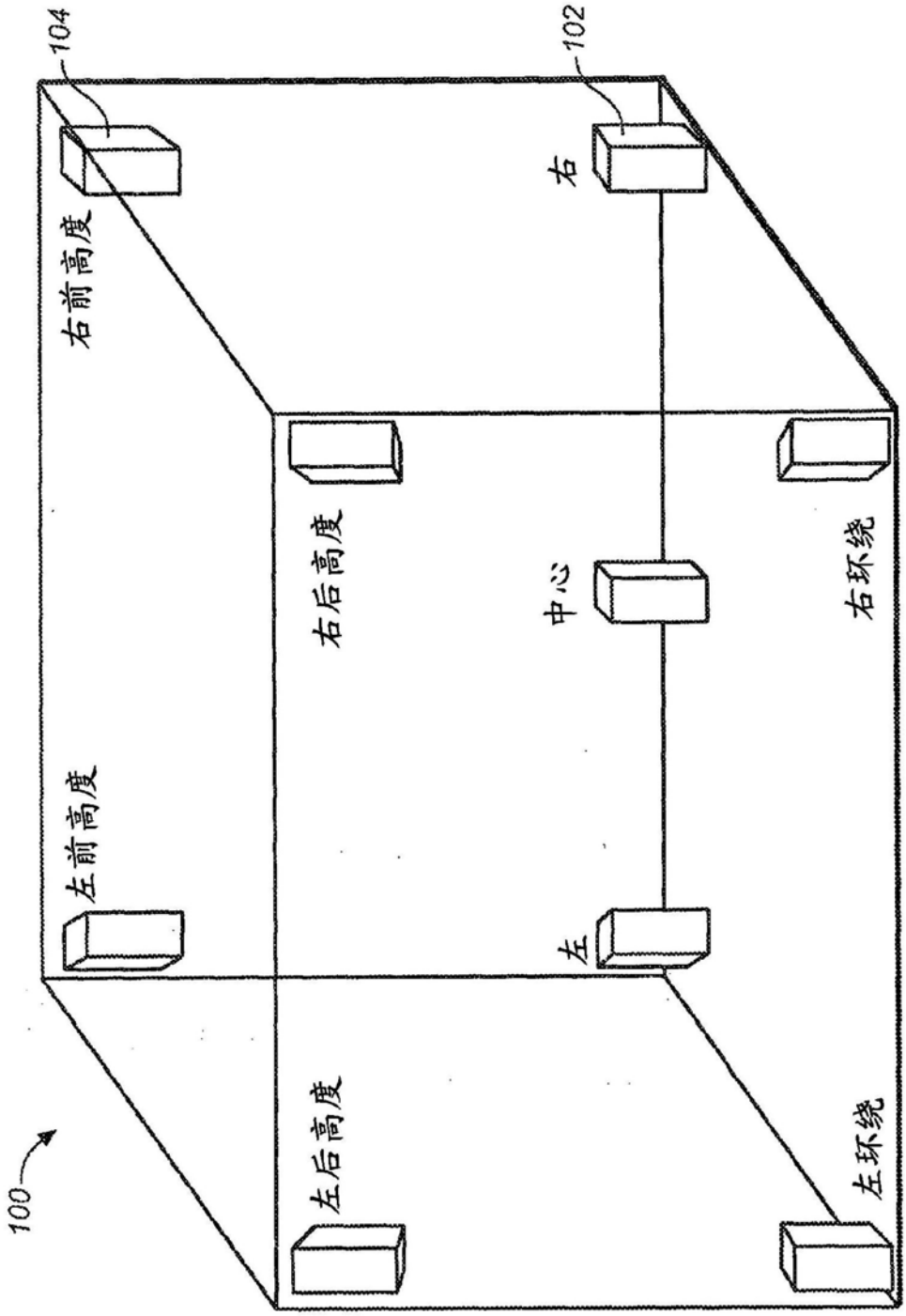


图1

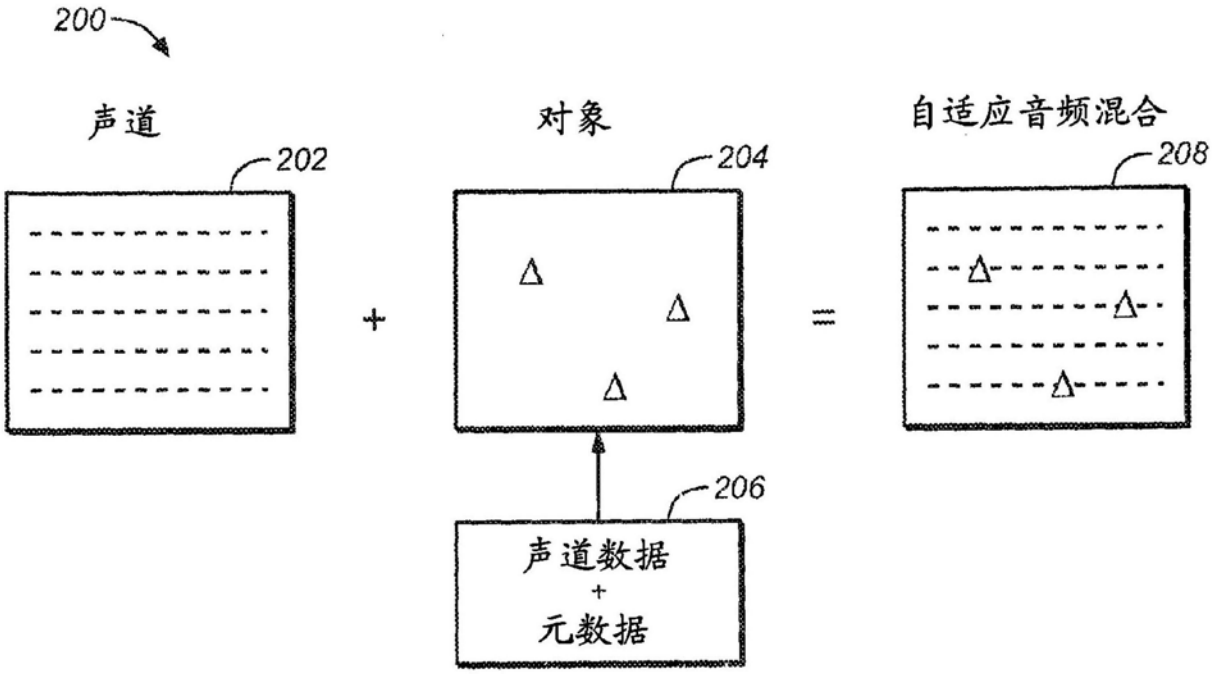

200
声道
202
对象
204
自适应音频混合
208
206
声道数据
+
元数据

图2

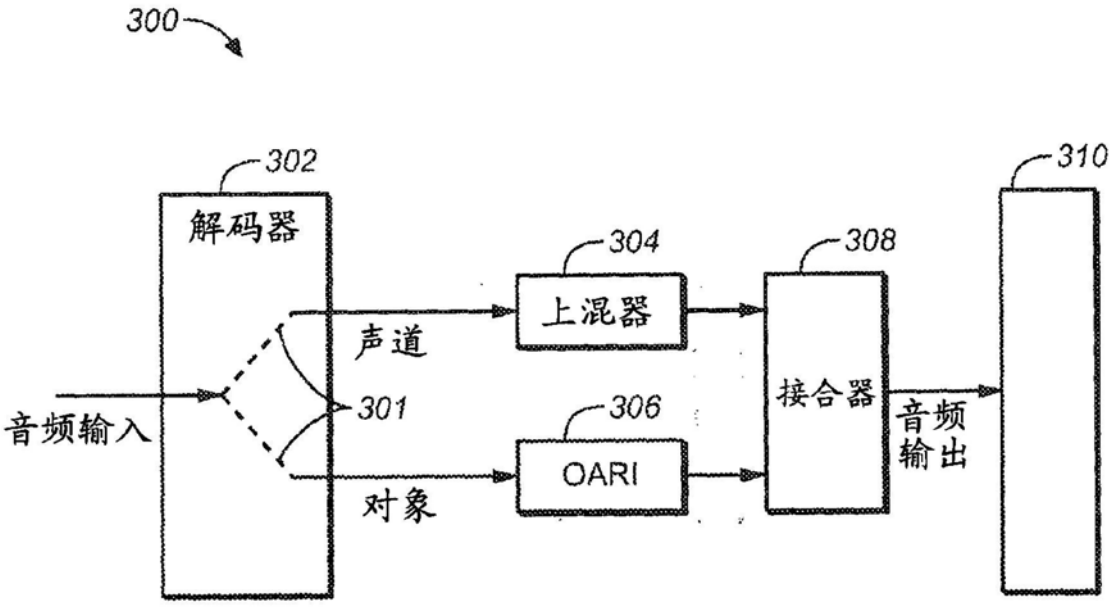

300
302
解码器
304
上混器
308
310
声道
音频输入
301
306
OARI
接合器
音频
输出
对象

图3

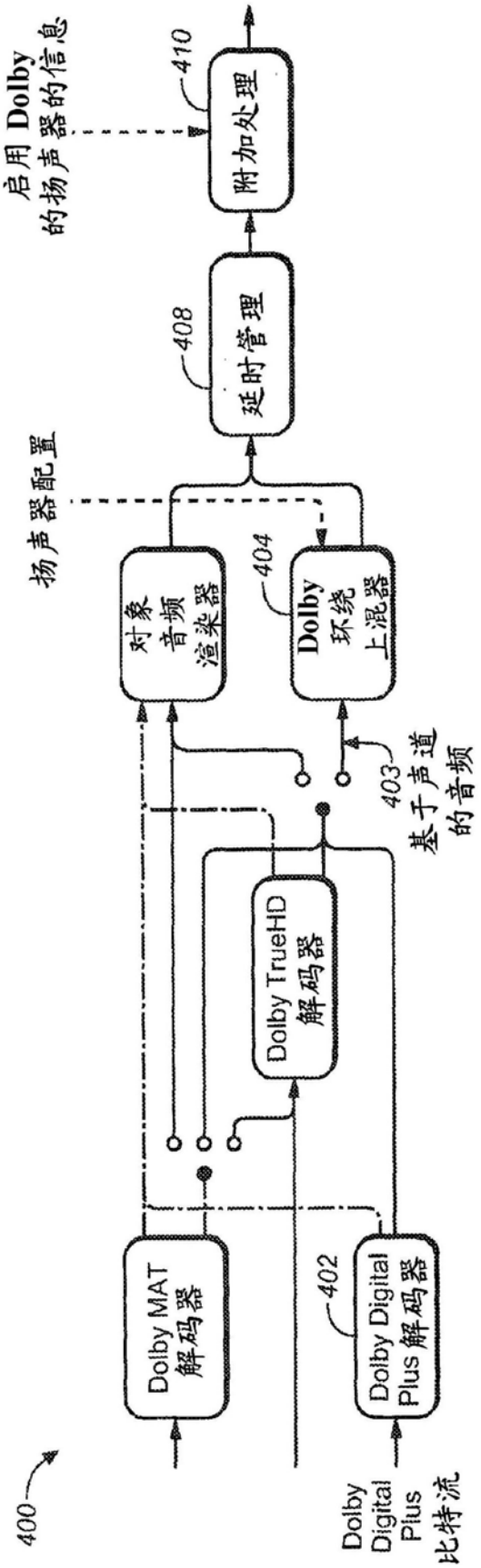

400
Dolby Digital Plus 比特流
Dolby MAT 解码器
402
Dolby Digital Plus 解码器
Dolby TrueHD 解码器
403 基于声道的音频
404
Dolby 环绕上混器
对象音频渲染器
扬声器配置
408
延时管理
410
附加处理
启用 Dolby 的扬声器的信息

图4A

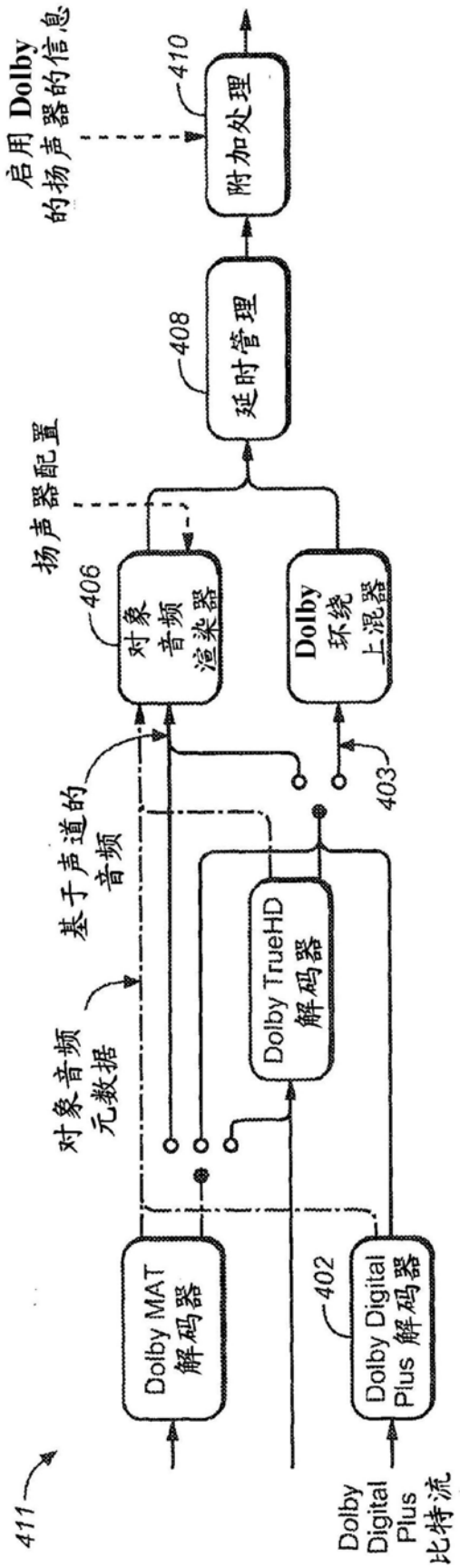

启用 Dolby 的扬声器的信息
410
附加处理
408
延时管理
扬声器配置
406
对象音频渲染器
Dolby 环绕上混器
403
基于声道的音频
对象音频元数据
Dolby TrueHD 解码器
Dolby MAT 解码器
402
Dolby Digital Plus 解码器
411
Dolby Digital Plus 比特流

图4B

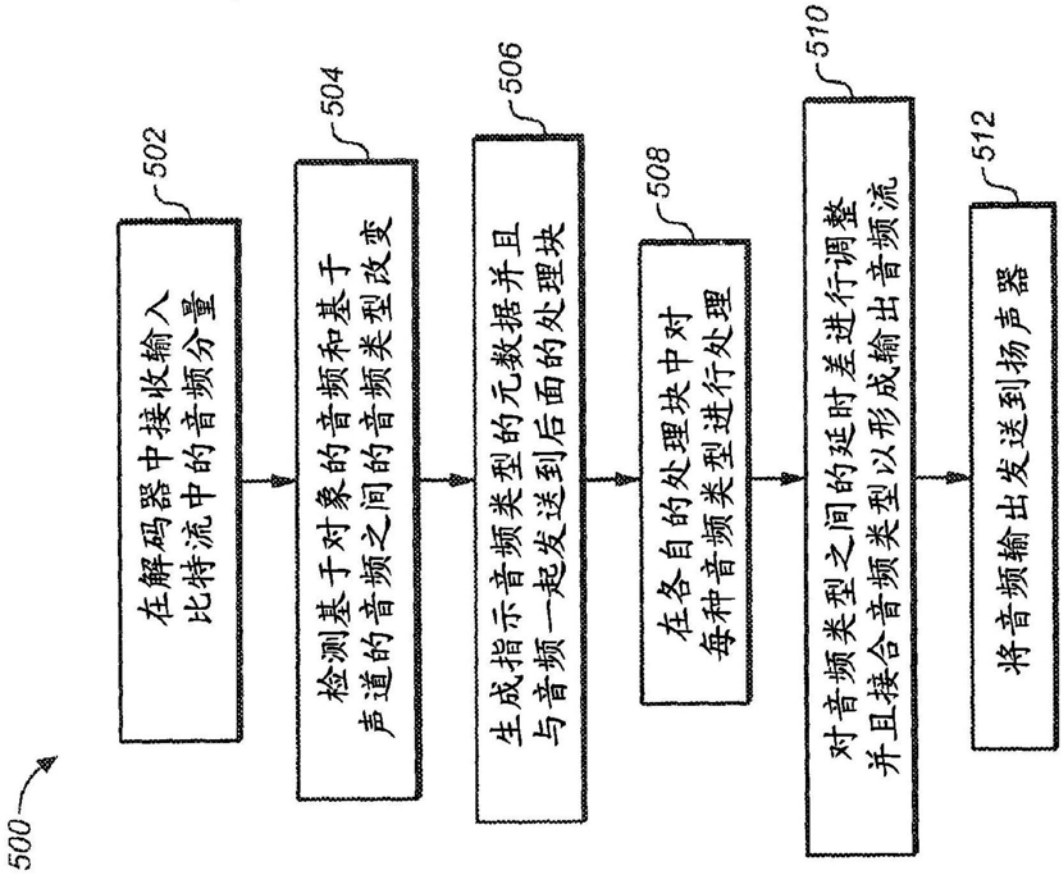

500
502
在解码器中接收输入比特流中的音频分量
504
检测基于对象的音频和基于声道的音频之间的音频类型改变
506
生成指示音频类型的元数据并且与音频一起发送到后面的处理块
508
在各自的处理块中对每种音频类型进行处理
510
对音频类型之间的延时差进行调整并且接合音频类型以形成输出音频流
512
将音频输出发送到扬声器

图5

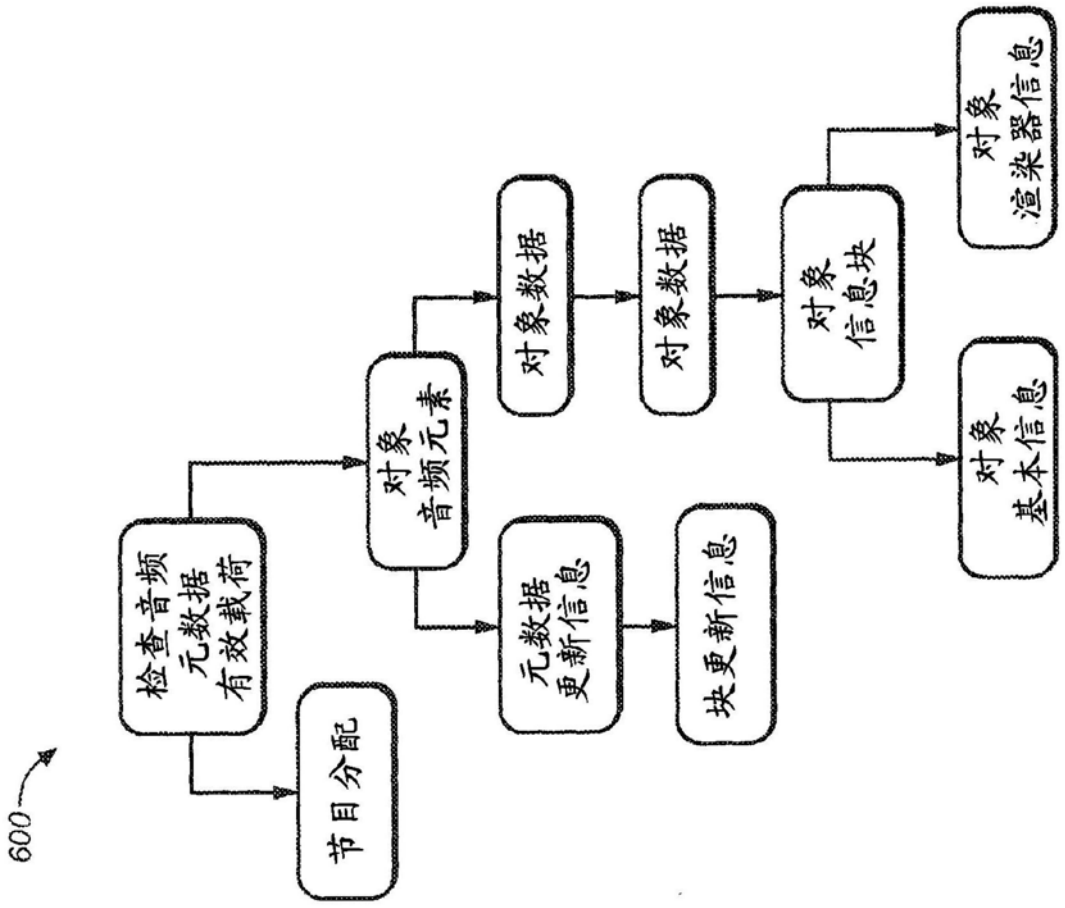

600
检查音频元数据有效载荷
节目分配
对象音频元素
元数据更新信息
块更新信息
对象数据
对象数据
对象信息块
对象基本信息
对象渲染器信息

图6

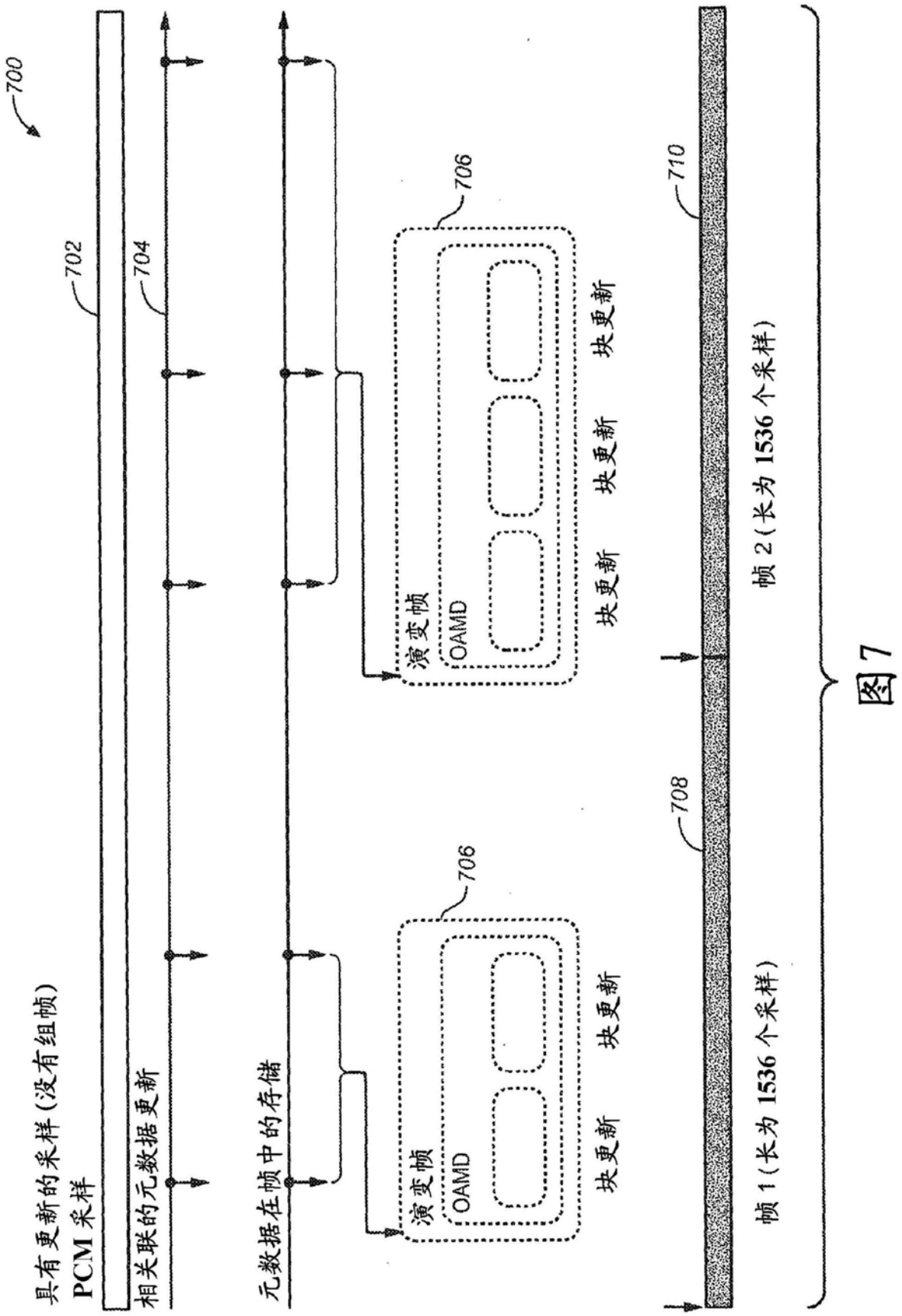

700
具有更新的采样(没有组帧)
PCM 采样
702
相关联的元数据更新
704
元数据在帧中的存储
706
演变帧
OAMD
块更新
块更新
块更新
706
演变帧
OAMD
块更新
708
710
帧 1(长为 1536 个采样)
帧 2(长为 1536 个采样)

图7

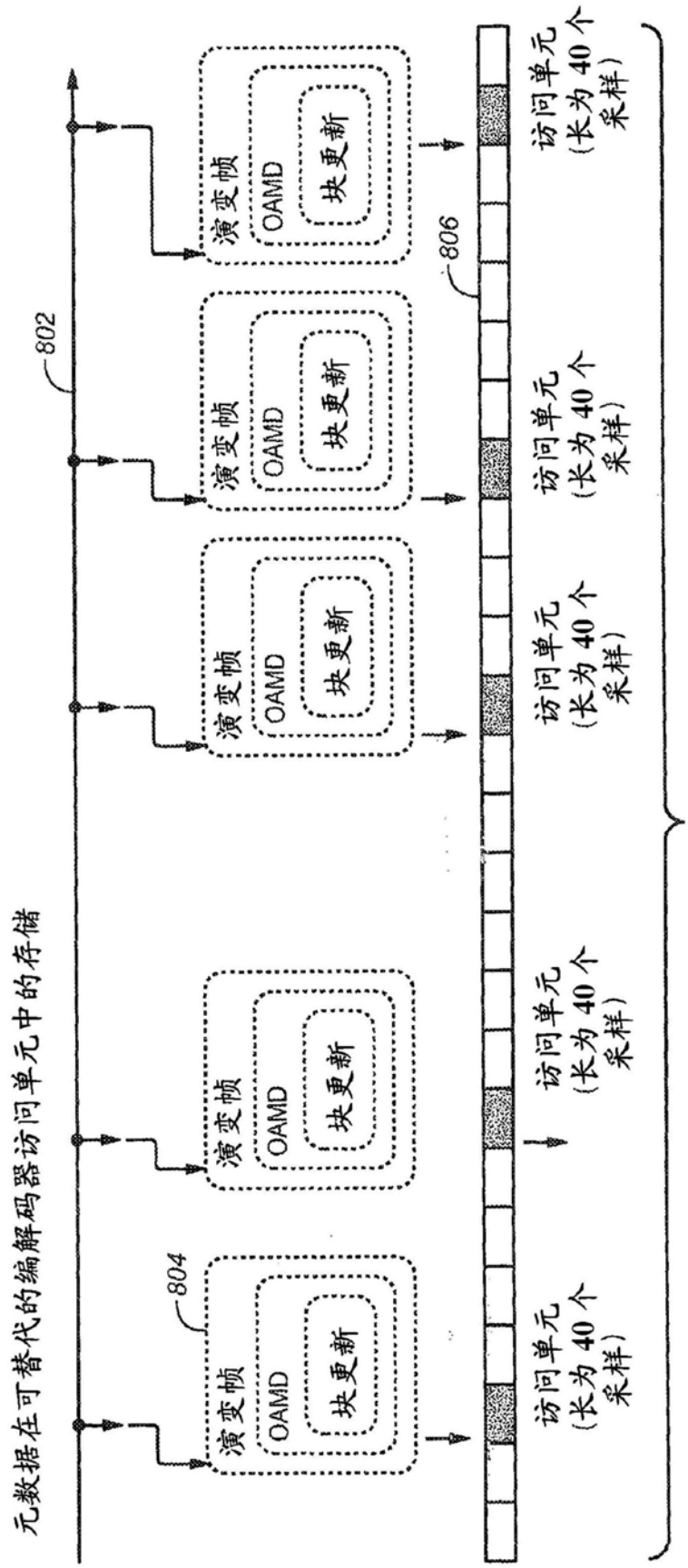

800
802
804
806
元数据在可替代的编解码器访问单元中的存储
演变帧
OAMD
块更新
访问单元
(长为 40 个
采样)

图8

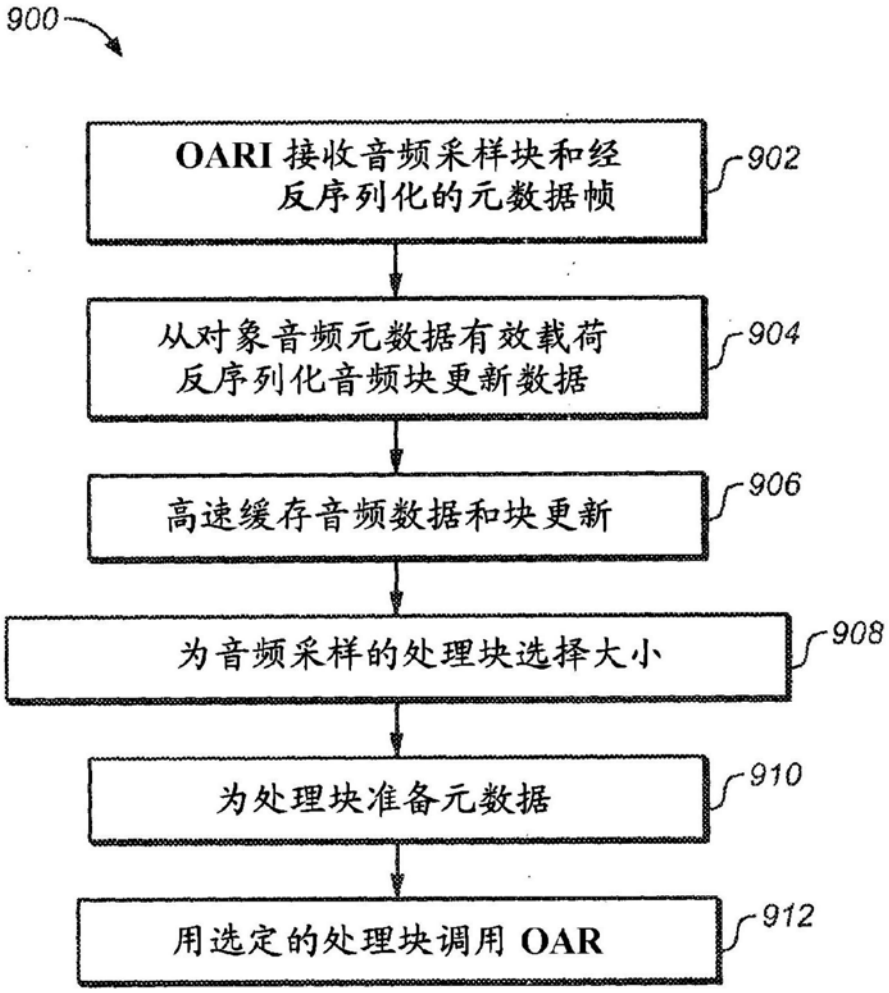

900
OARI 接收音频采样块和经反序列化的元数据帧
902
从对象音频元数据有效载荷反序列化音频块更新数据
904
高速缓存音频数据和块更新
906
为音频采样的处理块选择大小
908
为处理块准备元数据
910
用选定的处理块调用 OAR
912

图9

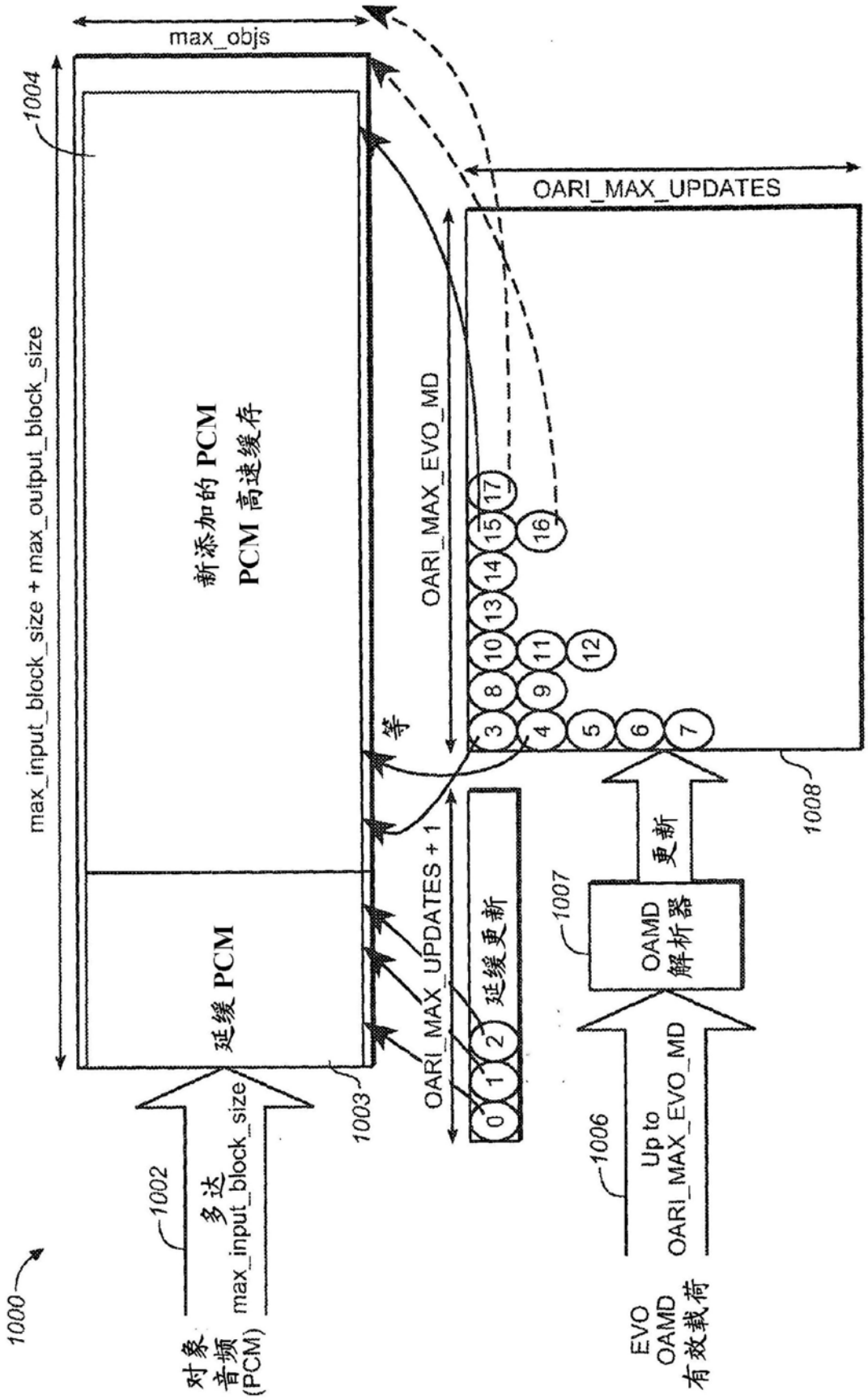

max_objs
1004
max_input_block_size + max_output_block_size
新添加的 PCM
PCM 高速缓存
OARI_MAX_EVO_MD
OARI_MAX_UPDATES
等
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
OARI_MAX_UPDATES + 1
延缓 PCM
延缓更新
1003
1007
OAMD 解析器
更新
1008
1002
多达 max_input_block_size
对象音频 (PCM)
1006
Up to OARI_MAX_EVO_MD
EVO OAMD 有效载荷
1000

图10

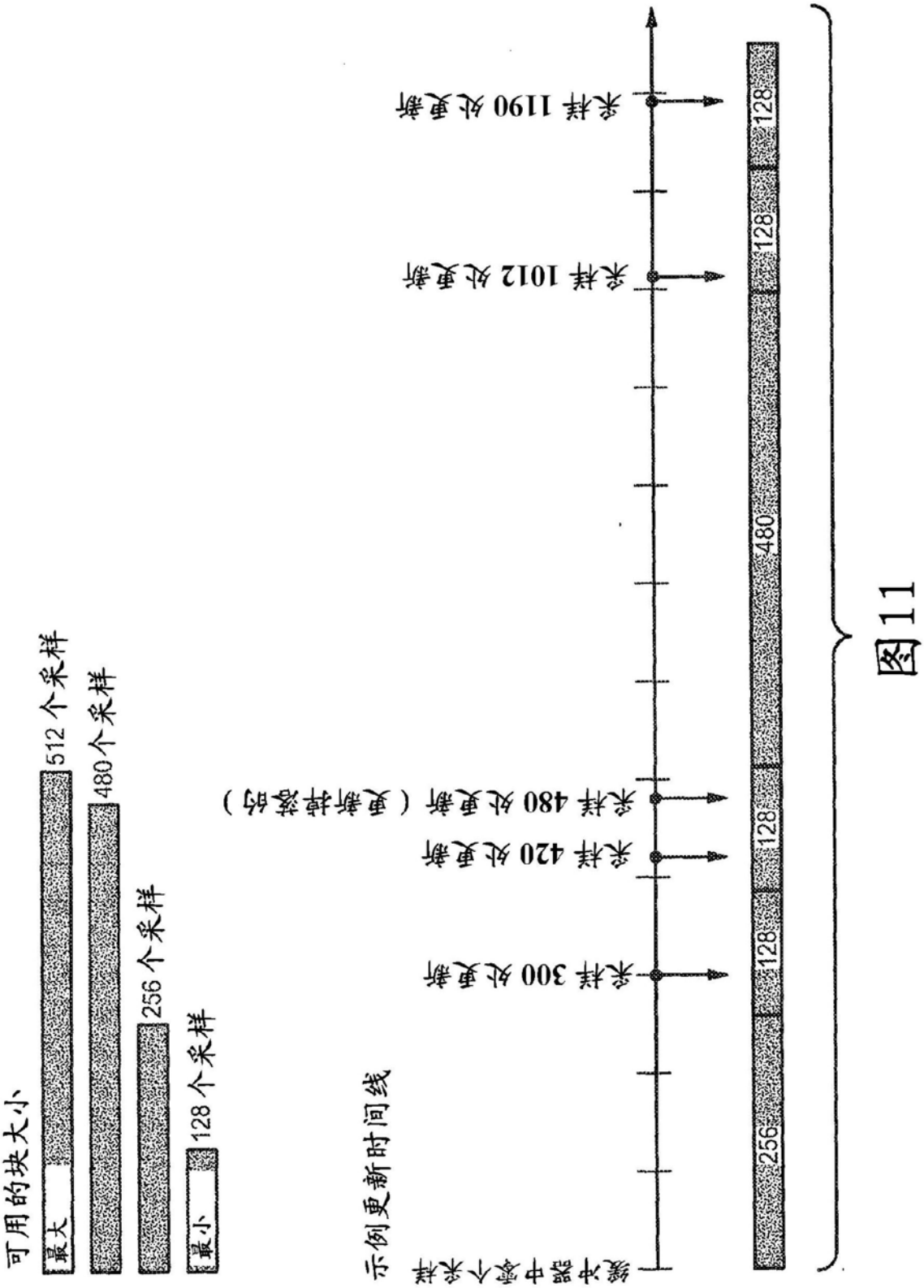

可用的块大小
最大
512 个采样
480 个采样
256 个采样
最小
128 个采样
示例更新时间线
缓冲器中第一个采样
采样 300 处更新
采样 420 处更新
采样 480 处更新（更新掉落的）
采样 1012 处更新
采样 1190 处更新
256
128
128
128
480
128

图11

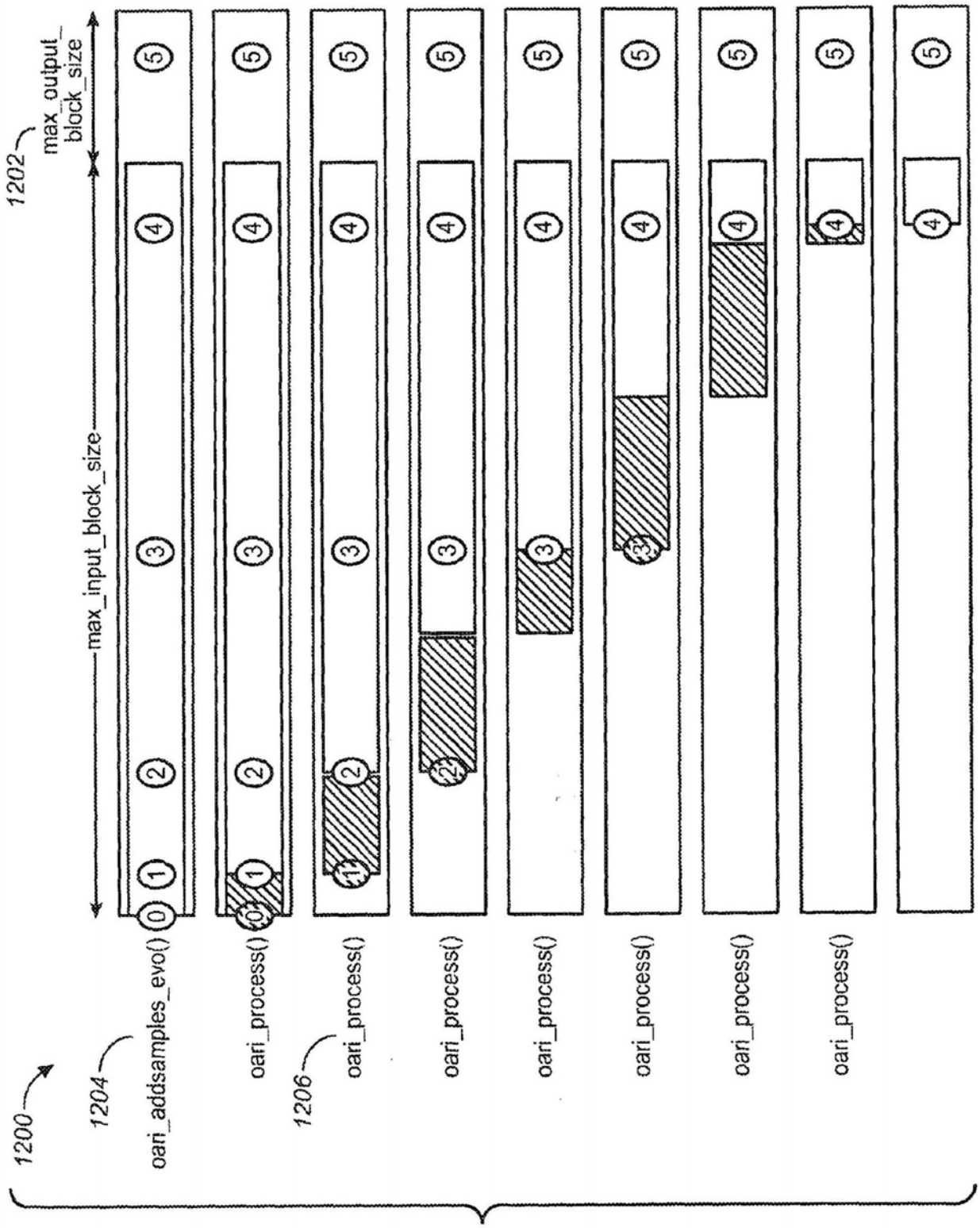

1200
1202
max_output_block_size
max_input_block_size
1204
oari_addsamples_evo()
1206
oari_process()
oari_process()
oari_process()
oari_process()
oari_process()
oari_process()
oari_process()
oari_process()
0
1
2
3
4
5

图12

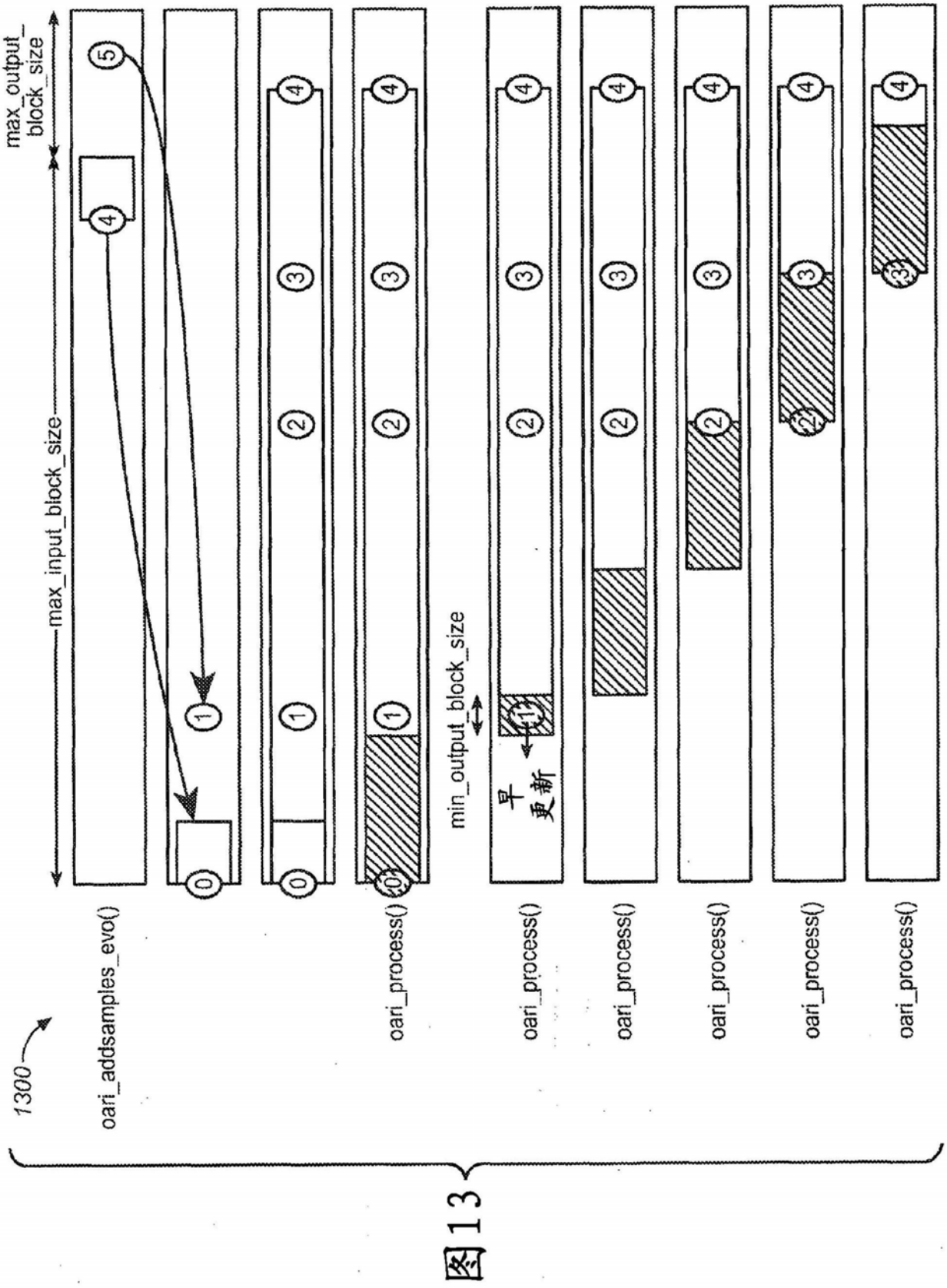

max_output_block_size
max_input_block_size
min_output_block_size
早更新
1300
oari_addsamples_evo()
oari_process()
oari_process()
oari_process()
oari_process()
oari_process()
oari_process()

图13

表 1

字段	编码	比特流中的有效范围	描述
sample_offset	无符号整型	[0, max_input_block_size-1]	该字段是对象音频渲染器接口的输入参数，该输入参数指示相对于应用输入演变框架有效载荷的输入缓冲器中的第一采样的偏移
timestamp	variable_bit(11)	[0, frame_duration-1]	该字段包含采样中的偏移，该偏移是相对于所讨论的有效载荷所属演变帧的开头的。
oa_sample_offset, oa_sample_offset_code, oa_sample_offset_type	u(5), v(2), v(2)	[0, 31]	对象音频元数据采样偏移值指示编解码器帧的开头和其中元数据块偏移值生效的采样之间的采样的数量。 oa_sample_offset 字段的值由 oa_sample_offset、oa_sample_offset_code 和 oa_sample_offset 字段的组合给出。
block_offset_factor	u(6)	[0, 63]	block_offset_factor 字段描述远离元数据块应用的 oa_sampe_offset 值的 32 采样块的数量

图14