(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 104641647 B
(45)授权公告日 2018.05.04

(21)申请号 201380031308.6
(22)申请日 2013.04.15
(65)同一申请的已公布的文献号
申请公布号 CN 104641647 A
(43)申请公布日 2015.05.20
(30)优先权数据
61/624,098 2012.04.13 US
61/666,185 2012.06.29 US
(85)PCT国际申请进入国家阶段日
2014.12.12
(86)PCT国际申请的申请数据
PCT/EP2013/057798 2013.04.15
(87)PCT国际申请的公布数据
WO2013/153226 EN 2013.10.17
(73)专利权人 GE视频压缩有限责任公司
地址 美国纽约
(72)发明人 托马斯·席尔 瓦莱里·乔治
阿纳斯塔西娅·亨克尔
德特勒夫·马佩
卡斯滕·格吕内贝格
罗伯特·斯库平
(74)专利代理机构 北京康信知识产权代理有限责任公司 11240
代理人 余刚 吴孟秋
(51)Int.Cl.
H04N 19/70(2014.01)
H04N 19/13(2014.01)
H04N 19/463(2014.01)
H04N 19/174(2014.01)
H04N 19/436(2014.01)
(56)对比文件
CN 101553988 A,2009.10.07,
CN 101842988 A,2010.09.22,
WO 03043345 A1,2003.05.22,
JP 2006180521 A,2006.07.06,
MISRA K.《Entropy Slices for Parallel Entropy Coding》.《MPEG.94 MEETING,MOTION PICTURE EXPERT GROUP OR ISO/IEC JTC1/SC29/WG11》.2010,
审查员 李颖

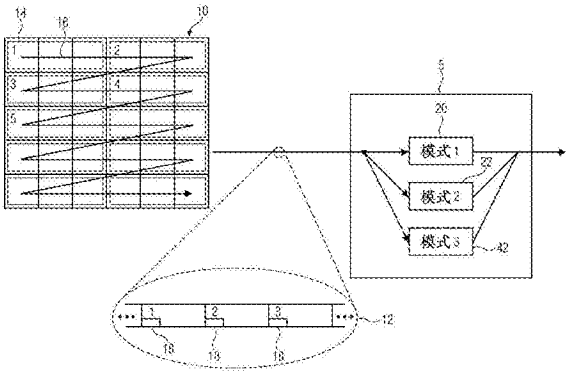
权利要求书4页 说明书29页 附图18页

(54)发明名称

低延迟图像写码

(57)摘要

诸如波前并行处理的并行处理概念是通过放弃普通片概念而实现具有减少的端至端延迟，据此，完全独立于在该相应的片的图像外部的区域或者有关的至少独立于在该相应的片外部的远至熵写码的区域而写码/译码片，例如，即有利于不同模式的片，即允许跨越片边界的相互依赖性的一个被称为从属片的片以及不允许跨越片边界的相互依赖性的被称为正常片的其它片。与该方面相组合或不组合，通过使用片的开始语法部分定位WPP进入点来较高效地进行WPP处理概念。



CN 104641647 B

1.一种用于从数据流(12)重构图像(10)的译码器,所述图像以所述图像(10)被分割成的片(14)为单位而被写码至所述数据流中,其中,所述译码器被配置为根据片次序(16)对来自所述数据流(12)的所述片(14)进行译码,并且所述译码器响应于所述片中的当前片内的语法元素部分(18)来根据至少两种模式(20,22)中的一个对所述当前片进行译码,并且

根据所述至少两种模式中的第一模式(20),使用上下文适应熵译码(24)对来自所述数据流(12)的所述当前片进行译码,所述上下文适应熵译码包括越过所述片边界的上下文的导出、所述上下文的符号概率的连续更新以及取决于先前译码的片的符号概率的保存状态的符号概率的初始化(38,40),并且

根据所述至少两种模式中的第二模式(22),使用所述上下文适应熵译码对来自所述数据流(12)的所述当前片进行译码,所述上下文适应熵译码具有将所述上下文的所述导出限制为不越过所述片边界、所述上下文的符号概率的连续更新以及独立于任意先前译码的片的所述符号概率的初始化。

2.根据权利要求1所述的译码器,其中,在写码区块(32)中分割所述图像(10),所述写码区块以列和行布置并且具有在彼此之间界定的光栅扫描次序(36),并且所述译码器被配置为使每个片(14)与所述光栅扫描次序(36)中的所述写码区块(32)的连续子集相关联,使得所述子集根据所述片次序而依所述光栅扫描次序(36)彼此相随。

3.根据权利要求2所述的译码器,其中,所述译码器被配置为保存如在根据所述光栅扫描次序(36)在上下文适应熵译码所述先前译码的片直至列中的第二写码区块时并且在根据第一模式初始化用于所述当前片的所述上下文适应熵译码的所述符号概率时所获得的符号概率,根据所述光栅扫描次序检查与所述当前片相关联的写码区块(32)的所述连续子集的第一写码区块是否是列中的第一写码区块,并且如果是,则依据如在根据所述光栅扫描次序(36)在上下文适应熵译码所述先前译码的片直至列中的第二写码区块时获得的保存的符号概率来初始化(40)用于所述当前片的所述上下文适应熵译码的所述符号概率,并且如果不是,则依据如在上下文适应熵译码所述先前译码的片直至所述先前译码的片的结尾时所获得的符号概率来初始化(38)用于所述当前片的所述上下文适应熵译码的所述符号概率。

4.根据权利要求1所述的译码器,其中,所述译码器被配置为响应于所述片(14)中的所述当前片内的所述语法元素部分(18)以根据至少三种模式中的一个,即以所述第一模式(20)和第三模式(42)中的一个或所述第二模式(22)来对所述当前片进行译码,其中,

所述译码器被配置为:

根据所述第一模式(20),使用越过所述片边界的预测性译码对所述当前片进行译码;

根据所述第二模式(22),使用将所述预测性译码限制为不越过所述片边界的预测性译码来对所述当前片进行译码;以及

根据所述第三模式(42),使用所述上下文适应熵译码和越过所述片边界的预测性译码对来自所述数据流的所述当前片进行译码,

所述上下文适应熵译码具有限制所述上下文的所述导出以不越过所述片边界、所述上下文的符号概率的连续更新以及独立于任意先前译码的片的所述符号概率的初始化,

其中,根据语法元素来选择所述第一模式和所述第三模式中的一个。

5.根据权利要求1所述的译码器,其中,所述译码器被配置为响应于所述数据流中的一

般语法元素以在至少两个操作模式中的一个中进行操作，并且根据第一操作模式来执行对每个片的所述语法元素部分的响应性，并且根据第二操作模式，必然地使用所述至少两种模式中的不同于所述第一模式的不同模式。

6.根据权利要求2所述的译码器，其中，所述译码器被配置为根据所述第一模式和所述第二模式，必然且不间断地继续从所述当前片的开头至结尾连续地更新所述符号概率。

7.根据权利要求2所述的译码器，其中，所述译码器被配置为保存如在所述上下文适应熵译码所述先前译码的片直至所述先前译码的片的结尾时以及在根据所述第一模式初始化用于所述当前片的所述上下文适应熵译码的所述符号概率时所获得的符号概率，根据保存的符号概率初始化用于所述当前片的所述上下文适应熵译码的所述符号概率。

8.根据权利要求4所述的译码器，其中，所述译码器被配置为在所述第一模式和所述第二模式下将所述预测性译码限制于所述图像被再分为的瓦片内。

9.根据权利要求1所述的译码器，其中，所述译码器被配置为在所述第一模式和所述第二模式下，从所述当前片读取揭示所述当前片再分为并行子部分的信息，在第一并行子部分的结尾处停止所述上下文适应熵译码并且在任一后续并行子部分的开头处重新恢复所述上下文适应熵译码包括：在所述第一模式下，依据在前并行子部分的符号概率的所保存的状态对所述符号概率进行初始化；以及在所述第二模式下，独立于任何先前译码的片和任何先前译码的并行子部分对所述符号概率进行初始化。

10.根据权利要求1所述的译码器，其中，所述译码器被配置为根据所述至少两种模式中的所述第一模式(20)，为所述当前片从在所述第二模式下译码的在前片拷贝片报头语法的一部分。

11.根据权利要求1所述的译码器，其中，所述译码器被配置为使用波前并行处理来处理从所述数据流(12)重构所述图像(10)，其中，每个片(14)包括开始语法部分(400)，所述开始语法部分指示相应的所述片的译码开头在所述图像(10)中的位置，并且其中，所述译码器被配置为：

通过使用所述片的开始语法部分识别在所述图像的左手侧处开始的片来识别所述片被分组成的波前并行处理子流的进入点，以及通过根据所述片次序顺序开始所述波前并行处理子流的所述译码来并行地译码所述波前并行处理子流。

12.一种用于将图像(10)以所述图像(10)被分割成的片(14)为单位编码至数据流(12)中的编码器，其中，所述编码器被配置为根据片次序(16)将所述片(14)编码至所述数据流(12)中，并且所述编码器被配置为：

判定用于所述片的当前片的语法元素部分(18)并且将所述语法元素部分写码至所述当前片中，使得用信号将所述语法元素部分发送给所述当前片以根据至少两种模式(20，22)中的一个进行写码，以及

如果根据所述至少两种模式中的第一模式(20)来写码所述当前片，则使用上下文适应熵编码(24)将所述当前片编码至所述数据流(12)中，所述上下文适应熵编码包括越过片边界的上下文的导出、所述上下文的符号概率的连续更新以及根据先前译码的片的符号概率所保存的状态的所述符号概率的初始化(38，40)，并且

如果根据所述至少两种模式中的第二模式(22)来写码所述当前片，则使用所述上下文适应熵编码将所述当前片编码至所述数据流(12)中，所述上下文适应熵编码具有将所述上

下文的所述导出限制为不越过所述片边界、所述上下文的符号概率的连续更新以及独立于任何先前编码的片的所述符号概率的初始化。

13.根据权利要求12所述的编码器，其中，在写码区块(32)中分割所述图像(10)，所述写码区块以列和行布置并且具有在彼此之间界定的光栅扫描次序(36)，并且所述编码器被配置为使每个片(14)与所述光栅扫描次序(36)中的所述写码区块(32)的连续子集相关联，使得所述子集根据所述片次序(36)而依所述光栅扫描次序彼此相随。

14.根据权利要求13所述的编码器，其中，所述编码器被配置为保存如在根据所述光栅扫描次序(36)在上下文适应熵编码所述先前编码的片直至列中的第二写码区块时并且在根据第一模式初始化用于所述当前片的所述上下文适应熵编码的所述符号概率时所获得的符号概率，检查与所述当前片相关联的写码区块(32)的所述连续子集的第一写码区块是否是根据所述光栅扫描次序的列中的第一写码区块，并且如果是，则依据如在根据所述光栅扫描次序(36)在上下文适应熵编码所述先前编码的片直至列中的第二写码区块时所获得的所保存符号概率初始化(40)用于所述当前片的所述上下文适应熵编码的所述符号概率，并且如果不是，则依据如在所述上下文适应熵编码所述先前编码的片直至所述先前编码的片的结尾时所获得的符号概率来初始化(38)用于所述当前片的所述上下文适应熵编码的所述符号概率。

15.根据权利要求12所述的编码器，其中，所述编码器被配置为根据至少三种模式中的一个，即以所述第一模式(20)和第三模式(42)中的中一个或所述第二模式(22)将所述语法元素部分(18)写码至所述片(14)中的所述当前片中，使得所述当前片被信号告知将被写码至其中，其中，所述编码器被配置为：

根据所述第一模式(20)，使用越过所述片边界的预测性编码来编码所述当前片；

根据所述第二模式(22)，使用具有将所述预测性编码限制为不越过所述片边界的预测性编码来编码所述当前片；以及

根据所述第三模式(42)，使用所述上下文适应熵编码和越过所述片边界的预测性写码将所述当前片编码至所述数据流中，所述上下文适应熵编码具有将所述上下文的所述导出限制为不越过所述片边界、所述上下文的符号概率的连续更新以及独立于任何先前编码的片的所述符号概率的初始化，

其中，所述编码器使用语法元素在所述第一模式与所述第三模式之间进行区分。

16.根据权利要求12所述的编码器，其中，所述编码器被配置为判定一般语法元素并且利用依据所述一般语法元素在至少两种操作模式中的一个下操作来将所述一般语法元素写入至所述数据流中，即根据第一操作模式，执行写码用于每个片的所述语法元素部分，并且根据第二操作模式，必然地使用所述至少两种模式中的不同于所述第一模式的不同模式。

17.根据权利要求13所述的编码器，其中，所述编码器被配置为根据所述第一模式和所述第二模式，必然且不间断地继续从所述当前片的开头至结尾连续地更新所述符号概率。

18.根据权利要求13所述的编码器，其中，所述编码器被配置为保存如在上下文适应熵编码所述先前编码的片直至所述先前编码的片的结尾时以及在根据所述第一模式初始化用于所述当前片的所述上下文适应熵编码的符号概率时所获得的所述符号概率，依据所保存的符号概率初始化用于所述当前片的所述上下文适应熵编码的所述符号概率。

19.根据权利要求15所述的编码器，其中，所述编码器被配置为在所述第一模式和所述第二模式下将所述预测性编码限制于所述图像被再分成的瓦片内。

20.一种用于从数据流(12)重构图像(10)的方法，所述图像以所述图像(10)被分割成的片(14)为单位而被写码至所述数据流中，其中，所述方法包括根据片次序(16)译码来自所述数据流(12)的所述片(14)，并且所述方法响应于所述片中的当前片内的语法元素部分(18)，来根据至少两种模式(20,22)中的一个对所述当前片进行译码，其中，

根据所述至少两种模式中的第一个(20)，使用上下文适应熵译码(24)对来自所述数据流(12)的所述当前片进行译码，所述上下文适应熵译码包括越过片边界的上下文的导出、所述上下文的符号概率的连续更新以及依据先前译码片的符号概率的所保存状态的所述符号概率的初始化(38,40)；并且

根据所述至少两种模式中的第二个(22)，使用所述上下文适应熵译码对来自所述数据流(12)的所述当前片进行译码，所述上下文适应熵译码具有将所述上下文的所述导出限制为不越过所述片边界、所述上下文的符号概率的连续更新以及独立于任何先前译码的片的所述符号概率的初始化。

21.一种用于将图像(10)以所述图像(10)被分割成的片(14)为单位而编码至数据流(12)中的方法，其中，所述方法包括根据片次序(16)将所述片(14)编码至所述数据流(12)中，并且所述方法包括：

判定用于所述片的当前片的语法元素部分(18)并且将所述语法元素部分写码至所述当前片中，使得用信号将所述语法元素部分发送至所述当前片以根据至少两种模式(20,22)中的一个进行写码；以及

如果将根据所述至少两种模式中的第一个(20)来写码所述当前片，则使用上下文适应熵编码(24)将所述当前片编码至所述数据流(12)中，所述上下文适应熵编码包括越过片边界的上下文的导出、所述上下文的符号概率的连续更新以及依据先前译码的片的符号概率的所保存状态的所述符号概率的初始化(38,40)，并且

如果根据所述至少两种模式中的第二个(22)来写码所述当前片，则使用所述上下文适应熵编码将所述当前片编码至所述数据流(12)中，所述上下文适应熵编码具有将所述上下文的所述导出限制为不越过所述片边界、所述上下文的符号概率的连续更新以及独立于任何先前编码片的所述符号概率的初始化。

22.一种计算机可读介质，具有存储在其上的计算机程序，所述计算机程序具有程序代码，当在计算机上运行所述程序代码时执行根据权利要求20的方法。

低延迟图像写码

技术领域

[0001] 本发明涉及图像的低延迟写码。

背景技术

[0002] 在当前HEVC设计片中，包含熵片(Entropy Slices)(从前的轻量片)瓦片(Tiles，小片)及WPP(波前并行处理)作为并行化工具。

[0003] 对于视频编码器及译码器的并行化，图像级分割与其它方法相比具有若干优势。在先前的视频编译码器中，如H.264/AVC[1]，图像分区仅对在写码效率方面具有高成本的规则片为可能。对于可缩放并行H.264/AVC译码，有必要将用于图像重构的宏区块级并行化与用于熵译码的帧级(frame-level)并行化组合。然而，此方法提供图像等待时间(picture latency)的有限减少以及较高的内存使用率。为了克服此等限制，已在HEVC编译码器中包含新的图像分区策略。当前参考软件版本(HM-6)含有4个不同方法：规则或正常片、熵片、波前并行处理(WPP)子流以及瓦片。通常，彼等图片分区包括一组最大写码单元(LCU)，或同义地，写码树单元(CTU)，如HEVC中所定义，或甚至那些单元的子集。

[0004] 图1展示为图像898，其按照图像中的LCU或宏区块的列902示例性地定位至规则片900中。规则片或正常片(如H.264[1]中所定义)具有最大写码损失，因为其打破了熵译码及预测依赖性。熵片(类似于片)打破熵译码依赖性，但允许预测(及滤波)越过片边界。

[0005] 在WPP中，图像分区经列交错，但允许熵译码及预测两者使用来自其它分区中之区块的数据。以此方式，写码损失得以最小化，同时波前并行化可得到利用。然而，交错违反了位流因果性，因为前一分区需要下一分区来译码。

[0006] 图2示例性地展示出图像898，其分为水平分区瓦片906之两个列904a、904b。瓦片界定水平边界908及垂直边界910，其将图像898分割为瓦片行912a、912b、912c以及列904a、904b。类似于规则片900，瓦片906打破熵译码及预测依赖性，但不需要用于每一瓦片的报头。

[0007] 对于此等技术中的每一个，编码器可自由选择分区的数目。一般而言，具有较多分区导致较高的压缩损失。然而，在WPP中，损失传播并非如此高，且因此甚至可将图像分区之数目固定为每列一个。此还产生若干优势。第一，对于WPP位流，因果性得以保证。第二，译码器实施方案可假定某一量的并行化为可用，该量还随分辨率而增加。并且，最后，当以波前次序译码时，不必打破上下文选择及预测依赖性两者，从而产生相对较低的写码损失。

[0008] 然而，迄今为止，变换概念中的所有并行写码均无法提供结合使延迟保持较低而达成高压缩效率。对于WPP概念，还为如此。在写码管线中，片为最小输送单位，且若干WPP子流仍必须串行输送。

发明内容

[0009] 因此，本发明的目标为提供一种图像写码概念，其允许根据例如波前并行处理以增加的效率(例如通过进一步减小端至端延迟，或通过减小将要花费的写码负担而增加写

码效率)来进行并行译码。

[0010] 此目标由独立权利要求的主题达成。

[0011] 本发明的基本发现为,若放弃片据此完全独立于图像在该相应的片外部的区域,且至少独立于在该相应的片外部远至熵写码与的有关的区域而写码/译码的普通片概念而有利于不同模式的片,即例如称为从属片的片(其允许跨片边界的相互依赖性),以及不允许跨片边界的相互依赖性的称为正常片的其它片,则可以减小的端至端延迟实现例如波前并行处理的并行处理概念。

[0012] 本发明的可与第一基本发现组合或相应的使用的进一步基本发现为,若片的开始语法部分用以定位WPP进入点,则可较高效地进行WPP处理概念。

附图说明

[0013] 下文关于各图来描述本申请的优选实施例,其中有利的实施例为从属权利要求的主题。附图中:

[0014] 图1展示出示例性地按照图像中的LCU或宏区块的列分割为规则片的图像;

[0015] 图2展示出示例性地分为水平分割瓦片的两个列的图像;

[0016] 图3示例性地展示出将并行编码分区指派给片或网络输送区段;

[0017] 图4展示出示出使用瓦片写码方法对帧进行一般分段来获得最小端至端延迟的示意图;

[0018] 图5展示出示出使用WPP写码方法对帧进行示例性分段来获得最小端至端延迟的示意图;

[0019] 图6显示出示出使用视频服务的对话的场景的示意性方块图;

[0020] 图7示意性地示出用具有最小端至端延迟的一般子集对瓦片进行编码、发射及译码的可能时间排程;

[0021] 图8示意性地展示出通常达成端至端延迟的时序进度表;

[0022] 图9示出示例性地具有11×9个写码树区块的图像,该图像分割为两个片;

[0023] 图10示出示例性地具有13×8个写码树区块的图像,该图像分割为三个瓦片;

[0024] 图11展示出序列参数组语法的实例;

[0025] 图12展示出图像参数组语法的实例;

[0026] 图13展示出片报头语法的实例;

[0027] 图14示出图像的用于WPP处理为规则片且用于低延迟处理为从属片的分区;

[0028] 图15展示出图像参数组语法内的一部分的实例;

[0029] 图16展示出可能的片报头语法;

[0030] 图17示意性地示出正常片(及从属片)的写码相互依赖性;

[0031] 图18展示出比较用于瓦片的低延迟输送的编码(使用从属片的波前并行处理)的示意图;

[0032] 图19示出时序进度表,其示出在使用利用如图18的右手侧处所示的从属片的波前并行处理时具有管线低延迟发射的示例性WPP写码;

[0033] 图20展示出示出通过使用规则片作为锚点(anchor)之稳健性改良的示意图;

[0034] 图21展示出片报头语法的另一实施例;

[0035] 图22展示出图像参数组语法的另一实施例；

[0036] 图23展示出示出在一左图像边界开始的情况下对从属片的符号概率初始化过程的示意图；

[0037] 图24展示出译码器的示意图；

[0038] 图25示意性地展示出译码器的方块图，并且示意性地示出将图像分割为写码块及片；

[0039] 图26示意性地展示出编码器的方块图；

[0040] 图27示意性地展示出分割为正常及从属片(此处称为片区段)的图像；

[0041] 图28a和图28b示意性地展示出分割为正常及从属片(此处一方面称为片区段，且另一方面称为瓦片)的图像；

[0042] 图29展示出示出使用从属片的上下文初始化过程的流程图；

[0043] 图30展示出示出用于使用从属片的上下文储存过程的流程图；以及

[0044] 图31示意性地展示出信号发送WPP进入点的不同的可能性。

具体实施方式

[0045] 在下文中，描述以如今的分别用于达成并行图像处理及低延迟写码的概念的描述来开始。当希望概述两种能力时，问题出现了。明确而言，如自以下论述将出来，目前所教示之WPP子流概念以某种方式与具有低延迟的愿望冲突，因为有必要通过将WPP子流分组为一个片来传达该WPP子流。以下实施例提出并行处理概念，例如WPP概念，其通过扩大片概念，即通过引入另一类型的片(后面称为从属片)来适用于需要更少延迟的应用。

[0046] 最小化自撷取至显示的端至端视频延迟为诸如视频会议等应用中的主要目的之一。

[0047] 用于数字视频发射的信号处理经由相机、撷取装置、编码器、包、发射、解多任务器、译码器、渲染器(renderer)及显示器组成。此等级中之每一者均通过在影像数据串行发射至后续级之前缓冲该影像数据来为端至端延迟作贡献。

[0048] 一些应用要求最小化此延迟，例如对有害区域中之对象的远程处置，而无需直接看到所处置之对象，或微创手术。甚至较短延迟均可导致恰当处置之严重困难，或甚至导致灾难性的错误。

[0049] 在许多情况下，在处理级内缓冲整个视频帧，例如来允许帧内处理。一些级搜集数据，以便形成可推进至下一级的包。一般而言，存在因本地处理之要求而导致的延迟之下部边界。下文较详细地针对每一个别级来分析此边界。

[0050] 相机内的处理不一定需要帧内信号处理，因此最小延迟由传感器的整合时间给出，该整合时间受帧速率以及硬件制造商的一些设计选择限制。相机输出通常与扫描次序有关，扫描次序通常开始左上角中的处理，移至右上角，且继续逐行至右下角。因此，在所有数据均自传感器传送至相机输出之前，花费约一个帧持续时间。

[0051] 撷取装置可在接收之后立即进送相机数据；然而，其将通常缓冲一些数据并产生丛发，以便最佳化对内存或储存装置之数据存取。此外，相机/撷取器与计算机之内存之间的连接通常限制将所撷取之影像数据进送至内存以供进一步处理(编码)的位率。通常，相机经由USB 2.0或很快与USB 3.0连接，USB 2.0或USB 3.0将总是包含影像数据向编码器的

部分输送。此举限制了极低延迟场景中编码器侧上的可并行化性，还即当数据变为自相机可用时，编码器将尝试尽可能快地开始编码，例如以光栅扫描次序自影像之顶部至底部。

[0052] 在编码器中，存在一些自由度，其允许依据某一视频保真度所需之数据速率来权衡编码效率，来减少处理延迟。

[0053] 编码器使用已经发送的数据来预测随后将要编码的影像。一般而言，可用比无预测的情况下所需的位少的位来编码实际影像与预测之间的差异。此预测值需要在译码器处可用，因此预测为基于同一影像之先前译码部分(帧内预测)，或基于早先已经处理之其它影像(帧间预测)。预HEVC视频编码标准仅使用上方或同一线中但在左侧的影像(其先前已编码)的部分来进行帧内预测、运动向量预测及熵写码(CABAC)。

[0054] 除预测结构的最佳化之外，可考虑并行处理的影响。并行处理要求识别可独立处理的图像区域。出于实践原因，选择诸如水平或垂直矩形之连续区，其通常被称为“瓦片”。在低延迟约束的情况下，彼等区应允许自撷取器传入至内存的尽可能快的并行化写码。假定光栅扫描内存传送，原始数据之垂直分区有意义，以便立即开始编码。在此等瓦片内，其将影像分为垂直分区(参看下文的图)，帧内预测、运动向量预测及熵写码(CABAC)可引起合理的写码效率。为了最小化延迟，图像之仅部分(自顶部开始)将传送至编码器之帧内存，且并行处理应在垂直瓦片中开始。

[0055] 允许并行处理之另一方式为在规则片内使用WPP，规则片将与瓦片(包含于单一片中的一“列”瓦片)进行比较。该瓦片内的数据可还使用该片内的WPP子流来并行编码。图3/1中以实例形式展示将图像分为片900及瓦片/WPP子流914。

[0056] 因此，图3展示出将经并行编码之分区(例如906或914)指派给片或网络输送区段(单一网络包或多个网络900包)。

[0057] 在发射之前或在编码过程期间将经编码数据封装至网络抽象化层(NAL)单元中(如H.264或HEVC中所界定)将某一报头添加至数据区块，此举允许识别每一区块，并在适用情况下对区块进行重新排序。在标准情况下，不需要额外发信号，因为写码元素之次序总是为译码次序，其为瓦片或一般写码片段之位置为给定的隐含指派。

[0058] 若并行处理被视为具有额外输送层用于低延迟并行输送，还即输送层可对用于瓦片之图像分区进行重新排序，以便允许低延迟发射，则意味着在片段被编码时将其发送出，如图4中所示。彼等片段还可为非完全编码片，其可为片之子集，或可包含于从属片中。

[0059] 在创建额外片段之情况下，存在效率(其对于大数据区块将为最高，因为报头信息为添加恒定数目个字节)与延迟之间的折衷，因为并行编码器之大数据区块在发射之前将需要缓冲。若垂直瓦片906之经编码表示被分为若干个片段916(其在片段一完成编码时就发射)，则可减小总延迟。可依据固定影像区(例如宏区块)、LCU或依据最大数据来判定每一片段之大小，如图4中所示。

[0060] 因此，图4展示出使用瓦片写码方法对帧进行一般分段来获得最小端至端延迟。

[0061] 类似地，图5展示出使用WPP写码方法对帧进行分段来获得最小端至端延迟。

[0062] 该发射可增加进一步延迟，例如若应用以额外区块为导向之处理，则例如正向错误校正码增加发射之稳健性。此外，网络基础结构(路由器等)或实体链接可增加延迟，此通常被称为连接之等待时间(latency)时。除等待时间之外，发射位率判定在如图6中所示的使用视频服务之对话中将数据自a方传送至b方的时间(延迟)。

[0063] 若无序发射经编码数据区块，则必须考虑重新排序延迟。

[0064] 数据单元一到达，就可开始译码，假定必须在此之前译码的其它数据单元为可用。

[0065] 在瓦片之情况下，瓦片之间无依赖性，因此可立即译码瓦片。若片段已产生为具有瓦片，例如每一片段若干个单独片，如图4中所示，则片段一分别被编码(其所含有的LCU或CU已被编码)，就可直接输送该片段。

[0066] 渲染器组合并行译码引擎之输出，并将经组合图像逐行进送至显示器。

[0067] 显示器不一定添加任何延迟，但在实践中，可在影像数据实际显示之前，可进行某一帧内处理。此由硬件制造商之设计选择判定。

[0068] 总之，可影响级编码、封装、发射及译码，以便达成最小端至端延迟。若使用并行处理、瓦片及瓦片内的分段，则与如图8中所示在此等级中之每一者处增加约一帧延迟的通常使用之处理链相比，总延迟可显著减小，如图7中所示。

[0069] 明确而言，虽然图7展示以最小端至端延迟对具有一般子集之瓦片的编码、发射及译码，但图8示出通常达成之端至端延迟。

[0070] HEVC允许使用片分区、瓦片分区，且以如下方式。

[0071] 瓦片：共同出现于一个行及一个列中的整数数目个树区块，其在瓦片之树区块光栅扫描中连续排序。将每一图像分为若干瓦片为分区。图像中之瓦片在该图像之瓦片光栅扫描中连续排序。尽管片含有在瓦片之树区块光栅扫描中连续的树区块，但此等树区块在图像之树区块光栅扫描中不一定为连续的。

[0072] 片：光栅扫描中连续排序的整数数目个树区块。将每一图像分为若干片为分区。自片中的第一树区块地址(如在片报头中表示)得出树区块地址。

[0073] 光栅扫描：将矩形二维图案映像至一维图案，使得该一维图案中之第一条目来自自左至右扫描之二维图案的第一顶部列，类似地接以该图案之第二、第三等列(下降)，其各自为自左至右扫描。

[0074] 树区块：具有三个样本数组之图像的亮度样本之N×N区块以及色度样本之两个对应区块，或单色图像或使用三个单独色彩平面写码之图像的NxN样本区块。将片分为若干树区块为分区。

[0075] 分区：将一组分为若干子集，使得该组之每一元素正好为该子集中之一者。

[0076] 四元树：其中母节点可分为四个子节点的树。子节点可成为再次分为四个子节点的母节点。

[0077] 在下文中，阐释图像、片及瓦片之空间细分。明确地说，以下描述指定图像如何分割为片、瓦片及写码树区块。将图像分为片及瓦片。片为写码树区块序列。同样地，瓦片为写码树区块序列。

[0078] 以写码树区块为单位来处理样本。样本中之每一树区块在宽度及高度上的亮度数组大小为CtbSize。每一写码树区块之色度数组的宽度及高度分别为CtbWidthC及CtbHeightC。举例而言，可将图像分为两个片，如下一图中所示。作为另一实例，可将图像分为三个瓦片，如接下来第二个图中所示。

[0079] 不同于片，瓦片总是为矩形，且总是在写码树区块光栅扫描中含有整数数目个写码树区块。瓦片可由一个以上片中所含有的写码树区块组成。类似地，片可包括一个以上瓦片中所含有的写码树区块。

[0080] 图9示出分割为两个片900a、900b的具有11乘9个写码树区块918的图像898。

[0081] 图10示出分割为三个瓦片的具有13乘8个写码树区块918的图像。

[0082] 为每一写码898树区块918指派一分区信令，以识别区块大小进行帧内或帧间预测且进行变换写码。分区为递归四元树分区。四元树之根部与写码树区块相关联。分裂四元树，直至达到叶为止，该叶称为写码区块。写码区块为两个树(预测树及变换树)之根节点。

[0083] 预测树指定预测区块之位置及大小。预测区块及相关联的预测数据称为预测单元。

[0084] 图11展示出示例性序列参数组RBSP语法。

[0085] 变换树指定变换区块之位置及大小。变换区块及相关联的变换数据称为变换单元。

[0086] 针对亮度及色度之分裂信息对于预测树来说为相同的，且对于变换树可或可不为相同的。

[0087] 写码区块、相关联的写码数据以及相关联的预测及变换单元一起形成写码单元。

[0088] 用于将写码树区块光栅次序之写码树区块地址转换为瓦片扫描次序的过程可为如下：

[0089] 此过程之输出为

[0090] -数组CtbAddrTS[ctbAddrRS]，具有在0至PicHeightInCtbs*PicWidthInCtbs-1(包含0及PicHeightInCtbs*PicWidthInCtbs-1)中的ctbAddrRS。

[0091] -数组TileId[ctbAddrTS]，具有在0至PicHeightInCtbs*PicWidthInCtbs-1(包含0及PicHeightInCtbs*PicWidthInCtbs-1)中的ctbAddrTS。

[0092] 数组CtbAddrTS[ ]如下得出：

[0093]
```
for( ctbAddrRS = 0; ctbAddrRS < PicHeightInCtbs * PicWidthInCtbs; ctbAddrRS++ ) {
    tbX = ctbAddrRS % PicWidthInCtbs
    tbY = ctbAddrRS / PicWidthInCtbs
    for( j = 0; j <= num_tile_columns_minus1; j++ )
        if( tbX < ColBd[ j + 1 ] )
            tileX = j
    for( i = 0; i <= num_tile_rows_minus1; i++ )
        if( tbY < RowBd[ i + 1 ] )
            tileY = i
    CtbAddrTS[ ctbAddrRS ] = ctbAddrRS - tbX
            for( i = 0; i < tileX; i++ )
        ctbAddrTS += RowHeight[ tileY ] * ColumnWidth[ i ]
    CtbAddrTS[ ctbAddrRS ] += ( tbY - RowBd[ tileY ] ) * ColumnWidth[ tileY ] + tbX - ColBd[ tileX ]
}
```

[0094] 数组TileId[ ]如下得出：

[0095]
```
for( j = 0, tileId = 0; j <= num_tile_columns_minus1; j++ )
    for( i = 0; i <= num_tile_rows_minus1; i++, tileId++ )
        for( y = RowBd[ j ]; y < RowBd[ j + 1 ]; y++ )
            for( x = ColBd[ i ]; x < ColBd[ i + 1 ]; x++ )
                TileId[ CtbAddrTS[ y*PicWidthInCtbs + x ] ] = tileId
```

[0096] 图11、图12及图13中展示对应的示例性语法，其中图12具有示例性图像参数组RBSP语法。图13展示示例性片报头语法。

[0097] 在语法实例中，以下语义可适用：

[0098] 等于1的entropy_slice_flag指定推断不存在的片报头语法元素的值等于前一片

中的片报头语法元素的值，其中将前一片界定为含有具有位置(SliceCtbAddrRS-1)的写码树区块的片。当SliceCtbAddrRS等于0时，entropy_slice_flag将等于0。

[0099] 等于0的tiles_or_entropy_coding_sync_idc指定经写码视频序列中的每一图像中仅存在一个瓦片，且在译码一列写码树区块之第一写码树区块之前，不调用上下文变量之特定同步过程。

[0100] 等于1的tiles_or_entropy_coding_sync_idc指定经写码视频序列中的每一图像中可存在一个以上瓦片，且在译码一列写码树区块之第一写码树区块之前，不调用上下文变量之特定同步过程。

[0101] 等于2的tiles_or_entropy_coding_sync_idc指定经写码视频序列中之每一图像中仅存在一个瓦片，在译码一列写码树区块之第一写码树区块之前，调用上下文变量之特定同步过程，且在译码一列写码树区块之两个写码树区块之后，调用上下文变量之特定存储过程。

[0102] tiles_or_entropy_coding_sync_idc的值将在0至2(包含0及2)的范围内。

[0103] num_tile_columns_minus1加1指定分割图像之瓦片行的数目。

[0104] num_tile_columns_minus1加1指定分割图像之瓦片列的数目。

[0105] 当num_tile_columns_minus1等于0时，num_tile_rows_minus1将不等于0。

[0106] 对于每一片及瓦片，将满足以下条件中的一者或两者：

[0107] -片中之所有经写码区块均属于同一瓦片。

[0108] -瓦片中之所有经写码区块均属于同一片。

[0109] 注意-在同一图像内，可存在含有多个瓦片的片以及含有多个片之瓦片。

[0110] 等于1的uniform_spacing_flag指定行边界且同样地列边界在图像上均匀地分布。等于0的uniform_spacing_flag指定行边界且同样地列边界不在图像上均匀地分布，而是使用语法元素column_width[i]及row_height[i]来明确地发信号。

[0111] column_width[i]指定以写码树区块为单位的第i个瓦片行的宽度。

[0112] row_height[i]指定以写码树区块为单位的第i个瓦片列的高度。

[0113] 如下得出行宽度[i]的值，其指定以写码树区块为单位之第i个瓦片行的宽度，以及ColumnWidthInLumaSamples[i]的值，其指定以亮度样本为单位的第i个瓦片行的宽度：

[0114]
```
for( i = 0; i <= num_tile_columns_minus1; i++ ) {
  if( uniform_spacing_flag )
    ColumnWidth[ i ] = ( ( i + 1 ) * PicWidthInCtbs ) / ( num_tile_columns_minus1 + 1 ) -
                       ( i * PicWidthInCtbs ) / ( num_tile_columns_minus1 + 1 )
  else
    ColumnWidth[ i ] = column_width[ i ]
  ColumnWidthInLumaSamples[ i ] = ColumnWidth[ i ] << Log2CtbSize
}
```

[0115] 如下得出RowHeight[i]的值，其指定以写码树区块为单位的第i个瓦片列的高度：

[0116]
```
for( i = 0; i <= num_tile_rows_minus1; i++ )
    if( uniform_spacing_flag )
        RowHeight[ i ] = ( ( i + 1 ) * PicHeightInCtbs ) / ( num_tile_rows_minus1 + 1 ) -
                         ( i * PicHeightInCtbs ) / ( num_tile_rows_minus1 + 1)
    else
        RowHeight[ i ] = row_height[ i ]
```

[0117] 如下得出ColBd[i]的值，其指定以写码树区块为单位的第i个瓦片行的左行边界的位置：

[0118] for(ColBd[0]＝0,i＝0;i＜＝num_tile_columns_minus1;i++)

[0119] ColBd[i+1]＝ColBd[i]+ColumnWidth[i]

[0120] 如下得出RowBd[i]的值，其指定以写码树区块为单位的第i个瓦片列的顶部列边界的位置：

[0121] for(RowBd[0]＝0,i＝0;i＜＝num_tile_rows_minus1;i++

[0122] RowBd[i+1]＝RowBd[i]+RowHeight[i]

[0123] 当tiles_or_entropy_coding_sync_idc等于2时，num_substreams_minus1加1指定包含于片中的子集的最大数目。当不存在时，推断num_substreams_minus1的值等于0。

[0124] num_entry_point_offsets指定片报头中的entry_point_offset[i]语法元素的数目。当tiles_or_entropy_coding_sync_idc等于1时，num_entry_point_offsets的值将在0至(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1)-1(包含0及(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1)-1)的范围内。当tiles_or_entropy_coding_sync_idc等于2时，num_entry_point_offsets的值将在0至num_substreams_minus1(包含0及num_substreams_minus1)的范围内。当不存在时，推断num_entry_point_offsets的值等于0。

[0125] offset_len_minus1加1指定entry_point_offset[i]语法元素的以位为单位的长度。

[0126] entry_point_offset[i]指定以字节为单位的第i个进入点偏移，且将由offset_len_minus1加1个位表示。经写码片NAL单元由num_entry_point_offsets+1个子集组成，其中子集索引值的范围为自0至num_entry_point_offsets，包含0及num_entry_point_offsets。子集0由经写码片NAL单元之字节0至entry_point_offset[0]-1(包含0及entry_point_offset[0]-1)组成，子集k(其中k在1至num_entry_point_offsets-1(包含1及num_entry_point_offsets-1)的范围内)由经写码片NAL单元的字节entry_point_offset[k-1]至entry_point_offset[k]+entry_point_offset[k-1]-1(包含entry_point_offset[k-1]及entry_point_offset[k]+entry_point_offset[k-1]-)组成，且最后一个子集(具有等于num_entry_point_offsets的子集索引)由经写码片NAL单元的其余字节组成。

[0127] 注意-经写码片NAL单元的NAL单元报头及片报头总是包含于子集0中。

[0128] 当tiles_or_entropy_coding_sync_idc等于1且num_entry_point_offsets大于0时，每一子集将含有一个或多个完整瓦片的所有经写码位，且子集之数目将等于或小于片中之瓦片的数目。

[0129] 当tiles_or_entropy_coding_sync_idc等于2且num_entry_point_offsets大于0时，对于所有可能k个值中之每一者，子集k将含有将在当前位流指针k之初始化过程期间使

用的所有位。

[0130] 关于片数据语义，以下可适用。

[0131] 等于0的end_of_slice_flag指定片中随后为另一宏区块。等于1的end_of_slice_flag指定片之结尾，且后面无进一步宏区块。

[0132] entry_point_marker_two_3bytes为等于0x000002之3个字节的固定值序列。此语法元素称为条目标记前缀。

[0133] tile_idx_minus_1指定光栅扫描次序中之TileID。图像中之第一瓦片将具有TileID 0。tile_idx_minus_1的值将在0至(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1)-1的范围内。

[0134] 对片数据之CABAC剖析过程可为如下：

[0135] 当以描述符ae(v)剖析语法元素时，调用此过程。

[0136] 此过程之输入为对语法元素之值以及先前剖析之语法元素之值的请求。

[0137] 此过程之输出为语法元素之值。

[0138] 当开始剖析片的片数据时，调用CABAC剖析过程之初始化过程。当tiles_or_entropy_coding_sync_idc等于2且num_substreams_minus1大于0时，如下得出具有指定将用于稍后的当前位流指针导出之位流指针表的num_substreams_minus1+1个条目的映像表BitStreamTable(位流表)。

[0139] -使BitStreamTable[0]初始化以含有位流指针。

[0140] -对于大于0且小于num_substreams_minus1+1的所有索引i，BitStreamTable[i]含有指向BitStreamTable[i-1]之后的entry_point_offset[i]个字节的位流指针。

[0141] 将当前位流指针设定为BitStreamTable[0]。

[0142] 例如如下使用当前写码树区块之左上亮度样本之位置(x0,y0)来得出含有空间相邻区块T之写码树区块的最小写码区块地址ctbMinCbAddrT。

[0143] x＝x0+2<<Log2CtbSize-1

[0144] y＝y0-1

[0145] ctbMinCbAddrT＝MinCbAddrZS[x>>Log2MinCbSize][y>>Log2MinCbSize]

[0146] 通过调用以ctbMinCbAddrT作为输入适当的写码区块可用性导出过程来获得变量availableFlagT。

[0147] 当开始剖析写码树，并且tiles_or_entropy_coding_sync_idc等于2，且num_substreams_minus1大于0时，以下适用。

[0148] -若CtbAddrRS%PicWidthInCtbs等于0，则以下适用。

[0149] -当availableFlagT等于1时，调用CABAC剖析过程之同步过程，如分单元“上下文变量之同步过程”中所指定。

[0150] -在算术译码引擎之初始化过程之后，调用终止之前的对二元决策的译码过程。

[0151] -设定当前位流指针，来指示具有如下得出之索引i的BitStreamTable[i]。

[0152] i＝(CtbAddrRS/PicWidthInCtbs)%(num_substreams_minus1+1)

[0153] -否则，若CtbAddrRS%PicWidthInCtbs等于2，则调用CABAC剖析过程之存储过程，如分单元“上下文变量之存储过程”中所指定。

[0154] 初始化过程可为如下：

[0155] 此过程之输出为经初始化的CABAC内部变量。

[0156] 当开始剖析片的片数据时，或当开始剖析写码树之数据且写码树为瓦片中的第一写码树时，调用其特殊过程。

[0157] 上下文变量之存储过程可为如下：

[0158] 此过程之输入为由ctxIdx索引之CABAC上下文变量。

[0159] 此过程之输出为变量TableStateSync及TableMPSSync，其含有指派给除片结尾标记之外的语法元素的上下文变量之初始化过程中所使用的变量m及n的值。

[0160] 对于每一上下文变量，使表TableStateSync及TableMPSSync之对应条目n及m初始化为对应的pStateIdx及valMPS。

[0161] 上下文变量之同步过程可为如下：

[0162] 此过程之输入为变量TableStateSync及TableMPSSync，其含有指派给除片结尾标记之外的语法元素的上下文变量之存储过程中所使用的变量n及m的值。

[0163] 此过程之输出为由ctxIdx索引之CABAC上下文变量。

[0164] 对于每一上下文变量，使对应的上下文变量pStateIdx及valMPS初始化为表TableStateSync及TableMPSSync之对应条目n及m。

[0165] 在下文中，阐释使用WPP之低延迟写码及输送。明确而言，以下论述揭示如图7中所描述的低延迟输送如何还可应用于WPP。

[0166] 首先，重要的是可在整个图像完成之前，发送该图像的子集。通常，此可使用片来达成，如图5中已经展示。

[0167] 为了减小与瓦片相比之延迟，如以下图中所示，需要按照LCU的列应用单一WPP子流，且进一步允许彼等列中之每一者的单独发射。为了使写码效率保持较高，不可使用每一列/子流的片。因此，在下文中，引入如下一部分中所界定之所谓的从属片。举例而言，此片不具有完整HEVC片报头之所有字段，但具有用于熵片之字段。此外，可存在开关，以断开列之间的CABAC的中断。在WPP之情况下，将允许使用CABAC上下文(图14中的箭头)及列之预测，以保持WPP在瓦片上的写码效率增益。

[0168] 明确而言，图14示出用于WPP为规则片900(Reg.SL)且用于低延迟处理为从属片(OS)920的图像10。

[0169] 目前，即将出现的HEVC标准提供两种类型的依据片的分割。存在规则(正常)片及熵片。规则片为除某一依赖性之外完全独立的图像分区，其可归因于片边界上的解块滤波过程而可用。熵片还为独立的，但仅在熵写码方面独立。图14之理念为概括切片概念。因此，即将出现的HEVC标准应提供两种一般类型的片：独立(规则)或从属。因此，引入一种新类型的片，从属片。

[0170] 从属片为对先前片具有依赖性的片。依赖性为可在熵译码过程及/或像素重构过程中的片之间利用的特定数据。

[0171] 在图14中，示例性地呈现从属片之概念。举例而言，图像总是以规则片开始。注意，在此概念中，规则片行为稍稍概念。通常，在如H264/AVC或HEVC之标准中，规则片为完全独立的分区，且除用于解块滤波过程之一些数据外，在译码之后不必保持任何数据。但即将出现的从属片920之处理仅通过参考以上片(此处为第一列中)：规则片900之数据而为可能的。为了建立此，规则片900应保持最后一个CU列之数据。此数据包括：

[0172] -CABAC写码引擎数据(一个CU之上下文模型状态,从属片之熵译码过程可自其初始化),

[0173] -从属CU之规则CABAC译码过程之CU的所有经译码语法元素,

[0174] -帧内及运动向量预测之数据。

[0175] 因此,每一从属片920将进行同一程序—保持同一图像中之即将出现的从属片的数据。

[0176] 实际上,此等额外步骤不应成为问题,因为一般而言总是迫使译码过程储存如语法元素等一些数据。

[0177] 在以下部分中,呈现达成从属片之概念所需的对HEVC标准语法之可能改变。

[0178] 举例而言,图15示出图像参数组RBSP语法中之可能改变。

[0179] 用于从属片之图像参数组语义可为如下:

[0180] 等于1的dependent_slices_present_flag指定图像含有从属片,且每一(规则或从属)片之译码过程将储存熵译码之状态以及可为从属片(其还可在规则片之后)之下一片的帧内及运动向量预测的数据。以下从属片可参考该所储存之数据。

[0181] 图16展示出具有相对于HEVC之当前状态的改变的可能slice_header语法。

[0182] 等于1的dependent_slice_flag指定推断不存在的片报头语法元素的值等于前一(规则)片中的片报头语法元素的值,其中将前一片界定为含有具有位置(SliceCtbAddrRS-1)的写码树区块的片。当SliceCtbAddrRS等于0时,dependent_slice_flag将等于0。

[0183] 等于1的no_cabac_reset_flag指定自先前译码的片(且不具有初始值)之所保存状态的CABAC初始化。否则,还即若no_cabac_reset_flag等于1,则指定独立于先前经译码片(还即具有初始值)之任何状态的CABAC初始化。

[0184] 等于1的last_ctb_cabac_init_flag指定自(例如用于总是等于1之瓦片的)先前译码的片之最后写码之树区块的所保存状态的CABAC初始化。否则(等于0),若当前片之第一写码之树区块为列中之第一经写码树区块(还即,WPP模式),则自先前译码的片的最后(相邻)ctb列之第二经写码树区块之所保存状态参考初始化数据,否则自先前译码的片的最后经写码树区块的所保存状态执行CABAC初始化。

[0185] 下文提供从属片与其它分区方案(信息性)的比较。

[0186] 在图17中,展示正常片与从属片之间的差异。

[0187] 如相对于图18所示出的从属片(DS)中之WPP子流的可能写码及发射比较瓦片(左)与WPP/DS(右)之低延迟输送的编码。图18中的粗体连续绘制交叉展示两种方法的时间的同一时间点,假定WPP列之编码与单一瓦片之编码花费相同时间。归因于写码依赖性,在所有瓦片均已编码之后,仅WPP之第一线为就绪。但一旦第一列被编码,就使用从属片方法允许WPP方法来发送出该第一列。此不同于早先的WPP子流指派,针对WPP将“子流”界定为将要由同一译码器执行绪(还即,同一核/处理器)进行WPP译码的片的CU列的序连连接。尽管每列及每熵片子流在以前还将可能,但熵片破坏熵写码依赖性,且因此具有较低的写码效率,还即丧失WPP效率增益。

[0188] 另外,在假定如图19中所示的发射的情况下,两种方法之间的延迟差异可为相当低。明确而言,图19说明具有管线化低延迟发射的WPP写码。

[0189] 假定图18中之WPP方法中之DS#1.1的后两个CU的编码不花费比第一列SL#1之发射

长的时间，在低延迟情况下，瓦片与WPP之间不存在差异。但WP/DS之写码效率胜过瓦片概念。

[0190] 为了增加WPP低延迟模式之稳健性，图20示出稳健性改良系通过使用规则片(RS)作为锚点来达成。在图20所示的图像中，(规则)片(RS)之后为从属片(DS)。此处，(规则)片充当锚点以打破对先前片之依赖性，因此在(规则)片之此插入点处提供较多稳健性。原则上，此无论自什么角度均与插入(规则)片相同。

[0191] 还可如下实施从属片之概念。

[0192] 此处，图21展示可能的片报头语法。

[0193] 片报头语义如下：

[0194] 等于1的dependent_slice_flag指定推断不存在的每一片报头语法元素的值等于含有写码树区块地址为SliceCtbAddrRS-1之写码树区块的先前片中之对应片报头语法元素的值。当不存在时，推断dependent_slice_flag的值等于0。当SliceCtbAddrRS等于0时，dependent_slice_flag的值将等于0。

[0195] slice_address指定片于其中开始的片粒度分辨率中的地址。slice_address语法元素之长度为(Ceil(Log2(PicWidthInCtbs*PicHeightInCtbs))+SliceGranularity)个位。

[0196] 如下得出指定片在其中以写码树区块光栅扫描次序开始的写码树区块。

[0197] SliceCtbAddrRS＝(slice_address>>SliceGranularity)

[0198] 如下得出指定片中之第一写码区块在z形扫描次序中的最小写码区块粒度中的地址的变量SliceCbAddrZS。

[0199] SliceCbAddrZS＝slice_address

[0200] <<((log2_diff_max_min_coding_block_size-SliceGranularity)<<1)

[0201] 片译码以片开始坐标处的可能最大写码单元或换言之CTU开始。

[0202] first_slice_in_pic_flag指示该片是否为图像之第一片。若first_slice_in_pic_flag等于1，则将SliceCbAddrZS及SliceCtbAddrRS均设定为0，且译码以图像中之第一写码树区块开始。

[0203] pic_parameter_set_id指定使用中的图像参数组。pic_parameter_set_id的值将在0至255(包含0及255)的范围内。

[0204] num_entry_point_offsets指定片报头中的entry_point_offset[i]语法元素的数目。当tiles_or_entropy_coding_sync_idc等于1时，num_entry_point_offsets的值将在0至(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1)-1(包含0及(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1)-1)的范围内。当tiles_or_entropy_coding_sync_idc等于2时，num_entry_point_offsets的值将在0至PicHeightInCtbs-1(包含0及PicHeightInCtbs-1)的范围内。当不存在时，推断num_entry_point_offsets的值等于0。

[0205] offset_len_minus1加1指定entry_point_offset[i]语法元素之以位为单位的长度。

[0206] entry_point_offset[i]指定以字节为单位的第i个进入点偏移，且将由offset_len_minus1加1个位表示。片报头之后的经写码片数据由num_entry_point_offsets+1个子

集组成，其中子集索引值的范围为自0至num_entry_point_offsets，包含0及num_entry_point_offsets。子集0由经写码片数据之字节0至entry_point_offset[0]-1(包含0及entry_point_offset[0]-1)组成，子集k(其中k在1至num_entry_point_offsets-1(包含1及num_entry_point_offsets-1)的范围内)由经写码片数据之字节entry_point_offset[k-1]至entry_point_offset[k]+entry_point_offset[k-1]-1(包含entry_point_offset[k-1]及entry_point_offset[k]+entry_point_offset[k-1]-1)组成，且最后一个子集(具有等于num_entry_point_offsets的子集索引)由经写码片数据之其余字节组成。

[0207] 当tiles_or_entropy_coding_sync_idc等于1，且num_entry_point_offsets大于0时，每一子集将含有恰好一个瓦片之所有经写码位，且子集之数目(还即，num_entry_point_offsets+1的值)将等于或小于片中之瓦片的数目。

[0208] 注意-当tiles_or_entropy_coding_sync_idc等于1时，每一片必须包含一个瓦片之子集(在此情况下，进入点之发信号为不必要的)，或完整瓦片之整数数目。

[0209] 当tiles_or_entropy_coding_sync_idc等于2且num_entry_point_offsets大于0时，每一子集k(其中k在0至num_entry_point_offsets-1的范围内，包含0及num_entry_point_offsets-1)将含有正好一列写码树区块之所有经写码位，最后一个子集(具有等于num_entry_point_offsets的子集索引)将含有包含于片中的其余写码区块的所有经写码位，其中其余写码区块由正好一列写码树区块或一列写码树区块之子集组成，且子集之数目(还即，num_entry_point_offsets+1的值)将等于片中之写码树区块之列的数目，其中还对片中之一列写码树区块的子集进行计数。

[0210] 注意，当tiles_or_entropy_coding_sync_idc等于2时，片可包含若干列写码树区块以及一列写码树区块之子集。举例而言，若片包含两个半列写码树区块，则子集的数目(还即，num_entry_point_offsets+1的值)将等于3。

[0211] 可如图22中所示选择对应的图像参数组RBSP语法。

[0212] 图像参数组RBSP语义可为如下：

[0213] 等于1的dependent_slice_enabled_flag指定参考图像参数组之经写码图像的片报头中语法元素dependent_slice_flag的存在。等于0的dependent_slice_enabled_flag指定参考图像参数组之经写码图像的片报头中语法元素dependent_slice_flag的不存在。当tiles_or_entropy_coding_sync_idc等于3时，dependent_slice_enabled_flag的值将等于1。

[0214] 等于0的tiles_or_entropy_coding_sync_idc指定参考图像参数组之每一图像中将仅存在一个瓦片，在译码参考该图像参数组之每一图像中之一列写码树区块之第一写码树区块之前，将不调用对上下文变量的特定同步过程，且参考该图像参数组之经写码图像之cabac_independent_flag及dependent_slice_flag的值将不两者均等于1。

[0215] 注意，当cabac_independent_flag及depedent_slice_flag两者对于一片均等于1时，该片为熵片。

[0216] 等于1的tiles_or_entropy_coding_sync_idc指定参考图像参数组之每一图像中可存在一个以上瓦片，在译码参考该图像参数组之每一图像中之一列写码树区块之第一写码树区块之前，将不调用对上下文变量的特定同步过程，且参考该图像参数组之经写码图像之cabac_independent_flag及dependent_slice_flag的值将不两者均等于1。

[0217] 等于2的tiles_or_entropy_coding_sync_idc指定参考图像参数组之每一图像中将仅存在一个瓦片，在译码参考该图像参数组之每一图像中之一列写码树区块之第一写码树区块之前，将调用对上下文变量之特定同步过程，且在译码参考该图像参数组之每一图像中之一列写码树区块之两个写码树区块之后，将调用对上下文变量之特定存储过程，且参考该图像参数组之经写码图像之cabac_independent_flag及dependent_slice_flag的值将不两者均等于1。

[0218] 等于3的tiles_or_entropy_coding_sync_idc指定参考图像参数组之每一图像中将仅存在一个瓦片，在译码参考该图像参数组之每一图像中之一列写码树区块之第一写码树区块之前，将不调用对上下文变量的特定同步过程，且参考该图像参数组之经写码图像之cabac_independent_flag及dependent_slice_flag的值可两者均等于1。

[0219] 当dependent_slice_enabled_flag将等于0时，tiles_or_entropy_coding_sync_idc将不等于3。

[0220] 位流一致性的要求为tiles_or_entropy_coding_sync_idc的值对于在经写码视频序列内启动之所有图像参数组均将为相同。

[0221] 对于参考图像参数组的每一片，当tiles_or_entropy_coding_sync_idc等于2且片中之第一写码区块并非一列写码树区块之第一写码树区块中的第一写码区块时，片中之最后一个写码区块将属于与该片中之第一写码区块相同的写码树区块列。

[0222] num_tile_columns_minus1加1指定分割图像之瓦片行的数目。

[0223] num_tile_columns_minus1加1指定分割图像之瓦片行的数目。

[0224] 当num_tile_columns_minus1等于0时，num_tile_rows_minus1将不等于0。等于1的uniform_spacing_flag指定行边界且同样地列边界在图像上均匀地分布。等于0的uniform_spacing_flag指定行边界且同样地列边界不在图像上均匀地分布，而是使用语法元素column_width[i]及row_height[i]来明确地发信号。

[0225] column_width[i]指定以写码树区块为单位的第i个瓦片行的宽度。

[0226] row_height[i]指定以写码树区块为单位之第i个瓦片列的高度。

[0227] 向量colWidth[i]指定以CTB为单位的第i个瓦片行的宽度，其中行i的范围为0至num_tile_columns_minus1，包含0及num_tile_columns_minus1。

[0228] 向量CtbAddrRStoTS[ctbAddrRS]指定自光栅扫描次序之CTB地址至瓦片扫描次序之CTB地址的转换，其中索引ctbAddrRS的范围自0至(picHeightInCtbs*picWidthInCtbs)-1，包含0及(picHeightInCtbs*picWidthInCtbs)-1。

[0229] 向量CtbAddrTStoRS[ctbAddrTS]指定自瓦片扫描次序之CTB地址至光栅扫描次序之CTB地址的转换，其中索引ctbAddrTS的范围自0至(picHeightInCtbs*picWidthInCtbs)-1，包含0及(picHeightInCtbs*picWidthInCtbs)-1。

[0230] 向量TileId[ctbAddrTS]指定自瓦片扫描次序之CTB地址至瓦片id的转换，其中ctbAddrTS的范围自0至(picHeightInCtbs*picWidthInCtbs)-1，包含0及(picHeightInCtbs*picWidthInCtbs)-1。

[0231] 通过调用以PicHeightInCtbs及PicWidthInCtbs作为输入且输出被指派给colWidth、CtbAddrRStoTS及TileId之CTB光栅及瓦片扫描转换过程来得出colWidth、CtbAddrRStoTS、CtbAddrTStoRS及TileId的值。

[0232] 将ColumnWidthInLumaSamples[i]的值(其指定以亮度样本为单位的第i个瓦片行的宽度),设定为等于colWidth[i]<<Log2CtbSize。

[0233] 通过调用以Log2MinCbSize、Log2CtbSize、PicHeightInCtbs、PicWidthInCtbs及向量CtbAddrRStoTS作为输入,且输出被指派给MinCbAddrZS的Z形扫描次序数组初始化过程来得出指定自以最小CB为单位的位置(x,y)至z形扫描次序的最小CB地址的转换的数组MinCbAddrZS[x][y],其中x的范围自0至picWidthInMinCbs-1,包含0及picWidthInMinCbs-1,且y的范围自0至picHeightInMinCbs-1,包含0及picHeightInMinCbs-1。

[0234] 等于1的loop_filter_across_tiles_enabled_flag指定越过瓦片边界执行回路内滤波操作。等于0的loop_filter_across_tiles_enabled_flag指定不越过瓦片边界执行回路内滤波操作。回路内滤波操作包含解块滤波器、样本适应偏移以及适应回路滤波器操作。当不存在时,推断loop_filter_across_tiles_enabled_flag的值等于1。

[0235] 等于1的cabac_independent_flag指定片中之写码区块的CABAC译码独立于先前译码的片的任何状态。等于0的cabac_independent_flag指定片中之写码区块的CABAC译码依赖于先前译码的片的状态。当不存在时,推断cabac_independent_flag的值等于0。

[0236] 具有最小写码区块地址的写码区块的可用性的导出过程可为如下:

[0237] 此过程的输入为

[0238] -z形扫描次序中的最小写码区块地址minCbAddrZS

[0239] -z形扫描次序中的当前最小写码区块地址currMinCBAddrZS

[0240] 此过程之输出为具有z形扫描次序中的最小写码区块地址cbAddrZS的写码区块的可用性cbAvailable。

[0241] 注意1-当调用此过程时,判定可用性之意义。

[0242] 注意2-任何写码区块(不管其大小如何)均与最小写码区块地址相关联,该地址为具有z形扫描次序中之最小写码区块大小的写码区块的地址。

[0243] -若以下条件中之一或多者为真,则将cbAvailable设定为假。

[0244] -minCbAddrZS小于0

[0245] -minCbAddrZS大于currMinCBAddrZS

[0246] -具有最小写码区块地址minCbAddrZS的写码区块属于与具有当前最小写码区块地址currMinCBAddrZS之写码区块不同的片,且含有具有当前最小写码区块地址currMinCBAddrZS之写码区块的片的dependent_slice_flag等于0。

[0247] -具有最小写码区块地址minCbAddrZS的写码区块包含于与具有当前最小写码区块地址currMinCBAddrZS的写码区块不同的瓦片中。

[0248] -否则,将cbAvailable设定为真。

[0249] 对片数据的CABAC剖析过程可为如下:

[0250] 当以描述符ae(v)剖析某些语法元素时,调用此过程。

[0251] 此过程之输入为对语法元素之值以及先前剖析之语法元素之值的请求。

[0252] 此过程之输出为语法元素之值。

[0253] 当开始剖析片的片数据时,调用CABAC剖析过程之初始化过程。

[0254] 图23示出如何使用空间相邻者T来调用相对于当前写码树区块(信息性)的写码树区块可用性导出过程。

[0255] 如下使用当前写码树区块之左上亮度样本之位置(x0,y0)来得出含有空间相邻区块T(图23)之写码树区块的最小写码区块地址ctbMinCbAddrT。

[0256] x＝x0+2<<Log2CtbSize-1

[0257] y＝y0-1

[0258] ctbMinCbAddrT＝MinCbAddrZS[x>>Log2MinCbSize][y>>Log2MinCbS ize]

[0259] 通过调用以ctbMinCbAddrT作为输入的写码区块可用性导出过程来获得变量availableFlagT。

[0260] 当开始如所指定之写码树的剖析时,以下经排序步骤适用。

[0261] 如下初始化算术译码引擎。

[0262] 若CtbAddrRS等于slice_address,dependent_slice_flag等于1且entropy_coding_reset_flag等于0,则以下适用。

[0263] 以TableStateIdxDS及TableMPSValDS作为输入调用CABAC剖

[0264] 析过程的同步过程。

[0265] 在算术译码引擎之初始化过程之后,调用终止之前的对二元决策

[0266] 的译码过程。

[0267] 否则,若tiles_or_entropy_coding_sync_idc等于2,且CtbAddrRS%PicWidthInCtbs等于0,则以下适用。

[0268] 当availableFlagT等于1时,以TableStateIdxWPP及TableMPSValWPP作为输入调用CABAC剖析过程的同步过程。

[0269] 在算术译码引擎之过程之后,调用终止之前的对二元决策的译码过程。

[0270] 当cabac_independent_flag等于0且dependent_slice_flag等于1时,或当tiles_or_entropy_coding_sync_idc等于2时,如下应用存储过程。

[0271] 当tiles_or_entropy_coding_sync_idc等于2且CtbAddrRS%PicWidthInCtbs等于2时,以TableStateIdxWPP及TableMPSValWPP作为输入调用CABAC剖析过程的存储过程。

[0272] 当cabac_independent_flag等于0、dependent_slice_flag等于1且end_of_slice_flag等于1时,以TableStateIdxDS及TableMPSValDS作为输入调用CABAC剖析过程的存储过程。

[0273] 语法元素的剖析如下进行:

[0274] 对于语法元素之每一所请求值,导出二元化。

[0275] 语法元素及所剖析频率区间序列之二元化判定译码过程流。

[0276] 对于语法元素之二元化的每一频率区间(其由变量binIdx索引),导出上下文索引ctxIdx。

[0277] 对于每一ctxIdx,调用算术译码过程。

[0278] 将所剖析频率区间之所得序列(b0..bbinIdx)与在每一频率区间之译码之后由二元化过程给出的频率区间串组进行比较。当该序列与给定组中之频率区间串匹配时,将对应值指派给语法元素。

[0279] 若处理对语法元素之值的请求以获得语法元素pcm标记,且pcm_flag的经译码值等于1,则在译码任何pcm_alignment_zero_bit、num_subsequent_pcm以及所有pcm_sample_luma及pcm_sample_chroma数据之后,初始化译码引擎。

[0280] 因此，以上描述揭示如图24中所示的译码器。此译码器大体上由参考符号5指示，此译码器自数据流12重构图像10，图像10以图像10所分割为的片14为单位写码至数据流12中，其中译码器5被配置为根据片次序16译码来自数据流12的片14。自然地，译码器5不限于串行译码片14。相反，译码器5可使用波前并行处理来译码片14，假定图像10分割为片14对于波前并行处理而言为合适。因此，译码器5可（例如）为能够通过考虑片次序16以便允许如上文已描述且下文还将描述之波前处理，来以错列方式与开始片14之译码并行地译码片14。

[0281] 译码器5响应片14中之当前片内的语法元素部分18，以便根据至少两个模式20及22中之一者来译码当前片。根据至少两种模式中的第一个，即模式20，使用包含越过片边界（还即，越过图24中的虚线）导出上下文的上下文适应熵译码，还即通过使用起源于其它“按片次序16在前片”之写码/译码的信息，来自数据流12译码当前片。另外，使用第一模式20自数据流12译码当前片包括连续更新编译码器的符号概率，以及在译码当前片的开始初始化符号概率，其取决于先前译码的片的符号概率的所保存状态。上文例如结合“编译码器变量之同步过程”描述此依赖性。最后，第一模式20还涉及越过片边界之预测性译码。此越过片边界之预测性译码可（例如）涉及越过片边界之帧内预测，即基于“按片次序16”的已经重构的样本值、在前片或越过片边界的写码参数的预测（例如运动向量、预测模式、写码模式等的预测），来预测当前片内的样本值。

[0282] 根据第二模式22，译码器5使用上下文适应熵译码来译码来自数据流12的当前片，还即当前将要译码的片，然而，将上下文的导出限制为不越过片边界。举例而言，若用于导出与当前片内之区块有关的某一语法元素的上下文的相邻位置的模板延伸至相邻片中，藉此越过当前片的片边界，则将相邻片之相应的部分的对应属性（例如相邻片之此相邻部分的对应语法元素的值）设定为默认值，以便抑制当前片与相邻片之间的相互依赖性。虽然上下文的符号概率的联系更新可发生，正如第一模式20中之情况一样，但第二模式22中之符号概率的初始化独立于任何先前译码的片。另外，通过将预测性译码限制为不越过片边界来执行预测性译码。

[0283] 为了容易理解图24的描述以及以下描述，参考图25，其展示出与图24相比更具结构意义的译码器5的可能实施方案。作为图24中的情况，译码器5为使用上下文适应熵译码来译码数据流以便获得（例如）预测残余及预测参数的预测性译码器。

[0284] 如图25中所示，译码器5可包括熵译码器24、解量化及逆变换模块26、组合器，其如图25中所示例如实施为加法器27与预测器28。熵译码器24、模块26及加法器27按其提及次序串联连接于译码器5之输入与输出之间，且预测器28连接于加法器27之输出与其另一输入之间，以便连同组合器和加法器27分别形成预测回路。因此，译码器24之输出额外连接至预测器28之写码参数输入。

[0285] 尽管图25提供译码器串行译码当前图像之印象，但译码器5可（例如）实施为并行译码图像10。译码器5可例如包括多个核，每一核根据图25中之组件24至28来操作。然而，串行处理为任择的，且串行操作之译码器5还能够译码在熵译码器24的输入处入站的数据流。

[0286] 为了高效地达成串行或并行译码当前图像10之刚刚提及之能力，译码器5以写码区块30为单位来操作以便译码图像10。写码区块30为例如叶区块，通过诸如四元树分割等递归多树分割来将写码树区块或最大写码区块32分割为页区块。代码树区块32又可规则地

布置成行及列，以便形成图像10分为此等代码树区块32的规则分割。在图25中，展示代码树区块32具有连续线，而展示写码区块30具有虚线。出于示出目的，仅展示一个代码树区块32将进一步分割为写码区块30，而代替地展示其它代码树区块32不进一步分割以便直接形成写码区块数据流12可包括关于如何将图像10分割为代码区块30的语法部分信令。

[0287] 数据流12为每一写码区块30传达语法元素，其揭露关于模块24至28将如何恢复该写码区块30内的图像内容。举例而言，此等语法元素包括：

[0288] 1)任选地，分割数据进一步将写码区块30分割为预测区块，

[0289] 2)任选地，分割数据进一步将写码区块30分割为残余及/或变换区块，

[0290] 3)关于将使用哪一预测模式来为写码区块30导出预测信号的预测模式信令，其中发信号此预测模式之粒度可取决于写码区块30及/或预测区块。

[0291] 4)可按写码区块或(若存在)按具有例如取决于预测模式而发送之一种预测参数的预测区块来发信号预测参数。可能的预测模式可例如包括帧内预测及/或帧间预测。

[0292] 5)还可存在其它语法元素，例如用于在写码区块30处对图像10进行滤波以便获得预测信号及/或将要再制之经重构信号的滤波信息。

[0293] 6)最后，尤其呈变换系数之形成的残余信息可包括于用于写码区块30的数据流中；以残余区块为单位，可发信号残余数据；按残余区块，若存在，则谱分解可例如以前面提及之变换区块为单位来执行。

[0294] 熵译码器24负责自数据流获得刚刚提及之语法元素。为此，熵译码器24使用上下文适应熵译码。即，熵译码器24提供若干上下文。为了自数据流12导出某一语法元素，熵译码器24在可能的上下文之中选择某一上下文。依据当前语法元素所属之图像10的部分的邻域的属性来执行可能上下文之中的选择。对于可能上下文中之每一者，熵译码器24管理符号概率，还即熵译码器24基于其操作的符号字母表的每一可能符号的机率估计。“管理”涉及上下文的符号概率的前面提及之连续更新，以便使与每一上下文相关联之符号概率适应实际图像内容。藉此，符号概率适于符号之实际机率统计。

[0295] 邻域之属性影响图像10之当前部分(例如当前写码区块30)之重构的另一情况为预测器28内的预测性译码。预测不仅限于当前写码区块30内的预测内容，而且可包含用于当前写码区块30的数据流12内所含的参数(例如预测参数、分割数据或甚至变换系数)的预测。还即，预测器28可自前面提及之邻域预测图像内容或此等参数，以便获得书面信号，其接着与如由模块26自数据流12获得的预测残余组合。在预测参数的情况下，预测器28可使用数据流内所含的语法元素作为预测残余来获得预测参数之实际值。预测器28使用后者预测参数值来获得刚刚提及之预测信号，其将在组合器中与预测残余组合。

[0296] 前面提及之“邻域”主要涵盖当前将要熵译码之语法元素或当前将要预测之语法元素所属之当前部分的周边的左上部。在图25中，在34处示例性地示出一个写码区块30的此邻域。

[0297] 在写码区块30之中界定写码/译码次序：在最粗糙等级处，以扫描次序36扫描图像10之代码树区块32，扫描次序36此处示出为自上而下逐列进行的光栅扫描。在每一代码树区块内，以深度第一遍历次序来扫描写码区块30，使得在每一层级中，还大体上以自上而下逐列进行的光栅扫描来扫描代码树区块32。

[0298] 写码区块30之中所界定之写码次序与用于导出邻域中之属性以便选择上下文且/

或执行空间预测的邻域34的界定一致，因为邻域34主要涵盖图像10之已经根据写码次序经受译码的部分。每当邻域34之一部分涵盖图像10之非可用部分时，例如替代地使用预设数据。举例而言，邻域模板34可延伸到图像10之外。然而，另一可能性为邻域34延伸至相邻片中。

[0299] 举例而言，片沿沿写码区块30界定之写码/译码次序来划分图像10，还即每一片为写码区块30沿前面提到之写码区块次序之连续非中断序列。在图25中，以点划线14指示片。在片14之中界定的次序由其如上文概述之顺序写码区块30之游程的成分产生。若某一片14之语法元素部分18指示其将以第一模式译码，则熵译码器24允许上下文适应熵译码来越过片边界导出上下文。还即，使用空间邻域34，以便在关于当前片14之熵译码数据中选择上下文。在图25之情况下，例如片编号3可为当前译码的片，且在与写码区块30或其中所含之某一部分有关的熵译码语法元素中，熵译码器24可使用起源于例如片编号1等相邻片内之译码部分的属性。预测器28表现相同：对于正为第一模式20的片，预测器28越过环绕当前片的片边界使用空间预测。

[0300] 然而，对于具有与之相关(还即，语法元素部分18指示第二模式22)的第二模式22的片，熵译码器24及预测器28将熵上下文的导出及预测性译码限制为取决于与位于仅当前片内之部分有关的属性。显然，写码效率遭受此限制。另一方面，第二模式22的片允许瓦解该序列的片之间的相互依赖性。因此，第二模式22的片可散布在图像10内或图像10所述之视频内，以便允许再同步点。然而，每一图像10在第二模式22下具有至少一个片系不必要的。

[0301] 如上文已经提及，第一及第二模式20及22还在其符号概率之初始化方面不同。第二模式22中写码的片导致熵译码器24重新初始化机率，而与任何先前译码的片(还即先前在片之中界定之次序的意义上译码)无关。举例而言，将符号概率设定为编码器及译码器侧均已知的默认值，或初始化值包含于在第二模式22下写码的片内。

[0302] 还即，对于正在第二模式22下写码/译码的片，符号概率之适应总是自此等片之开头立即开始。因此，适应准确度在此等片之开头对此等片而言较差。

[0303] 在第一模式20下写码/译码的片中，情况不同。对于后者片，由熵译码器24执行之符号概率的初始化取决于先前译码的片的符号概率的所保存状态。每当在第一模式20下写码/译码的片的开头例如并非位于图像10之左手侧，还即不在光栅扫描36从其开始逐列运行之后向底部进行至下一列的侧时，采用在熵译码紧接在前片结束时所得的符号概率。举例而言，此在图25中由4号片之箭头38示出。4号片之开头在图像10之右手侧与左手侧之间的某处，因此，在初始化符号概率时，熵译码器24在初始化符号概率时采用在熵译码紧接在前片(还即，3号片)时所获得之符号概率，直至其结束为止，还即在片3之熵译码期间包含符号概率之连续更新，直至其结束为止。

[0304] 具有与之相关联的第二模式22的片(然而，其开头位于图像10之左手侧，例如5号片)不适应如在结束紧接在前之4号片的熵译码之后获得的符号概率，因为此将阻止译码器5通过使用波前处理来并行译码图像10。相反，如上文所概述，熵译码器24适应如在结束如由箭头40示出熵译码紧接在前(以编码/译码次序36)代码树区块列中之第二(以编码/译码次序36)代码树区块32之后获得的符号概率。

[0305] 在图25中，例如，图像10示例性地分割为三列代码树区块以及四行写码树根区块

32，且每一代码树区块列再分为两个片14，使得每第二个片之开头与相应的代码树根区块列之写码单元次序中的第一写码单元重合。因此，熵译码器24将能够通过并行译码每一代码树跟区块列、通过开始以错列方式译码此等代码树跟区块列、以第一或最上代码树跟区块列、接着第二且接着第三来在译码图像10时使用波前处理。

[0306] 自然，以递归方式将区块32分割为进一步的写码区块30为任择的，且因此在较一般意义上，区块32还可称为“写码区块”。还即，较一般而言，可将图像10分割为布置成列及行且具有彼此之间界定之光栅扫描次序36的写码区块32，且可将译码器5视为使每一片14与光栅扫描次序36中之写码区块32之连续子集相关联，使得该子集根据片次序沿光栅扫描次序36彼此相随。

[0307] 如自以上论述还清楚，译码器5或更具体而言熵译码器24可被配置为保存如在根据光栅扫描次序36上下文适应熵译码任何片直至写码区块列中之第二写码区块时所获得之符号概率。在为上下文适应熵译码具有与之相关的第一模式20之当前片初始化符号概率时，译码器5或更具体而言熵译码器24根据光栅扫描次序36检查与当前片相关联之写码区块32之连续子集的第一写码区块32是否为写码区块列中之第一写码区块32。如果是，则如相对于箭头40所阐释来初始化当前片之上下文适应熵译码之符号概率，还即取决于如在根据光栅扫描次序36上下文熵译码先前译码的片直至写码区块列中之第二写码区块时所获得的所保存符号概率。如果不是，则依据如在上下文适应熵译码先前译码的片直至先前译码的片之结尾(还即，根据箭头38)时所获得之符号概率来执行为当前片之上下文适应熵译码初始化符号概率。并且，在根据38初始化的情况下，熵译码片次序36中之紧接在前片之结束时的所保存状态被表示，而在初始化40之情况下，先前译码的片包括区块次序36中之紧接在前区块32列之第二区块的结尾。

[0308] 如由图24中之虚线所示出，译码器可被配置为响应片14中之当前片内的语法元素部分18，以便根据至少三种模式中的一者译码当前片。还即，除其它模式20及22之外，可存在第三模式42。第三模式42与第二模式22之不同之处可在于允许越过片边界之预测，而熵写码/译码仍被限制为不越过片边界。

[0309] 上文呈现关于语法元素部分18的两个实施例。下表概述此等两个实施例。

[0310]

	实施例 1	实施例 2
语法元素部分	dependent_slice_flag，no-_cabac_reset_flag	dependent_slice_flag
模式 1	dependent_slice_flag = 1，no-_cabac_reset_flag = 1	dependent_slice_flag = 1，
模式 2	dependent_slice_flag = 0	dependent_slice_flag = 0
模式 3	dependent_slice_flag = 1，no-_cabac_reset_flag = 0	dependent_slice_flag = 1，cabac_independent_flag = 1，tiles_or_entropy_coding_sync_idc = 3

[0311] 在该实施例中，语法元素部分18由dependent_slice_flag相应的形成，而在另一实施例中，dependent_slice_flag与no_cabac_reset_flag之组合形成语法元素部分。参考上下文变量之同步过程，直至关系到取决于先前译码的片的符号概率的所保存状态来初始

化符号概率。明确而言，译码器可被配置为在last_ctb_cabac_init_flag＝0且tiles_or_entropy_coding_sync_idc＝2之情况下，保存如在根据光栅扫描次序上下文适应熵译码先前译码的片直至列中之第二写码区块时且在为根据第一模式为当前片之上下文适应熵译码初始化符号概率时所获得的符号概率，检查与当前片相关联之写码区块之连续子集的第一写码区块是否为根据光栅扫描次序之列中的第一写码区块，且如果是，依据如在根据光栅扫描次序上下文适应熵译码先前译码的片直至列中之第二写码区块时所获得之所保存符号概率为当前片之上下文适应熵译码初始化符号概率，且如果不是，则依据如在上下文适应熵译码先前译码的片直至先前译码的片之结尾时所获得之符号概率来为当前片之上下文适应熵译码初始化符号概率。

[0312] 因此，换言之，根据语法之第二实施例，译码器将自图像以图像(10)所分割为的片14为单位写码至的数据流12重构图像10，其中译码器被配置为根据片次序16自数据流12译码片14，且译码器响应语法元素部分18，还即该片中之当前片内的dependent_slice_flag，以便根据至少两个模式20、22中之一者译码当前片。根据至少两个模式中的第一个20，还即若dependent_slice_flag＝1，则译码器使用上下文适应熵译码24来自数据流12译码当前片，上下文适应熵译码24包含越过片边界之上下文的导出、上下文之符号概率的连续更新，以及依据先前译码的片的符号概率的所保存状态对符号概率之初始化38、40，以及越过片边界的预测性译码，且根据至少两种模式中的第二个22，还即dependent_slice_flag＝0，译码器使用：上下文适应熵译码及预测性译码来自数据流12译码当前片，该上下文适应熵译码具有将上下文的导出限制为不越过片边界、上下文之符号概率的连续更新，以及与任何先前译码片无关之符号概率的初始化，且该预测性译码具有将预测性译码限制为不越过片边界。可将图像10分割为布置成列及行且具有彼此之间界定之光栅扫描次序36的写码区块32，且译码器被配置为使每一片14与光栅扫描次序36中之写码区块32之连续子集相关联，使得该子集根据片次序沿光栅扫描次序36彼此相随。译码器可被配置为还即响应于tiles_or_entropy_coding_sync_idc＝2之情况下，保存如在根据光栅扫描次序36上下文适应熵译码先前译码的片直至列中之第二写码区块32时且在为根据第一模式为当前片之上下文适应熵译码初始化符号概率时所获得的符号概率，检查与当前片相关联之写码区块32之连续子集的第一写码区块是否为根据光栅扫描次序之列中的第一写码区块32，且如果是，依据如在根据光栅扫描次序36上下文适应熵译码先前译码的片直至列中之第二写码区块时所获得之所保存符号概率为当前片之上下文适应熵译码初始化40符号概率，且如果不是，则依据如在上下文适应熵译码先前译码的片直至先前译码的片之结尾时所获得之符号概率来为当前片之上下文适应熵译码初始化38符号概率。译码器可被配置为响应片14中之当前片内的语法元素部分(18)，以便根据至少三种模式中的一者(还即第一模式20及第三模式42或第二模式22中的一者)译码当前片，其中译码器被配置为根据第三模式42，还即在dependent_slice_flag＝1且tiles_or_entropy_coding_sync_idc＝3的情况下，使用上下文适应熵译码来自数据流译码当前片，该上下文适应熵译码具有将上下文的导出限制为不越过片边界，上下文之符号概率的连续更新，以及与任何先前译码片无关之符号概率的初始化，以及越过片边界之预测性译码，其中依据语法元素(还即cabac_independent_flag)选择第一及第三模式中的一者。译码器可进一步被配置为，还即在tiles_or_entropy_coding_sync_idc＝0、1及3(当cabac_independent flag＝0时为“3”)的情况下，保存如在

上下文适应熵译码先前译码的片直至先前译码的片的结尾时以及根据第一模式为当前片之上下文适应熵译码初始化符号概率时所获得的符号概率，依据所保存的符号概率为当前片的上下文适应熵译码初始化符号概率。译码器可被配置为，还即在tiles_or_entropy_coding_sync_idc＝1的情况下，在第一及第二模式下，将预测性译码限制于图像再分为的瓦片内。

[0313] 自然地，编码器能够相应的地设定上文所呈现的语法，以便使译码器能够获得上文概述的优势。编码器可为并行处理(例如多核)编码器，但无需为并行处理(例如多核)编码器。对于以片14为单位将图像10编码至数据流12中，编码器将被配置为根据片次序16将片14编码至数据流12中。编码器将为片中之当前片判定语法元素部分18，并将其写码至当前片中，使得该语法元素部分发信号将要根据至少两种模式20、22中之一者写码的当前片，且若将根据至少两种模式中的第一个20来写码当前片，则使用上下文适应熵编码24来将当前片编码至数据流12中，上下文适应熵编码24包含越过片边界之上下文的导出、上下文之符号概率的连续更新及取决于先前编码的片的符号概率的所保存状态对符号概率之初始化38、40，以及越过片边界之预测性编码，且若将根据至少两种模式中的第二个22来写码当前片，则使用上下文适应熵编码以及预测性编码来将当前片编码至数据流12中，该上下文适应熵编码具有将上下文的导出限制为不越过片边界，上下文之符号概率的连续更新及与任何先前编码的片无关的符号概率的初始化，且该预测性编码具有将预测性编码限制为不越过片边界。虽然可将图像10分割为布置成列及行且具有彼此之间界定之光栅扫描次序36的写码区块32，但编码器可被配置为使每一片14与光栅扫描次序36中之写码区块32之连续子集相关联，使得该子集根据片次序沿光栅扫描次序36彼此相随。编码器可被配置为保存如在根据光栅扫描次序36上下文适应熵译码先前编码的片直至列中之第二写码区块32时且在为根据第一模式为当前片之上下文适应熵编码初始化符号概率时所获得的符号概率，检查与当前片相关联之写码区块32之连续子集的第一写码区块是否为根据光栅扫描次序之列中的第一写码区块32，且如果是，依据如在根据光栅扫描次序36上下文适应熵编码先前编码的片直至列中之第二写码区块时所获得之所保存符号概率为当前片之上下文适应熵译码初始化40符号概率，且如果不是，则依据如在上下文适应熵编码先前编码的片直至先前编码的片之结尾时所获得之符号概率来为当前片之上下文适应熵译码初始化38符号概率。编码器可被配置为将语法元素部分(18)写码至片(14)中之当前片中，使得发信号给当前片以根据至少三种模式中的一者(还即第一模式(20)及第三模式(42)或第二模式(22)中的一者)写码至其中，其中编码器被配置为根据第三模式(42)，使用上下文适应熵编码来将当前片编码至数据流中，该上下文适应熵编码具有将上下文的导出限制为不越过片边界，上下文之符号概率的连续更新及与任何先前译码片无关之符号概率的初始化，以及越过片边界之预测性写码，其中编码器在第一及第三模式中之一者之间的差别为使用语法元素，例如还即cabac_independent_flag。编码器可被配置为判定一般语法元素(例如dependent_slices_present_flag)，并通过依据该一般语法元素在至少两种一般操作模式中之一者下操作来将其写入至数据流中，还即，根据第一一般操作模式，执行写码每一片之语法元素部分，且根据第二一般操作模式，必然地使用至少两种模式中不同于第一模式的不同模式。编码器可被配置为根据第一及第二模式，自当前片之开头至结尾，必然且不间断地继续连续更新符号概率。编码器可被配置为保存如在上下文适应熵编码先前编码的片直

至先前编码的片的结尾时以及根据第一模式为当前片之上下文适应熵编码初始化符号概率时所获得的符号概率，依据所保存之符号概率为当前片之上下文适应熵编码初始化符号概率。并且，在第一及第二模式下，编码器可将预测性编码限制于图像再分为的瓦片内。

[0314] 为了完整起见，图26中描绘编码器之可能结构。预测器70几乎与预测器28操作相同，还即执行预测，但还通过例如最佳化来判定包含预测参数及模式的写码参数。模块26及27还出现于译码器中。减法器72判定无损预测残余，其接着通过使用量化且任选地使用谱分解变换，在变换及量化模块74中有损写码。熵写码器76执行上下文适应熵编码。

[0315] 除以上具体语法实例之外，下文通过呈现后文使用之术语与上文所使用之术语之间的一致性来概述不同实例。

[0316] 明确而言，上文尚未明确概述，从属片不仅为“从属”的，因为其允许利用自其边界外部已知的知识，例如如上文所概述，具有较快适应的熵上下文，或归因于允许越过其边界而达成优选的空间预测。相反，为了节约通过将图像分为若干片来界定片报头所必须花费之速率成本，从属片采用来自先前片的片报头语法的一部分，还即此词组法报头部分不会为了从属片而再次发射。此例如在图16中在100处且在图21中在102处展示，根据此，例如采用来自先前片的片类型。藉此，将图像再分为若干片(例如独立片及从属片)在位消耗昂贵方面较便宜。

[0317] 在下文概述之实例中，刚刚提及之依赖性导致稍微不同的措辞：将片定义为图像之可个别设定片报头语法的单位部分。因此，片由一个(使用上文之命名法)独立/规则/正常片(现称为独立片区段)且无一或多个(使用上文之命名法)从属片(现称为从属片区段)组成。

[0318] 举例而言，图27展示出将要分割为两个片之图像，一个片由片区段14_1至14_3形成，且另一仅由片区段14_4形成。索引1至4以写码次序展示片次序。图28a及图28b展示出在将图像10再分为两个瓦片之情况下的不同实例，其中在图28a之情况下，一个片由全部五个片区段14(涵盖50_1及50_2两者)形成，索引在此以写码次序升高，且在图28a之情况下，两个片分别由再分瓦片50_1的片区段14_1及14_2及14_3及14_4形成，且另一片由涵盖瓦片50_2的片区段14_5至14_6形成。

[0319] 定义可为如下：

[0320] 从属片区段：片区段报头之一些语法元素的值系自在译码次序中在前独立片区段(在上文实施例中，先前称为从属片)的值推断的片区段。

[0321] 独立片区段：片区段报头之语法元素的值并非自在前片区段(在上文实施例中，先前称为正常片)的值推断的片区段。

[0322] 片：在同一存取单元/图像内之一个独立片区段及下一独立片区段(若存在)之前的所有后续从属片区段(若存在)中所含有的整数数目个写码树单元。

[0323] 片报头：作为当前片区段或作为在当前从属片区段之前的独立片区段的独立片区段的片区段报头。

[0324] 片区段：在瓦片扫描中连续排序且包含于单一NAL单元中的整数数目个写码树单元；将每一图像分为若干片区段为分区。

[0325] 片区段报头：经写码片区段之含有与片区段中所表示的第一或所有写码树单元有关的数据元素的部分。

[0326] “模式”20及22之发信号，还即“从属片区段”及“独立片区段”可为如下：

[0327] 在例如PPS等一些额外NAL单元中，可使用语法元素来发信号是否对甚至某些图像之序列中的某一图像使用从属片：

[0328] 等于1的dependent_slice_segments_enabled_flag指定片区段报头中语法元素dependent_slice_segment_flag的存在。等于0的dependent_slice_segments_enabled_flag指定片区段报头中语法元素dependent_slice_segment_flag的不存在。

[0329] dependent_slice_segments_enabled_flag在范围上类似于先前描述之dependent_slices_present_flag。

[0330] 类似地，dependent_slice_flag可称为dependent_slice_segment_flag，以便考虑相对于片的不同命名法。

[0331] 等于1的dependent_slice_segment_flag指定不存在于当前片区段之报头中的每一片区段报头语法元素的值被推断为等于在前独立片区段的片报头(还即片区段报头)中之对应片区段报头语法元素的值。

[0332] 在同一等级(例如图像等级)中，可包含以下语法元素：

[0333] 等于1的entropy_coding_sync_enabled_flag指定在译码包含涉及PPS之每一图像中之每一瓦片中之一列写码树区块中之第一写码树区块的写码树单元之前，调用对上下文变量的特定同步过程，且在译码包含涉及PPS之每一图像中之每一瓦片中之一列写码树区块中之第二写码树区块的写码树单元之后，调用对上下文变量之特定储存过程。等于0的entropy_coding_sync_enabled_flag指定在译码包含涉及PPS之每一图像中之每一瓦片中之一列写码树区块中之第一写码树区块的写码树单元之前，无需调用对上下文变量之特定同步过程，且在译码包含涉及PPS之每一图像中之每一瓦片中之一列写码树区块中之第二写码树区块的写码树单元之后，无需调用对上下文变化之特定储存过程。

[0334] 位流一致性的要求为entropy_coding_sync_enabled_flag的值对于在CVS内启动之所有PPS均将相同。

[0335] 当entropy_coding_sync_enabled_flag等于1且片中之第一写码树区块并非瓦片中之一列写码树区块中之第一写码树区块，则位流一致性的要求为片中之最后一个写码树区块将属于与片中之第一写码树区块相同的写码树区块列。

[0336] 当entropy_coding_sync_enabled_flag等于1且片区段中之第一写码树区块并非瓦片中之一列写码树区块中之第一写码树区块，则位流一致性的要求为片区段中之最后一个写码树区块将属于与片区段中之第一写码树区块相同的写码树区块列。

[0337] 如已经描述，CTB 30之中的写码/译码次序以光栅方式逐列自上而下，以扫描第一瓦片开始，接着访问下一瓦片，若图像中存在一个以上瓦片。

[0338] 译码器5且因此编码器在熵译码(写码)图像的片区段14时如下作用：

[0339] A1)每当当前译码/写码之语法元素synEl为瓦片50、片区段14或CTB列之第一语法元素时，开始图29之初始化过程。

[0340] A2)否则，此语法元素之译码使用当前熵上下文发生。

[0341] A3)若当前语法元素为CTB 30中的最后一个语法元素，则开始如图30中所示之熵上下文储存过程。

[0342] A4)该过程在A1)处以下一语法元素继续。

[0343] 在初始化过程中，检查(200)synEI是否为片区段14或瓦片50之第一语法元素。如果是，则在步骤202中，独立于任何先前片区段初始化上下文。如果不是，则检查(204)synEI是否为CTB 30之列的第一语法元素，且entropy_coding_sync_enabled_flag是否等于一。如果是，则检查(206)在相等瓦片之前一列CTB 30中，第二CTB 30是否可用(见图23)。若可用，则在步骤210中，使用针对类型40采用的当前储存之上下文机率来执行根据40之上下文采用。若不可用，则在步骤202中，独立于任何先前片区段初始化上下文。若检查204揭示否，则在步骤212中检查synEl是否为从属片区段14之第一CTB中的第一语法元素，且dependent_slice_segement_flag是否等于一，且如果是，则在步骤214中，使用针对类型38采用的当前所储存上下文机率，来执行根据38之上下文采用。在步骤214、212、210及202中的任一者之后，实际上开始译码/写码。

[0344] 具有等于一的dependent_slice_segement_flag的从属片区段因此有助于在几乎不具有写码效率惩罚之情况下进一步减小写码/译码延迟。

[0345] 在图30之储存过程中，在步骤300中，检查经写码/译码synEl是否为一列CTB 30之第二CTB 30之最后一个语法元素，且entropy_coding_sync_enabled_flag是否等于一。如果是，在步骤302中将当前熵上下文(还即上下文之熵写码机率)储存于特别用于按照40之采用的储存装置中。类似地，除步骤300或302之外，在步骤304中，检查经写码/译码synEl是否为片区段14之最后一个语法元素，且dependent_slice_segement_flag是否等于一。如果是，则在步骤306中将当前熵上下文(还即，上下文之熵写码机率)储存于特定用于按照38之采用的储存装置中。

[0346] 注意，查询语法元素是否为CTB列之第一synEl的任何检查例如利用片区段之报头内的语法元素slice_adress 400，还即揭示相应的片区段之开始沿译码次序的位置的开始语法元素。

[0347] 在使用WPP处理自数据流12重构图像10时，译码器能够恰好利用后者开始语法部分400来撷取WPP子流进入点。由于每一片区段包括一指示相应的片区段之译码开始在图像10内的位置的开始语法部分400，因此译码器能够通过使用片区段之开始语法部分400识别在图像之左手侧开始的片区段来识别片区段分组为的WPP子流的进入点。译码器可接着以与根据片次序顺序开始WPP子流之译码错列的方式并行地译码WPP子流。片区段可甚至小于一个图像宽度，以及一列CTB，使得其发射可在WPP子流之中交错，来进一步减小总的发射端至端延迟。编码器提供具有指示相应的片之写码开始在图像(10)内的位置的开始语法部分(400)的每一片(14)，且将该片分组为若干WPP子流，使得对于每一WPP子流，片次序中的第一片在图像之左手侧开始。编码器可甚至在编码图像时自行使用WPP处理：编码器可以与根据片次序顺序开始WPP子流之编码错列的方式并行地编码WPP子流。

[0348] 藉此，使用片区段之开始语法部分作为用于定位WPP子流之进入点的构件的后者方面可在无从属片概念之情况下使用。

[0349] 通过如下设定以上变量，对于并行处理图像10而言均将为可行的：

[0350]

	tiles_enabled_flag==0	tiles_enabled_flag==1	entropy_coding_sync_enable_flag ==1	tiles_enabled_flag==1	entropy_coding_sync_enable_flag ==1
dependent_slice_segment_flag	0/1	0/1	0/1	0/1	0/1
图像再分为瓦片	否	是	否	是	否
以进入点再分片，以指示 WPP 子流或瓦片之开始。	否	是	是	否	否
并行处理可能	以片为单位	以瓦片为单位	WPP	以瓦片为单位	WPP

[0351] 将WPP与瓦片分割混合将更加可行。在此情况下，可将瓦片视为个别图像：使用WPP之每一瓦片将由具有一或多个从属片区段的片组成，且步骤300及208中之检查将涉及同一瓦片中之上方CTB列中的第二CTB，正如步骤204将涉及当前瓦片之CTB 30列中之第一CTB 30!在此情况下，上表可扩展：

[0352]

	tiles_enabled_flag	tiles_enabled_flag==1	entropy_cooding_sync_enable_flag ==1	tiles_enabled_flag==1	entropy_cooding_sync_enable_flag ==1	entropy_cooding_sync_enable_flag ==1	entropy_cooding_sync_enable_flag ==1
dependent_slice_segment_flag	0/1	0/1	0/1	0/1	0/1	0/1	0/1
图像再分为瓦片	否	是	否	是	否	是	是
以进入点再分片，以指示 WPP 子流或瓦片之开始。	否	是	是	否	否	否(其自己的从属片中之瓦片的每一子流	是 (单独片中的每一瓦片以及瓦片的片内的 WPP 子流的进入点
并行处理可能	以片为单位	以瓦片为单位	WPP	以瓦片为单位	WPP	瓦片内的 WPP 子流	瓦片内的 WPP 子流

[0353] 作为简要介绍，后者扩展对于实施例2已将为可能。实施例2允许以下处理：

[0354]

	语法等级				
tiles_or_entropy_coding_sync_idc	每图像	0	1	2	3
cabac_independent_flag	每片	不存在	不存在	不存在	0/1
dependent_slice_flag	每片	0/1	0/1	0/1	0/1
图像再分为瓦片		否	是	否	否
以进入点再分片，以指示 WPP 子流或瓦片之开始。		否	是	是	否
并行处理可能		以片为单位	以瓦片为单位	WPP	片之熵写码

[0355] 但对于以下扩展，将得出下表：

[0356] 添加至图像参数组之语义：

[0357] 若tiles_or_entropy_coding_sync_idc等于4，除第一列CTB之外的每一CTB均将包含于从属片标记设定为1的不同片中。不同列之CTB不得存在于同一片中。每CTB列可存在一个以上片。

[0358] 若tiles_or_entropy_coding_sync_idc等于5，则除第一瓦片之外的每一瓦片之CTB必须包含于不同片中。不同瓦片之CTB不得存在于同一片中。每瓦片可存在一个以上片。

[0359] 见图31，进行进一步阐释。

[0360] 还即，上表可扩展：

[0361]

	语法等级								
tiles_or_entropy_coding_sync_idc	每图像	0	1	2	3	5	4	6	7
cabac_independent_flag	每片	不存在	不存在	不存在	0/1	不存在	不存在	不存在	不存在
dependent_slice_flag	每片	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
图像再分为瓦片		否	是	否	否	是	否	是	是
以进入点再分片，以指示 WPP 子流或瓦片之开始。		否	是	是	否	否	否	否(其自己的从属片中之瓦片的每一子流	是(单独片中的每一瓦片以及瓦片的片内的 WPP 子流的进入点
并行处理可能		以片为单位	以瓦片为单位	WPP	片之熵写码	以瓦片为单位	WPP	瓦片内的 WPP 子流	瓦片内的 WPP 子流

[0362] 关于以上实施例，应注意，译码器可被配置为例如响应于tiles_or_entropy_

coding_sync_idc＝1、2，在第一及第二模式下，自当前片读取揭示将当前片再分为并行子部分的信息，其中并行子部分可为WPP子流或瓦片；在第一并行子部分之结尾处停止上下文适应熵译码；且在任一后续并行子部分之开头处重新恢复上下文适应熵译码，其包含：在第一模式下，依据在前并行子部分之符号概率的所保存状态来初始化符号概率；且在第二模式下，独立于任何先前译码的片及任何先前译码之并行子部分来初始化符号概率。

[0363] 因此，以上描述揭示了用于如由新HEVC写码标准提供之结构化视频数据(例如以瓦片结构化)、波前并行处理(WPP)子流、片或熵片的低延迟编码、译码、封装及发射的方法。

[0364] 尤其已界定如何在对话场景中输送经并行编码之数据以便获得编码、译码及发射过程中的最小等待时间。因此，已描述了一种管线化并行写码、发射及译码方法，以便允许如游戏、远程手术等最小延迟应用。

[0365] 此外，以上实施例缩小了波前并行处理(WPP)之差距，来使其可在低延迟发射场景中使用。因此，已呈现用于WPP子流[2]的新封装格式，从属片。此从属片可含有熵片数据、WPP子流、一整列LCU、片的仅一片段，其中先前发射的片报头还适用于所含有的片段数据。在子片报头中发信号所包含之数据。

[0366] 最后注意，新片之命名还可为“子集/轻量片”，但已发现名称“从属片”为优选。

[0367] 已呈现信令，其描述写码与输送中的并行化等级。

[0368] 尽管在设备之上下文中已描述了一些方面，但清楚的是，此等方面还表示对应方法之描述，其中一区块或装置对应于一方法步骤或一方法步骤之一特征。类似地，方法步骤之上下文中所描述之方面还表示对应设备之对应区块或项目或特征的描述。一些或所有方法步骤可由(或使用)硬设备来执行，例如微处理器、可规划计算机或电子电路。在一些实施例中，最重要方法步骤中之一或多者可由此设备执行。

[0369] 取决于某些实施要求，本发明之实施例可在硬件中或软件中实施。可使用上面储存有电子可读控制信号之数字储存媒体(例如，软磁盘、DVD、蓝光、CD、ROM、PROM、EPROM、EEPROM或闪存)来执行该实施方案，该数字储存媒体与可规划计算机系统协作(或能够与之协作)，使得执行相应的方法。因此，数字储存媒体可为计算机可读的。

[0370] 根据本发明之一些实施例包括具有电子可读控制信号的数据载体，其能够与可规划计算机系统协作，使得执行本文所述方法中之一者。

[0371] 通常，本发明之实施例可实施为具有程序代码之计算机程序产品，当该计算机程序产品在计算机上运行时，该程序代码操作以用于执行该方法中之一者。该程序代码可例如储存于机器可读载体上。

[0372] 其它实施例包括储存于机器可读载体上之用于执行本文所述方法中之一者的计算机程序。

[0373] 换言之，本发明之方法的实施例因此为具有程序代码之计算机程序，当该计算机程序在计算机上运行时，该程序代码用于执行本文所述之方法中的一者。

[0374] 本发明之方法的另一实施例因此为数据载体(或数字储存媒体，或计算机可读媒体)，其上面记录有用于执行本文所述方法中之一者的计算机程序。该数据载体、该数字储存媒体或该所记录媒体通常为有形且/或非暂时的。

[0375] 本发明之方法的另一实施例因此为表示用于执行本文所述方法中之一者的计算机程序的数据流或信号序列。该数据流或信号序列可例如被配置为经由数据通信连接(例

如经由因特网)传送。

[0376] 另一实施例包括一种处理构件,例如计算机或可规划逻辑装置,其被配置为或适于执行本文所述方法中的一者。

[0377] 另一实施例包括一种计算机,其上面安装有用于执行本文所述方法中之一者的计算机程序。

[0378] 根据本发明之另一实施例包括一种设备或一种系统,其被配置为传送(例如,以电子方式或光学方式)用于执行本文所述方法中之一者的计算机程序至接收器。该接收可例如为计算机、行动装置、内存装置等。该设备或系统可例如包括用于将计算机程序传送至接收器的档案服务器。

[0379] 在一些实施例中,一种可规划逻辑装置(例如,现场可规划门数组)可用以执行本文所述方法之功能性中的一些或全部。在一些实施例中,现场可规划门数组可与微处理器协作,以便执行本文所述方法中之一者。通常,该方法优选由任何硬设备执行。

[0380] 上文所述之实施例仅示出本发明之原理。应理解,熟习此项技术者将明白对本文所述布置及细节的修改及变化。因此,本发明之意图仅由随附之权利要求之范畴限制,且不受藉助于描述及阐释本文之实施例而呈现的具体细节限制。

[0381] 参考文献

[0382] [1]托马斯·维甘德(Thomas Wiegand)、加里J·沙利文(Gary J.Sullivan)、吉赛尔.琼特高(Gisle Bjontegaard)、阿贾伊.鲁思拉(Ajay Luthra),“H.264/AVC视频写码标准综述(Overview of the H.264/AVC Video Coding Standard)”,IEEE电路系统视频技术学报,第N7期,第13卷,2003年7月。

[0383] [2]JCT-VC,“高效视频写码(HEVC)本文规范工作草案6(High-Efficiency Video Coding(HEVC)text specification Working Draft 6)”,JCTVC-H1003,2012年2月。

[0384] [3]ISO/IEC 13818-1:MPEG-2系统规范。

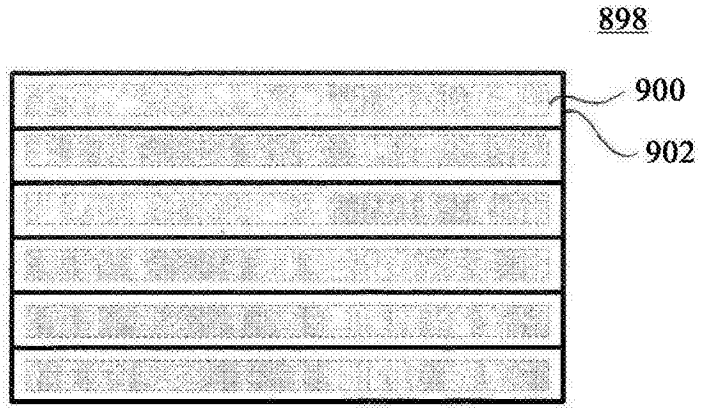

898
900
902

图1

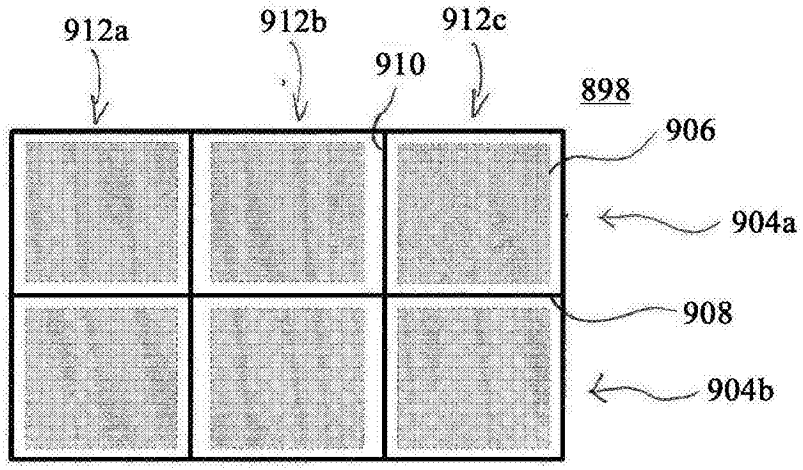

912a
912b
912c
910
898
906
904a
908
904b

图2

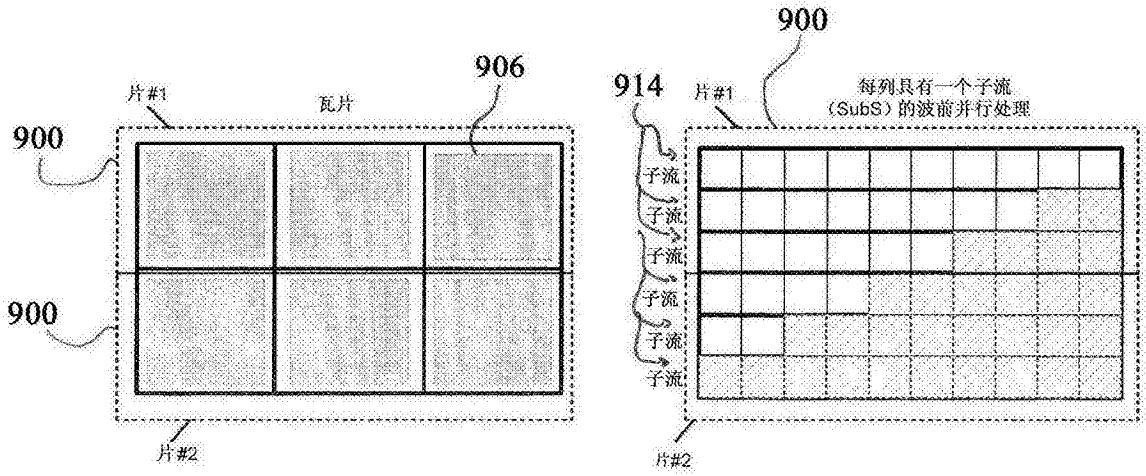

900
906
片#1
瓦片
900
900
片#2
914
片#1
每列具有一个子流
(SubS)的波前并行处理
子流
子流
子流
子流
子流
子流
片#2

图3

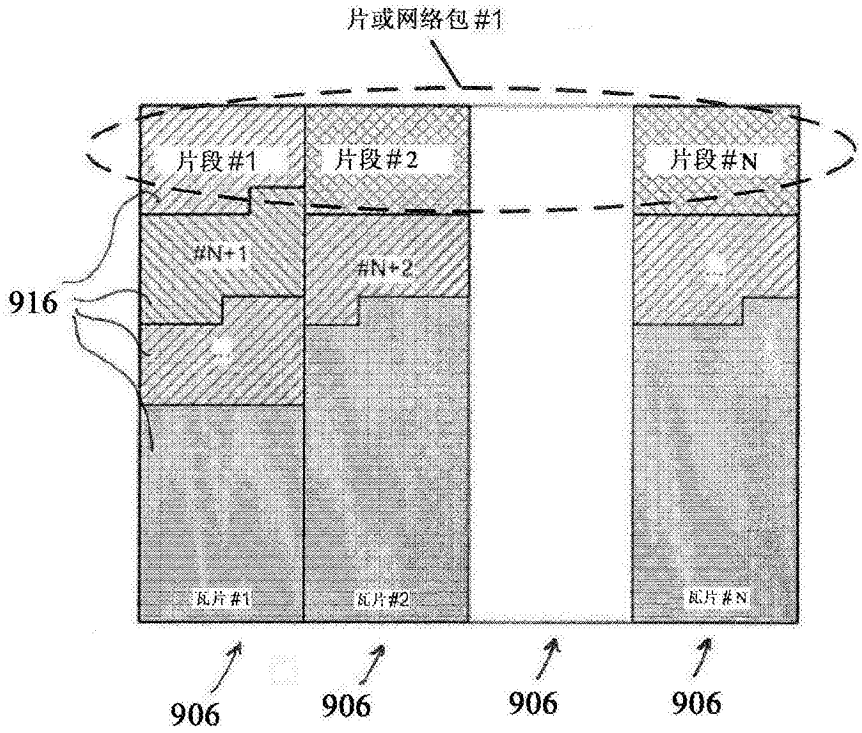

片或网络包#1
片段#1
片段#2
片段#N
#N+1
#N+2
916
瓦片#1
瓦片#2
瓦片#N
906
906
906
906

图4

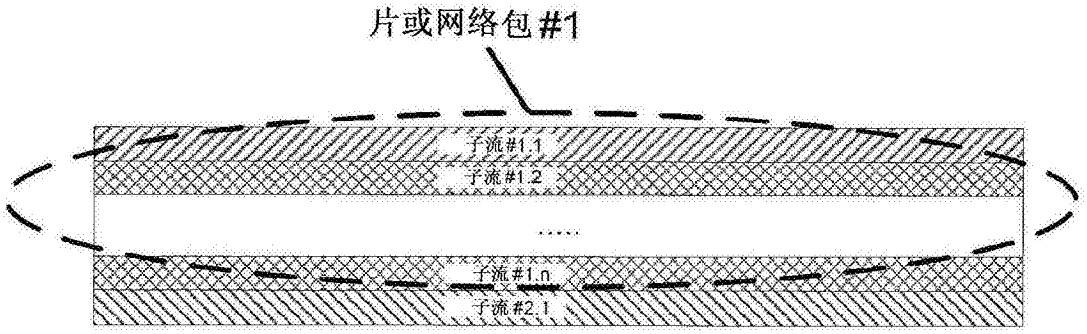

片或网络包#1
子流#1.1
子流#1.2
.....
子流#1.n
子流#2.1

图5

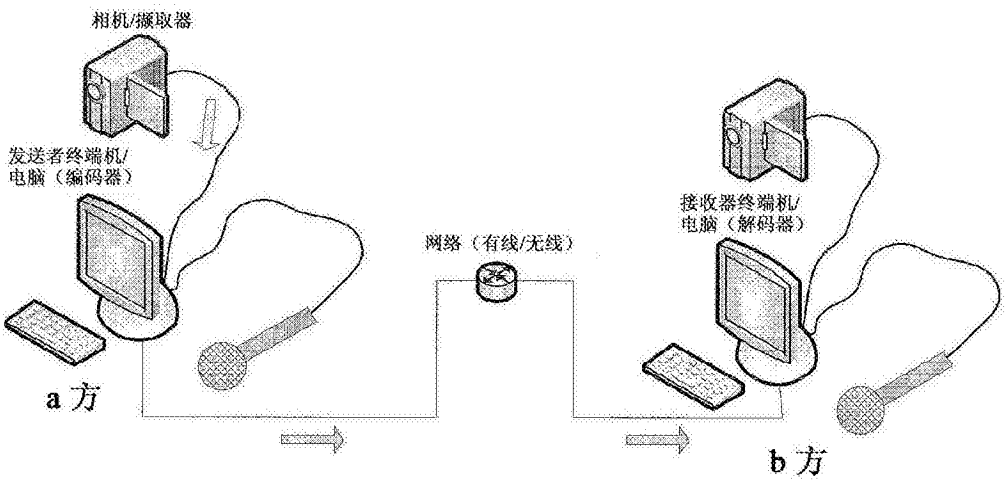

相机/摄取器
发送者终端机/
电脑（编码器）
网络（有线/无线）
接收器终端机/
电脑（解码器）
a 方
b 方

图6

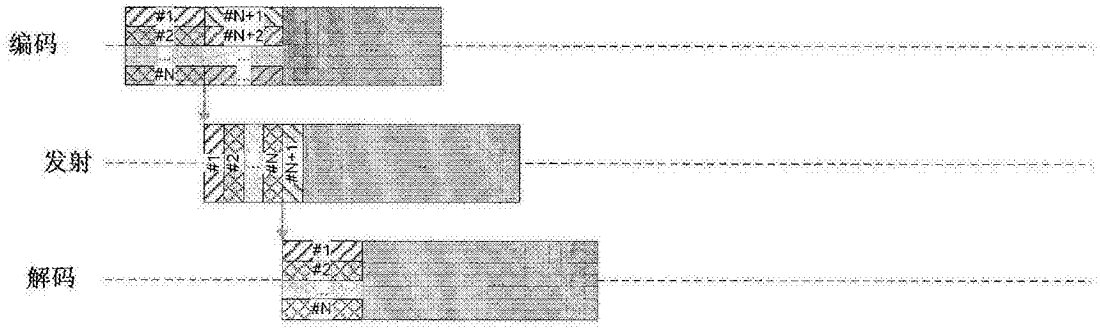

编码
发射
解码
#1
#2
#N
#N+1
#N+2

图7

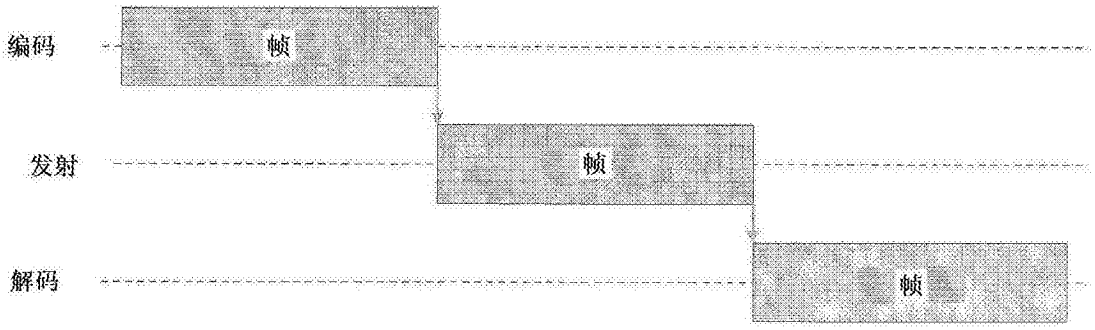

编码
发射
解码
帧
帧
帧

图8

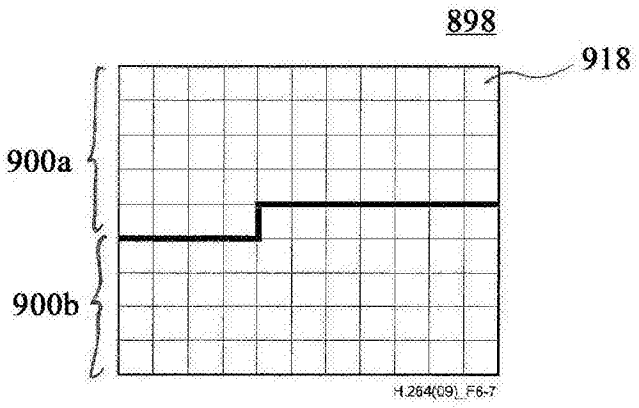

898
918
900a
900b
H.264(09)_F6-7

图9

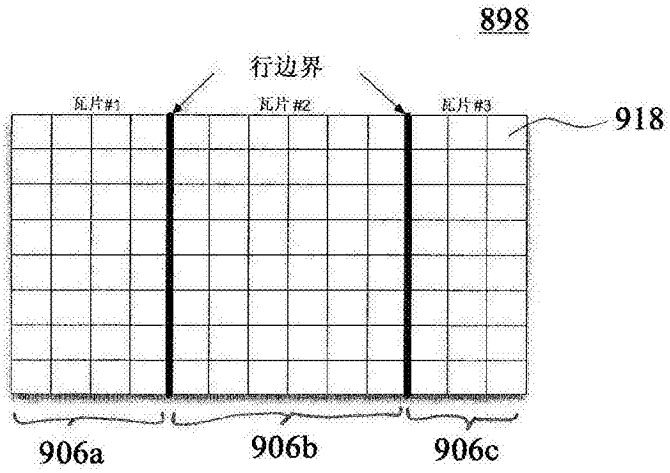

898
行边界
瓦片#1
瓦片#2
瓦片#3
918
906a
906b
906c

图10

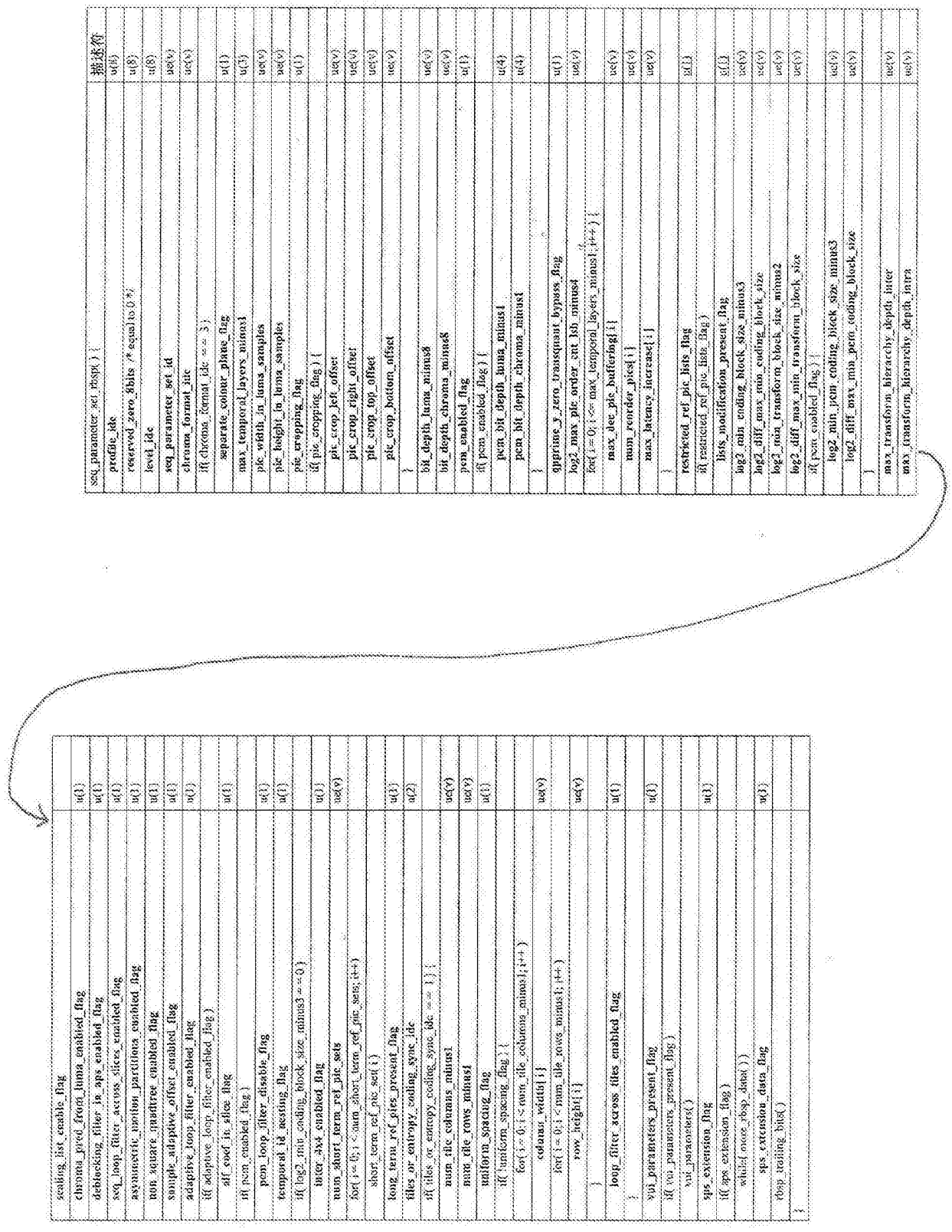

seq_parameter_set_rbsp( ) {
profile_idc
reserved_zero_8bits /* equal to 0 */
level_idc
seq_parameter_set_id
chroma_format_idc
if(chroma_format_idc == 3)
separate_colour_plane_flag
max_temporal_layers_minus1
pic_width_in_luma_samples
pic_height_in_luma_samples
pic_cropping_flag
if(pic_cropping_flag) {
pic_crop_left_offset
pic_crop_right_offset
pic_crop_top_offset
pic_crop_bottom_offset
}
bit_depth_luma_minus8
bit_depth_chroma_minus8
pcm_enabled_flag
if(pcm_enabled_flag) {
pcm_bit_depth_luma_minus1
pcm_bit_depth_chroma_minus1
}
gprime_s_zero_transquant_bypass_flag
log2_max_pic_order_cnt_lsb_minus4
for(i = 0; i <= max_temporal_layers_minus1; i++) {
max_dec_pic_buffering[i]
num_reorder_pics[i]
max_latency_increase[i]
}
restricted_ref_pic_lists_flag
if(restricted_ref_pic_lists_flag)
lists_modification_present_flag
log2_min_coding_block_size_minus3
log2_diff_max_min_coding_block_size
log2_max_transform_block_size_minus2
log2_diff_max_min_transform_block_size
if(pcm_enabled_flag) {
log2_min_pcm_coding_block_size_minus3
log2_diff_max_min_pcm_coding_block_size
}
max_transform_hierarchy_depth_inter
max_transform_hierarchy_depth_intra
描述符
u(8)
u(8)
u(8)
ue(v)
ue(v)
u(1)
u(3)
ue(v)
ue(v)
u(1)
ue(v)
ue(v)
ue(v)
ue(v)
u(1)
u(4)
u(4)
u(1)
ue(v)
ue(v)
ue(v)
ue(v)
u(1)
ue(v)
ue(v)
ue(v)
ue(v)
ue(v)
ue(v)
scaling_list_enable_flag
chroma_pred_from_luma_enabled_flag
deblocking_filter_in_aps_enabled_flag
seq_loop_filter_across_slices_enabled_flag
asymmetric_motion_partitions_enabled_flag
non_square_mattrice_enabled_flag
sample_adaptive_offset_enabled_flag
adaptive_loop_filter_enabled_flag
if(adaptive_loop_filter_enabled_flag)
alf_coef_in_slice_flag
if(pcm_enabled_flag)
pcm_loop_filter_disable_flag
temporal_id_nesting_flag
if(log2_min_coding_block_size_minus3 == 0)
inter_4x4_enabled_flag
num_short_term_ref_pic_sets
for(i = 0; i < num_short_term_ref_pic_sets; i++)
short_term_ref_pic_set(i)
long_term_ref_pics_present_flag
tiles_or_entropy_coding_sync_idc
if(tiles_or_entropy_coding_sync_idc == 1) {
num_tile_columns_minus1
num_tile_rows_minus1
uniform_spacing_flag
if(!uniform_spacing_flag) {
for(i = 0; i < num_tile_columns_minus1; i++)
column_width[i]
for(i = 0; i < num_tile_rows_minus1; i++)
row_height[i]
}
loop_filter_across_tiles_enabled_flag
}
wpl_parameters_present_flag
if(wpl_parameters_present_flag)
wpl_parameters( )
sps_extension_flag
if(sps_extension_flag)
while(more_rbsp_data( ))
sps_extension_data_flag
rbsp_trailing_bits( )
}
u(1)
u(2)
u(1)
u(1)
u(1)
ue(v)
u(2)
ue(v)
ue(v)
u(1)
ue(v)
u(1)
u(1)
u(3)
u(1)
u(1)

图11

pic_parameter_set_rbsp( ) {	描述符
pic_parameter_set_id	ue(v)
seq_parameter_set_id	ue(v)
sign_data_hiding_flag	u(1)
if(sign_data_hiding _flag)	
sign_hiding_threshold	u(4)
cabac_init_present_flag	u(1)
num_ref_idx_l0_default_active_minus1	ue(v)
num_ref_idx_l1_default_active_minus1	ue(v)
pic_init_qp_minus26	se(v)
constrained_intra_pred_flag	u(1)
enable_temporal_mvp_flag	u(1)
slice_granularity	u(2)
max_cu_qp_delta_depth	ue(v)
cb_qp_offset	se(v)
cr_qp_offset	se(v)
weighted_pred_flag	u(1)
weighted_bipred_idc	u(2)
output_flag_present_flag	u(1)
if(tiles_or_entropy_coding_sync_idc == 1) {	
tile_info_present_flag	u(1)
tile_control_present_flag	u(1)
if(tile_info_present_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
if(!uniform_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
column_width[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
row_height[i]	ue(v)
}	
}	
if(tile_control_present_flag)	
loop_filter_across_tiles_enabled_flag	u(1)
} else if(tiles_or_entropy_coding_sync_idc == 2)	
num_substreams_minus1	ue(v)
deblocking_filter_control_present_flag	u(1)
if(slice_type = =P | | slice_type = = B)	
log2_parallel_merge_level_minus2	ue(v)
pps_extension_flag	u(1)
if(pps_extension_flag)	
while(more_rbsp_data( ))	
pps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

图12

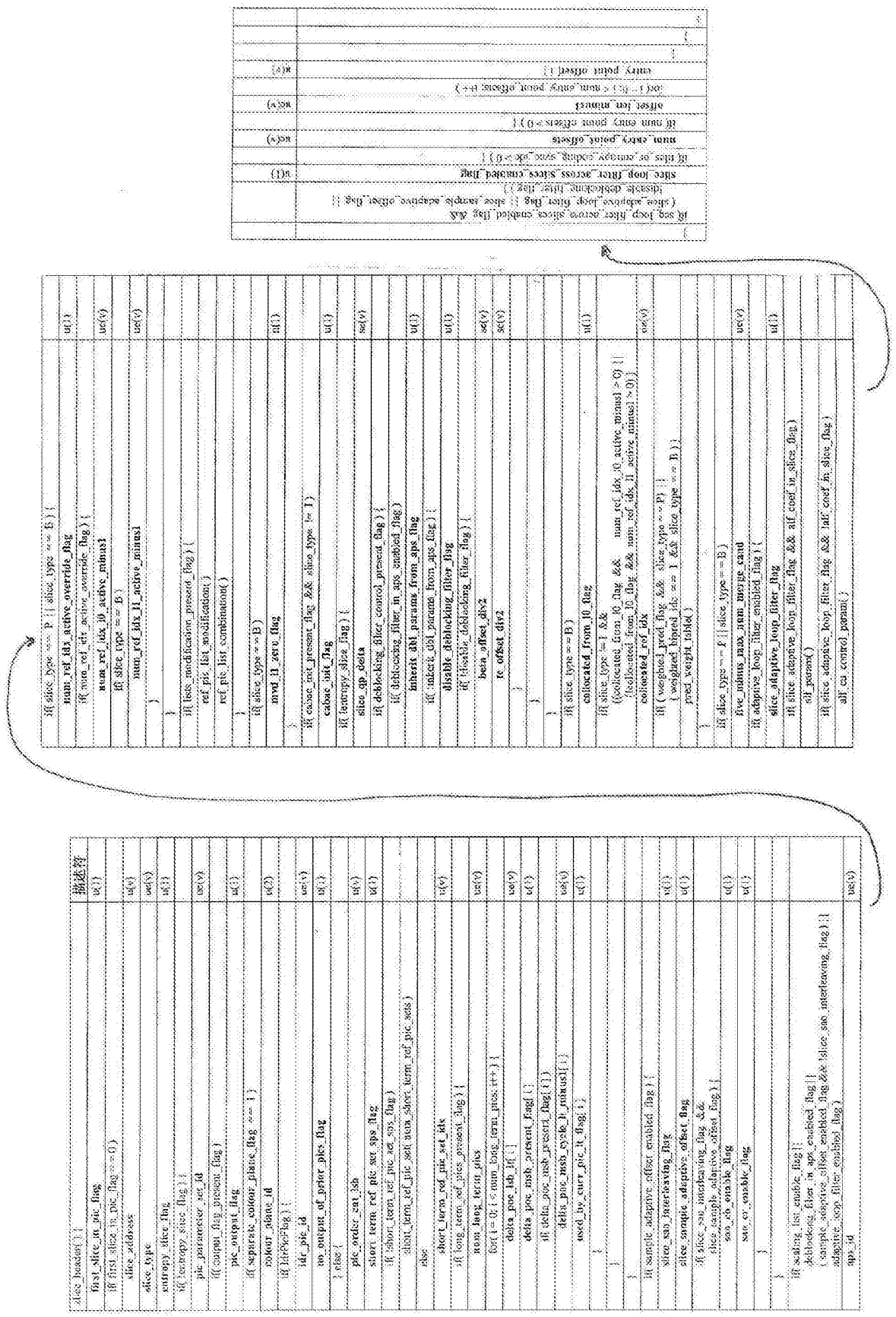


图13

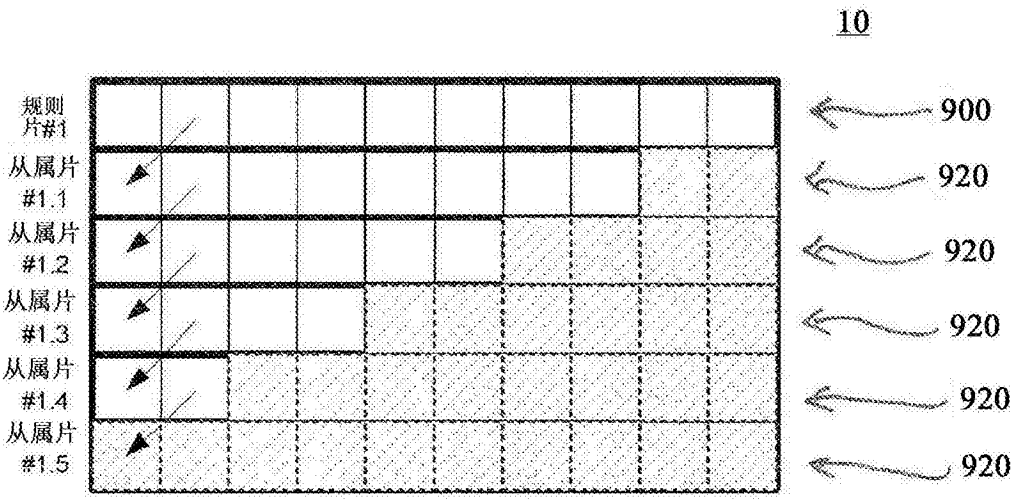
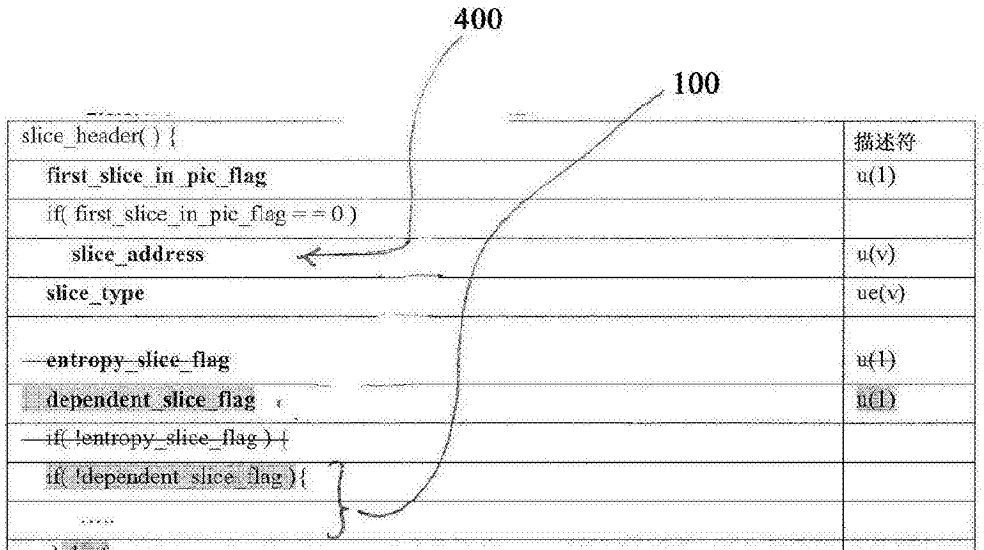


图14

pic_parameter_set_rbsp( ) {	描述符
.....	
dependent_slices_present_flag	u(1)
last_ctb_cabac_init_flg	u(1)
}	

图15



slice_header( ) {	描述符
first_slice_in_pic_flag	u(1)
if(first_slice_in_pic_flag = = 0)	
slice_address	u(v)
slice_type	ue(v)
~~entropy_slice_flag~~	~~u(1)~~
dependent_slice_flag	u(1)
~~if(!entropy_slice_flag) {~~	
if(!dependent_slice_flag){	
.....	
}else{	
no_cabac_reset_flag	u(1)
}	
....	
if(cabac_init_present_flag && slice_type != I && !no_cabac_reset_flag)	
cabac_init_flag	u(1)
....	
if(slice_type = = P || slice_type = = B)	
five_minus_max_num_merge_cand	ue(v)
if(adaptive_loop_filter_enabled_flag) {	
slice_adaptive_loop_filter_flag	u(1)
if(slice_adaptive_loop_filter_flag && alf_coef_in_slice_flag)	
alf_param( )	
if(slice_adaptive_loop_filter_flag && !alf_coef_in_slice_flag)	
alf_cu_control_param( )	
}	
if(seq_loop_filter_across_slices_enabled_flag && (slice_adaptive_loop_filter_flag || slice_sample_adaptive_offset_flag || !disable_deblocking_filter_flag))	
slice_loop_filter_across_slices_enabled_flag	u(1)
if(tiles_or_entropy_coding_sync_idc > 0) {	
num_entry_point_offsets	ue(v)
if(num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset[i]	u(v)
}	
}	
}	

图16

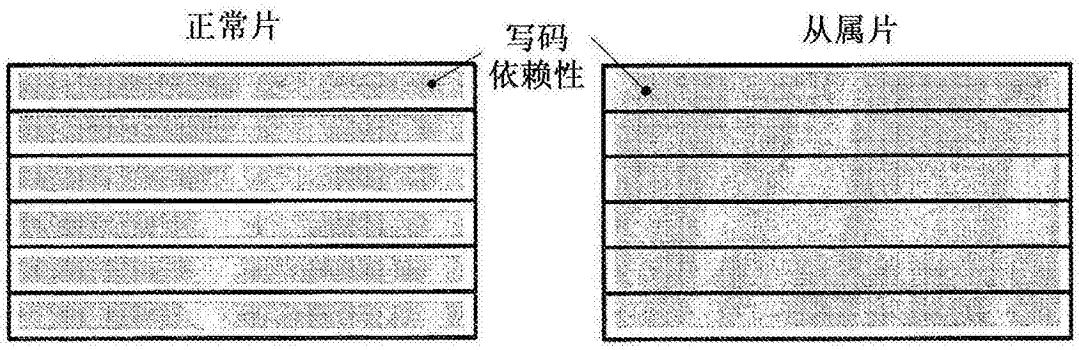

正常片
写码
依赖性
从属片

图17

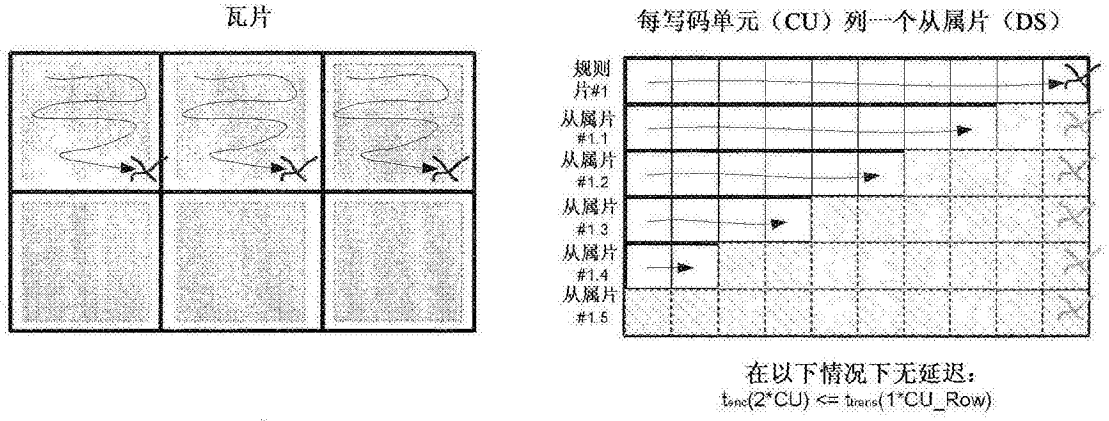

瓦片
每写码单元（CU）列一个从属片（DS）
规则
片#1
从属片
#1.1
从属片
#1.2
从属片
#1.3
从属片
#1.4
从属片
#1.5
在以下情况下无延迟：
tenc(2*CU) <= ttrans(1*CU_Row)

图18

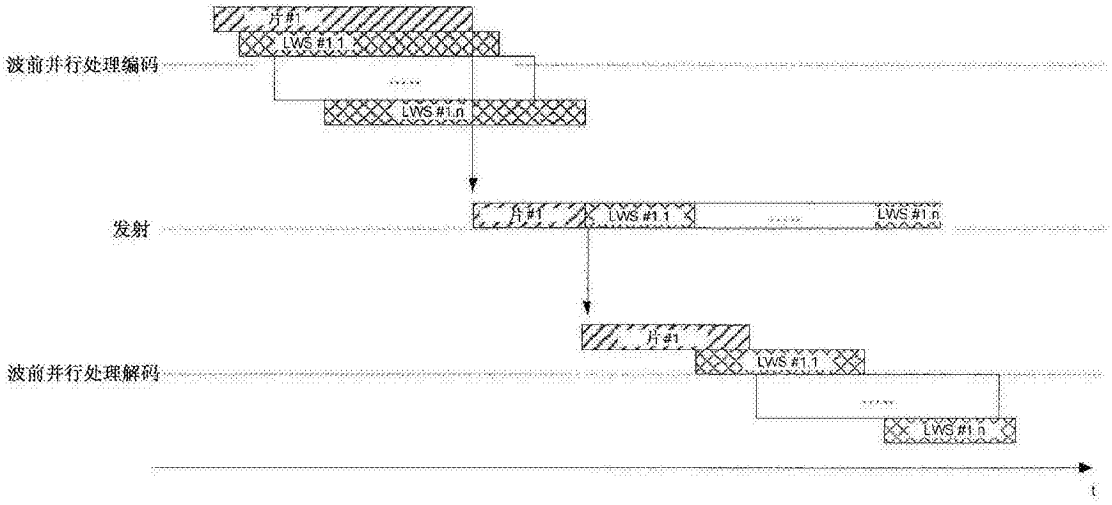

波前并行处理编码
片#1
LWS #1.1
.....
LWS #1.n
发射
片#1
LWS #1.1
.....
LWS #1.n
片#1
LWS #1.1
.....
LWS #1.n
波前并行处理解码
t

图19

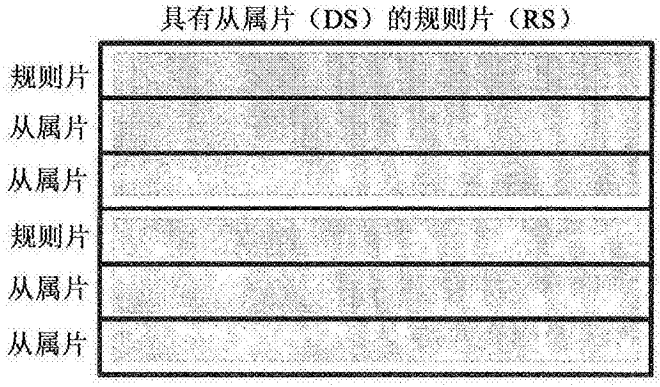
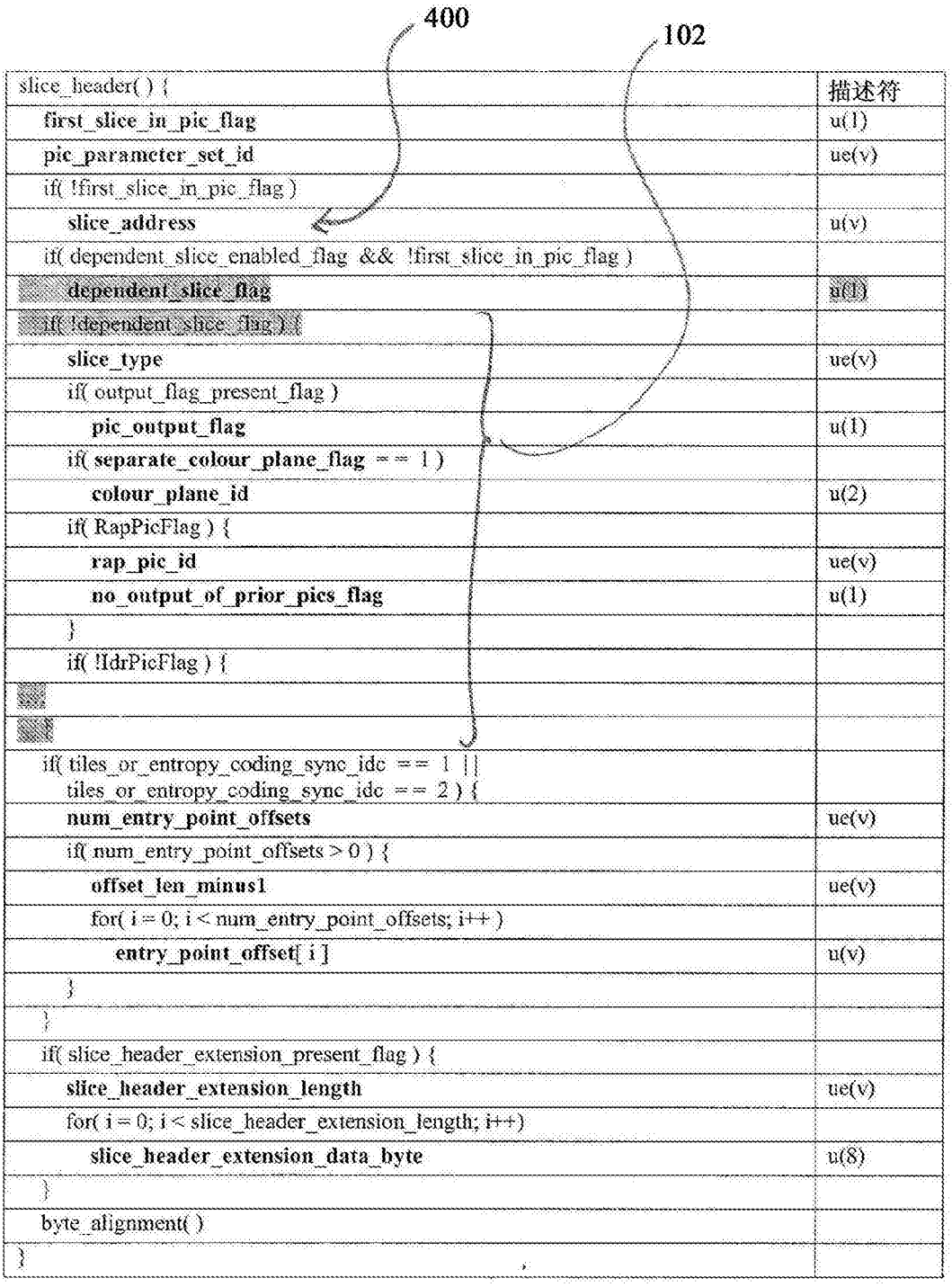


图20



slice_header( ) {	描述符
first_slice_in_pic_flag	u(1)
pic_parameter_set_id	ue(v)
if(!first_slice_in_pic_flag)	
slice_address	u(v)
if(dependent_slice_enabled_flag && !first_slice_in_pic_flag)	
dependent_slice_flag	u(1)
if(!dependent_slice_flag) {	
slice_type	ue(v)
if(output_flag_present_flag)	
pic_output_flag	u(1)
if(**separate_colour_plane_flag** == 1)	
colour_plane_id	u(2)
if(RapPicFlag) {	
rap_pic_id	ue(v)
no_output_of_prior_pics_flag	u(1)
}	
if(!IdrPicFlag) {	
...	
...	
if(tiles_or_entropy_coding_sync_idc == 1 || tiles_or_entropy_coding_sync_idc == 2) {	
num_entry_point_offsets	ue(v)
if(num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset[i]	u(v)
}	
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte	u(8)
}	
byte_alignment( )	
}	

图21

pic_parameter_set_rbsp( ) {	描述符
pic_parameter_set_id	ue(v)
seq_parameter_set_id	ue(v)
sign_data_hiding_flag	u(1)
if(sign_data_hiding _flag)	
sign_hiding_threshold	u(4)
cabac_init_present_flag	u(1)
num_ref_idx_l0_default_active_minus1	ue(v)
num_ref_idx_l1_default_active_minus1	ue(v)
[Ed. (BB): not present in HM software]	
pic_init_qp_minus26	se(v)
constrained_intra_pred_flag	u(1)
slice_granularity	u(2)
diff_cu_qp_delta_depth	ue(v)
cb_qp_offset	se(v)
cr_qp_offset	se(v)
weighted_pred_flag	u(1)
weighted_bipred_idc	u(2)
output_flag_present_flag	u(1)
transquant_bypass_enable_flag	u(1)
dependent_slice_enabled_flag	u(1)
tiles_or_entropy_coding_sync_idc	u(2)
if(tiles_or_entropy_coding_sync_idc == 1) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
if(!uniform_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
column_width[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
row_height[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
} else if(tiles_or_entropy_coding_sync_idc == 3)	
cabac_independent_flag	u(1)
...	
}	

图22

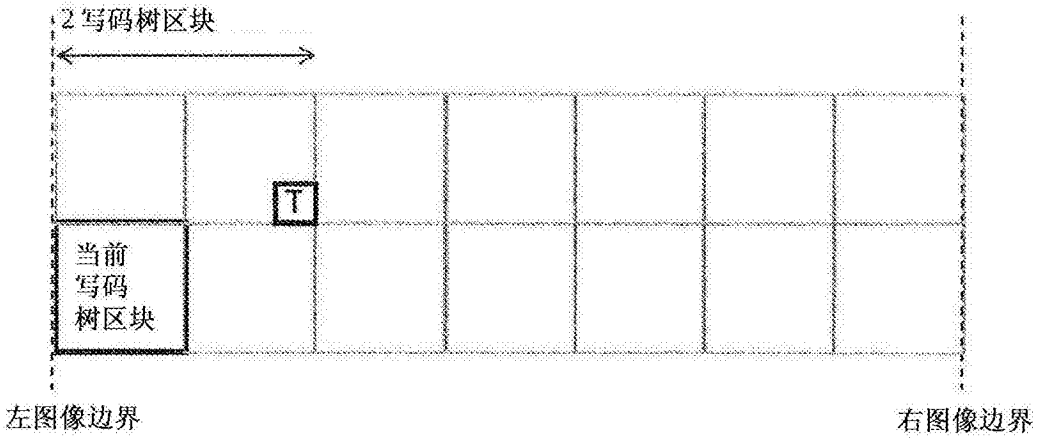

2 写码树区块
T
当前
写码
树区块
左图像边界
右图像边界

图23

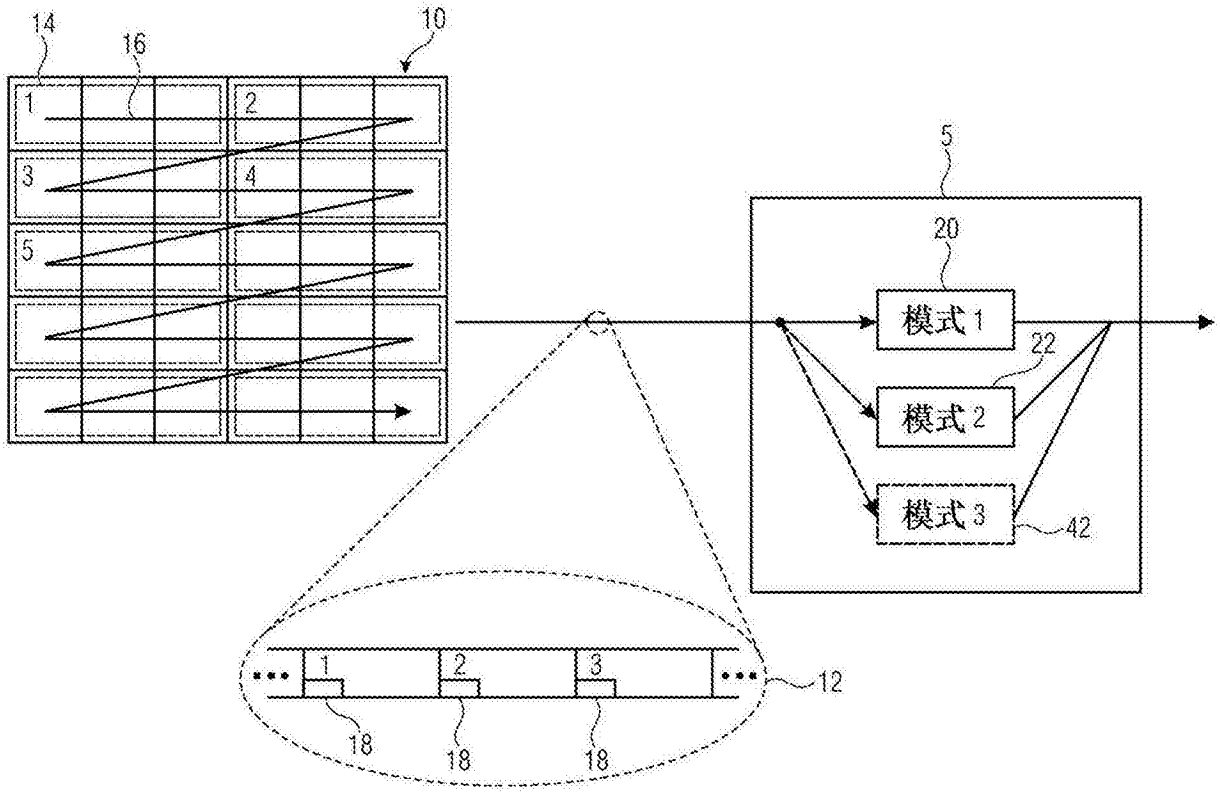

14
16
10
1
2
3
4
5
5
20
模式 1
22
模式 2
模式 3
42
1
2
3
12
18
18
18

图24

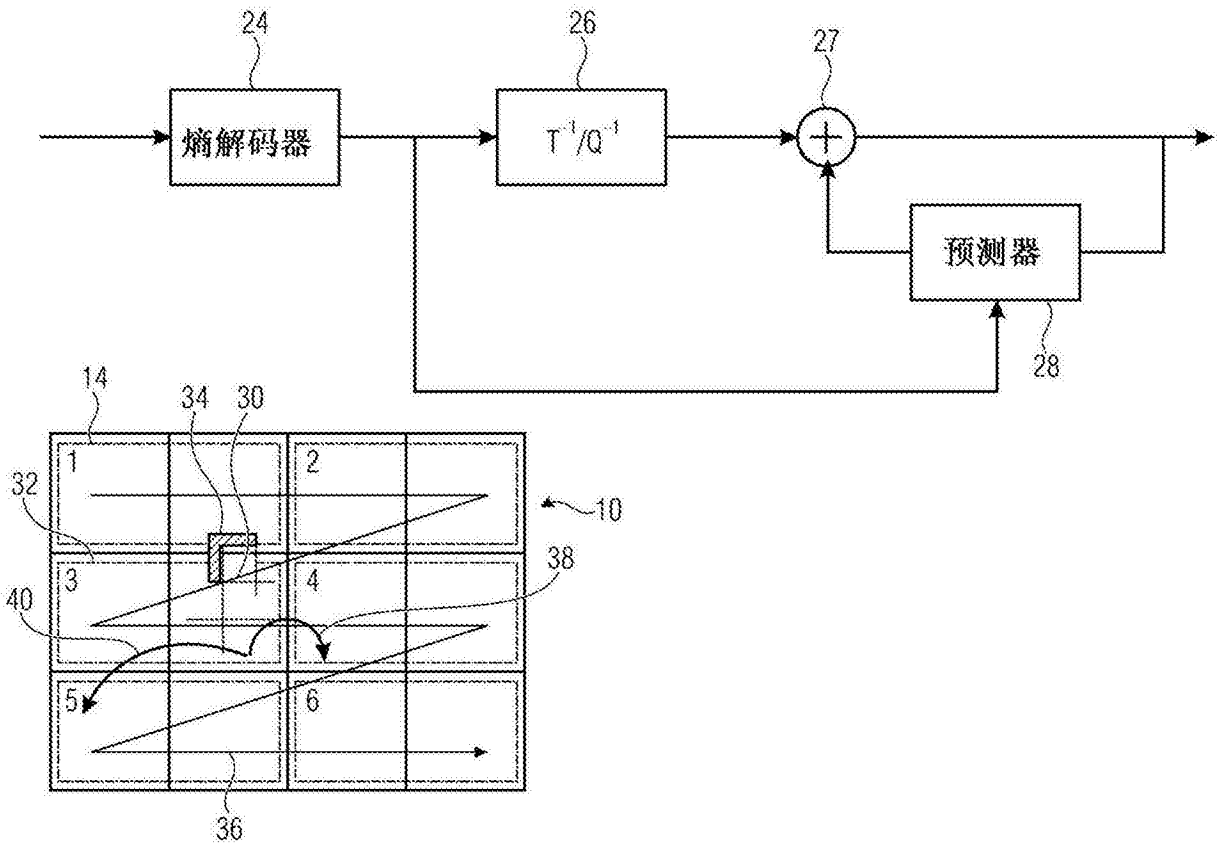

24
26
27
熵解码器
T^{-1}/Q^{-1}
预测器
28
14
34
30
1
2
32
10
3
4
38
40
5
6
36

图25

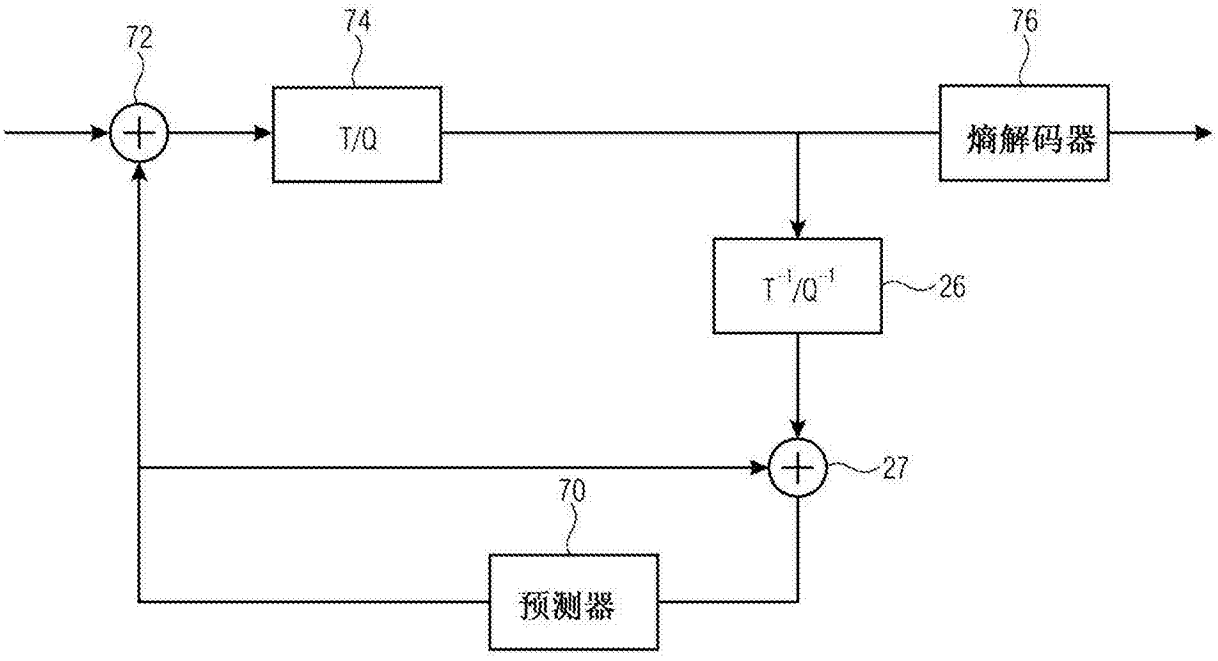

72
74
76
T/Q
熵解码器
T^{-1}/Q^{-1}
26
27
70
预测器

图26

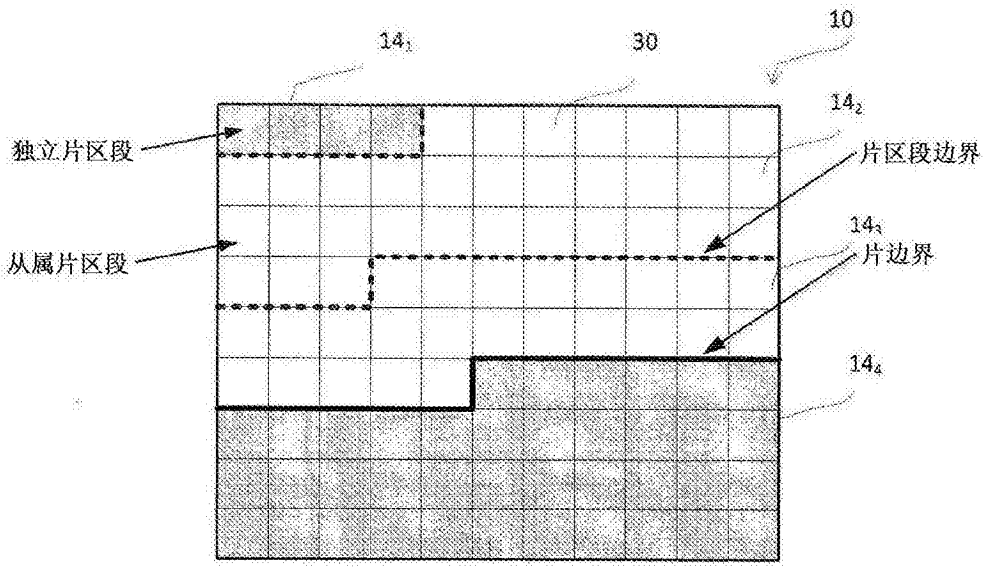


图27

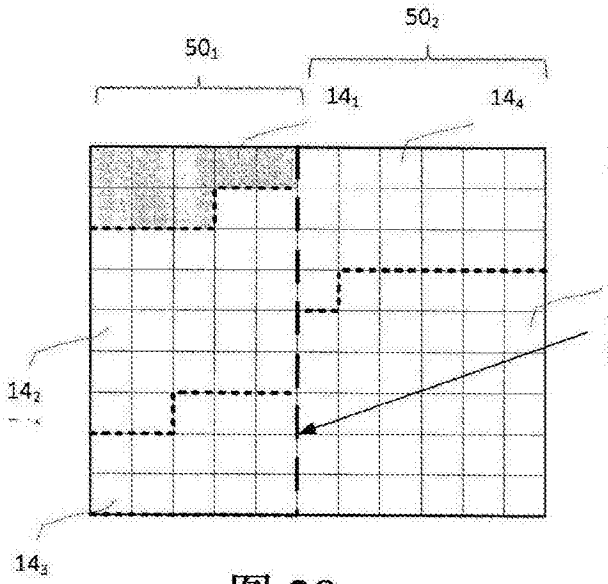


图 28a

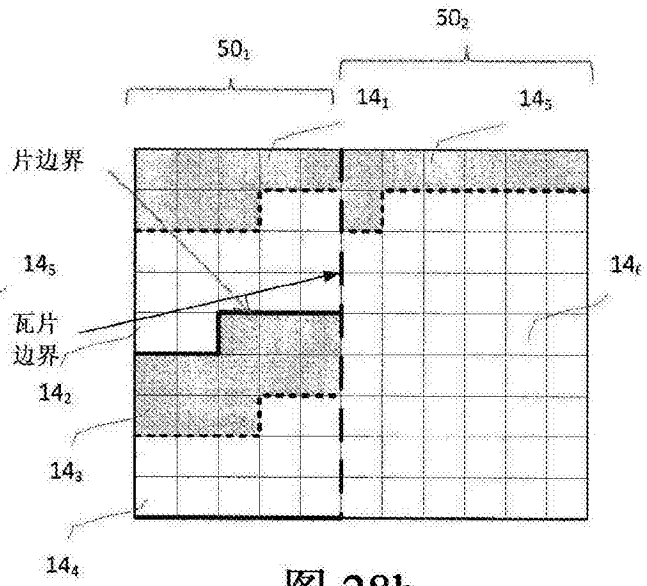


图 28b

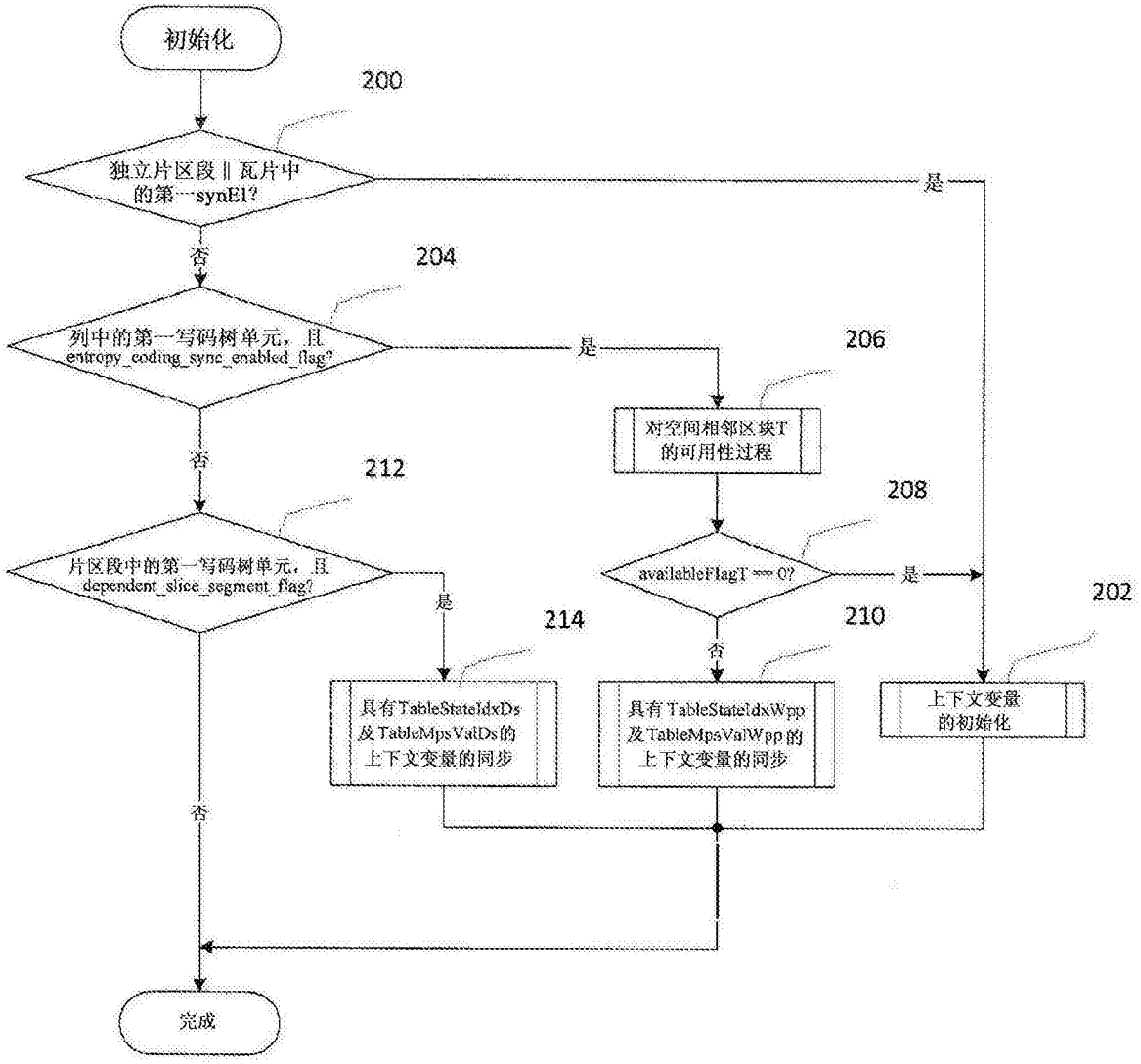

初始化
200
独立片区段‖瓦片中的第一synEl?
是
否
204
列中的第一写码树单元，且entropy_coding_sync_enabled_flag?
是
206
对空间相邻区块T的可用性过程
否
212
208
片区段中的第一写码树单元，且dependent_slice_segment_flag?
availableFlagT == 0?
是
202
是
214
210
否
具有TableStateIdxDs及TableMpsValDs的上下文变量的同步
具有TableStateIdxWpp及TableMpsValWpp的上下文变量的同步
上下文变量的初始化
否
完成

图29

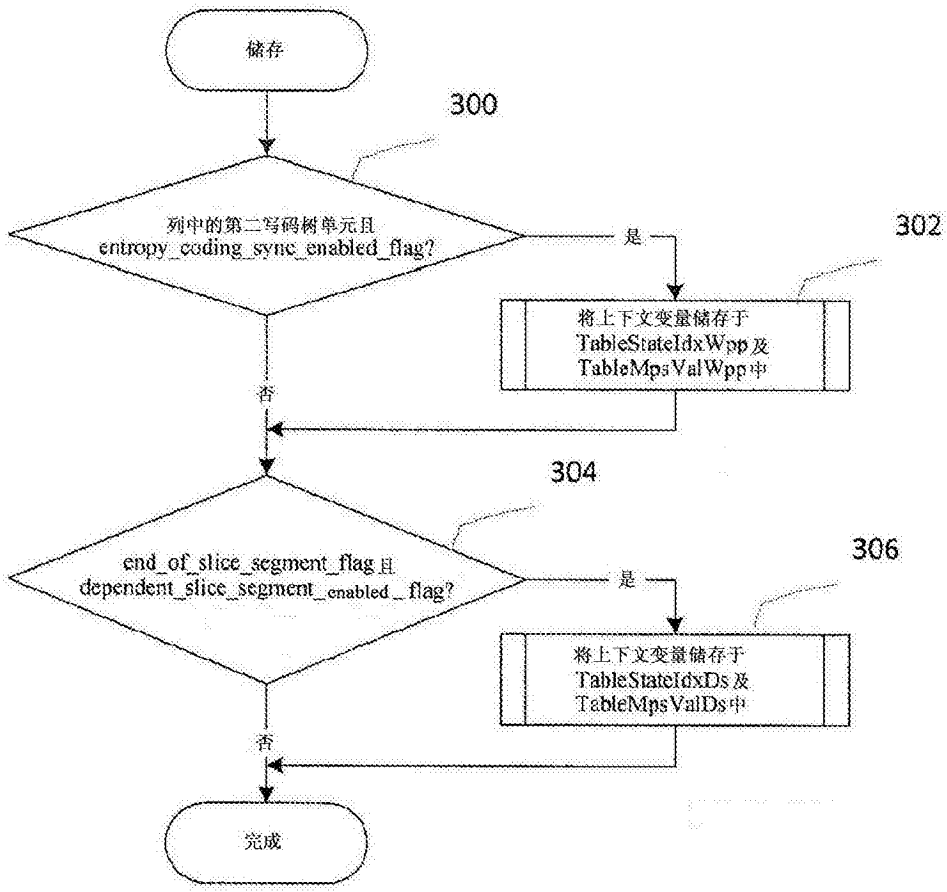

储存
300
列中的第二写码树单元且
entropy_coding_sync_enabled_flag?
是
302
将上下文变量储存于
TableStateIdxWpp及
TableMpsValWpp中
否
304
end_of_slice_segment_flag且
dependent_slice_segment_enabled_flag?
是
306
将上下文变量储存于
TableStateIdxDs及
TableMpsValDs中
否
完成

图30

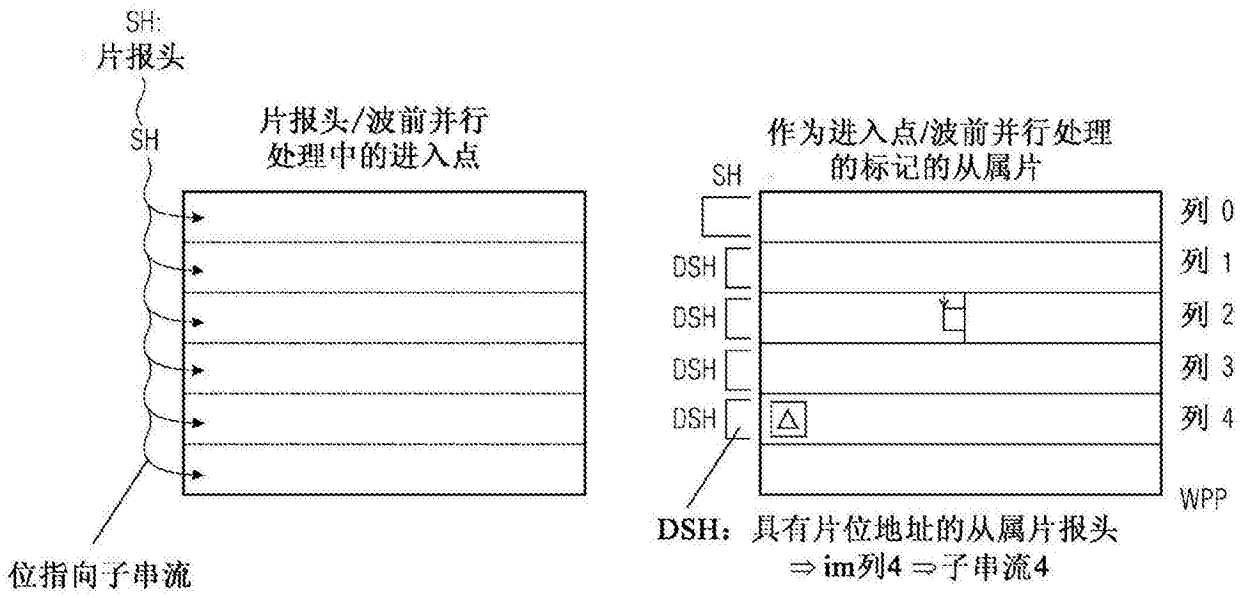

SH:
片报头
SH
片报头/波前并行
处理中的进入点
位指向子串流
作为进入点/波前并行处理
的标记的从属片
SH
DSH
DSH
DSH
DSH
列 0
列 1
列 2
列 3
列 4
WPP
DSH：具有片位地址的从属片报头
⇒im列4 ⇒子串流4

图31