

**UNITED STATES INTERNATIONAL TRADE COMMISSION  
WASHINGTON, D.C.**

**Before the Honorable Doris Johnson Hines  
Administrative Law Judge**

**In the Matter of:**

**CERTAIN VIDEO CAPABLE DEVICES,  
INCLUDING COMPUTERS,  
STREAMING DEVICES, TELEVISIONS,  
CAMERAS, AND COMPONENTS AND  
MODULES THEREOF**

**Investigation No. 337-TA-1379**

**COMPLAINANTS' OPENING CLAIM CONSTRUCTION BRIEF**

## TABLE OF CONTENTS

|      |                                                                                                                       |    |
|------|-----------------------------------------------------------------------------------------------------------------------|----|
| I.   | INTRODUCTION .....                                                                                                    | 1  |
| II.  | LEVEL OF ORDINARY SKILL IN THE ART .....                                                                              | 2  |
| III. | LEGAL STANDARD.....                                                                                                   | 2  |
| A.   | Lexicography .....                                                                                                    | 2  |
| B.   | Claim Differentiation .....                                                                                           | 3  |
| C.   | Technical Background to the '808 Patent.....                                                                          | 4  |
| 1.   | Video Coding and Motion Compensation .....                                                                            | 4  |
| 2.   | Skip Coding Modes at the Time of the Invention.....                                                                   | 9  |
| 3.   | The Problems Existing in the Art.....                                                                                 | 10 |
| 4.   | The '808 Patent's Redefined "Skip Coding Mode" Solved a Problem in the Art.....                                       | 11 |
| a.   | The Redefined Skip Coding Mode Skips Residual Information .....                                                       | 12 |
| b.   | Advantages of the Redefined Skip Coding Mode.....                                                                     | 12 |
| D.   | The Disputed Claim Term .....                                                                                         | 15 |
| 1.   | "skip coding mode" .....                                                                                              | 15 |
| IV.  | ARGUMENT .....                                                                                                        | 15 |
| A.   | The Inventor's Lexicography Governs .....                                                                             | 16 |
| B.   | Claim Differentiation Does Not Override Lexicography or the Teachings of the Specification .....                      | 17 |
| 1.   | Respondents' Construction Contradicts the Lexicography.....                                                           | 17 |
| 2.   | Respondents' Construction Contradicts the Repeated and Consistent Descriptions of Invention in the Specification..... | 19 |
| C.   | Respondents Misapply Claim Differentiation .....                                                                      | 20 |
| V.   | CONCLUSION.....                                                                                                       | 23 |

## TABLE OF AUTHORITIES

|                                                                                                                                            | Page(s)   |
|--------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| <b>CASES</b>                                                                                                                               |           |
| <i>Astrazeneca AB, Aktiebolaget Hassle, KBI-E, Inc. v. Mut. Pharm. Co.</i> ,<br>384 F.3d 1333 (Fed. Cir. 2004).....                        | 3         |
| <i>Bell Atl. Network Servs., Inc. v. Covad Commc'ns Grp., Inc.</i> ,<br>262 F.3d 1258 (Fed. Cir. 2001).....                                | 16        |
| <i>Curtiss-Wright Flow Control Corp. v. Velan, Inc.</i> ,<br>438 F.3d 1374 (Fed. Cir. 2006).....                                           | 3, 22     |
| <i>Edwards Lifesciences LLC v. Cook Inc.</i> ,<br>582 F.3d 1322 (Fed. Cir. 2009).....                                                      | 3, 21, 22 |
| <i>Howmedica Osteonics Corp. v. Zimmer, Inc.</i> ,<br>822 F.3d 1312 (Fed. Cir. 2016).....                                                  | 4, 17     |
| <i>Laitram Corp. v. Rexnord, Inc.</i> ,<br>939 F.2d 1533 (Fed. Cir. 1991).....                                                             | 3         |
| <i>Markman v. Westview Instruments, Inc.</i> ,<br>52 F.3d 967 (Fed. Cir. 1995) ( <i>en banc</i> ), <i>aff'd</i> , 517 U.S. 370 (1996)..... | 2         |
| <i>Multiform Desiccants, Inc. v. Medzam, Ltd.</i> ,<br>133 F.3d 1473 (Fed. Cir. 1998).....                                                 | 3, 4      |
| <i>Phillips v. AWH Corp.</i> ,<br>415 F.3d 1303 (Fed. Cir. 2005) ( <i>en banc</i> ) .....                                                  | 2, 16, 17 |
| <i>Schoenhaus v. Genesco, Inc.</i> ,<br>440 F.3d 1354 (Fed. Cir. 2006).....                                                                | 3         |
| <i>Tandon Corp. v. U.S. Int'l Trade Comm'n</i> ,<br>831 F.2d 1017 (Fed. Cir. 1987).....                                                    | 3         |
| <i>Tubular Rollers, LLC v. Maximus Oilfield Prod., LLC</i> ,<br>No. 2021-2319, 2023 WL 4230371 (Fed. Cir. June 28, 2023).....              | 4         |
| <b>STATUTES</b>                                                                                                                            |           |
| 35 U.S.C. § 112.....                                                                                                                       | 3         |

**MARKMAN EXHIBITS LIST**

| <b>Exhibit No.</b> | <b>Description</b>                                         |
|--------------------|------------------------------------------------------------|
| 1                  | JVT-A003 (JM1 of the JVT)<br>NOK_AMHP_ITC00079634-00079711 |

**DECLARATIONS**

| <b>Exhibit No.</b> | <b>Description</b>                 |
|--------------------|------------------------------------|
| 2                  | Declaration of Dr. Michael Orchard |

## I. INTRODUCTION

Per Order No. 13 (January 9, 2024), Complaints’ Nokia Technologies Oy and Nokia Corp. provide their opening claim construction brief regarding the disputed term “skip coding mode” in U.S. Patent No. 7,532,808 (the “’808 Patent”).

Nokia and the Commission Investigative Staff’s proposed construction for “skip coding mode” is copied directly from the lexicography provided in the patent. Respondents seek to alter this lexicography due to alleged claim differentiation. The ALJ should adopt Nokia and the Staff’s proposed construction for three reasons.

First, under *Phillips*, the inventor’s lexicography governs the meaning of a term. Here, there is a clear statement of lexicography applied by Nokia’s and the Staff’s proposed construction.

Second, claim differentiation is at most a rebuttable presumption that cannot override lexicography or the teachings of the specification. Respondents’ proposed construction contradicts both the inventor’s lexicography and the specification’s repeated and consistent descriptions of the invention, which state that no additional information (including residual information) is provided by the skip coding mode for a skip mode macroblock.

Third, Respondents’ application of claim differentiation is mistaken because the respective independent and dependent claims are different on their face. As will be explained, the ’808 Patent’s skip coding mode does not include “residual information.” Respondents’ apparent contention is that this fact renders four dependent claims superfluous. This is incorrect. The four dependent claims in question add the limitation “wherein no residual information is provided for the first segment [the skip mode macroblock] *in the encoded bitstream.*” (Emphasis added.) Not providing residual information anywhere *in the encoded bitstream* is a further limitation beyond the ’808 Patent’s skip coding mode, which itself is only one part of the bitstream. Thus,

this is not a situation where the independent claims require the same limitation added by certain dependent claims as Respondents appear to allege.

## **II. LEVEL OF ORDINARY SKILL IN THE ART**

A person of ordinary skill in the art (“POSITA”) for the ’808 Patent would have at least a bachelor’s degree in electrical engineering, computer engineering, or a related field, with about two years of experience in video decoding or encoding technologies. More work or practical experience may qualify one not having the requisite education as a person with ordinary skill in the art while a higher level of education could offset less experience.

## **III. LEGAL STANDARD**

Claim construction is a matter of law for the court. *Markman v. Westview Instruments, Inc.*, 52 F.3d 967, 970-971 (Fed Cir. 1995) (*en banc*), *aff’d*, 517 U.S. 370 (1996). Claim construction focuses on the intrinsic evidence, including the claims, patent specification, and prosecution history. *See Phillips v. AWH Corp.*, 415 F.3d 1303, 1314 (Fed. Cir. 2005) (*en banc*). Claims are given their ordinary and customary meaning to a POSITA at the time of the invention. *Id.* at 1313. Despite the importance of claim terms, *Phillips* made clear that “the POSITA is deemed to read the claim term not only in the context of the particular claim in which the disputed term appears, but in the context of the entire patent, including the specification.” *Id.* Accordingly the specification is the “primary basis for construing the claims.” *Id.* at 1316.

### **A. Lexicography**

The written description in a patent specification “may act as a sort of dictionary, which explains the invention and may define terms used in the claims.” *Markman*, 52 F.3d 967 at 979. This is called lexicography, where “a special definition [is] given to a claim term by the patentee that differs from the meaning it would otherwise possess.” *Phillips*, 415 F.3d at 1316 (citations

omitted). An inventor is free to act as his or her own lexicographer. *Schoenhaus v. Genesco, Inc.*, 440 F.3d 1354, 1358 (Fed. Cir. 2006) (citations omitted). An inventor “may set forth any special definitions of the claim terms in the patent specification or file history, either expressly or impliedly.” *Id.* Lexicography does not require “a statement in the form ‘I define \_\_\_\_\_ to mean \_\_\_\_\_’ ... such rigid formalism is not required.” *Astrazeneca AB, Aktiebolaget Hassle, KBI-E, Inc. v. Mut. Pharm. Co.*, 384 F.3d 1333, 1339 (Fed. Cir. 2004).

When the meaning of a claim term is at issue, “the inventor’s lexicography governs.” *Phillips*, 415 F.3d at 1316 (citation omitted).

## **B. Claim Differentiation**

By contrast, “claim differentiation is a rule of thumb that does not trump the clear import of the specification.” *Edwards Lifesciences LLC v. Cook Inc.*, 582 F.3d 1322, 1332 (Fed. Cir. 2009). “Claim differentiation is a guide, not a rigid rule.” *Laitram Corp. v. Rexnord, Inc.*, 939 F.2d 1533, 1538 (Fed. Cir. 1991).

The doctrine of claim differentiation “presumes that there is a difference in scope among the claims of a patent.” *Multiform Desiccants, Inc. v. Medzam, Ltd.*, 133 F.3d 1473, 1479 (Fed. Cir. 1998). As relevant here, claim differentiation “refers to the presumption that an independent claim should not be construed as requiring a limitation added by a dependent claim.” *Curtiss-Wright Flow Control Corp. v. Velan, Inc.*, 438 F.3d 1374, 1380 (Fed. Cir. 2006). This follows the statute. *Id.*; 35 U.S.C. § 112 (“[A] dependent form shall contain a reference to a claim previously set forth and then specify a further limitation of the subject matter claimed.”). However, “practice has long recognized that “claims may be multiplied ... to define the metes and bounds of the invention in a variety of different ways.” *Tandon Corp. v. U.S. Int’l Trade Comm’n*, 831 F.2d 1017, 1023 (Fed. Cir. 1987) (citations and quotation omitted).

Claim differentiation cannot “broaden claims beyond their correct scope.” *Multiform Desiccants*, 133 F.3d at 1479. The Federal Circuit has held that “claim differentiation is a rebuttable presumption that may be overcome by a contrary construction dictated by the written description or prosecution history.” *Howmedica Osteonics Corp. v. Zimmer, Inc.*, 822 F.3d 1312, 1324 (Fed. Cir. 2016). Indeed, courts have “adopted a construction rendering dependent claims meaningless when that construction was supported by either the specification or the prosecution history.” *Tubular Rollers, LLC v. Maximus Oilfield Prod., LLC*, No. 2021-2319, 2023 WL 4230371, at \*6 (Fed. Cir. June 28, 2023).

### **C. Technical Background to the '808 Patent**

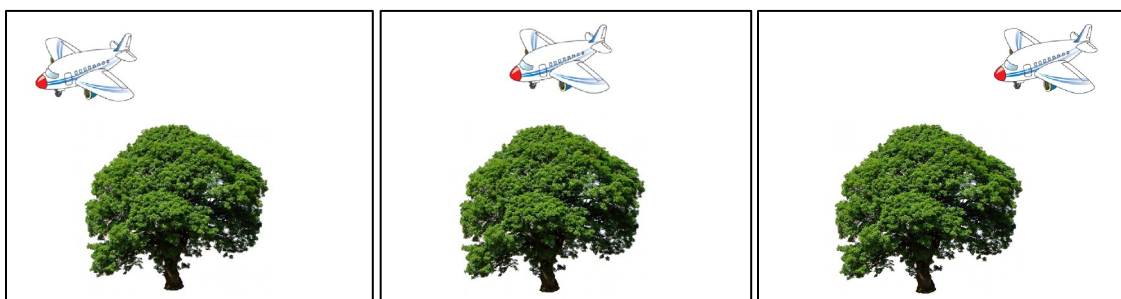
#### **1. Video Coding and Motion Compensation**

The '808 Patent relates to motion compensation in video coding. '808 Patent at 1:9-11. Video coding makes storage, playback, and transmission of digital video content more efficient through, for example, decreasing the size of digital video files—without noticeably degrading the picture quality—by “reducing the redundant or perceptually irrelevant parts of video sequences.” '808 Patent at 2:36-38. Efficient video coding makes it easier to send high quality videos without the need for high transmission bandwidth and without degrading the picture quality. This benefit is important to anyone who watches, stores, or sends videos and is used in many everyday applications, such as streaming videos over a wireless network.

The inventions of the '808 Patent disclose solutions that minimize the size of video files while maintaining high video quality for viewing. Specifically, the '808 Patent relates to specific motion compensated prediction techniques for coded video sequences. '808 Patent at 3:11-15. A coded video sequence is comprised of a number of frames (sometimes referred to as “images” or “pictures”). '808 Patent at 1:29-33. The image frames of a video sequence are displayed one after

the other at a relatively fast rate to create the illusion of motion. '808 Patent at 1:15-19. One way a video can be encoded or compressed without unnecessarily losing picture quality is by reducing redundancy between frames in a sequence—for example a stationary object that appears in view for some duration of a movie. One type of redundancy in video compression is temporal redundancy. The term “temporal redundancy expresses the fact that objects appearing in one frame of a [video] sequence are likely to appear in subsequent frames.” '808 Patent at 2:42-43.

For example, a moving object may travel across a stationary background, such as when an airplane flies over a stationary backdrop. As an additional example, a camera may pan across a scene such that stationary areas would appear moving while moving objects may look either slower or faster depending on the direction of panning relative to the motion of the moving objects. *See* '808 Patent at 1:26-31. The example below shows three video frames and illustrates how the redundant or perceptually irrelevant data (e.g., pixels constituting the background) may exist across several frames of a video sequence. For example, the tree in each picture does not move, and the image of the plane does not change but moves across the picture. In this example, even though the plane has moved, the plane itself contains no new information.<sup>1</sup>



In the exemplary video frames shown above, the tree may correspond to redundant or perceptually irrelevant data in a video file. Additionally, the video data relating to the plane, which

---

<sup>1</sup> This is distinct from a scenario where the image content making up the plane itself changes. This can occur, for instance, with lighting or motion changes that alter the appearance of the plane from the perspective of a camera.

involves the motion of an object across the screen, includes a redundant image of the plane, but must account for the plane's motion. Redundancy here refers to "temporal redundancy," that the objects (i.e., the tree and the plane) contain no new information across frames.

Motion compensated prediction can reduce temporal redundancy by allowing the image content of some frames in a digital video sequence to be "predicted" from one or more other frames in the sequence. '808 Patent at 3:15-20. For example, the encoder can provide motion information to the decoder that allows the decoder to predict where the plane should appear in the frame.

One way to do this involves different categories of frames when encoding a video. "Reference frames" (or "reference pictures") can serve as points of comparison from which the content of other frames can be predicted, for example, by tracking the motion of objects or regions between the reference frame and the other frames using "motion vectors."<sup>2</sup> '808 Patent at 3:18-21. Another category of frames (sometimes called "INTER-coded" or "P-frames") can be predicted from reference frames. '808 Patent at 3:18-28. Motion compensated prediction allows for transmitting less data while still achieving the same or similar quality of streaming video.

In order to, for example, better address motion within a frame, a video coding system may divide each frame of a video sequence into "macroblocks," which in the preferred embodiment are formed from regions of 16x16 image pixels. '808 Patent at 1:58-2:6. Figure 9 of the '808 Patent depicts a video frame, which consists of the entire rectangle divided into macroblocks, where each smaller square is an individual macroblock.

---

<sup>2</sup> Motion vectors, as a type of vector, convey both (1) direction and (2) magnitude. A "zero motion vector" will have a magnitude of zero. A non-zero motion vector will have a magnitude greater than zero and some direction. The direction of a vector can be expressed by multiple directional components (e.g., horizontal or vertical components). Negative numbers may be used to express certain directions. As an example, Figure 9 of the '808 Patent depicts multiple non-zero vectors with the same direction and magnitude.

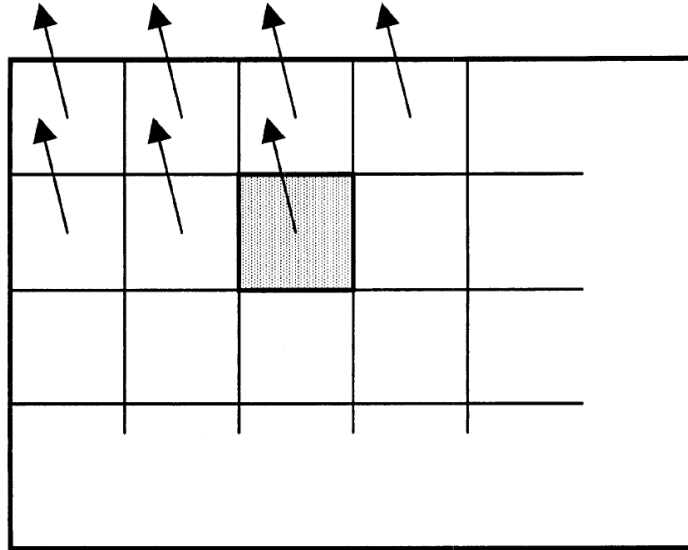


FIG. 9

In some video coding systems, “video pictures are divided into macroblocks...and coded on a macroblock-by-macroblock basis.” ’808 Patent at 10:41-42. Macroblocks can be assigned a “coding mode” by encoders based on the characteristics of the macroblock in question and also according to the motion in the video sequence. ’808 Patent at 10:47-50. The “coding mode” provides an indication to the decoder as to how to predict that macroblock. The ’808 Patent describes various coding modes associated with a particular prior art codec, referred to as JM1 of the JVT codec. ’808 Patent at 10:40-67. JM1 of the JVT codec was the video coding recommendation that was being developed prior to the ’808 Patent by the Joint Video Team of the Motion Pictures Expert Group (MPEG) and the Video Coding Experts Group (VCEG). *See* ’808 Patent at 10:27-39. The recommendation sought to improve many elements of the H.263 codec. The coding modes of JM1 are shown in Table 3 of the ’808 Patent.

| TABLE 3                               |         |                  |
|---------------------------------------|---------|------------------|
| <u>Macroblock Coding Modes of JM1</u> |         |                  |
| Codeword Index                        | Mode    | UVLC Codeword    |
| —                                     | SKIP    | Run-Length Coded |
| 0                                     | 16 × 16 | 1                |
| 1                                     | 16 × 8  | 001              |
| 2                                     | 8 × 16  | 011              |
| 3                                     | 8 × 8   | 00001            |
| 4                                     | 8 × 4   | 00011            |
| 5                                     | 4 × 8   | 01001            |
| 6                                     | 4 × 4   | 01011            |

“UVLC Codeword” in Table 3 refers to “Universal Variable Length Codes” that represent data elements in a video bitstream. ’808 Patent at 11:32-37. In JM1, the “codewords indicating the coding modes are assigned in such a way that the shortest codeword is used to represent the coding mode that is statistically most likely to occur.” ’808 Patent at 11:29-32. JM1 assumed SKIP was the most likely coding mode.<sup>3</sup> ’808 Patent at 12:18-20. As Dr. Michael Orchard explains in his declaration, a POSITA would understand that the non-skip coding modes of JM1 in Table 3 use additional information called prediction error values (residual information) for the reconstruction of pixel values of image blocks.<sup>4</sup> Ex. 2, Declaration of Michael Orchard (“Orchard Dec.”) at ¶ 18. By contrast, a POSITA would understand that the *SKIP* mode of JM1 *skips* this additional information and thus the pixel values of SKIP blocks are reconstructed directly from pixel values of the reference frame without residual information as part of the SKIP mode. Orchard Dec. at ¶ 19.

Decisions as to which coding mode will apply to a given macroblock are made by video encoders as part of the “motion estimation process performed by [] motion estimation block[s].”

<sup>3</sup> As will be discussed, the ’808 explained that this was not a valid assumption. ’808 Patent at 12:41-43.

<sup>4</sup> The ’808 Patent explains that these non-SKIP modes are INTER modes named based on the division of macroblocks into NxM pixel sub-blocks. ’808 Patent at 10:50-63. For example, the 4x4 mode involves sub-blocks of size 4x4 pixels. *See id.*

'808 Patent at 11:1-8. For example, an encoder can perform motion estimation operations to determine the most efficient coding mode for a given macroblock. '808 Patent at 11:14-18. Because encoders can assign particular coding modes to each macroblock, a corresponding video decoder must be aware of these coding modes “in order for it to correctly decode received information relating to the macroblock in question.” '808 Patent at 11:19-24. Encoders (after selecting coding modes through the motion estimation process) can provide “an indication of the coding mode assigned to each macroblock” in a video bitstream. '808 Patent at 11:24-27.

Depending on the coding mode in question, in the video bitstream this indication may accompany other information necessary for the decoder to appropriately predict the macroblock (e.g., “prediction error,” which is a form of residual information). '808 Patent at 3:25-39; Orchard Dec. at ¶ 18. For example, the '808 Patent describes that, where motion-compensated prediction alone does not provide “a sufficiently precise representation of the image content,” prediction error “represents the difference between a decoded version of the INTER-coded frame and the image content of the frame to be coded.” '808 Patent at 3:28-35. As explained, a POSITA would understand at the time of the invention that prediction error processes are used for inter prediction (or INTER) modes but are *skipped* for skip coding modes. Orchard Dec. at ¶¶ 18-19.

## **2. Skip Coding Modes at the Time of the Invention**

The prior art “SKIP (or skip) mode” of the JM1 of the JVT codec is particularly important for the '808 Patent. This mode was used to “indicate[] that the macroblock is to be copied from the reference video frame without using motion compensated prediction.” '808 Patent at 10:64-67; Orchard Dec. at ¶ 19. In other words, the macroblock in the frame being decoded is the exact same as a previous macroblock that was transmitted as part of the reference video frame. Because it is the exact same, the decoder can just copy the corresponding macroblock from the prior video

frame, and the encoder does not have to transmit the content of that macroblock a second time. Thus, in scenarios involving objects exhibiting no motion across frames, use of the prior art “SKIP (or skip) mode” provided compression benefits. As a person of skill in the art would understand, JM1 skip coding mode did not involve any prediction error or residual information as part of that coding mode. Orchard Dec. at ¶ 19; ’808 Patent at 10:63-67; Ex. 1, JVT-A003 (JM1 of the JVT) at Subclause 3.4.1 (defining skip mode macroblocks) NOK\_AMHP\_ITC00079643 at 79650. Indeed, as Dr. Michael Orchard explains in his declaration, a person of skill in the art would appreciate that “a key aspect of skip coding modes is that the coding, transmission, and decoding of residual information is ‘skipped’ for the skip coding mode macroblocks.” Orchard Dec. at ¶ 26.

### **3. The Problems Existing in the Art**

The ’808 Patent points out that a problem with JM1 of the JVT codec was its inability to address specific types of motion in video sequences, such as panning or zooming. ’808 Patent at 12:41-47. In particular, when certain motions were present, the older JM1 of the JVT coding system could not use the prior art skip mode and instead “the codec [was] forced to use high overhead []macroblock coding modes,” ’808 Patent at 12:45-47, which results in more data needing to be sent for a given video stream. Thus, in the prior art, skip coding mode could only be used when no motion was present. As a result, one problem with the JM1 approach was that “the assumption that skip mode is always the most probable is not valid.” ’808 Patent at 12:41-43.

The ’808 Patent describes two prior attempts to address the problem of efficiently coding motion such as panning or zooming. Both of these attempts continued to use the old skip modes and also had additional shortcomings that the ’808 Patent was able to overcome. These attempts were known as (a) “global motion compensation” (such as that used in H.263+) and (b) “Global Motion Vector Coding.” ’808 Patent at 12:48-58. The ’808 Patent describes how each of these

attempts to handle motions such as panning and zooming resulted in additional complexity. Both global motion compensation and Global Motion Vector Coding required more complex motion estimation operations to be done by encoders and also greater amounts of information to be included in encoded bitstreams. '808 Patent at 13:67-14:5 (“Although the related technique of global motion vector coding is computationally less demanding than global motion compensation, it does involve a certain increase in encoder complexity and additional information must be still transmitted in the video bit-stream to enable correct decoding of the video data.”); *see generally id.* at 12:59-14:5.

#### **4. The '808 Patent's Redefined “Skip Coding Mode” Solved a Problem in the Art**

The inventor of the '808 Patent recognized these limitations and set out to solve them by defining a new type of skip coding mode. As described below in the context of the claim term “skip coding mode,” the '808 Patent sought to provide improved coding efficiency with only a negligible increase in complexity “based on a redefinition of the skip mode concept used in JM1 of the JVT codec.” '808 Patent at 14:6-18. The '808 Patent provides the following definition of the redefined skip coding mode:

According to the invention, the coding modes used by encoder 600 correspond to those provided by JM1 of the JVT codec (shown in Table 3), with the exception that the SKIP mode is redefined to allow representation of global and regional motion. More specifically, the SKIP mode is modified in such a way that a zero (non-active) motion vector or a non-zero (active) motion vector is associated with each skip mode macroblock, depending on the characteristics of the motion in image segments surrounding the macroblock in question. In the following this type of motion vector will be referred to as a “skip mode motion vector”.

Unlike prior art skip coding modes, the '808 Patent's redefined skip coding mode is able to address both active and non-active motion scenarios. The '808 Patent's redefined skip coding mode can assign zero and predicted non-zero motion vectors to skip mode macroblocks in order

to reduce the amount of information required in a bitstream to address motions, such as when video frames have both non-active and active areas when there is camera or object movement. *See, e.g.*, '808 Patent at 17:43-53, 14:14-22.

**a. The Redefined *Skip* Coding Mode *Skips* Residual Information**

The '808 Patent explains that—just like in prior art skip coding modes—prediction error and residual information are skipped as part of skip coding mode: “[A]ccording to the invention, no additional information must be sent to the decoder in order to model global or regional motion.” '808 Patent at 14:62-64; Orchard Dec. at ¶¶ 20-25. Emphasizing the contrast between INTER-coding modes and the redefined skip coding mode according to the invention, the '808 Patent explains the following:

Depending on the coding mode, the compressed video data included in the bit-stream for an INTER-coded macroblock may comprise a combination of VLC encoded prediction error information for each block, motion vector information for the macroblock (or sub-blocks) and encoded control information including an indication of the coding mode used to encode the macroblock in question. If a macroblock is encoded in skip mode, no prediction error or motion vector information relating to the macroblock is included in the bit-stream.

'808 Patent at 20:43-55. In other words, additional information such as “prediction error information for each block” (residual information), *see* Orchard Dec. at ¶ 18, and “motion vector information” are included for INTER-coding modes *other than* the skip coding mode. For skip mode, “no prediction error or motion vector information relating to the macroblock is included in the bit-stream.” '808 Patent at 20:43-55.

**b. Advantages of the Redefined Skip Coding Mode**

One advantageous aspect of the '808 Patent is the redefined skip coding mode covered by the asserted claims allows for an analysis of the motion of “surrounding macroblocks or sub-blocks” for a given macroblock in order to determine whether active or non-active motion exists

and then, based on that analysis, determine a motion vector to be assigned. '808 Patent at 14:52-64. This is depicted in Figure 8 below.

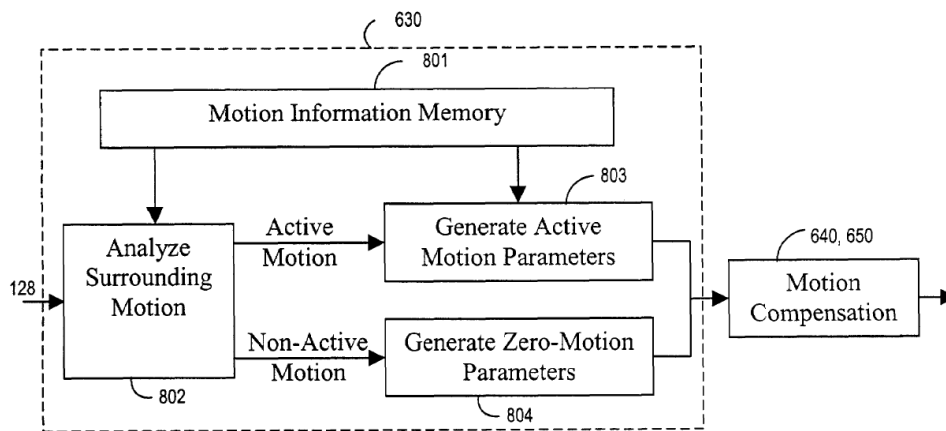


FIG. 8

Encoding and decoding for “skip mode macroblocks” in the '808 Patent requires analyzing surrounding motion in order to assign zero or non-zero motion vectors. An encoder uses a “motion estimation block” for these functions, '808 Patent at 18:28-33, and—if the '808 Patent’s redefined skip coding mode is the most efficient coding mode available—the encoder will include an indication of skip coding mode in an encoded bitstream. A decoder receives an indication of skip coding mode and uses a “motion compensated prediction block” to analyze and classify the “motion of previously decoded macroblocks and/or sub-blocks in a predefined region surrounding the macroblock to be decoded in a manner exactly corresponding to that used in encoder 600.” '808 Patent at 21:40-46. This duplication of “analysis and classification of motion in the surrounding region” for skip coding mode “means that according to the invention, no additional information must be sent to the decoder in order to model global or regional motion.” '808 Patent at 19:15-20. The '808 Patent also describes that the decision “to generate a zero valued skip mode motion vector or a non-zero valued skip mode motion vector” is made “by analysing and

classifying the motion of macroblocks to be coded using a predetermined analysis scheme.” ’808 Patent at 18:34-40. One embodiment of the ’808 Patent describes conditions under which surrounding motion may be classified as “non-active motion”:

In this embodiment the classification to the “non-active motion” category takes place if either or both of the two conditions below are true, otherwise the motion is classified as “active motion”:

Condition 1: The macroblock immediately above or the macroblock immediately to the left of the macroblock under consideration is not available (that is, is out of the picture or belongs to a different slice).

Condition 2: The macroblock or block immediately above, or the macroblock or block immediately to the left that are used in motion vector prediction for the 16×16 INTER mode has a zero motion vector and uses the latest picture as reference in motion compensation.

’808 Patent at 20:12-25.

A key benefit of the ’808 Patent’s redefined skip coding mode is that it obtains the significant compression savings of earlier skip coding modes even in active motion scenarios. *See, e.g.,* ’808 Patent at 14:62-64; Orchard. Dec. at ¶¶ 20-25. That is, “according to the invention, no additional information must be sent to the decoder in order to model global or regional motion,” and “a minimal amount of modification is required to incorporate the method according to the invention into existing video coding systems that employ the concept of skip mode macroblocks.” ’808 Patent at 14:62-64; 15:11-14; *see also id.* at 15:66-67, 19:17-20, and 20:53-55 (“If a macroblock is encoded in skip mode, no prediction error or motion vector information relating to the macroblock is included in the bit-stream.”). The ’808 Patent explains this point further:

It should be appreciated that by modifying the skip mode concept in the manner proposed by the invention and performing surrounding motion analysis in the decoder, it is possible to take account of global or region motion in a video sequence without requiring explicit information about such motion to be provided in video bit-stream.

'808 Patent at 21:57-62.

#### **D. The Disputed Claim Term**

##### **1. “skip coding mode”**

| <b>Claim Term</b>                                                                                    | <b>Nokia’s Proposed Construction</b>                                                                                                                                                                                                                               | <b>Staff’s Proposed Construction</b>                                                                                                                                                                                                                  | <b>Respondents’ Proposed Construction</b>                                                                                                                                                                                                                                                                                   |
|------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| “skip coding mode”<br><br>(claims 1, 6, 7, 9, 10, 15, 16, 20, 23, 25, 32, 34, 43, 44, 51-54, 62, 63) | “a coding mode in which a zero (non-active) motion vector or a non-zero (active) motion vector is associated with each skip mode macroblock, depending on the characteristics of the motion in image segments surrounding the macroblock in question” <sup>5</sup> | “a coding mode in which a zero (non-active) motion vector or a non-zero (active) motion vector is associated with each skip mode macroblock, depending on the characteristics of the motion in image segments surrounding the macroblock in question” | “a coding mode in which a zero (non-active) motion vector or a nonzero (active) motion vector is associate with each skip mode macroblock, depending on the characteristics of the motion in image segments surrounding the macroblock in question, and residual information can be provided for each skip mode macroblock” |

#### **IV. ARGUMENT**

The ALJ should adopt Nokia and the Staff’s proposed construction for three reasons. First, the inventor’s lexicography governs the meaning of a term. Second, claim differentiation is at most a rebuttable presumption, and Respondents’ proposed construction contradicts both the inventor’s lexicography and the specification’s repeated and consistent descriptions of the invention. Third, Respondents’ position misapplies claim differentiation.

---

<sup>5</sup> The '808 Patent was asserted in 337-TA-1208. In that investigation, the parties agreed to this same construction.

## A. The Inventor's Lexicography Governs

The '808 Patent provides the following clear and unambiguous definition of its redefined skip coding mode:

According to the invention, the coding modes used by encoder 600 correspond to those provided by JM1 of the JVT codec (shown in Table 3), with the exception that the SKIP mode is redefined to allow representation of global and regional motion. More specifically, the SKIP mode is modified in such a way that *a zero (non-active) motion vector or a non-zero (active) motion vector is associated with each skip mode macroblock, depending on the characteristics of the motion in image segments surrounding the macroblock in question.* In the following this type of motion vector will be referred to as a “skip mode motion vector”.

'808 Patent at 17:43-53 (emphasis added); *see also* 14:23-29 (“According to the invention, the skip mode concept is redefined...”). This “redefinition” is textbook lexicography. While “a claim term may be clearly redefined without an explicit statement of redefinition,” *Bell Atl. Network Servs., Inc. v. Covad Commc'ns Grp., Inc.*, 262 F.3d 1258, 1268 (Fed. Cir. 2001), the '808 Patent presents an easier case. The specification expressly says a *redefinition* of skip mode is provided.<sup>6</sup>

All parties ostensibly agree that column 17, lines 43-53, provides an express definition of the '808 Patent's redefined skip coding mode. “In such cases, the inventor's lexicography governs.” *Phillips*, 415 F.3d at 1316. Nokia and Staff's proposed construction is copied verbatim from this definition.<sup>7</sup>

The Respondents will likely endorse the '808 Patent's lexicography yet introduce their modification of that lexicography as a mere “addendum” based on claim differentiation. There are two significant problems with this theory. The first—and this is a big problem—is that the Respondents' “addendum” meaningfully alters the definition of skip coding mode away from the

---

<sup>6</sup> '808 Patent at 17:43-53; 14:65-15:2; 14:23-29; 14:16-18 (all using the words “redefined” or “redefinition”).

<sup>7</sup> Nokia and Staff's proposed construction adds “a coding mode in which” before the definition provided by the '808 Patent.

inventor's intended meaning while also contradicting the intrinsic record. The second—which is also a big problem—is that Respondents misapply claim differentiation to reach their proposed construction.

**B. Claim Differentiation Does Not Override Lexicography or the Teachings of the Specification**

Claim differentiation provides no basis to depart from the inventor's express definition. Lexicography, which “governs” claim construction, takes precedence over claim differentiation. *See Phillips*, 415 F.3d at 1316. Claim differentiation is merely “a rebuttable presumption that may be overcome by a contrary construction dictated by the written description or prosecution history.” *Howmedica*, 822 F.3d at 1324. Respondents' proposed construction of skip coding mode, which adds that the skip coding mode can provide “residual information...for each skip mode macroblock” contradicts the written description. A POSITA would understand based on the '808 Patent's unambiguous description of the redefined skip coding mode that no “residual information” is ever provided as part of the skip coding mode for any skip mode macroblock. Orchard Dec. at ¶ 25. Thus, any presumption from claim differentiation is rebutted (even assuming this is a situation where the independent claims require a limitation added by certain dependent claims, which—as explained below—this is not).

**1. Respondents' Construction Contradicts the Lexicography**

The first way in which Respondents' construction contradicts the written description comes from the inventors' lexicography itself. '808 Patent at 17:43-53. Respondents' proposed construction effectively removes the *skip* aspect from skip coding mode. The '808 Patent states that, “According to the invention, the coding modes used by encoder 600 correspond to those provided by JM1 of the JVT codec (shown in Table 3), with the exception that the SKIP mode is redefined to allow representation of global and regional motion.” By contrasting the redefinition

of “skip coding mode” with the JM1 of the JVT definition, the inventor clarified the lexicographical starting point, and “modified” the JM1 definition. ’808 Patent at 17:47-48. JM1, which is cited on the cover page of the ’808 Patent, specifies that no residual information is sent as part of skip coding mode for a skip mode macroblock. Ex. 1, NOK\_AMHP\_ITC00079643 at 79650; Orchard Dec. at ¶ 19.

As Dr. Orchard explains in his declaration, a POSITA would appreciate that the inventor of the ’808 Patent sought to replace JM1’s SKIP mode with a skip coding mode that can address both non-active and active motion scenarios. Orchard Dec. at ¶¶ 15, 26-28. However, a POSITA would not understand the inventor to be replacing JM1’s SKIP mode with a *skip* coding mode that does not itself *skip* “the coding transmission and decoding of residual information” for skip mode macroblock. Orchard Dec. at ¶¶ 26-28. Indeed, the ’808 Patent describes its modifications to JM1’s SKIP mode as relating to how skip mode motion vectors are assigned to skip mode macroblocks. But the ’808 Patent nowhere mentions any modification of the JM1’s SKIP mode involving the processing of any residual information after motion compensation. Orchard Dec. at ¶ 22. Indeed, the ’808 Patent touts the fact that “a minimal amount of modification is required to incorporate the method according to the invention into existing video coding systems that employ the concept of skip mode macroblocks.” ’808 Patent at 15:11-14; *see also* ’808 Patent at 21:57-62 (further explaining that the modified skip coding mode does not require “explicit information about such motion to be provided in video bit-stream” for skip mode macroblocks).

The underlying premise of Respondents’ construction—that *skip* coding mode could actually include residual information (rather than *skip* those related processes)—would be an oxymoron to a person of skill in the art. Orchard Dec. at ¶ 26.

## **2. Respondents' Construction Contradicts the Repeated and Consistent Descriptions of Invention in the Specification**

The second way in which Respondents' construction contradicts the written description comes from the '808 Patent's repeated and consistent descriptions of the invention. Dr. Orchard's declaration explains this point in detail by reference to the redefined skip coding mode's "non-active" and "active" motion scenarios. Orchard Dec. at ¶¶ 20-25.

For non-active motion scenarios, a POSITA would readily appreciate "that residual information can never be provided" in the redefined skip coding mode. Orchard Dec. at ¶ 20. This is because, in the "non-active" scenario, the skip coding mode directly assigns a zero motion vector to the skip mode macroblock, which means the skip mode macroblock is "copied directly from the reference frame." '808 Patent at 14:42-48, 18:6-11, and 21:50-53. A POSITA would understand from these disclosures that providing residual information as part of the redefined skip coding mode would be nonsensical in these non-active scenarios. *See* Orchard Dec. at ¶ 20; *see also* '808 Patent at 3:28-39 (describing the purpose of "prediction error").

For active motion scenarios, the '808 Patent repeatedly states that "no additional information" needs to be transmitted as part of the skip coding mode for reconstructing the pixel values of skip mode macroblocks. Orchard Dec. at ¶ 23 (citing '808 Patent at 15:63-67). For example, the '808 Patent states, "according to the invention, no additional information must be sent to the decoder in order to model global or regional motion." '808 Patent at 14:62-64; *see also id.* at 19:17-20. The '808 Patent explains this lack of additional information in the redefined skip coding mode is a benefit over "both the global motion compensation and global motion vector coding" prior art techniques. '808 Patent at 15:7-11. As Dr. Orchard explains in his declaration, "no additional information" means what it says: A POSITA would know this "precludes the

possibility that, in skip coding mode, any ‘residual information’ can be provided for the reconstruction of pixel values of the macroblock.” Orchard Dec. at ¶ 23.

Moreover, in an active motion scenario, the ’808 Patent explains that a skip mode macroblock will have its pixel values generated based on “a region of pixel values indicated by the non-zero motion vector.” ’808 Patent at 21:50-56. As Dr. Orchard explains, a POSITA would understand this disclosure to teach that “pixel values of ‘active’ skip mode macroblocks are generated directly from a region of pixel values from a reference frame pointed to by the non-zero motion vector that is calculated.” Orchard Dec. at ¶ 24. In the redefined skip coding mode, no “residual information” is provided for these active skip mode macroblocks. *Id.*

Indeed, the ’808 Patent expressly juxtaposes INTER-coding modes with prediction error (residual information) from the redefined skip coding mode, which does not include such additional information:

Depending on the coding mode, the compressed video data included in the bit-stream for an INTER-coded macroblock may comprise a combination of VLC encoded prediction error information for each block, motion vector information for the macroblock (or sub-blocks) and encoded control information including an indication of the coding mode used to encode the macroblock in question. If a macroblock is encoded in skip mode, no prediction error or motion vector information relating to the macroblock is included in the bit-stream.

’808 Patent at 20:43-55.

As Dr. Orchard concludes, the opposite of Respondents’ construction is more correct. Within skip coding mode, “residual information can never be provided for any skip mode macroblock.” Orchard Dec. at ¶ 25.

### **C. Respondents Misapply Claim Differentiation**

Nokia expects Respondents will point to dependent claims 30, 41, 49, and 60 as the basis for their proposed construction. Nokia further expects the Respondents will argue that these

dependent claims will be rendered superfluous unless the skip coding mode of the '808 Patent includes residual information. This is incorrect. The problem with Respondents' theory is these dependent claims do not add a limitation already required by other claims of the '808 Patent.

Claims 30, 41, 49, and 60 add the limitation “wherein no residual information is provided for the first segment *in the encoded bitstream*.” Thus, the threshold question for Respondents' position is whether this same limitation is added to the other claims of the '808 Patent by virtue of the redefined skip coding mode.

It is not. As explained at length in the specification, the '808 Patent's *skip coding mode* does not provide residual information for any skip mode macroblock. This is a distinct proposition from whether residual information is not provided for a skip mode macroblock *anywhere in an encoded bitstream*. Put differently, a restriction that “no residual information is provided...in the encoded bitstream” is a narrowing limitation beyond the fact that skip coding mode itself does not include residual information. *See Edwards Lifesciences LLC*, 582 F.3d at 1331 (holding a claim differentiation argument failed because “the dependent claims are narrower”). The basic, common-sense premise behind this conclusion is that the “skip coding mode” is *not the same* as “the encoded bitstream.” An indication of skip coding mode is only one small potential part of a bitstream. The encoded bitstream refers to the stream of data used to represent a video sequence, which can include information relating to any number of processes beyond skip coding mode.<sup>8</sup> By contrast, the skip coding mode is a coding mode that applies to particular macroblocks.

---

<sup>8</sup> *See* '808 Patent at 3:47-52 (a decoder uses the video bit-stream to produce a reconstructed video sequence); *see also, e.g.*, '808 Patent at 4:7-12 (a “video multiplex coder...forms a single bit-stream from the various types of information produced by the encoder...”), 5:23-28 (a video multiplex coder combines values that have been entropy coded with control information).

Thus, this is not a situation where “skip coding mode” in the independent claims requires the same limitation added by dependent claims 30, 41, 49, and 60 as Respondents appear to allege.

In *Edwards Lifesciences*, a party argued that—despite statements in the specification to the contrary—because certain dependent claims of a patent required “wires” and other claims did not, the claimed devices did not include wires. *Edwards Lifesciences LLC*, 582 F.3d at 1331. However, the Federal Circuit determined this claim differentiation argument failed because (1) claim differentiation “does not trump the clear import of the specification,” and (2) the dependent claims that added “wire limitations also add further limitations, so the dependent claims are narrower not only because of the addition of wires.” *Id.* Similar reasoning compels the same conclusion here. That the entire encoded bitstream does not contain residual information for the first segment is a different and narrower limitation than the fact the ’808 Patent’s skip coding mode itself does not include residual information for any skip mode macroblock.

The Federal Circuit’s reasoning in *Curtiss-Wright* is also instructive. In *Curtiss-Wright*, certain claims of a patent included the term “adjustment mechanism.” *Curtiss-Wright Flow Control Corp*, 438 F.3d at 1381; *see also* U.S. Pat. No. 6,565,714 (claims 1 and 18). A different claim included the term “an adjustable dynamic.” *Id.*; *see also* U.S. Pat. No. 6,565,714 (claim 14). A district court “relied on claim differentiation” in construing “adjustable” before issuing a preliminary injunction. In vacating and remanding, the Federal Circuit held the “adjustable” term did not render the “adjustment mechanism” term superfluous, pointing out that “in-use adjustability does not necessarily mean the same thing as the presence of an adjustment mechanism. After all, an adjustment mechanism might be present but not useable during de-heading.” *Id.* A similar principle applies here, albeit perhaps more straightforward: That the ’808 Patent’s *skip* coding mode *skips* processes associated with residual information is not the same as

a narrowing limitation that no residual information can be included anywhere in an encoded bitstream for a given macroblock.

## **V. CONCLUSION**

For the foregoing reasons, the ALJ should adopt Nokia and Staff's proposed construction of "skip coding mode," which is copied from the inventor's lexicography.

### **CERTIFICATE OF WORD COUNT COMPLIANCE**

Pursuant to Ground Rule 1.7 the undersigned hereby certifies that the forgoing brief complies with the word-count limitation, and contains 6,405 words according to the word-processing program used to prepare this document (Microsoft Word). The word count includes footnotes. The word count does not include captions, table of contents, table of authorities, words in figures, the word count certificate, the signature block, or the certificate of service.

Dated: February 13, 2024

Respectfully submitted,

*/s/ Joshua Newcomer*

---

Adam D. Swain

**Alston & Bird LLP**

950 F Street NW

Washington, DC 20004

Telephone: (202) 239-3300

Facsimile: (202) 239-3333

John D. Haynes

Nicholas T. Tsui

Lindsay Church

Shawn Gannon

**Alston & Bird LLP**

1201 West Peachtree Street

Atlanta, GA 30309

Telephone: (404) 881-7000

Facsimile: (404) 881-7777

Theodore Stevenson

**Alston & Bird LLP**

2200 Ross Avenue, Suite 2300

Dallas, TX 75201

Phone: (214) 922-3400

Fax: (214) 922-3899

Warren Lipschitz

wlipschitz@mckoolsmith.com

**McKool Smith P.C.**

300 Crescent Court, Suite 1500

Dallas, Texas 75201

Telephone: (214) 978-4000

Facsimile: (214) 978-4044

R. Mitch Verboncoeur

mverboncoeur@mckoolsmith.com

**McKool Smith P.C.**

303 Colorado, Suite 2100

Austin, Texas 78701

Telephone: (512) 692-8700

Facsimile: (214) 978-4044

Josh Newcomer  
jnewcomer@mckoolsmith.com  
**McKool Smith P.C.**  
600 Travis St., Suite 7000  
Houston, Texas 77002  
Telephone: (713) 485-7300  
Facsimile: (713) 485-7344

*Counsel for Complainants  
Nokia Technologies Oy and Nokia  
Corp.*

# **EXHIBIT 1**

**Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG**

Document **JVT-A003**

File: JVT-A003.doc

Generated: 2002-01-18

Pattaya, Thailand, 3-7 December 2001

*Title:* Joint Model Number 1 (JM-1)

*Status:* Approved Output Document

*Contact:* Thomas Wiegand  
Heinrich Hertz Institute (HHI), Einsteinufer 37, D-10587 Berlin, Germany  
Tel: +49 - 30 - 31002 617, Fax: +49 - 030 - 392 72 00, [wiegand@hhi.de](mailto:wiegand@hhi.de)

*Purpose:* Information

---

This document presents the Joint Model Number 1 of the Joint Video Team formed by ITU-T SG16 Q.6 (VCEG) and ISO/IEC JTC 1/SC 29/WG 11 (MPEG). It contains various corrections and improvements relative to TML-9 based on suggestions from the JVT experts reflector. Document JVT-A003a contains a list of unresolved issues.



## CONTENTS

|           |                                                              |    |
|-----------|--------------------------------------------------------------|----|
| 1         | Scope.....                                                   | 7  |
| 2         | Source Coder .....                                           | 8  |
| 2.1       | Picture formats.....                                         | 8  |
| 2.2       | Subdivision of a picture into macroblocks .....              | 8  |
| 2.3       | Order of the bitstream within a macroblock .....             | 8  |
| 3         | Syntax and Semantics .....                                   | 10 |
| 3.1       | Organization of syntax elements (NAL concept) .....          | 10 |
| 3.1.1     | Slice Mode.....                                              | 10 |
| 3.1.2     | Data Partitioning Mode .....                                 | 10 |
| 3.2       | Syntax diagrams .....                                        | 10 |
| 3.3       | Slice Layer .....                                            | 11 |
| 3.3.1     | Picture sync .....                                           | 12 |
| 3.3.2     | Picture type (Ptype).....                                    | 12 |
| 3.3.3     | Picture Number (PN).....                                     | 12 |
| 3.3.4     | Reference Picture Selection Layer (RPSL).....                | 13 |
| 3.3.5     | Re-Mapping of Picture Numbers Indicator (RMPNI).....         | 13 |
| 3.3.5.1   | Absolute Difference of Picture Numbers (ADPN).....           | 14 |
| 3.3.5.2   | Long-term Picture Index for Re-Mapping (LPIR).....           | 15 |
| 3.3.6     | Reference Picture Buffering Type (RPBT).....                 | 15 |
| 3.3.6.1   | Memory Management Control Operation (MMCO) .....             | 15 |
| 3.3.6.2   | Difference of Picture Numbers (DPN) .....                    | 16 |
| 3.3.6.3   | Long-term Picture Index (LPIN) .....                         | 16 |
| 3.3.6.4   | Maximum Long-Term Picture Index Plus 1 (MLIP1).....          | 16 |
| 3.3.7     | Supplemental Enhancement Information .....                   | 17 |
| 3.4       | Macroblock layer .....                                       | 17 |
| 3.4.1     | Number of Skipped Macroblocks (RUN) .....                    | 17 |
| 3.4.2     | Macro block type (MB_Type).....                              | 17 |
| 3.4.2.1   | Intra Macroblock modes .....                                 | 17 |
| 3.4.2.2   | Inter Macroblock Modes.....                                  | 17 |
| 3.4.3     | Intra Prediction Mode (Intra_pred_mode) .....                | 18 |
| 3.4.3.1.1 | Prediction for Intra-4x4 mode for luminance blocks.....      | 18 |
| 3.4.3.1.2 | Prediction for Intra-16x16 mode for luminance .....          | 20 |
| 3.4.3.1.3 | Prediction in intra coding of chrominance blocks .....       | 21 |
| 3.4.4     | Reference frame (Ref_frame).....                             | 22 |
| 3.4.5     | Motion Vector Data (MVD).....                                | 22 |
| 3.4.5.1   | Prediction of vector components .....                        | 22 |
| 3.4.5.1.1 | Median prediction .....                                      | 22 |
| 3.4.5.1.2 | Directional segmentation prediction .....                    | 23 |
| 3.4.5.2   | Chrominance vectors .....                                    | 23 |
| 3.4.6     | Coded Block Pattern (CBP) .....                              | 24 |
| 3.4.7     | Change of Quantizer Value (Dquant).....                      | 24 |
| 4         | Decoding Process.....                                        | 25 |
| 4.1       | Slice Decoding.....                                          | 25 |
| 4.1.1     | Multi-Picture Decoder Process.....                           | 25 |
| 4.1.1.1   | Decoder Process for Short/Long-term Picture Management ..... | 25 |
| 4.1.1.2   | Decoder Process for Reference Picture Buffer Mapping.....    | 26 |
| 4.1.1.3   | Decoder Process for Multi-Picture Motion Compensation.....   | 26 |
| 4.1.1.4   | Decoder Process for Reference Picture Buffering .....        | 27 |
| 4.2       | Motion Compensation .....                                    | 27 |
| 4.2.1     | Fractional pixel accuracy .....                              | 27 |
| 4.2.1.1   | 1/4 luminance sample interpolation .....                     | 27 |
| 4.2.1.2   | 1/8 Pel Luminance Samples Interpolation .....                | 28 |
| 4.3       | Transform Coefficient Decoding.....                          | 29 |

|             |                                                                          |    |
|-------------|--------------------------------------------------------------------------|----|
| 4.3.1       | Transform for Blocks of Size 4x4 Samples .....                           | 29 |
| 4.3.2       | Transform for Blocks of Size 2x2 Samples (DC vales for Chrominance)..... | 29 |
| 4.3.3       | Zig-zag Scanning and Quantization .....                                  | 30 |
| 4.3.3.1     | Simple Zig-Zag Scan .....                                                | 30 |
| 4.3.3.2     | Double Zig-Zag Scan.....                                                 | 30 |
| 4.3.3.3     | Quantization .....                                                       | 30 |
| 4.3.3.4     | Scanning and Quantization of 2x2 chrominance DC coefficients .....       | 31 |
| 4.3.4       | Use of 2-dimensional model for coefficient coding.....                   | 31 |
| 4.4         | Deblocking Filter .....                                                  | 31 |
| 4.4.1       | Picture Content Dependent Parameters for Filtering .....                 | 32 |
| 4.4.2       | Filtering Process.....                                                   | 33 |
| 4.4.3       | Stronger filtering for intra coded macroblocks .....                     | 33 |
| 4.5         | Entropy Coding.....                                                      | 34 |
| 4.5.1       | Universal Variable Length Coding (UVLC).....                             | 34 |
| 4.5.2       | Context-based Adaptive Binary Arithmetic Coding (CABAC).....             | 36 |
| 4.5.2.1     | Overview .....                                                           | 36 |
| 4.5.2.2     | Context Modeling for Coding of Motion and Mode Information.....          | 36 |
| 4.5.2.2.1   | Context Models for Macroblock Type.....                                  | 36 |
| 4.5.2.2.1.1 | Intra Pictures.....                                                      | 37 |
| 4.5.2.2.1.2 | P- and B-Pictures .....                                                  | 37 |
| 4.5.2.2.2   | Context Models for Motion Vector Data .....                              | 38 |
| 4.5.2.2.3   | Context Models for Reference Frame Parameter .....                       | 38 |
| 4.5.2.3     | Context Modeling for Coding of Texture Information .....                 | 38 |
| 4.5.2.3.1   | Context Models for Coded Block Pattern.....                              | 38 |
| 4.5.2.3.2   | Context Models for Intra Prediction Mode.....                            | 39 |
| 4.5.2.3.3   | Context Models for Run/Level .....                                       | 39 |
| 4.5.2.3.3.1 | Context-based Coding of LEVEL Information .....                          | 39 |
| 4.5.2.3.3.2 | Context-based Coding of RUN Information .....                            | 40 |
| 4.5.2.4     | Context Modeling for Coding of Dquant .....                              | 40 |
| 4.5.2.5     | Binarization of Non-Binary Valued Symbols.....                           | 40 |
| 4.5.2.6     | Adaptive Binary Arithmetic Coding .....                                  | 40 |
| 5           | B-pictures.....                                                          | 41 |
| 5.1         | Introduction .....                                                       | 41 |
| 5.2         | Five Prediction modes .....                                              | 41 |
| 5.3         | Syntax .....                                                             | 42 |
| 5.3.1       | Picture type (Ptype) and RUN.....                                        | 42 |
| 5.3.2       | Macro block type (MB_type).....                                          | 42 |
| 5.3.3       | Intra prediction mode (Intra_pred_mode) .....                            | 43 |
| 5.3.4       | Reference Frame (Ref_frame).....                                         | 43 |
| 5.3.5       | Block Size (Blk_size).....                                               | 44 |
| 5.3.6       | Motion vector data (MVDFW, MVDBW).....                                   | 44 |
| 5.4         | Decoder Process for motion vector .....                                  | 44 |
| 5.4.1       | Differential motion vectors .....                                        | 44 |
| 5.4.2       | Motion vectors in direct mode .....                                      | 45 |
| 6           | SP-pictures.....                                                         | 46 |
| 6.1         | Introduction .....                                                       | 46 |
| 6.2         | Syntax changes .....                                                     | 46 |
| 6.3         | SP-frame decoding .....                                                  | 46 |
| 7           | Hypothetical Reference Decoder.....                                      | 48 |
| 7.1         | Purpose .....                                                            | 48 |
| 7.2         | Operation of the HRD.....                                                | 48 |
| 7.3         | Decoding Time of a Picture .....                                         | 48 |
| 7.4         | Schedule of a Bit Stream .....                                           | 48 |
| 7.5         | Containment in a Leaky Bucket .....                                      | 48 |

|              |                                                                      |    |
|--------------|----------------------------------------------------------------------|----|
| 7.6          | Bit Stream Syntax .....                                              | 49 |
| 7.7          | Minimum Buffer Size and Minimum Peak Rate .....                      | 49 |
| 7.8          | Encoder Considerations (informative).....                            | 50 |
| Appendix I   | .....Non-normative Encoder Recommendation                            | 52 |
| 1.1          | Motion Estimation and Mode Decision.....                             | 52 |
| 1.1.1        | Low-complexity mode .....                                            | 52 |
| 1.1.1.1      | Finding optimum prediction mode.....                                 | 52 |
| 1.1.1.1.1    | SA(T)D0 .....                                                        | 52 |
| 1.1.1.1.2    | Block_difference.....                                                | 53 |
| 1.1.1.1.3    | Hadamard transform .....                                             | 53 |
| 1.1.1.1.4    | Mode decision.....                                                   | 53 |
| 1.1.1.2      | Encoding on macroblock level.....                                    | 53 |
| 1.1.1.2.1    | Intra coding .....                                                   | 53 |
| 1.1.1.2.2    | Table for intra prediction modes to be used at the encoder side..... | 53 |
| 1.1.1.2.3    | Inter mode selection.....                                            | 54 |
| 1.1.1.2.4    | Integer pixel search.....                                            | 54 |
| 1.1.1.2.5    | Fractional pixel search .....                                        | 54 |
| 1.1.1.2.6    | Decision between intra and inter .....                               | 55 |
| 1.1.2        | High-complexity mode.....                                            | 55 |
| 1.1.2.1      | Motion Estimation .....                                              | 55 |
| 1.1.2.1.1    | Integer-pixel search.....                                            | 55 |
| 1.1.2.1.2    | Fractional pixel search.....                                         | 55 |
| 1.1.2.1.3    | Finding the best motion vector .....                                 | 55 |
| 1.1.2.1.4    | Finding the best reference frame .....                               | 56 |
| 1.1.2.2      | Mode decision.....                                                   | 56 |
| 1.1.2.2.1    | Macroblock mode decision.....                                        | 56 |
| 1.1.2.2.2    | INTER 16x16 mode decision .....                                      | 57 |
| 1.1.2.2.3    | INTER 4x4 mode decision .....                                        | 57 |
| 1.1.2.3      | Algorithm for motion estimation and mode decision .....              | 57 |
| 1.2          | Quantization.....                                                    | 58 |
| 1.3          | Elimination of single coefficients in inter macroblocks.....         | 58 |
| 1.3.1        | Luma.....                                                            | 58 |
| 1.3.2        | Chroma.....                                                          | 58 |
| 1.4          | Encoding with Anticipation of Slice Losses.....                      | 58 |
| Appendix II  | .....Network Adaptation Layer for IP networks                        | 60 |
| II.1         | Assumptions .....                                                    | 60 |
| II.2         | Combining of Partitions according to Priorities .....                | 60 |
| II.3         | Packet Structure .....                                               | 61 |
| II.4         | Packetization Process .....                                          | 61 |
| II.5         | De-packetization.....                                                | 62 |
| II.6         | Repair and Error Concealment .....                                   | 62 |
| Appendix III | .....Interim File Format                                             | 63 |
| III.1        | General.....                                                         | 63 |
| III.2        | File Identification .....                                            | 63 |
| III.3        | Clump .....                                                          | 63 |
| III.3.1      | Definition .....                                                     | 63 |
| III.3.1.1    | Syntax .....                                                         | 63 |
| III.3.1.2    | Semantics .....                                                      | 64 |
| III.4        | Clump Order.....                                                     | 64 |
| III.5        | Clump Definitions .....                                              | 64 |
| III.5.1      | File Type Clump.....                                                 | 64 |

|             |                                                 |    |
|-------------|-------------------------------------------------|----|
| III.5.1.1   | Definition .....                                | 64 |
| III.5.1.2   | Syntax .....                                    | 65 |
| III.5.1.3   | Semantics .....                                 | 65 |
| III.5.2     | File Header Clump .....                         | 65 |
| III.5.2.1   | Definition .....                                | 65 |
| III.5.2.2   | Syntax .....                                    | 65 |
| III.5.2.3   | Semantics .....                                 | 66 |
| III.5.3     | Content Info Clump.....                         | 66 |
| III.5.3.1   | Definition .....                                | 66 |
| III.5.3.2   | Syntax .....                                    | 66 |
| III.5.3.3   | Semantics .....                                 | 67 |
| III.5.4     | Alternate Track Info Clump .....                | 67 |
| III.5.4.1   | Definition .....                                | 67 |
| III.5.4.2   | Syntax .....                                    | 68 |
| III.5.4.3   | Semantics .....                                 | 68 |
| III.5.5     | Parameter Set Clump.....                        | 68 |
| III.5.5.1   | Definition .....                                | 68 |
| III.5.5.2   | Syntax .....                                    | 68 |
| III.5.5.3   | Semantics .....                                 | 69 |
| III.5.6     | Segment Clump .....                             | 70 |
| III.5.6.1   | Definition .....                                | 70 |
| III.5.6.2   | Syntax .....                                    | 70 |
| III.5.6.3   | Semantics .....                                 | 70 |
| III.5.7     | Alternate Track Header Clump .....              | 71 |
| III.5.7.1   | Definition .....                                | 71 |
| III.5.7.2   | Syntax .....                                    | 71 |
| III.5.7.3   | Semantics .....                                 | 72 |
| III.5.8     | Alternate Track Media Clump.....                | 72 |
| III.5.8.1   | Definition .....                                | 72 |
| III.5.8.2   | Syntax .....                                    | 73 |
| III.5.9     | Switch Picture Clump.....                       | 73 |
| III.5.9.1   | Definition .....                                | 73 |
| III.5.9.2   | Syntax .....                                    | 73 |
| III.5.9.3   | Semantics .....                                 | 74 |
| Appendix IV | .....Non-Normative Error Concealment            | 75 |
| IV.1        | Introduction .....                              | 75 |
| IV.2        | INTRA Frame Concealment.....                    | 75 |
| IV.3        | INTER and SP Frame Concealment.....             | 76 |
| IV.3.1      | General .....                                   | 76 |
| IV.3.2      | Concealment using motion vector prediction..... | 77 |
| IV.3.3      | Handling of Multiple reference frames .....     | 78 |
| IV.4        | B Frame Concealment .....                       | 78 |
| IV.5        | Handling of Entire Frame Losses .....           | 78 |

# **1 Scope**

This document is a description of a reference coding method to be used for the development of a new video compression ITU-T Recommendation (H.26L) and ISO Standard (MPEG-4, Part 10). The basic configuration of the algorithm is similar to H.263 and MPEG-4, Part 2.

## 2 Source Coder

### 2.1 Picture formats

The image width and height of the source data are restricted to be multiples of 16. At the moment only colour sequences using 4:2:0 chrominance sub-sampling are supported.

### 2.2 Subdivision of a picture into macroblocks

Pictures are divided into macroblocks of 16x16 pixels. For instance, a QCIF picture is divided into 99 macroblocks as indicated in FIGURE 1. A number of consecutive macroblocks in coding order (see FIGURE 1) can be organized in slices. Slices represent independent coding units in a way that they can be decoded without referencing other slices of the same frame.

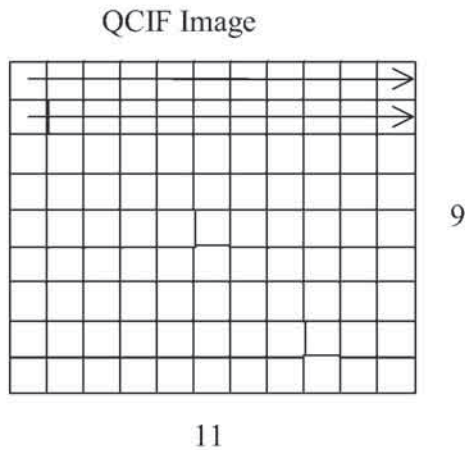


FIGURE 1

### Subdivision of a QCIF picture into 16x16 macroblocks

### 2.3 Order of the bitstream within a macroblock

FIGURE 2 and FIGURE 3 indicate how a macroblock is divided and the order of the different syntax elements resulting from coding a macroblock.

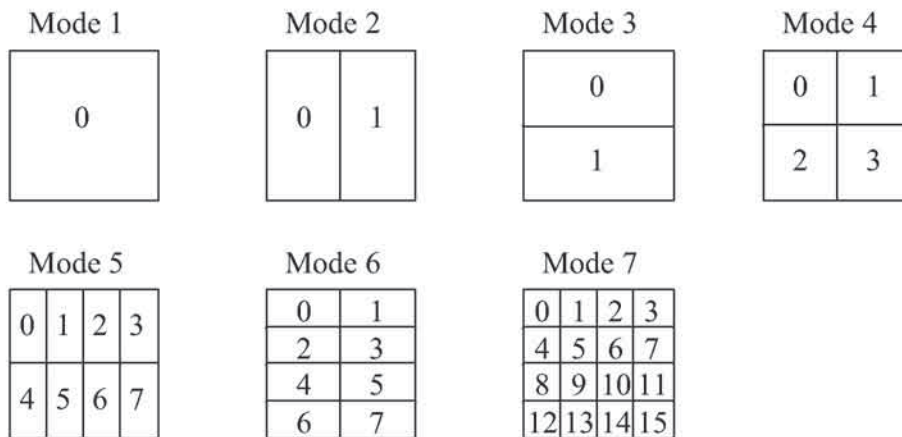


FIGURE 2

**Numbering of the vectors for the different blocks depending on the inter mode. For each block the horizontal component comes first followed by the vertical component**

CBPY 8x8 block order

|   |   |
|---|---|
| 0 | 1 |
| 2 | 3 |

Luma residual coding 4x4 block order

|    |    |    |    |
|----|----|----|----|
| 0  | 1  | 4  | 5  |
| 2  | 3  | 6  | 7  |
| 8  | 9  | 12 | 13 |
| 10 | 11 | 14 | 15 |

Chroma residual coding 4x4 block order

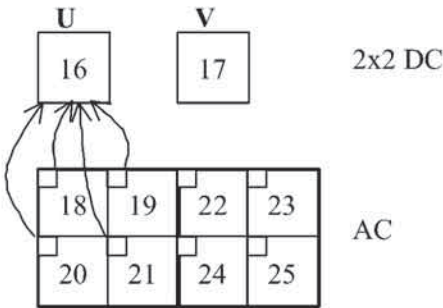


FIGURE 3

**Ordering of blocks for CBBPY and residual coding of 4x4 blocks**

### 3 Syntax and Semantics

#### 3.1 Organization of syntax elements (NAL concept)

Tbd.

##### 3.1.1 Slice Mode

Tbd.

##### 3.1.2 Data Partitioning Mode

Data Partitioning re-arranges the symbols in a way that all symbols of one data type (e.g. DC coefficients, macroblock headers, motion vectors) that belong to a single slice are collected in one VLC coded bitstream that starts byte aligned. Decoders can process such a partitioned data streams by fetching symbols from the correct partition. The partition to fetch from is determined through the decoder's state machine, according to the syntax diagram discussed in section 3.4.

Data Partitioning is implemented by concatenating all VLC coded symbols of one data type and one slice (or full picture if slices are not used). At the moment, for a few partitions as indicated below contain data of more than one data type that are so closely related that a finer diversion seems to be fruitless. The following data types are currently defined:

|   |               |                                        |
|---|---------------|----------------------------------------|
| 0 | TYPE_HEADER   | Picture or Slice Headers (Note 1)      |
| 1 | TYPE_MBHEADER | Macroblock header information (Note 2) |
| 2 | TYPE_MVD      | Motion Vector Data                     |
| 3 | TYPE_CBP      | Coded Block Pattern                    |
| 4 | TYPE_2x2DC    | 2x2 DC Coefficients                    |
| 5 | TYPE_COEFF_Y  | Luminance AC Coefficients              |
| 6 | TYPE_COEFF_C  | Chrominance AC Coefficients            |
| 7 | TYPE_EOS      | End-of-Stream Symbol                   |

Note 1: TYPE\_HEADER encompasses all Picture/Slice header information

Note 2: TYPE\_MBHEADER encompasses The MB-Type, Intra-Prediction mode and Reference Frame ID.

#### 3.2 Syntax diagrams

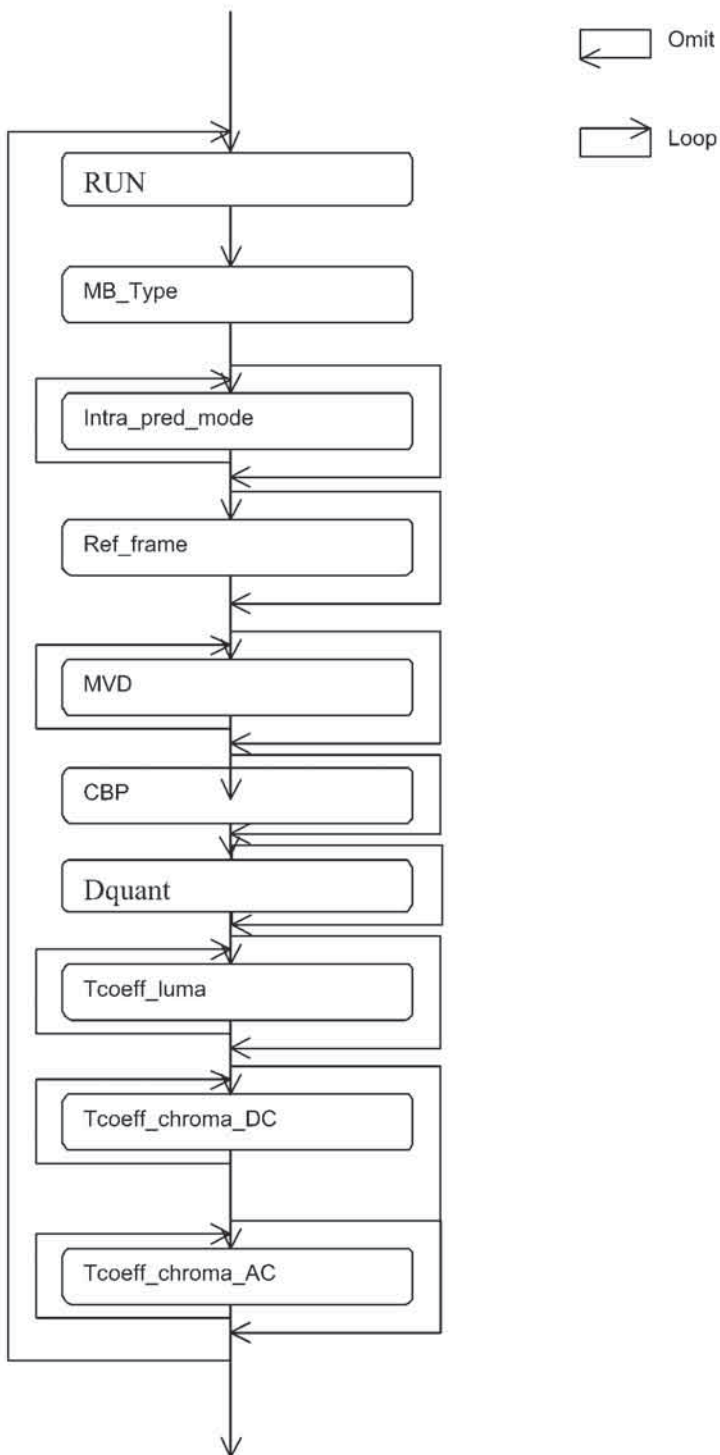


FIGURE 4

Syntax diagram for all the elements in the macroblock layer

### 3.3 Slice Layer

Editor: The slice layer needs a significant amount of work together with the NAL concept. The Picture sync and picture type codewords are present to ease the VCL development.

### 3.3.1 Picture sync

The first codeword in a picture is 31 bits long ( $L = 31$ ). It works as a picture sync, but it also contains an INFO part that has 15 bits. These bits are used for:

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TR(8 bits)  | Temporal reference. The value of TR is formed by incrementing its value in the temporally-previous reference picture header by one plus the number of skipped or non-reference pictures at the picture clock frequency since the previously transmitted one.                                                                                                                                                                                                                                                                                                                                                                                          |
| PQP(5 bits) | Picture QP: Information about the quantizer QUANT to be used for Luminance for the picture. (See under Quantization concerning QUANT for chrominance). The 5 bit representation is the natural binary representations of the values of QP which range from 0 to 31. QP is a pointer to the actual quantization parameter QUANT to be used. (See below under quantization). The range of quantization value is still about the same as for H.263, 1-31. An approximate relation between the QUANT in H.263 and QP is: $QUANT_{H.263}(QP) \approx QP_0(QP) = 2QP/6$ . $QP_0()$ will be used later for scaling purposes when selecting prediction modes. |
| Formats     | 0 indicates QCIF, 1 indicates CIF                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| EOS         | 0 for picture header. 1 indicates End Of Sequence                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

The sketch below indicates where the different bits are located within INFO.

|        |         |             |         |
|--------|---------|-------------|---------|
| 8b: TR | 5b: PQP | 1b: Formats | 1b: EOS |
|--------|---------|-------------|---------|

FIGURE 5

#### Syntax diagram for the Picture header.

### 3.3.2 Picture type (Ptype)

Code\_number=0: Inter picture with prediction from the most recent decoded picture only.

Code\_number=1: Inter picture with possibility of prediction from more than one previous decoded picture. For this mode information reference picture for prediction must be signalled for each macroblock.

Code\_number=2: Intra picture.

Code\_number=3: B picture with prediction from the most recent previous decoded and subsequent decoded pictures only.

Code\_number=4: B picture with possibility of prediction from more than one previous decoded picture and subsequent decoded picture. When using this mode, information reference frame for prediction must be signalled for each macroblock.

### 3.3.3 Picture Number (PN)

PN shall be incremented by 1 for each coded and transmitted picture, in modulo  $MAX\_PN$  operation, relative to the PN of the previous stored picture. For non-stored pictures, PN shall be incremented from the value in the most temporally recent stored picture, which precedes the non-stored picture in bitstream order.

The PN serves as a unique ID for each picture stored in the multi-picture buffer within  $MAX\_PN$  coded and stored pictures. Therefore, a picture cannot be kept in the buffer after more than  $MAX\_PN-1$  subsequent coded and stored pictures unless it has been assigned a long-term picture index as specified below. The encoder shall ensure that the bitstream shall not specify retaining any short-term picture after more than  $MAX\_PN-1$  subsequent stored pictures. A decoder which encounters a picture number on a current picture having a value equal to the picture number of some other short-term stored picture in the multi-picture buffer should treat this condition as an error.

### 3.3.4 Reference Picture Selection Layer (RPSL)

RPSL can be signaled with the following values:

Code number 0: The RPS layer is not sent,

Code number 1: The RPS layer is sent.

If RPSL is not sent, the default buffer indexing order presented in the next subsection shall be applied. RPS layer information sent at the slice level does not affect the decoding process of any other slice.

If RPSL is sent, the buffer indexing used to decode the current slice and to manage the contents of the picture buffer is sent using the following code words.

### 3.3.5 Re-Mapping of Picture Numbers Indicator (RMPNI)

RMPNI is present in the RPS layer if the picture is a P or B picture. RMPNI indicates whether any default picture indices are to be re-mapped for motion compensation of the current slice – and how the re-mapping of the relative indices into the multi-picture buffer is to be specified if indicated. If RMPNI indicates the presence of an ADPN or LPIR field, an additional RMPNI field immediately follows the ADPN or LPIR field.

A picture reference parameter is a relative index into the ordered set of pictures. The RMPNI, ADPN, and LPIR fields allow the order of that relative indexing into the multi-picture buffer to be temporarily altered from the default index order for the decoding of a particular slice. The default index order is for the short-term pictures (i.e., pictures which have not been given a long-term index) to precede the long-term pictures in the reference indexing order. Within the set of short-term pictures, the default order is for the pictures to be ordered starting with the most recent buffered reference picture and proceeding through to the oldest reference picture (i.e., in decreasing order of picture number in the absence of wrapping of the ten-bit picture number field). Within the set of long-term pictures, the default order is for the pictures to be ordered starting with the picture with the smallest long-term index and proceeding up to the picture with long-term index equal to the most recent value of MLIP1–1.

For example, if the buffer contains three short-term pictures with short-term picture numbers 300, 302, and 303 (which were transmitted in increasing picture-number order) and two long-term pictures with long-term picture indices 0 and 3, the default index order is:

default relative index 0 refers to the short-term picture with picture number 303,

default relative index 1 refers to the short-term picture with picture number 302,

default relative index 2 refers to the short-term picture with picture number 300,

default relative index 3 refers to the long-term picture with long-term picture index 0, and

default relative index 4 refers to the long-term picture with long-term picture index 3.

The first ADPN or LPIR field that is received (if any) moves a specified picture out of the default order to the relative index of zero. The second such field moves a specified picture to the relative index of one, etc. The set of remaining pictures not moved to the front of the relative indexing order in this manner shall retain their default order amongst themselves and shall follow the pictures that have been moved to the front of the buffer in relative indexing order.

If there is not more than one reference picture used, no more than one ADPN or LPIR field shall be present in the same RPS layer unless the current picture is a B picture. If the current picture is a B picture and more than one reference picture is used, no more than two ADPN or LPIR fields shall be present in the same RPS layer.

Any re-mapping of picture numbers specified for some slice shall not affect the decoding process for any other slice.

In a P picture an RMPNI “end loop” indication is followed by RPBT.

Within one RPS layer, RMPNI shall not specify the placement of any individual reference picture into more than one re-mapped position in relative index order.

**TABLE 1**

### RMPNI operations for re-mapping of reference pictures

| Code Number | Re-mapping Specified                                                                                       |
|-------------|------------------------------------------------------------------------------------------------------------|
| 0           | ADPN field is present and corresponds to a negative difference to add to a picture number prediction value |
| 1           | ADPN field is present and corresponds to a positive difference to add to a picture number prediction value |
| 2           | LPIR field is present and specifies the long-term index for a reference picture                            |
| 3           | End loop for re-mapping of picture relative indexing default order                                         |

#### 3.3.5.1 Absolute Difference of Picture Numbers (ADPN)

ADPN is present only if indicated by RMPNI. ADPN follows RMPNI when present. The code number of the UVLC corresponds to ADPN – 1. ADPN represents the absolute difference between the picture number of the currently re-mapped picture and the prediction value for that picture number. If no previous ADPN fields have been sent within the current RPS layer, the prediction value shall be the picture number of the current picture. If some previous ADPN field has been sent, the prediction value shall be the picture number of the last picture that was re-mapped using ADPN.

If the picture number prediction is denoted PNP, and the picture number in question is denoted PNQ, the decoder shall determine PNQ from PNP and ADPN in a manner mathematically equivalent to the following:

```

if (RMPNI == '1') { // a negative difference
    if (PNP - ADPN < 0)
        PNQ = PNP - ADPN + MAX_PN;
    else
        PNQ = PNP - ADPN;
}else{ // a positive difference
    if (PNP + ADPN > MAX_PN-1)
        PNQ = PNP + ADPN - MAX_PN;
    else
        PNQ = PNP + ADPN;
}

```

The encoder shall control RMPNI and ADPN such that the decoded value of ADPN shall not be greater than or equal to MAX\_PN.

As an example implementation, the encoder may use the following process to determine values of ADPN and RMPNI to specify a re-mapped picture number in question, PNQ:

```

DELTA = PNQ - PNP;
if (DELTA < 0) {
    if (DELTA < -MAX_PN/2-1)
        MDELTA = DELTA + MAX_PN;
    else
        MDELTA = DELTA;
}else{
    if (DELTA > MAX_PN/2)
        MDELTA = DELTA - MAX_PN;
    else
        MDELTA = DELTA;
}

```

}

$ADPN = \text{abs}(MDELTA);$

where  $\text{abs}()$  indicates an absolute value operation. Note that the code number of the UVLC corresponds to the value of  $ADPN - 1$ , rather than the value of  $ADPN$  itself.

RMPNI would then be determined by the sign of  $MDELTA$ .

### 3.3.5.2 Long-term Picture Index for Re-Mapping (LPIR)

LPIR is present only if indicated by RMPNI. LPIR follows RMPNI when present. LPIR is transmitted using UVLC codewords. It represents the long-term picture index to be re-mapped. The prediction value used by any subsequent  $ADPN$  re-mappings is not affected by LPIR.

### 3.3.6 Reference Picture Buffering Type (RPBT)

RPBT specifies the buffering type of the currently decoded picture. It follows an RMPNI "end loop" indication when the picture is not an I picture. It is the first element of the RPS layer if the picture is an I picture. The values for RPBT are defined as follows:

Code number 0: Sliding Window,

Code number 1: Adaptive Memory Control.

In the "Sliding Window" buffering type, the current decoded picture shall be added to the buffer with default relative index 0, and any marking of pictures as "unused" in the buffer is performed automatically in a first-in-first-out fashion among the set of short-term pictures. In this case, if the buffer has sufficient "unused" capacity to store the current picture, no additional pictures shall be marked as "unused" in the buffer. If the buffer does not have sufficient "unused" capacity to store the current picture, the picture with the largest default relative index among the short-term pictures in the buffer shall be marked as "unused". In the sliding window buffering type, no additional information is transmitted to control the buffer contents.

In the "Adaptive Memory Control" buffering type, the encoder explicitly specifies any addition to the buffer or marking of data as "unused" in the buffer, and may also assign long-term indices to short-term pictures. The current picture and other pictures may be explicitly marked as "unused" in the buffer, as specified by the encoder. This buffering type requires further information that is controlled by memory management control operation (MMCO) parameters.

#### 3.3.6.1 Memory Management Control Operation (MMCO)

MMCO is present only when RPBT indicates "Adaptive Memory Control", and may occur multiple times if present. It specifies a control operation to be applied to manage the multi-picture buffer memory. The MMCO parameter is followed by data necessary for the operation specified by the value of MMCO, and then an additional MMCO parameter follows – until the MMCO value indicates the end of the list of such operations. MMCO commands do not affect the buffer contents or the decoding process for the decoding of the current picture – rather, they specify the necessary buffer status for the decoding of subsequent pictures in the bitstream. The values and control operations associated with MMCO are defined in TABLE 2.

If MMCO is Reset, all pictures in the multi-picture buffer (but not the current picture unless specified separately) shall be marked "unused" (including both short-term and long-term pictures).

The picture height and width shall not change within the bitstream except within a picture containing a Reset MMCO command.

A "stored picture" does not contain an MMCO command in its RPS layer which marks that (entire) picture as "unused". If the current picture is not a stored picture, its RPS layer shall not contain any of the following types of MMCO commands:

- An Reset MMCO command,

- Any MMCO command which marks any other picture (other than the current picture) as “unused” that has not also been marked as “unused” in the ERPS layer of a prior stored picture, or
- Any MMCO command which assigns a long-term index to a picture that has not also been assigned the same long-term index in the ERPS layer of a prior stored picture

**TABLE 2**

**Memory Management Control Operation (MMCO) Values**

| Code Number | Memory Management Control Operation         | Associated Data Fields Following |
|-------------|---------------------------------------------|----------------------------------|
| 0           | End MMCO Loop                               | None (end of ERPS layer)         |
| 1           | Mark a Short-Term Picture as “Unused”       | DPN                              |
| 2           | Mark a Long-Term Picture as “Unused”        | LPIN                             |
| 3           | Assign a Long-Term Index to a Picture       | DPN and LPIN                     |
| 4           | Specify the Maximum Long-Term Picture Index | MLIP1                            |
| 5           | Reset                                       | None                             |

### 3.3.6.2 Difference of Picture Numbers (DPN)

DPN is present when indicated by MMCO. DPN follows MMCO if present. DPN is transmitted using UVLC codewords and is used to calculate the PN of a picture for a memory control operation. It is used in order to assign a long-term index to a picture, mark a short-term picture as “unused”, or mark sub-picture areas of a short-term picture as “unused”. If the current decoded picture number is PNC and the decoded UVLC code number is DPN, an operation mathematically equivalent to the following equations shall be used for calculation of PNQ, the specified picture number in question:

```
if (PNC - DPN < 0)
    PNQ = PNC - DPN + MAX_PN;
else
    PNQ = PNC - DPN;
```

Similarly, the encoder may compute the DPN value to encode using the following relation:

```
if (PNC - PNQ < 0)
    DPN = PNC - PNQ + MAX_PN;
else
    DPN = PNC - PNQ;
```

For example, if the decoded value of DPN is zero and MMCO indicates marking a short-term picture as “unused”, the current decoded picture shall be marked as “unused” (thus indicating that the current picture is not a stored picture).

### 3.3.6.3 Long-term Picture Index (LPIN)

LPIN is present when indicated by MMCO. LPIN specifies the long-term picture index of a picture. It follows DPN if the operation is to assign a long-term index to a picture. It follows MMCO if the operation is to mark a long-term picture as “unused”.

### 3.3.6.4 Maximum Long-Term Picture Index Plus 1 (MLIP1)

MLIP1 is present if indicated by MMCO. MLIP1 follows MMCO if present. If present, MLIP1 is used to determine the maximum index allowed for long-term reference pictures (until receipt of another value of MLIP1). The decoder shall initially assume MLIP1 is “0” until some other value has been received. Upon receiving an MLIP1 parameter, the decoder shall consider all long-term pictures having indices greater than the decoded value of MLIP1 – 1 as “unused” for referencing by the decoding process for

subsequent pictures. For all other pictures in the multi-picture buffer, no change of status shall be indicated by MLIP1.

### 3.3.7 Supplemental Enhancement Information

Supplemental enhancement information (SEI) is encapsulated into chunks of data separate from coded slices, for example. It is up to the network adaptation layer to specify the means to transport SEI chunks. Each SEI chunk may contain one or more SEI messages. Each SEI message shall consist of a SEI header and SEI payload. The SEI header starts at a byte-aligned position from the first byte of a SEI chunk or from the first byte after the previous SEI message. The SEI header consists of two codewords, both of which consist of one or more bytes. The first codeword indicates the SEI payload type. Values from 00 to FE shall be reserved for particular payload types, whereas value FF is an escape code to extend the value range to yet another byte as follows:

```
payload_type = 0;
for (;;) {
    payload_type += *byte_ptr_to_sei;
    if (*byte_ptr_to_sei < 0xFF)
        break;
    byte_ptr_to_sei++;
}
```

The second codeword of the SEI header indicates the SEI payload size in bytes. SEI payload size shall be coded similarly to the SEI payload type.

The SEI payload may have a SEI payload header. For example, a payload header may indicate to which picture the particular data belongs. The payload header shall be defined for each payload type separately.

## 3.4 Macroblock layer

Following the syntax diagram for the macroblock elements, the various elements are described.

### 3.4.1 Number of Skipped Macroblocks (RUN)

A macroblock is called skipped if no information is sent. In that case the reconstruction of an inter macroblock is made by copying the collocated picture material from the last decoded frame. For a B macroblock skip means direct mode without coefficients. RUN indicates the number of skipped macroblocks in an inter- or B-picture before a coded macroblock. If a picture or slice ends with one or more skipped macroblocks, they are represented by an additional RUN which counts the number of skipped macroblocks.

### 3.4.2 Macro block type (MB\_Type)

Refer to TABLE 7. There are different MB-Type tables for Intra and Inter frames.

#### 3.4.2.1 Intra Macroblock modes

Intra 4x4            Intra coding as defined in sections 0 to 0.

Imode, nc, AC      See definition in section 0. These modes refer to 16x16 intra coding.

#### 3.4.2.2 Inter Macroblock Modes

Skip                No further information about the macroblock is transmitted. A copy of the collocated macroblock in the most recent decoded picture is used as reconstruction for the present macroblock.

NxM (eg. 8x4) The macroblock is predicted from a past picture with block size NxM. For each NxM block motion vector data is provided. Depending on N and M there may be 1 to 16 sets of motion vector data for a macroblock.

Intra 4x4 4x4 intra coding.

Code numbers from 9 and upwards represent 16x16 intra coding.

### 3.4.3 Intra Prediction Mode (Intra\_pred\_mode)

Even in Intra mode, prediction is always used for each sub block in a macroblock. A 4x4 block is to be coded (pixels labeled a to p below). The pixels A to I from neighboring blocks are already decoded and may be used for prediction.

|   |   |   |   |   |
|---|---|---|---|---|
| I | A | B | C | D |
| E | a | b | c | d |
| F | e | f | g | h |
| G | i | j | k | l |
| H | m | n | o | p |

FIGURE 6

Syntax diagram for the Picture header.

#### 3.4.3.1.1 Prediction for Intra-4x4 mode for luminance blocks

For the luminance signal, there are 6 intra prediction modes labeled 0 to 5. Mode 0 is 'DC-prediction' (see below). The other modes represent directions of predictions as indicated below.

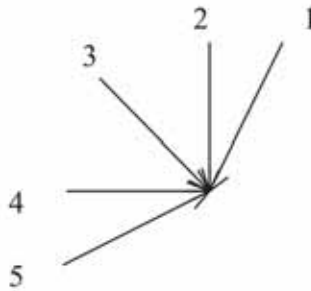


FIGURE 7

Syntax diagram for the Picture header.

#### Mode 0: DC prediction

Generally all pixels are predicted by  $(A+B+C+D+E+F+G+H)/8$ . If four of the pixels are outside the picture, the average of the remaining four is used for prediction. If all 8 pixels are outside the picture the prediction for all pixels in the block is 128. A block may therefore always be predicted in this mode.

#### Mode 1: Vertical/Diagonal Prediction

This mode is used only if all A,B,C,D are inside the picture.

a is predicted by:  $(A+B)/2$

e is predicted by B

b,i are predicted by  $(B+C)/2$

f,m are predicted by C

c,j are predicted by  $(C+D)/2$

d,g,h,k,l,n,o,p are predicted by D

#### Mode 2: Vertical prediction

If A,B,C,D are inside the picture, a,e,i,m are predicted by A, b,f,j,n by B etc.

### Mode 3: Diagonal prediction

This mode is used only if all A,B,C,D,E,F,G,H,I are inside the picture. This is a 'diagonal' prediction.

m is predicted by:  $(H+2G+F)//4$   
i,n are predicted by  $(G+2F+E)//4$   
e,j,o are predicted by  $(F+2E+I)//4$   
a,f,k,p are predicted by  $(E+2I+A)//4$   
b,g,l are predicted by  $(I+2A+B)//4$   
c,h are predicted by  $(A+2B+C)//4$   
d is predicted by  $(B+2C+D)//4$

### Mode 4: Horizontal prediction

If E,F,G,H are inside the picture, a,b,c,d are predicted by E, e,f,g,h by F etc.

### Mode 5: Horizontal/Diagonal prediction

This mode is used only if all E,F,G,H are inside the picture.

a is predicted by:  $(E+F)/2$   
b is predicted by F  
c,e are predicted by  $(F+G)/2$   
f,d are predicted by G  
i,g are predicted by  $(G+H)/2$   
h,j,k,l,m,n,o,p are predicted by H

### Coding of Intra 4x4 prediction modes

Since each of the 4x4 luminance blocks shall be assigned a prediction mode, this will require a considerable number of bits if coded directly. We have therefore tried to find more efficient ways of coding mode information. First of all we observe that the chosen prediction of a block is highly correlated with the prediction modes of adjacent blocks. This is illustrated in FIGURE 8a. When the prediction modes of A and B are known (including the case that A or B or both are outside the picture) an ordering of the most probable, next most probable etc. of C is given. When an adjacent block is coded by 16x16 intra mode, prediction mode is "mode 0: DC\_prediction"; when it is coded in inter mode, prediction mode is "mode 0: DC\_prediction" in the usual case and "outside" in the case of constrained intra update. This ordering is listed in TABLE 3.

For each prediction mode of A and B a list of 5 numbers is given. Example: Prediction mode for A and B is 2. The string 2 1 0 3 4 5 indicates that mode 2 is also the most probable mode for block C. Mode 1 is the next most probable one etc. In the bitstream there will for instance be information that Prob0 = 1 (see TABLE 7) indicating that the next most probable mode shall be used for block C. In our example this means Intra prediction mode 1. Use of '-' in the table indicates that this instance cannot occur because A or B or both are outside the picture.

For more efficient coding, information on intra prediction of two 4x4 luminance blocks are coded in one codeword (Prob0 and Prob1 in TABLE 7). The order of the resulting 8 codewords is indicated in FIGURE 8b.

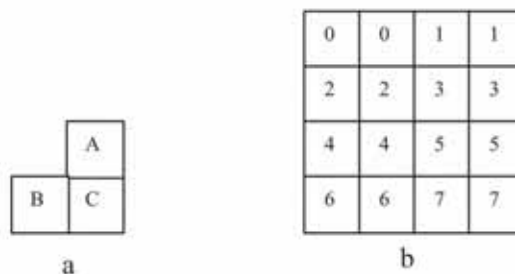


FIGURE 8

a) Prediction mode of block C shall be established. A and B are adjacent blocks. b) order of intra prediction information in the bitstream

TABLE 3

Prediction mode as a function of ordering signaled in the bitstream (see text)

| B\A     | outside | 0      | 1      | 2      | 3      | 4      | 5      |
|---------|---------|--------|--------|--------|--------|--------|--------|
| Outside | 0       | 021--- | 102--- | 201--- | 012--- | 012--- | 012--- |
| 0       | 045---  | 041352 | 104325 | 230415 | 304215 | 043152 | 043512 |
| 1       | 045---  | 014325 | 102435 | 203145 | 032145 | 041325 | 014352 |
| 2       | 045---  | 012345 | 102345 | 210345 | 302145 | 042135 | 013245 |
| 3       | 045---  | 304152 | 310425 | 231054 | 304215 | 403512 | 305412 |
| 4       | 405---  | 403512 | 401532 | 240351 | 430512 | 403512 | 405312 |
| 5       | 504---  | 540312 | 015432 | 201453 | 530412 | 450312 | 504132 |

#### 3.4.3.1.2 Prediction for Intra-16x16 mode for luminance

Assume that the block to be predicted has pixel locations 0 to 15 horizontally and 0 to 15 vertically. We use the notation  $P(i,j)$  where  $i,j = 0..15$ .  $P(i,-1)$ ,  $i=0..15$  are the neighboring pixels above the block and  $P(-1,j)$ ,  $j=0..15$  are the neighboring pixels to the left of the block.  $Pred(i,j)$   $i,j = 0..15$  is the prediction for the whole Luminance macroblock. We have 4 different prediction modes:

##### Mode 0 (vertical)

$$Pred(i,j) = P(i,-1), i,j=0..15$$

##### Mode 1 (horizontal)

$$Pred(i,j) = P(-1,j), i,j=0..15$$

##### Mode 2 (DC prediction)

$$Pred(i,j) = \left( \sum_{i=0}^{15} (P(-1,i) + P(i,-1)) \right) / 32 \quad i,j=0..15,$$

where only the average of 16 pixels are used when the other 16 pixels are outside the picture or slice. If all 32 pixels are outside the picture, the prediction for all pixels in the block is 128.

##### Mode 3 (Plane prediction)

$$Pred(i,j) = \max(0, \min(255, (a + bx(i-7) + cx(j-7) + 16)/32)),$$

where:

$$a = 16x(P(-1,15) + P(15,-1))$$

$$b = 5x(H/4)/16$$

$$c = 5x(V/4)/16$$

$$H = \sum_{i=1}^8 ix(P(7+i,-1) + P(7-i,-1))$$

$$V = \sum_{j=1}^8 jx(P(-1,7+j) + P(-1,7-j))$$

##### Residual coding

The residual coding is based on 4x4 transform. But similar to coding of chrominance coefficients, another 4x4 transform to the 16 DC coefficients in the macroblock are added. In that way we end up with an overall DC for the whole MB which works well in flat areas.

Since we use the same integer transform to DC coefficients, we have to perform additional normalization to those coefficients, which implies a division by 676. To avoid the division we performed normalization by  $49/2^{15}$  on the encoder side and  $48/2^{15}$  on the decoder side, which gives sufficient accuracy.

Only single scan is used for 16x16 intra coding.

To produce the bitstream, we first scan through the 16 ‘DC transform’ coefficients. There is no ‘CBP’ information to indicate no coefficients on this level. If  $AC = 1$  (see below) ac coefficients of the 16 4x4 blocks are scanned. There are 15 coefficients in each block since the DC coefficients are included in the level above.

#### Coding of mode information for Intra-16x16 mode

See TABLE 7. Three parameters have to be signaled. They are all included in MB-type.

Imode: 0,1,2,3

AC: 0 means there are no ac coefficients in the 16x16 block. 1 means that there is at least one ac coefficient and all 16 blocks are scanned.

nc: CBP for chrominance (see 3.4.6)

#### 3.4.3.1.3 Prediction in intra coding of chrominance blocks

For chrominance prediction there is only one mode. No information is therefore needed to be transmitted. The prediction is indicated in the figure below. The 8x8 chrominance block consists of 4 4x4 blocks A,B,C,D.  $S_0, S_1, S_2, S_3$  are the sums of 4 neighbouring pixels.

If  $S_0, S_1, S_2, S_3$  are all inside the frame:

$$A = (S_0 + S_2 + 4)/8$$

$$B = (S_1 + 2)/4$$

$$C = (S_3 + 2)/4$$

$$D = (S_1 + S_3 + 4)/8$$

If only  $S_0$  and  $S_1$  are inside the frame:

$$A = (S_0 + 2)/4$$

$$B = (S_1 + 2)/4$$

$$C = (S_0 + 2)/4$$

$$D = (S_1 + 2)/4$$

If only  $S_2$  and  $S_3$  are inside the frame:

$$A = (S_2 + 2)/4$$

$$B = (S_2 + 2)/4$$

$$C = (S_3 + 2)/4$$

$$D = (S_3 + 2)/4$$

If  $S_0, S_1, S_2, S_3$  are all outside the frame:  $A = B = C = D = 128$

(Note: This prediction should be considered changed)

|       |       |       |
|-------|-------|-------|
|       | $S_0$ | $S_1$ |
| $S_2$ | A     | B     |
| $S_3$ | C     | D     |

FIGURE 9

**Prediction of Chrominance blocks.**

#### 3.4.4 Reference frame (Ref\_frame)

If PTTYPE indicates possibility of prediction from more than one previous decoded picture, the exact frame to be used must be signalled. This is done according to the following table.

| Code_number | Reference frame                       |
|-------------|---------------------------------------|
| 0           | The last decoded frame (1 frame back) |
| 1           | 2 frames back                         |
| 2           | 3 frames back                         |
| ..          | ..                                    |

#### 3.4.5 Motion Vector Data (MVD)

If so indicated by MB\_type, vector data for 1-16 blocks are transmitted. For every block a prediction is formed for the horizontal and vertical components of the motion vector. MVD signals the difference between the vector component to be used and this prediction. The order in which vector data is sent is indicated in FIGURE 2. Motion vectors are allowed to point to pixels outside the reference frame. If a pixel outside the reference frame is referred to in the prediction process, the nearest pixel belonging to the frame (an edge or corner pixel) shall be used. All fractional pixel positions shall be interpolated as described below. If a pixel referred in the interpolation process (necessarily integer accuracy) is outside of the reference frame it shall be replaced by the nearest pixel belonging to the frame (an edge or corner pixel). Reconstructed motion vectors shall be clipped to +-19 integer pixels outside of the frame.

##### 3.4.5.1 Prediction of vector components

In all macroblocks with square motion vector blocks (16x16, 8x8, 4x4) "median prediction" (see 3.4.5.1.1) is used. In case the macroblock may be classified to have directional segmentation the prediction is defined in 3.4.5.1.2.

##### 3.4.5.1.1 Median prediction

In the figure below the vector component E of the indicated block shall be predicted. The prediction is normally formed as the median of A, B and C. However, the prediction may be modified as described below. Notice that it is still referred to as "median prediction"

- A The component applying to the pixel to the left of the upper left pixel in E
- B The component applying to the pixel just above the upper left pixel in E
- C The component applying to the pixel above and to the right of the upper right pixel in E
- D The component applying to the pixel above and to the left of the upper left pixel in E

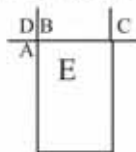


FIGURE 10

##### Median prediction of motion vectors.

A, B, C, D and E may represent motion vectors from different reference pictures. As an example we may be seeking prediction for a motion vector for E from the last decoded picture. A, B, C and D may represent vectors from 2, 3, 4 and 5 pictures back. The following substitutions may be made prior to median filtering.

- If A and D are outside the picture, their values are assumed to be zero and they are considered to have "different reference picture than E".
- If D, B, C are outside the picture, the prediction is equal to A (equivalent to replacing B and C with A before median filtering).

- If C is outside the picture or still not available due to the order of vector data (see FIGURE 2), C is replaced by D.

If any of the blocks A, B, C, D are intra coded they count as having "different reference frame. If one and only one of the vector components used in the median calculation (A, B, C) refer to the same reference picture as the vector component E, this one vector component is used to predict E.

#### 3.4.5.1.2 Directional segmentation prediction

If the macroblock where the block to be predicted belongs to has a directional segmentation (vector block size of 8x16, 16x8, 8x4 or 4x8) the prediction is generated as follows (refer to figure below and the definitions of A, B, C, E above):

Vector block size 8x16:

- Left block: A is used as prediction if it has the same reference picture as E, otherwise "Median prediction" is used
- Right block: C is used as prediction if it has the same reference picture as E, otherwise "Median prediction" is used

Vector block size 16x8:

- Upper block: B is used as prediction if it has the same reference picture as E, otherwise "Median prediction" is used
- Lower block: A is used as prediction if it has the same reference picture as E, otherwise "Median prediction" is used

Vector block size 8x4:

- For white blocks: "Median prediction" is used
- For shaded blocks: A is used as prediction

Vector block size 4x8:

- For white blocks: "Median prediction" is used
- For shaded blocks: B is used as prediction

If the indicated prediction block is outside the picture, the same substitution rules are applied as in the case of median prediction.



FIGURE 11

#### Directional segmentation prediction

#### 3.4.5.2 Chrominance vectors

Chrominance vectors are derived from the Luminance vectors. Since chrominance has half resolution compared to Luminance, the chrominance vectors are obtained by a division of two:

$\text{Croma\_vector} = \text{Luma\_vector} / 2$  - which means that the chrominance vectors have a resolution of 1/8 pixel.

Due to the half resolution, a chrominance vector applies to 1/4 as many pixels as the Luminance vector. For example if the Luminance vector applies to 8x16 Luminance pixels, the corresponding chrominance

vector applies to 4x8 chrominance pixels and if the Luminance vector applies to 4x4 Luminance pixels, the corresponding chrominance vector applies to 2x2 chrominance pixel.

For fractional pixel interpolation for chrominance prediction, bilinear interpolation is used. The result is rounded to the nearest integer.

#### 3.4.6 Coded Block Pattern (CBP)

The CBP contains information of which 8x8 blocks - Luminance and chrominance - contain transform coefficients. Notice that an 8x8 block contains 4 4x4 blocks meaning that the statement '8x8 block contains coefficients' means that 'one or more of the 4 4x4 blocks contain coefficients'. The 4 least significant bits of CBP contain information on which of 4 8x8 luminance blocks in a macroblock contains nonzero coefficients. Let us call these 4 bits CBPY. The ordering of 8x8 blocks is indicated in FIGURE 3. A 0 in position n of CBP (binary representation) means that the corresponding 8x8 block has no coefficients whereas a 1 means that the 8x8 block has one or more non-zero coefficients.

For chrominance we define 3 possibilities:

- nc=0: no chrominance coefficients at all.
- nc=1: There are nonzero 2x2 transform coefficients. All chrominance AC coefficients = 0. Therefore we do not send any EOB for chrominance AC coefficients.
- nc=2: There may be 2x2 nonzero coefficients and there is at least one nonzero chrominance AC coefficient present. In this case we need to send 10 EOBs (2 for DC coefficients and 2x4=8 for the 8 4x4 blocks) for chrominance in a macroblock.

The total CBP for a macroblock is: **CBP = CBPY + 16xnc**

The CBP is signalled with a different codeword for Inter macroblocks and Intra macroblocks since the statistics of CBP values are different in the two cases.

#### 3.4.7 Change of Quantizer Value (Dquant)

Dquant contains the possibility of changing QUANT on the macroblock level. It does not need to be present for macroblocks without nonzero transform coefficients. More specifically Dquant is present for non-skipped macroblocks:

- If CBP indicates that there are nonzero transform coefficients in the MB **or**
- If the MB is 16x16 based intra coded

The value of Dquant may range from -16 to +16 which enables the QP to be changed to any value in the range 0-31.

$QUANT_{new} = \text{modulo}_{32}(QUANT_{old} + Dquant + 32)$  (also known as "arithmetic wrap")

## 4 Decoding Process

### 4.1 Slice Decoding

#### 4.1.1 Multi-Picture Decoder Process

The decoder stores the reference pictures for inter-picture decoding in a multi-picture buffer. The decoder replicates the multi-picture buffer of the encoder according to the reference picture buffering type and any memory management control operations specified in the bitstream. The buffering scheme may also be operated when partially erroneous pictures are decoded.

Each transmitted and stored picture is assigned a Picture Number (PN) which is stored with the picture in the multi-picture buffer. PN represents a sequential picture counting identifier for stored pictures. PN is constrained, using modulo MAX\_PN arithmetic operation. For the first transmitted picture, PN should be "0". For each and every other transmitted and stored picture, PN shall be increased by 1. If the difference (modulo MAX\_PN) of the PNs of two consecutively received and stored pictures is not 1, the decoder should infer a loss of pictures or corruption of data. In such a case, a back-channel message indicating the loss of pictures may be sent to the encoder.

Besides the PN, each picture stored in the multi-picture buffer has an associated index, called the default relative index. When a picture is first added to the multi-picture buffer it is given default relative index 0 – unless it is assigned to a long-term index. The default relative indices of pictures in the multi-picture buffer are modified when pictures are added to or removed from the multi-picture buffer, or when short-term pictures are assigned to long-term indices.

The pictures stored in the multi-picture buffers can also be divided into two categories: long-term pictures and short-term pictures. A long-term picture can stay in the multi-picture buffer for a long time (more than MAX\_PN-1 coded and stored picture intervals). The current picture is initially considered a short-term picture. Any short-term picture can be changed to a long-term picture by assigning it a long-term index according to information in the bitstream. The PN is the unique ID for all short-term pictures in the multi-picture buffer. When a short-term picture is changed to a long-term picture, it is also assigned a long-term picture index (LPIN). A long-term picture index is assigned to a picture by associating its PN to an LPIN. Once a long-term picture index has been assigned to a picture, the only potential subsequent use of the long-term picture's PN within the bitstream shall be in a repetition of the long-term index assignment. The PNs of the long-term pictures are unique within MAX\_PN transmitted and stored pictures. Therefore, the PN of a long-term picture cannot be used for assignment of a long-term index after MAX\_PN-1 transmitted subsequent stored pictures. LPIN becomes the unique ID for the life of a long-term picture.

PN (for a short-term picture) or LPIN (for a long-term picture) can be used to re-map the pictures into re-mapped relative indices for efficient reference picture addressing.

##### 4.1.1.1 Decoder Process for Short/Long-term Picture Management

The decoder may have both long-term pictures and short-term pictures in its multi-picture buffer. The MLIP1 field is used to indicate the maximum long-term picture index allowed in the buffer. If no prior value of MLIP1 has been sent, no long-term pictures shall be in use, i.e. MLIP1 shall initially have an implied value of "0". Upon receiving an MLIP1 parameter, a new MLIP1 shall take effect until another value of MLIP1 is received. Upon receiving a new MLIP1 parameter in the bitstream, all long-term pictures with associated long-term indices greater than or equal to MLIP1 shall be considered marked "unused". The frequency of transmitting MLIP1 is out of the scope of this Recommendation. However, the encoder should send an MLIP1 parameter upon receiving an error message, such as an Intra request message.

A short-term picture can be changed to a long-term picture by using an MMCO command with an associated DPN and LPIN. The short-term picture number is derived from DPN and the long-term picture index is LPIN. Upon receiving such an MMCO command, the decoder shall change the short-term picture with PN indicated by DPN to a long-term picture and shall assign it to the long-term index indicated by LPIN. If a long-term picture with the same long-term index already exists in the buffer, the previously-

existing long-term picture shall be marked “unused”. An encoder shall not assign a long-term index greater than  $MLIP1-1$  to any picture. If LPIN is greater than  $MLIP1-1$ , this condition should be treated by the decoder as an error. For error resilience, the encoder may send the same long-term index assignment operation or MLIP1 specification message repeatedly. If the picture specified in a long-term assignment operation is already associated with the required LPIN, no action shall be taken by the decoder. An encoder shall not assign the same picture to more than one long term index value. If the picture specified in a long-term index assignment operation is already associated with a different long-term index, this condition should be treated as an error. An encoder shall only change a short-term picture to a long-term picture within  $MAX\_PN$  transmitted consecutive stored pictures. In other words, a short-term picture shall not stay in the short-term buffer after more than  $MAX\_PN-1$  subsequent stored pictures have been transmitted. An encoder shall not assign a long-term index to a short-term picture that has been marked as “unused” by the decoding process prior to the first such assignment message in the bitstream. An encoder shall not assign a long-term index to a picture number that has not been sent.

#### **4.1.1.2 Decoder Process for Reference Picture Buffer Mapping**

The decoder employs indices when referencing a picture for motion compensation on the macroblock layer. In pictures other than B pictures, these indices are the default relative indices of pictures in the multi-picture buffer when the fields ADPN and LPIR are not present in the current slice layer as applicable, and are re-mapped relative indices when these fields are present. In B pictures, the first one or two pictures (depending on BTPSM) in relative index order are used for backward prediction, and the forward picture reference parameters specify a relative index into the remaining pictures for use in forward prediction. (needs to be changed)

The indices of pictures in the multi-picture buffer can be re-mapped onto newly specified indices by transmitting the RMPNI, ADPN, and LPIR fields. RMPNI indicates whether ADPN or LPIR is present. If ADPN is present, RMPNI specifies the sign of the difference to be added to a picture number prediction value. The ADPN value corresponds to the absolute difference between the PN of the picture to be re-mapped and a prediction of that PN value. The first transmitted ADPN is computed as the absolute difference between the PN of the current picture and the PN of the picture to be re-mapped. The next transmitted ADPN field represents the difference between the PN of the previous picture that was re-mapped using ADPN and that of another picture to be re-mapped. The process continues until all necessary re-mapping is complete. The presence of re-mappings specified using LPIR does not affect the prediction value for subsequent re-mappings using ADPN. If RMPNI indicates the presence of an LPIR field, the re-mapped picture corresponds to a long-term picture with a long-term index of LPIR. If any pictures are not re-mapped to a specific order by RMPNI, these remaining pictures shall follow after any pictures having a re-mapped order in the indexing scheme, following the default order amongst these non-re-mapped pictures.

If the decoder detects a missing picture, it may invoke some concealment process, and may insert an error-concealed picture into the multi-picture buffer. Missing pictures can be identified if one or several picture numbers are missing or if a picture not stored in the multi-picture buffer is indicated in a transmitted ADPN or LPIR. Concealment may be conducted by copying the closest temporally preceding picture that is available in the multi-picture buffer into the position of the missing picture. The temporal order of the short-term pictures in the multi-picture buffer can be inferred from their default relative index order and PN fields. In addition or instead, the decoder may send a forced INTRA update signal to the encoder by external means (for example, Recommendation H.245), or the decoder may use external means or back-channel messages (for example, Recommendation H.245) to indicate the loss of pictures to the encoder. A concealed picture may be inserted into the multi-picture buffer when using the "Sliding Window" buffering type. If a missing picture is detected when decoding a Slice, the concealment may be applied to the picture as if the missing picture had been detected at the picture layer.

#### **4.1.1.3 Decoder Process for Multi-Picture Motion Compensation**

Multi-picture motion compensation is applied if the use of more than one reference picture is indicated. For multi-picture motion compensation, the decoder chooses a reference picture as indicated using the reference frame fields on the macroblock layer. Once, the reference picture is specified, the decoding process for motion compensation proceeds.

#### 4.1.1.4 Decoder Process for Reference Picture Buffering

The buffering of the currently decoded picture can be specified using the reference picture buffering type (RPBT). The buffering may follow a first-in, first-out ("Sliding Window") mode. Alternatively, the buffering may follow a customized adaptive buffering ("Adaptive Memory Control") operation that is specified by the encoder in the forward channel.

The "Sliding Window" buffering type operates as follows. First, the decoder determines whether the picture can be stored into "unused" buffer capacity. If there is insufficient "unused" buffer capacity, the short-term picture with the largest default relative index (i.e. the oldest short-term picture in the buffer) shall be marked as "unused". The current picture is stored in the buffer and assigned a default relative index of zero. The default relative index of all other short-term pictures is incremented by one. The default relative indices of all long-term pictures are incremented by one minus the number of short-term pictures removed.

In the "Adaptive Memory Control" buffering type, specified pictures may be removed from the multi-picture buffer explicitly. The currently decoded picture, which is initially considered a short-term picture, may be inserted into the buffer with default relative index 0, may be assigned to a long-term index, or may be marked as "unused" by the encoder. Other short-term pictures may also be assigned to long-term indices. The buffering process shall operate in a manner functionally equivalent to the following: First, the current picture is added to the multi-picture buffer with default relative index 0, and the default relative indices of all other pictures are incremented by one. Then, the MMCO commands are processed:

If MMCO indicates a reset of the buffer contents, all pictures in the buffer are marked as "unused" except the current picture (which will be the picture with default relative index 0).

If MMCO indicates a maximum long-term index using MLIP1, all long-term pictures having long-term indices greater than or equal to MLIP1 are marked as "unused" and the default relative index order of the remaining pictures are not affected.

If MMCO indicates that a picture is to be marked as "unused" in the multi-picture buffer and if that picture has not already been marked as "unused", the specified picture is marked as "unused" in the multi-picture buffer and the default relative indices of all subsequent pictures in default order are decremented by one.

If MMCO indicates the assignment of a long-term index to a specified short-term picture and if the specified long-term index has not already been assigned to the specified short-term picture, the specified short-term picture is marked in the buffer as a long-term picture with the specified long-term index. If another picture is already present in the buffer with the same long-term index as the specified long-term index, the other picture is marked as "unused". All short-term pictures that were subsequent to the specified short-term picture in default relative index order and all long-term pictures having a long-term index less than the specified long-term index have their associated default relative indices decremented by one. The specified picture is assigned to a default relative index of one plus the highest of the incremented default relative indices, or zero if there are no such incremented indices.

## 4.2 Motion Compensation

### 4.2.1 Fractional pixel accuracy

#### 4.2.1.1 1/4 luminance sample interpolation

The pixels labeled as 'A' represent original pixels at integer positions and other symbols represent the pixels to be interpolated

|          |   |          |   |          |
|----------|---|----------|---|----------|
| <b>A</b> | d | <b>b</b> | d | <b>A</b> |
| e        | h | f        | h |          |
| <b>b</b> | g | <b>c</b> | g | <b>b</b> |
| e        | h | f        | i |          |
| <b>A</b> |   | <b>b</b> |   | <b>A</b> |

The interpolation proceeds as follows

- 1 The 1/2 sample positions labeled as 'b' are obtained by first applying 6-tap filter (1,-5,20,20,-5,1) to nearest pixels at integer locations in horizontal or vertical direction. The resulting intermediate value  $b$  is divided by 32, rounded to the nearest integer and clipped to lie in the range [0, 255].
- 2 The 1/2 sample position labeled as 'c' is obtained using 6-tap filtering (1,-5,20,20,-5,1) of intermediate the values  $b$  of the closest 1/2 sample positions in vertical or horizontal direction. The obtained value is divided by 1024, rounded to the nearest integer and clipped to lie in the range [0, 255].
- 3 The 1/4 sample positions labeled as 'd', 'g', 'e' and 'f' are obtained by averaging (with truncation to the nearest integer) the two nearest samples at integer or 1/2 sample position using  $d=(A+b)/2$ ,  $g=(b+c)/2$ ,  $e=(A+b)/2$ ,  $f=(b+c)/2$ .
- 4 The 1/4 sample positions labeled as 'h' are obtained by averaging (with truncation to nearest integer) the two nearest pixels 'b' in diagonal direction.
- 5 The pixel labeled 'i' (the "funny" position) is computed as  $(A_1+A_2+A_3+A_4+2)/4$  using the four nearest original pixels.

#### 4.2.1.2 1/8 Pel Luminance Samples Interpolation

The pixels labeled as 'A' represent original pixels at integer positions and other symbols represent 1/2, 1/4 and 1/8 pixels to be interpolated.

|                      |   |                       |   |                       |   |                       |   |          |
|----------------------|---|-----------------------|---|-----------------------|---|-----------------------|---|----------|
| <b>A</b>             | d | <b>b<sup>1</sup></b>  | d | <b>b<sup>2</sup></b>  | d | <b>b<sup>3</sup></b>  | d | <b>A</b> |
| d                    | e | d                     | f | d                     | f | d                     | e |          |
| <b>b<sup>1</sup></b> | d | <b>c<sup>11</sup></b> | d | <b>c<sup>12</sup></b> | d | <b>c<sup>13</sup></b> | d |          |
| d                    | f | d                     | g | d                     | g | d                     | f |          |
| <b>b<sup>2</sup></b> | d | <b>c<sup>21</sup></b> | d | <b>c<sup>22</sup></b> | d | <b>c<sup>23</sup></b> | d |          |
| d                    | f | d                     | g | d                     | g | d                     | e |          |
| <b>b<sup>3</sup></b> | d | <b>c<sup>31</sup></b> | d | <b>c<sup>32</sup></b> | d | <b>c<sup>33</sup></b> | d |          |
| d                    | e | d                     | f | d                     | f | d                     | e |          |
| <b>A</b>             |   |                       |   |                       |   |                       |   | <b>A</b> |

The interpolation proceeds as follows

- 1 The 1/2 and 1/4 sample positions labeled as 'b' are obtained by first calculating intermediate values 'b' using 8 tap filtering of nearest pixels at integer locations in horizontal or vertical direction. The final value of 'b' is calculated as  $(b+128)/256$  and clipped to lie in the range [0, 255]. The 1/2 and 1/4 sample positions labeled as 'c' are obtained by 8 tap filtering of intermediate values  $b$  in horizontal or vertical direction, dividing the result of filtering by 65536, rounding it to the nearest integer and clipping it to lie in the range [0, 255]. The filter tap values depend on pixel position and are listed below:
 

| 'b', 'c' position | filter tabs                          |
|-------------------|--------------------------------------|
| 1/4               | (-3, 12, -37, 229, 71, -21, 6, -1)   |
| 2/4               | (-3, 12, -39, 158, 158, -39, 12, -3) |
| 3/4               | (-1, 6, -21, 71, 229, -37, 12, -3)   |
- 2 The 1/8 sample positions labeled as 'd' are calculated as the average (with truncation to the nearest integer) of the two closest 'A', 'b' or 'c' values in the horizontal or vertical direction.
- 3 The 1/8 sample positions labeled as 'e' are calculated by averaging with truncation of two 1/4 pixels labeled as 'b<sup>1</sup>', which are the closest in the diagonal direction [Editor: what about the e-position column 8, row 6 ?]. The 1/8 sample positions labeled as 'g' are calculated via  $(A+3c^{22}+2)/4$  and the 1/8 sample positions marked as 'f' are calculated via  $(3b^1 + b^1 + 2)/4$  (pixel 'b<sup>2</sup>' closer to 'f' is multiplied by 3).

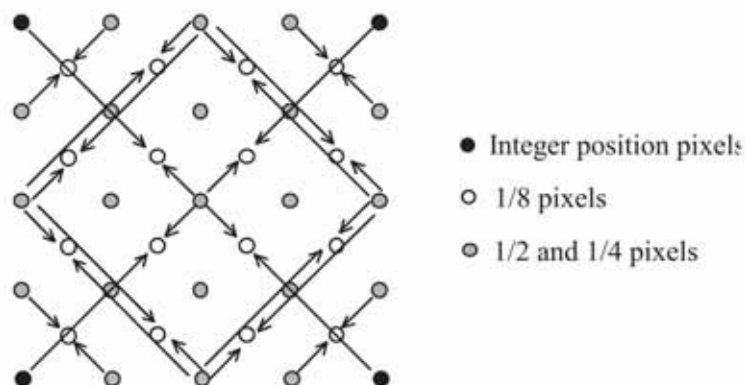


FIGURE 12

**Diagonal interpolation for 1/8 pel interpolation.**

### 4.3 Transform Coefficient Decoding

This section defines all elements related to transform coding and decoding. It is therefore relevant to all the syntax elements 'Tcoeff' in the syntax diagram.

#### 4.3.1 Transform for Blocks of Size 4x4 Samples

Instead of DCT, an integer transform with basically the same coding property as a 4x4 DCT is used. The transformation of the pixels a,b,c,d into 4 transform coefficients A,B,C,D is defined by:

$$A = 13a + 13b + 13c + 13d$$

$$B = 17a + 7b - 7c - 17d$$

$$C = 13a - 13b - 13c + 13d$$

$$D = 7a - 17b + 17c - 7d$$

The inverse transformation of transform coefficients A,B,C,D into 4 pixels a',b',c',d' is defined by:

$$a' = 13A + 17B + 13C + 7D$$

$$b' = 13A + 7B - 13C - 17D$$

$$c' = 13A - 7B - 13C + 17D$$

$$d' = 13A - 17B + 13C - 7D$$

The relation between a and a' is:  $a' = 676a$ . This is because the expressions defined above contain no normalisation. Normalisation will be performed in the quantization/dequantisation process and a final shift after inverse quantization.

The transform/inverse is performed both vertically and horizontally in the same manner as in H.263. By the above exact definition of inverse transform, the same operations will be performed on coder and decoder side, which means that we have no 'inverse transform mismatch'.

#### 4.3.2 Transform for Blocks of Size 2x2 Samples (DC vales for Chrominance)

With the low resolution of chrominance it seems to be preferable to have larger blocksize than 4x4. Specifically the 8x8 DC coefficient seems very useful for better definition of low resolution chrominance. The 2 dimensional 2x2 transform procedure is illustrated below. DC0,1,2,3 are the DC coefficients of 2x2 chrominance blocks.

$$\begin{array}{cc} \text{DC0} & \text{DC1} \\ \text{DC2} & \text{DC3} \end{array} \text{ Two dimensional 2x2 transform } \Rightarrow \begin{array}{cc} \text{DDC}(0,0) & \text{DDC}(1,0) \\ \text{DDC}(0,1) & \text{DDC}(1,1) \end{array}$$

Definition of transform:

$$\text{DCC}(0,0) = (\text{DC0} + \text{DC1} + \text{DC2} + \text{DC3})/2$$

$$DCC(1,0) = (DC0-DC1+DC2-DC3)/2$$

$$DCC(0,1) = (DC0+DC1-DC2-DC3)/2$$

$$DCC(1,1) = (DC0-DC1-DC2+DC3)/2$$

Definition of inverse transform:

$$DC0 = (DCC(0,0) + DCC(1,0) + DCC(0,1) + DCC(1,1))/2$$

$$DC1 = (DCC(0,0) - DCC(1,0) + DCC(0,1) - DCC(1,1))/2$$

$$DC2 = (DCC(0,0) + DCC(1,0) - DCC(0,1) - DCC(1,1))/2$$

$$DC3 = (DCC(0,0) - DCC(1,0) - DCC(0,1) + DCC(1,1))/2$$

### 4.3.3 Zig-zag Scanning and Quantization

#### 4.3.3.1 Simple Zig-Zag Scan

Except for Intra coding of luminance with  $QP < 24$ , simple scan is used. This is basically zig-zag scanning similar to the one used in H.263. The scanning pattern is:

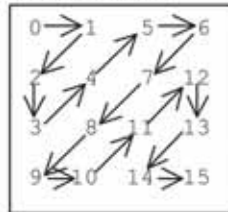


FIGURE 13

Simple zig-zag scan.

#### 4.3.3.2 Double Zig-Zag Scan

When using the VLC defined above, we use a one bit code for EOB. For Inter blocks and Intra with high QP the probability of EOB is typically 50%, which is well matched with the VLC. In other words this means that we on average have one non-zero coefficient per 4x4 block in addition to the EOB code (remember that a lot of 4x4 blocks only have EOB). On the other hand, for Intra coding we typically have more than one non-zero coefficient per 4x4 block. This means that the 1 bit EOB becomes inefficient. To improve on this the 4x4 block is subdivided into two parts that are scanned separately and with one EOB each. The two scanning parts are shown below – one of them in bold.

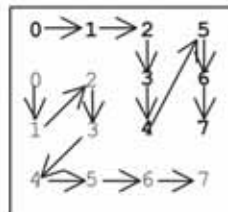


FIGURE 14

Double zig-zag scan.

#### 4.3.3.3 Quantization

The quantization/dequantization process shall perform 'normal' quantization/dequantization as well as take care of the fact that transform operations are kept very simple and therefore do not contain normalization of transform coefficients. 32 different QP values are used. They are arranged so that there is an increase of step size of about 12% from one QP to the next. Increase of QP by 6 means that the step size is about doubled. There is no 'dead zone' in the quantization process and the total range of step size from smallest to largest is about the same as for H.263.

The QP signalled in the bitstream applies for luminance quantization/dequantization. This could be called  $QP_{luma}$ . For chrominance quantization/dequantization a different value -  $QP_{chroma}$  - is used. The relation between the two is:

|               |                                                                                       |
|---------------|---------------------------------------------------------------------------------------|
| $QP_{luma}$   | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 |
| $QP_{chroma}$ | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 17 18 19 20 20 21 22 22 23 23 24 24 25 25 |

When QP is used in the following we mean  $QP_{luma}$  or  $QP_{chroma}$  depending on what is appropriate.

Two arrays of numbers are used for quantization/dequantization.

$A(QP=0,...,31)$

620,553,492,439,391,348,310,276,246,219,195,174,155,138,123,110,98,87,78,69,62,55,49,44,39,35,31,27,24,22,19,17

$B(QP=0,...,31)$

3881,4351,4890,5481,6154,6914,7761,8718,9781,10987,12339,13828,15523,17435,19561,21873,24552,27656,30847,34870,38807,43747,49103,54683,61694,68745,77615,89113,100253,109366,126635,141533

The relation between  $A()$  and  $B()$  is:  $A(QP) \times B(QP) \times 676^2 = 2^{40}$ .

It is assumed that a coefficient  $K$  is quantized in the following way:

$LEVEL = (K \times A(QP) + f \times 2^{20}) / 2^{20}$   $|f|$  is in the range (0-0.5) and  $f$  has the same sign as  $K$ .

Dequantization:

$K' = LEVEL \times B(QP)$

After inverse transform this results in pixel values that are  $2^{20}$  too high. A shift of 20 bits (with rounding) is therefore needed on the reconstruction side. The definition of transform and quantization is designed so that no overflow will occur with use of 32 bit arithmetic.

#### 4.3.3.4 Scanning and Quantization of 2x2 chrominance DC coefficients

$DDC()$  is quantized (and dequantized) separately resulting in  $LEVEL$ ,  $RUN$  and  $EOB$ . The scanning order is:  $DCC(0,0)$ ,  $DCC(1,0)$ ,  $DCC(0,1)$ ,  $DCC(1,1)$ . Inverse 2x2 transform as defined above is then performed after dequantization resulting in dequantized 4x4 DC coefficients:  $DC0'$   $DC1'$   $DC2'$   $DC3'$ .

Chrominance AC coefficients (4x4 based) are then quantized similarly to before. Notice that there are only 15 AC coefficients. The maximum size of  $RUN$  is therefore 14. However, for simplicity we use the same relation between  $LEVEL$ ,  $RUN$  and Code no. as defined for 'Simple scan' in TABLE 7.

#### 4.3.4 Use of 2-dimensional model for coefficient coding.

In the 3D model for coefficient coding (see H.263) there is no good use of a short codeword of 1 bit. On the other hand, with the use of 2D VLC plus End Of Block (EOB) (as used in H.261, H.262) and with the small block size, 1 bit for EOB is usually well matched to the VLC.

Furthermore, with the fewer non-zero coefficients per block, the advantage of using 3D VLC is reduced.

As a result we use a 2D model plus End Of Block (EOB) in the present model. This means that an event to be coded ( $RUN$ ,  $LEVEL$ ) consists of:

|         |                                                                              |
|---------|------------------------------------------------------------------------------|
| $RUN$   | which is the number of zero coefficients since the last nonzero coefficient. |
| $LEVEL$ | the size of the nonzero coefficient                                          |
| $EOB$   | signals that there are no more nonzero coefficients in the block             |

#### 4.4 Deblocking Filter

After the reconstruction of a macroblock a conditional filtering of this macroblock takes place, that effects the *boundaries* of the 4x4 block structure. For this purpose each 4x4 luminance block in a reconstructed macroblock is assigned a filtering **Strength**, which has the following value:

TABLE 4

## Assignment of filtering strength.

| 4x4 block condition                                    | Strength |
|--------------------------------------------------------|----------|
| Macro block is INTRA-coded, or member of SP-frame      | 2        |
| Macro block is non INTRA, but has nonzero coefficients | 1        |
| Else                                                   | 0        |

Filtering across 4x4 block boundary takes place if:

- Strength  $\neq 0$  on at least one of the sides of the boundary, or
- the absolute difference between one of the motion vector components of the two adjacent blocks is at least one integer pixel (four  $\frac{1}{4}$  pixels). For macro blocks in B-frames which are predicted "bidirectional" or "direct", this condition has to be valid for at least one of the involved vectors, or
- the adjacent motion vectors refer to different reference frames

Filtering takes place on a macroblock level. In a first step the 16 pel of the 4 vertical edges (horizontal filtering) of the 4x4 raster are filtered. Then the 4 horizontal edges (vertical filtering) follow. Note, that this process also affects the boundaries of the already reconstructed macroblocks above and to the right of the current macroblock.

Block boundaries of chrominance blocks always correspond to a block boundary of luminance blocks. Therefore corresponding **Strength** and **motion vector difference** for luminance are also used for chrominance boundaries. That is, for every 4x4 luminance block two 2x2 chrominance blocks are processed.

#### 4.4.1 Picture Content Dependent Parameters for Filtering

The set of eight pixels across a 4x4 block horizontal or vertical boundary is denoted as

$$p_4, p_3, p_2, p_1 \mid q_1, q_2, q_3, q_4$$

FIGURE 15

#### Illustration of de-blocking filtering.

with the actual boundary between  $p_1$  and  $q_1$ . Up to two pixels can be updated as a result of the filtering process on both sides of the boundary (that is at most  $p_2, p_1, q_1, q_2$ ). The number of updated pixels on both sides of the boundary depends on corresponding activity parameters  $a_p$  and  $a_q$ . They are limited by the so called overall activity parameter  $n$ :

$$n = \min(3, 4 - |p_1 - q_1| * \alpha(QP) / 128)$$

Activity parameter  $a_p$  is equal to the smallest integer  $i > 0$  not larger than  $n$  for which the inequality  $|p_i - p_{i+1}| \geq \beta(QP)$  holds. If no  $i < n$  satisfies the condition,  $a_p$  is set to  $n$ .  $a_q$  is obtained correspondingly. For the quantizer dependant threshold parameters  $\alpha$  and  $\beta$  see TABLE 5 below.

TABLE 5

#### QP dependent threshold parameters $\alpha$ and $\beta$ .

| QP       | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  | 11 | 12 | 13 | 14 | 15 |
|----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|----|----|----|----|
| $\alpha$ | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 122 | 96 | 75 | 59 | 47 | 37 |
| $\beta$  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 3   | 3   | 3   | 4  | 4  | 4  | 6  | 6  |
|          |     |     |     |     |     |     |     |     |     |     |     |    |    |    |    |    |
| QP       | 16  | 17  | 18  | 19  | 20  | 21  | 22  | 23  | 24  | 25  | 26  | 27 | 28 | 29 | 30 | 31 |

|          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| $\alpha$ | 29 | 23 | 18 | 15 | 13 | 11 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 3  | 2  | 2  |
| $\beta$  | 6  | 7  | 8  | 8  | 9  | 9  | 10 | 10 | 11 | 11 | 12 | 12 | 13 | 13 | 14 | 14 |

#### 4.4.2 Filtering Process

There are 3 conditions, that determine which of the boundary pixels are filtered:

- If  $(a_p > 1)$  and  $(a_q > 1)$  the pixels  $p_1$  and  $q_1$  are filtered by:
 
$$\Delta = \text{Clip}(-c, c, ((q_1 - p_1) << 2 + (p_2 - q_2) + 4) >> 3)$$

$$P_1 = \text{Clip}(0, 255, (p_1 + \Delta))$$

$$Q_1 = \text{Clip}(0, 255, (q_1 - \Delta))$$
- If  $(a_p = 3)$  and  $(a_q > 1)$   $p_2$  is filtered by:
 
$$\Delta = \text{Clip}(-c_p, c_p, (p_3 + p_1 - p_2 << 1) >> 1)$$

$$P_2 = \text{Clip}(0, 255, (p_2 + \Delta))$$
- If  $(a_p > 1)$  and  $(a_q = 3)$   $q_2$  is filtered by:
 
$$\Delta = \text{Clip}(-c_q, c_q, (q_3 + q_1 - q_2 << 1) >> 1)$$

$$Q_2 = \text{Clip}(0, 255, (q_2 + \Delta))$$

Here Clip() denotes a clipping function with the parameters Clip( Min, Max, Value) with the clipping constants:

$$c_p = \text{ClipTbl}(QP, \text{Strength}_p)$$

$$c_q = \text{ClipTbl}(QP, \text{Strength}_q)$$

$$c = (c_p + c_q + a_p + a_q) / 2$$

TABLE 6

ClipTbl (QP, Strength):

|               |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|---------------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| QP            | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| ClipTbl(qp,0) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| ClipTbl(qp,1) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 2  | 2  | 2  | 2  | 3  | 3  | 3  | 3  | 4  | 5  | 5  |
| ClipTbl(qp,2) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 2  | 2  | 2  | 2  | 3  | 3  | 3  | 4  | 4  | 5  | 5  | 5  | 7  | 8  | 9  |

#### 4.4.3 Stronger filtering for intra coded macroblocks

In case of intra coding and areas in the picture with little texture the subjective quality can be improved by stronger filtering across macroblock boundaries (that is: *not* 4x4 block boundaries). This stronger filtering is performed only if one or both of the adjacent macroblocks are intra coded and  $a_p = 3$  and  $a_q = 3$  and  $1 < |p_1 - q_1| < QP / 4$ .

In this case luminance filtering is performed using the formulas shown below. For chrominance pixels  $p_3$  to  $q_3$  are not filtered.

$$P_1 = (25 * p_3 + 26 * p_2 + 26 * p_1 + 26 * q_1 + 25 * q_2 + 64) / 128$$

$$P_2 = (25 * p_4 + 26 * p_3 + 26 * p_2 + 26 * P_1 + 25 * q_1 + 64) / 128$$

$$P_3 = (26 * p_4 + 51 * p_3 + 26 * P_2 + 25 * P_1 + 64) / 128$$

$$Q_1 = (25 * p_2 + 26 * p_1 + 26 * q_1 + 26 * q_2 + 25 * q_3 + 64) / 128$$

$$Q_2 = (25 * p_1 + 26 * Q_1 + 26 * q_2 + 26 * q_3 + 25 * q_4 + 64) / 128$$

$$Q_3 = (25 * Q_1 + 26 * Q_2 + 51 * q_3 + 26 * q_4 + 64) / 128$$

Note, that when filtering across the next 4x4 block boundary, pixel  $q_3$  may get overwritten. To compensate the effect  $Q_3$  is stored and used as  $P_2$  when filtering across the next 4x4 block boundary.

## 4.5 Entropy Coding

### 4.5.1 Universal Variable Length Coding (UVLC)

In the default entropy coding mode, a universal VLC is used to code all syntax. The table of codewords may be written in the following compressed form.

```

      1
    0 x0 1
  0 x1 0 x0 1
0 x2 0 x1 0 x0 1
0 x3 0 x2 0 x1 0 x0 1
.....

```

where  $x_n$  take values 0 or 1. We will sometimes refer to a codeword with its length in bits ( $L$ ) and  $\text{INFO} = x_n \dots x_1 x_0$ . Notice that the number of bits in  $\text{INFO}$  is  $L/2$  (division by truncation). The codewords are numbered from 0 and upwards. The definition of the numbering is:

$\text{Code\_number} = 2^{L/2} + \text{INFO} - 1$  ( $L/2$  use division with truncation.  $\text{INFO} = 0$  when  $L = 1$ )

Some of the first code numbers and codewords are written explicitly in the table below. As an example, for the code number 5,  $L = 5$  and  $\text{INFO} = 10$  (binary) = 2 (decimal)

Code number      Codewords in explicit form

|    |               |
|----|---------------|
| 0  | 1             |
| 1  | 0 0 1         |
| 2  | 0 1 1         |
| 3  | 0 0 0 0 1     |
| 4  | 0 0 0 1 1     |
| 5  | 0 1 0 0 1     |
| 6  | 0 1 0 1 1     |
| 7  | 0 0 0 0 0 0 1 |
| 8  | 0 0 0 0 0 1 1 |
| 9  | 0 0 0 1 0 0 1 |
| 10 | 0 0 0 1 0 1 1 |
| 11 | 0 1 0 0 0 0 1 |
| .. | . . . . .     |

When  $L$  and  $\text{INFO}$  is known, the regular structure of the table makes it easy to create a codeword bit by bit. Similarly, a decoder may easily read bit by bit until the last "1" which gives the end of the codeword.  $L$  and  $\text{INFO}$  is then readily available. For each parameter to be coded, there is a conversion rule from the parameter value to the code number (or  $L$  and  $\text{INFO}$ ). TABLE 7 lists the connection between code number and most of the parameters used in the present coding method.

TABLE 7

## Connection between codeword number and parameter values.

| Code number | RUN | MB_Type            |                    | Intra_pred_mode <sup>1</sup> |       | MVD<br>DQUANT | CBP   |       | Tcoeff_chroma_DC <sup>2</sup> |     | Tcoeff_chroma_AC <sup>2</sup><br>Tcoeff_luma <sup>2</sup><br>Simple scan |     | Tcoeff_luma <sup>2</sup><br>Double scan |     |
|-------------|-----|--------------------|--------------------|------------------------------|-------|---------------|-------|-------|-------------------------------|-----|--------------------------------------------------------------------------|-----|-----------------------------------------|-----|
|             |     | Intra              | Inter              | Prob0                        | Prob1 |               | Intra | Inter | Level                         | Run | Level                                                                    | Run | Level                                   | Run |
| 0           | 0   | Intra4x4           | 16x16              | 0                            | 0     | 0             | 47    | 0     | EOB                           | -   | EOB                                                                      | -   | EOB                                     | -   |
| 1           | 1   | 0,0,0 <sup>3</sup> | 16x8               | 1                            | 0     | 1             | 31    | 16    | 1                             | 0   | 1                                                                        | 0   | 1                                       | 0   |
| 2           | 2   | 1,0,0              | 8x16               | 0                            | 1     | -1            | 15    | 1     | -1                            | 0   | -1                                                                       | 0   | -1                                      | 0   |
| 3           | 3   | 2,0,0              | 8x8                | 0                            | 2     | 2             | 0     | 2     | 2                             | 0   | 1                                                                        | 1   | 1                                       | 1   |
| 4           | 4   | 3,0,0              | 8x4                | 1                            | 1     | -2            | 23    | 4     | -2                            | 0   | -1                                                                       | 1   | -1                                      | 1   |
| 5           | 5   | 0,1,0              | 4x8                | 2                            | 0     | 3             | 27    | 8     | 1                             | 1   | 1                                                                        | 2   | 2                                       | 0   |
| 6           | 6   | 1,1,0              | 4x4                | 3                            | 0     | -3            | 29    | 32    | -1                            | 1   | -1                                                                       | 2   | -2                                      | 0   |
| 7           | 7   | 2,1,0              | Intra4x4           | 2                            | 1     | 4             | 30    | 3     | 3                             | 0   | 2                                                                        | 0   | 1                                       | 2   |
| 8           | 8   | 3,1,0              | 0,0,0 <sup>3</sup> | 1                            | 2     | -4            | 7     | 5     | -3                            | 0   | -2                                                                       | 0   | -1                                      | 2   |
| 9           | 9   | 0,2,0              | 1,0,0              | 0                            | 3     | 5             | 11    | 10    | 2                             | 1   | 1                                                                        | 3   | 3                                       | 0   |
| 10          | 10  | 1,2,0              | 2,0,0              | 0                            | 4     | -5            | 13    | 12    | -2                            | 1   | -1                                                                       | 3   | -3                                      | 0   |
| 11          | 11  | 2,2,0              | 3,0,0              | 1                            | 3     | 6             | 14    | 15    | 1                             | 2   | 1                                                                        | 4   | 4                                       | 0   |
| 12          | 12  | 3,2,0              | 0,1,0              | 2                            | 2     | -6            | 39    | 47    | -1                            | 2   | -1                                                                       | 4   | -4                                      | 0   |
| 13          | 13  | 0,0,1              | 1,1,0              | 3                            | 1     | 7             | 43    | 7     | 1                             | 3   | 1                                                                        | 5   | 5                                       | 0   |
| 14          | 14  | 1,0,1              | 2,1,0              | 4                            | 0     | -7            | 45    | 11    | -1                            | 3   | -1                                                                       | 5   | -5                                      | 0   |
| 15          | 15  | 2,0,1              | 3,1,0              | 5                            | 0     | 8             | 46    | 13    | 4                             | 0   | 3                                                                        | 0   | 1                                       | 3   |
| 16          | 16  | 3,0,1              | 0,2,0              | 4                            | 1     | -8            | 16    | 14    | -4                            | 0   | -3                                                                       | 0   | -1                                      | 3   |
| 17          | 17  | 0,1,1              | 1,2,0              | 3                            | 2     | 9             | 3     | 6     | 3                             | 1   | 2                                                                        | 1   | 1                                       | 4   |
| 18          | 18  | 1,1,1              | 2,2,0              | 2                            | 3     | -9            | 5     | 9     | -3                            | 1   | -2                                                                       | 1   | -1                                      | 4   |
| 19          | 19  | 2,1,1              | 3,2,0              | 1                            | 4     | 10            | 10    | 31    | 2                             | 2   | 2                                                                        | 2   | 2                                       | 1   |
| 20          | 20  | 3,1,1              | 0,0,1              | 0                            | 5     | -10           | 12    | 35    | -2                            | 2   | -2                                                                       | 2   | -2                                      | 1   |
| 21          | 21  | 0,2,1              | 1,0,1              | 1                            | 5     | 11            | 19    | 37    | 2                             | 3   | 1                                                                        | 6   | 3                                       | 1   |
| 22          | 22  | 1,2,1              | 2,0,1              | 2                            | 4     | -11           | 21    | 42    | -2                            | 3   | -1                                                                       | 6   | -3                                      | 1   |
| 23          | 23  | 2,2,1              | 3,0,1              | 3                            | 3     | 12            | 26    | 44    | 5                             | 0   | 1                                                                        | 7   | 6                                       | 0   |
| 24          | 24  | 3,2,1              | 0,1,1              | 4                            | 2     | -12           | 28    | 33    | -5                            | 0   | -1                                                                       | 7   | -6                                      | 0   |
| 25          | 25  |                    | 1,1,1              | 5                            | 1     | 13            | 35    | 34    | 4                             | 1   | 1                                                                        | 8   | 7                                       | 0   |
| 26          | 26  |                    | 2,1,1              | 5                            | 2     | -13           | 37    | 36    | -4                            | 1   | -1                                                                       | 8   | -7                                      | 0   |
| 27          | 27  |                    | 3,1,1              | 4                            | 3     | 14            | 42    | 40    | 3                             | 2   | 1                                                                        | 9   | 8                                       | 0   |
| 28          | 28  |                    | 0,2,1              | 3                            | 4     | -14           | 44    | 39    | -3                            | 2   | -1                                                                       | 9   | -8                                      | 0   |
| 29          | 29  |                    | 1,2,1              | 2                            | 5     | 15            | 1     | 43    | 3                             | 3   | 4                                                                        | 0   | 9                                       | 0   |
| 30          | 30  |                    | 2,2,1              | 3                            | 5     | -15           | 2     | 45    | -3                            | 3   | -4                                                                       | 0   | -9                                      | 0   |
| 31          | 31  |                    | 3,2,1              | 4                            | 4     | 16            | 4     | 46    | 6                             | 0   | 5                                                                        | 0   | 10                                      | 0   |
| 32          | 32  |                    |                    | 5                            | 3     | -16           | 8     | 17    | -6                            | 0   | -5                                                                       | 0   | -10                                     | 0   |
| 33          | 33  |                    |                    | 5                            | 4     | 17            | 17    | 18    | 5                             | 1   | 3                                                                        | 1   | 4                                       | 1   |
| 34          | 34  |                    |                    | 4                            | 5     | -17           | 18    | 20    | -5                            | 1   | -3                                                                       | 1   | -4                                      | 1   |
| 35          | 35  |                    |                    | 5                            | 5     | 18            | 20    | 24    | 4                             | 2   | 3                                                                        | 2   | 2                                       | 2   |
| 36          | 36  |                    |                    |                              |       | -18           | 24    | 19    | -4                            | 2   | -3                                                                       | 2   | -2                                      | 2   |
| 37          | 37  |                    |                    |                              |       | 19            | 6     | 21    | 4                             | 3   | 2                                                                        | 3   | 2                                       | 3   |
| 38          | 38  |                    |                    |                              |       | -19           | 9     | 26    | -4                            | 3   | -2                                                                       | 3   | -2                                      | 3   |
| 39          | 39  |                    |                    |                              |       | 20            | 22    | 28    | 7                             | 0   | 2                                                                        | 4   | 2                                       | 4   |
| 40          | 40  |                    |                    |                              |       | -20           | 25    | 23    | -7                            | 0   | -2                                                                       | 4   | -2                                      | 4   |
| 41          | 41  |                    |                    |                              |       | 21            | 32    | 27    | 6                             | 1   | 2                                                                        | 5   | 2                                       | 5   |
| 42          | 42  |                    |                    |                              |       | -21           | 33    | 29    | -6                            | 1   | -2                                                                       | 5   | -2                                      | 5   |
| 43          | 43  |                    |                    |                              |       | 22            | 34    | 30    | 5                             | 2   | 2                                                                        | 6   | 2                                       | 6   |
| 44          | 44  |                    |                    |                              |       | -22           | 36    | 22    | -5                            | 2   | -2                                                                       | 6   | -2                                      | 6   |
| 45          | 45  |                    |                    |                              |       | 23            | 40    | 25    | 5                             | 3   | 2                                                                        | 7   | 2                                       | 7   |
| 46          | 46  |                    |                    |                              |       | -23           | 38    | 38    | -5                            | 3   | -2                                                                       | 7   | -2                                      | 7   |
| 47          | 47  |                    |                    |                              |       | 24            | 41    | 41    | 8                             | 0   | 2                                                                        | 8   | 11                                      | 0   |
| ..          | ..  |                    |                    |                              |       | ..            | ..    | ..    | ..                            | ..  | ..                                                                       | ..  | ..                                      | ..  |

<sup>1</sup> Prob0 and Prob1 defines the Intra prediction modes of two blocks relative to the prediction of prediction modes (see details in the section for Intra coding).

<sup>2</sup> For the entries above the horizontal line, the table is needed for relation between code number and Level/Run/EOB. For the remaining Level/Run combination there is a simple rule. The Level/Run combinations are assigned a code number according to the following priority: 1) sign of Level (+ -) 2) Run (ascending) 3) absolute value of Level (ascending).

<sup>3</sup> 16x16 based intra mode. The 3 numbers refer to values for (Imode,AC,nc) - see 0.

## 4.5.2 Context-based Adaptive Binary Arithmetic Coding (CABAC)

### 4.5.2.1 Overview

The entropy coding method of context-based adaptive binary arithmetic coding (CABAC) has three distinct elements compared to the default entropy coding method using a fixed, universal table of variable length codes (UVLC):

- *Context modeling* provides estimates of conditional probabilities of the coding symbols. Utilizing suitable context models, given inter-symbol redundancy can be exploited by switching between different probability models according to already coded symbols in the neighborhood of the current symbol to encode.
- *Arithmetic codes* permit non-integer number of bits to be assigned to each symbol of the alphabet. Thus the symbols can be coded almost at their entropy rate. This is extremely beneficial for symbol probabilities much greater than 0.5, which often occur with efficient context modeling. In this case, a variable length code has to spend at least one bit in contrast to arithmetic codes, which may use a fraction of one bit.
- *Adaptive* arithmetic codes permit the entropy coder to adapt itself to non-stationary symbol statistics. For instance, the statistics of motion vector magnitudes vary over space and time as well as for different sequences and bit-rates. Hence, an adaptive model taking into account the cumulative probabilities of already coded motion vectors leads to a better fit of the arithmetic codes to the current symbol statistics.

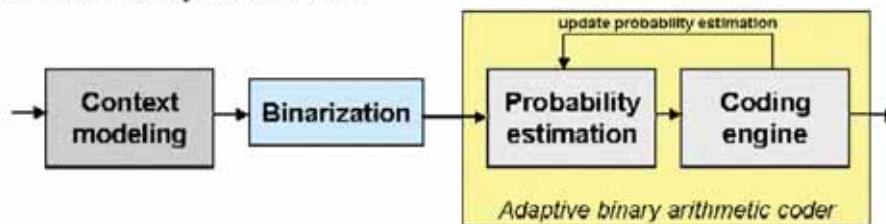


FIGURE 16

#### Generic block diagram of CABAC entropy coding scheme

Next we give a short overview of the main coding elements of the CABAC entropy coding scheme as depicted in FIGURE 16. Suppose a symbol related to an arbitrary syntax element is given, then, in a first step, a suitable model is chosen according to a set of past observations. This process of constructing a model conditioned on neighboring symbols is commonly referred to as *context modeling* and is the first step in the entropy coding scheme. The particular context models that are designed for each given syntax model are described in detail in Section 4.5.2.2 and Section 4.5.2.3. If a given symbol is non-binary valued, it will be mapped onto a sequence of binary decisions, so-called *bins*, in a second step. The actual *binarization* is done according to a given binary tree, as specified in Section 4.5.2.5. Finally, each binary decision is encoded with the *adaptive binary arithmetic coding* (AC) engine using the *probability estimates*, which have been provided either by the context modeling stage or by the binarization process itself. The provided models serve as a probability estimation of the related bins. After encoding of each bin, the related model will be updated with the encoded binary symbol. Hence, the model keeps track of the actual statistics.

### 4.5.2.2 Context Modeling for Coding of Motion and Mode Information

In this section we describe in detail the context modeling of our adaptive coding method for the syntax elements macroblock type (MB\_type), motion vector data (MVD) and reference frame parameter (Ref\_frame).

#### 4.5.2.2.1 Context Models for Macroblock Type

We distinguish between MB\_type for intra and inter frames. In the following, we give a description of the context models which have been designed for coding of the MB\_type information in both cases. The

subsequent process of mapping a non-binary valued MB\_type symbol to a binary sequence in the case of inter frames will be given in detail in section 4.5.2.5.

#### 4.5.2.2.1.1 Intra Pictures

For intra pictures, there are two possible modes for each macroblock, i.e. Intra4x4 and Intra16x16, so that signalling the mode information is reduced to transmitting a binary decision. Coding of this binary decision for a given macroblock is performed by means of context-based arithmetic coding, where the context of a current MB\_type C is build by using the MB\_types A and B1 of neighboring macroblocks (as depicted in FIGURE 17) which are located in the causal past of the current coding event C. Since A and B are binary decisions, we define the actual context number  $ctx\_mb\_type\_intra(C)$  of C by  $ctx\_mb\_type\_intra(C) = A + B$ , which results in three different contexts according to the 4 possible combinations of MB\_type states for A and B.

In the case of MB\_type Intra16x16, there are three additional parameters related to the chosen intra prediction mode, the occurrence of significant AC-coefficients and the coded block pattern for the chrominance coefficients, which have to be signaled. In contrast to the current test model, this information is not included in the mode information, but is coded separately by using distinct models as described in Section 4.5.2.3.

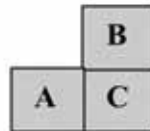
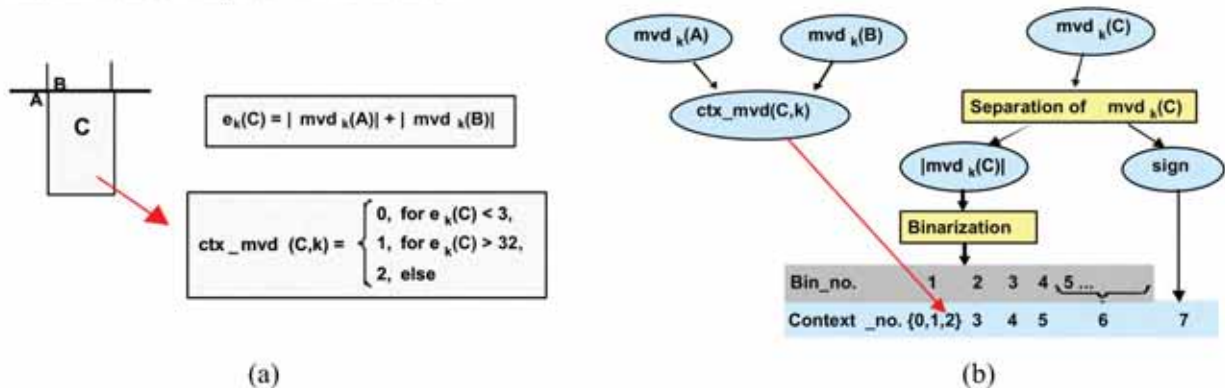


FIGURE 17

**Neighboring symbols A and B used for conditional coding of a current symbol C.**

#### 4.5.2.2.1.2 P- and B-Pictures

Currently there are 10 different macroblock types for P-frames and 18 different macroblock types for B-frames, provided that the additional information of the 16x16 Intra mode is not considered as part of the mode information. Coding of a given MB\_type information C is done similar to the case of intra frames by using a context model which involves the MB\_type information A and B of previously encoded (or decoded) macroblocks (cp. FIGURE 17). However, here we only use the information whether the neighboring macroblocks of the given macroblock are of type Skip (P-frame) or Direct (B-frame), such that the actual context number  $ctx\_mb\_type\_inter(C)$  is given in C-style notation for P-frame coding by  $ctx\_mb\_type\_inter(C) = ((A==Skip)?0:1) + ((B==Skip)?0:1)$  and by  $ctx\_mb\_type\_inter(C) = ((A==Direct)?0:1) + ((B==Direct)?0:1)$  for B-frame coding. Thus, we obtain 3 different contexts, which, however, are only used for coding of the first bin of the binarization  $b(C)$  of C, where the actual binarization of C will be performed as outlined in Section 4.5.2.5. For coding the second bin, a separate model is provided and for all remaining bins of  $b(C)$  two additional models are used as further explained in Sect. 4.5.2.3. Thus, a total number of 6 different models are supplied for coding of macroblock type information relating to P- and B-frames.



<sup>1</sup> For mathematical convenience the meaning of the variables A, B and C is context dependent.

FIGURE 18

**Illustration of the encoding process for a given residual motion vector component  $mvd_k(C)$  of a block C: (a) Context selection rule. (b) Separation of  $mvd_k(C)$  into sign and magnitude, binarization of the magnitude and assignment of context models to  $bin\_nos$ .**

#### 4.5.2.2.2 Context Models for Motion Vector Data

Motion vector data consists of residual vectors obtained by applying motion vector prediction. Thus, it is a reasonable approach to build a model conditioned on the local prediction error. A simple measure of the local prediction error at a given block C is given by evaluating the L1-norm  $e_k(A, B) = |mvd_k(A)| + |mvd_k(B)|$  of two neighboring motion vector prediction residues  $mvd_k(A)$  and  $mvd_k(B)$  for each component of a motion vector residue  $mvd_k(C)$  of a given block, where A and B are neighboring blocks of block C, as shown in FIGURE 18 (a). If one of the neighboring blocks belongs to an adjacent macroblock, we take the residual vector component of the leftmost neighboring block in the case of the upper block B, and in the case of the left neighboring block A we use the topmost neighboring block. If one of the neighboring blocks is not available, because, for instance, the current block is at the picture boundary, we discard the corresponding part of  $e_k$ . By using  $e_k$ , we now define a context model  $ctx\_mvd(C, k)$  for the residual motion vector component  $mvd_k(C)$  consisting of three different context models:

$$ctx\_mvd(C, k) = \begin{cases} 0, & e_k(C) < 3, \\ 1, & e_k(C) > 32, \\ 2, & \text{else.} \end{cases}$$

For the actual coding process, we separate  $mvd_k(C)$  in sign and modulus, where only the first bin of the binarization of the modulus  $|mvd_k(C)|$  is coded using the context models  $ctx\_mvd(C, k)$ . For the remaining bins of the modulus, we have 4 additional models: one for the second, one for the third bin, one for the fourth, and one model for all remaining bins. In addition, the sign coding routine is provided with a separate model. This results in a total sum of 8 different models for each vector component (see FIGURE 18).

In the case of B-frame coding, an additional syntax element has to be signaled when the bi-directional mode is chosen. This element represents the block size ( $Blk\_size$ ), which is chosen for forward or backward motion prediction. The related code number value ranges between 0 and 6 according to the 7 possible block shapes in FIGURE 2. Coding of  $Blk\_size$  is done by using the binarization of the  $P\_MB\_type$  as described in Section 4.5.2.3.

#### 4.5.2.2.3 Context Models for Reference Frame Parameter

If the option of temporal prediction from more than one reference frame is enabled, the chosen reference frame for each macroblock must be signaled. Given a macroblock and its reference frame parameter as a symbol C according to the definition in Sect. **Error! Reference source not found.**, a context model is built by using symbols A and B of the reference frame parameter belonging to the two neighboring macroblocks (cp. FIGURE 17). The actual context number of C is then defined by  $ctx\_ref\_frame(C) = ((A==0)?0:1) + 2*((B==0)?0:1)$ , such that  $ctx\_ref\_frame(C)$  indicates one of four models used for coding of the first bin of the binary equivalent  $b(C)$  of C. Two additional models are given for the second bin and all remaining bins of  $b(C)$ , which sums up to a total number of six different models for the reference frame information.

### 4.5.2.3 Context Modeling for Coding of Texture Information

This section provides detailed information about the context models used for the syntax elements of coded block pattern (CBP), intra prediction mode (IPRED) and (RUN, LEVEL) information.

#### 4.5.2.3.1 Context Models for Coded Block Pattern

Except for MB\_type Intra16x16, the context modeling for the coded block pattern is treated as follows. There are 4 luminance CBP bits belonging to 4 8x8 blocks in a given macroblock. Let C denote such a Y-CBP bit, then we define  $ctx\_cbp\_luma(C) = A + 2*B$ , where A and B are Y-CBP bits of the neighboring 8x8 blocks, as depicted in FIGURE 17. The remaining 2 bits of CBP are related to the chrominance coefficients. In our coding approach, these bits are translated into two dependant binary decisions, such that, in a first step, we send a bit *cbp\_chroma\_sig* which signals whether there are significant chrominance coefficients at all. The related context model is of the same kind as that of the Y-CBP bits, i.e.  $ctx\_cbp\_chroma\_sig(C) = A + 2*B$ , where A and B are now notations for the corresponding *cbp\_chroma\_sig* bits of neighboring macroblocks. If *cbp\_chroma\_sig* = 1 (non-zero chroma coefficients exist), a second bit *cbp\_chroma\_ac* related to the significance of AC chrominance coefficients has to be signalled. This is done by using a context model conditioned on the *cbp\_chroma\_ac* decisions A and B of neighboring macroblocks, such that  $ctx\_cbp\_chroma\_AC(C) = A + 2*B$ . Note, that due to the different statistics there are different models for Intra and Inter macroblocks, so that the total number of different models for CBP amounts to  $2*3*4=24$ . For the case of MB\_type Intra16x16, there are three additional models, one for the binary AC decision and two models for each of the two chrominance CBP bits.

#### 4.5.2.3.2 Context Models for Intra Prediction Mode

In Intra4x4 mode, coding of the intra prediction mode C of a given block is conditioned on the intra prediction mode of the previous block A to the left of C (cp. FIGURE 17). In fact, it is not the prediction mode number itself which is signaled and which is used for conditioning but rather its predicted order similar as it is described in Sect. 0. There are 6 different prediction modes and for each mode, two different models are supplied: one for the first bin of the binary equivalent of C and the other for all remaining bins. Together with two additional models for the two bits of the prediction modes of MB\_type Intra16x16 (in binary representation), a total number of 14 different models for coding of intra prediction modes is given.

#### 4.5.2.3.3 Context Models for Run/Level

Coding of (RUN, LEVEL) pairs is conditioned on the scanning mode, the DC/AC block type, the luminance/chrominance, and the intra/inter macroblock decision. Thus, a total number of 9 different block types are given according to TABLE 8, which, in turn, results in 9 different contexts. In contrast to the current test model, RUN and LEVEL are coded separately in our coding approach, as described in the following two subsections.

**TABLE 8**

**Numbering of the different context models used for coding of RUN and LEVEL**

| <i>Ctx_run_level</i> | Block Type         |
|----------------------|--------------------|
| 0                    | Double Scan        |
| 1                    | Single Scan, Inter |
| 2                    | Single Scan, Intra |
| 3                    | Intra16x16, DC     |
| 4                    | Intra16x16, AC     |
| 5                    | Chroma, DC, Inter  |
| 6                    | Chroma, DC, Intra  |
| 7                    | Chroma, AC, Inter  |
| 8                    | Chroma, AC, Intra  |

#### 4.5.2.3.3.1 Context-based Coding of LEVEL Information

For a given block C, the LEVEL information is first separated into sign and magnitude. According to its context  $ctx\_run\_level(C)$  four different models are chosen, where one model is used for coding of the sign information and the remaining 3 models are used for the first, the second and all remaining bins of

the binarization of LEVEL. If LEVEL≠0 (EOB), the corresponding RUN is coded in a subsequent coding routine, as described in the following section.

#### 4.5.2.3.3.2 Context-based Coding of RUN Information

For each context  $ctx\_run\_level(C)$  two separate models are provided for the coding of RUN; one model for the first bin and the second model for all remaining bins of the binary sequence related to RUN.

#### 4.5.2.4 Context Modeling for Coding of Dquant

For a given macroblock, the value of Dquant is first mapped to a positive value using the arithmetic wrap as described in section 3.4.7. This wrapped value  $C$  is then coded conditioned on the corresponding Dquant  $A$  of the left neighboring macroblock, such that that  $ctx\_dquant(C)=(A \neq 0)$ ; This results in 2 context models for the first bin. Two additional models are used for the second and all remaining bins of the binarized value  $C$ . Thus, a total sum of four context models are used for Dquant.

TABLE 9

Binarization by means of the unary code tree

| Code symbol | Binarization |   |   |   |   |   |   |
|-------------|--------------|---|---|---|---|---|---|
| 0           | 1            |   |   |   |   |   |   |
| 1           | 0            | 1 |   |   |   |   |   |
| 2           | 0            | 0 | 1 |   |   |   |   |
| 3           | 0            | 0 | 0 | 1 |   |   |   |
| 4           | 0            | 0 | 0 | 0 | 1 |   |   |
| 5           | 0            | 0 | 0 | 0 | 0 | 1 |   |
| 6           | 0            | 0 | 0 | 0 | 0 | 0 | 1 |
| ...         | .            | . | . | . | . | . | . |
| Bin_no.     | 1            | 2 | 3 | 4 | 5 | 6 | 7 |

#### 4.5.2.5 Binarization of Non-Binary Valued Symbols

A non-binary valued symbol will be decomposed into a sequence of binary decisions. Except for the MB\_type syntax element we use the binarization given by the unary code tree in TABLE 9.

TABLE 10

(a) Binarization for P-frame MB\_type and (b) for B-frame MB\_type

For the binary decomposition of the MB\_type symbols of P- or B-frames, which are of limited range (0 ... 9) or (0 ... 17) respectively, an alternative binarization is used, which is shown in TABLE 10.

#### 4.5.2.6 Adaptive Binary Arithmetic Coding

At the beginning of the overall encoding of a given frame the probability models associated with all 126 different contexts are initialized with a pre-computed start distribution. For each symbol to encode the frequency count of the related binary decision is updated, thus providing a new probability estimate for the next coding decision. However, when the total number of occurrences of a given model exceeds a pre-defined threshold, the frequency counts will be scaled down. This periodical rescaling exponentially weighs down past observations and helps to adapt to the non-stationarity of a

| P_MB_type | Binarization |
|-----------|--------------|
| 0         | 0            |
| 1         | 1 0 0        |
| 2         | 1 0 1        |
| 3         | 1 1 0 0 0    |
| 4         | 1 1 0 0 1    |
| 5         | 1 1 0 1 0    |
| 6         | 1 1 0 1 1    |
| 7         | 1 1 1 0 0    |
| 8         | 1 1 1 0 1    |
| 9         | 1 1 1 1 0    |
| Bin_no    | 1 2 3 4 5    |

| B_MB_type |
|-----------|
| 0         |
| 1         |
| 2         |
| 3         |
| 4         |
| 5         |
| 6         |
| 7         |
| .         |
| 17        |
| Bin_no    |

source. The binary arithmetic coding engine used in our presented approach is a straightforward implementation similar to that given in Witten et al., "Arithmetic Coding for Data Compression", *Comm. of the ACM*, 30 (6), 1987, pp.520-541.

## 5 B-pictures

### 5.1 Introduction

Temporal scalability is achieved using bi-directionally predicted pictures, or B pictures. The use of B pictures is indicated in PTYPE. The B pictures are predicted from either or both the previous and subsequent reconstructed pictures to achieve improved coding efficiency as compared to that of P pictures. FIGURE 19 illustrates the predictive structure with two B pictures inserted between I/P pictures.

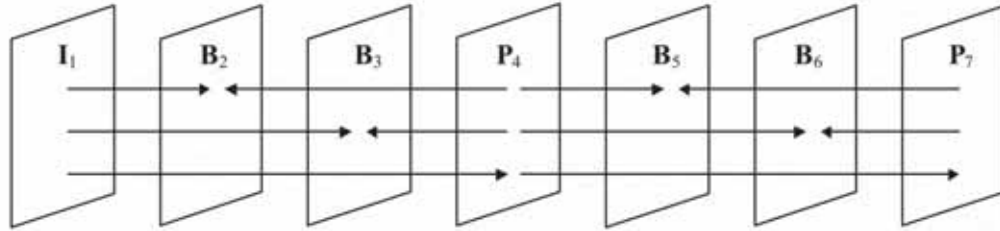


FIGURE 19

**Illustration of B picture concept.**

The location of B pictures in the bitstream is in a data-dependence order rather than in temporal order. Pictures that are dependent on other pictures shall be located in the bitstream after the pictures on which they depend. For example, as illustrated in Figure A.1, B<sub>2</sub> and B<sub>3</sub> are dependent on I<sub>1</sub> and P<sub>4</sub>, and B<sub>5</sub> and B<sub>6</sub> are dependent on P<sub>4</sub> and P<sub>7</sub>. Therefore the bitstream syntax order of the encoded pictures would be I<sub>1</sub>, P<sub>4</sub>, B<sub>2</sub>, B<sub>3</sub>, P<sub>7</sub>, B<sub>5</sub>, B<sub>6</sub>, ... However, the display order of the decoded pictures should be I<sub>1</sub>, B<sub>2</sub>, B<sub>3</sub>, P<sub>4</sub>, B<sub>5</sub>, B<sub>6</sub>, P<sub>7</sub>, ... The difference between the bitstream order of encoded pictures and the display order of decoded pictures will increase latency and memory to buffer the P pictures.

There is no limit to the number of B pictures that may be inserted between each I/P picture pair. The maximum number of such pictures may be signaled by external means (for example Recommendation H.245). The picture height, width, and pixel aspect ratio of a B picture shall always be equal to those of its temporally subsequent reference picture.

The B pictures support multiple reference frame prediction. The maximum number of previous reference frames that may be used for prediction in B pictures must be less than or equal to the number of reference frames used in the immediately following P frame, and it may be signaled by external means (for example Recommendation H.245). The use of this mode is indicated by PTYPE.

### 5.2 Five Prediction modes

There are five different prediction modes supported by B pictures. They are direct, forward, backward, bi-directional and the intra prediction modes. Both direct mode and bi-directional mode are bi-directional prediction. The only difference is that the bi-directional mode uses separate motion vectors for forward and backward prediction, whereas the forward and backward motion vectors of the direct mode are derived from the motion vectors used in the corresponding macroblocks of the subsequent reference frame. In the direct mode, the same number of motion vectors as in the co-located macroblock of the temporal following reference picture is used. To calculate prediction blocks for the direct and bi-directional prediction mode, the forward and backward motion vectors are used to obtain appropriate blocks from reference frames and then these blocks are averaged by dividing the sum of the two prediction blocks by two.

Forward prediction means prediction from a previous reference picture, and backward prediction means prediction from a temporally subsequent reference picture.

The intra prediction means to encode the macroblock by using intra coding.

### 5.3 Syntax

Some additional syntax elements are needed for B pictures. The structure of B picture related fields is shown in FIGURE 20. On the Ptype, two picture types shall be added to include B pictures with and without multiple reference frame prediction. On the MB\_type, different macroblock types are defined to indicate the different prediction types for B pictures. The fields of Blk\_size, MVDFW, and MVDBW are inserted to enable bi-directional prediction. These fields replace the syntax element MVD.

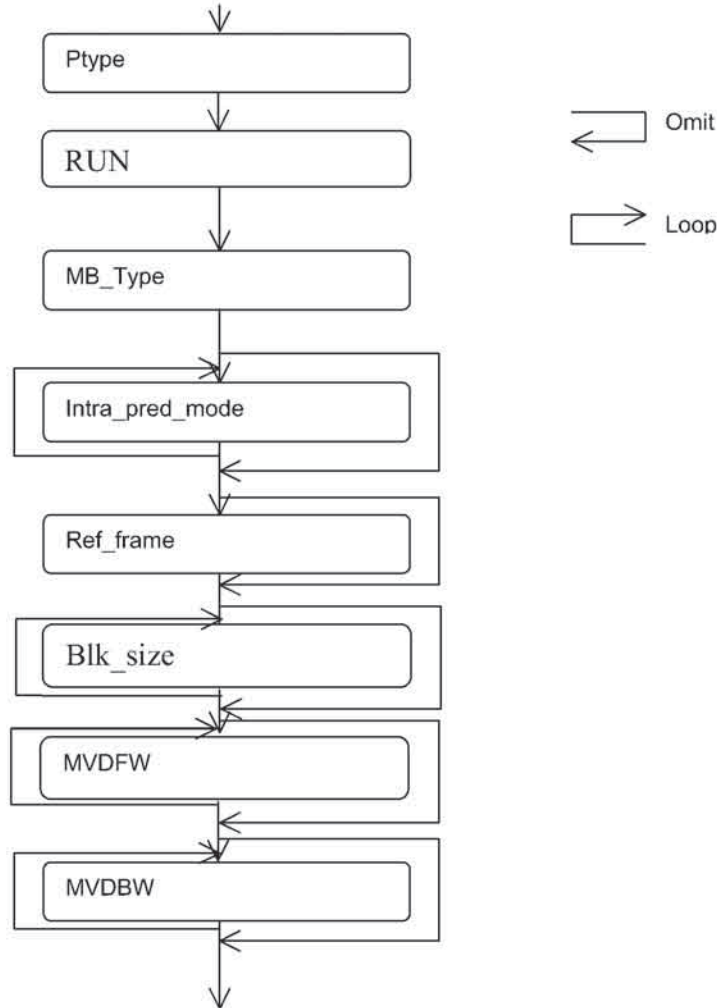


FIGURE 20

Syntax diagram for B pictures.

#### 5.3.1 Picture type (Ptype) and RUN

See section 2.1 for definition.

#### 5.3.2 Macro block type (MB\_type)

The MB\_type indicates the prediction mode and block size used to encode each macroblock. As mentioned earlier, five different prediction modes are supported in B pictures. For the forward, backward and bi-directional prediction modes, a macroblock is predicted from either or both of the previous and subsequent pictures with block size NxM. TABLE 11 shows the macroblock types and the included data elements for B pictures.

In "Direct" prediction type, no motion vector data is transmitted.

The “Forward\_NxM” indicates that the macroblock is predicted from a previous picture with block size NxM. The “backward\_NxM” indicates that the macroblock is predicted from a subsequent picture with block size NxM. For each NxM block, motion vector data is provided. Therefore, depending on N and M, up to 16 sets of motion vector data have to be transmitted for a macroblock.

For the “Bi-directional” prediction type, the parameter Blk\_size is used to indicate the block size used for forward and backward motion prediction (the Blk\_size field is described in detail below). Both forward and backward motion vector data sets are transmitted. Depending on the block size indicated in Blk\_size, up to 16 fields of motion vector data are transmitted for each of forward and backward prediction for a macroblock.

The “Intra\_4x4” and “Intra\_16x16” prediction type indicates that the macroblock is encoded by intra coding with different intra prediction modes which are defined in the same manner as in section 3.4.3 and TABLE 7. No motion vector data is transmitted for the intra modes.

### 5.3.3 Intra prediction mode (Intra\_pred\_mode)

As present, Intra\_pred\_mode indicates which intra prediction mode is used for a macroblock. Intra\_pred\_mode is present when Intra\_4x4 prediction type is indicated in the MB\_type. The code\_number is same as that described in the Intra\_pred\_mode entry of TABLE 7.

### 5.3.4 Reference Frame (Ref\_frame)

At present, Ref\_frame indicates the position of the reference frame in the reference frame buffer to be used for forward motion compensated for current macroblock. Ref\_frame is present only when the Ptype signals the use of multiple reference frames and only when the present MB\_type indicates Forward\_NxM or Bi-directional prediction type. Decoded I/P pictures are stored in the reference frame buffer in first-in-first-out manner and the most recently decoded I/P frame is always stored at position 0 in the reference frame buffer. The code\_number for Ref\_frame is described in TABLE 12.

**TABLE 11**

**MB\_Type and related data elements for B pictures**

| 0           | Direct          |                 |                        |          |       |       |
|-------------|-----------------|-----------------|------------------------|----------|-------|-------|
| Code_number | Prediction Type | Intra_pred_mode | Ref_frame <sup>1</sup> | Blk_size | MVDFW | MVDBW |
| 1           | Forward_16x16   |                 | X                      |          | X     |       |
| 2           | Backward_16x16  |                 |                        |          |       | X     |
| 3           | Bi-directional  |                 | X                      | X        | X     | X     |
| 4           | Forward_16x8    |                 | X                      |          | X     |       |
| 5           | Backward_16x8   |                 |                        |          |       | X     |
| 6           | Forward_8x16    |                 | X                      |          | X     |       |
| 7           | Backward_8x16   |                 |                        |          |       | X     |
| 8           | Forward_8x8     |                 | X                      |          | X     |       |
| 9           | Backward_8x8    |                 |                        |          |       | X     |
| 10          | Forward_8x4     |                 | X                      |          | X     |       |
| 11          | Backward_8x4    |                 |                        |          |       | X     |
| 12          | Forward_4x8     |                 | X                      |          | X     |       |
| 13          | Backward_4x8    |                 |                        |          |       | X     |
| 14          | Forward_4x4     |                 | X                      |          | X     |       |
| 15          | Backward_4x4    |                 |                        |          |       | X     |
| 16          | Intra_4x4       | X               |                        |          |       |       |

|     |                          |  |  |  |  |  |
|-----|--------------------------|--|--|--|--|--|
| 17  | Intra_16x16 <sup>2</sup> |  |  |  |  |  |
| ... | ...                      |  |  |  |  |  |

<sup>1</sup> Ref\_frame is a valid field only when the usage of multiple reference frames is present in Ptype, i.e. when Ptype=4..

<sup>2</sup> Intra\_16x16 indicates 16x16 based intra mode and should represent 24 different prediction modes as defined in section 3.4.2.1. For code\_number greater than 16 in TABLE 11, please see the code numbers from 9 and upwards in the field of inter MB\_type of TABLE 7 for reference.

**TABLE 12**

**Code\_number for ref\_frame**

| Code_number | Reference frame                               |
|-------------|-----------------------------------------------|
| 0           | The most recent previous frame (1 frame back) |
| 1           | 2 frames back                                 |
| 2           | 3 frames back                                 |
| ...         | ...                                           |

### 5.3.5 Block Size (Blk\_size)

If present, Blk\_size indicates which block size is used for forward and backward motion prediction in a macroblock as described in TABLE 13. Blk\_size is present only when Bi-directional prediction type is indicated in the MB\_type. There are two sets of Blk\_size data, one for forward motion vector data, and another for backward motion vector data.

**TABLE 13**

**Code\_number for Blk\_size**

| Code_number | Block Size    |
|-------------|---------------|
| 0           | 1 16x16 block |
| 1           | 2 16x8 blocks |
| 2           | 2 8x16 blocks |
| 3           | 4 8x8 blocks  |
| 4           | 8 8x4 blocks  |
| 5           | 8 4x8 blocks  |
| 6           | 16 4x4 blocks |

### 5.3.6 Motion vector data (MVDFW, MVDBW)

MVDFW is the motion vector data for the forward vector, if present. MVDBW is the motion vector data for the backward vector, if present. If so indicated by MB\_type or Blk\_size (bi-directional prediction type only), vector data for 1-16 blocks are transmitted. The order of transmitted motion vector data is the same as that indicated in FIGURE 2. For the code\_number of motion vector data, please refer to TABLE 7.

## 5.4 Decoder Process for motion vector

### 5.4.1 Differential motion vectors

Motion vectors for forward, backward, or bi-directionally predicted macroblock are differentially encoded. A prediction has to be added to the motion vector differences to get the motion vectors for the macroblock. The predictions are formed in way similar to that described in section 3.4.5. The only difference is that forward motion vectors are predicted only from forward motion vectors in surrounding

macroblocks, and backward motion vectors are predicted only from backward motion vectors in surrounding macroblocks.

If a neighboring macroblock does not have a motion vector of the same type, the candidate predictor for that macroblock is set to zero for that motion vector type.

#### **5.4.2 Motion vectors in direct mode**

In direct mode the same block structure as for the co-located macroblock in the temporally subsequent picture is used. For each of the sub-blocks the forward and backward motion vectors are computed as scaled versions of the corresponding vector components of the co-located macroblock in the temporally subsequent picture as described below.

If the multiple reference frame prediction is used, the forward reference frame for the direct mode is the same as the one used for the corresponding macroblock in the temporally subsequent reference picture. The forward and backward motion vectors for direct mode macroblocks are calculated as follows.

$$MV_F = (TR_B * MV) / TR_D$$

$$MV_B = (TR_B - TR_D) * MV / TR_D$$

Here,  $MV_F$  is the forward motion vector,  $MV_B$  is the backward motion vector, and  $MV$  represents the motion vector in the corresponding macroblock in the subsequent reference picture. Note that if the subsequent reference picture is an intra-coded frame or the reference macroblock is an intra-coded block, the motion vectors are set to zero.  $TR_D$  is the temporal distance between the temporally previous and next reference frame, and  $TR_B$  is the temporal distance between the current frame and previous reference frame.

It should be noted that when multiple reference frame prediction is in use, the actual reference frame (the same as for the co-located macroblock of the following picture) is used for the calculation of the temporal distances  $TR_D$  and  $TR_B$ .

## 6 SP-pictures

### 6.1 Introduction

SP-pictures make use of motion-compensated predictive coding to exploit temporal redundancy in the sequence similar to P-pictures while still allowing identical reconstruction of the frame even when different reference frames are being used. This picture type provides functionalities for bitstream switching, splicing, random access, VCR functionalities such as fast-forward and error resilience/recovery. SP-pictures make use of the existing coding blocks and the syntax elements for P-type frames while the difference is being forward transform coding and quantizing the predicted block during the reconstruction of blocks in SP-pictures.

### 6.2 Syntax changes

The only syntax change related to SP-frames is illustrated in FIGURE 21 as changes to syntax diagram to FIGURE 4. An additional quantization parameter is sent after the quantization parameter used for prediction error. This additional quantization parameter is used for (de)quantization of the predicted block. Otherwise, SP-frames have the identical syntax elements as P-frames. And all the syntax elements are interpreted identical to P-frames, i.e., 4x4 or 4x8 or 8x8 etc. blocks for motion compensation, reference frame number, etc.

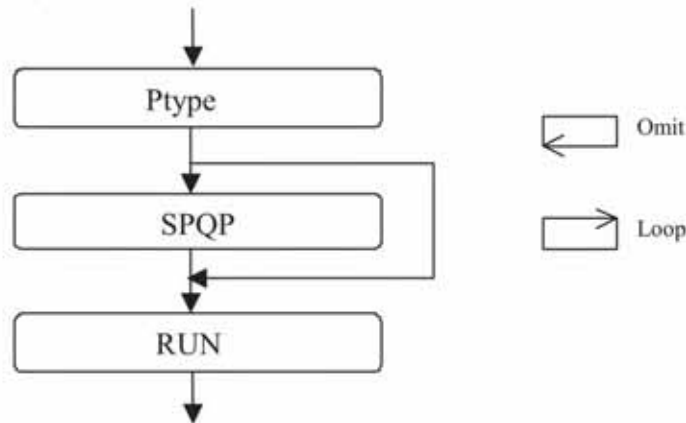


FIGURE 21

Syntax diagram for SP-frames

### 6.3 SP-frame decoding

SP-frames consist of intra and inter-type macroblocks. Intra macroblocks are encoded/decoded as in I and P-type pictures, therefore the following section refers to the decoding of the inter-type macroblocks in SP-frames.

Similar to P-type frame decoding, first the motion-compensated predicted block is formed by using received motion vector information, the reference frame number and the already decoded frames. Then, forward transform is applied to the predicted block. The resulting coefficients and the received level values are then used to calculate reconstructed image coefficients as follows:

$$L_{rec} = (K_{pred} \times A(QP_2) + L_{err} \times F(QP_1, QP_2) + f \times 2^{20}) / 2^{20}$$

where

$$F(QP_1, QP_2) = (A(QP_2) \times 2^{20} + A(QP_1) / 2) / A(QP_1)$$

and  $A(QP)$  is defined in Section 4.3.3.3. Here  $QP_1$  is signaled by PQP value and the  $QP_2$  is signaled by the additional QP parameter SPQP. Notice that when  $QP_1 = QP_2$ , then the calculation of  $L_{rec}$  reduces simply to the sum of the received level and the level found by quantizing the predicted block coefficient. The

coefficients,  $L_{rec}$ , are then dequantized using  $QP=QP_2$  and inverse transform is performed for these dequantized levels, as defined in sections 4.3.3 and 4.3.1, respectively. Finally, the result of the inverse transformation is shifted by 20 bit (with rounding).

While applying deblocking filter for macroblocks in SP-frames, all macroblocks are treated as Intra macroblocks as described in Section 4.4.

Since an additional 2x2 transform is performed for the DC coefficients of chrominance blocks in a macroblock after 4x4 transform of each, Section 4.3.2, the decoding of the chrominance component is performed in the similar manner as described above with the following difference: For chrominance macroblocks in SP-frames, an additional 2x2 transform is applied after 4x4 transform of the predicted chrominance blocks. And then the steps described above are repeated for the DC coefficients as well as for the AC coefficients. Note also that for chrominance blocks, the values of  $QP_1$  and  $QP_2$  are both changed according to the relation between QP values of luminance and chrominance specified in Section 4.3.3.3.

## 7 Hypothetical Reference Decoder

### 7.1 Purpose

The hypothetical reference decoder (HRD) is a mathematical model for the decoder, its input buffer, and the channel. The HRD is characterized by the channel's peak rate  $R$  (in bits per second), the buffer size  $B$  (in bits), and the initial decoder buffer fullness  $F$  (in bits). These parameters represent levels of resources (transmission capacity, buffer capacity, and delay) used to decode a bit stream.

A closely related object is the leaky bucket (LB), which is a mathematical constraint on a bit stream. A leaky bucket is characterized by the bucket's leak rate  $R_1$  (in bits per second), the bucket size  $B_1$  (in bits), and the initial bucket fullness  $B_1 - F_1$  (in bits). A given bit stream may be constrained by any number of leaky buckets  $(R_1, B_1, F_1), \dots, (R_N, B_N, F_N)$ ,  $N \geq 1$ . The LB parameters for a bit stream, which are encoded in the bit stream header, precisely describe the minimum levels of the resources  $R$ ,  $B$ , and  $F$  that are sufficient to guarantee that the bit stream can be decoded.

### 7.2 Operation of the HRD

The HRD input buffer has capacity  $B$  bits. Initially, the buffer begins empty. At time  $t_{start}$  it begins to receive bits, such that it receives  $S(t)$  bits through time  $t$ .  $S(t)$  can be regarded as the integral of the instantaneous bit rate through time  $t$ . The instant at which  $S(t)$  reaches the initial decoder buffer fullness  $F$  is identified as the decoding time  $t_0$  of the first picture in the bit stream. Decoding times  $t_1, t_2, t_3, \dots$ , for subsequent pictures (in bit stream order) are identified relative to  $t_0$ , per Section 7.3. At each decoding time  $t_i$ , the HRD instantaneously removes and decodes all  $d_i$  bits associated with picture  $i$ , thereby reducing the decoder buffer fullness from  $b_i$  bits to  $b_i - d_i$  bits. Between time  $t_i$  and  $t_{i+1}$ , the decoder buffer fullness increases from  $b_i - d_i$  bits to  $b_i - d_i + [S(t_{i+1}) - S(t_i)]$  bits. That is, for  $i \geq 0$ ,

$$b_0 = F$$

$$b_{i+1} = b_i - d_i + [S(t_{i+1}) - S(t_i)].$$

The channel connected to the HRD buffer has peak rate  $R$ . This means that unless the channel is idle (whereupon the instantaneous rate is zero), the channel delivers bits into the HRD buffer at instantaneous rate  $R$  bits per second.

### 7.3 Decoding Time of a Picture

The decoding time  $t_i$  of picture  $i$  is equal to its presentation time  $\tau_i$ , if there are no B pictures in the sequence. If there are B pictures in the sequence, then  $t_i = \tau_i - m_i$ , where  $m_i = 0$  if picture  $i$  is a B picture; otherwise  $m_i$  equals  $\tau_i - \tau_i'$ , where  $\tau_i'$  is the presentation time of the I or P picture that immediately precedes picture  $i$  (in presentation order). If there is no preceding I or P picture (i.e., if  $i = 0$ ), then  $m_i = m_0 = t_1 - t_0$ . The presentation time of a picture is determinable from its temporal reference and the frame rate.

### 7.4 Schedule of a Bit Stream

The sequence  $(t_0, d_0), (t_1, d_1), (t_2, d_2), \dots$  is called the schedule of a bit stream. The schedule of a bit stream is intrinsic to the bit stream, and completely characterizes the instantaneous coding rate of the bit stream over its lifetime. A bit stream may be pre-encoded, stored to a file, and later transmitted over channels with different peak bit rates to decoders with different buffer sizes. The schedule of the bit stream is invariant over such transmissions.

### 7.5 Containment in a Leaky Bucket

A leaky bucket with leak rate  $R_1$ , bucket size  $B_1$ , and initial bucket fullness  $B_1 - F_1$  is said to contain a bit stream with schedule  $(t_0, d_0), (t_1, d_1), (t_2, d_2), \dots$  if the bucket does not overflow under the following conditions. At time  $t_0$ ,  $d_0$  bits are inserted into the leaky bucket on top of the  $B_1 - F_1$  bits already in the bucket, and the bucket begins to drain at rate  $R_1$  bits per second. If the bucket empties, it remains empty until the next insertion. At time  $t_i$ ,  $i \geq 1$ ,  $d_i$  bits are inserted into the bucket, and the bucket continues to drain at rate  $R_1$  bits per second. In other words, for  $i \geq 0$ , the state of the bucket just prior to time  $t_i$  is

$$b_0 = B_1 - F_1$$

$$b_{i+1} = \max\{0, b_i + d_i - R_1(t_{i+1} - t_i)\}.$$

The leaky bucket does not overflow if  $b_i + d_i \leq B_1$  for all  $i \geq 0$ .

Equivalently, the leaky bucket contains the bit stream if the graph of the schedule of the bit stream lies between two parallel lines with slope  $R_1$ , separated vertically by  $B_1$  bits, possibly sheared horizontally, such that the upper line begins at  $F_1$  at time  $t_0$ , as illustrated in the figure below. Note from the figure that the same bit stream is containable in more than one leaky bucket. Indeed, a bit stream is containable in an infinite number of leaky buckets.

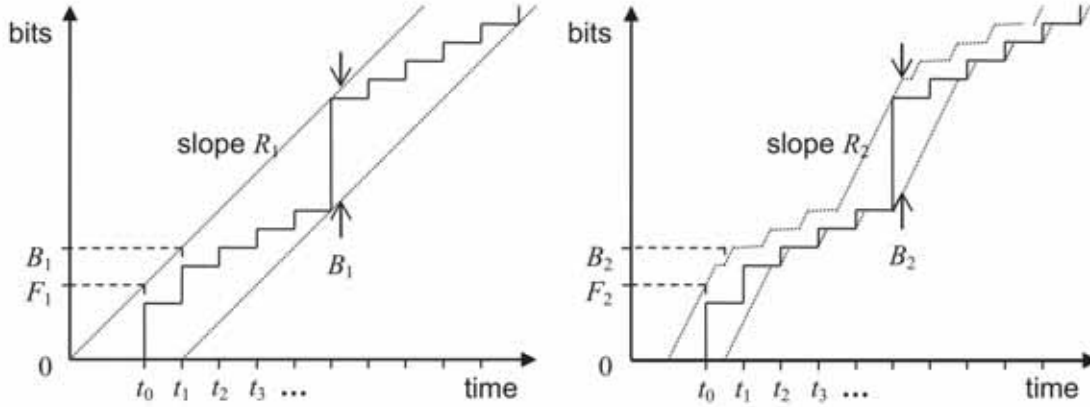


FIGURE 22

#### Illustration of the leaky bucket concept.

If a bit stream is contained in a leaky bucket with parameters  $(R_1, B_1, F_1)$ , then when it is communicated over a channel with peak rate  $R_1$  to a hypothetical reference decoder with parameters  $R=R_1$ ,  $B=B_1$ , and  $F=F_1$ , then the HRD buffer does not overflow or underflow.

### 7.6 Bit Stream Syntax

The header of each bit stream shall specify the parameters of a set of  $N \geq 1$  leaky buckets,  $(R_1, B_1, F_1), \dots, (R_N, B_N, F_N)$ , each of which contains the bit stream. In the current Test Model, these parameters are specified in the first  $1+3N$  32-bit integers of the Interim File Format, in network (big-endian) byte order:

$$N, R_1, B_1, F_1, \dots, R_N, B_N, F_N.$$

The  $R_n$  shall be in strictly increasing order, and both  $B_n$  and  $F_n$  shall be in strictly decreasing order.

These parameters shall not exceed the capability limits for particular profiles and levels, which are yet to be defined.

### 7.7 Minimum Buffer Size and Minimum Peak Rate

If a bit stream is contained in a set of leaky buckets with parameters  $(R_1, B_1, F_1), \dots, (R_N, B_N, F_N)$ , then when it is communicated over a channel with peak rate  $R$ , it is decodable (i.e., the HRD buffer does not overflow or underflow) provided  $B \geq B_{min}(R)$  and  $F \geq F_{min}(R)$ , where for  $R_n \leq R \leq R_{n+1}$ ,

$$B_{min}(R) = \alpha B_n + (1 - \alpha) B_{n+1}$$

$$F_{min}(R) = \alpha F_n + (1 - \alpha) F_{n+1}$$

$$\alpha = (R_{n+1} - R) / (R_{n+1} - R_n).$$

For  $R \leq R_1$ ,

$$B_{min}(R) = B_1 + (R_1 - R)T$$

$$F_{min}(R) = F_1,$$

where  $T = t_{L-1} - t_0$  is the duration of the bit stream (i.e., the difference between the decoding times for the first and last pictures in the bit stream). And for  $R \geq R_N$ ,

$$B_{min}(R) = B_N$$

$$F_{min}(R) = F_N.$$

Thus, the leaky bucket parameters can be linearly interpolated and extrapolated.

Alternatively, when the bit stream is communicated to a decoder with buffer size  $B$ , it is decodable provided  $R \geq R_{min}(B)$  and  $F \geq F_{min}(B)$ , where for  $B_n \geq B \geq B_{n+1}$ ,

$$R_{min}(B) = \alpha R_n + (1 - \alpha) R_{n+1}$$

$$F_{min}(B) = \alpha F_n + (1 - \alpha) F_{n+1}$$

$$\alpha = (B - B_{n+1}) / (B_n - B_{n+1}).$$

For  $B \geq B_1$ ,

$$R_{min}(B) = R_1 - (B - B_1) / T$$

$$F_{min}(B) = F_1.$$

For  $B \leq B_N$ , the stream may not be decodable.

In summary, the bit stream is guaranteed to be decodable in the sense that the HRD buffer does not overflow or underflow, provided that the point  $(R, B)$  lies on or above the lower convex hull of the set of points  $(0, B_1 + R_1 T)$ ,  $(R_1, B_1)$ , ...,  $(R_N, B_N)$ , as illustrated in the figure below. The minimum start-up delay necessary to maintain this guarantee is  $F_{min}(R) / R$ .

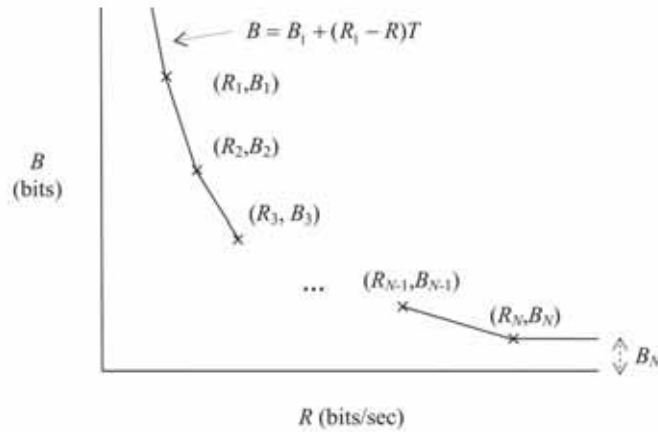


FIGURE 23

### Illustration of the leaky bucket concept.

A compliant decoder with buffer size  $B$  and initial decoder buffer fullness  $F$  that is served by a channel with peak rate  $R$  shall perform the tests  $B \geq B_{min}(R)$  and  $F \geq F_{min}(R)$ , as defined above, for any compliant bit stream with LB parameters  $(R_1, B_1, F_1), \dots, (R_N, B_N, F_N)$ , and shall decode the bit stream provided that  $B \geq B_{min}(R)$  and  $F \geq F_{min}(R)$ .

## 7.8 Encoder Considerations (informative)

The encoder can create a bit stream that is contained by some given  $N$  leaky buckets, or it can simply compute  $N$  sets of leaky bucket parameters after the bit stream is generated, or a combination of these. In the former, the encoder enforces the  $N$  leaky bucket constraints during rate control. Conventional rate control algorithms enforce only a single leaky bucket constraint. A rate control algorithm that simultaneously enforces  $N$  leaky bucket constraints can be obtained by running a conventional rate control algorithm for each of the  $N$  leaky bucket constraints, and using as the current quantization parameter (QP) the maximum of the QPs recommended by the  $N$  rate control algorithms.

Additional sets of leaky bucket parameters can always be computed after the fact (whether rate controlled or not), from the bit stream schedule for any given  $R_n$ , from the iteration specified in Section 7.5.

## Appendix I Non-normative Encoder Recommendation

### I.1 Motion Estimation and Mode Decision

#### I.1.1 Low-complexity mode

##### I.1.1.1 Finding optimum prediction mode

Both for intra prediction and motion compensated prediction, a similar loop as indicated in **Error! Reference source not found.** is run through. The different elements will be described below.

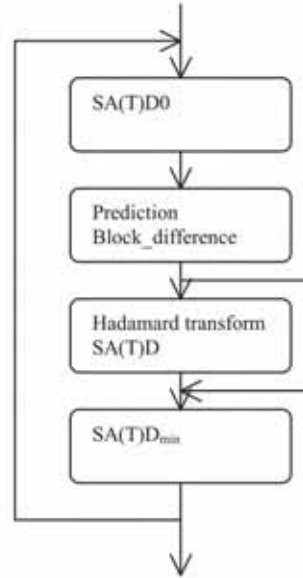


FIGURE 24

#### Loop for prediction mode decision

##### I.1.1.1.1 SA(T)D0

The SA(T)D to be minimised is given a 'bias' value SA(T)D0 initially in order to favour prediction modes that need few bits to be signalled. This bias is basically a parameter representing bit usage times  $QP_0(QP)$

Intra mode decision:  $SA(T)D0 = QP_0(QP) \times \text{Order\_of\_prediction\_mode}$  (see above)

Motion vector search:  $SA(T)D0 = QP_0(QP) \times (\text{Bits\_to\_code\_vector} + 2 \times \text{code\_number\_of\_ref\_frame})$

In addition there are two special cases:

- For motion prediction of a 16x16 block with 0 vector components,  $16 \times QP_0(QP)$  is subtracted from SA(T)D to favour the skip mode.
- For the whole intra 4x4 macroblock,  $24 \times QP_0(QP)$  is added to the SA(T)D before comparison with the best SA(T)D for inter prediction. This is an empirical value to prevent using too many intra blocks.

For flat regions having zero motion, B pictures basically fail to make effective use of zero motion and instead are penalized in performance by selecting 16x16 intra mode. Therefore, in order to prevent assigning 16x16 intra mode to a region with little details and zero motion, SA(T)D of direct mode is subtracted by  $16 \times QP_0(QP)$  to bias the decision toward selecting the direct mode.

The calculation of SA(T)D0 at each mode is as follows.

- Forward prediction mode :  
 $SA(T)D0 = QP_0(QP) \times (2 \times \text{code\_number\_of\_Ref\_frame} + \text{Bits\_to\_code\_MVDFW})$
- Backward prediction mode :  
 $SA(T)D0 = QP_0(QP) \times \text{Bits\_to\_code\_MVDBW}$

- Bi-directional prediction mode :  
 $SA(T)D0 = QP_0(QP) \times (2 \times \text{code\_number\_of\_Ref\_frame} + \text{Bits\_to\_code\_forward\_Blk\_size} + \text{Bits\_to\_code\_backward\_Blk\_size} + \text{Bits\_to\_code\_MVDFW} + \text{Bits\_to\_code\_MVDBW})$
- Direct prediction mode :  
 $SA(T)D0 = -16 \times QP_0(QP)$
- Intra 4x4 mode :  
 $SA(T)D0 = 24 \times QP_0(QP)$
- Intra 16x16 mode :  
 $SA(T)D0 = 0$

#### I.1.1.1.2 Block\_difference

For the whole block the difference between the original and prediction is produced

$$\text{Diff}(i,j) = \text{Original}(i,j) - \text{Prediction}(i,j)$$

#### I.1.1.1.3 Hadamard transform

For integer pixel search (see below) we use SAD based on  $\text{Diff}(i,j)$  for decision. Hence no Hadamard is done and we use SAD instead of SATD.

$$SAD = \sum_{i,j} |\text{Diff}(i,j)|$$

However, since we will do a transform of  $\text{Diff}(i,j)$  before transmission, we will do a better optimisation if a transform is done before producing SAD. Therefore a two dimensional transform is performed in the decision loop for selecting intra modes and for fractional pixel search (see below). To simplify implementation, the Hadamard transform is chosen in this mode decision loop. The relation between pixels and basis vectors (BV) in a 4 point Hadamard transform is illustrated below (not normalized):

$$\begin{array}{c|cccc} \text{Pixels} & \rightarrow & & & \\ \hline \text{B} & 1 & 1 & 1 & 1 \\ \text{V} & 1 & 1 & -1 & -1 \\ \downarrow & 1 & -1 & -1 & 1 \\ & 1 & -1 & 1 & -1 \end{array}$$

This transformation is performed horizontally and vertically and result in  $\text{DiffT}(i,j)$ . Finally SATD for the block and for the present prediction mode is produced.

$$SATD = (\sum_{i,j} |\text{DiffT}(i,j)|) / 2$$

#### I.1.1.1.4 Mode decision

Choose the prediction mode that results in the minimum value  $SA(T)D_{\min} = \min(SA(T)D + SA(T)D0)$ .

### I.1.1.2 Encoding on macroblock level

#### I.1.1.2.1 Intra coding

When starting to code a macroblock, intra mode is checked first. For each 4x4 block, full coding indicated in **Error! Reference source not found.** is performed. At the end of this loop the complete macroblock is intra coded and a  $SATD_{\text{intra}}$  is calculated.

#### I.1.1.2.2 Table for intra prediction modes to be used at the encoder side

**Error! Reference source not found.** gives the table of intra prediction modes according to probability of each mode to be used on the decoder side. On the encoder side we need a sort of inverse table. Prediction modes for A and B are known as in

TABLE 3. For the encoder we have found a Mode that we want to signal with an ordering number in the bitstream (whereas on the decoder we receive the order in the bitstream and want to convert this to a mode). **Error! Reference source not found.** is therefore the relevant table for the encoder. Example: Prediction mode for A and B is 2. The string in **Error! Reference source not found.** is 2 1 0 3 4 5. This indicates that prediction mode 0 has order 2 (third most probable). Prediction mode 1 is second most probable and prediction mode 2 has order 0 (most probable) etc. As in

TABLE 3 '-' indicates that this instance can not occur because A or B or both are outside the picture.

TABLE 14

Prediction ordering to be used in the bitstream as a function of prediction mode (see text).

| B\A     | outside | 0      | 1      | 2      | 3      | 4      | 5      |
|---------|---------|--------|--------|--------|--------|--------|--------|
| outside | 0-----  | 021--- | 102--- | 120--- | 012--- | 012--- | 012--- |
| 0       | 0---12  | 025314 | 104325 | 240135 | 143025 | 035214 | 045213 |
| 1       | 0---12  | 014325 | 102435 | 130245 | 032145 | 024315 | 015324 |
| 2       | 0---12  | 012345 | 102345 | 210345 | 132045 | 032415 | 013245 |
| 3       | 0---12  | 135024 | 214035 | 320154 | 143025 | 145203 | 145032 |
| 4       | 1---02  | 145203 | 125403 | 250314 | 245103 | 145203 | 145302 |
| 5       | 1---20  | 245310 | 015432 | 120534 | 245130 | 245301 | 135420 |

#### I.1.1.2.3 Inter mode selection

Next motion vector search is performed for motion compensated prediction. A search is made for all 7 possible block structures for prediction as well as from the 5 past decoded pictures. This result in 35 combinations of block sizes and reference frames. For B-frames, the motion search is also conducted for the temporal following reference picture to obtain backward motion vectors.

#### I.1.1.2.4 Integer pixel search

The search positions are organised in a 'spiral' structure around the predicted vector (see vector prediction). The numbering from 0 and upwards for the first positions are listed below:

```

. . . . .
. 15 9 11 13 16
. 17 3 1 4 18
. 19 5 0 6 20
. 21 7 2 8 22
. 23 10 12 14 24

```

A parameter MC\_range is used as input for each sequence. To speed up the search process, the search range is further reduced:

- Search range is reduced to: Range = MC\_range/2 for all block sizes except 16x16 in prediction from the most recent decoded picture.
- The range is further reduced to: Range = Range/2 for search relative to all older pictures.

After Range has been found, the centre for the spiral search is adjusted so that the (0,0) vector position is within the search range. This is done by clipping the horizontal and vertical positions of the search centre to  $\pm$ Range.

#### I.1.1.2.5 Fractional pixel search

Fractional pixel search is performed in two steps. This is illustrated below where capital letters represent integer positions, numbers represent  $\frac{1}{2}$  pixel positions and lower case letters represent  $\frac{1}{4}$  pixel positions.

```

  A      B      C
    1    2    3
D    4    E    5    F

```

|   |   |   |   |   |
|---|---|---|---|---|
|   | a | b | c |   |
| 6 | d | 7 | e | 8 |
|   | f | g | h |   |
| G |   | H |   | I |

Assume that the integer search points to position E. Then ½ pixel positions 1,2,3,4,5,6,7,8 are searched. Assume that 7 is the best position. Then the ¼ pixel positions a,b,c,d,e,f,g,h are searched. (Notice that by this procedure a position with ‘more low pass filtering’ – see 4.2.1 – is automatically checked). If motion compensation with 1/8 pixel accuracy is used, an additional sub-pixel refinement step is performed in the described way. After fractional pixel search has been performed for the complete macroblock, the SATD for the whole macroblock is computed: SATD<sub>inter</sub>.

#### I.1.1.2.6 Decision between intra and inter

If SATD<sub>intra</sub> < SATD<sub>inter</sub> intra coding is used. Otherwise inter coding is used.

### I.1.2 High-complexity mode

#### I.1.2.1 Motion Estimation

For each block or macroblock the motion vector is determined by full search on integer-pixel positions followed by sub-pixel refinement.

##### I.1.2.1.1 Integer-pixel search

As in low-complexity mode, the search positions are organized in a spiral structure around a prediction vector. The full search range given by MC\_range is used for all INTER-modes and reference frames. To speed up the search process, the prediction vector of the 16x16 block is used as center of the spiral search for all INTER-modes. Thus the SAD values for 4x4 blocks can be pre-calculated for all motion vectors of the search range and then used for fast SAD calculation of all larger blocks. The search range is not forced to contain the (0,0)-vector.

##### I.1.2.1.2 Fractional pixel search

The fractional pixel search is performed as in the low-complexity case.

##### I.1.2.1.3 Finding the best motion vector

The integer-pixel motion search as well as the sub-pixel refinement returns the motion vector that minimizes

$$J(\mathbf{m}, \lambda_{MOTION}) = SAD(s, c(\mathbf{m})) + \lambda_{MOTION} \cdot R(\mathbf{m} - \mathbf{p})$$

with  $\mathbf{m} = (m_x, m_y)^T$  being the motion vector,  $\mathbf{p} = (p_x, p_y)^T$  being the prediction for the motion vector, and  $\lambda_{MOTION}$  being the Lagrange multiplier. The rate term  $R(\mathbf{m} - \mathbf{p})$  represents the motion information only and is computed by a table-lookup. The rate is estimated by using the universal variable length code (UVLC) table, even if CABAC is used as entropy coding method. For integer-pixel search, SAD is used as distortion measure. It is computed as

$$SAD(s, c(\mathbf{m})) = \sum_{x=1, y=1}^{B, B} |s[x, y] - c[x - m_x, y - m_y]|, \quad B = 16, 8 \text{ or } 4.$$

with  $s$  being the original video signal and  $c$  being the coded video signal. In the sub-pixel refinement search, the distortion measure SATD is calculated after a Hadamard transform (see section **Error!**

**Reference source not found.**). The Lagrangian multiplier  $\lambda_{MOTION}$  is given by

$$\lambda_{MOTION, P} = \sqrt{5} \cdot e^{QP/20} \cdot \sqrt{\frac{QP+5}{34-QP}}$$

for P-frames and

$$\lambda_{MOTION,B} = 2 \cdot \sqrt{5} \cdot e^{QP/20} \cdot \sqrt{\frac{QP+5}{34-QP}}$$

for B-frames, where  $QP$  is the macroblock quantization parameter.

#### I.1.2.1.4 Finding the best reference frame

The determination of the reference frame  $REF$  and the associated motion vectors for the  $NxM$  inter modes in P-frames and the  $FWD NxM$  modes in B-frames is done after motion estimation by minimizing

$$J(REF | \lambda_{MOTION}) = SATD(s, c(REF, \mathbf{m}(REF))) + \lambda_{MOTION} \cdot (R(\mathbf{m}(REF)) - \mathbf{p}(REF)) + R(REF).$$

The rate term  $R(REF)$  represents the number of bits associated with choosing  $REF$  and is computed by table-lookup using UVLC. The reference frame and block sizes for the bi-directional mode are chosen as combination of the “best” forward and backward mode.

#### I.1.2.2 Mode decision

##### I.1.2.2.1 Macroblock mode decision

The macroblock mode decision is done by minimizing the Lagrangian functional

$$J(s, c, MODE | QP, \lambda_{MODE}) = SSD(s, c, MODE | QP) + \lambda_{MODE} \cdot R(s, c, MODE | QP)$$

where  $QP$  is the macroblock quantizer,  $\lambda_{MODE}$  is the Lagrange multiplier for mode decision, and  $MODE$  indicates a mode chosen from the set of potential prediction modes:

I-frame:  $MODE \in \{INTRA4x4, INTRA16x16\},$

P-frame:  $MODE \in \left\{ INTRA4x4, INTRA16x16, SKIP, \right. \\ \left. 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 \right\},$

B-frame:  $MODE \in \left\{ INTRA4x4, INTRA16x16, BIDIRECT, DIRECT, \right. \\ \left. FWD16x16, FWD16x8, FWD8x16, FWD8x8, FWD8x4, \right. \\ \left. FWD4x8, FWD4x4, BAK16x16, BAK16x8, BAK8x16, \right. \\ \left. BAK8x8, BAK8x4, BAK4x8, BAK4x4 \right\}.$

Note that the  $SKIP$  mode refers to the 16x16 mode where no motion and residual information is encoded.  $SSD$  is the sum of the squared differences between the original block  $s$  and its reconstruction  $c$  given as

$$SSD(s, c, MODE | QP) = \sum_{x=1, y=1}^{16,16} (s_Y[x, y] - c_Y[x, y, MODE | QP])^2 \\ + \sum_{x=1, y=1}^{8,8} (s_U[x, y] - c_U[x, y, MODE | QP])^2 + \sum_{x=1, y=1}^{8,8} (s_V[x, y] - c_V[x, y, MODE | QP])^2,$$

and  $R(s, c, MODE | QP)$  is the number of bits associated with choosing  $MODE$  and  $QP$  including the bits for the macroblock header, the motion, and all DCT blocks.  $c_Y[x, y, MODE | QP]$  and  $s_Y[x, y]$  represent the reconstructed and original luminance values;  $c_U, c_V$  and  $s_U, s_V$  the corresponding chrominance values.

The Lagrangian multiplier  $\lambda_{MODE}$  is given by

$$\lambda_{MODE,P} = 5 \cdot e^{QP/10} \cdot \left( \frac{QP+5}{34-QP} \right)$$

for I- and P-frames and

$$\lambda_{MODE,B} = 20 \cdot e^{QP/10} \cdot \left( \frac{QP+5}{34-QP} \right)$$

for B-frames, where  $QP$  is the macroblock quantization parameter.

#### I.1.2.2.2 INTER 16x16 mode decision

The *INTER16x16* mode decision is performed by choosing the *INTER16x16* mode which results in the minimum SATD value.

#### I.1.2.2.3 INTER 4x4 mode decision

For the *INTRA4x4* prediction, the mode decision for each 4x4 block is performed similar to the macroblock mode decision by minimizing

$$J(s, c, IMODE | QP, \lambda_{MODE}) = SSD(s, c, IMODE | QP) + \lambda_{MODE} \cdot R(s, c, IMODE | QP)$$

where  $QP$  is the macroblock quantizer,  $\lambda_{MODE}$  is the Lagrange multiplier for mode decision, and  $IMODE$  indicates an intra prediction mode:

$$IMODE \in \{DC, HOR, VERT, DIAG, DIAG\_RL, DIAG\_LR\}.$$

$SSD$  is the sum of the squared differences between the original 4x4 block luminance signal  $s$  and its reconstruction  $c$ , and  $R(s, c, IMODE | QP)$  represents the number of bits associated with choosing  $IMODE$ . It includes the bits for the intra prediction mode and the DCT-coefficients for the 4x4 luminance block. The rate term is computed using the UVLC entropy coding, even if CABAC is used for entropy coding.

#### I.1.2.3 Algorithm for motion estimation and mode decision

The procedure to encode one macroblock  $s$  in a I-, P- or B-frame in the high-complexity mode is summarized as follows.

1. Given the last decoded frames,  $\lambda_{MODE}$ ,  $\lambda_{MOTION}$ , and the macroblock quantizer  $QP$
2. Choose intra prediction modes for the *INTRA 4x4* macroblock mode by minimizing  $J(s, c, IMODE | QP, \lambda_{MODE}) = SSD(s, c, IMODE | QP) + \lambda_{MODE} \cdot R(s, c, IMODE | QP)$  with  $IMODE \in \{DC, HOR, VERT, DIAG, DIAG\_RL, DIAG\_LR\}$ .
3. Determine the best *INTRA16x16* prediction mode by choosing the mode that results in the minimum SATD.
4. Perform motion estimation and reference frame selection by minimizing  $J(REF, \mathbf{m}(REF) | \lambda_{MOTION}) = SA(T)D(s, c(REF, \mathbf{m}(REF))) + \lambda_{MOTION} \cdot (R(\mathbf{m}(REF) - \mathbf{p}(REF)) + R(REF))$  for each reference frame and motion vector of a possible macroblock mode.
5. Choose the macroblock prediction mode by minimizing  $J(s, c, MODE | QP, \lambda_{MODE}) = SSD(s, c, MODE | QP) + \lambda_{MODE} \cdot R(s, c, MODE | QP)$ , given  $QP$  and  $\lambda_{MODE}$  when varying  $MODE$ .  $MODE$  indicates a mode out of the set of potential macroblock modes:

I-frame:  $MODE \in \{INTRA4x4, INTRA16x16\},$

P-frame:  $MODE \in \left\{ INTRA4x4, INTRA16x16, SKIP, \right. \\ \left. 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 \right\},$

B-frame:  $MODE \in \left\{ INTRA4x4, INTRA16x16, BIDIRECT, DIRECT, \right. \\ \left. FWD16x16, FWD16x8, FWD8x16, FWD8x8, FWD8x4, \right. \\ \left. FWD4x8, FWD4x4, BAK16x16, BAK16x8, BAK8x16, \right. \\ \left. BAK8x8, BAK8x4, BAK4x8, BAK4x4 \right\}.$

The computation of  $J(s, c, SKIP | QP, \lambda_{MODE})$  and  $J(s, c, DIRECT | QP, \lambda_{MODE})$  is simple. The costs for the other macroblock modes are computed using the intra prediction modes or motion vectors and reference frames, which have been estimated in steps 2- 4.

## I.2 Quantization

For each transform coefficient K the quantized LEVEL is produced in the following way:

$LEVEL = (K \times A(QP) + f \times 2^{20}) / 2^{20}$     |f| is 1/3 for intra and 1/6 for inter blocks and f has the same sign as K.

## I.3 Elimination of single coefficients in inter macroblocks

### I.3.1 Luma

With the small 4x4 blocks, it may happen that for instance a macroblock has only one nonzero coefficient with  $|Level| = 1$ . This will probably be a very expensive coefficient and it could have been better to set it to zero. For that reason a procedure to check single coefficients have been implemented for inter luma blocks. During the quantization process, a parameter **Single\_ctr** is accumulated depending on Run and Level according to the following rule:

- If  $Level = 0$  or ( $|Level| = 1$  and  $Run > 5$ ) nothing is added to **Single\_ctr**.
- If  $|Level| > 1$ , 9 is added to **Single\_ctr**.
- If  $|Level| = 1$  and  $Run < 6$ , a value  $T(Run)$  is added to **Single\_ctr**, where  $T(0:5) = (3, 2, 2, 1, 1, 1)$
- If the accumulated **Single\_ctr** for a 8x8 block is less than 4, all coefficients of that luma block are set to zero. Similarly, if the accumulated **Single\_ctr** for the whole macroblock is less than 6, all coefficients of that luma macroblock are set to zero.

### I.3.2 Chroma

A similar method to the one for luma is used. **Single\_ctr** is calculated similarly for each chroma component, but for AC coefficients only and for the whole macroblock.

If the accumulated **Single\_ctr** for each chroma component of a macroblock is less than 7, all the AC chroma coefficients of that component for the whole macroblock are set to zero.

## I.4 Encoding with Anticipation of Slice Losses

Low delay video transmission may lead to losses of slices. The decoder may then stop decoding until the next I picture or P picture may conduct a concealment, for example as explained in Appendix IV, and continue decoding. In the latter case, spatio-temporal error propagation occurs if the concealed picture content is referenced for motion compensation. There are various means to stop spatio-temporal error propagation including the usage of multiple reference pictures and Intra coding of macroblocks. For the latter case, a Lagrangian mode selection algorithm is suggested as follows.

Since transmission errors occur randomly, the decoding result is also a random process. Therefore, the average decoder distortion is estimated to control the encoder for a specified probability of packet losses  $p$ . The average decoding result is obtained by running  $N$  complete decoders at the encoder in parallel. The statistical process of loosing a slice is assumed to be independent for each of the  $N$  decoders. The slice loss process for each decoder is also assumed to be i.i.d. and a certain slice loss probability  $p$  is assumed to be known at the encoder. Obviously for large  $N$  the decoder gets a very good estimate of the average decoder distortion. However, with increasing  $N$  a linear increase of storage and decoder complexity in the encoder is incurred. Therefore, this method might not be practical in real-time encoding processes and complexity and memory efficient algorithms are currently under investigation.

To encode a macroblock in a P picture, the set of possible macroblock types is given as

$$S_{MB} = \{ SKIP, INTER\_16x16, INTER\_16x8, INTER\_8x16, \\ INTER\_8x8, INTER\_8x4, INTER\_4x8, INTER\_4x4, \}$$

INTRA\_4x4, INTRA\_16x16}

For each macroblock the coding mode  $m'$  is selected according to

$$m' = \min_{m \in S_{MB}} \{D_m + \lambda / R_m\}$$

with  $D_m$  being the distortion in the current macroblock when selecting macroblock mode  $m$  and  $R_m$  being the corresponding rate, i.e. the number of bits. For the COPY\_MB and all INTER\_MxN types, the distortion  $D_m$  is computed as

$$D_m = \frac{1}{N} \sum_{n=1}^N \sum_i (f_i - \hat{f}_{i,n,m}(p))^2$$

with  $f_i$  being the original pixel value at position  $i$  within the macroblock and  $\hat{f}_{i,n,m}$  being the reconstructed pixel value at position  $i$  for coding macroblock mode  $m$  in the simulated decoder  $n$ . The distortion for the INTRA macroblocks remains unchanged. Since the various reconstructed decoders also contain transmission errors, the Lagrangian cost function for the COPY\_MB and all INTER\_MxN types increases making INTRA\_NxN types more popular.

The  $\lambda$  parameter for mode decision depends on the quantization parameter  $q$  as follows

$$\lambda = (1 - p) \frac{q + 5}{34 - q} e^{0.1q}.$$

## Appendix II Network Adaptation Layer for IP networks

This section covers the Network Adaptation Layer for non-managed, best effort IP networks using RTP [RFC1889] as the transport. The section will likely end up in the form of a standard's track RFC covering an RTP packetization for H.26L.

The NAL takes the information of the Interim File Format as discussed in section **Error! Reference source not found.** and converts it into packets that can be conveyed directly over RTP. It is designed to be able to take advantage of more than one virtual transport stream (either within one RTP stream by unequal packet content protection currently discussed in the IETF and as Annex I of H.323, or by using several RTP streams with network or application layer unequal error protection).

In doing so, it has to

- arrange partitions in an intelligent way into packets
- eventually split/recombine partitions to match MTU size constraints
- avoid the redundancy of (mandatory) RTP header information and information in the video stream
- define receiver/decoder reactions to packet losses. Note: the IETF tends more and more to do this in a normative way, whereas in the ITU and in MPEG this is typically left to implementers. Issue has to be discussed one day. Current document provides information without making any assumptions about that.

### II.1 Assumptions

Any packetization scheme has to make some assumptions on typical network conditions and constraints. The following set of assumptions have been used in earlier Q.15 research on packetization and are deemed to be still valid:

- MTU size: around 1500 bytes per packet for anything but dial-up links, 500 bytes for dial-up links.
- Packet loss characteristic: non-bursty (due to drop-tail router implementations, and assuming reasonable pacing algorithms (e.g. no bursting occurs at the sender).
- Packet loss rate: up to 20%

### II.2 Combining of Partitions according to Priorities

In order to allow unequal protection of more important bits of the bitstream, exactly two packets per slice are generated (see Q15-J-53 for a detailed discussion). Slices should be used to ensure that both packets meet the MTU size constraints to avoid network splitting/recombining processes.

The 'First' packet contains the following partitions:

- TYPE\_HEADER
- TYPE\_MBHEADER
- TYPE\_MVD
- TYPE\_EOS

The 'Second' packet is assembled using the rest of the partitions

- TYPE\_CBP      Coded Block Pattern
- TYPE\_2x2DC    2x2 DC Coefficients
- TYPE\_COEFF\_Y    Luminance AC Coefficients
- TYPE\_COEFF\_C    Chrominance AC Coefficients

This configuration allows decoding the first packet independently from the second (although not vice versa). As the first packet is more important both because motion information is important for struction

[for what?] and because the 'First' packet is necessary to decode the 'Second', UEP should be used to protect the 'First' packet s higher.

### II.3 Packet Structure

Each packet consists of a fixed 32 bit header a series of Part-of-Partition structures (POPs).

The packet header contains information

|             |                                                                           |
|-------------|---------------------------------------------------------------------------|
| Bit 31      | 1 == This packet contains a picture header                                |
| Bit 30      | 1 == This packet contains a slice header                                  |
| Bits 25..29 | Reserved                                                                  |
| Bits 10-24  | StartMB. It is assumed that no picture has more than $2^{14}$ Macroblocks |
| Bits 0..9   | SliceID. It is assumed that a picture does not have more than 1024 slices |

Note: The PictureID (TR) can be easily reconstructed from the RTP Timestamp and is therefore not coded again.

Note: The current software reconstructs QP and Format out of the VLC coded Picture/Slice header symbols. This is architecturally not nice and should be changed, probably by deleting these two values from the interim File Format.

Note: This header is likely to change once bigger picture formats etc. come into play.

Each Part-of-Partition structure contains a header of 16 bits whose format is as follows

|             |                                                                                                    |
|-------------|----------------------------------------------------------------------------------------------------|
| Bits 15..12 | Data Type                                                                                          |
| Bits 11..0  | Length of VLC-coded POP payload (in bits, starting byte-aligned, 0 indicates 4096 bits of payload) |

The reasoning behind the introduction of POP packets lies in avoiding large fixed length headers for (typically) small partitions. See Q15-J-53.

### II.4 Packetization Process

The packetization process converts the Interim File Format (or, in a real world system, data partitioned symbols arriving through a software interface) into packets. The following RTP header fields are used (see RFC1889 for exact semantics):

- Timestamp: is calculated according to the rules of RFC1889 and RFC2429 based on a 90 KHz timestamp.
- Marker Bit: set for the very last packet of a picture ('Second' packet of the last Slice), otherwise cleared.
- Sequence Number is increased by one for every generated packet, and starts with 0 for easier debugging (this in contrast to RFC1889, where a random initialization is mandatory for security purposes).
- Version (V): 2
- Padding (P): 0
- Extension (X): 0
- Csourcecount (CC): 0
- Payload Type (PT): 0 (This in contrast to RFC1889 where 0 is forbidden)

The RTP header is followed by the payload, which follows the packet structure of section II.3.

The RTP packet file, used as the input to packet loss simulators and similar tools (note that this format is identical to the ones used for IP-related testing during the H.263++ project, so that the loss simulators, error paytterns etc, can be re-used):

|        |                                               |
|--------|-----------------------------------------------|
| Int32  | size of the following packet in bytes         |
| []byte | packet content, starting with the RTP header. |

## **II.5 De-packetization**

The De-packetization process reconstructs a file in the Interim File Format from an RTP packet file (that is possible subject to packet losses). This task is straightforward and reverse to the packetization process. (Note that the QP and Format fields currently have to be reconstructed using the VLC-coded symbols in the TYPE\_HEADER partition. This bug in the Interim File Format spec should be fixed some time).

## **II.6 Repair and Error Concealment**

In order to take advantage of potential UEP for the 'First' packet and the ability of the decoder to reconstruct data where CBP/coefficient information was lost, a very simple error concealment strategy is used. This strategy repairs the bitstream by replacing a lost CBP partition with CBPs that indicate no coded coefficients. Unfortunately, the CBP codewords for Intra and Inter blocks are different, so that such a repair cannot be done context-insensitive. Instead of (partly) VLC-decoding the CBP partition in the NAL module in order to insert the correct type of CBP symbol in the (lost) partition, the decoder itself can be changed to report the appropriate CBP symbols saying "No coefficients" whenever the symbol fetch for a CBP symbol returns with the indication of a lost/empty partition.

## Appendix III Interim File Format

*[Editor: this text needs work by the proponent until the next meeting Jan 2002. If no improvements are made, the text will be dropped.]*

### III.1 General

A file is self-contained.

A file consists of clumps, which are object-like entities and similar to boxes of ISO/IEC 14496-1:2001 (ISO media file format). Name 'clump' was chosen to differentiate them from MPEG-4's boxes, chunks, and objects as well as from QuickTime's atoms. Note: In fact, the definition of a clump is the same as the definition of a box except for the extended type mechanism of a box. Thus, in order to emphasize the common definition of a clump and a box, it might be appropriate to refer to a box instead of a clump in this document. However, as this was not agreed in the Pattaya meeting, this change was not done yet.

A clump may contain other clumps. A clump may have member attributes. If a clump contains attributes and other clumps, clumps shall follow the attribute values.

The attribute values in the clumps are stored with the most significant byte first, commonly known as network byte order or big-endian format.

A number of clumps contain index values into sequences in other clumps. These indexes start with the value 0 (0 is the first entry in the sequence).

The Syntactic Description Language (SDL) of ISO/IEC 14496-1:2001 is used to define the file format. In addition to the existing basic data types, the UVLC elementary data type is defined in this document. It shall be used to carry variable-length bit-fields that follow the JVT UVLC design.

Unrecognized clumps should be skipped and ignored.

### III.2 File Identification

The File Type Clump is the first clump of the file. JVT files shall be identified from a major Brand field equal to 'jvt'.

The preferred file extension is '.jvt'.

### III.3 Clump

#### III.3.1 Definition

Clumps start with a header, which gives both size and type. The header permits compact or extended size (32 or 64 bits). Most clumps will use the compact (32-bit) size. The size is the entire size of the clump, including the size and type header, fields, and all contained clumps. This facilitates general parsing of the file.

##### III.3.1.1 Syntax

```
aligned(8) class clump (unsigned int(32) clumpType)
    unsigned int(32) size;
    unsigned int(32) type = clumpType;

    if (size==1) {
        unsigned int(64) largesize;
    } else if (size==0) {
        // clump extends to end of file
    }
}
```

### III.3.1.2 Semantics

- size is an integer that specifies the number of bytes in this clump, including all its fields and contained clumps; if size is 1 then the actual size is in the field largesize; if size is 0, then this clump is the last one in the file, and its contents extend to the end of the file (normally only used for an Alternate Track Media Clump)
- type identifies the clump type; standard clumps use a compact type, which is normally four printable characters, to permit ease of identification, and is shown so in the clumps below.

### III.4 Clump Order

An overall view of the normal encapsulation structure is provided in the following table.

The table shows those clumps that may occur at the top-level in the left-most column; indentation is used to show possible containment. Thus, for example, an Alternate Track Header Clump (atrh) is found in a Segment Clump (segm).

Not all clumps need be used in all files; the mandatory clumps are marked with an asterisk (\*). See the description of the individual clumps for a discussion of what must be assumed if the optional clumps are not present.

There are restrictions in which order the clumps shall appear in a file. See the clump definitions for these restrictions.

**TABLE 15**

#### **Clump types.**

|      |      |   |         |                                                                                      |
|------|------|---|---------|--------------------------------------------------------------------------------------|
| ftyp |      | * | III.5.1 | File Type Clump, identifies the file format                                          |
| javh |      | * | III.5.2 | File Header Clump, file-level meta-data                                              |
| cinf |      |   | III.5.3 | Content Info Clump, describes file contents                                          |
| atin |      | * | III.5.4 | Alternate Track Info Clump, describes characteristics of tracks                      |
| prms |      | * | III.5.5 | Parameter Set Clump, enumerated set of frequently changing coding parameters         |
| segm |      | * | III.5.6 | Segment Clump, contains meta- and media data for a defined period of time            |
|      | atrh | * | III.5.7 | Alternate Track Header Clump, meta-data for a track                                  |
|      | swpc |   | III.5.9 | Switch Picture Clump, identifies pictures that can be used to switch between tracks. |
|      | atrm | * | III.5.8 | Alternate Track Media Clump, media data for a track                                  |

### III.5 Clump Definitions

#### III.5.1 File Type Clump

##### III.5.1.1 Definition

Clump Type: 'ftyp'

Container: File

Mandatory: Yes

Quantity: Exactly one

A media-file structured according to the ISO media file format specification may be compatible with more than one detailed specification, and it is therefore not always possible to speak of a single 'type' or 'brand' for the file. This clump identifies a JVT file in a similar fashion without claiming compatibility with the ISO format. However, it enables other file readers to identify the JVT file type. It must be placed first in the file.

### III.5.1.2 Syntax

```
aligned(8) class FileTypeClump aligned(8) extends clump('ftyp') {
    unsigned int(32) majorBrand = 'jvt ';
    unsigned int(16) jmMajorVersion;
    unsigned int(16) jmMinorVersion;
    unsigned int(32) compatibleBrands[]; // to end of the clump
}
```

### III.5.1.3 Semantics

This clump identifies the specification to which this file complies.

majorBrand is a brand identifier for the interim JVT file format. Only 'jvt ' shall be used for majorBrand, as the file format is not compatible with any other format.

jmMajorVersion and jmMinorVersion define the version of the standard working draft the file complies with. For example, JM-1 files shall have jmMajorVersion equal to 1 and jmMinorVersion equal to 0.

compatibleBrands is a list, to the end of the clump, of brands. Should only include the entry 'jvt '.

Note: As the interim JVT file format is based on the ISO media file format, it might be appropriate to allow a combination of many ISO media file format based file types into the same file. In such a case, the majorBrand might not be equal to 'jvt ' but 'jvt ' should be one of the compatibleBrands. As this option was not discussed in the Pattaya meeting, it is not reflected in the current specification of the interim JVT file format (this document).

## III.5.2 File Header Clump

### III.5.2.1 Definition

Clump Type: 'jvth'  
Container: File  
Mandatory: Yes  
Quantity: One or more

This clump must be placed as the second clump of the file.

The clump can be repeated at any position of the file when no container clump is open. A File Header Clump identifies a random access point to the file. In other words, no data prior to a selected File Header Clump is required to parse any of the succeeding data. Furthermore, any segment can be parsed without a forward reference to any of the data succeeding the particular segment.

### III.5.2.2 Syntax

```
aligned(8) class fileHeaderClump extends clump('jvth') {
    unsigned int(8) majorVersion = 0x00;
    unsigned int(8) minorVersion = 0x00;
    unsigned int(32) timescale;
    unsigned int(32) numUnitsInTick;
    unsigned int(64) duration;
    unsigned int(16) pixAspectRatioX;
    unsigned int(16) pixAspectRatioY;
    unsigned int(16) maxPicId;
    unsigned int(8) numAlternateTracks;
    unsigned int(2) numBytesInPayloadCountMinusOne;
    unsigned int(2) numBytesInPictureOffsetMinusTwo;
    unsigned int(2) numBytesInPictureDisplayTimeMinusOne;
    unsigned int(2) numBytesInPictureCountMinusOne;
}
```

```

        unsigned int(2) numBytesInPayloadSizeMinusOne;
    }

```

### III.5.2.3 Semantics

majorVersion and minorVersion indicate the version of the file format. This specification defines the format for version 0.0 (majorVersion.minorVersion). Version numbering is independent of working draft document and joint model software as well as the version of the standard | recommendation. This allows parsers interpret the high-level syntax of the files, even if decoding of a file according to the indicated joint model or standard version was not supported.

timescale is the number of time units which pass in one second. For example, a time coordinate system that measures time in sixtieths of a second has a time scale of 60.

numUnitsInTick is the number of time units according to timescale that correspond to one clock tick. A clock tick is the minimum unit of time that can be presented in the file. For example, if the clock frequency of a video signal is (30 000) / 1001 Hz, timescale should be 30 000 and numUnitsInTick should be 1001.

duration is an integer that declares length of the file (in the indicated timescale). Value zero indicates that no duration information is available.

pixAspectRatioX and pixAspectRatioY define the pixel geometry, calculated by pixAspectRatioX / pixAspectRatioY. Value zero in either or both of the attributes indicate an unspecified pixel aspect ratio.

maxPicId gives the maximum value for the picture identifier.

numAlternateTracks gives the number of alternative encodings of the same source. Typically each encoding is targeted for different bit-rate. Each file shall contain at least one track.

numBytesInPayloadCountMinusOne indicates the number of bytes that are needed to signal the maximum number of payloads in any picture. For example, numBytesInPayloadCountMinusOne equal to zero indicates that one byte is needed to signal the number of payloads, and the maximum number of payloads is 255.

numBytesInPictureOffsetMinusTwo indicates the number of bytes that are needed to signal picture offsets. For example, numBytesInPictureOffsetMinusTwo equal to zero indicates that the offsets are two-byte integer values with a range of -32768 to 32767.

numBytesInPictureDisplayTimeMinusOne indicates the number of bytes that are needed to signal picture display time offsets.

numBytesInPictureCountMinusOne indicates the number of bytes that are needed to signal the maximum number of pictures in a segment.

numBytesInPayloadSizeMinusOne indicates the number of bytes to signal the maximum payload size in bytes.

## III.5.3 Content Info Clump

### III.5.3.1 Definition

Clump Type:     'cinf'  
 Container:     File  
 Mandatory:     No  
 Quantity:       Zero or more

This clump gives information about the content of the file.

The clump can be repeated at any position of the file when no container clump is open.

### III.5.3.2 Syntax

```

aligned(8) class contentInfoClump extends clump('cinf') {
    unsigned int(64) creationTime;
}

```

```

    unsigned int(64) modificationTime;

    unsigned int(8) titleNumBytes;
    if (titleNumBytes)
        unsigned int(8)[titleNumBytes] title;

    unsigned int(8) authorNumBytes;
    if (authorNumBytes)
        unsigned int(8)[authorNumBytes] author;

    unsigned int(8) copyrightNumBytes;
    if (copyrightNumBytes)
        unsigned int(8)[copyrightNumBytes] copyright;

    unsigned int(16) descriptionNumBytes;
    if (descriptionNumBytes)
        unsigned int(8)[descriptionNumBytes] description;

    unsigned int(16) URINumBytes;
    if (URINumBytes)
        unsigned int(8)[URINumBytes] URI;
}

```

### III.5.3.3 Semantics

creationTime is an integer that declares the creation time of the presentation (in seconds since midnight, Jan. 1, 1904).

modificationTime is an integer that declares the most recent time the presentation was modified (in seconds since midnight, Jan. 1, 1904).

titleNumBytes gives the number of bytes in title.

title, if present, contains the title of the file coded according to ISO/IEC 10646-1 UTF-8.

authorNumBytes gives the number of bytes in author.

author, if present, contains the author of the source or the encoded representation in the file coded according to ISO/IEC 10646-1 UTF-8.

copyrightNumBytes gives the number of bytes in copyright.

copyright shall be used only to convey intellectual property information regarding the source or the encoded representation in the file. copyright is coded according to ISO/IEC 10646-1 UTF-8.

descriptionNumBytes gives the number of bytes in description.

description shall be used only to convey descriptive information associated with the information contents of the file. description is coded according to ISO/IEC 10646-1 UTF-8.

URINumBytes gives the number of bytes in URI.

URI contains a uniform resource identifier (URI), as defined in IETF RFC 2396. URI is coded according to ISO/IEC 10646-1 UTF-8. URI shall be used to convey any related information to the file.

## III.5.4 Alternate Track Info Clump

### III.5.4.1 Definition

Clump Type: 'atin'

Container: File  
Mandatory: Yes  
Quantity: One or more.

This clump specifies the characteristics of alternate tracks. The clump shall precede the first Segment Clump. The clump can be repeated at any position of the file when no container clump is open.

#### III.5.4.2 Syntax

```
aligned(8) class alternateTrackInfo {
    unsigned int(16) displayWindowWidth;
    unsigned int(16) displayWindowHeight;
    unsigned int(16) maxSDUSize;
    unsigned int(16) avgSDUSize;
    unsigned int(32) avgBitRate;
}

aligned(8) class alternateTrackInfoClump
    extends clump('atin') {
    (class alternateTrackInfo) trackInfo[numAlternateTracks];
}
```

#### III.5.4.3 Semantics

displayWindowWidth and displayWindowHeight declare the preferred size of the rectangular area on which video images are displayed. The values are interpreted as amount of pixels.

An SDU is defined as the payload and the payload header. maxSDUSize gives the size in bytes of the largest SDU of the track. avgSDUSize gives the average size of the SDU over the entire track. Value zero in either attribute indicates that no information is available.

avgBitRate gives the average bit-rate in bits/second over the entire track. Payloads and payload headers taken into account in the calculation.

### III.5.5 Parameter Set Clump

#### III.5.5.1 Definition

Clump Type: 'prms'  
Container: File  
Mandatory: Yes  
Quantity: One or more

This clump specifies a parameter set.

Parameter sets can be repeated in the file to allow random access. A parameter set is uniquely identified within a file based on parameterSetID. Decoders can infer a repetition of a parameter set if a set with the same parameterSetID has already appeared in a file. A redundant copy of a parameter set can safely be ignored.

#### III.5.5.2 Syntax

```
aligned(8) class parameterSetClump
    extends clump('prms') {
    unsigned int(16) parameterSetID;
    unsigned int(8) profile;
    unsigned int(8) level;
```

```

    unsigned int(8) version;
    unsigned int(16) pictureWidthInMBs;
    unsigned int(16) pictureHeightInMBs;
    unsigned int(16) displayRectangleOffsetTop;
    unsigned int(16) displayRectangleOffsetLeft;
    unsigned int(16) displayRectangleOffsetBottom;
    unsigned int(16) displayRectangleOffsetRight;
    unsigned int(8) displayMode;
    unsigned int(16) displayRectangleOffsetFromWindowTop;
    unsigned int(16) displayRectangleOffsetFromWindowLeftBorder;
    unsigned int(8) entropyCoding;
    unsigned int(8) motionResolution;
    unsigned int(8) partitioningType;
    unsigned int(8) intraPredictionType;
};

```

### III.5.5.3 Semantics

parameterSetId gives the identifier of the parameter set. The identifier shall be unique within a file.

profile defines the coding profile in use.

level defines the level in use within the profile.

version defines the version in use within the profile and the level.

pictureWidthInMBs and pictureHeightInMBs define the extents of the coded picture in macroblocks.

displayRectangleOffsetTop, displayRectangleOffsetLeft, displayRectangleOffsetBottom, and displayRectangleOffsetRight define the rectangle to be displayed from the coded picture. Pixel units are used.

displayMode defines the preferred displaying mode. Value zero indicates that the display rectangle shall be rescaled to fit onto the display window. No scaling algorithm is defined. Image shall be as large as possible, no clipping shall be applied, image aspect ratio shall be maintained, and image shall be centered in the display window. Value one indicates that the display rectangle shall be located as indicated in displayRectangleOffsetFromWindowTop and displayRectangleOffsetFromWindowLeftBorder. No scaling shall be done and clipping shall be applied to areas outside the display window. No fill pattern is defined for areas in the display window that are not covered by the display rectangle.

displayRectangleOffsetFromWindowTop and displayRectangleOffsetFromWindowLeftBorder indicate the location of the top-left corner of the display rectangle within the display window. The values are given in pixels. The values are valid only if displayMode is one.

**Error! Reference source not found.** clarifies the relation of different display rectangle and window related attributes. The dashed rectangle of in the decoded picture represents the display rectangle, which is indicated by displayRectangleOffsetTop, displayRectangleOffsetLeft, displayRectangleOffsetBottom, and displayRectangleOffsetRight.

entropyCoding equal to zero stands for UVLC, whereas value one stands for CABAC.

motionResolution equal to zero stands for full-pixel motion resolution, one stands for half-pixel motion resolution, two stands for 1/4-pixel motion resolution, and three stands for 1/8-pixel motion resolution.

partitioningType equal to zero stands for the single slice mode and one stands for the data partitioning mode.

intraPredictionType equal to zero stands for normal INTRA prediction, whereas one stands for the constrained INTRA prediction.

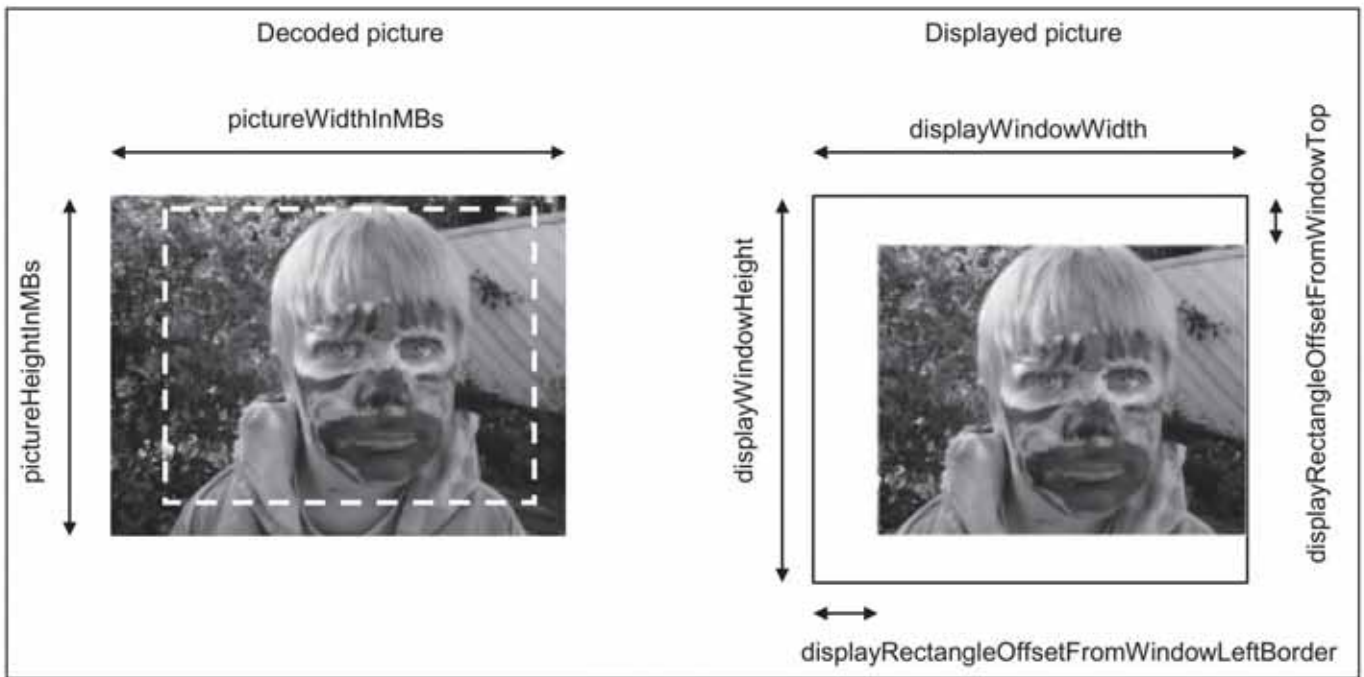


FIGURE 25

Relation of display window and rectangle attributes.

### III.5.6 Segment Clump

#### III.5.6.1 Definition

Clump Type: 'segm'  
 Container: File  
 Mandatory: Yes  
 Quantity: One or more

A segment clump contains data from a certain period of time. Segments shall not overlap in time. Segments shall appear in ascending order of time in the file. A segment clump is a container clump for several other clumps.

#### III.5.6.2 Syntax

```
aligned(8) class SegmentClump extends Clump('segm') {
    unsigned int(64) fileSize;
    unsigned int(64) startTick;
    unsigned int(64) segmentDuration;
}
```

#### III.5.6.3 Semantics

fileSize indicates the number of bytes from the beginning of the Segment Clump to the end of the file. Value zero indicates that no size information is available. When downloading a file to a device with limited storage capabilities, fileSize can be used to determine if a file fits into the available storage space. In a progressive downloading service, fileSize, startTick, and duration (in the File Header Clump) can be

used to estimate the average bit-rate of the file including meta-data. This estimation can then be used to decide how much initial buffering is needed before starting the playback.

startTick indicates the absolute time of the beginning of the segment since the beginning of the presentation (time zero). Any time offsets within the segment are relative to startTick.

segmentDuration indicates the duration of the segment. Value zero indicates that no duration information is available.

### III.5.7 Alternate Track Header Clump

#### III.5.7.1 Definition

Clump Type: 'ath'  
Container: Segment Clump ('segm')  
Mandatory: Yes  
Quantity: One or more

An alternate track represents an independent encoding of the same source as for the other alternate tracks. The Alternate Track Header Clump contains meta-data for an alternate track. The clumps shall appear in the same order in all Segment Clumps and they can be indexed starting from zero. Each succeeding clump is associated with an index one greater than the previous one. The index can be used to associate the clump with a particular track and with the information given in the Alternate Track Info Clump.

The clump contains an indication of the number of the pictures in the alternate track in this segment. In addition, the clump contains picture information for each of these pictures. Picture information shall appear in ascending order of picture identifiers (in modulo arithmetic). In other words, picture information shall appear in coding/decoding order of pictures.

A picture information block contains a pointer to the coded representation of the picture. A picture is associated with a display time and with a number of so-called payloads.

A payload refers to a slice, a data partition, or a piece of supplemental enhancement information. A payload header refers to an equivalent definition as in VCEG-N72R1. For example, a payload header of a single slice includes the "first byte", an indication of the parameter set in use, and the slice header.

#### III.5.7.2 Syntax

```
aligned(8) class payloadInfo {
    unsigned int((numBytesInPayloadSizeMinusOne + 1) * 8) payloadSize;
    unsigned int(8) headerSize;
    unsigned int(4) payloadType;
    unsigned int(1) errorIndication;
    unsigned int(3) reserved = 0;
    if (payloadType == 0) { // single slice
        UVLC parameterSet;
        sliceHeader;

    else if (payloadType == 1) { // partition A
        UVLC parameterSet;
        sliceHeader;
        UVLC sliceID;
    }
    else if (payloadType == 2 || partitionType == 3) { // Partition B
        or C
        UVLC pictureID;
```

```

        UVLC sliceID;
    }
    else if (payloadType == 5) { // Supplemental enhancement
information
        // no additional codewords
    }
}

aligned(8) class pictureInfo {
    bit intraPictureFlag;
    aligned(8) int((numBytesInPictureOffsetMinusTwo + 2) * 8)
pictureOffset;
    int((numBytesInPictureDisplayTimeMinusOne + 1) * 8)
pictureDisplayTime;
    unsigned int((numBytesInPayloadCountMinusOne + 1) * 8)
numPayloads;
    (class payloadInfo) payloadData[numPayloads];
}

aligned(8) class AlternateTrackHeaderClump extends Clump('atrh') {
    unsigned int((numBytesInPictureCountMinusOne + 1) * 8)
numPictures;
    (class pictureInfo) pictureData[numPictures];
}

```

### III.5.7.3 Semantics

payloadInfo gives information related to a payload. payloadSize indicates the number of bytes in the payload (excluding the payload header). The value of headerSize is the number of bytes in the payload header, i.e., the number of bytes remaining in the structure. The rest of the data is defined in VCEG-N72R1.

pictureInfo gives information related to a picture.

intraPictureFlag is set to one, when the picture is an INTRA picture. The flag is zero otherwise.

A picture pointer is maintained to point to the beginning of the latest picture in the corresponding Alternate Track Media Clump. The pointer is relative to the beginning of the Alternate Track Media Clump. pictureOffset gives the increment or the decrement (in bytes) for the picture pointer to obtain the coded data for the picture. Initially, before updating the pointer for the first picture of the alternate track in a segment, the picture pointer shall be zero.

pictureDisplayTime gives the time when the picture is to be displayed. It is assumed that the picture remains visible until the next picture is to be displayed. The value is relative to the corresponding value of the previous picture.

numPayloads indicates the number of payloads in the picture. payloadData is an array of payloadInfo structures signaling the characteristics of the payloads.

numPictures indicates the number of pictures in the track during the period of the segment. pictureData is an array of pictureInfo structures signaling the meta-data of the pictures.

## III.5.8 Alternate Track Media Clump

### III.5.8.1 Definition

Clump Type: 'atrm'

Container: Segment Clump ('segm')  
Mandatory: Yes  
Quantity: One or more

An alternate track represents an independent encoding of the same source as for the other alternate tracks. The Alternate Track Media Clump contains the media-data for an alternate track and for the duration of the segment. The clumps shall appear in the same order in all Segment Clumps and they can be indexed starting from zero. Each succeeding clump is associated with an index one greater than the previous one. The index can be used to associate the clump with a particular track and with the information given in other track-related clumps.

Pictures can appear in the clump in any order. This ensures that disposable pictures, such as conventional B pictures, can be located flexibly. Data for different pictures shall not overlap. Data for a picture consists of payloads, i.e., slices, data partitions, and pieces of supplemental enhancement information. Payloads shall appear in successive bytes, and the order of payloads shall be the same as in the Alternate Track Header Clump.

### III.5.8.2 Syntax

```
aligned(8) class AlternateTrackMediaClump extends Clump('atrm') {  
}
```

## III.5.9 Switch Picture Clump

### III.5.9.1 Definition

Clump Type: 'swpc'  
Container: Segment Clump ('segm')  
Mandatory: No  
Quantity: Zero or one

This clump defines which pictures can be used to switch from an alternate track to another. Typically these pictures are SP pictures.

### III.5.9.2 Syntax

```
aligned(8) class uniquePicture {  
    unsigned int(8) alternateTrackIndex;  
    unsigned int((numBytesInPictureCountMinusOne + 1) * 8)  
    pictureIndex;  
}  
  
aligned(8) class switchPictureSet {  
    unsigned int(8) numSyncPictures;  
    (class uniquePicture) syncPicture[numSyncPictures];  
}  
  
aligned(8) class switchPictureClump extends Clump('swpc') {  
    unsigned int((numBytesInPictureCountMinusOne + 1) * 8)  
    numSwitchPictures;  
    (class switchPictureSet) switchPicture[numSwitchPictures];  
}
```

### III.5.9.3 Semantics

uniquePicture uniquely identifies a picture within this segment. It contains two attributes: alternateTrackIndex and pictureIndex. alternateTrackIndex identifies the alternate track where the picture lies, and pictureIndex gives the picture index in coding order.

switchPictureSet gives a set of pictures that represent the same picture contents and can be used to replace any picture in the set as a reference picture for motion compensation. numSyncPictures gives the number of pictures in the set. syncPicture is an array of uniquePicture structures indicating which pictures belong to the set.

numSwitchPictures indicates the number of picture positions that have multiple interchangeable representations. switchPicture is an array of switchPictureSet structures indicating the set of pictures that can be used interchangeably for each picture position.

## Appendix IV Non-Normative Error Concealment

### IV.1 Introduction

It is assumed that no erroneous or incomplete slices are decoded. When all received slices of a picture have been decoded, skipped slices are concealed according to the presented algorithms. In practice, record is kept in a macroblock (MB) based status map of the frame. The status of an MB in the status map is "Correctly received" whenever the slice that the MB is included in was available for decoding, "Lost" otherwise. After the frame is decoded if the status map contains "Lost" MBs, concealment is started.

Given the slice structure and MB-based status map of a frame, the concealment algorithms were designed to work MB-based. The missing frame area (pixels) covered by MBs marked as "Lost" in the status map are concealed MB-by-MB (16x16 Y pixels, 8x8 U, V pixels). After an MB has been concealed it is marked in the status map as "Concealed". The order in which "Lost" MBs are concealed is important as also the "Concealed", and not only the "Correctly received" MBs are treated as reliable neighbors in the concealment process whenever no "Correctly received" immediate neighbor of a "Lost" MB exists. In such cases a wrong concealment can result in propagation of this concealment mistake to several neighbor concealed MBs. The processing order chosen is to take the MB columns at the edge of the frame first and then move inwards column-by-column so to avoid a concealment mistake made in the usually "difficult" (discontinuous motion areas, large coded prediction error) center part of the frame propagate to the "easy" (continuous motion area, similar motion over several frames) side parts of the frame.

FIGURE 26 shows a snapshot of the status map during the concealment phase where already concealed MBs have the status of "Concealed", and the currently processed (concealed) MB is marked as "Current MB".

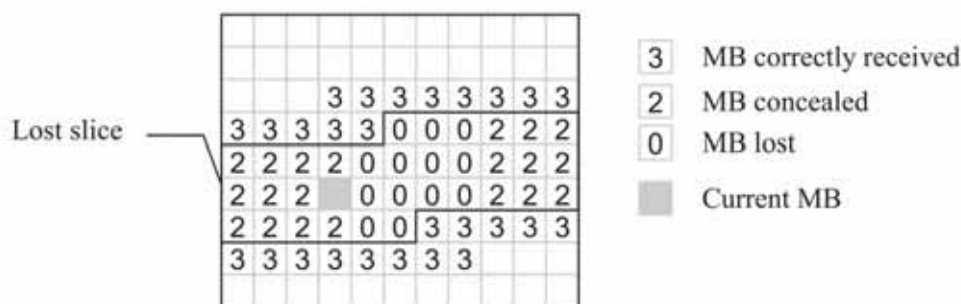


FIGURE 26

MB status map at the decoder

### IV.2 INTRA Frame Concealment

Lost areas in INTRA frames have to be concealed spatially as no prior frame may resemble the INTRA frame. The selected spatial concealment algorithm is based on weighted pixel averaging presented in *A. K. Katsaggelos and N. P. Galatsanos (editors), "Signal Recovery Techniques for Image and Video Compression and Transmission", Chapter 7, P. Salama, N. B. Shroff, and E. J. Delp, "Error Concealment in Encoded Video Streams", Kluwer Academic Publishers, 1998.*

Each pixel value in a macroblock to be concealed is formed as a weighted sum of the closest boundary pixels of the selected adjacent macroblocks. The weight associated with each boundary pixel is relative to the inverse distance between the pixel to be concealed and the boundary pixel. The following formula is used:

$$\text{Pixel value} = (\sum a_i \times (B - d_i)) / \sum (B - d_i)$$

where  $a_i$  is the pixel value of a boundary pixel in an adjacent macroblock,  $B$  is the horizontal or vertical block size in pixels, and  $d_i$  is the distance between the destination pixel and the corresponding boundary pixel in the adjacent macroblock.

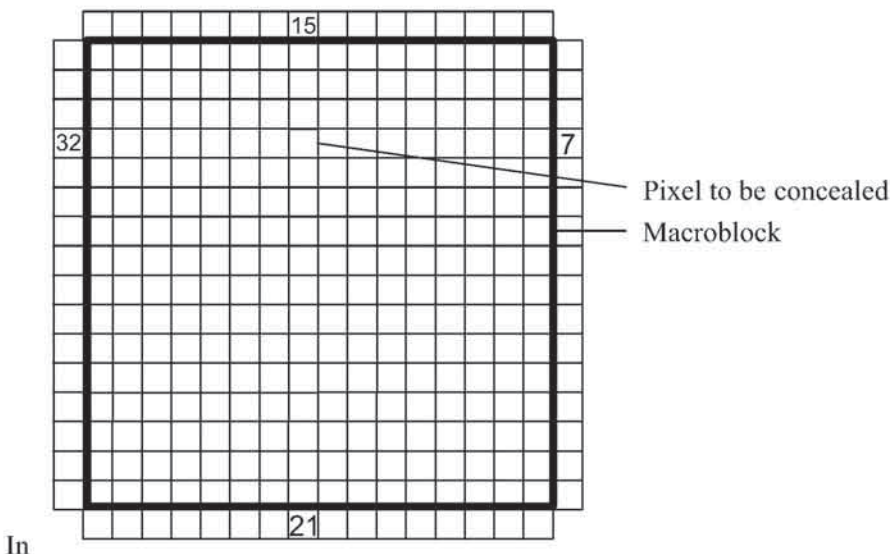


FIGURE 27, the shown destination pixel is calculated as follows

$$\text{Pixel value} = (15 \times (16-3) + 21 \times (16-12) + 32 \times (16-7) + 7 \times (16-8)) / (13 + 4 + 9 + 8) \approx 18$$

Only "Correctly received" neighboring MBs are used for concealment if at least two such MBs are available. Otherwise, neighboring "Concealed" MBs are also used in the averaging operation.

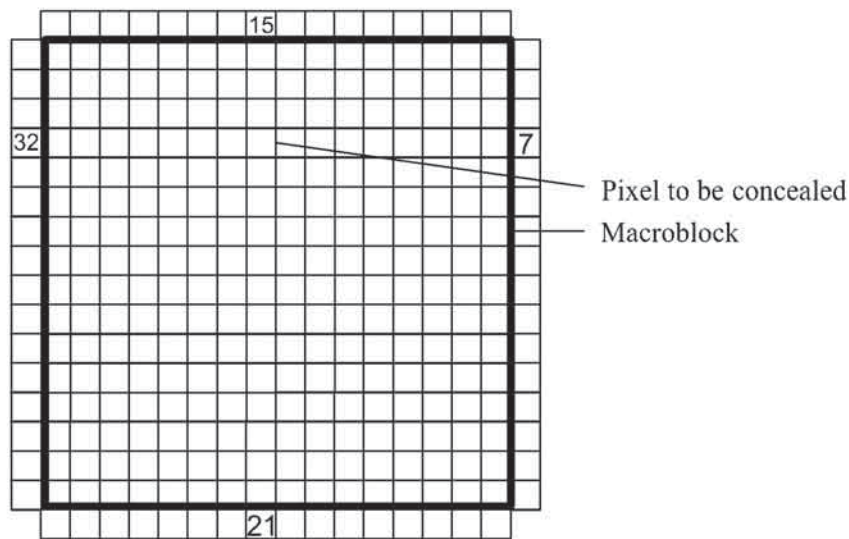


**FIGURE 27**

**Spatial concealment based on weighted pixel averaging**

### IV.3 INTER and SP Frame Concealment

#### IV.3.1 General

Instead of directly operating in the pixel domain a more efficient approach is to try to "guess" the motion in the missing pixel area (MB) by some kind of prediction from available motion information of spatial or temporal neighbors. This "guessed" motion vector is then used for motion compensation using the reference frame. The copied pixel values give the final reconstructed pixel values for concealment, and no additional pixel domain operations are used. The presented algorithm is based on *W.-M. Lam, A. R. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors," in Proc. ICASSP'93, Minneapolis, Apr. 1993, pp. V417-V420.*

### IV.3.2 Concealment using motion vector prediction

The motion activity of the correctly received slices of the current picture is investigated first. If the average motion vector is smaller than a pre-defined threshold (currently  $\frac{1}{4}$  pixels for each motion vector component), all the lost slices are concealed by copying from co-located positions in the reference frame. Otherwise, motion-compensated error concealment is used, and the motion vectors of the lost macroblocks are predicted as described in the following paragraphs.

The motion of a "Lost" MB is predicted from a spatial neighbor MB's motion relying on the statistical observation, that the motion of spatially neighbor frame areas is highly correlated. For example, in a frame area covered by a moving foreground scene object the motion vector field is continuous, which means that it is easy to predict.

The motion vector of the "Lost" MB is predicted from one of the neighbor MBs (or blocks). This approach assumes, that the motion vector of one of the neighbor MBs (or blocks) models the motion in the current MB well. It was found in previous experiments, that median or averaging over all neighbors' motion vectors does not give better results. For simplicity, in the current implementation the smallest neighbor block size that is considered separately as predictor is set to 8x8 Y pixels. The motion of any 8x8 block is calculated as the average of the motion of the spatially corresponding 4x4 or other shaped (e.g. 4x8) blocks.

The decision of which neighbor's motion vectors to use as prediction for the current MB is made based on the smoothness of the concealed (reconstructed) image. During this trial procedure the concealment pixel values are calculated using the motion vector of each candidate (motion compensated pixel values). The motion vector, which results in the smallest luminance change across block boundaries when the block is inserted into its place in the frame is selected. (see FIGURE 28). The zero motion vector case is always considered and this copy concealment (copy pixel values from the co-located MB in the reference frame) is evaluated similarly as the other motion vector candidates.

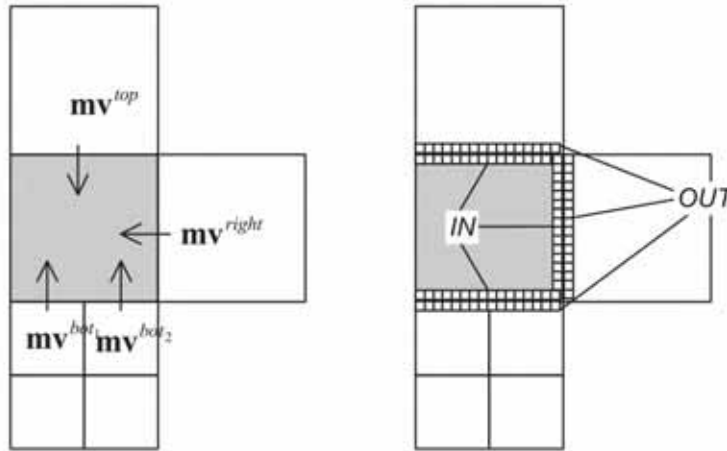


FIGURE 28

#### Selecting the motion vector for prediction

The winning predictor motion vector is the one which minimizes the side match distortion  $d_{sm}$ , which is the sum of absolute Y pixel value difference of the *IN*-block and neighboring *OUT*-block pixels at the boundaries of the current block:

$$\min_{dir \in \{top, bot, left, right\}} \arg \left\langle d_{sm} = \sum_{j=1}^{N} \left| \hat{Y}(\mathbf{mv}^{dir})_j^{IN} - \gamma_j^{OUT} \right| \right\rangle$$

When "Correctly received" neighbor MBs exist the side match distortion is calculated only for them. Otherwise all the "Concealed" neighbor MBs are included in the calculation.

# **EXHIBIT 2**

**Declaration of Dr. Michael Orchard in Support of Nokia and ITC Staff's  
Proposed Construction for "Skip Coding Mode" in U.S. Patent No. 7,532,808**

I, Michael Orchard, declare and state as follows:

1. I have been asked by Nokia to provide my professional opinion on how a person of ordinary skill in the art would have understood the term "skip coding mode" in U.S. Patent No. 7,532,808.
2. I am being compensated for my work on this case at my standard consulting fee of \$500 per hour. I am also being reimbursed for the reasonable expenses that I incur in this matter. My compensation does not depend on the outcome of this litigation, and I do not have any financial interest in any party to the case.
3. The opinions presented in this declaration are my own and, to the best of my knowledge and belief, are true and correct.
4. In reaching my opinions, I have relied upon my experience in the video coding, computer science, multimedia systems fields as well as my review of the specification and file history of the '808 Patent. In forming my opinions for this Declaration, I have also reviewed Respondents' proposed claim construction.

**I. Qualifications**

5. I am a Professor of Electrical and Computer Engineering at Rice University in Houston, Texas. I received my Bachelor of Science in Electrical Engineering from San Diego State University in 1981. In 1986, I received a Master of Science in Electrical Engineering from San Diego State University. In 1988, I received a Master of Arts in Electrical Engineering from Princeton University, and in 1990, I received a Ph.D. in Electrical Engineering from Princeton University.

6. My graduate studies concentrated in the areas of image and video processing techniques. For example, my doctoral dissertation detailed a method for displaying digital images on video display monitors with limited palette size, and advanced methods for video compression.

7. During my graduate studies, I also worked at AT&T Bell Laboratories in the Video Communications department. After I received my Ph.D. degree, I stayed on at AT&T Bell Laboratories as a consultant while I took a position as Assistant Professor in the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. During my time at the University of Illinois, I taught undergraduate courses related to fundamental digital signal processing techniques and graduate courses in advanced digital signal processing and image processing. I continued my academic career as an Associate Professor at Princeton University where I continued to teach classes in digital signal processing and image processing.

8. Since 2001, I have been a Professor in the Department of Electrical and Computer Engineering at Rice University. In this position, I teach a variety of undergraduate and graduate level courses in digital signal processing and digital image processing. My research continues to explore image and video compression algorithms, motion estimation and compensation, and image rendering and applications thereof, including use of compression and other techniques in digital cameras, cellular telephones, and multiple computer platforms.

9. I have consulted extensively in the fields of digital signal processing and image and video processing, and I have held consulting positions with AT&T Bell Laboratories (and later AT&T Laboratories), NEC Research Institute, Flextronics Corporation, Schlumberger, and Interdigital.

10. In the course of my academic and industrial career, I have published over 100 technical papers for journals and conferences on various aspects of my research, and I am a named inventor on seven issued U.S. patents.

11. I have served as an Associate Editor for the IEEE Transactions on Image Processing, and I have been Co-General Chair of OSA's Digital Image Processing and Analysis Conference, and Technical Program Chair for the IEEE International Conference on Image Processing.
12. I have supervised 12 Ph.D. students on dissertations related to various image and video processing topics, three of whom are now tenured professors at research universities. My students and I have made significant research contributions to the development of state-of-the-art digital signal processing algorithms for image and video compression. My research contributions in these areas led me to be named IEEE Fellow in 2000, "for contributions to the theory and development of image and video compression algorithms."
13. I have submitted my curriculum vitae (CV), which includes a more detailed summary of my background, experience, and publications, attached as Appendix A.
14. I have included a list of cases in which I have testified as an expert at trial or by deposition for the date range specified in Appendix A.

## **II. Opinion**

15. It is my opinion that the term "skip coding mode," as used in the claims of the '808 Patent would be understood by a person of ordinary skill in the art in light of the patent specification to mean "a coding mode in which a zero (non-active) motion vector or a non-zero (active) motion vector is associated with each skip mode macroblock, depending on the characteristics of the motion in image segments surrounding the macroblock in question," as Nokia and Staff both propose. Nokia and Staff's construction comes from the definition given by the '808 specification at 17:43-53.
16. I do not agree with the Respondents' construction that adds the clause: "and residual information can be provided for each skip mode macroblock." In my view, a person of ordinary

skill in the art would clearly understand that the '808 Patent teaches a “skip coding mode” that provides no “residual information” for any skip mode macroblocks. A person of ordinary skill in the art reaches this conclusion by following the unambiguous teachings of the specification of the '808 Patent.

17. The '808 Patent lists JM1 or “Joint Model 1 (JM-1)” under “Other Publications” on the cover page. JM1, which had the document title JVT-A003, was a reference document for the Joint Video Team of ISO/IEC MPEG and ITU-T VCEG.<sup>1</sup> JM1 included a number of coding modes for macroblocks, including a SKIP mode and a non-skip mode of interframe coding called INTER mode.

18. To contrast how pixels are reconstructed in skip-mode macroblocks vs. non-skip-mode macroblocks, a POSITA would find the '808 Patent's description of INTER mode reconstruction helpful:

“... the blocks of pixel values derived from the reference frame, are passed from motion compensated prediction block 240 to combiner 230 where they are combined with the decoded prediction error information. In practice, the pixel values of each predicted block are added to corresponding reconstructed prediction error values output by inverse DCT transformer 220. In this way an array of reconstructed pixel values for each block of the macroblock is obtained.” '808 Patent at 10:8-16

A POSITA would recognize the '808 Patent's description of the reconstruction of pixel values in INTER mode to accurately reflect the operation of the INTER mode decoding in JM1 of the JVT codec, and would understand that the “prediction error values” referred to in this description can also be referred to as “residual information.”

19. Contrasting with pixel value reconstruction for INTER mode macroblocks, the '808 Patent describes SKIP mode as:

---

<sup>1</sup> ISO/IEC MPEG stands for International Organization for Standardization / International Electrotechnical Commission Moving Pictures Expert Group, and ITU-T VCEG stands for International Telecommunications Union – Telecommunications Sector Video Coding Experts Group.

“... an eighth coding mode, known as SKIP (or skip) mode, is provided. If this mode is assigned to a macroblock, this indicates that the macroblock is to be copied from the reference video frame without using motion compensated prediction.” ’808 Patent at 10:63-67

Subclause 3.4.1 of JM1 states that, “A macroblock is called skipped if no information is sent. In that case the reconstruction of an inter macroblock is made by copying the collocated picture material from the last decoded frame.” A POSITA would understand that pixel values of skip-mode macroblocks in JM1 of the JVT codec are reconstructed directly from pixel values of the reference frame, with no “residual information” contributing to the reconstruction.

20. A key novelty of the ’808 Patent lies in its modified definition of “skip-mode”:

, “the present invention is based on a redefinition of the skip mode concept used in JM1 of the JVT codec. ... the skip mode concept is redefined in such a way that a macroblock assigned to skip mode is either associated with a zero (non-active) motion vector, in which case it is treated in the same way as a conventional skip mode macroblock and copied directly from the reference frame, or it is associated with a non-zero (active) motion vector” ’808 Patent at 14:16-29.

A POSITA would understand from this disclosure that the ’808 Patent’s definition of “skip mode” includes two versions, one associated with “active” motion vectors and the other associated with “non-active” motion vectors, and that at least one of those versions (“non-active”) operates like conventional JM1 JVT codec “skip mode” which unambiguously does not provide any “residual information.” Thus, based on the section of the ’808 Patent shown above, as well as disclosures of several similar sections (’808 Patent at 14:42-48, 18:6-11 and 21:50-53), a POSITA would recognize that residual information can never be provided for any skip mode macroblock associated with a “non-active” motion vector, and thus the Respondents’ proposed construction directly conflicts with the teachings of the ’808 specification.

21. But the Respondents’ proposed construction does not only conflict with teachings of the ’808 Patent specification relating to skip mode macroblocks associated with “non-active” motion

vectors. A POSITA would find several reasons to also conclude from the teachings that are provided that no residual information is involved in the reconstruction of skip mode macroblocks associated with “active” motion vectors.

22. First, the '808 specification defines its novel “skip modes” as modifications of the conventional JM1 codec “skip mode”, and then describes the modifications as only modifying the motion vector for skip mode macroblocks associated with “active” motion as shown in the example below:

If it is found that the surrounding region exhibits a certain type of motion, a non-zero motion vector representative of that motion is generated and associated with the current macroblock. In particular, the continuity, velocity or deviation of motion in the surrounding macroblocks or sub-blocks can be analyzed. For example, if the motion in the surrounding region exhibits a certain level of continuity, a certain common velocity, or a particular form of divergence, a motion vector representative of that motion can be assigned to the current macroblock to be coded. '808 Patent at 14:32-42

The fact that this disclosure, as well as all other disclosures about skip mode macroblocks associated with “active” motion, describe modifications to the motion vector used in conventional prior skip modes, while omitting any descriptions of modifications to any processing after motion compensation leads a POSITA to conclude that no such additional modifications are intended.

23. Second, several sections of the '808 specification make it clear that no residual information is involved in the reconstruction of the skip mode macroblocks associated with the new “active” motion vectors:

“If active motion is detected around a macroblock to be coded/decoded, motion parameters conforming to the motion are generated and the macroblock is motion compensated. In this way, no additional information needs to be transmitted from the encoder to the decoder.” '808 Patent at 15:63-67

A POSITA would understand that the fact that no additional information is transmitted clearly precludes the possibility that any “residual information” can be provided for the reconstruction of pixel values of macroblocks associated with “active” motion vectors in skip coding mode.

24. Lastly, and perhaps most convincingly, the '808 specification states that pixel values of “active” skip mode macroblocks are reconstructed directly from motion compensation from a reference frame:

“If the macroblock ... If, on the other hand, it is associated with a non-zero valued motion vector, a region of pixel values indicated by the non-zero motion vector is used to generate the pixel values for the macroblock.” '808 Patent at 21:50-56

A POSITA would understand this disclosure to teach that pixel values of macroblocks associated with “active” motion vectors are generated directly from a region of pixel values from a reference frame pointed to by the non-zero motion vector that is calculated, and thus no “residual information” is provided for these macroblocks associated with “active” motion vectors in skip coding mode.

25. Because the '808 specification clearly precludes providing any “residual information” for the reconstruction of skip mode macroblocks associated with “non-active” motion, and the teachings of the '808 also indicate that no “residual information” is provided for the reconstruction of skip mode macroblocks associated with “active” motion, a POSITA would understand that the Respondents’ proposed construction is completely inconsistent with the teachings of the '808 Patent specification. A POSITA would understand that a modified version of Respondents’ construction that would be more true: residual information can never be provided for any skip mode macroblock.

26. Under the Respondents’ proposed construction, the redefined skip coding mode (which would replace the skip coding mode of JM1) would include residual information. To a person of ordinary skill in the art, this would be an oxymoron. People of ordinary skill in the art understand that a key aspect of skip coding modes is that the coding, transmission, and decoding of residual information is “skipped” for the skip coding mode macroblocks.

27. The '808 Patent explains that the “redefinition of the skip mode concept as proposed by the present invention has significant technical advantages compared with the previously described prior art coding methods.” For example, the '808 Patent describes that “in contrast to both global motion compensation and global motion vector coding methods previously described, no additional information must be transmitted in the video bit-stream to enable correct decoding of the video data.” '808 Patent at 15:7-11. A person of ordinary skill in the art would understand the '808 Patent to be explaining that no additional information, such as residual information, can be transmitted as part of the redefined skip coding mode.

28. Additionally, the '808 Patent explains that “a minimal amount of modification is required to incorporate the method according to the invention into existing video coding systems that employ the concept of skip mode macroblocks.” '808 Patent at 15:11-14. A person of ordinary skill in the art would understand that it would require a substantial and inefficient modification to implement residual information within a skip coding mode and that such a design would not be based on the JM1 of the JVT codec. In fact, under the Respondents' proposed construction, the resulting coding modes in Table 3 '808 Patent would lack any coding mode in which no residual information is provided. A person of ordinary skill in the art would not understand this to be the intention of the '808 Patent.

### **III. Conclusion**

29. In conclusion, it is my opinion that a person of ordinary skill in the art would understand Complainant's and Staff's proposed construction of “skip coding mode” to be consistent with the definition and disclosures of the '808 Patent. A person of ordinary skill in the art would understand that the '808 Patent's redefined skip coding mode unambiguously does not include the possibility of providing any “residual information” as required by the construction proposed by Respondents.

M. F. Orchard  
Dr. Michael Orchard

2/12/2024  
Date

**CERTIFICATE OF SERVICE**

I, Melissa Kalka, hereby certify that on February 13, 2024, a true and correct copy of the foregoing document was served by the indicated means to the persons at the addresses below:

|                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| The Honorable Lisa R. Barton, Secretary<br><b>U.S. International Trade Commission</b><br>500 E Street, S.W., Room 112<br>Washington, DC 20436                                                                                                                                                                                                                                                                                        | <input type="checkbox"/> Via First Class Mail<br><input type="checkbox"/> Via Express Delivery<br><input type="checkbox"/> Via Hand Delivery<br><input checked="" type="checkbox"/> Via EDIS                                                                                                                                                                                                                                                                                                            |
| The Honorable Doris Johnson Hines<br>Administrative Law Judge<br><b>U.S. International Trade Commission</b><br>500 E Street, S.W.<br>Washington, DC 20436                                                                                                                                                                                                                                                                            | <input type="checkbox"/> Via Hand Delivery<br><input type="checkbox"/> Via Express Delivery<br><input type="checkbox"/> Via First Class Mail<br><input checked="" type="checkbox"/> Via EDIS<br><input type="checkbox"/> Other: Email to<br><a href="mailto:JohnsonHines1379@usitc.gov">JohnsonHines1379@usitc.gov</a><br><a href="mailto:Pyong.Yoon@usitc.gov">Pyong.Yoon@usitc.gov</a>                                                                                                                |
| Marissa Ducca, Esq.<br>Jeffrey Hsu, Esq.<br><b>Office of Unfair Import Investigations</b><br><b>U.S. International Trade Commission</b><br>500 E Street, S.W., Room 401<br>Washington, DC 20436                                                                                                                                                                                                                                      | <input type="checkbox"/> Via Hand Delivery<br><input type="checkbox"/> Via Express Delivery<br><input type="checkbox"/> Via First Class Mail<br><input checked="" type="checkbox"/> Other: Email and Box to<br><a href="mailto:Marissa.Ducca@usitc.gov">Marissa.Ducca@usitc.gov</a><br><a href="mailto:Jeffrey.Hsu@usitc.gov">Jeffrey.Hsu@usitc.gov</a>                                                                                                                                                 |
| Stefani E. Shanberg<br>Robin L. Brewer<br><b>PERKINS COIE LLP</b><br>505 Howard Street, Suite 1000<br>San Francisco, California 94105-3204<br><br>Thomas N. Millikan<br><b>PERKINS COIE LLP</b><br>11452 El Camino Real, Suite 300<br>San Diego, CA 92130-2080<br><br>Veronica S. Ascarrunz<br><b>PERKINS COIE LLP</b><br>700 13th Street, NW, Suite 800<br>Washington, DC 20005-3960<br><br>Stephen S. Korniczky<br>Martin R. Bader | <input type="checkbox"/> Via Hand Delivery<br><input type="checkbox"/> Via Express Delivery<br><input type="checkbox"/> Via First Class Mail<br><input checked="" type="checkbox"/> Other: Email to<br><a href="mailto:Nokia_Amazon_ITC_Perkins@perkinscoie.com">Nokia_Amazon_ITC_Perkins@perkinscoie.com</a><br><a href="mailto:LegalTm-Amazon-Nokia@sheppardmullin.com">LegalTm-Amazon-Nokia@sheppardmullin.com</a><br><a href="mailto:amazon-nokia-ITC@fenwick.com">amazon-nokia-ITC@fenwick.com</a> |

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>SHEPPARD,MULLIN, RICHTER &amp; HAMPTON LLP</b><br/> 12275 El Camino Real, Suite 100<br/> San Diego, CA 92130-4092</p> <p>J. David Hadden<br/> Vigen Salmastlian<br/> Allen Wang<br/> <b>FENWICK &amp; WEST LLP</b><br/> 801 California Street<br/> Mountain View, CA 94041</p> <p><i>Counsel for Respondents Amazon.com, Inc. and Amazon.com Services LLC</i></p>                                                                                                                                                                                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <p>Eric S. Namrow<br/> Stephanie L. Roberts<br/> Kandis C. Gibson<br/> Manu Bansal, Ph.D.<br/> Kevin J. Spinella<br/> Eunjean Je<br/> <b>Morgan, Lewis &amp; Bockius LLP</b><br/> 1111 Pennsylvania Ave, N.W.<br/> Washington, D.C. 20004-2541</p> <p>David J. Levy<br/> Rick L. Rambo<br/> Thomas R. Davis<br/> <b>Morgan, Lewis &amp; Bockius LLP</b><br/> 1000 Louisiana Street, Suite 4000<br/> Houston, TX 77002-5005</p> <p>Jason E. Gettleman<br/> Corey R. Houmand<br/> <b>Morgan, Lewis &amp; Bockius LLP</b><br/> 1400 Page Mill Road<br/> Palo Alto, CA 94304</p> <p>Kevin J. Post<br/> Lance W. Shapiro<br/> <b>ROPES &amp; GRAY LLP</b><br/> 1211 Avenue of the Americas<br/> New York, NY 10036<br/> Telephone: (212) 596-9000<br/> Facsimile: (212) 596-9090</p> <p>Matthew J. Rizzolo</p> | <p><input type="checkbox"/> Via Hand Delivery<br/> <input type="checkbox"/> Via Express Delivery<br/> <input type="checkbox"/> Via First Class Mail<br/> <input checked="" type="checkbox"/> Other: Email to<br/> <a href="mailto:HP-Nokia-ITC@morganlewis.com">HP-Nokia-ITC@morganlewis.com</a><br/> <a href="mailto:HP-Nokia-Ropes1379@ropesgray.com">HP-Nokia-Ropes1379@ropesgray.com</a><br/> <a href="mailto:matthew.rizzolo@ropesgray.com">matthew.rizzolo@ropesgray.com</a></p> |

|                             |  |
|-----------------------------|--|
| <b>ROPES &amp; GRAY LLP</b> |  |
|-----------------------------|--|

|                          |  |
|--------------------------|--|
| 2099 Pennsylvania Ave NW |  |
|--------------------------|--|

|                      |  |
|----------------------|--|
| Washington, DC 20006 |  |
|----------------------|--|

|                           |  |
|---------------------------|--|
| Telephone: (202) 508-4600 |  |
|---------------------------|--|

|                           |  |
|---------------------------|--|
| Facsimile: (202) 508-4650 |  |
|---------------------------|--|

|                                       |  |
|---------------------------------------|--|
| <i>Counsel for Respondent HP Inc.</i> |  |
|---------------------------------------|--|

/s/ *Melissa Kalka*

Melissa Kalka

Senior Paralegal

McKool Smith