

EXHIBIT B-4
MPEG-1
RENDERS OBVIOUS U.S. PATENT NO. 8,050,321

As described in the following claim chart, claims 8 and 10 of U.S. Patent No. 8,050,321 (the "'321 patent") are invalid in view of ISO/IEC 11172 ("MPEG-1"), entitled Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s. MPEG-1 was published on August 1, 1993, and is prior art under at least 35 U.S.C. § 102(a) and (b).

As set forth herein, it would have been obvious to a person of ordinary skill in the art as of the priority date of '321 patent to modify and/or to combine the teachings of MPEG-1 with the knowledge of one of ordinary skill in the art to show all the limitations of the claim, as set forth in detail in the chart below.

Specifically, MPEG-1 teaches a concept referred to as a "closed GOP" which is indicated using a "closed_gop_flag" as disclosed below. The MPEG-1 also expressly teaches that there are two orders of frames within a closed GOP, the bitstream order (which is the order the frames are located in the bitstream and decoded at the decoder), and the display order (which is the order the frames are displayed in the video). As MPEG-1 explains, a closed GOP must always start with an I-picture in bitstream order:

Property 1. A group of pictures, in bitstream order, must start with an I-picture and may be followed by any number of I, P or B-pictures in any order.

MPEG-1 at D.5.2 (p. 67)

However, in display order, a closed GOP can begin with either an I-picture or a B-picture:

Property 2. Another property of a group of pictures is that it must begin, in display order, with an I or a B-picture, and must end with an I or a P-picture. The smallest group of pictures consists of a single I-picture, whereas the largest size is unlimited.

MPEG-1 at D.5.2 (p. 67)

MPEG-1 provides two exemplary closed GOPs in display order: group (a) which begins with an I-picture and is referred to as a "typical example," and group (b) which begins with leading B-pictures and is referred to as "less typical":

To the extent that these Invalidity Contentions rely on or otherwise embody particular constructions of terms or phrases in the Asserted Claims, Respondents are not proposing any such constructions as proper constructions of those terms or phrases. Various positions put forth in this document are predicated on Complainants' incorrect and overly broad interpretation of its claims. Those positions are not intended to and do not necessarily reflect Respondents' interpretation of the true and proper scope of Complainants' claims, and Respondents reserve the right to adopt claim construction positions that differ from or even conflict with various positions put forth in this document.

A typical example of a closed group is shown in figure D.18a.

I	B	B	P	B	B	P	B	B	P	B	B	P
0	1	2	3	4	5	6	7	8	9	10	11	12

(a) closed group

B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

(b) open or closed group

Figure D.18 -- Example groups of pictures in display order

A less typical example of a closed group is shown in figure D.18b. In this example, the B-pictures which precede the first I-picture must use backward motion compensation only, i.e. any motion compensation must be based only on picture number 2 in the group.

MPEG-1 at D.5.2 (p. 69)

The decoding order or bitstream order of the closed GOP is “the order which the decoder needs [the pictures] to decode them.” MPEG-1 at D.5.3.2. The decoding order is different from the display order due to the inclusion of B-pictures—which can use both past and future reference pictures for motion prediction:

Because of the conflicting requirements of random access and highly efficient compression, three main picture types are defined. Intra-coded pictures (I-Pictures) are coded without reference to other pictures. They provide access points to the coded sequence where decoding can begin, but are coded with only a moderate compression ratio. Predictive coded pictures (P-Pictures) are coded more efficiently using motion compensated prediction from a past intra or predictive coded picture and are generally used as a reference for further prediction. Bidirectionally-predictive coded pictures (B-Pictures) provide the highest degree of compression but require both past and future reference pictures for motion compensation. Bidirectionally-predictive coded pictures are never used as references for prediction. The organisation of the three picture types in a sequence is very flexible. The choice is left to the encoder and will depend on the requirements of the application. Figure 1 illustrates the relationship between the three different picture types.

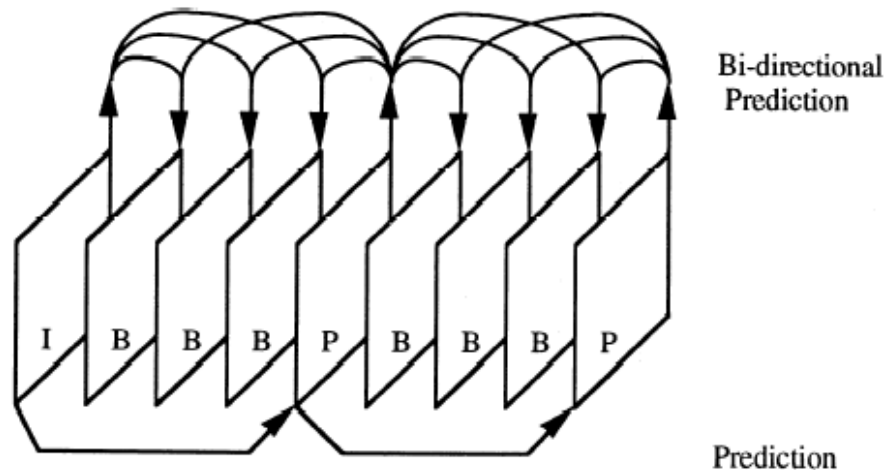


Figure 1 -- Example of temporal picture structure

MPEG-1 at 0.2.1 (p. v).

In other words, the sequence of only I-pictures and P-pictures is the same in both the display order and the decoding order. However, B-pictures must be decoded after the future P-picture or I-picture that is used for prediction, so the display order is different than the decoding order. In order to determine the display order, the decoder decodes a value called the “temporal reference” from the bitstream for each picture:

The Temporal Reference is a ten-bit number which can be used to define the order in which the pictures must be displayed. It may be useful since pictures are not transmitted in display order, but rather in the order which the decoder needs to decode them. The first picture, in display order, in each group must have Temporal Reference equal to zero. This is incremented by one for each picture in the group.

MPEG-1 at D.5.3.2 (p. 70).

It would have been obvious to a POSITA to modify MPEG-1, and specifically the structure of the closed GOP, to allow closed GOPs to begin only with an I-picture in display order. In other words, it would have been obvious to require closed GOPs to follow the general structure of D.18(a) and not permit closed GOPs to follow the general structure of D.18(b) (i.e., GOPs beginning with B-pictures in display order). In fact, a POSITA would have been motivated to make such a modification. A POSITA would have known that closed GOPs of the general structure of D.18(b) were rarely, if ever, utilized. A POSITA would have also known that closed GOPs of the general structure of D.18(b) were computationally more difficult to create and were impractical. As such, a POSITA would have been motivated to limit closed GOPs to the general structure of D.18(a) (i.e., GOP's beginning with I-pictures in display order). A POSITA would have also had a reasonable expectation of success in implementing such a modification. In fact, such closed GOPs were already described as the "typical example" of closed GOPs in the MPEG-1 standard. A POSITA would have had a reasonable expectation that using only the "typical example" of closed GOPs would be successful and would not pose any problems or challenges.

The obvious modified version of MPEG-1 described above, would mean that for every closed GOP, the first frame after the closed_GOP_flag = 0 would be an I-frame that is the first frame in decoding order and in display order (having a temporal reference of zero). As such, in a decoder using the obvious modified version of MPEG-1 described above, it would have been obvious to a POSITA to implement that the decoder would perform a hard-reset of the temporal reference number for the first frame in decoding order following the closed_GOP_flag = 0. A POSITA would have found two such implementations obvious:

First, it would have been obvious to implement a system in which the decoder would (1) decode the closed_GOP_flag = 0, (2) decode the temporal reference for the first frame (an I-picture) following the closed_GOP_flag=0, and (3) reset the temporal reference of the first frame (an I-picture) to zero. The proposed obvious code in the decoder would have a structure similar to the example below:

```
Decode GOP Start Code
  Picture Count = 0
  Decode GOP Header
    Decode closed_GOP_flag

  While next_start_code = Picture Start Code
    Picture Count = Picture Count + 1
    Decode Picture Header
      Decode temporal_reference
    If Picture Count == 1
```

temporal_reference = 0

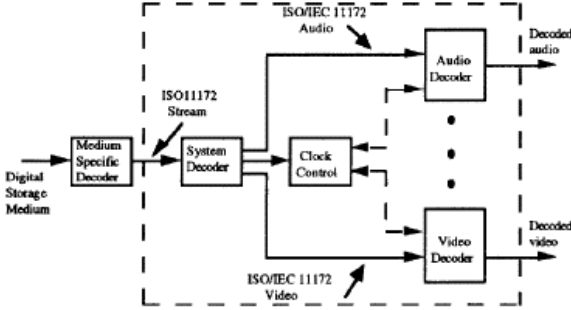
A POSITA would have been motivated to use this implementation in order to include a built-in redundancy into the decoder as part of an error prevention system. Such an implementation would mean that, even if the bitstream had been damaged such that the temporal_reference of the first frame in display order of the closed GOP was not accurate, the decoder would be able to correct the inaccuracy and correctly display the video sequence. A POSITA would also have had a reasonable expectation that such modification would have been successful. Such modification is a simple and ordinary modification to the code of the decoder, which would not have been difficult to implement.

Second, it would have been obvious to implement a system in which the decoder would (1) decode the closed_GOP_flag = 0, (2) reset the temporal reference of the first frame (an I-picture) to zero, and (3) only decode the temporal reference for the frames following the first frame in decoding order. The proposed obvious code in the decoder would have a structure similar to the example below:

```
Decode GOP Start Code
  Picture Count = 0
  Decode GOP Header
    Decode closed_GOP_flag

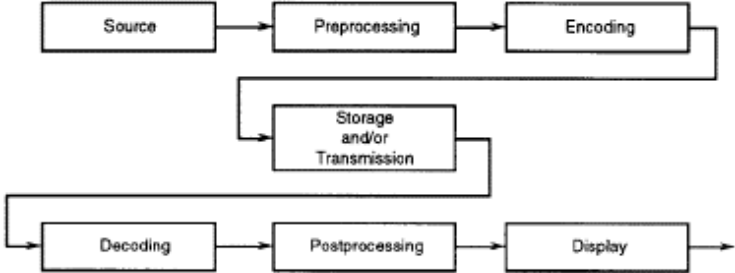
  While next_start_code = Picture Start Code
    Picture Count = Picture Count + 1
    Decode Picture Header
      If Picture Count == 1
        temporal_reference = 0
      Else
        Decode temporal_reference
```

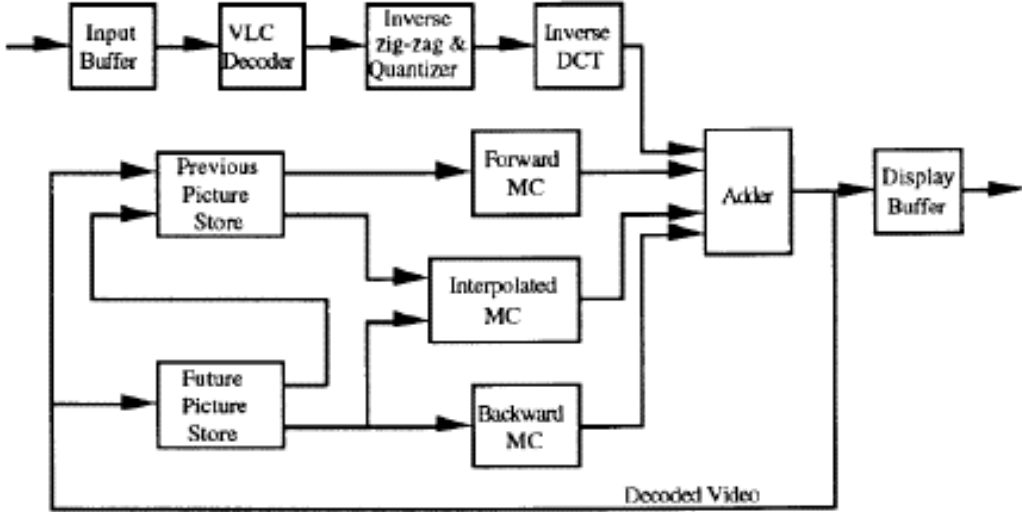
A POSITA would have been motivated to use this implementation in order to implement efficiency into the decoder and to avoid the redundancy of decoding the temporal reference for the first frame. Such an implementation would mean that a decoder would save the processing power and speed needed to decode the temporal reference of the first frame following the closed GOP flag and could automatically set the value of the temporal reference to zero. A POSITA would also have had a reasonable expectation that such modification would have been successful. Such modification is a simple and ordinary modification to the code of the decoder, which would not have been difficult to implement. Further details and explanations of these grounds are provided below.

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
8[preamble] A method for decoding a compressed video sequence, the method comprising:	MPEG-1, as evidenced by the example citations below, discloses a method for decoding a compressed video sequence. <i>See, e.g.,</i> MPEG-1, at iv-v. A prototypical audio/video decoder system is depicted in figure 1 to illustrate the function of an ISO/IEC 11172 decoder. The architecture is not unique -- System Decoder functions including decoder timing control might equally well be distributed among elementary stream decoders and the Medium Specific Decoder — but this figure is useful for discussion. The prototypical decoder design does not imply any normative requirement for the design of an ISO/IEC 11172 decoder. Indeed non-audio/video data is also allowed, but not shown.  Figure 1 -- Prototypical ISO/IEC 11172 decoder The prototypical ISO/IEC 11172 decoder shown in figure 1 is composed of System, Video, and Audio decoders conforming to Parts 1,2, and 3, respectively, of ISO/IEC 11172. In this decoder the multiplexed coded representation of one or more audio and/or video streams is assumed to be stored on a digital storage medium (DSM), or network, in some medium-specific format. The medium specific format is not governed by this International Standard, nor is the medium-specific decoding part of the prototypical ISO/IEC 11172 decoder. The prototypical decoder accepts as input an ISO/IEC 11172 multiplexed stream and relies on a System Decoder to extract timing information from the stream. The System Decoder demultiplexes the stream, and the elementary streams so produced serve as inputs to Video and Audio decoders, whose outputs are decoded video and audio signals. Included in the design, but not shown in the figure, is the flow of timing

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	information among the System Decoder, the Video and Audio Decoders, and the Medium Specific Decoder. The Video and Audio Decoders are synchronized with each other and with the DSM using this timing information. ISO/IEC 11172 multiplexed streams are constructed in two layers: a system layer and a compression layer. The input stream to the System Decoder has a system layer wrapped about a compression layer. Input streams to the Video and Audio decoders have only the compression layer. Operations performed by the System Decoder either apply to the entire ISO/IBC 11172 multiplexed stream ("multiplex-wide operations"), or to individual elementary streams ("stream-specific operations"). The ISO/IEC 11172 system layer is divided into two sub-layers, one for multiplex-wide operations (the pack layer), and one for stream-specific operations (the packet layer). MPEG-1, at viii. "Decoding is the inverse of the encoding operation. It is considerably simpler than encoding as there is no need to perform motion estimation and there are many fewer options. The decoding process is defined by this part of ISO/IEC 11172. The description that follows is a very brief overview of one possible way of decoding a bitstream. Other decoders with different architectures are possible. Figure 4 shows the main functional blocks."

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	MPEG-1, Fig. 4. Where DCT^{-1} is inverse discrete cosine transform Q^{-1} is dequantization MUX^{-1} is demultiplexing VLD is variable length decoding Figure 4 -- Basic video decoder block diagram MPEG-1, viii. As the decoder reads the bitstream, it identifies the start of a coded picture and then the type of the picture. It decodes each macroblock in the picture in turn. The macroblock type and the motion vectors, if present, are used to construct a prediction of the current macroblock based on past and future reference pictures that have been stored in the decoder. The coefficient data are decoded and dequantized. Each 8 by 8 block of coefficient data is transformed by an inverse DCT (specified in annex A), and the result is added to the prediction signal and limited to the defined range. After all the macroblocks in the picture have been processed, the picture has been reconstructed. If it is an I-picture or a P-picture it is a reference picture for subsequent pictures and is stored, replacing the oldest stored reference picture. Before the pictures are displayed they may need to be re-ordered from the coded order to their natural display order. After reordering, the pictures are available, in digital form, for post-processing and display in any manner that the application chooses. MPEG-1, Fig. D.1.

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	 <pre> graph LR Source[Source] --> Preprocessing[Preprocessing] Preprocessing --> Encoding[Encoding] Encoding --> Storage[Storage and/or Transmission] Storage --> Decoding[Decoding] Decoding --> Postprocessing[Postprocessing] Postprocessing --> Display[Display] Display --> Out[] </pre> Figure D.1 -- Coding and decoding process

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	MPEG-1, at 56. A simplified block diagram of a possible decoder implementation is shown below:  Figure D.7 -- Simplified decoder block diagram MPEG-1, at 56-57. "It is instructive to follow the method which the decoder uses to decode a bitstream containing the sequence of pictures given in Fig D.4, and display them in the order given in Fig D.3. The following description is simplified for clarity. The input bitstream is accumulated in the Input Buffer until needed. The Variable Length Code (VLC) Decoder decodes the header of the first picture, picture 0, and determines that it is an I-picture. The VLC Decoder produces quantized coefficients corresponding to the quantized DCT coefficients. These are assembled for each 8 by 8 block of pels in the image by inverse zig-zag scanning. The Inverse Quantizer produces the actual DCT coefficients using the quantization step size. The coefficients are then transformed into pel values by the

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	Inverse DCT transformer and stored in the Previous Picture Store and the Display Buffer. The picture may be displayed at the appropriate time. The VLC Decoder decodes the header of the next picture, picture 3, and determines that it is a P-picture. For each block, the VLC Decoder decodes motion vectors giving the displacement from the stored previous picture, and quantized coefficients corresponding to the quantized DCT coefficients of the difference block. These quantized coefficients are inverse quantized to produce the actual DCT coefficients. The coefficients are then transformed into pel difference values and added to the predicted block produced by applying the motion vectors to blocks in the stored previous picture. The resultant block is stored in the Future Picture Store and the Display Buffer. This picture cannot be displayed until B-pictures 1 and 2 have been received, decoded, and displayed. The VLC Decoder decodes the header of the next picture, picture 1, and determines that it is a B-picture. For each block, the VLC decoder decodes motion vectors giving the displacement from the stored previous or future pictures or both, and quantized coefficients corresponding to the quantized DCT coefficients of the difference block. These quantized coefficients are inverse quantized to produce the actual DCT coefficients. The coefficients are then inverse transformed into difference pel values and added to the predicted block produced by applying the motion vectors to the stored pictures. The resultant block is then stored in the Display Buffer. It may be displayed at the appropriate time. The VLC Decoder decodes the header of the next picture, picture 2, and determines that it is a B-picture. It is decoded using the same method as for picture 1. After decoding picture 2, picture 0, which is in the Previous Picture Store, is no longer needed and may be discarded. The VLC Decoder decodes the header of the next picture, picture 6, and determines that it is a P-picture. The picture in the Future Picture Store is copied into the Previous Picture Store, then decoding proceeds as for picture 3. Picture 6 should not be displayed until pictures 4 and 5 have been received and displayed. The VLC Decoder decodes the header of the next picture, picture 4, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 5, and determines that it is a B-picture. It is decoded using the same method as for picture 1.

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	The VLC Decoder decodes the header of the next picture, picture 9, and determines that it is a P-picture. It then proceeds as for picture 6. The VLC Decoder decodes the header of the next picture, picture 7, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 8, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 12, and determines that it is an I-picture. It is decoded using the same method as for picture 0. This process is repeated for the subsequent pictures. MPEG-1, AT 61. <pre> graph LR subgraph Model_Decoder [Model Decoder] Input_Buffer[Input Buffer] Picture_Decoder[Picture Decoder] Input_Buffer --> Picture_Decoder end Coded_Bitstream[Coded Bitstream] --> Input_Buffer Picture_Decoder --> Decoded_Pictures[Decoded Pictures] </pre> Figure D.14 -- Model decoder <i>Further, to the extent Complainant contends this element is not expressly or inherently disclosed by MPEG-1, it would have been obvious in view of the disclosures in MPEG-1, in combination with the knowledge of a person of ordinary skill in the art.</i> 8[a] decoding from the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of an independent MPEG-1, as evidenced by the example citations below, discloses decoding from the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of an independent sequence, wherein all motion-compensated temporal prediction references of the independent sequence refer only to image frames within the independent sequence.

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
sequence, wherein all motion-compensated temporal prediction references of the independent sequence refer only to image frames within said independent sequence;	<i>See, e.g.,</i> MPEG-1, at 3: 2.1.13 bidirectionally predictive-coded picture; B-picture [video]: A picture that is coded using motion compensated prediction from a past and/or future reference picture. MPEG-1, at 6: 2.1.71 group of pictures [video]: A series of one or more coded pictures intended to assist random access. The group of pictures is one of the layers in the coding syntax defined in this part of ISO/IEC 11172. MPEG-1, at 6: 2.1.79 intra-coded picture; I-picture [video]: A picture coded using information only from itself. MPEG-1, at 8: 2.1.111 predictive-coded picture; P-picture [video]: A picture that is coded using motion compensated prediction from the past reference picture. MPEG-1, at 15-16: Video sequence A coded video sequence commences with a sequence header and is followed by one or more groups of pictures and is ended by a sequence end code. Immediately before each of the groups of pictures there may be a sequence header. Within each sequence, pictures shall be decodable continuously. In each of these repeated sequence headers all of the data elements with the permitted exception of those defining the quantization matrices (load _intra_quantizer_matrix quantizer matrix and load non _intra_quantizer_matrix quantizer matrix and optionally intra_quantizer_matrix and non _intra_quantizer_matrix) quantizer matrix) shall have the same values as in the first sequence header. The quantization matrices may be redefined each time that a sequence header occurs in the bitstream. Thus the data elements load _intra_quantizer_matrix, quantizer matrix,

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																																		
	load non _intra_quantizer_matrix quantizer matrix and optionally intra_quantizer_matrix and non _intra_quantizer_matrix quantizer matrix may have any (non-forbidden) values. Repeating the sequence header allows the data elements of the initial sequence header to be repeated in order that random access into the video sequence is possible. In addition the quantization matrices may be changed inside the video sequence as required. Sequence header A video sequence header commences with a sequence header code and is followed by a series of data elements. Group of pictures A group of pictures is a series of one or more coded pictures intended to assist random access into the sequence. In the stored bitstream, the first coded picture in a group of pictures is an I-Picture. The order of the pictures in the coded stream is the order in which the decoder processes them in normal playback. In particular, adjacent B-Pictures in the coded stream are in display order. The last coded picture, in display order, of a group of pictures is either an I-Picture or a P-Picture. The following is an example of groups of pictures taken from the beginning of a video sequence. In this example the first group of pictures contains seven pictures and subsequent groups of pictures contain nine pictures. There are two B-pictures between successive P-pictures and also two B-pictures between successive I- and P-pictures. Picture 'I1' is used to form a prediction for picture `4P'. Pictures `4P' and '11' are both used to form predictions for pictures `2B' and `3B'. Therefore the order of pictures in the coded sequence shall be `11', `4P, `2B', `3B'. However, the decoder should display them in the order `1F, `2B', `3B', `4P'. At the encoder input, <table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td></tr><tr><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td></tr></table>	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	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P
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																											
I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P																											

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	At the encoder output, in the stored bitstream, and at the decoder input, 1 4 2 3 7 5 6 10 8 9 13 11 12 16 14 15 19 17 18 22 20 21 25 23 24 I P B B P B B I B B P B B P B B I B B P B B P B B where the double vertical bars mark the group of pictures boundaries. Note that in this example, the first group of pictures is two pictures shorter than subsequent groups of pictures, since at the beginning of video coding there are no B-pictures preceding the first I-Picture. However, in general, in display order, there may be B-Pictures preceding the first I-Picture in the group of pictures, even for the first group of pictures to be decoded. At the decoder output, 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 A group of pictures may be of any length. A group of pictures shall contain one or more I-Pictures. Applications requiring random access, fast-forward playback, or fast and normal reverse playback may use relatively short groups of pictures. Groups of pictures may also be started at scene cuts or other cases where motion compensation is ineffective. MPEG-1, at 18: 2.4.2.1 Start Codes

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																		
	Start codes are reserved bit patterns that do not otherwise occur in the video stream. All start codes are bytealigned. <table border="1" data-bbox="764 334 1461 743"> <thead> <tr> <th>Name</th><th>Hexadecimal value</th></tr> </thead> <tbody> <tr> <td>picture_start_code</td><td>00000100</td></tr> <tr> <td>slice_start_codes (including slice_vertical_positions)</td><td>00000101</td></tr> <tr> <td></td><td>through</td></tr> <tr> <td></td><td>000001AF</td></tr> <tr> <td>reserved</td><td>000001B0</td></tr> <tr> <td>reserved</td><td>000001B1</td></tr> <tr> <td>user_data_start_code</td><td>000001B2</td></tr> <tr> <td>sequence_header_code</td><td>000001B3</td></tr> <tr> <td>sequence_error_code</td><td>000001B4</td></tr> <tr> <td>extension_start_code</td><td>000001B5</td></tr> <tr> <td>reserved</td><td>000001B6</td></tr> <tr> <td>sequence_end_code</td><td>000001B7</td></tr> <tr> <td>group_start_code</td><td>000001B8</td></tr> <tr> <td>system start codes (see note)</td><td>000001B9</td></tr> <tr> <td></td><td>through</td></tr> <tr> <td></td><td>000001FF</td></tr> </tbody> </table> NOTE - System start codes are defined in ISO/IEC 11172-1. MPEG-1, at 24: 2.4.3.2 Sequence header Sequence header code -- The sequence header code is the bit string 000001B3 in hexadecimal. It identifies the beginning of a sequence header. MPEG-1, at 26: 2.4.3.3 Group of pictures layer group start code — The group start code is the bit string 000001B8 in hexadecimal. It identifies the beginning of a group of pictures. MPEG-1, at 26: closed gop — this is a one-bit flag which may be set to '1' if the group of pictures has been encoded without motion vectors pointing to the previous group of pictures."	Name	Hexadecimal value	picture_start_code	00000100	slice_start_codes (including slice_vertical_positions)	00000101		through		000001AF	reserved	000001B0	reserved	000001B1	user_data_start_code	000001B2	sequence_header_code	000001B3	sequence_error_code	000001B4	extension_start_code	000001B5	reserved	000001B6	sequence_end_code	000001B7	group_start_code	000001B8	system start codes (see note)	000001B9		through		000001FF
Name	Hexadecimal value																																		
picture_start_code	00000100																																		
slice_start_codes (including slice_vertical_positions)	00000101																																		
	through																																		
	000001AF																																		
reserved	000001B0																																		
reserved	000001B1																																		
user_data_start_code	000001B2																																		
sequence_header_code	000001B3																																		
sequence_error_code	000001B4																																		
extension_start_code	000001B5																																		
reserved	000001B6																																		
sequence_end_code	000001B7																																		
group_start_code	000001B8																																		
system start codes (see note)	000001B9																																		
	through																																		
	000001FF																																		

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																						
	MPEG-1, at 27: 2.4.3.4 Picture Layer picture start code -- The picture start code is a string of 32-bits having the value 00000100 in hexadecimal. temporal reference -- The temporal reference is a 10-bit unsigned integer associated with each input picture. It is incremented by one, modulo 1024, for each input picture. For the earliest picture (in display order) in each group of pictures, the temporal reference is reset to zero. MPEG-1, at 54: A GOP is a random access point, i.e. it is the smallest coding unit that can be independently decoded within a sequence, and consists of a header and some number of pictures. The GOP header contains time and editing information." MPEG-1, at 54: <table><tr><td>I</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td></tr><tr><td>0</td><td>3</td><td>1</td><td>2</td><td>6</td><td>4</td><td>5</td><td>9</td><td>7</td><td>8</td><td>12</td><td>10</td><td>11</td><td>15</td><td>13</td><td>14</td><td>18</td><td>16</td><td>17</td></tr></table> Figure D.4 -- Typical sequence of pictures in bitstream order MPEG-1, at 56-57: It is instructive to follow the method which the decoder uses to decode a bitstream containing the sequence of pictures given in Fig D.4, and display them in the order given in Fig D.3. The following description is simplified for clarity. The input bitstream is accumulated in the Input Buffer until needed. The Variable Length Code (VLC) Decoder decodes the header of the first picture, picture 0, and determines that it is an I-picture. The VLC Decoder produces quantized coefficients corresponding to the quantized DCT coefficients. These are assembled for each 8 by 8 block of pels in the image by inverse zig-zag scanning. The Inverse Quantizer produces the actual DCT coefficients using the quantization step size. The coefficients are then transformed into pel values by the Inverse	I	P	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	0	3	1	2	6	4	5	9	7	8	12	10	11	15	13	14	18	16	17
I	P	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B																					
0	3	1	2	6	4	5	9	7	8	12	10	11	15	13	14	18	16	17																					

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	DCT transformer and stored in the Previous Picture Store and the Display Buffer. The picture may be displayed at the appropriate time. The VLC Decoder decodes the header of the next picture, picture 3, and determines that it is a P-picture. For each block, the VLC Decoder decodes motion vectors giving the displacement from the stored previous picture, and quantized coefficients corresponding to the quantized DCT coefficients of the difference block. These quantized coefficients are inverse quantized to produce the actual DCT coefficients. The coefficients are then transformed into pel difference values and added to the predicted block produced by applying the motion vectors to blocks in the stored previous picture. The resultant block is stored in the Future Picture Store and the Display Buffer. This picture cannot be displayed until B-pictures 1 and 2 have been received, decoded, and displayed. The VLC Decoder decodes the header of the next picture, picture 1, and determines that it is a B-picture. For each block, the VLC decoder decodes motion vectors giving the displacement from the stored previous or future pictures or both, and quantized coefficients corresponding to the quantized DCT coefficients of the difference block. These quantized coefficients are inverse quantized to produce the actual DCT coefficients. The coefficients are then inverse transformed into difference pel values and added to the predicted block produced by applying the motion vectors to the stored pictures. The resultant block is then stored in the Display Buffer. It may be displayed at the appropriate time. The VLC Decoder decodes the header of the next picture, picture 2, and determines that it is a B-picture. It is decoded using the same method as for picture 1. After decoding picture 2, picture 0, which is in the Previous Picture Store, is no longer needed and may be discarded. The VLC Decoder decodes the header of the next picture, picture 6, and determines that it is a P-picture. The picture in the Future Picture Store is copied into the Previous Picture Store, then decoding proceeds as for picture 3. Picture 6 should not be displayed until pictures 4 and 5 have been received and displayed. The VLC Decoder decodes the header of the next picture, picture 4, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 5, and determines that it is a B-picture. It is decoded using the same method as for picture 1.

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	The VLC Decoder decodes the header of the next picture, picture 9, and determines that it is a P-picture. It then proceeds as for picture 6. The VLC Decoder decodes the header of the next picture, picture 7, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 8, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 12, and determines that it is an I-picture. It is decoded using the same method as for picture 0. This process is repeated for the subsequent pictures. MPEG-1, at 67-68: Two distinct picture orderings exist, the display order and the bitstream order (as they appear in the video bitstream). A group of pictures (gop) is a set of pictures which are contiguous in display order. A group of pictures must contain at least one I-picture. This required picture may be followed by any number of I and P-pictures. Any number of B-pictures may be interspersed between each pair of I or P-pictures, and may also precede the first I-picture. Property 1. A group of pictures, in bitstream order, must start with an I-picture and may be followed by any number of I, P or B-pictures in any order. Property 2. Another property of a group of pictures is that it must begin, in display order, with an I or a B-picture, and must end with an I or a P-picture. The smallest group of pictures consists of a single I-picture, whereas the largest size is unlimited. The original concept of a group of pictures was a set of pictures that could be coded and displayed independently of any other group. In the final version of this part of ISO/IEC 11172 this is not always true, and any B-pictures preceding (in display order) the first I-picture in a group may require the last picture in the previous group in order to be decoded. Nevertheless encoders can still construct groups of pictures which are independent of one another. One way to do this is to omit any B-pictures preceding the first I-picture. Another way is to allow such B-pictures, but to code them using only backward motion compensation.

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	Property 3. From a coding point of view, a concisely stated property is that a group of pictures begins with a group of pictures header, and either ends at the next group of pictures header or at the next sequence header or at the end of sequence, whichever comes first. MPEG-1, at 68: Some examples of groups of pictures are given below: <pre> I I P P I B P B P B B I B P B P B B I B B P B B P B B P B I B B B B P B I B B I I </pre> Figure D.17 -- Examples of groups of pictures in display order MPEG-1, at 69. Closed GOP A one bit flag follows the time code. It denotes whether the group of pictures is open or closed. Closed groups can be decoded without using decoded pictures of the previous group for motion compensation, whereas open groups require such pictures to be available. MPEG-1, AT 70: The picture layer contains all the coded information for one picture. The header identifies the temporal reference of the picture, the picture coding type, the delay in the video buffer verifier (VBV) and, if appropriate, the range of motion vectors used.

MPEG-1, at 70:

A picture begins with a picture header. The header starts with a picture start code. This code is byte-aligned and is 32 bits long. Its value is:

hex: 00000100

binary: 000000000000000000000000100000000

It may be preceded by any number of zeros.

MPEG-1, at 70:

The Temporal Reference is a ten-bit number which can be used to define the order in which the pictures must be displayed. It may be useful since pictures are not transmitted in display order, but rather in the order which the decoder needs to decode them. The first picture, in display order, in each group must have Temporal Reference equal to zero. This is incremented by one for each picture in the group.

Some example groups of pictures with their Temporal Reference numbers are given below:

Example (a) in display order	I	B	P	B	P								
	0	1	2	3	4								
Example (a) in decoding order	I	P	B	P	B								
	0	2	1	4	3								
Example (b) in display order	B	B	I	B	B	P	B	B	P	B	B	P	
	0	1	2	3	4	5	6	7	8	9	10	11	
Example (b) in coded order	I	B	B	P	B	B	P	B	B	P	B	B	
	2	0	1	5	3	4	8	6	7	11	9	10	
Example (c) in display order	B	I	B	B	B	B	P	B	I	B	B	I	I
	0	1	2	3	4	5	6	7	8	9	10	11	12
Example (c) in coded order	I	B	P	B	B	B	B	I	B	I	B	B	I
	1	0	6	2	3	4	5	8	7	11	9	10	12

Figure D.19 -- Examples of groups of pictures and temporal references

MPEG-1, at 71:

A three bit number follows the temporal reference. This is an index into the following table defining the type of picture.

Table D.5 -- Picture types

CODE	PICTURE TYPE
000	Forbidden
001	I-picture
010	P-picture
011	B-picture
100	D Picture
101	Reserved
110	Reserved
111	Reserved

MPEG-1, at 103:

At the beginning of a sequence, the decoder will decode the sequence header including the sequence parameters. If a parameter exceeds the capability of the decoder, then the decoder should report this. If the decoder determines that it can decode the bitstream, then it will set up its parameters to match those defined in the sequence header. This will include the horizontal and vertical resolutions and aspect ratio, the bit rate, and the quantization matrices.

Next the decoder will decode the group of pictures header, including the closed gop and broken-link information, and take any appropriate action. It will decode the first picture header in the group of pictures and read the vbv-delay field. If the decoder uses the vbv-delay information to start-up decoding rather than the information in the system stream (ISO/IEC 11172- 1) then it must delay displaying pictures until after a time determined by the vbv delay information and a knowledge of the decoder's architecture.

If the closed-gop flag is 0, indicating that the group is open, and the broken-link flag is 1, then any Bpictures preceding (in display order) the first I-picture in the group cannot be decoded. The decoder may adopt one of several strategies. It may display the first I-picture during the time that the undecodable Bpictures would be displayed. This strategy maintains audio synchronization and buffer fullness. However it is likely that the broken link has occurred because of post coding editing, in which case audio may be discontinuous. An alternative

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
8[b] starting the decoding of the video sequence from said first image frame of the independent sequence, whereby the video sequence is decoded without prediction from any image frame decoded prior to said first image frame;	strategy might be to discard the B-pictures entirely, and delay decoding the I-picture until the buffer fullness is within limits. If playback begins from a random point in the bitstream, the decoder should discard all the bits until it finds a sequence start code, a group of pictures start code, or a picture start code which introduces an I-picture. The slices and macroblocks in the picture are decoded and written into a display buffer, and perhaps into another buffer. The decoded pictures may be post processed and displayed in the order defined by the temporal reference at the picture rate defined in the sequence header. <i>Further, to the extent Complainant contends this element is not expressly or inherently disclosed by MPEG-1, it would have been obvious in view of the disclosures in MPEG-1, in combination with the knowledge of a person of ordinary skill in the art.</i> MPEG-1, as evidenced by the example citations below, discloses starting the decoding of the video sequence from the first image frame of the independent sequence, whereby the video sequence is decoded without prediction from any image frame decoded prior to the first image frame. <i>See, e.g.,</i> MPEG-1, at 6: 2.1.71 group of pictures [video]: A series of one or more coded pictures intended to assist random access. The group of pictures is one of the layers in the coding syntax defined in this part of ISO/IEC 11172. MPEG-1, at 9: 2.1.120 random access: The process of beginning to read and decode the coded bitstream at an arbitrary point. MPEG-1, at 15-16: Group of pictures A group of pictures is a series of one or more coded pictures intended to assist random access into the sequence. In the stored bitstream, the first coded picture in a group of pictures is an I-Picture. The order of the pictures in

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																																																																																																													
	the coded stream is the order in which the decoder processes them in normal playback. In particular, adjacent B-Pictures in the coded stream are in display order. The last coded picture, in display order, of a group of pictures is either an I-Picture or a P-Picture. The following is an example of groups of pictures taken from the beginning of a video sequence. In this example the first group of pictures contains seven pictures and subsequent groups of pictures contain nine pictures. There are two B-pictures between successive P-pictures and also two B-pictures between successive I- and P-pictures. Picture 'I1' is used to form a prediction for picture '4P'. Pictures '4P' and 'I1' are both used to form predictions for pictures '2B' and '3B'. Therefore the order of pictures in the coded sequence shall be 'I1', '4P', '2B', '3B'. However, the decoder should display them in the order '1F', '2B', '3B', '4P'. At the encoder input, <table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td></tr><tr><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td></tr></table> At the encoder output, in the stored bitstream, and at the decoder input, <table><tr><td>1</td><td>4</td><td>2</td><td>3</td><td>7</td><td>5</td><td>6</td><td>10</td><td>8</td><td>9</td><td>13</td><td>11</td><td>12</td><td>16</td><td>14</td><td>15</td><td>19</td><td>17</td><td>18</td><td>22</td><td>20</td><td>21</td><td>25</td><td>23</td><td>24</td></tr><tr><td>I</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td></tr></table> where the double vertical bars mark the group of pictures boundaries. Note that in this example, the first group of pictures is two pictures shorter than subsequent groups of pictures, since at the beginning of video coding there are no B-pictures preceding the first I-Picture. However, in general, in display order, there may be B-Pictures preceding the first I-Picture in the group of pictures, even for the first group of pictures to be decoded. At the decoder output, <table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td></tr></table> A group of pictures may be of any length. A group of pictures shall contain one or more I-Pictures. Applications requiring random access, fast-forward playback, or fast and normal reverse	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	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	1	4	2	3	7	5	6	10	8	9	13	11	12	16	14	15	19	17	18	22	20	21	25	23	24	I	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	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
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																																																																																																						
I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P																																																																																																						
1	4	2	3	7	5	6	10	8	9	13	11	12	16	14	15	19	17	18	22	20	21	25	23	24																																																																																																						
I	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B																																																																																																						
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																																																																																																						

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																																																												
	playback may use relatively short groups of pictures. Groups of pictures may also be started at scene cuts or other cases where motion compensation is ineffective. MPEG-1, at 26: closed gop -- This is a one-bit flag which may be set to '1' if the group of pictures has been encoded without motion vectors pointing to the previous group of pictures. group-start-code -- The group-start-code is the bit string 000001B2 in hexadecimal. It identifies the beginning of a group of pictures. MPEG-1, at 54: A GOP is a random access point, i.e. it is the smallest coding unit that can be independently decoded within a sequence, and consists of a header and some number of pictures. The GOP header contains time and editing information. MPEG-1, at 54: <table><tr><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td></tr><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td></tr></table> Figure D.3 -- Typical sequence of pictures in display order MPEG-1, AT 54: <table><tr><td>I</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td></tr><tr><td>0</td><td>3</td><td>1</td><td>2</td><td>6</td><td>4</td><td>5</td><td>9</td><td>7</td><td>8</td><td>12</td><td>10</td><td>11</td><td>15</td><td>13</td><td>14</td><td>18</td><td>16</td><td>17</td></tr></table> Figure D.4 -- Typical sequence of pictures in bitstream order	I	B	B	P	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	I	P	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B	0	3	1	2	6	4	5	9	7	8	12	10	11	15	13	14	18	16	17
I	B	B	P	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P																																																											
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18																																																											
I	P	B	B	P	B	B	P	B	B	I	B	B	P	B	B	P	B	B																																																											
0	3	1	2	6	4	5	9	7	8	12	10	11	15	13	14	18	16	17																																																											

	MPEG-1, at 56-57: It is instructive to follow the method which the decoder uses to decode a bitstream containing the sequence of pictures given in Fig D.4, and display them in the order given in Fig D.3. The following description is simplified for clarity. The input bitstream is accumulated in the Input Buffer until needed. The Variable Length Code (VLC) Decoder decodes the header of the first picture, picture 0, and determines that it is an I-picture. The VLC Decoder produces quantized coefficients corresponding to the quantized DCT coefficients. These are assembled for each 8 by 8 block of pels in the image by inverse zig-zag scanning. The Inverse Quantizer produces the actual DCT coefficients using the quantization step size. The coefficients are then transformed into pel values by the Inverse DCT transformer and stored in the Previous Picture Store and the Display Buffer. The picture may be displayed at the appropriate time. The VLC Decoder decodes the header of the next picture, picture 3, and determines that it is a P-picture. For each block, the VLC Decoder decodes motion vectors giving the displacement from the stored previous picture, and quantized coefficients corresponding to the quantized DCT coefficients of the difference block. These quantized coefficients are inverse quantized to produce the actual DCT coefficients. The coefficients are then transformed into pel difference values and added to the predicted block produced by applying the motion vectors to blocks in the stored previous picture. The resultant block is stored in the Future Picture Store and the Display Buffer. This picture cannot be displayed until B-pictures 1 and 2 have been received, decoded, and displayed. The VLC Decoder decodes the header of the next picture, picture 1, and determines that it is a B-picture. For each block, the VLC decoder decodes motion vectors giving the displacement from the stored previous or future pictures or both, and quantized coefficients corresponding to the quantized DCT coefficients of the difference block. These quantized coefficients are inverse quantized to produce the actual DCT coefficients. The coefficients are then inverse transformed into difference pel values and added to the predicted block produced by applying the motion vectors to the stored pictures. The resultant block is then stored in the Display Buffer. It may be displayed at the appropriate time. The VLC Decoder decodes the header of the next picture, picture 2, and determines that it is a B-picture. It is decoded using the same method as for picture 1. After decoding picture 2, picture 0, which is in the Previous Picture Store, is no longer needed and may be discarded.
--	---

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	The VLC Decoder decodes the header of the next picture, picture 6, and determines that it is a P-picture. The picture in the Future Picture Store is copied into the Previous Picture Store, then decoding proceeds as for picture 3. Picture 6 should not be displayed until pictures 4 and 5 have been received and displayed. The VLC Decoder decodes the header of the next picture, picture 4, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 5, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 9, and determines that it is a P-picture. It then proceeds as for picture 6. The VLC Decoder decodes the header of the next picture, picture 7, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 8, and determines that it is a B-picture. It is decoded using the same method as for picture 1. The VLC Decoder decodes the header of the next picture, picture 12, and determines that it is an I-picture. It is decoded using the same method as for picture 0. This process is repeated for the subsequent pictures. MPEG-1, at 62-63: A video sequence commences with a sequence header and is followed by one or more groups of pictures and is ended by a sequence end code. Additional sequence headers may appear within the sequence. In each such repeated sequence header, all of the data elements with the permitted exception of those defining quantization matrices (load intra quantizer matrix, load nonintra quantizer matrix and optionally inter quantizer matrix and nonintra quantizer matrix) shall have the same values as the first sequence header. Repeating the sequence header with its data elements makes random access into the video sequence possible. The quantization matrices may be redefined as required with each repeated sequence header.

	MPEG-1, at 67-68: Two distinct picture orderings exist, the display order and the bitstream order (as they appear in the video bitstream). A group of pictures (gop) is a set of pictures which are contiguous in display order. A group of pictures must contain at least one I-picture. This required picture may be followed by any number of I and P-pictures. Any number of B-pictures may be interspersed between each pair of I or P-pictures, and may also precede the first I-picture. Property 1. A group of pictures, in bitstream order, must start with an I-picture and may be followed by any number of I, P or B-pictures in any order. Property 2. Another property of a group of pictures is that it must begin, in display order, with an I or a B-picture, and must end with an I or a P-picture. The smallest group of pictures consists of a single I-picture, whereas the largest size is unlimited. The original concept of a group of pictures was a set of pictures that could be coded and displayed independently of any other group. In the final version of this part of ISO/IEC 11172 this is not always true, and any B-pictures preceding (in display order) the first I-picture in a group may require the last picture in the previous group in order to be decoded. Nevertheless encoders can still construct groups of pictures which are independent of one another. One way to do this is to omit any B-pictures preceding the first I-picture. Another way is to allow such B-pictures, but to code them using only backward motion compensation. Property 3. From a coding point of view, a concisely stated property is that a group of pictures begins with a group of pictures header, and either ends at the next group of pictures header or at the next sequence header or at the end of sequence, whichever comes first. MPEG-1, at 68:
--	--

Some examples of groups of pictures are given below:

```

      I
      I  P  P
      I  B  P  B  P
      B  B  I  B  P  B  P
      B  B  I  B  B  P  B  B  P  B  B  P
      B  I  B  B  B  B  P  B  I  B  B  I  I
  
```

Figure D.17 -- Examples of groups of pictures in display order

MPEG-1, at 69:

Closed GOP

A one bit flag follows the time code. It denotes whether the group of pictures is open or closed. Closed groups can be decoded without using decoded pictures of the previous group for motion compensation, whereas open groups require such pictures to be available.

MPEG-1, at 103-04:

0.7.1.1 Decoding for forward playback

At the beginning of a sequence, the decoder will decode the sequence header including the sequence parameters. If a parameter exceeds the capability of the decoder, then the decoder should report this. If the decoder determines that it can decode the bitstream, then it will set up its parameters to match those defined in the sequence header. This will include the horizontal and vertical resolutions and aspect ratio, the bit rate, and the quantization matrices.

Next the decoder will decode the group of pictures header, including the closed gop and broken link information, and take any appropriate action. It will decode the first picture header in the group of pictures and read the vbv delay field. If the decoder uses the vbv delay information to start-up decoding rather than the information in the system stream (ISO/IEC 11172- 1) then it must delay displaying pictures until after a time determined by the vbv delay information and a knowledge of the decoder's architecture.

If the closed gop flag is 0, indicating that the group is open, and the broken-link flag is 1, then any B-pictures preceding (in display order) the first I-picture in the group cannot be decoded. The decoder may adopt one of several strategies. It may display the first I-picture during the time that the undecodable B-pictures would be displayed. This strategy maintains audio synchronization and buffer fullness. However it is likely that the broken

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
8[c] decoding identifier values for image frames according to a numbering scheme; and	link has occurred because of post coding editing, in which case audio may be discontinuous. An alternative strategy might be to discard the B-pictures entirely, and delay decoding the I-picture until the buffer fullness is within limits. If playback begins from a random point in the bitstream, the decoder should discard all the bits until it finds a sequence start code, a group of pictures start code, or a picture start code which introduces an I-picture. The slices and macroblocks in the picture are decoded and written into a display buffer, and perhaps into another buffer. The decoded pictures may be post processed and displayed in the order defined by the temporal reference at the picture rate defined in the sequence header. Subsequent pictures are processed at the appropriate times to avoid buffer overflow and underflow. <i>Further, to the extent Complainant contends this element is not expressly or inherently disclosed by MPEG-1, it would have been obvious in view of the disclosures in MPEG-1, in combination with the knowledge of a person of ordinary skill in the art.</i> MPEG-1, as evidenced by the example citations below, discloses decoding identifier values for image frames according to a numbering scheme. <i>See, e.g.,</i> MPEG-1, at 27: 2.4.3.4 Picture Layer picture start code -- The picture start code is a string of 32-bits having the value 00000100 in hexadecimal. temporal reference -- The temporal reference is a 10-bit unsigned integer associated with each input picture. It is incremented by one, modulo 1024, for each input picture. For the earliest picture (in display order) in each group of pictures, the temporal reference is reset to zero. MPEG-1, at 70:

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																																																																																																																				
	The Temporal Reference is a ten-bit number which can be used to define the order in which the pictures must be displayed. It may be useful since pictures are not transmitted in display order, but rather in the order which the decoder needs to decode them. The first picture, in display order, in each group must have Temporal Reference equal to zero. This is incremented by one for each picture in the group. MPEG-1, at 79: Some example groups of pictures with their Temporal Reference numbers are given below: <table><tr><td>Example (a) in display order</td><td>I</td><td>B</td><td>P</td><td>B</td><td>P</td></tr><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>Example (a) in decoding order</td><td>I</td><td>P</td><td>B</td><td>P</td><td>B</td></tr><tr><td></td><td>0</td><td>2</td><td>1</td><td>4</td><td>3</td></tr><tr><td>Example (b) in display order</td><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td></tr><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td></tr><tr><td>Example (b) in coded order</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td></tr><tr><td></td><td>2</td><td>0</td><td>1</td><td>5</td><td>3</td><td>4</td><td>8</td><td>6</td><td>7</td><td>11</td><td>9</td><td>10</td></tr><tr><td>Example (c) in display order</td><td>B</td><td>I</td><td>B</td><td>B</td><td>B</td><td>B</td><td>P</td><td>B</td><td>I</td><td>B</td><td>B</td><td>I</td><td>I</td></tr><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td></tr><tr><td>Example (c) in coded order</td><td>I</td><td>B</td><td>P</td><td>B</td><td>B</td><td>B</td><td>B</td><td>I</td><td>B</td><td>I</td><td>B</td><td>B</td><td>I</td></tr><tr><td></td><td>1</td><td>0</td><td>6</td><td>2</td><td>3</td><td>4</td><td>5</td><td>8</td><td>7</td><td>11</td><td>9</td><td>10</td><td>12</td></tr></table> Figure D.19 -- Examples of groups of pictures and temporal references MPEG-1, at 103: Next the decoder will decode the group of pictures header, including the closed gop and broken-link information, and take any appropriate action. It will decode the first picture header in the group of pictures and read the vbv-delay field. If the decoder uses the vbv-delay information to start-up decoding rather than the information in the system stream (ISO/IEC 11172- 1) then it must delay displaying pictures until after a time determined by the vbvdelay information and a knowledge of the decoder's architecture. If playback begins from a random point in the bitstream, the decoder should discard all the bits until it finds a sequence start code, a group of pictures start code, or a picture start code which introduces an I-picture. The slices and macroblocks in the picture are decoded and written into a display buffer, and perhaps into another buffer. The	Example (a) in display order	I	B	P	B	P		0	1	2	3	4	Example (a) in decoding order	I	P	B	P	B		0	2	1	4	3	Example (b) in display order	B	B	I	B	B	P	B	B	P	B	B	P		0	1	2	3	4	5	6	7	8	9	10	11	Example (b) in coded order	I	B	B	P	B	B	P	B	B	P	B	B		2	0	1	5	3	4	8	6	7	11	9	10	Example (c) in display order	B	I	B	B	B	B	P	B	I	B	B	I	I		0	1	2	3	4	5	6	7	8	9	10	11	12	Example (c) in coded order	I	B	P	B	B	B	B	I	B	I	B	B	I		1	0	6	2	3	4	5	8	7	11	9	10	12
Example (a) in display order	I	B	P	B	P																																																																																																																																
	0	1	2	3	4																																																																																																																																
Example (a) in decoding order	I	P	B	P	B																																																																																																																																
	0	2	1	4	3																																																																																																																																
Example (b) in display order	B	B	I	B	B	P	B	B	P	B	B	P																																																																																																																									
	0	1	2	3	4	5	6	7	8	9	10	11																																																																																																																									
Example (b) in coded order	I	B	B	P	B	B	P	B	B	P	B	B																																																																																																																									
	2	0	1	5	3	4	8	6	7	11	9	10																																																																																																																									
Example (c) in display order	B	I	B	B	B	B	P	B	I	B	B	I	I																																																																																																																								
	0	1	2	3	4	5	6	7	8	9	10	11	12																																																																																																																								
Example (c) in coded order	I	B	P	B	B	B	B	I	B	I	B	B	I																																																																																																																								
	1	0	6	2	3	4	5	8	7	11	9	10	12																																																																																																																								

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
8[d] resetting the identifier value for the indicated first image frame of the independent sequence.	decoded pictures may be post processed and displayed in the order defined by the temporal reference at the picture rate defined in the sequence header. <i>Further, to the extent Complainant contends this element is not expressly or inherently disclosed by MPEG-1, it would have been obvious in view of the disclosures in MPEG-1, in combination with the knowledge of a person of ordinary skill in the art.</i> MPEG-1, at 27: 2.4.3.4 Picture Layer picture start code -- The picture start code is a string of 32-bits having the value 00000100 in hexadecimal. temporal reference -- The temporal reference is a 10-bit unsigned integer associated with each input picture. It is incremented by one, modulo 1024, for each input picture. For the earliest picture (in display order) in each group of pictures, the temporal reference is reset to zero. MPEG-1, at 70: The Temporal Reference is a ten-bit number which can be used to define the order in which the pictures must be displayed. It may be useful since pictures are not transmitted in display order, but rather in the order which the decoder needs to decode them. The first picture, in display order, in each group must have Temporal Reference equal to zero. This is incremented by one for each picture in the group.

MPEG-1, at 70:

Some example groups of pictures with their Temporal Reference numbers are given below:

Example (a) in display order	I	B	P	B	P								
	0	1	2	3	4								
Example (a) in decoding order	I	P	B	P	B								
	0	2	1	4	3								
Example (b) in display order	B	B	I	B	B	P	B	B	P	B	B	P	
	0	1	2	3	4	5	6	7	8	9	10	11	
Example (b) in coded order	I	B	B	P	B	B	P	B	B	P	B	B	
	2	0	1	5	3	4	8	6	7	11	9	10	
Example (c) in display order	B	I	B	B	B	B	P	B	I	B	B	I	I
	0	1	2	3	4	5	6	7	8	9	10	11	12
Example (c) in coded order	I	B	P	B	B	B	B	I	B	I	B	B	I
	1	0	6	2	3	4	5	8	7	11	9	10	12

Figure D.19 -- Examples of groups of pictures and temporal references

As MPEG-1 explains, a closed GOP must always start with an I-picture in bitstream order:

Property 1. A group of pictures, in bitstream order, must start with an I-picture and may be followed by any number of I, P or B-pictures in any order.

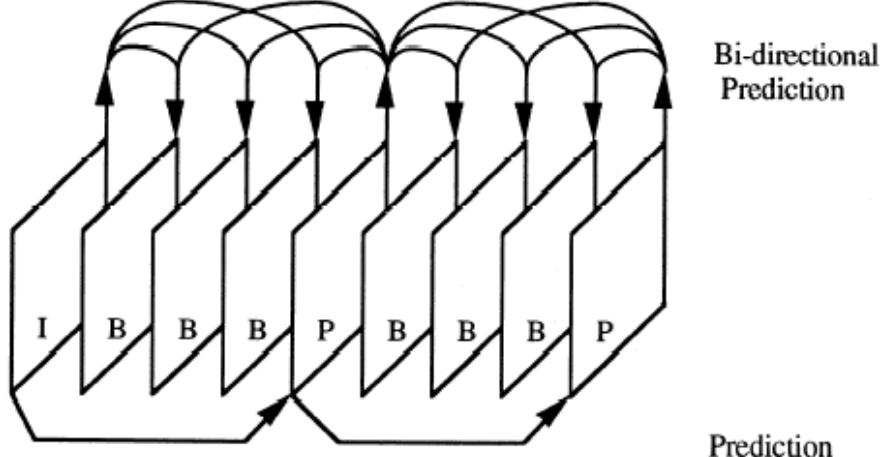
MPEG-1 at D.5.2 (p. 67)

However, in display order, a closed GOP can begin with either an I-picture or a B-picture:

Property 2. Another property of a group of pictures is that it must begin, in display order, with an I or a B-picture, and must end with an I or a P-picture. The smallest group of pictures consists of a single I-picture, whereas the largest size is unlimited.

MPEG-1 at D.5.2 (p. 67)

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																																								
	MPEG-1 provides two exemplary closed GOPs in display order: group (a) which begins with an I-picture and is referred to as a “typical example,” and group (b) which begins with leading B-pictures and is referred to as “less typical”: A typical example of a closed group is shown in figure D.18a. <table><tr><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td></tr><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td></tr></table> (a) closed group <table><tr><td>B</td><td>B</td><td>I</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td><td>B</td><td>B</td><td>P</td></tr><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td></tr></table> (b) open or closed group Figure D.18 -- Example groups of pictures in display order A less typical example of a closed group is shown in figure D.18b. In this example, the B-pictures which precede the first I-picture must use backward motion compensation only, i.e. any motion compensation must be based only on picture number 2 in the group. MPEG-1 at D.5.2 (p. 69) The decoding order or bitstream order of the closed GOP is “the order which the decoder needs [the pictures] to decode them.” MPEG-1 at D.5.3.2. The decoding order is different from the display order due to the inclusion of B-pictures—which can use both past and future reference pictures for motion prediction:	I	B	B	P	B	B	P	B	B	P	B	B	P	0	1	2	3	4	5	6	7	8	9	10	11	12	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
I	B	B	P	B	B	P	B	B	P	B	B	P																																													
0	1	2	3	4	5	6	7	8	9	10	11	12																																													
B	B	I	B	B	P	B	B	P	B	B	P	B	B	P																																											
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14																																											

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	Because of the conflicting requirements of random access and highly efficient compression, three main picture types are defined. Intra-coded pictures (I-Pictures) are coded without reference to other pictures. They provide access points to the coded sequence where decoding can begin, but are coded with only a moderate compression ratio. Predictive coded pictures (P-Pictures) are coded more efficiently using motion compensated prediction from a past intra or predictive coded picture and are generally used as a reference for further prediction. Bidirectionally-predictive coded pictures (B-Pictures) provide the highest degree of compression but require both past and future reference pictures for motion compensation. Bidirectionally-predictive coded pictures are never used as references for prediction. The organisation of the three picture types in a sequence is very flexible. The choice is left to the encoder and will depend on the requirements of the application. Figure 1 illustrates the relationship between the three different picture types.  Figure 1 -- Example of temporal picture structure MPEG-1 at 0.2.1 (p. v). In other words, the sequence of only I-pictures and P-pictures is the same in both the display order and the decoding order. However, B-pictures must be decoded after the future P-picture or I-picture that is used for prediction, so the display order is different than the decoding order. In order to determine the display order, the decoder decodes a value called the “temporal reference” from the bitstream for each picture:

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	The Temporal Reference is a ten-bit number which can be used to define the order in which the pictures must be displayed. It may be useful since pictures are not transmitted in display order, but rather in the order which the decoder needs to decode them. The first picture, in display order, in each group must have Temporal Reference equal to zero. This is incremented by one for each picture in the group. MPEG-1 at D.5.3.2 (p. 70). It would have been obvious to a POSITA to modify MPEG-1, and specifically the structure of the closed GOP, to allow closed GOPs to begin only with an I-picture in display order. In other words, it would have been obvious to require closed GOPs to follow the general structure of D.18(a) and not permit closed GOPs to follow the general structure of D.18(b) (i.e., GOPs beginning with B-pictures in display order). In fact, a POSITA would have been motivated to make such a modification. A POSITA would have known that closed GOPs of the general structure of D.18(b) were rarely, if ever, utilized. A POSITA would have also known that closed GOPs of the general structure of D.18(b) were computationally more difficult to create and were impractical. As such, a POSITA would have been motivated to limit closed GOPs to the general structure of D.18(a) (i.e., GOP's beginning with I-pictures in display order). A POSITA would have also had a reasonable expectation of success in implementing such a modification. In fact, such closed GOPs were already described as the "typical example" of closed GOPs in the MPEG-1 standard. A POSITA would have had a reasonable expectation that using only the "typical example" of closed GOPs would be successful and would not pose any problems or challenges. The obvious modified version of MPEG-1 described above, would mean that for every closed GOP, the first frame after the closed_GOP_flag = 0 would be an I-frame that is the first frame in decoding order and in display order (having a temporal reference of zero). As such, in a decoder using the obvious modified version of MPEG-1 described above, it would have been obvious to a POSITA to implement that the decoder would perform a hard-reset of the temporal reference number for the first frame in decoding order following the closed_GOP_flag = 0. A POSITA would have found two such implementations obvious: First, it would have been obvious to implement a system in which the decoder would (1) decode the closed_GOP_flag = 0, (2) decode the temporal reference for the first frame (an I-picture) following the

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	closed_GOP_flag=0, and (3) reset the temporal reference of the first frame (an I-picture) to zero. The proposed obvious code in the decoder would have a structure similar to the example below: <pre> Decode GOP Start Code Picture Count = 0 Decode GOP Header Decode closed_GOP_flag While next_start_code = Picture Start Code Picture Count = Picture Count + 1 Decode Picture Header Decode temporal_reference If Picture Count == 1 temporal_reference = 0 </pre> A POSITA would have been motivated to use this implementation in order to include a built-in redundancy into the decoder as part of an error prevention system. Such an implementation would mean that, even if the bitstream had been damaged such that the temporal_reference of the first frame in display order of the closed GOP was not accurate, the decoder would be able to correct the inaccuracy and correctly display the video sequence. A POSITA would also have had a reasonable expectation that such modification would have been successful. Such modification is a simple and ordinary modification to the code of the decoder, which would not have been difficult to implement. Second, it would have been obvious to implement a system in which the decoder would (1) decode the closed_GOP_flag = 0, (2) reset the temporal reference of the first frame (an I-picture) to zero, and (3) only decode the temporal reference for the frames following the first frame in decoding order. The proposed obvious code in the decoder would have a structure similar to the example below: <pre> Decode GOP Start Code Picture Count = 0 Decode GOP Header </pre>

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
10[preamble] A video decoder comprising at least one processor and at least one memory including computer program code, the at least one	Decode closed_GOP_flag While next_start_code = Picture Start Code Picture Count = Picture Count + 1 Decode Picture Header If Picture Count == 1 temporal_reference = 0 Else Decode temporal_reference A POSITA would have been motivated to use this implementation in order to implement efficiency into the decoder and to avoid the redundancy of decoding the temporal reference for the first frame. Such an implementation would mean that a decoder would save the processing power and speed needed to decode the temporal reference of the first frame following the closed GOP flag and could automatically set the value of the temporal reference to zero. A POSITA would also have had a reasonable expectation that such modification would have been successful. Such modification is a simple and ordinary modification to the code of the decoder, which would not have been difficult to implement. <i>Further, to the extent Complainant contends this element is not expressly or inherently disclosed by MPEG-1, it would have been obvious in view of the disclosures in MPEG-1, in combination with the knowledge of a person of ordinary skill in the art..</i> MPEG-1, as evidenced by the example citations below, discloses a video decoder comprising at least one processor and at least one memory including computer program code, the at least one memory and the computer program code configured to, with the at least one processor, cause the video decoder to decode a compressed video sequence. <i>See, e.g.,</i>

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
memory and the computer program code configured to, with the at least one processor, cause the video decoder to decode a compressed video sequence, by causing the video decoder to at least:	<i>See limitation 8[preamble]</i> MPEG-1, at 5: 2.144 decoder: An embodiment of a decoding process. 2.1.45 decoding (process): The process defined in ISO/IEC 11172 that reads an input coded bitstream and produces decoded pictures or audio samples. MPEG-1, at 10. 2.1.152 video buffering verifier; VBV [video]: A hypothetical decoder that is conceptually connected to the output of the encoder. Its purpose is to provide a constraint on the variability of the data rate that an encoder or editing process may produce MPEG-1, at 62: D.4.2 Decoder model Annex C contains the definition of a parameterized model decoder for this purpose. It is known as a Video Buffer Verifier (VBV). The parameters used by a particular encoder are defined in the bitstream. This really defines a model decoder that is needed if encoders are to be assured that the coded bitstreams they produce will be decodable. The model decoder looks like this:

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	<pre> graph LR CB[Coded Bitstream] --> MB[Model Decoder] subgraph MB [Model Decoder] IB[Input Buffer] --> PD[Picture Decoder] end PD --> DP[Decoded Pictures] </pre> Figure D.14 -- Model decoder A fixed-rate channel is assumed to put bits at a constant rate into the Input Buffer. At regular intervals, set by the picture rate, the Picture Decoder instantaneously removes all the bits for the next picture from the Input Buffer. If there are too few bits in the Input Buffer, i.e. all the bits for the next picture have not been received, then the Input Buffer underflows and there is an underflow error. If, during me time between picture starts, the capacity of the Input Buffer is exceeded, then there is an overflow error. Practical decoders differ from this model in several important ways. They may implement their buffering at a different point in the decoder, or distribute it throughout the decoder. They may not remove all the bits required to decode a picture from the Input Buffer instantaneously, they may not be able to control the start of decoding very precisely as required by the buffer fullness parameter in the picture header, and they take a finite time to decode. They may also be able to delay decoding for a short time to reduce the chances of underflow occurring. But these differences depend in degree and kind on the exact method of implementation. To satisfy requirements of different implementations, the MPEG video committee (ISO/IEC JTC1 SC29/WG11) chose a very simple model for the decoder. Practical implementations of decoders must ensure that they can decode the bitstream constrained by this model. In many cases this will be achieved by using an Input Buffer that is larger than the minimum required, and by using a decoding delay that is larger than the value derived from the vbv delay parameter. The designer must compensate for differences between the actual design and the model in order to guarantee that the decoder can handle any bitstream that satisfies the model. MPEG-1, at 65:

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	The buffer size is a 10-bit integer giving the minimum required size of the input buffer in the model decoder in units of 16 384 bits (2 048 bytes). For example, a buffer size of 20 would require an input buffer of 20 x 16 384 = 327 680 bits (= 40 960 bytes). Decoders may provide more memory than this, but if they provide less they will probably run into buffer overflow problems while the sequence is being decoded. MPEG-1 at ix: 0.6.3 Reverse playback Some applications may require the video signal to be played in reverse order. This can be achieved in a decoder by using memory to store entire groups of pictures after they have been decoded before being displayed in reverse order. An encoder can make this feature easier by reducing the length of groups of pictures. MPEG-1 at 104: To reduce decoder memory requirements, groups of pictures should be small. There is no mechanism in the syntax for the encoder to state what the decoder requirements are in order to playback in reverse. The amount of display buffer storage may be reduced by reordering the pictures, either by having the storage unit read and transmit them in another order, or by reordering the coded pictures in a decoder buffer. To MPEG-1 at 1:

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
	Section 1: General 1.1 Scope This part of ISO/IEC 11172 specifies the coded representation of video for digital storage media and specifies the decoding process. The representation supports normal speed forward playback, as well as special functions such as random access, fast forward playback, fast reverse playback, normal speed reverse playback, pause and still pictures. This part of ISO/IEC 11172 is compatible with standard 525- and 625-line television formats, and it provides flexibility for use with personal computer and workstation displays. ISO/IEC 11172 is primarily applicable to digital storage media supporting a continuous transfer rate up to about 1,5 Mbit/s, such as Compact Disc, Digital Audio Tape, and magnetic hard disks. Nevertheless it can be used more widely than this because of the generic approach taken. The storage media may be directly connected to the decoder, or via communications means such as busses, LANs, or telecommunications links. This part of ISO/IEC 11172 is intended for non-interlaced video formats having approximately 288 lines of 352 pels and picture rates around 24 Hz to 30 Hz. MPEG-1 at 11: 2.2 Symbols and abbreviations The mathematical operators used to describe this International Standard are similar to those used in the C programming language. However, integer division with truncation and rounding are specifically defined. The bitwise operators are defined assuming twos-complement representation of integers. Numbering and counting loops generally begin from zero.

MPEG- 1 , at 18 :

2.4.2.1 Start codes

Start codes are reserved bit patterns that do not otherwise occur in the video stream. All start codes are bytealigned.

Name	Hexadecimal value
picture_start_code	00000100
slice_start_codes (including slice_vertical_positions)	00000101
	through
	000001AF
reserved	000001B0
reserved	000001B1
user_data_start_code	000001B2
sequence_header_code	000001B3
sequence_error_code	000001B4
extension_start_code	000001B5
reserved	000001B6
sequence_end_code	000001B7
group_start_code	000001B8
system start codes (see note)	000001B9
	through
	000001FF
NOTE - System start codes are defined in ISO/IEC 11172-1.	

The use of the start codes is defined in the following syntax description with the exception of the sequence_error_code. The sequence_error_code has been allocated for use by the digital storage media interface to indicate where uncorrectable errors have been detected.

MPEG-1, AT 18:

2.4.2.2 Video sequence layer

Syntax	No. of bits	Mnemonic
video_sequences() { next_start_code() do { sequence_header() do { group_of_pictures() } while (nextbits() == group_start_code) } while (nextbits() == sequence_header_code) sequence_end_code }	32	bs1bf

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																																																																																			
	MPEG-1, at 19: 2.4.2.3 Sequence header <table><tr><th>Syntax</th><th>No. of bits</th><th>Mnemonic</th></tr><tr><td>sequence_header() {</td><td></td><td></td></tr><tr><td> sequence_header_code</td><td>32</td><td>bslbf</td></tr><tr><td> horizontal_size</td><td>12</td><td>uimsbf</td></tr><tr><td> vertical_size</td><td>12</td><td>uimsbf</td></tr><tr><td> pel_aspect_ratio</td><td>4</td><td>uimsbf</td></tr><tr><td> picture_rate</td><td>4</td><td>uimsbf</td></tr><tr><td> bit_rate</td><td>18</td><td>uimsbf</td></tr><tr><td> marker_bit</td><td>1</td><td>"1"</td></tr><tr><td> vby_buffer_size</td><td>10</td><td>uimsbf</td></tr><tr><td> constrained_parameters_flag</td><td>1</td><td></td></tr><tr><td> load_intra_quantizer_matrix</td><td>1</td><td></td></tr><tr><td> if (load_intra_quantizer_matrix)</td><td></td><td></td></tr><tr><td> intra_quantizer_matrix []</td><td>8*64</td><td>uimsbf</td></tr><tr><td> load_non_intra_quantizer_matrix</td><td>1</td><td></td></tr><tr><td> if (load_non_intra_quantizer_matrix)</td><td></td><td></td></tr><tr><td> non_intra_quantizer_matrix []</td><td>8*64</td><td>uimsbf</td></tr><tr><td> next_start_code()</td><td></td><td></td></tr><tr><td> if (nextbits() == extension_start_code) {</td><td></td><td></td></tr><tr><td> extension_start_code</td><td>32</td><td>bslbf</td></tr><tr><td> while (nextbits() != '0000 0000 0000 0000 0000 0001') {</td><td></td><td></td></tr><tr><td> sequence_extension_data</td><td>8</td><td></td></tr><tr><td> }</td><td></td><td></td></tr><tr><td> next_start_code()</td><td></td><td></td></tr><tr><td> }</td><td></td><td></td></tr><tr><td> if (nextbits() == user_data_start_code) {</td><td></td><td></td></tr><tr><td> user_data_start_code</td><td>32</td><td>bslbf</td></tr><tr><td> while (nextbits() != '0000 0000 0000 0000 0000 0001') {</td><td></td><td></td></tr><tr><td> user_data</td><td>8</td><td></td></tr><tr><td> }</td><td></td><td></td></tr><tr><td> next_start_code()</td><td></td><td></td></tr><tr><td> }</td><td></td><td></td></tr><tr><td>}</td><td></td><td></td></tr></table>	Syntax	No. of bits	Mnemonic	sequence_header() {			sequence_header_code	32	bslbf	horizontal_size	12	uimsbf	vertical_size	12	uimsbf	pel_aspect_ratio	4	uimsbf	picture_rate	4	uimsbf	bit_rate	18	uimsbf	marker_bit	1	"1"	vby_buffer_size	10	uimsbf	constrained_parameters_flag	1		load_intra_quantizer_matrix	1		if (load_intra_quantizer_matrix)			intra_quantizer_matrix []	8*64	uimsbf	load_non_intra_quantizer_matrix	1		if (load_non_intra_quantizer_matrix)			non_intra_quantizer_matrix []	8*64	uimsbf	next_start_code()			if (nextbits() == extension_start_code) {			extension_start_code	32	bslbf	while (nextbits() != '0000 0000 0000 0000 0000 0001') {			sequence_extension_data	8		}			next_start_code()			}			if (nextbits() == user_data_start_code) {			user_data_start_code	32	bslbf	while (nextbits() != '0000 0000 0000 0000 0000 0001') {			user_data	8		}			next_start_code()			}			}		
Syntax	No. of bits	Mnemonic																																																																																																		
sequence_header() {																																																																																																				
sequence_header_code	32	bslbf																																																																																																		
horizontal_size	12	uimsbf																																																																																																		
vertical_size	12	uimsbf																																																																																																		
pel_aspect_ratio	4	uimsbf																																																																																																		
picture_rate	4	uimsbf																																																																																																		
bit_rate	18	uimsbf																																																																																																		
marker_bit	1	"1"																																																																																																		
vby_buffer_size	10	uimsbf																																																																																																		
constrained_parameters_flag	1																																																																																																			
load_intra_quantizer_matrix	1																																																																																																			
if (load_intra_quantizer_matrix)																																																																																																				
intra_quantizer_matrix []	8*64	uimsbf																																																																																																		
load_non_intra_quantizer_matrix	1																																																																																																			
if (load_non_intra_quantizer_matrix)																																																																																																				
non_intra_quantizer_matrix []	8*64	uimsbf																																																																																																		
next_start_code()																																																																																																				
if (nextbits() == extension_start_code) {																																																																																																				
extension_start_code	32	bslbf																																																																																																		
while (nextbits() != '0000 0000 0000 0000 0000 0001') {																																																																																																				
sequence_extension_data	8																																																																																																			
}																																																																																																				
next_start_code()																																																																																																				
}																																																																																																				
if (nextbits() == user_data_start_code) {																																																																																																				
user_data_start_code	32	bslbf																																																																																																		
while (nextbits() != '0000 0000 0000 0000 0000 0001') {																																																																																																				
user_data	8																																																																																																			
}																																																																																																				
next_start_code()																																																																																																				
}																																																																																																				
}																																																																																																				

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1																																																																											
	MPEG-1, at 20: 2.4.2.4 Group of pictures layer <table><tr><th>Syntax</th><th>No. of bits</th><th>Mnemonic</th></tr><tr><td colspan="3">group_of_pictures() {</td></tr><tr><td>group_start_code</td><td>32</td><td>bslbf</td></tr><tr><td>time_code</td><td>25</td><td></td></tr><tr><td>closed_gop</td><td>1</td><td></td></tr><tr><td>broken_link</td><td>1</td><td></td></tr><tr><td>next_start_code()</td><td></td><td></td></tr><tr><td>if (nextbits() == extension_start_code) {</td><td></td><td></td></tr><tr><td>extension_start_code</td><td>32</td><td>bslbf</td></tr><tr><td>while (nextbits() != '0000 0000 0000 0000 0001') {</td><td></td><td></td></tr><tr><td>group_extension_data</td><td>8</td><td></td></tr><tr><td>}</td><td></td><td></td></tr><tr><td>next_start_code()</td><td></td><td></td></tr><tr><td>}</td><td></td><td></td></tr><tr><td>if (nextbits() == user_data_start_code) {</td><td></td><td></td></tr><tr><td>user_data_start_code</td><td>32</td><td>bslbf</td></tr><tr><td>while (nextbits() != '0000 0000 0000 0000 0001') {</td><td></td><td></td></tr><tr><td>user_data</td><td>8</td><td></td></tr><tr><td>}</td><td></td><td></td></tr><tr><td>next_start_code()</td><td></td><td></td></tr><tr><td>}</td><td></td><td></td></tr><tr><td>do {</td><td></td><td></td></tr><tr><td>picture()</td><td></td><td></td></tr><tr><td>} while (nextbits() != picture_start_code)</td><td></td><td></td></tr><tr><td>}</td><td></td><td></td></tr></table>	Syntax	No. of bits	Mnemonic	group_of_pictures() {			group_start_code	32	bslbf	time_code	25		closed_gop	1		broken_link	1		next_start_code()			if (nextbits() == extension_start_code) {			extension_start_code	32	bslbf	while (nextbits() != '0000 0000 0000 0000 0001') {			group_extension_data	8		}			next_start_code()			}			if (nextbits() == user_data_start_code) {			user_data_start_code	32	bslbf	while (nextbits() != '0000 0000 0000 0000 0001') {			user_data	8		}			next_start_code()			}			do {			picture()			} while (nextbits() != picture_start_code)			}		
Syntax	No. of bits	Mnemonic																																																																										
group_of_pictures() {																																																																												
group_start_code	32	bslbf																																																																										
time_code	25																																																																											
closed_gop	1																																																																											
broken_link	1																																																																											
next_start_code()																																																																												
if (nextbits() == extension_start_code) {																																																																												
extension_start_code	32	bslbf																																																																										
while (nextbits() != '0000 0000 0000 0000 0001') {																																																																												
group_extension_data	8																																																																											
}																																																																												
next_start_code()																																																																												
}																																																																												
if (nextbits() == user_data_start_code) {																																																																												
user_data_start_code	32	bslbf																																																																										
while (nextbits() != '0000 0000 0000 0000 0001') {																																																																												
user_data	8																																																																											
}																																																																												
next_start_code()																																																																												
}																																																																												
do {																																																																												
picture()																																																																												
} while (nextbits() != picture_start_code)																																																																												
}																																																																												

MPEG-1, at 21:

2.4.2.5 Picture layer

Syntax	No. of bits	Mnemonic
picture() {		
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if((picture_coding_type == 2) (picture_coding_type == 3)) {		
full_pel_forward_vector	1	
forward_f_code	3	uimsbf
}		
if(picture_coding_type == 3) {		
full_pel_backward_vector	1	
backward_f_code	3	uimsbf
}		
while(nextbits() == '1') {		
extra_bit_picture	1	"1"
extra_information_picture	8	
}		
extra_bit_picture	1	"0"
next_start_code()		
if(nextbits() == extension_start_code) {		
extension_start_code	32	bslbf
while(nextbits() != '0000 0000 0000 0000 0000 0001') {		
picture_extension_data	8	
}		
next_start_code()		
}		
if(nextbits() == user_data_start_code) {		
user_data_start_code	32	bslbf
while(nextbits() != '0000 0000 0000 0000 0000 0001') {		
user_data	8	
}		
next_start_code()		
}		
do {		
slice()		
} while(nextbits() == slice_start_code)		
}		

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
10[a] decode from the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of an independent sequence, wherein all motion-compensated temporal prediction references of the independent sequence refer only to image frames within said independent sequence; 10[b] start the decoding of the video sequence from said first image frame of the independent sequence, whereby the video sequence is decoded without prediction from any image frame decoded prior to said first image frame; and 10[c] decode identifier values for image frames according to a numbering scheme;	It would have been obvious to a POSITA that the decoder implementing the decoding process described in the MPEG-1 standard would do so using at least one processor and at least one memory including computer program code. These structures are common and known features of decoders and would necessarily be implemented by the decoder to perform the decoding process. <i>Further, to the extent Complainant contends this element is not expressly or inherently disclosed by MPEG-1, it would have been obvious in view of the disclosures in MPEG-1, in combination with the knowledge of a person of ordinary skill in the art.</i> MPEG-1, as evidenced by the example citations below, discloses decoding from the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of an independent sequence, wherein all motion-compensated temporal prediction references of the independent sequence refer only to image frames within the independent sequence. <i>See, e.g.,</i> <i>See limitation 8[a]</i> MPEG-1, as evidenced by the example citations below, discloses starting the decoding of the video sequence from the first image frame of the independent sequence, whereby the video sequence is decoded without prediction from any image frame decoded prior to the first image frame. <i>See, e.g.,</i> <i>See limitation 8[b]</i> MPEG-1, as evidenced by the example citations below, discloses decoding identifier values for image frames according to a numbering scheme.

ASSERTED CLAIMS	PRIOR ART REFERENCE: MPEG-1
10[d] reset the identifier value for the indicated first image frame of the independent sequence.	<i>See, e.g.,</i> <i>See limitation 8[c].</i> MPEG-1, as evidenced by the example citations below, discloses resetting the identifier value for the indicated first image frame of the independent sequence. <i>See, e.g.,</i> <i>See limitation 8[d]</i>