

EXHIBIT A-5

US Patent Publication No. 2005/0253740A1 (“Marpe ’740 Pub”)

ANTICIPATES / RENDERS OBVIOUS U.S. PATENT NO. 9,036,701

As described in the following claim chart, Asserted Claims 1, 3–4, 6–7, 8, 10–11, 13–14, 15, 17–18, and 20 of U.S. Patent No. 9,036,701 (the “’701 patent”) are invalid in view of U.S. Patent Publication No. 2005/0253740A1 (“Marpe ’740 Pub”), entitled Method and apparatus for binarization and arithmetic coding of a data value, published on November 17, 2005. Marpe ’740 Pub was filed on May 3, 2005, and is prior art under at least 35 U.S.C. §§ 102 (a), (b), and (e).

To the extent Complainants allege that Marpe ’740 Pub alone does not disclose any particular limitation of the Asserted Claims of the ’701 patent, either expressly or inherently, it would have been obvious to a person of ordinary skill in the art as of the priority date of ’701 patent to modify Marpe ’740 Pub and/or to combine the teachings of Marpe ’740 Pub with U.S. Patent Publication No. 2013/0028334 (“Bossen”), as set forth in the chart below.

To the extent Complainants allege that Marpe ’740 Pub alone or in combination with Bossen does not disclose any particular limitation of the Asserted Claims of the ’701 patent, either expressly or inherently, it would have been obvious to a person of ordinary skill in the art as of the priority date of ’701 patent to modify Marpe ’740 Pub and/or to combine the teachings of Marpe ’740 Pub with: (1) the knowledge of one of ordinary skill in the art to show all the limitations of the claim; (2) the teachings of the prior art references set forth in any of the references identified in Exhibit A-11.

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.

ASSERTED CLAIMS	PRIOR ART REFERENCE: Marpe '740 Pub
1. [Pre] A method comprising:	Marpe '740 Pub teaches and/or discloses a method comprising the following limitation: <i>See, e.g., Marpe '740 Pub at title (“Method and apparatus for binarization and arithmetic coding of a data value.”).</i>
[a] determining a frequency of occurrence threshold based on an expected frequency of occurrence of syntax elements in a bit stream;	Marpe '740 Pub teaches and/or discloses determining a frequency of occurrence threshold based on an expected frequency of occurrence of syntax elements in a bit stream: <i>See, e.g., Marpe '740 Pub at [0016]–[0017].</i> Furthermore, binary arithmetic coding enables context modeling on a sub-symbol level. For specific bins, i.e., the nodes in a binarization code tree of the binarization scheme by which the data value to be coded is binarized, conditional probabilities can be used. These specific bins will be, in general, the most frequently observed bins. Other, usually less frequently observed bins, can be treated using a joint, typically zero-order probability model. The use of the inventive binarization scheme enables, in connection with binary arithmetic coding, an effective way of adapting the binarization representation of the data value to the probability distribution of the data values. A suitably selected cut-off value may lead to the least probable data values to be binarized into code words having equally probable bit values, whereas the most probable data values may be binarized into code words leading to a very effective arithmetic coding bit stream when using adaptive context modeling. Thus, coding based on a static probability estimation may be employed for the second binarization scheme suffix of the code word, whereas

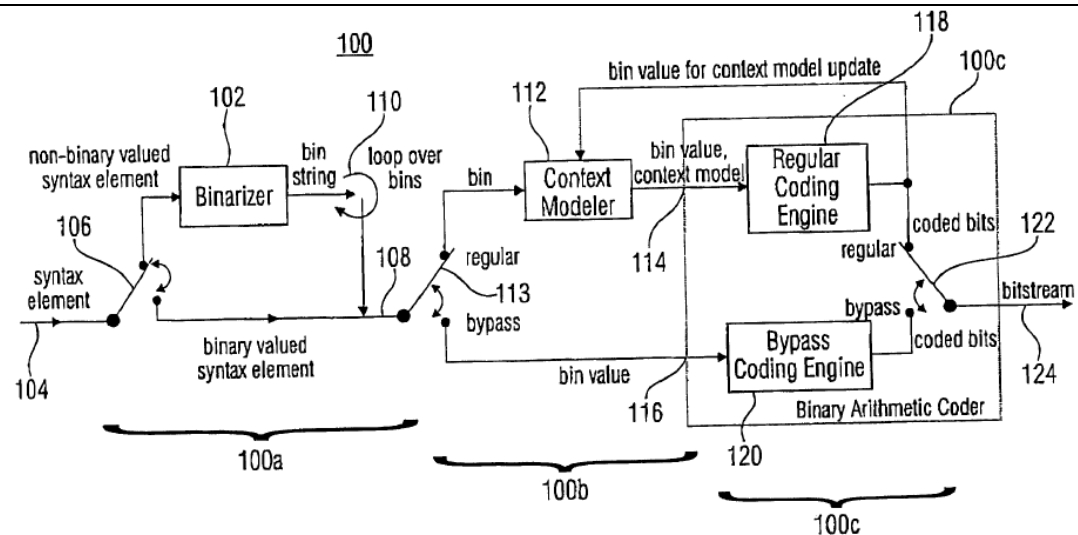
	adaptive binary arithmetic coding is effectively applied to bits of the first binarization scheme primary prefix of the code word. Marpe '740 Pub at [0076]. As can be seen from FIG. 8, the actual probability interval is getting narrower and narrower. Each bin is directed by switch 113 to context modeler 112. This changes, when the first bit of the primary suffix reaches stage 10 b. In this case, switch 113 passes the bin value to bypass coding engine 120. The bypass coding engine 120 operates on the same actual probability interval as regular coding engine 118, shown in FIG. 8 at 356. Contrary to the regular coding engine 118 the bypass coding engine 120 uses a fixed probability estimation or a static probability estimation. In particular, the fixed probability estimation used by bypass coding engine 120 assumes that the actual bin value is with equal probability a one or a zero, i.e. $P(0)=P(1)$. Thus, bypass coding engine 120 reduces the actual probability interval 356 to either the upper or lower half of the actual probability interval 356 when coding the actual bin of the primary suffix. Marpe '740 Pub at [0083]. If the bit is a bit of the primary suffix, in 404 the next bit or bin value is determined by performing arithmetic decoding based on a static probability model. Step 404 includes subdividing the actual probability interval according to the static probability estimation into two equally sized subintervals, assigning the bin value to be determined to the bit value associated with the subinterval in which the probability value indicated by the arithmetically coded bit stream lies, and reducing the actual probability interval to that subinterval. Marpe '740 Pub at [0123]–[0124]. Binary arithmetic coding is based on the principles of recursive interval subdivision that involves the following elementary multiplication operation. Suppose that an estimate of the probability $p_{LPS} \in (0, 0.5]$ of the least probable symbol (LPS) is given and that the given interval is represented by its lower bound L and its width (range) R.
--	--

	Based on that settings, the given interval is subdivided into two subintervals: one interval of width $R \text{ LPS} = R \times p \text{ LPS}$, which is associated with the LPS, and the dual interval of width $R \text{ MPS} = R - R \text{ LPS}$, which is assigned to the most probable symbol (MPS) having a probability estimate of $1 - p \text{ LPS}$. Depending on the observed binary decision, either identified as the LPS or the MPS, the corresponding sub-interval is then chosen as the new current interval. A binary value pointing into that interval represents the sequence of binary decisions processed so far, whereas the range of the interval corresponds to the product of the probabilities of those binary symbols. Thus, to unambiguously identify that interval and hence the coded sequence of binary decisions, the Shannon lower bound on the entropy of the sequence is asymptotically approximated by using the minimum precision of bits specifying the lower bound of the final interval. <i>Obviousness in view of Bossen:</i> To the extent that Complainants contend this element is not expressly or inherently disclosed by Marpe '740 Pub, this element would have been obvious in view of the combination of Marpe '740 Pub and Bossen. See Bossen claim chart served herewith. A POSITA would have been motivated to combine Marpe '740 Pub and Bossen at least for the reasons stated in the invalidity contentions cover pleading to which this claim chart is attached as an exhibit.
[b] categorizing a plurality of syntax elements of video content into first and second categories based on the frequency of occurrence threshold, wherein syntax elements which occur greater than the frequency of occurrence threshold are categorized into the second category and syntax elements	Marpe '740 Pub teaches and/or discloses categorizing a plurality of syntax elements of video content into first and second categories based on the frequency of occurrence threshold, wherein syntax elements which occur greater than the frequency of occurrence threshold are categorized into the second category and syntax elements which occur less than the frequency of occurrence are categorized into the first category: See limitation 1[a] above, which is incorporated herein by reference. See also, e.g., Marpe '740 Pub at [0048].

which occur less than the frequency of occurrence are categorized into the first category;¹	It is clear from the above, that the probability estimation determines the code and its efficiency in the first place, and that it is of paramount importance to have an adequate model that exploits the statistical dependencies of the syntax elements to a large degree and that this probability estimation or context model is kept up-to-date during encoding. However, there are significant model costs involved by additively estimating higher order conditional probabilities. Therefore, the binarization schemes of the embodiments described below are adapted to the statistics of the syntax elements such that context modeling leads to a efficient compression ratio even if context modeling is performed merely with respect to certain bits or bins of the binarized syntax elements at binarization stage output 108. With respect to the other bits, it is possible to use a static probability estimation and therefore reduce significantly the model costs, whereas the compression efficiency is affected merely to a moderate extent. Marpe '740 Pub at [0132]–[0133]. Both the third and fourth type of context models is applied to residual data only. In contrast to all other types of context models, both types depend on context categories of different block types. Moreover, the third type does not rely on past coded data, but on the position in the scanning path. For the fourth type, modeling functions are specified that involve the evaluation of the accumulated number of encoded (decoded) levels with a specific value prior to the current level bin to encode (decode). Besides these context models based on conditional probabilities, there are fixed assignments of probability models to bin indices for all those bins that have to be encoded in regular mode and to which no context model of the previous specified category can be applied. <i>Obviousness in view of Bossen:</i> To the extent that Complainants contend this element is not expressly or inherently disclosed by Marpe '740 Pub, this element would have been obvious in view of the
---	--

¹ For the convenience and clarity, the corrections from the certificate of correction have been incorporated into this limitation and the corresponding limitations of claims 8 and 15, below.

	combination of Marpe '740 Pub and Bossen. See Bossen claim chart served herewith. A POSITA would have been motivated to combine Marpe '740 Pub and Bossen at least for the reasons stated in the invalidity contentions cover pleading to which this claim chart is attached as an exhibit.
[c] entropy coding symbols that correspond to the first category of syntax elements and that have been subjected to a context update; and	Marpe '740 Pub teaches and/or discloses entropy coding symbols that correspond to the first category of syntax elements and that have been subjected to a context update: See, e.g., Marpe '740 Pub at Fig. 1. <pre> graph LR 10a[10a] --> 12[12 precoder] 12 -- "syntax elements" --> 14[14 entropy codes] 14 -- "arithmetically coded bitstream" --> Out[] </pre> The diagram illustrates a processing flow. It begins with a grid of 10a, which is part of a larger structure 10. An arrow points from 10a to a block labeled 12, which contains the word 'precoder'. From block 12, an arrow labeled 'syntax elements' points to a block labeled 14, which contains the words 'entropy codes'. Finally, an arrow points from block 14 to the right, labeled 'arithmetically coded bitstream'. FIG. 1 Marpe '740 Pub at Fig. 2.



Marpe '740 Pub at [0044].

Stage 100 b is a context modelling stage and comprises a context modeler 112 as well as a switch 110. The context modeler 112 comprises an input, an output and an optional feed-back input. The input of context modeler 112 is connected to the binarization stage output 108 via switch 110. The output of context modeler 112 is connected to a regular coding input terminal 114 of stage 100 c. The function of switch 113 is to pass the bits or bins of the bin string at binarization stage output 108 to either the context modeler 112 or to a bypass coding input terminal 116 of stage 100 c, thereby bypassing context modeler 112.

Marpe '740 Pub at [0046].

When receiving the bin string or codeword of a component of a motion vector difference or a transform coefficient level, it is the function of switch 113 to direct the bits of the codeword to either the context modeler 112 or the bypass input terminal 116 depending on to whether the bit belongs to a primary prefix part or a primary suffix part of the code word, as will be described in more detail below.

Marpe '740 Pub at [0049]–[0051].

The third stage 100 c of coding arrangement 100 is the arithmetic coding stage. It comprises a regular coding engine 118, a bypass coding engine 120, and a switch 122. The regular coding engine 118 comprises an input and an output terminal. The input terminal of regular coding engine 118 is connected to the regular coding input terminal 114. The regular coding engine 118 binary arithmetically codes the bin values passed from context modeler 112 by use of the context model also passed from context modeler 112 and outputs coded bits. Further, the regular coding engine 118 eventually passes bin values for context model updates to the feedback input of context modeler 112.

The bypass coding engine 112 has also an input and an output terminal, the input terminal being connected to the bypass coding input terminal 116. The bypass coding engine 120 is for binary arithmetically coding the bin values passed directly from binarization stage output 108 via switch 113 by use of a static predetermined probability distribution estimation and also outputs coded bits.

The coded bits output from regular coding engine 118 and bypass coding engine 120 are merged to a single bit stream at an output 124 of coding arrangement 100 by switch 122, the bit stream representing a binary arithmetic coded bit stream of the syntax elements as input in input terminal 104. Thus, regular coding engine 118 and bypass coding 120 cooperate in order to bit wise perform arithmetical coding based on either an adaptive or a static probability distribution model.

Marpe '740 Pub at [0074].

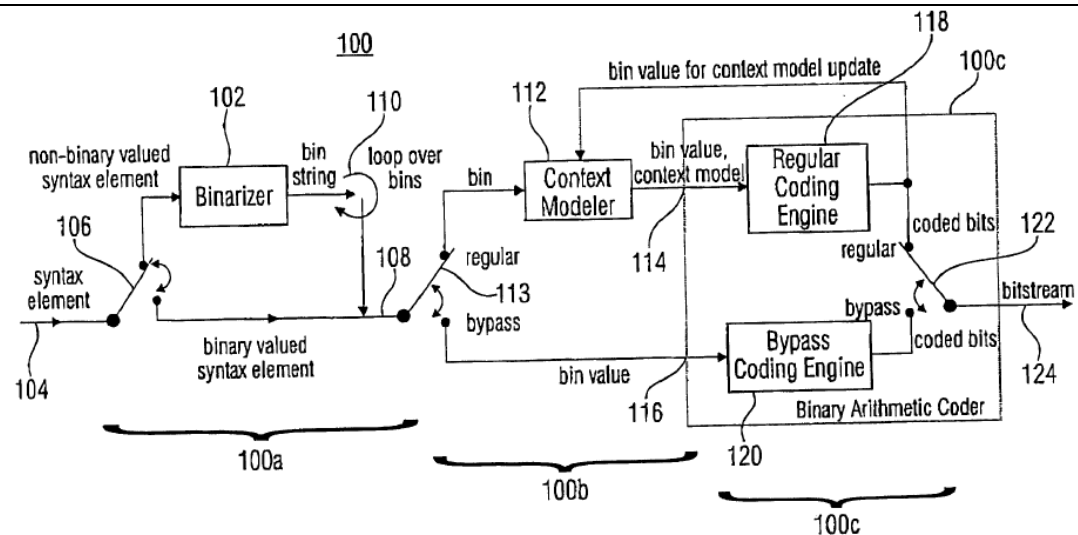
It is assumed that now the first bin of the syntax element binarized in accordance with the above-described binarization scheme is passed to the context modeling stage 100 b. Let us assume that the syntax element is an absolute value of a transform coefficient level. The first bin in the binarization scheme for transform coefficient levels is, as can be seen in FIG. 5, in the last line of the table, and in FIG. 3, which shows the

	binarization code tree for the binarization scheme of FIG. 5, the first bit of the TU prefix of the binary string of the transform coefficient level. This bin is regularly past by switch 113 to context modeler 112. The context modeler 112 adjusts an adaptive probability estimation 352, according to which the first bin of the bin string has a probability of $P(0)$ to be zero and a probability of $P(1)$ to be one, wherein $P(1)+P(0)=1$. Context modeler 112 needs just one parameter in order to store or manage the adaptive probability estimation. Context modeler 112 adjusts the adaptive probability estimation in accordance with past bins or prior bins and in accordance with the context model type associated with the respective bin. The context model types defining the dependencies of the probability estimation on the past bins may be different for the different bins of the bin string of a syntax element. The context modeler 112 passes bin value and adjusted adaptive probability estimation 352 to regular coding engine 118. Regular coding engine 118 reduces the actual probability interval 350 to a new probability interval having a range of $P(0)$ multiplied with the range of the actual probability interval 350, i.e. with a range of $P(0) \cdot (P_{high} - P_{low})$, if the bit value of the first bin of the primary prefix is zero, and $P(1)$ multiplied with the range of the actual probability interval 350, i.e. with a range of $P(1) \cdot (P_{high} - P_{low})$, if the first bit value of the first bin of the primary prefix is one. The new probability interval shares the lower limit with the actual probability interval 350, if the actual bin value of the first bin was zero, and the upper limit P_{high} otherwise. In other words, the probability estimation virtually subdivides the actual interval in two subintervals, wherein the new actual probability interval is one of the two subintervals depending on the bin value to be encoded. Instead of assigning zero bin value to the lower subinterval, the lower subinterval could be associated with the most probable symbol (MPS), i.e. 1 or 0 depending on which has the higher probability according to the context model. Marpe '740 Pub at [0076]–[0077]. As can be seen from FIG. 8, the actual probability interval is getting narrower and narrower. Each bin is directed by switch 113 to context modeler 112. This changes, when the first bit of the primary suffix reaches stage 10 b. In this case, switch 113 passes the bin value to bypass coding engine 120. The bypass coding engine 120 operates on the same actual probability interval as regular coding engine 118, shown in
--	--

	FIG. 8 at 356. Contrary to the regular coding engine 118 the bypass coding engine 120 uses a fixed probability estimation or a static probability estimation. In particular, the fixed probability estimation used by bypass coding engine 120 assumes that the actual bin value is with equal probability a one or a zero, i.e. $P(0)=P(1)$. Thus, bypass coding engine 120 reduces the actual probability interval 356 to either the upper or lower half of the actual probability interval 356 when coding the actual bin of the primary suffix. As will be apparent from FIG. 8, the actual probability interval is reduced from bin to bin such that the new probability interval lies within the actual probability interval. The arithmetically coded bit stream that the regular coding engine and the bypass coding engine cooperatively output via switch 122 represents a probability value lying in the probability interval that is obtained after operating the syntax element of a whole slice. Marpe '740 Pub at [0082]. The decoder updates this adaptively varying probability estimation based on past decoded bins indicated by the respective context model type. The determination 402 includes subdividing an actual probability interval according to the adaptive probability estimation into two subintervals, assigning the bin value to be determined to the bin value associated with the subinterval the probability value indicated by the arithmetically coded bit stream lies in, and reducing the actual probability interval to the that subinterval, thereby simulating the steps performed by the encoder when creating the arithmetically coded bit stream as shown in FIG. 8. Marpe '740 Pub at [0114]. The entropy encoder 800 in FIG. 8 in accordance with a coding arrangement of FIG. 2. A context modeler feeds a context model, i.e., a probability information, to an arithmetic encoder, which is also referred to as the regular coding engine. The to be encoded bit, i.e. a bin, is forwarded from the context modeler to the regular coding engine. This bin value is also fed back to the context modeler so that a context model update can be obtained. A bypass branch is provided, which includes an arithmetic encoder, which is also called the bypass coding engine. The bypass coding engine is
--	--

	operative to arithmetically encode the input bin values. Contrary to the regular coding engine, the bypass coding engine is not an adaptive coding engine but works preferably with a fixed probability model without any context adaption. A selection of the two branches can be obtained by means of switches. The binarizer device is operative to binarize non-binary valued syntax elements for obtaining a bin string, i.e., a string of binary values. In case the syntax element is already a binary value syntax element, the binarizer 78 is bypassed. Therefore, in CABAC (CABAC=Context-based Adaptive Binary Arithmetic Coding) the encoding process consists of at most three elementary steps: 1. binarization 2. context modeling 3. binary arithmetic coding Marpe '740 Pub at [0119]. In the co-called regular coding mode, prior to the actual arithmetic coding process the given binary decision, which, in the sequel, we will refer to as a bin, enters the context modeling stage, where a probability model is selected such that the corresponding choice may depend on previously encoded syntax elements or bins. Then, after the assignment of a context model the bin value along with its associated model is passed to the regular coding engine, where the final stage of arithmetic encoding together with a subsequent model updating takes place (see FIG. 2).
[d] entropy coding symbols that correspond to the second category of syntax elements and that have bypassed context updating.	Marpe '740 Pub teaches and/or discloses entropy coding symbols that correspond to the second category of syntax elements and that have bypassed context updating. <i>See</i> limitation 1[c] above, which is incorporated herein by reference. <i>See also, e.g.,</i> Marpe '740 Pub at [0055]. For binary arithmetic coding, each binary decision or node 152, 154 of tree 150, i.e. each bin, is associated with a bit value probability estimation or a context model, i.e. Ci. As is described in more detail below, not all of the nodes or bins is associated with adaptive probability estimation models but may be associated with a static probability

	estimation, thereby reducing the context modeling overhead, i.e. the management for adapting the estimation according to past bins in order to increase the compression ratio. Marpe '740 Pub at [0083]. If the bit is a bit of the primary suffix, in 404 the next bit or bin value is determined by performing arithmetic decoding based on a static probability model. Step 404 includes subdividing the actual probability interval according to the static probability estimation into two equally sized subintervals, assigning the bin value to be determined to the bit value associated with the subinterval in which the probability value indicated by the arithmetically coded bit stream lies, and reducing the actual probability interval to that subinterval. Marpe '740 Pub at [0120]. Alternatively, the bypass coding mode is chosen for selected bins in order to allow a speedup of the whole encoding (and decoding) process by means of a simplified coding engine without the usage of an explicitly assigned model. This mode is especially effective when coding the bins of the primary suffix of those syntax elements, concerning components of differences of motion vectors and transform coefficient levels.
3. A method according to claim 1 further comprising performing binarization of the symbols that correspond to the second category of syntax elements prior to bypassing the context updating.	Marpe '740 Pub teaches and/or discloses a method according to claim 1 further comprising performing binarization of the symbols that correspond to the second category of syntax elements prior to bypassing the context updating. <i>See, e.g., Marpe '740 Pub at Fig. 2.</i>



Marpe '740 Pub at [0040].

The first stage 10 a is the binarization stage and comprises a binarizer 102. An input of the binarizer 102 is connected to an input 104 of stage 100 a via a switch 106. At the same time, input 104 forms the input of coding arrangement 100. The output of binarizer 102 is connected to an output 108 of stage 10 a, which, at the same time, forms the input of stage 100 b. The switch 106 is able to pass syntax elements arriving at input 104 to either binarizer 102 or binarization stage output 108, thereby bypassing binarizer 102.

Marpe '740 Pub at [0044].

Stage 100 b is a context modelling stage and comprises a context modeler 112 as well as a switch 110. The context modeler 112 comprises an input, an output and an optional feed-back input. The input of context modeler 112 is connected to the binarization stage output 108 via switch 110. The output of context modeler 112 is connected to a regular coding input terminal 114 of stage 100 c. The function of switch 113 is to pass the bits or bins of the bin string at binarization stage output 108 to either

	the context modeler 112 or to a bypass coding input terminal 116 of stage 100 c, thereby bypassing context modeler 112. Marpe '740 Pub at [0046]. When receiving the bin string or codeword of a component of a motion vector difference or a transform coefficient level, it is the function of switch 113 to direct the bits of the codeword to either the context modeler 112 or the bypass input terminal 116 depending on to whether the bit belongs to a primary prefix part or a primary suffix part of the code word, as will be described in more detail below. Marpe '740 Pub at [0050]. The bypass coding engine 112 has also an input and an output terminal, the input terminal being connected to the bypass coding input terminal 116. The bypass coding engine 120 is for binary arithmetically coding the bin values passed directly from binarization stage output 108 via switch 113 by use of a static predetermined probability distribution estimation and also outputs coded bits.
4. A method according to claim 3 wherein performing binarization comprises performing binarization so as to target a predetermined probability.	Marpe '740 Pub teaches and/or discloses a method according to claim 3 wherein performing binarization comprises performing binarization so as to target a predetermined probability. <i>See, e.g.,</i> Marpe '740 Pub at [0017]. The use of the inventive binarization scheme enables, in connection with binary arithmetic coding, an effective way of adapting the binarization representation of the data value to the probability distribution of the data values. A suitably selected cut-off value may lead to the least probable data values to be binarized into code words having equally probable bit values, whereas the most probable data values may be binarized into code words leading to a very effective arithmetic coding bit stream when using adaptive context modeling. Thus, coding based on a static probability estimation may be employed for the second binarization scheme suffix of the code word, whereas

	adaptive binary arithmetic coding is effectively applied to bits of the first binarization scheme primary prefix of the code word. Marpe '740 Pub at [0040]–[0041]. The first stage 10 a is the binarization stage and comprises a binarizer 102. An input of the binarizer 102 is connected to an input 104 of stage 100 a via a switch 106. At the same time, input 104 forms the input of coding arrangement 100. The output of binarizer 102 is connected to an output 108 of stage 10 a, which, at the same time, forms the input of stage 100 b. The switch 106 is able to pass syntax elements arriving at input 104 to either binarizer 102 or binarization stage output 108, thereby bypassing binarizer 102. The function of switch 106 is to directly pass the actual syntax element at input 104 to the binarization stage output 108 if the syntax element is already in a wanted binarized form. For the following discussion, it is assumed that the syntax element is not in the correct binarization form and therefore, syntax elements are generally thought to be non-binary valued syntax elements. The non-binary valued syntax elements are passed via switch 106 to binarizer 102. Binarizer 102 maps the non-binary valued syntax elements to a code word, or a so called bin string, in accordance with a binarization scheme embodiments of which are described below with respect to the following figures. The meaning of bin will be described in more detail below with respect to FIG. 3. Marpe '740 Pub at [0125]. An important property of the arithmetic coding as described above is the possibility to utilize a clean interface between modeling and coding such that in the modeling stage, a model probability distribution is assigned to the given symbols, which then, in the subsequent coding stage, drives the actual coding engine to generate a sequence of bits as a coded representation of the symbols according to the model distribution. Since it is the model that determines the code and its efficiency in the first place, it is of importance to design an adequate model that explores the statistical dependencies to a
--	--

	large degree and that this model is kept “up to date” during encoding. However, there are significant model costs involved by adaptively estimating higher-order conditional probabilities. These costs are reduced by use of the binarization schemes as described above.
6. A method according to claim 1 wherein categorizing the plurality of syntax elements comprises categorizing the plurality of syntax elements based on a relationship of the frequency of occurrence of the syntax elements to a predefined threshold.	Marpe '740 Pub teaches and/or discloses a method according to claim 1 wherein categorizing the plurality of syntax elements comprises categorizing the plurality of syntax elements based on a relationship of the frequency of occurrence of the syntax elements to a predefined threshold. <i>See, e.g., Marpe '740 Pub at [0046].</i> When receiving the bin string or codeword of a component of a motion vector difference or a transform coefficient level, it is the function of switch 113 to direct the bits of the codeword to either the context modeler 112 or the bypass input terminal 116 depending on to whether the bit belongs to a primary prefix part or a primary suffix part of the code word, as will be described in more detail below. Marpe '740 Pub at [0050]. The bypass coding engine 112 has also an input and an output terminal, the input terminal being connected to the bypass coding input terminal 116. The bypass coding engine 120 is for binary arithmetically coding the bin values passed directly from binarization stage output 108 via switch 113 by use of a static predetermined probability distribution estimation and also outputs coded bits. Marpe '740 Pub at [0076]. As can be seen from FIG. 8, the actual probability interval is getting narrower and narrower. Each bin is directed by switch 113 to context modeler 112. This changes, when the first bit of the primary suffix reaches stage 10 b. In this case, switch 113 passes the bin value to bypass coding engine 120. The bypass coding engine 120 operates on the same actual probability interval as regular coding engine 118, shown in

	FIG. 8 at 356. Contrary to the regular coding engine 118 the bypass coding engine 120 uses a fixed probability estimation or a static probability estimation. In particular, the fixed probability estimation used by bypass coding engine 120 assumes that the actual bin value is with equal probability a one or a zero, i.e. $P(0)=P(1)$. Thus, bypass coding engine 120 reduces the actual probability interval 356 to either the upper or lower half of the actual probability interval 356 when coding the actual bin of the primary suffix.
7. A method according to claim 1 further comprising performing binarization of the symbols that correspond to the first category of syntax elements prior to entropy coding the symbols that correspond to the first category of syntax elements.	Marpe '740 Pub teaches and/or discloses A method according to claim 1 further comprising performing binarization of the symbols that correspond to the first category of syntax elements prior to entropy coding the symbols that correspond to the first category of syntax elements. <i>See, e.g., Marpe '740 Pub at [0074].</i> It is assumed that now the first bin of the syntax element binarized in accordance with the above-described binarization scheme is passed to the context modeling stage 100 b. Let us assume that the syntax element is an absolute value of a transform coefficient level. The first bin in the binarization scheme for transform coefficient levels is, as can be seen in FIG. 5, in the last line of the table, and in FIG. 3, which shows the binarization code tree for the binarization scheme of FIG. 5, the first bit of the TU prefix of the binary string of the transform coefficient level. This bin is regularly past by switch 113 to context modeler 112. The context modeler 112 adjusts an adaptive probability estimation 352, according to which the first bin of the bin string has a probability of $P(0)$ to be zero and a probability of $P(1)$ to be one, wherein $P(1)+P(0)=1$. Context modeler 112 needs just one parameter in order to store or manage the adaptive probability estimation. Context modeler 112 adjusts the adaptive probability estimation in accordance with past bins or prior bins and in accordance with the context model type associated with the respective bin. The context model types defining the dependencies of the probability estimation on the past bins may be different for the different bins of the bin string of a syntax element. The context modeler 112 passes bin value and adjusted adaptive probability estimation 352 to regular coding engine 118. Regular coding engine 118 reduces the actual probability interval 350 to a new

	probability interval having a range of $P(0)$ multiplied with the range of the actual probability interval 350, i.e. with a range of $P(0) \cdot (P_{high} - P_{low})$, if the bit value of the first bin of the primary prefix is zero, and $P(1)$ multiplied with the range of the actual probability interval 350, i.e. with a range of $P(1) \cdot (P_{high} - P_{low})$, if the first bit value of the first bin of the primary prefix is one. The new probability interval shares the lower limit with the actual probability interval 350, if the actual bin value of the first bin was zero, and the upper limit P_{high} otherwise. In other words, the probability estimation virtually subdivides the actual interval in two subintervals, wherein the new actual probability interval is one of the two subintervals depending on the bin value to be encoded. Instead of assigning zero bin value to the lower subinterval, the lower subinterval could be associated with the most probable symbol (MPS), i.e. 1 or 0 depending on which has the higher probability according to the context model.
8. [pre] An apparatus 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 processor, cause the apparatus to at least:	See claim 1 [pre] above, which is incorporated herein by reference. See also, e.g., Marpe '740 Pub at Title ("Method and apparatus for binarization and arithmetic coding of a data value."). Marpe at [0012]. In accordance with the second aspect of the present invention, this object is achieved by an apparatus for decoding a coded bit stream which represents a binarized representation of the data value, the binarized representation of the data value being a codeword having a primary prefix, which is a binarization of the minimum of the data value and a predetermined cut-off value in accordance with a first binarization scheme and, if the data value is greater than the predetermined cut-off value, a primary suffix appended to the primary prefix, the primary suffix being a binarization of the difference of the data value minus the predetermined cut-off value in accordance with a second binarization scheme, having: means for, for each bit in the codeword, if the bit of the codeword is part of the primary prefix, determining the bit by binary arithmetically decoding the coded bit stream by means of a adaptively varying bit value probability estimation; and if the bit of the codeword is part of the primary suffix, determining the bit by binary arithmetically decoding the bit by means of a static bit

	value probability estimation, thereby obtaining the codeword representing the data value.
[a] determine a frequency of occurrence threshold based on an expected frequency of occurrence of syntax elements in a bit stream;	See limitation 1[a] above, which is incorporated herein by reference.
[b] categorize a plurality of syntax elements of video content into first and second categories based on the frequency of occurrence threshold, wherein syntax elements which occur greater than the frequency of occurrence threshold are categorized into the second category and syntax elements which occur less than the frequency of occurrence are categorized into the first category;	See limitation 1[b] above, which is incorporated herein by reference.
[c] entropy code symbols that correspond to the first category of syntax elements and that have been subjected to a context update; and	See limitation 1[c] above, which is incorporated herein by reference.
[d] entropy code symbols that correspond to the second category of syntax elements and that have bypassed context updating.	See limitation 1[d] above, which is incorporated herein by reference.

10. An apparatus according to claim 8 wherein the at least one memory and the computer program code are further configured to, with the processor, cause the apparatus to perform binarization of the symbols that correspond to the second category of syntax elements prior to bypassing the context updating.	See claim 3 above, which is incorporated herein by reference.
11. An apparatus according to claim 10 wherein the at least one memory and the computer program code are configured to, with the processor, cause the apparatus to perform binarization by performing binarization so as to target a predetermined probability.	See claim 4 above, which is incorporated herein by reference.
13. An apparatus according to claim 8 wherein the at least one memory and the computer program code are configured to, with the processor, cause the apparatus to categorize the plurality of syntax elements by categorizing the plurality of syntax elements based on a relationship of the frequency of occurrence of the syntax elements to a predefined threshold.	See claim 6 above, which is incorporated herein by reference.

14. An apparatus according to claim 8 wherein the at least one memory and the computer program code are further configured to, with the processor, cause the apparatus to perform binarization of the symbols that correspond to the first category of syntax elements prior to entropy coding the symbols that correspond to the first category of syntax elements.	See claim 7 above, which is incorporated herein by reference.
15. [pre] A computer program product comprising at least one non-transitory computer-readable storage medium having computer-executable program code instructions stored therein, the computer-executable program code instructions comprising program code instructions to:	See claim 1 [pre] above, which is incorporated herein by reference. See also, <i>e.g.</i>, Marpe '740 Pub at [0013] In accordance with the third aspect of the present invention, this object is achieved by a computer program having instructions for performing, when running on a computer, a method for decoding a coded bit stream which represents a binarized representation of the data value, the binarized representation of the data value being a codeword having a primary prefix, which is a binarization of the minimum of the data value and a predetermined cut-off value in accordance with a first binarization scheme and, if the data value is greater than the predetermined cut-off value, a primary suffix appended to the primary prefix, the primary suffix being a binarization of the difference of the data value minus the predetermined cut-off value in accordance with a second binarization scheme, the method having the following steps: for each bit in the codeword, if the bit of the codeword is part of the primary prefix, determining the bit by binary arithmetically decoding the coded bit stream by means of a adaptively varying bit value probability estimation; and if the bit of the codeword is part of the primary suffix, determining the bit by binary arithmetically decoding the bit by means of a static bit value probability estimation, thereby obtaining the codeword representing the data value.

[a] determine a frequency of occurrence threshold based on an expected frequency of occurrence of syntax elements in a bit stream;	See limitation 1[a] above, which is incorporated herein by reference.
[b] categorize a plurality of syntax elements of video content into first and second categories based on the frequency of occurrence threshold, wherein syntax elements which occur greater than the frequency of occurrence threshold are categorized into the second category and syntax elements which occur less than the frequency of occurrence are categorized into the first category;	See limitation 1[b] above, which is incorporated herein by reference.
[c] entropy code symbols that correspond to the first category of syntax elements and that have been subjected to a context update; and	See limitation 1[c] above, which is incorporated herein by reference.
[d] entropy code symbols that correspond to the second category of syntax elements and that have bypassed context updating.	See limitation 1[d] above, which is incorporated herein by reference.
17. A computer program product according to claim 15 wherein the computer-executable program code instructions further comprise program code instructions to perform binarization of the symbols that correspond to the second category of syntax elements prior to bypassing the context updating.	See claim 3 above, which is incorporated herein by reference.

18. A computer program product according to claim 17 wherein the comprise program code instructions to perform binarization comprise program code instructions to perform binarization so as to target a predetermined probability.	See claim 4 above, which is incorporated herein by reference.
20. A computer program product according to claim 15 comprise the program code instructions to categorize the plurality of syntax elements comprise program code instructions to categorize the plurality of syntax elements based on a relationship of the frequency of occurrence of the syntax elements to a predefined threshold.	See claim 6 above, which is incorporated herein by reference.