(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 October 2008 (16.10.2008)

PCT

(10) International Publication Number
WO 2008/123710 A1

(51) International Patent Classification:
H04N 7/24 (2006.01)

(21) International Application Number:
PCT/KR2008/001928

(22) International Filing Date: 4 April 2008 (04.04.2008)

(25) Filing Language: Korean

(26) Publication Language: English

(30) Priority Data:
10-2007-0033490 4 April 2007 (04.04.2007) KR
10-2007-0033495 4 April 2007 (04.04.2007) KR

(71) Applicant *(for all designated States except US)*: **HUMAX CO., LTD.** [KR/KR]; HUMAX Venture Tower 271-2, Seohyeon-dong, Bundang-gu, Seongnam-si, Gyeonggi-do 463-050 (KR).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **JANG, Euee-Seon** [KR/KR]; #814, Information & Communication, Bldg., Hanyang Univ., Haengdang 1-dong, Seongdong-gu, Seoul 133-791 (KR). **LEE, Chung-Ku** [KR/KR]; 15-304, Dong-a Apt., Bupyeong 1-dong, Bupyeong-gu, Incheon 403-011 (KR). **KWAK, Eun-Kyung** [KR/KR]; 232-61, Junggok 1-dong, Gwangjin-gu, Seoul 143-221 (KR). **LEE, Sun-Young** [KR/KR]; #707, Sanhak Bldg., Hanyang Univ., Haengdang 1-dong, Seongdong-gu, Seoul 133-791 (KR).

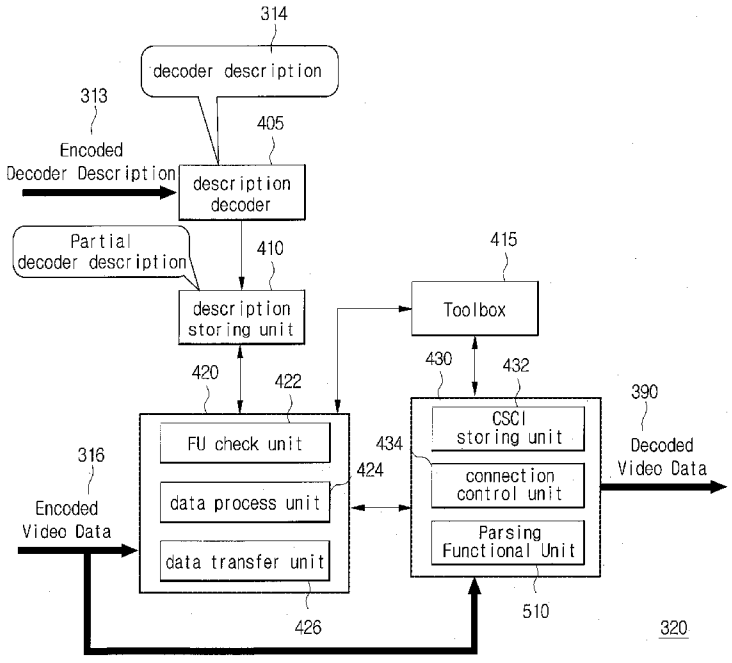
(74) Agent: **EZ INTERNATIONAL PATENT & TRADEMARK LAW OFFICE**; 404 BYC Bldg., 648-1 Yeoksam-dong, Gangnam-gu, Seoul 135-081 (KR).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO,

[Continued on next page]

(54) Title: BITSTREAM DECODING DEVICE AND METHOD HAVING DECODING SOLUTION

FIG. 5



(57) Abstract: A decoding device and method are disclosed. A decoding device includes: a decoder implementation unit generating and outputting Control Signal/Context Information (CSCI) control information and connection control information using partial decoder descriptions stored in a description storing unit; a tool box including a plurality of functional units, each of which is realized to perform a predetermined process; and a decoding solution loading the functional units in a hierarchical way and decoding a bitstream into video data by using the CSCI and the connection control information. With the present invention, various type of decoder can be reconfigured by using the decoder description.

WO 2008/123710 A1

RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report*

【DESCRIPTION】

【Invention Title】

BITSTREAM DECODING DEVICE AND METHOD HAVING DECODING SOLUTION

【Technical Field】

The present invention relates to a decoding device and method of video data and more particularly, to a bitstream decoding device and method having a decoding solution.

【Background】

Typically, video data is converted into bitstream data by an encoder. At this time, the bitstream is stored depending on an encoding type that satisfies the constraint condition of the encoder.

MPEG, which is the constraint condition of the bitstream, requests syntax and semantics.

The syntax, which refers to the structure, format, or length of data, shows the sequence of expressing the data. In other words, the syntax is to meet a rule for encoding/decoding and defines the sequence, length, format, and the like of each element in the bitstream.

The semantics refers to the meaning of each bit forming data. In other words, the semantics shows the meaning of each element in the bitstream.

Accordingly, various types of bitstreams can be generated depending on the encoding condition or the applied standard (or codec) of the encoder. Typically, each standard (e.g. MPEG-1, MPEG-2, MPEG-4, and MPEG-4 AVC, etc.) has different bitstream syntax.

Therefore, it can be said that the bitstream encoded according to each standard or encoding condition has different types (i.e. syntax and semantics). A decoder corresponding to a pertinent encoder may be used for the deciding of the bitstream.

As described above, the conventional bitstream decoder has a restriction that must satisfy the constraint condition of the encoder. This restriction makes it difficult to implement a united decoder corresponding to a plurality of standards.

【Disclosure】

【Technical Problem】

Accordingly, the present invention, which is designed to solve the aforementioned problems, provides a method and device for decoding a bitstream that is encoded by various types (syntax and semantics) in accordance with each standard (e.g. MPEG-1, MPEG-2, MPEG-4, MPEG-4 AVC) by using the same information recognizing method.

Further, the present invention provides a method and device for encoding a bitstream that can generate an extended bitstream including decoder descriptions or generate a bitstream and decoder descriptions respectively to decode bitstreams encoded by various types (syntax and semantics) in accordance with each standard by using the same information recognizing method.

Further, the present invention provides a method and device for decoding a bitstream that can parse a bitstream compressed by various encoding methods by using the same information analyzing method and systematically control each functional unit for decoding by using the parsed data.

Further, the present invention provides a method and device for encoding a bitstream that can describe decoder descriptions more efficiently by employing hierarchical structure of codec to the syntax parsing and the decoding process.

Further, the present invention provides a method and device for decoding a bitstream that can suggest scheduling management of each codec and systematic processing structure (i.e., parallel combination structure, sequential combination structure, independent processing structure, individual processing structure, etc.) of each functional unit by using the decoder description.

Further, the present invention provides a method and device for encoding a bitstream that can design and build various systems only with the above described decoder descriptions or hierarchy information.

Further, the present invention provides a method and device for encoding/decoding a bitstream that can universially apply a syntax analysis method to decode various types of bitstreams.

Further, the present invention provides a method and device for encoding/decoding a bitstream that can apply a new set of commands to parse various types of bitstreams by a common syntax analysis method.

Further, the present invention provides a method and device for decoding a bitstream that can decode bitstreams even when a syntax element is changed, added, or deleted.

Further, the present invention provides a method and device for decoding a bitstream that can share elements used for the bitstream decoding of the element information (i.e. a result from syntax parsing) of analyzed syntax.

Further, the present invention provides a method and device for decoding a bitstream that can use the element information of analyzed syntax for analyzing the syntax element of a following bitstream.

Further, the present invention provides a method and device for decoding a bitstream that can separate functions, which are included during various decoding processes suggested by several standards (codec), corresponding to each functional unit and provide the result into a tool-box

Further, the present invention provides a method and device for decoding a bitstream that can selectively use only necessary functional units from the tool box to decode a bitstream encoded by various types.

Further, the present invention provides a method and device for decoding a bitstream that can allow easy changing, inserting, or deleting functional units stored in the tool box.

Further, the present invention is to aim at international standardization of codec unification for the bitstream decoding, generation of decoder descriptions to be processed by the same information analyzing method and establishment of extended bitstream.

Other problems that the present invention solves will become more apparent through the following description.

【Technical solution】

To solve the above problems, an aspect of the present invention features a decoding device and/or a unified codec device that can be universally used for various standards.

According to an embodiment of the present invention, a decoding device can include A decoding device, comprising: a decoder implementation unit generating and outputting Control Signal/Context Information (CSCI) control

information and connection control information using partial decoder descriptions stored in a description storing unit; a tool box including a plurality of functional units, respectively, realized to perform a predetermined process; and a decoding solution loading the functional units in a hierarchical way and decoding a bitstream into video data by using the CSCI and the connection control information.

According to another embodiment of the present invention, A decoding method can include (a) generating and storing a plurality of partial decoder descriptions corresponding to the inputted decoder description; (b) generating CSCI control information and connection control information using the partial decoder descriptions; (c) storing a plurality of element information generated by syntax parsing of the bitstream using CSCI control information; (d) decoding the encoded video data of the bitstream using the connection control information and the element information and outputting decoded video data.

According to another embodiment of the present invention, a decoding device can include a tool box including a plurality of functional units, respectively, respectively, is realized to independently perform each process of at least one standard for decoding; description storing unit storing partial decoder descriptons for controlling operations of all or parts of the functional units; and a decoder implementation unit controlling to decode encoded video data into video data by hierarchically loading the functional units by referring to at least one partial decoder

description.

According to another embodiment of the present invention, a decoding method can include (a) generating and storing a plurality of partial decoder descriptions corresponding to the inputted decoder description wherein the decodescription comprses a plurality of partial decoder descriptions; and (b) loading a parsing functional unit and one or more of decoding functional units corresponding to the highest layer by using one or more of partial decoder descriptions, wherein, the parsing functional unit performs syntax parsing of the bitstream and the decoding functional units of the highest layer perform processing when control signal and context information (CSCI) for performing a predetermined process and data for decoding are stored in the storing unit by the parsing functional unit, when the control signal and context information (CSCI) for performing a first layer corresponding to the hierarchical structure and data for decoding are stored in the storing unit, the first layer is loaded by a second layer which is in an activation state.

【Advantageous Effects】

As described above, the present invention can decode a bit steam encoded by various types (syntax and semantics) in accordance with each standard (e.g. MPEG-1, MPEG-2, MPEG-4, MPEG-4 AVC) by using the same information recognizing method.

Further, the present invention can generate a bitstream, which is encoded by

various types (syntax and semantics) in accordance with each standard and decoder description, or extended bitstream including the decoder description and the bitstream such that the bitstream can be decoded by using the same information recognizing method.

Further, the present invention can efficiently describe a decoder descriptionby by applying hierachical structures of Codec to the process of syntax parsing and decoding.

Further, the present invention can provide a scheduling management of each codec and an organic processing structure (such as parallel connection structure, serial connection structure, independent processing structure and individual processing structure,ets) by using a decoder description

Further, the present invention can design and implement various codec systems with described decoder description and hierachical information.

Further, the present invention can parse a bit stream compressed by various methods by using the same information analyzing method and organically control each functional unit for decoding by using the parsed data.

Further, the present invention can commonly apply a syntax analyzing method for decoding various types of bitstreams.

Further, the present invention can apply a set of new commands for being capable of parsing a bit stream in various forms by using a common syntax analyzing

method.

Further, the present invention can easily decode a bit stream when a syntax element is changed, added or deleted.

Further, the present invention can share elements used for the bit stream decoding of the element information (i.e. a result from syntax parsing) of analyzed syntax.

Further, the present invention can use the element information of analyzed syntax to analyze following bit stream syntax element.

Further, the present invention can be used when unifying moving picture and still image codecs processed in units of block in addition to MPEG-1, MPEG-2, MPEG-4 and MPEG-4 AVC.

Further, the present invention can divide the functions forming various decoding methods suggested by diverse standards (codecs) per each functional unit and store the divided functions in a toolbox.

Further, the present invention can select in a toolbox and decode a functional unit necessary for decoding a bit stream encoded in various forms.

Further, the present invention can change, add or delete a functional unit stored in a tool box.

【Brief Description of the Drawings】

FIG. 1 is a schematic block diagram illustrating the structure of a typical decoder.

FIG. 2 is a schematic block diagram illustrating the structure of a typical encoder.

FIG. 3 is a schematic block diagram illustrating the structure of a decoder in accordance with an embodiment of the present invention.

FIG. 4 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with a first embodiment of the present invention.

FIG. 5 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with a second embodiment of the present invention.

FIG. 6 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with an embodiment of the present invention.

FIG. 7 illustrates function units for syntax (SYN) parsing in accordance with an embodiment of the present invention.

FIG. 8 illustrates function units for decoding processing in accordance with an embodiment of the present invention.

FIG. 9 illustrates the hierarchical structure assigned by a decoding hierarchy table(DHT) in accordance with an embodiment of the present invention.

FIG. 10 illustrates the calling structure between layers and the connection structure within layers assigned by a syntax rule table(S-RT) in accordance with an embodiment of the present invention.

FIG. 11 illustrates an interface set for functional units applied to 2 or more of standards in accordance with an embodiment of the present invention.

FIG. 12 illustrates the connection structure of a dedicated buffer space between nodes per each hierarchy in accordance with an embodiment of the present invention.

FIG. 13 illustrates the inclusion relationship of a dedicated buffer space for storing control signal and context information (CSCI) data in accordance with an embodiment of the present invention.

FIG. 14 illustrates the inclusion relation of a dedicated buffer space for storing output data by functional units in accordance with an embodiment of the present invention.

FIG. 15 schematically illustrates parallel processing of syntax parsing and data decoding in accordance with an embodiment of the present invention.

FIG. 16 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with a first embodiment of the present invention.

FIG. 17 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with a second embodiment of the present invention.

FIG. 18 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with a third embodiment of the present invention.

FIG. 19 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with a fourth embodiment of the present invention.

FIG. 20 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with a fifth embodiment of the present invention.

FIG. 21 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with a sixth embodiment of the present invention.

FIG. 22 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with a seventh embodiment of the present invention.

FIG. 23 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with an eigthth embodiment of the present invention.

FIG. 24 is a block diagram illustrating an encoder in accordance with an embodiment of the present invention.

FIG. 25 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with a third embodiment of the present invention.

FIG. 26 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with the forth embodiment of the present invention.

【Mode for Invention】

Since there can be a variety of permutations and embodiments of the present invention, certain embodiments will be illustrated and described with reference to the accompanying drawings. This, however, is by no means to restrict the present invention to certain embodiments, and shall be construed as including all permutations, equivalents and substitutes covered by the spirit and scope of the present invention.

Hereinafter, preferred embodiments will be described in detail with reference to the accompanying drawings.

FIG. 1 is a schematic block diagram illustrating the structure of a typical decoder, and FIG. 2 is a schematic block diagram illustrating the structure of the typical encoder.

As illustrated in FIG. 1, an MPEG-4 decoder 100 typically includes a variable length decoding processing unit 110, an inverse scanning unit 115, an inverse DC/AC prediction unit 120, an inverse quantization unit 125, an inverse discrete cosine transform unit 130 and a VOP reconstruction unit 135. It shall be evident that the decoder 100 can have a structure changed depending on an applying standard and some elements can be replaced with other elements.

If a transferred bitstream 105 is syntax-parsed and corresponding header information and encoded video data are extracted, the variable length decoding processing unit 110 forms a quantized discrete cosine transform (DCT) coefficient by

using predetermined Huffman table, the inverse scanning unit 115 generates data having the same sequence as pertinent video data 140 by performing inverse scanning. In other words, the inverse scanning unit 115 outputs a value in inverse order of scanning by various methods. In the encoding, after performing the quantization, a scanning direction can be defined depending on the distribution of frequency range. Typically, a zig-zag scanning method can be used. However, various scanning methods per codec can be used.

Syntax parsing can be integratedly performed in the variable length decoding processing unit 110 or in an element for processing the bitstream prior to the variable length decoding processing unit 110. In this case, since the same standard is applied to the corresponding encoder and decoder, the syntax parsing is processed by a predetermined setting only, to correspond to the pertinent standard.

The inverse DC/AC prediction unit 120 determines the direction of a reference block for prediction by using the size of the DCT coefficient at a frequency range.

The inverse quantization unit 125 performs the inverse quantization of inverse-scanned data. In other words, the inverse quantization unit 125 returns DC and AC coefficients by using a quantization parameter (QP) designed in an encoding process.

The inverse discrete cosine transform unit 130 calculates an actual video data pixel value to generate a video object plane (VOP) by performing inverse discrete

cosine transform.

The VOP reconstruction unit 135 decodes a video signal by using the VOP generated by the inverse discrete cosine transform unit 130 and outputs the decoded signal.

As illustrated in FIG. 2, an MPEG-4 encoder 200 typically includes a discrete cosine transform unit 210, a quantization unit 215, a DC/AC prediction unit 220, a scanning unit 230 and a variable length encoding unit 235.

Each element included in the encoder 200 performs the inverse functions of the corresponding elements of the decoder 100. This is well-known to those of ordinary skill in the art. Briefly describing, the encoder 200 converts a video signal (i.e. a digital video pixel value) to a frequency value through the DCT and the quantization and performs the encoding. Then, the encoder 200 performs variable length encoding to differentiate bit length according to the frequency number of information and outputs compressed bit stream format.

FIG. 3 is a schematic block diagram illustrating the structure of a decoder in accordance with an embodiment of the present invention, FIG. 4 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with a first embodiment of the present invention, FIG. 5 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with a second

embodiment of the present invention, FIG. 6 is a schematic block diagram illustrating the structure of an extended bitstream in accordance with an embodiment of the present invention, FIG. 7 illustrates function units for syntax (SYN) parsing in accordance with an embodiment of the present invention, FIG. 8 illustrates function units for decoding processing in accordance with an embodiment of the present invention, FIG. 9 illustrates the hierarchical structure assigned by a decoding hierarchy table(DHT) in accordance with an embodiment of the present invention, FIG. 10 illustrates the calling structure between layers and the connection structure within layers assigned by a syntax rule table(S-RT) in accordance with an embodiment of the present invention, FIG. 11 illustrates an interface set for functional units applied to 2 or more of standards in accordance with an embodiment of the present invention, FIG. 12 illustrates the connection structure of a dedicated buffer space between nodes per each hierarchy in accordance with an embodiment of the present invention, FIG. 13 illustrates the inclusion relationship of a dedicated buffer space for storing control signal and context information (CSCI) data in accordance with an embodiment of the present invention, FIG. 14 illustrates the inclusion relation of a dedicated buffer space for storing output data by functional units in accordance with an embodiment of the present invention, and FIG. 15 schematically illustrates parallel processing of syntax parsing and data decoding in accordance with an embodiment of the present invention.

As illustrated in FIG. 3, the decoder 300 of the present invention has a different

function from the conventional decoder (refer to FIG. 1).

The decoder description and bitstream are provided together to the decoder 300 for encoding and decoding processing in accordance with the present invention. The decoder description can be provided to the decoder 300 as the extended bitstream implemented with the bitstream or as independent data. If the information corresponding to the decoder description is memorized in a certain memory unit decoder description can not be provided. However, hereinafter the case that the decoder description is provided to the decoder 300 will be explained.

The decoder 300 in accordance with the embodiment of the present invention includes a dividing unit 310 and decoding processing unit 320. It shall be obvious that at least one of the illustrated elements (e.g. the dividing unit 310, the decoding processing unit 320 and its elements) can be realized as a software program (or the combination of program codes) embodied for performing a function to be described below.

The dividing unit 310 divides an inputted extended bitstream 305 into an encoded decoder description (DD) part and a typical bitstream (hereinafter, referred to as a conventional bitstream) part and outputs the encoded decoder description and the conventional bitstream to the decoding processing unit 320. The dividing unit 310 can output the encoded decoder description to a description decoder 405 and outputs the conventional bitstream to a decoder implementation unit 420. As mentioned above, it

the encoded decoder description and the conventional bitstream are received independently the dividing unit 310 can be omitted. The conventional bitstream can be the same or similar format of the bitstream 105 of Fig. 1.

An example of the extended bitstream 305 is illustrated in Fig. 6. As illustrated in FIG. 6, the extended bitstream 305 can include the decoder description 313 and the conventional bitstream 316. It is obvious that the extended bitstream 305 and the encoded decoder description 313 are not restriced by the examples in Fig. 4, which is to explan one of embodiments of the present invention.

The decoder description 314 that is decoded by the description decoder 405 is related to the structure information and encoding method (or the connection between functional units) of the conventional bitstream 316 and information of input/output data of functional units, in order to parse a bitstream, encoded by various encoding methods and/or by a function selected by a user among diverse functions by a common analyzing method. The decoder description 314 can be described by using a description method such as textual description or binary description. The decoder description can be ommited if the encoded decoder description 313 can be directly read by decoder implementation unit 420 without the processing of the description decoder 405.

The encoded decoder description 313 can be decoded into the decoder description 314 by the description decoder 405. The decoder description 314 can be

divided into partial decoder descriptions that are a functional unit list (FL) 610, a functional unit rule table (F-RT) 620, a functional unit CSCIT (FU-CSCIT) 630, a control signal and context information table (CSCIT) 640, , a decoding hierarchy table(DHT) 650, a syntax element table (SET) 660, a syntax-rule table (S-RT) 670 and a default value table (DVT) 680.

The partial decoder descriptions can be used as hierarchy information for performing predetermined process of functional units included in the decoder 300 being activated corresponding to hierarchical structures of the encoded video data.

Further, the partial decoder descriptions can be used as operation control information that define process and iput data of each functional unit and define the storing name of resulted data from the process performing. Of course, it is obvious that the operation control information can be included in the hierarchy information.

Futher, the hierarchy information can be specified by at least one of F-RT 620, DHT 645 and S-RT 660 as follows.

If it is assumed that the encoded video data have the hierarchical subordination structure like "the highest position layer, a next position layer, ..., a next position layer and the lowest position layer", the highest position layer is first activated and the next position layers are subsequently activated subsequently and the lowest position layer is finally activated.

Although each layer is subsequently activated, decoding process for encoded

video data can not be completed by perforing subsequently operations of functional units included in each layer. There can be the cases that the decoding process for encoded video data can be completed by perforing alternatively operations of functional units of many layers.

If one layer is activated, every decoding functional unit included in an object of the layer and at least one calling unit of a next position layer included in the object are activated.

The decoding functional units included in activated layers start their operations if Control Signal/Context Information (CSCI) data necessary for predetermined process and data for decoding are stored in CSCI storing unit 432.

Further, if CSCI data necessary for performing at least one process of decoding functional units of the same layer and data for decoding are stored in the CSCI storing unit 432, the calling unit of the activated next position layer generates and executes an object of the pertinent layer and therby every decoding functional unit included in the pertinent object and at least one calling unit included in the next position layer included in the pertinent object are activated. As mentioned above, a lower layer can be called by performing object of one layer.

It is evident that each partial decoder description in the decoder description can be arranged in various orders.

Here, the functional unit list(FL) 610, the functional unit rule table(F-RT) 620, the functional unit control signal and context information table(FU-CSCIT) 630, the control signal and context information table(CSCIT) 640, and the decoding hierarchy table(DHT) 650 may be used to build a recofiguable connection of each functional unit(as necessary, a partial decoder description may be referred to as "a first decoder description").

Among these, the decoding hierarchy table (DHT) 650 may be information describing for hierarchical structure of each codec and a lower layer per each hierarchy (see FIG. 9). The hierarchical structure of each codec may be described with upper layers and lower layers and may be subdivided into a sequence layer, a GOP layer, a VOP layer, a slice layer, a macro block layer, a block layer and the like. Each codec is composed of at least one of those layers and at least one object may be included in each layer.

A process of decoding encoded video data composed of 2 or more layers is described briefly as follows. However, it is not limited to this method for decoding bitstreams corresponding to the hierarchical structure.

During the decoding, after an object in a first layer, which is the highest layer, is generated and performed, all decoding functional units included in the corresponding object and one or more of calling units of a second layer, which is a lower layer, included in the correspondidng object are activated. Wherein, the calling unit can be an

element of the connection control unit 434 or contol operation of the connection control unit 434.

For example, if it is assumed that the hierarchical structure is constructed in order of a first layer, a second layer and a third layer, one or more of layer calling units (i.e., GOP-1 layer, GOP-2 layer, GOP-3 layer) included in the second layer is activated by performing an object of the first layer(i.e., a sequence layer), one or more of layer calling units (i.e., a picture-1 layer, a picture-2 layer with performing the object of the GOP-1 layer, picture-3, picture-4 with performing the object of the GOP-2 layer, etc.) included in the third layer is activated by performing at least one object in the second layer. It is not limited to only that an upper layer calls a lower layer in the activation process by calling in the hierarchical structure and it is apparent that an upper layer can be called by a lower layer, if necessary, based on decoder description or partial decoder description.

The functional units in the activated layer (the first layer) start operation if control signal and context information for performing a predetermined process and data for decoding are stored in the CSCI storing unit 432.

The calling unit in the lower layer (the second layer) of the activated layer (the first layer) generates and performs an object of the corresponding layer (the second layer) if control signal and context information for performing a process by one or more of the decoding functional units of the included layer and data for decoding are stored in

a CSCI storing unit 432. All decoding functional units in the corresponding object and at least one calling units of the lower layer (the third layer) included in the corresponding object are activated.

As described, when an object of a certain layer is performed, its lower layer is and decoding functional units in the called layer perform a predetermined process. Functional units selected by a decoder description among the functional units included in each layer perform more than once and the encoded video data therefrom is reconstructed to video data and the result data is outputted.

The FU-CSCIT may be referred to as partial decoder description information for mapping between element information stored in each functional unit of the toolbox 415 and the CSCI storing unit 432. In this case, the element information can be a conrol parameter for each functional unit.

The control signal and context information table(CSCIT) 640, the decoding hierarchy table(DHT) 650, the syntax element table(SET) 660, the syntax-rule table(S-RT) 670, the default value table(DVT) 680, and the like may be used for the syntax parsing of the conventional bitstream 316 (as necessary, a partial decoder description may be referred to as "a second decoder description"). The type and function of the each partial decoder description information will be described in detail below.

The description decoder 405 decodes the encoded decoder description inputted from the dividing unit 310 into the decoder description 314 and divides the decoder description 314 to 2 or more partial decoder descriptions which are the types to be recognized by the decoder implementation unit 420 or a decoding solution 430 and sotre the partial decoder descriptions in the description storing unit 410. It is not necessary that each partial decoder description stored in the description storing unit 410 is a typical type of table but enough that it is any information type to be recognized by the decoder implementation unit 420 or decoding solution 430.

The partial decoder descriptions, which are to be stored in the description storing unit 410 by the decoder description analysis of the decoder description decoder 405, may include a FL 610, a F-RT 620, a FU-CSCIT 630, a CSCIT 640, a DHT 650, a SET 660, a S-RT 670, a DVT 680, etc. The description decoder 405 may identify each partial decoder description by inserting a table indentifier (TI).

Of course, it is not necessary that all partial decoder descriptions should be included in the description decoder but may be enough if a codec number and a profile and layer number are included, or if a codec number and a profile and layer number for a part of partial decoder descriptions are included. When a codec number and a profile and layer number are included, the description decoder 405 may select a partial decoder description corresponding thereto from the partial decoder descriptions pre-stored in the description storing unit 410 to be used during the decoding without generating a new

partial decoder description for a whole partial decoder description or a part of partial decoder descriptions. When a codec number, a profile and layer number, and amendment information are included, the description decoder 405 may extract the partial decoder description corresponding to an appropriate code from the partial decoder descriptions pre-stored in the description storing unit 410 and generate a new partial decoder description reflecting the amendment information. When a codec number and a profile and layer number are not included and description for generating a partial decoder description is included, the description decoder 405 may generate a new partial decoder description for a whole partial decoder description or a part of partial decoder descriptions to use during the decoding.

In addition, the decoder description may further include revision information 1130 besides the decoder description (DD-T) for each partial decoder description.

Partial decoder descriptions divided by the description decoder 405 are stored in the description storing unit 410. The description storing unit 410 may pre-store one or more partial decoder descriptions which are corresponding to that, when the extended bitstream 305 includes a codec number and a profile and layer number, to be used by the decoding processing unit 320. Configuration of encoded decoder description 313 for extracting the partial decoder descriptions will be described below with reffrence to Fig. 16 through Fig. 23.

Embodiments of the decoding processsing unit 320 are illustrated in FIG. 4 and Fig. 5.

As illustrated in Fig.4, the first embodiment of the decoding processsing unit 320 can include a description decoder 405, a description storing unit 410, a toolbox 415, and decoder implementation Unit 420 and decoding solution 430.

The decoder implementation unit 420 can include a FU check unit 422, a data process unit 424, a data transfer unit 426.

The FU check unit 422 checks if the every functional unit is described in partial decoder descriptions (such as FL 610) exists in the toolbox 415 after the partial decoder descriptions are stored in description storing unit 410 by process of description decoder 405.

If there are not any functional units among functional units decribed in the partial decoder descriptions in toolbox 415, error massage can be displayed in a display unit or update of corresponding functional units can be request to a user. Of course, the FU check unit 422 can automatically update through the network if it is connected to a server for update of functional units.

The data process unit 424 classifies the partial decoder descriptions stored in the description storing unit 510 by the role of the partial decoder descriptions and process the partial decoder descriptions such that the partial decoder descriptions has the data format that can be operated by decoding solution 430. This is because the format

(such as table format) of the partial decoder description stored in description storing unit 410 is not suitable or the decoding solution 430 may need some information for efficiency.

As the function and format of the each partial decoder description is described above, only the processed data format is briefly described below.

The data process unit 424 process CSCI control information and connection control information after classifying the each partial decoder description according to the role for being used for generating or storing CSCI(Control Signal/Context Information) or the role for being for connecting the functional units.

For example, the partial decoder descriptions used for generating CSCI can be SET 660, S-RT 670, CSCIT 640 and DVT 680 and the partial decoder descriptions used for generating the connection control information can be FL 610, F-RT 620, S-RT 660 and FU-CSCIT 630.

The CSCI control information and connection control information processed by the data processing unit 424 can be described by the representation format of abstract decoder model ADM as follows.

Of course, it is obvious that the representation format is not restriced by the only ADM format.

Fisrt, the CSCI control information can be described as follows.

<CSCIs>

<csci_memory id="C0" name="CSCI #0" type="integer" />

<csci_memory id="C1" name="CSCI #1" type="integer" />

<csci_memory id="C2" name="CSCI #2" type="array" />

<csci_memory id="C3" name="CSCI #3" type="integer" />

<csci_memory id="C4" name="CSCI #4" type="integer" />

<csci_memory id="C5" name="CSCI #5" type="integer" />

<csci_memory id="C6" name="CSCI #6" type="integer" />

<csci_memory id="C7" name="CSCI #7" type="integer" />

<csci_memory id="C8" name="CSCI #8" type="integer" />

<csci_memory id="C9" name="CSCI #9" type="integer" />

<csci_memory id="C10" name="CSCI #10" type="integer" />

<csci_memory id="C11" name="CSCI #11" type="integer" />

<csci_memory id="C12" name="CSCI #12" type="integer" />

.....

</CSCI>

Next, the connection control information can be described as follows.

<Network name="Decoder">

<Package>

<QID>

```
<ID id="MPEG4 Simple Profile" />
</QID>
</Package>
<Port kind="Input" name="BITSTREAM" />
<Port kind="Ouput" name="YUV" />
<Instance id="1">
<Class name="Parser">
<QID>
<ID id="c" />
</QID>
</Class>
<Note kind="label" name="Stream Parser" />
</Instance>
<Instance id="2">
<Class name="VS">
<QID>
<ID id="c" />
</QID>
<Note kind="label" name="Video Session" />
</Class>
```

```
</Instance>
<Connection src="" src-port="BITSTREAM" dst="1"
dst-port="BITSTREAM" />
<Connection src="1" src-port="CSCI" dst="2" dst-port="CSCI" />
<Connection src="1" src-port="DATA" dst="2" dst-port="DATA" />
<Connection src="2" src-port="YUV" dst="" dst-port="YUV" />
</Network>
```

The data transfer unit 426 trnasfer the CSCI control information and connection control information processed by the data process unit 424 to the decoding solution 430. The data transfer unit 426 can transfer CSCI control information to CSCI storing unit 432 of wichi role is the storing and operating the CSCI information and can transfer connection control information to a connection control unit 434. In the case that the CSCI storing unit 432 store only CSCI information and the operation of CSCI information is performed by connection control unit 434, it is obvious that the data transfer unit 424 can CSCI control information and connection control information can be transferred.

The decoding solution 430 includes CSCI storing unit 432 and conncetion control unit 434. Although it is not illustrated, the decoder solution 430 can additionally include a work memory for loading the functional units by connection control unit 434

and performing the predetermined process using them.

The second embodiment of the decoding processsing unit 320 is illustrated in Fig. 2. The second embodiment of the decoding processsing unit 320 can include a description decoder 405, a decscription storing unit 410, a toolbox 415, a decoder implementation unit 420, and a decoding solution 430.

As compared to the Fig. 4, the second embodiment of the decoding processsing unit 320 can additionally include a parsing functional unit 510.

The parsing functional unit 510 is the functional unit for performing syntax parsing of bitstream.

The parsing functional unit 510 can be included the decoding solution 430 as an independent element. However, instead of the structure that the parsing functional unit 510 is included the decoding solution 430, it is also obvious that the decoding solution 430 includes two work memories and the connection control unit controls the one work memory to load only the functional units for decoding process and controls the other work memory to load only the parsing functional unit 510.

The both structures mentioned above have advantages that the parsing process of bitstream and decoding process can be parallel performed.

Besides that, compared to Fig. 4, the data process unit 424 processes syntax parsing control information and provides the syntax parsing control information to the decoding solution 430 for the process of the parsing functional unit 510. Accordingly,

the role of the partial decoder descriptions used for generating the CSCI control information, the connection control information and the syntax parsing control information.

That is, the data process unit 424 classifies the partial decoder descriptions by the role for generating and storing of CSCI, the role for syntax parsing, and the role for connecting the functional units and then processes the CSCI control information, the connection control information, and the syntax parsing control information.

For example, the partial decoder descriptions used for generating CSCI can be CSCIT 640 and the partial decoder descriptions used for generating the connection control information can be FL 610, F-RT 620, DHT 650 and FU-CSCIT 630 and the partial decoder descriptions used for generating the syntax parsing control information can be SET 660, S-RT 670, DHT 650, CSCIT 640 and DVT 680.

AS mentioned above, the CSCI control information and connection control information processed by the data processing unit 424 can be described by the representation format of abstract decoder model ADM. Below is the example of the representation of syntax parsing control information.

<syntax>

<syntax_element id="S0" name="Syntax #0">

```
<process>
  <cmd type="READ">
    <parameter index=0>32</parameter>
    <parameter index=1>B</parameter>
  </cmd>
  <cmd type="EXPRESSION">
    <parameter index=0>(C0=(IBS==HEX:1B0))</parameter>
  </cmd>
</process>
</syntax_element>

<syntax_element id="S1" name="Syntax #1">
  <process>
    <cmd type="READ">
      <parameter index=0>8</parameter>
      <output type="CSCI">C1</output>
    </cmd>
  </process>
</syntax_element>
(……)
```

</syntax>

The data transfer unit 426 trnasfer the CSCI control information and connection control information and syntax parsing control information processed by the data process unit 424 to the decoding solution 430. Here, the syntax parsing control information can include at least one information between the information for controlling selective load of parsing functional units in Fig. 7 and CSCI control information.

The data transfer unit 426 can transfer CSCI control information to CSCI storing unit 432 of wichi role is the storing and operating the CSCI information and can transfer connection control information to a connection control unit 434 and can transfer syntax parsing control information to the parsing functional unit 510.

In the case that the CSCI storing unit 432 store only CSCI information and the operation of CSCI information is performed by connection control unit 434, it is obvious that the data transfer unit 424 can CSCI control information and connection control information can be transferred.

Besides that, if the parsing functional unit 510 performs syntax parsing by the control of the connection control unit 434, it is also obvous that the data transfer unit 424 can transfer the syntax parsing control information to the connection control unit 434.

The decoding solution 430 includes a CSCI storing unit 432, a conncetion

control unit 434, and a parsing functional unit 510. The decoding solution 430 can additionally include a work memory for loading the functional units by connection control unit 434 and performing the predetermined process using them.

The parsing functional unit can be the functional unit being able to perform systax parsing of every bitstream regardless of their encoding format or the functional unit that are generated for syntax parsing of a certain type of bitstream. That is, the parsing functional unit 510 can be more than one functional unit for performing syntax parsing.

As illustrated in Fig. 4 and Fig. 5, the decoder 300 of the present invention can reconfigure the decoding process according to the encoding type of the inputted bitstream by selectively loading and operating the functional units in the toolbox 415.

As mentioned above, the decoder implementation unit 420 or the connection control unit 434 performs the function for implementing the decoder to decode the inputted bitstream and the decoding solution 430, which is the decoder configured by control of the decoder implementation unit 420 or the connection control unit 434, performs the decoding process of the bitstream.

The present invention can implement plural decoding solutions using a decoder implementation information by dividing two parts which are relation of cause and effect in their functions and then provide more efficient decoding process.

Hereinafter, the function and operation of each element of the decoding

processing unit 320 is described below with reference to pertinent figures.

As described above, the description decoder 405 decodes the encoded decoder description 313 inputted into the decoder description 314 and stores the partial decoder descriptions of the decoder description 314 in the description storing unit 410. The toolbox 415 includes functional units implemented for performing predetermined process. The functional units can include functional units for syntax parsing (see Fig. 7) and functional units for decoding process of encoded video data (see Fig. 8). The group of functional units for syntax parsing is called parsing function group 700 and the group of functional units for decoding process is called decoding function group 800.

For example, the parsing function group 700 can be realized as one functional unit. It is also possible that the parsing function group 700 and/or each function unit can be realized as a combination of each program codes.

In other words, the tool-box 510 is a region where the functional units can be realized to perform the corresponding functions, respectively so that each functional unit in the decoding function group 800, which is called decoding functional unit, is activated (selectively loaded) by the decoding control unit 434 and outputs the encoded video data in the conventional bitstream 316 into video data. The connection control unit 434 may control to sequentially activate the functional units corresponding to each hierarchical structure (for example, activation sequence of a first layer, a second layer, a third layer, etc., at the time when the second layer is activated, the first layer may be

kept in the activation state or be deactivated if the data processing is not necessary anymore). However, even though functional units in the same hierarchical structure are simultaneously activated, if a functional unit uses the result data processed by another functional unit as input data, the process is reserved till the result data is generated by the another functional unit and stored in the CSCI storing unit 432.

The decoding functional units each may belong to at least one divided layers according to the characteristics of codec. A called functional unit included in the parsing function group 700, which is called parsing functional unit, performs the syntax parsing of a bitstream and the decoding functional units perform the processing if CSCI required for performing a predetermined process and data for decoding are stored. An upper layer may call functional units in a lower layer according to flow of CSCI and data by the control of connection control unit 434 provided connection control information from the data transter unit 426 and video data corresponding to the bitstream is outputted by performing one or more times of the process according to the need of all functional units described in the decoder description. Thus, scheduling management of each codec and sysmtemic processing of each functional unit are provided by using the decoder description or the partial decoder descriptions.

Of course, as described with reference to Fig. 5, the parsing functional unit 510 can be set immediately to interprete the conventional bitstream 316 without the connection control of the connection control unit 434 by being included in decoding

solution 430 and using the syntax parsing control information provided from the decoder implementation unit 420. This is because following functional units can use the element information stored in CSCI storing unit 432, which is analyzed by the parsing functional unit 510, and mcroblock sized video data outputted form the parsing functional unit 510.

The parsing functional unit loaded by the control of the connection control unit 434 or the parsing functional unit (hereinafter parsing functional unit 510) in decoding solution 430 interpetes the inputted conventional bitstream 316 and stores element information, which is the result of the syntax parsing, in the CSCI storing unit 432. For example, the CSCI storing unit 432 may be a buffer memory. The element information may be control signal context information (CSCI). The element information, which is parsed by the parsing functional unit 510 and stored in the CSCI storing unit 532, may be the parsed result value of a pertinent step, and at the same time, may be an input value for determining the syntax of the following bitstream.

Further, the parsing functional unit 510 can perform entropy decoding for a header of the conventional bitstream 316 and video data and output predetermined video data macroblock sized to following functional unit according to the connection control of connection control unit 434.

Of course, the parsing functional unit 510 can also store the macroblock sized video data to predetermined buffer memory and following functional unit according to loading (that is, connection control) of connection control unit 434 can read and processe the macroblock sized video data and store video data processed in a pertiment buffer memory.

In other words, it is obvious that the parsing functional unit 510 can store the macroblock sized video data in CSCI storing unit 432 or a buffer memory and then the connection control unit 434 can provide the macroblock sized video data stored to a selected functional unit or the selected functional unit can read pertinent video data from the buffer memory. However, hereinafter, it is assumed that the macroblock sized video data, which is outputted from the parsing functional unit 510, is directly inputted to the functional unit according to the connection control of the connection control unit 434.

The parsing functional unit 510 can be realized as a software program (including the combination of program codes). Even though the parsing functional unit 510 is realized to perform 2 or more functions corresponding to 2 or more standards (for example, MPEG-1/2/4/AVC, etc.), respectively, the parsing functional unit 510 can perform an appropriate operation by using syntax parsing control information. Of course, the parsing functional unit 510 may be realized with subdivision into 2 or more of functional units as shown in FIG. 7 and it is apparent that each functional unit be realized as a combination of blocked program codes.

As described above, when the conventional bitstream 316 is inputted in the decoding processing unit 320, the connection control unit 434 can perform syntax parsing by loading the parsing functional unit using the connection control information inputted from the decoder implementation unit 420 or the parsing functional unit 510, inputted the syntax parsing control information from the decoder implementation unit 420, can perform syntax parsing.

Further, the connection control unit 434 controls elements to load or activate the functional units of the highest layer referring to the connection control information (which can be corresponding to F-RT 620).

Although, in the begining steps, one of decoder descriptions between the first decoder description and the second decoder description may be used, the first decoder description and the second decoder description is respectively used in the way of cross

reference or other ways in the end.

Each functional unit loaded selectively is on stanby until CSCI or/and data necessary for processing is stored in CSCI storing unit 432 and after the necessary CSCI or/and data is recored, each functional unit starts it's processing.

By this, the decoding processing unit 320 of the present invention can perform simultaneously the processing of syntax parsing and data decoding for video. (see Fig. 15)

Further, organic operation of each functional unit (such as, operations implemented by the serial or parallel connection structure of functional units) can be easily performed because activation of functional units included in the decoding function group 800 and reservation or beginning of the operation of each functional unit is determined by the hierarchical structure unit.

The parsing functional units or the parsing functional unit 510 analyzes the inputted conventional bitstream 316 and strores the element information, which is the result of syntax parsing by referring to the syntaz parsing control information corresponding to DHT 650, SET 660, S-RT 670, CSCIT 640, DVT680, etc.

For example, the CSCI storing unit 432 can be a buffer memory. For example, the element information can be Control Signal/Context Information (CSCI). The element information stored in the CSCI storing unit 432, parsed by the parsing functional unit 510, can be result values of a pertinent step and input values determining

following syntax of the conventional bitstream 316 at the same time.

Further, the prasing functional unit 510 can store predetermined sized (for example, one pixel unit, 4x4 unit, 8x8 unit, etc) encoded data in CSCI storing unit 432 for decoding function units (see Fig. 8) being able to use them after performing entropy decoding for a header and video data of the conventional bitstream 316 syntax parsed.

For example, the parsing functional unit 510 can store data for using in the decoding process such as DC/AC among syntax in the CSCI storing unit 432. The CSCI storing unit 432 can be a buffer memory and exclusive buffer space for each functional unit can be assigned. (See Fig. 12.) Further, each buffer space can be set to have inclusion relations of each hierarchical layer as illustrated in Fig. 13 and Fig. 14.

Thus, the functional unit loaded in a work memory can start its process if input data are recoded in CSCI storing unit 432 and in the case that result data processed are used as input data of other functional unit, the pertinent result data can be recorded in CSCI storing unit 432.

The parsing functional unit 510 can be realized as a software program (including the combination of program codes). Even though the parsing functional unit 510 is realized to perform 2 or more functions corresponding to 2 or more standards (for example, MPEG-1/2/4/AVC, etc.), respectively, the parsing functional unit 510 can perform an appropriate operation by using syntax parsing control information corresponding to DHT 650, SET 660, S-RT 670, CSCIT 640, DVT 680, etc.

Of course, the parsing functional unit 510 may be realized with subdivision into 2 or more of functional units as shown in FIG. 7 and it is apparent that each functional unit be realized as a combination of blocked program codes.

The function of the parsing functional unit 510 or functional units for performing syntax parsing in the toolbox 415 will be described below with particular description of each functional unit (parsing function group 700) as shown in FIG. 7.

The parsing function group 700, as illustrated in FIG. 7, can include a network abstraction layer parsing functional unit (NALP FU) 710, a syntax parsing functional unit (SYNP FU) 720, a context determination functional unit (CTX FU) 730, a variable length decoding functional unit (VLD FU) 740, a run length decoding functional unit (RLD FU) 750 and a macro block generator functional unit (MBG FU) 760, etc.

Of course, it shall be evident that the parsing function group 700 can include all functional units for syntax parsing regardless of the applied standard, functional units requested for a technology development process can be newly added, a previous functional unit can be amended and an unnecessary functional unit can be deleted. It is also evident that each functional unit included in the parsing function group 700 can be combined as one functional unit in case that they are independently not provided for each standard and can be identically processed regardless of the standards. Since the functions of each of the functional units are well-known to those ordinary skilled in the art, they will be described briefly.

The NALP FU 710 is a functional unit parsing the network abstraction layer of MPEG-4 AVC, and the SYNP FU 720 is a functional unit parsing the syntax of the bitstream. The SYNP FU 720 can be included in the VLD FU 740.

The CTX FU 730 is a functional unit determining the VLC table of MPEG-4 AVC, and the VLD FU 740 is a functional unit performing an entropy decoding.

The RLD FU 750 is a functional unit performing the entropy decoding of AC values, and the MBG FU 760 is a functional unit generating macro block data by coupling DC values with AC values. All or some of functional units in the aforementioned parsing functio group 700 can be included in the VLD FU 640 depending on a system realizing method.

As described above, the parsing function group 700 can be realized as one software program or 2 or more of software programs (e.g. the VLD FU 740 is realized as an independent software program). The parsing function group 700 can extract or generate element information by using syntax parsing control information corresponding to a first description information (that is, at least one of a decoding hierarchy table(DHT) 650, a syntax element table(SET) 660, a syntax-rule table(S-RT) 670, a control signal and context information table(CSCIT) 640, a default value table(DVT) 680) and store the element inforation in the CSCI storing unit 432.

The functions of decoding functional units are described below by explaining each functional unit of decoding function group 800 illustrated in Fig. 8. The decoding functional units can be implemented in the toolbox 415.

The decoding function group 800 processes the encoded data stored in the CSCI storing unit 432 in a predetermined processing unit and outputs into video data in a predetermined size by an arbitrary functional unit of the parsing function group 700 performing predetermined process. The processing unit of the encoded data can be predetermined to be applied differently for each functional unit or be generalized to be identical.

Functional units may be included in the decoding function group 800 to perform the above-described funtions according to each standard. Each functional unit can be realized by an independent processing block (e.g. software program, the combination of command codes, function, etc.) or the decoding function group 800 may be realized as one combined processing block. However, it is apparent that the decoding function group 800 perform an approptiate processing by a connection contol of the connection control unit 434 even though it is realized as one combined processing block.

As illustrated in FIG. 8, the decoding function group 800 can inlcude a de-blocking functional unit (DF FU) 810, a VOP reconstructor functional unit (VR FU) 815, a frame field reordering functional unit (FFR FU) 820, an intra prediction and

picture reconstruction functional unit (IPR FU) 830, an inverse transform functional unit (IT FU) 835, an inverse quantization functional unit 845, an inverse AC prediction functional unit (IAP FU) 855, an inverse scan functional unit (IS FU) 860 and a DC reconstruction (DCR FU) 865.

A IT 4 x 4 FU 840, an IQ 4 x 4 FU 850 and DCR 4 x 4 FU 870 process the block having 4 x 4 size. This is because there is a case that MPEG-4 AVC processes the block of 4 x 4 size, while MPEG-1/2/4 processes the block of 8 x 8 size in transform, quantization.

It shall be evident that the decoding function group 800 can include all functional units for data decoding function regardless of the applied standard, a functional unit requested for a technology development process can be newly added, a previous functional unit can be amended and an unnecessary functional unit can be deleted. For example, in the case of additionally requesting an IS 4 x 4 FU processing data with 4 x 4 block size, the decoding function group 800 can further include an appropriate functional unit. Also, the decoding function group 800 can further include a special prediction functional unit (SPR FU) for performing the intra prediction at the MPEG-4 AVC(not shown).

It is also evident that each functional unit included in the decoding function group 800 can be combined as one functional unit in case that they are independently not provided for each standard and can be identically processed regardless of the

standards. Since the functions of each of the functional units are well-known to those ordinary skilled in the art, they will be described briefly.

The DF FU 810 is the de-blocking filter of MPEG-4 AVC, and the VR FU 815 is a functional unit storing a reconstructed pixel.

The FFR FU 820 is a functional unit for an interlaced mode, and the IPR FU 830 is a functional unit performing the intra prediction of MPEG-4 AVC and storing the reconstructed pixel value. As described above, the intra prediction of MPEG-4 AVC can be performed by the SPR FU.

The IT FU 835 is a functional unit performing the inverse transform of DC values and AC values, and the IQ FU 845 is a functional unit performing the inverse quantization of AC vlaues.

The IAP FU 855 is a functional unit performing the inverse AC prediction of AC values, and the IS FU 860 is a functional unit performing the inverse scan of AC values. The DCR FU 865 is a functional unit performing the inverse prediction and inverse quantization of DC values.

Each operation of the aforementioned parsing function group 700 and the decoding function group 800 is indepedently performed, and each functional unit in the decoding function group 800 performs independently its operation when the CSCI and/or data for the processing in the CSCI storing unit 432 is(are) stored after the connection control unit 434 activates by the activation control.

Element information (for example, CSCI) which is the result value from the syntax parsing by using the first decoder description (that is, CSCIT 640, DHT 650, SET 660, S-RT 670, DVT 680) at the parsing function group 700 is stored in the CSCI storing unit 432. The element information is stored with corresponding to CSCIT 640. The CSCI storing unit 432 may be a buffer memory.

The element information stored in the CSCI storing unit 432 may be used as input data for performing the process of the SET 660 by the parsing function group 700 or as a control variable determining a connection index which is the following process by the S-RT 670

The element information stored in the CSCI storing unit 432 may be used when each functional unit, which the connection control unit 434 activated in the hierarchical structure unit, performs mapping the input CSCI designated in the FU-CSCIT 630 with the element information stored in the CSCI storing unit 432 by referring the connection control information (for example, information corresponding to F-RT 620). As described above, if the element information can be used by the functional units in the decoding function group 800, the parsing function group 700 can also store the element information in the CSCI storing unit 432.

That is, the CSCI storing unit 432 can store CSCI generated by the parsing function group 700 and data for decoding by a functional unit included in the decoding function group 800. It is evident that the CSCI can be divided into two parts, a CSCI

storing part for storing CSCI and a data storing part for data for decoding according to the type of data.

The connection control unit 434 controls the activation of the parsing function group 700 and/or each functional unit included in the decoding function group 800 in order to decode the bitstream ecoded by various standards. That is, the parsing function group 700 can be implemented in the decoding solution 430 as a type of parsing functional unit 510.

In other words, the connection control unit 434 controls each functional unit included in the decoding processing unit 550 to be activated per the hierarchical structure unit by referring to the connection control information. This is because input data for the functional unit in the lower layer can be result data processed by the functional unit of the upper layer.

As described above, the decoding solution 430 can include a work memory for performing predetermined process by loading at least one functional unit under the control of the connection control unit 434.

As described above, the connection control unit 434 can use the connection control information inputted from the decoder implementation unit 420. The connection control information can be generated using FL 610, F-RT 620, FU-CSCIT 630, CSCIT 640, and DHT 650 to process the activation of each functional unit and the data econded by the each functional unit to video data and output the result.

The encoded data can be converted to video data by systematical operation of 2 or more of functional units and the result can be output as illustrated in FIG. 8. The order of operations by the functional units to convert into video data may be different according to each encoding standard and this is apparent to those skilled in the art. Therefore, hereinafter a systemic relationship between partial decoder descriptions used for generation of the connection control information for systemic operation control of 2 or more of functional units, will be described.

First, the decoding hieracrchy table (DHT) 650 is a partial decoder description describing information for codec hierarchical structure and lower layers per each hierarchy as shown in FIG. 16 and Table 1.

Table 1. DHT(Decoding Hierarchy Table)

Index	Name	Children Hierarchy	Note
H0	VS	H1	Video Session
H1	GOP	H2, H3	Group of Picture
H2	VOP-1	H5	Video Object Plane : Type 1
H3	VOP-2	H4	Video Object Plane : Type 2
H4	MB-1	H6	
H5	MB-2	H7	

H6	B-1	None	
H7	B-2	None	
H8	...	...	...
...	...	...	...

The DHT 650, as shown in Table 1, may include an index catagory which is a layer number, a name catagory which is a name of each layer, a children hierarchy category which is a lower layer of a corresponding layer, etc. A number of layers or a number of children hierarchies may be different according to codec. For example, when there are further VOP-3, MB-3, etc., it is apparent that layer number be correspondingly added, as illustrated in FIG 9.

The hierarchical structure of the DHT 650 of the present invention may combine a codec which is different from a base codec at any position in any layer by any method based on the base codec. In other words, several different codecs may be combined to a unified codec to provide a new codec which further allows video compression and decompression.

A main function of the DHT 650 allows various implementations including sequential/parallel, single/multiple threads, sw/hw hybrid decoding of a decoding

solution to compose using decoding descritptions by introducing the hierarchical structure to a given codec.

The hierarchical structure described with the DHT 650 in Table 1 is commonly used in the syntax parsing and decoding process. It is also apparent that another hierarchical structure can exist to be independently applied in the syntax parsing or in the decoding process.

Then, the FU list(FL) 610, as shown in Table 2, is a partial decoder description describing information for a list of each functional unit in the decoding function group 800, volume of input/output data, etc.

Table 2. FL(FU List)

FU ID	FU name	input CSCI	output CSCI
D0092	FU-A	2	0
D0098	FU-B	2	0
...	...	...	...

The FL 610 may further include a name of a dedicated buffer region where input data is written (or recording address of a pertinent data or address of buffer memory where a pertinent data is written) and a name of a dedicated buffer region

where output data is to be written by a pertinent functional unit (or recording address of a pertinent data or address of buffer memory where a pertinent data is written). Each functional unit can read input data and write processed output data by using the FL 610. Input data and output data of the parsing functional unit 510, which generates element information, is not written in the FL 610. This is because the parsing function group 700 writes element information generated and stored in a determined position by using syntax parsing control information which can be data corresponding to SET 660.

The FL 610 may include FU ID which is an identification number of each functional unit in the tool-box 510, FU name which is a name of each functional unit, input CSCI representing volume of input data, and output CSCI representing volume of result data(output data), etc. An arrangement order of the functional units in the FL 610 corresponds to FU arrangement order or index number of the FU-CSCIT 630. A name of each functional unit, which is described by referring to FIG. 8, may be written in a column of FU name.

A particular functional unit which is activated by the connection control unit 434 reads necessary data from the data storing unit 524 by using the connection control information corresponding to FU-CSCIT 630, CSCIT 640, etc. and performs a predetermined process, and generates output data. The generated output data may be stored in the CSCI storing unit 432 and the stored data in the CSCI storing unit 432 can be used as input data for a following process by a functional unit. Here, the functional

unit is included in the decoding processing unit 550 and means a series of processings (e.g. ability, algorithm or function, etc.) which performs a predetermined process for input data and generates output data.

When the decoding processing unit 320 only uses one standard to decode encoded video data included in the conventional bitstream 316, the FL 610 may include only information for the functional units which perform the processing corresponding to that standard.

However, when a pertinent video data is encoded by using a plurality of standards (e.g., when encoding standard is applied with a plurality of frame units), information of the functional units according to a plurality of strandards is required to decode the corresponding encoded video data. Thus, in this case, the FL 610 should include information of functional units according to a plurality of standards required for decoding the encoded video data from all functional units according to the corresponding plurality of standards.

Even though video data appies different encoding standards by a plurality of frame units, if a plurality of conventional bitstreams 105 and extended bitstreams 305 can be generated and provided by each applied encoding standard, each FL 610 includes only information of functional units corresponding to each standard.

The FL 610 can be described not only by a textual description method or a binary description method (bit converted binary code format) but also by key minimum data of the table by using a pseudo script language.

Then, FU - Rule Table (F-RT) 620 provides connection information of functional units to be used for decoding inputted conventional bitstream 316. F-RT 620, as shown Table 3 below, is composed that an upper layer calls a lower layer by using the hierarchical structure specified in the DHT 650.

Table 3. F-RT(FU - Rule Table)

DecodingHierarchy	Children(FU/FH)	LOOP	LOOPCondition	Input	Description
FH0	FU1	NO	-	FH0	
	FH1	YES	CH1.C0==1	FU1	FH1[0..N-1] e.g., FH1[i] executes if CH1[i].C0 = 1.
	FH1	YES	CH1.C1==1	FH1[i-1]	
FH1	FU2	No	-	FH1	
	FH2	NO	-	FU2	
...	...	.	...	.	...

F-RT 620, as shown in Table 3, includes decoding hierarchy representing each hierarchical structure, children representing functional units to be activated or lower layers to call, loop representing determination of loop operation, loop condition representing loop conditions when loop operation is required, input representing decording hierarchy or children as input information of each functional unit or a lower layer(or activation order), etc.

As shown in Table 3, a FH0 layer will be activated very first and since a FH1 layer uses result data of FU1 as input data, it will be then activated. Or after the FH1 layer and the FH0 layer can be activated at the same time, processing operaton can be reserved till the result data of FU1 is stored in the CSCI storing unit 432.

Each layer is generated as an object with guaranteed independence based on instance/object concept during the calling. In other words, a pluraliry of FH1 is generated such as FH1[0], FH1[1], FH1[2], ..., FH1[n] and this structure may be applied to another layer(e.g., FH0, FH2, FH3, etc.). A dedicated buffer space per each layer can be assigned. As illustrated in FIG. 12 and FIG. 13, CSCI or data for the processing by functional units in a particular hierarchical structure or a pertinent hierarchical structure is stored in each dedicated buffer space.

The F-RT 620 can be described not only by a textual description method or a binary description method (bit converted binary code format) but also by key minimum data of the partial decoder description by using a pseudo script language.

Then, FU-CSCIT(FU CSCI Table) 630 is a partial decoder description for connecting element information stored in the CSCI storing unit 432 with element information(input CSCI) requested by each functional unit.

Table 4. FU-CSCIT(FU CSCI Table)

Index No.	CSCI information	CSCI table element
The num of interface set : 1		
1	Quantiser_scale	CH2.C6
	The number of MB	CH2.C1
The num of interface set : 2		
2	CBP_MPEG2	CH3.C1
	Ac_pred_flag_MPEG2	CH3.C6
	CBP_MPEG4	CH4.C2
	Ac_pred_flag_MPEG4	CH4.C5

As shown in Table 4, the FU-CSCIT 630 includes index number which is 1:1 mapping to FU ID of the FL 610, CSCI information, and CSCI table element of the CSCIT 640 for mapping. An arrangement order of the functional units of the FL 610 is mapped with that of the index number of the FU-CSCIT 630. In other words, FU-A

which is the first functional unit of Table 2 is mapped with index no. 1 which is the first index of the FU-CSCIT 630.

The FU-CSCIT 630 further includes information referring to how many interface sets are per index number. It means that a number of interface sets is a number of mappings. In other words, D0092 of the FL 610 is assigned to have 1 interface set by the FU-CSCIT 630 and thus uses only CH2.C6 and CH2.C1 as element information. However, D0098 of the FL 610 has 2 interface sets and thus uses CH3.C1 and CH3.C6 as element information in one case and CH4.C2 and CH4.C5 in the other case. Each functional unit performs a predetermined processing by reading element information from the CSCI storing unit 432 and stores the generated output data in the CSCI storing unit 432.

The reason to assign a number of interface sets per each index number in the FU-CSCIT 630 is that a functional unit can perform a different processing operation according to a different data flow. In other words, as shown in FIG. 11, if FU-A is a function which can be used in both MPEG-2 and MPEG-4, element information for processing encoded data by the MPEG-2 standard can be different from element information for processing encoded data by the MPEG-4 standard. Therefore, 2 or more of interface sets are required in order to classify element information requsted for encoded data processing according to each standard.

The FU-CSCIT 630 can be described not only by a textual description method or a binary description method (bit converted binary code form) but also by key minimum data of the partial decoder description by using a pseudo script language.

The CSCIT 640, as shown in Table 5 below, describes details of the element information (e.g., CSCI) as the result information that the parsing function group 700 processes by using the SET 660, S-RT 670, etc. In other words, the CSCIT 522 has all meaningful data (i.e., element information) that is processed from the conventional bitstream 316, stored in the CSCI storing unit 432 and used by the decoding function group 800.

Table 5. CSCIT

Decoding Hierarchy	Index	Element Name	Type	Note
CH0	CH0.C1	Profile and layer indication	Integer	
	CH0.C2	User data	Array	An array of arbitrary length of user data.
	CH0.C3	Visual object VerID	Integer	
	...			
CH1	CH1.C1	Time code (Hours)	Integer	

	CH1.C2	Time code (Minutes)	Integer	
	...			
CH2	CH2.C1	Macro-block type	Integer	
	...			

As shown in Table 5, the CSCIT 640 includes decoding hierarchy of each element information, an index as an identifier which is the identity number of the pertinent element information, a name of the pertinent element information, a type for designating the data structural characteristic of the pertinent element information (e.g., whether the pertinent element information is an integer type or an array type, etc.). The CSCI information composed of layers may be used by a type of several arrays, if necessary, during the decoding.

The CSCIT 640 can be described not only by a textual description method or a binary description method (bit converted binary code format) but also by key minimum data of the table by using a pseudo script language.

Hereinafter, the CSCIT 640, the DHT 650, the SET 660, the S-RT 670, and the DVT 680, which are used when the parsing function group 700 extracts and generates element information from the conventional bitstream 316 and stores the result in the

CSCIT storing unit 432, will be described. However, the CSCIT 640 and the DHT 650 have been explained above and their details will be thus omitted. 2 Or more of functional units as shown in Table 6 operate systematically each other to generate element information and store the result in the CSCI storing unit 432. A type of element information can be different according to each encoding standard and it is apparent to those skilled in the art. A sysmetic relationship between tables of the second decoder descriptions included in a extended bitstream, which 2 or more of functional units or the parsing function group 700, which is a combined functional unit, use to generate element information and store the result in the storing unit 520, will be explained.

First, the SET 660 is a partial decoder description that consists of information related to the syntax of the inputted conventional bitstream as shown in Table 6.

Table 6. SET(Syntax Element Table)

Index	Element Name	Input	Process by SET-PROC
S0	Visual object sequence start code	32 bit	READ 32 B; (IBS==HEX:1B0)>>;
S1	Profile and layer indication	8 bit	READ 8>>;
S2	Visual object sequence end code	32 bit	READ 32 B; ((IBS==HEX:1B1) ||

			(EOF))>>;
	...	...	...

The SET 660, as shown in Table 6, includes index of each syntax, element name, input data, and process by SET-PROC. Here, the index is an identifier S identifying each syntax used for the S-RT 670.

The element name can be named according to the meaning or function of the syntax. The input data refers to nominal bit length of data inputted at a time in the conventional bitstream. The SET-process describes which processing operation is undergone after receiving each bitstream syntax to generate the element information as the output data. Here, process by SET-PROC "READ 32 B:(IBS==HEX:IB0)>>:" corresponding to the index S0 means that if "(IBS==HEX:IB0)" is satisfied with reading 32 bits, output(>>) the corresponding information. Output data is element information (that is, CSCI information (C), e.g., CH0.C1, etc.), which is provided by the S-RT 670 and stored in the CSCI storing unit 432.

The SET 660 can be described not only by a textual description method or a binary description method (bit converted binary code format) but also by key minimum data of the partial decoder description by using a pseudo script language.

Then, the S-RT 670 refers to layers and connection information between each syntax in the conventional bitstream 316 as shown in Table 7. In other words, the S-RT

670 has information indicating for an upper layer to call syntax of a lower layer and to move to next syntax. The parsing function group 700 uses the R-ST 460 to read the conventional bitstream or to define the sequence of storing and/or updating the element information in the CSCI storing unit 432. As illustrated in FIG. 17, the parsing function group 700 is composed of several threads and performs syntax parsing per layer by referring to syntax parsing control information corresponding to the S-RT 670 since an upper thread calls a lower thread.

Table 7. S-RT(Syntax Rule Table)

Decoding Hierarchy	Index	Syntax #ID	CSCI	Branch Information	Note
SH0	SH0.SR1	S0	CH0.C1, CH0.C0	1: (CH0.C0==1) GO SR2; 2: GO ERR;	VS Start Code
	SH0.SR2	S1	...	1: GO SR3;	

	SH0.SR4	S5		While (conditions) {
				CALL
				SH1[CH0.C6];
				CH0.C6++;
				}	

SH1	SH1.SR1	S6		1: (CH0.C0==1) GO SR2; 2: GO ERR;	GOP Start Code
	SH1.SR2	S7		1: ([CH1.C13]==1 && [CH1.C14]==1) SR3; 2: GO SR4;	
SH2	...	...	...	...	...
...	...	...	...	...	...

As illustrated in Table 7, the S-RT 670 includes decoding hierarchy of each syntax, index R, syntax #ID, input data (CSCI), and branch information.

The index R identifies each connection information(rule). Since the index S of

syntax designates syntax that a particular connection index will process in each layer, the parsing function group 510 performs a designated process of the pertinent syntax by using the syntax parsing control information corresponding to SET 660.

The input data refers to a list of element information to be used for condition determination for controlling the connection of the pertinent connection index. The process result by the SET 660 is stored in the name of input data corresponding to syntax ID as described earlier.

The branch information refers to condition determining algorithm determining which connection index will be processed in next turn. The branch information can directly determine the sequence and contents of data to be read. If the number of branch is 1(e.g., in case that syntax #ID is S1), no input data is provided.

The S-RT 670 can be described not only by a textual description method or a binary description method (bit converted binary code format) but also by key minimum data of the partial decoder description by using a pseudo script language.

Finally, the DVT 680 is a partial decoder description describing Huffman table information used by each encoder/decoder as shown in Table 8. Entropy coding is performed during each encoding in MPEG-1/2/4/AVC. At this time, the Huffman coding method is mainly used and the used information is the Huffman table. There should be provided Huffman table information to be used in a pertinent decoder during each decoding process to implement a unified codec. Therefore, decoder descriptions of

the present invention include Huffman table information corresponding to each syntax during the syntax parsing. If Huffman table information corresponding to each standard is pre-stored in the description storing unit 410, the DVT 680 transmission may be omitted or only codec number 1020 and profile and level number 1030 can be included for designating Huffman table information.

Table 8. DVT(Default Value Table)

name	value	code
MCBPC	0	1
MCBPC	1	001
MCBPC	2	010
...	...	...
MCBPC	9	NULL
CBPY	0	0011
CBPY	1	00101
...	...	...
CBPY	17	000001
CBPY	18	NULL

intraDCy	0	011
intraDCy	1	11
…	…	…
intraDCy	12	00000000001
intraDCy	13	NULL
intraDCc	0	11
intraDCc	1	10
…	…	…
intraDCc	13	NULL
DCT intra	0	10
DCT intra	1	1111
…	…	…
DCT intra	101	000001011111
DCT intra	102	0000011
DCT intra	103	NULL
…	…	…

As shown in Table 8, the DVT 680 includes name of each Huffman table, actual value compressed and outputted by the Huffman decoding, and code value used when the compressed actual value is stored in the conventional bitstream 316. For example, if an actual value obtained with the compression of MCBPC value is 3, a code value 011 is written in the conventional bitstream 316 by the Huffman table mapping process (e.g., process by the SET 660). As another example, if VLD[1] is written in the process part of a particular index of the SET 660, a function of VLD is called, a code value can be obtained by reading the conventional bitstream 316 for the length predetermined by the corresponding function, and the corresponding actual value can be obtained by the Huffman table mapping process. Here, the used Huffman table is [1], which is the first table CBPY.

The DVT 680 can be described not only by a textual description method or a binary description method (bit converted binary code format) but also by key minimum data of the table by using a pseudo script language.

As an example, the DVT 680 can be the following textual description.

DVT{((0,1), (1,001), (2,010), (3,011), (4,0001), (5,000001), (6,000010), (7,000011), (8,000000001), (9,NULL)) ((0,0011), (1,00101), (2,00100), (3,1001), (4,00011),(5,0111), (6,000010), (7,1011), (8,00010), (9,000011), (10,0101), (11,1010), (12,0100), (13,1000), (14,0110), (15,11), (16,000000), (17,000001), (18,NULL)) ((0,011), (1,11), (2,10), (3,010), (4,001), (5,0001), (6,00001), (7,000001), (8,0000001),

(9,00000001), (10,000000001), (11,0000000001), (12,00000000001), (13,NULL))

((0,11), (1,10), (2,01), (3,001), (4,0001), (5,00001), (6,000001), (7,0000001),

(8,00000001), (9,000000001), (10,0000000001), (11,00000000001),

(12,000000000001), (13,NULL))...

As another example, the DVT 680 can be the following binary description.

00000011111111111111111111111101111100001100011001000110100001

10110010000010011000000100110000010001100000110100100000000100000111110

01000011001010010100101001000010010010010100011001000111001100000100010

01011001010001000110000011001000101001001010001000100001001000001000

11000010110011000000000110000001000001111100011011000101100010100001100

10000110010010000010010100001001100000010011100000010100000000000101001

00000000101010000000000101011000000000010000011111000101100010100001000

10001100100100000100101000010011000000010...

The storage space for each partial decoder description can be reduced with binary description, the processing efficiency is improved, and the transmission time of the extended bitstream 305 including the decoder description is further reduced.

Hereinafter, the linking process between each table used by the parsing function group 700 and/or the connection control unit 434 will be described.

First, the connection control unit 434 calls functional units of the highest layer assigned by the F-RT 620. The called functional units stands by till the parsing function

group 700 stores the CSCI information and/or data requested for the processing in the CSCI storing unit 432.

The parsing function group 700 reads syntax corresponding to the highest layer from rules of th S-RT 670, recognizes VS start code by the SET 660, and stores output value (that is, element information, C0) in the name of input data (CSCI) provided in the S-RT 670 in the CSCI storing unit 432 by reading a bit set as input value in the SET 660 (that is, 32 bits determined as input value at the SET 660) corresponding to the conventional bitstream 316 and performing the SET-process. What is element information stored in the CSCI storing unit 432 or in which layer will it be used, etc. is described in the CSCIT 640. The parsing function group 700 replaces the element information C0 stored in the CSCI storing unit 432 for pertinent branch information of the S-RT 670 and performs the index processing corresponding to the result. For example, since branch information corresponding to index SH0.SH1 is 'CH0.C0= =1', if satisfied, it proceeds to index SR2 in the SH0 layer but if not satisfied, it is determined as error.

However, in the process where the parsing function group 700 generates element information by using the SET 660 and stores the result in the CSCI storing unit 432, if the VLD function is called (e.g., index S74 of the SET 660), the entropy encoding is performed by using the DVT 680. During this process, if element information is generated, it will be stored in the CSCI storing unit 432.

During the parsing function group 700 stores the element information and/or data in the storing unit 520, each activated functional unit in the decoding processing unit 550 verifies if all requested element information and/or data is stored to perform a predetermined process. A pertinent functional unit can verify if all is stored by referring to the FU-CSCIT 630. Further, it can recognize what the corresponding element information is by mapping with the CSCIT 640. The functional unit, that stores the requested element information and/or data, performs a predetermined process and stores the processed result data in the data storing unit 524.

As described above, the functional units may be successively activated or called by hierarchical structure unit. The activation of the hierarchical structure unit can be controlled by the connection control unit 434 which, for example, watches if the element information and/or data processing requested by a certain layer completes. If it is recognized that the element information and/or data requested to perform a predetermined process is stored, the called functional units read this and process. When each functional unit is connected sequentially or in parallel and performs the process, it provides same effect through such processes.

When layer(s) of the functional units reqested for converting into video data is(are) activated, the decoding processing unit 320 outputs video data corresponding to the inputted conventional bitstream 316.

FIG. 15 illustrates parallel processing of syntax parsing and data decoding in accordance with an embodiment of the present invention. As illustrated in FIG. 15, the syntax parsing by the parsing function group 700 and the data decoding by the decoding function group 800 can be independently performed by using the decoder description information. In other words, when the element information and/or data requested by each functional unit is stored for the data decoding, the corresponding functional unit can perform a predetermined process. Further, each functional unit in the decoding function group 800 performing the decoding processing may perform independent function if result data of a different functional unit is not used as input data.

Therefore, the decoder 300 of the present invention is not limited to that the decoding function group 800 can perform the operation after the syntax parsing is completed, but allows that the parsing function group 700 and the decoding function group 800 can perform independently. It further allows that each functional unit in the decoding function group 800 can independently operate.

Hereinafter, commands of each partial decoder description will be described in detail.

Commands used in each partial decoder description shown above are provided. Information (that is, table) can be composed for the syntax parsing of standards such as MPEG-2/MPEG-4/MPEG-4 AVC, etc.

Commands used for composing each partial decoder description can be READ,

SEEK, FLUSH, IF, WHILE, UNTIL, DO~WHILE, DO~UNTIL, BREAK, SET, STOP, PUSH, etc. It is not necessary that all commands should be used in each partial decoder description. It is apparent that certain commands per each partial decoder description can be selected and used. Each command is described briefly.

First, the READ can be used for reading a certain bit in the bitstream. For example, the READ can be described as "READ bits B>CSCI;", in which "bits" is a number of bits to read, "B" is byte-alignment flag, ">CSCI" is CSCI index to store. "B" and ">CSCI" are optional and if ">CSCI" is not designated, it is set to store only in a variable IBS.

The SEEK is an instruction to read a certain bit in the bitstream but not to move file point. The file point is a standard position during the operation such as reading a certain bit. Parameters of the SEEK instruction can be the same as the READ.

The FLUSH can be used for moving a file point as much as a certain bit in the bitstream and parameters thereof can be similar to the READ.

The IF can be used in a form of "IF (condition) { ~ } ELSE { ~ }" and is a command to provide branch according to a given condition.

The WHILE can be used in a form of "WHILE (condition) { ~ }" and to repeatedly perform assigned blocks during a given condition is true.

The UNTIL can be used in a form of "UNTIL (condition) { ~ }" and to repeatedly perform assigned blocks till a given condition is true.

The DO~WHILE can be used in a form of "DO { ~ } WHILE (condition)" and to perform a block prior to condition determination by modifying the WHILE.

The DO~UNTIL is used in a form of "DO { ~ } UNTIL (condition)" and to perform a block prior to condition determination by modifying the UNTIL.

The (~) (compute) is used in a form of, for example, "(C11=(V2+3));". In other words, all equations of the SET-PROC can be described within the parenthesis, operators, such as four arithmetical operations, assignment, comparison, addition/subtraction (++/--), bitwise operation, OR/AND operation, determining the use of CSCI, etc., can be used.

The BREAK can be used to secede from the closest loop structure.

The SET can be used to set a flag determining the use of the designated CSCI and the CSCI to designate the flag is arranged and can be sorted with a comma (e.g., SET C0, C2;).

The STOP can be used to stop the syntax element processing which is currently in process and to move next.

The PUSH can be used to add data at the last region where data is written in an array CSCI and added value is arranged and sorted by comma(e.g., PUSH C8 8, 16, 32;).

The GO can be used to go a designated position. For example, in case of GO SR#;;, it is an instruction to go SR# and GO RT is an instruction to return to the called position.

The HEX can be used to represent that a value after the HEX command is hexadecimal digits.

The RLD is an interface for a RLD function which is supported in MPEG-4, and can be used in a form of "RLD index, layer, run, islastrun, t#;". Here, index, layer, run and islastrun can be CSCI storing RLD return values or variables and t# represents Huffman table ID used in the RLD.

The VLD2 is a VLD function for MPEG-2 and can be used in a form of "VLD2 [t#] in > v1, v2, v3;". Here, t# represents Huffman table ID used in the VLD, in represents an inputted index value, and v1~v3 represent output values.

Finally, the VLD4 is a VLD function for MPEG-4 and can be used in a form of "VLD4 [t#] > CSCI;". Here, t# represents Huffman table ID used in the VLD and ">CSCI" represents a CSCI index to be stored. If ">CSCI" is not designated as option, it is stored only in a variable IBS.

FIG. 16 is a schematic block diagram illustrating the structure of a extended bitstream in accordance with a first embodiment of the present invention, FIG. 17 is a schematic block diagram illustrating the structure of a extended bitstream in accordance with a second embodiment of the present invention, FIG. 18 is a schematic block

diagram illustrating the structure of a extended bitstream in accordance with a third embodiment of the present invention, and FIG. 19 is a schematic block diagram illustrating the structure of a extended bitstream in accordance with a fourth embodiment of the present invention.

As illustrated in FIG. 16 to FIG. 19, the decoder description included in the extended bitstream 305 can be composed to include not partial decoder description information but only standard information (no table), to include all table information (full tables), or to include a part of table information (partial tables). In order to classify each of them, the decoder description information may include stream indentifier (SI) information and the SI information is shown below.

Table 9. Stream Identifier

SI	Decoder description
00	No table
01	Full tables
10	Partial tables

As illustrated in FIG. 16, the extended bitstream 305 is the decoder description and may include SI 1610 (that is, 00) which has no table information, codec number 1620, and profile and level number 1630.

This is a case that the partial decoder description information is pre-stored in the description storing unit 410 and the partial decoder description information is not transmitted. Even though a pertinent conventional bitstream 316 sends basic information for which codec and profile and level are to be used, the decoding processing unit 320 can decode by using the designated partial decoder descriptions.

The SET 660, the CSCIT 640, the FL 610, the FU-CSCIT 630, the DVT 680, etc. can be described per the applied standard (that is, codec), the F-RT 620, S-RT 670, etc. can be described per profile of each applied standard (see Table 10 and Table 11).

Table 10. Table classification per codec

Standard	Tables				
MPEG-1	SET #1	FL #1	FU-CSCIT #1	CSCIT #1	DVT #1
MPEG-2	SET #2	FL #2	FU-CSCIT #2	CSCIT #2	DVT #2
MPEG-4	SET #3	FL #3	FU-CSCIT #3	CSCIT #3	DVT #3
AVC	SET #4	FL #4	FU-CSCIT #4	CSCIT #4	DVT #4

Table 11. Table classification per Profile and level

SI	Tables	
MPEG-1	F-RT #1-1	S-RT #1-1
MPEG-2 MP	F-RT #2-1	S-RT #2-1
MPEG-4 SP	F-RT #3-1	S-RT #3-1
MPEG-4 ASP	F-RT #3-2	S-RT #3-2
AVC BP	F-RT #4-1	S-RT #4-1

In case of MPEG-4 SP, the decoding method will be explained with using SET#3, FL#3, CSCIT#3, FU-CSCIT#3, DVT#3, F-RT#3-1, and S-RT#3-1 and when codec #3 and profile and level #2 are designated and transmitted, the decoding processing unit 320 can perform the decoding process by referring to the corresponding tables.

As illustrated in FIG. 17, the extended bitstream 305 is the decoder description and may include the aforementioned partial decoder description information. Referring to table 9, the SI 810 is set to be 01. Each table may include a table identifier (TI) 1710, a table start code (TS code) 1720, a table decription (TD) 1730, and a table end code (TE code) 1740. An order of the table identifier (TI) 1710 and table start code (TS code) 1720 may be exchanged and the table decription (TD) 1730 may be described by a description method. Of course, an order of each table can be changed.

As illustrated in FIG. 18, the extended bitstream 305 is the decoder description and may include the aforementioned table information and codec number corresponding to a part of the table information. In this case, the SI 1610 is set to be 10 referring to table 9. Since types of the partial decoder description information are not unified, a configuration identifier 1010 can be included at the end of the table identifier 910 in order to determine which type the corresponding partial decoder description information is composed of.

Further, as illustrated in FIG. 19, the extended bitstream 305 is the decoder description for the table information and updating information. The decoder description for the partial decoder description information can be one of the decoder descriptions explained by referring to pertinent figures is set to be a pertinent value. The updating information may include a revision start code (RS code) 1920 and revision 1930.

The revision 1930 can be contents to insert, delete, or update rules of a particular table. A form thereof may be 'insert index into table-name (...);', 'delete index from table-name;', or 'update index in table-name(...);', etc. For example, when S100 is inserted to SET#4, the revision 1130 may be in a form of "insert S100 into SET#4 ("READ 1;IF(IBS= =1){SET C31;}");".

During the description decoder 420 reads the revision 1930 and performs the decoding of the corresponding extended bitstream 305, partial decoder descriptions of revised contents are stored in the description storing unit 410. However, when the decoding completes, the partial decoder descriptions stored in the description storing unit 410 should be restored to the original state. Determining if the decoding completes, it can be recognized when the decoding processing unit 320 or trigger provides completion notice to the description decoder 320 or the description decoder 320 detects the completion of the decoding processing unit 320.

As described above, according to the present invention, existing profiles can be used by using the functional units provided in the conventional standard (that is, codec), a new decoder can be implemented by using the conventional functional units, and a new decoder can be implemented by using new functional units. In other words, various decoders can be implemented without any limitation.

Only, when new functional units are added in the toolbox 415, the algorithm corresponding to those functional units is added and the corresponding information is

added to the FL 610. In this case, a compile process for the algorithm may be needed additionally.

To realize unified codec, each element must be systematically controlled such that a bitstream can be decoded with a decoding method corresponding to a pertinent encoding method by parsing the bitstream compressed by various encoding methods.

In this case, the pertinent bitstream can be a bitstream of various shapes mixed with various standards (codecs) or another bitstream of various shapes generated by various encoding methods in one standard. Also, to support various encoding / decoding methods, it is necessary that various functions can be separated into independent units and desired units can be selected among the independent units to make them one codec (encoder and decoder).

Also, described above, the present invention can systemically connect and control each functional unit by use of the same information analyzing method regardless of an encoding method encoded with a bitstream by providing together decoder description information.

Also, the present invention can deal with the change or new addition of the syntax of a bitstream by only amending pertinent information or inserting the additional information. The present invention also can set the connection relationship of the functional units of the decoding function group 550 of the pertinent decoder and

functional units by allowing a user to select a desired function of bitstream-level, frame-level and macro block-level (MB level) and forms the F-RT 620.

FIG. 20 is a schematic block diagram illustrating the structure of a extended bitstream in accordance with a fifth embodiment of the present invention, FIG. 21 is a schematic block diagram illustrating the structure of a extended bitstream in accordance with a sixth embodiment of the present invention, FIG. 22 is a schematic block diagram illustrating the structure of a extended bitstream in accordance with a seventh embodiment of the present invention, and FIG. 23 is a schematic block diagram illustrating the structure of a extended bitstream in accordance with an eigthth embodiment of the present invention.

The extended bitstream 305 according to the present invention is composed of the decoder description region and the conventional bitstream 316. It is apparent to those skilled in the art that the conventional bitstream 316 is composed of encoded video data(or/and encoded audio data).

Here, the encoded decoder description 313 can be formed in a different structure according to codec characteristics to be applied for the decoding of the conventional bitstream 316. In other words, first, a first decoder description structure may be applied when one codec standardized before is used.

Second, when a part of one codec standardized before is changed (that is, some tables of partial decoder descriptions uses partial decoder description contents

corresponding to the pertinent codec as they are and another part of the partial decoder descriptions changes and uses), a second decoder description structure may be applied.

Third, when table information of a plurality of codecs standardized before is processed and used (that is, a part of the above-described partial decoder descriptions selectively uses partial decoder description contents of the conventional plurality of codecs and another part of those partial decoder descriptions is changed and used), a third decoder description structure may be applied.

Fourth, when a new codec which is not standardized before is used (that is, the partial decoder descriptions composed with new contents all are transmitted), a fourth decoder description structure may be applied.

The described four decoder description structures can be classified according to codec type information. For example, in case of the first decoder description structure, "codec_type = 0" is set, in case of the second decoder description structure, "codec_type = 1" may be set, in case of the third decoder description structure, "codec_type = 2" may be set, and in case of the fourth decoder description structure, "codec_type = 3" may be set.

The first decoder description structure is illustrated in FIG. 20. According to the first decoder description structure, the decoder description region may be composed of codec type 2010, codec number 2020, and profile and lever number 2030. In other words, according to the first decoder description structure, information for the codec to

be applied is only described in the decoder description region. It is apprant that a size of each field may be balanced according to a size of information to be described even though each field is illustrated as 8 bits in FIG. 20.

The codec type is set as 0 (that is, codec_type=0) and it means that one codec among various conventional standardized codecs is used as it is.

The second decoder description structure is illustrated in FIG. 21.

According to the second decoder description structure illustrated in FIG. 21, the encoded decoder description 313 may be composed of codec type 2010, codec number 2020, profile and lever number 2030, and table description 2110. In other words, according to the second decoder description structure, information for the codec to be applied and changed contents from the partial decoder descriptions will be described in the encoded decoder description 313. Here, the table description is independently provided to each partial decoder description. That is, there can be many partial decoder descriptions in the encoded decoder description 313.

Each table description 2110 may include table start code 2120, table identifier 2130, table type 2140, contents 2145, and table end code 2140. A size of each field may be balanced, if necessary. As described below, the contents 2145 may be omitted or included according to the table type 2140.

For example, if the table type value is 0, it may be recognized that a conventional partial decoder description (that is, a partial decoder description

recognized by codec type 2010, codec number 2020, profile and level number 2030 and tabe identifier 2130) is to be applied without any changing. Here, the contents 2145 may be omitted.

If the table type value is 1, it may be recognized that a part of a conventional table (that is, a partial decoder description recognized by codec type 2010, codec number 2020, profile and level number 2030 and tabe identifier 2130) is to be changed and applied. In this case, the changed content (e.e., update command) may be described in the contents 2145. The changed content (e.g., update command) including a command such as update, insert, and/or delete, etc. may be information to revise table contents of the index correspond to the pertinent table.

If the table type value is 2, it may be recognized that a conventional table (that is, a partial decoder description recognized by the codec type 2010, the codec number 2020, the profile and leverl number 1154, and the tabe identifier 2130) is to be completely changed (that is, change into the content defined in the content field 2145) and applied. In this case, the changed content (e.g., content to newly define the pertinent table such as new command, etc.) may be described in the content field 2145.

The third decoder description structure is illustrated in FIG. 22. According to the third decoder description structure illustrated in FIG. 22, the encoded decoder description 313 may be composed of a codec type 2010 and a table description 2110. In other words, according to the third decoder description structure, information about

codec to be applied and content to be changed among the partial decoder descriptions may be described in the decoder description region. Here, the table description is independently provided to each partial decoder description. That is, there can be more than one partial decoder description in the encoded decoder description 313.

Each table description 2110 may include a table start code 2120, a table identifier 2130, a table type 2140, a content 2145, and a table end code 2150. A size of each field may be balanced, if necessary.

For example, if the table type value is 0, it may be recognized that a conventional partial decoder description (that is, a partial decoder description recognized by the codec number 2020, the profile and level number 2030 and the tabe identifier 2130) is to be applied without any changing. In other words, the codec number(codec_num) 2020 and the profile and level number(profile_layer_num) 2030 corresponding to a partial decoder description to be applied are described in the content field 2145.

However, if the table type 2140 value is 1, it may be recognized that a part of a conventional partial decoder description (that is, a partial decoder description recognized by the codec number 2020, the profile and level number 2030 and the tabe identifier 2130) is to be changed and applied. In this case, the codec number(codec_num) 2020 and the profile and level number(profile_layer_num) 2030

corresponding to a partial decoder description to be applied are described in the content field 2145 and the changed contents (e.g., update command, etc) is described in the changed content field .

However, if the table type 2140 value is 2, it may be recognized that a conventional partial decoder description (that is, a partial decoder description recognized by the tabe identifier 2130) is to be completely changed (that is, change to contents defined in the content field 2145) and applied. In this case, the changed content (e.g., content to newly define the pertinent table such as new command, etc.) may be described in the content field 2145. In other words, as described above, if the table type 2140 is 0 or 1, a certain codec is used as it is or a part of tables is changed, so that codec information (e.g., the codec number 2020, the profile and level number 2030) is needed. On the other hand, if the table type is 2, new table information is defined, so that codec information is not needed.

The fourth decoder description structure is illustrated in FIG. 23. According to the fourth decoder description structure illustrated in FIG. 23, the decoder description region may be composed of a codec type 2010 and a table description 2110. In other words, according to the third decoder description structure, information about codec to be applied and content to be changed among the partial decoder descriptions may be described in the encoded decoder description 313. Here, the table description is independently provided to each partial decoder description.

Each table description 2110 may include a table start code 2120, a table identifier 2130, a table type 2140, a content 2145, and a table end code 2150. A size of each field may be balanced, if necessary.

For example, if the table type 2140 is a predetermined value (e.g., 2), information (e.g., content to newly define a pertinent partial decoder description such as new command) to describe a new partial decoder description corresponding to the table indetifier 2130 is described in the content field 2145. As described above, if the codec type is 3, it is recognized that the decoding is performed by using new partial decoder descriptions, the table bype 2140 is designated with 1 or the table type 2140 can be omitted.

Hereinafter, the syntax structure of the encoded decoder description 313 and the syntax structure of each field are shown as the following tables.

Table 12. Decoder description

Decoder_Description() {	No. of bits
codec_type	8
If ((codec_type==0x00) || (codec_type==0x01)) {	
Codec_Description()	
}	

If (codec_type!=0x00) {	
do {	
Table_Description()	
} while (next_bits()==table_idetifier)	
}	
}	

Table 13. Codec Description

Codec_Description() {	No. of bits
codec_num	8
profile_layer_num	8
}	

Table 14. Table Description

Table_Description() {	No. of bits
table_start_code	24
table_identifier	4
table_type	4

if ((table_type ==’0000’) || (table_type ==’0001’)) {	
if (codec_type==0x02)	
Codec_Description()	
if (table_type ==’0001’)	
Update_Description()	
}	
if (table_type ==’0010’) {	
New_Description()	
}	
table_end_code	24
}	

Table 15. Update Description

Update_Description() {	No. of bits	Mnemonic
Update_Command		vlclbf
}		

Table 16. New Description

New_Description() {	No. of bits	Mnemonic
New_Command		vlclbf
}		

Hereinafter, semantics of the decoder description is described with the following tables.

Table 17. Decoder description

codec_type	Meaning
0x00	A profile@layer of an existing MPEG standard
0x01	Some parts of the existing one profile@layer changed
0x02	Some parts of the existing multiple profile@layer changed
0x03	A new decoding solution
0x04-0xFF	RESERVED

Here, the codec type is a 8 bit code which can be information to identify the codec type.

Table 18. Codec Description

codec_num	MPEG standards and others
01	MPEG-1
02	MPEG-2
03	MPEG-4 Part 2
04	MPEG-4 Part 10 (AVC)
05-FF	RESERVED

Here, the codec number is a 8 bit code which can be information the codec code used. The profile and lever number is also a 8 bit code which can be information to indicate the profile and level number for the codec. The profile and level number may correspond to each MPEG standard.

Table 19. Table Description(Table Indetifier)

table_identifier	table name
0000	SET (Syntax Element Table)
0001	S-RT (Syntax Rule Table)
0010	CSCIT (CSCI Table)
0011	DVT (Default Value Table)
0100	FL (FU List)

0101	F-RT (FU Rule Table)
0110	FU-CSCIT (FU CSCI Table)
0111-1111	RESERVED

Here, the table start code may be 26 bit character sequesce 0xFFFFFE of hexadecimal digits which means the beginning of the table description. The table indetifier may a 4 bit code as shown in Table 12.

Table 20. Table Description(Table Type)

table_type	Meaning
0000	conventional table
0001	updated table
0010	new table
0011-1111	RESERVED

Here, the table type is a 4 bit code which is information to determine if a conventional table is to be kept or updated or a new table is to be generated. The table end cod may be 26 bit character sequesce 0xFFFFFF of hexadecimal digits which means the end of the table description.

Table 21. Update_Command for Intraction Set

Code	Instruction	Usage
00	UPDATE	UPDATE [index#] in [table#] [a record];
01	INSERT	INSERT into [table#] [a record];
10	DELETE	DELETE [index#] from [table#];
11	RESERVED	

Here, the index nuber may be a 4 bit character sequence indicating a certain table index and the table number may be a 32 bit character sequence which is a table identifier.

Table 22. New_Command for Instruction Set

Code	Instruction	Usage
00000001	READ	READ bits B > CSCI;
00000010	SEEK	SEEK bits B > CSCI;
00000011	FLUSH	FLUSH bits B;
00000100	IF	IF (condition) { ~ } ELSE { ~ }
00000101	WHILE	WHILE (condition) { ~ }
00000110	UNTIL	UNTIL (condition) { ~ }

00000111~0	DO~WHILE	DO { ~ } WHILE (condition)
00000111~1	DO~UNTIL	DO { ~ } UNTIL (condition)
00001000	(~) (compute)	(.....)
00001001	BREAK	BREAK;
00001010	SET	SET CSCI, CSCI;
00001011	STOP	STOP;
00001100	PUSH	PUSH CSCI Value, Value ;
00001101	RLD	RLD index, layer, run, islastrun, t#;
00010010	VLD2	VLD2 [T#] in > v1, v2, v3;
00010100	VLD4	VLD4 [T#] > CSCI;

Here, the bit is a random value of 3 to 34 bits representing a number of bits requested and B is one bit character sequence representing byte alignment. ">" is one bit character sequence for printing output of the left side and VLD2(for MPEG-2) and VLD4(for MPEG-4) are functions for the entropy coding.

5

FIG. 24 is a block diagram illustrating an encoder in accordance with an embodiment of the present invention.

An encoder 2400 according to the present invention further includes a extended bitstream generating and outputting unit 2410 compared to the conventional encoder

200 as described in FIG. 2. The extended bitstream generating and outputting unit 2410 generates decoder descriptions by using a control information list and reconfigurable connection for the conventional bitstream generation process generated by previous processings, input data for pertinent functional units, syntax information, and syntax connection information, etc. A extended bitstream 305 is generated by using the generated decoder descriptions and the conventional bitstream 316 and transmitted to the decoder 300. Since the method of generating decoder description is easily understood to those skilled in the art, detailed explanation thereof is omitted. It is also apparent to those skilled in the art that the encoder 2400 has a tool-box including 2 or more of functional units and one or more of bitstreams are generated according to one or more of encoding standards by sequential combinations or systemical combinations of these functional units.

Further, the variable length encoding unit 230 merely refers to an element (e.g. encoding unit) finally performing an encoding to generate the conventional bitstream 316 in the encoder 2400, but the present invention is not limited to the variable length encoding unit 235. Also this does not restrict the scope of claims of the present invention.

FIG. 24 assumes the case that the extended bitstream 305 generated by using the decoder description information and the conventional bitstream 316 is provided to the decoder.

However, as described above, the decoder description information can be provided to the decoder 300 by an additional data format or a bitstream format. In this case, it is evidently possible that the encoded decoder description generating and outputting unit can be provided at the end of the variable length encoding unit 235 and the encoded decoder description generating and outputting unit is independently provided from the conventional encoding unit 200 to provide independently-generated information to the decoder 300.

In the present description, the unified codec device and method in accordance with the present invention is mainly described based on the decoder. Considering that the relationship between the decoder and encoder is well-known to those skilled in the art and the encoder can easily be structured with the description related to the decoder only, it is evident that the present invention is not limited to the decoder.

As described above, the unified codec device and method of the present invention makes it easy to analyze a syntax element in a standard (or codec) or between different standards or to control the connection of pertinent functional units. In other words, in the present invention, there is no problem when changing the sequence of syntax elements in the bitstream generated according to a particular standard, inserting new elements and deleting a previous syntax element.

Also, in accordance with a prior art, there is a problem that the decoder is not able to properly decode a corresponding bitstream in the manipulation of the syntax

element. For example, if a structure of ABC bitstream information is changed to a structure of ACB, the decoder may not recognize the structure of ACB and thus impossible to perform a proper decoding. Similarly, for a structure of ABFC by adding F or another structure of AC by deleting B, the decoder may not recognize the structure and thus impossible to perform the proper decoding.

However, in the unified codec device and method of the present invention, since the decoder description included in the extended bitstream or the decoder description as independent data is provided to the decoder 300, the decoder 300 can perform the proper decoding operation.

It shall be obvious that although the above description related to a decoding device and method of the present invention is based on MPEG-4 AVC, MPEG-1, MPEG-2, MPEG-4 and other moving picture encoding/decoding standards can be applied without any restriction.

Beside that, it is obvious that the information included in each partial decoder description can be described by using not only the information the connection of functional units for performing the decoding by one standard and the information related to the processing operation requested for the pertinent functional unit but also the information for performing the decoding by a plurality of standards.

For example, it is assumed that an initial plurality of frames of encoded video data included in the universal bitstream are encoded by using MPEG-2, the following

plurality of frames are encoded by using MPEG-4 and the other frames are encoded by using MPEG-1. In this case, it is obvious that partial decoder description information included in the decoder description for decoding the encoded video data will be realized such that the functional units according to each standard included in the toolbox 415 can be organically coupled and operated, in order that each frame having different encoding methods can be decoded.

FIG. 25 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with a third embodiment of the present invention.

As illustrated in Fig. 25, a decoding processing unit 320 can include a description decoder 405, a description storing unit 410, a toolbox 415 and a decoder implementation unit 4200. the decoder implementation unit 4200 can include a connection control unit 4250 and CSCI storing unit 4300.

Although it is not illustrated, the decoder implementation unit 4200 can further include a work memory for loading the functional units by connection control unit 4250 and performing the predetermined process.

FIG. 26 is a schematic block diagram illustrating the structure of a decoding processing unit in accordance with the forth embodiment of the present invention.

As compared to the Fig. 25, the forth embodiment of the decoding processsing

unit 320 in Fig. 26 includes additionally a decoding solution 510. The decoding solution 510 can be a work memory for loading the functional units by the calling of connection control unit 4250 and performing the predetermined process.

The function and operation of the above each element of decoding processing unit 320 is same as descrived above.

The drawings and detailed description are only examples of the present invention, serve only for describing the present invention and by no means limit or restrict the spirit and scope of the present invention. Thus, any person of ordinary skill in the art shall understand that a large number of permutations and other equivalent embodiments are possible. The true scope of the present invention must be defined only by the spirit of the appended claims.

【Industrial Applicability】

The present invention is applicable to a video codec.

【CLAIMS】

【Claim 1】

A decoding device, comprising:

a decoder implementation unit generating and outputting Control Signal/Context Information (CSCI) control information and connection control information using partial decoder descriptions stored in a description storing unit;

a tool box including a plurality of functional units, each of the functional units realized to perform a predetermined process; and

a decoding solution loading the functional units in a hierarchical way and decoding a bitstream into video data by using the CSCI and the connection control information.

【Claim 2】

The decoding device of claim 1, further comprising a description decoder generating a decoder description by decoding an inputted encoded decoder description and storing a plurality of partial decoder descriptions in the description storing unit, the partial decoder description extracted from the decoder description.

【Claim 3】

The decoding device of claim 1, wherein the toolbox comprises:

at least one parsing functional unit performing syntax parsing of the bitstream; and

a plurality of decoding functional units for decoding the bitstream.

【Claim 4】

The decoding device of claim 1, wherein the decoder implementation unit comprises:

an FU check unit checking whether the functional units described in the partial decoder description exists in the toolbox; and

a data process unit generating the syntax parsing control information and the connection control information by using the partial decoder descriptions.

【Claim 5】

The decoding device of claim 1, wherein the decoding solution comprises:

a CSCI storing unit storing a plurality of elements information generated by syntax parsing the bitstream by processing of at least one functional unit; and

a connection control unit controlling an operation of each functional unit by selectively loading a plurality of functional units by referring to the syntax parsing control information and the connection control information.

【Claim 6】

The decoding device of claim 1, wherein the decoding solution comprises:

a CSCI storing unit storing a plurality of elements information generated by syntax parsing the bitstream by processing of at least one functional unit;

at least one parsing functional unit performing syntax parsing of the bitstream according to the syntax parsing control information; and

a connection control unit controlling an operation of each functional unit by selectively loading a plurality of functional units by referring to the CSCI control information and the connection control information,

whereas the toolbox includes a plurality of decoding functional units for decoding the bitstream.

【Claim 7】

The decoding device of claim 6, wherein the decoding solution comprises a work memory for loading and operating at least one functional uinit.

【Claim 8】

The decoding device of claim 2, further comprising a dividing unit deviding an extended bitstream, including the encoded decoder description and the bitstream, into the encoded bitstream and the bitstream.

【Claim 9】

The decoding device of claim 1, wherein the decoding solution generates and outputs the video data by loading functional units performing a process of a plurality of standards by referring to the connection control information in the case the encoded video data is encoded by the plurality of standards.

【Claim 10】

The decoding device of claim 1, wherein the connection control information includes information of hierarchical layers of functional units, and the decoding solution performs a calling operation between the layers for at least one functional unit.

【Claim 11】

The decoding device of claim 10, wherein the decoding solution controls generation and excution of objects for a higest layer, and functional units included in the higest layer and a calling unit for calling a layer assigned by the information of hierarchical layers are loaded by excecution of the object.

【Claim 12】

The decoding device of claim 11, wherein the calling unit loads a

corresponding layer when data, necessary for performing a predetermined process of at least one functional unit included in an assigned layer, are stored in a storing unit.

【Claim 13】

The decoding device of claim 12, wherein the loaed functional unit starts its operation when processing data for performing a predeterminded process are stored in the storing unit.

【Claim 14】

The decoding device of claim 12, wherein result data after performing a process of each functional unit are stored in the storing unit.

【Claim 15】

The decoding device of claim 10, wherein a hierarchical structure recognized by the information on hierarchical layers comprises at least one layer among a sequence layer, a GOP layer, a picture layer, a slice layer, a macro block layer, and a block layer, and a lower layer is called by a higher layer.

【Claim 16】

The decoding device of claim 12, wherein exclusive storing space for each

decoding functional unit is assigned in the storing unit.

【Claim 17】

The decoding device of Claim 12, wherein the storing unit comprises:

a control signal and context information(CSCI) storing unit storing control signal/context information(CSCI) generated by the parsing functional unit; and

a data storing unit storing at least one of data for decoding processing of data corresponding to the encoded video data generated by the parsing functional unit and process data processed by the decoding functional unit.

【Claim 18】

The decoding device of Claim 1, wherein the partial decoder descriptions stored in the description storing unit include:

a decoding hierarchy table (DHT) representing hierarchical structure for decoding the encoded video data and information of the lower layer per each hierarchy;

a syntax element table (SET) representing a process for generating information for bitstream syntax and element information corresponding to the bitstream syntax;

a syntax rule table (S-RT) designating names of connection information between the bitstream syntax, information for the lower layer per each hierarchy to call, and CSCI information which is result data generated by performing the SET process;

a control signal and context information table (CSCIT) representing detailed information of the CSCI information per each hierarchical structure;

an FU rule table (F-RT) representing an order of call or activation among at least one decoding functional units based on the hierarchical structure;

an FU list (FL) representing a list of the decoding functional units; and

an FU-control signal and context information table (FU-CSCIT) representing CSCI information for performing a process by the decoding functional units.

【Claim 19】

The decoding device of Claim 18, wherein a default value table (DVT) representing a relationship between actual values and code values during the entropy coding is further stored as a partial decoder description in the description storing unit.

【Claim 20】

The decoding device of Claim 19, wherein the syntax parsing control information is generated by using at least one of the SET, S-RT, DHT, CSCIT, and DVT.

【Claim 21】

The decoding device of Claim 18, wherein the connection control information is generated by using at least one of the FL, F-RT, DHT, and FU-CSCIT.

【Claim 22】

The decoding device of Claim 28, wherein the decoder description is configured to include at least one seperable part, and information for configuring the partial decoder description is inserted into the part.

【Claim 23】

The decoding device of Claim 22, wherein the partial decoder description comprises a codec number for decoding the bit-stream, and designating information corresponding to a profile and level number, and

the description decoder extracts n tables, corresponding to the designating information, from a plurality of partial decoder descriptions pre-stored in the description storing unit.

【Claim 24】

The decoding device of Claim 22, wherein m separable parts of the parts, m being a natural number, comprises a codec number related to a corresponding partial decoder description and designating information corresponding to a profile and level

number, and k separable parts comprises binary code information for forming a corresponding decoder description, and

the description decoder extracts m partial decoder descriptions, corresponding to the designating information, from a plurality of pre-stored partial decoder descriptions and generates k tables by using the binary code information and stores the generated k tables in the description storing unit.

【Claim 25】

A decoding method, comprising:

(a) generating and storing a plurality of partial decoder descriptions corresponding to an inputted decoder description;

(b) generating CSCI control information and connection control information using the partial decoder descriptions;

(c) storing a plurality of element information generated by syntax parsing a bitstream using CSCI control information; and

(d) decoding encoded video data of the bitstream using the connection control information and the element information and outputting the decoded video data.

【Claim 26】

The decoding method of Claim 25, wherein each of the steps (c) and (d) is

performed by the functional units loaded among the functional units in a hierarchical way in a toolbox by a connection control unit referring to the CSCI control information or the connection control information.

【Claim 27】

The decoding method of Claim 25, wherein the step (d) is performed repeatedly until the result of the process performed by the functional units operated by a connection control unit loading becomes the video data.

【Claim 28】

The decoding method of Claim 27, wherein the predetermined process of the respective functional units is realized to independently perform each function suggested by a plurality of standards for decoding the bit-stream.

【Claim 29】

The decoding method of Claim 26, wherein result data of a previous functional unit among hierarchically loaded function units is written in a buffer memory that is accessible by a following functional unit.

【Claim 30】

The decoding method of Claim 26, wherein the connection control unit provides result data from a previous functional unit among successively loaded functional units to a following functional unit as input data.

【Claim 31】

The decoding method of Claim 25, wherein the partial decoder descriptions stored in the description storing unit include:

a decoding hierarchy table (DHT) representing a hierarchical structure for decoding encoded video data and information on a lower layer per each hierarchy;

a syntax element table (SET) representing a process for generating information for bitstream syntax and element information corresponding to the bitstream syntax;

a syntax rule table (S-RT) designating names of connection information between the bitstream syntax, information on a lower layer per each hierarchy to call, and CSCI information in which result data generated by performing the SET process is to be stored;

a control signal and context information table (CSCIT) representing detailed information of the CSCI information per each hierarchical structure;

an FU rule table (F-RT) representing an order of call or activation between at least one decoding functional unit based on the hierarchical structure;

an FU list (FL) representing a list of the decoding functional units;

an FU-control signal and context information table (FU-CSCIT) representing CSCI information for performing a process by the decoding functional units.

【Claim 32】

The decoding device of Claim 31, wherein a default value table (DVT) representing a relationship between actual values and code values during the entropy coding is further stored as a partial decoder description in the description storing unit.

【Claim 33】

A decoding device, comprising:

a tool box including a plurality of functional units, the functional units realized to independently perform each process of at least one standard for decoding;

a description storing unit storing partial decoder descriptons for controlling operations of all or parts of the functional units; and

a decoder implementation unit controlling to decode encoded video data into video data by hierarchically loading the functional units by referring to at least one partial decoder description.

【Claim 34】

The decoding device of Claim 33, wherein the decoder implementation unit

comprises or is connected to a work memory for performing a process of the functional unit loaded by the control of the decoder implementation unit.

【Claim 35】

The decoding device of Claim 34, wherein the decoder implementation unit comprises:

a storing unit storing at least one of Control Signal/Context Information (CSCI) information, generated by performing a process by at least one functional unit, and data for decoding process; and

a connection control unit controlling selective load of the functional units.

【Claim 36】

The decoding device of Claim 33, wherein the partial decoder description stored in the description storing unit includes information on layers of the functional units, and the decoder implementation unit performs a calling process between the layers of at least one functional unit by using the information on layers.

【Claim 37】

The decoding device of claim 36, wherein the decoder implementation unit controls generation and excution of objects for a higest layer, and functional units

included in the higest layer and a calling unit for calling a layer assigned by the information of hierarchical layers are loaded by the excecution of the object.

【Claim 38】

The decoding device of claim 37, wherein the calling unit loads a corresponding layer when process data, necessary for performing a predetermined process of at least one functional unit included in an assigned layer, are stored in a storing unit.

【Claim 39】

The decoding device of claim 36, wherein a hierarchical structure recognized by the information on hierarchical layers comprises at least one layer among a sequence layer, a GOP layer, a picture layer, a slice layer, a macro block layer, and a block layer, and a lower layer is called by a higher layer.

【Claim 40】

The decoding device of Claim 33, wherein the partial decoder descriptions stored in the description storing unit include:

a decoding hierarchy table (DHT) representing a hierarchical structure for decoding the encoded video data and information on a lower layer per each hierarchy;

a syntax element table (SET) representing a process for generating information for bitstream syntax and element information corresponding to the bitstream syntax;

a syntax rule table (S-RT) designating names of connection information between the bitstream syntax, information for the lower layer per each hierarchy to call, and CSCI information which is result data generated by performing the SET process;

a control signal and context information table (CSCIT) representing detailed information of the CSCI information per each hierarchical structure;

an FU rule table (F-RT) representing an order of call or activation between at least one decoding functional unit based on the hierarchical structure;

an FU list (FL) representing a list of the decoding functional units; and

an FU-control signal and context information table (FU-CSCIT) representing CSCI information for performing a process by the decoding functional units.

【Claim 41】

The decoding device of Claim 40, wherein a default value table (DVT) representing a relationship between actual values and code values during the entropy coding is further stroed as a partial decoder description in the description storing unit.

【Claim 42】

The decoding device of Claim 40, wherein the connection control unit controls to load at least one functional unit corresponding to a higest layer using the F-RT.

【Claim 43】

The decoding device of Claim 40, wherein the parsing functional unit generates at least one of the CSCI information and the data for decoding by using at least one of the SET, S-RT, and CSCIT.

【Claim 44】

A decoding method comprising:

(a) generating and storing a decoder description corresponding to an inputted decoder description, wherein the decoder description comprises a plurality of partial decoder descriptions; and

(b) loading a parsing functional unit and one or more of decoding functional units corresponding to a highest layer by using one or more of partial decoder descriptions,

whereas the parsing functional unit performs syntax parsing of the bitstream, and the decoding functional units of the highest layer perform processing when control signal and context information (CSCI) needed for performing a predetermined process and data for decoding are stored in the storing unit by the parsing functional unit, and

when the control signal and context information (CSCI) for performing a first layer corresponding to the hierarchical structure and data for decoding are stored in the storing unit, the first layer is loaded by an activated second layer.

【Claim 45】

The decoding method of Claim 44, wherein the decoding functional units determine activation or call depending on the hierarchical structure of the encoded video data, and video data corresponding to the bitstream is outputted by performing one or more times of the process of each decoding functional unit specified in the decoder description.

【Claim 46】

The decoding method of Claim 44, comprising partial decoder descriptions, wherein the decoder description includes:

a decoding hierarchy table (DHT) representing a hierarchical structure for decoding encoded video data included in the bitstream and information on a lower layer per each hierarchy;

a syntax element table (SET) representing a process for generating information for bitstream syntax and element information corresponding to the bitstream syntax;

a syntax rule table (S-RT) designating names of connection information between the bitstream syntax, information for a lower layer per each hierarchy to call, and CSCI information in which result data generated by performing the SET process is to be stored;

a control signal and context information table (CSCIT) representing detailed information of the CSCI information per each hierarchical structure;

an FU rule table (F-RT) representing an order of call or activation between at least one decoding functional unit based on the hierarchical structure;

an FU list (FL) representing a list of the decoding functional units; and

an FU-control signal and context information table (FU-CSCIT) representing CSCI information for performing a process by the decoding functional units.

【Claim 47】

The decoding method of Claim 46, wherein an activation or calling operation of at least one functional unit corresponding to a higest layer is controlled by using the F-RT.

FIG. 1

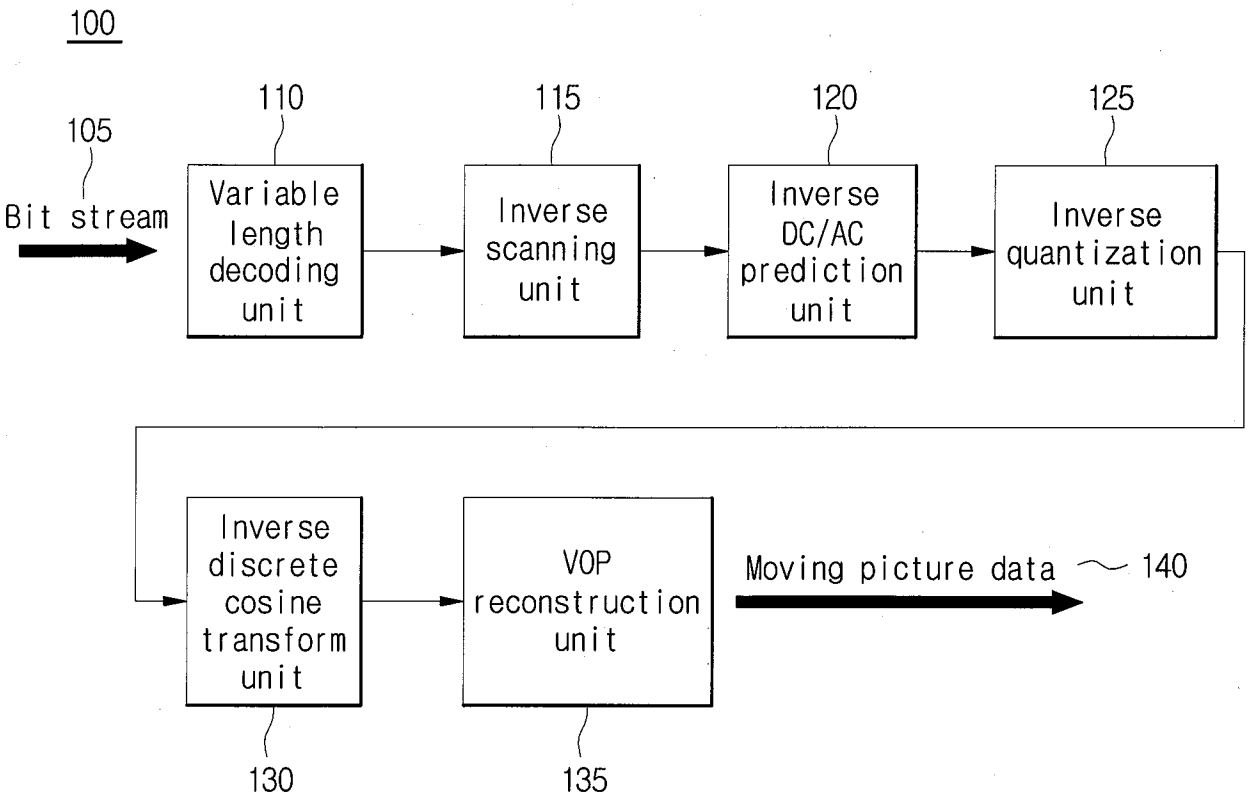

100
105
Bit stream
110
Variable length decoding unit
115
Inverse scanning unit
120
Inverse DC/AC prediction unit
125
Inverse quantization unit
Inverse discrete cosine transform unit
130
VOP reconstruction unit
135
Moving picture data
140

FIG. 2

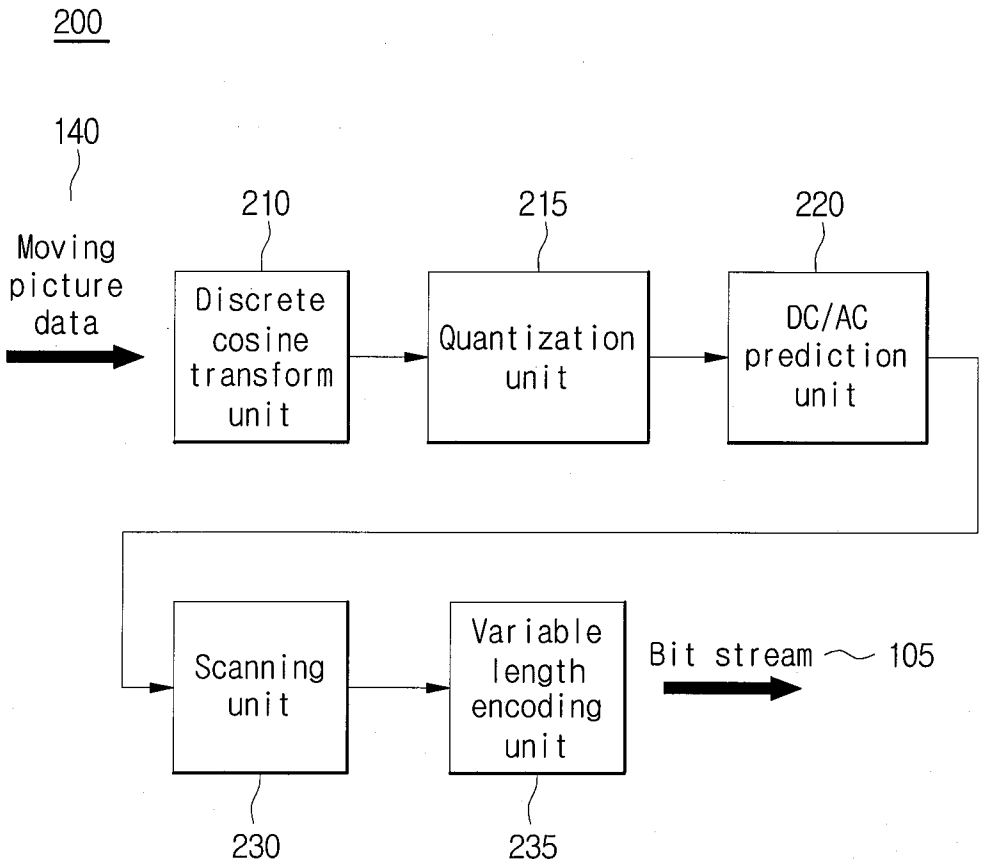
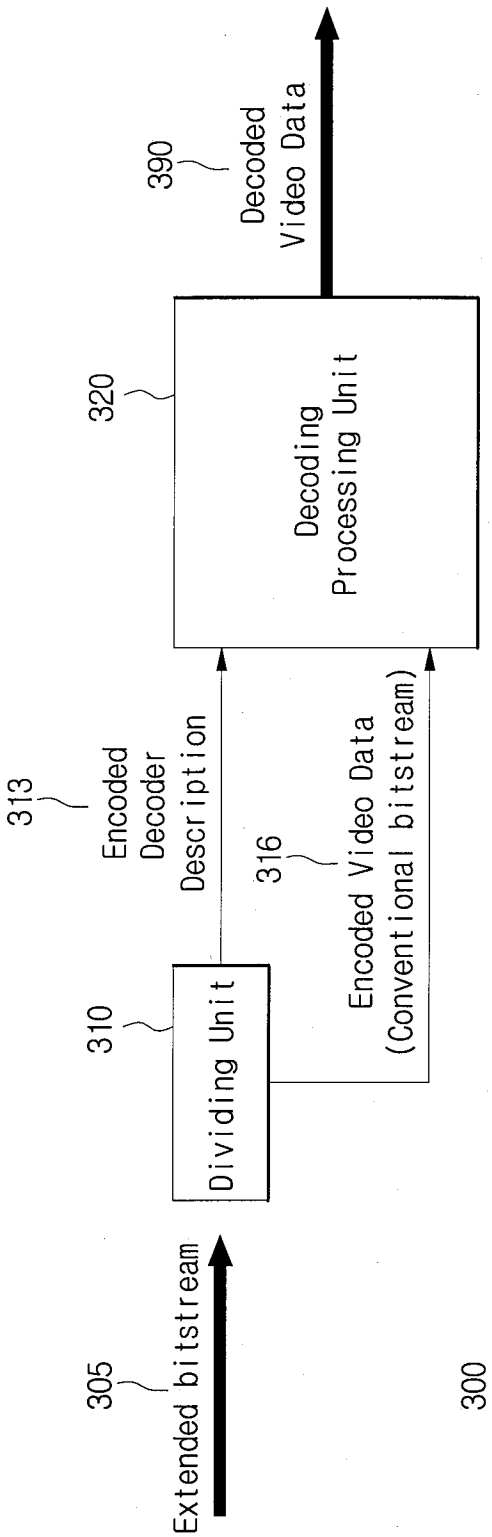
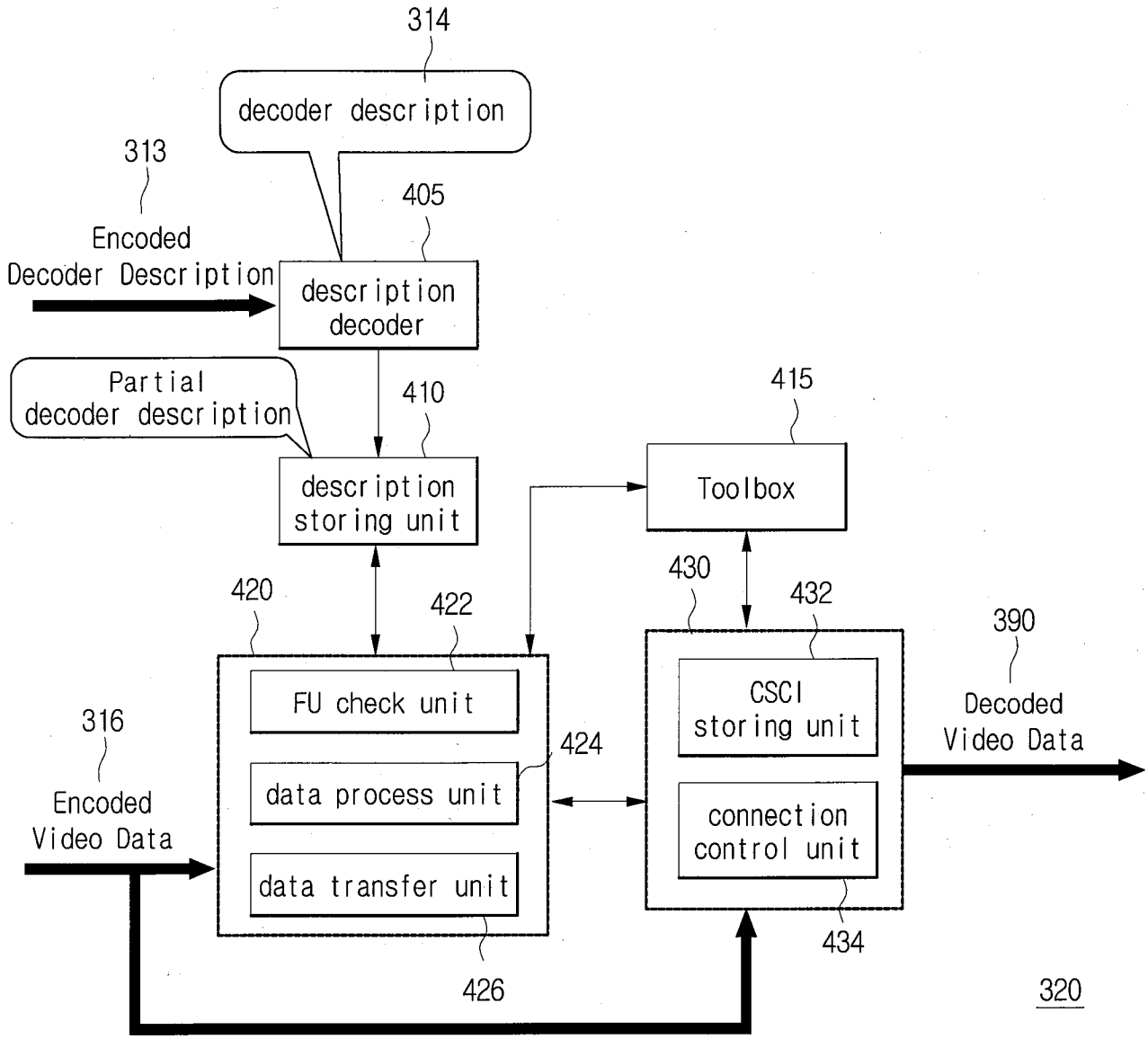

200
140
Moving picture data
210
Discrete cosine transform unit
215
Quantization unit
220
DC/AC prediction unit
Scanning unit
230
Variable length encoding unit
235
Bit stream
105

FIG. 3


300
305
Extended bitstream
310
Dividing Unit
313
Encoded Decoder Description
316
Encoded Video Data (Conventional bitstream)
320
Decoding Processing Unit
390
Decoded Video Data

4/26
FIG. 4


314
decoder description
313
405
Encoded
Decoder Description
description decoder
Partial decoder description
410
415
description storing unit
Toolbox
420
422
430
432
390
FU check unit
CSCI storing unit
316
424
Decoded Video Data
data process unit
Encoded Video Data
connection control unit
data transfer unit
434
426
320

5/26
FIG. 5

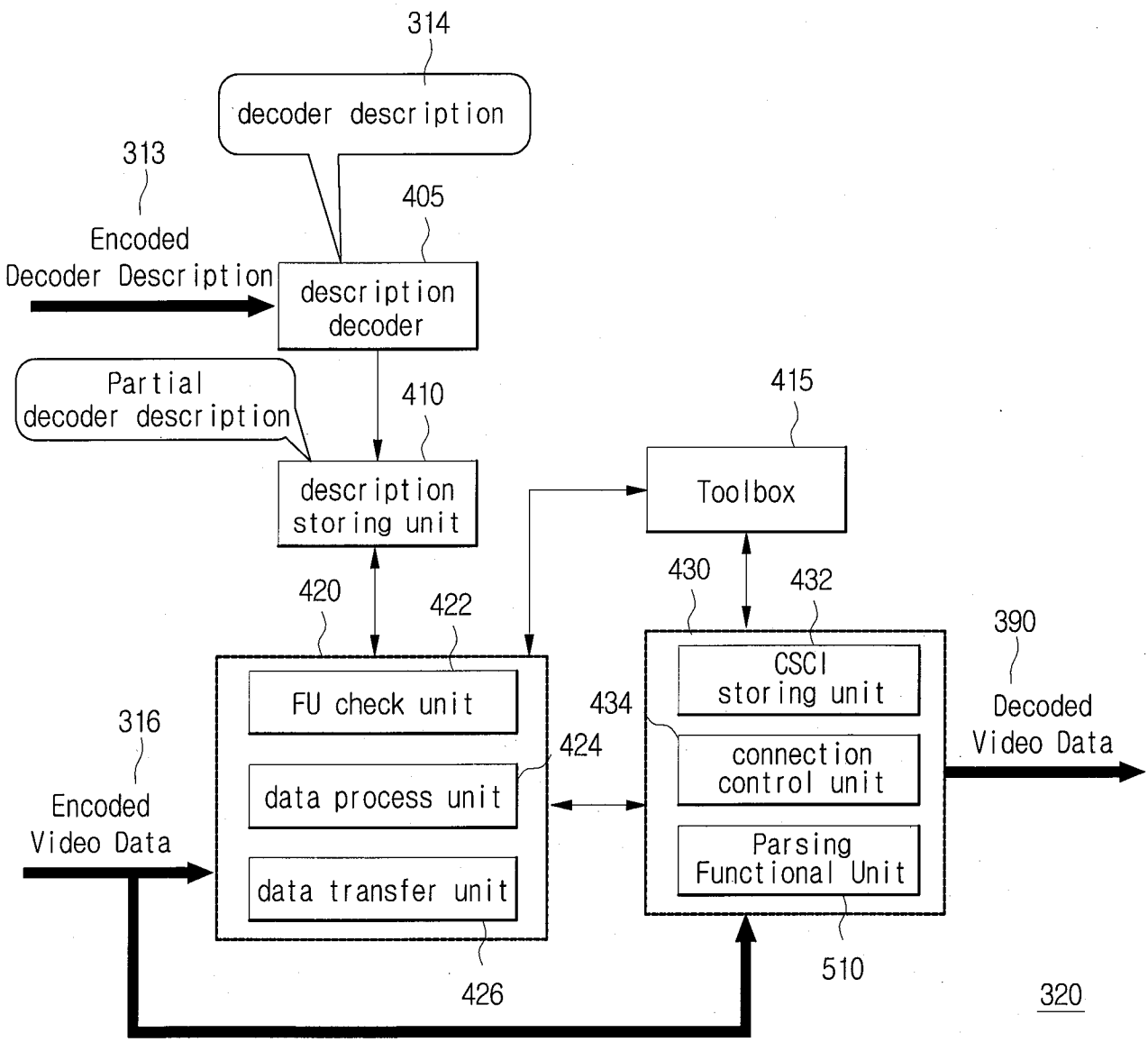

314
decoder description
313
405
Encoded
Decoder Description
description decoder
Partial decoder description
410
415
description storing unit
Toolbox
420
422
430
432
390
CSCI storing unit
434
316
424
FU check unit
data process unit
connection control unit
Decoded Video Data
Encoded Video Data
data transfer unit
Parsing Functional Unit
426
510
320

FIG. 6

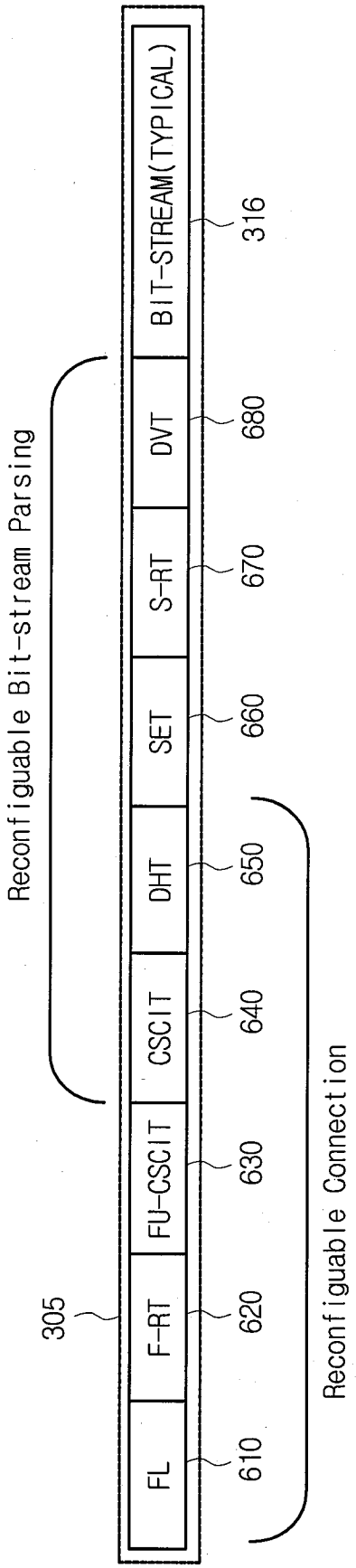
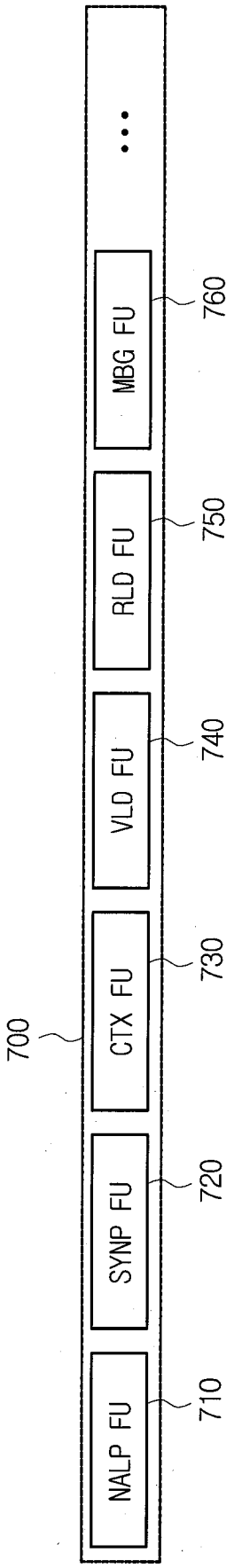
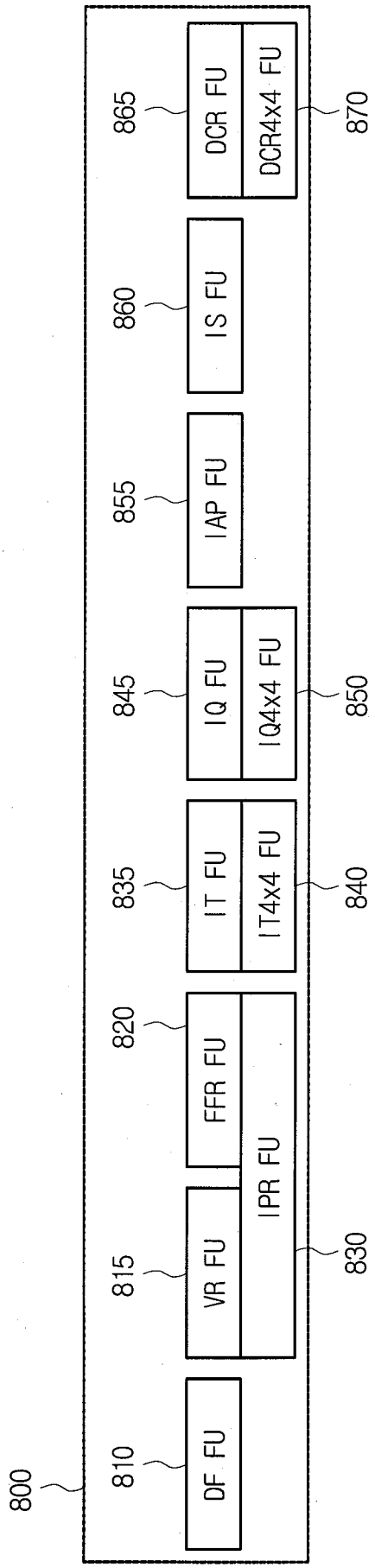

Reconfiguable Bit-stream Parsing
Reconfiguable Connection
305
FL
F-RT
FU-CSCIT
CSCIT
DHT
SET
S-RT
DVT
BIT-STREAM(TYPICAL)
610
620
630
640
650
660
670
680
316

FIG. 7



Sample of Functional Uits for Syntax Parsing

FIG. 8



Sample of Functional units for Decoding Process

FIG. 9

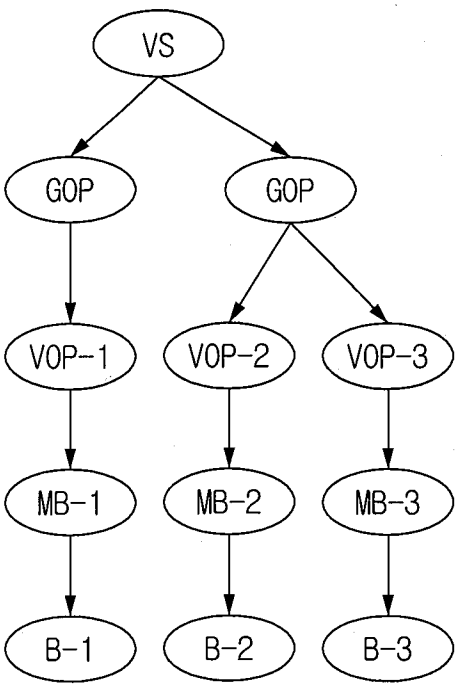


FIG. 10

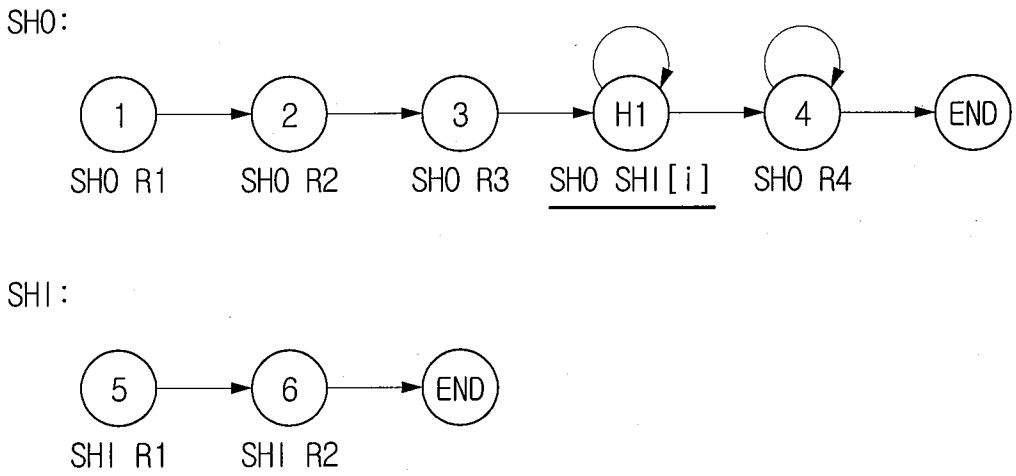

SH0:
1
2
3
H1
4
END
SH0 R1
SH0 R2
SH0 R3
SH0 SH1[i]
SH0 R4
SH1:
5
6
END
SH1 R1
SH1 R2

FIG. 11

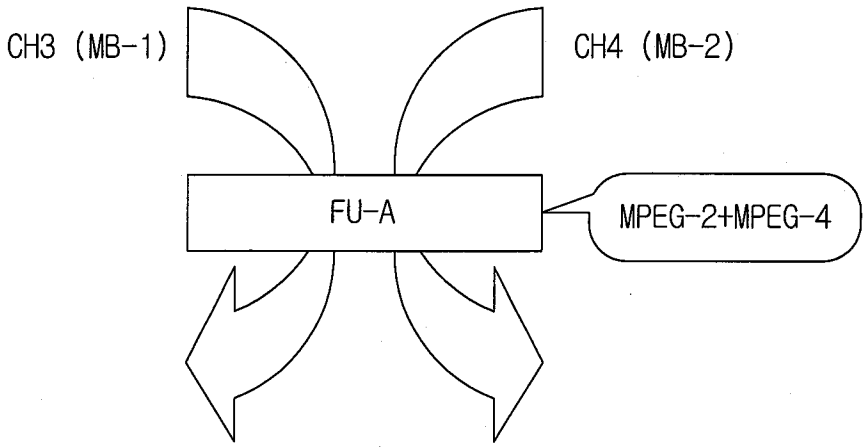


FIG. 12

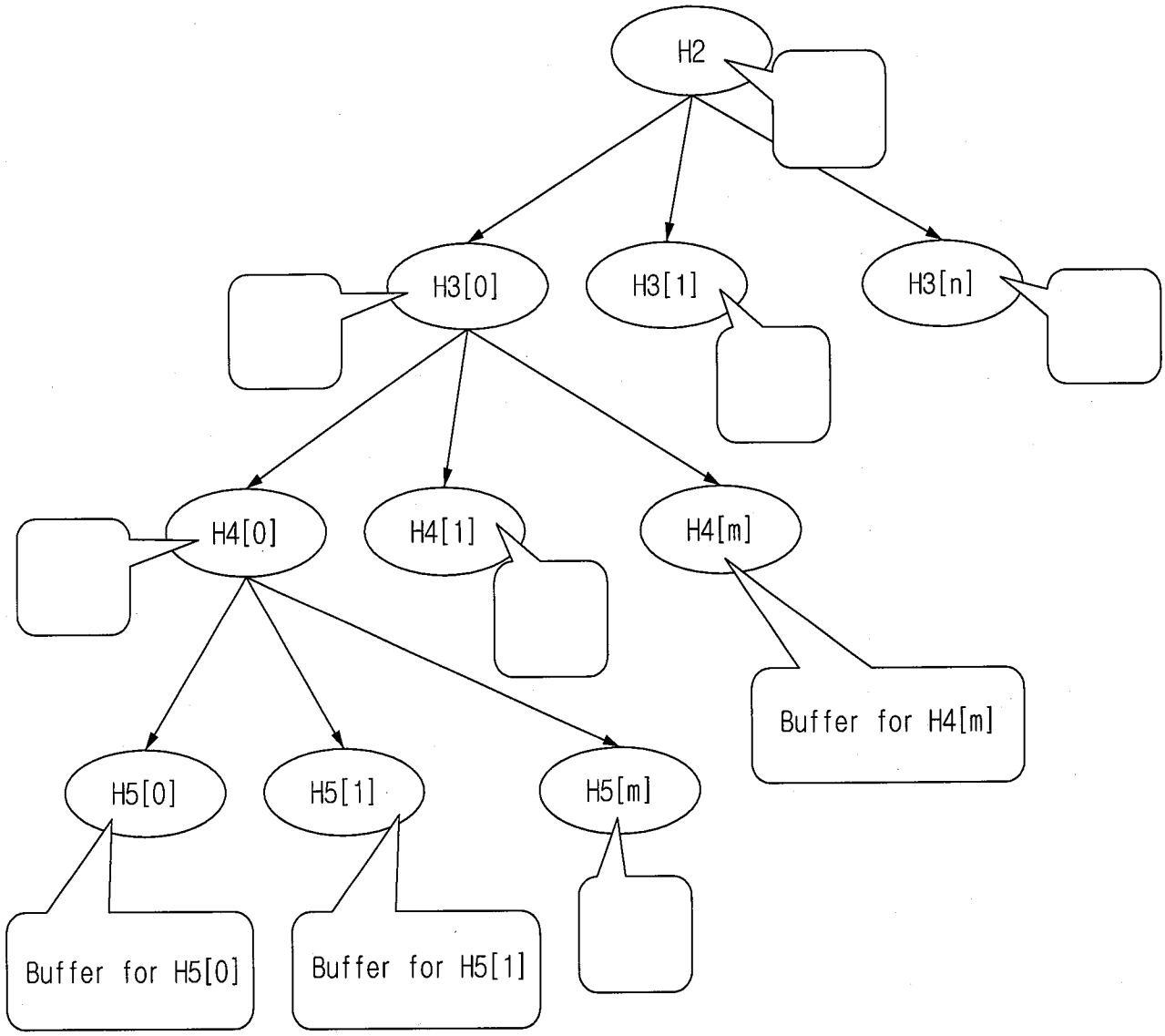

H2
H3[0]
H3[1]
H3[n]
H4[0]
H4[1]
H4[m]
Buffer for H4[m]
H5[0]
H5[1]
H5[m]
Buffer for H5[0]
Buffer for H5[1]

FIG. 13

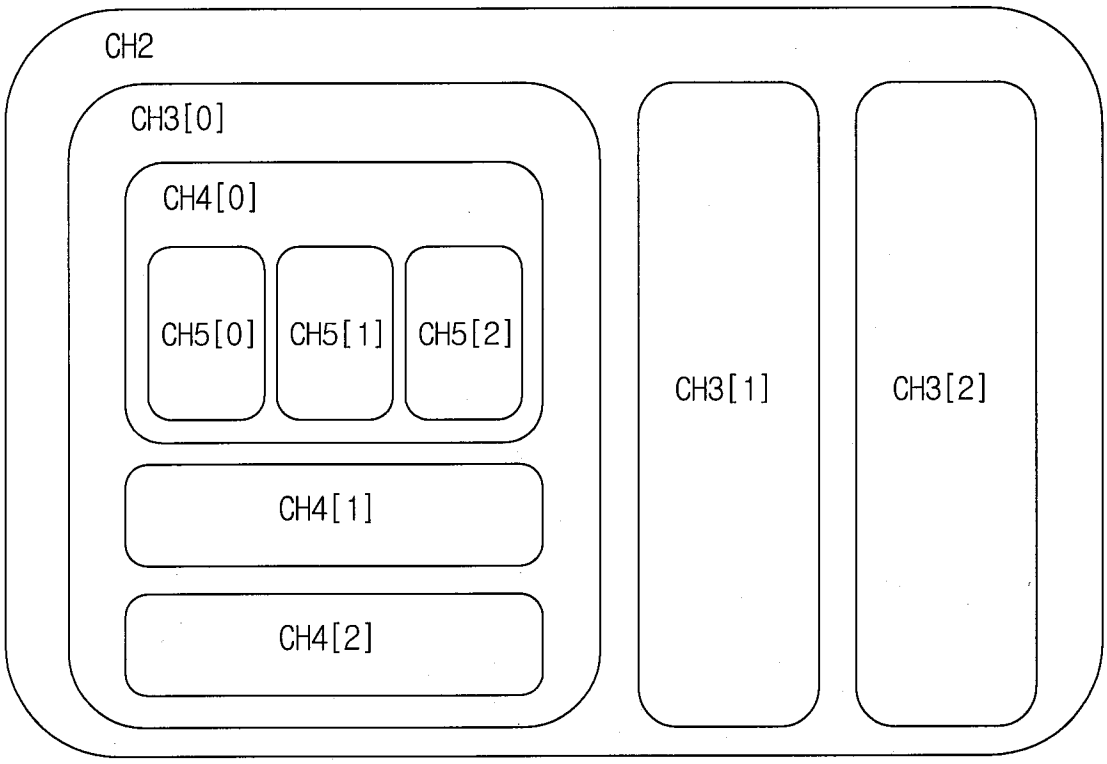


FIG. 14

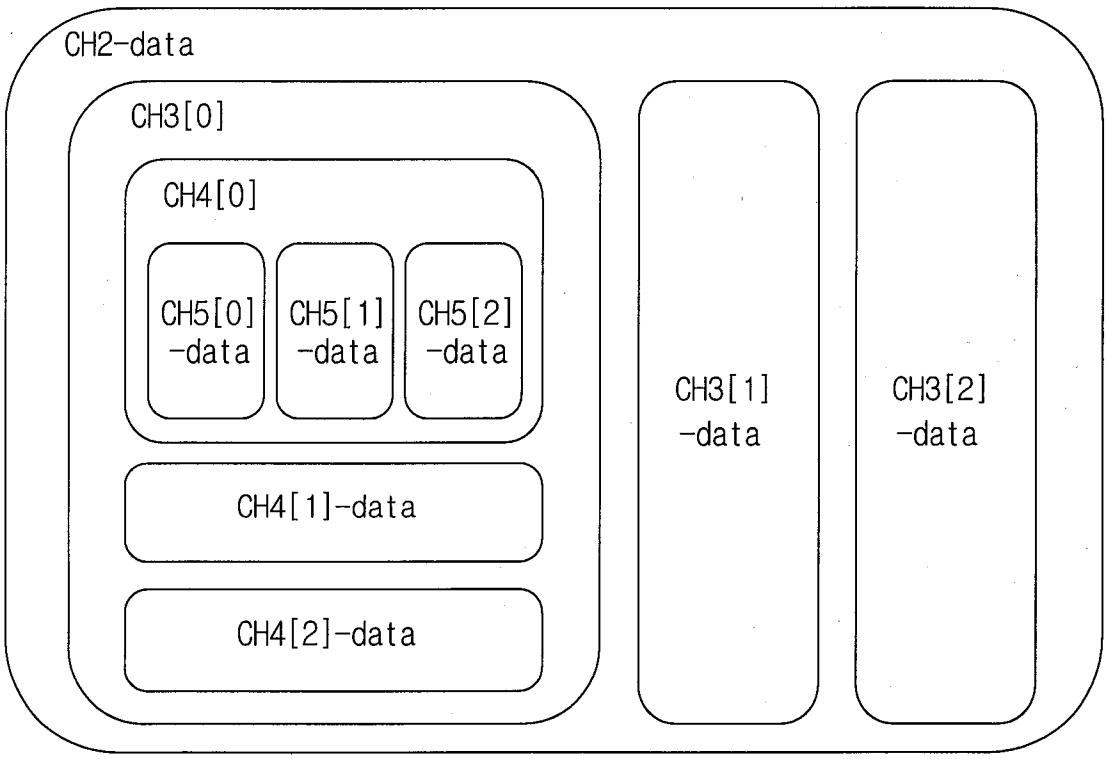

CH2-data
CH3[0]
CH4[0]
CH5[0] -data
CH5[1] -data
CH5[2] -data
CH4[1]-data
CH4[2]-data
CH3[1] -data
CH3[2] -data

FIG. 15

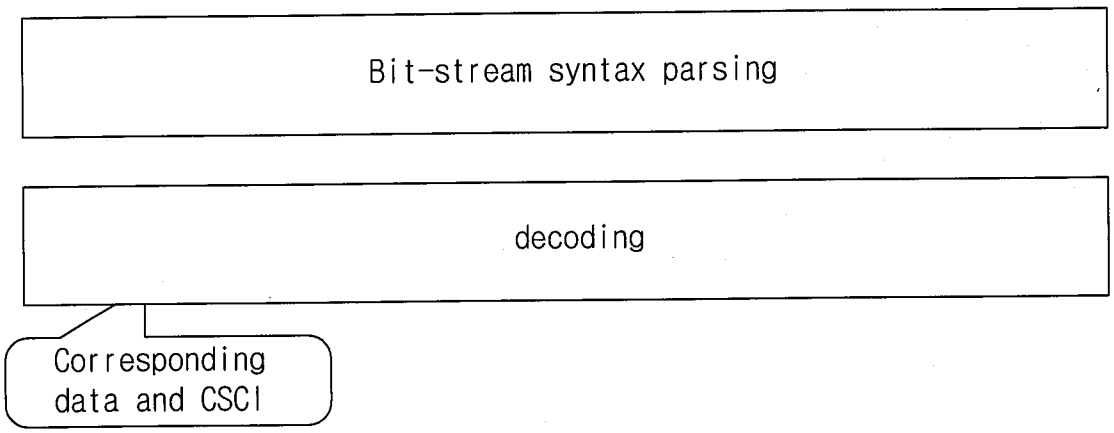

Bit-stream syntax parsing
decoding
Corresponding data and CSCI

FIG. 16

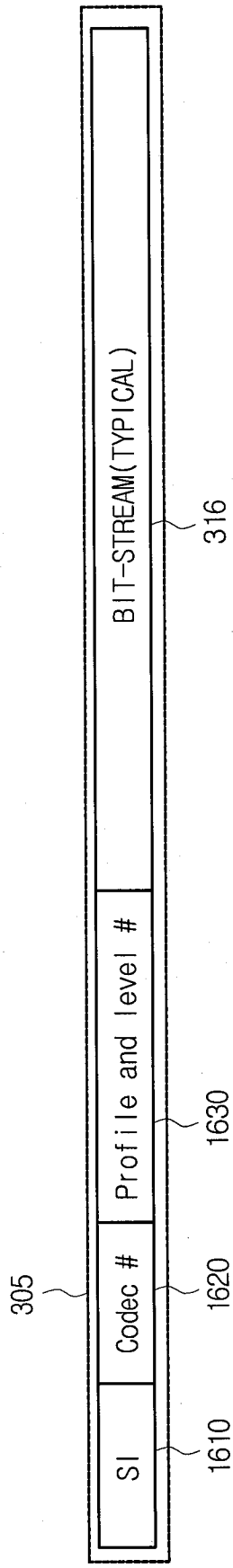

305
SI
Codec #
Profile and level #
BIT-STREAM(TYPICAL)
1610
1620
1630
316

FIG. 17

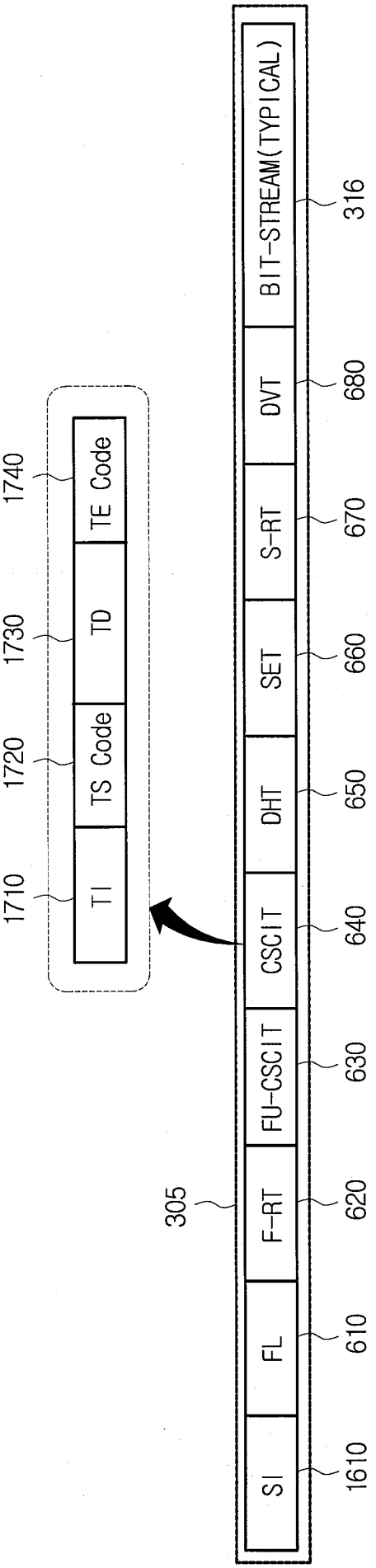
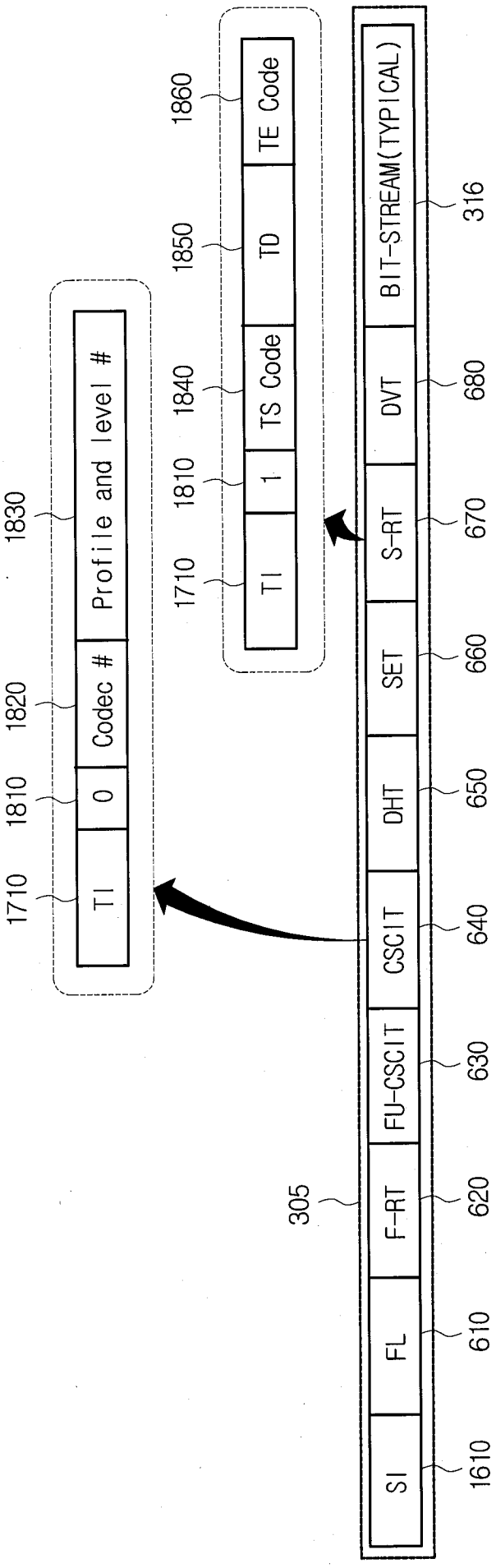

SI
FL
F-RT
FU-CSCIT
CSCIT
DHT
SET
S-RT
DVT
BIT-STREAM(TYPICAL)
1610
610
620
630
640
650
660
670
680
316
305
TI
TS Code
TD
TE Code
1710
1720
1730
1740

FIG. 18


SI
FL
F-RT
FU-CSCIT
CSCIT
DHT
SET
S-RT
DVT
BIT-STREAM(TYPICAL)
305
1610
610
620
630
640
650
660
670
680
316
TI
0
Codec #
Profile and level #
1710
1810
1820
1830
TI
1
TS Code
TD
TE Code
1710
1810
1840
1850
1860

19/26
FIG. 19

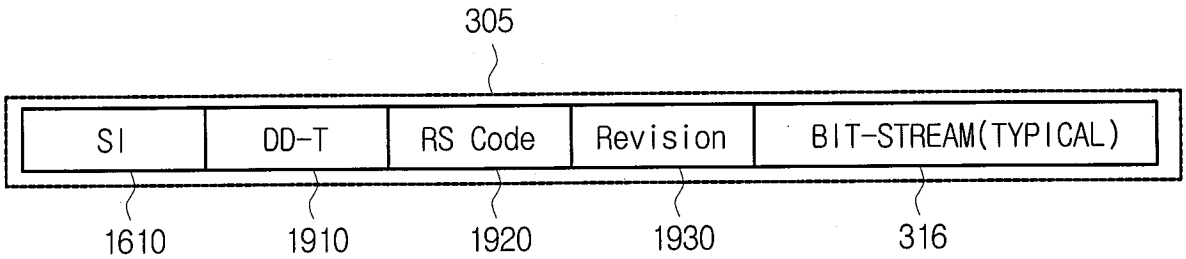

305
SI
DD-T
RS Code
Revision
BIT-STREAM(TYPICAL)
1610
1910
1920
1930
316

FIG. 20

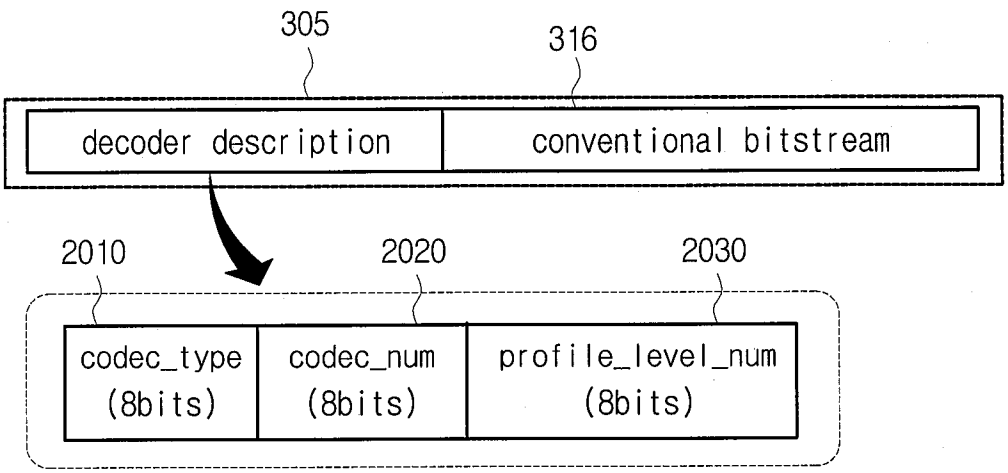

305
316
decoder description
conventional bitstream
2010
2020
2030
codec_type
(8bits)
codec_num
(8bits)
profile_level_num
(8bits)

FIG. 21

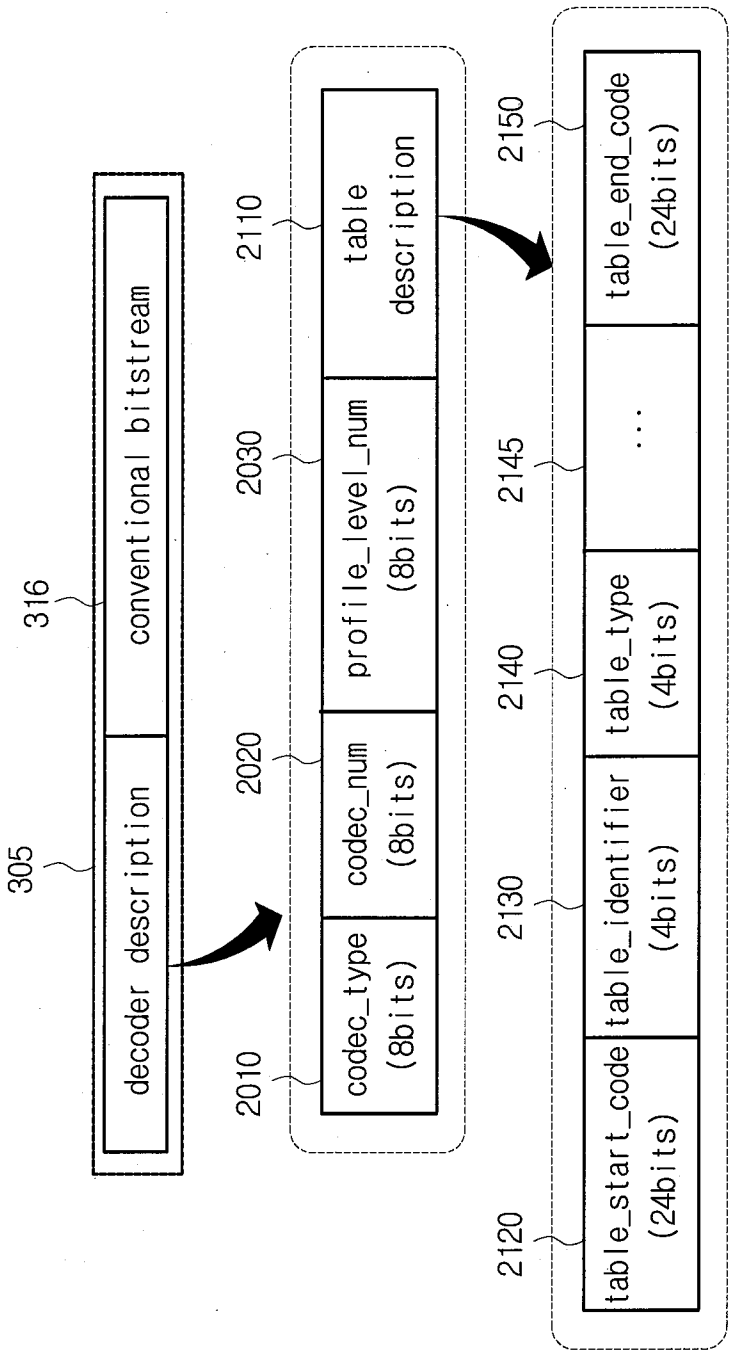

305
316
decoder description
conventional bitstream
2010
codec_type (8bits)
2020
codec_num (8bits)
2030
profile_level_num (8bits)
2110
table description
2120
table_start_code (24bits)
2130
table_identifier (4bits)
2140
table_type (4bits)
2145
...
2150
table_end_code (24bits)

FIG. 22

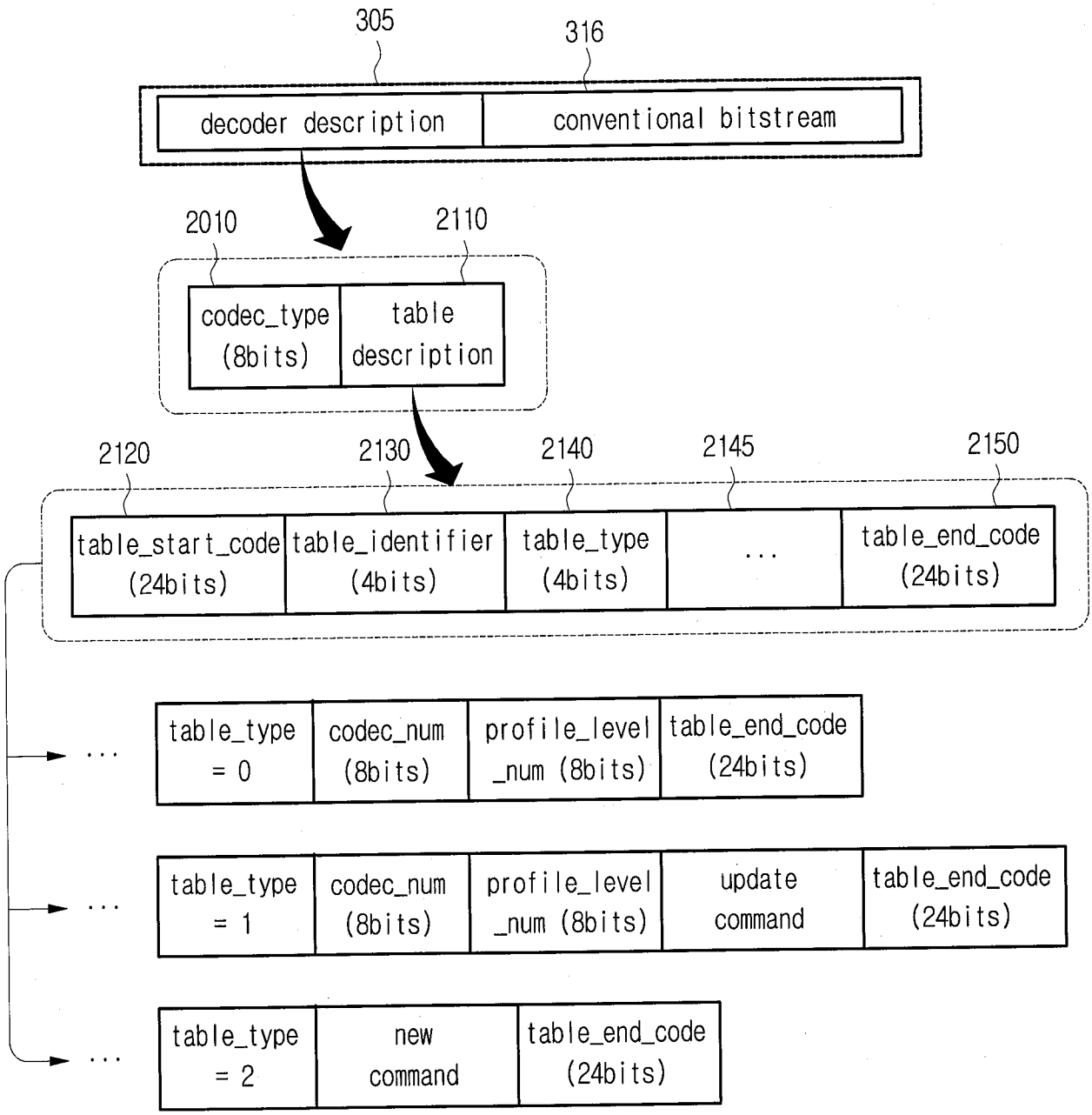

305
316
decoder description
conventional bitstream
2010
2110
codec_type (8bits)
table description
2120
2130
2140
2145
2150
table_start_code (24bits)
table_identifier (4bits)
table_type (4bits)
...
table_end_code (24bits)
table_type = 0
codec_num (8bits)
profile_level _num (8bits)
table_end_code (24bits)
table_type = 1
codec_num (8bits)
profile_level _num (8bits)
update command
table_end_code (24bits)
table_type = 2
new command
table_end_code (24bits)

FIG. 23

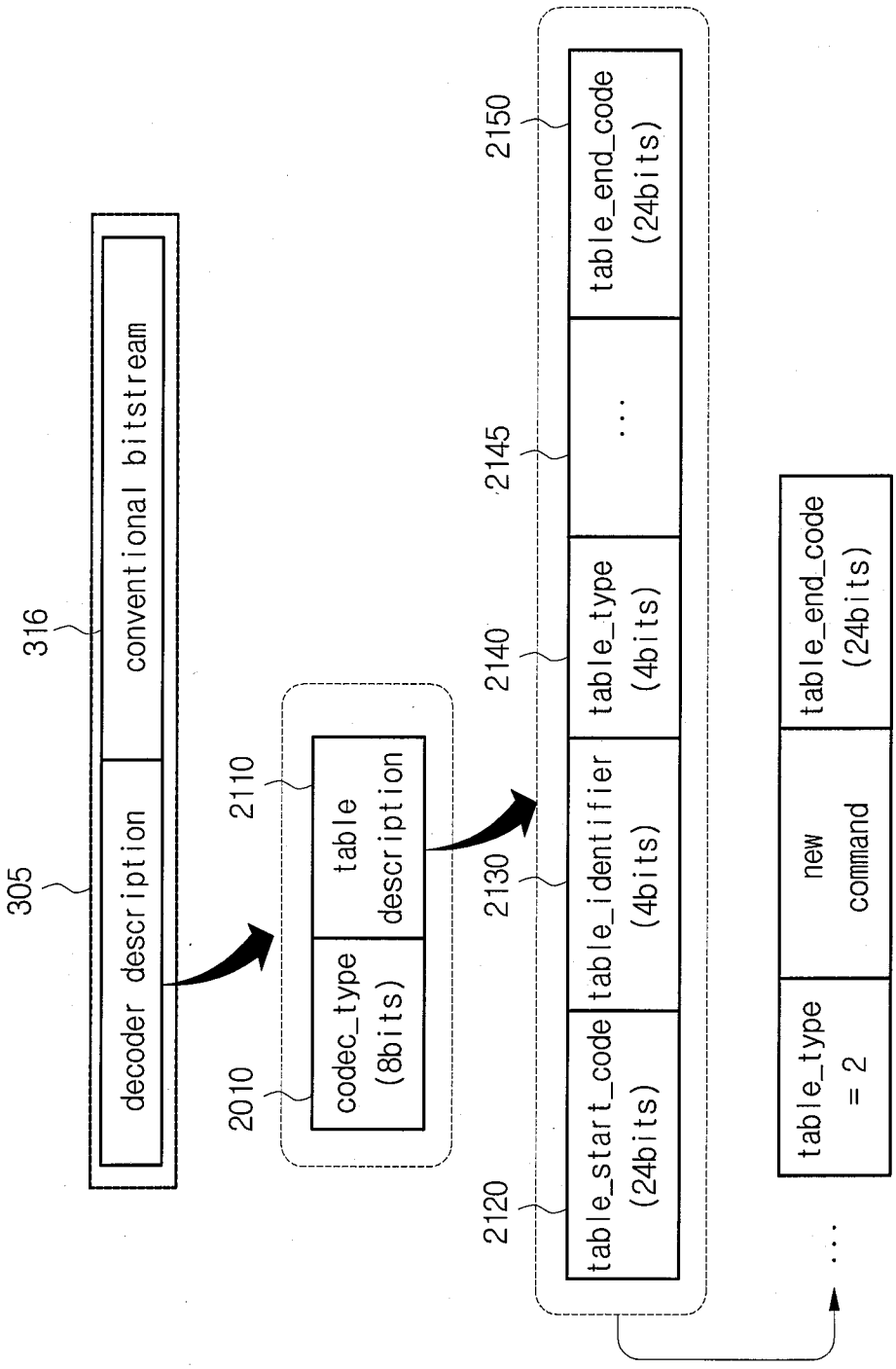

316
305
decoder description
conventional bitstream
2110
table description
2010
codec_type (8bits)
2150
table_end_code (24bits)
2145
...
2140
table_type (4bits)
2130
table_identifier (4bits)
2120
table_start_code (24bits)
table_type = 2
new command
table_end_code (24bits)
...

FIG. 24

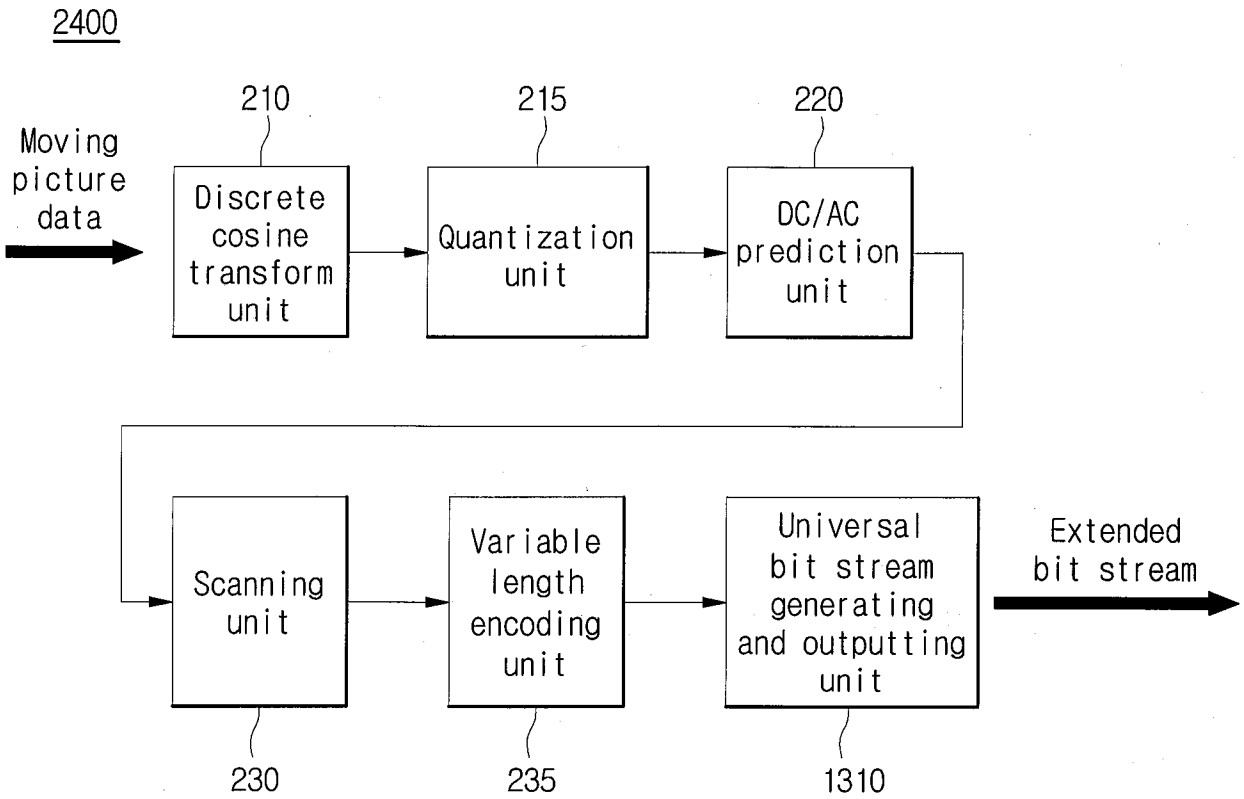

2400
Moving picture data
210
Discrete cosine transform unit
215
Quantization unit
220
DC/AC prediction unit
Scanning unit
230
Variable length encoding unit
235
Universal bit stream generating and outputting unit
1310
Extended bit stream

FIG. 25

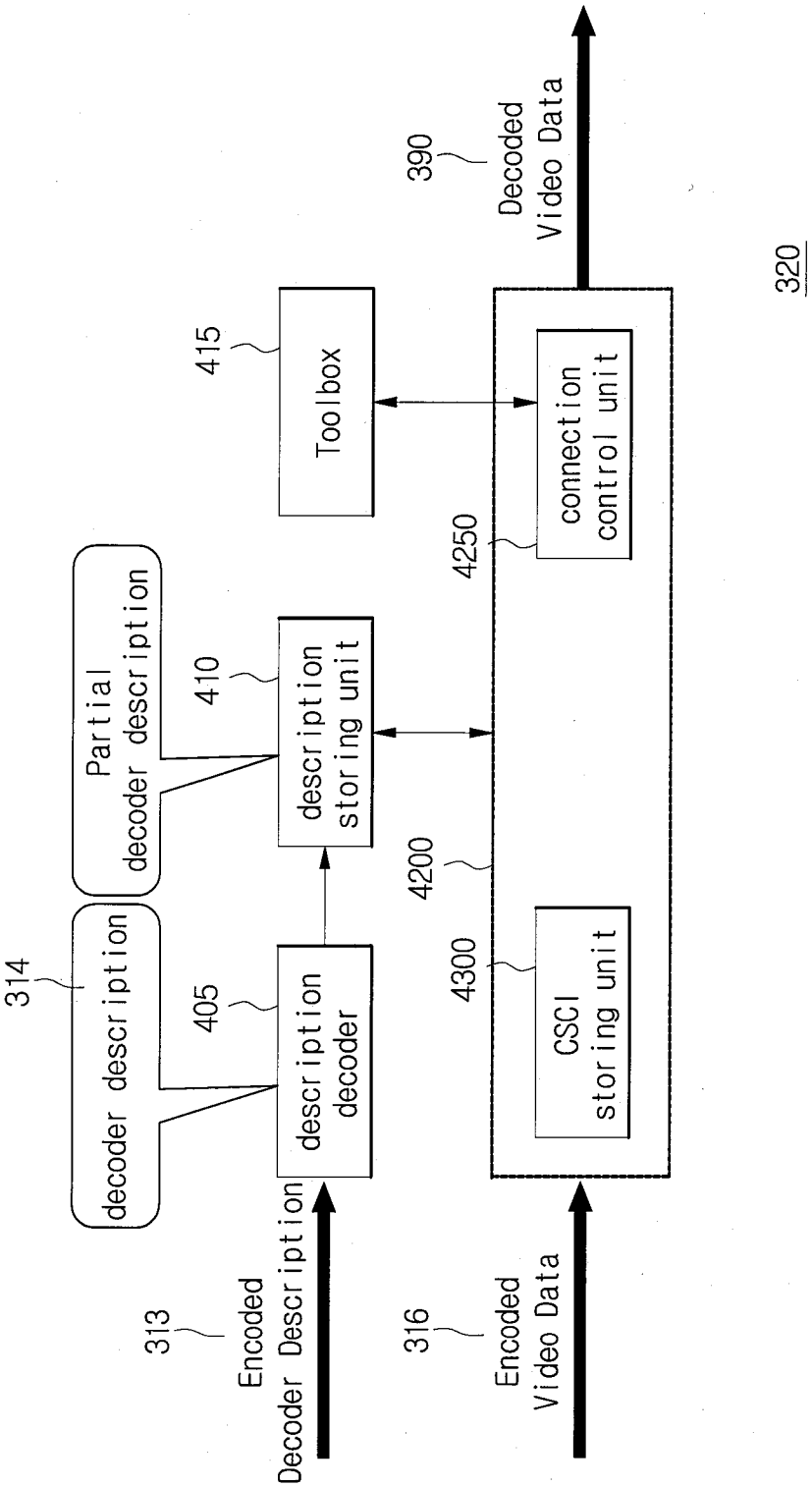
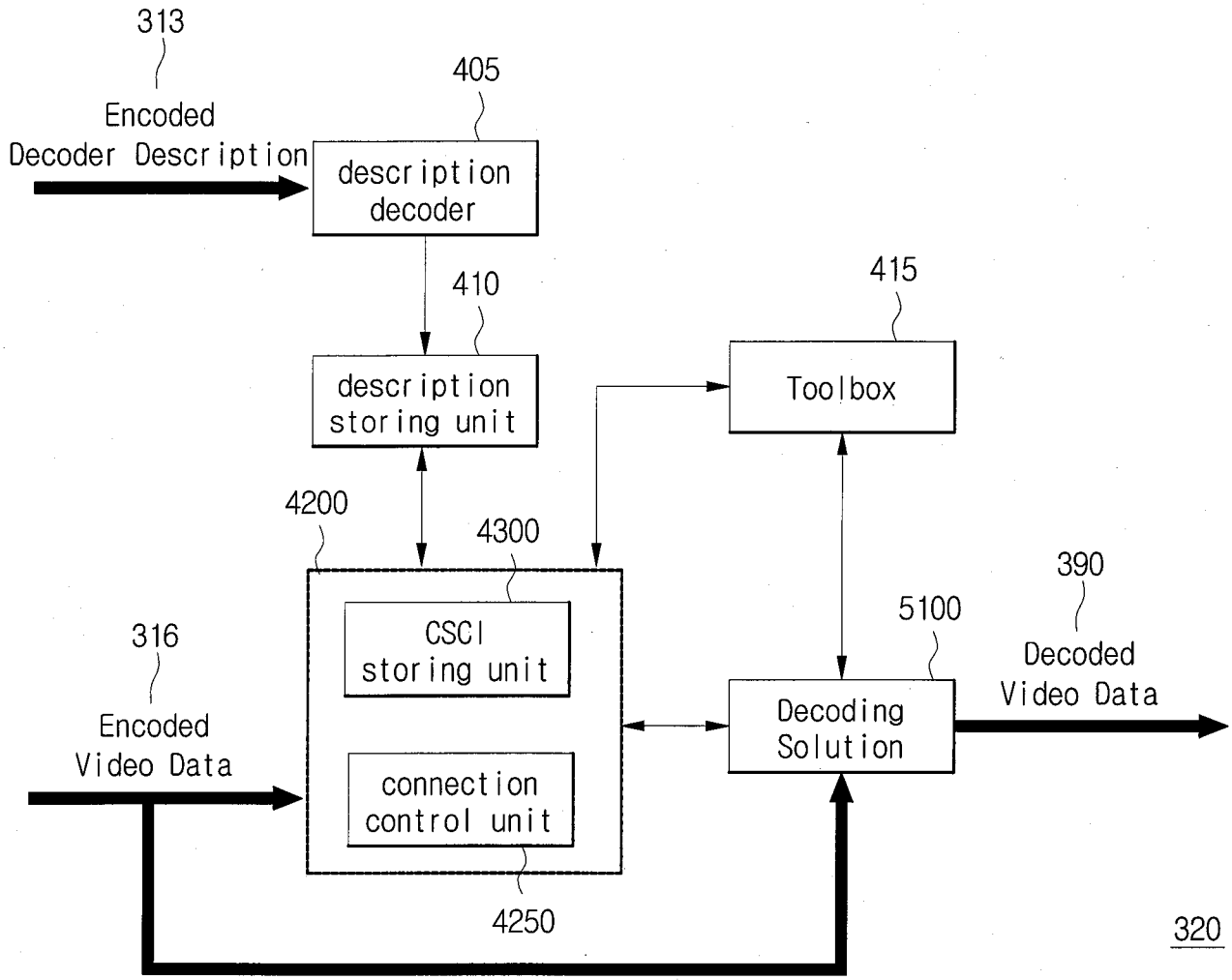

313
Encoded Decoder Description
316
Encoded Video Data
314
decoder description
405
description decoder
Partial decoder description
410
description storing unit
415
Toolbox
4200
4300
CSCI storing unit
4250
connection control unit
390
Decoded Video Data
320

FIG. 26


313
Encoded
Decoder Description
405
description decoder
410
description storing unit
415
Toolbox
4200
4300
CSCI storing unit
connection control unit
4250
316
Encoded
Video Data
5100
Decoding Solution
390
Decoded
Video Data
320

INTERNATIONAL SEARCH REPORT

International application No.
PCT/KR2008/001928

A. CLASSIFICATION OF SUBJECT MATTER

H04N 7/24(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC 8 H04N H04B

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean Utility models and applications for Utility models since 1975

Japanese Utility models and applications for Utility models since 1975

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKIPASS, SEARCH TERMS:Syntax, Parsing, Bitstream, Decoder and similar terms

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 20070071103 A1 (BI et al.) 29 MARCH 2007 See abstract; Paragraph[0009] - Paragraph[0011]; Paragraph[0050] - Paragraph[0057]; Figures 4-6.	1-47
A	US 20060109912 A1 (WINGER et al.) 25 MAY 2006 See abstract; Paragraph[0020] - Paragraph[0021]; Paragraph[0029] - Paragraph[0041]; Figures 2-5.	1-47
A	US 20020110193 A1 (YOO et al.) 15 AUGUST 2002 See abstract; Paragraph[0005] - Paragraph[0017]; Paragraph[0120] - Paragraph[0123]; Figure 13.	1-47

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
22 JULY 2008 (22.07.2008)	**22 JULY 2008 (22.07.2008)**

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
Government Complex-Daejeon, 139 Seonsa-ro, Seo-gu, Daejeon 302-701, Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

PARK, Sang Chol

Telephone No. 82-42-481-8372

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/KR2008/001928

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2007-071103 A1	29.03.2007	NONE	
US 2006-109912 A1	25.05.2006	NONE	
US 2002-0110193 A1	15.08.2002	JP 2000-299687	24.10.2000
		JP 2000-299687 A2	24.10.2000
		US 6999512	14.02.2006
		US 2002-110193 A1	15.08.2002
		US 2002-110193 AA	15.08.2002
		US 6999512 BB	14.02.2006