(12) STANDARD PATENT (11) Application No. **AU 2021206151 B2**
(19) AUSTRALIAN PATENT OFFICE

(54) Title
Method and apparatus for signaling skip mode flag

(51) International Patent Classification(s)
H04N 19/50 (2014.01) ***H04N 19/44*** (2014.01)

(21) Application No: **2021206151** (22) Date of Filing: **2021.01.04**

(87) WIPO No: **WO21/141851**

(30) Priority Data

(31) Number	(32) Date	(33) Country
62/959,621	**2020.01.10**	**US**
17/095,583	**2020.11.11**	**US**

(43) Publication Date: **2021.07.15**
(44) Accepted Journal Date: **2023.02.09**

(71) Applicant(s)
Tencent America LLC

(72) Inventor(s)
XU, Xiaozhong;LI, Guichun;LIU, Shan

(74) Agent / Attorney
Griffith Hack, Level 15, 376-390 Collins Street, Melbourne, VIC, 3000, AU

(56) Related Art
US 2016/0241858 A1
US 2015/0264372 A1

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 July 2021 (15.07.2021)

WIPO | PCT

(10) International Publication Number
WO 2021/141851 A1

(51) International Patent Classification:
H04N 19/50 (2014.01) *H04N 19/44* (2014.01)

(21) International Application Number:
PCT/US2021/012078

(22) International Filing Date:
04 January 2021 (04.01.2021)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/959,621 10 January 2020 (10.01.2020) US
17/095,583 11 November 2020 (11.11.2020) US

(71) Applicant: **TENCENT AMERICA LLC** [US/US]; 2747 Park Boulevard, Palo Alto, California 94306 (US).

(72) Inventors: **XU, Xiaozhong**; C/o Tencent America LLC, 2747 Park Boulevard, Palo Alto, California 94306 (US). **LI, Guichun**; C/o Tencent America LLC, 2747 Park Boulevard, Palo Alto, California 94306 (US). **LIU, Shan**; C/o Tencent America LLC, 2747 Park Boulevard, Palo Alto, California 94306 (US).

(74) Agent: **MA, Johnny** et al.; Oblon, McClelland, Maier & Neustadt, L.L.P., 1940 Duke Street, Alexandria, Virginia 22314 (US).

(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, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW,

(54) Title: METHOD AND APPARATUS FOR SIGNALING SKIP MODE FLAG

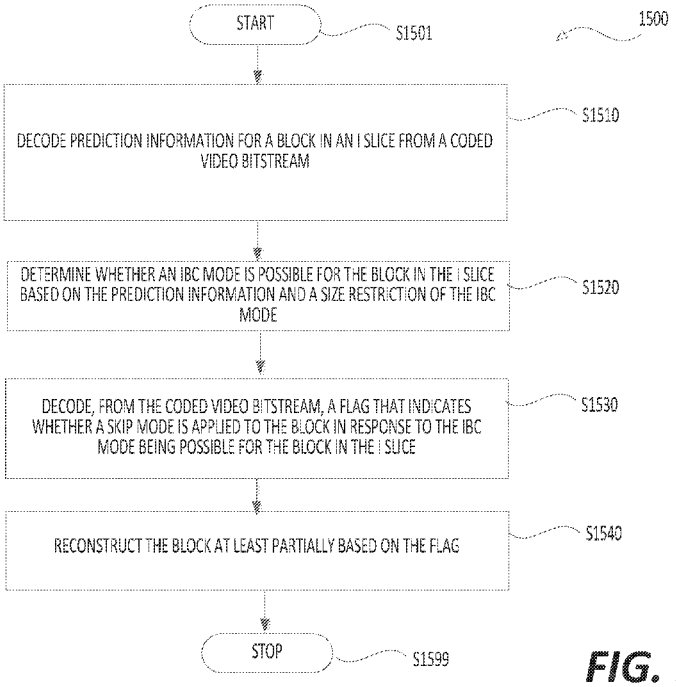

FIG. 15

(57) Abstract: Aspects of the disclosure provide methods and apparatuses for video encoding/decoding. In some examples, an apparatus for video decoding includes receiving circuitry and processing circuitry. In some embodiments, processing circuitry decodes, prediction information for a block in an I slice from a coded video bitstream, and determines, whether an intra block copy (IBC) mode is possible for the block in the I slice. In response to a slice type parameter indicating I slice and at least a width or height of the block being greater than 64, the processing circuitry sets a current mode type parameter to MODE_TYPE_INTRA. Further, in an embodiment, the processing circuitry decodes a flag that indicates whether a skip mode is applied on the block from the coded video bitstream. Then, the processing circuitry reconstructs the block at least partially based on the flag.

WO 2021/141851 A1

[Continued on next page]

SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

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

Published:

— *with international search report (Art. 21(3))*

METHOD AND APPARATUS FOR SIGNALING SKIP MODE FLAG

INCORPORATION BY REFERENCE

[0001] This present application claims the benefit of priority to U.S. Patent Application No. 17/095,583, "METHOD AND APPARATUS FOR SIGNALING SKIP MODE FLAG" filed on November 11, 2020, which claims the benefit of priority to U.S. Provisional Application No. 62/959,621, "METHODS FOR SIGNALING OF SKIP MODE FLAG" filed on January 10, 2020. The entire disclosures of the prior applications are hereby incorporated by reference in their entirety.

TECHNICAL FIELD

[0002] The present disclosure describes embodiments generally related to video coding.

BACKGROUND

[0003] The background description provided herein is for the purpose of generally presenting the context of the disclosure. Work of the presently named inventors, to the extent the work is described in this background section, as well as aspects of the description that may not otherwise qualify as prior art at the time of filing, are neither expressly nor impliedly admitted as prior art against the present disclosure.

[0004] Video coding and decoding can be performed using inter-picture prediction and intra-picture prediction. Uncompressed digital video can include a series of pictures, each picture having a spatial dimension of, for example, 1920 x 1080 luminance samples and associated chrominance samples. The series of pictures can have a fixed or variable picture rate (informally also known as frame rate), of, for example 60 pictures per second or 60 Hz. Uncompressed video has significant bitrate requirements. For example, 1080p60 4:2:0 video at 8 bit per sample (1920x1080 luminance sample resolution at 60 Hz frame rate) requires close to 1.5 Gbit/s bandwidth. An hour of such video requires more than 600 GBytes of storage space.

[0005] One purpose of video coding and decoding can be the reduction of redundancy in the input video signal, through compression. Compression can help reduce the aforementioned bandwidth or storage space requirements, in some cases by two orders of magnitude or more. Both lossless and lossy compression, as well as a combination thereof can be employed. Lossless compression refers to techniques where an exact copy of the original signal can be reconstructed from the compressed original signal. When using lossy compression, the reconstructed signal may not be identical to the original signal, but the distortion between

original and reconstructed signals is small enough to make the reconstructed signal useful for the intended application. In the case of video, lossy compression is widely employed. The amount of distortion tolerated depends on the application; for example, users of certain consumer streaming applications may tolerate higher distortion than users of television distribution applications. The compression ratio achievable can reflect that: higher allowable/tolerable distortion can yield higher compression ratios.

[0006] A video encoder and decoder can utilize techniques from several broad categories, including, for example, motion compensation, transform, quantization, and entropy coding.

[0007] Video codec technologies can include techniques known as intra coding. In intra coding, sample values are represented without reference to samples or other data from previously reconstructed reference pictures. In some video codecs, the picture is spatially subdivided into blocks of samples. When all blocks of samples are coded in intra mode, that picture can be an intra picture. Intra pictures and their derivations such as independent decoder refresh pictures, can be used to reset the decoder state and can, therefore, be used as the first picture in a coded video bitstream and a video session, or as a still image. The samples of an intra block can be exposed to a transform, and the transform coefficients can be quantized before entropy coding. Intra prediction can be a technique that minimizes sample values in the pre-transform domain. In some cases, the smaller the DC value after a transform is, and the smaller the AC coefficients are, the fewer the bits that are required at a given quantization step size to represent the block after entropy coding.

[0008] Traditional intra coding such as known from, for example MPEG-2 generation coding technologies, does not use intra prediction. However, some newer video compression technologies include techniques that attempt, from, for example, surrounding sample data and/or metadata obtained during the encoding/decoding of spatially neighboring, and preceding in decoding order, blocks of data. Such techniques are henceforth called "intra prediction" techniques. Note that in at least some cases, intra prediction is only using reference data from the current picture under reconstruction and not from reference pictures.

[0009] There can be many different forms of intra prediction. When more than one of such techniques can be used in a given video coding technology, the technique in use can be coded in an intra prediction mode. In certain cases, modes can have submodes and/or parameters, and those can be coded individually or included in the mode codeword. Which codeword to use for a given mode/submode/parameter combination can have an impact in the

coding efficiency gain through intra prediction, and so can the entropy coding technology used to translate the codewords into a bitstream.

[0010] A certain mode of intra prediction was introduced with H.264, refined in H.265, and further refined in newer coding technologies such as joint exploration model (JEM), versatile video coding (VVC), and benchmark set (BMS). A predictor block can be formed using neighboring sample values belonging to already available samples. Sample values of neighboring samples are copied into the predictor block according to a direction. A reference to the direction in use can be coded in the bitstream or may itself be predicted.

[0011] Referring to FIG. 1, depicted in the lower right is a subset of nine predictor directions known from H.265's 33 possible predictor directions (corresponding to the 33 angular modes of the 35 intra modes). The point where the arrows converge (101) represents the sample being predicted. The arrows represent the direction from which the sample is being predicted. For example, arrow (102) indicates that sample (101) is predicted from a sample or samples to the upper right, at a 45 degree angle from the horizontal. Similarly, arrow (103) indicates that sample (101) is predicted from a sample or samples to the lower left of sample (101), in a 22.5 degree angle from the horizontal.

[0012] Still referring to FIG. 1, on the top left there is depicted a square block (104) of 4 x 4 samples (indicated by a dashed, boldface line). The square block (104) includes 16 samples, each labelled with an "S", its position in the Y dimension (e.g., row index) and its position in the X dimension (e.g., column index). For example, sample S21 is the second sample in the Y dimension (from the top) and the first (from the left) sample in the X dimension. Similarly, sample S44 is the fourth sample in block (104) in both the Y and X dimensions. As the block is 4 x 4 samples in size, S44 is at the bottom right. Further shown are reference samples that follow a similar numbering scheme. A reference sample is labelled with an R, its Y position (e.g., row index) and X position (column index) relative to block (104). In both H.264 and H.265, prediction samples neighbor the block under reconstruction; therefore no negative values need to be used.

[0013] Intra picture prediction can work by copying reference sample values from the neighboring samples as appropriated by the signaled prediction direction. For example, assume the coded video bitstream includes signaling that, for this block, indicates a prediction direction consistent with arrow (102)—that is, samples are predicted from a prediction sample or samples to the upper right, at a 45 degree angle from the horizontal. In that case, samples S41, S32, S23,

and S14 are predicted from the same reference sample R05. Sample S44 is then predicted from reference sample R08.

[0014] In certain cases, the values of multiple reference samples may be combined, for example through interpolation, in order to calculate a reference sample; especially when the directions are not evenly divisible by 45 degrees.

[0015] The number of possible directions has increased as video coding technology has developed. In H.264 (year 2003), nine different direction could be represented. That increased to 33 in H.265 (year 2013), and JEM/VVC/BMS, at the time of disclosure, can support up to 65 directions. Experiments have been conducted to identify the most likely directions, and certain techniques in the entropy coding are used to represent those likely directions in a small number of bits, accepting a certain penalty for less likely directions. Further, the directions themselves can sometimes be predicted from neighboring directions used in neighboring, already decoded, blocks.

[0016] FIG. 2 shows a schematic (201) that depicts 65 intra prediction directions according to JEM to illustrate the increasing number of prediction directions over time.

[0017] The mapping of intra prediction directions bits in the coded video bitstream that represent the direction can be different from video coding technology to video coding technology; and can range, for example, from simple direct mappings of prediction direction to intra prediction mode, to codewords, to complex adaptive schemes involving most probable modes, and similar techniques. In all cases, however, there can be certain directions that are statistically less likely to occur in video content than certain other directions. As the goal of video compression is the reduction of redundancy, those less likely directions will, in a well working video coding technology, be represented by a larger number of bits than more likely directions.

[0018] Motion compensation can be a lossy compression technique and can relate to techniques where a block of sample data from a previously reconstructed picture or part thereof (reference picture), after being spatially shifted in a direction indicated by a motion vector (MV henceforth), is used for the prediction of a newly reconstructed picture or picture part. In some cases, the reference picture can be the same as the picture currently under reconstruction. MVs can have two dimensions X and Y, or three dimensions, the third being an indication of the reference picture in use (the latter, indirectly, can be a time dimension).

[0019] In some video compression techniques, an MV applicable to a certain area of sample data can be predicted from other MVs, for example from those related to another area of sample data spatially adjacent to the area under reconstruction, and preceding that MV in decoding order. Doing so can substantially reduce the amount of data required for coding the MV, thereby removing redundancy and increasing compression. MV prediction can work effectively, for example, because when coding an input video signal derived from a camera (known as natural video) there is a statistical likelihood that areas larger than the area to which a single MV is applicable move in a similar direction and, therefore, can in some cases be predicted using a similar motion vector derived from MVs of neighboring area. That results in the MV found for a given area to be similar or the same as the MV predicted from the surrounding MVs, and that in turn can be represented, after entropy coding, in a smaller number of bits than what would be used if coding the MV directly. In some cases, MV prediction can be an example of lossless compression of a signal (namely: the MVs) derived from the original signal (namely: the sample stream). In other cases, MV prediction itself can be lossy, for example because of rounding errors when calculating a predictor from several surrounding MVs.

[0020] Various MV prediction mechanisms are described in H.265/HEVC (ITU-T Rec. H.265, "High Efficiency Video Coding", December 2016). Out of the many MV prediction mechanisms that H.265 offers, described here is a technique henceforth referred to as "spatial merge".

[0021] Referring to FIG. 3, a current block (301) comprises samples that have been found by the encoder during the motion search process to be predictable from a previous block of the same size that has been spatially shifted. Instead of coding that MV directly, the MV can be derived from metadata associated with one or more reference pictures, for example from the most recent (in decoding order) reference picture, using the MV associated with either one of five surrounding samples, denoted A0, A1, and B0, B1, B2 (302 through 306, respectively). In H.265, the MV prediction can use predictors from the same reference picture that the neighboring block is using.

SUMMARY

[0022] Aspects of the disclosure provide methods and apparatuses for video encoding/decoding. In some examples, an apparatus for video decoding includes receiving circuitry and processing circuitry. In some embodiments, processing circuitry decodes, prediction information for a block in an I slice from a coded video bitstream, and determines,

whether an intra block copy (IBC) mode is possible for the block in the I slice based on the prediction information and a size restriction of the IBC mode. Further, in an embodiment, the processing circuitry decodes, in response to the IBC mode being possible for the block in the I slice, a flag that indicates whether a skip mode is applied on the block from the coded video bitstream. Then, the processing circuitry reconstructs the block at least partially based on the flag.

[0023] Further, in some examples, the processing circuitry infers the flag in response to the IBC mode being impossible for the block in the I slice.

[0024] In some embodiments, the processing circuitry determines that the IBC mode is impossible for the block in the I slice in response to a size of the block in the I slice being larger than a threshold. In an embodiment, the processing circuitry determines that the IBC mode is impossible for the block in the I slice in response to at least one of a width of the block and a height of the block being larger than the threshold. In some examples, the processing circuitry determines an existence of the flag in the coded video bitstream based on an added condition that compares the size of the block with the threshold. The added condition excludes the existence of the flag in response to the size of the block in the I slice being larger than the threshold.

[0025] In some examples, the processing circuitry determines an existence of the flag in the coded video bitstream based on an existing condition that is modified to base on a comparison of the size and the threshold. In an example, the existing condition determines whether the block is an inter coded block. In another example, the existing condition determines whether the IBC mode is enabled.

[0026] Aspects of the disclosure also provide a non-transitory computer-readable medium storing instructions which when executed by a computer for video decoding cause the computer to perform the method for video decoding.

BRIEF DESCRIPTION OF THE DRAWINGS

[0027] Further features, the nature, and various advantages of the disclosed subject matter will be more apparent from the following detailed description and the accompanying drawings in which:

[0028] FIG. 1 is a schematic illustration of an exemplary subset of intra prediction modes.

[0029] FIG. 2 is an illustration of exemplary intra prediction directions.

[0030] FIG. 3 is a schematic illustration of a current block and its surrounding spatial merge candidates in one example.

[0031] FIG. 4 is a schematic illustration of a simplified block diagram of a communication system (400) in accordance with an embodiment.

[0032] FIG. 5 is a schematic illustration of a simplified block diagram of a communication system (500) in accordance with an embodiment.

[0033] FIG. 6 is a schematic illustration of a simplified block diagram of a decoder in accordance with an embodiment.

[0034] FIG. 7 is a schematic illustration of a simplified block diagram of an encoder in accordance with an embodiment.

[0035] FIG. 8 shows a block diagram of an encoder in accordance with another embodiment.

[0036] FIG. 9 shows a block diagram of a decoder in accordance with another embodiment.

[0037] FIG. 10 shows an example of intra block copy according to an embodiment of the disclosure.

[0038] FIG. 11 shows an exemplary syntax table for signaling prediction mode at the coding unit level.

[0039] FIGs. 12A-12E show an exemplary syntax table of coding tree unit level.

[0040] FIG. 13 shows an exemplary syntax table at coding unit level according to some embodiments of the disclosure.

[0041] FIG. 14 shows an exemplary syntax table at coding tree level according to some embodiments of the disclosure.

[0042] FIG. 15 shows a flow chart outlining a process example according to some embodiments of the disclosure.

[0043] FIG. 16 is a schematic illustration of a computer system in accordance with an embodiment.

DETAILED DESCRIPTION OF EMBODIMENTS

[0044] FIG. 4 illustrates a simplified block diagram of a communication system (400) according to an embodiment of the present disclosure. The communication system (400) includes a plurality of terminal devices that can communicate with each other, via, for example, a network (450). For example, the communication system (400) includes a first pair of terminal

devices (410) and (420) interconnected via the network (450). In the FIG. 4 example, the first pair of terminal devices (410) and (420) performs unidirectional transmission of data. For example, the terminal device (410) may code video data (e.g., a stream of video pictures that are captured by the terminal device (410)) for transmission to the other terminal device (420) via the network (450). The encoded video data can be transmitted in the form of one or more coded video bitstreams. The terminal device (420) may receive the coded video data from the network (450), decode the coded video data to recover the video pictures and display video pictures according to the recovered video data. Unidirectional data transmission may be common in media serving applications and the like.

[0045] In another example, the communication system (400) includes a second pair of terminal devices (430) and (440) that performs bidirectional transmission of coded video data that may occur, for example, during videoconferencing. For bidirectional transmission of data, in an example, each terminal device of the terminal devices (430) and (440) may code video data (e.g., a stream of video pictures that are captured by the terminal device) for transmission to the other terminal device of the terminal devices (430) and (440) via the network (450). Each terminal device of the terminal devices (430) and (440) also may receive the coded video data transmitted by the other terminal device of the terminal devices (430) and (440), and may decode the coded video data to recover the video pictures and may display video pictures at an accessible display device according to the recovered video data.

[0046] In the FIG. 4 example, the terminal devices (410), (420), (430) and (440) may be illustrated as servers, personal computers and smart phones but the principles of the present disclosure may be not so limited. Embodiments of the present disclosure find application with laptop computers, tablet computers, media players and/or dedicated video conferencing equipment. The network (450) represents any number of networks that convey coded video data among the terminal devices (410), (420), (430) and (440), including for example wireline (wired) and/or wireless communication networks. The communication network (450) may exchange data in circuit-switched and/or packet-switched channels. Representative networks include telecommunications networks, local area networks, wide area networks and/or the Internet. For the purposes of the present discussion, the architecture and topology of the network (450) may be immaterial to the operation of the present disclosure unless explained herein below.

[0047] FIG. 5 illustrates, as an example for an application for the disclosed subject matter, the placement of a video encoder and a video decoder in a streaming environment. The

disclosed subject matter can be equally applicable to other video enabled applications, including, for example, video conferencing, digital TV, storing of compressed video on digital media including CD, DVD, memory stick and the like, and so on.

[0048] A streaming system may include a capture subsystem (513), that can include a video source (501), for example a digital camera, creating for example a stream of video pictures (502) that are uncompressed. In an example, the stream of video pictures (502) includes samples that are taken by the digital camera. The stream of video pictures (502), depicted as a bold line to emphasize a high data volume when compared to encoded video data (504) (or coded video bitstreams), can be processed by an electronic device (520) that includes a video encoder (503) coupled to the video source (501). The video encoder (503) can include hardware, software, or a combination thereof to enable or implement aspects of the disclosed subject matter as described in more detail below. The encoded video data (504) (or encoded video bitstream (504)), depicted as a thin line to emphasize the lower data volume when compared to the stream of video pictures (502), can be stored on a streaming server (505) for future use. One or more streaming client subsystems, such as client subsystems (506) and (508) in FIG. 5 can access the streaming server (505) to retrieve copies (507) and (509) of the encoded video data (504). A client subsystem (506) can include a video decoder (510), for example, in an electronic device (530). The video decoder (510) decodes the incoming copy (507) of the encoded video data and creates an outgoing stream of video pictures (511) that can be rendered on a display (512) (e.g., display screen) or other rendering device (not depicted). In some streaming systems, the encoded video data (504), (507), and (509) (e.g., video bitstreams) can be encoded according to certain video coding/compression standards. Examples of those standards include ITU-T Recommendation H.265. In an example, a video coding standard under development is informally known as Versatile Video Coding (VVC). The disclosed subject matter may be used in the context of VVC.

[0049] It is noted that the electronic devices (520) and (530) can include other components (not shown). For example, the electronic device (520) can include a video decoder (not shown) and the electronic device (530) can include a video encoder (not shown) as well.

[0050] FIG. 6 shows a block diagram of a video decoder (610) according to an embodiment of the present disclosure. The video decoder (610) can be included in an electronic device (630). The electronic device (630) can include a receiver (631) (e.g., receiving circuitry).

The video decoder (610) can be used in the place of the video decoder (510) in the FIG. 5 example.

[0051] The receiver (631) may receive one or more coded video sequences to be decoded by the video decoder (610); in the same or another embodiment, one coded video sequence at a time, where the decoding of each coded video sequence is independent from other coded video sequences. The coded video sequence may be received from a channel (601), which may be a hardware/software link to a storage device which stores the encoded video data. The receiver (631) may receive the encoded video data with other data, for example, coded audio data and/or ancillary data streams, that may be forwarded to their respective using entities (not depicted). The receiver (631) may separate the coded video sequence from the other data. To combat network jitter, a buffer memory (615) may be coupled in between the receiver (631) and an entropy decoder / parser (620) ("parser (620)" henceforth). In certain applications, the buffer memory (615) is part of the video decoder (610). In others, it can be outside of the video decoder (610) (not depicted). In still others, there can be a buffer memory (not depicted) outside of the video decoder (610), for example to combat network jitter, and in addition another buffer memory (615) inside the video decoder (610), for example to handle playout timing. When the receiver (631) is receiving data from a store/forward device of sufficient bandwidth and controllability, or from an isosynchronous network, the buffer memory (615) may not be needed, or can be small. For use on best effort packet networks such as the Internet, the buffer memory (615) may be required, can be comparatively large and can be advantageously of adaptive size, and may at least partially be implemented in an operating system or similar elements (not depicted) outside of the video decoder (610).

[0052] The video decoder (610) may include the parser (620) to reconstruct symbols (621) from the coded video sequence. Categories of those symbols include information used to manage operation of the video decoder (610), and potentially information to control a rendering device such as a render device (612) (e.g., a display screen) that is not an integral part of the electronic device (630) but can be coupled to the electronic device (630), as was shown in FIG. 6. The control information for the rendering device(s) may be in the form of Supplemental Enhancement Information (SEI messages) or Video Usability Information (VUI) parameter set fragments (not depicted). The parser (620) may parse / entropy-decode the coded video sequence that is received. The coding of the coded video sequence can be in accordance with a video coding technology or standard, and can follow various principles, including variable length

coding, Huffman coding, arithmetic coding with or without context sensitivity, and so forth. The parser (620) may extract from the coded video sequence, a set of subgroup parameters for at least one of the subgroups of pixels in the video decoder, based upon at least one parameter corresponding to the group. Subgroups can include Groups of Pictures (GOPs), pictures, tiles, slices, macroblocks, Coding Units (CUs), blocks, Transform Units (TUs), Prediction Units (PUs) and so forth. The parser (620) may also extract from the coded video sequence information such as transform coefficients, quantizer parameter values, motion vectors, and so forth.

[0053] The parser (620) may perform an entropy decoding / parsing operation on the video sequence received from the buffer memory (615), so as to create symbols (621).

[0054] Reconstruction of the symbols (621) can involve multiple different units depending on the type of the coded video picture or parts thereof (such as: inter and intra picture, inter and intra block), and other factors. Which units are involved, and how, can be controlled by the subgroup control information that was parsed from the coded video sequence by the parser (620). The flow of such subgroup control information between the parser (620) and the multiple units below is not depicted for clarity.

[0055] Beyond the functional blocks already mentioned, the video decoder (610) can be conceptually subdivided into a number of functional units as described below. In a practical implementation operating under commercial constraints, many of these units interact closely with each other and can, at least partly, be integrated into each other. However, for the purpose of describing the disclosed subject matter, the conceptual subdivision into the functional units below is appropriate.

[0056] A first unit is the scaler / inverse transform unit (651). The scaler / inverse transform unit (651) receives a quantized transform coefficient as well as control information, including which transform to use, block size, quantization factor, quantization scaling matrices, etc. as symbol(s) (621) from the parser (620). The scaler / inverse transform unit (651) can output blocks comprising sample values, that can be input into aggregator (655).

[0057] In some cases, the output samples of the scaler / inverse transform (651) can pertain to an intra coded block; that is: a block that is not using predictive information from previously reconstructed pictures, but can use predictive information from previously reconstructed parts of the current picture. Such predictive information can be provided by an intra picture prediction unit (652). In some cases, the intra picture prediction unit (652) generates a block of the same size and shape of the block under reconstruction, using

surrounding already reconstructed information fetched from the current picture buffer (658). The current picture buffer (658) buffers, for example, partly reconstructed current picture and/or fully reconstructed current picture. The aggregator (655), in some cases, adds, on a per sample basis, the prediction information the intra prediction unit (652) has generated to the output sample information as provided by the scaler / inverse transform unit (651).

[0058] In other cases, the output samples of the scaler / inverse transform unit (651) can pertain to an inter coded, and potentially motion compensated block. In such a case, a motion compensation prediction unit (653) can access reference picture memory (657) to fetch samples used for prediction. After motion compensating the fetched samples in accordance with the symbols (621) pertaining to the block, these samples can be added by the aggregator (655) to the output of the scaler / inverse transform unit (651) (in this case called the residual samples or residual signal) so as to generate output sample information. The addresses within the reference picture memory (657) from where the motion compensation prediction unit (653) fetches prediction samples can be controlled by motion vectors, available to the motion compensation prediction unit (653) in the form of symbols (621) that can have, for example X, Y, and reference picture components. Motion compensation also can include interpolation of sample values as fetched from the reference picture memory (657) when sub-sample exact motion vectors are in use, motion vector prediction mechanisms, and so forth.

[0059] The output samples of the aggregator (655) can be subject to various loop filtering techniques in the loop filter unit (656). Video compression technologies can include in-loop filter technologies that are controlled by parameters included in the coded video sequence (also referred to as coded video bitstream) and made available to the loop filter unit (656) as symbols (621) from the parser (620), but can also be responsive to meta-information obtained during the decoding of previous (in decoding order) parts of the coded picture or coded video sequence, as well as responsive to previously reconstructed and loop-filtered sample values.

[0060] The output of the loop filter unit (656) can be a sample stream that can be output to the render device (612) as well as stored in the reference picture memory (657) for use in future inter-picture prediction.

[0061] Certain coded pictures, once fully reconstructed, can be used as reference pictures for future prediction. For example, once a coded picture corresponding to a current picture is fully reconstructed and the coded picture has been identified as a reference picture (by, for example, the parser (620)), the current picture buffer (658) can become a part of the reference

picture memory (657), and a fresh current picture buffer can be reallocated before commencing the reconstruction of the following coded picture.

[0062] The video decoder (610) may perform decoding operations according to a predetermined video compression technology in a standard, such as ITU-T Rec. H.265. The coded video sequence may conform to a syntax specified by the video compression technology or standard being used, in the sense that the coded video sequence adheres to both the syntax of the video compression technology or standard and the profiles as documented in the video compression technology or standard. Specifically, a profile can select certain tools as the only tools available for use under that profile from all the tools available in the video compression technology or standard. Also necessary for compliance can be that the complexity of the coded video sequence is within bounds as defined by the level of the video compression technology or standard. In some cases, levels restrict the maximum picture size, maximum frame rate, maximum reconstruction sample rate (measured in, for example megasamples per second), maximum reference picture size, and so on. Limits set by levels can, in some cases, be further restricted through Hypothetical Reference Decoder (HRD) specifications and metadata for HRD buffer management signaled in the coded video sequence.

[0063] In an embodiment, the receiver (631) may receive additional (redundant) data with the encoded video. The additional data may be included as part of the coded video sequence(s). The additional data may be used by the video decoder (610) to properly decode the data and/or to more accurately reconstruct the original video data. Additional data can be in the form of, for example, temporal, spatial, or signal noise ratio (SNR) enhancement layers, redundant slices, redundant pictures, forward error correction codes, and so on.

[0064] FIG. 7 shows a block diagram of a video encoder (703) according to an embodiment of the present disclosure. The video encoder (703) is included in an electronic device (720). The electronic device (720) includes a transmitter (740) (e.g., transmitting circuitry). The video encoder (703) can be used in the place of the video encoder (503) in the FIG. 5 example.

[0065] The video encoder (703) may receive video samples from a video source (701) (that is not part of the electronic device (720) in the FIG. 7 example) that may capture video image(s) to be coded by the video encoder (703). In another example, the video source (701) is a part of the electronic device (720).

[0066] The video source (701) may provide the source video sequence to be coded by the video encoder (703) in the form of a digital video sample stream that can be of any suitable bit depth (for example: 8 bit, 10 bit, 12 bit, ...), any colorspace (for example, BT.601 Y CrCB, RGB, ...), and any suitable sampling structure (for example Y CrCb 4:2:0, Y CrCb 4:4:4). In a media serving system, the video source (701) may be a storage device storing previously prepared video. In a videoconferencing system, the video source (701) may be a camera that captures local image information as a video sequence. Video data may be provided as a plurality of individual pictures that impart motion when viewed in sequence. The pictures themselves may be organized as a spatial array of pixels, wherein each pixel can comprise one or more samples depending on the sampling structure, color space, etc. in use. A person skilled in the art can readily understand the relationship between pixels and samples. The description below focuses on samples.

[0067] According to an embodiment, the video encoder (703) may code and compress the pictures of the source video sequence into a coded video sequence (743) in real time or under any other time constraints as required by the application. Enforcing appropriate coding speed is one function of a controller (750). In some embodiments, the controller (750) controls other functional units as described below and is functionally coupled to the other functional units. The coupling is not depicted for clarity. Parameters set by the controller (750) can include rate control related parameters (picture skip, quantizer, lambda value of rate-distortion optimization techniques, ...), picture size, group of pictures (GOP) layout, maximum motion vector search range, and so forth. The controller (750) can be configured to have other suitable functions that pertain to the video encoder (703) optimized for a certain system design.

[0068] In some embodiments, the video encoder (703) is configured to operate in a coding loop. As an oversimplified description, in an example, the coding loop can include a source coder (730) (e.g., responsible for creating symbols, such as a symbol stream, based on an input picture to be coded, and a reference picture(s)), and a (local) decoder (733) embedded in the video encoder (703). The decoder (733) reconstructs the symbols to create the sample data in a similar manner as a (remote) decoder also would create (as any compression between symbols and coded video bitstream is lossless in the video compression technologies considered in the disclosed subject matter). The reconstructed sample stream (sample data) is input to the reference picture memory (734). As the decoding of a symbol stream leads to bit-exact results independent of decoder location (local or remote), the content in the reference picture memory

(734) is also bit exact between the local encoder and remote encoder. In other words, the prediction part of an encoder "sees" as reference picture samples exactly the same sample values as a decoder would "see" when using prediction during decoding. This fundamental principle of reference picture synchronicity (and resulting drift, if synchronicity cannot be maintained, for example because of channel errors) is used in some related arts as well.

[0069] The operation of the "local" decoder (733) can be the same as of a "remote" decoder, such as the video decoder (610), which has already been described in detail above in conjunction with FIG. 6. Briefly referring also to FIG. 6, however, as symbols are available and encoding/decoding of symbols to a coded video sequence by an entropy coder (745) and the parser (620) can be lossless, the entropy decoding parts of the video decoder (610), including the buffer memory (615), and parser (620) may not be fully implemented in the local decoder (733).

[0070] An observation that can be made at this point is that any decoder technology except the parsing/entropy decoding that is present in a decoder also necessarily needs to be present, in substantially identical functional form, in a corresponding encoder. For this reason, the disclosed subject matter focuses on decoder operation. The description of encoder technologies can be abbreviated as they are the inverse of the comprehensively described decoder technologies. Only in certain areas a more detail description is required and provided below.

[0071] During operation, in some examples, the source coder (730) may perform motion compensated predictive coding, which codes an input picture predictively with reference to one or more previously coded picture from the video sequence that were designated as "reference pictures." In this manner, the coding engine (732) codes differences between pixel blocks of an input picture and pixel blocks of reference picture(s) that may be selected as prediction reference(s) to the input picture.

[0072] The local video decoder (733) may decode coded video data of pictures that may be designated as reference pictures, based on symbols created by the source coder (730). Operations of the coding engine (732) may advantageously be lossy processes. When the coded video data may be decoded at a video decoder (not shown in FIG. 7), the reconstructed video sequence typically may be a replica of the source video sequence with some errors. The local video decoder (733) replicates decoding processes that may be performed by the video decoder on reference pictures and may cause reconstructed reference pictures to be stored in the reference picture cache (734). In this manner, the video encoder (703) may store copies of reconstructed

reference pictures locally that have common content as the reconstructed reference pictures that will be obtained by a far-end video decoder (absent transmission errors).

[0073] The predictor (735) may perform prediction searches for the coding engine (732). That is, for a new picture to be coded, the predictor (735) may search the reference picture memory (734) for sample data (as candidate reference pixel blocks) or certain metadata such as reference picture motion vectors, block shapes, and so on, that may serve as an appropriate prediction reference for the new pictures. The predictor (735) may operate on a sample block-by-pixel block basis to find appropriate prediction references. In some cases, as determined by search results obtained by the predictor (735), an input picture may have prediction references drawn from multiple reference pictures stored in the reference picture memory (734).

[0074] The controller (750) may manage coding operations of the source coder (730), including, for example, setting of parameters and subgroup parameters used for encoding the video data.

[0075] Output of all aforementioned functional units may be subjected to entropy coding in the entropy coder (745). The entropy coder (745) translates the symbols as generated by the various functional units into a coded video sequence, by lossless compressing the symbols according to technologies such as Huffman coding, variable length coding, arithmetic coding, and so forth.

[0076] The transmitter (740) may buffer the coded video sequence(s) as created by the entropy coder (745) to prepare for transmission via a communication channel (760), which may be a hardware/software link to a storage device which would store the encoded video data. The transmitter (740) may merge coded video data from the video coder (703) with other data to be transmitted, for example, coded audio data and/or ancillary data streams (sources not shown).

[0077] The controller (750) may manage operation of the video encoder (703). During coding, the controller (750) may assign to each coded picture a certain coded picture type, which may affect the coding techniques that may be applied to the respective picture. For example, pictures often may be assigned as one of the following picture types:

[0078] An Intra Picture (I picture) may be one that may be coded and decoded without using any other picture in the sequence as a source of prediction. Some video codecs allow for different types of intra pictures, including, for example Independent Decoder Refresh ("IDR") Pictures. A person skilled in the art is aware of those variants of I pictures and their respective applications and features.

[0079] A predictive picture (P picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most one motion vector and reference index to predict the sample values of each block.

[0080] A bi-directionally predictive picture (B Picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most two motion vectors and reference indices to predict the sample values of each block. Similarly, multiple-predictive pictures can use more than two reference pictures and associated metadata for the reconstruction of a single block.

[0081] Source pictures commonly may be subdivided spatially into a plurality of sample blocks (for example, blocks of 4x4, 8x8, 4x8, or 16x16 samples each) and coded on a block-by-block basis. Blocks may be coded predictively with reference to other (already coded) blocks as determined by the coding assignment applied to the blocks' respective pictures. For example, blocks of I pictures may be coded non-predictively or they may be coded predictively with reference to already coded blocks of the same picture (spatial prediction or intra prediction). Pixel blocks of P pictures may be coded predictively, via spatial prediction or via temporal prediction with reference to one previously coded reference picture. Blocks of B pictures may be coded predictively, via spatial prediction or via temporal prediction with reference to one or two previously coded reference pictures.

[0082] The video encoder (703) may perform coding operations according to a predetermined video coding technology or standard, such as ITU-T Rec. H.265. In its operation, the video encoder (703) may perform various compression operations, including predictive coding operations that exploit temporal and spatial redundancies in the input video sequence. The coded video data, therefore, may conform to a syntax specified by the video coding technology or standard being used.

[0083] In an embodiment, the transmitter (740) may transmit additional data with the encoded video. The source coder (730) may include such data as part of the coded video sequence. Additional data may comprise temporal/spatial/SNR enhancement layers, other forms of redundant data such as redundant pictures and slices, SEI messages, VUI parameter set fragments, and so on.

[0084] A video may be captured as a plurality of source pictures (video pictures) in a temporal sequence. Intra-picture prediction (often abbreviated to intra prediction) makes use of spatial correlation in a given picture, and inter-picture prediction makes uses of the (temporal or

other) correlation between the pictures. In an example, a specific picture under encoding/decoding, which is referred to as a current picture, is partitioned into blocks. When a block in the current picture is similar to a reference block in a previously coded and still buffered reference picture in the video, the block in the current picture can be coded by a vector that is referred to as a motion vector. The motion vector points to the reference block in the reference picture, and can have a third dimension identifying the reference picture, in case multiple reference pictures are in use.

[0085] In some embodiments, a bi-prediction technique can be used in the inter-picture prediction. According to the bi-prediction technique, two reference pictures, such as a first reference picture and a second reference picture that are both prior in decoding order to the current picture in the video (but may be in the past and future, respectively, in display order) are used. A block in the current picture can be coded by a first motion vector that points to a first reference block in the first reference picture, and a second motion vector that points to a second reference block in the second reference picture. The block can be predicted by a combination of the first reference block and the second reference block.

[0086] Further, a merge mode technique can be used in the inter-picture prediction to improve coding efficiency.

[0087] According to some embodiments of the disclosure, predictions, such as inter-picture predictions and intra-picture predictions are performed in the unit of blocks. For example, according to the HEVC standard, a picture in a sequence of video pictures is partitioned into coding tree units (CTU) for compression, the CTUs in a picture have the same size, such as 64x64 pixels, 32x32 pixels, or 16x16 pixels. In general, a CTU includes three coding tree blocks (CTBs), which are one luma CTB and two chroma CTBs. Each CTU can be recursively quadtree split into one or multiple coding units (CUs). For example, a CTU of 64x64 pixels can be split into one CU of 64x64 pixels, or 4 CUs of 32x32 pixels, or 16 CUs of 16x16 pixels. In an example, each CU is analyzed to determine a prediction type for the CU, such as an inter prediction type or an intra prediction type. The CU is split into one or more prediction units (PUs) depending on the temporal and/or spatial predictability. Generally, each PU includes a luma prediction block (PB), and two chroma PBs. In an embodiment, a prediction operation in coding (encoding/decoding) is performed in the unit of a prediction block. Using a luma prediction block as an example of a prediction block, the prediction block includes a matrix of

values (e.g., luma values) for pixels, such as 8x8 pixels, 16x16 pixels, 8x16 pixels, 16x8 pixels, and the like.

[0088] FIG. 8 shows a diagram of a video encoder (803) according to another embodiment of the disclosure. The video encoder (803) is configured to receive a processing block (e.g., a prediction block) of sample values within a current video picture in a sequence of video pictures, and encode the processing block into a coded picture that is part of a coded video sequence. In an example, the video encoder (803) is used in the place of the video encoder (503) in the FIG. 5 example.

[0089] In an HEVC example, the video encoder (803) receives a matrix of sample values for a processing block, such as a prediction block of 8x8 samples, and the like. The video encoder (803) determines whether the processing block is best coded using intra mode, inter mode, or bi-prediction mode using, for example, rate-distortion optimization. When the processing block is to be coded in intra mode, the video encoder (803) may use an intra prediction technique to encode the processing block into the coded picture; and when the processing block is to be coded in inter mode or bi-prediction mode, the video encoder (803) may use an inter prediction or bi-prediction technique, respectively, to encode the processing block into the coded picture. In certain video coding technologies, merge mode can be an inter picture prediction submode where the motion vector is derived from one or more motion vector predictors without the benefit of a coded motion vector component outside the predictors. In certain other video coding technologies, a motion vector component applicable to the subject block may be present. In an example, the video encoder (803) includes other components, such as a mode decision module (not shown) to determine the mode of the processing blocks.

[0090] In the FIG. 8 example, the video encoder (803) includes the inter encoder (830), an intra encoder (822), a residue calculator (823), a switch (826), a residue encoder (824), a general controller (821), and an entropy encoder (825) coupled together as shown in FIG. 8.

[0091] The inter encoder (830) is configured to receive the samples of the current block (e.g., a processing block), compare the block to one or more reference blocks in reference pictures (e.g., blocks in previous pictures and later pictures), generate inter prediction information (e.g., description of redundant information according to inter encoding technique, motion vectors, merge mode information), and calculate inter prediction results (e.g., predicted block) based on the inter prediction information using any suitable technique. In some examples,

the reference pictures are decoded reference pictures that are decoded based on the encoded video information.

[0092] The intra encoder (822) is configured to receive the samples of the current block (e.g., a processing block), in some cases compare the block to blocks already coded in the same picture, generate quantized coefficients after transform, and in some cases also intra prediction information (e.g., an intra prediction direction information according to one or more intra encoding techniques). In an example, the intra encoder (822) also calculates intra prediction results (e.g., predicted block) based on the intra prediction information and reference blocks in the same picture.

[0093] The general controller (821) is configured to determine general control data and control other components of the video encoder (803) based on the general control data. In an example, the general controller (821) determines the mode of the block, and provides a control signal to the switch (826) based on the mode. For example, when the mode is the intra mode, the general controller (821) controls the switch (826) to select the intra mode result for use by the residue calculator (823), and controls the entropy encoder (825) to select the intra prediction information and include the intra prediction information in the bitstream; and when the mode is the inter mode, the general controller (821) controls the switch (826) to select the inter prediction result for use by the residue calculator (823), and controls the entropy encoder (825) to select the inter prediction information and include the inter prediction information in the bitstream.

[0094] The residue calculator (823) is configured to calculate a difference (residue data) between the received block and prediction results selected from the intra encoder (822) or the inter encoder (830). The residue encoder (824) is configured to operate based on the residue data to encode the residue data to generate the transform coefficients. In an example, the residue encoder (824) is configured to convert the residue data from a spatial domain to a frequency domain, and generate the transform coefficients. The transform coefficients are then subject to quantization processing to obtain quantized transform coefficients. In various embodiments, the video encoder (803) also includes a residue decoder (828). The residue decoder (828) is configured to perform inverse-transform, and generate the decoded residue data. The decoded residue data can be suitably used by the intra encoder (822) and the inter encoder (830). For example, the inter encoder (830) can generate decoded blocks based on the decoded residue data and inter prediction information, and the intra encoder (822) can generate decoded blocks based on the decoded residue data and the intra prediction information. The decoded blocks are

suitably processed to generate decoded pictures and the decoded pictures can be buffered in a memory circuit (not shown) and used as reference pictures in some examples.

[0095] The entropy encoder (825) is configured to format the bitstream to include the encoded block. The entropy encoder (825) is configured to include various information according to a suitable standard, such as the HEVC standard. In an example, the entropy encoder (825) is configured to include the general control data, the selected prediction information (e.g., intra prediction information or inter prediction information), the residue information, and other suitable information in the bitstream. Note that, according to the disclosed subject matter, when coding a block in the merge submode of either inter mode or bi-prediction mode, there is no residue information.

[0096] FIG. 9 shows a diagram of a video decoder (910) according to another embodiment of the disclosure. The video decoder (910) is configured to receive coded pictures that are part of a coded video sequence, and decode the coded pictures to generate reconstructed pictures. In an example, the video decoder (910) is used in the place of the video decoder (510) in the FIG. 5 example.

[0097] In the FIG. 9 example, the video decoder (910) includes an entropy decoder (971), an inter decoder (980), a residue decoder (973), a reconstruction module (974), and an intra decoder (972) coupled together as shown in FIG. 9.

[0098] The entropy decoder (971) can be configured to reconstruct, from the coded picture, certain symbols that represent the syntax elements of which the coded picture is made up. Such symbols can include, for example, the mode in which a block is coded (such as, for example, intra mode, inter mode, bi-predicted mode, the latter two in merge submode or another submode), prediction information (such as, for example, intra prediction information or inter prediction information) that can identify certain sample or metadata that is used for prediction by the intra decoder (972) or the inter decoder (980), respectively, residual information in the form of, for example, quantized transform coefficients, and the like. In an example, when the prediction mode is inter or bi-predicted mode, the inter prediction information is provided to the inter decoder (980); and when the prediction type is the intra prediction type, the intra prediction information is provided to the intra decoder (972). The residual information can be subject to inverse quantization and is provided to the residue decoder (973).

[0099] The inter decoder (980) is configured to receive the inter prediction information, and generate inter prediction results based on the inter prediction information.

[0100] The intra decoder (972) is configured to receive the intra prediction information, and generate prediction results based on the intra prediction information.

[0101] The residue decoder (973) is configured to perform inverse quantization to extract de-quantized transform coefficients, and process the de-quantized transform coefficients to convert the residual from the frequency domain to the spatial domain. The residue decoder (973) may also require certain control information (to include the Quantizer Parameter (QP)), and that information may be provided by the entropy decoder (971) (data path not depicted as this may be low volume control information only).

[0102] The reconstruction module (974) is configured to combine, in the spatial domain, the residual as output by the residue decoder (973) and the prediction results (as output by the inter or intra prediction modules as the case may be) to form a reconstructed block, that may be part of the reconstructed picture, which in turn may be part of the reconstructed video. It is noted that other suitable operations, such as a deblocking operation and the like, can be performed to improve the visual quality.

[0103] It is noted that the video encoders (503), (703), and (803), and the video decoders (510), (610), and (910) can be implemented using any suitable technique. In an embodiment, the video encoders (503), (703), and (803), and the video decoders (510), (610), and (910) can be implemented using one or more integrated circuits. In another embodiment, the video encoders (503), (703), and (703), and the video decoders (510), (610), and (910) can be implemented using one or more processors that execute software instructions.

[0104] Aspects of the disclosure provide signaling techniques for skip mode flag.

[0105] Block based compensation can be used for inter prediction and intra prediction. For the inter prediction, block based compensation from a different picture is known as motion compensation. For intra prediction, block based compensation can also be done from a previously reconstructed area within the same picture. The block based compensation from reconstructed area within the same picture is referred to as intra picture block compensation, current picture referencing (CPR) or intra block copy (IBC). A displacement vector that indicates the offset between the current block and the reference block in the same picture is referred to as a block vector (or BV for short). Different from a motion vector in motion compensation, which can be at any value (positive or negative, at either x or y direction), a block vector has a few constraints to ensure that the reference block is available and already

reconstructed. Also, in some examples, for parallel processing consideration, some reference area that is tile boundary or wavefront ladder shape boundary is excluded.

[0106] The coding of a block vector could be either explicit or implicit. In the explicit mode (or referred to as advanced motion vector prediction (AMVP) mode in inter coding), the difference between a block vector and its predictor is signaled; in the implicit mode, the block vector is recovered from a predictor (referred to as block vector predictor), in a similar way as a motion vector in merge mode. The resolution of a block vector, in some implementations, is restricted to integer positions; in other systems, the block vector is allowed to point to fractional positions.

[0107] In some examples, the use of intra block copy at block level, can be signaled using a block level flag that is referred to as an IBC flag. In an embodiment, the IBC flag is signaled when the current block is not coded in merge mode. In other examples, the use of the intra block copy at block level is signaled by a reference index approach. The current picture under decoding is then treated as a reference picture. In an example, such a reference picture is put in the last position of a list of reference pictures. This special reference picture is also managed together with other temporal reference pictures in a buffer, such as decoded picture buffer (DPB).

[0108] There are also some variations for intra block copy, such as flipped intra block copy (the reference block is flipped horizontally or vertically before used to predict current block), or line based intra block copy (each compensation unit inside an MxN coding block is an Mx1 or 1xN line).

[0109] FIG. 10 shows an example of intra block copy according to an embodiment of the disclosure. Current picture (1000) is under decoding. The current picture (1000) includes a reconstructed area (1010) (doted area) and to-be-decoded area (1020) (white area). A current block (1030) is under reconstruction by a decoder. The current block (1030) can be reconstructed from a reference block (1040) that is in the reconstructed area (1010). The position offset between the reference block (1040) and the current block (1030) is referred to as a block vector (1050) (or BV (1050)).

[0110] In some examples, the intra block copy is considered as a separate mode other than the intra prediction mode or the inter prediction mode, the signaling of intra block copy can be specified at block level using a combination of flags, such as pred_mode_ibc_flag and pred_mode_flag.

[0111] In a related example, a coding block may be coded in the intra prediction mode or the inter prediction mode. In the related example, a prediction mode flag "pred_mode_flag" with 1 bit can be signaled or inferred at coding block level to differentiate the coding modes for the current block. For example, if pred_mode_flag is equal to 0, MODE_INTER (inter prediction mode) is used; otherwise (pred_mode_flag is equal to 1), MODE_INTRA (intra prediction mode) is used.

[0112] When intra block copy is used, the mode decision can be determined based on both the values of pred_mode_flag and pred_mode_ibc_flag.

[0113] In some examples (e.g., a version of VVC), the maximum block size for an inter coded CU can be as large as 128x128; the maximum block size for an intra coded CU can be as large as 128x128; the maximum block size for an intra block copy coded CU can be as large as 64x64, all in luma coded samples. When luma/chroma separate coding tree is used, the largest possible chroma CU size in 4:2:0 color format will be 32x32.

[0114] In some examples, for blocks that are inter coded (coded using inter prediction mode) or IBC coded (coded using IBC mode), motion parameters can be predicted, for example, in a merge mode or a skip mode. In the merge mode, a video coder constructs a candidate list of motion parameters (e.g., reference pictures and motion vectors) using as candidates motion parameters from neighboring blocks, including spatial neighboring blocks and temporal neighboring blocks. The chosen motion parameters can be signaled from a video encoder to a video decoder by transmitting an index of the selected candidate from the candidate list. At the video decoder, once the index is decoded, the motion parameters of the corresponding block of the selected candidate can be inherited. The video encoder and the video decoder are configured to construct the same lists based on already coded blocks. Therefore, based on the index, the video decoder can identify the motion parameters of the candidate selected by the video encoder.

[0115] In the skip mode, the motion parameters can be similarly predicted as in the merge mode. Further, in the skip mode, no residual data is added to the predicted block, whereas in merge mode, residual data is added to the predicted block. The constructing of a list and transmitting of an index to identify a candidate in the list described above with reference to the merge mode is generally also performed in the skip mode. In some embodiments, a skip flag in the coding block level can be used to indicate whether a block is coded using skip mode or not.

[0116] FIG. 11 shows an exemplary syntax table (1100) for signaling prediction mode at the coding unit (or coding block) level. The syntax table (1100) conditionally extracts, from the

coded video bitstream, three flags, cu_skip_flag, pred_mode_flag and pred_mode_ibc_flag, as shown by (1101), (1102) and (1103) in FIG. 11.

[0117] Specifically, cu_skip_flag[x0][y0] is the skip flag. The skip flag cu_skip_flag[x0][y0] equal to 1 specifies that the current coding unit is coded in the skip mode. Thus, when the current coding unit is in a P (coded by inter prediction) or B (coded by bi directional inter prediction) slice, no more syntax elements except one or more of the following are parsed after cu_skip_flag[x0][y0]: the IBC mode flag pred_mode_ibc_flag [x0][y0], and the merge_data( ) syntax structure. When the current coding unit is in an I slice, no more syntax elements except merge_idx[x0][y0] are parsed after cu_skip_flag[x0][y0].

[0118] The skip flag cu_skip_flag[x0][y0] equal to 0 specifies that the coding unit is not in the skip mode. The array indices x0, y0 specify the location (x0, y0) of the top-left luma sample of the considered coding block relative to the top-left luma sample of the picture.

[0119] When cu_skip_flag[x0][y0] is not present in the coded video bitstream, cu_skip_flag[x0][y0] can be inferred to be equal to 0.

[0120] Further, pred_mode_flag equal to 0 specifies that the current coding unit is coded in the inter prediction mode; and pred_mode_flag equal to 1 specifies that the current coding unit is coded in intra prediction mode.

[0121] In some embodiments, pred_mode_flag is not present in the coded video bitstream, and pred_mode_flag can be inferred. The inference of the pred_mode_flag can be based on a variable modeType. Generally, the variable modeType specifies whether intra prediction mode (MODE_INTRA), IBC mode (MODE_IBC), palette mode (MODE_PLT), and inter prediction mode can be used (MODE_TYPE_ALL), or whether only intra, IBC and palette coding modes can be used (MODE_TYPE_INTRA), or whether only inter coding modes can be used (MODE_TYPE_INTER) for coding units inside the coding tree node. For example, when the variable modeType is MODE_TYPE_ALL, all of the intra prediction mode, IBC mode, palette mode, and inter prediction mode can be used; when the variable modeType is MODE_TYPE_INTRA, the intra prediction mode, IBC mode and palette mode can be used; and when the variable modeType is MODE_TYPE_INTER, only inter prediction mode can be used.

[0122] In some examples, the variable, modeType, is used to control the possible allowed prediction mode types for certain small CU sizes. At the CTU root (where coding_tree() syntax is firstly called), the value of this variable is set to be MODE_TYPE_ALL (meaning no constraints). Therefore, for a large CU, such as 128x128, 128x64, 64x128, modeType should be

set to MODE_TYPE_ALL. When the large CU is split into smaller CUs, the variable modeType for the small CUs can be refined, such as based on a variable treeType. In an embodiment, the coding tree scheme supports the ability for a luma component and corresponding chroma component(s) to have separate block tree structures. In an example, for P and B slices, luma and chroma CTBs in a CTU share a same coding tree structure (e.g., a single tree). For I slices, luma and chroma CTBs in a CTU can have separate block tree structures (e.g., dual tree), and the partition case of the CTU using separate block tree structures is referred to as dual tree partition. When dual tree partition is applied, a luma CTB can be partitioned into luma CUs by a luma coding tree structure (e.g., DUAL_TREE_LUMA), and chroma CTBs can be partitioned into chroma CUs by a chroma coding tree structure (e.g., DUAL_TREE_CHROMA). Thus, a CU in an I slice can include a coding block of the luma component and can include coding blocks of two chroma components, and a CU in a P or B slice includes coding blocks of all three color components unless the video is monochrome. In an example, the variable treeType specifies whether a single tree (SINGLE_TREE) or a dual tree is used for a coding unit split from a coding tree unit; when a dual tree is used, the variable treeType specifies whether the luma (DUAL_TREE_LUMA) or chroma components (DUAL_TREE_CHROMA) are currently processed.

[0123] In some embodiments, pre_mode_flag is be inferred as follows:

- If cbWidth (width of the coding unit) is equal to 4 and cbHeight (height of the coding unit) is equal to 4, pred_mode_flag is inferred to be equal to 1 due to the reason in some example, inter prediction mode is not applied to 4x4 blocks;
- otherwise, if modeType is equal to MODE_TYPE_INTRA, pred_mode_flag is inferred to be equal to 1;
- otherwise, if modeType is equal to MODE_TYPE_INTER, pred_mode_flag is inferred to be equal to 0;
- otherwise, pred_mode_flag is inferred to be equal to 1 when decoding an I slice, and equal to 0 when decoding a P or B slice, respectively.

[0124] The variable CuPredMode[chType][x][y] is prediction mode derived as follows for x = x0..x0 + cbWidth ? 1 and y = y0..y0 + cbHeight ? 1:

- If pred_mode_flag is equal to 0, CuPredMode[chType][x][y] is set equal to MODE_INTER (inter prediction mode);

- otherwise (pred_mode_flag is equal to 1), CuPredMode[chType][x][y] is set equal to MODE_INTRA (intra prediction mode).

[0125] Further, the flag pred_mode_ibc_flag equal to 1 specifies that the current coding unit is coded in IBC mode. The flag pred_mode_ibc_flag equal to 0 specifies that the current coding unit is not coded in IBC mode.

[0126] In some embodiments, pred_mode_ibc_flag is not present in the coded bitstream and can be inferred. The inference of pred_mode_ibc_flag can be based on the variable modeType and the variable treeType.

[0127] For example, the pred_mode_ibc_flag can be inferred as follows:

- If cu_skip_flag[x0][y0] is equal to 1, and cbWidth is equal to 4, and cbHeight is equal to 4, pred_mode_ibc_flag is inferred to be equal 1;
- otherwise, if either cbWidth or cbHeight are equal to 128, pred_mode_ibc_flag is inferred to be equal to 0;
- otherwise, if modeType is equal to MODE_TYPE_INTER, pred_mode_ibc_flag is inferred to be equal to 0;
- otherwise, if treeType is equal to DUAL_TREE_CHROMA, pred_mode_ibc_flag is inferred to be equal to 0;
- otherwise, pred_mode_ibc_flag is inferred to be equal to the value of sps_ibc_enabled_flag when decoding an I slice, and 0 when decoding a P or B slice, respectively.

[0128] When pred_mode_ibc_flag is equal to 1, the variable CuPredMode[chType][x][y] is set to be equal to MODE_IBC for x = x0..x0 + cbWidth - 1 and y = y0..y0 + cbHeight - 1.

[0129] FIGs. 12A-12E show an exemplary syntax table (1200) at coding tree unit level. In the example, the input parameter modeTypeCurr is the modeType variable inherited from its parent node.

[0130] It is noted that the syntax and semantics in some examples, such as in the syntax in FIG. 11, cannot correctly reflect the block size constraints applied among intra/inter/IBC modes. For example, conditions (1110) in FIG. 11 is equivalent to perform a logical or (| |) of a first condition (!(cbWidth = = 4 && cbHeight = = 4) && (modeType ! = = MODE_TYPE_INTRA)) and a second condition (sps_ibc_enabled_flag). The first condition checks whether the logical value of (!(cbWidth = = 4 && cbHeight = = 4) && (modeType ! = = MODE_TYPE_INTRA)) is 1 or 0. The second condition checks whether the logical value of

(sps_ibc_enabled_flag) is 1 or 0. The first condition being 1 specifies that the current block under coding is an inter coded block. The second condition being 1 specifies that IBC mode is enabled. In the FIG. 11 example, when at least one of the first condition and the second condition is equal to 1, and the treeType is not equal to DUAL_TREE_CHROMA, the skip flag (e.g., cu_skip_flag[x0][y0]) is presented in the coded video bitstream, and can be decoded. However, in an example, a coding unit in an I slice can have at least one size (width and/or height) larger than 64. Parameters of the coding unit can cause the first condition to be 1 and/or the second condition to be 1. However, the coding unit cannot be inter coded (due to in the I slice) and cannot be IBC coded (due to the reason that at least one size is larger than 64). Thus, the conditions (1110) cannot correctly reflect the block size constraints applied among intra/inter/IBC modes.

[0131] Some aspects of the disclosure provide techniques to correctly reflect the block size constrains applied among intra/inter/IBC modes. The following descriptions are based on the assumption that the intra block copy (IBC) mode is considered as a separate mode, different from intra or inter modes. The techniques can signal the skip flag (e.g., cu_skip_flag[x0][y0]) only when the IBC mode is possible in an I slice. If only intra mode is possible, the skip flag does not need to be signaled; instead, the skip flag can be inferred to be 0.

[0132] In an embodiment, additional condition(s) can be imposed on the signaling of skip flag such that when the CU block size suggests that only intra mode is possible (the block size exceed maximum IBC block size), the skip flag is not signaled (is not presented in the coded video bitstram). Instead, the skip flag can be inferred to be 0.

[0133] In an example, when a block in the I slice is larger than 64 luma samples at either width side or height side, the block cannot be inter coded and cannot be IBC coded, and thus only intra mode (intra prediction mode) is possible for the block.

[0134] FIG. 13 shows an exemplary syntax table (1300) according to some embodiments of the disclosure. As shown by (1310) in FIG. 13, third condition (!(slice_type = I && (cbWidth > 64 | | cbHeight > 64))) is added to combine (use logical and operator &&) with the first condition and the second condition. When a block in the I slice is larger than 64 luma samples at either width side or height side, the third condition can be 0. Then, the combination of the first condition, the second condition and the third condition is equal to 0. Thus, the skip flag (cu_skip_flag[x0][y0]) is not signaled (is not presented in the coded video bitstream).

[0135] In another embodiment, the first condition and/or the second condition can be suitably modified to apply the size restrictions for IBC mode. In some examples, the assignment of modeType is modified such that when the CU block size suggests that only intra mode is possible in I slice(the block size exceeds maximum IBC block size), the modeType is set to MODE_TYPE_INTRA that can cause the first condition to be equal to 0. In some embodiments, if (slice_type == I && (cbWidth > 64 || cbHeight > 64)), then the modeType is set to MODE_TYPE_INTRA.

[0136] FIG. 14 shows an exemplary syntax table (1400) at coding tree level according to some embodiments of the disclosure. As shown by (1410), when a block in an I slice is larger than 64 luma samples at either width side or height side, a variable modeTypeCurrr is set to MODE_TYPE_INTRA . The variable modeTypeCurr is the input parameter to coding_unit() syntax as shown by the syntaxes in FIG. 14 and FIG. 13. Therefore, in the coding_unit() syntax table, the result of the first condition is 0. In another example, in the coding_unit() syntax, the variable modeType is set to MODE_TYPE_INTRA before the first "if" clause when a block in an I slice is larger than 64 luma samples at either width side or height side. Thus, the result of the first condition is 0.

[0137] In some examples, the second conditions can be modified to apply the size restrictions for IBC mode. For example, the second condition is modified to be (sps_ibc_enabled_flag && cbWidth <= 64 && cbHeight <= 64). Thus, when a block in an I slice is larger than 64 luma samples at either width side or height side, the result of the second condition is 0.

[0138] It is noted that the embodiments in the present disclosure can be used separately or combined in any order.

[0139] FIG. 15 shows a flow chart outlining a process (1500) according to an embodiment of the disclosure. The process (1500) can be used in the reconstruction of a block, so to generate a prediction block for the block under reconstruction. In various embodiments, the process (1500) are executed by processing circuitry, such as the processing circuitry in the terminal devices (410), (420), (430) and (440), the processing circuitry that performs functions of the video encoder (503), the processing circuitry that performs functions of the video decoder (510), the processing circuitry that performs functions of the video decoder (610), the processing circuitry that performs functions of the video encoder (703), and the like. In some embodiments, the process (1500) is implemented in software instructions, thus when the

processing circuitry executes the software instructions, the processing circuitry performs the process (1500). The process starts at (S1501) and proceeds to (S1510).

[0140] At (S1510), prediction information for a block in an I slice is decoded from a coded video bitstream.

[0141] At (S1520), whether the IBC mode is possible for the block is determined based on the prediction information and a size restriction of the IBC mode. In some embodiments, the IBC mode is determined to be impossible for the block in the I slice in response to a size of the block being larger than a threshold. In an embodiment, the IBC mode is determined to be impossible for the block when at least one of a width or a height of the block is larger than the threshold.

[0142] In some embodiments, the possibility of the IBC mode for the block in the I slice determines whether a flag (skip mode flag) for the block is in the coded video bitstream. In some examples, the syntax includes some existing conditions to determine the existence of the skip mode flag. In an embodiment, the existence of the flag in the coded video bitstream is also determined based on an added condition that compares the size of the block with the threshold. The added condition excludes the existence of the flag in response to the size of the block in the I slice being larger than the threshold.

[0143] In another embodiment, the existence of the flag in the coded video bitstream is determined based on an existing condition that is modified to compare the size of the block with the threshold. In an example, the existing condition is the first condition that determines whether the block is an inter coded block, and the first condition is modified to base on a comparison of the size of the block with the threshold. In another example, the existing condition is the second condition that determines whether the IBC mode is enabled, and the second condition is modified to base on a comparison of the size of the block with the threshold.

[0144] At (S1530), a flag that indicates whether a skip mode is applied on the block is decoded from the bitstream in response to the IBC mode being possible for the block in the I slice in some examples. Further, in some examples, the flag can be inferred in response to the IBC mode being impossible for the block in the I slice.

[0145] At (S1540), the block is reconstructed at least partially based on the flag. In an example, when the flag indicates a skip mode, the block is reconstructed based on the skip mode. Then, the process proceeds to (S1599) and terminates.

[0146] The techniques described above, can be implemented as computer software using computer-readable instructions and physically stored in one or more computer-readable media. For example, FIG. 16 shows a computer system (1600) suitable for implementing certain embodiments of the disclosed subject matter.

[0147] The computer software can be coded using any suitable machine code or computer language, that may be subject to assembly, compilation, linking, or like mechanisms to create code comprising instructions that can be executed directly, or through interpretation, micro-code execution, and the like, by one or more computer central processing units (CPUs), Graphics Processing Units (GPUs), and the like.

[0148] The instructions can be executed on various types of computers or components thereof, including, for example, personal computers, tablet computers, servers, smartphones, gaming devices, internet of things devices, and the like.

[0149] The components shown in FIG. 16 for computer system (1600) are exemplary in nature and are not intended to suggest any limitation as to the scope of use or functionality of the computer software implementing embodiments of the present disclosure. Neither should the configuration of components be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary embodiment of a computer system (1600).

[0150] Computer system (1600) may include certain human interface input devices. Such a human interface input device may be responsive to input by one or more human users through, for example, tactile input (such as: keystrokes, swipes, data glove movements), audio input (such as: voice, clapping), visual input (such as: gestures), olfactory input (not depicted). The human interface devices can also be used to capture certain media not necessarily directly related to conscious input by a human, such as audio (such as: speech, music, ambient sound), images (such as: scanned images, photographic images obtain from a still image camera), video (such as two-dimensional video, three-dimensional video including stereoscopic video).

[0151] Input human interface devices may include one or more of (only one of each depicted): keyboard (1601), mouse (1602), trackpad (1603), touch screen (1610), data-glove (not shown), joystick (1605), microphone (1606), scanner (1607), camera (1608).

[0152] Computer system (1600) may also include certain human interface output devices. Such human interface output devices may be stimulating the senses of one or more human users through, for example, tactile output, sound, light, and smell/taste. Such human interface output

devices may include tactile output devices (for example tactile feedback by the touch-screen (1610), data-glove (not shown), or joystick (1605), but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as: speakers (1609), headphones (not depicted)), visual output devices (such as screens (1610) to include CRT screens, LCD screens, plasma screens, OLED screens, each with or without touch-screen input capability, each with or without tactile feedback capability—some of which may be capable to output two dimensional visual output or more than three dimensional output through means such as stereographic output; virtual-reality glasses (not depicted), holographic displays and smoke tanks (not depicted)), and printers (not depicted).

[0153] Computer system (1600) can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW (1620) with CD/DVD or the like media (1621), thumb-drive (1622), removable hard drive or solid state drive (1623), legacy magnetic media such as tape and floppy disc (not depicted), specialized ROM/ASIC/PLD based devices such as security dongles (not depicted), and the like.

[0154] Those skilled in the art should also understand that term "computer readable media" as used in connection with the presently disclosed subject matter does not encompass transmission media, carrier waves, or other transitory signals.

[0155] Computer system (1600) can also include an interface to one or more communication networks. Networks can for example be wireless, wireline, optical. Networks can further be local, wide-area, metropolitan, vehicular and industrial, real-time, delay-tolerant, and so on. Examples of networks include local area networks such as Ethernet, wireless LANs, cellular networks to include GSM, 3G, 4G, 5G, LTE and the like, TV wireline or wireless wide area digital networks to include cable TV, satellite TV, and terrestrial broadcast TV, vehicular and industrial to include CANBus, and so forth. Certain networks commonly require external network interface adapters that attached to certain general purpose data ports or peripheral buses (1649) (such as, for example USB ports of the computer system (1600)); others are commonly integrated into the core of the computer system (1600) by attachment to a system bus as described below (for example Ethernet interface into a PC computer system or cellular network interface into a smartphone computer system). Using any of these networks, computer system (1600) can communicate with other entities. Such communication can be uni-directional, receive only (for example, broadcast TV), uni-directional send-only (for example CANbus to certain CANbus devices), or bi-directional, for example to other computer systems using local or wide

area digital networks. Certain protocols and protocol stacks can be used on each of those networks and network interfaces as described above.

[0156] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core (1640) of the computer system (1600).

[0157] The core (1640) can include one or more Central Processing Units (CPU) (1641), Graphics Processing Units (GPU) (1642), specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) (1643), hardware accelerators for certain tasks (1644), and so forth. These devices, along with Read-only memory (ROM) (1645), Random-access memory (1646), internal mass storage such as internal non-user accessible hard drives, SSDs, and the like (1647), may be connected through a system bus (1648). In some computer systems, the system bus (1648) can be accessible in the form of one or more physical plugs to enable extensions by additional CPUs, GPU, and the like. The peripheral devices can be attached either directly to the core's system bus (1648), or through a peripheral bus (1649). Architectures for a peripheral bus include PCI, USB, and the like.

[0158] CPUs (1641), GPUs (1642), FPGAs (1643), and accelerators (1644) can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM (1645) or RAM (1646). Transitional data can be also be stored in RAM (1646), whereas permanent data can be stored for example, in the internal mass storage (1647). Fast storage and retrieve to any of the memory devices can be enabled through the use of cache memory, that can be closely associated with one or more CPU (1641), GPU (1642), mass storage (1647), ROM (1645), RAM (1646), and the like.

[0159] The computer readable media can have computer code thereon for performing various computer-implemented operations. The media and computer code can be those specially designed and constructed for the purposes of the present disclosure, or they can be of the kind well known and available to those having skill in the computer software arts.

[0160] As an example and not by way of limitation, the computer system having architecture (1600), and specifically the core (1640) can provide functionality as a result of processor(s) (including CPUs, GPUs, FPGA, accelerators, and the like) executing software embodied in one or more tangible, computer-readable media. Such computer-readable media can be media associated with user-accessible mass storage as introduced above, as well as certain storage of the core (1640) that are of non-transitory nature, such as core-internal mass storage (1647) or ROM (1645). The software implementing various embodiments of the present

disclosure can be stored in such devices and executed by core (1640). A computer-readable medium can include one or more memory devices or chips, according to particular needs. The software can cause the core (1640) and specifically the processors therein (including CPU, GPU, FPGA, and the like) to execute particular processes or particular parts of particular processes described herein, including defining data structures stored in RAM (1646) and modifying such data structures according to the processes defined by the software. In addition or as an alternative, the computer system can provide functionality as a result of logic hardwired or otherwise embodied in a circuit (for example: accelerator (1644)), which can operate in place of or together with software to execute particular processes or particular parts of particular processes described herein. Reference to software can encompass logic, and vice versa, where appropriate. Reference to a computer-readable media can encompass a circuit (such as an integrated circuit (IC)) storing software for execution, a circuit embodying logic for execution, or both, where appropriate. The present disclosure encompasses any suitable combination of hardware and software.

Appendix A: Acronyms

JEM: joint exploration model

VVC: versatile video coding

BMS: benchmark set

MV: Motion Vector

HEVC: High Efficiency Video Coding

SEI: Supplementary Enhancement Information

VUI: Video Usability Information

GOPs: Groups of Pictures

TUs: Transform Units,

PUs: Prediction Units

CTUs: Coding Tree Units

CTBs: Coding Tree Blocks

PBs: Prediction Blocks

HRD: Hypothetical Reference Decoder

SNR: Signal Noise Ratio

CPUs: Central Processing Units

GPUs: Graphics Processing Units

CRT: Cathode Ray Tube

LCD: Liquid-Crystal Display

OLED: Organic Light-Emitting Diode

CD: Compact Disc

DVD: Digital Video Disc

ROM: Read-Only Memory

RAM: Random Access Memory

ASIC: Application-Specific Integrated Circuit

PLD: Programmable Logic Device

LAN: Local Area Network

GSM: Global System for Mobile communications

LTE: Long-Term Evolution

CANBus: Controller Area Network Bus

USB: Universal Serial Bus

PCI: Peripheral Component Interconnect

FPGA: Field Programmable Gate Areas

SSD: solid-state drive

IC: Integrated Circuit

CU: Coding Unit

[0161] While this disclosure has described several exemplary embodiments, there are alterations, permutations, and various substitute equivalents, which fall within the scope of the disclosure. It will thus be appreciated that those skilled in the art will be able to devise numerous systems and methods which, although not explicitly shown or described herein, embody the principles of the disclosure and are thus within the spirit and scope thereof.

WHAT IS CLAIMED IS:

1. A method for video decoding in a decoder, comprising:

decoding, by a processor, prediction information for a block in an I slice from a coded video bitstream;

determining, by the processor, whether an intra block copy (IBC) mode is possible for the block in the I slice based on the prediction information and a size restriction of the IBC mode;

wherein in response to any one of: (i) a slice type parameter indicating the I slice and a size of the block being greater than a threshold, and (ii) a current mode type parameter is set to MODE_TYPE_INTRA which indicates that only intra prediction mode is possible for the block in the I slice, determining that the IBC mode is impossible for the block in the I slice;

in response to determining the IBC mode being impossible for the block in the I slice, determining that a flag indicating whether a skip mode is applied on the block is not signaled in the coded video bitstream, and inferring the flag being 0 which indicates that the skip mode is not applied to the block;

in response to determining the IBC mode being possible for the block in the I slice, decoding, by the processor, a flag that indicates whether a skip mode is applied on the block from the coded video bitstream; and

reconstructing, by the processor, the block at least partially based on the flag.

2. The method of claim 1, further comprising:

determining that the IBC mode is impossible for the block in the I slice in response to at least one of a width of the block and a height of the block being larger than the threshold.

3. The method of claim 1, further comprising:

determining an existence of the flag in the coded video bitstream based on an added condition that compares the size of the block with the threshold, the added condition excluding the existence of the flag in response to the size of the block in the I slice being larger than the threshold.

4. The method of claim 1, further comprising:

determining an existence of the flag in the coded video bitstream based on an existing condition that is modified to compare the size of the block with the threshold.

5. The method of claim 4, wherein the existing condition determines whether the block is an inter coded block.

6. The method of claim 4, wherein the existing condition determines whether the IBC mode is enabled.

7. An apparatus for video decoding, comprising

processing circuitry configured to:

decode, prediction information for a block in an I slice from a coded video bitstream;

determine, whether an intra block copy (IBC) mode is possible for the block in the I slice based on the prediction information and a size restriction of the IBC mode;

wherein in response to any one of: (i) a slice type parameter indicating the I slice and a size of the current block being greater than a threshold, and (ii) a current mode type parameter is set to MODE_TYPE_INTRA which indicates that only intra prediction mode is possible for the block in the I slice, the processing circuitry determines that the IBC mode is impossible for the block in the I slice;

in response to determining the IBC mode being impossible for the block in the I slice, determine that a flag indicating whether a skip mode is applied on the block is not signaled in the coded video bitstream, and infer the flag being 0 which indicates that the skip mode is not applied to the block;

in response to determining the IBC mode being possible for the block in the I slice, decode a flag that indicates whether a skip mode is applied on the block from the coded video bitstream; and

reconstruct the block at least partially based on the flag.

8. The apparatus of claim 7, wherein the processing circuitry is configured to:

determine that the IBC mode is impossible for the block in the I slice in response to at least one of a width of the block and a height of the block being larger than the threshold.

9. The apparatus of claim 7, wherein the processing circuitry is configured to:

determine an existence of the flag in the coded video bitstream based on an added condition that compares the size of the block with the threshold, the added condition excluding the existence of the flag in response to the size of the block in the I slice being larger than the threshold.

10. The apparatus of claim 7, wherein the processing circuitry is configured to:

determine an existence of the flag in the coded video bitstream based on an existing condition that is modified to base on a comparison of the size and the threshold.

11. The apparatus of claim 10, wherein the existing condition determines whether the block is an inter coded block.

12. The apparatus of claim 10, wherein the existing condition determines whether the IBC mode is enabled.

13. A non-transitory computer-readable medium storing instructions which when executed by a computer for video decoding cause the computer to perform:

decoding prediction information for a block in an I slice from a coded video bitstream;

determining, whether an intra block copy (IBC) mode is possible for the block in the I slice based on the prediction information and a size restriction of the IBC mode;

wherein in response to any one of: (i) a slice type parameter indicating the I slice and a size of the current block being greater than a threshold, and (ii) a current mode type parameter is set to MODE_TYPE_INTRA to which indicates that only an intra prediction mode is possible for the block in the I slice, determining that the IBC mode is impossible for the block in the I slice;

in response to determining the IBC mode being impossible for the block in the I slice, determining that a flag indicating whether a skip mode is applied on the block is not signaled in the coded video bitstream, and inferring the flag being 0 which indicates that the skip mode is not applied to the block;

in response to determining the IBC mode being possible for the block in the I slice, decoding a flag that indicates whether a skip mode is applied on the block from the coded video bitstream; and

reconstructing the block at least partially based on the flag.

14. The non-transitory computer-readable medium of claim 13, wherein the instructions cause the computer to perform:

determining that the IBC mode is impossible for the block in the I slice in response to at least one of a width of the block and a height of the block being larger than the threshold.

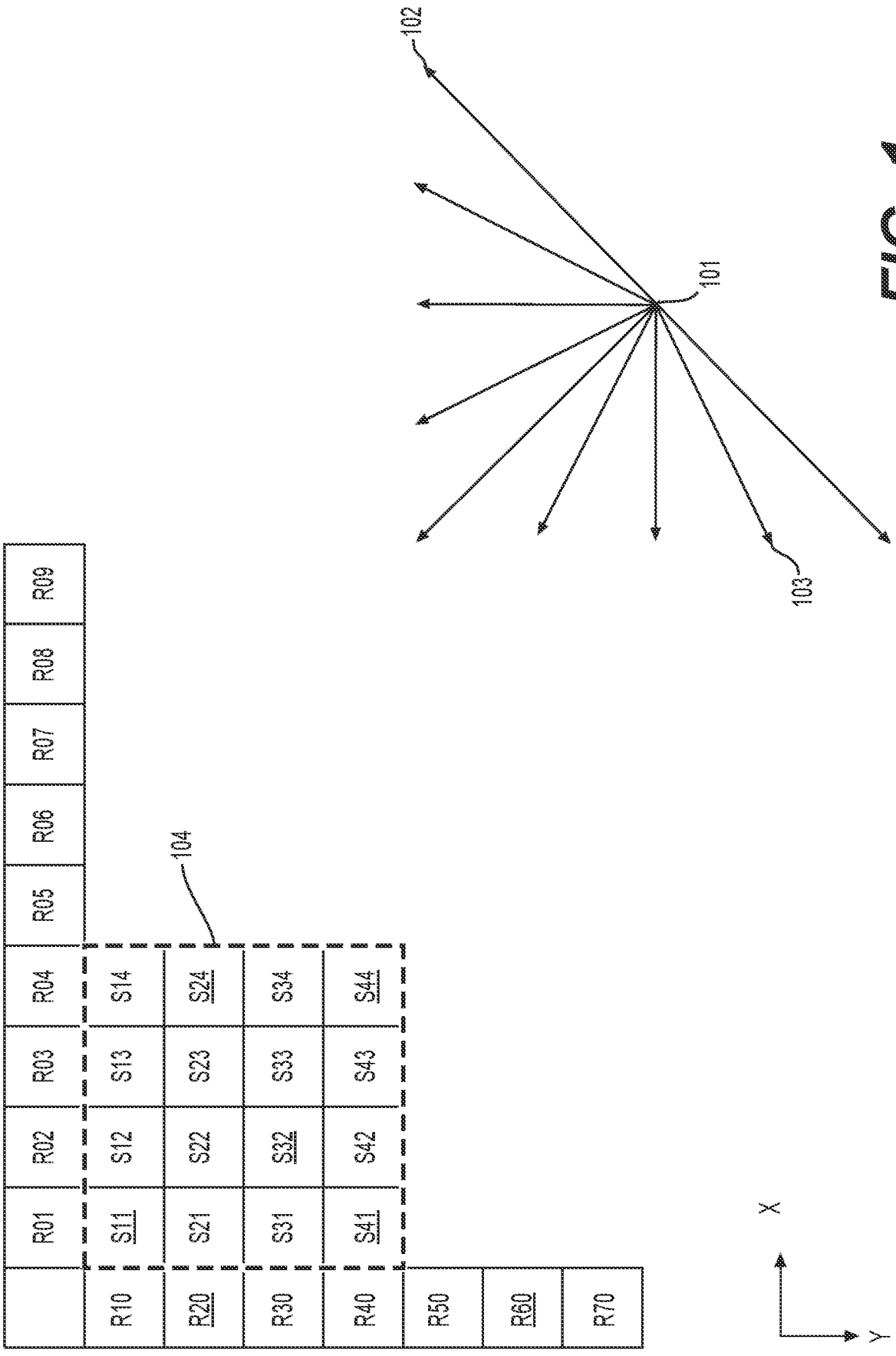
R01
R02
R03
R04
R05
R06
R07
R08
R09
R10
S11
S12
S13
S14
R20
S21
S22
S23
S24
R30
S31
S32
S33
S34
R40
S41
S42
S43
S44
R50
R60
R70
104
102
101
103
X
Y

FIG. 1

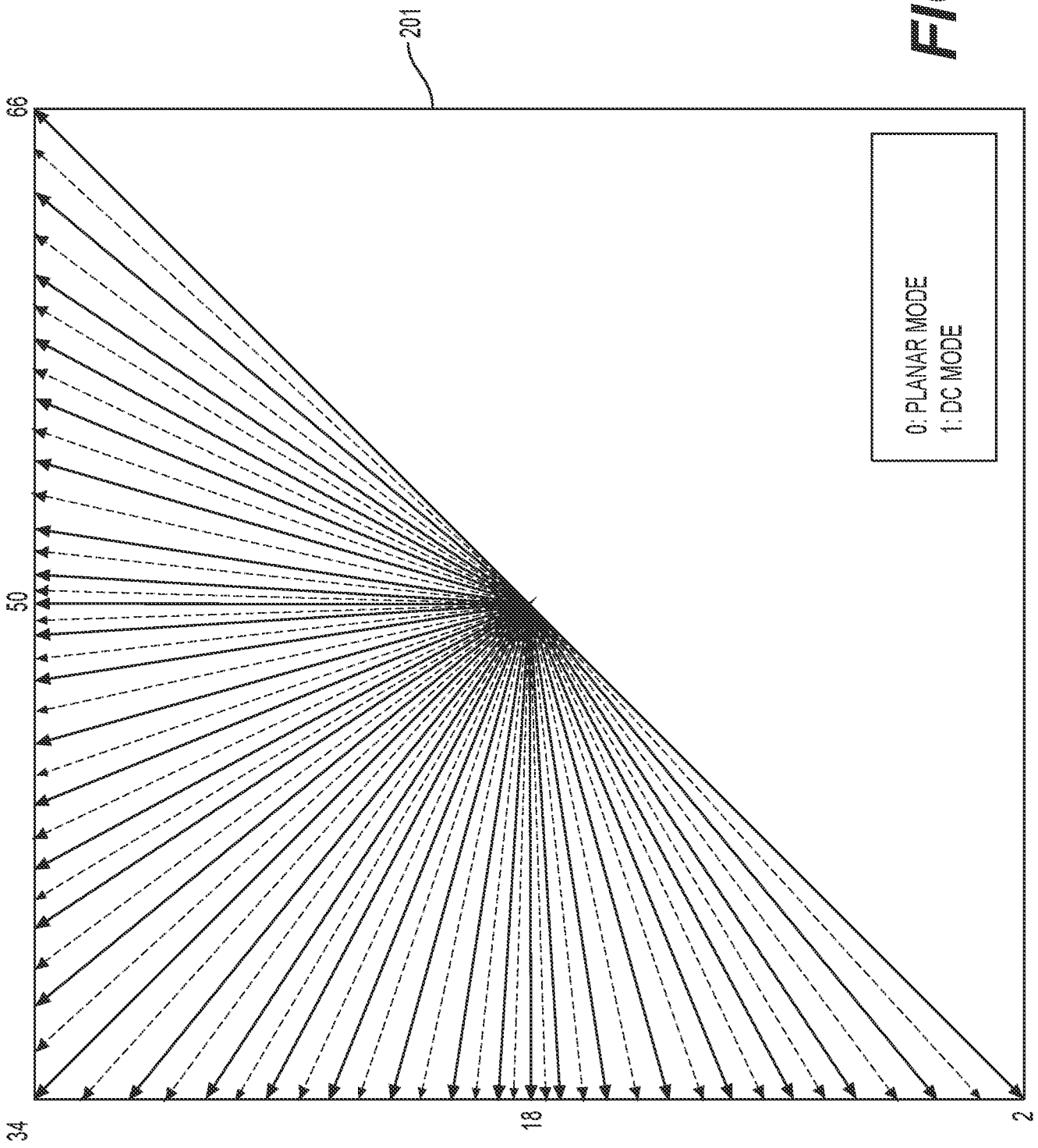

FIG. 2

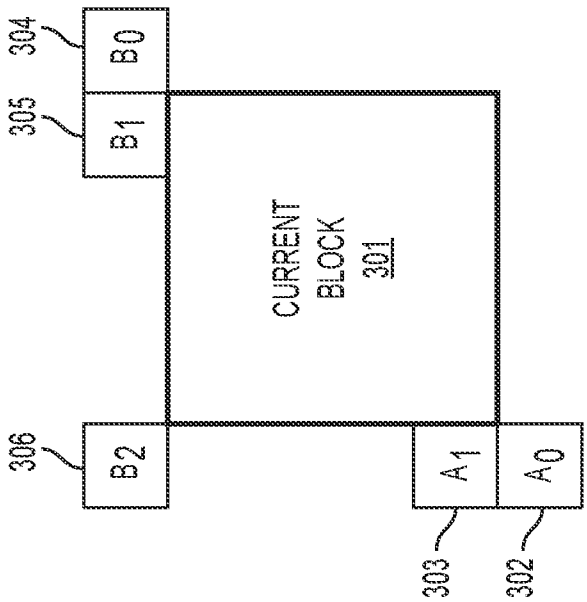

FIG. 3

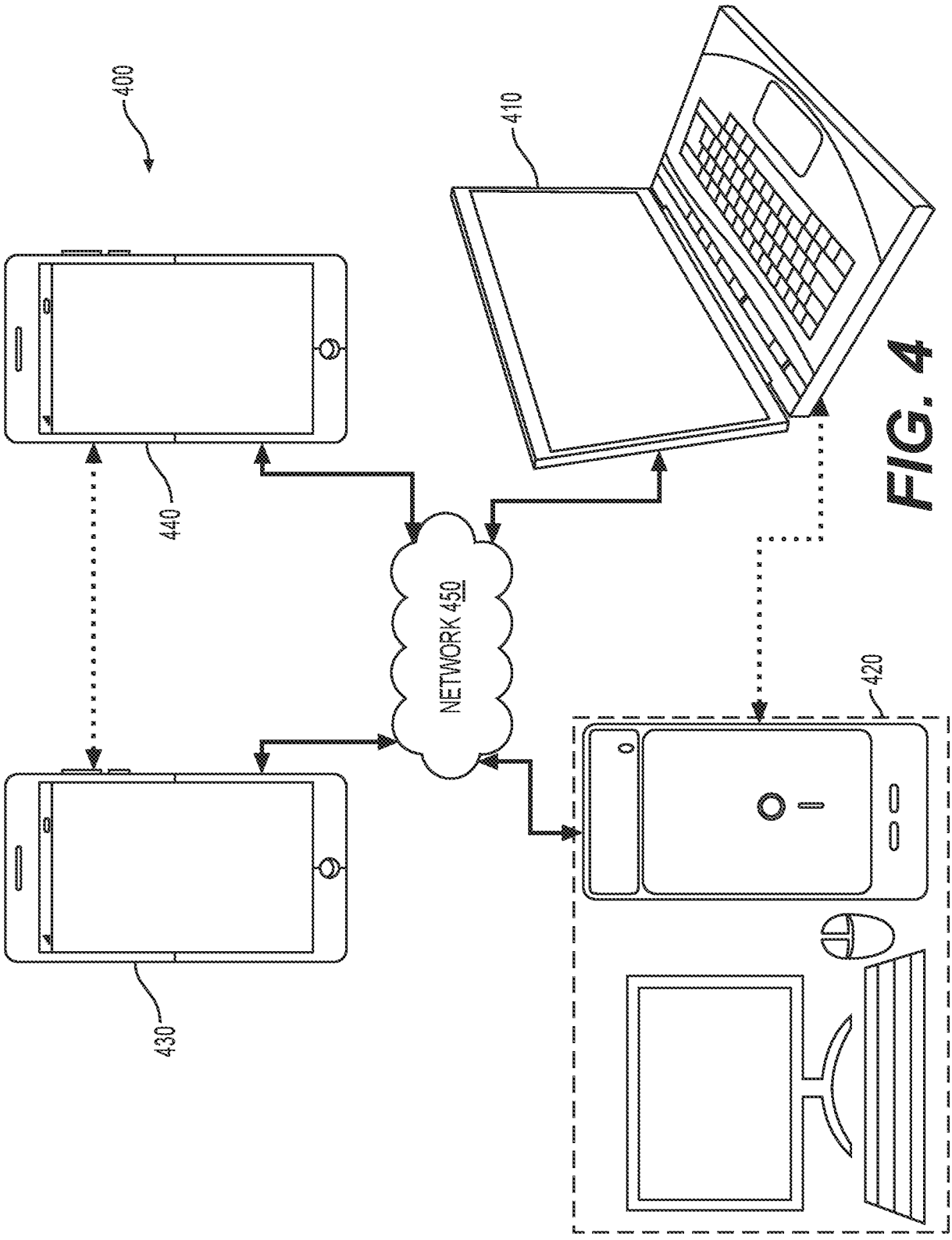

FIG. 4

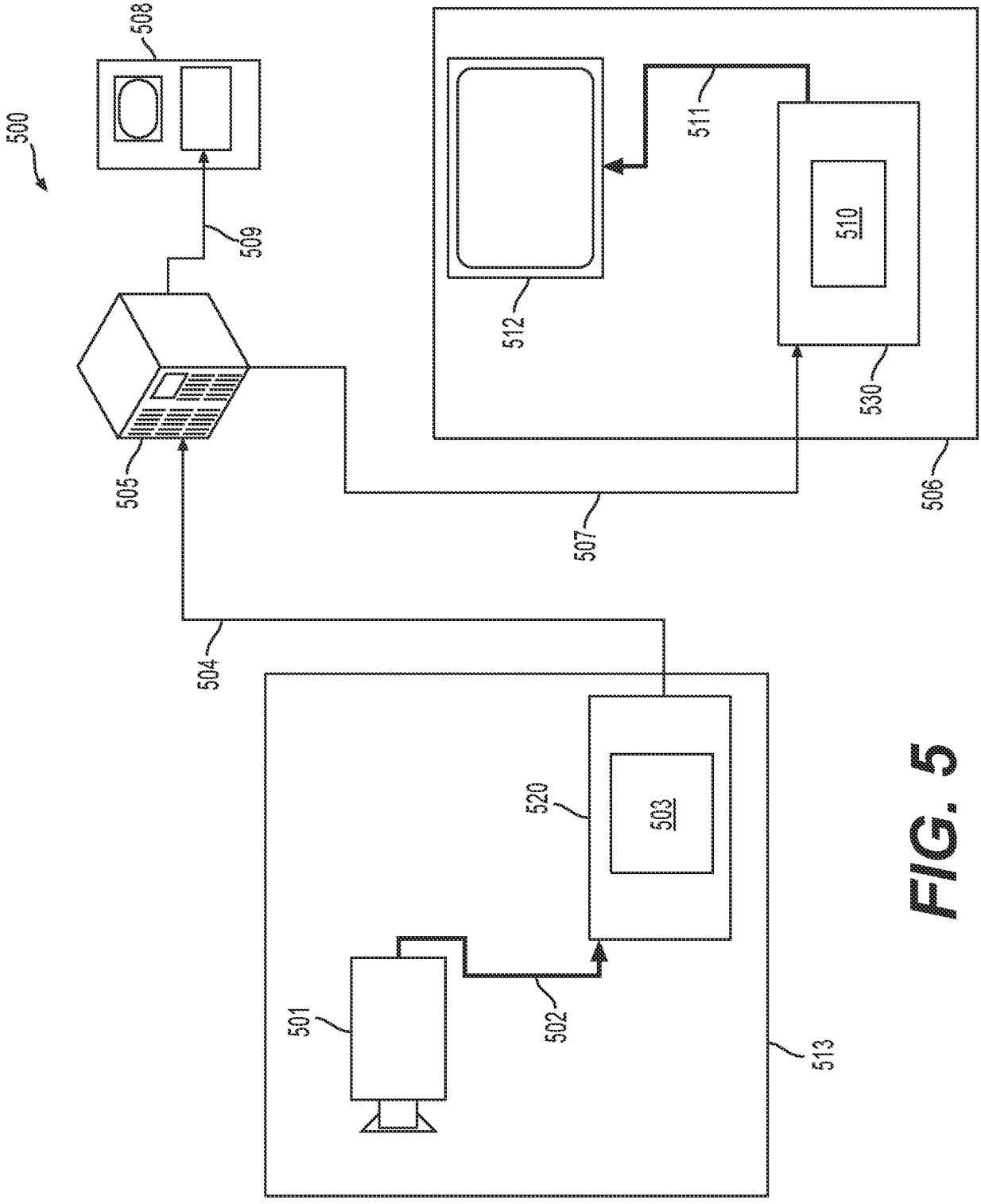

FIG. 5

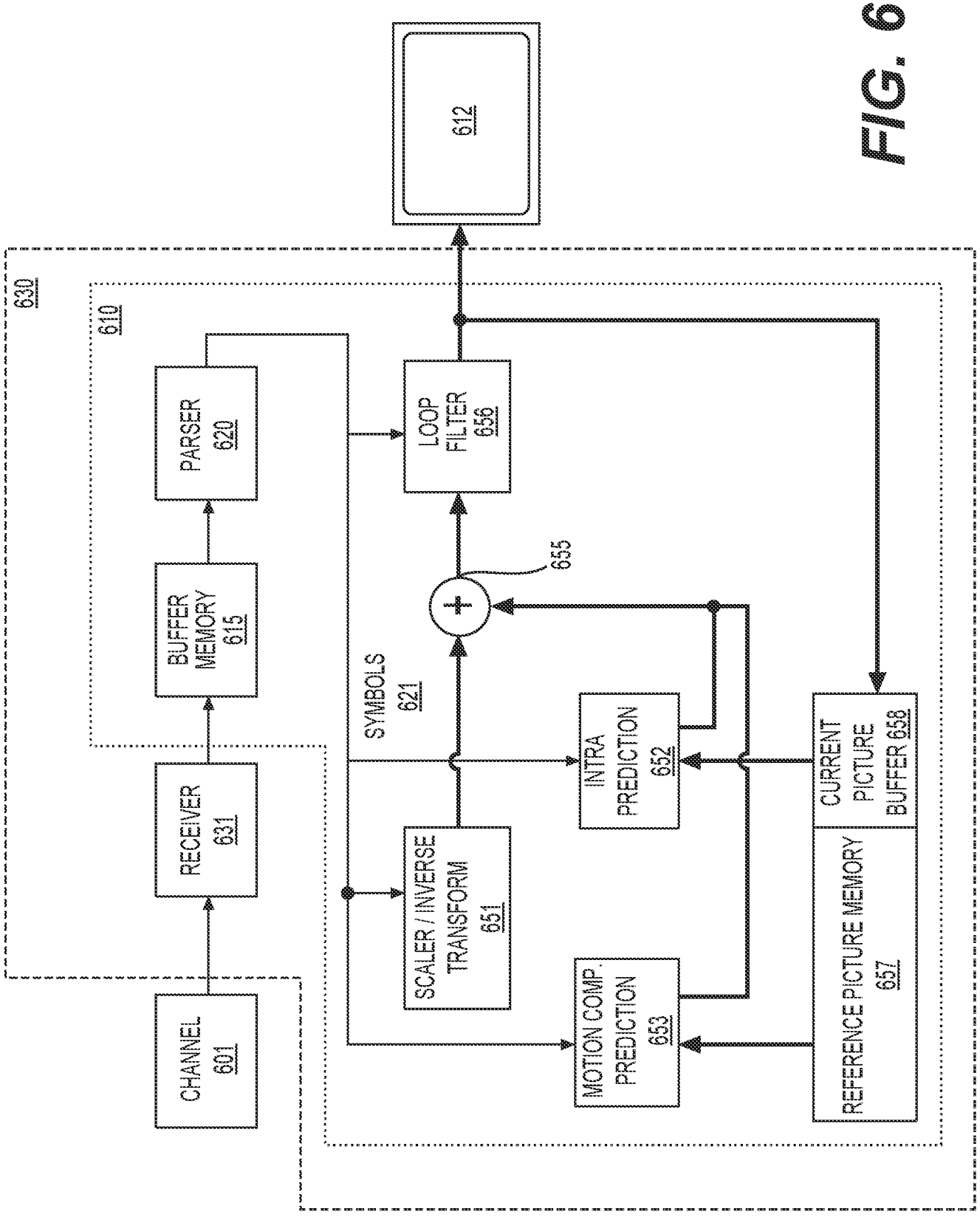
612
630
610
PARSER 620
BUFFER MEMORY 615
RECEIVER 631
CHANNEL 601
LOOP FILTER 656
655
SYMBOLS 621
SCALER / INVERSE TRANSFORM 651
INTRA PREDICTION 652
MOTION COMP. PREDICTION 653
CURRENT PICTURE BUFFER 658
REFERENCE PICTURE MEMORY 657

FIG. 6

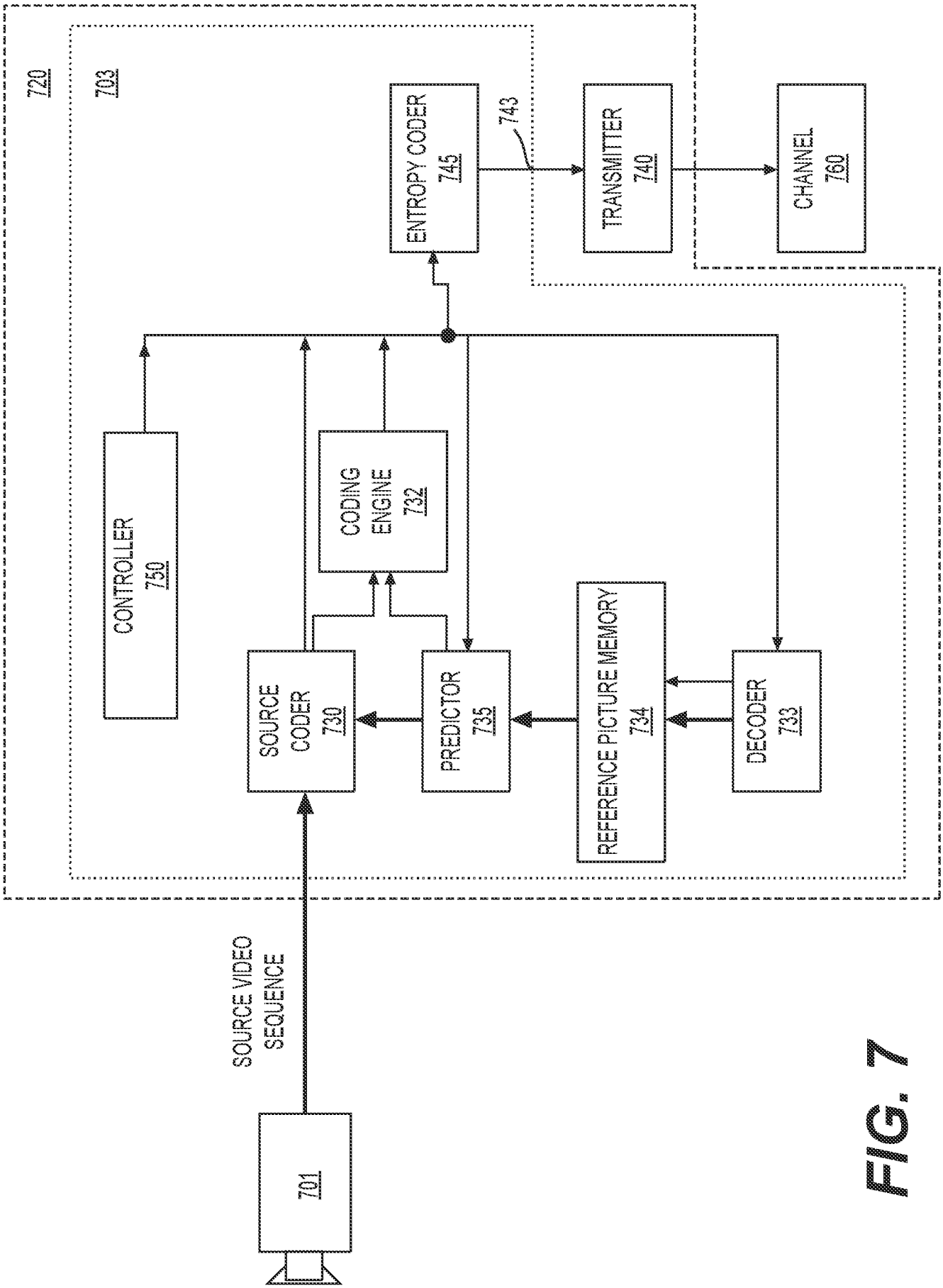
720
703
ENTROPY CODER
745
743
TRANSMITTER
740
CHANNEL
760
CONTROLLER
750
CODING
ENGINE
732
SOURCE
CODER
730
PREDICTOR
735
REFERENCE PICTURE MEMORY
734
DECODER
733
SOURCE VIDEO
SEQUENCE
701

FIG. 7

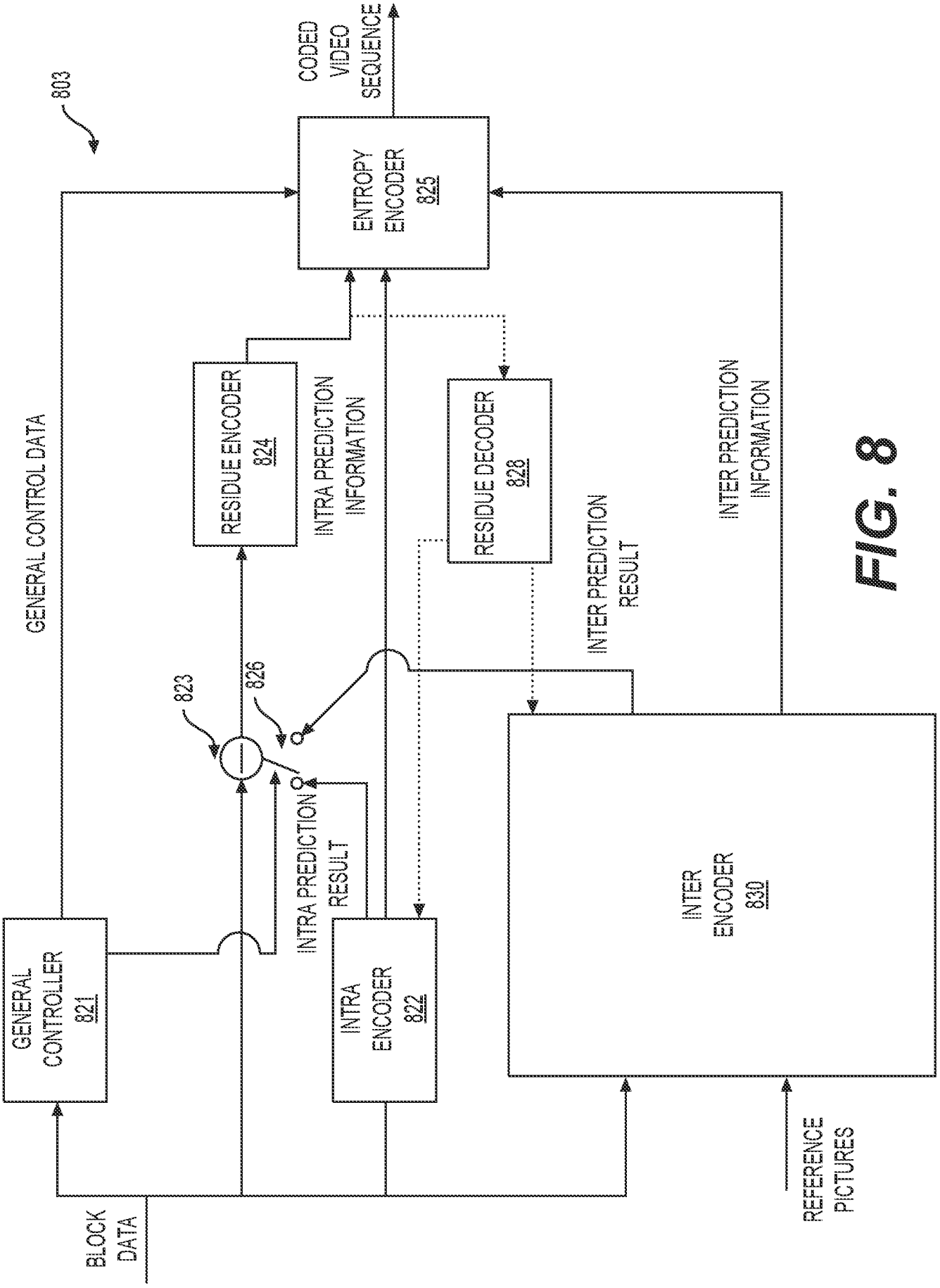
803
CODED VIDEO SEQUENCE
ENTROPY ENCODER 825
GENERAL CONTROL DATA
RESIDUE ENCODER 824
INTRA PREDICTION INFORMATION
RESIDUE DECODER 828
INTER PREDICTION RESULT
INTER PREDICTION INFORMATION
823
826
INTRA PREDICTION RESULT
GENERAL CONTROLLER 821
INTRA ENCODER 822
INTER ENCODER 830
BLOCK DATA
REFERENCE PICTURES

FIG. 8

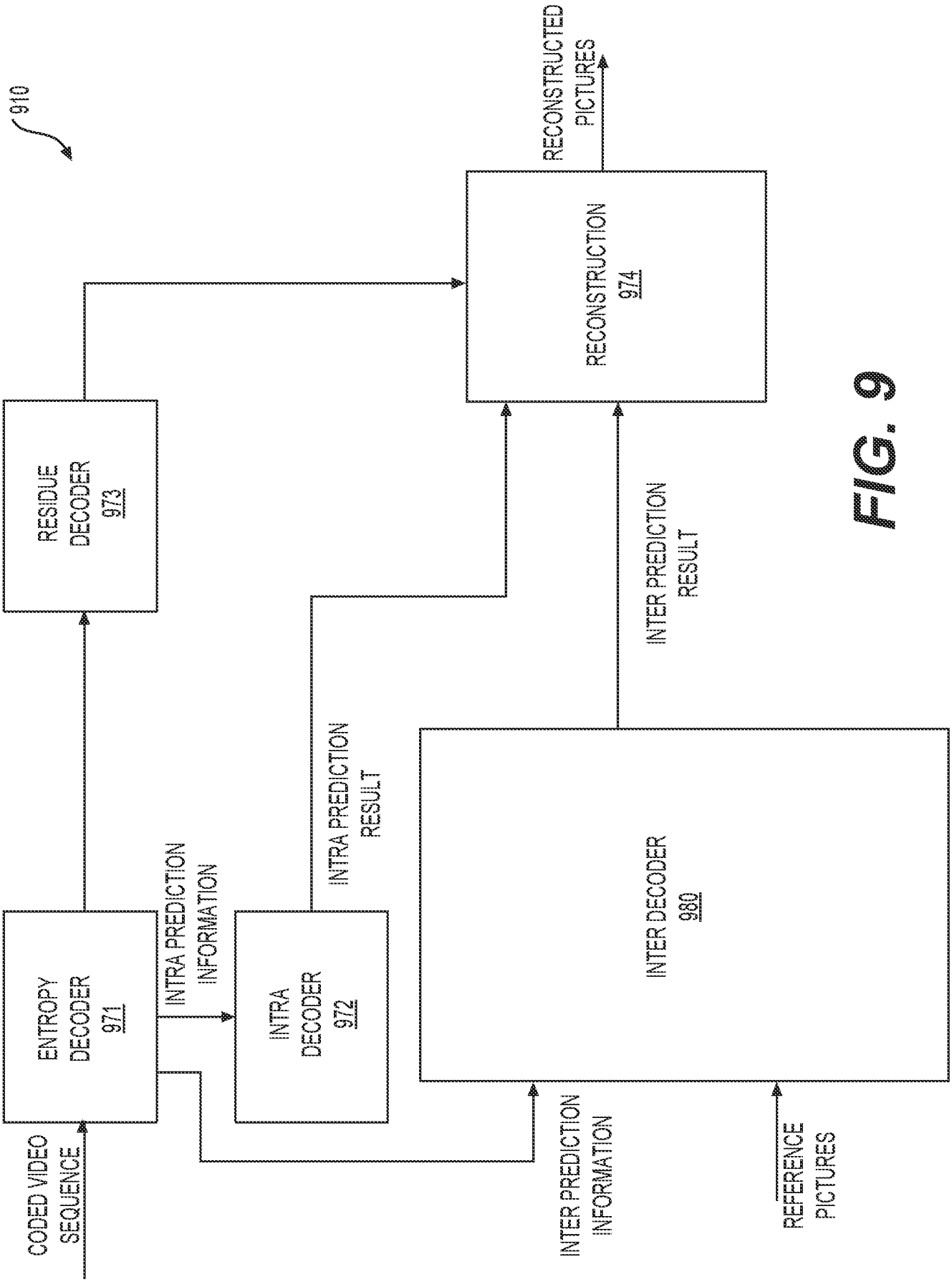
910
CODED VIDEO SEQUENCE
ENTROPY DECODER 971
RESIDUE DECODER 973
INTRA PREDICTION INFORMATION
INTRA DECODER 972
INTRA PREDICTION RESULT
INTER DECODER 980
INTER PREDICTION INFORMATION
REFERENCE PICTURES
INTER PREDICTION RESULT
RECONSTRUCTION 974
RECONSTRUCTED PICTURES

FIG. 9

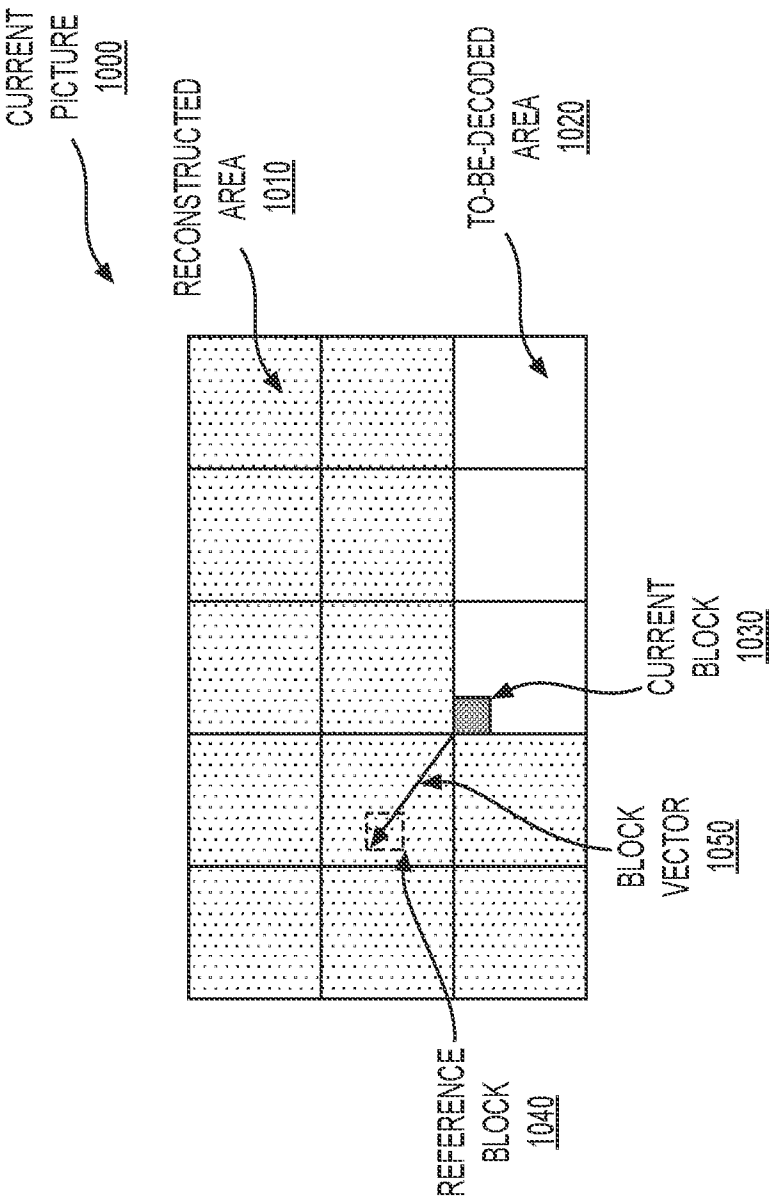
CURRENT PICTURE 1000
RECONSTRUCTED AREA 1010
TO-BE-DECODED AREA 1020
CURRENT BLOCK 1030
BLOCK VECTOR 1050
REFERENCE BLOCK 1040

FIG. 10

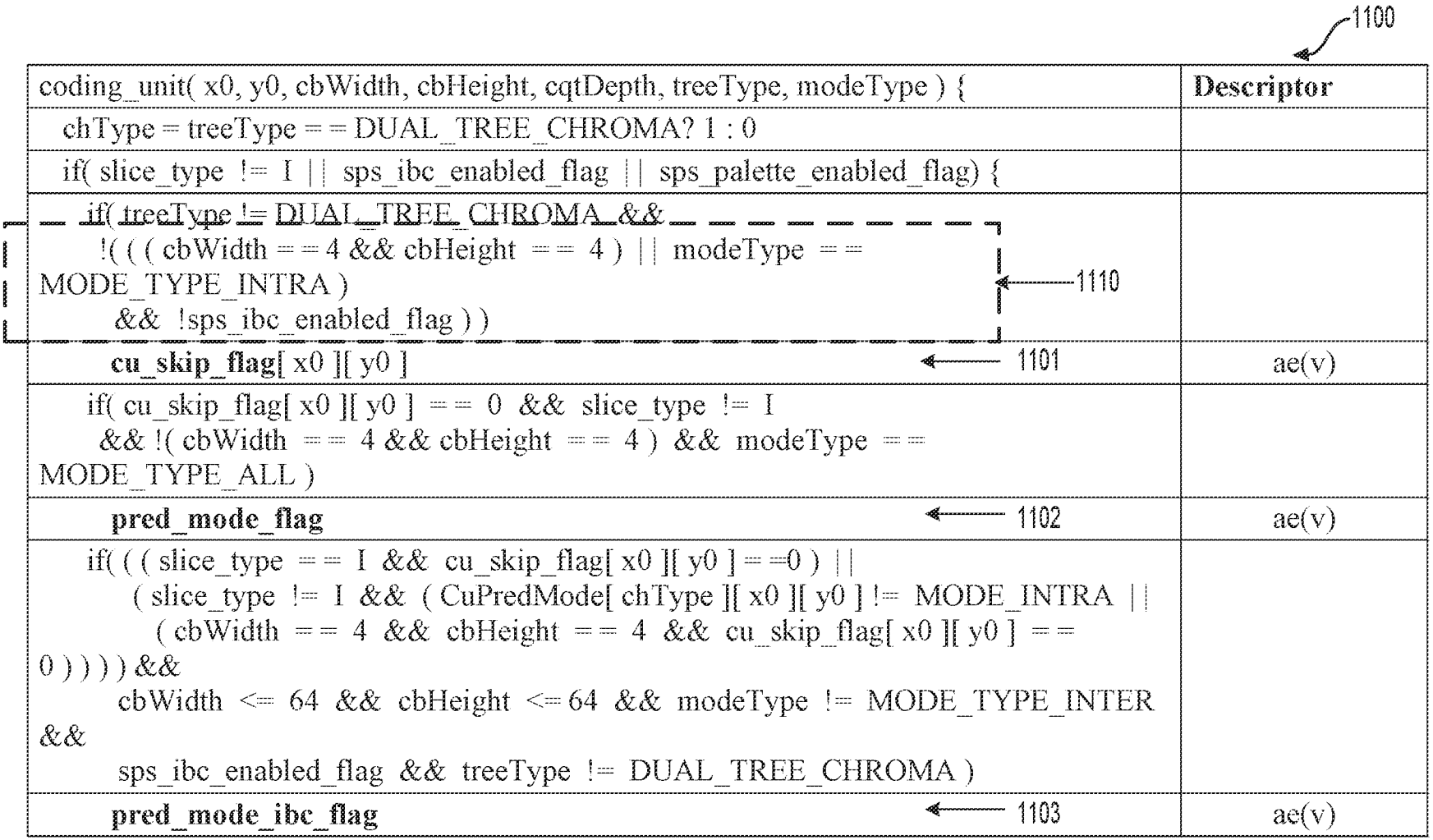
1100
coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType) {
Descriptor
chType = treeType = = DUAL_TREE_CHROMA? 1 : 0
if(slice_type != I || sps_ibc_enabled_flag || sps_palette_enabled_flag) {
if(treeType != DUAL_TREE_CHROMA &&
!(((cbWidth = = 4 && cbHeight = = 4) || modeType = = MODE_TYPE_INTRA)
&& !sps_ibc_enabled_flag))
1110
cu_skip_flag[x0][y0]
1101
ae(v)
if(cu_skip_flag[x0][y0] = = 0 && slice_type != I
&& !(cbWidth = = 4 && cbHeight = = 4) && modeType = = MODE_TYPE_ALL)
pred_mode_flag
1102
ae(v)
if(((slice_type = = I && cu_skip_flag[x0][y0] = =0) ||
(slice_type != I && (CuPredMode[chType][x0][y0] != MODE_INTRA ||
(cbWidth = = 4 && cbHeight = = 4 && cu_skip_flag[x0][y0] = = 0)))) &&
cbWidth <= 64 && cbHeight <= 64 && modeType != MODE_TYPE_INTER &&
sps_ibc_enabled_flag && treeType != DUAL_TREE_CHROMA)
pred_mode_ibc_flag
1103
ae(v)

FIG. 11

1200

coding_tree(x0, y0, cbWidth, cbHeight, qgOnY, qgOnC, cbSubdiv, cqtDepth, mttDepth, depthOffset, partIdx, treeTypeCurr, modeTypeCurr) {	**Descriptor**
if((allowSplitBtVer || allowSplitBtHor || allowSplitTtVer || allowSplitTtHor || allowSplitQT) && (x0 + cbWidth <= pic_width_in_luma_samples) && (y0 + cbHeight <= pic_height_in_luma_samples))	
split_cu_flag	ae(v)
if(cu_qp_delta_enabled_flag && qgOnY && cbSubdiv <= CuQpDeltaSubdiv) {	
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
CuQgTopLeftX = x0	
CuQgTopLeftY = y0	
}	
if(cu_chroma_qp_offset_enabled_flag && qgOnC && cbSubdiv <= CuChromaQpOffsetSubdiv)	
IsCuChromaQpOffsetCoded = 0	
if(split_cu_flag) {	
if((allowSplitBtVer || allowSplitBtHor || allowSplitTtVer || allowSplitTtHor) && allowSplitQT)	
split_qt_flag	ae(v)
if(!split_qt_flag) {	
if((allowSplitBtHor || allowSplitTtHor) && (allowSplitBtVer || allowSplitTtVer))	
mtt_split_cu_vertical_flag	ae(v)
if((allowSplitBtVer && allowSplitTtVer && mtt_split_cu_vertical_flag) || (allowSplitBtHor && allowSplitTtHor && !mtt_split_cu_vertical_flag))	
mtt_split_cu_binary_flag	ae(v)
}	
if(modeTypeCondition == 1)	
modeType = MODE_TYPE_INTRA	

FIG. 12A

Continue from FIG. 12A

1200

else if(modeTypeCondition == 2) {	
mode_constraint_flag	ae(v)
modeType = mode_constraint_flag ? MODE_TYPE_INTRA : MODE_TYPE_INTER	
} else {	
modeType = modeTypeCurr	
}	
treeType = (modeType == MODE_TYPE_INTRA) ? DUAL_TREE_LUMA : treeTypeCurr	
if(!split_qt_flag) {	
if(MttSplitMode[x0][y0][mttDepth] == SPLIT_BT_VER) {	
depthOffset += (x0 + cbWidth > pic_width_in_luma_samples) ? 1 : 0	
x1 = x0 + (cbWidth / 2)	
coding_tree(x0, y0, cbWidth / 2, cbHeight, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 0, treeType, modeType)	
if(x1 < pic_width_in_luma_samples)	
coding_tree(x1, y0, cbWidth / 2, cbHeightY, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 1, treeType, modeType)	

FIG. 12B

Continue from FIG. 12B

1200

} else if(MttSplitMode[x0][y0][mttDepth] == SPLIT_BT_HOR) {	
depthOffset += (y0 + cbHeight > pic_height_in_luma_samples) ? 1 : 0	
y1 = y0 + (cbHeight / 2)	
coding_tree(x0, y0, cbWidth, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 0, treeType, modeType)	
if(y1 < pic_height_in_luma_samples)	
coding_tree(x0, y1, cbWidth, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 1, treeType, modeType)	
} else if(MttSplitMode[x0][y0][mttDepth] == SPLIT_TT_VER) {	
x1 = x0 + (cbWidth / 4)	
x2 = x0 + (3 * cbWidth / 4)	
qgOnY = qgOnY && (cbSubdiv + 2 <= CuQpDeltaSubdiv)	
qgOnC = qgOnC && (cbSubdiv + 2 <= CuChromaQpOffsetSubdiv)	
coding_tree(x0, y0, cbWidth / 4, cbHeight, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth, mttDepth + 1, depthOffset, 0, treeType, modeType)	
coding_tree(x1, y0, cbWidth / 2, cbHeight, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 1, treeType, modeType)	
coding_tree(x2, y0, cbWidth / 4, cbHeight, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth, mttDepth + 1, depthOffset, 2, treeType, modeType)	

FIG. 12C

Continue from FIG. 12C

1200

} else { /* SPLIT_TT_HOR */	
y1 = y0 + (cbHeight / 4)	
y2 = y0 + (3 * cbHeight / 4)	
qgOnY = qgOnY && (cbSubdiv + 2 <= CuQpDeltaSubdiv)	
qgOnC = qgOnC && (cbSubdiv + 2 <= CuChromaQpOffsetSubdiv)	
coding_tree(x0, y0, cbWidth, cbHeight / 4, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth, mttDepth + 1, depthOffset, 0, treeType, modeType)	
coding_tree(x0, y1, cbWidth, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 1, treeType, modeType)	
coding_tree(x0, y2, cbWidth, cbHeight / 4, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth, mttDepth + 1, depthOffset, 2, treeType, modeType)	
}	
} else {	
x1 = x0 + (cbWidth / 2)	
y1 = y0 + (cbHeight / 2)	
coding_tree(x0, y0, cbWidth / 2, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth + 1, 0, 0, 0, treeType, modeType)	
if(x1 < pic_width_in_luma_samples)	
coding_tree(x1, y0, cbWidth / 2, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth + 1, 0, 0, 1, treeType, modeType)	
if(y1 < pic_height_in_luma_samples)	
coding_tree(x0, y1, cbWidth / 2, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth + 1, 0, 0, 2, treeType, modeType)	
if(y1 < pic_height_in_luma_samples && x1 < pic_width_in_luma_samples)	

FIG. 12D

Continue from FIG. 12D

1200

coding_tree(x1, y1, cbWidth / 2, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth + 1, 0, 0, 3, treeType, modeType)	
}	
if(modeTypeCur = = MODE_TYPE_ALL && modeType = = MODE_TYPE_INTRA) {	
coding_tree(x0, y0, cbWidth, cbHeight, 0, qgOnC, cbSubdiv, cqtDepth, mttDepth, 0, 0 DUAL_TREE_CHROMA , modeType)	
}	
} else	
coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeTypeCurr , mode TypeCurr)	
}	

FIG. 12E

1300

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType) {	Descriptor
chType = treeType = = DUAL_TREE_CHROMA? 1 : 0	
if(slice_type != I || sps_ibc_enabled_flag || sps_palette_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA && !((cbWidth = = 4 && cbHeight = = 4) || modeType = = MODE_TYPE_INTRA) && sps_ibc_enabled_flag && !(slice_type = I && (cbWidth > 64 || cbHeight > 64)))	1310
cu_skip_flag[x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] = = 0 && slice_type != I && !(cbWidth = = 4 && cbHeight = = 4) && modeType = = MODE_TYPE_ALL)	
pred_mode_flag	ae(v)
if(((slice_type = = I && cu_skip_flag[x0][y0] = =0) || (slice_type != I && (CuPredMode[chType][x0][y0] != MODE_INTRA || (cbWidth = = 4 && cbHeight = = 4 && cu_skip_flag[x0][y0] = = 0))) && cbWidth <= 64 && cbHeight <= 64 && modeType != MODE_TYPE_INTER && sps_ibc_enabled_flag && treeType != DUAL_TREE_CHROMA)	
pred_mode_ibc_flag	ae(v)

FIG. 13

coding_tree(x0, y0, cbWidth, cbHeight, qgOnY, qgOnC, cbSubdiv, cqtDepth, mttDepth, depthOffset, partIdx, treeTypeCurr, modeTypeCurr) {	**Descriptor**
if((allowSplitBtVer || allowSplitBtHor || allowSplitTtVer || allowSplitTtHor || allowSplitQT) && (x0 + cbWidth <= pic_width_in_luma_samples) && (y0 + cbHeight <= pic_height_in_luma_samples))	
split_cu_flag	ae(v)
if(cu_qp_delta_enabled_flag && qgOnY && cbSubdiv <= CuQpDeltaSubdiv) {	
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
CuQgTopLeftX = x0	
CuQgTopLeftY = y0	
}	
if(cu_chroma_qp_offset_enabled_flag && qgOnC && cbSubdiv <= CuChromaQpOffsetSubdiv)	
IsCuChromaQpOffsetCoded = 0	
if(split_cu_flag) {	
...	
} else {	
if(slice_type == I && (cbWidth > 64 || cbHeight > 64))	
modeTypeCurr = MODE_TYPE_INTRA	
coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeTypeCurr, modeTypeCurr)	
}	
}	

1400

1410

FIG. 14

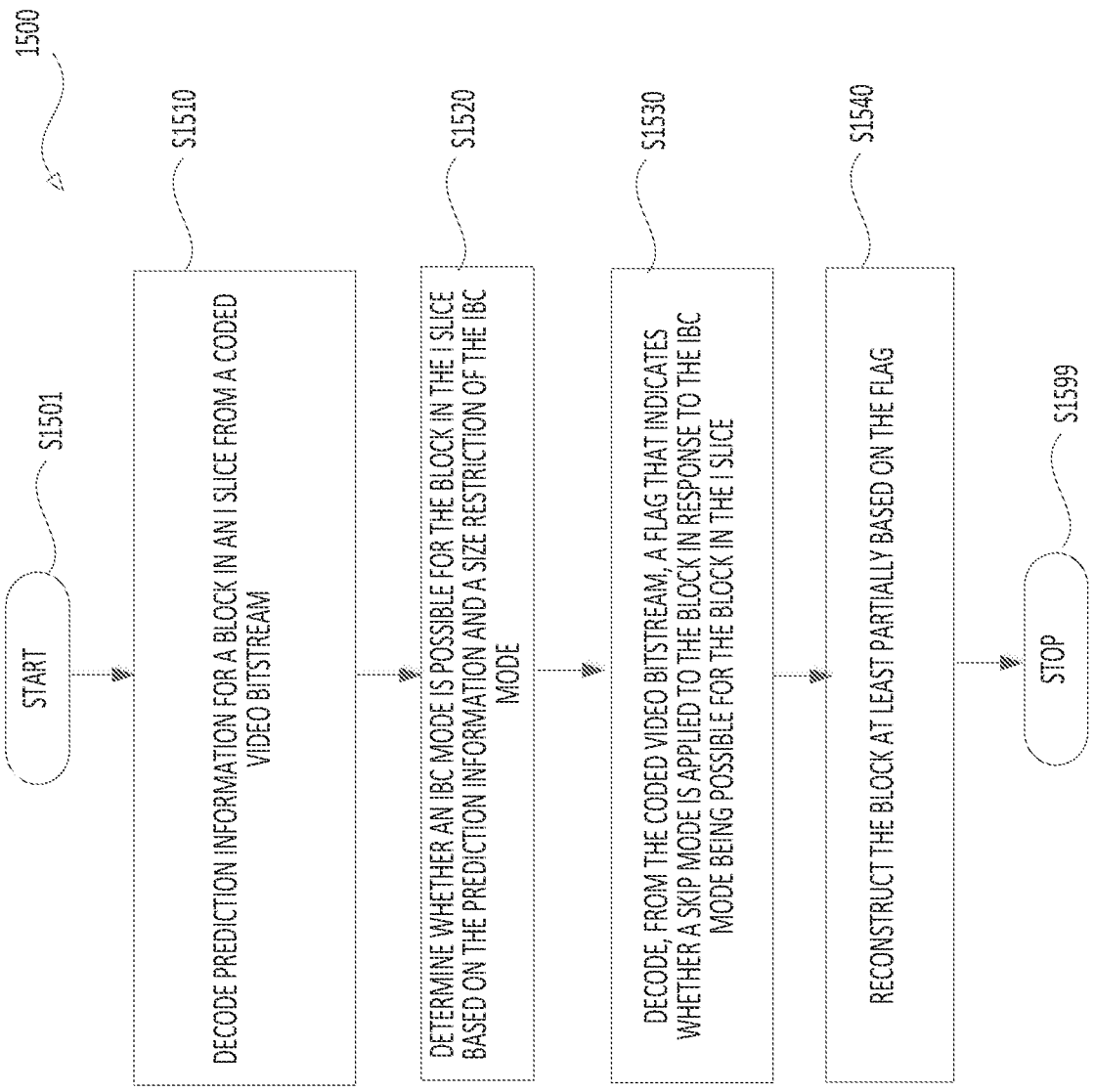
1500
START
S1501
DECODE PREDICTION INFORMATION FOR A BLOCK IN AN I SLICE FROM A CODED VIDEO BITSTREAM
S1510
DETERMINE WHETHER AN IBC MODE IS POSSIBLE FOR THE BLOCK IN THE I SLICE BASED ON THE PREDICTION INFORMATION AND A SIZE RESTRICTION OF THE IBC MODE
S1520
DECODE, FROM THE CODED VIDEO BITSTREAM, A FLAG THAT INDICATES WHETHER A SKIP MODE IS APPLIED TO THE BLOCK IN RESPONSE TO THE IBC MODE BEING POSSIBLE FOR THE BLOCK IN THE I SLICE
S1530
RECONSTRUCT THE BLOCK AT LEAST PARTIALLY BASED ON THE FLAG
S1540
STOP
S1599

FIG. 15

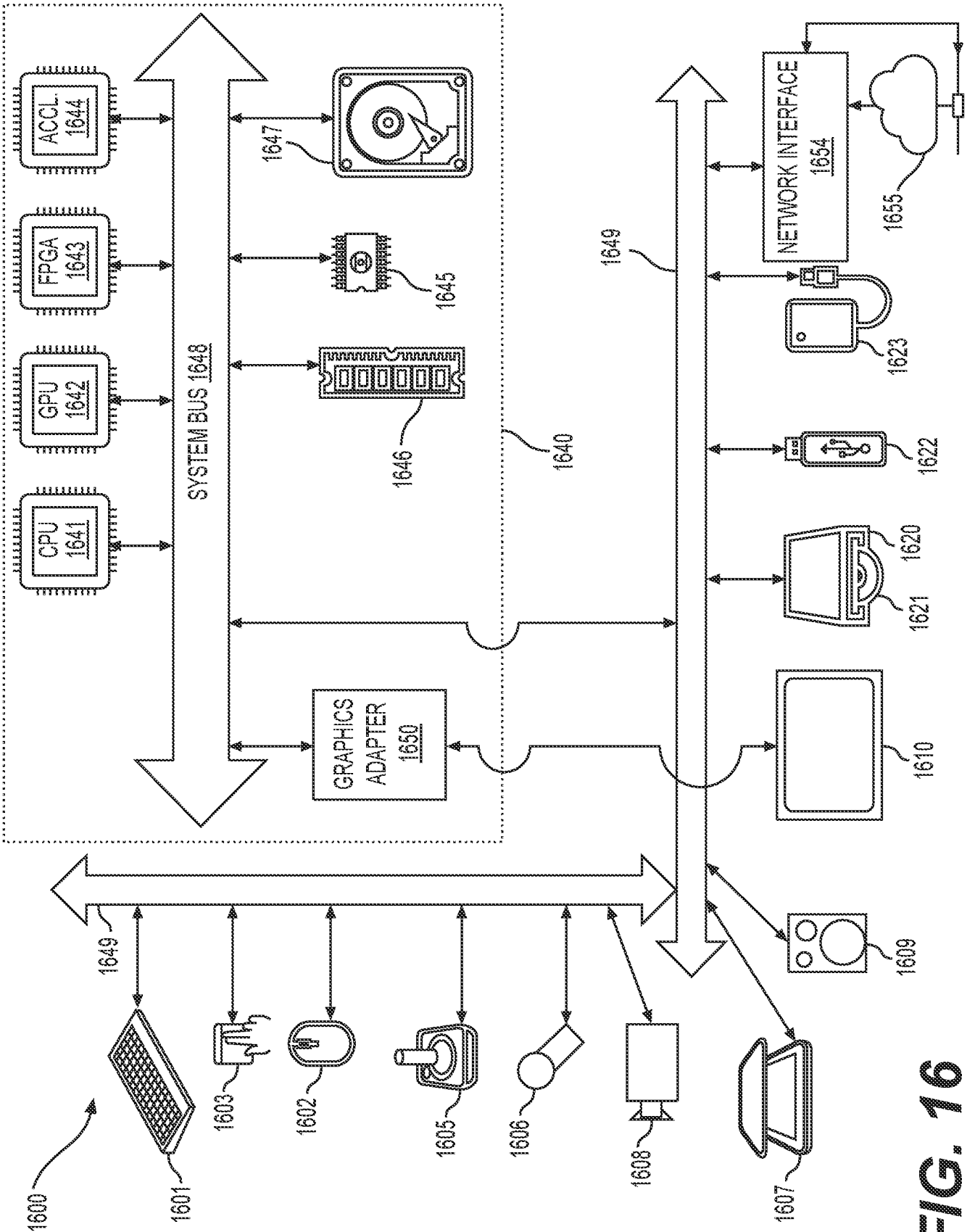
ACCL 1644
FPGA 1643
GPU 1642
CPU 1641
SYSTEM BUS 1648
1647
1645
1646
1640
GRAPHICS ADAPTER 1650
1649
NETWORK INTERFACE 1654
1655
1623
1622
1620
1621
1610
1609
1600
1601
1603
1602
1605
1606
1608
1607

FIG. 16