

【公報種別】特許法第 17 条の 2 の規定による補正の掲載

【部門区分】第 7 部門第 3 区分

【発行日】平成30年4月19日(2018.4.19)

【公表番号】特表2017-515339(P2017-515339A)

【公表日】平成29年6月8日(2017.6.8)

【年通号数】公開・登録公報2017-021

【出願番号】特願2016-557937(P2016-557937)

【国際特許分類】

H 0 4 N 19/70 (2014.01)

H 0 4 N 19/90 (2014.01)

【F I】

H 0 4 N 19/70

H 0 4 N 19/90

【手続補正書】

【提出日】平成30年3月9日(2018.3.9)

【手続補正 1】

【補正対象書類名】明細書

【補正対象項目名】全文

【補正方法】変更

【補正の内容】

【発明の詳細な説明】

【発明の名称】無損失ビデオコーディングのシグナリングのための方法および装置

【技術分野】

【0001】

本発明は、無損失ビデオコーディングのシグナリングのための方法および装置に関する。

【背景技術】

【0002】

関連出願の相互参照

本出願は、2014年3月16日に出願された米国特許仮出願第61/953,922号明細書および2015年1月15日に出願された米国特許仮出願第62/103,916号明細書の非仮出願であり、米国特許法第119条(e)下でのこれらの米国特許仮出願からの利益を主張する。これらの特許出願の内容は、参照によりそれらの全体が本明細書に援用される。

【0003】

過去20年にわたって、効率的なデジタルビデオ通信、配信および消費を可能にする様々なデジタルビデオ圧縮技術が開発され、標準化されてきた。H.261、MPEG-1、MPEG-2、H.263、MPEG-4(part-2)およびH.264/AVC(MPEG-4 part 10 Advance Video Coding)等、商業的に広く展開されている規格のほとんどは、ISO/IECおよびITU-Tによって開発されている。新たな高度なビデオ圧縮技術の登場および成熟に起因して、ITU-Tビデオコーディングエキスパートグループ(VEG)およびISO/IEC MPEGによる共同開発の下で、新たなビデオコーディング規格、高効率ビデオコーディング(HEVC)である、HEVC(ITU-T H.265/ISO/IEC 23008-2)は2013年前半に国際規格として承認され、現行の最新技術であるH.264/AVCよりも実質的に高いコーディング効率を達成することができる。

【0004】

従来のデジタルビデオサービス(衛星、ケーブルおよび陸上伝送チャネルを介したTV

信号の送信等)と比較して、IPTV、ビデオチャット、モバイルビデオおよびストリーミングビデオ等のますます多くの新たなビデオアプリケーションが異種環境において展開されている。そのような異種性は、ネットワークのみでなくクライアントにも存在する。クライアント側において、Nスクリーンのシナリオ、すなわち、スマートフォン、タブレット、PCおよびTVを含む様々なスクリーンサイズおよびディスプレイ機能を有するデバイス上でビデオコンテンツを消費することが、既に市場を支配しており、また支配し続けることが予想されている。ネットワーク側において、ビデオは、インターネット、Wi-Fiネットワーク、モバイル(3Gおよび4G)ネットワーク、および/またはそれらの任意の組み合わせにわたって伝送されている。

【先行技術文献】

【非特許文献】

【0005】

【非特許文献1】R. Joshi, J. Xu, HEVC Screen Content Coding Draft Text 2. Document No. JCTVC-S1005, Oct. 2014 (Joshi 2014)

【非特許文献2】D. Flynn, M. Naccari, C. Rosewarne, J. Sole, G. Sullivan, T. Suzuki, "High Efficiency Video Coding (HEVC) Range Extensions text specification: Draft 6," Document No. JCTVC-P1005, Jan 2014

【非特許文献3】B. Bross, W.-J. Han, G. J. Sullivan, J.-R. Ohm, Y. K. Wang, T. Wiegand. High Efficiency Video Coding (HEVC) text specification draft 10. Document No. JCTVC-L1003. Jan. 2013

【発明の概要】

【0006】

無損失符号化の高水準シグナリングに係るシステムおよび方法が本明細書に記載される。いくつかの実施形態では、本方法は、無損失コーディングが用いられることを示す高水準シンタックス要素を含むビデオデータビットストリームを生成することを含む。高水準シグナリングシンタックスは、ピクチャパラメータセット(PPS)、シーケンスパラメータセット(PPS)、ビデオパラメータセット(VPS)またはスライスセグメントヘッダのうちの1つであり得る。無損失コーディングシンタックス要素は、量子化、変形、変形スキップ、変形スキップ回転、およびループ内フィルタリングプロセスに係る付けられた1または複数のPPSシンタックス要素を生成するための条件として用いられ得る。

【0007】

いくつかの実施形態では、方法は、transquant_bypass_default_flagを含むピクチャパラメータセット(PPS)を生成することを含む。PPSを参照するスライス内の全てのコーディングユニットのcu_transquant_bypass_flagのデフォルト値を示すtransquant_bypass_default_flagは1に設定される。PPSは、0に等しいtransquant_bypass_enabled_flagも有することができる。

【0008】

デコーダ側において、無損失コーディングが適用されるとき、逆量子化、逆変換、デブロッキングフィルタ、サンプル適応オフセット(SAO)等を含む、複数の処理ブロックがバイパスされ得る。したがって、デコーダが高水準無損失コーディングインジェクションを受信する場合、デコーダは、コーディングユニット(CU)の大きな群がこれらの処理ブロックを必要としないと判断する。これらの処理ブロックを、復号の前にシャットダウンすることは、電力低減、処理サイクル低減、より良好な負荷プロビジョニング等の観

点で有利であり得る。

【0009】

このため、いくつかの実施形態では、方法は、ビデオデコーダにおいて、高水準無損失コーディングインジェクションを受信し、それに応じて複数の処理ブロックをシャットダウンすることを含む。複数の処理ブロックは、以下のハードウェアブロック、すなわち、逆量子化、逆変換、デブロックフィルタおよび/またはSAOのうちの任意のもののうちの1つまたは複数を含むことができる。更に、処理ブロックは、ビデオ復号の前にシャットダウンされ、処理ブロックハードウェアコンポーネントの少なくとも一部分の電力消費の低減を引き起こすことが可能である。

【図面の簡単な説明】

【0010】

更なる詳細な理解は、例として添付の図面と併せて提示される以下の説明から得られることが可能である。

【図1A】ブロックベースのビデオエンコーダの例を示すブロック図である。

【図1B】ブロックベースの無損失ビデオエンコーダの例を示すブロック図である。

【図2A】ブロックベースのビデオデコーダの例を示すブロック図である。

【図2B】ブロックベースの無損失ビデオデコーダの例を示すブロック図である。

【図3】8個の方向予測モードの例の図である。

【図4】33個の方向予測モードおよび2つの非方向予測モードの例を示す図である。

【図5】水平予測の例の図である。

【図6】平面モードの例の図である。

【図7】動き予測の例を示す図である。

【図8】ピクチャ内のブロックレベルの動きの例を示す図である。

【図9】例示的な通信システムを示す図である。

【図10】例示的な無線送信/受信ユニット(WTRU)を示す図である。

【図11】コーディングされたビットストリーム構造の例を示す図である。

【図12】フラグ`transquant_bypass_default_flag`を用いて変更されたPPS拡張シンタックスを復号する方法のフローチャートである。

【発明を実施するための形態】

【0011】

ここで、様々な図面を参照して、例示的な実施形態の詳細な説明が提供される。この説明は、可能な実施態様の詳細な例を提供するが、提供される詳細は、例示であることが意図され、本出願の範囲をいかなる形においても限定することは意図されていないことに留意されるべきである。

【0012】

ビデオの符号化および復号

図1Aは、ブロックベースのビデオエンコーダ、例えば、ハイブリッドビデオ符号化システムの例を示すブロック図である。ビデオエンコーダ100は、入力ビデオ信号102を受信することができる。入力ビデオ信号102は、ブロックごとに処理され得る。ビデオブロックは任意のサイズであり得る。例えば、ビデオブロックユニットは、16×16ピクセルを含むことができる。16×16ピクセルのビデオブロックユニットは、マクロブロック(MB)と呼ばれる場合がある。高効率ビデオコーディング(HEVC)では、拡張されたブロックサイズ(例えば、これらはコーディングツリーユニット(CTU)またはコーディングユニット(CU)と呼ばれる場合があり、本発明ではこれらの2つの用語は同義である)が、高分解能(例えば、1080p以上)のビデオ信号を効率的に圧縮するために用いられ得る。HEVCにおいて、CUは最大64×64ピクセルとすることができる。CUは、予測ユニット(PU)に区分することができ、これらのユニットに対し、別個の予測方法を適用することができる。

【0013】

入力ビデオブロック(例えば、MBまたはCU)について、空間予測160および/ま

たは時間予測 162 を行うことができる。空間予測（例えば、「イントラ予測」）は、同じビデオピクチャ／スライス内の既にコーディングされた近隣ブロックからのピクセルブロックを用いて現在のビデオブロックを予測することができる。空間予測は、ビデオ信号内に固有の空間冗長性を低減することができる。時間予測（例えば、「インター予測」または「動き補償予測」）は、既にコーディングされたビデオピクチャ（例えば、「参照ピクチャ」と呼ばれる場合がある）からのピクセルを用いて、現在のビデオブロックを予測することができる。時間予測は、ビデオ信号内に固有の時間冗長性を低減することができる。ビデオブロックのための時間予測信号は、1 または複数の動きベクトルによってシグナリングされることが可能であり、これは、現在のブロックと、参照ピクチャにおけるその予測ブロックとの間の動きの量および／または方向を示すことができる。（例えば、H.264 / AVC および／または H.265 の場合のように）複数の参照ピクチャがサポートされる場合、ビデオブロックについて、その参照ピクチャインデックスが送信され得る。参照ピクチャインデックスは、参照ピクチャストア 164 内のいずれの参照ピクチャから時間予測信号が到来するかを特定するのに用いられ得る。

【0014】

エンコーダにおけるモード決定ブロック 180 は、例えば、空間および／または時間予測の後に予測モードを選択することができる。予測ブロックは、116 において、現在のビデオブロックから減算され得る。予測残差は、変換 104 および／または量子化 106 され得る。量子化ブロック 106 は、予測残差をコーディングするのに必要なビット数を効果的に低減することができる。量子化パラメータ（QP）が、量子化の厳密度（severity）を制御するのに用いられ得る。QP 値が増大すると、より厳密な量子化が適用されることが可能であるが、結果として、コーディングされるビデオビットレートが低減される場合があり、同時に、復号化されるビデオ品質が劣化する場合がある。量子化に起因する一般的に知られている視覚的アーチファクトは、ブロックングアーチファクト、ブラー、不鮮明化、リングング、フリッカ等を含む。図 1A および図 2A に示されるビデオコーディングシステム内の他の処理ブロックも、特に、これらの処理ブロックが、処理パイプラインにおける中間データのビット深度に対し上限を必要とする固定点操作を適用するとき、情報損失を引き起こす場合がある。例えば、変換ブロック 104 において、水平方向の変換が最初に適用されることが可能であり、その後、垂直方向の変換が適用され得る。変換はデータビット深度を増大させるため（乗算に起因する）、水平変換の後、垂直変換のために入力データビット深度を低減させるために、水平変換の出力に右シフトが適用され得る。そのようなシフト操作は、（データビットの深度を低減することによって）実装コストを低減するのに役立つことができるが、処理パイプラインにおける情報損失も引き起こす場合がある。更に、固定点操作を可能にするために、H.264 / AVC および H.265 等の最新のビデオ標準規格における変換は、積分値変換である。これらの積分値変換のうちのいくつかは、ほぼ直交であるが完全に直交でない場合がある。変換（および逆変換）行列が完全に直交でない場合、これらは完全な再構成を補償することができない。換言すれば、量子化を一切行わない場合であっても、非直交変換および逆変換が入力データブロックに適用された後、出力データブロック（スケーリング係数が出力に適用され得る）は、入力データブロックと数学的に同一のままでない場合がある。

【0015】

量子化された残差係数は、逆量子化 110 および／または逆変換 112 され、再構成された残差を形成することができる。この再構成された残差は予測ブロック 126 に戻して加算され、再構成されたビデオブロックを形成することができる。

【0016】

ループ内フィルタリング（例えば、デブロックフィルタ、サンプル適応オフセット、適応ループフィルタ等）が再構成されたビデオブロックに適用される（166）ことが可能であり、その後、これは参照ピクチャストア 164 に入れられ、かつ／または未来のビデオブロックをコーディングするのに用いられる。ビデオエンコーダ 100 は、出力ビデオストリーム 120 を出力することができる。出力ビデオビットストリーム 120 を形

成するために、コーディングモード（例えば、インター予測モードまたはイントラ予測モード）、予測モード情報、動き情報および／または量子化された残差係数がエントロピーコーディングユニット 108 に送信され、圧縮および／またはパッキングされ、ビットストリームを形成することができる。参照ピクチャストア 164 は、復号されたピクチャバッファ（DPB）と呼ばれる場合がある。

【0017】

図 2A は、ブロックベースのビデオデコーダの例を示すブロック図である。ビデオデコーダ 200 は、ビデオビットストリーム 202 を受信することができる。ビデオビットストリーム 202 は、エントロピー復号ユニット 208 においてアンパッキングおよび／またはエントロピー復号され得る。ビデオビットストリームを符号化するのに用いられるコーディングモードおよび／または予測情報は、空間予測ユニット 260（例えば、イントラコーディングされている場合）および／または時間予測ユニット 262（例えば、インターコーディングされている場合）に送信され、予測ブロックを形成することができる。インターコーディングされている場合、予測情報は、予測ブロックサイズ、1 つもしくは複数の動きベクトル（例えば、動きの方向および量を示すことができる）、および／または 1 つもしくは複数の基準インデックス（例えば、これはいずれの参照ピクチャから予測信号が得られるかを示すことができる）を含むことができる。

【0018】

動き補償予測が時間予測ユニット 262 によって適用され、時間予測ブロックを形成することができる。残差変換係数は、逆量子化ユニット 210 および逆変換ユニット 212 に送信され、残差ブロックを再構成することができる。予測ブロックおよび残差ブロックは、226 において共に加算され得る。再構成されたブロックは、ループフィルタ 266 によるループ内フィルタリングを受け、その後、参照ピクチャストア 264 内に記憶される。参照ピクチャストア 264 内の再構成されたビデオは、表示デバイスを駆動するために用いられることが可能であり、かつ／または未来のビデオブロックを予測するのに用いられることが可能である。ビデオデコーダ 200 は、再構成されたビデオ信号 220 を出力することができる。参照ピクチャストア 264 は、復号されたピクチャバッファ（DPB）と呼ばれることもできる。

【0019】

ビデオエンコーダおよび／またはデコーダ（例えば、ビデオエンコーダ 100 またはビデオデコーダ 200）は、空間予測を行うことができる（例えば、これはイントラ予測と呼ばれることが可能である）。空間予測は、複数の予測方向のうちの 1 つに従って、既にコーディングされた近傍ピクセルから予測することによって実行され得る（例えば、これは方向イントラ予測と呼ばれる場合がある）。

【0020】

図 3 は、8 つの方向予測モードの例の図である。図 3 の 8 つの方向予測モードは、H.264 / AVC においてサポートされ得る。9 個のモード（DC モード 2 を含む）は、以下である。

- ・モード 0：垂直予測
- ・モード 1：水平予測
- ・モード 2：DC 予測
- ・モード 3：斜め左下予測
- ・モード 4：斜め右下予測
- ・モード 5：垂直右予測
- ・モード 6：水平下予測
- ・モード 7：垂直左予測
- ・モード 8：水平上予測

【0021】

空間予測は、様々なサイズおよび／または形状のビデオブロックにおいて実行され得る。例えば、（例えば、H.264 / AVC における） 4×4 、 8×8 および 16×16 ピ

クセルのブロックサイズについて、ビデオ信号のルーマ成分の空間予測が実行され得る。例えば、（例えば、H. 264 / AVCにおける） 8×8 のブロックサイズについて、ビデオ信号のクロマ成分の空間予測が実行され得る。 4×4 または 8×8 のルーマブロックについて、合計で9個の予測モード、例えば、（例えば、H. 264 / AVCにおいて）8個の方向予測モードおよびDCモードがサポートされ得る。例えば、サイズ 16×16 のルーマブロックについて、4つの予測モード、すなわち、水平、垂直、DCおよび平面予測がサポートされ得る。

【0022】

方向イントラ予測モードおよび非方向予測モードがサポートされ得る。図4は、33個の方向予測モードおよび2つの非方向予測モードの例を示す図である。図4の33個の方向予測モードおよび2つの非方向予測モードがHEVCによってサポートされ得る。より大きなブロックサイズを用いる空間予測がサポートされ得る。例えば、空間予測は、任意のサイズ、例えば、 4×4 、 8×8 、 16×16 、 32×32 または 64×64 の平方ブロックサイズのブロック上で実行され得る。（例えば、HEVCにおける）方向イントラ予測は、 $1/32$ ピクセルの精度で実行され得る。

【0023】

例えば、方向イントラ予測に加えて、非方向イントラ予測モードが（例えば、H. 264 / AVC、HEVC等において）サポートされ得る。非方向イントラ予測モードは、DCモードおよび/または平面モードを含むことができる。DCモードの場合、予測値は、利用可能な近傍ピクセルを平均化することによって得られることが可能であり、予測値は、ブロック全体に様に適用され得る。平面モードの場合、低速な遷移で平滑な領域を予測するために線形補間が用いられ得る。H. 264 / AVCは、 16×16 のルーマブロックおよびクロマブロックについて平面モードの使用を可能にすることができる。

【0024】

エンコーダ（例えば、エンコーダ100）は、（例えば、図1Aのブロック180において）モード決定を実行し、ビデオブロックのための最良のコーディングモードを決定することができる。エンコーダが（例えばインター予測の代わりに）イントラ予測を適用することを決定するとき、エンコーダは、利用可能なモードの組から最適なイントラ予測モードを決定することができる。選択される方向イントラ予測モードは、入力ビデオブロックにおける任意のテクスチャ、エッジおよび/または構造の方向に関して強力なヒントを与えることができる。図5は、（例えば、 4×4 のブロックについての）水平予測の例の図である。現在の 4×4 のビデオブロックにおけるピクセルを予測するために、既に再構成されたピクセルP0、P1、P2およびP3（例えば、影付きのボックス）が、用いられ得る。水平予測において、再構成されたピクセル、例えば、ピクセルP0、P1、P2および/またはP3は、対応する行の方向に沿って水平方向に伝搬され、 4×4 のブロックを予測することができる。例えば予測は、以下の式（1）に従って実行され得る。ここで、 $L(x, y)$ は、 (x, y) 、 $x, y = 0 \dots 3$ において予測されるピクセルであり得る。

【0025】

$$L(x, 0) = P0$$

$$L(x, 1) = P1$$

$$L(x, 2) = P2$$

$$L(x, 3) = P3$$

(1)

【0026】

図6は、平面モードの例の図である。平面モードは、状況に応じて実行され得る。最上行の最も右側のピクセル（例えば、Tによってマーキングされている）は、最も右側の列内のピクセルを予測するために複製され得る。左列内の最下ピクセル（例えば、Lによってマーキングされている）は、最下行内のピクセルを予測するために複製され得る。（例えば、左ブロックに示すように）水平方向における双線形補間は、中心ピクセルの第1の予測 $H(x, y)$ を生成するように実行され得る。（例えば、右ブロックに示すように

）垂直方向における双線形補間は、中心ピクセルの第2の予測 $V(x, y)$ を生成するように実行され得る。水平予測および垂直予測間の平均化は、 $L(x, y) = (H(x, y) + V(x, y)) / 2$ を用いて最終予測 $L(x, y)$ を得るように実行され得る。

【0027】

図7および図8は、（例えば、図1Aの動き予測ユニット162を用いた）ビデオブロックの動き予測の例を示す図である。図8は、例えば、参照ピクチャ「Ref pic 0」、「Ref pic 1」および「Ref pic 2」を含む例示的な復号されたピクチャバッファを示す図である。現在のピクチャ内のブロックB0、B1およびB2は、それぞれ、参照ピクチャ「Ref pic 0」、「Ref pic 1」および「Ref pic 2」内のブロックから予測され得る。動き予測は、近傍のビデオフレームからのビデオブロックを用いて現在のビデオブロックを予測することができる。動き予測は、時間相関を利用し、かつ/またはビデオ信号に固有の時間冗長性を取り除くことができる。例えば、H.264/AVCおよびHEVCにおいて、時間予測は、様々なサイズのビデオブロック（例えば、ルーマ成分の場合、時間予測ブロックサイズは、H.264/AVCにおいて $16 \times 16 \sim 4 \times 4$ 、HEVCにおいて $64 \times 64 \sim 4 \times 4$ に変動し得る）において実行され得る。 (mv_x, mv_y) の動きベクトルを用いて、時間予測は、式(1)によって与えられるように実行され得る。

【0028】

$$P(x, y) = ref(x - mv_x, y - mv_y) \quad (1)$$

ここで、 $ref(x, y)$ は、参照ピクチャ内のロケーション (x, y) におけるピクセル値であることが可能であり、 $P(x, y)$ は予測されたブロックであることが可能である。ビデオコーディングシステムは、小数ピクセル精度でインター予測をサポートすることができる。動きベクトル (mv_x, mv_y) が小数ピクセル値を有する場合、1または複数の補間フィルタが適用されて、小数ピクセル位置におけるピクセル値を得ることができる。ブロックベースのビデオコーディングシステムは、時間予測を改善するために、複数仮説による予測を用いることができ、例えば、ここで、予測信号は、様々な参照ピクチャから複数の予測信号を組み合わせることによって形成され得る。例えば、H.264/AVCおよび/またはHEVCは、2つの予測信号を組み合わせることができる二重予測を用いることができる。二重予測は、それぞれ参照ピクチャからの2つの予測信号を組み合わせ、以下の式(2)等の予測を形成することができる。

【0029】

【数1】

$$P(x, y) = \frac{P_0(x, y) + P_1(x, y)}{2} = \frac{ref_0(x - mv_{x0}, y - mv_{y0}) + ref_1(x - mv_{x1}, y - mv_{y1})}{2} \quad (2)$$

【0030】

ここで、 $P_0(x, y)$ および $P_1(x, y)$ は、それぞれ第1の予測ブロックおよび第2の予測ブロックであることが可能である。式(2)に示すように、2つの予測ブロックは、それぞれ2つの動きベクトル (mv_{x0}, mv_{y0}) および (mv_{x1}, mv_{y1}) を用いて2つの参照ピクチャ $ref_0(x, y)$ および $ref_1(x, y)$ から動き補償された予測を実行することによって得られることが可能である。予測ブロック $P(x, y)$ は、（例えば、加算器116において）ソースビデオブロックから減算され、予測残差ブロックを形成することができる。予測残差ブロックは、（例えば、変換ユニット104において）変換され、かつ/または（例えば、量子化ユニット106において）量子化され得る。量子化された残差変換係数ブロックは、エントロピーコーディングユニット（例えば、エントロピーコーディングユニット108）に送信され、ビットレートを低減するようにエントロピーコーディングされ得る。エントロピーコーディングされた残差係数は、パッキングされ、出力ビデオビットストリーム（例えば、ビットストリーム120）の一部を形成することができる。

【0031】

図11は、コーディングされたビットストリーム構造の例を示す図である。コーディングされたビットストリーム1000は複数のNAL（ネットワーク抽出層）ユニット1001からなる。NALユニットは、コーディングされたスライス1006等のコーディングされたサンプルデータ、またはパラメータセットデータ、スライスヘッダデータ1005もしくは補助的強化情報データ1007（これは、SEIメッセージと呼ばれる場合がある）等の高水準シンタックスメタデータを含むことができる。パラメータセットは、複数のビットストリーム層（例えば、ビデオパラメータセット1002（VPS））に適用することができるか、または1つの層内のコーディングされたビデオシーケンス（例えば、シーケンスパラメータセット1003（SPS））に適用することができるか、または1つのコーディングされたビデオシーケンス内の複数のコーディングされたピクチャ（例えば、ピクチャパラメータセット1004（PPS））に適用することができる重要なシンタックス要素を含む高水準シンタックス構造である。パラメータセットは、ビデオビットストリームのコーディングされたピクチャとともに送信され得るか、または他の手段（信頼性のあるチャネル、ハードコーディング等を用いた帯域外伝送を含む）を通じて送信され得る。スライスヘッダ1005も、比較的小さいか、またはある特定のスライスまたはピクチャタイプのみに関係するいくつかのピクチャ関連情報を含むことができる高水準シンタックス構造である。SEIメッセージ1007は、復号プロセスによって必要とされない場合があるが、ピクチャ出力タイミングもしくは表示および/または損失検出および秘匿等の様々な他の目的で用いられ得る情報を搬送する。

【0032】

図9は、通信システムの例を示す図である。通信システム1300は、エンコーダ1302と、通信ネットワーク1304と、デコーダ1306とを備えることができる。エンコーダ1302は、接続1308を介して通信ネットワーク1304と通信することができる。接続1308は、有線接続または無線接続であり得る。エンコーダ1302は、図1Aのブロックベースのビデオエンコーダに類似し得る。エンコーダ1302は、単層コーデック（例えば、図1A）または多層コーデックを含むことができる。

【0033】

デコーダ1306は、接続1310を介して通信ネットワーク1306と通信することができる。接続1310は、有線接続または無線接続であり得る。デコーダ1306は、図2Aのブロックベースのビデオデコーダに類似し得る。デコーダ1306は、単層コーデック（例えば、図2A）または多層コーデックを含むことができる。例えば、デコーダ1306は、ピクチャレベルのILPサポートを有する多層（例えば2層）スケーラブルデコーディングシステムであり得る。

【0034】

エンコーダ1302および/またはデコーダ1306は、デジタルテレビ、無線ブロードキャストシステム、ネットワーク要素/端末、コンテンツサーバまたはウェブサーバ等のサーバ（例えば、ハイパーテキスト転送プロトコル（HTTP）サーバ）、携帯情報端末（PDA）、ラップトップまたはデスクトップコンピュータ、タブレットコンピュータ、デジタルカメラ、デジタル記録デバイス、ビデオゲームデバイス、ビデオゲームコンソール、セルラまたは衛星無線電話、デジタルメディアプレーヤ等であるが、これらに限定されない、多岐にわたる有線通信デバイスおよび/または無線送信/受信ユニット（WTRU）に組み込まれ得る。

【0035】

通信ネットワーク1304は、任意の適したタイプの通信ネットワークであり得る。例えば、通信ネットワーク1304は、音声、データ、ビデオ、メッセージ、ブロードキャスト等のコンテンツを複数の無線ユーザに提供する複数のアクセスシステムであり得る。通信ネットワーク1304は、複数の無線ユーザが、無線帯域幅を含むシステムリソースの共有を通じてそのようなコンテンツにアクセスすることを可能にすることができる。例えば、通信ネットワーク1304は、符号分割多重アクセス（CDMA）、時分割多重ア

クセス (T D M A)、周波数分割多重アクセス (F D M A)、直交 F D M A (O F D M A)、単一搬送波 F D M A (S C - F D M A) 等の 1 または複数のチャネルアクセス方法を利用することができる。通信ネットワーク 1 3 0 4 は、複数の接続された通信ネットワークを含むことができる。通信ネットワーク 1 3 0 4 は、インターネット、および / または、セルラネットワーク、W i F i ホットスポット、インターネットサービスプロバイダネットワーク (I S P) 等の 1 つもしくは複数の商用プライベートネットワークを含むことができる。

【0036】

図 1 0 は、例示的な W T R U のシステム図である。W T R U 9 0 0 は、プロセッサ 9 1 8、送受信機 9 2 0、送信 / 受信要素 9 2 2、スピーカ / マイクフォン 9 2 4、キーパッドまたはキーボード 9 2 6、ディスプレイ / タッチパッド 9 2 8、非取り外し可能メモリ 9 3 0、取り外し可能メモリ 9 3 2、電源 9 3 4、グローバルポジショニングシステム (G P S) チップセット 9 3 6 および / または他の周辺機器 9 3 8 を含むことができる。W T R U 9 0 0 は、実施形態との一貫性を保ちながら、上記の要素の任意の部分的組み合わせを含むことができることが理解されよう。更に、エンコーダ (例えば、エンコーダ 1 3 0 2) および / またはデコーダ (例えば、デコーダ 1 3 0 6) が組み込まれ得る端末は、図 1 0 の W T R U 9 0 0 を参照して本明細書において描かれ、説明された要素のうちのいくつかまたは全てを含むことができる。

【0037】

プロセッサ 9 1 8 は、汎用プロセッサ、専用プロセッサ、従来のプロセッサ、デジタル信号プロセッサ (D S P)、グラフィック処理ユニット (G P U)、複数のマイクロプロセッサ、D S P コアに関連付けられた 1 または複数のマイクロプロセッサ、コントローラ、マイクロコントローラ、特定用途向け集積回路 (A S I C)、フィールドプログラマブルゲートアレイ (F P G A) 回路、任意の他のタイプの集積回路 (I C)、状態マシン等であり得る。プロセッサ 9 1 8 は、信号コーディング、データ処理、電力制御、入出力処理、および / または W T R U 9 0 0 が有線および / または無線環境で動作することを可能にする任意の他の機能を実行することができる。プロセッサ 9 1 8 は、送受信機 9 2 0 に結合されることが可能であり、送受信機 9 2 0 は、送信 / 受信要素 9 2 2 に結合されることが可能である。図 1 0 は、プロセッサ 9 1 8 および送受信機 9 2 0 を別個の要素として描いているが、プロセッサ 9 1 8 および送受信機 9 2 0 は、電子パッケージおよび / またはチップにおいて共に集積され得ることが理解されよう。

【0038】

送信 / 受信要素 9 2 2 は、エアインタフェース 9 1 5 を介して別の端末に信号を送信しかつ / または別の端末から信号を受信するように構成され得る。例えば、1 または複数の実施形態において、送信 / 受信要素 9 2 2 は、R F 信号を送信および / または受信するように構成されたアンテナであり得る。1 または複数の実施形態において、送信 / 受信要素 9 2 2 は、例えば、I R、U V または可視光信号を送信および / または受信するように構成されたエミッタ / 検出器であり得る。1 または複数の実施形態では、送信 / 受信要素 9 2 2 は、R F 信号および光信号の双方を送信および / または受信するように構成され得る。送信 / 受信要素 9 2 2 は、無線信号の任意の組み合わせを送信および / または受信するように構成され得る。

【0039】

更に、送信 / 受信要素 9 2 2 が図 1 0 において単一の要素として示されているが、W T R U 9 0 0 は、任意の数の送信 / 受信要素 9 2 2 を含むことができる。より詳細には、W T R U 9 0 0 は、M I M O 技術を採用することができる。このため、いくつかの実施形態では、W T R U 9 0 0 は、エアインタフェース 9 1 5 を介して無線信号を送信および受信するための 2 つ以上の送信 / 受信要素 9 2 2 (例えば、複数のアンテナ) を含むことができる。

【0040】

送受信機 9 2 0 は、送信 / 受信要素 9 2 2 によって送信されることになる信号を変調し

、かつ／または送信／受信要素 9 2 2 によって受信される信号を復調するように構成され得る。上記に示すように、W T R U 9 0 0 は、マルチモード機能を有することができる。このため、送受信機 9 2 0 は、例えば、U T R A および I E E E 8 0 2 . 1 1 等の、W T R U 9 0 0 が複数の R A T を介して通信することを可能にするための複数の送受信機を含むことができる。

【 0 0 4 1 】

W T R U 9 0 0 のプロセッサは、スピーカ／マイクロフォン 9 2 4、キーパッド 9 2 6 および／またはディスプレイ／タッチパッド 9 2 8（例えば、液晶ディスプレイ（L C D）表示ユニットまたは有機発光ダイオード（O L E D）表示ユニット）に結合されることが可能であり、これらからユーザ入力データを受信することができる。プロセッサ 9 1 8 は、スピーカ／マイクロフォン 9 2 4、キーボード 9 2 6 および／またはディスプレイ／タッチパッド 9 2 8 にユーザデータを出力することもできる。更に、プロセッサ 9 1 8 は、非取り外し可能メモリ 9 3 0 および／または取り外し可能メモリ 9 3 2 等の任意のタイプの適切なメモリからの情報にアクセスし、このメモリにデータを記憶することができる。非取り外し可能メモリ 9 3 0 は、ランダムアクセスメモリ（R A M）、リードオンリーメモリ（R O M）、ハードディスクまたは任意の他のタイプのメモリストレージデバイスを含むことができる。取り外し可能メモリ 9 3 2 は、加入者アイデンティティモジュール（S I M）カード、メモリスティック、セキュアデジタル（S D）メモリカード等を含むことができる。1 または複数の実施形態では、プロセッサ 9 1 8 は、サーバまたはホームコンピュータ（図示せず）等の W T R U 9 0 0 上に物理的に位置していないメモリからの情報にアクセスし、このメモリにデータを記憶することができる。

【 0 0 4 2 】

プロセッサ 9 1 8 は、電源 9 3 4 から電力を受信することができ、W T R U 9 0 0 内の他のコンポーネントに対し電力を分配および／または制御するように構成され得る。電源 9 3 4 は、W T R U 9 0 0 に電力供給するための任意の適切なデバイスであり得る。例えば、電源 9 3 4 は、1 または複数の乾電池（例えば、ニッケルカドミウム（N i C d）、ニッケル亜鉛（N i Z n）、ニッケル水素（N i M H）、リチウムイオン（L i イオン）等）、太陽電池、燃料電池等を含むことができる。

【 0 0 4 3 】

プロセッサ 9 1 8 は、G P S チップセット 9 3 6 に結合され得る。これは、W T R U 9 0 0 の現在のロケーションに関するロケーション情報（例えば、緯度および経度）を提供するように構成され得る。G P S チップセット 9 3 6 からの情報に加えて、またはこの情報に代えて、W T R U 9 0 0 は端末（例えば、基地局）からのエアインタフェース 9 1 5 を介してロケーション情報を受信し、かつ／または 2 つ以上の近隣の基地局から信号が受信されるタイミングに基づいてそのロケーションを求めることができる。W T R U 9 0 0 は、実施形態との一貫性を保ちながら、任意の適切なロケーション決定方法によってロケーション情報を取得することができることが理解されよう。

【 0 0 4 4 】

プロセッサ 9 1 8 は、更なる特徴、機能および／または有線もしくは無線接続性を提供する 1 または複数のソフトウェアおよび／またはハードウェアモジュールを含むことができる他の周辺機器 9 3 8 に更に結合され得る。例えば、周辺機器 9 3 8 は、加速度計、方位センサ、動きセンサ、近接性センサ、電子コンパス、衛星送受信機、デジタルカメラおよび／またはビデオレコーダ（例えば、写真および／またはビデオ用）、ユニバーサルシリアルバス（U S B）ポート、振動デバイス、テレビ送受信機、ハンズフリーヘッドセット、B l u e t o o t h（登録商標）モジュール、周波数変調（F M）無線ユニット、およびデジタル音楽プレーヤ、メディアプレーヤ、ビデオゲームプレーヤモジュール、インターネットブラウザ等のソフトウェアモジュールを含むことができる。

【 0 0 4 5 】

例として、W T R U 9 0 0 は、無線信号を送信および／または受信するように構成されることが可能であり、ユーザ機器（U E）、移動局、固定または移動加入者ユニット、ペ

ージャ、セルラ電話、携帯情報端末（PDA）、スマートフォン、ラップトップ、ネットブック、タブレットコンピュータ、パーソナルコンピュータ、無線センサ、消費者家電、または圧縮されたビデオ通信を受信および処理することが可能な任意の他の端末を含むことができる。

【0046】

WTRU 900 および / または通信ネットワーク（例えば、通信ネットワーク 1304）は、広帯域CDMA（WCDMA）を用いてエアインタフェース915を確立することができるユニバーサル移動体通信システム（UMTS）陸上無線アクセス（UTRA）等の無線技術を実施することができる。WCDMAは、高速パケットアクセス（HSPA）および / または発展型HSPA（HSPA+）等の通信プロトコルを含むことができる。HSPAは、高速ダウンリンクパケットアクセス（HSDPA）および / または高速アップリンクパケットアクセス（HSUPA）を含むことができる。WTRU 900 および / または通信ネットワーク（例えば、通信ネットワーク 1304）は、発展型UMTS 陸上無線アクセス（E-UTRA）等の無線技術を実装することができる。この無線技術は、ロングタームエボリューション（LTE）および / またはLTEアドバンスド（LTE-A）を用いてエアインタフェース915を確立することができる。

【0047】

WTRU 900 および / または通信ネットワーク（例えば、通信ネットワーク 1304）は、IEEE 802.16（例えば、マイクロ波アクセスのための世界規模の相互運用性（WiMAX））、CDMA 2000、CDMA 2000 1X、CDMA 2000 EV-DO、暫定基準2000（IS-2000）、暫定基準95（IS-95）、暫定基準856（IS-856）、移動通信のためのグローバルシステム（GSM）、GSMエボリューションのための強化されたデータレート（EDGE）、GSM EDGE（GERAN）等の無線技術を実施することができる。WTRU 900 および / または通信ネットワーク（例えば、通信ネットワーク 1304）は、IEEE 802.11、IEEE 802.15等の無線技術を実施してもよい。

【0048】

無損失コーディング

医療ビデオアプリケーションおよびハイエンドプロフェッショナルビデオアプリケーション等のいくつかのビデオアプリケーションについて、損失を伴うことなく元のビデオ信号内に全ての情報を保持することが望ましい場合がある。そのようなビデオアプリケーションについて、無損失コーディングを用いることができる。無損失コーディングにおいて、変換および量子化等の情報損失を引き起こす場合があるビデオコーデックにおける処理ブロックが変更および / またはバイパスされ得る。それぞれ図1Bおよび図2Bのエンコーダおよびデコーダ構成は、無損失コーディングを達成するために用いられ得る。無損失コーディングにおいて、変換、量子化、逆変換および逆量子化のための処理ブロックは適用されない。更に、図1Bの加算器126および図2Bの加算器226の結果として再構成されたビデオブロックは、元のビデオブロックと数学的に同じであるため、ループ内フィルタリングは必要でない場合があり、実際に、望ましくない歪みを引き起こす場合がある。したがって、いくつかの実施形態では、ループ内フィルタリングのための処理ブロックも適用されない。

【0049】

無線ディスプレイおよびクラウドコンピューティング等の急速に成長するビデオアプリケーションに起因して、スクリーンコンテンツコーディング（SCC）が近年、学界および産業界からの多くの関心を得ている。HEVCは、以前のビデオコーディング規格と比較してコーディング効率において大きな改善を達成しているが、これは、主にカメラによって捕捉される自然のビデオのために設計されたものである。一方、通常、テキストおよびグラフィック等のコンピュータにより生成されたコンテンツからなるスクリーンコンテンツビデオは、自然のコンテンツの特性と極めて異なる特性を示す。これを考慮すると、スクリーンコンテンツコーディングのためにHEVCを拡張することが望ましい。プロッ

ク内コピー（IBC）は、非特許文献1に記載されているように、HEVCスクリーンコンテンツコーディング拡張に採用されている1つのコーディング方法である。IBCは、同じピクチャの既に再構成された領域のピクセルから現在のPUのピクセルを予測することによって、（特に、ピクチャが、テキストおよびグラフィックが豊富なかなりの量のスクリーンコンテンツを含む場合に）1つのピクチャにおいて固有のピクチャ内冗長性を利用するように設計された。インターモードと同様に、IBCモードを用いてコーディングされたCUの場合、1つの予測されたPUとその基準ブロックとの間の変位は、ブロックベクトル（BV）によって表される。BVは、ビットストリーム内の対応する残差とともにコーディングされる。

【0050】

HEVCおよびその拡張において、`transquant_bypass_enabled_flag`と呼ばれるシンタックス要素は、ピクチャパラメータセット（PPS）においてシグナリングされ、変換および量子化が、ブロックごとの単位でバイパスされることが可能であるか否かを示す。非特許文献2に記載されているように、HEVCにおけるPPSシンタックステーブルが以下の表1に示され、`transquant_bypass_enabled_flag`が行22に示されている。

【0051】

【表 1 - 1】

1	pic_parameter_set_rbsp() {	Descriptor
2	pps_pic_parameter_set_id	ue(v)
3	pps_seq_parameter_set_id	ue(v)
4	dependent_slice_segments_enabled_flag	u(1)
5	output_flag_present_flag	u(1)
6	num_extra_slice_header_bits	u(3)
7	sign_data_hiding_enabled_flag	u(1)
8	cabac_init_present_flag	u(1)
9	num_ref_idx_l0_default_active_minus1	ue(v)
10	num_ref_idx_l1_default_active_minus1	ue(v)
11*	init_qp_minus26	se(v)
12	constrained_intra_pred_flag	u(1)
13*	transform_skip_enabled_flag	u(1)
14*	cu_qp_delta_enabled_flag	u(1)
15	if(cu_qp_delta_enabled_flag)	
16*	diff_cu_qp_delta_depth	ue(v)
17*	pps_cb_qp_offset	se(v)
18*	pps_cr_qp_offset	se(v)
19*	pps_slice_chroma_qp_offsets_present_flag	u(1)
20	weighted_pred_flag	u(1)
21	weighted_bipred_flag	u(1)
22	transquant_bypass_enabled_flag	u(1)
23	tiles_enabled_flag	u(1)
24	entropy_coding_sync_enabled_flag	u(1)
25	if(tiles_enabled_flag) {	
26	num_tile_columns_minus1	ue(v)
27	num_tile_rows_minus1	ue(v)
28	uniform_spacing_flag	u(1)
29	if(!uniform_spacing_flag) {	
30	for(i = 0; i < num_tile_columns_minus1; i++)	
31	column_width_minus1[i]	ue(v)
32	for(i = 0; i < num_tile_rows_minus1; i++)	
33	row_height_minus1[i]	ue(v)
34	}	
35*	loop_filter_across_tiles_enabled_flag	u(1)
36	}	
37*	pps_loop_filter_across_slices_enabled_flag	u(1)
38*	deblocking_filter_control_present_flag	u(1)
39	if(deblocking_filter_control_present_flag) {	

【 0 0 5 2 】

【表 1 - 2】

40*	deblocking_filter_override_enabled_flag	u(1)
41*	pps_deblocking_filter_disabled_flag	u(1)
42	if(!pps_deblocking_filter_disabled_flag) {	
43*	pps_beta_offset_div2	se(v)
44*	pps_tc_offset_div2	se(v)
45	}	
46	}	
47*	pps_scaling_list_data_present_flag	u(1)
48	if(pps_scaling_list_data_present_flag)	
49	scaling_list_data()	
50	lists_modification_present_flag	u(1)
51	log2_parallel_merge_level_minus2	ue(v)
52	slice_segment_header_extension_present_flag	u(1)
53	pps_extension_present_flag	u(1)
54	if(pps_extension_present_flag) {	
55	for(i = 0; i < 1; i++)	
56	pps_extension_flag[i]	u(1)
57	pps_extension_7bits	u(7)
58	}	
59	if(pps_extension_flag[0]) {	
60	if(transform_skip_enabled_flag)	
61*	log2_max_transform_skip_block_size_minus2	ue(v)
62	cross_component_prediction_enabled_flag	u(1)
63*	chroma_qp_adjustment_enabled_flag	u(1)
64	if(chroma_qp_adjustment_enabled_flag) {	
65*	diff_cu_chroma_qp_adjustment_depth	ue(v)
66*	chroma_qp_adjustment_table_size_minus1	ue(v)
67	for(i = 0; i <= chroma_qp_adjustment_table_size_minus1; i++) {	
68*	cb_qp_adjustment[i]	se(v)
69*	cr_qp_adjustment[i]	se(v)
70	}	
71	}	
72*	log2_sao_offset_scale_luma	ue(v)
73*	log2_sao_offset_scale_chroma	ue(v)
74	}	
75	if(pps_extension_7bits)	
76	while(more_rbsp_data())	
77	pps_extension_data_flag	u(1)
78	rbp_trailing_bits()	
79	}	

表1: HEVC範囲拡張草案6におけるPPSシンタックステーブル

【 0 0 5 3 】

現在のスライスが、`transquant_bypass_enabled_flag` (表1の行22) が1に設定されている(これは、スライスヘッダ(表2における行5)内の`slice_pic_parameter_set_id`を、適切なPPSを特定するための適切な値に設定することによって行われる) PPSを参照する場合、コーディングユニットまたはCUレベルにおいて、現在のスライスにおける全てのCUについて`cu_transquant_bypass_flag`と呼ばれる追加のフラグがシグナリングされる。`coding_unit`シンタックステーブルが表3に示されている。

【 0 0 5 4 】

【 表 2 - 1 】

1	slice_segment_header() {	Descriptor
2	first_slice_segment_in_pic_flag	u(1)
3	if(nal_unit_type >= BLA_W_LP && nal_unit_type <= RSV_IRAP_VCL23)	
4	no_output_of_prior_pics_flag	u(1)
5	slice_pic_parameter_set_id	ue(v)
6	if(!first_slice_segment_in_pic_flag) {	
7	if(dependent_slice_segments_enabled_flag)	
8	dependent_slice_segment_flag	u(1)
9	slice_segment_address	u(v)
10	}	
11	if(!dependent_slice_segment_flag) {	
12	for(i = 0; i < num_extra_slice_header_bits; i++)	
13	slice_reserved_flag[i]	u(1)
14	slice_type	ue(v)
15	if(output_flag_present_flag)	
16	pic_output_flag	u(1)
17	if(separate_colour_plane_flag == 1)	
18	colour_plane_id	u(2)
19	if(nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) {	
20	slice_pic_order_cnt_lsb	u(v)
21	short_term_ref_pic_set_sps_flag	u(1)
22	if(!short_term_ref_pic_set_sps_flag)	
23	short_term_ref_pic_set(num_short_term_ref_pic_sets)	
24	else if(num_short_term_ref_pic_sets > 1)	
25	short_term_ref_pic_set_idx	u(v)
26	if(long_term_ref_pics_present_flag) {	
27	if(num_long_term_ref_pics_sps > 0)	
28	num_long_term_sps	ue(v)
29	num_long_term_pics	ue(v)
30	for(i = 0; i < num_long_term_sps + num_long_term_pics; i++) {	
31	if(i < num_long_term_sps) {	
32	if(num_long_term_ref_pics_sps > 1)	
33	lt_idx_sps[i]	u(v)
34	} else {	
35	poc_lsb_lt[i]	u(v)
36	used_by_curr_pic_lt_flag[i]	u(1)
37	}	
38	delta_poc_msb_present_flag[i]	u(1)
39	if(delta_poc_msb_present_flag[i])	
40	delta_poc_msb_cycle_lt[i]	ue(v)
41	}	
42	}	
43	if(sps_temporal_mvp_enabled_flag)	

【 0 0 5 5 】

【表 2 - 2】

44	slice_temporal_mvp_enabled_flag	u(1)
45	}	
46	if(sample_adaptive_offset_enabled_flag) {	
47*	slice_sao_luma_flag	u(1)
48	if(ChromaArrayType != 0)	
49*	slice_sao_chroma_flag	u(1)
50	}	
51	if(slice_type == P slice_type == B) {	
52	num_ref_idx_active_override_flag	u(1)
53	if(num_ref_idx_active_override_flag) {	
54	num_ref_idx_l0_active_minus1	ue(v)
55	if(slice_type == B)	
56	num_ref_idx_l1_active_minus1	ue(v)
57	}	
58	if(lists_modification_present_flag && NumPocTotalCurr > 1)	
59	ref_pic_lists_modification()	
60	if(slice_type == B)	
61	mvd_l1_zero_flag	u(1)
62	if(cabac_init_present_flag)	
63	cabac_init_flag	u(1)
64	if(slice_temporal_mvp_enabled_flag) {	
65	if(slice_type == B)	
66	collocated_from_l0_flag	u(1)
67	if((collocated_from_l0_flag && num_ref_idx_l0_active_minus1 > 0) (!collocated_from_l0_flag && num_ref_idx_l1_active_minus1 > 0))	
68	collocated_ref_idx	ue(v)
69	}	
70	if((weighted_pred_flag && slice_type == P) (weighted_bipred_flag && slice_type == B))	
71	pred_weight_table()	
72	five_minus_max_num_merge_cand	ue(v)
73	}	
74*	slice_qp_delta	se(v)
75	if(pps_slice_chroma_qp_offsets_present_flag) {	
76*	slice_cb_qp_offset	se(v)
77*	slice_cr_qp_offset	se(v)
78	}	
79	if(chroma_qp_adjustment_enabled_flag)	
80*	slice_chroma_qp_adjustment_enabled_flag	u(1)
81	if(deblocking_filter_override_enabled_flag)	
82*	deblocking_filter_override_flag	u(1)
83	if(deblocking_filter_override_flag) {	
84*	slice_deblocking_filter_disabled_flag	u(1)
85	if(!slice_deblocking_filter_disabled_flag) {	
86*	slice_beta_offset_div2	se(v)
87*	slice_tc_offset_div2	se(v)

【表 2 - 3】

88	}	
89	}	
90	if(pps_loop_filter_across_slices_enabled_flag && (slice_sao_luma_flag slice_sao_chroma_flag !slice_deblocking_filter_disabled_flag))	
91*	slice_loop_filter_across_slices_enabled_flag	u(1)
92	}	
93	if(tiles_enabled_flag entropy_coding_sync_enabled_flag) {	
94	num_entry_point_offsets	ue(v)
95	if(num_entry_point_offsets > 0) {	
96	offset_len_minus1	ue(v)
97	for(i = 0; i < num_entry_point_offsets; i++)	
98	entry_point_offset_minus1[i]	u(v)
99	}	
100	}	
101	if(slice_segment_header_extension_present_flag) {	
102	slice_segment_header_extension_length	ue(v)
103	for(i = 0; i < slice_segment_header_extension_length; i++)	
104	slice_segment_header_extension_data_byte[i]	u(8)
105	}	
106	byte_alignment()	
107	}	

表2:HEVC範囲拡張草案6におけるスライスセグメントヘッダシンタックステーブル

【 0 0 5 7 】

【表 3】

1	coding_unit(x0, y0, log2CbSize) {	Descriptor
2	if(transquant_bypass_enabled_flag)	
3	cu_transquant_bypass_flag	ae(v)
4	if(slice_type != I)	
5	cu_skip_flag[x0][y0]	ae(v)
6	
7	}	

表3:HEVC範囲拡張草案6におけるコーディングユニット (CU) シンタックステーブル

【 0 0 5 8 】

cu_transquant_bypass_flagの値(表3の行3)は、(cu_transquant_bypass_flagが1に等しい場合)変換および量子化が現在のCUについてバイパスされているか、または(cu_transquant_bypass_flagが0に等しい場合)変換および量子化が現在のCUについて適用されるかを示す。PPSにおけるtransquant_bypass_enabled_flagが0に設定されているとき、追加のCUレベルフラグであるcu_transquant_bypass_flagはシグナリングされず、0である(すなわち、変換および量子化が現在のCUについて適用される)ことが推測される。

【 0 0 5 9 】

フラグcu_transquant_bypass_flagが現在のCUについて1に設定されているとき、変換および量子化は予測残差に適用されない。代わりに、予測残差は、その予測モード情報、動き情報等とともに直接エントロピーコーディングされ、ピ

デオビットストリームにパッキングされる。更に、デブロッキングおよびサンプル適応オフセット (SAO) は、現在のCUのためにバイパスされる。このようにして、ブロックレベル無損失コーディングが達成され、すなわち、再構成されたCUは元のCUと数学的に同じである。

【0060】

無損失コーディングを必要とするビデオアプリケーションの場合、シーケンスレベルにおいて無損失コーディングを適用する (すなわち、シーケンス全体が損失を伴うことなくコーディングされる) か、またはピクチャ/スライスレベルにおいて無損失コーディングを適用する (すなわち、ピクチャ/スライス全体が損失を伴うことなくコーディングされる) ことが望ましい場合がある。第1のバージョンのHEVC規格およびHEVC範囲拡張は、現在、JCT-VCにおける開発下にあり、シーケンス/ピクチャ/スライスレベルの無損失コーディングを示すための高水準シグナリングを含まず、他の方法でも提供しない。代わりに、既存のシグナリング方式を用いてシーケンス/ピクチャ/スライスレベルの無損失コーディングを達成するために、以下の動作、すなわち、(i) `transquant_bypass_enabled_flag` が1に設定されるPPSの作成、(ii) ビデオシーケンス/ピクチャ/スライスのスライスセグメントヘッダにおいて、`transquant_bypass_enabled_flag` が1に等しいPPSを参照すること、(iii) シーケンス/ピクチャ/スライス内の全てのCUについて、フラグ `cu_transquant_bypass_flag` の値を1に設定すること、が実行され得る。

【0061】

無損失モードのブロックレベルのシグナリングのみに依拠することは、いくつかの欠点を有する。特に、これは、シーケンス/ピクチャ/スライスにおける全てのCUについて `cu_transquant_bypass_flag` を送信することを必要とし、これは効率的でない場合がある。このCUレベルのフラグ `cu_transquant_bypass_flag` をコーディングするのにコンテキスト適応型バイナリ算術コーディング (CABAC) が用いられ、これはシグナリングオーバーヘッドを効果的に低減することができるが、これは依然として、デコーダがシーケンス/ピクチャ/スライス内の全てのCUについて追加のシンタックス要素をパースすることを依然として必要とし、これは冗長な動作となる場合がある。

【0062】

更に、無損失モードのためのブロックレベルシグナリングのみが利用可能である場合、デコーダは、完全に無損失のコーディングのために適切に準備することができない場合がある。なぜなら、逆量子化、逆変換、デブロッキングフィルタ、SAO等の処理ブロックが、いくつかの未来のCUのために依然として必要とされる場合がある (すなわち、いくつかの未来のCUの `cu_transquant_bypass_flag` の値が0に設定される場合がある) ことが起こり得るためである。これにより、そうでなければ、不要な処理ブロックをシャットダウンすることにより得られたはずの利用可能な電力節減が制限される。

【0063】

同様に、現在のビデオビットストリームは、無損失符号化に必要とされない様々なレベルの複数のシンタックス要素を含む。これらの要素は、高水準シンタックス構造における逆量子化、逆変換、デブロッキングおよびSAOに関係する。例えば、PPSにおいて、いくつかのシンタックス要素 (例えば、それぞれ表1の行11、14、17、18に示すように `init_qp_minus26`、`cu_qp_delta_enabled_flag`、`pps_cb/cr_qp_offset`) は、逆量子化プロセスに関係付けられ、いくつかのシンタックス要素 (例えば、それぞれ表1の行40、43、44に示すように `deblocking_filter_override_enabled_flag`、`pps_beta_offset_div2`、`pps_tc_offset_div2`) は、デブロッキングプロセスに関係付けられ、いくつかのシンタックス要素 (それぞ

れ表 1 の行 7 2、7 3 に示すように `log2_sao_offset_scale_luma`、`log2_sao_offset_scale_chroma` は S A O プロセスに関係付けられている。同様に、スライスセグメントヘッダにおけるいくつかのシンタックス要素は、逆量子化（それぞれ表 2 の行 7 4、7 6、7 7 に示すように、`slice_qp_delta`、`slice_cb_qp_offset`、`slice_cr_qp_offset`）、デブロッキング（行 8 6、8 7 に示すように、`slice_beta_offset_div2`、`slice_tc_offset_div2`）、および S A O（表 2 の行 4 7 および 4 9 に示すように、`slice_sao_luma_flag`、`slice_sao_chroma_flag`）に関係付けられている。これらのシンタックス要素および他のシンタックス要素は、表 1 ~ 表 3 においてアスタリスクでマーキングされている。無損失コーディングがシーケンス / ピクチャ / スライスレベルで適用される時、これらのシンタックス要素をシグナリングする必要がある場合があり、これによってシグナリングオーバーヘッドが低減する。一方、無損失コーディングモードの高水準のインジェクションがない場合、これらの高水準シンタックス要素は、ビットストリーム内に符号化され、それぞれのデコーダに送信されなくてはならない。

【0064】

また更に、ビデオビットストリームは、ブロックレベルにおいて（例えば、C U レベルにおいて）変換に関係するシグナリングを含むことができる。例えば、`transform_tree`（）シンタックス構造（`transform_tree`（）の単純化されたバージョンが表 7 に示されている）において、変換ユニットの四分木分割が行われているか否か（`transform_split_flag`）および / または変換ブロックにおいて、ルーマおよびクロマのための任意の非ゼロ係数が存在するか否かを示すフラグ（`cbf_luma`、`cbf_cb` および `cbf_cr`）がシグナリングされる。無損失コーディングモードにおいて、`transform_tree`（）シンタックスは、これらのフラグのシグナリングを明示的にバイパスし、代わりに、これらのフラグを適切なデフォルト値に設定することによって単純化され得る。

【0065】

シグナリング無損失コーディングモード

本明細書において記載されているのは、上記の不利な点のうちの 1 つまたは複数を克服することができる、無損失コーディングモードにおいて用いられるシグナリング方法の様々な実施形態である。1 つのそのような実施形態では、無損失コーディングモードのシグナリングは、P P S シンタックス構造を変更することによって行われ得る。表 4 は、いくつかの実施形態による変更された P P S シンタックステーブルを示す。ここで、この P P S を参照するスライス内の全てのコーディングユニットのフラグ `cu_transquant_bypass_flag` のデフォルト値を示すための追加のフラグである `transquant_bypass_default_flag` が追加される。この実施形態では、ビットストリームは、各個々のコーディングユニットにおいてフラグ `cu_transquant_bypass_flag` をシグナリングする必要がある。この新たなフラグを 1 に設定し、フラグ `transquant_bypass_enabled_flag` を 0 に設定することによって、エンコーダは、シーケンスおよび / またはピクチャおよび / またはスライスレベルにおいて、無損失コーディングが適用されることをデコーダに示すことができる。すなわち、C U レベルのシグナリングを一切伴うことなく、この現在の P P S を参照する全ての C U の変換、変換スキップ、量子化およびループ内フィルタリングプロセスがバイパスされる。

【0066】

`cu_transquant_bypass_flag` が存在しないとき、新たなフラグ `transquant_bypass_default_flag` が `cu_transquant_bypass_flag` のデフォルト値を指定するとみなされ得る。

【0067】

0 の C U レベルフラグ `cu_transquant_bypass_flag` の既存の

セマンティクスは、以下のように変更され得る。1に等しい`cu__transquant__bypass__flag`は、副節8.6に指定されるようなスケーリングおよび変換プロセスを指定し、副節8.7において指定されるようなループ内フィルタプロセスはバイパスされる。`cu__transquant__bypass__flag`が存在しないとき、このフラグは`transquant__bypass__default__flag`に等しいことが推測される。表4に示されているように、この新たなフラグ`transquant__bypass__default__flag`（表4、行11）は、表4においてアスタリスクでマーキングされている逆量子化、逆変換およびループ内フィルタリングプロセスに関係するPPS内の（表4の行12、15、17、21、28、42、45、57、74および84に示すような）複数のシンタックス要素の存在を条件付けるために用いられ得る。これらの条件は、これらのシンタックス要素が、`transquant__bypass__default__flag`が0に等しいとき（すなわち、損失ありコーディングが適用されるとき）にのみ送信されるように設定される。`transquant__bypass__default__flag`が1に等しいとき（すなわち、無損失コーディングが適用されるとき）、これらのシンタックス要素は送信されない。代わりに、それらの値は0であることが推測される。例えば、`transquant__bypass__default__flag`が1に設定されているときに、`cu__qp__delta__enabled__flag`の値が0であると推測することによって、デルタQP関連シンタックス要素がCUレベルにおいてシグナリングされないことが示され、これにより、ビットを節減し、シンタックスパースを単純化する。

【0068】

更に、`transform__skip__enabled__flag`の存在を条件付けるために新たなフラグ`transquant__bypass__default__flag`が用いられる。`transform__skip__enabled__flag`は、変換プロセスのみをバイパスする（量子化プロセスをバイパスしない）ために用いられる。したがって、これは`transquant__bypass__enabled__flag`のサブセットである。更に、新たなフラグ`transquant__bypass__default__flag`は、`transquant__bypass__enabled__flag`の存在を条件付けるために用いられる。このようにして、`transquant__bypass__default__flag`が1に設定されるとき（すなわち、無損失コーディングモード）、`transquant__bypass__enabled__flag`が0であることが推測され、CUレベルにおける`cu__transquant__bypass__flag`のシグナリングがスキップされる。

【0069】

【表4 - 1】

1	<code>pic_parameter_set_rbsp() {</code>	Descriptor
2	<code>pps_pic_parameter_set_id</code>	<code>ue(v)</code>
3	<code>pps_seq_parameter_set_id</code>	<code>ue(v)</code>
4	<code>dependent_slice_segments_enabled_flag</code>	<code>u(1)</code>
5	<code>output_flag_present_flag</code>	<code>u(1)</code>
6	<code>num_extra_slice_header_bits</code>	<code>u(3)</code>
7	<code>sign_data_hiding_enabled_flag</code>	<code>u(1)</code>
8	<code>cabac_init_present_flag</code>	<code>u(1)</code>
9	<code>num_ref_idx_l0_default_active_minus1</code>	<code>ue(v)</code>
10	<code>num_ref_idx_l1_default_active_minus1</code>	<code>ue(v)</code>
11†	<code>transquant_bypass_default_flag</code>	<code>u(1)</code>
12†	<code>if(!transquant_bypass_default_flag)</code>	
13*	<code>init_qp_minus26</code>	<code>se(v)</code>
14	<code>constrained_intra_pred_flag</code>	<code>u(1)</code>

【 0 0 7 0 】

【 表 4 - 2 】

15†	if(!transquant_bypass_default_flag)	
16*	transform_skip_enabled_flag	u(1)
17†	if(!transquant_bypass_default_flag)	
18*	cu_qp_delta_enabled_flag	u(1)
19	if(cu_qp_delta_enabled_flag)	
20*	diff_cu_qp_delta_depth	ue(v)
21†	if(!transquant_bypass_default_flag) {	
22*	pps_cb_qp_offset	se(v)
23*	pps_cr_qp_offset	se(v)
24*	pps_slice_chroma_qp_offsets_present_flag	u(1)
25†	}	
26	weighted_pred_flag	u(1)
27	weighted_bipred_flag	u(1)
28†	if(!transquant_bypass_default_flag)	
29	transquant_bypass_enabled_flag	u(1)
30	tiles_enabled_flag	u(1)
31	entropy_coding_sync_enabled_flag	u(1)
32	if(tiles_enabled_flag) {	
33	num_tile_columns_minus1	ue(v)
34	num_tile_rows_minus1	ue(v)
35	uniform_spacing_flag	u(1)
36	if(!uniform_spacing_flag) {	
37	for(i = 0; i < num_tile_columns_minus1; i++)	
38	column_width_minus1[i]	ue(v)
39	for(i = 0; i < num_tile_rows_minus1; i++)	
40	row_height_minus1[i]	ue(v)
41	}	
42†	if(!transquant_bypass_default_flag)	
43*	loop_filter_across_tiles_enabled_flag	u(1)
44	}	
45†	if(!transquant_bypass_default_flag) {	
46*	pps_loop_filter_across_slices_enabled_flag	u(1)
47*	deblocking_filter_control_present_flag	u(1)
48†	}	
49	if(deblocking_filter_control_present_flag) {	
50*	deblocking_filter_override_enabled_flag	u(1)
51*	pps_deblocking_filter_disabled_flag	u(1)
52	if(!pps_deblocking_filter_disabled_flag) {	
53*	pps_beta_offset_div2	se(v)
54*	pps_tc_offset_div2	se(v)
55	}	
56	}	
57†	if(!transquant_bypass_default_flag)	
58*	pps_scaling_list_data_present_flag	u(1)
59	if(pps_scaling_list_data_present_flag)	
60	scaling_list_data()	
61	lists_modification_present_flag	u(1)

【 0 0 7 1 】

【 表 4 - 3 】

62	log2_parallel_merge_level_minus2	ue(v)
63	slice_segment_header_extension_present_flag	u(1)
64	pps_extension_present_flag	u(1)
65	if(pps_extension_present_flag) {	
66	for(i = 0; i < 1; i++)	
67	pps_extension_flag[i]	u(1)
68	pps_extension_7bits	u(7)
69	}	
70	if(pps_extension_flag[0]) {	
71	if(transform_skip_enabled_flag)	
72*	log2_max_transform_skip_block_size_minus2	ue(v)
73	cross_component_prediction_enabled_flag	u(1)
74†	if(!transquant_bypass_default_flag)	
75*	chroma_qp_adjustment_enabled_flag	u(1)
76	if(chroma_qp_adjustment_enabled_flag) {	
77*	diff_cu_chroma_qp_adjustment_depth	ue(v)
78*	chroma_qp_adjustment_table_size_minus1	ue(v)
79	for(i = 0; i <= chroma_qp_adjustment_table_size_minus1; i++) {	
80*	cb_qp_adjustment[i]	se(v)
81*	cr_qp_adjustment[i]	se(v)
82	}	
83	}	
84†	if(!transquant_bypass_default_flag) {	
85*	log2_sao_offset_scale_luma	ue(v)
86*	log2_sao_offset_scale_chroma	ue(v)
87†	}	
88	}	
89	if(pps_extension_7bits)	
90	while(more_rbsp_data())	
91	pps_extension_data_flag	u(1)
92	rbp_trailing_bits()	
93	}	

表4:無損失コーディングモードシグナリングを用いたPPSシンタックステーブル

【 0 0 7 2 】

別の実施形態において、HEVC範囲拡張を通じた提案される追加のシンタックス要素を実施するために、`transquant_bypass_default_flag`のロケーションが、PPS拡張の一部として（すなわち、`pps_extension_flag[0]`の「if」条件内で）更に下方に動かされ得る。この構成は、HEVC範囲拡張のPPSシンタックスが、HEVC標準規格の第1のバージョンと最大限の後方互換性を有することを確実にすることができる（非特許文献3）。表5は、そのような構成の一例を示す。この構成において、新たなフラグ`transquant_bypass_default_flag`は、`transquant_bypass_enabled_flag`の値で条件付けられ得る。すなわち、`transquant_bypass_default_flag`は、`transquant_bypass_enabled_flag`が0に等しいときにしかシグナリングされない。`transquant_bypass_enabled_flag`が1に等しいとき、`transquant_bypass_default_flag`はシグナリングされず、このPPSを参照する各コーディン

グユニットにおける `cu__transquant__bypass__flag` の値は、コーディングユニットレベルのビットストリームから明示的に受信される。一方、この実施形態の構成において、新たなフラグは、主要な PPS シンタックスの一部でないため、上記で説明したような表 4 内の量子化、変形、変形スキップおよびループ内フィルタリングに關係する既存の PPS シンタックス要素（アスタリスクでマーキングされたシンタックス要素）の存在を条件付けするために用いられ得ない。別のフラグ `lossless__coding__conformance__flag` は、`transquant__bypass__default__flag` が 1 である場合にシグナリングされる。フラグ `lossless__coding__conformance__flag` が 1 である場合、無損失コーディングモードにおいて用いられていないシンタックス要素のシグナリングされた値が適切な値を有することを確実にするために、ビットストリーム適合性要件が適用され得る。例えば、適合するビットストリームにおいて、`cu__qp__delta__enabled__flag`、`pps__loop__filter__across__slices__enabled__flag`、`deblock__filter__control__present__flag`、`loop__filter__across__tiles__enabled__flag`、`pps__scaling__list__data__present__flag` 等を含むシンタックス要素の値は、新たなフラグ `transquant__bypass__default__flag` が 1 に設定されている場合に、0 に設定されるように要求され得る。そのようなビットストリーム適合性要件は、無損失コーディングにおいて用いられていないシンタックス要素に關係するシグナリングオーバーヘッドを最小にするのに役立つことができる。

【0073】

上記で説明したように、いくつかの実施形態では、PPS は、無損失コーディングのインジェクションのために、新たなフラグ `transquant__bypass__default__flag` を搬送するように用いられる。一方、他の実施形態では、提案されたフラグを搬送するために、シーケンスパラメータセット（SPS）またはビデオパラメータセット（VPS）等の他の高水準シンタックス構造も用いられてもよい。あるいは、スライスレベルの無損失コーディングインジェクションのみが望ましい場合、提案された新たなフラグを搬送するためにスライスセグメントヘッダが用いられてもよい。

【0074】

量子化、変換およびループ内フィルタリングプロセスに關係するいくつかのシンタックス要素は、SPS の一部としてシグナリングされ得ることに留意されたい。そのような SPS シンタックス要素の例は、`log2__min__transform__block__size__minus2`、`log2__diffmax__min__transform__block__size`、`max__transform__hierarchy__depth__inter`、`max__transform__hierarchy__depth__intra`、`scaling__list__enabled__flag`、`sample__adaptive__offset__enabled__flag`、`pcm__loop__filter__disabled__flag`、`transform__skip__rotation__enabled__flag`、`transform__skip__context__enabled__flag` 等を含む。無損失コーディングモードが SPS レベルまたは VPS レベルにおけるフラグ `transquant__bypass__default__flag` を用いて示される場合、提案されるフラグは、量子化、変形、変形スキップ、変形スキップ回転、およびループ内フィルタリングプロセスに關係する SPS シンタックス要素の存在を条件付けるために用いられ得る。あるいは、同様のビットストリーム適合性要件もまた、これらのシンタックス要素のために適切な値がシグナリングされることを確実にするために、例えば、適合性フラグが設定される場合にループ内フィルタが無効にされることを確実にするために適用され得る。

【0075】

【表 5】

1	pic_parameter_set_rbsp() {	Descriptor
2	pps_pic_parameter_set_id	ue(v)
3	pps_seq_parameter_set_id	ue(v)
4	...	
5	if(pps_extension_flag[0]) {	
6†	if(!transquant_bypass_enabled_flag)	
7†	transquant_bypass_default_flag	u(1)
8†	if (transquant_bypass_default_flag)	
9†	lossless_coding_conformance_flag	u(1)
10†	if(transform_skip_enabled_flag && !transquant_bypass_default_flag)	
11*	log2_max_transform_skip_block_size_minus2	ue(v)
12	cross_component_prediction_enabled_flag	u(1)
13†	if(!transquant_bypass_default_flag)	
14*	chroma_qp_adjustment_enabled_flag	u(1)
15	if(chroma_qp_adjustment_enabled_flag) {	
16*	diff_cu_chroma_qp_adjustment_depth	ue(v)
17*	chroma_qp_adjustment_table_size_minus1	ue(v)
18	for(i = 0; i <= chroma_qp_adjustment_table_size_minus1; i++) {	
19*	cb_qp_adjustment[i]	se(v)
20*	cr_qp_adjustment[i]	se(v)
21	}	
22	}	
23†	if(!transquant_bypass_default_flag) {	
24*	log2_sao_offset_scale_luma	ue(v)
25*	log2_sao_offset_scale_chroma	ue(v)
26†	}	
27	}	
28	...	
29	}	

表5:PPS拡張におけるtransquant_bypass_default_flagのシグナリング

【 0 0 7 6 】

スライスヘッダシグナリング

更なる実施形態において、スライスヘッダシグナリングが用いられ得る。PPSと同様に、表2におけるスライスセグメントヘッダも、変形、量子化、およびループ内フィルタリング処理ブロックのために用いられる複数のシンタックス要素（アスタリスクでマーキングされたシンタックス要素）を含む。これらのシンタックス要素は、新たなフラグtransquant_bypass_default_flagの値で条件付けられることが可能であり、無損失コーディングが、transquant_bypass_default_flagを1に設定することによって示されているとき、これらのシンタックス要素はシグナリングされる必要がない場合がある。表6はそのような例を示す。

【 0 0 7 7 】

表2に示すように、スライスセグメントヘッダは、量子化、変換およびループ内フィルタリングプロセスに関係付けられたいくつかのシンタックス要素を含む。そのようなスライスセグメントヘッダシンタックス要素（それらの行番号がアスタリスクでマーキングされている）は、slice_sao_luma_flag、slice_sao_chroma_flag、slice_qp_delta、slice_cb_qp_offset、slice_cr_qp_offset、slice_chroma_qp_ad

justment__enabled__flag、deblocking__filter__override__flag、slice__deblocking__filter__disabled__flag、slice__beta__offset__div2、slice__tc__offset__div2およびslice__loop__filter__across__slices__enabled__flagを含む。スライスレベル無損失コーディングが、スライスセグメントヘッダにおいて提案されたフラグtransquant__bypass__default__flagをシグナリングすることによって有効にされている場合、提案されたフラグは、スライスセグメントヘッダにおける量子化、変換およびループ内フィルタリングプロセスに関係付けられたシンタックス要素の存在を条件付けるために用いられ得る。このため、1つの実施形態では、フラグtransquant__bypass__default__flagは、スライスセグメントヘッダにおける量子化、変換およびループ内フィルタリングに関係付けられたシンタックス要素の前に配置される。別の実施形態では、フラグtransquant__bypass__default__flagは、スライスセグメントヘッダにおける代替的なロケーション、例えば、量子化、変換およびループ内フィルタリングに関係付けられたシンタックス要素の後に配置され得る。この場合、ビットストリーム適合性要件は、transquant__bypass__default__flagが1に設定されている場合に、これらのシンタックス要素の値が適切に設定されることを確実にするように適用されるべきである。transquant__bypass__default__flagを1に設定することは、現在のスライス内の全てのコーディングユニットが、無損失モードにおいて、ビットストリーム内の各個々のコーディングユニットのcu__transquant__bypass__flagをシグナリングすることなくコーディングされることを示す。更に、以下のビットストリーム適合性要件、すなわち、提案されるフラグtransquant__bypass__default__flagが1に設定されているとき、現在のスライスが参照するPPSのフラグtransquant__bypass__enabled__flagの値が1に等しいことが適用され得る。

【0078】

表6は、提案されるフラグtransquant__bypass__default__flagが、スライスセグメントヘッダにおける量子化、変換およびループ内フィルタリングに関係付けられたシンタックス要素の前にシグナリングされるときの変更されたスライスセグメントヘッダの1つの例を示す。表6の例は、提案されるtransquant__bypass__default__flagがスライスセグメントヘッダにおいてシグナリングされることを示しているが、このフラグは代わりにPPS(表5)においてシグナリングされてもよく、アスタリスクで示されるスライスセグメントヘッダシンタックス要素の存在を条件付けるように用いられてもよい。

【0079】

【表 6】

1	slice_segment_header() {	Descriptor
2	...	
3	if(!dependent_slice_segment_flag) {	
4	for(i = 0; i < num_extra_slice_header_bits; i++)	
5	slice_reserved_flag[i]	u(1)
6	slice_type	ue(v)
7	if(output_flag_present_flag)	
8	pic_output_flag	u(1)
9	if(separate_colour_plane_flag == 1)	
10	colour_plane_id	u(2)
11†	transquant_bypass_default_flag	u(1)
12	...	
13†	if(sample_adaptive_offset_enabled_flag && !transquant_bypass_default_flag) {	
14*	slice_sao_luma_flag	u(1)
15	if(ChromaArrayType != 0)	
16*	slice_sao_chroma_flag	u(1)
17	}	
18	...	
19†	if(!transquant_bypass_default_flag)	
20*	slice_qp_delta	se(v)
21†	if(pps_slice_chroma_qp_offsets_present_flag && !transquant_bypass_default_flag) {	
22*	slice_cb_qp_offset	sc(v)
23*	slice_cr_qp_offset	se(v)
24	}	
25†	if(chroma_qp_adjustment_enabled_flag && !transquant_bypass_default_flag)	
26*	slice_chroma_qp_adjustment_enabled_flag	u(1)
27†	if(deblocking_filter_override_enabled_flag && !transquant_bypass_default_flag)	
28*	deblocking_filter_override_flag	u(1)
29†	if(deblocking_filter_override_flag && !transquant_bypass_default_flag) {	
30*	slice_deblocking_filter_disabled_flag	u(1)
31	if(!slice_deblocking_filter_disabled_flag) {	
32*	slice_beta_offset_div2	se(v)
33*	slice_tc_offset_div2	se(v)
34	}	
35	}	
36†	if(pps_loop_filter_across_slices_enabled_flag && !transquant_bypass_default_flag && (slice_sao_luma_flag slice_sao_chroma_flag !slice_deblocking_filter_disabled_flag))	
37*	slice_loop_filter_across_slices_enabled_flag	u(1)
38	}	
39	...	
40	}	

表6:transquant_bypass_default_flagを用いて変更されたスライスセグメントヘッダシンタックス

【 0 0 8 0 】

t r a n s q u a n t _ b y p a s s _ d e f a u l t _ f l a g が P P S 拡張の一部としてシグナリングされる表 5 を例として用いると、図 1 2 は、変更された P P S 拡張シンタックスをデコード側でパースするためのアルゴリズムの 1 つの実施形態を示す。変更された P P S 拡張において、提案されるフラグ t r a n s q u a n t _ b y p a s s _ d e f a u l t _ f l a g がパースされ、その値がステップ 1 2 0 2 において検査される。t r a n s q u a n t _ b y p a s s _ d e f a u l t _ f l a g が 0 に等しい場合、高効率ビデオコーディング (H E V C) 範囲拡張テキスト仕様 (草案 6) における既存の P P S 拡張シンタックス要素がパースされ、処理される (ステップ 1 2 0 4) 。 t r a n s

`quant_bypass_default_flag` が 1 に等しい場合、シンタックス要素 `chroma_qp_adjustment_enabled_flag`、`log2_sao_offset_scale_luma` および `log2_sao_offset_scale_chroma` がパースされず、代わりにそれらの値が 0 であると推測される (ステップ 1206)。フラグ `lossless_coding_conformance_flag` が 1 である場合 (ステップ 1207)、ビットストリーム適合性要件が適用される。適合性要件は、量子化、変換、およびループ内フィルタリングに関係付けられた既存の PPS シンタックス要素 (例えば、`init_qp_minus26`、`cu_qp_delta_enabled_flag`、`pps_cb_cr_qp_offset`、`cu_qp_delta_enabled_flag`、`pps_loop_filter_across_slices_enabled_flag`、`deblocking_filter_control_present_flag`、`loop_filter_across_tiles_enabled_flag`、`pps_scaling_list_data_present_flag` 等) のシグナリングされた値をチェックし、これらのシンタックス要素が適切に無効にされていることを確実にすることによって適用される (ステップ 1208)。これらのシンタックス要素のうちの 1 つまたは複数が無効にされていない場合、デコーダはビットストリーム適合性違反を報告し (ステップ 1210)、そうでない場合、PPS 拡張のパースが正常に完了する。

【0081】

変換木シンタックスシグナリング

無損失コーディングのための変換四分木分割

HEVC および HEVC 範囲拡張は、変換木分割シンタックスを用いて、変換ユニット (TU) のサイズをシグナリングする。1 つの実施形態では、高効率ビデオコーディング (HEVC) 範囲拡張テキスト仕様 (草案 6) のセクション 7.3.8.8 において指定される変換木シンタックスは、新たな提案されるフラグ `transquant_bypass_default_flag` に基づいて変更される必要がない。別の実施形態では、変換木シンタックスは、`transquant_bypass_default_flag` が 1 に等しいとき、四分木分割フラグ (`split_transform_flag`)、ならびに / またはルーマおよびクロマ成分のためのコーディングされたブロックフラグ (`cbf_luma`、`cbf_cb`、`cbf_cr`) のシグナリングをバイパスするように単純化され得る。単純化された `transform_tree()` シンタックスが表 7 に示されている。`transquant_bypass_default_flag` は、`split_transform_flag` の存在の更なる条件として用いられる。存在しないとき、ほとんどの場合、`split_transform_flag` の値は 0 であると推測され (すなわち、変換四分木分割は適用されない)、変換四分木分割が施行されるとき (例えば、 $N \times N$ の区分がイントラコーディングにおいて用いられるとき、または現在の CU サイズが 32×32 の最も大きな変換サイズよりも大きいとき等) のいくつかの既存の特殊な事例の場合、1 であると推測される (すなわち、変換四分木分割が適用される)。更に、表 7 に示すように、`transquant_bypass_default_flag` が 1 に等しいとき、`transform_tree()` 内の (ルーマ成分およびクロマ成分についての) 全ての `cbf` フラグのシグナリングがスキップされ得る。代わりにそれらの値は、1 に等しいと推定され得る。なぜなら、無損失コーディングにおいて量子化プロセスがないことに起因して、`cbf` フラグが非ゼロ値を有する可能性が最も高いためである。

【0082】

`cbf_luma`、`cbf_cb`、および `cbf_cr` のセマンティクスも変更され得る。1 に等しい `cbf_luma[x0][y0][trafoDepth]` の値は、ルーマ変換ブロックが、0 に等しくない 1 または複数の変換係数レベルを含むことを指定するために用いられ得る。アレイインデックス `x0`、`y0` は、ピクチャの左上のルーマサンプルに対する検討中の変換ブロックの左上のルーマサンプルのロケーション (`x0`、`y0`

)を指定する。アレイインデックス `trafoDepth` は、変換コーディングの目的での、ブロックへのコーディングブロックの現在の細分化レベルを指定する。コーディングブロックに対応するブロックについて、`trafoDepth` は 0 に等しい。`cbf_luma[x0][y0][trafoDepth]` は、存在しないとき、1 に等しいと推測される。

【0083】

1 に等しい `cbf_cb[x0][y0][trafoDepth]` の値は、Cb 変換ブロックが、0 に等しくない 1 または複数の変換係数レベルを含むことを示すために用いられ得る。アレイインデックス `x0`、`y0` は、検討される変換ユニットの左上ロケーション (`x0`、`y0`) を指定する。アレイインデックス `trafoDepth` は、変換コーディングの目的での、ブロックへのコーディングブロックの現在の細分化レベルを指定する。コーディングブロックに対応するブロックについて、`trafoDepth` は 0 に等しい。`cbf_cb[x0][y0][trafoDepth]` は、存在しないとき、`cbf_cb[x0][y0][trafoDepth]` の値は、以下のように推測され得る。

【0084】

- ・ `transquant_bypass_default_flag` が 1 に等しい場合、`cbf_cb[x0][y0][trafoDepth]` は 1 に等しいと推測される
- ・ そうではなく、`trafoDepth` が 0 よりも大きく、かつ `log2TrafoSize` が 2 に等しい場合、`cbf_cb[x0][y0][trafoDepth]` は `cbf_cb[xBase][yBase][trafoDepth - 1]` に等しいと推測される
- ・ そうでない場合、`cbf_cb[x0][y0][trafoDepth]` は 0 に等しいと推測される。

【0085】

1 に等しい `cbf_cr[x0][y0][trafoDepth]` の値は、Cr 変換ブロックが、0 に等しくない 1 または複数の変換係数レベルを含むことを指定する。アレイインデックス `x0`、`y0` は、検討される変換ユニットの左上のロケーション (`x0`、`y0`) を指定する。アレイインデックス `trafoDepth` は、変換コーディングの目的での、ブロックへのコーディングブロックの現在の細分化レベルを指定する。コーディングブロックに対応するブロックについて、`trafoDepth` の値は 0 に等しい。

【0086】

`cbf_cr[x0][y0][trafoDepth]` が存在しないとき、`cbf_cr[x0][y0][trafoDepth]` の値は以下のように推測され得る。

【0087】

- ・ `transquant_bypass_default_flag` が 1 に等しいとき、`cbf_cr[x0][y0][trafoDepth]` は 1 に等しいと推測される
- ・ そうではなく、`trafoDepth` が 0 よりも大きく、かつ `log2TrafoSize` が 2 に等しい場合、`cbf_cr[x0][y0][trafoDepth]` が `cbf_cr[xBase][yBase][trafoDepth - 1]` に等しいと推測される
- ・ そうでない場合、`cbf_cr[x0][y0][trafoDepth]` は 0 に等しいと推測される。

【0088】

表 7 は、提案される高水準フラグ `transquant_bypass_default_flag` を用いて `transform_tree()` シンタックスを単純化する例を示すが、同じことに従って、`split_transform_flag`、`cbf_luma`、`cbf_cb` および `cbf_cr` の存在を条件付けるために、既存のブロックレベルのフラグ `cu_transquant_bypass_flag` が代わりに用いられてもよい。

【 0 0 8 9 】

【 表 7 】

1	transform_trce(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	Descriptor
2†	if(!transquant_bypass_default_flag && log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && (trafoDepth == 0)))	
3	split_transform_flag[x0][y0][trafoDepth]	ae(v)
4†	if(!transquant_bypass_default_flag && log2TrafoSize > 2 && ChromaArrayType != 0) {	
5	if(trafoDepth == 0 cbf_cb[xBase][yBase][trafoDepth - 1]) {	
6	cbf_cb[x0][y0][trafoDepth]	ae(v)
7	if(ChromaArrayType == 2 && !split_transform_flag[x0][y0][trafoDepth])	
8	cbf_cb[x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth]	
9	}	
10	if(trafoDepth == 0 cbf_cr[xBase][yBase][trafoDepth - 1]) {	
11	cbf_cr[x0][y0][trafoDepth]	ae(v)
12	if(ChromaArrayType == 2 && !split_transform_flag[x0][y0][trafoDepth])	
13	cbf_cr[x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth]	
14	}	
15	}	
16	if(split_transform_flag[x0][y0][trafoDepth]) {	
17	x1 = x0 + (1 << (log2TrafoSize - 1))	
18	y1 = y0 + (1 << (log2TrafoSize - 1))	
19	transform_tree(x0, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 0)	
20	transform_trce(x1, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 1)	
21	transform_tree(x0, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 2)	
22	transform_trce(x1, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 3)	
23	} else {	
24†	if(!transquant_bypass_default_flag && ((CuPredMode[x0][y0] == MODE_INTRA && intra_bc_flag[x0][y0] != 1) trafoDepth != 0 cbf_cb[x0][y0][trafoDepth] cbf_cr[x0][y0][trafoDepth] (ChromaArrayType == 2 && (cbf_cb[x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth] cbf_cr[x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth]))))	
25	cbf_luma[x0][y0][trafoDepth]	ae(v)
26	transform_unit(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx)	
27	}	
28	}	

表7:単純化された変換木シンタックス

【 0 0 9 0 】

表 7 に示す一実施形態において、異なるサイズの、かつ異なるコーディングモードでコーディングされたCUについて、四分木分割フラグsplit_transform_flagおよびコーディングされたブロックフラグcbf_luma、cbf_cbおよびcbf_crはバイパスされ、フラグtransquant_bypass_default_flagが1に等しいとき、対応するデフォルト値であると推測される。一方、入力ビデオの特性に依拠して、異なるブロックサイズおよびコーディングモードを用いるCUの残差は、独特の統計的特性を呈することができる。この場合、1つのピクチャまたはシーケンス内の全てのCUについて変換四分木分割を無効にすることは有益でない場合がある。代わりに、コーディング性能を改善するために、開示の1つの実施形態として、提案されるtransquant_bypass_default_flagが、コーディングされるCUのブロックサイズ、コーディングモード（すなわち、インター、イントラ

またはIBC)、またはブロックサイズおよびコーディングモードの組み合わせに依拠して、条件に応じて、変換四分木分割フラグおよび/またはコーディングされたブロックフラグのシグナリングをバイパスするために用いられる。例えば、複雑度および性能のトレードオフの観点において、ブロックが非イントラモードを用いてコーディングされ(すなわち、インターまたはIBCモードを用いてコーディングされ)、ブロックサイズが 8×8 または 16×16 である場合にのみ変換四分木分割を可能にすることが有益であり得る。

【0091】

1つの実施形態では、フラグ`transquant_bypass_default_flag`が1に等しいとき、以下が適用される。全てのイントラコーディングされたCUについて、および $64 \times 64 \sim 32 \times 32$ のブロックサイズを有する全てのインターコーディングおよびIBCコーディングされたCUについて、`split_transform_flag`および/またはコーディングされたブロックフラグ(`cbf_luma`、`cbf_cb`および`cbf_cr`)はシグナリングされず、それらは上記で論考したように対応するデフォルト値であると推測される。インターモードまたはIBCモードでコーディングされた 8×8 および 16×16 のCUの場合、更なる分割が許可され得る。`split_transform_flag`および/またはコーディングされたブロックフラグ(`cbf_luma`、`cbf_cb`および`cbf_cr`)は、現在のブロックが4つの四分木に更に区分されるか否かおよび/または1つのTU内の係数がオールゼロであるか否かをそれぞれ示すように、依然としてシグナリングされる。

【0092】

いくつかの実施形態では、2つのシンタックス要素`log2_intra_max_no_transform_split_coding_block_size_minus3`および`log2_inter_max_no_transform_split_coding_block_size_minus3`がSPSまたはPPSに加えられ、それぞれイントラおよびインター/IBCコーディングされたCUに変換四分木分割が適用される最大CUサイズを指定する。例えば、上記の条件1)および2)を用いて、`log2_intra_max_no_transform_split_coding_block_size_minus3`および`log2_inter_max_no_transform_split_coding_block_size_minus3`の値がそれぞれ $\log_2(64) - 3 = 3$ および $\log_2(16) - 3 = 1$ に設定される。表8は、2つの提案されるシンタックス要素を有する変更されたSPSスクリーンコンテンツコーディング拡張シンタックステーブルを示す。

【0093】

表8は、2つの追加のシンタックス要素を示すが、本開示の別の実施形態では、`log2_intra_max_no_transform_split_coding_block_size_minus3`の値はシグナリングされない場合がある。代わりに、値は、許可される最大CUサイズと常に同じであると推測され得る。許可される最大CUサイズは、最小CUサイズ(SPS内のシンタックス要素`log2_min_luma_coding_block_size_minus3`によって指定される)を、最大CUサイズと最小CUサイズとの差(SPS内のシンタックス要素`log2_diff_max_min_luma_coding_block_size`によって指定される)と加算することによって導出され得る。そのような実施形態において、CUがイントラモードでコーディングされ、無損失コーディングが適用されているとき、変換四分木分割は許容されない。

【0094】

【表 8】

1	<code>sps_scc_extensions() {</code>	Descriptor
2	<code> intra_block_copy_enabled_flag</code>	<code>u(1)</code>
3	<code> palette_mode_enabled_flag</code>	<code>u(1)</code>
4	<code> if(palette_mode_enabled_flag) {</code>	
5	<code> palette_max_size</code>	<code>ue(v)</code>
6	<code> palette_max_predictor_size</code>	<code>ue(v)</code>
7	<code> }</code>	
8	<code> adaptive_mv_resolution_enabled_flag</code>	<code>u(1)</code>
9	<code> intra_boundary_filtering_disabled_flag</code>	<code>u(1)</code>
10†	<code> log2_intra_max_no_transform_partition_coding_block_size_minus3</code>	<code>ue(v)</code>
11†	<code> log2_inter_max_no_transform_partition_coding_block_size_minus3</code>	<code>ue(v)</code>
12	<code>}</code>	

表8:シーケンスパラメータセットスクリーンコンテンツコーディングシンタックス

【0095】

`log2_intra_max_no_transform_partition_coding_block_size_minus3`の値に3をプラスした値は、コーディングユニットがイントラコーディングされる時、および`cu_transquant_bypass_flag`が1に等しいときに変換四分木分割が適用されるコーディングユニットの最大ブロックサイズを指定する。

【0096】

`log2_inter_max_no_transform_partition_coding_block_size_minus3`の値に3をプラスした値は、コーディングユニットがインターコーディングまたはイントラブロックコピーコーディングされる時、および`cu_transquant_bypass_flag`が1に等しいときに変換四分木分割が適用されるコーディングユニットの最大ブロックサイズを指定する。

【0097】

表9は、ブロックサイズおよび現在のCUのコーディングモードを条件とした`split_transform_flag`、`cbf_luma`、`cbf_cb`および`cbf_cr`の提案されるシグナリング制約を有する変更された`transform_tree()`シンタックステーブルを示す。この例示的な実施形態は、`split_transform_flag`および`cbf`のシグナリングを可能にするか否かを条件付けるためにCUコーディングモードおよびCUサイズの双方を用いるが、他の変更された条件が用いられてもよいことに留意されたい。例えば、コーディングモードおよびブロックサイズのいずれか（双方ではない）が用いられてもよい。更に、別個の（および異なる）条件が`split_transform_flag`シグナリングまたは`cbf`シグナリングに適用されてもよい。

【0098】

【表 9】

		Descriptor
1	transform_tree(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	
2†	if((!cu_transquant_bypass_flag ((CuPredMode[x0][y0] == MODE_INTRA && log2TrafoSize <= log2_intra_max_no_transform_partition_coding_unit_size_minus3+3) (CuPredMode[x0][y0] != MODE_INTRA && log2TrafoSize <= log2_inter_max_no_transform_partition_coding_unit_size_minus3+3))) && log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && (trafoDepth == 0)))	
3	split_transform_flag [x0][y0][trafoDepth]	ac(v)
4†	if((!cu_transquant_bypass_flag ((CuPredMode[x0][y0] == MODE_INTRA && log2TrafoSize <= log2_intra_max_no_transform_partition_coding_unit_size_minus3+3) (CuPredMode[x0][y0] != MODE_INTRA && log2TrafoSize <= log2_inter_max_no_transform_partition_coding_unit_size_minus3+3))) && log2TrafoSize > 2 && ChromaArrayType != 0) {	
5	if(trafoDepth == 0 cbf_cb[xBase][yBase][trafoDepth - 1]) {	
6	cbf_cb [x0][y0][trafoDepth]	ac(v)
7	if(ChromaArrayType == 2 && ! split_transform_flag[x0][y0][trafoDepth])	
8	cbf_cb [x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth]	
9	}	
10	if(trafoDepth == 0 cbf_cr[xBase][yBase][trafoDepth - 1]) {	
11	cbf_cr [x0][y0][trafoDepth]	ac(v)
12	if(ChromaArrayType == 2 && ! split_transform_flag[x0][y0][trafoDepth])	
13	cbf_cr [x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth]	
14	}	
15	}	
16	if(split_transform_flag[x0][y0][trafoDepth]) {	
17	x1 = x0 + (1 << (log2TrafoSize - 1))	
18	y1 = y0 + (1 << (log2TrafoSize - 1))	
19	transform_tree(x0, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 0)	
20	transform_tree(x1, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 1)	
21	transform_tree(x0, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 2)	
22	transform_tree(x1, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 3)	
23	} else {	
24†	if((!cu_transquant_bypass_flag ((CuPredMode[x0][y0] == MODE_INTRA && log2TrafoSize <= log2_intra_max_no_transform_partition_coding_unit_size_minus3+3) (CuPredMode[x0][y0] != MODE_INTRA && log2TrafoSize <= log2_inter_max_no_transform_partition_coding_unit_size_minus3+3))) && ((CuPredMode[x0][y0] == MODE_INTRA && intra_bc_flag[x0][y0] != 1) trafoDepth != 0 cbf_cb[x0][y0][trafoDepth] cbf_cr[x0][y0][trafoDepth] (ChromaArrayType == 2 && (cbf_cb[x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth] cbf_cr[x0][y0 + (1 << (log2TrafoSize - 1))][trafoDepth]))))	
25	cbf_luma [x0][y0][trafoDepth]	ae(v)
26	transform_unit(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx)	
27	}	
28	}	

表9: 変更された変換木シンタックス

【 0 0 9 9 】

別の実施形態では、無損失の場合に `split_transform_flag` のシグナリングをスキップするために、SPSにおけるシンタックス要素 `max_transform_hierarchy_depth_inter` および `max_transform_hierarchy_depth_intra` が 0 に設定され得る。この方法は、現在の `transform_tree()` シンタックスに対する低レベルの変更を必要とせ

ず、シーケンスレベル無損失コーディングの場合に `transform_skip_flag` のシグナリングをバイパスする。そのような実施形態において、`split_transform_flag` はシグナリングされず、代わりに、デフォルト値であると推測される。デフォルト値は、TU四分木分割が施行される場合を除いて、ほとんどの場合に0である（例えば、イントラコーディングについて、またはCUサイズが最も大きなTUサイズよりも大きいとき等に $N \times N$ の区分が用いられる）。このため、シーケンスレベル無損失コーディングが適用されるとき、`max_transform_hierarchy_depth_inter` および `max_transform_hierarchy_depth_intra` が適切に0に設定されることを必要とするビットストリーム制約を課すことによって、ブロックレベル変更を一切必要とすることなく `transform_split_flag` のシグナリングがバイパスされ得る。同様に、`transform_split_flag` のシグナリングは、シーケンスレベルの無損失コーディングの場合、最大変換サイズおよび最小変換サイズが同じでなくてはならないというビットストリーム制約を課すことによって、ブロックレベル変更を一切必要とすることなくバイパスされ得る。この制約は、シーケンスレベル無損失コーディングの場合に、SPSシンタックス `log2_diff_max_min_transform_block_size` が0に設定されることを必要とすることによって達成され得る。`max_transform_hierarchy_depth_inter`、`max_transform_hierarchy_depth_intra` および `log2_diff_max_min_transform_block_size` はSPS内に位置するので、この場合、`transquant_bypass_default_flag` もSPS内に入れることがより好ましい場合がある。

【0100】

開示の別の実施形態において、フラグ `transquant_bypass_default_flag` を追加することなく無損失コーディングのためのエンコードのみの方法が提案される。そのような一実施形態において、`transquant_bypass_default_flag` を用いる条件付きの項は、ダガー（†）を用いてマーキングされた表7および表9におけるシンタックス要素から省かれ得る。この実施形態は、SPS内のシンタックス要素 `max_transform_hierarchy_depth_inter` および `max_transform_hierarchy_depth_intra` または `transform_tree()` 内のシンタックス要素の値の変更を必要としない。符号化複雑度を低減するために、フラグ `split_transform_flag` は依然としてエンコードのみの方法のためにシグナリングされているが、表7または表9に記載されているような `split_transform_flag` のデフォルト値によって示される変換四分木分割のみがCUごとに試験される。

【0101】

いくつかの実施形態では、現在のCUについてフラグ `cu_transquant_bypass_flag` が1に等しいとき、エンコードは、ほとんどの場合について、非変形四分木区分のレート-歪み（R-D）性能のみを試験し、 $N \times N$ のPU区分がイントラコーディングされたCUに適用されるかまたは現在のCUサイズが閾値よりも大きいとき等のいくつかの特殊な場合について、ワンタイム変換四分木区分（one-time transform quadtree partition）のR-D性能を試験する。このようにして、このエンコードのみの方法は、ダガー（†）を用いてマーキングされたシンタックス要素における表7内のフラグ `split_transform_flag` に適用される条件と適合している。

【0102】

別の実施形態では、フラグ `cu_transquant_bypass_flag` が1に等しいとき、エンコードはCUが厳密に1つの予測ユニットを含むとき、 $64 \times 64 \sim 32 \times 32$ のブロックサイズを有する全てのイントラコーディングされたCUおよび全てのインター/IBCコーディングされたCUについて変換四分木区分のR-D性能のみを試験し、CUが少なくとも2つの予測ユニットを含むとき、 $64 \times 64 \sim 32 \times 32$ のブ

ブロックサイズを有する全てのイントラコーディングされたCUおよび全てのインター／IBCコーディングされたCUについてワнтаム変換四分木区分のR-D性能のみを試験し、ブロックサイズ16×16および8×8を有するインター／IBCコーディングされたCUについて、非変換四分木分割および更なる変換四分木分割の双方のR-D性能がエンコーダによって試験される。このため、このエンコーダのみの方法は、表9におけるダガー(†)を用いてマーキングされたフラグsplit_transform_flagに適用される条件に適合する。更に、フラグsplit_transform_flagは依然として上記のエンコーダのみの方法のためにビットストリームにおいてシグナリングされるが、1つのビットストリーム適合性制約が、ブロックサイズおよびブロックコーディングモードに依拠してシンタックス要素split_transform_flagの値がそのデフォルト値に設定されることを要求するために適用されてもよい。

【0103】

いくつかの実施形態では、変換四分木分割フラグのデフォルト値の決定は、少なくとも部分的に、関連コーディングユニットにおける複数の予測ユニットに基づく。1つのそのような実施形態において、デフォルト値の決定は、コーディングユニットがイントラコーディングされているか否か、コーディングユニットのサイズがサイズ閾値よりも大きいかな否か、およびコーディングユニットが厳密に1つの予測ユニットを含むかな否かを決定することを含む。コーディングユニットがイントラコーディングされておらず、コーディングユニットがサイズ閾値より大きく、コーディングユニットが厳密に1つの予測ユニットを含むという判断に回答して、変換四分木分割フラグのデフォルト値は、非変換四分木区分を示すように設定される。

【0104】

別のそのような実施形態では、変換四分木分割フラグのデフォルト値の決定は、関連コーディングユニットがイントラコーディングされているか否か、コーディングユニットのサイズがサイズ閾値よりも大きいかな否か、および第1のコーディングユニットが少なくとも2つの予測ユニットを含むかな否かを判断することを含む。コーディングユニットがイントラコーディングされておらず、コーディングユニットがサイズ閾値より大きく、コーディングユニットが少なくとも2つの予測ユニットを含むという判断に回答して、変換四分木分割フラグのデフォルト値は、ワнтаム変換四分木区分を示すように設定される。

【0105】

更なる実施形態では、変換四分木分割フラグのデフォルト値の決定は、関連コーディングユニットがイントラコーディングされているか否か、および第1のコーディングユニットが厳密に1つの予測ユニットを含むかな否かを判断することを含む。コーディングユニットがイントラコーディングされ、厳密に1つの予測ユニットを含むという判断に回答して、変換四分木分割フラグのデフォルト値は、非変換四分木区分を示すように設定される。

【0106】

別の実施形態では、変換四分木分割フラグのデフォルト値の決定は、関連コーディングユニットがイントラコーディングされているか否か、およびコーディングユニットが少なくとも2つの予測ユニットを含むかな否かを判断することを含む。コーディングユニットがイントラコーディングされておらず、かつコーディングユニットが少なくとも2つの予測ユニットを含むという判断に回答して、変換四分木分割フラグのデフォルト値が、ワнтаム変換四分木区分を示すように設定される。

【0107】

コーディングユニットが厳密に1つの予測ユニットを含むかまたは少なくとも2つの予測ユニットを含むかの判断は、コーディングユニットの区分モードを判断することによって行われ得る。例えば、2N×2Nの区分モードを用いるコーディングユニットは厳密に1つの予測ユニットを含む一方、例えば、2N×N、N×2N、2N×nU、2N×nD、nL×2N、nR×2NまたはN×Nの区分モードを用いるコーディングユニットは、少なくとも2つの予測ユニットを含む。

【0108】

イントラブロックコピーモードのための変換木分割

変換四分木分割の最大深度は、符号化および復号の複雑度と密接に関連している。符号化効率と計算複雑度との間の柔軟なトレードオフを与えるために、HEVCおよびその拡張は、SPSにおいてシンタックス要素を用いて、TUサイズおよびTU分割深度を指定する。値 $\log_2_min_luma_transform_block_size_minus2$ および $\log_2_diff_max_min_luma_transform_block_size$ は、ビデオシーケンスをコーディングするのに用いられるTUサイズの組を示し、 $max_transform_hierarchy_depth_inter$ および $max_transform_hierarchy_depth_intra$ は、それぞれ、イントラおよびインターコーディングされたCUのための最大分割深度を示す。ある特定の条件において、変換四分木分割は適用されない場合がある。例えば、 $max_transform_hierarchy_depth_intra/inter$ が0に設定される場合、変換四分木分割は現在のイントラ/インターコーディングされたCUに適用されない。

【0109】

HEVC、その範囲拡張、およびSCC草案において、変換四分木分割が無効にされているとき、1つの暗黙的なTU区分方法が、損失ありコーディングおよび無損失コーディングの双方に適用され、それによって、 $split_transform_flag$ の値は、複数の予測ユニット(PU)に区分され、インターモードにおいてコーディングされたCUについて、常に1である(すなわち、変換四分木分割が適用される)と推測される。これは、CU内のPUの異なる動きベクトルが人工的な高周波数情報を引き起こし、近傍のPU間の境界にわたって一貫性のない残差をもたらすことに起因している。この場合、CUをより小さなTUに分割することは、CUのサイズと同じ大きさのTUサイズを用いるよりも良好なコーディング効率を提供することができる。

【0110】

HEVCスクリーンコンテンツコーディング拡張[0057]の研究草案において、上記の暗黙的なTU区分は、IBCモードにおいてコーディングされたCUに適用されない。より詳細には、変換四分木分割が無効にされているとき、 $split_transform_flag$ の値は、全てのIBCコーディングされたCUについて常に0である(すなわち、変換ユニットサイズがCUのものと同じに設定される)と推測される。IBCモードとインターモードとの間の固有の類似性を与られると、IBCコーディングされたCUの残差は、インターコーディングされたCUのものと類似した特性を呈することができる。したがって、変換コーディングの効率を更に改善するために、IBCコーディングされたCUに暗黙的なTU区分を適用する(すなわち、 $split_transform_flag$ の値が1であると推測する)ことも有利であり得る。本明細書において説明される一実施形態において、変換四分木区分が無効にされているとき、インターモードに適用される同じ暗黙的なTU区分方法も、損失ありコーディングおよび無損失コーディングの双方におけるIBCコーディングされたCUに用いられる。換言すれば、2つ以上のPU区分(例えば、 $2N \times N$ 、 $N \times 2N$ および $N \times N$)がIBCモードでコーディングされた現在のCUに存在するとき、 $split_transform_flag$ の値は1である(すなわち、変換四分木が分割される)と推測される。

【0111】

$split_transform_flag$ の値の導出プロセスは、[0057]のセクション7.4.9.9において指定されている。本明細書に開示する例示的な実施形態におけるIBCモードのための暗黙的なTU分割の有効化に伴い、 $split_transform_flag$ のセマンティクスは以下のように動作する。

【0112】

アレイ $split_transform_flag[x0][y0][trafoDepth]$ は、ブロックが、変換コーディングの目的で2分の1の水平サイズおよび2分の1の垂直サイズを有する4つのブロックに分割されるか否かを指定する。アレイインデッ

クス x_0 、 y_0 は、ピクチャの左上ルーマサンプルに対する検討中のブロックの左上ルーマサンプルのロケーション(x_0 、 y_0)を指定する。アレイインデックス $trafoDepth$ は、変換コーディングの目的で、ブロックへのコーディングブロックの現在の細分化レベルを指定する。コーディングブロックに対応するブロックについて、 $trafoDepth$ の値は0に等しい。

【0113】

変数 $interSplitFlag$ は、以下のように導出される。 $interSplitFlag$ は、以下の条件、すなわち、 $max_transform_hierarchy_depth_inter$ が0に等しくかつ $CuPredMode[x_0][y_0]$ が $MODE_INTER$ に等しい、または $intra_bc_flag[x_0][y_0]$ が1に等しくかつ $PartMode$ が $PART_2Nx2N$ に等しくなく、かつ $trafoDepth$ が0に等しい、のうちの1つまたは複数が適用されるとき、1に等しく設定される。そうでない場合、 $interSplitFlag$ が0に等しく設定される。

【0114】

$split_transform_flag[x_0][y_0][trafoDepth]$ が存在しないとき、その値は以下のように推測される。以下の条件、すなわち、 $log2TrafoSize$ が $MaxTbLog2SizeY$ よりも大きい、 $IntraSplitFlag$ が1に等しくかつ $trafoDepth$ が0に等しい、または $interSplitFlag$ が1に等しい、のうちの1つまたは複数が真である場合、 $split_transform_flag[x_0][y_0][trafoDepth]$ の値が1に等しいと推測される。そうでない場合、 $split_transform_flag[x_0][y_0][trafoDepth]$ の値は0に等しいと推測される。

【0115】

実施形態

例示的な実施形態において、スライスセグメントヘッダおよび複数のコーディングユニットを含むビデオスライスをコーディングする方法が提供される。本方法は、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているか否かを示す、スライスセグメントヘッダにおけるバイパスフラグを生成することを含む。

【0116】

いくつかのそのような実施形態において、バイパスフラグは、 $transquant_bypass_default_flag$ である。本方法は、 $transquant_bypass_enabled_flag$ を含むピクチャパラメータセット(PPS)を生成することを含むことができ、ここで、スライスは、ピクチャパラメータセットを参照し、 $transquant_bypass_enabled_flag$ は、 $transquant_bypass_default_flag$ が1に設定されているとき、ゼロに設定される。

【0117】

いくつかの実施形態では、バイパスフラグは、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているわけではないことを示し、本方法は、スライスセグメントヘッダにおいて、損失ありコーディングに関係付けられたシンタックス要素を生成することを含む。バイパスフラグは、損失ありコーディングに関係付けられたシンタックス要素の前に位置決めされ得る。

【0118】

いくつかの実施形態において、バイパスフラグは、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているわけではないことを示し、本方法は、スライスセグメントヘッダにおいて、量子化、変換およびループ内フィルタリングプロセスに関係付けられたシンタックス要素を生成することを更に含む。バイパスフラグは、量子化、変換およびループ内フィルタリングプロセスに関係付けられたシンタックス要素の前に位置決めされ得る。

【0119】

いくつかの実施形態では、バイパスフラグは、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされるわけではないことを示し、本方法は、スライスセグメントヘッダにおいて、`slice_sao_luma_flag`、`slice_sao_chroma_flag`、`slice_qp_delta`、`slice_cb_qp_offset`、`slice_cr_qp_offset`、`slice_chroma_qp_adjustment_enabled_flag`、`deblocking_filter_override_flag`、`slice_deblocking_filter_disabled_flag`、`slice_beta_offset_div2`、`slice_tc_offset_div2`および`slice_loop_filter_across_slices_enabled_flag`からなる群から選択された1または複数のシンタックス要素を生成することを更に含む。バイパスフラグは、`slice_sao_luma_flag`、`slice_sao_chroma_flag`、`slice_qp_delta`、`slice_cb_qp_offset`、`slice_cr_qp_offset`、`slice_chroma_qp_adjustment_enabled_flag`、`deblocking_filter_override_flag`、`slice_deblocking_filter_disabled_flag`、`slice_beta_offset_div2`、`slice_tc_offset_div2`および`slice_loop_filter_across_slices_enabled_flag`からなる群から選択された1または複数のシンタックス要素の前に位置決めされ得る。

【0120】

いくつかの実施形態では、バイパスフラグは、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされていることを示し、本方法は、スライスセグメントヘッダから、損失ありコーディングに関係付けられたシンタックス要素を除外することを更に含む。

【0121】

いくつかの実施形態では、バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされていることを示す場合、本方法は、スライスセグメントヘッダから、量子化、変換およびループ内フィルタリングプロセスに関係付けられたシンタックス要素を除外することを更に伴う。

【0122】

いくつかの実施形態では、バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされていることを示す場合、本方法は、スライスセグメントヘッダから、`slice_sao_luma_flag`、`slice_sao_chroma_flag`、`slice_qp_delta`、`slice_cb_qp_offset`、`slice_cr_qp_offset`、`slice_chroma_qp_adjustment_enabled_flag`、`deblocking_filter_override_flag`、`slice_deblocking_filter_disabled_flag`、`slice_beta_offset_div2`、`slice_tc_offset_div2`および`slice_loop_filter_across_slices_enabled_flag`からなる群から選択された1または複数のシンタックス要素を除外することを更に伴う。

【0123】

いくつかの実施形態では、バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされるわけではないことを示す場合、本方法は、スライス内のコーディングユニットごとに`cu_transquant_bypass_flag`をシグナリングすることを更に含む。

【0124】

いくつかの実施形態では、バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされていることを示す場合、本方法は、ス

ライス内の各コーディングユニットから `cu__transquant__bypass__flag` を除外することを更に伴う。

【0125】

例示的な実施形態では、ピクチャパラメータセットおよびピクチャパラメータセット (PPS) を参照する少なくとも1つのスライスを含むビデオをコーディングする方法が提供される。ここで、スライスは、スライスセグメントヘッダおよび複数のコーディングユニットを含む。この実施形態において、本方法は、ピクチャパラメータセットにおいて、ピクチャパラメータセットを参照するスライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているか否かを示すバイパスフラグを生成することを含む。

【0126】

いくつかのそのような実施形態では、バイパスフラグは、`transquant__bypass__default__flag` である。ピクチャパラメータセット (PPS) は、`transquant__bypass__enabled__flag` を含み、`transquant__bypass__enabled__flag` は、`transquant__bypass__default__flag` が1に設定されるとき、0に設定される。

【0127】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているわけではないことを示すいくつかの実施形態では、本方法は、スライスセグメントヘッダにおいて、損失ありコーディングに関係付けられたシンタックス要素を生成することを更に含む。

【0128】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているわけではないことを示すいくつかの実施形態において、本方法は、スライスセグメントヘッダにおいて、量子化、変換およびループ内フィルタリングプロセスに関係付けられたシンタックス要素を生成することを更に含む。

【0129】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているわけではないことを示すいくつかの実施形態において、本方法は、スライスセグメントヘッダにおいて、`slice__sao__luma__flag`、`slice__sao__chroma__flag`、`slice__qp__delta`、`slice__cb__qp__offset`、`slice__cr__qp__offset`、`slice__chroma__qp__adjustment__enabled__flag`、`deblocking__filter__override__flag`、`slice__deblocking__filter__disabled__flag`、`slice__beta__offset__div2`、`slice__tc__offset__div2` および `slice__loop__filter__across__slices__enabled__flag` からなる群から選択された1または複数のシンタックス要素を生成することを更に含む。

【0130】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされていることを示すいくつかの実施形態において、本方法は、スライスセグメントヘッダから、損失ありコーディングに関係するシンタックス要素を除外することを更に伴う。

【0131】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされていることを示すいくつかの実施形態において、本方法は、スライスセグメントヘッダから、量子化、変換およびループ内フィルタリングプロセスに関係付けられたシンタックス要素を除外することを更に伴う。

【0132】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを

用いてコーディングされていることを示すいくつかの実施形態において、本方法は、スライスセグメントヘッダから、`slice_sao_luma_flag`、`slice_sao_chroma_flag`、`slice_qp_delta`、`slice_cb_qp_offset`、`slice_cr_qp_offset`、`slice_chroma_qp_adjustment_enabled_flag`、`deblocking_filter_override_flag`、`slice_deblocking_filter_disabled_flag`、`slice_beta_offset_div2`、`slice_tc_offset_div2`および`slice_loop_filter_across_slices_enabled_flag`からなる群から選択された1または複数のシンタックス要素を除外することを伴う。

【0133】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているわけではないことを示すいくつかの実施形態において、本方法は、スライス内のコーディングユニットごとに、`cu_transquant_bypass_flag`をシグナリングすることを更に含む。

【0134】

バイパスフラグが、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているわけではないことを示すいくつかの実施形態において、本方法は、スライス内の各コーディングユニットから`cu_transquant_bypass_flag`を除外することを更に伴う。

【0135】

例示的な実施形態において、ビデオをコーディングする方法が提供され、ここで、ビデオは、高水準シンタックス構造および高水準シンタックス構造を参照する少なくとも1つのスライスを含み、スライスは複数のコーディングユニットを含む。本方法は、スライス内の全てのコーディングユニットが無損失コーディングを用いてコーディングされているか否かを示す高水準シンタックス構造においてバイパスフラグを生成することを含む。各それぞれのコーディングユニットについて、四分木分割フラグを生成すべきか否かの判断が行われる。この判断は、少なくとも部分的に、それぞれのコーディングユニットのブロックサイズおよびコーディングモードからなる群から選択されたパラメータに基づく。四分木分割フラグは、四分木分割フラグを生成する判断を行った後にのみ、それぞれのコーディングユニットについて生成される。

【0136】

そのような実施形態において、高水準シンタックス構造は、ピクチャパラメータセット、セグメントスライスヘッダまたは`split_transform_flag`であり得る。いくつかのそのような実施形態において、四分木分割フラグは、ブロックが非イントラモードを用いてコーディングされており、かつブロックサイズが 8×8 または 16×16 であるときにのみ生成される。四分木分割フラグは、ブロックサイズが $64 \times 64 \sim 32 \times 32$ である場合、生成されない。

【0137】

いくつかの実施形態は、最大非変換四分木分割ブロックコーディングサイズを示す高水準シンタックス要素を生成することを更に伴う。いくつかの実施形態では、最大非変換四分木分割ブロックコーディングサイズを示す高水準シンタックス要素がシーケンスパラメータセット (SPS) において生成される。いくつかの実施形態では、最大非変換四分木分割ブロックコーディングサイズを示す高水準シンタックス要素は、ピクチャパラメータセット (PPS) において生成される。

【0138】

いくつかの実施形態では、高水準シンタックス要素は、最大非変換四分木分割ブロックコーディングサイズが、非イントラモードにおいてコーディングされたブロックのための最大非変換四分木分割ブロックコーディングサイズを示すことを示すのに用いられる。

【0139】

いくつかの実施形態では、ブロックが非イントラモードを用いてコーディングされており、かつ非イントラモードにおいてコーディングされたブロックについて、ブロックサイズが最大非変換四分木分割ブロックコーディングサイズよりも大きくない場合に四分木分割フラグが生成される。

【0140】

いくつかの実施形態において、最大非変換四分木分割ブロックコーディングサイズを示す高水準シンタックス要素が、イントラモードにおいてコーディングされたブロックについて、最大非変換分割ブロックコーディングサイズを示す。

【0141】

いくつかの実施形態において、四分木分割フラグは、ブロックがイントラモードを用いてコーディングされており、かつイントラモードにおいてコーディングされたブロックについて、ブロックサイズが最大非変換四分木分割ブロックコーディングサイズ以下である場合に生成される。

【0142】

例示的な実施形態において、ビデオをコーディングする方法が提供される。ビデオは、高水準シンタックス構造および高水準シンタックス構造を参照する少なくとも1つのスライスを含み、スライスは、複数のコーディングユニットを含む。本方法は、高水準シンタックス構造において、スライス内のコーディングユニットの全てが無損失コーディングでコーディングされているか否かを示すバイパスフラグを生成することを含む。各それぞれのコーディングユニットについて、コーディングされたブロックフラグを生成するか否かの判断が行われる。判断は、少なくとも部分的に、それぞれのコーディングユニットのブロックサイズおよびコーディングモードからなる群から選択されたパラメータに基づく。コーディングされたブロックフラグは、コーディングされたブロックフラグを生成する判断を行った後にのみ、それぞれのコーディングユニットについて生成される。

【0143】

そのような実施形態において、高水準シンタックス構造は、ピクチャパラメータセットまたはセグメントスライスヘッダであり得る。コーディングされたブロックフラグは、`cbf_luma_flag`、`cbf_cbf_flag`または`cbf_crf_flag`のうちの1または複数であり得る。いくつかの実施形態では、コーディングされたブロックフラグは、ブロックが非イントラモードを用いてコーディングされており、かつブロックサイズが 8×8 または 16×16 であるときにのみ生成されるのに対し、コーディングされたブロックフラグは、ブロックがイントラモードを用いてコーディングされるかまたはブロックサイズが $64 \times 64 \sim 32 \times 32$ である場合、生成されない。

【0144】

いくつかの実施形態では、最大非変換四分木分割ブロックコーディングサイズを示す高水準シンタックス要素が生成される。最大非変換四分木分割ブロックコーディングサイズを示す高水準シンタックス要素は、シーケンスパラメータセット (SPS) またはピクチャパラメータセット (PPS) において生成され得る。

【0145】

いくつかの実施形態では、最大非変換四分木分割ブロックコーディングサイズを示す高水準シンタックス要素は、非イントラモードにおいてコーディングされたブロックのための最大非変換四分木分割ブロックコーディングサイズを示す。ブロックが非イントラモードを用いてコーディングされており、かつイントラモードにおいてコーディングされたブロックについて、ブロックサイズが最大非変換分割ブロックコーディングサイズ以下の場合に、コーディングされたフラグが生成される。

【0146】

いくつかの実施形態において、最大非変換分割ブロックコーディングサイズを示す高水準シンタックス要素が、イントラモードにおいてコーディングされたブロックのための最大非変換分割ブロックコーディングサイズを示す。

【0147】

いくつかの実施形態では、コーディングされたブロックフラグは、ブロックがイントラモードを用いてコーディングされており、かつイントラモードにおいてコーディングされたブロックについて、ブロックサイズが最大非変換分割ブロックコーディングサイズ以下である場合に生成される。

【0148】

例示的な実施形態において、ビデオをコーディングする方法が提供される。ここで、ビデオは高水準シンタックス構造および高水準シンタックス構造を参照する少なくとも1つのスライスを含み、スライスは、複数のコーディングユニットを含む。本方法は、フラグ `cu__transquant__bypass__flag` がコーディングユニットについて1に等しいと判断することを含む。判断は、コーディングユニットサイズおよびコーディングユニットのコーディングモードのうちの少なくとも一方に基づいて、ワンタイム変換四分木区分のレート-歪み性能を試験するために行われる。判断後、ワンタイム変換四分木区分のレート-歪み性能が試験される。

【0149】

いくつかのそのような実施形態では、本方法は、更なるコーディングユニットについて、ワンタイム変換四分木区分のレート-歪み性能を試験しないことの判断を行うことを更に含む。レート-歪み性能の試験は、追加のコーディングユニットのための非変換四分木区分についてのみ実行される。

【0150】

例示的な実施形態において、ビデオをコーディングする方法が提供される。ここで、ビデオは、分割変換フラグを含む変換木シンタックスを含み、複数のコーディングユニットを更に含む。本方法は、それぞれのコーディングユニットについて、コーディングユニットがイントラブロックコピーモードでコーディングされているか否か、および2つ以上の予測ユニット区分がコーディングユニット内に存在するか否かを判断することを含む。コーディングユニットがイントラブロックコピーモードでコーディングされており、かつ2つ以上の予測ユニット区分がコーディングユニット内に存在するという判断にตอบสนองして、分割変換フラグの値が1であると推測される。

【0151】

例示的な方法において、変換および量子化プロセスがバイパスされることを示すコーディングユニットフラグのためのデフォルト値を含む高水準シンタックス構造が生成される。

【0152】

いくつかのそのような実施形態では、高水準シンタックス構造を参照するスライス内の全てのコーディングユニットの `cu__transquant__bypass__flag` のデフォルト値を示すデフォルト値、`transquant__bypass__default__flag` が1に設定される。高水準シンタックス構造は、ピクチャパラメータセット (PPS)、シーケンスパラメータセット (SPS)、ビデオパラメータセット (VPS) またはスライスセグメントヘッダのうちの少なくとも1つであり得る。

【0153】

いくつかの実施形態では、本方法は、`transquant__bypass__enabled__flag` が0に等しいPPSを含むビットストリームを生成することを含む。いくつかの実施形態では、本方法は、ビットストリームを生成することを含み、コーディングユニットパラメータは `cu__transquant__bypass__flag` を含まない。

【0154】

例示的な実施形態では、特定の高水準シンタックス構造における高水準シンタックス要素を介して、その特定の高水準シンタックス構造を参照する全てのCUについて、変換、変換スキップ、量子化およびループ内フィルタリングプロセスをバイパスするようにデコードにシグナリングする方法が提供される。

【0155】

別の例示的な実施形態では、逆量子化、逆変換およびループ内フィルタリングプロセスのうちの任意の1または複数に関係付けられた複数のPPSシンタックス要素の存在を特定するための高水準シンタックス要素を受信および処理するようにデコーダを動作させる方法が提供される。

【0156】

いくつかのそのような方法において、高水準シンタックス要素は、デフォルトフラグ値(`transquant__bypass__default__flag`)である。いくつかのそのような方法において、高水準シンタックス要素は、`transform__skip__enabled__flag`要素の存在を特定するために用いられる。

【0157】

いくつかの実施形態では、本方法は、推測により`transquant__bypass__enabled__flag`が0であると判断し、これに応じて、CUレベルにおいて`cu__transquant__bypass__flag`のシグナリングを削除またはスキップすることを含む。いくつかの実施形態では、デフォルトフラグは、PPS拡張パラメータセットまたはSPS拡張パラメータセットのうちの少なくとも1つに含まれる。

【0158】

いくつかの実施形態において、デフォルトフラグの存在は、`transquant__bypass__enabled__flag`の値を条件とする。いくつかの実施形態において、デフォルトフラグは`transquant__bypass__default__flag`である。

【0159】

いくつかの実施形態において、本方法は、無損失コーディングモードにおいて使用されないシンタックス要素のシグナリングされた値が適切に設定されたことを示す追加の適合性フラグのシグナリングを受信することを更に含む。

【0160】

いくつかの実施形態では、`cu__qp__delta__enabled__flag`、`pps__loop__filter__across__slices__enabled__flag`、`deblocking__filter__control__present__flag`、`loop__filter__across__tiles__enabled__flag`、`pps__scaling__list__data__present__flag`を含むシンタックス要素は、フラグ`transquant__bypass__default__flag`が1に設定される場合に0に設定される。

【0161】

例示的な実施形態では、無損失コーディングが用いられることを示す高水準シグナリング無損失コーディングシンタックス要素を含むビデオデータビットストリームを受信する方法が提供される。

【0162】

いくつかのそのような方法では、高水準シグナリングシンタックスは、ピクチャパラメータセット(PPS)、シーケンスパラメータセット(PPS)、ビデオパラメータセット(VPS)またはスライスセグメントヘッダのうちの1つである。いくつかのそのような方法では、無損失コーディングシンタックス要素は、量子化、変形、変形スキップ、変形スキップ回転、およびループ内フィルタリングプロセスに関係付けられた1または複数のSPSシンタックス要素を表すための条件として用いられる。

【0163】

例示的な実施形態において、スライスセグメントヘッダが受信され、デフォルトフラグは、変換、量子化およびループ内フィルタリング処理ブロックのために用いられるスライスセグメントシンタックス要素の特定を条件付けるためのものである。いくつかのそのような方法において、デフォルトフラグは`transquant__bypass__default__flag`である。

【0164】

例示的な実施形態において、高水準無損失コーディングインジケーションを受信し、これに応じて複数の処理ブロックをシャットダウンする方法がビデオデコードにおいて実行される。いくつかのそのような実施形態では、高水準無損失コーディングインジケーションは、PPS、SPS、VPSまたはスライスヘッダのうちの1つのパラメータ要素である。いくつかの実施形態では、複数の処理ブロックは、以下のハードウェアブロック、すなわち、逆量子化、逆変換、デブロッキングフィルタ、SAOのうちの任意のもののうちの1または複数を含む。

【0165】

上記の技法のうちの任意のものを用いて符号化されたビデオが、任意の適切な有線もしくは無線伝送媒体を用いて送信されることが可能であり、かつ/または任意の適切な非一時的デジタルストレージ媒体上に記録されることが可能である。

【0166】

特徴および要素が、上記において特定の組み合わせで説明されたが、当業者であれば、各特徴または要素は、単独で、または他の特徴と要素との任意の組み合わせで用いられ得ることを理解するであろう。更に、本明細書において説明される方法は、コンピュータまたはプロセッサによって実行するためのコンピュータ可読媒体に組み込まれたコンピュータプログラム、ソフトウェアまたはファームウェアにおいて実施され得る。コンピュータ可読媒体の例は、(有線または無線接続を介して送信される)電子信号およびコンピュータ可読記憶媒体を含む。コンピュータ可読記憶媒体の例は、限定ではないが、リードオンリーメモリ(ROM)、ランダムアクセスメモリ(RAM)、レジスタ、キャッシュメモリ、半導体メモリデバイス、内部ハードディスクおよび取り外し可能ディスク等の磁気媒体、磁気光媒体、ならびにCD-ROMディスクおよびデジタル多用途ディスク(DVD)等の光媒体を含む。ソフトウェアに関連付けられたプロセッサは、WTRU、UE、端末、基地局、RNCまたは任意のホストコンピュータにおいて用いるための無線周波数送受信機を実装するように用いられ得る。