(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2025년06월18일
(11) 등록번호 10-2822841
(24) 등록일자 2025년06월16일

(51) 국제특허분류(Int. Cl.)
H04N 19/70 (2014.01) *H04N 19/105* (2014.01)
H04N 19/172 (2014.01) *H04N 19/174* (2014.01)
H04N 19/30 (2014.01) *H04N 19/44* (2014.01)
H04N 19/463 (2014.01) *H04N 19/61* (2014.01)
H04N 19/82 (2014.01)
(52) CPC특허분류
H04N 19/70 (2015.01)
H04N 19/105 (2015.01)
(21) 출원번호 10-2022-7015181
(22) 출원일자(국제) 2020년10월06일
심사청구일자 2022년05월04일
(85) 번역문제출일자 2022년05월04일
(65) 공개번호 10-2022-0070039
(43) 공개일자 2022년05월27일
(86) 국제출원번호 PCT/US2020/054454
(87) 국제공개번호 WO 2021/022273
국제공개일자 2021년02월04일
(30) 우선권주장
62/911,808 2019년10월07일 미국(US)
(56) 선행기술조사문헌
B. Bross, et. al, "Versatile Video Coding (Draft 5)", Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 14th Meeting: Geneva, CH, 19-27 Mar. 2019, JVET-N1001.*
비특허*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
후아웨이 테크놀러지 컴퍼니 리미티드
중국 518129 광동성 센젠 롱강 디스트릭트 반티안 후아웨이 어드미니스트레이션 빌딩
(72) 발명자
왕 예-쿠이
미국 92130 캘리포니아주 샌디에고 선로즈 크레스트 웨이 6264
(74) 대리인
유미특허법인

전체 청구항 수 : 총 16 항 심사관 : 최재륜

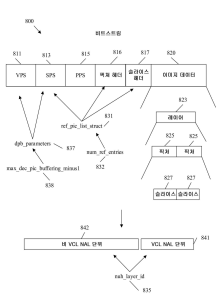
(54) 발명의 명칭 **DPB 사이즈 기반의 참조 픽쳐 엔트리 제약**

(57) 요 약

비디오 코딩 메커니즘이 개시된다. 메커니즘은, 제로 내지 최대 디코딩된 픽쳐 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 참조 엔트리의 개수(num_ref_entries)를 포함하는 참조 픽쳐 목록 신택스 구조(ref_pic_list_struct()) 및 현재 픽쳐를 포함하는 비트스트림을 수신하는 것을 포함한다. 현재 픽쳐는 디코딩된

(뒷면에 계속)

대 표 도 - 도8



픽쳐를 생성하도록 ref_pic_list_struct()에 기초하여 디코딩된다. 디코딩된 픽쳐는 디코딩된 비디오 시퀀스의 일부로서 디스플레이를 위해 포워딩된다.

(52) CPC특허분류
H04N 19/172 (2015.01)
H04N 19/174 (2015.01)
H04N 19/30 (2015.01)
H04N 19/44 (2015.01)
H04N 19/463 (2015.01)
H04N 19/61 (2015.01)
H04N 19/82 (2015.01)

명 세 서

청구범위

청구항 1

디코더에 의해 구현되는 방법으로서,

상기 디코더에 의해, 제로 내지 최대 디코딩된 픽처 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 참조 엔트리의 개수(num_ref_entries)를 포함하는 참조 픽처 목록 신택스 구조(ref_pic_list_struct()) 및 현재 픽처를 포함하는 비트스트림을 수신하는 단계 - 상기 num_ref_entries는 상기 ref_pic_list_struct()에서의 엔트리의 개수를 명시하고, 상기 비트스트림은 디코딩된 픽처 버퍼 파라미터 신택스 구조(dpb_parameters())를 포함하고, 상기 dpb_parameters()는 최대 디코딩된 픽처 버퍼 마이너스 1(max_dec_pic_buffering_minus1)을 포함하고, 상기 최대 디코딩된 픽처 버퍼 사이즈는 상기 max_dec_pic_buffering_minus1에 대응하고, 상기 dpb_parameters()가 비디오 파라미터 세트(VPS)에 포함되는 경우, 상기 dpb_parameters()가 적용되는 출력 레이어 세트(OLS)는 상기 VPS에 의해 명시되고, 상기 dpb_parameters()가 시퀀스 파라미터 세트(SPS)에 포함되는 경우, 상기 dpb_parameters()는 상기 SPS를 참조하는 독립 레이어만을 포함하는 OLS에 적용됨 -; 및

상기 디코더에 의해, 상기 ref_pic_list_struct() 및 상기 dpb_parameters()에 기초하여 상기 현재 픽처를 디코딩하여 디코딩된 픽처를 생성하는 단계

를 포함하는 디코더에 의해 구현되는 방법.

청구항 2

제1항에 있어서,

상기 ref_pic_list_struct()는, ref_pic_list_struct(listIdx, rplsIdx)로서 나타내어지는 목록 인덱스(listIdx) 및 참조 픽처 목록 구조 인덱스(rplsIdx)에 따라 참조되고, 상기 num_ref_entries는 num_ref_entries [listIdx][rplsIdx]로 나타내어지고, 상기 num_ref_entries [listIdx][rplsIdx]는 상기 ref_pic_list_struct(listIdx, rplsIdx)에서의 엔트리의 개수를 명시하는, 디코더에 의해 구현되는 방법.

청구항 3

제1항에 있어서,

상기 ref_pic_list_struct에서 참조되는 참조 픽처의 세트(setOfRefPics)에서의 픽처의 개수는 상기 최대 디코딩된 픽처 버퍼 사이즈 마이너스 1 이하가 되도록 제한되는, 디코더에 의해 구현되는 방법.

청구항 4

제3항에 있어서,

상기 ref_pic_list_struct()는 참조 픽처 목록 0(RefPicList[0]) 및 참조 픽처 목록 1(RefPicList[1])을 포함하고, 상기 setOfRefPics는 상기 현재 픽처와 동일한 네트워크 추상화 레이어(network abstraction layer; NAL) 단위 헤더 레이어 식별자(nuh_layer_id)를 갖는 RefPicList[0] 내의 모든 엔트리 및 상기 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 세트 고유의 픽처인, 디코더에 의해 구현되는 방법.

청구항 5

제3항에 있어서,

상기 setOfRefPics는 각각의 픽처의 모든 슬라이스에 대해 동일한 세트인, 디코더에 의해 구현되는 방법.

청구항 6

인코더에 의해 구현되는 방법으로서,

상기 인코더에 의해, 참조 픽처에 기초하여 현재 픽처를 비트스트림에 인코딩하는 단계;

상기 인코더에 의해, 상기 참조 픽처를 나타내며 제로 내지 최대 디코딩된 픽처 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 참조 엔트리의 개수(num_ref_entries)를 포함하는 참조 픽처 목록 신택스 구조(ref_pic_list_struct())를 상기 비트스트림에 인코딩하는 단계 - 상기 num_ref_entries는 상기 ref_pic_list_struct()에서의 엔트리의 개수를 명시하고, 상기 비트스트림은 디코딩된 픽처 버퍼 파라미터 신택스 구조(dpb_parameters())를 포함하고, 상기 dpb_parameters()는 최대 디코딩된 픽처 버퍼 마이너스 1(max_dec_pic_buffering_minus1)을 포함하고, 상기 최대 디코딩된 픽처 버퍼 사이즈는 상기 max_dec_pic_buffering_minus1에 대응하고, 상기 dpb_parameters()가 비디오 파라미터 세트(VPS)에 포함되는 경우, 상기 dpb_parameters()가 적용되는 출력 레이어 세트(OLS)는 상기 VPS에 의해 명시되고, 상기 dpb_parameters()가 시퀀스 파라미터 세트(SPS)에 포함되는 경우, 상기 dpb_parameters()는 상기 SPS를 참조하는 독립 레이어만을 포함하는 OLS에 적용됨 -; 및

상기 인코더에 의해, 디코더를 향한 통신을 위해 상기 비트스트림을 저장하는 단계

를 포함하는 인코더에 의해 구현되는 방법.

청구항 7

제6항에 있어서,

상기 ref_pic_list_struct()는, ref_pic_list_struct(listIdx, rplsIdx)로서 나타내어지는 목록 인덱스(listIdx) 및 참조 픽처 목록 구조 인덱스(rplsIdx)에 따라 참조되고, 상기 num_ref_entries는 num_ref_entries [listIdx][rplsIdx]로 나타내어지고, 상기 num_ref_entries [listIdx][rplsIdx]는 상기 ref_pic_list_struct(listIdx, rplsIdx)에서의 엔트리의 개수를 명시하는, 인코더에 의해 구현되는 방법.

청구항 8

제6항에 있어서,

상기 ref_pic_list_struct에서 참조되는 참조 픽처의 세트(setOfRefPics)에서의 픽처의 개수는 상기 최대 디코딩된 픽처 버퍼 사이즈 마이너스 1 이하가 되도록 제한되는, 인코더에 의해 구현되는 방법.

청구항 9

제8항에 있어서,

상기 ref_pic_list_struct()는 참조 픽처 목록 0(RefPicList[0]) 및 참조 픽처 목록 1(RefPicList[1])을 포함하고, 상기 setOfRefPics는 상기 현재 픽처와 동일한 네트워크 추상화 레이어(NAL) 단위 헤더 레이어 식별자(nuh_layer_id)를 갖는 RefPicList[0] 내의 모든 엔트리 및 상기 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 세트 고유의 픽처인, 인코더에 의해 구현되는 방법.

청구항 10

제8항에 있어서,

상기 setOfRefPics는 각각의 픽처의 모든 슬라이스에 대해 동일한 세트인, 인코더에 의해 구현되는 방법.

청구항 11

비디오 코딩 디바이스로서,

프로세서, 상기 프로세서에 커플링되는 수신기, 상기 프로세서에 커플링되는 메모리, 및 상기 프로세서에 커플링되는 송신기를 포함하고, 상기 프로세서, 수신기, 메모리, 및 송신기는 제1항 내지 제10항 중 어느 한 항의 상기 방법을 수행하도록 구성되는, 비디오 코딩 디바이스.

청구항 12

비디오 코딩 디바이스에 의한 사용을 위한 컴퓨터 프로그램 제품을 포함하는 비일시적 컴퓨터 판독 가능 매체로서,

상기 컴퓨터 프로그램 제품은, 프로세서에 의해 실행될 때 상기 비디오 코딩 디바이스로 하여금 제1항 내지 제10항 중 어느 한 항의 상기 방법을 수행하게 하도록 상기 비일시적 컴퓨터 판독 가능 매체 상에 저장되는 컴퓨터 실행 가능 명령어를 포함하는, 비디오 코딩 디바이스에 의한 사용을 위한 컴퓨터 프로그램 제품을 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 13

디코더로서,

제로 내지 최대 디코딩된 픽쳐 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 참조 엔트리의 개수(num_ref_entries)를 포함하는 참조 픽쳐 목록 신택스 구조(ref_pic_list_struct()) 및 현재 픽쳐를 포함하는 비트스트림을 수신하기 위한 수신 수단 - 상기 num_ref_entries는 상기 ref_pic_list_struct()에서의 엔트리의 개수를 명시하고, 상기 비트스트림은 디코딩된 픽쳐 버퍼 파라미터 신택스 구조(dpb_parameters())를 포함하고, 상기 dpb_parameters()는 최대 디코딩된 픽쳐 버퍼 마이너스 1(max_dec_pic_buffering_minus1)을 포함하고, 상기 최대 디코딩된 픽쳐 버퍼 사이즈는 상기 max_dec_pic_buffering_minus1에 대응하고, 상기 dpb_parameters()가 비디오 파라미터 세트(VPS)에 포함되는 경우, 상기 dpb_parameters()가 적용되는 출력 레이어 세트(OLS)는 상기 VPS에 의해 명시되고, 상기 dpb_parameters()가 시퀀스 파라미터 세트(SPS)에 포함되는 경우, 상기 dpb_parameters()는 상기 SPS를 참조하는 독립 레이어만을 포함하는 OLS에 적용됨 -; 및

상기 ref_pic_list_struct() 및 상기 dpb_parameters()에 기초하여 상기 현재 픽쳐를 디코딩하여 디코딩된 픽쳐를 생성하기 위한 디코딩 수단

을 포함하는 디코더.

청구항 14

제13항에 있어서,

상기 디코더는 또한 제2항 내지 제5항 중 어느 한 항의 상기 방법을 수행하도록 구성되는, 디코더.

청구항 15

인코더로서,

참조 픽쳐에 기초하여 현재 픽쳐를 비트스트림에 인코딩하기 위한; 그리고

상기 참조 픽쳐를 나타내며 제로 내지 최대 디코딩된 픽쳐 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 참조 엔트리의 개수(num_ref_entries)를 포함하는 참조 픽쳐 목록 신택스 구조(ref_pic_list_struct())를 상기 비트스트림에 인코딩하기 위한 - 상기 num_ref_entries는 상기 ref_pic_list_struct()에서의 엔트리의 개수를 명시하고, 상기 비트스트림은 디코딩된 픽쳐 버퍼 파라미터 신택스 구조(dpb_parameters())를 포함하고, 상기 dpb_parameters()는 최대 디코딩된 픽쳐 버퍼 마이너스 1(max_dec_pic_buffering_minus1)을 포함하고, 상기 최대 디코딩된 픽쳐 버퍼 사이즈는 상기 max_dec_pic_buffering_minus1에 대응하고, 상기 dpb_parameters()가 비디오 파라미터 세트(VPS)에 포함되는 경우, 상기 dpb_parameters()가 적용되는 출력 레이어 세트(OLS)는 상기 VPS에 의해 명시되고, 상기 dpb_parameters()가 시퀀스 파라미터 세트(SPS)에 포함되는 경우, 상기 dpb_parameters()는 상기 SPS를 참조하는 독립 레이어만을 포함하는 OLS에 적용됨 -

인코딩 수단; 및

디코더를 향한 통신을 위해 상기 비트스트림을 저장하기 위한 저장 수단

을 포함하는 인코더.

청구항 16

제15항에 있어서,

상기 인코더는 또한, 제7항 내지 제10항 중 어느 한 항의 상기 방법을 수행하도록 구성되는, 인코더.

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

발명의 설명

기 술 분 야

[0001] 관련 출원에 대한 교차 참조

[0002] 이 특허 출원은 Ye-Kui Wang에 의해 2019년 10월 7일자로 출원된 발명의 명칭이 "Scalability In Video Coding"인 미국 특허 임시출원 번호 제62/911,808호의 이익을 주장하는데, 이 임시출원은 참조에 의해 본원에 편입된다.

[0003] 본 개시는 일반적으로 비디오 코딩에 관한 것으로, 특히 다중 레이어 비트스트림에 대해 서브 비트스트림 추출이 수행될 때 에러를 방지하기 위한 메커니즘에 관한 것이다.

배 경 기 술

[0004] 심지어 상대적으로 짧은 비디오를 묘사하는 데 필요한 비디오 데이터의 양은 상당할 수 있는데, 이것은 데이터가 스트리밍되어야 하거나 또는 다르게는 제한된 대역폭 용량을 가지고 통신 네트워크를 통해 통신될 때 어려움을 초래할 수도 있다. 따라서, 비디오 데이터는 현대의 원격 통신 네트워크를 통해 통신되기 이전에 일반적으로 압축된다. 메모리 리소스가 제한될 수도 있기 때문에, 비디오가 스토리지 디바이스 상에 저장될 때 비디오의 사이즈도 또한 문제가 될 수 있다. 비디오 압축 디바이스는 송신 또는 저장 이전에 비디오 데이터를 코딩하기 위해 소스에서 소프트웨어 및/또는 하드웨어를 종종 사용하고, 그에 의해, 디지털 비디오 이미지를 나타내는 데 필요한 데이터의 양을 감소시킨다. 그 다음, 압축된 데이터는 비디오 데이터를 디코딩하는 비디오 압축 해제 디바이스에 의해 목적지에서 수신된다. 제한된 네트워크 리소스 및 더 높은 비디오 품질에 대한 계속 증가하는 요구로 인해, 이미지 품질을 거의 또는 전혀 희생하지 않으면서 압축 비율을 향상시키는 향상된 압축 및 압축 해제 기술이 바람직하다.

발명의 내용

[0005] 한 실시예에서, 본 개시는 디코더에 의해 구현되는 방법을 포함하는데, 그 방법은 다음의 것을 포함한다: 디코더에 의해, 제로 내지 최대 디코딩된 픽처 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 참조 엔트리의 개수(num_ref_entries)를 포함하는 참조 픽처 목록 신택스 구조(ref_pic_list_struct()) 및 현재 픽처를 포함하는 비트스트림을 수신하는 것; 디코더에 의해, ref_pic_list_struct()에 기초하여 현재 픽처를 디코딩하여 디코딩된 픽처를 생성하는 것.

[0006] 픽처는 인트라 예측, 인터 예측 및/또는 인터레이어 예측에 따라 코딩될 수도 있다. 인트라 예측에서, 픽처의 블록은 동일한 픽처의 다른 블록에 대한 참조에 의해 코딩된다. 인터 예측에서, 현재 픽처의 블록은 하나 이상의 다른 픽처의 블록에 대한 참조에 의해 코딩된다. 인터레이어 예측에서, 픽처는 레이어로 코딩되고, 출력 레이어의 픽처의 블록은 참조 레이어(들)의 픽처의 블록에 대한 참조에 의해 코딩된다. 인터 예측된 픽처의 재구성을 지원하도록 픽처 사이의 참조를 추적하기 위해 ref_pic_list_struct()가 활용될 수도 있다. 몇몇 비디오 코딩 시스템에서, ref_pic_list_struct()는 현재 픽처에 대해 활용될 수 있는 참조 엔트리의 최대 개수를 포함한다. 그러한 시스템에서, 현재 픽처에 대한 참조 엔트리의 최대 개수는 모든 레이어에 대해 전역적인 정적으로 정의된 값이다. 이 접근법에서의 문제는, 참조 레이어가 디코딩된 픽처 버퍼에서 출력 레이어와는 상이한 양의 공간을 사용한다는 것이다. 예를 들면, 참조 레이어는 픽처 재구성을 위해 공간을 사용하고, 한편 출력 레이어는 픽처 재구성 및 저장 보류 출력(storage pending output) 둘 모두를 위해 공간을 사용한다. 따라서, 참조 레이어에 대해 사용되는 더 적은 양의 공간을 지원하기 위해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수

는 출력 레이어의 픽쳐에 적용될 때 과도하게 제한적일 수도 있다. 대안적으로, 출력 레이어에 대해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 참조 레이어의 픽쳐를 디코딩하는 데 필요한 것보다 더 많은 공간을 제공할 수도 있고, 그러므로, 메모리 리소스를 낭비할 수도 있다. 본 예는 상이한 타입의 레이어에 대해 상이한 픽쳐 버퍼 사용량을 지원하도록 ref_pic_list_struct()를 제한하기 위한 메커니즘을 포함한다. 예를 들면, num_ref_entries 신택스 엘리먼트는 ref_pic_list_struct()에 포함될 수 있다. num_ref_entries는 각각의 픽쳐에서 사용되는 엔트리의 개수를 나타낸다. num_ref_entries는 최대 디코딩된 픽쳐 버퍼 사이즈에 기초하여 선택되는 범위를 유지하도록 제한될 수 있다. 최대 디코딩된 픽쳐 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변한다. 따라서, 최대 디코딩된 픽쳐 버퍼 사이즈에 기초하여 num_ref_entries를 제한하는 것은, 출력 레이어 및 참조 레이어에 대해 상이한 개수의 참조 픽쳐가 활용되는 것을 허용한다. 게다가, 각각의 픽쳐에 대한 참조 픽쳐의 세트(setofRefPics)는, 모든 레이어에 대해 전역적인 정적인 값 대신 레이어에 기초하여 변하는 최대 디코딩된 픽쳐 버퍼 사이즈에 의해 제한될 수 있다. 그러한 제약을 활용하는 것은 디코딩된 픽쳐 버퍼에서 메모리를 더욱 효율적인 할당을 지원하고, 그러므로, 더욱 최적의 메모리 사용량이 더욱 효율적인 인코딩을 촉진하기 때문에 증가된 코딩 효율성을 지원한다. 결과적으로, 인코더 및 디코더의 기능성이 증가된다. 게다가, 코딩 효율성이 증가되는데, 이것은 인코더 및 디코더 둘 모두에서의 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0007] 옵션 사항으로(optionally), 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 ref_pic_list_struct()는, ref_pic_list_struct(listIdx, rplsIdx)로서 나타내어지는 목록 인덱스(listIdx) 및 참조 픽쳐 목록 구조 인덱스(rplsIdx)에 따라 참조되고, num_ref_entries는 num_ref_entries [listIdx][rplsIdx]로 나타내어지고, num_ref_entries [listIdx][rplsIdx]는 ref_pic_list_struct(listIdx, rplsIdx)에서의 엔트리의 개수를 명시한다.

[0008] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 ref_pic_list_struct에서 참조되는 setOfRefPics에서의 픽쳐의 개수는 최대 디코딩된 픽쳐 버퍼 사이즈 마이너스 1 이하가 되도록 제한된다.

[0009] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 ref_pic_list_struct()는 참조 픽쳐 목록 0(RefPicList[0]) 및 참조 픽쳐 목록 1(RefPicList[1])을 포함하고, setOfRefPics는 현재 픽쳐와 동일한 네트워크 추상화 레이어(network abstraction layer; NAL) 단위 헤더 레이어 식별자(nuh_layer_id)를 갖는 RefPicList[0] 내의 모든 엔트리 및 현재 픽쳐와 동일한 nuh_layer_id를 갖는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 세트 고유의 픽쳐이다.

[0010] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 setOfRefPics는 각각의 픽쳐의 모든 슬라이스에 대해 동일한 세트이다.

[0011] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 비트스트림은 디코딩된 픽쳐 버퍼 파라미터 신택스 구조(dpb_parameters())를 포함하고, dpb_parameters()는 최대 디코딩된 픽쳐 버퍼 마이너스 1(max_dec_pic_buffering_minus1)을 포함하고, 최대 디코딩된 픽쳐 버퍼 사이즈는 max_dec_pic_buffering_minus1에 대응한다.

[0012] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 dpb_parameters()는 비디오 파라미터 세트(video parameter set; VPS) 또는 시퀀스 파라미터 세트(sequence parameter set; SPS)에 포함된다.

[0013] 한 실시예에서, 본 개시는 인코더에 의해 구현되는 방법을 포함하는데, 그 방법은 다음의 것을 포함한다: 인코더에 의해, 참조 픽쳐에 기초하여 현재 픽쳐를 비트스트림에 인코딩하는 것; 인코더에 의해, 참조 픽쳐를 나타내며 제로 내지 최대 디코딩된 픽쳐 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 num_ref_entries를 포함하는 ref_pic_list_struct()를 비트스트림에 인코딩하는 것; 및 인코더에 의해, 디코더를 향한 통신을 위해 비트스트림을 저장하는 것.

[0014] 픽쳐는 인트라 예측, 인터 예측 및/또는 인터레이어 예측에 따라 코딩될 수도 있다. 인트라 예측에서, 픽쳐의 블록은 동일한 픽쳐의 다른 블록에 대한 참조에 의해 코딩된다. 인터 예측에서, 현재 픽쳐의 블록은 하나 이상의 다른 픽쳐의 블록에 대한 참조에 의해 코딩된다. 인터레이어 예측에서, 픽쳐는 레이어로 코딩되고, 출력 레이어의 픽쳐의 블록은 참조 레이어(들)의 픽쳐의 블록에 대한 참조에 의해 코딩된다. 인터 예측된 픽쳐의 재구성을 지원하도록 픽쳐 사이의 참조를 추적하기 위해 ref_pic_list_struct()가 활용될 수도 있다. 몇몇 비디오

코딩 시스템에서, ref_pic_list_struct()는 현재 픽처에 대해 활용될 수 있는 참조 엔트리의 최대 개수를 포함한다. 그러한 시스템에서, 현재 픽처에 대한 참조 엔트리의 최대 개수는 모든 레이어에 대해 전역적인 정적으로 정의된 값이다. 이 접근법에서의 문제는, 참조 레이어가 디코딩된 픽처 버퍼에서 출력 레이어와는 상이한 양의 공간을 사용한다는 것이다. 예를 들면, 참조 레이어는 픽처 재구성을 위해 공간을 사용하고, 한편 출력 레이어는 픽처 재구성 및 저장 보류 출력 둘 모두를 위해 공간을 사용한다. 따라서, 참조 레이어에 대해 사용되는 더 적은 양의 공간을 지원하기 위해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 출력 레이어의 픽처에 적용될 때 과도하게 제한적일 수도 있다. 대안적으로, 출력 레이어에 대해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 참조 레이어의 픽처를 디코딩하는 데 필요한 것보다 더 많은 공간을 제공할 수도 있고, 그러므로, 메모리 리소스를 낭비할 수도 있다. 본 예는 상이한 타입의 레이어에 대해 상이한 픽처 버퍼 사용량을 지원하도록 ref_pic_list_struct()를 제한하기 위한 메커니즘을 포함한다. 예를 들면, num_ref_entries 신택스 엘리먼트는 ref_pic_list_struct()에 포함될 수 있다. num_ref_entries는 각각의 픽처에서 사용되는 엔트리의 개수를 나타낸다. num_ref_entries는 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 선택되는 범위를 유지하도록 제한될 수 있다. 최대 디코딩된 픽처 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변한다. 따라서, 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 num_ref_entries를 제한하는 것은, 출력 레이어 및 참조 레이어에 대해 상이한 개수의 참조 픽처가 활용되는 것을 허용한다. 게다가, 각각의 픽처에 대한 setofRefPics는, 모든 레이어에 대해 전역적인 정적인 값 대신 레이어에 기초하여 변하는 최대 디코딩된 픽처 버퍼 사이즈에 의해 제한될 수 있다. 그러한 제약을 활용하는 것은 디코딩된 픽처 버퍼에서 메모리를 더욱 효율적인 할당을 지원하고, 그러므로, 더욱 최적의 메모리 사용량이 더욱 효율적인 인코딩을 촉진하기 때문에 증가된 코딩 효율성을 지원한다. 결과적으로, 인코더 및 디코더의 기능성이 증가된다. 게다가, 코딩 효율성이 증가되는데, 이것은 인코더 및 디코더 둘 모두에서의 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0015] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, ref_pic_list_struct()는, ref_pic_list_struct(listIdx, rplsIdx)로서 나타내어지는 listIdx 및 rplsIdx에 따라 참조되고, num_ref_entries는 num_ref_entries [listIdx][rplsIdx]로 나타내어지고, num_ref_entries [listIdx][rplsIdx]는 ref_pic_list_struct(listIdx, rplsIdx)에서의 엔트리의 개수를 명시한다.

[0016] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 ref_pic_list_struct에서 참조되는 setOfRefPics에서의 픽처의 개수는 최대 디코딩된 픽처 버퍼 사이즈 마이너스 1 이하가 되도록 제한된다.

[0017] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 ref_pic_list_struct()는 RefPicList[0] 및 RefPicList[1]을 포함하고, setOfRefPics는 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[0] 내의 모든 엔트리 및 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 세트 고유의 픽처이다.

[0018] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 setOfRefPics는 각각의 픽처의 모든 슬라이스에 대해 동일한 세트이다.

[0019] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 비트스트림은 dpb_parameters()를 포함하고, dpb_parameters()는 max_dec_pic_buffering_minus1을 포함하고, 최대 디코딩된 픽처 버퍼 사이즈는 max_dec_pic_buffering_minus1에 대응한다.

[0020] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 dpb_parameters()는 VPS 또는 SPS에 포함된다.

[0021] 한 실시예에서, 본 개시는, 프로세서, 프로세서에 커플링되는 수신기, 프로세서에 커플링되는 메모리, 및 프로세서에 커플링되는 송신기를 포함하는 비디오 코딩 디바이스를 포함하는 비디오 코딩 디바이스를 포함하는데, 여기서 프로세서, 수신기, 메모리, 및 송신기는 전술한 양태 중 임의의 것의 방법을 수행하도록 구성된다.

[0022] 한 실시예에서, 본 개시는 비디오 코딩 디바이스에 의한 사용을 위한 컴퓨터 프로그램 제품을 포함하는 비일시적 컴퓨터 판독 가능 매체를 포함하는데, 컴퓨터 프로그램 제품은, 프로세서에 의해 실행될 때 비디오 코딩 디바이스로 하여금 전술한 양태 중 임의의 것의 방법을 수행하게 하는 비일시적 컴퓨터 판독 가능 매체 상에 저장되는 컴퓨터 실행 가능 명령어를 포함한다.

[0023] 한 실시예에서, 본 개시는 다음의 것을 포함하는 디코더를 포함한다: 제로 내지 최대 디코딩된 픽처 버퍼 사이

즈 플러스 오프셋의 범위로 제한되는 num_ref_entries를 포함하는 ref_pic_list_struct() 및 현재 픽처를 포함하는 비트스트림을 수신하기 위한 수신 수단; ref_pic_list_struct()에 기초하여 현재 픽처를 디코딩하여 디코딩된 픽처를 생성하기 위한 디코딩 수단; 및 디코딩된 비디오 시퀀스의 일부로서 디스플레이를 위해 디코딩된 픽처를 포워딩하기 위한 포워딩 수단.

[0024] 픽처는 인트라 예측, 인터 예측 및/또는 인터레이어 예측에 따라 코딩될 수도 있다. 인트라 예측에서, 픽처의 블록은 동일한 픽처의 다른 블록에 대한 참조에 의해 코딩된다. 인터 예측에서, 현재 픽처의 블록은 하나 이상의 다른 픽처의 블록에 대한 참조에 의해 코딩된다. 인터레이어 예측에서, 픽처는 레이어로 코딩되고, 출력 레이어의 픽처의 블록은 참조 레이어(들)의 픽처의 블록에 대한 참조에 의해 코딩된다. 인터 예측된 픽처의 재구성을 지원하도록 픽처 사이의 참조를 추적하기 위해 ref_pic_list_struct()가 활용될 수도 있다. 몇몇 비디오 코딩 시스템에서, ref_pic_list_struct()는 현재 픽처에 대해 활용될 수 있는 참조 엔트리의 최대 개수를 포함한다. 그러한 시스템에서, 현재 픽처에 대한 참조 엔트리의 최대 개수는 모든 레이어에 대해 전역적인 정적으로 정의된 값이다. 이 접근법에서의 문제는, 참조 레이어가 디코딩된 픽처 버퍼에서 출력 레이어와는 상이한 양의 공간을 사용한다는 것이다. 예를 들면, 참조 레이어는 픽처 재구성을 위해 공간을 사용하고, 한편 출력 레이어는 픽처 재구성 및 저장 보류 출력 둘 모두를 위해 공간을 사용한다. 따라서, 참조 레이어에 대해 사용되는 더 적은 양의 공간을 지원하기 위해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 출력 레이어의 픽처에 적용될 때 과도하게 제한적일 수도 있다. 대안적으로, 출력 레이어에 대해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 참조 레이어의 픽처를 디코딩하는 데 필요한 것보다 더 많은 공간을 제공할 수도 있고, 그러므로, 메모리 리소스를 낭비할 수도 있다. 본 예는 상이한 타입의 레이어에 대해 상이한 픽처 버퍼 사용량을 지원하도록 ref_pic_list_struct()를 제한하기 위한 메커니즘을 포함한다. 예를 들면, num_ref_entries 신택스 엘리먼트는 ref_pic_list_struct()에 포함될 수 있다. num_ref_entries는 각각의 픽처에서 사용되는 엔트리의 개수를 나타낸다. num_ref_entries는 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 선택되는 범위를 유지하도록 제한될 수 있다. 최대 디코딩된 픽처 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변한다. 따라서, 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 num_ref_entries를 제한하는 것은, 출력 레이어 및 참조 레이어에 대해 상이한 개수의 참조 픽처가 활용되는 것을 허용한다. 게다가, 각각의 픽처에 대한 setofRefPics는, 모든 레이어에 대해 전역적인 정적인 값 대신 레이어에 기초하여 변하는 최대 디코딩된 픽처 버퍼 사이즈에 의해 제한될 수 있다. 그러한 제약을 활용하는 것은 디코딩된 픽처 버퍼에서 메모리를 더욱 효율적인 할당을 지원하고, 그러므로, 더욱 최적의 메모리 사용량이 더욱 효율적인 인코딩을 촉진하기 때문에 증가된 코딩 효율성을 지원한다. 결과적으로, 인코더 및 디코더의 기능성이 증가된다. 게다가, 코딩 효율성이 증가되는데, 이것은 인코더 및 디코더 둘 모두에서의 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0025] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 디코더는 전술한 양태 중 임의의 것의 방법을 수행하도록 추가로 구성된다.

[0026] 한 실시예에서, 본 개시는 다음의 것을 포함하는 인코더를 포함한다: 참조 픽처에 기초하여 현재 픽처를 비트스트림에 인코딩하기 위한; 그리고 참조 픽처를 나타내며 제로 내지 최대 디코딩된 픽처 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 num_ref_entries를 포함하는 ref_pic_list_struct()를 비트스트림에 인코딩하기 위한 인코딩 수단; 및 디코더를 향한 통신을 위해 비트스트림을 저장하기 위한 저장 수단.

[0027] 픽처는 인트라 예측, 인터 예측 및/또는 인터레이어 예측에 따라 코딩될 수도 있다. 인트라 예측에서, 픽처의 블록은 동일한 픽처의 다른 블록에 대한 참조에 의해 코딩된다. 인터 예측에서, 현재 픽처의 블록은 하나 이상의 다른 픽처의 블록에 대한 참조에 의해 코딩된다. 인터레이어 예측에서, 픽처는 레이어로 코딩되고, 출력 레이어의 픽처의 블록은 참조 레이어(들)의 픽처의 블록에 대한 참조에 의해 코딩된다. 인터 예측된 픽처의 재구성을 지원하도록 픽처 사이의 참조를 추적하기 위해 ref_pic_list_struct()가 활용될 수도 있다. 몇몇 비디오 코딩 시스템에서, ref_pic_list_struct()는 현재 픽처에 대해 활용될 수 있는 참조 엔트리의 최대 개수를 포함한다. 그러한 시스템에서, 현재 픽처에 대한 참조 엔트리의 최대 개수는 모든 레이어에 대해 전역적인 정적으로 정의된 값이다. 이 접근법에서의 문제는, 참조 레이어가 디코딩된 픽처 버퍼에서 출력 레이어와는 상이한 양의 공간을 사용한다는 것이다. 예를 들면, 참조 레이어는 픽처 재구성을 위해 공간을 사용하고, 한편 출력 레이어는 픽처 재구성 및 저장 보류 출력 둘 모두를 위해 공간을 사용한다. 따라서, 참조 레이어에 대해 사용되는 더 적은 양의 공간을 지원하기 위해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 출력 레이어의 픽처에 적용될 때 과도하게 제한적일 수도 있다. 대안적으로, 출력 레이어에 대해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 참조 레이어의 픽처를 디코딩하는 데 필요한 것보다 더 많은 공간을 제공할 수도 있고, 그러

므로, 메모리 리소스를 낭비할 수도 있다. 본 예는 상이한 타입의 레이어에 대해 상이한 픽쳐 버퍼 사용량을 지원하도록 ref_pic_list_struct()를 제한하기 위한 메커니즘을 포함한다. 예를 들면, num_ref_entries 신택스 엘리먼트는 ref_pic_list_struct()에 포함될 수 있다. num_ref_entries는 각각의 픽쳐에서 사용되는 엔트리의 개수를 나타낸다. num_ref_entries는 최대 디코딩된 픽쳐 버퍼 사이즈에 기초하여 선택되는 범위를 유지하도록 제한될 수 있다. 최대 디코딩된 픽쳐 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변한다. 따라서, 최대 디코딩된 픽쳐 버퍼 사이즈에 기초하여 num_ref_entries를 제한하는 것은, 출력 레이어 및 참조 레이어에 대해 상이한 개수의 참조 픽쳐가 활용되는 것을 허용한다. 게다가, 각각의 픽쳐에 대한 setofRefPics는, 모든 레이어에 대해 전역적인 정적인 값 대신 레이어에 기초하여 변하는 최대 디코딩된 픽쳐 버퍼 사이즈에 의해 제한될 수 있다. 그러한 제약을 활용하는 것은 디코딩된 픽쳐 버퍼에서 메모리를 더욱 효율적인 할당을 지원하고, 그러므로, 더욱 최적의 메모리 사용량이 더욱 효율적인 인코딩을 촉진하기 때문에 증가된 코딩 효율성을 지원한다. 결과적으로, 인코더 및 디코더의 기능성이 증가된다. 게다가, 코딩 효율성이 증가되는데, 이것은 인코더 및 디코더 둘 모두에서의 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0028] 옵션 사항으로, 전술한 양태 중 임의의 것에서, 양태의 다른 구현예가 제공되는데, 여기서 인코더는 전술한 양태 중 임의의 것의 방법을 수행하도록 추가로 구성된다.

[0029] 명확화의 목적을 위해, 전술한 실시예 중 임의의 하나는 다른 전술한 실시예 중 임의의 하나 이상과 조합되어 본 개시의 범위 내에 있는 새로운 실시예를 생성할 수도 있다.

[0030] 이들 및 다른 피쳐(feature)는 첨부의 도면 및 청구범위와 연계하여 취해지는 다음의 상세한 설명으로부터 더욱 명확하게 이해될 것이다.

도면의 간단한 설명

[0031] 본 개시의 더욱 완전한 이해를 위해, 이제, 첨부의 도면 및 상세한 설명과 관련하여 취해지는 다음의 간략한 설명에 대한 참조가 이루어지는데, 첨부의 도면 및 상세한 설명에서 같은 참조 번호는 같은 부분을 나타낸다.

도 1은 비디오 신호를 코딩하는 예시적인 방법의 플로우차트이다.

도 2는 비디오 코딩을 위한 예시적인 코딩 및 디코딩(코덱) 시스템의 개략도이다.

도 3은 예시적인 비디오 인코더를 예시하는 개략도이다.

도 4는 예시적인 비디오 디코더를 예시하는 개략도이다.

도 5는 예시적인 가상 참조 디코더(HRD)를 예시하는 개략도이다.

도 6은 인터레이어 예측을 위해 구성되는 예시적인 다중 레이어 비디오 시퀀스를 예시하는 개략도이다.

도 7은 예시적인 참조 픽쳐 목록 구조를 예시하는 개략도이다.

도 8은 예시적인 비트스트림을 예시하는 개략도이다.

도 9는 예시적인 비디오 코딩 디바이스의 개략도이다.

도 10은, 최대 디코딩된 픽쳐 버퍼 사이즈에 따라 참조 엔트리의 개수가 제한되는 참조 픽쳐 목록 구조에 기초하여 비트스트림에 비디오 시퀀스를 인코딩하는 예시적인 방법의 플로우차트이다.

도 11은, 최대 디코딩된 픽쳐 버퍼 사이즈에 따라 참조 엔트리의 개수가 제한되는 참조 픽쳐 목록 구조에 기초하여 비트스트림으로부터 비디오 시퀀스를 디코딩하는 예시적인 방법의 플로우차트이다.

도 12는 최대 디코딩된 픽쳐 버퍼 사이즈에 따라 참조 엔트리의 개수가 제한되는 참조 픽쳐 목록 구조에 기초하여 비트스트림에 비디오 시퀀스를 코딩하기 위한 예시적인 시스템의 개략도이다.

발명을 실시하기 위한 구체적인 내용

[0032] 하나 이상의 실시예의 예시적인 구현예가 하기에서 제공되지만, 개시되는 시스템 및/또는 방법은, 현재 공지되어 있든 또는 존재하고 있든 간에, 임의의 개수의 기술을 사용하여 구현될 수도 있다는 것이 최초부터 이해되어야 한다. 본 개시는, 본원에서 예시되고 설명되는 예시적인 설계 및 구현예를 비롯한, 하기에서 예시되는 예시적인 구현예, 도면, 및 기술로 어떤 식으로든 제한되어서는 안되며, 첨부된 청구범위의 범위 내에서, 덧붙여,

그들의 등가물의 전체 범위 내에서 수정될 수도 있다.

[0033] 다음의 용어는 본원에서 반대되는 맥락에서 사용되지 않는 한 다음과 같이 정의된다. 구체적으로, 다음의 정의는 본 개시에 추가적인 명확성을 제공하도록 의도된다. 그러나, 용어는 상이한 맥락에서 상이하게 설명될 수도 있다. 따라서, 다음의 정의는 보충으로서 간주되어야 하며 본원에서 그러한 용어에 대해 제공되는 설명의 임의의 다른 정의를 제한하는 것으로서 간주되어서는 안된다.

[0034] 비트스트림은 인코더와 디코더 사이의 송신을 위해 압축되는 비디오 데이터를 포함하는 비트의 시퀀스이다. 인코더는 비디오 데이터를 비트스트림으로 압축하기 위해 인코딩 프로세스를 활용하도록 구성되는 디바이스이다. 디코더는 디스플레이를 위해 비트스트림으로부터 비디오 데이터를 재구성하기 위해 디코딩 프로세스를 활용하도록 구성되는 디바이스이다. 픽처는, 프레임 또는 그 필드를 생성하는 루마(luma) 샘플의 어레이 및/또는 크로마(chroma) 샘플의 어레이이다. 인코딩 또는 디코딩되고 있는 픽처는 논의의 명확화를 위해 현재 픽처로서 지칭될 수 있다. 코딩된 픽처는, 액세스 단위(access unit; AU) 내의 NAL 단위 헤더 레이어 식별자(nuh_layer_id)의 특정한 값을 갖는 비디오 코딩 레이어(video coding layer; VCL) 네트워크 추상화 레이어(NAL) 단위를 포함하며 픽처의 모든 코딩 트리 단위(coding tree unit; CTU)를 포함하는 픽처의 코딩된 표현이다. 디코딩된 픽처는, 코딩된 픽처에 디코딩 프로세스를 적용하는 것에 의해 생성되는 픽처이다. 슬라이스는, 단일의 NAL 단위(VCL NAL 단위)로 배타적으로 포함되는 픽처의 정수 개수의 완전한 타일 또는 (예를 들면, 타일 내의) 정수 개수의 연속적인 완전한 코딩 트리 단위(CTU) 행이다. NAL 단위는, 원시 바이트 시퀀스 페이로드(Raw Byte Sequence Payload; RBSP)의 형태의 데이터, 데이터의 타입의 지시(indication), 및 소망에 따라 산재되는 에뮬레이션 방지 바이트를 포함하는 신택스 구조이다. VCL NAL 단위는 픽처의 코딩된 슬라이스와 같은 비디오 데이터를 포함하도록 코딩되는 NAL 단위이다. 비 VCL NAL 단위(non-VCL NAL unit)는, 비디오 데이터의 디코딩, 적합성 검사의 수행, 또는 다른 동작을 지원하는 신택스 및/또는 파라미터와 같은 비 비디오 데이터(non-video data)를 포함하는 NAL 단위이다. 레이어는, 레이어 Id(식별자)에 의해 나타내어지는 바와 같은 명시된 특성(예를 들면, 공통 해상도, 프레임 레이트, 이미지 사이즈, 등등) 및 관련된 비 VCL NAL 단위(non-VCL NAL unit)를 공유하는 VCL NAL 단위의 세트이다. NAL 단위 헤더 레이어 식별자(nuh_layer_id)는 NAL 단위를 포함하는 레이어의 식별자를 명시하는 신택스 엘리먼트이다.

[0035] 가상 참조 디코더(hypothetical reference decoder; HRD)는, 명시된 제약을 가지고 적합성을 검증하기 위해, 인코딩 프로세스에 의해 생성되는 비트스트림의 가변성을 검사하는 인코더에 대해 동작하는 디코더 모델이다. 비트스트림 적합성 테스트는, 인코딩된 비트스트림이 다기능 비디오 코딩(Versatile Video Coding; VVC)과 같은 표준을 준수하는지의 여부를 결정하기 위한 테스트이다. VPS(비디오 파라미터 세트)는 전체 비디오에 관련되는 파라미터를 포함하는 신택스 구조이다. 시퀀스 파라미터 세트(SPS)는 제로 개 이상의 전체 코딩된 레이어 비디오 시퀀스(coded layer video sequence; CLVS)에 적용되는 신택스 엘리먼트를 포함하는 신택스 구조이다. CLVS는 동일한 nuh_layer_id 값을 갖는 코딩된 픽처의 시퀀스이다. 참조 픽처는, 단기간 참조 픽처, 장기간 참조 픽처, 또는 인터레이어 참조 픽처인 픽처로서 정의될 수도 있다. 예를 들면, 참조 픽처는 인터 예측에 따른 참조에 의해 다른 픽처의 블록 및/또는 샘플을 코딩하기 위해 사용되는 블록 및/또는 샘플을 포함하는 임의의 픽처일 수도 있다. 참조 픽처 목록 0(RefPicList[0])는 단방향 예측(P) 슬라이스의 인터 예측을 위해 사용되는 참조 픽처 목록(예를 들면, 대응하는 참조 픽처의 목록을 포함함) 또는 양방향 예측(B) 슬라이스의 인터 예측을 위해 사용되는 두 개의 참조 픽처 목록 중 제1의 것이다. 참조 픽처 목록 1(RefPicList[1])은 (예를 들면, RefPicList[0]과 연계하여) B 슬라이스의 인터 예측을 위해 사용되는 제2 참조 픽처 목록이다. 참조 픽처 목록 신택스 구조(ref_pic_list_struct())는 RefPicList[0] 및 RefPicList[1]을 포함하는 신택스 구조이다. 참조 엔트리는 참조 인덱스에 기초하여 대응하는 참조 픽처를 나타내는 참조 픽처 목록 내의 엔트리이다. 목록 인덱스(listIdx)는 RefPicList[0] 및/또는 RefPicList[1]과 같은, 대응하는 참조 픽처 목록을 나타내는 인덱스이다. 참조 픽처 목록 구조 인덱스(rplsIdx)는 대응하는 참조 픽처 목록 내의 참조 엔트리를 나타내는 인덱스이다. 참조 엔트리의 개수(num_ref_entries)는 ref_pic_list_struct()에서의 참조 엔트리의 개수를 나타내는 신택스 엘리먼트이다. 참조 픽처의 세트(setOfRefPics)는, 현재 픽처와 동일한 nuh_layer_id 값을 갖는 RefPicList[0] 및/또는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 고유의 픽처의 세트이다. 디코딩된 픽처 버퍼(decoded picture buffer; DPB)는, 예를 들면, 디코더 및/또는 HRD에서, 참조, 출력 재정렬, 또는 출력 지연을 위해 디코딩된 픽처를 유지하도록 구성되는 버퍼이다. 디코딩된 픽처 버퍼 파라미터 신택스 구조(dpb_parameters())는, 하나 이상의 출력 레이어 세트(output layer set; OLS)에 대한 DPB 사이즈, 최대 픽처 재정렬 개수(maximum picture reorder number), 및 최대 레이턴시에 관한 정보를 제공하는 신택스 구조이다. 최대 디코딩된 픽처 버퍼 사이즈는 DPB의 최대 요구 사이즈(maximum required size)를 픽처 저장 버퍼의 단위로 명시하는 유도된 변수이다. 최대 디코딩된 픽처 버퍼 마이너스 1(max_dec_pic_buffering_minus1)은 DPB의 최대

요구 사이즈를 픽쳐 저장 버퍼의 단위로 명시하는 신택스 엘리먼트이다. 액세스 단위(AU)는 동일한 출력 시간과 모두 관련되는 상이한 레이어의 코딩된 픽쳐의 세트이다. 코딩된 비디오 시퀀스는 하나 이상의 코딩된 픽쳐의 세트이다. 디코딩된 비디오 시퀀스는 하나 이상의 디코딩된 픽쳐의 세트이다.

[0036] 다음의 두문자어(acronym), 액세스 단위(Access Unit; AU), 코딩 트리 블록(Coding Tree Block; CTB), 코딩 트리 단위(Coding Tree Unit; CTU), 코딩 단위(Coding Unit; CU), 코딩된 레이어 비디오 시퀀스(Coded Layer Video Sequence; CLVS), 코딩된 레이어 비디오 시퀀스 시작(Coded Layer Video Sequence Start; CLVSS), 코딩된 비디오 시퀀스(Coded Video Sequence; CVS), 코딩된 비디오 시퀀스 시작(Coded Video Sequence Start; CVSS), 공동 비디오 전문가 팀(Joint Video Expert Team; JVET), 가상 참조 디코더(Hypothetical Reference Decoder; HRD), 모션 제한 타일 세트(Motion Constrained Tile Set; MCTS), 최대 전송 단위(Maximum Transfer Unit; MTU), 네트워크 추상화 레이어(Network Abstraction Layer; NAL), 출력 레이어 세트(Output Layer Set; OLS), 동작 포인트(Operation Point; OP) 픽쳐 순서 카운트(Picture Order Count; POC), 랜덤 액세스 포인트(Random Access Point; RAP), 원시 바이트 시퀀스 페이로드(Raw Byte Sequence Payload; RBSP), 시퀀스 파라미터 세트(Sequence Parameter Set; SPS), 비디오 파라미터 세트(Video Parameter Set; VPS), 다기능 비디오 코딩(Versatile Video Coding; VVC)이 본원에서 사용된다.

[0037] 데이터의 손실을 최소화하면서 비디오 파일의 사이즈를 감소시키기 위해, 많은 비디오 압축 기술이 활용될 수 있다. 예를 들면, 비디오 압축 기술은, 비디오 시퀀스에서 데이터 중복성을 감소 또는 제거하기 위해, 공간적(예를 들면, 인트라 픽쳐(intra-picture)) 예측 및/또는 시간적(예를 들면, 인터 픽쳐(inter-picture)) 예측을 수행하는 것을 포함할 수 있다. 블록 기반의 비디오 코딩의 경우, 비디오 슬라이스(예를 들면, 비디오 픽쳐 또는 비디오 픽쳐의 일부)는, 트리블록, 코딩 트리 블록(coding tree block; CTB), 코딩 트리 단위(CTU), 코딩 단위(CU), 및/또는 코딩 노드로도 또한 지칭될 수도 있는 비디오 블록으로 구획될 수도 있다. 픽쳐의 인트라 코딩된(I) 슬라이스의 비디오 블록은 동일한 픽쳐의 이웃 블록에 있는 참조 샘플에 대한 공간적 예측을 사용하여 코딩된다. 픽쳐의 인터 코딩된 단방향 예측(P) 또는 양방향 예측(B) 슬라이스의 비디오 블록은 동일한 픽쳐의 이웃 블록에 있는 참조 샘플에 대한 공간적 예측 또는 다른 참조 픽쳐의 참조 샘플에 대한 시간적 예측을 활용하는 것에 의해 코딩될 수도 있다. 픽쳐는 프레임 및/또는 이미지로서 지칭될 수도 있고, 참조 픽쳐는 참조 프레임 및/또는 참조 이미지로서 지칭될 수도 있다. 공간적 또는 시간적 예측은 이미지 블록을 나타내는 예측 블록으로 귀결된다. 잔차 데이터는 원래의 이미지 블록과 예측 블록 사이의 픽셀 차이를 나타낸다. 따라서, 인터 코딩된 블록은, 예측 블록을 형성하는 참조 샘플의 블록을 가리키는 모션 벡터 및 코딩된 블록과 예측 블록 사이의 차이를 나타내는 잔차 데이터에 따라 인코딩된다. 인트라 코딩된 블록은 인트라 코딩 모드 및 잔차 데이터에 따라 인코딩된다. 추가적인 압축을 위해, 잔차 데이터는 픽셀 도메인으로부터 변환 도메인으로 변환될 수도 있다. 이들은, 양자화될 수도 있는 잔차 변환 계수로 귀결된다. 양자화된 변환 계수는 초기에 이차원 어레이로 배열될 수도 있다. 양자화된 변환 계수는 변환 계수의 일차원 벡터를 생성하기 위해 스캐닝될 수도 있다. 더욱 더 많은 압축을 달성하기 위해 엔트로피 코딩이 적용될 수도 있다. 그러한 비디오 압축 기술은 하기에서 더욱 상세하게 논의된다.

[0038] 인코딩된 비디오가 정확하게 디코딩될 수 있는 것을 보장하기 위해, 비디오는 대응하는 비디오 코딩 표준에 따라 인코딩 및 디코딩된다. 비디오 코딩 표준은, 국제전기통신연합(International Telecommunication Union; ITU) 표준화 부문(ITU-T) H.261, 국제 표준화 기구/국제 전기 기술 위원회(International Organization for Standardization/International Electrotechnical Commission; ISO/IEC) 동영상 전문가 그룹(Motion Picture Experts Group; MPEG)-1 파트 2, ITU-T H.262 또는 ISO/IEC MPEG-2 파트 2, ITU-T H.263, ISO/IEC MPEG-4 파트 2, 고급 비디오 코딩(Advanced Video Coding; AVC) - ITU-T H.264 또는 ISO/IEC MPEG-4 파트 10으로서 또한 알려져 있음 - , 및 고효율 비디오 코딩(High Efficiency Video Coding; HEVC) - ITU-T H.265 또는 MPEG-H 파트 2로서 또한 알려져 있음 - 을 포함한다. AVC는, 스케일러블 비디오 코딩(Scalable Video Coding; SVC), 멀티뷰 비디오 코딩(Multiview Video Coding; MVC) 및 멀티뷰 비디오 코딩 플러스 깊이(Multiview Video Coding plus Depth; MVC+D), 및 삼차원(3D) AVC(3D-AVC)와 같은 확장안을 포함한다. HEVC는, 스케일러블 HEVC(Scalable HEVC; SHVC), 멀티뷰 HEVC(Multiview HEVC; MV-HEVC), 및 3D HEVC(3D-HEVC)와 같은 확장안을 포함한다. ITU-T 및 ISO/IEC의 공동 비디오 전문가 팀(JVET)은 다기능 비디오 코딩(VVC)으로서 또한 지칭되는 비디오 코딩 표준을 개발하기 시작하였다. VVC는 JVET-O2001-v14를 포함하는 작업 초안(Working Draft; WD)에 포함된다.

[0039] 픽쳐는 인트라 예측, 인터 예측 및/또는 인터레이어 예측에 따라 코딩될 수도 있다. 인트라 예측에서, 픽쳐의 블록은 동일한 픽쳐의 다른 블록에 대한 참조에 의해 코딩된다. 인터 예측에서, 현재 픽쳐의 블록은 하나 이상

의 다른 픽처의 블록에 대한 참조에 의해 코딩된다. 인터레이어 예측에서, 픽처는 레이어로 코딩되고, 출력 레이어의 픽처의 블록은 참조 레이어(들)의 픽처의 블록에 대한 참조에 의해 코딩된다. 인터 예측된 픽처의 재구성을 지원하도록 픽처 사이의 참조를 추적하기 위해 참조 픽처 목록 신택스 구조(ref_pic_list_struct())가 활용될 수도 있다. 몇몇 비디오 코딩 시스템에서, ref_pic_list_struct()는 현재 픽처에 대해 활용될 수 있는 참조 엔트리의 최대 개수를 포함한다. 그러한 시스템에서, 현재 픽처에 대한 참조 엔트리의 최대 개수는 모든 레이어에 대해 전역적인 정적으로 정의된 값이다. 이 접근법에서의 문제는, 참조 레이어가 디코딩된 픽처 버퍼에서 출력 레이어와는 상이한 양의 공간을 사용한다는 것이다. 예를 들면, 참조 레이어는 픽처 재구성을 위해 공간을 사용하고, 한편 출력 레이어는 픽처 재구성 및 저장 보류 출력 둘 모두를 위해 공간을 사용한다. 따라서, 참조 레이어에 대해 사용되는 더 적은 양의 공간을 지원하기 위해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 출력 레이어의 픽처에 적용될 때 과도하게 제한적일 수도 있다. 대안적으로, 출력 레이어에 대해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 참조 레이어의 픽처를 디코딩하는 데 필요한 것보다 더 많은 공간을 제공할 수도 있고, 그러므로, 메모리 리소스를 낭비할 수도 있다.

[0040] 상이한 타입의 레이어에 대해 상이한 픽처 버퍼 사용량을 지원하기 위해 ref_pic_list_struct()를 제한하기 위한 메커니즘이 본원에서 개시된다. 예를 들면, 참조 엔트리의 개수(num_ref_entries) 신택스 엘리먼트가 ref_pic_list_struct()에 포함될 수 있다. num_ref_entries는 각각의 픽처에서 사용되는 엔트리의 개수를 나타낸다. num_ref_entries는 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 선택되는 범위를 유지하도록 제한될 수 있다. 최대 디코딩된 픽처 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변한다. 따라서, 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 num_ref_entries를 제한하는 것은, 출력 레이어 및 참조 레이어에 대해 상이한 개수의 참조 픽처가 활용되는 것을 허용한다. 게다가, 각각의 픽처에 대한 참조 픽처의 세트(setofRefPics)는, 모든 레이어에 대해 전역적인 정적인 값 대신 레이어에 기초하여 변하는 최대 디코딩된 픽처 버퍼 사이즈에 의해 제한될 수 있다. 그러한 제약을 활용하는 것은 디코딩된 픽처 버퍼에서 메모리를 더욱 효율적인 할당을 지원하고, 그러므로, 더욱 최적의 메모리 사용량이 더욱 효율적인 인코딩을 촉진하기 때문에 증가된 코딩 효율성을 지원한다. 결과적으로, 인코더 및 디코더의 기능성이 증가된다. 게다가, 코딩 효율성이 증가되는데, 이것은 인코더 및 디코더 둘 모두에서의 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0041] 도 1은 비디오 신호를 코딩하는 예시적인 동작 방법(100)의 플로우차트이다. 구체적으로, 비디오 신호는 인코더에서 인코딩된다. 인코딩 프로세스는 비디오 파일 사이즈를 감소시키기 위해 다양한 메커니즘을 활용하는 것에 의해 비디오 신호를 압축한다. 더 작은 파일 사이즈는, 관련된 대역폭 오버헤드를 감소시키면서, 압축된 비디오 파일이 유저를 향해 송신되는 것을 허용한다. 그 다음, 디코더는, 엔드 유저에 대한 디스플레이를 위해 원래의 비디오 신호를 재구성하기 위해, 압축된 비디오 파일을 디코딩한다. 디코딩 프로세스는 디코더가 비디오 신호를 일치하게 재구성하는 것을 허용하기 위해 인코딩 프로세스를 일반적으로 미러링한다.

[0042] 단계(101)에서, 비디오 신호는 인코더에 입력된다. 예를 들면, 비디오 신호는 메모리에 저장되는 비압축 비디오 파일일 수도 있다. 다른 예로서, 비디오 파일은 비디오 카메라와 같은 비디오 캡쳐 디바이스에 의해 캡쳐될 수도 있고, 비디오의 라이브 스트리밍을 지원하도록 인코딩될 수도 있다. 비디오 파일은 오디오 컴포넌트와 비디오 컴포넌트 둘 모두를 포함할 수도 있다. 비디오 컴포넌트는, 시퀀스에서 봤을 때, 모션의 시각적 인상을 제공하는 일련의 이미지 프레임을 포함한다. 프레임은 본원에서 루마 성분(또는 루마 샘플)로서 지칭되는 광, 및 크로마 성분(또는 컬러 샘플)으로서 지칭되는 컬러의 관점에서 표현되는 픽셀을 포함한다. 몇몇 예에서, 프레임은 삼차원 뷰잉(three dimensional viewing)을 지원하기 위해 깊이 값을 또한 포함할 수도 있다.

[0043] 단계(103)에서, 비디오는 블록으로 구획된다. 구획화(partitioning)는 압축을 위해 각각의 프레임의 픽셀을 정사각형 및/또는 직사각형 블록으로 세분화하는 것을 포함한다. 예를 들면, 고효율 비디오 코딩(HEVC)(H.265 및 MPEG-H 파트 2로서 또한 알려져 있음)에서, 프레임은, 먼저, 사전 정의된 사이즈(예를 들면, 64 픽셀×64 픽셀)의 블록인 코딩 트리 단위(CTU)로 분할될 수 있다. CTU는 루마 및 크로마 샘플 둘 모두를 포함한다. CTU를 블록으로 분할하기 위해, 그 다음, 추가적인 인코딩을 지원하는 구성이 달성될 때까지 블록을 재귀적으로 세분화하기 위해, 코딩 트리가 활용될 수도 있다. 예를 들면, 프레임의 루마 성분은, 개개의 블록이 상대적으로 균질한 조명 값을 포함할 때까지, 세분화될 수도 있다. 게다가, 프레임의 크로마 성분은, 개개의 블록이 상대적으로 균질한 컬러 값을 포함할 때까지, 세분화될 수도 있다. 따라서, 구획화 메커니즘은 비디오 프레임의 콘텐츠에 따라 변한다.

[0044] 단계(105)에서, 단계(103)에서 구획되는 이미지 블록을 압축하기 위해 다양한 압축 메커니즘이 활용된다. 예를 들면, 인터 예측 및/또는 인트라 예측이 활용될 수도 있다. 인터 예측은, 공통 장면의 오브젝트가 연속 프레임

에서 나타나는 경향이 있다는 사실을 이용하도록 설계된다. 따라서, 참조 프레임의 오브젝트를 묘사하는 블록은 인접 프레임에서 반복적으로 설명될 필요가 없다. 구체적으로, 테이블과 같은 오브젝트는 다수의 프레임에 걸쳐 일정한 포지션에 남아 있을 수도 있다. 그러므로 테이블은 한 번 설명되고 인접 프레임은 참조 프레임을 다시 참조할 수 있다. 다수의 프레임에 걸쳐 오브젝트를 매치시키기 위해 패턴 매칭 메커니즘이 활용될 수도 있다. 게다가, 움직이는 오브젝트는, 예를 들면, 오브젝트 움직임 또는 카메라 움직임에 기인하여, 다수의 프레임에 걸쳐 표현될 수도 있다. 특정한 예로서, 비디오는 다수의 프레임에 걸쳐 스크린을 가로질러 움직이는 자동차를 보여줄 수도 있다. 모션 벡터는 그러한 움직임을 설명하기 위해 활용될 수 있다. 모션 벡터는, 한 프레임에서의 오브젝트의 좌표로부터 참조 프레임에서의 오브젝트의 좌표로의 오프셋을 제공하는 이차원 벡터이다. 그러한 만큼, 인터 예측은 현재 프레임의 이미지 블록을, 참조 프레임의 대응하는 블록으로부터의 오프셋을 나타내는 모션 벡터의 세트로서 인코딩할 수 있다.

[0045] 인트라 예측은 공통 프레임의 블록을 인코딩한다. 인트라 예측은 루마 및 크로마 성분이 프레임에서 클러스터화되는 경향이 있다는 사실을 이용한다. 예를 들면, 트리의 한 부분에 있는 녹색의 패치는 녹색의 유사한 패치에 인접하게 배치되는 경향이 있다. 인트라 예측은 다수의 방향성 예측 모드(예를 들면, HEVC에서 33 개), 평면 모드, 및 직류(direct current; DC) 모드를 활용한다. 방향성 모드는, 현재 블록이 대응하는 방향에서 이웃 블록의 샘플과 유사/동일하다는 것을 나타낸다. 평면 모드는, 행/열(예를 들면, 평면)을 따르는 일련의 블록이 행의 가장자리에 있는 이웃 블록에 기초하여 보간될 수 있다는 것을 나타낸다. 평면 모드는, 사실상, 값을 변경함에 있어서 상대적으로 일정한 기울기를 활용하는 것에 의해 행/열에 걸친 광/컬러의 부드러운 전이를 나타낸다. DC 모드는 경계 평활화를 위해 활용되며, 블록이, 방향성 예측 모드의 각도 방향과 관련되는 모든 이웃 블록의 샘플과 관련되는 평균 값과 유사/동일하다는 것을 나타낸다. 따라서, 인트라 예측 블록은 이미지 블록을, 실제 값 대신, 다양한 관계형 예측 모드 값으로서 표현할 수 있다. 게다가, 인터 예측 블록은 이미지 블록을, 실제 값 대신, 모션 벡터 값으로서 나타낼 수 있다. 어느 경우든, 예측 블록은, 몇몇 경우에, 이미지 블록을 정확하게 나타내지 않을 수도 있다. 임의의 차이는 잔차 블록에 저장된다. 파일을 추가로 압축하기 위해, 잔차 블록에 변환이 적용될 수도 있다.

[0046] 단계(107)에서, 다양한 필터링 기술이 적용될 수도 있다. HEVC에서, 필터는 루프내 필터링 스킴(in-loop filtering scheme)에 따라 적용된다. 상기에서 논의되는 블록 기반의 예측은 디코더에서 농담이 고르지 않은(blocky) 이미지의 생성을 초래할 수도 있다. 게다가, 블록 기반의 예측 스킴은 블록을 인코딩할 수도 있고, 그 다음, 참조 블록으로서의 나중의 사용을 위해 인코딩된 블록을 재구성할 수도 있다. 루프내 필터링 스킴은 노이즈 억제 필터, 블록화 제거 필터(de-blocking filter), 적응적 루프 필터, 및 샘플 적응적 오프셋(sample adaptive offset; SAO) 필터를 블록/프레임에 반복적으로 적용한다. 이들 필터는, 인코딩된 파일이 정확하게 재구성될 수 있도록 그러한 차단 아티팩트(artifact)를 완화한다. 게다가, 재구성된 참조 블록에 기초하여 인코딩되는 후속하는 블록에서 아티팩트가 추가적인 아티팩트를 생성할 가능성이 적도록, 이들 필터는 재구성된 참조 블록에서 아티팩트를 완화한다.

[0047] 일단 비디오 신호가 구획, 압축, 및 필터링되면, 결과적으로 나타나는 데이터는, 단계(109)에서, 비트스트림에서 인코딩된다. 비트스트림은 상기에서 논의되는 데이터뿐만 아니라 디코더에서 적절한 비디오 신호 재구성을 지원하기 위해 소망되는 임의의 시그널링 데이터를 포함한다. 예를 들면, 그러한 데이터는 구획 데이터, 예측 데이터, 잔차 블록, 및 디코더에 코딩 명령어를 제공하는 다양한 플래그를 포함할 수도 있다. 비트스트림은 요청시 디코더를 향한 송신을 위해 메모리에 저장될 수도 있다. 비트스트림은 또한 복수의 디코더를 향해 브로드캐스트 및/또는 멀티캐스트될 수도 있다. 비트스트림 생성은 반복적인 프로세스이다. 따라서, 단계(101, 103, 105, 107, 및 109)는 많은 프레임 및 블록에 걸쳐 연속적으로 및/또는 동시적으로 발생할 수도 있다. 도 1에서 도시되는 순서는 논의의 명확성 및 용이성을 위해 제시되며, 비디오 코딩 프로세스를 특정한 순서로 제한하도록 의도되지는 않는다.

[0048] 디코더는 비트스트림을 수신하고 단계(111)에서 디코딩 프로세스를 시작한다. 구체적으로, 디코더는 비트스트림을 대응하는 신택스 및 비디오 데이터로 변환하기 위해 엔트로피 디코딩 스킴을 활용한다. 디코더는 단계(111)에서 프레임에 대한 구획을 결정하기 위해 비트스트림으로부터의 신택스 데이터를 활용한다. 구획화는 단계(103)의 블록 구획화의 결과와 매치해야 한다. 이제, 단계(111)에서 활용되는 바와 같은 엔트로피 인코딩/디코딩이 설명된다. 인코더는, 입력 이미지(들)에서의 값의 공간적 위치 결정에 기초하여 여러 가지 가능한 선택지로부터 블록 구획화 스킴을 선택하는 것과 같은, 압축 프로세스 동안 많은 선택을 행한다. 정확한 선택을 시그널링하는 것은 많은 개수의 빈(bin)을 활용할 수도 있다. 본원에서 사용되는 바와 같이, 빈은 변수로서 취급되는 이진 값(예를 들면, 컨텍스트에 따라 변할 수도 있는 비트 값)이다. 엔트로피 코딩은, 허용 가능한 옵션의

세트를 남기면서, 특정한 경우에 대해 명확하게 실행 가능하지 않은 임의의 옵션을 인코더가 폐기하는 것을 허용한다. 그 다음, 각각의 허용 가능한 옵션은 코드 워드를 할당받는다. 코드 워드의 길이는 허용 가능한 옵션의 개수에 기초한다(예를 들면, 두 개의 옵션의 경우 하나의 빈, 세 개 내지 네 개의 옵션의 경우 두 개의 빈, 등등). 그 다음, 인코더는 선택된 옵션에 대한 코드 워드를 인코딩한다. 이 스킴은, 모든 가능한 옵션의 잠재적으로 큰 세트로부터 선택하는 것을 고유하게 나타내는 것과는 대조적으로, 허용 가능한 옵션의 작은 서브세트로부터의 선택을 고유하게 나타내기 위해 소망되는 만큼 코드 워드가 크기 때문에, 코드 워드의 사이즈를 감소시킨다. 그 다음, 디코더는 인코더와 유사한 방식으로 허용 가능한 옵션의 세트를 결정하는 것에 의해 선택을 디코딩한다. 허용 가능한 옵션의 세트를 결정하는 것에 의해, 디코더는 코드 워드를 판독할 수 있고 인코더에 의해 만들어지는 선택을 결정할 수 있다.

[0049] 단계(113)에서, 디코더는 블록 디코딩을 수행한다. 구체적으로, 디코더는 잔차 블록을 생성하기 위해 역변환을 활용한다. 그 다음, 디코더는, 구획화에 따라 이미지 블록을 재구성하기 위해, 잔차 블록 및 대응하는 예측 블록을 활용한다. 예측 블록은, 단계(105)에서, 인코더에서 생성되는 바와 같은 인트라 예측 블록 및 인터 예측 블록 둘 모두를 포함할 수도 있다. 그 다음, 재구성된 이미지 블록은, 단계(111)에서 결정되는 구획화 데이터에 따라 재구성된 비디오 신호의 프레임에 배치된다. 단계(113)에 대한 신택스는 상기에서 논의되는 바와 같이 엔트로피 코딩을 통해 비트스트림에서 또한 시그널링될 수도 있다.

[0050] 단계(115)에서, 재구성된 비디오 신호의 프레임에 대해, 인코더에서의 단계(107)와 유사한 방식으로, 필터링이 수행된다. 예를 들면, 블록화 아티팩트(blocking artifact)를 제거하기 위해, 노이즈 억제 필터, 블록화 제거 필터, 적응적 루프 필터, 및 SAO 필터가 프레임에 적용될 수도 있다. 일단 프레임이 필터링되면, 비디오 신호는 엔드 유저에 의한 뷰잉을 위해 단계(117)에서 디스플레이로 출력될 수 있다.

[0051] 도 2는 비디오 코딩을 위한 예시적인 코딩 및 디코딩(코덱) 시스템(200)의 개략도이다. 구체적으로, 코덱 시스템(200)은 동작 방법(100)의 구현을 지원하기 위한 기능성을 제공한다. 코덱 시스템(200)은 인코더 및 디코더 둘 모두에서 활용되는 컴포넌트를 묘사하도록 일반화된다. 코덱 시스템(200)은 동작 방법(100)의 단계(101 및 103)와 관련하여 논의되는 바와 같이 비디오 신호를 수신 및 구획하는데, 이것은 구획된 비디오 신호(201)를 초래한다. 코덱 시스템(200)은, 그 다음, 방법(100)의 단계(105, 107, 및 109)와 관련하여 논의되는 바와 같이 인코더로서 작용할 때 구획된 비디오 신호(201)를 코딩된 비트스트림으로 압축한다. 디코더로서 작용할 때, 코덱 시스템(200)은 동작 방법(100)의 단계(111, 113, 115, 및 117)와 관련하여 논의되는 바와 같이 비트스트림으로부터 출력 비디오 신호를 생성한다. 코덱 시스템(200)은 일반 코더 제어 컴포넌트(211), 변환 스케일링 및 양자화 컴포넌트(213), 인트라 픽쳐 추정 컴포넌트(215), 인트라 픽쳐 예측 컴포넌트(217), 모션 보상 컴포넌트(219), 모션 추정 컴포넌트(221), 스케일링 및 역변환 컴포넌트(229), 필터 제어 분석 컴포넌트(227), 루프내 필터 컴포넌트(225), 디코딩된 픽쳐 버퍼 컴포넌트(223), 및 헤더 포맷팅(header formatting) 및 컨텍스트 적응 이진 산술 코딩(context adaptive binary arithmetic coding; CABAC) 컴포넌트(231)를 포함한다. 그러한 컴포넌트는 도시되는 바와 같이 커플링된다. 도 2에서, 검은색 라인은 인코딩/디코딩될 데이터의 이동을 나타내고, 한편, 파선(dashed line)은 다른 컴포넌트의 동작을 제어하는 제어 데이터의 이동을 나타낸다. 코덱 시스템(200)의 컴포넌트 모두는 인코더에서 존재할 수도 있다. 디코더는 코덱 시스템(200)의 컴포넌트의 서브세트를 포함할 수도 있다. 예를 들면, 디코더는 인트라 픽쳐 예측 컴포넌트(217), 모션 보상 컴포넌트(219), 스케일링 및 역변환 컴포넌트(229), 루프내 필터 컴포넌트(225), 및 디코딩된 픽쳐 버퍼 컴포넌트(223)를 포함할 수도 있다. 이들 컴포넌트가 이제 설명된다.

[0052] 구획된 비디오 신호(201)는 코딩 트리에 의해 픽셀 블록으로 구획된 캡쳐된 비디오 시퀀스이다. 코딩 트리는 픽셀의 블록을 픽셀의 더 작은 블록으로 세분화하기 위해 다양한 분할 모드를 활용한다. 그 다음, 이들 블록은 더 작은 블록으로 추가로 세분화될 수 있다. 블록은 코딩 트리 상의 노드로서 지칭될 수도 있다. 더 큰 부모 노드(parent node)는 더 작은 자식 노드(child node)로 분할된다. 노드가 세분되는 횟수는 노드/코딩 트리의 깊이로서 지칭된다. 분할된 블록은, 몇몇 경우에, 코딩 단위(CU)에 포함될 수 있다. 예를 들면, CU는, CU에 대한 대응하는 신택스 명령어와 함께, 루마 블록, 적색 차이 크로마(Cr) 블록(들), 및 청색 차이 크로마(Cb) 블록(들)을 포함하는 CTU의 하위 부분일 수 있다. 분할 모드는, 노드를, 활용되는 분할 모드에 따라 다양한 형상의 두 개, 세 개, 또는 네 개의 자식 노드로, 각각, 구획하기 위해 활용되는 이진 트리(binary tree; BT), 트리플 트리(triple tree; TT) 및 쿼드트리(quad tree; QT)를 포함할 수도 있다. 구획된 비디오 신호(201)는 일반 코더 제어 컴포넌트(211), 변환 스케일링 및 양자화 컴포넌트(213), 인트라 픽쳐 추정 컴포넌트(215), 필터 제어 분석 컴포넌트(227), 및 압축을 위한 모션 추정 컴포넌트(221)로 포워딩된다.

[0053] 일반 코더 제어 컴포넌트(211)는 애플리케이션 제약에 따라 비디오 시퀀스의 이미지를 비트스트림으로 코딩하는

것에 관련되는 결정을 내리도록 구성된다. 예를 들면, 일반 코더 제어 컴포넌트(211)는 비트레이트/비트스트림 사이즈 대 재구성 품질의 최적화를 관리한다. 그러한 결정은 저장 공간/대역폭 이용 가능성 및 이미지 해상도 요청에 기초하여 이루어질 수도 있다. 일반 코더 제어 컴포넌트(211)는 버퍼 언더런 및 오버런 문제를 완화하기 위해 송신 속도를 고려하여 버퍼 활용을 또한 관리한다. 이들 문제를 관리하기 위해, 일반 코더 제어 컴포넌트(211)는 다른 컴포넌트에 의한 구획화, 예측, 및 필터링을 관리한다. 예를 들면, 일반 코더 제어 컴포넌트(211)는 해상도를 증가시키고 대역폭 사용을 증가시키기 위해 압축 복잡도를 동적으로 증가시킬 수도 있거나 또는 해상도 및 대역폭 사용을 감소시키기 위해 압축 복잡도를 감소시킬 수도 있다. 그러므로, 비디오 신호 재구성 품질을 비트 레이트 문제와 균형을 맞추기 위해, 일반 코더 제어 컴포넌트(211)는 코덱 시스템(200)의 다른 컴포넌트를 제어한다. 일반 코더 제어 컴포넌트(211)는 다른 컴포넌트의 동작을 제어하는 제어 데이터를 생성한다. 제어 데이터는, 디코더에서의 디코딩을 위한 신호 파라미터로 비트스트림에서 인코딩되도록, 헤더 포맷팅 및 CABAC 컴포넌트(231)로 또한 포워딩된다.

[0054] 구획된 비디오 신호(201)는 인터 예측을 위해 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)로 또한 전송된다. 구획된 비디오 신호(201)의 프레임 또는 슬라이스는 다수의 비디오 블록으로 분할될 수도 있다. 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)는, 시간적 예측을 제공하기 위해, 하나 이상의 참조 프레임의 하나 이상의 블록에 대해 수신된 비디오 블록의 인터 예측 코딩을 수행한다. 코덱 시스템(200)은, 예를 들면, 비디오 데이터의 각각의 블록에 대한 적절한 코딩 모드를 선택하기 위해, 다수의 코딩 패스(coding pass)를 수행할 수도 있다.

[0055] 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)는 고도로 통합될 수도 있지만, 그러나 개념적 목적을 위해 개별적으로 예시된다. 모션 추정 컴포넌트(221)에 의해 수행되는 모션 추정은, 비디오 블록에 대한 모션을 추정하는 모션 벡터를 생성하는 프로세스이다. 모션 벡터는, 예를 들면, 예측 블록에 대한 코딩된 오브젝트의 변위를 나타낼 수도 있다. 예측 블록은, 픽셀 차이의 관점에서, 코딩될 블록과 밀접하게 매치하는 것으로 밝혀지는 블록이다. 예측 블록은 참조 블록으로서 또한 지칭될 수도 있다. 그러한 픽셀 차이는 절대 차이의 합(sum of absolute difference; SAD), 제곱 차이의 합(sum of square difference; SSD), 또는 다른 차이 메트릭에 의해 결정될 수도 있다. HEVC는 CTU, 코딩 트리 블록(CTB), 및 CU를 포함하는 여러 가지 코딩된 오브젝트를 활용한다. 예를 들면, CTU는 CTB로 분할될 수 있는데, CTB는, 그 다음, CU에서의 포함을 위해 CB로 분할될 수 있다. CU는 예측 데이터를 포함하는 예측 단위(prediction unit; PU) 및/또는 CU에 대한 변환된 잔차 데이터를 포함하는 변환 단위(transform unit; TU)로서 인코딩될 수 있다. 모션 추정 컴포넌트(221)는, 레이트 왜곡 최적화 프로세스의 일부로서 레이트 왜곡 분석을 사용하는 것에 의해 모션 벡터, PU, 및 TU를 생성한다. 예를 들면, 모션 추정 컴포넌트(221)는 현재 블록/프레임에 대한 다수의 참조 블록, 다수의 모션 벡터, 등등을 결정할 수도 있고, 최상의 레이트 왜곡 특성을 갖는 참조 블록, 모션 벡터, 등등을 선택할 수도 있다. 최상의 레이트 왜곡 특성은, 비디오 재구성의 품질(예를 들면, 압축에 의한 데이터 손실의 양) 및 코딩 효율성(예를 들면, 최종 인코딩의 사이즈) 둘 모두의 균형을 유지한다.

[0056] 몇몇 예에서, 코덱 시스템(200)은 디코딩된 픽쳐 버퍼 컴포넌트(223)에 저장되는 참조 픽쳐의 정수 미만(sub-integer) 픽셀 포지션에 대한 값을 계산할 수도 있다. 예를 들면, 비디오 코덱 시스템(200)은 참조 픽쳐의 1/4 픽셀 포지션, 1/8 픽셀 포지션, 또는 다른 분수(fractional) 픽셀 포지션의 값을 보간할 수도 있다. 따라서, 모션 추정 컴포넌트(221)는 전체 픽셀 포지션 및 분수 픽셀 포지션에 대한 모션 검색을 수행할 수도 있고 분수 픽셀 정밀도를 갖는 모션 벡터를 출력할 수도 있다. 모션 추정 컴포넌트(221)는, PU의 포지션을 참조 픽쳐의 예측 블록의 포지션에 비교하는 것에 의해, 인터 코딩된 슬라이스에서 비디오 블록의 PU에 대한 모션 벡터를 계산한다. 모션 추정 컴포넌트(221)는 계산된 모션 벡터를 모션 데이터로서, 인코딩을 위해, 헤더 포맷팅 및 CABAC 컴포넌트(231)에 출력하고 모션을 모션 보상 컴포넌트(219)로 출력한다.

[0057] 모션 보상 컴포넌트(219)에 의해 수행되는 모션 보상은 모션 추정 컴포넌트(221)에 의해 결정되는 모션 벡터에 기초하여 예측 블록을 페치하는 것 또는 생성하는 것을 수반할 수도 있다. 다시, 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)는, 몇몇 예에서, 기능적으로 통합될 수도 있다. 현재 비디오 블록의 PU에 대한 모션 벡터를 수신하면, 모션 보상 컴포넌트(219)는 모션 벡터가 가리키는 예측 블록의 위치를 알아낼 수도 있다. 그 다음, 코딩되고 있는 현재 비디오 블록의 픽셀 값으로부터 예측 블록의 픽셀 값을 감산하여 픽셀 차이 값을 형성하는 것에 의해 잔차 비디오 블록이 형성된다. 일반적으로, 모션 추정 컴포넌트(221)는 루마 성분에 대한 모션 추정을 수행하고, 모션 보상 컴포넌트(219)는 크로마 성분 및 루마 성분 둘 모두에 대해 루마 성분에 기초하여 계산되는 모션 벡터를 사용한다. 예측 블록 및 잔차 블록은 변환 스케일링 및 양자화 컴포넌트(213)로 포워딩된다.

[0058] 구획된 비디오 신호(201)는 인트라 픽쳐 추정 컴포넌트(215) 및 인트라 픽쳐 예측 컴포넌트(217)로 또한 전송된다. 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)에서와 같이, 인트라 픽쳐 추정 컴포넌트(215) 및 인트라 픽쳐 예측 컴포넌트(217)는 고도로 통합될 수도 있지만, 그러나 개념적 목적을 위해 별개로 예시된다. 인트라 픽쳐 추정 컴포넌트(215) 및 인트라 픽쳐 예측 컴포넌트(217)는, 상기에서 설명되는 바와 같이, 프레임 사이에서 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)에 의해 수행되는 인터 예측에 대한 대안으로서, 현재 프레임의 블록에 대해 현재 블록을 인트라 예측한다. 특히, 인트라 픽쳐 추정 컴포넌트(215)는 현재 블록을 인코딩하기 위해 사용할 인트라 예측 모드를 결정한다. 몇몇 예에서, 인트라 픽쳐 추정 컴포넌트(215)는 다수의 테스트된 인트라 예측 모드로부터 현재 블록을 인코딩하기 위해 적절한 인트라 예측 모드를 선택한다. 그 다음, 선택된 인트라 예측 모드는 인코딩을 위해 헤더 포맷팅 및 CABAC 컴포넌트(231)로 포워딩된다.

[0059] 예를 들면, 인트라 픽쳐 추정 컴포넌트(215)는 다양한 테스트된 인트라 예측 모드에 대한 레이트 왜곡 분석을 사용하여 레이트 왜곡 값을 계산하고, 테스트된 모드 중에서 최상의 레이트 왜곡 특성을 갖는 인트라 예측 모드를 선택한다. 레이트 왜곡 분석은, 인코딩된 블록과 인코딩된 블록을 생성하기 위해 인코딩되었던 원래의 인코딩되지 않은 블록 사이의 왜곡(또는 에러)의 양뿐만 아니라, 인코딩된 블록을 생성하기 위해 사용되는 비트레이트(예를 들면, 비트의 수)를 일반적으로 결정한다. 어떤 인트라 예측 모드가 인트라 픽쳐 추정 컴포넌트(215)는 블록에 대해 최상의 레이트 왜곡 값을 나타내는지를 결정하기 위해 다양한 인코딩된 블록에 대한 왜곡 및 레이트로부터 비율을 계산한다. 또한, 인트라 픽쳐 추정 컴포넌트(215)는 레이트 왜곡 최적화(rate-distortion optimization; RDO)에 기초한 깊이 모델링 모드(depth modeling mode; DMM)를 사용하여 깊이 맵의 깊이 블록을 코딩하도록 구성될 수도 있다.

[0060] 인트라 픽쳐 예측 컴포넌트(217)는, 인코더 상에서 구현될 때 인트라 픽쳐 추정 컴포넌트(215)에 의해 결정되는 선택된 인트라 예측 모드에 기초하여 예측 블록으로부터 잔차 블록을 생성할 수도 있거나 또는 디코더 상에서 구현될 때 비트스트림으로부터 잔차 블록을 판독할 수도 있다. 잔차 블록은, 매트릭스로서 표현되는, 예측 블록과 원래의 블록 사이의 값에서의 차이를 포함한다. 그 다음, 잔차 블록은 변환 스케일링 및 양자화 컴포넌트(213)로 포워딩된다. 인트라 픽쳐 추정 컴포넌트(215) 및 인트라 픽쳐 예측 컴포넌트(217)는 루마 및 크로마 성분 둘 모두에 대해 동작할 수도 있다.

[0061] 변환 스케일링 및 양자화 컴포넌트(213)는 잔차 블록을 추가로 압축하도록 구성된다. 변환 스케일링 및 양자화 컴포넌트(213)는 이산 코사인 변환(discrete cosine transform; DCT), 이산 사인 변환(discrete sine transform; DST), 또는 개념적으로 유사한 변환과 같은 변환을 잔차 블록에 적용하여, 잔차 변환 계수 값을 포함하는 비디오 블록을 생성한다. 웨이블릿 변환(wavelet transform), 정수 변환(integer transform), 하위 대역 변환(sub-band transform) 또는 다른 타입의 변환이 또한 사용될 수 있다. 변환은 잔차 정보를 픽셀 값 도메인으로부터 주파수 도메인과 같은 변환 도메인으로 변환할 수도 있다. 변환 스케일링 및 양자화 컴포넌트(213)는, 예를 들면, 주파수에 기초하여, 변환된 잔차 정보를 스케일링하도록 또한 구성된다. 그러한 스케일링은, 상이한 주파수 정보가 상이한 세분성(granularity)에서 양자화되도록 잔차 정보에 스케일 팩터(scale factor)를 적용하는 것을 수반하는데, 이것은 재구성된 비디오의 최종 시각적 품질에 영향을 끼칠 수도 있다. 변환 스케일링 및 양자화 컴포넌트(213)는 비트 레이트를 추가로 감소시키기 위해 변환 계수를 양자화하도록 또한 구성된다. 양자화 프로세스는 계수의 일부 또는 모두와 관련되는 비트 깊이를 감소시킬 수도 있다. 양자화의 정도는 양자화 파라미터를 조정하는 것에 의해 수정될 수도 있다. 몇몇 예에서, 변환 스케일링 및 양자화 컴포넌트(213)는, 그 다음, 양자화된 변환 계수를 포함하는 매트릭스의 스캔을 수행할 수도 있다. 양자화된 변환 계수는 비트스트림에서 인코딩되도록 헤더 포맷팅 및 CABAC 컴포넌트(231)로 포워딩된다.

[0062] 스케일링 및 역변환 컴포넌트(229)는 모션 추정을 지원하기 위해 변환 스케일링 및 양자화 컴포넌트(213)의 역동작을 적용한다. 스케일링 및 역변환 컴포넌트(229)는, 예를 들면, 다른 현재 블록에 대한 예측 블록이 될 수도 있는 참조 블록으로서의 나중의 사용을 위해 잔차 블록을 픽셀 도메인에서 재구성하기 위해, 역 스케일링, 변환, 및/또는 양자화를 적용한다. 모션 추정 컴포넌트(221) 및/또는 모션 보상 컴포넌트(219)는, 나중의 블록/프레임의 모션 추정에서의 사용을 위해 대응하는 예측 블록에 잔차 블록을 다시 추가하는 것에 의해 참조 블록을 계산할 수도 있다. 필터는 스케일링, 양자화, 및 변환 동안 생성되는 아티팩트를 완화하기 위해 재구성된 참조 블록에 적용된다. 그렇지 않으면, 그러한 아티팩트는, 후속하는 블록이 예측될 때, 부정확한 예측을 야기할 수 있다(그리고 추가적인 아티팩트를 생성할 수도 있음).

[0063] 필터 제어 분석 컴포넌트(227) 및 루프내 필터 컴포넌트(225)는 필터를 잔차 블록 및/또는 재구성된 이미지 블록에 적용한다. 예를 들면, 스케일링 및 역변환 컴포넌트(229)로부터의 변환된 잔차 블록은 인트라 픽쳐 예측 컴포넌트(217) 및/또는 모션 보상 컴포넌트(219)로부터의 대응하는 예측 블록과 결합되어 원래의 이미지 블록을

재구성할 수도 있다. 그 다음 필터가 재구성된 이미지 블록에 적용될 수도 있다. 몇몇 예에서, 필터는, 대신, 잔차 블록에 적용될 수도 있다. 도 2의 다른 컴포넌트에서와 같이, 필터 제어 분석 컴포넌트(227) 및 루프내 필터 컴포넌트(225)는 고도로 통합되어 함께 구현될 수도 있지만, 그러나 개념적 목적을 위해 별개로 묘사된다. 재구성된 참조 블록에 적용되는 필터는 특정한 공간 영역에 적용되며 그러한 필터가 적용되는 방법을 조정하기 위해 다수의 파라미터를 포함한다. 필터 제어 분석 컴포넌트(227)는 그러한 필터가 적용되어야 하는 곳을 결정하기 위해 재구성된 참조 블록을 분석하고 대응하는 파라미터를 설정한다. 그러한 데이터는 인코딩을 위한 필터 제어 데이터로서 헤더 포맷팅 및 CABAC 컴포넌트(231)로 포워딩된다. 루프내 필터 컴포넌트(225)는 필터 제어 데이터에 기초하여 그러한 필터를 적용한다. 필터는 블록화 제거(deblocking) 필터, 노이즈 억제 필터, SAO 필터, 및 적응적 루프 필터를 포함할 수도 있다. 그러한 필터는, 예에 따라, 공간/픽셀 도메인에서 (예를 들면, 재구성된 픽셀 블록에 대해) 또는 주파수 도메인에서 적용될 수도 있다.

[0064] 인코더로서 동작할 때, 필터링된 재구성된 이미지 블록, 잔차 블록, 및/또는 예측 블록은, 상기에서 논의되는 바와 같이 모션 추정에서의 나중의 사용을 위해, 디코딩된 픽쳐 버퍼 컴포넌트(223)에 저장된다. 디코더로서 동작할 때, 디코딩된 픽쳐 버퍼 컴포넌트(223)는 재구성되고 필터링된 블록을 저장하고 출력 비디오 신호의 일부로서 디스플레이를 향해 포워딩한다. 디코딩된 픽쳐 버퍼 컴포넌트(223)는 예측 블록, 잔차 블록, 및/또는 재구성된 이미지 블록을 저장할 수 있는 임의의 메모리 디바이스일 수도 있다.

[0065] 헤더 포맷팅 및 CABAC 컴포넌트(231)는 코덱 시스템(200)의 다양한 컴포넌트로부터 데이터를 수신하고 디코더를 향한 송신을 위해 그러한 데이터를 코딩된 비트스트림에 인코딩한다. 구체적으로, 헤더 포맷팅 및 CABAC 컴포넌트(231)는, 일반 제어 데이터 및 필터 제어 데이터와 같은 제어 데이터를 인코딩하기 위해 다양한 헤더를 생성한다. 게다가, 인트라 예측 및 모션 데이터를 비롯한 예측 데이터뿐만 아니라, 양자화된 변환 계수 데이터 형태의 잔차 데이터가 모두 비트스트림에서 인코딩된다. 최종 비트스트림은 원래의 구획된 비디오 신호(201)를 재구성하기 위해 디코더에 의해 소망되는 모든 정보를 포함한다. 그러한 정보는 인트라 예측 모드 인덱스 테이블(코드워드 매핑 테이블로서 또한 지칭됨), 다양한 블록에 대한 인코딩 컨텍스트의 정의, 가장 가능성 있는 인트라 예측 모드의 지시, 구획 정보의 지시, 등등을 또한 포함할 수도 있다. 그러한 데이터는 엔트로피 코딩을 활용하는 것에 의해 인코딩될 수도 있다. 예를 들면, 정보는, 컨텍스트 적응 가변 길이 코딩(context adaptive variable length coding; CAVLC), CABAC, 신택스 기반의 컨텍스트 적응 이진 산술 코딩(syntax-based context-adaptive binary arithmetic coding; SBAC), 확률 간격 구획화 엔트로피(probability interval partitioning entropy; PIPE) 코딩, 또는 다른 엔트로피 코딩 기술을 활용하는 것에 의해 인코딩될 수도 있다. 엔트로피 코딩에 후속하여, 코딩된 비트스트림은 다른 디바이스(예를 들면, 비디오 디코더)로 송신될 수도 있거나 또는 나중의 송신 또는 검색을 위해 보관될(archived) 수도 있다.

[0066] 도 3은 예시적인 비디오 인코더(300)를 예시하는 블록도이다. 비디오 인코더(300)는 코덱 시스템(200)의 인코딩 기능을 구현하도록 및/또는 동작 방법(100)의 단계(101, 103, 105, 107, 및/또는 109)를 구현하도록 활용될 수도 있다. 인코더(300)는 입력 비디오 신호를 구획하는데, 구획된 비디오 신호(201)와 실질적으로 유사한 구획된 비디오 신호(301)로 나타나게 된다. 그 다음, 구획된 비디오 신호(301)는 인코더(300)의 컴포넌트에 의해 압축되어 비트스트림에 인코딩된다.

[0067] 구체적으로, 구획된 비디오 신호(301)는 인트라 예측을 위해 인트라 픽쳐 예측 컴포넌트(317)로 포워딩된다. 인트라 픽쳐 예측 컴포넌트(317)는 인트라 픽쳐 추정 컴포넌트(215) 및 인트라 픽쳐 예측 컴포넌트(217)와 실질적으로 유사할 수도 있다. 구획된 비디오 신호(301)는, 디코딩된 픽쳐 버퍼 컴포넌트(323)의 참조 블록에 기초한 인터 예측을 위해 모션 보상 컴포넌트(321)로 또한 포워딩된다. 모션 보상 컴포넌트(321)는 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)와 실질적으로 유사할 수도 있다. 인트라 픽쳐 예측 컴포넌트(317) 및 모션 보상 컴포넌트(321)로부터의 예측 블록 및 잔차 블록은 잔차 블록의 변환 및 양자화를 위해 변환 및 양자화 컴포넌트(313)로 포워딩된다. 변환 및 양자화 컴포넌트(313)는 변환 스케일링 및 양자화 컴포넌트(213)와 실질적으로 유사할 수도 있다. 변환되고 양자화된 잔차 블록 및 대응하는 예측 블록은 (관련된 제어 데이터와 함께) 비트스트림으로의 코딩을 위해 엔트로피 코딩 컴포넌트(331)로 포워딩된다. 엔트로피 코딩 컴포넌트(331)는 헤더 포맷팅 및 CABAC 컴포넌트(231)와 실질적으로 유사할 수도 있다.

[0068] 변환되고 양자화된 잔차 블록 및/또는 대응하는 예측 블록은, 모션 보상 컴포넌트(321)에 의한 사용을 위한 참조 블록으로의 재구성을 위해 변환 및 양자화 컴포넌트(313)로부터 역변환 및 양자화 컴포넌트(329)로 또한 포워딩된다. 역변환 및 양자화 컴포넌트(329)는 스케일링 및 역변환 컴포넌트(229)와 실질적으로 유사할 수도 있다. 루프내 필터 컴포넌트(325)의 루프내 필터는, 예에 따라, 잔차 블록 및/또는 재구성된 참조 블록에도 또한 적용된다. 루프내 필터 컴포넌트(325)는 필터 제어 분석 컴포넌트(227) 및 루프내 필터 컴포넌트(225)와 실질적

으로 유사할 수도 있다. 루프내 필터 컴포넌트(325)는 루프내 필터 컴포넌트(225)와 관련하여 논의되는 바와 같이 다수의 필터를 포함할 수도 있다. 필터링된 블록은, 그 다음, 모션 보상 컴포넌트(321)에 의한 참조 블록으로서의 사용을 위해 디코딩된 픽쳐 버퍼 컴포넌트(323)에 저장된다. 디코딩된 픽쳐 버퍼 컴포넌트(323)는 디코딩된 픽쳐 버퍼 컴포넌트(223)와 실질적으로 유사할 수도 있다.

[0069] 도 4는 예시적인 비디오 디코더(400)를 예시하는 블록도이다. 비디오 디코더(400)는 코덱 시스템(200)의 디코딩 기능을 구현하도록 및/또는 동작 방법(100)의 단계(111, 113, 115, 및/또는 117)를 구현하도록 활용될 수도 있다. 디코더(400)는, 예를 들면 인코더(300)로부터, 비트스트림을 수신하고, 엔드 유저에 대한 디스플레이를 위해 비트스트림에 기초하여 재구성된 출력 비디오 신호를 생성한다.

[0070] 비트스트림은 엔트로피 디코딩 컴포넌트(433)에 의해 수신된다. 엔트로피 디코딩 컴포넌트(433)는, CAVLC, CABAC, SBAC, PIPE 코딩, 또는 다른 엔트로피 코딩 기술과 같은 엔트로피 디코딩 스킴을 구현하도록 구성된다. 예를 들면, 엔트로피 디코딩 컴포넌트(433)는, 비트스트림에서 코드워드로서 인코딩되는 추가적인 데이터를 해석하기 위한 컨텍스트를 제공하기 위해, 헤더 정보를 활용할 수도 있다. 디코딩된 정보는 비디오 신호를 디코딩하기 위해 임의의 소망되는 정보, 예컨대 일반 제어 데이터, 필터 제어 데이터, 구획 정보, 모션 데이터, 예측 데이터, 및 잔차 블록으로부터의 양자화된 변환 계수를 포함한다. 양자화된 변환 계수는 잔차 블록으로의 재구성을 위해 역변환 및 양자화 컴포넌트(429)로 포워딩된다. 역변환 및 양자화 컴포넌트(429)는 역변환 및 양자화 컴포넌트(329)와 유사할 수도 있다.

[0071] 재구성된 잔차 블록 및/또는 예측 블록은 인트라 예측 동작에 기초한 이미지 블록으로의 재구성을 위해 인트라 픽쳐 예측 컴포넌트(417)로 포워딩된다. 인트라 픽쳐 예측 컴포넌트(417)는 인트라 픽쳐 추정 컴포넌트(215) 및 인트라 픽쳐 예측 컴포넌트(217)와 유사할 수도 있다. 구체적으로, 인트라 픽쳐 예측 컴포넌트(417)는 프레임에서 참조 블록을 찾기 위해 예측 모드를 활용하고 인트라 예측된 이미지 블록을 재구성하기 위해 결과에 잔차 블록을 적용한다. 재구성된 인트라 예측된 이미지 블록 및/또는 잔차 블록 및 대응하는 인터 예측 데이터는 루프내 필터 컴포넌트(425)를 통해 디코딩된 픽쳐 버퍼 컴포넌트(423)로 포워딩되는데, 이들은 루프내 필터 컴포넌트(225) 및 디코딩된 픽쳐 버퍼 컴포넌트(223)와, 각각, 실질적으로 유사할 수도 있다. 루프내 필터 컴포넌트(425)는 재구성된 이미지 블록, 잔차 블록 및/또는 예측 블록을 필터링하고, 그러한 정보는 디코딩된 픽쳐 버퍼 컴포넌트(423)에 저장된다. 디코딩된 픽쳐 버퍼 컴포넌트(423)로부터의 재구성된 이미지 블록은 인터 예측을 위해 모션 보상 컴포넌트(421)로 포워딩된다. 모션 보상 컴포넌트(421)는 모션 추정 컴포넌트(221) 및/또는 모션 보상 컴포넌트(219)와 실질적으로 유사할 수도 있다. 구체적으로, 모션 보상 컴포넌트(421)는 참조 블록으로부터의 모션 벡터를 활용하여 예측 블록을 생성하고 잔차 블록을 결과에 적용하여 이미지 블록을 재구성한다. 결과적으로 나타나는 재구성된 블록은 루프내 필터 컴포넌트(425)를 통해 디코딩된 픽쳐 버퍼 컴포넌트(423)로 또한 포워딩될 수도 있다. 디코딩된 픽쳐 버퍼 컴포넌트(423)는, 구획 정보를 통해 프레임으로 재구성될 수 있는 추가적인 재구성된 이미지 블록을 계속 저장한다. 그러한 프레임은 시퀀스에 또한 배치될 수도 있다. 시퀀스는 재구성된 출력 비디오 신호로서 디스플레이를 향해 출력된다.

[0072] 도 5는 예시적인 HRD(500)를 예시하는 개략도이다. HRD(500)는 코덱 시스템(200) 및/또는 인코더(300)와 같은 인코더에서 활용될 수도 있다. HRD(500)는 비트스트림이 디코더(400)와 같은 디코더로 포워딩되기 이전에 방법(100)의 단계(109)에서 생성되는 비트스트림을 검사할 수도 있다. 몇몇 예에서, 비트스트림은 비트스트림이 인코딩됨에 따라 HRD(500)를 통해 연속적으로 포워딩될 수도 있다. 비트스트림의 일부가 관련된 제약을 준수하는 데 실패하는 경우, HRD(500)는, 인코더로 하여금 상이한 메커니즘을 사용하여 비트스트림의 대응하는 섹션을 재인코딩하게 하기 위해 그러한 실패를 인코더에게 나타낼 수 있다.

[0073] HRD(500)는 가상 스트림 스케줄러(HSS)(541)를 포함한다. HSS(541)는 가상 전달 메커니즘을 수행하도록 구성되는 컴포넌트이다. 가상 전달 메커니즘은 HRD(500)에 입력되는 비트스트림(551)의 타이밍 및 데이터 흐름과 관련하여 비트스트림 또는 디코더의 적합성을 검사하기 위해 사용된다. 예를 들면, HSS(541)는 인코더로부터 출력되는 비트스트림(551)을 수신할 수도 있고, 비트스트림(551)에 대한 적합성 테스팅 프로세스를 관리할 수도 있다. 특정한 예에서, HSS(541)는 코딩된 픽쳐가 HRD(500)를 통해 이동하는 레이트를 제어할 수 있고 비트스트림(551)이 비적합 데이터를 포함하지 않는다는 것을 검증할 수 있다.

[0074] HSS(541)는 비트스트림(551)을 사전 정의된 레이트에서 CPB(543)에 포워딩할 수도 있다. HRD(500)는 데이터를 디코딩 단위(DU)(553)로 관리할 수도 있다. DU(553)는 액세스 단위(AU) 또는 AU 및 관련된 비 비디오 코딩 레이어(VCL) 네트워크 추상화 레이어(NAL) 단위의 서브세트이다. 구체적으로, AU는 출력 시간과 관련되는 하나 이상의 픽쳐를 포함한다. 예를 들면, AU는 단일 레이어 비트스트림에 단일의 픽쳐를 포함할 수도 있고, 다중 레이어

비트스트림에 각각의 레이어에 대한 픽쳐를 포함할 수도 있다. AU의 각각의 픽쳐는 대응하는 VCL NAL 단위에 각각 포함되는 슬라이스로 분할될 수도 있다. 그러므로, DU(553)는 하나 이상의 픽쳐, 픽쳐의 하나 이상의 슬라이스, 또는 이들의 조합을 포함할 수도 있다. 또한, AU/DU, 픽쳐, 및/또는 슬라이스를 디코딩하기 위해 사용되는 파라미터가 비 VCL NAL 단위에 포함될 수 있다. 그러한 만큼, DU(553)는, DU(553)에서 VCL NAL 단위의 디코딩을 지원하는 데 필요한 데이터를 포함하는 비 VCL NAL 단위를 포함한다. CPB(543)는 HRD(500)의 선입선출(first-in first-out) 버퍼이다. CPB(543)는 디코딩 순서대로 비디오 데이터를 포함하는 DU(553)를 포함한다. CPB(543)는 비트스트림 적합성 검증 동안의 사용을 위해 비디오 데이터를 저장한다.

[0075] CPB(543)는 DU(553)를 디코딩 프로세스 컴포넌트(545)로 포워딩한다. 디코딩 프로세스 컴포넌트(545)는 VVC 표준을 준수하는 컴포넌트이다. 예를 들면, 디코딩 프로세스 컴포넌트(545)는 엔드 유저에 의해 활용되는 디코더(400)를 에뮬레이팅할 수도 있다. 디코딩 프로세스 컴포넌트(545)는, 예시적인 엔드 유저 디코더에 의해 달성될 수 있는 레이트에서 DU(553)를 디코딩한다. 디코딩 프로세스 컴포넌트(545)가 CPB(543)의 오버플로우를 방지할(또는 버퍼 언더런을 방지할) 만큼 충분히 빠르게 DU(553)를 디코딩할 수 없는 경우, 그러면, 비트스트림(551)은 표준을 준수하지 않으며 재인코딩되어야 한다.

[0076] 디코딩 프로세스 컴포넌트(545)는 DU(553)를 디코딩하는데, 이것은 디코딩된 DU(555)를 생성한다. 디코딩된 DU(555)는 디코딩된 픽쳐를 포함한다. 디코딩된 DU(555)는 DPB(547)로 포워딩된다. DPB(547)는 디코딩된 픽쳐 버퍼 컴포넌트(223, 323, 및/또는 423)와 실질적으로 유사할 수도 있다. 인터 예측을 지원하기 위해, 디코딩된 DU(555)로부터 획득되는 참조 픽쳐(556)로서의 사용을 위해 마킹되는 픽쳐는 추가적인 디코딩을 지원하기 위해 디코딩 프로세스 컴포넌트(545)로 리턴된다. DPB(547)는 디코딩된 비디오 시퀀스를 일련의 픽쳐(557)로서 출력한다. 픽쳐(557)는, 인코더에 의해 비트스트림(551)으로 인코딩된 픽쳐를 일반적으로 미러링하는 재구성된 픽쳐이다.

[0077] 픽쳐(557)는 출력 크롭핑 컴포넌트(output cropping component; 549)로 포워딩된다. 출력 크롭핑 컴포넌트(549)는 픽쳐(557)에 적합성 크롭핑 윈도우를 적용하도록 구성된다. 이것은 출력 크롭 픽쳐(output cropped picture; 559)로 귀결된다. 출력 크롭 픽쳐(559)는 완전히 재구성된 픽쳐이다. 따라서, 출력 크롭 픽쳐(559)는, 비트스트림(551)을 디코딩할 때 엔드 유저가 보게 될 것을 모방한다. 그러한 만큼, 인코더는 인코딩이 만족되는 것을 보장하기 위해 출력 크롭 픽쳐(559)를 재검토할(review) 수 있다.

[0078] HRD(500)는 비트스트림(551)의 HRD 파라미터에 기초하여 초기화된다. 예를 들면, HRD(500)는 VPS, SPS, 및/또는 SEI 메시지로부터 HRD 파라미터를 판독할 수도 있다. 그 다음, HRD(500)는 그러한 HRD 파라미터의 정보에 기초하여 비트스트림(551)에 대해 적합성 테스팅 동작을 수행할 수도 있다. 특정한 예로서, HRD(500)는 HRD 파라미터로부터 하나 이상의 CPB 전달 스케줄을 결정할 수도 있다. 전달 스케줄은, CPB 및/또는 DPB와 같은 기억 장소로의 및/또는 기억 장소로부터의 비디오 데이터의 전달을 위한 타이밍을 명시한다. 그러므로, CPB 전달 스케줄은 CPB(543)로의/로부터의, AU, DU(553), 및/또는 픽쳐의 전달을 위한 타이밍을 명시한다. HRD(500)는, CPB 전달 스케줄과 유사한 DPB 전달 스케줄을 DPB(547)에 대해 활용할 수도 있다는 것을 유의해야 한다.

[0079] 비디오는 다양한 레벨의 하드웨어 성능을 갖는 디코더에 의한 사용을 위해, 뿐만 아니라 다양한 네트워크 조건에 대해 상이한 레이어 및/또는 OLS로 코딩될 수도 있다. CPB 전달 스케줄은 이들 문제를 반영하도록 선택된다. 따라서, 최적의 하드웨어 및 네트워크 조건을 위해 상위 레이어(higher layer) 서브 비트스트림이 지정되고, 그러므로, 상위 레이어는 DPB(547)를 향한 DU(553)의 전달을 위한 짧은 지연 및 CPB(543)의 많은 양의 메모리를 활용하는 하나 이상의 CPB 전달 스케줄을 수신할 수도 있다. 마찬가지로, 하위 레이어(lower layer) 서브 비트스트림은 제한된 디코더 하드웨어 성능 및/또는 불량한 네트워크 조건에 대해 지정된다. 그러므로, 하위 레이어는 DPB(547)를 향한 DU(553)의 전달을 위한 긴 지연 및 CPB(543)의 적은 양의 메모리를 활용하는 하나 이상의 CPB 전달 스케줄을 수신할 수도 있다. 그 다음, OLS, 레이어, 서브 레이어, 또는 이들의 조합은, 서브 비트스트림에 대해 예상되는 조건 하에서 결과적으로 나타나는 서브 비트스트림이 올바르게 디코딩될 수 있는 것을 보장하기 위해, 대응하는 전달 스케줄에 따라 테스트될 수 있다. 따라서, 비트스트림(551)의 HRD 파라미터는 CPB 전달 스케줄을 나타낼 수 있고, 뿐만 아니라, HRD(500)가 CPB 전달 스케줄을 결정하는 것 및 CPB 전달 스케줄을 대응하는 OLS, 레이어, 및/또는 서브 레이어에 상관시키는 것을 허용하기에 충분한 데이터를 포함할 수 있다.

[0080] 도 6은 인터레이어 예측(621)을 위해 구성되는 예시적인 다중 레이어 비디오 시퀀스(600)를 예시하는 개략도이다. 다중 레이어 비디오 시퀀스(600)는, 예를 들면, 방법(100)에 따라, 인코더, 예컨대 코덱 시스템(200) 및/또는 인코더(300)에 의해 인코딩될 수도 있고 디코더, 예컨대 코덱 시스템(200) 및/또는 디코더(400)에 의해 디코딩될 수도 있다. 게다가, 다중 레이어 비디오 시퀀스(600)는 HRD(500)와 같은 HRD에 의해 표준 적합성에 대해

검사될 수 있다. 다중 레이어 비디오 시퀀스(600)는 코딩된 비디오 시퀀스의 레이어에 대한 예시적인 적용을 묘사하기 위해 포함된다. 다중 레이어 비디오 시퀀스(600)는 레이어 N(631) 및 레이어 N+1(632)과 같은 복수의 레이어를 활용하는 임의의 비디오 시퀀스이다.

[0081] 한 예에서, 다중 레이어 비디오 시퀀스(600)는 인터레이어 예측(621)을 활용할 수도 있다. 인터레이어 예측(621)은 상이한 레이어의 픽처(611, 612, 613, 및 614)와 픽처(615, 616, 617, 및 618) 사이에서 적용된다. 도시되는 예에서, 픽처(611, 612, 613, 및 614)는 레이어 N+1(632)의 일부이고 픽처(615, 616, 617, 및 618)는 레이어 N(631)의 일부이다. 레이어 N(631) 및/또는 레이어 N+1(632)과 같은 레이어는, 유사한 사이즈, 품질, 해상도, 신호 대 노이즈 비율, 성능, 등등과 같은 특성의 유사한 값과 모두 관련되는 픽처의 그룹이다. 레이어는 VCL NAL 단위 및 관련된 비 VCL NAL 단위의 세트로서 공식적으로서 정의될 수도 있다. VCL NAL 단위는 픽처의 코딩된 슬라이스와 같은 비디오 데이터를 포함하도록 코딩되는 NAL 단위이다. 비 VCL NAL 단위는, 비디오 데이터의 디코딩, 적합성 검사의 수행, 또는 다른 동작을 지원하는 신택스 및/또는 파라미터와 같은 비 비디오 데이터를 포함하는 NAL 단위이다.

[0082] 도시되는 예에서, 레이어 N+1(632)은 레이어 N(631)보다 더 큰 이미지 사이즈와 관련된다. 따라서, 레이어 N+1(632)의 픽처(611, 612, 613, 및 614)는, 이 예에서, 레이어 N(631)의 픽처(615, 616, 617, 및 618)보다 더 큰 픽처 사이즈(예를 들면, 더 큰 높이와 폭 그러므로 더 많은 샘플)를 갖는다. 그러나, 그러한 픽처는 다른 특성에 의해 N+1 레이어(632)와 N 레이어(631) 사이에서 분리될 수 있다. 단지 두 개의 레이어인 레이어 N+1(632) 및 레이어 N(631)만이 도시되지만, 픽처의 세트는 관련된 특성에 기초하여 임의의 개수의 레이어로 분리될 수 있다. 레이어 N+1(632) 및 레이어 N(631)은 레이어 Id에 의해 또한 나타내어질 수도 있다. 레이어 Id는 픽처와 관련되며 픽처가 나타내어진 레이어의 일부이다는 것을 나타내는 데이터의 아이템이다. 따라서, 각각의 픽처(611-618)는, 어떤 레이어 N+1(632) 또는 레이어 N(631)가 대응하는 픽처를 포함하는지를 나타내기 위해 대응하는 레이어 Id와 관련될 수도 있다. 예를 들면, 레이어 Id는, NAL 단위를 포함하는(예를 들면, 레이어의 픽처의 슬라이스 및/또는 파라미터를 포함하는) 레이어의 식별자를 명시하는 신택스 엘리먼트인 NAL 단위 헤더 레이어 식별자(nuh_layer_id)를 포함할 수도 있다. 레이어 N(631)과 같은, 더 낮은 품질/비트스트림 사이즈와 관련되는 레이어는 하위 레이어(lower layer) Id를 일반적으로 할당받고 하위 레이어로서 지칭된다. 게다가, 레이어 N+1(632)과 같은, 더 높은 품질/비트스트림 사이즈와 관련되는 레이어는 상위 레이어(higher layer) Id를 일반적으로 할당받고 상위 레이어로서 지칭된다.

[0083] 상이한 레이어(631-632)의 픽처(611-618)는 대안예에서 디스플레이되도록 구성된다. 구체적인 예로서, 디코더는 더 작은 픽처가 소망되는 경우 현재 디스플레이 시간에 픽처(615)를 디코딩하여 디스플레이할 수도 있거나 또는 디코더는 더 큰 픽처가 소망되는 경우 현재 디스플레이 시간에 픽처(611)를 디코딩 및 디스플레이할 수도 있다. 그러한 만큼, 상위 레이어 N+1(632)의 픽처(611-614)는 (픽처 사이즈에서의 차이에도 불구하고) 하위 레이어 N(631)의 대응하는 픽처(615-618)와 실질적으로 동일한 이미지 데이터를 포함한다. 구체적으로, 픽처(611)는 픽처(615)와 실질적으로 동일한 이미지 데이터를 포함하고, 픽처(612)는 픽처(616)와 실질적으로 동일한 이미지 데이터를 포함하고, 등등이다.

[0084] 픽처(611-618)는 동일한 레이어 N(631) 또는 N+1(632)의 다른 픽처(611-618)에 대한 참조에 의해 코딩될 수 있다. 동일한 레이어의 다른 픽처에 대한 참조에 의해 픽처를 코딩하는 것은 인터 예측(623)으로 귀결된다. 인터 예측(623)은 실선 화살표에 의해 묘사된다. 예를 들면, 픽처(613)는 참조로서 레이어 N+1(632)의 픽처(611, 612, 및/또는 614) 중 한 개 또는 두 개를 사용하여 인터 예측(623)을 활용하는 것에 의해 코딩될 수도 있는데, 여기서 하나의 픽처는 단방향 인터 예측을 위해 참조되고 및/또는 두 개의 픽처는 양방향 인터 예측을 위해 참조된다. 게다가, 픽처(617)는 참조로서 레이어 N(631)의 픽처(615, 616, 및/또는 618) 중 한 개 또는 두 개를 사용하여 인터 예측(623)을 활용하는 것에 의해 코딩될 수도 있는데, 여기서 하나의 픽처는 단방향 인터 예측을 위해 참조되고 및/또는 두 개의 픽처는 양방향 인터 예측을 위해 참조된다. 인터 예측(623)을 수행할 때 픽처가 동일한 레이어의 다른 픽처에 대한 참조로서 사용되는 경우, 픽처는 참조 픽처로서 지칭될 수도 있다. 예를 들면, 픽처(612)는 인터 예측(623)에 따라 픽처(613)를 코딩하기 위해 사용되는 참조 픽처일 수도 있다. 인터 예측(623)은 다중 레이어 컨텍스트에서 인트라 레이어 예측으로 또한 지칭될 수 있다. 그러한 만큼, 인터 예측(623)은 현재 픽처와는 상이한 참조 픽처의 지시된 샘플(indicated sample)에 대한 참조에 의해 현재 픽처의 샘플을 코딩하는 메커니즘인데, 여기서 참조 픽처 및 현재 픽처는 동일한 레이어에 있다.

[0085] 픽처(611-618)는 상이한 레이어의 다른 픽처(611-618)에 대한 참조에 의해 또한 코딩될 수 있다. 이 프로세스는 인터레이어 예측(621)으로서 공지되어 있으며, 파선의 화살표에 의해 묘사된다. 인터레이어 예측(621)은 참조 픽처의 지시된 샘플에 대한 참조에 의해 현재 픽처의 샘플을 코딩하는 메커니즘인데, 여기서 현재 픽처 및 참조

픽쳐는 상이한 레이어에 있고 그러므로 상이한 레이어 ID를 갖는다. 예를 들면, 하위 레이어 N(631)의 픽쳐는 상위 레이어 N+1(632)의 대응 픽쳐를 코딩하기 위한 참조 픽쳐로서 사용될 수 있다. 구체적인 예로서, 픽쳐(611)는 인터레이어 예측(621)에 따른 픽쳐(615)에 대한 참조에 의해 코딩될 수 있다. 그러한 경우, 픽쳐(615)는 인터레이어 참조 픽쳐로서 사용된다. 인터레이어 참조 픽쳐는 인터레이어 예측(621)을 위해 사용되는 참조 픽쳐이다. 대부분의 경우, 인터레이어 예측(621)은, 픽쳐(611)와 같은 현재 픽쳐가, 동일한 AU에 포함되는 그리고 하위 레이어에 있는 인터레이어 참조 픽쳐(들), 예컨대 픽쳐(615)만을 사용할 수 있도록 제한된다. 다수의 레이어(예를 들면, 두 개보다 더 많음)가 이용 가능한 경우, 인터레이어 예측(621)은 현재 픽쳐보다 더 낮은 레벨에서 다수의 인터레이어 참조 픽쳐(들)에 기초하여 현재 픽쳐를 인코딩/디코딩할 수 있다.

[0086] 비디오 인코더는 인터레이어 예측(623) 및 인터레이어 예측(621)의 많은 상이한 조합 및/또는 순열(permutation)을 통해 픽쳐(611-618)를 인코딩하기 위해 다중 레이어 비디오 시퀀스(600)를 활용할 수 있다. 예를 들면, 픽쳐(615)는 인트라 예측에 따라 코딩될 수도 있다. 그 다음, 픽쳐(616-618)는 참조 픽쳐로서 픽쳐(615)를 사용하는 것에 의해 인터 예측(623)에 따라 코딩될 수 있다. 게다가, 픽쳐(611)는 픽쳐(615)를 인터레이어 참조 픽쳐로서 사용하는 것에 의해 인터레이어 예측(621)에 따라 코딩될 수도 있다. 그 다음, 픽쳐(612-614)는 참조 픽쳐로서 픽쳐(611)를 사용하는 것에 의해 인터 예측(623)에 따라 코딩될 수 있다. 그러한 만큼, 참조 픽쳐는 상이한 코딩 메커니즘에 대한 단일의 레이어 참조 픽쳐 및 인터레이어 참조 픽쳐 둘 모두로서 역할을 할 수 있다. 하위 레이어 N(631) 픽쳐에 기초하여 상위 레이어 N+1(632) 픽쳐를 코딩하는 것에 의해, 상위 레이어 N+1(632)은, 인터 예측(623) 및 인터레이어 예측(621)보다 훨씬 더 낮은 코딩 효율성을 갖는 인트라 예측을 활용하는 것을 방지할 수 있다. 그러한 만큼, 인트라 예측의 불량한 코딩 효율성은 가장 작은/가장 낮은 품질 픽쳐로 제한될 수 있고,그러므로, 가장 적은 양의 비디오 데이터를 코딩하는 것으로 제한될 수 있다. 참조 픽쳐 및/또는 인터레이어 참조 픽쳐로서 사용되는 픽쳐는 참조 픽쳐 목록 구조에 포함되는 참조 픽쳐 목록(들)의 엔트리에서 나타내어질 수 있다.

[0087] 레이어 N+1(632) 및 레이어 N(631)과 같은 레이어가 출력 레이어 세트(OLS)에 포함될 수 있다는 것을 유의해야 한다. OLS는 하나 이상의 레이어의 세트인데, 여기서 적어도 하나의 레이어는 출력 레이어이다. 예를 들면, 레이어 N(631)은 제1 OLS에 포함될 수 있고 레이어 N(631) 및 레이어 N-1(632) 둘 모두는 제2 OLS에 포함될 수 있다. 이것은, 디코더 측 조건에 따라, 상이한 OLS가 상이한 디코더로 전송되는 것을 허용한다. 예를 들면, 서브 비트스트림 추출 프로세스는, 타깃 OLS가 디코더로 전송되기 이전에, 타깃 OLS에 관련되지 않는 데이터를 다중 레이어 비디오 시퀀스(600)로부터 제거할 수 있다. 그러한 만큼, 다중 레이어 비디오 시퀀스(600)의 인코딩된 사본은 인코더(또는 대응하는 콘텐츠 서버)에 저장될 수 있고, 다양한 OLS가 추출되어, 요청시, 상이한 디코더로 전송될 수 있다.

[0088] 또한, 레이어 N+1(632) 및 레이어 N(631)과 같은 상이한 레이어가 HRD 및/또는 디코더에서 상이한 메모리 요건과 관련될 수도 있다는 것을 유의해야 한다. 구체적으로, DPB의 디코딩된 픽쳐는, 장기간 참조용으로 사용됨, 단기간 참조용으로 사용됨, 또는 참조용으로 사용되지 않음으로 마킹될 수도 있다. 참조 픽쳐 마킹 프로세스는, 새로운 픽쳐가 디코딩될 때마다(예를 들면, 새로운 픽쳐가 현재 픽쳐가 될 때) 호출될 수도 있다. 구체적으로, 참조 픽쳐 마킹 프로세스는, 새로운 픽쳐에 디코딩 프로세스가 적용될 때마다 DPB의 각각의 픽쳐에 적용될 수도 있다. 참조용으로 사용되지 않음으로 마킹되는 픽쳐는, 픽쳐가 출력될 수 있을 때까지 저장되거나 또는 픽쳐가 출력을 위해 예약되지 않는 경우 DPB로부터 즉시 제거된다. 따라서, 레이어 N+1(632) 및/또는 레이어 N(631)과 같은 레이어, 즉 출력 레이어는 참조 픽쳐를 유지하기 위해 그리고 그러한 픽쳐가 출력될 수 있을 때까지 픽쳐를 유지하기 위해 DPB 공간을 사용한다. 그러나, 인터레이어 예측(621)을 위한 참조 레이어로서만 사용되는(그리고 출력 레이어가 아닌) 레이어 N+1(632) 및/또는 레이어 N(631)과 같은 레이어는 참조용으로 사용되는 픽쳐만을 유지한다. 출력 레이어가 아닌 레이어는, 비 출력 레이어(non-output layer)로부터의 픽쳐가 절대 출력되지 않기 때문에, 참조용으로 사용되지 않음으로 마킹되는 픽쳐를 유지할 필요가 없다. 그러한 만큼, 출력 레이어는 참조를 위해서만 사용되는 레이어(예를 들면, 비 출력 레이어)보다 더 많은 DPB 공간을 활용한다. 예를 들면, 출력 레이어는, DPB에서, 출력 레이어가 아닌 참조 레이어보다 약 두 배만큼 많은 메모리 공간을 활용할 수도 있다.

[0089] 도 7은 예시적인 참조 픽쳐 목록(reference picture list; RPL) 구조(700)를 예시하는 개략도이다. RPL 구조(700)는, 다중 레이어 비디오 시퀀스(600)와 같은 비디오 시퀀스를 코딩할 때, 인터 예측(623) 및/또는 인터레이어 예측(621)에서 사용되는 참조 픽쳐 및/또는 인터레이어 참조 픽쳐의 지시를 저장하기 위해 활용될 수 있다. 그러므로, RPL 구조(700)는, 방법(100)을 수행할 때, 코덱 시스템(200), 인코더(300), 및/또는 디코더(400)에 의해 활용될 수 있다. 게다가, RPL 구조(700)는, 인코딩된 비디오 시퀀스를 포함하는 비트스트림에 대

해 적합성 테스트를 수행할 때, HRD(500)와 같은 HRD에 의해 활용될 수 있다.

[0090] RPL 구조(700)는, 참조 픽쳐 목록 0(RefPicList[0])(711) 및 참조 픽쳐 목록 1(RefPicList[1])(712)과 같은 다수의 참조 픽쳐 목록을 포함하는 주소 지정 가능한 신택스 구조이다. RPL 구조(700)는, 예에 따라, 시퀀스 파라미터 세트(SPS), 픽쳐 파라미터 세트(PPS), 및/또는 비트스트림의 슬라이스 헤더에 저장될 수도 있다. RefPicList[0](711) 및 RefPicList[1](712)와 같은 참조 픽쳐 목록은, 인터 예측 및/또는 인터레이어 예측을 위해 사용되는 참조 픽쳐의 목록이다. RefPicList[0](711) 및 RefPicList[1](712) 각각은 복수의 엔트리(715)를 포함할 수도 있다. 참조 픽쳐 목록 구조 엔트리(715)는, RefPicList[0](711) 및/또는 RefPicList[1](712)와 같은 참조 픽쳐 목록과 관련되는 참조 픽쳐를 나타내는 RPL 구조(700)에서의 주소 지정 가능한 위치이다. 각각의 엔트리(715)는, 인터 예측을 위해 사용되는 픽쳐를 참조하는 픽쳐 순서 카운트(POC) 값(또는 다른 포인터 값)을 포함할 수도 있다. 구체적으로, 단방향 인터 예측에 의해 사용되는 픽쳐에 대한 참조는 RefPicList[0](711)에 저장되고 양방향 인터 예측에 의해 사용되는 픽쳐에 대한 참조는 RefPicList[0](711) 및 RefPicList[1](712) 둘 모두에 저장된다. 예를 들면, 단방향 인터 예측은 RefPicList[0](711)에 의해 나타내어지는 하나의 참조 픽쳐 내의 블록에 대한 참조에 의해 현재 픽쳐의 블록을 코딩할 수도 있다. 게다가, 양방향 인터 예측은, RefPicList[0](711)에 의해 나타내어지는 하나의 참조 픽쳐 내의 그리고 RefPicList[1](712)에 의해 나타내어지는 하나의 참조 픽쳐 내의 블록에 대한 참조에 의해 현재 픽쳐의 블록을 코딩할 수도 있다. 단방향 예측에 따라 코딩되는 슬라이스는 P 슬라이스로 지칭되고 양방향 예측에 따라 코딩되는 슬라이스는 B 슬라이스로 지칭된다는 것을 유의해야 한다. 그러한 만큼, RefPicList[0](711)는 단방향 예측(P) 슬라이스의 인터 예측을 위해 사용되는 참조 픽쳐 목록(예를 들면, 대응하는 참조 픽쳐의 목록을 포함함) 또는 양방향 예측(B) 슬라이스의 인터 예측을 위해 사용되는 두 개의 참조 픽쳐 목록 중 제1의 것이다. 게다가, RefPicList[1](712)는 (예를 들면, RefPicList[0]과 연계하여) B 슬라이스의 인터 예측을 위해 사용되는 제2 참조 픽쳐 목록이다.

[0091] 특정한 예에서, RPL 구조(700)는 ref_pic_list_struct(listIdx, rplsIdx)로서 나타내어질 수 있는데, 여기서 listIdx(721)는, RefPicList[0](711) 및/또는 RefPicList[1](712)와 같은 대응하는 참조 픽쳐 목록을 식별하는 인덱스이고, rplsIdx(725)는 대응하는 참조 픽쳐 목록에서 참조 엔트리(715)를 나타내는 인덱스이다. 따라서, ref_pic_list_struct()는 listIdx(721) 및 rplsIdx(725)에 기초하여 참조 엔트리(715)를 반환하는 신택스 구조이다. ref_pic_list_struct()는 RefPicList[0] 및 RefPicList[1]을 포함하는 신택스 구조로서 또한 정의될 수 있다. 인코더는 비디오 시퀀스의 각각의 인트라 코딩되지 않은 슬라이스(non-intra-coded slice)에 대한 RPL 구조(700)의 일부를 인코딩할 수 있다. 디코더는, 그 다음, 코딩된 비디오 시퀀스의 각각의 인트라 코딩되지 않은 슬라이스를 디코딩하기 이전에, RPL 구조(700)의 대응하는 부분을 해결할 수 있다. 예를 들면, 시퀀스의 많은 픽쳐에 관련되는 RPL 구조(700)의 부분은 SPS에 저장될 수 있고, 적은 개수의 픽쳐에 적용되는 RPL 구조(700)의 부분은 PPS에 저장될 수 있으며, 특정한 슬라이스에 적용되는 RPL 구조(700)의 부분은 슬라이스 헤더에 저장될 수 있다.

[0092] 참조 픽쳐의 세트(setOfRefPics)(733)는, 현재 픽쳐와 동일한 nuh_layer_id를 갖는 RefPicList[0](711) 내의 모든 엔트리 및 현재 픽쳐와 동일한 nuh_layer_id를 갖는 RefPicList[1](712) 내의 모든 엔트리에 의해 참조되는 고유의 픽쳐의 세트이다. RefPicList[0](711) 및 RefPicList[1](712)은 슬라이스에 고유할 수도 있다. 따라서, setOfRefPics(733)는 현재 픽쳐의 현재 슬라이스에 대한 고유의 참조 픽쳐의 세트를 포함할 수도 있는데, 여기서 고유의 참조 픽쳐의 세트는 현재 픽쳐와 동일한 레이어에 포함된다. 몇몇 시스템에서, setOfRefPics(733)에서의 참조 픽쳐의 개수는 모든 레이어에 대해 전역적인 정적인 값에 의해 제한될 수도 있다. 그러나, 상기에서 언급되는 바와 같이, 사용되는 DPB 공간의 양은, 레이어가 출력 레이어인지 또는 아닌지의 여부에 따라 변할 수도 있다. 그러한 만큼, 레이어에 관계없이 픽쳐의 동일한 정적인 최대 값을 setOfRefPics(733)에 적용하는 것은 비효율적인 메모리 할당을 초래할 수도 있다. 예를 들면, 정적인 최대 값은, 출력 레이어에 대한 setOfRefPics(733)에서 존재할 수 있는 픽쳐의 개수를 과도하게 제한할 수도 있고 및/또는 참조 레이어(비 출력 레이어)에 대해 충분히 제한적이지 않을 수도 있다. 본 예는, 최대 디코딩된 픽쳐 버퍼 사이즈에 기초하여 setOfRefPics(733)를 제한하는 것에 의해 이 문제를 해결한다. 최대 디코딩된 픽쳐 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변한다. 그러한 만큼, setOfRefPics(733)는, 임의의 정적인 값에 기초하여 제한되는 것이 아니라, 대신, 각각의 레이어에 대한 DPB에서의 이용 가능한 메모리 공간의 양에 기초하여 제한된다. 따라서, setOfRefPics(733)는, 대응하는 레이어가 출력 레이어인지 또는 참조 레이어인지의 여부에 따라, 상이한 사이즈로 동적으로 제한될 수 있다.

[0093] 몇몇 예에서, 인코더는 RPL 구조(700)에서 참조 엔트리의 개수(num_ref_entries)(732)를 또한 시그널링할 수 있다. num_ref_entries(732)는 RPL 구조(700)에서의 참조 엔트리의 개수를 나타내는 신택스 엘리먼트이다.

num_ref_entries(732)는 num_ref_entries [listIdx][rplsIdx]로서 나타내어질 수 있다. setOfRefPics(733)에서와 같이, 몇몇 비디오 코딩 시스템은, 레이어가 출력 레이어인지 또는 단지 참조 레이어인지의 여부에 따라, 과도하게 제한적일 수도 있는 또는 과도하게 허용할 수도 있는 정적인 값에 기초하여 num_ref_entries(732)를 제한한다. 본 개시는 최대 디코딩된 픽쳐 버퍼 사이즈에 기초하여 num_ref_entries(732)를 또한 제한할 수도 있다. 그러한 만큼, num_ref_entries(732)는, 모든 레이어에 대해 전역적인 임의의 정적인 값에 기초하여 제한되는 것이 아니라, 대신, 대응하는 레이어에 대한 DPB에서의 이용 가능한 메모리 공간의 양에 기초하여 제한된다. 따라서, num_ref_entries(732)는, 대응하는 레이어가 출력 레이어인지 또는 참조 레이어인지의 여부에 따라 상이한 사이즈로 동적으로 제한될 수 있다.

[0094] 도 8은 예시적인 비트스트림(800)을 예시하는 개략도이다. 예를 들면, 비트스트림(800)은, 방법(100)에 따라, 코덱 시스템(200) 및/또는 디코더(400)에 의한 디코딩을 위해 코덱 시스템(200) 및/또는 인코더(300)에 의해 생성될 수 있다. 게다가, 비트스트림(800)은 RPL 구조(700)를 활용하는 것에 의해 코딩될 수도 있는 인코딩된 다중 레이어 비디오 시퀀스(600)를 포함할 수도 있다. 또한, 비트스트림(800)은 HRD(500)와 같은 HRD의 동작을 제어하기 위한 다양한 파라미터를 포함할 수도 있다. 그러한 파라미터에 기초하여, HRD(500)는 디코딩을 위한 디코더를 향한 송신 이전에, 표준과의 적합성에 대해 비트스트림(800)을 검사할 수 있다.

[0095] 비트스트림(800)은 VPS(811), 하나 이상의 SPS(813), 복수의 픽쳐 파라미터 세트(PPS)(815), 복수의 픽쳐 헤더(816), 복수의 슬라이스 헤더(817), 및 이미지 데이터(820)를 포함한다. VPS(811)는 전체 비트스트림(800)에 관련되는 데이터를 포함한다. 예를 들면, VPS(811)는 비트스트림(800)에서 사용되는 OLS, 레이어, 및/또는 서브레이어에 관련되는 데이터를 포함할 수도 있다. SPS(813)는 비트스트림(800)에 포함되는 코딩된 비디오 시퀀스의 모든 픽쳐에 공통적인 시퀀스 데이터를 포함한다. 예를 들면, 각각의 레이어는 하나 이상의 코딩된 비디오 시퀀스를 포함할 수도 있고, 각각의 코딩된 비디오 시퀀스는 대응하는 파라미터에 대해 SPS(813)를 참조할 수도 있다. SPS(813)의 파라미터는 픽쳐 사이즈 조정, 비트 심도, 코딩 도구 파라미터, 비트 레이트 제한, 등등을 포함할 수 있다. 각각의 시퀀스가 SPS(813)를 참조하지만, 몇몇 예에서, 단일의 SPS(813)가 다수의 시퀀스에 대한 데이터를 포함할 수 있다는 것을 유의해야 한다. PPS(815)는 전체 픽쳐에 적용되는 파라미터를 포함한다. 그러므로, 비디오 시퀀스의 각각의 픽쳐는 PPS(815)를 참조할 수도 있다. 각각의 픽쳐가 PPS(815)를 참조하지만, 단일의 PPS(815)가 몇몇 예에서 다수의 픽쳐에 대한 데이터를 포함할 수 있다는 것을 유의해야 한다. 예를 들면, 다수의 유사한 픽쳐가 유사한 파라미터에 따라 코딩될 수도 있다. 그러한 경우에, 단일의 PPS(815)는 그러한 유사한 픽쳐에 대한 데이터를 포함할 수도 있다. PPS(815)는 대응하는 픽쳐의 슬라이스, 양자화 파라미터, 오프셋, 등등에 대해 이용 가능한 코딩 도구를 나타낼 수 있다.

[0096] 슬라이스 헤더(817)는 픽쳐(825)의 대응하는 슬라이스(827)에 고유한 파라미터를 포함한다. 그러므로, 비디오 시퀀스의 슬라이스(827)마다 하나의 슬라이스 헤더(817)가 있을 수도 있다. 슬라이스 헤더(817)는 슬라이스 타입 정보, 픽쳐 순서 카운트(POC), 참조 픽쳐 목록, 예측 가중치, 타일 진입 포인트, 또는 블록화 제거 파라미터, 등등을 포함할 수도 있다. 몇몇 예에서, 비트스트림(800)은, 단일의 픽쳐(825)의 모든 슬라이스(827)에 적용되는 파라미터를 포함하는 신택스 구조인 픽쳐 헤더(816)를 또한 포함할 수도 있다는 것을 유의해야 한다. 이러한 이유 때문에, 픽쳐 헤더(816) 및 슬라이스 헤더(817)는 몇몇 상황에서 상호 교환 가능하게 사용될 수도 있다. 예를 들면, 소정의 파라미터는, 그러한 파라미터가 특정한 슬라이스(827)에 적용되는지 또는 픽쳐(825)의 모든 슬라이스(827)에 공통인지의 여부에 따라 슬라이스 헤더(817)와 픽쳐 헤더(816) 사이에서 이동될 수도 있다.

[0097] 이미지 데이터(820)는 인터 예측, 인터레이어 예측, 및/또는 인트라 예측에 따라 인코딩되는 비디오 데이터뿐만 아니라, 대응하는 변환되고 양자화된 잔차 데이터를 포함한다. 예를 들면, 이미지 데이터(820)는 레이어(823), 픽쳐(825), 및/또는 슬라이스(827)를 포함할 수도 있다. 레이어(823)는, nuh_layer_id(835)와 같은 레이어 ID에 의해 나타내어지는 바와 같은 명시된 특성(예를 들면, 공통 해상도, 프레임 레이트, 이미지 사이즈, 등등), 및 관련된 비 VCL NAL 단위(842)를 공유하는 VCL NAL 단위(841)의 세트이다. 예를 들면, 레이어(823)는, 레이어(823)의 픽쳐(825)를 디코딩하기 위해 사용되는 임의의 파라미터 세트와 함께 동일한 nuh_layer_id(835)를 공유하는 픽쳐(825)의 세트를 포함할 수도 있다. 예를 들면, 레이어(823)는 도 6으로부터의 레이어 N(631) 및/또는 레이어 N+1(632)과, 각각, 실질적으로 유사하다.

[0098] nuh_layer_id(835)는 적어도 하나의 NAL 단위를 포함하는 레이어(823)의 식별자를 명시하는 신택스 엘리먼트이다. 예를 들면, 기본 레이어로서 공지되어 있는 최저 품질 레이어는, 더 높은 품질의 레이어에 대해 nuh_layer_id(835)의 증가하는 값을 가지면서, nuh_layer_id(835)의 최저 값을 포함할 수도 있다. 그러므로, 하위 레이어는 nuh_layer_id(835)의 더 작은 값을 갖는 레이어(823)이고 상위 레이어는 nuh_layer_id(835)의

더 큰 값을 갖는 레이어(823)이다. 레이어(823)의 데이터는 nuh_layer_id(835)에 기초하여 상관된다. 예를 들면, 파라미터 세트 및 비디오 데이터는, 그러한 파라미터 세트/비디오 데이터를 포함하는 최하위 레이어(823)에 대응하는 nuh_layer_id(835)의 값과 관련될 수도 있다. 그러한 만큼, VCL NAL 단위(841)의 세트는, VCL NAL 단위(841)의 세트가 모두 nuh_layer_id(835)의 특정한 값을 가질 때의 레이어(823)의 부분이다.

[0099] 픽처(825)는 프레임 또는 그 필드를 생성하는 루마 샘플의 어레이 및/또는 크로마 샘플의 어레이이다. 예를 들면, 픽처(825)는 디스플레이를 위해 출력될 수도 있는 또는 출력을 위해 다른 픽처(들)(825)의 코딩을 지원하기 위해 사용될 수도 있는 코딩된 이미지이다. 픽처(825)는 하나 이상의 슬라이스(827)를 포함한다. 슬라이스(827)는, 단일의 NAL 단위에서, 예컨대 VCL NAL 단위(841)에서 배타적으로 포함되는 픽처(825)의 정수 개수의 완전한 타일 또는 (예를 들면, 타일 내의) 정수 개수의 연속적인 완전한 코딩 트리 단위(CTU) 행으로서 정의될 수도 있다. 슬라이스(827)는 CTU 및/또는 코딩 트리 블록(CTB)으로 추가로 분할된다. CTU는 코딩 트리에 의해 구획될 수 있는 사전 정의된 사이즈의 샘플의 그룹이다. CTB는 CTU의 서브세트이며 CTU의 루마 성분 또는 크로마 성분을 포함한다. CTU/CTB는 코딩 트리에 기초하여 코딩 블록으로 추가로 분할된다. 그 다음, 코딩 블록은 예측 메커니즘에 따라 인코딩/디코딩될 수 있다.

[0100] 비트스트림(800)은 NAL 단위의 시퀀스로서 코딩될 수 있다. NAL 단위는 비디오 데이터 및/또는 지원 신택스를 위한 컨테이너(container)이다. NAL 단위는 VCL NAL 단위(841) 또는 비 VCL NAL 단위(842)일 수 있다. VCL NAL 단위(841)는 비디오 데이터, 예컨대 이미지 데이터(820) 및 관련된 슬라이스 헤더(817)를 포함하도록 코딩되는 NAL 단위이다. 특정한 예로서, 각각의 슬라이스(827) 및 관련된 슬라이스 헤더(817)는 단일의 VCL NAL 단위(841)로 인코딩될 수 있다. 비 VCL NAL 단위(842)는, 비디오 데이터의 디코딩, 적합성 검사의 수행, 또는 다른 동작을 지원하는 신택스 및/또는 파라미터와 같은 비 비디오 데이터를 포함하는 NAL 단위이다. 예를 들면, 비 VCL NAL 단위(842)는 VPS(811), SPS(813), PPS(815), 픽처 헤더(816), 또는 다른 지원 신택스를 포함할 수 있다. 그러한 만큼, 비트스트림(800)은 일련의 VCL NAL 단위(841) 및 비 VCL NAL 단위(842)이다. 각각의 NAL 단위는 nuh_layer_id(835)를 포함하는데, 이것은 인코더 또는 디코더가 대응하는 NAL 단위를 어떤 레이어(823)가 포함하는지를 결정하는 것을 허용한다.

[0101] 다수의 레이어(823)를 포함하는 비트스트림(800)은 디코더에 의해 요청될 때까지 인코딩되어 저장될 수도 있다. 예를 들면, 디코더는 레이어(823), 및/또는 다수의 레이어(823)를 포함하는 OLS를 요청할 수 있다. 특정한 예에서, 레이어(823)는 베이스 레이어 및 하나 이상의 향상 레이어를 포함할 수도 있다. 인코더 및/또는 콘텐츠 서버는 요청된 출력 레이어(들)를 디코딩하는 데 필요한 레이어(823)만을 디코더로 전송해야 한다.

[0102] 비트스트림은 도 7의 RPL 구조(700)와 실질적으로 유사할 수도 있는 ref_pic_list_struct(831)를 포함할 수도 있다. ref_pic_list_struct(831)는, 인터 예측 및/또는 인터레이어 예측에 따라 슬라이스(827)의 블록을 코딩하기 위해 사용되는 setofRefPics를 참조하는 RefPicList[0] 및 RefPicList[1]을 포함할 수 있다. ref_pic_list_struct(831)는 도 7의 num_ref_entries(732)와 실질적으로 유사한 num_ref_entries(832)를 또한 포함할 수 있다. 따라서, num_ref_entries(832)는 ref_pic_list_struct(831) 내의 참조 엔트리의 개수를 나타낸다. ref_pic_list_struct(831)는, ref_pic_list_struct(831)의 범위에 따라, SPS(813), 픽처 헤더(816), 및/또는 슬라이스 헤더(817)에서 저장될 수 있다. 예를 들면, 전체 시퀀스에 대한 참조 픽처를 참조하는 ref_pic_list_struct(831)는 SPS(813)에 포함하고, 전체 픽처(825)에 대한 참조 픽처를 참조하는 ref_pic_list_struct(831)는 픽처 헤더(816)에 포함되고, 그리고 슬라이스(827)에 대한 참조 픽처를 참조하는 ref_pic_list_struct(831)는 슬라이스 헤더(817)에 포함된다.

[0103] 비트스트림(800)은 디코딩된 픽처 버퍼 파라미터(dpb_parameters)(837)를 또한 포함할 수도 있다. dpb_parameters(837)는, 하나 이상의 OLS에 대한 DPB 사이즈, 최대 픽처 재정렬 개수, 및 최대 레이턴시의 정보를 제공하는 신택스 구조이다. 따라서, dpb_parameters(837)는 디코딩 프로세스 동안 DPB의 기능을 명시한다. 구체적인 예로서, dpb_parameters(837)는 최대 디코딩된 픽처 버퍼 마이너스 1(max_dec_pic_buffering_minus1)(838)을 포함할 수도 있으며, 이것은 DPB의 최대 요구 사이즈를 픽처 저장 버퍼의 단위로 명시하는 신택스 엘리먼트이다. dpb_parameters(837)는, 범위에 따라, VPS(811) 및/또는 SPS(813)에서 저장될 수도 있다. 예를 들면, 전체 비디오에 적용되는 dpb_parameters(837)는 VPS(811)에서 저장될 수 있고, 한편, 특정한 비디오 시퀀스 및/또는 특정한 레이어(823)에 적용되는 dpb_parameters(837)는 SPS(813)에서 저장될 수 있다.

[0104] 상기에서 언급되는 바와 같이, ref_pic_list_struct(listIdx, rplsIdx)로서 또한 나타내어지는 ref_pic_list_struct(831)는 인터 예측된 픽처의 재구성을 지원하도록 픽처 사이의 참조를 추적하기 위해 활용

될 수도 있다. 몇몇 비디오 코딩 시스템에서, ref_pic_list_struct(831)는 현재 픽처에 대해 활용될 수 있는 참조 엔트리의 최대 개수를 포함한다. 구체적으로, 몇몇 비디오 코딩 시스템에서, 현재 픽처에 대한 참조 엔트리의 최대 개수는 모든 레이어에 대해 전역적인 정적으로 정의된 값이다. 이것은, 레이어에 관계없이 정적으로 정의된 값에 기초하여 num_ref_entries [listIdx][rplsIdx]로서 또한 나타내어지는 num_ref_entries(832)를 제한하는 것으로 귀결된다. 이것은 또한, 레이어에 관계없이 정적으로 정의된 값에 기초하여 ref_pic_list_struct(831)에 의해 참조되는 도 7의 setOfRefPics(733)를 제한하는 것으로 귀결된다. 이 접근법에서의 문제는, 참조 레이어가 디코딩된 픽처 버퍼에서 출력 레이어와는 상이한 양의 공간을 사용한다는 것이다. 예를 들면, 참조 레이어는 픽처 재구성을 위해 공간을 사용하고, 한편 출력 레이어는 픽처 재구성 및 저장 보류 출력 둘 모두를 위해 공간을 사용한다. 따라서, 참조 레이어에 대해 사용되는 더 적은 양의 공간을 지원하기 위해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 출력 레이어의 픽처에 적용될 때 과도하게 제한적일 수도 있다. 대안적으로, 출력 레이어에 대해 선택되는 참조 엔트리의 정적으로 정의된 최대 개수는 참조 레이어의 픽처를 디코딩하는 데 필요한 것보다 더 많은 공간을 제공할 수도 있고, 그러므로, 메모리 리소스를 낭비할 수도 있다.

[0105] 상기 언급된 문제를 해결하기 위해, 비트스트림(800)은 상이한 타입의 레이어(823)에 대해 상이한 픽처 버퍼 사용량을 지원하도록 ref_pic_list_struct(831)를 제한한다. 예를 들면, 각각의 픽처에 대해 사용되는 ref_pic_list_struct(831)에서의 참조 엔트리의 개수를 나타내는 num_ref_entries(832)는, 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 선택되는 범위를 유지하도록 제한된다. 최대 디코딩된 픽처 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변한다. 따라서, 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 num_ref_entries(832)를 제한하는 것은, 출력 레이어 및 참조 레이어에 대해 상이한 개수의 참조 픽처가 활용되는 것을 허용한다. 특정한 예에서, dpb_parameters(837)에서의 max_dec_pic_buffering_minus1(838)은 대응하는 레이어에 대한 DPB의 최대 요구 사이즈를 명시한다. 따라서, 인코더 및/또는 HRD는 max_dec_pic_buffering_minus1(838)에 기초하여 최대 디코딩된 픽처 버퍼 사이즈를 유도할 수 있고 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 num_ref_entries(832)를 제한할 수 있다. 디코더는 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 제한된 num_ref_entries(832)를 갖는 ref_pic_list_struct(831)를 또한 수신할 수 있다. 예에 관계없이, num_ref_entries(832)는, 모든 레이어에 대해 전역적인 정적으로 정의된 값이 아니라 레이어에 고유한 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 제한된다. 또 다른 예에서, 도 7의 setOfRefPics(733)는 유사한 방식으로 제한될 수 있다. 예를 들면, 인코더 및/또는 HRD는 max_dec_pic_buffering_minus1(838)에 기초하여 최대 디코딩된 픽처 버퍼 사이즈를 유도할 수 있고 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 setOfRefPics(733)를 제한할 수 있다. 디코더는 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 제한된 setOfRefPics(733)를 갖는 ref_pic_list_struct(831)를 또한 수신할 수 있다. 예에 관계없이, setOfRefPics(733)는, 모든 레이어에 대해 전역적인 정적으로 정의된 값이 아니라 레이어에 고유한 최대 디코딩된 픽처 버퍼 사이즈에 기초하여 제한된다. 그러한 제약을 활용하는 것은 디코딩된 픽처 버퍼에서 메모리를 더욱 효율적인 할당을 지원하고, 그러므로, 더욱 최적의 메모리 사용량이 더욱 효율적인 인코딩을 촉진하기 때문에 증가된 코딩 효율성을 지원한다. 결과적으로, 인코더 및 디코더의 기능성이 증가된다. 게다가, 코딩 효율성이 증가되는데, 이것은 인코더 및 디코더 둘 모두에서의 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0106] 전술한 정보는, 이제, 본원의 하기에서 더욱 상세하게 설명된다. 레이어화된 비디오 코딩은 스케일러블 비디오 코딩 또는 스케일러빌러티를 갖는 비디오 코딩으로서 또한 지칭된다. 비디오 코딩에서의 스케일러빌러티는 다중 레이어 코딩 기술을 사용하는 것에 의해 지원될 수도 있다. 다중 레이어 비트스트림은 베이스 레이어(base layer; BL) 및 하나 이상의 향상 레이어(enhancement layer; EL)를 포함한다. 스케일러빌러티의 예는, 공간적 스케일러빌러티, 품질/신호 대 노이즈 비율(SNR) 스케일러빌러티, 멀티뷰 스케일러빌러티, 프레임 레이트 스케일러빌러티, 등등을 포함한다. 다중 레이어 코딩 기술이 사용되는 경우, 픽처 또는 그 일부는 참조 픽처를 사용하지 않으면서 코딩될 수도 있고(인트라 예측), 동일한 레이어에 있는 참조 픽처를 참조하는 것에 의해 코딩될 수도 있고(인터 예측), 및/또는 다른 레이어(들)에 있는 참조 픽처를 참조하는 것에 의해 코딩될 수도 있다(인터레이어 예측). 현재 픽처의 인터레이어 예측을 위해 사용되는 참조 픽처는 인터레이어 참조 픽처(inter-layer reference picture; ILRP)로서 지칭된다. 도 6은 상이한 레이어의 픽처가 상이한 해상도를 갖는 공간적 스케일러빌러티를 위한 다중 레이어 코딩의 한 예를 예시한다.

[0107] 몇몇 비디오 코딩 제품군은 단일 레이어 코딩을 위한 프로파일(들)과는 분리된 프로파일(들)에서 스케일러빌러티에 대한 지원을 제공한다. 스케일러블 비디오 코딩(SVC)은, 공간적, 시간적 및 품질 스케일러빌러티에 대한 지원을 제공하는 고급 비디오 코딩(AVC)의 스케일러블 확장이다. SVC의 경우, EL MB가 하위 레이어로부터의 동

일 위치에 위치된 블록(collocated block)을 사용하여 예측되는지의 여부를 나타내기 위해, EL 픽처의 각각의 매크로블록(macroblock; MB)에서 플래그가 시그널링된다. 동일 위치된 블록으로부터의 예측은 텍스쳐, 모션 벡터, 및/또는 코딩 모드를 포함할 수도 있다. SVC의 구현예는 수정되지 않은 AVC 구현예를 그들의 설계에서 바로 재사용할 수 없을 수도 있다. SVC EL 매크로블록 신택스 및 디코딩 프로세스는 AVC 신택스 및 디코딩 프로세스와는 상이하다.

[0108] 스케일러블 HEVC(SHVC)는 공간 및 품질 스케일러빌러티에 대한 지원을 제공하는 HEVC의 확장이다. 멀티뷰 HEVC(MV-HEVC)는 멀티뷰 스케일러빌러티에 대한 지원을 제공하는 HEVC의 확장이다. 3D HEVC(3D-HEVC)는, MV-HEVC보다 더 발전되고 더 효율적인 3D 비디오 코딩에 대한 지원을 제공하는 HEVC의 확장이다. 시간적 스케일러빌러티는 단일 레이어 HEVC 코덱의 필수적인 부분으로서 포함될 수도 있다. HEVC의 다중 레이어 확장에서, 인터레이어 예측을 위해 사용되는 디코딩된 픽처는 동일한 AU로부터만 유래하며 장기간 참조 픽처(long-term reference picture; LTRP)로서 취급된다. 그러한 픽처는, 현재 레이어의 다른 시간 참조 픽처와 함께 참조 픽처 목록(들)에서의 참조 인덱스를 할당받는다. 인터레이어 예측(inter-layer prediction; ILP)은, 참조 픽처 목록(들)의 인터레이어 참조 픽처(들)를 참조하도록 참조 인덱스의 값을 설정하는 것에 의해, 예측 단위(PU) 레벨에서 달성된다. 공간적 스케일러빌러티는, ILRP가 인코딩 또는 디코딩되고 있는 현재 픽처와는 상이한 공간 해상도를 가질 때 참조 픽처 또는 그 일부를 재샘플링한다. 참조 픽처 재샘플링은, 픽처 레벨 또는 코딩 블록 레벨 중 어느 하나에서 실현될 수 있다.

[0109] VVC는 레이어화된 비디오 코딩을 또한 지원할 수도 있다. VVC 비트스트림은 다수의 레이어를 포함할 수 있다. 레이어는 모두 서로 독립적일 수 있다. 예를 들면, 각각의 레이어는 인터레이어 예측을 사용하지 않고 코딩될 수 있다. 이 경우, 레이어는 동시 송출 레이어로서 또한 지칭된다. 몇몇 경우에, 레이어 중 일부는 ILP를 사용하여 코딩된다. VPS의 플래그는, 레이어가 동시 송출 레이어인지의 여부 또는 일부 레이어가 ILP를 사용하는지의 여부를 나타낼 수 있다. 일부 레이어가 ILP를 사용하는 경우, 레이어 사이의 레이어 종속성 관계도 또한 VPS에서 시그널링된다. SHVC 및 MV-HEVC와는 달리, VVC는 OLS를 명시하지 않을 수도 있다. OLS는 레이어의 명시된 세트를 포함하는데, 여기서 레이어의 세트의 하나 이상의 레이어는 출력 레이어인 것으로 명시된다. 출력 레이어는 출력되는 OLS의 레이어이다. VVC의 몇몇 구현예에서, 레이어가 동시 송출 레이어인 경우, 디코딩 및 출력을 위해 단지 하나의 레이어만이 선택될 수도 있다. VVC의 몇몇 구현예에서, 모든 레이어를 포함하는 전체 비트스트림은 임의의 레이어가 ILP를 사용할 때 디코딩되도록 명시된다. 게다가, 레이어 중 소정의 레이어가 출력 레이어인 것으로 명시된다. 출력 레이어는 오로지 최상위 레이어(highest layer), 모든 레이어, 또는 최상위 레이어 플러스 지시된 하위 레이어의 세트인 것으로 나타내어질 수도 있다.

[0110] 전술한 양태는 소정의 스케일러빌러티 관련 문제를 포함한다. 그러한 시스템에서의 스케일러빌러티 설계는, 레이어 고유의 프로파일, 계층(tier), 및 레벨(PTL)뿐만 아니라, 레이어 고유의 코딩된 픽처 버퍼(coded picture buffer; CPB) 동작을 포함한다. PTL 시그널링 효율성이 개선되어야 한다. 서브 레이어에 대한 시퀀스 레벨 HRD 파라미터에 대한 신호 효율성이 개선되어야 한다. DPB 파라미터 시그널링이 개선되어야 한다. 몇몇 설계는 단일 레이어 비트스트림으로 하여금 VPS를 참조하게 한다. 그러한 설계에서의 num_ref_entries[][]의 값 범위는 정확하지 않으며 디코더에 대해 예기치 않은 에러를 야기한다. 그러한 설계에서의 디코딩 프로세스는 서브 비트스트림 추출을 수반하는데, 이것은 디코더 구현에 부담을 부가한다. 그러한 설계에 대한 일반적인 디코딩 프로세스는 인터레이어 예측을 갖는 다수의 레이어를 포함하는 스케일러블 비트스트림에 대해 작동하지 않을 수도 있다. 그러한 설계에서의 변수 NoOutputOfPriorPicsFlag의 값의 유도는 그러한 설계에서 AU 기반이 아니라 픽처 기반일 수도 있다. 그러한 설계에서의 스케일러블 내포 SEI 메시지는, nesting_ols_flag가 1과 동일한 경우, OLS의 레이어 대신, OLS에 직접적으로 적용되도록 단순화되어야 한다. 비 스케일러블 내포 SEI 메시지(non-scalable-nested SEI message)는 payloadType이 0(버퍼링 기간), 1(픽처 타이밍), 또는 130(디코딩 단위 정보)와 동일한 경우, 0 번째 OLS에만 적용되도록 명시될 수도 있다.

[0111] 일반적으로, 본 개시는 비디오 코딩에서의 스케일러빌러티에 대한 다양한 접근법을 설명한다. 그 기술의 설명은 VVC에 기초한다. 그러나, 그 기술은 다른 비디오 코덱 명세에 기초하는 레이어화된 비디오 코딩에도 또한 적용된다. 상기 언급된 문제 중 하나 이상은 다음과 같이 해결될 수도 있다. 구체적으로, 본 개시는 비디오 코딩에서의 개선된 스케일러빌러티 지원을 위한 방법을 포함한다.

[0112] 다음의 것은 다양한 예시적인 정의이다. OP는, OLS 인덱스 및 TemporalId의 가장 높은 값에 의해 식별되는 OLS의 시간적 서브세트일 수도 있다. 출력 레이어는 출력되는 OLS의 레이어일 수도 있다. OLS는 레이어의 세트일 수도 있는데, 여기서 레이어의 세트의 하나 이상의 레이어는 출력 레이어인 것으로 명시된다. OLS 레이어 인덱스는 OLS의 레이어의 목록에 대한, OLS의 레이어의 인덱스일 수도 있다. 서브 비트스트림 추출 프로세스는, 타

깃 OLS 인덱스 및 타깃 최고 TemporalId에 의해 결정되는, 타깃 세트에 속하지 않는 비트스트림의 NAL 단위가 비트스트림으로부터 제거되게 하는 명시된 프로세스일 수도 있는데, 출력 서브 비트스트림은 타깃 세트에 속하는 비트스트림의 NAL 단위를 포함한다.

[0113] 예시적인 비디오 파라미터 세트 RBSP 신택스는 다음과 같다.

video_parameter_set_rbsp( ) {	디스크립터
vps_video_parameter_set_id	u(4)
vps_max_layers_minus1	u(6)
vps_max_sub_layers_minus1	u(3)
if(vps_max_layers_minus1 > 0 && vps_max_sub_layers_minus1 > 0)	
vps_all_layers_same_num_sub_layers_flag	u(1)
if(vps_max_layers_minus1 > 0)	
vps_all_independent_layers_flag	u(1)
...	
vps_num_ptls	u(8)
for(i = 0; i < vps_num_ptls; i++) {	
if(i > 0)	
pt_present_flag[i]	u(1)
if(vps_max_sub_layers_minus1 > 0 && !vps_all_layers_same_num_sub_layers_flag)	
ptl_max_temporal_id[i]	u(3)
}	
while(!byte_aligned( ))	
vps_ptl_byte_alignment_zero_bit /* equal to 0 */	u(1)
for(i = 0; i < vps_num_ptls; i++)	
profile_tier_level(pt_present_flag[i], ptl_max_temporal_id[i])	
for(i = 0; i < TotalNumOlss; i++)	
if(NumLayersInOls[i] > 1 && vps_num_ptls > 1)	
ols_ptl_idx[i]	u(8)
if(!vps_all_independent_layers_flag)	
vps_num_dpb_params	ue(v)
if(vps_num_dpb_params > 0) {	
same_dpb_size_output_or_nonoutput_flag	u(1)

[0114]

if(vps_max_sub_layers_minus1 > 0)	
vps_sub_layer_dpb_params_present_flag	u(1)
}	
for(i = 0; i < vps_num_dpb_params; i++) {	
dpb_size_only_flag[i]	u(1)
if(vps_max_sub_layers_minus1 > 0 && !vps_all_layers_same_num_sub_layers_flag)	
dpb_max_temporal_id[i]	u(3)
dpb_parameters(dpb_size_only_flag[i], dpb_max_temporal_id[i], vps_sub_layer_dpb_params_present_flag)	
}	
for(i = 0; i < vps_max_layers_minus1 && vps_num_dpb_params > 1; i++) {	
if(!vps_independent_layer_flag[i])	
layer_output_dpb_params_idx[i]	ue(v)
if(LayerUsedAsRefLayerFlag[i] && !same_dpb_size_output_or_nonoutput_flag)	
layer_nonoutput_dpb_params_idx[i]	ue(v)
}	
general_hrd_params_present_flag	u(1)
if(general_hrd_params_present_flag) {	
num_units_in_tick	u(32)
time_scale	u(32)
general_hrd_parameters( )	
}	
vps_extension_flag	u(1)
if(vps_extension_flag)	
while(more_rbsp_data( ))	
vps_extension_data_flag	u(1)

[0115]

rbsp_trailing_bits( )	
}	

[0116]

[0117] 예시적인 시퀀스 파라미터 세트 RBSP 신택스는 다음과 같다.

seq_parameter_set_rbsp( ) {	디스크립터
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sub_layers_minus1	u(3)
sps_reserved_zero_4bits	u(4)
sps_ptl_dpb_present_flag	u(1)
if(sps_ptl_dpb_present_flag)	
profile_tier_level(1, sps_max_sub_layers_minus1)	
gdr_enabled_flag	u(1)
sps_seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
...	
log2_max_pic_order_cnt_lsb_minus4	ue(v)
poc_msb_in_rap_pics_flag	u(1)
if(poc_msb_in_rap_pics_flag > 0)	
poc_msb_len_minus1	ue(v)
if(sps_max_sub_layers_minus1 > 0)	
sps_sub_layer_dpb_params_flag	u(1)
if(sps_ptl_dpb_present_flag)	
dpb_parameters(0, sps_max_sub_layers_minus1, sps_sub_layer_dpb_params_flag)	
for(i = (sps_sub_layer_ordering_info_present_flag ? 0 : sps_max_sub_layers_minus1); i <= sps_max_sub_layers_minus1; i++) {	
sps_max_dec_pic_buffering_minus1[i]	ue(v)

[0118]

sps_max_num_reorder_pics[i]	ue(v)
sps_max_latency_increase_plus1[i]	ue(v)
}	
long_term_ref_pics_flag	u(1)
...	
sps_scaling_list_enabled_flag	u(1)
general_hrd_parameters_present_flag	u(1)
if(general_hrd_parameters_present_flag) {	
num_units_in_tick	u(32)
time_scale	u(32)
sub_layer_cpb_parameters_present_flag	u(1)
if(sub_layer_cpb_parameters_present_flag)	
general_hrd_parameters(0, sps_max_sub_layers_minus1)	
else	
general_hrd_parameters(sps_max_sub_layers_minus1, sps_max_sub_layers_minus1)	
}	
vui_parameters_present_flag	u(1)
if(vui_parameters_present_flag)	
vui_parameters( )	
sps_extension_flag	u(1)
if(sps_extension_flag)	
while(more_rbsp_data( ))	
sps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

[0119]

[0120] 예시적인 DPB 파라미터 신택스는 다음과 같다.

dpb_parameters(dpbSizeOnlyFlag, maxSubLayersMinus1, subLayerInfoFlag) {	디스크립터
for(i = (subLayerInfoFlag ? 0 : maxSubLayersMinus1); i <= maxSubLayersMinus1; i++) {	
max_dec_pic_buffering_minus1[i]	ue(v)
if(!dpbSizeOnlyFlag) {	
max_num_reorder_pics[i]	ue(v)
max_latency_increase_plus1[i]	ue(v)
}	
}	
}	

[0121]

[0122] 예시적인 일반적 HRD 파라미터 신택스는 다음과 같다.

general_hrd_parameters( ) {	디스크립터
general_nal_hrd_params_present_flag	u(1)
general_vcl_hrd_params_present_flag	u(1)
if(general_nal_hrd_params_present_flag || general_vcl_hrd_params_present_flag) {	
decoding_unit_hrd_params_present_flag	u(1)
if(decoding_unit_hrd_params_present_flag) {	
tick_divisor_minus2	u(8)
decoding_unit_cpb_params_in_pic_timing_sei_flag	u(1)
}	
bit_rate_scale	u(4)
cpb_size_scale	u(4)
if(decoding_unit_hrd_params_present_flag)	
cpb_size_du_scale	u(4)
}	
if(vps_max_sub_layers_minus1 > 0)	
sub_layer_cpb_params_present_flag	u(1)
if(TotalNumOlss > 1)	
num_ols_hrd_params_minus1	ue(v)
hrd_cpb_cnt_minus1	ue(v)
for(i = 0; i <= num_ols_hrd_params_minus1; i++) {	
if(vps_max_sub_layers_minus1 > 0 && !vps_all_layers_same_num_sub_layers_flag)	
hrd_max_temporal_id[i]	u(3)
ols_hrd_parameters(hrd_max_temporal_id[i])	
}	
if(num_ols_hrd_params_minus1 > 0)	

[0123]

for(i = 1; i < TotalNumOlss; i++)	
ols_hrd_idx[i]	ue(v)
}	

[0124]

[0125] 예시적인 OLS HRD 파라미터 신택스는 다음과 같다.

ols_hrd_parameters(hrdMaxTid) {	디스크립터
firstSubLayer = sub_layer_cpb_params_present_flag ? 0: hrdMaxTid	
for(i = firstSubLayer; i <= hrdMaxTid; i++) {	
fixed_pic_rate_general_flag[i]	u(1)
if(!fixed_pic_rate_general_flag[i])	
fixed_pic_rate_within_cvs_flag[i]	u(1)
if(fixed_pic_rate_within_cvs_flag[i])	
elemental_duration_in_tc_minus1[i]	ue(v)
else if(hrd_cpb_cnt_minus1 == 0)	
low_delay_hrd_flag[i]	u(1)
if(general_nal_hrd_params_present_flag)	
sub_layer_hrd_parameters(i)	
if(general_vcl_hrd_params_present_flag)	
sub_layer_hrd_parameters(i)	
}	
}	

[0126]

[0127] 예시적인 서브 레이어 HRD 파라미터 신택스는 다음과 같다.

sub_layer_hrd_parameters(subLayerId) {	디스크립터
for(j = 0; j <= hrd_cpb_cnt_minus1; j++) {	
bit_rate_value_minus1[subLayerId][j]	ue(v)
cpb_size_value_minus1[subLayerId][j]	ue(v)
if(decoding_unit_hrd_params_present_flag) {	
cpb_size_du_value_minus1[subLayerId][j]	ue(v)
bit_rate_du_value_minus1[subLayerId][j]	ue(v)
}	
cbr_flag[subLayerId][j]	u(1)
}	
}	

[0128]

[0129] 예시적인 비디오 파라미터 세트 RBSP 의미론(semantics)은 다음과 같다. vps_max_layers_minus1 플러스 1은, VPS를 참조하는 각각의 CVS에서 레이어의 최대 허용되는 개수를 명시한다. vps_max_sub_layers_minus1 플러스 1은, VPS를 참조하는 각각의 CVS에서 존재할 수도 있는 시간적 서브 레이어의 최대 개수를 명시한다. vps_max_sub_layers_minus1의 값은 0 이상 6 이하의 범위 내에 있을 수도 있다. 1과 동일한

vps_all_layers_same_num_sub_layers_flag는 시간적 서브 레이어의 개수가 VPS를 참조하는 각각의 CVS의 모든 레이어에 대해 동일하다는 것을 명시한다. 0과 동일한 vps_all_layers_same_num_sub_layers_flag는 VPS를 참조하는 각각의 CVS의 레이어가 동일한 개수의 시간적 서브 레이어를 가질 수도 있거나 또는 가지지 않을 수도 있다는 것을 명시한다. 존재하지 않는 경우, vps_all_layers_same_num_sub_layers_flag의 값은 1과 동일한 것으로 추론될 수도 있다. 1과 동일한 vps_all_independent_layers_flag는 CVS의 모든 레이어가 인터레이어 예측을 사용하지 않고 독립적으로 코딩된다는 것을 명시한다. 0과 동일한 vps_all_independent_layers_flag는, CVS의 레이어 중 하나 이상이 인터레이어 예측을 사용할 수도 있다는 것을 명시한다. 존재하지 않는 경우, vps_all_independent_layers_flag의 값은 1과 동일한 것으로 추론될 수도 있다. vps_all_independent_layers_flag가 1과 동일한 경우, vps_independent_layer_flag[i]의 값은 1과 동일한 것으로 추론된다. vps_all_independent_layers_flag가 0과 동일한 경우, vps_independent_layer_flag[0]의 값은 1과 동일한 것으로 추론된다.

[0130] 0과 동일한 vps_direct_dependency_flag[i][j]는, 인덱스 j를 갖는 레이어가 인덱스 i를 갖는 레이어에 대한 직접적인 참조 레이어가 아니다는 것을 명시한다. 1과 동일한 vps_direct_dependency_flag[i][j]는, 인덱스 j를 갖는 레이어가 인덱스 i를 갖는 레이어에 대한 직접적인 참조 레이어이다는 것을 명시한다. vps_direct_dependency_flag[i][j]가 0 이상 vps_max_layers_minus1 이하의 범위 내의 i 및 j에 대해 존재하지 않는 경우, 플래그는 0과 동일한 것으로 추론된다. i 번째 레이어의 j 번째 직접적인 종속 레이어를 명시하는 변수 DirectDependentLayerIdx[i][j], 및 레이어 인덱스 j를 갖는 레이어가 임의의 다른 레이어에 의해 참조 레이어로서 사용되는지의 여부를 명시하는 변수 LayerUsedAsRefLayerFlag[j]는 다음과 같이 유도될 수도 있다:

```
for( i = 0; i <= vps_max_layers_minus1; i++ )
    LayerUsedAsRefLayerFlag[ j ] = 0
for( i = 1; i < vps_max_layers_minus1; i++ )
    if( !vps_independent_layer_flag[ i ] )
        for( j = i − 1, k = 0; j >= 0; j− − )
            if( vps_direct_dependency_flag[ i ][ j ] ) {
                DirectDependentLayerIdx[ i ][ k++ ] = j
                LayerUsedAsRefLayerFlag[ j ] = 1
            }
```

[0131]

[0132] vps_layer_id[i]와 동일한 nuh_layer_id를 갖는 레이어의 레이어 인덱스를 명시하는 변수 GeneralLayerIdx[i]는 다음과 같이 유도될 수도 있다:

```
for( i = 0; i <= vps_max_layers_minus1; i++ )
    GeneralLayerIdx[ vps_layer_id[ i ] ] = i
```

[0133]

[0134] 1과 동일한 each_layer_is_an_ols_flag는, 각각의 출력 레이어 세트가 단지 하나의 레이어만을 포함하고 비트스트림에서의 각각의 레이어 그 자체가 출력 레이어 세트이되 단일의 포함된 레이어가 유일한 출력 레이어이다는 것을 명시한다. 제로와 동일한 each_layer_is_an_ols_flag는, 출력 레이어 세트가 하나보다 더 많은 레이어를 포함할 수도 있다는 것을 명시한다. vps_max_layers_minus1이 제로와 동일하면, each_layer_is_an_ols_flag의 값은 1과 동일한 것으로 추론된다. 그렇지 않고, vps_all_independent_layers_flag가 0과 동일한 경우, each_layer_is_an_ols_flag의 값은 0과 동일한 것으로 추론된다.

[0135] 제로와 동일한 ols_mode_idc는, VPS에 의해 명시되는 OLS의 총 개수가 vps_max_layers_minus1 + 1과 동일하다는 것, i 번째 OLS가 0 이상 i 이하의 레이어 인덱스를 갖는 레이어를 포함한다는 것, 및 각각의 OLS에 대해 OLS의 최상위 레이어만이 출력된다는 것을 명시한다. 1과 동일한 ols_mode_idc는, VPS에 의해 명시되는 OLS의 총 개수가 vps_max_layers_minus1 + 1과 동일하다는 것, i 번째 OLS가 0 이상 i 이하의 레이어 인덱스를 갖는 레이어를 포함한다는 것, 및 각각의 OLS에 대해 OLS의 모든 레이어가 출력된다는 것을 명시한다. 2와 동일한 ols_mode_idc는, VPS에 의해 명시되는 OLS의 총 개수가 명시적으로 시그널링된다는 것 및 각각의 OLS에 대해 OLS의 하위 레이어의 명시적으로 시그널링된 세트 및 최상위 레이어가 출력된다는 것을 명시한다. ols_mode_idc의 값은 0 이상 2 이하의 범위 내에 있을 수도 있다. vps_all_independent_layers_flag가 1과 동일하고

each_layer_is_an_ols_flag가 0과 동일한 경우, ols_mode_idc의 값은 2와 동일한 것으로 추론된다. num_output_layer_sets_minus1 플러스 1은, ols_mode_idc가 2와 동일할 때 VPS에 의해 명시되는 OLS의 총 개수를 명시한다.

[0136] VPS에 의해 명시되는 OLS의 총 개수를 명시하는 변수 TotalNumOlss는 다음과 같이 유도될 수도 있다:

```
if( vps_max_layers_minus1 == 0 )
    TotalNumOlss = 1
else if( each_layer_is_an_ols_flag || ols_mode_idc == 0 || ols_mode_idc == 1 )
    TotalNumOlss = vps_max_layers_minus1 + 1
else if( ols_mode_idc == 2 )
    TotalNumOlss = num_output_layer_sets_minus1 + 1
```

[0137]

[0138] layer_included_flag[i][j]는, ols_mode_idc가 2와 동일할 때 j 번째 레이어(예를 들면, vps_layer_id[j]와 동일한 nuh_layer_id를 갖는 레이어)가 i 번째 OLS에 포함되는지의 여부를 명시한다. 1과 동일한 layer_included_flag[i][j]는 j 번째 레이어가 i 번째 OLS에 포함된다는 것을 명시한다. 제로와 동일한 layer_included_flag[i][j]는 j 번째 레이어가 i 번째 OLS에 포함되지 않는다는 것을 명시한다. i 번째 OLS에서의 레이어의 개수를 명시하는 변수 NumLayersInOls[i], 및 i 번째 OLS에서의 j 번째 레이어의 nuh_layer_id 값을 명시하는 변수 LayerIdInOls[i][j]는 다음과 같이 유도될 수도 있다:

```
NumLayersInOls[ 0 ] = 1
LayerIdInOls[ 0 ][ 0 ] = vps_layer_id[ 0 ]
for( i = 1, i < TotalNumOlss; i++ ) {
    if( each_layer_is_an_ols_flag ) {
        NumLayersInOls[ i ] = 1
        LayerIdInOls[ i ][ 0 ] = vps_layer_id[ i ]
    } else if( ols_mode_idc == 0 || ols_mode_idc == 1 ) {
        NumLayersInOls[ i ] = i + 1
        for( j = 0; j < NumLayersInOls[ i ]; j++ )
            LayerIdInOls[ i ][ j ] = vps_layer_id[ j ]
    } else if( ols_mode_idc == 2 ) {
        for( k = 0, j = 0; k <= vps_max_layers_minus1; k++ )
            if( layer_included_flag[ i ][ k ] )
                LayerIdInOls[ i ][ j++ ] = vps_layer_id[ k ]
        NumLayersInOls[ i ] = j
    }
}
```

[0139]

[0140] LayerIdInOls[i][j]와 동일한 nuh_layer_id를 갖는 레이어의 OLS 레이어 인덱스를 명시하는 변수 OlsLayeIdx[i][j]는 다음과 같이 유도될 수도 있다:

```
for( i = 0, i < TotalNumOlss; i++ )
    for j = 0; j < NumLayersInOls[ i ]; j++ )
        OlsLayeIdx[ i ][ LayerIdInOls[ i ][ j ] ] = j
```

[0141]

[0142] 각각의 OLS의 최하위 레이어는 독립 레이어이어야 한다. 다시 말하면, 0 이상 TotalNumOlss - 1 이하의 범위 내의 각각의 i에 대해, vps_independent_layer_flag[GeneralLayerIdx[LayerIdInOls[i][0]]]의 값은 1과 동일해야 한다. 각각의 레이어는 VPS에 의해 명시되는 적어도 하나의 OLS에 포함되어야 한다. 다시 말하면, 0 이상

vps_max_layers_minus1 이하의 범위 내에 있는 k에 대해 vps_layer_id[k]중 하나와 동일한 nuh_layer_id의 특정한 값(nuhLayerId)을 갖는 각각의 레이어에 대해, i 및 j의 값의 적어도 하나의 쌍이 있어야 하는데, 여기서 i는 0 이상 TotalNumOlss - 1 이하의 범위 내에 있고, j는 0 이상 NumLayersInOls[i] - 1 이하의 범위 내에 있고, 그 결과, LayerIdInOls[i][j]의 값은 nuhLayerId와 동일하다. OLS의 임의의 레이어는 OLS의 출력 레이어 또는 OLS의 출력 레이어의 (직접적인 또는 간접적인) 참조 레이어이어야 한다.

[0143] vps_output_layer_flag[i][j]는, ols_mode_idc가 2와 동일할 때 i 번째 OLS의 j 번째 레이어가 출력되는지의 여부를 명시한다. 1과 동일한 vps_output_layer_flag[i]는 i 번째 OLS의 j 번째 레이어가 출력된다는 것을 명시한다. 0과 동일한 vps_output_layer_flag[i]는, i 번째 OLS의 j 번째 레이어가 출력되지 않는다는 것을 명시한다. vps_all_independent_layers_flag가 1과 동일하고 each_layer_is_an_ols_flag가 0과 동일한 경우, vps_output_layer_flag[i]의 값은 1과 동일한 것으로 추론된다. i 번째 OLS의 j 번째 레이어가 출력된다는 것을 값 1이 명시하고 i 번째 OLS의 j 번째 레이어가 출력되지 않는다는 것을 값 제로가 명시하는 변수 OutputLayerFlag[i][j]는 다음과 같이 유도될 수도 있다.

```
for( i = 0, i < TotalNumOlss; i++ ) {
        OutputLayerFlag[ i ][ NumLayersInOls[ i ] − 1 ] = 1
        for( j = 0; j < NumLayersInOls[ i ] − 1; j++ )
                if( ols_mode_idc[ i ] == 0 )
                        OutputLayerFlag[ i ][ j ] = 0
                else if( ols_mode_idc[ i ] == 1 )
                        OutputLayerFlag[ i ][ j ] = 1
                else if( ols_mode_idc[ i ] == 2 )
                        OutputLayerFlag[ i ][ j ] = vps_output_layer_flag[ i ][ j ]
```

[0144] }

[0145] 0 번째 OLS는 최하위 레이어(예를 들면, vps_layer_id[0]과 동일한 nuh_layer_id를 갖는 레이어)만을 포함하고, 0 번째 OLS에 대해, 포함된 레이어만이 출력된다는 것을 유의해야 한다. vps_num_ptls는 VPS에서의 profile_tier_level() 신택스 구조의 개수를 명시한다. 1과 동일한 pt_present_flag[i]는 프로파일, 계층 및 일반적인 제약 정보가 VPS의 i 번째 profile_tier_level() 신택스 구조에 존재한다는 것을 명시한다. 0과 동일한 pt_present_flag[i]는, 프로파일, 계층, 및 일반적인 제약 정보가 VPS의 i 번째 profile_tier_level() 신택스 구조에서 존재하지 않는다는 것을 명시한다. pt_present_flag[0]의 값은 0과 동일한 것으로 추론된다. pt_present_flag[i]가 0과 동일한 경우, VPS의 i 번째 profile_tier_level() 신택스 구조에 대한 프로파일, 계층, 및 일반적인 제약 정보는, VPS의 (i - 1) 번째 profile_tier_level() 신택스 구조에 대한 것과 동일한 것으로 추론된다.

[0146] ptl_max_temporal_id[i]는 VPS의 i 번째 profile_tier_level() 신택스 구조에서 레벨 정보가 존재하는 가장 높은 서브 레이어 표현의 TemporalId를 명시한다. ptl_max_temporal_id[i]의 값은 0 이상 vps_max_sub_layers_minus1 이하의 범위 내에 있어야 한다. vps_max_sub_layers_minus1이 0과 동일한 경우, ptl_max_temporal_id[i]의 값은 0과 동일한 것으로 추론된다. vps_max_sub_layers_minus1이 제로보다 더 크고 vps_all_layers_same_num_sub_layers_flag가 1과 동일한 경우, ptl_max_temporal_id[i]의 값은 vps_max_sub_layers_minus1과 동일한 것으로 추론된다. vps_ptl_byte_alignment_zero_bit는 제로와 동일해야 한다.

[0147] ols_ptl_idx[i]는, i 번째 OLS에 적용되는 profile_tier_level() 신택스 구조의, VPS의 profile_tier_level() 신택스 구조 목록에 대한, 인덱스를 명시한다. 존재하는 경우, ols_ptl_idx[i]의 값은 0 이상 vps_num_ptls - 1 이하의 범위 내에 있어야 한다. NumLayersInOls[i]가 1과 동일한 경우, i 번째 OLS에 적용되는 profile_tier_level() 신택스 구조는 i 번째 OLS의 레이어에 의해 참조되는 SPS에서 존재한다. vps_num_dpb_params는 VPS에서의 dpb_parameters() 신택스 구조의 개수를 명시한다. vps_num_dpb_params의 값은 0 이상 16 이하의 범위 내에 있어야 한다. 존재하지 않는 경우, vps_num_dpb_params의 값은 제로와 동일한 것으로 추론될 수도 있다. 1과 동일한 same_dpb_size_output_or_nonoutput_flag는, VPS에 layer_nonoutput_dpb_params_idx[i] 신택스 엘리먼트가 존재하지 않는다는 것을 명시한다. 0과 동일한

same_dpb_size_output_or_nonoutput_flag는 VPS에 layer_nonoutput_dpb_params_idx[i] 신택스 엘리먼트가 존재할 수도 있거나 또는 존재하지 않을 수도 있다는 것을 명시한다. vps_sub_layer_dpb_params_present_flag는 VPS의 dpb_parameters() 신택스 구조의 max_dec_pic_buffering_minus1[], max_num_reorder_pics[], 및 max_latency_increase_plus1[] 신택스 엘리먼트의 존재를 제어하기 위해 사용된다. 존재하지 않는 경우, vps_sub_dpb_params_info_present_flag는 0과 동일한 것으로 추론된다.

[0148] 1과 동일한 dpb_size_only_flag[i]는, VPS의 i 번째 dpb_parameters() 신택스 구조에, max_num_reorder_pics[] 및 max_latency_increase_plus1[] 신택스 엘리먼트가 존재하지 않는다는 것을 명시한다. 1과 동일한 dpb_size_only_flag[i]는, VPS의 i 번째 dpb_parameters() 신택스 구조에, max_num_reorder_pics[] 및 max_latency_increase_plus1[] 신택스 엘리먼트가 존재할 수도 있다는 것을 명시한다. dpb_max_temporal_id[i]는, VPS의 i 번째 dpb_parameters() 신택스 구조에서 DPB 파라미터가 존재할 수도 있는 가장 높은 서브 레이어 표현의 TemporalId를 명시한다. dpb_max_temporal_id[i]의 값은 0 이상 vps_max_sub_layers_minus1 이하의 범위 내에 있어야 한다. vps_max_sub_layers_minus1이 0과 동일한 경우, dpb_max_temporal_id[i]의 값은 제로와 동일한 것으로 추론될 수도 있다. vps_max_sub_layers_minus1이 0보다 더 크고 vps_all_layers_same_num_sub_layers_flag가 1과 동일한 경우, dpb_max_temporal_id[i]의 값은 vps_max_sub_layers_minus1과 동일한 것으로 추론된다. layer_output_dpb_params_idx[i]는, i 번째 레이어가 OLS의 출력 레이어일 때 i 번째 레이어에 적용되는 dpb_parameters() 신택스 구조의, VPS의 dpb_parameters() 신택스 구조 목록에 대한, 인덱스를 명시한다. 존재하는 경우, layer_output_dpb_params_idx[i]의 값은 0 이상 vps_num_dpb_params - 1 이하의 범위 내에 있어야 한다.

[0149] vps_independent_layer_flag[i]가 1과 동일한 경우, i 번째 레이어가 출력 레이어일 때 적용되는 dpb_parameters() 신택스 구조는 대응하는 레이어에 의해 참조되는 SPS에 존재하는 dpb_parameters() 신택스 구조이다. 그렇지 않으면(vps_independent_layer_flag[i]가 1과 동일함), 다음의 것이 적용된다. vps_num_dpb_params가 1과 동일한 경우, layer_output_dpb_params_idx[i]의 값은 0과 동일한 것으로 추론된다. 비트스트림 적합성은, layer_output_dpb_params_idx[i] 의 값이, dpb_size_only_flag[layer_output_dpb_params_idx[i]]가 0과 동일하도록 하는 그러한 것이어야 한다는 것을 규정할 수도 있다.

[0150] layer_nonoutput_dpb_params_idx[i]는, i 번째 레이어가 OLS의 비 출력 레이어일 때 i 번째 레이어에 적용되는 dpb_parameters() 신택스 구조의, VPS의 dpb_parameters() 신택스 구조 목록에 대한, 인덱스를 명시한다. 존재하는 경우, layer_nonoutput_dpb_params_idx[i]의 값은 0 이상 vps_num_dpb_params - 1 이하의 범위 내에 있어야 한다. same_dpb_size_output_or_nonoutput_flag가 1과 동일하면, 다음의 것이 적용된다. vps_independent_layer_flag[i]가 1과 동일한 경우, i 번째 레이어가 비 출력 레이어일 때 i 번째 레이어에 적용되는 dpb_parameters() 신택스 구조는 레이어에 의해 참조되는 SPS에 존재하는 dpb_parameters() 신택스 구조이다. 그렇지 않으면(vps_independent_layer_flag[i]가 1과 동일함), layer_nonoutput_dpb_params_idx[i]의 값은 layer_output_dpb_params_idx[i]와 동일한 것으로 추론된다. 그렇지 않으면(same_dpb_size_output_or_nonoutput_flag가 0과 동일함), vps_num_dpb_params가 1과 동일할 때, layer_output_dpb_params_idx[i]의 값은 0과 동일한 것으로 추론된다.

[0151] 1과 동일한 general_hrd_params_present_flag는, 신택스 엘리먼트 num_units_in_tick 및 time_scale 및 신택스 구조 general_hrd_parameters()가 SPS RBSP 신택스 구조에서 존재한다는 것을 명시한다. 0과 동일한 general_hrd_params_present_flag는, 신택스 엘리먼트 num_units_in_tick 및 time_scale 및 신택스 구조 general_hrd_parameters()가 SPS RBSP 신택스 구조에서 존재하지 않는다는 것을 명시한다. num_units_in_tick은, 클록 틱 카운터(clock tick counter)의 1 증분(클록 틱으로서 칭해짐)에 대응하는 주파수 time_scale 헤르츠(Hz)에서 동작하는 클록의 시간 단위의 개수이다. num_units_in_tick는 0보다 더 커야 한다. 초 단위의 클록 틱은, time_scale에 의해 나누어진 num_units_in_tick의 몫(quotient)과 동일하다. 예를 들면, 비디오 신호의 픽쳐 레이트가 25 Hz인 경우, time_scale은 27,000,000과 동일할 수도 있고 num_units_in_tick은 1,080,000과 동일할 수도 있고, 결과적으로, 클록 틱은 0.04 초와 동일할 수도 있다. time_scale은 1초에 지나가는 시간 단위의 개수이다. 예를 들면, 27 MHz 클록을 사용하여 시간을 측정하는 시간 좌표 시스템은27,000,000의 time_scale을 갖는다. time_scale의 값은 0보다 더 커야 한다.

[0152] 0과 동일한 vps_extension_flag는, vps_extension_data_flag 신택스 엘리먼트가 VPS RBSP 신택스 구조에서 존재하지 않는다는 것을 명시한다. 1과 동일한 vps_extension_flag는 VPS RBSP 신택스 구조에 vps_extension_data_flag 신택스 엘리먼트가 존재한다는 것을 명시한다. vps_extension_data_flag는 임의의 값

을 가질 수도 있다. vps_extension_data_flag의 존재 및 값은, 프로파일에 대한 디코더 적합성에 영향을 끼치지 않을 수도 있다. 적합한 디코더는 모든 vps_extension_data_flag 신택스 엘리먼트를 무시할 수도 있다.

[0153] 예시적인 시퀀스 파라미터 세트 RBSP 의미론은 다음과 같다. SPS RBSP는 참조되기 이전에 디코딩 프로세스에게 이용 가능해야 하거나, 0과 동일한 TemporalId를 갖는 적어도 하나의 액세스 단위에 포함되어야 하거나, 또는 외부 수단을 통해 제공되어야 하며, SPS RBSP를 포함하는 SPS NAL 단위는, SPS NAL 단위를 참조하는 PPS NAL 단위의 가장 낮은 nuh_layer_id 값과 동일한 nuh_layer_id를 가져야 한다. CVS에서 sps_seq_parameter_set_id의 특정한 값을 갖는 모든 SPS NAL 단위는 동일한 콘텐츠를 가져야 한다. sps_decoding_parameter_set_id는, 0보다 더 큰 경우, SPS에 의해 참조되는 DPS에 대한 dps_decoding_parameter_set_id의 값을 명시한다. sps_decoding_parameter_set_id가 0과 동일한 경우, SPS는 DPS를 참조하지 않으며, SPS를 참조하는 각각의 CLVS를 디코딩할 때 어떠한 DPS도 참조되지 않는다. sps_decoding_parameter_set_id의 값은 비트스트림에서 코딩된 픽처에 의해 참조되는 모든 SPS에서 동일해야 한다.

[0154] sps_video_parameter_set_id는, 0보다 더 큰 경우, SPS에 의해 참조되는 VPS에 대한 vps_video_parameter_set_id의 값을 명시한다. sps_video_parameter_set_id가 0과 동일한 경우, SPS는 VPS를 참조하지 않을 수도 있고, SPS를 참조하는 각각의 CLVS를 디코딩할 때 어떠한 VPS도 참조되지 않으며, GeneralLayerIdx[nuh_layer_id]의 값은 제로와 동일한 것으로 추론되어야 하며, vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]의 값은 1과 동일한 것으로 추론될 수도 있다. vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]가 1과 동일한 경우, 특정한 nuh_layer_id 값(nuhLayerId)을 갖는 CLVS에 의해 참조되는 SPS는 nuhLayerId와 동일한 nuh_layer_id를 가져야 한다.

[0155] sps_max_sub_layers_minus1 플러스 1은, SPS를 참조하는 각각의 CLVS에서 존재할 수도 있는 시간적 서브 레이어의 최대 개수를 명시한다. sps_max_sub_layers_minus1의 값은 0 이상 vps_max_sub_layers_minus1 이하의 범위 내에 있어야 한다. 적합한 비트스트림(conforming bitstream)에서 sps_reserved_zero_4bits는 0과 동일해야 한다. sps_reserved_zero_4bits에 대한 다른 값은 예약될 수도 있다.

[0156] 1과 동일한 sps_ptl_dpb_present_flag는, profile_tier_level() 신택스 구조 및 dpb_parameters() 신택스 구조가 SPS에서 존재한다는 것을 명시한다. 0과 동일한 sps_ptl_dpb_present_flag는 profile_tier_level() 신택스 구조 및 dpb_parameters() 신택스 구조가 SPS에서 존재하지 않는다는 것을 명시한다. sps_ptl_dpb_present_flag의 값은 vps_independent_layer_flag[nuh_layer_id]와 동일해야 한다. vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]가 1과 동일한 경우, 변수 MaxDecPicBuffMinus1은 SPS의 dpb_parameters() 신택스 구조에서 max_dec_pic_buffering_minus1[sps_max_sub_layers_minus1]과 동일하게 설정된다. 그렇지 않으면, MaxDecPicBuffMinus1은 VPS의 layer_nonoutput_dpb_params_idx[GeneralLayerIdx[nuh_layer_id]] 번째 dpb_parameters() 신택스 구조의 max_dec_pic_buffering_minus1[sps_max_sub_layers_minus1]과 동일하게 설정된다. 1과 동일한 gdr_enabled_flag는 GDR 픽처가 SPS를 참조하는 CLVS에서 존재할 수도 있다는 것을 명시한다. 0과 동일한 gdr_enabled_flag는 GDR 픽처가 SPS를 참조하는 CLVS에서 존재하지 않는다는 것을 명시한다.

[0157] sps_sub_layer_dpb_params_flag는 SPS의 dpb_parameters() 신택스에서 max_dec_pic_buffering_minus1[i], max_num_reorder_pics[i], 및 max_latency_increase_plus1[i] 신택스 엘리먼트의 존재를 제어하기 위해 사용된다. 존재하지 않는 경우, sps_sub_dpb_params_info_present_flag는 0과 동일한 것으로 추론된다. 0과 동일한 long_term_ref_pics_flag는 CLVS의 임의의 코딩된 픽처의 인터 예측을 위해 어떠한 LTRP도 사용되지 않는다는 것을 명시한다. 1과 동일한 long_term_ref_pics_flag는, CLVS의 하나 이상의 코딩된 픽처의 인터 예측을 위해 LTRP가 사용될 수도 있다는 것을 명시한다.

[0158] 예시적인 일반적 프로파일, 계층, 및 레벨 의미론은 다음과 같다. profile_tier_level() 신택스 구조는 레벨 정보 및, 옵션 사항으로, 프로파일, 계층, 서브 프로파일, 및 일반적인 제약 정보(PT 정보로서 나타내어짐)를 제공한다. profile_tier_level() 신택스 구조가 DPS에 포함되는 경우, OlsInScope는 DPS를 참조하는 전체 비트스트림의 모든 레이어를 포함하는 OLS이다. profile_tier_level() 신택스 구조가 VPS에 포함되는 경우, OlsInScope는 VPS에 의해 명시되는 하나 이상의 OLS이다. profile_tier_level() 신택스 구조가 SPS에 포함되는 경우, OlsInScope는 SPS를 참조하는 레이어 중 최하위 레이어(lowest layer)인 레이어 - 이것은 독립 레이어이어야 함 - 만을 포함하는 OLS이다.

[0159] general_profile_idc는 OlsInScope가 준수하는 프로파일을 나타낸다. general_tier_flag는 general_level_idc의 해석을 위한 계층 컨텍스트를 명시한다. num_sub_profiles는 general_sub_profile_idc[i] 신택스 엘리먼트

의 개수를 명시한다. general_sub_profile_idc[i]는 등록되는 i 번째 상호 운용성 메타데이터(interoperability metadata)를 나타낸다. general_level_idc는 OlsInScope가 준수하는 레벨을 나타낸다. general_level_idc의 더 큰 값은 더 높은 레벨을 나타낸다는 것을 유의해야 한다. OlsInScope에 대해 DPS에서 시그널링되는 최대 레벨은, OlsInScope 내에 포함되는 CVS에 대해 SPS에서 시그널링되는 레벨보다 더 높을 수도 있다. OlsInScope가 다수의 프로파일을 준수하는 경우, general_profile_idc는, 인코더에 의해 결정되는 바와 같은, 선호하는 디코딩된 결과 또는 선호하는 비트스트림 식별 정보(identification)를 제공하는 프로파일을 나타내어야 한다는 것을 또한 유의해야 한다. profile_tier_level() 신택스 구조가 DPS에 포함되고 OlsInScope의 CVS가 상이한 프로파일을 준수하는 경우, general_profile_idc 및 level_idc는 OlsInScope를 디코딩할 수 있는 디코더에 대한 프로파일 및 레벨을 나타내어야 한다는 것을 또한 유의해야 한다.

[0160] 1과 동일한 sub_layer_level_present_flag[i]는, i와 동일한 TemporalId를 갖는 서브 레이어 표현에 대한 profile_tier_level() 신택스 구조에 레벨 정보가 존재한다는 것을 명시한다. 0과 동일한 sub_layer_level_present_flag[i]는, i와 동일한 TemporalId를 갖는 서브 레이어 표현에 대한 profile_tier_level() 신택스 구조에 레벨 정보가 존재하지 않는다는 것을 명시한다. ptl_alignment_zero_bits는 0과 동일해야 한다. 신택스 엘리먼트 sub_layer_level_idc[i]의 의미론은, 존재하지 않는 값의 추론의 명세를 제외하고, 신택스 엘리먼트 general_level_idc와 동일하지만, 그러나 i와 동일한 TemporalId를 갖는 서브 레이어 표현에 적용된다.

[0161] 예시적인 DPB 파라미터 의미론은 다음과 같다. dpb_parameters(maxSubLayersMinus1, subLayerInfoFlag) 신택스 구조는 CVS의 각각의 CLVS에 대한 DPB 사이즈, 최대 픽처 재정렬 개수, 및 최대 레이턴시의 정보를 제공한다. dpb_parameters() 신택스 구조가 VPS에 포함되는 경우, dpb_parameters() 신택스 구조가 적용되는 OLS는 VPS에 의해 명시된다. dpb_parameters() 신택스 구조가 SPS에 포함되는 경우, dpb_parameters() 신택스 구조는, SPS를 참조하는 레이어 중 최하위 레이어인 레이어 - 이것은 독립 레이어이어야 함 - 만을 포함하는 OLS에 적용된다.

[0162] max_dec_pic_buffering_minus1[i] 플러스 1은, CVS의 각각의 CLVS에 대해, Htid가 i와 동일할 때 디코딩된 픽처 버퍼의 최대 요구 사이즈를 픽처 저장 버퍼의 단위로 명시한다. max_dec_pic_buffering_minus1[i] 값은 0 이상 MaxDpbSize - 1 이하의 범위 내에 있어야 한다. i가 0보다 더 큰 경우, max_dec_pic_buffering_minus1[i]는 max_dec_pic_buffering_minus1[i - 1] 이상이어야 한다. max_dec_pic_buffering_minus1[i]가 0 이상 maxSubLayersMinus1 - 1 이하의 범위 내의 i에 대해 존재하지 않는 경우, subLayerInfoFlag가 0과 동일한 것에 기인하여, max_dec_pic_buffering_minus1[i]는 max_dec_pic_buffering_minus1[maxSubLayersMinus1]과 동일한 것으로 추론된다.

[0163] max_num_reorder_pics[i]는, CVS의 각각의 CLVS에 대해, Htid가 i와 동일할 때 디코딩 순서에서 CLVS의 임의의 픽처에 선행하며 출력 순서에서 그 픽처를 따를 수 있는 CLVS의 픽처의 최대 허용된 개수를 명시한다. max_num_reorder_pics[i]의 값은, 0 이상 max_dec_pic_buffering_minus1[i] 이하의 범위 내에 있어야 한다. i가 0보다 더 큰 경우, max_num_reorder_pics[i]는 max_num_reorder_pics[i - 1] 이상이어야 한다. max_num_reorder_pics[i]가 0 이상 maxSubLayersMinus1 - 1 이하의 범위에서 i에 대해 존재하지 않는 경우, subLayerInfoFlag가 0과 동일한 것에 기인하여, max_num_reorder_pics[i]는 max_num_reorder_pics[maxSubLayersMinus1]과 동일한 것으로 추론된다.

[0164] 0과 동일하지 않은 max_latency_increase_plus1[i]는, MaxLatencyPictures[i]의 값을 계산하기 위해 사용되는데, 이것은, CVS의 각각의 CLVS에 대해, Htid가 i와 동일한 경우 출력 순서에서 CLVS의 임의의 픽처보다 선행할 수 있고 디코딩 순서에서 그 픽처를 뒤따르는 CLVS에서의 픽처의 최대 개수를 명시한다. max_latency_increase_plus1[i]가 0과 동일하지 않은 경우, MaxLatencyPictures[i]의 값은 다음과 같이 명시될 수도 있다.

[0165] $$\mathrm{MaxLatencyPictures}[\,i\,] = \mathrm{max\_num\_reorder\_pics}[\,i\,] + \mathrm{max\_latency\_increase\_plus1}[\,i\,] - 1$$

[0166] max_latency_increase_plus1[i]가 0과 동일한 경우, 어떠한 대응하는 제한도 표현되지 않는다.

[0167] max_latency_increase_plus1[i]의 값은 0 이상 232 - 2 이하의 범위 내에 있어야 한다. 0 이상 maxSubLayersMinus1 - 1 이하의 범위 내의 i에 대해 max_latency_increase_plus1[i]가 존재하지 않는 경우, subLayerInfoFlag가 0과 동일한 것에 기인하여, max_latency_increase_plus1[i]는 max_latency_increase_plus1[maxSubLayersMinus1]와 동일한 것으로 추론된다.

[0168] 예시적인 일반적 HRD 파라미터 의미론은 다음과 같다. general_hrd_parameters() 신택스 구조는 HRD 동작에서 사용되는 HRD 파라미터를 제공한다. num_ols_hrd_params_minus1 플러스 1은, general_hrd_parameters() 신택스 구조에서 존재하는 ols_hrd_parameters() 신택스 구조의 개수를 명시한다. num_ols_hrd_params_minus1의 값은 0 이상 63 이하의 범위 내에 있어야 한다. TotalNumOlss가 1보다 더 큰 경우, num_ols_hrd_params_minus1의 값은 0과 동일한 것으로 추론된다. hrd_cpb_cnt_minus1 플러스 1은 CVS의 비트스트림에서의 대안적 CPB 명세의 개수를 명시한다. hrd_cpb_cnt_minus1의 값은 0 이상 31 이하의 범위 내에 있어야 한다. hrd_max_temporal_id[i]는, HRD 파라미터가 i 번째 layer_level_hrd_parameters() 신택스 구조에 포함되는 가장 높은 서브 레이어 표현의 TemporalId를 명시한다. hrd_max_temporal_id[i]의 값은 0 이상 vps_max_sub_layers_minus1 이하의 범위 내에 있어야 한다. vps_max_sub_layers_minus1이 0과 동일한 경우, hrd_max_temporal_id[i]의 값은 0과 동일한 것으로 추론된다. ols_hrd_idx[i]는 i 번째 OLS에 적용되는 ols_hrd_parameters() 신택스 구조의 인덱스를 명시한다. ols_hrd_idx[i]의 값은 0 이상 num_ols_hrd_params_minus1 이하의 범위 내에 있어야 한다. 존재하지 않는 경우, ols_hrd_idx[i]의 값은 0과 동일한 것으로 추론된다.

[0169] 예시적인 참조 픽쳐 목록 구조 의미론은 다음과 같다. ref_pic_list_struct(listIdx, rplsIdx) 신택스 구조는 SPS에서 또는 슬라이스 헤더에서 존재할 수도 있다. 신택스 구조가 슬라이스 헤더에 포함되는지 또는 SPS에 포함되는지의 여부에 따라, 다음의 것이 적용된다. 슬라이스 헤더에서 존재하는 경우, ref_pic_list_struct(listIdx, rplsIdx) 신택스 구조는 현재 픽쳐(슬라이스를 포함하는 픽쳐)의 참조 픽쳐 목록(listIdx)을 명시한다. 그렇지 않은 경우(SPS에서 존재함), ref_pic_list_struct(listIdx, rplsIdx) 신택스 구조는 참조 픽쳐 목록(listIdx)의 후보를 명시하고, 이 절(clause)의 나머지 부분에서 명시되는 의미론에서의 용어 현재 픽쳐는, SPS에 포함되는 ref_pic_list_struct(listIdx, rplsIdx) 신택스 구조의 목록에 대한 인덱스와 동일한 ref_pic_list_idx[listIdx]를 포함하는 하나 이상의 슬라이스를 갖는, 그리고 SPS를 가리키는 CVS 내에 있는 각각의 픽쳐를 지칭한다. num_ref_entries[listIdx][rplsIdx]는 ref_pic_list_struct(listIdx, rplsIdx) 신택스 구조에서의 엔트리의 개수를 명시한다. num_ref_entries[listIdx][rplsIdx]의 값은 0 이상 MaxDecPicBuffMinus1 + 14 이하의 범위 내에 있어야 한다.

[0170] 예시적인 일반적 디코딩 프로세스는 다음과 같다. 이 프로세스에 대한 입력은 비트스트림(BitstreamToDecode)이다. 이 프로세스의 출력은 디코딩된 픽쳐의 목록이다. 디코딩 프로세스는, 명시된 프로파일 및 레벨을 준수하는 모든 디코더가, 그 프로파일 및 레벨을 준수하는 비트스트림에 대해 그 프로파일과 관련되는 디코딩 프로세스를 호출할 때, 수치적으로 동일한 크롭된 디코딩된 출력 픽쳐를 생성하도록 명시된다. (명시되는 바와 같은, 정확한 출력 순서 또는 출력 타이밍을 가지고) 본원에서 설명되는 프로세스에 의해 생성되는 것들과 동일한 크롭된 디코딩된 출력 픽쳐를 생성하는 임의의 디코딩 프로세스는 디코딩 프로세스 요건을 준수한다.

[0171] 비트스트림의 각각의 IRAP AU에 대해, 다음의 것이 적용된다. AU가 디코딩 순서에서 비트스트림의 제1 AU이거나, 각각의 픽쳐가 순시 디코딩 리프레시(instantaneous decoding refresh; IDR) 픽쳐이거나, 또는 각각의 픽쳐가 디코딩 순서에서 시퀀스 NAL 단위의 끝을 뒤따르는 레이어의 제1 픽쳐인 경우, 변수 NoIncorrectPicOutputFlag는 1과 동일하게 설정된다. 그렇지 않고, 변수 HandleCraAsCvsStartFlag가 AU에 대한 값으로 설정되는 경우, HandleCraAsCvsStartFlag는 외부 메커니즘에 의해 제공되는 값과 동일하게 설정되고 NoIncorrectPicOutputFlag는 HandleCraAsCvsStartFlag와 동일하게 설정된다. 그렇지 않으면, HandleCraAsCvsStartFlag 및 NoIncorrectPicOutputFlag는 둘 모두 0과 동일하게 설정된다.

[0172] 비트스트림의 각각의 점진적 디코딩 리프레시(gradual decoding refresh; GDR) AU에 대해, 다음의 것이 적용된다. AU가 디코딩 순서에서 비트스트림의 제1 AU이거나 또는 각각의 픽쳐가 디코딩 순서에서 시퀀스 NAL 단위의 끝을 뒤따르는 레이어의 제1 픽쳐인 경우, 변수 NoIncorrectPicOutputFlag는 1과 동일하게 설정된다. 그렇지 않고, 변수 HandleGdrAsCvsStartFlag를 AU에 대한 값으로 설정하기 위해 몇몇 외부 메커니즘이 이용 가능한 경우, HandleGdrAsCvsStartFlag는 외부 메커니즘에 의해 제공되는 값과 동일하게 설정되고 NoIncorrectPicOutputFlag는 HandleGdrAsCvsStartFlag와 동일하게 설정된다. 그렇지 않으면, HandleGdrAsCvsStartFlag 및 NoIncorrectPicOutputFlag는 둘 모두 0과 동일하게 설정된다. IRAP 픽쳐 및 GDR 픽쳐 둘 모두에 대한 상기의 동작은, 비트스트림에서 CVS의 식별을 위해 사용된다. 디코딩은 디코딩 순서대로 BitstreamToDecode의 각각의 코딩된 픽쳐에 대해 반복적으로 호출된다.

[0173] 참조 픽쳐 목록 구성을 위한 예시적인 디코딩 프로세스는 다음과 같다. 이 프로세스는 비 IDR 픽쳐의 각각의 슬라이스에 대한 디코딩 프로세스의 시작에서 호출된다. 참조 픽쳐는 참조 인덱스를 통해 주소 지정된다. 참조 인덱스는 참조 픽쳐 목록에 대한 인덱스이다. I 슬라이스를 디코딩하는 경우, 슬라이스 데이터의 디코딩에서 어떠한 참조 픽쳐 목록도 사용되지 않는다. P 슬라이스를 디코딩하는 경우, 슬라이스 데이터의 디코딩에서 참조 픽

쳐 목록 0(예를 들면, RefPicList[0])만이 사용된다. B 슬라이스를 디코딩하는 경우, 슬라이스 데이터의 디코딩에서 참조 픽쳐 목록 0 및 참조 픽쳐 목록 1(예를 들면, RefPicList[1]) 둘 모두가 사용된다.

[0174] 비트스트림 적합성을 위해 다음의 제약이 적용된다. 0 또는 1과 동일한 각각의 i에 대해, num_ref_entries[i][rplsIdx[i]] 는 NumRefIdxActive[i]보다 더 작아서는 안된다. RefPicList[0] 또는 RefPicList[1]의 각각의 활성 엔트리에 의해 참조되는 픽쳐는 DPB에서 존재해야 하며 현재 픽쳐의 것 이하의 TemporalId를 가져야 한다. RefPicList[0] 또는 RefPicList[1]의 각각의 엔트리에 의해 참조되는 픽쳐는 현재 픽쳐가 아니어야 하며 0과 동일한 non_reference_picture_flag를 가져야 한다. 픽쳐의 슬라이스의 RefPicList[0] 또는 RefPicList[1]의 단기간 참조 픽쳐(short term reference picture; STRP) 엔트리 및 동일한 픽쳐의 동일한 슬라이스 또는 상이한 슬라이스의 RefPicList[0] 또는 RefPicList[1]의 장기간 참조 픽쳐(long term reference picture; LTRP) 엔트리는 동일한 픽쳐를 참조해서는 안된다. RefPicList[0] 또는 RefPicList[1]에는, 현재 픽쳐의 PicOrderCntVal과 엔트리에 의해 참조되는 픽쳐의 PicOrderCntVal 사이의 차이가 224 이상인 어떠한 LTRP 엔트리도 있어서는 안된다.

[0175] setOfRefPics를, 현재 픽쳐와 동일한 nuh_layer_id를 갖는 RefPicList[0] 내의 모든 엔트리 및 현재 픽쳐와 동일한 nuh_layer_id를 갖는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 고유의 픽쳐의 세트라 하자. setOfRefPics에서의 픽쳐의 개수는 MaxDecPicBuffMinus1 이하여야 하고 setOfRefPics는 픽쳐의 모든 슬라이스에 대해 동일해야 한다. 현재 픽쳐가 단계별 시간적 서브 레이어 액세스(step-wise temporal sublayer access; STSA) 픽쳐인 경우, RefPicList[0] 또는 RefPicList[1]에는, 현재 픽쳐의 것과 동일한 TemporalId를 갖는 어떠한 활성 엔트리도 없어야 한다. 현재 픽쳐가, 디코딩 순서에서, 현재 픽쳐의 것과 동일한 TemporalId를 갖는 STSA 픽쳐를 뒤따르는 픽쳐인 경우, 디코딩 순서에서 STSA 픽쳐에 선행하는 RefPicList[0] 또는 RefPicList[1]에는 활성 엔트리로서 포함되는 현재 픽쳐의 것과 동일한 TemporalId를 갖는 어떠한 픽쳐도 존재하지 않아야 한다.

[0176] 현재 픽쳐의 슬라이스의 RefPicList[0] 또는 RefPicList[1]의 각각의 인터레이어 참조 픽쳐(ILRP) 엔트리에 의해 참조되는 픽쳐는 현재 픽쳐와 동일한 액세스 단위 내에 있어야 한다. 현재 픽쳐의 슬라이스의 RefPicList[0] 또는 RefPicList[1]의 각각의 ILRP 엔트리에 의해 참조되는 픽쳐는, DPB에서 존재해야 하고 현재 픽쳐의 것보다 더 작은 nuh_layer_id를 가져야 한다. 슬라이스의 RefPicList[0] 또는 RefPicList[1]의 각각의 ILRP 엔트리는 활성 엔트리이어야 한다.

[0177] 예시적인 HRD 명세는 다음과 같다. HRD는 비트스트림 및 디코더 적합성을 검사하기 위해 사용된다. 비트스트림 적합성 테스트의 세트는, entireBitstream으로 나타내어지는, 전체 비트스트림으로 지칭되는 비트스트림의 적합성을 검사하기 위해 사용된다. 비트스트림 적합성 테스트의 세트는 VPS에 의해 명시되는 각각의 OLS의 각각의 OP의 적합성을 테스트하기 위한 것이다.

[0178] 각각의 테스트에 대해, 다음의 순서화된 단계가 나열된 순서대로 적용되고, 이 절의 이들 단계 이후에 설명되는 프로세스가 후속된다. targetOp로서 나타내어지는 테스트 하에 있는 동작 포인트는, OLS 인덱스(opOlsIdx) 및 가장 높은 TemporalId 값(opTid)를 갖는 타깃 OLS를 선택하는 것에 의해 선택된다. opOlsIdx의 값은 0 이상 TotalNumOlss - 1 이하의 범위 내에 있다. opTid의 값은 0 이상 vps_max_sub_layers_minus1 이하의 범위 내에 있다. opOlsIdx 및 opTid의 선택된 값의 각각의 쌍은, 입력으로서 entireBitstream, opOlsIdx 및 opTid를 가지고 서브 비트스트림 추출 프로세스를 호출하는 것에 의한 출력인 서브 비트스트림이 다음의 조건을 충족하도록 하는 그런 것이어야 한다. BitstreamToDecode에는 LayerIdInOls[opOlsIdx]의 nuh_layer_id 값 각각과 동일한 nuh_layer_id를 갖는 적어도 하나의 VCL NAL 단위가 존재한다. BitstreamToDecode에는 opTid와 동일한 TemporalId를 갖는 적어도 하나의 VCL NAL 단위가 존재한다.

[0179] targetOp의 레이어가 entireBitstream의 모든 레이어를 포함하고 opTid가 entireBitstream의 모든 NAL 단위 중 가장 높은 TemporalId 값보다 더 큰 경우, BitstreamToDecode는 entireBitstream과 동일하게 설정된다. 그렇지 않으면, BitstreamToDecode는, 입력으로서 entireBitstream, opOlsIdx 및 opTid를 가지고 서브 비트스트림 추출 프로세스를 호출하는 것에 의한 출력이 되도록 설정된다. TargetOlsIdx 및 Htid의 값은 targetOp의 opOlsIdx 및 opTid와, 각각, 동일하게 설정된다. ScIdx 값이 선택된다. 선택된 ScIdx는 0 이상 hrd_cpb_cnt_minus1 이하의 범위 내에 있어야 한다. TargetOlsIdx에 적용 가능한 (TargetLayerBitstream에서 존재하거나 또는 외부 메커니즘을 통해 이용 가능한) 버퍼링 기간 SEI 메시지와 관련되는 BitstreamToDecode에서의 액세스 단위는, HRD 초기화 포인트로서 선택되며 타깃 OLS의 각각의 레이어에 대한 액세스 단위 0으로서 지칭된다.

[0180] 후속하는 단계는, 타깃 OLS에서 OLS 레이어 인덱스(TargetOlsLayerIdx)를 갖는 각각의 레이어에 적용된다. BitstreamToDecode에 적용 가능한 ols_hrd_parameters() 신택스 구조와 sub_layer_hrd_parameters() 신택스 구조는 다음과 같이 선택된다. VPS의(또는 외부 메커니즘을 통해 제공되는) ols_hrd_idx[TargetOlsIdx] 번째 ols_hrd_parameters() 신택스 구조가 선택된다. 선택된 ols_hrd_parameters() 신택스 구조 내에서, BitstreamToDecode가 타입 I 비트스트림인 경우, 조건 if(general_vcl_hrd_params_present_flag)에 바로 후속하는 sub_layer_hrd_parameters(Htid) 신택스 구조가 선택되고 변수 NalHrdModeFlag는 0과 동일하게 설정된다. 그렇지 않으면(BitstreamToDecode가 타입 II 비트스트림임), 조건 if(general_vcl_hrd_params_present_flag)(이 경우 변수 NalHrdModeFlag는 0과 동일하게 설정됨) 또는 조건 if(general_nal_hrd_params_present_flag)(이 경우 변수 NalHrdModeFlag는 1과 동일하게 설정됨) 중 어느 하나에 바로 후속하는 sub_layer_hrd_parameters(Htid) 신택스 구조가 선택된다. BitstreamToDecode가 타입 II 비트스트림이고 NalHrdModeFlag가 0과 동일한 경우, 필러 데이터(filler data) NAL 단위를 제외한 모든 비 VCL NAL 단위, 및, 존재하는 경우, NAL 단위 스트림으로부터 바이트 스트림을 형성하는 모든 leading_zero_8bits, zero_byte, start_code_prefix_one_3bytes 및 trailing_zero_8bits 신택스 엘리먼트는 BitstreamToDecode으로부터 삭제되고 나머지 비트스트림은 BitstreamToDecode에 할당된다.

[0181] decoding_unit_hrd_params_present_flag가 1과 동일한 경우, CPB는 액세스 단위 레벨(이 경우 변수 DecodingUnitHrdFlag는 0과 동일하게 설정됨) 또는 디코딩 단위 레벨(이 경우 변수 DecodingUnitHrdFlag는 1과 동일하게 설정됨) 중 어느 하나에서 동작하도록 스케줄링된다. 그렇지 않으면, DecodingUnitHrdFlag는 0과 동일하게 설정되고 CPB는 액세스 단위 레벨에서 동작하도록 스케줄링된다.

[0182] 액세스 단위 0으로부터 시작하는 BitstreamToDecode의 각각의 액세스 단위에 대해, 액세스 단위와 관련되고 TargetOlsIdx에 적용되는 버퍼링 기간 SEI 메시지(BitstreamToDecode에서 존재하거나 또는 외부 메커니즘을 통해 이용 가능함)가 선택되고, 액세스 단위와 관련되고 TargetOlsIdx에 적용되는 픽처 타이밍 SEI 메시지(BitstreamToDecode에서 존재하거나 또는 외부 메커니즘을 통해 이용 가능함)가 선택되고, 그리고 DecodingUnitHrdFlag가 1과 동일하고 decoding_unit_cpb_params_in_pic_timing_sei_flag가 0과 동일한 경우, 액세스 단위의 디코딩 단위와 관련되고 TargetOlsIdx에 적용되는 디코딩 단위 정보 SEI 메시지(BitstreamToDecode에서 존재하거나 또는 외부 메커니즘을 통해 이용 가능함)가 선택된다.

[0183] 각각의 적합성 테스트는 상기의 단계 각각에서 하나의 옵션의 조합을 포함한다. 단계에 대해 하나보다 더 많은 옵션이 있는 경우, 임의의 특정한 적합성 테스트에 대해, 하나의 옵션만이 선택된다. 모든 단계의 모든 가능한 조합은 적합성 테스트의 전체 세트를 형성한다. 테스트 하에 있는 각각의 동작 포인트에 대해, 수행될 비트스트림 적합성 테스트의 개수는 n0 * n1 * n2 * n3과 동일한데, 여기서 n0, n1, n2 및 n3의 값은 다음과 같이 명시된다. n1은 hrd_cpb_cnt_minus1 + 1과 동일하다. n1은 버퍼링 기간 SEI 메시지와 관련되는 BitstreamToDecode에서의 액세스 단위의 개수이다. n2는 다음과 같이 유도된다. BitstreamToDecode가 타입 I 비트스트림인 경우, n0은 1과 동일하다. 그렇지 않으면(BitstreamToDecode는 타입 II 비트스트림임), n0은 2와 동일하다. n3은 다음과 같이 유도된다. decoding_unit_hrd_params_present_flag가 0과 동일한 경우, n3은 1과 동일하다. 그렇지 않으면, n3은 2와 동일하다.

[0184] HRD는 비트스트림 추출기(옵션 사항으로 존재함), 코딩된 픽처 버퍼(CPB), 순시 디코딩 프로세스(instantaneous decoding process), 개념적으로 각각의 레이어에 대한 서브 DPB를 포함하는 디코딩된 픽처 버퍼(DPB), 및 출력 크롭핑을 포함한다. 각각의 비트스트림 적합성 테스트에 대해, CPB 사이즈(비트의 수)는 CpbSize[Htid][ScIdx]이고, 각각의 레이어에 대한 DPB 파라미터 max_dec_pic_buffering_minus1[Htid], max_num_reorder_pics[Htid], 및 MaxLatencyPictures[Htid]는, 레이어가 독립 레이어인지의 여부 및 레이어가 타깃 OLS의 출력 레이어인지의 여부에 따라 레이어에 적용되는 dpb_parameters() 신택스 구조에서 발견되거나 또는 그로부터 유도된다.

[0185] HRD는 다음과 같이 동작할 수도 있다. HRD는 디코딩 단위 0에서 초기화되는데, CPB 및 DPB의 각각의 서브 DPB는 둘 모두는 비어 있는 것으로 설정된다(각각의 서브 DPB에 대한 서브 DPB 충만도(fullness)는 0과 동일하게 설정됨). 초기화 이후, HRD는 후속하는 버퍼링 기간 SEI 메시지에 의해 다시 초기화되지 않을 수도 있다. 명시된 도달 스케줄에 따라 각각의 CPB로 흐르는 디코딩 단위와 관련되는 데이터는 가상 스트림 스케줄러(hypothetical stream scheduler; HSS)에 의해 전달된다. 각각의 디코딩 유닛과 관련되는 데이터는 디코딩 유닛의 CPB 제거 시간에서 순시 디코딩 프로세스에 의해 즉시 제거 및 디코딩된다. 각각의 디코딩된 픽처는 DPB에서 배치된다. 디코딩된 픽처는, 디코딩된 픽처가 인터 예측 참조를 위해 더 이상 필요로 되지 않고 출력을 위해 더 이상 필요로 되지 않을 때, DPB로부터 제거된다.

[0186] 디코딩된 픽처 버퍼의 예시적인 동작은 다음과 같다. 이들 명세는 선택되는 디코딩된 픽처 버퍼(DPB) 파라미터의 각각의 세트에 독립적으로 적용될 수도 있다. 디코딩된 픽처 버퍼는, 개념적으로, 서브 DPB를 포함하고, 각각의 서브 DPB는 하나의 레이어의 디코딩된 픽처의 저장을 위한 픽처 저장 버퍼를 포함한다. 픽처 저장 버퍼 각각은, 참조용으로 사용됨으로 마킹되는 또는 나중의 출력을 위해 유지되는 디코딩된 픽처를 포함할 수도 있다. 본원에서 설명되는 프로세스는, OLS의 최하위 레이어로부터 시작하여, OLS에서 레이어의 nuh_layer_id 값의 증가하는 순서로, 순차적으로 적용되며, 각각의 레이어에 대해 독립적으로 적용된다. 이들 프로세스가 특정한 레이어에 적용되는 경우, 특정한 레이어에 대한 서브 DPB만이 영향을 받는다. 이들 프로세스의 설명에서, DPB는 특정한 레이어에 대한 서브 DPB를 참조하고, 특정한 레이어는 현재 레이어로서 지칭된다.

[0187] 출력 타이밍 DPB의 동작에서, 동일한 액세스 단위에서 1과 동일한 PicOutputFlag를 갖는 디코딩된 픽처는 디코딩된 픽처의 nuh_layer_id 값의 오름차순으로 연속적으로 출력된다. 픽처 n 및 현재 픽처가 nuh_layer_id의 특정한 값에 대한 액세스 단위 n - 여기서 n은 음이 아닌 정수임 - 의 코딩된 픽처 또는 디코딩된 픽처라고 하자. 현재 픽처의 디코딩 이전에 DPB로부터 픽처를 제거하는 것은 다음과 같이 발생한다. 현재 픽처의 디코딩 이전에 (그러나 현재 픽처의 제1 슬라이스의 슬라이스 헤더를 파싱한 이후) DPB로부터의 픽처의 제거는 (현재 픽처를 포함하는) 액세스 단위 n의 제1 디코딩 단위의 CPB 제거 시간에 실질적으로 순시적으로 발생하고 다음과 같이 진행한다.

[0188] 참조 픽처 목록 구성을 위한 디코딩 프로세스가 호출되고 참조 픽처 마킹을 위한 디코딩 프로세스가 호출된다. 현재 AU가 AU 0이 아닌 코딩된 비디오 시퀀스 시작(coded video sequence start; CVSS) AU인 경우, 다음의 순서화된 단계가 적용된다. 변수 NoOutputOfPriorPicsFlag수는 다음과 같이 테스트 하에 있는 디코더에 대해 유도된다. 현재 AU의 임의의 픽처에 대해 유도되는 pic_width_max_in_luma_samples, pic_height_max_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 또는 max_dec_pic_buffering_minus1[Htid]의 값이, 동일한 CLVS의 선행 픽처에 대해 유도되는 pic_width_in_luma_samples, pic_height_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 또는 max_dec_pic_buffering_minus1[Htid]의 값과, 각각, 상이한 경우, no_output_of_prior_pics_flag의 값에 관계없이, NoOutputOfPriorPicsFlag는 테스트 하에 있는 디코더에 의해 1로 설정될 수도 있다. NoOutputOfPriorPicsFlag를 no_output_of_prior_pics_flag와 동일하게 설정하는 것이 이들 조건 하에서 선호될 수도 있지만, 테스트 하에 있는 디코더는 이 경우 NoOutputOfPriorPicsFlag를 1로 설정하도록 허용된다. 그렇지 않으면, NoOutputOfPriorPicsFlag는 no_output_of_prior_pics_flag와 동일하게 설정된다.

[0189] 테스트 하에 있는 디코더에 대해 유도되는 NoOutputOfPriorPicsFlag의 값은 HRD에 대해 적용되고, 그 결과, NoOutputOfPriorPicsFlag의 값이 1과 동일한 경우, DPB의 모든 픽처 저장 버퍼는, 그들이 포함하는 픽처의 출력 없이 비워지고, DPB 충만도는 0과 동일하게 설정된다. DPB의 임의의 픽처 k에 대해 다음의 조건 둘 모두가 참인 경우, DPB의 모든 그러한 픽처 k는 DPB로부터 제거된다. 픽처 k는 참조용으로 사용되지 않음으로 마킹될 수 있거나, 또는 픽처 k는 0과 동일한 PictureOutputFlag를 가질 수 있거나 또는 DPB 출력 시간은 현재 픽처 n의 제1 디코딩 단위(디코딩 단위 m으로 나타내어짐)의 CPB 제거 시간 이하인데, 여기서 DpbOutputTime[k]는 DuCpbRemovalTime[m] 이하이다. DPB로부터 제거되는 각각의 픽처에 대해, DPB 충만도는 1만큼 감분된다.

[0190] 출력 순서 DPB의 동작은 다음과 같을 수도 있다. 이들 프로세스는 선택되는 디코딩된 픽처 버퍼(DPB) 파라미터의 각각의 세트에 독립적으로 적용될 수도 있다. 디코딩된 픽처 버퍼는 개념적으로 서브 DPB를 포함하고, 각각의 서브 DPB는 하나의 레이어의 디코딩된 픽처의 저장을 위한 픽처 저장 버퍼를 포함한다. 픽처 저장 버퍼 각각은 참조용으로 사용됨으로 마킹되거나 또는 향후 출력을 위해 유지되는 디코딩된 픽처를 포함한다. 현재 픽처의 디코딩 이전에 DPB로부터 픽처의 출력 및 제거를 위한 프로세스가 호출되고, 현재 디코딩된 픽처 마킹 및 저장을 위한 프로세스의 호출이 후속되고, 마지막으로, 추가적인 범핑을 위한 프로세스의 호출이 후속된다. 이들 프로세스는, OLS의 최하위 레이어로부터 시작하여, OLS의 레이어의 nuh_layer_id 값의 증가하는 순서로, 각각의 레이어에 대해 독립적으로 적용된다. 이들 프로세스가 특정한 레이어에 적용되는 경우, 특정한 레이어에 대한 서브 DPB만이 영향을 받는다.

[0191] 출력 순서 DPB의 동작에서, 출력 타이밍 DPB의 동작에서와 동일하게, 동일한 액세스 단위에서 1과 동일한 PicOutputFlag를 갖는 디코딩된 픽처는 디코딩된 픽처의 nuh_layer_id 값의 오름차순으로 연속적으로 또한 출력된다. 픽처 n 및 현재 픽처가 nuh_layer_id의 특정한 값에 대한 액세스 단위 n - 여기서 n은 음이 아닌 정수임 - 의 코딩된 픽처 또는 디코딩된 픽처라고 하자. DPB로부터의 픽처의 출력 및 제거는 다음과 같이 설명된다.

[0192] 현재 픽처의 디코딩 이전에(그러나 현재 픽처의 제1 슬라이스의 슬라이스 헤더를 파싱한 이후에) DPB로부터의 픽처의 출력 및 제거는, 현재 픽처를 포함하는 액세스 단위의 제1 디코딩 단위가 CPB로부터 제거될 때 실질적으로 순시적으로 발생하고 다음과 같이 진행한다. 참조 픽처 목록 구성을 위한 디코딩 프로세스 및 참조 픽처 마킹을 위한 디코딩 프로세스가 호출된다. 현재 AU가 AU 0이 아닌 CVSS AU인 경우, 다음의 순서화된 단계가 적용된다. 변수 NoOutputOfPriorPicsFlag수는 다음과 같이 테스트 하에 있는 디코더에 대해 유도된다. 현재 AU의 임의의 픽처에 대해 유도되는 pic_width_max_in_luma_samples, pic_height_max_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 또는 max_dec_pic_buffering_minus1[Htid]의 값이, 동일한 CLVS의 선행 픽처에 대해 유도되는 pic_width_in_luma_samples, pic_height_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 또는 max_dec_pic_buffering_minus1[Htid]의 값과, 각각, 상이한 경우, no_output_of_prior_pics_flag의 값에 관계없이, NoOutputOfPriorPicsFlag는 테스트 하에 있는 디코더에 의해 1로 설정될 수도 있다.

[0193] NoOutputOfPriorPicsFlag를 no_output_of_prior_pics_flag와 동일하게 설정하는 것이 이들 조건 하에서 선호될 수도 있지만, 테스트 하에 있는 디코더는 이 경우 NoOutputOfPriorPicsFlag를 1로 설정하도록 허용된다. 그렇지 않으면, NoOutputOfPriorPicsFlag는 no_output_of_prior_pics_flag와 동일하게 설정된다. 테스트 하에 있는 디코더에 대해 유도되는 NoOutputOfPriorPicsFlag의 값은 다음과 같이 HRD에 대해 적용된다. NoOutputOfPriorPicsFlag가 1과 동일한 경우, DPB의 모든 픽처 저장 버퍼는 그들이 포함하는 픽처의 출력 없이 비워지고 DPB 충만도는 0과 동일하게 설정된다. 그렇지 않으면(NoOutputOfPriorPicsFlag가 0과 동일함), 출력을 위해 필요로 되지 않음 및 참조용으로 사용되지 않음으로 마킹되는 픽처를 포함하는 모든 픽처 저장 버퍼는 (출력 없이) 비워지고 DPB의 모든 비어 있지 않은 픽처 저장 버퍼는 범핑을 반복적으로 호출하는 것에 의해 비워지고 DPB 충만도는 0과 동일하게 설정된다.

[0194] 그렇지 않으면(현재 픽처가 CLVSS 픽처가 아님), 출력을 위해 필요로 되지 않음 및 참조용으로 사용되지 않음으로 마킹되는 픽처를 포함하는 모든 픽처 저장 버퍼는 (출력 없이) 비워진다. 비워지는 각각의 픽처 저장 버퍼에 대해, DPB 충만도는 1만큼 감분된다. 다음의 조건 중 하나 이상이 참인 경우, 범핑 프로세스는, 다음의 조건 중 어느 것도 참이 아닐 때까지, 비워지는 각각의 추가적인 픽처 저장 버퍼에 대해 DPB 충만도를 1씩 추가로 감분시키면서 반복적으로 호출된다. 출력을 위해 필요로 됨으로 마킹되는 DPB에서의 픽처의 개수는 max_num_reorder_pics[Htid]보다 더 크다. max_latency_increase_plus1[Htid]는 0과 동일하지 않고 관련된 변수 PicLatencyCount가 MaxLatencyPictures[Htid] 이상인, 출력을 위해 필요로 됨으로 마킹되는 적어도 하나의 픽처가 DPB에서 존재한다. DPB에서의 픽처의 개수는 max_dec_pic_buffering_minus1[Htid] + 1 이상이다.

[0195] 한 예에서, 추가적인 범핑이 다음과 같이 발생할 수도 있다. 명시되는 프로세스는, 현재 픽처를 포함하는 액세스 단위 n의 마지막 디코딩 단위가 CPB로부터 제거될 때 실질적으로 순시적으로 발생할 수도 있다. 현재 픽처가 1과 동일한 PictureOutputFlag를 갖는 경우, 출력을 위해 필요로 됨으로 마킹되고 출력 순서에서 현재 픽처를 뒤따르는 DPB의 각각의 픽처에 대해, 관련된 변수 PicLatencyCount는 PicLatencyCount + 1과 동일하게 설정된다. 다음의 것이 또한 적용된다. 현재 디코딩된 픽처가 1과 동일한 PictureOutputFlag를 갖는 경우, 현재 디코딩된 픽처는 출력을 위해 필요로 됨으로 마킹되고 관련 변수 PicLatencyCount는 0과 동일하게 설정된다. 그렇지 않으면(현재 디코딩된 픽처가 0과 동일한 PictureOutputFlag를 가짐), 현재 디코딩된 픽처는 출력을 위해 필요로 되지 않음으로 마킹된다.

[0196] 다음의 조건 중 하나 이상이 참인 경우, 다음의 조건 중 어느 것도 참이 아닐 때까지 범핑 프로세스는 반복적으로 호출된다. 출력을 위해 필요로 됨으로 마킹되는 DPB에서의 픽처의 개수는 max_num_reorder_pics[Htid]보다 더 크다. max_latency_increase_plus1[Htid]는 0과 동일하지 않고 관련된 변수 PicLatencyCount가 MaxLatencyPictures[Htid] 이상인, 출력을 위해 필요로 됨으로 마킹되는 적어도 하나의 픽처가 DPB에서 존재한다.

[0197] 범핑 프로세스는 다음의 순서화된 단계를 포함한다. 먼저, 출력을 위한 픽처 또는 픽처들은, 출력을 위해 필요로 됨으로 마킹되는 DPB 내의 모든 픽처 중 PicOrderCntVal의 가장 작은 값을 갖는 것으로서 선택된다. 이들 픽처 각각은, 오름차순 nuh_layer_id 순서로, 픽처에 대한 적합성 크롭핑 윈도우를 사용하여 크롭되고, 크롭된 픽처는 출력되고, 픽처는 출력을 위해 필요로 되지 않음으로 마킹된다. 참조용으로 사용되지 않음으로 마킹되며 크롭되어 출력되는 픽처 중 하나였던 픽처를 포함하는 픽처 저장 버퍼 각각은 비워지고 관련된 서브 DPB의 충만도는 1만큼 감분된다. 동일한 CVS에 속하며 범핑 프로세스에 의해 출력되는 임의의 두 개의 픽처(picA 및 picB)에 대해, picA가 picB보다 더 빨리 출력되는 경우, picA의 PicOrderCntVal의 값은 picB의 PicOrderCntVal의

값보다 더 작다.

[0198] 예시적인 서브 비트스트림 추출 프로세스는 다음과 같다. 이 프로세스에 대한 입력은 비트스트림(inBitstream), 타깃 OLS 인덱스(targetOlsIdx), 및 타깃 최고 TemporalId 값(tIdTarget)이다. 이 프로세스의 출력은 서브 비트스트림(outBitstream)이다. 비트스트림 적합성은, 임의의 입력 비트스트림에 대해, 비트스트림, VPS에 의해 명시되는 OLS의 목록에 대한 인덱스와 동일한 targetOlsIdx, 및 0 이상 6 이하의 범위 내의 임의의 값과 동일한 tIdTarget을, 입력으로서, 갖는 이 프로세스로부터의 출력이며, 다음의 조건을 충족하는 출력 서브 비트스트림은 적합한 비트스트림이어야 한다는 것을 규정할 수도 있다. 출력 서브 비트스트림은 LayerIdInOls[TargetOlsIdx]의 nuh_layer_id 값 각각과 동일한 nuh_layer_id를 갖는 적어도 하나의 VCL NAL 단위를 포함한다. 출력 서브 비트스트림은 tIdTarget와 동일한 TemporalId를 갖는 적어도 하나의 VCL NAL 단위를 포함한다. 적합한 비트스트림은 0과 동일한 TemporalId를 갖는 하나 이상의 코딩된 슬라이스 NAL 단위를 포함하지만, 그러나 0과 동일한 nuh_layer_id를 갖는 코딩된 슬라이스 NAL 단위를 포함할 필요는 없다.

[0199] 출력 서브 비트스트림(OutBitstream)은 다음과 같이 유도된다. 비트스트림(outBitstream)은 비트스트림(inBitstream)과 동일하게 설정된다. tIdTarget보다 더 큰 TemporalId를 갖는 모든 NAL 단위는 outBitstream으로부터 제거된다. 목록 LayerIdInOls[TargetOlsIdx]에 포함되지 않는 nuh_layer_id를 갖는 모든 NAL 단위는 outBitstream으로부터 제거된다. 1과 동일한 nesting_ols_flag를 가지며 0 이상 nesting_num_olss_minus1 이하의 범위 내의 i의 값이 존재하지 않는, 그 결과, NestingOlsIdx[i]가 targetOlsIdx와 동일한 스케일러블 내포 SEI 메시지를 포함하는 모든 SEI NAL 단위는 outBitstream으로부터 제거된다. targetOlsIdx가 0보다 더 큰 경우, 0(버퍼링 기간), 1(픽쳐 타이밍), 또는 130(디코딩 단위 정보)과 동일한 payloadType을 갖는 비 스케일러블 내포 SEI 메시지(non-scalable-nested SEI message)를 포함하는 모든 SEI NAL 단위는 outBitstream으로부터 제거된다.

[0200] 예시적인 스케일러블 내포 SEI 메시지 신택스는 다음과 같다.

scalable_nesting(payloadSize) {	디스크립터
nesting_ols_flag	u(1)
if(nesting_ols_flag) {	
nesting_num_olss_minus1	ue(v)
for(i = 0; i <= nesting_num_olss_minus1; i++)	
nesting_ols_idx_delta_minus1[i]	ue(v)
} else {	
nesting_all_layers_flag	u(1)
if(!nesting_all_layers_flag) {	
nesting_num_layers_minus1	ue(v)
for(i = 1; i <= nesting_num_layers_minus1; i++)	
nesting_layer_id[i]	u(6)
}	
}	
nesting_num_seis_minus1	ue(v)
while(!byte_aligned( ))	
nesting_zero_bit /* equal to 0 */	u(1)
for(i = 0; i <= nesting_num_seis_minus1; i++)	
sei_message( )	
}	

[0201]

[0202] 예시적인 일반적 SEI 페이로드 의미론은 다음과 같다. 다음의 것이 비 스케일러블 내포 SEI 메시지의 적용 가능한 레이어 또는 OLS에 대해 적용된다. 비 스케일러블 내포 SEI 메시지의 경우, payloadType이 0(버퍼링 기간), 1(픽쳐 타이밍), 또는 130(디코딩 단위 정보)과 동일한 경우, 비 스케일러블 내포 SEI 메시지는 오로지 0 번째 OLS에만 적용된다. 비 스케일러블 내포 SEI 메시지의 경우, payloadType이 VclAssociatedSeiList 중 임의의 값과 동일한 경우, 비 스케일러블 내포 SEI 메시지는, VCL NAL 단위가 SEI 메시지를 포함하는 SEI NAL 단위의 nuh_layer_id와 동일한 nuh_layer_id를 레이어에만 적용된다.

[0203] 비트스트림 적합성은 SEI NAL 단위의 nuh_layer_id의 값에 대해 다음의 제한이 적용되어야 한다는 것을 규정할 수도 있다. 비 스케일러블 내포 SEI 메시지가 0(버퍼링 주기), 1(픽쳐 타이밍), 또는 130(디코딩 단위 정보)과 동일한 payloadType을 갖는 경우, 비 스케일러블 내포 SEI 메시지를 포함하는 SEI NAL 단위는 vps_layer_id[0]과 동일한 nuh_layer_id를 가져야 한다. 비 스케일러블 내포 SEI 메시지가 VclAssociatedSeiList 중 임의의 값과 동일한 payloadType을 갖는 경우, 비 스케일러블 내포 SEI 메시지를 포함하는 SEI NAL 단위는 SEI NAL 단위와 관련되는 VCL NAL 단위의 nuh_layer_id의 값과 동일한 nuh_layer_id를 가져야 한다. 스케일러블 내포 SEI 메시지를 포함하는 SEI NAL 단위는, 스케일러블 내포 SEI 메시지가 적용되는 모든 레이어의 nuh_layer_id의 가장 낮은 값(스케일러블 내포 SEI 메시지의 nesting_ols_flag가 0과 동일한 경우) 또는 스케일러블 내포 SEI 메시지가 적용되는 OLS의 모든 레이어의 nuh_layer_id의 가장 낮은 값(스케일러블 내포 SEI 메시지의 nesting_ols_flag가 1인 경우)과 동일한 nuh_layer_id를 가져야 한다.

[0204] 예시적인 스케일러블 내포 SEI 메시지 의미론은 다음과 같다. 스케일러블 내포 SEI 메시지는 SEI 메시지를 특정

한 OLS와 또는 특정한 레이어와 관련시키기 위한 메커니즘을 제공한다. 스케일러블 내포 SEI 메시지는 하나 이상의 SEI 메시지를 포함한다. 스케일러블 내포 SEI 메시지에 포함되는 SEI 메시지는 스케일러블 내포 SEI 메시지로서 또한 지칭된다. 비트스트림 적합성은 스케일러블 내포 SEI 메시지에서의 SEI 메시지의 포함에 대해 다음의 제한이 적용되어야 한다는 것을 규정할 수도 있다.

[0205] 132(디코딩된 픽쳐 해시) 또는 133(스케일러블 내포)과 동일한 payloadType을 갖는 SEI 메시지는, 스케일러블 내포 SEI 메시지에 포함되지 않을 수도 있다. 스케일러블 내포 SEI 메시지가 버퍼링 기간, 픽쳐 타이밍, 또는 디코딩 단위 정보 SEI 메시지를 포함하는 경우, 스케일러블 내포 SEI 메시지는 0(버퍼링 기간), 1(픽쳐 타이밍), 또는 130(디코딩 단위 정보)과 동일하지 않은 payloadType을 갖는 어떠한 다른 SEI 메시지도 포함하지 않아야 한다.

[0206] 비트스트림 적합성은, 스케일러블 내포 SEI 메시지를 포함하는 SEI NAL 단위의 nal_unit_type의 값에 대해 다음의 제한이 적용되어야 한다는 것을 규정할 수도 있다. 스케일러블 내포 SEI 메시지가 0(버퍼링 기간), 1(픽쳐 타이밍), 130(디코딩 단위 정보), 145(종속 RAP 지시), 또는 168(프레임 필드 정보)과 동일한 payloadType을 갖는 SEI 메시지를 포함하는 경우, 스케일러블 내포 SEI 메시지를 포함하는 SEI NAL 단위는 PREFIX_SEI_NUT와 동일한 nal_unit_type을 가져야 한다.

[0207] 1과 동일한 nesting_ols_flag는, 스케일러블 내포 SEI 메시지가 특정한 OLS에 적용된다는 것을 명시한다. 0과 동일한 nesting_ols_flag는 스케일러블 내포 SEI 메시지가 특정한 레이어에 적용된다는 것을 명시한다. 비트스트림 적합성은 nesting_ols_flag의 값에 대해 다음의 제한이 적용되어야 한다는 것을 규정할 수도 있다. 스케일러블 내포 SEI 메시지가 0(버퍼링 기간), 1(픽쳐 타이밍), 또는 130(디코딩 단위 정보)과 동일한 payloadType을 갖는 SEI 메시지를 포함하는 경우, nesting_ols_flag의 값은 1과 동일해야 한다. 스케일러블 내포 SEI 메시지가 VclAssociatedSeiList의 값과 동일한 payloadType을 갖는 SEI 메시지를 포함하는 경우, nesting_ols_flag의 값은 0과 동일해야 한다. nesting_num_olss_minus1 플러스 1은 스케일러블 내포 SEI 메시지가 적용되는 OLS의 개수를 명시한다. nesting_num_olss_minus1의 값은 0 이상 TotalNumOlss - 1 이하의 범위 내에 있어야 한다.

[0208] nesting_ols_idx_delta_minus1[i]는, nesting_ols_flag가 1과 동일할 때 스케일러블 내포 SEI 메시지가 적용되는 i 번째 OLS의 OLS 인덱스를 명시하는 변수 NestingOlsIdx[i]를 유도하기 위해 사용된다. nesting_ols_idx_delta_minus1[i]의 값은 0 이상 TotalNumOlss - 2 이하의 범위 내에 있어야 한다. 변수 NestingOlsIdx[i]는 다음과 같이 유도될 수도 있다.

```
if( i == 0 )
    NestingOlsIdx[ i ] = nesting_ols_idx_delta_minus1[ i ]
else
    NestingOlsIdx[ i ] = NestingOlsIdx[ i − 1 ] + nesting_ols_idx_delta_minus1[ i ] + 1
```

[0209]

[0210] 1과 동일한 nesting_all_layers_flag는, 현재 SEI NAL 단위의 nuh_layer_id 이상의 nuh_layer_id를 갖는 모든 레이어에 스케일러블 내포 SEI 메시지가 적용된다는 것을 명시한다. 0과 동일한 nesting_all_layers_flag는, 현재 SEI NAL 단위의 nuh_layer_id 이상의 nuh_layer_id를 갖는 모든 레이어에 스케일러블 내포 SEI 메시지가 적용될 수도 있거나 또는 적용되지 않을 수도 있다는 것을 명시한다. nesting_num_layers_minus1 플러스 1은 스케일러블 내포 SEI 메시지가 적용되는 레이어의 개수를 명시한다. nesting_num_layers_minus1의 값은 0 이상 vps_max_layers_minus1 - GeneralLayerIdx[nuh_layer_id] 이하의 범위 내에 있어야 하는데, 여기서 nuh_layer_id는 현재 SEI NAL 단위의 nuh_layer_id이다. nesting_layer_id[i]는, nesting_all_layers_flag가 0과 동일할 때 스케일러블 내포 SEI 메시지가 적용되는 i 번째 레이어의 nuh_layer_id 값을 명시한다. nesting_layer_id[i]의 값은 nuh_layer_id보다 더 커야 하는데, 여기서 nuh_layer_id는 현재 SEI NAL 단위의 nuh_layer_id이다.

[0211] nesting_ols_flag가 0과 동일한 경우, 스케일러블 내포 SEI 메시지가 적용되는 레이어의 개수를 명시하는 변수 NestingNumLayers, 및 스케일러블 내포 SEI 메시지가 적용되는 레이어의 nuh_layer_id 값의 목록을 명시하는, 0 이상 NestingNumLayers - 1 이하의 범위 내의 i에 대한 목록 NestingLayerId[i]는 다음과 같이 유도될 수도 있는데, 여기서 nuh_layer_id는 현재 SEI NAL 단위의 nuh_layer_id이다.

```
if( nesting_all_layers_flag ) {
        NestingNumLayers = vps_max_layers_minus1 + 1 − GeneralLayerIdx[ nuh_layer_id ]
        for( i = 0; i < NestingNumLayers; i ++)
                NestingLayerId[ i ] = vps_layer_id[ GeneralLayerIdx[ nuh_layer_id ] + i ]
} else {
        NestingNumLayers = nesting_num_layers_minus1 + 1
        for( i = 0; i < NestingNumLayers; i ++)
                NestingLayerId[ i ] = ( i  = = 0 ) ? nuh_layer_id : nesting_layer_id[ i ]
}
```

[0212]

[0213] nesting_num_seis_minus1 플러스 1은 스케일러블 내포 SEI 메시지의 개수를 명시한다. nesting_num_seis_minus1의 값은 0 이상 63 이하의 범위 내에 있어야 한다. nesting_zero_bit는 0과 동일해야 한다.

[0214] 도 9는 예시적인 비디오 코딩 디바이스(900)의 개략도이다. 비디오 코딩 디바이스(900)는 본원에서 설명되는 바와 같이 개시된 예/실시예를 구현하기에 적합하다. 비디오 코딩 디바이스(900)는, 다운스트림 포트(920), 업스트림 포트(950), 및/또는 네트워크를 통해 데이터 업스트림 및/또는 다운스트림을 통신하기 위한 송신기 및/또는 수신기를 포함하는 트랜스시버 유닛(Tx/Rx)(910)을 포함한다. 비디오 코딩 디바이스(900)는, 데이터를 프로세싱하기 위한 로직 유닛 및/또는 중앙 프로세싱 유닛(central processing unit; CPU) 및 데이터를 저장하기 위한 메모리(932)를 포함하는 프로세서(930)를 또한 포함한다. 비디오 코딩 디바이스(900)는 전기, 광학 대 전기(optical-to-electrical; OE) 컴포넌트, 전기 대 광학(electrical-to-optical; EO) 컴포넌트, 및/또는 전기, 광학, 또는 무선 통신 네트워크를 통한 데이터의 통신을 위한 업스트림 포트(950) 및/또는 다운스트림 포트(920)에 커플링되는 무선 통신 컴포넌트를 또한 포함할 수도 있다. 비디오 코딩 디바이스(900)는 유저에게 그리고 유저로부터 데이터를 전달하기 위한 입력 및/또는 출력(I/O) 디바이스(960)를 또한 포함할 수도 있다. I/O 디바이스(960)는 비디오 데이터를 디스플레이하기 위한 디스플레이, 오디오 데이터를 출력하기 위한 스피커, 등등과 같은 출력 디바이스를 포함할 수도 있다. I/O 디바이스(960)는, 키보드, 마우스, 트랙볼, 등등과 같은 입력 디바이스, 및/또는 그러한 출력 디바이스와 상호 작용하기 위한 대응하는 인터페이스를 또한 포함할 수도 있다.

[0215] 프로세서(930)는 하드웨어 및 소프트웨어에 의해 구현된다. 프로세서(930)는 하나 이상의 CPU 칩으로서, 코어로서(예를 들면, 멀티 코어 프로세서), 필드 프로그래머블 게이트 어레이(field-programmable gate array; FPGA)로서, 주문형 집적 회로(application specific integrated circuit; ASIC)로서, 그리고 디지털 신호 프로세서(digital signal processor; DSP)로서 구현될 수도 있다. 프로세서(930)는 다운스트림 포트(920), Tx/Rx(910), 업스트림 포트(950), 및 메모리(932)와 통신한다. 프로세서(930)는 코딩 모듈(914)을 포함한다. 코딩 모듈(914)은, 다중 레이어 비디오 시퀀스(600), RPL 구조(700), 및/또는 비트스트림(800)을 활용할 수도 있는 방법(100, 1000, 및 1100)과 같은, 본원에서 설명되는 개시된 실시예를 구현한다. 코딩 모듈(914)은 본원에서 설명되는 임의의 다른 방법/메커니즘을 또한 구현할 수도 있다. 게다가, 코딩 모듈(914)은 코덱 시스템(200), 인코더(300), 디코더(400), 및/또는 HRD(500)를 구현할 수도 있다. 예를 들면, 코딩 모듈(914)은 최대 디코딩된 픽처 사이즈에 기초하여 RPL 구조의 참조 엔트리의 개수를 제한할 수도 있다. 최대 디코딩된 픽처 버퍼 사이즈는, 레이어가 참조 레이어인지 또는 출력 레이어인지의 여부에 따라 변할 수도 있다. 그러한 만큼, 제약을 적용하는 것에 의해, 코딩 모듈(914)은 상이한 개수의 참조 픽처가 출력 레이어 및 참조 레이어에 대해 활용되는 것을 허용할 수 있다. 이것은 코딩 모듈(914)이 더욱 최적의 메모리 사용량을 제공하는 것 및 코딩 효율성을 증가시키는 것을 허용한다. 따라서, 코딩 모듈(914)은 상기에서 논의되는 문제 중 하나 이상을 해결하기 위한 메커니즘을 수행하도록 구성될 수도 있다. 그러므로, 코딩 모듈(914)은, 비디오 데이터를 코딩할 때 비디오 코딩 디바이스(900)로 하여금 추가적인 기능성 및/또는 코딩 효율성을 제공하게 한다. 그러한 만큼, 코딩 모듈(914)은 비디오 코딩 디바이스(900)의 기능성을 향상시킬 뿐만 아니라, 비디오 코딩 기술에 고유한 문제를 해결한다. 게다가, 코딩 모듈(914)은 상이한 상태로의 비디오 코딩 디바이스(900)의 변환을 실행한다. 대안적으로, 코딩 모듈(914)은 메모리(932)에 저장되며 프로세서(930)에 의해 실행되는 명령어로서(예를 들면, 비일시적 매체 상에 저장되는 컴퓨터 프로그램 제품으로서) 구현될 수 있다.

[0216] 메모리(932)는 디스크, 테이프 드라이브, 솔리드 스테이트 드라이브, 리드 온리 메모리(read only memory; ROM), 랜덤 액세스 메모리(random access memory; RAM), 플래시 메모리, 터너리 콘텐츠 어드레서블 메모리(ternary content-addressable memory; TCAM), 정적 랜덤 액세스 메모리(static random-access memory; SRAM)와 같은 하나 이상의 메모리 타입을 포함한다. 메모리(932)는, 그러한 프로그램이 실행을 위해 선택될 때 프로그램을 저장하기 위해, 그리고 프로그램 실행 동안 판독되는 명령어 및 데이터를 저장하기 위해, 오버플로우 데이터 스토리지 디바이스로서 사용될 수도 있다.

[0217] 도 10은, 최대 디코딩된 픽처 버퍼 사이즈에 따라 참조 엔트리의 개수가 제한되는, RPL 구조(700)와 같은, 참조 픽처 목록 구조에 기초하여, 비트스트림(800)과 같은 비트스트림에 비디오 시퀀스를 인코딩하는 예시적인 방법 방법(1000)의 플로우차트이다. 방법(1000)은, 방법(100)을 수행할 때 코덱 시스템(200), 인코더(300), 및/또는 비디오 코딩 디바이스(900)와 같은 인코더에 의해 활용될 수도 있다. 게다가, 방법(1000)은 HRD(500) 상에서 동작할 수도 있고, 그러므로, 다중 레이어 비디오 시퀀스(600) 및/또는 그 추출된 레이어에 대해 적합성 테스트를 수행할 수도 있다.

[0218] 방법(1000)은, 인코더가 비디오 시퀀스를 수신하고, 그 비디오 시퀀스, 예컨대 다중 레이어 비디오 시퀀스를, 예를 들면, 유저 입력에 기초하여, 비트스트림에 인코딩할 것을 결정하는 경우 시작될 수도 있다. 단계(1001)에서, 인코더는 비디오 시퀀스로부터의 픽처를 비트스트림에 인코딩한다. 예를 들면, 인코더는 인터 예측 및/또는 인터레이어 예측에 따라 현재 픽처를 인코딩할 수 있다. 그러한 만큼, 인코더는 참조 픽처에 기초하여 현재 픽처를 인코딩한다.

[0219] 단계(1003)에서, 인코더는 ref_pic_list_struct()를, 예를 들면, SPS, 픽처 헤더, 및/또는 슬라이스 헤더에서, 비트스트림에 인코딩할 수 있다. ref_pic_list_struct()는 인터 예측에 따라 현재 픽처를 코딩할 때 사용되는 참조 픽처를 나타낸다. 구체적으로, ref_pic_list_struct()는 RefPicList[0] 및 RefPicList[1]을 포함한다. RefPicList[0] 및 RefPicList[1] 각각은 현재 픽처를 코딩하기 위해 사용되는 임의의 참조 픽처를 참조하는 하나 이상의 참조 엔트리를 포함한다. ref_pic_list_struct()는, ref_pic_list_struct(listIdx, rplsIdx)로서 나타내어지는, 목록 인덱스(listIdx) 및 참조 픽처 목록 구조 인덱스(rplsIdx)에 따라 참조될 수 있다. ref_pic_list_struct()는, 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[0] 내의 모든 엔트리 및 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 세트 고유의 픽처인 setOfRefPics를 참조한다. 따라서, setOfRefPics는 현재 픽처와 동일한 레이어 내에 있는 현재 픽처에 대한 모든 참조 픽처를 포함한다. setOfRefPics는 각각의 픽처의 모든 슬라이스에 대해 동일한 세트가 되도록 제한될 수도 있다. 그러한 만큼, setOfRefPics는 현재 픽처의 모든 슬라이스에 대해 동일한 세트가 되도록 제한될 수도 있다. 게다가, ref_pic_list_struct(), 그러므로 비트스트림은 num_ref_entries를 포함한다. num_ref_entries는 num_ref_entries [listIdx][rplsIdx]로서 나타내어질 수도 있다. num_ref_entries [listIdx][rplsIdx]는 ref_pic_list_struct(listIdx, rplsIdx)에서의 엔트리의 개수를 명시한다. num_ref_entries는 0 내지 최대 디코딩된 픽처 버퍼 사이즈 범위 플러스 13 개 또는 14 개의 엔트리와 같은 오프셋의 범위와 같은, 최대 디코딩된 픽처 버퍼 사이즈에 기초한 범위로 제한된다. 게다가, ref_pic_list_struct에서 참조되는 setOfRefPics는, 최대 디코딩된 픽처 버퍼 사이즈 마이너스 1 이하가 되도록 제한된다. 이러한 방식으로, num_ref_entries 및 setOfRefPics 둘 모두는, 모든 레이어에 대해 전역적인 정적인 값에 기초하여 제한되는 것이 아니라, 레이어에 기초하여 변하는 DPB에서 이용 가능한 메모리에 기초하여 제한된다. 이것은, 현재 픽처가 더 많은 메모리 공간을 활용하는 출력 레이어와 관련되는지, 또는 더 적은 공간을 활용하는 참조 레이어와 관련되는지의 여부에 기초하여 이들 값이 변하는 것을 허용한다. 그러한 만큼, 최대 DPB 사이즈에 기초하여 num_ref_entries 및 setOfRefPics를 제한하는 것은 증가된 코딩 효율성(더 나은 압축)을 지원하는데, 이것은 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0220] 인코더는 비트스트림에서, 예를 들면, VPS 및/또는 SPS에서, dpb_parameters()를 또한 인코딩한다. dpb_parameters()는 max_dec_pic_buffering_minus1을 포함할 수 있다. max_dec_pic_buffering_minus1은, DPB의 최대 요구 사이즈를 픽처 저장 버퍼의 단위로 명시한다. 그러한 만큼, num_ref_entries 및 setOfRefPics를 제한하기 위해 사용되는 최대 디코딩된 픽처 버퍼 사이즈는 max_dec_pic_buffering_minus1에 대응한다.

[0221] 단계(1005)에서, 인코더에서의 HRD는 비트스트림에 대해 비트스트림 적합성 테스트의 세트를 수행할 수 있다. 특정한 예로서, HRD는, num_ref_entries 및 setOfRefPics가, 예를 들면, dpb_parameters()의 max_dec_pic_buffering_minus1에 의해 설명되는 바와 같은 최대 DPB 사이즈에 기초하여 제한된다는 것을 확인하기 위해, 비트스트림을 검사할 수 있다. 그 다음, 인코더는 요청시 디코더를 향한 통신을 위해, 단계(1007)에서, 비트스트림을 저장할 수 있다. 예를 들면, 인코더는 비트스트림을 로컬하게 저장할 수 있고 및

/또는 저장을 위해 비트스트림을 콘텐츠 서버로 포워딩할 수 있다. 게다가, 인코더 및/또는 콘텐츠 서버는, 요청시 디코더로의 송신을 위해 레이어 및/또는 OLS에 관련되는 콘텐츠를 분리하기 위해, 비트스트림에 대해 서브 비트스트림 추출을 수행할 수 있다.

[0222] 도 11은, 최대 디코딩된 픽처 버퍼 사이즈에 따라 참조 엔트리의 개수가 제한되는, RPL 구조(700)와 같은, 참조 픽처 목록 구조에 기초하여, 비트스트림(800)과 같은, 비트스트림으로부터 비디오 시퀀스를 디코딩하는 예시적인 방법(1100)의 플로우차트이다. 방법(1100)은, 방법(100)을 수행할 때 코덱 시스템(200), 디코더(400), 및/또는 비디오 코딩 디바이스(900)와 같은 디코더에 의해 활용될 수도 있다. 게다가, 방법(1100)은 HRD(500)와 같은 HRD에 의해 적합성에 대해 검사된 다중 레이어 비디오 시퀀스(600), 또는 그 추출된 레이어에 대해 활용될 수도 있다.

[0223] 방법(1100)은, 디코더가, 예를 들면, 방법(1000)의 결과로서, 다중 레이어 비트스트림의 하나 이상의 레이어로부터 코딩된 레이어 비디오 시퀀스와 같은 코딩된 비디오 시퀀스를 포함하는 비트스트림의 수신을 시작하는 경우 시작될 수도 있다. 단계(1101)에서, 디코더는 비트스트림을 수신한다. 비트스트림은, 시간적으로 명시된 순간에 디코딩되고 있는 임의의 명시된 픽처일 수도 있는 현재 픽처를 포함하는 픽처의 시퀀스를 포함한다. 이 예에서, 현재 픽처는 인터 예측 및/또는 인터레이어 예측에 따라 코딩된다. 그러한 만큼, 인코더는 참조 픽처에 기초하여 현재 픽처를 인코딩한다.

[0224] 비트스트림은, 예를 들면, SPS, 픽처 헤더 및/또는 슬라이스 헤더에서, ref_pic_list_struct()를 또한 포함한다. ref_pic_list_struct()는 인터 예측에 따라 현재 픽처를 코딩할 때 사용되는 참조 픽처를 나타낸다. 구체적으로, ref_pic_list_struct()는 RefPicList[0] 및 RefPicList[1]을 포함한다. RefPicList[0] 및 RefPicList[1] 각각은 현재 픽처를 코딩하기 위해 사용되는 임의의 참조 픽처를 참조하는 하나 이상의 참조 엔트리를 포함한다. ref_pic_list_struct()는, ref_pic_list_struct(listIdx, rplsIdx)로서 나타내어지는, 목록 인덱스(listIdx) 및 참조 픽처 목록 구조 인덱스(rplsIdx)에 따라 참조될 수 있다. ref_pic_list_struct()는, 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[0] 내의 모든 엔트리 및 현재 픽처와 동일한 nuh_layer_id를 갖는 RefPicList[1] 내의 모든 엔트리에 의해 참조되는 세트 고유의 픽처인 setOfRefPics를 참조한다. 따라서, setOfRefPics는 현재 픽처와 동일한 레이어 내에 있는 현재 픽처에 대한 모든 참조 픽처를 포함한다. setOfRefPics는 각각의 픽처의 모든 슬라이스에 대해 동일한 세트가 되도록 제한될 수도 있다. 그러한 만큼, setOfRefPics는 현재 픽처의 모든 슬라이스에 대해 동일한 세트가 되도록 제한될 수도 있다. 게다가, ref_pic_list_struct(), 그러므로 비트스트림은 num_ref_entries를 포함한다. num_ref_entries는 num_ref_entries [listIdx][rplsIdx]로서 나타내어질 수도 있다. num_ref_entries [listIdx][rplsIdx]는 ref_pic_list_struct(listIdx, rplsIdx)에서의 엔트리의 개수를 명시한다. num_ref_entries는 0 내지 최대 디코딩된 픽처 버퍼 사이즈 범위 플러스 13 개 또는 14 개의 엔트리와 같은 오프셋의 범위와 같은, 최대 디코딩된 픽처 버퍼 사이즈에 기초한 범위로 제한된다. 게다가, ref_pic_list_struct에서 참조되는 setOfRefPics는, 최대 디코딩된 픽처 버퍼 사이즈 마이너스 1 이하가 되도록 제한된다. 이러한 방식으로, num_ref_entries 및 setOfRefPics 둘 모두는, 모든 레이어에 적용되는 정적인 값에 기초하여 제한되는 것이 아니라 대응하는 레이어에 대한 DPB에서 이용 가능한 메모리에 기초하여 제한된다. 이것은, 현재 픽처가 더 많은 메모리 공간을 활용하는 출력 레이어와 관련되는지, 또는 더 적은 공간을 활용하는 참조 레이어와 관련되는지의 여부에 기초하여 이들 값이 변하는 것을 허용한다. 그러한 만큼, 최대 DPB 사이즈에 기초하여 num_ref_entries 및 setOfRefPics를 제한하는 것은 증가된 코딩 효율성(더 나은 압축)을 지원하는데, 이것은 프로세서, 메모리, 및/또는 네트워크 시그널링 리소스 사용량을 감소시킨다.

[0225] 비트스트림은, 예를 들면, VPS 및/또는 SPS에서 dpb_parameters()를 더 포함한다. dpb_parameters()는 max_dec_pic_buffering_minus1을 포함할 수 있다. max_dec_pic_buffering_minus1은, DPB의 최대 요구 사이즈를 픽처 저장 버퍼의 단위로 명시한다. 그러한 만큼, num_ref_entries 및 setOfRefPics를 제한하기 위해 사용되는 최대 디코딩된 픽처 버퍼 사이즈는 max_dec_pic_buffering_minus1에 대응한다.

[0226] 단계(1103)에서, 디코더는 ref_pic_list_struct() 및/또는 dpb_parameters()에 기초하여 현재 픽처를 디코딩하여 디코딩된 픽처를 생성할 수 있다. 예를 들면, 디코더는 dpb_parameters() 및/또는 max_dec_pic_buffering_minus1을 활용하여 DPB에서 메모리 공간을 할당할 수 있다. 디코더는, 현재 픽처를 코딩하기 위해 사용되는 참조 픽처를 나타내는 RefPicList[0] 및/또는 RefPicList[1]의 참조 엔트리를 결정하기 위해 현재 픽처에 대한 ref_pic_list_struct()를 또한 파싱할 수 있다. 디코더의 개수는 num_ref_entries에 기초하여 관련 참조 엔트리를 결정할 수 있고 참조 엔트리에 기초하여 setOfRefPics를 결정할 수 있다. 디코더는, ref_pic_list_struct()에 기초하여 장기간 참조를 위해 사용되는 또는 단기간 참조를 위해 사용되는

setOfRefPics의 모든 픽쳐를 마킹하기 위해 참조 픽쳐 마킹 프로세스를 또한 활용할 수 있다. 참조 픽쳐 마킹 프로세스는, setOfRefPics에 포함되지 않는 픽쳐 중 하나 이상을 참조용으로 사용되지 않음으로 마킹할 수도 있다. 그러한 픽쳐는, 일단 그들이 출력을 위해 더 이상 필요로 되지 않으면, DPB로부터 제거될 수도 있다. 일단 참조 픽쳐가 마킹되면, 디코더는 setOfRefPics에 기초하여 현재 픽쳐를 디코딩할 수 있다. 디코더는 참조 픽쳐 마킹 프로세스 이후 다양한 픽쳐의 마킹에 기초하여 DPB를 또한 관리할 수 있다.

[0227] 그 다음, 디코더는, 단계(1105)에서, 디코딩된 비디오 시퀀스의 일부로서 디스플레이를 위해 디코딩된 픽쳐를 포워딩할 수 있다. 예를 들면, 디코더는 디코딩된 픽쳐 및/또는 디코딩된 비디오 시퀀스를, 엔드 유저가 보도록, 스크린, 헤드 마운트형 디스플레이, 또는 다른 디스플레이 디바이스를 향해, 포워딩할 수 있다.

[0228] 도 12는 최대 디코딩된 픽쳐 버퍼 사이즈에 따라 참조 엔트리의 개수가 제한되는, RPL 구조(700)와 같은 참조 픽쳐 목록 구조에 기초하여 비트스트림(800)에 비디오 시퀀스를 코딩하기 위한 예시적인 시스템(1200)의 개략도이다. 시스템(1200)은, 코덱 시스템(200), 인코더(300), 디코더(400), 및/또는 비디오 코딩 디바이스(900)와 같은 인코더 및 디코더에 의해 구현될 수도 있다. 게다가, 시스템(1200)은 다중 레이어 비디오 시퀀스(600) 및/또는 비트스트림(800)에 대한 적합성 테스트를 수행하기 위해 HRD(500)를 활용할 수도 있다. 또한, 시스템(1200)은 방법(100, 1000, 및/또는 1100)을 구현할 때 활용될 수도 있다.

[0229] 시스템(1200)은 비디오 인코더(1202)를 포함한다. 비디오 인코더(1202)는 참조 픽쳐에 기초하여 현재 픽쳐를 비트스트림에 인코딩하기 위한 인코딩 모듈(1205)을 포함한다. 인코딩 모듈(1205)은 또한, 참조 픽쳐를 나타내며 최대 디코딩된 픽쳐 버퍼 사이즈에 기초한 범위로 제한되는 참조 엔트리의 개수(num_ref_entries)를 포함하는 ref_pic_list_struct()를 비트스트림에 인코딩하기 위한 것이다. 비디오 인코더(1202)는, 디코더를 향한 통신을 위해 비트스트림을 저장하기 위한 저장 모듈(1206)을 더 포함한다. 비디오 인코더(1202)는 비디오 디코더(1210)를 향해 비트스트림을 송신하기 위한 송신 모듈(1207)을 더 포함한다. 비디오 인코더(1202)는 방법(1000)의 단계 중 임의의 것을 수행하도록 추가로 구성될 수도 있다.

[0230] 시스템(1200)은 비디오 디코더(1210)를 또한 포함한다. 비디오 디코더(1210)는, 현재 픽쳐를 포함하는 비트스트림 및 최대 디코딩된 픽쳐 버퍼 사이즈에 기초한 범위로 제한되는 num_ref_entries를 포함하는 ref_pic_list_struct()를 수신하기 위한 수신 모듈(1211)을 포함한다. 비디오 디코더(1210)는, ref_pic_list_struct()에 기초하여 현재 픽쳐를 디코딩하여 디코딩된 픽쳐를 생성하기 위한 디코딩 모듈(1213)을 더 포함한다. 비디오 디코더(1210)는, 디코딩된 비디오 시퀀스의 일부로서 디스플레이를 위해 디코딩된 픽쳐를 포워딩하기 위한 포워딩 모듈(1215)을 더 포함한다. 비디오 디코더(1210)는 방법(1100)의 단계 중 임의의 것을 수행하도록 추가로 구성될 수도 있다.

[0231] 제1 컴포넌트와 제2 컴포넌트 사이에 라인, 궤적(trace), 또는 다른 매체를 제외한 개재하는 컴포넌트가 없을 때, 제1 컴포넌트는 제2 컴포넌트에 직접적으로 커플링된다. 제1 컴포넌트와 제2 컴포넌트 사이에 라인, 궤적 또는 다른 매체 이외의 개재하는 컴포넌트가 있는 경우, 제1 컴포넌트는 제2 컴포넌트에 간접적으로 커플링된다. 용어 "커플링되는" 및 그것의 변형어는, 직접적으로 커플링되는 것 및 간접적으로 커플링되는 것 둘 모두를 포함한다. 용어 "약"의 사용은, 달리 언급되지 않는 한, 후속하는 숫자의 ±10 %를 포함하는 범위를 의미한다.

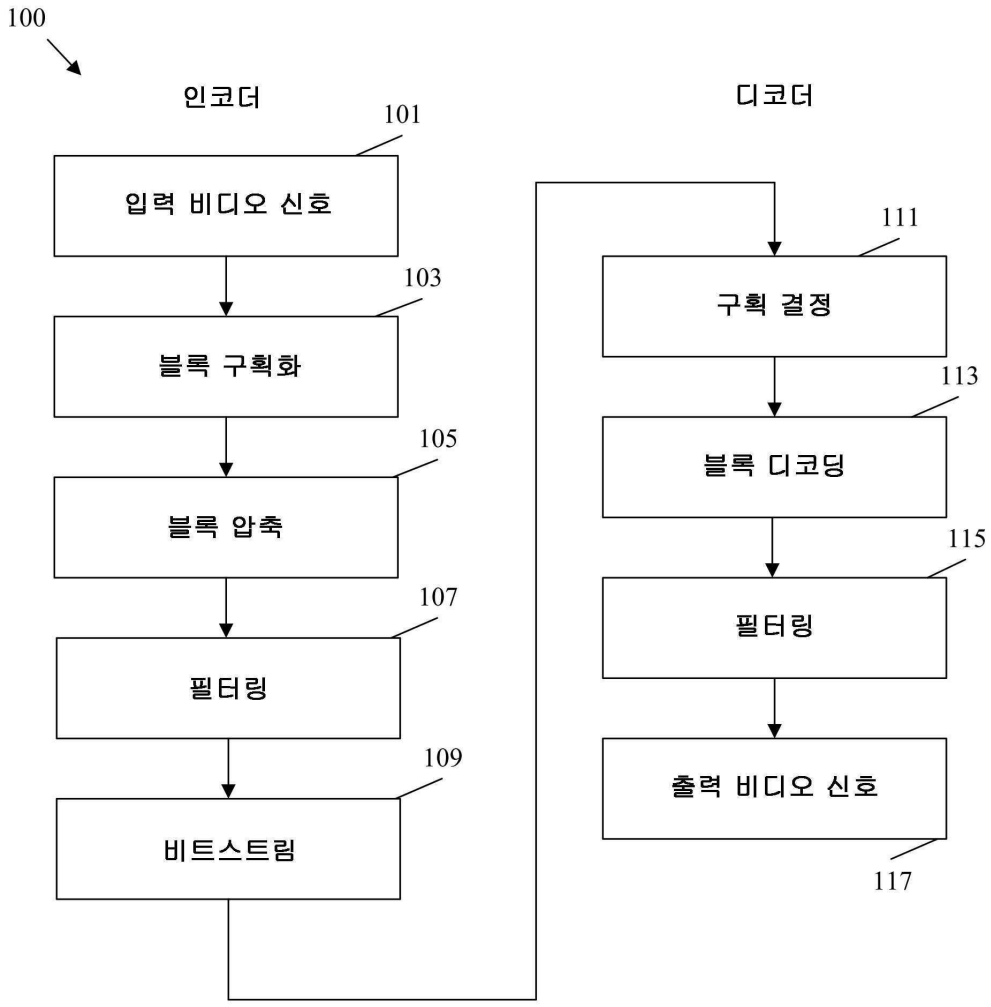
[0232] 또한, 본원에서 기술되는 예시적인 방법의 단계는 반드시 설명되는 순서대로 수행될 필요는 없다는 것이 이해되어야 하며, 그러한 방법의 단계의 순서는 단지 예시적인 것으로 이해되어야 한다. 마찬가지로, 본 개시의 다양한 실시예와 부합하는 방법에서, 추가적인 단계가 그러한 방법에 포함될 수도 있고, 소정의 단계가 생략 또는 조합될 수도 있다.

[0233] 본 개시에서 몇몇 실시예가 제공되었지만, 개시된 시스템 및 방법은 본 개시의 취지 또는 범위로부터 벗어나지 않으면서 많은 다른 특정한 형태로 구체화될 수도 있다는 것이 이해될 수도 있다. 본 예는 제한적인 것이 아니라 예시적인 것으로서 간주되어야 하며, 의도는 본원에서 주어지는 세부 사항으로 제한되지는 않는다. 예를 들면, 다양한 엘리먼트 또는 컴포넌트가 다른 시스템에서 결합 또는 통합될 수도 있거나 또는 소정의 피쳐가 생략될 수도 있거나, 또는 구현되지 않을 수도 있다.

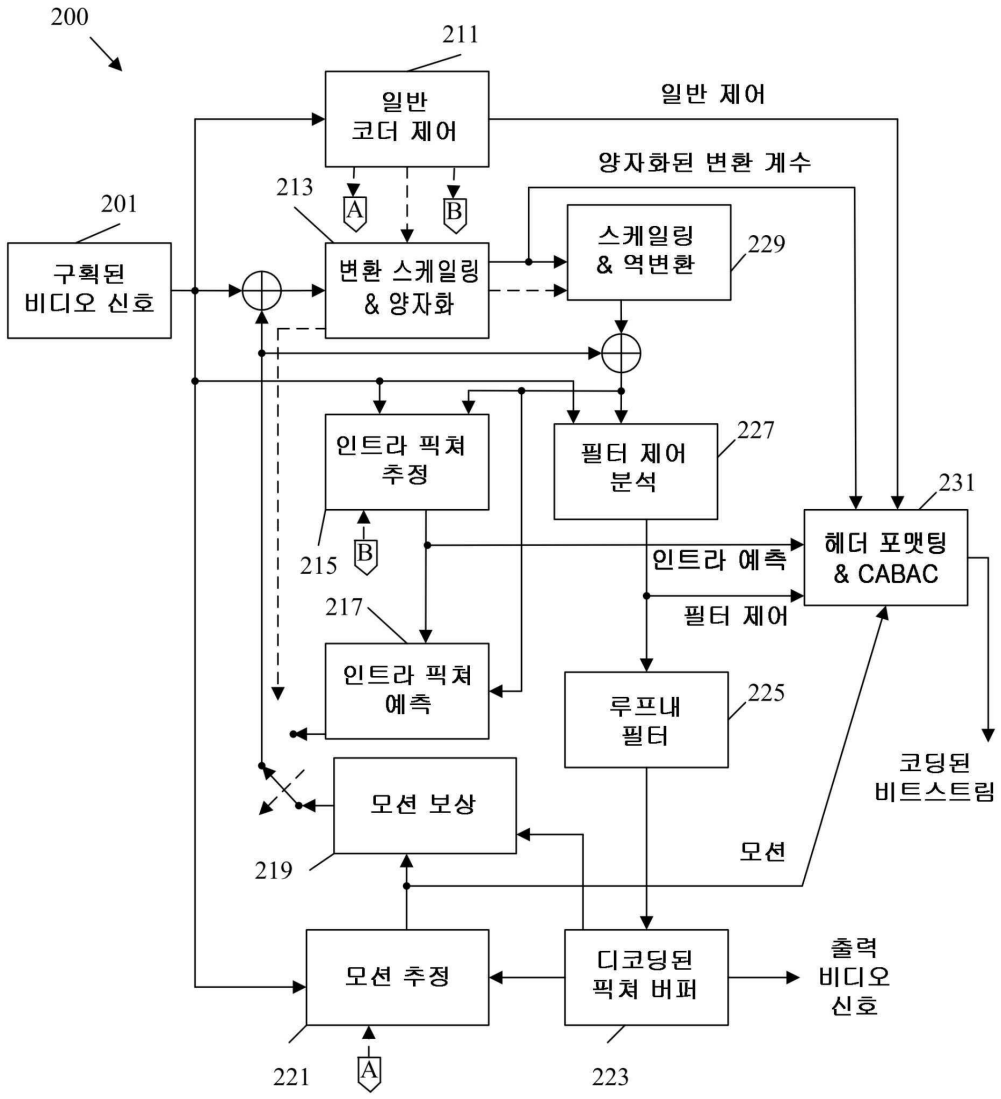
[0234] 또한, 다양한 실시예에서 별개인 것으로 또는 분리된 것으로 설명되고 예시되는 기술, 시스템, 서브시스템, 및 방법은, 본 개시의 범위로부터 벗어나지 않으면서, 다른 시스템, 컴포넌트, 기술, 또는 방법과 결합 또는 통합될 수도 있다. 다른 변경예, 대체예, 및 수정예가 기술 분야에서 통상의 기술을 가진 자에 의해 확인 가능하고, 본원에서 개시되는 취지 및 범위로부터 벗어나지 않으면서 이루어질 수도 있다.

도면

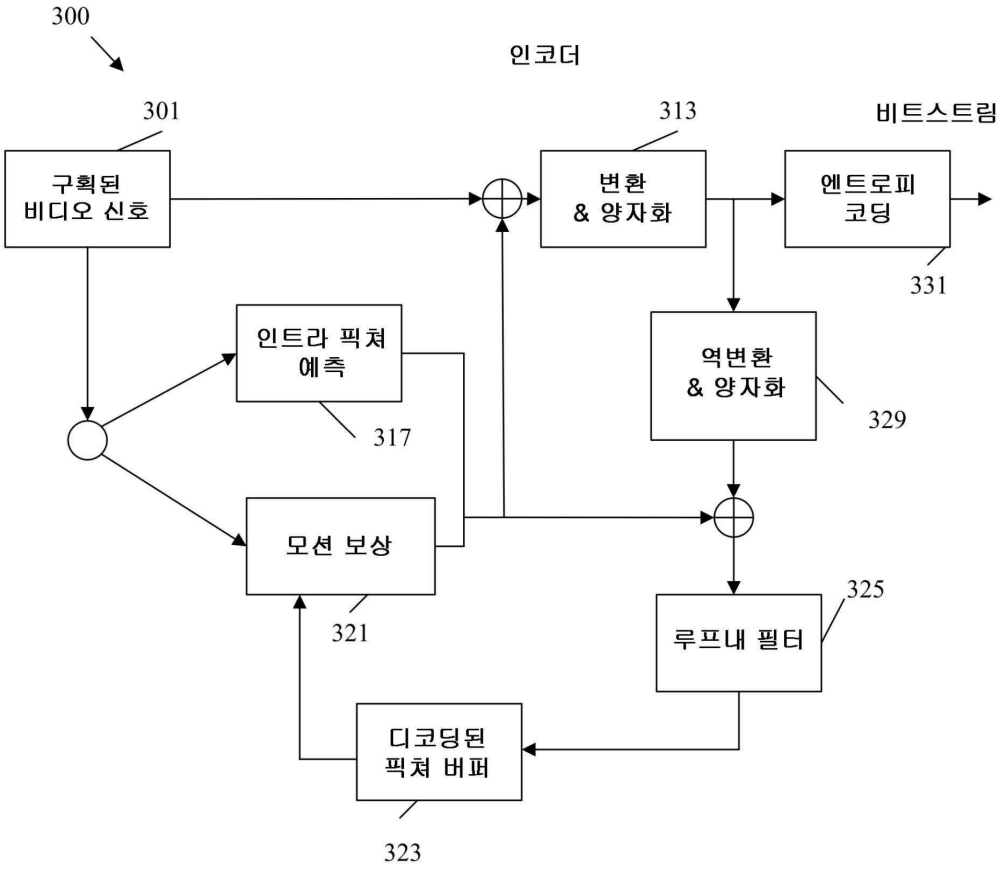
도면1


100
인코더
101
입력 비디오 신호
103
블록 구획화
105
블록 압축
107
필터링
109
비트스트림
디코더
111
구획 결정
113
블록 디코딩
115
필터링
출력 비디오 신호
117

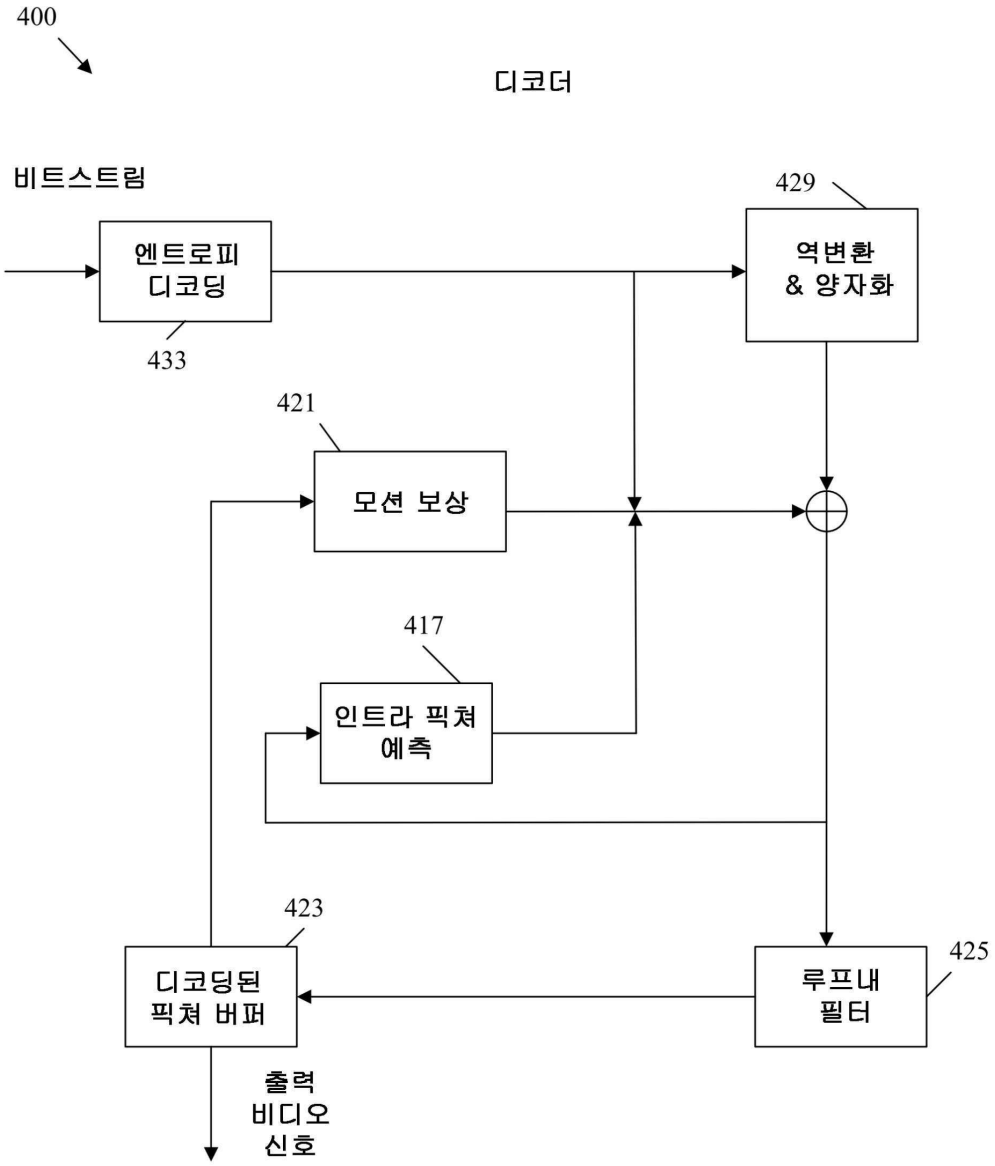
도면2


200
211
일반
코더 제어
일반 제어
양자화된 변환 계수
213
201
A
B
구획된
비디오 신호
변환 스케일링
& 양자화
스케일링
& 역변환
229
인트라 픽쳐
추정
필터 제어
분석
227
231
헤더 포맷팅
& CABAC
215
B
인트라 예측
217
필터 제어
인트라 픽쳐
예측
루프내
필터
225
코딩된
비트스트림
모션 보상
219
모션
모션 추정
디코딩된
픽쳐 버퍼
출력
비디오
신호
221
A
223

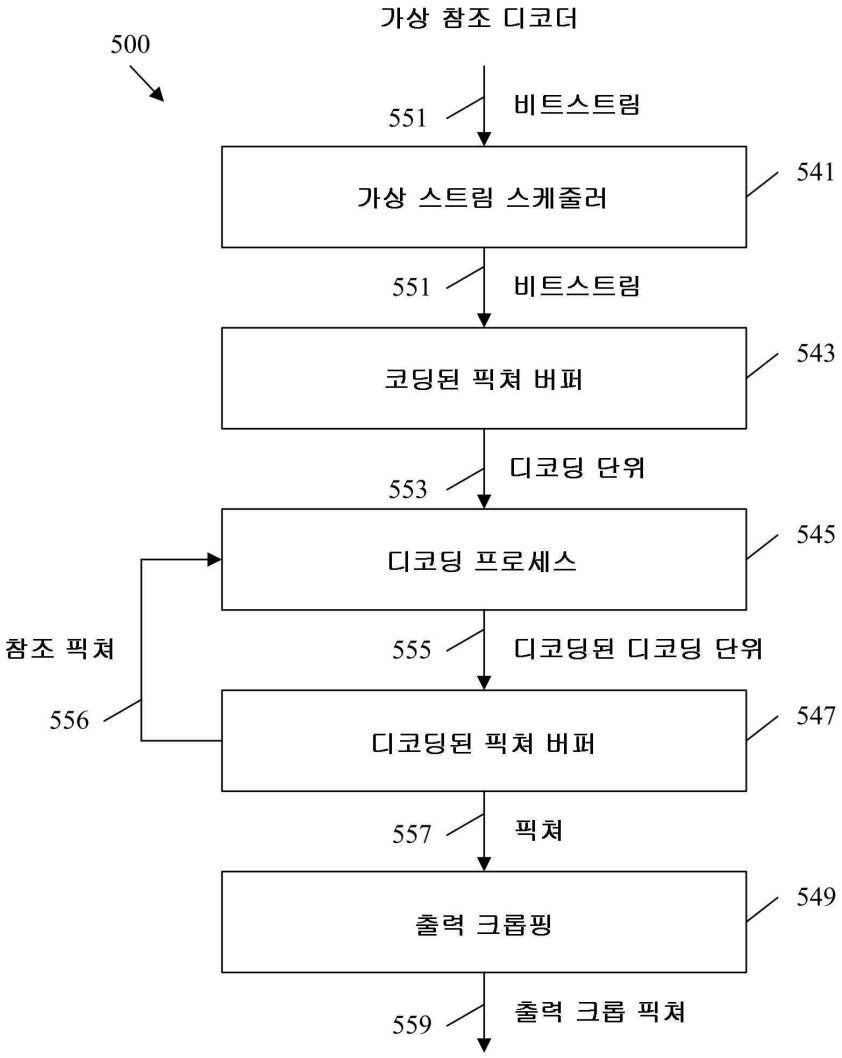
도면3


300
인코더
301
구획된
비디오 신호
313
변환
& 양자화
비트스트림
엔트로피
코딩
331
인트라 픽쳐
예측
317
역변환
& 양자화
329
모션 보상
321
325
루프내 필터
디코딩된
픽쳐 버퍼
323

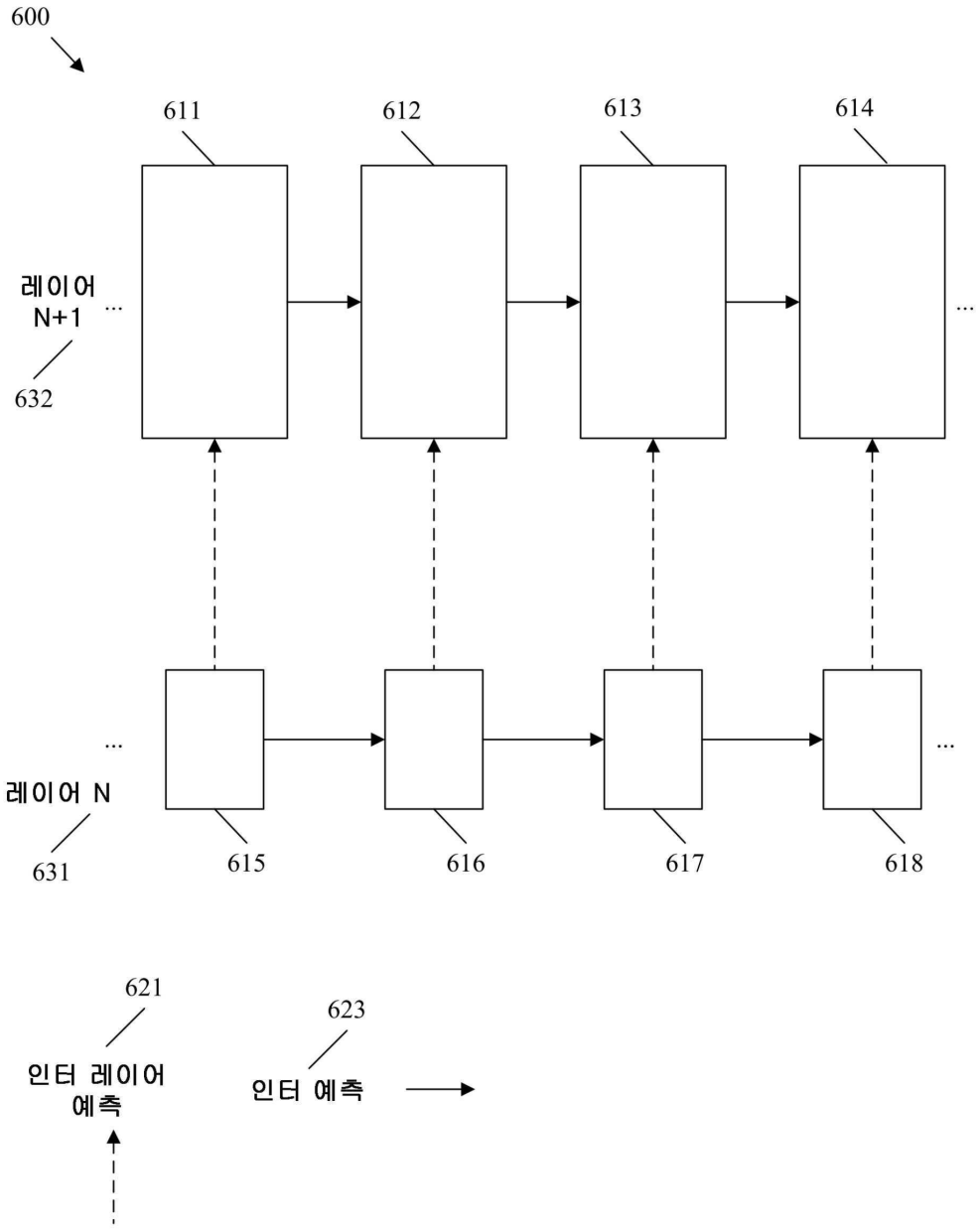
도면4


400
디코더
비트스트림
엔트로피 디코딩
433
429
역변환 & 양자화
421
모션 보상
417
인트라 픽쳐 예측
423
디코딩된 픽쳐 버퍼
425
루프내 필터
출력 비디오 신호

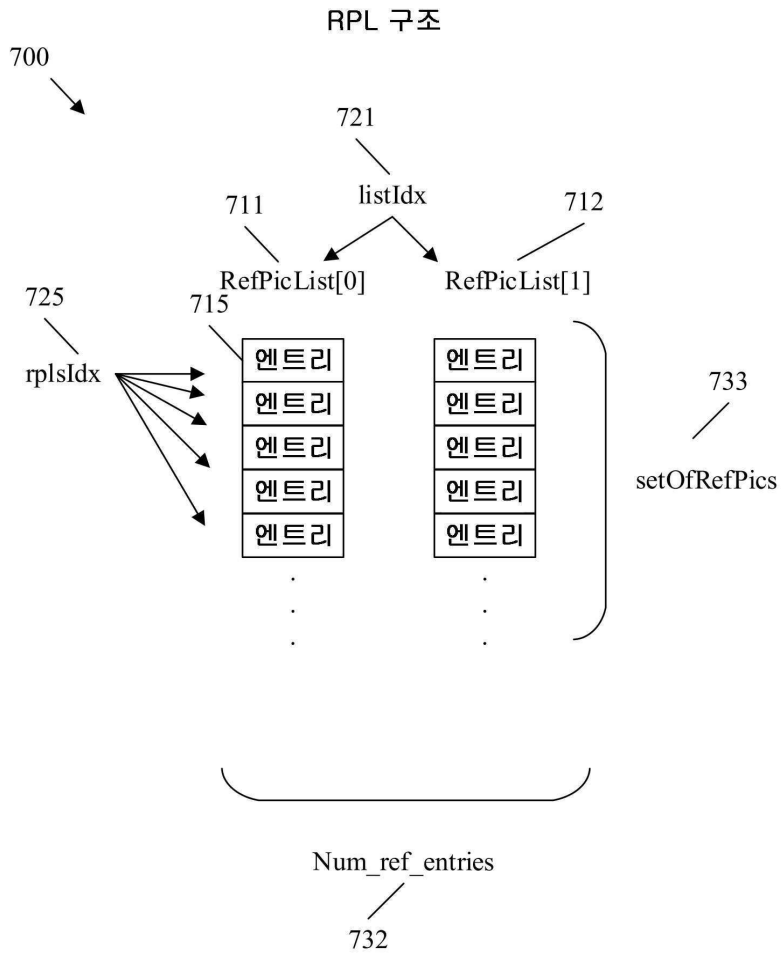
도면5


가상 참조 디코더
500
551
비트스트림
가상 스트림 스케줄러
541
551
비트스트림
코딩된 픽쳐 버퍼
543
553
디코딩 단위
디코딩 프로세스
545
참조 픽쳐
555
디코딩된 디코딩 단위
556
디코딩된 픽쳐 버퍼
547
557
픽쳐
출력 크롭핑
549
559
출력 크롭 픽쳐

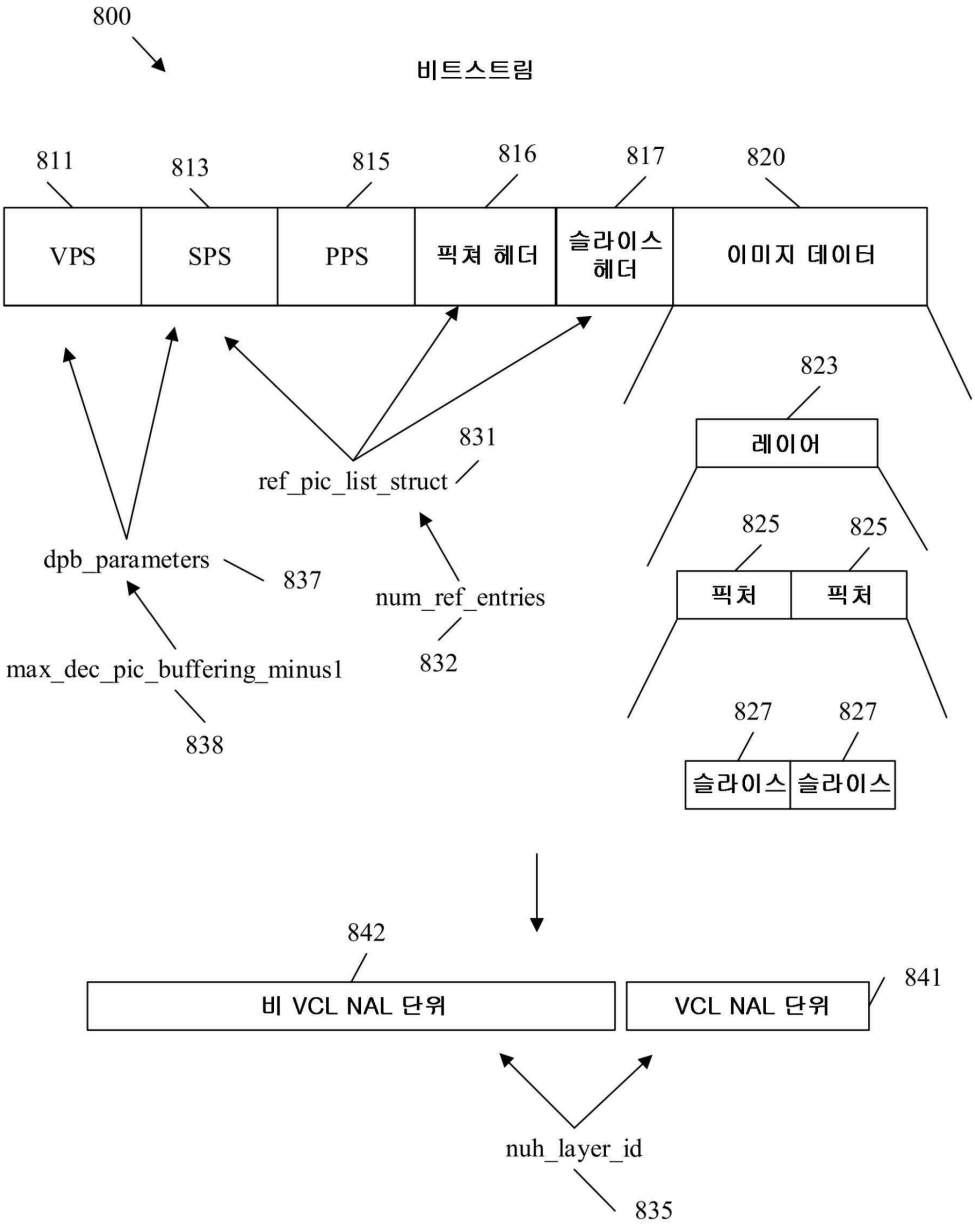
도면6


600
611
612
613
614
레이어 N+1
632
레이어 N
631
615
616
617
618
621
인터 레이어 예측
623
인터 예측

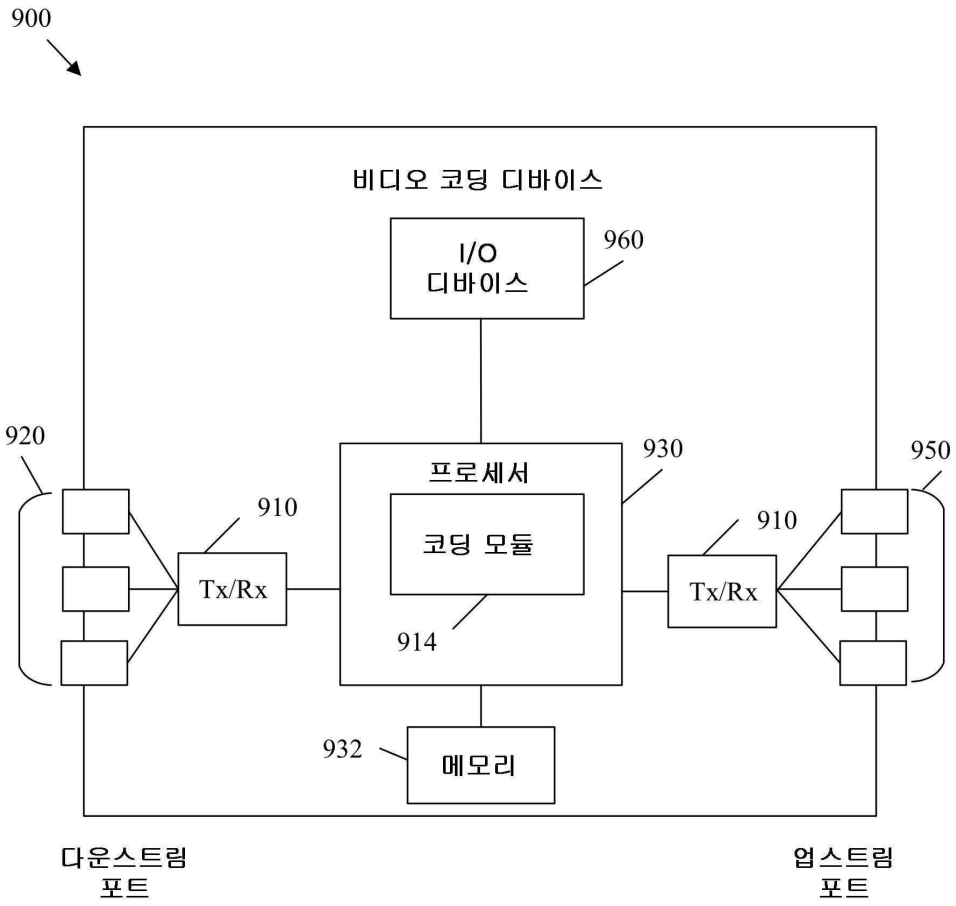
도면7


RPL 구조
700
721
listIdx
711
712
RefPicList[0]
RefPicList[1]
725
715
rplsIdx
엔트리
엔트리
엔트리
엔트리
엔트리
엔트리
엔트리
엔트리
엔트리
엔트리
733
setOfRefPics
Num_ref_entries
732

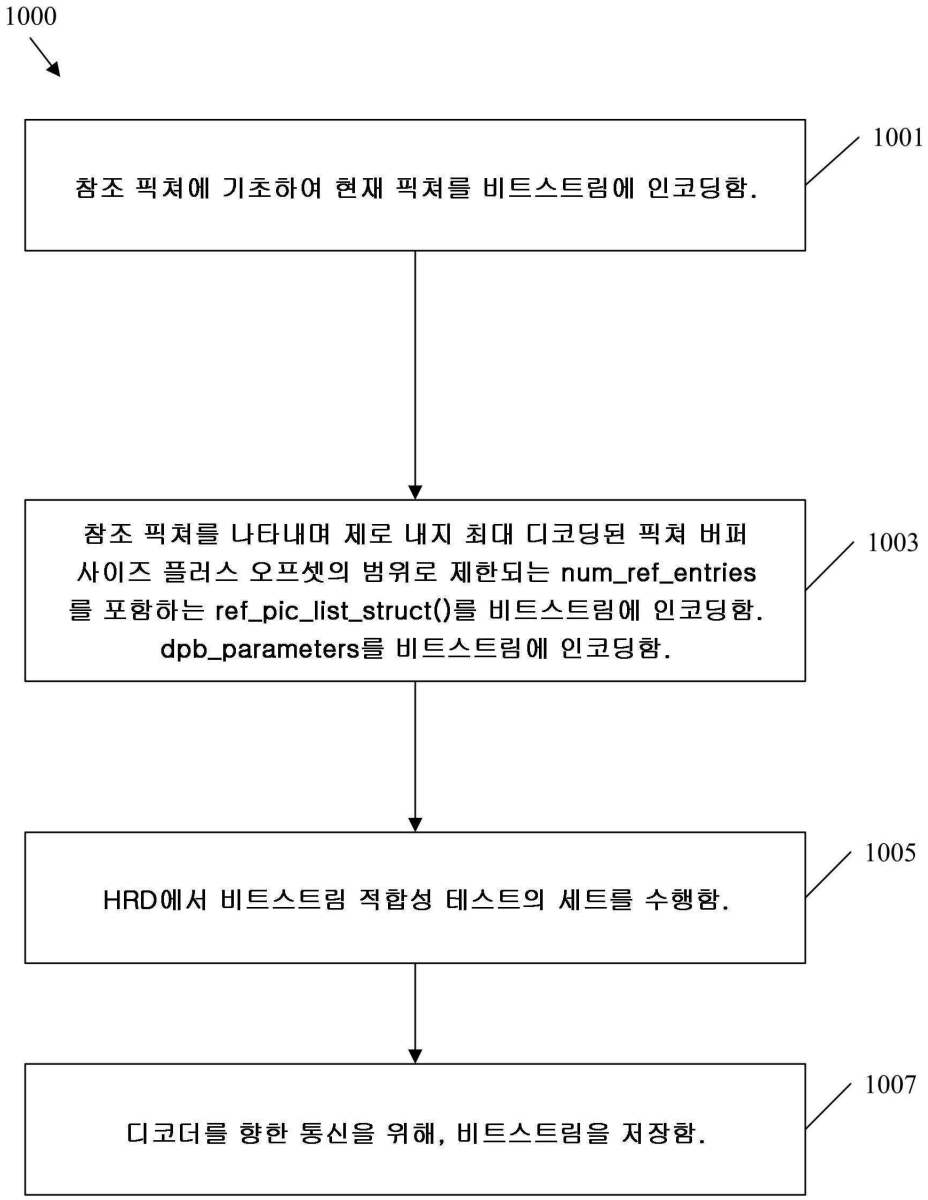
도면8


800
비트스트림
811
813
815
816
817
820
VPS
SPS
PPS
픽쳐 헤더
슬라이스 헤더
이미지 데이터
823
레이어
831
ref_pic_list_struct
825
825
픽처
픽처
dpb_parameters
837
num_ref_entries
832
max_dec_pic_buffering_minus1
838
827
827
슬라이스
슬라이스
842
841
비 VCL NAL 단위
VCL NAL 단위
nuh_layer_id
835

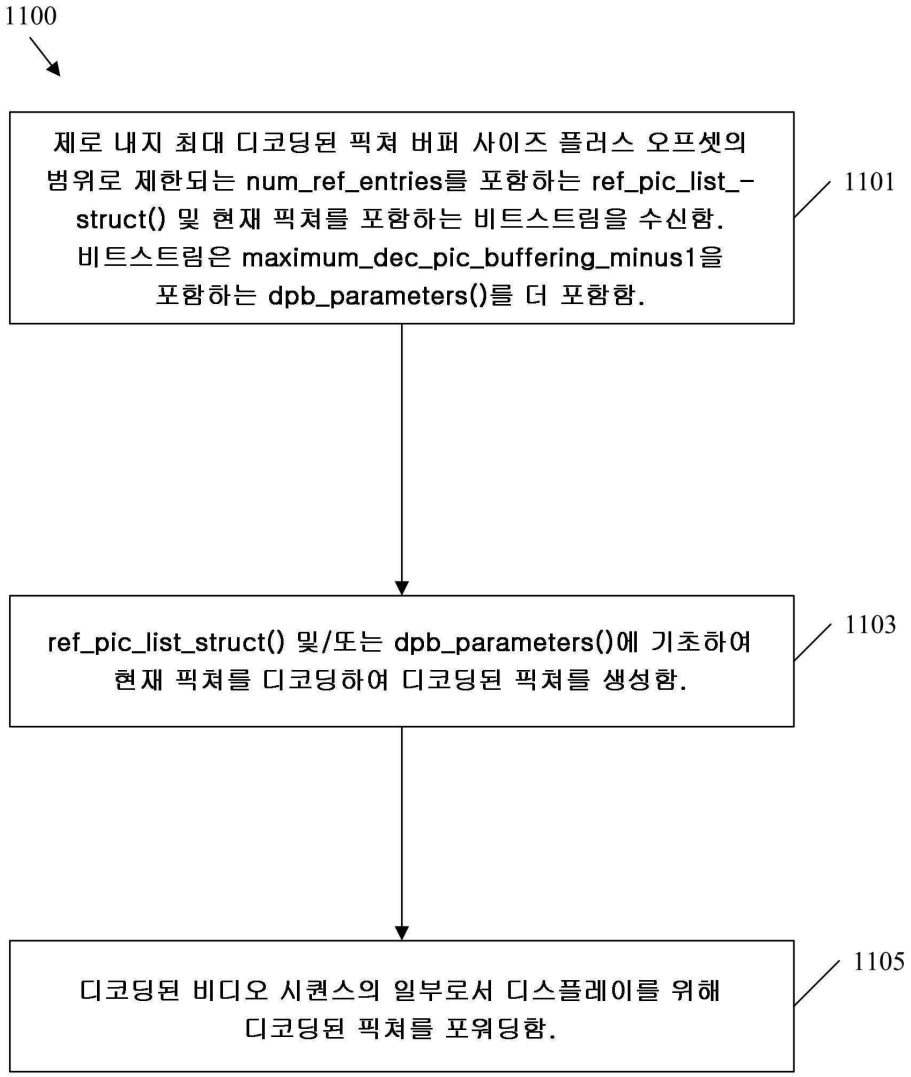
도면9


900
비디오 코딩 디바이스
I/O
디바이스
960
920
910
Tx/Rx
프로세서
930
코딩 모듈
914
910
Tx/Rx
950
932
메모리
다운스트림
포트
업스트림
포트

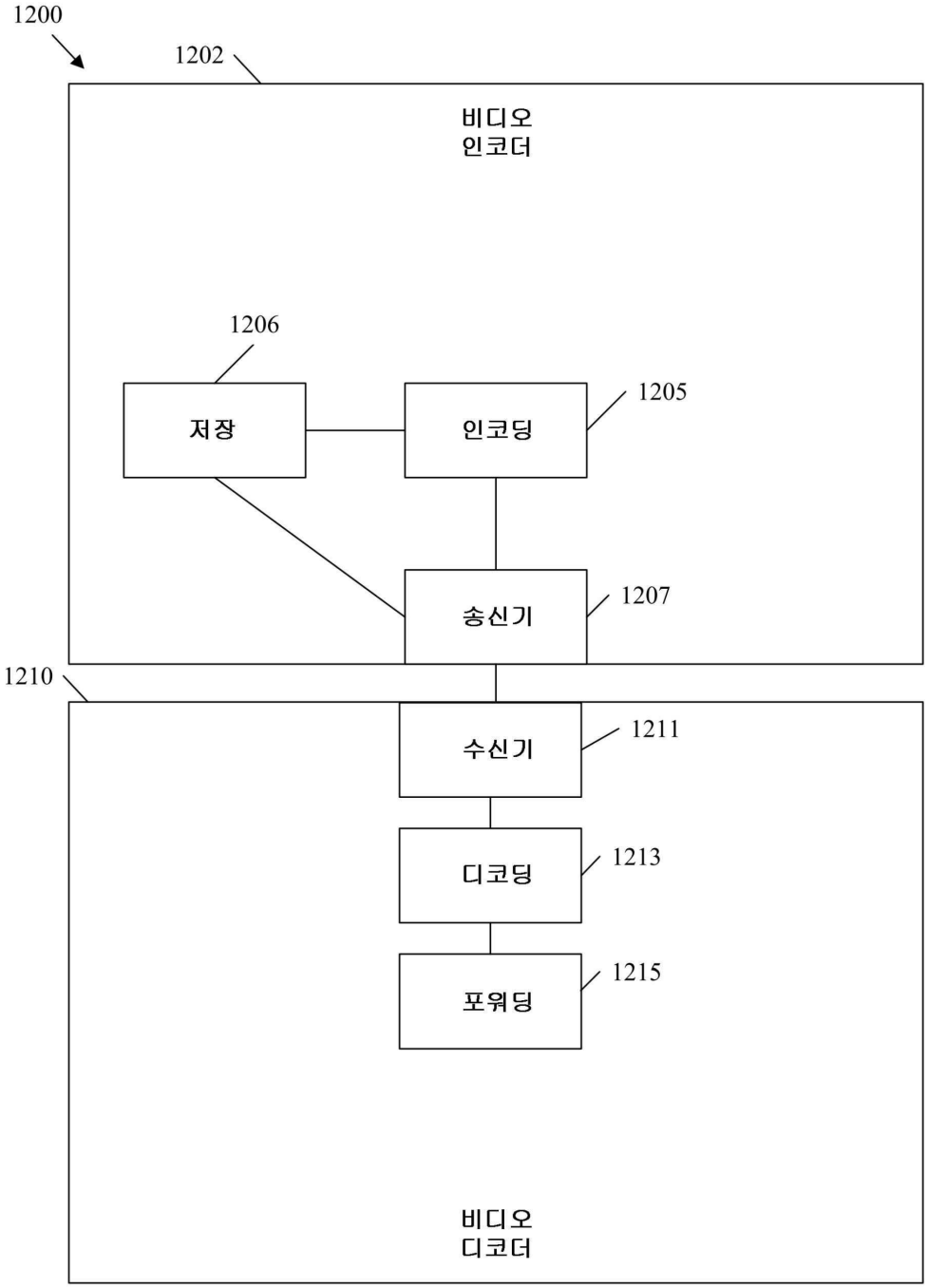
도면10


1000
참조 픽쳐에 기초하여 현재 픽쳐를 비트스트림에 인코딩함.
1001
참조 픽쳐를 나타내며 제로 내지 최대 디코딩된 픽쳐 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 num_ref_entries 를 포함하는 ref_pic_list_struct()를 비트스트림에 인코딩함. dpb_parameters를 비트스트림에 인코딩함.
1003
HRD에서 비트스트림 적합성 테스트의 세트를 수행함.
1005
디코더를 향한 통신을 위해, 비트스트림을 저장함.
1007

도면11


1100
제로 내지 최대 디코딩된 픽쳐 버퍼 사이즈 플러스 오프셋의 범위로 제한되는 num_ref_entries를 포함하는 ref_pic_list_struct() 및 현재 픽쳐를 포함하는 비트스트림을 수신함. 비트스트림은 maximum_dec_pic_buffering_minus1을 포함하는 dpb_parameters()를 더 포함함.
1101
ref_pic_list_struct() 및/또는 dpb_parameters()에 기초하여 현재 픽쳐를 디코딩하여 디코딩된 픽쳐를 생성함.
1103
디코딩된 비디오 시퀀스의 일부로서 디스플레이를 위해 디코딩된 픽쳐를 포워딩함.
1105

도면12


1200
1202
비디오
인코더
1206
저장
인코딩
1205
송신기
1207
1210
수신기
1211
디코딩
1213
포워딩
1215
비디오
디코더