(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2020년07월07일
(11) 등록번호 10-2131183
(24) 등록일자 2020년07월01일

(51) 국제특허분류(Int. Cl.)
G10L 19/02 (2006.01) *G10L 19/022* (2013.01)
(52) CPC특허분류
G10L 19/0212 (2013.01)
G10L 19/022 (2013.01)
(21) 출원번호 10-2017-7036140
(22) 출원일자(국제) 2016년06월10일
심사청구일자 2017년12월14일
(85) 번역문제출일자 2017년12월14일
(65) 공개번호 10-2018-0021704
(43) 공개일자 2018년03월05일
(86) 국제출원번호 PCT/EP2016/063371
(87) 국제공개번호 WO 2016/202701
국제공개일자 2016년12월22일
(30) 우선권주장
15172282.4 2015년06월16일
유럽특허청(EPO)(EP)
15189398.9 2015년10월12일
유럽특허청(EPO)(EP)
(56) 선행기술조사문헌
WO2013142650 A1
Juin-hwey Chen, A high-fidelity speech and audio codec with low delay and low complexity, IEEE ICASSP 2013. 2013.05.26.
EVS Codec Detailed Algorithmic Description (3GPP TS 26.445 version 12.0.0 Release 12). ETSI TS 126 445 V12.0.0. 2014.11.

(73) 특허권자
프라운호퍼 게젤샤프트 쭈르 푀르데룽 데어 안겐반텐 포르슝 에. 베.
독일 80686 뮌헨 한자슈트라쎄 27 체
(72) 발명자
슈넬, 마르쿠스
독일 90409 뉘른베르크 라벤볼프슈트라쎄 15
루츠키 만프레드
독일 90427 뉘른베르크 하인리히-폰-브렌타노-슈트라쎄 9
(뒷면에 계속)
(74) 대리인
윤의섭, 김수진

전체 청구항 수 : 총 21 항 심사관 : 이남숙

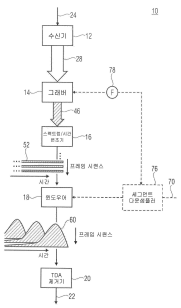
(54) 발명의 명칭 **다운스케일링된 디코딩**

(57) 요 약

다운스케일링된 오디오 디코딩에 사용된 합성 윈도우가 다운샘플링된 샘플링 속도 및 원래의 샘플링 속도가 편차를 나타내고, 프레임 길이의 1/4의 세그먼트에서 세그먼트 보간을 사용하여 다운샘플링되는 다운샘플링 인자만큼 다운샘플링함으로써 다운스케일링되지 않은 오디오 디코딩 절차에 수반된 기준 합성 윈도우의 다운스케일링된 버

(뒷면에 계속)

대 표 도 - 도2



전이라면, 오디오 디코딩 절차의 다운스케일링된 버전이 보다 효율적일 수 있고/있거나 개선된 컴플라이언스 유지보수가 달성될 수 있다.

(72) 발명자

포토포우로우, 엘레니
독일 90409 뉘른베르크 베르크하우세르슈트라쎄 33

슈미트, 콘스탄틴
독일 90459 뉘른베르크 란드그라벤슈트라쎄 128

벤도르프, 콘라드
독일 90408 뉘른베르크 헤롤드슈트라쎄 13

토마세크, 아드리안
독일 90513 찌른도르프 아이헨발트슈트라쎄 30

알베르트, 토비아스
독일 97348 뢰델제 도르프그라벤 11

자이들, 티몬
독일 91126 슈바바흐 슈타트파르크슈트라쎄 6

명 세 서

청구범위

청구항 1

오디오 신호가 제2 샘플링 속도로 변환 코딩된 데이터 스트림(24)으로부터 제1 샘플링 속도로 오디오 신호를 디코딩하도록 구성된 오디오 디코더(10)에 있어서,

상기 제1 샘플링 속도는 상기 제2 샘플링 속도의 1/F이고, 상기 오디오 디코더(10)는

상기 오디오 신호의 길이 N의 프레임 당, N개의 스펙트럼 계수(28)를 수신하도록 구성된 수신기(12);

각각의 프레임에 대해, 상기 N개의 스펙트럼 계수(28)에서 길이 N/F의 저주파 부분을 잡아내도록 구성된 그래버(14);

각각의 프레임(36)에 대해, 상기 저주파 부분이 각각의 프레임 및 (E + 1)개의 이전 프레임에 걸쳐 시간적으로 확장되는 길이 (E + 2)·N/F의 변조 함수를 갖는 역 변환을 받게 하여 길이 (E + 2)·N/F의 시간 부분을 획득하도록 구성된 스펙트럼-시간 변조기(16);

각각의 프레임(36)에 대해, 선단에서 길이 1/4·N/F의 제로 부분을 포함하고 합성 윈도우의 시간 간격 내에 피크를 갖는 길이 (E + 2)·N/F의 합성 윈도우를 사용하여 상기 시간 부분을 윈도윙하도록 구성된 윈도우어(18)로서, 상기 시간 간격은 상기 제로 부분이 이어지고 길이 (E + 7/4)·N/F를 가져, 상기 윈도우어가 길이 (E + 2)·N/F의 윈도윙된 시간 부분을 획득하는, 윈도우어(18); 및

프레임의 윈도윙된 시간 부분이 오버랩-가산 프로세스를 받게 하여 현재 프레임의 윈도윙된 시간 부분의 길이 (E + 1)/(E + 2)의 말단 부분이 이전 프레임의 윈도윙된 시간 부분의 길이 (E + 1)/(E + 2)의 선단에 오버랩하도록 구성된 시간 도메인 앨리어싱 제거기(20)를 포함하고,

상기 역 변환은 역 MDCT 또는 역 MDST이고,

상기 합성 윈도우는 길이 1/4·N 의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 다운샘플링된, 길이 (E + 2)·N 의 기준 합성 윈도우의 다운샘플링된 버전인 것을 특징으로 하는 오디오 디코더(10).

청구항 2

제1항에 있어서,

상기 합성 윈도우는 길이 1/4·N/F의 스플라인 함수의 연결인 것을 특징으로 하는 오디오 디코더(10).

청구항 3

제1항에 있어서,

상기 합성 윈도우는 길이 1/4·N/F의 입방체 스플라인 함수의 연결인 것을 특징으로 하는 오디오 디코더(10).

청구항 4

제1항에 있어서,

E = 2인 것을 특징으로 하는 오디오 디코더(10).

청구항 5

제1항에 있어서,

상기 역 변환은 역 MDCT인 것을 특징으로 하는 오디오 디코더(10).

청구항 6

제1항에 있어서,

상기 합성 윈도우의 80%를 초과하는 집단(mass)이 상기 제로 부분에 뒤이어 오고 길이 7/4·N/F를 갖는 시간 서브-간격 내에 포함되는 것을 특징으로 하는 오디오 디코더(10).

청구항 7

제1항에 있어서,

상기 오디오 디코더(10)는 보간을 수행하거나 스토리지로부터 상기 합성 윈도우를 도출하도록 구성되는 것을 특징으로 하는 오디오 디코더(10).

청구항 8

제1항에 있어서,

상기 오디오 디코더(10)는 F에 대해 상이한 값을 지원하도록 구성되는 것을 특징으로 하는 오디오 디코더(10).

청구항 9

제1항에 있어서,

상기 F는 1.5 및 10을 포함하여 1.5 내지 10 사이에 있는 것을 특징으로 하는 오디오 디코더(10).

청구항 10

제1항에 있어서,

상기 기준 합성 윈도우는 단봉형인 것을 특징으로 하는 오디오 디코더(10).

청구항 11

제1항에 있어서,

상기 오디오 디코더(10)는 상기 합성 윈도우의 계수의 다수가 상기 기준 합성 윈도우의 2개를 초과하는 계수에 의존하는 방식으로 보간을 수행하도록 구성되는 것을 특징으로 하는 오디오 디코더(10).

청구항 12

제1항에 있어서,

상기 오디오 디코더(10)는 세그먼트 경계로부터 2개를 초과하는 계수에 의해 분리된 합성 윈도우의 각각의 계수가 상기 기준 합성 윈도우의 2개를 초과하는 계수에 의존하는 방식으로 보간을 수행하도록 구성되는 것을 특징으로 하는 오디오 디코더(10).

청구항 13

제1항에 있어서,

상기 윈도우어(18) 및 상기 시간 도메인 앨리어싱 제거기가 협력하여 상기 윈도우어가 상기 합성 윈도우를 사용하여 상기 시간 부분에 가중치를 적용할 시에 상기 제로 부분을 스킵하고, 상기 시간 도메인 앨리어싱 제거기(20)는 상기 오버랩-가산 프로세스에서 윈도윙된 시간 부분의 대응하는 가중되지 않은 부분을 무시하여 단지 E+1 윈도윙된 시간 부분만이 합쳐져 대응하는 프레임의 대응하는 가중되지 않은 부분이 되고, E+2 윈도윙된 부분은 대응하는 프레임의 나머지 내에 합산되는 것을 특징으로 하는 오디오 디코더(10).

청구항 14

제1항에 따른 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하기 위한 오디오 디코더로서,

E=2여서, 합성 윈도우 함수가 길이 2·N/F의 나머지 절반이 선행하는 길이 2·N/F의 절반과 관련된 커널을 포함하고, 스펙트럼-시간 변조기(16), 윈도우어(18), 및 시간 도메인 앨리어싱 제거기(20)는 리프팅 구현에서 협력하도록 구현되고,

상기 스펙트럼-시간 변조기(16)는 각각의 프레임(36)에 대해, 저주파 부분이 각각의 프레임 및 (E + 1) 개의 이전 프레임에 걸쳐 시간적으로 확장되는 길이 (E + 2)·N/F의 변조 함수를 갖는 역 변환, 각각의 프레임 및 하나

의 이전 프레임과 일치하는 변환 커널을 겪어, 시간 부분 $x_{k,n}$(여기서 n = 0...2M-1)을 획득하도록 한정하며, M=N/F는 샘플 인덱스이고, k는 프레임 인덱스이고;

상기 윈도우어(18)는 각각의 프레임(36)에 대해, $z_{k,n}$ = $\omega_n \cdot x_{k,n}$(여기서 n = 0,...,2M-1)에 따라 시간 부분 $x_{k,n}$을 윈도윙하여 윈도윙된 시간 부분 $z_{k,n}$(여기서 n = 0...2M-1)을 획득하고;

상기 시간 도메인 앨리어싱 제거기(20)는 $m_{k,n} = z_{k,n} + z_{k-1,n+M}$(여기서 n = 0,...,M-1)에 따라 중간 시간 부분 $m_k(0),...,m_k(M-1)$을 생성하고;

상기 오디오 디코더는

n = M/2,...,M-1인 경우에, $u_{k,n} = m_{k,n} + l_{n-M/2} \cdot m_{k-1,M-1-n}$, 및

n=0,...,M/2-1인 경우에, $u_{k,n} = m_{k,n} + l_{M-1-n} \cdot m_{k-1,M-1-n}$

에 따라 프레임 $u_{k,n}$(여기서 n = 0...M-1)을 획득하도록 구성된 리프터(80)를 포함하고,

l_n(여기서 n = 0...M-1)은 리프팅 계수이고, l_n(여기서 n = 0...M-1) 및 ω_n(여기서 n = 0,...,2M-1)은 상기 합성 윈도우의 계수 w_n(여기서 n = 0...(E+2)M-1)에 의존하는 것을 특징으로 하는 오디오 디코더(10).

청구항 15

제1항에 따른 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하기 위한 장치에 있어서,

상기 장치는 동일한 길이의 4·(E + 2)개의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 길이 (E + 2)·N의 기준 합성 윈도우를 다운샘플링하도록 구성되는 것을 특징으로 하는 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하기 위한 장치.

청구항 16

제1항에 따른 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하는 방법에 있어서,

상기 방법은 동일한 길이의 4·(E + 2)개의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 길이 (E + 2)·N의 기준 합성 윈도우를 다운샘플링하는 단계를 포함하는 것을 특징으로 하는 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하는 방법.

청구항 17

오디오 신호가 제2 샘플링 속도로 변환 코딩된 데이터 스트림(24)으로부터 제1 샘플링 속도로 오디오 신호(22)를 디코딩하도록 구성된 오디오 디코더(10)에 있어서,

상기 제1 샘플링 속도는 상기 제2 샘플링 속도의 1/F이고, 상기 오디오 디코더(10)는

상기 오디오 신호의 길이 N의 프레임 당, N개의 스펙트럼 계수(28)를 수신하도록 구성된 수신기(12);

각각의 프레임에 대해, 상기 N개의 스펙트럼 계수(28)에서 길이 N/F의 저주파 부분을 잡아내도록 구성된 그래버(14);

각각의 프레임(36)에 대해, 상기 저주파 부분이 각각의 프레임 및 하나의 이전 프레임에 걸쳐 시간적으로 확장되는 길이 2·N/F 의 변조 함수를 갖는 역 변환을 받게 하여 길이 2·N/F의 시간 부분을 획득하도록 구성된 스펙트럼-시간 변조기(16);

각각의 프레임(36)에 대해, $z_{k,n} = \omega_n \cdot x_{k,n}$(여기서 n = 0,...,2M-1)에 따라 시간 부분 $x_{k,n}$을 윈도윙하여 윈도윙된 시간 부분 $z_{k,n}$(여기서 n = 0...2M-1이며 M=N/F)을 획득하도록 구성된 윈도우어(18);

$m_{k,n} = z_{k,n} + z_{k-1,n+M}$(n = 0,...,M-1)에 따라 중간 시간 부분 $m_k(0),...,m_k(M-1)$을 생성하도록 구성된 시간 도메인 앨리어싱 제거기(20); 및

리프터(80)로서, 상기 리프터(80)는

$n = M/2,...,M-1$인 경우, $u_{k,n} = m_{k,n} + l_{n-M/2} \cdot m_{k-1,M-1-n}$ 및,

$n=0,...,M/2-1$인 경우, $u_{k,n} = m_{k,n} + l_{M-1-n} \cdot m_{k-1,M-1-n}$

에 따라 상기 오디오 신호의 프레임 $u_{k,n}$(여기서 $n = 0...M-1$)을 획득하도록 구성되는, 리프터(80)를 포함하고,

l_n(여기서 $n = 0...M-1$)은 리프팅 계수이고,

상기 역 변환은 역 MDCT 또는 역 MDST이고,

l_n(여기서 $n = 0...M-1$) 및 ω_n(여기서 $n = 0,...,2M-1$)은 합성 윈도우의 계수 w_n(여기서 $n = 0...(E+2)M-1$)에 의존하고, 상기 합성 윈도우는 길이 1/4 ·N의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 다운샘플링된, 길이 4·N의 기준 합성 윈도우의 다운샘플링된 버전인 것을 특징으로 하는 오디오 디코더(10).

청구항 18

제17항에 따른 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하기 위한 장치에 있어서,

상기 장치는 동일한 길이의 4·(E + 2)개의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 길이 (E + 2)·N의 기준 합성 윈도우를 다운샘플링하도록 구성되는 것을 특징으로 하는 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하기 위한 장치.

청구항 19

제17항에 따른 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하는 방법에 있어서,

상기 방법은 동일한 길이의 4·(E + 2)개의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 길이 (E + 2)·N의 기준 합성 윈도우를 다운샘플링하는 단계를 포함하는 것을 특징으로 하는 오디오 디코더(10)의 합성 윈도우의 다운스케일링된 버전을 생성하는 방법.

청구항 20

오디오 신호가 제2 샘플링 속도로 변환 코딩된 데이터 스트림(24)으로부터 제1 샘플링 속도로 오디오 신호(22)를 디코딩하는 방법에 있어서,

상기 제1 샘플링 속도는 상기 제2 샘플링 속도의 1/F이고, 상기 방법은

상기 오디오 신호의 길이 N의 프레임 당, N개의 스펙트럼 계수(28)를 수신하는 단계;

각각의 프레임에 대해, 상기 N개의 스펙트럼 계수(28)에서 길이 N/F의 저주파 부분을 잡아내는 단계;

각각의 프레임(36)에 대해, 상기 저주파 부분이 각각의 프레임 및 (E + 1)개의 이전 프레임에 걸쳐 시간적으로 확장되는 길이 (E + 2)·N/F 의 변조 함수를 갖는 역 변환을 받게 하여 길이 (E + 2)·N/F의 시간 부분을 획득하도록 구성된 스펙트럼-시간 변조를 수행하는 단계;

각각의 프레임(36)에 대해, 선단에서 길이 1/4·N/F의 제로 부분을 포함하고 합성 윈도우의 시간 간격 내에 피크를 갖는 길이 (E + 2)·N/F의 합성 윈도우를 사용하여 상기 시간 부분을 윈도윙하는 단계로서, 상기 시간 간격은 상기 제로 부분이 이어지고 길이 (E + 7/4)·N/F를 가져, 윈도우어가 길이 (E + 2)·N/F의 윈도윙된 시간 부분을 획득하는, 윈도윙하는 단계; 및

프레임의 윈도윙된 시간 부분이 오버랩-가산 프로세스를 받게 하여 현재 프레임의 윈도윙된 시간 부분의 길이 (E + 1)/(E + 2)의 말단 부분이 이전 프레임의 윈도윙된 시간 부분의 길이 (E + 1)/(E + 2)의 선단에 오버랩하도록 시간 도메인 앨리어싱 제거를 수행하는 단계를 포함하고,

상기 역 변환은 역 MDCT 또는 역 MDST이고,

상기 합성 윈도우는 길이 1/4·N의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 다운샘플링된, 길이 (E + 2)·N 의 기준 합성 윈도우의 다운샘플링된 버전인 것을 특징으로 하는 오디오 신호(22)를 디코딩하는 방법.

청구항 21

컴퓨터 상에서 실행되는 경우, 제19항 또는 제20항에 따른 방법을 수행하기 위한 프로그램 코드를 갖는 컴퓨터 프로그램을 저장한 컴퓨터 판독가능 저장 매체.

발명의 설명

기 술 분 야

[0001] 본 출원은 다운스케일링된 디코딩 개념에 관한 것이다.

배 경 기 술

[0002] MPEG-4 향상된 저 지연 AAC(Enhanced Low Delay AAC, AAC-ELD)는 보통 최대 48kHz의 샘플 속도로 연산하며, 이는 15ms의 알고리즘 지연을 초래한다. 일부 애플리케이션, 예를 들어 오디오의 립-싱크 송신에 있어서는, 더욱 낮은 지연이 바람직하다. AAC-ELD는 더 높은 샘플 속도, 예를 들어 96kHz로 연산함으로써 이러한 옵션을 이미 제공하고, 따라서 더욱 저 지연, 예를 들어 7.5ms를 갖는 연산 모드를 제공한다. 그러나, 이 연산 모드는 높은 샘플 속도로 인해 불필요하게 높은 복잡성을 수반한다.

[0003] 이 문제에 대한 해결책은 필터 뱅크의 다운스케일링된 버전을 적용하고, 따라서 오디오 신호를 더 낮은 샘플 속도, 예를 들어 96 kHz 대신에 48 kHz로 렌더링하는 것이다. 다운스케일링 연산은 AAC-ELD에 대한 기초의 역할을 하는 MPEG-4 AAC-LD 코덱에서 상속되므로 이미 AAC-ELD의 일부이다.

[0004] 그러나, 남아 있는 의문은 특정 필터 뱅크의 다운스케일링된 버전을 찾는 방법이다. 즉, 유일한 불확실성은 AAC-ELD 디코더의 다운스케일링된 연산 모드의 명확한 적합성 테스트를 가능하게 하면서 윈도우 계수가 도출되는 방식이다.

[0005] 다음에서는, AAC-(E)LD 코덱의 다운스케일링된 연산 모드의 원리가 설명된다.

[0006] 다운스케일링된 연산 모드 또는 AAC-LD는 <ISO/IEC 14496-3:2009 in section 4.6.17.2.7 “Adaptation to systems using lower sampling rates”>에서 AAC-LD에 대해 다음과 같이 설명된다:

[0007] *"특정 애플리케이션에서는, 저 지연 디코더를 더 낮은 샘플링 속도(예를 들어, 16kHz)로 실행되는 한편 비트스트림 페이로드의 공칭 샘플링 속도는 훨씬 더 높은(예를 들어, 약 20ms의 알고리즘 코덱에 해당하는 48kHz) 오디오 시스템에 통합할 필요가 있다. 그러한 경우에, 디코딩 후에 추가적인 샘플링 속도 컨버젼 연산을 사용하기보다는 목표 지연 샘플링 속도로 직접 저 지연 코덱의 출력을 디코딩하는 것이 바람직하다.*

[0008] *이는 프레임 크기 및 샘플링 속도 양자 모두를 몇 가지 정수 인자(예를 들어, 2, 3)만큼 적절히 다운스케일링함으로써 근사화되어, 코덱의 동일한 시간/주파수 해상도를 초래할 수 있다. 예를 들어, 코덱 출력은 합성 필터 뱅크 전에 스펙트럼 계수의 가장 낮은 3분의 1(즉, 480/3 = 160) 만 유지하고 역 변환 크기를 3분의 1로 감소시킴으로써(즉, 윈도우 크기 960/3 = 320) 공칭 48kHz 대신 16kHz 샘플링 속도로 생성될 수 있다.*

[0009] *그 결과, 더 낮은 샘플링 속도에 대한 디코딩은 메모리 및 계산 요구 사항 양자 모두를 감소시키지만, 전체 대역폭 디코딩과 정확히 동일한 출력을 생성하지 않아, 대역폭 제한 및 샘플 속도 컨버젼이 뒤따를 수 있다.*

[0010] *전술한 바와 같이 더 낮은 샘플링 속도로 디코딩하는 것은 AAC 저 지연 비트스트림 페이로드의 공칭 샘플링 속도를 참조하는 수준 해석에 영향을 미치지 않는다는 것에 유의한다."*

[0011] AAC-LD는 표준 MDCT 프레임워크 및 2개의 윈도우 형상, 즉 사인 윈도우 및 낮은 오버랩 윈도우와 함께 작동한다는 것에 유의한다. 두 윈도우는 공식으로 완전히 설명되고, 따라서 임의의 변환 길이에 대한 윈도우 계수가 결정될 수 있다.

[0012] AAC-LD와 비교하여, AAC-ELD 코덱은 두 가지 주요 차이점을 보여준다:

[0013] • 저 지연 MDCT 윈도우(Low Delay MDCT, LD-MDCT)

[0014] • 저 지연 SBR 도구를 이용할 수 있는 가능성

[0015] 저 지연 MDCT 윈도우를 사용하는 IMDCT 알고리즘은 [1]의 4.6.20.2에 기술되어 있는데, 이는 예를 들어 사인 윈도우를 사용하는 표준 IMDCT 버전과 매우 유사하다. 저 지연 MDCT 윈도우의 계수(480 및 512 샘플 프레임 크기)가 [1]의 표 4.A.15 및 4.A.16에 나와 있다. 계수는 최적화 알고리즘의 결과이므로 계수는 공식으로 결정될

수 없다는 것에 유의한다. 도 9는 프레임 크기 512에 대한 윈도우 형상의 플롯을 도시한다.

[0016] 저 지연 SBR(low delay SBR, LD-SBR) 도구가 AAC-ELD 코더와 함께 사용되는 경우에, LD-SBR 모듈의 필터 뱅크도 다운스케일링된다. 이는 SBR 모듈이 동일한 주파수 해상도로 연산하는 것을 보장하므로, 더 이상의 적응이 필요하지 않다.

[0017] 따라서, 위의 설명은, 예를 들어, AAC-ELD에서 디코딩을 다운스케일링하는 것과 같은 다운스케일링 디코딩 연산에 대한 필요성을 나타낸다. 새로운 다운스케일링된 합성 윈도우 함수에 대한 계수를 찾는 것이 실현 가능할 것이지만, 이것은 번거로운 작업이며, 다운스케일링된 버전을 저장하기 위한 추가적인 스토리지를 필요로 하고, 다운스케일링되지 않은 디코딩과 다운스케일링된 디코딩 사이의 적합성 체크를 보다 복잡하게 만들거나, 다른 관점에서, 예를 들어 AAC-ELD에서 필요로 하는 다운스케일링의 방식에 부합하지 않는다. 다운스케일 비율, 즉 원래의 샘플링 속도와 다운스케일링된 샘플링 속도 사이의 비율에 따라, 단순히 다운샘플링하여, 즉 원래의 합성 윈도우 함수의 매 두 번째, 세 번째, ... 윈도우 계수를 선택하여 다운스케일링된 합성 윈도우 함수를 도출할 수 있지만, 이 절차는 다운스케일링되지 않은 디코딩 및 다운스케일링된 디코딩의 충분한 적합성을 가져오지 않는다. 합성 윈도우 함수에 적용된 보다 정교한 데시메이션 절차를 사용하면, 원래의 합성 윈도우 함수 형상으로부터의 받아들일 수 없는 편차를 야기한다. 따라서, 본 기술분야에서는 개선된 다운스케일링된 디코딩 개념에 대한 필요성이 있다.

발명의 내용

해결하려는 과제

[0018] 따라서, 본 발명의 목적은 이러한 개선된 다운스케일링된 디코딩을 할 수 있게 하는 오디오 디코딩 방식을 제공하는 것이다.

과제의 해결 수단

[0019] 이 목적은 독립항의 주제에 의해 달성된다.

[0020] 본 발명은 다운스케일링된 오디오 디코딩에 사용된 합성 윈도우가 다운샘플링된 샘플링 속도 및 원래의 샘플링 속도가 편차를 나타내고, 프레임 길이의 1/4의 세그먼트에서 세그먼트 보간을 사용하여 다운샘플링되는 다운샘플링 인자만큼 다운샘플링함으로써 다운스케일링되지 않은 오디오 디코딩 절차에 수반된 기준 합성 윈도우의 다운스케일링된 버전이라면, 오디오 디코딩 절차의 다운스케일링된 버전이 보다 효율적일 수 있고/있거나 개선된 컴플라이언스 유지보수가 달성될 수 있다는 발견에 기초한다.

도면의 간단한 설명

[0021] 본 출원의 유리한 양태는 종속항의 주제이다. 본 출원의 바람직한 실시 예는 도면과 관련하여 아래에서 설명되며, 그 중에서:

도 1은 완전한 재구성을 보전하기 위해 디코딩을 다운스케일링하는 경우에 따르기 위해 필요한 완벽한 재구성 요구 사항을 도시하는 개략도를 도시한다;

도 2는 일 실시예에 따른 다운스케일링된 디코딩을 위한 오디오 디코더의 블록도를 도시한다.

도 3은 도 2의 오디오 디코더의 연산 모드를 설명하기 위해, 오디오 신호가 원래의 샘플링 속도로 데이터 스트림으로 코딩되는 상반부에서의 방법, 및 상반부로부터 파선된 수평 라인에 의해 분리된 하반부에서의, 감소된 또는 다운스케일링된 샘플링 속도로 데이터 스트림으로부터 오디오 신호를 재구성하기 위한 다운스케일링된 디코딩 연산을 도시하는 개략도를 도시한다;

도 4는 도 2의 윈도우어와 시간 도메인 앨리어싱 제거기의 협력을 도시하는 개략도를 도시한다;

도 5는 스펙트럼-시간 변조된 시간 부분의 0이 가중된 부분의 특별한 처리를 사용하여 도 4에 따른 재구성을 달성하기 위한 가능한 구현예를 도시한다;

도 6은 다운샘플링된 합성 윈도우를 획득하기 위한 다운샘플링을 도시하는 개략도를 도시한다;

도 7은 저 지연 SBR 도구를 포함하는 AAC-ELD의 다운스케일링된 연산을 도시하는 블록도를 도시한다;

도 8은 리프팅 구현에 따라 변조기, 윈도우어, 및 제거기가 구현되는 실시예에 따른 다운스케일링된 디코딩을

위한 오디오 디코더의 블록도를 도시한다; 그리고

도 9는 다운샘플링될 기준 합성 윈도우의 예로서 512 샘플 프레임 크기에 대한 AAC-ELD에 따른 저 지연 윈도우의 윈도우 계수의 그래프를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0022] 다음의 설명은 AAC-ELD 코덱과 관련하여 다운스케일링된 디코딩에 대한 실시예의 설명으로 시작한다. 즉, 다음의 설명은 AAC-ELD에 대한 다운스케일링된 모드를 형성할 수 있는 실시예에서 시작한다. 이 설명은 동시에 본 출원의 실시예의 기초가 되는 동기에 대한 일종의 설명을 형성한다. 이후, 이 설명은 일반화되어, 본 출원의 실시예에 따른 오디오 디코더 및 오디오 디코딩 방법을 설명한다.

[0023] 본 출원의 명세서의 서론 부분에서 설명된 바와 같이, AAC-ELD는 저 지연 MDCT 윈도우를 사용한다. 다운스케일링된 버전, 즉 다운스케일링된 저 지연 윈도우를 생성하기 위해, AAC-ELD에 대한 다운스케일링된 모드를 형성하는 것에 대한 후술된 제안은 매우 높은 정밀도로 LD-MDCT 윈도우의 완벽한 재구성 특성(PR)을 유지하는 세그먼트 스플라인(spline) 보간 알고리즘을 사용한다. 따라서, 알고리즘은 ISO / IEC 14496-3:2009에 기술된 바와 같이, 또한 [2]에서 설명한대로 리프팅 형식에서, 호환되는 방식으로 직접 형태의 윈도 계수 생성할 수 있게 한다. 이것은 두 가지 구현이 16 비트 규격 출력을 생성함을 의미한다.

[0024] 저 지연 MDCT 윈도우의 보간은 다음과 같이 수행된다.

[0025] 일반적으로, 스플라인 보간은 주파수 응답 및 대부분 완벽한 재구성 특성(약 170dB SNR)을 유지하기 위해 다운스케일링된 윈도우 계수를 생성하는 데 사용된다. 보간은 완벽한 재구성 특성을 유지하기 위해 특정 세그먼트에서 제한적일 필요가 있다. 변환의 DCT 커널을 커버하는 윈도우 계수 c에 대해 (도 1, c(1024)..c(2048) 참조), 다음의 제약이 필요하다.

[0026] $i = 0 \ldots N/2 - 1$ 인 경우, $1 = \left|(\mathrm{sgn} \cdot c(i) \cdot c(2N-1-i) + c(N+i) \cdot c(N-1-i))\right|$ (1)

[0027] 여기서 N 은 프레임 크기를 표시한다. 일부 구현예는 여기에서 sgn으로 표시된 복잡성을 최적화하기 위해 상이한 기호를 사용할 수 있다. (1)의 요구 사항은 도 1에 의해 설명될 수 있다. 간단히 F=2인 경우에도, 즉 샘플 속도를 절반으로 낮춘 경우에도, 다운스케일링된 합성 윈도우를 획득하기 위해 기준 합성 윈도우의 모든 제2 윈도우 계수를 생략하는 것은 요구 사항을 충족시키지 못한다는 것을 상기해야 한다.

[0028] 계수 c(0)...c(2N-1)은 다이아몬드 형상을 따라 나열된다. 필터 뱅크의 지연 감소를 담당하는 윈도우 계수의 N/4개의 0은 굵은 화살표를 사용하여 표기된다. 도 1은 MDCT에 수반된 폴딩에 의해 야기되는 계수의 종속성, 및 원하지 않는 종속성을 피하기 위해 보간이 제약되어야 하는 지점을 도시한다.

[0029] • 모든 N/2 계수에 대해, 보간은 (1)을 유지하기 위해 중지되어야 한다.

[0030] • 또한, 보간 알고리즘은 삽입된 0으로 인해 모든 N/4 계수를 중지해야 한다. 이는 0이 유지되고 PR을 유지하는 보간 에러가 확산되지 않도록 한다.

[0031] 제2 제약은 0을 포함하는 세그먼트뿐만 아니라 다른 세그먼트에도 필요하다. DCT 커널의 일부 계수가 최적화 알고리즘에 의해 결정되지는 않았지만 PR을 가능하게 하기 위해 공식 (1)에 의해 결정된 것을 알면, 윈도우 형상의 몇 가지 불연속성이 예를 들어 도 1 의 c(1536+128)에 대해 설명될 수 있다. PR 에러를 최소화하기 위해, N/4 그리드에 나타나는 지점에서 보간은 중지되어야 한다.

[0032] 그 이유 때문에, 다운스케일링된 윈도우 계수를 생성하기 위해 세그먼트 스플라인 보간에 대해 N/4의 세그먼트 크기가 선택된다. 소스 윈도우 계수는 항상 N = 512, 또는 N = 240 또는 N = 120의 프레임 크기를 초래하는 다운스케일링 연산에 사용되는 계수로 제공된다. 기본 알고리즘은 MATLAB 코드로 다음에서 매우 간단하게 설명된다:

```
FAC = Downscaling factor    % e.g. 0.5
sb  = 128;                  % segment size of source window
w_down = [];                % downscaled window
nSegments = length(W)/(sb);% number of segments; W=LD window
                             coefficients for N=512

xn=((0:(FAC*sb-1))+0.5)/FAC-0.5;  % spline init
for i=1:nSegments,
  w_down=[w_down,spline([0:(sb-1)],W((i-1)*sb+(1:(sb))),xn)];
end;
```

[0033]

[0034] 스플라인 함수가 완전히 결정적이지 않을 수 있기 때문에, AAC-ELD에서 개선된 다운스케일링된 모드를 생성하기 위해 ISO/IEC 14496-3:2009에 포함될 수 있는 다음 섹션에서 전체 알고리즘이 정확하게 명시한다.

[0035] 다시 말해, 다음 섹션은 위에서 설명한 아이디어가 ER AAC ELD에 어떻게 적용될 수 있는지에 관한, 즉 낮은 복잡도의 디코더가 제1 데이터 레이트보다 낮은 제2 데이터 레이트로 제1 데이터 속도로 코딩된 ER AAC ELD 비트스트림을 어떻게 디코딩할 수 있는지에 관한 제안을 제공한다. 그러나, 다음에서 사용되는 N의 정의는 표준을 준수한다는 점이 강조된다. 본 명세서에서, N은 DCT 커널의 길이에 해당하지만, 본 명세서, 청구 범위 및 후술된 일반화된 실시예에서, N은 프레임 길이, 즉 DCT 커널의 상호 오버랩 길이, 즉 DCT 커널 길이의 절반에 해당한다. 따라서, 예를 들면, N은 512인 것으로 위에서 나타내지만, 예를 들어 다음에서는 1024로 나타낸다.

[0036] 다음 문단은 개정을 통해 14496-3:2009에 포함시키기 위해 제안되었다.

[0037] **A.0 낮은 샘플링 속도를 사용하는 시스템에 대한 적응**

[0038] 특정 애플리케이션의 경우, ER AAC LD는 추가적인 리샘플링 단계를 피하기 위해 재생 샘플 속도를 변경할 수 있다 (4.6.17.2.7 참조). ER AAC ELD는 저 지연 MDCT 윈도우 및 LD-SBR 도구를 사용하여 유사한 다운스케일링 단계를 적용할 수 있다. AAC-ELD가 LD-SBR 도구와 함께 연산하는 경우, 다운스케일링 인자는 2의 배수로 제한된다. LD-SBR이 없으면, 다운스케일링된 프레임 크기는 정수여야 한다.

[0039] **A.1 저 지연 MDCT 윈도우의 다운스케일링**

[0040] N=1024인 경우에 LD-MDCT 윈도우 w_{LD}는 세그먼트 스플라인 보간을 사용하여 인자 F로 다운스케일링된다. 윈도우 계수의 선행하는 0의 수, 즉 N/8이 세그먼트 크기를 결정한다. 다운스케일링된 윈도우 계수 w_{LD_d}는 4.6.20.2에서 설명된 바와 같이 역 MDCT에 사용되지만, 다운스케일링된 윈도우 길이 N_d = N/F를 갖는다. 알고리즘은 또한 LD-MDCT의 다운스케일링된 리프팅 계수를 생성할 수 있음에 유의한다.

```
fs_window_size  = 2048; /* Number of fullscale window coefficients. According to ISO/IEC 14496-3:2009,
                  use 2048. For lifting implemenations, please adjust this variable accordingly */
ds_window_size  = N * fs_window_size / (1024 * F); /* downscaled window coefficients; N determines the
                                  transformation length according to 4.6.20.2 */
fs_segment_size = 128;
num_segments    = fs_window_size / fs_segment_size;
ds_segment_size = ds_window_size / num_segments;
tmp[128], y[128]; /* temporary buffers */
```

[0041]

```
/* loop over segments */
for (b = 0; b < num_segments; b++) {
   /* copy current segment to tmp */
   copy(&W_LD[b * fs_segment_size], tmp, fs_segment_size);

   /* apply cubic spline interpolation for downscaling */
```

[0042]

```
/* calculate interpolating phase */
phase = (fs_window_size - ds_window_size) / (2 * ds_window_size);

/* calculate the coefficients c of the cubic spline given tmp */
/* array of precalculated constants */
m = {0.166666672, 0.25, 0.266666681, 0.267857134,
   0.267942578, 0.267948717, 0.267949164};
n = fs_segment_size;  /* for simplicity */
```

[0043]

```
/* calculate vector r needed to calculate the coefficients c */
for (i = n - 3; i >= 0; i--)
   r[i] = 3 * ((tmp[i + 2] - tmp[i + 1]) - (tmp[i + 1] - tmp[i]));
for (i = 1; i < 7; i++)
   r[i] -= m[i - 1] * r[i - 1];
for(i = 7; i < n - 4; i++)
   r[i] -= 0.267949194 * r[i - 1];

/* calculate coefficients c */
c[n - 2] = r[n - 3] / 6;
c[n - 3] = (r[n - 4] - c[n - 2]) * 0.25;
for (i = n - 4; i > 7; i--)
   c[i] = (r[i - 1] - c[i + 1]) * 0.267949194;
for (i = 7; i > 1; i--)
   c[i]=(r[i-1]-c[i+1])*m[i-1];
c[1]  = r[0] * m[0];
c[0]  = 2 * c[1] - c[2];
c[n-1] = 2 * c[n - 2] - c[n - 3];
```

[0044]

```
/* keep original samples in temp buffer y because samples of
   tmp will be replaced with interpolated samples */
copy(tmp, y, fs_segment_size);

/* generate downscaled points and do interpolation */
for (k = 0; k < ds_segment_size; k++) {
   step = phase + k * fs_segment_size / ds_segment_size;
   idx  = floor(step);
   diff = step - idx;
   di = (c[idx + 1] - c[idx]) / 3;
   bi = (y[idx + 1] - y[idx]) - (c[idx + 1] + 2 * c[idx]) / 3;
   /* calculate downscaled values and store in tmp */
   tmp[k] = y[idx] + diff * (bi + diff * (c[idx] + diff * di));
}
```

[0045]

```
  /* assemble downscaled window */
  copy(tmp, &W_LD_d[b * ds_segment_size], ds_segment_size);
}
```

[0046]

[0047] **A.2 저 지연 SBR 도구의 다운스케일링**

[0048] 저 지연 SBR 도구가 ELD와 함께 사용되는 경우, 이 도구는 적어도 2의 배수의 다운스케일링 인자에 대해 샘플 속도를 낮추기 위해 다운스케일링될 수 있다. 다운스케일 인자 F는 CLDFB 분석 및 합성 필터 뱅크에 사용되는 대역 수를 제어한다. 다음 두 단락은 다운스케일링된 CLDFB 분석 및 합성 필터 뱅크에 대해 설명한다 (4.6.19.4 참조).

[0049] **4.6.20.5.2.1다운스케일링된 분석 CLDFB 필터 뱅크**

[0050] • 다운스케일링된 CLDFB 대역의 수를 B = 32/F로 정의한다.

[0051] • 배열 x의 샘플을 B 위치만큼 이동시킨다. 가장 오래된 B 샘플은 버려지고, B개의 새로운 샘플은 위치 0 내지 B-1에 저장된다.

[0052] • 배열 x의 샘플에 윈도우 계수 ci를 곱하여 배열 z를 얻는다. 윈도우 계수 ci는 계수 c의 선형 보간에 의해, 즉, 다음의 방정식에 의해 획득된다.

[0053] $$ci(i) = \frac{1}{2}[c(2F \cdot i + 1 + p) + c(2F \cdot i + p)], \quad 0 \le i < (10B), \qquad p = int\left(\frac{64}{2B} - 0.5\right)$$

[0054] c의 윈도우 계수는 표 4.A.90에서 찾을 수 있다.

[0055] • 샘플을 합하여 2B 요소 배열 u를 만든다:

[0056] $$u(n) = z(n) + z(n + 2B) + z(n + 4B) + z(n + 6B) + z(n + 8B), \quad 0 \le n < (2B)$$

[0057] • 행렬 연산 Mu에 의해 B개 새로운 서브 대역 샘플을 계산하며, 여기서

[0058] $$M(k,n) = 2 \cdot \exp\left(\frac{j \cdot \pi \cdot (k+0.5) \cdot (2n-(3B-1))}{2B}\right), \quad \begin{cases} 0 \le k < B \\ 0 \le n < 2B \end{cases}$$

[0059] 이다.

[0060] 방정식에서, exp()는 복소 지수 함수를 나타내고, j는 허수 단위이다.

[0061] **4.6.20.5.2.2다운스케일링된 합성 CLDFB 필터 뱅크**

[0062] • 다운스케일링된 CLDFB 대역의 수를 B = 64/F로 정의한다.

[0063] • 배열 v의 샘플을 2B 위치만큼 이동시킨다. 가장 오래된 2B 샘플은 버려진다.

[0064] • B개의 새로운 복소수 값 서브 대역 샘플에 행렬 N이 곱해지며, 여기서

[0065] $$N(k,n) = \frac{1}{64} \cdot \exp\left(\frac{j \cdot \pi \cdot (k+0.5) \cdot (2 \cdot n-(B-1))}{2B}\right), \quad \begin{cases} 0 \le k < B \\ 0 \le n < 2B \end{cases}$$

[0066] 이다.

[0067] 방정식에서, exp()는 복소 지수 함수를 나타내고, j는 허수 단위이다. 이 연산으로부터의 출력의 실수부는 배열 v의 위치 0 내지 2B-1에 저장된다.

[0068] • v에서 샘플을 추출하여 10B 요소 배열 g를 만든다.

[0069] $$\begin{matrix} g(2B \cdot n + k) = v(4B \cdot n + k) \\ g(2B \cdot n + B + k) = v(4B \cdot n + 3B + k) \end{matrix}, \quad \begin{cases} 0 \le n \le 4 \\ 0 \le k < B \end{cases}$$

[0070] • 배열 w를 생성하기 위해 윈도우 계수 ci에 배열 g의 샘플을 곱한다. 윈도우 계수 ci는 계수 c의 선형 보간에 의해, 즉, 다음의 방정식에 의해 획득된다.

[0071] $$ci(i) = \frac{1}{2}[c(2F \cdot i + 1 + p) + c(2F \cdot i + p)], \quad 0 \le i < (10B), p = int\left(\frac{64}{2B} - 0.5\right)$$

[0072] c의 윈도우 계수는 표 4.A.90에서 찾을 수 있다.

[0073] • 다음의 방정식에 따라 배열 w의 샘플 합계로 B개의 새로운 출력 샘플을 계산한다.

[0074] $$\text{output}(n) = \sum_{i=0}^{i \le 9} w(Bi + n), \quad 0 \le n < B$$

[0075] 4.6.19.4.3에 따라 F=2로 설정하면 다운샘플링된 합성 필터 뱅크가 제공됨에 유의한다. 따라서, 추가적인 다운스케일 인자 F를 갖는 다운샘플링된 LD-SBR 비트스트림을 처리하기 위해서는, F에 2를 곱할 필요가 있다.

[0076] **4.6.20.5.2.3 다운스케일링된 실수 값 CLDFB 필터 뱅크**

[0077] CLDFB의 다운스케일링은 저 전력 SBR 모드의 실수 값 버전에도 적용될 수 있다. 예를 들어, 4.6.19.5도 고려한다.

[0078] 다운스케일링된 실수 값 분석 및 합성 필터 뱅크의 경우, 4.6.20.5.2.1 및 4.6.20.2.2의 설명을 따르고, cos() 변조기로 M의 exp() 변조기를 교환한다.

[0079] **A.3 저 지연 MDCT 분석**

[0080] 이 하위 절은 AAC ELD 인코더에서 이용되는 저 지연 MDCT 필터 뱅크를 설명한다. 핵심 MDCT 알고리즘은 대체로 변경되지 않지만, 긴 윈도우를 사용하여, n은 이제 (0에서 N-1이 아니라) -N 내지 N-1에서 실행된다.

[0081] 스펙트럼 계수 $X_{i,k}$는 다음과 같이 정의된다:

[0082] $0 \le k < N/2$에 있어서, $X_{i,k} = -2 \cdot \sum_{n=-N}^{N-1} z_{i,n} \cos\left(\frac{2\pi}{N}(n+n_0)\left(k+\frac{1}{2}\right)\right)$

[0083] 여기서:

[0084] z_{in} = 윈도윙된 입력 시퀀스

[0085] N = 샘플 인덱스

[0086] K = 스펙트럼 계수 인덱스

[0087] I = 블록 인덱스

[0088] N = 윈도우 길이

[0089] n_0 = (-N / 2 + 1) / 2

[0090] 윈도우 길이 N(사인 윈도우에 기초함)은 1024 또는 960이다.

[0091] 저 지연 윈도우의 윈도우 길이는 2*N이다. 윈도윙은 다음과 같은 방식으로 과거에 확장된다:

[0092] $z_{i,n} = w_{LD}(N-1-n) \cdot x'_{i,n}$

[0093] n=-N,...,N-1인 경우에, 합성 윈도우 w는 순서를 반전시킴으로써 분석 윈도우로서 사용된다.

[0094] **A.4 저 지연 MDCT 합성**

[0095] 합성 필터 뱅크는 저 지연 필터 뱅크를 채택하기 위해 사인 윈도우를 사용하는 표준 IMDCT 알고리즘과 비교하여 수정된다. 핵심 IMDCT 알고리즘은 대부분 변경되지 않지만, 더 긴 윈도우를 사용하여, n은 이제 (최대 N-1이 아니라) 2N-1까지 실행된다.

[0096] $0 \le n < 2N$에 있어서, $x_{i,n} = -\frac{2}{N}\sum_{k=0}^{\frac{N}{2}-1} spec[i][k] \cos\left(\frac{2\pi}{N}(n+n_0)\left(k+\frac{1}{2}\right)\right)$

[0097] 여기서:

[0098] n = 샘플 인덱스

[0099] i = 윈도우 인덱스

[0100] k = 스펙트럼 계수 인덱스

[0101] N = 윈도우 길이/프레임 길이의 2배

[0102] n_0 = (-N / 2 + 1) / 2

[0103] N = 960 또는 1024이다.

[0104] 윈도윙 및 오버랩 가산은 다음의 방식으로 행해진다:

[0105] 길이 N 윈도우는 길이가 2N인 윈도우로 대체되며, 과거에는 더 오버랩하게 미래에는 덜 오버랩한다 (N/8 값은 실제로 0이다).

[0106] 저 지연 윈도우에 대한 윈도윙:

[0107] $z_{i,n} = w_{LD}(n) \cdot x_{i,n}$

[0108] 여기서 윈도우는 이제2N의 길이를 가지므로, n=0,...,2N-1이다.

[0109] 오버랩 및 가산:

[0110] 0<=n<N/2인 경우에

[0111] $out_{i,n} = z_{i,n} + z_{i-1,n+\frac{N}{2}} + z_{i-2,n+N} + z_{i-3,n+N+\frac{N}{2}}$

[0112] 본 명세서에서, 단락은 14496-3:2009에 개정안 끝까지 포함되도록 위해 제안되었다.

[0113] 당연히, AAC-ELD에 대한 가능한 다운스케일링된 모드에 대한 상기 설명은 단지 본 출원의 일 실시예를 나타내고, 몇몇 수정이 가능하다. 일반적으로, 본 출원의 실시예는 AAC-ELD 디코딩의 다운스케일링된 버전을 수행하는 오디오 디코더에 제한되지 않는다. 다시 말해, 본 출원의 실시예는 예를 들어 스펙트럼 엔벨로프의 스케일 인자 기반 송신, TNS(temporal noise shaping) 필터링, 스펙트럼 대역 복제(spectral band replication, SBR) 등과 같은 예를 들어 다양한 AAC-ELD 특정 추가 작업을 지원하거나 사용하지 않고 다운스케일링된 방식으로 역 변환 프로세스를 수행할 수 있는 오디오 디코더를 형성함으로써 도출될 수 있다.

[0114] 이어서, 오디오 디코더에 대한 보다 일반적인 실시예가 설명된다. 설명된 다운스케일링된 모드를 지원하는 AAC-ELD 오디오 디코더에 대한 전술 한 예는 따라서 후술된 오디오 디코더의 구현예를 나타낼 수 있다. 특히, 후술하는 디코더가 도 2에 도시되어 있고, 한편 도 3은 도 2의 디코더에 의해 수행되는 단계를 도시하고 있다.

[0115] 일반적으로 참조 기호 10을 사용하여 나타내어진 도 2의 오디오 디코더는 , 수신기(12), 그래버(grabber, 14), 스펙트럼-시간 변조기(16), 윈도우어(18), 및 시간 도메인 앨리어싱 제거기(20)를 포함하며, 이들 모두는 언급된 순서대로 서로 직렬로 연결되어 있다. 오디오 디코더(10)의 블록(12 내지 20)의 상호 작용 및 기능은 도 3과 관련하여 다음에서 설명된다. 본 출원의 설명의 말미에서 설명된 바와 같이, 블록(12 내지 20)은 컴퓨터 프로그램, FPGA 또는 적절하게 프로그래밍된 컴퓨터, 프로그래밍된 마이크로프로세서 또는 애플리케이션 특정 통합 회로와 같은 소프트웨어, 프로그램 가능한 하드웨어 또는 하드웨어로 구현될 수 있으며, 블록(12 내지 20)은 각각의 서브 루틴, 회로 경로 등을 나타낸다.

[0116] 아래에서 보다 자세하게 설명되는 방식으로, 도 2의 오디오 디코더(10)는 오디오 디코더(10)의 요소가 적절하게 협동하도록 구성되고, 데이터 스트림(24)으로부터의 오디오 신호(22)를 디코딩하도록 구성되며, 오디오 디코더(10)는 오디오 신호(22)가 인코딩 측에서 데이터 스트림(24)으로 변환 코딩된 샘플링 속도의 1/F인 샘플링 속도로 신호(22)를 디코딩한다는 것에 주목할 만한다. F는 예를 들어 1보다 큰 임의의 유리수일 수 있다. 오디오 디코더는 상이한 또는 상이한 또는 다양한 다운스케일링 인자 F 또는 고정된 인자에서 동작하도록 구성될 수 있다. 대안예가 아래에서 보다 자세히 설명된다.

[0117] 오디오 신호(22)가 인코딩 또는 원래의 샘플링 속도에서 데이터 스트림으로 변환 코딩된 방식이 도 3의 상반부에 도시되어 있다. 26에서, 도 3은 각각 도 3에서 수평으로 연장되는 시간축(30) 및 도 3에서 수직으로 연장되는 주파수 축(32)을 따라 스펙트럼 시간(spectrotemporal) 방식으로 배열된 작은 박스 또는 정사각형(28)을 사용하는 스펙트럼 계수를 도시한다. 스펙트럼 계수(28)는 데이터 스트림(24) 내에서 송신된다. 스펙트럼 계수(28)가 획득된 방식, 및 그에 따른 스펙트럼 계수(28)가 오디오 신호(22)를 나타내는 방식이 도 3의 34에 도시되어 있으며, 이는 시간축(30)의 일부분에 대해 어떻게 스펙트럼 계수(28) 각각의 시간 부분에 속하거나, 각각의 시간 부분을 나타내거나, 오디오 신호로부터 획득되었는지를 도시한다.

[0118] 특히, 데이터 스트림(24) 내에서 송신된 계수(28)는 원래의 또는 인코딩 샘플링 속도로 샘플링된 오디오 신호(22)가 미리 결정된 길이 N의 즉시 시간적으로 연속적이고 오버랩하지 않는 프레임으로 분할되도록 오디오 신호(22)의 랩핑 변환의 계수이며, 여기서 N개의 스펙트럼 계수가 각각의 프레임(36)에 대해 데이터 스트림(24)에서 송신된다. 즉, 변환 계수(28)는 임계 샘플링된 랩핑된 변환을 사용하여 오디오 신호(22)로부터 획득된다. 스펙트럼 시간 스펙트로그램 표현(26)에서, 스펙트럼 계수(28)의 열의 시간 시퀀스의 각각의 열은 프레임 시퀀스의 프레임(36)의각각의 하나에 대응한다. N개의 스펙트럼 계수(28)는 스펙트럼 분해 변환 또는 시간-스펙트럼 변조에 의해 대응하는 프레임(36)에 걸쳐 획득되며, 변조 함수는 시간적으로 연장되나 결과 스펙트럼 계수(28)가 속하는 프레임(36)뿐만 아니라 E + 1 이전 프레임에 걸쳐 연장되며, 여기서 E는 0보다 큰 임의의 정수 또는 임의의 짝수일 수 있다. 즉, 특정 프레임(36)에 속하는 26에서의 스펙트로그램의 하나의 컬럼의 스펙트럼 계수(28)는 변환 윈도우 상에 변환을 적용함으로써 획득되며, 또한 각각의 프레임은 현재 프레임에 대해 과거에 존재하는 E + 1개의 프레임을 포함한다. 34에 도시된 부분의 중간 프레임(36)에 속하는 변환 계수(28)의 열에 대한 도 3에 도시된 이 변환 윈도우(38) 내의 오디오 신호의 샘플의 스펙트럼 분해는 변환 윈도우(38) 내의 스펙트럼 샘플이 동일한 MDCT 또는 MDST 또는 상이한 스펙트럼 분해 변환을 겪기 전에 가중되는 저 지연 단일 모드 분석 윈도우 함수(40)를 사용하여 달성된다. 인코더 측 지연을 낮추기 위해, 분석 윈도우(40)는 그 시간상 선단에 제로 간격(42)을 포함하여, 인코더는 이 현재 프레임(36)에 대한 스펙트럼 계수(28)를 계산하기 위해 현재 프레임(36) 내의 최신 샘플의 대응하는 부분을 기다릴 필요가 없다. 즉, 제로 간격(42) 내에서, 저 지연 윈도우 함수(40)는 0이거나 윈도우 계수가 0이므로, 현재 프레임(36)의 동일 위치의 오디오 샘플은 윈도우 가중치(40)로 인해 해당 프레임 및 데이터 스트림(24)에 대해 송신된 변환 계수(28)에 기여하지 않는다. 즉, 위의 내용을 요약

하면, 현재 프레임(36)에 속하는 변환 계수(28)는 현재 프레임뿐만 아니라 시간적으로 선행하는 프레임을 포함하고 시간적으로 이웃하는 프레임에 속하는 스펙트럼 계수(28)를 결정하기 위해 사용된 대응하는 변환 윈도우와 시간적으로 오버랩되는 변환 윈도우(38) 내의 오디오 신호의 샘플의 윈도윙 및 스펙트럼 분해에 의해 획득된다.

[0119] 오디오 디코더(10)의 설명을 다시 시작하기 전에, 지금까지 제공되는 바와 같이 데이터 스트림(24) 내의 스펙트럼 계수(28)의 송신에 대한 설명은 스펙트럼 계수(28)가 양자화되거나 데이터 스트림(24)으로 코딩되는 방식 및/또는 오디오 신호(22)가 오디오 신호가 랩핑 변환을 겪기 전에 사전 처리된 방식과 관련하여 단순화되었다는 것에 유의해야 한다. 예를 들어, 오디오 신호(22)를 데이터 스트림(24)으로 변환 코딩하는 오디오 인코더는 심리 음향 모델을 통해 제어될 수 있거나, 양자화 노이즈를 유지하고 청취자가 지각할 수 없고/없거나 마스킹 임계 함수 아래로 스펙트럼 계수(28)를 양자화하기 위해 심리 음향 모델을 사용하여, 양자화되고 송신된 스펙트럼 계수(28)가 스케일링되는 스펙트럼 대역에 대한 스케일 인자를 결정할 수 있다. 스케일 인자는 또한 데이터 스트림(24)에서 시그널링될 것이다. 대안적으로, 오디오 인코더는 TCX(transform coded excitation) 유형의 인코더일 수 있다. 그 다음, 오디오 신호는 여기 신호, 즉 선형 예측 잔여 신호 상에 랩핑된 변환을 적용함으로써 스펙트럼 계수(28)의 스펙트럼 시간 표현(26)을 형성하기 전에 선형 예측 분석 필터링을 받게 될 것이다. 예를 들어, 선형 예측 계수는 또한 데이터 스트림(24)에서 시그널링될 수 있고, 스펙트럼 계수(28)를 획득하기 위해 스펙트럼 균일 양자화가 적용될 수 있다.

[0120] 또한, 지금까지의 설명은 프레임(36)의 프레임 길이 및 / 또는 저 지연 윈도우 함수(40)에 대하여 단순화되었다. 실제로, 오디오 신호(22)는 가변 프레임 크기 및/또는 상이한 윈도우(40)를 사용하는 방식으로 데이터 스트림(24)으로 코딩될 수 있다. 그러나, 후술하는 설명은 하나의 윈도우(40) 및 하나의 프레임 길이에 집중되지만, 후속하는 설명은 엔트로피 인코더가 오디오 신호를 데이터 스트림으로 코딩하는 동안 이들 파라미터를 변경하는 경우에도 쉽게 확장될 수 있다.

[0121] 도 2의 오디오 디코더(10) 및 그 설명으로 되돌아 가서, 수신기(12)는 데이터 스트림(24)을 수신하고, 따라서 각각의 프레임(36)에 대해 N개의 스펙트럼 계수(28), 즉 도 3에 도시된 계수(28)의 각각의 열을 수신한다. 원래의 또는 인코딩 샘플링 속도의 샘플에서 측정된 프레임(36)의 시간적 길이는 도 3의 34에서 나타내어진 바와 같이 N이지만, 도 2의 오디오 디코더(10)는 감소된 샘플링 속도로 오디오 신호(22)를 디코딩하도록 구성된다는 것을 상기해야 한다. 오디오 디코더(10)는 예를 들어 이하에서 설명되는 이러한 다운스케일링된 디코딩 기능만을 지원한다. 대안적으로, 오디오 디코더(10)는 원래의 또는 인코딩 샘플링 속도로 오디오 신호를 재구성할 수 있지만, 다운스케일링되지 않은 디코딩 모드와 다운스케일링된 디코딩 모드 사이에서 스위칭될 수 있으며, 다운스케일링된 디코딩 모드는 이후 설명되는 오디오 디코더(10)의 동작 모드와 일치한다. 예를 들어, 오디오 인코더(10)는 낮은 배터리 레벨, 감소된 재생 환경 능력 등의 경우에 다운스케일링된 디코딩 모드로 스위칭될 수 있다. 상황이 바뀔 때마다, 오디오 디코더(10)는 예를 들어 다운스케일링된 디코딩 모드로부터 다운스케일링되지 않은 디코딩 모드로 다시 스위칭될 수 있다. 어느 경우에나, 이하에서 설명되는 디코더(10)의 다운스케일링된 디코딩 프로세스에 따라, 오디오 신호(22)는 프레임(36)이 감소된 샘플링 속도에서, 이 감소된 샘플링 속도의 샘플에서 측정된 더 낮은 길이, 즉 감소된 샘플링 속도에서의 N/F 샘플의 길이를 갖는 샘플링 속도에서 재구성된다.

[0122] 수신기(12)의 출력은 N개의 스펙트럼 계수의 시퀀스, 즉 프레임(36) 당 N개의 스펙트럼 계수의 하나의 세트, 즉 도 3의 하나의 열이다. 수신기(12)가 프레임(36) 당 N개의 스펙트럼 계수를 획득하는 데 있어서 다양한 작업을 적용할 수 있는 데이터 스트림(24)을 형성하기 위한 변환 코딩 프로세스의 상기 간략한 설명으로부터 이미 밝혀졌다. 예를 들어, 수신기(12)는 데이터 스트림(24)으로부터 스펙트럼 계수(28)를 판독하기 위해 엔트로피 디코딩을 사용할 수 있다. 수신기(12)는 또한 데이터 스트림에 제공된 스케일 인자 및/또는 데이터 스트림(24) 내에 전달된 선형 예측 계수에 의해 도출된 스케일 인자로 데이터 스트림으로부터 판독된 스펙트럼 계수를 스펙트럼적으로 형성할 수 있다. 예를 들어, 수신기(12)는 데이터 스트림(24)으로부터 스케일 인자를, 즉 프레임 당 및 서브 대역 단위로 획득할 수 있고, 데이터 스트림(24) 내에 전달된 스케일 인자를 스케일링하기 위해 이들 스케일 인자를 사용할 수 있다. 대안적으로, 수신기(12)는 각각의 프레임(36)에 대해 데이터 스트림(24) 내에서 전달된 선형 예측 계수로부터 스케일 인자를 도출하고, 송신된 스펙트럼 계수(28)를 스케일링하기 위해 이들 스케일 인자를 사용할 수 있다. 선택적으로, 수신기(12)는 프레임 당 N개의 스펙트럼 계수(18)의 세트 내의 0으로 양자화된 부분을 합성적으로 채우기 위해 갭 충전을 수행할 수 있다. 추가적으로 또는 대안적으로, 수신기(12)는 데이터 스트림으로부터 스펙트럼 계수(28)의 재구성을 돕기 위해 TNS 합성 필터를 프레임 당 송신된 TNS 필터 계수에 적용할 수 있으며, TNS 계수는 또한 데이터 스트림(24) 내에서 송신된다. 방금 설명한 수신기(12)의 가능한 작업은 가능한 측정치의 배타적이지 않은 목록으로 이해되어야 하고, 수신기(12)는 데이터 스트림(24)으

로부터의 스펙트럼 계수(28)의 판독과 관련하여 추가 또는 다른 작업을 수행할 수 있다.

[0123] 따라서, 그래버(14)는 수신기(12)로부터 스펙트럼 계수(28)의 스펙트로그램(26)을 수신하고, 각각의 프레임(36)에 대해 각각의 프레임(36)의 N개의 스펙트럼 계수의 저주파 부분(44), 즉 N/F 최저 주파수 스펙트럼 계수를 부여한다.

[0124] 즉, 스펙트럼-시간 변조기(16)는 그래버(14)로부터, 스펙트로그램(26)에서의 낮은 주파수 슬라이스에 대응하고, 도 3에서 인덱스 "0"을 사용하여 도시된 최저 주파수 스펙트럼 계수에 스펙트럼적으로 등록되고, 인덱스 N/F - 1의 스펙트럼 계수까지 확장되는, 프레임(36) 당 N/F 스펙트럼 계수(28)의 스트림 또는 시퀀스(46)를 수신한다.

[0125] 스펙트럼-시간 변조기(16)는 각각의 프레임(36)에 대해, 스펙트럼 계수(28)의 대응하는 저주파 부분(44)이 역 변환(48)을 받게 하여, 도 3 의 50에서 도시된 바와 같이 길이 (E + 2)·N/F의 변조 함수가 프레임 및 E + 1 이전 프레임에 걸쳐 시간적으로 확장되게 함으로써, 길이 (E + 2)·N/F의 시간 부분, 즉 아직 윈도윙되지 않은 시간 세그먼트(52)를 획득한다. 즉, 스펙트럼-시간 변조기는 예를 들어 전술된 제안된 대체 섹션 A.4의 제1 공식을 사용하여 동일한 길이의 변조 함수를 가중하고 합함으로써, 감소된 샘플링 속도의 (E + 2)·N/F 샘플의 시간적 시간 세그먼트를 획득할 수 있다. 시간 세그먼트(52)의 최신 N/F 샘플은 현재 프레임(36)에 속한다. 변조 함수는 나타내어진 바와 같이, 예를 들어 역 MDCT인 역 변환의 경우의 코사인 함수, 또는 역 MDCT인 역 변환의 경우에의 사인 함수가 될 수 있다.

[0126] 따라서, 윈도우어(52)는 각각의 프레임에 대해 시간 부분(52)을 수신하며, 그 선두에 있는 N/F 샘플은 각각의 프레임에 시간적으로 대응하고, 한편 각각의 시간 부분(52)의 다른 샘플은 대응하는 시간적으로 선행하는 프레임에 속한다. 윈도우어(18)는 각각의 프레임(36)에 대해, 그 선두에서 길이 1/4·N/F의 제로 부분(56), 즉 1/F·N/F 제로 값 윈도우 계수를 포함하고, 시간적으로 연속하는 시간적 간격 내에 피크(58), 제로 부분(56), 즉 제로 부분(52)에 의해 커버되지 않은 시간적 부분(52)의 시간적 간격을 갖는, 길이 (E + 2)·N/F의 단봉형 합성 윈도우(54)를 사용하여 시간적 부분(52)을 윈도윙한다. 후자의 시간 간격은 윈도우(58)의 비-제로 부분이라고 불릴 수 있으며, 감소된 샘플링 속도의 샘플, 즉 7/4·N/F 윈도우 계수에서 측정된 7/4·N/F의 길이를 갖는다. 윈도우어(18)는 예를 들어 윈도우어(58)를 사용하여 시간 부분(52)을 가중한다. 각각의 시간 부분(52)의 윈도우(54)에 대한 가중 또는 곱셈(58)은 윈도윙된 시간 부분(60)을 각각의 프레임(36)에 대해 하나씩 발생시키고, 시간 커버리지와 관련되는 한 각각의 시간 부분(52)과 일치한다. 위에서 제안한 섹션 A,4에서, 윈도우(18)에 의해 사용될 수 있는 윈도윙 처리는 $z_{i,n}$ 내지 $x_{i,n}$에 관한 공식에 의해 설명되며, 여기서 $x_{i,n}$은 아직 윈도윙되지 않은 전술한 시간 부분(52)에 대응하고, $z_{i,n}$은 윈도윙된 시간 부분(60)에 대응하고, i는 프레임/윈도우의 시퀀스를 인덱싱하고, n은 각각의 시간 부분(52/60) 내에서 감소된 샘플링 속도에 따라 각각의 부분(52/60)의 샘플 또는 값을 인덱싱한다.

[0127] 따라서, 시간 영역 앨리어싱 제거기(20)는 윈도윙된 시간 부분(60)의 시퀀스, 즉 프레임 당 하나의 윈도우를 윈도우어(18)로부터 수신한다. 제거기(20)는 대응하는 프레임(36)과 일치하도록 각각의 윈도윙된 시간 부분(60)을 그 선두의 N/F 값과 함께 등록함으로써 프레임(36)의 윈도윙된 시간 부분(60)이 오버랩 가산 프로세스(62)를 받게 한다. 이 방식에 의해, 현재 프레임의 윈도윙된 시간 부분(60)의 길이 (E + 1)/(E + 2)의 말단(trailing-end) 부분, 즉 길이 (E + 1) N/F을 갖는 나머지는 직전의 프레임의 시간 부분의 대응하는 동등하게 긴 선단과 오버랩된다. 공식에서, 시간 도메인 앨리어싱 제거기(20)는 섹션 A.4의 상기 제안된 버전의 마지막 공식에 도시된 바와 같이 동작할 수 있으며, 여기서, $out_{i,n}$은 감소된 샘플링 속도로 재구성된 오디오 신호(22)의 오디오 샘플에 대응한다.

[0128] 윈도우어(18) 및 시간 도메인 앨리어싱 제거기(20)에 의해 수행되는 윈도윙(58) 및 오버랩 가산(62)의 프로세스는 도 4와 관련하여 아래에보다 상세히 예시된다. 도 4는 위의 A.4 절에서 적용된 명명법 및 도 3 및 도 4에서 적용된 참조 부호를 사용한다. $x_{0,0}$ 내지 $x_{0,(E+2)\cdot N/F-1}$은 0번째 프레임(36)에 대해 공간-시간 변조기(16)에 의해 획득된 0번째 시간 부분(52)을 나타낸다. x의 제1 인덱스는 시간 순서에 따라 프레임(36)을 인덱싱하고, x의 제2 인덱스는 시간 순서에 따른 시간의 샘플을 순서를 정하고, 샘플 간 피치는 감소된 샘플 속도에 속한다. 그러면, 도 4에서, w_0 내지 $w_{(E+2)\cdot N/F-1}$은 윈도우(54)의 윈도우 계수를 나타낸다. x의 제2 인덱스, 즉 변조기(16)에 의해 출력된 시간 부분(52)과 같이, w의 인덱스는 윈도우(54)가 각각의 시간 부분(52)에 적용되는 경우에, 인덱스 0이 가장 오래된 것에 대응하고 인덱스 (E + 2)·N/F-1은 최신 샘플 값에 대응한다. 윈도우어(18)는 윈도우(54)를 사용하여 시간 부분(52)을 윈도윙하여 윈도윙된 시간 부분(60)을 획득하여, 0번째 프레임에 대한 윈도윙된 시간 부분(60)을 나타내는 $z_{0,0}$ 내지 $z_{0,(E+2)\cdot N/F-1}$이 $z_{0,0} = x_{0,0}\cdot w_0$, …, $z_{0,(E+2)\cdot N/F-1} = x0,(E+2)\cdot N/F-1\cdot w(E+2)\cdot N/F-1$에

따라 획득된다. z의 인덱스는 x와 동일한 의미를 갖는다. 이러한 방식으로, 변조기(16) 및 윈도우어(18)는 x 및 z의 제1 인덱스에 의해 인덱싱된 각각의 프레임에 대해 작용한다. 제어기(20)는 E + 2의 바로 직전에 연속하는 프레임을 E + 2 윈도윙되 시간 부분(60)을 합하여, 하나의 프레임만큼, 즉 프레임(36)당 샘플의 수만큼, 즉 N/F 만큼 서로에 대해 윈도윙된 시간 부분(60)의 샘플을 오프셋하여, 하나의 현재 프레임의 샘플 u를 획득하며, 여기서 $u_{-(E+1),0}$ … $u_{-(E+1),N/F-1}$이다. 여기서, 다시, u의 제1 인덱스는 프레임 번호를 나타내고, 제2 인덱스는 시간 순서에 따라 이 프레임의 샘플을 순서를 매긴다. 제거기는 재구성된 프레임을 결합하고 따라서 $u_{-(E+1),0}$ … $u_{-(E+1),N/F-1}$, $u_{-E,0}$, … $u_{-E,N/F-1}$, $u_{-(E-1),0}$, … 에 따라 서로 뒤따르는 연속적인 프레임(36) 내에서 재구성된 오디오 신호(22)의 샘플을 획득한다. 제거기(22)는 $u_{-(E+1),0} = z_{0,0} + z_{-1,N/F} + \cdots z_{-(E+1),(E+1)N/F}$, …, $u_{-(E+1)N/F-1} = z_{0,N/F-1} + z_{-1,2\cdot N/F-1} + \cdots + z_{-(E+1),(E+2)\cdot N/F-1}$에 따라 -(E+1)번째 프레임 내의 오디오 신호(22)의 각각의 샘플을 계산한다, 즉 현재 프레임의 샘플 u 당 (e+2) 가수를 합산한다.

[0129] 도 5는 프레임 -(E + 1)의 오디오 샘플 u에 기여하는 방금 윈도윙된 샘플 중에서, 윈도우(54)의 제로 부분(56), 즉 $z_{-(E+1),(E+7/4)\cdot N/F}$ … $z_{-(E+1),(E+2)\cdot N/F-1}$에 대응하고 그를 사용하여 윈도윙된 샘플이 제로 값을 갖는 가능한 이용예를 도시한다. 따라서, E+2 가수를 사용하여 오디오 신호 u의 -(E+1)번째 프레임(36) 내의 모든 N/F 샘플을 획득하는 대신에, 제거기(20)는 $u_{-(E+1),(E+7/4)\cdot N/F} = z_{0,3/4\cdot N/F} + z_{-1,7/4\cdot N/F} + \cdots + z_{-E,(E+3/4)\cdot N/F}$, …, $u_{-(E+1),(E+2)\cdot N/F-1} = z_{0,N/F-1} + z_{-1,2\cdot N/F-1} + \cdots + z_{-E,(E+1)\cdot N/F-1}$에 따라, 단지 E+1 가수를 사용하여, 그 선단, 즉 $u_{-(E+1),(E+7/4)\cdot N/F}$ … $u_{-(E+1),(E+2)\cdot N/F-1}$을 계산할 수 있다. 이러한 방식으로, 윈도우어는 제로 부분(56)에 대한 가중치(58)의 성능을 효과적으로 제거할 수 있다. 따라서, 현재 -(E+1)번째 프레임의 샘플 $u_{-(E+1),(E+7/4)\cdot N/F}$ … $u_{-(E+1),(E+2)\cdot N/F-1}$은 E+1 가수만을 사용하여 획득될 것이고, 한편 $u_{-(E+1),(E+1)\cdot N/F}$ … $u_{-(E+1),(E+7/4)\cdot N/F-1}$은 E+2 가수를 사용하여 획득될 것이다.

[0130] 따라서, 전술한 방식으로, 도 2의 오디오 디코더(10)는 데이터 스트림(24)으로 코딩된 오디오 신호를 다운스케일링된 방식으로 재생한다. 이를 위해, 오디오 디코더(10)는 길이 (E+2)·N의 기준 합성 윈도우의 다운샘플링된 버전인 윈도우 함수(54)를 사용한다. 도 6과 관련하여 설명된 바와 같이, 이 다운샘플링된 버전, 즉 윈도우(54)는 세그먼트 보간, 즉 아직 다운스케일링되지 않은 체제에서 측정되는 경우 길이 1/4·N의 세그먼트, 다운스케일링된 체제에서 길이 1/4·N/F의 세그먼트, 샘플링 속도와 독립적으로 시간적으로 측정되고 표현된 프레임(36)의 프레임 길이의 1/4의 세그먼트, 사용하여, 인자 F, 즉 다운샘플링 인자로 기준 합성 윈도우를 다운샘플링함으로써 획득된다. 따라서, 4 (E + 2)에서, 보간이 수행되어, 4 (E + 2) 배의 1/4N/F 길이의 세그먼트를 생성하며, 이는 길이 (E+2)·N의 기준 합성 윈도우의 다운샘플링된 버전을 연결하여 나타낸다. 예시를 위해 도 6을 참조한다. 도 6은 단봉형이고 길이가 (E+2)·N인 기준 합성 윈도우(70) 하에서 다운샘플링된 오디오 디코딩 절차에 따라 오디오 디코더(10)에 의해 사용되는 합성 윈도우(54)를 도시한다. 즉, 기준 합성 윈도우(70)로부터 다운샘플링된 디코딩을 위해 오디오 디코더(10)에 의해 실제로 사용되는 합성 윈도우(54)으로 이어지는 다운샘플링 절차(72)에 의해, 윈도우 계수의 수는 인자 F만큼 감소된다. 도 6에서, 도 5 및 도 6의 명명법이 사용되었다, 즉 w는 다운샘플링된 버전 윈도우(54)를 나타내기 위해 사용되고, 한편 w'는 기준 합성 윈도우(70)의 윈도우 계수를 나타내는 데 사용되었다.

[0131] 방금 언급한 바와 같이, 다운샘플링(72)을 수행하기 위해, 기준 합성 윈도우(70)는 동일한 길이의 세그먼트(74)로 처리된다. 번호에는, (E+2)·4개의 세그먼트(74)가 있다. 원래의 샘플링 속도, 즉 기준 합성 윈도우(70)의 윈도우 계수의 수로 측정되면, 각각의 세그먼트(74)는 1/4·N 윈도우 계수 w' 길이이고, 감소된 또는 다운샘플링된 샘플링 속도로 측정되면, 각각의 세그먼트(74)는 1/4·N/F 윈도우 계수 w 길이이다.

[0132] 당연히, 단순히 $w_i = w'_j$이고, 샘플링 시간 w_i이 w'_j의 샘플링 시간과 일치하도록 설정함으로써, 및/또는 선형 보간에 의해 2개의 윈도우 계수 w'_j 및 w'_{j+2} 사이에 일시적으로 존재하는 임의의 윈도우 계수 w_i를 선형적으로 보간함으로써, 기준 합성 윈도우(70)의 윈도우 계수 중 임의의 것 w'_j과 우연히 일치하는 각각의 다운샘플링된 윈도우 계수 w_i에 대해 다운샘플링(72)을 수행하는 것이 가능할 것이나, 이 절차는 기준 합성 윈도우(70)의 좋지 않은 근사치를 초래할 것이다, 즉 다운샘플링된 디코딩을 위해 오디오 디코더(10)에 의해 사용되는 합성 윈도우(54)는 기준 합성 윈도우(70)의 좋지 않은 근사치를 나타낼 것이며, 따라서 데이터 스트림(24)으로부터의 오디오 신호의 다운스케일링되지 않은 디코딩에 비해 다운스케일링된 디코딩의 적합성 테스트를 보장하는 요구를 만족시키지 않을 것이다. 따라서, 다운샘플링(72)은 보간 절차를 수반하며, 보간 절차에 따라 다운샘플링된 윈도

우(54)의 윈도우 계수 w_i의 대부분, 즉 세그먼트(74)의 경계로부터 오프셋된 위치에 있는 윈도우 계수 w_i는 기준 윈도우(70)의 2개를 초과하는 윈도우 계수 w'에 대한 다운샘플링 절차(72)에 의존한다. 특히, 다운샘플링된 윈도우(54)의 윈도우 계수 w_i의 대부분은 보간/다운샘플링 결과의 품질, 즉 근사화 품질을 증가시키기 위해 기준 윈도우(70)의 2개를 초과하는 윈도우 계수 w'_j에 의존하는데 반해, 다운샘플링된 버전(54)의 모든 윈도우 계수 w_i에 대해, 이는 동일한 세그먼트가 상이한 세그먼트(74)에 속하는 윈도우 계수 w'_j에 의존하지 않는다는 것을 유지한다. 오히려, 다운샘플링 절차(72)는 세그먼트 보간 절차이다.

[0133] 예를 들어, 합성 윈도우(54)는 길이 1/4/·N/F의 스플라인 함수의 연결일 수 있다. 입방체 스플라인 함수가 사용될 수 있다. 이러한 예는 섹션 A.1에서 위에 설명하였으며, 여기서 다음 루프에 대한 외부의 것은 세그먼트(74)에 대해 순차적으로 루프되며, 여기서, 각각의 세그먼트(74)에서, 다운샘플링 또는 보간(72)은 예를 들어 섹션의 다음 절의 첫 번째 부분 "계수 c를 계산하는 데 필요한 벡터를 계산한다" 에서 현재 세그먼트(74) 내의 연속적인 윈도우 계수들 w '의 수학적 조합을 포함한다. 그러나, 세그먼트에 적용된 보간은 다르게 선택될 수도 있다. 즉, 보간은 스플라인 또는 입방체 스플라인에만 국한되지 않다. 오히려, 선형 보간 또는 임의의 다른 보간 방법이 또한 사용될 수 있다. 임의의 경우에, 보간의 세그먼트 구현은 다른 세그먼트에 있는 기준 합성 윈도우의 윈도우 계수에 의존하지 않도록, 다운스케일링된 합성 윈도우의 샘플, 즉 다른 세그먼트에 인접하는, 다운스케일링된 합성 윈도우의 세그먼트의 최외측 샘플의 계산을 야기할 것이다.

[0134] 윈도우어(18)는 이 다운샘플링된 합성 윈도우(54)의 윈도우 계수 wi가 다운샘플링(72)을 사용하여 획득된 후에 저장되어 있는 스토리지로부터 다운샘플링된 합성 윈도우(54)를 획득 할 수 있다. 대안적으로, 도 2에 도시된 바와 같이, 오디오 디코더(10)는 기준 합성 윈도우(70)에 기초하여 도 6의 다운샘플링(72)을 수행하는 세그먼트 다운샘플러(76)를 포함할 수 있다.

[0135] 도 2의 오디오 디코더(10)는 단지 하나의 고정된 다운샘플링 인자 F만을 지원하도록 구성될 수 있거나 상이한 값을 지원할 수 있음에 유의해야 한다. 그 경우에, 오디오 디코더(10)는 도 2의 78에서 도시된 바와 같이 F에 대한 입력 값에 응답할 수 있다. 예를 들어 그래버(14)는 전술한 바와 같이, 프레임 스펙트럼 당 N/F 스펙트럼 값을 얻기 위해 이 값 F에 응답할 수 있다. 유사한 방식으로, 임의적인 세그먼트 다운샘플러(76)가 또한 전술한 바와 같이 동작하는 이 F 값에 응답할 수 있다. S/T 변조기(16)는 F에 응답하여, 예를 들어, 변조 함수의 다운스케일링된/다운샘플링된 버전을 계산적으로 도출하고, 다운스케일링되지 않은 동작 모드에서 사용된 것과 비교하여 다운스케일링/다운샘플링할 수 있으며, 여기서 재구성은 전체 오디오 샘플 속도를 야기한다.

[0136] 당연히, 변조기(16)는 또한 F 입력(78)에 응답할 것인데, 변조기(16)는 변조 함수의 적절히 다운샘플링된 버전을 사용할 것이고, 감소된 샘플링 속도 또는 다운샘플링된 샘플링 속도의 프레임의 실제 길이의 적응에 관해서는 윈도우어(18) 및 제거기(20)에 대해 동일하게 적용될 것이기 때문이다.

[0137] 예를 들어, F는 1.5 및 10을 포함하여, 1.5와 10 사이에 있을 수 있다.

[0138] 도 2 및 도 3의 디코더 또는 본 명세서에 설명된 임의의 수정예는 예를 들어, EP 2 378 516 B1에 개시된 바와 같이 저 지연 MDCT의 리프팅 구현을 사용하여 스펙트럼-시간 전이를 수행하도록 구현될 수 있음에 유의한다.

[0139] 도 8은 리프팅 개념을 사용하는 디코더의 구현예를 도시한다. S/T 변조기(16)는 예시적으로 역 DCT-IV를 수행하고, 뒤이어서, 윈도우어(18) 및 시간 도메인 앨리어싱 제거기(20)의 연결을 나타내는 블록이 도시된다. 도 8의 예에서, E는 2, 즉 E=2이다.

[0140] 변조기(16)는 역 타입 iv 이산 코사인 변환 주파수/시간 컨버터를 포함한다. (E+2)N/F 긴 시간 부분(52)의 시퀀스를 출력하는 대신에, 그것은 N/F 긴 스펙트럼(46)의 시퀀스로부터 유도된 길이 2N/F의 시간 부분(52)을 출력할 뿐이고, 이들 단축된 부분(52)은 DCT 커널, 즉 상기 기술된 부분의 2 N/F 최신 샘플에 대응한다.

[0141] 윈도우어(18)는 전술한 바와 같이 동작하고 각각의 시간 부분(52)에 대해 윈도윙 시간 부분(60)을 생성하지만, 단지 DCT 커널에서만 동작한다. 이를 위해, 윈도우어(18)는 커널 크기를 갖는, i=0 ... 2N / F-1인 윈도우 함수 ω_i를 사용한다. i=0...(E+2)N/F-1인 wi 사이의 관계가 추후 설명될 것이며, 후속하여 설명된 리프팅 계수와 i=0...(E+2)N/F-1인 w_i 사이의 관계가 설명될 것이다.

[0142] 위에 적용된 명명법을 사용하여, 지금까지 설명된 프로세스는 다음을 산출한다:

[0143] $n = 0,\ldots,2M-1$ 인 경우에, $z_{k,n} = \omega_n \cdot x_{k,n}$

[0144] M = N/F으로 재 정의하며, M은 다운스케일링된 도메인에서 표현된 프레임 크기에 대응하고, 도 2-6의 명명법을 사용하며, 여기서, 그러나, $z_{k,n}$ 및 $x_{k,n}$은 크기 2M을 가지며 도 4의 샘플 EN/F... (E+2)N/F-1에 시간적으로 대응하는 DCT 커널 내의 윈도윙된 시간 부분 및 아직 윈도윙되지 않은 시간 부분의 샘플만을 포함할 것이다. 즉, n은 샘플 인덱스를 나타내는 정수이고, ω_n은 샘플 인덱스 n에 대응하는 실수 값 윈도우 함수 계수이다.

[0145] 제거기(20)의 오버랩/가산 프로세스는 상기 설명과 비교하여 상이한 방식으로 동작한다. 다음의 방정식 또는 수식에 기초하여 중간 보간 부분 mk(0),...mk(M-1)을 생성한다.

[0146] $n = 0,\ldots,M-1$인 경우에, $m_{k,n} = z_{k,n} + z_{k-1,n+M}$

[0147] 도 8의 실시예에서, 장치는 변조기(16) 및 윈도우어(18)의 일부로서 해석될 수 있는 리프터(80)를 더 포함하는데, 리프터(80)는 변조기 및 윈도어가 확장이 도입되어 제로 부분(56)을 보상하는 과거를 향해서 커널을 넘어서 변조 함수 및 합성 윈도우의 확장의 처리 대신에 DCT 커널에 대한 처리를 제한한 것을 보상하기 때문이다. 리프터(80)는 지연기 및 승산기(82) 및 가산기(84)의 프레임워크를 사용하여 다음의 방정식 또는 표현에 기초하여 바로 연속하는 프레임의 쌍에서 길이 M의 최종적으로 재구성된 시간 부분 또는 프레임을 생성한다.

[0148] $n = M/2,\ldots,M-1$인 경우에, $u_{k,n} = m_{k,n} + l_{n-M/2} \cdot m_{k-1,M-1-n}$

[0149] 및

[0150] $n=0,\ldots,M/2-1$ 인 경우에, $u_{k,n} = m_{k,n} + l_{M-1-n} \cdot out_{k-1,M-1-n}$

[0151] 여기서 ln(여기서 n=0...M-1)은 아래에서 보다 상세하게 설명되는 방식으로 다운스케일링된 합성 윈도우와 관련된 실수 값 리프팅 계수이다.

[0152] 다시 말해, E 프레임이 과거로 확장된 오버랩의 경우, 리프터(80)의 프레임워크에서 볼 수 있는 바와 같이 M개의 추가 승수-가산 연산만 필요하다. 이러한 추가 연산은 때로는 "제로 지연 행렬"이라고도 한다. 때로는 이러한 연산은 "리프팅 단계"라고도 알려져 있다. 도 8의 효율적인 구현은 어떤 상황 하에서는 직접 구현보다 효율적일 수 있다. 보다 구체적으로, 구체적인 구현에 의존하여, 그러한 보다 효율적인 구현은 M 연산에 대한 직접 구현의 경우와 같이 M 연산을 절약하게 할 수 있으며, 도 19에 도시된 구현예와 같이 구현하는 것이 바람직할 수 있으며, 원칙적으로 모듈(820)의 프레임 워크에서의 2M 연산 및 리프터(830)의 프레임워크에서의 M 연산을 필요로 한다.

[0153] 합성 윈도우어 w_i(여기서 i = 0...(E+2)M-1)에 대한 ω_n(여기서 n=0...2M-1) 및 l_n(여기서 n=0...M-1)의 의존성에 관해서는 (E=2임을 상기한다), 다음 공식은 그것들을 치환하는 것과의 관계를 설명하고 있지만, 지금까지 각각의 변수에 따라 괄호 안에 사용된 첨자 인덱스는 다음과 같다:

[0154] $i, n = 0 \ldots \frac{M}{2} - 1$ 인 경우,

$$w(i) = l\left(\frac{M}{2} - 1 - n\right) \cdot l(M - 1 - n) \cdot \omega(M + n)$$

$$w(M/2 + i) = l(n) \cdot l(M/2 + n) \cdot \omega(3M/2 + n)$$

$$w(M + i) = l\left(\frac{M}{2} - 1 - n\right) \cdot \omega(M + n)$$

$$w(3M/2 + i) = -l(n) \cdot \omega(3M/2 + n)$$

$$w(2M + i) = -\omega(M + n) - l(M - 1 - n) \cdot \omega(n)$$

$$w(5M/2 + i) = -\omega(3M/2 + n) - l(M/2 + n) \cdot \omega(M/2 + n)$$

$$w(3M + i) = -\omega(n)$$

[0155]

[0156] $w(7M/2 + i) = \omega(M + n)$

[0157] 윈도우 wi는 이 공식의 우측에, 즉 인덱스 2M과 인덱스 4M-1 사이에 피크 값을 포함한다는 것에 유의한다. 위의

공식은 다운스케일링된 합성 윈도우의 계수 ω_n(여기서 n=0...(E+2)M)에 계수 l_n(여기서 n = 0...M-1 및 n n = 0,...,2M-1)을 관련시킨다. 알 수 있는 바와 같이, l_n(여기서 n=0...M-1)은 실제로는 단지 다운샘플링된 합성 윈도우의 계수의 ¾, 즉 ωn(여기서 n=0...(E+1)M-1)에 의존한다.

[0158] 전술한 바와 같이, 윈도우어(18)는 wi 스토리지로부터 다운샘플링된 합성 윈도우(54, ω_n, 여기서 n=0...(E+2)M-1)를 획득할 수 있으며, 스토리지는 이 다운샘플링된 합성 윈도우(54)의 윈도우 계수가 다운샘플링(72)을 사용하여 획득된 후에 저장되는 곳이고, 이 스토리지로부터 다운샘플링된 합성 윈도우(54)의 윈도우 계수가 판독되어 위의 관계식을 사용하여 계수 l_n(여기서 n=0...M-1) 및 ω_n(여기서 n=0,...,2M-1)을 계산하고, 대안적으로, 윈도우어(18)는 계수 l_n(여기서 n = 0...M-1) 및 ω_n(여기서 n = 0,…,2M-1)을 검색하고, 따라서 스토리지로부터 직접, 사전 다운샘플링된 합성 윈도우로부터 계산할 수 있다. 대안적으로, 전술한 바와 같이, 오디오 디코더(10)는 기준 합성 윈도우(70)기초하여 도 6의 다운샘플링(72)을 수행함으로써, ω_n(여기서 n=0...(E+2)M-1)을 산출하는 세그먼트 다운샘플러(76)를 포함할 수 있으며, 이에 기초하여 윈도우어(18)는 위의 관계식/공식을 사용하여 계수 l_n(여기서 n = 0,...,M-1) 및 ω_n(여기서 n = 0,...,2M-1)을 계산한다. 리프팅 구현을 사용하더라도, F에 대해 하나를 초과하는 값이 지원될 수 있다.

[0159] 리프팅 구현을 간략하게 요약하면, 오디오 신호가 제2 샘플링 속도로 변화 코딩되는 데이터 스트림(24)으로부터 제1 샘플링 속도에서 오디오 신호(22)를 디코딩하도록 구성된 오디오 디코더(10)에서도 동일한 결과를 얻으며, 제1 샘플링 속도는 제2 샘플링 속도의 1/F이고, 오디오 디코더(10)는 오디오 신호의 길이 N의 프레임 당, N개의 스펙트럼 계수(28)를 수신하는 수신기(12), 각각의 프레임에 대해, N개의 스펙트럼 계수(28)에서 길이 N/F의 저주파 부분을 잡아내는(grab) 그래버(14), 각각의 프레임(36)에 대해, 저주파 부분이 각각의 프레임 및 이전 프레임에 걸쳐 시간적으로 확장되는 길이 2N/F의 변조 함수를 갖는 역 변환을 받게 하여 길이 2N/F의 시간 부분을 획득하도록 구성된 스펙트럼-시간 변조기(16), 및 각각의 프레임(36)에 대해, $z_{k,n} = \omega_n \cdot x_{k,n}$(여기서 n = 0,...,2M-1)에 따라 시간 부분 $x_{k,n}$을 윈도윙하여 윈도윙된 시간 부분 $z_{k,n}$(여기서 n=0,...,2M-1)을 획득하는 윈도우어(18)를 포함한다. 시간 도메인 앨리어싱 제거기(20)는 $m_{k,n} = z_{k,n} + z_{k-1,n+M}$(여기서 n = 0,...,M-1)에 따라 중간 시간 부분 $m_k(0),...m_k(M-1)$을 생성한다. 마지막으로, 리프터(80)는 $u_{k,n} = m_{k,n} + l_{n-M/2} \cdot m_{k-1,M-1-n}$(여기서n = M/2,...,M-1) 및 $u_{k,n} = m_{k,n} + l_{M-1-n} \cdot out_{k-1,M-1-n}$(여기서 n=0,...,M/2-1)에 따라 오디오 신호의 프레임 $u_{k,n}$(여기서 n = 0...M-1)을 계산하고, 여기서 l_n(여기서 n = 0...M-1)은 리프팅 계수이고, 여기서 역 변환은 역 MDCT 또는 역 MDST이고, 여기서 l_n(여기서 n = 0…M-1) 및 ω_n(여기서 n = 0,...,2M-1)은 합성 윈도우의 계수 ω_n(여기서 n = 0,...,(E+2)M-1)에 의존하고, 합성 윈도우는 길이 1/4·N의 세그먼트에서의 세그먼트 보간에 의해 F 인자만큼 다운샘플링된, 길이 4·N의 기준 합성 윈도우의 다운샘플링된 버전이다.

[0160] 도 2의 오디오 디코더가 저 지연 SBR 도구를 수반할 수 있는 다운스케일링된 디코딩 모드와 관련하여 AAC-ELD의 확장을 위한 제안에 대한 상기 논의로부터 이미 밝혀졌다. 다음은 예를 들어 AAC-ELD 코더가 위에서 제안된 다운스케일링된 동작 모드를 지원하기 위해 확장된 방법이 저 지연 SBR 도구를 사용하는 경우에 동작할 것을 개략적으로 설명한다. 본 출원의 명세서의 소개 부분에서 이미 언급한 바와 같이, 저 지연 SBR 도구가 AAC-ELD 코더와 관련하여 사용되는 경우, 저 지연 SBR 모듈의 필터 뱅크가 또한 다운스케일링된다. 이는 SBR 모듈이 동일한 주파수 해상도로 연산하는 것을 보장하므로, 더 이상의 적응이 필요하지 않다. 도 7은 다운샘플링된 SBR 모드이고, 다운스케일링 계수 F가 2인, 96kHz에서 480 샘플의 프레임 크기로 동작하는 AAC-ELD 디코더의 신호 경로를 개략적으로 설명한다.

[0161] 도 7에서, 도착한 비트스트림은 블록의 시퀀스, 즉 AAC 디코더, 역 LD-MDCT 블록, CLDFB 분석 블록, SBR 디코더, 및 CLDFB 합성 블록(CLDFB = complex low delay filter bank)에 의해 처리된다. 비트스트림은 도 3 내지 도 6과 관련하여 앞서 논의된 데이터 스트림(24)과 동일하나,역 저 지연 MDCT 블록의 출력에서 다운스케일링된 오디오 디코딩에 의해 획득된 오디오 신호의 스펙트럼 주파수를 확장하는 스펙트럼 확장 대역의 스펙트럼 복제물의 스펙트럼 정형을 보조하는 파라메트릭 SBR 데이터를 부가적으로 수반하며, 상기 스펙트럼 정형은 수행된다 SBR 디코더에 의해 수행된다. 특히, AAC 디코더는 적절한 파싱 및 엔트로피 디코딩에 의해 필요한 모든 구문 요소를 검색한다. AAC 디코더는 도 7에서 역 저 지연 MDCT 블록에 의해 구현되는 오디오 디코더(10)의 수신기(12)와 부분적으로 일치할 수 있다. 도 7에서, F는 예시적으로 2와 동일하다. 즉, 도 7의 역 저 지연 MDCT 블록은 도 2의 재구성된 오디오 신호(22)에 대한 예로서, 오디오 신호가 원래 도착한 비트스트림으로 코딩된 속도의

절반으로 다운샘플링된 48kHz 시간 신호를 출력한다. CLDFB 분석 블록은 이 48kHz 시간 신호, 즉 다운스케일링된 오디오 디코딩에 의해 획득된 오디오 신호를 N개(여기서 N=16))의 대역으로 세분화하고, SBR 디코더는 이 대역에 대한 재정형 계수를 계산하고, 그에 따라 N개의 대역을 재정형하며, 이는 ACC 디코더에 도착하는 입력 비트스트림에서 SBR 데이터를 통해 제어되고, CLDFB 합성 블록은 스펙트럼 도메인에서 시간 도메인으로 재전이시킴으로써, 역 저 지연 MDCT 블록에 의해 출력되는 원래의 디코딩된 오디오 신호에 가산되는 고주파 확장 신호를 획득한다.

[0162] SBR의 표준 연산은 32 대역 CLDFB를 사용한다는 점에 유한다. 32 대역 CLDFB 윈도우 계수 ci_{32}에 대한 보간 알고리즘은 이미 [1]의 4.6.19.4.1에서 다음과 같이 주어져 있다.

[0163] $$ci_{32}(i) = \frac{1}{2}[c_{64}(2i+1) + c_{64}(2i)], \qquad 0 \le i < 320,$$

[0164] 여기서 c_{64}는 [1]의 표 4.A.90에 주어진 64 대역 윈도우의 윈도우 계수이다. 이 공식은 또한 더 낮은 수의 대역 B에 대한 윈도우 계수를 정의하기 위해 더 일반화될 수 있다.

[0165] $$ci_B(i) = \frac{1}{2}[c_{64}(2F \cdot i + 1 + p) + c_{64}(2F \cdot i + p)], \qquad 0 \le i < (10B), p = int\left(\frac{64}{2B} - 0.5\right)$$

[0166] 여기서 F는 F = 32/B인 다운스케일링 계수를 나타낸다. 윈도우 계수의 이러한 정의에 따라, CLDFB 분석 및 합성 필터 뱅크는 위의 섹션 A.2의 예에서 간략히 설명된 바와 같이 완전히 설명될 수 있다.

[0167] 따라서, 위의 예는 더 낮은 샘플 속도의 시스템에 코덱을 적용하기 위해 AAC-ELD 코덱에 대한 일부 누락된 정의를 제공했다. 이러한 정의는 ISO/IEC 14496-3:2009 표준에 포함될 수 있다.

[0168] 따라서, 위의 논의에서, 그것은 별칭으로 기술되었다:

[0169] 오디오 디코더는 오디오 신호가 제2 샘플링 속도로 변환 코딩되는 데이터 스트림으로부터 제1 샘플링 속도로 오디오 신호를 디코딩하도록 구성될 수 있으며, 제1 샘플링 속도는 제2 샘플링 속도의 1/F이고, 오디오 디코더는 오디오 신호의 길이 N의 프레임 당, N개의 스펙트럼 계수를 수신하도록 구성된 수신기; 각각의 프레임에 대해, N개의 스펙트럼 계수에서 길이 N/F의 저주파 부분을 잡아내도록 구성된 그래버; 각각의 프레임에 대해, 저주파 부분이 각각의 프레임 및 E+1 이전 프레임에 걸쳐 시간적으로 확장되는 길이 (E + 2)·N/F 의 변조 함수를 갖는 역 변환을 받게 하여 길이 (E + 2)·N/F의 시간 부분을 획득하도록 구성된 스펙트럼-시간 변조기; 각각의 프레임에 대해, 선단에서 길이 1/4·N/F의 제로 부분을 포함하고 단봉형 합성 윈도우의 시간 간격 내에 피크를 갖는 길이 (E + 2)·N/F의 단봉형 합성 윈도우를 사용하여 시간 부분을 윈도윙하도록 구성된 윈도우어로서, 시간 부분은 제로 부분이 연속되고 7/4·N/F의 길이를 가져, 윈도우어가 길이 (E + 2)·N/F의 윈도윙된 시간 부분을 획득하는, 윈도우어; 및 프레임의 윈도윙된 시간 부분이 오버랩-가산 프로세스를 받게 하여 현재 프레임의 윈도윙된 시간 부분의 길이 (E + 1)/(E + 2)의 말단 부분이 이전 프레임의 윈도윙된 시간 부분의 길이 (E + 1)/(E + 2)의 선단에 오버랩하도록 구성된 시간 도메인 앨리어싱 제거기를 포함하고, 여기서 역 변환은 역 MDCT 또는 역 MDST이고, 단봉형 합성 윈도우는 길이 1/4· N/F 의 세그먼트에서 세그먼트 보간에 의해 인자 F만큼 다운샘플링된, 길이 (E + 2)·N 의 기준 단봉형 합성 윈도우의 다운샘플링된 버전이다.

[0170] 일 실시예에 따른 오디오 디코더에 있어서, 단봉형 합성 윈도우는 길이 1/4·N/F의 스플라인 함수의 연결이다.

[0171] 일 실시예에 따른 오디오 디코더에 있어서, 단봉형 합성 윈도우는 길이 1/4·N/F의 입방체 스플라인 함수의 연결이다.

[0172] 이전의 실시예 중 어느 하나에 따른 오디오 디코더에 있어서, E=2이다.

[0173] 이전의 실시예 중 어느 하나에 따른 오디오 디코더에 있어서, 역 변환은 역 MDCT이다.

[0174] 이전의 실시예 중 어느 하나에 따른 오디오 디코더에 있어서, 단봉형 합성 윈도우의 80%를 초과하는 집단(mass)이 제로 부분에 뒤이어 오고 길이 7/4·N/F를 갖는 시간 간격 내에 포함된다.

[0175] 이전의 실시예 중 어느 하나에 따른 오디오 디코더에 있어서, 오디오 디코더는 상기 보간을 수행하거나 스토리지로부터 단봉형 합성 윈도우를 도출하도록 구성된다.

[0176] 이전의 실시예 중 어느 하나에 따른 오디오 디코더에 있어서, 오디오 디코더는 F에 대해 상이한 값을 지원하도

록 구성된다.

[0177] 이전의 실시예 중 어느 하나에 따른 오디오 디코더에 있어서, F는 1.5 및 10을 포함하여 1.5 내지 10 사이에 있다.

[0178] 이전의 실시예 중 어느 하나에 따른 오디오 디코더에 의해 수행되는 방법.

[0179] 컴퓨터 상에서 실행되는 경우, 일 실시예에 따른 방법을 수행하기 위한 프로그램 코드를 갖는 컴퓨터 프로그램.

[0180] 용어 "길이 ..."에 관한 한, 이 용어는 샘플의 길이를 측정하는 것으로 해석되어야 한다는 점에 유의한다. 제로 부분 및 세그먼트의 길이에 관해서는, 그것이 정수 값일 수 있다는 것에 유의해야 한다. 대안적으로, 그것은 정수가 아닌 값일 수 있다.

[0181] 피크가 위치되는 시간 간격에 관해서, 도 1은 기준 단봉형 합성 윈도우(여기서 E = 2 및 N = 512)의 예를 위해 예시적으로 이러한 피크뿐만 아니라 시간 간격을 도시한다는 것에 유의한다: 피크는 대략 샘플 번호 1408에서 최대치를 가지며 시간 간격은 샘플 번호 1024에서 샘플 번호 1920까지 확장된다. 따라서, 시간 간격은 DCT 커널의 7/8만큼 길다.

[0182] 용어 "다운샘플링된 버전"에 관해서는, 상기 명세서에서,이 용어 대신에, "다운스케일링된 버전"이 동의어로 사용되었다는 것에 유의한다.

[0183] 용어 "일정 간격 내에서 함수의 질량"은 각각의 간격 내에서 각각의 함수의 한정된 적분을 나타낸다는 것에 유의한다.

[0184] F에 대해 상이한 값을 지원하는 오디오 디코더의 경우, 기준 단봉형 합성 윈도우의 그에 따라 세그먼트로 보간된 버전을 갖는 스토리지를 포함할 수 있거나, F의 현재 활성 값에 대한 세그먼트 보간을 수행할 수 있다. 부분적으로 보간된 상이한 버전은 보간이 세그먼트 경계에서 불연속성에 부정적인 영향을 미치지 않는다는 공통점을 갖는다. 전술한 바와 같이, 함수는 스플라인 함수일 수 있다.

[0185] 위의 도 1에서 도시된 것과 같이 기준 단봉형 합성 윈도우로부터 세그먼트 보간에 의해 단봉형 합성 윈도우를 도출함으로써, 스플라인 근사에 의해 4(E + 2) 개의 세그먼트가 형성될 수 있으며, 이는 지연이 보정되는 것을 낮추기 위한 수단으로서 합성하여 도입된 제로 부분 때문에 1/4 N/F의 피치에서 단봉형 합성 윈도우에 존재할 것이다.

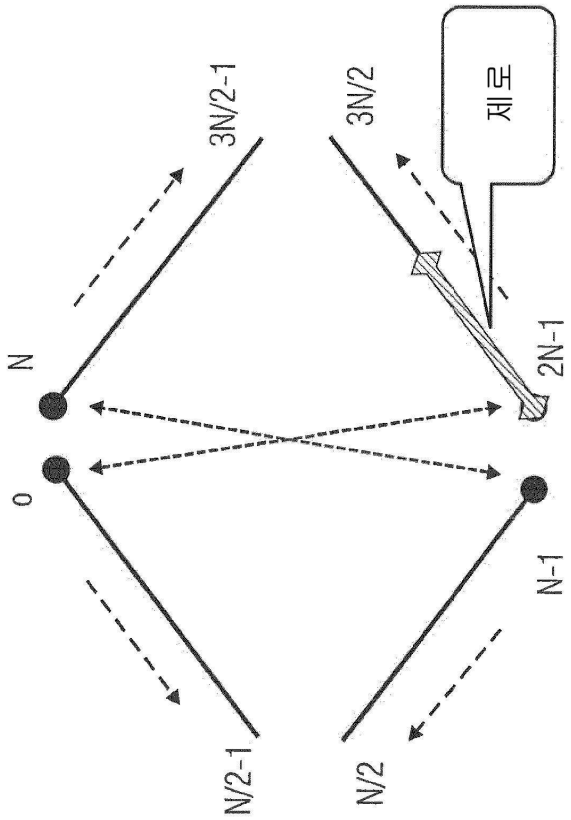
[0186] **참조문헌**

[0187] [1] ISO/IEC 14496-3:2009

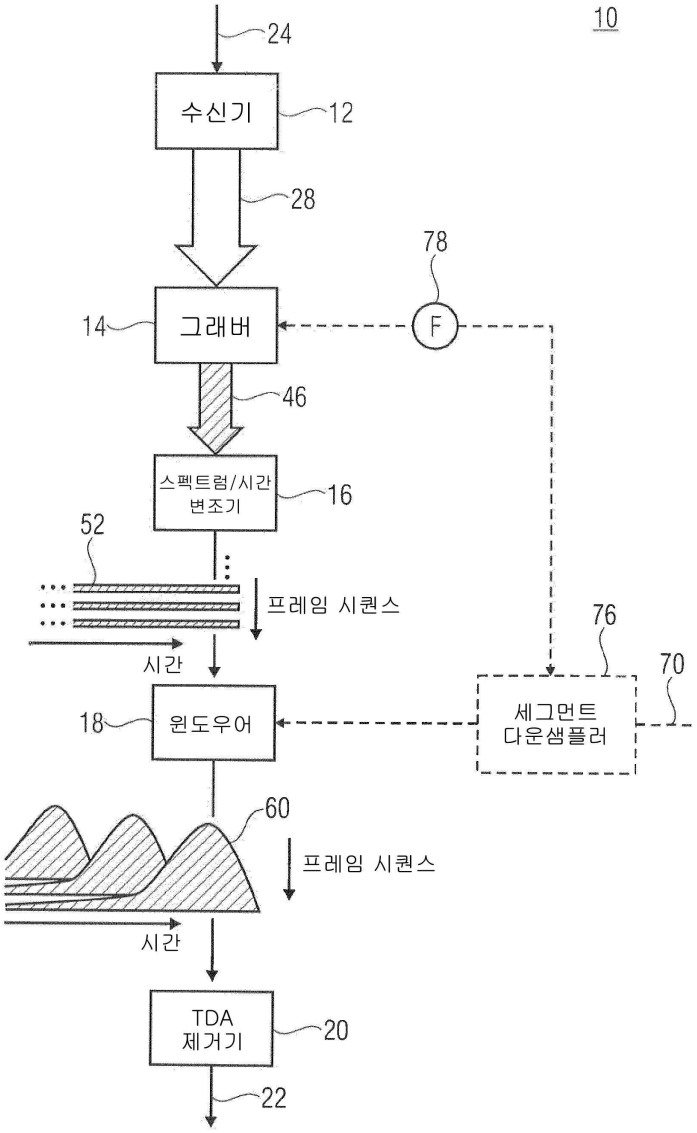
[0188] [2] M13958, “Proposal for an Enhanced Low Delay Coding Mode”, October 2006, Hangzhou, China

도면

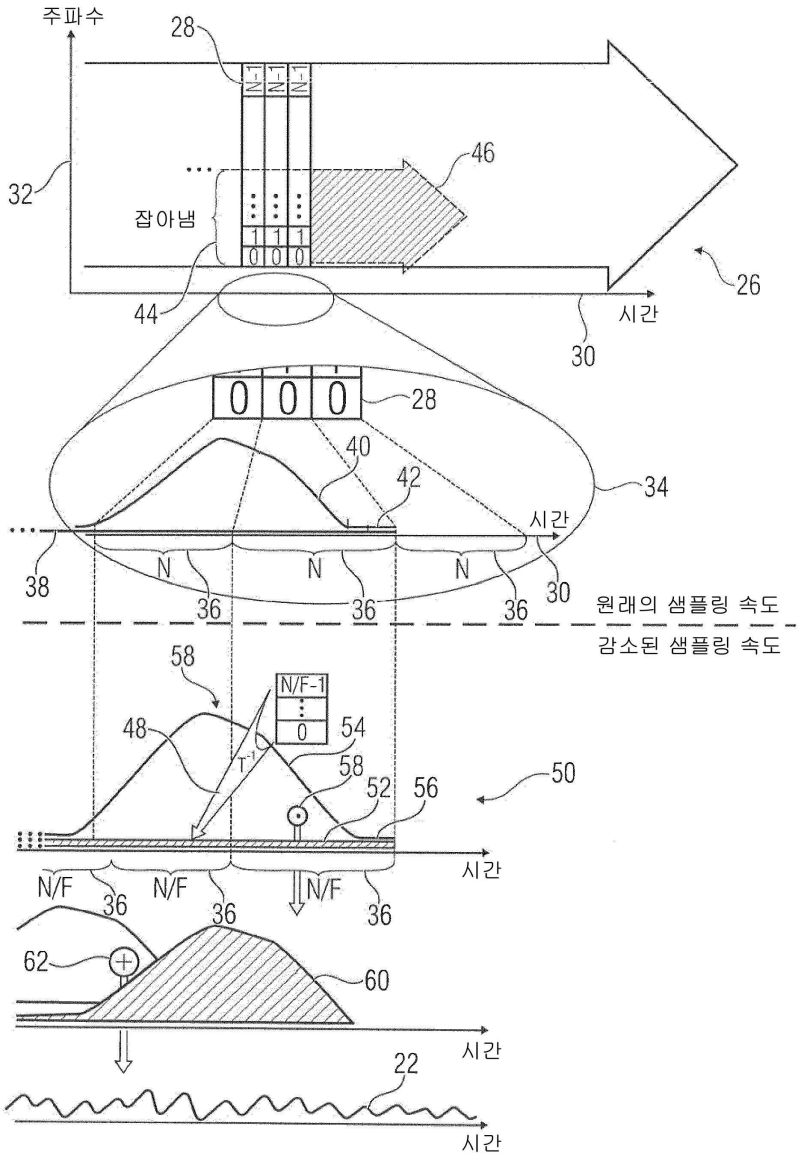
도면1



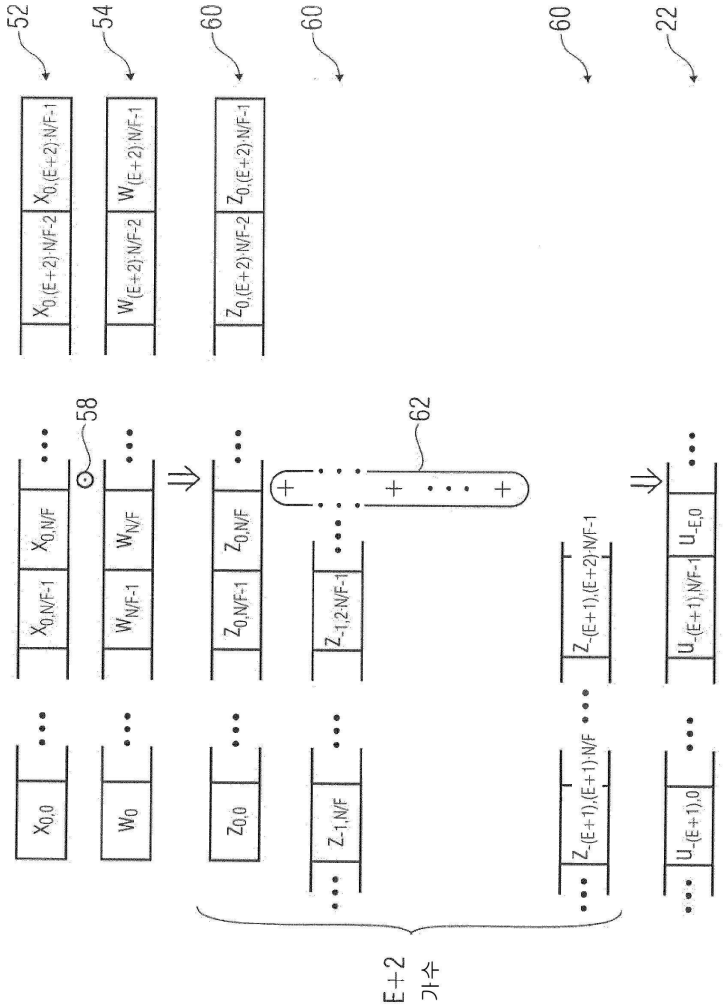
도면2


10
24
수신기
12
28
78
14
그래버
F
46
스펙트럼/시간
변조기
16
52
프레임 시퀀스
시간
76
70
18
윈도우어
세그먼트
다운샘플러
60
프레임 시퀀스
시간
TDA
제거기
20
22

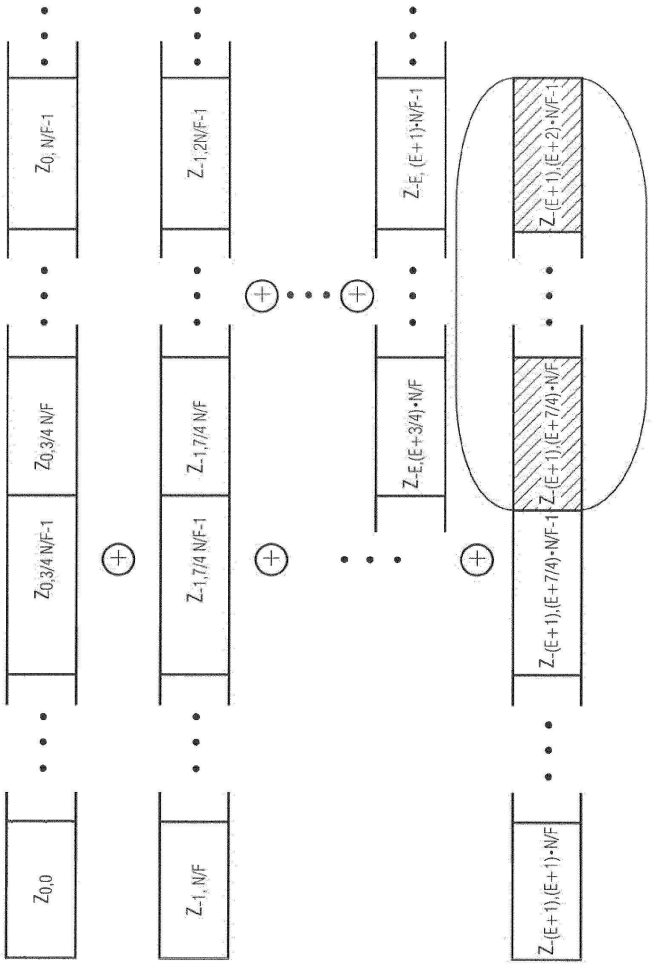
도면3


주파수
28
32
잡아냄
46
44
26
시간
30
0 0 0
28
40
42
34
시간
38
N
N
N
36
36
36
30
원래의 샘플링 속도
감소된 샘플링 속도
58
N/F-1
0
48
54
58
52
56
50
시간
N/F
36
N/F
36
N/F
36
62
60
시간
22
시간

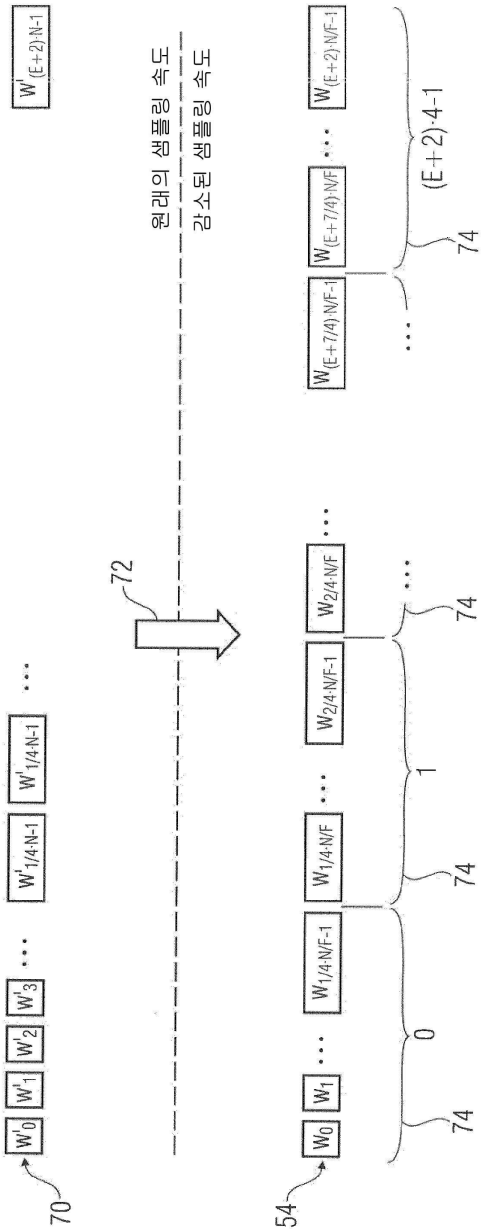
도면4



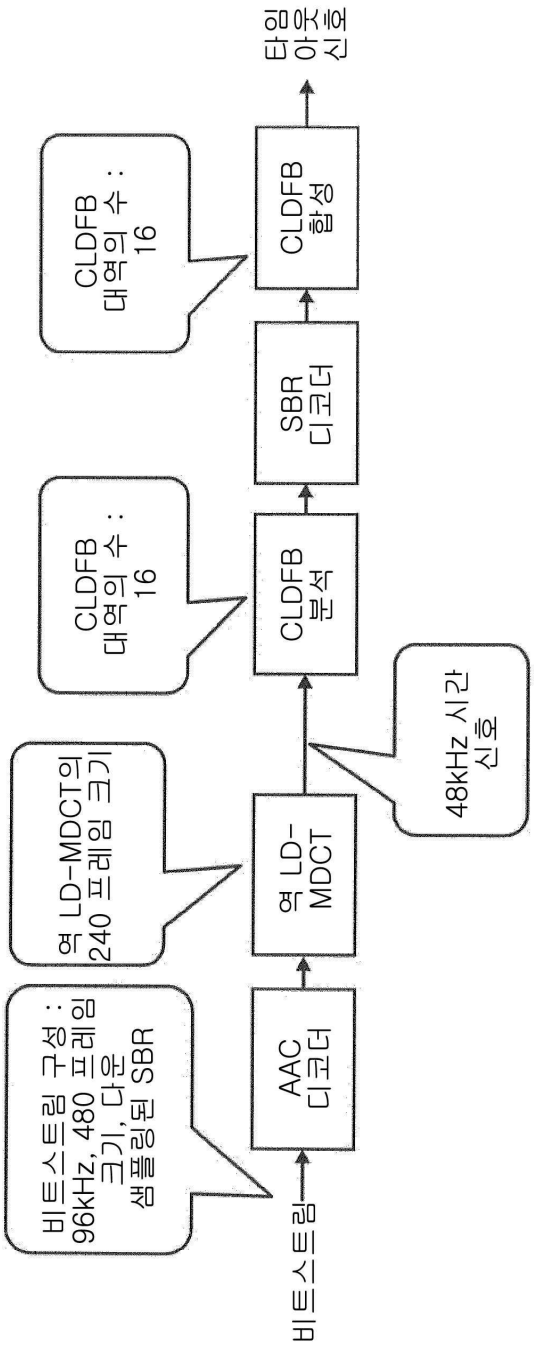
도면5


$Z_{0,0}$
$Z_{0,3/4\ N/F-1}$
$Z_{0,3/4\ N/F}$
$Z_{0,\ N/F-1}$
$Z_{-1,\ N/F}$
$Z_{-1,7/4\ N/F-1}$
$Z_{-1,7/4\ N/F}$
$Z_{-1,2N/F-1}$
$Z_{-E,(E+3/4)\cdot N/F}$
$Z_{-E,\ (E+1)\cdot N/F-1}$
$Z_{-(E+1),(E+1)\cdot N/F}$
$Z_{-(E+1),(E+7/4)\cdot N/F-1}$
$Z_{-(E+1),(E+7/4)\cdot N/F}$
$Z_{-(E+1),(E+2)\cdot N/F-1}$

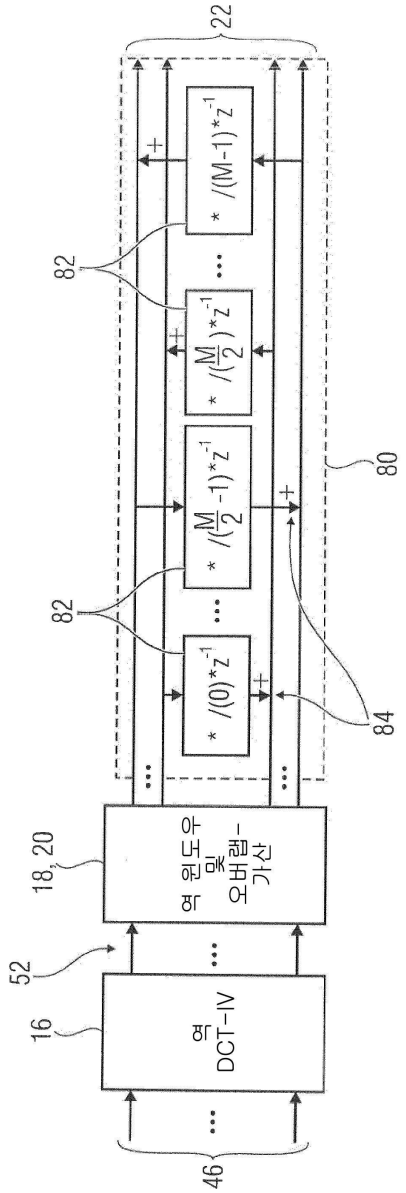
도면6



도면7


비트스트림
AAC 디코더
역 LD-MDCT
CLDFB 분석
SBR 디코더
CLDFB 합성
타임 아웃 신호
비트스트림 구성 : 96kHz, 480 프레임 크기, 다운 샘플링된 SBR
역 LD-MDCT의 240 프레임 크기
CLDFB 대역의 수 : 16
CLDFB 대역의 수 : 16
48kHz 시간 신호

도면8


22
82
82
80
84
18, 20
52
16
46
* /(M-1)*z-1
* /(M/2)*z-1
* /(M/2-1)*z-1
* /(0)*z-1
+
...
행 윈도우 및 오버랩-가산
행 DCT-IV

도면9