(12) 특허협력조약에 의하여 공개된 국제출원

(19) 세계지식재산권기구
국제사무국

(43) 국제공개일
2010 년 8 월 5 일 (05.08.2010)

PCT

(10) 국제공개번호
WO 2010/087614 A2

(51) 국제특허분류:
G10L 19/00 (2006.01) *H03M 13/00* (2006.01)
G11B 20/10 (2006.01)

(21) 국제출원번호: PCT/KR2010/000495

(22) 국제출원일: 2010 년 1 월 27 일 (27.01.2010)

(25) 출원언어: 한국어

(26) 공개언어: 한국어

(30) 우선권정보:
10-2009-0006668 2009 년 1 월 28 일 (28.01.2009) KR
10-2010-0007067 2010 년 1 월 26 일 (26.01.2010) KR

(71) 출원인 (US 을(를) 제외한 모든 지정국에 대하여): **삼성전자주식회사 (SAMSUNG ELECTRONICS CO., LTD.)** [KR/KR]; 경기도 수원시 영통구 매탄동 416, 443-742 Gyeonggi-do (KR).

(72) 발명자; 겸
(75) 발명자/출원인 (US 에 한하여): **주기현 (CHOO, Ki Hyun)** [KR/KR]; 경기도 수원시 영통구 매탄동 416 삼성전자주식회사 내, 443-742 Gyeonggi-do (KR). **김중회 (KIM, Jung-Hoe)** [KR/KR]; 경기도 수원시 영통구 매탄동 416 삼성전자주식회사 내, 443-742 Gyeonggi-do (KR).

(74) 대리인: **특허법인 무한 (MUHANN PATENT & LAW FIRM)**; 서울시 강남구 논현동 51-8 명림빌딩 2,5,6 층, 135-814 Seoul (KR).

(81) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 국내 권리의 보호를 위하여): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

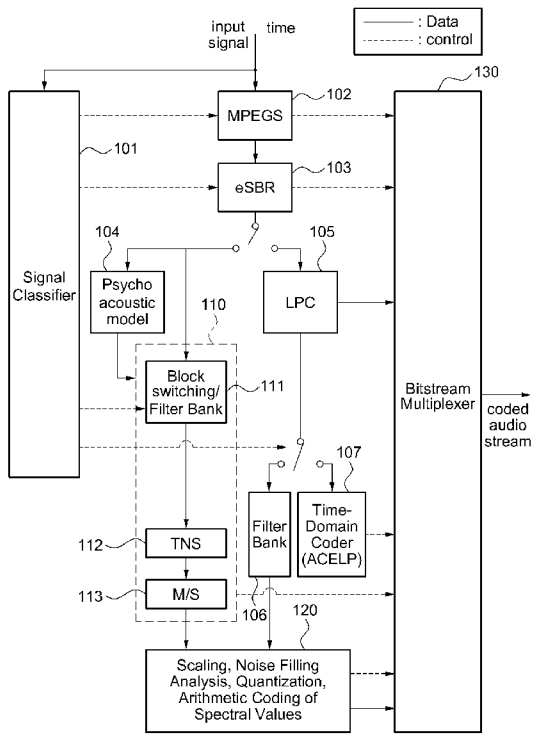
(84) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 역내 권리의 보호를 위하여): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), 유라시아 (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), 유럽 (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT,

[다음 쪽 계속]

(54) **Title**: METHOD FOR ENCODING AND DECODING AN AUDIO SIGNAL AND APPARATUS FOR SAME

(54) **발명의 명칭** : 오디오 신호의 부호화 및 복호화 방법 및 그 장치

[Fig. 1]



(57) **Abstract**: The present invention relates to a method for encoding and decoding an audio signal or a speech signal and to an apparatus using the method.

(57) **요약서**: 오디오 신호 또는 스피치 신호에 대한 부호화 및 복호화를 수행하는 방법 및 상기 방법이 채용된 장치가 개시된다.

WO 2010/087614 A2

NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

공개:

— 국제조사보고서 없이 공개하며 보고서 접수 후 이를 별도 공개함 (규칙 48.2(g))

명세서

발명의 명칭: 오디오 신호의 부호화 및 복호화 방법 및 그 장치

기술분야

[1] 오디오 신호 또는 스피치 신호의 부호화 및 복호화 방법 및 이를 수행하는 장치가 개시된다.

배경기술

[2] 오디오(audio) 신호는 주로 주파수 도메인에서 부호화 및 복호화를 수행한다. 대표적인 예로 AAC(Advanced Audio Coding)를 들 수 있다. AAC 코덱이 그 예인데, AAC 코덱은 주파수 도메인으로 변환하기 위한 MDCT(Modified Discrete Cosine Transform)을 수행하고, 심리음향 관점에서 신호의 마스킹 정도를 이용하여 주파수 스펙트럼 양자화를 수행한다. 수행된 양자화 결과를 더 압축하기 위해서 무손실 압축 방식을 적용하는데, AAC에서는 허프만 코딩(Huffman coding)을 사용한다. 무손실 압축 방식으로 허프만 코딩 방식 대신에 산술 코딩(Arithmetic coding)을 적용한 BSAC(Bit-Sliced Arithmetic Coding) 코덱도 사용될 수 있다.

[3]

[4] 음성(speech) 신호는 주로 시간 도메인에서 부호화 및 복호화를 수행한다. 시간 도메인에서 압축하는 음성 코덱의 대부분은 CELP 계열이다. CELP(Code Excitation Linear Prediction)는 음성 부호화 기술로 현재 거의 주로 사용되는 G.729, AMR-NB, AMR-WB, iLBC EVRC 등이 CELP 기반의 음성 부호화기이다. 이러한 코딩 기법은 음성 신호가 선형예측 (linear prediction)을 통해 얻을 수 있다는 가정 하에서 개발된다. 음성을 부호화 하는데 있어서, 선형예측 계수와 여기 신호(excitation signal)가 필요하다. 일반적으로 선형 예측 계수는 LSP를 이용하여 부호화 되고, 여기 신호는 몇 개의 codebook을 이용하여 부호화될 수 있다. CELP를 기반으로 하는 부호화 기법으로 ACELP, CS-CELP 등이 있다.

[5]

[6] 전송율에 대한 제약과 심리 음향 관점에서 저주파수 대역과 고주파수 대역의 민감도의 차이가 존재한다. 저주파수 대역은 음성/음악의 주파수 상의 미세 구조에 민감하고, 고주파수 대역은 미세 구조에 저주파수 대역에 비해 덜 민감하다는 사실에 근거하여, 저주파수 대역은 많은 비트를 할당하여 미세 구조를 자세히 코딩하고, 고주파수 대역은 적은 비트를 할당하여 코딩하는 방식이 적용되고 있다. 이러한 기술은 저주파수 대역은 AAC와 같은 코덱을 사용하여 미세 구조를 자세히 코딩하는 방식을 적용하고, 고주파수는 에너지 정보 및 조절 정보로 표현하는 방식이 SBR (Spectral Band Replication) 기술이다. SBR은 QMF(Quadrature Mirror Filter) 도메인에서 저주파수 신호를 복사하여 고주파수 신호를 생성하는 방식이다.

[7]

[8] 사용 비트를 줄이는 방식은 스테레오 신호에 대해서도 적용된다. 더욱 상세하게는, 스테레오 신호를 모노 신호로 변환한 후에 스테레오 정보를 표현하는 파라미터를 추출하여, 스테레오 파라미터 및 모노 신호를 압축한 데이터를 전송하여 전송된 파라미터를 이용해서 복호화기에서 스테레오 신호를 복호화할 수 있다. 이런 스테레오 정보를 압축 하는 방식으로 Parametric Stereo (PS) 기술이 있고, 스테레오뿐만 아니라 멀티 채널 신호의 파라미터를 추출하여 전송하는 방식으로 MPEG surround 기술이 있다.

[9]

[10] 또한, 위에서 설명한 무손실 부호화의 대상을 좀 더 구체적으로 살펴보면 양자화된 스펙트럼의 양자화 인덱스를 한 개의 심볼로 보고 무손실 부호화를 수행할 수 있다. 그리고 양자화된 스펙트럼의 인덱스를 비트플레인 상에서 매핑시켜서 비트를 묶는 방법으로 부호화 하는 경우도 있다.

[11] 컨텍스트(context) 기반 무손실 부호화를 수행하는 경우에 이전 프레임(frame)의 정보를 사용하여 무손실 부호화를 수행하는 것이 가능하다. 이 때, 복호화기가 이전 프레임에 대한 정보를 가지고 있지 않는 경우, 무손실 복호화기는 잘못된 복호화를 수행할 것이고, 극단적인 경우에는 시스템이 정지될 수도 있다. 예를 들어 오디오 부호화/복호화기를 사용하는 응용 방식의 하나인 디지털 오디오 방송의 경우에 있어서, 청취자는 임의의 시간에 라디오를 켜고 청취를 시작할 수 있고, 이러한 오디오 스트림에 대한 임의 접속(random access) 시에, 복호화기는 정확한 복호화를 위해 이전 프레임에 대한 정보를 사용하여야 하는데, 이전 프레임에 대한 정보가 없기 때문에 재생이 어렵다.

발명의 상세한 설명

과제 해결 수단

[12] 본 발명의 일실시예에 따르면, 입력 신호 중 저주파수 대역은 주파수 도메인 부호화 또는 선형 예측 도메인 부호화 중 선택하여 부호화하고, 선형 예측 도메인 부호화는 ACELP 또는 TCX로 부호화하고, 고주파수 대역은 향상된 SBR 툴로 부호화 하는 음성 오디오 부호화기가 제공된다.

발명의 효과

[13] 본 발명의 일측에 따르면, 오디오 신호 또는 스피치 신호의 부호화 시에 복호화를 위한 리던던트(redundant)한 비트 정보를 줄일 수 있다.

[14] 또한, 본 발명의 또 다른 일측에 따르면, 복호화를 위한 특정 비트 정보를 오디오 신호 또는 스피치 신호 복호화 초기에 참조할 수 있도록 함으로써, 복호화기의 연산 동작을 줄일 수 있다.

[15] 또한, 본 발명의 또 다른 일측에 따르면, 오디오 신호 또는 스피치 신호에 대한 임의 접속 시에도 디코딩이 가능하다.

도면의 간단한 설명

[16] 도 1은 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 부호화기를 도시한 블록도이다.

[17] 도 2는 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 부호화기에서 수행되는 부호화 방법의 일례가 도시된 흐름도이다.

[18] 도 3은 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 복호화기를 도시한 블록도이다.

[19] 도 4는 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 복호화기에서 수행되는 복호화 방법의 일례가 도시된 흐름도이다.

[20] 도 5는 본 발명의 또 다른 일실시예에 따른 복호화 방식을 설명하기 위한 부호화된 오디오 신호 또는 스피치 신호의 비트스트림을 예시한 도면이다.

[21] 도 6은 본 발명의 또 다른 일실시예에 따른, 부호화된 오디오 신호 또는 스피치 신호를 복호화하는 방법이 도시된 흐름도이다.

[22] 도 7은 본 발명의 일실시예에 따른 복호화기에서 수행되는, 주파수 도메인 복호화 또는 선형 예측 도메인 복호화 판단에 따른 복호화 방법이 도시된 흐름도이다.

[23] 도 8은 본 발명의 일실시예에 따른 복호화기에서 수행되는 코어 복호화 방법의 흐름도가 도시되어 있다.

발명의 실시를 위한 형태

[24] 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 부호화 및 복호화 방법에 따르면, 3GPP에서 표준화된 EAAC+(Enhanced AAC Pluse) 코덱 및 AMR-WB+ 코덱의 일부 툴들의 조합을 통해 코덱을 만들 수 있다.

[25]

[26] 도 1은 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 부호화기를 도시한 블록도이다.

[27] EAAC+는 AAC 코덱과 SBR, PS 기술을 기반으로 표준화 되었다. 여기에서, AMR-WB+는 AMR-WB 기반의 CELP 코덱과 TCX(Transform Coded eXcitation) 기술을 저주파수 대역을 압축하는데 적용되고, 고주파수 대역은 BandWidth Extension(BWE) 기술로 압축되며, 선형 예측 기반의 스테레오 기술을 적용하여 압축될 수 있다.

[28] 도 1에 도시된 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 부호화기에서, 저주파수 대역의 신호는 코어(core) 부호화기로 코딩되고, 고주파수 대역의 신호는 enhanced SBR(eSBR)(103)을 이용하여 부호화 되며, 스테레오 부분은 MPEG surround(MPEGS)(102)로 부호화될 수 있다.

[29] 저주파수 대역 신호의 부호화를 수행하는 코어(core) 부호화기는 주파수 도메인 코딩(frequency domain coding; FD)과 선형 예측 도메인 코딩(LP domain coding, Linear Prediction, LPD)의 2가지 부호화 모드로 동작할 수 있다. 이 중 선형 예측 도메인 코딩은 ACELP(Algebraic Code Excitation Linear Prediction) 와

TCX(Transform Coded Excitation)의 2가지 코딩 모드로 구성될 수 있다.

[30] 저주파수 대역 신호의 부호화를 수행하는 코어(core) 부호화기(102, 103)는 신호 분류기(Signal Classifier)(101)를 통하여 신호에 따라 주파수 도메인 부호화기(110)를 사용하여 부호화 할 지 아니면, 선형 예측 부호화기(105)를 사용하여 부호화 할지를 선택할 수 있다. 예를 들어, 음악 신호와 같은 오디오 신호는 주파수 도메인 부호화기(110)에서 부호화하는 것으로 스위칭하고, 스피치(음성) 신호는 선형 예측 도메인 부호화기(105)에서 부호화하는 것으로 스위칭할 수 있다. 스위칭된 부호화 모드 정보는 비트스트림에 저장된다. 주파수 도메인 부호화기(110)로 스위칭된 경우에는 주파수 도메인 부호화기(110)를 통해 부호화를 수행한다.

[31]

[32] 코어 부호화기의 동작은 아래의 구문(syntax)으로 표현될 수 있다.

[33]

[34]

Syntax	No. of bits
single_channel_element() { Core_mode	1

if (core_mode == 1) {
  lpd_channel_stream();
  }
  else {
  fd_channel_stream(0,0,noiseFilling);
  }
}	

[35]

[36] 주파수 도메인 부호화기(110)는 블록 스위칭/필터 뱅크 모듈(111)에서 신호에 적합한 윈도우 길이에 따라 변환(transform)을 수행한다. 상기 변환에는 MDCT(Modified Discrete Cosine Transform)가 사용될 수 있다. MDCT는 critically sampled transform으로서, 50% 오버랩을 수행하며 변환을 수행하고 윈도우 길이의 절반 길이에 해당되는 주파수 계수를 생성한다. 예를 들어 주파수 도메인 부호화기(110)에서 사용되는 1개 프레임의 길이는 1024이고, 1024의 2배 길이 2048 샘플 길이의 윈도우를 사용할 수 있다. 또한 1024 샘플을 8개로 나누어 256 윈도우 길이의 MDCT를 8번 수행할 수도 있다. 또한 코어 부호화 모드의 변환에 따라서는 2304 윈도우 길이를 사용해서 1152 주파수 계수를 생성할 수 있다.

[37]

[38] 변환된 주파수 도메인 데이터는 필요에 따라 TNS(Temporal Noise Shaping)(112)가 적용될 수 있다. TNS(112)는 주파수 도메인에서 선형 예측을 수행하는 방식으로서, 시간 특성과 주파수 특성의 Duality 관계에 의해서 attack이 강한 신호에 주로 사용될 수 있다. 예를 들어 시간 도메인에서 attack이 강한 신호는 상대적으로 주파수 도메인에서 flat한 신호로 표현될 수 있고, 이러한 신호를 선형 예측을 수행하면 코딩 효율을 높일 수 있다.

[39]

[40] TNS(112)로 처리된 신호가 스테레오일 경우에 Mid Side(M/S) 스테레오 코딩(113)이 적용될 수 있다. 스테레오 신호를 Left와 Right 신호로 그대로 코딩을 수행하면 압축 효율이 떨어지는 경우가 있는데, 이러한 경우에는 Left와 Right 신호의 합과 차로 표현함으로써, 신호를 압축 효율이 높은 신호로 변환하여 코딩할 수 있다.

[41]

[42] 주파수 변환, TNS, M/S가 적용된 신호는 양자화(quantization)를 수행하는데, 양자화는 통상 스칼라 양자화기가 사용될 수 있다. 이 때 스칼라 양자화를 주파수 전 대역에 대해서 동일하게 적용하면, 양자화된 결과의 dynamic range가 너무 크기 때문에 양자화 특성이 열화될 가능성이 있다. 이를 방지하기 위해서 주파수 대역을 심리 음향 모델(104)에 근거하여 분할하는데, 이를 스케일 팩터 밴드(scale factor band)라고 정의한다. 스케일 팩터 밴드 각각에 대해 스케일링 정보를 보내주며, 심리 음향 모델(104)에 근거하여 비트 사용량을 고려하여 스케일링 팩터를 계산하면서 양자화를 수행할 수 있다. 양자화 된 데이터 중에 0으로 양자화 된 경우는 복호화를 수행하더라도 0으로 표현된다. 0으로 양자화된 데이터가 많을수록 복호화된 신호의 왜곡이 생길 가능성이 높아지며, 이를 방지하기 위해서 복호화 시에 노이즈를 부가해 주는 기능이 수행될 수 있다. 이를 위해서 부호화기에서는 노이즈에 대한 정보를 생성해서 전송할 수 있다.

[43]

[44] 양자화된 데이터는 무손실 부호화를 수행하는데, 무손실 부호화기(120)로는 context arithmetic coding이 사용될 수 있으며, 이전 프레임의 스펙트럼 정보와 현재까지 복호화된 스펙트럼 정보를 context로 사용하여 무손실 부호화를 수행한다. 무손실 부호화된 스펙트럼 정보는 이전에 계산된 스케일링 팩터 정보, 노이즈 정보, TNS 정보, M/S 정보 등과 같이 비트스트림에 저장된다.

[45]

[46] 코어 부호화기에서 선형 예측 도메인 부호화기(105)로 스위칭된 경우, 하나의 수퍼프레임을 복수 개의 프레임으로 분할하여 각 프레임의 부호화 모드를 ACELP(107) 혹은 TCX(106)로 선택하여 부호화가 수행될 수 있다. 예를 들어 1개의 수퍼프레임은 1024 샘플로, 1개의 수퍼프레임은 4개의 프레임 256 샘플로 구성할 수 있다. 주파수 도메인 부호화기(110)의 1개의 프레임과 선형 예측 도메인 부호화기(105)의 1개의 수퍼프레임은 동일한 길이로 구성할 수 있다.

[47]

[48] ACELP와 TCX 중 부호화 모드를 선택하는 방식은 ACELP TCX의 부호화를 해본 후 SNR과 같은 측정 방식을 통해 선택하는 폐루프(closed loop) 방식이 있을 수 있고, 신호의 특성을 파악하여 결정하는 개루프(open loop) 방식이 있을 수 있다.

[49] 아래는 기존 기술에 따른 선형 예측 도메인 부호화기(105)의 동작을 표현한 구문(syntax)의 예가 도시되어 있다.

[50]

[51]

```
Syntax                                                          No. of bits
lpd_channel_stream()
{
    acelp_core_mode                                             3
    lpd_mode                                                    5
    First_tcx_flag=TRUE;
    k = 0;
    if (first_lpd_flag) {   last_lpd_mode = 0;    }
    while (k < 4) {
        if (mod[k] == 0) {
            acelp_coding(acelp_core_mode);
            last_lpd_mode=0;
            k += 1;
        }
        else {
            tcx_coding( lg(mod[k], last_lpd_mode) , first_tcx_flag);
            last_lpd_mode=mod[k];
            k += 2^(mod[k]-1);
            first_tcx_flag=FALSE;
        }
    }

    lpc_data(first_lpd_flag)
}
```

[52] ACELP의 경우에는 고정된 몇 개의 비트 할당 모드 중 선택하여 부호화하고, 이 경우 ACELP의 비트 할당 정보가 비트스트림에 포함된다. TCX의 경우는 몇 개의 고정된 비트 할당 모드 중 선택하는 방식이 아닌 가변 비트 할당 방식으로 부호화 되기 때문에, 비트 할당 정보를 필요로 하지 않는다.

[53]

[54] 이에 대해 상세히 설명하면 아래와 같다.

[55] TCX는 선형 예측 되고, 남은 여기(excitation) 신호에 대해 주파수 도메인으로 변환하여 주파수 도메인에서 압축을 수행한다. 주파수 도메인으로 변환하는 방식은 MDCT가 사용될 수 있다. 선형 예측 부호화 기술은 4개의 프레임으로 구성된 1개의 수퍼프레임 단위로 압축이 수행된다. 4개의 프레임에 대해서, 각 프레임들이 ACELP 혹은 TCX 중 어떤 것을 사용하여 압축되었는지 대한 모드 정보를 전송하여야 하고, ACELP의 경우는 1개 frame에 대해서 항상 같은 비트 사용량을 갖도록 구성이 되기 때문에, 사용된 비트량에 대한 정보를 비트스트림에 포함하여 전송해야 한다. TCX는 변환(transform) 길이에 따라 3가지의 모드가 존재한다. 선형 예측 부호화 모드에서 ACELP는 비트 사용량에 따라 몇 가지의 모드로 구성된다.

[56]

[57] 아래의 표 1 및 표 2에는 위에서 설명한 ACELP 및 TCX 모드와 관련된 비트 할당 모드의 예가 도시되어 있다.

[58] 표 1

	meaning of bits in bit-field mode					remaining mod[] entries
lpd_mode	bit 4	bit 3	bit 2	bit 1	bit 0	
0..15	0	mod[3]	mod[2]	mod[1]	mod[0]	
16..19	1	0	0	mod[3]	mod[2]	mod[1]=2 mod[0]=2
20..23	1	0	1	mod[1]	mod[0]	mod[3]=2 mod[2]=2
24	1	1	0	0	0	mod[3]=2 mod[2]=2 mod[1]=2 mod[0]=2
25	1	1	0	0	1	mod[3]=3 mod[2]=3 mod[1]=3 mod[0]=3
26..31						reserved

[59] 표 2

value of mod[x]	coding mode in frame	bitstream element
0	ACELP	acelp_coding()
1	one frame of TCX	tcx_coding()
2	TCX covering half a superframe	tcx_coding()
3	TCX covering entire superframe	tcx_coding()

[60] 일례로, TCX의 경우는 TCX의 변환 길이에 따라서 3가지 모드가 존재하며, 표 1에 도시된 것과 같이, (1) 1개의 프레임을 TCX를 사용하여 부호화, (2) 수퍼프레임의 절반을 TCX로 부호화, (3) 1개의 수퍼프레임 전체를 TCX로 부호화 하는 3가지의 모드로 구성될 수 있다. 이 경우에 lpd_mode는 표 1과 같이 구성할 수 있다. 예를 들어 lpd_mode가 0인 경우는 bit 4~bit 0의 순서로 00000으로 표현되고, 이 경우는 표 1에 도시된 것과 같이 1개의 수퍼프레임 내의 모든 프레임이 ACELP 방식으로 부호화되었다는 것을 의미한다.

[61]

[62] 표 1에 도시된 26개의 lpd_mode중 ACELP가 사용되지 않는 경우는 5가지 경우로서, 15, 19, 23, 24, 25가 될 수 있다. 즉, lpd_mode가 15,19,23,24,25의 경우는 1개의 수퍼프레임 내에 ACELP가 전혀 사용되지 않고, TCX만 사용된 경우이다.

[63] 따라서, 현재 수퍼프레임에서 ACELP가 사용되지 않을 경우 acelp_core_mode는 전송할 필요가 없다. 이러한 경우에 ACELP가 1개의 수퍼프레임 내에 사용되지 않았다는 정보 또는 TCX만 사용되었다는 정보를 생성하는 단계를 포함하여, acelp_core_mode 정보를 추가적으로 분석할 지를 결정한다. 즉, 1개의 수퍼프레임이 TCX만 사용된 경우에는 ACELP의 비트 할당 정보를 비트스트림에 포함하지 않고, ACELP를 1개 이상의 프레임에서 사용하고 있는 경우에만, ACELP의 비트 할당 정보를 비트스트림에 포함시킬 수 있고 이를 통해 부호화된 오디오 신호 또는 스피치 신호의 크기를 줄일 수 있다.

[64]

[65] 아래에는 상술한 ACELP의 비트 할당 정보를 줄일 수 있는 선형 예측 도메인

부호화기의 동작을 설명하는 구문(syntax)의 예가 기재되어 있다.

[66]

[67]

```
Syntax                                                          No. of bits
lpd_channel_stream()
{
    lpd_mode                                                    5
    if( is_tcx_only(lpd_mode) == 0 )
        acelp_core_mode                                         3
    first_tcx_flag=TRUE;
    k = 0;
    if (first_lpd_flag) {   last_lpd_mode = 0;    }
    while (k < 4) {
        if (mod[k] == 0) {
            acelp_coding(acelp_core_mode);
            last_lpd_mode=0;
            k += 1;
        }
        else {
            tcx_coding( lg(mod[k], last_lpd_mode) , first_tcx_flag);
            last_lpd_mode=mod[k];
            k += 2^(mod[k]-1);
            first_tcx_flag=FALSE;
        }
    }

    lpc_data(first_lpd_flag)
}
```

[68]

[69] TCX로 선택된 경우 시간 도메인 신호를 선형 예측하여 여기 신호를 추출한다. 추출된 여기 신호를 transform을 수행하여 주파수 도메인으로 변환한다. 예를 들어 MDCT가 적용될 수 있다. 주파수 변환된 여기 신호는 1개의 global gain으로 정규화한 후 스칼라 양자화를 수행한다. 양자화된 인덱스 정보는 양자화된 데이터는 무손실 부호화를 수행하게 되는데, 무손실 부호화기로는 context arithmetic coding이 사용되며, 이전 프레임의 스펙트럼 정보와 현재까지 복호화된 스펙트럼 정보를 context로 사용하여 무손실 부호화를 수행하게 된다. 무손실 부호화된 스펙트럼 정보는 global gain 과 선형예측 계수 정보, 등과 같이 비트스트림에 저장된다. 저장된 비트스트림은 비트스트림 다중화기(130)를 통해 부호화된 오디오 스트림으로 출력된다.

[70]

[71] 도 2는 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 부호화기에서 수행되는 부호화 방법의 일례가 도시된 흐름도이다.

[72] 도 2를 참조하면, 입력된 저주파수 신호에 대해 선형 예측 도메인 부호화 방식으로 부호화할 지 여부를 판단한다(201). 단계(201) 판단 결과, 선형 예측 도메인 부호화 방식으로 부호화하는 경우, 선형 예측 도메인의 모드를 결정하고 모드 정보를 저장한다(203). TCX만 사용하여 부호화할지 여부를 결정하고(204), TCX만 사용하여 부호화하는 경우 선형 예측 도메인 부호화를 수행한다(206). 단계(204) 판단 결과, TCX만 사용하여 부호화하지 않는 경우에는 ACELP 비트 할당 정보를 저장하고 단계(206)의 선형 예측 도메인 부호화를 수행한다.

[73] 단계(201)에서 선형 예측 도메인 부호화 방식으로 부호화하지 않는 경우,

주파수 도메인 부호화를 수행한다(202).

[74]

[75] 도 3은 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 복호화기를 도시한 블록도이다.

[76] 도 3을 참조하면 본 발명의 일실시예에 따른 복호화기는 비트스트림 역다중화기(301), 산술 복호화부(302), 필터 뱅크(303), 시간 도메인 복호화부(ACELP)(304), 트랜지션 윈도우부(305, 307), LPC(306), Bass Postfilter(308), eSBR(309), MPEGS 복호화부(320), M/S(311), TNS(312), 블록 스위칭/필터 뱅크(313)를 포함한다. 본 발명의 일실시예에 따른 복호화기는 도 1에 도시된 부호화기 또는 도 2에 도시된 부호화 방법에 의해 부호화된 오디오 신호 또는 스피치 신호를 복호화한다.

[77] 도 3에 도시된 복호화기는 부호화 모드 정보를 바탕으로 주파수 도메인 부호화 모드로 부호화 된 경우는 주파수 도메인에서 복호화를 수행하고, 선형 예측 도메인 부호화 모드로 부호화 된 경우는 1개의 수퍼프레임 의 각 프레임에 대한 CELP 또는 TCX로 부호화 되었는지에 대한 정보를 바탕으로 각 부호화 모드에 맞게 복호화를 수행한다.

[78]

[79] 도 3에 도시된 본 발명의 일실시예에 따른 복호화기에서 코어 복호화 방식은 아래의 구문(syntax)으로 표현될 수 있다.

[80]

[81]

Syntax	No. of bits
single_channel_element() { Core_mode	1

if (core_mode == 1) {
  lpd_channel_stream();
  }
  else {
  fd_channel_stream(0,0,noiseFilling);
  }
}	

[82]

[83] 위의 구문으로 판단된 정보를 바탕으로, 주파수 도메인 부호화 방식이 적용된 경우에는 주파수 도메인 복호화를 수행한다. 주파수 도메인 복호화는 스케일 팩터 정보와 arithmetic coding으로 무손실 부호화된 양자화된 스펙트럼을 복원해주고, 양자화된 스펙트럼을 역양자화 하고, 스케일 팩터를 곱해서 스펙트럼을 생성한다. 생성된 스펙트럼을 바탕으로 TNS와 M/S 정보를 이용하여 스펙트럼을 변경하여 복원된 스펙트럼을 생성한다. 잡음 부가가 필요한 경우에는 잡음을 부가할 수 있다. 복원된 스펙트럼을 역변환을 통해 최종 코어 시간 도메인 신호를 생성한다. 역변환 시에는 MDCT를 사용할 수 있다.

[84]

[85] 이러한 주파수 도메인 복호화 방식은 아래의 구문(syntax)으로 표현될 수 있다.

[86]

[87]

Syntax	No. of bits
fd_channel_stream(common_window, common_tw, noiseFilling)	
{	
global_gain;	8
if (noiseFilling) {	
Noise_offset	3
Noise_level	5
}	
else {	
noise_level = 0	
}	
if (!common_window) {	
ics_info();	
}	
if (tw_mdct) {	
if (! common_tw) {	
Tw_data();	
}	
}	
scale_factor_data ();	
tns_data_present;	1
if (tns_data_present) {	
tns_data ();	
}	
ac_spectral_data ();	
}	

[88]

[89] 복호화기에서 부호화된 저주파수 대역 신호에 대한 코어 부호화 모드를 판단하는 방식은 아래 표현된 구문(syntax)과 같다.

[90]

[91]

Syntax	No. of bits
lpd_channel_stream()	
{	
lpd_mode	5
if(is_tcx_only(lpd_mode) == 0)	
acelp_core_mode	3
first_tcx_flag=TRUE;	
k = 0;	
if (first_lpd_flag) { last_lpd_mode = 0; }	
while (k < 4) {	
if (mod[k] == 0) {	
acelp_coding(acelp_core_mode);	
last_lpd_mode=0;	
k += 1;	
}	
else {	
tcx_coding(lg(mod[k], last_lpd_mode) , first_tcx_flag);	
last_lpd_mode=mod[k];	
k += 2^(mod[k]-1);	
first_tcx_flag=FALSE;	
}	
}	
lpc_data(first_lpd_flag)	
}	

[92]

[93] LPD의 부호화 모드 정보(lpd_mode)는 1개의 수퍼프레임이 ACELP와 TCX의 어떤 조합으로 구성이 되었는지에 대한 정보를 포함한다. TCX만 사용한 경우에는 ACELP의 비트 할당 정보(acelp_core_mode)를 비트스트림 내에 포함하지 않아도 ACELP 복호화를 수행하지 않기 때문에, 복호화가 가능하다. 따라서, ACELP가 1개의 수퍼프레임 내에 사용되지 않았다는 정보 또는 TCX만 사용되었다는 정보를 읽어서, ACELP의 비트 할당 정보(acelp_core_mode)를

추가적으로 분석할지 결정한다. TCX만 사용한 것으로 판단한 경우, 수퍼프레임에 대해 TCX 복호화를 수행한다. ACELP가 포함된 경우에는 ACELP의 비트 할당 정보(acelp_core_mode) 정보를 추가적으로 분석한 후 ACELP 혹은 TCX 복호화를 수행할 수 있다.

[94]

[95] 이와 같이, 부호화된 오디오 신호 또는 스피치 신호의 비트스트림을 분석하여, 선형 예측 도메인 부호화 방식으로 부호화 되었는지를 확인하고, 선형 예측 도메인 부호화로 부호화 되지 않은 경우에는 주파수 도메인 복호화를 수행하여, 복호화된 저주파수 신호를 생성한다. 선형 예측 도메인 부호화로 부호화 된 경우에는 선형 예측 도메인의 부호화 모드를 분석하고, 분석 결과 TCX만 사용이 된 경우에는 선형 예측 도메인 복호화를 수행하고, 1개 이상의 프레임이 ACELP 부호화를 사용하여 부호화 된 경우에는 ACELP의 비트 할당 정보를 분석한 후 선형 예측 복호화를 수행한다.

[96]

[97] 선형 예측 도메인으로 부호화 된 경우는 1개의 수퍼프레임 내에 4개의 프레임에 대한 ACELP 또는 TCX의 부호화 모드 정보를 바탕으로 각 모드에 맞도록 복호화를 수행한다. TCX의 경우는 산술 부호화(arithmetic coding) 된 스펙트럼을 생성하고 전송된 global gain을 곱하여 스펙트럼을 생성한다. 생성된 스펙트럼을 IMDCT로 역변환을 수행한다. 전송된 선형 예측 계수를 바탕으로 선형 예측 합성(LP synthesis)을 수행하여 코어 복호화된 신호를 생성한다. ACELP 모드의 경우는 Adaptive and innovation codebook의 인덱스(index)와 이득(gain) 정보를 바탕으로 여기(excitation) 신호를 생성하고, 생성된 여기 신호에 선형 예측 합성(LP synthesis)을 수행하여 코어 복호화된 신호를 생성한다.

[98]

[99] 도 4는 본 발명의 일실시예에 따른 오디오 신호 또는 스피치 신호의 복호화기에서 수행되는 복호화 방법의 일례가 도시된 흐름도이다.

[100] 도 4를 참조하면, 입력된 비트스트림에 대해 선형 예측 도메인 부호화 방식으로 부호화되었는지 여부를 판단한다(401). 단계(401) 판단 결과, 선형 예측 도메인 부호화 방식으로 부호화된 경우, 선형 예측 도메인의 모드를 분석한다(403). TCX만 사용하여 부호화되었는지 여부를 결정하고(404), TCX만 사용하여 부호화된 경우 선형 예측 도메인 복호화를 수행한다(406). 단계(404) 판단 결과, TCX만 사용하여 부호화되지 않는 경우에는 ACELP 비트 할당 정보를 분석하고 단계(406)의 선형 예측 도메인 복호화를 수행한다.

[101] 단계(401)에서 선형 예측 도메인 부호화 방식으로 부호화되지 않는 경우, 주파수 도메인 복호화를 수행한다(402).

[102]

[103] 이하, 도 5 내지 도 7을 참조하여, 본 발명의 또 다른 실시예에 따른 복호화기 및

복호화 방법을 상세히 설명한다.

[104]

[105] 도 5는 본 발명의 또 다른 일실시예에 따른 복호화 방식을 설명하기 위한 부호화된 오디오 신호 또는 스피치 신호의 비트스트림을 예시한 도면이다.

[106] 도 5를 참조하면, 컨텍스트 리셋 플래그(context reset flag)가 적용된 예가 도시되어 있다. 이러한 컨텍스트 리셋 플래그(context reset flag)는 AAC/TCX 엔트로피 코딩(entropy coding) 시 적용되고, 산술 코딩 컨텍스트(Arithmetic coding context) 초기화를 지시(indication)하기 위한 구문(syntax)이다. 상기 컨텍스트 리셋 플래그는 주기적으로 1로 세팅되어 컨텍스트 리셋이 수행된다. 기존의 AAC의 경우 프레임 별로 디코딩이 가능하므로 브로드캐스팅 시 임의의 시점에서 디코딩을 시작해도 디코딩될 수 있으나, MPEG USAC(Unified Speech and Audio Coding) 등의 경우에는 이전 프레임을 컨텍스트로 이용하므로 이전 프레임에 대한 디코딩이 안 되어 있으면 현재 프레임의 디코딩이 불가능하다.

[107] 따라서, 사용자가 부호화된 오디오 신호 또는 스피치 신호에 대해 재생을 원하는 임의의 위치로 임의 접속(random access) 하는 경우, 해당 위치의 프레임을 복호화할 때 해당 프레임의 컨텍스트 리셋 플래그가 어떻게 설정되어 있는지, 즉 현재 프레임이 디코딩 가능한지 여부를 판단하는 것이 필요하다.

[108]

[109] 도 6은 본 발명의 또 다른 일실시예에 따른, 부호화된 오디오 신호 또는 스피치 신호를 복호화하는 방법이 도시된 흐름도이다.

[110] 도 6을 참조하면, 입력되는 비트스트림에 대한 사용자의 복호화 시작 명령을 수신한다. 비트스트림에 대해 코어 비트스트림 복호화가 가능한지 여부를 판단하고(601), 코어 비트스트림 복호화가 가능한 경우 코어 복호화를 수행한다(603). eSBR 비트스트림 분석 및 복호화를 수행하고(604), MPEGS 분석 및 복호화를 수행한다(605). 단계(601)에서 코어 비트스트림 복호화가 불가능한 경우, 현재 프레임에 대한 복호화를 종료하고(602), 다음 프레임에 대해 코어 복호화 가능 여부를 판단한다. 이와 같이, 코어 복호화 가능 여부에 대한 확인을 수행하다가 복호화 가능한 프레임이 발견되면 해당 프레임부터 복호화를 수행할 수 있다. 코어 복호화가 가능한지의 여부는 이전 프레임의 정보를 참조할 수 있는지 여부로 확인할 수 있고, 이전 프레임의 정보를 참조할 수 있는지의 여부는 산술 부호화의 컨텍스트 정보를 초기화할지 여부, 즉 arith_reset_flag 정보를 읽어서 판단할 수 있다.

[111]

[112] 본 발명의 또 다른 일실시예에 따른 복호화 방법은 아래 도시된 구문(syntax)으로 표현될 수 있다.

[113]

Syntax	No. of bits

single_channel_element()
{
core_mode
if (core_mode == 1) {
lpd_channel_stream();
}
else {
fd_channel_stream(0,0,noiseFilling);
}
}
	1

Syntax	No. of bits

channel_pair_element()
{
core_mode0
core_mode1
if (core_mode0 == 0 && core_mode1 == 0) {
common_window;
if (common_window) {
ics_info();
ms_mask_present;
if (ms_mask_present == 1) {
for (g = 0; g < num_window_groups; g++) {
for (sfb = 0; sfb < max_sfb; sfb++) {
ms_used[g][sfb];
}
}
}
}
if (tw_mdct) {
common_tw;
If (common_tw) {
tw_data();
}
}
}
else {
common_window = 0;
common_tw = 0;
}
if (core_mode0 == 1) {
lpd_channel_stream();
}
else {
fd_channel_stream(common_window, common_tw,
noiseFilling);
}
if (core_mode1 == 1) {
lpd_channel_stream();
}
else {
fd_channel_stream(common_window, common_tw,
noiseFilling);
}
}
	1
	1
	1
	2
	1
	1

[114]

[115] 위의 구문을 참조하면, 1개의 채널 정보만 비트스트림에 포함된 경우는 single_channel_element를 사용하여 복호화를 수행하고, 2개의 채널 정보를 동시에 포함하여 복호화를 수행하는 경우는 channel_pair_element를 사용하여 복호화를 수행한다. 코어 부호화 모드(core_mode)를 분석하여 주파수 도메인인지, 선형 예측 도메인인지의 여부를 판단한다. Channel_pair_element의 경우는 2개 채널에 대한 정보를 포함하기 때문에, 코어 부호화 모드 정보가 2개 존재한다.

[116]

[117] 상기 판단을 근거로 하여, 주파수 도메인 부호화된 경우는 context를 초기화

하여 복호화를 수행할지에 대한 정보를 맨 먼저 분석한 다음 주파수 도메인 복호화를 수행할 수 있다.

[118] 이와 같은 방식을 표현한 구문(syntax)의 일례가 아래에 도시되어 있다.

[119]

[120]

Syntax	No. of bits
fd_channel_stream(common_window, common_tw, noiseFilling)	
{	
arith_reset_flag	**1**
global_gain;	**8**
if (noiseFilling) {	
noise_offset	**3**
noise_level	**5**
}	
else {	
noise_level = 0	
}	
if (!common_window) {	
ics_info();	
}	
if (tw_mdct) {	
if (! common_tw) {	
tw_data();	
}	
}	
scale_factor_data ();	
tns_data_present;	**1**
if (tns_data_present) {	
tns_data ();	
}	
ac_spectral_data (arith_reset_flag);	
}	
ac_spectral_data(arith_reset_flag)	
{	
for (win = 0; win < num_windows; win++) {	
arith_data(num_bands, arith_reset_flag)	
}	
}	

[121]

[122] 위의 구문을 참조하면, arith_reset_flag가 상기 구문 앞에 위치해 있다. arith_reset_flag가 설정된 경우, 즉 컨텍스트 초기화가 수행되는 경우 주파수 도메인 복호화를 수행하고, 컨텍스트 초기화가 수행되지 않는 경우에는 현재 프레임에 대한 복호화를 수행하지 않을 수 있다.

[123]

[124] 도 7은 본 발명의 일실시예에 따른 복호화기에서 수행되는, 주파수 도메인 복호화 또는 선형 예측 도메인 복호화 판단에 따른 복호화 방법이 도시된 흐름도이다.

[125] 도 7을 참조하면, 입력되는 부호화된 비트스트림에 대해 주파수 도메인 복호화를 수행하는지 아니면 선형 예측 도메인 복호화를 수행하는지를 판단하고(701), 판단 결과에 따라 주파수 도메인 또는 선형 예측 도메인으로 각각 복호화가 수행되는 방법이 도시되어 있다.

[126] 단계(701)의 판단 결과, 주파수 도메인 복호화가 수행되는 경우, 상술한 바와 같이 컨텍스트 초기화 여부를 먼저 판단하고(702), 컨텍스트 초기화가

수행되어야 하는 경우에는 현재 프레임부터 복호화를 시작한다(703). 컨텍스트 초기화가 수행되지 않는 경우에는 현재 프레임에 대한 복호화를 종료하고(704) 다음 프레임에 대한 판단을 수행한다.

[127] 선형 예측 도메인 복호화가 수행되는 경우, 즉 비트스트림이 선형 예측 도메인 부호화된 경우에는, 수퍼프레임 내의 선형 예측 부호화 모드를 판단하고(705), 판단된 근거를 바탕으로 적어도 1개 이상의 TCX가 사용되었는지의 여부를 판단하여(706) TCX가 사용된 경우에는 해당 컨텍스트를 초기화하는지 여부를 분석한다(707). 단계(707)의 판단 결과, 컨텍스트 초기화가 수행되어야 하는 경우에는 현재 프레임부터 복호화를 시작한다(708). 컨텍스트 초기화가 수행되지 않는 경우에는 현재 프레임에 대한 복호화를 종료하고(704) 다음 프레임에 대한 판단을 수행한다.

[128] 상술한 바와 같이, 선형 예측 부호화 모드가 0인 경우는 ACELP만 사용된 경우이기 때문에, 이를 제외한 다른 lpd_mode 일 경우에는 TCX가 적어도 1개 이상은 사용되었다는 사실을 알 수 있다. 따라서, TCX가 적어도 1개 이상이 사용된 경우에는 컨텍스트를 초기화할지 아니면 초기화하지 않을지에 대한 정보가 부호화된 비트스트림에 포함된다. 여기에서, 이전 프레임의 context를 사용하지 않는 ACELP의 경우는 위에서 설명한 산술 부호화의 context 정보를 초기화할지 여부에 대한 판단과 무관하게 복호화를 수행할 수 있다.

[129] 본 발명의 일실시예에 따른 복호화기에서는 상기 TCX가 사용된 프레임에 포함되는, 컨텍스트 초기화 여부에 대한 arith_reset_flag를 이용하여 복호화기의 연산 효율을 극대화할 수 있다.

[130] 본 발명의 일실시예에 따른 복호화기에서 입력된 비트스트림을 분석하여 선형 예측 도메인 복호화를 수행하는 경우, 아래의 구문(syntax)으로 표시된 방법에 따라 해당 비트스트림을 복호화할 수 있다.

[131]

[132]

Syntax	No. of bits

lpd_channel_stream()
{
acelp_core_mode
lpd_mode
If (lpd_mode != 0)
arith_reset_flag
k = 0;
if (first_lpd_flag) { last_lpd_mode = 0; }
while (k < 4) {
if (mod[k] == 0) {
acelp_coding(acelp_core_mode);
last_lpd_mode=0;
k += 1;
}
else {
tcx_coding(lg(mod[k], last_lpd_mode) , arith_reset_flag);
last_lpd_mode=mod[k];
k += 2^(mod[k]-1);
arith_reset_flag = FALSE;
}
}
 	
lpc_data(first_lpd_flag)
}
	
	3
	5
	
	1

tcx_coding(lg, arith_reset_flagfirst_tcx_flag)
{
noise_factor
global_gain
 	
if (first_tcx_flag) {
arith_reset_flag
}
else {
arith_reset_flag=0
}
arith_data(lg, arith_reset_flag)
}
	
	3
	7
	
	1

[133]

[134] 종합하면, 본 발명의 일실시예에 따른 복호화 방식 채용 전의 주파수 도메인 복호화의 경우, 스펙트럴 데이터(spectral data) 이전까지 모든 데이터를 파싱(parsing)하여야 컨텍스트 초기화 여부를 판단할 수 있고, TCX의 경우 ACELP/TCX 조합에 따라 모두 복호화하여야 컨텍스트 초기화 여부를 판단할 수 있었다. 이는 컨텍스트 초기화 여부 판단을 위해서는 스케일 팩터(scale factor) 등 정보가 필요하기 때문이다. 그러나, 본 발명의 일실시예에 따른 복호화 방식에 따르면, AAC의 경우 첫 번째 비트가 arith_reset_flag가 되고, 선형 예측 모드인 경우, 처음 6 비트를 파싱하여 arith_reset_flag 값을 읽으면 컨텍스트 초기화 여부를 판단할 수 있으므로, 모든 프레임에 대한 복호화 없이도 해당 프레임에 대한 복호화 가능 여부를 판단할 수 있고, 이를 통해 복호화기의 연산 효율을 극대화할 수 있다.

[135]

[136] 본 발명의 또 다른 일실시예에 따른 부호화기는 부호화 수행 시, 현재 프레임이 Random access가 가능한지의 여부에 대한 정보를 더 포함시켜 부호화할 수 있다.

[137] 도 8은 본 발명의 일실시예에 따른 복호화기에서 수행되는 코어 복호화 방법의 흐름도가 도시되어 있다.

[138] 도 8을 참조하면, 임의의 시간에, 사용자가 부호화된 오디오 신호 또는 스피치 신호에 대한 복호화를 실행하고자 하는 경우, 부호화된 비트스트림을 입력 받아

현재 프레임이 복호화가 가능한지 여부를 임의 접속(random access)이 가능한 프레임인지 확인한 후(801) 코어 복호화를 수행한다(803). 코어 복호화가 불가능한 경우는 다음 프레임에 대해 복호화 가능 여부를 판단한다. 이와 같이, 현재 프레임의 복호화 가능 여부에 대한 확인을 수행하다가 복호화 가능한 프레임이 발견되면 해당 프레임부터 복호화를 수행할 수 있다. 복호화가 가능한지의 여부는 현재 프레임이 임의 접속(random access)이 가능한 프레임인지 확인을 통해 판단할 수 있다.

[139] 단계(803)의 코어 복호화 이후, eSBR 비트스트림 분석 및 복호화(804), MPEGS 분석 및 복호화(805)를 수행하고, 복호화된 신호를 재생한다(806).

[140]

[141] 주파수 도메인 부호화의 경우, 윈도우 타입의 정보는 총 8가지인데, 8가지의 type을 이전 프레임의 코어 부호화 모드 및 이전 프레임의 윈도우 타입에 따라 2비트로 표현할 수 있다. 이러한 경우, 이전 프레임에 대한 코어 부호화 모드 정보가 없는 경우에는 복호화가 불가능하며, 복호화가 불가능하게 되는 것을 방지하기 위해서 해당 프레임이 random access가 가능한지 프레임인지에 대한 정보를 추가하고, 이를 이용해 window_sequence 정보를 다르게 표현할 수 있다.

[142] 상기 window_sequence는 스펙트럼의 개수 및 역변환을 수행하기 위해서 필요한 정보이다. random_access가 불가능한 프레임에 대해서는 window_sequence 정보를 기존 방식인 2비트로 표시를 하고, random_access가 가능한 프레임에 대해서는 window_sequence 정보를 3비트로 표현할 수 있다. 또한, random access가 가능한 프레임의 경우, 복호화를 위해 컨텍스트 초기화를 수행하여야 하므로, arith_reset_flag는 항상 1로 설정된다. 이러한 경우에는 arith_reset_flag를 따로 포함시켜 전송할 필요가 없다.

[143] 아래 구문(syntax)은 도 6을 참조하여 설명한 원 구문이다.

[144]

[145]

Syntax	No. of bits
single_channel_element() { core_mode	1

if (core_mode == 1) {
lpd_channel_stream();
}
else {
fd_channel_stream(0,0,noiseFilling);
}
}	

[146]

[147] 또한, 아래 구문(syntax)은 본 발명의 변형예에 따라 수정된 구문이다.

[148]

[149]

Syntax	No. of bits

single_channel_element()
{
random_access
core_mode
if (core_mode == 1) {
lpd_channel_stream(random_access)
}
else {
fd_channel_stream(random_access,0,0,
}
}
	
	1
	1

[150]

[151] 위의 구문(syntax)에 기재된 일례와 같이, random_access 필드(1 bit)를 추가할 수 있다.

[152] Randon_access 필드의 플래그 값이 설정된 경우, 복호화를 위한 window_sequence 정보를 다르게 표현할 수 있다.

[153] 이에 대한 구문(syntax)의 일례는 아래와 같다.

[154]

[155]

Syntax	No. of bits
channel_pair_element()	
{	
random_access	1
core_mode0	1
core_mode1	1
if (core_mode0 == 0 && core_mode1 == 0) {	
common_window;	1
if (common_window) {	
ics_info(random_access);	
ms_mask_present;	2
if (ms_mask_present == 1) {	
for (g = 0; g < num_window_groups; g++) {	
for (sfb = 0; sfb < max_sfb; sfb++) {	
ms_used[g][sfb];	1
}	
}	
}	
}	
if (tw_mdct) {	
Common_tw;	1
If (common_tw) {	
tw_data();	
}	
}	
}	
else {	
common_window = 0;	
common_tw = 0;	
}	
if (core_mode0 == 1) {	
lpd_channel_stream(random_access);	
}	
else {	
fd_channel_stream(random_access,common_window, common_tw, noiseFilling);	
}	
if (core_mode1 == 1) {	
lpd_channel_stream(random_access);	
}	
else {	
fd_channel_stream(random_access,common_window, common_tw, noiseFilling);	
}	
}	

[156]

[157] 상기 구문(syntax)에서, fd_channel_stream은 아래의 구문으로 표현될 수 있다.

[158]

Syntax	No. of bits
fd_channel_stream(random_access, common_window, common_tw, noiseFilling)	
{	
global_gain;	**8**
if (noiseFilling) {	
noise_offset	**3**
noise_level	**5**
}	
else {	
noise_level = 0	
}	
if (!common_window) {	
ics_info(random_access);	
}	
if (tw_mdct) {	
if (! common_tw) {	
tw_data();	
}	
}	
scale_factor_data ();	
tns_data_present;	**1**
if (tns_data_present) {	
tns_data ();	
}	
ac_spectral_data (random_access);	
}	

[159]

[160] 상기 구문(syntax)으로 표현된 fd_channel_stream에서, ics_info는 아래의 구문으로 표현될 수 있고, 아래 구문에서 window_sequence 정보에는 3 bit가 할당될 수 있다.

[161]

[162]

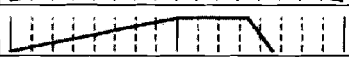
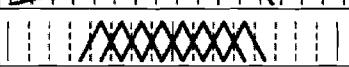
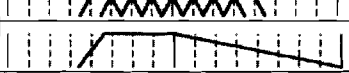
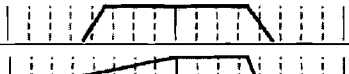
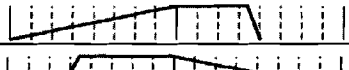
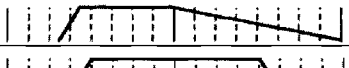
Syntax	No. of bits
ics_info(random_access)	
{	
If (random_access == TRUE)	
window_sequence;	**3**
else	
window_sequence;	**2**
window_shape;	**1**
if (window_sequence == EIGHT_SHORT_SEQUENCE) {	
max_sfb;	**4**
scale_factor_grouping;	**7**
}	
else {	
max_sfb;	**6**
}	
}	

[163]

[164] 상술한 바와 같이, 본 발명의 일실시예에 따른 부호화기에서 window_sequence 정보가 3 bit 할당되는 경우 window_sequence의 정의는 아래와 같다. 우선 본 발명의 일실시예에 따른 window_sequence의 기존 정의는 아래와 같다.

[165] <기존 window_sequence의 정의>

[166]

Value	window_sequence	num_windows	looks like
0	ONLY_LONG_SEQUENCE = LONG_WINDOW	1	
1	LONG_START_SEQUENCE = LONG_START_WINDOW	1	
2	EIGHT_SHORT_SEQUENCE = 8 * SHORT_WINDOW	8	
3	LONG_STOP_SEQUENCE = LONG_STOP_WINDOW	1	
1	STOP_START_SEQUENCE = STOP_START_WINDOW	1	
3	LPD_START_SEQUENCE = START_WINDOW_LPD	1	
3	STOP_1152_SEQUENCE = STOP_WINDOW_1152	1	
1	STOP_START_1152_SEQUENCE = STOP_START_WINDOW_1152	1	

[167]

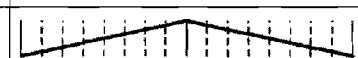
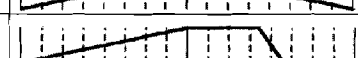
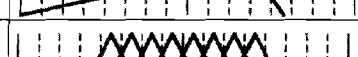
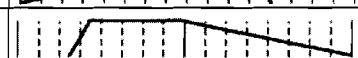
[168] 위의 정의에서, Value가 1일 경우를 살펴보면 총 3가지 경우가 있다. LONG_START_SEQUENCE, STOP_START_SEQUENCE, STOP_START_1152_SEQUENCE의 3가지 경우이다. 예를 들어 STOP_START_1152_SEQUENCE의 경우, 복호화하기 위해서는 이전 프레임이 LPD로 부호화되었다는 정보가 있어야 한다. Value가 3일 경우도 마찬가지로, 이전 프레임이 LPD로 부호화 되었다는 정보가 있어야 복호화 할 수 있다.

[169]

[170] 아래 구문은 본 발명의 일실시예에 따라 정의된 window_sequence 정보의 일례이다. 아래의 window_sequence 정보는 8개의 window_sequence 종류에 8개의 값을 순차적으로 할당한 것으로 다른 방식으로도 표현이 가능하다.

[171] <본 발명의 일실시예에 따른 window_sequence의 정의>

[172]

Value	window_sequence	num_windows	looks like
0	ONLY_LONG_SEQUENCE = LONG_WINDOW	1	
1	LONG_START_SEQUENCE = LONG_START_WINDOW	1	
2	EIGHT_SHORT_SEQUENCE = 8 * SHORT_WINDOW	8	
3	LONG_STOP_SEQUENCE = LONG_STOP_WINDOW	1	
4	STOP_START_SEQUENCE = STOP_START_WINDOW	1	
5	LPD_START_SEQUENCE = START_WINDOW_LPD	1	
6	STOP_1152_SEQUENCE = STOP_WINDOW_1152	1	
7	STOP_START_1152_SEQUENCE = STOP_START_WINDOW_1152	1	

[173]

[174] 또한, 위에서 설명한 바와 같이, random access 프레임의 경우, 복호화를 위해 컨텍스트 초기화를 수행하여야 하므로, arith_reset_flag는 항상 1로 설정된다. 이러한 경우에는 arith_reset_flag를 따로 포함시켜 전송할 필요가 없다. 따라서,

종래의 방식에 채용되었던 방식은 물론, 도 6을 참조하여 설명한 아래의 구문은 다음과 같이 수정될 수 있다.

[175] <원 구문>

[176]

Syntax	No. of bits

ac_spectral_data()
{
arith_reset_flag
 	
for (win = 0; win < num_windows; win++) {
arith_data(num_bands, arith_reset_flag)
}
}
	
	1

[177]

[178] <수정에 구문>

[179]

[180]

Syntax	No. of bits

ac_spectral_data(random_access)
{
if (random_access==TRUE)
arith_reset_flag = TRUE
else
arith_reset_flag
for (win = 0; win < num_windows; win++) {
arith_data(num_bands, arith_reset_flag)
}
}
	
	1

[181]

[182] 상술한 바와 같이, random access가 가능한 프레임의 경우, 복호화를 위해 컨텍스트 초기화를 수행하여야 하므로, arith_reset_flag는 항상 1로 설정될 수 있다. 이러한 경우에는 부호화 시 arith_reset_flag를 따로 포함시켜 전송할 필요가 없다

[183] TCX의 경우도 random access 프레임인 경우에는 이전 수퍼프레임을 참고할 수 없기 때문에, 주파수 도메인 부호화의 경우와 마찬가지로 arith_reset_flag는 random access 프레임의 경우는 전송하지 않을 수 있다. 본 방식에 따른 TCX 관련 구문의 일례는 아래와 같다.

[184] <본 발명의 일실시예에 따른 TCX 관련 구문>

[185]

Syntax	No. of bits
tcx_coding(random_access, lg, first_tcx_flag)	
{	
noise_factor	**3**
global_gain	**7**
if (first_tcx_flag) {	
if (random_access=TRUE) {	
arith_reset_flag=TRUE	
}	
else {	
arith_reset_flag	**1**
}	
}	
else {	
arith_reset_flag=0	
}	
arith_data(lg, arith_reset_flag)	
}	

[186]

[187] 또 다른 실시예로서, random access 프레임인 경우에는 현재 수퍼프레임이 선형 예측 도메인으로 부호화된 수퍼프레임인지에 대한 정보를 비트스트림에 포함시킬 수 있다. 즉, random access가 가능한 프레임에 대해서 현재 프레임에 이전 프레임이 주파수 도메인 부호화된 것인지 아니면 선형 예측 도메인 부호화된 것인지에 대한 정보(first_lpd_flag)를 포함시킬 수 있다. 선형 예측 도메인 부호화 방식 중 TCX의 경우, 복호화를 수행하기 위해서는 이전 수퍼프레임을 부호화할 때 사용했던 코어 부호화 모드 정보가 있어야 하지만, 현재 수퍼프레임의 TCX의 역양자화된 스펙트럼의 개수 즉 변환된 스펙트럼의 개수를 알 수 있기 때문에, 이를 위해서는 이전 프레임의 코어 부호화 모드 정보가 필요하므로, 구문(syntax)에 이전 수퍼프레임의 코어 부호화 모드 정보 또는 현재 수퍼프레임이 첫 번째 선형 예측 프레임인지의 정보 또는 이전 수퍼프레임이 선형 예측 프레임이었는지의 정보(first_lpd_flag)를 추가할 수 있다. 코드 부호화 모드 정보(first_lpd_flag)에 이전 수퍼프레임이 선형 예측 도메인 부호화되었다는 정보를 설정할 수 있다.

[188] <본 발명의 일실시예에 따른 first_lpd_flag의 설정 기준>

[189]

[190]

core_mode of previous frame (superframe)	core_mode of current frame (superframe)	first_lpd_flag
0	1	1
1	1	0

[191]

[192] <본 발명의 일실시예에 따른 first_lpd_flag가 포함된 구문>

[193]

[194]

Syntax	No. of bits
lpd_channel_stream(random_access)	
{	
acelp_core_mode	**3**
lpd_mode	**5**
first_tcx_flag=TRUE;	
k = 0;	
if (random_access==TRUE) {	
first_lpd_flag	**1**
}	
if (first_lpd_flag) { last_lpd_mode = 0; }	
while (k < 4) {	
if (mod[k] == 0) {	
acelp_coding(acelp_core_mode);	
last_lpd_mode=0;	
k += 1;	
}	
else {	
tcx_coding(random_access, lg(mod[k], last_lpd_mode) , first_tcx_flag);	
last_lpd_mode=mod[k];	
k += 2^(mod[k]-1);	
first_tcx_flag=FALSE;	
}	
}	
lpc_data(first_lpd_flag)	
}	

[195]

[196] 위에서 설명한 random access 프레임의 정보를 전송하는 방식으로, 각 tool에 따라 (1) random_access 프레임 각각에 대해 정보를 전송하는 방식, (2) USAC 전체 payload에 1번 전송하는 방식이 있을 수 있다. 예를 들어 주파수 도메인 정보가 담겨있는 single_channel_element에 random_access를 선언하여 주파수 도메인 부호화 부분에 대한 random access 정보를 담는 방식도 있을 수 있고, single_channel_element 대신에 USAC 전체 payload에 대한 정보를 담고 있는 부분에 random access 관련 정보를 포함시켜 모든 tool에 적용하는 방식도 있을 수 있다.

[197]

[198] 본 발명의 또 다른 일실시예에 따르면, Enhanced SBR의 경우 header가 있는지 없는지에 대한 정보를 1 비트를 사용해서 전송할 수 있다. 여기에서, random access 프레임의 경우 해당 프레임만으로 복호화가 가능하여야 하므로 header 정보가 전송되어야 한다. 따라서, random access가 선언되었는지에 따라 header에 대한 파싱(parsing)을 수행하도록 할 수 있다. 즉, random access가 선언된 경우에만 header 분석을 수행하고, random access가 선언되지 않은 경우에는 header 분석을 하지 않을 수 있다. 이를 위해, SBR header가 있는지 없는지에 대한 1 bit 정보를 마련하여, random access 프레임이 아닌 경우에만 1 bit 전송하여 random access 프레임이 아닌 프레임에 대해서 header parsing을 하는 불필요한 연산을 줄일 수 있다. 이를 수행하는 구문(syntax)의 일례가 도시되어 있다.

[199]

[200]

Syntax	No. of bits
sbr_extension_data(id_aac, crc_flag, random_access)	
{	
num_sbr_bits = 0;	
if (crc_flag) {	
bs_sbr_crc_bits;	**10**
num_sbr_bits += 10;	
}	
if (sbr_layer != SBR_STEREO_ENHANCE) {	
if (random_access==TRUE)	
num_sbr_bits += sbr_header();	
else {	
num_sbr_bits += 1;	
if (**bs_header_flag**)	**1**
num_sbr_bits += sbr_header();	
}	
}	
num_sbr_bits += sbr_data(id_aac, bs_amp_res);	
}	

[201]

[202] 위의 구문을 참조하면, random_access가 true 인 경우 bs_header_flag가 true로 설정되고, SBR header 분석이 수행되고, random_access가 false 인 경우에는 SBR header 정보가 필요한지에 대한 정보를 전송하여 확인 후에 필요한 경우에만 SBR header 분석이 수행되지 아니한다.

[203]

[204] 이하, 본 발명의 일실시예에 따른 복호화 방법의 다양한 방식을 상세히 설명한다.

[205] <방법 1: first_acelp_flag 추가로 첫 번째 ACELP 프레임에서 acelp_core_mode를 복호화 하는 방법>

[206]

[207] LPD의 부호화 모드 정보(lpd_mode)는 1개의 수퍼프레임이 ACELP와 TCX의 어떤 조합으로 구성이 되었는지에 대한 정보를 포함한다. ACELP가 사용된 경우에만 ACELP의 비트 할당 정보(acelp_core_mode)가 필요하다. 따라서, 이 경우에만 acelp_core_mode 정보를 복호화하는 형태로 구문을 구성할 수 있다. 현재 수퍼프레임 내에서 처음으로 ACELP가 부호화 된 프레임인 경우 acelp_core_mode를 읽고, 이후의 ACELP로 부호화된 프레임에 대해서는 acelp_core_mode를 읽지 않고 처음으로 읽은 acelp_core_mode를 바탕으로 ACELP 복호화를 수행할 수 있다.

[208] 상술한 방법은 다음과 같은 구문으로 표현이 가능하다. ACELP로 부호화된 프레임에서만 first_acelp_flag가 1인지의 여부를 읽고, 1인 경우에 acelp_core_mode를 읽어 ACELP 비트 할당 정보를 독출한 후, first_acelp_flag를 0으로 셋팅함으로써 다음의 ACELP 부호화된 프레임의 경우는

acelp_core_mode를 읽지 않는 방식으로 복호화를 수행한다.

[209]

Syntax	No. of bits
lpd_channel_stream()	
{	
lpd_mode	5
first_tcx_flag=TRUE;	
first_acelp_flag=TRUE;	
k = 0;	
if (first_lpd_flag) { last_lpd_mode = 0; }	
while (k < 4) {	
if (mod[k] == 0) {	
if (first_acelp_flag)	
acelp_core_mode	3
acelp_coding(acelp_core_mode);	
last_lpd_mode=0;	
k += 1;	
first_acelp_flag=FALSE;	
}	
else {	
tcx_coding(lg(mod[k], last_lpd_mode) , first_tcx_flag);	
last_lpd_mode=mod[k];	
k += (1<<(mod[k]-1));	
first_tcx_flag=FALSE;	
}	
}	
lpc_data(first_lpd_flag)	
}	

[210]

[211] <방법 2: is_tcx_only()를 통해 TCX 만 사용된 경우 acelp_core_mode를 복호화하지 않는 구성 및 lpd_mode!=0를 추가하여 이 경우에만 arith_reset_flag(비트 절감 0)를 복호화하는 구성>

[212] LPD 부호화 모드 정보(lpd_mode)는 1개의 수퍼프레임이 ACELP와 TCX의 어떤 조합으로 구성이 되었는지에 대한 정보를 포함한다. TCX만 사용한 경우, ACELP 비트 할당 정보(acelp_core_mode)를 비트스트림 내에 포함하지 않더라도, ACELP 복호화를 수행하지 않기 때문에 복호화가 가능하다. 따라서 ACELP가 1개의 수퍼프레임 내에 사용되지 않았다는 정보 또는 TCX만 사용되었다는 정보를 읽어서, acelp_core_mode 정보를 추가적으로 분석할 지를 결정한다. TCX만 사용한 것으로 판단한 경우, 수퍼프레임에 대해 TCX 복호화를 수행한다. ACELP가 포함된 경우, ACELP 비트 할당 정보(acelp_core_mode)를 추가적으로 분석한 후 ACELP 혹은 TCX 복호화를 수행할 수 있다. 또한 수퍼프레임 내의 모든 프레임이 ACELP로 복호화된 경우(lpd_mode=0), TCX 복호화가 수행되지 않기 때문에 arith_reset_flag가 필요 없다. 따라서, lpd_mode가 0인 경우에 arith_reset_flag 정보를 독출한 후 tcx_coding()을 통해 TCX 복호화를 수행할 수 있다. 이 때 수퍼프레임 내의 첫 번째 TCX 프레임에서만 컨텍스트를 초기화 하는 정보인 arith_reset_flag를 독출하기 때문에 arith_reset_flag를 독출한 후에 arith_reset_flag를 0으로 초기화를 수행하는 형태로 복호화를 수행한다.

[213] 아래는 lpd_channel_stream()의 구문의 일례를 보여준다.

[214]

```
Syntax                                                          No. of bits
lpd_channel_stream()
{
    lpd_mode                                                    5
    if (is_tcx_only(lpd_mode)==0)
        acelp_core_mode                                         3
    if (lpd_mode!=0)
        arith_reset_flag                                        1
    k = 0;
    if (first_lpd_flag) {   last_lpd_mode = 0;    }
    while (k < 4) {
        if (mod[k] == 0) {
            acelp_coding(acelp_core_mode);
            last_lpd_mode=0;
            k += 1;
        }
        else {
            tcx_coding( lg(mod[k], last_lpd_mode) , arith_reset_flag);
            last_lpd_mode=mod[k];
            k += (1<<(mod[k]-1));
            arith_reset_flag=0;
        }
    }

    lpc_data(first_lpd_flag)
}
```

[215]

[216] 또한, 아래는 tcx_coding() 구문의 일례를 보여준다.

[217]

[218]

```
Syntax                                                          No. of bits
tcx_coding(lg, arith_reset_flag)
{
    noise_factor                                                3
    global_gain                                                 7

    arith_data(lg, arith_reset_flag)
}
```

[219]

[220] <방법 3: first_acelp_flag를 추가하여 첫 번째 ACELP 프레임에서 acelp_core_mode를 복호화 하는 구성 및 lpd_mode!=0를 추가하여 이 경우에만 arith_reset_flag(비트 절감 0)를 복호화 하는 구성>

[221] ACELP로 부호화된 첫 프레임에서만 first_acelp_flag가 1인지의 여부를 판단하고, 1인 경우 acelp_core_mode를 읽어 ACELP 비트 할당 정보를 독출한 후, first_acelp_flag를 0으로 셋팅하여 다음의 ACELP 부호화된 프레임의 경우는 acelp_core_mode를 읽지 않는 방식으로 복호화를 수행한다. 또한 수퍼프레임 내의 모든 프레임이 ACELP로 복호화된 경우(lpd_mode=0), TCX 복호화가 수행되지 않기 때문에, arith_reset_flag가 필요 없다. 따라서, lpd_mode가 0인 경우 arith_reset_flag 정보를 독출한 후 tcx_coding()을 통해 TCX 복호화를 수행할 수 있다. 이 때 수퍼프레임 내의 첫 번째 TCX 프레임인 경우에만 컨텍스트를 초기화 하는 정보인 arith_reset_flag를 독출하기 때문에, arith_reset_flag를 독출한

후에 arith_reset_flag를 0으로 초기화를 수행하는 형태로 복호화를 수행한다.

[222]

[223] 아래는 lpd_channel_stream() 구문의 일례이다.

[224]

[225]

```
Syntax                                                          No. of bits
lpd_channel_stream()
{
    lpd_mode                                                    5
    if (lpd_mode!=0)
        arith_reset_flag                                        1
    first_acelp_flag=TRUE;
    k = 0;
    if (first_lpd_flag) {   last_lpd_mode = 0;    }
    while (k < 4) {
        if (mod[k] == 0) {
            if (first_acelp_flag)
                acelp_core_mode                                 3
            acelp_coding(acelp_core_mode);
            last_lpd_mode=0;
            k += 1;
            first_acelp_flag=FALSE;
        }
        else {
            tcx_coding( lg(mod[k], last_lpd_mode) , arith_reset_flag);
            last_lpd_mode=mod[k];
            k += (1<<(mod[k]-1));
            arith_reset_flag=0;
        }
    }

    lpc_data(first_lpd_flag)
}
```

[226]

[227] 아래는 tcx_coding() 구문의 일례이다.

[228]

[229]

```
Syntax                                                          No. of bits
tcx_coding(lg, arith_reset_flag)
{
    noise_factor                                                3
    global_gain                                                 7

    arith_data(lg, arith_reset_flag)
}
```

[230]

[231] <방법 4: 새로운 flag의 추가 없이 acelp_core_mode의 복호화 여부를 선택할 수 있는 구성>

[232] lpd_mode table을 아래와 같이 수정한다. 이 수정의 방향은 부호화 방식에 따라 table을 재 구성하는 것이다. 그래서 수퍼프레임 내의 모드 정보를 이용하여 ACELP 부호화만으로 이루어진 모드, ACELP 부호화와 TCX 부호화가 섞여 있는 모드, 및 TCX 부호화만으로 이루어진 모드를 각각 그룹핑하여 순서대로 정렬하는 방식으로 table을 재 구성한다. 이를 위하여 본 명세서에서는

new_lpd_mode를 새로 정의한다. 아래는 재 구성된 table의 한가지 예이다.

[233]

[234]

New_lpd_mode	mod[] entries based on frames in super-frame				Explanation
	frame 0	frame 1	frame 2	frame 3	
0	0	0	0	0	All ACELP frame
1..20	mod[0]	mod[1]	mod[2]	mod[3]	mod[x]=0,1,and 2
21	1	1	1	1	All TCX256 frame
22	1	1	2	2	
23	2	2	1	1	
24	2	2	2	2	All TCX512 frame
25	3	3	3	3	All TCX1024 frame
26..31					reserved

[235]

[236] 위의 table을 정의하면 다음과 같은 구문으로 표현할 수 있다. New_lpd_mode가 21 이상인 경우에 대해서만 acelp_core_mode를 읽어 ACELP 비트 할당 정보를 독출하는 방식으로 복호화를 수행한다. 이렇게 되면 디코더가 더 단순해 질 수 있다.

[237]

[238]

Syntax	No. of bits
lpd_channel_stream()	
{	
new_lpd_mode	**5**
if(**new_lpd_mode** < 21)	
acelp_core_mode	**3**
first_tcx_flag=TRUE;	
k = 0;	
if (first_lpd_flag) { last_lpd_mode = 0; }	
while (k < 4) {	
if (mod[k] == 0) {	
acelp_coding(acelp_core_mode);	
last_lpd_mode=0;	
k += 1;	
}	
else {	
tcx_coding(lg(mod[k], last_lpd_mode) , first_tcx_flag);	
last_lpd_mode=mod[k];	
k += 2^(mod[k]-1);	
first_tcx_flag=FALSE;	
}	
}	
lpc_data(first_lpd_flag)	
}	

[239]

[240] 제안된 방식은 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서, 부호화 방식에 따른 순서로 재구성된 새로운 선형 예측 부호화 모드(new_lpd_mode)를 분석하는 단계; 및 상기 새로운 선형 예측 부호화 모드 값의 크기에 따라 ACELP 부호화 모드의 독출 여부를 결정하는 단계; 필요한 경우 ACELP 부호화 모드를 독출하는 단계; 결정된 ACELP 부호화 모드와 new_lpd_mode에 따라 복호화를 수행하는 단계로 구성된다.

[241]

[242] 상술한 구성과 함께, acelp_core_mode와 lpd_mode의 리던던시(redundancy)를 이용하여 사용되는 비트를 절감하는 방법을 제안한다. 이를 위하여 위에서 언급된 new_lpd_mode table을 다음과 같이 4개의 그룹으로 나누어 이를 하위 lpd_mode_table로 정의할 수 있다.

[243]

[244]

하위 new_lpd_mode(sub_new_lpd_mode)	Contents	Meaning
New lpd_mode_0	New_lpd_mode = 0 (All ACELP frame)	Super-frame 전체가 ACELP로만 이루어짐
New lpd_mode_1	New_lpd_mode = 1~20	Super-frame내에서 ACELP 모드와 TCX모드가 혼합되어 있음
New lpd_mode_2	New_lpd_mode = 21~24 All TCX frame with 256 and 512	TCX256과 TCX512로만 이루어진 Super-frmae
New lpd_mode_3	New_lpd_mode = 25 (ALL TCX 1024 frame)	Super-frame 전체가 TCX1024로 이루어짐

[245]

[246] <방법 5: 신규 모드 정의하여 비트 수 절감하는 방식>

[247] 기본적으로 lpd_mode는 5비트를 사용하지만 실제로 사용 가능한 가지 수는 26가지의 경우를 지정하고 있고, 나머지 6가지는 reserved로 남겨둔 상태이다. 그리고 acelp_core_mode에서 mode 6과 mode7의 사용 가능성이 매우 적으므로, 사용 가능성이 낮은 mode 6과 mode 7 대신에 mode 6과 mode 7에 새로운 모드를 정의하여 추가적으로 비트를 줄이는 방식이 제안된다. 여기서는 새롭게 acelp_core_mode의 모드를 정의해야 하므로 명칭 수정이 필요하다. 본 명세서에서는 이전에 사용된 acelp_core_mode와 구분하기 위해 temporal_core_mode로 명명하며 그 의미는 아래와 같이 재 정의될 수 있다. 새롭게 정의된 temporal_core_mode는 acelp_core_mode에서 많이 사용되는 모드와 하위 new_lpd_mode에서 많이 사용되는 모드로 구성된다. 아래는 모드를 할당한 한가지 예이다. 여기서 Temporal_core_mode가 0~5인 경우에 선택 가능한 new_lpd_mode는 하위 new_lpd_mode(sub_new_lpd_mode)인 new_lpd_mode_0과 new_lpd_mode_1에 해당하는 new_lpd_mode이다. 이를 new_lpd_mode_01 이라고 정의한다. 여기서 New lpd_mode_01은 총 21개의 element가 있으므로 부호화를 위해 최대로 사용 가능한 비트는 5비트이다. 아래에서 New lpd_mode_01은 위의 new_lpd_mode 0~20의 21가지 경우를 의미하며 이 21가지를 부호화 하기 위해서

다양한 부호화 방식을 적용할 수 있다. 예를 들어 엔트로피 부호화(Entropy coding)를 이용하면 추가적으로 비트를 절감할 수 있다. 그리고 Temporal_core_mode가 6인 경우에 선택 가능한 new_lpd_mode는 New lpd_mode_2에 해당하는 new_lpd_mode이며, 21~24의 4가지 경우에 대해 2비트로 부호화가 가능하다. 그리고 Temporal_core_mode가 7인 경우에 선택 가능한 new_lpd_mode는 New lpd_mode_3에 해당하는 new_lpd_mode이며, 이는 new_lpd_mode 25로 한정되어 비트할당이 필요 없다.

[248]

Temporal_core_mode의 조건	선택가능한 New_lpd_mode	Meaning	Temporal_core_mode & new_lpd_mode Encoding bits
0~5	0~20	사용되는 Acelp_core_mode는 0~5이며 ACELP로만 구성된 프레임과 ACELP와 TCX가 혼합된 경우를 나타냄	Temporal_core_mode = 3 bits, New lpd_mode_01 = 5 bits
6	21~24	TCX256과 TCX512로만 구성된 프레임을 의미함	Temporal_core_mode = 3 bits, New lpd_mode_2 = 2bits
7	25	기존 방식에서 TCX 1024 프레임만 할당함 (ACELP 가 사용 안되는 경우)	Temporal_core_mode = 3 bits, No bit is needed for new_lpd_mode

[249] 이와 같이 테이블을 정의한 후, 이 테이블을 이용한 구문은 아래와 같이 구성할 수 있다.

[250]

[251]

Syntax	No. of bits

lpd_channel_stream()
{
  Temporal_core_mode
  if(**Temporal_core_mode** == 6){
  New_lpd_mode_2
  New_lpd_mode= New_lpd_mode_2+21;}
  else if(**Temporal_core_mode** == 7){
  New_lpd_mode = 25;}
  Else{
  New_lpd_mode_01
  New_lpd_mode=New_lpd_mode_01;}
  first_tcx_flag=TRUE;
  k = 0;
  if (first_lpd_flag) { last_lpd_mode = 0; }
  while (k < 4) {
  if (mod[k] == 0) {
  acelp_coding(acelp_core_mode);
  last_lpd_mode=0;
  k += 1;
  }
  else {
  tcx_coding(lg(mod[k], last_lpd_mode) , first_tcx_flag);
  last_lpd_mode=mod[k];
  k += 2^(mod[k]-1);
  first_tcx_flag=FALSE;
  }
  }
 	
  lpc_data(first_lpd_flag)
}
	
	3
	
	2
	
	5

[252]

[253] 제안된 방식은 ACELP 부호화 모드에서 사용 가능성이 낮은 모드 대신에 사용 가능성이 높은 하위 new_lpd_mode를 할당하여 재구성된 temporal_core_mode를 분석하는 단계; 선택된 temporal_core_mode에 따라 ACELP 부호화 모드와 하위 new_lpd_mode를 독출하는 단계; 독출된 상기 ACELP 부호화 모드와 하위 new_lpd_mode를 이용하여 ACELP 부호화 모드 및 new_lpd_mode를 결정하는 단계; 그리고 결정된 ACELP 부호화 모드와 new_lpd_mode에 따라 복호화를 수행하는 단계로 구성된다.

[254]

[255] <방법 6: 사용 가능성이 낮은 모드도 그대로 유지하는 구성>

[256] 위에서 설명한 <방법 5>는 사용가능성이 낮은 모드에 하위 new_lpd_mode를 할당하는 것이었으나, 만일 사용 가능성이 낮지만 이들을 모두 유지하려면 다음과 같은 방식이 가능하다. 먼저 acelp_core_mode의 mode 6과 mode 7을 원래대로 사용하는 경우에는 총 8개 모드가 되므로 정확히 3비트가 필요하다. 이런 경우 new_lpd_mode의 그룹핑된 하위 group을 할당하기 위해 frame_mode를 도입한다. Frame_mode는 2비트로 부호화되며 각 frame_mode의 의미는 아래의 테이블과 같다.

[257]

[258]

Frame_mode (2bits)	Meaning	Ace lp_core_mode & new_lpd_mode Encoding bits
0	New_lpd_mode = 0 (All ACELP frame)	acelp_core_mode = 3bits New_lpd_mode_0 = 0bits
1	New_lpd_mode = 25 (ALL TCX 1024 frame)	acelp_core_mode = 0bits New_lpd_mode_3 = 0bits
2	New_lpd_mode = 21~24 All TCX frame with 256 and 512	acelp_core_mode = 0bits New_lpd_mode_2 = 2bits
3	New_lpd_mode = 1~20	acelp_core_mode = 3bits New_lpd_mode_1 = 5bits

[259]

[260] 여기에서는 frame_mode에 따라 하위 new_lpd_mode가 선택되며, 각 하위 new_lpd_mode에 따라 사용 비트가 달라진다. 이와 같이 테이블을 정의한 후, 이 테이블을 이용한 구문은 아래와 같이 구성할 수 있다.

[261]

[262]

Syntax	No. of bits
lpd_channel_stream() { Frame_mode	2
if(Frame_mode == 0){ New_lpd_mode = 0; Acelp_core_mode}	3
else if(Frame_mode == 1){ New_lpd_mode = 25;}	
else if(Frame_mode == 2){ New_lpd_mode_2	2
New_lpd_mode = New_lpd_mode_2 + 21;}	
else{ New_lpd_mode_1	5
Acelp_core_mode	3
New_lpd_mode=New_lpd_mode_1+1;}	
first_tcx_flag=TRUE;	
k = 0;	
if (first_lpd_flag) { last_lpd_mode = 0; }	
while (k < 4) {	
if (mod[k] == 0) {	
acelp_coding(acelp_core_mode);	
last_lpd_mode=0;	
k += 1;	
}	
else {	
tcx_coding(lg(mod[k], last_lpd_mode) , first_tcx_flag);	
last_lpd_mode=mod[k];	
k += 2^(mod[k]-1);	
first_tcx_flag=FALSE;	
}	
}	
lpc_data(first_lpd_flag)	
}	

[263]

[264] 제안된 방식은 new_lpd_mode의 그룹핑된 하위 group을 할당하기 위해 마련된 Frame_mode를 분석하는 단계; 선택된 frame_mode에 해당하는 ACELP 부호화 모드와 하위 new_lpd_mode를 독출하는 단계; 독출된 상기 ACELP 부호화 모드와 하위 new_lpd_mode를 이용하여 ACELP 부호화 모드 및 new_lpd_mode를 결정하는 단계; 그리고 결정된 ACELP 부호화 모드와 new_lpd_mode에 따라 복호화를 수행하는 단계를 포함하는 오디오 신호 또는 스피치 신호의 복호화를 수행하는 단계로 구성된다.

[265] 이상과 같이 본 발명은 비록 한정된 실시예와 도면에 의해 설명되었으나, 본 발명은 상기의 실시예에 한정되는 것은 아니며, 본 발명이 속하는 분야에서 통상의 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형이 가능하다.

[266] 본 발명에 따른 오디오 또는 스피치 신호의 부호화 및 복호화 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 신호 파일, 신호 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 본 발명을 위하여 특별히 설계되고 구성된

것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 그러므로, 본 발명의 범위는 설명된 실시예에 국한되어 정해져서는 아니 되며, 후술하는 특허청구범위뿐 아니라 이 특허청구범위와 균등한 것들에 의해 정해져야 한다.

청구범위

[청구항 1] 오디오 신호 또는 스피치 신호의 부호화 방법에 있어서,
선정된 기준에 따라 입력 신호를 선형 예측 도메인 부호화 모드 또는 주파수 도메인 부호화 모드 중 어느 하나로 스위칭하는 단계;
상기 선형 예측 도메인 부호화 모드로 스위칭된 경우, 하나의 수퍼프레임(수퍼프레임)에 포함된 복수의 프레임들의 선형 예측 부호화 모드를 선택하는 단계; 및
상기 선형 예측 부호화 모드 중 적어도 하나가 ACELP(Algebraic Code Excited Linear Prediction)인 경우에만 ACELP 비트 할당 정보(acelp_core_mode)를 비트스트림에 포함시키도록 제어하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 부호화 방법.

[청구항 2] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
비트스트림에 포함된 선형 예측 부호화 모드(lpd_mode)를 분석하는 단계; 및
상기 선형 예측 부호화 모드의 분석 결과, 수퍼프레임에 포함된 복수의 프레임이 TCX로 부호화된 경우 상기 수퍼프레임에 대한 TCX 복호화를 수행하되, 상기 수퍼프레임의 적어도 일부가 ACELP로 부호화된 경우 ACELP 비트 할당 정보를 더 독출하여 ACELP 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 3] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
비트스트림에 포함된 코어 모드 정보(core_mode)를 분석하여 주파수 도메인 복호화 모드인지 선형 예측 도메인 복호화 모드인지를 스위칭하는 단계;
상기 주파수 도메인 복호화 모드인 경우, 프레임에 대한 컨텍스트 초기화 여부 정보(arith_reset_flag)를 읽어 상기 컨텍스트 초기화 정보가 설정된 경우에 주파수 도메인 복호화를 수행하는 단계;
상기 선형 예측 도메인 복호화 모드인 경우, 수퍼프레임의 선형 예측 부호화 모드(lpd_mode)를 분석하되, 상기 수퍼프레임에 포함된 상기 대상 프레임이 ACELP로 부호화된 경우 ACELP 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 4] 제3항에 있어서,
상기 선형 예측 부호화 모드가 TCX 인 경우, 상기 대상 프레임에 대한 상기 컨텍스트 초기화 여부 정보(arith_reset_flag)를 읽는 단계; 및

상기 컨텍스트 초기화 정보가 설정된 경우, TCX 복호화를 수행하는 단계
를 더 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 5] 제3항에 있어서,
상기 주파수 도메인 복호화는 USAC(Unified Speech and Audio Codec)의 주파수 도메인 복호화USAC(Unified Speech and Audio Codec) 인 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 6] 오디오 신호 또는 스피치 신호의 부호화 방법에 있어서,
선정된 기준에 따라 입력 신호를 선형 예측 도메인 부호화 모드 또는 주파수 도메인 부호화 모드 중 어느 하나로 스위칭하는 단계;
및
상기 선형 예측 도메인 부호화 모드 또는 상기 주파수 도메인 부호화 모드로 상기 입력 신호를 부호화하되, 부호화된 수퍼프레임에 포함된 복수의 프레임에 대한 랜덤 억세스 가능 정보(random_access)를 포함시키는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 부호화 방법.

[청구항 7] 제6항에 있어서,
상기 랜덤 억세스 가능 정보는 상기 프레임 각각에 대해 할당되는 오디오 신호 또는 스피치 신호의 부호화 방법.

[청구항 8] 제6항에 있어서,
상기 랜덤 억세스 가능 정보는 페이로드(payload)에 포함되는 오디오 신호 또는 스피치 신호의 부호화 방법.

[청구항 9] 제6항에 있어서,
랜덤 억세스 가능한 프레임인 경우 컨텍스트 초기화 정보(arith_reset_flag)는 1로 설정되는 오디오 신호 또는 스피치 신호의 부호화 방법.

[청구항 10] 제6항에 있어서,
랜덤 억세스 가능한 프레임인 경우, 상기 프레임의 윈도우 타입 정보(window_sequence)는 3비트 값을 갖는 오디오 신호 또는 스피치 신호의 부호화 방법.

[청구항 11] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
비트스트림에 포함된 랜덤 억세스 가능 정보(random_access)를 분석하여 수퍼프레임에 포함된 프레임이 랜덤 억세스 가능한 프레임인지 판단하는 단계;
랜덤 억세스 가능한 프레임이 아닌 경우, 다음 프레임에 대해 판단을 계속하되, 랜덤 억세스 가능한 프레임인 경우, 코어 모드 정보(core_mode)를 분석하여 주파수 도메인 복호화 모드인지 선형 예측 도메인 복호화 모드인지를 스위칭하고, 주파수 도메인

복호화 또는 선형 예측 도메인 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 12] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
비트스트림에 포함된 선형 예측 부호화 모드(lpd_mode)를
분석하는 단계; 및
상기 선형 예측 부호화 모드의 분석 결과, 수퍼프레임에 포함된 한
개 이상의 프레임 중 처음으로 ACELP로 부호화된 경우 ACELP
비트 할당 정보를 독출하여 ACELP 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 13] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
비트스트림에 포함된 선형 예측 부호화 모드(lpd_mode)를
분석하는 단계; 및
상기 선형 예측 부호화 모드의 분석 결과, 수퍼프레임에 포함된 한
개 이상의 프레임 중 ACELP로 부호화된 경우에 있어서, 두번 째
ACELP 프레임에서는 ACELP 비트 할당 정보를 첫번째 ACELP
부호화 프레임 복호화 과정 중 독출된 ACELP 비트 할당 정보를
사용하여 ACELP 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 14] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
비트스트림에 포함된 선형 예측 부호화 모드(lpd_mode)를
분석하는 단계; 및
상기 선형 예측 부호화 모드의 분석 결과, 수퍼프레임에 포함된 한
개 이상의 프레임 중 ACELP로 부호화된 경우, 두 번째 ACELP
프레임에서는 ACELP 비트 할당 정보를 첫 번째 ACELP 부호화
프레임 복호화 과정 중 독출된 ACELP 비트 할당 정보를 사용하여
ACELP 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

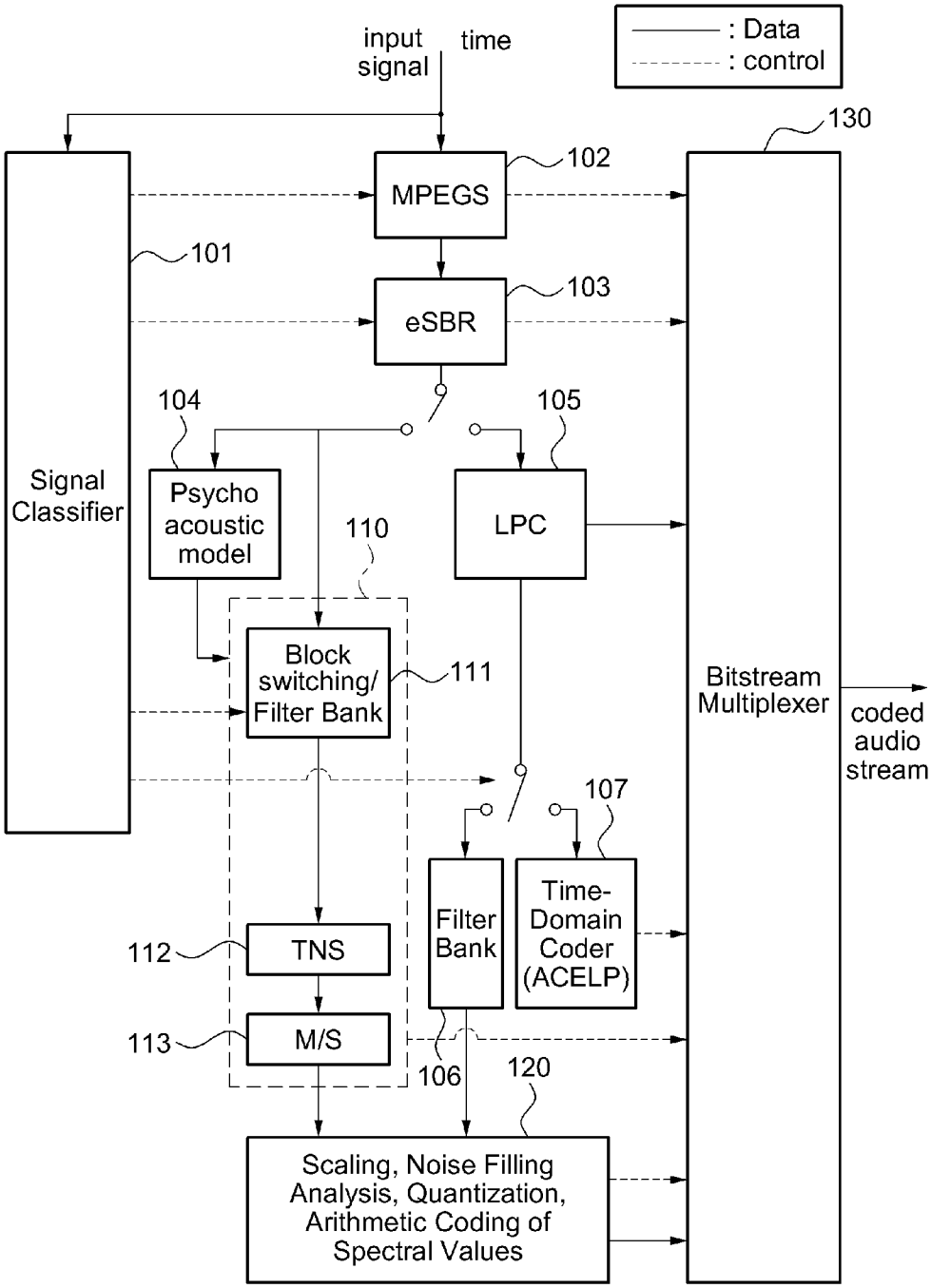
[청구항 15] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
부호화 방식에 따른 순서로 재구성된 신규 선형 예측 부호화
모드(new_lpd_mode)를 분석하는 단계; 및
상기 새로운 선형 예측 부호화 모드값의 크기에 따라 ACELP
부호화 모드의 독출 여부를 결정하는 단계;
ACELP 부호화 모드를 독출하는 단계;
상기 ACELP 부호화 모드와 상기 신규 선형 예측 부호화
모드(new_lpd_mode)에 따라 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 16] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
ACELP 부호화 모드에서 가용 확률이 높은 하위 신규 선형 예측

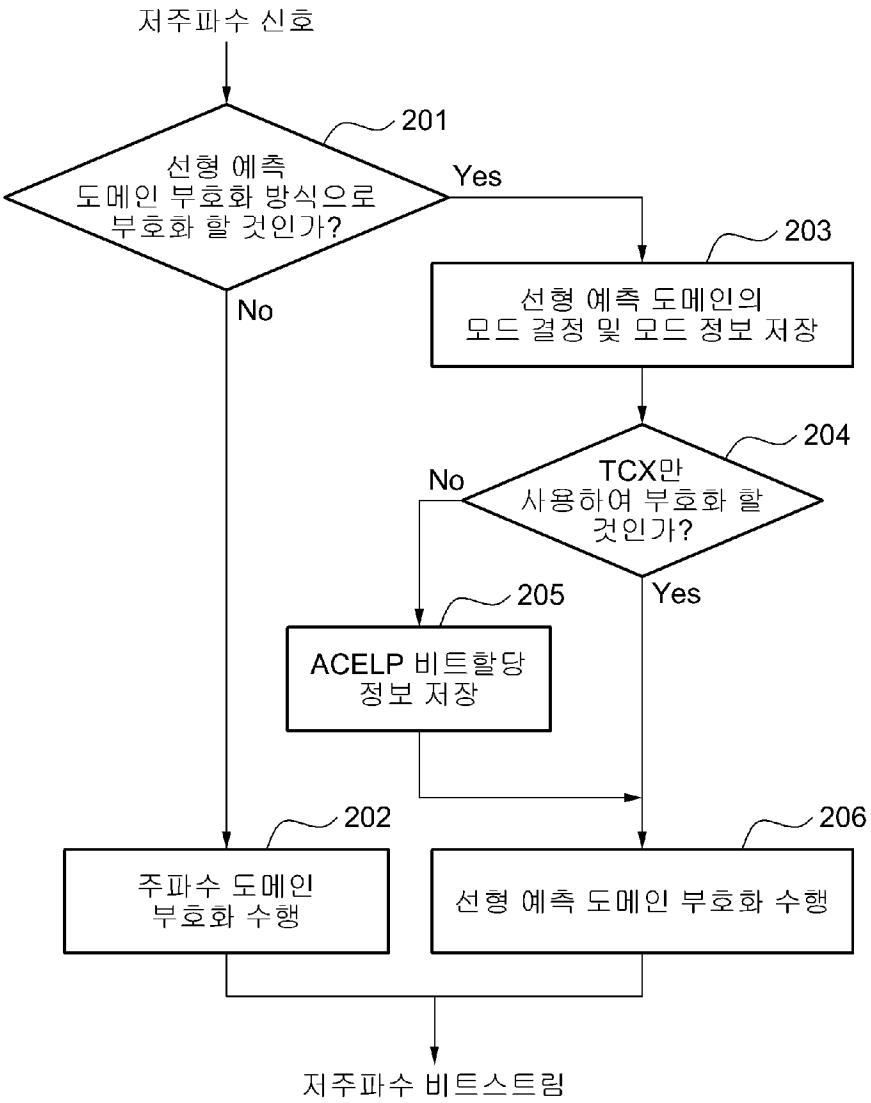
부호화 모드(new_lpd_mode)를 할당하여 재구성된 시간 코어 모드(temporal_core_mode)를 분석하는 단계; 및
상기 시간 코어 모드(temporal_core_mode)에 따라 상기 ACELP 부호화 모드와 상기 하위 신규 선형 예측 부호화 모드(new_lpd_mode)를 결정하는 단계;
상기 ACELP 부호화 모드와 상기 하위 신규 선형 예측 부호화 모드(new_lpd_mode)에 따라 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

[청구항 17] 오디오 신호 또는 스피치 신호의 복호화 방법에 있어서,
신규 선형 예측 부호화 모드(new_lpd_mode)의 그룹핑된 하위 그룹을 할당하기 위한 프레임 모드(frame_mode)를 분석하는 단계; 및
상기 프레임 모드(frame_mode)에 해당하는 ACELP 부호화 모드와 하위 신규 선형 예측 부호화 모드(new_lpd_mode)를 결정하는 단계;
독출된 상기 ACELP 부호화 모드와 하위 new_lpd_mode를 이용하여 ACELP 부호화 모드 및 new_lpd_mode를 결정하는 단계;
상기 ACELP 부호화 모드와 상기 하위 신규 선형 예측 부호화 모드(new_lpd_mode)에 따라 복호화를 수행하는 단계
를 포함하는 오디오 신호 또는 스피치 신호의 복호화 방법.

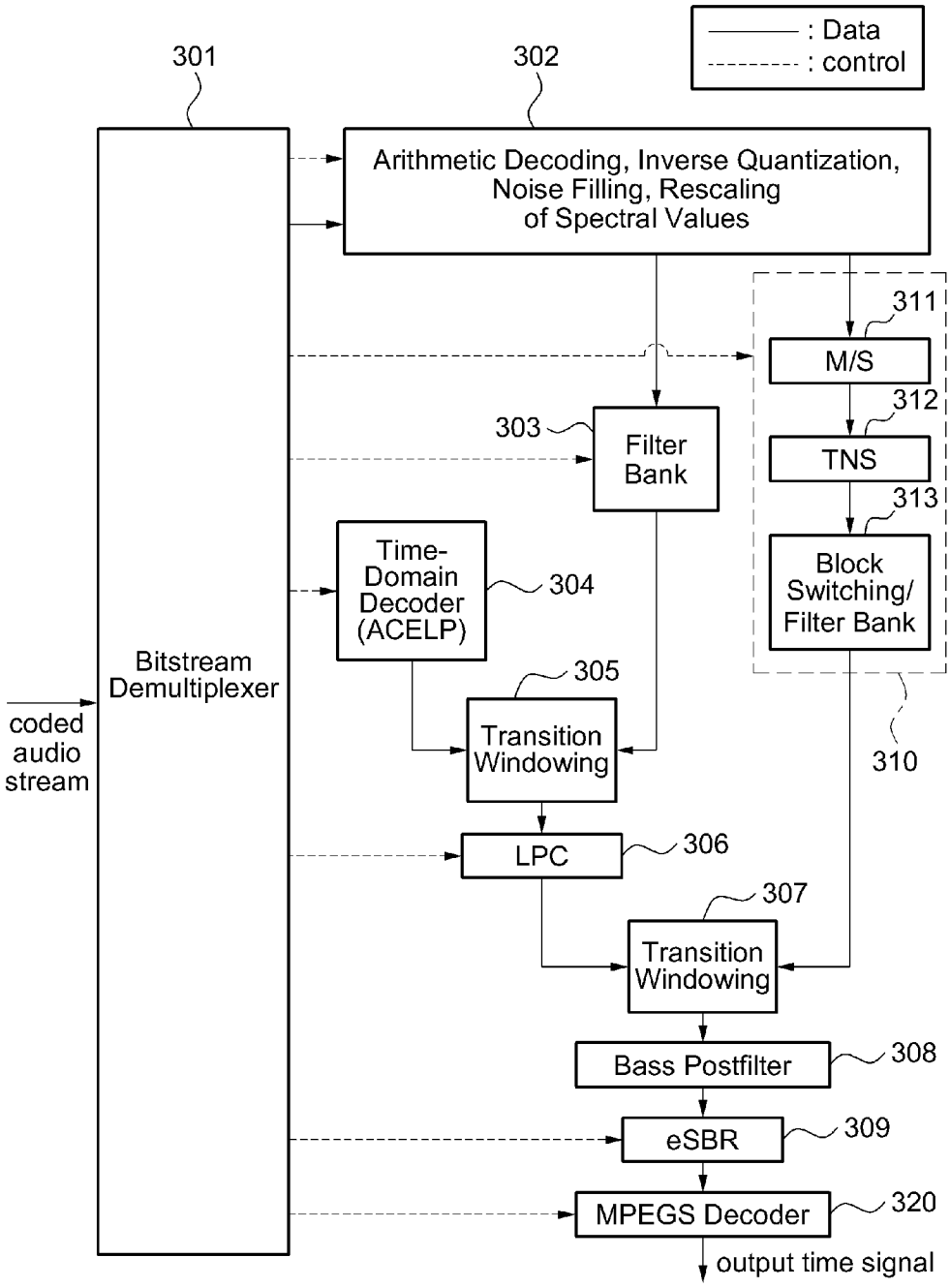
[Fig. 1]


input signal
time
: Data
: control
130
102
MPEGS
101
103
eSBR
104
105
Signal Classifier
Psycho acoustic model
LPC
110
Block switching/ Filter Bank
111
Bitstream Multiplexer
coded audio stream
107
Filter Bank
Time-Domain Coder (ACELP)
112
TNS
113
M/S
106
120
Scaling, Noise Filling Analysis, Quantization, Arithmetic Coding of Spectral Values

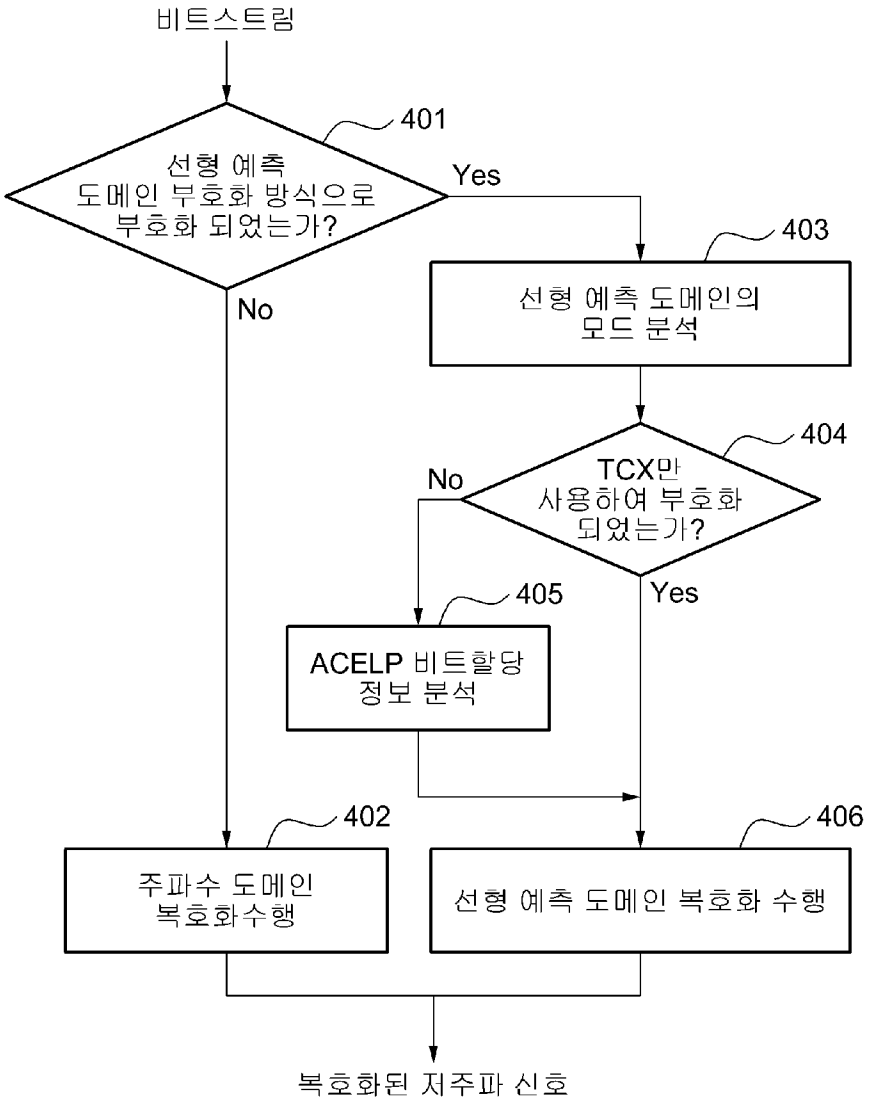
[Fig. 2]


저주파수 신호
201
선형 예측 도메인 부호화 방식으로 부호화 할 것인가?
Yes
No
203
선형 예측 도메인의 모드 결정 및 모드 정보 저장
204
TCX만 사용하여 부호화 할 것인가?
No
Yes
205
ACELP 비트할당 정보 저장
202
주파수 도메인 부호화 수행
206
선형 예측 도메인 부호화 수행
저주파수 비트스트림

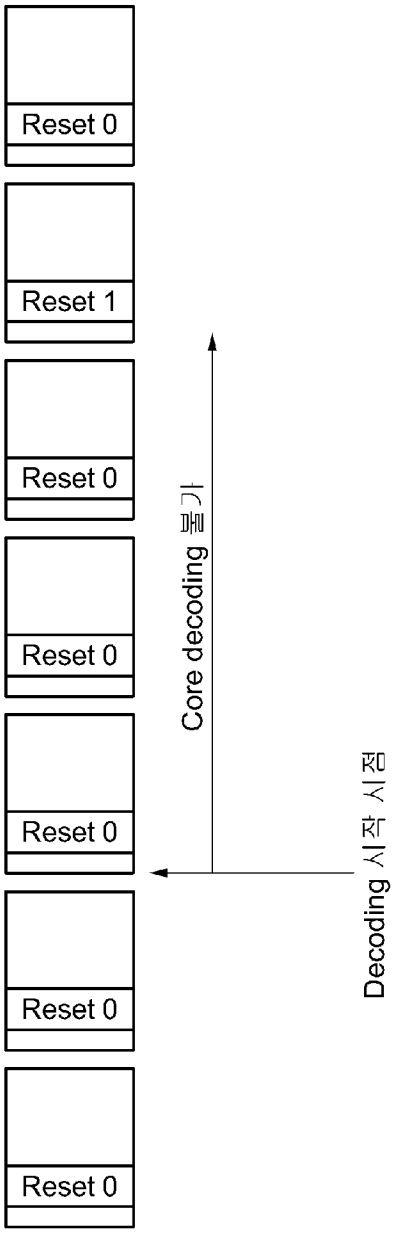
[Fig. 3]


: Data
: control
301
302
Arithmetic Decoding, Inverse Quantization, Noise Filling, Rescaling of Spectral Values
311
M/S
312
TNS
313
Block Switching/ Filter Bank
310
303
Filter Bank
Time-Domain Decoder (ACELP)
304
Bitstream Demultiplexer
coded audio stream
305
Transition Windowing
LPC
306
307
Transition Windowing
Bass Postfilter
308
eSBR
309
MPEGS Decoder
320
output time signal

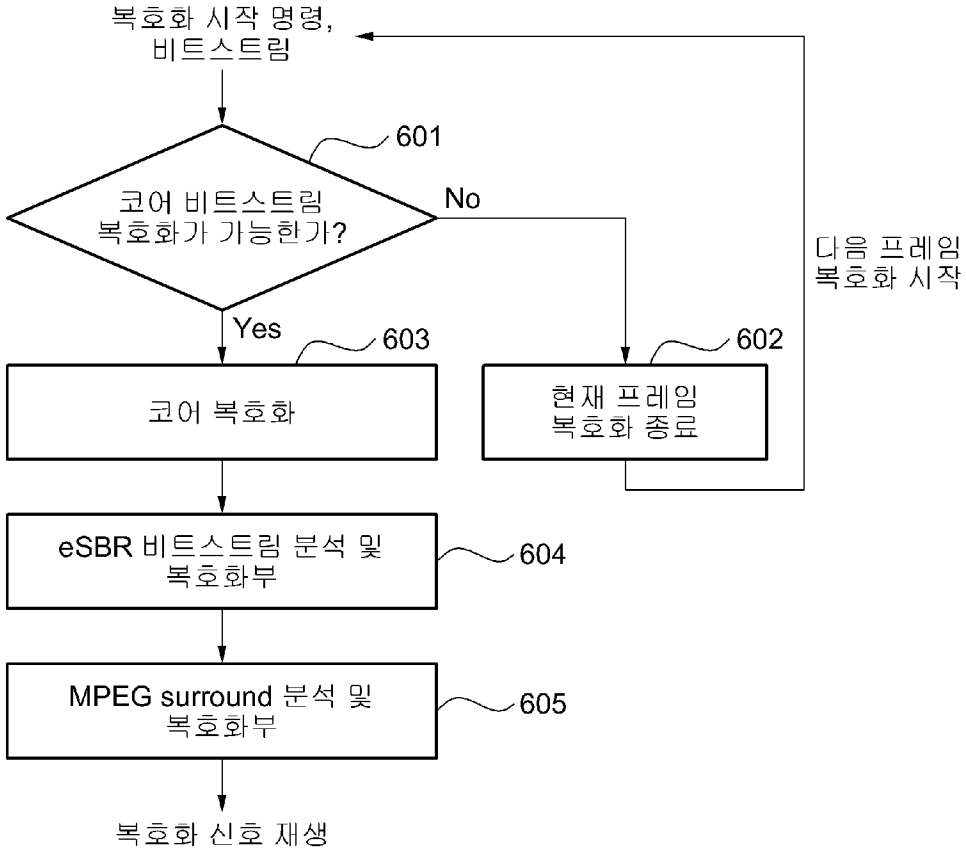
[Fig. 4]


비트스트림
401
선형 예측
도메인 부호화 방식으로
부호화 되었는가?
Yes
No
403
선형 예측 도메인의
모드 분석
404
TCX만
사용하여 부호화
되었는가?
No
Yes
405
ACELP 비트할당
정보 분석
402
주파수 도메인
복호화수행
406
선형 예측 도메인 복호화 수행
복호화된 저주파 신호

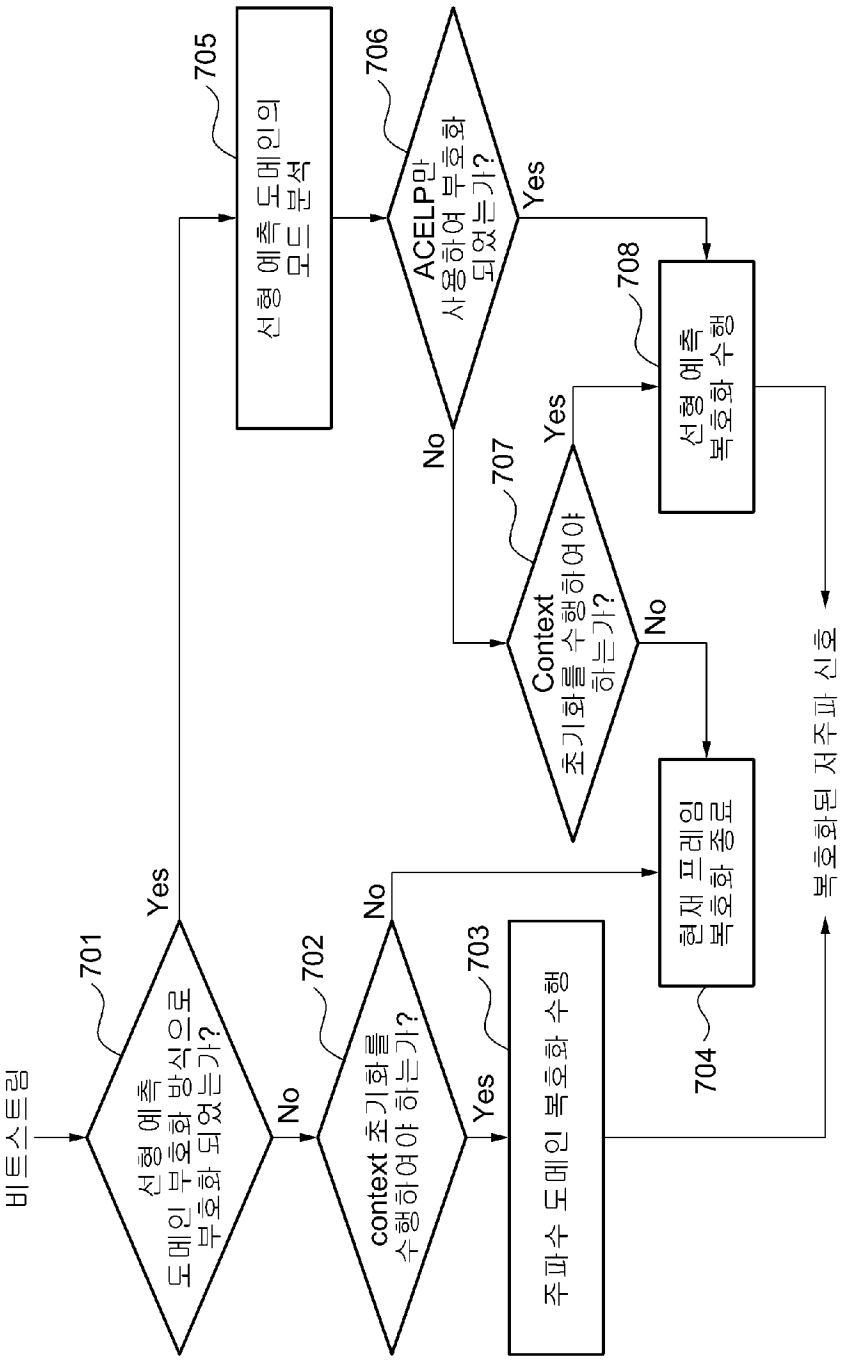
[Fig. 5]


Reset 0
Reset 1
Reset 0
Reset 0
Reset 0
Reset 0
Reset 0
Core decoding 불가
Decoding 시작 시점

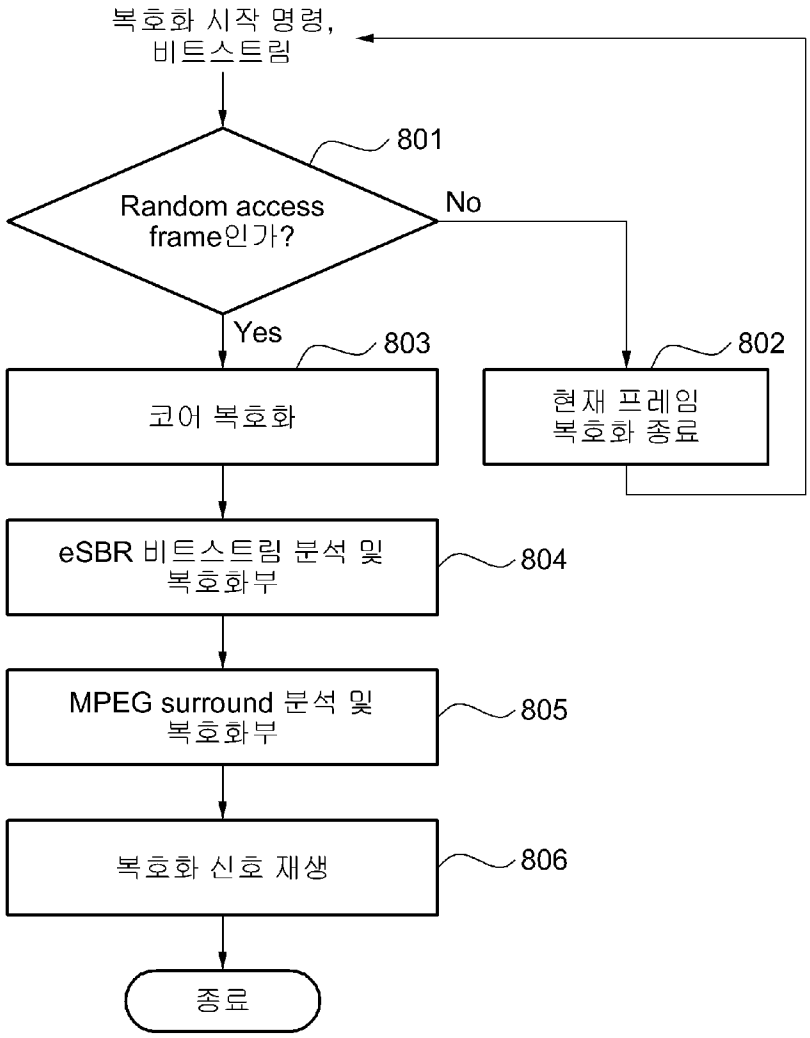
[Fig. 6]


복호화 시작 명령,
비트스트림
601
코어 비트스트림
복호화가 가능한가?
No
Yes
603
602
코어 복호화
현재 프레임
복호화 종료
다음 프레임
복호화 시작
eSBR 비트스트림 분석 및
복호화부
604
MPEG surround 분석 및
복호화부
605
복호화 신호 재생

[Fig. 7]


비트스트림
701
선형 예측 도메인 부호화 방식으로 부호화 되었는가?
Yes
No
702
context 초기화를 수행하여야 하는가?
No
Yes
703
주파수 도메인 복호화 수행
704
현재 프레임 복호화 종료
705
선형 예측 도메인의 모드 분석
706
ACELP만 사용하여 부호화 되었는가?
Yes
No
707
Context 초기화를 수행하여야 하는가?
Yes
No
708
선형 예측 복호화 수행
복호화된 저주파 신호

[Fig. 8]


복호화 시작 명령,
비트스트림
801
Random access
frame인가?
No
Yes
803
802
코어 복호화
현재 프레임
복호화 종료
eSBR 비트스트림 분석 및
복호화부
804
MPEG surround 분석 및
복호화부
805
복호화 신호 재생
806
종료