

Post-processor, Pre-processor, Audio Encoder, Audio Decoder and related Methods for Enhancing Transient Processing

Specification

5

The present invention is related to audio processing and, particularly, to audio processing in the context of audio pre-processing and audio post-processing.

10 **Pre-echoes: The temporal masking problem**

Classic filterbank based perceptual coders like MP3 or AAC are primarily designed to exploit the perceptual effect of *simultaneous masking*, but also have to deal with the temporal aspect of the masking phenomenon: Noise is masked a short time prior to and after the presentation of a masking signal (*pre-masking* and *post-masking* phenomenon).

15 Post-masking is observed for a much longer period of time than pre-masking (in the order of 10.0-50.0ms instead of 0.5-2.0ms, depending on the level and duration of the masker).

Thus, the temporal aspect of masking leads to an additional requirement for a perceptual coding scheme: In order to achieve perceptually transparent coding quality the
20 quantization noise also must not exceed the time-dependent masked threshold.

In practice, this requirement is not easy to achieve for perceptual coders because using a spectral signal decomposition for quantization and coding implies that a quantization error introduced in this domain will be spread out in time after reconstruction by the synthesis
25 filterbank (time/frequency uncertainty principle). For commonly used filterbank designs (e.g. a 1024 lines MDCT) this means that the quantization noise may be spread out over a period of more than 40 milliseconds at CD sampling rate. This will lead to problems when the signal to be coded contains strong signal components only in parts of the analysis filterbank window, i. e. for transient signals. In particular, quantization noise is spread out
30 *before* the onsets of the signal and in extreme cases may even exceed the original signal components in level during certain time intervals. A well-known example of a critical percussive signal is a castanets recording where after decoding quantization noise components are spread out a certain time before the “attack” of the original signal. Such a constellation is traditionally known as a “pre-echo phenomenon” [Joh92b].

35

Due to the properties of the human auditory system, such “pre-echoes” are masked only if no significant amount of coding noise is present longer than ca. 2.0ms before the onset of the signal. Otherwise, the coding noise will be perceived as a pre-echo artifact, i.e. a short noise-like event preceding the signal onset. In order to avoid such artifacts, care has to be taken to maintain appropriate temporal characteristics of the quantization noise such that it will still satisfy the conditions for temporal masking. This *temporal noise shaping problem* has traditionally made it difficult to achieve a good perceptual signal quality at low bit-rates for transient signals like castanets, glockenspiel, triangle etc.

10 **Applause-like signals: An extremely critical class of signals**

While the previously mentioned transient signals may trigger pre-echoes in perceptual audio codecs, they exhibit single isolated attacks, i.e. there is a certain minimum time until the next attack appears. Thus, a perceptual coder has some time to recover from processing the last attack and can, e.g., collect again spare bits to cope with the next attack (see ‘bit reservoir’ as described below). In contrast to this, the sound of an applauding audience consists of a steady stream of densely spaced claps, each of which is a transient event of its own. Fig. 11 shows an illustration of the high frequency temporal envelope of a stereo applause signal. As can be seen, the average time between subsequent clap events is significantly below 10ms.

For this reason, applause and applause-like signals (like rain drops or crackling fireworks) constitute a class of extremely difficult to code signals while being common to many live recordings. This is also true when employing parametric methods for joint coding of two or more channels [Hot08].

25 **Traditional approaches to coding of transient signals**

A set of techniques has been proposed in order to avoid pre-echo artifacts in the encoded / decoded signal:

30 **Pre-echo control and bit reservoir**

One way is to increase the coding precision for the spectral coefficients of the filterbank window that first covers the transient signal portion (so-called “pre-echo control”, [MPEG1]). Since this considerably increases the amount of necessary bits for the coding of such frames this method cannot be applied in a constant bit rate coder. To a certain degree, local variations in bit rate demand can be accounted for by using a bit reservoir ([Bra87], [MPEG1]). This technique permits to handle peak demands in bit rate using bits

that have been set aside during the coding of earlier frames while the average bit rate still remains constant.

Adaptive window switching

- 5 A different strategy used in many perceptual audio coders is adaptive window switching as introduced by Edler [Edl89]. This technique adapts the size of the filterbank windows to the characteristics of the input signal. While stationary signal parts will be coded using a long window length, short windows are used to code the transient parts of the signal. In this way, the peak bit demand can be reduced considerably because the region for which
- 10 a high coding precision is required is constrained in time. Pre-echoes are limited in duration implicitly by the shorter transform size.

Temporal Noise Shaping (TNS)

- Temporal Noise Shaping (TNS) was introduced in [Her96] and achieves a temporal
- 15 shaping of the quantization noise by applying open-loop predictive coding along frequency direction on time blocks in the spectral domain.

Gain modification (gain control)

- Another way to avoid the temporal spread of quantization noise is to apply a dynamic gain
- 20 modification (gain control process) to the signal prior to calculating its spectral decomposition and coding.

The principle of this approach is illustrated in Fig. 12. The dynamics of the input signal is reduced by a gain modification (multiplicative pre-processing) prior to its encoding. In this

25 way, “peaks” in the signal are attenuated prior to encoding. The parameters of the gain modification are transmitted in the bitstream. Using this information the process is reversed on the decoder side, i.e. after decoding another gain modification restores the original signal dynamics.

- 30 [Lin93] proposed a gain control as an addition to a perceptual audio coder where the gain modification is performed on the time domain signal (and thus to the entire signal spectrum).

- Frequency dependent gain modification / control has been used before in a number of
- 35 instances:

Filter-based Gain Control: In his dissertation [Vau91], Vaupel notices that full band gain control does not work well . In order to achieve a frequency dependent gain control he proposes a compressor and expander filter pair which can be dynamically controlled in their gain characteristics. This scheme is shown in Figs. 13a and 13b.

5

The variation of the filter's frequency response is shown in Fig. 13b.

Gain Control With Hybrid Filterbank (illustrated in Fig. 14): In the SSR profile of the MPEG-2 Advanced Audio Coding [Bos96] scheme, gain control is used within a hybrid
10 filterbank structure. A first filterbank stage (PQF) splits the input signal into four bands of equal width. Then, a gain detector and a gain modifier perform the gain control encoder processing. Finally, as a second stage, four separate MDCT filterbanks with a reduced size (256 instead of 1024) split the resulting signal further and produce the spectral
15 components that are used for subsequent coding.

15

Guided envelope shaping (GES) is a tool contained in MPEG Surround that transmits channel-individual temporal envelope parameters and restores temporal envelopes on the decoder side. Note that, contrary to HREP processing, there is no envelope flattening on the encoder side in order to maintain backward compatibility on the downmix. Another tool
20 in MPEG Surround that functions to to perform envelope shaping is Subband Temporal Processing (STP). Here, low order LPC filters are applied within a QMF filterbank representation of the audio signals.

Related prior art is documented in Patent publications WO 2006/045373 A1, WO
25 2006/045371 A1, WO2007/042108 A1, WO 2006/108543 A1, or WO 2007/110101 A1.

References

[Bos96] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fielder, K. Akagiri, H. Fuchs,
30 M. Dietz, J. Herre, G. Davidson, Oikawa: "MPEG-2 Advanced Audio Coding", 101st AES Convention, Los Angeles 1996

[Bra87] K. Brandenburg: "OCF - A New Coding Algorithm for High Quality Sound
35 Signals", Proc. IEEE ICASSP, 1987

35

- [Joh92b] J. D. Johnston, K. Brandenburg: "Wideband Coding Perceptual Considerations for Speech and Music", in S. Furui and M. M. Sondhi, editors: "Advances in Speech Signal Processing", Marcel Dekker, New York, 1992
- 5
- [Edl89] B. Edler: "Codierung von Audiosignalen mit überlappender Transformation und adaptiven Fensterfunktionen", Frequenz, Vol. 43, pp. 252-256, 1989
- [Her96] J. Herre, J. D. Johnston: "Enhancing the Performance of Perceptual Audio Coders by Using Temporal Noise Shaping (TNS)", 101st AES Convention, Los Angeles 1996, Preprint 4384
- 10
- [Hot08] Gerard Hotho, Steven van de Par, and Jeroen Breebaart: "Multichannel coding of applause signals", EURASIP Journal of Advances in Signal Processing, Hindawi, January 2008, doi: 10.1155/2008/531693
- 15
- [Lin93] M. Link: "An Attack Processing of Audio Signals for Optimizing the Temporal Characteristics of a Low Bit-Rate Audio Coding System", 95th AES convention, New York 1993, Preprint 3696
- 20
- [MPEG1] ISO/IEC JTC1/SC29/WG11 MPEG, International Standard ISO 11172-3 "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s"
- 25
- [Vau91] T. Vaupel: "Ein Beitrag zur Transformationscodierung von Audiosignalen unter Verwendung der Methode der 'Time Domain Aliasing Cancellation (TDAC)' und einer Signalkompandierung im Zeitbereich", PhD Thesis, Universität-Gesamthochschule Duisburg, Germany, 1991
- 30
- A bit reservoir can help to handle peak demands on bitrate in a perceptual coder and thereby improve perceptual quality of transient signals. In practice, however, the size of the bit reservoir must be unrealistically large in order to avoid artifacts when coding input signals of a very transient nature without further precautions.
- 35
- Adaptive window switching limits the bit demand of transient parts of the signal and reduced pre-echoes through confining transients into short transform blocks. A limitation

of adaptive window switching is given by its latency and repetition time: The fastest possible turn-around cycle between two short block sequences requires at least three blocks ("short"→"stop"→"start"→"short", ca. 30.0 – 60.0ms for typical block sizes of 512 - 1024 samples) which is much too long for certain types of input signals including
5 applause. Consequently, temporal spread of quantization noise for applause-like signals could only be avoided by permanently selecting the short window size, which usually leads to a decrease in the coder's source-coding efficiency.

10 TNS performs temporal flattening in the encoder and temporal shaping in the decoder. In principle, arbitrarily fine temporal resolution is possible. In practice, however, the performance is limited by the temporal aliasing of the coder filterbank (typically an MDCT, i.e. an overlapping block transform with 50% overlap). Thus, the shaped coding noise appears also in a mirrored fashion at the output of the synthesis filterbank.

15 Broadband gain control techniques suffer from a lack of spectral resolution. In order to perform well for many signals, however, it is important that the gain modification processing can be applied independently in different parts of the audio spectrum because transient events are often dominant only in parts of the spectrum (in practice the events that are difficult to code are present almost always in the high frequency part of the
20 spectrum). Effectively, applying a dynamic multiplicative modification of the input signal prior to its spectral decomposition in an encoder is equivalent to a dynamic modification of the filterbank's analysis window. Depending on the shape of the gain modification function the frequency response of the analysis filters is altered according to the composite window function. However, it is undesirable to widen the frequency response of the
25 filterbank's low frequency filter channels because this increases the mismatch to the critical bandwidth scale.

Gain Control using hybrid filterbank has the drawback of increased computational complexity since the filterbank of the first stage has to achieve a considerable selectivity in
30 order to avoid aliasing distortions after the latter split by the second filterbank stage. Also, the cross-over frequencies between the gain control bands are fixed to one quarter of the Nyquist frequency, i.e. are 6, 12 and 18kHz for a sampling rate of 48kHz. For most signals, a first cross-over at 6kHz is too high for good performance.

35 Envelope shaping techniques contained in semi-parametric multi-channel coding solutions like MPEG Surround (STP, GES) are known to improve perceptual quality of transients

through a temporal re-shaping of the output signal or parts thereof in the decoder. However, these techniques do not perform temporal flattening prior to the encoder. Hence, the transient signal still enters the encoder with its original short time dynamics and imposes a high bitrate demand on the encoders bit budget.

5

It is an object of the present invention to provide an improved concept of audio pre-processing, audio post-processing or audio encoding, or audio decoding on the other hand.

10 This object is achieved by an audio post-processor of claim 1, an audio pre-processor of claim 32, an audio encoding apparatus of claim 53, an audio decoding apparatus of claim 55, a method of post-processing of claim 57, a method of pre-processing of claim 58, a method of encoding of claim 59, a method of audio decoding of claim 60 or a computer program of claim 61.

15

A first aspect of the present invention is an audio post-processor for post-processing an audio signal having a time-variable high frequency gain information as side information, comprising a band extractor for extracting a high frequency band of the audio signal and a low frequency band of the audio signal; a high band processor for performing a time-
20 variable modification of the high band in accordance with the time-variable high frequency gain information to obtain a processed high frequency band; and a combiner for combining the processed high frequency band and the low frequency band.

A second aspect of the present invention is an audio pre-processor for pre-processing an
25 audio signal, comprising a signal analyzer for analyzing the audio signal to determine a time-variable high frequency gain information; a band extractor for extracting a high frequency band of the audio signal and a low frequency band of the audio signal; a high band processor for performing a time-variable modification of the high band in accordance with the time-variable high frequency gain information to obtain a processed high
30 frequency band; a combiner for combining the processed high frequency band and the low frequency band to obtain a pre-processed audio signal; and an output interface for generating an output signal comprising the pre-processed audio signal and the time-variable high frequency gain information as side information.

35 A third aspect of the present invention is an audio encoding apparatus for encoding an audio signal, comprising the audio pre-processor of the first aspect, configured to

generate the output signal having the time-variable high frequency gain information as side information; a core encoder for generating a core encoded signal and core side information; and an output interface for generating an encoded signal comprising the core encoded signal, the core side information and the time-variable high frequency gain information as additional side information.

A fourth aspect of the present invention is an audio decoding apparatus, comprising an input interface for receiving an encoded audio signal comprising the core encoded signal, the core side information and the time-variable high frequency gain information as additional side information; a core decoder for decoding the core encoded signal using the core side information to obtain a decoded core signal; and a post-processor for post-processing the decoded core signal using the time-variable high frequency gain information in accordance with the second aspect above.

A fifth aspect of the present invention is a method of post-processing an audio signal having a time-variable high frequency gain information as side information, comprising extracting a high frequency band of the audio signal and a low frequency band of the audio signal; performing a time-variable modification of the high band in accordance with the time-variable high frequency gain information to obtain a processed high frequency band; and combining the processed high frequency band and the low frequency band.

A sixth aspect of the present invention is a method of pre-processing an audio signal, comprising analyzing the audio signal to determine a time-variable high frequency gain information; extracting a high frequency band of the audio signal and a low frequency band of the audio signal; performing a time-variable modification of the high band in accordance with the time-variable high frequency gain information to obtain a processed high frequency band; combining the processed high frequency band and the low frequency band to obtain a pre-processed audio signal; and generating an output signal comprising the pre-processed audio signal and the time-variable high frequency gain information as side information.

A seventh aspect of the present invention is a method of encoding an audio signal, comprising the method of audio pre-processing of the sixth aspect, configured to generate the output signal have the time-variable high frequency gain information as side information; generating a core encoded signal and core side information; and generating

an encoded signal comprising the core encoded signal, the core side information, and the time-variable high frequency gain information as additional side information.

5 An eighth aspect of the present invention is a method of audio decoding, comprising receiving an encoded audio signal comprising a core encoded signal, core side information and the time-variable high frequency gain information as additional side information; decoding the core encoded signal using the core side information to obtain a decoded core signal; and post-processing the decoded core signal using the time-variable high frequency gain information in accordance with the fifth aspect.

10

A ninth aspect of the present invention is related to a computer program or a non-transitory storage medium having stored thereon the computer program for performing, when running on a computer or a processor, any one of the methods in accordance with the fifth, sixth, seventh or the eighth aspect above.

15

The present invention provides a band-selective high frequency processing such as a selective attenuation in a pre-processor or a selective amplification in a post-processor in order to selectively encode a certain class of signals such as transient signals with a time-variable high frequency gain information for the high band. Thus, the pre-processed signal is a signal having the additional side information in the form of straightforward time-variable high frequency gain information and the signal itself, so that a certain class of signals, such as transient signals, does not occur anymore in the pre-processed signal or only occur to a lesser degree. In the audio post-processing, the original signal shape is recovered by performing the time-variable multiplication of the high frequency band in accordance with the time-variable high frequency gain information associated with the audio signal as side information so that, in the end, i.e., subsequent to a chain consisting of pre-processing, coding, decoding and post-processing, the listener does not perceive substantial differences to the original signal and, particularly, does not perceive a signal having a reduced transient nature, although the inner core encoder/core decoder blocks wherein the position to process a less-transient signal which has resulted, for the encoder processing, in a reduced amount of necessary bits on the one hand and an increased audio quality on the other hand, since the hard-to-encode class of signals has been removed from the signal before the encoder actually started its task. However, this removal of the hard-to-encode signal portions does not result in a reduced audio quality, since these signal portions are reconstructed by the audio post-processing subsequent to the decoder operation.

20

25

30

35

In preferred embodiments, the pre-processor also amplifies parts slightly quieter than the average background level and the post-processor attenuates them. This additional processing is potentially useful both for individual strong attacks and for parts between
5 consecutive transient events.

Subsequently, particular advantages of preferred embodiments are outlined.

HREP (High Resolution Envelope Processing) is a tool for improved coding of signals that
10 predominantly consist of many dense transient events, such as applause, rain drop sounds, etc. At the encoder side, the tool works as a pre-processor with high temporal resolution before the actual perceptual audio codec by analyzing the input signal, attenuating and thus temporally flattening the high frequency part of transient events, and generating a small amount of side information (1-4 kbps for stereo signals). At the
15 decoder side, the tool works as a post-processor after the audio codec by boosting and thus temporally shaping the high frequency part of transient events, making use of the side information that was generated during encoding. The benefits of applying HREP are two-fold: HREP relaxes the bitrate demand imposed on the encoder by reducing short time dynamics of the input signal; additionally, HREP ensures proper envelope restoration
20 in the decoder's (up-)mixing stage, which is all the more important if parametric multi-channel coding techniques have been applied within the codec.

Furthermore, the present invention is advantageous in that it enhances the coding performance for applause-like signals by using appropriate signal processing methods, for
25 example, in the pre-processing on the one hand or the post-processing on the other hand.

A further advantage of the present invention is that the inventive high resolution envelope processing (HREP), i.e., the audio pre-processing or the audio post-processing solves prior art problems by performing a pre-flattening prior to the encoder or a corresponding
30 inverse flattening subsequent to a decoder.

Subsequently, characteristic and novel features of embodiments of the present invention directed to an HREP signal processing is summarized and unique advantages are described.

35

HREP processes audio signals in just two frequency bands which are split by filters. This makes the processing simple and of low computational and structural complexity. Only the high band is processed, the low band passes through in an unmodified way.

- 5 These frequency bands are derived by low pass filtering of the input signal to compute the first band. The high pass (second) band is simply derived by subtracting the low pass component from the input signal. In this way, only one filter has to be calculated explicitly rather than two which reduces complexity. Alternatively, the high pass filtered signal can be computed explicitly and the low pass component can be derived as the difference
10 between the input signal and the high pass signal.

For supporting low complexity post-processor implementations, the following restrictions are possible

- Limitation of active HREP channels/objects
- 15 • Limitation to the maximum transmitted gain factors $g(k)$ that are non-trivial (trivial gain factors of 0dB alleviate the need for an associated DFT/iDFT pair)
- Calculation of the DFT/iDFT in an efficient split-radix 2 sparse topology.

In an embodiment the encoder or the audio pre-processor associated with the core
20 encoder is configured to limit the maximum number of channels or objects where HREP is active at the same time, or the decoder or the audio post-processor associated with the core decoder is configured to only perform a postprocessing with the maximum number of channels or objects where HREP is active at the same time. A preferred number for the limitation of active channels or objects is 16 and an even more preferred is 8.

25 In a further embodiment the HREP encoder or the audio pre-processor associated with the core encoder is configured to limit the output to a maximum of non-trivial gain factors or the decoder or the audio post-processor associated with the core decoder is configured such that trivial gain factors of value "1" do not compute a DFT/iDFT pair, but pass
30 through the unchanged (windowed) time domain signal. A preferred number for the limitation of non-trivial gain factors is 24 and an even more preferred is 16 per frame and channel or object.

In a further embodiment the HREP encoder or the audio pre-processor associated with
35 the core encoder is configured to calculate the DFT/iDFT in an efficient split-radix 2

sparse topology or the decoder or the audio post-processor associated with the core decoder is configured to also calculate the DFT/iDFT in an efficient split-radix 2 sparse topology.

- 5 The HREP low pass filter can be implemented efficiently by using a sparse FFT algorithm. Here, an example is given starting from a $N=8$ point decimation-in-time radix-2 FFT topology, where only $X(0)$ and $X(1)$ are needed for further processing; consequently, $E(2)$ and $E(3)$ and $O(2)$ and $O(3)$ are not needed; next, imagine both $N/2$ -point DFTs being further subdivided into two $N/4$ -point DFTs + subsequent butterflies each. Now one can
10 repeat the above described omissions in an analogous way, etc., as illustrated in Fig. 15.
- In contrast to a gain control scheme based on hybrid filterbanks (where the processing band cross-over frequencies are dictated by the first filterbank stage, and are practically tied to power-of-two fractions of the Nyquist frequency), the split-frequency of HREP can/could be adjusted freely by adapting the filter. This enables optimal adaptation to the
15 signal characteristics and psychoacoustic requirements.

In contrast to a gain control scheme based on hybrid filterbanks there is no need for long filters to separate processing bands in order to avoid aliasing problems after the second filterbank stage. This is possible because HREP is a stand-alone pre-/post-processor
20 which does not have to operate with a critically-sampled filterbank.

In contrast to other gain control schemes, HREP adapts dynamically to the local statistics of the signal (computing a two-sided sliding mean of the input high frequency background energy envelope). It reduces the dynamics of the input signal to a certain fraction of its
25 original size (so-called alpha factor). This enables a 'gentle' operation of the scheme without introducing artifacts by undesirable interaction with the audio codec.

In contrast to other gain control schemes, HREP is able to compensate for the additional loss in dynamics by a low bitrate audio codec by modeling this as "losing a certain fraction
30 of energy dynamics" (so-called beta factor) and reverting this loss.

The HREP pre-/post-processor pair is (near) perfectly reconstructing in the absence of quantization (i.e. without a codec).

- 35 To achieve this, the post-processor uses an adaptive slope for the splitting filter depending on the high frequency amplitude weighting factor, and corrects the interpolation

error that occurs in reverting the time-variant spectral weights applied to overlapping T/F transforms by applying a correction factor in time domain.

5 HREP implementations may contain a so-called Meta Gain Control (MGC) that gracefully controls the strength of the perceptual effect provided by HREP processing and can avoid artifacts when processing non-applause signals. Thus, it alleviates the accuracy requirements of an external input signal classification to control the application of HREP.

10 Mapping of applause classification result onto MGC and HREP settings.

HREP is a stand-alone pre-/post-processor which embraces all other coder components including bandwidth extension and parametric spatial coding tools.

15 HREP relaxes the requirements on the low bitrate audio coder through pre-flattening of the high frequency temporal envelope. Effectively, fewer short blocks will be triggered in the coder and fewer active TNS filters will be required.

20 HREP improves also on parametric multi-channel coding by reducing cross talk between the processed channels that normally happens due to limited temporal spatial cue resolution.

Codec topology: interaction with TNS/TTS, IGF and stereo filling

25 Bitstream format: HREP signaling

Preferred embodiments of the present invention are subsequently described in the context of the attached figures, in which:

30 Fig. 1 illustrates an audio post-processor in accordance with an embodiment;

Fig. 2 illustrates a preferred implementation of the band extractor of Fig. 1;

35 Fig. 3a is a schematic representation of the audio signal having a time-variable high frequency gain information as side information;

- Fig. 3b is a schematic representation of a processing by the band extractor, the high band processor or the combiner with overlapping blocks having an overlapping region;
- 5 Fig. 3c illustrates an audio post-processor having an overlap adder;
- Fig. 4 illustrates a preferred implementation of the band extractor of Fig. 1;
- Fig. 5a illustrates a further preferred implementation of the audio post-processor;
- 10 Fig. 5b illustrates a preferred embedding of the audio post-processor (HREP) in the framework of an MPEG-H 3D audio decoder;
- Fig. 5c illustrates a further preferred embedding of the audio post-processor (HREP) in the framework of an MPEG-H 3D audio decoder;
- 15 Fig. 6a illustrates a preferred embodiment of the side information containing corresponding position information;
- 20 Fig. 6b illustrates a side information extractor combined with a side information decoder for an audio post-processor;
- Fig. 7 illustrates an audio pre-processor in accordance with a preferred embodiment;
- 25 Fig. 8a illustrates a flow chart of steps performed by the audio pre-processor;
- Fig. 8b illustrates a flow chart of steps performed by the signal analyzer of the audio pre-processor;
- 30 Fig. 8c illustrates a flow chart of procedures performed by the signal analyzer, the high band processor and the output interface of the audio pre-processor;
- Fig. 8d illustrates a procedure performed by the audio pre-processor of Fig. 7;
- 35 Fig. 9a illustrates an audio encoding apparatus with an audio pre-processor in accordance with an embodiment;

- Fig. 9b illustrates an audio decoding apparatus comprising an audio post-processor;
- Fig. 9c illustrates a preferred implementation of an audio pre-processor;
- 5 Fig. 10a illustrates an audio encoding apparatus with a multi-channel/multi-object functionality;
- Fig. 10b illustrates an audio decoding apparatus with a multi-channel/multi object
10 functionality;
- Fig. 10c illustrates a further implementation of an embedding of the pre-processor and the post-processor into an encoding/decoding chain;
- 15 Fig. 11 illustrates a high frequency temporal envelope of a stereo applause signal;
- Fig. 12 illustrates a functionality of a gain modification processing;
- Fig. 13a illustrates a filter-based gain control processing;
- 20 Fig. 13b illustrates different filter functionalities for the corresponding filter of Fig. 13a;
- Fig. 14 illustrates a gain control with hybrid filter bank;
- 25 Fig. 15 illustrates an implementation of a sparse digital Fourier transform implementation;
- Fig. 16 illustrates a listening test overview;
- 30 Fig. 17a illustrates absolute MUSHRA scores for 128 kbps 5.1ch test;
- Fig. 17b illustrates different MUSHRA scores for 128 kbps 5.1ch test;
- Fig. 17c illustrates absolute MUSHRA scores for 128 kbps 5.1ch test applause signals;
- 35 Fig. 17d illustrates different MUSHRA scores for 128 kbps 5.1ch test applause signals;

Fig. 17e illustrates absolute MUSHRA scores for 48 kbps stereo test;

Fig. 17f illustrates different MUSHRA scores for 48 kbps stereo test;

5

Fig. 17g illustrates absolute MUSHRA scores for 128 kbps stereo test; and

Fig. 17h illustrates different MUSHRA scores for 128 kbps stereo test.

10 Fig. 1 illustrates a preferred embodiment of an audio post-processor 100 for post-processing an audio signal 102 having a time-variable high frequency gain information 104 as side information 106 illustrated in Fig. 3a. The audio post-processor comprises a band extractor 110 for extracting a high frequency band 112 of the audio signal 102 and a low frequency band 114 of the audio signal 102. Furthermore, the audio post-processor in accordance with this embodiment comprises a high band processor 120 for performing a time-variable modification of the high frequency band 112 in accordance with the time-variable high frequency gain information 104 to obtain a processed high frequency band 122. Furthermore, the audio post-processor comprises a combiner 130 for combining the processed high frequency band 122 and the low frequency band 114.

20

Preferably, the high band processor 120 performs a selective amplification of a high frequency band in accordance with the time-variable high frequency gain information for this specific band. This is to undo or reconstruct the original high frequency band, since the corresponding high frequency band has been attenuated before in an audio pre-processor such as the audio pre-processor of Fig. 7 that will be described later on.

25

Particularly, in the embodiment, the band extractor 110 is provided, at an input thereof, with the audio signal 102 as extracted from the audio signal having associated side information. Further, an output of the band extractor is connected to an input of the combiner. Furthermore, a second input of the combiner is connected to an output of the high band processor 120 to feed the processed high frequency band 122 into the combiner 130. Furthermore, further output of the band extractor 110 is connected to an input of the high band processor 120. Furthermore, the high band processor additionally has a control input for receiving the time-variable high frequency gain information as illustrated in Fig. 1.

35

Fig. 2 illustrates a preferred implementation of the band extractor 110. Particularly, the band extractor 110 comprises a low pass filter 111 that, at its output, delivers the low frequency band 114. Furthermore, the high frequency band 112 is generated by subtracting the low frequency band 114 from the audio signal 102, i.e., the audio signal that has been input into the low pass filter 111. However, the subtractor 113 can perform some kind of pre-processing before the actual typically sample-wise subtraction as will be shown with respect to the audio signal windower 121 in Fig. 4 or the corresponding block 121 in Fig. 5a. Thus, the band extractor 110 may comprise, as illustrated in Fig. 2, a low pass filter 111 and the subsequently connected subtractor 113, i.e., subtractor 113 having an input being connected to an output of the low pass filter 111 and having a further input being connected to the input of the low pass filter 111.

Alternatively, however, the band extractor 110 can also be implemented by actually using a high pass filter and by subtracting the high pass output signal or high frequency band from the audio signal to get the low frequency band. Or, alternatively, the band extractor can be implemented without any subtractor, i.e., by a combination of a low pass filter and a high pass filter in the way of a two-channel filterbank, for example. Preferably, the band extractor 110 of Fig. 1 (or Fig. 2) is implemented to extract only two bands, i.e., a single low frequency band and a single high frequency band while these bands together span the full frequency range of the audio signal.

Preferably, a cutoff or corner frequency of the low frequency band extracted by the band extractor 110 is between $1/8$ and $1/3$ of a maximum frequency of the audio signal and preferably equal to $1/6$ of the maximum frequency of the audio signal.

Fig. 3a illustrates a schematic representation of the audio signal 102 having useful information in the sequence of blocks 300, 301, 302, 303 where, for illustration reasons, block 301 is considered as a first block of sampling values, and block 302 is considered to be a second later block of sampling values of the audio signal. Block 300 precedes the first block 301 in time and block 303 follows the block 302 in time and the first block 301 and the second block 302 are adjacent in time to each other. Furthermore, as illustrated at 106 in Fig. 3a, each block has associated therewith side information 106 comprising, for the first block 301, the first gain information 311 and comprising, for the second block, second gain information 312.

Fig. 3b illustrates a processing of the band extractor 110 (and the high band processor 120 and the combiner 130) in overlapping blocks. Thus, the window 313 used for calculating the first block 301 overlaps with window 314 used for extracting the second block 302 and both windows 313 and 314 overlap within an overlap range 321.

5

Although the scale in Figs. 3a and 3b outline that the length of each block is half the size of the length of a window, the situation can also be different, i.e., that the length of each block is the same size as a window used for windowing the corresponding block. Actually, this is the preferred implementation for these subsequent preferred embodiments illustrated in Fig. 4 or, particularly, Fig. 5a for the post-processor or Fig. 9c for the pre-processor.

10

Then, the length of the overlapping range 321 is half the size of a window corresponding to half the size or length of a block of sampling values.

15

Particularly, the time-variable high frequency gain information is provided for a sequence 300 to 303 of blocks of sampling values of the audio signal 102 so that the first block 301 of sampling values has associated therewith the first gain information 311 and the second later block 302 of sampling values of the audio signal has a different second gain information 312, wherein the band extractor 110 is configured to extract, from the first block 301 of sampling values, a first low frequency band and a first high frequency band and to extract, from the second block 302 of sampling values, a second low frequency band and a second high frequency band. Furthermore, the high band processor 120 is configured to modify the first high frequency band using the first gain information 311 to obtain the first processed high frequency band and to modify the second high frequency band using the second gain information 312 to obtain a second processed high frequency band. Furthermore, the combiner 130 is then configured to combine the first low frequency band and the first processed high frequency band to obtain a first combined block and to combine the second low frequency band and the second processed high frequency band to obtain a second combined block.

20

25

30

As illustrated in Fig. 3c, the band extractor 110, the high band processor 120 and the combiner 130 are configured to operate with the overlapping blocks illustrated in Fig. 3b. Furthermore, the audio post-processor 100 furthermore comprises an overlap-adder 140 for calculating a post-processed portion by adding audio samples of a first block 301 and audio samples of a second block 302 in the block overlap range 321. Preferably, the

35

overlap adder 140 is configured for weighting audio samples of a second half of a first block using a decreasing or fade-out function and for weighting a first half of a second block subsequent to the first block using a fade-in or increasing function. The fade-out function and the fade-in function can be linear or non-linear functions that are
5 monotonically increasing for the fade-in function and monotonically decreasing for the fade-out function.

At the output of the overlap-adder 140, there exists a sequence of samples of the post-processed audio signal as, for example, illustrated in Fig. 3a, but now without any side
10 information, since the side information has been “consumed” by the audio post-processor 100.

Fig. 4 illustrates a preferred implementation of the band extractor 110 of the audio post-processor illustrated in Fig. 1 or, alternatively, of the band extractor 210 of audio pre-processor 200 of Fig. 7. Both, the band extractor 110 of Fig. 1 or the band extractor 210 of
15 Fig. 7 can be implemented in the same way as illustrated in Fig. 4 or as illustrated in Fig. 5a for the post-processor or Fig. 9c for the pre-processor. In an embodiment, the audio post-processor comprises the band extractor that has, as certain features, an analysis windower 115 for generating a sequence of blocks of sampling values of the audio signal using an analysis window, where the blocks are time-overlapping as illustrated in Fig. 3b
20 by an overlapping range 321. Furthermore, the band extractor 110 comprises a DFT processor 116 for performing a discrete Fourier transform for generating a sequence of blocks of spectral values. Thus, each individual block of sampling values is converted into a spectral representation that is a block of spectral values. Therefore, the same number of
25 blocks of spectral values is generated as if they were blocks of sampling values.

The DFT processor 116 has an output connected to an input of a low pass shaper 117. The low pass shaper 117 actually performs the low pass filtering action, and the output of the low pass shaper 117 is connected to a DFT inverse processor 118 for generating a
30 sequence of blocks of low pass time domain sampling values. Finally, a synthesis windower 119 is provided at an output of the DFT inverse processor for windowing the sequence of blocks of low pass time domain sampling values using a synthesis window. The output of the synthesis windower 119 is a time domain low pass signal. Thus, blocks 115 to 119 correspond to the “low pass filter” block 111 of Fig. 2, and blocks 121 and 113
35 correspond to the “subtractor” 113 of Fig. 2. Thus, in the embodiment illustrated in Fig. 4, the band extractor further comprises the audio signal windower 121 for windowing the

audio signal 102 using the analysis window and the synthesis window to obtain a sequence of windowed blocks of audio signal values. Particularly, the audio signal windower 121 is synchronized with the analysis windower 115 and/or the synthesis windower 119 so that the sequence of blocks of low pass time domain sampling values output by the synthesis windower 119 is time synchronous with the sequence of windowed blocks of audio signal values output by block 121, which is the full band signal.

However, the full band signal is now windowed using the audio signal windower 121 and, therefore, a sample-wise subtraction is performed by the sample-wise subtractor 113 in Fig. 4 to finally obtain the high pass signal. Thus, the high pass signal is available, additionally, in a sequence of blocks, since the sample-wise subtraction 113 has been performed for each block.

Furthermore, the high band processor 120 is configured to apply the modification to each sample of each block of the sequence of blocks of high pass time domain sampling values as generated by block 110 in Fig. 3c. Preferably, the modification for a sample of a block depends on, again, information of a previous block and, again, information of the current block, or, alternatively or additionally, again, information of the current block and, again, information of the next block. Particularly, and preferably, the modification is done by a multiplier 125 of Fig. 5a and the modification is preceded by an interpolation correction block 124. As illustrated in Fig. 5a, the interpolation correction is done between the preceding gain values $g[k-1]$, $g[k]$ and again factor $g[k+1]$ of the next block following the current block.

Furthermore, as stated, the multiplier 125 is controlled by a gain compensation block 126 being controlled, on the one hand, by β_factor 500 and, on the other hand, by the gain factor $g[k]$ 104 for the current block. Particularly, the β_factor is used to calculate the actual modification applied by multiplier 125 indicated as $1/gc[k]$ from the gain factor $g[k]$ associated with the current block.

Thus, the β_factor accounts for an additional attenuation of transients which is approximately modelled by this β_factor , where this additional attenuation of transient events is a side effect of either an encoder or a decoder that operates before the post-processor illustrated in Fig. 5a.

The pre-processing and post-processing are applied by splitting the input signal into a low-pass (LP) part and a high-pass (HP) part. This can be accomplished: a) by using FFT to compute the LP part or the HP part, b) by using a zero-phase FIR filter to compute the LP part or the HP part, or c) by using an IIR filter applied in both directions, achieving zero-phase, to compute the LP part or the HP part. Given the LP part or the HP part, the other part can be obtained by simple subtraction in time domain. A time-dependent scalar gain is applied to the HP part, which is added back to the LP part to create the pre-processed or post-processed output.

10 **Splitting the signal into a LP part and a HP part using FFT (Figs. 5a, 9c)**

In the proposed implementation, the FFT is used to compute the LP part. Let the FFT transform size be N , in particular $N = 128$. The input signal s is split into blocks of size N , which are half-overlapping, producing input blocks $ib[k][i] = s[k \times \frac{N}{2} + i]$, where k is the block index and i is the sample position in the block k . A window $w[i]$ is applied (115, 215) to $ib[k]$, in particular the sine window, defined as

$$w[i] = \sin \frac{\pi(i + 0.5)}{N}, \text{ for } 0 \leq i < N,$$

20 and after also applying FFT (116, 216), the complex coefficients $c[k][f]$ are obtained as

$$c[k][f] = \text{FFT}(w[i] \times ib[k][i]), \text{ for } 0 \leq f \leq \frac{N}{2}.$$

On the encoder side (Fig. 9c) (217a), in order to obtain the LP part, an element-wise multiplication (217a) of $c[k][f]$ with the processing shape $ps[f]$ is applied, which consists of the following:

$$ps[f] = \begin{cases} 1, & \text{for } 0 \leq f < lp_size \\ 1 - \frac{f - lp_size + 1}{tr_size + 1}, & \text{for } lp_size \leq f < lp_size + tr_size \\ 0, & \text{for } lp_size + tr_size \leq f \leq \frac{N}{2} \end{cases}$$

30 The $lp_size = \text{lastFFTLLine}[sig] + 1 - \text{transitionWidthLines}[sig]$ parameter represents the width in FFT lines of the low-pass region, and the $tr_size = \text{transitionWidthLines}[sig]$

parameter represents the width in FFT lines of the transition region. The shape of the proposed processing shape is linear, however any arbitrary shape can be used.

The LP block $lpb[k]$ is obtained by applying IFFT (218) and windowing (219) again as

5

$$lpb[k][i] = w[i] \times IFFT(ps[f] \times c[k][f]), \text{ for } 0 \leq i < N.$$

The above equation is valid for the encoder/pre-processor of Fig. 9c. For the decoder or post-processor, the adaptive processing shape $rs[f]$ is used instead of $ps[f]$.

10

The HP block $hpb[k]$ is then obtained by simple subtraction (113, 213) in time domain as

$$hpb[k][i] = in[k][i] \times w^2[i] - lpb[k][i], \text{ for } 0 \leq i < N.$$

15 The output block $ob[k]$ is obtained by applying the scalar gain $g[k]$ to the HP block as (225) (230)

$$ob[k][i] = lpb[k][i] + g[k] \times hpb[k][i]$$

20 The output block $ob[k]$ is finally combined using overlap-add with the previous output block $ob[k-1]$ to create $\frac{N}{2}$ additional final samples for the pre-processed output signal o as

$$o\left[k \times \frac{N}{2} + j\right] = ob[k-1][j + \frac{N}{2}] + ob[k][j], \text{ with } j = \{0, \dots, \frac{N}{2} - 1\}.$$

25

All processing is done separately for each input channel, which is indexed by sig .

Adaptive reconstruction shape on the post-processing side (Fig. 5a)

30 On the decoder side, in order to get perfect reconstruction in the transition region, an adaptive reconstruction shape $rs[f]$ (117b) in the transition region must be used, instead of the processing shape $ps[f]$ (217b) used at the encoder side, depending on the processing shape $ps[f]$ and $g[k]$ as

$$rs[f] = 1 - (1 - ps[f]) \times \frac{g[k]}{1 + (g[k] - 1) \times (1 - ps[f])}$$

In the LP region, both $ps[f]$ and $rs[f]$ are one, in the HP region both $ps[f]$ and $rs[f]$ are zero, they only differ in the transition region. Moreover, when $g[k] = 1$, then one has
5 $rs[f] = ps[f]$.

The adaptive reconstruction shape can be deducted by ensuring that the magnitude of a FFT line in the transition region is restored after post-processing, which gives the relation

$$10 \quad (ps[f] + (1 - ps[f]) \times g[k]) \times (rs[f] + (1 - rs[f]) \times \frac{1}{g[k]}) = 1.$$

The processing is similar to the pre-processing side, except $rs[f]$ is used instead of $ps[f]$ as

$$15 \quad lpb[k][i] = w[i] \times IFFT(rs[f] \times c[k][f]), \text{ with } i = \{0, \dots, N - 1\}$$

and the output block $ob[k][i]$ is computed using the inverse of the scalar gain $g[k]$ as
(125)

$$20 \quad ob[k][i] = lpb[k][i] + \frac{1}{g[k]} \times hpb[k][i].$$

Interpolation correction (124) on the post-processing side (Fig. 5a)

The first half of the output block k contribution to the final pre-processed output is given by
25 $o[k \times \frac{N}{2} + j] = ob[k - 1][j + \frac{N}{2}] + ob[k][j]$, with $j = \{0, \dots, \frac{N}{2}\}$. Therefore, the gains $g[k - 1]$ and $g[k]$ applied on the pre-processing side are implicitly interpolated due to the windowing and overlap-add operations. The magnitude of each FFT line in the HP region is effectively multiplied in the time domain by the scalar factor $g[k - 1] \times w^2[j + \frac{N}{2}] + g[k] \times w^2[j]$.
30

Similarly, on the post-processing side, the magnitude of each FFT line in the HP region is effectively multiplied in the time domain by the factor

$$\frac{1}{g[k-1]} \times w^2 \left[j + \frac{N}{2} \right] + \frac{1}{g[k]} \times w^2[j].$$

In order to achieve perfect reconstruction, the product of the two previous terms,

$$\begin{aligned} \text{corr}[j] = & \left(g[k-1] \times w^2 \left[j + \frac{N}{2} \right] + g[k] \times w^2[j] \right) \times \left(\frac{1}{g[k-1]} \times w^2 \left[j + \frac{N}{2} \right] + \right. \\ & \left. \frac{1}{g[k]} \times w^2[j] \right), \end{aligned}$$

which represents the overall time domain gain at position j for each FFT line in the HP region, should be normalized in the first half of the output block k as

$$\text{ob}[k][j] = \text{lpb}[k][j] + \frac{1}{g[k]} \times \text{hpb}[k][j] \times \frac{1}{\text{corr}[j]}.$$

The value of $\text{corr}[j]$ can be simplified and rewritten as

$$\text{corr}[j] = 1 + \left(\frac{g[k-1]}{g[k]} + \frac{g[k]}{g[k-1]} - 2 \right) \times w^2[j] \times (1 - w^2[j]), \text{ for } 0 \leq j < \frac{N}{2}.$$

The second half of the output block k contribution to the final pre-processed output is given by $o[(k+1) \times \frac{N}{2} + j] = \text{ob}[k][j + \frac{N}{2}] + \text{ob}[k+1][j]$, and the interpolation correction can be written based on the gains $g[k]$ and $g[k+1]$ as

$$\text{corr} \left[j + \frac{N}{2} \right] = 1 + \left(\frac{g[k]}{g[k+1]} + \frac{g[k+1]}{g[k]} - 2 \right) \times w^2[j] \times (1 - w^2[j]), \text{ for } 0 \leq j < \frac{N}{2}.$$

The updated value for the second half of the output block k is given by

$$\text{ob}[k] \left[j + \frac{N}{2} \right] = \text{lpb}[k] \left[j + \frac{N}{2} \right] + \frac{1}{g[k]} \times \text{hpb}[k] \left[j + \frac{N}{2} \right] \times \frac{1}{\text{corr}[j + \frac{N}{2}]}.$$

Gain computation on the pre-processing side (Fig. 9c)

At the pre-processing side, the HP part of block k , assumed to contain a transient event, is adjusted using the scalar gain $g[k]$ in order to make it more similar to the background in its neighborhood. The energy of the HP part of block k will be denoted by $\text{hp_e}[k]$ and the

average energy of the HP background in the neighborhood of block k will be denoted by $hp_bg_e[k]$.

The parameter $\alpha \in [0, 1]$, which controls the amount of adjustment is defined as

5

$$g_{float}[k] = \begin{cases} \frac{\alpha \times hp_bg_e[k] + (1 - \alpha) \times hp_e[k]}{hp_e[k]}, & \text{when } hp_e[k] \geq T_{quiet}, \\ 1, & \text{otherwise} \end{cases}$$

The value of $g_{float}[k]$ is quantized and clipped to the range allowed by the chosen value of the extendedGainRange configuration option to produce the gain index $gainIdx[k][sig]$

10 as

$$g_{idx} = \lfloor \log_2(4 \times g_{float}[k]) + 0.5 \rfloor + GAIN_INDEX_0dB,$$

$$gainIdx[k][sig] = \min(\max(0, g_{idx}), 2 \times GAIN_INDEX_0dB - 1).$$

15

The value $g[k]$ used for the processing is the quantized value, defined at the decoder side as

$$g[k] = 2^{\frac{gainIdx[k][sig] - GAIN_INDEX_0dB}{4}}.$$

20

When α is 0, the gain has value $g_{float}[k] = 1$, therefore no adjustment is made, and when α is 1, the gain has value $g_{float}[k] = hp_bg_e[k]/hp_e[k]$, therefore the adjusted energy is made to coincide with the average energy of the background. The above relation can be rewritten as

25

$$g_{float}[k] \times hp_e[k] = hp_bg_e[k] + (1 - \alpha) \times (hp_e[k] - hp_bg_e[k]),$$

indicating that the variation of the adjusted energy $g_{float}[k] \times hp_e[k]$ around the corresponding average energy of the background $hp_bg_e[k]$ is reduced with a factor of $(1 - \alpha)$. In the proposed system, $\alpha = 0.75$ is used, thus the variation of the HP energy of each block around the corresponding average energy of the background is reduced to 25 % of the original.

30

Gain compensation (126) on the post-processing side (Fig. 5a)

The core encoder and decoder introduce additional attenuation of transient events, which is approximately modeled by introducing an extra attenuation step, using the parameter $\beta \in [0, 1]$ depending on the core encoder configuration and the signal characteristics of the frame, as

$$gc_{float}[k] = \frac{\beta \times hp_bg_e[k] + (1 - \beta) \times [g_{float}[k] \times hp_e[k]]}{hp_e[k]}$$

indicating that, after passing through the core encoder and decoder, the variation of the decoded energy $gc_{float}[k] \times hp_e[k]$ around the corresponding average energy of the background $hp_bg_e[k]$ is further reduced with an additional factor of $(1 - \beta)$.

Using just $g[k]$, α , and β , it is possible to compute an estimate of $gc[k]$ at the decoder side as

$$gc[k] = (1 + \frac{\beta \times (1 - \alpha)}{\alpha}) \times g[k] - \frac{\beta \times (1 - \alpha)}{\alpha}$$

The parameter $\beta_factor = \frac{\beta \times (1 - \alpha)}{\alpha}$ is quantized to $\beta_factorIdx[sig]$ and transmitted as side information for each frame. The compensated gain $gc[k]$ can be computed using β_factor as

$$gc[k] = (1 + \beta_factor) \times g[k] - \beta_factor$$

Meta Gain Control (MGC)

Applause signals of live concerts etc. usually do not only contain the sound of hand claps, but also crowd shouting, pronounced whistles and stomping of the audiences' feet. Often, the artist gives an announcement during applause or instrument (handling) sounds overlap with sustained applause. Here, existing methods of temporal envelope shaping like STP or GES might impair these non-applause components if activated at the very instant of the interfering sounds. Therefore, a signal classifier assures deactivation during such signals. HREP offers the feature of so-called Meta Gain Control (MGC). MGC is used to gracefully relax the perceptual effect of HREP processing, avoiding the necessity

of very accurate input signal classification. With MGC, applauses mixed with ambience and interfering sounds of all kind can be handled without introducing unwanted artefacts.

As discussed before, a preferred embodiment additionally has a control parameter 807 or, alternatively, the control parameter β indicated at 500 in Fig. 5a. Alternatively, or additionally, the individual factors α or β as discussed before can be transmitted as additional side information, but it is preferred to have the single control parameter β that consists of β on the one hand and α on the other hand, where β is the parameter between 0 and 1 and depends on the core encoder configuration and also optionally on the signal characteristics, and additionally, the factor α determines the variation of a high frequency part energy of each block around the corresponding average energy of the background, and α is also a parameter between 0 and 1. If the number of transients in one frame is very small, like 1-2, then TNS can potentially preserve them better, and as a result the additional attenuation through the encoder and decoder for the frame may be reduced. Therefore, an advanced encoder can correspondingly reduce β slightly to prevent over-amplification.

In other words, MGC currently modifies the computed gains g (denoted here by $g_{\text{float}}[k]$) using a probability-like parameter p , like $g' = g \wedge p$, which *squeezes* the gains toward 1 before they are quantized. The β parameter is an additional mechanism to control the *expansion* of the quantized gains, however the current implementation uses a fixed value based on the core encoder configuration, such as the bitrate.

β is determined by $\beta \times (1-\alpha)/\alpha$ and is preferably calculated on the encoder-side and quantized, and the quantized β index β_{Idx} is transmitted as side information once per frame in addition to the time-variable high frequency gain information $g[k]$.

Particularly, the additional control parameter 807 such as β or β_{Idx} has a time resolution that is lower than the time resolution of the time-varying high frequency gain information or the additional control parameter is even stationary for a specific core encoder configuration or audio piece.

Preferably, the high band processor, the band extractor and the combiner operate in overlapping blocks, wherein an overlap ranges between 40% and 60% of the block length and preferably a 50% overlap range 321 is used.

In other embodiments or in the same embodiments, the block length is between 0.8ms and 5.0ms.

- 5 Furthermore, preferably or additionally, the modification performed by the high band processor 120 is an time-dependent multiplicative factor applied to each sample of a block in time domain in accordance with $g[k]$, additionally in accordance with the control parameter 500 and additionally in line with the interpolation correction as discussed in the context of block 124 of Fig. 5a.

10

Furthermore, a cutoff or corner frequency of the low frequency band is between $1/8$ and $1/3$ of a maximum frequency of the audio signal and preferably equal to $1/6$ of the maximum frequency of the audio signal.

- 15 Furthermore, the low pass shaper consisting of 117b and 117a of Fig. 5a in the preferred embodiment is configured to apply the shaping function $rs[f]$ that depends on the time-variable high frequency gain information for the corresponding block. A preferred implementation of the shaping function $rs[f]$ has been discussed before, but alternative functions can be used as well.

20

Furthermore, preferably, the shaping function $rs[f]$ additionally depends on a shaping function $ps[f]$ used in an audio pre-processor 200 for modifying or attenuating a high frequency band of the audio signal using the time-variable high frequency gain information for the corresponding block. A specific dependency of $rs[f]$ from $ps[f]$ has been discussed
25 before, with respect to Fig. 5a, but other dependencies can be used as well.

- Furthermore, as discussed before with respect to block 124 of Fig. 5a, the modification for a sample of a block additionally depends on a windowing factor applied for a certain sample as defined by the analysis window function or the synthesis window function as
30 discussed before, for example, with respect to the correction factor that depends on a window function $w[j]$ and even more preferably from a square of a window factor $w[j]$.

- As stated before, particularly with respect to Fig. 3b, the processing performed by the band extractor, the combiner and the high band processor is performed in overlapping
35 blocks so that a latter portion of an earlier block is derived from the same audio samples of the audio signal as an earlier portion of a later block being adjacent in time to the earlier

block, i.e., the processing is performed within and using the overlapping range 321. This overlapping range 321 of the overlapping blocks 313 and 314 is equal to one half of the earlier block and the later block has the same length as the earlier block with respect to a number of sample values and the post-processor additionally comprises the overlap adder 5 140 for performing the overlap add operation as illustrated in Fig. 3c.

Particularly, the band extractor 110 is configured to apply the slope of splitting filter 111 between a stop range and a pass range of the splitting filter to a block of audio samples, wherein this slope depends on the time-variable high frequency gain information for the 10 block of samples. A preferred slope is given with respect to the slope $rs[f]$ that depends on the gain information $g[k]$ as defined before and as discussed in the context of Fig. 5a, but other dependencies can be useful as well.

Generally, the high frequency gain information preferably has the gain values $g[k]$ for a 15 current block k , where the slope is increased stronger for a higher gain value compared to an increase of the slope for a lower gain value.

Fig. 6a illustrates a more detailed representation of the side information 106 of Fig. 3. Particularly, the side information comprises a sequence of gain indices 601, gain precision 20 information 602, a gain compensation information 603 and a compensation precision information 604.

Preferably, the audio post-processor comprises a side information extractor 610 for extracting the audio signal 102 and the side information 106 from an audio signal with side 25 information and the side information is forwarded to a side information decoder 620 that generates and calculates a decoded gain 621 and/or a decoded gain compensation value 622 based on the corresponding gain precision information and the corresponding compensation precision information.

30 Particularly, the precision information determines a number of different values, where a high gain precision information defines a greater number of values that the gain index can have compared to a lower gain precision information indicating a lower number of values that a gain value can have.

35 Thus, a high precision gain information may indicate a higher number of bits used for transmitting a gain index compared to a lower gain precision information indicating a lower

number of bits used for transmitting the gain information. The high precision information can indicate 4 bits (16 values for the gain information) and the lower gain information can be only 3 bits (8 values) for the gain quantization. Therefore, the gain precision information can, for example, be a simple flag indicated as "extendedGainRange". In the latter case, the configuration flag extendedGainRange does not indicate accuracy or precision information but whether the gains have a normal range or an extended range. The extended range contains all the values in the normal range and, in addition, smaller and larger values than are possible using the normal range. The extended range that can be used in certain embodiments potentially allows to apply a more intense pre-processing effect for strong transient events, which would be otherwise clipped to the normal range.

Similarly, for the beta factor precision, i.e., for the gain compensation precision information, a flag can be used as well, which outlines whether the beta_factor indices use 3 bits or 4 bits, and this flag may be termed extendedBetaFactorPrecision.

Preferably, the FFT processor 116 is configured to perform a block-wise discrete Fourier transform with a block length of N sampling values to obtain a number of spectral values being lower than a number of N/2 complex spectral values by performing a sparse discrete Fourier transform algorithm, in which calculations of branches for spectral values above a maximum frequency are skipped, and the band extractor is configured to calculate the low frequency band signal by using the spectral values up to a transition start frequency range and by weighting the spectral values within the transition frequency range, wherein the transition frequency range only extends until the maximum frequency or a frequency being smaller than the maximum frequency.

This procedure is illustrated in Fig. 15, for example, where certain butterfly operations are illustrated. An example is given starting from N=8 point decimation-in-time radix-2 FFT topology, where only X(0) and X(1) are needed for further processing; consequently, E(2) and E(3) and O(2) and O(3) are not needed. Next, imagine both N/2 point DFTs being further subdivided into two N/4 point DFT and subsequent butterflies each. Now one can repeat the above described omission in an analogous way as illustrated in Fig. 15.

Subsequently, the audio pre-processor 200 is discussed in more detail with respect to Fig. 7.

35

The audio pre-processor 200 comprises a signal analyzer 260 for analyzing the audio signal 202 to determine a time-variable high frequency gain information 204.

Additionally, the audio pre-processor 200 comprises a band extractor 210 for extracting a high frequency band 212 of the audio signal 202 and a low frequency band 214 of the audio signal 202. Furthermore, a high band processor 220 is provided for performing a time-variable modification of the high frequency band 212 in accordance with the time-variable high frequency gain information 204 to obtain a processed high frequency band 222.

The audio pre-processor 200 additionally comprises a combiner 230 for combining the processed high frequency band 222 and the low frequency band 214 to obtain a pre-processed audio signal 232. Additionally, an output interface 250 is provided for generating an output signal 252 comprising the pre-processed audio signal 232 and the time-variable high frequency gain information 204 as side information 206 corresponding to the side information 106 discussed in the context of Fig. 3.

Preferably, the signal analyzer 260 is configured to analyze the audio signal to determine a first characteristic in a first time block 301 as illustrated by block 801 of Fig. 8a and a second characteristic in a second time block 302 of the audio signal, the second characteristic being more transient than the first characteristic as illustrated in block 802 of Fig. 8a.

Furthermore, analyzer 260 is configured to determine a first gain information 311 for the first characteristic and a second gain information 312 for the second characteristic as illustrated at block 803 in Fig. 8a. Then, the high band processor 220 is configured to attenuate the high band portion of the second time block 302 in accordance with the second gain information stronger than the high band portion of the first time block 301 in accordance with the first gain information as illustrated in block 804 of Fig. 8a.

Furthermore, the signal analyzer 260 is configured to calculate the background measure for a background energy of the high band for one or more time blocks neighboring in time placed before the current time block or placed subsequent to the current time block or placed before and subsequent to the current time block or including the current time block or excluding the current time block as illustrated in block 805 of Fig. 8b. Furthermore, as illustrated in block 808, an energy measure for a high band of the current block is

calculated and, as outlined in block 809, a gain factor is calculated using the background measure on the one hand, and the energy measure on the other hand. Thus, the result of block 809 is the gain factor illustrated at 810 in Fig. 8b.

- 5 Preferably, the signal analyzer 260 is configured to calculate the gain factor 810 based on the equation illustrated before g_{float} , but other ways of calculation can be performed as well.

10 Furthermore, the parameter α influences the gain factor so that a variation of an energy of each block around a corresponding average energy of a background is reduced by at least 50 % and preferably by 75 %. Thus, the variation of the high pass energy of each block around the corresponding average energy of the background is preferably reduced to 25 % of the original by means of the factor α .

15 Furthermore, the meta gain control block/functionality 806 is configured to generate a control factor p . In an embodiment, the MGC block 806 uses a statistical detection method for identifying potential transients. For each block (of e.g. 128 samples), it produces a probability-like "confidence" factor p between 0 and 1. The final gain to be applied to the block is $g' = g \wedge p$, where g is the original gain. When p is zero, $g' = 1$, therefore no
20 processing is applied, and when p is one, $g' = g$, the full processing strength is applied.

MGC 806 is used to squeeze the gains towards 1 before quantization during pre-processing, to control the strength of the processing between no change and full effect. The parameter β_factor (which is an improved parameterization of parameter β) is
25 used to expand the gains after dequantization during post-processing, and one possibility is to use a fixed value for each encoder configuration, defined by the bitrate.

In an embodiment, the parameter α is fixed at 0.75. Hence, factor α is the reduction of energy variation around an average background, and it is fixed in the MPEG-H
30 implementation to 75%. The control factor p in Fig. 8b serves as the probability-like "confidence" factor p .

As illustrated in Fig. 8c, the signal analyzer is configured to quantize and clip a raw sequence of gain information values to obtain the time-variable high frequency gain
35 information as a sequence of quantized values, and the high band processor 220 is

configured to perform the time-variable modification of the high band in accordance with the sequence of quantized values rather than the non-quantized values.

5 Furthermore, the output interface 250 is configured to introduce the sequence of quantized values into the side information 206 as the time-variable high frequency gain information 204 as illustrated in Fig. 8c at block 814.

10 Furthermore, the audio pre-processor 200 is configured to determine 815 a further gain compensation value describing a loss of an energy variation introduced by a subsequently connected encoder or decoder, and, additionally, the audio pre-processor 200 quantizes 816 this further gain compensation information and introduces 817 this quantized further gain compensation information into the side information and, additionally, the signal analyzer is preferably configured to apply Meta Gain Control in a determination of the time-variable high frequency gain information to gradually reduce or gradually enhance an effect of the high band processor on the audio signal in accordance with additional control data 807.

20 Preferably, the band extractor 210 of the audio pre-processor 200 is implemented in more detail as illustrated in Fig. 4, or in Fig. 9c. Therefore, the band extractor 210 is configured to extract the low frequency band using a low pass filter device 111 and to extract a high frequency band by subtracting 113 the low frequency band from the audio signal in exactly the same way as has been discussed previously with respect to the post-processor device.

25 Furthermore, the band extractor 210, the high band processor 220 and the combiner 230 are configured to operate in overlapping blocks. The combiner 230 additionally comprises an overlap adder for calculating a post-processed portion by adding audio samples of a first block and audio samples of a second block in the block overlap range. Therefore, the overlap adder associated with the combiner 230 of Fig. 7 may be implemented in the same way as the overlap adder for the post-processor illustrated in Fig. 3c at reference numeral 130.

35 In an embodiment, for the audio pre-processor, the overlap range 320 is between 40% of a block length and 60% of a block length. In other embodiments, a block length is between 0.8ms and 5.0ms and/or the modification performed by the high band processor

220 is a multiplicative factor applied to each sample of a block in a time domain so that the result of the whole pre-processing is a signal with a reduced transient nature.

5 In a further embodiment, a cutoff or corner frequency of the low frequency band is between $1/8$ and $1/3$ of the maximum frequency range of the audio signal 202 and preferably equal to $1/6$ of the maximum frequency of the audio signal.

As illustrated, for example, in Fig. 9c and as has also been discussed with respect to the post-processor in Fig. 4, the band extractor 210 comprises an analysis windower 215 for
10 generating a sequence of blocks of sampling values of the audio signal using an analysis window, wherein these blocks are time-overlapping as illustrated at 321 in Fig. 3b. Furthermore, a discrete Fourier transform processor 216 for generating a sequence of blocks of spectral values is provided and also a subsequently connected low pass shaper 217a, 217b is provided, for shaping each block of spectral values to obtain a sequence of
15 low pass shaped blocks of spectral values. Furthermore, a discrete Fourier inverse transform processor 218 for generating a sequence of blocks of time domain sampling values is provided and, a synthesis windower 219 is connected to an output of the discrete Fourier inverse transform processor 218 for windowing the sequence of blocks for low pass time domain sampling values using a synthesis window.

20

Preferably, the low pass shaper consisting of blocks 217a, 217b applies the low pass shape $ps[f]$ by multiplying individual FFT lines as illustrated by the multiplier 217a. The low pass shape $ps[f]$ is calculated as indicated previously with respect to Fig. 9c.

25 Additionally, the audio signal itself, i.e., the full band audio signal is also windowed using the audio signal windower 221 to obtain a sequence of windowed blocks of audio signal values, wherein this audio signal windower 221 is synchronized with the analysis windower 215 and/or the synthesis windower 219 so that the sequence of blocks of low pass time domain sampling values is synchronous with the sequence of window blocks of
30 audio signal values.

Furthermore, the analyzer 260 of Fig. 7 is configured to additionally provide the control parameter 807, used to control the strength of the pre-processing between none and full effect, and 500, i.e., the β_factor as a further side information, where the high band
35 processor 220 is configured to apply the modification also under consideration of the additional control parameter 807, wherein the time resolution of the β_factor parameter

is lower than a time resolution of the time-varying high frequency gain information or the additional control parameter is stationary for a specific audio piece. As mentioned before, the probability-like control parameter from MGC is used to squeeze the gains towards 1 before quantization, and it is not explicitly transmitted as side information.

5

Furthermore, the combiner 230 is configured to perform a sample-wise addition of corresponding blocks of the sequence of blocks of low pass time domain sampling values and the sequence of modified, i.e., processed blocks of high pass time domain sampling values to obtain a sequence of blocks of combination signal values as illustrated, for the
10 post-processor side, in Fig. 3c.

Fig. 9a illustrates an audio encoding apparatus for encoding an audio signal comprising the audio pre-processor 200 as discussed before that is configured to generate the output signal 252 having the time-variable high frequency gain information as side information.

15 Furthermore, a core encoder 900 is provided for generating a core encoded signal 902 and a core side information 904. Additionally, the audio encoding apparatus comprises an output interface 910 for generating an encoded signal 912 comprising the core encoded signal 902, the core side information 904 and the time-variable high frequency gain information as additional side information 106.

20

Preferably, the audio pre-processor 200 performs a pre-processing of each channel or each object separately as illustrated in Fig. 10a. In this case, the audio signal is a multichannel or a multi-object signal. In a further embodiment, illustrated in Fig. 5c, the audio pre-processor 200 performs a pre-processing of each SAOC transport channel or
25 each High Order Ambisonics (HOA) transport channel separately as illustrated in Fig. 10a. In this case, the audio signal is a spatial audio object transport channel or a High Order Ambisonics transport channel.

Contrary thereto, the core encoder 900 is configured to apply a joint multichannel encoder
30 processing or a joint multi-object encoder processing or an encoder gap filling or an encoder bandwidth extension processing on the pre-processed channels 232.

Thus, typically, the core encoded signal 902 has less channels than were introduced into the joint multichannel / multi-object core encoder 900, since the core encoder 900 typically
35 comprises a kind of a downmix operation.

An audio decoding apparatus is illustrated in Fig. 9b. The audio decoding apparatus has an audio input interface 920 for receiving the encoded audio signal 912 comprising a core encoded signal 902, core side information 904 and the time-variable high frequency gain information 104 as additional side information 106. Furthermore, the audio decoding apparatus comprises a core decoder 930 for decoding the core encoded signal 902 using the core side information 904 to obtain the decoded core signal 102. Additionally, the audio decoding apparatus has the post-processor 100 for post-processing the decoded core signal 102 using the time-variable high frequency gain information 104.

10 Preferably, and as illustrated in Fig. 10b, the core decoder 930 is configured to apply a multichannel decoder processing or a multi-object decoder processing or a bandwidth extension decoder processing or a gap-filling decoder processing for generating decoded channels of a multichannel signal 102 or decoded objects of a multi-object signal 102. Thus, in other words, the joint decoder processor 930 typically comprises some kind of upmix in order to generate, from a lower number of channels in the encoded audio signal 902, a higher number of individual objects/channels. These individual channels/objects are input into a channel-individual post-processing by the audio post-processor 100 using the individual time-variable high frequency gain information for each channel or each object as illustrated at 104 in Fig. 10b. The channel-individual post-processor 100 outputs post-processed channels that can be output to a digital/analog converter and subsequently connected loudspeakers or that can be output to some kind of further processing or storage or any other suitable procedure for processing audio objects or audio channels.

25 Fig. 10c illustrates a situation similar to what has been illustrated in Figs. 9a or 9b, i.e., a full chain comprising of a high resolution envelope processing pre-processor 100 connected to an encoder 900 for generating a bitstream and the bitstream is decoded by the decoder 930 and the decoder output is post-processed by the high resolution envelope processor post-processor 100 to generate the final output signal.

30

Fig. 16 and Figs. 17a to 17h illustrate listening test results for a 5.1 channel loudspeaker listening (128 kbps). Additionally, results for a stereo headphone listening at medium (48 kbps) and high (128 kbps) quality are provided. Fig. 16a summarizes the listening test setups. The results are illustrated in Figs. 17a to 17h.

35

In Fig. 17a, the perceptual quality is in the “good” to “excellent” range. It is noted that applause-like signals are among the lowest-scoring items in the range “good”.

Fig. 17b illustrates that all applause items exhibit a significant improvement, whereas no significant change in perceptual quality is observed for the non-applause items. None of the items is significantly degraded.

Regarding Figs. 17c and 17d, it is outlined that the absolute perceptual quality is in the “good” range. In the differences, overall, there is a significant gain of seven points. Individual quality gains range between 4 and 9 points, all being significant.

In Fig. 17e, all signals of the test set are applause signals. The perceptual quality is in the “fair” to “good” range. Consistently, the “HREP” conditions score higher than the “NOHREP” condition. In Fig. 17f, it is visible that, for all items except one, “HREP” scores significantly better than “NOHREP”. Improvements ranging from 3 to 17 points are observed. Overall, there is a significant average gain of 12 points. None of the items is significantly degraded.

Regarding Fig. 17g and 17h, it is visible that, in the absolute scores, all signals score in the range “excellent”. In the differences scores it can be seen that, even though perceptual quality is near transparent, for six out of eight signals there is a significant improvement of three to nine points overall amounting to a mean of five MUSHRA points. None of the items are significantly degraded.

The results clearly show that the HREP technology of the preferred embodiments is of significant merit for the coding of applause-like signals in a wide range of bit rates/absolute qualities. Moreover, it is shown that there is no impairment whatsoever on non-applause signals. HREP is a tool for improved perceptual coding of signals that predominantly consist of many dense transient events, such as applause, rain sounds, etc. The benefits of applying HREP are two-fold: HREP relaxes the bit rate demand imposed on the encoder by reducing short-time dynamics of the input signal; additionally, HREP ensures proper envelope restoration in the decoder’s (up-)mixing stage, which is all the more important if parametric multichannel coding techniques have been applied within the codec. Subjective tests have shown an improvement of around 12 MUSHRA points by HREP processing at 48 kbps stereo and 7 MUSHRA points at 128 kbps 5.1 channels.

Subsequently, reference is made to Fig. 5b illustrating the implementation of the post-processing on the one hand or the pre-processing on the other hand within an MPEG-H 3D audio encoder/decoder framework. Specifically, Fig. 5b illustrates the HREP post-processor 100 as implemented within an MPEG-H 3D audio decoder. Specifically, the inventive post-processor is indicated at 100 in Fig. 5b.

It is visible that the HREP decoder is connected to an output of the 3D audio core decoder illustrated at 550. Additionally, between element 550 and block 100 in the upper portion, an MPEG surround element is illustrated that, typically performs an MPEG surround-implemented upmix from base channels at the input of block 560 to obtain more output channels at the output of block 560.

Furthermore, Fig. 5b illustrates other elements in addition to the audio core portion. These are, in the audio rendering portion, a drc_1 570 for channels on the one hand and objects on the other hand. Furthermore, a former conversion block 580, an object renderer 590, an object metadata decoder 592, an SAOC 3D decoder 594 and a High Order Ambisonics (HOA) decoder 596 are provided.

All these elements feed a resampler 582 and the resampler feeds its output data into a mixer 584. The mixer either forwards its output channels into a loudspeaker feed 586 or a headphone feed 588, which represent elements in the “end of chain” and which represent an additional post-processing subsequent to the mixer 584 output.

Fig. 5c illustrates a further preferred embedding of the audio post-processor (HREP) in the framework of an MPEG-H 3D audio decoder. In contrast to Fig. 5b, the HREP processing is also applied to the SAOC transport channels and/or to the HOA transport channels. The other functionalities in Fig. 5c are similar to those in Fig. 5b.

It is to be noted that attached claims related to the band extractor apply for the band extractor in the audio post-processor and the audio pre-processor as well even when a claim is only provided for a post-processor in one of the post-processor or the pre-processor. The same is valid for the high band processor and the combiner.

Particular reference is made to the further embodiments illustrated in the Annex and in the Annex A.

While this invention has been described in terms of several embodiments, there are alterations, permutations, and equivalents which fall within the scope of this invention. It should also be noted that there are many alternative ways of implementing the methods and compositions of the present invention. It is therefore intended that the following
5 appended claims be interpreted as including all such alterations, permutations and equivalents as fall within the true spirit and scope of the present invention.

Although some aspects have been described in the context of an apparatus, it is clear that these aspects also represent a description of the corresponding method, where a block or device corresponds to a method step or a feature of a method step. Analogously, aspects
10 described in the context of a method step also represent a description of a corresponding block or item or feature of a corresponding apparatus. Some or all of the method steps may be executed by (or using) a hardware apparatus, like for example, a microprocessor, a programmable computer or an electronic circuit. In some embodiments, some one or more of the most important method steps may be executed by such an apparatus.

15 The inventive encoded audio signal can be stored on a digital storage medium or can be transmitted on a transmission medium such as a wireless transmission medium or a wired transmission medium such as the Internet.

Depending on certain implementation requirements, embodiments of the invention can be implemented in hardware or in software. The implementation can be performed using a
20 digital storage medium, for example a floppy disk, a DVD, a Blu-Ray, a CD, a ROM, a PROM, an EPROM, an EEPROM or a FLASH memory, having electronically readable control signals stored thereon, which cooperate (or are capable of cooperating) with a programmable computer system such that the respective method is performed. Therefore, the digital storage medium may be computer readable.

25 Some embodiments according to the invention comprise a data carrier having electronically readable control signals, which are capable of cooperating with a programmable computer system, such that one of the methods described herein is performed.

Generally, embodiments of the present invention can be implemented as a computer
30 program product with a program code, the program code being operative for performing one of the methods when the computer program product runs on a computer. The program code may for example be stored on a machine readable carrier.

Other embodiments comprise the computer program for performing one of the methods described herein, stored on a machine readable carrier.

In other words, an embodiment of the inventive method is, therefore, a computer program having a program code for performing one of the methods described herein, when the
5 computer program runs on a computer.

A further embodiment of the inventive methods is, therefore, a data carrier (or a digital storage medium, or a computer-readable medium) comprising, recorded thereon, the computer program for performing one of the methods described herein. The data carrier, the digital storage medium or the recorded medium are typically tangible and/or non-
10 transitional.

A further embodiment of the inventive method is, therefore, a data stream or a sequence of signals representing the computer program for performing one of the methods described herein. The data stream or the sequence of signals may for example be configured to be transferred via a data communication connection, for example via the
15 Internet.

A further embodiment comprises a processing means, for example a computer, or a programmable logic device, configured to or adapted to perform one of the methods described herein.

A further embodiment comprises a computer having installed thereon the computer
20 program for performing one of the methods described herein.

A further embodiment according to the invention comprises an apparatus or a system configured to transfer (for example, electronically or optically) a computer program for performing one of the methods described herein to a receiver. The receiver may, for example, be a computer, a mobile device, a memory device or the like. The apparatus or
25 system may, for example, comprise a file server for transferring the computer program to the receiver.

In some embodiments, a programmable logic device (for example a field programmable gate array) may be used to perform some or all of the functionalities of the methods described herein. In some embodiments, a field programmable gate array may cooperate
30 with a microprocessor in order to perform one of the methods described herein. Generally, the methods are preferably performed by any hardware apparatus.

The apparatus described herein may be implemented using a hardware apparatus, or using a computer, or using a combination of a hardware apparatus and a computer.

The methods described herein may be performed using a hardware apparatus, or using a computer, or using a combination of a hardware apparatus and a computer.

- 5 The above described embodiments are merely illustrative for the principles of the present invention. It is understood that modifications and variations of the arrangements and the details described herein will be apparent to others skilled in the art. It is the intent, therefore, to be limited only by the scope of the impending patent claims and not by the specific details presented by way of description and explanation of the embodiments
- 10 herein.

Annex

Description of a further embodiment of HREP in MPEG-H 3DAudio

- 5 High Resolution Envelope Processing (HREP) is a tool for improved perceptual coding of signals that predominantly consist of many dense transient events, such as applause, rain drop sounds, etc. These signals have traditionally been very difficult to code for MPEG audio codecs, particularly at low bitrates. Subjective tests have shown a significant improvement of around 12 MUSHRA points by HREP processing at 48 kbps stereo.

10 Executive Summary

The HREP tool provides improved coding performance for signals that contain densely spaced transient events, such as applause signals as they are an important part of live recordings. Similarly, raindrops sound or other sounds like fireworks can show such characteristics. Unfortunately, this class of sounds presents difficulties to existing audio codecs, especially when
15 coded at low bitrates and/or with parametric coding tools.

- Figure 10c depicts the signal flow in an HREP equipped codec. At the encoder side, the tool works as a preprocessor that temporally flattens the signal for high frequencies while generating a small amount of side information (1-4 kbps for stereo signals). At the decoder side, the tool works as a postprocessor that temporally shapes the signal for high frequencies, making use of the side
20 information. The benefits of applying HREP are two-fold: HREP relaxes the bitrate demand imposed on the encoder by reducing short time dynamics of the input signal; additionally, HREP ensures proper envelope restoration in the decoder's (up-)mixing stage, which is all the more important if parametric multi-channel coding techniques have been applied within the codec.

Figure 10c: Overview of signal flow in an HREP equipped codec.

- 25 The HREP tool works for all input channel configurations (mono, stereo, multi-channel including 3D) and also for audio objects.

In the core experiment, we present MUSHRA listening test results, which show the merit of HREP for coding applause signals. Significant improvement in perceptual quality is demonstrated for the following test cases

- 30
- 7 MUSHRA points average improvement for 5.1 channel at 128kbit/s
 - 12 MUSHRA points average improvement for stereo 48kbit/s
 - 5 MUSHRA points average improvement for stereo 128kbit/s

- 35 Exemplary, through assessing the perceptual quality for 5.1ch signals employing the full well-known MPEG Surround test set, we prove that the quality of non-applause signals is not impaired by HREP.

Detailed description of HREP

- 40 Figure 10c depicts the signal flow in an HREP equipped codec. At the encoder side, the tool works as a preprocessor with high temporal resolution before the actual perceptual audio codec by analyzing the input signal, attenuating and thus temporally flattening the high frequency part of transient events, and generating a small amount of side information (1-4 kbps for stereo signals). An applause classifier may guide the encoder decision whether or not to activate HREP. At the decoder side, the tool works as a postprocessor after the audio codec by boosting and thus
45 temporally shaping the high frequency part of transient events, making use of the side information that was generated during encoding.

Figure 9c: Detailed HREP signal flow in the encoder.

Figure 9c displays the signal flow inside the HREP processor within the encoder. The preprocessing is applied by splitting the input signal into a low pass (LP) part and a high pass (HP) part. This is accomplished by using FFT to compute the LP part, Given the LP part, the HP part is obtained by subtraction in time domain. A time-dependent scalar gain is applied to the HP part, which is added back to the LP part to create the preprocessed output.

The side information comprises low pass (LP) shape information and scalar gains that are estimated within an HREP analysis block (not depicted). The HREP analysis block may contain additional mechanisms that can gracefully lessen the effect of HREP processing on signal content (“non-applause signals”) where HREP is not fully applicable. Thus, the requirements on applause detection accuracy are considerably relaxed.

Figure 5a: Detailed HREP signal flow in the decoder.

The decoder side processing is outlined in Figure 5a. The side information on HP shape information and scalar gains are parsed from the bit stream (not depicted) and applied to the signal resembling a decoder post-processing inverse to that of the encoder pre-processing. The post-processing is applied by again splitting the signal into a low pass (LP) part and a high pass (HP) part. This is accomplished by using FFT to compute the LP part, Given the LP part, the HP part is obtained by subtraction in time domain. A scalar gain dependent on transmitted side information is applied to the HP part, which is added back to the LP part to create the preprocessed output.

All HREP side information is signaled in an extension payload and embedded backward compatibly within the MPEG-H 3DAudio bit stream.

Specification text

The necessary WD changes, the proposed bit stream syntax, semantics and a detailed description of the decoding process can be found in the Annex A of the document as a diff-text.

Complexity

The computational complexity of the HREP processing is dominated by the calculation of the DFT/IDFT pairs that implement the LP/HP splitting of the signal. For each audio frame comprising 1024 time domain values, 16 pairs of 128-point real valued DFT/IDFTs have to be calculated.

For inclusion into the low complexity (LC) profile, we propose the following restrictions

- Limitation of active HREP channels/objects
- Limitation to the maximum transmitted gain factors $g(k)$ that are non-trivial (trivial gain factors of 0dB alleviate the need for an associated DFT/IDFT pair)
- Calculation of the DFT/iDFT in an efficient split-radix 2 sparse topology

Evidence of merit

Listening tests

As an evidence of merit, listening test results will be presented for 5.1 channel loudspeaker listening (128kbps). Additionally, results for stereo headphone listening at medium (48kbps) and high (128kbps) quality are provided. Figure 16 summarizes the listening test setups.

Fig. 16 — Listening tests overview.

Results

128kbps 5.1ch

5 Figure shows the absolute MUSHRA scores of the 128kbps 5.1ch test. Perceptual quality is in the “good” to “excellent” range. Note that applause-like signals are among the lowest-scoring items in the range “good”.

Figure 17a: Absolute MUSHRA scores for 128kbps 5.1ch test.

10 Figure 17b depicts the difference MUSHRA scores of the 128kbps 5.1ch test. All applause items exhibit a significant improvement, whereas no significant change in perceptual quality is observed for the non-applause items. None of the items is significantly degraded.

Figure 17b: Difference MUSHRA scores for 128kbps 5.1ch test.

15 Figure 17c depicts the absolute MUSHRA scores for all applause items contained in the test set and Figure 17d depicts the difference MUSHRA scores for all applause items contained in the test set. Absolute perceptual quality is in the “good” range. In the differences, overall, there is a significant gain of 7 points. Individual quality gains range between 4 and 9 points, all being significant.

Figure 17c: Absolute MUSHRA scores for 128kbps 5.1ch test applause signals.

20 Figure 17d: Difference MUSHRA scores for 128kbps 5.1ch test applause signals.

48kbps stereo

25 Figure 17e shows the absolute MUSHRA scores of the 48kbps stereo test. Here, all signals of the set are applause signals. Perceptual quality is in the “fair” to “good” range. Consistently, the “hrep” condition scores higher than the “nohrep” condition. Figure 17f depicts the difference MUSHRA scores. For all items except one, “hrep” scores significantly better than “nohrep”. Improvements ranging from 3 to 17 points are observed. Overall, there is a significant average gain of 12 points. None of the items is significantly degraded.

Figure 17e: Absolute MUSHRA scores for 48kbps stereo test.

30 Figure 17f: Difference MUSHRA scores for 48kbps stereo test.

128kbps stereo

35 Figure 17g and Figure 17h show the absolute and the difference MUSHRA scores of the 128kbps stereo test, respectively. In the absolute scores, all signals score in the range “excellent”. In the differences scores it can be seen that, even though perceptual quality is near transparent, for 6 out of 8 signals there is a significant improvement of 3 to 9 points, overall amounting to a mean of 5 MUSHRA points. None of the items is significantly degraded.

Figure 17g: Absolute MUSHRA scores for 128kbps stereo test.

Figure 17h: Difference MUSHRA scores for 128kbps stereo test.

40

The results clearly show that the HREP technology of the CE proposal is of significant merit for the coding of applause-like signals in a large range of bitrates/absolute qualities. Moreover, it is proven that there is no impairment whatsoever on non-applause signals.

Conclusion

- 5 HPREP is a tool for improved perceptual coding of signals that predominantly consist of many dense transient events, such as applause, rain drop sounds, etc. The benefits of applying HREP are two-fold: HREP relaxes the bitrate demand imposed on the encoder by reducing short time dynamics of the input signal; additionally, HREP ensures proper envelope restoration in the decoder's (up-)mixing stage, which is all the more important if parametric multi-channel coding
- 10 techniques have been applied within the codec. Subjective tests have shown an improvement of around 12 MUSHRA points by HREP processing at 48 kbps stereo, and 7 MUSHRA points at 128kbps 5.1ch.

Annex A

Preferred Embodiment of HREP within MPEG-H 3DAudio

- 5 Subsequently, data modifications for changes required for HREP relative to ISO/IEC 23008-3:2015 and ISO/IEC 23008-3:2015/EAM3 documents are given.

Add the following line to Table 1, “MPEG-H 3DA functional blocks and internal processing domain. $f_{s,core}$ denotes the core decoder output sampling rate, $f_{s,out}$ denotes the decoder output sampling rate.”, in Section 10.2:

Table 1 — MPEG-H 3DA functional blocks and internal processing domain. $f_{s,core}$ denotes the core decoder output sampling rate, $f_{s,out}$ denotes the decoder output sampling rate.

Processing Context	Functional Block		Processing Domain	Delay Samples [$1/f_{s,core}$] or [$1/f_{s,out}$]	Contribution to Maximum Delay High Profile Samples [$1/f_{s,out}$]	Contribution to Maximum Delay Low Complexity Profile Samples [$1/f_{s,out}$]
Audio Core	HREP		TD, Core frame length = 1024	64		64 * RSR_{max}
		QMF-Synthesis and QMF-Analysis pair and alignment to 64 sample grid	FD TD FD	64 + 257 + 320 + 63	(64 + 257 + 320 + 63) * RSR_{max}	

- 15

Add the following Case to Table 13, “Syntax of mpeg3daExtElementConfig()”, in Section 5.2.2.3:

Table 13 – Syntax of mpeg3daExtElementConfig()

...	case ID_EXT_ELE_HREP:
	HREPCfg(current_signal_group);
	break;
...	

Add the following value definition to Table 50, “Value of usacExtElementType”, in Section 5.3.4:

Table 50 – Value of usacExtElementType

ID_EXT_ELE_HREP	12
/* reserved for ISO use */	13 – 127

- 5 Add the following interpretation to Table 51, “Interpretation of data blocks for extension payload decoding”, in Section 5.3.4:

Table 51 – Interpretation of data blocks for extension payload decoding

ID_EXT_ELE_HREP	HREPFramE(outputFrameLength, current_signal_group)
-----------------	---

Add new subclause at the end of 5.2.2 and add the following Table:

- 10 **5.2.2.X Extension Element Configurations**

Table 2 — Syntax of HREPConfig()

Syntax	No. of bits	Mnemonic
<pre> HREPConfig(current_signal_group) { signal_type = signalGroupType[current_signal_group]; signal_count = bsNumberOfSignals[current_signal_group] + 1; if (signal_type == SignalGroupTypeChannels) { channel_layout = audioChannelLayout[current_signal_group]; } extendedGainRange; extendedBetaFactorPrecision; for (sig = 0; sig < signal_count; sig++) { if ((signal_type == SignalGroupTypeChannels) && <i>isLFEChannel</i>(channel_layout, sig)) { isHREPAActive[sig] = 0; } else { isHREPAActive[sig]; } if (isHREPAActive[sig]) { if (sig == 0) { lastFFTLLine[0]; transitionWidthLines[0]; defaultBetaFactorIdx[0]; } else { if (useCommonSettings) { lastFFTLLine[sig] = lastFFTLLine[0]; transitionWidthLines[sig] = transitionWidthLines[0]; defaultBetaFactorIdx[sig] = defaultBetaFactorIdx[0]; } else { lastFFTLLine[sig]; transitionWidthLines[sig]; defaultBetaFactorIdx[sig]; } } } } } </pre>		
extendedGainRange;	1	uimsbf
extendedBetaFactorPrecision;	1	uimsbf
	NOTE 1	
isHREPAActive[sig];	1	uimsbf
	NOTE 2	
lastFFTLLine[0];	4	uimsbf
transitionWidthLines[0];	4	uimsbf
defaultBetaFactorIdx[0];	nBitsBeta	uimsbf
	NOTE 3	
useCommonSettings	1	uimsbf
lastFFTLLine[sig];	4	uimsbf
transitionWidthLines[sig];	4	uimsbf
defaultBetaFactorIdx[sig];	nBitsBeta	uimsbf
NOTE 1: The helper function <i>isLFEChannel</i>(channel_layout, sig) returns 1 if the channel on position sig in channel_layout is a LFE channel or 0 otherwise. NOTE 3: nBitsBeta = 3 + extendedBetaFactorPrecision.		

At the end of 5.2.2.3 add the following Tables:

Table 3 — Syntax of HREPFrame()

Syntax	No. of bits	Mnemonic
<pre> HREPFrame(outputFrameLength, current_signal_group) { gain_count = outputFrameLength / 64; signal_count = bsNumberOfSignals[current_signal_group] + 1; useRawCoding; if (useRawCoding) { for (pos = 0; pos < gain_count; pos++) { for (sig = 0; sig < signal_count; sig++) { if (isHREPAActive[sig] == 0) continue; } } } } </pre>		
useRawCoding;	1	uimsbf
	NOTE 1	


```
    return res;
}

5  ari_decode_bit(*ari_state)
    {
        prob_scale = 1 << 14;
        tbl[0] = prob_scale >> 1;
        tbl[1] = 0;
10  res = arith_decode(ari_state, tbl, 2);
    return res;
    }

15  HREP_decode_ac_data(gain_count, signal_count)
    {
        cnt_mask[2] = {1, 1};
        cnt_sign[2] = {1, 1};
        cnt_neg[2] = {1, 1};
20  cnt_pos[2] = {1, 1};

        arith_start_decoding(&ari_state);

        for (pos = 0; pos < gain_count; pos++) {
25  for (sig = 0; sig < signal_count; sig++) {
            if (!isHREPActive[sig]) {
                continue;
            }
            mask_bit = ari_decode_bit_with_prob(&ari_state, cnt_mask[0], cnt_mask[0] + cnt_mask[1]);
30  cnt_mask[mask_bit]++;

            if (mask_bit) {
                sign_bit = ari_decode_bit_with_prob(&ari_state, cnt_sign[0], cnt_sign[0] + cnt_sign[1]);
                cnt_sign[sign_bit] += 2;
35  if (sign_bit) {
                    large_bit = ari_decode_bit_with_prob(&ari_state, cnt_neg[0], cnt_neg[0] + cnt_neg[1]);
                    cnt_neg[large_bit] += 2;
                    last_bit = ari_decode_bit(&ari_state);
40  gainIdx[pos][sig] = -2 * large_bit - 2 + last_bit;
                } else {
                    large_bit = ari_decode_bit_with_prob(&ari_state, cnt_pos[0], cnt_pos[0] + cnt_pos[1]);
                    cnt_pos[large_bit] += 2;
                    if (large_bit) {
45  gainIdx[pos][sig] = 3;
                    } else {
                        last_bit = ari_decode_bit(&ari_state);
                        gainIdx[pos][sig] = 2 - last_bit;
                    }
                }
            }
        }
    }
}
```

```
        }
    }
    } else {
        gainIdx[pos][sig] = 0;
5    }

    if (extendedGainRange) {
        prob_scale = 1 << 14;
        esc_cnt = prob_scale / 5;
10    tbl_esc[5] = {prob_scale - esc_cnt, prob_scale - 2 * esc_cnt, prob_scale - 3 * esc_cnt, prob_scale -
        4 * esc_cnt, 0};
        sym = gainIdx[pos][sig];
        if (sym <= -4) {
            esc = arith_decode(ari_state, tbl_esc, 5);
15    sym = -4 - esc;
        } else if (sym >= 3) {
            esc = arith_decode(ari_state, tbl_esc, 5);
            sym = 3 + esc;
        }
20    gainIdx[pos][sig] = sym;
    }

    gainIdx[pos][sig] += GAIN_INDEX_0dB;
    }
25 }

    arith_done_decoding(&ari_state);
}
```

30 *Add the following new subclauses “5.5.X High Resolution Envelope Processing (HREP) Tool” at the end of subclause 5.5:*

5.5.X High Resolution Envelope Processing (HREP) Tool

5.5.X.1 Tool Description

35 The HREP tool provides improved coding performance for signals that contain densely spaced transient events, such as applause signals as they are an important part of live recordings. Similarly, raindrops sound or other sounds like fireworks can show such characteristics. Unfortunately, this class of sounds presents difficulties to existing audio codecs, especially when coded at low bitrates and/or with parametric coding tools.

40 Fig. 5b or 5c depicts the signal flow in an HREP equipped codec. At the encoder side, the tool works as a pre-processor that temporally flattens the signal for high frequencies while generating a small amount of side information (1-4 kbps for stereo signals). At the decoder side, the tool works as a post-processor that temporally shapes the signal for

high frequencies, making use of the side information. The benefits of applying HREP are two-fold: HREP relaxes the bit rate demand imposed on the encoder by reducing short time dynamics of the input signal; additionally, HREP ensures proper envelope restauration in the decoder's (up-)mixing stage, which is all the more important if parametric multi-channel coding techniques have been applied within the codec. The HREP tool works for all input channel configurations (mono, stereo, multi-channel including 3D) and also for audio objects.

5.5.X.2 Data and Help Elements

10	<code>current_signal_group</code>	The <code>current_signal_group</code> parameter is based on the <code>Signals3d()</code> syntax element and the <code>mpeg3daDecoderConfig()</code> syntax element.
	<code>signal_type</code>	The type of the current signal group, used to differentiate between channel signals and object, HOA, and SAOC signals.
15	<code>signal_count</code>	The number of signals in the current signal group.
	<code>channel_layout</code>	In case the current signal group has channel signals, it contains the properties of speakers for each channel, used to identify LFE speakers.
20	<code>extendedGainRange</code>	Indicates whether the gain indexes use 3 bits (8 values) or 4 bits (16 values), as computed by <code>nBitsGain</code> .
	<code>extendedBetaFactorPrecision</code>	Indicates whether the beta factor indexes use 3 bits or 4 bits, as computed by <code>nBitsBeta</code> .
	<code>isHREPActive[sig]</code>	Indicates whether the tool is active for the signal on index <code>sig</code> in the current signal group.
25	<code>lastFFTLLine[sig]</code>	The position of the last non-zero line used in the low-pass procedure implemented using FFT.
	<code>transitionWidthLines[sig]</code>	The width in lines of the transition region used in the low-pass procedure implemented using FFT.
30	<code>defaultBetaFactorIdx[sig]</code>	The default beta factor index used to modify the gains in the gain compensation procedure.
	<code>outputFrameLength</code>	The equivalent number of samples per frame, using the original sampling frequency, as defined in the USAC standard.
35	<code>gain_count</code>	The number of gains per signal in one frame.

	useRawCoding	Indicates whether the gain indexes are coded raw, using nBitsGain each, or they are coded using arithmetic coding.
	gainIdx[pos][sig]	The gain index corresponding to the block on position pos of the signal on position sig in the current signal group. If
5		extendedGainRange = 0, the possible values are in the range {0, ..., 7}, and if extendedGainRange = 1, the possible values are in the range {0, ..., 15}.
	GAIN_INDEX_0dB	The gain index offset corresponding to 0 dB, with a value of
10		4 being used if extendedGainRange = 0, and with a value of 8 being used if extendedGainRange = 1. The gain indexes are transmitted as unsigned values by adding
		GAIN_INDEX_0dB to their original signed data ranges.
	all_zero	Indicates whether all the gain indexes in one frame for the current signal are having the value GAIN_INDEX_0dB.
15	useDefaultBetaFactorIdx	Indicates whether the beta factor index for the current signal has the default value specified by defaultBetaFactor[sig].
	betaFactorIdx[sig]	The beta factor index used to modify the gains in the gain compensation procedure.

20 5.5.X.2.1 Limitations for Low Complexity Profile

If the total number of signals counted over all signal groups is at most 6 there are no limitations.

Otherwise, if the total number of signals where HREP is active, indicated by the isHREPAActive[sig] syntax element in HREPConfig(), and counted over all signal groups is
25 at most 4, there are no further limitations.

Otherwise, the total number of signals where HREP is active, indicated by the isHREPAActive[sig] syntax element in HREPConfig(), and counted over all signal groups, shall be limited to at most 8.

Additionally, for each frame, the total number of gain indexes which are different than
30 GAIN_INDEX_0dB, counted for the signals where HREP is active and over all signal groups, shall be at most $4 \times \text{gain_count}$. For the blocks which have a gain index equal with GAIN_INDEX_0dB, the FFT, the interpolation correction, and the IFFT shall be skipped. In this case, the input block shall be multiplied with the square of the sine window and used directly in the overlap-add procedure.

35

5.5.X.3 Decoding Process

5.5.X.3.1 General

In the syntax element *mpegh3daExtElementConfig()* the field *usacExtElementPayloadFrag* shall be zero in the case of an ID_EXT_ELE_HREP element. The HREP tool is applicable only to signal groups of type
5 SignalGroupTypeChannels and SignalGroupTypeObject, as defined by SignalGroupType[grp] in the Signals3d() syntax element. Therefore, the ID_EXT_ELE_HREP elements shall be present only for the signal groups of type SignalGroupTypeChannels and SignalGroupTypeObject.

The block size and correspondingly the FFT size used is $N = 128$.

- 10 The entire processing is done independently on each signal in the current signal group. Therefore, to simplify notation, the decoding process is described only for one signal on position sig.

Figure 5a: Block Diagram of the High Resolution Envelope Processing (HREP) Tool at Decoding Side

5.5.X.3.2 Decoding of quantized beta factors

The following lookup tables for converting beta factor index *betaFactorIdx[sig]* to beta factor *beta_factor* should be used, depending on the value of
20 *extendedBetaFactorPrecision*.

```
tab_beta_factor_dequant_coarse[8] = {  
    0.000f, 0.035f, 0.070f, 0.120f, 0.170f, 0.220f, 0.270f, 0.320f  
}
```

25

```
tab_beta_factor_dequant_precise[16] = {  
    0.000f, 0.035f, 0.070f, 0.095f, 0.120f, 0.145f, 0.170f, 0.195f,  
    0.220f, 0.245f, 0.270f, 0.295f, 0.320f, 0.345f, 0.370f, 0.395f  
}
```

- 30 If *extendedBetaFactorPrecision* = 0, the conversion is computed as

```
beta_factor = tab_beta_factor_dequant_coarse[betaFactorIndex[sig]]
```

If *extendedBetaFactorPrecision* = 1, the conversion is computed as

```
beta_factor = tab_beta_factor_dequant_precise[betaFactorIndex[sig]]
```

35 5.5.X.3.3 Decoding of quantized gains

One frame is processed as *gain_count* blocks consisting of N samples each, which are half-overlapping. The scalar gains for each block are derived, based on the value of *extendedGainRange*.

$$g[k] = 2^{\frac{\text{gainIdx}[k][sig] - \text{GAIN_INDEX_0dB}}{4}}, \text{ for } 0 \leq k < \text{gain_count}$$

5.5.X.3.4 Computation of the LP part and the HP part

- 5 The input signal s is split into blocks of size N , which are half-overlapping, producing input blocks $\text{ib}[k][i] = s[k \times \frac{N}{2} + i]$, where k is the block index and i is the sample position in the block k . A window $w[i]$ is applied to $\text{ib}[k]$, in particular the sine window, defined as

$$w[i] = \sin \frac{\pi(i + 0.5)}{N}, \text{ for } 0 \leq i < N,$$

10

and after also applying FFT, the complex coefficients $c[k][f]$ are obtained as

$$c[k][f] = \text{FFT}(w[i] \times \text{ib}[k]), \text{ for } 0 \leq f \leq \frac{N}{2}.$$

- 15 On the encoder side, in order to obtain the LP part, we apply an element-wise multiplication of $c[k]$ with the processing shape $\text{ps}[f]$, which consists of the following:

$$\text{ps}[f] = \begin{cases} 1, & \text{for } 0 \leq f < \text{lp_size} \\ 1 - \frac{f - \text{lp_size} + 1}{\text{tr_size} + 1}, & \text{for } \text{lp_size} \leq f < \text{lp_size} + \text{tr_size} \\ 0, & \text{for } \text{lp_size} + \text{tr_size} \leq f \leq \frac{N}{2} \end{cases}$$

- 20 The $\text{lp_size} = \text{lastFFTLines}[sig] + 1 - \text{transitionWidthLines}[sig]$ parameter represents the width in FFT lines of the low-pass region, and the $\text{tr_size} = \text{transitionWidthLines}[sig]$ parameter represents the width in FFT lines of the transition region.

On the decoder side, in order to get perfect reconstruction in the transition region, an adaptive reconstruction shape $\text{rs}[f]$ in the transition region must be used, instead of the

- 25 the processing shape $\text{ps}[f]$ used at the encoder side, depending on the processing shape $\text{ps}[f]$ and $g[k]$ as

$$\text{rs}[f] = 1 - (1 - \text{ps}[f]) \times \frac{g[k]}{1 + (g[k] - 1) \times (1 - \text{ps}[f])}$$

The LP block $\text{lpb}[k]$ is obtained by applying IFFT and windowing again as

30

$$\text{lpb}[k][i] = w[i] \times \text{IFFT}(\text{rs}[f] \times c[k][f]), \text{ for } 0 \leq i < N,$$

The HP block $\text{hpb}[k]$ is then obtained by simple subtraction in time domain as

$$\text{hpb}[k][i] = \text{in}[k][i] \times w^2[i] - \text{lpb}[k][i], \text{ for } 0 \leq i < N.$$

5.5.X.3.5 Computation of the interpolation correction

The gains $g[k-1]$ and $g[k]$ applied on the encoder side to blocks on positions $k-1$ and k are implicitly interpolated due to the windowing and overlap-add operations. In order to achieve perfect reconstruction in the HP part above the transition region, an interpolation correction factor is needed as

$$\begin{aligned} \text{corr}[j] &= 1 + \left(\frac{g[k-1]}{g[k]} + \frac{g[k]}{g[k-1]} - 2 \right) \times w^2[j] \times (1 - w^2[j]), \text{ for } 0 \leq j < \frac{N}{2}. \\ \text{corr}\left[j + \frac{N}{2}\right] &= 1 + \left(\frac{g[k]}{g[k+1]} + \frac{g[k+1]}{g[k]} - 2 \right) \times w^2[j] \times (1 - w^2[j]), \text{ for } 0 \leq j < \frac{N}{2}. \end{aligned}$$

5.5.X.3.6 Computation of the compensated gains

The core encoder and decoder introduce additional attenuation of transient events, which is compensated by adjusting the gains $g[k]$ using the previously computed beta_factor as

$$gc[k] = (1 + \text{beta_factor})g[k] - \text{beta_factor}$$

5.5.X.3.7 Computation of the output signal

Based on $gc[k]$ and $\text{corr}[i]$, the value of the output block $\text{ob}[k]$ is computed as

$$\text{ob}[k][i] = \text{lpb}[k][i] + \frac{1}{gc[k]} \times \frac{1}{\text{corr}[i]} \times \text{hpb}[k][i], \text{ for } 0 \leq i < N$$

Finally, the output signal is computed using the output blocks using overlap-add as

$$\begin{aligned} o\left[k \times \frac{N}{2} + j\right] &= \text{ob}[k-1]\left[j + \frac{N}{2}\right] + \text{ob}[k][j], \text{ for } 0 \leq j < \frac{N}{2} \\ o\left[(k+1) \times \frac{N}{2} + j\right] &= \text{ob}[k]\left[j + \frac{N}{2}\right] + \text{ob}[k+1][j], \text{ for } 0 \leq j < \frac{N}{2} \end{aligned}$$

5.5.X.4 Encoder Description (informative)

Figure 9c: Block Diagram of the High Resolution Envelope Processing (HREP) Tool at Encoding Side

5 **5.5.X.4.1 Computation of the gains and of the beta factor**

At the pre-processing side, the HP part of block k , assumed to contain a transient event, is adjusted using the scalar gain $g[k]$ in order to make it more similar to the background in its neighborhood. The energy of the HP part of block k will be denoted by $hp_e[k]$ and the average energy of the HP background in the neighborhood of block k will be denoted by $hp_bg_e[k]$.

We define the parameter $\alpha \in [0, 1]$, which controls the amount of adjustment as

$$g_{float}[k] = \begin{cases} \frac{\alpha \times hp_bg_e[k] + (1 - \alpha) \times hp_e[k]}{hp_e[k]}, & \text{when } hp_e[k] \geq T_{quiet}, \\ 1, & \text{otherwise} \end{cases}$$

- 15 The value of $g_{float}[k]$ is quantized and clipped to the range allowed by the chosen value of the extendedGainRange configuration option to produce the gain index $gainIdx[k][sig]$ as

$$g_{idx} = \lfloor \log_2(4 \times g_{float}[k]) + 0.5 \rfloor + \text{GAIN_INDEX_0dB},$$

$$gainIdx[k][sig] = \min(\max(0, g_{idx}), 2 \times \text{GAIN_INDEX_0dB} - 1).$$

- 20 The value $g[k]$ used for the processing is the quantized value, defined at the decoder side as

$$g[k] = 2^{\frac{gainIdx[k][sig] - \text{GAIN_INDEX_0dB}}{4}}.$$

- When α is 0, the gain has value $g_{float}[k] = 1$, therefore no adjustment is made, and when α is 1, the gain has value $g_{float}[k] = hp_bg_e[k]/hp_e[k]$, therefore the adjusted energy is made to coincide with the average energy of the background. We can rewrite the above relation as

$$g_{float}[k] \times hp_e[k] = hp_bg_e[k] + (1 - \alpha) \times (hp_e[k] - hp_bg_e[k]),$$

30

indicating that the variation of the adjusted energy $g_{float}[k] \times hp_e[k]$ around the corresponding average energy of the background $hp_bg_e[k]$ is reduced with a factor of $(1 - \alpha)$. In the proposed system, $\alpha = 0.75$ is used, thus the variation of the HP energy of

each block around the corresponding average energy of the background is reduced to 25 % of the original.

The core encoder and decoder introduce additional attenuation of transient events, which is approximately modeled by introducing an extra attenuation step, using the parameter $\beta \in [0, 1]$ depending on the core encoder configuration and the signal characteristics of the frame, as

$$g_{c_{float}}[k] = \frac{\beta \times hp_bg_e[k] + (1 - \beta) \times [g_{float}[k] \times hp_e[k]]}{hp_e[k]}$$

indicating that, after passing through the core encoder and decoder, the variation of the decoded energy $g_{c_{float}}[k] \times hp_e[k]$ around the corresponding average energy of the background $hp_bg_e[k]$ is further reduced with an additional factor of $(1 - \beta)$.

Using just $g[k]$, α , and β , it is possible to compute an estimate of $gc[k]$ at the decoder side as

$$gc[k] = \left(1 + \frac{\beta \times (1 - \alpha)}{\alpha}\right) \times g[k] - \frac{\beta \times (1 - \alpha)}{\alpha}$$

The parameter $\beta_factor = \frac{\beta \times (1 - \alpha)}{\alpha}$ is quantized to $\beta_factorIdx[sig]$ and transmitted as side information for each frame. The compensated gain $gc[k]$ can be computed using β_factor as

$$gc[k] = (1 + \beta_factor) \times g[k] - \beta_factor$$

5.5.X.4.2 Computation of the LP part and the HP part

The processing is identical to the corresponding one at the decoder side defined earlier, except that the processing shape $ps[f]$ is used instead of the adaptive reconstruction shape $rs[f]$ in the computation of the LP block $lpb[k]$, which is obtained by applying IFFT and windowing again as

$$lpb[k][i] = w[i] \times IFFT(ps[f] \times c[k][f]), \text{ for } 0 \leq i < N.$$

5.5.X.4.3 Computation of the output signal

Based on $g[k]$, the value of the output block $ob[k]$ is computed as

$$ob[k][i] = lpb[k][i] + g[k] \times hpb[k][i], \text{ for } 0 \leq i < N.$$

Identical to the decoder side, the output signal is computed using the output blocks using overlap-add as

$$\begin{aligned} o \left[k \times \frac{N}{2} + j \right] &= ob[k-1] \left[j + \frac{N}{2} \right] + ob[k][j], \text{ for } 0 \leq j < \frac{N}{2}, \\ 5 \quad o \left[(k+1) \times \frac{N}{2} + j \right] &= ob[k] \left[j + \frac{N}{2} \right] + ob[k+1][j], \text{ for } 0 \leq j < \frac{N}{2}. \end{aligned}$$

5.5.X.4.4 Encoding of gains using arithmetic coding

The helper function HREP_encode_ac_data(gain_count, signal_count) describes the writing of the gain values from the array gainIdx using the following USAC low-level arithmetic coding functions:

```
arith_encode(*ari_state, symbol, cum_freq),  
arith_encoder_open(*ari_state),  
arith_encoder_flush(*ari_state).
```

Two additional helper functions are introduced,

```
15  ari_encode_bit_with_prob(*ari_state, bit_value, count_0, count_total),  
    which encodes the one bit bit_value with  $p_0 = \text{count\_0}/\text{count\_total}$  and  $p_1 = 1 - p_0$ , and  
    ari_encode_bit(*ari_state, bit_value),  
    which encodes the one bit bit_value without modeling, with  $p_0 = 0.5$  and  $p_1 = 0.5$ .
```

```
20  ari_encode_bit_with_prob(*ari_state, bit_value, count_0, count_total)  
    {  
        prob_scale = 1 << 14;  
        tbl[0] = prob_scale - (count_0 * prob_scale) / count_total;  
        tbl[1] = 0;  
25  arith_encode(ari_state, bit_value, tbl);  
    }
```

```
    ari_encode_bit(*ari_state, bit_value)  
    {  
30  prob_scale = 1 << 14;  
        tbl[0] = prob_scale >> 1;  
        tbl[1] = 0;  
        ari_encode(ari_state, bit_value, tbl);  
    }
```

```
35  HREP_encode_ac_data(gain_count, signal_count)  
    {  
        cnt_mask[2] = {1, 1};  
        cnt_sign[2] = {1, 1};
```

```
cnt_neg[2] = {1, 1};
cnt_pos[2] = {1, 1};

arith_encoder_open(&ari_state);

5
for (pos = 0; pos < gain_count; pos++) {
    for (sig = 0; sig < signal_count; sig++) {
        if (!isHREPAActive[sig]) {
            continue;
10        }

        sym = gainIdx[pos][sig] - GAIN_INDEX_0dB;
        if (extendedGainRange) {
            sym_ori = sym;
15            sym = max(min(sym_ori, GAIN_INDEX_0dB / 2 - 1), -GAIN_INDEX_0dB / 2);
        }

        mask_bit = (sym != 0);
        arith_encode_bit_with_prob(ari_state, mask_bit, cnt_mask[0], cnt_mask[0] + cnt_mask[1]);
20        cnt_mask[mask_bit]++;

        if (mask_bit) {
            sign_bit = (sym < 0);
            arith_encode_bit_with_prob(ari_state, sign_bit, cnt_sign[0], cnt_sign[0] + cnt_sign[1]);
25            cnt_sign[sign_bit] += 2;

            if (sign_bit) {
                large_bit = (sym < -2);
                arith_encode_bit_with_prob(ari_state, large_bit, cnt_neg[0], cnt_neg[0] + cnt_neg[1]);
30                cnt_neg[large_bit] += 2;
                last_bit = sym & 1;
                arith_encode_bit(ari_state, last_bit);
            } else {
                large_bit = (sym > 2);
                arith_encode_bit_with_prob(ari_state, large_bit, cnt_pos[0], cnt_pos[0] + cnt_pos[1]);
35                cnt_pos[large_bit] += 2;
                if (large_bit == 0) {
                    last_bit = sym & 1;
                    ari_encode_bit(ari_state, last_bit);
40                }
            }
        }

        if (extendedGainRange) {
45            prob_scale = 1 << 14;
            esc_cnt = prob_scale / 5;
            tbl_esc[5] = {prob_scale - esc_cnt, prob_scale - 2 * esc_cnt, prob_scale - 3 * esc_cnt, prob_scale -
4 * esc_cnt, 0};
```

```

    if (sym_ori <= -4) {
        esc = -4 - sym_ori;
        arith_encode(ari_state, esc, tbl_esc);
    } else if (sym_ori >= 3) {
5       esc = sym_ori - 3;
        arith_encode(ari_state, esc, tbl_esc);
    }

10     }

    arith_encode_flush(ari_state);
}

15
```

Claims

1. Audio post-processor for post-processing an audio signal having a time-variable high frequency gain information as side information, comprising:

5

a band extractor for extracting a high frequency band of the audio signal and a low frequency band of the audio signal;

10

a high band processor for performing a time-variable amplification of the high frequency band in accordance with the time-variable high frequency gain information to obtain a processed high frequency band;

a combiner for combining the processed high frequency band and the low frequency band.

15

2. Audio post-processor of claim 1, in which the band extractor is configured to extract the low frequency band using a low pass filter device and to extract the high frequency band by subtracting the low frequency band from the audio signal.

20

3. Audio post-processor of claim 1 or 2, in which the time-variable high frequency gain information is provided for a sequence of blocks of sampling values of the audio signal so that a first block of sampling values has associated therewith a first gain information and a second later block of sampling values of the audio signal has a different second gain information, wherein the band extractor is configured to extract, from the first block of sampling values, a first low frequency band and a first high frequency band and to extract, from the second block of sampling values, a second low frequency band and a second high frequency band, and

25

wherein the high band processor is configured to modify the first high frequency band using the first gain information to obtain a first processed high frequency band and to modify the second high frequency band using the second gain information to obtain a second processed high frequency band, and

30

wherein the combiner is configured to combine the first low frequency band and the first processed high frequency band to obtain a first combined block and to

35

combine the second low frequency band and the second processed high frequency band to obtain a second combined block.

4. Audio post-processor of one of the preceding claims,

5

wherein the band extractor and the high band processor and the combiner are configured to operate in overlapping blocks, and

wherein the audio post-processor further comprises an overlap-adder for calculating a post-processed portion by adding audio samples of a first block and audio samples of a second block in a block overlap range.

10

5. Audio post-processor of one of the preceding claims, wherein the band extractor comprises:

15

an analysis windower for generating a sequence of blocks of sampling values of the audio signal using an analysis window, wherein the blocks are time-overlapping;

20

a discrete Fourier transform processor for generating a sequence of blocks of spectral values;

a low pass shaper for shaping each block of spectral values to obtain a sequence of low pass shaped blocks of spectral values;

25

a discrete Fourier inverse transform processor for generating a sequence of blocks of low pass time domain sampling values; and

a synthesis windower for windowing the sequence of blocks of low pass time domain sampling values using a synthesis window.

30

6. Audio post-processor of claim 5, wherein the band extractor further comprises:

an audio signal windower for windowing the audio signal using the analysis window and the synthesis window to obtain a sequence of windowed blocks of audio signal values, wherein the audio signal windower is synchronized with the

35

windower so that the sequence of blocks of low pass time domain sampling values is synchronous with the sequence of windowed blocks of audio signal values.

7. Audio post-processor of claim 5 or 6,

5

wherein the band extractor is configured to perform a sample-wise subtraction of the sequence of blocks of low pass time domain values from a corresponding sequence of blocks derived from the audio signal to obtain a sequence of blocks of high pass time domain sampling values.

10

8. Audio post-processor of claim 7,

wherein the high band processor is configured to apply the time-variable amplification to each sample of each block of the sequence of blocks of high pass time domain sampling values,

15

wherein the time-variable amplification for a sample of a block depends on

a gain information of a previous block and a gain information of a current block, or

20

a gain information of the current block and a gain information of the next block.

9. Audio post-processor of one of the claims 1 to 7, wherein the audio signal comprises an additional control parameter as a further side information, wherein the high band processor is configured to apply the time-variable amplification also under consideration of the additional control parameter, wherein a time resolution of the additional control parameter is lower than a time resolution of the time-variable high frequency gain information or the additional control parameter is stationary for a specific audio piece.

25

30

35

10. Audio post-processor of claim 8,

wherein the combiner is configured to perform a sample-wise addition of corresponding blocks of the sequence of blocks of low pass time domain sampling values and the sequence of amplified blocks of high pass time domain sampling values to obtain a sequence of blocks of combination signal values.

11. Audio post-processor of claim 10, further comprising:

an overlap-add processor for calculating a post-processed audio signal portion by adding audio samples of a first block of the sequence of combination signal values and audio samples of a neighboring second block adjacent to the first block in a block overlap range.

12. Audio post-processor of one of the preceding claims,

wherein the band extractor, the high band processor and the combiner operate in overlapping blocks, wherein an overlap range is between 40% of a block length and 60% of a block length, or

wherein a block length is between 0.8 milliseconds and 5 milliseconds, or

wherein the time-variable amplification performed by the high band processor is an multiplicative factor applied to each sample of a block in a time domain, or

wherein a cutoff or corner frequency of the low frequency band is between $1/8$ and $1/3$ of a maximum frequency of the audio signal and preferably equal to $1/6$ of the maximum frequency of the audio signal.

13. Audio post-processor of claim 5,

wherein the low pass shaper is configured to apply a shaping function depending on the time-variable high frequency gain information for a corresponding block.

14. Audio post-processor of claim 13,

wherein the shaping function additionally depends on a shaping function used in an audio pre-processor for modifying or attenuating a high frequency band of the audio signal using the time-variable high frequency gain information for a corresponding block.

15. Audio post-processor of claim 8,

wherein the modification for a sample of a block additionally depends on a windowing factor applied for a certain sample as defined by the analysis window function or the synthesis window function.

16. Audio post-processor of one of the preceding claims, wherein the band extractor, the high band processor and the combiner are configured to process sequences of blocks derived from the audio signal as overlapping blocks, so that a later portion of an earlier block is derived from the same audio samples of the audio signal as an earlier portion of a later block being adjacent in time to the earlier block.

17. Audio post-processor of claim 16, wherein an overlap range of the overlapping blocks is equal to one half of the earlier block and wherein the later block has the same length as the earlier block with respect to a number of sample values, and wherein the post processor additionally comprises an overlap adder for performing the overlap add operation.

18. Audio post-processor of claim 16 or 17, wherein the band extractor is configured to apply a slope of a splitting filter between a stop range and a pass range of the splitting filter to a block of audio samples, wherein the slope depends on the time-variable high frequency gain information for the block of samples.

19. Audio post-processor of claim 18,

wherein the high frequency gain information comprises gain values, wherein the slope is increased stronger for a higher gain value compared to an increase of the slope for a lower gain value.

20. Audio post-processor of one of claims 17 to 19,

wherein the slope of the splitting filter is defined based on the following equation:

5

$$rs[f] = 1 - (1 - ps[f]) \times \frac{g[k]}{1 + (g[k] - 1) \times (1 - ps[f])}$$

wherein $rs[f]$ is the slope of the splitting filter, wherein $ps[f]$ is a slope of splitting filter used when generating the audio signal, wherein $g[k]$ is a gain factor derived from the time-variable high frequency gain information, wherein f is a frequency index and wherein k is a block index.

10

21. Audio post-processor of any one of the claims 16 to 20,

wherein the high frequency gain information comprises gain values for adjacent blocks, wherein the high band processor is configured to calculate a correction factor for each sample depending on the gain values for the adjacent blocks and depending on window factors for corresponding samples.

15

22. Audio post-processor of claim 21, wherein the high band processor is configured to operate based on the following equations:
- 20

$$\text{corr}[j] = 1 + \left(\frac{g[k-1]}{g[k]} + \frac{g[k]}{g[k-1]} - 2 \right) \times w^2[j] \times (1 - w^2[j]), \text{ for } 0 \leq j < \frac{N}{2},$$
$$\text{corr}\left[j + \frac{N}{2}\right] = 1 + \left(\frac{g[k]}{g[k+1]} + \frac{g[k+1]}{g[k]} - 2 \right) \times w^2[j] \times (1 - w^2[j]), \text{ for } 0 \leq j < \frac{N}{2}.$$

wherein $\text{corr}[j]$ is a correction factor for a sample with an index j , wherein $g[k-1]$ is a gain factor for a preceding block, wherein $g[k]$ is a gain factor a current block, wherein $w[j]$ is a window function factor for a sample with a sample index j , wherein N is the length in samples of a block and wherein $g[k+1]$ is the gain factor for the later block, wherein k is the block index and wherein the upper equation from the above equations is for a first half of an output block k , and wherein the lower equation of the above equations is for a second half of the output block k .

25

30

23. Audio post-processor of one of claims 17 to 22,

wherein the high band processor is configured to additionally compensate for an attenuation of transient events introduced into the audio signal by a processing performed before a processing by the audio post-processor.

24. Audio post-processor of claim 23,

wherein the high band processor is configured to operate based on the following equation:

$$gc[k] = (1 + \text{beta_factor}) \times g[k] - \text{beta_factor}$$

wherein $gc[k]$ is the compensated gain for a block with a block index k , wherein $g[k]$ is a non-compensated gain as indicated by the time-variable high frequency gain information included as the side information and wherein beta_factor is an additional control parameter value included within the side information.

25. Audio post-processor of one of claims 22 and 24, wherein the high band processor is configured to calculate the processed high band based on the following equation:

$$\text{phpb}[k][i] = \frac{1}{gc[k]} \times \frac{1}{\text{corr}[i]} \times \text{hpb}[k][i], \text{ for } 0 \leq i < N$$

wherein $\text{phpb}[k][i]$ indicates the processed high band for a block k and a sample value i , wherein $gc[k]$ is the compensated gain, wherein $\text{corr}[i]$ is a correction factor, wherein k is a block index, wherein i is a sampling value index and wherein $\text{hpb}[k][i]$ is the high band for a block and a sampling value i and wherein N is the length in samples of a block.

26. Audio post-processor of claim 25,

wherein the combiner is configured to calculate the combined block as

$$\text{ob}[k][i] = \text{lpb}[k][i] + \text{phpb}[k][i],$$

wherein $lpb[k][i]$ is the low frequency band for a block k and a sample index i .

27. Audio post-processor of one of the claims 16 to 26, further comprising an overlap-adder operating based on the following equation:

$$\begin{aligned} o\left[k \times \frac{N}{2} + j\right] &= ob[k-1]\left[j + \frac{N}{2}\right] + ob[k][j], \text{ for } 0 \leq j < \frac{N}{2} \\ o\left[(k+1) \times \frac{N}{2} + j\right] &= ob[k]\left[j + \frac{N}{2}\right] + ob[k+1][j], \text{ for } 0 \leq j < \frac{N}{2} \end{aligned}$$

wherein $o[]$ is a value of a sample of a post-processed audio output signal for a sample index derived from k and j , wherein k is a block value, N is the length in samples of a block, j is a sampling index within a block and $ob[]$ indicates a combined block for the earlier block index $k-1$, the current block index k or a later block index $k+1$.

28. Audio post-processor of one of claims 1 to 27, wherein the time-variable high frequency gain information comprises a sequence of gain indices and a gain precision information or wherein the side information additionally comprises a gain compensation information and a gain compensation precision information,

wherein the audio post-processor comprises

a decoder for decoding the gain indices depending on the gain precision information to obtain a decoded gain of a first number of different values for a first precision information or a decoded gain of a second number of different values for a second precision information, the second number being greater than the first number, or

a decoder for decoding the gain compensation indices depending on the gain compensation precision information to obtain a decoded gain compensation value of a first number of different values for a first gain compensation precision information or a decoded gain compensation value of a second different number of values for a second different gain compensation precision information, the first number being greater than the second number.

29. Audio post-processor of claim 28,

wherein the decoder is configured to calculate a gain factor for a block:

5
$$g[k] = 2^{\frac{\text{gainIdx}[k][\text{sig}] - \text{GAIN_INDEX_0dB}}{4}},$$

wherein $g[k]$ is the gain factor for a block with a block index k , wherein $\text{gainIdx}[k][\text{sig}]$ is a quantized value included in the side information as the time-variable high frequency gain information, and wherein GAIN_INDEX_0dB is a gain index offset corresponding to 0dB with a first value when the gain precision information has a first status and with the second different value when the gain precision information has a second status.

30. Audio post-processor of one of the preceding claims,

wherein the band extractor is configured to perform a block wise discrete Fourier transform with a block length of N sampling values to obtain a number of spectral values being lower than a number of $N/2$ complex spectral values by performing a sparse discrete Fourier transform algorithm in which calculations of branches for spectral values above a maximum frequency are skipped, and

wherein the band extractor is configured to calculate the low frequency band signal by using the spectral values up to a transition start frequency range and by weighting spectral values within the transition start frequency range, wherein the transition start frequency range only extends until the maximum frequency or a frequency being smaller than the maximum frequency.

31. Audio post-processor of one of claims 1 to 30,

being configured to only perform a post-processing with a maximum number of channels or objects, for which side information for the time-variable amplification of the high frequency band is available and to not perform any post-processing with a number of channels or objects for which any side information for the time-variable amplification of the high frequency band is not available, or

wherein the band extractor is configured to not perform any band extraction or to not compute a Discrete Fourier Transform and inverse Discrete Fourier Transform pair for trivial gain factors for the time-variable amplification of the high frequency band, and to pass through an unchanged or windowed time domain signal associated with the trivial gain factors.

32. Audio pre-processor for pre-processing an audio signal, comprising:

a signal analyzer for analyzing the audio signal to determine a time-variable high frequency gain information;

a band extractor for extracting a high frequency band of the audio signal and a low frequency band of the audio signal;

a high band processor for performing a time-variable modification of the high frequency band in accordance with the time-variable high frequency gain information to obtain a processed high frequency band;

a combiner for combining the processed high frequency band and the low frequency band to obtain a pre-processed audio signal; and

an output interface for generating an output signal comprising the pre-processed audio signal and the time-variable high frequency gain information as side information.

33. Audio pre-processor of claim 32,

wherein the signal analyzer is configured to analyze the audio signal to determine a first characteristic in a first time block of the audio signal and a second characteristic in a second time block of the audio signal, the second characteristic being more transient than the first characteristic or being a larger high frequency energy level than the first characteristic,

wherein the signal analyzer is configured to determine a first gain information for the first characteristic and a second gain information for the second characteristic, and

wherein the high band processor is configured to apply a multiplicative factor to the high band portion of the second time block in accordance with the second gain information stronger than the high band portion of the first time block in accordance with the first gain information.

34. Audio pre-processor of any of claims 32 to 33, wherein the signal analyzer is configured:

to calculate a background measure for a background energy of the high band for one or more time blocks neighboring in time placed before the current time block or placed subsequent to the current time block or placed before and subsequent to the current time block or including the current time block or excluding the current time block;

to calculate an energy measure for a high band of the current block; and

to calculate a gain factor using the background measure and the energy measure.

35. Audio pre-processor of claim 33, wherein the signal analyzer is configured to calculate the gain factor based on the following equation:

$$g_{float}[k] = \begin{cases} \frac{\alpha \times hp_bg_e[k] + (1 - \alpha) \times hp_e[k]}{hp_e[k]}, & \text{when } hp_e[k] \geq T_{quiet}, \\ 1, & \text{otherwise} \end{cases}$$

wherein g_{float} is a non-quantized gain factor, wherein k is a block index, wherein α is a variation influencing factor, wherein $hp_bg_e[k]$ is the background energy of the high band for a block k , wherein $hp_e[k]$ is the energy of the high band of the current block, wherein T_{quiet} is a quiet threshold, and wherein the factor α and the quiet threshold are predetermined or controllable by additional control parameters.

36. Audio pre-processor of one of the claims 32 to 35, wherein the signal analyzer and the high band processor are configured to calculate the time-variable high frequency gain information and to apply the time-variable high frequency gain

information so that a variation of an energy of each block around a corresponding average energy of a background is reduced by at least 50% and preferably by 75%.

5 37. Audio pre-processor of one of claims 32 to 36,

wherein the signal analyzer is configured to quantize and clip a raw sequence of gain information values to obtain the time-variable high frequency gain information as a sequence of quantized values,

10

wherein the high band processor is configured to perform the time-variable modification of the high band in accordance with the sequence of quantized values, and

15

wherein the output interface is configured to introduce the sequence of quantized values into the side information as the time-variable high frequency gain information.

20 38. Audio pre-processor of one of claims 32 to 37, wherein the audio pre-processor is configured

to determine a further gain compensation value describing a loss of an energy variation introduced by a subsequently connected encoder or decoder, and

25

to quantize the further gain compensation information, and

wherein the output interface is configured to introduce the quantized further gain compensation information into the side information.

30

39. Audio pre-processor of one of claims 32 to 38, wherein the signal analyzer is configured to apply a meta gain control which controls a modification of the time-variable high frequency gain information to gradually reduce or gradually enhance an effect of the high band processor on the audio signal in accordance with additional control data additionally derived from the audio signal, or

35

where in the signal analyzer is configured to influence a factor alpha when calculating the gain information based on the following equation, wherein increasing alpha results in a stronger influence and reducing alpha results in a lower influence,

5

$$g_{float}[k] = \begin{cases} \frac{\alpha \times hp_bg_e[k] + (1 - \alpha) \times hp_e[k]}{hp_e[k]}, & \text{when } hp_e[k] \geq T_{quiet}, \\ 1, & \text{otherwise} \end{cases}$$

wherein g_{float} is a non-quantized gain factor, wherein k is a block index, wherein $hp_bg_e[k]$ is the high frequency background energy for a block k , wherein $hp_e[k]$ is the energy of the high frequency block, wherein T_{quiet} is a quiet threshold, and wherein the factor α and the quiet threshold are predetermined or controllable by additional control parameters.

10

15 40. Audio pre-processor of one of claims 32 to 39, in which the band extractor is configured to extract the low frequency band using a low pass filter device and to extract the high frequency band by subtracting the low frequency band from the audio signal.

20 41. Audio pre-processor of one of claims 32 to 40,

in which the time-variable high frequency gain information is provided for a sequence of blocks of sampling values of the audio signal so that a first block of sampling values has associated therewith a first gain information and a second later block of sample values of the audio signal has a different second gain information, wherein the band extractor is configured to extract, from the first block of sampling values, a first low frequency band and a first high frequency band and to extract, from the second block of sampling values, a second low frequency band and a second high frequency band, and

25

30

wherein the high band processor is configured to modify the first high frequency band using the first gain information to obtain a first processed high frequency band and to modify the second high frequency band using the second gain information to obtain a second processed high frequency band, and

wherein the combiner is configured to combine the first low frequency band and the first processed high frequency band to obtain a first combined block and to combine the second low frequency band and the second processed high frequency band to obtain a second combined block.

42. Audio pre-processor in accordance with one of claims 32 to 41,

wherein the band extractor and the high band processor and the combiner are configured to operate in overlapping blocks, and

wherein the combiner further comprises an overlap-adder for calculating a pre-processed portion by adding audio samples of a first block and audio samples of a second block in a block overlap range, or

wherein the band extractor, the high band processor and the combiner operate in overlapping blocks, wherein an overlap range is between 40% of a block length and 60% of a block length, or

wherein a block length is between 0.8 milliseconds and 5 milliseconds, or

wherein the modification performed by the high band processor is an attenuation applied to each sample of a block in a time domain, or

wherein a cutoff or corner frequency of the low frequency band is between $1/8$ and $1/3$ of a maximum frequency of the audio signal and preferably equal to $1/6$ of the maximum frequency of the audio signal.

43. Audio pre-processor in accordance with one of claims 32 to 42, wherein the band extractor comprises:

an analysis windower for generating a sequence of blocks of sampling values of the audio signal using an analysis window, wherein the blocks are time-overlapping;

a discrete Fourier transform processor for generating a sequence of blocks of spectral values;

a low pass shaper for shaping each block of spectral values to obtain a sequence of low pass shaped blocks of spectral values;

a discrete Fourier inverse transform processor for generating a sequence of blocks of low pass time domain sampling values; and

a synthesis windower for windowing the sequence of blocks of low pass time domain sampling values using a synthesis window.

44. Audio pre-processor in accordance with claim 43, wherein the low pass shaper is configured to operate based on the following equation:

$$ps[f] = \begin{cases} 1, & \text{for } 0 \leq f < lp_size \\ 1 - \frac{f - lp_size + 1}{tr_size + 1}, & \text{for } lp_size \leq f < lp_size + tr_size \\ 0, & \text{for } lp_size + tr_size \leq f \leq \frac{N}{2} \end{cases}$$

wherein $ps[f]$ indicates the shaping factor of the shaping to be applied by the low pass shaper for a frequency value f , wherein f is a frequency value, wherein tr_size is a value determining the width in spectral lines of a transition region, wherein a lp_size indicates a size of a low pass portion without the transition region, wherein N indicates a number of sampling values for a block.

45. Audio pre-processor of one of claims 43 or 44, wherein the block extractor further comprises:

an audio signal windower for windowing the audio signal using the analysis window and the synthesis window to obtain a sequence of windowed blocks of audio signal values, wherein the audio signal windower is synchronized with the windower so that the sequence of blocks of low pass time domain sampling values is synchronous with the sequence of windowed blocks of audio signal values.

46. Audio pre-processor of one of claims 43 to 45,

wherein the band extractor is configured to perform a sample-wise subtraction of the sequence of blocks of low pass time domain values from a corresponding sequence of blocks derived from the audio signal to obtain a sequence of blocks of high pass time domain sampling values.

47. Audio pre-processor of one of claims 32 to 46, wherein the analyzer additionally provides a control parameter computed by meta gain control, wherein the high band processor is configured to apply the modification also under consideration of the control parameter.

48. Audio pre-processor of one of claims 43 to 47,

wherein the combiner is configured to perform a sample-wise addition of corresponding blocks of the sequence of blocks of low pass time domain sampling values and the sequence of modified blocks of high pass time domain sampling values to obtain a sequence of blocks of combination signal values.

49. Audio pre-processor of claim 48, further comprising:

an overlap-add processor for calculating a pre-processed audio signal portion by adding audio samples of a first block of the sequence of combination signal values and audio samples of a neighboring second block adjacent to the first block in a block overlap range.

50. Audio pre-processor of one of claims 32 to 49, wherein

wherein the band extractor, the high band processor and the combiner operate in overlapping blocks, wherein an overlap range is between 40% of a block length and 60% of a block length, or

wherein a block length is between 0.8 milliseconds and 5 milliseconds, or

wherein the modification performed by the high band processor is an multiplicative factor applied to each sample of a block in time domain.

51. Audio pre-processor of one of claims 32 to 50,

wherein an overlap range of the overlapping blocks is equal to one half of the earlier block and wherein the later block has the same length as the earlier block with respect to a number of sample values, and wherein the combiner comprises and overlap adder for performing the overlap add operation.

52. Audio pre-processor of one of claims 32 to 51,

being configured to only perform a preprocessing with a maximum number of channels or objects to generate the side information for the maximum number of channels or objects and not perform any preprocessing with a number of channels or objects for which the side information is not generated, or

wherein the band extractor is configured to not perform any band extraction or to not compute a Discrete Fourier Transform and inverse Discrete Fourier Transform pair for trivial gain factors for the time-variable high frequency gain information determined by the signal analyzer and to pass through an unchanged or windowed time domain signal associated with the trivial gain factors.

53. Audio encoding apparatus for encoding an audio signal, comprising:

the audio pre-processor of any one of claims 32 to 52, configured to generate the output signal having the time-variable high frequency gain information as side information;

a core encoder for generating a core encoded signal and core side information; and

an output interface for generating an encoded signal comprising the core encoded signal, the core side information and the time-variable high frequency gain information as additional side information.

54. Audio encoding apparatus of claim 53, wherein the audio signal is a multichannel or multi-object signal, wherein the audio pre-processor is configured to pre-process each channel or each object separately and wherein the core encoder is

configured to apply a joint multichannel encoder processing or a joint multi object encoder processing or an encoder bandgap filling or an encoder bandwidth extension processing on the pre-processed channels.

5 55. Audio decoding apparatus, comprising:

an input interface for receiving an encoded audio signal comprising a core encoded signal, core side information and a time-variable high frequency gain information as additional side information;

10

a core decoder for decoding the core encoded signal using the core side information to obtain a decoded core signal; and

15

a post-processor for post-processing the decoded core signal using the time-variable high frequency gain information in accordance with any one of the claims 1 to 31.

56. Audio decoding apparatus in accordance with claim 55,

20

wherein the core decoder is configured to apply a multichannel decoder processing or a multi object decoder processing or a bandwidth extension decoder processing or a gap filling decoder processing for generating decoded channels of a multichannel signal or decoded objects of a multi object signal, and

25

wherein the post-processor is configured to apply the post-processing individually on each channel or each object using the individual time-variable high frequency gain information for each channel or each object.

57. Method of post-processing an audio signal having a time-variable high frequency gain information as side information, comprising:

30

extracting a high frequency band of the audio signal and a low frequency band of the audio signal;

performing a time-variable modification of the high band in accordance with the time-variable high frequency gain information to obtain a processed high frequency band; and

5 combining the processed high frequency band and the low frequency band.

58. Method of pre-processing an audio signal, comprising:

10 analyzing the audio signal to determine a time-variable high frequency gain information;

extracting a high frequency band of the audio signal and a low frequency band of the audio signal;

15 performing a time-variable modification of the high frequency band in accordance with the time-variable high frequency gain information to obtain a processed high frequency band;

20 combining the processed high frequency band and the low frequency band to obtain a pre-processed audio signal; and

generating an output signal comprising the pre-processed audio signal and the time-variable high frequency gain information as side information.

25 59. Method of encoding an audio signal, comprising:

the method of audio pre-processing of claim 58 configured to generate the output signal having the time-variable high frequency gain information as side information;

30 generating a core encoded signal and core side information; and

generating an encoded signal comprising the core encoded signal, the core side information and the time-variable high frequency gain information as additional side information.

35

60. Method of audio decoding, comprising:

receiving an encoded audio signal comprising a core encoded signal, core side
information and a time-variable high frequency gain information as additional side
information;

decoding the core encoded signal using the core side information to obtain a
decoded core signal; and

post-processing the decoded core signal using the time-variable high frequency
gain information in accordance with the method of claim 57.

61. Computer program for performing, when running on a computer or a processor,
any one of the methods in accordance with claim 57, 58, 59, 60.

**Post-processor, Pre-processor, Audio Encoder, Audio Decoder and related
Methods for Enhancing Transient Processing**

Abstract

5

An audio post-processor (100) for post-processing an audio signal (102) having a time-variable high frequency gain information (104) as side information comprises: a band extractor (110) for extracting a high frequency band (112) of the audio signal (102) and a low frequency band (114) of the audio signal (102); a high band processor (120) for
10 performing a time-variable modification of the high frequency band (112) in accordance with the time-variable high frequency gain information (104) to obtain a processed high frequency band (122); and a combiner (130) for combining the processed high frequency band (122) and the low frequency band (114). Furthermore, a pre-processor is illustrated.

15

Figure 1

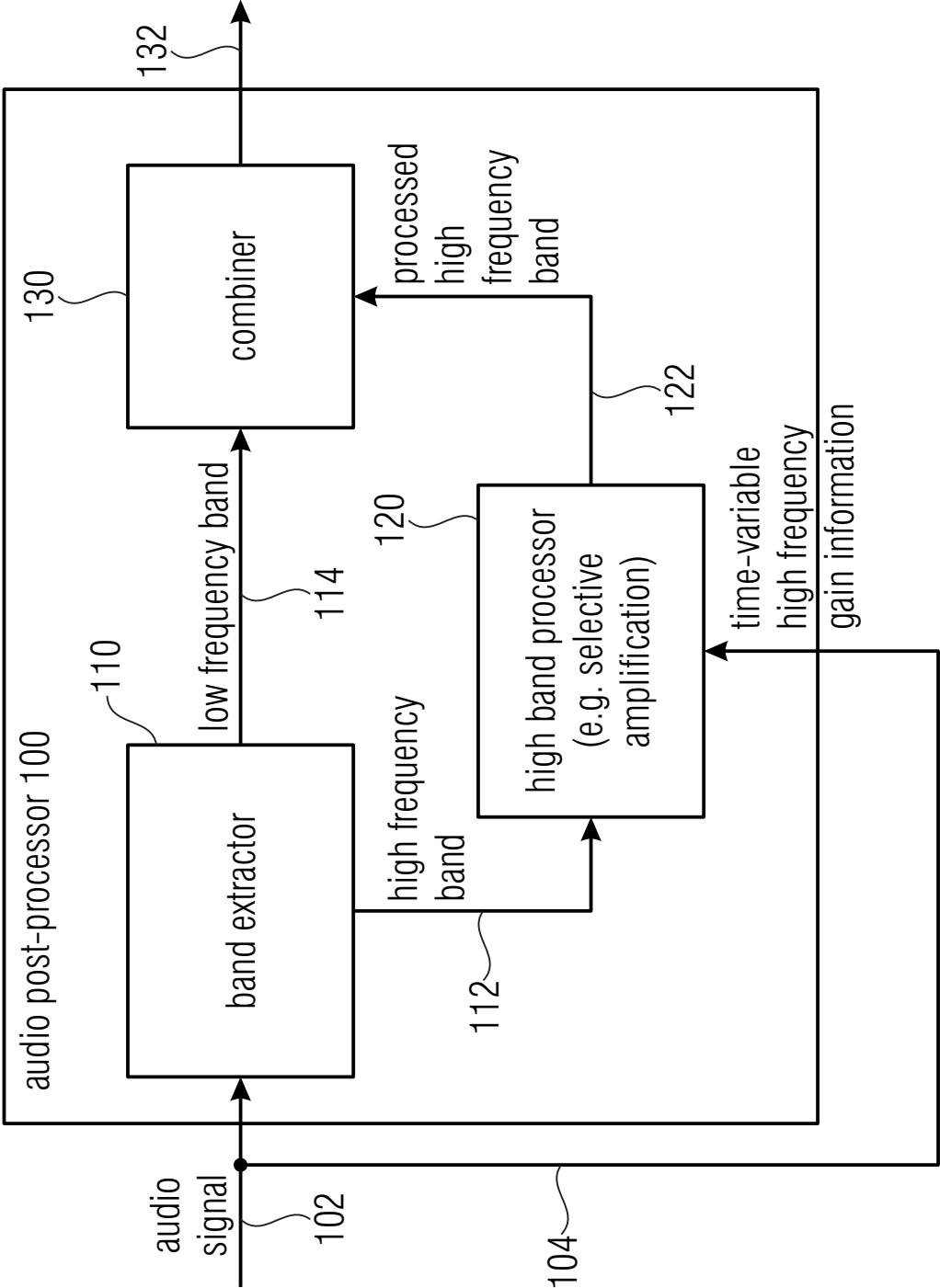


Fig. 1

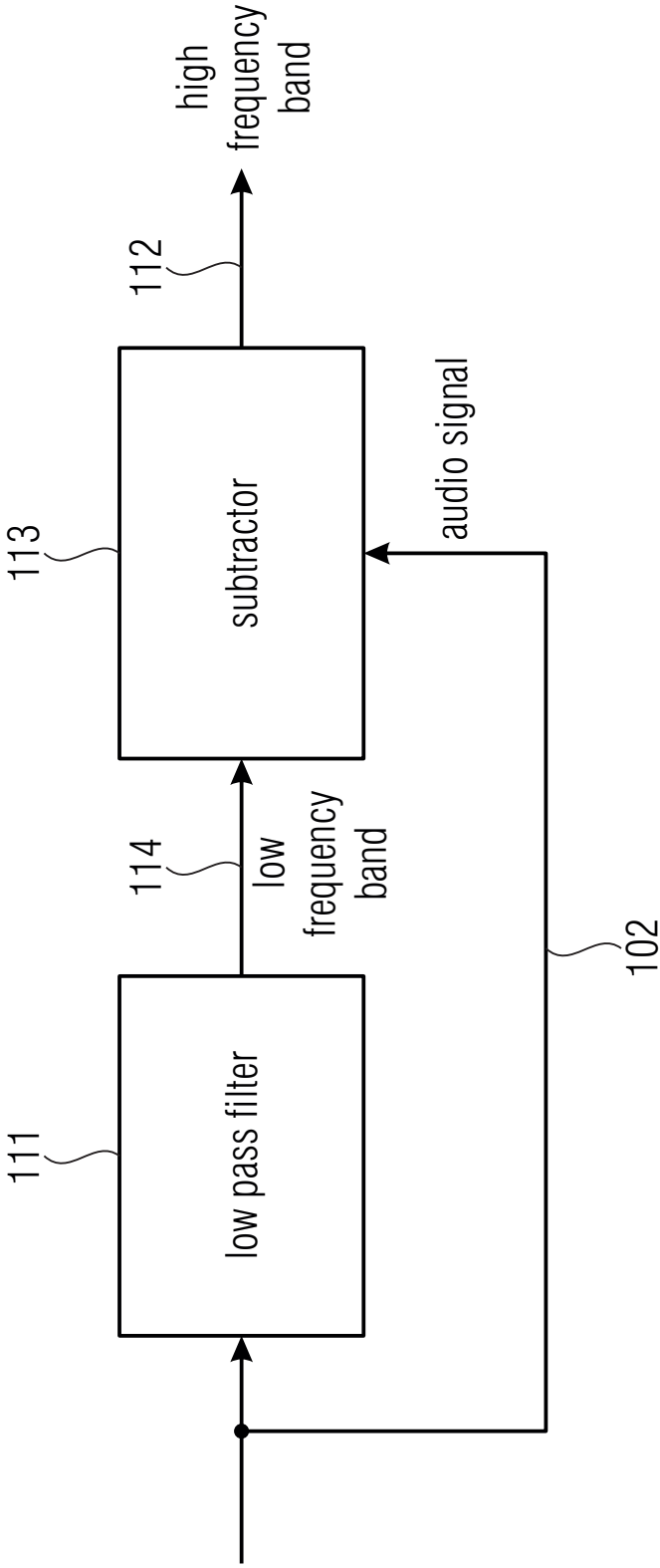


Fig. 2

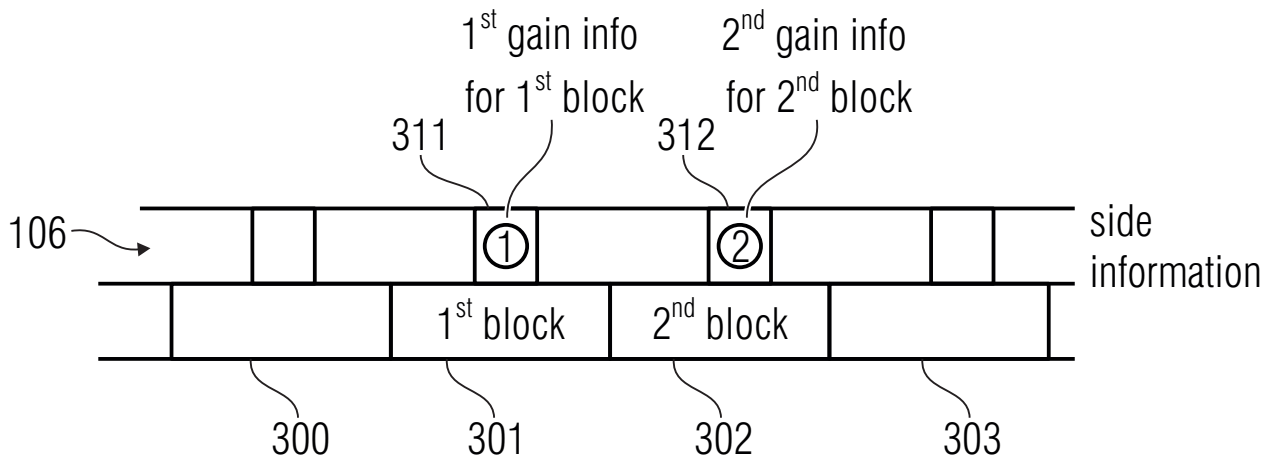


Fig. 3a

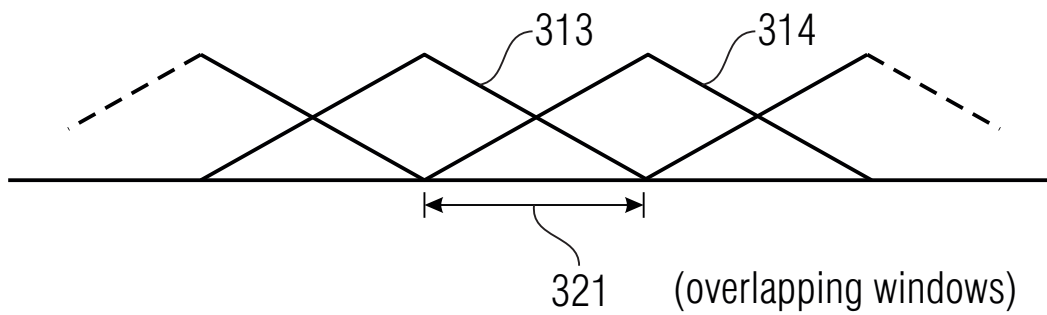


Fig. 3b

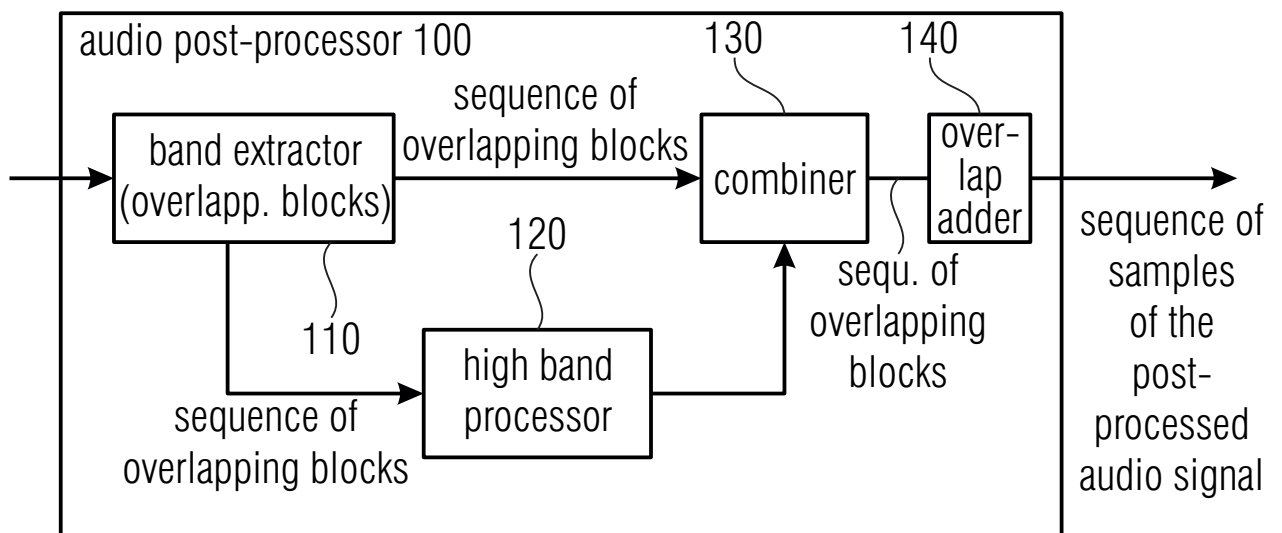


Fig. 3c

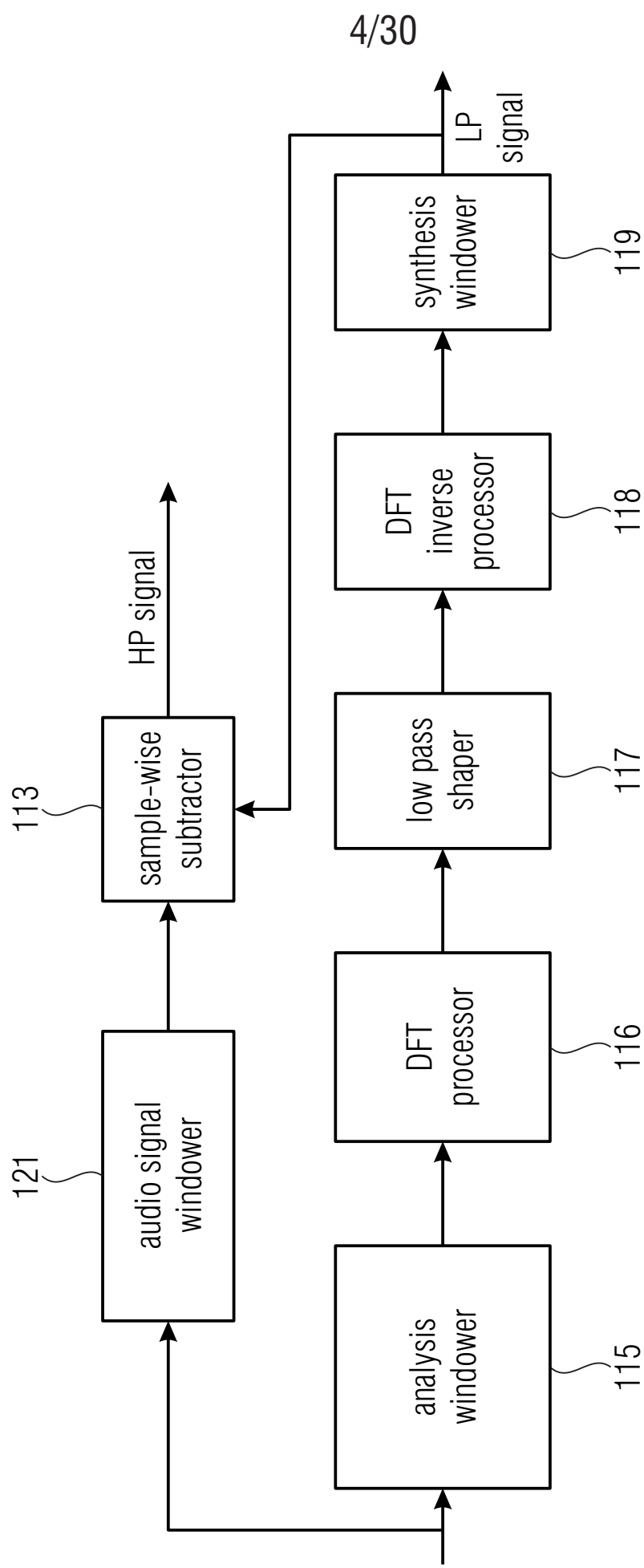


Fig. 4

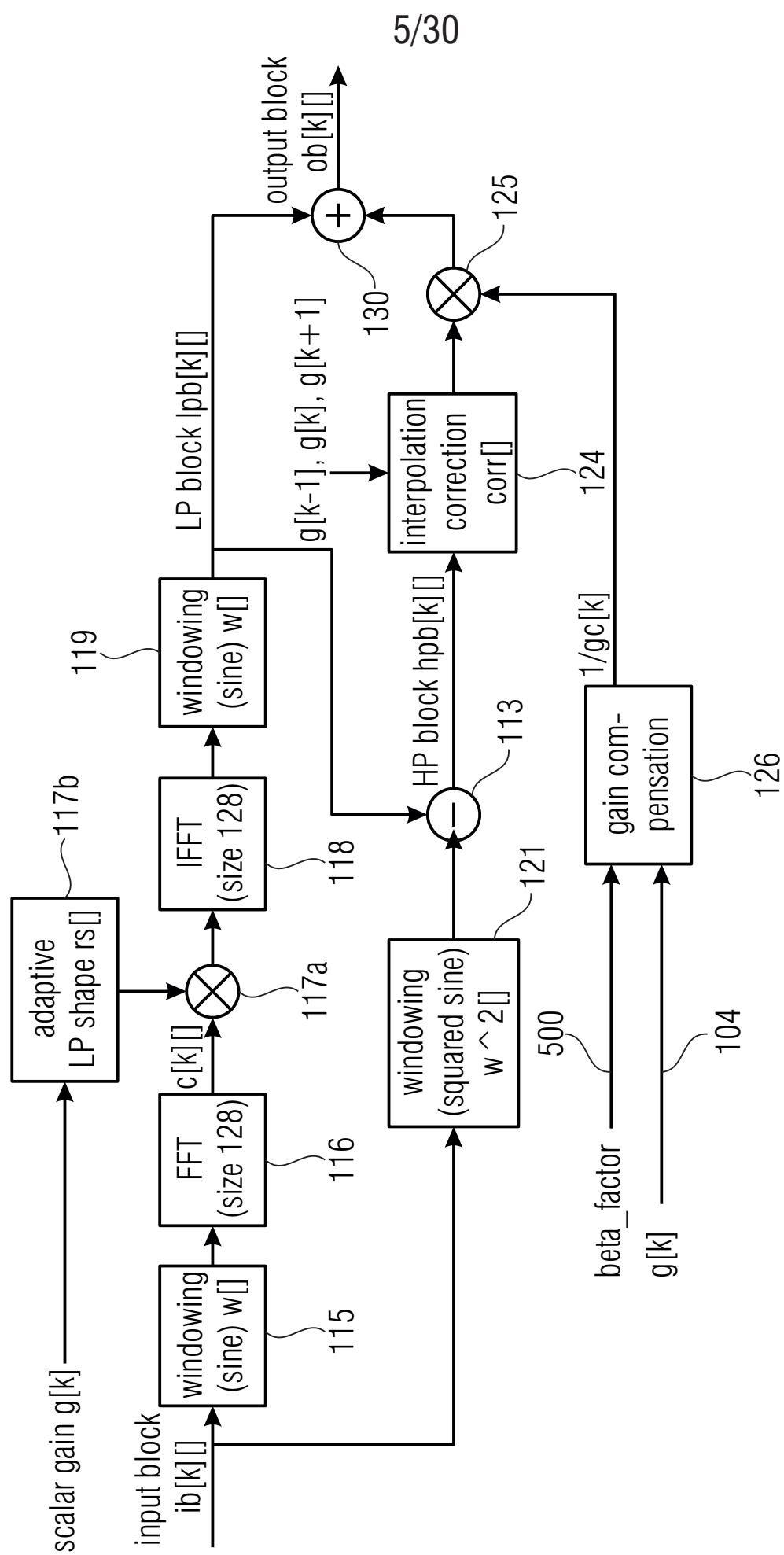


Fig. 5a

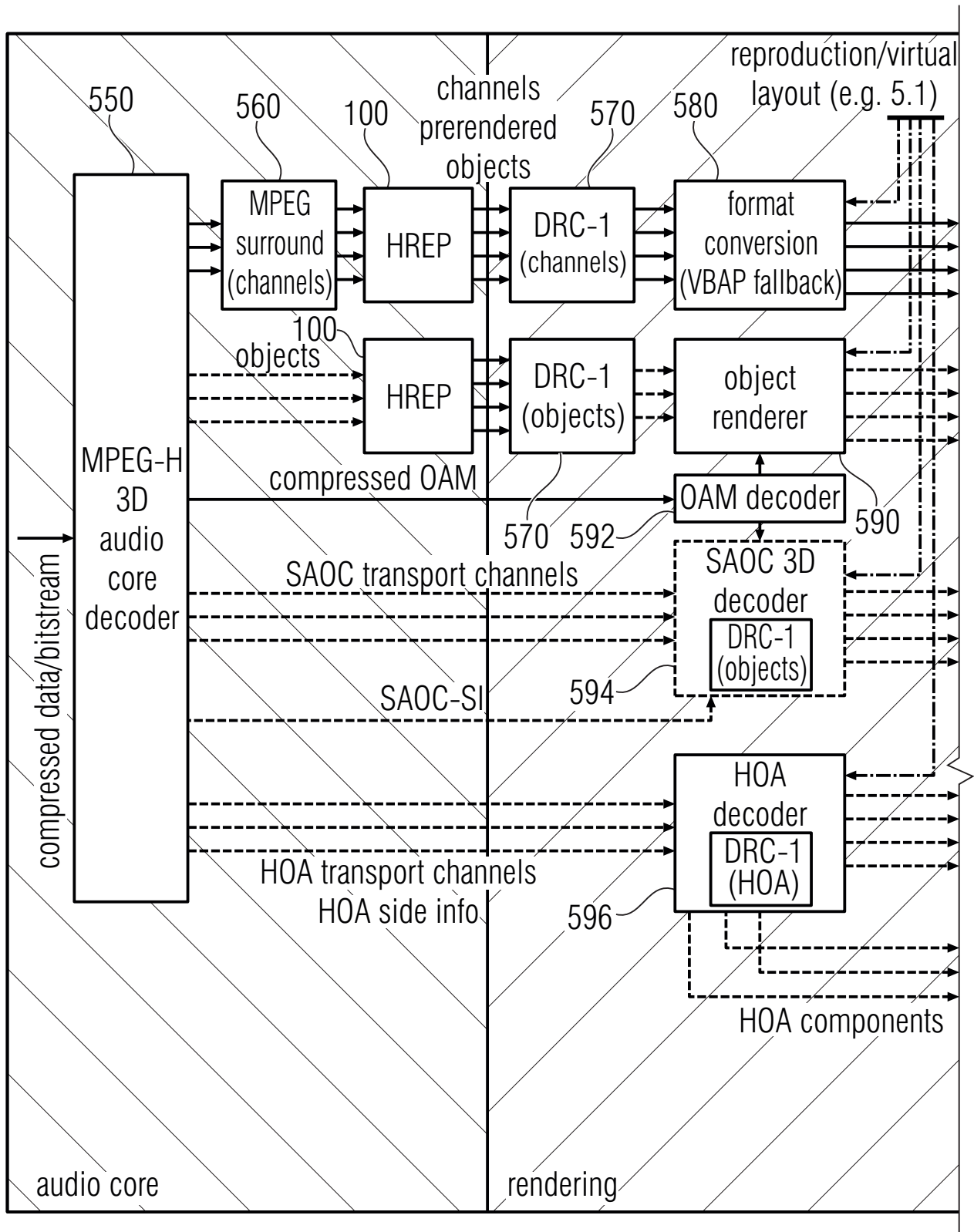


Fig. 5b1

Fig. 5b	Fig. 5b1
	Fig. 5b2

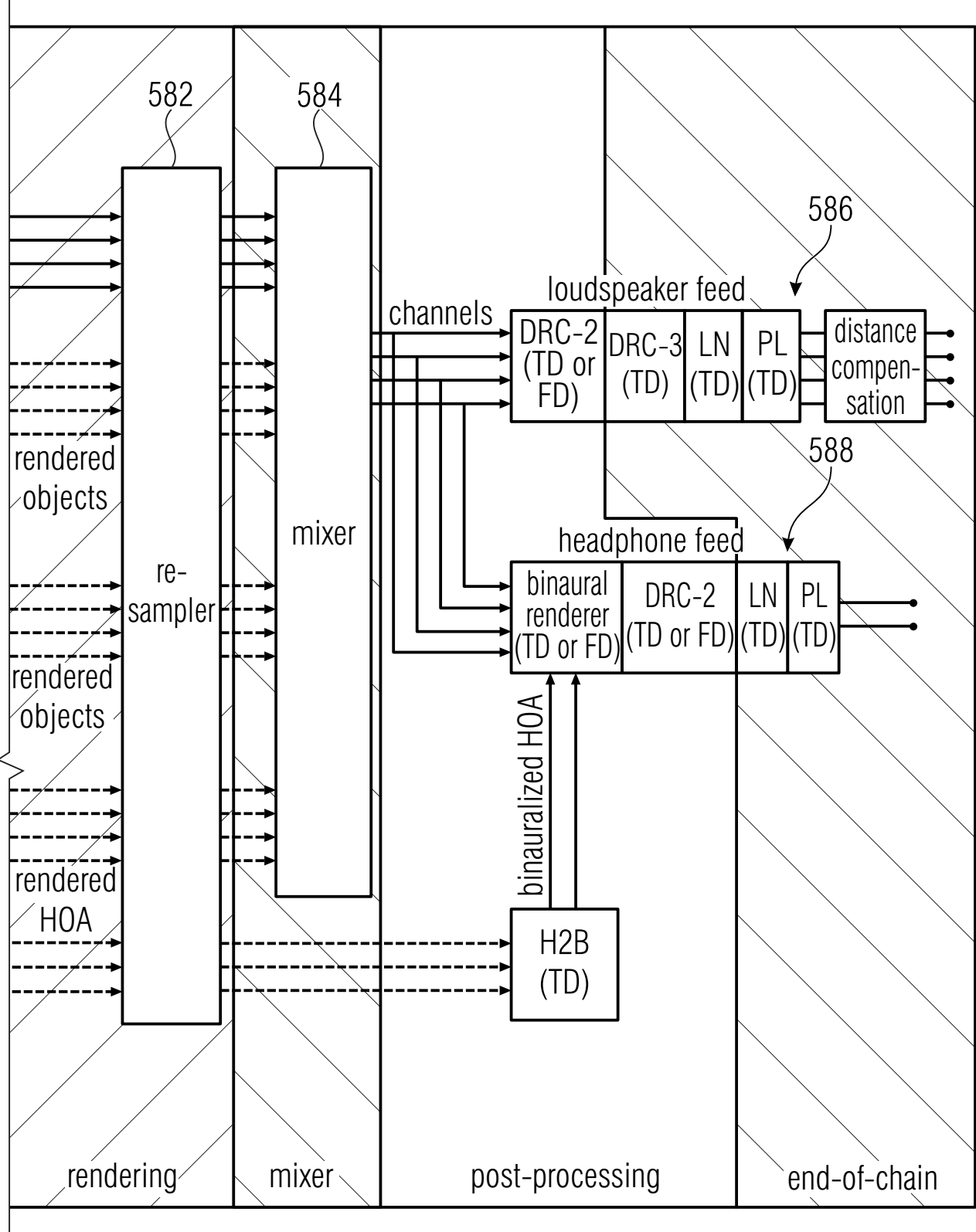


Fig. 5b2

Fig. 5b	Fig. 5b1
	Fig. 5b2

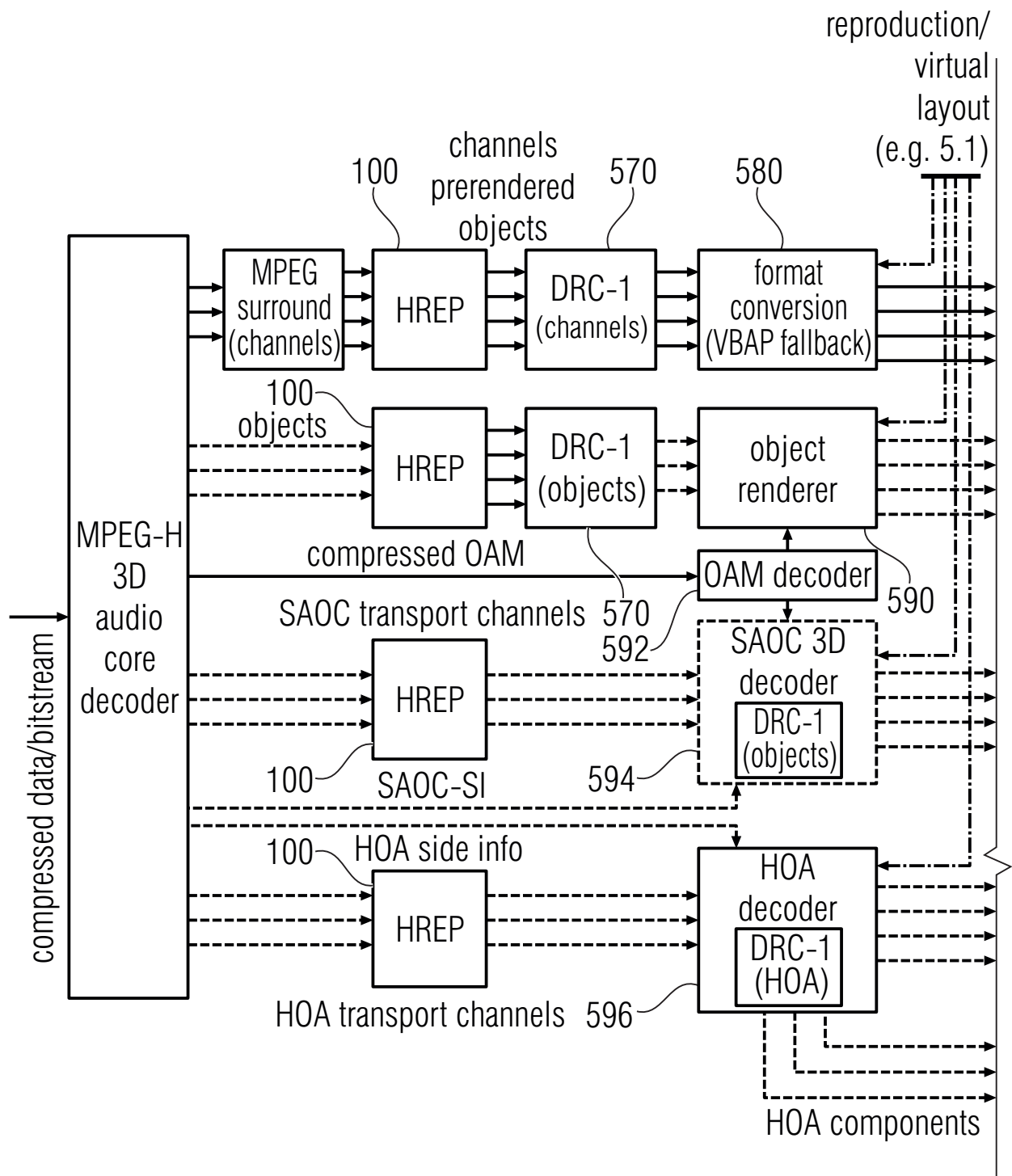


Fig. 5c1

Fig. 5c	Fig. 5c1
	Fig. 5c2

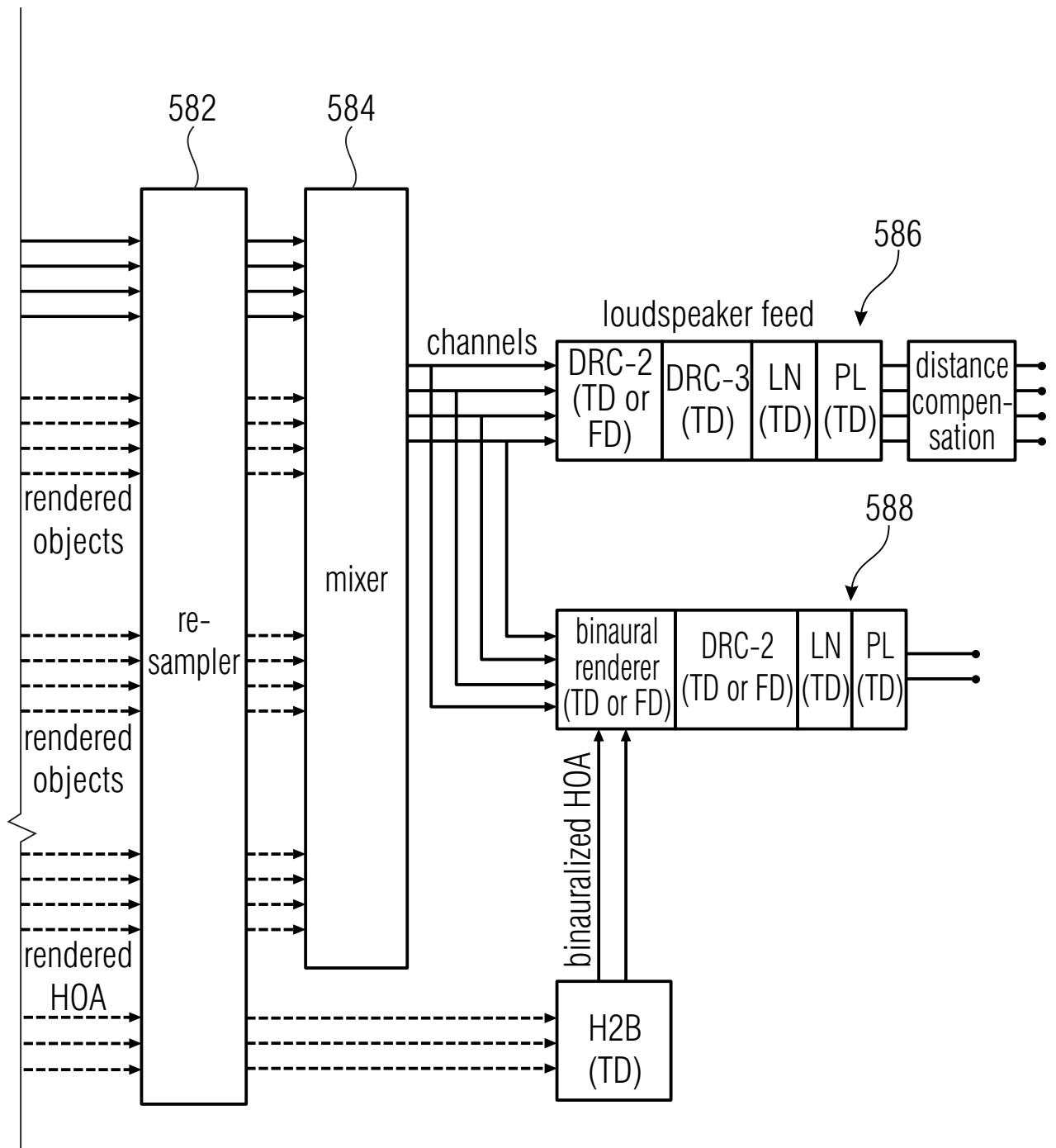


Fig. 5c2

Fig. 5c	Fig. 5c1
	Fig. 5c2

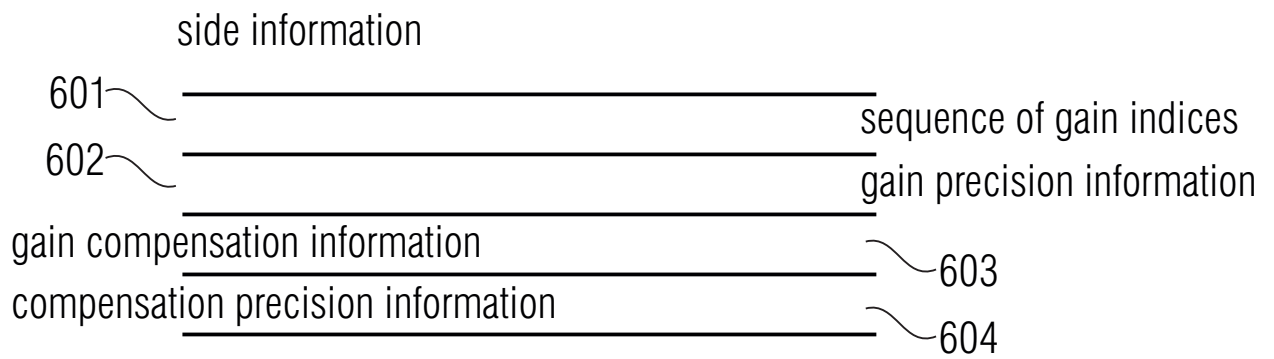


Fig. 6a

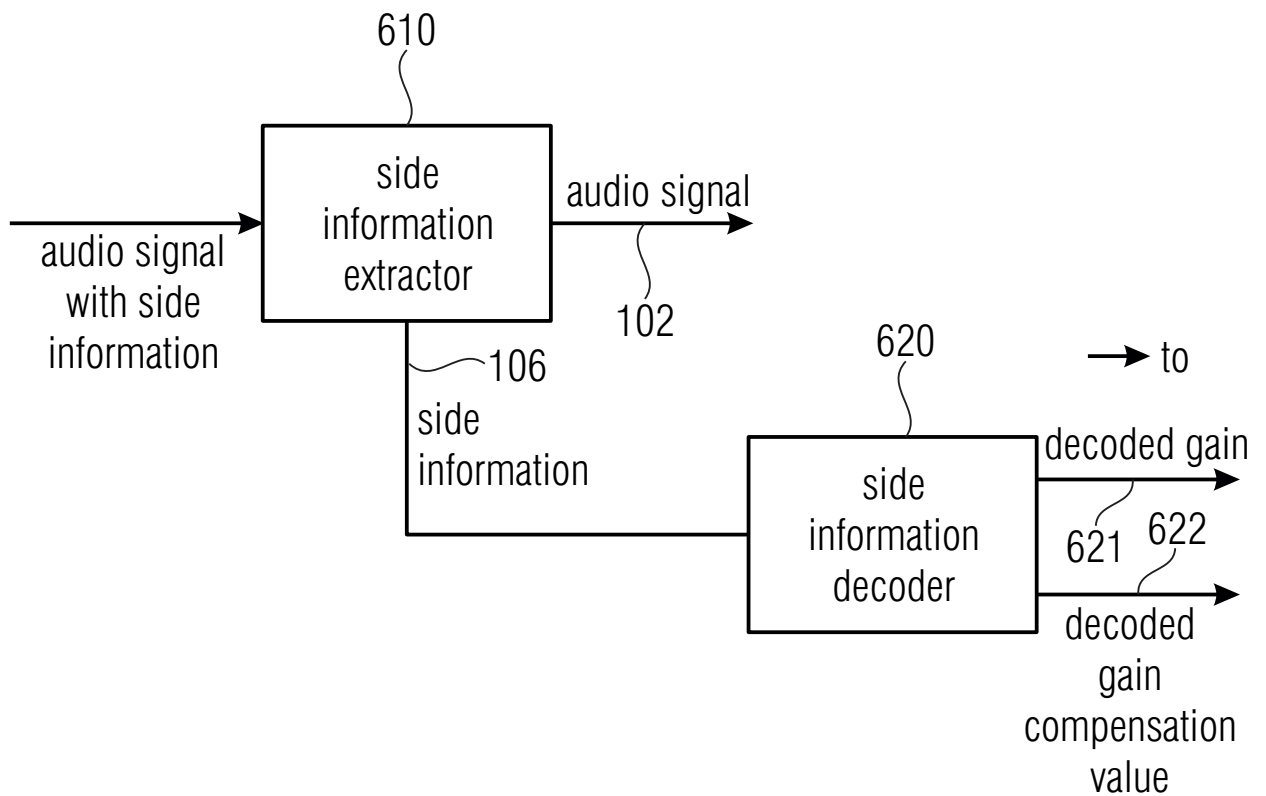


Fig. 6b

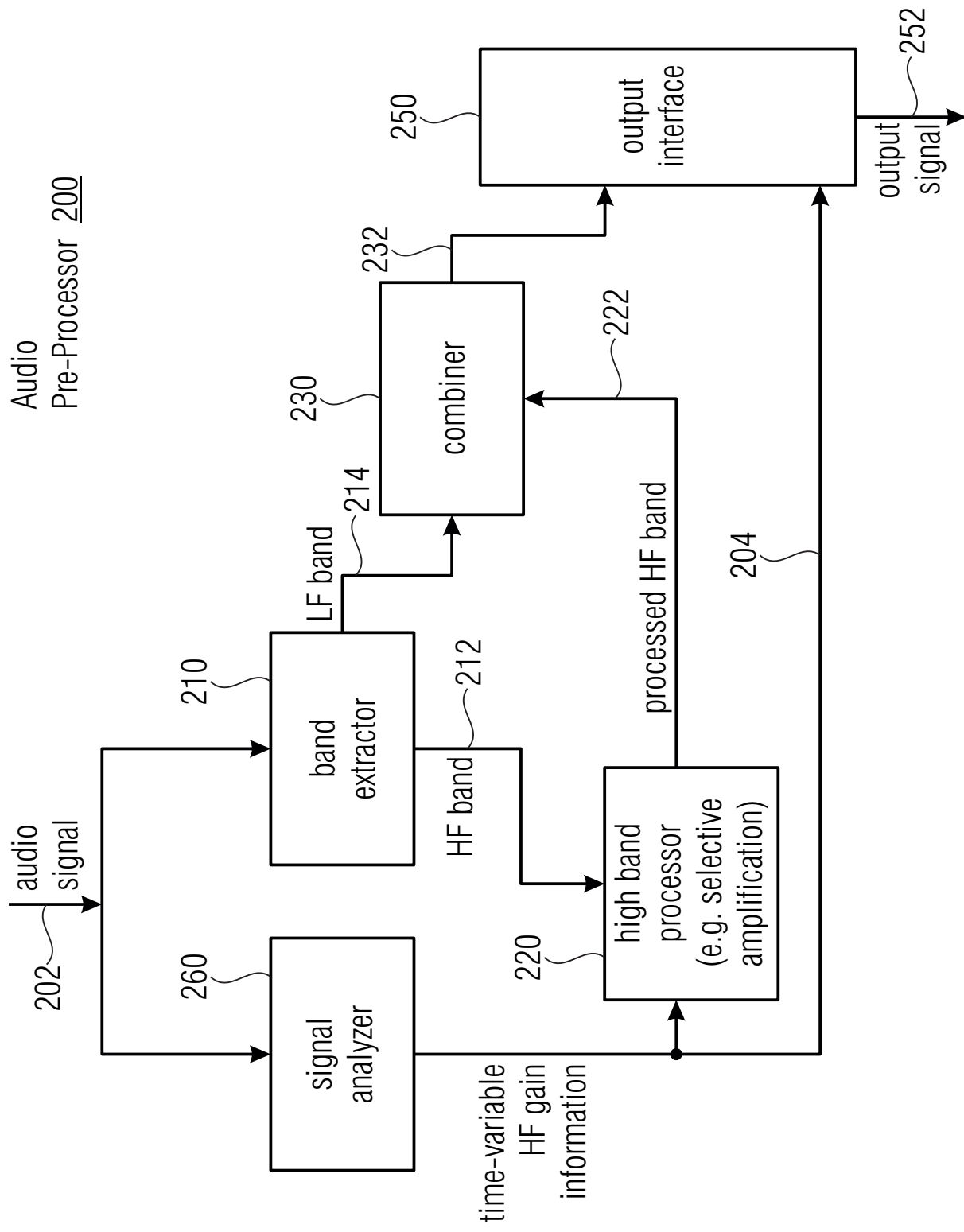


Fig. 7

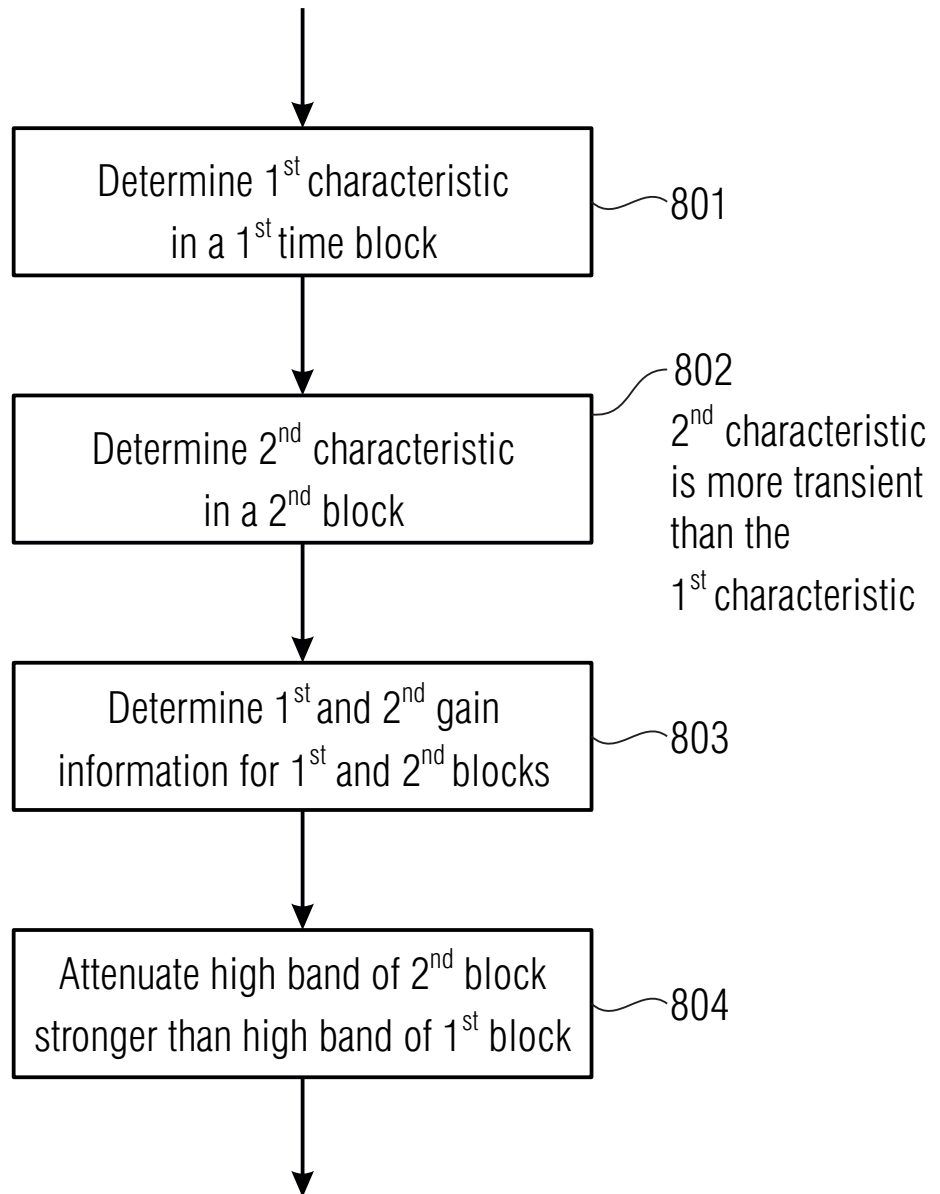


Fig. 8a

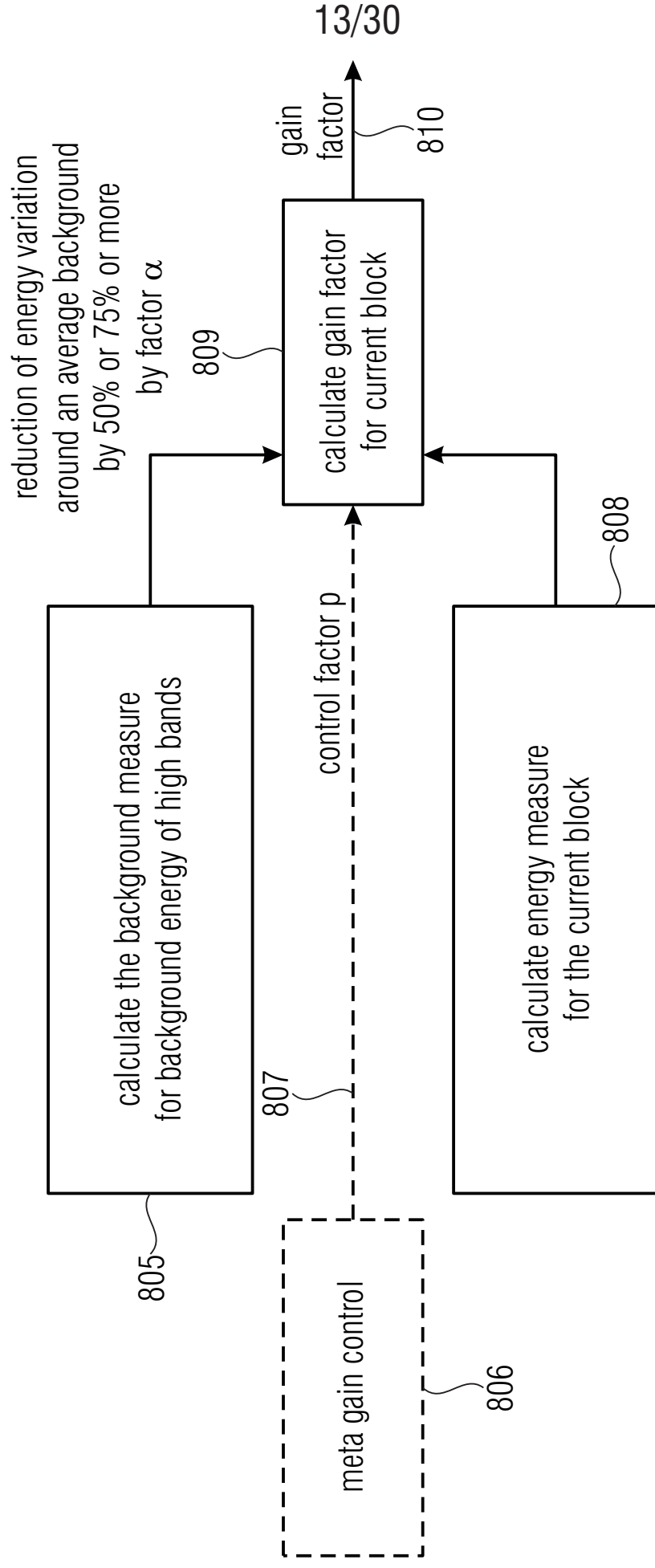


Fig. 8b

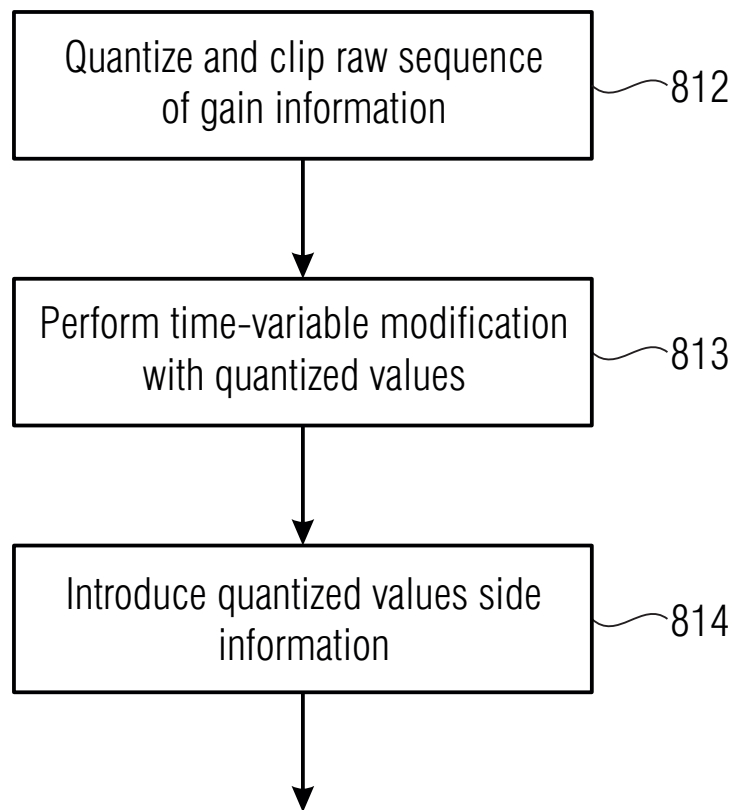


Fig. 8c

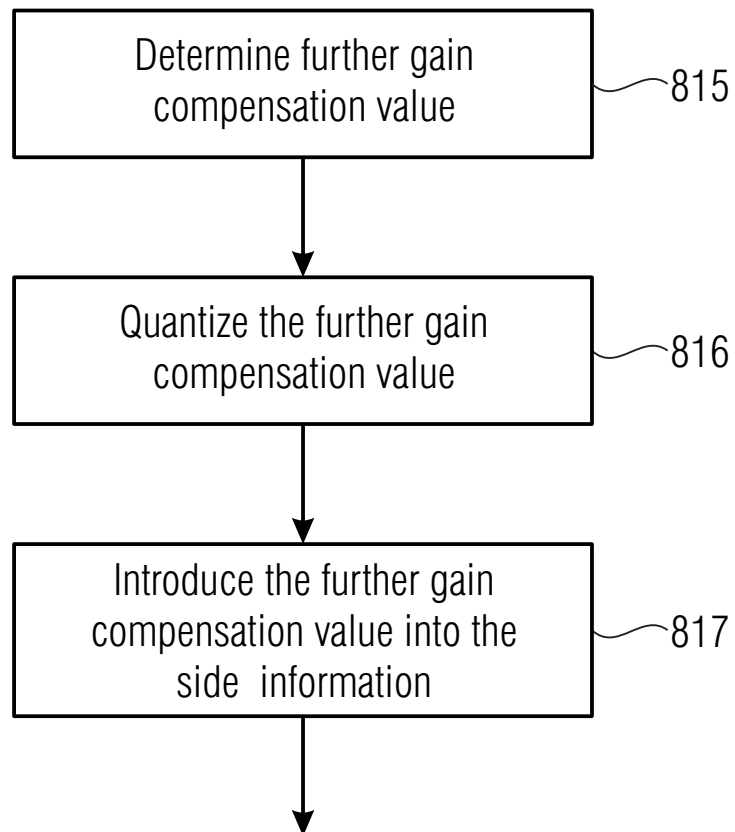


Fig. 8d

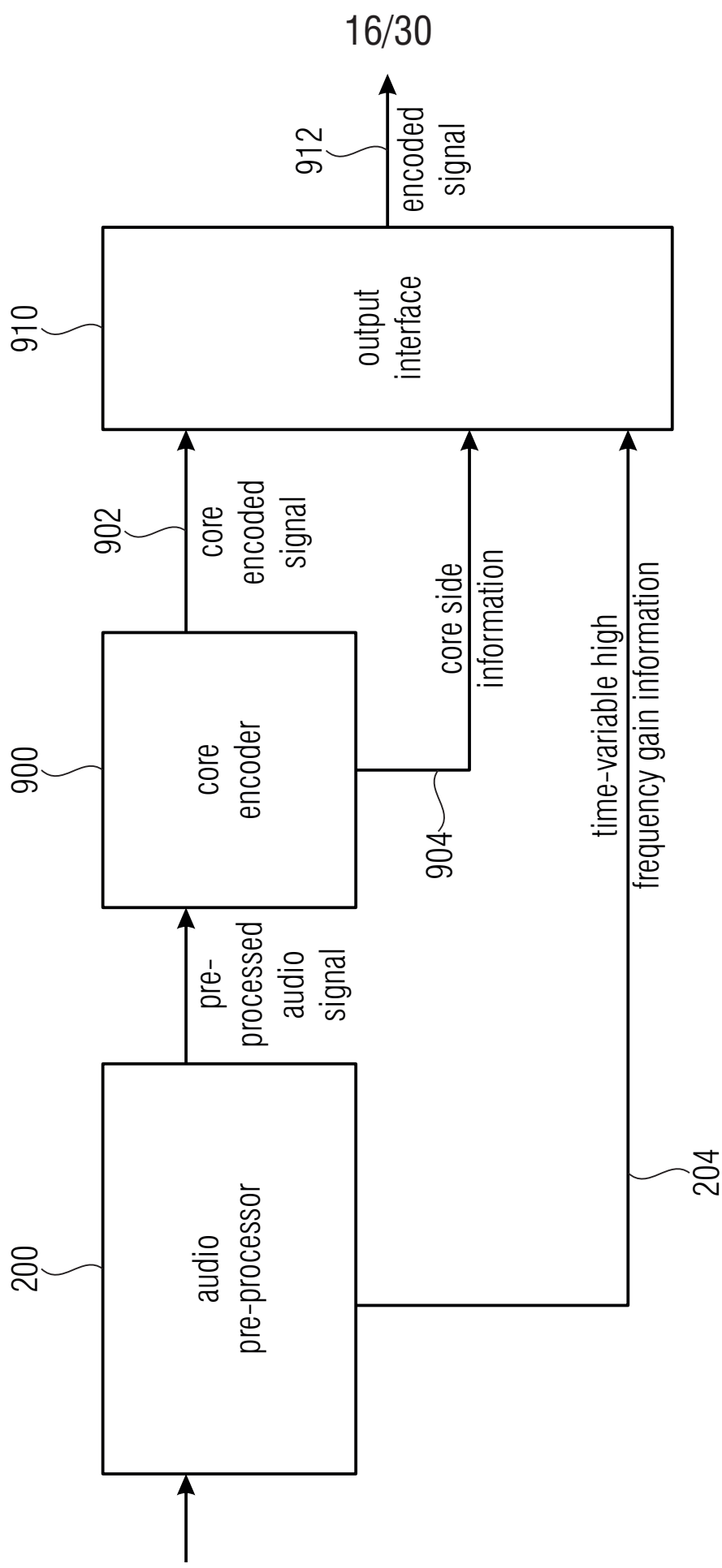


Fig. 9a

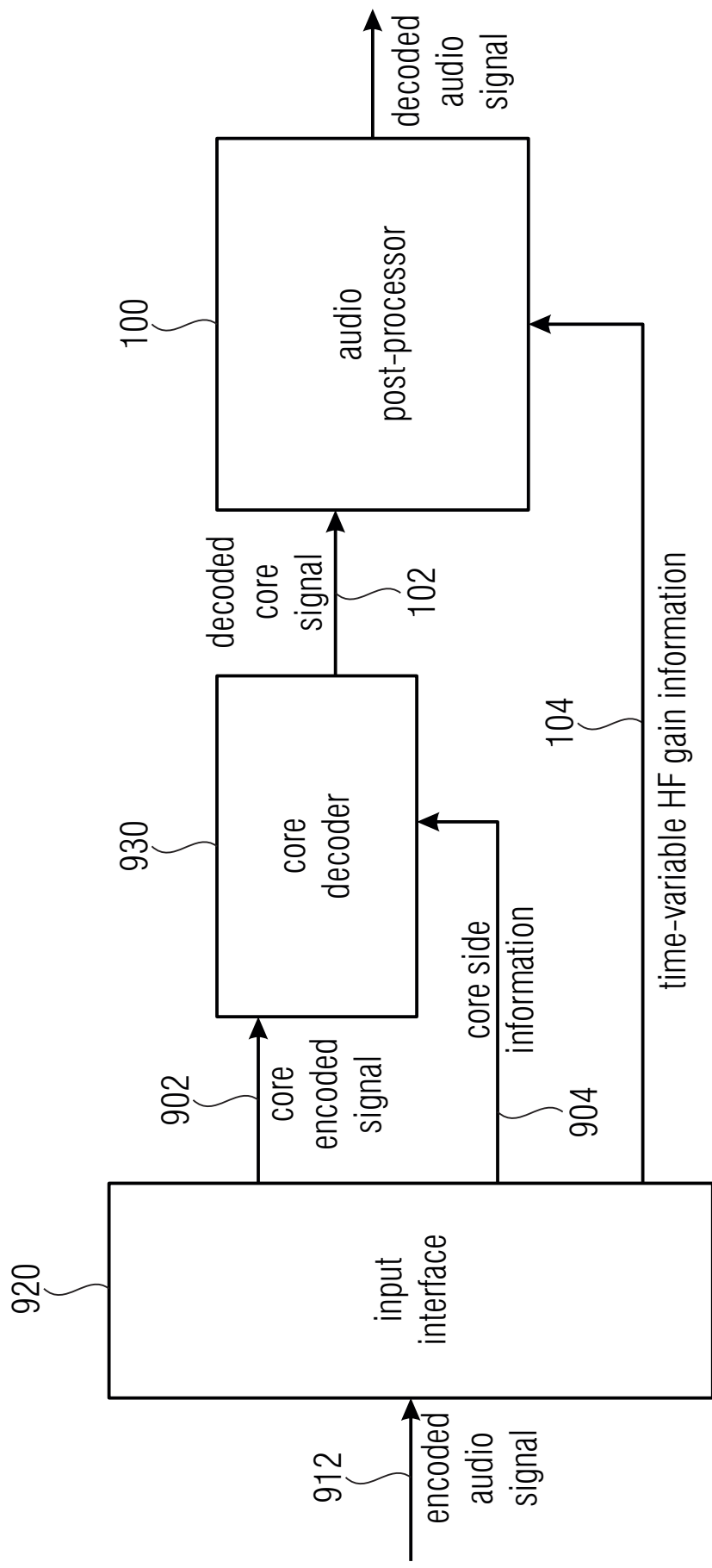


Fig. 9b

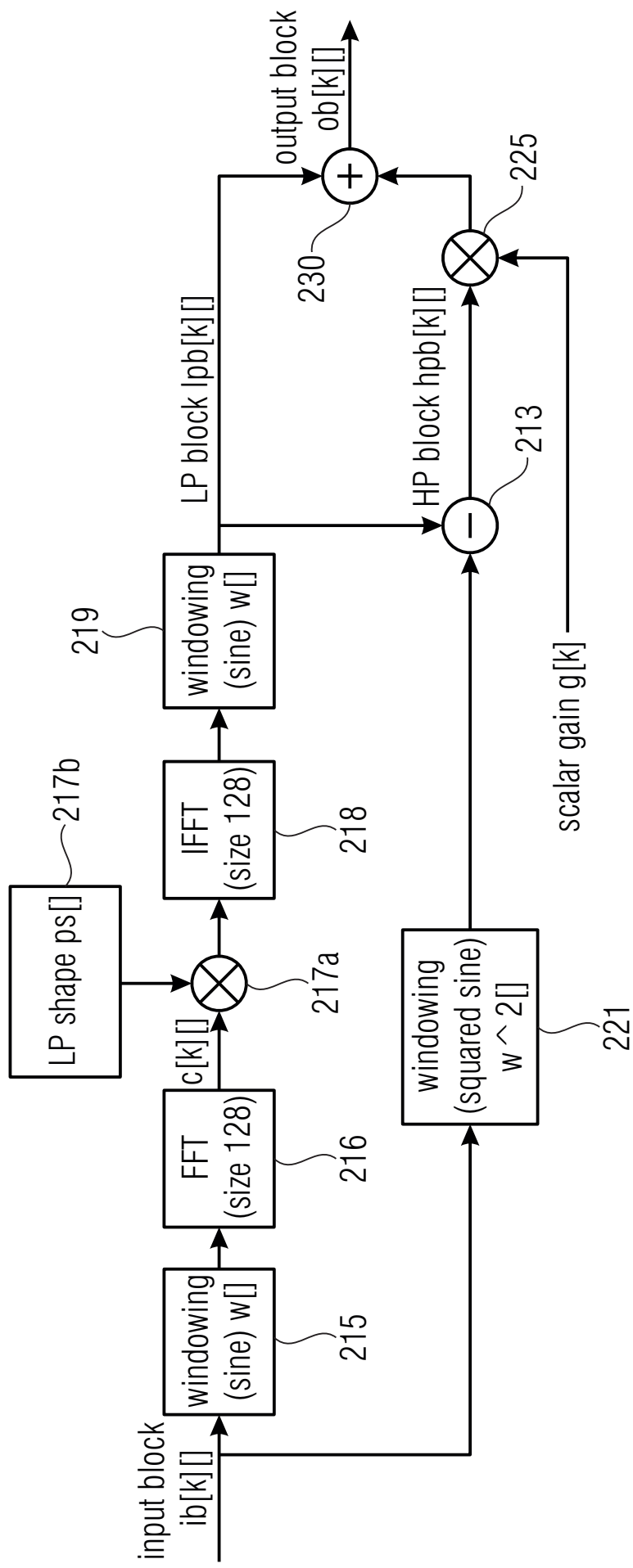


Fig. 9c

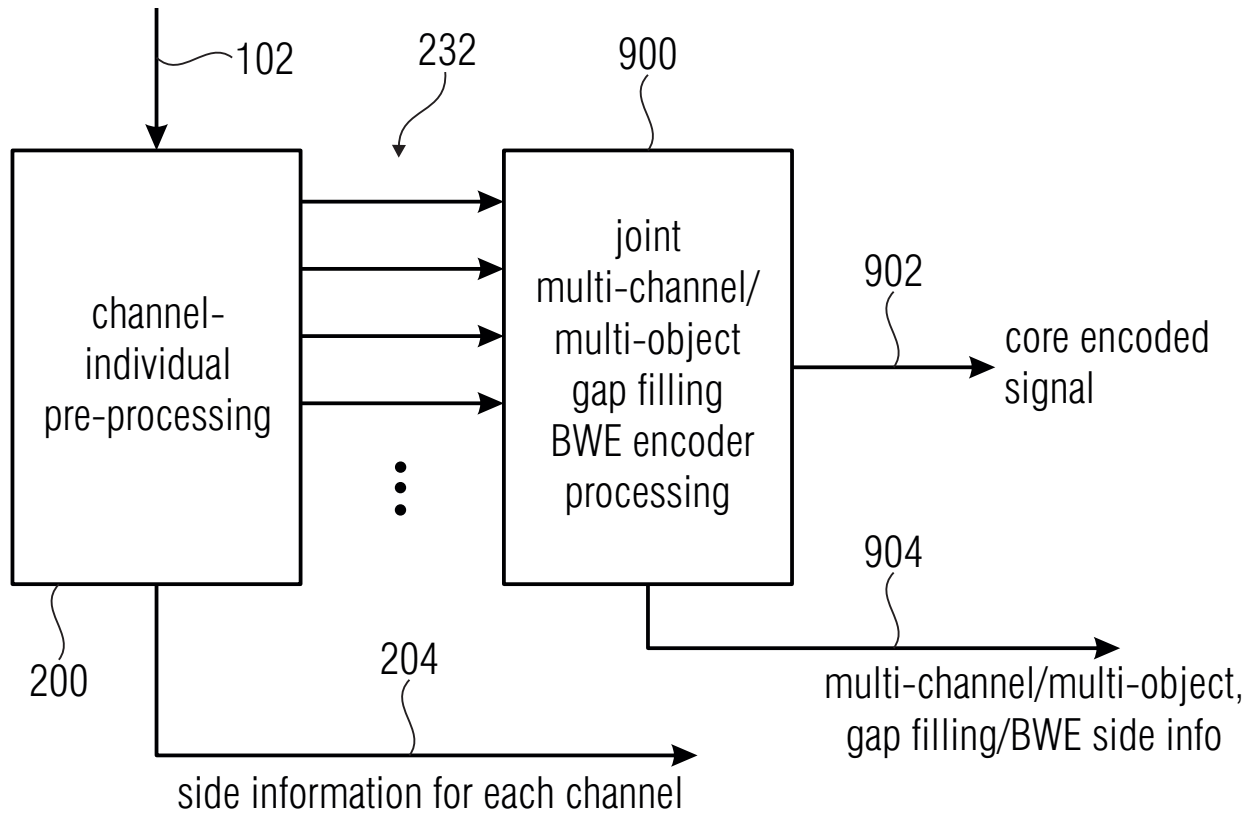


Fig. 10a

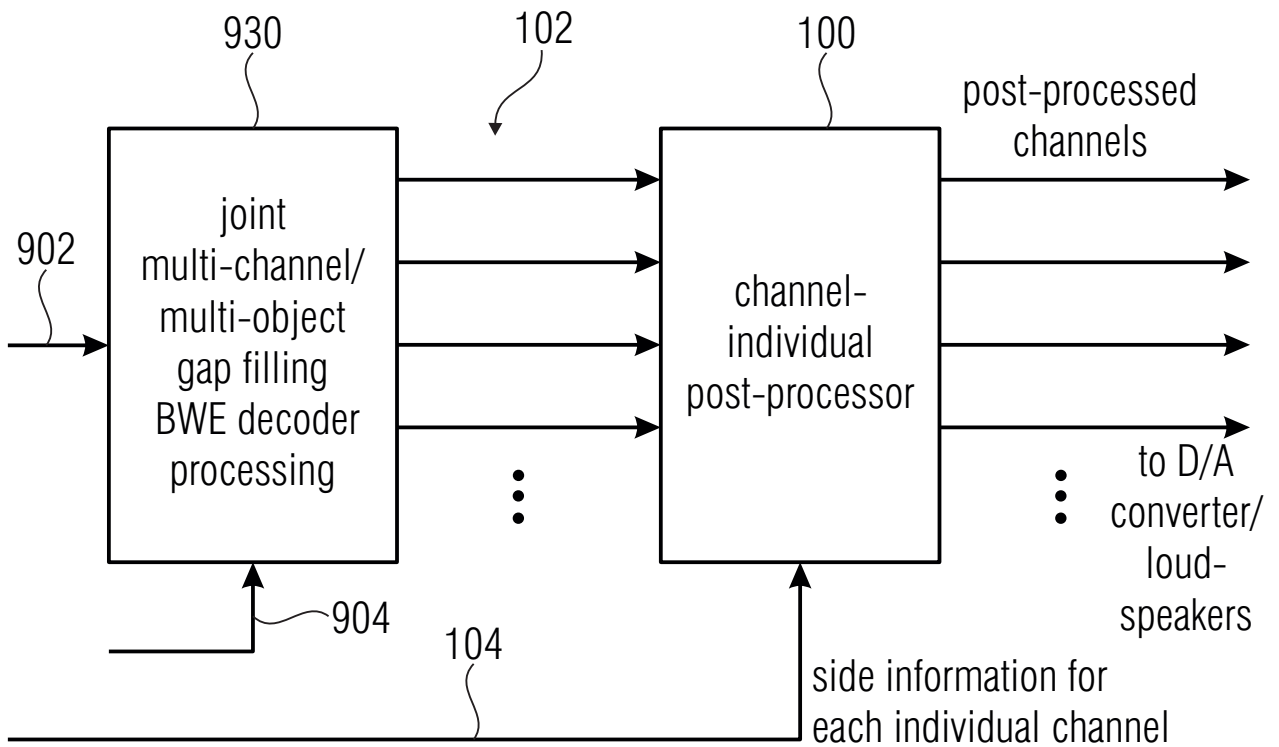


Fig. 10b

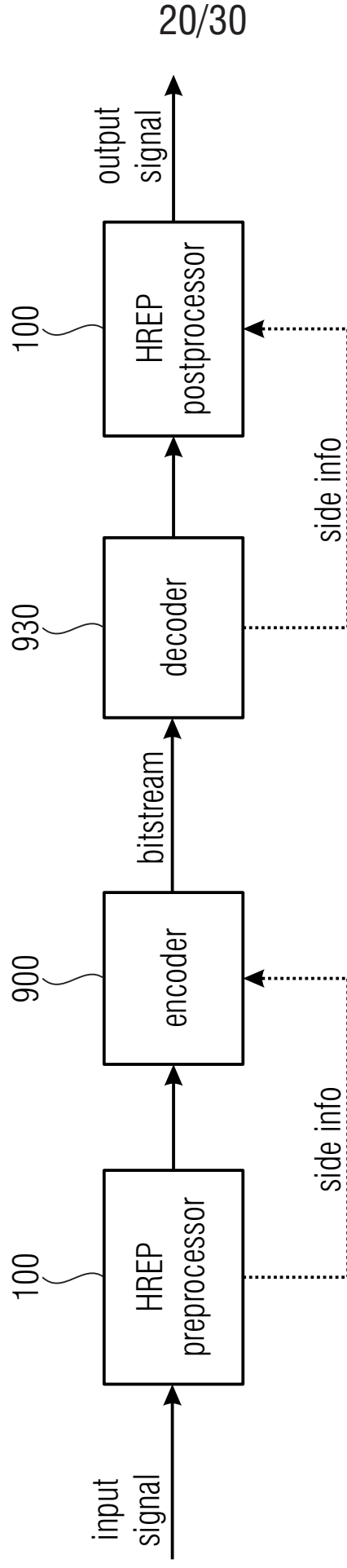


Fig. 10c

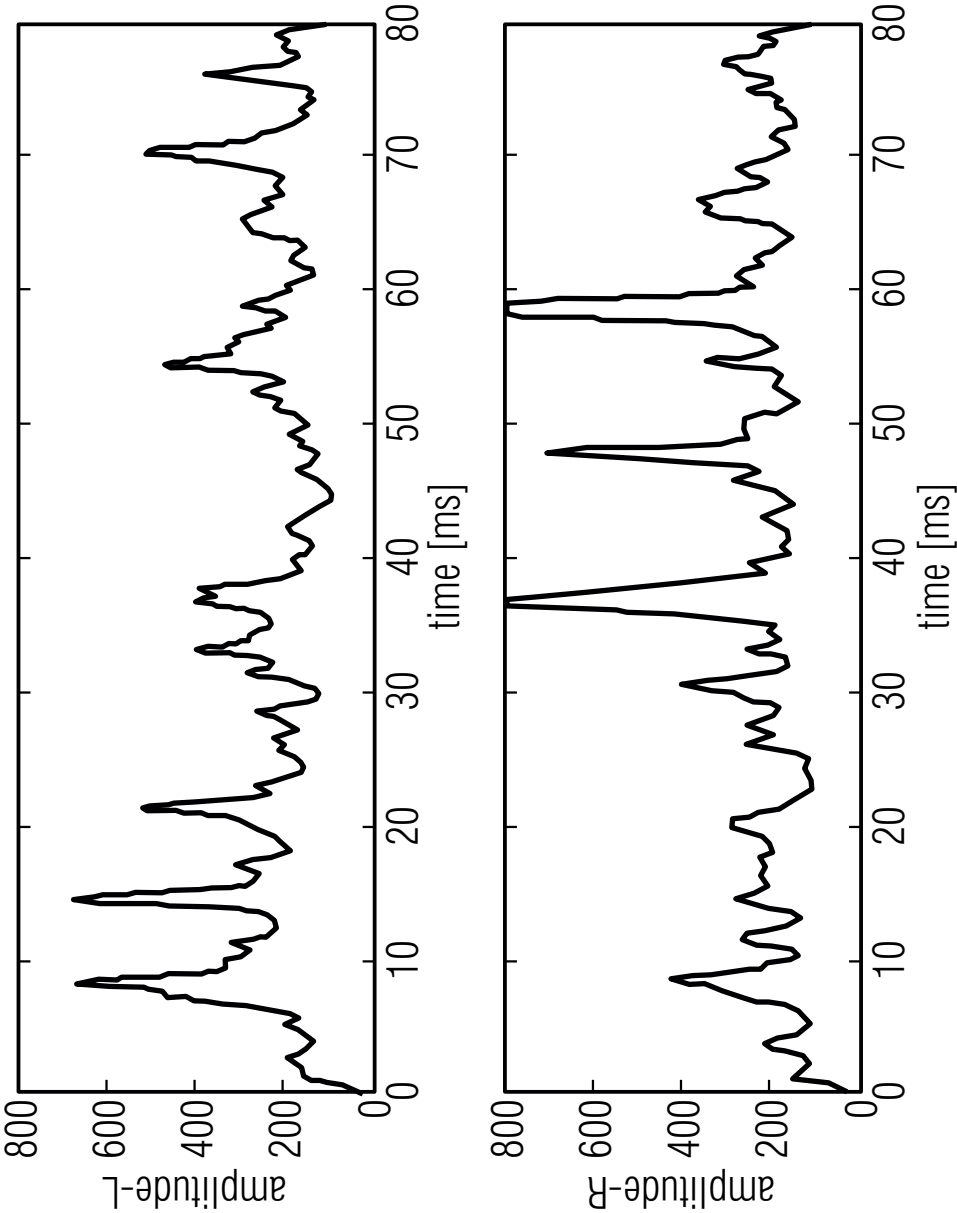


Fig. 11

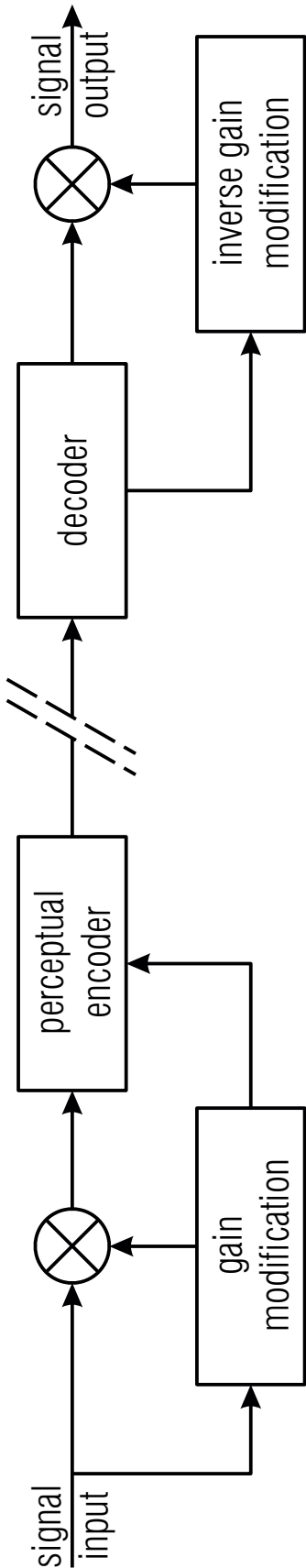


Fig. 12

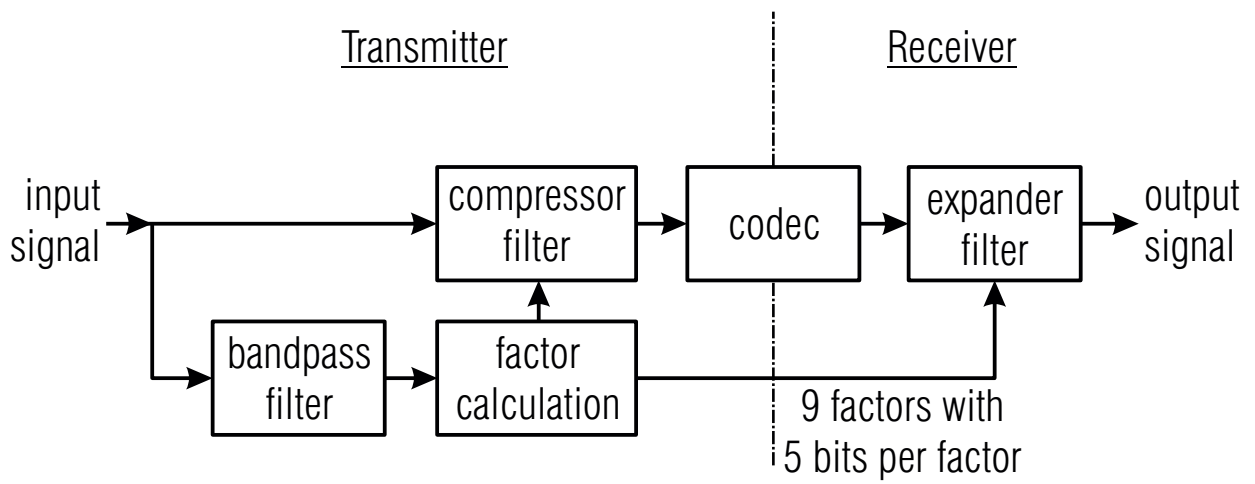


Fig. 13a

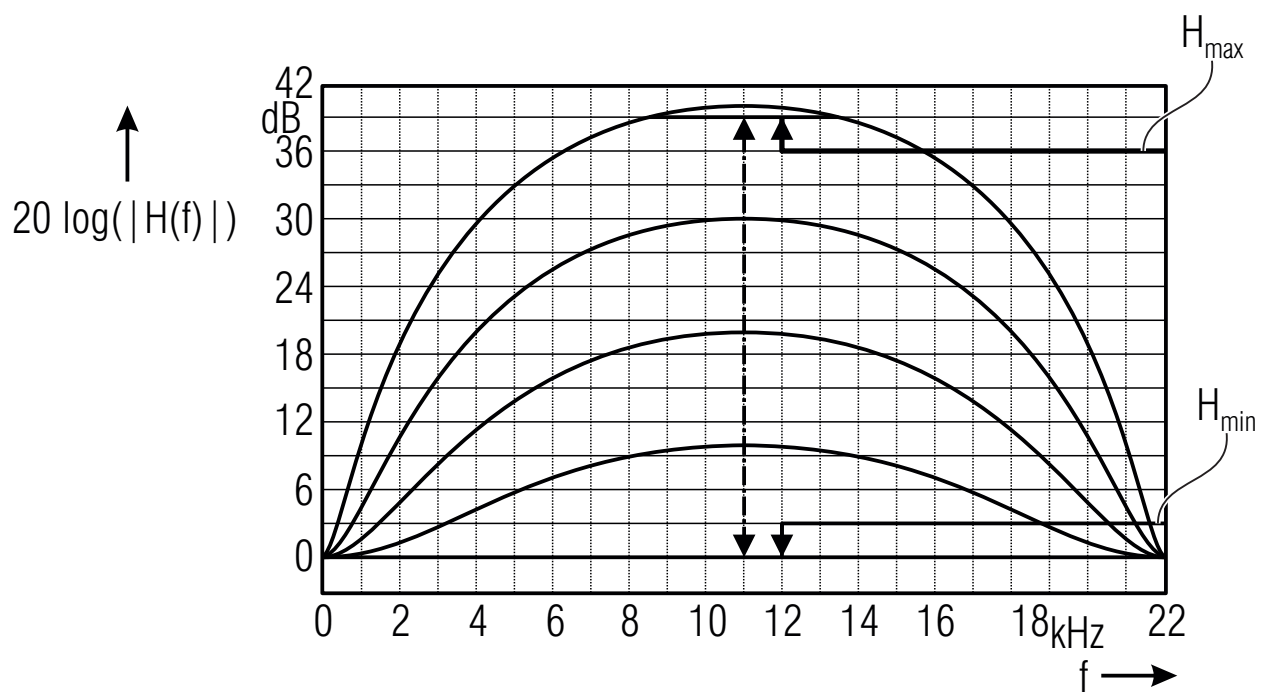


Fig. 13b

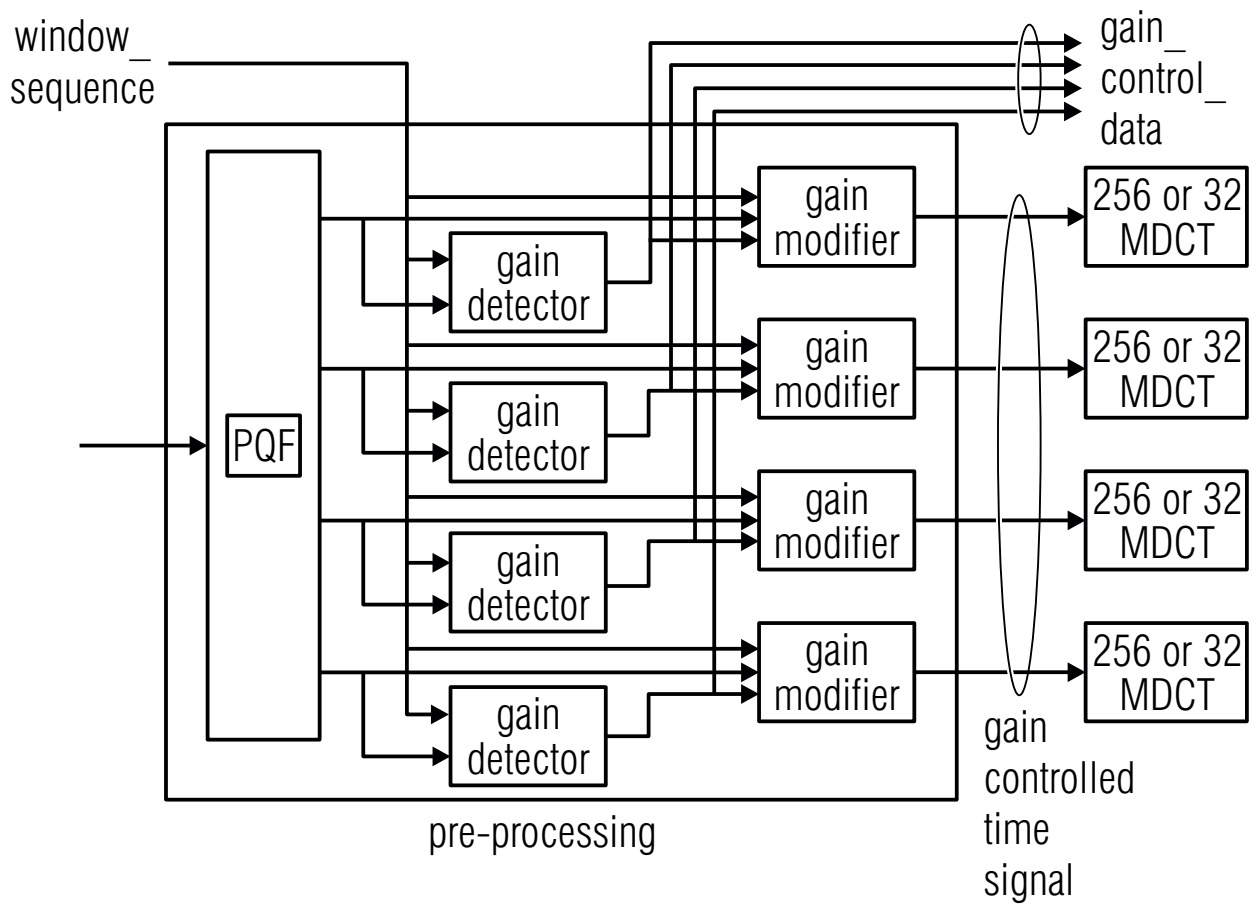


Fig. 14

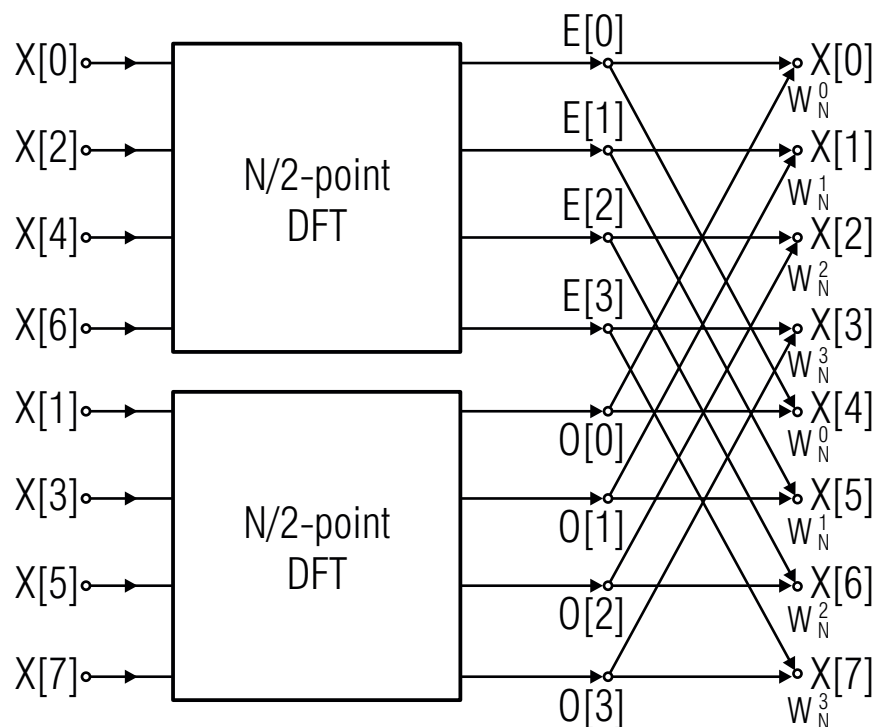


Fig. 15

Methodology	MUSHRA ITU-R BS.1534
Lab	Fraunhofer IIS
Test Conditions	"nohrep": Reference Quality Encoder (FD core using IGF) "hrep": Proposed Configuration
Bit Rate / Channels	128 kbps 5.1 channel
Number of listeners after post-screening	14
Test Signals	• 1: ARL_applause • 2: Appl3D • 3: StudioAppl • 4: applse • 5: chostakovitch • 6: fountain_music • 7: glock • 8: indie2 • 9: jackson1 • 10: pops • 11: poulenc • 12: rock_concert • 13: stomp

Fig. 16a

Fig. 16	Fig. 16a
	Fig. 16b

Bit Rate / Channels	48 kbps stereo
Number of listeners after post-screening	15
Test Signals	• 1: ARLapplause • 2: Exercise • 3: HeavyApplause • 4: Intro • 5: Klatschen • 6: MediumApplause • 7: SallyBrown • 8: applse
Bit Rate / Channels	128 kbps stereo
Number of listeners after post-screening	14
Test Signals	• 1: ARLapplause • 2: Exercise • 3: HeavyApplause • 4: Intro • 5: Klatschen • 6: MediumApplause • 7: SallyBrown • 8: applse

Fig. 16b

Fig. 16

Fig. 16a
Fig. 16b

27/30

Average and 95% Confidence Intervals

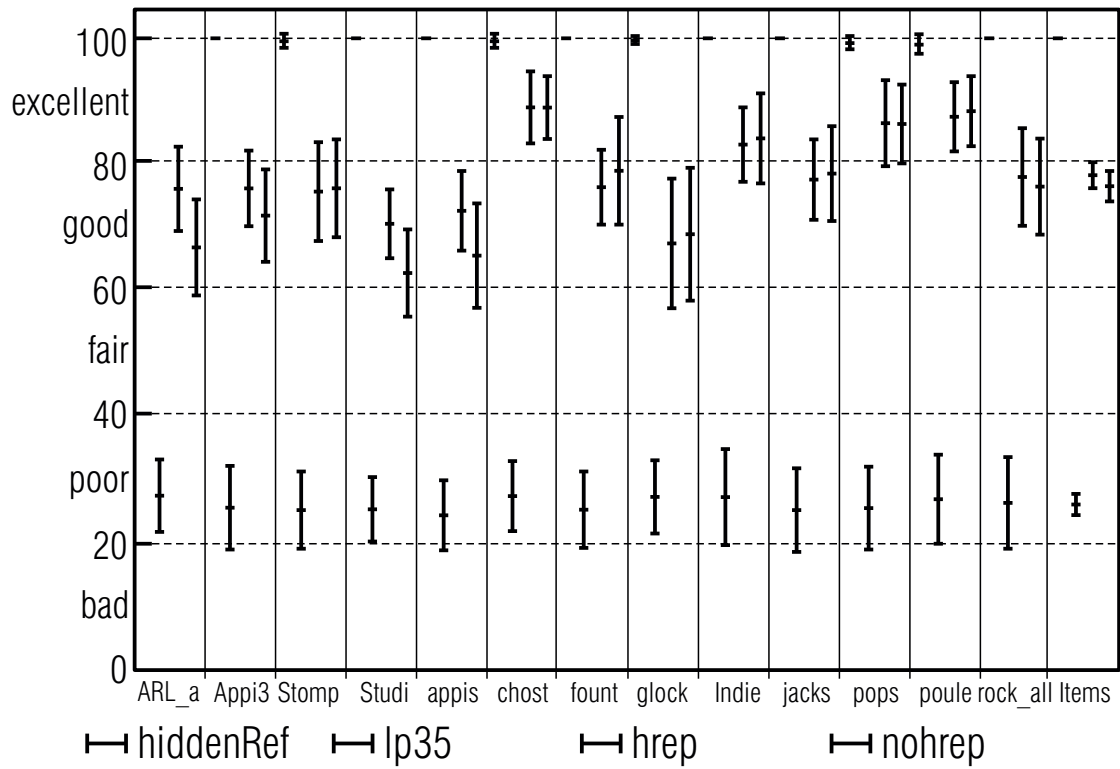


Fig. 17a

Average and 95% Confidence Intervals (Differences: hrep-nohrep)

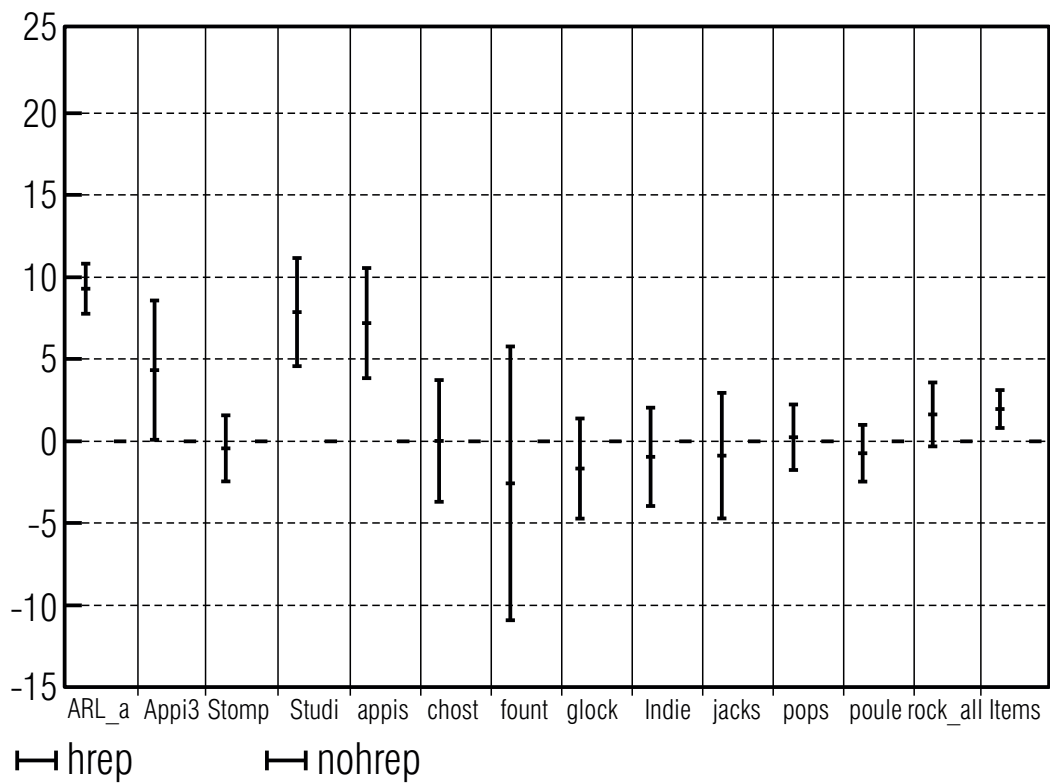


Fig. 17b

28/30

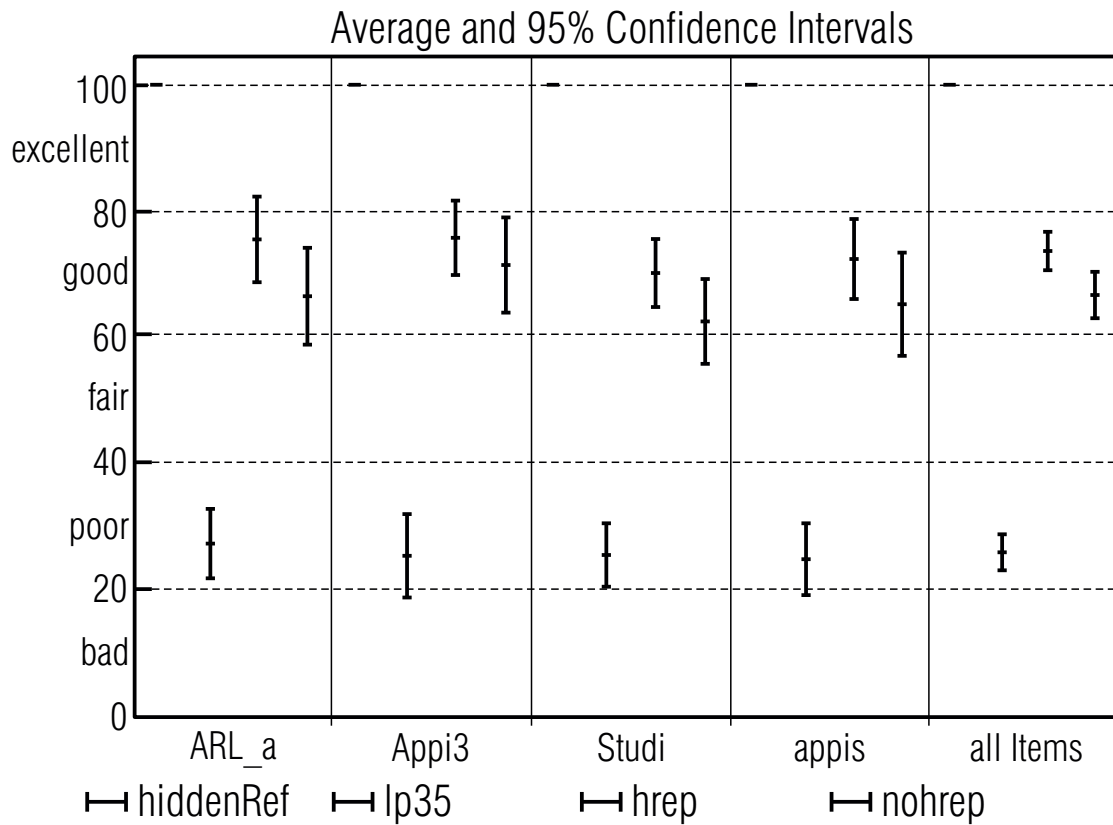


Fig. 17c

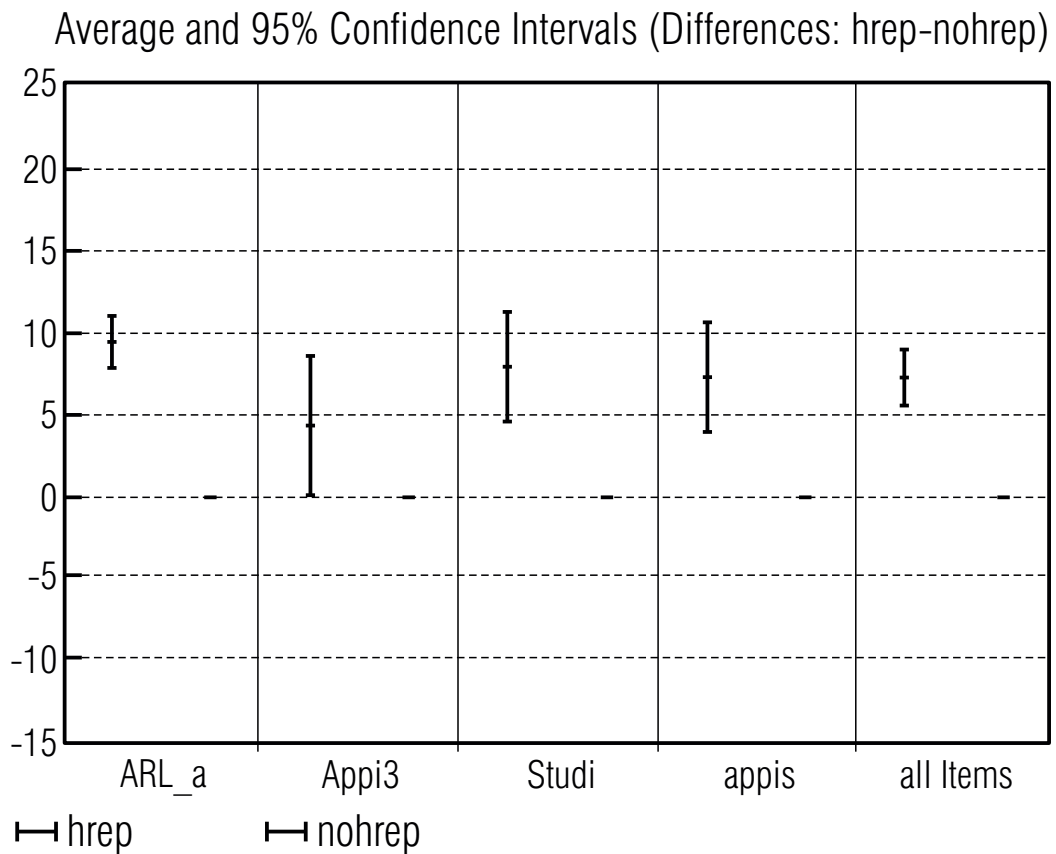


Fig. 17d

29/30

Average and 95% Confidence Intervals

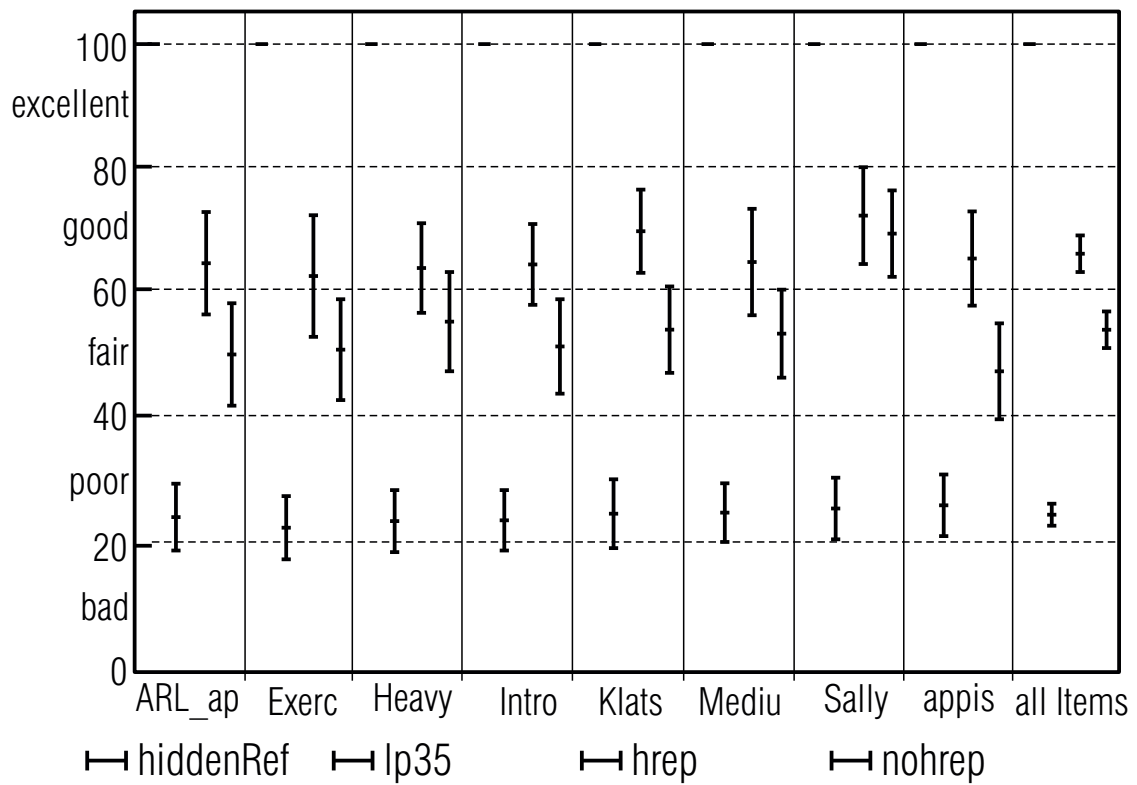


Fig. 17e

Average and 95% Confidence Intervals (Differences: hrep-nohrep)

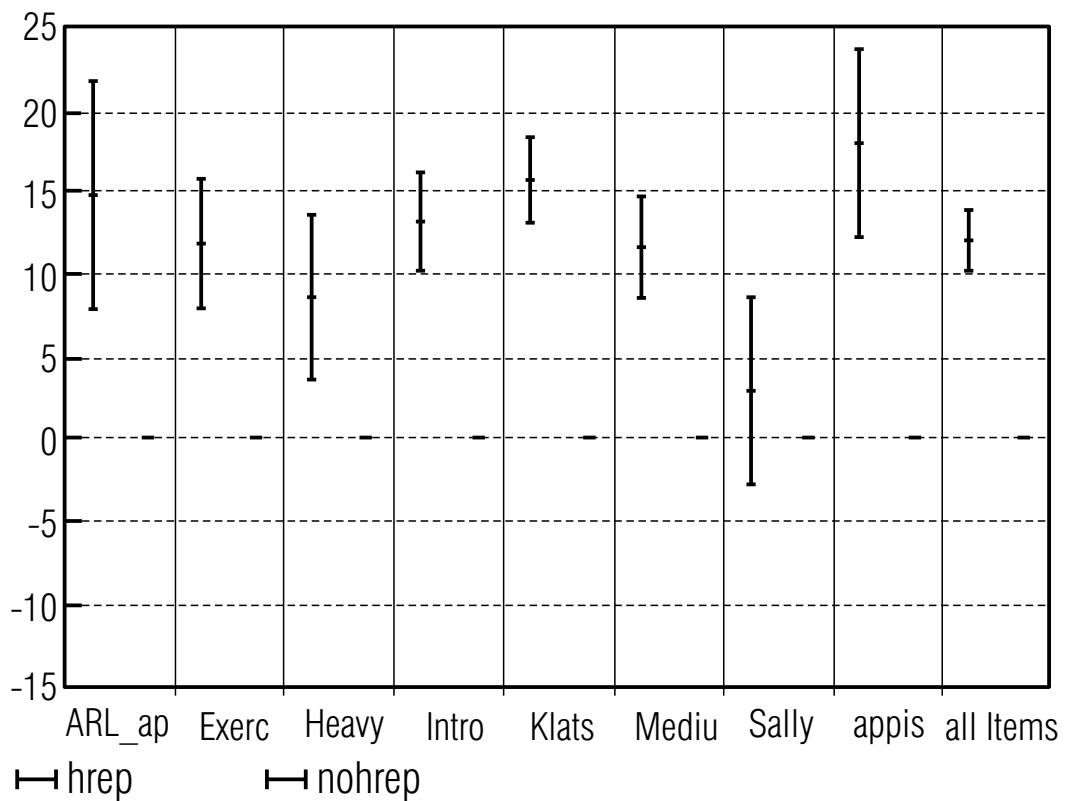


Fig. 17f

30/30

Average and 95% Confidence Intervals

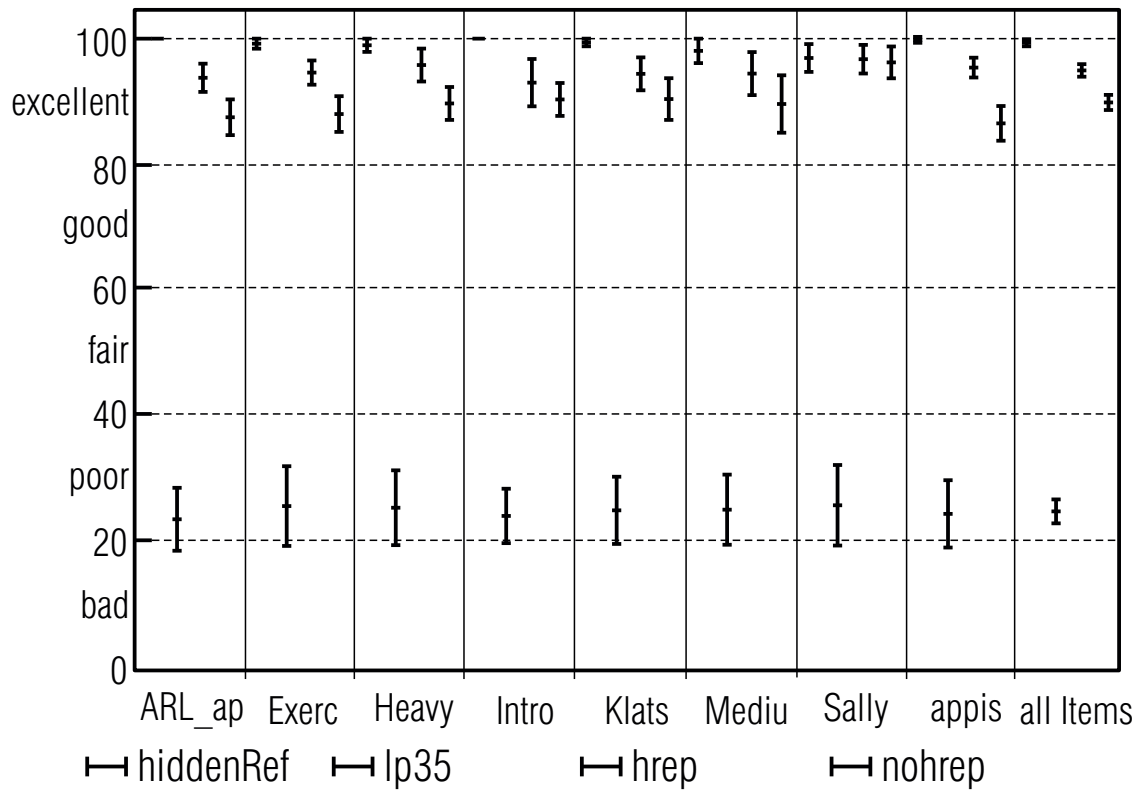


Fig. 17g

Average and 95% Confidence Intervals (Differences: hrep-nohrep)

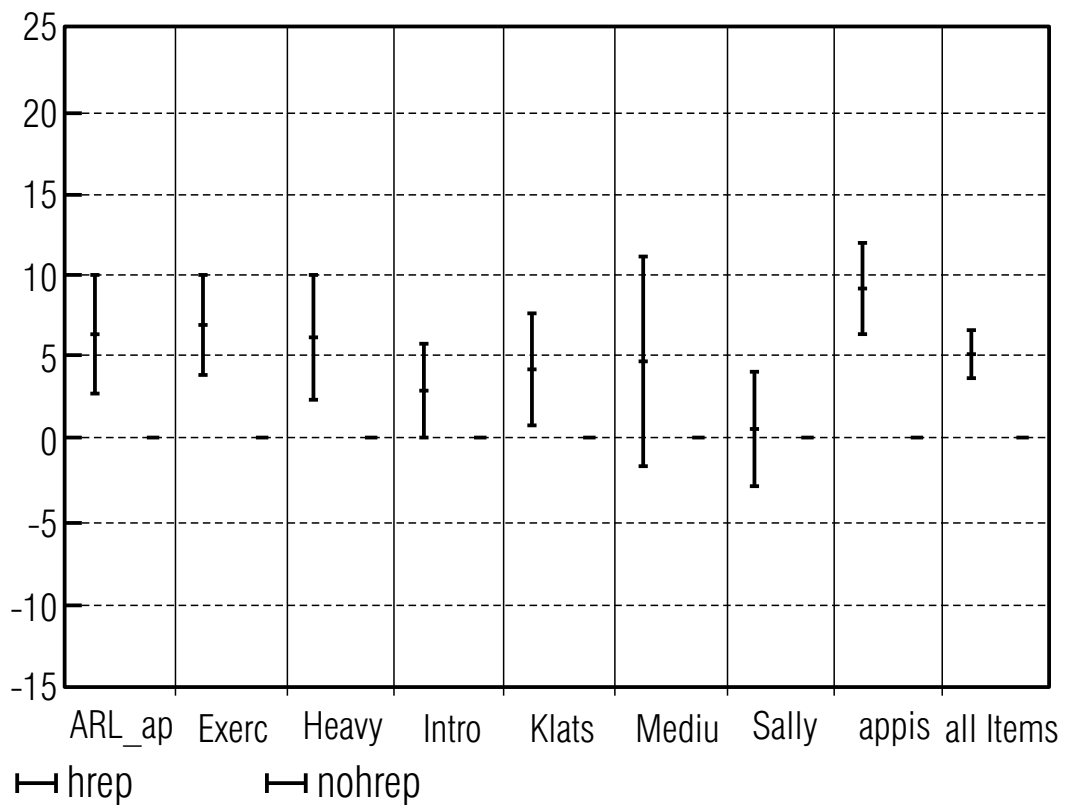


Fig. 17h