Institute for Communications Technology
Institut für Nachrichtentechnik
Technische Universität Braunschweig
Schleinitzstrasse 22, 38106 Braunschweig

Bachelorarbeit

Survey of Error Concealment Schemes for Real-Time Audio Transmission Systems

by

Aránzazu Robles Moya

Juni 2012

Supervisors:
Dipl.-Ing. F. Pflug
Prof. Dr.-Ing. T. Fingscheidt

Abstract

This thesis presents an overview of the main strategies employed for error detection and error concealment in different real-time transmission systems for digital audio. The “Adaptive Differential Pulse-Code Modulation (ADPCM)”, the “Audio Processing Technology Apt-x100”, the “Extended Adaptive Multi-Rate Wideband (AMR-WB+)”, the “Advanced Audio Coding (AAC)”, the “MPEG-1 Audio Layer II (MP2)”, the “MPEG-1 Audio Layer III (MP3)” and finally the “Adaptive Transform Coder 3 (AC3)” are considered. As an example of error management, a simulation of the AMR-WB+ codec is included. The simulation allows an evaluation of the mechanisms included in the codec definition and enables also an evaluation of the different bit error sensitivities of the encoded audio payload.

Contents

1 Introduction

In real-time transmission systems for digital audio the effects of noisy channels should be avoided or at least minimized. In order to improve the quality of speech communication and the compression mechanisms, several techniques can be applied. These techniques include interpolation, reconstruction, waveform repetition, muting, etc. However, following the specific characteristics of each mechanism, new approaches can be defined in order to allow a better and more precise management of errors.

2 Audio Coding Methods / Algorithms / Standards

The following chapter discusses various audio coding techniques in order to evaluate the mechanisms they used to deal with errors. The content of this survey is formed by:

- Adaptive Differential Pulse-Code Modulation (ADPCM)
- Audio Processing Technology Apt-x100
- Extended Adaptive Multi-Rate Wideband (AMR-WB+)
- Advanced Audio Coding (AAC)
- MPEG-1 Audio Layer II (MP2)
- MPEG-1 Audio Layer III (MP3)
- Adaptive Transform Coder 3 (AC3)

For each technology its main characteristics, its functioning are described and subsequently, the error concealment approaches found for each one are depicted.

2.1 ADPCM

ADPCM (Adaptive Differential Pulse-Code Modulation) [IT90] was developed at Bell Labs and standardized by ITU-T. It is a compression algorithm that provides a conversion of a 64 kbit/s pulse-code modulation (PCM) channel to and from a 40, 32, 24 or 16 kbit/s channel. It performs a lossy data compression, where irrelevant information is eliminated. ADPCM determines and removes redundancy from an audio signal by eliminating the predictable parts of the signal. The algorithm does not encode all the waveform, it decodes just the difference between the current sample and the prediction obtained from a number of past samples. An example of ADPCM application is the Digital Circuit Multiplication Equipment (DCME). DCME is a type of voice compression equipment that is used in long-distance links, like communications satellite or submarine communications cable.

2.1.1 Description

In ADPCM a difference signal (between the input signal and an estimated version) is calculated, quantized and sent to the decoder. ADPCM is composed by an adaptive predictor and an adaptive quantizer. The predictor creates the estimated value by using a certain number of preceding samples, so it performs "backward prediction". This value is subtracted from the signal value to form an error signal that is quantized. In order to use this value for the prediction of future values, the quantized value is inverse quantized and sent to the predictor. Since the encoder and the decoder work with the same samples, there is no need to send additional information to the decoder.

The ADPCM coding process makes use of the following elements:

- **Input PCM format conversion.**
  This block converts the input signal to a uniform PCM signal.

- **Difference signal computation.**
  The subtraction between the estimated signal $s_e(k)$ and the PCM signal $s(k)$ is calculated. The result is the difference signal $d(k)$.

  $$d(k) = s(k) - s_e(k) \tag{2.1}$$

  The ADPCM algorithm uses the redundancy contained in the input signal to perform the compression. If the input signal is very predictable, the previous value and the current value differ just in a small value. If the audio codec compresses just this difference, the number of necessary bits is lower.

- **Adaptive quantizer.**
  The difference value is quantized in this block. The input is the $d(k)$ signal and the output is a binary word, that represents the quantized value $I(k)$ of the difference signal. First, a base 2 logarithm is calculated for the $d(k)$ signal and then, this value is scaled by using $y(k)$ as a scale factor. The result of this operation is compared with the quantization table to obtain its corresponding value. This value is represented with different resolution depending on the ADPCM mode. The following equation is the mathematical representation of the quantizer block.

  $$I(k) = log_2|d(k)| - y(k) \tag{2.2}$$

The quantized values for 40kbit/s are represented within 5 bits: the first bit corresponds to the sign of $d(q)$ and the next four bits represent its magnitude. With 4 bits 16 possibles values can be quantized. For 32 kbit/s, 24kbit/s and 16 kbit/s, 4, 3 and 2 bits are used, respectively. The quantization table for 24 kbit/s operation is shown in table 2.1.

Normalized Quantizer Input Range $log_2\|d(k)\| - y(k)$	$\|I(k)\|$	Normalized Quantizer Output $log_2\|d_q(k)\| - y(k)$
$[2.58, +\infty)$	3	2.91
[1.70, 2.58)	2	2.13
[0.06, 1.70)	1	1.05
$[-\infty, -0.06)$	0	$-\infty$

Table 2.1. 24 kbit/s Quantizer Characteristics

- **Inverse adaptive quantizer.**
  This block performs the inverse operation (2.3) as the previous block. Its inputs are the quantized value $I(k)$ and the scale factor $y(k)$. The result is the quantized version of the difference signal $d_q(k)$.
  $$d_q(k) = 2^{[|I(k)|+y(k)]} \tag{2.3}$$

- **Quantizer scale factor adaptation.**
  This element performs the calculation of the scale factor employed in the quantizer and inverse quantizer. The inputs are the output of the quantizer $I(k)$ and the adaptation speed control $v(k)$. As a result the adaptation step size $y(k)$ is produced. The step size or scale factor is formed by two terms: the fast scale factor $y_u(k)$ and the slow scale factor $y_l(k)$.
  $$y(k) = v(k)y_l(k-1) + (1 - v(k))y_l(k-1) \tag{2.4}$$
  where
  $$y_u(k) = (1 - 2^{-5})y(k) + 2^{-5}F[I(k)] \tag{2.5}$$
  $$y_l(k) = (1 - 2^{-6})y_l(k-1) + 2^{-6}y_u(k) \tag{2.6}$$
  and $F[I(q)]$ is a discrete function defined in the standard documentation.

- **Adaptive predictor.**
  The predictor calculates the estimated signal by using the quantized difference value $d_q(k)$ and the reconstructed signal $s_r(q)$ as inputs. The predictor includes a sixth order predictor and a second order predictor. The equation for the reconstructed signal is:
  $$s_r(k-i) = s_e(k-i) + d_q(k-i) \tag{2.7}$$
  where
  $$s_e(k) = \sum_{i=1}^{2} a_i(k-1)s_r(k-i) + \sum_{j=1}^{6} b_i(k-1)d_q(k-j) \tag{2.8}$$
  $a_i(k)$ and $b_i(k)$ are the predictor coefficients.

ADPCM Encoder

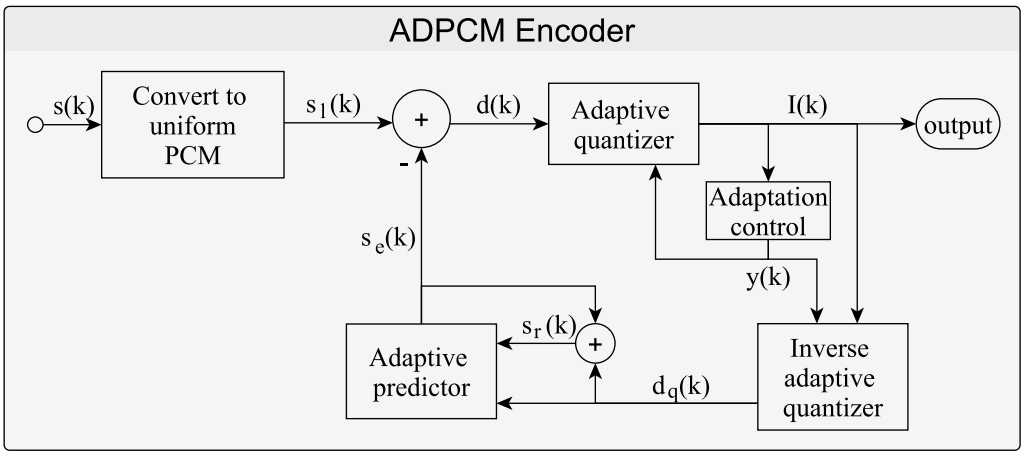

Fig. 2.1. ADPCM Encoder

The ADPCM encoder processes an input signal, first by converting it to an uniform PCM signal. Then, an estimated signal is obtained from the previous quantized values of the signal, this signal is utilized to calculate the difference signal. By using a subtract operation between the input signal and the estimated signal, the difference signal is obtained. This value is quantized using the quantized tables and represented in a fixed number of bits. This encoded word is sent to the decoder and is also used in the calculation of the predicted signal value and in the quantizer (and inverse quantizer) adaptation. In order to use this information in the prediction block, it should pass through an inverse quantizer to recover the original value. With it the new difference signal is calculated.

ADPCM Decoder

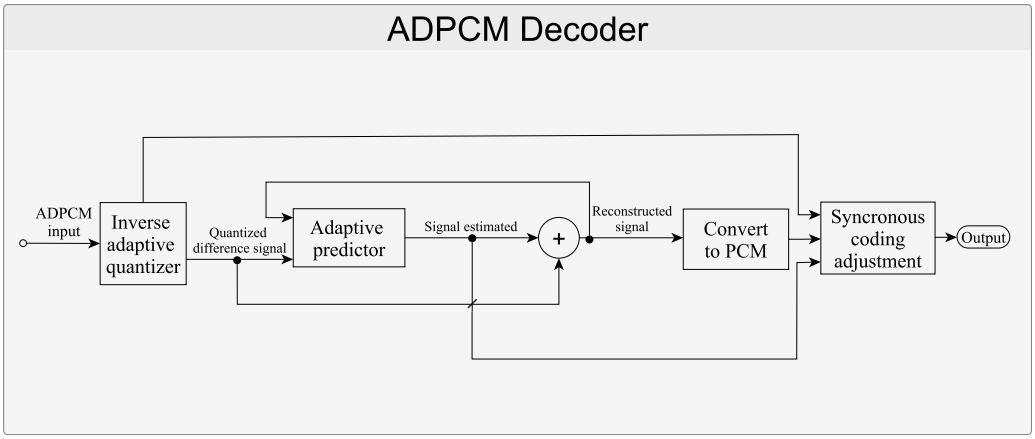

Fig. 2.2. ADPCM Decoder

In the decoder, the encoded signal is processed within the inverse quantizer block and the quantized difference signal is obtained. A predicted value is calculated and this value is added to the quantized difference signal. The reconstructed signal results from this operation. After a PCM conversion and a coding adjustment operation the decoded signal is finished.

2.1.2 Error Concealment Approaches

The main techniques of error management in ADPCM are examined in the following section.

Conventional Muting

The conventional ADPCM muting scheme uses a cyclic redundancy check code (CRC) for error detection in the received burst. When a CRC error is detected, the difference between the received signal and the preceding one is calculated. The scheme replaces the received ADPCM signals that have a difference larger than a defined threshold, with a zero difference signal. For example, for 32 kbps ADPCM signals when an error is detected, just the codes 0111 or 1000 (which correspond to a difference value of 7 and -7) are replaced. For the replacement the code that corresponds to a difference value of 0 (1111) is used.

Super Mute Scheme

Super-Mute Scheme [SK95] improves the ADPCM voice quality in slow fading channels by using the probability distribution of the ADPCM codewords without error bits and the error patterns. In this procedure when an error is detected, the received codeword is replaced with the codeword with higher likelihood. This code is calculated in the decoder making use of the following procedure.

The probability distribution $P(d_q(k))$ of an ADPCM code without error is presented in Fig. 2.3.

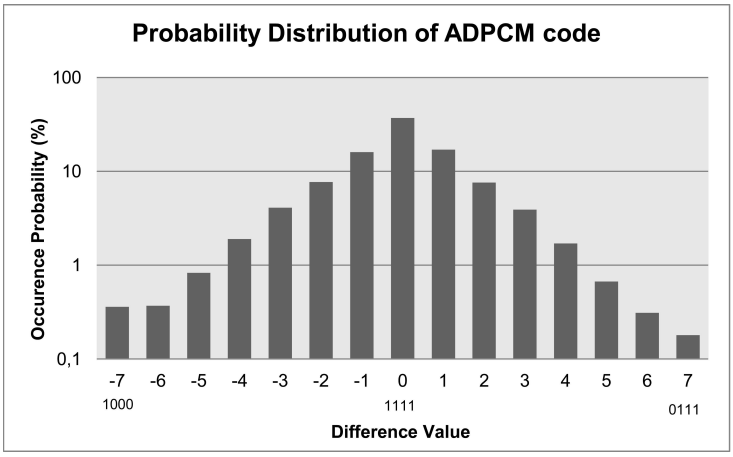

Fig. 2.3. ADPCM Probability Distribution

The output signal of the quantizer, encoded in 4-bit form, and d_k is the difference signal and is shown in table 2.2.

All the received codes are optimized to minimize the difference with the value of the transmitted codes. The expected difference value of the transmitted ADPCM code $E(R_1R_2R_3R_4)$ is calculated according to equation (2.9).

$$E(R_1R_2R_3R_4) = \sum_{d_q(k)=-7}^{7} d_q(k) \cdot P(d_q(k)) \cdot P_e(d_q(k)) / \sum_{d_q(k)=-7}^{7} P_e(d_q(k)) \qquad (2.9)$$

I(K)	Difference Value
1111	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1110	-1
1101	-2
1100	-3
1011	-4
1010	-5
1001	-6
1000	-7

Table 2.2. Table of the difference signal and the output of the quantizer

where $d_q(k)$ is the difference value, $P(d_q(k))$ is the occurrence probability of $d_q(k)$ (Fig. 2.3). $P_e(d_q(k))$ is the occurrence probability of the current error pattern, this value is obtained empirically.

For every received code the best replacement candidate is calculated, in order to use it if errors in the communication are produced. A table with the 15 replacement candidates of the original code is calculated and then, the expected difference signal following the equation (2.9). As the difference value of $d'_q(k)$ is an integer without decimals and the result of $E(R_1R_2R_3R_4)$ has decimals, there are two replacement candidates for each code. The optimum ADPCM code is the code that has higher probability before it is disturbed by an error pattern.

Conventional and Super Muting Schemes are presented in figure Fig. 2.4.

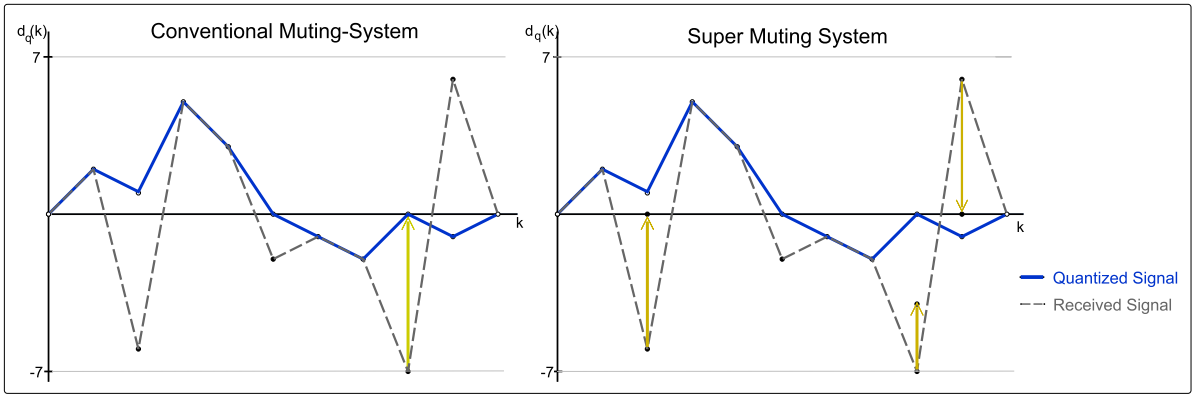

Fig. 2.4. ADPCM Muting Schemes

ADPCM Code Detection (ACD) Scheme

In the ACD scheme [SK98], when a frame error is detected, the channel quality degradation is estimated by counting the ADPCM codes with large absolute difference value in the error-detected frame. These absolute difference values should be greater than a certain threshold, the optimum are 5 or 6, according to the method included in [SK98]. For the absolute difference value of more than 6, there are 4 ADPCM codes: 0111, 0110, 1001 y 1000.

The proposed method calculates the number of times that one of the previous codes appears. With that number a channel quality estimation can be make. The quality of the channel is divided into 4 groups depending on the sum of ADPCM codes with large absolute difference value (see table 2.3).

Channel quality degradation	Sum of ADPCM codes	Estimated quality
1	0	Good
2	1	
3	$2\tilde{3}$	
4	more than 4	Bad

Table 2.3. Channel Quality Degradation

The probability distribution of the 4 groups are known and according with them, when the frame degradation is small, the ADPCM code of low occurrence probability and whose difference absolute value is large, is probably correct; however, when the channel quality degradation is large, there is a high probability that the ADPCM code has errors. Super-Mute and ACD can be used together to reduce the degradation of signal quality due to fatal ADPCM code errors.

Packet Loss Concealment

This method [SN02] proposes a packet loss concealment method. It uses pitch waveform repetition to update the internal states in the decoder. This is made in order to regenerate the signal and reduce the speech quality degradation caused by the packet loss.

The process consists of these steps:

- *Waveform repetition.* If the decoder detects a missing frame, applies waveform repetition by employing previous decoded values. However, if just the signal is generated but the internal states are not updated, click noise appears in the decoded signal.
- *Internal state update by encoding the decoded speech.* In order to maintain the internal states, the above generated waveform is encoded. Is important to keep the consistence of these states, because they are the basis of the backwards adaptive predictor and quantizer.
- *Forgetting factor control.* The forgetting factors take into account the values of the previously processed values. If the current value is missing and this factors are not renewed, the predictor does not work correctly.

The internal states and the forgetting factors appear in the following equations.

$$y_u(k) = (1-\mu)y(k-1) + \mu F[I(k)] \tag{2.10}$$

where $y_u(k)$ is the scale factor for the quantizer, μ is a forgetting factor and the inverse quantizer and $F[I(k)]$ is a logarithmic scaling factor.

$$s_r(k) = s_e(k) + d(k) \tag{2.11}$$

where $s_r(k)$ is the reconstructed signal, $s_e(k)$ is the estimated signal and $d(k)$ is the difference signal.

$$s_e(k) = \sum_{i=1}^{2} a_i(k-1)s_r(k-i) + \sum_{j=1}^{6} b_i(k-1)d_q(k-j) \tag{2.12}$$

The equation (2.12) corresponds to the calculation of the estimated signal, where a_i and b_i are the coefficients of the second order predictor and the sixth order predictor, respectively.

$$a_1(k) = (1-\gamma)a_1(k-1) + 3\gamma \cdot \mathrm{sgn}[p(k)] \cdot \mathrm{sgn}[p(k-1)] \tag{2.13}$$

$$a_2(k) = (1-\eta)a_1(k-1) + \eta \cdot \mathrm{sgn}[p(k)] \cdot \mathrm{sgn}[p(k-2)] \tag{2.14}$$

$$b_i(k) = (1-\kappa_1)b_i(k-1) + \kappa_2 \cdot \mathrm{sgn}[p(k)] \cdot \mathrm{sgn}[p(k-1)] \tag{2.15}$$

$y(k-1)$, $s_e(k)$, $d_q(k-i)$, $a_i(k-1)$ and $b_i(k-1)$ are the internal states and μ, γ and η are the forgetting factors.

Soft Decision Speech Decoder

This approach [FV01] consist of a modification of the speech decoder such that softbits (joint information of a received bit and its estimated instantaneous probability) are used instead of hardbits (just the hard-decided received information).

To employ softbit decoding, some probability information should be known, like the bit error probability of the received bits. In [FV01] two examples are provide in order to show how to calculate the a posteriori probability for an ADPCM parameter. The softbit speech decoding process consist of three steps:

- The first step is the calculation of the parameters transition probabilities. In order to obtain this value, the bit error probability is used.(Channel Information)
- The second step is the calculation of the a posteriori probabilities of the transmitted bit combination. For example, this a priori knowledge can be modeled by Markov processes. Markov models are used in order to model the transmitted source signal.
- The last step is the parameter estimation. The estimation criterion should be appropriate to the subjective speech quality. Two estimation methods, the Minimum Mean Square (MS) estimation and Maximum a Posteriori (MAP) estimation are proposed.

This mechanism can be applied to ADPCM for the 4 bits representation of the quantized residual signal sample $d_q(k)$ in the logarithmic domain. The equation (2.16) shows the representation of this parameter.

$$|d_q(k)| = 2^{|I(k)|} \cdot 2^{y(k)} \tag{2.16}$$

The entropy values for this codec, calculated by using the steps above, show that a zero order Markov process can be used. The best entropy value is 3.62 bits, so there are 0.38 redundancy bits (4 - 3.62). For the first order Markov process the obtained redundancy value is 0.43. For 32 Kbps ADPCM, the softbit decoder uses a estimation of the residual signal sample $d_q(k)$ in the linear domain, not in the logarithmic domain. By using this linear representation, an MS estimator can be employed, because there is a strong correlation between the squared residual signal and the speech quality. This MS estimation can be used instead the ADPCM defined quantization table. In order to make the term 2^y more error resilient, an additional MS estimation is employed.

Click-Noise Detection

This scheme [NDSK93] eliminates noise in the decoded signal by employing an error-detection mechanism, together with PCM differential detection and PCM overflow detection. The basis of the detection mechanism is that normal speech signals have few components in high-frequencies but error signals have more high-frequency components. Thus, the frequencies which correspond to an error can be detected and eliminated, by using a high pass filter. This filter is a differential detector in the time domain. The detection process is made through the following steps:

1. **PCM Differential Detection**. The detection process uses the PCM reconstructed signal $s_r(k)$ in the decoder. The difference between the current value and the previous one is calculated. This allows the decoder to know if the signal contains channel errors because errors increase the high-frequency components. The PCM difference between two consecutive PCM samples is calculated as follows:

$$\Delta\text{PCM}(k) = |s_r(k) - s_r(k-1)| \tag{2.17}$$

   If $\Delta\text{PCM}(k)$ is greater or equal than a predefined threshold, then the differential detection signal is activated.

2. **PCM Overflow Detection**. The click noise caused by channel errors in ADPCM signals induces a higher magnitude that can exceed the maximum value in the PCM dynamic range. If $|s_r(k)|$ is greater or equal than the threshold of overflow detection, the overflow indicator is activated.

3. **Click-Noise Cancel Signal**. If the differential detection signal or the overflow indicator signal are active, the click-noise cancel signal is generated.

By using a suppression circuit and the signal obtained after the above process, the click noise error can be eliminated by employing a suited low-pass filter.

Error Correcting Code (ECC)

In this method [KPHH86] the reliability of the words decoded by the ECC decoder is utilized to control the update of the inverse quantizer and also to stabilize the predictor in the receiver. The system model can be seen in Fig. 2.5.

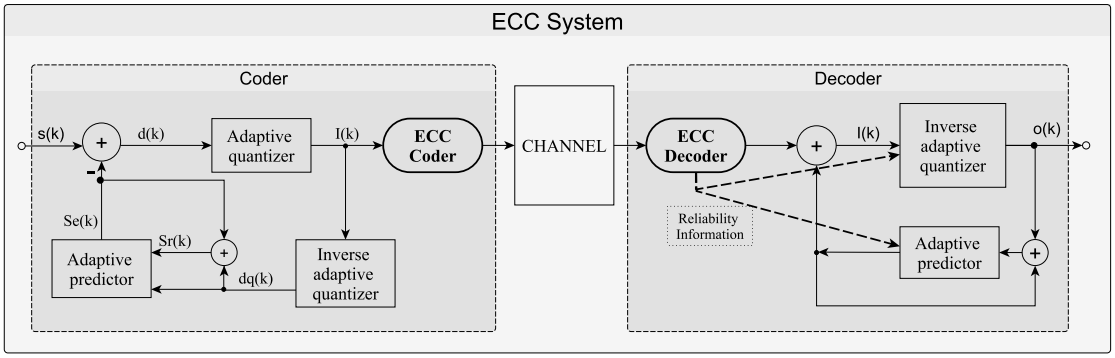

Fig. 2.5. Error Correcting Code system

The reliability of the words decoded by the ECC decoder is utilized:

- in the inverse quantizer to control the updating factor, in order to minimize the effects of undetected channel errors and miscorrected received words.

  In the conventional quantizer, the quantization is updated according to

  $$y(k) = y(k-1)^{\chi} \cdot W[I(k)] \tag{2.18}$$

  where χ is a damping factor and $W[I(k)]$ is a constant value determined by $y(k)$.

  In the modified version of the quantizer, the term $\rho(k)$ is a constant that determines the reliability of the decoded word

  $$y(k) = y(k-1)^{\chi \cdot \rho(k)} \cdot W[I(k)] \tag{2.19}$$

  where $\rho_i(k) \leq 1$ and i indicates the number of corrected bits in the word that contains $I(k)$.

- to stabilize the predictor in the receiver. The predictor coefficients are updated by including a term with the reliability of the decoded word.

2.2 Apt-x100

Audio Processing Technology Apt-x100 is a proprietary audio codec developed by APT Licensing [För01] [SS96]. The basis of the Apt-x100 compression process are techniques for signal redundancy removal. It is capable to reduce a 16 bits PCM audio sample to a 4 bits compressed audio sample, by performing a compression ratio of 4:1 without causing a perceptible degradation. The mentioned codec offers a notable bit error tolerance and low coding delay with low hardware complexity. Examples of applications that make use of Apt-x100 audio coding system are DBS radio, studio-transmitter microwave links, digital wireless microphones, Bluetooth, satellite broadcast links and A2DP stereo.

2.2.1 Description

The good performance of this codec is derived from the jointly utilization of the following components: sub-band ADPCM, Quadrature Mirror Filters (QMF), linear prediction, adaptive quantization and quantization with fixed bit allocation.

Sub-band ADPCM

Sub-band ADPCM is a coding system standardized by ITU-T [IT88]. It describes the coding system process for an audio input signal with bandwidth from 50 to 7000 Hz with a bit rate of 64 kbit/s. The spectrum is divided into four independent frequency sub-bands with the same bandwidth and in each band, a separated ADPCM coding method is achieved. The system has three basic modes of operation: 64, 56 and 48 kbit/s. Through the use of this sub-band encoding, the spectral redundancies included in the audio signal are reduced. This redundancy is caused by the energy content difference in each band. Since every subband has a different energy content, by separating the signal into subbands, the energies are separated too and the redundancies can be eliminated. If the band coding resolution is modified according to its energy, the quantization errors are reduced too.Subband coding primarily reduces irrelevance due to masking effects.

As a first step, the analog input signal is converted into a digital signal by using PCM with a resolution of 16 bits per sample. This 16 bit word is the input of the Apt-x100 codec. A succession of four 16-bits samples are processed in the Quadrature Mirror Filters block [EG77] [Joh80], which separates the input signal into frequency bands. The use of QMF allows independent encoding for each sub-band, avoids the aliasing effects due to samples decimation, reduces the effect of quantizing noise (by localizing it in narrow frequency bands) and avoids noise interference within sub-bands. It is used together with a particular bit allocation that follows perceptual criterion, such as allocating the energy levels by utilizing different number of bits. This allocation improves the signal-to-noise ratio and enhances the perception of the quantization noise.

The stages of the QMF process are:

- First the signal is passed through **antialiasing filters** which limit the bandwidth and satisfy the Nyquist theorem. The used filters are symmetrical with finite impulse response (FIR). One filter is a low pass filter and the other one is the high pass symmetrical filter.

The Fourier transform of the filters (Fig. 2.6) follows the relation:

$$|H_0(e^{j\Omega})| = |H_1(e^{j(\Pi - \Omega)})| \tag{2.20}$$

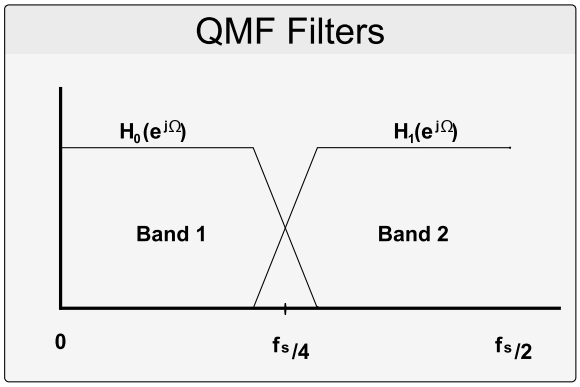

Fig. 2.6. QMF Filters

- Then, a **decimation** process is performed by using the half of the sampling frequency and as a result, two signals corresponding to the high and low frequency are obtained. In the decoder, an interpolation process allows a perfect reconstruction of the signal. The specific QMF operation applied to Apt-x100 divides the input signal into four frequency sub-bands. In this case, a two step QMF filtering is applied. The first one is a 64-coefficients QMF filter that divides the signal into two bands. Then, each band is divided again into two more bands by using a 32-coefficients filter. Both splitting processes are made according to:

$$s_L(k) = s_A + s_B \tag{2.21}$$

$$s_H(k) = s_A - s_B \tag{2.22}$$

where $s_L(k)$ and $s_H(k)$ are the signals corresponding to the low and hight frequency, respectively and

$$s_A = \sum_{i=0}^{\text{n° coefficients}} h_{2i} \cdot s(j-2i) \tag{2.23}$$

$$s_B = \sum_{i=0}^{\text{n° coefficients}} h_{2i+1} \cdot s(j-2i-1) \tag{2.24}$$

s_A and s_B are the splitted signals and h_i are the coefficients of the filter and j is the value corresponding to the current sampling interval.

The four independent frequency sub-bands resulting from the previous block are coded through a separated ADPCM coding method. The performing of ADPCM is explained in detail in section 2.1. For the signal within each sub-band, a "difference signal" is calculated from the subtraction of the input signal and a predicted value. Consequently, the difference signal can be seen as an error in the estimation of the current value. For the input signal, a predicted value is obtained by using the 122 previous predicted values of the signal. This value is subtracted from the current value and then adaptively quantized in a predefined number of bits. The adaptation adjusts the step size of the quantizer by using recent quantized values. It allows a dynamic adaptation to the signal and an optimal signal to noise ratio.

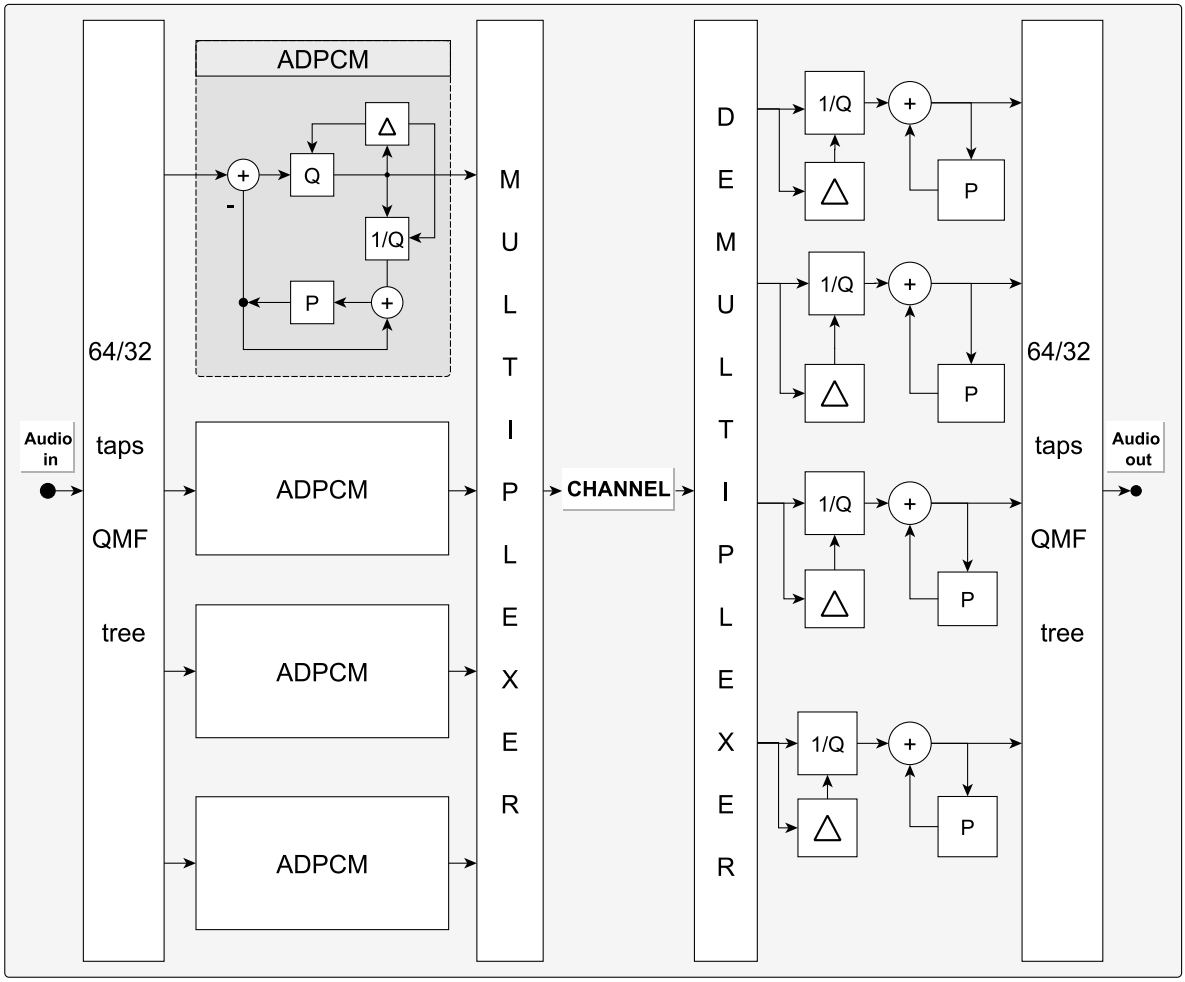

Fig. 2.7. Apt-x100

The chosen number of bands in Apt-x100 supports a balance between complexity and time delay in the filter, and achieves a good coding gain. With a higher number of sub-bands the gain of the coding is better, but the use of linear prediction counteracts possible gain losses derived from the use of a low number of sub-bands. The predictor utilized is a Backward Adaptive Predictor which removes the spectral redundancy that stays after the

sub-band coding. Its input is the inverse quantized signal and it is composed by two sections: a second-order predictor that models poles, and a sixth-order section that models zeros. The algorithm eliminates the remaining redundancy by calculating a predicted value of the current signal and by subtracting it from the present value, creating so the "difference signal". The better the prediction is, the smaller the difference signal is, and so more accurate is the quantization. The employed prediction method is adaptive and uses the past 122 samples to make the prediction coefficients and the step size adaptation. The term backwards means that there is no need to send any side information to the decoder, since the predictor uses just preceding values, which can be reconstructed by an inverse quantizer. With this procedure, it is assured that both encoder and decoder work with the same values and can generate identical values without sharing any information.

The difference signal is quantized by using a Laplacian Backward Adaptive Quantizer. The quantizer adapts itself to the signal by adjusting the quantization step size by taking into account the previous values of the quantizer output. The quantization resolution is smaller than the PCM resolution and is different for each sub-band. That is because the human ear reacts different to errors depending on the frequency where it is produced. The ear behavior and its relation with the perceived frequencies is the base of the psycho-acoustic model. This model is not directly implemented in Apt-x100, although the codec makes use of its properties. With the adaptive quantization, the low variations in the energy levels of the signal are used by adapting the step size in a dynamic way, in order to suit the signal levels. For the adaptation of the quantizer the 122 past samples are used. The step size adaptation is utilized for both the quantizer and the inverse quantizer, which inverse generates the output value to form the input of the linear predictor.

The quantized values are multiplexed to obtain the 16 bit word, that is the output of the Apt-x100 codec. In each band the number of assigned bits used to allocate these values is different. The number of bits for each band is: 8 bits for the lower frequency band, 4 bits for the medium-low frequency and 2 bits for medium-high and high frequencies. This mechanism is used jointly with sub-band encoding, since fixed bit allocation allows the encoding of the more energetic parts of the signal using a higher bit proportion than the parts with less energy content. The ear reacts with more tolerance to the parts of the signal with low energy levels, therefore the high-energy content is coded with less bit resolution. The output signal obtained after this process is a 16-bit word obtained from another word of 64 bits. In other words, a 4:1 reduction relation is achieved using Apt-x100.

Apt-x100 Encoder

A scheme of the encoding process is shown in Fig. 2.7. As a first step the input signal is divided into four frequency bands, with the same bandwidth, by using two consecutive QMF filter banks. The signal is splitted first into two frequency bands and then, each band is splitted again into another two frequency bands. ADPCM coding is applied in every band. The ADPCM blocks contain a backward adaptive prediction loop, that evaluate the current input samples together with the 122 preceding samples. In this loop a predicted value is obtained by using the previous predicted samples. The difference between this predicted value and the current value is calculated and quantized. The utilization of these difference

values instead of the complete signal allows a better quantization resolution. The process is repeated in every sub-band and finally four codewords are obtained (one from each band). The bit allocation depends on the sub-band, bands with high energy are encoded with more bits. At the end, the codewords are multiplexed to form a 16 bit word that is transmitted to the receiver.

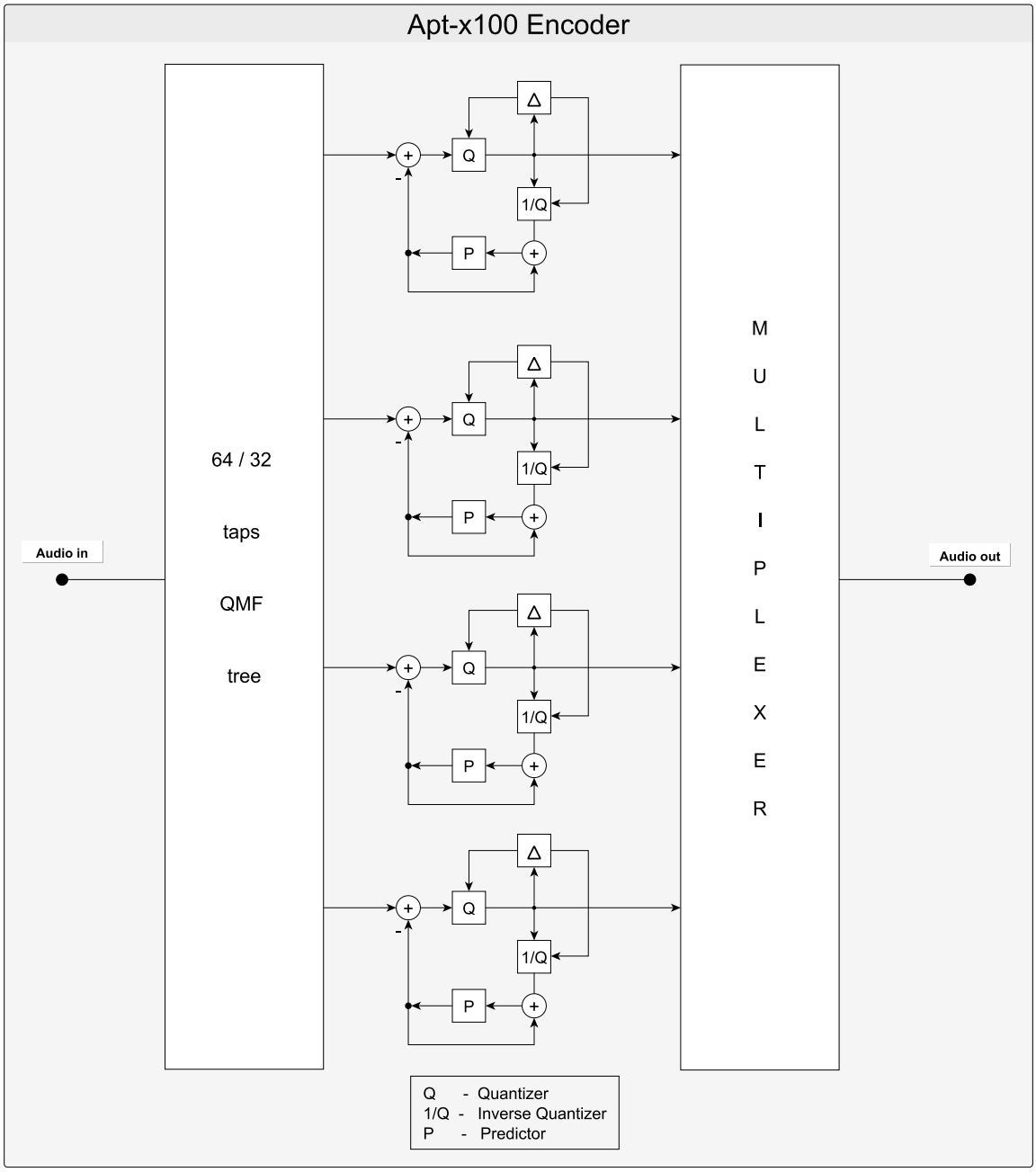

Fig. 2.8. Apt-x100 Encoder

Apt-x100 Decoder

In Fig. 2.9 a detailed scheme of the Aptx-100 decoder is shown. The input of the decoder is the 16 bit codeword obtained after the encoding process. This codeword is separated again, by utilizing a demultiplexer, into four codewords which are sent to the corresponding sub-band ADPCM decoder block. In each one, an inverse quantization operation is performed and as output the difference signal value is obtained. The signal passes through a prediction loop, where by using the 122 previous samples, the 16 bit codeword corresponding to the decoded value for the respective frequency is calculated. At last, the four 16 bit codewords are transformed using the inverse QMF filter banks to the decoded signal with the complete frequency range.

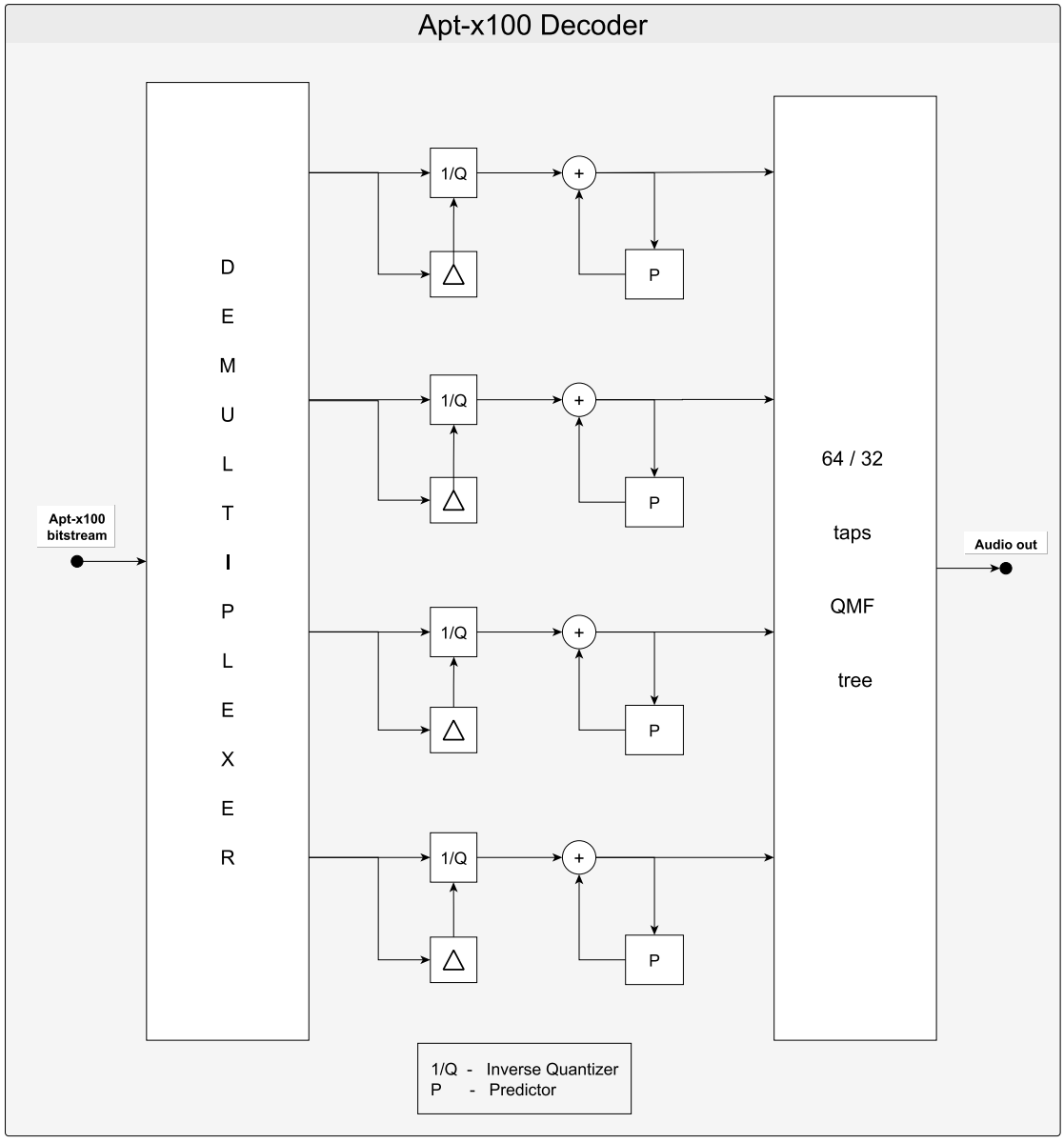

Fig. 2.9. Apt-x100 Decoder

2.2.2 Error Concealment Approaches

The codec presents a high resilience to bit errors and as a result, there is no need to include additional protection mechanisms. A bit error rate (BER) of 1:10,000 for normal programme material is normally inaudible, since the ear has a nonlinear response to different intensity levels. This resilience is derived from the utilization of sub-coding, backward adaptive prediction and quantization. With sub-band coding the errors remain in the frequency band and are limited just to that frequency range. Its effect in the complete frequency range is minimized due to the basis of the psychoacoustic model, where the masking effects hide the signals at nearby frequencies, making them inaudible. Sub-band audio coding together with backward adaptive prediction and quantization, where the past 122 samples are evaluated, make a pseudo concealment process by spreading the error effect to the adjacent samples within a sub-band. The effect of a bit error is proportional to the difference signal that is decoded in each sub-band. For signals that are highly predictable, the signal difference is too small and so, the effect of a bit error in the predictor and in the quantizer is small too. Evaluation was not possible due to unavailable encoder/decoder software.

2.3 AMR-WB+

The Extended Adaptive Multi-Rate Wideband codec (AMR-WB+) [ETS11] [MBB+05] [IEE06] was developed by the Third Generation Partnership Project (3GPP) as an expansion of AMR-WB. The new functionalities include one transform coding model, bandwidth extension, and stereo coding. The codec is designed to compress speech and audio signals at low bit-rate and good quality, hence was recommended for Packet-Switched Streaming Service (PSS), Multimedia Messaging Service (MMS), Multimedia Broadcast/Multicast Service (MBMS), IP Multimedia Subsystem (IMS), streaming audio application over the Internet, digital radio, video broadcasting and language-learning applications.

2.3.1 Description

The AMR-WB+ audio codec makes use of an hybrid technology that includes a time-coding model and a transform-domain coding model. ACELP (Adaptive Code Excited Linear Prediction) performs the time-domain encoding and provides good handle of speech signal, whereas TCX (Transform Coded eXcitation) accomplishes the transform coding model and adds the coding of rich sounds like music. Both ACELP and TCX employ linear predictive coding (LPC) analysis for modeling spectral envelope with different types of excitation for the LPC filter. ACELP uses Algebraic Codebook Excitation and Long-Term Prediction (LTP) analysis and synthesis and TCX quantizes the Fourier transformed weighted signal by using algebraic vector quantization (AVQ).

The input signal is processed in blocks of 2048 samples, then splitted into two frequency bands and down-sampled to obtain a critically sampled signal. After this process, the bands are segmented into blocks of 1024 samples, the so-called superframes. For the high-frequency band, each superframe is encoded with 64 bits using a bandwidth extension method and only the energy and spectral envelope are transmitted to the decoder. For the low-frequency band, each superframe is divided into four frames of 256 samples, which should pass through the core ACELP/TCX coder, where the mode selection in performed. This ACELP/TCX coder performs the selection of the encoding configuration.
Each frame of 256 samples can be encoded into four possible modes:

- In an ACELP frame of 256 samples
- In a TCX frame of 256 samples (short TCX frame)
- As a part of a 512 samples TCX frame (medium TCX frame)
- As a part of a 1024 samples TCX frame (long TCX frame)

The selection of the coding mode is based on the signal properties and can be done in either open-loop or closed-loop. There are 26 possible combinations for the four possible modes within a superframe. As for example, 4 ACELP frames; 4 short TCX frames; 2 medium TCX frames; 1 long TCX fame; 1 ACELP frame, 1 short TCX frame and 1 medium TCX frame, ... Closed-loop is a more complex algorithm that adds complexity to the encoder, since in this mode the input signal is encoded in all of the 26 possible combinations in order to obtain the best one. The coding mode selected maximizes the average segmental signal-to-noise Ratio (SNR) between the weighted speech and the synthesized weighted speech.
In open loop mode selection, the best combination is selected by using predefined algorithms

that examine and categorize the input signal. The open-loop mode selection is achieved through a three step process:

1. Excitation classification (EC). The first classification is done before the Linear Predictor(LP) analysis. The EC algorithm bases its selection on the frequency content of the input signal.

2. Excitation classification refinement (ECR). ECR is done after open-loop LTP analysis. The mode selection of the fist step is checked and verified.

3. TCX selection (TCXS). This step is done only if the number of ACELP modes selected in the previous steps is less than three. If this is the case, TCX mode is select.

One superframe is formed by one or more frames with 256 samples frames each. Each one of them follows always the same structure: a header followed by the data octets and the bandwidth extension part. If the frame encodes stereo audio, it includes the low and mid band parameters and one stereo bandwidth extension (BWE) before the normal bandwidth extension part (see Fig. 2.10)

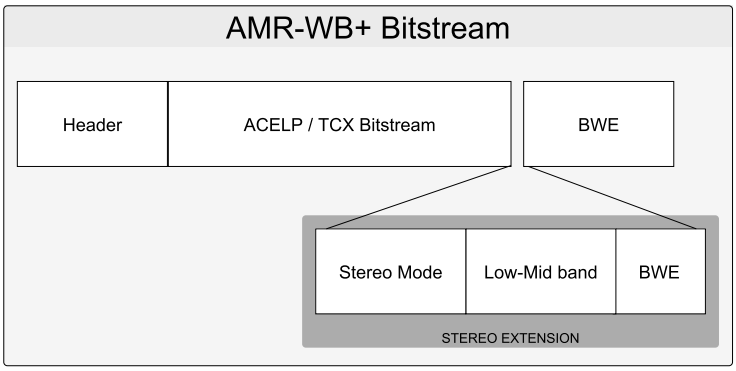

Fig. 2.10. AMR-WB+ Bit Stream

The header is formed by 2 octets, as presented in Fig. 2.11. This header indicates the coding parameters: the frame type, the position of the frame in the superframe and the sampling frequency.

- Frame type, indicates the combination of mono or stereo mode, the bit rate and the number of octets per frame.

- ISF mode, indicates the internal sampling frequency employed in the frame.

- TFI (Transport Frame Index), corresponds to two bits that show the position of this frame in the superframe.

The header follows the core bitstream. The octets corresponding to the data begin with two bits that represent the coding mode for the frame. There are four possible coding modes represented for the bit combination: 00 for ACELP, 01 for short TCX, 10 form medium TCX and 11 for long TCX.

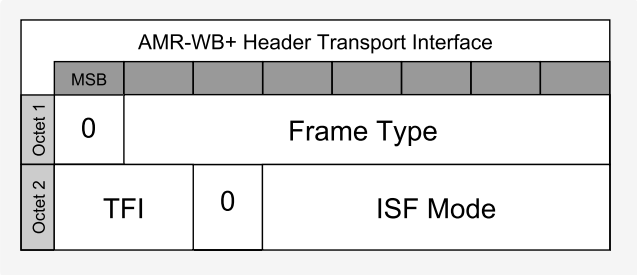

Fig. 2.11. AMR-WB+ Header

AMR-WB+ processes mono and stereo input signals. The coding process of the **mono** input signals begins with the down-mixing and resampling of the signal at the internal sampling frequency (25.6 kHz). Then, the signal is divided into two equal bands of frequency. One for the low-frequency band and the other one for the high frequency band. Both are down-sampled to the half of the internal sampling frequency to obtain a critically sampled signal. For the encoding, two different approaches are used: The high-frequency band is encoded using a bandwidth extension (BWE) method, and the low-frequency band is encoded using the ACELP/TCX encoding. Once the signal is encoded, the mode selection bits (indicating ACELP or TCX) and the low and high encoding parameters are send to the decoder.

For **stereo** input signals, the left and right channels are split into low and high frequency. The low-band of each channel is then down-mixed to form a mono signal which is encoded using the ACELP/TCX encoding method. The low-frequency part of the two channels is further decomposed into two bands.The very low frequency (VLF) band is critically down-sampled, and the side signal is computed. The resulting signal is semi-parametrically encoded in the frequency domain using the algebraic VQ. The frequency domain encoding is performed in closed loop by choosing among 40-, 80- and 160-sample frame lengths. The high frequency part of LF signals (Midband) are parametrically encoded. In the decoder, the parametric model is applied to the mono signal excitation in order to restore the high frequency part of the original LF part of the two channels. The HF part of the two channels are encoded by using parametric BWE. This method extracts a parametric representation of the spectral envelope and gains, which once quantized are sent to the decoder.

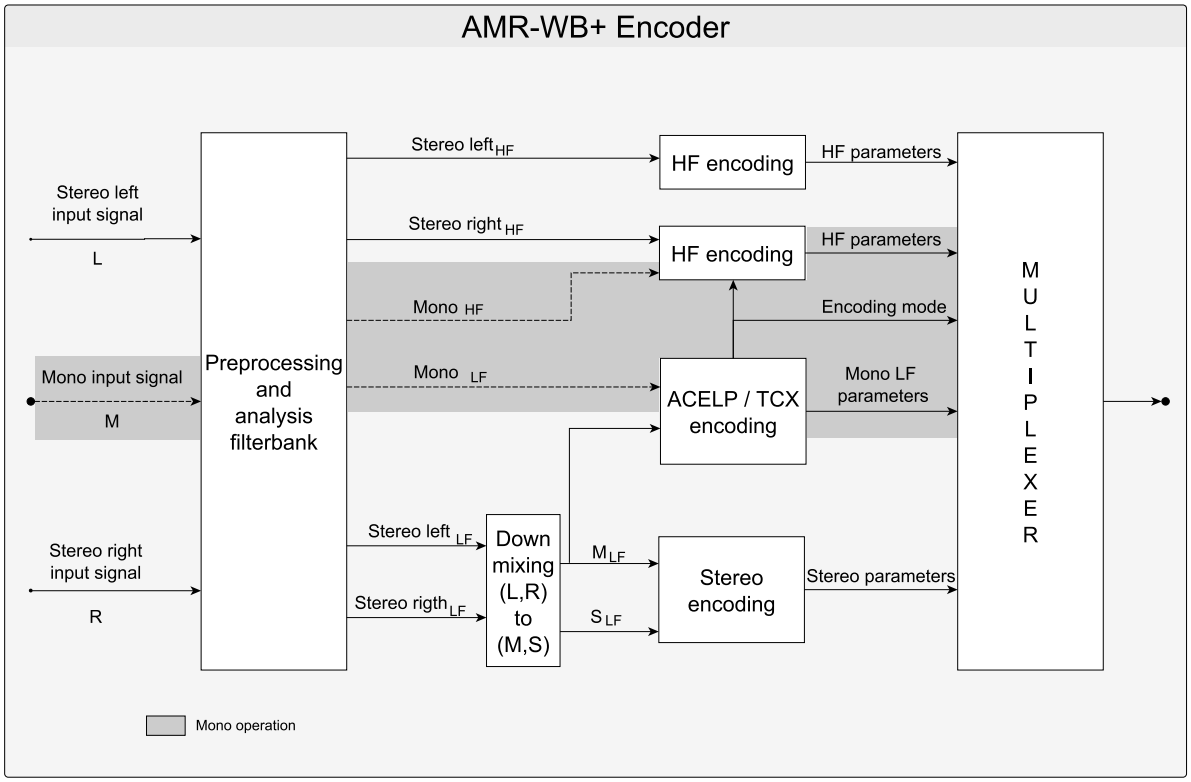

Fig. 2.12. AMR-WB+ Encoder

AMR-WB+ Encoder

TCX Encoding

For all TCX frame types a similar encoding method is used, whose only differences are the interpolation filter and the windowing [ETS11]. A detailed layout of the process can be seen in Fig. 2.13. In outline is the following:
As a first step, the signal is filtered using a weighting filter, which is time-variant and whose coefficients are linearly interpolated in the Immittance Spectral Pair (ISP) domain. As result, a weighted signal is obtained. This signal is windowed according to Eq. (2.25) to apply the overlap-add method. This method allows to easily deal with long signals, breaking them into smaller divisions, and minimizing the effects of quantization. The obtained signal is then converted to the frequency domain using a Discrete Fourier Transform (DFT) with a Fast Fourier Transform (FFT) algorithm to calculate the coefficients in a faster manner. Later on, the low frequencies of the spectrum are emphasized in order to decrease the perceived distortion. At this point, the spectrum quantization begins.

The window is defined as the concatenation of three sub-windows:

$$\begin{aligned} w_1 &= \sin(\tfrac{2\pi n}{4L_3}) \quad \text{for} \quad n = 0, ..., L_3 - 1 \\ w_2 &= 1 \quad \text{for} \quad n = 0, ..., N - L_3 - 1 \\ w_3 &= \sin(\tfrac{2\pi n}{4L_4}) \quad \text{for} \quad n = L_4, ..., 2L_4 - 1 \end{aligned} \tag{2.25}$$

$$\begin{aligned}
&L_3 = 0 \text{ when the previous frame is a 256-sample ACELP frame} \\
&L_3 = 32 \text{ when the previous frame is a 256-sample TCX frame} \\
&L_3 = 64 \text{ when the previous frame is a 512-sample ACELP frame} \\
&L_3 = 128 \text{ when the previous frame is a 1024-sample ACELP frame} \\
&\\
&\text{For 256-sample TCX: N=256 and } L_4 = 32 \\
&\text{For 512-sample TCX: N=512 and } L_4 = 64 \\
&\text{For 1024-sample TCX: N=1024 and } L_4 = 128
\end{aligned} \tag{2.26}$$

The quantizer is an Algebraic Vector Quantizer (EAVQ), in other words, an eight dimensional quantizer of variable rate, formed by spherical subquantizers, the origin and 5 concentric spheres, based in a lattice quantization scheme (specifically, the Gosset lattice E_8. For more reference, see [XA96]). The spectrum is divided into blocks of 8 dimensions and the quantization process consists of finding the index that represents that block in the base codebook. That is in fact, find the index of the nearest neighbor in the lattice.

The resulting index consists of three parts:

1. A codebook index, with the bit allocation for each 8-dimensional vector. The number of bits is the result of the multiplication of 4 times the sphere number that includes the vector.

2. A vector index, that identifies the vector in the base codebook.

3. An extension index k, for special cases where it is necessary to extend the base codebook.

To enhance the perceived quality of the TCX, a comfort noise factor is also calculated and quantized. After quantization, the spectrum is de-shaped and a inverse transformation is applied to achieve the quantized signal in the time-domain. In the quantizer, a global gain is applied to make the data fit the bit budget. Since that value of gain does not maximize the correlation between the two weighted signals (the original and the quantized signal), a new gain factor with this attribute must be calculated and quantized. This gain is employed to scale the quantized signal, then windowed and processed by an overlap-and-add method. The final signal employed for the excitation is obtained by filtering the signal.

The parameters sent to the decoder are: the Immittance Spectral Frequency (ISF) values, the comfort noise, the global gain and the algebraic VQ parameters. Due to TCX packets having different lengths, a management of the spectral information is necessary. To this end, the spectrum is split in parts, starting from the low frequencies, to form tracks, that are interleaved into the TCX frames. In each split, the VQ parameters are reordered and placed in different parts of the packet. The codebook numbers are represented by an unary code and allocated at the end of the packet. Its bits are placed downwards starting from the last position of the packet, whereas the corresponding index is allocated at the start of the packet.

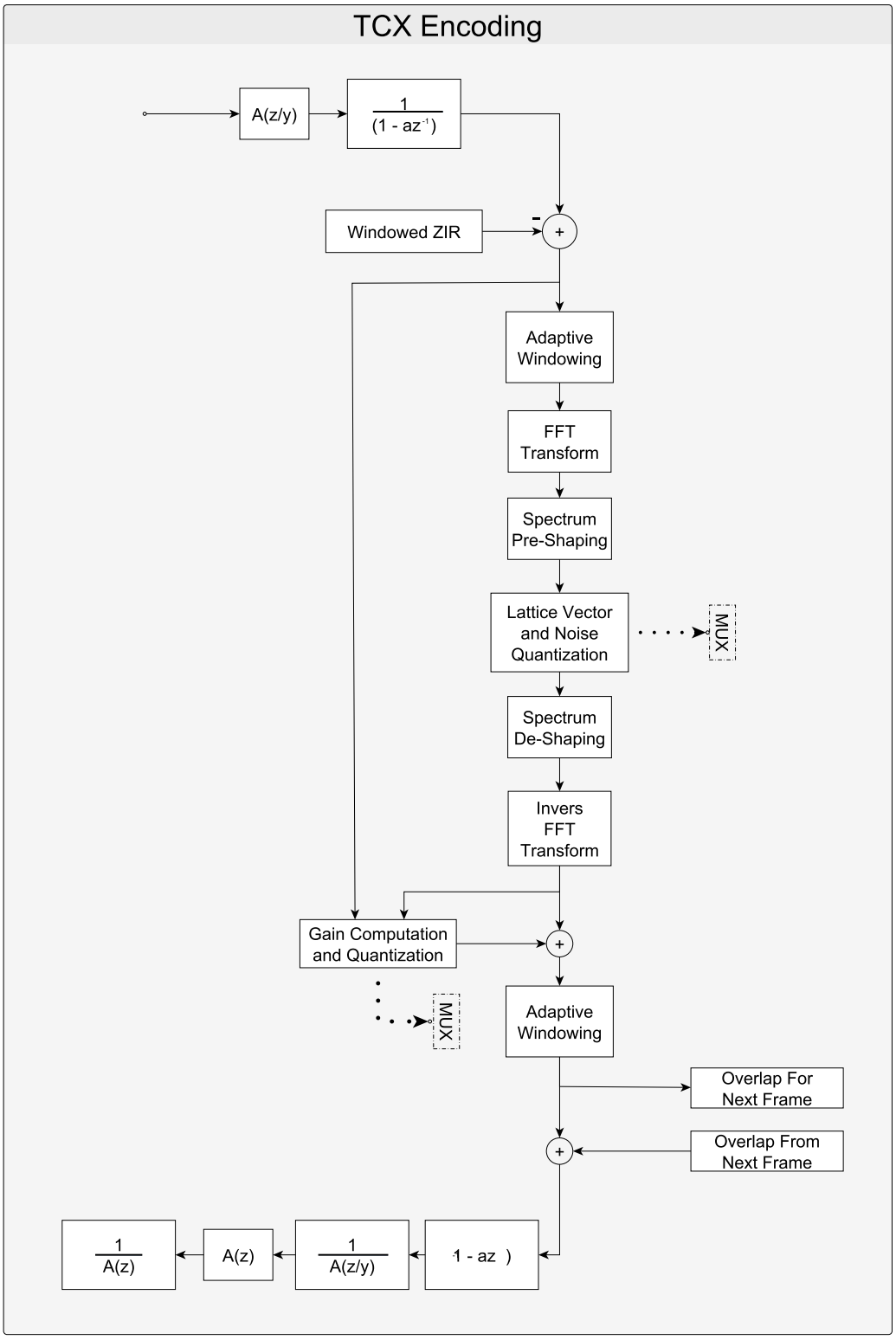

Fig. 2.13. TCX Encoding

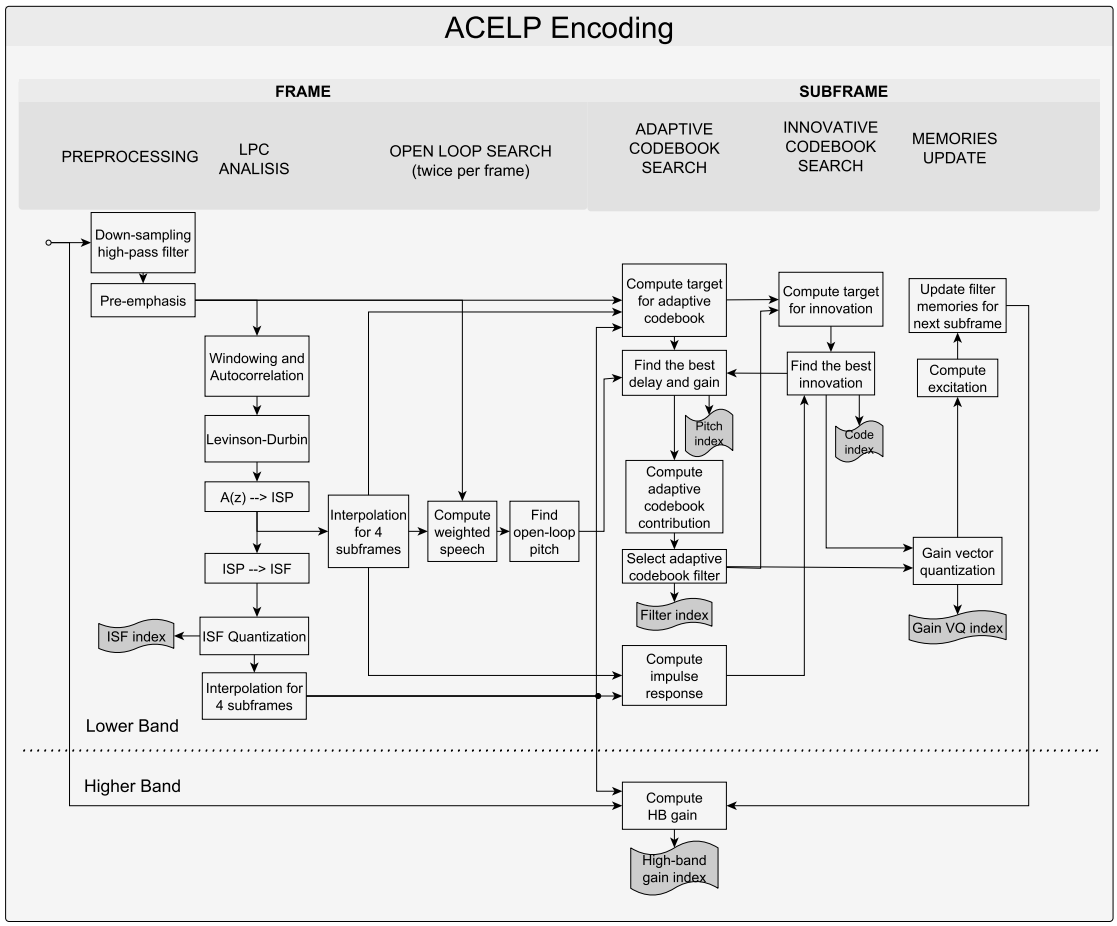

Fig. 2.14. ACELP Encoding

ACELP Encoding

ACELP is an hybrid coder based on the Code Excited Linear Prediction (CELP) [ETS11]. To extract the parameters that will be sent to the decoder, an analysis of the fixed codebook parameters, Long Term Prediction(LTP) and Linear Predictive Coding (LPC) are conducted in the encoder. The parameters include: the coefficients of the linear predictor (LP), the adaptive and fixed codebooks, the indices and the gains. To obtain these parameters the input signal is decimated, pre-emphasized and filtered by a high-pass filter and linear prediction is performed once per 20 ms frame.
The predictor parameters are then transformed to immittance spectrum pairs (ISP) to be quantized by using the split-multistage vector quantization (S-MSVQ). For each speech frame, a closed-loop analysis is performed to find the pitch lag and gain. The codebook arrangement is based on interleaved single-pulse permutation (ISPP) and has the following structure: vectors have 64 positions that are divided to form 4 tracks whose positions are interleaved. Signed pulses are arranged in the tracks to form the codebooks at different rates. The codebook indices correspond to the positions and signs of these pulses within the track.

AMR-WB+ Decoder

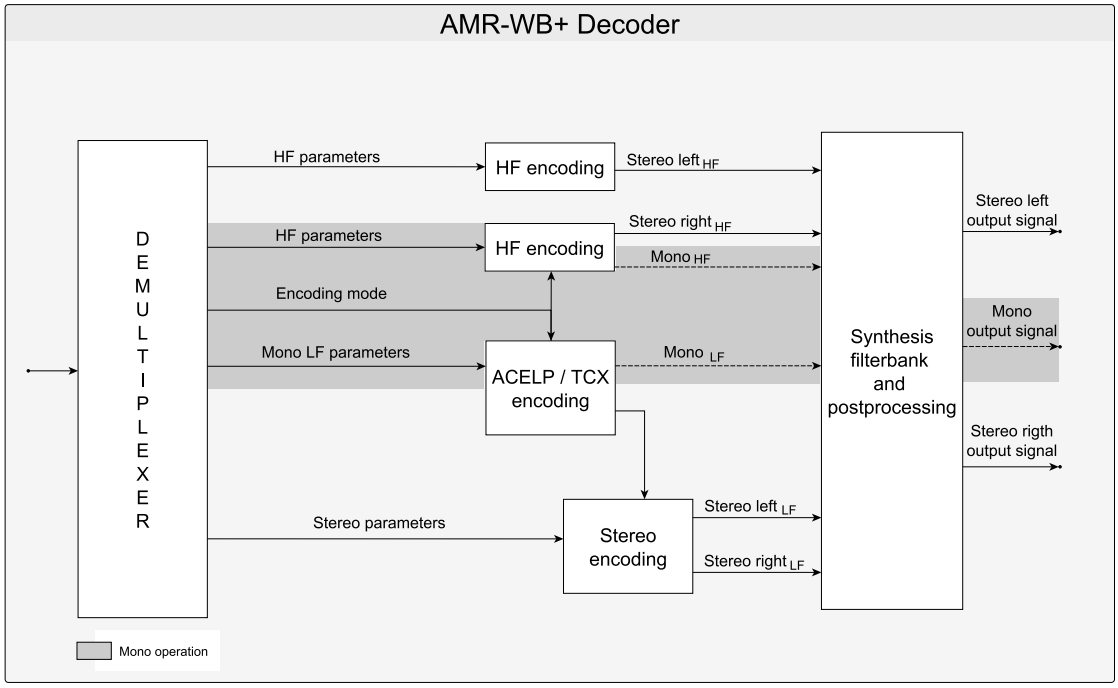

Fig. 2.15. AMR-WB+ Decoder

The function of the decoder consists of the decoding of the transmitted parameters (LP parameters, ACELP/TCX mode, adaptive codebook vector, adaptive codebook gain, fixed codebook vector, fixed codebook gain, TCX parameters, highband parameters, stereo information) and performing synthesis to obtain the reconstructed low-frequency and high-frequency signals.

2.3.2 Error Concealment Approaches

The AMR-WB+ codec includes a packet-loss management mechanism [IT03] at the decoder, so that when a packet loss case occurs, a concealment algorithm is employed and the algorithm takes into account the successive and previous packets. The applied mechanisms are different for mono and stereo decoding.

Mono Decoding

For mono decoding the employed processes are:

1. Mode extrapolation.

   With the mode extrapolation algorithm, the decoding mode of missing frames is recovered. For this purpose, it makes use of the last mode indicator of the previous mode. The decoder receives the four modes that the superframe includes in the form MODE = (m_0,m_1,m_2,m_3). Each m_k represents the mode of the 256 samples frame: 0 indicates ACELP, 1 indicates short TCX, 2 indicates medium TCX and 3 indicates long TCX. For example, if the superframe is encoded with long TCX, the mode vector is (3,3,3,3). The decoder uses a "Bad Frame Indicator" (BFI) vector in order to internally mark the missing frames. The vector has the form BFI = (bfi_0,bfi_1,bfi_2,bfi_3) where bfi_k = 0 denotes that the related frame of the superframe is correctly received and bfi_k = 1, denotes that the frame is missing. In that case the corresponding position in the mode vector is set to -1 and the missing mode is extrapolated. A counter with the number of missing frames is used in addition.
   The extrapolation logic follows the rules:

   - In the absence of bit errors, the mode can be extrapolated using the redundancy in the mode assignation logic for TCX. The long TCX packets are described by (3,3,3,3), and medium TCX packets by (2,2,-,-) or (-,-,2,2).
   - The extrapolation method selects ACELP mode only if the previous frame was encoding using ACELP too, because the ACELP frame-erasure method employs the pitch delay and codebook gains of the previous frames, and otherwise the concealment parameters would be out of date.
   - When three packets are lost and the only received mode is 3, long TCX mode (3,3,3,3) is not chosen. The extrapolated mode would be (1,1,1,1). Since in long TCX the gain and the spectral values are sent in four packets, and with just one packet is impossible to properly recover the information.

2. TCX bad frame concealment.

   For long TCX frames, the signal is supposed to be quasi stationary and the values of the missing frames can be interpolated from the values of the preceding frames. Spectrum de-shapping and spectrum extrapolation mechanisms are employed.

   - Spectrum de-shapping consists of the calculation of a ratio and its application to the quantized values of the spectrum before the quantization. The ratio is calculated as a division of the maximum energy and the energy of the current vector of

8 dimensions. If there is a missing packet, the maximum value should be predicted using the spectrum previously saved.

- Spectrum extrapolation consists of amplitude and phase extrapolation mechanisms. To obtain the missing amplitude, the amplitude spectrum of the preceding frame and current the frame is computed. Then, the gain difference between the energy of the received spectral coefficients from the current and previous frame is computed. The missing spectral coefficient amplitude can be extrapolated, multiplying this value by the amplitude of the previous frame.

  $A[k] = \text{gain} \cdot A_{\text{old}}[k]$ and $\text{gain} = \sqrt{\sum_{k,S(k)\neq 0} A[k]^2 / \sum_{k,S(k)\neq 0} S_{\text{old}}[k]^2}$

  To obtain the missing phase, the principle of group delay conservation for quasi-stationary signals is applied. The estimation of the group delay is calculated for the previous frame using:
  $\Delta\varphi_{old}(k) = \varphi_{old}(k) - \varphi_{old}(k-1)$

  Then, this value is used to calculate the phase of the missing spectral coefficients in a recursive algorithm.

  $\hat{\varphi}(k) = \hat{\varphi}(k-1) + \Delta\varphi_{old}(k) + \Delta\varphi_c(k), k = K+1, \ldots, K+N-1$

  where $\Delta\varphi_c(k) = (1/N) \cdot (\varphi(K+N) - \varphi_{old}(K+N) - \varphi(K) - \varphi_{old}(K))$

  $\hat{\varphi}(k) = \varphi(k)$ starts the recursion and k are the received bins.

Stereo Decoding

For stereo decoding the employed operations make use of the bad frame indicator and particular actions are used in low and mid band.

- Low-band

  The side signal error signal is used, this value is a flag that indicates if the stereo frame is bad and is derived in the TCX decoder. When one frame is lost, the balance factor, the signal error signal, the side signal and the left/right signal reconstruction need to be altered. The balance factor employed in the reconstruction of the side signal is derived from the previous value but attenuated by 0.9. Medium TCX and long TCX frames are calculated as in the mono case, but short TCX frames are reconstructed using a parametric model applied to the windowed mono signal. Finally, the signal is attenuated by a factor which is an estimation of the present average frame loss rate.

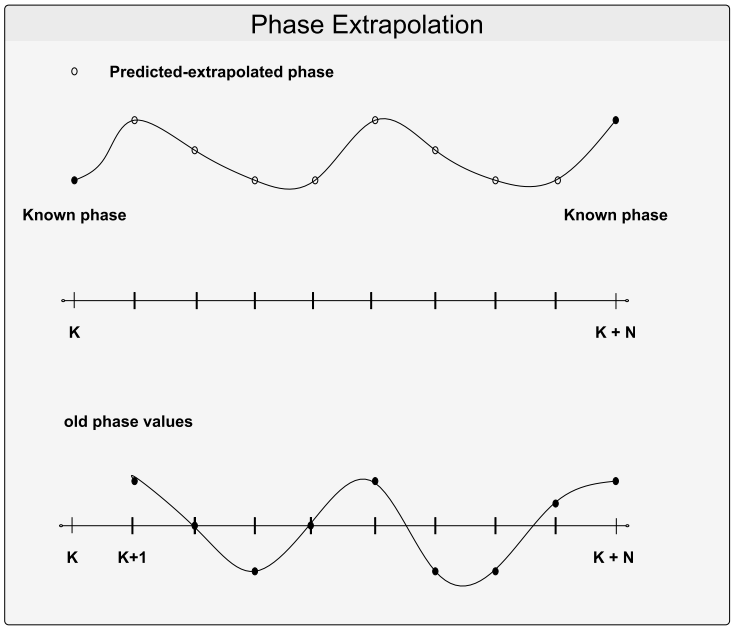

Fig. 2.16. Phase Extrapolation

- Mid-band

  In frame loss cases, the filter coefficients and the channel gains should be extrapolated, using a zero error signal in the predictive decoders.

2.4 AAC

The **Advanced Audio Coding (AAC)** is a perceptual audio codec developed by the Moving Picture Experts Group (MPEG) which performs a lossy compression encoding. Is part of the MPEG-2 [ISO04] and MPEG-4 [ISO05] specifications and has been standardized by ISO and IEC. The codec is used in real-time telephony and in high-quality audio compression applications like digital radio, mobile television and mobile music players.

2.4.1 Description

AAC bases its functioning on the elimination of the audio signal redundancy and on the elimination of the signal components that are not audible for the human ear. MPEG-4 AAC supports all the functionalities of MPEG-2 AAC and provides additional features. As most remarkable functions included in MPEG-2 can be mentioned: window shape adaptation using filterbank, spectral coefficient prediction, temporal noise shaping (TNS), bandwidth operation and bit rate scalability. The codec provides a set of required and optional tools and three different "profiles" (main, low and scalable), depending on the sampling rate. The selection of one profile can be determined by the desired quality for the output signal or the available resources.

The main tools can be described as follows:

- **Filterbank**
  The filterbank provides the spectral representation of the input signal. It consists of a set of Modified Discrete Cosine Transform (MDCT) filters with high resolution, a windowing process and an overlap-add function. The filterbank processes a block of samples using a window function and then applies the MDCT filters, overlap is performed using the half part of the preceding block and the half part of the following block. The result of the process is a sequence with either 256 or 2048 samples. The design of the filterbank allows a good adaptation to the input signal by using different window functions. The filter-bank performs a signal adaptive resolution by using four window sequences and two window shapes.

  The MDCT filterbank uses two different window shapes: a sine window (2.27) (2.28) and a Kaiser-Bessel derived (KBD) window (2.29) (2.30), taking into account the characteristics of the input signal to maximize the perceptual coding gain of the filter bank. Both windows allow perfect reconstruction and aliasing cancellation. An overlap-and-add function is applied after the windowing process, which adds the first half of every sequence to the second half of the previous block.

$$w_{\text{SIN\_LEFT}}(n) = \sin\left[(\frac{\pi}{N}) \cdot (n + \frac{1}{2})\right] \quad \text{for} \quad 0 < n \leq \frac{N}{2} \tag{2.27}$$

$$w_{\text{SIN\_RIGHT}}(n) = \sin\left[(\frac{\pi}{N}) \cdot (n + \frac{1}{2})\right] \quad \text{for} \quad \frac{N}{2} < n \leq N \tag{2.28}$$

$$w_{\text{KBD\_LEFT}}(n) = \sqrt{\frac{\sum_{i=0}^{k}[w'(i,\beta)]}{\sum_{i=0}^{N/2}[w'(i,\beta)]}} \quad \text{for} \quad 0 < n \leq \frac{N}{2} \tag{2.29}$$

$$w_{\text{KBD\_RIGHT}}(n) = \sqrt{\frac{\sum_{i=0}^{N-n-1}[w'(i,\beta)]}{\sum_{i=0}^{N/2}[w'(i,\beta)]}} \quad \text{for} \quad \frac{N}{2} < n \leq N \tag{2.30}$$

where N is the window length, w' is the Kaiser - Bessel kernel window function and β is the kernel window factor.

$$w'(n,\beta) = \frac{\sum_{i=0}^{\infty}\left[\frac{1}{i!}\left(\frac{\pi\beta}{2}\sqrt{1.0-\left(\frac{n-N/4}{N/4}\right)^2}\right)^i\right]^2}{\sum_{i=0}^{\infty}\left[\frac{1}{i!}\left(\frac{\pi\beta}{2}\right)^i\right]^2} \tag{2.31}$$

$$\beta = \begin{cases} 4 \text{ for } N = 2048 \\ 6 \text{ for } N = 256 \end{cases}$$

The window used for the left half of the first window is determined by the window utilized for the previous block. The four window sequence are ONLY_LONG_SEQUENCE (OLS), LONG_START_SEQUENCE (LSTS), EIGHT_SHORT_SEQUENCE (ESS) and LONG_STOP_SEQUENCE (LSPS). The window length for OLS, LSTS and LSPS is 2048 samples, whereas ESS is composed by 256 samples repeated eight times. ESS is used to analyze the transient part of the signal and the 2048 samples windows analyze quasi-stationary segments. The spectral coefficients are calculated using:

$$X(k) = 2\sum_{k=0}^{N-1} z(n)\cdot\cos\left[\frac{2\pi}{N}(n+n_0)\cdot\left(k+\frac{1}{2}\right)\right] \quad \text{for} \quad 0 \leq k < N/2 \tag{2.32}$$

where $z(n)$ is the windowed input sequence formed by $z(n) = w(n)\cdot s(n)$, n is the sample index, k is the index of the spectral coefficients, N is the window length and $n_0 = (N/2+1)/2$.

- **Spectral coefficient prediction**
  Prediction is applied to the spectral components resulting from the previous block, but only for OOLS, LSTS and LSPS and only if a coding gain is obtained. ESS is used for transient segments whose autocorrelation value can not be used for prediction. A second order backward-adaptive lattice predictor is employed, based on an Least Mean Squares (LMS) adaptation algorithm. The LMS algorithm uses the quantized spectral components of the two preceding frames to calculate the predictor coefficients and sends no additional side information. To ensure that prediction is only performed in coding gain cases, the predictors are grouped into scalefactor bands.

- **Temporal Noise Shaping (TNS)**
  This process handles the temporal shape of the quantization noise within each window by using a filtering process in certain parts of the spectral data.

- **Quantization**
  The quantization of the spectral data vector is made in three iterative levels. A first loop treats the input vector. A second inner loop quantizes the input vector by obtaining the Huffman codewords, corresponding to the quantized scale factors and the quantized coefficients. It adapts the step size to make that the output vector can be coded with the proper bit number. Finally, the outer loop controls the quantization noise included in the inner loop in order to attenuate it, if necessary, and calls the inner loop again.

- **Noise Coding**
  This module is performed inside the quantizer inner loop and processes a block of 1024 quantized spectral coefficients within a three steps algorithm. First, a spectrum clipping method that rejects the spectrum parts that lie below a threshold level, is applied in order to form noiseless coding segments. This segments are divided into sections and then coded using Huffman codebooks. The number of bits of the quantized spectral coefficients is minimized using a merge algorithm.

MPEG-4 AAC supports all the functionalities of MPEG-2 AAC and provides additional features like speed control, improved coding efficiency, error resilience and scalability. Among the tool improvements are perceptual noise substitution (PNS), the long-term prediction (LTP) and transform-domain weighted interleave VQ (TwinVQ).

- **Perceptual Noise Substitution**
  PNS performs noise substitution decoding of noise-like signal components. These components are detected through a scale-factor band and grouped into separate categories that are not quantized or included in the coding process. Instead, a noise substitution flag is transmitted and replaced at the decoder by pseudo-random vectors with a defined noise power.

- **Long-Term Prediction**
  The LTP is a forward adaptive predictor that decreases the redundancy between coding frames. It is performed in the frequency domain and used in the coding of tonal signals, since they require more accurate coding precision. Like in MPEG2 is applied only for long windows.

- **Transform domain Weighted Interleave Vector Quantization**
  The TwinVQ is based on an interleaved vector quantization of the transformed spectral coefficients and a perceptually weighted model. The first step of the quantization consists of flattening and normalization of the spectral coefficients by using a linear predictive coding. Then the coefficients are interleaved into subvectors and the perceptual weight of each vector is calculated prior to the codebook search.

AAC Encoder

Fig. 2.17 shows the encoder scheme for MPEG-4 in its last standardized version, since it includes the MPEG-2 functionalities with extended features. Three technologies are combined in this version: Advanced Audio Coding (AAC), Spectral Band Replication (SBR) and Parametric Stereo (PS). SBR is mechanism based on a bandwidth extension that allows the same coding results but with half bit rate. PS improves the coding efficiency by using the parametric representation of the audio signal stereo image.

The AAC codec is used to encode the low band, SBR encodes the high band, and PS encodes the stereo image in a parameterized form. In a typical aacPlus encoder implementation, the audio input signal at an input sampling rate of f_s is fed into a 64-band Quadrature Mirror Filter bank and transformed into the QMF domain.

AAC Decoder

The decoder demultiplexes the bitstream in order to separate the data stream into the parts that each block needs, parses that information to find the description of the quantized spectral data and the reconstruction data (like the Huffman and DPCM coded scalefactors). Then, it forms the constructed spectrum and processes it through all the active optional tools (prediction, intensity, TNS...), and finally forms the time domain reconstructed audio signal. In Fig. 2.18 appears the decoder scheme for MPEG-4 in its last standardized version.

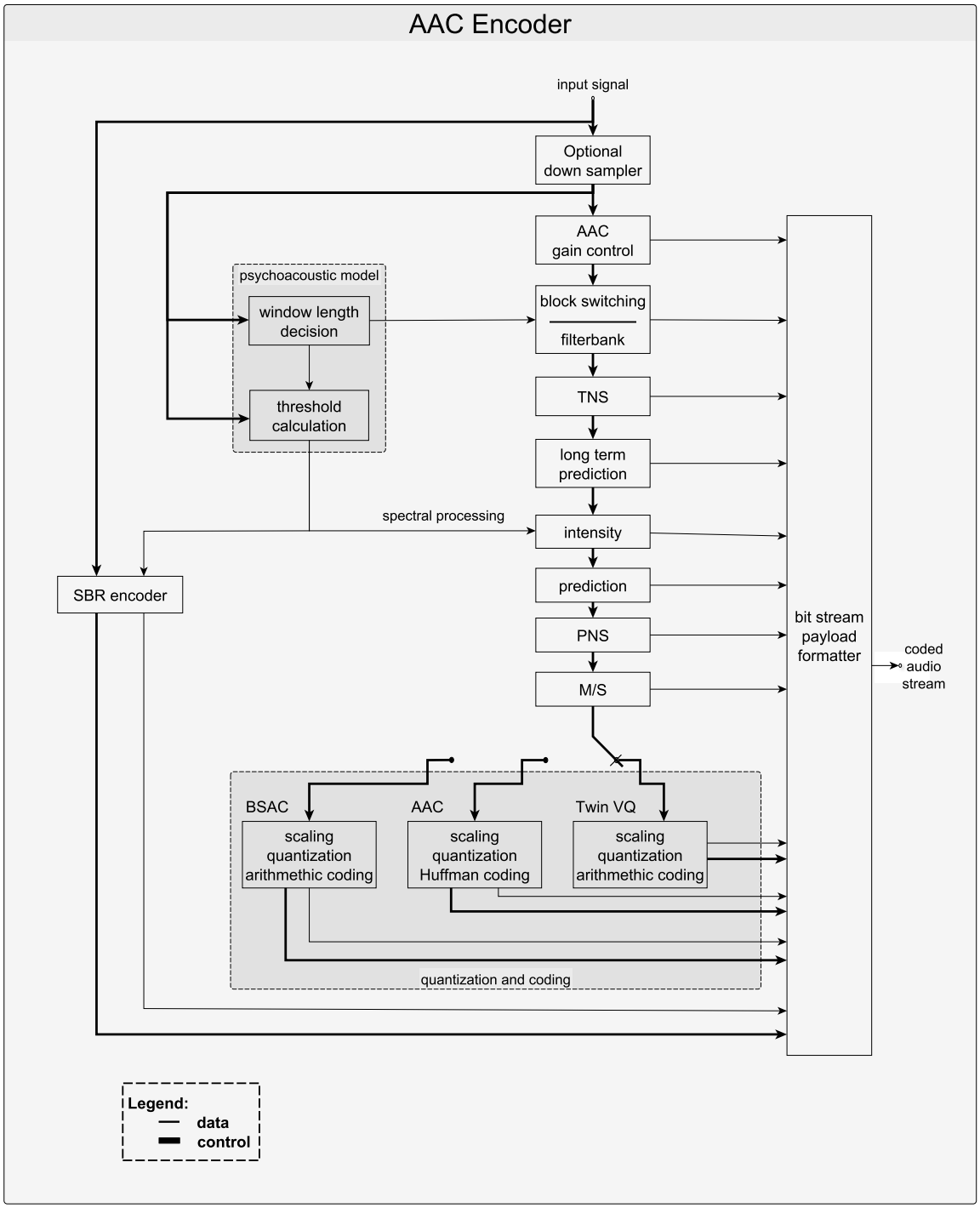

Fig. 2.17. AAC Encoder

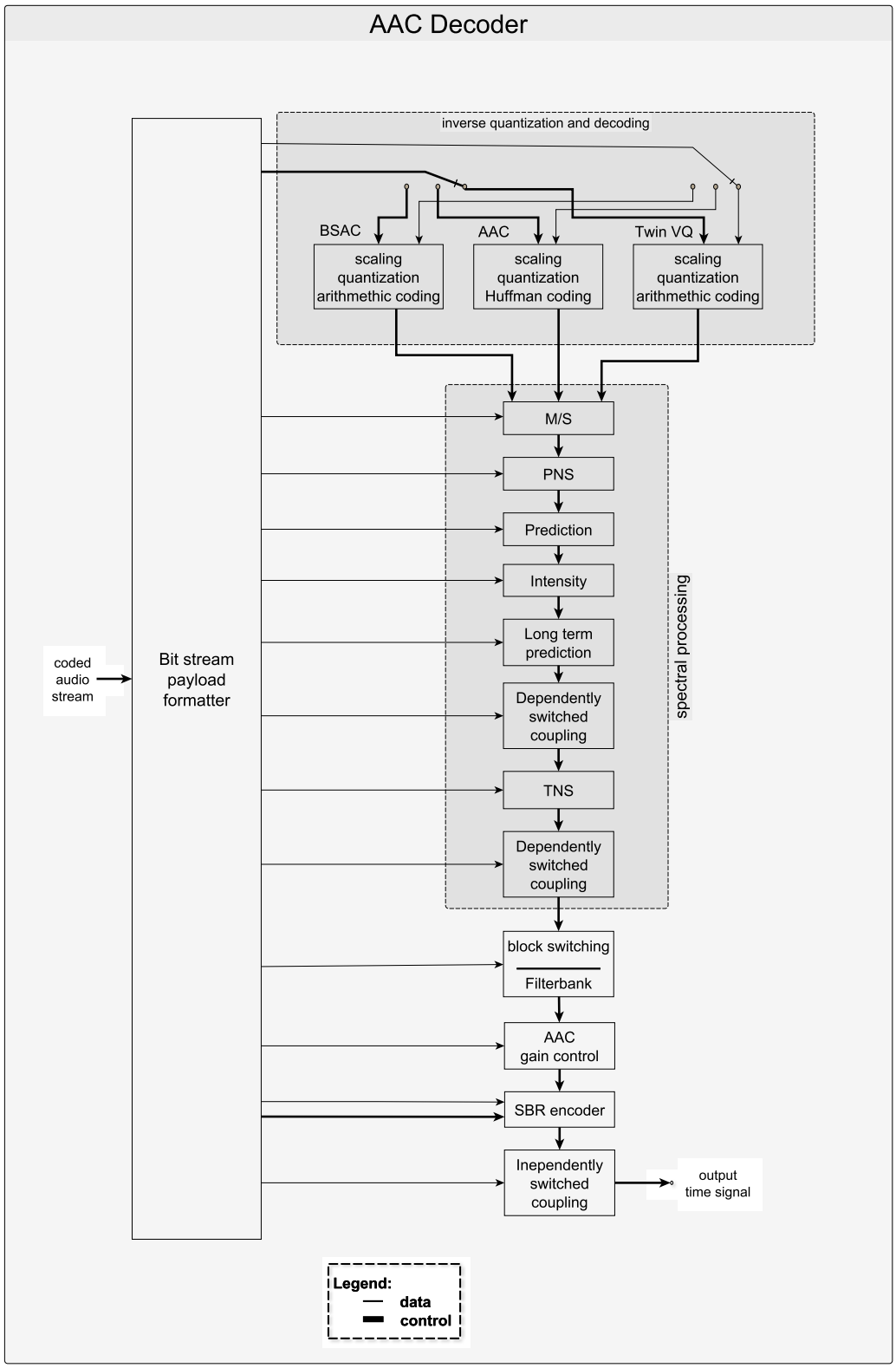

Fig. 2.18. AAC Decoder

2.4.2 Error Concealment Approaches

Various mechanisms for error management exist in AAC. Some of them are defined in the standard documentation and also new methods have been implemented. The related approaches are based on improvements in the modified discrete cosine transform (MDCT), in the data recovering and in the management of the critical data.

AAC Error Robustness Tool

AAC defines in the standard documentation a set of tools, which is dedicated to improve the decoded audio quality of the data transmitted over noisy and fast time-varying channels. The provided tools perform error detection, error concealment, error protection and error resilience techniques.

- Error Detection.
  Cyclic redundancy code (CRC) is the employed mechanism to detect errors. Due to AAC payload can be divided in different segments associated with different error sensitivity, various independent CRCs can be applied to each segment in order to protect, at least, the more sensitive parts against error-bits.

- Error Protection.
  The main error protection method in AAC is interleaving. The bits are reordered and grouped according to their error sensitivity and priority. For each group a different error protection can be applied, as for instance, Shortened Reed-Solomon (SRS) codes.

- Error Resilience.
  AAC defines three techniques for error resilience: Huffman codeword reordering (HCR), reversible variable length coding (RVLC) and virtual codebooks (VCB11). HCR avoids error propagation by sorting the codewords according to their priority and placing them at fixed segments in the bitstream. The priority codewords (PCW) are allocated at the beginning positions of this segments, which allow an independent decoding. The other codewords are allocated in the remaining positions of the segments by using an algorithm that minimizes the error propagation. RVLC allows the error-propagation reduction for scalefactors. The Huffman codes are replaced by RVLC codes, that perform an entropy coding of the scalefactors and achieve forward and backward instantaneous decoding with the same efficiency and therefore, with a decrease in the effects of bit errors. The VCB11 functioning is based on the idea of virtual codebooks. The codebooks are designed using a limited range of values where maximum and minimum spectral values are defined. If a too large spectral value appears, the codebook corresponding to the maximum value is selected. Errors often have a large spectral content. Therefore if a codebook is defined so that its values correspond to spectral values larger than the defined maximum, the errors can be identified and management techniques can be applied. In this technique the codebook 11 is used for this purpose, it allows 16 more codebook indices that are called virtual codebooks.

Click Noise Reduction Method

An MDCT-based click noise reduction [CBD+11] is a noise reduction technique applied in the MPEG-4 audio codec in order to improve the audio quality of decoded signals. This approach can be included as a function module of audio restoration. The related method involves two parts: click noise detection and click noise removal.

In the first step the click noise should be detected. That is why the window type used in MDCT is important. The method employed is the perceptual entropy of the psychoacoustic model. The click noise is a transient signal, owing to its short-time and abrupt property. So the window type that best adapts to these signals is the Eight Short Sequence (ESS) window, where the 2048 samples frame is divided in eight short overlapped frames of 256 samples. In the next step, the frame distorted by click noise is searched among the eight available. Making use of experimental results, which show that the high-frequency band spectrum of the noisy signal is flatter than the signal without noise, the 256 samples frame is retrieved.

The spectral entropy represents the flatness of the signal and is defined as:

$$H(m) = -\sum_{k=w\_low}^{\text{w\_high}} p(n,k) \cdot \log(p(n,k)) \tag{2.33}$$

where p(m,k) is the spectral probability, n is the frame index, k is the spectrum index and w_high and w_low are the maximum and minimum frequency band indices. The spectral probability is represented by:

$$p(n,k) = \frac{|S(n,k)|}{\sum_{k=w\_low}^{w\_high} |S(n,k)|} \tag{2.34}$$

Y(n,k) represents the k^{th} spectrum point for the n^{th} frame.

Once the spectral probability is calculated, a threshold to separate the signal with noise from the signal without noise can be chosen, as can be seen in Fig. 2.19. In order to decide the appropriate threshold value, a minimum error Bayesian decision is applied to minimize the probability of error when the probability of both signals is the same. The next action to take is to sharp the error. As the energy content is greater for the corrupt signals than for the clean ones, mainly at high frequencies, the frame with a higher energy value can be found looking at the energy changes among adjacent frames. When the frame has a spectral entropy larger than the threshold and the energy content is greater than the energy of the other frames, then the frame is marked as a frame with click noise, and its index is used in the click noise removal part.

When the click noise is detected, a Median Absolute Deviation (MAD) soft-thresholding principle is applied to estimate the noise and finally a spectral smoothing mechanism is used. The MAD soft-thresholding method estimates the noise level in the marked frames. The estimation is made by using the MDCT coefficients for high frequencies, because the energy of the audio signal is held in the low frequencies range. The estimated noise level is defined as follows:

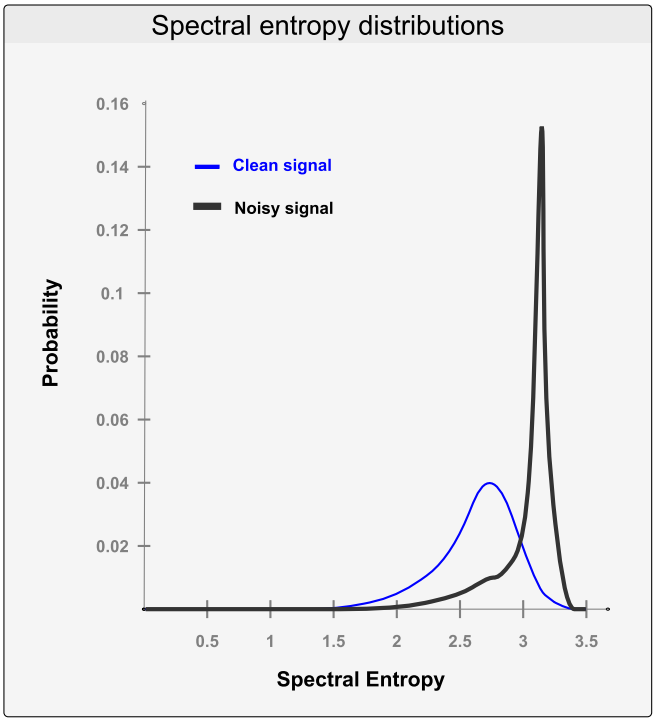

Fig. 2.19. Spectral entropy distribution of noisy and clean signals

$$\sigma_{v+} = \frac{MAD}{0.6745} \tag{2.35}$$

Since in the above calculation the contribution of the signal coefficients appears, a modified estimated noise level needs to be calculated. In order to obtain this value two hypotheses are defined:

$E_0 : Q_k = V_k$ just click noise is present
$E_1 : Q_k = U_k + V_k$ click noise and signal are present

where U_k and V_k are the k^{th} MDCT coefficients of the signal and the click noise. According to this hypotheses the modified estimated noise level is calculated using (2.36):

$$\sigma_{v,corr} = \frac{MAD(\text{high frequency of } Q_k \text{ where } \; p(E_1|Q_k) < v)}{0.6745} \tag{2.36}$$

$p(E_1|Q_k)$ can be computed according to

$$p(E_1|Q_k) = \frac{p(E_1) \cdot p(Q_k|E_1)}{p(E_1) \cdot p(Q_k|E_1) + p(E_0) \cdot p(Q_k|E_0)} = \frac{1}{1 + \sqrt{1+\xi_k} \cdot \exp(\frac{-\gamma_k \xi_k}{2(1+\xi_k)})} \tag{2.37}$$

where γ_k and ξ_k are the a posteriori and a priori SNR respectively. v is the threshold parameter that indicates if the value is appropriate for the calculation. Its value is obtained by

using a training result for different SNR.

Finally, the MDCT coefficients $\hat{X}_k$ for the recovered signal are derived by using the modified estimated noise level:

$$\hat{X}_k = \text{sign}(Q_k)(|Q_k| - \sigma_{v,\text{corr}}) \quad \text{if } |Q_k| > \sigma_{v,\text{corr}} \quad (0 \quad \text{otherwise}) \tag{2.38}$$

$$\tag{2.39}$$

Once the coefficients are retrieved, a spectral smoothing method is applied, because the energy in contiguous frames is not constant and the thresholding method applied is independent in contiguous frames. The smoothing process is achieved according to:

$$\tilde{X}_m(k) = \alpha(k) \cdot \hat{X}_m(k) \tag{2.40}$$

$\alpha(k)$ is the energy scaling factor:

$$\alpha(k) = \frac{\sum_{k \in b} |X_{m+1}(k)|^2 \cdot \sum_{k \in b} |X_{m-1}(k)|^2}{2 \sum_{k \in b} |\hat{X}_m(k)|^2} \tag{2.41}$$

Robust Critical Data Recovery

In the following method [HHK+10], transmission channels that generate noise at bit level are presupposed. Also assumes that reliable information is handled by the decoder because the whole process is done in the receiver terminal. Frames in AAC consist of two headers (a fixed one and a variable one), one data segment and an "end of frame" indication. In the data block are located a global gain, an individual channel stream (ICS), a section field, a scalefactor field, the spectral data and the optional data stream element. The critical data for AAC are the headers, the ICS and the section field. A CRC is defined an optional CRC that identifies errors in the 192 first bit position of the frames. However, this value does not cover all critical field of the frame.

This approach makes use of:

- The CRC value.
- The residual redundancy that appears in the headers after the encoding process.
- The redundancy in terms of correlation between the headers of consecutive frames.

The fields covered by the CRC, defined as c, can be separated into four types:

- Constant fields, K_n
  These fields are supposed to be known.
- Predictable fields, E_n
  They can be entirely calculated by using another information R_n, as for instance the correlation between consecutive packets and information from the lower layers according to the Open System Interconnection model.
- Unknown fields, D_n
- Other fields, O_n
  It contains data, which is less critical than the rest and whose values can be decoded with the appropriate methods.

C_n accumulates the critical data of the past frame: $C_n = \{K_{n-1}, E_{n-1}, D_{n-1}\}$.
The soft information received by the decoder is represented by: $R = [R_K, R_E, R_U, R_O, R_c]$ and only the information in D needs to be estimated. By using all the other values, a Maximum A Posteriori (MAP) criterion can be employed to retrieve the D fields.

$$\hat{u}_{\text{MAP}} = \arg\max_{D\in\Omega_D} P(D|K,E,C,R_D,R_O,R_c) = \arg\max_{D\in\Omega_D} (P(R_D|D)\cdot\psi) \quad (2.42)$$

where

$$\psi = \sum_{O} P(O)\cdot P(R_O|O)\cdot P(R_c)|F([K,E,D,O]) \quad (2.43)$$

and where F is the CRC encoding function.

Although the last equation requires big computational complexity, there are various approaches that reduce the complexity of the operations.

Error Resilient Scheme

This scheme applies interframe shuffling as a method of error concealment [Kor02] [KW03]. In perceptually audio codecs, the information can be divided into categories according to their sensitivity to errors:

- Critical data
  Without this information the decoding process can not be carried out. This data includes the length and the type of the MDCT window, the section data that includes the Huffman codebook indices and the global gain for scalefactors.

- Important but not vital data
  Without this data the decoding process can be achieved, but the audio output could be severely disturbed. This classification corresponds to the scalefactors, which characterizes the encoded spectral data range.

- The encoded spectral data
  Without this section the frame can be properly decoded and the audio quality remains acceptable. In this section the quantized MDCT spectral coefficients are included.

The aim of this mechanism is the best preservation of critical data and achieve that losses occur within the less critical part, in order to facilitate the error management. Two mechanisms can be applied: inserting multiple copies of the critical data into various packets or forming separate payloads for critical and non critical information. The critical information can be transported over the network interface level in a TCP packet and the remaining part over UDP.

The scheme proposes two mechanism for error resilience:

- Fixed slot frame allocation:
  Each packet is divided in parts of fixed length, one for each information type and one more section reserved to manage the overflow in the other parts.

- Linear coding for scalefactors:
  The value of the scalefactors is approximated by using a linear model and the Minimum Least Squares (MLS) method. The values of the approximated scalefactors are calculated through the following equation:

$$S_{F_i} = \beta + \alpha \cdot i \tag{2.44}$$

  where i is the index of the scalefactors, N is the number of scale factors and the β and α parameters are calculated according to:

$$\beta = \frac{\sum_{i=0}^{N} i^2 \cdot \sum_{i=0}^{N} S_{F_i} \; - \; \sum_{i=0}^{N} i \cdot \sum_{i=0}^{N} i \cdot S_{F_i}}{N \sum_{i=0}^{N} i^2 \; - \; \left(\sum_{i=0}^{N} i\right)^2} \tag{2.45}$$

$$\alpha = \frac{N \cdot \sum_{i=0}^{N} i \cdot S_{F_i} \; - \; \sum_{i=0}^{N} i \cdot \sum_{i=0}^{N} S_{F_i}}{N \sum_{i=0}^{N} i^2 \; - \; \left(\sum_{i=0}^{N} i\right)^2} \tag{2.46}$$

The parameter d replaces the previous value of the global gain and as a result, α should be included as critical data and a new Huffman table needs to be designed.

The three employed mechanisms for error concealment are:

- muting for the critical data, because if the packets are delivered using different network protocols, losses in critical data are very improbable.
- linear approximation for the scalefactors by using the previous model.
- linear interpolation for the QMDCT spectral coefficients with adjacent coefficients.

2.5 MPEG-1 Layer II and Layer III

MPEG-1 [Pan95] is an audio compression mechanism developed by the Moving Picture Experts Group (MPEG) and standardized by ISO-IEC [ISO93]. It operates at sample rates of 32, 44.1, and 48 kHz on 16-bit PCM input data with available bit rates of 32-192 kb/s for mono and 64-384 kb/s for stereo. MPEG-1 supports various modes: mono, stereo, dual independent mono, and joint stereo. The codec performs a lossy compression that can be perceptually lossless by exploiting the perceptual properties of the human auditory system. The reduction is made by applying perceptual techniques, where the irrelevant information included in the audio is removed. Since the human auditory system has perceptual limitations, this operation does not produce audible distortions. MPEG-1 is used, for instance, as audio and video storage on CD-ROMS, in Internet Streaming, for the audio coding in Digital Audio Broadcasting (DAB), for audio transmission over ISDN and in DVD, DVB-T, DVB-C and DVB-S.

2.5.1 Description

MPEG-1 provides a three layer architecture with different complexity, delay and audio quality, depending on the application. Each layer improves the compression performance by increasing the encoder and decoder complexity.

- **Layer I** is the simplest one. It performs the mapping of the input signal in 32 sub-bands, formats the data into blocks, applies adaptive bit allocation by using a psychoacoustic model and quantizes the obtained data through compressing and formatting techniques.
- **Layer II** includes more functionalities, by adding scalefactors, a different frame format and a modified bit allocation.
- **Layer III** is the most complex layer. It uses a filterbank with a finer frequency resolution, a nonuniform quantizer, adaptive segmentation and entropy coding.

Fig. 2.25 and Fig. 2.26 show bock diagrams of the encoding and decoding process in MPEG-1. The coding mechanism processes the input audio simultaneously through two different blocks: a filter bank and a psychoacoustic model. The filterbank splits the input signal into 32 frequency sub-bands that approximate critical bands. In the psychoacoustic model block, the noise masking is calculated to decide the quantization and coding of the input signal. In the bit/noise allocation block, the output of the psychoacoustic model block is employed to obtain the code bits number that can be used for the quantization of the subbands. The samples are quantized, coded and allocated within a frame in the last block. Additional data can be included. The decoder reads the bitstream created in the encoder, recovers the quantized subband values and uses these values to obtain the audio signal.

In the following a description of the filterbank and the psychoacoustic model is presented.

- **MPEG-1 filterbank.**
  The filterbank [Shl94] is employed in all three layers. The employed filter is a “polyphase filter”. It splits the input signal into a given number of equidistant sub-bands, which are subsampled by a factor equal to the number of subbands. So they are critically sampled. The number of frequency subbands with the same bandwith in MPEG-1 is 32.

In the encoder, the analysis phase of the filterbank is performed. In this phase (shown in Fig. 2.32), the input is processed by using 32 samples at the same time and shifting them into a 512 sample buffer B_1. The content of B_1 is multiplied by a window function W and the result of the preceding operation is stored in the buffer B_2. The content of B_2 is divided into eight vectors of 64 elements which after an addition operation are stored in another vector. This vector passes through a variant of the Modified Discrete Cosine Transform (MDCT) and as a result, the 32 sub-band values are obtained.

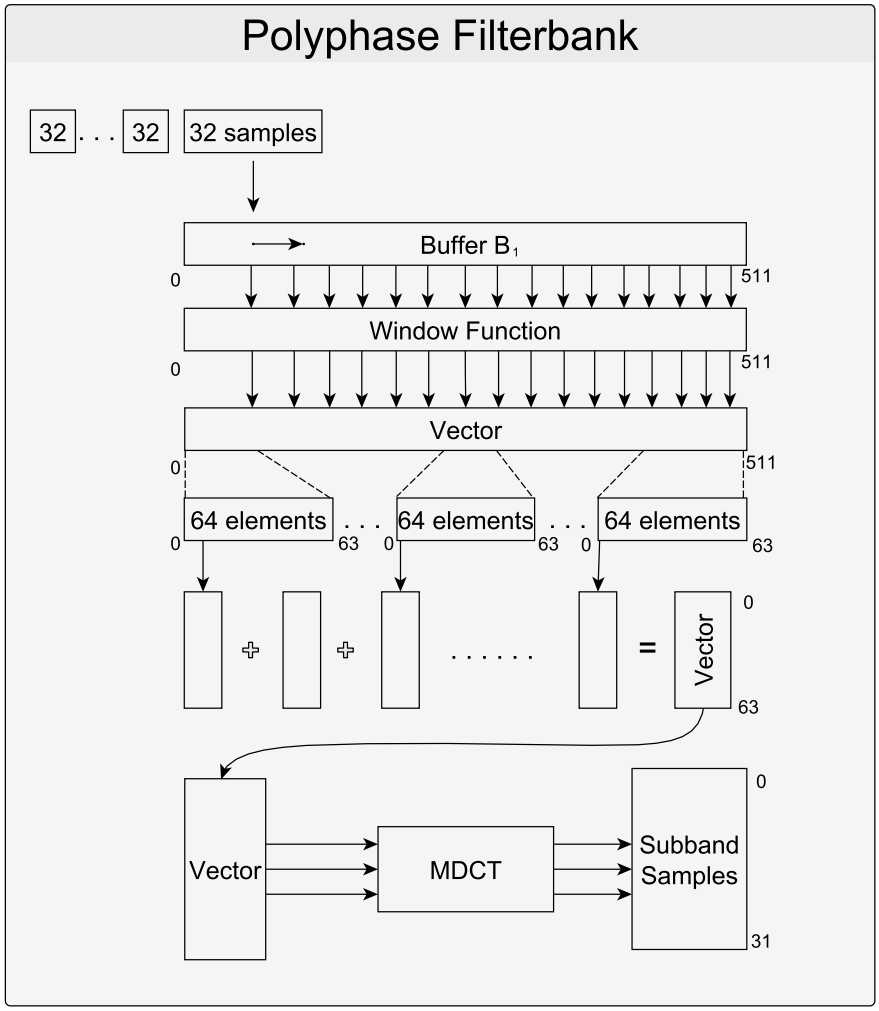

Fig. 2.20. MPEG-1 Polyphase Filterbank

The following equation (2.47) is the representation of the filter bank output.

$$X[m] = \sum_{l=0}^{63} \sum_{j=0}^{7} \cos\left[\frac{(2 \cdot m + 1) \cdot (l - 16) \cdot \pi}{64}\right] \cdot W[l + 64j] \cdot B[l + 64j] \qquad (2.47)$$

where m is the subband index that has values between 0 and 31, $X[m]$ is the filter output sample for subband m, W is the analysis window and B is an audio input sample from

the 512-sample buffer. In the decoder, the synthesis operation reverses the previous operations in order to obtain the reconstructed samples.

- **MPEG-1 psychacoustic model.**
  The resultant frequency subbands can be seen as an approximation to the critical bands. Critical bands are frequency ranges where one tone blocks the perception of another tone. These bands simulate the passband filters found in the cochlear within the human ear. In the psychacoustic model block, the audio signal is analyzed in order to calculate the masking noise available for the bands. The codec provides two methods to implement this model and the differences between them are based on the implementation complexity. The steps employed for both methods are:
  - Time-align audio data.
    This step centers the relevant information contained in each frame. So that information corresponds to the center of the psychoacoustic analysis window.
  - Convert audio to a frequency domain representation.
    A finer time-to-frequency mapping should be done in order to make a more accurate masking threshold calculation. Both models use a Fast Fourier transform (FFT).
  - Process spectral values in groups related to critical band widths (by employing perceptual quanta).
  - Separate spectral values into tonal and non-tonal components. That is because their masking abilities are different.
  - Apply a spreading function to distribute the signal masking properties across its surrounding critical band.
  - Set a lower bound for the threshold values, by utilizing an empirically determined absolute masking threshold.
  - Find the masking threshold for each subband.
  - Calculate the signal-to-mask ratio as a relation between the signal energy within the subband and its minimum masking threshold.

Layer Description

The three layers have different implementation complexity. Layer I is the simplest and Layer III is the most complex. The three layers use the same header as presented in Fig. 2.21 but have different payloads.

- **Layer I**.
  This layer uses frames of 384 audio samples formed by 12 samples from each subband. The frame has the following fields: the header, an optional CRC, the fields corresponding to the quantized samples and one field for auxiliary data. The quantized data part consists of a bit allocation field, a scalefactor field and the sample values. The fields in the frame can be observed in Fig. 2.22. Each of the 12 samples group is represented by a number of bits that is indicated in the "bit allocation field". The scalefactor is the number that multiplied by the quantizer output value produces the quantized value for the subband.

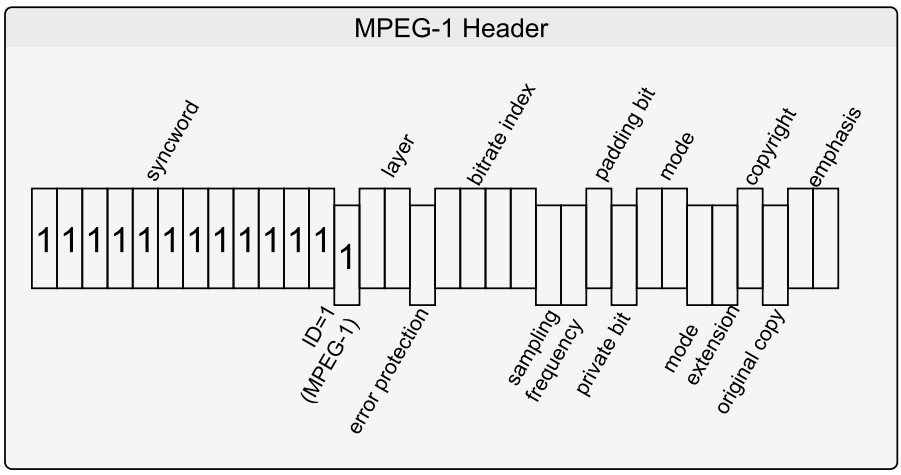

Fig. 2.21. MPEG-1 Header

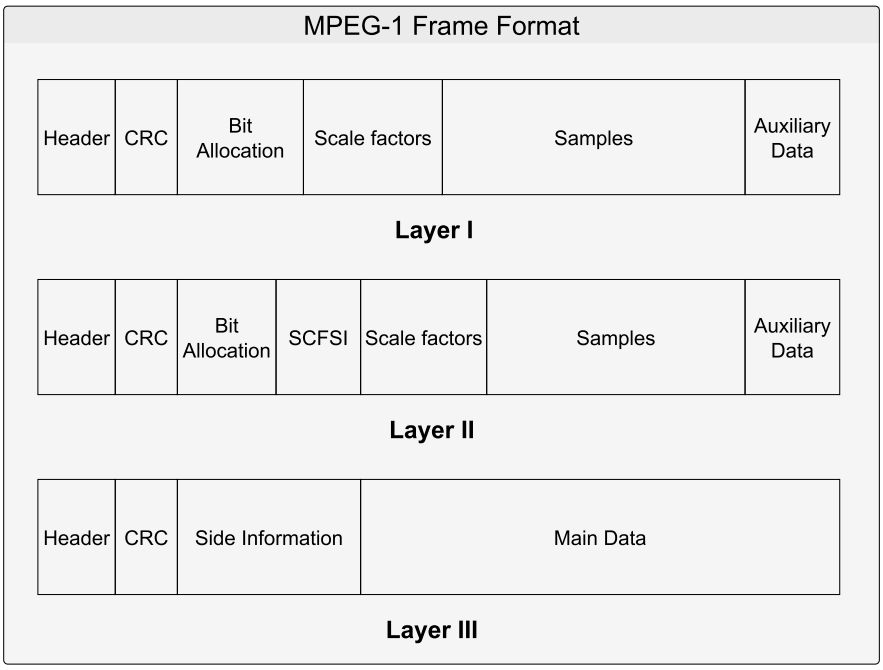

Fig. 2.22. MPEG-1 Format Frames

- **Layer II**.
  This layer is an improvement of Layer I because it uses larger group of samples, obtains better audio quality by employing more bits for the quantized samples allocation and limits the bit allocation for the middle and higher subbands. The frames are formed by 1152 samples per channel corresponding to three groups of 12 samples for each subband. Each of the three 12-samples group has a bit allocation. The scalefactors can be shared between the components of the group, if no audible distortion is produced,

if the scalefactor values for the three groups are very close or if the human auditory masking hides the distortion produced by using a shared scalefactor. This sharing situation is handled with a new frame field called Scale-Factor Selection Information (SCFSI). The encoding process for Layer I and II is shown in Fig. 2.23.

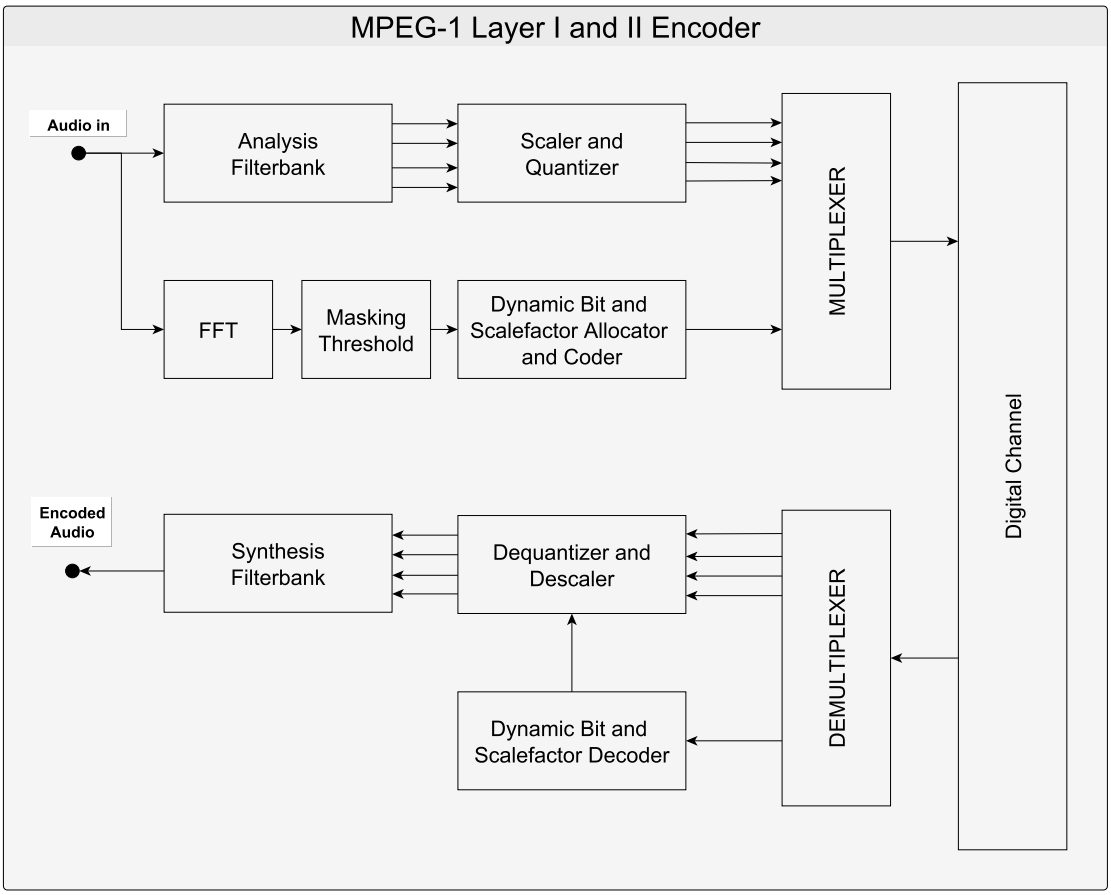

Fig. 2.23. MPEG-1 Layer I and II Encoder

- **Layer III**.
  This layer includes more advanced mechanisms by including Audio Spectral Perceptual Entropy Coding (ASPEC) and Optimal Coding in the Frequency Domain (OCF) algorithms. The effects of the filterbank can be compensated with the utilization of a Modified Discrete Cosine Transform (MDCT). In Layer III, an MDCT transformation is added for every subband after the filterbank processing. This MDCT implementation allows the cancellation of aliasing due to the filterbank, because MDCT subdivides the subband outputs into frequency to obtain better spectral resolution. MDCT provides two different block lengths (long and short) with 50 % overlap. The long block contains 18 samples and the corresponding window length is 36 samples. It is employed to code stationary parts of the signal. The short block contains 6 samples and a window length of 12 samples. It is utilized for transient signals, due to its greater frequency resolution. The number of samples encoded in a frame is the same for both blocks, since a long signal contains the same samples as three short blocks. The MDCT used in Layer III

can have two configurations: the first uses the same block length (short or long) and the second uses short blocks for the 30 upper frequency subbands and long blocks for the two lower frequencies. This configuration allows a better resolution for the low frequencies, where the human ear is more sensibile. Layer III integrates mechanisms to reduce the pre-echo caused by the quantization of the MDCT values.

Layer III also includes:

- Alias reduction. The codec provides methods which eliminate artifacts produced in the filterbank overlapping bands.
- Nonuniform quantization.
- Scale-factor bands. Layer III does not use an unique value to represent the scale-factor. Instead it uses a band of values with a width that approximates the critical-band width. With these bands various MDCT coefficients can be covered.
- Entropy coding of data values. Layer III includes variable-length Huffman codes for the coding of quantized samples.
- A "bit reservation mechanism". The coded information can be less than the assigned number of bits. So in Layer III these bits can be used for the encoding of information from another frame.

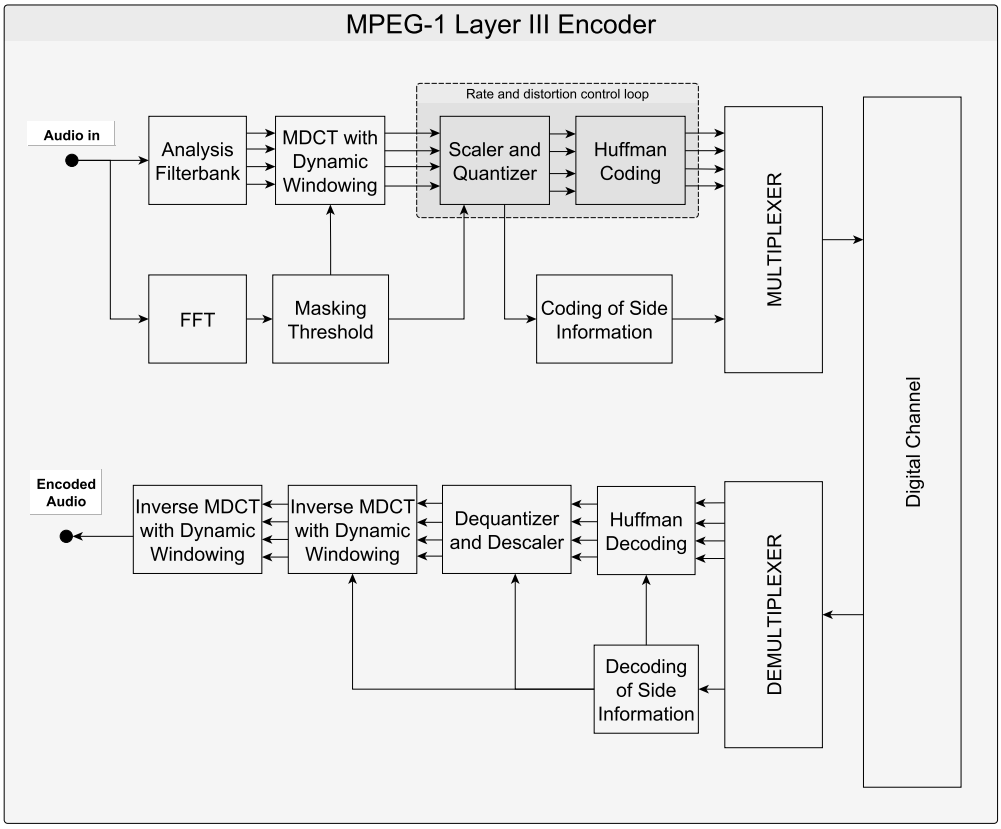

Fig. 2.24. MPEG-1 Layer III Encoder

MPEG-1 Encoder

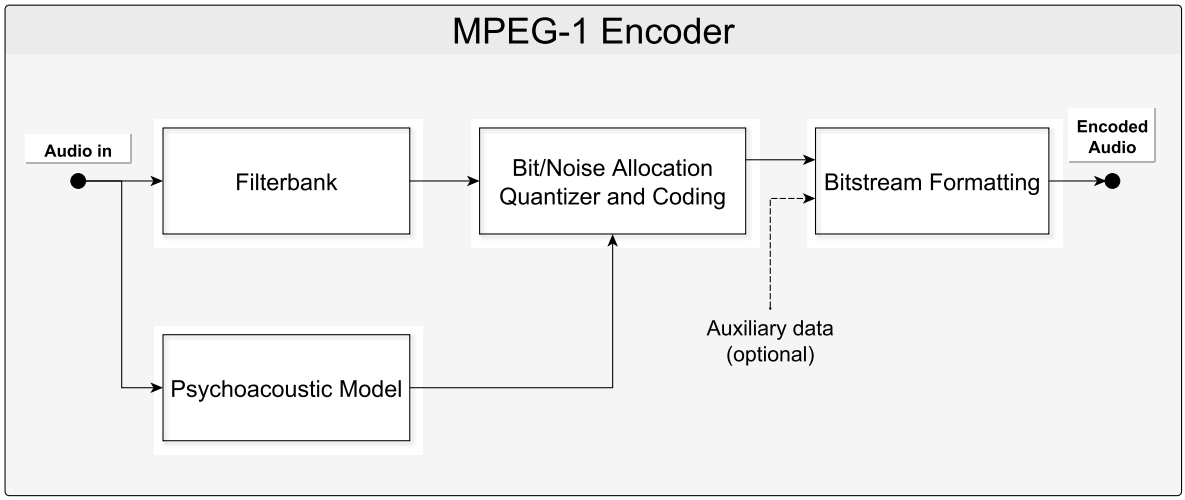

Fig. 2.25. MPEG-1 Encoder

Through the filterbank the input signal is splitted into 32 frequency sub-bands that approximate to critical bands. The masking noise is calculated in the psychoacoustic model block in order to obtain the quantization and coding for the signal. This information is employed in the bit/noise allocation block to obtain the number of bit employed for the quantization. The samples are quantized, coded and allocated within a frame in the last block.

MPEG-1 Decoder

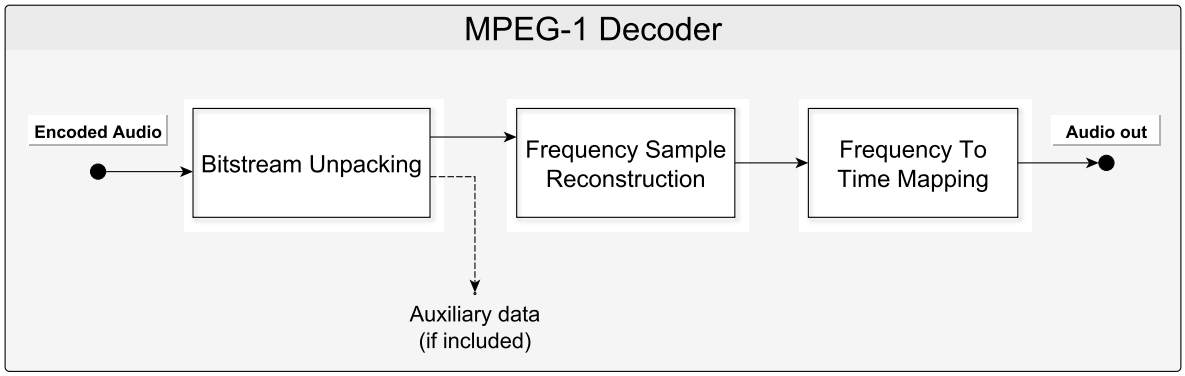

Fig. 2.26. MPEG-1 Decoder

The decoder reads the coded bitstream, recovers the quantized value for the subbands and uses these values to obtain the audio signal.

2.5.2 Error Concealment Approaches

As an error detection method, MPEG-1 provides an optional Cyclic Redundancy Check (CRC). The CRC is controlled by one bit at the beginning of the frame: the "error protection bit". If this bit is set to 0, then error detection is applied and the 16-bits CRC word that follows the header is examined. The same CRC word is utilized for the three layers, but the CRC calculation involves different parts of the frame. The employed detection algorithm is CRC-16 with a generator polynom:

$$x^{16} + x^{15} + x^2 + 1$$

Even though the CRC covers different parts of the encoded bitstream depending on the layer, the two last bits of the header are always included. These header positions contain important information about the encoding process, such as the bitrate index, the sampling rate index or the channel mode. The variable number of bits covered by the CRC belongs to the encoded audio data. A table with the number of protected audio data bits is represented in 2.4.

Layer	Configuration	No bits for single channel	No bits for other modes
Layer I	-	128	256
Layer II	Case 1	142	284
Layer II	Case 2	154	308
Layer II	Case 3	42	84
Layer II	Case 4	62	124
Layer III	-	136	256

Table 2.4. Bits protected by the CRC in MPEG-1

- Case 1 includes:
  - F_s = 48 kHz; Bit rates per channel = 56, 64, 80, 96, 112, 128, 160, 192 kbits/s, and free format.
  - F_s = 44.1 kHz; Bit rates per channel = 56, 64, 80 kbits/s
  - F_s = 32 kHz; Bit rates per channel = 56, 64, 80 kbits/s
- Case 2 includes:
  - F_s = 48 kHz; Bit rates are not relevant
  - F_s = 44.1 kHz; Bitrates per channel = 96, 112, 128, 160, 192 kbits/s and free format.
  - F_s = 32 kHz; Bitrates per channel = 96, 112, 128, 160, 192 kbits/s and free format.
- Case 3 includes:
  - F_s = 48 kHz; Bitrates per channel = 32, 48 kbits/s
  - F_s = 44.1 kHz; Bitrates per channel = 32, 48 kbits/s
  - F_s = 32 kHz; Bit rates are not relevant

- Case 4 includes:
  - F_s = 48 kHz; Bit rates are not relevant
  - F_s = 44.1 kHz; Bit rates are not relevant
  - F_s = 32 kHz; Bitrates per channel = 32, 48 kbits/s

The preceding tables show the different bit positions protected in each layer. The bits included in the CRC calculation for **Layer I** are obtained by employing the following equation:

```
4*[number_channels x bound_intensity_stereo + (32 - bound_intensity_stereo)]
```

For mono encoding, `number_channels`= 1 and `bound_intensity_stereo`= 32. Therefore the number of bits is 128. This number of bits correspond to the bit allocation. For **Layer II**, the calculation includes more variables and so is more complex. For **Layer III**, the protected part corresponds to the side information field, that follows the header. It should be noted, that the quantized samples have no CRC protection. Just the information relative to the encoding is protected.

The CRC-16 method used for the calculation is depicted in Fig. 2.27.

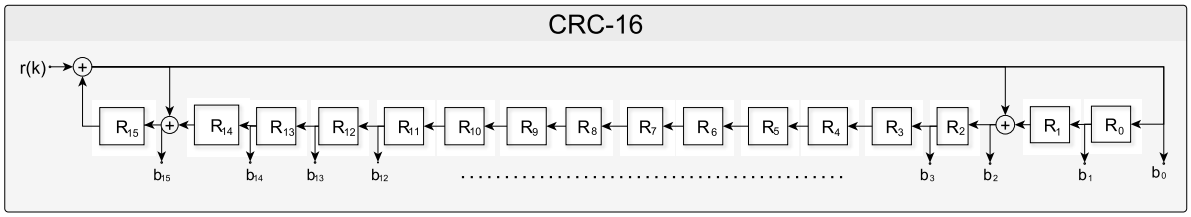

Fig. 2.27. MPEG-1 CRC Mechanisms

In MPEG-1 the CRC value is stored in the decoder. Then, the CRC value that corresponds to the received frame is calculated by using the method above. At the end, the two values are compared. The bits are passed through the circuit and a 16-bits word is obtained as result. This word corresponds to the values of b_0, b_1, ... and b_{15}. The initial states for the registers T are "1111111111111111". The comparison between the stored CRC value and the calculated 16-bit word is performed and if both words are identical, no error is detected at the decoder. If there are differences between the two words, an "error concealment" mechanism can be applied. However, there are no mechanisms defined in the standard documentation, although some techniques, like muting of the actual frame or repetition of the previous frame are recommended.

2.6 AC3

Dolby Laboratories developed the AC3 audio compression technology [Dav99] [HM95] [Ver95]. Originally named "Dolby Digital", is capable of encoding audio into a bit stream with data rates varying from 32 kb/s to 640 kb/s by using sample rates of 32, 44.1, and 48 kHz. AC3 supports various channel configurations like mono, stereo or surround 5.1 format. The last configuration refers to a 5 channel (left, center, right, left surround, right surround) and a ".1"' extension, which includes the low-frequency effects channel (LFE) or subwoofer.
The basis of the compression mechanism is to generate a digital version of the audio signal that uses a minimum number of bits for its representation by employing perceptual coding techniques. AC-3 is the audio compression technology used for High Definition Television (HDTV) in the United States, in direct satellite broadcast and in commercial uses like digital video disc, DVD players, DVB-S, Blueray, DVB-C and is established in home cinemas.

2.6.1 Description

A brief description of the encoding and decoding processes are shown in Fig. 2.28 and Fig. 2.29 respectively.

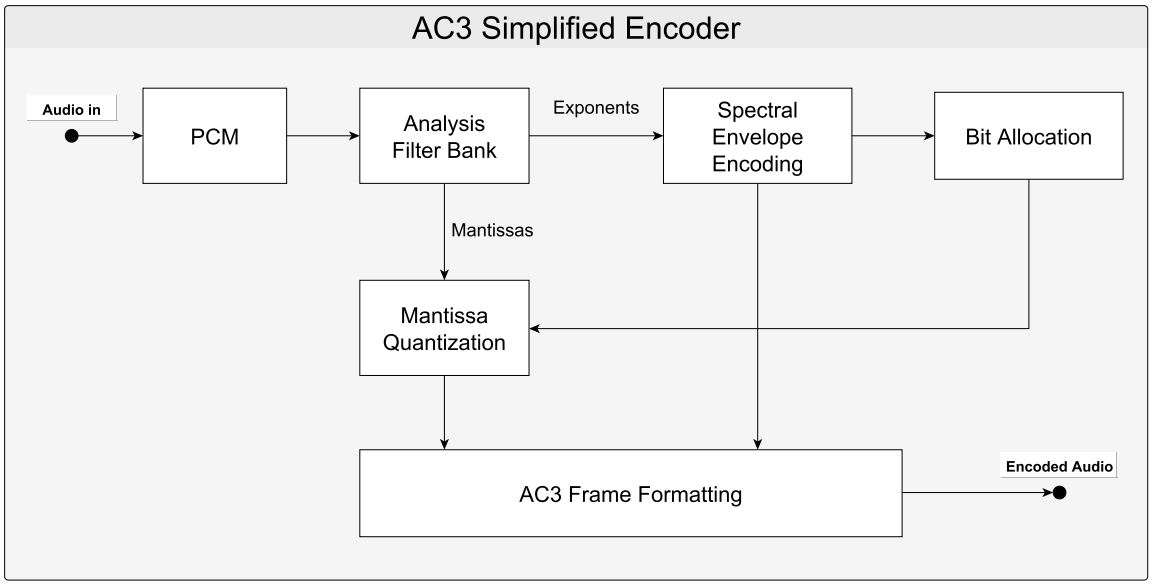

Fig. 2.28. AC3 Simplified Encoder

The most important elements, in terms of AC3 functionality, are: the filterbank block (an analysis one in the encoder and a synthesis one in the decoder), the spectral envelope, the parametric bit allocation and the quantization and coding block.

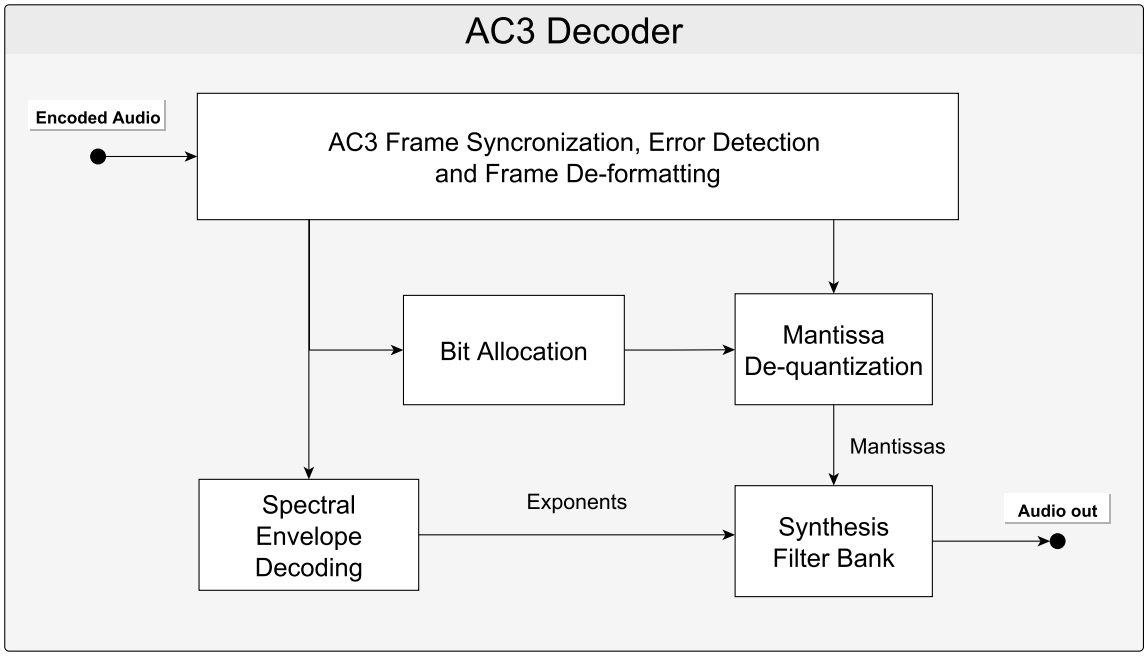

Fig. 2.29. AC3 Simplified Decoder

- **Filterbank design.**

  In the filterbank used in the encoder, a sequence of audio PCM time samples are transformed into a sequence of frequency coefficients. The input of the filterbank is a 512-samples block, which is multiplied by a window function in order to improve the frequency selectivity and reduce the effect of the transforming by resolving the close frequency components. Each audio block is then transformed into the frequency domain using a 512-point Modified Discrete Cosine Transform (MDCT).

  The chosen window is a Kaiser-Bessel window, that satisfies the MDCT window design constraints and follows the equations:

$$w_a(k) = w_s(k) = \sqrt{\frac{\sum_{i=L_1}^{L_2} w(i) \cdot w_r(k-i)}{\sum_{i=0}^{K} w(i)}} \tag{2.48}$$

  where w_a is the analysis window, w_s is the synthesis window, w is the kernel window and K must satisfy $0 \leq K \leq N/2$. For the Kaiser-Bessel window in AC3, a β parameter of 5 is chosen. w_r is a rectangular window, that together with L_1 and L_2 can be calculated according to (2.49).

$$L_1 = \begin{cases} 0 & \text{for } 0 \le k < N-K \\ n-N+K+1 & \text{for } N-K \le k < N \end{cases}$$

$$L_2 = \begin{cases} n & \text{for } 0 \le k < K \\ K & \text{for } K \le k < N \end{cases} \tag{2.49}$$

$$w_r(k) = \begin{cases} 0 & \text{for } 0 \le k < (N/2-K)/2 \text{ and } (3N/2-K)/2 \le k < N-K \\ 1 & \text{for } (N/2-K)/2 \le k < (3N/2-K)/2 \end{cases}$$

In the encoder, each audio block is transformed with the forward transform equation, into the frequency domain using either a long point transform (N=512) or two short point transforms (N=256) according to (2.50). The selection is based on an analysis of the input signal characteristics. For both long blocks and short blocks the same number of transform coefficients are computed. The short blocks are generated by splitting 512 windowed audio samples to form two contiguous sub-blocks of length 256, and each 256-samples sub-block produces 128 unique nonzero transform coefficients that are interleaved together. Similarly, a long block produces 256 coefficients. Both type of blocks are quantized and transmitted identically. An overlap of 50% is achieved with adjacent samples, so each PCM sample appears in two contiguous blocks.

In the decoder an inverse procedure is applied to obtain the blocks of time samples from the blocks of frequency coefficients. The transform equation is defined by

$$X(j) = 1/N \sum_{k=0}^{N-1} z(k) \cdot \cos\left[\frac{2\pi}{N} \cdot (j+1/2) \cdot (k+k_0)\right] \quad \text{for} \quad 0 \le j < N-1 \tag{2.50}$$

With the parameter k_0 being a time offset of the modulator basis vectors used in the transform kernel ($k_0 = 1/2$ for the first short block and otherwise $k_0 = 257/2$), N denoting the audio samples number and $z(k)$ being the windowed input sequence formed by $z(k) = w(k) \cdot s(k)$.

The inverse transformation is defined by

$$X'(k) = 1/N \sum_{j=0}^{N-1} Z(j) \cdot \cos\left[\frac{2\pi}{N} \cdot (j+1/2) \cdot (k+k_0)\right] \quad \text{for} \quad 0 \le k < N-1 \tag{2.51}$$

With Z(k) being the resulting sequence of transform coefficients.

- **Spectral envelope.**

  The term spectral envelope refers to a floating-point representation of the individual frequency coefficients obtained from the filterbank. The coefficients are converted into a binary exponential notation, where they are represented by an exponent and a mantissa, according to $mantissa \times base^{exponent}$. An example of this notation is the representation of the binary number 10010000000. This number can be represented as 1001×2^{111} where the mantissa is 1001 and the exponent is 111 (111 is the binary representation of seven, where seven is the number of zeros in the original binary number).

  The exponents are utilized in AC3 to represent an estimation of the global spectral content. The initial exponent value is the number of leading zeros in the binary representation of each MDCT coefficient. As six transform blocks (the MDCT coefficients corresponding to 1536 input samples) are combined to form a single frame, there are six spectral envelopes in each AC3 frame. These six values represent a signal that varies in frequency and in time. The time variation is represented by the index of each block within the frame and the frequency variation is represented by the exponent. The exponent representation is optimized through "exponent strategies", where the frequency and time variation is examined in order to eliminate temporal redundancy and minimize the difference between costs and benefits. This is made by considering variables like, among others, the available bitrate or the noise-to-mask ratio. Three exponent strategies depending on the bitrate used for each exponent and the frequency resolution are included: D15, D25, or D45. The first mode uses one exponent per each frequency coefficient, providing the highest frequency resolution, while the D45 uses the lowest bit-rate for each coefficient.

- **Parametric bit allocation.**

  The bitstream used in AC3 is constituted by a sequence of synchronization frames. Each frame corresponds to 1536 samples of digital audio, constructed in order to minimize the bit stream length, which can be decoded without using further information. Each frame is independent, shares no data with other frames and contains six "audio blocks", one for each 256-samples block. The blocks begin with a synchronization field (SI), that carries four fields: a syncword corresponding to the hexadecimal value 0B77, a CRC that applies to the first 5/8 of the frame, the sampling rate values and the frame size information. Then follows a Bit Stream Information header (BSI) with the parameters that describe the coded audio service, like the number of input channels, the dynamic compression control word and the program time codes. After these two fields, the six coded Audio Blocks (AB) follow, which represent the 256 quantized audio samples from the input channels. Finally, an auxiliary data field (AUX) and an additional CRC follows. The AUX field can be used to embed additional information not defined within AC3. In each audio block the necessary coding information, like the exponent strategy, the exponents, the bit allocation parameters or the mantissas, is included. This bitstream format is represented in Fig. 2.30.

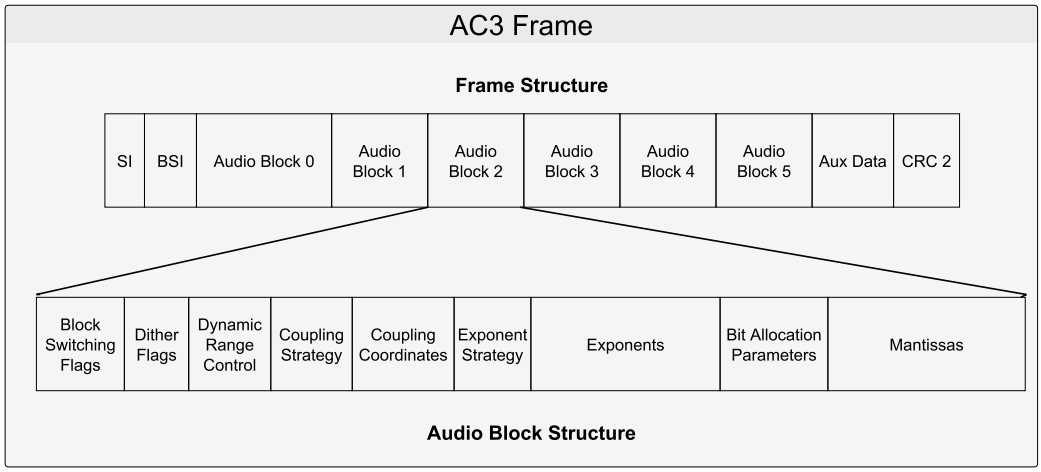

Fig. 2.30. AC3 Frame

Within the bit allocation procedure, the amplitude resolution required to encode each coefficient value corresponding to the mantissa is calculated by using a parametric bit allocation technique. The model is called parametric, since is defined by many parameters that have an influence on the bit assignment. Both encoder and decoder implement this model, therefore there is no need to share additional information. The procedure employs psychoacoustic masking mechanisms to perform the bit allocation decision by estimating the masked threshold on each frame. These mechanisms implement an iterative procedure to determine the best set of parameters according to the perceptual model, which include, among others, the spectrum subdivision into frequency bands and the spectrum samples integration within these bands. Also, it includes the convolution to model masking effects and the obtaining of a predicted masking curve. The procedure keep the specified bitrate limitation and the masking threshold.

- **Quantization and coding.**

The mantissas obtained from the floating-point representation need to be quantized in order to allocate them into the corresponding audio block in the bitstream. The bit allocation pointer (bap) represents the quantization precision of the mantissas. For example, a bap value of 3 represents a quantizer level 7, and the corresponding bits per mantissa are 3. The quantization can group various mantissa values into a common codeword. That is the case of the level 3 quantizer (corresponding to a bap 1), where 3 quantized values are grouped together in a 5-bit codeword. The codec includes the possibility of dithering by using random noise values instead of quantized values for mantissas allocated within 0 bits.

AC3 Encoder

The encoding process, shown in Fig. 2.32, starts by processing the input signal using a PCM modulation to form a sequence of samples. The samples obtained from the previous block are processed within a filterbank, where the PCM sample blocks are transformed to frequency coefficients by using a 512-point MDCT transformation. An overlap with the 256 adjacent samples (overlap of 50%) is achieved so at the end, each PCM sample appears in two contiguous blocks. Following, decimation is applied to obtain a reduction in the number of coefficients within each block from 512 to 256. Each coefficient is converted into a floating point representation, where the binary number is represented by an exponent and a mantissa. The set of exponents from the spectral envelope of the input signal and the mantissas, are quantized in a variable number of bits through a parametrical allocation model with psychoacustical masking. The parameters are sent to the decoder which represent the quantized mantissas and the exponents corresponding to 6 input blocks, as part of a syncronization frame.

AC3 Decoder

As a first step in the decoding process, shown in Fig. 2.33, frame alignment is performed. This step consists of identifying the sync word within the bitstream. Once the synchronization part is found, the CRC should be verified. If the CRC is correct and thus, no error are present, the BSI field is retrieved in order to obtain important information about the frame, like the number of input channels. After the retrieving of the information in BSI, the audio blocks are unpacked and processed to get the encoded spectral envelope and the quantized mantissas. This process is accomplished by computing the bit allocation used in the encoding, by using this information to unpack the mantissas, by reversing the float-point transformation to obtain the frequency coefficients, by calculating the inverse MDCT transformation and finally, by downmixing the samples to form the PCM decoded samples.

2.6.2 Error Concealment Approaches

As error detection techniques, AC3 provides two CRCs that can be used in order to check the presence of errors within the frame. However, the codec does not implement any mechanisms to apply when errors are detected. Those mechanisms are user definable, some possibilities are muting, block repeating, or frame repeating.

CRC Checking

The two CRC words included in a AC3 frame have a length of 16 bits. One is allocated at the begin of the frame (crc1) after the synchronization word and the other at the end of the frame (crc2). Both CRC cover different segments of the frame: crc1 covers the first 5/8, excluding the sync word, and crc2 covers the last 3/8. The codec design ensures that at least two audio blocks are contained within the first 5/8. Just allowing the first CRC to cover this part of the frame, the decoding process can begin without receiving the complete frame, decreasing the coding latency. Once the two first audio blocks are received and decoded, the frame has been received completely and the second CRC can be processed. The polynomial function, which generates both CRCs is:

$$x^{16} + x^{15} + x^2 + 1 \tag{2.52}$$

The function to obtain the part of the frame, where the crc1 is applied is the following:

```
num_bits_crc1 = truncate( frame_size/2) + truncate(frame_size/8)
or
num_bits_crc1 = (int) (frame_size >> 1) + (int) (frame_size >> 3)
```

For the implementation of the CRC calculation there are various approaches. The method included in [HM95] consists of an hardware implementation based on a Linear Feedback Shift Register (LFSR). The LFSR mechanism consists of various shift-registers with a small number of xor gates. Fig. 2.33 represents the LFSR implementation. It should be noted that in AC3 the CRC checking is not performed by comparing the CRC included in the frame with the value generated after the application of LFSR. This is because the CRC calculation in the decoder is performed by taking into account that this value should produce a zero value in the LFSR registers of the decoder.

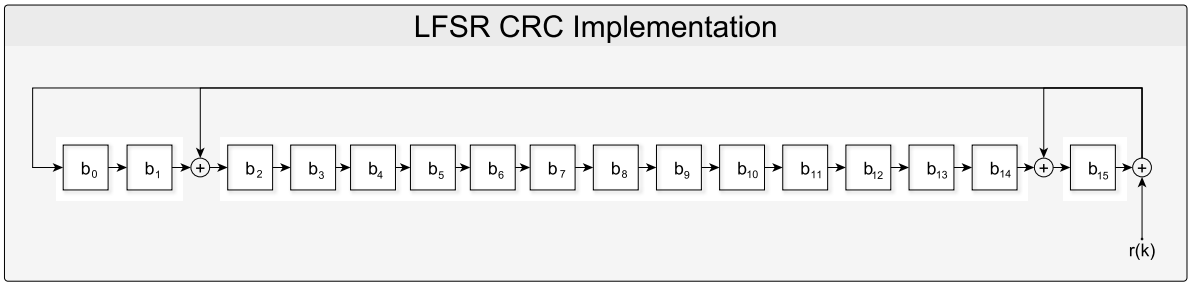

Fig. 2.31. LFSR CRC Implementation

As a first step in the CRC calculation, the CRC block is reseted by setting all the values in the registers to zero. Afterwards, all the bits belonging to the positions of the frame that is necessary to process (the bit positions corresponding to crc1), are passed through the LFSR

implementation sequentially one after the other and in the same order in which they arrived at the decoder. Once all necessary bits have passed through the CRC implementation, if the value contained in all registers is zero, the CRC is valid and the corresponding positions are error free. Various techniques exist for the crc2 calculation. In one of them, the crc1 value is calculated. If that value is valid, then the calculation continues for the following bits to the end of the frame. If after the processing of the last bit the values in the registers are zero again, the crc2 is valid too. If crc1 is invalid an error management mechanism can be applied. In another implementation, if crc1 is invalid the processing of the frame continues in order to evaluate the second part of the frame. The registers corresponding to the first calculation are set to zero to simulate a correct result and then, the calculation of crc2 begins. If crc2 is correct, it implies that the last 3/8 of the frame contains no error. However this implementation seems to be not very useful, because if errors are present in the 5/8 part of the frame, it could be impossible to successfully recover of the original audio. In the first part of the frame the Bit Stream Information (BSI) is allocated with important information about the coded audio service.

Bitstream consistency checking

In the decoding process, the sync word can be correctly identified and the CRC can indicate that the frame is free of errors. However, the audio decoding can be impossible. This case can happen, if the error included in the frame produces a valid CRC or if an error in the encoding process occurs. In order to prevent this situation from happening, analysis of the bitstream and checking tests can be performed. When this type of error appears, the probability of certain illegal syntaxes rises. Therefore by sintax checking many errors can be avoided. A list with some of these error combinations can be found in [HM95]. Some of these errors can cause a system instability and therefore its occurrence should be avoided.

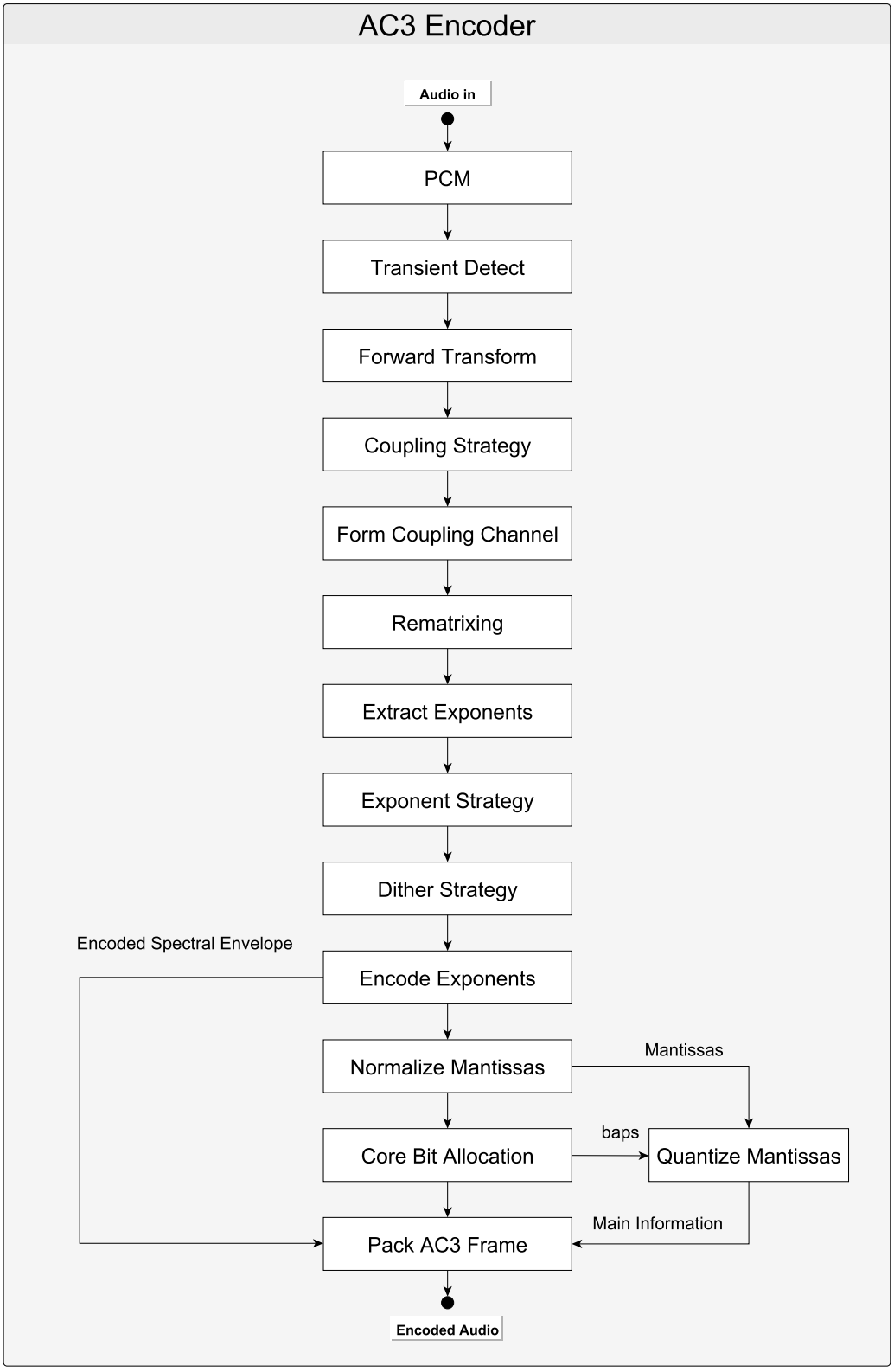

Fig. 2.32. AC3 Encoder

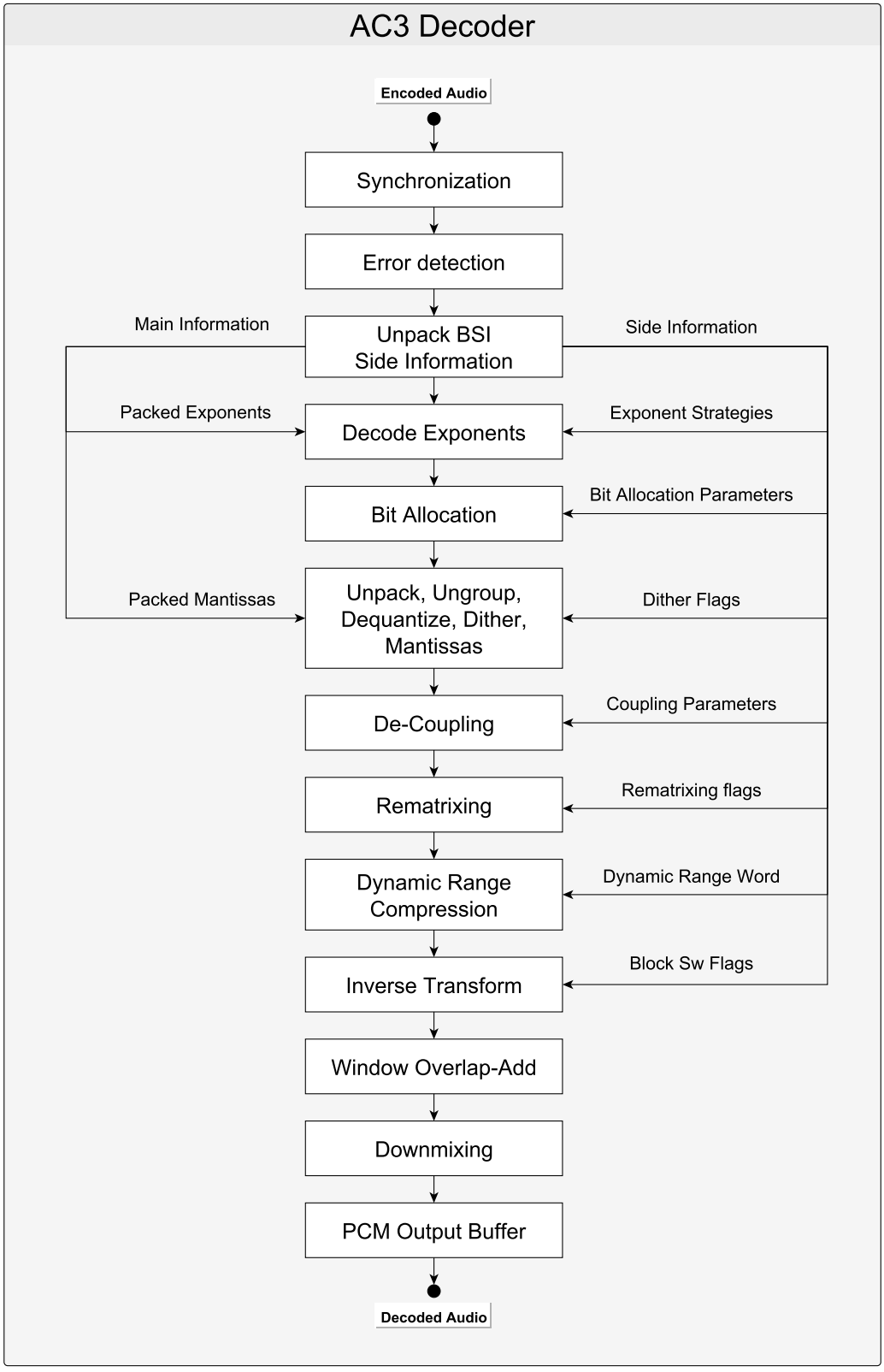

Fig. 2.33. AC3 Decoder

3 Simulation

In order to evaluate the response of one audio codec to bit errors, one Matlab simulation is made. The chosen codec is AMR-WB+ because of its inherent robustness to errors and its good performance for audio and speech.

3.1 AMR-WB+

To understand the results of the simulation is important to know how the bitstream in AMR-WB+ is organized. The chosen format for the encoded files was "Transport Interface Format". The differences between the Transport Interface Format and the 3GPP Format reside in the content of some bits and the position of the frame header. The choice of this format implies the presence of one header per frame, and not once per superframe as in 3GPP format.
In the simulation one set of 13 audio files were used with the following features: mono signals, 16bit PCM, 48 kHz of sampling rate.

The C++ source code [ETS05] used in the simulation is the ANSI-C code for the Floating-point Extended Adaptive Multi-Rate Wideband codec that 3GPP defines in the document. By using the ANSI-C code to generate an executable file with the AMR-WB+ encoding program, 13 encoded audio files were obtained. The MATLAB program simulates the modification of one bit position in each frame of the encoded files. The algorithm performs the modification of all bits in the same frame type. That is, for the first audio encoded file, the first bit of all the ACELP frames was modified and the result was save in a new file. Then, the second bit of all the ACELP frames was modified and the result file saved. That process was repeated for all bits in ACELP frames and then for every TCX frame.

After the decoding of the modified version of each file a comparison is performed with the "PEAQ Eval" tool (Perceptual Evaluation of Audio Quality) [Kab02]. PEAQ Eval is a Matlab implementation for objectively measuring the perceived quality of audio signals. The result is the "mean value for the objective difference grade" (ODG). It indicates the perceived difference between the original audio file and its modified version in the form of a number between 0 and -4. -4 indicates very annoying impairment and 0 imperceptible impairment. The goal of the simulation is to obtain a representative mean ODG value for each bit position within the four AMR-WB+ frame types.

Header	Data
1409	60EC71848793200000000000000000000000000000
	0000000000000000000000000000000000000042AF

Table 3.1. Example of one frame. Hexadecimal values

	MSB							LSB
First Octet	0	0	0	1	0	1	0	0
Second Octet	0	0	0	0	1	0	0	1

Table 3.2. Exemplary AMR-WB+ Header of encoded files in the simulation

One example of a processed frame of the simulation is shown in Table 3.1 and in Table 3.2. By examining the header format seen in AMR-WB+ chapter, the following values are obtained:

- An ISF value of 9.
- A Frame Type value of 20, the frame is encoded in mono mode with 336 octets per frame and a bit rate of 16.8 kbit/s.
- A TFI value of 0, indicates that that frame is the first frame of the superframe.

The bit allocation of the audio frame according to this information is given in Table 3.3. The two first bits show the core coding mode for the frame. In this case the value is 01, therefore the frame is encoded with medium TCX.

PARAMETER	BITS (MSB - LSB)
Mode bits	b0 - b1
First ISP subvector	b2 - b9
Second ISP subvector	b10 - b17
Noise factor	b18 - b20
Global gain	b21 - b27
Split Algebraic VQ	b28 - b319
Index of HF ISP	b320 - b328
Index of HF gain	b329 - b335

Table 3.3. Example of one frame. Hexadecimal values

The Fig. 3.1, Fig. 3.2, Fig. 3.3, Fig. 3.4, Fig. 3.5, Fig. 3.6, Fig. 3.7 and Fig. 3.8 show the results of the experiment for each type of coding mode in AMR-WB+. Along the X axis are the bits contained in the bitstream excluding the header. The Y axis represents the mean values after the PEAQ calculation.

As can be seen in Fig. 3.1, for ACELP the most sensitive bits are located in the begin of the frame. Specifically the first bit is the most sensitive one, because the two first bits represent the coding mode. Errors in that position cause a misinterpretation of the decoding method used for the entire packet. The following bits correspond to the first and second ISP subvector of the quantizer.

Afterwards, four groups of bits appear with higher sensitivity to errors. To these groups belong the values of the “Adaptive Codebooks Index” and the “Mean Energy Index”. Both indices are sent once per every ACELP subframe. In the same figure, the indices of the

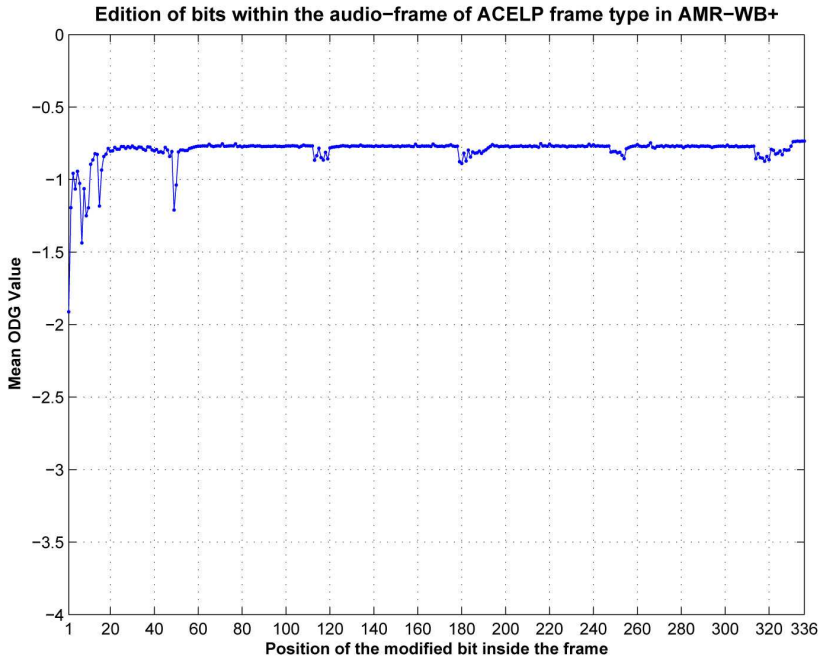

Fig. 3.1. ACELP

Adaptive Codebooks are located from bit 49 to bit 58 for the first group, from bits 112 to bits 124 for the second group, from bit 178 to bit 193 for the third group and from bit 247 to bit 259 for the last group.

All the TCX figures show a similar pattern. There are two areas with high sensitivity to errors: the first one appears in the initial positions of the frame and the second one near the end of the frame. Like in ACELP frames the first two bits correspond to the core coding bits. It should be noted that the bits of core coding mode do not have the same response to errors for all TCX frames. For short TCX and medium TCX, an error in the core coding mode bits is worse than the same error in a long TCX frame. This behavior is caused by its repetition along the four packets of the superframe.

After the core mode indication bits the first ISP subvectors of the LPC quantizer and then the global gain are present.
Fig. 3.2 illustrates this situation. The first and the second ISP subvectors are allocated from bit 2 to bit 17, and the global gain from bit 51 to bit 57. The main part of the bistream corresponds to the Algebraic Vector Quantizer (AVQ) bits. The AVQ bits fill the positions from the bit 58 to the bit 319 in Fig. 3.2.

In Fig. 3.3, Fig. 3.4, Fig. 3.5, Fig. 3.6, Fig. 3.7 and Fig. 3.8, the TCX superframes are divided into several frames. The most significant bits of the first frame have a worse performance than the same positions of the following frames. This is due to the fact that the first packet contains the ISP subvectors, which are more sensitive to errors, and because the global gain value only appears once in the first frame. In the remaining frames, either some bits are repeated or the parity bits are formed with some bits of the global gain. The global

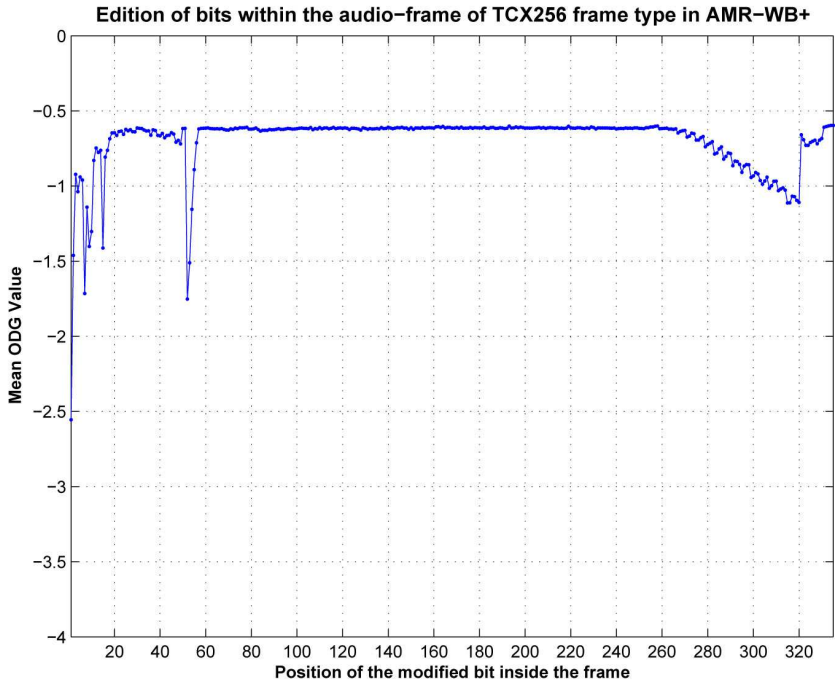

Fig. 3.2. Short TCX

gain is an important parameter in AMR-WB+ because it helps to maintain audible quality, specially in episodes of packet loss and therefore is encoded redundantly.

The second area, with high sensitivity to errors, belongs to the AVQ bitstream. These bits contain the quantization parameters of the spectral information of the input signal.

As has been seen in the AMR-WB+ section, the parameters consist of codebook numbers and codebook indices.
Codebook indices are allocated sequentially from the beginning of the AVQ bits, whereas the bits corresponding to the codebook numbers are allocated sequentially from the end of the "Algebraic Vector Quantization (AVQ)" bits.
In the figures, the effect of the AVQ ordering and the fact that the impact of an error in codebooks numbers has worse consequences than the same error in codebook indices.
Failures in codebook numbers may lead to the suppression or addition of subvectors and to an erroneous interpretation of the codebook indices.

Since the spectrum is splitted into bins starting from the low frequencies and then interleaved in the packets of the superframe, codebook numbers of the lower part of the spectrum are always allocated at the end of the AVQ bitstream. This is the reason why the figures show a descending curve at the end of the frame. Low frequencies contain more energy and thus, the bit errors cause a greater deterioration of the decoded signal quality.

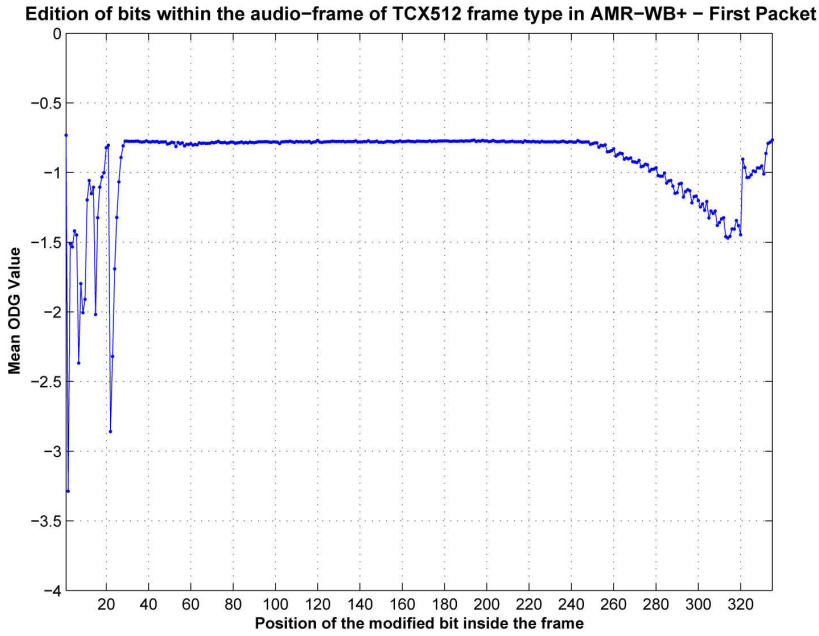

Fig. 3.3. First Packet of Medium TCX

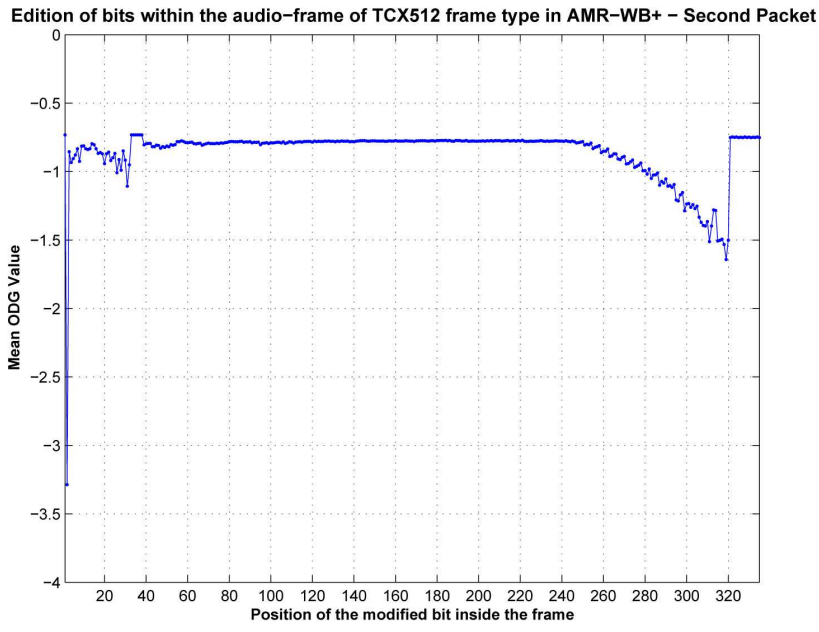

Fig. 3.4. Second Packet of Medium TCX

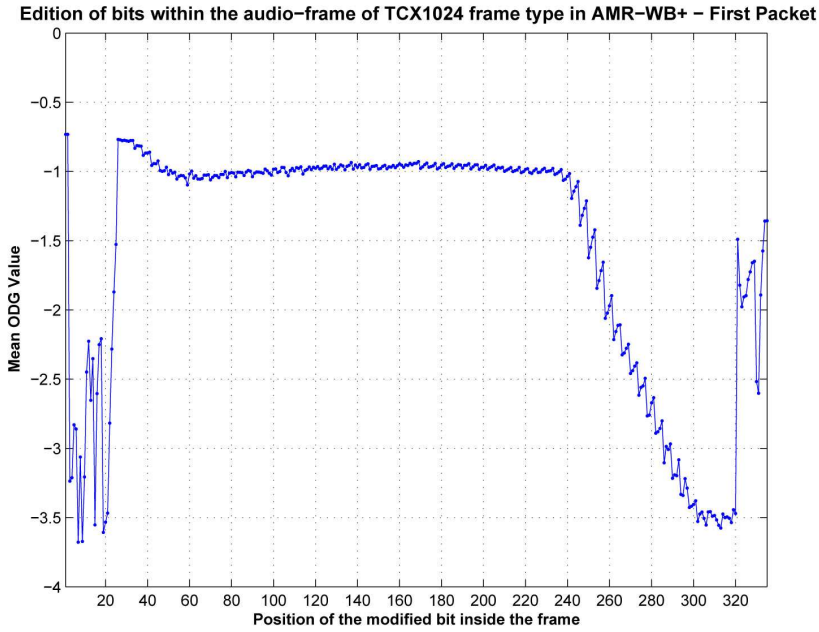

Fig. 3.5. First Packet of Long TCX

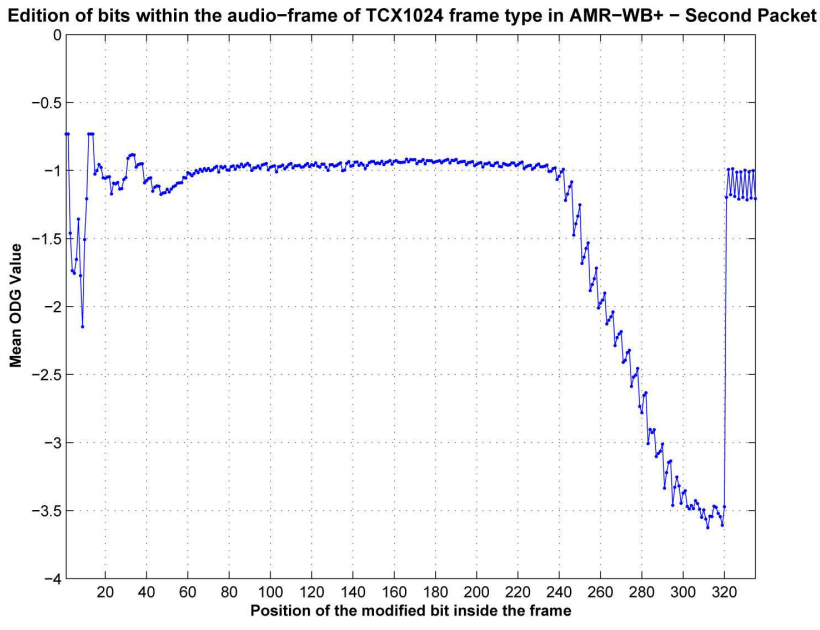

Fig. 3.6. Second Packet of Long TCX

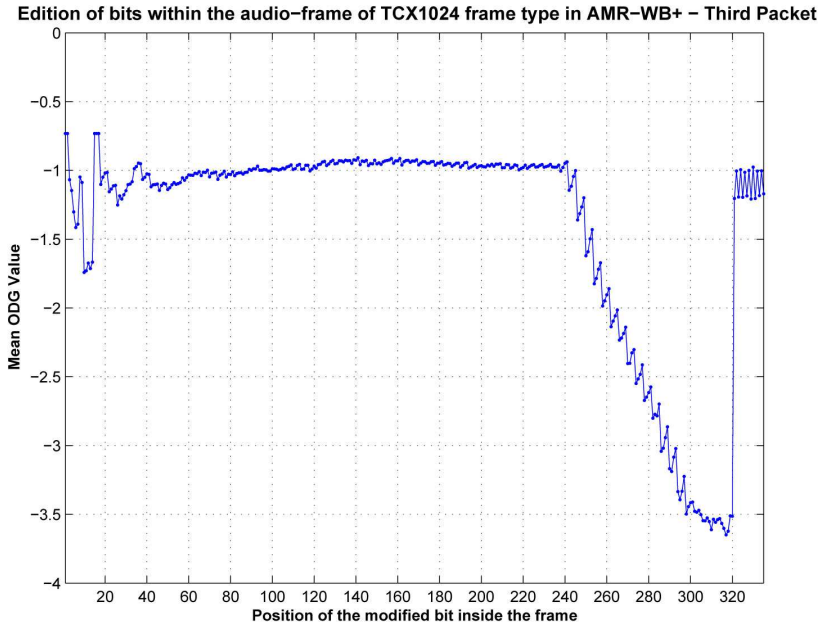

Fig. 3.7. Third Packet of Long TCX

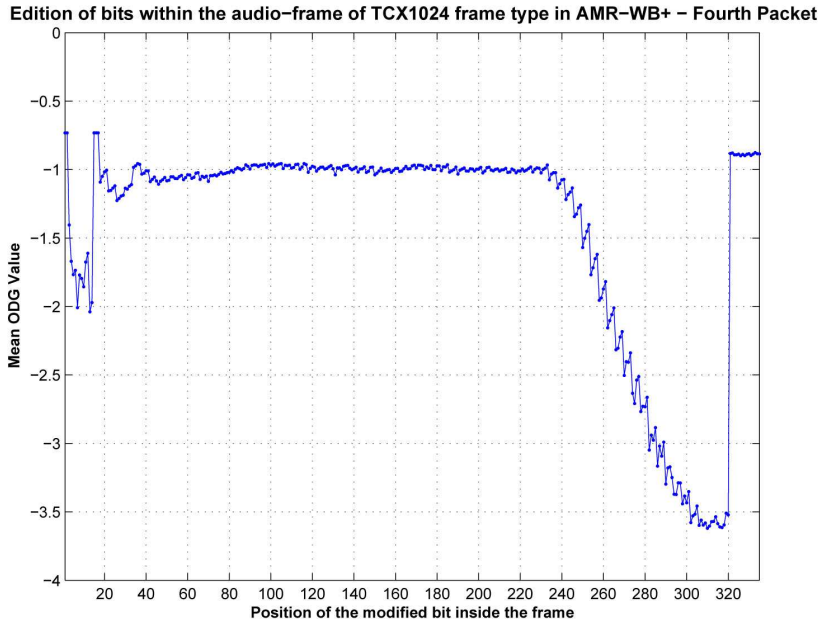

Fig. 3.8. Fourth Packet of Long TCX

4 Conclusion

In this thesis, various techniques of error management used in audio codec implementations are examined. For the "Adaptive Differential Pulse-Code Modulation (ADPCM)" strategies like advanced muting, detection code scheme, packet loss concealment, soft decision mechanisms, click-noise detection or error correcting codes are analyzed. For "Advanced Audio Coding (AAC)" the explained techniques include: the "AAC Error Robustness Tool" which is implemented in the codec, click noise reduction, robust critical data recovery and error resilient schemes. "Audio Processing Technology Apt-x100" is a robust codec, therefore the mechanisms which provides robustness to the codec are considered. Codecs like "Extended Adaptive Multi-Rate Wideband (AMR-WB+)" provide novel approaches to error management, that are studied in this document. Different CRC mechanisms are reviewed for "MPEG-1 Audio Layer II (MP2)", "MPEG-1 Audio Layer III (MP3)" and "Adaptive Transform Coder 3 (AC3)". Finally, one Matlab simulation for the AMR-WB+ codec is made in order to evaluate the response of this codec to bit errors.

Principal Symbols and Abbreviations

Principal Symbols

$\mathbb{N}$	the set of positive integers
$\mathbb{Z}$	the set of all integers
A	amplitude
B	Buffer
M	number of subbands
N	window length
$P(\cdot)$	probability distribution
$P(\cdot \mid \cdot)$	conditional probability distribution
$S(k)$	spectral function
W	window function
X	MDCT coefficients
d	difference signal
d_q	quantized difference signal
k	frequency bin index
m	mixture index
n	discrete time index
$p(\cdot)$	probability density function
$p(\cdot \mid \cdot)$	conditional probability density function
s_e	estimated signal
s_r	reconstructed signal

Abbreviations

AAC	advanced audio coding
ACD	ADPCM code detection
ACELP	adaptive code excited linear prediction
ADPCM	adaptive differential pulse-code modulation
AMR-WB+	Extended Adaptive Multi-Rate Wideband
APT	Audio Processing Technology
ASPEC	audio spectral perceptual entropy coding
AVQ	algebraic vector quantization
BER	bit error rate
BWE	bandwidth extension
CRC	cyclic redundancy check
DAB	Digital Audio Broadcasting
DCT	discrete cosine transform
DCME	digital circuit multiplication equipment
DFT	discrete Fourier transform
DSP	digital signal processing
DTFT	discrete-time Fourier transform
DVD	digital video disc
EC	excitation classification
ECC	error correcting code
ECR	excitation classification refinement
ESS	eight short sequence
FFT	fast Fourier transform
FIR	finite impulse response
GMM	Gaussian Mixture Model
HCR	Huffman codeword reordering
HDTV	High Definition Television
HMM	hidden Markov model
ICS	individual channel stream
IDFT	inverse discrete Fourier transform
IMS	IP Multimedia Subsystem
ISF	internal sampling frequency
ISF	immittance spectral frequency
ISP	immittance spectral pairs
ISPP	interleaved single-pulse permutation
LFE	low-frequency effects
LFSR	linear feedback shift register
LMS	least mean squares
LP	linear predictor
LPC	linear predictive coding
LSTS	long start sequence
LSPS	long stop sequence
LTP	long-term prediction
MAD	median absolute deviation

MAP	maximum a posteriori
MBMS	Multimedia Broadcast/Multicast Service
MDCT	modified discrete cosine transform
MLS	minimum least squares
MMS	multimedia messaging service
MMSE	minimum mean-square error
MPEG	Moving Picture Experts Group
OCF	optimal coding in the frequency domain
OLA	overlap-add
OLS	only long sequence
PCM	pulse code modulation
PCW	priority codewords
PDF	probability density function
PS	parametric stereo
PSD	power spectral density
PSS	packet-switched streaming service
QMF	quadrature mirror filters
RMS	root-mean-square
RVLC	reversible variable length coding
SLF	standard lattice format
S-MSVQ	split-multistage vector quantization
SBR	spectral band replication
SNR	signal-to-noise ratio
STFT	short-time Fourier transform
STSA	short-time spectral amplitude
TCX	transform coded excitation
TFI	transport frame index
TNS	temporal noise shaping
TwinVQ	transform-domain weighted interleave VQ
VQ	vector quantizer
3GPP	third generation partnership project

Bibliography

[CBD+11] Hao Chen, Changchung Bao, Feng Deng, Dawei Zhang, and Maoshen Jia, *A MDCT-based Click Noise Reduction Method for MPEG-4 AAC Codec*, Tech. report, IEEE, 2011.

[Dav99] Grant A. Davidson, *Digital Audio Coding: Dolby AC-3*, Tech. report, Dolby Laboratories, Inc., 1999.

[EG77] D. Esteban and C. Galand, *Application of Quadrature Mirror Filters to split Band Voice Coding Schemes*, Tech. report, IBM Laboratory, 1977.

[ETS05] ETSI, *3GPP TS 26.304: ANSI-C code for the floating point Extended AMR Wideband codec*, Tech. report, ETSI, 2005.

[ETS11] ———, *3GPP TS 26.290: Audio codec processing functions; Extended AMR Wideband codec; Transcoding functions*, Tech. report, ETSI, 2011.

[För01] Hartmut Förster, *Technische Beschreibung: apt-x100 Coding-System*, Tech. report, Pro-Audio Systems, 1999/2001.

[FV01] Tim Fingscheidt and Peter Vary, *Softbit Speech Decoding: A New Approach to Error Concealment*, Tech. report, IEEE, 2001.

[HHK+10] Ruijing Hu, Xucen Huang, Michel Kieffer, Olivier Derrien, and Pierre Duhamel, *Robust Critical Data Recovery for MPEG-4 AAC Encoded Bitstreams*, Tech. report, IEEE, 2010.

[HM95] Dr. Robert Hopkins and James C. McKinney, *Digital Audio Compression Standard (AC3)*, Tech. report, Advanced Television Systems Commitee, 1995.

[IEE06] IEEE, *Extended AMR-WB for High-Quality Audio on Mobile Devices*, Tech. report, IEEE, 2006.

[ISO93] ISO/IEC, *11172-3: Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 3: Audio*, Tech. report, ISO/IEC, 1993.

[ISO04] ———, *Information technology - Generic coding of moving pictures and associated audio information. Part 7: Advanced Audio Coding (AAC)*, Tech. report, ISO/IEC, 2004.

[ISO05] ———, *Information technology - Coding of audio-visual objects - Part 3: Audio*, Tech. report, ISO/IEC, 2005.

[IT88] ITU-T, *ITU-T Recommendation G.722*, Tech. report, ITU-T, 1988.

[IT90] _____, *Recommendation G.726: 40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)*, Tech. report, ITU-T, 1990.

[IT03] _____, *Wideband coding of speech at around 16 kbit/s using Adaptive Multi-Rate Wideband (AMR-WB)*, Tech. report, ITU-T, 2003.

[Joh80] J. D. Johnston, *A Filter Family Designed for Use in Quadrature Mirror Filter Banks*, Tech. report, IEEE, 1980.

[Kab02] Peter Kabal, *An Examination and Interpretation of ITU-R BS.1387: Perceptual Evaluation of Audio Quality*, TSP Lab Technical Report, Dept. Electrical & Comp. Eng., McGill University, Montreal, QC, Canada, May 2002.

[Kor02] Jari Korhonen, *Error Robustness Scheme for Perceptualy Coded Audio based on Interframe Shuffling of samples*, Tech. report, IEEE, 2002.

[KPHH86] Ryuji Kohno, Subbarayan Pasupathy, HidekiImaiit, and Mitsutoshi Hatori, *A Robust ADPCM System Using an Error-Correcting Code*, Tech. report, IEEE, 1986.

[KW03] Jari Korhonen and Ye Wang, *Schemes for Error Resilient Streaming of Preceptually Coded Audio*, Tech. report, IEEE, 2003.

[MBB+05] Jari Mäkinen, Bruno Bessette, Stefan Bruhn, Pasi Ojala, Redwan Salami, and Anisse Taleb, *AMR-WB+: A New Audio Coding Standard for 3rd Generation Mobile Audio Services*, Tech. report, IEEE, 2005.

[NDSK93] Osamu Nakamura, Akira Dobashi, Kazuhiko Seki, and Shuji Kubata Shuzo Kato, *Improved ADPCM Voice Transmission for TDMA-TDD Systems*, Tech. report, IEEE, 1993.

[Pan95] Davis Pan, *A Tutorial on MPEG/Audio Compression*, Tech. report, IEEE, 1995.

[Shl94] Seymour Shlien, *Guide to MPEG-I Audio Standard*, Tech. report, IEEE, 1994.

[SK95] Masanobu Suzuki and Shuji Kubota, *A Voice Transmission Quality Improvement Scheme for Personal Communication Systems -Super Mute Scheme-*, Tech. report, IEEE, 1995.

[SK98] _____, *Prompt Estimation of Channel Quality for Personal Communication Systems* , Tech. report, Electronics and Communications in Japan, 1998.

[SN02] Masahiro Serizawa and Yoshiaki Nozawa, *A Packet Loss Concealment Method using Pitch Waveform Repetition and Internal State Update on the Decoded Speech for the Sub-band ADPCM Wideband Speech Codec*, Tech. report, IEEE, 2002.

[SS96] Stephen Smyth and Mike Smyth, *Apt-x100: A Low-Delay, Low Bit-Rate, Sub-Band ADPCM Audio Coder for Broadcasting*, Tech. report, AES, 1996.

[Ver95] Steve Vernon, *Design and Implementation of AC3 Coders*, Tech. report, IEEE, 1995.

[XA96] Minjie Xie and Jean-Pierre Adoul, *Embedded Algebraic Vector Quantizers (EAVQ) with Application to Wideband Speech Coding*, Tech. report, IEEE, 1996.

Declaration

I certify that the work presented here is, to the best of my knowledge and belief, original and the result of my own investigations, except as acknowledged. Any uses made within it of the works of any other author are properly acknowledged at their point of use....

Hiermit versichere ich, die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt zu haben. Die verwendete Literatur und sonstige Hilfsmittel sind vollständig angegeben.

Aránzazu Robles Moya

Braunschweig, Juni 2012