Uwe Gbur*, Markus Werner**, and Martin Dietz*
*Fraunhofer Institut Integrierte Schaltungen IIS, Erlangen, Germany
**M. Werner Software Development, Karlsruhe, Germany

Presented at
the 101st Convention
1996 November 8–11
Los Angeles, California

AES

AN AUDIO ENGINEERING SOCIETY PREPRINT

Realtime Implementation of an ISO/MPEG Layer 3 Encoder on Pentium PCs

Uwe Gbur, Markus Werner** and Martin Dietz**
** Fraunhofer Institut Integrierte Schaltungen IIS, Erlangen, Germany*
*** M.Werner Software Development, Karlsruhe, Germany*

Abstract

During the last five years the well known ISO/MPEG standards for compression of high quality audio have been established. Within ISO/MPEG characteristically the decoder is rather simple compared to the encoder. Whereas realtime software decoders are already available, realtime software encoders can hardly be found. This paper describes the software only implementation of an ISO/MPEG Layer 3 encoder for Pentium PCs running in realtime under Windows and describes the various applications for this kind of software.

1 Introduction

Today, a rapidly growing number of applications is using the audio compression algorithms defined in the international standards ISO/IEC 11172-3 and ISO/IEC 13818-3 ([3], [4]). One concept of the ISO/MPEG audio coding schemes is an asymmetric performance distribution between encoder and decoder. In order to achieve cheap decoders, they are kept as simple as possible. The rather time consuming task to calculate the psychoacoustic model and the bit allocation are up to the encoder. Therefore, the encoding according to ISO/MPEG is approximately five times more complex than the decoding.
A realtime Layer 3 decoder software exists already for PC, Macintosh, SUN, SGI, Next, linux, HP. A portation to different platforms could be achieved easily. On a Pentium CPU the windows decoder supports all standardized sampling- and bitrates. With the MASC3503C from ITT Layer 3 single chip decoders are also available (see [6]).
Up to now realtime encoding required hardware solutions. However, regarding the fact that the performance of modern Pentium PCs is comparable or even higher than the performance of Digital Signal Processors, the implementation of realtime encoders become possible. Since a sound board is already a standard part of a modern PC extra hardware is not required.
Amongst the standardized audio coding schemes, ISO/MPEG Layer 3 is known to be the most efficient algorithm. For high quality audio compression as well as mid and low quality compression Layer 3 was proven to offer the best quality at a given bitrate (see [11]).

This paper describes the implementation of a realtime ISO/MPEG Layer 3 software encoder running on a Pentium PC. A realtime encoder on a Pentium requires an optimized code. After a brief explanation of the Layer 3 algorithm the different steps of optimization are shown and explained.
The paper describes the use of different optimization possibilities in a Layer 3 encoder and demonstrates their use in a software only realtime encoder for Pentium PCs.
In the last part some possible applications are described.

2 Overview and Concept of the Layer 3 Audio Coding Algorithm

For the analysis of the computation complexity of the Layer 3 encoding algorithm it is necessary to understand its algorithmic principles. They are described in this paragraph.

Layer 3 is the algorithm of the hierarchical MPEG Layers providing the highest compression rate of all schemes of the MPEG family. Audio coding of audio signals with sampling rates from 32 to 48 kHz and bitrates from 32 to 448 kbit/s is defined in the first MPEG standard ISO/IEC 11172-3 (MPEG 1) [3]. This standard, mainly designed for high quality coding, is completed with the standard ISO/IEC 13818-3 [4], the so-called MPEG 2 standard. The „low sampling frequency extension" of this standard is aiming at very low bitrates with reduced audio quality, using sampling rates of 16, 22.05 and 24 kHz. This allows audio signals with bitrates down to 8 kbit/s, that can even be transmitted via a slow modem connection.
MPEG 1 and MPEG 2 Layer 3 algorithms are quite similar and differ only in details. Both are already described in many publications ([1], [2], [5], [7]).

Layer 3 Encoder

The basic block diagram of the Layer 3 encoder is given in figure 1.

A Layer 3 Encoder mainly consists of four modules:

- the psychoacoustic model,
- the hybrid filterbank,
- the quantization and coding kernel and
- the bitstream packing module.

Psychoacoustic model

This program part calculates the allowed distortion that has to be fulfilled by the subsequent quantization and coding module.
A block of incoming time domain values is windowed and transformed in the frequency domain using a 512 line FFT. Since masking of tonal and noiselike components of a signal is different, an estimation of the tonality is necessary. One method to determine the tonality is a polynomial predictor. An optimization of this rather time consuming task will be explained in paragraph 3.
After a combination of the frequency lines according to the critical bands of the human auditory system this representation is convoluted with the cochlea spreading function. The allowed distortion is estimated using tonality and the convoluted spectrum and by applying the threshold in quiet. With the so-called „pre-echo control", time domain artefacts caused by the limited time resolution of the system are recognized and reduced by holding the threshold on a low level if a sudden increase of the signal energy occurs. If a bit estimate („perceptual entropy" as introduced in [11]) exceeds a certain value the time resolution is enhanced by switching to a so-called short window. The thresholds for short blocks must be calculated based on three FFTs of length 128. Finally the thresholds (for long or short windows) have to be combined to the so-called „scalefactor band" representation which is used in the quantization part of Layer 3.

Hybrid Filterbank

The block diagram in figure 3 describes the structure of the Layer 3 hybrid filterbank. Time domain values are splitted into 32 equally spaced subbands by a polyphase filterbank.
The output of each subband of the polyphase filterbank serves as input of a MDCT of length 18, finally leading to 32 * 18 = 576 frequency lines. Fast algorithms are existing for the polyphase filterbank and the MDCT.

Quantization and Coding Kernel

The Layer 3 algorithm uses a nonuniform quantization. In a next step the data rate of the quantized values is reduced by entropy coding, removing most of the signal redundancy. The all-zero upper part of the spectrum is run length coded. The redundancy of all remaining quantized data is reduced by Huffman coding. Data with an absolute value below two use a four-dimensional Huffman code. All other remaining „big values" are split into 3 subregions and are coded with a two-dimensional Huffman code. For each subregion a specific Huffman table can be selected. Amongst multiple tables the one with the lowest demand of bits is choosen.

The coding kernel uses the principle of „analysis-by-synthesis" based on two nested loops. The so-called „inner loop" or „rate-loop" changes the quantization step size until the allowed amount of bits for this frame is reached. It is called by the „outer loop" or „distortion loop". This loop calculates the actual quantization error and compares it to the masking thresholds from the psychoacoustic module. A quantization error above the threshold in a certain scalefactor band is reduced by amplifying the spectral values in this band with a weighting factor, the so-called „scale factor". After this „noise coloring", the inner loop needs to be called again. This iteration process is continued until the masking conditions are fulfilled in each scalefactor band, or certain amplification/time limits are reached.
Since this module costs most of the available computation time, an optimization of parts of this module has been necessary for a realtime encoder software (see 3.2.1 and 3.2.3).

Bitstream Packing Module

Finally the quantized data, the header and side information are multiplexed to a bitstream. Examples for header information are the sampling frequency or the bitrate. Windowtype and the scalefactors are examples of a side information. This module is only called once per frame and therefore not very time consuming.

Layer 3 Decoder

Figure 2 shows the block diagram of a Layer 3 decoder as described in the standard ISO/IEC 11172-3 [3]. The decoder unpacks the bitstream data received from the encoder, carries out the reconstruction, followed by an inverse mapping. As this paper focuses on the encoder, the decoder is not described in more detail here (see [5] for more information).

3 Optimization

There are several ways to optimize a software code:

1) Usage of faster algorithms
2) Optimization of the used algorithms
   2a) platform independent
   2b) platform dependent
3) Change of the strategy, other algorithms

The standard only describes the bitstream format and the decoding process, not the encoder. Therefore, some parts of the encoder which must match the decoder, like the hybrid filterbank, are mandatory and can only be influenced by optimizations of type 1 and 2. Hereby type 2b should be used as less as possible to keep the software as platform independent as possible. Parts which are only in the encoder, like the psychoacoustic model, can be optimized using all 3 choices. Of course optimization by change of the algorithm has to be used very carefully in order not to decrease the audio quality.

Starting-point of the encoder optimization

The software only real-time encoder for Pentium PCs is based on a simulation Layer 3 encoder used for simulation purposes and development only. Therefore this code offers full flexibility and does not take PC realtime requirements into consideration. On various Pentium Processors, this code achieves the following performance coding PCM-Files of 10 seconds duration:

	Pentium 100 MHz	Pentium 120MHz	Pentium 133 MHz
MPEG2 16 kHz mono	11.5 sec	9.8 sec	8.7 sec
MPEG2 16 kHz stereo	22.9 sec	19.1 sec	17.1 sec
MPEG2 22.05 kHz mono	16 sec	13.5 sec	12.1 sec
MPEG2 22.05 kHz stereo	31.7 sec	26.4 sec	23.6 sec
MPEG1 44.10 kHz mono	31.5 sec	26 sec	23.4 sec
MPEG1 44.10 kHz stereo	59.7 sec	49,1 sec	44.6 sec

Figure 4 shows the distribution of the calculation time over the basic Layer 3 encoder modules.
Using this code real time encoding with satisfying audio bandwidth is not possible. Even on a Pentium 133 MHz processor only monaural audio signals of a sampling rate of 16 kHz can be encoded in real-time.
The primary optimization goal was to reduce the encoding time about a factor of four. Encoding of stereo 32 kHz input signals in real-time is possible on a Pentium 133 MHz processor then. In a first step the performance of the hybrid filterbank has been enhanced significantly using faster algorithms.

Besides, all other modules have been speeded up. Some examples of the optimization work are summarized as follows.

Optimization examples

Parallel Huffman Bit counter

Layer 3 is the only MPEG-Layer using Entropy Coding for audio signals. It provides 32 different Huffman tables. For every subregion of the „big value" spectrum part an extra table can be selected. Which table is to be used depends on the one hand on the maximum quantized value in the respective subregion. On the other hand there are up to three possible different tables for each length, representing different signal statistics. The number of bits for each possible table has to be calculated. Among all possible tables, the table coding a subregion with the minimum amount of bits is selected. Since counting these Bits is located in the so-called „inner loop" of the coding kernel, this process is repeated quite often.

Instead of counting the Bits for each table separately a better performance can be achieved using parallel Bit counting. On a 32 Bit processor like the Intel Pentium this is possible because the Bit counting result for each table has a limited range. The following code fragment shows this technique for Huffman tables 2 and 3:

```
#define HD(a,b) ((a<<16)+b)
/*
     Table contains bit usage for huffman tables 2 and 3.
*/
static unsigned long len_2_3_tab[3][3] =
{
   { HD(1,2),HD(4,3),HD(7,7)},
   { HD(4,4),HD(5,4),HD(7,7)},
   { HD(6,6),HD(7,7),HD(8,8)}
};

/*
  Bit counting function for huffman tables 2 and 3.
*/
void countFunc_2_3(unsigned short *quant_ptr,int no_of_pairs,int
*bitsum)
{
  int i;
  unsigned long bit_cnt=0;

  for(i=0;i<no_of_pairs;i++)
    bit_cnt+=len_2_3_tab[quant_ptr[i*2]][quant_ptr[i*2+1]];

  bitsum[0] = (bit_cnt>>16);
  bitsum[1] = (bit_cnt & 0xffff);
}
```


Using this technique, a performance increase of 2 or 3 (in the case of 3 possible huffman tables like tables 7,8 and 9) for the Bit counting process is possible, since only half (or third) of the calls of the counting functions are necessary.

Square root based unpredictability measure calculation

One part in the threshold calculation process of the psychoacoustic model is to determine the unpredictability measure c_ω for each spectral line from the complex FFT spectrum of the input signal. This unpredictability measure is needed to calculate the tonality. The suggested method in the ISO/IEC 11172 standard [3] to perform this calculation contains 3 major steps:

1. Calculate the polar representation of the FFT transform S.

   $$r_\omega = \sqrt{re(S_i)re(S_i) + im(S_i)im(S_i)}$$

   $$f_\omega = \arctan\left(\frac{im(S_i)}{re(S_i)}\right)$$

2. Calculate a predicted magnitude $\hat{r}_\omega$ and phase $\hat{f}_\omega$.

   $$\hat{r}_\omega = 2r_{\omega(t-1)} - r_{\omega(t-2)}$$

   $$\hat{f}_\omega = 2f_{\omega(t-1)} - f_{\omega(t-2)}$$

3. Calculate the unpredictability measure c_ω

   $$c_\omega = \frac{\sqrt{(r_\omega \cos f\omega - \hat{r}_w \cos \hat{f}_w)^2 + (r_\omega \sin f\omega - \hat{r}_w \sin \hat{f}_w)^2}}{r_\omega + abs(\hat{r}_w)}$$

Overall, 5 trigonometric functions (1 arcus tangent, 2 sine, 2 cosine) have to be calculated per spectral line. On a Pentium Processor, trigonometric functions are relatively slow compared to basic floating point operations like multiplication and addition [8] [9]. Even divisions and square roots are less time consuming.
The following table gives an overview of typical processor cycles of mathematic operations on a Pentium CPU(see [8]).

Mathematic operation	Processor clock cycles
Addition/Multiplication	1
Division	39
Square root	70
Arcus tangent	173 (worst case)
Sine/Cosine	124 (worst case)

Taking basic vector arithmetic into account the formulas mentioned above can be modified in a way so that trigonometric functions are not needed any longer.
Figure 5 shows the basic strategy behind this approach which is described as follows:

1. Calculate a temporary vector v'_ω.

$$v'_\omega = v_{\omega(t-2)} + 2\left(v_{\omega(t-1)} \frac{v_{\omega(t-2)} \bullet v_{\omega(t-1)}}{v_{\omega(t-1)} \bullet v_{\omega(t-1)}} - v_{\omega(t-2)}\right)$$

$$\left|v'_\omega\right| = \left|v_{\omega(t-2)}\right|$$

(v_ω represents here the two dimensional vector representation of the complex spectrum line S_i respectively)

2. Calculate a predicted vector $\hat{v}_\omega$

$$\hat{v}_\omega = v'_\omega - 2\frac{\left|v_{\omega(t-2)}\right| - \left|v_{\omega(t-1)}\right|}{\left|v_{\omega(t-2)}\right|} v'_\omega$$

$$\left|\hat{v}_\omega\right| = 2\left|v_{\omega(t-1)}\right| - \left|v_{\omega(t-2)}\right|$$

3. Calculate the unpredictability measure c_ω

$$c_\omega = \frac{\sqrt{(v_\omega - \hat{v}_\omega) \bullet (v_\omega - \hat{v}_\omega)}}{|v_\omega| + |\hat{v}_\omega|}$$

By avoiding trigonometric functions this calculation scheme runs significantly faster (about 2.5 times) on Intel Pentium processors than the method suggested in the standard. Only 2 square roots and 2 divisions per spectral line are necessary to calculate the chaos measure in an efficient implementation of this scheme.

Other optimization approaches reduce the complexity of the psychoacoustic model e.g. by using simple look-up tables [12]. The optimization mentioned here neither simplifies nor reduces the capabilities of the psychoacoustic module itself.

Use of a threshold simulator instead of iteration loops

As it has already beendescribed the suggested method in the ISO/IEC 11172 standard [3] to code the spectral data is to perform the so-called „iteration loops". The loops module quantizes an input vector of spectral data in an iterative process according to several demands. The inner loop quantizes the input vector and increases the quantizer step size until the output vector can be coded with the available number of Bits. After completion of the inner loop the distortion in each scalefactor band is calculated and compared with the allowed threshold in this band. A scalefactor band is amplified if the allowed distortion is exceeded. The inner loop is called again.
Typically, 10-16 calls of the inner loop are necessary to achieve a satisfying coding result. Therefore, the spectral data has to be quantized very often (about 6 times in every inner loop call)
A quicker algorithm which can be used for the Pentium real-time encoder follows a somewhat different aproach. It is based on a threshold simulator algorithm and contains the following three steps:

1. Determination of the scalefactor amplification coefficients for each scalefactor band to meet the psychoacoustic requirements. This process can be speeded up by using a binary dividing algorithm. So for each scalefactor band a maximum of 8 iterations is needed. Note that in this step no Bit counting is necessary and a optimized form of quantization can be used, since the quantization results must not be stored externally; they can be held in an internal processor register.

2. Mapping of the scalefactor amplification coefficients of step 1 to a scalefactor scheme supported by the Layer 3 bitstream format. This is necessary because the range and the granularity of the scalefactors are limited in Layer 3.

3. Coding of the spectral input data using the scalefactor scheme of step 2. If the available number of bits is exceeded, the amplification for the scalefactor bands is reduced and step 3 is repeated. In the actual version, all scalefactor bands are reduced by the same value by lowering the so-called „global gain". More sophisticated algorithms for this job are being developed at the moment.

The use of this algorithm reduces the number of quantization and Bit counting events typically to 10 times. This is a major improvement in comparison to the original algorithm. As the hereby achieved audio quality is slightly worse, this approach is currently not included in the commercial application. All numbers outlined as follows are received with the conventional loops realization. Optimization work is under progress.

End-point

Using optimizations outlined in the previous section real-time Layer 3 encoding on an Intel Pentium processor is possible without a loss in quality. The actual version of the real-time Pentium encoder gives the following performance:

	Pentium 90MHz	Pentium 100 MHz	Pentium 120 MHz	Pentium 133 MHz	Pentium 200 MHz*
MPEG 2 16 kHz	mono/stereo	mono/stereo	mono/stereo	mono/stereo	mono/stereo
MPEG 2 22.05 kHz	mono	mono/stereo	mono/stereo	mono/stereo	mono/stereo
MPEG 2 24 kHz	mono	mono	mono/stereo	mono/stereo	mono/stereo
MPEG 1 32 kHz	mono	mono	mono	mono/stereo	mono/stereo
MPEG 1 44.1 kHz	--	mono	mono	mono	mono/stereo
MPEG1 48 kHz	--	--	mono	mono	mono

*estimated

To provide a comparison between the original and the optimized encoder, the following table shows the time necessary to encode PCM Files with a duration of 10 seconds with the optimized encoder at a sample rate of 44.1 kHz:

	Pentium 100 MHz	Pentium 120 MHz	Pentium 133 MHz
MPEG1 44.10 kHz mono	7.0 sec	6.1 sec	5.4 sec
MPEG1 44.10 kHz stereo	14.7 sec	12.8 sec	11.3 sec

Figure 6 shows the distribution of the calculation time over the basic Layer 3 encoder modules after the optimizations.

On a 200 MHz Pentium processor all sampling and bitrates are supported, except for 48 kHz. With more optimization work a complete MPEG Layer 3 realtime software encoder supporting all sampling rates will be possible in near future.

4 Applications

The transmission links for consumer data connections are usually low-bitrate data links, e.g. ISDN with 56/64 kbps or 14.4/28.8 kbps modem links. With this slow connections the download of uncompressed audio data would result in very long transmission times. One minute of CD music would require a transmission time of about 22 min with a 64 kbps ISDN connection. By using highly compressed Layer 3 encoded bitstreams this transmission can even occur in realtime with a satisfying sound quality and without expensive hardware (e.g. DSP boards). A Pentium PC and a standard sound board are the only system requirements. Examples of possible new applications using a realtime software only encoder are:

Hard-Disk-Recoding

The new Layer 3 software encoder offers the possibility to save compressed audio data on a data storage medium, e.g. a hard disk, in real time. No intermediate raw pcm data needs to be stored anymore. Hard disk recording is advantagous for many reasons. It is fast, all data are randomly seekable on a hard disk and the price of possible data storage media is rather inexpensive these days. With the realtime decoder the recorded items can be replayed immediatly.

Internet Radio

The Internet radio is an example for a distribution application of the Internet. In realtime an encoded and data compressed radio program can be transmitted via low bit rate connections to the consumers in good quality. The quality depends of course on the used transmission rate. The received data can be decoded with the Layer 3 realtime decoder and played at once.

Internet Phone

With the internet phone a typical point to point application is mentioned here. Both sides need a realtime encoder and decoder. Two communication lines are necessary to establish a „full duplex" connection. The sound quality of such a new generation phone will be at least equal to the used performance of a conventional phone connection. The large delay, introduced by the net transmission and the coding algorithm itself, is the main disadvantage of such an application.

5 Conclusion

This paper outlined some principles of the MPEG Layer 3 audio coding scheme.
Optimization examples of a Layer 3 encoder simulation code are discussed, which are necessary to achieve a software only realtime solution of a Layer 3 encoder on Pentium PCs. The final audio quality of the realtime version is comparable to the quality of the Layer 3 simulation code, which is known to be the best amongst the MPEG family.
With the realtime Layer 3 software only encoder the gate to various applications is opened now without expensive DSP hardware solutions. A normal PC with a Pentium processor and a sound board can serve as „multimedia platform". Even no-commercial users can participate now in new applications like hard-disk recording or Internet radio.

6 Acknowledgement

The described work has been partly funded by the Bavarian Ministry for Economy, Transportation and Technology. The authors thank all members of the audio groups at Fraunhofer Institute Integrierte Schaltungen for their support.

7 References

[1] K. Brandenburg, G. Stoll, Y.F. Dehery, J.D. Johnston, L.v.d. Kerkhof, E.F. Schröder: "The ISO/MPEG-Audio Codec: A Generic Standard for Coding of High Quality Digital Audio", 92nd AES Convention, Vienna 1992, preprint 3336

[2] E. Eberlein, H. Popp, B. Grill, J. Herre: "Layer III - A Flexible Coding Standard", 94th AES Convention, Berlin 1993, preprint 3493

[3] ISO/IEC JTC1/SC29/WG11 MPEG, International Standard ISO 11172 "Coding of moving pictures and associated audio for digital storage media up to 1.5 Mbit/s", Part 3: Audio

[4] ISO/IEC JTC1/SC29/WG11 MPEG, International Standard ISO 13818 "Generic Coding of Moving Pictures und Associated Audio", Part 3: Audio

[5] B. Grill, H. Herre, K. Brandenburg et al.: "Improved MPEG-2 Audio Multi Channel Encoding", 96th AES Convention, Amsterdam 1994, preprint 3865

[6] F.-O. Witte: "Single Chip Implementation of an ISO/MPEG Layer III Decoder", 96th AES Convention, Amsterdam 1994, preprint 3805

[7] M. Dietz, H. Popp, K. Brandenburg: "Audio Compression for Network Transmission", 99th AES Convention, New York 1995, preprint 4129

[8] INTEL Application Note AP-500, „Optimizations for Intel's 32-Bit Processors", 1993

[9] Michael L.Schmitt, „Pentium Processor Optimization Tools", Academic Press, New York, 1995

[10] J. D. Johnston: „Estimation of Perceptual Entropy Using Noise Masking Criteria", IEEE ICASSP 1988, pp. 2524-2527

[11] ISO/IEC JTC1/SC29/WG11 MPEG94/0848, "Report on the subjective testing of coders at low sampling frequencies", Singapore 1994

[12] D. Kim, Y. Seo: „Digital Audio Coding Based on Look-Up Tables", 99th AES Convention, New York 1995, preprint 4091

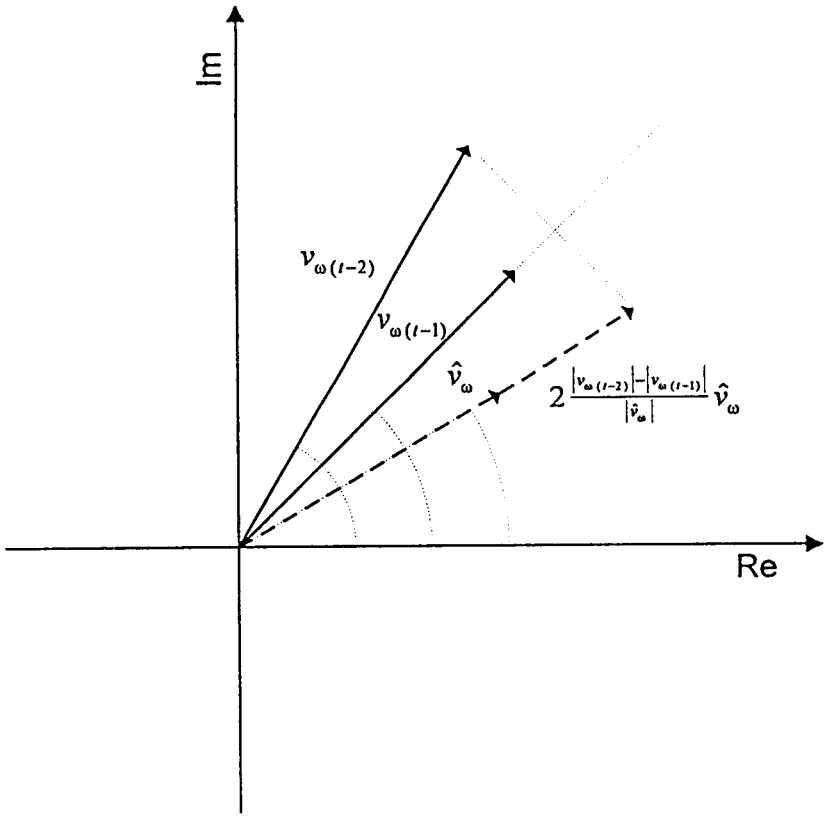

Figure 5: Calculation of the predicted vector $\hat{v}_{\omega}$

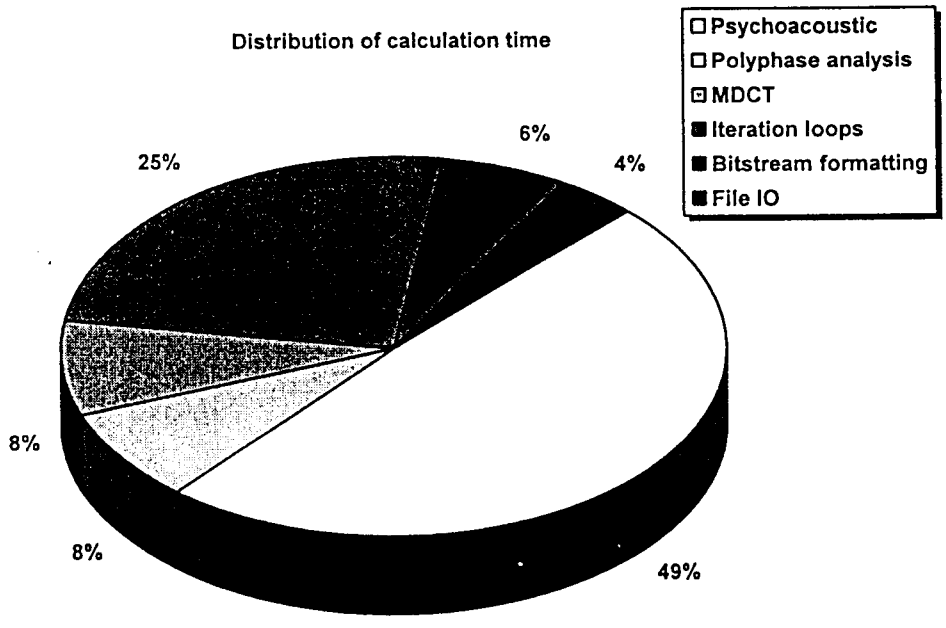

Figure 6: Distribution of calculation time of the real-time Intel Pentium encoder

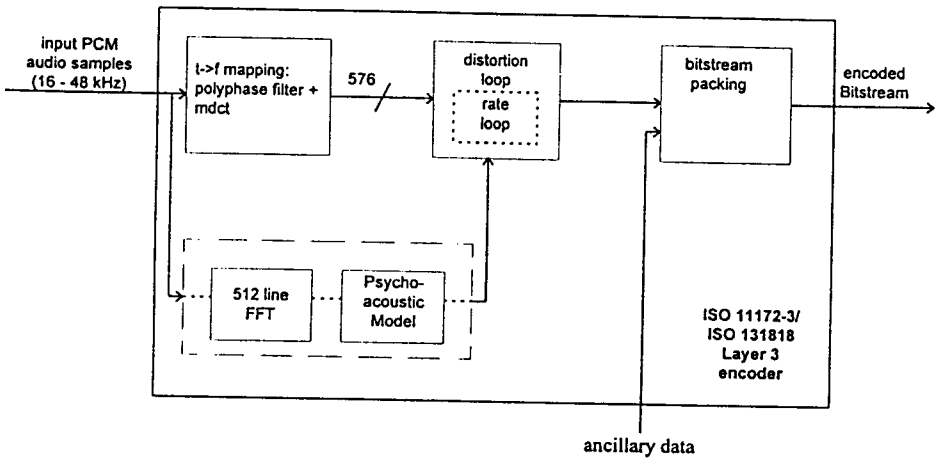

Figure 1: Structure of the MPEG Layer 3 audio encoder

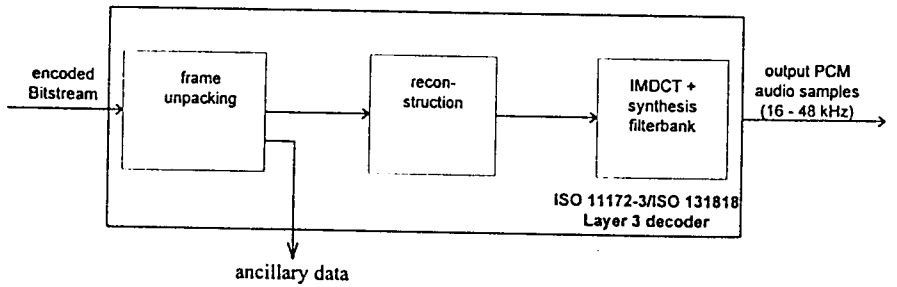

Figure 2: Structure of the MPEG Layer 3 audio decoder

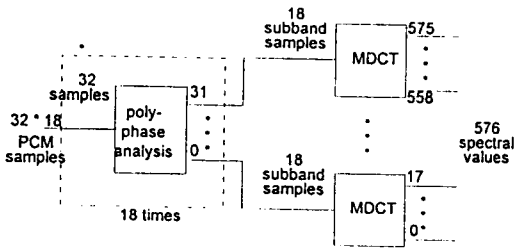

Figure 3: Hybrid filterbank of the Layer 3 encoder

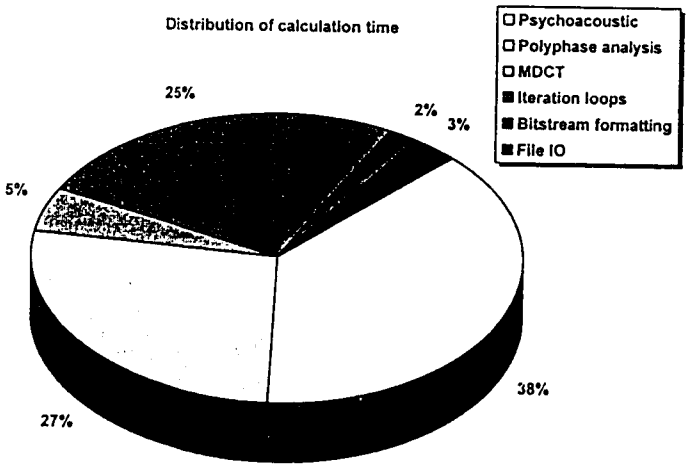

Figure 4: Distribution of calculation time of the simulation encoder