A Low-complexity MP3 Algorithm that Uses a New Rate Control and a Fast Dequantization

Chih-Hsu Yen, Yu-Shiang Lin , and Bing-Fei Wu, *Senior Member, IEEE*

Abstract — *This work presents a low-complexity MP3 algorithm over a fixed-point arithmetic. A new rate control is introduced for the MP3 encoding algorithm, rather than the rate control in ordinary MP3. The computational complexity of the rate control is reduced by taking the loop-independent components outside the loop and accelerating the nonuniform quantizer using a hybrid scheme. The hybrid scheme includes a lookup-table method for smaller numbers and a linear piecewise approximation for larger numbers. A precise method for predicting the quantizer parameter is developed to decrease the number of times the rate control is executed. A hybrid scheme is also used in MP3 decoding algorithm to accelerate the dequantization. However, the approximation for larger numbers is two-tier. The first tier is a linear piecewise approximation that yields a rough value. The second tier uses the rough value as the initial value of the first -order Newton's method to obtain a more closely-approximated value. The precise method for prediction has a statistically hit rate of 43%, and the new rate control consumes no more than 4.5 MIPS. The proposed dequantization consumes no more than 2.38 MIPS, and has an error-to-signal ratio of under 0.012%. The implementation of the complexity-reduced MP3 algorithm over 16bit fixed-point arithmetic is subjectively tested to evaluate the quality of the complexity-reduced MP3 algorithm.***Index Terms — MP3, fixed point, low complexity.**

I. Introduction

The MPEG/AUDIO Layer III [15], also referred as MP3, is the most popular format for digital audio on the Internet. With the help of the Internet, MP3 has also gained popularity as a portable solid-state audio format. Recently, various types of handheld devices, which support MP3 applications, have become available in the consumer market. Most are built with fixed-point processors and raise concerns about computational power and electric power. Therefore, the computational complexity must be reduced to enable the MP3 algorithm could to be ported on power-limited handheld devices; in particular, that of the encoder must be reduced because the MP3 encoder is more complex than the decoder. Many studies have dealt with the reduction of the complexity of the MP3 encoder, Oh et al. [12] removed the psychoacoustic model (PAM) from MP3 algorithm over fixed-point arithmetic, and proposed a fast bit allocation algorithm. Wang et al. [14] simplified the PAM, and presented a bit allocation algorithm. Lee et al. [10] minimized the error, caused by the fixed-point arithmetic, after dequantization of the MP3 decoder.

This work presents a low-complexity MP3 algorithm without loss of quality. A new rate control is developed for the MP3 encoding algorithm, which precisely predicts the quantization parameter; dynamically updates the step size; performs low-complexity nonlinear quantization, and exhibits reduced complexity. The new rate control exhibits a greatly reduced number of iteration and the reduced computational complexity. For the MP3 decoding algorithm, an implementation of dequantization is proposed using a lookup table method, the linear piecewise approximation, and the first-order Newton's root-finding method. The 16-bit fixed-point devices are very common, so quality is measured over a 16-bit fixed-point implementation. The implementation of the new rate control costs no more than 4.5 MIPS, and the dequantization costs no more than 2.38 MIPS. The approximation of nonuniform quantization has an approximated error no more than 1% and the dequantization of not over 0.012%. The quality of the proposed encoder is specified by a average misidentification of around 51% and that of the proposed decoder is about 61%.

The rest of this paper is organized as follows. Section II introduces the MP3 scheme. Section III and Section IV address the proposed low-complexity of the MP3 encoding algorithm and decoding algorithm. Section V explicates the realization of MP3 on a fixed-point 16-bit DSP and compares its performance with those of other studies. Finally, Section VI summarizes the results of this work and suggests directions for future work.

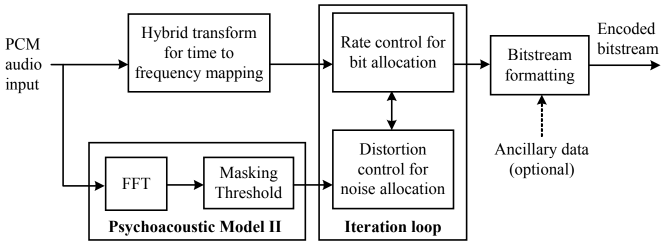

PCM audio input
Hybrid transform for time to frequency mapping
Rate control for bit allocation
Bitstream formatting
Encoded bitstream
Distortion control for noise allocation
Ancillary data (optional)
FFT
Masking Threshold
Psychoacoustic Model II
Iteration loop

Fig. 1. MPEG/audio encoding process.

C.H. Yen is with the Department of Electrical and Control Engineering, National Chiao Tung University, 1001 Ta Hsueh Rd., Hsinchu, Taiwan 300, R.O.C. (e-mail: zsyian@cssp.cn.nctu.edu.tw).

Y.S. Lin is with the Department of Electrical and Control Engineering, National Chiao Tung University, 1001 Ta Hsueh Rd., Hsinchu, Taiwan 300, R.O.C. (e-mail: yslin@cssp.cn.nctu.edu.tw).

Bing-Fei~Wu is with the Department of Electrical and Control Engineering, National Chiao Tung University, 1001 Ta Hsueh Rd., Hsinchu, Taiwan 300, R.O.C. (e-mail: bwu@cssp.cn.nctu.edu.tw).

Contributed Paper
Manuscript received October 13, 2004

0098 3063/05/$20.00 © 2005 IEEE

II. MP3 Overview

This section introduces the MP3 algorithm. Fig. 1 presents a block diagram of the ordinary MP3 encoding process. The audio input is transformed frame-by-frame into spectral components by a time-to-frequency mapping. In the hybrid transformation block, MP3 employs a polyphase filter bank followed by a Modified Discrete Cosine Transform (MDCT) to increase the spectral resolution. Based on the critical-band rate, the spectral components are grouped into several scalefactor bands. The audio input simultaneously undergoes the PAMII to determine the threshold of the ratio of signal energy to masking for each scalefactor band.

The encoding bitrate is constrained by the rate control, which varies the quantizer in an orderly way; quantizes the spectral values, and counts the number of Huffman code bits required to code the quantized values. Huffman coding is chosen as the lossless coding tool, using the predefined Huffman tables [3]. MP3 also adapts the scalefactors to amplify the energy of the spectral band when the quantization noise exceeds the masking threshold. The distortion control adjusts the scalefactors to control the quality. Finally, the information, needed by the decoder, is packaged with compressed audio data to yield a valid MP3 stream.

In the ISO MP3 algorithm, the quantizer is nonuniform. Before the i^{th} spectral component $x_{f,g}(i)$ of the g^{th} granule in the f_{th} frame is quantized, $x_{f,g}(i)$ is first pre-emphasized by (1), then amplified by (2).

$$x'_{f,g}(i) = x_{f,g}(i) \times \sqrt{2}^{(1+z)\times P(b_i)}. \quad (1)$$

$$x''_{f,g}(i) = x'_{f,g}(i) \times \sqrt{2}^{(1+z)\times C(b_i)}. \quad (2)$$

where b_i represents the scalefactor band of the i^{th} spectral line; z determines whether the scalefactors are logarithmically quantized with a step size of 2 or of $\sqrt{2}$; $P(\cdot)$ is the preemphasis table defined in [3], and $C(b_i)$ obtains the scalefactor of the scalefactor band b_i.

Then, $x_{f,g}(i)$ is quantized by the following nonuniform quantizer.

$$y_{f,g}(i) = \mathtt{nint}\left[\left(\left|x''_{f,g}(i)\right| \Big/ 2^{(\delta+q)/4}\right)^{0.75} - 0.0946\right] \quad (3)$$

where `nint` is the rounding function; q is the lower bound of quantization parameter, and δ is the increasing variable.

Fig. 2 shows that the rate control loop, also called the inner iteration loops, allocates bit resources to each spectral line by continually quantizing the audio data by using (3); performing Huffman code, and evaluating the number of bits. The challenge is to find an optimal quantizer parameter, also called the global gain, and to select an appropriate Huffman table. An iterative technique is applied to reach the optimal parameters from an initial value, as determined by the spectral flatness measure. Many iterations in quantization process are tested to guarantee the quantization output in the input range of Huffman coding.

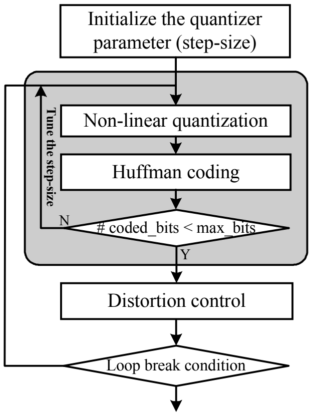


Fig. 2. Rate control in iteration loops.

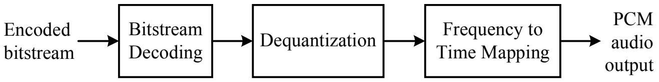


Fig. 3. Decoding block diagram of MPEG/Audio Layer 3.

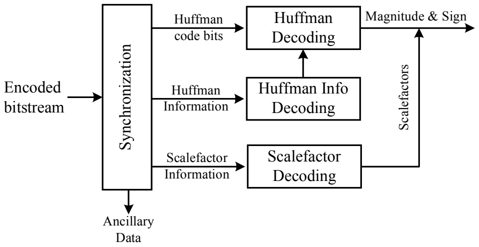


Fig. 4. Bitstream decoding.

The MP3 decoding process has three main parts[3] — bitstream decoding, dequantization and frequency-to time mapping, as shown in Fig. 3. The first part synchronizes the encoded bitstream, and extracts the quantized frequency coefficients and other information about each frame. Fig. 4 details the function blocks.

Dequantization reconstructs a perceptually identical data of the frequency coefficients generated by the MDCT block during encoding. Based on the output of Huffman decoding and scalefactor information, the dequantization calculation is

$$x_{f,g}(i) = (-1)^{s(i)} \cdot y_{f,g}(i)^{4/3} \cdot 2^{\frac{1}{4}(\Delta_{f,g} - 8\Delta_s(w_i))} \Big/ 2^{(1+z)\,(C(b_i)+P(b_i))} \quad (4)$$

where $s(i)$ represents the sign bit of $y_{f,g}(i)$; w_i is the short-block window of the corresponding i^{th} spectral line, and $\Delta_s(w_i)$ is the gain of a short-block window.

The last part is a set of reversed operations of the MDCT and the analysis subband filter bank in the encoder. The frequency-to-time mapping yields the audio PCM output from dequantized coefficients. The alias reduction block adds alias artifacts to dequantized coefficients, to obtain a correct reconstruction of analysis subband filter bank. Then, the inverse MDCT (IMDCT) transforms the coefficients into subband signals in the time domain. Frequency inversion is then applied to compensate for the decimation used in the analysis subband filter bank. After that, the synthesis subband filter bank is applied to the subband signals to yield the audio PCM output.

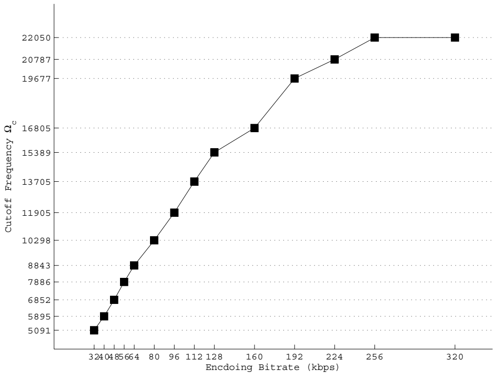


Fig. 5. Coefficients of bandwidth control: the corresponding cutoff frequency at each bit rate is obtained from LAME[7].

TABLE I
PREDICTED COMPLEXITY OF IMPLEMENTING MPEG/AUDIO ENCODER[12]

MP3 Encoder	MIPS
Hybrid transform	25
PAMII	90
Bi t and noise allocation	70
Etc.	5
Total	190

III. COMPLEXITY REDUCTION OF MP3 ENCODING ALGORITHM

In MP3 encoding, PAMII and distortion control are time-consuming. PAMII normally requires transcendental computations such as logarithms, exponentiation and power operations, which are typically computationally demanding. The bit and noise allocation process are also computationally demanding. The quantizer is nonuniform, so MP3 adopts an iterative approach to evaluate the parameters and scalefactors. In the experiment, the number of iterations per audio granule reached 50. For the real-time encoding, it decreases the efficiency of resource usages. Table I summarizes the complexity of the MP3 encoding algorithm in DSP MIPS. It takes 90 MIPS to execute PAMII and 70 MIPS to execute bit and noise allocation. Both require about 84% of overall computation. Therefore, the optimization of the encoder focuses on these two parts.

A. Simplified PAMII

Table I reveals that the ordinary MP3 encoding algorithm consumes too much MIPS and is hard to be implemented on power-limited devices. Complexity analysis shows that both of the most computationally demanding processes are associated to ISO PAMII, so the possibility of encoding without ISO PAMII [12] is addressed at first.

By analyzing the number of distortion control iteration and the results of subjective quality preference tests, Oh et al. [12] found that PAMII is ineffective when the bit rate exceeds 256 kbps. Therefore, the PAMII and processes, related to PAMII, including distortion control and window switching in the hybrid transform were removed.

Many implementation [12][7] involve the bandwaidth control scheme to recover the audio quality at a low bitrate without PAMII or distortion control, because people prefer sound with a limited bandwidth to the sound with full bandwidth but unmasked distortion. This work employs bandwidth control to the input signal. A low pass filter (5) is applied in the bandwidth control, and the cutoff frequency is defined as the bandwidth coefficients in Fig. 5.

$$L(x_{f,g}(i)\,) = \begin{cases} x_{f,g}(i)\,, & \text{if } i \le \mathtt{nint}\left(\dfrac{576 \cdot \Omega_c}{f_s\ /\ 2}\right), \\ 0, & \text{if} > \mathtt{nint}\left(\dfrac{576 \cdot \Omega_c}{f_s\ /\ s}\right), \end{cases} \qquad (5)$$

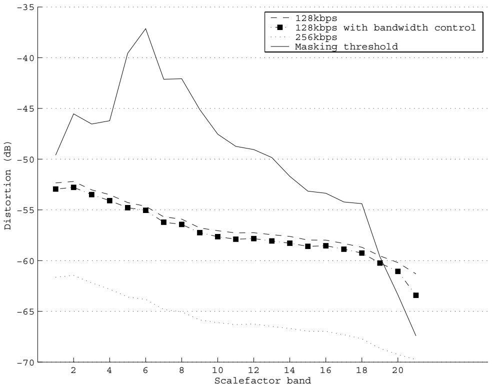


Fig. 6. Decline of average distortion before distortion control is applied to iteration loop.

where the cutoff frequency Ω_c is defined as the bandwidth coefficient in Fig. 5. In other words, the bandwidth control also allows more bits for low-frequency signal, and thus the quality can be improved. Fig. 6 presents the decline in the average distortion when the bandwidth control is employed. Experiments indicate that the bandwidth control scheme is effective when the MP3 is encoded at lower bit rate.

1) Removal of window switching: Many modern audio coding algorithms frequently use dynamic window switching to avoid preechoes. Preecho occurs when the audio signals, which the amplitude raises violently in an instant, are encoded. The PAMII detects the preecho by calculating the perceptual entropy (PE) — that is, by predicting the amount of bits required to encode the granule. However, PAMII is not implemented in the proposed design for power-limited devices. Oh et al.[12] showed that encoding without window switching did not significantly negatively effects the audio quality. This work removes the window switching scheme, and uses only the long window to encode the audio signal.

Contributed Paper
Manuscript received October 13, 2004

B. New rate control loop

This works proposes a new rate control algorithm instead of the rate control in ISO MP3. As PAMII and the distortion control loop are removed, the iteration loop is simplified as depicted in Fig. 7. The nonuniform quantization is split into two steps — one is performed outside the loop, and the other is calculated inside the loop. A dynamic bit allocation is proposed to allocate the available bits for each granule. A *prediction* is used to guess the initial value of the step size of quantization. The *first trial* checks whether the initial step size meets the bit requirement or not. The *iterative search* modifies continually the step size, while the prediction fails. The *fine-tuning process* updates the step size in a finer way than the *iterative search*. Each block is detailed below.

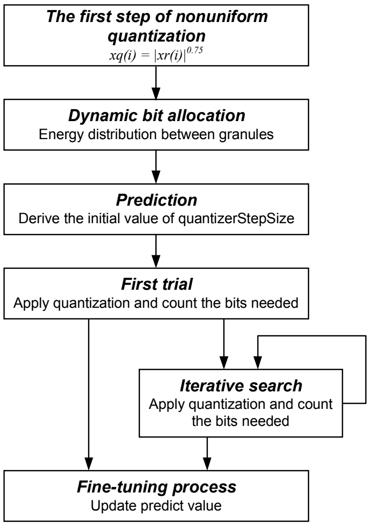

The first step of nonuniform quantization
xq(i) = |xr(i)|^0.75
Dynamic bit allocation
Energy distribution between granules
Prediction
Derive the initial value of quantizerStepSize
First trial
Apply quantization and count the bits needed
Iterative search
Apply quantization and count the bits needed
Fine-tuning process
Update predict value

Fig. 7. New rate control algorithm.

1) Nonuniform quantization: The proposed algorithm does not use the distortion control, so (1) and (2) no longer exist. Therefore, $x''_{f,g}(i)= x_{f,g}(i)$. Letting $\Delta_{f,g} =\delta+q$, then (3) can be simplified to (6).

$$y_{f,g}(i) = \mathtt{nint}\left[\left(\left|x_{f,g}(i)\right| \Big/ 2^{\Delta_{f,g}/4}\right)^{0.75} - 0.0946\right] \quad (6)$$

In Fig. 2, (6) is executed continually in the process of finding an optimal step size $\Delta_{f,g}$. The rounding function `nint()` is unnecessary in fixed-point implementation. (6) can be further rearranged to (7).

$$y_{f,g}(i) = \left|x_{f,g}(i)\right|^{0.75} \times 2^{-\frac{3\Delta_{f,g}}{16}} - 0.0946 \quad (7)$$

In the rate control iteration, $\Delta_{f,g}$ is the only variable that involves the iterative search, so $|x_{f,g}(i)|^{0.75}$ is removed from the iteration to be calculated once only. Therefore, the quantizer can be decomposed into two equations, where (8) is calculated outside the iteration and (9) is calculated inside the iteration.

TABLE II
THE NUMBER OF DSP INSTRUCTION CYCLES IN CALCULATION OF TWO REGIONS

Input range	Probability	Table Size
0 – 31	>60%	32 words
32 – 65536	<40%	22 words

$$\hat{x}_{f,g}(i) =| x_{f,g}(i) |^{0.75} \quad (8)$$

$$y_{f,g}(i) = \hat{x}_{f,g}(i) \times 2^{-\frac{3\Delta_{f,g}}{16}} - 0.0946 \quad (9)$$

The decomposition reduces the computation of the nonuniform quantizer in the *iterative search*. The most computationally demanding process, the $|x_{f,g}(i)|^{0.75}$ function, is calculated only once in each granule. The fixed-point implementation of (9) can be implemented simply using one multiplication, one shift operation and one subtraction.

The implementation of $|x_{f,g}(i)|^{0.75}$ is optimized for 16-bit fixed-point arithmetic. The unsigned 16-bit fixed-point inputs, ranging from 0 to 65,535, are divided into two regions. The first region covers the range 0 to 31 and is implemented using a 32-word lookup table to accelerate the calculation. The statistics revealed that the first region covered over 60% of the inputs. The second region from 32 to 65,535 is approximated by a piecewise linear interpolation. 11 subregions are present between 32 to 65,535. The approximation error was analyzed in (10).

$$\varepsilon(x) = \frac{x^{0.75} - Q_1(x)}{x^{0.75}}, \text{ for } x = 0,1,2, \Lambda ,65536, \quad (10)$$

where $Q_1(\cdot)$ is the proposed implementation, described above, of (8).

(9) can be rewritten as (11)

$$y_{f,g}(i) = \hat{x}_{f,g}(i) \times 2^{\Delta_Q} \times 2^{\Delta_N} - 0.0946 \quad (11)$$

where Δ_N is the integer part of $-3\times\Delta_{f,g}$, and Δ_Q is the fractional part of $-3\times\Delta_{f,g}$. In fixed-point implementation, the multiplication of $2^{\Delta N}$ can be easily performed by the hardware barrel shifter, and the $2^{\Delta Q}$ can be derived from a 16-word lookup table that contains fixed point values, $2^{0/16}, 2^{1/16}, \ldots, 2^{15/16}$

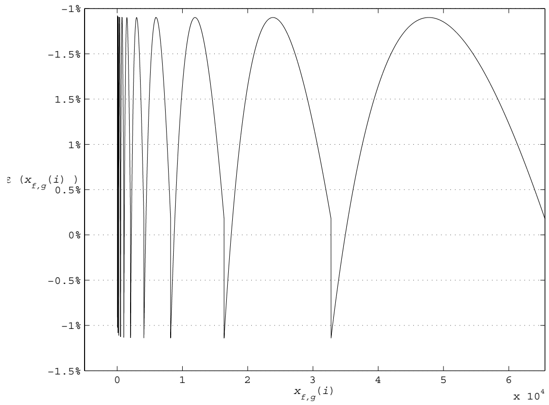


Fig. 8. Error of $x^{0.75}$ approximation.

Fig. 8 presents the error after approximation $x^{0.75}$. The error ratio is bounded in |1.2%|.

2) Fast search for the optimal quantizer parameter: This works also proposes a fast search approach with two parts. The first is *the precise prediction of the quantization parameter* $\Delta_{f,g}$. The other is *the fast search of the optimal quantizer parameter.* The former is applied in the *prediction* block, and the later is in the *iterative search* block.

The *precise prediction of the quantization parameter* is described first. In the ISO MP3 algorithm, the initial value of the quantization parameter is derived as,

$$q = 8\ln(m)\cdot \tag{12}$$

The spectral flatness measure m is defined as (13). The derivation contains complex nonlinear mathematics, and is inefficient for fixed-point implementation.

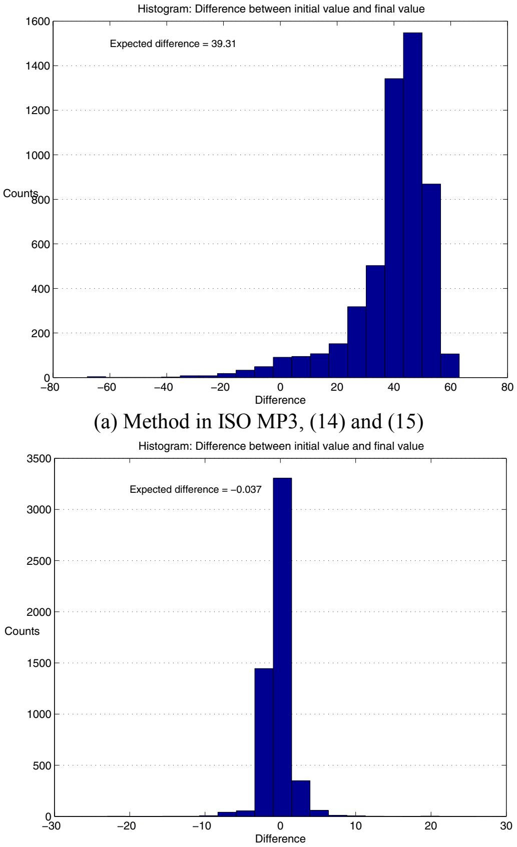
$$m = \frac{e^{1/576}(\sum_{i=0}^{575}\ln x_{f,g}(i)^2)}{\sum_{i=0}^{575} x_{f,g}(i)^2 / 576} \tag{13}$$

This work presents that the initial step size $\Delta_{f,g}(0)$ is predicted from the one of previous frame and a derived lower bound Δ_l of (14). The lower bound is derived by considering (9). The properties of the Huffman table are such that the quantized value, $y_{f,g}(i)$, has a upper bound, 8,207. Therefore, the lower bound of $\Delta_{f,g}$ is derived as follows.

$$\begin{aligned} & 8207 > \hat{x}_{f,g}(i) \times 2^{-3\Delta_{f,g}/16} - 0.0946, \\ \Rightarrow\ & 8207 > \hat{X}_{f,g} \times 2^{-3\Delta_{f,g}/16} - 0.0946, \\ \Rightarrow\ & \Delta_l = \left\lceil \frac{16}{3}\log_2 \hat{X}_{f,g} - 69.35 \right\rceil, \\ & \text{where } \hat{X}_{f,g} = \max_i\{\hat{x}_{f,g}(i)\}. \end{aligned} \tag{14}$$

Δ_l guarantees the quantized value in the range of Huffman table. Therefore, we define the *prediction* as the case n=0 in (15), and the *iterative search* as the 2nd case.

$$\Delta_{f,g}(n) = \begin{cases} \max\{\Delta_l, \Delta_{f,g-1}\}, & \text{if } n = 0 \\ \max\{\Delta_l, \Delta_{f,g}(n-1) + \sigma\}, & \text{if } n = 1, 2, K \end{cases}, \tag{15}$$



(a) Method in ISO MP3, (14) and (15)

(b) Proposed method, (14) and (15)

Fig. 9. Histograms of difference between initial value and final value of $\Delta_{f,g}$. The accuracy is determined by the expected difference between the initial value and the final value. (a) ISO method initializes $\Delta_{f,g}$ by the measure of spectral flatness. (b) Proposed method initializes $\Delta_{f,g}$ by the one of previous granule and a lower bound.

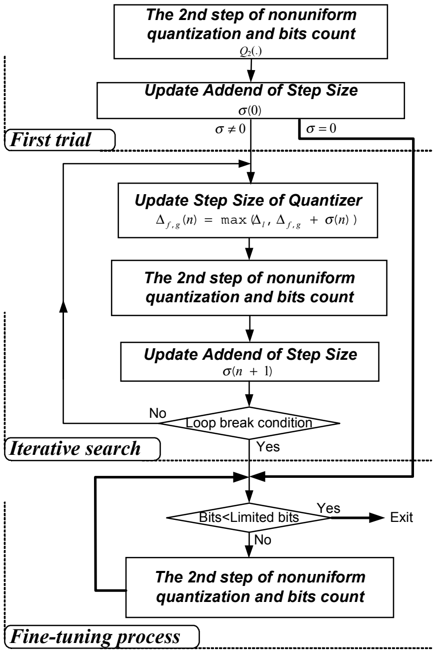

The 2nd step of nonuniform quantization and bits count
$Q_2(.)$
Update Addend of Step Size
$\sigma(0)$
$\sigma \neq 0$
$\sigma = 0$
First trial
Update Step Size of Quantizer
$\Delta_{f,g}(n) = \max(\Delta_l, \Delta_{f,g} + \sigma(n))$
The 2nd step of nonuniform quantization and bits count
Update Addend of Step Size
$\sigma(n+1)$
No
Loop break condition
Yes
Iterative search
Bits<Limited bits
Yes
Exit
No
The 2nd step of nonuniform quantization and bits count
Fine-tuning process

Fig. 10. The adaptive approach to iterative search optimum parameter.

Contributed Paper
Manuscript received October 13, 2004

where the$\Delta_{f,0}$=150; σ is an updated addend of the step size, and $\Delta_{f,g}(-1) = \Delta_{f,g-1}$.

The great achievement of this approach is demonstrated by comparing the difference between the initial value and the final value. Fig. 9 shows the difference histograms of ISO MP3 method and this approach. Statistics reveal that over 60% of predicted initial values are very close to the final values, i.e. the difference $|\varepsilon| \leq 1$. The precise decision of the initial value reduces the number of iterations in the following iterative search.

Fig. 10 illustrates the *iterative search* developed in this work. The fast search can be divided into three parts. The first part, the *first trial*, performs quantization where the quantization parameter, $\Delta_{f,g}(n)$, is calculated by (15). The subroutine $Q_2()$ is the implementation of (11). Then the number of bits needed by Huffman coding is counted to check whether the bit requirement is met or not. A dynamically updated addend σ is used to tune the step size, and it is obtained by

$$\sigma(n+1) = \mathrm{nint}\left(\frac{b(n) - B_{f,g}}{p(n)}\right) \qquad (16)$$

$$p(n+1) = \mathrm{nint}\left(\frac{3}{4}p(n-1) + \frac{1}{4}\frac{b(n+1) - b(n)}{\sigma(n)}\right) \qquad (17)$$

where $b(n)$ is the number of bits needed by Huffman coding, and $p(n)$ is defined as an estimated value of the difference number of bits used in Huffman coding, while $\Delta_{f,g}$(n) is increased by one.

The *iterative search* is then applied, when the *first trial* fails. In the n^{th} iteration, the $\Delta_{f,g}(n)$ is modified by $\sigma(n)$ determined in the $(n-1)^{th}$ iteration and the lower bound Δ_l. After executing the nonuniform quantization $Q_2()$, the number of Huffman coded bits is calculated. Then, $\sigma(n)$ is updated in the same way in the *first trial*. Finally, some loop break condition will be tested. The final trial is used to guarantee that the used bits are fewer than the allocated bits. Unlike in the *iterative search*, $\Delta_{f,g}(n)$ is fine tuned as

$$\Delta_{f,g}(n) = \Delta_{f,g}(n-1) + 1\,. \qquad (18)$$

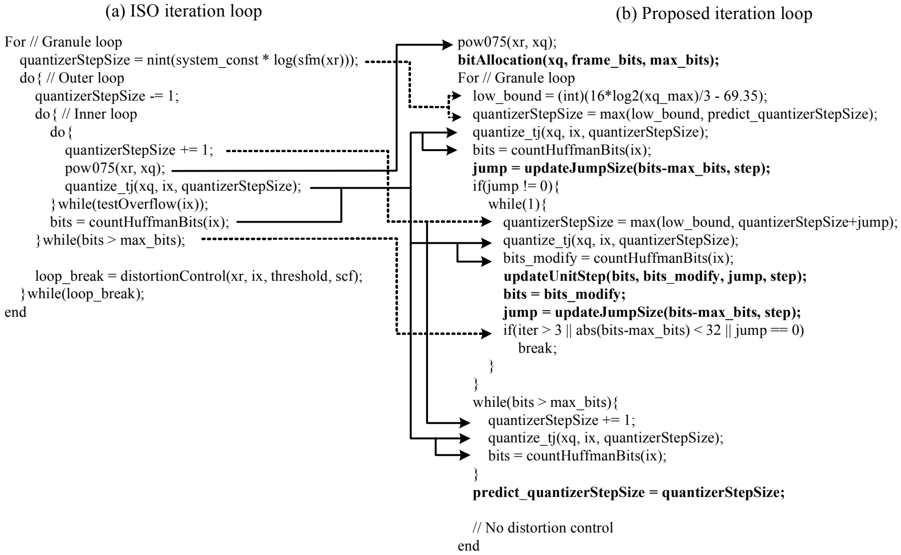
This formula is applied to prevent the deadlock loop condition.

3) Dynamic bit allocation from energy distribution: In MP3 bitstream, each frame has a fixed amount of bit resources at a constant bit rate. With the help of bit reservoir control, rest bits are stored in the reservoir, if the distortion of the quantization is imperceptible. It benefits the encoding of succeeding frames. However, the proposed algorithm is without PAMII and distortion control, and so the quality constraint no longer exists. Then, the rate control loop exploits all bit resource all possible bit resources to encode one audio granule.

An audio frame includes normally two (mono) or four (stereo) granules. The MP3 algorithm equally apportions the total bits to granules in each frame. This work proposes an asymmetric allocation of bit resources, determined by energy distribution of granules to equalize the quality, i.e. allowed distortion, between granules in the same frame.

In the MP3 algorithm, a transformed coefficient of higher amplitude is quantized to a larger integer value. The property of the Huffman encoding is such that a lager value is usually coded with more bits. The concept is extended to the granule scale. If the granule has more energy or more possible coefficients of higher amplitude, then it requires more bits to maintain the same quality. The nonuniform property of the quantizer and the power function are considered. Experiments reveal that the frequency lines below 4,000 Hz dominate the full bandwidth (22,050 Hz) energy.

In the proposed approach, at sampling rate of 44.1kHz, the score of the granule energy defined as (19) takes only (4000/(44100/2))×576≅105 spectral lines of $|x_{f,g}(i)|^{0.75}$ output for calculation.



(a) ISO iteration loop

```
For // Granule loop
   quantizerStepSize = nint(system_const * log(sfm(xr)));
   do{ // Outer loop
      quantizerStepSize -= 1;
      do{ // Inner loop
         do{
            quantizerStepSize += 1;
            pow075(xr, xq);
            quantize_tj(xq, ix, quantizerStepSize);
         }while(testOverflow(ix));
         bits = countHuffmanBits(ix);
      }while(bits > max_bits);

      loop_break = distortionControl(xr, ix, threshold, scf);
   }while(loop_break);
end
```

(b) Proposed iteration loop

```
pow075(xr, xq);
bitAllocation(xq, frame_bits, max_bits);
For // Granule loop
   low_bound = (int)(16*log2(xq_max)/3 - 69.35);
   quantizerStepSize = max(low_bound, predict_quantizerStepSize);
   quantize_tj(xq, ix, quantizerStepSize);
   bits = countHuffmanBits(ix);
   jump = updateJumpSize(bits-max_bits, step);
   if(jump != 0){
      while(1){
         quantizerStepSize = max(low_bound, quantizerStepSize+jump);
         quantize_tj(xq, ix, quantizerStepSize);
         bits_modify = countHuffmanBits(ix);
         updateUnitStep(bits, bits_modify, jump, step);
         bits = bits_modify;
         jump = updateJumpSize(bits-max_bits, step);
         if(iter > 3 || abs(bits-max_bits) < 32 || jump == 0)
            break;
      }
   }
   while(bits > max_bits){
      quantizerStepSize += 1;
      quantize_tj(xq, ix, quantizerStepSize);
      bits = countHuffmanBits(ix);
   }
   predict_quantizerStepSize = quantizerStepSize;

   // No distortion control
end
```

Fig. 11. Pseudo code of iteration loops (a) ISO method (b) Proposed method.

$$E_{f,g} = \sum_{i=1}^{105} \hat{x}_{f,g}(i), \tag{19}$$

where $\hat{x}_{f,g}(i)$ are determined from (8), and $E_{f,g}$ is the energy score of the granule.

The resources are allocated not in exactly proportional to the granule score. The modification shown in (20) takes the minimum resources into account. The maximum encoding bits $B_{f,g}$ for each granule are given by

$$B_{f,g} = b_{f,g} + (E_{f,g} / \sum_g E_{f,g}) \times B_p, \tag{20}$$

where $b_{f,g}$ is the minimum encoding bits; B_f is the total number of bits in the frame, and B_p is the preserve bits for each frame. $B_{f,g}$ and $b_{f,g}$ are given as follows.

$$b_{f,g} = \begin{cases} B_f/6, & \text{mono,} \\ B_f/12, & \text{left/right channel} \\ B_f/9 & \text{mid channel} \\ B_f/18 & \text{side channel} \end{cases} \tag{22}$$

$$B_p = B_f - \sum_g b_{f,g}$$

Fig. 11 explains the difference between the ISO rate control and the proposed new rate control in pseudo C code. With removal of PAMII, the distortion control is also removed. The solid lines in Fig. 11 link the blocks with the same functionality, but reduced the complexity in the proposed method. The dotted lines link the blocks with different measurements in the proposed method. And the texts of boldface represent the new blocks of the proposed method.

TABLE III
THE AVERAGE NUMBER OF INNER ITERATION

	ISO	Oh et al.[12]	Ours
Average	45	2.1	1.8
Maximum	>100	3	8

The number of iterations was analyzed by calculating the number of times for executing $Q_1(\cdot)$, $Q_2(\cdot)$, and counting of Huffman bits for each granule, to evaluate the performance of the new rate control process. The computational complexities of $Q_1(\cdot)$, $Q_2(\cdot)$ and counting of Huffman bits are denoted as ϕ_1, ϕ_2 and ϕ_3, respectively. In encoding stereo MP3 with 128kbps, the experiments show that the ISO method takes $45\phi_1 + 45\phi_2 + 47\phi_3$ in average while the proposed method requires only $1\phi_1 + 2\phi_2 + 2\phi_3$. Table III lists the number of inner iteration. The proposed method requires fewer iterations than other methods. The decomposition of the nonuniform quantizer, (8) and (9), is such that the computational complexity of inner iteration much less than that associated with other methods.

IV. COMPLEXITY REDUCTION OF MP3 DECODING ALGORITHM

In the ISO MP3 decoder, dequantization, IMDCT, and subband synthesis in particular, require a large number of arithmetic operations, and produce quantization noise in fixed-point implementation. This work proposes a new implementation of dequantization, and adopts a fast algorithm on IMDCT and synthesis polyphase filterbank.

A. Dequantization

The dequantization equation is shown in (4). The complexity arises from calculating $y_{f,g}(i)^{4/3}$, where $y_{f,g}(i)$ is an integer ranging from 0 to 8,207. The direct derivation using mathematical libraries is too time-consuming and not suited in real-time applications. For simplicity, the index of spectral line, i, is ignored herein.

First, the calculation $y_g^{4/3}$ is decomposed into (23) to minimize the quantization noise of fixed-point implementation.

$$y_g^{4/3} = y_g^{1/3} \cdot y_g \tag{23}$$

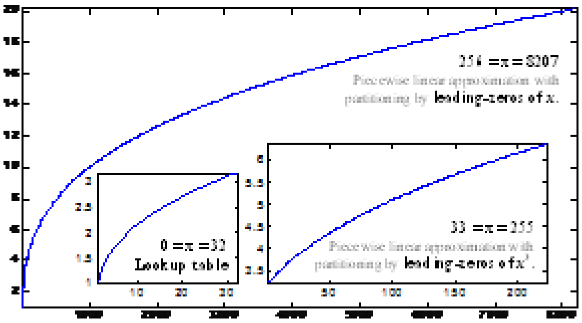


Fig. 12. The implementation of $y_g^{1/3}$.

Comparing the dynamic range of $y_g^{4/3}$ (0 to 165543.68) to that of $y_g^{1/3}$ (0 to 20.171) indicates that the fixed-point implementation of $y_g^{1/3}$ produces less quantization noise than $y_g^{4/3}$, because smaller dynamic range is smaller.

As in the case of the encoder, the power function is implemented using a hybrid scheme. First, the input range is split into three sections, as shown in Fig. 12. The first section of $0 \le y_g \le 32$ utilizes a small lookup table, to obtain the real value directly. The other two sections adopt the piecewise linear approximation method.

The approximation error was analyzed; the error to real output ratio is around ±1%, and the SNR is around 46dB. The error is still too large, and will probably cause the subsequent processes, such as IMDCT and Subband synthesis, to produce more error, especially in fixed-point implementation. Nevertheless, in the encoding case, the subsequent process, Huffman coding, is noiseless.

Newton's method in the section of $33 \le y_{f,g} \le 8207$ is used

Contributed Paper
Manuscript received October 13, 2004

to perform the fine approximation, using the result of piecewise linear approximation as initial value. (24) is an another form of $u = y_g^{1/3}$ and suitable for Newton's method of root-finding to yield a value of *u*, which approximates $y_{f,g}^{1/3}$.

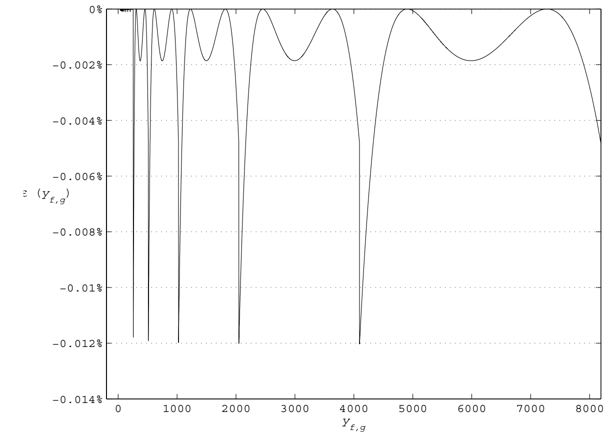


Fig. 13. The error to real output ratio of $y_g^{4/3}$ approximation, $\varepsilon(y_g) = \frac{y^{4/3} - pow3(y_g)y_g}{y^{4/3}}$, where pow3 is the proposed implementation of $y_g^{1/3}$.

$$u^3 - y_{f,g} = 0 \tag{24}$$

The function result is calculated by repeated iterations, which successively reduces the residual error $|\ u - y_{f,g}^{1/3}\ |$. The iteration formula is

$$\tilde{u}_1 = \tilde{u}_0 - \frac{\tilde{u}_0^3 - y_{f,g}}{3\tilde{u}_0^2} = \frac{1}{3}\left(2\tilde{u}_0 + \frac{y_{f,g}}{\tilde{u}_0^2}\right), \tag{25}$$

TABLE IV
The Comparison of Peak Consumed MIPS in Different MP3 Encoders

	Ours	Wang et al.[20]	Oh et al.[15]
Iteration loops	4.5	11.87	18.43
Huffman encoding and bitstream	5.47	5.86	2.07
Overall Computing	21.05	36.07	30.9

The initial value $\tilde{u}_0$ is calculated from the piecewise linear approximation described above, and yields the desired accuracy in only one iteration. Fig. 13 shows the error to real output ratio. The ratio is around ±0.01% and the SNR is increased to 86dB.

B. IMDCT and Subband Synthesis

The frequency-to-time mapping tool is also computationally demanding. Based on the analysis of Lee et al. [10], Lee's Fast DCT algorithm [8] is adopted herein in the fast algorithms of IMDCT and the subband synthesis blocks. For the IMDCT block, Lee's 9-point Fast IDCT is applied, and the Lee's 64-point Fast DCT is used in matrixing routine in subband synthesis block.

V. Comparisons and Results

The low-complexity MP3 algorithm is applied on a 16-bit fixed-point DSP. Table IV presents the computational complexity and compares it with those of corresponding values for the encoders of Oh et al.[12] and Wang et al.[14] encoders. Oh et al.[12] implemented an MP3 encoder over a specific 20-bit fixed-point DSP with fast bit allocation, and with the removal of PAMII and window switching. Wang et al.[14] implemented a real-time encoder on a 50 MIPS 16-bit fixed--point DSP, with new PAM, based on MDCT, and with fast bit allocation. The worst-case results are listed for signal-dependent functions, such as iteration loops and Huffman encoding. The proposed encoder has many fewer computational loads of iteration loops and overall computation than does either of the other two encoders.

TABLE V
The Comparison of Peak Consumed MIPS in Different MP3 Decoders

	Ours	Lee et al.[10]	Bang et al.[1]
Dequantization	2.38	5.4	4.5
IMDCT	4.45	2.8	2.85
Subband synthesis	4.45	6.3	5.97
Overall Computing	17.67	20.7	13.33

TABLE VI
Test Audio Samples

Signal characteristic	Time	Abbreviation
Violin solo in arpeggio	0:37	VL
Melodious quartet	0:28	QT
German female speech	0:21	GF

Table V presents the computational complexity of each part of the proposed MP3 decoder and compares the values with those for the decoders of Lee et al. [10] and Bang et al. [1]. Lee et al. implemented the MP3 decoder on a dual-core processor with a 32-bit RISC MCU and a 16-bit fixed-point DSP; Bang et al. implemented it on a specific dual-core processor with a 20-bit fixed-point DSP core, which supports Huffman decoder in hardware. The great achievement of the MP3 decoder is the dequantization and subband synthesis. However, the decoder proposed by Bang et al. has special instruction for Huffman decoding, hence it yields a very low computational complexity.

The audio quality of the proposed MP3 encoder and decoder is evaluated subjectively using "Double blind triple

TABLE VII
The Subjective Evaluation Results (1), DG: Diffgrade. MI: Number of Misidentification over 11 (Proposed encoder/ISO decoder)

Bit rate		VL	QT	GF
192 kbps	DG	-0.04	0.02	0.2
	MI	7	7	10
128 kbps	DG	-0.4	-0.3	0.04
	MI	6	6	9
96 kbps	DG	-0.7	-0.55	-0.46
	MI	3	4	6

stimulus with hidden reference" listening tests [4]. Three audio samples, as summarized in Table VI, are used herein. All samples are stereophonic, and were sampled at 44.1kHz. The experiment involves 11 listeners.

TABLE VIII
THE SUBJECTIVE EVALUATION RESULTS (2). ISO ENCODER/PROPOSED DECODER

Bit rate		VL	QT	GF
192 kbps	DG	-0.02	-0.02	0.01
	MI	9	8	9
128 kbps	DG	-0.1	-0.04	-0.04
	MI	8	8	9
96 kbps	DG	-0.1	-0.04	-0.06
	MI	9	9	9

TABLE IX
THE SUBJECTIVE EVALUATION RESULTS (3). PROPOSED ENCODER/PROPOSED DECODER

Bit rate		VL	QT	GF
192 kbps	DG	-0.25	-0.4	0.0
	MI	6	7	7
128 kbps	DG	-0.7	-0.5	-0.5
	MI	3	3	6
96 kbps	DG	-1.02	-0.8	-0.6
	MI	2	3	5

The "Diffgrade" and the "number of misidentification items" are presented in three tests. Diffgrade is the subjective rating of the coded test item, minus the rating of the hidden reference. The Diffgrade scale covers into five ranges — "imperceptible (> 0.00)", "perceptible but not annoying (0.00 ~−1.00)", "slightly annoying (-1.00 ~ −2.00)", "annoying (-2.00 ~ −3.00)" and "very annoying (−4.00)". The "number of misidentifications" is the number of subjects who incorrectly identify the test item and the hidden reference.

Table VII shows the results of encoding MP3 using the proposed encoder and then decoding using the ISO decoder. Table VIII presents the results of encoding MP3 using the ISO encoder and then decoding using the proposed decoder. Table IX lists the results of encoding MP3 using the proposed encoder and then decoding using the proposed decoder. These tables show that most Diffgrades fall in the range (0.00 ~ 1.00). The VL sample is the only exception, and is graded −1.02 over 96kbps in the third case.

VI. CONCLUSIONS

This work develops a low-complexity MP3 codec using fixed-point arithmetic without loss of quality. The complexity of iteration loops requires 4.5 MIPS and the proposed encoder requires 21.05MIPS on a fixed-point 16-bit DSP. The proposed dequantization of the MP3 decoder costs 2.38 MIPS and the overall computation of the decoder requires 17.67 MIPS. The new rate control and precise prediction for the quantization parameter indeed reduce the computational complexity of the MP3 encoder. The proposed approximation for nonuniform dequantization also reduces the computational complexity of the MP3 decoder. Hence, it requires less computational power and memory, and so is easily ported on portable devices, such as PDAs, mobile phones, voice recorders, and others.

REFERENCES

[1] Kyoung Ho Bang, Nam Hun Jeong, Joon Seok Lim, Young Cheol Park, and Dae He Youn, "Design and VLSI Implementation of a digital Audiospecific DSP Core for MP3/AAC", *International Conference on Consumer Electronics, Digest of Technical Papers*, pp.220-221, June 2002.

[2] European Broadcasting Union, EBU SQAM, http://www.tnt.uni-hannover.de/project/mpeg/audio/sqam/

[3] ISO/IEC JTC1/SC29/WG11 MPEG, International Standard IS 111723, "*Coding of Moving Pictures and Associated Audio for Digital Storage media at up to about 1.5M bit/s, Part 3: Audio*," 1993.

[4] ITUR Rec. BS.1116, "Methods for the Subjective Assessment of Small Impairment in Audio Systems Including Multichannel Sound Systems", *International Telecommunication Union*, Geneva, Switzerland, 1994.

[5] ITUR Recommendation BS.13871, "*Method for Objective Measurements of Perceived Audio Quality*", Dec. 1998.

[6] Minseep Jeong, Seehyun Kim, Jongseo Sohn, and JiYang Kang, "Finite Wordlength Effects Evaluation of the MPEG2 Audio Decoder", *International Conference on Signal Processing Applications & Technology*, pp.351-355, Jan. 1996

[7] Lame Aint an MP3 Encoder (LAME), http://sourceforge.net/projects/lame/

[8] Byoung Gi Lee, "A New Algorithm to Compute the Discrete Cosine Transform", *IEEE Trans. On Acoustic, Speech and Signal Processing*, Vol. ASSP32, NO.6, pp.1243-1245, 1984.

[9] Kyu Ha Lee, KeunSup Lee, TaeHoon Hwang, YoungCheol Park, and Dae Hee Youn, "An Architecture and Implementation of MPEG Audio Layer III Decoder Using Dualcore DSP", *IEEE Trans. on Consumer Electronics*, Vol. 47, No. 4, pp.928-933, Nov. 2001.

[10] KeunSup Lee, HyenO Oh, YoungCheol Park, and Dae Hee Youn, "High Quality MPEG audio Layer III Algorithm For a 16bit DSP", *Proceedings of IEEE International Symposiumon Circuit and Systems*, vol. II, pp.205-208, Sydney, Australia, May 69, 2001.

[11] Peter Noll, "MPEG digital audio processing", *IEEE Signal Processing Magazine*, pp.59-81, Sep. 1997.

[12] H. Oh, J. Kim, C. Song, Y. Park and D. Youn, "Low Power MPEG/Audio Encoders Using Simplified Psychoacoustics Model and Fast Bit Allocation", *IEEE Transaction on Consumer Electronics*, Vol.47, No.3, pp. 613-621, Aug. 2001.

[13] D. Pan, "A Tutorial on MPEG/Audio Compression", *IEEE Trans. on Multimedia*, Vol.2, No.2, pp.60-74, 1995.

[14] Xin Wang, Weibei DOU and Zhaorong HOU, "An Improved Audio Encoding Architecture Based on 16Bit FixedPoint DSP", IEEE 2002 International Conference on Communications, Circuits and Systems and West Sino Expositions, Vol. 2, pp.918-921, Jul. 2002.

[15] E. Zwicker, and H. Fastl, "*Psychoacoustics: Facts and Models*", SpringerVerlag, New York, 1990.

Contributed Paper
Manuscript received October 13, 2004