第30卷 第15期 计 算 机 工 程 2004年8月
Vol.30 №15 Computer Engineering August 2004
·软件技术与数据库· 文章编号：1000—3428(2004)15—0083—02 文献标识码：A 中图分类号：TP301.6

MP3定点解码算法的设计与实现

江 巍，杨 军，罗 岚，胡 晨
（东南大学国家ASIC系统工程技术研究中心，南京 210096）

摘 要： 提出了一种用于嵌入式系统的MP3定点解码算法。该算法的核心是用定点数和定点计算代替浮点算法，并对MP3解码的各个过程进行优化设计。该算法在以ARM处理器为核心的嵌入式系统上完成了代码实现，经过FPGA验证和性能分析表明，其解码速度和音质完全可以达到设计要求。

关键词： 定点算法；MPEG；音频解码

Design and Implementation of Fixed-point Algorithm for MP3 Audio Decoder

JIANG Wei, YANG Jun, LUO Lan, HU Chen
(National ASIC System Engineering Research Center of Southeast Univ., Nanjing 210096)

【Abstract】 This paper presents a fixed-point MP3 decoding algorithm for embedded systems. The key improvement is fixed-point integers and routines are used instead of float point ones and the most procedures are optimized for high performance. This algorithm has been implemented in ARM-based SoC system and the result satisfies decoding efficiency and music quality for real-time playback in FPGA development board.

【Key words】 Fixed-point algorithm; MPEG; Audio decoder

1 概述

MPEG-1/2 Audio Layer 3，简称MP3，是专门针对音乐和语音数据进行设计的有损压缩算法。MP3的编码和解码算法比较复杂，但压缩率较高。一般情况下，CD音质MP3音乐可以达到10:1的压缩率，因而在数字音乐的存储和播放领域，尤其在手持式设备中得到了广泛的应用。

传统的MP3解码算法需要完成大量的高精度浮点计算，其解码速度和效率依赖于通用浮点运算单元的支持。大部分的专用MP3解码播放系统均需要使用DSP处理器或浮点协处理器处理浮点指令并完成高速浮点运算过程，因而系统设计复杂、成本高、通用性差、难以升级换代。

中低端嵌入式系统受其应用范围的影响，比较注重于成本、体积和功耗等因素，因此CPU核心的处理能力较弱，主频大多数在16MHz~66MHz之间，且一般没有浮点运算支持。这些嵌入式系统中往往通过软件仿真库完成浮点运算，速度慢、效率低。为了能够在无浮点处理能力的嵌入式系统中实现MP3音乐的实时解码和播放，本文提出了一种MP3定点解码算法，能够提高MP3的解码速度和效率，并且对音乐的播放质量没有太大影响。

本文提出的MP3定点解码算法已经在自主开发的以32位ARM7TDMI为核心的SoC系统上得到了实现，并在ARM仿真调试环境和FPGA开发板上分别进行了验证和测试，结果表明ARM7TDMI处理器使用定点解码算法完全符合音乐实时播放要求。

2 解码算法分析

MP3数据以帧为单元，每帧由帧首部、边带信息、编码数据和辅助数据等4个主要部分组成，每帧包括1 152个压缩PCM采样点数据。设MP3的PCM采样频率为sample_freq，则可由式(1)得出每帧的时间；假设MP3位速率为bit_rate，则可由式(2)求出帧的数据长度。

$$frametime = \frac{1152}{sample\_freq} \quad (1)$$

$$framelen = 144 \cdot \frac{bit\_rate}{sample\_freq} \quad (2)$$

CD音质的音乐一般为双声道立体声，采样频率为44.1kHz，位速率为128kbps，则可由式(1)和式(2)得每帧播放时间约为0.026 12ms，或为38帧/s，每一帧需要处理的数据长度约为418byte。或为了达到实时解码和播放的效果，要求MP3解码器处理MP3数据的时间必须少于MP3的播放时间，否则就不能实现实时连续播放。

MP3的解码流程框图如图1所示，可以大致分为两个阶段，即数据流控制阶段和数值计算阶段。

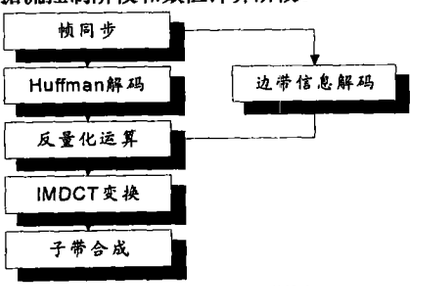

图1 MP3解码流程框图

数据流控制阶段包括帧同步、边带信息解码和Huffman解压缩等几个过程，其中Huffman解压缩是对编码数据进行操作，其他过程则是对帧控制部分进行操作。这几个过程以

基金项目： 2001年度江苏省科技招标资助项目“基于目标产品的SoC芯片设计及产业化”（BG2001069）
作者简介： 江 巍(1976—)，男，硕士，主研方向为嵌入式系统；杨 军，博士；罗 岚，讲师；胡 晨，教授
收稿日期： 2003-07-08 **E-mail：** jwei@seu.edu.cn

位操作为主，几乎没有数值计算，因此我们对这些过程进行了指令级优化，以提高其执行速度。

定点算法主要对数值计算阶段的过程进行算法上的设计和优化。这一阶段主要包括反量化、IMDCT变换、子带合成及其他中间过程。这一阶段数值计算量大，计算精度要求较高。

定点算法核心设计思想是使用定点数代替浮点数，并使用定点运算代替浮点运算。同浮点运算相比，嵌入式系统的定点处理能力较强，如ARM7TDMI可以支持32位定点整数的算术和逻辑运算，以及高达64位定点乘法和乘加运算，因此定点算法可以提高嵌入式系统的解码性能。

此外，为了进一步提高解码性能，我们还在定点运算的基础上又进行了数值算法的改进和优化。MP3解码过程中使用了大量比较复杂的计算函数，包括乘方运算、指数运算和三角函数运算等。这些函数难以使用定点算法实现，主要是因为定点算法的计算精度无法达到函数本身的要求，并且实时计算这些函数将消耗大量的CPU时间，降低解码速度。因此在本算法中选取一部分典型参数并预先计算出函数结果，构造典型数据查找表作为常量数组。实际进行计算时，只需要根据需要计算的函数检索相应的查找表，并对取出的典型结果进行线性插值即可求出近似值。使用查找表代替实时计算可以将大量的计算转化为内存访问，从而大大降低计算量并且减小误差，同时避免占用大量内存。

IMDCT变换可以根据帧类型分别进行MDCT36或MDCT12变换；子带合成可以转化为1次DCT64变换和32次向量乘法运算。使用快速DCT算法可以提高DCT的运算速度。而且ARM7TDMI支持32位和64位精度的定点乘法和乘加运算指令。这样可以在计算速度和精度间作出选择，即使用32位运算达到较高的计算速度或使用64位运算得到较高的计算精度。

在对MP3解码过程及所占用的CPU时间进行分析后发现，70%以上的CPU时间用于最后进行的IMDCT变换和子带合成运算；只有约20%左右的CPU时间用于完成其他数值计算过程，对解码速度的影响远小于最后两个过程。同时，由于IMDCT变换和子带合成运算需要使用其他数值运算过程的计算结果，因而从误差传递的角度考虑，前几个计算过程产生误差对最终PCM采样输出音质的影响要比最后两个计算过程的误差影响要大。从这两个角度分析可以得出一致的结论：反量化、立体声解码和反锯齿等数值计算过程的计算精度对PCM采样的输出音质有较大影响，但其运算速度对整个MP3解码的性能作用不大；而IMDCT变换和子带合成两个过程需要进行大量的乘法和乘加运算，其计算速度直接关系到MP3帧的解码速度，对MP3的解码性能影响极大。基于以上考虑，我们在前3个数值处理过程中使用速度较低但精度较高的64位乘法和乘加指令，牺牲计算速度换取较高的精度，以提高MP3的解码输出音质；而在IMDCT和子带合成两处过程中使用速度较快但精度较低的32位运算，以满足MP3解码的实时性要求。

3 设计实现

根据第2节的算法分析结果，我们使用C语言实现该MP3定点解码算法。

(1) 定点算法

由式(3)可以将浮点数表示成定点数值。其中I表示定点整数，受机器字长影响；F表示浮点数；N_F表示浮点数的扩大倍数；[]表示取整运算。显然，I的位数越多，则计算精度越高，但计算速度则会下降。

$$I = [F \cdot 2^{N_F}] \tag{3}$$

定点数值计算直接使用整数或长整数运算指令实现，需要注意的主要是防止操作数太大或大小导致的运算溢出，另外还需要防止乘积溢出。选择合适的N_F可以减少因溢出而产生的音爆现象。

(2) 4/3乘方计算

4/3乘方计算主要用于反量化过程中，所用查找表中只保存[0,127]之间的计算结果并用定点数表示，其他数值采用插值算法间接计算。设在查找表pow43中求输入值i的4/3乘方运算结果，当i取值范围在

1) [128,1023]之间，则令p=i div 8，q=p+1，r=i mod 8，分别对p和q查表并将得到的结果分别左移4位令为P和Q，由式(4)求出i的计算结果

$$\begin{aligned} P &= pow43[p] << 4 \\ Q &= pow43[q] << 4 \\ I &= [P \cdot (8-r) + Q \cdot r] >> 3 \end{aligned} \tag{4}$$

2) [1024,8191]之间，则令p=i div 64，q=p+1，r=i mod 64，分别对p和q查表并将得到的结果分别左移8位令为P和Q，由式(5)求出i的计算结果

$$\begin{aligned} P &= pow43[p] << 8 \\ Q &= pow43[q] << 8 \\ I &= [P \cdot (64-r) + Q \cdot r] >> 6 \end{aligned} \tag{5}$$

(3) 指数运算

对反量化运算中的2的指数运算进行分析，可以将指数运算转化为如式(6)和式(7)的形式，其中n为整数。

$$X = 2^{-\frac{210+x}{4}} \qquad x \in [-256, 118] \tag{6}$$

$$X = 2^{n+p} \qquad p = 0, \pm\tfrac{1}{4}, \pm\tfrac{1}{2}, \pm\tfrac{3}{4} \tag{7}$$

受32位字长限制，当x>-134时，X值过小，从而导致X的定点数值为0。因此为了提高计算精度，代码中仅用2^n的定点值参与计算，而将最终计算结果左移n位(即乘上2^n)。

(4) DCT变换

IMDCT变换过程中对长窗类型的帧使用MDCT36变换函数，对短窗类型的帧使用MDCT12变换函数，其中MDCT是种改进的DCT变换；在子带合成过程中，需要执行1次DCT64变换。MDCT的计算公式如式(8)。其中n为36或12。

$$x_i = \sum_{k=0}^{\frac{n}{2}-1} X_k \cos\left(\frac{\pi}{2n}(2i+1+\frac{n}{2})(2k+1)\right) \tag{8}$$

目前已经有多种MDCT快速算法可以快速完成MDCT/IMDCT运算。我们在代码实现中使用两次快速DCT变换完成MDCT变换。

4 验证

为了验证本文提出的定点算法及相应实现的性能和音质特性，我们分别在ARM模拟器和FPGA上进行了测试。

表1是在AXD Debugger模拟器上使用定点算法对一段CD音质的MP3音乐片断分别使用浮点算法和定点算法进行解码运算所得的性能比较。其中S周期指访问连续内存的时钟周期；N周期指访问非连接内存的时钟周期；I周期指执行内部功能并且无须访问内存的时钟周期。

由表1可见，在ARM平台上定点算法的实现平均比浮点算法快5.56倍，完全可以满足嵌入式系统的需要。

表1 浮点算法与定点算法性能比较

	浮点算法	定点算法
指令数	42 886 816	8 086 575
核心	65 870 468	11 415 443
S周期数	50 854 693	8 793 861
N周期数	11 454 429	1 858 162
I周期数	9 907 116	2 321 354
总时钟	72 216 237	12 973 377

对上述浮点算法和定点算法的解码结果进行分析，可以根据式（9）计算得到各自的信噪比，如表2所示。由于定点算法的运算精度受机器字长的限制，因此定点算法比浮点算法的信噪比低16dB。

$$SNR = 10 \log_{10} \frac{\sum_i PCM_i^2}{\sum_i \Delta PCM_i^2} \tag{9}$$

表2 浮点算法与定点算法音质比较

	信噪比SNR(dB)
浮点算法	76.0731
定点算法	60.3190

5 结论

使用定点整数算法对MP3音频进行解码运算在低端处理器上可以极大地提高运行速度，不需要额外的浮点运算单元或专用DSP设备的支持，仅使用通用CPU便可以完成解码、播放和显示整个过程，从而使运用低性能CPU处理多媒体数据成为可能，扩展了嵌入式手持设备的应用范围，同时降低了生产成本，提高了产品的竞争力。

参考文献

1 ISO/IEC 13818-3. Generic Coding of Moving Pictures and Associated Audio (Part 3: MPEG-Audio). 1997

2 Lee K, Park Y C, Youn D H. Software Optimization of the MPEG-Audio Decoder Using a 32-bit MCU RISC Processor. IEEE Trans.on Consumer Electronics, 2002,48(3)

3 ARM. ARM7TDMI (Rev3) Technical Reference Manual. 2000

4 Winograd S. On Computing the Discrete Fourier Transform. Mathematics of Computation, 1978

☆☆

（上接第27页）

中点后，向前调整若干字节，使中点稍微偏移，取在某一词条的起始位置。用这种改进的二分法检索次首字为 S_2 的词条，若不存在，则S_1 为单字；否则，继续检索。

设以S_1 S_2为首字和次首字的某词条$W=W_1W_2\cdots W_k$，若

$S_i=W_i$（i=1,2,…,r　r≤k）

称词条 W和字符串S 的匹配计数为 r，若 r=k，称 W与 S_1 $S_2\cdots S_k$ 完全匹配；若词典中不再存在词条 $W'=W_1W_2\cdots W_k\cdots W_{k+j}$能与字符串 S_1 $S_2\cdots S_k\cdots S_{k+j}$ 完全匹配,则称k为最大匹配计数。

直接匹配算法就是用词典中以S_1 S_2为首字和次首字的词条直接与 $S=S_1$ S_2 $S_3\cdots$ 匹配，寻找匹配计数最大且能与字符串S 完全匹配的词，该词即为切分结果。然后继续下一待切分序列 $S=S_2$ S_3 $S_4\cdots$。

在分词及分析文本结构的基础上，按照式(2)计算各个词的权重，并适当地选取向量维数(去除“噪声”)，形成文本向量 $D=(v_1, v_2, \cdots, v_n)$。

3.2.3 向量匹配

需求向量与文本向量的匹配采用余弦函数

$$sim(C,D)=\cos\theta=\frac{C\cdot D}{\|C\|\cdot\|D\|}=\frac{\sum_{i=1}^{n}u_i v_i}{\sqrt{\sum_{i=1}^{n}u_i^2\sum_{i=1}^{n}v_i^2}} \tag{3}$$

两向量之间夹角越小，其余弦值越大，说明相似程度越大，文本符合过滤需求的可能性增加。设定一个过滤阈值Ψ，当$sim(C, D)\geq\Psi$时，说明特征向量D所对应的内容符合过滤需求，应禁止在网络中传输和扩散。

3.2.4 长文本的过滤

根据上述方法建立的内容过滤模型不但可以提高过滤精度，而且对于一般长度的文本也能满足实时性要求。但若文本太长，过滤时间也会较长，这时，可以不过滤全文，而是根据文本的某些特殊区域能更好地表达文本主题的特点，从中选择标题、首段和尾段进行过滤。

选择标题、首段和尾段的特殊区域进行过滤，仍要提取向量，方法与从全文中提取向量的方法相同，只是词汇权重函数定义中的参数a 应有些变化，用以区分标题和其它部分的不同重要性。当词位于文本的标题位置时，为加大词权重取$a=0.5$，否则$a=0$。

文本的标题、首段和尾段等特殊区域不仅能在很大程度上表达文本的主题内容，而且选择它们代替全文进行过滤，可以大大缩短文本的实际操作长度，从而减少过滤时间，满足实时性要求。

4 结论

基于向量空间模型的内容过滤是基于全文的信息内容分析与控制技术，它与网站、URL的内容过滤、基于数据包的Web过滤以及邮件过滤网关技术不同，与使用关键词进行内容过滤的方法相比，在精度上也有很大提高，并且能在过滤精度与实时性之间较好地达成平衡，实用性很强，可应用于传输文本数据的各种应用级协议。

参考文献

1 Salton. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-westey, Reading, Mass, 1989

2 林鸿飞, 战学刚, 姚天顺. 文本结构分析与基于示例的文本过滤. 小型微型计算机系统, 2000-04

3 刘开瑛.中文文本中抽取特征信息的区域与技术.中文信息学报, 1998-02

4 揭春雨, 刘　源, 梁南元. 论汉语自动分词方法.中文信息学报, 1989-01

5 张国煊.快速书面汉语自动分词系统及其算法设计.计算机研究与发展,1993-01

6 Yan T W, Hector G M. Index Structures for Information Filtering Under the Vector Space Model. AD Report, 1993-11