G.723.1 语音编码器在 Blackfin 平台上的实时实现①

曾海平[1,2]　孙建伟[1]　吕　斌[3]　杨海波[1]

(1.中国科学院沈阳计算技术研究所 网络与通信实验室 辽宁 沈阳 110171; 2.中国科学院研究生院 北京 100049; 3.山东金岭铁矿 山东 淄博 255081)

摘　要： ITU-T G.723.1 是一种用于多媒体通信的双码率语音编码标准，几乎在所有的语音网关设备上面 g723.1 音频编解码器都是必须支持的一个标准编解码器。针对 G.723.1 音频编解码算法尚未在 BF532+uClinux 平台上实时实现的情况，基于 BF532+uClinux 平台提出了该算法实时实现的优化方案。方案减少了编解码的时延,降低了算法的复杂度，编解码整体性能提升约 10 倍，满足了 BF532+uClinux 平台的实时性要求，并全部通过 ITU 测试向量的测试。最后将优化好的 G723.1 编解码器应用到嵌入式语音网关中，实验表明语音通话效果良好。

关键词： 优化；G .723.1 语音编解码；BF 532；Blackfin；uClinux ；嵌入式语音网关；

Real-Time Implementation of G.723.1 Speech Coder Using BF 532

ZENG Hai-Ping[1,2], SUN Jian-Wei[1], LV Bin[3], YANG Hai-Bo[1]

(1.Shenyang Institute of Computing Technology, Chinese Academy of Science, Shenyang 110171, China; 2.Graduate University of Chinese Academy of Sciences, Beijing 100049, China; 3.Shandong Jinling Iron Mine, Zibo 255081, China)

Abstract: This paper presents a full duplex, real-time implementation of ITU-G.723.1 speech coder using the BF 532 DSP chip with the uClinux embedded operating system. An optimization method is proposed in order to reduce the total necessary cycle time and the algorithm redundancy consumed in real-time implementation. The effect of the overall codec performance is about 10 times better than the traditional coding method and meet the real-time requirements of BF532 + uClinux platform, and all pass the test with the ITU test vectors. Finally, the optimized G723.1 code is applied to the embedded voice gateway. The experiments show that the voice calls are good .

Keywords: optimization; G .723.1 speech coder; BF532；uClinux; embedded voice gateway;

1　引言

ITU-T G.723.1[1]能以 5.3kbps 或 6.3 kbps 的速率压缩语音。高速率编码器采用多脉冲最大似然量化(MP-MLQ)，低速率编码器采用代数码激励线性预测(ACELP)。G.723.1 可用于以较低速率压缩语音和多媒体应用中采用的其它音频信号，还可以用作 H.324/H.323 或语音网络(VoN)应用的一部分。它是国际电信联盟(ITU)制定的多媒体通信标准中的一

① 收稿时间:2009-11-25;收到修改稿时间:2010-01-11

个组成部分，在低码率率条件下对多媒体服务中的语音或其他音频信号进行压缩，有很好的音质保证。

目前，语音编解码算法 G.723.1 提供了低码率，以及高质量的重建语音信号。本文基于 BF532+uClinux 平台提出了 G.723.1 算法实时实现的优化方案。编解码整体性能提升约 10 倍，满足了 BF532+uClinux 平台的实时性要求，使其在语音网关上语音编解码方面有广泛的应用前景。

2 G.723.1 算法

G.723.1 语音编解码器是基于线性预测理论，采用合成分析、矢量量化等方法，以经过感觉加权后的残差信号能量最小为准则进行编码的。G.723.1 是一种双码率的语音编解码方案，对语音或音频信号进行 8KHz 采样，16bit 量化，可以提供 5.3kbps 或 6.3kbps 压缩数据输出；高比特率编码输出能提供很好的语音质量，低比特率编码输出在提供较好语音质量的同时，可以为其他数据服务提供较好的机动性和可行性。

G.723.1 算法以 30ms 的语音信号作为一个处理帧，其两种编码速率可以在每个处理帧的基础上进行切换。参考文献[1]详细介绍了 G.723.1 算法的编码和解码过程以及算法各个功能模块，其中 G.723.1 的编码过程如图 1 所示：

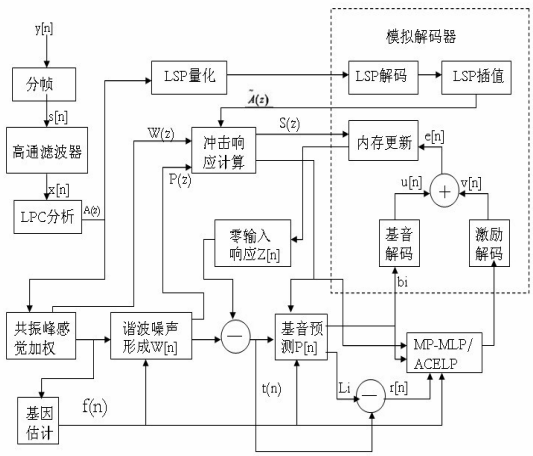

图 1　G.723.1 编码器功能框图

3 G.723.1 实时实现

G．723.1 的实时实现包含移植 ITU C 代码到 uClinux 平台上和依据 BF532 平台特性把移植好的 C 代码改写成高效的 DSP 代码这两部分。现基于 BF532+uClinux 平台提出了该算法实时实现的优化方案。具体方案实施中须使优化代码在 C 语言和汇编语言的使用量上寻找一个平衡。比如，执行控制操作的过程可以使用 C 代码，这样做的好处就是有很好的维护性；对信号处理的关键部位可以使用汇编或者嵌入式汇编实现。这样能极大的提高信号的处理能力。

因此，G.723.1 的优化分两步进行：一是将 ITU-T C 代码移植到 BF532 和 uClinux 平台上并完成算法的优化；二是为了充分利用 BF532 体系结构的优势，把音频编解码中信号处理的的关键部位进行汇编化处理。

3.1 G.723.1 算法级优化

在进行算法优化前，要分析各个函数的运算耗时，以确定优化重点应放在运算非常耗时的函数上。

利用 linux 平台下的 gprof 工具统计了各个函数占用的运行时间比率，统计结果如下：

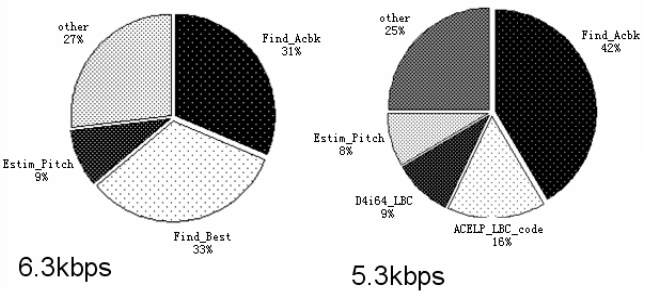

图 2　两种速率下算法核心函数计算量统计图

在 6.3kb/s 的速率下自适应码本搜索函数 Find_Acbk，固定码本搜索函数 Find_Best，基音分析函数 Estim_Pitch 这三函数占时间超过 73.1%。

在 5.3kib/s 速率下自适应码本搜索函数 Find_Acbk，固定码本搜索 ACELP_LBC_code，代数码本搜索 d4i64_LBC，基音分析函数 Estim_Pitch 四个函数占用时间达到 74.9%。因此，应该集中优化码本搜索和基音分析这几个模块。

3.1.1 自适应码本搜索的优化

在 G.723.1 语音编解码算法中，基音预测分布被看作传统的自适应码本分布。该算法使用一个 5 阶预测器作为基音预测器。因此问题就成为寻找最优基音预测器 G[k]和闭环基音标记，使基音预测器的分布 p[n]近似于目标矢量 t[n]，误差信号 r[n]的均方最小。即：

$$\min\{\sum_{n=0}^{59}(t[n]-p_{i,j}[n])^2\}$$

其中 *j* 是自适应码本的索引。对于子帧 *0* 和 *2*，在开环基音周期 OLP 附近±1 即(OLP-1，OLP，OLP+1)的范围内选择最优闭环基音周期 Lag。对于子帧 1 和 3，则在 OLP 的-1，0，+1 或+2 范围内选择最优闭环基音周期 Lag。基音预测分布的计算公式[1]如下：

$$p_{i,j}[n]=\sum_{k=0}^{4}G_j[k]y_{i,k}[n] \quad 0\le n\le 59$$

其中 $y_{i,k}[n]$是开环基音周期 $i=Lag$ 下，激励矢量 Tv[n]与组合滤波器的冲激响应 h(n)的卷积。G 是自适应码本增益，*j* 是自适应码本的索引，*k* 为 5 阶滤波器阶数。

$$y_{i,k}[n]=\sum_{t=0}^{n}Tv_i[k]h[n-t] \quad 0\le n\le 59$$

分析代码时发现：

```
Tv[i]=PrevExc[PitchMax-Lag-2+i];
Tv[2+i]=PrevExc[PitchMax-Lag+i%Lag];
```

式中的 PrevExc(n)为前一帧预测激励矢量。在闭环基音 *Lag*<62 的情况下，在获取过去的激励矢量时有一定的规律，推导如下：

```
Tv[2+Lag+i]=PrevExc[PitchMax-Lag+(i+Lag)%Lag]
           =PrevExc[PitchMax-Lag+i%Lag]=
Tv[2+i]
```

即 Tv[2+Lag+i]=Tv[2+i],出现了周期性，周期为 *Lag*。在自适应码本搜索计算 $y_{i,k}[n]$时，利用 Tv[i+Lag+2]=Tv[2+i]这个特性进行优化，当 *i*≥0 时，为了便于分析不妨设 *Lag*=2，有 Tv[*i*+4]=Tv[*i*+2]，所以有：

$$y_{i,k}(0)=Tv_i[k]h[0]$$

$$y_{i,k}(1)=Tv_i[k]h[1]+Tv_i[k+1]h[0]$$

$$y_{i,k}(2)=Tv_i[k]h[2]+Tv_i[k+1]h[1]+Tv_i[k+2]h[0]$$

由 Tv[*i*+4]=Tv[*i*+2]，所以有：

$$y_{i,k}(2)=Tv_i[k]h[2]+Tv_i[k+1]h[1]+y_{i,k}(0)$$

以此类推，有，

$$y_{Lag,k}(n)=Tv_i[k]h[n]+Tv_i[k+1]h[n-1]+Tv_i[k+1]h[n-2]+\ldots+y_{i,k}(n-Lag) \quad Lag\le n\le 59$$

通过上述优化，使得优化前完成一个子帧的卷积运算需要的运算为：1+2+3+.....+60=1830 次乘加运算，而优化后完成一个子帧卷积运算则需要 *Lag*/2+60**Lag*-*Lag***Lag*/2 次乘加运算，*Lag* 越小，优化越明显。

以上优化都是针对 *Lag*<62 时激励矢量 *Tv*[n]出现周期性变化来优化的。当 *Lag*≥62(n)时，激励矢量 *Tv*[n]将不具有周期性[2,3]。

3.1.2 固定码本搜索的优化

因为编码速率为 6.3kbps 跟 5.3kbps 相比，其使用的技术 MP-MLQ 比 ACELP 需要更多的时钟周期[1,4]；所以为了减少 G.723.1 的编码运算量，MP-MLQ 模块的优化势在必行。当仔细分析 MP-MLQ 模块时，发现只有输入激励的偶数相关系数是必需的，奇数的计算是可以移除。同时脉冲增益量化表没有零值，所以在码表搜索中直接设定为从固定最大激励值往最小激励值搜索,一旦找到最接近的量化因子就直接跳出循环，这样可以节省大量的循环计算量。

更进一步，根据最大似然特点，这脉冲位于最大相关值附近。因此根据这个特性，采用了一种搜索 MP-MLQ 激励简化方法。为了寻找这些似然值，这相关系数函数一阶导数和二阶导数如下：

$\frac{\partial d(t)}{\partial t}=0,\frac{\partial^2 d(t)}{\partial t^2}<0$ ，当 d(t)≥0时；

$\frac{\partial d(t)}{\partial t}=0,\frac{\partial^2 d(t)}{\partial t^2}>0$ ，当 d(t)<0时；

这里的 d(t)是冲击响应 h(t)和目标向量 v(t)之间的相关系数，M 值满足以下等式[5,6]：

$$G_{\max}=\frac{\max\{|d[j]|\}_{j=0\ldots59}}{\sum_{n=0}^{59}h[n]\bullet h[n]}$$

在离散区域内，一个简单的可以寻找候选位置方法：

首先，寻找满足下面表达式的索引值；同时记下当前互相关系数的相应绝对值。

(d(n)≥0&&(d[n]-d[n-1]≥0)&&((d[n]-d[n+1])≥0)

其次，选择 *M* 索引值，该值拥有相关系统第一个最大的记录值。一个合适 *M* 值为 10，候选位置为所选索引值的附近，因此，少于 30 个候选位置拥有第一次的脉冲位置。根据这些候选位置，只有这些大于候选位置值时，新的相关系数函更新才执行。因此，通过这个方法，算法的复杂度降低为原始方法的(30/60)*100%=50%。

3.1.3 FIR 和 IIR 过滤器优化

通过优化 FIR 和 IIR 过滤器，可以获得性能很大的提升。因为 G.723.1 编码器中，10 阶 FIR 和 10 阶 IIR 过滤器的不同组合被用于基因估算、零输入响应计算、冲击响应计算和过滤器内存更新。解码器在合成和共振谐波后置滤波器模块利用他们。在 G.723.1 中，FIR 和 IIR 过滤器的形式分别如下面两式[1]所示：

$$v_{out}[n]=v_m[n]+\sum_{k=1}^{10}a_k v_m[n-k],\qquad 0\le n\le 59$$

$$v_{out}[n]=v_m[n]+\sum_{k=1}^{10}b_k v_m[n-k],\qquad 0\le n\le 59$$

这部分减少算法复杂度方法是通过避免过滤器内存数据移动来实现的。通过在输入数组之前(用于 FIR 滤波器的情况下)或输出数组之前(IIR 滤波器的情况下)拷贝过滤器的内存。由于 FIR 过滤器内存是在新的输入样本时候更新，并且 IIR 过滤器是在新的输出样本时候更新，当使用指向内存的指针的后置增量后，过滤器内存拷贝可以删除。这时当过滤完成后，最后 10 个样本被复制回过滤内存。通过这部分优化，这部分算法的复杂度进一步减少约 45MIPS。

因此，几个主要的算法模块都通过一系列的分析和优化，最后算法复杂度极大限度的降低了，测试结果显示仅为 G.723.1 算法原来复杂度的 15%。

3.2 程序实现级优化

本项目中采用方案是集中处理运算复杂度比较高的模块；并且采用的方法是 C 和汇编混合编程模式；这样比较节省软件开发时间，同时也方便代码维护，并且执行效率跟全汇编优化相比相差不大。

下面就本项目用到的几个简单且效果相当明显优化方法进行介绍.

(1)针对大量数据移动部分，本项目采用指针代替数组，使得数据处理效率提升一倍。

在数组中，每次循环中都必须计算下一次索引值，在编译器中可以看出每次赋值占用 4 个指令周期。而指针指向数组时，每次循环只需要指针进行增量操作，每次赋值只占用 2 个指令周期。

(2)使用 Blackfin gcc 内联函数。

在分析 G.723.1 代码中，发现一些基本操作的调用次数和运算量非常巨大。如下图所示，如果未处理这些基本操作，这几个基本操作中将占总的编码时间的 61.81%。因此对这些基本操作优化势在必行。

表 1 G.723.1 算法基本操作调用次数和计算量统计图

基本操作	函数调用耗时	调用次数	占用编码时间
L_mult	4.68s	30529630	28.21%
L_add	2.48s	28255582	14.98%
L_mac	2.27s	21904010	13.65%
L_shr	0.82s	59521503	4.97%
			61.81%（total）

本项目采用 Blackfin gcc 内联汇编的方法，这些函数都是非常的小，采用内联汇编带来的空间大小影响非常小，但是效率提升却是非常明显的。优化后编码效率整体提升了近 4 倍。

(3)循环处理。

循环语句在 G.723.1 中出现的频率还是非常高的。对其进行编码优化能带来的性能极大提升，这里主要采用两个方法：一个是循环展开；循环展开指将循环展开，减少循环迭代次数，这是一种简单而高效的优化方法。另一个是用内嵌汇编指令方法；由于 ADI 汇编自带有循环指令[6]，这些指令在处理简单循环时，能在常数时间内完成。

(4)软件流水化。

软件流水化一种重要的指令调度技术[7]，它通过重叠地执行不同的程序体来提高指令的并行性。充分运用软件流水来优化代码，能提高程序的运行效率。比如 Blackfin 系列处理器是基于一个 10 级 RISC MCU/DSP 流水线和一个专为实现最佳代码密度而设计的混合 16/32 位指令集架构。采用流水技术，可以使得多个功能单元并行运行。

(5)面向汇编的代码优化

G.723.1 算法中有些基本操作非常耗费计算量，诸如 L_mult，L_add，L_mac 等，可以将这些函数全部用汇编实现。同时 Blackfin 系列 DSP

处理器本身也提供了 L_mac 函数的指令[6]，诸如 Find_Acbk，Find_Best 等这些核心函数需要大量的 L_mac 函数调用，采用的方法是将这些核心函数完全用汇编代码编写，这样直接调用 MAC 指令，并同时将一些操作并行化处理，因此能极大地加快程序的执行

3.3 向量测试

由于G.723.1 代码的优化项目必须对代码进行大量改动，为了更好优化代码，又要保证代码准确度。这就需要不断地测试优化代码各个编解码分支的准确性。利用 ITU-T G.723.1 自带的测试向量能非常好的完成此项任务。所以本项目在不断地优化过程中，我们不断用测试向量进行代码检测，以确保代码的准确性。本项目最后优化代码都能完全通过 G.723.1 自带的测试向量的测试。

3.4 G.723.1编解码优化结果

通过上面优化方案对G.723.1 编解码器进行代码优化后，编解码整体性能提升了约 10 倍。其测试结果如下：

表 2　G.723.1 代码优化前后运算量 MIPS 对比图

测试数据	数据帧数	编码速率 (kbps)	编解码	优化前 (MIPS)	优化后 (MIPS)
PATHC63H.TIN	1019	6.3	编码	234.00	24.10
PATHC53.TIN	1028	5.3	编码	195.00	15.00
PATHD53.TCO	4	5.3	解码	67.00	6.5
PATHD63P.TCO	100	6.3	解码	163.00	21.17

表 3　G.723.1 代码优化前后帧时延对比图

测试数据	数据帧数	编码速率（kbps）	编解码	优化前帧时延（ms）	优化后帧时延（ms）
PATHC63H.TIN	1019	6.3	编码	109.315	11.179
PATHC53.TIN	1028	5.3	编码	80.000	7.097
PATHD53.TCO	4	5.3	解码	0.000	0.000
PATHD63P.TCO	100	6.3	解码	7.287	1.267

如上表所示，基于 BF 532+uClinux 平台语音网关上实现 20 路的语音通话是完全可行的，并且较低的时延能保证非常好的通话效果。

4　结论

基于 BF 532 的硬件平台和 uClinux 软件平台，实时实现了 ITU G.723.1 双速低码率语音编解码算法。最后将优化好的 G723.1 编解码器应用到嵌入式语音网关中，实验证明语音通话效果良好。为语音通信提供了语音编解码实时实现的解决方案，对于互联网语音通信的发展具有实际意义。

参考文献

1 Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbps, ITU-T Recommendation G.723.1, March, 2006.

2 乔光云,石翠仙等.G.723.1 语音编解码优化方案.电子测量技术, 2007,30(7):125－127.

3 明芳.嵌入式系统程序代码优化方法.计算机与数字工程, 2006,34(10):189－192.

4 Sung-Kyo Jung, et al. A Fast Adaptive-Codebook Search Algorithm for G.723.1 Speech Coder. IEEE Signal Processing Letters, 2005,12(1):75－78.

5 汪国有,段敏涛.基于定点 DSP 的 G.723.1 语音编码器的实时实现.计算机与数字工程, 2006,34(1):107－110.

6 Blackfin R. Processor Programming Reference, Revision 1.3, USA:Analog Devices, IncLSeptember 2008.

7 周为宽,吴百锋.ACELP 在 RISC 处理器上的移植与实时性化.计算机应用与软件, 2008,25(9):236－238.