幸存路径存储和输出单元的低功耗设计

段华蓉
(重庆大学 通信工程学院,重庆 400030)

摘 要:Viterbi译码器中幸存路径存储管理一直沿用两种传统方法——寄存器交换法和回索法。寄存器交换法内连关系过于复杂,而且功耗较大;回索法需采用大量额外存储单元作为缓冲,译码延迟亦较大。文中对传统的寄存器交换法进行了一些改进,减少了芯片使用面积,同时减少内存的存取次数,达到了降低功耗的目的。

关键词:Viterbi译码器;幸存路径存储管理;寄存器交换法
中图分类号:TP391 **文献标识码**:A **文章编号**:1673-629X(2006)12-0142-02

Low Power Design for Survivor Memory Unit

DUAN Hua-rong
(College of Communication Engineering, Chongqing University, Chongqing 400030, China)

Abstract: There are two traditional techniques for the realization of survival memory unit in Viterbi decoder—register exchange(RE) and trace back (TB) method. RE has a very complicated interconnections and needs a high power consumption. TB needs a large quantity of buffers and has long decoding delay. In this paper a modified register-exchange (RE) method is presented, which reduce its memory access rate and its amount of memory, thus, reduces the power consumption.

Key words: Viterbi decoder; survival memory management; register exchange

0 引 言

卷积码是一类重要的前向纠错编码,由于编码简单,易于实现最佳译码[1],在第二代移动通信系统和第三代移动通信系统中都得到了应用。Viterbi译码算法是一种用来解卷积编码的最大似然译码算法,它具有译码效率高、速度快及译码器实现结构简单的优点,被认为是卷积码的最佳译码算法[2,3]。但是Viterbi译码算法的复杂度、计算量和存储空间会随着卷积码约束长度呈指数倍增长,功耗也很大,在卷积码约束长度为7的第二代移动通信系统中,Viterbi译码器大约耗费了整个移动modem 1/3的能量[4],因此,对于电池供电的移动终端,降低Viterbi译码器的能量消耗是非常重要的。Viterbi译码器主要由分支度量计算单元BMU(branch metrics unit),加比选单元ACS(add-compare-select unit),路径度量存储单元PMU(path metrics unit),幸存路径存储及输出单元SMU(survivor memory unit)组成。其实现的结构框图可用图1表示。

在SMU中,由于要进行频繁的存储器读写,功耗很大,成为整个Viterbi译码器中发热最厉害的单元,为了降低Viterbi译码器的功耗,文中对SMU进行了优化设计。

收稿日期:2006-02-27

作者简介:段华蓉(1971-),女,四川德阳人,硕士研究生,研究方向为信号与信息处理;导师:朱冰莲,副教授,研究方向为信号与信息处理。

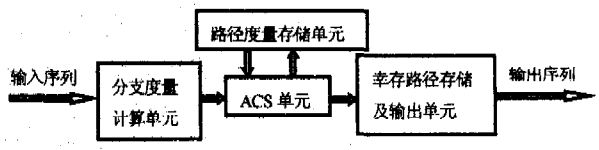

图1 Viterbi译码器结构图

寄存器交换法(RE,Register Exchenge)与回索法(TB,Trace Back)被认为是SMU的两种传统实现方法[5]。

寄存器交换法采用专用寄存器作为存储主体,存储的是路径上的信号信息,利用数据在寄存器阵列中的不断交换实现信息的译码。RE方法概念简单清楚、不需要回索、译码速度快,但有两点不足:①内连关系过于复杂,成为占据VLSI芯片面积的主要因素并影响芯片速度。②在每一个判决周期,寄存器阵列中的每一个寄存数据都要发生交换,造成很大功耗。而解决芯片散热问题又将给制造带来过高成本。

回索法则采用通用的RAM作为存储主体,存储的是幸存路径的格状连接关系,通过读写RAM来完成数据的写入和回索输出。其优点是内连关系简单、规则,但也有两点不足:①该方法中幸存路径存储器所存储的内容既包括路径的信息又包括前序状态的指针,因而需要大量的存储空间。②每输出一个译码结果需要回索整个译码深度,译码延时较长。

从上述分析可知,传统的RE法和TB法都存在不足,不能同时满足移动终端高速低延时低功耗的要求。文中对传统的寄存器交换法进行了一些改进,在不影响性能的

前提下减少寄存器的使用数量,同时减少内存的存取次数来达到降低功耗的目的。

1　传统的寄存器交换法

本节以(2,1,9)卷积码为例,分析用寄存器交换法进行幸存路径存储及输出的过程。

由于幸存路径存储器读出和写入的地址不同,因此设计两个状态存储器,记为 RAM0 和 RAM1。每个存储器有 256 个单元,对应 256 个状态,每个单元为 40 比特,满足 T=5·m 的截尾译码原理要求。

从 ACS 单元得到幸存路径的判决比特 dec,它是一个 256 比特的向量,每一个状态对应 1 比特,寄存器交换单元根据 dec 的值判断前一个状态。对状态 i 而言,如果 dec[i] = 0,则前一状态来自上支路,可推知前一状态为 $i/2$;如果 dec[i] = 1,则前一状态来自下支路,可推知前一状态为 $i/2+128$。

如果 i 为偶数,则当前时刻源码输入是 0;如果 i 为奇数,则当前时刻源码输入是 1。

两个存储器之间按蝶形关系进行交换,通过一个控制信号 CTRL 来控制。当 CTRL=0 时,把 RAM1 的值写入 RAM0,然后把 RAM0 的最后一列作为译码结果输出;当 CTRL=1 时,把 RAM0 值写入 RAM1,然后把 RAM1 的最后一列作为译码结果输出。

2　改进的寄存器交换法

从上节的分析可知,当 Viterbi 译码器的约束长度为 9 时,幸存路径存储单元需要两个 256×40 的存储器阵列;在每一个判决周期,存储器阵列中的每一个数据都要发生交换,占用芯片面积和功耗都十分惊人。有没有方法对它进行改进,减少寄存器的使用数量,或减少寄存器的数据交换次数,这正是文中要研究的问题。

Viterbi 算法可以用网格图来表示,它是一个时间序列的状态图。一个简单的表示 2 个状态的状态转移网格图如图 2 所示。

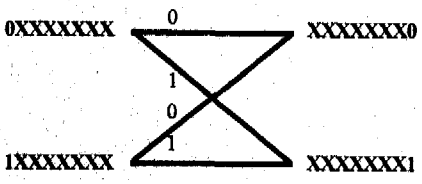

图 2　(2,1,9) Viterbi 译码器状态转移图

在该图中的 $t-1$ 时刻,2 个可能状态根据输入的信息可能转移到 t 时刻的 2 个状态。

对同一组前序状态 0XXXXXXX 和 1XXXXXXX,设为 i 状态和 $i+128$ 状态($i<128$)向后序状态转移,根据 ACS 单元的判决结果有两种情况:①i 和 $i+128$ 状态都只向一个后序状态转移;②i 和 $i+128$ 状态中有一个向两个后序状态转移,另一个不向后序状态转移。

对于所有的 256 个状态,有两种极端的情况。一是所有状态都只向一个后序状态转移,对于这种情况,如果设计两组状态寄存器进行内存交换实在没有必要,只要为每一个状态设计一个 8 位的存储器,称为指针寄存器,指示寄存器所代表的状态,则只需要改变这些指针寄存器的内容,就完成了状态的转移。二是每一组前序状态中,只有一个状态向后序状态转移,另一个不转移。对于这种情况,有一半的寄存器需要改变内容和对应的指针寄存器的值,还有一半的寄存器只需要改变对应的指针寄存器的值。可见寄存器的交换次数至少可减少 50%。

根据以上的分析知道,改进寄存器交换法,减少存储器使用数目和减少内存交换次数是可能的,下面讨论改进的寄存器交换法 SMU 单元的具体方法。

对于状态寄存器,由于新的判决结果要移入,最高位要移出作为输出结果,所以采用移位寄存器;对于指针寄存器,状态数的变化也是十分有规律的,前序状态 i 和 $i+128$,可能转移到 $2i$ 和 $2i+1$ 状态,采用移位寄存器,左移时只需移入 0 或 1 即可,移出的高位丢弃。所以指针寄存器也需要采用移位寄存器。

传统的寄存器交换法,判断一个状态转移自哪个状态,有两种情况,用 1 比特就可表示,为 0 时来自上支路,为 1 时来自下支路。文中改进的寄存器交换法,判断一个状态转移向哪个状态,有 4 种情况:不向任何状态转移,用 00 表示;只从上支路转移,用 01 表示;只从下支路转移,用 10 表示;向两条支路同时转移,用 11 表示。可见需用 2 比特才能表示。

改进的方法增加了 256×8 比特的指针寄存器,判断比特又多了 256 比特,但只用了一组状态寄存器,总存储空间只有改进前的 60%左右。

改进后控制要复杂一点,但是也很有规律。

如果每一个状态都只向一个后序状态转移,这种情况最简单,状态寄存器内容不用交换,只需根据判决比特对指针寄存器和状态寄存器的内容左移,移入的比特由判决比特决定。

如果一个状态同时向两个状态转移,那么与它同组的状态就不向任何状态转移,这时,需要把判决比特为 11 的状态寄存器内容复制到判决比特为 00 的状态存储器中,同时把判决比特一个改为 01,一个改为 10,然后再进行状态转移。

可见改进后增加了一个判断和寄存器内容复制的过程,不过并不复杂。

改进的寄存器交换法 SMU 结构如图 3 所示。

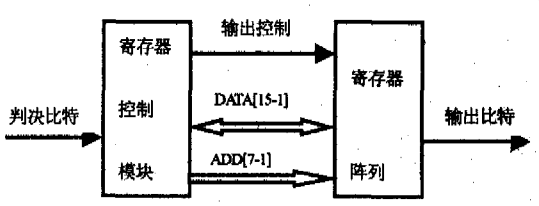

图 3　改进的寄存器交换法 SMU 结构图

(下转第 147 页)

从图 1 中可以清楚地看到离群值,离群样本点为样品 12 和 17。关注这两个样品发现,它们的扇出、模块控制流路径值都较大,因此需要对这两个异常样品进行调查分析,从技术因素和人为因素着手,查找引起异常的原因,及时发现潜在问题并作适当调整和改进,以避免在后续开发中带来不必要的损失。图 2 是提取主成分之前,标准化数据集中 X_2(扇出)、X_3(扇入)、X_4(控制流路径)这三个属性的三维三点图,和图 1 相比较,离群值和数据密集部分均不如图 1 明显。在此主成分分析实验中,不仅去除了数据的相关性,而且使数据得到了降维,由四个属性简化为三个主成分,最后由主成分得到的散点图更加清晰地呈现出离群值较大的样品。

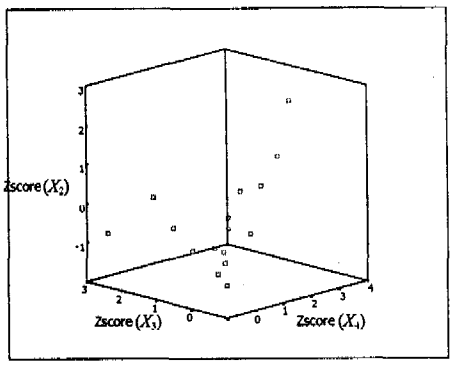

图 2　扇入、扇出及路径的散点图

3　小　结

主成分分析的应用可以减少在软件度量过程中所收集的数据信息在一定程度上的重叠,降低计算量和分析问题的复杂性。因为软件开发中的众多因素之间可能存在一定的相关性,在对某一问题进行研究时很有可能对某些因素的效用进行了重复累加,这样会导致度量结果不能真实地揭示软件开发过程中存在的问题,也就不能进行准确的预测和有效的决策[6]。利用主成分分析,可以找出影响某一软件过程的几个综合指标,使综合指标为原来度量属性的线性组合。综合指标不仅保留了原始度量属性的主要信息,彼此之间又不相关,这样就保证了在软件度量的大量数据中容易抓住问题的主要方面。

参考文献:

[1] Fenton N E, Pfleeger S L. Software Metrics: A Rigorous & Practical Approach[M]. 第 2 版. 北京:清华大学出版社, 2003.

[2] Pillai K, Nair V S S. Statistical analysis of nonstationary software metrics[J]. Information and Software Technology, 1997, 39:363 - 373.

[3] Fenton N E, Neil M. Software metrics: successes, failures and new directions[J]. The Journal of Systems and Software, 1999, 47:149 - 157.

[4] 袁　卫,庞　皓,曾五一. 统计学[M]. 北京:高等教育出版社,2000.

[5] 王　芳. 主成分分析与因子分析的异同比较及应用[J]. 统计教育,2003,56(5):14 - 17.

[6] Maxwell K D, Kusters R J. Software project control and metrics[J]. Information and Software Technology, 2000, 42:963 - 964.

(上接第 143 页)

控制模块包括 256 个指针寄存器,每一个存储器对应一个状态寄存器,指示状态存储器所代表的状态。

从 ACS 单元输入 256 比特的判决信号,控制模块根据这些判决信号,判断每一个前序状态怎样向后序状态转移,形成新的判决信号:不向任何状态转移,用 00 表示;只从上支路转移,用 01 表示;只从下支路转移,用 10 表示;向两条支路同时转移,用 11 表示。

如果一个状态同时向两个状态转移,那么与它同组的状态就不向任何状态转移,这时,通过控制信号把判决比特为 11 的状态寄存器内容复制到判决比特为 00 的状态寄存器中,同时把判决比特一个改为 01,一个改为 10。

最后,根据新的判决信号,为 01 时,状态寄存器和指针寄存器移入 0;为 10 时,状态寄存器和指针寄存器移入 1。

3　结　论

幸存路径存储及输出模块是 Viterbi 译码四大组成模块之一,文中提出的改进的寄存器交换法和传统的寄存器交换法相比,具有占用资源少、存储器访问次数少的特点,降低了功耗,加上寄存器交换法本身速度快,适合在移动通信系统上使用。

参考文献:

[1] 王新梅,肖国镇. 纠错码原理与方法[M]. 西安:西安电子科技大学出版社,2001.

[2] Viterbi A J. Error bounds for convolutional codes and asymptotically optimum decoding algorithm[J]. IEEE Trans Inf Theory, 1967, IT - 13(2):260 - 269.

[3] Forney J G D. The Viterbi algorithm[J]. Proc IEEE, 1973, 61(3):268 - 278.

[4] Kang I, Jr Willson A N. Low - power Viterbi decoder for CDMA mobile terminals[J]. IEEE J. Solid - State Circuits, 1998, 33:473 - 482.

[5] Wicker S B. Error Control Systems for Digital Communication and Storage[M]. Englewood Cliffs, NJ: Prentice - Hall, 1995.