中图分类号: TN919.81　　文献标识码: B　　文章编号: 1007-7022(2007)24-2304-04

基于 BSP-16的 H.264熵编码器的实现

□陈　路, 石江宏(厦门大学, 福建 厦门 361005)

摘　要: 为提高在 DSP上实现的 H.264编码器效率, 利用 Equator BSP-16处理器特有的协处理器(VLx), 提出了一种并行处理熵编码(CAVLC)的方法, 最终实现了 CIF格式视频的实时编码。实验数据表明, 该方法提高了编码器的效率, 达到了较好的效果。

关键词: H.264; BSP-16; 熵编码; CAVLC; VLx协处理器

Implementation of the H.264 Entropy Encoder Based on Equator BSP-16

□ CHEN Lu, SHI Jiang-hong
(Xiamen University, Fujian Xiamen 361005, China)

Abstract This paper, in view of enhancing the efficiency of H.264 encoder based on DSP, uses the unique co-processor (VLx) of Equator BSP-16 processor to put forward a parallel processing method of entropy code (CAVLC), and finally reallizes the real-time encoding of CIF. The experiment results show that, this method has a good performance in enhancing the encoder efficiency.

Key words H.264; BSP-16; entropy encoder; CAVLC; VLx co-processor

1　引言

ITU-T于 2003年正式颁布了 H.264标准。作为继 MPEG-4和 H.263之后的新一代视频压缩编码标准, H.264凭借其优秀的压缩性能及良好的网络亲和性, 在数字电视广播、视频实时通信、网络视频流媒体传递尤其是无线网络下的多媒体应用上发挥了重要作用。相比于 MPEG-4的灵活的技术特点, H.264更加着重于提高压缩效率和传输的可靠性。在较低的码率下, H.264能保持较高的图像质量, 而在相同的图像质量下, 比 MPEG-4降低约 50%的比特率。然而性能上的提升带来的是算法复杂度的倍增, 因此, 如何通过优化 H.264算法和有效地利用 DSP的资源实现 H.264的实时编码是当今研究的热点。

H.264编码器的功能组成如图 1所示。

由图 1可以发现, H.264的熵编码模块独立于整个编码流程的循环之外, 去除编码后的数据冗余, 完成最终的码流输出工作。由于熵编码按位输出的处理方式, 对于主要进行并行多位数据处理的 DSP来说并不能达到很好的运行效率, 因此在优化的时候我们考虑能否采用一种适宜于位操作的处理器, 以并行的方式, 在 DSP进行宏块数据处理的同时, 协同完成码流的输出, 从而提高编码的效率和速度。Equator公司开发的

基金项目: 厦门市高校创新项目(3502Z20063003)。

作者简介: 陈　路(1983-　), 男, 硕士研究生, 研究方向为视频压缩编码和无线通信, E-mail: chenlu0729@163.com; 石江宏(1968-　), 男, 讲师, 研究方向为无线通信。

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

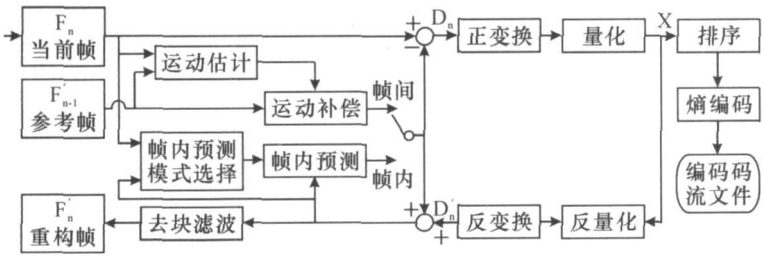

图 1　H264编码器框图

宽带信号处理器 BSP－16正好给我们的这种思路提供了解决方案, 其特有的 VLIW 主处理器加 VLx协处理的工作方式, 充分发挥了处理不同类型数据时处理器的特点, 使我们得以高效地完成编码工作。

2　CAVLC

2 1　熵编码的基本原理

熵编码是无损压缩编码算法, 设信息源 X 可发出的消息符号集合为:

$$A = \{a_i \mid i = 1, 2, \cdots, m\}$$

并设 X 发出符号 a_i 的概率为 $p(a_i)$, 则定义符号 a_i 出现的自信息量为:

$$I(a_i) = -\log_2 p(a_i)$$

如果各符号 a 出现是独立的, 那么 X 称为无记忆源, 对 X 的各符号的自信息量取统计平均, 可得平均信息量:

$$H(X) = -\sum_{i=1}^{m} p(a_i)\log_2 p(a_i)$$

$H(X)$称为信息源 X 的熵 (entropy), 也就是 X 发任意一个符号的平均信息量。当信源中各事件是等概分布时, 熵具有极大值, 而信源的熵与其最大值的差值则反映了该信源的冗余度, 信源的冗余度越小, 每个符号携带的信息量就越大, 就可以用较短的序列来传送。

在 H. 264基本档次 (Baseline Profile)中, 采用 CAVLC(基于上下文自适应的可变长编码)对残差块量化后的系数进行编码去除统计冗余, 而对其他编码单元使用指数哥伦布 (Exp-Golomb)变长编码。

2 2　CAVLC基本原理

CAVLC用于亮度和色度残差数据的编码, 通过参考已编码句法元素的情况, 动态调整编码中使用的码表, 减少了数据中的冗余信息, 为 H. 264卓越的编码效率奠定了基础, 其编码的过程如图 2所示。

2 3　指数哥伦布熵编码

指数哥伦布熵编码是指有规则构造的变长码字, 其构造方式为: [M 0] [1] [INFO]

INFO 是携带信息的 M 位域, 每个哥伦布码字的

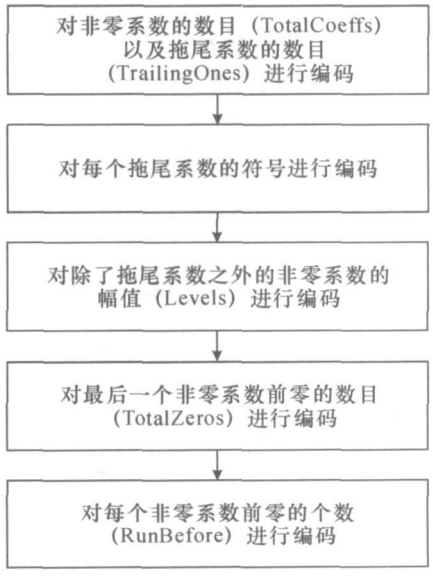

图 2　CAVLC编码流程

长度为 $(2M + 1)$比特, 根据每个码字的索引 code_num, 具体为码字构造如下:

$$M = floor[\log_2(code\_num + 1)]$$

$$INFO = code\_num + 1 - 2^M$$

3　BSP－16处理器及熵编码实现

BSP－16是 Equator在 2005年底发布的第五代视频处理器, 其主运算引擎: VLIW (Very Long Instruction Word)超长指令字处理器, 拥有 4条流水管线, 可以同时并行运行 4条指令, 并且加入了 SIMD(Single Instruction Multiple Data)单指令多数据流技术, 允许将多个 8、16、32或者 64比特的操作数并作 1个 128比特的指令字处理。此外, BSP－16还包括了 2个专为视频信号处理设计的协处理器: VLx与 Vf。VLx用来完成可变长编码, Vf则用于实现视频滤波。针对视频处理中需要传递的大量数据, 还使用了一个高性能的 DataStreamer(64通道 DMA引擎)来完成缓存中数据在不同的片上存储器子系统或外接存储器与 I/O间的数据传送。这样设计, 极大地拓宽了并行空间, 从而对视频信号的实时处理提供了有力的支持。

3 1　VLx协处理器

可变长编解码协处理器 VLx是一个 16 bit的 RISC协处理器, 拥有 32个 16 bit寄存器, 1个用于码流处理的 GB(GetBit)引擎, 以及 8 k的 VMem片上存储空间, 其结构如图 3所示。

VLx针对熵编码中大量使用的位操作和查表操作做了专门的优化, 而在与 VLIW 进行数据通信时, 可以采用内存直接访问或是使用 GB引擎两种方式, 前者

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

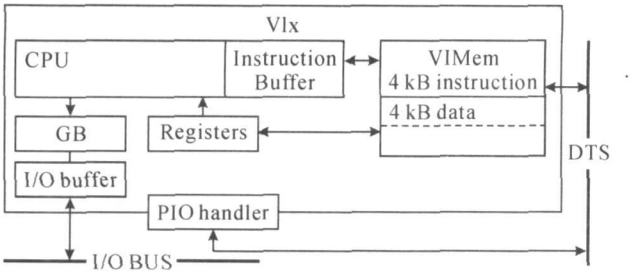

图 3　Vlx协处理器结构图

适合于数据块的传递, VLIW 可以通过直接访问 VIMem 来完成, 后者则是针对比特流输入/输出方式设计的一个高速的输入输出通道, GB引擎中置有 1个 112 bit的输入缓冲区和 1个 96 bit的输出缓冲区, 并有相应机制对缓冲区的状态进行监控处理, 以保证码流处理时的高效。

3 2　熵编码 (CAVLC)的实现

在设计中, 首先对 VIMem 中的数据区进行规划, 在有限的 4 kB空间中, 需要存储 CAVLC中用到的码表, 开设输入缓存以及与 VLIW 通信需要用到的命令通道缓存。

码表内的数据需要进行优化, 以提高查表速度, 压缩存储空间。表内数据包含有比特信息 (bits)和位宽信息 (size), VLx存储器为 16 bit, 表内数据存储如图 4所示。

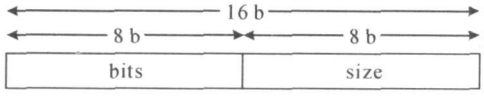

图 4　数据存储结构

之后需要确定与 VLIW 之间的数据通信方式。

对于数据输入, 采用 SDRAM - VIMEM 的方式, 如图 5所示。为了提高通信效率和可靠性, 在数据区 data0和 data1分别开设 2个输入缓存, 大小根据输入残差数据的最大值开设, 通过 DS传递数据, 此外增加了 sync同步标志位来标明当前缓存状态, 以避免数据被错误写入。

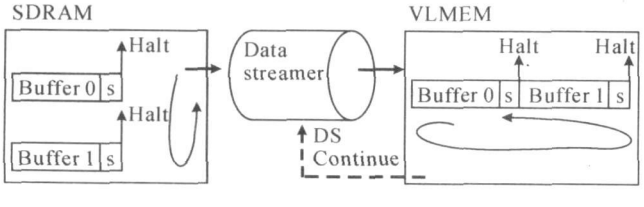

图 5　数据输入

而在输出的时候使用 GB- SDRAM 的方式, 如图 6所示, 通过调用 VLx库函数 VLxGbSplice(bits, size), 使用 GB引擎的 GB out通道输出码流, 因为当 GB引擎的输出满 32 bit时, 码流才从 GB out缓存中输出到 SDRAM, 所以一帧的数据结束后需要通过刷新 GB out缓存以保证所有的码流得以输出。

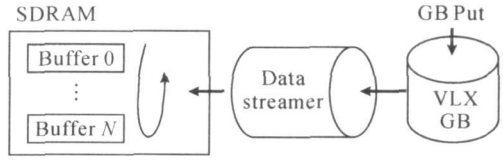

图 6　数据输出

VLx的熵编码过程是一个与 VLIW 同步处理的过程, 其协同关系如图 7所示。

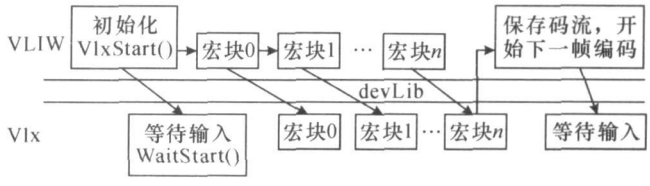

图 7　VLx- VLIW 协同关系

可见, VLx与 VLIW 通过驱动库函数 (devLib)实现命令交互, 在每一帧的码流中, 除了首尾 2个宏块外, 当 VLx进行宏块 n 的熵编码时, VLIW 就已经转入到宏块 $n+1$ 的处理工作中。在最后一个宏块的熵编码结束后, 将一帧的数据合并保存, 转入下一帧的编码。相比传统的串行编码方式, 这样在每一帧的编码过程中节省了 $n-2$个宏块的熵编码时间。

对于使用 VLx协处理器的实际效果, 我们通过编码测试序列 foreman cif 300帧, I:P = 1:100来进行分析。

首先使用 BSP提供的 egprof工具分析未经优化的 x264代码, 也就是使用串行的方式完成熵编码, 其中 CAVLC模块数据如表 1所示。

表 1　未优化的数据

函数名	调用次数	时间 (s)	占用总时间 (%)
x264_macroblock_write_cavlc	1 757 124	0. 97	2. 35
block_residual_write_cavlc	12 379 786	5. 34	12. 94

可见使用协处理器前 CAVLC总共占用了编码总时间的 15. 29%。

由于优化后的代码性能无法通过 egprof分析, 我们通过开关 CAVLC模块分别测试编码总耗时, 将测得时间相减得到的时间差来间接说明 CAVLC在编码

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

总时间中占用的比例, 如表 2所示。

表 2　优化后数据

开启 CAVLC (μs)	关闭 CAVLC (μs)	差值 (μs)	CAVLC占用 (%)
8 072 406	7 786 815	285 591	3. 54

可见采用协处理器后 CAVLC 占用编码总时间仅为 3. 54%, 并且降低了 VLIW 的负担, 提高了整个编码器的效率。

实验数据表明, VLx对宏块的熵编码处理要快于 VLIW 的宏块编码速度, 熵编码已经不是影响编码速度的主要部分, 进一步的优化工作重点应该放在提高 VLIW 编码的效率上。另一方面, VMem 的 8 kB 寄存器中, 指令区已使用 3 924 B, 数据区已使用 4 088 B, 使用率分别达到 95. 8%和 99. 8%, 如果想要进一步实现更加复杂的熵编码算法, 如 CABAC, 那么 VMem 的容量将会成为"瓶颈", 虽然 BSP－16支持分次载入 VLx代码, 但是势必带来编码器复杂度的提高和效率的降低。

4　结语

最终在 EquatorBSP－16平台上实现的基于 Linux 系统的 H. 264编码器中, 得益于本文提到的熵编码优化模块以及指令集优化, DMA 通道的有效使用, 实现了 CIF大小图像的 H. 264实时编码, 平均每帧图像处理时间约为 31 ms, 帧速率为 32. 3 fps。

参考文献:

[1] Iain E. G. Richardson 视频编解码器设计: 开发图像与视频压缩系统 [M]. 欧阳合, 韩　军, 译. 北京: 国防科技大学出版社, 2005.

[2] 毕厚杰. 新一代视频压缩编码标准——H. 264/AVC[M]. 北京: 人民邮电出版社, 2005.

[收稿日期: 2007-05-15]

自装硬盘播出系统应用实践

□廖永平 (黎平县文体广播电视局, 贵州 黎平　557300)

黎平县有线广播电视台 2004年投资 13多万元, 购买了 1套计算机硬盘播出系统。由于资金紧缺, 当时只配备了 1台单播机, 仍与原来的磁带播放机组合成"盘带"播出系统, 主播采用新的硬盘播出机, 磁带播出机作为备用播出, 这样试播了 1年多。因为硬盘播出的图像质量高出磁带播放机很多, 所以台里想再购置 1套硬盘播出系统, 按当时的价格一套最少近 10万元, 这对于一个县级台来说, 要拿出这么大一笔资金十分困难。对此, 台里懂计算机的技术人员想, 既然硬盘播出系统是用微机与相关播出板卡通过组装而构成的, 为什么我们不能自己动手组装呢? 于是, 采购来相关部件、板卡等器材, 用不到 10天时间即组装了 1套硬盘播出系统, 通过近两年的使用, 没有出现过大的问题, 播出的图像质量完全可以与高价购买来的硬盘播出机相媲美, 而造价不到硬盘播出机的五分之一。具体做法如下。

1　制定系统方案

因为硬盘播出系统是由电脑和相关播出板卡器件等硬件合成, 再通过播出软件来运行的, 要组装一个系统, 首先要确定方案: 一是要对电脑有比较熟练的了解; 二是要弄清硬盘播出系统的原理和构造; 三是确定采用什么品牌的播出板卡来组装; 四是根据播出板卡的运行条件来确定电脑的档次和配置; 五是选用哪些软件。

电脑和硬盘播出板卡配置如下:

(1)计算机配置

主板: 华硕 P4P800E－Deluxe主板

CPU: P4 3. 0E

内存: 金丝顿 DDR400 512M × 2

硬盘: 系统希捷 80G IDE硬盘

　　素材希捷 160G SATA × 2RAID0阵列

光驱: SONY CDROM

机箱: 4U 工程机箱

电源: 磐石 550服务器电源

显卡: FX5200

显示器: 飞利浦 107S

鼠标 + 键盘: 微软光电套装

(2)播出板卡及软件

播出板卡: TOPACK－2, 也可用更便宜的"安桥 VCARD"播出板卡。

播出软件: 播出软件是从互联网上直接下载的"AUTOPLAY"(自动播出系统)。

2　系统组装

(1)按照组装电脑的要求把电脑组装调试好, 装上一些常用的应用软件, 并使电脑各种功能运行正常。

(2)在板卡线槽装上播出板卡, 再安装上板卡驱动软件。

(3)用"AV"或"S"端子线连接好显示器。

3　播出运行

系统组装完成后, 从互联网上直接下载并安装"AUTOPLAY"播出系统软件 (根据播出板卡的不同, 也可用其他的播出软件来运行), 打开"AUTOPLAY"软件, 把节目素材导入"节目编播表", 按照节目编播表界面上显示的功能进行播出操作。如安装上没有问题, 显示器就会播出导入"节目编播表"的声像节目, 这样 1套硬盘播出系统即组装完成。

[收稿日期: 2007-11-05]

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net