H.264/AVC 中 CAVLC 解码器 IP 核的设计

艾明晶,张 哲,邓 媛
AI Ming-jing,ZHANG Zhe,DENG Yuan

北京航空航天大学 计算机学院,北京 100083
School of Computer Engineering,Beihang University,Beijing 100083,China
E-mail:zhangzhestory@126.com

AI Ming-jing,ZHANG Zhe,DENG Yuan.Design of CAVLC decoder IP core in H.264/AVC standard.Computer Engineering and Applications,2007,43(20):109-112.

Abstract: CAVLC is the entropy module of H.264/AVC,which can affect H.264/AVC decoder's performance directly.Based on existing CAVLC decoder,this paper proposes architecture of CAVLC decoder based on FPGA.This architecture uses the dispersive control strategy to simplify the design,improves some decoder module,and design parallel register array to support the following inverse quantization and inverse transform.After the synthesis in QuatrusII 5.0 of Altera company and timing simulation in ModelSim6.1,this design can fulfill the performance requirement of baseline profile,level 3.0 in H.264 standard.
Key words: H.264;variable decoder;CAVLC;decoder

摘 要:CAVLC(Context-Adaptive Variable Length Coding,基于上下文的变长变码)是 H.264/AVC 的熵解码模块,其性能优劣直接影响 H.264/AVC 解码器的性能。在现有的 CAVLC 解码器基础上,提出了一种基于 FPGA 的 CAVLC 解码器的体系结构,采用分散控制的策略,简化了设计,对 CAVLC 的部分解码模块作了改进,并设计了并行化寄存器组,适于后续快速反量化反变换模块的设计。通过在 Altera 公司的 QuartusII5.0 进行综合并在 ModelSim6.1 下进行时序仿真可知,该设计至少能够满足 H.264 标准 BaseLine 档次、级数(Level)3.0 的要求。
关键词:H.264;变长解码;Context-Adaptive Variable Length Coding(CAVLC);解码器
文章编号:1002-8331(2007)20-0109-04 **文献标识码**:A **中图分类号**:TP39

1 引言

H.264/AVC 是 ITU-T VCEG 组织和 ISO/IEC MPEG 组织共同提出的新型视频压缩标准。H.264/AVC 引进了当前视频编码的许多新技术,在相同视觉感知质量上,H.264 的编码效率相对于其他标准有很大提高,但同时计算复杂度相对于其他标准也成倍增长。CAVLC(Context-Adaptive Variable Length Coding,基于上下文的变长变码)的复杂度在整个解码器设计中占了 28%[7],同时由于 CAVLC 担负着块数据的解码,它在整个解码器工作时占了相当的比重,因此 CAVLC 的设计对整个解码器的性能起着至关重要的作用。

近年来,有很多关于 CAVLC 的解码结构提出来[4-6]。文[4]提出了一种 VLSI 的实现方式,并提出了 CALVC 中首 1 检测器的结构。文[5]提出了一种有效的低复杂度的解码器结构,并给出了比较完整的实现结构。文[6]提出了一种能有效降低存储器访问次数的解码器结构。本论文在上述论文的基础上,取得了以下 4 个成果:

(1)本文并没有采用传统上的全局控制器—各模块数据通道[5]的结构,而是采用了分散控制的方式,每个小模块都有一个控制器,该控制器存储本模块所需的数据,只要本模块所需的数据都到达,那么本模块就开始工作,而本模块的输出数据作为下一级模块的输入数据以及控制信号。采用这样的结构,不仅可以节省控制器的开销,同时可以简化设计。

(2)本文提出了一种快速合并及反 Zig-Zag 扫描的算法,该算法能够在所有的非零系数前连零数目(RunBefore)的解码完成的同时,完成非零系数和 RunBefore 的合并。

(3)本文提出了并行化寄存器组的结构。并行化寄存器组用于保存经 CAVLC 译码并进行反 Zig-Zag 扫描后的块数据,该并行化寄存器组对后续反量化反变换模块提供了一个 4×4 块数据的接口,能够在一个周期就得到一个块的全部数据。

(4)本文针对 CAVLC 的部分解码模块进行了优化,使之工作的速度更高,所需的资源更少。

2 CAVLC 原理

CAVLC 用于亮度块和色度块残差系数的编码,块系数在变换量化之后具有如下特性:变换量化之后,块通常是稀疏的;Zig-Zag 扫描之后,值较大的系数集中在低频端,而值比较小的系数则比较集中在高频端,并且大部分为+1 或-1(CAVLC 把这些高频的+1,-1 称为拖尾系数(TrailingOne));相临块的非零系数的数目是相关的。CAVLC 充分利用了这些特性,进一步减少了数据的冗余信息,为 H.264 卓越的编码效率奠定了基础。

作者简介:张哲,男,硕士研究生,研究方向为嵌入式系统设计,视频编解码技术;邓媛,女,硕士研究生,研究方向为嵌入式系统设计,视频编解码技术;艾明晶,女,副教授,主要从事计算机测控技术、电子系统设计和嵌入式系统设计及应用的研究。

CAVLC 的编码流程包括：

(1)编码非零系数的数目(TotalCoeff)以及拖尾系数的数目(TrailingOnes)：TotalCoeff 和 TrailingOnes 统称为 Coeff_token。TotalCoeff 的范围为 0~16，TrailingOnes 为 0~3。在解码时共有 5 个表可以选择，表选择信号(nC)是根据当前块左边 4×4 块的非零系数数目(nA)和当前块上边 4×4 块的非零系数数目(nB)计算的，如果 nA 和 nB 是可用的，那么 $nC=(nA+nB+1)>>1$；否则如果 nA 是可用的 $nC=nA$；如果 nB 是可用的，$nC=nB$；如果都不可用，$nC=0$[1]。

(2)拖尾系数(TrailingOne)的符号：对于每个拖尾系数，其符号用一个 bit(0=+，1=−)来编码，编码顺序是从高频系数开始按照反向扫描的顺序进行的。

(3)除拖尾系数之外的非零系数(Level)：Level 组成包括两个部分：前缀(Level_prefix)和后缀(Level_suffix)。前缀由码字中首 1 的位置决定，而后缀的更新则与后缀的值以及已经编码的 Level 的幅度有关。Level 编码是从高频系数开始按照反向扫描的顺序进行的。

(4)最后一个非零系数前零的总数目(TotalZeros)：TotalZeros 是指按正向扫描顺序从第一个系数到最后一个非零系数之间零的总数目。

(5)每个非零系数前连零的数目(RunBefore)：RunBefore 是指每个非零系数前连零的数目。

图 1 给出了一个解码过程的实例[8] 、解码的状态转移图以及各个过程的结果。从图 1 中可以看出，一个块的解码并不是所有的状态都经历的。另外，解码的状态转移是由各个小模块的控制器控制的，小模块的控制器根据本模块的解码结果决定将使能数据信号输出给下一级的哪个模块。

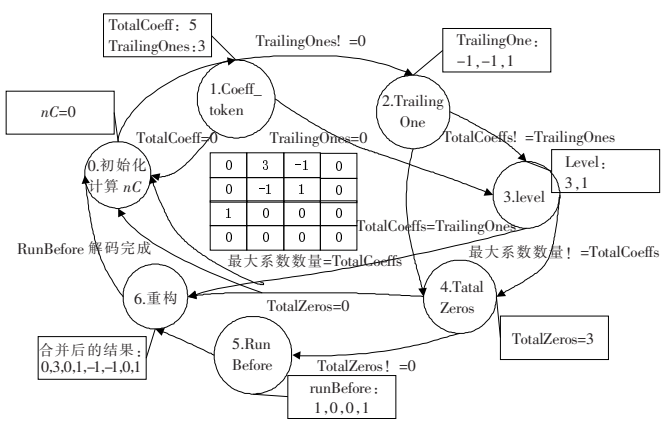


图 1 解码状态转移图

3 论文所提出的结构

根据图 1 的解码器状态转移图，本文提出了图 2 的解码器结构，图 2 中实线表示数据，虚线表示使能数据信号。

整个结构包括输入缓冲、更新缓冲累加器、初始化模块、首 1 检测器、Coeff_token 解码器、拖尾系数解码器、Level 解码器、TotalZeros 解码器、RunBefore 解码器、合并模块以及并行化寄存器组共 11 个模块组成。

解码器的工作过程：CAVLC 的各个解码模块都从输入缓冲(BarrelShifter)中获得解码所需的数据，某个模块的输出送给后面需要该数据的解码模块，后续解码模块将上一级模块的输出保存在自己的内部缓存中，本模块的控制器判断本模块运行所需要的数据是否都已准备好，若准备好就使本模块开始工作，如 Coeff_token 解码模块解码时需要当前块的 nC、该块系数的最大数目(MaxNumCoeff)以及待解码数据，那么在初始化模块计算完当前块的 nC 以及该块的 MaxNumCoeff，并且输入缓冲准备好数据时，本模块就开始工作，在本模块工作完成并将译码结果 TotalCoeff 和 TrailingOnes 送到后续需要这些值的模块缓冲之后，后续模块就按照图 1 的解码状态转移图工作。

3.1 Coeff_token 解码器结构

在对 Coeff_token 解码时，共需要 5 个表，4 个变长表、1 个定长表：VLC0($0\leqslant nC<2$)、VLC1($2\leqslant nC<4$)、VLC2($4\leqslant nC<8$)、VLCCHROMADC($nC=-1$)、FLC($8\leqslant nC$)。在 VLC0~VLC2 表中，可能的码字的最大长度是 16，如果采用单一的 ROM 结构最多可能需要 2^{16}=65 536 个单元，而实际上码表仅有 62 个单元。观察变长表可以发现，只要确定当前码字中首 1 在码字中的位置，接下来就可以确定下一步需要读取的码字的位数。本文在上述思想的基础上提出了基于首 1 检测器的 Coeff_token 解码器，将码字中首 1 位置相同的码字组成一个小表，这样每个小表都能根据首 1 位置确定需要继续读取的 bit 数，之后就可以确定当前码字对应的 TotalCoeff 和 TrailingOnes。由于后续读取的 bit 数不超过 3，所需的 ROM 资源就会非常接近实际所需的 ROM 资源。同样，对于 VLCCHROMADC 表来说，也可以采用这样的结构的。而在 FLC 表中，观察该表可以发现，除了第一个

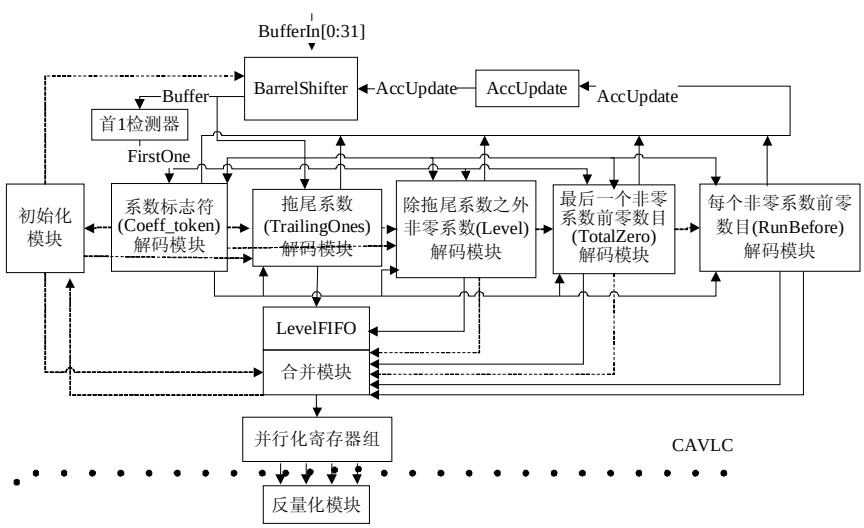


图 2 CAVLC 整体结构

元素之外,其余的元素的前 4 位+1 就是 TotalCoeff,后两位就是 TrailingOnes,因此该表仅仅用一个加法器和一个多路选择器就可以得到结果,省去了查找表的开销。

首 1 检测器可以采用图 3[4]的结构,而 FLC 的解码器则采用图 4 的结构。

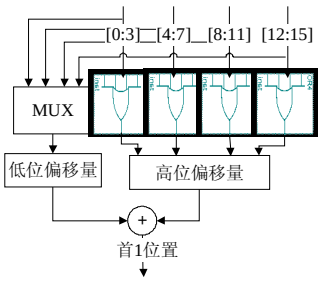


图 3　首 1 检测器

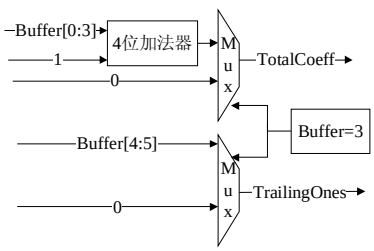


图 4　FLC 解码器

3.2　Level 解码器体系结构

Level 解码器是整个 CAVLC 中比较复杂的模块，在计算 Level 时需要很多的算术运算,一般来说,这些算术运算所需的总时间一般会比较长,如果不能很好地对这些算术运算进行划分,则会在设计中造成较长的时间延迟,成为系统的瓶颈。图 5 给出了 Level 解码器的体系结构,整个 Level 的解码过程分为 3个时钟周期。第一周期采用首 1 检测器来检测比特流中首 1 的位置,然后根据首 1 位置计算出前缀的值(Level_prefix)和首 1 位置之后 6 bit 的数据(Buffer05Pos);在下一个周期,计算出 level 的绝对值(LevelAbs)以及 Level,并给出本模块消耗的 bit 数 (AccUpdate)；在第三个时钟周期更新后缀长度(SuffixLength)。LevelControl 控制器用于控制整个解码过程,判断当前的解码过程是否已经完成，并给出 TotalZero 解码器的使能信号。NumLoopCal 模块则用于更新 Level 解码器需要运行的次数。

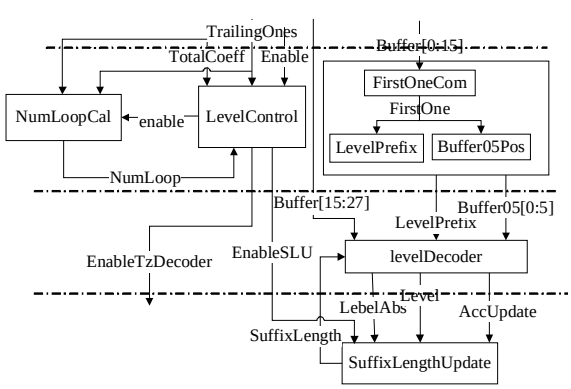


图 5　Level 解码器体系结构

3.3　合并模块的体系结构

合并模块是用于合并来自 Level 解码模块和 RunBefore 解码模块的解码数据(如果有的话)。在合并模块中,本文采用了双缓冲的结构，利用 ping-pong 操作使一个缓冲正在忙的时候,另一个缓冲可以开始工作,这样可以避免后续块的 CAVLC 解码的暂停。

传统的合并方法[5]是先将 Level 和 RunBefore 分别放入不同的堆栈中,之后再不断的弹栈,根据 RunBefore 的值,将 level 放入合并缓存对应的位置。观察合并的过程可以发现,解码过程可以参见表 1(该表的 Level 和 RunBefore 列从上到下为解码顺序)。

表 1　合并过程

Level	RunBefore	Level 位置	逻辑关系
1	1	7	Pos=5+3−1
−1	0	5	Pos=7−1−1
−1	0	4	Pos=pos−0−1
1	1	3	Pos=pos−0−1
3	1	1	Pos=pos−1−1

从表 1 可以看出,在出现第一个 RunBefore 的时候,合并后该 level 的位置为 TotalCoeff+TotalZero−1(合并后块数据的坐标是从 0 开始),之后每出现一个 RunBefore,当前 level 的位置就是上次 level 位置 Pos−上次的 RunBefore−1。从上面的关系可以看出,在 RunBefore 译码完成之后,level 和 RunBefore 的合并也就完成了,而当 TotalZero=0 时,此时 RunBefore 都为 0,其位置关系仍然满足。合并过程不需要额外的存储 RunBefore 的缓冲。

3.4　并行化寄存器组的设计

如果采用传统的设计,一个块数据使用一个 16 位宽 16 深度的 Ram 保存，那么在反 Zig-Zag 扫描阶段需要将 Ram 中块数据逐个取出,并放到反 Zig-Zag 扫描之后的某个位置,这样最多情况下需要 16 个周期，同样在反量化阶段对块数据的读取也要重复同样的过程，这样就会使程序的运行速度大大降低,并行性也受到了很大的束缚。在本文提出的寄存器组的结构中,在保存块数据的同时,就进行反 Zig-Zag 扫描。另外本设计还能在一个周期读出 1 个块的全部数据。

本文是将 1 个块的数据放到 16 个不同的寄存器中，每个寄存器的宽度为 16 位。反 Zig-Zag 扫描是这样进行的,首先对一个块中某个数据在合并后的位置 Pos 译码,根据该数据经反 Zig-Zag 扫描后在块中的位置,将 AddrRamNZig(N=0,1,分别对应两套寄存器组)的对应位置 1,得到一个 16 位的值。如在图 1 中第二个数据 3，其对应的反 Zig-Zag 扫描后的地址应为 1，此时就使 addrRamNZig=16′b0000_0000_0000_0010 (N=0,1)。在需要将非零系数写入寄存器时,写使能信号 write 加到 16 个寄存器的所有写使能端,将 AddrRamNZig 的第 2 位作为第二个寄存器的选择信号。每个数据只需根据 write 信号和 AddrRamNZig 共同作用就可以确定具体需要写入哪个寄存器,这样当合并模块合并完一个块数据时,在对该数据进行保存的同时就完成了反 Zig-Zag 扫描。同样,在读取 1 个块的数据时,只需要将读信号加到 16 寄存器的读使能端就可以一次读取 1 个块的所有的数据。

4　性能分析

本文使用 Verilog HDL 实现了 CAVLC 的解码器，并用 ModelSim6.1 进行了仿真。本设计采用 Altera 公司的 QuartusII-I5.0 作为综合软件,使用该公司的 CycloneII(2C35)FPGA 器件,综合后的结果如表 2 所示,关键路径延迟为 6.219 ns,综合

后设计所能达到的最大速度为 134.03 MHz。

表 2 FPGA 资源使用情况

FPGA 资源	使用量	总量	百分比/%
逻辑单元数量	1 629	33 216	4.9
存储器 bit 数	1 126	483 840	0.2

设 MaxMbPs 为不同级数情况下每秒钟最多能够处理的宏块数;MaxCyclePMb 为处理每个宏块所需的最大时钟周期数;Speed 为设计所能达到的最大速度，这里该值为 134.03 MHz。为使解码器能够正常工作则需要 MaxMbPs×MaxCyclePMb<= Speed。表 3 给出了量化参数为 24 时不同的测试序列完成一个宏块的 CAVLC 解码以及反扫描所需的时钟周期数。表 4 给出了在不同级数情况下的处理一个宏块所需的最大时钟周期数 MaxCyclePMb。

表 3 解码一个宏块所需的时钟周期数

akiyo	foreman	mobile	news
370	928	1125	265

表 4 处理一个宏块所需的最大时钟周期数 MaxCyclePMb

级数	Speed/MHz	MaxMbPs	MaxCyclePMb
1	134.03	1 485	90 255
2	134.03	11 880	11 281
3	134.03	40 500	3 309

从表 3 和表 4 可以看出本论文至少能够满足级数为 3 的实时解码要求。在实际应用中,MaxMbPs 往往比表 4 中所列的要小,同时一个宏块中并不是所有的块都需要进行解码,这样本文所提出的设计在实际应用中性能还会有所提高。

5 结论

本文提出了一种 CAVLC 的解码器架构，采用分散控制的策略使设计更加简单,并且对 Coeff_token 和 Level 解码模块作了改进;另外,还设计了能够对 Level 和 RunBefore 进行快速合并的模块;为了使反量化模块能够进行快速运算,本文提出了并行化寄存器组结构。结果表明,本论文所设计的 CAVLC 解码器结构能够满足实际的解码要求，并且适于将其集成到整个 H.264/AVC 解码系统中。(收稿日期:2006 年 11 月)

参考文献:

[1] Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG. Joint Video Specification(ITU- T Rec,H.264|ISO/IEC 14496-10 AVC)-Joint Committee Draft[S].Document JVT_G050,2003-03:151-159.

[2] Joint Video Team(JVT) reference software JM8.6[S/OL].http://iphome.hhi.de/suehring/tml/.

[3] 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,2005:84-167.

[4] Wu Di,Gao Wen,Hu Ming-zeng,et al.A VLSI architecture design of CAVLC decoder[C]//2003 Proceedings 5th International Conference on ASIC,21-24 Oct 2003,2003,2:962-965.

[5] Chang Hsiu-cheng,Lin Chien-chang,Guo Jiun-in.A novel low-cost high -performance VLSI architecture for MPEG -4 AVC/H.264 CAVLC decoding[C]//ISCAS 2005 IEEE International Symposium on Circuits and Systems,23-26 May 2005,2005,6:6110-6113.

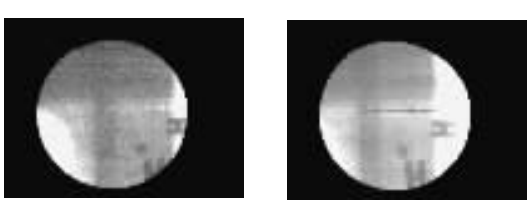
[6] Moon Y H,Kim G Y,Kim J H.An efficient decoding of CAVLC in H.264/AVC video coding standard[J].IEEE Transactions on Consumer Electronics,2005,51(3):933-938.

[7] Xue Quan,Liu Ji-lin,Wang Shi-jie,et al.H.264/AVC baseline profile decoder optimization on independent platform[C]//2005 Proceedings,2005 International Conference on Wireless Communications,Networking and Mobile Computing,23-26 Sept 2005,2005,2:1253-1256.

[8] Iain E G Richardson.H.264 and MPEG-4 Video Compression Video Coding for Next-generation Multimedia.www.vcodex.com,2003:201-207.

(上接 108 页)

计算结果与实际深度比对,绝对误差为 17.65 mm,相对误差为 5.88%。可以看到,误差很大。



(a)第一次曝光的图像 (b)第二次曝光的图像

图 5 定位误差分析中所采用的立体像对

4.2 基于能量中心匹配实现缺陷定位

用能量中心匹配的方法,对图 3 所示的立体像对中的钢珠进行深度定位,利用式(4)计算得到左右图像中钢珠的能量中心坐标分别是:(436.82,415.6)、(370.24,415.6);因此钢珠的投影视差为:

B=66.58 像素

利用式(2)计算得到钢珠的深度值为:

$h\approx$298.83 mm

与钢珠的实际深度比较,计算结果的绝对误差为 1.17 mm,从上面的实验结果可以看出,采用本文提出的能量中心匹配技术后,深度定位精度得到了很大的提高,相对误差达到 0.39%。

5 结论

提高缺陷的深度定位精度,是缺陷三维坐标准确获取的保证,同时也使得对目标实际尺寸的准确测量成为可能。该技术不仅可以用来提高缺陷自动判定系统的检测性能,而且也是实现基于图像的物体内部结构尺寸精确测量的技术基础。

(收稿日期:2007 年 1 月)

参考文献:

[1] Chen S,Metz C.Improved determination of biplane imaging geometry from two projection images and its application to three dimensional reconstruction of coronary arterial treesp[J].Medical Physics,1997,24:633-654.

[2] Jiang H,Liu H,Wang G,et al.A localization algorithm and error analysis for stereo X-ray image guidance[J].Medical Physics,2000,27:885-893.

[3] 付丽琴,韩焱,陈树越.基于立体视觉的 DR 图像定位技术[J].应用基础与工程科学学报,2005(增刊):130-135.

[4] Taylor,John R.An introduction to error analysis:the study of uncertainties if physical measurements.University Science Books,1982.

[5] Taza A,Suen C Y.Discrimination of planar shapes using shape matrices[J].IEEE Trans Syst,Man Cybern,1989,19(5):1281-1289.