文章编号:1673 - 0062(2012)01 - 0030 - 04

H.264 中 4×4 整数 DCT 变换在 DM642 上的优化

黄国玉,邓贤君,王 彦

(南华大学 电气工程学院,湖南 衡阳 421001)

摘 要:基于 TI 公司 DM642 实现了 H.264 整数变换. 首先对 H.264 中整数 DCT 变换编码进行理论分析,然后根据 DM642 的硬件特点对其进行线性汇编改写优化,有效地降低了整个模块运行的时钟周期数. 实验结果表明该方法的优化取得了良好的效果.

关键词:H.264;整数 DCT 变化;线性汇编;DM642

中图分类号:TN911.72 **文献标识码**:B

4×4 Integer DCT Transform Optimization in H.264 Based on DM642

HUANG Guo-yu, DENG Xian-jun, WANG Yan

(School of Electric Engineering, University of South China, Hengyang, Hunan 421001, China)

Abstract: Based on DM642 DSP made by TI Company, integer transform in H.264 is realized. Firstly the theory of the integer transforms in H.264 is analyzed, and then according to the features of DM642 the H.264 integer transform code is rewritten using the linear assembly language, which effectively reduce the clock cycles to run the module. Experimental results indicate that better performance can be achieved for this work.

key words: H.264; integer DCT transform; linear assembly; DM642

0 引 言

随着现代通信技术的发展和数字时代的到来,多媒体信息的压缩编码技术成为当今研究的热点. 到目前为止,国际上视频压缩编码标准共有两大系列:国际标准化组织和国际电工委员会第一联合技术组(ISO/IEC JTC1)制定的 MPEG 系列标准;ITU 针对多媒体通信制定的 H.264x 系列视频编码标准[1]. H.264 采用一系列新的压缩方法,获得了更好的压缩效果. 以往的视频编码标准采用的是 8×8 离散余弦变换,计算量大而且容易带来解码数据失配. H.264 的一个新特点是采用了 4×4 离散余弦变换,可以克服上述问题[2].

由于 H.264 的良好的压缩效率、高质量的图像和较低码率等特点,适合于视频的压缩存储和传输,也便于嵌入式应用. DSP 的价格低廉、性能优越、适合实现计算复杂度高的算法. 因此,本文首先介绍H.264采用的整数变换,着重探讨整数

收稿日期:2011 - 11 - 10

作者简介:黄国玉(1979 -),女,湖南郴州人,南华大学电气工程学院讲师. 主要研究方向:图像视频信号压缩.

变换移植到 DSP 上的代码优化方法.

1　整数 DCT 变换

整数 DCT 变换算法以 4×4 像素子块为单位,在变换和反变换过程中只包含整数运算. 整数 DCT 变换算法采用了全新的变换核和量化公式,该算法可以通过 16 位运算来实现,有效地降低对存储器的要求,同时简化了运算步骤,在计算时只使用加法和移位运算,而无须使用乘法运算[3]. 使用这种算法做变换和反变换同样是可逆的,不存在误匹配问题. 其变换主要是在以下三种情况下作用:

1)对于经过帧内/帧间预测形成的残差矩阵,进行 4×4 整数 DCT 变换;

2)在帧内预测的时候,当对一个宏块进行 4×4 变换之后,将各个 4×4 块的亮度 DC 系数提取出来进行哈达玛变换;

3)和 2 的过程类似,对色度块的 2×2 块的 DC 系数进行哈达玛变换.

4×4DCT 整数变换矩阵 A 为:

$$Y(x,y) = C(x)C(y)\sum_{i=0}^{3}\sum_{j=0}^{3}X(i,j)\times \cos\frac{(2j+1)y\pi}{8}\cos\frac{(2i+1)x\pi}{8} \quad (1)$$

其中 $C(x),C(y) = \frac{1}{2}(x=0),C(x),C(y) = \sqrt{\frac{1}{2}}(x>0,y>0)$. 如果写成矩阵形式,也可以写成:

$$A = \begin{bmatrix} a & a & a & a \\ b & c & -c & b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix}, a = \frac{1}{2},$$

$$b = \sqrt{\frac{1}{2}}\cos(\frac{\pi}{8}) \text{ 及 } c = \sqrt{\frac{1}{2}}\cos(\frac{3\pi}{8}) \quad (2)$$

A 中的 a、b、c 是实数,而要做变换的图像矩阵中的值都是整数. 对于实数 DCT 变换,由于存在浮点计算精度的问题,会造成解码时的数据不匹配. H.264 采用整数 DCT 技术:

$$Y = (CXC^{\mathrm{T}}) \otimes E = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix} X \begin{bmatrix} 1 & 1 & 1 & d \\ 1 & d & -1 & -1 \\ 1 & -d & -1 & 1 \\ 1 & -1 & 1 & -d \end{bmatrix}\right) \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \quad (3)$$

其中,$d = c/b \approx 0.414$,符号⊗表示两个矩阵中的对应位置上的值进行相乘. 为了简化计算,取 $d = 0.5$. 为了保持变换的正交性,取 $b = \sqrt{\frac{2}{5}}$,对矩阵 C 中的第 2 行和第 4 行,以及矩阵 C^{T} 中的第 2 列和第 4 列元素乘以 2,相应地改造矩阵 E 为 E_f,以保持式(2)成立. 得到式(4):

$$Y = (C_f X C_f^{\mathrm{T}}) \otimes E_f = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & -2 & -1 \end{bmatrix} X \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix}\right) \otimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix} \quad (4)$$

将矩阵 E_f 中的计算转移到量化中实现,这样,变换的核心计算就是 $C_f X C_f^{\mathrm{T}}$. 通过式(4)可以看出,在变换中,只要计算加法,减法,和 2 相乘用移位实现.

相应的反变换定义如下:

$$X' = C_i^{\mathrm{T}}(Y \otimes E_i)C_i = \begin{bmatrix} 1 & 1 & 1 & \frac{1}{2} \\ 1 & \frac{1}{2} & -1 & -1 \\ 1 & -\frac{1}{2} & -1 & 1 \\ 1 & -1 & 1 & -\frac{1}{2} \end{bmatrix} \left([Y] \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \right) \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \frac{1}{2} & -\frac{1}{2} & -1 \\ 1 & -1 & -1 & 1 \\ \frac{1}{2} & -1 & 1 & -\frac{1}{2} \end{bmatrix} \quad (5)$$

上式中 Y 为整数变换后的系数,X' 为反变换后的恢复值,$a = \frac{1}{2}, b = \sqrt{\frac{2}{5}}$.

2 基于 DM642 的算法实现和优化

图像压缩编码从本质上来说是数字信号处理,DSP 比通用微处理器更适合

数字信号处理.采用改进的哈佛总线结构,内部有硬件乘法器、累加器,使用流水线结构,具有良好的并行性,有专门设计的适合信号处理的指令系统等.目前市场上的 DSP 芯片以德国德州公司(TI)的 TMS320 系列为主,其中 C64 系列的 DSP 芯片,是 TI 公司专门为视频音频算法的实现设计的.因其良好的并行性,多级缓存和 DMA,EDMA 的传输特性,超长指令字结构使实现视频实时编码容易实现[4].

本文采用的是北京瑞泰公司的 ICETEK-DM642-PCI 评估版,集成了音视频及网络接口,是一个低功耗独立开发平台,便于用户对 TI 的 C64 系列的 DSP 进行测评和开发应用.本文所采用的软件开发平台是 TI 公司的集成开发环境 CCS(Code Composer Studio),该软件平台提供了基于 C 语言的系统调试工具、汇编优化器和链接器.

本文所运行的程序是参考 X264 的程序移植到 DM642 的.因受到 DSP 的内存,内部结构等特性的影响,需要对 X264 的代码结构进行一些修改,以适应嵌入式应用[5].主要步骤如下:

1)编码流程结构的优化;

2)存储空间的优化[6];

3)利用 EDMA 进行乒乓缓存

4)程序代码级优化[4].

本文着重探讨使用线性汇编对 DCT 变换和 IDCT 变换进行优化.线性汇编和汇编源代码相似,但是,线性汇编代码中没有指令延迟和寄存器使用信息.通常可以针对一些计算量不算很大,但是需要经常调用的函数,写成线性汇编的形式.结合以上特性,将 DCT 变换、IDCT 变换函数,写成了线性汇编形式[7].对于这两种代码的共同特点就是都具有矩阵相乘的运算,充分利用汇编语句中的计算乘法加法的超长指令字,来进行合并运算,并且每次读取 64 bit 的数据,传送 64 位的数据,使得该函数的三个最耗时间的部分得到处理.

部分源代码如下:

```
static void dct2x2dc(int16_t d[2][2])
{
int tmp[2][2];
tmp[0][0] = d[0][0] + d[0][1];
tmp[1][0] = d[0][0] - d[0][1];
tmp[0][1] = d[1][0] + d[1][1];
tmp[1][1] = d[1][0] - d[1][1];
    d[0][0] = tmp[0][0] + tmp[0][1];
    d[1][0] = tmp[1][0] + tmp[1][1];
    d[0][1] = tmp[0][0] - tmp[0][1];
    d[1][1] = tmp[1][0] - tmp[1][1];
}
```

这部分计算相当于 DCT 中残差矩阵和系数矩阵相乘的实现.在线性汇编中,利用打包指令和加减指令,实现这部分功能:

```
.global _dct2x2dc
_dct2x2dc:  .cproc d
            .reg tmp1:tmp0
            LDNDW.D1T1  *d,tmp1:tmp0;
            PACK2.S1 tmp1,tmp0,tmp0   ;
          ||PACKH2.L1 tmp1,tmp0,tmp1;
            ADD2.L1 tmp0,tmp1,tmp0   ;
          ||SUB2.S1 tmp0,tmp1,tmp1   ;
            PACK2.S1 tmp1,tmp0,tmp0   ;
          ||PACKH2.L1 tmp1,tmp0,tmp1   ;
            ADD2.L1 tmp0,tmp1,tmp0   ;
          ||SUB2.S1 tmp0,tmp1,tmp1   ;
            PACK2.S1 tmp1,tmp0,tmp0   ;
          ||PACKH2.L1 tmp1,tmp0,tmp1   ;
            STNDW.D1 tmp1:tmp0,*d
            .endproc
```

3 实验结果

测试中,对标准序列“test7.yuv”(176×144)分别编写了两个版本的代码进行测试.第一个是对 X264 中整数 DCT 变换的直接移植,第二个是进行了对整数 DCT 变换进行线性汇编改写的代码.实验中,总共编码了 90 帧,I 帧为 45 帧,P 帧为 45 帧.实验结果如表 1、表 2、表 3 所示.

表 1 DCT2×2 程序运行结果

Table 1 Result of theprogram about DCT2×2

优化前所需时钟周期数	优化后所需时钟周期数	速度提高百分比
5 261 212	675 752	87.2%

表 2　DCT4×4 程序运行结果

Table 2　Result of theprogram about DCT4×4

优化前所需时钟周期数	优化后所需时钟周期数	速度提高百分比
269 790	22 440	91.7%

表 3　IDCT4×4 程序运行结果

Table 3　Result of theprogram about IDCT4×4

优化前所需时钟周期数	优化后所需时钟周期数	速度提高百分比
263 670	17 595	93.4%

由表 1、表 2、表 3 可以看出,经过线性汇编改写的代码效率远远高于按 X264 直接移植的代码效率.

参考文献:

[1] Iain E G. Richardson. H.264 and MPEG-4 video compression[M]. Wiley Publishing, Inc. 2004.

[2] 雷国平,周混,吉吟东. MPEG 标准发展和研究综述[J]. 计算机工程,2003,29(12):1-2.

[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.

[4] 陈宝远,吴孟泽,张清喜,等. H.264 标准中整数 DCT 图像压缩算法的优化及实现[J]. 哈尔滨理工大学学报,2009,14(5):27-30.

[5] Yun Zhang, Gang yi Jiang, Mei Yu. Low-complexity quantization for H.264/AVC[J]. Joural of Real-Time Image Processing, 2009, 4(1):3-12.

[6] 曹玲芝,张恒. 二维 DCT 编码的 DSP 实现与优化[J]. 微计算机信息,2008,24(17):178-179.

[7] 彭启琮,管庆. DSP 集成开发环境——CCS 及 DSP/BIOS 的原理与应用[M]. 北京:电子工业出版社,2004.