CN43-1258/TP
ISSN 1007-130X

计算机工程与科学
COMPUTER ENGINEERING & SCIENCE

2006 年第 28 卷第 11 期
Vol. 28, No. 11, 2006

文章编号:1007-130X(2006)11-0051-03

# H.264 帧内预测模块在 TMS320C6416 上的并行实现*

# Parallel Implementation of the H.264 Intra-Frame Prediction Module on TMS 320C6416

杜谋辉[1],周媛媛[2],余圣发[2],林嘉宇[2]
DU Mou-hui[1], ZHOU Yuan-yuan[2], YU Sheng-fa[2], LIN Jia-yu[2]
(1. 63726 部队,内蒙古 呼和浩特 010000;2. 国防科技大学电子科学与工程学院,湖南 长沙 410073)
(1. Corps 63726, Huhhot 010000; 2. School of Electronics Science and Engineering, National University of Defense Technology, Changsha 410073, China)

**摘 要**:本文结合 DSP 芯片 TMS320C6416 的结构特征,在分析 H.264 帧内预测模块并行特性的基础上提出了 H.264 帧内预测模块的并行实现方法。工程实践结果表明,并行处理效果比较理想。

**Abstract**: Based on the structural features of the DSP chip TMS320C6416, as well as analysing the parallelism of the H.264 intra-frame prediction module, we put forth a parallel implementation method for the H.264 intra-frame prediction module. The results of engineering practice show the ideal effect of parallel processing.

**关键词**:H.264;TMS320C6416;帧内预测;并行;VLIW;SIMD

**Key words**: H.264; TMS320C6416; intra-frame prediction; parallel; VLIW; SIMD

**中图分类号**:TP391.41 **文献标识码**:A

## 1 引言

H.264[1]是由 ITU-T 成立的视频编码专家组 VCEG(Video Coding Experts Group,简称 VCEG)和 ISO/IEC 成立的运动图像编码专家组 MPEG 组成的联合视频小组 JVT(Joint Video Team,简称 JVT)制定的新一代视频压缩国际标准,于 2003 年 3 月正式发布。H.264 采用 DPCM 加变换编码的混合编码模式,具有很高的编码效率,在相同的重构图像质量下,能够比 H.263 节约 50%左右的码率,但相应地增加了算法的复杂度,从而对算法的开发应用不利。一个较好的方法就是并行处理。TMS320C6416[2]芯片是 TI 公司推出的新一代并行处理的定点数字信号处理器,非常适用于 MPEG2、MPEG4 和 H.264 等视频编解码器,其市场应用前景十分广泛。如何在通用 DSP 芯片上实现实时的视频压缩和解压就成为一个热门的课题。本文对 H.264 帧内预测模块在 TMS320C6416 上的并行实现进行了研究和实现,取得了非常理想的并行处理效果。

## 2 TMS320C6416 简介

TMS320C6416 是 TI 公司推出的新一代数字信号处理器,其 DSP 主频 1.1GHz,最高处理能力高达 8 800 MIPS,采用超长指令字 VLIW 结构:拥有两个各自独立的数据通道,64 个 32-bit 字长的通用寄存器,八个独立功能单元:六个算术运算单元 ALU(32-/40-bit)支持每时钟周期单 32-bit、双 16-bit 或者四个 8-bit 算术运算;两个乘法器支持每时钟周期八个 16 * 16-bit 乘法运算(32-bit 结果)或者八个 8 * 8-bit 乘法运算(16-bit 结果)。采用 L1/L2 两级高速缓存结构:128K-bit 程序高速缓存 L1P、128K-bit 数据高速缓存 L1D、8M-bit 内存 L2(RAM/Cache 可设置)。

TMS320C6416 的 CPU 具有 32 位的数据和程序通道,使其可以每个机器周期并行执行八条指令,从而为并行处理在硬件上提供了可能。TI 公司同时提供了 CCS(Code Composer Studio,简称 CCS)平台和针对 TMS320C6416 的汇编语言。该汇编语言中包含了大量诸如 ABS2、ADD4、

* 收稿日期:2005-05-11;修订日期:2005-09-09
作者简介:杜谋辉(1976-),男,湖南慈利人,硕士生,工程师,研究方向为图像处理和数据压缩;林嘉宇,博士生,副教授,研究方向为信息与编码理论、语音、图像处理和数据压缩。
通讯地址:010000 内蒙古呼和浩特市金川开发区蒙吉利路 63726 部队;Tel:13404843128;E-mail:jupterdmh@163.com
Address: Corps 63726, Mengjili Rd, Jinchuan Develoment Zone, Huhhot, Inner Mongolia 010000, P. R. China

AVG2、CMPEQ2、CMPGT4、DOTP2 等[3]针对 32 位寄存器的高低半字或四个字节处理的汇编指令，从而为并行处理在软件支持上提供了可能。

H.264 算法的基本系统是用 C 编写的，而且 CCS 平台包括了优化 C 编译器，可以在 C 源程序进行开发调试，大大提高了算法的开发速度和可读性，方便了软件的调试、修改；同时可以在该 CCS 平台上用汇编语言进行开发，更为合理地利用芯片硬件资源，提高代码执行效率。为充分利用 DSP 芯片资源，更好发挥 C 语言和汇编语言的各自优点，采用 C 和汇编语言混合编程开发：系统结构用 C 开发，具体模块用汇编语言开发。

## 3 H.264 帧内预测模块分析

H.264 算法是将视频图像在 YUV 空间中进行压缩处理，利用亮度信号(Y)和色差信号(U、V)相互独立来对单色图分别进行编码，利用人类视觉对亮度信号较之色度信号敏感的特性来降低数字彩色图像所需的存储空间。

H.264 的编码方法分为预测、运动补偿、4×4 块的整数变换、量化和熵编码，其中熵编码包括基于上下文的自适应二进制算术编码 CABAC(Context-Adaptive Binary Arithmetic Coding，简称 CABAC)和通用变长编码 UVLC(Universal Variable Length Coding，简称 UVLC)。

预测分为帧内预测模式和帧间预测模式，输入图像帧的每个宏块通过帧内帧间选择器来判定采用的预测模式。如果一个块以帧内预测模式编码时预测块是由以前重构的块来获得，在编码时从当前块中减去预测块，对得到的残差进行编码。对于亮度采样，预测块可以是 4×4 块或者 16×16 宏块：4×4 块的亮度预测有 9 种模式，16×16 宏块的亮度预测有四种模式；对于色度采样，只对 8×8 宏块进行预测，色度预测有四种模式。

4×4 块的亮度预测模式：为了保持解码的独立性，只有同一个片(Slice)内的像素才可以用来预测。对于每个 4×4 块(即图 1 中小写字母所标记的 16 个像素点)，每个像素都可用 15 个最接近的先前已编码像素的不同加权和来预测，即此像素所在块的左侧和上边的 15 个像素(即图 1 中大写字母所标记的像素点)。显然，这种帧内预测是在空间域上进行的预测编码算法，可以除去相邻块之间的空间冗余度，压缩更为有效。编码器为每块选择预测模式的准则是：使预测块与原始块之间的残差 SAE(Sum of Absolute Errors，简称 SAE)最小。它共有九种预测方式(如图 2 所示，模式 2 未标示)：模式 0(垂直预测)、模式 1(水平预测)、模式 2(直流预测)、模式 3(左下对角线预测)、模式 4(右下对角线预测)、模式 5(垂直偏右预测)、模式 6(水平偏下预测)、模式 7(垂直偏左预测)和模式 8(水平偏上预测)。

| X | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| I | a | b | c | d | | | | |
| J | e | f | g | h | | | | |
| K | i | j | k | l | | | | |
| L | m | n | o | p | | | | |

图 1 4×4 块相邻像素

16×16 宏块的亮度预测模式，有模式 0(垂直预测)、模式 1(水平预测)、模式 2(直流预测)和模式 3(平面预测)，如图 3 所示。其中，中间的正方形表示原始像素点所在的块，左边及上方的长方形表示用来预测的像素点，箭头表示预测模式。模式 3 是一个线性平面函数，对于亮度变化平缓的区域效果比较好。

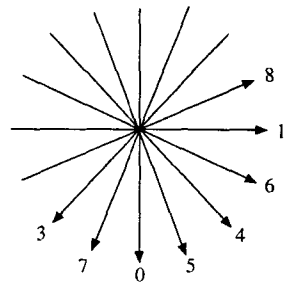


图 2 4×4 块预测模式方向

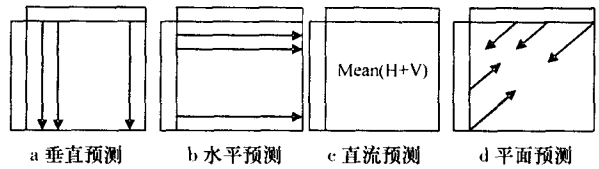


图 3 16×16 宏块亮度预测模式

8×8 宏块的色度预测模式与 16×16 宏块亮度预测模式类似，色度数据 U、V 联合处理，色度预测模式分为模式 0(直流预测)、模式 1(水平预测)、模式 2(垂直预测)和模式 3(平面预测)。

## 4 H.264 帧内预测模块并行实现研究

### 4.1 并行算法的定义与分类[4]

一般来说，并行算法是指在各种并行计算机上求解问题和处理数据的算法，其本质是把多任务映射到多处理机中执行，或是把多维问题映射到具有特定拓扑结构的多处理机上求解。并行算法的目标是尽可能减少时间复杂性，通常是通过增加空间复杂性来实现。

并行算法可以从不同的角度分类：(1)根据实现算法所依赖的并行机类型的不同可以分为单指令流多数据流 SIMD 算法、多指令流多数据流 MIMD 算法、分布式算法和超大规模集成 VLSI 算法；(2)根据实现算法所依赖的器件级别的不同可以分为机器级并行、芯片级并行(如多内核)、线程级并行(如超线程 Hyper-Threading)和指令级并行(如超长指令字 VLIW)；(3)根据并行算法处理问题类型的不同可以分为数值算法和非数值算法；(4)根据算法规则中并行运算之间的相互关系可以分为同步算法和异步算法。

### 4.2 H.264 帧内预测模块中的并行算法

在 H.264 帧内预测模块中，原始数据与重构数据为 char 类型(8 位)，预测数据与待编码残差数据为 short 类型(16 位)。由于 TMS320C6416 寄存器为 32 位，故可以用 SIMD 并行算法中的寄存器高低位并行，每个寄存器同时处理两个 short 数据或四个 char 数据。

由于 TMS320C6416 的 CPU 芯片采用 VLIW 技术，一个机器周期可以执行多条指令，故可以使用超长指令字

VLIW并行。由于TMS320C6416拥有A、B两个寄存器组(各32个寄存器),故可采用寄存器组并行,每个寄存器组同时处理两个数据;由于TMS320C6416拥有四个运算器功能单元(L、S、D、M),故还可以采用运算器功能单元并行,每个机器周期对L、S、D、M运算器功能单元进行操作。

## 5 具体实现及指标

在H.264帧内预测模块并行算法的具体实现中,鉴于C源程序复杂度高,对C源程序进行简化处理,删去Hadamard变换等程序。根据TMS320C6416的特性,运用流水线技术对帧内预测模块亮度数据编制流水线流程:4×4预测→4×4DCT变换→16×16预测→SAE判别→16×16DCT变换→熵编码。4×4预测编制成从模式0依次到模式8的流水线流程,16×16预测编制成从模式0依次到模式3的流水线流程。

在4×4预测模式0到模式8流程中,运用流水线技术和超级步调整将并行粒度细化,将原来的一个步骤经过提取相同数值计算细化为三个步骤,从而提高并行效率。例如,在4×4预测模式3(左下对角线模式)中,a由(A+2B+C+2)>>2预测得到;b、e由(B+2C+D+2)>>2预测得到;c、f、i由(C+2D+E+2)>>2预测得到;d、g、j和m由(D+2E+F+2)>>2预测得到;h、k和n由(E+2F+G+2)>>2预测得到;l和o由(F+2G+H+2)>>2预测得到;p由(G+3H+2)>>2预测得到。经过分析发现:第一步,可以提取公因子p:A+B+1,B+C+1,C+D+1,D+E+1,E+F+1,F+G+1,G+H+1,H+H+1;第二步,进一步提取公因子q:B+1,D+1,F+1,H+1。在编程中实现并行处理:第一个机器周期(Cycle),计算公因子q;第二、三个指令周期,用公因子q计算公因子p;第三、四个机器周期,用公因子p计算预测值a和b;往后依次交错计算c、d……。在计算中采用寄存器组并行和运算器功能单元(L、S、D、M)并行,以发挥一个机器周期并行执行八条指令的潜力。理论上只需五个机器周期,但由于移位指令(shl、shr)必须在S功能单元进行操作,而且A→B、B→A的交叉通道各只有一条,数据依赖性较强,故实际用七个机器周期完成模式3的计算。相对于不采用并行处理要花费20个机器指令周期,获得了理想的并行加速比。

由于4×4预测九种预测模式均对同一块原始数据进行处理,故取原始数据放在程序的开始,只需执行一次,往后保留寄存器即可。为提高并行处理效率,采用寄存器高低位并行处理两个数据,计算残差diff用sub2指令,计算残差和SAE用abs2、add2或者subabs4、add4指令,存储预测值、残差值用stw指令,以便同时对寄存器高低位并行处理。在对寄存器高低位拆分求和时,用shr结合swap2指令并行处理,相对于C线性汇编中shl、shr指令要节约50%的机器指令周期。

在4×4预测九种预测模式的流水线处理中,每计算出该种模式下的残差和SAE,立即与最小残差和min_SAE(初始值为1<<20)进行比较:如果小于最小残差和,则最小残差和min_SAE用SAE代替,同时存储预测值、残差值;否则直接跳转到下一预测模式进行计算。由于4×4预测的最佳模式一般集中于前三种模式,故可减少存储预测值和残差值的次数,同时存储预测值和残差值的内存空间减少为原来的1/9。在执行跳转指令时,由于跳转指令B有五个指令周期的延迟槽(Slot),可充分利用:即先执行跳转指令,随后的五个机器周期内继续进行计算或处理。

在16×16预测中,采用寄存器高低位并行处理两个4×4块的数据,采用两对寄存器组并行处理4×4块内的四个数据;同时采用运算器功能单元(L、S、D、M)并行,以发挥一个机器周期并行执行八条指令的潜力。这样可以同时并行处理八个数据,4×4块内只需循环四次,4×4块外只需循环八次便可处理完一个16×16宏块。

在16×16预测四种预测模式的流水线处理中,每计算出该种模式下残差和SAE后的处理与4×4预测相似,只是最小残差和min_SAE为16×16宏块中16个4×4块的最佳预测模式下的min_SAE之和。存储预测值内存空间减少为原来的1/4,跳转指令B的延迟槽(Slot)的利用也一样。

经过以上并行处理后,H.264帧内预测模块对一帧QCIF(176×144)图像处理仅耗费约155万个机器周期,获得了较好的并行效果。4×4预测九种预测模式各自的并行处理加速比见表1,平均加速比约4.7,内存节省78%。16×16预测四种预测模式各自的并行处理加速比见表2,平均加速比约6.9,内存节省35%。

**表1 4×4预测加速比**

| 4×4预测模式 | 串行(周期) | 并行(周期) | 加速比 |
|---|---|---|---|
| 模式0 | 174 | 39 | 4.46 |
| 模式1 | 174 | 41 | 4.24 |
| 模式2 | 185 | 57 | 3.24 |
| 模式3 | 211 | 41 | 5.14 |
| 模式4 | 211 | 41 | 5.14 |
| 模式5 | 211 | 44 | 4.74 |
| 模式6 | 211 | 39 | 5.41 |
| 模式7 | 211 | 43 | 4.90 |
| 模式8 | 211 | 36 | 5.86 |

**表2 16×16预测加速比**

| 16×16预测模式 | 串行(周期) | 并行(周期) | 加速比 |
|---|---|---|---|
| 模式0 | 2 354 | 417 | 5.64 |
| 模式1 | 2 354 | 371 | 6.34 |
| 模式2 | 2 524 | 358 | 7.05 |
| 模式3 | 4 316 | 518 | 8.33 |

经过计算,16×16宏观预测平均耗费1 568个机器周期,平均每个像素耗费约1.5个机器周期;4×4块预测平均耗费488个机器周期,平均每个像素耗费约3.4个机器周期。与4×4块预测比较,16×16宏块预测平均每个像素耗费机器周期量减少56%,并行处理效果更好。

## 6 结束语

H.264帧内预测模块并行算法具有这样的规律:并行处理主要受交叉通道、寄存器数量、移位指令及数据依赖性的限制,其中移位指令及数据依赖性的限制无法避免。块愈大、数据量愈多,则可以采用寄存器高低位并行处理多块

(下转第59页)

表 4 *k*-summary 算法与 *k*-prototypes 算法在 KDDCUP99 上的聚类结果统计

| 算法 | 平均执行时间(s) | 平均聚类精度(%) | 最大聚类精度(%) | 最小聚类精度(%) |
|---|---|---|---|---|
| *k*-summary 算法 | 2 586 | 98.67 | 98.93 | 98.31 |
| *k*-prototypes 算法 | 2 357 | 95.38 | 97.94 | 92.70 |

表 5 *k*-summary 算法在不同规模数据集上的时间比较

| 聚类子集 | $T$ | $T_1$ | $T_2$ |
|---|---|---|---|
| 平均执行时间(s) | 2 586 | 253 | 107 |

### 4.4 Congressional Votes 数据集

Congressional Votes 数据集记录了美国国会 1984 年 435 个国会议员的投票情况。每条记录为一个议员在 16 个属性上的表决情况，并有一个分类标志 Republican 或 Democrat，用以表明对应的议员所属党派。每条记录由 16 个属性描述，每个属性仅取三个值(y 表示同意，n 表示不同意，? 表示不表态)。*k*-summary 算法与 *k*-prototypes 算法在该数据子集上重复运行 10 次的统计结果如表 6 所示。

表 6 *k*-summary 算法与 *k*-prototypes 算法在 Votes 上的聚类结果统计

| 算法 | 平均执行时间(s) | 平均聚类精度(%) | 最大聚类精度(%) | 最小聚类精度(%) |
|---|---|---|---|---|
| *k*-summary 算法 | 4.8 | 88.78 | 95.17 | 61.61 |
| *k*-prototypes 算法 | 4.6 | 78.30 | 86.67 | 61.38 |

*k*-summary 算法的执行时间比 *k*-prototypes 算法平均高出约 4%，平均聚类精度高出 10%。

### 4.5 DARPA98 数据集

DARPA98 数据集[6]是用于评估入侵检测系统的数据集。这个数据集有六周的训练数据，包含 15 636 条会话记录，每条会话记录由五个分类属性：Service identifier Source IP address、Destination IP address、Source port、Destination port 和一个数值属性 duration 来描述。*k*-summary 算法与 *k*-prototypes 算法在该数据子集上重复运行 10 次的统计结果如表 7 所示。

表 7 *k*-summary 算法与 *k*-prototypes 算法在 DARPA98 上的聚类结果统计

| 算法 | 平均执行时间(s) | 平均聚类精度(%) | 最大聚类精度(%) | 最小聚类精度(%) |
|---|---|---|---|---|
| *k*-summary 算法 | 52.6 | 99.46 | 99.60 | 97.01 |
| *k*-prototypes 算法 | 45.3 | 96.93 | 97.67 | 95.83 |

*k*-summary 算法的执行时间比 *k*-prototypes 算法平均高出约 16%，平均聚类精度高出约 2.5%。

与 *k*-means 算法类似[7~9]，*k*-summary 算法的聚类结果对聚类个数及初始对象的选择较敏感。由于本文重在说明数据划分的策略和距离定义，对聚类个数与初始对象的选择方法不予讨论。

## 5 结束语

本文在分析 *k*-modes 算法和 *k*-prototypes 算法不足的基础上提出了一种新的聚类表示模型和距离定义，对 *k*-modes 算法和 *k*-prototypes 算法进行了改进，理论分析和实验结果表明，改进后的算法 *k*-summary 较 *k*-prototypes 算法提高了聚类精度，但时间代价略有增加。

### 参考文献：


[1] J MacQueen. Some Methods for Classification and Analysis of Multivariate Observations[A]. Proc 5th Berkeley Symp Mathematics Statist and Probaility[C]. 1967. 281-297.

[2] H Ralambondrainy. A Conceptual Version of the k-Means Algorithm[J]. Pattern Recognition Letters, 1995, 16(11): 1147-1157.

[3] Zhexue Huang. A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining[A]. Proc SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery[C]. 1997.

[4] Zhexue Huang. Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values[J]. Data Mining and Knowledge Discovery, 1998, 2(3): 283-304.

[5] C J Merz, P Merphy. UCI Repository of Machine Learning Databases[EB/OL]. http://www.ics.uci.edu/ mlearn/ MLRRepository.html, 2004-09.

[6] MIT Lincoln Labs. 1999 DARPA Intrusion Detection Evaluation[EB/OL]. http://www.ll.mit.edu/IST/ideval/index.html, 1999-12.

[7] G W Milligan, M C Cooper. An Examination of Procedures for Determining the Number of Clusters in a Data Set[J]. Psychometrika, 1987, 50(2): 159-179.

[8] M Meila, D Heckerman. An Experimental Comparison of Several Clustering and Initialization Methods[A]. Proc of the 14th Conf on Uncertainty in Artificial Intelligence[C]. 1998. 386-395.

[9] C Fraley, A E Raftery. How Many Clusters? Which Clustering Method? Answers via Model-Based Cluster Analysis[J]. Computer Journal, 1998, 41 (8): 578-588.


(上接第 53 页)

数据，同时采用多组寄存器并行处理多个块内的数据，以避开交叉通道的限制(一个机器周期内由 A 计算到 B 和由 B 计算到 A 各自只能执行一次)。但是，数据增多后，受寄存器数量限制影响增大。根据寄存器数量合理选择处理数据量的大小，可使并行效率达到最高。

### 参考文献：


[1] ITU-T Rec. H.264|ISO/IEC 14496-10 AVC, Text of Final Committee Draft of Joint Video Specification[S]. 2002.

[2] TMS320C64x Technical Overview[Z]. Texas Instruments Incorporated, 2001.

[3] TMS320C6000 CPU and Instruction Set Reference Guide [Z]. Texas Instruments Incorporated, 2000.

[4] George R Desrochers. Principles of Parallel and Multiprocessing[M]. New York: McGraw-Hill Book Company, 1987.