基于TMS320DM6446的H.264视频编码算法的移植与优化

程鸿芳[1,2],赵森严[1]
(1.安徽工程科技学院,2.芜湖职业技术学院,安徽 芜湖 241000)

摘 要:本文详细阐述了TMS320DM6446的硬件平台及软件架构,该平台以H.264的参考代码x264为实验平台,针对其实际的应用,采用相应的方法,在保证性能的前提下,对原始代码作了与平台无关的裁剪,着重介绍了编码的移植过程以及编码的系统级和汇编级优化,给出了相应的优化结果。

关键词:视频编码;优化;移植
中图分类号:TN919.81 **文献标识码**:A **文章编号**:1007-4260(2010)01-0030-05

0 引 言

20世纪90年代以来,世界向着信息化社会发展的速度明显加快,而多媒体技术的应用在这一发展过程中发挥了极其重要的作用。作为多媒体技术中最重要,最复杂的数字视频处理技术,也取得了重大的进步。目前已制定了MPEGx和H.26X等系列视频压缩的国际标准。H.264作为新一代视频编解码标准,采用了成熟的技术,具有压缩比高、纠错能力强、适应范围广等优点,在追求更高的编码效率和简洁的表达形式的同时也提高了视频图像的质量,是目前最高效的视频压缩方法。本文以TI的TMS320DM6446为例,在分析该处理器硬件资源特点以及视频压缩算法特点的基础上,提出了一种基于DaVinci平台H.264视频编码算法的移植,并作了相应的优化。优化以后的编码器可以达到4路CIF格式图像实时编码的效果,即一路D1格式图像实时编码的要求,达到了目前市场的主流要求。

1 DaVinci TMSDM3206446的硬件平台以及软件架构

1.1 DaVinci TMSDM3206446的硬件平台

TI的DaVinci系统是一套基于DSP组件的解决方案,它适用于视频安全、数码相机、高级医学影像、便携视频播放器或其他的视频应用的高效且高质量的的数字视频。DaVinci系统包括以下几个部分:DaVinci软件、流行的操作系统(比如Linux)的APIs;DaVinci开发工具包、完整的开发系统、参考设计和全面的ARM/DSP系统级集成开发环境;DaVinci处理器,包括可伸缩、可编程的DSP和基于DSP芯片(片上系统)处理器;DaVinci技术支持系统。

TI DaVinci的双核架构其特点如下:

(1)高性能。采用低功耗、高性能的32位TMS320C64x内核和ARM926EJ-S内核,工作频率分别高达594MHz和297MHz;支持多媒体处理技术,采用的是TMS320C64x DSP内核,增强了对视频和音频的解码能力。

(2)低功耗。多电源管理模式,双内核电压供给为1.6V;ARM926EJ-S内核具有16KB指令和8KB数据Cache;TMS320C64x+ DSP内核具有32KB程序RAM/Cache、80KB数据RAM/Cache及64KB未定义RAM/Cache;支持3.3V或1.8V的I/O接口和存储器接口。

(3)专用的视频图像处理器和视频处理子系统。专用的视频图像处理器用于对视频数据处理,视频处理子系统包括1个视频前端输入接口和1个视频末端输出接口,视频前端输入接口用于接收外部传感器或视频译码器等图像,视频末端输出接口输出图像到SDTV、LCD、HDTV等显示屏上。

(4)存储容量。有256MB的32位DDR2 SDRAM存储空间,128MB的16位FLASH存储空间。

* **收稿日期**:2009-04-14
作者简介:程鸿芳,女,安徽黄山人,芜湖职业技术学院教师。

(5)众多的外设。64 通道增强型 DMA 控制器；串行端口(3 个 UARTs、SPI、音频串口)；3 个 64 位通用定时器；10/100M 以太网；USB2.0 端口；3 个 PWM 端口；多达 71 个通用 I/O 口；支持 MMC/SD/CF 卡等。

(6)时钟控制。时钟源：27MHz 系统振荡器；24MHzUSB 振荡器。

达芬奇数字媒体片上系统(DMSoC)提供：两个内核(ARM+DSP)；视频处理子系统(VPSS)；多种 Boot 模式(NOR Flash/NAND Flash/UART0 Boot Mode)；两个电源域；多个时钟树；多个引脚独立或复用的外设。

1.2　TI DaVinci 的软件架构

CCS3.2(Code Composer Studio)是目前普遍使用的 TI 的 DSP 软件开发工具，它内部集成了以下软件工具：(1)C6000 代码产生工具(包括 C 的编译器、汇编优化器、汇编器和连接器)；(2)软件模拟器(Simulator)；(3)实时基础软件 DSP/BIOS；(4)主机与目标机之间的实时数据交换软件 RTDX；(5)实时分析和数据可视化软件。

在 CCS3.2 下，开发时可以对软件进行编辑、编译、调试、代码的性能分析和项目管理等所有工作，大大降低了 DSP 的系统开发难度。

CCS 内部包含的实时操作系统(RTOS)DSP/BIOS，其主要是为需要多任务实时调度和同步以及主机/目标系统通信和实时监测的应用而设计的。作为一种 RTOS，DSP/BIOS 是 DSP 应用软件的基础和开发平台。随着应用系统复杂度的提高，系统要管理的资源越来越多，如存储器、外设、网络协议等，DSP/BIOS 可以帮助开发者管理这些资源，使编程更加标准化，降低软件开发的难度，缩短了建立 DSP 应用程序的时间。DSP/BIOS 组件包括抢先式多任务内核、硬件抽象层、实时分析工具和配置工具等。

2　H.264 在 TMS320DM6446 平台上的移植

原始的 H.264 只能在 PC 机上运行，因此我们的首要工作是移植 X264，让源程序能在达芬奇平台上良好地运行起来。由于 H.264 算法的复杂性，算法在达芬奇平台上不能实现实时编码，但是达芬奇平台良好的数据处理能力，使我们能够在移植成功之后，对代码做各个级别的充分优化，以满足我们实时编码性能的要求。

2.1　H.264 编码器在 DaVinci 平台上的移植

2.1.1　平台无关的代码的实现

H.264 标准仅规定了一个编码后的视频比特流的句法和该比特流的解码方法，没有明确规定编码器和解码器的具体实现，因此，H.264 编码器在实现上具有很大的灵活性。本文参考 H.264 标准参考软件 X264，深入研究了 H.264 的编码原理及流程，并根据确定的编码技术方案，实现了平台无关的 H.264 编码器。

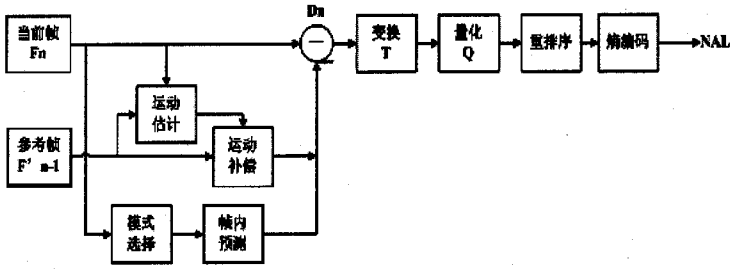

图 1　编码器前向通路

H.264 编码器包括两个数据流通路，一个前向通路(从左到右)，如图 1 所示，一个重构通路(从右到左)，如图 2 所示。

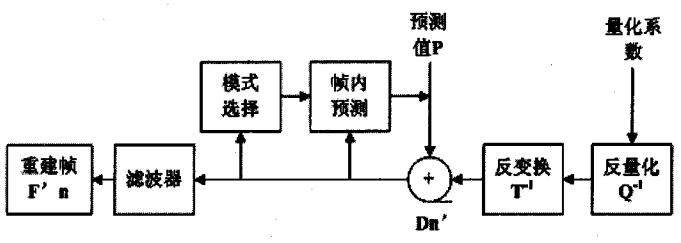

图 2　编码器重构通路

在前向通路中，以宏块为单位来处理输入的一帧数据 F_n，并以帧内或帧间方式来编码每个宏块，对宏块中的每个子块，基于重构的图像来产生预测值。帧内模式中，预测值从当前已经编码、解码、重构的宏块中产生，具体的搜索范围确定为当前编码宏块的左、左上和上三个宏块。帧间模式中，预测值从前一帧已编码图像通过运动补偿预测产生，运动估计采用钻石搜索算法。

从当前块减去预测值就得到残差数据 D_n，并对残差数据 D_n 进行变换，其中帧内 16 * 16 模式预测的宏块中的 DC 系数的 4 * 4 矩阵使用 Hadamard 变换，任何宏块的色度 DC 系数的 2 * 2 矩阵使用

Hadamard 变换,所以其他类型的残差数据使用整数 DCT 变换,对变换系数量化后得到量化变换系数,再进行重排序和熵编码(CAVLC),对熵编码系数与辅助信息进行 NAL 编码,得到输出的 H.264 码流。

在重构通路中,重构宏块是为了提供下一编码宏块的预测。对量化系数进行反量化,反变换产生一个差分块 D_n',预测值加上差分块就得到一个重构的块。一系列的重构块就得到一帧重构的参考图像。

代码的平台无关的裁剪包括和平台相关的函数的去除和 C 语言级的优化。我们需要实现的是 Baseline 的编码,所以需要去除与 Baseline 不相关的的函数,还包括大量的统计、测试和输出统计结果的代码,这些表达编译性能和状态的代码对 DSP 平台毫无用处,可以直接去除。C 语言级的优化,通常指优化程序代码和程序执行的速度。常用的优化方法有:使用内联函数;尽量减少全局变量,使用多的局部变量;减少判断语句;规范数据类型—因为定点 DSP 芯片只支持 4 种数据类型:short(16bit)、int(32bit)、long 型(40bit)和 double 型(64bit),因此必须对数据进行重新规范,把浮点数的运算部分近似用定点表示,或者用定点实现浮点;循环技术变量应该定义成为无符号的 int 类型,以避免不必要的符号的扩展等等。

用裁剪以后的代码编码视频流文件,得到的 264 文件经 H.264 标准的参考软件 JM11.0 解码后,可以正确播放,验证了算法的正确性。

2.1.2 DaVinci 平台上的移植

在移植的过程中,DM6446 开发板上运行的代码首先必须符合嵌入式的规则,在进行编程的时候要特别注意。很多 PC 机上的应用特点在 DSP 上面完全不实用,所以要做相应修改。具体移植步骤:

(1)库文件和头文件的改动:CCS(Code Composer Studio)是 TI 公司的专用于 DSP 开发的环境,其支持库和 DSP 硬件相关,与 PC 下 VC 的库函数不尽相同,所以,对 VC 的部分库函数及其包含的头文件做出了相应的改变,使之适应 CCS 开发和运行环境。还有其他一些函数如计时器函数,在 CCS 下与 VC 下是截然不同的,根据 CCS 支持的库函数和文件对原来的代码进行了修改。

(2)变量存取方式的调整:在 CCS 中,程序按照段式存储,不同功能的代码在编译连接时将被放在不同的段,各个段由配置文件中不同的段名标识。对于 C64x+的核,各主要段段名及其存放的代码内容如表 1 所示。将平台无关的 H.264 编码器代码在 CCS 中按照表 1 的要求安排,同时,为了移植的简单易行,将所有的段都放在 DDR2 存储空间中,并验证 H.264 编码器的正确性。

表 1 C64x+各段名称和存放内容

段 名	存放代码内容
.text	可执行的代码
.cinit	初始化全局变量和静态变量表
.switch	分支跳转表
.bss	静态和全局变量
.far	声明为 far 调用的全局和局部变量
.stack	系统堆栈
.system	动态存储空间分配堆

(3)存储空间的分配:在 CCS 下,程序存储空间的分配是通过配置.cmd 文件实现的,存储空间分配前,必须了解芯片整个可用的内外存储空间的大小,还要熟悉 H.264 数据流的情况,以便最高效合理的利用 DSP 缓存,达到最快、最理想的速度。

3 H.264 编码器在 TI DaVinci 平台上的优化

本文主要采用系统级优化和并行汇编优化的两级优化方案,对移植到 TI DaVinci 平台的 H.264 编码器进行优化。

3.1 系统级优化

系统级优化是在移植成功之后首要考虑的问题。系统级优化的主要思路是通过并行计算技术来充分利用 DSP 的硬件资源,提高编码性能,以便尽可能满足实时性的要求。

3.2 汇编优化

线性汇编语言是 TMS320DM6000 中独有的一种编程语言,介于高级语言和汇编语言之间。线性汇编语言的指令系统和汇编语言的指令系统完全相同,但是它有自己的汇编优化器指令系统,使用线性汇编语言时不需要考虑指令的延时、寄存器的使用和功能单元的分配,完全可以按照高级语言的方式进行编写。但它的代码效率远远高于高级语言,而且开发难度和开发周期比汇编语言要小得多。

线性汇编程序一个周期执行一条指令,简洁明了,较容易读懂。但是一个周期只执行一条指令的话,并不能充分利用 DSP 的资源。TMS320DM6446 DSP 的 C64x+核提供了 8 个功能单元,那么一个指令周期数最多可以并行执行 8 条指令。因此,应该尽量利用 DSP 核提供的功能,对核心的功能模块进行并行汇编优化,用最少的指令周期数实现 C 程序的功能。

3.2.1 汇编优化基本思路

C64x+的核提供了两个通用目的寄存器文件 A 和 B,有 A0～A31 和 B0～B31 总共 64 个寄存器。还提供了 L1、L2、S1、S2、D1、D2、M1、M2 八个功能单元,LD1、LD2 两个读数据通路和 ST1、ST2 两个存数据通路,1X 和 2X 两个交叉通路。每个功能单元可以直接与所处的数据通路的寄存器组进行读写操作,即. L1、. S1、. M1、. D1 可以直接读写寄存器组 A,而. L2、. S2、. M2、. D2 可以直接读写寄存器组 B。两个寄存器组可以通过 1X、2X 交叉通路也可以和另一侧的功能单元相连。

8 个不同的功能单元决定 DSP 可以同时处理 8 条并行指令,因为充分的利用这种指令的并行性,可以很大程度上节省程序的执行时间。由于篇幅原因汇编优化过程中关键指令分析不再论述。

3.3.2　DCT 代码的汇编实现

DCT 变换可以用公式:$\boldsymbol{Y}=\boldsymbol{AXA}^T$;$\boldsymbol{X}=\boldsymbol{A}^T\boldsymbol{YA}$ 表示,其中,$\boldsymbol{X}$ 表示图像或残差值的矩阵,$\boldsymbol{Y}$ 是相应的频率点上的 DCT 系数矩阵,其中,4 * 4 变换矩阵 $\boldsymbol{A}$ 中的系数为:

$$\boldsymbol{A}=\begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix}$$

其中 $a=c=1, b=2$。

根据 DCT 的特点和 C64x+核提供的指令的功能特点,确定 DCT 模块的汇编优化的思路如下:

DCT 块之间的并行运算:直接以 8 * 8 子块为单位,把 8 * 8 的子块分为 4 个 4 * 4 的子块,同时对 4 个 4 * 4 子块进行 DCT 变换;

矩阵相乘的并行性:在每个子块的运算中,充分利用 C64x+的指令系统,合理安排矩阵乘法的计算顺序,以最少的时钟周期完成 DCT 变换。

C64x+的寄存器的使用有其严格规范。C64X+的核提供了 A0～A31,B0－B31 一共 64 个 32 位寄存器。在 C/C++环境下,一些具体的操作要使用哪些寄存器来完成,是有严格规定的。寄存器使用规范规定了编译器使用寄存器的方法以及函数调用过程中数值保存的方法。要在 C/C++程序中嵌入汇编语言,必须理解并遵循寄存器使用规范。

(1)C 程序和汇编代码的接口实现

调用约定:函数(父函数)在调用另一个函数(子函数)的时候,将传递到子函数的参数放入寄存器或堆栈。A4、B4、A6、B6、A8、B8…是用于传递参数 1,2,3,4,5,6…. 的。函数名(参数 1 参数 2 参数 3 参数 4 参数 5…)

```
int function(int a,float b,char * c,long d,struct A…);
                A4   B4     A6      B6     A8   B8
```

A3:返回结构体指针,B3:返回地址寄存器,A15:帧指针(Frame Pointer,FP),B15:栈指针(Stack Pointer,SP),B14:数据页指针寄存器

(2)并行汇编程序结构

并行汇编程序,其文件的扩展名是. asm。在汇编中对函数定义,用到. global,. def 这两个定义宏。比如:. global _h264_dct4 * 4, . def _h264_dct4 * 4

(3)读数操作部分

在函数体中,首先要把 C 函数的参数传入。对于函数 cvs_dct_luma_input(p_src,p_dst,&dct4 * 4[1]);p_src 是指向源操作数的指针,p_dst 是指向目的操作数的指针,该函数需要将源操作数和目的操作数相减,放入数组 dct4 * 4[16 * 16]中。那么就需要用 LDNDW 语句把参数 1 和参数 2 里的数传入到寄存器中。LDNDW . D1T1 * A4,A9:A8 LDNDW . D2T2 * B4,B9:B8

(4)功能实现部分

实现部分就是用汇编指令实现原函数的功能。常用的指令如 ADD2、SUB2、PACK2、PACKH2 等。需要注意的是各个指令所限定的功能单元和延迟周期,合理地进行并行性安排。

(5)存数部分

对数据操作完毕后,需要将数据返回给主调函数的参数,这通过 STNDW 指令来实现。STNDW . D2T1 A29:A28, * B4,STNDW 指令不需要延迟。

(6)返回函数功能

数据存储完毕需要从汇编返回到主调函数,这通过 RET 指令来实现。RET 指令之后要经过 5 个

指令周期的延迟才返回，所以应当保证 RET 之后应该还有五个指令周期，这样函数方可正确返回。在返回之前要把所有的数据保存，并且和 C 中相关的寄存器不能改动，5 个延迟之后返回。

```
RET. S2 B3
STDW . D2T1 A29:A28, * +B4
STDW . D2T1 A31:A30, * +B4
NOP3
```

3.3.3 DCT 优化以后的性能分析

表 2 DCT 变换的实验结果比较

	C 语言的 8 * 8DCT	4 个 4 * 4 汇编	8 * 8DCT 汇编
时钟周期数	3 846	104	73

实验证明，8 * 8 的 DCT 变换，需要 73 个时钟周期，而 4 * 4 的 DCT 变换，需要 26 个时钟周期，C 语言的 8 * 8 的 DCT 变换，需要 3 846 个时钟周期，如表 2 所示。

从实验结果可以看出，通过指令优化，DCT 变换的效率提高了 50 倍左右。而 8 * 8 的 DCT 变换比 4 个 4 * 4 的 DCT 变换效率提高了，是因为增加的时钟周期，主要是以 4 * 4 子块作为单位来进行 DCT 变换的时候，数据的相关性比较高，这样导致执行时指令的并行性不够，取数指令的效率也没有完全发挥出来。而采用 8 * 8 为一个单元做 DCT 变换的时候，4 * 4 块之间没有数据相关性，可以充分安排好数据的流水线，避免了指令周期的浪费，充分发挥了 DSP 的优势。

4 实现结果

我们对 264 中耗费时间比较长的模块都做了并行汇编的优化，从很大程度上提高了编码性能，实现并行汇编的模块如下：

(1)DCT 部分：dct8×8，idct8×8，dct4×4 _ dc，idct4×4 _ dc

(2)量化部分：quant4×4，dequant4×4，quant4×4 _ dc，dequant4×4 _ dc，quant2×2 _ dc，dequant2×2 _ dc

(3)扫描部分：zigzag4×4，zigzag4×4 _ full

(4)SAD 部分：sad，satd

(5)预测部分：predict _ 16×16 _ dc，predict _ 16×16 _ dc _ left，predict _ 16×16 _ dc _ top，predict _ 16×16 _ dc _ 128，predict _ 16×16 _ h，predict _ 16×16 _ v，predict _ 16×16 _ p。

在实现了系统级的优化，以及 DCT、SAD、量化、扫描、预测等关键模块的并行汇编优化之后，实验结果如表 3 所示：

表 3 实验结果比较

测试序列	汇编后的编码时间(cycle)	汇编前的编码时间(cycle)	前后对比
akiyo. yuv	5 492 970	38 450 842	1:7.01
news. yuv	5 161 344	37 129 612	1:7.19
foreman. yuv	5 766 493	43 325 487	1:7.51
football. yuv	7 549 885	49 840 195	1:6.60

在视频监控中，我们的要求是达到 D1(704×576)的实时，相当于 4 路 cif 的实时。PAL 制的实时要求是 25 帧/秒，因此 4 路 cif 的时候就要求每秒至少能够处理 4×25=100 帧 cif 图像。

DaVinci 平台的 DSP TMS320 DM644 的主频是 594Mhz，594/100 = 5.94(Mhz)

因此处理每帧图像的 cycle 数要限制在 5.94 MHz 以内。从目前的优化结果来看，我们的编码器达到了这个要求。

5 结 论

本文主要采用系统级优化和并行汇编优化的两级优化方案，对移植到 TI DaVinci 平台的 H.264 编码器进行优化，实验证明，优化后的编码速度提高了将近 100 倍，能达到 4cif 的实时编码，为基于 DSP 嵌入式系统的 H.264 视频编码算法实现奠定了基础，这对今后进行更深层次 H.264 算法的理论研究及开发 H.264 算法内核并将其移植到其他 DSP 芯片上开发产品，都有重要的意义。

参考文献：

[1] 崔海燕，王卿. 基于 ADSP—BF561 的 H.264 视频编码器的实现[J]. 电子元器件应用，2008，10(4)：29－33.

[2] 李娓江，陈抗生. 一种 H.264 视频流自适应率失真优化编码算法[J]. 电路与系统学报，2005(6)：33－36.

[3] 朱琪. H.264/AVC 编解码算法分析与优化研究[D]. 武汉理工大学，2006.

[4] 王晓慧. AVS－M 编码器算法研究和解码器 DSP 移植和优化[D]. 中国海洋大学，2006.

[5] 汪毅. H.264 视频编码技术研究及在 DM642 上的移植与优化[D]. 武汉理工大学，2007.

[6] 郭建军. H.264 视频编码若干关键优化技术的研究与实现[D]. 国防科学技术大学，2005.

(下转第 38 页)

意义。

参考文献:

[1] Jiawei Han, Micheline Kamber. 数据挖掘概念与技术[M]. 范明,孟小峰,译. 机械工业出版社,2001.

[2] Borges J, Levene M. Data mining of user navigation patterns[C]. In: Proceedings of the 1999 KDD Workshop on Web Mining, CA: SpringerVerlag Press, 1999: 92－111.

[3] R Sarukkai. Link Prediction and Path Analysis Using Markov Chains[J]. Computer Networks, 2000, 33(1/6): 377－386.

[4] M Deshpande, G Karypis. Selective markov models for predicting web page Accesses[J]. ACM Transaction on Internet Technology, 2004, 4(2): 163－184.

[5] Jianhan Zhu, Jun Hong, John G Hughes. Using Markov Models for Web Site Link Perdiction[C]. In: Proceedings of the thirteenth ACM conferenceon Hypertext And hypermedia. Maryland(USA), 2002, 169－170.

[6] 周晟,俞建家. 基于动态马尔可夫模型的智能网页推荐[J]. 计算机工程与应用,2006(S1):84－87.

[7] 邢永康,马少平. 多马尔可夫链用户浏览预测模型[J]. 计算机学报,2003,26(11):1510－1517.

[8] 许欢庆,王永成. 基于用户访问路径分析的网页预取模型[J]. 软件学报,2003,14(6):1142－1147.

[9] 赵亮,胡乃静,张守志. 个性化推荐算法设计[J]. 计算机研究与发展,2002(8):986－991.

[10] 叶海琴,石磊,王意锋. 基于网络访问行为的混合阶 Markov 预测模型[J]. 计算机工程与设计,2008,29(2):333－336.

[11] 王实,高文等. 基于隐马尔可夫模型的兴趣迁移模式发现[J]. 计算机学报,2001(2):152－156.

Research of Markov Prediction Model and Algorithm Based on Web Log Mining

ZHANG You-zhi, CHENG Yu-sheng, WANG Yi-bin

(School of Computer and Information , Anqing Teachers College, Anqing 246133, China)

Abstract: Through Web log mining technology , We can analyze user's browsing patterns and provide intelligent and personalized service. The chain structure of Markov model is convenient and easy for forecasting user's browsing patterns. This paper studies for Markov prediction model deeply, introduces it's realizable algorithm and validates the method of mixed Markov model through an example.

Key words: web log mining, browsing pattern, Markov model, chain structure, algorithm

(上接第 34 页)

Optimization and Transplantation of H. 264 Video Coding Algorithm Based on TMS320DM6446

CHEN Hong-fang, ZHAO Sen-yan

(1. Anhui University of Technology and Science; 2. Wuhu Institute of Technology, Wuhu 241000, China)

Abstract: This article elaborates TMS320DM6446 hardware platform and software architecture. The platforms is based on x264 code, and uses the appropriate methods to cut the original code with platform－independent under ensuring performance for its practical application. It focuses on the transplantation coding encoding process, system－level optimization and compilation of class, and gives the results of the corresponding optimization.

Key words: video coding, optimization, transplantation