第34卷 第22期 计 算 机 工 程 2008年11月
Vol.34 No.22 Computer Engineering November 2008
·开发研究与设计技术· 文章编号：1000—3428(2008)22—0258—03 文献标识码：A 中图分类号：TN911.73

图形处理器的流执行模型

李海燕，张春元，李 礼，任 巨
(国防科技大学计算机学院，长沙 410073)

摘 要：图形处理器极高的流计算能力使其成为实现实时流应用的有效方案。该文抽象出图形处理器的流执行模型，描述图形处理器流处理机制的执行过程，在图形处理器上实现了二维离散余弦变换。实验结果表明，图形处理器对标清格式的视频压缩编码效率可达70 fps。
关键词：图形处理器；流处理；流执行模型

Stream Execution Model of Graphics Processing Unit

LI Hai-yan, ZHANG Chun-yuan, LI Li, REN Ju
(School of Computer, National University of Defense Technology, Changsha 410073)

【Abstract】 Graphics Processing Unit(GPU), which has powerful stream computing capability, makes itself as an efficient scheme for real-time stream applications. This paper abstracts a stream execution model of GPU, describes its stream execution process, and implements the discrete cosine transform on GPU. Experimental results show that the coding efficiency of standard definition video compression on GPU can reach 70 fps.
【Key words】 Graphics Processing Unit(GPU); stream processing; stream execution model

1 概述

随着多核时代的到来，流处理已成为主流计算方式之一，它能很好地解决流应用领域的一些相关问题，例如图形绘制、图像压缩、数字媒体回放等，这些媒体应用具有显著的流特征，即计算密集性、并行性与“生产者-消费者”局部性。流处理流程如下：将处理元素描述为流，将处理过程描述为流互连的核心，核心的各个功能单元簇组同时对输入流中的每个元素执行大量相同(SIMD模式)操作或不同(MIMD模式)操作，生成的结果以流的形式输出。在流处理计算方式中，计算被显式地分割为多个计算核心，将密集计算集中在核心内部，有利于开发核心内部的局部性与并行性。流处理将访存从计算中分离出来，以流的形式组织数据，可以发现核心之间的“生产者-消费者”局部性，有利于挖掘数据级并行与任务级并行。

流应用对实时性与高性能的需求日益增长，而集成电路制造工艺水平的不断进步，为流处理器的设计与实现提供了广阔空间。基于流处理思想，出现了大批流处理器体系结构，如Imagine[1], Cell[2]等。图形处理器(Graphics Processing Unit, GPU)也可以看作是一种流处理器。NVIDIA公司与AMD公司陆续推出了自己的流处理器。Owens[3]在NVIDIA公司出版的《GPU Gem2》一书中清楚地将图形处理器与流处理器联系在一起，论述了GPU在流处理方面的能力与潜力。基于GPU的通用计算近年来取得了巨大突破，GPU正在向通用流处理器的方向发展[4]，它在价格与市场方面均有很大优势，因此，充分利用GPU来开发实时高效的流应用具有重要 意义。

2 GPU流执行模型

GPU有2个流处理部件即可编程并行处理部件：顶点处理器(Vertex Processor)和片断处理器(Fragment Processor)。从体系结构角度来看，顶点处理器是MIMD处理单元，片断处理器是SIMD处理单元。

2.1 流执行模型

如图1所示的GPU流执行模型包含6个主要功能模块：

(1)主处理器。区别于流处理单元而言，将负责标量计算以及负责与流处理单元进行指令和数据传递的处理器部件称为主处理器(Host Processor)。主处理器执行流程序，流出流指令，控制指令由主处理器下达到流处理单元中的控制单元，待程序执行结束时，将完成信号返回到主处理器。流处理单元可以看作是主处理器的协处理器。

(2)主控制器。作为流处理单元中的流控制单元(Stream Control Unit)，在功能上类似于超标量处理器的流出单元[5]。它接收由主处理器发送过来的流指令，并控制指令发射到流处理单元的其他部件。当指令队列中的指令相关性得到满足时，主控制器发射该指令给相应功能模块(如顶点/片断处理器、纹理存储空间等)执行。

(3)顶点/片断处理器。是流处理单元中的核心执行单元(Kernel Execution Unit)，主要由ALU运算单元(包括加法器、乘法器等)和本地寄存器文件单元组成。顶点/片断处理器受控于主处理器发射的流指令，通常是对输入流中的每个元素以循环的形式执行运算操作，从而生成结果流。顶点/片断处理器中流的读取与保存都在纹理存储空间中进行。

(4)纹理存储空间。专门负责流存储的地址空间即片上流寄存器文件(On-chip Stream Register File)。它能为核心提供计算所需的输入流及保存核心运算结束后的输出流，是片外存储器和本地寄存器之间的桥梁存储空间，可以充分捕获计算

基金项目：国家自然科学基金资助项目(60673148)；高等学校博士学科点专项科研基金资助项目(20069998025)
作者简介：李海燕(1981－)，女，博士研究生，主研方向：高性能微处理器体系结构与应用；张春元，教授、博士生导师；李 礼、任 巨，博士研究生
收稿日期：2008-04-27 **E-mail：**hy_lee@163.com

核心之间的“生产者-消费者”局部性，在流执行过程中发挥着重要作用。

(5)存储分割。将大块数据集合分割为若干子集再传送回片外存储空间。相关功能模块包括存储控制器、地址生成器等，可称其为存储系统接口(Memory Interface)，它们负责片上流存储空间与片外存储系统之间的流数据传输。

(6)DRAM(s)。是 GPU 的全局存储空间，相对于流执行单元而言是片外存储系统(Off-chip Memory)。在流处理过程中，DRAM(s)用于存放流应用的输入输出流和无法存放在片上流寄存器文件中的中间结果流。

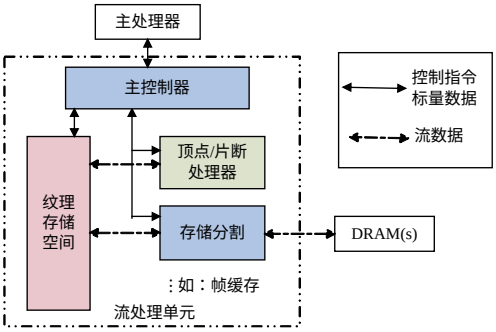

图 1　GPU 流执行模型

在 GPU 流执行模型中，流是一组同一数据类型记录有序组成的集合，如视频编码中的像素点；顶点处理器或片断处理器对流进行有效计算，形成一系列计算核心，如视频帧内像素点颜色空间变换。在顶点处理阶段，流记录是空间的几何点，即顶点图元。在片段处理阶段，流记录是片段，本质上是像素。顶点处理器采用 MIMD 方式，片段处理器采用 SIMD 方式，从而使 GPU 具有很好的数据级并行特征。由于 GPU 的顶点程序和片段程序可以同时执行，因此数据传输可以与计算重叠，易于实现流水线。

2.2　流执行过程

GPU 流执行模型突出了流控制模块、流计算模块和流存储模块的功能，显式表示了主要功能模块之间的控制关系与数据流向。由图 1 可以获得 GPU 流处理机制的具体执行过程：

Step1 主处理器执行程序时遇到可以流化的程序段，转入流处理单元进行高效的并行操作；

Step2 流处理单元依据主控制器获取的流控制信息，分别调用其他功能模块开始工作；

Step3 纹理存储空间为核心准备流数据(可能需要通过片外 DRAM(s)补充流数据)；

Step4 准备好流数据后，将其加载到核心执行单元的本地寄存器中，启动顶点/片断处理器开始流计算过程；

Step5 计算结束后，核心的输出结果以流的形式传送回片上的纹理存储空间；

Step6 若流程序需要该输出流作为后续的操作对象，则以它作为输入流进入下一个核心；若该输出流不再参与流计算，则通过存储分割将其写回片外 DRAM(s)。

GPU 体系结构可以抽象为如图 1 所示的流执行模型，并将处理过程抽象为流与核心，这为 GPU 的流计算方式提供了有力证据。

3　离散余弦变换在 GPU 上的流实现

离散余弦变换(Discrete Cosine Transform, DCT)被广泛应用于数字通信、图像处理、视频压缩等领域。其算法高效、结构规律性强，适用于针对体系结构特征设计相应的快速算法。本文主要讨论用于图像、视频压缩编码的二维DCT[6]。

将一帧图像划分为 8×8 大小的子图像块，采用经典行列分解法，将二维 DCT 变换转换为 2 次一维 DCT 变换映射在 GPU 上，按先水平变换后竖直变换的顺序进行，即

$$\boldsymbol{Z}=\boldsymbol{C}\times\boldsymbol{X}\times\boldsymbol{C}^{\mathrm{T}} \tag{1}$$

其中，$\boldsymbol{C}$ 为变换矩阵；$\boldsymbol{X}$ 为待变换矩阵；$\boldsymbol{Z}$ 为变换后的矩阵。

对于一维 DCT 的计算，由于转置操作不适合 GPU，而 GPU 上有专门的硬件结构确保如乘加等特殊计算指令的执行速度能满足性能要求，因此本文选用直接矩阵乘的方法实现一维 DCT 计算。

在 GPU 实现中，将待变换的矩阵块与变换后的系数矩阵块分别组织成流，存放在纹理存储空间。纹理与视频帧在二维空间上是相互对应的。核心程序运行于片断处理器，充分利用片断处理器的 SIMD 模式开发数据级并行性。

GPU 的主要实现过程如下：

(1)绘制一个全屏幕的矩阵区域，初始化二维纹理数据，形成输入流格式。将整个 W×H 大小的帧图像划分为每 8 行一个片组，共 H/8 个片组，再将每个片组划分为 W/8 个 8×8 大小的待变换矩阵块，按行排列顺序自然地将待变换矩阵块组织成一条长度为(W/8)×(H/8)的输入流，其中每个流记录是一个待变换矩阵块。

(2)从纹理存储空间中获取流记录，采用轮转片方式将流记录加载到片断处理器的不同图形引擎(GE)中，以SIMD方式执行片段程序，即对每个流记录执行计算核心的操作。加载纹理数据时，采用基于行或列的带状加载方法，如图 2 所示，分别将左乘($\boldsymbol{C}$)和右乘($\boldsymbol{C}^{\mathrm{T}}$)的矩阵预先放置于 8×8 大小的纹理空间中。待变换矩阵($\boldsymbol{X}$)则按行或列划分的 8 个像素点宽的带状依次加载到片段处理器中进行计算，从而避免变换核心纹理数据重复加载的开销。计算核心主要描述 2 次矩阵乘法的程序，对输入流执行一次“读取-计算-写回”的混合操作，直至将计算结果输出，从而避免流在本地寄存器与纹理存储空间之间来回地进行传递。

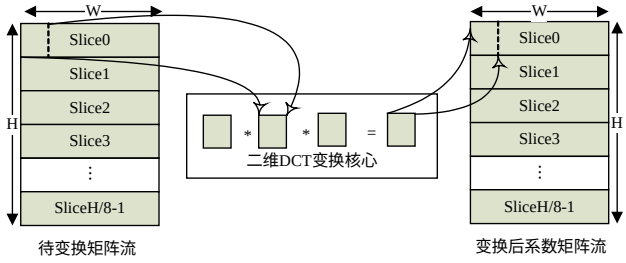

图 2　二维 DCT 变换的 GPU 实现方法

(3)将计算结果写入纹理存储空间并返回主存。输出流格式与输入流类似，每个变换后的系数矩阵块依次放入与输入流对应的位置。

4　性能分析与讨论

本文实验使用的PC机配置如下：AMD Sempron$^{\mathrm{TM}}$ 2600+ (1.84GHz)处理器，512 MB内存和NVIDIA GeForce FX 5700显卡。显卡具体参数指标如下：内存为 128 MB，核心频率为 425 MHz，带宽为 9.6 GB/s，显存位宽为 128 bit，存储频率为 600 MHz，总线接口为AGP 8X，顶点处理器数为 3，片段处理器数为 4。

分别对单精度浮点数的 QCIF(176×144), CIF(352×288), SD(704×576)这 3 种图像格式进行数据测试，得到如表 1 所示的 GPU 执行时间。

表 1　各种图像格式 GPU 实现的执行时间

图像格式	GPU 实现执行时间/μs
QCIF	15.5
CIF	65.9
SD	289.8

观察 DCT 变换在 GPU 上的实现可知，当视频帧大小增加时，处理一帧 DCT 变换的时间必然随之增长，但时间增长率越来越大。例如，与 QCIF 相比，CIF 图像格式的分辨率是其 4 倍，而 DCT 执行时间为 4.25 倍左右。这是由于表示视频帧的输入流变长后，流处理单元根据数据相关性和指令级并行性分配功能单元时引入了一些计算的停顿，使得处理时间的增长率略大于 4 倍。特别是当 GPU 处理 SD 图像格式视频帧时，执行时间增长率达到 4.4 倍，这与存储带宽受限有关。但这种类线性增长表明，GPU 的流处理计算方式在开发数据级并行性方面是很有效的。

GPU上的DCT实现在实验中取得了很好的性能，若将其用于MPEG-2, AVS等视频压缩应用中能满足实时性需求。例如，按DCT执行时间占MPEG-2 视频编码总时间的比例，可以推得对于SD格式的视频序列，MPEG-2 的编码效率可达 70 fps。实验结果有力地证明了GPU在除图形渲染任务之外的通用计算方面具有极大潜力，主要原因是GPU硬件本身对流应用具有很好的支持[4]，具体如下：当前GPU通常具有多个渲染管道和RGBA的 4 色通道，可以同时计算来自纹理的数据，具有很好的数据并行性；GPU显存位宽大于CPU内存位宽，因此，整个计算带宽大幅度提高，GPU相对于CPU更适应传输大块数据、实现高密集运算。这是GPU可以被视为流处理器的重要原因。GPU的优势在于处理流应用核心计算时，无须与CPU进行多次数据交换，可以将CPU解放出来去完成其他任务。

目前 GPU 发展很快，现阶段 GPU 性能远高于本文实验所用的 NVIDIA Geforce FX 5700。因此，如果能有效利用当前强大的图形处理器来加速视频应用，将在很大程度上提高消费者的视觉质量，具有巨大市场价值与实用意义。

5　结束语

本文抽象出 GPU 流执行模型，强调了 GPU 的流计算能力。它能满足当前视频应用的需求，为未来高清视频的应用打下很好的基础。

参考文献

[1] Khailany B, William J D, Ujval J K, et al. Imagine: Media Processing with Streams[Z]. 2001.

[2] Flachs B, Asano S, Dhong S H, et al. The Microarchitecture of the Streaming Processor for a CELL Processor[C]//Proc. of IEEE International Solid-state Circuits Symposium. [S. l.]: IEEE Press, 2005.

[3] Owens J. Streaming Architectures and Technology Trends[C]//Proc. of International Conference on Computer Graphics and Interactive Techniques. Los Angeles, California, USA: ACM Press, 2005.

[4] 吴恩华, 柳有权. 基于图形处理器(GPU)的通用计算[J]. 计算机辅助设计与图形学学报, 2004, 16(5): 601-612.

[5] 李　礼. 流体系结构存储访问机制的研究[D]. 长沙: 国防科技大学, 2006.

[6] 黄贤武, 王加俊, 李家华. 数字图像处理与压缩编码技术[M]. 成都: 电子科技大学出版社, 2000.

(上接第 257 页)

应用。当手机用户发送的查询文本传入到分词模块时，该语句被划分为：我想|知道|怎么去|迎春|路。系统通过对关键词数组循环搜索找出数据库中的迎春路一项，空间信息引擎渲染生成迎春路的空间信息，并以彩信的方式返回给手机用户。生成的图片资源见图 6，手机终端显示见图 7。

图 6　生成图片资源

图 7　手机显示

3　结束语

根据艾瑞集团公司的调查表明[8]，在 2006 年 11 月~2006 年 12 月网民使用的手机增值服务中，MMS是占有率最多的移动增值服务。随着 3G 网络的全面投入运营和 MMS 的普及和终端设备价格的降低，基于 MMS 和 WebGIS 的移动空间地理信息查询将成为移动增值服务的一大亮点。

本文搭建了一个即时、高效、准确的空间地理信息查询系统，是移动增值服务的可行方案，具有一定的现实和推广意义。

参考文献

[1] 中国移动通信集团公司. 中国移动手机定位业务 LBS 技术规范 V1.0.0[S]. 2002.

[2] 陆天波, 方滨兴, 孙毓忠, 等. 点对点匿名通信协议 WonGoo 的性能分析[J]. 计算机工程, 2006, 32(2): 26-28.

[3] 余　涛, 余　彬. 从 SMS、MMS 到 LBS——移动增值服务的发展趋势[J]. 移动通信, 2005, 29(10): 42-44.

[4] 孙德炜, 靳法伦, 殷惠康, 等. 行文管理系统的设计和实现[J]. 计算机工程, 1998, 24(5), 7-10.

[5] 中国移动通信集团公司. 中国移动通信互联网短信网关接口协议 V3.0.6[Z]. 2003.

[6] 中国移动通信集团公司. 多媒体信息业务(MMS)总体技术要求 V1.0.5[Z]. 2006.

[7] 陈士杰, 张玥杰. 基于 Lucene 的英汉跨语言信息检索[J]. 计算机工程, 2005, 31(7): 62-64.

[8] 艾瑞市场咨询有限公司. 中国移动增值服务市场研究报告[Z]. 2007.