

图形处理显示列表的设计与实现

刘 晖^{1,2}, 田 泽^{1,2}, 马城城^{1,2}, 张 骏^{1,2}, 薛凌艺³

(1. 中国航空工业西安航空计算技术研究所, 陕西 西安 710068;

2. 集成电路与微系统设计航空科技重点试验室, 陕西 西安 710068;

3. 西安翔腾微电子科技有限公司, 陕西 西安 710068)

摘 要:显示列表功能是图形处理器 3D 处理引擎的核心功能, 通过将一段图形绘制指令存储到图形处理器内部, 来完成复杂场景下同一物体的重复绘制。该功能极大地减少了主机与图形处理器之间的数据吞吐量, 降低数据带宽的压力、高效地绘制复杂场景。文中基于 Xilinx XC7VX1140T 构建 FPGA 原型系统, 参考 ALT-M9 芯片的实现能力, 提出了一种软硬件协同方式的显示列表设计实现方法。该方法充分利用了 CPU 与 GPU 的处理能力, 灵活实现了图形处理命令在不同实现方式下显示列表的设计与实现。验证结果表明, 软硬件协同方式实现的显示列表性能优越, 空间分配管理灵活, 可靠性高。

关键词:图形处理器; OpenGL; 显示列表; 空间管理

中图分类号: TP39

文献标识码: A

文章编号: 1673-629X(2016)04-0119-04

doi:10.3969/j.issn.1673-629X.2016.04.026

Design and Realization of Display List Function in Graph Process

LIU Hui^{1,2}, TIAN Ze^{1,2}, MA Cheng-cheng^{1,2}, ZHANG Jun^{1,2}, XUE Ling-yi³

(1. Aeronautical Computing Technique Research Institute, Xi'an 710068, China;

2. Key Lab of Aeronautics Science and Technology of Integrate Circuit and Micro-system Design,
Xi'an 710068, China;

3. Xiangteng Micro-electronics Technology Co., Ltd., Xi'an 710068, China)

Abstract: Display list is the core function of 3D process engine in GPU, through storing a number of graph commands to GPU to complete repeat drawing for the same object in complex scene. It reduces the amount of data between CPU and GPU in a large degree, and lows pressure of data bandwidth and draws complex scene efficiently. Based on FPGA system built by Xilinx XC7VX1140T, taking a consideration in the capability of ALT-M9 chip, a method of collaborating hardware and software in display list design and realization is proposed. This method makes full use of the ability with CPU and GPU to complie the display list flexibly based on different way in graph commands realization. Verification results show that performance of collaborating hardware and software in display list is superior, with flexible the space allocation management and high reliability.

Key words: GPU; OpenGL; display list; memory management

0 引 言

显示列表功能广泛应用于物体重复绘制、物体运动轨迹描述、仪表盘绘制等复杂场景绘制中, 是一种简单、高效的使用方式, 极大地降低了 CPU 与 GPU 之间的带宽压力, 提高了编程的效率与灵活性^[1]。显示列表已成为图形处理器应用的一个新方向。

显示列表分为编译模式和编译执行模式。在编译

模式下, 图形绘制指令仅存储到图形处理器内部; 在编译执行模式下, 图形绘制指令存储到图形处理器内部的同时发送给 3D 处理单元进行执行。传统的硬件实现方式灵活性较差^[2], 浪费了主机处理器的性能, 限制了图形处理命令的实现方式。文中提出了一种软硬件协同的实现方式, 其适用范围不局限于图形处理语言和图形处理命令的实现方式^[3]。

1 显示列表原理

在图形处理器的应用中显示列表用于多次绘制同一场景或物体,其原理如图 1 所示。

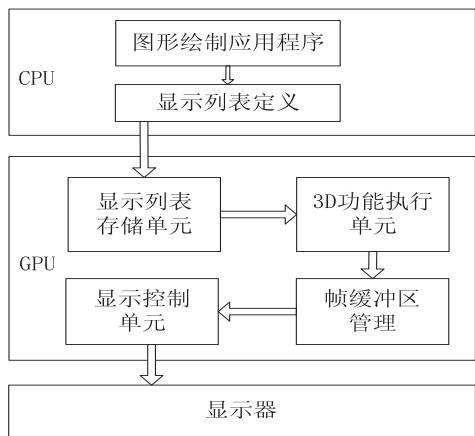


图 1 显示列表原理

当图形处理器收到除显示列表外的图形绘制应用程序时,直接将图形处理命令发送到 3D 功能执行单元进行处理。当图形处理器收到定义显示列表命令,并且当前定义的显示列表为仅编译模式时,图形处理命令发送到显示列表存储单元,当前显示列表中包括的图形绘制命令仅做存储,对当前的图形绘制没有影响。当图形处理器收到定义显示列表命令,并且当前定义的显示列表为编译执行模式时,图形处理命令会同时发送给显示列表存储单元进行存储和 3D 功能执行单元进行命令执行。当图形处理器收到调用显示列表命令时,图形处理器从显示列表存储单元获取对应的显示列表命令,并将其发送到 3D 功能单元执行。当图形处理器收到删除显示列表命令时,图形处理器释放对应显示列表的存储空间。

1.1 传统的显示列表实现方式

传统的显示列表实现方式中,每一条图形处理指令都在对应的功能单元执行。这样的实现方式方便了显示列表的存储、调用,但却在很大程度上增加了图形处理器的硬件资源开销,增大了芯片的整体功耗^[4],而且浪费了主机处理器的处理性能。

例如曲线曲面功能,需要根据用户规定的控制点信息,通过 Bernstein 多项式计算出绘制点信息,并按照点的处理方式进行剪裁、消隐、归一化等处理并最终绘制出来。若采用硬件方式实现,需要实现阶乘等超越函数,对硬件资源的开销较大,浪费了主机处理器的处理性能^[5]。

1.2 软硬件协同方式

HKM9000 图形处理器是中航工业计算所自主研发的一款功能性能与 ALT-M9 相当的图形处理器。文中以 HKM9000 图形处理器为设计载体,设计方法的应用不局限于某款图形处理器。

在 HKM9000 图形处理器设计中,将图形处理命令进行分类,曲线曲面功能、顶点数组功能及查询类相关功能由主机软件实现;其他功能由硬件电路实现,因此需要将显示列表命令分为软件记录 and 硬件记录。软件记录占用主处理器的空间,可由用户分配,硬件记录占用图形处理器内的存储空间,由显示存储单元分配空间。这种设计方式极大地提高了图形指令设计的灵活性。

2 基于 HKM9000 图形处理器的显示列表设计与实现

2.1 显示列表数据管理

显示列表存储数据包括硬件数据节点和软件数据节点。软件数据节点需要进行软件空闲节点池管理,记录软件节点函数、函数参数及后续节点号;硬件数据节点需要进行硬件空闲节点池管理,记录硬件节点地址和数据量及后续节点号。

对软件记录进行数据管理如下所示:

```
typedef struct
{
    /* 指向软件实现函数的函数指针 */
    GLVoid    (* Func_Pointer) (GLvoid);
    /* 函数指针映射表的项数 */
    GLint     Func_Pointer_num;
    /* 记录指向函数的参数 */
    GLfloat   Params[SW_API_PAMS_NUM];
    /* 指向当前软件记录节点的下一个节点 */
    GLushort  Next;
} ST_SW_NODE;

typedef struct
{
    /* 记录显示列表中空闲软件节点个数 */
    Glushort  SW_Idle_Cnt;
    /* 记录显示列表中空闲软件节点头 */
    Glushort  SW_Idle_Head;
    /* 记录显示列表中空闲软件节点尾 */
    Glushort  SW_Idle_Tail;
    /* 记录软件实现的 API 在显示列表中的信息 */
    ST_SW_NODE SW_Node[SW_API_SIZE];
} ST_SW_NODE_INFO;
```

对硬件记录进行数据管理如下所示:

```
typedef struct
{
    /* 保存显示列表号 */
    GLuint    NewList_Num;
    /* 保存对应的首节点号 */
    Glushort  Node_Num;
    /* 显示列表是否可用 */
    GLboolean NewList_Enable;
} ST_NEWLIST_NODE;

typedef struct
{
    /* 当前节点中存储的 API 个数 */
```

```
GLubyte  Size;  
/* 指向当前节点的下一节点 */  
GLushort Next;  
} ST_CALLLIST_NODE;
```

2.2 显示列表资源

经测试,ALT-M9 内部能存放约 200 万条 OpenGL 命令,所以在 HKM9000 图形处理器内部分配 50 MB 的存储空间,最多能够存储 1 638 400 条图形处理指令,其处理能力与 ALT-M9 相当。

显示列表可供支配的资源如图 2 所示,包括

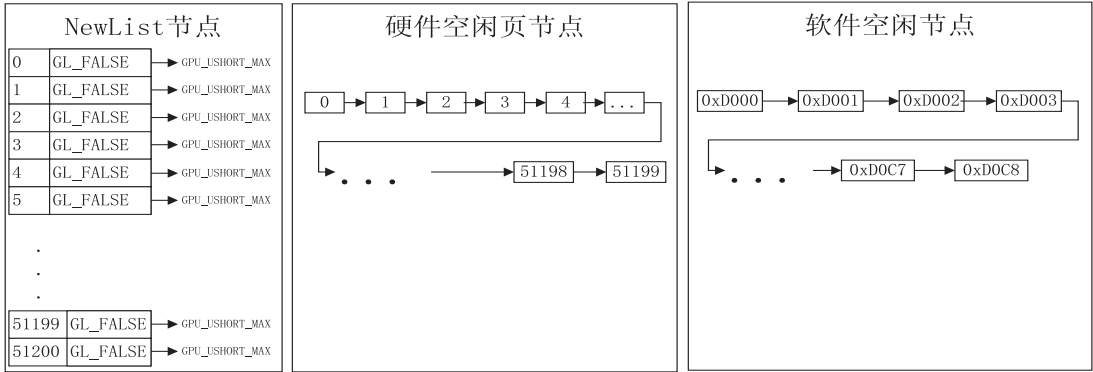


图 2 显示列表资源

2.2.2 硬件空闲页节点

为了方便存储空间寻址和内存管理,采用页式管理的方法,将 50 MB 的 GPU 存储空间划分为 50K 个 1 kB 大小的页,每一页最多可存储 32 条图形指令,且每一页只能对应唯一的显示列表。当多个节点对应一个显示列表时,各节点按照定义顺序依次组成节点链^[6]。

硬件空闲节点池由头尾指针维护,显示列表定义时从头指针处获取硬件空闲节点,显示列表删除时从尾指针处添加已释放的空闲节点^[7]。

2.2.3 软件空闲节点

软件空闲节点的使用及管理方式与硬件空闲页节点类似,用来记录软件方式实现的显示列表函数,软件节点占用主处理器的存储空间可在使用时灵活调整。

2.3 显示列表设计与实现

显示列表各接口按照实现功能分为软件实现和软硬件协同实现。

2.3.1 glGenLists 函数

GLuint glGenLists (GLsizei range) 图形命令由软件实现,返回在 0 到最大整数之间 range 个连续的数,作为可用现实列表号的选择。

2.3.2 glNewList/glEndList 函数

void glNewList (GLuint list, GLenum mode)/ void glEndList (void) 图形指令由硬件实现,它们指定了当前要声明的显示列表及显示列表内包含的图形指令。首先在 NewList 节点中查找下标为 list 的节点中存储的显示列表号,若与当前定义的显示列表号相同,则删

51 201个可被记录的显示列表 NewList 节点,51 200 个硬件记录的页,及 2 000 个软件记录的函数节点。

2.2.1 NewList 节点

当定义显示列表时,首先判断当前的显示列表号在 NewList 节点中是否已被占用:若已被占用但节点号不相同,则当前显示列表定义不成功;若已被占用且节点号相同则需要先删除之前的显示列表释放占用空间,再记录当前定义的显示列表;若节点号未被占用,则当前显示列表可定义。

除之前定义的显示列表并释放其占用的空间节点,否则当前 NewList 节点被占用,显示列表定义失败。

显示列表空间按照页式管理,每个显示列表所占用的空间可在 NewList 节点号对应的后续节点中查找。为方便显示列表的数据调用与地址查找^[8],更新图形指令接口如下: void glNewList (GLuint addr, GLenum mode, GLuint size)。其中,addr 表示当前使用的节点号,mode 表示当前显示列表类型,size 表示当前节点号中包括的命令条数。显示列表按照 API 的实现方式来分配当前的显示列表空间。

2.3.3 glListBase 函数

void glListBase (GLuint base)由软件实现,它将显示列表索引数组按照指定的偏移进行排序,并最终由显示列表调用。

2.3.4 glCallList/glCallLists 函数

void glCallList (GLuint list)/void glCallLists (GLsizei n, GLenum type, const GLvoid * lists)用于调用显示列表,glCallList 函数功能由硬件实现,根据调用的显示列表号来查找 NewList 节点中对应的节点数据。若与调用的显示列表号相等,查找后续链接的节点并依次调用,调用的函数接口更新为: void glCallList (GLuint addr, GLuint size),更方便于数据调用与地址查找。glCallLists 封装 glCallList 实现^[9],依次按照显示列表索引数组获得当前调用号并查找 NewList 节点计算。

2.3.5 glDeleteLists 函数

void glDeleteLists (GLuint list, GLsizei range) 功能

是删除已定义的显示列表。它由软件实现,先查找显示列表索引判断当前的显示列表是否存在,对于存在的显示列表,释放链接节点的地址空间,并在软/硬件空闲节点池中存储,以备后续调用。

2.3.6 glIsList 函数

GLboolean glIsList(GLuint list) 功能是查询当前号是否为一个显示列表号。它由软件实现,通过遍历 NewList 节点查找是否有对应的号,若有则当前号是一个显示列表号,否则不是。

3 显示列表测试

为评估以上设计中显示列表的功能及性能,搭建 HKM9000 FPGA 验证平台与 ALT-M9 图形处理器进行对比实验。

HKM9000 FPGA 原型测试平台如图 3 所示。

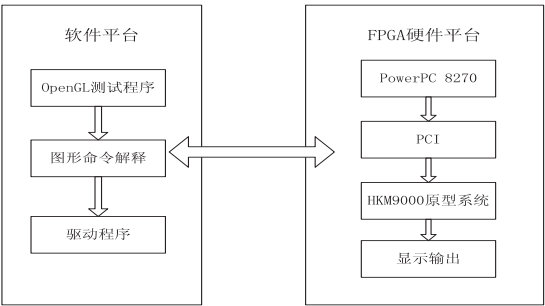


图 3 HKM9000 FPGA 验证平台

将经过编译、综合后生成的逻辑 bin 文件加载到 FPGA 原型系统中^[10],主机端运行图形绘制软件,图形处理驱动程序将上层的图形应用程序翻译为硬件可识别的命令码流,通过 PCI 总线将命令码流传输给 HKM9000 原型系统,并在其上执行功能,最终的结果由显控单元输出到显示器上^[11]。

软硬件测试环境如表 1 所示。

表 1 软硬件测试环境

软(硬)件名称	标识或版本	用途
VxWorks	5.5 及以上版本	操作系统
Tornado	2.2 及以上版本	程序编译及运行环境
OpenGL	1.3 版本	图形处理接口
PC 机	CPU: Inter Core 2 Duo	编写测试用例, 执行测试程序
	E7400 2.80 GHz	
	内存: 1.99 GB	
	系统: Windows XP SP3	
FPGA	XC7VX1140T	搭建测试环境
PCI	V 2.2	传输图形指令
PowerPC	PPC8270	搭建测试环境
电源	EL871103	供电电压: 5 V 供电电流: 6 A
显示器	电脑显示器	显示输出图形

3.1 功能测试

功能测试结果如图 4 所示。

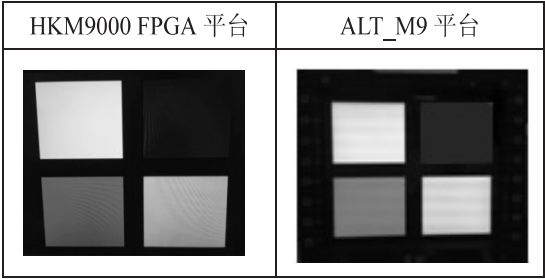


图 4 显示列表功能测试效果图

在 HKM9000 FPGA 平台上测试了显示列表的定义、调用、查询及删除操作,其绘图结果与返回值与 ALT-M9 平台一致,基本功能实现正确。

3.2 性能测试

性能测试结果如图 5 所示。

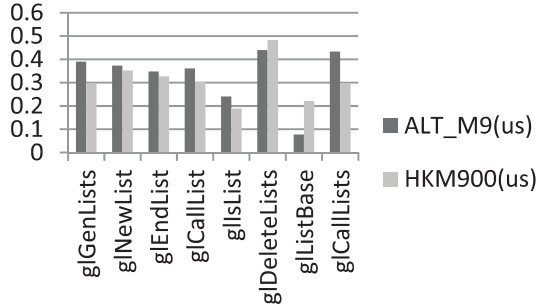


图 5 显示列表接口性能测试

测试结果为 100 万次调用的平均运行时间。从结果可以看出, HKM9000 图形处理器上显示列表接口的处理性能与 ALT-M9 平台下的性能基本相当,其中某些接口的处理性能还要优于 ALT-M9 平台^[12]。

4 结束语

文中介绍了图形处理器中的显示列表功能,在分析 ALT-M9 芯片功能性能的基础上,提出了一种软硬件结合方式来实现显示列表功能的方法^[13]。该方法充分利用了图形处理的显示列表空间,灵活地适用于多种图形处理接口实现方式^[14],具有可靠性高、编程灵活、易于实现、易于维护、成本低廉等优点。

参考文献:

[1] 谭显强. 基于 FPGA 的 3D 图形处理器 IP 核的设计与实现 [D]. 南京: 南京航空航天大学, 2010.

[2] Sefraoui O, Aissaoui M, Eleuldj M. Openstack: toward an open-source solution for cloud computing[J]. International Journal of Computer Applications, 2012, 55 (3) : 38-42.

[3] Nvidia. Dedicated GPU technology for virtual desktops [EB/OL]. [2013-11-05]. <http://www.nvidia.com/object/dedicated-gpus.html>.

慢、信息不通畅的顾虑,使企业安心提高其实际管理水平和竞争能力;为管理部门提供区域内、第一手食品安全监管的信息和资料,同时其不需要再因为食品安全问题前往企业调研,降低了人力、物力、财力的消耗。全程信息互通,这个问题是以往食品安全监管很难实现的问题,虽然文中所提云服务框架模型没有减少食品生产到销售的各个环节,但是只要其生产到销售的各个环节都在云服务的范围内,就可以形成生产到销售一条龙食品安全监管服务。其解决了用户和管理部门对食品生产到销售各环节食品安全不了解的问题,更解决了企业原料进货所担心的对原料原有保存环境不了解的问题。

4 结束语

文中旨在建立一个既能适用于物联网又能适用于融合环境的食品安全云。首先介绍了目前食品安全系统主要的研究成果,其次提出了云服务框架模型,之后研究了该模型所产生的影响,但没有对该环境下的用户模型和云平台架构进行研究,在以后的研究中将逐步完善。

参考文献:

- [1] 陈雨生,梁杰,尹世久.我国食品安全认证与追溯体系耦合监管研究[J].经济问题,2015(3):93-97.
- [2] Madoff L. ProMED-mail: an early warning system for emerging diseases[J]. Clinical Infectious Diseases, 2004, 39(2): 227-232.
- [3] 黎建辉,杨风雷,崔建业,等.全球食品安全信息监控与分析云平台架构研究[J].计算机应用研究,2014,31(8): 2361-2366.
- [4] 刘洋,张钢,韩璐.基于物联网与云计算服务的农业

温室智能化平台研究与应用[J].计算机应用研究,2013, 30(11):3331-3335.

- [5] 崔文顺,张芷怡,袁力哲,等.基于云计算的日光温室群物联网服务平台[J].计算机工程,2015,41(6):294-299.
- [6] Yachin D. 2009 M&A overview: cloud computing[R]. [s. l.]: International Data Corporation, 2010.
- [7] 马蕾,龚戈萍,刘建平.基于物联网的云数据存储访问和隐私保护机制研究[J].计算机应用与软件,2013,30(9): 319-322.
- [8] 苏伟,刘琪,张宏科.一体化标识网络体系及关键技术[J].中兴通讯技术,2011,17(2):1-4.
- [9] Vaquero L M, Rodero-Merino L, Caceres J, et al. A break in the clouds: towards a cloud definition[J]. ACM SIGCOMM Computer Communication Review, 2009, 39(1): 50-55.
- [10] Arutyunov V V. Cloud computing: its history of development, modern state, and future considerations[J]. Scientific and Technical Information Processing, 2012, 39(3): 173-178.
- [11] 王晓明.三网融合环境区域云数字图书馆构建研究[J].计算机技术与发展,2014,24(5):227-230.
- [12] 袁超伟,张金波,姚建波.三网融合的现状与发展[J].北京邮电大学学报,2010,33(6):1-8.
- [13] 张龙昌,刘冬升.三网融合下区域数字图书馆云服务框架研究[J].计算机技术与发展,2015,25(1):177-182.
- [14] 李春杰,王晓明,张龙昌.云计算环境图书档案管理系统用户模型研究[J].计算机技术与发展,2015,25(5):233-236.
- [15] 王晓明,王春阳,张龙昌,等.云图书档案系统环境下的终端体系结构研究[J].计算机技术与发展,2015,25(3):99-102.
- [16] 王晓明,张龙昌.三网融合区域云数字图书馆跨库检索服务研究[J].计算机技术与发展,2014,24(9):245-248.

(上接第 122 页)

- [4] Lindholm E, Nickolls J, Oberman S, et al. Nvidia tesla: a unified graphics and computing architecture[J]. IEEE Micro, 2008, 28(2): 39-55.
- [5] Brodtkorb A R, Hagen T R, Sætra M L. Graphics Processing Unit (GPU) programming strategies and trends in GPU computing[J]. Journal of Parallel and Distributed Computing, 2013, 73(1): 4-13.
- [6] 邱航,陈雷霆.基于点的计算机图形学研究进展[J].计算机科学,2009,36(6):10-15.
- [7] Shreiner D. OpenGL 编程指南[M].第6版.北京:机械工业出版社,2009.
- [8] Shreiner D, Woo M, Neider J, et al. OpenGL 编程指南[M].李军,徐波,译.第7版.北京:机械工业出版社, 2010:77-79.

- [9] James F, Andriesvan D, Steven K, et al. 计算机图形学导论[M].董士海,唐泽圣,李华,等,译.北京:机械工业出版社,2004.
- [10] 武丹,许如星.浅析《计算机图形学》中线裁剪算法的讲授方法[J].科技创新导报,2009(30):182-182.
- [11] 韩俊刚,蒋林,杜慧敏,等.一种图形加速器和着色器的体系结构[J].计算机辅助设计与图形学学报,2010,22(3):363-372.
- [12] 杨毅.面向移动设备的真实感图形处理系统设计与实现[D].合肥:中国科技大学,2008.
- [13] 刘鑫,蒋林.2D图形加速器设计与实现[J].微电子学与计算机,2013,30(6):75-79.
- [14] 卢俊,颜哲,田泽.一种高效GPU存储系统体系架构设计[J].计算机技术与发展,2015,25(4):6-9.