2D 图像硬件加速引擎的设计优化

袁扬智[1], 韦 明[2], 曾献君[3]
(1 国防科学技术大学 计算机学院,湖南 长沙 410073;
2 香港创辉电脑公司 深圳办事处,广东 深圳 518031; 3 福建省集成电路重点实验室,福建 福州 350002)

摘 要: 针对便携式设备中图形用户界面图像显示需求的不断提高,分析讨论了 2D 图像加速引擎的实现技术,基于 GBA(Game Boy Advance)的模拟器 VBA(Visual Boy Advance)的绘图方法提出了 2D 图像硬件加速引擎的实现架构,该架构对速度、面积及总线带宽等因素进行了优化处理,减轻了 CPU 的负荷、同时达到了低功耗设计的目的. 基于该架构实现的 2D 图像硬件加速引擎通过了正确的验证与测试,能有效地支持 GBA 等 2D 游戏,根据需要可以应用到不同的图形用户界面,提供高效的图像加速显示能力.

关键词: 图像加速;2D 引擎;GBA 游戏;低功耗
中图分类号: TP391 **文献标识码**: A **文章编号**: 1000 - 7180(2009)04 - 0241 - 04

Design and Optimization of 2D Graphics Hardware-accerleration Engine

YUAN Yang-zhi[1], WEI Ming[2], ZENG Xian-jun[3]
(1 School of Computer, National University of Defense Technology, Changsha 410073, China;
2 Agent at Shenzhen, DTK Computer Company, Shenzhen 518031, China;
3 Key laboratory of Integrate Circuit in Fujian Province, Fuzhou 350002, China)

Abstract: Considering that the requirement of graphics display of graphical user interface in portable devices increases continuously, the technology of 2D graphics acceleration engine is analyzed and discussed. A realizable framework of 2D engine base on the drawing method of VBA which is a simulator of GBA is put forward. Speed, area and bus band width are optimized, it can reduce the CPU's load and reach the purpose of low power consumption. The 2D engine implemented base on the framework has passed verification and test, can support 2D games such as GBA efficiently, and also could be used for different graphical user interface according to different requirement to provide graphics accelerator effectively.

Key words: graphics accelerator; 2D engine; GBA game; low power consumption

1 引言

目前绝大多数休闲游戏与动漫游戏都采用 2D 游戏技术开发[1],2D 游戏技术已经成为手机及一系列小型游戏设备开发的主导技术,而 2D 图像主要用在 2D 游戏中.

在 2D 游戏中,为了追求透明光影的效果,通常都会使用到 alpha 混合、马赛克、淡入淡出、缩放旋转等颜色特效,对这些颜色特效进行处理消耗大量的 CPU 资源[2-3],用软件实现绘图操作,速度慢且能量消耗大,很难达到游戏的要求.

游戏引擎是游戏开发的基础,日本任天堂公司开发的 GBA(Game Boy Advance)游戏是 2D 掌上型游戏发展的标志[4],如果 2D 图像引擎支持 GBA 游戏,GBA 等 2D 游戏便可以在手机等便携设备中开发设计了,这将使通常的图像引擎技术得到改进.

文中分析讨论了 2D 图像加速引擎实现技术,提出了 2D 图像硬件加速引擎的实现架构,基于该

收稿日期: 2008 - 06 - 10
基金项目: 国家自然科学基金项目(60676016)

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

架构实现的引擎能支持 GBA 以及 GBC 等相关系列游戏,功耗低、图像加速效率高.

2 2D 引擎绘图分析

GBA 绘图模式有三大特色:bitmap 模式、tile 模式以及 sprite 影像.

Bitmap 模式:bitmap 模式的原理类似 DOS 模式下的绘图方式. 它提供了一块存储空间(Video Ram),Video Ram 跟计算机的显示内存一样,屏幕上的一个点对应一个显示内存地址,要在屏幕上绘图就必须计算出像素对应的 Video Ram 的地址值,并填入像素的资料.

Tile 模式:在 tile 模式中,Video Ram 被分割成多个区块,这些区块被用来存储两类资料,一类是存放图块的 tile data,另一类是存放 tile data 是如何被放置在画面上的 map data. 绘图时依 map data 的内容将 tile data 显示在画面上.

Sprite 动画独立于这两种绘图模式之外,不论是 bitmap 模式,还是 tile 模式,都可以使用 sprite 动画,它的储存方式有点像 tile 模式,它需要从 video ram 中取一段存储体作为存放 sprite 资料的区段,另外还需要一块存储体存放每个 sprite 的属性的空间,这一段空间被称作空间属性存储体(object attribute memory,OAM).

GBA 的 6 种背景模式中,其中三种是 Tile 模式,另三种是 Bitmap 模式,表 1、表 2 分别给出了各个模式的显示能力.

表 1 Tile 模式显示能力表

模 式	Mode 0	Mode 1	Mode 2
可使用图层数	4 层	3 层(0,1,2)	2 层(2,3)
提供旋转图层	无	BG2	BG2,BG3

表 2 Bitmap 模式显示能力表

模 式	分辨率	最大同时显示色彩数	调色板使用方式
Mode 3	240 ×160	32 768	直接定值
Mode 4	240 ×160	256	调色板索引
Mode 5	160 ×128	32 768	直接定值

VBA(Visual Boy Advance)是 GBA 最好的模拟器,通过对 VBA 的绘图过程进行分析,得出其工作流程为:加载游戏文件(CPULoadRom) →初始化 CPU (CPUInit、CPUReset) →初始化 SDL (SDL-Init、SDL-InitSubSystem) →初始化显示模式(SDL-SetVideoMode) →进入 CPU 循环(CPULOOP) →响应按键等事件.

进入 CPULOOP 后会不停地去检查游戏程序填充的 Video Ram 和寄存器的值并解析指令,而 CPUUpdateRender 会调用不同显示模式下的相关函数进行绘图,6 种模式下最终都会调用最基本的绘制 text(普通)图层、rot(旋转)图层以及精灵和特效处理的函数进行绘图,这些函数会根据指令及寄存器的值计算所要绘制的点在 Ram 中的地址(如存放 map data 和 tile data 的地址),最后得到该点的像素值,在处理每个点的同时根据需要也会进行马赛克特效处理. GBA 支持多图层显示,计算出各图层及精灵的数据后需要进行优先级比较,同时作 alpha 混合等其它特效的处理,得到最终要显示的像素值.

VBA 绘图操作都是在进入 CPULOOP 后进行的,文中设计的 2D 图像硬件加速引擎基于 VBA 的绘图方法实现,2D 引擎的功能实质上是替代 CPU-UpdateRender 所调用的绘图函数实现绘图的全部操作,这样软件只负责解析指令配置引擎工作,绘图速度大大提高. 2D 引擎绘图过程如图 1 所示,首先进行模式选择,若强制中止画面显示被设置则直接将白色写入显示 Ram,正常显示则按选定的模式进行绘图. 各个模式下支持表 1、表 2 所示的显示能力,例如 Mode0 最多需要处理 4 个 text 图层,同时各模式下都需绘制精灵. 6 种显示模式首先会调用绘制 text 图层、rot 图层和精灵的功能模块进行绘图,然后作颜色特效处理、优先级比较,最终把要显示的数据写入显示 Ram 等待显示.

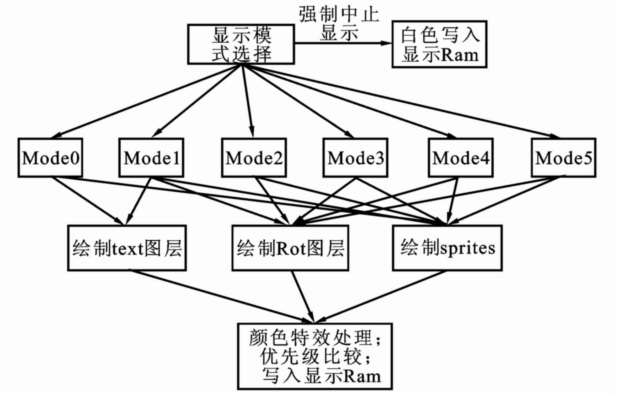

图 1 2D 引擎绘图流程

3 2D 图像硬件加速引擎架构设计

3.1 引擎总体结构设计

在基于 AMBA 总线的系统结构中,如图 3 所示,可以说明 2D 引擎是如何工作的:

(1) 启动游戏.

(2) 软件从 Nand Flash 中读取游戏数据到主

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

存,同时配置 DMA 传输游戏数据到 Video Ram、OAM 和 Palette Ram 中.

(3) 当显示数据经 DMA 传输完成后,软件配置相关寄存器,发送显示指令给 2D 引擎硬件.

(4) 硬件接收到指令后,根据模式和寄存器信息计算各图层和精灵的数据.

(5) 图层和精灵的数据处理完后,作颜色特效处理和优先级比较,并将最终的显示数据放入 LCD Ram 中,等待 LCD 显示.

(6) 当处理完一帧数据后,2D Engine 处于等待状态,软件可调整图像资料,重新配置 DMA 传输数据到 Video Ram、OAM 和 Palette Ram 中.

(7) 继续下一帧数据的处理,直到游戏结束.

GBA 的分辨率一般为 240 ×160,2D 引擎在绘完每行的第 240 个像素点之后,会进入 Horizontal Blank(水平空白区间,简称 H-Blank),同样地,在绘完第 160 行后,会进入 Vertical Blank(垂直空白区间,简称 V-Blank),如图 2 所示,空白区间内的处理在画面上是不会显示的,可以利用这些时间更改图像资料以避免画面闪烁.

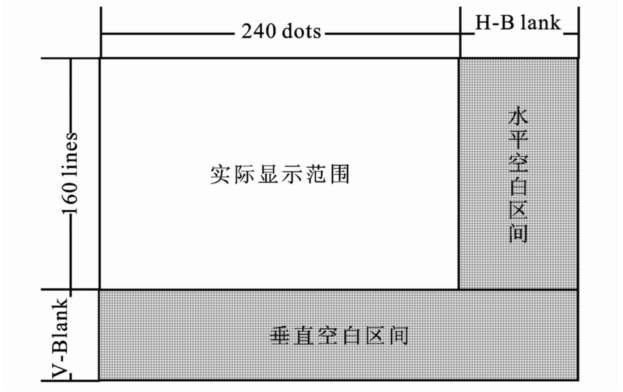

图 2　引擎绘图范围

2D Engine 在绘图过程中对每一帧图像都是按行处理,绘完每行 240 个点进入 H-Blank 后,硬件将状态寄存器中的 H-Flag 位置 1,同时 VCOUNT(记录当前所画行的寄存器)加 1,软件检测到 H-Flag 为 1 后可更新寄存器及 OAM 等,当把资料准备好后软件清零 H-flag 位并通知硬件开始处理下一行数据. 进入 V-Blank 后(VCOUNT > 160),VCOUNT 重新置为 0,硬件将状态寄存器中的 V-Flag 位置 1,软件检测到 V-Flag 为 1 后会自动清零 V-Flag,重新配置 DMA 传输数据到 Video RAM、OAM 和 Palette RAM 中,图像资料准备好后通知引擎进行下一帧数据的处理. 这样,通过软硬件间的协调工作,2D Engine 便正确地实现了图像的绘制.

从图 3 可以看出,2D Engine 所需直接处理的游戏数据存放在片上 RAM,即通过 CPU 配置 DMA 传输游戏数据到片上 RAM,而不是在内存中,因为主要考虑到游戏运行时要求图像处理速度很快,如果 2D Engine 直接到主存中读取游戏数据,再将经处理后的数据放回内存再等待显示的话必然会带来速度以及总线带宽的问题,很难达到加速的效果. 通常的手机等系统中,DMA 都是公用的,不需要为 2D Engine 设计专门的 DMA,只需要软件作相应的寄存器配置,DMA 即可完成图像数据的传输.

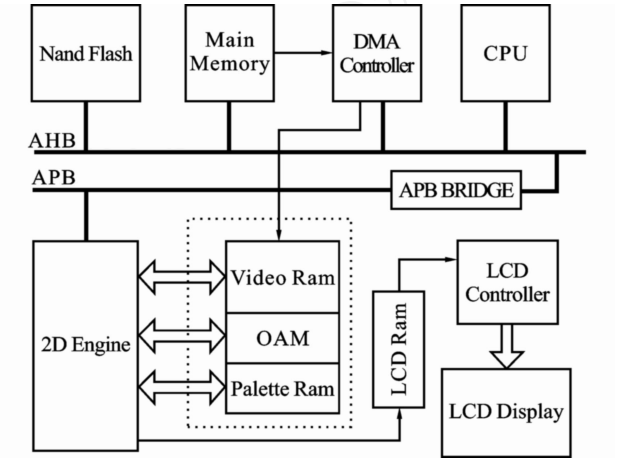

图 3　2D 引擎在基于 amba 总线的系统中的配置

一般地,进行图像处理时,显示占用很大部分时间,本架构的优越性还在于经过处理后需要显示的数据直接放在 LCD Ram 中,而 LCD Controller 无需挂总线,直接到 LCD Ram 中读取数据并加以显示. 这样,只需要缓冲 2 行游戏运行的色彩数据,采用乒乓机制,把 LCD Ram 做成双端口 Ram,处理完一行后,下一行的色彩数据存放到第二行缓存,同时 LCD 控制器读取上一行的数据显示,在 2 行缓存间反复交换. 所以图像数据处理完后也就基本显示完毕,显示过程由硬件单独完成无需软件进行处理,取得良好的图像加速和显示效果.

3.2　引擎的模块实现

为了实现方便,引擎划分为如下几个主要功能模块:

Apbslave: 提供 2D 引擎与 Apb 总线的接口,主要完成寄存器的配置.

DMAinterface: 提供 DMA 接口,以使 DMA 从内存中快速搬运游戏数据到 on chip RAM 中.

Modesel: 实现游戏显示模式选择,以及图层优先级比较和颜色特效处理功能,最后完成显示数据输出.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

Screentext: 负责 text 图层游戏数据的处理,该模块根据寄存器的值,经过一系列计算,查找到存放在 video ram 中的 map data、tile data,从而得到取调色板颜色的索引值.

Screenrot: 负责 rot 图层游戏数据的处理,该模块根据寄存器值,经过一系列计算,查找到存放在 video ram 中的 map data、tile data,从而得到取调色板的索引值.

Sprites: 实现精灵和精灵 window 的绘制. 与 text 图层和 rot 图层不同,精灵需要对 OAM 进行操作. 该模块输出精灵调色板的索引值.

Window:实现 window 特效的处理.

LCD Controller:接收优先级比较后最终要显示的像素数据,并根据 LCD 扫描屏幕的时序输出对应的 R,G,B 值.

具体的 2D Engine 实现结构如图 4 所示.

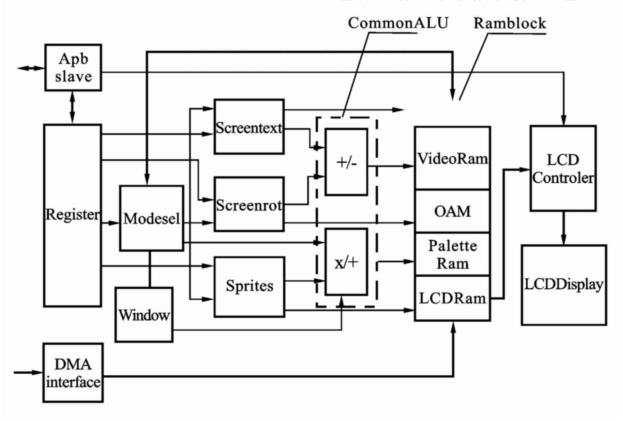

图 4 2D Engine 具体实现结构图

在绘图的过程中,Screentext、Screenrot 和 Sprites 三个功能模块分别完成 text 图层、rot 图层和精灵的绘制,由于在不同的显示模式下,都可能使用精灵动画,为了更快地加速图像处理,硬件设计中 Sprites 模块和其他两个模块并行进行,这样保证了绘制时间的要求. 而绘制精灵的模块 Sprites 和 Modesel 模块以及 window 模块可以共享运算单元如 OBJmult16 ×8,OBJadd8 等.

由图 1 可知,只有在模式 1 的情况下需同时处理 text 和 rot 图层,而 text 图层只有 2 层,rot 图层 1 层,可先进行 text 图层的处理,再作 rot 图层的处理,其他模式下要么只有 text 图层,要么只有 rot 图层,这样在硬件实现的时候,两个模块不并行工作,可以最大限度地共享硬件资源. Screentext 模块和 Screenrot 模块共享两个加法器(分别为 28 位和 8 位的加法器),寄存器,以及 ram 等.

4 引擎测试与综合

基于 VBA 本身的完全正确性,引擎测试只需将硬件加速后的结果同 VBA 本身运行 GBA 游戏的结果进行对比,看结果是否一致,若完全一致表明 2D Engine 正确地实现了绘图加速功能.

在 VC 6.0 中运行 VBA 模拟器,对 VBA 的 C 代码作相应修改,即对用硬件实现加速的那些 C 代码加入抓取游戏数据的函数,把硬件所需的激励和预期的结果的游戏数据存放到文本文件中,硬件读取存放在文本中的测试激励后运行得到的结果与存放在文本中的结果自动进行对比.

在时钟频率 100MHz,编译器 Sparc-elf-gcc,LCD 屏分辨率 240 ×160,帧速率为 60 帧/s 下,对引擎进行测试,引擎绘制一行像素点需要 120μs,相较于 VBA 模拟器绘制一行需要时间为 1.6ms,速度整整提高了 10 倍以上,图像加速效率高.

使用 TSMC 90nm 的工艺库,在 100MHz 的频率下对 GBA 各个模块用 DC 进行综合,结果如表 3 所示. (其中面积大小和功耗不包括 RAM 的面积和读写 RAM 的功耗).

表 3 DC 综合结果

总面积(&micro)	总动态功耗/mW	最大余量/ns
42 266.571 862	3.532 5	5.63

5 结束语

文中描述了一种 2D 图像硬件加速引擎的设计和实现方法,并具体实现了引擎. 本设计中的 2D Engine 能很好地支持 GBA 等游戏,使得能够在手机等手持设备中开发设计上千种备受欢迎的 GBA 游戏. 可以将 GBA 游戏的多图层显示和精灵动画等效果应用到其他游戏或图形用户界面,从而达到良好的图像加速显示效果.

参考文献:

[1] Mc Cormack J, Mc Namara R, Gianos C, et al. Implementing Neon: a 256 bit graphics accelerator [J]. MicroIEEE, 1999(19):58 - 69.

[2] 李淀,乔永强,贺骊,等. 图像改造算法的实现[J]. 微电子学与计算机,2006,23(4):45 - 47.

[3] 章立,徐立鸿,姜磊,等. 嵌入式数字视频录像机 GUI 系统的设计与实现[J]. 微电子学与计算机,2006,23(2):58 - 61.

(下转第 248 页)

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

针对 GIF 图像前后像素点重复出现概率比较高的特点,进行硬件结构的优化,对每次当前输入的压缩数据都与前一个压缩数据进行比较,如果相同,则可以直接将缓存在 RAM2 中的解码数据不做任何改变的再输出一次.另外,在当前压缩数据分解一次后,看其前缀是否与前一个输入数据相同,如果相同,则只需要多输出这次的后缀(一个字符)到缓存 RAM2 中,也就是只需要从 RAM1 中读取一个数据到 RAM2 中,最后输出这个数据,而 RAM2 中的其他数据不做改变输出.这种实现方式,硬件的控制电路只增加一个比较器,却能减少从 RAM1 搬运同样的数据到 RAM2 的次数.通过减少读写 RAM 的次数,从而减少能耗.

5 仿真和综合结果

通过 modelsim 仿真提取出解码后的数据.在 MATLAB 中,对原图与硬件解码出的图像数据进行显示和对比,如图 3 所示.可以看出图像完全无失真的恢复.对各种大小规格不同的图像解码出的数据,在验证平台上比较其对应的原始图像数据也 100 % 吻合.说明能够完全不失真的恢复图像数据.

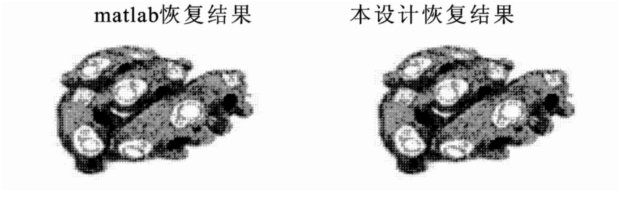

图 3 MATLAB 进行图象恢复与本设计恢复结果对比图

使用台积电 90nm 的工艺库,在 100MHz 的频率下对 GIF 解码核心模块用 DC 进行初步综合,结果如表 1 所示.

表 1 DC 综合结果报告

Total cell area	4 441.751 953(&micro)
Total dynamic power	204.541 0/μW
Max slack	6.44/ ns

6 结束语

文中有别于以往用软件实现 GIF 解码,在用硬件实现 GIF 解码的同时,利用 GIF 图像"突变性"的特点,改进硬件结构,通过减少读写 RAM 的次数来降低硬件解码能耗.通过 EDA 工具验证和 MATLAB 恢复对比,完全满足 GIF 图像硬件解码的要求.

参考文献:

[1] 吴宇新,余松煜. 对 LZW 算法的改进及其在图像无损压缩中的应用[J]. 上海交通大学学报, 1998,32(9):110 - 113.

[2] 崔业勤,刘玉贵. 基于 LZW 的多模式自适应的无损压缩算法[J]. 微电子与计算机, 2007,26(3):99 - 101.

[3] Israel Koren. Computer arithmetic algorithms[M]. New York: Prentice Hall, 1993.

[4] Michael D Ciletti. Advanced digital design with the verilog HDL[M]. New York: Prentice Hall, 2003.

[5] Mark D Birnbaum. Essential electronic design automation (EDA)[M]. New York: Prentice Hall PTR, 2003.

[6] 金卫民. 数据通讯中 LZW 算法的应用研究[J]. 计算机工程与科学, 2004, 26(5):46 - 48.

作者简介:

廖 腾 男,(1985 -),硕士.研究方向为集成电路设计.

(上接第 244 页)

[4] Duardo O, Da Graca P, Hosotani S, et al. A cost effective HDTV decoder IC with integrated system controller, down converter, graphics engine anddisplay processor[J]. IEEE Transactions on Consumer Electronics, 1999(45):879 - 883.

作者简介:

袁扬智 男,(1985 -),硕士研究生.研究方向为微电子与固体电子学.

韦 明 男,(1979 -),硕士研究生.研究方向为微电子与固体电子学.

曾献君 男,(1966 -),教授,博士生导师.研究方向为微电子与固体电子学.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net