基于 SoC 平台 2D 游戏动画技术①

胡溢文[1]，赵 珂[1]，张先庭[1]，熊小亮[2]

[1](南昌航空大学 信息工程学院，南昌 330063)
[2](衡阳电业局，耒阳 421001)

摘 要：根据 SoC(片上系统)平台的特点，分析嵌入式系统用于 2D 游戏动画实现存在的问题。采用特殊数据结构对画面进行分块，结合硬件特性并利用 μ COS-II 操作系统来解决资源利用、更新和渲染的问题，从而在系统整体性能不受影响的条件下使游戏画面流畅。

关键词：游戏动画；更新；渲染；2D；SoC

2D Game Animation Technology Based on SoC

HU Yi-Wen[1], ZHAO Ke[1], ZHANG Xian-Ting[1], XIONG Xiao-Liang[2]

[1](School of Information Engineering, Nanchang Hangkong University, Nanchang 330063, China)
[2](Hengyang Electric Power Bureau, Leiyang 421001, China)

Abstract: According to the characteristic of SoC(System on Chip) platform, this paper analyzes the present problem of realization of 2D game animation for embeded system. Using special data structure to divide picture into blocks, combining hardware characteristic and using μCOS-II operating system to solve problem of resource occupation, updating and rendering, then make the game picture fluent in terms of no affects on system overall performance.

Key words: game animation; updating; rendering; 2D; SoC

个人电脑(PC)以其通用的平台和完善的硬件资源和软件支持的巨大优势，能够很好的实现二维、三维游戏。近些年，嵌入式系统发展迅速，硬件资源越来越丰富，在手持设备方面有大量的应用。在 21 世纪初期任天堂发布 GBA(Gameboy Advance)，它采用的就是 32 位 ARM 的处理器，以此来取代传统的 Gameboy。可见在嵌入式系统上开发游戏产品已经相当的成熟。但较之与 PC 相比，嵌入式系统的资源相对有限。首先是处理器的运行速度限制；其次是存储容量的限制，游戏中为显示丰富的画面，将会占用大量的空间；最后，游戏要以一定的帧速率来显示游戏画面，保证动画的流畅，系统的总线带宽必须考虑。因此，2D 游戏在嵌入式平台上的应用受到诸多限制，本文研究的重点是采用特殊的数据结构解决存储容量并结合硬件特性来处理显示和更新的问题。

1 系统硬件

系统采用 GeneralPlus 公司推出的 GPL32600A 处理器，该处理器以 ARM7DMI 为核心，并有图形处理单元(PPU)和声音处理单元(SPU)[1]。PPU 中有电视编码器，可以输出各种制式的电视用的复合同步信号。整个游戏系统上电时，从 SD 卡等外部存储设备中获取游戏的图形、音频等资源，由处理器为其分配存储空间，数据分别经由 SPU 和 PPU 处理后，播放音频和动画。键盘控制画面中游戏角色运动和参数的设置，实现人机交互。系统硬件结构如图 1 所示。

PPU 支持帧模式，要求系统配置高速内存，如 Page RAM 或 SDRAM。当基于帧模式使用时，PPU 能够异步到 TV 或 TFT 输出，但需要额外的带宽将数据写入到系统内存和从系统内存中读取数据。在帧缓冲模式下所需要的带宽如表 1 所示。

① 收稿时间:2011-08-20;收到修改稿时间:2011-09-26

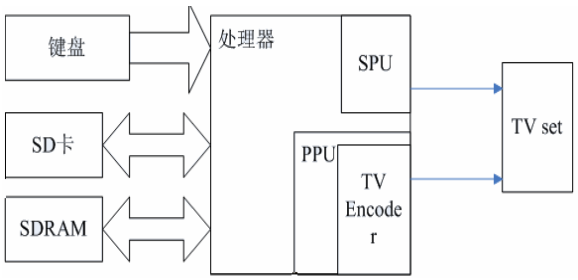

图 1　硬件结构

表 1　在帧缓冲模式下带宽要求

模式	QVGA	VGA
帧缓冲输出(15 帧/秒)	2.304MB/秒	9.216MB/秒
帧缓冲输入(60 帧/秒@QVGA)(30 帧/秒@VGA)	9.216MB/秒	18.432MB/秒
共　计	11.52MB/秒	27.468MB/秒
百分比在基于 ROM 方式下(70ns RAM,9 周期的访问时间)	54%	130%
百分比在基于 SDRAM 方式下(96MHz,使用 70%)	8,57%	20.57%

从上表可以看出，基于 ROM/RAM 的系统在 VGA 模式下，带宽的占用率达 130%，故无法在基于 ROM/RAM 的系统上使用帧模式。基于行模式也存在着技术缺陷，即关键行不能与非关键行共享带宽，同时帧率必须与 TV 或是 TFT 同步，这在 2.5D 或是旋转操作中是相当苛刻的。若在基于帧模式下采用 SDRAM 作为帧缓冲区，其带宽占用率在 VGA 模式下仅为 20.57%，可以有效解决带宽占用问题。故本系统采用基于帧模式并选择 SDRAM 作为帧缓冲区。

2　游戏图形元素的使用

游戏画面的丰富离不开图形元素，合理使用图形元素可以减少存储容量同时又不影响画质。通常采用位图技术对游戏中的精灵、地图和背景进行编码，而位图是以像素进行处理的，将占用大量的内存空间。为解决内存占用问题，系统利用均匀平铺[2]的数据结构来减少数据容量。

将图形进行划分成一个 n*n 像素的小块，充分利用图形中的相同块，这样图形所占的存储容量就会随着块的减少而减少。游戏画面分成许多块被称为均匀平铺，在场景显示常用的做法就是采用游戏单元填充的方式来实现场景地图，采用这种方法仅需准备游戏场景中特定的游戏单元背景资源，就可以实现丰富庞大的游戏背景地图[3]。

PPU 支持 4 层结构，可以实现 4 层精灵和 4 层的背景，每个层都可以设置不同的尺寸。本课题采用 VGA 输出模式，分辨率为 640*480[4]。在每一层上，图形被分割成 32*32 的像素块，像素块的标号构成整个游戏画面，根据标号可以找到对应像素块。PPU 中带有调色板 RAM，分为 16 位和 25 位两种。16 位色对游戏动画已经够用，而且 25 位方式需要 32 位来存储，占用更多的内存，故选择 16 位调色板模式，每个颜色索引为 8 位，这样构建的调色板大小有 256 种颜色，占用 512 字节。

系统为了得到均匀平铺方式的数据结构，解决内存占用问题，采用 GeneralPlus 公司的 G+ Director V1.27[5]软件对图片进行编辑和制作。图片文件包括调色板文件、像素块索引文件、颜色索引文件和记录文件。颜色索引文件保存着该像素块数据的颜色索引值，通过该索引值查找调色板文件索引号，就可得到该像素处的颜色值。像素块索引文件是由像素块的索引组成，所有像素块的索引构成整个游戏画面。记录文件记录颜色位数、调色板信息、像素块大小、地图大小等基本信息。其实现的原理如图 2，可视区域为屏幕显示区。

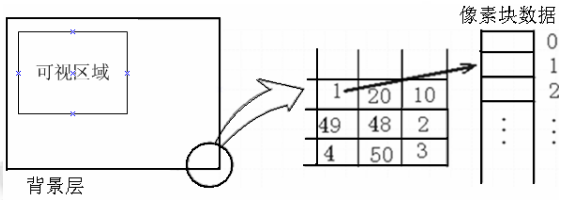

图 2　基于像素块的数据结构

采用均匀平铺的方式，像素块的起始地址为

Real Address = Base Address + (Character Number * Character Size * 2)　　(1)

注：Base Address 为起始像素块的地址，Character Number 为像素块的索引，Charater Size 为像素块的大小，这里为 32*32。

3　渲染与更新

视频游戏是实时的应用软件，受时间的严格限制，必须通过一定的速率显示信息（通常是高于 25 帧/秒）来实现无缝的交互。根据游戏显示信息，与用户进行交互以及响应事件，可以将游戏划分为渲染和更新两

个部分，这两个部分对游戏实时性至关重要。处理处理渲染和更新时常采用：1) 在一个循环中执行，渲染在更新之后，这种方法会因复杂等级的变化而影响帧率；2)使用双线程，会因操作系统定时功能不是很精确而造成频率的变化；3)渲染被频繁的调用，而更新与时间同步并维持一定的速率[6]，其帧率的不稳定会在长时间的用户交互中表现出来。

3.1 PPU 中断描述及更新渲染实现

考虑到传统方法实现存在的问题，本系统结合硬件特性利用 PPU 视频信号的消隐中断来控制更新和渲染，保证帧率的稳定。PPU 有垂直消隐中断、TV 垂直消隐中断、TFT 垂直消隐中断和 DMA(直接内存访问)完成中断。PPU 中有一个帧缓冲输出指针的寄存器 P_PPU_FBO_ADDR 和一个帧缓冲输入指针的寄存器 P_TV_FBI_ADDR 和 P_TFT_FBI_ADDR。FBO_ADDR 指示计算输出的帧缓冲地址，只有在 PPU 的一帧结束时该地址才有效。FBI_ADDR 指示显示用的帧缓冲地址，只有在 TV 或是 TFT 一帧完成时才有效。为了实现更好的效果，在程序中开辟两个内存空间作为帧缓冲区，即双缓冲的方式来进行流控制。一个用于更新时，另一个就作为显示用的缓冲区。

根据 PPU 的这些特点，首先完成 PPU 寄存器的配置，然后开启 PPU 垂直消隐中断。当 PPU 完成一帧的计算后，PPU 将产生垂直消隐中断，接着开启 TV(或 TFT)垂直消隐中断，中断产生时，就将一帧数据输出。由于 PPU 的寄存器众多，这里用到 DMA 来实现寄存器的设置。寄存器要设置的值保存在结构变量中，之后开启 DMA 中断，一次性的进行设置 PPU 寄存器。

3.2 操作系统实现信号通信

游戏的主要逻辑是作为系统中的一个任务存在的，在 μC/OS-II 可以使用的优先级为 56 个[7]。游戏中要处理多种文件(音频、图片等)的操作，对每种文件的操作分别建立任务，设置不同的优先级[8-10]。本系统的主要逻辑包括更新和渲染，其优先级比前面的优先级高。在主要逻辑开始时，开辟两个空间用于帧缓冲区，建立 sem_ppu_engine、sem_update_register_done、sem_ppu_frame_output_done 信号量。并创建 free_frame_buffer_queue 和 display_frame_buffer_queue 的消息队列用来传递帧缓冲区地址指针的消息，并将游戏资源载入。接着执行更新渲染，更新渲染的执行过程的部分代码如下：

```
//请求可用的帧缓冲区
frame = (INT32U) OSQPend(free_frame_buffer_queue, 0, &err);
ppu_current_frame = frame;
R_FBO_ADDR = frame;//PPU 计算结果的输出地址
OSSemPend(sem_ppu_engine, 0, &err);//无限等待直到 PPU 可用
OSSemPend(sem_update_register_done, 0, &err);//等待 PPU 寄存器的更新，更新消息队列来自于 DMA 传送完成。
OSSemPend(sem_ppu_frame_output_done, 0, &err);//等待输出的消息队列，由 TV 或 TFT 中断时发送。
OSSemPost(sem_ppu_engine);//一帧完成输出后使 PPU 可用。
```

4 结语

本系统采用金积嘉 8 英寸彩色液晶迷你电视(JV-805D)作为显示终端。连接线采用的是秋叶原的影音线，保证传输过程中不会出现明显的延时和干扰，从而不会给画质造成影响。系统输出整个显示的效果如图 3 所示：

图 3 显示的效果

图中是采用的 VGA 分辨率时的效果，显示效果很好，没有出现画面的抖动。色彩明亮丰富。游戏中的角色进行了旋转、缩放。

利用 SoC 平台实现 2D 游戏系统将具有很大的潜力，甚至在 3D 游戏方面也将会有巨大的影响。本文结合当前的 SoC 技术进行 2D 游戏开发，同时对动画实现问题进行较深入的研究和实践，结果表明，这种

(下转第 212 页)

达性，对于 e 可达就说明在函数 M 中存在一系列这样的赋值语句：v=new C(),x1=v,x2=x1,……，$x_n = x_{n-1}$，e= x_n。显然，在 main 中，类 B 的实例化信息对 e 而言是可达的，而 A 和 C 是不可达的，所以据此可知，e 的运行时的可能类型就只包括 B，那么只有类 B 中的 m 函数是可达的。从以上分析，我们可以看到改进的 XTA 方法能够取得最好的效果。对 b3.m()这个函数调用点来说，因为没有类型传播，所以结果跟 XTA 是相同的（如图 5）。

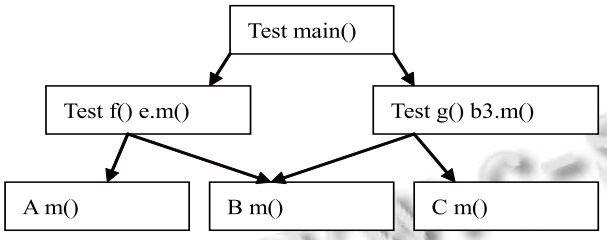

图 4　XTA 分析得到的函数调用图

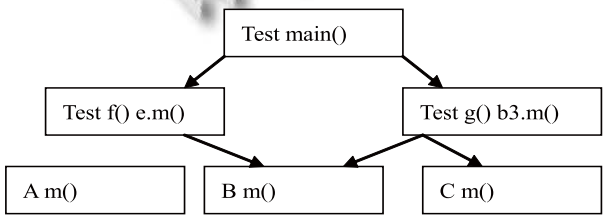

图 5　改进算法得到的函数调用图

4　结语

在类层次分析、快速类型分析和 XTA 三种的构建函数调用图的算法中，类层次分析是基础。这三种方法比较起来，XTA 方法是对快速类型分析的一种有效改进，能够更好的满足实用性和程序扩展性的要求，也获得了很好的分析精度。但本文所提到的对 XTA 算法的改进，在 XTA 的基础上，能够更好的使函数调用图得到归类，提高了绘制函数调用图的效率。

参考文献

1 Ryder BG. Constructing the call graph of a program. IEEE Trans. on Software Engineering, 1979,5(3):216– 226.

2 Gagnon EM, Hendren LJ, Marceau G. Efficient inference of static types for Java bytecode. Static Analysis Symposium 2000. Santa Barbara, 2000: 199–219.

3 Diwan A, Eliot J, Moss B. Simple and effective analysis of statically-typed object-oriented programs. Proc. of the Conference on Object-Oriented Programming Systems, Languages, and Applications. New York, l 996: 292–305.

(上接第 215 页)

做法切实可行。

参考文献

1 GeneralPlus. GPL32600A Programing Guide. Taiwan: GeneralPlus, 2009.

2 LaMothe A.Windows 游戏编程大师技巧.沙鹰,译.第 2 版.北京:中国电力出版社,2004.

3 李志敏.基于嵌入式平台的 2D 游戏引擎的研究与实现[硕士学位论文].武汉:武汉理工大学,2006.

4 孙乾恒.基于凌阳μ'nSPTM 系列单片机的电视游戏设计与实现.西安:西安电子科技大学,2009.

5 GeneralPlus. G+ Director User Guide. Taiwan: GeneralPlus, 2009.

6 Dalmau DSC. Core Techniques and Algorithms. New Riders Publishing, 2003.

7 Labrosse JJ.嵌入式实时操作系统μC/OS-II.邵贝贝等,译.第 2 版.北京:北京航空航天大学出版社,2005.

8 李玉刚等.嵌入式操作系统 uC/OS-II 在 ARM 上的移植研究.微计算机信息,2010,8(2):97–98.

9 李晓玫,杨小平,等.基于μc/os-Ⅱ的 I/O 系统的设计与实现.微计算机信息,2010,12(2):77–79.

10 王田苗.魏洪兴.嵌入式系统设计与实例开发.第 2 版,北京:清华大学出版社,2008.