Journal of Computer Applications
计算机应用,2013,33(5):1456 - 1458
ISSN 1001-9081
CODEN JYIIDU
2013-05-01
http://www.joca.cn

文章编号:1001 - 9081(2013)05 - 1456 - 03
doi:10.3724/SP.J.1087.2013.01456

嵌入式浏览器软硬件混合渲染层的研究与设计

唐成戬*,雷　航,郭文生
(电子科技大学 计算机科学与工程学院,成都 611731)
(*通信作者电子邮箱 tangchengjian@tom.com)

摘　要:广泛使用的 WebKit 内核嵌入式浏览器因具有良好的设计架构和优良的跨平台特性已被移植到许多嵌入式平台上。由于嵌入式平台的硬件多样性,WebKit 开源版本并没有充分利用嵌入式平台的特点。通过研究 WebKit 显示系统,实现了充分利用嵌入式硬件加速渲染与软件渲染特点的软硬件混合渲染层。该混合渲染层解决了嵌入式平台上全功能浏览器运行缓慢造成用户体验较差的问题。经对比测试验证了该混合渲染层的可行性,与原始版本相比网站打开时间减少 48% 以上,网页动画渲染速度提高 130% 以上。

关键词:WebKit;嵌入式系统;DirectFB;软硬件混合渲染层
中图分类号:TP393.092　　**文献标志码**:A

Research and design of hadware and software fusion render layer for embedded browser

TANG Chengjian*, LEI Hang, GUO Wensheng
(*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu Sichuan* 611731, *China*)

Abstract: Widely used WebKit of excellent architecture has been ported to many embedded platforms, with excellent cross-platform features. Due to the diversity of hardware for embedded platforms, WebKit open source version does not take full advantage of the characteristics of embedded platforms. Through studying WebKit render architecture, taking full advantage of the embedded hardware feature and the benefit of the software rendering design, a hardware and software fusion render layer was designed. This layer sped up the browser rendering on the embedded platform and improved the user experience. The layer was verified, the time of opening website was reduced by 48% and the speed of rendering of html animation increased by 130% compared to the original WebKit.

Key words: WebKit; embedded system; DirectFB; software and hardware fusion render layer

0　引言

目前基于 Web 的应用程序越来越丰富,很多应用已经可以替代本地应用程序的功能。同时随着 HTML 5 的推出,各种 Web 技术的发展,PC 的本地应用将逐渐过渡到基于浏览器的 Web 应用。因此,作为 Web 应用运行的容器,浏览器将不仅仅是一个应用软件,而将成为各种 Web 应用程序运行的平台[1]。

WebKit[2] 是目前使用最广泛的浏览器内核,其拥有清晰的源码结构和极快的渲染速度,且高效稳定,兼容性好,有着良好的设计架构,已经被移植到各种嵌入式系统。但是由于嵌入式系统专用性的特点,嵌入式硬件加速体系没有统一规范[3],各制造商在设计底层硬件时会加入新的特性,WebKit 显示系统并没有利用这些特性,其渲染只采用了软件图形库方式实现。

嵌入式平台硬件加速渲染提供的功能有限,一般仅限于块搬移、矩形填充、颜色扩展、画线、透明混合和矩形裁剪等图形操作。软件图形库通常具有更好的兼容性和灵活性,其功能也更为强大。虽然嵌入式硬件渲染速度是软件渲染的几倍甚至更高,但其功能不能满足现代图形系统的要求。因此可以设计一个软硬件混合的图形渲染层,对 WebKit 渲染请求进行分发,深度挖掘硬件加速性能。该层还提供对具体渲染实现提供抽象层,降低 WebKit 与具体嵌入式平台硬件渲染耦合度。

本文在数字电视主板 MS28L 平台上实现了该混合渲染层。MS28L 提供了 DirectFB 硬件加速接口,对块搬移、矩形填、画线和透明混合的提供加速渲染。实验结果表明该渲染层能正确完成网页显示,并且渲染速度得到了极大的提高。

1　WebKit 嵌入式浏览器显示系统

WebKit 实现网页显示在具体平台上有具体实现,广泛研究的有 Qt 移植版本[4]。本文基于 SDL 平台[5] 进行研究。WebKit 通过 SDL 初始化的 Primary Surface 实现绘图,Primary Surface 是 Framebuffer 的映射。FrameBufer 是 Linux 下的通用的图形接口,它拥有良好的平台无关性,可以支持绝大多数的硬件[6],其内容将被显示在屏幕上。绘图操作通过 Skia 软件库完成。如果网页上有透明层、图片,则需要创建新的 Surface,然后在上面绘制透明层内容或者图片。完成后将这些 surface 混合到 Primary Surface。

图 1 展示了 SDL 平台的 WebKit 嵌入式浏览器显示系统。该系统分为三层,WebKit 渲染接口层、Skia 图形库层和平台 SDL 层。WebKit 渲染接口层主要是 GraphicsContext 对象,WebKit 内部的绘画请求都是通过该对象发出。Skia 图形库[7] 是对 GraphicsContext 的绘画请求的具体实现,实现了各种图元、透明混合和图片的绘制。绘图的目标是各种 Surface,从图中虚线可以看出图片 Surface 可以混合到具有透明的属性的 Surface 上,也可以混合到 Primary Surface 上面,这取决于具体网页的组织。这里的 Surface 不是指具体软件库

收稿日期:2012-12-04;**修回日期**:2012-12-28。　**基金项目**:国家核高基重大专项(2012ZX01033-001-001)。
作者简介:唐成戬(1985 -),男,四川南充人,硕士研究生,主要研究方向:嵌入式系统;　雷航(1960 -),男,四川自贡人,教授,博士生导师,主要研究方向:嵌入式实时系统、软件可靠性测试和评价;　郭文生(1976 -),男,辽宁铁岭人,副教授,博士,主要研究方向:实时网络、实时系统。

的数据结构,只是表示存放像素信息的缓冲区抽象数据结构。在 Skia 中,surface 对应 Skbitmap 极其附属的 SkDevice。SDL 平台负责创建 Primary Surface。SDL 库还负责收集事件和分发事件给 WebKit。

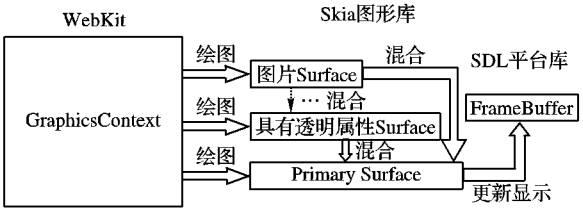

图 1　WebKit 显示系统

如上所述,现有的 WebKit 显示系统图形绘制和透明混合完全是由 Skia 软件库实现的,由于嵌入式系统 CPU 主频较低,该架构在网页显示时极为缓慢,图片显示需要较长等待时间。在网页显示过程中,移动鼠标或者按键,浏览器响应很不及时。

2　基于 DirectFB 加速的混合渲染层设计

2.1　硬件加速渲染的基本原理

硬件加速渲染是指 CPU 要通过某种方式告诉显示硬件需要做什么操作,具体在显存上将像素“填”出来由硬件实现,对于内存块搬移则启动 DMA 操作。由于不同的显示芯片具有不同的加速能力,具体的使用和定义也各自不同,就需要针对加速芯片的不同类型来编写程序实现不同的加速功能[8],因此引入软件抽象层,可以剥离应用程序与嵌入式底层平台的高耦合。

2.2　DirectFB 硬件加速

DirectFB 项目[9]是由德国 Convergence 公司面向嵌入式设备开发的小巧、强大、灵活和易于使用的图形系统,提供一些基本的图形加速、输入设备处理提取、透明窗口和多重显示层的功能,以最低限度的资源使用和开销,提供最大的硬件加速度性能[10]。DirectFB 使用了可载入的驱动模块,这些驱动模块由嵌入式芯片厂商提供,可以通过 API 函数获取到哪些渲染操作具有硬件加速功能。本文使用的 ms28L 具有硬件加速渲染的模块有块搬移、矩形填充、画线和透明混合。

2.3　软硬件混合渲染层设计

图 2 是加入软硬件混合渲染层的 WebKit 显示系统。DirectFB 也提供了屏幕显示的管理,因此加入混合渲染层之后直接使用 DirectFB 的屏幕更新控制,这样可以更有效地利用 DirectFB 提供的加速功能。

软硬件混合混合层由三层组成,分别为 API、模块(MODULES)和后端(BACKENDS)。模块层包块设备能力探测模块(Device Capability Detection)、Surface 管理模块(Surface Management)、绘图操作分发模块(Draw Funciton Dispatch)、绘图参数转换模块(Draw Parameter Converter)、剪切域支持模块(Clip Support)和坐标转换模块(Coordinate Transformation)。后端层目前由 Skia 后端和 DirectFB 后端组成。

API 层提供了抽象的绘图操作 API 集合。对于 WebKit 显示系统,只需要用这些 API 发出绘画请求即可,不必关心具体硬件加速接口细节。绘图操作分发模块根据运行时的设备能力探测模块进行抽象绘图操作的分发至不同后端实现。后端实现需要操作相同的图形缓冲区,因此需要统一的 Surface 管理。分发至后端实现时,还需要将 WebKit 的抽象绘图参数转换成具体后端实现参数才能正确绘图。剪切域支持模块主要是对请求的绘图区域进行裁剪。剪切域是指绘图时会对绘图区域进行裁剪的区域。文献[11]对剪切域算法进行了详细的描述。坐标转换模块完成网页坐标到设备坐标的转换。后端实现模块与具体的嵌入式硬件平台紧密相关。对于软件后端实现采用现有的 Skia 库,对应于硬件加速渲染后端实现则根据不同的驱动有不同实现。本文研究的 MS26L 硬件平台提供了 DirectFB 驱动接口,因此该实现就是 DirectFB 接口的封装。

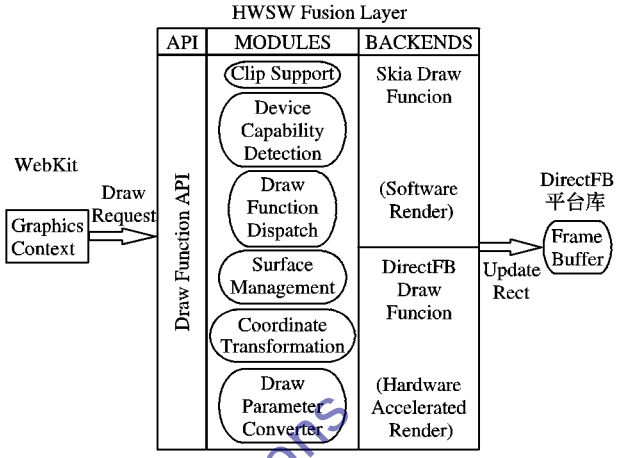

图 2　混合渲染系统

3　关键模块设计

在上述的六个模块中,与硬件加速有关的模块为设备能力探测模块、Surface 管理模块、绘图操作分发模块和参数转换模块。另外,DfbDevice 通过封装 DirectFB 加速函数实现 Device 抽象绘图操作。

3.1　设备能力探测模块

运行时读取环境变量,获取当前启动的加速驱动,加载对应后端实现。用 Device 类抽象设备绘图接口,该类提供了通用图形库操作接口的集合如路径、图片、几何图形、字体渲染等操作。

每一个具体后端实现都是 Device 的子类,DirectFB 的实现为 DfbDevice,Skia 后端实现为 SkDevice。

通过工厂模式完成具体设备实现加载,同时系统中存在一个全局的软件 Skia 后端实现。其探测流程如下:

1)初始化 Skia 软件后端实现;

2)读取驱动名称环境变量,通过工厂模式加载对应硬件加速后端实现;

3)执行硬件加速后端硬件能力探测,初始化硬件加速操作掩码,该掩码作为绘图分发时的依据。

3.2　Surface 管理模块

软件后端与硬件后端混合绘制的时候需要同时操作同一片像素缓冲区(Surface)。而硬件加速对该缓存区要求必须是物理地址,因此采用硬件驱动分配 Surface 方式,然后将该 Surface 地址传送给软件后端。对应 DirectFB 后端实现,其 Surface 管理原理如图 3。

创建 Surface 的顺序如下:

1)DirectFB 后端请求 DirectFB 库创建一个在显存内的 Surface;

2)、3)DirectFB 从显存中创建 Surface 并返回给 DirectFB 后端;

4)Skia 后端将自己的 Buffer 指针指向上一步返回的 Surface 缓冲区。

由于显存大小总是有限的,DirectFB 从显存中分配 Surface 有可能失败,这时 Surface 管理模块就需要 Skia 从内存中分配 Buffer。虽然不能再使用硬件加速,但是仍然能完成上层的渲染请求。

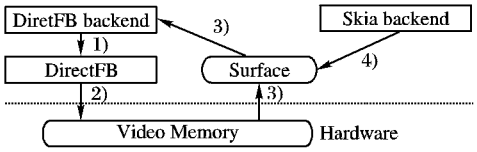

图 3 Surface 管理

3.3 绘图操作分发模块

在设备能力探测模块中已经初始化硬件后端实现和能被加速的函数后,绘图操作按通过 Device 统一的抽象接口进行分发。检查抽象设备层中是否已经有硬件后端实现(hwDevice 不为 NULL)且对应的 HWFUNCTIONMASK 对应操作是否为 1,如果成功则分发至硬件后端,如不成功则直接分发至软件后端。

3.4 参数转换模块

参数转换模块完成 WebKit 绘图参数到 Skia 后端与 Direct 后端参数的转换。DirectFB 提供了透明混合的加速,因此需要做参数转换。透明混合模式的定义由 Thomas Porter 和 Tom Duff 提出[12]。WebKit 混合模式设置为 CompositePlusDarker 时,DirectFB 无法支持。因此该模式使用软件后端实现。

3.5 DirectFB 后端实现

Device 类提供的抽象接口函数与图形相关但并不具体实现,其子类 DfbDevice 通过封装 DirectFB 驱动程序来实现对应接口功能。DfbDevice 屏蔽了底层驱动的细节,实现底层驱动相关的功能。具体的硬件加速功能由底层的 DirectFB 来完成。MS28L 的 DirectFB 库提供的 API 指明一部分 API 是硬件加速的,同时还要求操作的 Surface 必须在显存中。这些条件的判断和驱动程序状态设置、参数设置等操作需要由 DfbDevice 来完成。

4 实验结果

将软硬件混合渲染浏览器与原版进行速度对比测试,运行在 MS28L 开发板之上。该系统 CPU 为 MIPS 300 MHz,RAM 为 128 MB,操作系统为 Linux2.6.32。

4.1 自制网页性能评估

MS28L 硬件加速功能主要有图片渲染和透明混合。针对该特点,设计一个含有一个透明层作为背景图片的遮罩,同时具有多个菜单栏,每个菜单栏有一行图标的本地网页。当前焦点所在图标会有上下跳动的动画,并且图标下部加亮显示来提示用户当前焦点位置。用户按上、下键切换相邻菜单,按左、右键切换相邻图标,切换过程有过渡动画。测试方式为观察一定时间内动画的帧数(frame/s),帧数越多越平滑,用户体验越好。表 1 为该自制网页动画速度测试数据。

表 1 图片渲染和透明混合速度

测试类型	混合速度/(frmae · s^{-1})		提高
	未加速版本	加速版本	百分比/%
图标跳动	3	7	133.3
过渡动画	2	7	250

从表 1 可以看出,未加速版本的动画极为缓慢,特别是过渡动画速度(每秒 3 帧降为 2 帧)。因为过渡动画需要在大区域内进行透明混合、背景图标重绘和图标绘制,而未加速版本所有的图片渲染和透明混合均由软件完成。使用硬件加速后,图标跳动和过渡动画帧率稳定,这是由于硬件加速对应各种操作消耗时间基本一致。相比未加速版本图标跳动动画提高 133.3%,而过渡动画提高 250%。

4.2 门户网站性能评估

对应一般网站,使用元素较大,网页布局复杂,硬件加速操作并不能满足所有的渲染操作,很多元素仍需要软件渲染,因此性能对比不会有上述自制网页测试结果明显。表 2 是门户网站从输入地址到第一屏完整显示所花的时间。由于测试在一定程度上受网速影响,所以表 2 结果都是通过运行几次将其中偏差较大的值去除然后求平均值得到的。

表 2 门户网站时间测试 s

测试版本	环球网	凤凰网	搜狐	腾讯	新浪
未加速版本	38.3	27.2	32.5	18.3	28.6
加速版本	10.8	13.9	16.9	8.7	14.3
降低百分比/%	71.8	48.9	48.0	52.5	50.0

实验结果表明由于硬件加速支持的功能有限,普通网页中能够被硬件加速渲染的元素有图片绘制和透明图层混合;同时绘制线条与绘制矩形也能被硬件加速,但这两种操作在普通网页中很少使用。网页仍有大量元素使用软件渲染的方式完成,比如字体绘制、圆角矩形、大量图片解码和复杂的图形变换,因此提升比例低于完全硬件加速的自制网页。但是相比纯软件渲染,部分元素硬件加速仍然明显降低了渲染时间,同时减轻了 CPU 的负担,网页的平均打开时间减少超过 48%,环球网的网页复杂性低于凤凰网和搜狐网,打开时间减少了 71.8%。

5 结语

本文研究了 WebKit 体系结构,针对嵌入式平台设计了高性能浏览器架构。实验结果表明,该混合渲染层对网站和网页动画的显示速度提升明显。本文工作只针对 MS28L 的 DirectFB 实现了硬件加速层的相关逻辑,下一步工作将进行更加普适性的优化工作,在广泛使用于嵌入式平台的 OpenGL ES 加速驱动中的后端增加该混合渲染层的实现,从而使得该浏览器在更广泛的平台得到应用。

参考文献:

[1] GROSSKURTH A, GODFREY M W. Architecture and evolution of the modern Web browser[EB/OL]. [2010 - 09 - 18]. http://www.grosskurth.ca/papers/browser-archevol-20060619.pdf.

[2] The WebKit Open Source Project[EB/OL]. [2010 - 09 - 18]. http://webkit.org.

[3] 蒋永刚. 嵌入式多媒体系统中硬件加速技术的应用[D]. 上海:上海交通大学,2009.

[4] 宋杰,曹竹冬,王书菊,等. 基于 Qt/Embedded 的 Web 浏览器的设计与实现[J]. 计算机与现代化,2010(10):136 - 138.

[5] MATTHEW C. Simple DirectMedia Layer[EB/OL]. [2010 - 09 - 18]. http://www.libsdl.org.

[6] 吴峰,王自强. 基于 FrameBuffer 的嵌入式 GUI 系统设计[J]. 计算机应用与软件,2005,22(3):128 - 130.

[7] Google. Skia. 2D Graphics Library[EB/OL]. (2008)[2012 - 12 - 01]. http://code.google.com/p/skia..

[8] 张荣芬,杨鲁平,刘宇红. 具备图形加速能力的嵌入式应用系统[J]. 中国测试技术,2005,31(4):78 - 80.

[9] Convergence Co. DirectFB[EB/OL]. [2010 - 09 - 18]. http://www.directfb.org.

[10] 杨霄雪,王力虎,叶佳宁,等. DirectFB 在嵌入式远程桌面控制系统中的应用[J]. 计算机工程与设计,2010,31(9):2127 - 2130.

[11] 王浩朋. 二维图形裁剪算法研究与改进[D]. 西安:西安电子科技大学,2011.

[12] PORTER T, DUFF T. Compoiting digital images[J]. Computer Graphics, 1984,18(3): 253 - 259.