第33卷 第20期 计 算 机 工 程 2007年10月
Vol.33 No.20 Computer Engineering October 2007
·开发研究与设计技术· 文章编号：1000—3428(2007)20—0256—03 文献标识码：A 中图分类号：TP311

基于 QT/E 的嵌入式图形支持系统

倪红波，周兴社，谷建华
(西北工业大学计算机学院，西安 710072)

摘 要：面向消费电子的嵌入式技术迅速发展，对图形支持系统提出了更高的要求。该文给出了一种资源受限系统的轻型高效的嵌入式图形系统解决方案，结合数字电视应用的具体需求，阐述了该系统的体系结构、移植技术和实现方法。该图形支持系统已经成功运行于目标平台，证明了技术方案是可行有效的。

关键词：嵌入式图形支持系统；QT/E；数字电视中间件

Embedded GUI Supporting System Based on QT/E

NI Hong-bo, ZHOU Xing-she, GU Jian-hua
(School of Computer Science, Northwestern Polytechnical University, Xi'an 710072)

【Abstract】 The fast development of embedded technology in consumer electronics requires a more applicable GUI supporting system. This paper gives a solution to a light-weight and efficient GUI supporting system for a resource limited system. With the specific requirement of DTV applications, the architecture, porting and realization of this embedded GUI supporting system is expounded. This GUI system has successfully run on the target platform, which proves the validity of the proposed solution.

【Key words】 embedded GUI supporting system; QT/E; middleware for DTV

1 概述

数字电视是典型的嵌入式设备，具有广泛的应用前景。笔者承担的国家“863”课题“面向DTV嵌入式软件平台”，是基于MIPS处理器的ATIXilleon220芯片和嵌入式Linux操作系统研究并开发符合多媒体家用平台(multimedia home platform, MHP)[1]规范的数字电子中间件。在DTV这样的资源受限系统中，为呈现高品质的图形界面，需要对底层图形库进行深入研究。QT/E是针对嵌入式系统特征的图形库，它具有跨平台性以及本身符合构件化设计思想的特点，在数字电视中间件的研究中，已被作为对上层应用提供支持的图形库。在遵循MHP规范的数字电视中间件中，QT/E库的位置如图1所示。

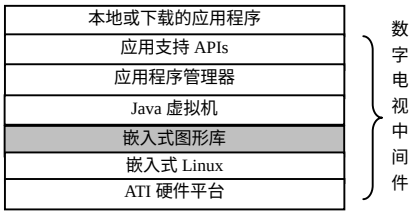

图1 QT/E 在数字电视中间件中的位置

2 嵌入式图形库概述

目前，嵌入式系统对图形库的需求越来越明显，但图形库的实现方法不尽相同。通常做法是对较成熟的图形库系统进行综合分析和研究，然后将其移植到目标平台上，并在目标平台进行功能扩展和性能优化，以实现整个图形支持系统。

比较常用的图形库系统有如下几种：QT/E, MiniGUI, Microwindow, OpenGUI等。尽管它们都能支持嵌入式系统，但是在不同的平台和应用系统中性能差异还是比较明显的，对比分析如下：

(1)QT/E是著名的QT库开发商Trolltech的面向嵌入式系统的QT版本。它具有跨平台、源码开放、界面简洁漂亮、可移植性好等特点，是许多嵌入式系统图形库实现的理想选择。

(2)Microwindow是一个著名的开放源码的嵌入式GUI软件。它提供了现代图形窗口系统的一些特性。Microwindow API支持类Win32 API，实现了一些Win32用户模块功能。采用分层设计方法，基本上用C语言实现。它支持Intel 16位和32位CPU, MIPS R4000以及ARM芯片。但作为一个窗口系统，该图形库提供的窗口处理功能还有许多不完善的地方，比如控件的实现还很不完备、键盘和鼠标的支持不够等。

(3)MiniGUI是一种面向嵌入式系统或者实时系统的图形用户界面系统。它主要运行于Linux系统上，是国内最早出现的几个自由软件项目之一。它面向基于Linux的实时嵌入式系统，使用现有成熟的图形引擎，如SVGALib，采用类似WindowsCE的线程机制，实现了简化的类Windows98风格的图形用户界面，具有资源消耗小、速度快、效率高的特点，但同时也带来了图形引擎的局限性以及由于采用多线程机制造成的系统脆弱性等问题。

3 面向数字电视的 QT/E 嵌入式图形库

3.1 QT/E 的技术特点

QT/E库是基于QT的Linux应用程序开发框架，是一个应

基金项目：国家“863”计划基金资助项目“支持新一代数字电视及嵌入式软件平台”(2003HM1101)
作者简介：倪红波(1975－)，男，博士研究生，主研方向：嵌入式网络技术；周兴社，教授、博士生导师；谷建华，教授
收稿日期：2006-10-27 **E-mail：**nihb@nwpu.edu.cn

用于嵌入式设备的图形用户界面和应用程序开发包，由挪威Trolltech公司出品。QT/E支持多种处理器，包括Inter X86, MIPS, ARM, StrongARM, Motorola68000 和PowerPC 等。QT使用面向对象语言开发，跨平台的特性让使用QT 编写的程序，只要一次编写就可以在多种平台执行。虽然是商业公司的产品，但是QT/E却提供开放源码下载，因此，QT/E 适合研究使用[2]。

QT/E 直接对显卡的帧缓冲 FrameBuffer 进行操作，去掉了对 X 窗口系统的依赖，增加了类的动态库。QT/E 是为嵌入式设备设计的 QT C++ API 的一个版本，它提供了和 QT 其他版本一样的 API 和工具，包含了支持嵌入式开发的特定的类和工具。QT/E 是一个为嵌入式设备的 GUI 和应用程序开发设计的 C++工具包。它是 QT C++ API 移植到嵌入式设备上的版本，具有如下特性：

(1)高级的绘图功能

QT/E 提供的绘图引擎 QPainter，支持多种图形设备，如：widget, pixmap, picture 和 printer。QPainter 也支持复杂的坐标转换功能，可以很容易地绘制旋转的文字或图形。QT/E 应用程序通过直接写入 Framebuffer，消除了对 X-window 系统的依赖。它提供了与 QT/X11, QT/Windows 以及 QT/Mac 等版本相同的 API 和工具集，同时也包含了支持嵌入式环境的库和工具集。

(2)对组件开发的支持

QT/E 采用消息/槽机制来取代回调(callback)机制，这种机制不仅强调类型，而且使得对象根本不需要知道彼此的情况就可以互相协作。这个特点使得它非常适合于组件编程。

(3)面向对象

正是由于面向对象在处理 GUI 应用程序上的优越性，QT/E 采用 C++来组织应用程序框架和 APIs。注重模块化，更注重于开发可重用的软件组件。支持 C++编译器，同时可以通过重新编译去扩展其他的特性，达到改善存储的目的，包含了自己的窗口化的系统，支持许多输入设备。

(4)可移植性

QT/E 是跨平台的 GUI 开发工具，可运行于其上的平台现在有 Microsoft Windows 95/98, Microsoft Windows NT, Linux, Solaris, SunOS, HP-UX, Digital UNIX (OSF/1), Irix, FreeBSD, BSD/OS, SCO, AIX 等，也可以运行于 IBM OS390 R2.5。

(5)先进的跨平台 GUI 库的实现方式

典型的实现方式有 3 种：API 加层方式(API layering)，API 模拟方式(API emulation)，GUI 模拟方式(GUI emulation)。GUI 模拟方式是其中最高效的一种，它不像 API 模拟方式那样使用不同运行平台的高层 API 调用，而是采用最基本的图形绘制功能来实现自己的 API。

QT/E 可以运行在不同的处理器上，通常使用嵌入式 Linux 作为操作系统。基于 QT/E 设计的应用程序可以在相同的编程环境下用标准的 QT API 在 Windows 和 UNIX 系统上进行开发。

QT/E 消除了对 X-window 的依赖，直接对 Framebuffe 进行操作。对于 X-window 系统，其所的占存储空间和运行空间都是嵌入式系统所不能满足的。同时，由于其功能强大、设计复杂，其运行效率并不能符合实时嵌入式中间件的要求。QT/E 库可以通过一定的配置，将库文件本身大小缩减至 4MB 左右，而且其精练的设计使得它具有较高的运行效率，这样就能满足实时嵌入式中间件对系统部件的要求。

3.2 QT/E 体系结构

QT/E 向下使用 FrameBuffer 技术，向上提供抽象的绘图、事件响应接口。QT/E 系统结构如图 2 所示。

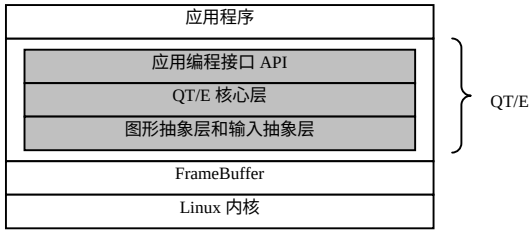

图 2 基于 QT/E 的嵌入式软件结构系统中的位置

QT/E 基本上可以分为 3 层[3]：(1)最底层是图形抽象层和输入抽象层，它们直接与显示和输入设备打交道，为核心层提供抽象的绘图接口；(2)核心层就是一个图形引擎层，它使用图形抽象层和输入抽象层提供的接口完成对画线、区域填充、文本、多边形、色彩等支持；(3)最上层是API层，提供给图形应用程序调用。QT/E 为应用开发提供了统一的编程接口，通过调用QT/E提供的API，可方便地开发出丰富的嵌入式图形用户界面，提高图形用户界面的开发效率。QT/E 的重要特点是它使QT/E上层应用程序代码建立在高层架构上，与硬件无关，在针对不同硬件和操作系统跨平台移植时，只需要改动QT/E中与硬件或操作系统相关模块，也就是图 2 中的图形抽象层和输入抽象层，便可实现在不同硬件平台上的开发应用，而且QT/E在底层定义了统一的编程接口，这为QT/E的跨平台实现带来很大的方便。QT/E库采用面向对象的方法实现，可以最大限度地简化代码编写，通过继承和派生使代码能够重用，而且QT/E的这种体系结构，使QT/E的可扩充性强，可以很方便地添加键盘、鼠标及显卡等外围设备。

4 QT/E 图形库移植

与大多数的图形库相同，QT/E 库基于 FrameBuffer 设计，通过 mmap()系统调用把 FrameBuffer 映射到进程地址空间作为显存的抽象，QT/E 库绘图函数可以直接对 FrameBuffer 进行读写，从而实现对显存的操作。这种机制对于绘图是非常方便和高效的。但是一些特殊的嵌入式设备并没有提供 FrameBuffer 接口，而只提供了简单的图形函数，如绘制点、线等图形。在需要比较复杂图形功能时，就不能满足要求，而需要使用成熟的图形库(如 QT/E 库)提供支持。由于不能使用 FrameBuffer，就必须对图形库进行改造，需要对硬件提供的图形函数进行封装，以符合图形库内部的调用接口，同时，修改图形库绘图类中的方法，剥离对 FrameBuffer 的依赖，加入封装后的图形函数。

在整个移植过程中，考虑到 QT/E 库的完整性，只作水平层面上的添加，即增加对于无 FrameBuffer 接口的硬件的支持。这样即保存了原有的功能，又适用于无 FrameBuffer 的硬件平台。接口移植工作包括以下几部分。

4.1 图形库初始化部分的修改

将图形库中负责初始化的类 QLinuxFbScreen 进行修改，该类中对 FrameBuffer 的设置部分必须去除，采用共享内存来替代 FrameBuffer 空间，使图形库的初始化工作在这段共享内存中完成。同时，将 QLinuxFbScreen 类中对 FrameBuffer 初始化的函数改为对 ATI 硬件设备的初始化函数。采用如下共享内存替代 FrameBuffer 空间。

```
shmId = shmget( (key_t)1234, 6000, 0777 | IPC_CREAT );
```

```
if ( shmId != -1 )
    shmrgn = (unsigned char *)shmat( shmId, 0, 0 );
else {
    qDebug("Cannot create a share memory");
    return FALSE;
}
if ( (int)shmrgn == -1 || shmrgn == 0 ) {
    qDebug("No shmrgn %d", (int)shmrgn);
    return FALSE;
}
```

4.2 自定义接口类的添加

在 QT/E 库中构造一个自定义接口类，这个类将 QT/E 库中的绘图函数与硬件系统的底层函数相连接，从而去除了 QT/E 与 FrameBuffer 的关系。

在本文的数字电视环境中，底层的图形函数是 ATI 公司的图形函数，构造类 QGfxAti2DM 来实现转换接口。面向图形库，在绘图函数中调用 QGfxAti2DM 类对象绘图；面向硬件，QGfxAti2DM 类的成员函数通过直接调用 ATI 相对应的绘图函数实现在 ATI 显卡上绘图。QGfxAti2DM 类的主要函数如图 3 所示。

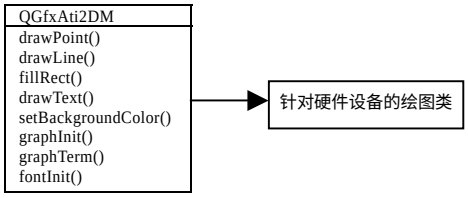

图 3　QGfxAti2DM 类的主要函数

4.3 图形库绘图过程的修改

在 QT/E 库中，图形是通过一系列绘图类实现的。复杂的组件，如 Widget, Frame, Button 等都有自己的类定义，但这些类都不负责绘制图形，它们只进行逻辑上的组件图形组合。由于任何组件都是由基本的图形组成的(点、线、矩形等)，因此实际的绘制动作需要由包含这些基本图形绘制函数的类来完成。

在 QT/E 库中 QPainter 类实现绘图引擎的功能，它支持多种图形设备，如 widget, pixmap, picture 和 printer，同时也支持复杂的坐标转换功能，可以很容易地绘制旋转的文字或图形。绘图引擎 QPainter 定义了基本图形的绘制函数，如图 4 所示，这些函数通过调用库中不同显卡类来实现绘图。由此找到了将自定义接口类对象加入整个绘图过程并替代原有显卡类对象的切入点。

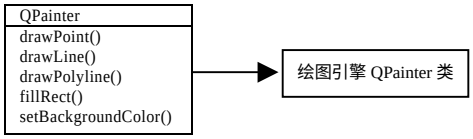

图 4　QPainter 类中主要的绘图函数

在将自定义接口类对象加入 QPainter 类相应的绘图方法中后，调用 QGfxAti2DM 类中的绘图函数替代原有的绘图函数，这样就实现了对 FrameBuffer 的剥离，完成了 QT/E 库向不支持 FrameBuffer 平台的移植。

至此，QT/E 图形库的移植基本完成。采用交叉编译工具重新生成适合在 MIPS 体系结构中应用的 QT/E 库后，可以采用它来为数字电视中间件提供必要的图形支持。

5　结束语

QT/E 是一种嵌入式图形支持系统，它具有基于组件开发、可跨平台、移植性好等特点，本文针对数字电视中间件平台的需求，详细阐述了 QT/E 的功能、性能、体系结构和移植方法。该图形支持系统已经成功运行于目标平台，证明本文的技术方案是有效可行的。本文的研究内容对其他基于 QT/E 的嵌入式系统研究和开发有较好的参考价值。

参考文献

1 石　峻. 多媒体家庭平台的系统结构与应用[J]. 数字电视与数字视频, 2002, (4): 7-9.

2 Griffith A. KDE 2/QT 编程宝典[M]. 高寿福, 张　华, 译. 北京: 电子工业出版社, 2002-01.

3 Trolltech. QT/E Whitepaper[Z]. (2004-11-20). http://www.trolltech.com/products/whitepapers.html.

(上接第 252 页)

(3)在系统正常工作时，将主服务器播出进程中的磁盘 I/O 部分中止，并在一定时间间隙后重新恢复。

在第 1 种情况下，当某节点机由于断电或网络失连等原因使得系统不再保持双机同步状态时，系统能够立即检测到故障。若故障机是主服务器，系统将输出切换到备服务器；若故障机是备服务器，系统发警报提醒监控人员进行处理以恢复双机同步状态。第 2 种情况下，主备之间的“心跳线”将立即监测到延迟情况，并将输出切换到备服务器，备服务器主动播出。第 3 种情况下，故障代理将发现磁盘 I/O 和 CPU 利用率在急剧减小，总控立即将输出切换到备服务器。原主服务器重新恢复后，总控将其纳入系统并设置为备服务器。实验结果证明，在上述 3 种情况下，系统均能立即发现故障。当切换发生时，系统播出仍然能够保持帧连续。

4　结论

本文建立系统 SPN 模型对切换时间、恢复时间等与系统可用性之间的关系进行了推导。给出了主动检测与节点报告相结合的故障检测方法，标准验时和网络验时相结合的校时方法，“心跳线”帧内校验方法，以及切入切出的实现。利用仿真故障的方法对系统进行了测试，直接证明本文所作研究是有效的。

参考文献

1 Wu Tao, Feng Dan, Zhang Jiangling. Monitoring Faults Self-consciously in the Real Time Dual Working System Based on High Performance Network[C]//Proceedings of 2002 International Conference on Machine Learning and Cybernetics. 2002, 2: 582-585.

2 姜瑞新, 蔡　凌, 汪晋宽. 冗余测试系统的测试与应用[J]. 仪器仪表学报, 2005, 26(8): 608-610.

3 谢长生, 胡庆平, 谭志虎. Heartbeat-Gear: 一种新型的实时心跳监测技术[J]. 计算机工程与科学, 2004, 26(5): 62-65.

4 胡华平, 肖晓强, 金士尧. 考虑切换的强实时双机系统的可靠性研究与实现[J]. 计算机学报, 1999, 22(10): 1080-1084.