Mini GUI 在嵌入式 Linux 下的移植方法与过程

近几年，嵌入式 Linux 系统已得到广泛的应用，相应的图形用户界面的开发也日趋重要，MiniGUI 正是其中的一个轻量级的图形用户界面支持系统。本文分析了嵌入式操作系统下图形用户界面 MiniGUI 的结构和特点，描述了将 MiniGUI 在 ARM Linux 下的移植方法与过程，并对这种基于 MiniGUI 的嵌入式系统开发做了展望。

0 前言

近年来随着嵌入式设备与市场需求的广泛结合，手机、PDA 等产品的应用对可视化操作界面的简洁和方便提出了更高的要求，这都需要一个稳定可靠的高性能 GUI 系统来提供支持。图形用户界面(Graphic User Interface，简称 GUI)的广泛流行是当今计算机技术的重要成就之一，它极大地方便了非专业用户的使用，人们可以通过窗口、菜单方便地进行操作。嵌入式系统对 GUI 的基本要求包括有轻型、占用资源少、高性能、高可靠性以及可配置等。MiniGUI 是目前比较常用的几种 GUI 系统之一，与其他的 GUI 相比，MiniGUI 最显着的特点就是轻型、占用资源少，而且在这几年的发展里，MiniGUI 已经非常成熟和稳定了，在许多产品和项目中都已得到了实际应用。

1 MiniGUI 的特点和体系结构

1. 1 MiniGUI 的特点

MiniGUI 是由原清华大学教师魏永明主持开发的轻量级图形系统，是一种面向嵌入式或实时系统的图形用户界面支持系统。它遵循 GPL 公约，是基于 SVGALib 及 LinuxThread 库的多窗口 GUI 支持系统。能跨多种操作系统，主要运行于 linux 及一切具有 POSIX 线程支持的 POSIX 兼容系统，包括普通嵌入式 Linux、eCos、uC/OS-II、VxWorks 等系统，是国内最早的自由软件之一。

MiniGUI 的主要特点有：(1) 遵循 GPL 条款的纯自由软件；(2) 提供了完备的多窗口机制；(3) 多字符集和多字体支持，目前支持 ISO8859-1、GB2312 及 Big5 等字符集，并且支持各种光栅字体和 TrueType、Type1 等矢量字体；(4) 全拼和五笔等汉字输入法支持；(5) BMP、GIF、JPEG 及 PCX 等常见图像文件的支持；(6) Windows 的资源文件支持，如位图、图标、光标、插入符、定时器及加速键等；(7) 可移植性好。

1.2 MiniGUI 的体系结构

1.2.1 多线程的分层设计

从整体结构上看，MiniGUI 是分层设计的，结构如图 1 所示。在最底层，GAL（图形抽象层）和 IAL（输入抽象层）及鼠标和键盘的驱动；中间层是 MiniGUI 的核心层，包括窗口系统必不可少的各个模块；最顶层是 API，即编程接口。GAL

和 IAL 为 MiniGUI 提供了底层的 Linux 控制台或者 X Window 上的图形接口以及输入接口，而 Pthread 用于提供内核级线程支持的 C 函数库。利用 GAL 和 IAL，大大提高了 MiniGUI 的可移植性，并且使程序的开发和调试变得更加容易。

MiniGUI 本身运行在多线程模式下，它的许多模块都以单独的线程运行，同时，MiniGUI 还利用线程来支持多窗口。从本质上讲，每个线程有一个消息队列，消息队列是实现线程数据交换和同步的关键数据结构。一个线程向消息队列中发送消息，而另一个线程从这个消息队列中获取消息，同一个线程中创建的窗口可共享同一个消息队列。一个线程向消息队列中发送消息，而另一个线程从这个消息队列中获取消息，同一个线程中创建的窗口可共享同一个消息队列。利用消息队列和多线程之间的同步机制，可以实现下面要讲到的微客户/服务器机制。

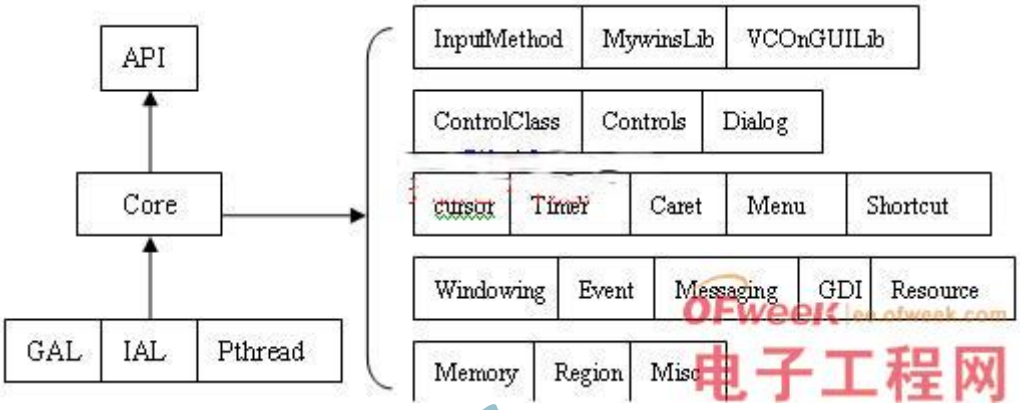

图 1　MiniGUI 的层次结构

1.2.2 微客户/服务器结构

在多线程环境中，与多进程间的通讯机制类似，线程之间也有交互和同步的需求。比如， 用来管理窗口的线程维持全局的窗口列表， 而其他线程不能直接修改这些全局的数据结构， 而必须依据“先来先服务”的原则，依次处理每个线程的请求，这就是一般性的客户/服务器模式。MiniGUI 利用线程之间的同步操作实现了客户线程和服务器线程之间的微客户/服务器机制。

2 开发环境

H2410EB 开发板由北京恒颐高科技术有限公司设计开发，它基于 Samsung 公司的 S3C2410A20 嵌入式 ARM 处理器。S3C2410A20 内嵌 ARM920T 核，带有全性能的 MMU，具有高性能、低功耗、低成本、小体积等优点，适用于手持设备、汽车等领域。

H2410EB 除带有大容量的 SDRAM 和 Flash 以外，还扩展了 RS-232C 串行接口、10Mbps 以太网接口、触摸屏接口、音频输入/输出接口、USB Host、USB Slave、UART 接口、IIC 接口、用户自定义键盘、LCD 显示器，方便用户使用和进行参考设计。它支持嵌入式 Linux 操作系统的运行，支持 MP3/MPEG 播放、GUI、Web 服务及其它服务，同时可根据用户需求开发特定软件与设备驱动程序。

操作系统采用裁减后的 Linux，Linux Kernel 版本为 v2.4.18，系统使用的交叉编译器是 arm-Linux-gcc。另外，还有驱动程序源码和测试程序源码等代码模块。

3 MiniGUI 的移植

本文中使用的主机系统为 Red Hat Linux 9.0,移植目标系统为 Linux2.4.18，MiniGUI 的版本是 1.6.9。在主机上交叉编译 MiniGUI 链接库，然后将针对目标机编译的库文件，与根文件系统一起烧写到目标板的 RAM 空间，以后将运行在目标板上的图形用户界面直接链接到该库，脱离主机独立运行。

3.1 Linux 交叉编译环境的构建

GUI 的编译通常都是在 PC 机上执行的，也就是说，编译器本身能够在 PC 机上执行，同时编译源代码生成的二进制文件必须能在目标机上执行，这类编译器通常称为交叉编译器。对于 ARM 平台，我们安装了 cross-arm-binutils-2.10-1.i386.rpm、cross-arm-gcc-2.95.3-2.i386.rpm、cross-arm-glibc-2.1.3-2.i386.rpm 这三个包。这些包都可以从网上免费获取。arm-binutils 这个包一般包含了一些针对 ARM 平台的二进制工具，比如 arm-strip、arm-ar 等命令；arm-glibc 这个包包含的是标准 C 的函数库的 ARM 的版本以及对应的头文件；arm-gcc 中包含的则是生成 ARM 平台代码的 x86 上的交叉编译器。执行 rpm 命令将这些包安装到 PC 机上，若不在系统默认搜索目录下，则必须将安装目录加到系统的 PATH 环境变量中，这样在每次编译时系统才能找得到编译器。

以 root 用户登陆 Linux 系统，在主机上用 rpm 指令安装交叉编译工具，arm-linux-gcc 将被安装到/usr/local/arm/2.95.3/下面。此时，gcc 为 /usr/local/arm/2.95.3/bin/arm-linux-gcc，而它的 include 为 /usr/local/arm/2.95.3/arm-linux/include，对应的 lib 为 /usr/local/arm/2.95.3-/arm-linux/lib。然后，在你的 bashrc 中添加环境变量即可。执行 vi .bashrc，最后一行加入：export PATH=$PATH:/usr/local/arm/2.95.3/bin 路径,保存退出后执行 source .bashrc。

另外需要注意的是,编译时所用的函数库版本要与目标版上运行时所用的函数库版本一致。经过上述步骤，就已经建立了交叉编译环境，接下来的就是进行 MiniGUI 的选项配置和交叉编译。

3.2 MiniGUI 的配置和交叉编译

我们可以从网上免费得到 MiniGUI-1.6.9 的资源文件压缩包,MiniGUI1.6.9 的源程序包包括以下三个部分：libminigui-1.6.9.tar.gz－MiniGUI 函数库源代码；miniguires-1.6.9.tar.gz－MiniGUI 所使用的资源，包括基本字体、图标、位图、输入法等；mde-1.6.9.tar.gz－MiniGUI 的综合演示程序。

3.2.1 MiniGUI 函数库的安装和编译

进入目录 libminigui-1.6.9，再运行./configure 脚本：CC=
arm-linux-gcc\ ./configure --prefix=/mnt/nfs/local\
--build=i386-linux\ --host=arm-linux\ --target=arm-linux\
--disable-lite\ --disable-micemoveable\ --disable-cursor\

在这里，CC 是用来指定所使用的编译器，arm-linux-gcc 即为安装到主机上的交叉编译工具。另外，－－prefix 为 MiniGUI 函数库的安装目标路径；－－build 是指执行编译的主机；－－host 交叉编译后的程序将运行的系统；－－target 是运行该编译器所产生的目标文件的平台；－－disable－lite 建立 MiniGUI-Threads 版本的应用程序；－－disable－micemoveable 禁止窗口移动；－－disable－cursor 由于系统采用触摸屏，所以用此选项用来关闭鼠标光标显示。

如果运行./configure 脚本成功通过，就可继续进行下面的编译了，执行 make 和 make install 命令编译安装 libminigui。这里要注意的是，执行 make install 命令时要切换到 Root 用户权限下，不然安装时没法把文件装到指定目录下。安装成功后，MiniGUI 的函数库和头文件以及配置文件等资源将被安装到 /usr/local/arm/2.95.3/arm-linux/目录中，具体情况为：函数库被装在 lib/ 子目录中；头文件被装在 include/ 子目录中；手册被装在 man/ 子目录中；配置文件被装在 etc/ 子目录中。

3.2.2 MiniGUI 资源的编译安装

主机上解压资源文件：tar zxf miniguires-1.6.9.tar.gz，可生成 miniguires-1.6.9 目录。在安装之前先要修改目录中的 configure.linux 文件，执行 vi configure.linux 打开文件，把 prefix 选项部分的默认值 /usr/local/ 改为 /usr/local/arm/2.95.3/arm-linux/，这样运行 make install 安装命令后 MiniGUI 资源将被安装到目标系统中的 /usr/local/arm/2.95.3/arm-linux/lib/minigui-/res 的目录下。

3.2.3 实例程序的编译安装

解压 mde-1.6.9.tar.gz 并进入该目录，修改目录下配置文件 configure.in，把其中的 AC_CHECK_HEADERS(minigui/commmon.h, have_libminigui=yes, foo=bar)中的 minigui/-commmon.h 改为$prefix/include/minigui/common.h，来指定交叉编译时搜 minigui 的头文件路径，防止编译时系统找不到头文件；在所有 LIB="$LIB 后加入 -L{prefix}/lib 来指定编译时所需要库文件的路径。并将 libpopt-dev-arm-cross-1.6.tgz 解压所生成的头文件和库文件分别放入目标目录的 include 和 lib 中，用以支持 mde 中程序在 ARM 下的交叉编译。然后执行./autogen.sh，重新生成 configure 脚本，使用上面配置的脚本然后执行 make 命令，即可完成实例程序的编译。

4 拷贝 MiniGUI 资源到开发板

编译完 MiniGUI 和实例程序之后，需要把 MiniGUI 库、资源和应用程序拷贝到为目标机器准备的文件系统目录中，然后生成文件系统映像，再下载到目标板上运行。可以通过串口、USB 口或以太网口将文件系统映像下载到目标机器中。在执行程序之前，还有一件重要的事情要做，就是在开发板上的 Linux 中配置好 MiniGUI 的运行环境。

5 板载 Linux 的环境配置

MiniGUI 可以使用多种图形引擎进行图像显示，有 qvfb、SVGALib、LibGGI 等等，当然也可以自己编写一个图形引擎供 MiniGUI 使用。这里我们使用 qvfb 来作为 MiniGUI 的图形引擎进行图像显示。qvfb（vitural framebuffer）是在宿主机上模拟帧缓冲的，它是 X Window 用来运行和测试应用程序的系统程序，使用了共享存储区域（虚拟的帧缓冲）来模拟帧缓冲并且在一个窗口中模拟一个应用来显示帧缓冲。

首先对 qvfb 进行安装，可以从网上下载，下载下来后进行解压：tar zxf qvfb-1.0.tar.gz 并进入到 qvfb-1.0 目录，执行./configure 脚本后即可用 make 和 make install 命令进行编译安装。

更改 MiniGUI 的配置文件 MiniGUI.cfg 设置设备驱动程序，设置显示区域及字体等内容。修改/usr/local/etc 目录下的配置文件 MiniGUI.cfg，将其中的驱动引擎 gal_engine 和 ial_engine 设置为 qvfb，再将其中 qvfb 的 defaultmode 设置为合适的显示模式。然后把 qvfb 加到可执行路径中去，执行 vi .bashrc 命令，在 .bashrc 最后面加上 export PATH=/usr/local/arm/2.95.3/bin -:$PATH，保存退出后用 source .bashrc 命令执行一下即可。

在 X Window 中，打开一个终端仿真程序，执行 qvfb &命令。在 qvfb 中选中 File Configure，将 qvfb 设置成嵌入式开发系统的液晶屏的大小。合理设置 MiniGUI 的配置文件后，接着就可以运行 MiniGUI 应用程序了。

执行应用程序顺利的话，屏幕上可以看到程序的运行界面。至此，MiniGUI 已经成功移植到目标系统上。此后，我们可以根据需要，继续修改 MiniGUI 库函数及各种资源，并且编写自己的应用程序，使图形用户界面更加完善。

6 结束语

随着嵌入式产品应用领域的日益增长，开发出优秀的人机交互界面，是嵌入式发展的趋势，拥有广阔的市场前景。MiniGUI 可以稳定可靠的运行在 Linux 系统下，通过上述具体的移植和后续的 MiniGUI 下嵌入式软件的开发过程，能快速构建一个嵌入式可视化软件系统，相信这种嵌入式系统将会得到越来越多的应用。

本文创新点：成功实现了图形用户界面 MiniGUI 的移植开发和对开发板 H2410EB 的支持，这也适用于其他多种开发板，完成了构建嵌入式图形界面系统的前期工作。