嵌入式 Linux 系统在 PDA 设备中的应用

桑 江, 陈 震
(吉林大学 计算机科学与技术学院, 吉林 长春 130012)

摘 要:嵌入式 Linux 操作系统应用于 PDA 设备时, 会遇到几个主要技术难题, 比如如何引导加载内核, 如何开发设备驱动、如何架构文件系统等。针对以上问题, 根据项目中使用的特定硬件环境, 提出了 PDA 设备中嵌入式 Linux 系统的开发方法和实现过程。阐述的实现方法能够在 PDA 设备中建立起高效、稳定、安全的嵌入式 Linux 系统。

关键词:嵌入式;引导装载程序;图形用户界面;Qt/Embedded
中图法分类号:TP316 **文献标识码**:A **文章编号**:1000-7024(2005)04-1016-04

Application of embedded Linux system in PDA devices

SANG Jiang, CHEN Zhen
(Institute of Computer Science and Technology, Jilin University, Changchun 130012, China)

Abstract: Several difficult problems will emerge while linux operating system applies to PDA devices, such as how to boot kernel, how to develop driver program of devices, how to integrate root filesystem and so on. To solve these problems, the method of development and the process of realization is introduced based on the environment of special hardware. Finally, embedded linux system can be built through the implementation method described and can run in the PDA devices effectively, stably and safely.

Key words: embedded; bootloader programme; GUI; Qt/embedded

1 引 言

嵌入式 Linux 操作系统拥有很多优势,但应用在 PDA 设备上仍是一个新课题,面临诸多的挑战。本文就嵌入式 Linux 应用于 PDA 设备所用到的技术和可能遇到的问题进行了详细的分析,并针对技术难题提出了解决的方法。

2 项目需求及硬件平台

本文讨论的项目是笔者参加的一个 PDA 手机开发项目,该项目需要包含以下主要功能:电话及短信功能、GPRS 拨号上网功能、音频及视频播放功能、任务管理功能、录音功能、记事本功能、科学计算器功能、日历功能等。

此 PDA 项目采用的硬件平台包含以下主要部件。

MC9328MX1(以下简称 MX1)是 Motorola 公司基于 ARM 核心的第 1 款 MCU,主要面向高端嵌入式应用。内部采用 ARM920T 内核,并集成了 SDRAM/Flash、LCD、USB、蓝牙、多媒体闪存卡(MMC/SD, Memory Stick)、CMOS 摄像头等控制器。

3 准备开发环境

因为 Motorola MX1 处理器是基于 ARM 架构的,所以我们需要创建基于 ARM 的交叉开发环境,进行 Linux 嵌入式开发。

表 1 PDA 硬件设备表

处理器	Motorola 公司龙珠系列的 MC9328 MX1
存储器	64M 字节 SDRAM 存储器,32M 字节 NOR FLASH 存储器
存储器扩展	MMC 卡存储器扩展
显示器	采用 Sharp 公司 3.5 寸 TFT LCD,分辨率 320×240,65K 色
输入设备	触摸屏及按键
GSM/GPRS 无线数据传输	采用 WAVECOM 公司的双频 900/1800M 无线数据传输模块
PC 同步接口	USB Slave 接口和 UART 接口

这套 ARM 交叉开发环境由一套用于编译、汇编和链接内核及应用程序的组件组成。这些组件包括:

Binutils——用于操作二进制文件的实用程序集合。包括诸如 ar、as、objdump、objcopy 等实用程序;

Gcc——GNU C 编译器;

Glibc——所有用户应用程序都将链接到的 C 库(没有使用任何 C 库函数的内核和其它应用程序可以在没有该库的情况下进行编译)。

由上述组件建立的交叉编译器运行在 x86 处理器上,却可以编译我们需要的 ARM 处理器的指令。需要注意的是这个彻底的构建过程对内存和硬盘的需求是巨大的。如果没有

收稿日期:2004-05-16。
作者简介:桑江(1979-),男,黑龙江鹤岗人,硕士生,研究方向为嵌入式系统和网络安全; 陈震,教授,硕士生导师,研究方向为数据库系统、网络安全和嵌入式系统等。

足够的内存和硬盘空间,那么在构建阶段会冒出许多问题。

4 嵌入式 Linux 系统的实现

嵌入式 Linux 开发大致涉及 5 个层次:引导装载程序、Linux 内核、设备驱动程序和图形用户界面(或称 GUI)以及整合文件系统。在本文中,我们将集中讨论涉及这 5 层的一些基本概念;深入了解引导装载程序、内核和文件系统是如何交互的;并详细介绍以上几部分内容的技术实现。

4.1 引导装载程序 BootLoader

嵌入式系统中,BootLoader 的意义与作用于 PC 上的 BIOS 有点类似,它对开发板上的主要部件如 CPU、SDRAM、FLASH、串口等进行了初始化。引导装载程序都用于基于 ARM 设备上的 Linux,本 PDA 系统采用 Blob 作为引导装载程序。

将引导装载程序安装到目标的闪存后,它就会执行上面提到的所有初始化工作,准备接收来自主机的内核和文件系统。一旦装入了内核,引导装载程序就将控制转给内核。

在大多数情况中嵌入式设备没有 BIOS,那么由谁来引导载入装载程序呢?要解决这个问题最好的方法是采用专用软件或微小的引导代码(Tiny BootCode)。

专用软件可以直接与远程系统上的闪存设备进行交互并将引导装载程序安装在闪存的给定位置中。这种软件使用目标板上的 JTAG 端口,它是用于执行外部输入(通常来自开发主机)的指令的接口。JFlash-Linux 就是这种用于直接写闪存的流行工具。可以在命令行中用以下命令启动它:

Jflash-Linux <bootloader>

微小的引导代码指的是几个字节的指令,它将初始化一些 DRAM 设置并启用目标板上的一个串行(或者 USB,或者以太网)端口与主机程序通信。然后,主机程序或装入程序可使用这个连接将引导装载程序传送到目标上,并写入闪存。

Blob 经过适当的修改才能够在芯片上正常工作。我们采用 Blob BootLoader 为开发原型,将 Motolola MX1 的硬件驱动合并到 LART(Linux Advanced Radio Terminal)设备的文件中,进行更改的驱动主要为 UART 和 FLASH 驱动,Blob 的所有地址操作都在 MMU 关闭状态下完成,所有地址为物理地址。

整个 Blob BootLoader 文件由两部分映像而成,第 1 部分为 start.S(最大 1K),第 2 部分为 trampoline.S(最大 63K),两个部分最大不超过 64K。如图 1 所示。

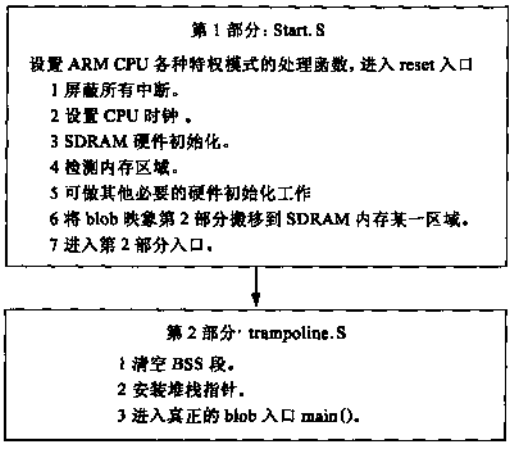

图 1 BLOB-MX1 的结构图

从图 1 可以看出,第 1 部分的 Start.S 主要要完成对第 2 部分 trampoline. S 的装入工作。实际上第 1 部分是在 flash 上面运行的,由于 flash 运行的速度比较慢,所以要求这部分代码要足够的短。在第 1 部分的第 6 个步骤中,第 2 部分的整个代码区域都从 flash 中装入 sdram。第 2 部分的代码完成一些 bss 和堆栈的初始化工作后才真正地调用了 Blob 的入口 main()。main 完成的工作:①初始化串口驱动,设置串口;②初始化命令行;③设置 blob 内部状态参数;④计算内存数量;⑤将 FLASH 中 BLOB 映像、KERNEL 映像、RAMDISK 映像搬移到 SDRAM 内存中;⑥初始化 usb 设备;⑦ 加入中断处理函数,以便进行 usb 传输;⑧ 进入命令行交互状态。

4.2 Linux 内核配置及编译

在嵌入式 Linux 设计与应用中,为了使目标板能够支持更多的外设,将不得不自己设计驱动程序,或者有时需要重新设计内核中的某些模块。为了正确合理地设置内核编译配置选项,只编译系统需要的功能的代码,一般主要考虑下面 4 点:① 自己定制编译的内核运行更快(具有更少的代码);②系统将拥有更多的内存(内核部分将不会被交换到虚拟内存中);③不需要的功能编译进入内核可能会增加被系统攻击者利用的漏洞;④将某种功能编译为模块方式会比编译到内核内的方式速度要慢一些。

虽然以上 4 点是针对成熟的 Linux 套件如 Redhat Linux(运行于 X86 平台)而言,但也是为建造嵌入式 Linux 操作系统做准备,也是必由之路。

在本系统中,我们采用 2.4.18 版本的内核,同时还需要打上一些额外的补丁,以使内核支持我们的硬件系统。

linux-2.4.18.tgz----------2.4.18 内核源码

patch-2.4.18-rmk4-mx1bsp0.2.0----------MOTOLORA patch for Linux 2.4.18

据以上内核配置原则及本 PDA 项目的实际需求,对内核功能和部件进行有效的裁减,利用交叉编译环境,用 arm-linux-gcc 编译器编译出运行于 arm 环境下的嵌入式 Linux 内核。

4.3 设备驱动程序

嵌入式系统通常有许多设备用于与用户交互,像触摸屏、小键盘、滚动轮、传感器、RA232 接口、LCD 等。除了这些设备外,还有许多其它专用设备,包括闪存、USB、GSM 等。内核通过所有这些设备各自的设备驱动程序来控制它们,包括 GUI 用户应用程序也通过访问这些驱动程序来访问设备。本节着重讨论在本 PDA 嵌入式环境中使用的一些重要设备的设备驱动程序。

4.3.1 帧缓冲区驱动程序

这是最重要的驱动程序之一,通过这个驱动程序才能使系统屏幕显示内容。帧缓冲区驱动程序通常有 3 层。最底层是基本控制台驱动程序 drivers/char/console.c,它提供了文本控制台常规接口的一部分。通过使用控制台驱动程序函数,我们能将文本打印到屏幕上,但图形或动画还不能(这样做需要使用视频模式功能,通常出现在中间层,也就是 drivers/video/fbcon.c 中)。这个第 2 层驱动程序提供了视频模式中绘图的常规接口。

帧缓冲区是显卡上的内存，需要将这段内存映射到用户空间以便可以将图形和文本写到这个内存段上，然后这些信息将反映到屏幕上。对帧缓冲区的支持提高了绘图的速度和整体性能。这也是顶层驱动程序引人注意之处，它们是非常特定于硬件的驱动程序，需要支持显卡的各种硬件特性，像启用/禁用显卡控制器、深度和模式的支持以及调色板等。所有这 3 层都相互依赖以实现正确的视频功能。与帧缓冲区有关的设备是 /dev/fb0（主设备号 29，次设备号 0）。

4.3.2 输入设备驱动程序

触摸屏是用于嵌入式设备的最基本的用户交互设备之一。触摸屏设备的主要功能是随时报告用户的触摸并标识触摸的坐标。这通常在每次发生触摸时通过生成一个中断来实现。然后，这个设备驱动程序的角色是每当出现中断时就查询触摸屏控制器，并请求控制器发送触摸的坐标。一旦驱动程序接收到坐标，它就将有关触摸和任何可用数据的信号发送给用户应用程序，并将数据发送给应用程序，然后用户应用程序根据它的需要处理数据。

同理 MX1 板上的几个操作按键的驱动程序也完成了非常重要的功能，当按键被按下时，也产生一个中断，驱动程序将这个 keydown 事件的信号传送给应用程序，应用程序就会产生例如焦点转移、选中、取消等相关操作。

4.3.3 闪存 MTD 驱动程序

MTD 驱动程序是专门为基于闪存的设备所设计的，通常具有很好的支持、管理和基于扇区的擦除和读写操作的接口。Linux 下的 MTD 驱动程序接口被划分为两类模块：用户模块和硬件模块，其中用户模块提供从用户空间直接使用的接口，硬件模块提供对内存设备的物理访问。

下面介绍一下 MTD 驱动程序的相关设置。

为了访问特定的闪存设备并将文件系统置于其上，需要将 MTD 子系统编译到内核中。这包括选择适当的 MTD 硬件和用户模块。

有两个流行的用户模块可启用对闪存的访问：MTD_CHAR 和 MTD_BLOCK。MTD_CHAR 提供对闪存的原始字符访问，而 MTD_BLOCK 将闪存设计为可以在上面创建文件系统的常规块设备（像 IDE 磁盘）。与 MTD_CHAR 关联的设备是 /dev/mtd0、mtd1、mtd2 等，而与 MTD_BLOCK 关联的设备是 /dev/mtdblock0、mtdblock1 等。由于 MTD_BLOCK 设备提供像块设备那样的模拟，通常更可取的是在这个模拟基础上创建像 FTL 和 JFFS2 那样的文件系统。

我们的系统需要建立一个 JFFS2 文件存储区来保存一些必要的配置文件和用户的资源文件（在下面相关内容中详细说明）。为了进行这个操作，需要创建分区表将闪存设备中的一部分建立一个 mtd 分区。样本分区表可能包含以下信息：

```
struct mtd_partition sample_partition = {
  {                          /* First partition */
    name : JFFS2,            /* JFFS2 filesystem */
    size   : 0x00200000,     /* Occupy last 2M of flash */
    offset : 0x01b00000      /* Append after kernel section */
  }
}
```

上面的分区表中指定的偏移量是 0x01b00000，它是指 32M 闪存地址中减去 blob 和 kernel 占用的 3M 后为起始地址计算的偏移地址。如上的分区表使用了 MTD_BLOCK 接口对闪存设备进行分区。在系统启动时，可以把这个 JFFS2 存储区域 mount 到 root 文件系统的某个目录加以利用，可以通过执行 mount -t jffs2 /dev/mtdblock0 /jffs2 这样的类似命令来完成。

Linux 中 MTD 子系统的主要目标是在系统的硬件驱动程序和上层或用户模块之间提供通用接口。硬件驱动程序不需要知道如 JFFS2 和 FTL 那样的用户模块使用的方法。所有它们真正需要提供的就是一组对底层闪存系统进行 read、write 和 erase 操作的简单例程。

4.4 图形用户界面（GUI）

从用户的观点来看，图形用户界面（GUI）是系统的一个至关重要的方面，用户通过 GUI 与系统进行交互，所以 GUI 应该易于使用并且非常可靠。但它还需要是有内存意识的，以便在内存受限的微型嵌入式设备上可以无缝执行。所以，它应该是轻量级的，并且能够快速装入。

4.4.1 嵌入式 Linux 常用的 GUI 系统

MiniGUI：由原清华大学教师魏永明先生开发，是一种面向嵌入式系统或者实时系统的图形用户界面支持系统。

MicroWindows：提供了现代图形窗口系统的一些特性。MicroWindows API 接口支持类 Win32 API，接口试图和 Win32 完全兼容。它还实现了一些 Win32 用户模块功能。Micro Windows 采用分层设计，以便不同层面能够在需要的时候改写。

Qt/Embedded：QT 是 Trolltech 公司的产品，Qt/Embedded 以原始 Qt 为基础，并做了许多出色的调整以适用于嵌入式环境。Qt Embedded 通过 Qt API 与 Linux I/O 设施直接交互，而且面向对象的体系结构使代码结构化、可重用并且运行快速。

QPE：Trolltech 还推出了 Qt 掌上机环境（Qt Palmtop Environment，俗称 QPE）。它提供了一个基本桌面窗口，并且该环境为开发提供了一个易于使用的界面。Qpe 包含全套的个人信息管理（Personal Information Management（PIM））应用程序、因特网客户机、实用程序等。它的优点包括：面向对象的体系结构有助于更快地执行；占用很少的资源，大约 800 K；抗锯齿文本和混合视频的像素映射。

4.4.2 本项目的 GUI 实现

鉴于本项目的实际情况，选用的 GUI 就是 Qt/Embedded 2.3.7+QPE1.7.0 集成环境，QPE1.7.0GUI 是一个专为手持设备开发的桌面环境，环境中已经预先包含了大多数本 PDA 项目需求中应有的大部分应用程序包括记事本、任务管理器、科学计算器、媒体播放器、图片浏览器等。我们在 GUI 部分中需要做的主要集中在两方面。

（1）针对本系统硬件平台开发的输入设备驱动程序，对 Qt/Embedded 源码进行相应的修改。比如针对触摸屏驱动，就要修改 Qt/Embedded 中鼠标设备初始化以及从触摸屏驱动读取点击事件及数据转化为鼠标点击事件相应的源码，这样才能通过触摸屏正常地操作 GUI。同理，本 PDA 设备上仅有的几个按键不同于普通 PC 键盘，也应根据新开发的键盘驱动来修改 Qt/Embedded 相关源码，使 GUI 能正确地响应本系统中

的键盘事件。

(2) 根据本项目的需求,现有的QPE桌面环境还没有包含所有应具备的应用程序,比如电话拨打程序、短信接收程序、中文输入法程序、网页浏览程序等。所以需要在 QPE 现有应用程序的基础上,在 QPE 开发环境中完成以上所说的应用程序,另外,整个桌面环境的中文化也是很重要的一部分工作,但庆幸的是在 QPE 环境中开发新的应用程序以及实现 QPE 桌面环境的本地化是非常容易的事情,在这里就不再详细说明了,有兴趣的读者可以察看有关 Qt/Embedded 和 QPE 的技术文档。

4.5 整合文件系统

4.5.1 架构 root 文件系统

我们建立一个工作目录当作 root 文件系统的根目录,在这个目录中创建 root 文件系统所需的全部目录和文件。

在这个步骤中我们会采用嵌入式 Linux 系统安排 root 文件系统时的一个常用的利器:BusyBox。Busybox 编译出一个单个的独立执行程序,就叫做 busybox。它可以根据配置,执行 ash shell 的功能以及几十个各种小应用程序的功能,大小也不过 300K 左右,这就大大节省了有限的资源。用户还可以根据自己的需要,决定到底要在 busybox 中编译进哪几个应用程序的功能。

需要特殊说明的是,像内核一样,所有整合进 root 文件系统的二进制执行文件和共享库都要经过 arm 交叉编译环境编译成 arm 格式的文件,这些文件包括 busybox 等命令文件和/lib 等目录下的共享库文件以及 Qt/Embedded 和 QPE 环境中包含的应用程序执行文件和共享库文件。

4.5.2 制作文件系统映像文件

根据本项目的实际情况,我们的文件系统映像分为两个部分,第 1 部分采用 cramfs 文件系统格式来容纳上述工作目录中的 root 文件系统的内容,并且它是只读的;第 2 部分是建立在 mtd 设备上层的 jffs2 文件系统用来保存那些需要可写的内容。

(1) cramfs 格式的 root 文件系统:考虑到根目录下面的绝大多数内容在系统运行的时候是不需要被改写的,我们决定选择只读的压缩文件系统 cramfs 来容纳根目录下面的绝大多数内容。

我们需要把工作目录下的全部内容制成一个 cramfs 的 image 文件。这可以用 mkcramfs 命令完成。得到了一个 rootfs.img 文件。cramfs 的压缩效率一般都能达到将近 50%,这样一来,原本超过 40M 的 root 文件系统制成 cramfs 映像文件后就是 20M 左右,也就是说在 flash 盘中也就占用和映像文件相同的空间,这极大地节省了有限的资源。

(2) 制作 jffs2 文件系统:在我们的系统中有一部分文件需要可保存,这些文件主要是 QPE 环境的一些配置文件以及某些应用程序可能要保存的资源,比如电话程序需要下载的铃声、上网时需要保存的资料等,所以我们就要预留一个可写的空间,能够保存GUI系统的设置和用户自己保存的文件资源。鉴于此目的,我们把这部分内容集中在一起制成 jffs2 文件系统保存在内核中预定出来的 mtd 分区上。

在 Linux 下,用 mkfs.jffs2 命令创建 JFFS2 文件系统。

4.6 烧录闪存

MX1 芯片板上集成的闪存容量是 32M,实际的物理地址是从 0x10000000 至 0x12000000。我们制作完成的 kernel 在 blob 环境通过 xwrite kernel 命令烧录到 0x10100000 起始的闪存区域中,而我们制作的 rootfs.img 映像文件则通过 xwrite ramdisk 烧录到 0x10300000 起始的闪存区域中。另外我们制作的 jffs.image 映像文件是在系统启动后通过 cp jffs.image /dev/mtdblock0 命令来写到闪存的最后两兆的闪存区域里,而 blob 本身则是存在 0x10000000 开始的 1M 空间里。具体的存储布局如图 2 所示。

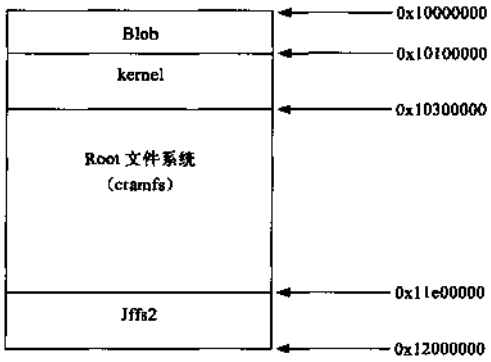

图 2 闪存存储布局

5 结束语

按照以上的实现方法,基于嵌入式 Linux 开发的本 PDA 手机系统就基本完成了。目前我们在基于 Motorola MX1 的硬件平台上已经实现了引导加载程序、定制的内核、触摸屏等各种设备驱动程序、root 文件系统以及项目需求的全部 GUI 应用功能,并且系统也继承了 Linux 占用资源少、运行效率高等种种优点,在硬件平台上能够长期稳定地运行。嵌入式 Linux 系统在 PDA 设备上的应用现在已经不是一种新的尝试,而是越来越成熟的开发选择,相信不久的将来会有多款的 Linux PDA、Linux 手机出现在我们的生活中,Linux 系统的"无线"春天即将来临。

参考文献:

[1] 沃尔夫. 嵌入式计算系统设计原理[M]. 北京:机械工业出版社, 2002.351-353.

[2] Craig Hollabaugh. 嵌入式 Linux 一硬件、软件与接口[M].北京:电子工业出版社, 2003. 20-44.

[3] 王学龙. 嵌入式 Linux 系统设计与应用[M]. 北京:清华大学出版社, 2001.342-421.

[4] Alessandro Rubini. Linux 设备驱动程序[M]. 北京:中国电力出版社, 2000.

[5] Alessandro Rubini, Jonathan Corbet. Linux 设备驱动程序[M]. 第 2 版. 北京:中国电力出版社, 2002.

[6] 毛德操, 胡希明. Linux 内核源代码情景分析[M]. 杭州:浙江大学出版社, 2001.

[7] Daniel Solin. 24 小时学通 Qt 编程[M]. 北京:人民邮电出版社, 2000.