基于嵌入式 Linux 的电子书包设计与实现

Designing and Implementing an Embedded Linux Based Ebook

陈　雷　王文杰　（中国科学院研究生院 北京市 100049）

摘　要：嵌入式设备越来越广泛的应用于各个领域，嵌入式系统的研究与开发也成为一个热点。Linux 由于其种种优点成为很多厂家开发嵌入式应用产品的底层操作系统，文中描述了基于 Intel(r) 公司 Assabet 硬件平台，应用嵌入式 Linux 操作系统来开发手持式电子书包(Ebook)产品的技术架构，提出了手持设备中嵌入式 Linux 系统开发的方法和实现过程，以及探讨了在开发过程中所需要注意的一些问题。

关键词：嵌入式 Linux；装载器 驱动程序 液晶显示器

1　引言

经过多年的发展，作为计算机应用最早领域之一的嵌入式系统在各个行业已得到了广泛的应用和发展。

硬件成本的不断降低以及性能的逐步提高，使得对于一些高端嵌入式系统，例如，智能手机，个人多媒体终端，个人游戏终端等的需求急剧增加。Linux 是一个开发源码的操作体统。由于具有稳定、网络功能强大、配置灵活、可移植性强、开发资源丰富等特点，被越来越多的厂商用所采用。本文介绍了使用嵌入式 Linux 开发的，基于 Intel 公司 SA1110 处理器 Assabet 硬件开发平台的电子书包(Ebook)设备的设计和实现。

2　系统设计

2.1　系统描述

本文讨论的系统是一个笔者参加的学生用手持电子书包设备。其主要目标是实现课本的电子化，即该设备中存储有所有学生所需的电子课本资料(文字，图片，动画等等)。学生在课堂上通过该设备进行课程学习，课后可以在该系统上完成以及提交作业。

2.2　硬件平台

本系统基于 Intel 公司的 Assabet 开发板，该平台使用的处理器为基于 ARM 公司的 32 位处理器核心 SA1110。根据本项目需要对其修改，主要改动包括增加外设电路以及出于成本节约目的而进行的电路修改。本系统主要硬件设备如表 1 所示：

表 1　Ebook 硬件设备表

CPU 处理器	Intel 公司 SA1110
内存	64MB SDRAM
存储器	64MB NOR Flash
显示设备	Hitachi SX21V001 – Z4 STN 液晶显示器，分辨率 640x480 256 色
输入设备	Cypress USB host 控制器 PS2 键盘接口 触摸屏 按键 麦克风
存储扩展	PCMCIA 插槽 MMC WinBond 控制器
网络设备	10MB 以太网卡 Modem IRDA 红外设备
声音设备	AC97 声卡及喇叭
调试接口	JTAG 接口 UART 串口
供电系统	AC 200 – 220V 接口 大容量可充锂电池

2.3　软件系统

下表所示为电子书包的软件系统构成：

表 2 Ebook 软件系统

操作系统	ARMLinux Linux – 2.4.18 – rmk3 – pxa2
装载器	RedBoot
编译环境	GNU 交叉编译工具链 GNU gcc，ld，binutils，Glibc
调试环境	串口、网口 + gdb 远程调试
文件系统	Cramfs 以及 jffs2
GUI	MiniGUI、QT/Embedded

3 软件系统的实现

3.1 开发和调试环境

操作系统方面采用了 ARMLinux 内核，该内核为标准 Linux 内核专门面向 ARM 体系结构定制的。

对于启动装载器，本项目中我们在 RedBoot 的基础之上加以修改来满足本项目的需求。

3.1.1 开发和调试环境

我们使用 x86 架构的、装有 ReadHat 的 PC 作为开发宿主机，起作用主要用来编写源代码、交叉编译启动装载器、交叉编译 Linux 内核、生成文件系统、以及作为 gdb 调试主机等等。

在嵌入式系统的开发中，由于没有体系结构为目标体系结构（本项目中为 ARM）的拥有丰富运算资源和内存、存储资源的系统（例如 PC），通常使用交叉编译器来在 x86 平台上产生 ARM 体积结构的程序。本系统所使用的编译环境为 GNU GCC、ld、binutils、Glibc 交叉编译工具链。图 1 所示，为开发和调试环境示意图：

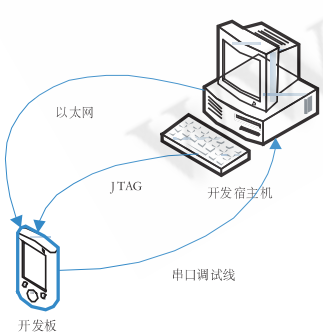

图 1 开发和调试环境

3.1.2 调试环境的改进

如下是典型的嵌入式系统开发步骤：

I. 将开发板与宿主机之间用 JTAG 调试线，串口线，以太网线连接。

II. 在宿主机开发启动装载器，Linux 内核，应用程序等，并交叉编。制作文件系统映像。

III. 使用 JTAG 将装载器烧录到目标板上，重新启动目标板，此时串口终端会出现装载器的提示符。

IV. 使用装载器给开发板上的 Flash 分区。

V. 使用装载器将交叉编译好的内核以及文件系统上传并烧录到开发板的 Flash 分区上。

VI. 使用装载器将 Linux 内核拷贝到内存中，并且把运行权交给 Linux 内核，同时将记录启动参数的地址传递给 Linux 内核（其中包括文件系统的分区以及根文件系统地址），Linux 内核最终加载根文件系统。

VII. 在宿主机启动 gdb 并设定远程调试口为串口，在开发板上启动 gdbserver，使用 gdbserver 加载被调试的程序。在宿主机上使用 gdb 调试目标板上的程序。

我们可以看到，对于 kernel 和文件系统的烧写是通过 JTAG 接口来进行的，其最高速率只有 115200b。而高端嵌入式设备所需要的内核大概为 1.2MB，包含有应用程序的文件系统大概需要 15 至 20MB 不等。由于调试期间对于内核和文件的变动非常频繁，使用 JTAG 进行反复烧写会影响调试的效率。由于开发版上支持网卡，在调试阶段，我们使用以太网络来实现 kernel 和文件系统的下载任务从而大大提高调试的效率。

具体方案为对启动装载器作如下修改：

I. 初始化网卡

II. 设置自身 IP 地址

III. 接受参数，从以太网络使用 tftp 协议下载数据并且写入 Flash 设备。

3.2 构建文件系统

我们把文件系统划分为根文件系统、数据文件系统以及临时文件系统。

根文件系统存储系统运行所必要的程序，函数库。例如，init 程序，libc 库，启动脚本和关键应用程序等。数据文件系统主要负责存储系统的可配置信息及用户所产生、下载的文件资料。该文件系统需要具备非易

失性。临时文件系统只提供一个临时的空间,其内容在系统掉电之后随即丢失。例如 /var /tmp 等。

3.2.1 根文件系统

首先,根文件系统需要是只读的以确保稳定性;其次,为节省空间,需要对根文件系统进行压缩。

我们选择 cramfs 作为根文件系统的文件系统类型,它是一个只读的压缩文件系统。Linux 内核首先在内存中将 cramfs 映像解压,然后创建只读的 ramdisk,即"内存磁盘"。

busybox

我们使用 busybox 来有效缩小文件系统的空间,它是一个集成了绝大部分系统常用命令的可执行文件。例如:cp,ls,find,等等。上述命令都是对 /bin/busybox 的符号链接。

3.2.2 数据文件系统

在选择数据文件系统的时候有以下几点考虑:

①手持设备在运行过程中随时有可能由于电源耗尽而关机。需要文件系统具有良好的容错功能。

②目前嵌入式设备上普遍使用 Flash 设备作为存储设备,这种设备的特点是每一次以块为单位进行数据的擦除和写入。而 Flash 的寿命取决于其中块的擦写次数。jffs2 文件系统是一个日志型、针对 Flash 设备特点进行了相应的优化,在本项目中我们使用其作为数据文件系统。

3.2.3 临时文件系统

我们使用 tempfs 作为临时文件系统。在启动脚本里面对其进行定义。

mount -t tempfs /tmp /tmp

mount -t tempfs /var /var

3.3 交叉编译 Linux 内核

①从 kernel.org 下载 2.4.18 标准内核源代码以及 ARMLinux 的补丁。

② 配置 Linux 内核

执行命令:make menuconfig 会出现当前 kernel 的配置菜单,注意选择 ARM Linux 作为目标平台。

③交叉编译 Linux 内核

为节省空间,我们选择生成压缩的 Linux 内核映像。所生成的内核映像头部是一个自解压程序,当启动装载器将内核映像拷贝到内存并把执行权交给内核映像后,其头部的自解压程序会先将内核映像自解压,然后从头执行。这样做的好处是节省 Flash 的存储空间,缺点是延长系统启动时间。

在命令行下,执行以下命令:

#make dep

#make clean

#make zImage

#make bzImage

#make modules

#make modules_install

#depmod -a

其中,make modules 及其后的命令用来生成内核模块。

3.4 为各设备开发驱动程序

ARM Linux 中提供了本项目中需要的如下设备的驱动程序,不过有一些需要额外的调试:

PCMCIA

IRDA

Touch Screen

IRDA

Modem

声卡

调试设备驱动是一件比较困难的事情,这里介绍一些调试技巧和注意事项:

①如果有支持内核调试的集成开发环境则最好,不过,这通常需要很多钱来购买。

②多使用 printk()。

③与硬件工程师一块工作,他们会从硬件的设计等提供很多帮助。

④如果条件允许,使用逻辑分析仪。这样可以清楚的看到总线上所传输的各种数据,加快设备相关驱动程序的调试速度。

还有一些是本项目特有的设备、或者在硬件设计上发生了改动,需要从头开发相应的驱动程序:LCD、MMC、按键、以及电源管理等。由于篇幅所限,本文的后半部分将着重讲述 LCD 驱动程序开发过程以及注意事项。

在本项目中,使用 Hitachi SX21V001 - Z4 DSTN 640x480 8 位色深的 LCD 显示器。

3.4.1 帧缓冲设备(FrameBuffer)

帧缓冲设备,是一个抽象的软件层,它提供了一套定义良好的用户接口,这些接口不随底层显示设备而改变。因此只要针对特定显示设备,实现其定义的各个接口函数,那么基于帧缓冲接口之上的各个应用程序,例如 console,GUI 程序等,就可以无缝的运行其上了。Linux 下可支持多个帧缓冲设备,最多可达 32 个,即从/dev/fb0 到/dev/fb31。通常情况下,缺省的帧缓冲设备为/dev/fb0。对于本项目,我们实现缺省的设备,即,/dev/fb0。

3.4.2 LCD 的种类

现在市面上流行的 LCD 可以分为两类,即 TFT 和 STN。当要驱动 TFT 时必须把 SA1110 的 LCD Controller 设为 Active Color Mode。而驱动 STN 时用 Passive Color Mode。TFT 和 STN 一个重要的区别是 STN 每个点实际上对应的是 3 个 bit(RGB),而 TFT 是对应的为 12 或 16 个 bit。

3.4.3 SA1110 的几个重要管脚

- LDD 7 :0

LCD 的数据线。对于单显来说,每个管脚对应的是一个象素点。对于 Passive Mode 来说每三个管脚代表一个相素点。

- GPIO 9 :2

当我们使用 Dual panel 模式或 16 bit TFT 模式时才会用到这些管脚,否则和其它的外围管脚没什么两样。我们需要在 GPDR 和 GPFR 设置这些管脚的用途。

- L_PCLK

表示的是 LCD 每显示一个相素点所需要的时钟。

- L_LCLK

表示每扫描一行所需的时钟。对于 TFT 来说,它就是水平扫描信号。

- L_FCLK

表示的是每扫描一屏所需的时钟。对于 TFT 来说,它就是垂直扫描信号。

- L_BIAS

这个信号对于 STN LCD 来说是防止 LCD 过分极化而设的,可以不用。对 TFT 来说有重要意义,只有这个信号有效是才可有效输出数据,它和相素时钟共同作用于 LCD 的数据输入。

3.4.4 帧缓冲驱动程序

下面主要介绍一下编写驱动可能遇到的技术难点和可能的解决方案。

本系统采用 Dual – panel mode 来驱动 LCD 屏幕。Dual – panel mode 就是把整个 LCD 分为上下两部分,使用相同的 LCD Controller Register,不同的是他们使用不同的 DMA 通道,对于比较大的 LCD 来说(例如,本项目所使用的 640x480),这样做的好处是可以成倍提高显示的刷新频率。

本项目的驱动程序由 a1100fb. c 改进而来。sa1100fb. c 里面为了使用者的方便对上述寄存器进行了封装,不用自己直接去填写上述寄存器的值,而是使用结构中给出的参数计算出来。例如

```
static struct sa1100fb_mach_info  {
  pixclock,
  bpp,
  xres,
  yres,
  hsync_len,
  vsync_len,
  left_margin,
  upper_margin,
  right_margin,
  lower_margin,
  sync,
  lccr0,
  lccr3,
  } ;
```

这个结构是针对标准的 Assabet 板的设置。我们仿造它给出自己的 LCD 结构参数,然后在 sa1100fb_get_machine_info 函数中的更改成自己定义的结构。下面讲解一下这几个参数的用法。

Pixclock 表示 LCD 每显示一个相素点所需要的时钟,直接对应 SA1110 的 L_PCLK,它的单位是 pico second。这个值计算的过程如下。我们以 Hitachi SX21V001 – Z4 为例,首先可以在 LCD 的 Data Sheet 当中得知 LCD 的扫描频率为 100HZ ~ 150HZ。我们选定为 120Hz。由于选用的是 Dual Panel 模式(LCCR0_Dual)。所以实际的行频率为 480/2。每秒扫描的点数:

Pixels = (xres * 3/8 + hsync_len + left_margin + right_margin) * (yres/2 + vsync_len) * VHZ

= (640 * 3/8 + 6 + 31 + 31) * (480/2 + 19) * 120

=9572640

pixclock = 1/Pixels * 1012 = 104464

bpp　表示的是每个像素点颜色深度。这里设的是 8bit。

Xres　屏幕的行宽：640

Yres　屏幕的列宽：480

hsync_len　水平信号宽度，可以从 Data Sheet 查到 CL1（对应 L_LCLK）信号宽度为 65ns，65ns/104464pico = 6

vsync_len　垂直信号宽度，根据 Data Sheet，得出宽度为 19

left_margin　左边距值 31

right_margin　右边距值 31

upper_margin　上边距值，对于 STN 来说为 0

lower_margin　下边距值，对于 STN 来说为 0

sync　行扫描频率和垂直扫描频率是否高电平有效：FB_SYNC_HOR_HIGH_ACT | FB_SYNC_VERT_HIGH_ACT

lccr0　LCD Control Register 0：

LCCR0_Color | LCCR0_Pas | LCCR0_Dual | LCCR0_DMADel(2)，

LCCR0_Pas　设 LCD Controller 为 Passive Mode

LCCR0_Color　设 LCD Controller 为 Color Mode

LCCR0_Dual　设 LCD Controller 为 Dual 模式

LCCR0_DMADel(2)　由于 DMA 在 Assabet 的优先级是最高的，所以当 LCD 在装入颜色映射表时，有可能造成 CPU 的 starving，所以有时候我们应该在连续两次的两次 DMA 操作间暂停一段时间，以便 CPU 取得总线。

lccr3　LCD Control Register 3：LCCR3_ACBsDiv(2)，对于 STN 的 LCD 来说，L_BIAS 信号没用所以应该取消 L_BIAS 信号。

3.5 GUI 的选择

本项目中分别使用 MiniGUI 和 QT 作为 GUI 开发出两套应用程序。应用程序的开发阶段选择先在 x86 桌面环境中进行调试，待程序稳定之后再加入文件系统，进行开发板环境的调试，这样可以缩短开发周期。

参考文献

1 Alessandro Rubini. Linux 设备驱动程序 第二版. Lisoleg. 北京，中国电力出版社，2002：150－180.

2 陈莉君. Linux 操作系统内核分析. 北京：人民邮电出版社 2000：80－100.

3 邹思轶. 嵌入式 Linux 设计与应用. 北京：清华大学出版社，2002：20－30.