基于 Linux Redhat 7.3 平台下嵌入式操作系统的设计与实现

汤燕娴[1]，张少锋[2]

（1.广州市地下铁道总公司，广东广州 510100；2.广东工业大学实验中心，广东广州 510643）

摘要：本文描述了嵌入式系统的构成元素以及 linux 系统的启动过程，介绍了构建嵌入式系统所需的软件包、工具、库以及配置文件。最后详细介绍了如何定制最简单的嵌入式 linux 系统。

关键词：嵌入式系统；内核；文件系统；脚本

中图分类号：TP316.2　　**文献标识码**：B　　**文章编号**：1009-9492 (2004) 07-0048-03

1 前言

嵌入式系统被定义为：以应用为中心、以计算机技术为基础、软件硬件可裁剪、适应应用系统对功能、可靠性、成本、体积、功耗严格要求的专用计算机系统，也称之为定制的系统[1]。目前嵌入式系统广泛应用于工业流水线控制、通讯、仪器仪表、汽车、船舶、航空航天、军事装备、消费类产品等领域。下面讲述如何构建简单地嵌入式系统。

2 嵌入式系统的构成元素

一个小型的嵌入式 linux 系统只需要下面三个基本元素[2]：

(1) 引导工具：开启计算机并导致其操作系统被加载的过程就是引导，引导工具可有几种选择，如 lilo、grub 等，本文采用的是 lilo 引导程序。

(2) linux 微内核：由内存管理、进程管理和事务管理构成，系统内核的主要目的是让机器配置的硬件能被检测到，并且所有的配置都能正常工作，顺利开机。

(3) 初始化进程：linux 被加载后，首先初始化硬件和设备驱动，然后运行 init。Init 启动其他进程，进入 shell 操作界面。

3 linux 启动过程

了解 linux 系统的基本启动过程是成功实现定制系统的关键[3]。Linux 启动过程如下：所有的 pc 机都是通过执行 ROM 中的代码加载启动盘的 0 柱面 0 扇区中的代码来启动整个系统。在 Linux 系统中启动盘的 0 柱面 0 扇区中含有的是启动装载器 LILO，它定位内核并装载它，最后执行它。一旦内核装载后，它先是进行基本设备初始化，接着试图加载并登陆磁盘中根文件系统，如果内核找不到可装载的根文件系统，启动过程就会停止。如果根文件系统装载完毕并登陆成功后，就会出现如下信息：

VFS：Mounted root（ext2 filesystem）readonly。

之后，系统发现 init 程序并执行它，init 程序寻找它的配置文件/etc/inittab，并开始执行其中的脚本（script），这些脚本是一些 SHELL 命令的组合，用来执行如下命令，如加载所需模块、装载 SWAP、初始化网络、装载 fstab 中列出的所有驱动器等。最后启动 getty 程序，getty 程序负责 console 和 ttys 之间的通信，它在显示器上打印 login 提示符并激活 login 程序，login 处理登陆的有效性并建立与用户的对话。至此，启动过程完毕。

4 定制前的准备工作

（1）机器的硬件有详细的了解，这主要是为构建系统微内核作准备[4]

（2）把 32M 的电子盘连在目标机器的 IDE 线上。这样在目标机器上，有两块硬盘。一块是电子盘（hdc1），一块是工作硬盘（hda2），用标准步骤在工作硬盘（hda2）上安装一个 Red Hat 7.3 系统，作为工作平台。

（3）下载相关的软件：

linux-2.4.20 .tar.bz2：一个稳定的内核（kernel）版本，当然也可以用其他的版本。

busybox-0.60.3.tar.gz：构建系统文件时的一个很好的工具，它编译出一个单个的可执行文件 busybox，提供许多其他常用命令行工具的功能，所有这些功能都合为一体。

uClibc-0.9.15.tar.gz：系统所需的 c 库，发行版 linux 使用 Glibc，但 Glibc 使用的内存数量较多，uClibc 是一个稳定的，具有高度兼容性的 Glibc 替代品，适合用于嵌入式 linux 系统。

lilo-22.3.2.tar.gz ：引导装入程序，即 boot loader。

5 定制过程

（1）构建嵌入式系统的内核

由于系统内核的作用，决定了编译内核前一定要对自

收稿日期：2004—03—03

己使用的机器硬件很熟悉，否则配置内核之后，内核无法检测到机器硬件而顺利开机 [5]。

①将内核源码包 linux-2.4.20.tar.bz2 拷贝到/usr/src 目录下并将指向以前内核版本的链接去掉：

```
#cp linux-2.4.20.tar.bz2  /usr/src
#cd /usr/src
#rm -f  linux
```

进行下一步前保存系统原来的 linux 目录，重命名即可。

②开压缩包，建立新链接，进入该目录：

```
#bzip2 -d linux-2.4.20.tar.bz2
#tar -xvf linux-2.4.20.tar
#ln -s linux-2.4.20 linux
```

③在 X Window 下编译，也可以用文本模式：

```
#make xconfig  (文本模式用命令 make menuconfig)
```

根据所要定制的系统的实际功能选择合适的选项后保存退出后。执行如下命令：

```
#make dep;make clean;make bzImage;make modules;make modules_install
```

执行完毕后在/usr/src/linux/arch/i386/boot 目录下产生了内核映象 bzImage，就是我们所要的内核

④测试内核

编译好内核后，要测试新内核是否可用。测试新内核前要保存旧内核，以防新内核无法启动而改用旧内核启动。保存完毕后把新内核和产生的 System.map 文件拷贝到/boot 目录下。然后修改 lilo.conf 文件，在 lilo.conf 文件末尾加入以下内容：

```
image=/boot/bzImage
label=new
read_only
root=/dev/hdc1  (根文件系统的分区)
```

保存退出，执行 lilo 命令。然后重启机器，用新内核引导。若引导失败，则重新编译内核。

(2) 构建 root 文件系统

现有 linux 发布和安装的标准文件系统类型是 ext2 文件系统。ext2 文件系统灵活、功能强大。但占用磁盘空间比 minix、romfs 文件系统大，由于笔者的电子盘有 32M，所以采用了 ext2 文件系统，若读者的磁盘容量有限，则可考虑使用 minix 或 romfs 文件系统。

在笔者定制的系统中，主要包括以下目录：

/dev：该目录包含了系统所需要的设备文件，MAKDEV 命令可以帮助用户建立这些文件。为了节约空间，在该目录下最好只放置所需要的文件。

/boot：该目录存放引导加载器 (Bootstrap Loader) 使用的文件，核心映像也放在这里。

/bin：该目录包含了引导启动所需的命令，这些命令都是二进制文件的可执行程序，是系统中重要的系统文件。

/etc：该目录存放各种系统配置文件。Linux 正是靠这些文件才得以正常的运行。

/lib：该目录是根文件系统上的程序所需的共享库，存放了根文件系统程序运行所需的共享文件。这些文件包含了可被许多程序共享的代码。

/mnt：通常该目录为空。但如果想要挂载另外的文件系统时，可以使用该目录。

/proc：该目录为空。内核将系统的状态文件放置到该目录下。

/sbin：该目录类似/bin，也用于存储二进制文件。大部分文件是系统管理员使用的基本的系统程序。

/usr：该目录包含了用户的应用程序。

下面开始构建 ext2 文件系统：

①在电子盘上建立 ext2 文件系统，以 root 用户身份执行以下操作：

```
#mke2fs -m 0 -N 2000 /dev/hdc1  (创建 ext2 文件系统)
#mount -t ext2 /dev/hdc1 /mnt/lfs  (把电子盘装载到 Red Hat 7.3 的/mnt/lfs 目录下)
```

这样我们就在 hdc1 盘上创建了一个标准的 ext2 文件系统

②为根文件系统建立一些标准的目录。执行以下命令：

```
#mkdir dev etc etc/rc.d proc mnt tmp var bin boot
#chmod 755 dev etc etc/rc.d mnt tmp var bin
#chmod 555 proc
```

进入/dev 目录下建立一般终端机设备：

```
#cd dev
#mknod tty c 5 0
#mkdir console c 5 1
#chmod 666 tty console
```

接着建立 VGA Display 虚拟终端机设备：

```
#mknod tty0 c 4 0
#chmod 666 tty0
```

建立 RAM disk 设备：

```
#mknod ram0 b1 0
#chmod 600 ram
```

建立 null 设备：

```
#mknod null c 1 3
#chmod 666 null
```

③构建 uClibc 库

uClibc 软件包构建两个相关组件。第一个组件是支持嵌入式系统的实用程序和应用程序的运行时库。第二个组件是一个开发环境。uClibc 开发环境使构建使用 uClibc 的实用程序和应用程序变得简单，甚至在本身不使用 uClibc 的系统上也是如此。uClibc 创建并安装用于 gcc 及相关工具的封装器。下面的步骤描述了如果构建 uClibc 软件包，采用这些步骤的前提是已将压缩文档解压缩到名为 uClibc-0.9.15 的目录中：

步骤 1：cd uClibc-0.9.15 (进入 uclibc 目录)

步骤2：ln - s ./extra/Configs/Config.i386 ./Co onfig（创建指向 Config 文件的链接）

步骤3：根据实际需要编辑 Config 文件

步骤4：make（构建、编译软件包）

步骤5：make PREFIX=/mnt/lfs install_target（安装开发环境）

(4) 构建 BusyBox

构建和安装 BusyBox 软件包可以简化构建 linux 系统的过程，BusyBox 是一个可执行文件，它提供许多其他常用命令行工具的功能，所有这些功能合为一体。在把压缩文档解压缩到名为 busybox-0.60.5 的前提下，执行如下步骤：

步骤1：编辑 Config.h 文件，根据个人需要对 busybox 进行配置

步骤2：编译安装

#make CC=/tmp/uClibc-0.9.15/extra/gcc-uClibc/i386-uClibc-gcc

#make PREFIX=/mnt/lfs install

⑤编写系统启动的配置文件。

Linux 系统在启动的过程中至少要用到三个配置文件：inittab、rc.sysinit、fstab，这三个配置文件的关系在 linux 启动过程中已描述过。下面编写系统启动所需的简单的启动脚本（配置文件）：

(i) 在/mnt/etc/rc.d 目录下编写 inittab 脚本，内容如下：

::sysinit:/etc/rc.d/rc.sysinit

::askfirst:/bin/sh

修改 inittab 的权限：

#chmod 644 inittab

(ii) 在/mnt/etc/rc.d 目录下编写 rc.sysinit，其内容如下：

#! /bin/sh

mount -a

变更其权限：

#chmod 755 rc.sysinit

(iii) 在/etc 目录下编写 fstab，其内容如下：

proc /proc proc defaults 0 0

变更其权限：

#chmod 644 fstab

(3) 整合并安装引导程序

把编译内核时所获得的 bzImage 和 system.map 文件拷贝新建目录/boot 下，并在/boot 目录下编辑 LILO 的配置文件 lilo.conf，内容如下：

boot=/dev/hdc1

install=/boot/boot.b

map=/boot/map

read-write

image=/boot/bzImage

label=new

root=/dev/hdc1

最后，按以下步骤安装 LILO：

步骤1：chroot /mnt/lfs/bin/sh

步骤2：/sbin/lilo

然后设置系统直接从硬盘驱动器引导，笔者定制的系统成功引导如下：

RAMDISK driver intialized:16 RAM disks of 4896k size 1024 blockize

Linux agpgart interface v0.99 (c) Jeff Hartmann

agpgart: Maximum main memory to use for agp memory: 28M

[drm] Initialized tdfx 1.0.0 20020216 on minor 0

[drm] Initialized radeon 1.1.1 20020405 on minor1

NET4:Linux TCP/IP 1.0 for NET4.0

IP Protocols:ICMP ,UDP, TCP, IGMP

IP: routing cache hash table of 512 buckets 4Kbytes

TCP: Hash tables configured (established 4096 bind 4096)

NET4:Unix domain sockets 1.0/SMP for Linux NET4.0.

RAMDISK:Compressed image found at block 0

VFS: Mounted root (ext2 filesystem)

Freeing unused kernel memory :108k freed

init start: BusyBox v0.60.3 (2003.08.18-19:50+0000) mlti-call binary

Bummer, could not run ´/etc/init.d/rcS´:No such file or directory

Please press Enter to activeate this console.

BusyBox v0.60.3 (2003.08.18-19:50+0000) Built-in shell (ash)

Enter ´help´ for a list of bullt-in commard.

5 结束语

与通用计算机不同，嵌入式系统是针对具体应用的专用系统。具有成本敏感性，软硬件都必须高效率地设计，量体裁衣去除冗余。目前的嵌入式产品主要分为信息电器、移动计算设备、网络设备、工控、仿真等几类。因此要在实现最简单嵌入式系统的前提下，根据需要进行各类功能的配置，并对系统进行优化。

参考文献：

[1] Richard Petersen. Linux:The Complete Reference, Fourth Edition [M]. 北京：机械工业出版社，2002.

[2] 龚永罡，宁宇鹏. Linux 系统管理 [M]. 北京：国防工业出版社，2000.9

[3] 赵敏哲. 64位 Linux 操作系统与应用实例 [M]. 北京：机械工业出版社，2000.

[4] 吴绍炜. Linux 系统管理 [M]. 北京：人民邮电出版社 2002.

第一作者简介：汤燕娴，女，1963年生，广州市花都区人，大学本科，工程师。研究领域：无线通信、程控交换机、SDH 和 OTN 传输和计算机及计算机通信。

（编辑：滕召旗）