

Linux 操作系统的安装以及配置.....	3
1 如何安装 RedHat9.0	3
2 在 RedHat 中添加新用户	16
3 配置 PC 机 Linux 的 ftp 服务	16
4 配置 PC 机 Linux 的 telnet	17
5 建立交叉编译环境	17
6 编译内核.....	17
Linux 的移植	19
1 Bootloader 的移植	19
1.1 vivi 的配置与编译	19
1.2 配置和编译 vivi	20
1.3 vivi 代码分析.....	21
1.4 vivi 的运行	21
1.5 启动代码执行流程图.....	45
1.6 vivi 的配置文件	45
2 Linux 内核移植.....	45
2.1 Linux 内核移植要点	45
2.2 平台相关代码结构.....	47
2.3 建立目标平台工程框架	47
2.4 建立目标平台代码框架	51
2.5 构建目标板代码.....	54
3 linux 2.6 内核配置选项详细说明	59
3.1 Code maturity level options --->	59
3.2 General setup --->.....	59
3.3 module support--->	60
3.4 Block layer --->	60
3.5 Processor type and features--->	61
3.6 Power management options (ACPI, APM)--->.....	63
3.7 Executable file formats--->	68
3.8 Networking--->.....	68
3.9 Device Drivers--->	70
4 如何建立 yaffs 文件系统映像	72
Linux 外设驱动程序以及用户程序的编写	72
1、Helloworld.....	72
2、编写第一个驱动	72
3、GPIO 驱动.....	73
3.1、硬件分析	74
3.2 LED 驱动的编写	74
3.3 在内核源码中添加对 LED 驱动的支持.....	77
附录一 Linux 命令详解	78

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

文件列表 - ls.....	78
目录切换 - cd.....	78
删除 - rm.....	78
复制 - cp.....	79
移动 - mv.....	79
比较 - diff.....	79
回显 - echo.....	79
文件内容查看 - cat.....	79
时间日期 - date.....	80
容量查看 - du.....	80
查找 - find.....	80
搜索 - grep.....	80
编辑 - vi.....	80
读取 - man.....	81
重启 - reboot.....	81
关机 - halt.....	81
压缩与解压 - tar.....	81
权限设置 - chmod.....	81
网卡配置 - ifconfig.....	81
创建设备 - mknod.....	82
装载模块 - insmod.....	82
删除模块 - rmmod.....	82
挂接 - mount.....	82
卸载 - umount.....	83
关于我们.....	83
联系方式.....	84

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

Linux 操作系统的安装以及配置

本篇中要求的开发环境为 linux 操作系统，建议使用 Redhat9.0。

Linux 操作系统的主要特点：

1、设备独立性：是指操作系统把所有外部设备统一当作成文件来看待，只要安装它们的驱动程序，任何用户都可以象使用文件一样，操纵、使用这些设备，而不必知道它们的具体存在形式。Linux 是具有设备独立性的操作系统，它的内核具有高度适应能力。

2、丰富的网络功能：完善的内置网络是 Linux 的一大特点

3、可靠的安全系统：Linux 采取了许多安全技术措施，包括对读、写控制、带保护的子系统、审计跟踪、核心授权等，这为网络多用户环境中的用户提供了必要的安全保障。

4、良好的可移植性：是指将操作系统从一个平台转移到另一个平台使它仍然能按其自身的方式运行的能力。Linux 是一种可移植的操作系统，能够在从微型计算机到大型计算机的任何环境中和任何平台上运行。

1 如何安装 RedHat9.0

最简单，最方便的安装方法当然是从 CD 安装，你可以享受最人性化的，类似于 Windows 的安装。你只要将计算机设置成光驱引导，把安装 CD1 放入光驱，重新引导系统，在安装界面中直接按回车，即进入如图 1 形化安装界面：



图 1

由图可见，在提供“豪华”的图形化 GUI 安装界面的同时，RedHat Linux

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

9.0 仍然保留了以往版本中的字符模式安装界面,这对于追求安装速度与效率的用户一直是很有吸引力的。因为许多用户是将 RedHat9 安装成 服务器 来使用的,不需要 X-Window 以及 GUI 安装界面。

RedHat9 的安装步骤中比以往多了一个环节,那就是对安装光盘介质的检测。它允许在开安装过程前对安装光盘介质进行内容校验,以防止在安装的中途由于光盘无法读取或是内容误造成意外的安装中断,导致前功尽弃。如图 2 所示:

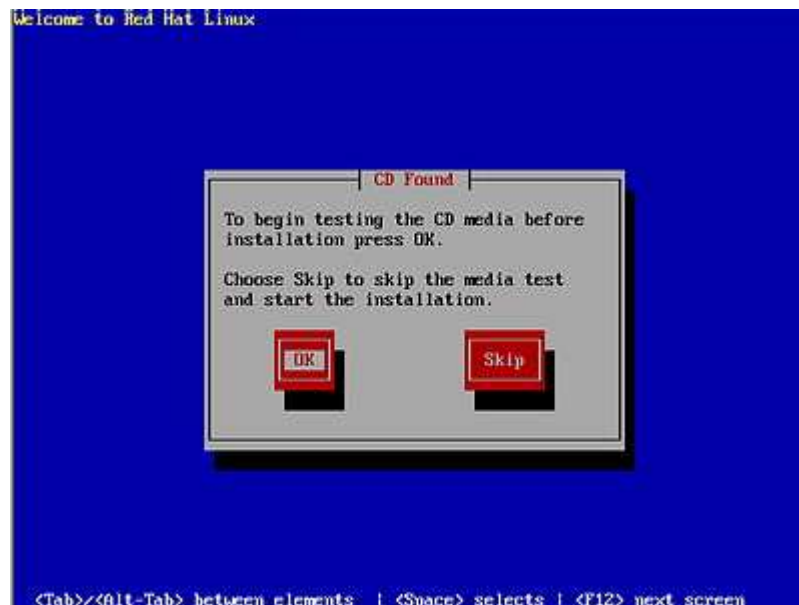


图 2

上图中提示是否测试安装 CD 的内容的完整性,选“OK”开始测试安装 CD;选“Skip”不测试安装 CD 开始出现图 7 所示,如果是第一次安装当然要测试安装 CD,选“OK”后回车,出现如下图 3:



杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

图 3

选“Test”测试安装光盘的文件，选“Eject CD”测试光盘以外的安装文件，这里选择“Test”后或侧，出现如下图 4：

正在测试第一张安装 CD，测试完后显示如下图 5 所示，看到图中最后一行英文“It is Ok to install from this media.”说明这张安装 CD 是 OK 的，按“Enter”键回车后显示如下图 6 所示：

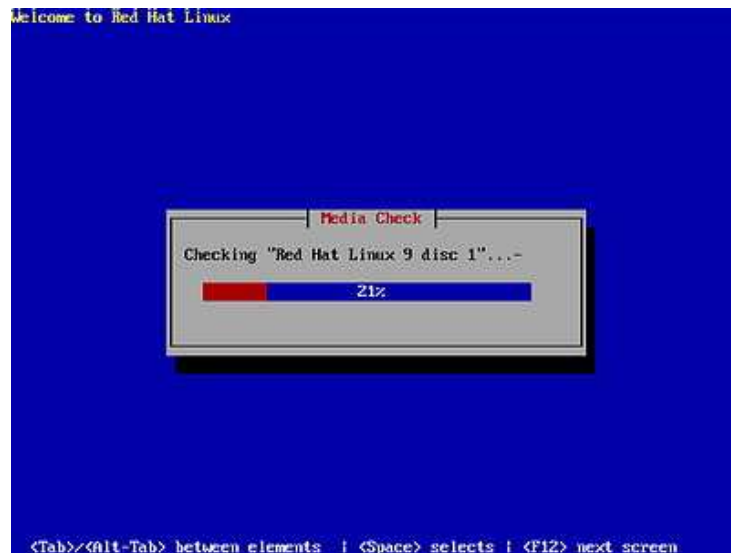


图 4



图 5

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



图 6



图 7

选择“Continue”并按回车开始安装。

1) 语言选择

RedHat 支持世界上几乎所有国家的语言，这里只要选中 简体中文，并将系统默认语言选择简体中文如图 7，那么在安装过程结束，系统启动后，整个操作系统的界面都将是简体中文的，用户不用做任何额外的中文化操作和设置。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



图 8

2) 键盘的配置

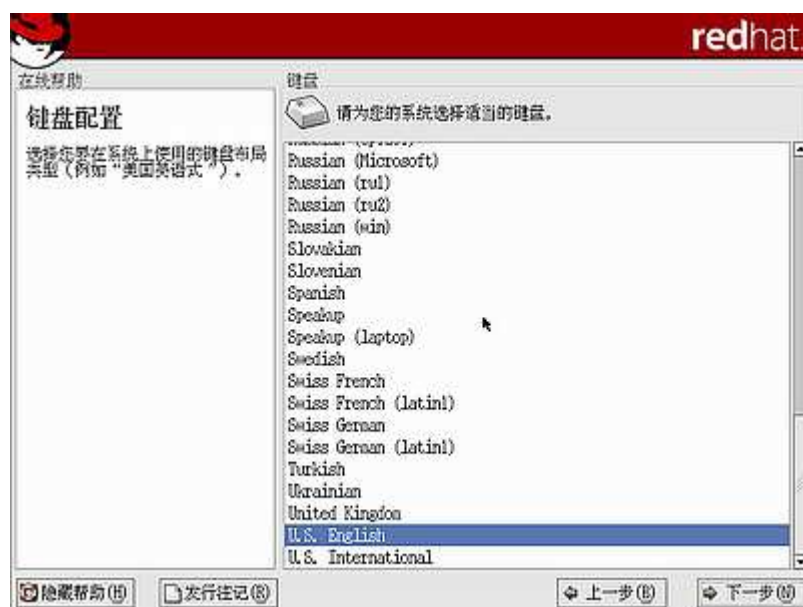


图 9

3) 鼠标的配置

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.netEmail: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

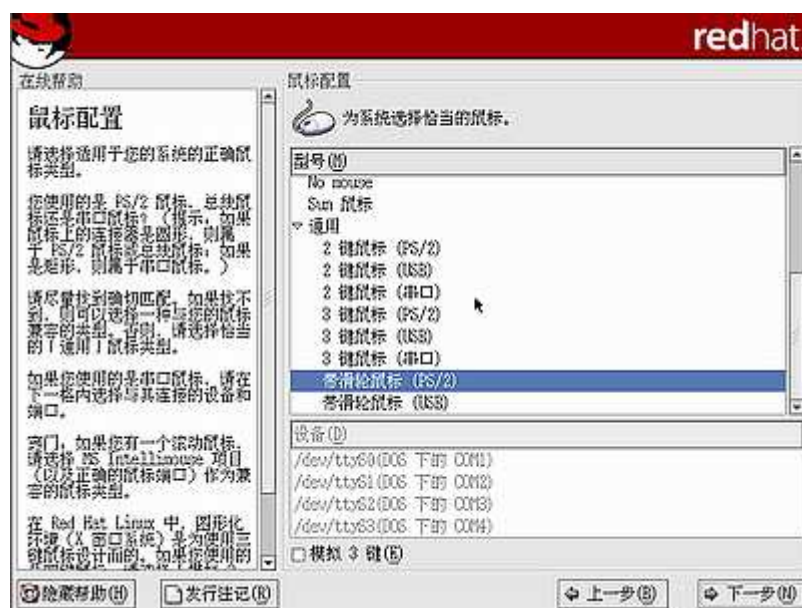


图 10

4) 选择安装类型

安装类型我们选择“个人桌面”。



图 11

5) 磁盘分区设置

这也许是整个安装过程中惟一需要用户较多干预的步骤，REDHAT Linux 9.0 提供了两种分区方式——自动分区和使用 DISK DRUID 程序进行手动分区。如图 12 所示：

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



图 12

(1) 自动分区：如果是全新的计算机，上面没有任何操作系统，建议使用“自动分区”功能，它会自动根据磁盘以及内存的大小，分配磁盘空间和 SWAP 空间。

这是一个“危险”的功能，因为它会自动删除原先硬盘上的数据并格式化成为 Linux 的分区文件系统（EXT3、REISERFS 等），所以除非计算机上没有任何其他操作系统或是没有任何需要保留的数据，你才可以使用“自动分区”功能。

(2) 手动分区：如果硬盘上有其他操作系统或是需要保留其他分区上的数据，建议采用 DISKDRUID 程序进行手动分区。DISK DRUID 是一个 GUI 的分区程序，它可以对磁盘的分区进行方便的删除、添加和修改属性等操作，它比以前版本中使用的字符界面 Fdisk 程序的界面更加友好，操作更加直观。下面我们来看看如何使用 DISK DRUID 程序对硬盘进行分区。

因为 Linux 操作系统需要有自己的文件系统分区，而且 Linux 的分区和微软 Windows 的分区不同，不能共用，所以，需要为 Linux 单独开辟一个（或若干个）分区。Linux 一般可以采用 EXT3 分区，这也是 REDHAT Linux 9.0 默认采用的文件系统。

6) 为系统分区

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



为 Linux 建立文件分区可以有两种办法，一种是利用空闲的磁盘空间新建一个 Linux 分区，另一种是编辑一个现有的分区，使它成为 Linux 分区。如果没有空闲的磁盘空间，就需要将现有的分区删除后，腾出空间，以建立 Linux 分区。

DISK DRUID 程序中有明显的新建、删除、编辑、重设等按钮。用户可以直观地对磁盘进行操作。在使用 DISK DRUID 对磁盘分区进行操作时，有四个重要的参数需要仔细设定：它们是挂载点、文件系统类型、驱动器、分区大小。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



挂载点：它指定了该分区对应 Linux 文件系统的哪个目录，Linux 允许将不同的物理磁盘上的分区映射到不同的目录，这样可以实现将不同的服务程序放在不同的物理磁盘上，当其中一个物理磁盘损坏时不会影响到其他物理磁盘上的数据。

文件系统类型：它指定了该分区的文件系统类型，可选项有 EXT2 、 EXT3 、 REISERFS 、 JFS 、 SWAP 等。Linux 的数据分区创建完毕后，有必要创建一个 SWAP 分区，它实际上是用硬盘模拟的虚拟内存，当系统内存使用率比较高的时候，内核会自动使用 SWAP 分区来模拟内存。

大小：指分区的大小（以 MB 为单位），Linux 数据分区的大小可以根据用户的实际情况进行填写，而 SWAP 大小根据经验可以设为物理内存的两倍，但是当物理内存大于 1GB 时，SWAP 分区可以设置为 2GB。

允许的驱动器：如果计算机上有多个物理磁盘，就可以在这个菜单选项中选中需要进行分区操作的物理磁盘。

经过磁盘分区操作，安装过程中相对最复杂的一个步骤已经过去，接下来的安装将是一马平川。

7) 网络配置

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



8) 防火墙配置



杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.netEmail: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

9) 设置根口令

即 root 管理员密码，root 账号在系统中具有最高权限，平时登入系统一般不用该账号。



10) 选择软件包组（这步要小心）

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



11) 安装软件包 （需要一定的时间）



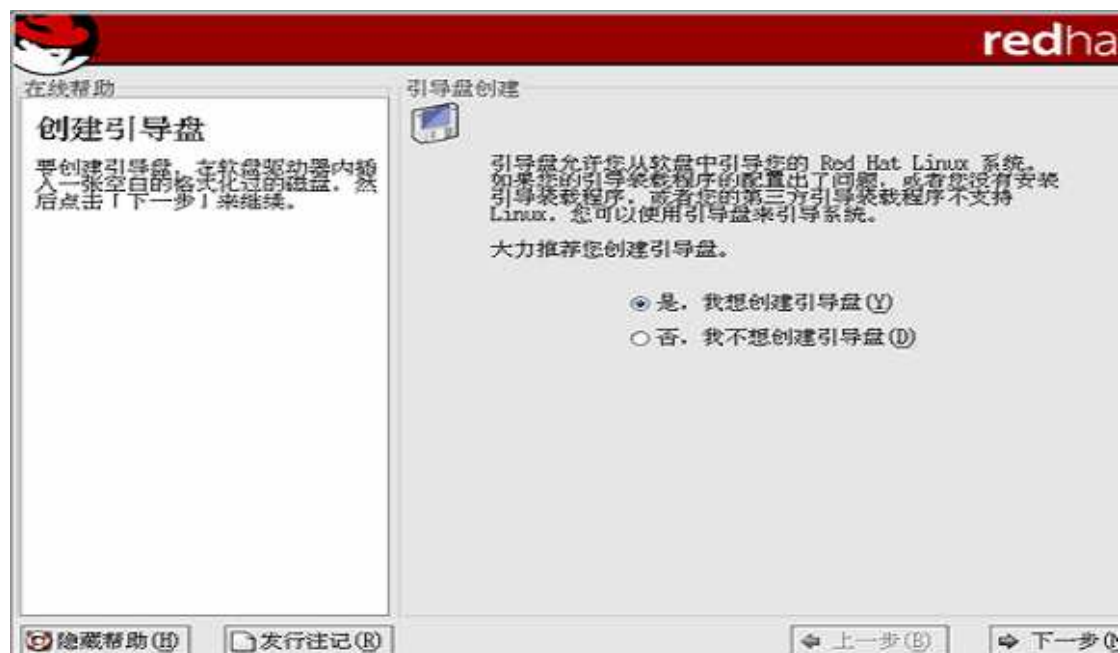
杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.netEmail: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

12) 创建引导盘



这样系统安装完成, 取出光盘后, 重启后将首次出现选择菜单如下图所示



杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

2 在 RedHat 中添加新用户

Linux 是一个多用户和多任务的操作系统。用户要登录 Linux 首先要有个合法的账号，所以必须创建平常使用的用户账号。

现在我们以 root(管理员账号) 身份登录 RH9 系统进行账户管理为例说明：在主菜单中选择“系统设置→用户和组群”，弹出“Redhat 用户管理器”对话框，点击工具栏的“添加用户”按钮，弹出“创建新用户”对话框 如下图，在“用户名”栏中输入用户账号名，然后在“口令”和“确认口令”栏中输入账号密码，其他的设置可以使用系统的默认设置，一般不需要手工更改。最后点击“确定”按钮。



除了以上方式，还可以在终端命令行窗口中创建用户账号：点击 GNOME 面板中的终端命令行窗口快捷工具按钮，在“#”提示符下输入“adduser tjrao1027 -p 123wefg”命令，其中“tjrao1027”为账号名，“123wefg”为账号密码。

提示：在 Linux 中，如果你以 root 身份登录系统，则可以对系统进行任何更改，包括对内核的重新编译，这样很容易造成对系统的损坏。所以在一般情况下，我们要创建另外的用户来进行日常使用，root 用户只在系统维护、故障排除等情况下使用。

3 配置 PC 机 Linux 的 ftp 服务

在 PC 机命令行输入

```
# redhat-config-services
```

打开系统服务配置窗口，在左侧一栏找到 vsftpd 服务选项框，并选中它，然后点 File→Save Changes 保存设置。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

4 配置 PC 机 Linux 的 telnet

开发板上电进入 linux 操作系统并进入到命令行，在命令行输入

```
# redhat-config-services
```

打开系统服务配置窗口，在左侧一栏找到 telnet 服务选项框，并选中它，然后点击 File->Save Changes 保存设置，下次开机就启动了 telnet 服务。

5 建立交叉编译环境

什么是交叉编译呢？

在一种计算机环境中运行的编译程序，能编译出在另外一种环境下运行的代码，我们就称这种编译器支持交叉编译。这个编译过程就叫交叉编译。简单地说，就是在一个平台上生成另一个平台上的可执行代码。如 keil 软件，在 keil 上编译，但在单片机上运行，典型的交叉编译。

编译嵌入式 linux 内核前，要先安装交叉编译工具 arm-linux-gcc，随板光盘已附带编译工具。

编译工具 arm-linux-gcc-3.4.1.tar.bz2，安装此编译器只需要在 /usr/local 目录下建一个 arm 的目录，先将光盘 arm-linux-gcc-3.4.1.tar.bz2 拷贝到某个目录下，然后进入该目录，执行解包命令 tar xjvf cross-3.4.1.tar.bz2 即可，之后可编辑 /etc/bashrc 文件，在最后增加路径 export PATH=/usr/local/arm/3.4.1/bin:\$PATH，以后编译内核或其他应用程序均可用 arm-linux- 来指定交叉编译器。可以使用 echo \$PATH 来查看环境变量中是否有 /usr/local/arm/3.4.1/bin 路径变量。

6 编译内核

1、解压内核包

将光盘目录下的 linux 内核包解压到某一目录下，然后进行解压，命令：

```
tar -xjvf linux-2.6.12.tar.bz2
```

注意：2.6.12 为 linux 内核版本号。

2、编辑 Makefile 文件

进入解压的目录后，运行命令：vi Makefile 找到 “CROSS_COMPILE=” 这行，将它改为 CROSS_COMPILE=/usr/local/arm/3.4.1/bin/arm-linux-（或者你的系统所安装的 ARM-Linux-

gcc 的编译器路径）。设置好后保存退出。这一行是指明所用交叉编译器的版本，位置。

修改 “ARCH :=” 为 “ARCH :=arm” 指定处理器。

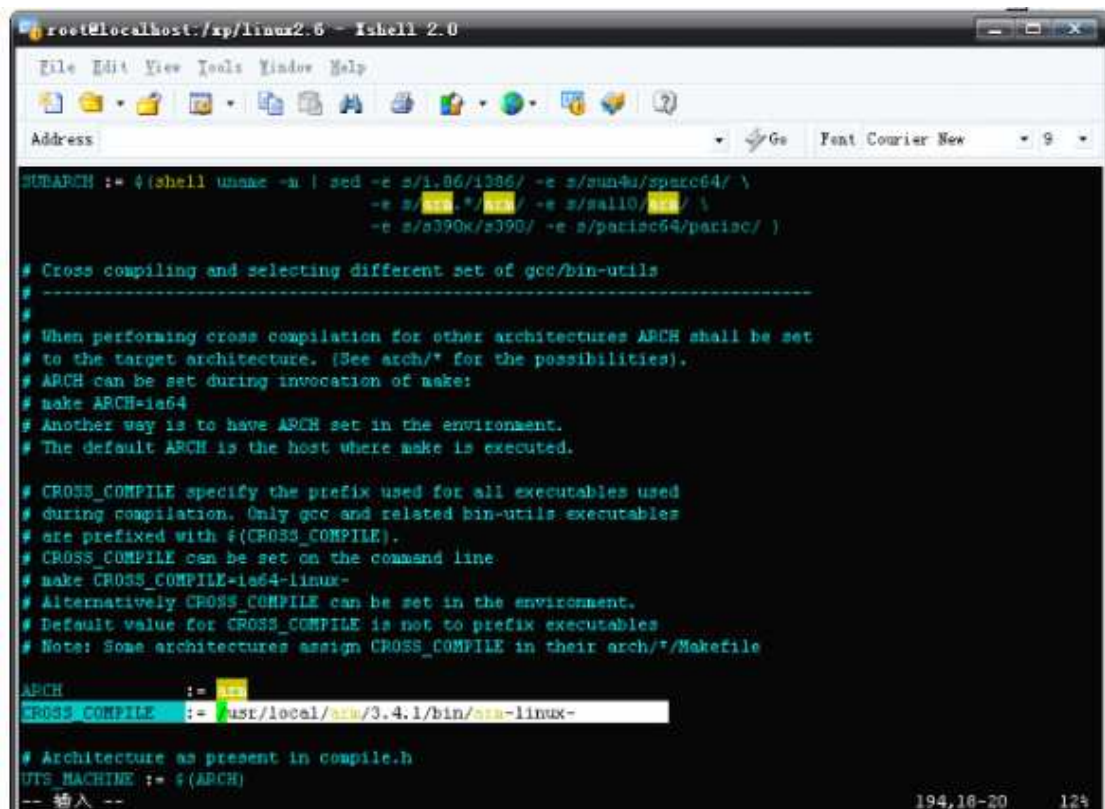
杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号



```
root@localhost: /sp/linux2.6 - Xshell 2.0
File Edit View Tools Window Help
Address
SUBARCH := $(shell uname -m | sed -e s/i.86/i386/ -e s/sun4u/sparc64/ \
-e s/ia.*/* -e s/s390/* -e s/parisc64/parisc/ )

# Cross compiling and selecting different set of gcc/bin-utils
#
# When performing cross compilation for other architectures ARCH shall be set
# to the target architecture. (See arch/* for the possibilities).
# ARCH can be set during invocation of make:
# make ARCH=ia64
# Another way is to have ARCH set in the environment.
# The default ARCH is the host where make is executed.
#
# CROSS_COMPILE specify the prefix used for all executables used
# during compilation. Only gcc and related bin-utils executables
# are prefixed with $(CROSS_COMPILE).
# CROSS_COMPILE can be set on the command line
# make CROSS_COMPILE=ia64-linux-
# Alternatively CROSS_COMPILE can be set in the environment.
# Default value for CROSS_COMPILE is not to prefix executables
# Note: Some architectures assign CROSS_COMPILE in their arch/*/Makefile

ARCH := arm
CROSS_COMPILE := /usr/local/arm/3.4.1/bin/arm-linux-

# Architecture as present in compile.h
MACHINE := $(ARCH)
-- 输入 --
194,16-20 124
```

3、配置内核

make config （基于文本的最为传统的配置界面，不推荐使用）

make menuconfig （基于文本选单的配置界面，字符终端下推荐使用）

make xconfig （基于图形窗口模式的配置界面，Xwindow 下推荐使用）

make oldconfig （如果只想在原来内核配置的基础上修改一些小地方，会省去不少麻烦）

这三个命令中， make xconfig 的界面最为友好，但如果你不能使用 Xwindow，那么就使用 make menuconfig 好了。界面虽然比上面一个差点，总比 make config 的要好多了。

在这里我们输入命令： # make menuconfig

下图为 make menuconfig 的界面：

选择相应的配置时，有三种选择，它们分别代表的含义如下：

Y — 将该功能编译进内核

N — 不将该功能编译进内核

M — 将该功能编译成可以在需要时动态插入到内核中的模块
需要使用空格键进行选取。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

Linux 的移植

1 Bootloader 的移植

Bootloader(Vivi)源代码分析

Bootloader(Vivi)源代码分析 CSDN Blog 推出文章指数概念，文章指数是对 Blog 文章综合评分后推算出的，综合评分项分别是该文章的点击量，回复次数，被网摘收录数量，文章长度和文章类型；满分 100，每月更新一次。

Vivi 是韩国 mizi 公司开发的 bootloader，适用于 ARM9 处理器。Vivi 有两种工作模式：启动加载模式和下载模式。启动加载模式可以在一段时间后（这个时间可更改）自行启动 linux 内核，这时 vivi 的默认模式。在下载模式下，vivi 为用户提供一个命令行接口，通过接口可以使用 vivi 提供的一些命令，见下表：

命令	功能
Load	把二进制文件载入 Flash 或 RAM
Part	操作 MTD 分区信息。显示、增加、删除、复位、保存 MTD 分区
Param	设置参数
Boot	启动系统
Flash	管理 Flash，如删除 Flash 的数据

1.1 vivi 的配置与编译

建立交叉开发环境, 宿主机安装标准 linux 操作系统: RedHat9, 建立交叉编译环境 arm-linux-gcc-2.95.3。

先以 root 用户的身份登陆到 linux 下。进入/usr/local 目录, 创建名为 arm 的目录:

```
cd /usr/local
```

```
mkdir arm
```

将光盘提供的 cross-2.95.3.tar.bz2 解压到/usr/local/arm 目录:

```
tar jxvf cross-2.95.3.tar.bz2 -C /usr/local/arm
```

然后修改修改 PATH 变量: 为了可以方便使用 arm-linux-gcc 编译器系统, 把 arm-linux 工具链目录加入到环境变量 PATH 中:

修改/etc/profile 文件, 添加 pathmunge /usr/local/arm/2.95.3/bin 即可。

```
# Path manipulation
```

```
if [ `id -u` = 0 ]; then
```

```
pathmunge /sbin
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
pathmunge /usr/sbin
```

```
pathmunge /usr/local/sbin
```

```
pathmunge /usr/local/arm/2.95.3/bin
```

设置环境变量后，最好是重启或注销一下，这样设置的环境变量才能生效

1.2 配置和编译 vivi

如果 vivi 的源代码已根据开发板作了相应改动，则需要对源代码进行配置和编译，以生成烧入 flash 的 vivi 二进制映像文件。

由于 vivi 要用到 kernel 的一些头文件，所以需要 kernel 的源代码，所以先要把 linux 的 kernel 准备好。将 vivi 和 kernel 都解到相应目录下（例如我将光盘提供的 vivi 源代码解压到 /home/chenjun 目录下，光盘提供的 Linux kernel 源码 kernel-h2410eb.041024.tar.gz 也解压到 /home/chenjun 目录下，解压后的文件名为 kerne-h2410eb）。

然后需修改 /vivi/Makefile 里的一些变量设置：

```
Ø LINUX_INCLUDE_DIR = /kernel/include/
```

（LINUX_INCLUDE_DIR 为 kernel/include 的对应目录，我的是 /home/chen/kerne-h2410eb/include/）

因此修改为：

```
LINUX_INCLUDE_DIR = /home/chenjun/ kerne-h2410eb/include/
```

```
Ø CROSS_COMPILE = /usr/local/arm/2.95.3/bin/arm-linux-
```

（CROSS_COMPILE 为 arm-linux 安装的相应目录，我的是 /usr/local/arm/2.95.3/bin/arm-linux-）

因此修改为：

```
CROSS_COMPILE = /usr/local/arm/2.95.3/bin/arm-linux-
```

```
Ø ARM_GCC_LIBS = /usr/local/arm/2.95.3/lib/gcc-lib/arm-linux/2.95.3
```

（需根据你 arm-linux 的安装目录修改，我的是

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

/usr/local/arm/2.95.3/lib/gcc-lib/arm-linux/2.95.3)

进入/vivi 目录执行 make distclean。(目的是确保编译的有效性,在编译之前将 vivi 里所有的 “*.o” 和 “*.o.flag” 文件删掉)

进入/vivi 目录里,输入 “make menuconfig”,开始选择配置。可以 Load 一个写好的配置文件也可以自己修改试试。注意 Exit 时一定要选 “Yes” 保存配置。

再输入 “make” 正式开始编译,一会儿就完了。如果不报错,在/vivi 里面就有你自己的 “vivi” 了。这个就是后面要烧写到 flash 中的 bootloader。

1.3 vivi 代码分析

vivi 的代码包括 arch, init, lib, drivers 和 include 等几个目录,共 200 多条文件。Vivi 主要包括下面几个目录:

arch: 此目录包括了所有 vivi 支持的目标板的子目录,例如 s3c2410 目录。

drivers: 其中包括了引导内核需要的设备的驱动程序 (MTD 和串口)。MTD 目录下分 map、nand 和 nor 三个目录。

init: 这个目录只有 main.c 和 version.c 两个文件。和普通的 C 程序一样, vivi 将从 main 函数开始执行。

lib: 一些平台公共的接口代码,比如 time.c 里的 udelay() 和 mdelay()。

include: 头文件的公共目录,其中的 s3c2410.h 定义了这块处理器的一些寄存器。Platform/smdk2410.h 定义了与开发板相关的资源配置参数,我们往往只需要修改这个文件就可以配置目标板的参数,如波特率、引导参数、物理内存映射等。

1.4 vivi 的运行

vivi 的运行也可以分为两个阶段:

1.4.1 vivi 的第一阶段

完成含依赖于 CPU 的体系结构硬件初始化的代码,包括禁止中断、初始化串口、复制自身到 RAM 等。相关代码集中在 head.S(\vivi\arch\s3c2410 目录下):

```
Head.S:
#include "config.h"
#include "linkage.h"
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
#include "machine.h"

@ Start of executable code

ENTRY(_start)

ENTRY(ResetEntryPoint)
@

@ Exception vector table (physical address = 0x00000000) ; 异常向量表物理地址

@

@ 0x00: Reset ; 复位

b Reset

@ 0x04: Undefined instruction exception ; 未定义的指令异常

UndefEntryPoint:

b HandleUndef

@ 0x08: Software interrupt exception ; 软件中断异常

SWIEntryPoint:

b HandleSWI

@ 0x0c: Prefetch Abort (Instruction Fetch Memory Abort) ; 内存操作异常

PrefetchAbortEntryPoint:

b HandlePrefetchAbort

@ 0x10: Data Access Memory Abort ; 数据异常

DataAbortEntryPoint:

b HandleDataAbort
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

@ 0x14: Not used ; 未使用

NotUsedEntryPoint:

b HandleNotUsed

@ 0x18: IRQ(Interrupt Request) exception ; 慢速中断处理

IRQEntryPoint:

b HandleIRQ

@ 0x1c: FIQ(Fast Interrupt Request) exception ; 快速中断处理

FIQEntryPoint:

b HandleFIQ

@

@ VIVI magics

@

@ 0x20: magic number so we can verify that we only put

.long 0

@ 0x24:

.long 0

@ 0x28: where this vivi was linked, so we can put it in memory in the right place

.long _start

@ 0x2C: this contains the platform, cpu and machine id

.long ARCHITECTURE_MAGIC

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
@ 0x30: vivi capabilities

.long 0

#ifdef CONFIG_PM ; vivi 考虑不需要使用电源管理

@ 0x34:

b SleepRamProc

#endif

#ifdef CONFIG_TEST

@ 0x38:

b hmi

#endif

@

@ Start VIVI head

@

Reset:

@ disable watch dog timer ; 禁止看门狗计时器

mov r1, #0x53000000 ; WTCN 寄存器地址是

0x53000000, 清 0

mov r2, #0x0

str r2, [r1]

#ifdef CONFIG_S3C2410_MPORT3 ; 不符合条件，跳到下面的关中断
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
/**** 在/vivi/include/autoconf.h 中#undef CONFIG_S3C2410_MPORT3*****/
```

```
mov r1, #0x56000000 ; GPACON 寄存器地址是
```

```
0x56000000
```

```
mov r2, #0x00000005
```

```
str r2, [r1, #0x70] ; 配置 GPHCON 寄存器
```

```
mov r2, #0x00000001
```

```
str r2, [r1, #0x78] ; 配置 GPHUP 寄存器
```

```
mov r2, #0x00000001
```

```
str r2, [r1, #0x74] ; 配置 GPHDAT 寄存器
```

```
#endif
```

```
@ disable all interrupts ; 禁止全部中断
```

```
mov r1, #INT_CTL_BASE
```

```
mov r2, #0xffffffff
```

```
str r2, [r1, #oINTMSK] ; 掩码关闭所有中断
```

```
ldr r2, =0x7ff
```

```
str r2, [r1, #oINTSUBMSK]
```

```
@ initialise system clocks ; 初始化系统时钟
```

```
mov r1, #CLK_CTL_BASE
```

```
mvn r2, #0xff000000
```

```
str r2, [r1, #oLOCKTIME]
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
@ldr r2, mpll_50mhz

@str r2, [r1, #oMPLLCON]

#ifdef CONFIG_S3C2410_MPORT1 ; 满足条件，向下执行

/**** 在/vivi/include/autoconf.h 中#undef CONFIG_S3C2410_MPORT1*****/

@ 1:2:4

mov r1, #CLK_CTL_BASE

mov r2, #0x3

str r2, [r1, #oCLKDIVN]


mrc p15, 0, r1, c1, c0, 0 @ read ctrl register

orr r1, r1, #0xc0000000 @ Asynchronous

mcr p15, 0, r1, c1, c0, 0 @ write ctrl register

@ now, CPU clock is 200 Mhz ; CPU 的频率是 200MHz

mov r1, #CLK_CTL_BASE

ldr r2, mpll_200mhz

str r2, [r1, #oMPLLCON]

#else

@ 1:2:2

mov r1, #CLK_CTL_BASE

ldr r2, clock_clkdivn
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
str r2, [r1, #oCLKDIVN]

mrc p15, 0, r1, c1, c0, 0 @ read ctrl register

orr r1, r1, #0xc0000000 @ Asynchronous

mcr p15, 0, r1, c1, c0, 0 @ write ctrl register

@ now, CPU clock is 100 Mhz ; CPU 的频率是 100MHz

mov r1, #CLK_CTL_BASE

ldr r2, mp11_100mhz

str r2, [r1, #oMPLLCON]

#endif

bl memsetup ; 跳转到 memsetup 函数
```

```
/******
```

Memsetup 函数的实现:

```
ENTRY(memsetup)
```

```
@ initialise the static memory
```

```
@ set memory control registers ; 设置内存控制寄存器的初值
```

```
mov r1, #MEM_CTL_BASE
```

```
adrl r2, mem_cfg_val
```

```
/******
```

```
@
```

```
@ Data Area
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

@

@ Memory configuration values

.align 4

mem_cfg_val: ; 定义好的 13*4=52 个字节初值

.long vBWSCON ; 在/vivi/include/platform/smdk2410.h 中赋值

/***** SDRAM 从 32 位变成 16 位，需要修改 vBWSCON 的值 *****/

.long vBANKCON0

.long vBANKCON1

.long vBANKCON2

.long vBANKCON3

/***** 网卡控制器 vBANKCON3 的值可能需要修改 *****/

.long vBANKCON4

.long vBANKCON5

.long vBANKCON6

/***** SDRAM 从 32 位变成 16 位，可能需要修改 vBANKCON6 的值 *****/

.long vBANKCON7

.long vREFRESH

.long vBANKSIZE

/***** SDRAM 从 64MB 变成 32MB，需要修改 vBANKSIZE 的值 *****/

.long vMRSRB6

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
.long vMRSRB7
```

```
*****/
```

```
add r3, r1, #52
```

```
1: ldr r4, [r2], #4
```

```
str r4, [r1], #4
```

```
cmp r1, r3
```

```
bne 1b ; 循环操作，直到 13 个寄存器赋值完成
```

```
mov pc, lr
```

```
*****/
```

```
#ifdef CONFIG_PM ; vivi 考虑不需要使用电源管理
```

```
@ Check if this is a wake-up from sleep
```

```
ldr r1, PMST_ADDR
```

```
ldr r0, [r1]
```

```
tst r0, #(PMST_SMR)
```

```
bne WakeupStart ; 查看状态，判断是否需要跳转到 WakeupStart
```

```
#endif
```

```
#ifdef CONFIG_S3C2410_SMDK ; SMDK 开发板使用
```

```
@ All LED on ; 点亮开发板上的 LED
```

```
mov r1, #GPIO_CTL_BASE
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号


```
add r1, r1, #oGPIO_F ; LED 使用 GPIOF 组的管脚

ldr r2, =0x55aa ; 使能 EINT0, EINT1, EINT2, EINT3,

; 另四个管脚配置成输出, 屏蔽 EINT4, 5, 6, 7

str r2, [r1, #oGPIO_CON]

mov r2, #0xff

str r2, [r1, #oGPIO_UP] ; disable the pull-up function

mov r2, #0x00

str r2, [r1, #oGPIO_DAT]

#endif

#if 0

@ SVC

mrs r0, cpsr

bic r0, r0, #0xdf

orr r1, r0, #0xd3

msr cpsr_all, r1

#endif

@ set GPIO for UART ; 设置串口

mov r1, #GPIO_CTL_BASE

add r1, r1, #oGPIO_H ; 设置 GPIO_H 组管脚为串口
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
ldr r2, gpio_con_uart

str r2, [r1, #oGPIO_CON]

ldr r2, gpio_up_uart

str r2, [r1, #oGPIO_UP]

/*****

@ inital values for GPIO

gpio_con_uart:

.long vGPHCON ; vGPHCON 在/vivi/include/platform/smdk2410.h 中赋值

; #define vGPHCON 0x0016faaa

; GPIO_H 配置为 nCTS0, nRTS0, RXD0, TXD0, RXD1,

; TXD1, nCTS1, nRTS1,

/**** 三个串口都使能，可能需要修改#define vGPHCON 0x0016aaaa ****/

gpio_up_uart:

.long Vgphup ; 同上#define vGPHUP 0x000007ff

; The pull-up function is disabled.

*****/

bl InitUART ; 跳转到 InitUART 串口初始化函数

/*****

@ Initialize UART
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

@

@ r0 = number of UART port

InitUART:

ldr r1, SerBase

/*****

.align 4 ; 缺省情况下在 vivi 中只初始化了 UART0

SerBase:

#if defined(CONFIG_SERIAL_UART0)

.long UART0_CTL_BASE ; 基地址在/vivi/include/s3c2410.h 中定义

#elif defined(CONFIG_SERIAL_UART1)

.long UART1_CTL_BASE

#elif defined(CONFIG_SERIAL_UART2)

.long UART2_CTL_BASE

#else

#error not defined base address of serial

#endif

*****/

mov r2, #0x0

str r2, [r1, #oUFCON]

str r2, [r1, #oUMCON]

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
mov r2, #0x3

str r2, [r1, #oULCON]

ldr r2, =0x245

str r2, [r1, #oUCON]

#define UART_BRD ((50000000 / (UART_BAUD_RATE * 16)) - 1)

mov r2, #UART_BRD

str r2, [r1, #oUBRDIV]


mov r3, #100

mov r2, #0x0

1: sub r3, r3, #0x1

tst r2, r3

bne 1b


#if 0

mov r2, #'U'

str r2, [r1, #oUTXHL]

1: ldr r3, [r1, #oUTRSTAT]

and r3, r3, #UTRSTAT_TX_EMPTY

tst r3, #UTRSTAT_TX_EMPTY
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
bne lb

mov r2, #'0'

str r2, [r1, #oUTXHL]

1: ldr r3, [r1, #oUTRSTAT]

and r3, r3, #UTRSTAT_TX_EMPTY

tst r3, #UTRSTAT_TX_EMPTY

bne lb

#endif

mov pc, lr

/*****/

#ifdef CONFIG_DEBUG_LL ; 打印调试信息，缺省未定义

@ Print current Program Counter

ldr r1, SerBase

mov r0, #'r'

bl PrintChar

mov r0, #'n'

bl PrintChar

mov r0, #'@'

bl PrintChar

mov r0, pc
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
bl PrintHexWord

#endif

#ifdef CONFIG_BOOTUP_MEMTEST

@ simple memory test to find some DRAM faults.

bl memtest

#endif

#ifdef CONFIG_S3C2410_NAND_BOOT ; 从 NAND Flash 启动

bl copy_myself ; 跳转到 copy_myself 函数

/***** /

@

@ copy_myself: copy vivi to ram

@

copy_myself:

mov r10, lr

@ reset NAND

mov r1, #NAND_CTL_BASE

ldr r2, =0xf830 @ initial value

str r2, [r1, #oNFCNF]

ldr r2, [r1, #oNFCNF]
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
bic r2, r2, #0x800 @ enable chip

str r2, [r1, #oNFCNF]

mov r2, #0xff @ RESET command

strb r2, [r1, #oNFCMD]

mov r3, #0 @ wait

1: add r3, r3, #0x1

cmp r3, #0xa

blt 1b

2: ldr r2, [r1, #oNFSTAT] @ wait ready

tst r2, #0x1

beq 2b

ldr r2, [r1, #oNFCNF]

orr r2, r2, #0x800 @ disable chip

str r2, [r1, #oNFCNF]

@ get read to call C functions (for nand_read())

ldr sp, DW_STACK_START @ setup stack pointer

mov fp, #0 @ no previous frame, so fp=0

@ copy vivi to RAM

ldr r0, =VIVI_RAM_BASE
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

/*****在/vivi/linux/platform/smdk2410.h 中定义

```
#define VIVI_RAM_BASE (DRAM_BASE + DRAM_SIZE - VIVI_RAM_SIZE)
```

```
*****/
```

```
mov r1, #0x0
```

```
mov r2, #0x20000 ; 0x20000→ 128k 字节
```

```
bl nand_read_ll ; nand_read_ll 在/vivi/arch/s3c2410/nand_read.c 中定义
```

```
; r0, r1, r2 分别为函数的三个参数
```

```
; 从 NANDFlash 的 0 地址拷贝 128k 到 SDRAM 指定处
```

```
tst r0, #0x0
```

```
beq ok_nand_read
```

```
#ifdef CONFIG_DEBUG_LL
```

```
bad_nand_read:
```

```
ldr r0, STR_FAIL
```

```
ldr r1, SerBase
```

```
bl PrintWord
```

```
1: b lb @ infinite loop
```

```
#endif
```

```
ok_nand_read:
```

```
#ifdef CONFIG_DEBUG_LL
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
ldr r0, STR_OK

ldr r1, SerBase

bl PrintWord

#endif


@ verify

mov r0, #0

ldr r1, =0x33f00000

mov r2, #0x400 @ 4 bytes * 1024 = 4K-bytes

go_next:

ldr r3, [r0], #4

ldr r4, [r1], #4

teq r3, r4

bne notmatch

subs r2, r2, #4

beq done_nand_read

bne go_next

notmatch:

#ifdef CONFIG_DEBUG_LL

sub r0, r0, #4
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
ldr r1, SerBase

bl PrintHexWord

ldr r0, STR_FAIL

ldr r1, SerBase

bl PrintWord

#endif

1: b 1b

done_nand_read:

#ifdef CONFIG_DEBUG_LL

ldr r0, STR_OK

ldr r1, SerBase

bl PrintWord

#endif

mov pc, r10 ; vivi 拷贝到 SDRAM 完成，函数返回

*****/

@ jump to ram

ldr r1, =on_the_ram

add pc, r1, #0

nop
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
nop

1: b 1b @ infinite loop

on_the_ram:

#endif

#ifdef CONFIG_DEBUG_LL

ldr r1, SerBase

ldr r0, STR_STACK

bl PrintWord

ldr r0, DW_STACK_START

bl PrintHexWord

#endif

@ get read to call C functions

ldr sp, DW_STACK_START @ setup stack pointer

mov fp, #0 @ no previous frame, so fp=0

mov a2, #0 @ set argv to NULL

bl main @ call main

mov pc, #FLASH_BASE @ otherwise, reboot

@

@ End VIVI head

@
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

1.4.2 vivi 的第二阶段

vivi 的第二阶段是从 main () 函数开始, 同一般的 C 语言程序一样, 该函数在 /init/main.c 文件中, 总共可以分为 8 个步骤。

(1) 函数开始, 通过 putstr(vivi_banner) 打印出 vivi 的版本。Vivi_banner 在 /init/version.c 文件中定义

(2) 对开发板进行初始化 (board_init 函数), board_init 是与开发板紧密相关的, 这个函数在 /arch/s3c2410/smdk.c 文件中。开发板初始化主要完成两个功能, 时钟初始化 (init_time ()) 和通用 IO 口设置 (set_gpios())。

```
void set_gpios(void)
```

```
{
```

```
GPACON = vGPACON;
```

```
GPBCON = vGPBCON;
```

```
GPBUP = vGPBUP;
```

```
GPCCON = vGPCCON;
```

```
GPCUP = vGPCUP;
```

```
GPDCON = vGPDCON;
```

```
GPDUP = vGPDUP;
```

```
GPECON = vGPECON;
```

```
GPEUP = vGPEUP;
```

```
GPFCON = vGPFCON;
```

```
GPFUP = vGPFUP;
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
GPGCON = vGPGCON;  
  
GPGUP = vGPGUP;  
  
GPHCON = vGPHCON;  
  
GPHUP = vGPHUP;  
  
EXTINT0 = vEXTINT0;  
  
EXTINT1 = vEXTINT1;  
  
EXTINT2 = vEXTINT2;  
  
}
```

其中，GPIO 口在 smdk2410.h (\vivi\include\platform\目录下) 文件中定义。

(3) 内存映射初始化和内存管理单元的初始化工作：

```
mem_map_init();  
  
mmu_init();
```

这两个函数都在/arch/s3c2410/mmu.c 文件中。

```
void mem_map_init(void)  
{  
  
#ifdef CONFIG_S3C2410_NAND_BOOT  
  
mem_map_nand_boot();  
  
#else  
  
mem_map_nor();  
  
#endif
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
cache_clean_invalidate();

tlb_invalidate();

}
```

如果配置 vivi 时使用了 NAND 作为启动设备，则执行 mem_map_nand_boot()，否则执行 mem_map_nor()。这里要注意的是，如果使用 NOR 启动，则必须先把 vivi 代码复制到 RAM 中。这个过程是由 copy_vivi_to_ram() 函数来完成的。代码如下：

```
static void copy_vivi_to_ram(void)

{

    putstr_hex("Evacuating 1MB of Flash to DRAM at 0x", VIVI_RAM_BASE);

    memcpy((void *)VIVI_RAM_BASE, (void *)VIVI_ROM_BASE, VIVI_RAM_SIZE);

}
```

VIVI_RAM_BASE、VIVI_ROM_BASE、VIVI_RAM_SIZE 这些值都可以在 smdk2410.h 中查到，并且这些值必须根据自己开发板的 RAM 实际大小修改。这也是在移植 vivi 的过程中需要注意的一个地方。

mmu_init() 函数中执行了 arm920_setup 函数。这段代码是用汇编语言实现的，针对 arm920t 核的处理器。

(4) 初始化堆栈，heap_init()。(定义在\ vivi\lib\heap.c 文件中)

```
int heap_init(void)
{

    return mmalloc_init((unsigned char *) (HEAP_BASE), HEAP_SIZE);

}
```

(5) 初始化 mtd 设备，mtd_dev_init()。

```
int mtd_init(void)

{
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
int ret;

#ifdef CONFIG_MTD_CFI

ret = cfi_init();

#endif

#ifdef CONFIG_MTD_SMC

ret = smc_init();

#endif

#ifdef CONFIG_S3C2410_AMD_BOOT

ret = amd_init();

#endif

if (ret) {

mymtd = NULL;

return ret;

}

return 0;

}
```

这几个函数可以在/drivers/mtd/maps/s3c2410_flash.c 里找到。

(6) 初始化私有数据, init_priv_data()。(定义在\ vivi\lib\priv_data\rw.c 文件中)

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

(7) 初始化内置命令, `init_builtin_cmds()`。

通过 `add_command` 函数, 加载 `vivi` 内置的几个命令。

(8) 启动 `boot_or_vivi()`。

启动成功后, 将通过 `vivi_shell()` 启动一个 `shell` (如果配置了 `CONFIG_SERIAL_TERM`), 此时 `vivi` 的任务完成。

1.5 启动代码执行流程图

(1) `head.s` 代码执行流程 (2) `main.c` 代码执行流程

1.6 vivi 的配置文件

Vivi 的初始配置文件位置: `/vivi/arch/def-configs/smkd2410`, 通过 `make menuconfig` 修改后的配置保存在这个文件中, 我们也可以载入一个自己的配置文件来进行编译。

2 Linux 内核移植

软件移植的概念简单地说就是让一套软件在指定的硬件平台上正常运行。移植至少包括了两个不同的硬件或者软件平台。对于应用软件来说, 移植主要考虑操作系统的差异, 重点在修改系统调用。本章的重点是 Linux 内核移植, 需要考虑硬件平台的差异, 涉及较多知识。主要内容如下:

- 1、Linux 内核移植要点;
- 2、内核体系结构框架;
- 3、从现有代码移植内核。

2.1 Linux 内核移植要点

Linux 的代码完全开放以及其良好的结构设计非常适于嵌入式系统。移植 Linux 系统包括内核、程序库和应用程序, 其中最主要的就是内核移植。由于 Linux 内核的开放性, 出现

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

了许多针对嵌入式硬件系统的内核版本，其中著名的包括 μ cLinux、RT-Linux 等。

Linux 本身对内存管理（MMU）有很好的支持。因此，在移植的时候首先要考虑到目标硬件平台是否支持 MMU。以 ARM 平台为例，ARM7 内核的 CPU 不支持 MMU，无法直接把 Linux 内核代码移植到 ARM7 核的硬件平台上。ucLinux 是专门针对 ARM7 这类没有 MMU 的硬件平台上设计的，它精简了 MMU 部分代码。本书的目标平台是 S3C2440A，该处理器基于 ARM9 核，支持 MMU，可以直接移植 Linux 2.6 版本的内核代码。

一个硬件平台最主要的是处理器，因此在移植之前需要了解目标平台的处理器。ARM 处理器内部采用 32 位的精简指令架构（RISC），核心结构设计相对简单，有低功耗电量的优势，被广泛应用到各种领域。下面介绍一下移植 Linux 内核对硬件平台需要考虑的几个问题。

2.1.1 目标平台

目标平台包括了嵌入式处理器和周围器件，处理器可能整合了一些周围器件，如中断控制器、定时器、总线控制器等。在移植之前需要确定被移植系统对外部设备和总线的支持情况。本书的 ARM 开发板采用 mini2440 平台，在 S3C2440A 外围连接了许多外围设备，包括 NOR Flash 存储器、NAND Flash 存储器、网络接口芯片、USB 控制器等。在 S3C2440A 处理器内部集成了许多常用的控制器以及嵌入式领域常用的总线控制器。对于移植 Linux 内核来说，操作处理器内部的控制器要比外部的设备容易得多。

2.1.2 内存管理单元（MMU）

对现代计算机来说，MMU 负责内存地址保护、虚拟地址和物理地址相互转换工作。在使用 MMU 的硬件平台上，操作系统通过 MMU 可以向应用程序提供大于实际物理内存的地址空间，使应用程序获得更高性能。Linux 的虚拟内存管理功能就是借助 MMU 实现的。在移植的时候要考虑目标平台的 MMU 操作机制，这部分代码是较难理解的，最好能在相似代码基础上修改，降低开发难度。

2.1.3 内存映射

嵌入式系统大多都没有配备硬盘，外部存储器只有 Flash，并且系统内存也非常有限。内存控制器（Memory Controller）负责内部和外部存储器在处理器地址空间的映射，由于硬件预设的地址不同导致每种平台内存映射的地址也不同。在移植时需要参考硬件的用户手册，得到内存地址的映射方法。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

2.1.4 存储器

由于嵌入式系统多用 Flash 存储器作为存储装置。对于文件系统来说，在 PC 流行的 ext2、ext3 文件系统在嵌入式系统无法发挥作用。幸好 Linux 支持许多文件系统，针对 Flash 存储器可以使用 JFFS2 文件系统。在移植的时候，不必要的文件系统都可以裁剪掉。

2.2 平台相关代码结构

移植 Linux 是一项复杂的工作，不仅对目标硬件平台的资源要充分了解，还需要了解 Linux 内核代码，尤其是与体系结构有关的部分。本节从内核的平台相关代码入手，先介绍内核的工作原理，然后讲解如何移植一个普通的 Linux 内核到 S3C2440A 为目标平台的开发板。

与平台相关的代码主要存放在 arch 目录下，对应的头文件在 include 目录下。以 ARM 平台为例，在 arch 目录下有一个 arm 子目录，存放所有与 ARM 体系有关的内核代码。

Linux 内核代码目录基本是按照功能块划分的，每个功能块的代码存放在一个目录下。如 mm 目录存放内存管理单元相关代码；ipc 存放了进程间通信相关的代码；kernel 存放进程调度相关代码等。

arch 目录下每个平台的代码都采用了与内核代码相同的目录结构。以 arch/arm 目录为例，该目录下 mm、lib、kernel、boot 目录与内核目录下对应目录的功能相同。此外，还有一些以字符串 mach 开头的目录，对应不同处理器特定的代码。从 arch 目录结构可以看出，平台相关的代码都存放到 arch 目录下，并且使用与内核目录相同的结构。使用 SourceInsight 工具可以看到许多的同名函数，原因就是内核代码调用的函数是平台相关的，每个平台都有自己的实现方法。对于内核来说，使用相同的名字调用，通过编译选项选择对应平台的代码。

移植内核到新的平台主要任务是修改 arch 目录下对应体系结构的代码。一般来说，已有的体系结构提供了完整的代码框架，移植只需要按照代码框架编写对应具体硬件平台的代码即可。在编写代码过程中，需要参考硬件的设计包括图纸、引脚连线、操作手册等。

2.3 建立目标平台工程框架

Linux 内核 2.6 版本已经对 ARM 处理器有很好的支持，并且对三星公司的 S3C2440 提供一定支持。但是，嵌入式硬件系统的差别很大，移植 Linux 内核到新的开发板仍然需要修改或者增加针对特定硬件的代码。

Linux 内核使用了复杂的工程文件结构，向内核添加新的代码文件需要让内核工程文件知道才行。对于 ARM 处理器来说，相关的文件都存放在 arch/arm 目录下：

```
-rwxr--r-- 1 gonglei gonglei 21371 Dec 16 2006 Kconfig
// 选项菜单配置文件
-rw-r--r-- 1 gonglei gonglei 3847 Aug 29 2005 Kconfig.debug
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
-rw-r--r--  1 gonglei  gonglei      7886 Oct 12  2005 Makefile
// make 使用的配置文件
drwxr-xr-x  4 gonglei  gonglei      4096 Apr 16  19:17 boot
// ARM 处理器通用启动代码
drwxr-xr-x  2 gonglei  gonglei      4096 Apr 16  19:17 common
// ARM 处理器通用函数
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 configs
// 基于 ARM 处理器的各种开发板配置
drwxr-xr-x  2 gonglei  gonglei      4096 Apr 16  19:17 kernel
// ARM 处理器内核相关代码
drwxr-xr-x  2 gonglei  gonglei      4096 Apr 16  19:17 lib
// ARM 处理器用到的库函数
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-aaec2000
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-clps711x
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-clps7500
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-ebsa110
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-epxa10db
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-footbridge
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-h720x
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-imx
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-integrator
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-iop3xx
drwxr-xr-x  2 gonglei  gonglei      4096 Apr 17  10:12 mach-ixp2000
// Intel IXP2xxx 系列网络处理器
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-ixp4xx
// Intel IXP4xx 系列网络处理器
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-l7200
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-lh7a40x
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-omap1
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-pxa
// Intel PXA 系列处理器
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-rpc
drwxr-xr-x  2 gonglei  gonglei      4096 Apr 16  19:21 mach-s3c2410
// 三星 S3C24xx 系列处理器
drwxr-xr-x  2 gonglei  gonglei      4096 Oct 10  2005 mach-sa1100
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-shark
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 mach-versatile
drwxr-xr-x  2 gonglei  gonglei      4096 Apr 17  10:11 mm
// ARM 处理器内存函数相关代码
drwxr-xr-x  2 gonglei  gonglei      4096 Apr 16  19:17 nwfpe
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 oprofile
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 plat-omap
drwxr-xr-x  2 gonglei  gonglei      4096 Jun 24  2008 tools // 编译工具
drwxr-xr-x  2 gonglei  gonglei      4096 Aug 29  2005 vfp
```

在 arch/arm 目录下有许多的子目录和文件。其中以 mach 字符串开头的子目录存放某种特定的 ARM 内核处理器相关文件，如 mach-s3c2410 目录存放 S3C2410、S3C2440 相关的文件。另外，在 mach 目录下还会存放针对特定开发板硬件的代码。

- ❑ boot 目录存放了 ARM 内核通用的启动相关的文件；kernel 是与 ARM 处理器相关的内核代码；mm 目录是与 ARM 处理器相关的内存管理部分代码。以上这些目录的代码一般不需要修改，除非处理器有特殊的地方，只要是基于 ARM 内核的处理一般都使用相同的内核管理代码。
- ❑ Kconfig 文件是内核使用的选项菜单配置文件，在执行 make menuconfig 命令的时候会显示出菜单。Kconfig 文件描述了菜单项，包括菜单项的属性，与其他菜单项

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

的依赖关系等。通过修改 Kconfig 文件可以告知内核有关编译的宏，内核顶层的 Makefile 通过 Kconfig 文件知道需要编译哪些文件，以及连接关系。

- Makefile 文件是一个工程文件，每个体系结构的代码中都有该文件。Makefile 文件描述了当前体系结构目录下需要编译的文件以及对应的宏的名称。内核顶层 Makefile 通过 Kconfig 文件配置的宏，结合 Makefile 定义的宏关联的代码文件去链接用户编写的代码。

通过分析 ARM 处理器体系目录的结构，加入针对 mini2440 开发板的代码主要是修改 Kconfig 文件、Makefile 文件、以及向 mach-s3c2410 目录加入针对特定硬件的代码。

2.3.1 加入编译菜单项

修改 arch/arm/mach-s3c2410/Kconfig 文件，在 endmenu 之前加入下面的内容：

```
87 config ARCH_MINI2440      // 开发板名称宏定义
88     bool "mini2440"        // 开发板名称
89     select CPU_S3C2440      // 开发板使用的处理器类型
90     help
91     Say Y here if you are using the mini2440.    // 帮助信息
```

在笔者的机器上是在第 87 行加入，读者可以根据自己机器的配置在正确位置加入配置代码。Kconfig 文件与开发板有关的代码定义在 startmenu 和 endmenu 之间，使用 config 关键字标示一个配置选项。使用 config 配置的选项会出现在 make menuconfig 的菜单项中。

2.3.2 设置宏与代码文件的对应关系

在设置宏与代码文件对应关系之前，首先建立一个空的代码文件。在 arch/arm/mach-s3c2410 目录下建立 mach-mini2440.c 文件，用于存放与 mini2440 开发板相关的代码。

建立 mach-mini2440.c 文件后，修改 arch/arm/mach-s3c2410/Makefile 文件，在文件最后加入 mach-mini2440.c 文件的编译信息：

```
43 obj-$(CONFIG_ARCH_MINI2440) += mach-mini2440.o
```

在笔者的机器上是在第 43 行加入 CONFIG_ARCH_MINI2440 宏，对应了 mach-mini2440.o 目标文件。make 工具在解析该 Makefile 的时候，会找到 mach-mini2440.o 目标文件对应的 mach-mini2440.c 文件并且编译，同时会建立一个名为 obj-CONFIG_ARCH_MINI2440 的宏。

注意：在 Makefile 中设置的宏名称是有规则的，要使用 CONFIG 开头，并且后面要与 Kconfig 菜单项配置的名称对应，否则在编译内核的时候无法找到对应的代码文件。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

2.3.3 测试工程框架

工程框架配置修改完毕后,需要进行简单的测试,根据测试结果判断框架是否搭建成功。回到内核代码顶层目录,输入 `make ARCH=arm CROSS_COMPILE=arm-linux- menuconfig` 命令,出现内核设置图形界面。

在内核配置界面选择 Load an Alternate Configuration File 菜单,进入后输入“`arch/arm/ configs/s3c2410_defconfig`”命令,确定后会加载 s3c2410 默认的配置文件。加载默认配置文件的好处是已经经过验证,用户只需要在默认配置文件的基础上修改自己的配置,减轻了配置的工作量。

加载默认配置文件后,可以开始配置新增加的菜单。进入 System Types 菜单项,打开 S3C24XX Implementations 菜单,出现一个目标开发板的列表:

```
[ ] Simtec Electronics BAST (EB2410ITX)
[ ] IPAQ H1940
[ ] Acer N30
[ ] SMDK2410/A9M2410
[ ] SMDK2440
[ ] AESOP2440
[ ] QQ2440/mini2440
[ ] Thorcom VR1000
[ ] HP iPAQ rx3715
[ ] NexVision OTOM Board
[ ] NexVision NEXCODER 2440 Light Board
[ ] mini2440
```

列表最后一项是在 20.3.1 节中添加的 mini2440 菜单项。把光标移到 mini2440 菜单项使用回车选中。选择 mini2440 开发板完毕后,保存退出内核配置界面。在命令行输入“`make ARCH=arm CROSS_COMPILE=arm-linux- bzImage`”命令编译内核代码。请注意,此时编译内核代码可能会有好多的错误,并且不会编译通过,问题是虽然建立了目标板工程框架,但是在源代码文件中没有任何内容。下面是在笔者机器上报错的部分信息提示:

```
arch/arm/kernel/traps.c: In function '__bug':
arch/arm/kernel/traps.c:627: warning: 'noreturn' function does return
kernel/intermodule.c:179: warning: 'inter_module_register' is deprecated (declared at
kernel/intermodule.c:38)
kernel/intermodule.c:180: warning: 'inter_module_unregister' is deprecated (declared at
kernel/intermodule.c:79)
kernel/intermodule.c:182: warning: 'inter_module_put' is deprecated (declared at
kernel/intermodule.c:160)
fs/yaffs2/yaffs_fs.c:198: warning: initialization from incompatible pointer type
fs/yaffs2/yaffs_fs.c:228: warning: initialization from incompatible pointer type
fs/yaffs2/yaffs_fs.c:229: warning: initialization from incompatible pointer type
fs/yaffs2/yaffs_fs.c: In function 'yaffs_proc_write':
fs/yaffs2/yaffs_fs.c:1865: warning: 'len' might be used uninitialized in this function
fs/yaffs2/yaffs_fs.c: At top level:
fs/yaffs2/yaffs_fs.c:1305: warning: 'yaffs_do_sync_fs' defined but not used
fs/yaffs2/yaffs_guts.c: In function 'yaffs_ObjectHasCachedWriteData':
fs/yaffs2/yaffs_guts.c:2997: warning: unused variable 'cache'
fs/yaffs2/yaffs_guts.c: In function 'yaffs_Scan':
fs/yaffs2/yaffs_guts.c:4490: warning: unused variable 'hl'
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号


```
fs/yaffs2/yaffs_qsort.c:77:1: warning: "min" redefined
In file included from fs/yaffs2/yportenv.h:35,
                 from fs/yaffs2/yaffs_qsort.c:30:
include/linux/kernel.h:243:1: warning: this is the location of the previous definition
drivers/char/keyboard.c:1018:2: warning: #warning "Cannot generate rawmode keyboard for
your architecture yet."
drivers/net/dm9000x.c: In function 'dmfe_probe':
drivers/net/dm9000x.c:288: warning: assignment makes integer from pointer without a cast
drivers/serial/serial_core.c:2427: warning: 'uart_register_port' is deprecated (declared
at drivers/serial/serial_core.c:2348)
drivers/serial/serial_core.c:2428: warning: 'uart_unregister_port' is deprecated (declared
at drivers/serial/serial_core.c:2405)
```

从报错信息看，主要集中在函数未定义和函数重定义两种错误。在编译的时候提示出错的函数名称需要关注，可能需要在新增的代码文件中重新定义或者调用。内核编译会出错退出，报错信息如下：

```
arm-linux-ld:arch/arm/kernel/vmlinux.lds:815: parse error
make: *** [.tmp_vmlinux1] Error 1
```

该信息提示解析 arch/arm/kernel/vmlinux.lds 文件第 815 行出错，出现该错误表示已经完成内核代码编译，在链接代码时产生问题，在后面章节会详细分析出错原因。到目前为止，向内核新增的代码框架已经可以正常工作，第 20.4 节将介绍如何编写对应开发板的代码

2.4 建立目标平台代码框架

在 2.3.3 节编译的内核代码最后出现了链接错误，提示 vmlinux.lds 文件链接失败。lds 文件是 GNU ld 工具使用的一种脚本文件，该文件描述了如何分配链接后的内存区域和地址等信息，通过 lds 文件报的错误可以推理分析问题产生的原因。

2.4.1 ARM 处理器相关结构

首先打开 arch/arm/kernel/vmlinux.lds 文件，找到 815 行，代码如下：

```
815 ASSERT((__proc_info_end - __proc_info_begin), "missing CPU support")
```

该行代码使用一个 ASSERT 宏判断 __proc_info_end 标号的地址与 __proc_info_begin 标号地址是否相同。__proc_info_end 和 __proc_info_begin 标号之间定义了一个初始化阶段使用的结构，在 ARM 处理器启动的时候，内核会调用该结构初始化 ARM 处理器。

在 arch/arm 目录下搜索 __proc_info_begin 标号：

```
$ grep -nR '__proc_info_begin' *
kernel/head.S:516:         .long   __proc_info_begin
kernel/vmlinux.lds.S:25:         __proc_info_begin = .;
kernel/vmlinux.lds.S:166:ASSERT((__proc_info_end - __proc_info_begin), "missing CPU
support")
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

使用 grep 命令搜索得到 3 条结果。从结果看出, __proc_info_begin 标号定义在 kernel/head.S 文件的第 516 行, 在 kernel/vmlinux.lds.S 文件使用到了 __proc_info_begin 标号。打开 kernel/vmlinux.lds.S 文件查看:

```
25  __proc_info_begin = .;          // .proc.info 段的起始地址
26  *(.proc.info)                  // 段名称
27  __proc_info_end = .;           // .proc.info 段的结束地址
```

在 vmlinux.lds.S 文件中定义了 .proc.info 代码段, 该段代码的起始地址和结束地址分别由 __proc_info_begin 和 __proc_info_end 标号标示, 这两个标号表示的地址是通过计算得到的。

在 ARM 体系代码中, 使用 machine_desc 结构描述与处理器相关的代码, 该结构定义在 include/asm-arm/mach/arch.h 头文件定义如下:

```
17 struct machine_desc {
18     /*
19      * Note! The first five elements are used
20      * by assembler code in head-armv.S
21      */
22     unsigned int    nr; /* architecture number */
23     unsigned int    phys_ram; /* start of physical ram */
24     unsigned int    phys_io; /* start of physical io */
25     unsigned int    io_pg_offst; /* byte offset for io
26     * page table entry */
27
28     const char      *name; /* architecture name */
29     unsigned long    boot_params; /* tagged list */
30
31     unsigned int     video_start; /* start of video RAM */
32     unsigned int     video_end; /* end of video RAM */
33
34     unsigned int     reserve_lp0 :1; /* never has lp0 */
35     unsigned int     reserve_lp1 :1; /* never has lp1 */
36     unsigned int     reserve_lp2 :1; /* never has lp2 */
37     unsigned int     soft_reboot :1; /* soft reboot */
38     void             (*fixup)(struct machine_desc *,
39     struct tag *, char **,
40     struct meminfo *);
41     void             (*map_io)(void); /* IO mapping function */
42     void             (*init_irq)(void);
43     struct sys_timer *timer; /* system tick timer */
44     void             (*init_machine)(void);
45 };
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

machine_desc 结构描述了处理器体系结构编号、物理内存大小、处理器名称、I/O 处理函数、定时器处理函数等。每种 ARM 核的处理器都必须实现一个 machine_desc 结构，内核代码会使用该结构

2.4.2 建立 machine_desc 结构

使用，建议使用宏建立结构。打开 arch/arm/mach-s3c2410/mach-mini2440.c 文件，加入下面的代码：

```
53 MACHINE_START(MINI2440, "MINI2440")           // 定义结构名称
54   .phys_ram = S3C2410_SDRAM_PA,                // 物理内存起始地址
55   .phys_io  = S3C2410_PA_UART,                 // 物理端口起始地址
56   .io_pg_offst = (((u32)S3C24XX_VA_UART) >> 18) & 0xfffc,
57   .boot_params = S3C2410_SDRAM_PA + 0x100, // 启动参数存放地址
58
59   .init_irq = mini2440_init_irq,                // 中断初始化函数
60   .map_io   = mini2440_map_io,                 // I/O 端口内存映射函数
61   .init_machine = mini2440_init,              // 初始化函数
62   .timer    = &s3c24xx_timer,                 // 定时器
63 MACHINE_END
```

MACHINE_START 宏定义了一个名为 MINI2440 的结构，并且定义了相关的内存和端口地址、处理函数等。请读者注意定义结构的行号不是从第 1 行开始的，前面的代码行包含了头文件。头文件可以从其他文件复制一份，如从 mach-smdk2440.c 文件复制一份头文件定义。

提示：参考其他类似工程的代码是一个捷径，对于能复用的代码和变量建议使用已经定义好的，这样可以减轻编码和调试的工作量，减少出错机会。

MINI2440 结构中使用到了 S3C2410_SDRAM_PA 和 S3C2410_PA_UART 宏，这两个宏分别定义了开发板物理内存起始地址和物理端口起始地址。由于 2410 和 2440 处理器对内存地址映射关系相同，可以直接使用 S3C2410_SDRAM_PA 和 S3C2410_PA_UART 宏。有关 S3C2440 处理器内存映射请参考处理器手册的内存管理章节。

2.4.3 加入处理函数

在 mach-mini2440.c 文件中加入 MINI2440 结构指定的几个函数，定义如下：

```
52 void __init mini2440_init_irq(void)           // 中断初始化函数
53 {
54 }
55
56 void __init mini2440_init(void)                // 处理器初始化函数
57 {
58 }
59
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
60 void __init mini2440_map_io(void)    // I/O 端口映射初始化函数
61 {
62 }
```

在 mach-mini2440.c 文件中加入了 mini2440_init_irq()、mini2440_init() 和 mini2440_map_io() 这 3 个函数。请读者注意这 3 个函数定义的时候使用了 __init 关键字，__init 关键字告诉 ld 链接器把函数放在初始化段，初始化段的代码仅在初始化的时候被调用一次。

提示：在本节函数留空即可，后面的章节会不断增加代码，本节主要是搭建代码框架。

2.4.4 加入定时器结构

在 MINI2440 结构定义中，使用了一个名为 s3c24xx_timer 的 sys_timer 结构变量，该变量定义在 arch/arm/mach-s3c2410/timer.c 文件定义如下：

```
252 struct sys_timer s3c24xx_timer = {
253     .init    = s3c2410_timer_init,    // 定时器初始化函数
254     .offset  = s3c2410_gettimeoffset, // 读取定时器延时
255     .resume  = s3c2410_timer_setup    // 恢复定时器
256 };
```

S3C24xx 系列处理器定时器的操作相同，因此使用内核代码已经定义好的定时器结构即可，无须从头开发。

2.4.5 测试代码结构

回到内核源代码根目录，执行 make ARCH=arm CROSS_COMPILE=arm-linux- bzImage 开始编译内核。这次编译没有出错信息，会得到正确的编译结果。查看 arch/arm/boot 目录已经有目标文件 Image.gz，表示已经编译生成运行于 ARM 处理器的内核。

到目前为止，已经可以编译工作在 ARM 处理器上的代码，但是内核代码还不能启动，因为还没有加入实际的代码，在 20.5 节中将介绍如何加入目标平台相关的代码。

2.5 构建目标板代码

Linux 内核已经为 ARM 处理器设计好了代码框架，只要按照这个框架加入针对某种开发板和处理器的代码即可工作。加入代码还是按照前面提到的原则，能使用已有的通用代码尽量使用，并且尽可能地参考现有开发板代码的处理方法。

2.5.1 处理器初始化

首先在 mach-mini2440.c 文件中加入处理器初始化代码如下：

```
56 void __init mini2440_init(void)
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
57 {  
58   set_s3c2410ts_info(&mini2440_ts_cfg);           // 注册触摸屏结构  
59   set_s3c2410udc_info(&mini2440_udc_cfg);        // 注册 UDC 结构  
60   set_s3c2410fb_info(&mini2440_lcdcfcg);         // 注册 LCD 结构  
61 }
```

在 mini2440_init() 函数中注册了 3 个结构, 分别用于初始化触摸屏、UDC 和 LCD, 这 3 个结构都是针对三星 ARM9 处理器的。接下来定义这三个结构代码如下:

```
53 static struct s3c2410_ts_mach_info mini2440_ts_cfg __initdata = {  
54   .delay = 20000,  
55   .presc = 55,  
56   .oversampling_shift = 2,  
57 };  
58  
59  
60 static struct s3c2410_udc_mach_info mini2440_udc_cfg __initdata = {  
61   .udc_command = pullup,           // 设置 udc 处理函数  
62 };  
63  
64  
65 static struct s3c2410fb_mach_info mini2440_lcdcfcg __initdata = {  
66   .regs = {                        // 设置控制寄存器  
67     .lcdcon1 = S3C2410_LCDCON1_TFT16BPP | \  
68               S3C2410_LCDCON1_TFT | \  
69               S3C2410_LCDCON1_CLKVAL(0x04),  
70  
71     .lcdcon2 = S3C2410_LCDCON2_VBPD(1) | \  
72               S3C2410_LCDCON2_LINEVAL(319) | \  
73               S3C2410_LCDCON2_VFPD(5) | \  
74               S3C2410_LCDCON2_VSPW(1),  
75  
76     .lcdcon3 = S3C2410_LCDCON3_HBPD(36) | \  
77               S3C2410_LCDCON3_HOZVAL(239) | \  
78               S3C2410_LCDCON3_HFPD(19),  
79  
80     .lcdcon4 = S3C2410_LCDCON4_MVAL(13) | \  
81               S3C2410_LCDCON4_HSPW(5),  
82  
83     .lcdcon5 = S3C2410_LCDCON5_FRM565 |  
84               S3C2410_LCDCON5_INVVLINE |  
85               S3C2410_LCDCON5_INVVFRAME |  
86               S3C2410_LCDCON5_PWREN |  
87               S3C2410_LCDCON5_HWSWP,  
88   },  
89  
90   .lpcsel = 0xf82,  
91  
92   .gpcccon = 0xaa955699,  
93   .gpcccon_mask = 0xffc003cc,  
94   .gpcup = 0x0000ffff,  
95   .gpcup_mask = 0xffffffff,  
96  
97   .gpdcon = 0xaa95aaa1,           // 设置通用控制寄存器
```

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

```
98 .gpdcon_mask = 0xffc0fff0, // 通用控制寄存器掩码
99 .gpdup = 0x0000faff,      // 设置通用控制寄存器
100 .gpdup_mask = 0xffffffff, // 通用控制寄存器掩码
101
102 .fixed_syncs = 1,          // 同步
103 .width = 240,              // 设置屏幕宽度像素值
104 .height = 320,            // 设置屏幕高度像素值
105
106 .xres = {                  // 设置 LCD 屏幕横向像素值
107     .min = 240,            // 最小值
108     .max = 240,            // 最大值
109     .defval = 240,         // 默认值
110 },
111
112 .yres = {                  // 设置 LCD 屏幕纵向像素值
113     .max = 320,            // 最大值
114     .min = 320,            // 最小值
115     .defval = 320,         // 默认值
116 },
117
118 .bpp = {                   // 颜色位数
119     .min = 16,             // 最小值
120     .max = 16,             // 最大值
121     .defval = 16,          // 默认值
122 },
123 };
```

mini2440_ts_cfg 结构定义了触摸屏相关参数；mini2440_lcdcfg 结构定义了 LCD 控制器相关参数，S3C2440 提供了一组 LCD 控制器，用于设置 LCD 的颜色、像素值、数据传输方式、同步方式等结构。读者可以参考 S3C2440 处理器手册，对照代码中的值得到配置的方式。提示：LCD 控制器的配置比较复杂，S3C2440 提供的 LCD 控制器仅支持数字信号输入的液晶屏，在连接液晶屏之前需要参考液晶屏的手册。

在 mini2440_udc_cfg 结构中使用了一个 pullup 回调函数，定义如下：

```
52 static void pullup(unsigned char cmd)
53 {
54     switch (cmd)
55     {
56         case S3C2410_UDC_P_ENABLE :    // 打开 UDC
57             break;
58         case S3C2410_UDC_P_DISABLE :   // 关闭 UDC
59             break;
60         case S3C2410_UDC_P_RESET :     // 重启 UDC
61             break;
62         default: break;
63     }
64 }
```

pullup() 函数不做任何处理，如果以后需要添加对 UDC 的处理可以在 pullup() 函数中加入处理代码。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

2.5.2 端口映射

端口映射函数设置 S3C2440 处理器的 I/O 端口描述结构、时钟频率、串口等，代码如下：

```
150 void __init mini2440_map_io(void)
151 {
152     s3c24xx_init_io(mini2440_iodesc, ARRAY_SIZE(mini2440_iodesc));
153                                     // 初始化 I/O 结构
154     s3c24xx_init_clocks(12000000); // 设置时钟频率
155     s3c24xx_init_uarts(mini2440_uartcfgs, ARRAY_SIZE(mini2440_uartc
156     fgs)); // 设置串口结构
157     s3c24xx_set_board(&mini2440_board); // 设置开发板结构
158     s3c_device_nand.dev.platform_data = &bit_nand_info;
159 }
160 }
```

在 mini2440_map_io() 函数中，初始化了 3 个结构，定义如下：

```
66 static struct map_desc mini2440_iodesc[] __initdata = {
67     {vSMDK2410_ETH_IO, pSMDK2410_ETH_IO, SZ_1M, MT_DEVICE},
68                                     // 网卡接口映射
69     {0xe0000000, 0x08000000, 0x00100000, MT_DEVICE},
70     {0xe0100000, 0x10000000, 0x00100000, MT_DEVICE},
71     { (unsigned long)S3C24XX_VA_IIS, S3C2410_PA_IIS, S3C24XX_SZ_IIS, MT_
72     DEVICE },
73 };
74 static struct s3c2410_uartcfg mini2440_uartcfgs[] = {
75     [0] = {
76         .hwport      = 0, // 串口 0
77         .flags        = 0,
78         .ucon         = 0x3c5,
79         .ulcon        = 0x03,
80         .ufcon        = 0x51,
81     },
82     [1] = {
83         .hwport      = 1, // 串口 1
84         .flags        = 0,
85         .ucon         = 0x3c5,
86         .ulcon        = 0x03,
87         .ufcon        = 0x51,
88     },
89     [2] = {
90         .hwport      = 2, // 红外线接口
91         .flags        = 0,
92         .uart_flags   = 0,
93         .ucon         = 0x3c5,
94         .ulcon        = 0x03,
95         .ufcon        = 0x51,
96     }
97 };
98 static struct s3c24xx_board mini2440_board __initdata = {
99     .devices          = mini2440_devices, // 开发板设备列表
100     .devices_count    = ARRAY_SIZE(mini2440_devices) // 结构大小
101 };
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

map_desc 结构描述了内存虚拟地址和物理地址之间的关系，供配置 MMU 使用。mini2440_uartcfgs[] 结构描述了开发板上两个串口和一个红外线接口的定义。mini2440_board 结构描述了开发板上存在的设备列表 mini2440_devices，定义如下：

```
66 static struct platform_device *mini2440_devices[] __initdata = {
67     &s3c_device_lcd,           // LCD 控制器
68     &s3c_device_wdt,           // 看门狗控制器
69     &s3c_device_i2c,           // I2C 控制器
70     &s3c_device_ts,            // 触摸屏控制器
71     &s3c_device_nand,          // NAND Flash 控制器
72     &s3c_device_rtc,           // RTC 控制器
73 };
```

在 mini2440_devices 结构中定义了 LCD 控制器、看门狗控制器、I2C 控制器、触摸屏控制器等结构，这些结构的定义都是使用内核提供的标准结构，定义在 devs.c 文件中。

20.5.3 中断处理

内核提供了一个 s3c24xx_init_irq() 处理函数，因此中断处理函数直接引用即可。

```
186 void __init mini2440_init_irq(void)
187 {
188     s3c24xx_init_irq(); // 调用系统提供的中断处理函数
189 }
```

2.5.4 定时器处理

内核提供了一个定时器处理函数结构如下。

```
struct sys_timer s3c24xx_timer = {
    .init      = s3c2410_timer_init,           // 定时器初始化函数
    .offset    = s3c2410_gettimeoffset,       // 获取定时器值
    .resume    = s3c2410_timer_setup           // 恢复定时器设置
};
```

在代码中直接使用 s3c24xx_timer 结构即可。

2.5.5 编译最终代码

到目前为止，已经添加了所有与 mini2440 开发板有关的代码，保存文件后，可以开始编译内核。回到内核代码根目录，执行“make ARCH=arm CROSS_COMPILE=arm-linux- bzImage”重新编译代码，最终在 arch/arm/boot 目录下生成 bzImage 文件，针对开发板的内核代码编译成功。

通过 U-Boot 或者其他的 Bootloader 工具可以把代码烧写到开发板的 Flash 存储器上，然后重新启动开发板，可以从液晶屏看到启动过程打印的提示信息。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

提示：由于没有设置 USB 控制器和声音控制器，因此开发板上和 USB 及声音相关的功能都无法使用。

移植内核到新的硬件平台是一件繁琐的事情，读者在移植过程中需要有耐心。移植内核代码要涉及软件和硬件相关的知识，读者应该结合硬件提供的电路图纸和手册，利用现有的软件工具和代码，构建符合需求的内核代码

3 linux 2.6 内核配置选项详细说明

3.1 Code maturity level options --->

- [*] Prompt for development and/or incomplete code/drivers (y)
; 选择尚未完全测试的代码（alpha-test 态），事实上它是安全的，建议选择。
- [*] Select only drivers expected to compile cleanly(y)
; 隐藏可能存在问题的驱动，建议选择，如果没找到对应设备的驱动，将它取消试试。

3.2 General setup --->

- () Local version - append to kernel release (enter,输入字符窜)
; 从 2.6.8 的版本起，可以在内核版本号后面添加个性化字符窜。
 - [] Automatically append version information to the version string (NEW)(n)
; 这个没看懂,先不选上。
 - [*] Support for paging of anonymous memory (swap)(y)
; 如果使用了 swap 分区提供虚拟内存,一定要选上它。
 - [*] System V IPC (y)
; System V 的进程间通信, 选上。
 - [*] POSIX Message Queues (y)
; POSIX 消息队列, 选上。
 - [*] BSD Process Accounting (y)
; 如果选上,user process 可以通过系统调用使内核在它退出时将相关信息写入某个文件(如进程创建时间,拥有者,命令,内存使用量...)选上它,可以在应用程序中利用这些信息。
 - [*] BSD Process Accounting version 3 file format (y)
; 将前面所述的进程信息记录到 v3 格式的文件中, 选上它
 - [*] Sysctl support (y)
; 提供动态更改内核参数与变量的接口, 而不需要重新启动系统。打开这个选项将会增加内核的体积至少 8KB。
- 如果你的内核仅用制作安装与恢复系统系统盘那么可以不选，以减少对内存的占用。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

[] Auditing support(n)

; 允许其他内核子系统的内核审查,不知道什么鸟意思,不选上.

[*] Kernel Userspace Events (y)

; 开启内核-用户空间事件层,它是比 socket 简单的 kernel-user 通信机制. 这样应用程序就可以通过监听不用轮询系统设备或文件

[] Kernel .config support(n)

; 将.config 文件编译到内核中, 以显示运行中的内核使用哪个选项.不要选择.

() Initramfs source file(s)(n)

; 好像是 RAM FS 初始化的吧,不知道什么鸟东西,不选!

[] Optimize for size (Look out for broken compilers!)(n)

; 用 gcc 编译内核时,优化选项是 -O2,选择它将改为-Os, 生成比较小的内核.(老版本的 gcc 可能因此产生错误代码)

[] Configure standard kernel features (for small systems)---> (n)

; 针对小系统 (embedded)裁减内核, 桌面系统不用选择.

3.3 module support--->

[*] Enable loadable module support (y)

; 使内核支持模块,当然要选择! (使用 modprobe, lsmod, modinfo, insmod, rmmod 工具...)

[*] Module unloading (y)

; 卸载模块,选择!(有些模块一旦加载就不能卸载, 不管是否选择了这个选项)

[*] Forced module unloading(y)

; 强制卸载内核, 即便内核认为该行为不安全的时候.(rmmod -f 强制卸载,不等停止使用模块)

[] Module versioning support (EXPERIMENTAL)(n)

; 一般地,我们编译的模块是用于当前运行的内核, 选择该选项可以针对其他的内核编译模块. 先不选择.

[] Source checksum for all modules (n)

; 查看模块中是哪些代码的,不选

[*] Automatic kernel module loading(y)

; 内核在任务中要使用一些被编译为模块的驱动或特性时, 先使用 modprobe 命令来加载它
该选项自动调用 modprobe 加载需要的模块.当然选择!

3.4 Block layer --->

[] Support for Large Block Devices (n)

; 如果有超过 2T 的块设备,则选择它以支持大容量块设备

[] IO Schedulers---->

<*> Anticipatory I/O scheduler (y)

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

<*> Deadline I/O scheduler(y)

<*> CFQ I/O scheduler (y)

Default I/O scheduler (Anticipatory)--->

(X) Anticipatory

上述 3 中调度方式都选择,默认的调度方式选为 anticipatory (按空格选择)

3.5 Processor type and features--->

Subarchitecture Type (PC-compatible) --->

; 用的 PC,选 PC-compatible

Processor family (Pentium-4/Celeron(P4-based)/Pentium-4 M/Xeon)--->

; 选择 CPU 类型, 可通过 cat /proc/cpuinfo 查看

[] Generic x86 support(n)

; 如果没有列出你所使用的 CPU 类型,而你的 CPU 又是 X86 的,选上它,否则不选.它增加了通用性却降低了特定 CPU 的性能.

[*] HPET Timer Support (y)

; 这也是一个新的特性, HPET 是 intel 制定的新的用以代替传统的 8254(PIT)中断定时器与 RTC 的定时器,全称叫作高精度事件定时器.如果你有一台较新的机器就选它吧,一般它是一个安全的选项, 即使你的硬件不支持 HPET 也不会造成问题,因为它会自动用 8254 替换.

[] Symmetric multi-processing support (n)

; 只有一个 CPU,SMP 不用选择

Preemption Model (Preemptible Kernel (Low-Latency Desktop)) --->

(X) Preemptible Kernel (Low-Latency Desktop)

; 2.6 内核的特点:抢占式内核.选择可抢占式内核以提升桌面系统的交互性能或实时性.

[*] Preempt The Big Kernel Lock (NEW)(y)

; 抢占大内核锁?不清楚,不过选择了可加强桌面系统性能.

[*] Local APIC support on uniprocessors(y)

; 单 CPU 的本地 APIC (advanced programmable interrupt controller)支持,它内嵌在 cpu 中支持 cpu 自身产生的中断.建议选择,就算 cpu 不支持 APIC,也没有影响.

[*] IO-APIC support on uniprocessors (y)

; 同上,支持 I/O 高级可编程中断控制器.

[*] Machine Check Exception (y)

; 如果系统出现问题, 内核采取一定的措施,比如打印警告信息或挂起系统.

cat /proc/cpuinfo | grep mce 若 CPU flags 中有 mce,

这个功能是需要硬件支持的.你可以查看/proc/cpuinfo 看看是否有 mce 标志, 则说明 CPU 支持该选项.

启动时加 nomce 参数可关闭它.

< > Check for non-fatal errors on AMD Athlon/Duron / Intel Pentium 4(n)

; 启动一个 5 秒的定时器,跟踪非致命错误并更正,记录它.如果不是特定的 CPU,不要选择.(我的

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

本本是 Pentium M,不选择)

[] check for P4 thermal throttling interrupt.

; P4 的 CPU 温度过高时会在屏幕上显示出相关的信息,只适用于 Pentium 4

< > Toshiba Laptop support(n)

< > Dell laptop support (n)

; 上述两项都针对东芝和戴尔 IInspire 8000 笔记本,不选.

< > Enable X86 board specific fixups for reboot (n)

; 修正主板/芯片组以正确重启或工作,目前只针对 GX1, CS5530A, TROM2.1

lspci -v | grep CX1 ...若系统不支持,则不用选择.

< > /dev/cpu/microcode - Intel IA32 CPU microcode support(n)

; 更新 intel IA32 cpu 的微码(内核是不自带的,需要另外下载)

< > /dev/cpu/*/msr - Model-specific register support (n)

; 让 privileged 进程访问 X86 的 MSRs(model-specific registers),一般用于 intel 的 Embedded cpu.

< > /dev/cpu/*/cpuid - CPU information support(n)

; 在/dev/cpu 中建立一系列的设文件,以使过程访问指定的 CPU.

Firmware Drivers --->

< > BIOS Enhanced Disk Drive calls determine boot disk (EXPERIMENTAL)(n)

; 可以打开实模式下 BIOS 中的增强磁盘设备服务,以决定从哪个磁盘上启动.一般的 BIOS 不支持.

< > BIOS update support for DELL systems via sysfs(n)

; 戴尔的 BIOS 更新系统,需要一些应用软件的支持.

< > Dell Systems Management Base Driver (n)

; 为上述的 DELL BIOS 更新系统提供 sysfs 借口,先不设,以后再研究.我的本本是 Dell D600.

High Memory Support (off) --->

; 如果有大于 4G 的内存,可选择.我内存只有 512M.

Memory model (Flat Memory) --->

; 选择内存模式, flat memory 记忆被盗,查查资料了再说,先选上.

[] Math emulation(n)

; 在你的 CPU 上如果没有数学协处理器的话,打开这个选项可以让内核模拟一个.以提升浮点计算能力,不过慢的可以.如果你使用的不是古董 CPU 的话(486SX 以前的),这一项你永远都不需要.

[*] MTRR (Memory Type Range Register) support(y)

; 在 Intel p6 家族的处理中(Ppro、PII 和更新的)有一个内存类型范围寄存器,可用来控制处理器访问的内存范围.打开它一般可以提升显卡的显示性能(2.5 倍).

[] Boot from EFI support (EXPERIMENTAL)(n)

;EFI 规范基本上是一个让 PC 可以在开机前(preboot)进行扫毒及诊断的执行环境.英特尔已经利用 EFI 建立一个可以取代 BIOS 的开机前软件框架.这个框架名为"EFI 平台创新开发框

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

架”，其工程代码为 Tiano，这个框架让电脑厂商可以撰写开机前软件的模组，这种模组很像 Windows 的驱动程序。它需要 ELILO bootloader, grub 或 lilo 不支持 EFI.

[] Use register arguments (EXPERIMENTAL)(n)

；使用寄存器参数,(编写系统调用的时候可能有用,参考<Linux Kernel Development>,v2 5.3 它使用不同的 ABI,将函数的前三个参数通过寄存器传递.(gcc-3.0 或更新的才支持).先不设置,学习编写系统调用的时候再看.

[*] Enable seccomp to safely compute untrusted bytecode(y)

；使用 seccomp 将计算程序孤立到它们各自的地址空间.如果不是用于嵌入式系统，还是在这里选 yes

Timer frequency (250 HZ)--->

；时钟频率,选为 250 赫兹.

(0x100000) Physical address where the kernel is loaded

；设定内核加载的物理地址.默认为 0x100000.不要更改!

[] kexec system call (EXPERIMENTAL)(n)

；kexec 能够关闭当前内核,运行另外一个内核.

3.6 Power management options (ACPI, APM)--->

[*] Legacy Power Management API (y)

；为 pm_regiter()提供支持.

[] Power Management Debug Support(n)

；支持电源管理的调试

[*] Software Suspend (EXPERIMENTAL) (y)

；支持系统 suspend(休眠),打开这项功能后,可用 swsusp 或者 shutdown -z <time> 来挂起系统.这样系统会把你当前正在进行的工作(也就是当前内存中的内容)作成一个镜象保存到你的交换分区中,在你下一次启动时使用启动参数"resume=/dev/交换分区".内核就会将上一次的工作内核从镜象文件中恢复到内存,这可以大大提高系统的启动速度.当你不想恢复上次的工作时向内核传递参数"noresume".不过系统启动后你的交换分区将不可以使用,你可以使用 mkswap 命令来重新格式化你的交换分区.这个功能不需要高级电源管理的支持.

() Default resume partition

与上述 suspend 选项配合,来指定保存镜象的分区.如果上面那个有开启就要选择用来做 suspend to disk 用的 partition

ACPI (Advanced Configuration and Power Interface) Support--->

[*] ACPI Support (y)

；使用 ACPI 来管理电源.想让它起作用,还要在系统中安装 acpid 守护程序.

[*] Sleep States(y)

；选择这个选项可以使你的系统具有挂起的功能,也就是说你可以暂时中断你的工作,让你的系统处与一种低电能消耗的状态(sleep state),你此时的系统状态会保存在内存或者磁盘上(取决

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

于挂起的深度), 当你需要时再恢复到正常的工作状态。但由于各种系统之间的差别, 目前这项功能并不完善。只有很少的设备可以完美的支持这个特性, 所以不建议一般用户使用。

<M> AC Adapter (m)

; 显示本本是使用 AC 交流电还是 DC 直流电.编译成模块!

<M> Battery (m)

; 同上,读取/proc/acpi/battery 目录中的电池信息,反馈给用户.

<M> Button (m)

; 按下电源键时,守护进程读取/proc/acpi/event, 并执行用户在这些事件上定义的动作,比如关机.

< > Video (n)

; 对主板集成显卡执行一些操作,比如定义 video POST device, 获得 EDID 信息, 设置视频输出.独立显卡就不用设了.

< > Generic Hotkey (EXPERIMENTAL)(n)

; 通用热键驱动,不需要装.

<M> Fan (m)

; 对 ACPI 风扇设备的控制支持, 通过用户程序控制风扇(打开, 关闭, 读取运行状态等).

<M> Processor (m)

; 处理器在空闲时节省电能.

<M> Thermal Zone (m)

; CPU 温度过高时,ACPI 调整工作状态以保护 CPU,强烈推荐!(大部分的本本都支持)

< > ASUS/Medion Laptop Extras(n)

< > IBM ThinkPad Laptop Extras (n)

< > Toshiba Laptop Extras (n)

上述是对华硕,IBM,东芝笔记本的扩展支持.为什么木有 DELL 的呢!

(0) Disable ACPI for systems before Jan 1st this year (默认为 0)

; 什么鸟东西,ACPI 也存在千年虫问题?按默认的

[] Debug Statements(n)

; ACPI 驱动的调试语句,会增加 50K 的内核大小,不要.

[] Power Management Timer Support (n)

; 支持电源管理定时器.如果在内核 log 中看到"Losing too many ticks!",或使用笔记本却不支持 HPET 时选择.

< > ACPI0004,PNP0A05 and PNP0A06 Container Driver (EXPERIMENTAL)(n)

; 我靠,这强!支持 CPU,内存的热插拔.不过我怕触电,不选.

APM (Advanced Power Management) BIOS Support--->

<M> APM (Advanced Power Management) BIOS support(m)

; 高级电源管理 BIOS 支持,一般是笔记本用(或使用电池的系统,不知道用 UPS 的算不算,应该不算).台式机不用选择.

[] Ignore USER SUSPEND (n)

; 不选择,否则会忽略用户的挂机请求.如果你不幸用了 NEC 的笔记本,必须选 Y,因为有 BUG.(抵

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

制日货!)

[] Enable PM at boot time(n)

; 开机时允许 PM,一般不用选择(可能在启动时宕机)

[] Make CPU Idle calls when idle (n)

; CPU 空闲时调用 CPU idle 进程.还是不选吧...可能在启动或空闲时宕机.

[] Enable console blanking using APM(n)

; 当 Linux 虚拟控制台关闭显示(黑屏)时,关闭 LCD 背光.都黑屏了,还关背光搞莫事撒.

[] RTC stores time in GMT (n)

; RTC: Real Time Clock. GMT: Greenwich Mean Time

推荐将 GMT 时间存储到 RTC 中以,但如果装了别的不能够识别 GMT 的系统,不要选择.(比如 windows)

[] Allow interrupts during APM BIOS calls(n)

; 一般是不选择的,但如果挂起系统时出现了宕机,可将它选择试试. 调用 BIOS 是开中断是不良少年的行为!

[] Use real mode APM BIOS call to power off(n)

; 针对某些带 bug 的 BIOS 的补救措施:如果系统不能自己断电,选上它.

CPU Frequency scaling--->

[*] CPU Frequency scaling

; 动态调节 CPU 频率以节电.有人提到频率降低,影响了处理速度,导致 deadline 问题.留意一下.

[] Enable CPUfreq debugging

; 我是个懒人,所有调试的都不打开

<M> CPU frequency translation statistics

; 通过 sysfs 文件系统输出 CPU 频率信息.

[*] CPU frequency translation statistics details

; 显示上述的详细的 CPU 频率信息.

Default CPUFreq governor (userspace)--->

; 默认选择动态调整 CPU 频率.

<M> 'performance' governor

; performance 将 CPU 频率设定在支持的最高频率,而不动态调节.

<M> 'powersave' governor

; 将 CPU 频率设置为最低

<M> 'ondemand' cpufreq policy governor

; 快速动态调整 CPU 频率, Pentium M 的 CPU 可以使用

<M> 'conservative' cpufreq governor

; 与 ondemand 不同,平滑地调整 CPU 频率,适合于用电池工作时.

CPUFreq processor drivers

< > ACPI Processor P-States driver

不选

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

< > AMD Mobile K6-2/K6-3 PowerNow!

< > AMD Mobile Athlon/Duron PowerNow!

< > AMD Opteron/Athlon64 PowerNow!

AMD 的移动处理器省电技术

< > Cyrix MediaGX/NatSemi Geode Suspend Modulation

<M> Intel Enhanced SpeedStep

; intel 的 SpeedStep 技术.可以让处理器在 2 种工作模式之间随意地切换,即通电状态时的最高性能模式 (Maximum Performance Mode) 和电池状态时的电池优化模式 Battery Optimized Mode

[*] Use ACPI tables to decode valid frequency/voltage pairs

[] Built-in tables for Banias CPUs

;如果选择了 Speedstep,就选上 Use ACPI tables.

< > Intel Speedstep on ICH-M chipsets (ioport interface)

< > Intel SpeedStep on 440BX/ZX/MX chipsets (SMI interface)

< > Intel Pentium 4 clock modulation

< > nVidia nForce2 FSB changing

; 上述选项根据具体 CPU,显卡类型选择.

Bus options (PCI, PCMCIA, EISA, MCA, ISA)--->

; I/O 总线选项,由 Interl 在 1992 年初制订.现在一般的总线类型都是 PCI.使用 PCI 总线的系统于 1993 年中期出现,从此成为主流.

--- PCI support

PCI access mode (Any)--->

; PCI 访问模式,选 Any

[*] PCI Express support

; 自动打开 PCI 快速总线支持,选上.

< > PCI Express Hotplug driver

PCI 快速热拔插驱动,不选择.

[] Use polling mechanism for hot-plug events (for testing purpose)

; 对热拔插时间使用轮询,用于早期的实验系统,不选

[] Message Signaled Interrupts (MSI and MSI-X)

; 使用 MSI(Message Signaled Interrupts),当中断产生时,使用 inbound memory 写 PCI 总线,而不断言设备的 IRQ 引脚.不选.

[] Legacy /proc/pci interface

; 使用/proc/pci提供系统中的PCI设备信息.实际上,使用 lspci(8)能提供相同甚至更多的信息.发行版都装有 lspci 包.可不选.

[] PCI Debugging

; PCI 调试,不选.

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

[*] ISA support

; ISA 总线支持, 运行 `lspci | grep ISA`, 若有, 则选上.

[] MCA support

; MCA (microChannel Architecture) 总线是 IBM 为解决快速微处理器和相对慢的 ISA 总线之间的差异而开发的一种总线结构, 并被用于 IBM 的 PS/2 系统, 目前 MCA 总线主要用于 IBM PS/2 计算机和一些笔记本中. 不选.

< > NatSemi SCx200 support

; 支持 National Semiconductor 的 SCx200 处理器的, 不选.

PCCARD (PCMCIA/CardBus) support--->

; 这些选项都是本本用的, 台式机不选

<M> PCCard (PCMCIA/CardBus) support

; 笔记本选为 module

[] Enable PCCARD debugging

; PCMCIA 的调试, 不选.

<M> 16-bit PCMCIA support

; 16 位 PCMCIA 支持, 还是编译为模块吧, `lspci -v | grep 16` 发现还是有 16 位的设备, 不知道什么意思, 还是谨慎为好.

[*] Load CIS updates from userspace (EXPERIMENTAL)

; 有的 PCMCIA 卡需要这个功能, 选上.

[*] PCMCIA control ioctl (obsolete)

; 提供 PCMCIA 的 ioctl 接口. 不懂什么意思, 帮助文档建议选上.

[*] 32-bit CardBus support

; 早前的 PCMCIA 卡是 16 位的, 这里允许使用 32 位的 CardBus, 新的 PC-card 实际上是 CarBus 卡, 选上.

--- PC-card bridges

<M> CardBus yenta-compatible bridge support

<M> Cirrus PD6729 compatible bridge support

<M> i82092 compatible bridge support

; 对于什么鸟桥的支持? 不懂, 选为模块.

PCI Hotplug Support--->

; 支持 PCI 的热拔插, 怕触电, 都不选

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

3.7 Executable file formats--->

[*] Kernel support for ELF binaries

; 支持 ELF 可执行文件格式,一定要选上!

<M> Kernel support for a.out and ECOFF binaries

; .out 的执行文件是比较古老的可执行码,用在比较早期的 UNIX 系统上. Linux 最初也是使用这种码来执行程序,一直到 ELF 格式的可执行码出来后,有愈来愈多的程序码随着 ELF 格式的优点而变成了 ELF 的可执码.将来势必完全取代 a.out 格式的可执行码.目前还有一些.out 格式的代码.选为模块.

<M> Kernel support for MISC binaries

; 支援别的种类的 binary 执行档(如: Java、Python ... etc).编译为模块.

3.8 Networking--->

--- Networking support

Networking options--->

<M> Packet socket

; 类似于 tcpdump 的应用程序会绕过 IP 层直接访问网络设备(原始 socket),选为模块.

[*] Packet socket: mmaped IO

; mmaped IO 让传输加速,要开启 MapleBBS 内的 MMIO 也需要它,选上.

<M> Unix domain sockets

; syslogd、x-windows 等都是用 socket 来传输,即便电脑没网络,选为模块.

<M> IPsec user configuration interface

; 支持 IPsec 用户设置接口,选为模块.

<M> PF_KEY sockets

; 于 IPsec 有关,编译为模块.

[*] TCP/IP networking

; 支持 TCP/IP,当然要选上.

[] IP: multicasting

; 支持 IP 多播,一般用于 MBONE(因特网上的音频、视频多播).

[] IP: advanced router

; 用于路由器的选项,不选.

[] IP: kernel level autoconfiguration

; 内核启动时自动配置 IP 地址,之用于无盘系统,不选.

< > IP: tunneling

; IP 隧道,在多个网络中移动不需改变 IP 地址,不用选.

< >IP: GRE tunnels over IP

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

; GRE(Generic Routing Encapsulation)隧道技术,允许在现有的 IPv4 架构上封包 IPv6.不选吧.

[] IP: multicast routing

; 不做服务器,不用选择.

[] IP: ARP daemon support (EXPERIMENTAL)

; 将 ARP 缓存在内核中,不选.

[*] IP: TCP syncookie support (disabled per default)

; 防止 SYN flooding 攻击.如果选择,SYN cookies 默认不会开启.

<M>IP: AH transformation

<M>IP: ESP transformation

<M>IP: IPComp transformation

; 以上 3 个都是 IPsec 需要的,编为模块.

<M>IP: tunnel transformation

; 支持通用 IP 隧道传输.编为模块.

<*>INET: socket monitoring interface

; 支持 socket 监听接口,选上.

[]TCP: advanced congestion control

; TCP 高级拥塞控制,可不选择.

IP: Virtual Server Configuration--->

Virtual Server Configuration 中的选项设定都用 M, 按默认负载均衡集群是在应用服务器高负载的情况下, 由多台节点提供可伸缩的, 高负载的服务器组以保证对外提供良好的服务响应; 而 LVS 就是实现这一功能的技术, 它通过使内核支持 ipvs 来实现 LVS/Direct Routing (DR)、LVS/IP Tunnel、LVS/NAT 的功能.

< > IP virtual server support (EXPERIMENTAL)

; 集群或多台服务器用的,不选择,以后好好研究...虚拟服务器...

<M> The IPv6 protocol

; 支持 IPv6,编译为模块. IPv6 有空再看,现按照帮助说明的来,该选的选,该编为模块的编为模块.

...

后面的再慢慢看吧...按 menuconfig 中的帮助文档来,该不选的绝不选.

[]Amateur Radio support--->

; 无线电设备支持,不选.

< > IrDA (infrared) subsystem support--->

; 红外设备支持,不选.

< > Bluetooth subsystem support--->

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

; 蓝牙设备支持,不选.
<M> Generic IEEE 802.11 Networking Stack
; 802.11 网络栈支持,编译为模块.

3.9 Device Drivers--->

Generic Driver Options--->

[*] Select only drivers that don't need compile-time external firmware
; 只选择不需要 compile-time 外部 firmware,没搞懂,选上.
[*] Prevent firmware from being built
; 禁止编译 firmware, firmware 一般与硬件一起绑定,只在更新时才需要重新编译,选上.
<M> Hotplug firmware loading support
; 热插拔固件加载,没懂,编为模块.
[] Driver Core verbose debug messages
; 不选.

Connector - unified userspace <-> kernelspace linker--->

< > Connector - unified userspace <-> kernelspace linker
; 支持基于 netlink socket 协议的用户空间与内核空间的连接.

Memory Technology Devices (MTD)--->

< > Memory Technology Device (MTD) support
; 支持 MTD 设备(flash, ram 等芯片).一般用于嵌入式系统,不选

Parallel port support--->

< > Parallel port support
; 并口支持.没有打印机,先不选.(选前两项,以防用到并口,比如 LDD3 中的例子)

Plug and Play support--->

[*] Plug and Play support
; 支持既插即用设备,选上.并选上该选项下面的一些协议支持, /proc 接口不选.

Block devices--->

< > Normal floppy disk support
; 软盘,没人用了,不选.
< > XT hard disk support
; 石器时代的 8 位硬盘,不选.
< > Compaq SMART2 support

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

< > Compaq Smart Array 5xxx support

; compaq 用的,不选.

<M> Loopback device support

; 大部分的人这个选项都选 N, 因为没有必要。但是如果你要 mount iso 文件的话, 你得选上 Y。这个选项的意思是说, 可以将一个文件挂成一个文件系统。如果要烧光盘片的, 那么您很有可能在把一个文件烧进去之前, 看看这个文件是否符合 ISO9660 的文件系统的内容, 是否符合您的需求。而且, 可以对这个文件系统加以保护。不过, 如果您想做到这点的话, 您必须有最新的 mount 程序, 版本是在 2.5X 版以上的。而且如果您希望对这个文件系统加上保护, 则您必须有 des.1.tar.gz 这个程序。注意: 此处与网络无关。建议编译成模块

< > Cryptoloop Support

; 不用选

<M> Network block device support

; 使本机成为网络块设备的客户机.将主机的分区挂载到本地?...先编译成模块看.

< > Promise SATA SX8 support

; SATA 接口的 16 位 I/O CPU 支持, 不用选.

< > Low Performance USB Block driver

;不选,否则可能与 USB 存储驱动冲突.

< > RAM disk support

; 把内存当作块设备使用,一般用于在最初安装 Linux 时从软盘向 RAM 中复制最小根文件系统.不选.

<M> Packet writing on CD/DVD media

; 支持刻录机的 packet writing. ? 编为模块.

(8) Free buffers for data gathering

; 设置刻录时同步的 packet 数.多的 packet 能增加刻录速度,但耗费更多内存.(一个约 64K),默认为 8.

[] Enable write caching

; 写缓冲.不要选.如果刻录盘是坏的,系统不会处理延迟的写错误.

<M> ATA over Ethernet support

; ATA 什么意思?编为模块吧

ATA/ATAPI/MFM/RLL support--->

<M> ATA/ATAPI/MFM/RLL support

; 支持 ATA/ IDE/ATAPI 设备,除非你的系统是纯 SCSI 的,否则一定要选上!

<M> Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support

; 如果有多个 IDE 设备,一定选上

[] Support for SATA (deprecated; conflicts with libata SATA driver)

[] Use old disk-only dri

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号

4 如何建立 yaffs 文件系统映像

先将文件系统压缩包 yaffs.tar.bz2（不同开发板厂商文件名字有所不同）拷贝到某个目录下，进入这个目录，然后解压这个压缩包，命令：

```
tar -jxvf yaffs.tar.bz2
```

解压后得到 yaffs 目录，文件系统的所有文件都在该目录下，可根据需要修改。

最后用 mkyaffs2image 工具来制作文件系统：

```
./mkyaffs2image yaffs test.yaffs 0
```

回车以后，少等片刻，一个 yaffs 的映像文件 test.yaffs 在当前目录下生成了。

Linux 外设驱动程序以及用户程序的编写

1、Hello world

首先我们在 PC 机 Linux 环境下写一个简单的应用程序：Hello World

在 PC 命令行执行如下命令：

```
#vi helloworld.c
```

在弹出的界面按 PC 机上 “Insert”，编写如下代码：

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
printf( " Hello World !\n " )
```

```
return 0;
```

```
}
```

```
}
```

编写完毕之后，按 PC 机的 “ESC” 键，然后是 “w” 和 “q” 键（注意是小写的，意思是保存退出），保存为 helloworld.c。

在 PC 机 Linux 命令行下进入 helloworld.c 文件所在的目录，输入下面的命令进行交叉编译：

```
arm-linux-gcc -o helloworld helloworld.c
```

编译成功后可看到新生成的 helloworld 文件，传到开发板就可以直接运行了。

2、编写第一个驱动

下面先编写一个简单的 Hello 驱动，在内核源码的 “drivers/char” 目录下创建一个名为 “MY-hello.c” 的文件，内容如下：

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
#include <linux/init.h>
#include <linux/module.h>
MODULE_LICENSE("Dual BSD/GPL");
static int hello_init(void)
{
    printk(KERN_ALERT "Hello, world\n");
    return 0;
}
static void hello_exit(void)
{
    printk(KERN_ALERT "Goodbye, cruel world\n");
}
module_init(hello_init);
module_exit(hello_exit);
```

然后修改同一目录下的“Kconfig”文件，在合适的位置添加如下内容：

```
config MY-HELLO
    tristate "S3C2440 Hello Driver"
    depends on ARCH_S3C2440
    help
        Hello on S3C2440
```

修改同一目录下的“Makefile”文件，添加如下内容：

```
Obj-$(CONFIG_MY_HELLO) += My_hello.o
```

保持完毕后，输入#make menuconfig，然后进行配置：

```
Device Driver  ->
    Character devices  ->
        <M> S3C2440 Hello Driver
```

将其选择为“M”（模块），然后保存配置，编译出内核镜像烧写到开发板中。

然后使用命令# make SUBDIR=drivers/char/ modules，然后编译出驱动模块，在内核目录下面的“drivers/char”里面，名为 My_hello.ko，将其复制到开发板中，然后加载该驱动模块和卸载该驱动模块。

3、GPIO 驱动

下面将用 GPIO 去控制 LED 的亮与灭来详细说明驱动编写。

杭州茂葳科技©所有

电话：13758237754

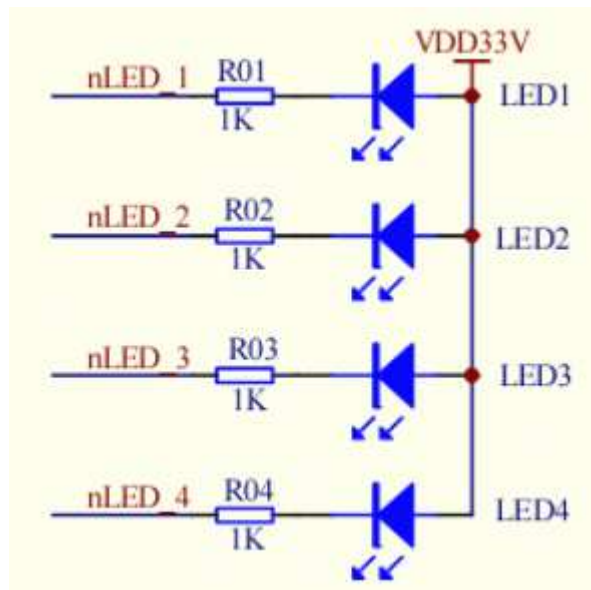
网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

3.1、硬件分析

我们用开发板的 GPB5、GPB6、GPB7、GPB8 来控制 4 个 LED 灯，如下图所示：



根据上图可以知道，当 CPU 的 GPB5 到 GPB8 是低电平时，LED 灯亮；当为高电平时 LED 灯灭。现在在驱动中实现对 GPB 口的电平控制来实现对灯进行开闭操作。

3.2 LED 驱动的编写

同样在内核源码“drivers/char”目录下创建一个名为“S3C2440_leds.c”的文件，内容如下：

```
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/fs.h>
#include <linux/init.h>
#include <linux/delay.h>
#include <asm/irq.h>
#include <asm/arch/regs-gpio.h>
#include <asm/hardware.h>

#define DEVICE_NAME    "leds" /* 加载模式后，执行" cat /proc/devices" 命令看到的设备名称 */
#define LED_MAJOR      231    /* 主设备号 */
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
/* 应用程序执行 ioctl(fd, cmd, arg) 时的第 2 个参数 */
#define IOCTL_LED_ON    0
#define IOCTL_LED_OFF   1

/* 用来指定 LED 所用的 GPIO 引脚 */
static unsigned long led_table [] = {
    S3C2410_GPB5,
    S3C2410_GPB6,
    S3C2410_GPB7,
    S3C2410_GPB8,
};

/* 用来指定 GPIO 引脚的功能：输出 */
static unsigned int led_cfg_table [] = {
    S3C2410_GPB5_OUTP,
    S3C2410_GPB6_OUTP,
    S3C2410_GPB7_OUTP,
    S3C2410_GPB8_OUTP,
};

/*应用程序对设备文件/dev/leds 执行 open()时，
*就会调用 s3c24xx_leds_open
*/
static int s3c24xx_leds_open(struct inode *inode, struct file *file)
{
    int i;

    for(i=0; i<4; i++) {
        s3c2410_gpio_cfpin(led_table[i], led_cfg_table[i])
    }

    return 0;
}

/*应用程序对设备文件/dev/leds 执行 ioctl()时，
*就会调用 s3c24xx_leds_ioctl
*/
static int s3c24xx_leds_ioctl(struct inode *inode, struct file *file, unsigned int
cmd, unsigned long arg)
{
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号


```
    if (arg > 4) {
        return -EINVAL;
    }

    switch(cmd) {
        case IOCTL_LED_ON:
            s3c2410_gpio_setpin(led_table[arg], 0);
            return 0;
        case IOCTL_LED_OFF:
            s3c2410_gpio_setpin(led_table[arg], 1);
            return 0;
        default:
            return -EINVAL;
    }
}

/*这个结构是字符设备驱动程序的核心
*当应用程序操作设备文件时调用的 open、read 等函数，
*最终会调用这个结构中指定的对应函数
static struct file_operations s3c24xx_leds_fops = {
    .owner = THIS_MODULE,
    .open  = s3c24xx_leds_open,
    .ioctl = s3c24xx_leds_ioctl
};

/*模块的初始化函数*/
static int __init s3c24xx_leds_init(void)
{
    int ret;

    ret = register_chrdev(LED_MAJOR, DEVICE_NAME, &s3c24xx_leds_fops);
    if(ret < 0) {
        printk(DEVICE_NAME "can't register major number\n");
        return ret;
    }

    printk(DEVICE_NAME "initialized\n");
    return 0;
}
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
/*模块的撤销函数*/
static void __exit s3c24xx_leds_exit(void)
{
    unregister_chrdev(LED_MAJOR, DEVICE_NAME);
}

/*指定驱动程序的初始化函数和卸载函数*/
module_init(s3c24xx_leds_init);
module_exit(s3c24xx_leds_exit);

/*加入描述信息*/
MODULE_AUTHOR("ckz");
MODULE_DESCRIPTION("S3C2440 LED Driver");
MODULE_LICENSE("GPL");
```

3.3 在内核源码中添加对 LED 驱动的支持

修改同目录下的“Kconfig”文件，在合适的地方添加如下内容：

```
Config S3C2440_LEDS
    tristate "S3C2440 LEDS Driver"
    depends on ARCH_S3C2440
    help
        LEDS on S3C2440
```

然后再通目录下修改“Makefile”，添加如下内容：

```
Obj-$(CONFIG_S3C2440_LEDS) += S3C2440_leds.o
```

添加完成以上内容之后，输入#make menuconfig，然后配置如下

```
Device Drivers  ->
    Character devices  ->
        <M> S3C2440 LEDS Driver
```

将其选择为“M”，然后保存配置，编译出内核镜像烧写到开发板中。

然后再使用命令#make SUBDIR=drivers/char modules，编译出驱动模块，在内核目录下的“drivers/char”下面，名为 S3C2440_leds.Ko，将其复制到开发板中。

也可将其配置为“*”（添加到内核中），然后编译出镜像，烧写到开发板中，LED 灯就可以进行控制了。

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

附录一 Linux 命令详解

Linux 是一个真正的多用户操作系统，它可以同时接受多个用户登录。Linux 还允许一个用户进行多次登录，这是因为 Linux 和 UNIX 一样，提供了虚拟控制台的访问方式，允许用户在同一时间从控制台进行多次登录。虚拟控制台的选择可以通过按下 Alt 键和一个功能键来实现，通常使用 F1 - F6 例如，用户登录后，按一下 Alt - F2 键，用户又可以看到 "login:" 提示符，说明用户看到了第二个虚拟控制台。然后只需按 Alt - F1 键，就可以回到第一个虚拟控制台。一个新安装的 Linux 系统默认允许用户使用 Alt - F1 到 Alt - F6 键来访问前六个虚拟控制台。虚拟控制台可使用户同时在多个控制台上工作，真正体现 Linux 系统多用户的特性。用户可以在某一虚拟控制台上进行的工作尚未结束时，切换到另一虚拟控制台开始另一项工作。当然我们也可以在 KDE 环境下使用终端方式输入命令。常见的命令如下：

文件列表 - ls

ls # 以默认方式显示当前目录文件列表
ls - a # 显示所有文件包括隐藏文件
ls - l # 显示文件属性，包括大小，日期，符号连接，是否可读写及是否可执行
ls - -color=never *.so > obj # 不显示文字颜色，将所有 so 文件记录到 obj 文件中

目录切换 - cd

cd dir # 切换到当前目录下的 dir 目录
cd / # 切换到根目录
cd .. # 切换到上一级目录
cd ../.. # 切换到上二级目录
cd ~ # 切换到用户目录，比如是 root 用户，则切换到 /root

删除 - rm

rm file # 删除某一个文件
rm - fr dir # 删除当前目录下叫 dir 的整个目录

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

复制 - cp

cp source target # 将文件 source 复制为 target
cp /root/source . # 将 /root 下的文件 source 复制到当前目录
cp - av soure_dir target_dir # 将整个目录复制，两目录完全一样
cp - fr source_dir target_dir # 将整个目录复制，并且是以非链接方式复制，当
sourc e
目录带有符号链接时，两个目录不相同

移动 - mv

mv source target # 将文件 source 更名为 target

比较 - diff

diff dir1 dir2 # 比较目录 1 与目录 2 的文件列表是否相同，但不比较文件的实际内
容，
不同则列出
diff file1 file2 # 比较文件 1 与文件 2 的内容是否相同，如果是文本格式的文件，则
将
不相同的内容显示，如果是二进制代码则只表示两个文件是不同的
comm file1 file2 # 比较文件，显示两个文件不相同的内容

回显 - echo

echo message # 显示一串字符
echo “ essage message2 ” # 显示不连续的字符串

文件内容查看 - cat

cat file # 显示文件的内容，和 DOS 的 type 相同
cat file | more # 显示文件的内容并传输到 more 程序实现分页显示，使用命令
less fil e
可实现相同的功能
more # 分页命令，一般通过管道将内容传给它，如 ls | more
设置环境变量 - export

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

```
export LC_ALL=zh_CN.GB2312 # 将环境变量 LC_ALL 的值设为 zh_CN.GB2312
export DISPLAY=0:0 # 通过该设置，当前字符终端下运行的图形程序可直接运行于
Xserve r
```

时间日期 - date

```
date # 显示当前日期时间
date - s 20:30:30 # 设置系统时间为 20:30:30
date - s 2002 - 3 - 5 # 设置系统时期为 2003 - 3 - 5
clock - r # 对系统 Bios 中读取时间参数
clock - w # 将系统时间 ( 如由 date 设置的时间 ) 写入 Bios
```

容量查看 - du

```
du # 计算当前目录的容量
du - sm /root # 计算 /root 目录的容量并以 M 为单位
```

查找 - find

```
find - name /path file # 在 /path 目录下查找看是否有文件 file
```

搜索 - grep

```
grep - ir “ chars ” # 在当前目录的所有文件查找字符串 chars ，并忽略大小写，
- i 为大
小写， - r 为下一级目录
```

编辑 - vi

```
vi file # 编辑文件 file
vi 原基本使用及命令：
输入命令的方式为先按 ctrl+c ，然后输入 :x( 退出 ),:x!( 退出并保存 ):w( 写入文
件 ),:w!( 不
询问方式写入文件)， :r file( 读文件 file) ， :%s/oldchars/newchars/g( 将所有字
串 oldchar s
换成 newchars) 这一类的命令进行操作
```

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

读取 - **man**

`man ls` # 读取关于 `ls` 命令的帮助
`man ls | grep color` # 读取关于 `ls` 命令的帮助并通过 `grep` 程序在其中查找 `color` 字符串

重启 - **reboot**

`reboot` # 重新启动计算机

关机 - **halt**

`halt` # 关闭计算机

压缩与解压 - **tar**

`tar xfv file.tar.gz` # 将文件 `file.tar.gz` 解压
`tar xfv file.tar.gz - C target_path` # 将文件 `file.tar.gz` 解压到 `target_path` 目录下
`tar czv file.tar.gz source_path` # 将文件 `source_path` 压缩为 `file.tar.gz`
`tar c directory > directory.tar` # 将目录 `directory` 打包成不压缩的 `directory.tar`
`gzip directory.tar` # 将覆盖原文件生成压缩的 `directory.tar.gz`
`gunzip directory.tar.gz` # 覆盖原文件解压生成不压缩的 `directory.tar`。
`tar xf directory.tar` # 可将不压缩的文件解包

权限设置 - **chmod**

`chmod a+x file` # 将 `file` 文件设置为可执行，脚本类文件一定要这样设置一个，否则得用

`bash file` 才能执行

`chmod 666 file` # 将文件 `file` 设置为可读写

`chown user /dir` # 将 `/dir` 目录设置为 `user` 所有

网卡配置 - **ifconfig**

`ifconfig eth0 192.168.1.1 netmask 255.255.255.0` # 设置网卡 `1` 的地址 `192.168.1.1`，

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

掩码为 255.255.255.0，不写 netmask 参数则默认为 255.255.255.0
ifconfig eth0:1 192.168.1.2 # 捆绑网卡 1 的第二个地址为 192.168.1.2
ifconfig eth0:x 192.168.1.x # 捆绑网卡 1 的第二个地址为 192.168.1.x
ifconfig down eth1 # 关闭第二块网卡，使其停止工作

创建设备 - mknod

mknod /dev/hda1 b 3 1 # 创建块设备 hda1，主设备号为 3，从设备号为 1，
即 master 硬盘
的的第一个分区
mknod /dev/tty1 c 4 1 # 创建字符设备 tty1，主设备号为 4，从设备号为 1，即
第一个 tty
终端

装载模块 - insmod

insmod rtl8139.o # 装载驱动程序 rtl8139.o
insmod sb.o io=0x280 irq=7 dma=3 dma16=7 mpu_io=330 # 装载驱动程序并设置相关的
irq,dma 参数

删除模块 - rmmod

rmmod rtl8139 # 删除名为 rtl8139 的驱动模块

挂载 - mount

mount - t ext2 /dev/hda1 /mnt # 把 /dev/hda1 装载到 /mnt 目录
mount - t iso9660 /dev/cdrom /mnt/cdrom # 将光驱加载到 /mnt/cdrom 目录
mount - t smb //192.168.1.5/sharedir /mnt - o username=fangtan,password=fangtan # 将 windows 的共享目录加载到 /mnt/smb 目录，用户名及密码均为 fangtan
mount - t nfs 192.168.1.1:/sharedir /mnt # 将 nfs 服务的共享目录 sharedir 加载到
/mnt/nfs 目录

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

卸载 - umount

umount /mnt # 将 /mnt 目录卸载， /mnt 目录必须处于空闲状态
umount /dev/hda1 # 将 /dev/hda1 设备卸载，设备必须处于空闲状态
进程查看 - ps
ps # 显示当前系统进程信息
ps - ef # 显示系统所有进程信息
杀死进程 - kill
kill - 9 500 # 将进程编号为 500 的程序杀死

关于我们

杭州茂葳科技是一家专注于无线通信领域，集研发、生产、销售、服务于一体的高科技企业，与杭州电子科技大学电路与系统重点实验室、浙江大规模集成电路重点实验室、信息产业部系统集成芯片（SOC）重点实验室保持良好的合作关系。

公司为各大高校提供无线传感网络、物联网实验室建设并成功应用于重庆大学、天津医科大学。

主要产品有基于 315MHz、433MHz、868MHz、915MHz、2.4GHz 的无线数字通信模块/无线模块/无线通信模块/无线收发模块/无线数传模块/无线数据传输模块/无线开发套件 ARM 开发平台/视频采集卡，产品广泛应用于工业控制、安防领域、有源 RFID 系统、无源超高频读写器系统。

茂葳科技在无线通信、嵌入式开发领域有着丰富的开发经验，提供研发项目的技术支持、技术方案等的定制服务和项目管理咨询，帮

杭州茂葳科技©所有

电话：13758237754

网址：www.moreway.net

Email：xiazhaojiandiyi@163.com

地址：杭州下沙经济技术开发区学林街 608 号

助客户避免开发风险，加快研发进度。

欢迎行业同仁前来访问和洽谈项目合作。

目前主要专注领域有：

- 1、基于 315M/433M/868 的无线模块 MW905、MW1100 系列
- 2、基于 2.4G 的无线模块 MW24L01 系列
- 3、针对大功率远程开发的 MW905-1000（功率可达 1W）
- 4、USB 接口的无线数传模块 MWUSB-905、MWUSB-1100、MWUSB24L01 系列
- 5、无线开发套件
- 6、无线 485 系列开发套件
- 7、嵌入式平台的开发与应用

联系方式

电话:13758237754

QQ:253816584

地 址:杭州下沙经济技术开发区学林街 608 号

邮 编:310018

邮 箱: xiazhaojiandiyi@163.com

网 址: <http://www.moreway.net>

杭州茂葳科技©所有

电话: 13758237754

网址: www.moreway.net

Email: xiazhaojiandiyi@163.com

地址: 杭州下沙经济技术开发区学林街 608 号