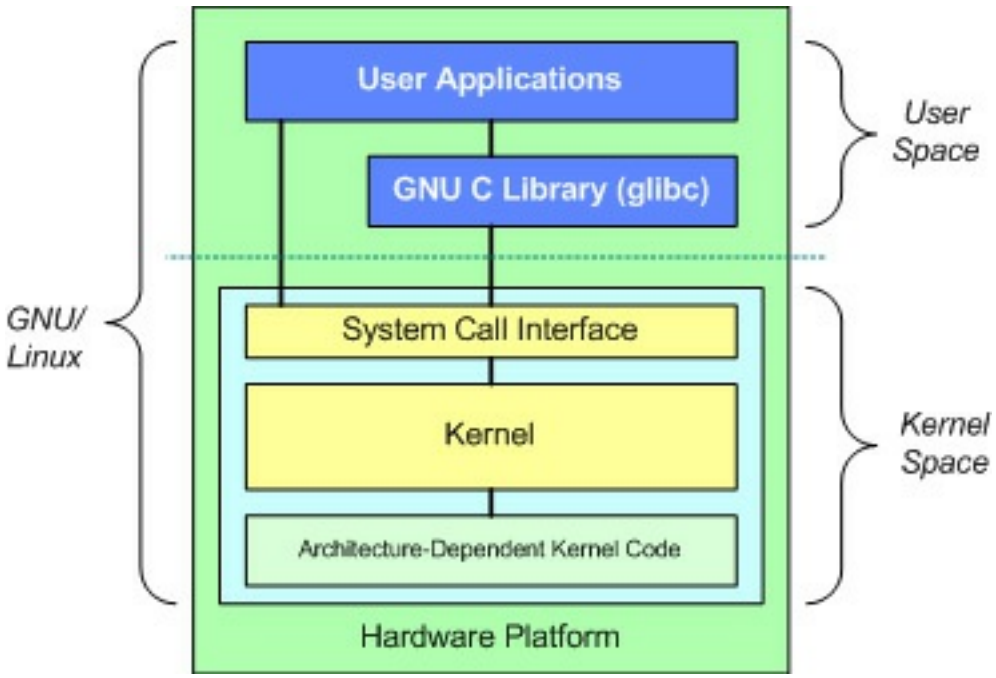
嵌入式系统 - 第六章

> *本章介绍Linux内核及文件系统*

Linux 内核（Kernel）

Linux内核是整个Linux系统的灵魂，Linux内核负责整个系统的内存管理，进程调度和文件管理。Linux内核的容量并不大，并且大小可以裁减，这个特性对于嵌入式是非常有好处的。一般一个功能比较全面的内核也不会超过1M。合理的配置Linux内核是嵌入式开发中很重要的一步，对内核的充分了解是嵌入式Linux开发的基本功。

内核结构



GNU C Library提供了连接内核的系统调用接口，还提供了在用户空间应用程序和内核之间进行转换的机制。这点非常重要，因为内核和用户空间的应用程序使用的是不同的保护地址空间。每个用户空间的进程都使用自己的虚拟地址空间，而内核则占用单独的地址空间。

Linux 内核可分为3层。最上面是**系统调用接口**（System Call Interface），它实现了一些基本的功能，例如 read 和 write。系统调用接口之下是**内核代码**（Kernel），可以更精确地定义为独立于体系结构的内核代码。这些代码是 Linux 所支持的所有处理器体系结构所通用的。在这些代码之下是**依赖于体系结构的代码**（Architecture-Dependent Kernel Code），构成了通常称为**BSP（Board Support Package）**的部分。这些代码用作给定体系结构的处理器和特定于平台的代码。

Linux内核源码顶层目录说明

目录名	说明
arch/	体系结构相关的代码，如arch/i386、arch/arm、arch/ppc
drivers/	各种设备驱动程序，例如，drivers/char、drivers/block、…
Documentation/	内核文档
fs/	文件系统，例如，fs/ext3、fs/jffs2、…
include/	内核头文件：include/asm是体系结构相关的头文件，它是include/asm-arm、include/asm-i386等目录的链接。include/Linux是Linux内核基本的头文件。
init/	Linux初始化，如main.c
ipc/	进程间通信的代码
kernel/	Linux内核核心代码（这部分很小）
lib/	各种库子程序，如zlib、crc32
mm/	内存管理代码
net/	网络支持代码，主要是网络协议

sound/	声音驱动的支持
scripts/	内部或者外部使用的脚本
usr/	用户的代码

内核功能

Linux内核的功能大致分成如下几个部分：

1．进程管理：进程管理功能负责创建和撤销进程以及处理他们和外部世界的连接。不同进程之间的通信是整个系统的基本功能，因此也由内核处理。除此之外，控制进程如何共享CPU资源的调度程序也是进程管理的一部分。概括的说，内核的进程管理活动就是在单个或多个CPU上实现多进程的抽象。

2．内存管理：内存是计算机的主要资源之一，用来管理内存的策略是决定系统性能的一个关键因素。内核在有限的可用资源上为每个进程都创建了一个虚拟寻址空间。内核的不同部分在和内存管理子系统交互时使用一套相同的系统调用，包括从简单的malloc/free到其他一些不常用的系统调用。

3．文件系统：Linux在很大程度上依赖于文件系统的概念，Linux中的每个对象几乎都是可以被视为文件。内核在没有结构硬件上构造结构化的文件系统。所构造的文件系统在整个系统中广泛使用。另外，Linux支持多种文件系统类型，即在物理介质上组织的结构不同。

4．设备控制：几乎每个系统操作最终都会映射到物理设备上。除了处理器，内存以及其他很有限的的几个实体外，所有的设备控制操作都由与被控制设备相关的代码完成。这段代码叫做设备驱动程序，内核必须为系统中的每件外设嵌入相应的驱动程序。

5．网络功能：网络功能也必须有操作系统来管理，因为大部分网络操作都和具体的进程无关。在每个进程处理这些数据之前，数据报必须已经被收集、标识、和分发。系统负责在应用程序和网络之间传递数据。另外，所有的路由和地址解析问题

都由内核处理。

内核配置

Linux内核的配置系统由三个部分组成，分别是：

- **Makefile**：分布在 Linux 内核源代码中的 Makefile，定义 Linux 内核的编译规则；
- **配置文件（`config.in`）**：给用户提供配置选择的功能；
- **配置工具**：包括配置命令解释器（对配置脚本中使用的配置命令进行解释）和配置用户界面（提供基于字符界面、基于 Ncurses 图形界面以及基于 Xwindows 图形界面的用户配置界面，各自对应于 Make config、Make menuconfig 和 make xconfig）

Makefile的作用是根据配置的情况，构造出需要编译的源文件列表，然后分别编译，并把目标代码链接到一起，最终形成 Linux 内核二进制文件。
由于 Linux 内核源代码是按照树形结构组织的，所以 Makefile 也被分布在目录树中。

Linux 内核中 Makefile 及与 Makefile 直接相关的文件有：
Makefile：顶层 Makefile，是整个内核配置、编译的总体控制文件。
.config：内核配置文件，包含由用户选择的配置选项，用来存放内核配置后的结果。
arch/*/Makefile：位于各种 CPU 体系目录下的 Makefile，如 arch/arm/Makefile，是针对特定平台的 Makefile。
各个子目录下的 Makefile：比如 drivers/Makefile，负责所在子目录下源代码的管理。
Rules.make：规则文件，被所有的 Makefile 使用。

用户通过 make config 配置后，产生了 .config。顶层 Makefile 读入 .config 中的配置选择。
顶层 Makefile 有两个主要的任务：产生 vmlinux 文件和内核模块（module）。为了达到此目的，顶层 Makefile 递归的进入到内核的各个子目录中，分别调用位于这些子目录中的 Makefile。至于到底进入哪些子目录，取决于内核的配置。在顶层 Makefile 中，有一句：include arch/$(ARCH)/Makefile，包含了特定 CPU 体系结构下的 Makefile，这个 Makefile 中包含了平台相关的信息。

内核的模块式结构

内核模块是Linux内核向外部提供的一个接口，其全称为动态可加载内核模块(Loadable Kernel Module，LKM)，简称为模块。Linux内核之所以提供模块机制，是因为它本身是一个单内核(monolithic kernel)。单内核的最大优点是效率高，因为所有的内容都集成在一起，但其缺点是可扩展性和可维护性相对较差，模块机制就是为了弥补这一缺陷。

模块：模块是具有独立功能的程序，它可以被单独编译，但不能独立运行。它在运行时被链接到内核作为内核的一部分在内核空间运行，这与运行在用户空间的进程是不同的。
模块通常由一组函数和数据结构组成，用来实现一种文件系统、一个驱动程序或内核中其他上层的功能。

应用程序与内核模块的比较

	C语言应用程序	**内核模块程序**
使用函数	libc库	内核函数
运行空间	用户空间	内核空间

运行权限	普通用户	超级用户
入口函数	main()	init_module()
出口函数	exit()	cleanup_module()
编译	gcc -c	gcc –c –D __KERNEl__ _DMOKULE
连接	gcc	gcc
运行	直接运行	insmod
调试	gdb	kdbug，kdb，kgdb等

模块主要函数及定义

- 头文件及宏定义

  #define __KERNEL__

  #define MODULE

  #include<linux／module.h>

  #include<linux／kernel.h>
- module_init():模块的初始化函数

  module_exit()：模块的卸载函数,

  初始化函数和卸载函数**必须成对出现**。
- 模块常用信息：作者、描述、版权等，

  MODULE_AUTHOR(“author”);

  MODULE_DESCRIPTION(“the description”);

  MODULE_LICENSE(“GPL”);

内核模块实例：

```
#define __KERNEL__
```

```
#define  MODULE
#include<linux／module.h>
#include<linux／kernel.h>
int init_module(void)
{
        printk("Hellow World!＼n”)；
        return 0；
)
void cleanup_module(void)
{
        printk("Goodbye, cruel world!＼n”)；
)
module_init(hello_init);
module_exit(hello_exit);
```

modutils

modutils是管理内核模块的一个软件包。

常用命令：

insmod命令

调用insmod程序，把需要插入的模块以目标代码的形式插入到内核中(加载模块)。在插入的时候，insmod自动调用init_module()函数运行。注意，只有**超级用户**才能使用这个命令。

格式

`#insmod [path]modulename.o`

rmmod命令

调用rmmod程序，将已经插入内核的模块从内核中移出（卸载模块）。rmmod会自动运行 cleanup_module()函数。

格式：

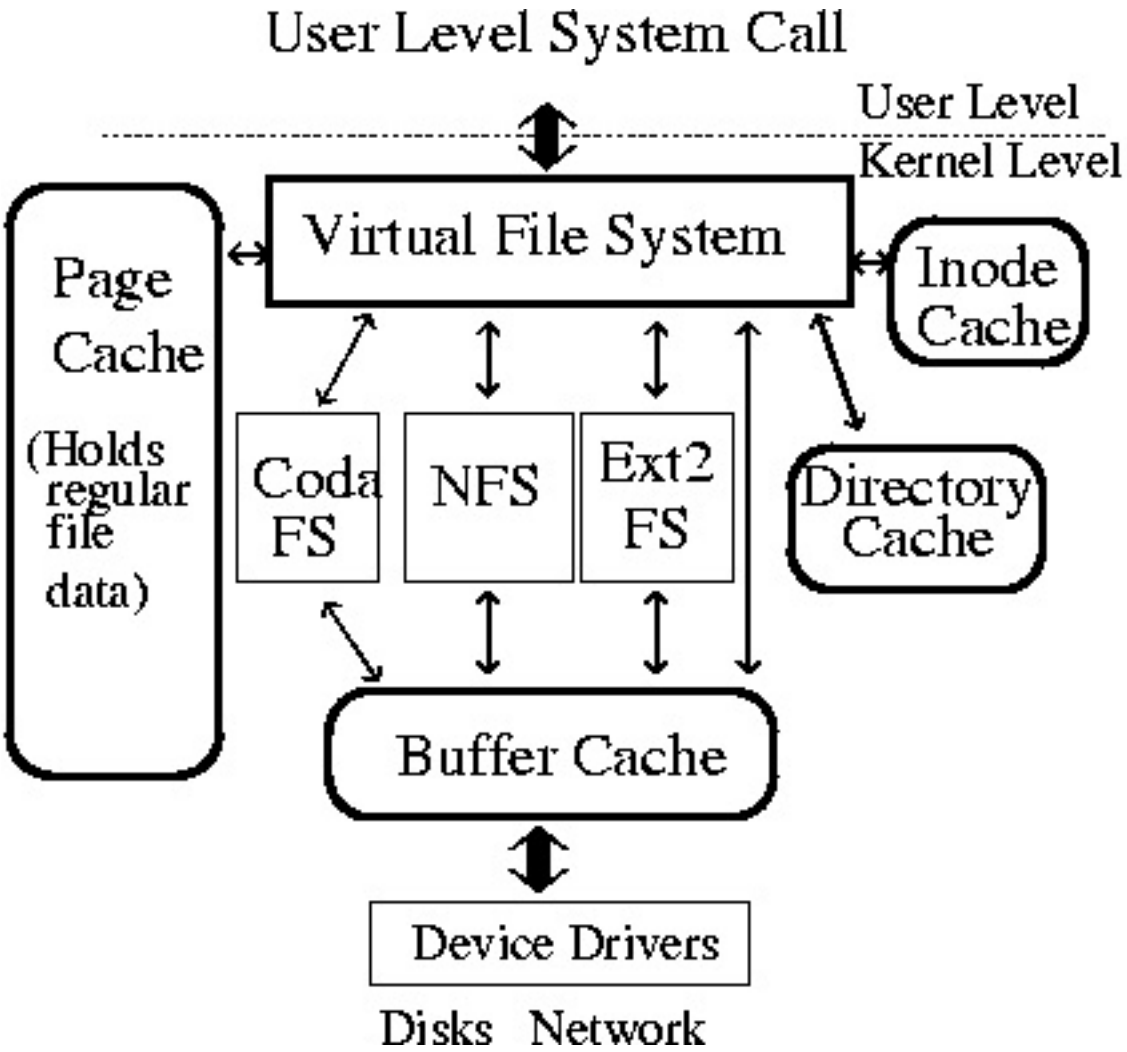
```
#rmmod [path]modulename.o
```

Linux文件系统

在UNIX系统中，文件系统是最基本的资源。在系统内核和文件系统之间制定一个标准的接口而实现的，不同文件结构之间可以通过该接口方便地交换数据。Linux正是使用这种方式，在系统内核和文件系统之间提供了一种标准接口——**VFS（virtual file system，虚拟文件系统）**。

Linux下的文件系统，由虚拟文件系统和实际的文件系统两个层次组成。目前常用的有EXT2、CRAMFS、JFFS2、NFTL、NFS和RAM 磁盘文件系统等。

VFS 在用户和文件系统之间提供了一个交换层



Flash简介

Flash主要分为**NOR**和**NAND**两类：

NOR的读取速度比NAND稍快一些。

NAND的写入速度比NOR快很多。

NAND的擦除速度远比NOR快。

大多数写入操作需要先进行擦除操作。

NAND的擦除单元更小，相应的擦除电路更少

常用文件系统：

EXT2,EXT3

Ext2是 GNU/Linux 系统中标准的文件系统，其特点为存取文件的性能极好，对于中小型的文件更显示出优势，这主要得利于其簇快取层的优良设计。

Ext3是一种**日志式文件系统**,是对ext2系统的扩展,它兼容ext2。

日志式文件系统的优越性在于:由于文件系统都有快取层参与运作，如不使用时必须将文件系统卸下。因此每当系统要关机时，必须将其所有的文件系统全部shutdown后才能进行关机。如果在文件系统尚未shutdown前就关机 (如停电) 时，下次重开机后会造成文件系统的资料不一致，故这时必须做文件系统的重整工作，将不一致与错误的地方修复。使用所谓‘日志式文件系统 (Journal File System) ’最大的特色是，它会将整个磁盘的写入动作完整记录在磁盘的某个区域上，以便有需要时可以回溯追踪。

Romfs

传统型的Romfs文件系统是最常使用的一种文件系统，它是一种简单的、紧凑的、只读的文件系统，不支持动态擦写保存;它按顺序存放所有的文件数据，所以这种文

件系统格式支持应用程序以XIP方式运行，在系统运行时，可以获得可观的RAM节省空间。

Cramfs

Cramfs是针对Linux内核2.4之后的版本所设计的一种新型文件系统，也是压缩和只读格式的。它主要的优点是将文件数据以压缩形式存储，在需要运行的时候进行解压缩。由于它存储的文件形式是压缩的格式，所以文件系统不能直接在 Flash上运行。虽然这样可以节约很多Flash存储空间，但是文件系统运行需要将大量的数据拷贝进RAM中，消耗了RAM空间。

JFFS和JFFS2

JFFS和JFFS2日志式文件可以通过使用MTD driver，在Flash头部建立根文件系统（Root Filesystem）。
日志式文件系统可以免受系统突然掉电的危险，并且在下一次系统引导时不需要文件系统的检查。由于JFFS和JFFS2文件格式是特别为Flash存储器设计的，二者都具一种称为“损耗平衡”的特点，也就是说Flash的所有被擦写的单元都保持相同的擦写次数。利用这些特有保护措施，Flash的使用周期得到相当大的提升。JFFS2使用压缩的文件格式，为Flash节省了大量的存储空间，它更优于JFFS格式在系统中使用。

Ramdisk

Ramdisk也是Linux 系统中的一种虚拟设备,是从内存中划分出来作为高速缓存的一部分,将它虚拟为磁盘。Ramdisk device (如:/ dev/ ram0 ,/ dev/ ram1) 可以在任何时候被创建和加载, 同其他磁盘分区一样被操作。Ramdisk 通常用于系统启动。
如果使用RAM disk，一般应选择EXT2文件格式。

由于存在RAM disk上，所以任何改变在下一次启动后都会丢失。

Ramdisk的优点

提升系统性能，能把对慢速硬盘的操作转换为对高速内存的操作，既提高了系统资源利用率，又极大提高了系统运行效率。
提高软件性能，由于系统内存存取速度远快于硬件磁盘速度，所以对于要频繁进行磁盘存取的应用程序（如数据库程序、磁盘文件交换程序、网站服务程序）使用Ramdisk，能有效提高应用程序性能能，虚拟内存盘还有一个优点就是不会磨损磁头，特别适合于多线程，大吞吐量的磁盘操作。
适合存放临时文件，由于虚拟内存盘的特性是将数据完全存储在内存中，所以一旦关闭电脑，就会导致虚拟内存盘中的数据完全丢失，这个特性使得虚拟内存盘特别适合于存储一些临时文件，如IE缓存、Windows和应用程序运行时产生的临时文件，都非常适合放到虚拟内存盘中，从而减少硬盘上文件碎片的产生，并且不需要主动删除这些临时文件，一旦重启，这些临时文件就会自动消失，当然也正是这一特性使得虚拟内存盘不适合存储重要的数据和文档。

Linux的根文件系统

每台机器都有根文件系统，它包含系统引导和mount其他文件系统所必需的文件。还应该包括修复损坏系统、恢复备份等的工具。
根文件系统一般比较小，根目录一般不含任何文件，除了可能的标准的系统引导映象，通常叫/vmlinuz，所有其他文件在根文件系统的子目录中。
根文件系统损坏一般意味着系统无法引导。

Linux 的根文件系统的主要目录：

/bin：包含基本的用户命令工具程序
/sbin：包含基本的系统管理程序

/boot：包含内核映像及启动相关文件

/etc：（excutive time config）包含系统配置文件和脚本

/lib：包含系统库和内核模块

/usr：用户程序及库目录

/home：用户主目录

/root：root 用户主目录

/dev：设备文件目录，目录下的每个文件代表一个设备

/opt：额外软件包所在目录

/mnt：文件系统临时挂装目录

/var：包含运行时改变的文件，例如lock 和log 文件

/proc：内核创建和使用的虚拟文件系统，存放运行时系统信息

/tmp：临时文件目录

文件存放规则

1．把全局配置文件放入/etc目录下。

2．将设备文件信息放入/dev目录下，设备名可以作为符号链接定位在/dev中或/dev子目录中的其他设备存在。

3．操作系统核心定位在/或/boot。

4．库存放的目录是/lib。

5．存放系统编译后的可执行文件、命令的目录是/bin，/sbin，/usr。

BusyBox

Busybox被非常形象地称为嵌入式Linux系统中的“瑞士军刀”，因为它将许多常用的UNIX命令和工具结合到了一个单独的可执行程序中。它是一个集成了一百多个最常用linux命令和工具的软件,它甚至还集成了一个http服务器和一个telnet服务器。虽然与相应的GNU工具比较起来，Busybox所提供的功能和参数略少，但在比较小

的系统(例如启动盘)或者嵌入式系统中，这已经足够了。

BusyBox的用法

`busybox ls` 其功能就相当运行ls命令

最常用的用法是建立指向busybox的链接,不同的链接名完成不同的功能.

`ln -s busybox ls`

`ln -s busybox rm`

`ln -s busybox mkdir`

然后分别运行这三个链接:

`./ls`

`./rm`

`./mkdir`

就可以分别完成了ls 、rm和mkdir命令的功能。