1. bootloader

当我们拿到一款手机，第一件事应该就是按下电源键开机，那么从开机到进入到桌面程序这中间发生了些什么呢，我们从下面这张简化了的手机结构图开始：



注意：该结构图并不反映手机的实际分区顺序和位置，只是一个逻辑结构图。

大家可以简单的把手机的ROM存储类比为我们电脑上的硬盘，这个硬盘被分成了几个分区：bootloader分区，boot分区，system分区等等。所谓的刷机我们可以简单的理解成把软件安装在手机的某些分区中，类似于我们在电脑上安装Windows系统。

当按下电源键手机上电启动后，首先从bootloader分区中一个固定的地址开始执行指令，如图所示，bootloader分区分成两个部分，分别叫做primary bootloader和secondary stage bootloader。Primary bootloader主要执行硬件检测，确保硬件能正常工作后将secondary stage bootloader拷贝到内存(RAM)开始执行。Secondary stage bootloader会进行一些硬件初始化工作，获取内存大小信息等，然后根据用户的按键进入到某种启动模式。比如说大家所熟知的通过电源键和其它一些按键的组合，可以进入到recovery，fastboot或者选择启动模式的启动界面等。我们在论坛上看到的bootloader通常指的就是secondary stage bootloader。不过我们不需要关心太多的细节，可以简单的理解为bootloader就是一段启动代码，根据用户按键有选择的进入某种启动模式。

**fastboot模式：**fastboot是android定义的一种简单的刷机协议，用户可以通过fastboot命令行工具来进行刷机。比如说fastboot flash boot boot.img这个命令就是把boot.img的内容刷写到boot分区中。一般的手机厂商不直接提供fastboot模式刷机，而是为了显示他们的牛B之处，总是会提供自己专有的刷机工具和刷机方法。比如说三星的Odin，摩托的RSD，华为的粉屏等等。但是其本质实际上是相同的，都是将软件直接flash到各个分区中。

**recovery模式：**recovery是android定义的一个标准刷机协议。当进入recovery模式时，secondary stage bootloader从recovery分区开始启动，recovery分区实际上是一个简单的Linux系统，当内核启动完毕后，开始执行第一个程序init(init程序是Linux系统所有程序的老祖宗)。init会启动一个叫做recovery的程序（recovery模式的名称也由此而来）。通过recovery程序，用户可以执行清除数据，安装刷机包等操作。一般的手机厂商都提供一个简单的recovery刷机，而大名鼎鼎的CWM Recovery就是一个加入了很多增强功能的recovery，要想用上CWM Recovery前提是recovery分区可以被刷写。大家在论坛上看到的解锁bootloader，通常指的就是解锁recovery或fastboot，允许刷写recovery分区，这样大家就可以用上喜爱的CWM Recovery了。

手机除了普通的CPU芯片以外，还有MODEM处理器芯片。该芯片的功能就是实现手机必需的通信功能，大家通常所的刷RADIO就是刷写modem分区。

**2.正常启动**

当我们只是按下电源键开机时，会进入正常启动模式。Secondary stage bootloader会从boot分区开始启动。Boot分区的格式是固定的，首先是一个头部，然后是Linux内核，最后是用作根文件系统的ramdisk。

一般针对每个机型的完整刷机包中会有一个boot.img文件，这就是boot分区镜像文件。

当Linux内核启动完毕后，就开始执行根文件系统中的init程序，init程序会读取启动脚本文件(init.rc和init.xxxx.rc)。启动脚本文件的格式大家可以在网上找到很多参考资料，这里就不写了，而且我们在原厂ROM上移植MIUI的原则是不修改boot分区，因为有一些机型无法修改boot分区。

根文件系统中有一个重要的配置文件，叫default.prop，该文件的内容一般为：

#  
# ADDITIONAL\_DEFAULT\_PROPERTIES  
#  
ro.secure=1  
ro.allow.mock.location=1  
ro.debuggable=0  
persist.service.adb.enable=1。

文件中的每一行对某个属性赋值，在后续的文章中我们还会谈到属性。这里面大家需要注意的两个属性：ro.secure和ro.debuggable。如果ro.secure=0允许我们运行adb root命令。通常大家说得内核ROOT指的就是ro.secure=0。ROOT权限只是的手机上有一个名为授权管理的程序（Superuser.apk)可以授予程序root用户的权限。ro.deguggable=1允许调试系统APP。

init程序读取启动脚本，执行脚本中指定的动作和命令，脚本中的一部分是运行system分区的程序，下一节我们就来看看system分区的结构。**3.system分**

在讲system分区之前，我们先来看下面这张Android的软件系统架构图。



从上到下依次为：

核心应用层：这一层就是大家平常所接触的各种各样的系统自带应用，比如联系人，电话，音乐等。应用层往下就是开发人员所接触的。

框架层：这一层是Android系统的核心，它提供了整个Android系统运作的机制，像窗口管理，程序安装包管理，开发人员所接触的Activity, Service, broadcast等等。

JNI层：JNI层是Java程序和底层操作系统通信的一个机制，它使得Java代码可以调用C/C++代码来访问底层操作系统的API。

Dalvik虚拟机：Android开发使用Java语言，应用程序的Java代码会被编译成dalvik虚拟机字节码，这些字节码由dalvik虚拟机解释执行。

本地库：本地库一般是由C/C++语言所开发，直接编译成相应CPU的机器码，这其中包含标准C库，用以绘制图形的skia库，浏览器核心引擎webkit等。

HAL：硬件抽象层，为了和各个厂家的不同硬件工作，Android定义了一套硬件接口，比如说为了使用相机，厂家的相机驱动必须提供的接口方法。这样使得上层的代码可以独立于不同的硬件运行。

厂家适配层：本来Android定义的HAL层是直接和厂家提供的设备驱动打交道的，但是目前厂家不想开源HAL部分的代码，因此很多厂家都提供了一个我称之为厂家适配层的代码，这样在HAL层接口的实现只是一个简单的对厂家适配层接口函数的调用。

内核：这一层就是大家熟悉的Linux内核，内核中包含有各种硬件驱动，这些驱动不同的手机厂商不同的手机是不一样的。Linux内核是支持驱动模块化机制的，简单的说就是允许用户动态的加载或者卸载某个硬件驱动，但是目前来看，手机厂商除了提供WIFI驱动单独加载外，其它驱动都是和内核绑定在一起的。

从这张软件结构图来看，除了内核是放在boot分区外，其它层的代码都是在system分区中。

下面结合这张图来介绍system分区的主要目录内容：  
system/app: app目录下存放的是核心应用，也就是大家熟知的系统APP，这些系统自带的程序是不能简单的卸载的，要通过一些特殊的方式才能删除（大家熟悉的一种方法是用RE文件管理器）。

system/lib: lib目录下存放的是组成JNI层，Dalvik虚拟机，本地库，HAL层和厂家适配层的所有动态链接库(.so文件)。

system/framework：该目录下存放的是框架层的JAR包，其中对MIUI移植来说有3个最重要的JAR包(framework.jar, android.policy.jar, services.jar)。后续的文章会重点介绍这3个包。

system/fonts：该目录下存放的是系统缺省的字体文件。

system/media：该目录下存放的是系统所使用的各种媒体文件，比如说开机音乐，动画，壁纸文件等。不同的手机该目录的组织方式可能不一样。如何修改这些文件请参考网上对应机型形形色色的教程，这里不再赘叙。

system/bin：该目录下存放的是一些可执行文件，基本上是由C/C++编写的。其中有一个重要的命令叫app\_process下一节单独介绍。

system/xbin：该目录下存放的是一些扩展的可执行文件，既该目录可以为空。大家常用的busybox就放在该目录下。Busybox所建立的各种符号链接命令都是放在该目录。

system/build.prop：build.prop和上节说得根文件系统中的default.prop文件格式一样，都称为属性配置文件。它们都定义了一些属性值，代码可以读取或者修改这些属性值。属性值有一些命名规范：

ro开头的表示只读属性，即这些属性的值代码是无法修改的。  
persist开头的表示这些属性值会保存在文件中，这样重新启动之后这些值还保留。

其它的属性一般以所属的类别开头，这些属性是可读可写的，但是对它们的修改重启之后不会保留。

很多ROM制作者都会修改一下build.prop信息，里面的一些以ro.build开头的属性就是你在手机设置中的关于手机里看到的。可以通过修改build.prop文件来将这个ROM打上自己的印记（XXX所修改）。我见过一个只是删了system/app的一些程序，然后修改build.prop中的ro.build.display.id和ro.build.version.incremental中的两个属性值打上自己的大名的ROM。

system/etc：该目录存放一些配置文件，和属性配置文件不一样，这下面的配置文件可能稍微没那么的有规律。一般来说，一些脚本程序，还有大家所熟悉GPS配置文件(gps.conf)和APN配置文件(apns-conf.xml)放在这个目录。像HTC将相机特效所使用的一些文件也放在这个目录下。

上一节提到init会执行一个重要的命令程序叫app\_process，一般大家称之为Zygote。（Zygote是卵的意思，所有的Android进程都是由它生出来的)。Zygote首先会加载dalvik虚拟机，然后产生一个叫做system\_server的进程。system\_server顾名思义被称作Android的系统服务程序，它主要管理整个android系统。system\_server启动完成后开始寻找一个叫做启动器的程序，找到之后由zygote开始启动执行启动器，这就是我们常见到的桌面程序。

上面描述的是一个相当简化的启动过程，了解这些对于适配MIUI基本上就够了，如果大家对这些想进一步了解的话，请关注市面上各种Android内幕书籍。

这一节简单的介绍一下data和cache分区。当我们开机进入桌面程序后，一般来说我们都会下载安装一些APP，这些APP都安装在data/app目录下。所有的Android程序生成的数据基本上都保存在data/data目录下。wipe data实质上就是格式化data分区，这样我们安装的所有APP和程序数据就都丢失了。

cache分区从名字上来看是用来缓存一些文件的，比如说一些音乐下载的临时文件，或者下载管理下载的内容基本上放在这个分区。**. 小结**

本章主要是介绍了一下Android手机的硬软件结构以及主要分区的内容，并简要的介绍了一些开机启动过程。了解这些内容有助于我们从整体上理解ROM移植。