Mboot模式设计

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Number:** |  | **Document Version:** | 0.1 |
| **Owner:** | Luther.Ge Haibin.Xu | **Date:** | 2020/02/17 |
| **Document Type:** | HLD | | |
| **NOTE:** | ALL MATERIALS INCLUDED HEREIN ARE COPYRIGHTED AND CONFIDENTIAL UNLESS OTHERWISE INDICATED. The information is intended only for the person or entity to which it is addressed and may contain confidential and/or privileged material. Any review, retransmission, dissemination, or other use of or taking of any action in reliance upon this information by persons or entities other than the intended recipient is prohibited.  This document is subject to change without notice. Please verify that your company has the most recent specification.  Copyright © 2020 UNISOC Communications Inc. | | |

**UNISOC_logo**

[www.UNISOC.com](http://www.spreadtrum.com)

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 作者 | 说明 |
| V0.1 | 2020-02-17 | Haibin.Xu | 初稿 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[Mboot模式设计 1](#_Toc32963537)

[修订记录 2](#_Toc32963538)

[目录 3](#_Toc32963539)

[前言 4](#_Toc32963540)

[1. mboot介绍 5](#_Toc32963541)

[2. mboot功能 6](#_Toc32963542)

[3. mboot使用 7](#_Toc32963543)

[3.1 mboot Android版本编译 7](#_Toc32963544)

[3.2 mboot模式启动 8](#_Toc32963545)

[4. mboot应用 10](#_Toc32963546)

# 前言

Android设备从芯片上电到进入Android启动这整个过程中，一般是不存在停留的。但是对于研发人员而言，在系统内核启动后，Android启动前这个时间段，经常需要进行一些检查、设置、验证、测试等相关的操作，比如，在手机性能测试中，为避免一些不必要的影响，需要让性能测试程序在这个阶段运行以获取更有价值的测试数据。再比如，在项目BringUp时，通常情况下，Android系统基本上不会一次性就启动成功，经常会遇到诸如init不能正常启动，adb无法正常连接，console控制台启动失败等问题，解决此类问题，就更需要对该阶段进行仔细排查。

为此，我们设计了mboot模式，就是为在系统内核启动后，Android启动前这个时间段实现停留。

# mboot介绍

mboot是一种Android设计模式，该模式可让Android设备在启动过程中的某个时间段内实现停留，这样便于研发人员分析当前Android设备的启动状态，测试某些可执行程序，检查系统启动问题等等。

mboot模式的相关代码集成在tiny-formatter框架内，其基本原理就是在Kernel启动后通过执行特定的busybox，进入到shell环境，从而实现系统当前状态的停留，然后通过busybox内部集成的大量Linux命令来对当前环境进行操作。

# mboot功能

mboot的主要功能便是实现Android设备的启动过程中的停留，具体指的是Kernel启动后，console启动前这一阶段；在这一阶段，可以实现两次停留。

停留状态下，即进入mboot模式，因为mboot集成了busybox，所以busybox包含的所有Linux命令都可以在mboot模式下使用，通过这些Linux命令，可以对当前的环境进行必要的操作，比如：

mboot模式下，可以手动启动adb，adb启动后，我们可以在未启动Android情况下传输文件；

mboot模式下，可以进行一些debug，比如修改文件权限，增加相关属性，移除相关模块等等；

mboot模式下，对于user版本，可以获取root权限；

mboot模式下，我们可以修改log级别；

使用mboot模式，对我们的Android优化裁剪有很大便利，因为我们操作的大部分模块都是在内存中操作的，所以当某个模块裁剪后，当手机再次重启时，该模块会被重新恢复；

另外，mboot模式，可以同时应用在Android裁剪版本和非裁剪版本。

# mboot使用

Android设备若要具备mboot模式，需要烧录mboot Android版本。mboot Android版本有两种，即mboot Android裁剪版本和mboot Android完整版本。

## 3.1 mboot Android版本编译

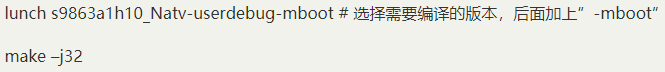
mboot Android版本编译编译简单灵活。

mboot Android完整版本编译有三种方式：

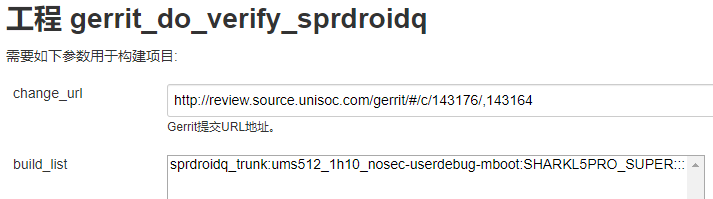
1. 在Android源码编译前，配置mboot编译环境变量，然后正常编译即可，如下图所示：



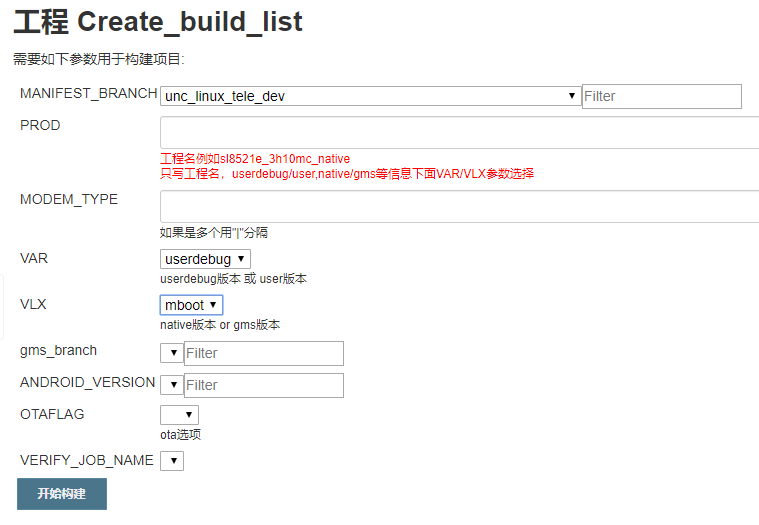
1. 在Android源码编译前，选择需要编译的版本，后面加上“-mboot”，然后正常编译即可，如下图所示：



1. 在verify服务器上编译，只需在build\_list中选择自己所要编译的带有“-mboot”版本的即可，如下图所示：



如果build\_list没有找到所需的mboot版本，可自行创建，我们已在创建新增verify工程的系统中增加了mboot选项，如下图所示：

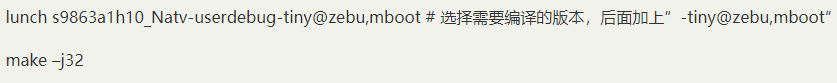


mboot Android裁剪版本编译有两种方式：

1. 在Android源码编译前，配置裁剪以及mboot编译的环境变量，然后正常编译即可，如下图所示：



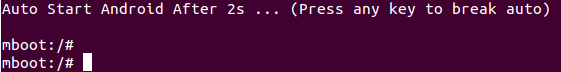
1. 在Android源码编译前，选择需要编译的版本，后面加上“tiny@zebu,mboot”或者“tiny@haps,mboot”，然后正常编译即可，如下图所示：



## 3.2 mboot模式启动

使用ResearchDownload工具，将已编译mboot Android版本的boot.img、dtb.img、 system.img、vendor.img替换掉原始pac中已有的相应的img文件，重新烧机。

mboot模式的启动，需要借助串口线，手机烧录成功后，链接串口线，重新开机，在kernel启动后，会有2秒时间等待输入任意键来进入mboot模式。２秒后如果没有收到输入，则自动启动android。如下图所示：



如果当前启动状态下，没有所需的相关操作对象，可在当前mboot模式下执行“android.start”命令，继续启动android，后面还会有一次5秒时间等待输入任意键来进入mboot模式，此时，基本上已经满足所需要的相关操作对象。

# mboot应用