



Unisoc Confidential For hiar

Android 9.0 10.0 省电管理介绍及配置指南

文档版本

V1.4

发布日期

2021-01-04

版权所有 © 紫光展锐（上海）科技有限公司。保留一切权利。

本文件所含数据和信息都属于紫光展锐（上海）科技有限公司（以下简称紫光展锐）所有的机密信息，紫光展锐保留所有相关权利。本文件仅为信息参考之目的提供，不包含任何明示或默示的知识产权许可，也不表示有任何明示或默示的保证，包括但不限于满足任何特殊目的、不侵权或性能。当您接受这份文件时，即表示您同意本文件中内容和信息属于紫光展锐机密信息，且同意在未获得紫光展锐书面同意前，不使用或复制本文件的整体或部分，也不向任何其他方披露本文件内容。紫光展锐有权在未经事先通知的情况下，在任何时候对本文件做任何修改。紫光展锐对本文件所含数据和信息不做任何保证，在任何情况下，紫光展锐均不负任何与本文件相关的直接或间接的、任何伤害或损失。

请参照交付物中说明文档对紫光展锐交付物进行使用，任何人对紫光展锐交付物的修改、定制化或违反说明文档的指引对紫光展锐交付物进行使用造成的任何损失由其自行承担。紫光展锐交付物中的性能指标、测试结果和参数等，均为在紫光展锐内部研发和测试系统中获得的，仅供参考，若任何人需要对交付物进行商用或量产，需要结合自身的软硬件测试环境进行全面的测试和调试。

Unisoc Confidential For hiar

紫光展锐（上海）科技有限公司



前言

概述

本文档主要介绍了 Android 9.0/Android 10.0 上的省电机制，包括原生省电机制和展锐自研的省电机制。

读者对象


本文档主要适用于开发、测试工程师。

缩略语

缩略语	英文全名	中文解释
SMD	Significant Motion Detector	有效运动检测器

符号约定

在本文中可能出现下列标志，它所代表的含义如下。

符号	说明
 说明	用于突出重要/关键信息、补充信息和小窍门等。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害。

变更信息

文档版本	发布日期	修改说明
V1.0	2019-10-31	初稿。
V1.1	2019-12-26	适用产品信息增加 UIS8581E\SL8541E 平台。
V1.2	2020-04-23	1. 文档名由《省电管理介绍及配置指导》修改为《省电管理介绍及配置指南》。 2. 结构调整、内容优化、格式更新等。

文档版本	发布日期	修改说明
V1.3	2020-09-16	更新模板
V1.4	2021-01-04	<ol style="list-style-type: none">1. 文档名由《省电管理介绍及配置指南》修改为《Android 9.0 10.0 省电管理介绍及配置指南》2. 内容变更：<ul style="list-style-type: none">- 2.5 自启动管理，自启动默认禁止修改为默认允许，并添加了“说明”。- 2.6 应用待机优化，增加了“说明”。- 增加了 3.5 修改自启动管理默认行为和 3.6 修改应用待机优化默认配置。3. 格式优化

关键字

Doze、Light Doze、App Standby、App Standby Buckets、省电管理、心跳对齐、省电模式、定时省电、低电量自动省电、自启动管理、应用待机优化、锁屏清理应用、高耗电应用。

Unisoc Confidential For hiar

目 录

1 原生省电机制.....	1
1.1 概述.....	1
1.2 Doze 介绍.....	1
1.2.1 进入与退出 Doze.....	2
1.2.2 豁免白名单.....	2
1.2.3 时间参数.....	3
1.3 Light Doze 介绍.....	4
1.3.1 进入与退出 Light Doze.....	4
1.3.2 豁免白名单.....	5
1.3.3 Light Doze 与 Doze 共存.....	5
1.3.4 时间参数.....	5
1.4 App Standby 介绍.....	6
1.5 App Standby Buckets 介绍.....	7
2 自研省电管理功能介绍.....	8
2.1 心跳对齐.....	9
2.2 省电模式.....	10
2.3 定时省电.....	16
2.4 低电量自动省电.....	18
2.5 自启动管理.....	20
2.6 应用待机优化.....	22
2.7 锁屏清理应用.....	24
2.8 高耗电应用.....	26
2.9 充电时退出省电模式.....	26
2.10 低分辨率省电.....	27
3 相关配置.....	29
3.1 恢复 Android 原生省电功能.....	29
3.2 心跳对齐配置.....	29
3.2.1 心跳对齐功能开关.....	29
3.2.2 预置黑名单.....	30
3.2.3 菜单设置.....	30
3.3 第三方应用默认配置.....	31
3.4 超级省电默认应用添加.....	32
3.5 修改自启动管理默认行为.....	32
3.6 修改应用待机优化默认配置.....	33
4 测试建议.....	35

图目录

图 1-1 原生省电机制演进	1
图 1-2 系统进入 Doze 后的状态示意图	2
图 1-3 Light Doze	4
图 1-4 Doze override Light Doze	5
图 2-1 省电管理功能	8
图 2-2 设置中电池界面	9
图 2-3 心跳功能对齐示意图	9
图 2-4 心跳对齐算法	10
图 2-5 省电模式选择界面	10
图 2-6 Android 9.0 省电模式	12
图 2-7 Android 10.0 省电模式	12
图 2-8 超级省电模式桌面及锁屏界面	13
图 2-9 超级省电模式允许添加/删除应用	14
图 2-10 超级省电模式下拉状态栏	14
图 2-11 进入/退出超级省电模式的提示框	15
图 2-12 省电模式设置入口	15
图 2-13 省电模式下拉状态栏入口	16
图 2-14 定时省电选项	16
图 2-15 定时省电设置界面	17
图 2-16 定时省电自动进入/退出的提示	18
图 2-17 低电量自动省选项	18
图 2-18 低电量自动省电设置界面	19
图 2-19 低电量自动省电进入/退出的提示	19
图 2-20 低电量提示	20
图 2-21 自启动管理选项	21
图 2-22 自启动管理设置界面	21

图 2-23 应用待机优化选项	22
图 2-24 应用待机优化设置界面	23
图 2-25 应用待机优化各项设置界面	24
图 2-26 锁屏清理应用界面	25
图 2-27 高耗电应用界面	26
图 2-28 充电时退出省电模式界面	27
图 2-29 低分辨率省电界面	28

Unisoc Confidential For hiar

表目录

表 1-1 Doze 时间参数表	3
表 1-2 Light Doze 时间参数表	6
表 1-3 应用分组	7
表 3-1 配置文件参数	31

Unisoc Confidential For hiar

1 原生省电机制

1.1 概述

Android 原生系统从 5.0 开始引入优化待机功耗的机制，6.0 引入 Doze 功能和 App Standby 功能，7.0 引入 Light Doze，8.0 对应用的后台行为进行限制，9.0 引入待机分组。如图 1-1 所示。

图1-1 原生省电机制演进

Lollipop	Marshmallow	Nougat	Oreo	Pie
Project Volta	App Restrictions	More App Restrictions	MOAR App Restrictions	MOAR App Restrictions
Job Scheduler	Doze (Screen off + stationary)	Doze-lite (while in pocket)	Background Limits	App standby buckets
Battery Historian	App Standby (Unused apps)	Limit implicit broadcasts	Background Location Limits	Battery saver improvements
			Cached wakelock release	

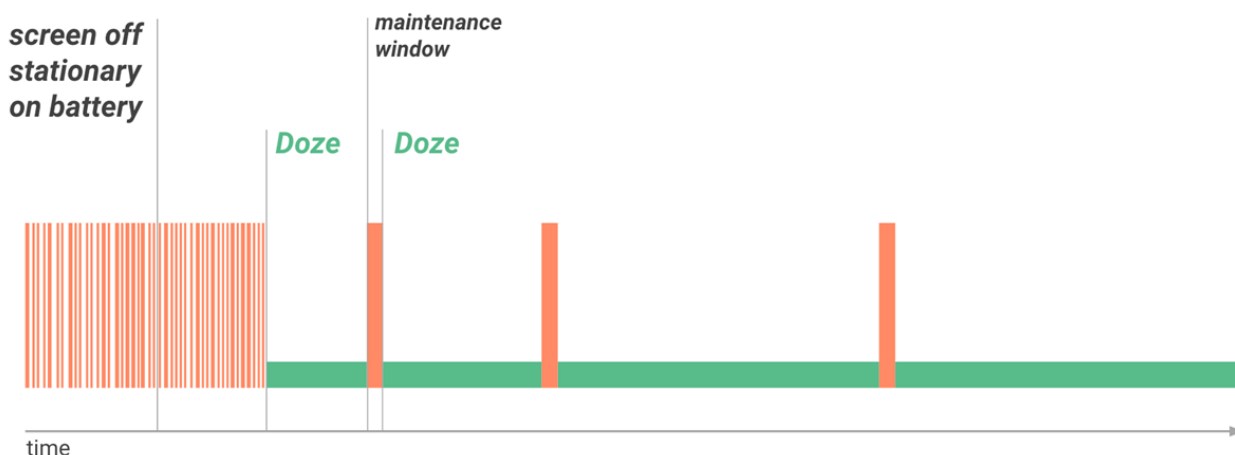
1.2 Doze 介绍

Doze 功能从 Android6.0 开始引入，当进入 idle 状态时，会限制相关应用的下列功能：

- 禁止网络访问
- Jobs/syncs 被延迟
- 忽略 Wakelock
- Alarms 被延迟
- GPS/Wifi 停止 scan

Doze 会周期性退出 idle 状态以便让应用去执行被延迟的 Jobs/syncs 及 Alarms 等。同时这个周期会随着进入 Doze 的时间越长而越大。默认初始值为 1 小时，往后以 2 倍的速度增加。如图 1-2 所示。

图1-2 系统进入 Doze 后的状态示意图



1.2.1 进入与退出 Doze

在 config.xml 中设置 config_enableAutoPowerModes 为 true 时，会使能 Doze 功能。

- 当系统满足：灭屏&&不插电&&静止达到一定时间时，就会开始进入 Doze。
静止不动的时间由下面几段时间组成：
 - INACTIVE_TIMEOUT**：即灭屏待机后等待的时间，默认为 30 分钟。
 - IDLE_AFTER_INACTIVE_TIMEOUT**：即在 INACTIVE_TIMEOUT 时间超时后，系统会启动 Motion Sensor（一般是 SMD 即 Significant Motion Detector）来检测手机是否在运动。检测的时长为 IDLE_AFTER_INACTIVE_TIMEOUT，默认为 30 分钟。
 - SENSING_TIMEOUT**：即当手机在 IDLE_AFTER_INACTIVE_TIMEOUT 超时前一直保持没有运动，在 IDLE_AFTER_INACTIVE_TIMEOUT 超时后，会打开 AnyMotionDetector 检测手机是否静止，检测的最长时长为 SENSING_TIMEOUT，如果在 SENSING_TIMEOUT 超时前，AnyMotionDetector 返回静止的结果，则会使状态提前进入下一个状态。否则，等到 SENSING_TIMEOUT 超时后，直接进入下一个状态。SENSING_TIMEOUT 默认是 4 分钟。
 - LOCATION_TIMEOUT**：即当手机通过 AnyMotionDetector 检测为静止，或者 SENSING_TIMEOUT 超时后，会打开 Location 请求定位，通过网络和 GPS 进行定位。并启动超时时间为 LOCATION_TIMEOUT 的定时器，在该定时器超时后会进入下一个状态，即 idle 状态。该超时时间默认为 30 秒。如果在超时前，AnyMotionDetector 和 Location 都成功返回，并满足手机静止的条件，则会提前进入 idle 状态。

综上，从开始灭屏待机到进入 Doze 的 idle 状态默认耗时一般是大于

INACTIVE_TIMEOUT+IDLE_AFTER_INACTIVE_TIMEOUT 而小于等于

INACTIVE_TIMEOUT+IDLE_AFTER_INACTIVE_TIMEOUT+SENSING_TIMEOUT+LOCATION_TIMEOUT。默认大概是 1 个小时。

- 当进入 idle 状态的过程中或者已经进入 idle 状态，系统检测到：手机运动（通过 Motion Sensor）|| 亮屏||插电||alarm clock alarm（闹钟）||其他用户事件时，会退出 idle 状态，即退出 Doze。

1.2.2 豁免白名单

当系统进入 Doze 后，满足下列条件时，会对应用的 wakelock&network&job/sync jobs 产生限制：

转入后台的应用

`procState<=ActivityManager.PROCESS_STATE_BOUND_FOREGROUND_SERVICE`。

而 alarm 的调整则是对所有的应用都产生限制。

因此为避免对应用产生限制，可以将应用添加入 Doze 白名单。加入 Doze 白名单后，对该应用就不做任何限制。

将应用添加到 Doze 白名单的方法：

- 针对内置应用，可以通过将应用的包名添加到 `frameworks/base/data/etc/platform.xml` 文件中。添加格式如下：

```
<allow-in-power-save package="com.android.shell" />
```

- 针对非内置应用，安装后，在调试时可以通过命令添加：

```
adb shell dumpsys deviceidle whitelist //查看 doze 白名单
```

```
adb shell dumpsys deviceidle whitelist +<packageName> // 添加 doze 白名单
```

```
adb shell dumpsys deviceidle whitelist -<packageName> // 删除 doze 白名单
```

1.2.3 时间参数

Doze 相关的时间参数如表 1-1。

表1-1 Doze 时间参数表

参数	说明
INACTIVE_TIMEOUT	指定灭屏待机后等待的时间，默认是 30 分钟。
IDLE_AFTER_INACTIVE_TIMEOUT	指定打开 Motion Sensor 检测手机是否运动的持续时间，默认是 30 分钟。
SENSING_TIMEOUT	指定打开 AnyMotionDetector 检测手机是否静止的超时时间，默认是 4 分钟。
LOCATION_TIMEOUT	指定通过 LocationManager 检测手机定位的超时时间，默认是 30 秒。
MOTION_INACTIVE_TIMEOUT	当在 AnyMotionDetector 阶段检测手机不是静止时，会重新进入 idle 的流程，此时灭屏待机的等待时间会被指定为 MOTION_INACTIVE_TIMEOUT，默认为 5 分钟。
IDLE_PENDING_TIMEOUT	指定从 idle 状态退出进入 maintenance 的最短持续时间，默认为 5 分钟。
MAX_IDLE_PENDING_TIMEOUT	指定从 idle 状态退出进入 maintenance 的最长持续时间，默认为 10 分钟。
IDLE_TIMEOUT	指定保持在 idle 状态的最小持续时间，默认为 1 小时。
MAX_IDLE_TIMEOUT	指定保持在 idle 状态的最大持续时间，默认为 6

参数	说明
	小时。
IDLE_FACTOR	随着进入 doze 的时间越长，停留在 idle 状态的持续时间会按 IDLE_FACTOR 指定速率进行增加，直至达到 MAX_IDLE_TIMEOUT。 IDLE_FACTOR 默认为 2 倍。

上述时间参数在 DeviceIdleController::Constants.updateConstants()中指定默认值。在实际运行中，可以通过修改设置 Settings.Global.DEVICE_IDLE_CONSTANTS 的值来动态调整。如：

```
"inactive_to=60000,sensing_to=400000"
```

指定 INACTIVE_TIMEOUT 为 60 秒，SENSING_TIMEOUT 为 400 秒。

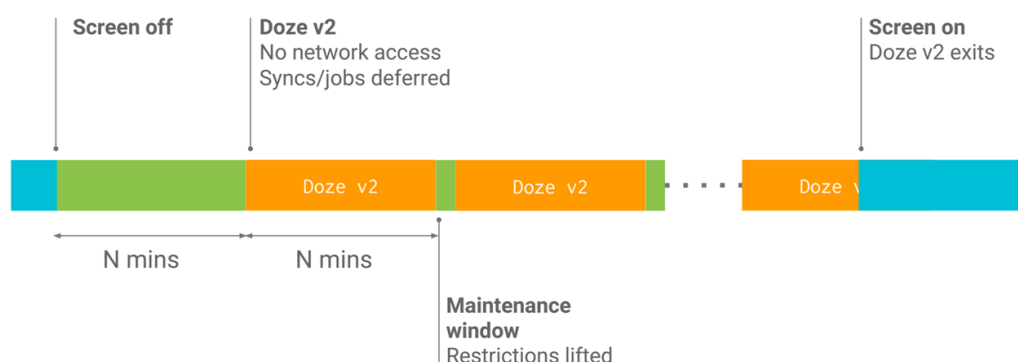
1.3 Light Doze 介绍

Light Doze 从 Android 7.0 开始引入，是 Doze 的轻量级功能，当进入 Light Doze 后，会限制应用的下列功能：

- 禁止网络访问
- Jobs/syncs 被延迟

Light Doze 会周期性地让应用去执行被延迟的 Jobs/syncs。而这个周期默认在 5~15 分钟。如图 1-3 所示。

图1-3 Light Doze



1.3.1 进入与退出 Light Doze

在 config.xml 中设置 config_enableAutoPowerModes 为 true 时，会使能 Light Doze 功能。

- 当系统满足：不插电&&灭屏一段时间时，就会开始进入 Light Doze。
灭屏待机的等待时间由下面组成：
 - **LIGHT_IDLE_AFTER_INACTIVE_TIMEOUT**：即灭屏待机后的等待时间，默认为 3 分钟

- **LIGHT_PRE_IDLE_TIMEOUT**: 当 **LIGHT_IDLE_AFTER_INACTIVE_TIMEOUT** 超时后, 会检查系统当前是否有正在处理的 jobs/alarms 等, 如果有还会等待 **LIGHT_PRE_IDLE_TIMEOUT**, 该值默认为 3 分钟; 如果当前系统空闲, 则直接进入 Light Doze 的 idle 状态。
- 当系统发生: 亮屏||插电||其他用户事件时, 会退出 Light Doze。

系统进入 Light Doze 后会在 Light Doze 的 idle 状态停留一段时间, 最长为 **LIGHT_MAX_IDLE_TIMEOUT** (默认为 15 分钟), 最短为 **LIGHT_IDLE_TIMEOUT** (默认为 5 分钟)。

说明

因为 Light Doze 只限制满足条件的应用的网络访问和 Jobs/sync, 并且周期唤醒的周期比较短, 在 5~15 分钟之间。因此对于系统中应用的 Jobs/sync 很少且没有网络访问的场景, 功耗优化有限, 有可能还会增加。

1.3.2 豁免白名单

当系统进入 Light Doze 后, 满足下列条件时, 会对应用的 network & job/sync jobs 产生限制:

转入后台的应用&&

应用的 `procState<=ActivityManager.PROCESS_STATE_BOUND_FOREGROUND_SERVICE`。

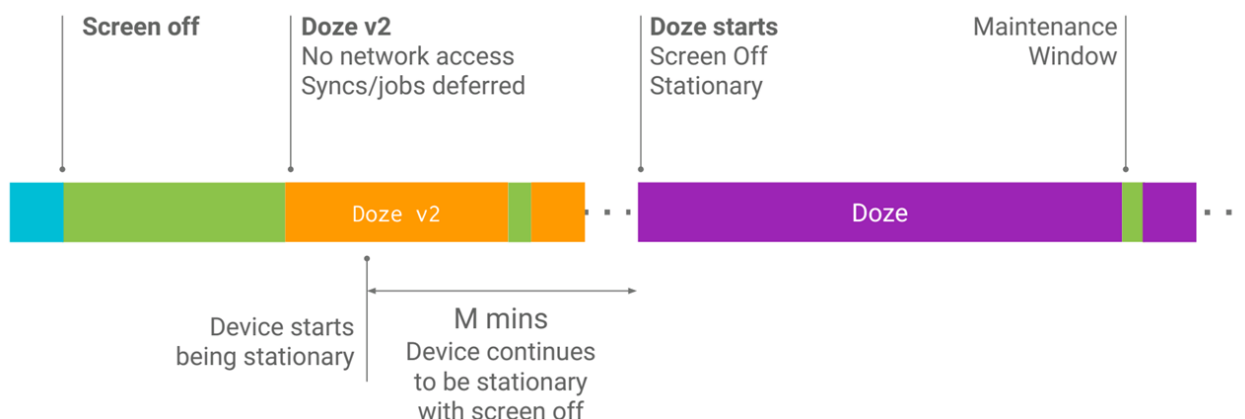
因此为避免对应用产生限制, 可以将应用添加入 Doze 白名单。加入 Doze 白名单后, 对该应用就不做任何限制。

添加 Doze 白名单的方法同 [1.2.2 豁免白名单](#)。

1.3.3 Light Doze 与 Doze 共存

Doze 与 Light Doze 是并存的, 当 Doze 进入 idle 状态后, Light Doze 将处于被 Doze override 的状态, 即此时 Light Doze 不再起作用, 被 Doze 取代。其整个过程如图 1-4。

图1-4 Doze override Light Doze



1.3.4 时间参数

Light Doze 的相关时间参数如表 1-2。

表1-2 Light Doze 时间参数表

参数	说明
LIGHT_IDLE_AFTER_INACTIVE_TIMEOUT	指定灭屏待机后需要等待的时间，默认为 3 分钟。
LIGHT_PRE_IDLE_TIMEOUT	当系统当前有正在处理的 jobs/alarms 时，会再等待 LIGHT_PRE_IDLE_TIMEOUT 时间，再进入 Light idle 状态，该时间默认为 3 分钟。
LIGHT_IDLE_TIMEOUT	指定保持在 Light idle 状态的最小持续时间，默认为 5 分钟。
LIGHT_MAX_IDLE_TIMEOUT	指定保持在 Light idle 状态的最大持续时间，默认为 15 分钟。
LIGHT_IDLE_FACTOR	随着进入 Light Doze 的时间越长，停留在 Light idle 状态的持续时间会按 LIGHT_IDLE_FACTOR 指定速率进行增加，直至达到 LIGHT_MAX_IDLE_TIMEOUT。 LIGHT_IDLE_FACTOR 默认为 2 倍。
LIGHT_IDLE_MAINTENANCE_MIN_BUDGET	指定停留在 maintenance 状态的最小持续时间。
LIGHT_IDLE_MAINTENANCE_MAX_BUDGET	指定停留在 maintenance 状态的最长持续时间。
MIN_LIGHT_MAINTENANCE_TIME	当从 Light idle 退出时，会发送广播并调用相关模块接口通知系统各模块此时退出了 Light idle，为了能让各模块可以做出相关处理，DeviceIdleController 会在等待 MIN_LIGHT_MAINTENANCE_TIME 后，检查系统当前是否有 alarms 或者 jobs 在处理，如果没有，则会提前结束 maintenance 状态。 MIN_LIGHT_MAINTENANCE_TIME 的默认值为 5 秒。

上述时间参数在 DeviceIdleController::Constants.updateConstants()中指定默认值。在实际运行中，可以通过修改设置 Settings.Global.DEVICE_IDLE_CONSTANTS 的值来动态调整。

如：

```
"light_after_inactive_to=60000,light_pre_idle_to=400000"
```

指定 LIGHT_IDLE_AFTER_INACTIVE_TIMEOUT 为 60 秒，LIGHT_PRE_IDLE_TIMEOUT 为 400 秒。

1.4 App Standby 介绍

App Standby 从 Android 6.0 开始引入。当应用进入 Standby 状态时，会限制下列功能：

- 禁止网络访问
- Jobs/syns 被延迟

进入与退出 App Standby

- 应用进入 Standby 的条件：不插电&&不活动（非 active）一段时间（一般要求 24 小时以上）。
- 应用退出 Standby 的条件：插电||应用被使用。
- 应用被认为 Active（被使用）的条件：
 - 自身是前台应用或前台服务，或者正被其他前台应用或服务使用，如壁纸服务等。
 - 有状态栏提示的应用。
 - 被用户打开的正使用的应用。

因为应用进入 Standby 后，会被限制网络和 jobs/syns，为避免进入 Standby 状态而被限制，可以将应用加入 Doze 白名单，加入的方法同 [1.2.2 豁免白名单](#)。

1.5 App Standby Buckets 介绍

App Standby Buckets（应用待机分组）从 Android9.0 开始引入，是对 App Standby 的增强，基于 App 最近使用时间和使用频率对应用进行分组，系统根据 App 所属的群组限制每个 App 的 alarms、jobs、network 访问等。原先的 App Standby 对应下面的分组 Rare。

根据用户的使用模式，App 被划分到五个群组之一。五个群组为：

- Active：当前正在使用的 App。
- Working set：经常使用，但当前未处于 active 状态的 App。
- Frequent：定期使用，但不是每天都必须使用的 App。
- Rare：不经常使用的 App。
- Never：安装但是从未运行过的 App。

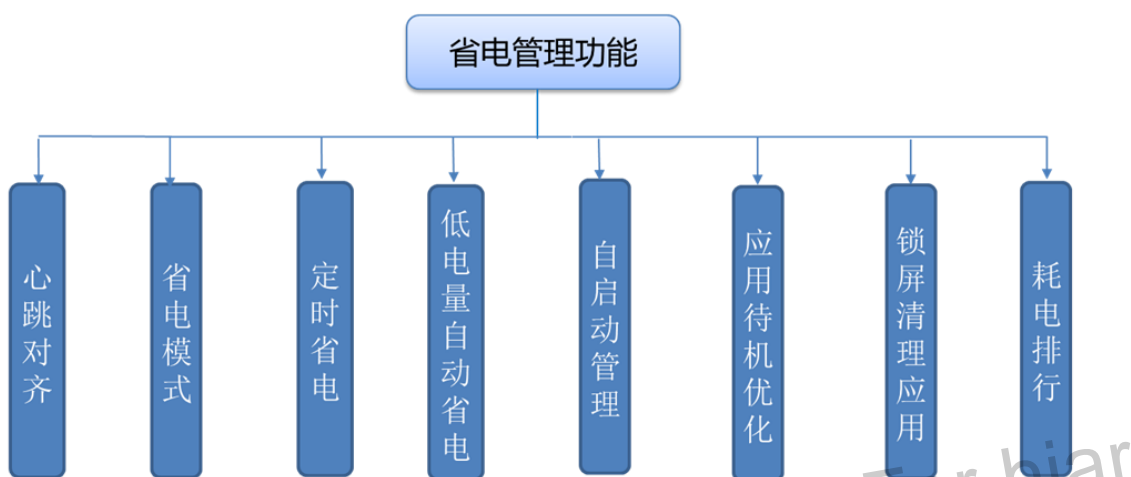
表1-3 应用分组

应用分组	Jobs/sync	Alarms	网络	高优先级 FCM 信息
Active	无限制	无限制	无限制	无限制
Working set	2 小时一次	间隔不小于 6mins	无限制	无限制
Frequent	8 小时一次	间隔不小于 30mins	无限制	10/day
Rare	24 小时一次	间隔不小于 2 小时	24 小时一次	5/day

2 自研省电管理功能介绍

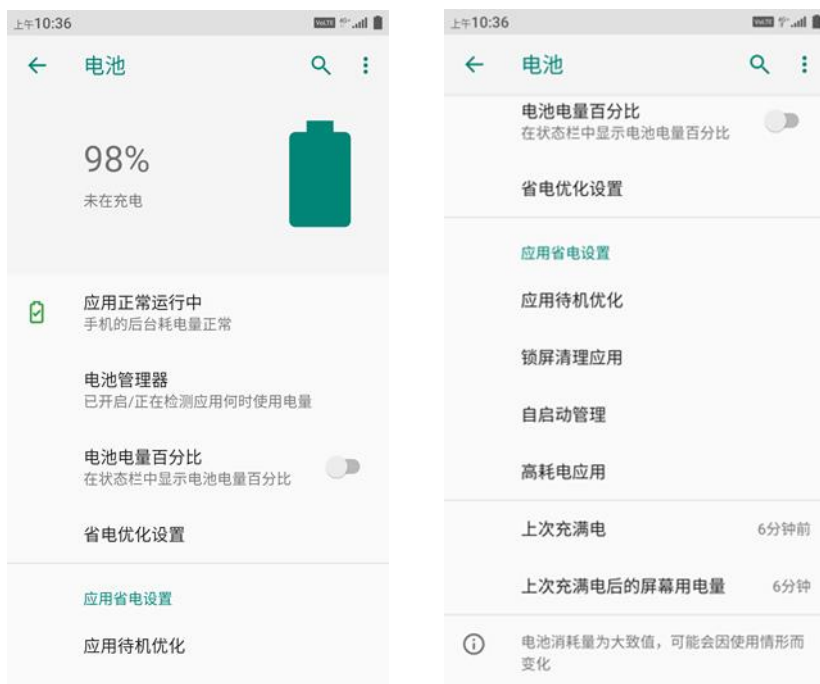
自研省电管理功能包含如图 2-1 所示的内容。

图2-1 省电管理功能



设置中电池界面如图 2-2 所示。

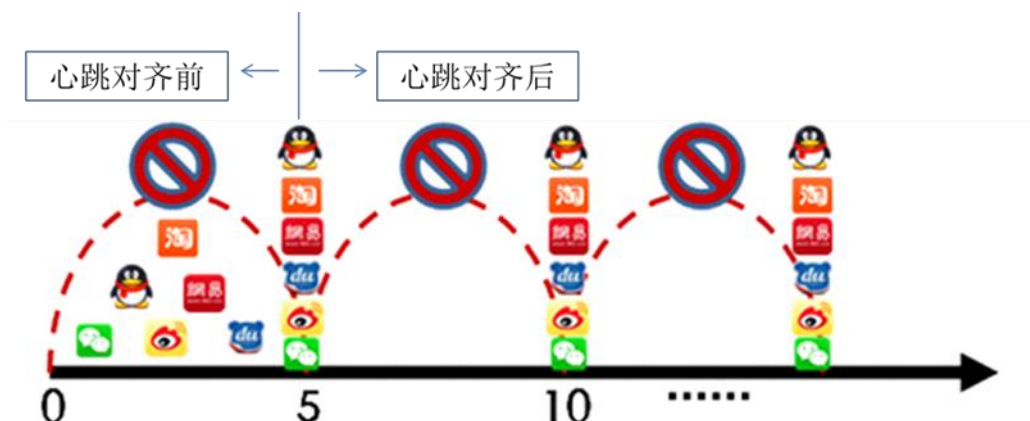
图2-2 设置中电池界面



2.1 心跳对齐

目前大部分应用都会利用设置 alarm 来定时或周期唤醒系统来做一些包括进行网络连接在内的操作，这种随机的 alarm 唤醒会增加系统的待机功耗。引入心跳对齐功能，就是为了将应用设置的这些 alarm 的唤醒时机进行对齐，以达到减少系统唤醒次数的目的，进而减少系统的待机功耗。心跳对齐通过调整应用设置的 alarm 唤醒时间，将其按指定的间隔进行对齐，对齐效果如图 2-3 所示。

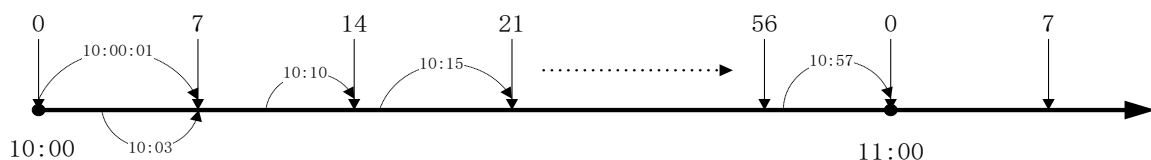
图2-3 心跳功能对齐示意图



心跳对齐算法说明

采用向后对齐的方式对 alarm 的触发时间进行调整。下面以 7 分钟为对齐长度进行说明。如图 2-4 所示，对齐的起点为整点时刻，7 分钟的整数倍时刻点为对齐点，点画曲线表示对齐过程。在两个对齐点之间的 alarm 都会被调整到后一个对齐点上。比如：10:00:01 和 10:03 会向 10:07 对齐，10:10 会向 10:14 对齐，10:15 会向 10:21 对齐，10:56 会向 11:00 对齐，10:57 会向 11:00 对齐。

图2-4 心跳对齐算法

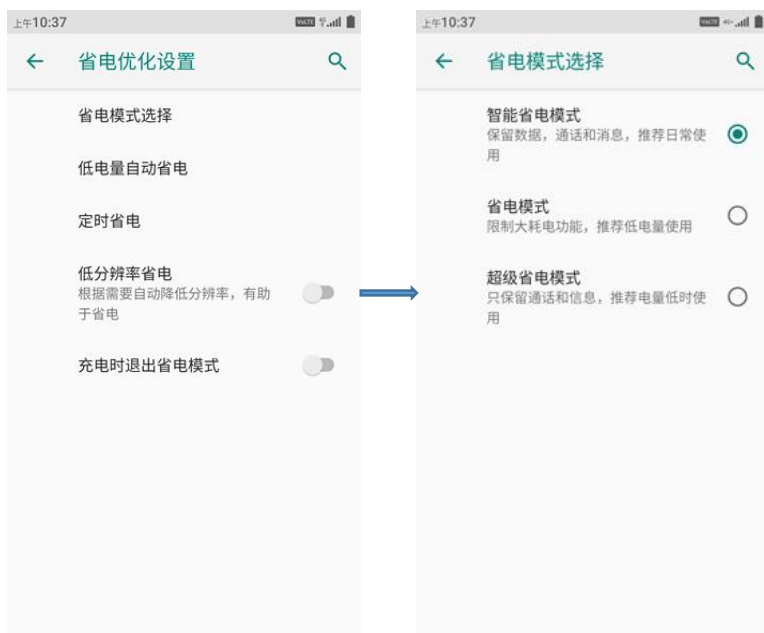


采用向后对齐的优点是可保证 alarm 不会丢失，且不会立即触发；缺点是 alarm 的触发时间偏差变大了。另外，还可以采用就近对齐的方式进行调整，这样误差会小很多，但是在实际调试中发现，就近对齐之后的时间可能会变成过去时或者离当前时间非常近，那么这个 alarm 就会被立即分发，如果同时应用设置了循环的 alarm，则会引起短时间内大量的 alarm 被触发。进而使得机器无法快速休眠。综合考虑上面的因素，采用向后对齐方式。

2.2 省电模式

省电模式包含三种，智能省电模式、省电量模式以及超级省电模式，如图 2-5 所示。

图2-5 省电模式选择界面



智能省电模式

为日常工作模式，在该模式下会根据系统状态，自动对应用的 alarm 唤醒、持锁、后台数据访问、后台 GPS 访问进行优化，以不影响应用使用和性能的前提，对系统待机功耗进行尽可能的自动优化。

其具体策略包括：

- 使用心跳对齐（Alarm 对齐）机制控制应用的 alarm 唤醒。
- 使用 App Standby 机制控制后台应用的频繁数据访问。
- 自动识别异常的长时间持锁的应用，限制应用的最长持锁时间。
- 自动后台清理，在待机一定时间后，对后台进程进行适当清理，并禁止后台进程自动启动。
- 自动识别后台使用 GPS 的应用，暂停后台应用对 GPS 的使用。
- 在凌晨时间段（00：00～6：00）强制进入 Doze。

省电模式

推荐在电量紧张的情况下使用。进入该模式后，会关闭振动、调暗亮度、关闭 GPS/WIFI/BT/NFC 等功能。待机时会根据系统状态自行断开网络，在退出待机时，恢复网络连接。在该模式下，应用转入后台后将有可能无法接收网络数据，直到重新打开该应用。

其具体策略包括：

- 智能省电的策略都使能。
- 打开 Android 原生的低电量模式，Android 原生低电量模式打开后会限制后台数据、限制振动、灭屏时限制 GPS 使用。
- 进入该模式后，关闭无线数据连接、GPS、WIFI、WIFI 热点、蓝牙和 NFC 功能，休眠时间设置为 30 秒内，屏幕亮度设置为不超过 30%。
- 灭屏待机后，强制进入 Doze 状态。
- 锁屏待机 15 分钟后，针对非前台应用进行清理。
- 待机 30 分钟后网络断开，亮屏后如果之前网络打开，则自动连接网络。若发现有应用在下载或者在线播放音乐，则不进行网络断开。

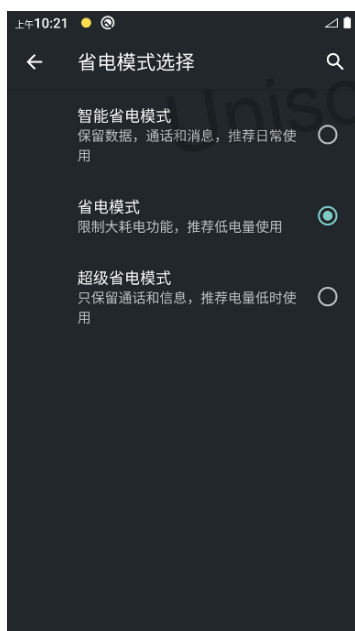
Android 9.0 进入省电模式后，状态栏电池图标显示为红色并包含+号，如图 2-6 所示。

图2-6 Android 9.0 省电模式



Android 10.0 进入省电模式后，默认会开启深色主题，如图 2-7 所示。

图2-7 Android 10.0 省电模式



超级省电模式

推荐在电量紧张的情况下使用。进入该模式后，会关闭振动、调暗亮度、关闭 GPS/WIFI/BT/NFC 等功能，并清理不在运行列表中的应用。待机时会自行断开网络，在退出待机时，恢复网络连接。在该模式下，应用转入后台后将有可能无法接收网络数据，直到重新打开该应用。

其具体策略包括：

- 使能低电量模式的相关策略。
- 只允许指定的几个应用运行（针对第三方应用）。
- 进入该模式后，清理不允许运行的应用（主要针对第三方应用）。
- 待机后网络断开，亮屏后如果之前网络打开，则自动连接网络。

超级省电模式下，桌面和锁屏界面都显示为黑色，如图 2-8；允许添加/删除应用，如图 2-9；下拉状态栏只保留 WIFI、移动数据连接以及省电模式切换的快捷菜单，如图 2-10。在进入超级省电模式时，会有提示，而退出时同样会有提示，如图 2-11。

图2-8 超级省电模式桌面及锁屏界面

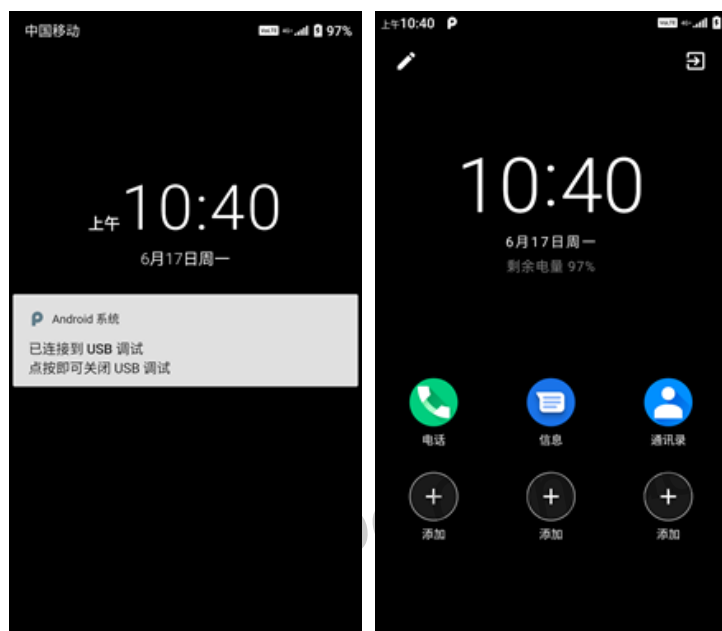


图2-9 超级省电模式允许添加/删除应用

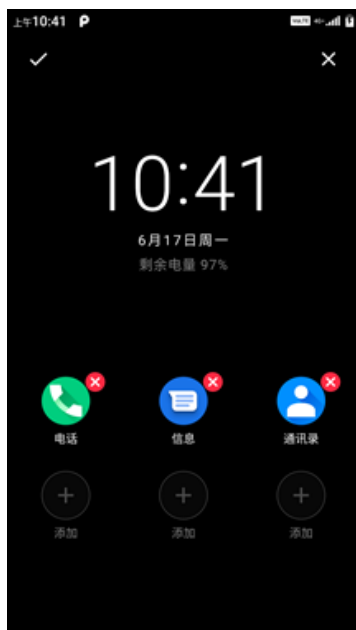
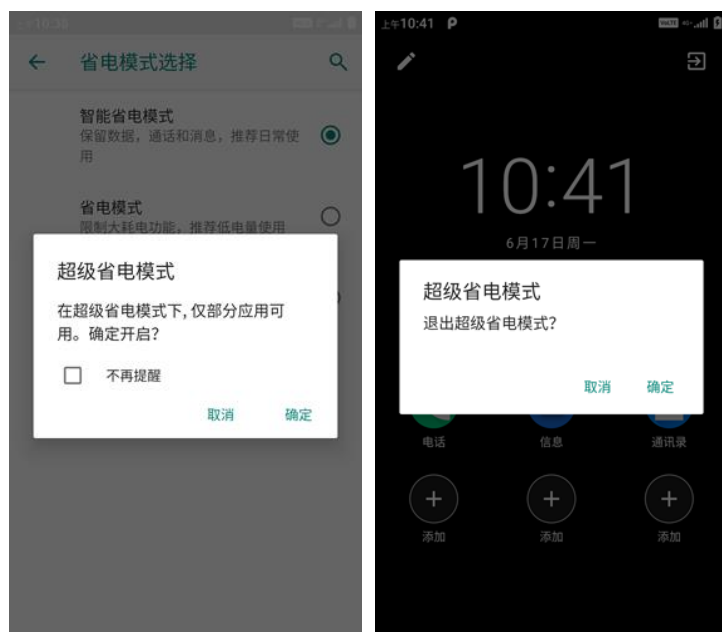


图2-10 超级省电模式下拉状态栏



图2-11 进入/退出超级省电模式的提示框



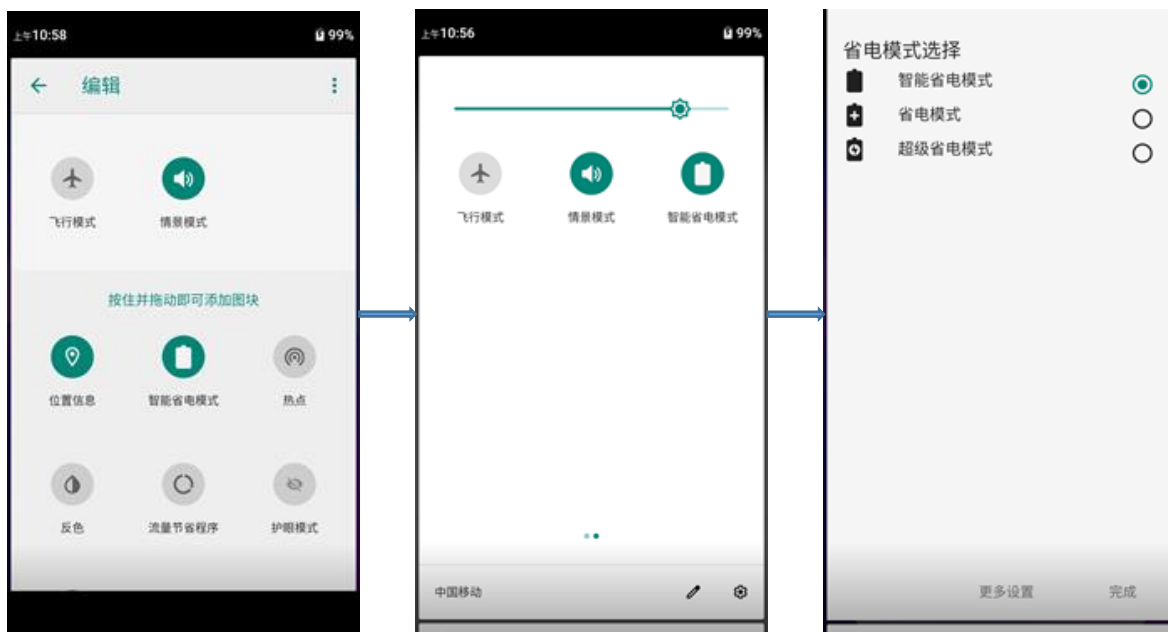
省电模式手动设置入口

可以在设置菜单中设置省电模式，如图 2-12 所示；也可以从下拉状态栏中的电池选项设置省电模式，如图 2-13 所示。

图2-12 省电模式设置入口



图2-13 省电模式下拉状态栏入口



2.3 定时省电

定时省电提供指定时间段自动进入指定省电模式的界面配置，定时省电选项如图 2-14 所示；设置界面如图 2-15 所示。

图2-14 定时省电选项

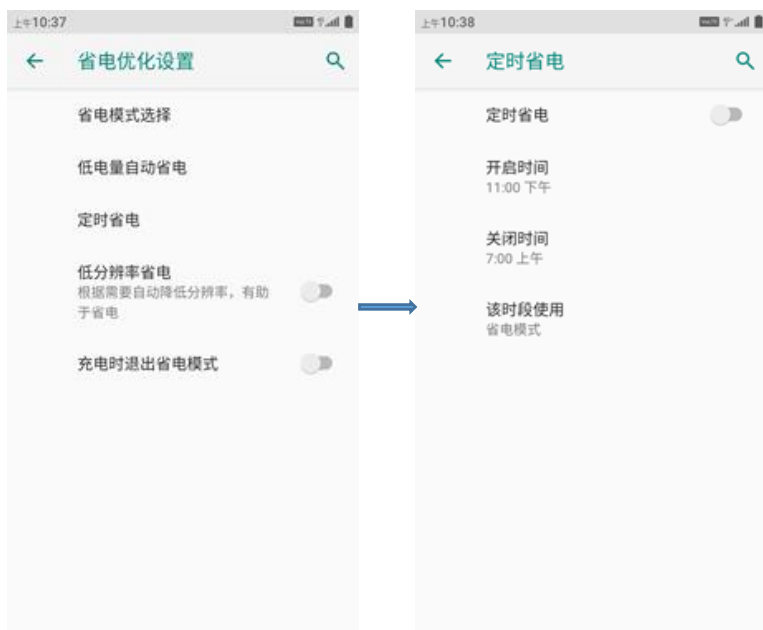
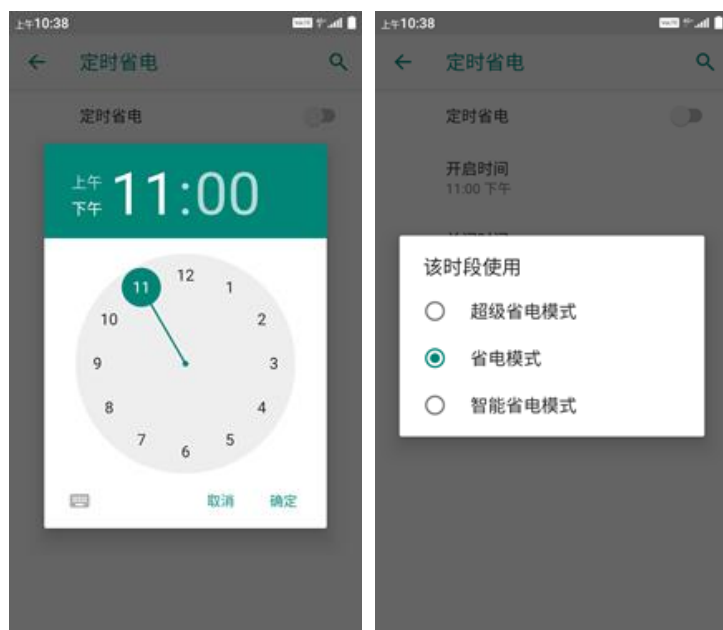


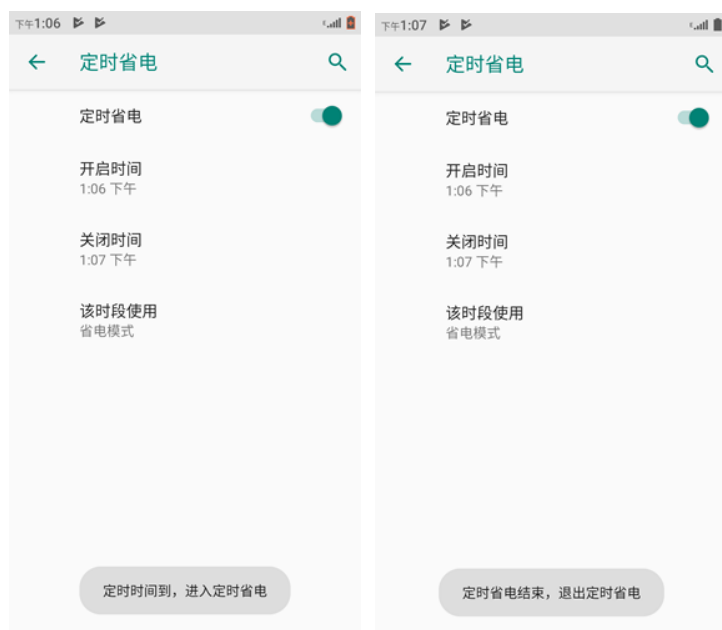
图2-15 定时省电设置界面



- 定时省电：该开关控制定时省电功能是否使能，打开后
 - 在当前时间大于指定的开始时间时，会自动进入指定的省电模式。
 - 在当前时间大于指定的关闭时间时，会退出指定的省电模式，而回到之前用户设置的省电模式。
- 开启时间和关闭时间：分别用于设置指定时间段的开始时间和关闭时间。
- 该时段使用：该项用于设置进入的省电模式，包括智能省电模式、省电模式和超级省电模式。

定时省电打开后，在指定的时间段内会自动进入指定的省电模式；在过了时间段后，会自动退出，返回到原来的省电模式，同时会有相关的 toast 提示。如图 2-16 所示。

图2-16 定时省电自动进入/退出的提示



2.4 低电量自动省电

低电量自动省电提供低电量时自动切换指定省电模式的界面配置。低电量自动省电选项如图 2-17 所示，设置界面如图 2-18。

图2-17 低电量自动省选项

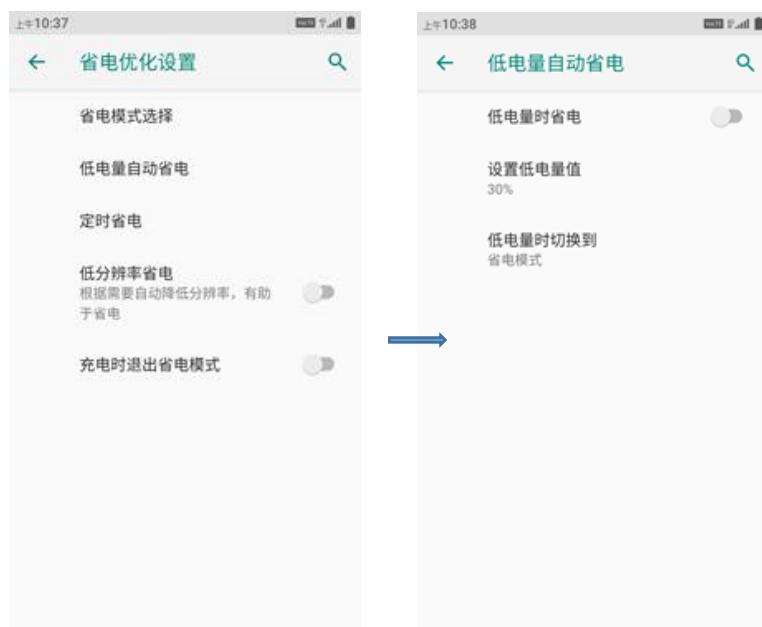
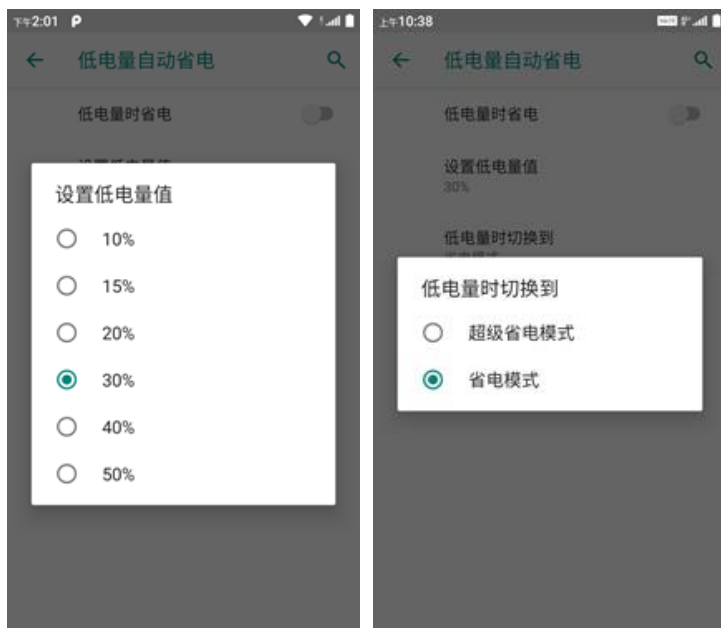


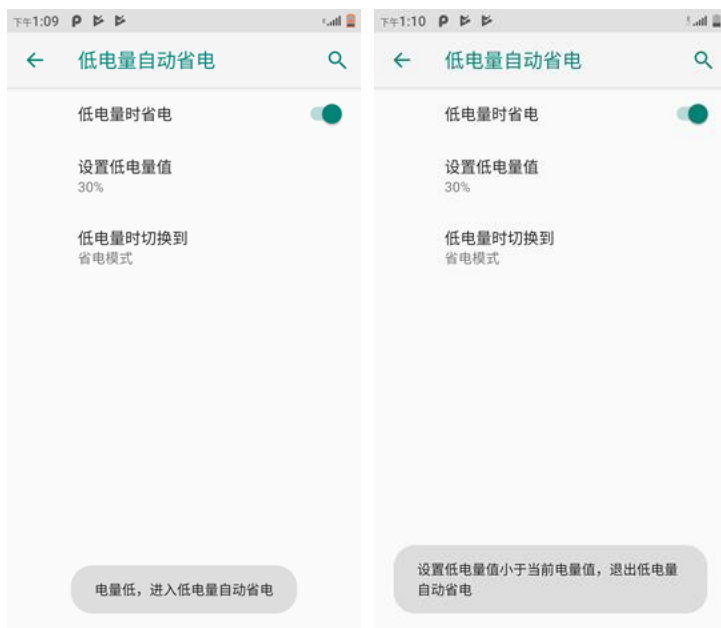
图2-18 低电量自动省电设置界面



- 设置低电量值：用于指定当电量低于多少时进行模式的切换。
- 低电量时切换到：用于指定切换的省电模式，可以选择超级省电模式或者省电模式。

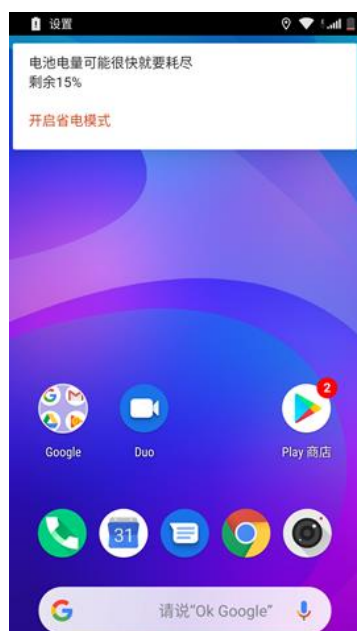
当低电量自动省电功能打开后，当电量低于设定阈值时，系统会自动切换到设定的模式；当电量增高直至超过阈值 5% 时，会自动退出“低电量自动省电”设置的模式返回到用户之前在“省电模式选择”里设置的模式。如果是用户直接将阈值设置到低于当前电量时，不论低多少，都会退出“低电量自动省电”设置的模式。在自动切换模式时，会有 toast 提示。如图 2-19 所示。

图2-19 低电量自动省电进入/退出的提示



当电量低于 15%，且系统没有处于省电模式，此时状态栏会弹出低电量提示，提示用户是否进入省电模式。如图 2-20 所示，该提示在插入电源或者进入省电模式或超级省电模式后，会消失。当退出省电模式或超级省电模式后不会再弹出，需要插拔电源才会再次弹出。

图2-20 低电量提示



2.5 自启动管理

自启动管理用于管理应用的开机自启动、后台自启动、关联自启动。应用自启动的管理，以包名（应用名）进行限制，不区分 user（用户）。自启动管理选项如图 2-21 所示，设置界面如图 2-22 所示。

图2-21 自启动管理选项

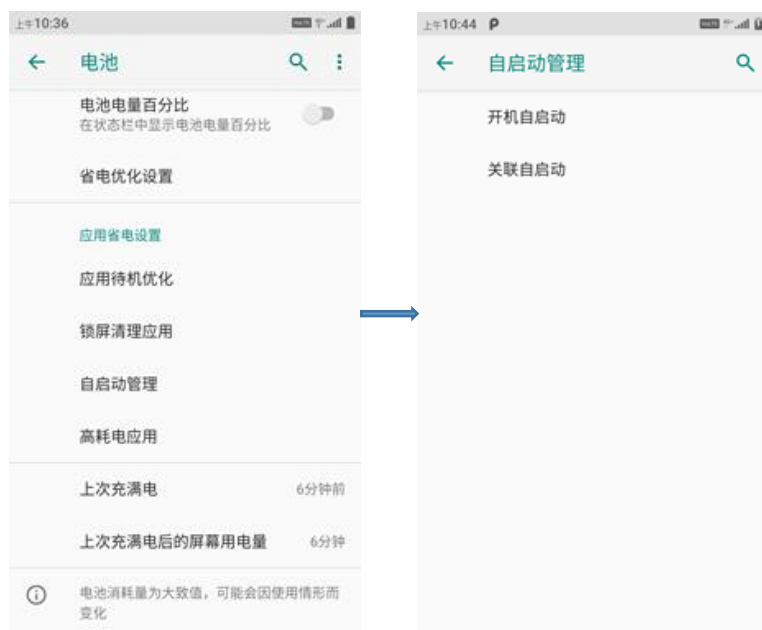


图2-22 自启动管理设置界面



- 自启动：指开机自启动和后台自启动。应用可能会监听系统的一些开机广播，从而在系统开机后自动进行启动。同时应用也可能会监听系统的任何广播，如网络连接的广播，从而在网络连接上的时候，进行自启动。
- 关联启动：关联启动指不同的应用之间进行互起。这里指的是，该应用被其他应用后台启动。比如打开百度浏览器，百度浏览器就会在后台自己启动爱奇艺/百度视频等百度系的应用。

系统默认**允许**应用自启动或者被其他应用关联启动。用户可以从上述界面进行“禁止”设置。

说明

默认允许第三方应用自启动，对一些安装了多个三方应用的 UE 性能和功耗场景会有消极影响。体现在应用启动，滑动等场景体验变差；只打开少数应用的待机功耗场景，其待机功耗会变大；但默认禁止三方应用自启动会与 google CDD 里相关规定冲突。具体参见 [3.5 修改自启动管理默认行为](#) 里的说明。

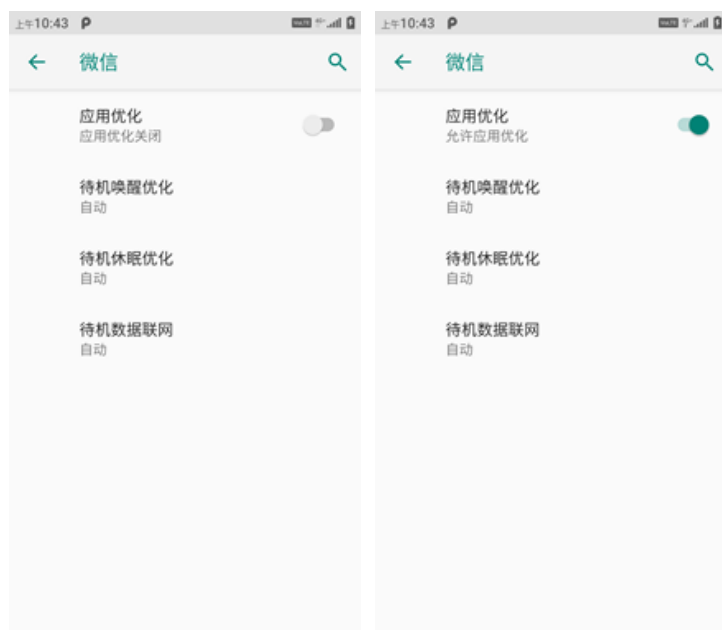
2.6 应用待机优化

应用待机优化界面提供针对应用的各项省电设置。应用待机优化选项如 [图 2-23](#) 所示，设置界面如 [图 2-24](#) 所示。

图2-23 应用待机优化选项



图2-24 应用待机优化设置界面



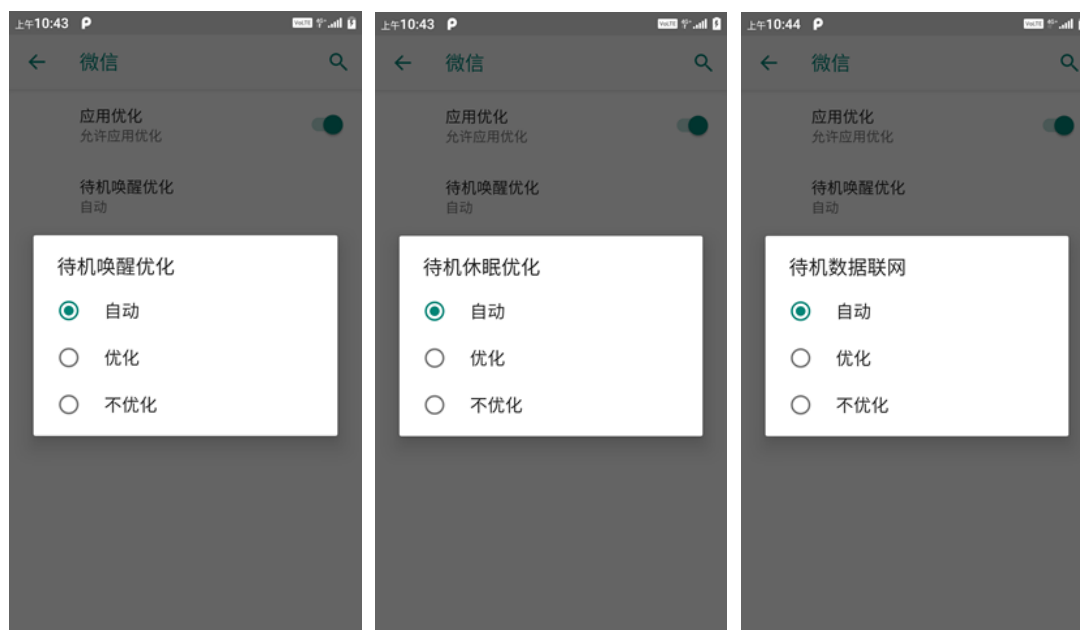
- 应用优化的总开关控制对该应用是否进行省电优化。
 - 当关闭时，表示不对该应用进行任何形式的省电优化。
 - 当打开时，表示会对该应用进行省电优化。

用户还可以对待机唤醒优化、待机休眠优化和待机数据联网这三项进行单独设置。

- 待机唤醒优化：会减少应用唤醒系统的频率，从而降低待机功耗。是应用 **alarm** 的一项优化，使用心跳对齐机制优化应用的 **Alarm** 唤醒频率。
- 待机休眠优化：会减少应用阻止系统睡眠的时间，从而降低待机功耗。是应用 **wakelock** 持锁的一项优化，限制应用最长持锁时间。
- 待机数据联网：会控制应用后台数据访问的频率，从而降低待机功耗。是应用后台频繁访问网络的优化。

每项优化选项都有“自动/优化/不优化”3个设置项，如图 2-25。

图2-25 应用待机优化各项设置界面



- 自动：省电策略根据系统和应用的运行状态自行决定是否对应用进行相关项的优化。
- 优化：由用户设置为优化后，将直接对应用进行相关项的优化，而不需要再考虑其他状态信息。
- 不优化：由用户设置为不优化后，将不再对应用做相关项的优化操作。

应用待机优化默认为**允许优化**，其中待机唤醒优化、待机休眠优化和待机数据联网默认是**自动**，即系统根据系统和应用状态自行做出决策。

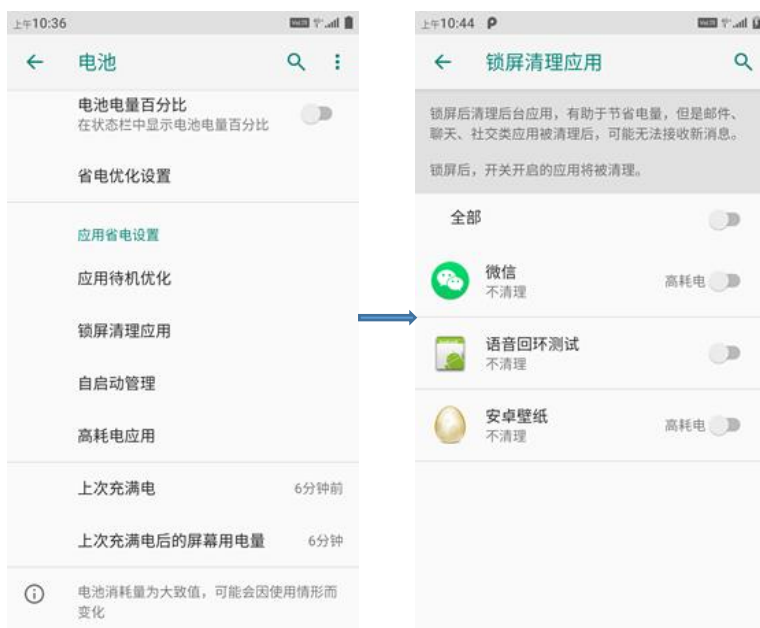
说明

可以对这些默认配置进行修改，参见 [3.6 修改应用待机优化默认配置](#)。

2.7 锁屏清理应用

锁屏清理应用指在锁屏待机后，根据对应用的设置对应用进行清理操作。其界面如[图 2-26](#) 所示。

图2-26 锁屏清理应用界面



- 锁屏清理的初始默认设置都是不清理。用户可以通过界面进行设置。
- 可以通过设置属性“persist.sys.pwctl.bgclean.to”的值，单位 ms，来设置灭屏待机多长时间后开始清理（默认是 1 分钟）。因为这里没有采用可唤醒的 alarm，所以实际清理的时间会大于属性设置的值。
- 关于锁屏清理，需要关注下面几点：
 - 锁屏清理目的是减少待机应用从而减少待机功耗。锁屏清理是在待机一段时间后才开始进行，该时间值大于 1 分钟。出于功耗考虑没有采用可唤醒的 alarm 来设置该 1 分钟定时器，而是采用非可唤醒的 alarm。而非可唤醒 alarm 需要有可唤醒 alarm 才会触发，因此这会导致开始清理的时间远大于 1 分钟。但一般 10 分钟内都会有可唤醒 alarm，也就是 10 分钟内会开始清理。
 - 一些应用不会被清理，即使在已经被设置为清理的情况下。这些应用包括：
 - 输入法应用
 - 默认的系统服务应用，如壁纸服务应用，默认短信服务应用等
 - 当前灭屏前可见的应用
 - 正在播放音乐的应用
 - 正在下载的应用
 - 内置系统应用

上述这些类型的应用不能清理，清理后会导致使用上的体验问题。

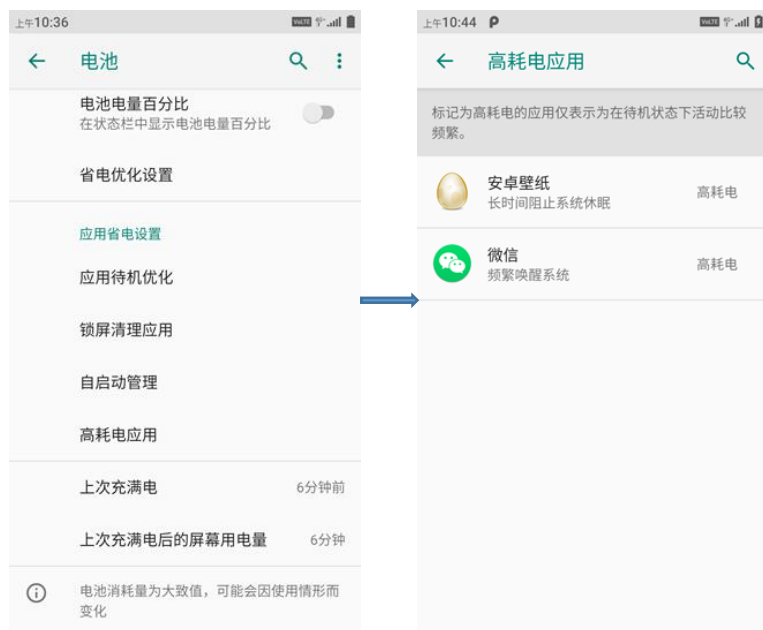
说明

查看应用是否被清理，不能通过 recent 应用来查看，recent 应用是指最近使用过的应用，在 recent 应用列表里并不表示该应用还在后台运行。要查看应用是否被清理，可以通过 ps 命令查看，或者通过“开发者模式→正在运行的服务”查看。

2.8 高耗电应用

高耗电应用界面用于显示对待机功耗影响较大的应用列表，如图 2-27 所示。

图2-27 高耗电应用界面



影响待机功耗的应用，主要从以下方面考虑：

- 频繁设置 alarm 唤醒系统
- 频繁使用 gps
- 长时间持锁导致系统无法休眠

当有应用在后台持续异常持锁超过 30 分钟，此时会有提示通知用户：发现后台高耗电应用。

2.9 充电时退出省电模式

该开关用于控制在充电时是否退出省电模式（包括超级省电模式）回到智能模式下。界面如图 2-28 所示。

图2-28 充电时退出省电模式界面



- 开关关闭时，即使充电也会一直停留在当前模式，不管当前模式是否是省电模式（包括超级省电模式）。
- 开关打开时，充电时如果当前是省电模式（包括超级省电模式），则会退回到智能模式下，并且设置菜单中无法进行模式选择。拔出电源时，恢复到原先的省电模式（包括超级省电模式）。

2.10 低分辨率省电

该功能仅在支持高分辨率的平台上才支持。

当打开该功能时，显示分辨率会切到较低的分辨率上，如由 1080p 切换到 720p。可以一定程度上减少系统功耗。相关设置界面如[图 2-29](#) 所示。

图2-29 低分辨率省电界面



Unisoc Confidential For hiar

3

相关配置

3.1 恢复 Android 原生省电功能

- debug 版本调试时，可以直接设置下列属性为 0，恢复原生省电功能：

- “persist.sys.pwctl.enable”
- “ro.sys.pwctl.ultrasaving”

当只需要关闭超级省电功能时，则只需设置 “ro.sys.pwctl.ultrasaving=0”。

- 编译配置，从配置文件中进行配置。
 - 在 common/DeviceCommon.mk 中将 TARGET_PWCTL_ULTRASAVING 设置为 false，并添加设置属性 persist.sys.pwctl.enable 为 0。如下：

```
# Power framework ultra-saving mode
TARGET_PWCTL_ULTRASAVING ?= false
ifeq (true,$(strip $(TARGET_PWCTL_ULTRASAVING)))
# Add PowerSaveModeLauncher
PRODUCT_PACKAGES += \
    PowerSaveModeLauncher
endif
```

```
# Power save mode
PRODUCT_PROPERTY_OVERRIDES += \
    persist.sys.pwctl.enable=0
```

- 在 common/features/base/config.mk 中将 TARGET_PWCTL_ULTRASAVING 设置为 false，如下：

```
##powercontroller-ultrasaving start##
TARGET_PWCTL_ULTRASAVING ?= false
ifeq (true,$(strip $(TARGET_PWCTL_ULTRASAVING)))
# ultrasaving mode
ADDITIONAL_BUILD_PROPERTIES += \
    ro.sys.pwctl.ultrasaving=1
else
ADDITIONAL_BUILD_PROPERTIES += \
    ro.sys.pwctl.ultrasaving=0
endif
```

3.2 心跳对齐配置

3.2.1 心跳对齐功能开关

- 关闭心跳对齐功能：

```
adb root
```

```
adb shell
setprop persist.sys.heartbeat.enable 0
getprop | grep persist.sys.heartbeat.enable //确认修改成功
[persist.sys.heartbeat.enable]: [0]
reboot //记得重启手机
```

- 开启心跳对齐功能：

该功能默认是开启的。如果使用上述方法进行关闭后，需要重新开启。相关命令如下：

```
adb root
adb shell
setprop persist.sys.heartbeat.enable 1
getprop | grep persist.sys.heartbeat.enable //确认修改成功
[persist.sys.heartbeat.enable]: [1]
reboot //记得重启手机
```

- 编译配置，从配置文件中关闭心跳对齐功能。

可以在 common/DeviceCommon.mk 中设置属性 persist.sys.heartbeat.enable 为 0，配置如下：

```
# Disable Alarm alignment
PRODUCT_PROPERTY_OVERRIDES += \
    persist.sys.heartbeat.enable=0
```

3.2.2 预置黑名单

心跳对齐调整机制是自动学习应用设置的 alarm 频率，将频繁的 alarm 进行对齐调整。当将应用包名添加到黑名单后，来自该应用的所有 alarm 将在待机后直接被对齐调整，而不再经历学习过程，即不再考虑该 alarm 是否是频繁设置，而是直接进行对齐调整。

黑名单保存在文件/system/etc/blackAppList.xml 里。

黑名单文件示例：

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<app_list>
<packageItem>
<pkg_name>com.sohu.sohuvideo</pkg_name>
</packageItem>
<packageItem>
<pkg_name>com.tencent.hero</pkg_name>
</packageItem>
<packageItem>
<pkg_name>com.tencent.pao</pkg_name>
</packageItem>
<packageItem>
<pkg_name>com.tiantian.android.player.app</pkg_name>
</packageItem>
</app_list>
```

3.2.3 菜单设置

通过如下菜单：设置-->电池-->应用待机优化-->选择具体的应用-->待机唤醒优化，有 3 个选项，具体含义参见 2.6 应用待机优化。

- 默认为“自动”。
- 当用户希望在待机后对该应用在待机后设置的 alarm 都进行对齐的话，可以选择“优化”。

- 当用户不希望在此待机时对该应用的 alarm 进行对齐操作，则可以选择“不优化”。

3.3 第三方应用默认配置

配置文件 vendor/sprd/modules/power/fw-power-config/appPowerSaveConfig.xml 预置保存在/system/etc/目录下。系统启动时，会检查/data/system/目录下是否有该文件，如果没有会从/system/etc/下读取，若都没有，上述各功能使用默认的策略。因此需要预置对应策略时，可以通过配置该配置文件来达到。

该配置文件针对应用省电优化的格式内容如下：

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<app_powersave_config>
<package name="com.kugou.android" optimize="1" alarm="0" wakelock="0" network="0" autolaunch="2"
secondarylaunch="1" lockscreencleanup="0" consumertype="0" />
</app_powersave_config>
```

其中相关参数见表 3-1。

表3-1 配置文件参数

参数	说明
name	指定对应的 App 名称。
optimize	对应是否对该应用进行省电优化的总开关。 <ul style="list-style-type: none"> 0 为不优化 1 为优化
alarm	对应待机唤醒优化。 <ul style="list-style-type: none"> 0 为自动 1 为优化 2 为不优化
wakelock	对应待机休眠优化。 <ul style="list-style-type: none"> 0 为自动 1 为优化 2 为不优化
network	对应待机数据联网。 <ul style="list-style-type: none"> 0 为自动 1 为优化 2 为不优化
autolaunch	对应应用自启动。 <ul style="list-style-type: none"> 1 为禁止 2 为允许
secondarylaunch	对应应用关联自启。

参数	说明
	<ul style="list-style-type: none"> 1 为禁止 2 为允许
lockscreencleanup	对应锁屏清理。 <ul style="list-style-type: none"> 1 为清理 2 为不清理
consumertype	对应该应用的耗电情况。 <ul style="list-style-type: none"> 0 为耗电情况未知 0x01 表示 alarm 频繁唤醒 0x02 表示长时间持锁 0x04 表示长时间使用 gps 实际值可能为上述各值的组合

针对 Android 10.0，增加了 `exemptsource` 一项用于指定该配置的来源，预置在 `/system/etc/appPowerSaveConfig.xml` 中的各配置，需要设置 `exemptsource="0"`。

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
```

```
<app_powersave_config>
```

```
<package name="com.kugou.android" optimize="1" alarm="0" wakelock="0" network="0" autolaunch="2"
```

```
secondarylaunch="1" lockscreencleanup="0" consumertype="0" exemptsource="0"/>
```

```
</app_powersave_config>
```

3.4 超级省电默认应用添加

在 `vendor/sprd/modules/power/fw-power-config/powercontroller.xml` 中添加对应应用。

如下面默认添加的是“拨号/联系人”。添加的格式为“`component 名#位置`”，`com.android.dialer/.app.DialtactsActivity` 是拨号应用对应 `component name`，0 指对应添加到超级省电允许应用列表界面的第一个位置。

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
```

```
<powercontroller>
```

```
<applist_in_ultrasave>
```

```
<appname>com.android.dialer/.app.DialtactsActivity#0</appname>
```

```
<appname>com.android.contacts/.activities.PeopleActivity#1</appname>
```

```
</applist_in_ultrasave>
```

```
</powercontroller>
```

3.5 修改自启动管理默认行为

可以通过下面方式修改自启动管理的默认行为，将默认允许第三方应用自启动修改为禁止自启动。

- 方法一：

按照下面方式修改 *AppPowerSaveConfig.java*，其文件地址为：
frameworks/base/core/java/android/os/sprdpower/AppPowerSaveConfig.java

```
- private final static int AUTOLAUNCH_DEF = SystemProperties.getInt(AUTOLAUNCH_DEF_PROP,
VALUE_NO_OPTIMIZE);
- private final static int SECONDARYLAUNCH_DEF = SystemProperties.getInt(SECONDARYLAUNCH_DEF_PROP,
VALUE_NO_OPTIMIZE);
+ private final static int AUTOLAUNCH_DEF = SystemProperties.getInt(AUTOLAUNCH_DEF_PROP,
VALUE_OPTIMIZE);
+ private final static int SECONDARYLAUNCH_DEF = SystemProperties.getInt(SECONDARYLAUNCH_DEF_PROP,
VALUE_OPTIMIZE);

private static int[] mDefConfig = {VALUE_OPTIMIZE, // for optimize
    VALUE_AUTO, // for alarm
    VALUE_AUTO, // for wakelock
    VALUE_AUTO, // for network
-    VALUE_NO_OPTIMIZE, // for autolaunch
-    VALUE_NO_OPTIMIZE, // for 2ndlaunch
+    VALUE_OPTIMIZE, // for autolaunch
+    VALUE_OPTIMIZE, // for 2ndlaunch
    VALUE_AUTO, // for lockscreen cleanup
```

• 方法二：

在 device 目录下添加设置属性 persist.sys.pwctl.auto 和 persist.sys.pwctl.secondary 的默认值为 1，其中 1 为禁止，2 为允许。

📖 说明

将应用自启动修改为默认禁止后，会与 google CDD 中的规定冲突。Android Q CDD 中 3.2.3.4. Broadcast Intents 一节规定第三方应用可以接收到指定的 broadcast，其中包括 ACTION_BOOT_COMPLETED。

- 该规定参见链接：https://source.android.com/compatibility/10/android-10-cdd#3_2_3_intent_compatibility
- 指定的 Broadcast 列表参见链接：<https://developer.android.com/about/versions/11/reference/broadcast-intents-30>

3.6 修改应用待机优化默认配置

应用待机优化默认为允许优化，其中待机唤醒优化、待机休眠优化和待机数据联网默认为“自动”，即系统根据系统和应用状态进行决策。可以通过下面方法来修改上述的默认配置。

按照下面方式修改 *AppPowerSaveConfig.java*，其文件地址为：
frameworks/base/core/java/android/os/sprdpower/AppPowerSaveConfig.java

```
- private static int[] mDefConfig = {VALUE_OPTIMIZE, // for optimize
+ private static int[] mDefConfig = {0, // for optimize, 0 for not optimize
    VALUE_AUTO, // for alarm
    VALUE_AUTO, // for wakelock
    VALUE_AUTO, // for network
    VALUE_NO_OPTIMIZE, // for autolaunch
    VALUE_NO_OPTIMIZE, // for 2ndlaunch
```

VALUE_AUTO, // for lockscreen cleanup

- 当“optimize”设置为 0 时，表示对应用不做待机优化，此时“for alarm”，“for wakelock”，“for network”这 3 项无论设置成什么值都不会起作用。
- 当“optimize”设置为 **VALUE_OPTIMIZE** 时，即对应用做待机优化时，可以分别对“for alarm”，“for wakelock”，“for network”这 3 项做对应配置。
 - **VALUE_OPTIMIZE** 对应直接优化
 - **VALUE_NO_OPTIMIZE** 对应不优化
 - **VALUE_AUTO** 对应系统自动决策

Unisoc Confidential For hiar

4

测试建议

省电管理功能，对应用做了诸多限制，包括：

- 待机 alarm 对齐
- 待机后台数据访问
- 待机后台持锁
- 待机后台 GPS 访问
- 应用的关联启动

因此应用的健全测试很重要。相关的应用健全测试建议务必涵盖下面这些方面：

- 闹钟等定时类应用，能正常工作。
- 即时通讯类应用，在待机后能正常接收信息。建议除了每隔一段时间进行验证外，增加长时间待机的情况，如待机几个小时或一个晚上后，再确认是否能接收信息。同时验证国内外常用应用。
- 音乐类应用，如 FM，音乐播放器，待机后台本地播放音乐和待机后台在线听音乐。
- 应用待机后台下载。
- 导航类应用，如地图导航。
- 骑行类应用，如摩拜单车骑行等。
- 运动类应用，如 keep 跑步记录等。
- 其他可能因待机限制 wakelock/gps/网络而受影响的场景。