Doze and APP Standby

Analysis & Design

MT6000

Doc No: DS6000-E13A-DMT-V1.1EN
Version: V1.1
Release date: 2016-08-28
Classification: Internal

Keywords

Doze & App Standby Design Document

MediaTek Inc.

Postal address

No. 1, Dusing 1st Rd. , Hsinchu Science Park, Hsinchu City, Taiwan 30078

MTK support office address

No. 1, Dusing 1st Rd. , Hsinchu Science Park, Hsinchu City, Taiwan 30078

Internet

http://www.mediatek.com/

Document Revision History

Revision	Date	Author	Description
V1.0	2016-08-28	Cuihua Li	Initial Release

Table of Contents

Lists of Tables

Lists of Figures

1 Introduction

1.1 Purpose

This document describes the overall design of doze and app standby, how enable /disable and some FAQ.

1.2 Scope

This design document is common part for Android Platform (start from Android M).

1.3 Who Should Read This Document

This document is primarily intended for:

- Engineers with technical knowledge of LowPower
- Customers who want to use this feature for saving Power of device

2 Abbreviations

Please note the abbreviations and their explanations provided in Table 2-1. They are used in many fundamental definitions and explanations in this document and are specific to the information that this document contains.

Table 2-1. Abbreviations

Abbreviations	Explanation
MTK	MediaTek, Asia's largest fabless IC design company.

3 Doze

3.1 What is Doze

Starting from Android 6.0 (API level 23), Android introduces two power-saving features(Doze & App Standby) that extend battery life for users by managing how apps behave when a device is not connected to a power source. If a user leaves a device unplugged and stationary for a period of time, with the screen off, the device enters Doze mode. In Doze mode, the system attempts to conserve battery by restricting apps' access to network and CPU-intensive services. It also prevents apps from accessing the network and defers their jobs, syncs, and standard alarms.Periodically, the system exits Doze for a brief time to let apps complete their deferred activities. During this maintenance window, the system runs all pending syncs, jobs, and alarms, and lets apps access the network.

Please see figure 4-1 for learn the doze.

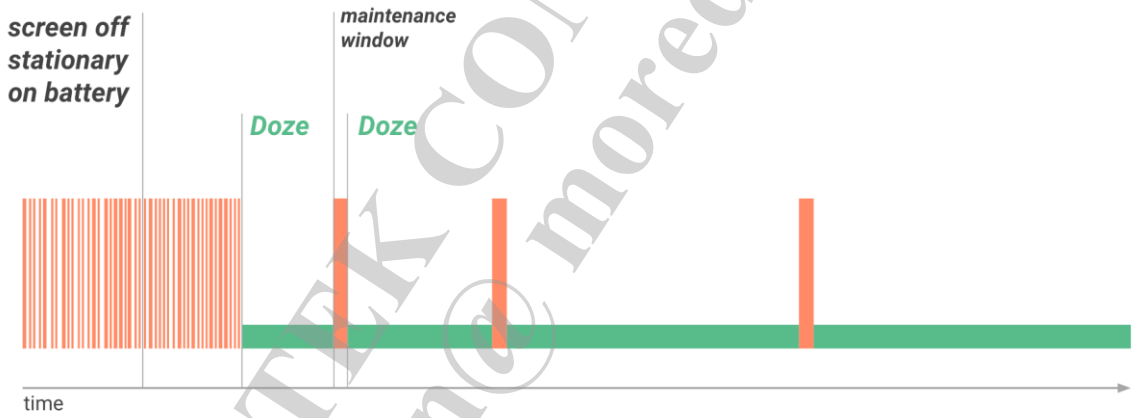

Figure 3-1. Doze maintenance window.

3.2 Description Doze restrictions

The following restrictions apply to your apps while in Doze:

- Network access is suspended.
- The system ignores wake locks.
- Standard AlarmManager alarms (including setExact() and setWindow()) are deferred to the next maintenance window.If you need to set alarms that fire while in Doze, use setAndAllowWhileIdle() or setExactAndAllowWhileIdle().Alarms set with setAlarmClock() continue to fire normally — the system exits Doze shortly before those alarms fire.

- The system does not perform Wi-Fi scans.
- The system does not allow sync adapters to run.

3.3 Testing your app with Doze

You can test Doze mode by following these steps:

- Configure a hardware device or virtual device with an Android 6.0 (API level 23) or higher system image.
- Connect the device to your development machine and install your app.
- Run your app and leave it active.
- Force the system into idle mode by running the following command:
  - adb shell dumpsys battery unplug
  - adb shell dumpsys deviceidle step
  - adb shell dumpsys deviceidle

3.4 Doze State Machine

There are many states for doze,the conditions of these states are as fllows.The network,waklock .etc(refer 4.2) will be restricted while in idle state.For idle maintenance state,the system exits idle state for a brief time to let apps complete their deferred activities.

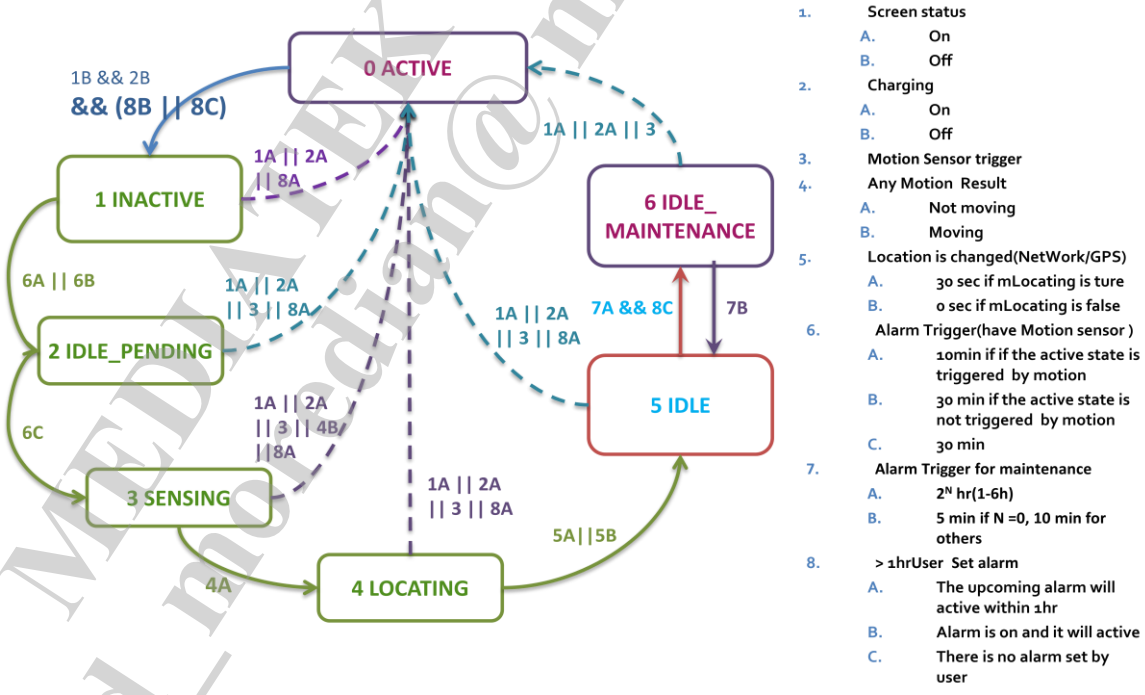

Figure 3-2. Doze state machine.

3.5 Android N Behavior

Android 7.0 brings further enhancements to Doze by applying a subset of CPU and network restrictions while the device is unplugged with the screen turned off, but not necessarily stationary, for example, when a handset is traveling in a user's pocket. It is named Light Doze. When a device is on battery power, and the screen has been off for a certain time, the device enters Light Doze and applies the first subset of restrictions: It shuts off app network access, and defers jobs and syncs. If the device is stationary for a certain time after entering Full Doze, the system applies the rest of the Doze restrictions to PowerManager.WakeLock, AlarmManager alarms, GPS, and Wi-Fi scans.

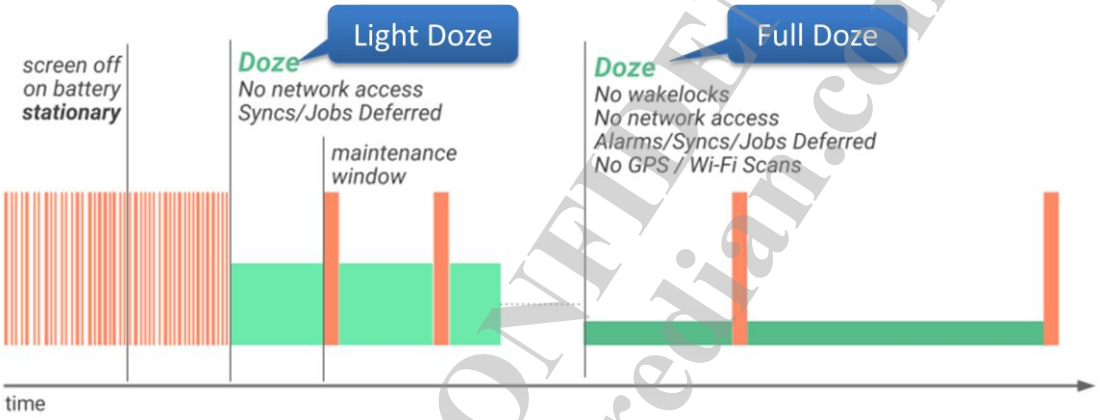

Figure 3-3. Light doze and full doze.

The light doze has its own state machine,which is shown as follows,

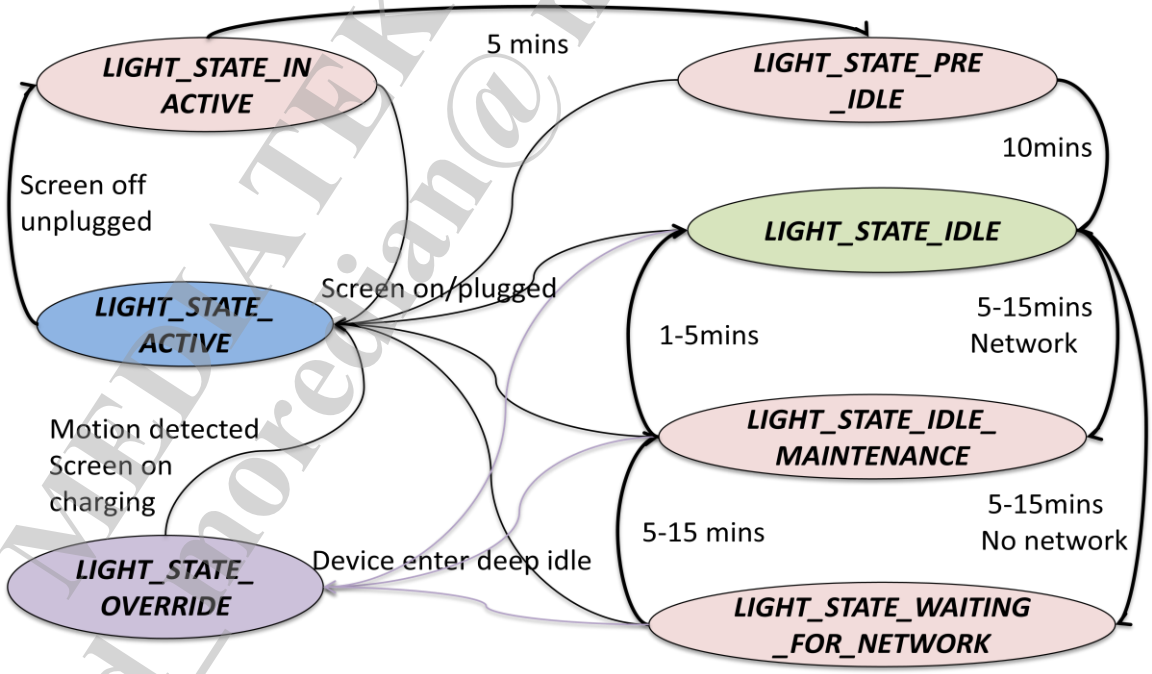

Figure 3-4. Light doze state machine.

4 App standby

4.1 What is App standby

App Standby defers background network activity for apps with which the user has not recently interacted.It will be restricted by buckets while app standby:

Bucket	Jobs (run for 10mins max)	Alarms (run 10sec max)	Network (parole duration is 10mins)	FCM High Pri.
ACTIVE	No restriction	No restriction	No restriction	No restriction
WORKING_SET	Every 2 hours	Every 6 min	No restriction	No restriction
FREQUENT	Every 8 hours	Every 30 min	No restriction	10/day & then demoted to normals
RARE	Every 24 hours	Every 2 hours	Every 24 hours	5/day & then demoted to normals
NEVER	Never	Never	Never	Never

You can manually test App Standby using the following ADB commands:

- adb shell am set-standby-bucket <PACKAGE> active|working_set|frequent|rare
- adb shell am get-standby-bucket <PACKAGE>

4.2 App Standby Key flow

The app standby key flow is as follows,

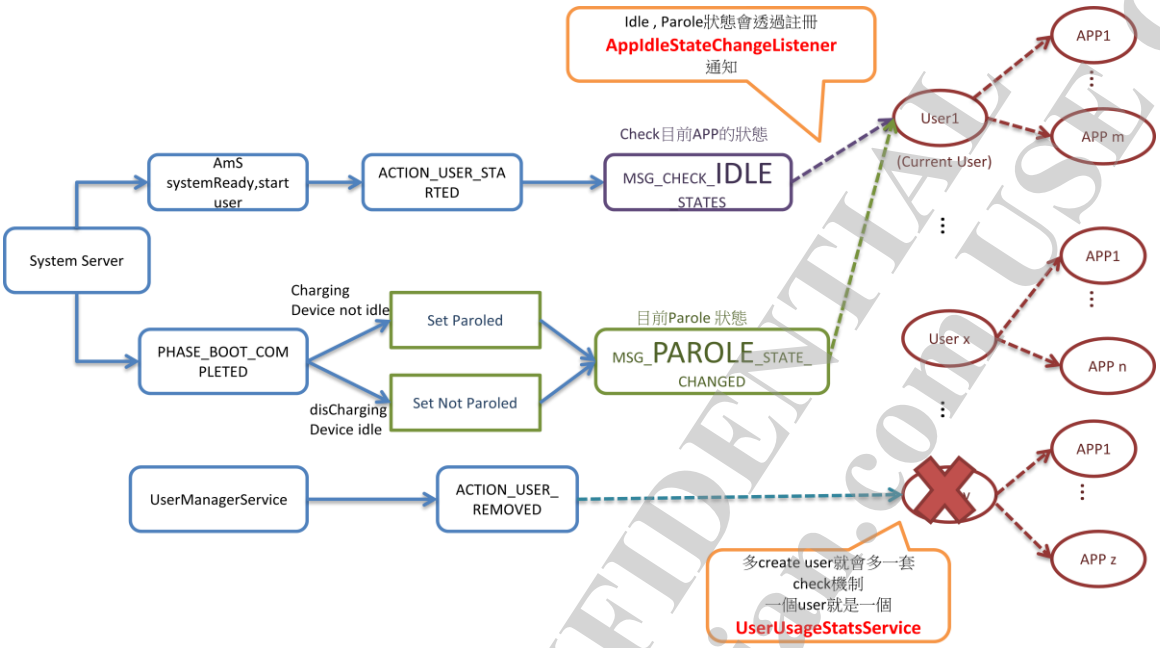

Figure 4-1. App Standby flow.

App standby's state machine is as follows,

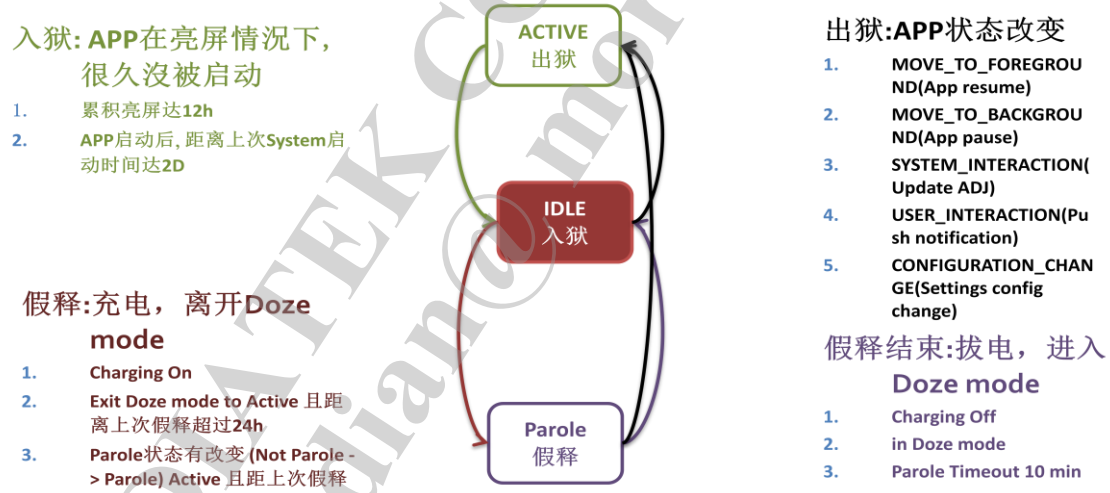

Figure 4-2. App Standby state machine.

5 FAQ

5.1 How enable doze/app standby

1.Doze/app standby controls by config_enableAutoPowerModes in frameworks/base/core/res/res/values/

config.xml.set config_enableAutoPowerModes is true

we can also use command:adb shell dumpsys deviceidle enable

5.2 How to config whitelist for doze

Doze have whitelist to filter the app which not be restricted by doze. We can config the whitelist:

For system applications,we can set the whitelist by /system/etc/sysconfig/xxx.xml,need the key “allow-in-power-save”,such as,

- <allow-in-power-save package="com.google.android.gms" />
- <allow-in-power-save package="com.android.vending" />
- <allow-in-power-save package="com.google.android.volta" />

For third part applications,we can set it by settings UI(Settings -> Battery -> Battery optimizations), or we can use adb command,such as,

- adb shell dumpsys deviceidle whitelist +com.example.myapplication