SC60 Android Boot Speed Acceleration Guide

Smart LTE Module Series

Rev. SC60_Android_Boot_Speed_Acceleration_Guide_V1.0

Date: 2017-10-12

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.
7th Floor, Hongye Building, No.1801 Hongmei Road, Xuhui District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local office. For more information, please visit:
http://quectel.com/support/sales.htm

For technical support, or to report documentation errors, please visit:
http://quectel.com/support/technical.htm
Or email to: support@quectel.com

GENERAL NOTES

QUECTEL OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS' REQUIREMENTS. QUECTEL MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. QUECTEL DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2017-10-12	Hank HAN	Initial

Contents

1 Introduction

This document mainly introduces how to make Quectel SC60 module boot up faster through optimizing the boot time of its Android system, and how to calculate the boot time according to the logs.

2 Optimize Boot Time

This chapter mainly introduces how to optimize the boot time of SC60 module.

1. First, use the following three commands to compile the software of SC60 as user mode.

```
source build/envsetup.sh
lunch msm8953_64-user
make –jn
```

   The “n” in the above command “make –jn” means the thread number of CPU.

2. After compiling the software of SC60 as user mode, download the generated BIN files into SC60. Then power on SC60 and save the kernel logs via debug UART port.

NOTE

The first boot of Android after downloading the generated BIN files into SC60 would take too much time, please do not save the kernel logs or user space logs generated during the first boot.

3. Remove the following config items in the file *$ANDROID_TOP \kernel\msm-3.18\arch\arm64\configs\msmcortex-perf_defconfig*.

```
CONFIG_SERIAL_MSM_HSL_CONSOLE=y
CONFIG_MSM_ADSPRPC=y
CONFIG_CORESIGHT=y
CONFIG_CORESIGHT_EVENT=y
CONFIG_CORESIGHT_FUSE=y
CONFIG_CORESIGHT_CTI=y
CONFIG_CORESIGHT_CTI_SAVE_DISABLE=y
CONFIG_CORESIGHT_TMC=y
CONFIG_CORESIGHT_TPIU=y
CONFIG_CORESIGHT_FUNNEL=y
CONFIG_CORESIGHT_REPLICATOR=y
CONFIG_CORESIGHT_STM=y
CONFIG_CORESIGHT_HWEVENT=y
CONFIG_CP_ACCESS64=y
CONFIG_MSM_SMD_DEBUG=y
```

```
CONFIG_MSM_FORCE_WDOG_BITE_ON_PANIC=y
```

After recompiling the code, check the file *$ANDROID_TOP\out\target\product\msm8953_64\obj\KERNEL_OBJ\.config* to see whether the above modification take effect. If there are no above config items in the file, then the modification takes effect.

4. Remove the logs “earlycon=msm_hsl_uart,0x78af000” in file *$ANDROID_TOP\device\qcom\msm8953_64\BoardConfig.mk*.

3 Calculate Boot Time

After the kernel logs have been got according to the procedures 1 and 2 mentioned in ***Chapter 2***. You can modify the software code according to the procedures mentioned in ***Chapter 2***. Then recompile the software and download the generated BIN files into SC60. And the following command can be used to get the user space logs. Similarly, please note that do not save the user space logs generated during the first boot.

```
adb logcat -b events -d >boot_events.txt
```

The following figure shows the user space logs in each boot phase.

	Boot phases	Log messages
1	PBL/SBL/HYP/ LK boot loader	Bootloader time
2	Kernel boot up + Initialize native daemons	boot_progress_start
3	Zygote class preloading	boot_progress_preload_start boot_progress_preload_start boot_progress_preload_end boot_progress_preload_end boot_progress_system_run
4	Package scanning	boot_progress_pms_start boot_progress_pms_system_scan_start boot_progress_pms_data_scan_start boot_progress_pms_scan_end boot_progress_pms_ready
5	Service initialization	boot_progress_ams_ready
6	UI/Apps start	boot_progress_enable_screen
	Total boot up time	= Bootloader time (1) + Time from boot_progress_enable_screen (6)

Figure 1: User Space Logs in Each Boot Phase

In kernel logs, there are logs as below.

```
[      1.916497] KPI: Bootloader start count = 26514
[      1.920012] KPI: Bootloader end count = 65190
[      1.924323] KPI: Bootloader display count = 38956
```

```
[       1.929030] KPI: Bootloader load kernel count = 2585
[       1.933959] KPI: Kernel MPM timestamp = 154222    //C    bootloader end time
[       1.938401] KPI: Kernel MPM Clock frequency = 32768    //D    clock
```

So the bootloader time can be calculated as C/D-Kmsg(C) = 154222/32768-1.93 = 2.78s

In user space logs, there are logs as below.

```
01-01 00:00:09.549     764     764 I boot_progress_start: 7920
01-01 00:00:10.781     764     764 I boot_progress_preload_start: 9152
01-01 00:00:11.177     765     765 I boot_progress_preload_start: 9548
01-01 00:00:12.152     764     764 I boot_progress_preload_end: 10523
01-01 00:00:12.232     765     765 I boot_progress_preload_end: 10604
01-01 00:00:12.337    1575    1575 I boot_progress_system_run: 10708
01-01 00:00:12.647    1575    1575 I boot_progress_pms_start: 11018
01-01 00:00:12.923    1575    1575 I boot_progress_pms_system_scan_start: 11294
01-01 00:00:13.709    1575    1575 I boot_progress_pms_data_scan_start: 12081
01-01 00:00:13.735    1575    1575 I boot_progress_pms_scan_end: 12106
01-01 00:00:14.296    1575    1575 I boot_progress_pms_ready: 12667
01-01 00:00:15.237    1575    1575 I boot_progress_ams_ready: 13608
01-01 00:00:16.149    1575    1645 I boot_progress_enable_screen: 14520
```

According to ***Figure 1***, the total boot time of SC60 is calculated according to Bootloader time + Time from boot_progress_enable_screen, so the result is 2.78s + 14.520s = 17.3s. Therefore, it is concluded that the boot time of SC60 can be optimized as soon as about 17s. If the boot time of your own device is much longer than 17s, there must be some errors in your own software or hardware modification, which are made according to your own requirements, please check it!