Android RIL Integration Guide

AirPrime Modules

SIERRA WIRELESS®

4117325
1.4
Jun 20, 2017

Important Notice

Due to the nature of wireless communications, transmission and reception of data can never be guaranteed. Data may be delayed, corrupted (i.e., have errors) or be totally lost. Although significant delays or losses of data are rare when wireless devices such as the Sierra Wireless modem are used in a normal manner with a well-constructed network, the Sierra Wireless modem should not be used in situations where failure to transmit or receive data could result in damage of any kind to the user or any other party, including but not limited to personal injury, death, or loss of property. Sierra Wireless accepts no responsibility for damages of any kind resulting from delays or errors in data transmitted or received using the Sierra Wireless modem, or for failure of the Sierra Wireless modem to transmit or receive such data.

Safety and Hazards

Do not operate the Sierra Wireless modem in areas where cellular modems are not advised without proper device certifications. These areas include environments where cellular radio can interfere such as explosive atmospheres, medical equipment, or any other equipment which may be susceptible to any form of radio interference. The Sierra Wireless modem can transmit signals that could interfere with this equipment. Do not operate the Sierra Wireless modem in any aircraft, whether the aircraft is on the ground or in flight. In aircraft, the Sierra Wireless modem **MUST BE POWERED OFF**. When operating, the Sierra Wireless modem can transmit signals that could interfere with various onboard systems.

Note: *Some airlines may permit the use of cellular phones while the aircraft is on the ground and the door is open. Sierra Wireless modems may be used at this time.*

The driver or operator of any vehicle should not operate the Sierra Wireless modem while in control of a vehicle. Doing so will detract from the driver or operator's control and operation of that vehicle. In some states and provinces, operating such communications devices while in control of a vehicle is an offence.

Limitations of Liability

This manual is provided "as is". Sierra Wireless makes no warranties of any kind, either expressed or implied, including any implied warranties of merchantability, fitness for a particular purpose, or noninfringement. The recipient of the manual shall endorse all risks arising from its use.

The information in this manual is subject to change without notice and does not represent a commitment on the part of Sierra Wireless. SIERRA WIRELESS AND ITS AFFILIATES SPECIFICALLY DISCLAIM LIABILITY FOR ANY AND ALL DIRECT, INDIRECT, SPECIAL, GENERAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES INCLUDING, BUT NOT LIMITED TO, LOSS OF PROFITS OR REVENUE OR ANTICIPATED PROFITS OR REVENUE ARISING OUT OF THE USE OR INABILITY TO USE ANY SIERRA WIRELESS PRODUCT, EVEN IF SIERRA WIRELESS AND/OR ITS AFFILIATES HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR THEY ARE FORESEEABLE OR FOR CLAIMS BY ANY THIRD PARTY.

Notwithstanding the foregoing, in no event shall Sierra Wireless and/or its affiliates aggregate liability arising under or in connection with the Sierra Wireless product, regardless of the number of events, occurrences, or claims giving rise to liability, be in excess of the price paid by the purchaser for the Sierra Wireless product.

Customer understands that Sierra Wireless is not providing cellular or GPS (including A-GPS) services. These services are provided by a third party and should be purchased directly by the Customer.

SPECIFIC DISCLAIMERS OF LIABILITY: CUSTOMER RECOGNIZES AND ACKNOWLEDGES SIERRA WIRELESS IS NOT RESPONSIBLE FOR AND SHALL NOT BE HELD LIABLE FOR ANY DEFECT OR DEFICIENCY OF ANY KIND OF CELLULAR OR GPS (INCLUDING A-GPS) SERVICES.

Patents

This product may contain technology developed by or for Sierra Wireless Inc.

This product includes technology licensed from QUALCOMM®.

This product is manufactured or sold by Sierra Wireless Inc. or its affiliates under one or more patents licensed from InterDigital Group and MMP Portfolio Licensing.

Copyright

Trademarks

Sierra Wireless®, AirPrime®, AirLink®, AirVantage®, WISMO®, ALEOS® and the Sierra Wireless and Open AT logos are registered trademarks of Sierra Wireless, Inc. or one of its subsidiaries.

Watcher® is a registered trademark of NETGEAR, Inc., used under license.

Windows® and Windows Vista® are registered trademarks of Microsoft Corporation.

Macintosh® and Mac OS X® are registered trademarks of Apple Inc., registered in the U.S. and other countries.

QUALCOMM® is a registered trademark of QUALCOMM Incorporated. Used under license.

Other trademarks are the property of their respective owners.

Contact Information

Sales Desk:	Phone:	1-604-232-1488
	Hours:	8:00 AM to 5:00 PM Pacific Time
	Contact:	http://www.sierrawireless.com/sales
Post:	Sierra Wireless 13811 Wireless Way Richmond, BC Canada V6V 3A4	
Technical Support:	support@sierrawireless.com	
RMA Support:	repairs@sierrawireless.com	
Fax:	1-604-231-1109	
Web:	http://www.sierrawireless.com/	

Consult our website for up-to-date product descriptions, documentation, application notes, firmware upgrades, troubleshooting tips, and press releases: www.sierrawireless.com

Document History

Version	Date	Updates
1.0	April 30, 2015	Creation
1.1	May 27, 2016	Update for Hikey platform
1.2	Dec 01, 2016	Update for HL and EM74xx modem firmware download
1.3	Mar 31, 2017	Update for Nougat x86 platform
1.4	Jun 20, 2017	Update for new integration instructions

Contents

1. GENERAL 7
   1.1. Purpose 7
   1.2. Scope 7
   1.3. Terminology and Acronyms 7
2. SUPPORTED ANDROID VERSIONS 8
3. INTEGRATION 9
   3.1. Non-RIL Files 9
      3.1.1. Device Drivers 9
         3.1.1.1. QMI modules 9
         3.1.1.2. AT modules 10
         3.1.1.3. HL modules 10
         3.1.1.4. Android version 6.x 10
      3.1.2. Framework 11
   3.2. Android RIL Files 13
      3.2.1. RIL Source Files 13
      3.2.2. RIL Binary Files 13
   3.3. Network Interface Name 15
   3.4. SELinux 15
   3.5. Switching Between HL / AT / QMI RIL 15
      3.5.1. Android Version 5.0 or Newer 15
4. RIL SOURCE FILE DOCUMENTATION 16
5. USAGE INSTRUCTIONS 17
   5.1. Custom RIL Commands 17
      5.1.1. CDMA Activation 17
      5.1.2. User Initiated PRL Update 17
      5.1.3. Factory Reset 18
      5.1.4. Get Current SAR Back-off State 18
      5.1.5. Set Current SAR Back-off State 18
   5.2. CDMALTEPhone Object 18
   5.3. Firmware Image Management 19
      5.3.1. Multi-Carrier Image Management on MC8355 and SL9090 19
         5.3.1.1. Overview 19
         5.3.1.2. Application Layer Operation 19
         5.3.1.3. Image Directory 20
   5.4. Firmware Image Download on MC73xx 21
   5.5. Firmware Image Download on EM74xx 21
   5.6. Firmware Image Download on SL809x 22
   5.7. Firmware Image Download on HL Modem 22

5.8. Firmware Trace for HL Modem 23
5.9. Remote DM Logging....23
5.10. SAR Back-off Tool 24
5.11. Java App for Capturing Radio and Main Logs 25
5.12. RIL Properties....25
5.12.1. persist.sierra.sim_ready_delay 25
5.12.2. persist.sierra.block_init_reg 26
5.12.3. ro.sierra.voice....26
5.12.4. persist.sierra.sim_poll_delay....26
5.12.5. persist.sierra.gpsxtra....26
5.12.6. persist.sierra.debug.sdk....26
5.12.7. sys.ril.type 27
5.13. Detecting LTE Modules 27
6. TROUBLESHOOTING 28
6.1. Radio Log 28
6.2. Kernel Log 28
6.3. Verifying Proper Driver Operation 28

1. General

1.1. Purpose

This document contains references and development notes to aid in the software integration of different AirPrime modules on Android platforms. This document is intended for OEMs and third party application developers.

1.2. Scope

This document describes how to integrate the Radio Interface Layer (RIL) into Android systems. Hardware integration and other operation systems are not covered in this document.

1.3. Terminology and Acronyms

Term/Acronym	Definition
API	Application Programming Interface
OEM	Original Equipment Manufacturer
QMI	Qualcomm Modem Interface; Qualcomm MSM Interface
SMS	Short Message Service
RIL	Radio Interface Layer
SDK	Software Development Kit for Linux

2. Supported Android Versions

The following Android versions are supported:

- x86
  - 4.4 (KitKat)
  - 5.0 (Lollipop)
  - 5.1 (Lollipop)
  - 6.0 (Marshmallow)
  - 7.1 (Nougat)
- ARM
  - 2.2 (Froyo)
  - 2.3 (Gingerbread)
  - 3.1 (Honeycomb)
  - 4.0 (Ice cream sandwich)
  - 4.2 (Jelly bean)
  - 4.4 (KitKat)
  - 5.0 (Lollipop)
  - 5.1 (Lollipop)
  - 6.0 (Marshmallow)
  - 7.0 (Nougat)
  - 7.1 (Nougat)

3. Integration

3.1. Non-RIL Files

These files should be installed in the source tree as in the following sub-sections and re-compiled in the Android BSP. For more information about these files, please see the “Readme.txt” of the Android RIL release notes.

3.1.1. Device Drivers

The path of all files in this sub-section are taken from the reference platform build.

3.1.1.1. QMI modules

The following are Gobi network driver files for QMI type devices (e.g. EM/MC73xx, EM/MC74xx) Please refer to Release Note for other supported QMI modules.

- **kernel/hikey-linaro/drivers/net/usb/Kconfig**
- **kernel/hikey-linaro/drivers/net/usb/Makefile**
- **kernel/hikey-linaro/drivers/net/usb/Structs.h**
- **kernel/hikey-linaro/drivers/net/usb/QMI.h**
- **kernel/hikey-linaro/drivers/net/usb/QMIDevice.h**
- **kernel/hikey-linaro/drivers/net/usb/QMIDevice.c**
- **kernel/hikey-linaro/drivers/net/usb/QMI.c**
- **kernel/hikey-linaro/drivers/net/usb/GobiUSBNet.c**
- **kernel/hikey-linaro/drivers/net/usb/gobi_usbnet.h**

These should be merged with existing files and the new files copied to the appropriate locations. Ensure that these driver files are in the kernel .config file.

kernel/hikey-linaro/drivers/usb/serial/sierra.c is the Sierra Wireless serial driver file. It should be copied to the appropriate location, overwriting any existing file. Ensure that this driver is ENABLED while the qcserial driver is DISABLED in the kernel .config file.

3.1.1.2. AT modules

The following are network and serial driver files for AT type devices (e.g. SL8xxx)
Please refer to Release Note for other supported AT modules.

kernel/hikey-linaro/drivers/net/usb/sierra_net.c is the Sierra Wireless network driver file. It should be copied to the appropriate location, overwriting any existing file. Ensure that this driver is enabled in the kernel .config file.

kernel/hikey-linaro/drivers/usb/serial/sierra.c is the Sierra Wireless serial driver file. It should be copied to the appropriate location, overwriting any existing file. Ensure that this driver is ENABLED while the qcserial driver is DISABLED in the kernel .config file.

3.1.1.3. HL modules

The following are Ethernet driver files for HL type devices (e.g. HL7xxx, HL8xxx)
Please refer to Release Note for other supported HL modules.

kernel/hikey-linaro/drivers/net/usb/cdc_ncm.c is the Network Control Model Driver for HL7xxx modules. Ensure that this driver is enabled in the kernel .config file.

kernel/hikey-linaro/drivers/net/usb/cdc_ether.c is the CDC Ethernet driver file for HL8xxx. It should be copied to the appropriate location, overwriting any existing file. Ensure that this driver is enabled in the kernel .config file.

kernel/hikey-linaro/drivers/usb/class/cdc-acm.c is the Abstract Control Model driver for HL modules. Ensure that this driver is enabled in the kernel .config file.

3.1.1.4. Android version 6.x

kernel/hikey-linaro/include/uapi/linux/netfilter/xt_socket.h

kernel/hikey-linaro/net/netfilter/xt_socket.c

These should be merged with existing files to add support xt_qtaguid in the kernel for Android 6 platform.

3.1.2. Framework

If desired, these files can be merged with existing files. All Sierra changes are surrounded by comments containing SWISTART and SWISTOP.

Changes to logd_write.c are optional; they will cause the dhcpcd log messages to be put into the radio log instead of the main log. Changes in the following are conditionally compiled with the SIERRA flag. This flag is normally not defined, so either define the flag in an appropriate place, or remove the conditional compilation

- **system/core/init/property_service.c**
- **system/core/liblog/logd_write.c**

device/asus/tf300t/overlay/frameworks/base/core/res/res/values/config.xml is taken from the Sierra Wireless reference platform (ASUS). On Hikey platform, these changes will be found at **device/linaro/hikey/overlay/frameworks/base/core/res/res/values/config.xml** . It may be necessary to merge Sierra specific changes into the corresponding config.xml file on your platform.

external/stlport source is used to build libstlport.so**.** It is necessary for Android 5.x and newer versions to execute /system/bin/Downloadtool for HL modem firmware download/upgrade.

external/dhcpcd source is used to build dhcpcd**.** It is necessary for Android 7.x and newer versions to execute /system/bin/dhcpcd for start DHCP session.

The following files modify the definitions associated with LTE and eHRPD to be consistent with public Android documentation. It is required for Android 6.x and older versions with new RIL (version Android RIIL_7.0.1.0 and newer)

- **frameworks/base/telephony/java/android/telephony/ServiceState.java**
- **frameworks/opt/telephony/src/java/com/android/internal/telephony/cdma/CdmaServiceStateTracker.java (For Android 4.4 and later)**
- **framework/base/telephony/java/com/android/internal/telephony/cdma/CdmaServiceStateTracker.java (For Android versions lower than 4.4)**

frameworks/opt/telephony/src/java/com/android/internal/telephony/ImsSMSDispatcher.java fixes 3GPP2 format to be used to send SMS. If IMS is not supported in Dual Mode LTE, the framework will send an SMS via 3GPP format. This is only applicable for Android version 4.4 or newer.

frameworks/base/telephony/java/com/android/internal/telephony/DctConstants.java is necessary to get the CDMAPhone object to work correctly for MC8355 modules. This is applicable for Android 4.4 or newer.

frameworks/opt/telephony/src/java/com/android/internal/telephony/cdma/CdmaLteServiceStateTracker.java is necessary to get the CDMALTEPhone object to work well with MC7750 modules. For Android versions older than 4.4 this file is located at **frameworks/base/telephony/java/com/android/internal/telephony/cdma/CdmaLteServiceStateTracker.java**

frameworks/base/telephony/java/com/android/internal/telephony/RILConstants.java defines constants used for custom commands.

frameworks/base/core/res/res/xml/eri.xml corrects the roaming icon; no issue.

frameworks/base/telephony/java/android/telephony/SignalStrength.java fixes the signal strength not shown correctly issue.

frameworks/opt/telephony/src/java/com/android/internal/telephony/PhoneProxy.java fixes the data call not happening when coming out of Airplane mode problem. For Android versions older than 4.4 this file is located at
frameworks/base/telephony/java/com/android/internal/telephony/PhoneProxy.java

frameworks/opt/telephony/src/java/com/android/internal/telephony/cdma/CDMALTEPhone.java is the fix for the MDN is not shown for CdmaLtePhone under LTE problem. For Android versions older than 4.4 this file is located at
frameworks/base/telephony/java/com/android/internal/telephony/CDMALTEPhone.java

packages/apps/Settings/src/com/android/settings/deviceinfo/Status.java is the fix for the incorrect MDN shown for technology changed in Dual Mode problem.

frameworks/base/core/java/com/android/server/BootReceiver.java and frameworks/base/core/res/AndroidManifest.xml is the fix for running com.sierra.logs app for Android 5.x and newer versions.

The following changes are necessary to add support for OMADM GUI (For Android 4.4 and newer):

- **frameworks/opt/telephony/src/java/com/android/internal/telephony/BaseCommands.java**
- **frameworks/opt/telephony/src/java/com/android/internal/telephony/RIL.java**
- **frameworks/opt/telephony/src/java/com/android/internal/telephony/cdma/CDMAPhone.java**
- **frameworks/opt/telephony/src/java/com/android/internal/telephony/cdma/CdmaServiceStateTracker.java**
- **frameworks/opt/telephony/src/java/com/android/internal/telephony/dataconnection/DcTracker.java**
- **packages/apps/Dialer/src/com/android/dialer/SpecialCharSequenceMgr.java**

3.2. Android RIL Files

3.2.1. RIL Source Files

- **hardware/sierra/***

  This should be placed in the Android source tree at the specified location. Note that this location is hard-coded in the Android.mk files.

- **build/core/user_tags.mk**

  This should be merged with the existing file. All Sierra changes are surrounded by comments containing SWISTART and SWISTOP.

3.2.2. RIL Binary Files

These binary/target files should be used on the target device file system as described below.

The following files should be copied to the target device file system for 64 bit platform:

- **AndroidFS/system/bin/swisdk**
- **AndroidFS/system/bin/swifwdnld**
- **AndroidFS/system/bin/swihltrace**
- **AndroidFS/system/lib64/libsierra-ril.so**
- **AndroidFS/system/lib64/libsierraat-ril.so**
- **AndroidFS/system/lib64/libsierrahl-ril.so**
- **AndroidFS/system/lib64/libswisdkapi.so**
- **AndroidFS/system/bin/slqssdk**
- **AndroidFS/system/bin/SierraDMLog**
- **AndroidFS/system/bin/SierraFwDl7xxx**
- **AndroidFS/system/bin/SierraImgMgr**
- **AndroidFS/system/bin/SierraSARTool**
- **AndroidFS/system/bin/DownloadTool**
- **AndroidFS/system/bin/swihltrace**
- **AndroidFS/system/lib64/libswims.so**
- **AndroidFS/system/lib64/libswiqmiapi.so**
- **AndroidFS/system/lib64/libDownloadTool.so**
- **AndroidFS/system/lib64/libstlport.so**
- **AndroidFS/system/lib64/libDownloadTool.so**
- **AndroidFS/system/lib64/libstlport.so**
- **AndroidFS/system/app/com.sierra.logs/com.sierra.logs.apk**

For 32 bit platform, following files should be copied:

- **AndroidFS/system/bin/swisdk**
- **AndroidFS/system/bin/swifwdnld**
- **AndroidFS/system/bin/swihltrace**
- **AndroidFS/system/lib/libsierra-ril.so**
- **AndroidFS/system/lib/libsierraat-ril.so**
- **AndroidFS/system/lib/libsierrahl-ril.so**

- **AndroidFS/system/lib/libswisdkapi.so**
- **AndroidFS/system/bin/slqssdk**
- **AndroidFS/system/bin/SierraDMLog**
- **AndroidFS/system/bin/SierraFwDl7xxx**
- **AndroidFS/system/bin/SierraImgMgr**
- **AndroidFS/system/bin/SierraSARTool**
- **AndroidFS/system/bin/DownloadTool**
- **AndroidFS/system/bin/swihltrace**
- **AndroidFS/system/lib/libswims.so**
- **AndroidFS/system/lib/libswiqmiapi.so**
- **AndroidFS/system/lib/libDownloadTool.so**
- **AndroidFS/system/lib/libstlport.so**
- **AndroidFS/system/app/com.sierra.logs/com.sierra.logs.apk**

For QMI RIL, **AndroidFS/system/lib64/libswigpsqmi.so** should be copied to /system/lib64/hw on the target device file system and renamed appropriately for the target device.

For AT/HL RIL, **AndroidFS/system/lib64/libswigpsat.so** should be copied to /system/lib64/hw on the target device file system and renamed appropriately for the target device.

For Android 7.0 and newer, **AndroidFS/system/bin/dhcpcd** should be copied to /system/bin on the target device file system.

The following files should be copied to the target device system.

- **AndroidFS/system/bin/init.dhcpcd**
- **AndroidFS/system/bin/init.config.ip**

If a file already exists with the same name, do not merge the files; instead rename this file to something else, and update init.rc with the previously existing file name.

The following files are taken from the Sierra Wireless reference platform.

- **AndroidFS/init.rc**
- **AndroidFS/ueventd.rc**
- **AndroidFS/system/etc/init/rild.rc**

Start RIL service when Android system is started and merge Sierra specific changes into your files. All Sierra changes are surrounded by comments containing SWISTART and SWISTOP.

Note that some lines are only applicable to a specific Sierra Wireless module, and so can be omitted. This line is: *insmod /system/lib/modules/sierra_net.ko*.

3.3. Network Interface Name

The network interface name is "wwanx" for HL, QMI or AT RIL where x is an integer, usually 0 or 1. The exact value depends on what other USB network devices are on the target device.

The files:

- **AndroidFS/system/bin/init.dhcpcd**
- **AndroidFS/system/bin/init.config.ip**

are hard-coded with usb0/wwan0. If this is not the correct network name on your target device, please change usb0/wwan0 to the correct name.

3.4. SELinux

By default, Security Enhanced Linux (SELinux) is enforced in Android 5.0. Make sure that this feature is disabled in the kernel .config file. Otherwise, settings to the Android operation system have to be customized.

The path of the following files are taken from the reference platform build.

- **device/linaro/hikey/sepolicy/device.te**
- **device/linaro/hikey/sepolicy/file.te**
- **device/linaro/hikey/sepolicy/file_contexts**
- **device/linaro/hikey/sepolicy/gpsd.te**
- **device/linaro/hikey/sepolicy/netd.te**
- **device/linaro/hikey/sepolicy/property_contexts**
- **device/linaro/hikey/sepolicy/radio.te**
- **device/linaro/hikey/sepolicy/rild.te**
- **device/linaro/hikey/sepolicy/dhcp.te**
- **device/linaro/hikey/sepolicy/sierra_dhcpcd.te**

3.5. Switching Between HL / AT / QMI RIL

3.5.1. Android Version 5.0 or Newer

HL, AT and QMI RILs are included in the release package. The default setting is QMI RIL, however, it is possible to switch between these RILs by changing the system property.

To do this, follow these steps:

1. sudo adb root
2. adb remount
3. adb shell setprop sys.ril.type [HL | AT | QMI]
4. adb reboot

4. RIL Source File Documentation

Preliminary doxygen generated documentation in HTML format is available at ***hardware/sierra/doc/html/index.html***.

5. Usage Instructions

This section provides information and instructions on using the RIL with AirPrime modules.

Note: Some sub-sections are only applicable to specific modules. In such cases, it will be mentioned in the sub-section's description.

5.1. Custom RIL Commands

Applications use **`Phone.invokeOemRilRequestStrings()`** to invoke custom RIL commands. This method is defined in **Phone.java** as:

```
void invokeOemRilRequestStrings(String[] strings, Message response);
```

Custom commands are specified in **`strings[0]`**; arguments for custom commands are specified in **`strings[1]`**, etc. Currently supported custom commands are defined in **RILConstants.java**:

```
String OEM_HOOK_STRING_RIL_VERSION = "4";
String OEM_HOOK_STRING_CDMA_ACTIVATION = "5";
String OEM_HOOK_STRING_FACTORY_RESET = "6";
String OEM_HOOK_STRING_PRL_UPDATE = "7";
String OEM_HOOK_STRING_SAR_GET = "8";
String OEM_HOOK_STRING_SAR_SET = "9";
```

5.1.1. CDMA Activation

For CDMA activation, **`strings`** should be:

```
strings[0] = RILConstants.OEM_HOOK_STRING_CDMA_ACTIVATION;
```

There are no arguments for this custom command. This command is only applicable to AirPrime MC8355 Sprint and Verizon images.

5.1.2. User Initiated PRL Update

For user initiated PRL updates, **`strings`** should be:

```
strings[0] = RILConstants.OEM_HOOK_STRING_PRL_UPDATE;
```

There are no arguments for this custom command. The module will reset after a successful PRL update. If the update fails, then either the RIL command will return an error right away, or **`Phone.CDMA_OTA_PROVISION_STATUS_OTAPA_ABORTED`** will be sent.

This command should not be used when there is an active data session. If an attempt to use this command is made when there is an active data session, the command will fail, and **`Phone.CDMA_OTA_PROVISION_STATUS_OTAPA_ABORTED`** will be sent.

This command is only applicable to AirPrime MC8355 Sprint images.

5.1.3. Factory Reset

For factory reset, **strings** should be:

```
strings[0] = RILConstants.OEM_HOOK_STRING_FACTORY_RESET;
strings[1] = <SPC>;
```

The service programming code (SPC) to use depends on the carrier. For carriers other than Sprint, it should be six zeros, i.e. “000000”; while for Sprint, it is modem specific and should be obtained directly from Sprint.

The module will reset after a successful factory reset. If the factory reset fails, then the RIL command will return an error right away.

This command is only applicable to AirPrime MC8355 modules. It is targeted for Sprint images, but this command should work for any carrier image.

5.1.4. Get Current SAR Back-off State

For getting the current SAR state, **strings** should be:

```
strings[0] = RILConstants.OEM_HOOK_STRING_SAR_GET;
```

There are no arguments for this custom command. The response will contain a value between 0 and 8 that indicates the current SAR back-off state.

This command is applicable to both AirPrime MC8355 and MC77xx modules. MC7700 modules only support 1.x FW versions.

5.1.5. Set Current SAR Back-off State

For setting the current SAR state, **strings** should be:

```
strings[0] = RILConstants.OEM_HOOK_STRING_SAR_SET;
strings[1] = <SAR state>;
```

The <SAR state> should be a number between 0 and 8, represented as a string.

This command is applicable to both AirPrime MC8355 and MC77xx modules. MC7700 modules only support 1.x FW versions.

5.2. CDMALTEPhone Object

This is a new phone object to support Sierra Wireless LTE modules on the Sprint and/or Verizon networks. In order to enable this phone object, the following must be done:

1. Add the property setting **telephony.lteOnCdmaDevice=1** to an appropriate property file, such as **system.prop**.
2. Ensure that the network mode is 7. If the network mode is not 7, this can be changed by deleting the database file **/data/data/com.android.providers.settings/databases/settings.db** This file will get regenerated when the system boots. Assuming that the property in step 1 is set, the network mode will be initialized to 7.

Initial support has been added in the RIL to auto-detect the LTE module and set the appropriate property.

5.3. Firmware Image Management

5.3.1. Multi-Carrier Image Management on MC8355 and SL9090

5.3.1.1. Overview

The AirPrime MC8355 and SL9090 support a multi-carrier image management solution. This solution allows the host client to switch between different carrier firmware images and configuration files in order to support operation on those carriers.

In order to support this within an Android OS, a custom application must be written. This application must be able to do the following:

- Read a list of supported carrier images
- Display the list to the end user
- Allow the end user to select the requested carrier from the list
- Trigger switching to a new carrier image

The MC8355 and SL9090 support different carrier images such as:

- Verizon
- Sprint
- AT&T
- T-Mobile
- Rogers
- Generic HSPA

5.3.1.2. Application Layer Operation

The application layer should call the **SierraImgMgr** executable described below to run image switching.

Print List of Supported Carrier Images

```
SierraImgMgr --list <image directory>
```

A list of all available carrier images in <image directory> will be printed. The list will include the following comma-separated fields: Id, Carrier, Technology, Region, Version and Status.

For example, if there are 3 images with the Sprint image being currently selected, the output might be:

```
Id,Carrier,Technology,Region,Version,Status
2,AT&T,UMTS,NA,0E123456,Inactive
3,Sprint,CDMA,NA,0E223456,Active
6,Generic HSPA,UMTS,Global,0E323456,Inactive
```

Id values will always be the same for any given carrier.

Trigger S**witch**ing **to** a N**ew** C**arrier** I**mage**

```
SierraImgMgr --select <Id> <image directory>
```

The Id value must be one of the values returned by the --list option, and is used to select the active image. The --select option may cause the module to reset. The command will not return until the module has finished resetting and the new carrier image is successfully running.

Update I**mages on the** Module

```
SierraImgMgr --update <image directory>
```

This will update all images that are currently on the module with newer image files, if available. If there is no newer image file available for a particular carrier, then nothing will be done with that carrier image on the module.

The --update option does not change the active carrier selection.

Print L**ist of** C**arrier** I**mages on the** Module

```
SierraImgMgr --listdevice
```

A list of available carrier images on the module will be printed. The list will follow the same format used by the --list option described in section 0.

Trigger S**witch**ing **to** a N**ew** C**arrier** I**mage on the** Module

```
SierraImgMgr --selectdevice <Id> [<config file path>]
```

The Id value must be one of the values returned by the --listdevice option, and is used to select the active image.

The --selectdevice option may cause the module to reset. The command will not return until the module has finished resetting and the new carrier image is successfully running.

Note that:

- The <image directory> argument must be the absolute path to the images directory. It should not contain any spaces.
- The --select, --update, --listdevice and --selectdevice options conflict with the RIL daemon. To ensure correction operation, always stop the RIL daemon before using these options, and restart the daemon afterwards. For example (from shell command line):

```
stop ril-daemon-qmi
SierraImgMgr --list /data/modem/images
SierraImgMgr --select 2 /data/modem/image
start ril-daemon-qmi
```

The example above can be adapted easily to be called from a Java application.

5.3.1.3. Image Directory

The image distribution package will contain the images directory. This directory will itself contain a number of sub-directories, usually named with small numbers. It is essential that these sub-directories are never renamed, and that files are not moved from one sub-directory to another sub-directory.

This images directory can be put in any customer specific location. The absolute path to the image library must always be given to the **SierraImgMgr** executable. This path should include the images directory, as given in the example above.

If a new image distribution package is released, use the following steps to update the images used by the module:

- To keep any images that have not been updated, merge the new images directory with the existing directory, replacing any existing files.
- To only use the newly released images, delete the existing images directory first and then put the new directory in the same place.
- Execute **SierraImgMgr** with the --update option as described above.

5.4. Firmware Image Download on MC73xx

The **SierraFwDl7xxx** executable is provided to download firmware to AirPrime MC73xx modules. It is a command line tool and is available in /system/bin. The -h option can be used to get usage information. All options of this tools are not applicable for MC78xx modules Command options and usage of applicable options is provided below:

- -h --help

  Display this information and exit.
- -? --help

  Display this information and exit.
- -v --verbose

  Display extra information while running. Note that “verbose” has to be specified first.
- -V --version

  Display version of the application
- -g --get

  Display information for the executing device image.
- -d --download <CWEimagepath>

  Download CWE image at <CWEimagepath>. Note that this must be an absolute path.
- -i --info <CWEimagepath>

  Display information for a particular CWE image file located at <CWEimagepath>. Note that this must be an absolute path.
- -f –force

  Force firmware download regardless of host firmware version.

5.5. Firmware Image Download on EM74xx

The SierraFwDl7xxx is used to download/upgrade firmware for EM74xx modules also. This tool is available in /system/bin. All options of this tools are not applicable for EM74xx modules. Command options and usage of options which are applicable for EM74xx is provided below:

- -h --help
  Display command line options and exit.
- -? --help

Display this information and exit.

- -v --verbose

  Display extra information while running. Note that “verbose” has to be specified first.

- -V --version

  Display version of the application.

- -l --listimages

  Display images information for 9x30 modules (located on Device as well as Host). Use –p option as well to get the information of images located on Host

- -p --pathhost <imagepath>

  Path of 9x30 image located on host. Provide path as -p /data/EM7455 if firmware is located at /data/EM7455/1. It is used with -l and –s.

- -s --selectimage <imageindex>

  Select image for 9x30 with <image index>. Image index is displayed by -l

5.6. Firmware Image Download on SL809x

Firmware updates are released as CWE files. The **swifwdnld** executable is provided to update the module’s firmware from the adb shell.

To use, type:

```
cd /system/data
./swifwdnld -p ./swisdk -f /data/<file.cwe> -x
```

Firmware update may take several minutes. The module will reboot at the end of the update.

5.7. Firmware Image Download on HL Modem

The DownloadTool is used to download the firmware image for HL series modems. This tool is copied to /system/bin/. It also requires shared libraries as libstlport.so and libDownloadtoool.so in /system/lib. It is currently built only as a 32 bit executable and is used to run on 64 bit platform also. DownloadTool detects the attached HL modem and after that a manual power reset is needed to take HL modem in “download mode”. It may take several minutes to complete the download. The tools consist of lot of options but for HL modem image download over USB, it can be executed with the minimal options as :

DownloadTool /data/HL8/SAM_6260.fls -v4 -d USB

(Manual power reset is needed to trigger firmware download start)

- /data/HL8/SAM_6260.fls is the path of HL modem image along with its name
- -v is used for verbose level. It may vary to [0-4], where 0 is for minimum and 4 is for maximum logs.
- -d is used for debugging and debugging over “USB” is used always

Another tool /system/bin/HLDownload is also used to download HL Modem firmware. Manual power reset is not needed with this tool during download, it will automatically reset the modem whenever

required. This tool, in turn, invokes /system/bin/DownloadTool to download the firmware image. This tool is invoked with these options:

- -h
  Display help information and exit

- -i <image path>
  HL firmware image full path with image name (mandatory parameter)

- -p <tool path>
  Path of the DownoadTool. If not specifies /system/bin/DownloadTool is taken into consideration

- -v <level>
  For verbose level, value from [0-4]

5.8. Firmware Trace for HL Modem

Firmware logs of HL modem are captured with swihltrace command line tool. This tool opens the tty port of the HL modem to capture the log. Command options and usage of the options is provided below:

- -d <port name>
  /dev/ttyACMx port for HL firmware logging. This is an optional parameter. By default, ttyACM1 is opened.

- -o <log file>
  Name of the file with complete path. This is also an optional parameter. By default, log file will be stored as /data/tracelog-xxx.bin. Make sure that swihltrace tool has write permission at the path.

5.9. Remote DM Logging

Remote DM logging can be used to connect the DM port on the module under Android to a Windows PC application that uses the DM port, such as QXDM. There are several options to enable remote DM logging (also called DM port forwarding). The **SierraDMLog** executable is provided to help with this process. The -h option can be used to get usage information. Command options and usage are provided below.

- Local Logging: **SierraDMLog -l [-d device] [-f filter] [-o logfile]**

  Logs DM packets to "logfile" if specified, or to a script generated log file stored in the /data directory with name swidmlog if not specified.

- Remote Logging: **SierraDMLog [-d device] [-f filter] [-p netport] [-r rhost]**

  Establishes a TCP connection with a remote machine "rhost" using port number "netport", or a default port number of 2500 if not specified. DM packets are exchanged over the TCP connection.

  Without the -r option, this acts as a server application waiting for an incoming connection request. Otherwise, it acts as a client and attempts to establish a connection with "rhost".

Available options are as follows:

- -l

  Local logging if specified, remote logging otherwise

- -d device

  (Optional) /dev/ttyUSBx port for DM logging
- -f filter

  (Optional) DM filter to send to the device prior to logging
- -p netport

  (Optional) Remote logging TCP port (defaults to 2500)
- -r rhost

  (Optional) remote host to connect to
- -o logfile

  (Optional) fully qualified DM log (output) file name

If the DM port is not specified using the -d option, it will automatically be detected by the command. In this case, the port name will be printed out since it is needed for some of the other steps as described in the examples below.

All of the examples below assume that the DM port on the Android device is /dev/ttyUSB0.

Example 1: Using a USB cable with a Windows host computer:

1. Execute "**adb shell SierraDMLog -d /dev/ttyUSB0**"
2. Execute "**adb forward tcp:2500 dev:dev/ttyUSB0**"
3. On the Windows host using QPST, configure "IP Server" to use IP 127.0.0.1 and port number 2500 to connect to the device.

Example 2: Alternative for a USB connection with Windows host computer:

1. Enable USB tethering in phone/tablet *Settings* > *Wireless & Networks* > *Tethering & Portable hotspot* > *USB tethering*.
2. Use "**adb shell netcfg**" to get the IP address of USB0.
3. Use a USB cable to connect to the host then use "ipconfig" to find the IP address of the host device.
4. Use "**adb shell SierraDMLog -d /dev/ttyUSB0 -r xx.xx.xx.xx**"
5. On the host computer, launch QPST to add port.
6. On the host computer, run QXDM.

Example 2: Using a Wi-Fi hotspot provided by the phone or tablet:

1. Enable Wi-Fi in phone/tablet *Settings* > *Wireless & Networks* > *Wifi*.
2. Enable hotspot functionality in *Settings* > *Wireless & Networks* > *Tethering & Portable hotspot* > *Portable Wifi hotspot*.
3. On the host computer, use Wi-Fi to connect to access point "AndroidAP".
4. Use "**adb shell SierraDMLog -d /dev/ttyUSB0 –r xx.xx.xx.xx**".
5. In the host computer, launch QPST to add port.
6. In the host computer, run QXDM.

5.10. SAR Back-off Tool

A command line tool, SierraSARTool, runs in the adb shell or an Android console is used to get and set the current SAR state value.

This tool is targeted at MC8355 modules, although the set command also works with some FW versions of the MC7700. For MC77xx modules, AT commands are available which provide similar functionality.

1. Query the current SAR value:

   ```
   SierraSARTool      /* without arguments */
   ```

2. Set the current value to new SAR state:

   ```
   SierraSARTool <new SAR State>
   ```

Note that this tool cannot run at the same time as the RIL daemon. Always stop the RIL daemon before using this command, and restart the daemon afterwards. For example (from adb shell command line):

```
stop ril-daemon-qmi
SierraSARTool            /* to query current value */
SierraSARTool 4          /* to set the new value to 4 */
start ril-daemon-qmi
```

5.11. Java App for Capturing Radio and Main Logs

Warning: *This app is only intended for use with Sierra Wireless modules, and should only be used for diagnostic and testing purposes. This App should not be included in any shipping product due to potential privacy and security issues.*

The app should be installed as any other third party non-Android Marketplace application.

To use:

- The radio log, main log or both can be collected by selecting the appropriate option.
- Output file name prefix can be specified in the "Log label" field.
- A timestamp is appended to each output file name, giving the time that the log collection started.
- For Android 5.x and newer versions, due to stringent access permissions **frameworks/base/core/java/com/android/server/BootReceiver.java and frameworks/base/core/res/AndroidManifest.xml** is updated to execute logcat in root mode. Otherwise radio logs will not be captured
- For Android versions older than 5.x, output files are located in /data/data/com.sierra.logs/files or on the root of the SD card, if "Save to SD card" is selected.
- “Save to SD card” is disabled for Android 5.x and later versions and log files are stored at /data/backup

5.12. RIL Properties

RIL operation can be modified by setting various Android properties.

5.12.1. persist.sierra.sim_ready_delay

During RIL initialization, after the RIL has determined that the SIM is ready, it will send a RADIO_STATE_CHANGED notification to the telephony framework indicating that the SIM is ready. The framework usually responds by issuing a number of SIM related commands.

This property can be used to delay sending the notification that the SIM is ready until after the modem has successfully registered. The value of the property is used to specify a timeout value in seconds. If the module is not registered within the timeout period, then the SIM_READY notification is sent regardless of the current registration state. This value should typically be at least 10-15 seconds, and normally not more than 60 seconds.

This setting can be useful for certain modules such as the MC7750 with 1.x FW and the MC8355 where SIM related commands sent by the framework during initialization may interface with the registration process. If this happens, then the SIM related command may hang the RIL. The RIL will then have to recover and restart the initialization process.

5.12.2. persist.sierra.block_init_reg

The telephony framework will send either an automatic or a manual registration command depending on what was previously selected during system start-up. This is separate from the scanning and registration that the module will do on its own when it is powered up. This initial registration command can sometimes cause problems for test equipment that is not expecting it. However, no problem has ever been observed on a live network with the initial registration command.

If this property is defined and has a value greater than 0, then any initial registration commands will be ignored by the RIL. The RIL will return success, but will not do anything further. Once an explicit network scan has been performed using RIL_REQUEST_QUERY_AVAILABLE_NETWORKS, the registration commands will work as normal.

Normally, the user should not notice any difference in behaviour if this property is defined because the module will normally retain the last registration settings even after a power cycle.

5.12.3. ro.sierra.voice

This property must be set to 1 in order to enable the voice related RIL commands. If this property is not set, or is set to some other value, voice related RIL commands will return an error.

5.12.4. persist.sierra.sim_poll_delay

In some cases, the module will return SIM_FAILURE during SIM busy. Adding this property enables extra SIM polling when the module returns SIM_FAILURE.

Android RIL will perform 10 times of retry if this property is defined. If this property is set to 6, the RIL will add an extra 1 minute of SIM status polling.

A typical use case is the MC7750 (FW3.5.10.6Z) in CDMA mode.

5.12.5. persist.sierra.gpsxtra

Set persist.sierra.gpsxtra to 1 to enable the GPS XTRA feature for the modules that support GPS XTRA, such as the AirPrime MC73xx and SL9090.

5.12.6. persist.sierra.debug.sdk

QMI SDK creates lot of traffic in radio logs. By default QMI SDK logs are disabled. To enable these logs, this property is set to 1.

5.12.7. sys.ril.type

This property is used to switch between HL, AT and QMI RILs.

5.13. Detecting LTE Modules

Initial support has been added in the RIL to auto-detect LTE modules and to set properties used by the telephony framework, such as the following:

- UMTS LTE module detected
  - telephony.lteOnGsmDevice=1
  - telephony.lteOnCdmaDevice=0
- CDMA LTE module detected
  - telephony.lteOnGsmDevice=0
  - telephony.lteOnCdmaDevice=1

If neither of these modules is detected, then both properties are set to 0.

Auto-detection is performed as early in the RIL startup as possible in order to try to ensure that these properties are set before the corresponding phone object is created.

Note:	*Properties will not be modified by the RIL as described above if either of the properties is already set when the RIL starts up. Thus, the behaviour above can be disabled by explicitly setting either of the properties before the RIL starts up.*

6. Troubleshooting

6.1. Radio Log

The Android logging system provides a mechanism for collecting and viewing system debug output. You can run logcat through ADB to read the messages in real time.

To do this, enter:

1. sudo adb root
2. adb logcat –v time –b radio

6.2. Kernel Log

The Gobi network and serial driver log are routed as event logs to dmesg. It is possible to capture the ongoing denial logs by running ***cat/proc/kmsg***.

Follow these steps:

1. sudo adb root
2. adb shell
3. cat /proc/kmsg.

6.3. Verifying Proper Driver Operation

To verify proper driver operation, follow these steps:

1. Plug in the Sierra Wireless device.
2. Check the existence of wwan0 network interface (ie. ifconfig –a).
3. Check **/dev/** for the existence of the following devices. Please note that the second QMI interface is only available when the device is in multi-pdn mode (**/dev/qcqmiy**).
   - **/dev/ttyUSBx** where x is an integer range from 0 to 4
   - **/dev/qcqmix** where x is an integer range from 0 to 1
   - **/dev/ttyACMx** where x is an integer range from 0 to 5 (for HL7 and HL8 series modules)