



**AirPrime**

# **Android RIL Integration Guide**



**SIERRA**  
WIRELESS®

4117325  
2.0  
October 05, 2020

## 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.

## 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 MMP Portfolio Licensing.

## Copyright

© 2020 Sierra Wireless. All rights reserved.

## 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 information and technical support, including warranty and returns	Web: <a href="http://sierrawireless.com/company/contact-us/">sierrawireless.com/company/contact-us/</a> Global toll-free number: 1-877-687-7795 6:00 am to 5:00 pm PST
Corporate and product information	Web: <a href="http://sierrawireless.com">sierrawireless.com</a>

# 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
1.5	May 4, 2018	Update for Oreo platform
1.6	Jun 15, 2018	Update for Sabre platform
1.7	Sep 28, 2018	Update for new integration instructions
1.8	Apr 24, 2019	Update for Pie platform
1.9	Nov 27, 2019	Update for Q platform
1.10	Apr 24, 2020	Update for new integration instructions
1.11	Jun 26, 2020	Update for new HAL radio@1.4 and gnss@2.0
1.12	Jul 31,2020	Update for add integration instructions.
2.0	October 5, 2020	Added firmware support for RC76 series.



# Contents

<b>1. GENERAL .....</b>	<b>7</b>
1.1. Purpose .....	7
1.2. Scope.....	7
1.3. Terminology and Acronyms.....	7
<b>2. SUPPORTED ANDROID VERSIONS .....</b>	<b>8</b>
<b>3. INTEGRATION .....</b>	<b>9</b>
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.....	9
3.1.1.3. HL modules.....	10
3.1.1.4. Android version 6.x .....	10
3.1.2. Framework.....	10
3.2. Android RIL Files .....	12
3.2.1. RIL Source Files .....	12
3.2.2. RIL Binary Files.....	12
3.3. Network Interface Name.....	14
3.4. SELinux .....	14
3.5. HAL Definitions.....	15
3.5.1. Radio.....	15
3.5.2. GNSS.....	15
3.6. Switching Between HL / AT / QMI RIL .....	16
3.6.1. Android Version 5.x to 9.x.....	16
3.6.2. Android Version 10.x or newer .....	16
<b>4. RIL SOURCE FILE DOCUMENTATION .....</b>	<b>17</b>
<b>5. USAGE INSTRUCTIONS .....</b>	<b>18</b>
5.1.1. Custom RIL Commands .....	18
5.1.2. CDMA Activation.....	19
5.1.3. User Initiated PRL Update .....	19
5.1.4. Factory Reset.....	20
5.1.5. Get Current SAR Back-off State .....	20
5.1.6. Set Current SAR Back-off State .....	20
5.2. CDMALTEPhone Object.....	20
5.3. Firmware Image Management.....	21
5.3.1. Multi-Carrier Image Management on MC8355 and SL9090.....	21
5.3.1.1. Overview .....	21
5.3.1.2. Application Layer Operation.....	21
5.3.1.3. Image Directory.....	22
5.4. Firmware Image Download on EM/MC73xx.....	23
5.5. Firmware Image Download on EM/MC74xx, EM75xx, WP76xx and RC76xx. ....	23
5.6. Firmware Image Download on WP75xx .....	24

---

5.7.	Firmware Image Download on SL809x .....	25
5.8.	Firmware Image Download on HL Modem .....	25
5.9.	Firmware Trace for HL Modem.....	26
5.10.	Remote DM Logging.....	26
5.11.	SAR Back-off Tool .....	27
5.12.	Java App for Capturing Radio and Main Logs.....	28
5.13.	RIL Properties.....	28
5.13.1.	persist.vendor.sierra.sim_ready_delay .....	28
5.13.2.	persist.vendor.sierra.block_init_reg .....	29
5.13.3.	ro.vendor.sierra.voice .....	29
5.13.4.	persist.vendor.sierra.sim_poll_delay .....	29
5.13.5.	persist.sierra.gpsxtra .....	29
5.13.6.	persist.sierra.sdk.debug.....	29
5.13.7.	persist.vendor.sys.ril.type .....	29
5.14.	Detecting LTE Modules .....	30
<b>6.</b>	<b>TROUBLESHOOTING .....</b>	<b>31</b>
6.1.	Radio Log .....	31
6.2.	Kernel Log .....	31
6.3.	Verifying Proper Driver Operation .....	31



# 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
  - 5.0 (Lollipop)
  - 5.1 (Lollipop)
  - 6.0 (Marshmallow)
  - 7.1 (Nougat)
  - 8.1 (Oreo)
  - 9.0 (Pie)
- ARM
  - 5.0 (Lollipop)
  - 5.1 (Lollipop)
  - 6.0 (Marshmallow)
  - 7.0 (Nougat)
  - 7.1 (Nougat)
  - 8.0 (Oreo)
  - 8.1 (Oreo)
  - 9.0 (Pie)
  - 10.0 (Q)

## 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/MC7xxx, WP7xxx) 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.c
- kernel/hikey-linaro/drivers/net/usb/QMI.h
- kernel/hikey-linaro/drivers/net/usb/QMIDevice.c
- kernel/hikey-linaro/drivers/net/usb/QMIDevice.h
- kernel/hikey-linaro/drivers/net/usb/GobiUSBNet.c
- kernel/hikey-linaro/drivers/net/usb/usbnet.c

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/net/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/net/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\_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.

### 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 logger\_write.c and property\_service.cpp are optional; they will cause the dhcpd 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 (It is only for Android 5.x or before)

- system/core/init/property\_service.cpp
- system/core/liblog/logger\_write.c

system/core/init/property\_service.cpp is necessary to grant permission to radio group to access android properties (for Android 5.x or before). For Android 6.x to Android 9.x, the changes is necessary to RIL type switching.

system/core/init/devices.cpp is optional. It will cause module switch from MBIM mode to QMI mode during first enumeration. (it is only for Android 8.x or later)

packages/services/Telephony/res/values/config.xml and device/linaro/hikey/overlay/frameworks/base/core/res/res/values/config.xml are taken from the Sierra Wireless reference platform. 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 to Android 7.x versions to execute /system/bin/Downloadtool for HL modem firmware download/upgrade.

system/core/libcutils/fs\_config.c is necessary for Android 7.x to Android 9.x or device/linaro/hikey/config.fs is necessary for Android 10.x HL RIL to execute /(vendor | system)/bin/dhcpd for start DHCP session.

external/dhccpd source are used to build dhccpd and dhccpd-run-hooks. This is applicable for Android 7.x to Android 9.x and Android 10 HL RIL only. The source code could download from AOSP (ie: git clone https://android.googlesource.com/platform/external/dhccpd). It may be necessary to include below header files for Android 7.x to Android 9.x. or apply patch for Android 10.

```
#include <netinet/if_ether.h>
#include <net/ethernet.h>
#include <linux/if_arp.h>
```

and also define below lines in an appropriate makefile.

```
PRODUCT_PACKAGES += \
    dhccpd \
    dhccpd-run-hooks
```

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 RIL\_7.0.1.0 and newer)

- frameworks/base/telephony/java/android/telephony/ServiceState.java (For Android versions lower than 7.x)
- frameworks/opt/telephony/src/java/com/android/internal/telephony/cdma/CdmaServiceStateTracker.java (For Android versions lower than 7.x)
- frameworks/opt/telephony/src/java/com/android/internal/telephony/ServiceStateTracker.java (For Android 7.x or later)

frameworks/base/telephony/java/com/android/internal/telephony/DctConstants.java is necessary to get the CDMA Phone object to work correctly for MC8355 modules.

frameworks/base/telephony/java/android/telephony/SignalStrength.java fixes the signal strength not shown correctly issue. (For Android versions lower than 8.x)

The following files modify to fix the data call not happening when coming out of Airplane mode problem. Add support OEM unsolicited response from AT command base modem. The changes are also necessary for HL series FOTA through AVMS. This is applicable for Android 6.x or later.

- frameworks/opt/telephony/src/java/com/android/internal/telephony/Phone.java
- frameworks/opt/telephony/src/java/com/android/internal/telephony/GsmCdmaPhone.java
- frameworks/opt/telephony/src/java/com/android/internal/telephony/OemHookIndication.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. This is applicable for Android 6.x or lower.

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. This is applicable for Android 5.x or later.

The following changes are necessary to add support for OMADM GUI:

- frameworks/opt/telephony/src/java/com/android/internal/telephony/BaseCommands.java
- frameworks/opt/telephony/src/java/com/android/internal/telephony/GsmCdmaPhone.java
- frameworks/opt/telephony/src/java/com/android/internal/telephony/dataconnection/DcTracker.java
- packages/apps/Dialer/java/com/android/dialer/dialpadview/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:

For Android 10 or later, AndroidFS/vendor/\* should be used instead of AndroidFS/system/vendor/\*

- AndroidFS/(vendor|system/vendor)/bin/swihltrace
- AndroidFS/(vendor|system/vendor)/bin/slqssdk
- AndroidFS/(vendor|system/vendor)/bin/SierraDMLog
- AndroidFS/(vendor|system/vendor)/bin/SierraFwDI7xxx
- AndroidFS/(vendor|system/vendor)/bin/SierralmgMgr
- AndroidFS/(vendor|system/vendor)/bin/SierraSARTool
- AndroidFS/(vendor|system/vendor)/bin/DownloadTool
- AndroidFS/(vendor|system)/bin/dhpcpd
- AndroidFS/(vendor|system/vendor)/lib64/libsierra-ril.so
- AndroidFS/(vendor|system/vendor)/lib64/libsierraat-ril.so
- AndroidFS/(vendor|system/vendor)/lib64/libsierrahl-ril.so
- AndroidFS/(vendor|system/vendor)/lib64/libswims.so
- AndroidFS/(vendor|system/vendor)/lib64/libswiqmiapi.so
- AndroidFS/(vendor|system/vendor)/lib64/libDownloadTool.so
- AndroidFS/(vendor|system/vendor)/lib/libDownloadTool.so
- AndroidFS/system/app/com.sierra.logs/com.sierra.logs.apk
- AdnroidFS/system/app/swifota/swifota.apk
- AndroidFS/system/app/swiomadm/swiomadm.apk
- AndroidFS/system/app/simswitcher/simswitcher.apk

For 32 bit platform, following files should be copied:

- AndroidFS/(vendor|system/vendor)/bin/swihltrace
- AndroidFS/(vendor|system/vendor)/bin/slqssdk
- AndroidFS/(vendor|system/vendor)/bin/SierraDMLog
- AndroidFS/(vendor|system/vendor)/bin/SierraFwDI7xxx
- AndroidFS/(vendor|system/vendor)/bin/SierralmgMgr
- AndroidFS/(vendor|system/vendor)/bin/SierraSARTool
- AndroidFS/(vendor|system/vendor)/bin/DownloadTool
- AndroidFS/(vendor|system)/bin/dhccpd
- AndroidFS/(vendor|system/vendor)/lib/libsierra-ril.so
- AndroidFS/(vendor|system/vendor)/lib/libsierraat-ril.so
- AndroidFS/(vendor|system/vendor)/lib/libsierrahl-ril.so
- AndroidFS/(vendor|system)/lib/libswims.so
- AndroidFS/(vendor|system)/lib/libswiqmiapi.so
- AndroidFS/(vendor|system)/lib/libDownloadTool.so
- AndroidFS/system/app/com.sierra.logs/com.sierra.logs.apk
- AndroidFS/system/app/swifota/swifota.apk
- AndroidFS/system/app/swiomadm/swiomadm.apk
- AndroidFS/system/app/simswitcher/simswitcher.apk

For Android 7.0 and newer, AndroidFS/(vendor|system)/bin/dhccpd should be copied to /(vendor|system)/bin on the target device file system.

For QMI RIL, AndroidFS/(vendor|system/vendor)/lib64/hw/gps.default.so should be copied to /(vendor|system)/lib/hw on the target device file system and renamed appropriately for the target device.

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

For Android 10.0 with gnss@2.0 HAL (RIL release V10.0.3.0\_ARM64 or later) AndroidFS/vendor/bin/hw/android.hardware.gnss@2.0-service-swi should be copied to /(vendor|system)/bin/hw on target device file system and renamed appropriately for the target device.

**AndroidFS/vendor/etc/init/android.hardware.gnss@2.0-service-swi.rc** should be copied to /(vendor|system)/etc/init.on target device file system and renamed appropriately for the target device and remove android.hardware.gnss@1.0-service in platform.

The following files should be copied to the target device system for Android 10.0 with radio@1.4 HAL (RIL release V10.0.3.0\_ARM64 or later)

- AndroidFS/vendor/bin/hw/rild-swi
- AndroidFS/vendor/lib64/libril-swi.so
- AndroidFS/vendor/lib64/librilutils.so
- AndroidFS/vendor/etc/init/rild\_swi.rc
- Set ENABLE\_VENDOR\_RIL\_SERVICE to true in platform makefile. And add PRODUCT\_PROPERTY\_OVERRIDES += vendor.rild.libpath=/vendor/lib64/libsierra-ril.so

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

- AndroidFS/(vendor|system/vendor)/bin/init.dhccpd
- AndroidFS/(vendor|system/vendor)/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.hikey.rc
- AndroidFS/ueventd.hikey.rc
- AndroidFS/(vendor|system/vendor)/etc/init/rild.legacy.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/(vendor|system/vendor)/bin/init.dhcpd
- AndroidFS/(vendor|system/vendor)/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 must 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/dhcpd.te
- device/linaro/hikey/sepolicy/file.te
- device/linaro/hikey/sepolicy/file\_contexts
- device/linaro/hikey/sepolicy/init.te
- device/linaro/hikey/sepolicy/netd.te
- device/linaro/hikey/sepolicy/init-insmod-sh.te
- device/linaro/hikey/sepolicy/property\_contexts
- device/linaro/hikey/sepolicy/rild.te
- device/linaro/hikey/sepolicy/sierra\_config\_ip.te
- device/linaro/hikey/sepolicy/sierra\_dhcpd.te
- device/linaro/hikey/sepolicy/hal\_gnss\_default.te
- device/linaro/hikey/sepolicy/vendor\_init.te
- device/linaro/hikey/sepolicy/shell.te
- device/linaro/hikey/sepolicy/webview\_zygote.te

Visit <https://source.android.com/security/selinux> for more information about Android SELinux.

## 3.5. HAL Definitions

The following changes are necessary for Android 9.x or newer to add definition into device manifest.xml:

### 3.5.1. Radio

```
<hal format="hidl">
  <name>android.hardware.radio</name>
  <transport>hwbinder</transport>
  <version>1.4</version>
  <interface>
    <name>IRadio</name>
    <instance>slot1</instance>
  </interface>
</hal>
```

### 3.5.2. GNSS

```
<hal format="hidl">
  <name>android.hardware.gnss</name>
  <transport>hwbinder</transport>
  <version>1.1</version>
  <version>2.0</version>
  <interface>
    <name>IGnss</name>
    <instance>default</instance>
  </interface>
</hal>
```

## 3.6. Switching Between HL / AT / QMI RIL

### 3.6.1. Android Version 5.x to 9.x

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 persist.sys.ril.type [HL | AT | QMI]`
4. `adb reboot`

### 3.6.2. Android Version 10.x 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.

```
persist.sys.ril.type=[QMI | HL | AT]
```

```
rild.libpath=/vendor/lib64/[libsierra-ril.so | libsierrahl-ril.so | libsierraat-ril.so ]
```



**4.**

## **RIL Source File Documentation**

Preliminary 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.

---

## 5.1.1. Custom RIL Commands

(Deprecated on Android 10 or later)

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 here:

<code>SWI_CURRENT_STATE</code>	<code>= 1</code>
<code>SWI_ON_SUPPORTS</code>	<code>= 2</code>
<code>SWI_ON_CANCEL</code>	<code>= 3</code>
<code>SWI_GET_VERSION</code>	<code>= 4</code>
<code>SWI_CDMA_ACTIVATION</code>	<code>= 5</code>
<code>SWI_DO_FACTORY_RESET</code>	<code>= 6</code>
<code>SWI_PRL_UPDATE</code>	<code>= 7</code>
<code>SWI_GET_SAR_STATE</code>	<code>= 8</code>
<code>SWI_SET_SAR_STATE</code>	<code>= 9</code>
<code>SWI_SET_SMS_WAKE</code>	<code>= 10</code>
<code>SWI_GET_ppDATAp</code>	<code>= 11</code>
<code>SWI_SET_ppDATAp</code>	<code>= 12</code>
<code>SWI_VALIDATE_SPC</code>	<code>= 13</code>
<code>SWI_READ_SIM_PB</code>	<code>= 14</code>
<code>SWI_PRL_CANCEL</code>	<code>= 15</code>
<code>SWI_GET_NOISE_SP</code>	<code>= 16</code>
<code>SWI_SET_NOISE_SP</code>	<code>= 17</code>
<code>SWI_GET_MODEM_VOL</code>	<code>= 18</code>
<code>SWI_SET_MODEM_VOL</code>	<code>= 19</code>
<code>SWI_GET_COLP</code>	<code>= 20</code>
<code>SWI_GET_COLR</code>	<code>= 21</code>
<code>SWI_SET_COLP</code>	<code>= 22</code>
<code>SWI_CLEAR_COLP</code>	<code>= 23</code>
<code>SWI_SET_COLR</code>	<code>= 24</code>
<code>SWI_CLEAR_COLR</code>	<code>= 25</code>
<code>SWI_OMA_DM_START_SESS</code>	<code>= 30</code>

```
SWI_OMA_DM_CANCEL_SESS = 31
SWI_OMA_DM_SET_SELECTION = 32
SWI_OMA_DM_GET_DESCRIPTION=33
SWI_CDMA_ACTIVATION_VERIZON_SL9090= 34
SWI_VOICE_GET_CONFIG = 35
SWI_SET_DDTM_PREF = 36
SWI_GET_AGPS_CONFIG = 37
SWI_SET_AGPS_CONFIG = 38
SWI_GET_3GPP2_INFO = 39
SWI_SET_3GPP2_INFO = 40
SWI_GET_CDMA_PROVISION_STATUS= 41
SWI_RESET_CDMA_RADIO_STATE= 42
SWI_GET_CDMA_CARRIER = 43
SWI_SET_AUDIO_PROFILE = 44
SWI_CANCEL_QUERY_AVAILABLE_NETWORKS = 45
SWI_GO_DORMANT = 46
SWI_GET_DORMANT_STATE = 47
```

## 5.1.2. CDMA Activation

For CDMA activation, `strings` should be:

```
strings[0] = "5"; //SWI_CDMA_ACTIVATION
```

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

## 5.1.3. User Initiated PRL Update

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

```
strings[0] = "7"; // SWI_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.

## 5.1.4. Factory Reset

For factory reset, `strings` should be:

```
strings[0] = "6"; // SWI_DO_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.

## 5.1.5. Get Current SAR Back-off State

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

```
strings[0] = "8"; // SWI_GET_SAR_STATE
```

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.

## 5.1.6. Set Current SAR Back-off State

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

```
strings[0] = "9"; // SWI_SET_SAR_STATE
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. To enable this phone object, the following must be done:

5. Add the property setting `telephony.lteOnCdmaDevice=1` to an appropriate property file, such as `system.prop`.
6. 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 Switching to a New Carrier Image

```
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 Images 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 List of Carrier Images 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 Switching to a New Carrier Image 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 EM/MC73xx.

The SierraFwDI7xxx executable is provided to download firmware to AirPrime EM/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 EM/MC73xx 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
- -d --download  
Download image to modem. Note that use -p option as well for provide path of images.
- -l --listimages  
Display image information on device (located on Device as well as Host). Use -p option as well to get the information of images located on Host.
- -m --module <type>  
Module type: 9x15, 9x30, 9x50 or 9x07. Note that this option is mandatory parameter.
- -p --pathhost <imagepath>  
Absolute path of images located on host. Note that this option is provided either with -d or with -l.
- -f --force  
Force firmware download regardless of host firmware version. Note that this option only valid for MC73xx.

## 5.5. Firmware Image Download on EM/MC74xx, EM75xx, WP76xx and RC76xx.

The SierraFwDI7xxx is used to download/upgrade firmware for EM74xx and EM7565 modules also. This tool is available in /system/bin. All options of this tools are not applicable for EM74xx and EM7565 modules. Command options and usage of options which are applicable options are 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 on modules (located on Device as well as Host). Use `-p` option as well to get the information of images located on Host.
- `-d --download`  
Download image to modem. Note that use `-p` option as well for provide path of images.
- `-m --module <type>`  
Module type: 9x15, 9x30, 9x50 or 9x07. Note that this option is mandatory parameter.
- `-p --pathhost <imagepath>`  
Absolute path of images 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 modem with `<image index>`. Image index is displayed by `-l`

## 5.6. Firmware Image Download on WP75xx

The SierraFwDI7xxx is used to download/upgrade firmware for WP75xx modules also. This tool is available in `/system/bin`. All options of this tools are not applicable for WP75xx modules. Command options and usage of options which are applicable options are 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 on modules (located on Device as well as Host). Use `-p` option as well to get the information of images located on Host.
- `-d --download`  
Download image to modem. Note that use `-p` and `-s` option as well for provide path of images and slot located on Device.
- `-m --module <type>`  
Module type: 9x15Y for WP75xx. Note that this option is mandatory parameter.

- `-p --pathhost <imagepath>`  
Absolute path of images located on host. Provide path as `-p /data/EM7455` if firmware is located at `/data/EM7455/1..`
- `-s --selectimage <imageindex>`  
Switch image for modem with `<image index>`. Image index is displayed by `-l`

## 5.7. 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 ./swisdsk -f /data/<file.cwe> -x
```

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

## 5.8. 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 `libDownloadtool.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 DownloadTool. If not specifies `/system/bin/DownloadTool` is taken into consideration
- `-v <level>`  
For verbose level, value from `[0-4]`

## 5.9. 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.

## 5.10. 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.

## 5.11. 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
```

## 5.12. 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

## 5.13. RIL Properties

RIL operation can be modified by setting various Android properties.

### 5.13.1. persist.vendor.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.

This is may be not necessary for Android 9.x or newer.

### 5.13.2. `persist.vendor.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.

### 5.13.3. `ro.vendor.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.

### 5.13.4. `persist.vendor.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.

### 5.13.5. `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.

### 5.13.6. `persist.sierra.sdk.debug`

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.

### 5.13.7. `persist.vendor.sys.ril.type`

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

## 5.14. 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
  - (vendor.)telephony.lteOnGsmDevice=1
  - (vendor.)telephony.lteOnCdmaDevice=0
- CDMA LTE module detected
  - (vendor.)telephony.lteOnGsmDevice=0
  - (vendor.)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)