Abstract
This document describes how to build and use the RIL library for u-blox cellular modules on the Android operating system.
Document information

<table>
<thead>
<tr>
<th>Title</th>
<th>Android RIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtitle</td>
<td>Source code</td>
</tr>
<tr>
<td>Document type</td>
<td>Application note</td>
</tr>
<tr>
<td>Document number</td>
<td>UBX-13002041</td>
</tr>
<tr>
<td>Revision and date</td>
<td>R24 06-Sep-2019</td>
</tr>
</tbody>
</table>

Disclosure restriction

This document applies to the following products:

<table>
<thead>
<tr>
<th>Product name</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOBY-L2 series</td>
</tr>
<tr>
<td>TOBY-L4 series</td>
</tr>
<tr>
<td>TOBY-R2 series</td>
</tr>
<tr>
<td>LARA-R2 series</td>
</tr>
<tr>
<td>SARA-R4 series</td>
</tr>
<tr>
<td>LISA-U2 series</td>
</tr>
<tr>
<td>SARA-U2 series</td>
</tr>
<tr>
<td>LISA-C2 series</td>
</tr>
<tr>
<td>LEON-G1 series</td>
</tr>
<tr>
<td>SARA-G3 series</td>
</tr>
<tr>
<td>SARA-G4 series</td>
</tr>
</tbody>
</table>

u-blox or third parties may hold intellectual property rights in the products, names, logos and designs included in this document. Copying, reproduction, modification or disclosure to third parties of this document or any part thereof is only permitted with the express written permission of u-blox.

The information contained herein is provided “as is” and u-blox assumes no liability for its use. No warranty, either express or implied, is given, including but not limited to, with respect to the accuracy, correctness, reliability and fitness for a particular purpose of the information. This document may be revised by u-blox at any time without notice. For the most recent documents, visit www.u-blox.com.

Copyright © u-blox AG.
Android RIL - Application note

C.3.1 BoardConfig.mk configuration...........................................................................17
C.3.2 Init.rc configuration.............................................................................................17

D AT pass through commands .........................................................................................19

E Default EPS bearer in LTE (initial PDP context) .........................................................21
E.1 Default EPS bearer in LTE.........................................................................................21
E.2 apns-conf.xml configurations (manually/during build process)...............................21
E.3 apns-conf.xml configurations (Android UI) .............................................................21
E.4 Single default EPS bearer allowed – RIL handling .................................................22

F Repository file configuration .........................................................................................23
F.1 TOBY-L2 USB profile configuration .........................................................................23
  F.1.1 USB profiles configuration +UUSBCONF............................................................23
  F.1.2 Boot mode configuration +UBMCONF ...............................................................23
F.2 CDMA network operator name .................................................................................24
F.3 Timeouts ..................................................................................................................24
F.4 Emergency numbers .................................................................................................24

G Module firmware update ..............................................................................................25

H Multi module support in Android RIL ........................................................................26
H.1 Multi module .............................................................................................................26
  H.1.1 Prerequisites ......................................................................................................26
  H.1.2 Mode switching .................................................................................................26
  H.1.3 Switching status .................................................................................................26

I Verizon network ............................................................................................................27
I.1 init.ublox.rc ..............................................................................................................27
I.2 core_ublox.mk ...........................................................................................................27

J Module specific configurations ....................................................................................27
J.1 SARA-R4 ...................................................................................................................27
  J.1.1 Enable USB serial drivers in the kernel ...............................................................27

K Audio codec ................................................................................................................28
K.1 Configuration ............................................................................................................28
K.2 Example ....................................................................................................................28

L i.MX platform specific notes .......................................................................................28
L.1 imx6.mk ...................................................................................................................28
L.2 apn-config.xml ........................................................................................................28

M Android RIL integration FAQ ....................................................................................29
Related documents ..........................................................................................................31
Revision history .............................................................................................................32
Contact ..........................................................................................................................33
1 Introduction

☞ An index finger points out key information pertaining to integration and performance.

⚠ A warning symbol indicates actions that could negatively impact performance or damage the device.

☞ This revision of the application note applies to the Android RIL v12.00 and onwards.

The software was developed for the following Android versions (AOSP) and platforms:

- Android 2.3 (Gingerbread): BeagleBoard-XM
- Android 4.0 (Ice Cream Sandwich): BeagleBoard-XM
- Android 4.1, 4.2, 4.3 (Jelly Bean): PandaBoard
- Android 4.4 (KitKat): PandaBoard
- Android 5.0 (Lollipop): Nexus 5
- Android 6.0 (Marshmallow): Nexus 5
- Android 7.0 (Nougat): Nexus 5X
- Android 8.0 (Oreo): Nexus 5X
- Android 9.0 (Pie): Google Pixel 2

See the RIL release note for the list of Android software deliveries and interfaces supported by u-blox cellular modules.

In the following chapters, <name> indicates a parameter that can:

- be customized
- be set corresponding to the system configuration
- provide software version

The following chapters describe the production packages delivered by u-blox.

An overview of the system setup is provided as well as the procedure to perform a log.

The following symbols are used to highlight important information within this document:

☞ The RIL source code provided by u-blox is for reference purposes only.

☞ u-blox RIL does not control or manage the device (e.g. u-blox module or any other peripheral) GPIOs, such as to power off/on or reset the module. Power management is under the responsibility of the Android OS and the kernel subsystem.

⚠ u-blox assumes no responsibility for inappropriate use of RIL by customers.
2 Production delivery

2.1 Software release

The delivery consists of the RIL library source code.

⚠ Android RIL does not offer a standard interface for powering off the module, so it is necessary to modify the Android Java framework application that manages system power-off and sends the AT+CPWROFF command.

2.2 Android delivery contents

☞ The Android delivery contents structure for all Android versions is described below. If the RIL_sc_<version>.zip does not include a component mentioned in the delivery contents, it means that the package has been updated for that release and the component is no longer required.

2.2.1 Android 2.3 delivery contents

The RIL production delivery is provided in the RIL_sc_<version>.zip compressed file with the content structured as follows:

RIL_sc_<version>.zip
ril_sc_<version> Source overlay for Android platform
external/ppp/ Data connection files
hardware/ril/rapid_ril/ RIL core directory
system/core/init/property_service.c Set GPRS service property
system/core/rootdir/Android.mk Compilation script
system/core/rootdir/uventd.rc Device permission
system/core/rootdir/etc/init.gprs-pppd Script for starting PPP service
system/core/rootdir/etc/ppp/ PPP script directory
system/core/rootdir/etc/rril-repo.sh RIL script
system/core/rootdir/etc/rril/repository.txt RIL parameters

2.2.2 Android 4.x, 5.x, 6.x, 7.x, 8.x, 9.x delivery contents

The RIL production delivery is provided in the RIL_sc_<version>.zip compressed file with the content structured as follows:

RIL_sc_<version>.zip
ril_sc_<version> Source overlay for Android platform
external/ppp/chat PPP chat files
external/ppp/pppd Data connection files for PPP
hardware/ril/ublox_ril/ RIL core directory
system/core/liblog/logd_write.c Log configuration file
system/core/rootdir/init.ublox.rc u-blox Init.rc file
build/target/product Device configuration folder files
build/target/product/core_ublox.mk Make file include all scripts
build/target/product/rootdir/etc RIL data scripts for PPP and RNDIS modes
build/target/product/rootdir/etc/ppp Scripts for data connection
build/target/product/rootdir/etc/uril RIL configuration repository file

For android 4.x specific delivery contents

system/core/rootdir/Android.mk Compilation script
system/core/init/property_service.c Set RIL services property

For android 9.x specific delivery contents

system/core/libnetutils/Android.mk Build dhcプット for RNDIS mode
system/netd/netutils_wrappers/NetUtilsWrapper-1.0.cpp networking configuration file
3 Build source code

☞ This section applies to the Linux operating system.

- Create the build directory using the following commands, where `<android_root>` is the directory name:
  
  ```
  mkdir <android_root>
  cd <android_root>
  ```

- Download an Android distribution for the platform. Follow the tutorial, manual, etc. related to the platform’s distribution.

⚠ Android distribution must be one of the deliveries listed in section 1. Full functionality of software in other Android OS versions is not guaranteed.

- Create a directory and decompress the RIL file

⚠ The password provided from u-blox to decrypt the compressed RIL file is needed.

  ```
  mkdir <ril_dir>
  cd <ril_dir>
  cp <path_of_ril>/RIL_sc_<version>.zip .
  unzip --P <ril_password> RIL_sc_<version>.zip
  ```

- Copy the RIL source code into the Android distribution

- For Android 4.x/5.x/6.x/7.x/8.x/9.x delivery:

  ```
  cp -pvRf ril_sc_<version>/Android<version>/external/ppp/* <android_root>/external/ppp
  cp -pvRf ril_sc_<version>/Android<version>/system/* <android_root>/system
  cp -pvRf ril_sc_<version>/Common/build/target/product/<android_root>/build/target/product
  cp -pvRf ril_sc_<version>/Common/system/core/rootdir/<android_root>/system/core/rootdir
  cp -pvRf ril_sc_<version>/Common/hardware/ril/ublox_ril <android_root>/hardware/ril/
  ```

☞ The delivery overwrites or adds some Android system files.

- Modify the kernel configuration as in appendix A and configure the script files module connection to the Android platform as described in:
  
  o Appendix B: Android 4.1/4.2/4.3/4.4/5.0/6.0/7.0/8.0/9.0 module configuration
  o Appendix Error! Reference source not found.: for Android 4.0 (Ice Cream Sandwich)
  o Appendix C: for Android 2.3 (Gingerbread)

☞ For module specific modifications, see appendix J.

☞ For information on MUX interfacing, see the Multiplexer implementation application note [5].

- Build the Android system for the device or for the emulator

☞ Use the tutorial, manual, etc. related to the platform’s distribution.

- Insert a microSD card (minimum 2 GB) into the PC. Determine the SD card device name. Prompt `dmesg` command on a system shell. An example of the result of this command is below

  ```
  [85560.292608] sd 6:0:0:2: [sdd] Assuming drive cache: write through
  [85560.292613] sd 6:0:0:2: [sdd] Attached SCSI removable dis
  ```

  The device name is written in the brackets (e.g. sdd)
• The next step partitions and populates the microSD card with the root file system, bootloader, etc.
• Use the command provided in the platform distribution to create an SD card for the platform

⚠ This step can erase the hard disk drive if the SD card device name entered after the script name is incorrect.

• Insert the SD card into the board's SD slot
• Connect the board with the u-blox cellular module using one of the supported interfaces
• Power on both devices
4 Debug the RIL

4.1 Host PC configuration

4.1.1 Windows configuration

- On a Windows host, download and extract the Android SDK [4]
- Update the Android SDK using the command:
  ```
  \<home\>\android-sdk-windows\tools\android.bat
  ```
- Download the USB driver for Android [3] and extract it into
  ```
  \<home\>\android-sdk-windows\usb-drivers\n  ```
- Modify the section [Google.NTx86] of the `android_winusb.inf` in the `<home>\android-sdk-windows\usb-drivers\usb_driver_r03-windows` directory with the following lines:
  ```
  \;TI_EVM
  \%SingleAdbInterface%        = USB_Install, USB\!VID_18D1!\&PID_9018
  \%CompositeAdbInterface%     = USB_Install, USB\!VID_18D1!\&PID_9018\&MI_01
  ```
- Run a shell and type the following command:
  ```
  echo 0x18D1 > "%USERPROFILE%\android\adb_usb.ini"
  ```
- Install the downloaded USB drivers for the Android OS
- Connect the device to the PC
- The device is now ready to start communications

4.1.2 Linux configuration

- On a Linux host, download and extract the Android SDK [4]
- On Linux, type the following commands in a shell:
  ```
  sudo su <passwd>
  mkdir ~/.android
  vi ~/.android/adb_usb.ini
  echo "0x0451" > ~/.android/adb_usb.ini
  ```
- Connect the device to the PC
- The device is now ready to start communications

4.2 RIL log

Set up the Android Debugger (ADB) as mentioned above. By means of the ADB, it is possible to choose which RIL radio- and telephony-related log messages to receive by using the following command:

```
adb logcat -v time -b radio
```

The above command generates a log output like the following example:

```
02-24 01:55:22.034 D/RILD   (  512): **RIL Daemon Started**
02-24 01:55:22.034 D/RILD   (  512): **RILd param count=6**
```
The logging level can be set in the Group Logging section of the repository.txt file by changing the value of the LogLevel field.

**File path:** build/target/product/rootdir/etc/uril/repository.txt

**Allowed values:** Table 1 lists the various values that are currently supported.
<table>
<thead>
<tr>
<th>LogLevel</th>
<th>Logging levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Verbose</td>
</tr>
<tr>
<td>2</td>
<td>Info</td>
</tr>
<tr>
<td>3</td>
<td>Warning</td>
</tr>
<tr>
<td>4</td>
<td>Critical</td>
</tr>
</tbody>
</table>

Table 1: Supported logging levels

**Example:** The following lines are present in the `Repository.txt` file to define the log level.

```plaintext
// Log levels
Group Logging
LogLevel 2  // 4: Critical, 3: Warning, 2: Info, 1: Verbose
```

## 4.4 PPPD debug log

To enable the PPPD log, **debug** should be uncommented in the corresponding files.

```plaintext
build/target/product/rootdir/etc/ppp/gprsX

(where X is the PPPD interface used by RIL)
```

```plaintext
# uncomment the following line to debug pppd
debug
ttyACM1
crtscts
nodetach
usepeerdns
defaultroute
ipcp-max-failure 30
connect "chat -t 15 -f /system/etc/ppp/chat-ispl"
```
Appendix

A Kernel configuration

Configure the kernel to support the u-blox cellular module and the applicable interface connection.

A.1 USB kernel configuration

Enable the kernel configuration as follows:

```
Device Drivers
  USB Support
    <*>USB Modem (CDC ACM) support
```

A.2 UART kernel configuration

This device is enabled in the default kernel configuration.

A.3 SPI kernel configuration

Enable the kernel configuration as follows:

```
Device Drivers
  SPI Support
    <*>Debug support for SPI driver
    <*>GPIO-based bitbangning SPI Master
    <*>McSPI driver for OMAP24xx/OMAP34xx
```

⚠️ Implement the dedicated SPI protocol (see the SPI interface application note [1]) on the SPI kernel driver to correctly work with the cellular modules supporting the SPI interface.

A.4 RNDIS / ECM kernel configuration

Enable the kernel configuration as follows:

```
Device Drivers
  Network device support
  USB Network Adapters
    <*>Multi-purpose USB Networking Framework
    <*>CDC Ethernet support
    <*>Host for RNDIS and ActiveSync devices
```

⚠️ This configuration is required only for the TOBY-L2 series modules configured with 1 CDC-ACM port and 1 RNDIS (factory-programmed configuration) or 1 ECM (AT+UUSBCONF=2).

A.5 PPP kernel configuration

Enable the kernel configuration as follows:

```
Device Drivers
  Network device support
    <*>PPP (point-to-point protocol) support
    <*>PPP support for async serial ports
    <*>PPP support for sync tty ports
    <*>PPP Deflate compression
```

☞ Check if the “chat” package is included in the build. If not, add it using the following line in one of the .mk files included in the build.

```
PRODUCT_PACKAGES += chat
```
B. Android 4.x/5.x/6.x/7.x/8.x/9.0 module configuration

TOBY-L2 series only: See appendix F.1 for the automatic profile configuration.

B.1 core.mk configuration

Following modifications are required to include core_ublox.mk in android source.

File Path: build/target/product/core.mk

Modifications: Add below mentioned line at end of core.mk

```make
#***u-blox Modifications***
$(call inherit-product, $(SRC_TARGET_DIR)/product/core_ublox.mk)
#***u-blox Modifications***
```

B.2 ueventd.rc configuration

This file is used to create or remove the device node (/dev/xxx) by receiving uevent messages from the kernel. To create the node for the u-blox modules, add the following configuration in ueventd.[platform].rc or ueventd.rc. Where [platform] labels depend on the Board Support Package (BSP), following these permissions:

Modifications:

- **For USB/ttyACM mode:**
  ```
  /dev/ttyACM* 0660 radio radio # [if using CDC-ACM Interface]
  ```

- **For UART/MUX mode:**
  ```
  /dev/<serial device> 0660 radio radio # [if using VCP Interface e.g. ttyUSB, ttyS or ttymxc]
  /dev/pts* 0660 radio radio # [if using pts Interface with MUX]
  ```

- **For PPP mode:**
  ```
  /dev/pts 0660 radio radio
  ```

  If PPP is already added with different permissions other than radio, then modify it to “radio radio”.

B.3 device.mk

File Path [AOSP]: [ANDROID_SOURCE]/device/<platform>/device.mk
File Path [i-MX8]: [ANDROID_SOURCE]/device/fsl/imx8m/evk_8mq/evk_8mq.mk

Modifications:

- Set port path and ril lib path using following lines.
  - **For Android 9.0 Pie**
    ```
    vendor.rild.libpath=/vendor/lib64/librapid-ril-core.so
    vendor.rild.libargs=-a/dev/ttyACM0
    ```
  - **For Android 4.x/5.x/6.x/7.x/8.x**
    ```
    rild.libpath=/system/lib64/librapid-ril-core.so <For 64-bit architecture>
    rild.libpath=/system/lib/librapid-ril-core.so  <For 32-bit architecture>
    rild.libargs=-a/dev/ttyACM0
    ```

  Android 9.0 Pie has changed the binaries paths from /system to /vendor.

  For TOBY-L4 series only use interface name ttyACM2.

  Update following property to set default network type to global (LTE, GSM, WCDMA).

```
ro.telephony.default_network=9
```
B.4 init.rc configuration

Following modifications are required to include init.ublox.rc in Android source.

File Path [AOSP]: [android_source]/system/core/rootdir/init.rc
File Path [i-MX6]: [android_source]/device/fsl/sabresd_6dq/init.rc

Modifications: Add below mentioned line at top of init.rc.

```bash
#***u-blox Modifications***#
import /init.ublox.rc
#***u-blox Modifications***#
```

B.5 init.ublox.rc configuration

The init.ublox.rc file contains all the necessary information used for the RIL integration. All the services, properties, permissions used for u-blox RIL are mentioned in this file.

This file is self-explanatory and contains all the steps used for the RIL initialization. Follow these steps to configure this file as per requirements.

☞ In the RIL delivery package the init.ublox.rc file at the following path:
ril_sc_<version>/Common/system/core/rootdir

☞ [For Android 5 and above] After the integration follow the steps mentioned in init.ublox.rc (the default setting is for (usb)ttyACM).

B.6 SEPolicy configuration

To get started with SELinux:

1. Enable SELinux in the kernel: CONFIG_SECURITY_SELINUX=y
   (By default it is enabled in the Android Open Source Project AOSP)

2. Change the kernel_cmdline parameter (device/<vendor>/<target>/BoardConfig.mk) in the following way:
   ```bash
   BOARD_KERNEL_CMDLINE := androidboot.selinux=permissive
   ```

Permissive mode is only set for initial development phase. Once RIL integration is completed, it is recommended to set the device again to enforcing mode.

☞ For further details about the SEPolicy configuration, please contact the u-blox technical support.
C Android 2.3 module configuration

Configure the kernel to support the u-blox cellular module and the applicable interface connection.

C.1 USB

To enable the USB module connection, configure the BoardConfig.mk, init.rc and ueventd.[platform].rc files as follows.

C.1.1 BoardConfig.mk configuration

Add the following lines to the BoardConfig.mk file to build the RIL library for the USB connection:

```makefile
# u-blox RIL communication interface
RIL_COM_INTERFACE := usb
```

C.1.2 Init.rc configuration

Add the following lines to the init.rc file:

```bash
# Change permissions for modem
chmod 0660 /dev/ttyACM0
chown radio radio /dev/ttyACM0
chmod 0660 /dev/ttyACM1
chown radio radio /dev/ttyACM1
chmod 0660 /dev/ttyACM2
chown radio radio /dev/ttyACM2

# Set permissions for the u-blox RIL Repository
chown radio radio /system/etc/rril
chmod 0770 /system/etc/rril
chmod 0660 /system/etc/rril/repository.txt

# Set the u-blox RIL repository state
setprop net.rril.repository notready

# Prepare the u-blox RIL repository
service rril-repo /system/bin/rril-repo.sh
user root
group radio
oneshot

# Load the u-blox RIL
service ril-daemon /system/bin/rild -l /system/lib/librapid-ril-core.so -- -a
/dev/ttyACM0 -n /dev/ttyACM1
socket rild stream 660 root radio
socket rild-debug stream 660 radio system
user root
group radio cache inet misc audio

service pppd_gprs /system/bin/init.gprs-pppd
user root radius
group radio cache inet misc
disabled
oneshot

service pppd_term /system/bin/stop_pppd 15
disabled
oneshot

service pppd_kill /system/bin/stop_pppd 9
```
C.1.3 Ueventd.[platform].rc configuration

Add the following permissions to the ueventd.[platform].rc, where "platform" label depends on the BSP:

/dev/ttyACM0 0660 radio radio
/dev/ttyACM1 0660 radio radio
/dev/ttyACM2 0660 radio radio

C.2 UART

☞ To enable the UART module connection, configure the BoardConfig.mk, init.rc and ueventd.rc files as follows.

C.2.1 BoardConfig.mk configuration

☞ Not required for RIL release 09.00 and above.

To build the RIL library for a UART connection, add the following lines to the BoardConfig.mk file:

```makefile
# u-blox RIL communication interface
RIL_COM_INTERFACE := uart
```

C.2.2 Init.rc configuration

Add the following lines to the init.rc file:

```bash
# Change permissions for the modem
chmod 0660 /dev/<serial_port>
chown radio radio /dev/<serial_port>

# Set permissions for the RIL repository
chown radio radio /system/etc/rril
chmod 0770 /system/etc/rril

   chown radio radio /system/etc/rril/repository.txt
   chmod 0660 /system/etc/rril/repository.txt

# Set the RIL repository state
setprop net.rril.repository notready

# Multiplexing device
service gsmmuxd /system/bin/logwrapper /system/bin/gsm0710muxd -s /dev/<serial_port> -n3 -v7 -mbasic
user radio
   group radio cache inet misc
oneshot

# Use this service for stopping gsmmuxd
service mux_stop /system/bin/stop_muxd 15
disabled
oneshot

# Prepare the u-blox RIL repository
service rril-repo /system/bin/rril-repo.sh
user root
   group radio
oneshot

# Load the u-blox RIL
```
service ril-daemon /system/bin/rild -l /system/lib/librapid-ril-core.so -- -a /dev/pts/0 -n /dev/pts/1
   socket rild stream 660 root radio
   socket rild-debug stream 660 radio system
   user root
   group radio cache inet misc audio

service pppd_gprs /system/bin/init.gprs-pppd
   user root radio
   group radio cache inet misc
   disabled
   oneshot

# Use these services for stopping pppd
#
# terminate pppd with SIGTERM
service pppd_term /system/bin/stop_pppd 15
   disabled
   oneshot

# forcefully kill pppd with SIGKILL
service pppd_kill /system/bin/stop_pppd 9
disabled
oneshot

C.2.3  Ueventd.rc configuration

Add the following permissions to the ueventd.rc file:

# Permissions for virtual devices created by MUX
/dev/pts*                 0660   radio      radio
# Permissions for the USB serial device
/dev/ttyUSB*               0660   radio      radio

☞ ttyUSB shall be replaced with ttyS when using the UART serial port of the platform.

C.3  SPI

☞ To enable the SPI module connection, configure the BoardConfig.mk and init.rc files as follows.

C.3.1  BoardConfig.mk configuration

To build the RIL library for the SPI connection, add the following lines to the Boardconfig.mk file:

    # u-blox RIL communication interface
    RIL_COM_INTERFACE := spi

C.3.2  Init.rc configuration

Add the following lines to the init.rc file:

    # Change permissions for modem
    chmod 0660 /dev/<spi_port>
    chown radio radio /dev/<spi_port>

    # Set permissions for u-blox RIL repository
    chown radio radio /system/etc/rril
    chmod 0770 /system/etc/rril
    chmod 0660 /system/etc/rril/repository.txt
    chown 0660 /system/etc/rril/repository.txt

    # Set u-blox RIL repository state
setprop net.rril.repository notready

# Multiplexing device
service gsmmuxd /system/bin/logwrapper /system/bin/gsm0710muxd -s/dev/<spi_port> -n3 -v7 -m basic
  user radio
  group radio cache inet misc
  oneshot

# Use this service for stopping gsmmuxd
service mux_stop /system/bin/stop_muxd 15
  class main
  disabled
  oneshot

# Prepare u-blox RIL repository
service rril-repo /system/bin/rril-repo.sh
  user root
  group radio
  oneshot

# Load u-blox RIL
service ril-daemon /system/bin/rild -l /system/lib/librapid-ril-core.so -- -a
  /dev/pts/0 -n /dev/pts/1
  socket rild stream 660 root radio
  socket rild-debug stream 660 radio system
  user root
  group radio cache inet misc audio
service pppd_gprs /system/bin/init.gprs-pppd
  user root radio
  group radio cache inet misc
  disabled
  oneshot

# use these services for stopping pppd
#
# terminate pppd with SIGTERM
service pppd_term /system/bin/stop_pppd 15
  disabled
  oneshot

# forcefully kill pppd with SIGKILL
service pppd_kill /system/bin/stop_pppd 9
  disabled
  oneshot
DAT pass through commands

The Android operating system does not allow sending commands to the module without using the RIL software layer, so two requests provided by RIL’s API must be used:

- `RIL_REQUEST_OEM_HOOK_RAW`: it passes raw byte arrays back and forth
- `RIL_REQUEST_OEM_HOOK_STRINGS`: it passes strings back and forth

The following is sample code that could be used to send commands directly to the module:

⚠ Make a backup of the `<android_root>/packages/apps/Settings` folder before making any changes to it.

- Add the following lines in the bottom of `packages/apps/Settings/AndroidManifest.xml` before the `<application>` field.

```xml
<activity android:name="RilOemHookTest" android:label="@string/testing_RIL_OEMHook"
    android:process="com.android.phone">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.DEVELOPMENT_PREFERENCE" />
    </intent-filter>
</activity>
```

- Add the following lines in the bottom of `packages/apps/Settings/res/values/strings.xml` file before the `</resources>` field.

```xml
<!-- For RIL OEM HOOK testing -->
<string name="ril_oem_choose">Choose a RIL OEM Hook API to test:</string>
<string name="foat_fls">Start FOAT (fls file=C800)</string>
<string name="foat_dffs">Start FOAT (dffs file=C801)</string>
<string name="radio_usb">Set Prop (modInf=USB)</string>
<string name="radio_uart">Set Prop (modInf=UART)</string>
<string name="radio_api1">API 1 (datalen=0)</string>
<string name="radio_api2">API 2 (datalen=1)</string>
<string name="radio_api3">API 3 (datalen=6)</string>
<string name="radio_api4">API 4 Type command :</string>
<string name="radio_run">RUN</string>
<string name="ril_oem_response">Module response:</string>
```

At the bottom of the `packages/apps/Settings/res/xml/testing_settings.xml` file before the `</PreferenceScreen>` field, add the following lines:

```xml
<PreferenceScreen
    android:title="@string/testing_RIL_OEMHook">
    <intent
        android:action="android.intent.action.MAIN"
        android:targetPackage="com.android.settings"
        android:targetClass="com.android.settings.RilOemHookTest" />
</PreferenceScreen>
```

- Run the following command to merge the OEM Hook test application code into the Android source.

```bash
cp -pvRf ril_sc_<version>/packages/apps/Settings/* <android_root>/packages/apps/Settings/
```

- Add the following Java code to the relevant file depending on the Android version being used:
It is possible to start this application from an Android shell using the following command or it can also be accessible using the Android secret code *#*#4636#*#*:

```
am start -a android.intent.action.MAIN -n com.android.settings/com.android.settings.RilOemHookTest
```

or

```
am start -a android.intent.action.MAIN -n com.android.settings/com.android.settings.TestingSettings
```

Now click on the “RIL OEM Hook Test” menu. A screen such as Figure 1 is displayed.

![Figure 1: RIL OEM Hook application](image)

The “Start FOAT” options are to be used for the module firmware update if the provided RIL and module are supported. See appendix G for more information on this. Apart from that, the first three APIs send bytes to the RIL layer to execute a set of commands.

The fourth API sends the command string to the module. Type the command into the white field and click the “RUN” button to execute the command. The module response is displayed in the bottom field.

⚠️ RIL uses the numeric error value (AT+CMEE=1) as the error result codes format. A change in the error report setting causes unexpected errors in the RIL behavior.
E Default EPS bearer in LTE (initial PDP context)

E.1 Default EPS bearer in LTE
An EPS bearer is established when the UE connects to a PDN, and remains established throughout the lifetime of the PDN connection to provide the UE with always-on IP connectivity to that PDN. This bearer refers to the default EPS bearer. The UE can have additional default bearers as well. Each default bearer comes with an IP address and it has nominal QoS applied by the LTE network.

A default EPS bearer is allocated for an APN. To create multiple "Default EPS bearers" in an LTE network, more than one APN should be working in the LTE network.

E.2 apns-conf.xml configurations (manually/during build process)
The apns-conf.xml file should have at least one corresponding MCC/MNC entry/record with type field "ia", where "ia" stands for "Initial Attach". Multiple values can be defined in the type field e.g. default,supl,mms,ia

Example: during the source build process, the user can enter the corresponding network settings in the apns-conf.xml file. The sample values are given below.

```xml
<apn carrier="Cosmote Wireless Internet"
mcc="202"
mnc="01"
apn=""
type="ia"
/>  
<apn carrier="Cosmote Wireless Internet"
mcc="202"
mnc="01"
apn="internet"
type="default,supl,mms"
/>   
```

E.3 apns-conf.xml configurations (Android UI)
The user can also set the APN according to the service provider manually through the Android UI.

- Go to Android’s main settings.
- Under Wireless & Networks, go to mobile networks (Figure 2). Click more settings to get this option. (Figure 3).
- Tap on Access Point Names (that is what APN stands for).
- Press menu, and then click New APN.
- Enter in all of the required information in each field by clicking into it and then clicking OK to save (see Figure 4).
- After having finished entering all the settings, click menu again and then Save.
Some LTE networks allow only one default EPS bearer establishment: a single working APN is available for the user.

In the scenario where only a single working APN is available, the RIL tries to establish data on CID=1 (CID=1 is requested by OS) when `RIL_REQUEST_SETUP_DATA_CALL` is triggered by the OS. If a single default bearer case is encountered, RIL failed in its maximum tries while establishing data on CID=1.

Once the RIL has failed to establish the data on the requested CID by the OS, the RIL tries to establish data on CID=4, which is already established during the time of the LTE attach process, on the next try of `RIL_REQUEST_SETUP_DATA_CALL` by the OS.
F Repository file configuration

Repository.txt is a part of the u-blox RIL source, and this file is used to set various user defined values for various fields. These fields are split into sections. Operational timeouts, TTY constants, RIL settings, last values, USB/UBM and log level etc. are some examples of the fields/values defined in this file.

F.1 TOBY-L2 USB profile configuration

The USB and UBM configurations can be set in the Group USBUBM section of the repository.txt file by setting the value of the USBCONF and UBMCONF fields respectively. The various values that are currently supported are shown in Table 2 and Table 3. If these settings are not configured, then the module’s default settings will be used.

<table>
<thead>
<tr>
<th>[USBCONF]</th>
<th>Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>CDC-ACM</td>
</tr>
<tr>
<td>2</td>
<td>CDC-ECM</td>
</tr>
<tr>
<td>3</td>
<td>RNDIS</td>
</tr>
</tbody>
</table>

Table 2: Supported USB configurations

<table>
<thead>
<tr>
<th>[UBMCONF]</th>
<th>Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Router</td>
</tr>
<tr>
<td>2</td>
<td>Bridge</td>
</tr>
</tbody>
</table>

Table 3: Supported u-blox modem configurations

F.1.1 USB profiles configuration +UUSBCONF

This command configures the active USB profile. Each USB profile includes a certain number of USB endpoints, depending on the overall USB functions of the USB profile. The USB profile switch is not performed at run-time. The settings are saved in the NVM at module power-off; the new configuration will be effective at the subsequent module reboot. The command syntax is:

AT+UUSBCONF=[<id>[,<network>[,<audio>]]]

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Command setting</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 CDC-ACM ports</td>
<td>AT+UUSBCONF=0 and AT+CFUN=16</td>
<td>The data connection uses the PPP process</td>
</tr>
<tr>
<td>1 CDC-ACM port and 1 RNDIS port</td>
<td>AT+UUSBCONF=3 and AT+CFUN=16</td>
<td>This is the factory-programmed configuration. The data connection uses RNDIS port (network usb&lt;X&gt;)</td>
</tr>
<tr>
<td>1 CDC-ACM port and 1 ECM port</td>
<td>AT+UUSBCONF=2 and AT+CFUN=16</td>
<td>The data connection uses ECM port (network eth&lt;X&gt;).</td>
</tr>
</tbody>
</table>

Table 4: TOBY-L2 series allowed profile configurations

F.1.2 Boot mode configuration +UBMCONF

This command configures the system networking mode. The chosen configuration is stored in the NVM and it will be applied at the subsequent module reboot. The command syntax is:

AT+UBMCONF=[<networking_mode>]

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Command setting</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Router mode</td>
<td>AT+UBMCONF=1 and AT+CFUN=16</td>
<td>Default and factory-programmed value: router mode</td>
</tr>
<tr>
<td>Bridge mode</td>
<td>AT+UBMCONF=2 and AT+CFUN=16</td>
<td>bridge mode</td>
</tr>
</tbody>
</table>

Table 5: TOBY-L2 series possible boot mode configurations
F.2 CDMA network operator name

The network operator names for CDMA are set in the “Operator Settings” group of the repository.txt file by setting the “OperatorNames” field to its appropriate name and the MCC and MNC values.

Example: the following lines are present in the Repository.txt file to define the network operator name.

```plaintext
//Operator Settings (only for CDMA)
Group OperatorNames
Supported Operator Sprint
MCCMNC 310120
```

F.3 Timeouts

The timeout values in the Request timeouts and other timeouts groups present in the repository file have already been tuned by u-blox for optimum functionality, but they can be modified according to need.

F.4 Emergency numbers

From the developer’s point of view, emergency numbers can be set in Android by assigning either the ro.ril.ecclist or ril.ecclist property, where the first one can be set at build-time only and is read-only, while the latter is dynamic and can be set/modified at runtime. When dialing emergency numbers, by default Android first checks the ril.ecclist property for the specific number, but if this property is not defined, then it checks the number in the ro.ril.ecclist property. The numbers currently present in the list are shown in Table 6. For more information on these numbers, kindly consult this Wikipedia link.

<table>
<thead>
<tr>
<th>Numbers</th>
<th>Listing</th>
</tr>
</thead>
<tbody>
<tr>
<td>111,113,115,117,122,125,127</td>
<td>Common emergency numbers used in various countries</td>
</tr>
<tr>
<td>112,911,000,08,110,999,118,119</td>
<td>ETSI standard</td>
</tr>
</tbody>
</table>

Table 6: Supported emergency numbers
G Module firmware update

☞ For further details about the module firmware update in an Android OS system, please contact the u-blox technical support.
H Multi module support in Android RIL

⚠ This section only applies to RIL version 09.01 and above.

H.1 Multi module

This method allows the u-blox RIL to switch between two modules connected on different interfaces. Only the USB and UART interfaces can be used to set up this feature.

When the module is configured in CDC-ACM or RNDIS mode, then it will be symbolized it as a USB interface. If the module is connected using a serial port or virtual COM port on a serial interface, then it is symbolized it as a UART interface.

The main purpose of this feature is to use two modules on both interfaces but activate one at a time; RIL has a functionality to switch between the two modules on a particular trigger.

H.1.1 Prerequisites

Both modules should be connected on the respective interfaces according to need. By default, it is in “ttyACM[usb]” mode. The default mode can be configured through the szInterfaceDefault variable in the function named mainloop in the rildmain.cpp file.

H.1.2 Mode switching

u-blox also provides a sample test application source for the user to trigger between both modules by setting a specific property. See appendix D to integrate the testing application in the Android source code. This application uses OEM_HOOK_API to communicate with RIL and perform specific operations.

The switching process is initialized by selecting the required interface property in the RIL OEM HOOK Test APK, e.g. Set Property (modinf=ttyACM) for the USB interface and Set Property (modinf=pts) for the UART interface.

By setting the following property switching, operations can be performed.

    persist.radio.uril.modInf

Possible values of this property are

- For CDC-ACM interface: “ttyACM”
- For UART interface with GSMMUX: “pts”

☞ Comment out the defined value of this property (persist.radio.uril.modInf) from the init.rc file to persist its desired set value.

Follow these steps to use this feature manually:

3. Set the property (persist.radio.uril.modInf) for the desired interface.
4. Power off the module by means of the AT command.
5. Cut off the power IO to stop the module power.
6. Now enable the power of the second module.
7. The RIL will automatically connect to the set interface if it is available at.

H.1.3 Switching status

After the property has been successfully set, the power of the currently active module should be cut off, which can be done by issuing the AT+CPWROFF command from the RIL OEM HOOK test APK. Then turn the power back on to the new active module.
### I Verizon network

⚠ This section only applies to RIL version 09.03 and onwards.

#### I.1 init.ublox.rc

The following service needs to be added to the service section of `init.ublox.rc` file as follows:

```bash
# APN update service for Verizon Network
service static_apn /system/bin/static_apn
user root
group radio cache inet misc
class main
disabled
oneshot
```

#### I.2 core_ublox.mk

Update the following file to copy the target directory on the device:

- `ANDROID_SOURCE/build/target/product/core_ublox.mk`

Add the following line to `PRODUCT_COPY_FILES`

```bash
$(LOCAL_PATH)/rootdir/etc/static_apn:system/bin/static_apn
```

### J Module specific configurations

This section applies to modules requiring some customized modifications in the Android source or kernel to properly communicate with the u-blox RIL driver.

#### J.1 SARA-R4

##### J.1.1 Enable USB serial drivers in the kernel

SARA-R4 series modules enumerate as a USB serial device `/dev/ttyUSB` instead of a ttyACM when connected to an Android device. USB serial driver support needs to be enabled in the kernel for a proper enumeration of the SARA-R4 module.

After a successful enumeration of SARA-R4, three ttyUSB* ports namely `ttyUSB0`, `ttyUSB1` and `ttyUSB2` will be visible in `/dev`.

The ttyUSB1 port is used for communication purposes with the u-blox RIL.
K Audio codec

⚠ LISA-C2 product series does not support external codec management by AT commands.

K.1 Configuration

This section describes a set of u-blox proprietary AT commands to be used for the audio features configuration. Customers can configure this section for custom audio features according to their platform needs.

The section below in the RIL code is divided into two main sections:

- Settings to manage external codec or other external audio IC
- Setting the audio path and configure I2S interfaces

For further details, see the u-blox AT commands manual [6].

K.2 Example

Below, a RIL code snapshot is provided to mark the location where the customer can update/add/manage their own custom audio codec.

```cpp
(ublox_ril\core\nd\Systemmanager.cpp under::InitializeModemFeatures() )

//Initialized digital audio features (I2C interface)
pCmd = new CCommand(RIL_CHANNEL_ATCMD, NULL,ND_REQ_ID_NONE, 
"AT+UMCLK=2,1;+UI2CO=1,0,0,0x10,0;+UI2CW="00000000108F20240000103300250000008A",18;+UI2CW="049E",2;+UI2CC\r", &CTE::ParseSupportedFeatures);

//Initialized digital audio features (I2S interface)
pCmd = new CCommand(RIL_CHANNEL_ATCMD, NULL,ND_REQ_ID_NONE, 
"AT+USPM=1,1,0,0,2;+UI2S=1,1,0,3,0\r", &CTE::ParseSupportedFeatures);
```

L i MX platform specific notes

Consider the following points during the integration on the i.MX 6 platform.

See section B for details about changes required in the following file.

L.1 imx6.mk

Open the imx6.mk file for the path device/fsl/imx6/imx6.mk and apply the following changes.

```
#Remove following lines from imx6.mk file
- chat
- ip-up-vpn
- ip-up-ppp0
- ip-down-ppp0
- device/fsl/imx6/etc/ppp/init.gprs-pppd:system/etc/ppp/init.gprs-pppd 
```

L.2 apn-config.xml

Place the apns-config.xml file at /device/fsl/imx6/ for any possible data connectivity issues.
M  Android RIL integration FAQ

Q: RIL cannot open the COM port to connect ttyACMX or ttyUSBX.
A: Make the following verification checks if the COM port is not opened:

- In the Android platform (while the u-blox module is connected), use adbshell and verify that the ttyACMX or ttyUSBX ports are present in the /dev path. If these are not present, see appendix A.
- Make sure that the necessary permissions (ttyACMx, ttyUSBx) are added in the uevent.platform.rc file.
- Verify the SE Policies for the required port, see appendix B.6.

Q: All the scripts are not working or RIL is not working after integration in Android.
A: It is recommended that the RIL package is downloaded and opened on the Linux machine. If the zip file is opened on a Windows machine, it may result in script corruption because of illegal characters (^M).

Q: How to enable different log levels in RIL?
A: In the repository.txt file, the “LogLevel” field can be used to set the log level. The default value is set to “2=info logs”. The following values are available:

- 4: critical
- 3: warning
- 2: info
- 1: verbose

Q: How to shift from PPP mode to RNDIS mode or vice versa?
A: u-blox RIL supports both PPP and RNDIS modes, and this can be selected from the repository.txt file.

- For PPP mode, the setting is “USBCONF=0”.
- For RNDIS (router) mode, the settings are “USBCONF=3” and “UBMCONF=1”.

Q: Data in PPP mode is not working after integrating the u-blox RIL.
A: In PPP mode, check that the pppd_data service is running properly. If not, check the permissions and SE Policy for this script. The pppd_data service starts the PPP daemon and sets many properties, so it should be verified that the necessary SE Policies for pppd should be set.

Also verify that the “chat” script is added and running properly in pppd.

Q: Data in RNDIS is not working after integrating u-blox RIL.
A: USB0 is used to connect data in RNDIS mode so it should be verified that ttyUSB0 is used for the u-blox module. The “netd_ena” service should also be running properly.

Q: How to increase log buffer size?
A: There is a setting in the “Log buffer sizes” developer option that can be set from 256 k to 16 M. If this setting is not present in the developer options, then an alternative way to change the buffer size is to set following property at the start of the “boot” section in the init.rc file.

setprop persist.logd.size 16777216

Q: How do you go back to 4G as there is no option in the preferred network type?
A: This can be done by dialing “*#*#4636#*#*” on an Android device and a menu would pop up. Select the phone information menu option and then it is possible to set the preferred network type from there.
Q: How to get the MDN info displayed in Android?
A: The MDN is saved in a special memory area by the service provider, with the name "ON": Own number phonebook (read/write) and the content can be shown by means of +CNUM. Sometimes it is written/saved/available by the SIM provider and sometimes not. So the +CNUM AT command can read this number (if it is available).

Q: Why gsm0710muxd is used?
A: gsm0710muxd is used for serial communications with the GSM modem. It uses the AT+CMUX command to enable the multiplexing protocol control to make the various channels available to RIL. These channels are virtually mapped with the /pts channel of gsm0710muxd and connect RIL with the /pts channel.

Q: How to check the properties set in RIL?
A: The properties set in RIL can be checked in the adb shell using the "getprop" command, and any property can be set manually using the "setprop" command.

Q: pppd exits with error code 17; how to resolve this error?
A: This error happens when the ppp server starts up too slowly. When the ppp daemon sends the link configuring packets, they get bounced from the remote modem, thereby giving a serial loopback error. To resolve this, in the gprs script, build/target/product/rootdir/etc/ppp/peers, add the following line.
   "lcp-max-configure 50"

Q: How to set the default preferred network settings in Android?
A: To set the default preferred network in Android, set the ro.telephony.default.network property in device.mk to the desired network settings. For example,
   ro.telephony.default.network=10 /* LTE, CDMA, EvDo, GSM/WCDMA */

Recompile the source after setting the property and verifying the change from the build.prop file in the out directory.

Q: How to enable adb logging using Wi-Fi in RIL?
A: Add the following line to the boot section of the init.rc file, that is:
   setprop service.adb.tcp.port 5555

Q: How to enable the USB serial driver (e.g. FTDI) in the Android kernel?
A: It can be configured by enabling the USB serial driver in the USB section using the “make menuconfig” command. Alternatively, it can also be enabled by adding the following lines into the defconfig file of the respective hardware.

   CONFIG_USB_SERIAL=y
   CONFIG_USB_SERIAL_CONSOLE=y
   CONFIG_USB_SERIAL_FTDI_SIO=y
   CONFIG_USB_SERIAL_QUALCOMM=y
Related documents

[1] u-blox SPI interface application note, doc. no. UBX-13001919
[5] u-blox multiplexer implementation application note, doc. no. UBX-13001887

☞ For regular updates to u-blox documentation and to receive product change notifications, register on our homepage (www.u-blox.com).
## Revision history

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>11-Jul-2011</td>
<td>fpic</td>
<td>Initial release</td>
</tr>
<tr>
<td>1</td>
<td>05-Sep-2011</td>
<td>fpic</td>
<td>Document aligned to RIL delivery 01.023</td>
</tr>
<tr>
<td>2</td>
<td>23-Sep-2011</td>
<td>lpah</td>
<td>Added chapter Appendix 1: Module connection</td>
</tr>
<tr>
<td>3</td>
<td>26-Jan-2012</td>
<td>lpah</td>
<td>Extended to include LISA-U2 series</td>
</tr>
<tr>
<td>4</td>
<td>26-Mar-2012</td>
<td>fpic</td>
<td>Extended to include Android 4.0 delivery</td>
</tr>
<tr>
<td>5</td>
<td>21-Jun-2012</td>
<td>fpic</td>
<td>Extended to include LEON-G100 and LEON-G200 series</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Added compatibility matrix for Android software deliveries and supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>interface by u-blox wireless modules</td>
</tr>
<tr>
<td>6</td>
<td>20-Nov-2012</td>
<td>fpic / lpah</td>
<td>Android 4.1 supported (Last revision with docu number 3G.G2-CS-11003)1</td>
</tr>
<tr>
<td>A</td>
<td>30-Aug-2012</td>
<td>fpic</td>
<td>Extended to include SARA-G350 series</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Insert minor changes and support for 2G modules into Android 4.113</td>
</tr>
<tr>
<td>R08</td>
<td>08-Apr-2014</td>
<td>fpic</td>
<td>Android 4.2 and 4.3 supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Added note on power off handling</td>
</tr>
<tr>
<td>R09</td>
<td>31-Jul-2014</td>
<td>fpic</td>
<td>Android 4.4 supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Extended the document applicability to SARA-U2 and TOBY-L2 series</td>
</tr>
<tr>
<td>R10</td>
<td>24-Oct-2014</td>
<td>fpic</td>
<td>Included modification for latest TOBY-L2 delivery</td>
</tr>
<tr>
<td>R11</td>
<td>11-Sep-2015</td>
<td>yasi</td>
<td>PPP support for TOBY-L2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Android 4.4 supported with MUX interface for SARA-G310</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Android 5.0 supported</td>
</tr>
<tr>
<td>R12</td>
<td>05-Feb-2016</td>
<td>msin</td>
<td>Extended to RIL version 08.01</td>
</tr>
<tr>
<td>R13</td>
<td>03-May-2016</td>
<td>bqam / bahm</td>
<td>Extended to RIL version 09.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Android 6.0 supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Firmware update over AT (FOAT) section added for Android 2.3 (SARA-U270-0GS)</td>
</tr>
<tr>
<td>R14</td>
<td>31-Mar-2017</td>
<td>bkha / fdil</td>
<td>Extended to RIL version 09.02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Extended the document applicability to TOBY-R2 and LARA-R2 series</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Updated TOBY-L2 series profile setting section</td>
</tr>
<tr>
<td>R15</td>
<td>28-Jul-2017</td>
<td>bahm / yasi</td>
<td>Android 7.0 supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Added section about Verizon network settings</td>
</tr>
<tr>
<td>R16</td>
<td>30-Aug-2017</td>
<td>uafz</td>
<td>Added section about module specific modifications</td>
</tr>
<tr>
<td>R17</td>
<td>10-Oct-2017</td>
<td>bkha</td>
<td>Updated the init.rc configuration</td>
</tr>
<tr>
<td>R18</td>
<td>27-Nov-2017</td>
<td>bkha</td>
<td>Updated multi module and OEM Hook App sections</td>
</tr>
<tr>
<td>R19</td>
<td>17-Jan-2018</td>
<td>bkha</td>
<td>Android Oreo support and init.ublox.rc, Core_ublox.rc file addition.</td>
</tr>
<tr>
<td>R20</td>
<td>16-May-2018</td>
<td>bkha</td>
<td>Extended document applicability to TOBY-L4 series.</td>
</tr>
<tr>
<td>R21</td>
<td>24-Oct-2018</td>
<td>bkha / fdil</td>
<td>Added section regarding the integration on i.MX 6 platform.</td>
</tr>
<tr>
<td>R22</td>
<td>21-Dec-2018</td>
<td>iazi</td>
<td>Added sections regarding Android 9.0 Pie – New supporting Android version</td>
</tr>
<tr>
<td>R23</td>
<td>15-Apr-2018</td>
<td>bkha</td>
<td>Added support for RIL v12.01</td>
</tr>
<tr>
<td>R24</td>
<td>06-Sep-2019</td>
<td>bkha</td>
<td>i-MX 8 Support, remove android 4.0 section, RILv12.03</td>
</tr>
</tbody>
</table>
## Contact

For complete contact information, visit us at [www.u-blox.com](http://www.u-blox.com).

### u-blox Offices

**North, Central and South America**

- **u-blox America, Inc.**
  - Phone: +1 703 483 3180
  - E-mail: info_us@u-blox.com

- **Regional Office West Coast:**
  - Phone: +1 408 573 3640
  - E-mail: info_us@u-blox.com

- **Technical Support:**
  - Phone: +1 703 483 3185
  - E-mail: support@u-blox.com

**Europe, Middle East, Africa**

- **u-blox AG**
  - Phone: +41 44 722 74 44
  - E-mail: info@u-blox.com
  - Support: support@u-blox.com

**Asia, Australia, Pacific**

- **u-blox Singapore Pte. Ltd.**
  - Phone: +65 6734 3811
  - E-mail: info_ap@u-blox.com
  - Support: support_ap@u-blox.com

- **Regional Office Australia:**
  - Phone: +61 2 8448 2016
  - E-mail: info_au@u-blox.com
  - Support: support_ap@u-blox.com

- **Regional Office China (Beijing):**
  - Phone: +86 10 68 133 545
  - E-mail: info_cn@u-blox.com
  - Support: support_cn@u-blox.com

- **Regional Office China (Chongqing):**
  - Phone: +86 23 6815 1588
  - E-mail: info_cn@u-blox.com
  - Support: support_cn@u-blox.com

- **Regional Office China (Shanghai):**
  - Phone: +86 21 6090 4832
  - E-mail: info_cn@u-blox.com
  - Support: support_cn@u-blox.com

- **Regional Office China (Shenzhen):**
  - Phone: +86 755 8627 1083
  - E-mail: info_cn@u-blox.com
  - Support: support_cn@u-blox.com

- **Regional Office India:**
  - Phone: +91 80 405 092 00
  - E-mail: info_in@u-blox.com
  - Support: support_in@u-blox.com

- **Regional Office Japan (Osaka):**
  - Phone: +81 6 6941 3660
  - E-mail: info_jp@u-blox.com
  - Support: support_jp@u-blox.com

- **Regional Office Japan (Tokyo):**
  - Phone: +81 3 5775 3850
  - E-mail: info_jp@u-blox.com
  - Support: support_jp@u-blox.com

- **Regional Office Korea:**
  - Phone: +82 2 542 0861
  - E-mail: info_kr@u-blox.com
  - Support: support_kr@u-blox.com

- **Regional Office Taiwan:**
  - Phone: +886 2 2657 1090
  - E-mail: info_tw@u-blox.com
  - Support: support_tw@u-blox.com